2007－AL－113 (3)
2007／7／3

容量付木状経路問題に対する近似解法

Ehab Morsy, 永持 仁
京都大学大学院情報学研究科数理工学専攻

概要

枝重み $w : E \to R^+$ を持つグラフ $G = (V, E)$, シンク $s \in V$, 要求量 $q : M \to R^+$ をもつターミナル集合 $M \subseteq V$, 経路容量 $\kappa > 0$, 整数値枝容量 $\lambda \geq 1$ が与えられたとき，容量付木状経路問題とは，ターミナル集合をいくつかの部分集合 Z_i, $i = 1, 2, \ldots, Z_\ell$ に分割し，それぞれ G のある部分木 T_i に沿ってシンク s との間に経路を確保するとき，必要な枝の設置コストを最小にする問題である．ここで，ターミナル集合要求量の制約 $\sum_{v \in Z_i} q(v) \leq \kappa$ を満たすように分割することが必要であり，1本の枝を通過することのできる要求量の和の上限は λ で抑えられ，設置コストは $\sum_{e \in E} \lceil |\{1 \leq i \leq \ell \mid T_i \text{ contains } e\}| / \lambda \rceil w(e)$ として表される．本論文では，容量付木状経路問題に対する $(2 + \rho_{\mathrm{ST}})$-近似アルゴリズムを与える．ただし，ρ_{ST} はスタイナー木問題に対する近似比である．

Approximating Capacitated Tree-Routings in Networks

Ehab Morsy, Hiroshi Nagamochi
Department of Applied Mathematics and Physics,
Graduate School of Informatics, Kyoto University

abstract

The capacitated tree-routing problem (CTR) in a graph $G = (V, E)$ consists of an edge weight function $w : E \to R^+$, a sink $s \in V$, a terminal set $M \subseteq V$ with a demand function $q : M \to R^+$, a routing capacity $\kappa > 0$, and an integer edge capacity $\lambda \geq 1$. The CTR asks to find a partition $\mathcal{M} = \{Z_1, Z_2, \ldots, Z_\ell\}$ of M and a set $\mathcal{T} = \{T_1, T_2, \ldots, T_\ell\}$ of trees of G such that each T_i spans $Z_i \cup \{s\}$ and satisfies $\sum_{v \in Z_i} q(v) \leq \kappa$. A subset of trees in $\mathcal{T}$ can pass through a single copy of an edge $e \in E$ as long as the number of these trees does not exceed the edge capacity λ; any integer number of copies of e are allowed to be installed, where the cost of installing a copy of e is $w(e)$. The objective is to find a solution $(\mathcal{M}, \mathcal{T})$ that minimizes the installing cost $\sum_{e \in E} \lceil |\{T \in \mathcal{T} \mid T \text{ contains } e\}| / \lambda \rceil w(e)$. In this paper, we propose a $(2 + \rho_{\mathrm{ST}})$-approximation algorithm to the CTR, where ρ_{ST} is any approximation ratio achievable for the Steiner tree problem.

1 Introduction

we consider a capacitated routing problem under a multi-tree model. Under this model, we are interested in constructing a set $\mathcal{T}$ of tree-routings that connects given terminals to a sink s in a network with a routing capacity $\kappa > 0$ and an edge capacity $\lambda > 0$. A network is modeled with an edge-weighted undirected

graph. Each terminal has a demand > 0, and a tree in the graph can connect a subset of terminals to s if the total demands in the subset does not exceed the routing capacity κ. The weight of an edge in a network stands for the cost of installing a copy of the edge. A subset of trees can pass through a single copy of an edge e as long as the number of these trees does not exceed the edge capacity λ; any integer number of copies of e are allowed to be installed. The goal is to find a feasible set of tree-routings that minimizes the total weight of edges installed in the network. We call this problem the *capacitated tree-routing problem* (CTR for short), which can be formally stated as follows, where we denote the vertex set and edge set of a graph G by $V(G)$ and $E(G)$, respectively, and R^+ denotes the set of nonnegative reals.

