CS880: Approximations Algorithms	
Scribe: Siddharth Barman	**Lecturer:** Shuchi Chawla
Topic: Steiner Tree; Greedy Approximation Algorithms	**Date:** 01/25/07

In this lecture we give an algorithm for Steiner tree and then discuss greedy algorithms.

2.1 Steiner Tree

Problem Statement: Given a weighted graph $G = (V, E)$ and a set $R \subseteq V$, our goal is to determine the least cost connected subgraph spanning R. Vertices in R are called terminal nodes and those in $V \backslash R$ are called Steiner vertices.

Note that the least cost connected subgraph spanning R is a tree. Also we are free to use non terminal vertices of the graph (the Steiner nodes) in order to determine the Steiner tree.

We first show that we can assume without loss of generality that G is a complete graph. In particular, let G^c denote the metric completion of G: the cost of an edge (u, v) in G^c for all $u, v \in V$ is given by the length of the shortest path between u and v in G. As an exercise verify that costs in G^c form a metric.

Lemma 2.1.1 *Let T be a tree spanning R in G^c, then we can find a subtree in G, T' that spans R and has cost at most the cost of T in G^c.*

Proof: We can get T' by taking a union of the paths represented by edges in T. Cost of edges in G^c is the same as the cost of the paths (shortest) of G they represent. Hence the sum of costs of all the paths making up T' is at most the cost of T. ■

As seen in the previous lecture a lower bound is a good place to start looking for an approximate solution. We will use ideas from the previous lecture relating trees to traveling salesman tours to obtain a lower bound on the optimal Steiner tree.

Lemma 2.1.2 *Let G^c_R be the subgraph of G^c induced by R. Then the cost of the MST in G^c_R is at most the cost of the optimal TSP tour spanning G^c_R.*

Proof: See previous lecture. ■

Lemma 2.1.3 *The cost of the optimal TSP tour spanning G^R_c is at most twice OPT (the optimal Steiner tree in G^c spanning R)*

Proof: As shown in Figure 2.1.1 traversing each edge of the Steiner tree twice gives us a path which covers all the terminal nodes i.e. a tour covering all of R. Hence TSP spanning $R \leq 2OPT$. ■

Corollary 2.1.4 *The cost of the MST of G^c_R is at most twice OPT.*

Proof: Follows from the above mentioned lemmas. ■

The lower bound in itself suggests a 2-approximation to the Steiner tree. The algorithm is to simply determine the MST on R in G^c_R i.e.

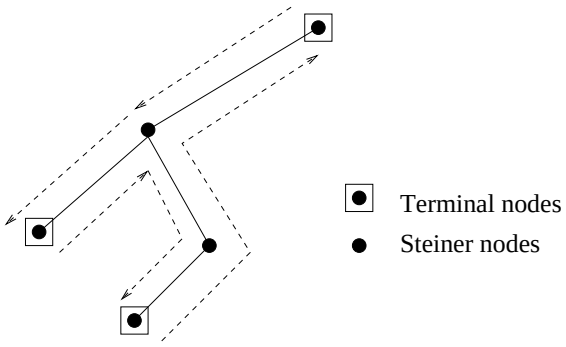

Figure 2.1.1: Constructing tour spanning R from the optimal Steiner tree

- Consider G^c, the metric completion of graph G.
- Get G^c_R, the subgraph induced by set R on G_c.
- Determine MST on G^c_R, translate it back to a tree in G as described in the proof of lemma 2.1.1.
- Output this tree

Theorem 2.1.5 *The algorithm above gives a* 2*-approximation to Steiner tree.*

Proof: Follows from the three lemmas stated above. ■

By a more careful analysis the algorithm can be shown to give a $2\left(1-\frac{1}{|R|}\right)$ approximation. This is left as an exercise.

The best known approximation factor for the Steiner tree problem is 1.55 [5]. Also from the hardness of approximation side it is known that Steiner tree is "$APX - Hard$", i.e. there exists some constant $c > 1$ s.t. Steiner tree is $\mathcal{NP}$- Hard to approximate better than c [1].

2.2 Greedy Approximation Algorithms—the min. multiway cut problem

Next we look at greedy approximation algorithms. The design strategy carries over from greedy algorithms for exact algorithm design i.e. our aim here is to pick the myopic best action at each step and not be concerned about the global picture.

First we consider the **Min Multiway Cut Problem**.

Problem Statement: Given a weighted graph G with a set of terminal node T and costs (weights) on edges our goal is to find the smallest cut separating all the terminals.

Let $k = |T|$ denote the cardinality of the terminal set. When $k = 2$, the problem reduces to simple min cut and hence is polytime solvable. For $k \geq 3$ it is known that the problem is $\mathcal{NP}$-Hard and

also $APX\text{-}Hard$ [3]. Note that multiway cut is different from the multicut problem. We will study the latter later on in the course.

We provide a greedy approximation algorithm for the min multiway cut problem and give a tight analysis to show that it achieves an approximation factor of $2\left(1-\frac{1}{k}\right)$. The algorithm and analysis is due to Dahlhaus et al. [3]

Algorithm: For every terminal $t_i \in T$, find the min-cut C_i separating t_i from $T\backslash\{t_i\}$. A Multiway cut is obtained by taking the union of the $(k-1)$ smallest cuts out of C_1,C_2,...C_k.

