CMPUT 675: Approximation Algorithms	**Winter 2005**
Lecture 15: Mar 9	
Lecturer: Mohammad R. Salavatipour	*Scribe: Jianjun Zhou*

Last time we saw an $O(m^{\frac{1}{2}+\epsilon})$-approximation algorithm ($m = |E|$) for the problem of Packing Edge Capacitated Directed Steiner Trees. As we can see, the approximation ratio is not good, but actually we can not improve it much because the problem is hard. In this lecture, we are going to study a proof showing the reason.

15.1 Packing Directed Steiner Trees

First we introduce two different versions of the problem. The problem can be defined based on edge constraints or vertex constraints.

Definition 15.1 (Packing Edge Capacitated Directed Steiner Tree (PED)) ***Input:*** *Given a directed graph $G(V, E)$, and a set of terminals $T \subseteq V$ containing root r. There is a capacity $C_e \in Z^+$ on every edge e.* ***Goal:*** *Compute the maximal number of Steiner trees such that each edge e is in at most C_e trees.*

Definition 15.2 (Packing Vertex Capacitated Directed Steiner Tree (PVD)) *We have capacities on Steiner nodes. Find the maximal number of Steiner tree such that every Steiner node $v \in T$ belongs to at most C_v tree.*

PED and PVD are equally hard. There is a theorem showing that they can be converted into each other.

Theorem 15.3 *Given $I = (G(V, E), T)$ is an instance of PED, then there is an instance $I' = (G'(V', E'), T')$ for PVD such that $|G'| = poly(|G|)$ and I has k Steiner trees (satisfying edge capacities) iff I' has k Steiner trees (satisfying vertex capacities).*

Proof: (1st direction) Create a new vertex v_{xy} on every edge xy with a capacity equal to the capacity of that edge. All the other Steiner nodes have ∞ capacities. The root and the other terminals are the same in G and G'. It can be seen that G has k (directed) Steiner trees satisfying edge capacities if and only if G' has k (directed) Steiner trees satisfying vertex capacities.

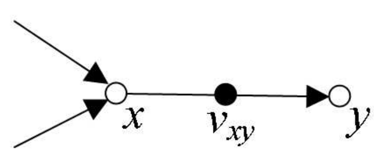


Figure 15.1: A new vertex v_{xy} on edge xy with $C_{v_{xy}} = C_{xy}$.

(2nd direction) For each node $v \in V$, G' contains two nodes v_1, v_2. If $v \in T$ then both v_1 and v_2 become terminals in G', and if $r \in T$ is the root then r_1 becomes the root in G'. We add v_1v_2 to E' and give it the same capacity as vertex v in G. If $v \in T$, then we give infinite capacity to v_1v_2. Furthermore, for every edge $uv \in E$ we create an edge u_2v_1 (with infinite capacity) in E' and for every edge $vw \in E$ we create an edge v_2w_1 (with infinite capacity) in E'. It is easy to see that if $\mathcal{T}$ is a collection of k Steiner trees in G that satisfy vertex capacities then there is a collection $\mathcal{T}'$ of k Steiner trees in G' that satisfy edge capacities. Conversely, suppose that $\mathcal{T}'$ is a collection of k Steiner trees in G' satisfying edge capacities. Then for every edge v_1v_2 (corresponding to a vertex $v \in V(G)$ with capacity c_v in G) there are at most c_v trees containing that edge. Therefore, by contracting the edges of the form v_1v_2 on each tree of $\mathcal{T}'$ we obtain a collection of k Steiner trees in G such that for every vertex v there are at most c_v trees containing it.