Capacitated Tree-Routing Problem (CTR):
Input: A graph G, an edge weight function $w : E(G) \to R^+$, a sink $s \in V(G)$, a set $M \subseteq V(G)$ of terminals, a demand function $q : M \to R^+$, a routing capacity $\kappa > 0$, and an integer edge capacity $\lambda \geq 1$.
Feasible solution: A partition $\mathcal{M} = \{Z_1, Z_2, \ldots, Z_\ell\}$ of M and a set $\mathcal{T} = \{T_1, T_2, \ldots, T_\ell\}$ of trees of G such that $Z_i \cup \{s\} \subseteq V(T_i)$ and $\sum_{v \in Z_i} q(v) \leq \kappa$ hold for each i.
Goal: Minimize

$$\sum_{e \in E(G)} h_{\mathcal{T}}(e) w(e),$$

where $h_{\mathcal{T}}(e) = \lceil |\{T \in \mathcal{T} \mid e \in E(T)\}| / \lambda \rceil$, $e \in E$.

The CTR is our new problem formulation which includes several important routing problems as its special cases. First of all, the CTR with $\lambda = 1$ and $\kappa = +\infty$ is equivalent to the *Steiner tree problem*. Given an edge-weighted graph G and a subset $Z \subseteq V(G)$, the Steiner tree problem asks to find a minimum weighted tree T of G with $Z \subseteq V(T)$. The Steiner tree problem is NP-hard, and the current best approximation ratio for the Steiner tree problem is about 1.55 [6].

Secondly the CTR is closely related to the *capacitated network design problem* (CND), which has received a number of attentions in the recent study [2, 4, 7]. The problem is described as follows.

Capacitated Network Design Problem (CND):
Input: A graph G, an edge weight function $w : E(G) \to R^+$, a sink $s \in V(G)$, a set $M \subseteq V(G)$ of sources, a demand function $q : M \to R^+$, and an integer edge capacity $\lambda \geq 1$.
Feasible solution: A set $\mathcal{P} = \{P_v \mid v \in M\}$ of paths of G such that $\{s, v\} \subseteq V(P_v)$ holds for each $v \in M$.
Goal: Minimize

$$\sum_{e \in E(G)} h_{\mathcal{P}}(e) w(e),$$

where $h_{\mathcal{P}}(e) = \lceil \sum_{v : e \in E(P_v)} q(v) / \lambda \rceil$, $e \in E$.

Salman et al. [7] designed a 7-approximation algorithm for the CND by using approximate shortest path trees. Afterwards Hassin et al. [2] gave a $(2+\rho_{\mathrm{ST}})$-approximation algorithm, where ρ_{ST} is any approximation ratio achievable for the Steiner tree problem. By using of a slight intricate version of this algorithm, they improved the approximation ratio to $(1 + \rho_{\mathrm{ST}})$ when every source has unit demand. Note that the CTR and the CND are equivalent in the case where $\kappa = 1$ and $q(v) = 1$ for every $v \in M$.

The third variant of the CTR is the *capacitated multicast tree routing problem* (CMTR) which can be formally stated as follows.

Capacitated Multicast Tree Routing Problem (CMTR):
Input: A graph G, an edge weight function $w : E(G) \to R^+$, a source s, a set $M \subseteq V(G)$ of terminals, a demand function $q : M \to R^+$, and a routing capacity $\kappa > 0$.
Feasible solution: A partition $\mathcal{M} = \{Z_1, Z_2, ..., Z_\ell\}$ of M and a set $\mathcal{T} = \{T_1, T_2, ..., T_\ell\}$ of trees induced on vertices of G such that $Z_i \cup \{s\} \subseteq V(T_i)$ and $\sum_{v \in Z_i} q(v) \leq \kappa$ hold for each i.
Goal: Minimize

$$\sum_{e \in E(G)} h_{\mathcal{T}}(e) w(e) = \sum_{T_i \in \mathcal{T}} w(T_i),$$

where $h_{\mathcal{T}}(e) = |\{T \in \mathcal{T} \mid e \in E(T)\}|$, $e \in E$, and $w(T_i)$ denotes the sum of weights of all edges in T_i.