The size of multiway cut determined by the greedy algorithm is $\sum_{i\leq k}|C_i|$.

We proceed to analyze the relative goodness of the greedy solution with respect to that of the optimal. The idea behind the analysis is captured by figure 2.2.2.

Lemma 2.2.1 $\sum_i |C_i| \leq 2OPT$, *where* OPT *is the cost of the minimum multiway cut.*

Proof: Consider the components generated by OPT, call them S_1,...,S_k. For each t_i, consider the full cut C_i' consisting of all the edges in OPT with *exactly one* end point in S_i. We know that $|C_i| \leq |C_i'|$, because C_i' separates t_i from $T\backslash\{t_i\}$.

Also (Figure 2.2.2) on summing up the capacities of C_i's each edge of the min multiway cut is counted twice hence we have $\sum_{i=1}^{k}|C_i'| = 2OPT$. Combining the two we get $\sum_{i=1}^{k}|C_i| \leq 2OPT$ ■

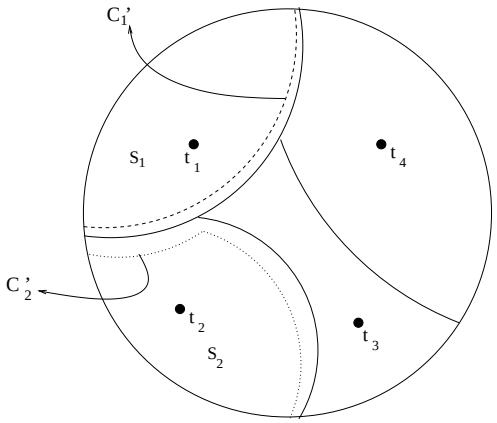

Figure 2.2.2: Min multiway cut

A tighter version of the above may be stated as follows.

Lemma 2.2.2 *Let* $j = argmax_i|C_i|$, *i.e.* j *is the largest of the cuts then*

$$\sum_{i\in[k],i\neq j}|C_i| \leq 2\left(1-\frac{1}{k}\right)OPT$$

Proof:

As $|C_j|$ is greater than the average value of the cuts we have,

$$\sum_{i\in[k],i\neq j} |C_i| \leq \left(1-\frac{1}{k}\right)\sum_{i\in[k]} |C_i|$$

Combining this with the previous lemma we get the desired result. ■

Theorem 2.2.3 *The above algorithm gives a* $2\left(1-\frac{1}{k}\right)$ *approximation to min multiway cut.*

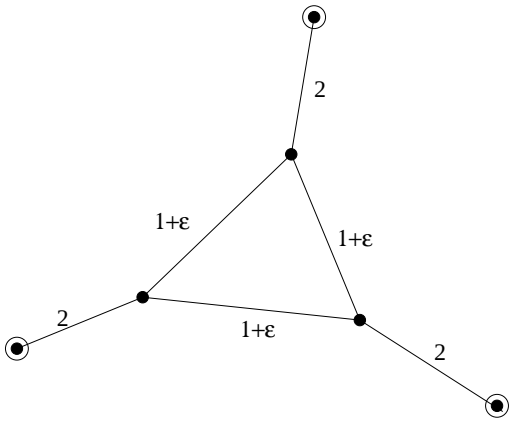

Figure 2.2.3: Bad example for Min multiway cut greedy algorithm

We now give a tight example for the algorithm. First let us examine the case of $k = 3$ as in the figure above. In this case the algo. returns a cut of cost 4, whereas the min. multiway cut has cost $3(1+\epsilon)$. Generalizing the triangle of Figure 2.2.3 to a k cycle we find that the analysis of the algorithm provided above is in fact tight. In particular for the k cycle case, $|C_i| = 2$ for each i and hence the greedy multiway cut is of size $2(k-1)$. The min multiway cut is in fact the k cycle hence $OPT = (1+\epsilon)k$. The example shows that our analysis of the provided greedy algorithm is in fact tight.

Though the analysis provided here is tight, better algorithms exist for the min multiway cut problem. Calinescu et al. [2] gave a $\left(\frac{3}{2}-\frac{1}{k}\right)$ approx. which was subsequently improved to a 1.3438 approx by Karger et al. [4].

References

[1] M. Bern, P. Plassmann. The steiner problem with edge lengths 1 and 2. In *Information Processing Letters* Volume 32-1 (1989), pp: 171-176.

[2] G. Calinescu, H.J. Karloff, Y. Rabani. An Improved Approximation Algorithm for Multiway Cut. In *STOC*(1998), pp: 48-52.

[3] E. Dahlhaus, D.S. Johnson, C.H. Papadimitriou, P.D. Seymour, M. Yannakakis. The Complexity of Multiterminal Cuts. In *SIAM J. Comput.* Volume 23 (1994), pp: 864-894.

[4] D.R. Karger, P.N. Klein, C. Stein, M. Thorup, N.E. Young. Rounding Algorithms for a Geometric Embedding of Minimum Multiway Cut. In *STOC*(1999), pp: 668-678.

[5] G. Robins, A. Zelikovsky. Tighter Bounds for Graph Steiner Tree Approximation. In *SIAM Journal on Discrete Mathematics* Volume 19-1 (2005), pp: 122-134.