■

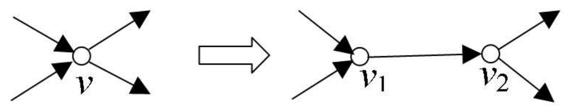
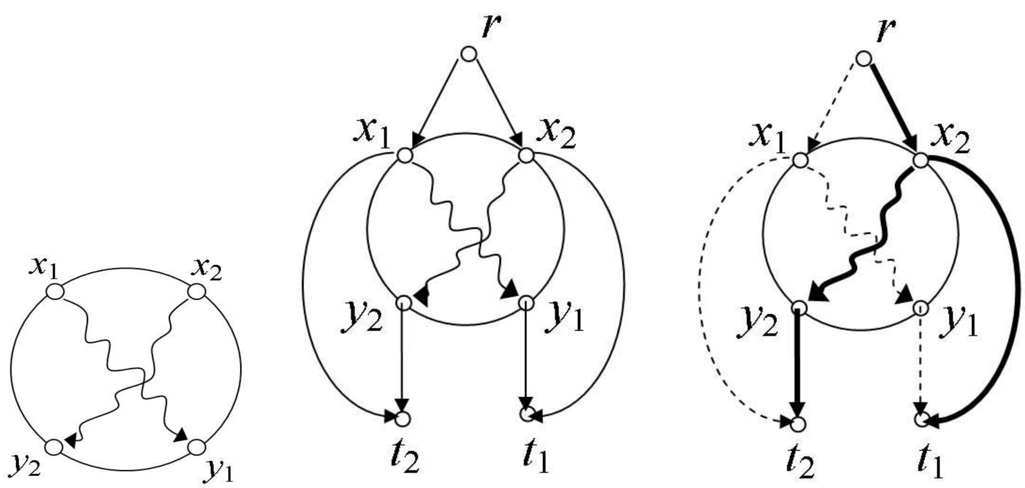


Figure 15.2: Split v into v_1 and v_2 with $C_{v_1v_2} = C_v$.

Since the two problems can be easily converted into each other, in the remaining of this lecture, we will focus on PVD only. First we show that even finding two disjoint Steine trees even if $|T| = 3$ is NP-hard. We will use a reduction from the following useful problem called 2DIRPATH. The NP-compeleteness of this problem has been used to prove hardness results for other problems, the most notably one being the disjoint paths problems.



(a) G (b) G' (c) The two Steiner trees in G', one in dash lines and one in bold lines.

Figure 15.3: The problem of 2DIRPATH

Definition 15.4 (2DIRPATH) *Given a directed graph G with distinct vertices x_1, y_1, x_2, and y_2. Question: are there two vertex-disjoint paths, one from x_1 to y_1 and one from x_2 to y_2?*

Theorem 15.5 *2DIRPATH is NP-hard.*

Theorem 15.6 *Given an instance of PVD with only 3 terminals (including the root), it is NP-hard to find even two Steiner trees even if all capacities are 1.*

Proof: Given a graph G as instance of 2DIRPATH, we construct a graph G' as follows. We add a root r, two terminals t_1 and t_2, and a set of edges: rx_1, rx_2, y_1t_1, y_2t_2, x_1t2, and x_2t_1. If there are (vertex) disjoint paths $x_1P_1y_1$ and $x_2P_2y_2$ in G then clearly $x_1P_1y_1 \cup \{rx_1, y_1t_1, x_1t_2\}$ and $x_2P_2y_2 \cup \{rx_2, y_2t_2, x_2t_1\}$ form two vertex-disjoint directed Steiner trees.

Conversely, if there are two vertex-disjoint directed Steiner trees T_1 and T_2 in G' then, since r has only two outgoing edges, we may assume that $rx_1 \in T_1$ and $rx_2 \in T_2$. Therefore, there is a path from x_1 to t_1 in T_1, which must go through y_1 (since x_2 is not in T_1), and a path from x_2 to t_2 in T_2, which must go through y_2 (since x_1 is not in T_2). These two paths are vertex-disjoint because T_1 and T_2 are vertex-disjoint. ■

With the knowledge of 2DIRPATH, we can prove the theorem below.

Theorem 15.7 *Unless P = NP, every approximation algorithm for PVD has factor $\Omega(n^{\frac{1}{3}-\epsilon})$ (for PED, the factor is $\Omega(m^{\frac{1}{3}-\epsilon})$), for any $\epsilon > 0$.*

Proof: We will prove a slightly modified version of the theorem with factor $\Omega(n^{\frac{1}{4}-\epsilon})$. The proof of the original factor $\Omega(n^{\frac{1}{3}-\epsilon})$ is similar.

