COMPSCI 590.1: Graph Algorithms	March 22, 2015

Lecture 16-17

Lecturer: Debmalya Panigrahi *Scribe: Yuhao Hu*

1 Overview

The main topics of this lecture are the *offline edge-weighted Steiner tree and Steiner forest problems*. The former generalizes the *minimum spanning tree problem* by requiring connectivity of a subset of all the vertices, called *terminals*, rather than the entire vertex set. The latter further generalizes the former problem by requiring connectivity between pairs of vertices.

For the Steiner tree problem, we introduce an algorithm which uses the idea of shortest paths and minimum spanning trees [AKR95]. This algorithm gives a 2-approximation to the optimal solution. For the Steiner forest problem, a *primal-dual* algorithm is given. The approximation is also 2OPT [GW95].

In this lecture, for simplicity, "the Steiner tree(forest) problem" would always mean "the offline edge-weighted Steiner tree(forest) problem" unless otherwise stated.

2 The Steiner Tree Problem

Let $G = (V, E)$ be a connected, edge-weighted, undirected graph. Given $R \subseteq V$, called the *terminals*, the *offline edge-weighted Steiner tree problem* asks one to find a cost-minimizing connected subgraph of G which contains all the terminals. Such a solution is of course a tree.

In another setting, one may consider G to be node-weighted instead. Correspondingly, we have the *offline node-weighted Steiner tree problem*. It is an easy observation that, in this problem, one could assume all the terminals to have weight zero, as they must be included in any solution by the connectivity requirement. Non-terminal vertices are called *Steiner vertices*.

As is mentioned in the overview, currently, we are only concerned with the edge-weighted problems.

2.1 Steiner tree: a 2-approximation algorithm

Let us assume that an optimal solution T is obtained for the Steiner tree problem. An immediate observation is that all the leaves in the tree T must be terminals. Otherwise, one could simply delete the non-terminal leaves, yielding a feasible solution with less cost.

Now, look at the graph in Figure 1, which suggests an optimal solution to some Steiner tree problem. A DFS on this tree gives rise to a cycle(in blue) which visits each terminal exactly twice. Note that this cycle can be decomposed into paths between terminals adjacent in the DFS. Fix a pair of such adjacent terminals, and consider the shortest path(in orange) between them. The cost of the shortest path is of course no more than the cost of the path in the optimal tree.

The argument above inspires us to consider the following: let $\tilde{G}$ be a graph whose vertices are the terminals in G; for each pair of terminals r_1 and r_2, add an edge (r_1, r_2) to $\tilde{G}$ with weight equaling the

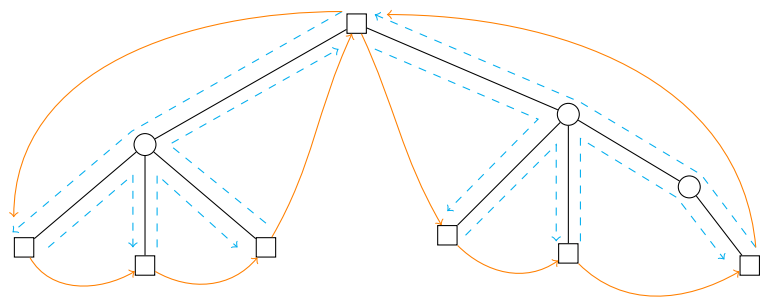

Figure 1: Idea behind the algorithm: 2-approximation to the Steiner tree problem.

distance $d(r_1, r_2)$. Since the edges in $\tilde{G}$ represent paths between terminals in G, we have the relations:

$$\begin{aligned} 2\text{OPT} &= \text{cost of the cycle on terminals in the OPT tree} \\ &\geq \text{cost of some spanning tree of } \tilde{G} \\ &\geq \text{cost of the MST of } \tilde{G}. \end{aligned}$$

The algorithm is summarized as follows.

2-approximation algorithm for Steiner tree
• For every pair of terminals $r1$ and r_2, add an edge of cost equaling the distance between r_1 and r_2;
• Find an MST on the terminals. Recover the (shortest) paths which represent the added edges. Delete edges, if necessary, to result in a tree for output.

3 The Steiner Forest Problem

Let $G = (V, E)$ be as in the description of the Steiner tree problem. Instead of having a subset $R \subseteq V$ as terminals, we consider a set of terminal pairs

$$\{(s_i, t_i) : s_i, t_i \in V\}.$$

The *Steiner forest problem* then asks one to find a cost-minimizing subgraph of G which connects each (s_i, t_i) pair. Note that this generalizes the Steiner tree problem. In fact, it is quite obvious that the Steiner tree problem with the terminal set

$$R = \{v_1, \ldots, v_k\} \subseteq V$$

is equivalent to the Steiner forest problem with the terminal pairs

$$\{(v_1, v_i)\}_{i=2}^{k}.$$

3.1 A primal-dual algorithm

In this subsection, we introduce a primal-dual algorithm which gives a 2-approximation for the Steiner forest problem. What is unusual is that the linear optimization is only used to prove the approximation ratio, but not to give a solution to the problem. The primal and the dual LP are described below. Warning: one might have exponentially many variables in the dual LP.

