CMPUT 675: Approximation Algorithms **Fall 2014**

Lecture 18 (Oct 17): Steiner Forest and k-Median

Lecturer: Zachary Friggstad *Scribe: Nikos Fasarakis-Hilliard*

18.1 Steiner Forest Generalizations

We continue our discussion of Steiner Forest in a more general context.

Let $f : 2^V \to \{0, 1\}$ be a function that satisfies the following properties:

1. $f(\emptyset) = f(V) = 0$.
2. $f(S) = f(V - S),$ for all $\emptyset \subseteq S \subseteq V$.
3. $f(S \cup T) \leq \max\{f(S), f(T)\}$ for any two disjoint sets $S, T \subseteq V$.

Call such a function *proper*. For example, the function $f(.)$ from last lecture for the Steiner Forest problem is easily seen to be proper. So is the function f with $f(S) = |S| \pmod 2$ when $|V|$ is even. More generally, if b is an integer such that $|V|$ is a multiple of b, then the function f with $f(S) = 1$ if and only if b does not divide $|S|$ is proper.

In Lecture 16, we made the following claim (using the notation from the algorithm description in Lecture 16).

Claim 1 *Consider any iteration i and let F be the final set of returned edges. We have $\sum_{S \in \mathcal{C}_i} |F \cap \delta(S)| \leq 2|\mathcal{C}_i|$, i.e. the average degree of the "active" sets in iteration i is at most 2.*

In fact, the algorithm from Lecture 16 can be executed if f is a proper function where, instead of being explicitly supplied in the input, we are able to compute $f(S)$ efficiently for any $S \subseteq V$. Essentially the only thing that needs to be proven to establish correctness and efficiency of the algorithm is the following.

Lemma 1 *If f is a proper function and $F \subseteq E$ is not feasible, then the* minimal *sets S with $\delta(S) \cap F = \emptyset$ and $f(S) = 1$ are connected components of (V, F).*

Proof. Suppose S is such that $\delta(S) \cap F = \emptyset$ and $f(S) = 1$. Because $\delta(S) \cap F = \emptyset$ then S is the union of some connected components of (V, F). Because $f(S) = 1$, we know $S \neq \emptyset$.

Assume S is not a single connected component. Let $C \subseteq S$ be any connected component and note that $C - S$ is also a union of connected components. Because f is proper, we then have either $f(S) = 1$ or $f(C - S) = 1$. Therefore, S is not a minimal subset with $f(S) = 1$ and $\delta(S) \cap F = \emptyset$. ■

Proof of Claim 1. First recall that in any tree on n nodes, the total degree is $2(n - 1)$. If $S \subseteq$ (nodes in T)

is such that all leaves are is S, then:

$$\begin{aligned}\sum_{v\in S}\deg_T(v) &= \sum_{v\in T}\deg_T(v) - \sum_{v\notin S}\deg_T(v)\\ &= 2(n-1) - \sum_{v\notin S}\deg_T(v)\\ &\le 2(n-1) - 2(n-|S|) && \text{(Because } \deg_T(v)\ge 2 \text{ for a non-leaf } v)\\ &= 2|S|-2. && (18.1)\end{aligned}$$

The total degree of nodes in S is $2|S|-2$ hence the average degree of $v\in S$ is ≤ 2.

We will heavily use the notation from the STEINER FOREST algorithm from last lecture, with the understanding that the algorithm applies to arbitrary proper functions. Now consider iteration i. Let $\mathcal{C}'_i$ be the set of connected components of (V, F_i) and note that $\mathcal{C}_i \subseteq \mathcal{C}'_i$ is the set of components S of (V, F_i) with $f(S) = 1$.

Consider the graph $H = (\mathcal{C}'_i, E_i)$ where we have an edge in E_i for every $(u, v) \in F$ with u and v in different components of $\mathcal{C}'_i$. First, we claim that H is a forest. To see this, first note that $F_{i'}$ is a forest for every iteration i' because we only add a single edge per iteration and this edge bridges two connected components (so it cannot create a cycle).

Consider the final set of edges F' after the first loop but before the pruning. Because the components of (V, F_i) are connected subtrees of F', then contracting them results in a forest. The edges in E_i are a subset of the edges that remain after these components are pruned.

Finally, we prove that all leaves in H are active. To achieve this, we are going to use the fact that function f is proper and that F is minimal.

Suppose that S is an inactive leaf on H with parent edge e. Let B be the collection of connected components in $\mathcal{C}'_i$ that are connected to S. Because H is a forest, then the restriction of H to B and all incident edges is a tree.

Then:

1. $f(S) = 0$: This is simply because S is inactive.
2. $f(B - S) = 1$: To see this, note by minimality that F is feasible but $F - \{e\}$ is infeasible. Lemma 1 implies there is some connected component S' of $(V, F - \{e\})$ that has $f(S') = 1$. But the only connected components in $(V, F - \{e\})$ that are not also in (V, F) are S and $B - S$. We know $f(S) = 0$, so it must be that $f(B - S) = 1$. This is illustrated in Figure 18.1.
3. Since f is proper, $f(V - (B - S)) = 1$ by property 2. Equivalently, $f((V - B) \cup S) = 1$

Therefore, either $f(V - B) = 1$ or $f(S) = 1$ by property 3. Again, since $f(S) = 0$ it must be that $f(V - B) = 1$. But if $f(V - B) = 1$, then $f(B) = 1$ by property 2. However, $\delta(S) \cap F = \emptyset$ which contradicts F being feasible.

■

18.2 The k-Median problem

Now we take a short break from LPs.