Given an instance $I(G(V,E), x_1, y_1, x_2, y_2)$ for 2DIRPATH, we construct a graph H as follows. Let $N = |V(G)|^{\frac{1}{\epsilon}}$ for an arbitrary small $\epsilon > 0$. Create two row of vertices $a_1, a_2, \cdots, a_N$ and $b_1, b_2, \cdots, b_N$. Draw a line from a_i to b_j with $i \neq j$ (the solid lines in the figure) such that no 3 lines intersect at one point. Every line segment between two intersection points is going to be an edge. All the edges are directed from top to bottom. Now put a copy of G (for 2DIRPATH) at every intersection point of two solid lines, with the vertices x_1, x_2 being the two points that the directed edge that were going into the intersection point now enter the copy of G and y_1 and y_2 being the points that the directed lines that were going out of the intersection point now go out from. Finally add the edges connecting a_i and b_i (dash lines in the figure). Let the set of terminal be $T = \{r, b_1, b_2, \cdots, b_N\}$.

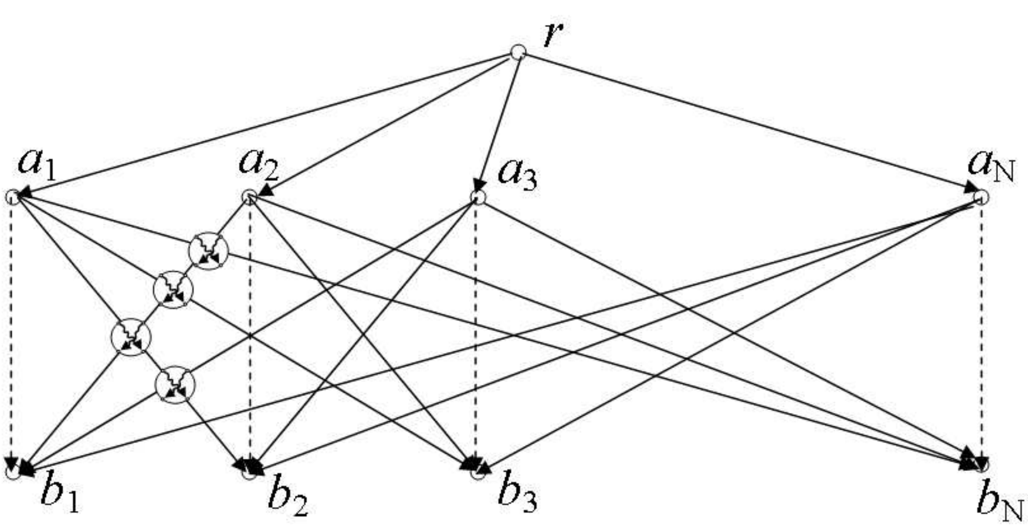


Figure 15.4: Construct a new graph H

Lemma 15.8 *if G is a yes-instance, then H has N vertex-disjoint Steiner tree.*

Proof: For every $1 \leq i \leq N$, consider the following set of edges (which will form a tree T_i): edge ra_i, all the directed edges on the line that connected a_i to b_j (for $i \neq j$), edge $a_i b_i$, plus the directed path from x_1 to y_1 in every copy of G placed in this graph. First note that each T_i is a Steiner tree and there are N of them. So we only need to show that they are vertex-disjoint. The only possible places that these trees can have a common vertex are within the copies of graph G placed at the intersection of the lines. But since G has vertex-disjoint paths, the paths from x_1 to y_1 and from x_2 to y_2 are vertex-disjoint at every copy of G. So the paths in these trees can "cross" each other without having any vertex in common at every intersection point (i.e. copy of G). ■

Lemma 15.9 *if G is a No-instance, then H does not even have two vertex-disjoint Steiner trees.*

Proof: Let's assume (by way of contradition) that there are a set of vertex-disjoint Steiner trees $\mathcal{L} = \{T_1, \cdots, T_k (k \geq 2)\}$ in H.

