CS880: Approximation and Online Algorithms	**Scribe:** Luke Swanson
Lecture 10: Steiner Forest Problem (listed as Metric Embeddings & Multicut)	**Date:** 10/7/2019

10.1 Steiner Forest Problem

Consider the graph $G = (V, E)$, with edge costs c_e and k vertex pairs, denoted by (s_i, t_i).
The Steiner Forest Problem asks us to find a subgraph $F \subseteq E$ of least cost, where $\forall i$, (s_i, t_i) are connected in the graph (V, F).

10.2 Additional Definitions

We define some terms and notation which will be useful in our analysis:

Definition 10.2.1 *Let* $\mathcal{S} = \{S \subseteq V : \exists i$ *where* $|S \cap \{s_i, t_i\}| = 1\}$

Informally, we can interpret $\mathcal{S}$ as the set of vertex subsets created by cuts splitting a (s_i, t_i) pair.

Definition 10.2.2 *Let* $S \in \mathcal{S}$*. Then* $\delta(S) = \{(u, v) \in E : |\{u, v\} \cap S| = 1\}$

Informally, we can interpret $\delta(S)$ as the set of edges with one endpoint in S and one in $V \setminus S$.

10.3 Primal-Dual Algorithm

Our LP for the Steiner Tree Problem is as follows:
Primal:

$$\min \sum_e c_e \cdot x_e \tag{10.3.1}$$

$$\sum_{e \in \delta(S)} x_e \geq 1 \tag{10.3.2}$$

$$x_e \geq 0 \ \forall e \tag{10.3.3}$$

Dual:

$$\max \sum_{s \in \mathcal{S}} y_s \tag{10.3.4}$$

$$\sum_{S \in \mathcal{S}: e \in \delta(S)} y_s \leq c_e \forall e \tag{10.3.5}$$

$$y_s \geq 0 \tag{10.3.6}$$

Interpreting the LP above, we minimize the cost of edges e included in solution F (where edges may be partially included, e.g. $x = 0.4$). For a given cut S, the sum of edge inclusions among edges cut by S must be at least 1 (i.e. at least on edge is cut). Edges can not have negative inclusion.
In the case of the dual, we are maximizing the amount spent on each cut, where the total spent on cuts crossed by a given edge e can be no more than c_e. Again, we can not spend a negative amount on any cut.

Applying our template for Primal-Dual algorithms, we have the following:
`- Start with` x = 0, y = 0
`- Raise some unsatisfied` y_s `variables until some edge becomes tight (that is,` $\sum y_s = c_e$ `)`
`- Pick any tight edge e, and set` $x_e = 1$
`- Freeze` $y_s \; \forall s$ `where` $\delta(s) \ni e$ `(that is, all cuts that e crosses)`
`- Repeat until all s are satisfied`

The invariants of our algorithm are:
- y is feasible
- $\forall e$, $x_e = 0$ or $\sum\limits_{s: e \in \delta(s)} y_s = c_e$

Now, initially $y = 0$, and y_s are only ever increased. Whenever an edge e becomes tight in algorithm step 2 (i.e. $\sum\limits_{s: e \in \delta(s)} y_s = c_e$), we freeze all y_s crossed by e.
Hence, y is always feasible and the first invariant holds.
Further, $x = 0$ initially, and the value x_e is only changed when an edge becomes tight.
As noted, this happens when $\sum\limits_{s: e \in \delta(s)} y_s = c_e$, after which we freeze y_s for all S crossed by e, so the second invariant holds.

We note that x_e is only raised when e is tight, hence

$$\sum_e x_e c_e = \sum_e x_e \Big(\sum_{s: e \in \delta(s)} y_s \Big) = \sum_{s \in \mathcal{S}} y_s \Big(\sum_{e \in \delta(s)} x_e \Big)$$

.
This expression gives us the cost paid by our algorithm. We must show the cost is not "much larger" than the dual's objective function.
For example, consider a case where $\forall S \in \mathcal{S}$ with $y_s > 0$, $\sum\limits_{e \in \delta(S)} x_e \leq 2$.
Then $\sum\limits_e x_e c_e \leq 2 \sum\limits_{e \in \delta(s)} y_s = 2 \cdot \text{(dual-cost)} \implies$ 2-approximation.

In general, we have $\sum\limits_{e \in \delta(s)} x_e = \deg_f(S) = |F \cap \delta(S)|$. We want to show this value is always small.
Note that there are exponentially many y_s, so we must determine a good way to choose which y_s to raise.