Observe that the CMTR is equivalent to the CTR with $\lambda = 1$. For the CMTR with a general demand, a $(2+\rho_{\rm ST})$-approximation algorithm is known [3]. If $q(v) = 1$ for all $v \in M$, and κ is a positive integer in an instance of the CMTR, then we call the problem of such instances the *unit demand case* of the CMTR. For the unit demand case of the CMTR, Cai et al. [1] gave a $(2+\rho_{\rm ST})$-approximation algorithm, and Morsy and Nagamochi [5] recently proposed a $(3/2+(4/3)\rho_{\rm ST})$-approximation algorithm.

In this paper, we prove that the CTR admits a $(2+\rho_{\rm ST})$-approximation algorithm. For this, we derive a new result on tree covers in graphs.

The rest of this paper is organized as follows. Section 2 introduces some notations and two lower bounds on the optimal value of the CTR. Section 3 describes some results on tree covers. Section 4 presents our approximation algorithm for the CTR and analyzes its approximation factor. Section 5 concludes.

2 Preliminaries

This section introduces some notations and definitions. Let G be a simple undirected graph. We denote by $V(G)$ and $E(G)$ the sets of vertices and edges in G, respectively. For two subgraphs G_1 and G_2 of a graph G, let $G_1 + G_2$ denote the subgraph induced from G by $E(G_1) \cup E(G_2)$. An edge-weighted graph is a pair (G, w) of a graph G and a nonnegative weight function $w : E(G) \to R^+$. The length of a shortest path between two vertices u and v in (G, w) is denoted by $d_{(G,w)}(u, v)$. Given a vertex weight function $q : V(G) \to R^+$ in G, we denote by $q(Z)$ the sum $\sum_{v \in Z} q(v)$ of weights of all vertices in a subset $Z \subseteq V(G)$.

Let T be a tree. A *subtree* of T is a connected subgraph of T. A set of subtrees in T is called a *tree cover* of T if each vertex in T is contained in at least one of the subtrees. For a subset $X \subseteq V(T)$ of vertices, let $T\langle X\rangle$ denote the minimal subtree of T that contains X (note that $T\langle X\rangle$ is uniquely determined).

Now let T be a rooted tree. We denote by $L(T)$ the set of leaves in T. For a vertex v in T, let $Ch(v)$ and $D(v)$ denote the sets of children and descendants of v, respectively, where $D(v)$ includes v. A *subtree* T_v *rooted* at a vertex v is the subtree induced by $D(v)$, i.e., $T_v = T\langle D(v)\rangle$. For an edge $e = (u, v)$ in a rooted tree T, where $u \in Ch(v)$, the subtree induced by $\{v\} \cup D(u)$ is denoted by T_e, and is called a *branch* of T_v. For a rooted tree T_v, the *depth* of a vertex u in T_v is the length (the number of edges) of the path from v to u.

The rest of this section introduces two lower bounds on the optimal value to the CTR. The first lower bound is based on the Steiner tree problem.

Lemma 1 *Given a* CTR *instance* $I = (G, w, s, M, q, \kappa, \lambda)$, *the minimum cost of a Steiner tree to* $(G, w, M \cup \{s\})$ *is a lower bound on the optimal value to the* CTR *instance* I.

Proof. Consider an optimal solution $(\mathcal{M}^*, \mathcal{T}^*)$ to the CTR instance I. The edge set $E^* = \cup_{T' \in \mathcal{T}^*} E(T') (\subseteq E(G))$ contains a tree T that spans $M \cup \{s\}$ in G. We see that the cost $w(T)$ of T in G is at most that of the CTR solution. Hence the minimum cost of a Steiner tree to $(G, w, M \cup \{s\})$ is no more than the optimal value to the CTR instance I. □

The second lower bound is derived from an observation on the distance from vertices to sink s.