Claim 15.10 *There can not be a directed path from a_i to b_j (for any $i < j$) in any Steiner tree of $\mathcal{L}$.*

We prove this by induction on i. First consider the case $i = 1$ and suppose we have a path $a_1 \rightsquigarrow b_j$ $(j > 1)$ in some tree T_α. Then there can not be a path from any $a_i (i \geq 2)$ to b_1 in any Steiner tree T_β. The reason is that any such path must cross the path $a_1 \rightsquigarrow b_j$ at some intersection point, i.e. a copy of G. But because G is a No-instance, it doesn't have 2 vertex-disjoint paths from x_1 to y_1 and from x_2 to y_2. This shows that there cannot be a directed path from a_1 to any b_j for $j > 1$. For the induction step, let $i \geq 2$ and assume that there is a path $P_(a_i, b_j)$ from a_i to b_j $(j > i)$ in some tree T_α. Let $T_\beta \in \mathcal{T}$ be any other tree in $\mathcal{T}$ and $P_(r, b_i)$ be a path from r to b_i in T_β. We assume this path goes through a_l, for some $1 \leq l \leq N$. By induction hypothesis, there is no path from $a_1, \ldots, a_{i-1}$ to b_i in any tree. Also, $a_i \in T_\alpha$. So $l > i$. Again, if we consider the embeddings of these two paths $P_\alpha(a_i, b_j)$ and $P_\beta(r, b_i)$ on the plane, there is an intersection point (a copy of G) in which these two paths cross each other without having any vertex in common. But this is impossible because G is a "No" instance.

Therefore, the only possible path from r to b_N goes through a_N. Thus, there can be only one Steiner tree in $\mathcal{T}$: the one that contains a_N. ■

Of course H always has one Steiner tree: the union of all ra_i and $a_i b_i$ for $1 \leq i \leq N$. By Lemmas 15.8 and 15.9, deciding between N Steiner tree and 1 Steiner tree becomes deciding between if G is a Yes-instance or No-instance, which is NP-hard. So we have a gap of N. Since H has $\Theta(N^4)$ (every pair of vertices from a_i's and a pair from b_i's yields an intersection point) copies of G and $N = |V(G)|^{\frac{1}{\epsilon}}$, the number of vertices of H is $\Theta(N^4 N^\epsilon) = n$. Representing the gap N in terms of the number vertices, we get a hardness of $\Omega(n^{\frac{1}{4} - \epsilon'})$. ■

15.2 Minimum Steiner Forest

Next, we will switch to a new topic. The corresponding chapter in the textbook is chapter 22.

Steiner Forest Problem

Input: given undirected graph $G(V, E)$, capacities $C : E \rightarrow Q^+$, and a collection of disjoint subsets $S_1, \cdots, S_k$, $S_i \subseteq V$.

Question: find a min cost subgraph such that the vertices of each S_i are in one connected component.

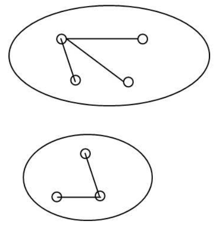


Figure 15.4: Illustration of the problem.

To derived a solution for the problem, we first define the connectivity requirement function r. For any two vertices u and v in graph G,

$$r(u,v) = \begin{cases} 1 & \text{if } u,\ v \text{ are in the same set } S_i; \\ 0 & \text{otherwise.} \end{cases}$$

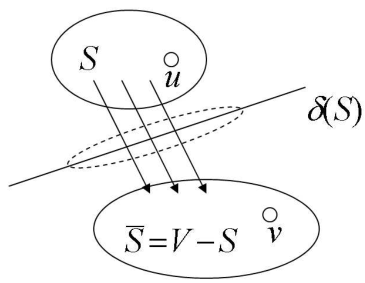


Figure 15.5: $\delta(S)$ is the set of edges with exactly one end-point in S

Recall that for a set $S \subseteq V$, $\delta(S)$ is the set of edges with exactly one end-point in S. The minimal number of edges that must cross cut S is :

$$f(s) = \begin{cases} 1 & \exists u \in S, v \in \bar{S}, r(u,v) = 1; \\ 0 & \text{otherwise.} \end{cases}$$

Then we can derive the LP formulation of the problem.

Primal LP

$$\begin{array}{ll} \text{minimize} & \sum C_e x_e, \\ \text{such that} & \sum_{e \in \delta(S)} x_e \geq f(S) \text{ for all } S \subseteq V, \\ & x_e \in \{0,1\}. \end{array}$$

Dual LP

$$\begin{array}{ll} \text{maximize} & y_s \sum f(S), \\ \text{such that} & \sum_{S:e\in S} y_s \leq C_e, \forall e \in E, \\ & y_s \geq 0. \end{array}$$