10.4 Star Graph Example and Algorithm Modifications

Consider a star graph, with a central vertex s and k outer vertices $t_1, t_2, \ldots, t_k$ adjacent to s (with respect to the Steiner forest pairs (s_i, t_i), we have $s = s_1 = s_2 = \ldots = s_k$).
Suppose in algorithm step 2 we raise y_s corresponding to a cut around vertex s to $y_s = 1$. Then all edges cut (which in this case is all edges in the graph) get $x_e = 1$ in step 3 of the algorithm. Now, the primal cost has correct value of k since k edges have $x_e = 1$, but the dual cost is just 1, since $y_s = 1$ but $y_{t_1}, y_{t_2}, \ldots, y_{t_k}$ are all still 0. Thus, the gap is k. On the other hand, suppose we raised $y_{t_1}, y_{t_2}, \ldots, y_{t_k}$ simultaneously until all were equal to 1.
Then the primal cost would still be k, because all k edges are cut, but the dual would rise to k, since we now have k y-variables set to 1, and $y_s = 0$.
Then the gap is reduced to 1, a preferable result.

Our goal is to bound the gap between the dual cost and the cost paid by our algorithm, and given the difference in gaps between the examples above, it is clear the selection of y-variables in step 2 is an important consideration.

Thus, we modify the algorithm above as follows:
In step 2, `Raise, at uniform rate,` y_s `corresponding to all minimal unsatisfied S.`

Note that, if we have some connected component formed from edges picked in the algorithm, then the corresponding S is satisfied, as are all subsets of S. Only supersets of S may remain unsatisfied. Hence, we always raise the smallest unsatisfied S, and as they become satisfied, we work "upwards" to the larger unsatisfied S.

In the star graph example, the first set of minimal subsets are just the sets containing individual vertices. Each edge in the graph connects some t_i to s, so all edges will simultaneously become tight at $y_s = y_{t_1} = y_{t_2} = \ldots = y_{t_k} = 0.5$.
Again, the primal is k because all edges are taken, but now the dual is $\frac{k+1}{2}$, since $k+1$ y-variables are set to 0.5. This is a gap of approximately 2, which seems reasonable.

However, suppose we had a complete graph, rather than a star graph, with the same $s, t_1, t_2, \ldots t_k$. In this case, we would have arrived at the same dual value of $\frac{k+1}{2}$, but a primal of $\frac{k(k+1)}{2}$, i.e. all edges in the graph are selected. In this case, we are back to a gap of k between the primal and dual.

To deal with this case, we must make an additional modification to the algorithm. We now add a sixth step to the algorithm, called a "reverse delete" step:

```
- Consider edges e in reverse order of inclusion.
  Remove e if connectivity is still satisfied after removal of e.
```

Note, this is not equivalent to greedily removing edges by weight, since the heaviest edges are not necessarily added last.

10.5 Analysis

Recall, we showed the primal cost $\sum\limits_{e} x_e c_e = \sum\limits_{s\in\mathcal{S}} y_s(\sum\limits_{e\in\delta(s)} x_e) = \sum\limits_{s\in\mathcal{S}} y_s \deg_f(S)$.

We shall prove the following lemma:

Lemma 10.5.1 *Consider any algorithm iteration, and let $\mathcal{S}'$ be the collection of unsatisfied sets at the iteration. Then* $\sum\limits_{s\in\mathcal{S}'} \deg_f(S) \leq 2|\mathcal{S}'|$.

If we prove this lemma, it follows that for any iteration, the increase in primal cost per iteration is just $\sum\limits_{s\in\mathcal{S}'} \Delta y_s \deg_F(S) = \epsilon \sum\limits_{s\in\mathcal{S}} \deg_F(S) \leq \epsilon \cdot (2|\mathcal{S}'|) = 2 \sum\limits_{s\in\mathcal{S}} |\Delta y_s| = 2 \cdot$ increase in dual cost, where $\epsilon = \Delta y_s$. Then the total primal cost is at most $2 \cdot$ dual cost. This implies a 2-approximation, as noted in 10.3.

Hence, we have only to show that the lemma holds.

Now, the lemma says that the count of edges of F for any minimum unsatisfied S is no more than twice the number of minimum unsatisfied sets. The change in primal cost is the change in dual times edges crossed, so the change in primal can not be more than twice the change in dual.

For any iteration, as in Lemma 10.5.1, consider a graph G' where each connected component of F is collected into a "meta-node." The edges left in G' then represent edges in unsatisfied $S \in \mathcal{S}'$, since any satisfied S would be contracted. Alternatively, we say all leaf nodes in G' must have some unsatisfied subset.

Now, consider any $e \in F \cap \delta(S)$, for some $S \in \mathcal{S}'$.

The newly created component in $F \setminus \{e\}$ when adding e is an unsatisfied component S, which has some subset in $\mathcal{S}'$.

Then we must show that the number of edges of unsatisfied S in F is no more than twice the number of unsatisfied S, i.e. the average $\deg_F(S) \leq 2$.