The k-MEDIAN problem is a variant of the k-SUPPLIERS problem we have seen in Assignment 2. We are given a set of vertices V portioned into clients C and facilities $F = V - C$ as well as distances $d(i, j)$ for all $i, j \in V$.

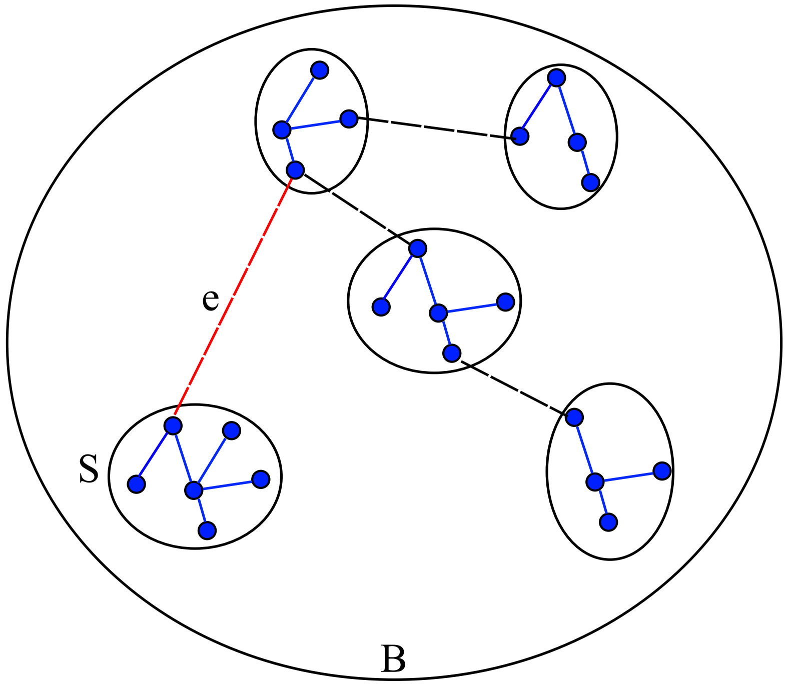

Figure 18.1: Illustration for proof of claim 1: The set B contains all nodes connected to inactive leaf S in the graph H and e denotes the parent edge of S.

We are also given integer $1 \leq k \leq |F|$. The goal is to find a $S \subseteq F$, with $|S| = k$, that minimizes

$$f(S) \triangleq \sum_{j \in C} \max_{i \in S} d(i,j) = \sum_{j \in C} d(j,S). \tag{18.2}$$

Note that rather that tying to minimize the maximum distance of a client to a facility as in k-SUPPLIERS, we minimize the sum of distances between clients and their nearest facility.

A simple greedy algorithm for the k-MEDIAN problem is presented below. For $i \in S, i' \in F - S$ we let $S - i + i'$ denote $(S - \{i\}) \cup \{i'\}$.

Algorithm 1 Local Search for k-MEDIAN

```
S ← any subset of F of size k
while ∃ i ∈ S, i' ∈ F − S s.t f(S) > f(S − i + i')  do
    S ← S − i + i'
end while
return  S
```

We will show the above algorithm is a 5-approximation for the k-MEDIAN problem. However, it is not guaranteed to run in polynomial time. To overcome this issue, we consider a slight variant of Algorithm 1 where ϵ is a parameter we may specify.

Claim 2 *Algorithm 2 runs in polynomial time in the input size and* $\frac{1}{\epsilon}$.

Proof. It is easy to check if there is a 0-cost solution. If some client j has $d(i,j) > 0$ for any facility then there is no 0-cost solution. Otherwise, we may assume $d(i,i') > 0$ for any two i,i', otherwise we can discard one of

Algorithm 2 Polynomial-Time Local Search for k-MEDIAN

```
If there is a solution with cost 0, then return it.
δ ← ε/(10 · k)
S ← any subset of F of size k
while ∃ i ∈ S, i′ ∈ F − S s.t (1 − δ) · f(S) > f(S − i + i′)  do
   S ← S − i + i′
end while
return S
```

them which does not change the optimum (clearly there is no point of opening both if $d(i, i') = 0$). But then there is a 0-cost solution if and only if the number of remaining facilities is at most k.

Next, let

$$\Delta = \frac{n \cdot \max_{i,j}\ d(i,j)}{\min_{i,j:d(i,j)>0} d(i,j)}.$$

Note that $\log \Delta$ is polynomial in the input size (i.e. number of bits used to describe the input).

Let S_I and S_F denote the initial and final set S used by the algorithm respectively. If t equals the number of iterations performed by the algorithm then

$$f(S_F) \leq (1-\delta)^t f(S_I).$$

The claim is that $t \leq \frac{10 \cdot k}{\epsilon} \ln \Delta$, which is polynomial in the input size and $\frac{1}{\epsilon}$.

Otherwise, we would have

$$(1-\delta)^t < e^{-\ln \Delta} = \frac{1}{\Delta},$$

where we have used the fact that $(1-\delta)^{1/\delta} \leq \frac{1}{e}$. In other words, $\Delta < f(S_I)/f(S_F)$.

However, we know $f(S_I) \leq n \cdot \max_{i,j} d(i,j)$ because each of the n clients travels distance at most the maximum distance in the metric. and we also know $f(S_F) \geq \min_{i,j:d(i,j)>0} d(i,j)$ because there is no 0-cost solution so some client has to travel distance at least the minimum nonzero distance in the metric. Thus, $f(S_I)/f(S_F) \leq \Delta$ which contradicts what we just saw.

Therefore, this is a polynomial-time algorithm. ■