Network Design and Game Theory
Spring 2011
Lecture 6

Instructor: Mohammad T. Hajiaghayi
Scribe: Marek Cygan

October 5, 2011

1 Overview

During this lecture we will present approximation algorithms for three network design problems: Group Steiner Tree, k-median and k-center.

2 Group Steiner Tree

Let us recall the problem definition:
Group Steiner Tree

Input : An undirected weighted graph $G = (V, E)$ and subsets $S_1, \ldots, S_p \subseteq V$.

Goal: Find a tree in G that connects at least one vertex from each S_i.

Observe that we may guess one vertex $r \in V$ (e.g. by trying all vertices from S_1), that is a part of an optimum solution. We want to prove that if G is a tree rooted at r of depth h, than we can in polynomial time find a solution of cost at most $O(h \log p)$ times more than the optimum solution. The following theorem is due to Garg, Konjevod and Ravi [2], but we present a simpler proof here.

Theorem 1 *There is an $O(h \log p)$-approximation algorithm for* Group Steiner Tree *when $G = (V, E)$ is a tree rooted at r of depth h.*

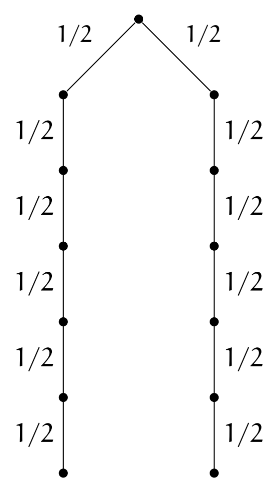
Proof: For simplicity we first modify the given tree to make sure that each group S_i contains only leaves of the tree G. That is if $v \in V$ is an internal node and $v \in S_i$, then we connect to v a leaf v' by an edge of cost zero and for each group that contains v we remove v from it and add v' instead.

Consider the following integer programming formulation of the problem.

$$\begin{array}{lllr}
\min & \sum_{e \in E} \mathrm{cost}(e) x_e & & (1)\\
& f^g_r = 1 & \forall g = 1, \ldots, p & (2)\\
& f^g_e \le x_e & \forall g, e & (3)\\
& f^g_v = \sum_{v' \in \mathrm{children}(v)} f^g_{vv'} & \forall g, \text{internal node } v & (4)\\
& f^g_v = f^g_{\mathrm{par}(v)v} & \forall g, v \ne r & (5)\\
& f^g_v = 0 & \forall g, v \text{ is a leaf}, v \notin S_g & (6)\\
& x_e, f^g_v, f^g_e \in \{0,1\} & \forall e, v, g & (7)
\end{array}$$

Observe that the optimum value of the above integer program is exactly the same as the optimum value for the Group Steiner Tree problem, since $x_e = 1$ denotes buying an edge and the variables f^g_v, f^g_e represent the flow of value one that goes from r to some leaf of S_g for each group g. Therefore by relaxing constraint (7) to $0 \le x_e, f^g_v, f^g_e \le 1$ we obtain a linear programming relaxation and its solution OPT^* is at most OPT. Note that w.l.o.g. we may assume that in the optimum solution to the LP relaxation we have $x_e \le x'_e$ when e is a descendant of e'.

Observe that the example below shows that we can not simply use independent rounding for each edge, since we would obtain probability of connection 2^{-h}.



Consider the following randomized rounding process. Initially we place exactly one particle at the root r. Next for each child v of the root r independently we buy the edge rv with probability x_{rv} and if we buy the edge then we place a particle in v. We proceed in a similar way in a top-down manner, that is when we consider a vertex v, then we check whether there is a particle at v, and if there is a particle at v than for each edge e going down from v we buy e with probability $x_e/x_{v\mathrm{par}(v)}$ and if we buy e than we place a particle in its other endpoint, where by par(v) we denote the parent of v in the rooted tree.

Lemma 1 *The expected cost of a tree bought by the above rounding process is equal to* OPT^*.