Lemma 2 *Let* $I = (G, w, s, M, q, \kappa, \lambda)$ *be an instance of* CTR. *Then,*

$$\sum_{v \in M} d_{(G,w)}(s, v) q(v) / (\kappa\lambda)$$

is a lower bound on the optimal value to the CTR *instance* I.

Proof. Consider an optimal solution $(\mathcal{M}^* = \{Z_1, \ldots, Z_p\}, \mathcal{T}^* = \{T_1, \ldots, T_p\})$ to the CTR instance I. Let $opt(I) = \sum_{e \in E(G)} h_{\mathcal{T}^*}(e) w(e)$ be the optimal value of the CTR instance I. Since $|\{T' \in \mathcal{T}^* \mid e \in E(T')\}| \leq \lambda h_{\mathcal{T}^*}(e)$ holds for all $e \in E(G)$, we see that

$$\sum_{T_i \in \mathcal{T}^*} w(T_i) \leq \sum_{e \in E(G)} \lambda h_{\mathcal{T}^*}(e) w(e). \quad (1)$$

Also, for each tree $T_i \in \mathcal{T}^*$, we have

$$\sum_{v \in Z_i} d_{(G,w)}(s,v)q(v) \leq w(T_i) \sum_{v \in Z_i} q(v) \leq \kappa w(T_i), \tag{2}$$

since $w(T_i) \geq d_{(G,w)}(s,v)$ for all $v \in V(T_i)$. Hence the proof is completed by summing (2) overall trees in $\mathcal{T}^*$ and using (1). □

3 Tree Cover

The purpose of this section is to present some results on the existence of tree covers, based on which we design our approximation algorithm to the CTR in the next section.

We first review a basic result on tree covers.

Lemma 3 [3] Given a tree T rooted at r, an edge weight function $w : E(T) \to R^+$, a terminal set $M \subseteq V(T)$, a demand function $q : M \to R^+$, and a routing capacity κ with $\kappa \geq q(v)$, $v \in M$, there is a partition $\mathcal{Z} = \mathcal{Z}_1 \cup \mathcal{Z}_2$ of M such that:

(i) For each $Z \in \mathcal{Z}$, there is a child $u \in Ch(r)$ such that $Z \subseteq V(T_u)$. Moreover, $|\{Z \in \mathcal{Z}_1 \mid Z \subseteq V(T_u)\}| \leq 1$ for all $u \in Ch(r)$;

(ii) $q(Z) < \kappa/2$ for all $Z \in \mathcal{Z}_1$;

(iii) $\kappa/2 \leq q(Z) \leq \kappa$ for all $Z \in \mathcal{Z}_2$; and

(iv) Let $\mathcal{T} = \{T\langle Z \cup \{r\}\rangle \mid Z \in \mathcal{Z}_1\} \cup \{T\langle Z\rangle \mid Z \in \mathcal{Z}_2\}$. Then $E(T') \cap E(T'') = \emptyset$ for all $T', T'' \in \mathcal{T}$, and hence $\sum_{T' \in \mathcal{T}} w(T') \leq w(T)$.

Furthermore, such a partition $\mathcal{Z}$ can be obtained in polynomial time. □

From the construction of a partition $\mathcal{Z}$ in Lemma 3, the following corollary is straightforward.

Corollary 1 *Let $\mathcal{Z} = \mathcal{Z}_1 \cup \mathcal{Z}_2$ be defined as in Lemma 3 to (T, r, w, M, q, κ). Then:*

(i) *$E(T\langle Z\rangle) \cap E(T\langle \cup_{Z \in \mathcal{Z}_1} Z\rangle) = \emptyset$ for all $Z \in \mathcal{Z}_2$.*