Primal LP

$$X_e : \text{whether edge } e \text{ is chosen or not.}$$

$$\text{minimize} : \sum_{e \in E} c_e X_e.$$

$$\sum_{e \in (S,\bar{S})} X_e \geq 1, \text{for all } S \text{ separating some } (s_i, t_i) \text{ pair.}$$

$$X_e \geq 0.$$

Dual LP

y_S: defined for each cut S separating some (s_i, t_i) pair.

$$\text{maximize} : \sum_{\substack{S \text{ separating} \\ \text{some } (s_i,t_i) \\ \text{pair}}} y_S.$$

$$\sum_{S: e \in (S,\bar{S})} y_S \leq c_e.$$

$$y_S \geq 0.$$

Naturally, one may now wonder how to combinatorially understand the dual LP. A simple case is when there is only one pair of terminals connected by an edge e, and the cuts are around the (terminal) singleton vertices (Figure 2). Now, view y_S as the radii of the balls centering at all S and imagine that the radii of the balls are growing in the same rate. If a new vertex enters the region of some growing ball, the corresponding y_S stops growing since the cut S has changed. Otherwise, the constraints in the dual LP is simply that the (growing) balls cannot overlap along e. In other words, in this particular case,

$$\sum_S y_S \leq \text{cost of the Steiner forest.} \tag{1}$$

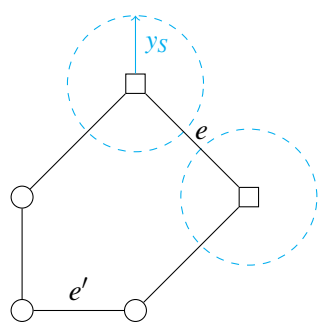

Figure 2: Combinatorial interpretation of the dual LP: a simple case.

To generalize a bit, let us allow the growing balls to cross through some new vertex before overlapping each other, as illustrated in Figure 3 below. In this particular case, $y_{\{v_2\}}$ attains its maximum possible value when the ball centered at v_2 grows to a radius equaling $c_{(u,v_2)}$. Once the radii of the balls exceed $c_{(u,v_2)}$, it is no surprise that the variable $y_{\{v_2\}}$ becomes inactive and it is the values of $y_{\{v_2,u\}}$ and $y_{\{v_1\}}$ that are increasing.

Noting these, it may still be worthwhile to see how the constraints in the dual LP are reflected in the graph. Take the edge $e = (v_1, v_2)$ as an example. Initially, among all cuts which separate v_1, v_2 and contain

e, only the degree cuts at v_1 and v_2 have active associated variables. By the time that $y_{\{v_2\}}$ equals $c_{(v_2,u)}$, the sum of $y_{\{v_1\}}$ and $y_{\{v_2\}}$ is clearly less than c_e. Once the ball centering at v_2 passes u, $y_{\{v_2\}}$ stops increasing and $v_{\{v_2,u\}}$ starts increasing from zero. From the point of view of e, the constraint now becomes

$$y_{\{v_1\}} + y_{\{v_2,u\}} + c_{(u,v_2)} \leq c_e, \tag{2}$$

which matches exactly the condition that the two growing balls centered at v_1 and v_2 do not overlap along e.

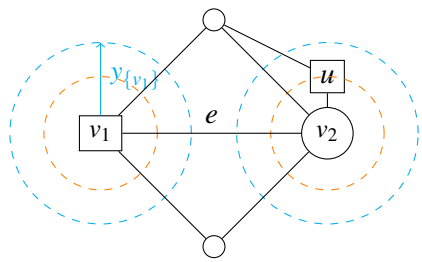

Figure 3: Combinatorial interpretation of the dual LP: generalizing a bit.

Now we are ready to consider the case when the two growing balls, for the first time, collide with each other along some edge (x,y) (Figure 4). It is easy to see that this collision corresponds to the shortest path between the pair of terminals s,t. Moreover, the dual LP constraints at all edges along this path are saturated.

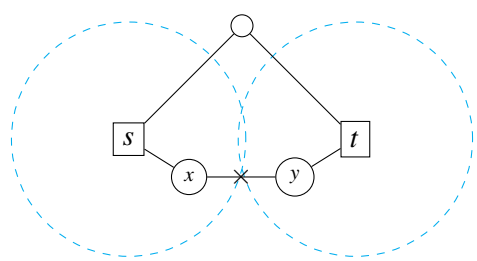

Figure 4: Combinatorial interpretation of the dual LP: collision between balls.

