COMPSCI 638: Graph Algorithms October 16, 2019

Lecture 15

Lecturer: Debmalya Panigrahi *Scribe: Kevin Sun*

1 Overview

In this lecture, we give an algorithm for the online Steiner tree problem. To bound its performance guarantee, we will use a technique known as dual fitting.

2 Online Steiner Tree

Recall that in the Steiner tree problem, we are given a graph $G = (V, E)$ where each edge e has weight $c(e) \geq 0$, as well as a set of terminals $T \subseteq V$. Our goal was to find a minimum-weight subset of edges F such that the graph (V, F) contains all of the terminals T in one connected component.

In the online Steiner tree problem, we are given the same edge-weighted graph $G = (V, E)$. However, now set of k terminals T is revealed one at a time. At each time step i, the algorithm is shown a terminal t_i and must add a subset of edges to connect t_i with the terminals t_j for $j < i$. As in the offline setting, the goal is to minimize the total cost of edges in the solution. Furthermore, the algorithm is not permitted to remove any edges from the solution once they are added.

The *competitive ratio* of an algorithm is essentially the "approximation ratio": it is the worst-case ratio of the cost of the solution found by the algorithm to the cost of the optimal (offline) solution. If the competitive ratio of an algorithm is at most c, then we say that algorithm is *c-competitive*.

2.1 The Greedy Algorithm

A natural algorithm for the online Steiner tree problem is the following: when terminal t_i arrives, the algorithm simply adds the edges of the shortest path from t_i to $\{t_1, t_2, \ldots, t_{i-1}\}$.

Theorem 1. *The greedy algorithm is $O(\log k)$-competitive for the online Steiner tree problem.*

Remark: This result is due to Imase and Waxman [IW91], but our proof is simpler and uses a technique known as dual fitting. Furthermore, Imase and Waxman [IW91] showed that any algorithm for the online Steiner tree must be $\Omega(\log k)$-competitive, and in fact, the constant hidden in the $O(\log k)$ competitive ratio is quite small, meaning that the lower bound is very tight.

Dual fitting: Before giving the proof, we briefly sketch the overall strategy. Our algorithm does not consider the dual at all (unlike primal-dual algorithms), but in the analysis, we will create a set of feasible dual solutions. Recall from Lecture 13 that the dual LP is the following:

$$\max \sum_{S \in \mathcal{S}} y_S$$

$$\sum_{S : e \in \delta(S)} y_S \leq c(e) \quad \forall e \in E$$

$$y_S \geq 0 \quad \forall S \in \mathcal{S},$$

where $\mathcal{S}$ denotes the subsets of V that separate at least one pair of terminals. We will bound ALG against the total cost of the dual solutions, and since the cost of each dual solution is at most OPT, this will yield our final competitive ratio.

Placing a dual ball: To construct a dual solution, we will use the notion of placing a "ball" around some terminal t. Algebraically, this procedure proceeds as follows: let $B_r(u)$ denote the ball with radius r around vertex u. Starting with the set $S = \{t\} = B_0(t)$, we increase the variable y_S until $B_0(t) \subsetneq B_{y_S}(t)$; i.e., the ball contains vertices other than t. At this point, we freeze y_S and begin increasing $y_{S'}$ where $S' = B_{y_S}(t)$. This process continues our ball has the desired radius.

To summarize, "placing a ball" centered at t with radius r corresponds to increasing multiple dual variables, each corresponding to an increasingly larger cut containing t. The radius of the ball specifies the total amount of increase across the affected dual variables.

Proof. Consider the following procedure that, as the greedy algorithm proceeds, constructs a set of feasible dual solutions. When the algorithm connects t_i to t_j for some $j < i$, let C_i denote the incurred cost, and suppose $C_i \in (2^\ell, 2^{\ell+1}]$. Then in the ℓ-th dual solution, we place a ball of radius $2^{\ell-1}$ centered at t_i. We continue this process until the greedy algorithm terminates.

Now we'll show that the ℓ-th dual solution is feasible by showing that its balls are pairwise disjoint. Note that the radius of every ball is $2^{\ell-1}$. For contradiction, assume two balls centered at $t_i \neq t_j$ overlap, where $i > j$. Then the distance from t_i to t_j is at most $2 \cdot 2^{\ell-1} = 2^\ell$ (because the balls overlap). However, since t_i arrived after t_j, one option that the greedy algorithm considered was connecting t_i to t_j incurring cost at most $2^\ell < C_i$, violating the fact that $C_i \in (2^\ell, 2^{\ell+1}]$.

Let $D(\ell)$ denote the cost of the ℓ-th dual solution, and let $T(\ell)$ denote the time steps i such that $C_i \in (2^\ell, 2^{\ell+1}]$. At any time $i \in T(\ell)$, the algorithm incurs cost $C_i \leq 2^{\ell+1} = 4 \cdot 2^{\ell-1}$ and the dual cost $D(\ell)$ increases by exactly $2^{\ell-1}$. Since the initial costs are all zero, this implies

$$\sum_{i \in T(\ell)} C_i \leq 4 \cdot D(\ell). \tag{1}$$

Finally, we shall bound $ALG = \sum_{i=1}^k C_i$ against OPT. First, notice that for every time step i, we have $C_i \leq OPT$ because the optimal Steiner tree must contain a path from t_i to every t_j for $j < i$. Now partition the C_i according to the value of OPT/k:

$$ALG = \sum_{i: C_i \leq OPT/k} C_i + \sum_{i: C_i \in (1/k, 1] \cdot OPT} C_i \leq OPT + \sum_{i: C_i \in (1/k, 1] \cdot OPT} C_i. \tag{2}$$

We will bound the final summation by applying (1). Let L denote the set of all dual solution indices that get updated at any time i such that $C_i \in (1/k, 1] \cdot OPT$. Then from (1), we see that

$$\sum_{i: C_i \in (1/k, 1] \cdot OPT} C_i \leq 4 \sum_{\ell \in L} D(\ell) \leq 4|L| \cdot OPT,$$

where the second inequality holds because every dual cost is at most OPT. Every dual solution corresponds to a particular range on the value of C_i, and the length of these ranges doubles with each dual solution. Thus, the number of ranges in $(1/k, 1] \cdot OPT$, i.e., the value of $|L|$, is at most $\log_2 k$. Combining this with (2) proves that the competitive ratio of our algorithm is $O(\log k)$. $\square$

3 Summary

In this lecture, we proved that the greedy algorithm for the online Steiner tree problem is $O(\log k)$-competitive. The proof constructs a set of feasible dual solutions, bounds the cost of the algorithm against the sum of their costs, and finally bounds the number of dual solutions.

References

[IW91] Makoto Imase and Bernard M Waxman. Dynamic steiner tree problem. *SIAM Journal on Discrete Mathematics*, 4(3):369–384, 1991.