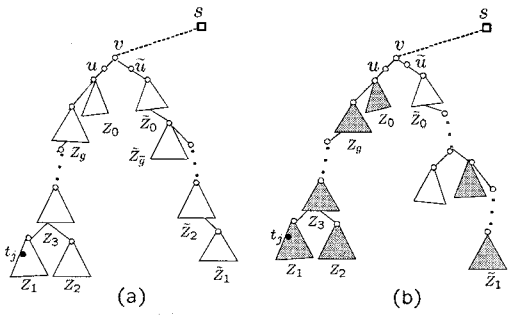

Figure 1: Illustration of the case of $|\mathcal{Z}_2| = g + \tilde{g} \geq \lambda$ in an iteration of algorithm TREECOVER; (a) Line 3.4 identifies a terminal $t_j \in V(T_v)$ with the minimum vertex weight d, where $t_j \in V(T_u)$ in this figure; (b) Line 3.6.3 or 3.6.4 constructs $\mathcal{C}_j$ that contains all subsets in $\{Z_0, Z_1, \ldots, Z_g\}$ and some subsets in $\{\tilde{Z}_1, \ldots, \tilde{Z}_{\tilde{g}}\}$ so that $|\mathcal{C}_j| = \lambda$, where the gray subtrees indicate the subsets in $\mathcal{C}_j$.

(ii) *Let $Z_0 \in \mathcal{Z}_1$ be a subset such that $Z_0 \subseteq V(T_u)$ for some $u \in Ch(r)$. If $\mathcal{Z}' = \{Z \in \mathcal{Z}_2 \mid Z \subseteq V(T_u)\} \neq \emptyset$, then $\mathcal{Z}'$ contains a subset Z' such that $E(T\langle Z_0 \cup Z'\rangle) \cap E(T\langle \mathcal{Z}\rangle) = \emptyset$ for all $Z \in \mathcal{Z} - \{Z_0, Z'\}$.* □

We now describe a new result on tree covers. For an edge weighted tree T rooted at s, a set $M \subseteq V(T)$ of terminals, and a vertex weight function $d : M \to R^+$, we want to find a partition $\mathcal{M}$ of M and to construct a set of induced trees $T\langle Z \cup \{t_Z\}\rangle$, $Z \in \mathcal{M}$ by choosing a vertex $t_Z \in V(T)$ for each subset $Z \in \mathcal{M}$, where we call such a vertex t_Z the *hub vertex* of Z. To find a "good" hub vertex t_Z for each $Z \in \mathcal{M}$, the following lemma classifies a partition $\mathcal{M}$ of M into disjoint collections $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_f$ and then computes t_Z, $Z \in \mathcal{M}$, such that $t_Z = \text{argmin}\{d(t) \mid t \in \cup_{Z \in \mathcal{C}_j} Z\}$ for each $Z \in \mathcal{C}_j$, $j \leq f-1$, and $t_Z = s$ for each $Z \in \mathcal{C}_f$.

Lemma 4 *Given a tree T rooted at s, an edge weight function $w : E(T) \to R^+$, a terminal set $M \subseteq V(T)$, a demand function $q : M \to R^+$, a vertex weight function $d : M \to R^+$, a*

real κ with $\kappa \geq q(v)$, $v \in M$, and a positive integer λ, there exist a partition $\mathcal{M} = \cup_{1 \leq j \leq f} \mathcal{C}_j$ of M, and a set $\mathcal{B} = \{t_j = \mathrm{argmin}\{d(t) \mid t \in \cup_{Z \in \mathcal{C}_j} Z\} \mid j \leq f-1\} \cup \{t_f = s\}$ of hub vertices such that:

(i) $|\mathcal{C}_j| \leq \lambda$ *for all* $j = 1, 2, \ldots, f$;

(ii) $q(Z) \leq \kappa$ *for all* $Z \in \mathcal{C}_j$, $j = 1, 2, \ldots, f$;

(iii) $\sum_{Z \in \mathcal{C}_j} q(Z) \geq \kappa\lambda/2$ *for all* $j = 1, 2, \ldots, f-1$;

(iv) $E(T\langle Z \rangle) \cap E(T\langle Z' \rangle) = \emptyset$ *for all distinct* $Z, Z' \in \mathcal{M}$; *and*