Proof: We prove by induction that for each node $v \neq r$ the probability that a particle reaches v is equal to $x_{\mathrm{par}(v)v}$. For v at depth 1 the lemma clearly holds. For v at depth $i > 1$ we have

$$\begin{aligned} \mathsf{P}[\text{a particle reaches } v] =& \mathsf{P}[\text{a particle reaches } \mathrm{par}(v)] * \frac{x_{\mathrm{par}(v)v}}{x_{\mathrm{par}(v)\mathrm{par}(\mathrm{par}(v))}} = \\ =& x_{\mathrm{par}(v)v} \end{aligned}$$

Hence we buy an edge e with probability x_e and by the linearity of expectations the lemma follows. ■

Lemma 2 *Let* S_g *be an arbitrary group. For any node* v *at depth* $i > 0$ *such that* $x_{\mathrm{par}(v)v} > 0$*, the probability that a node from the group* S_g *gets connected to* v*, assuming that there is a particle at* v*, is at least* $\frac{f_v^g}{(h-i+1)x_{\mathrm{par}(v)v}}$*. Moreover for the root* r *the probability that at least one vertex from* S_g *gets connected to* r *is at least* $\frac{1}{h+1}$.

Proof: We prove the lemma by induction on $h - i + 1$. In the base of the induction we assume that v is a leaf and the thesis clearly follows, since if $f_v^g > 0$ then $v \in S_g$.

Now consider a non-leaf vertex v at depth $i > 0$. By P_v we denote the probability that the group S_g is **not** connected to v assuming a particle reached

v.

$$P_v = \prod_{v' \in \text{children}(v), e=vv'} (1 - \frac{x_e}{x_{\text{par}(v)v}}(1 - P_{v'})) \quad (8)$$

$$(\text{by induction}) \leq \prod_{v' \in \text{children}(v), e=vv'} (1 - \frac{x_e}{x_{\text{par}(v)v}} \cdot \frac{f^g_{v'}}{(h-i)x_e}) \quad (9)$$

$$= \prod_{v' \in \text{children}(v), e=vv'} (1 - \frac{f^g_{v'}}{(h-i)x_{\text{par}(v)v}}) \quad (10)$$

$$(\text{since } 1 - x \leq e^{-x}) \leq \exp(-\sum f^g_{v'}/((h-i)x_{\text{par}(v)v})) \quad (11)$$

$$(\text{by constraints } (3,4,5)) = \exp(\frac{-f^g_v}{(h-i)x_{\text{par}(v)v}})) \quad (12)$$

$$= \exp(\frac{-1}{(h-i)x_{\text{par}(v)v}/f^g_v})) \quad (13)$$

$$(\text{since } e^{1/(x-1)} \leq 1 - \frac{1}{x}) \leq 1 - \frac{1}{(h-i)x_{\text{par}(v)v}/f^g_v + 1} \quad (14)$$

$$\leq 1 - \frac{f^g_v}{(h-i+1)x_{\text{par}(v)v}} \quad (15)$$

Therefore $1 - P_v \leq \frac{f^g_v}{(h-i+1)x_{\text{par}(v)v}}$. The analysis for the root is almost the same as above, the only difference is that the term $x_{\text{par}(v)v}$ is absent. ■

By Lemma 2 if we repeat the randomized rounding process $(h+1)\log p$ times, the probability that a group will not be connected to r is at most $(1 - \frac{1}{h+1})^{(h+1)\log p} \leq e^{-\log p} = \frac{1}{p}$. Thus after the rounding process we may connect each of the not yet connected groups independently, by paying at most OPT for each group, which gives total cost of the solution $(h+1)\log p\mathsf{OPT}^* + p \cdot \frac{1}{p}\mathsf{OPT} = O(h \log p\mathsf{OPT})$ and leads to $O(h \log p)$ approximation. Consequently we prove Theorem 1. ■

As already mentioned in the previous lecture, for general graphs Theorem 1 together with FRT probabilistic embedding into trees (which have depth bounded by $O(\log n)$) result in the following theorem.

Theorem 2 *There is an* $O(\log^2 n \log p)$ *approximation of the* Group Steiner Tree *problem.*

3 k-center

In this section we consider the following problem.

k-center

Input : A network $G = (V, d)$ and an integer k, where $d : V \times V \to \mathbb{R}_+$ is a distance function (a metric).

Goal: Find a set X of exactly k vertices such that $\max_{v \in V} d(v, X)$ is minimized.

Note that there are only $\binom{n}{2}$ possible optimum value for the problem, hence we may guess the solution value by trying all possibilities (or by using a binary search). Assume that δ is the guessed optimum value.

Consider an auxiliary undirected graph $G_\delta = (V, E)$, where $uv \in E$ iff $d(u, v) \leq \delta$.