In the general setting, it is likely that more than one pair of terminals are under consideration and more than two dual variables are active at a given stage. Once two balls collide along an edge, the respective dual LP variables become inactive. In this case, one could still view the union of the two balls as defining a new cut. This corresponds to a new dual LP variable $y_{S_1 \cup S_2}$, where y_{S_1} and y_{S_2} are the variables associated to the two balls(cuts) right before the collision. This process, combined with all the steps discussed above, gives rise to an algorithm which runs until there are no active dual LP variable remaining (a dual variable y_S is considered inactive if either some edge in $(S,\bar{S})$ is saturated, or $(S,\bar{S})$ does not separate any pair of terminals). Intuitively, this is illustrated in Figure 5. The dual LP variables active at each time are recorded in Table 1.

Suppose now that the procedure previously described ends. Let F denote the set of edges whose (dual) constraints are saturated. On the one hand, note that F is certainly a feasible solution to the Steiner forest problem, except that the approximation ratio to the OPT is unclear. Otherwise, some (s_i,t_i) pair is separated thus there still exist active dual variables. On the other hand, F could contain many more edges than needed. For example, in Figure 6, F contains all the edges in the graph, while the OPT solution contains only one edge! Thus, as the final step of the algorithm, we order the set F increasingly in the time that an edge become

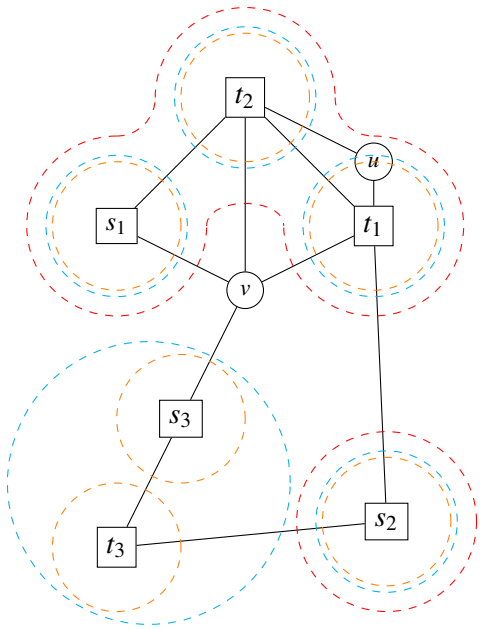

Figure 5: Combinatorial interpretation of the dual LP: the general case.

Stage	Active Dual Variables
1	$y_{\{s_1\}}, y_{\{t_1\}}, y_{\{s_2\}}, y_{\{t_2\}}, y_{\{s_3\}}, y_{\{t_3\}}$
2	$y_{\{s_1\}}, y_{\{t_1,u\}}, y_{\{s_2\}}, y_{\{t_2\}}, y_{\{s_3\}}, y_{\{t_3\}}$
3	$y_{\{s_1\}}, y_{\{t_1,u\}}, y_{\{s_2\}}, y_{\{t_2\}}, y_{\{s_3,t_3\}}$
4	$y_{\{t_1,t_2,s_1,u\}}, y_{\{s_2\}}, y_{\{s_3,t_3\}}$
5	$y_{\{t_1,t_2,s_1,u\}}, y_{\{s_3,t_3,s_2\}}$
6	$y_{\{t_1,t_2,s_1,u,v\}}, y_{\{s_3,t_3,s_2\}}$
7	$y_{\{t_1,t_2,s_1,s_3,t_3,s_2,u,v\}}$

Table 1: Active dual LP variables in Figure 5.

saturated. Starting from the last moment and backwards in time, discard e from F' if $F' \setminus \{e\}$ preserves the feasibility of the solution, until no such deletion is possible. No surprise that, in the end, F' is a forest. These final steps are called the *reverse delete*.

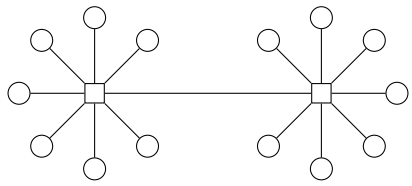

Figure 6: Inefficiency of buying all the edges in the end.

3.2 The approximation ratio analysis

Let y_S (S separating some (s_i, t_i) pair) be the values of the dual variables at the end of the augmenting procedure. And let F' be the forest selected after the reverse delete step. The following equalities hold.

$$\text{OPT} \geq \sum_S y_S, \qquad \text{(dual solution)} \tag{3}$$

$$\sum_{e \in F'} c_e = \sum_{e \in F'} \sum_{S: e \in S} y_S \qquad \text{(because each } e \in F' \text{ is saturated)} \tag{4}$$

$$= \sum_S y_S \cdot \deg_{F'}(S), \qquad \text{(primal solution)} \tag{5}$$

where $\deg_{F'}(S)$ equals the number of edges in F' which also lies in the cut $(S, \bar{S})$. Next, we show that, in fact,

$$\sum_S y_s \cdot \deg_{F'}(S) \leq 2 \sum_S y_S, \tag{6}$$

thus proving that *the algorithm in the previous subsection is a 2-approximation to the optimal solution*.