(v) *Let* $\mathcal{T}' = \{T\langle Z \cup \{t_j\} \rangle \mid Z \in \mathcal{C}_j,\ 1 \leq j \leq f\}$, *and let all edges of each* $T\langle Z \cup \{t_j\} \rangle \in \mathcal{T}'$, $Z \in \mathcal{C}_j$, $1 \leq j \leq f$ *be directed toward* t_j. *Then for each edge* $e \in E(T)$, *the number of trees in* $\mathcal{T}'$ *passing through* e *in each direction is at most* λ.

Furthermore, a tuple $(\mathcal{M}, \mathcal{B}, \mathcal{T}')$ can be computed in polynomial time. □

To prove Lemma 4, we can assume without loss of generality that in a given tree T, (i) all terminals are leaves, i.e., $M = L(T)$, and (ii) $|Ch(v)| = 2$ holds for every non-leaf $v \in V(T)$, i.e., T is a binary tree rooted at s, by splitting vertices of degree more than 3 with new edges of weight zero [5]. We prove Lemma 4 by showing that the next algorithm actually delivers a desired tuple $(\mathcal{M}, \mathcal{B}, \mathcal{T}')$. The algorithm constructs collections $\mathcal{C}_1, \mathcal{C}_2, \ldots,$ by repeating a procedure that first chooses a certain vertex v in the current tree, computes a partition $\mathcal{Z}$ of the set of terminals in the subtree rooted at v by Lemma 3, and then selects several subsets in $\mathcal{Z}$ to form the next new collection $\mathcal{C}_j$.

Algorithm TREECOVER
Input: A binary tree $\widehat{T}$ rooted at s, an edge weight function $w : E(\widehat{T}) \to R^+$, a terminal set $M = L(\widehat{T})$ with $q(v) \geq 0$, $v \in M$, a vertex weight function $d : M \to R^+$, a routing capacity κ with $\kappa \geq q(v)$, $v \in M$, and a positive integer λ.
Output: A tuple $(\mathcal{M}, \mathcal{B}, \mathcal{T}')$ that satisfies Conditions (i)-(v) in Lemma 4.
Initialize: $T := \widehat{T}$, $\mathcal{T}' := \emptyset$, and $j := 0$.