Observation 1 *The optimum solution forms a dominating set in* G_δ*.*

By a power of an undirected graph $H = (V, E)$ we denote the following operation. For a positive integer c we define $H^c = (V, E')$, where $uv \in E'$ iff the distance in H between u and v is at most c (note that we measure just the number of edges traversed).

Lemma 3 *If* X *is a dominating set in* G_δ^c *for some positive integer* c*, and* $|X| \leq k$*, than* X *is a* c*-approximation for the* k-CENTER *problem.*

Proof: Since each vertex in V is within distance c in the undirected unweighted graph G_δ, by the fact that each two adjacent vertices in G_δ are within distance δ with respect to the metric d, the lemma follows. ■

Our goal is to find a set X that is a dominating set in G_δ^2 which consequently gives a two approximation for the k-MEDIAN problem by Lemma 3. As the set X we take any inclusion maximal independent set in G_δ^2. By the definition of X we know that it is a dominating set in G_δ^2 since otherwise it would not be an inclusion maximal independent set in G_δ^2. If $|X| \leq k$ then we are done since we have found a 2-approximation. However if $|X| > k$ than it means that $\mathsf{OPT} > \delta$ because vertices of X have disjoint neighbourhoods in G_δ and for each $v \in X$ any solution needs to select at least one vertex from $N_{G_\delta}(v)$.

By a simple reduction from the dominating set problem one can show that there is no $(2 - \epsilon)$-approximation for the k-CENTER problem.

4 k-median

In the last section we consider the k-MEDIAN problem.

k-MEDIAN

INPUT : A network $G = (V, d)$ and an integer k, where $d : V \times V \to \mathbb{R}_+$ is a distance function (a metric).

GOAL: Find a set X of exactly k vertices such that $\sum_{v \in V} d(v, X)$ is minimized.

For a set $X \subseteq V$ by $\mathrm{cost}(X) = \sum_{v \in V} d(v, X)$ we denote the cost of having X as a solution. We denote vertices from X as facilities which we have to open and $d(v, X)$ is the cost of serving a client at the vertex v.

Consider the following local search algorithm. At the beginning pick any k vertices to the set X. Next as long as there exists a pair of vertices v_1, v_2

such that $v_1 \in X$, $v_2 \in V \setminus X$, and $\text{cost}(X \setminus \{v_1\} \cup \{v_2\}) < \text{cost}(X)$ we set $X := X \setminus \{v_1\} \cup \{v_2\}$.

The following theorem is due to Arya et al. [1] but we present a simpler proof due to Gupta and Tangwongsan [3].

Theorem 3 *Assume that X is any local minimum with respect to the above swaps. Then* $\text{cost}(X) \leq 5\mathsf{OPT}$.

Proof: To show that a local optimum is a good approximation, the standard approach is to consider a carefully chosen subset of potential swaps: if we are at a local optimum, each of these swaps must be non-improving. This gives us some information about the cost of the local optimum.

Let X^* be the set of k facilities chosen by an optimum solution. We define a mapping $\eta : X^* \to X$ that maps each facility $f^* \in X^*$ from the optimum solution X^* to a closest facility $\eta(f^*) \in X$. Now we define a set of pairs $S \subseteq X \times X^*$, which we later use as potential swaps.

Lemma 4 *There exists a set $S \subseteq X \times X^*$, such that all the following conditions hold:*

- *each $f^* \in X^*$ appears in exactly one pair in S (in the second coordinate),*
- *each $f \in X$ appears in at most two pairs in S (in the first coordinate),*
- *for each $(f, f^*) \in S$ and for each $f_2^* \in X^* \neq f^*$ we have $\eta(f_2^*) \neq f$.*

Proof: Let us partition the set X into the following three sets:

- $X_0 = \{f \in F : |\eta^{-1}(f)| = 0\}$,
- $X_1 = \{f \in F : |\eta^{-1}(f)| = 1\}$,
- $X_{2+} = \{f \in F : |\eta^{-1}(f)| \geq 2\}$.