The idea of the proof is fixing the forest F' and run the augmenting procedure on G from the beginning in ε-small steps. Note that initially, all y_S have value zero. Thus, to prove the fact above, it suffices to show that in each ε-small step of augmentation, the differences

$$\Delta P = \varepsilon \cdot \sum_{\substack{D:\ active \\ dual}} \deg_{F'}(D) \tag{7}$$

and

$$\Delta D = \varepsilon \cdot \{\text{\# of active dual variables}\} =: \varepsilon \cdot n_A \tag{8}$$

satisfy

$$\Delta P \leq 2 \Delta D. \tag{9}$$

This follows from the lemma below.

Lemma 1. *At any particular stage during the augmenting procedure, let F_0' be the forest obtained from F' after contracting each dual-defining subgraph of G. Then, F_0' is a forest and each leaf in the forest F_0' is an active dual.*

Before proving the lemma, we show that, assuming the lemma, the inequality (9) holds. Take a tree T in F'. Imagine that the augmentation procedure runs on this tree. When two variables merge, simply union and contract the underlying subgraphs. This yields a tree T' whose nodes represent dual variables (Figure 7). We claim that, in this tree, the average degree of the active nodes is no more than 2. In fact, on the one hand, by lemma 1, all the inactive nodes must have degree greater than or equal to 2. On the other hand, the average degree of among all the nodes in a tree is no more than 2. Thus, excluding the inactive nodes would give only a smaller average. Therefore, we have

$$\sum_D \deg_T(D) \leq 2 \cdot \{\text{\# of active nodes in the tree } T'\} \leq 2 \cdot \{\text{\# of active dual variables in the original graph}\}. \tag{10}$$

The inequality (9) follows immediately.

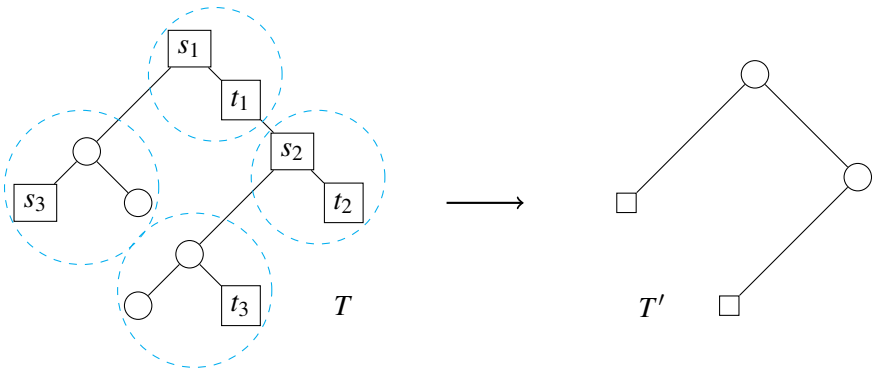

Figure 7: Tree whose nodes are duals

Proof. (of Lemma 1) Suppose that, after some augmentation of the dual variables, an edge e is saturated, creating a cycle in F_0'. This means that e is saturated earlier than the existing edges in F_0'. Since deleting any one of the existing edges will not change the feasibility of the solution, by the reverse delete steps, one of the edges must be deleted before e, which is a contradiction. Therefore, F_0' are forests.

Suppose that some leaf v of T' is inactive. This means that v corresponds to a cut in the original graph which does not separate any pair of terminals. Thus, the solution remains feasible with the edge in the leaf deleted. The reverse delete step then implies that v cannot be a node of the tree T', which is a contradiction. □

4 Summary

In this lecture, we introduced the offline edge-weighted Steiner tree/forest problems. In particular, we used the idea of shortest path and minimum spanning tree to obtain a 2-approximation to the Steiner tree problem. Moreover, a primal-dual algorithm is given to obtain a 2-approximation algorithm for the Steiner forest problem. In the upcoming lectures, we will study algorithms for online Steiner tree/forest and node-weighted Steiner tree/forest problems.

References

[AKR95] Ajit Agrawal, Philip Klein, and R Ravi. When trees collide: An approximation algorithm for the generalized steiner problem on networks. *SIAM Journal on Computing*, 24(3):440–456, 1995.

[GW95] Michel X Goemans and David P Williamson. A general approximation technique for constrained forest problems. *SIAM Journal on Computing*, 24(2):296–317, 1995.