1. Choose a maximum depth vertex $v \in V(T)$ with $q(V(T_v) \cap M) \geq \kappa\lambda/2$, and let $j := j + 1$.
2. If v is a leaf of T, then let $Z := \{v\}$, $\mathcal{C}_j := \{Z\}$, and $t_j := v$.
3. If v is not a leaf of T, then
 3.1. Denote $Ch(v) = \{u, \tilde{u}\}$ and $Z_v = V(T_v) \cap M$.
 3.2. Find a partition $\mathcal{Z}_1 \cup \mathcal{Z}_2$ of Z_v by applying Lemma 3 with $(T_v, w, v, Z_v, q, \kappa)$.
 3.3. Denote $\mathcal{Z}_1 = \{Z_0, \widetilde{Z}_0\}$ and $\mathcal{Z}_2 = \{Z_1, \ldots, Z_g\} \cup \{\widetilde{Z}_1, \ldots, \widetilde{Z}_{\tilde{g}}\}$, where $Z_0 \cup Z_1 \cup \cdots \cup Z_g \subseteq V(T_u)$ and $\widetilde{Z}_0 \cup \widetilde{Z}_1 \cup \cdots \cup \widetilde{Z}_{\tilde{g}} \subseteq V(T_{\tilde{u}})$ (see Fig. 1).
 3.4. Let $t_j \in Z_v$ be such that $d(t_j)$ is minimum, where $t_j \in V(T_u)$ w.o.l.g.
 3.5. If $|\mathcal{Z}_2| = |g| + |\tilde{g}| < \lambda$, then let $\mathcal{C}_j := \{Z_0 \cup \widetilde{Z}_0, Z_1, \ldots, Z_g, \widetilde{Z}_1, \ldots, \widetilde{Z}_{\tilde{g}}\}$.
 3.6. If $|\mathcal{Z}_2| \geq \lambda$, then
  3.6.1. Find $Z_b \in \{Z_1, \ldots, Z_g\}$ such that $E(T\langle Z \rangle) \cap E(T\langle Z_0 \cup Z_b \rangle) = \emptyset$ for all $Z \in \mathcal{Z}_1 \cup \mathcal{Z}_2 - \{Z_0, Z_b\}$, by using Corollary 1(ii).
  3.6.2. Let $\tilde{x}_i \in V(T\langle \widetilde{Z}_i \rangle)$, $i = 1, 2, \ldots, \tilde{g}$ be the vertex closest to v in T, where the distance from $\tilde{x}_{i+1}$ to v in T is not larger than that from $\tilde{x}_i$ to v, $1 \leq i \leq \tilde{g} - 1$, w.o.l.g.
  3.6.3. If $q(Z_0 \cup Z_b) \leq \kappa$, then let $\mathcal{C}_j := \{Z_1, \ldots, Z_{b-1}, Z_0 \cup Z_b, Z_{b+1}, \ldots, Z_g\} \cup \{\widetilde{Z}_1, \ldots, \widetilde{Z}_{\lambda-g}\}$.
  3.6.4. If $q(Z_0 \cup Z_b) > \kappa$, then let $\mathcal{C}_j := \{Z_0, Z_1, \ldots, Z_g\} \cup \{\widetilde{Z}_1, \ldots, \widetilde{Z}_{\lambda-g-1}\}$.
4. For each $Z \in \mathcal{C}_j$, let $t_Z := t_j$ and $\mathcal{T}' := \mathcal{T}' \cup \{T\langle Z \cup \{t_Z\} \rangle\}$.
5. Remove the set of terminals in $\mathcal{C}_j$ from M and let $T := T\langle M \cup \{s\} \rangle$.
6. Repeat steps in lines 1-5 with the current tree T as long as $q(M) \geq \kappa\lambda/2$.
7. Let $f := j + 1$, $t_f := s$, and $\mathcal{C}_f := \emptyset$.
8. If M is not empty, then
 8.1. Find a partition $\mathcal{Z}_1 \cup \mathcal{Z}_2$ of M by applying Lemma 3 with (T, w, s, M, q, κ).
 8.2. Let $\mathcal{C}_f := \{Z_0 \cup \widetilde{Z}_0\} \cup \mathcal{Z}_2$, where $\mathcal{Z}_1 = \{Z_0, \widetilde{Z}_0\}$.
9. Let $\mathcal{M} := \cup_{1 \leq j \leq f} \mathcal{C}_j$, $\mathcal{B} := \{t_j \mid 1 \leq j \leq f\}$, and $\mathcal{T}' := \mathcal{T}' \cup \{T\langle Z \cup \{s\} \rangle \mid Z \in \mathcal{C}_f\}$.
10. Output $(\mathcal{M}, \mathcal{B}, \mathcal{T}')$.

Now we prove that the tuple $(\mathcal{M}, \mathcal{B}, \mathcal{T}')$ output from algorithm TREECOVER satisfies Conditions (i)-(v) in Lemma 4.

(i) Clearly, $|\mathcal{C}_j| = 1 \leq \lambda$ for any collection

$\mathcal{C}_j$ computed in line 2. Consider a collection $\mathcal{C}_j$ computed in line 3.5. We have $|\mathcal{C}_j| = g + \tilde{g} + 1 \leq \lambda$ since $g + \tilde{g} < \lambda$. For any collection $\mathcal{C}_j$ computed in line 3.6.3 or 3.6.4, it is easy to see that $|\mathcal{C}_j| = \lambda$ holds. Note that $|\mathcal{Z}_2| < \lambda$ in a partition $\mathcal{Z}_1 \cup \mathcal{Z}_2$ of the current M computed in line 8.1 since $q(Z) \geq \kappa/2$, $Z \in \mathcal{Z}_2$ and $q(M) < \kappa\lambda/2$. Hence $|\mathcal{C}_f| = |\mathcal{Z}_2| + 1 \leq \lambda$ for a collection $\mathcal{C}_f$ computed in line 8.2. This proves (i).