Note that by a simple averaging argument one can show $|X_0| \geq |X_{2+}|$. Since $|X| = |X^*|$ we infer that:

$$2|X_0| \geq |X_0| + |X_{2+}| = |X \setminus X_1| = |X^* \setminus \eta^{-1}(X_1)|. \quad (16)$$

To obtain the set S we first take all the pairs (f, f^*) such that $\eta^{-1}(f) = \{f^*\}$. Next we group all the vertices from $A^* = X^* \setminus \eta^{-1}(X_1)$ in pairs and for each pair $u^*, v^* \in A^*$ we select a vertex f from X_0 in such a way that each vertex in X_0 is selected at most once (by (16) it is always possible). We add to the set S the pairs (f, u^*) and (f, v^*).

It is easy to check that our construction satisfies all three conditions needed by the lemma. ■

Now we use the fact that each swap in S is non-improving to show that X has small cost. Assume that $\varphi : V \to X$ and $\varphi^* : V \to X^*$ are functions mapping each client to its closest facility in X and X^* respectively (breaking ties arbitrarily). For any client $v \in V$ let $O_v = d(v, X^*) = d(v, \varphi^*(v))$ and $A_v = d(v, X) = d(v, \varphi(v))$ be costs of serving the client v in the optimum solution and in the local minimum X. Let $N^*(f^*) = \{v \in V : \varphi^*(v) = f^*\}$ be the set of clients assigned to f^* in the optimum solution and similarly define $N(f) = \{v \in V : \varphi(v) = f\}$.

Lemma 5 *For each swap* $(f, f^*) \in S$ *we have*

$$0 \leq \text{cost}(X \setminus \{f\} \cup \{f^*\}) - \text{cost}(X) \leq \sum_{v \in N^*(f^*)} (O_v - A_v) + \sum_{v \in N(f)} 2O_v \,. \tag{17}$$

Proof: Since we are removing f we need to reassign all the clients that were previously mapped to f, that is clients of $N(f)$. Map each client from $N^*(f^*)$ to f^*. Consider a client in $N(f) \setminus N^*(f^*)$. Let $\hat{f^*} = \varphi^*(v)$, that is $\hat{f^*}$ is the facility from X^* serving v, assign v to $\hat{(f)} = \eta(\hat{f^*})$, the closest facility in X to $\hat{f^*}$. Note that by the third property of Lemma 4 we have $\hat{f} \neq f$ and our assignment is valid. All clients in $V \setminus (N(f) \cup N^*(f^*))$ stay assigned as they were in φ.

Observe that for each client $v \in N^*(f^*)$ the change in cost is equal to exactly $O_v - A_v$, which gives the first sum in the inequality we are proving. Now consider any $v \in N(f) \setminus N^*(f^*)$. Its change in cost is equal to

$$\begin{aligned} d(v, \hat{f}) - d(v, f) &\leq d(v, \hat{f^*}) + d(\hat{f^*}, \hat{f}) - d(v, f) && \text{(by triangle inequality)} \\ &\leq d(v, \hat{f^*}) + d(\hat{f^*}, f) - d(f, v) && \text{(by definition of } \eta) \\ &\leq d(v, \hat{f^*}) + d(\hat{f^*}, v) && \text{(by triangle inequality)} \\ &= 2O_v \,. \end{aligned}$$

Since $\sum_{v \in N(f) \setminus N^*(f^*)} O_v \leq \sum_{v \in N(f)} O_v$ the lemma follows. ■

Observe that summing (17) over all pairs in S together with the fact that each $f^* \in X^*$ appears exactly once and each $f \in X$ at most twice finishes the proof of Theorem 3. ■

Note that Theorem 3 as is does not suffice to obtain a polynomial time constant factor approximation algorithm, since it does not specify how fast the set X will converge to a local minimum. It is however possible to prove that if one considers the best swap possible than in poly$(n, 1/\epsilon)$ time one can obtain $(5 + \epsilon)$-approximation local search algorithm.

Currently the best approximation algorithm for k-MEDIAN has $(3 + \epsilon)$ approximation ratio and is due to Arya et al. [1] and it is also a local search algorithm but it allows for swaps involving a bigger number of vertices.

References

[1] Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristics for k-median and facility location problems. *SIAM J. Comput.*, 33(3):544–562, 2004.

[2] Naveen Garg, Goran Konjevod, and R. Ravi. A polylogarithmic approximation algorithm for the group steiner tree problem. *J. Algorithms*, 37(1):66–84, 2000.

[3] Anupam Gupta and Kanat Tangwongsan. Simpler analyses of local search algorithms for facility location. *CoRR*, abs/0809.2554, 2008.