(ii) For a collection $\mathcal{C}_j$ computed in line 2, $q(Z) \leq \kappa$, $Z \in \mathcal{C}_j$, by the assumption that $q(v) \leq \kappa$ for all $v \in M$. Consider a partition $\mathcal{Z}_1 \cup \mathcal{Z}_2$ computed in line 3.2 by applying Lemma 3 to $(T_v, w, v, Z_v, q, \kappa)$. Lemma 3(ii)-(iii) implies that $q(Z_0 \cup \widetilde{Z}_0) \leq \kappa$ and $q(Z) \leq \kappa$ for all $Z \in \mathcal{Z}_2$. Furthermore, for a collection $\mathcal{C}_j$ computed in line 3.6.3, we have $q(Z_0 \cup Z_b) \leq \kappa$. Hence each subset Z added to $\mathcal{C}_j$ in line 3.5, 3.6.3, or 3.6.4 has demand at most κ. Lemma 3(ii)-(iii) implies also that each subset of $\mathcal{C}_f$ computed in line 8.2 has demand at most κ. This proves (ii).

(iii) This condition holds for a collection $\mathcal{C}_j$ computed in line 2 since $q(v) = q(V(T_v) \cap M) \geq \kappa\lambda/2$. Consider a collection $\mathcal{C}_j$ computed in line 3.5. We have $\sum_{Z \in \mathcal{C}_j} q(Z) = \sum_{Z \in \mathcal{Z}_1 \cup \mathcal{Z}_2} q(Z) = q(Z_v) \geq \kappa\lambda/2$ since $\mathcal{Z}_1 \cup \mathcal{Z}_2$ computed in line 3.2 is a partition of Z_v and $q(Z_v) \geq \kappa\lambda/2$ by using the condition in line 1. For a collection $\mathcal{C}_j$ computed in line 3.6.3, Lemma 3(iii) implies that $\sum_{Z \in \mathcal{C}_j} q(Z) \geq \lambda(\kappa/2)$ since $q(Z) \geq \kappa/2$, $Z \in \mathcal{C}_j$. For a collection $\mathcal{C}_j$ computed in line 3.6.4, we have $\sum_{Z \in \mathcal{C}_j} q(Z) = \sum_{1 \leq i \leq b-1} q(Z_i) + q(Z_0 \cup Z_b) + \sum_{b+1 \leq i \leq g} q(Z_i) + \sum_{1 \leq i \leq \lambda-g-1} q(\widetilde{Z}_i) > (b-1)\kappa/2 + \kappa + ((g-b) + (\lambda - g - 1))\kappa/2 = \kappa\lambda/2$ since $q(Z_0 \cup Z_b) > \kappa$. This completes the proof of property (iii).

(iv) Consider the execution of the jth iteration of the algorithm. By the construction of $\mathcal{C}_j$ and Lemma 3(iv), we have $E(T\langle Z_1 \rangle) \cap E(T\langle Z_2 \rangle) = \emptyset$ for all distinct $Z_1, Z_2 \in \mathcal{C}_j$. Moreover, since any collection computed in line 2, 3.5, or 8.2 contains all subsets in a partition $\mathcal{Z}_1 \cup \mathcal{Z}_2$ of Z_v computed in line 3.2 and by the assumption in line 3.6.2 used in constructing $\mathcal{C}_j$ in line 3.6.3 or 3.6.4, we conclude that $E(T\langle Z' \rangle) \cap E(T\langle M - \cup_{Z \in \mathcal{C}_j} Z \rangle)) = \emptyset$ for all $Z' \in \mathcal{C}_j$. Hence for any distinct subsets $Z_1, Z_2 \in \mathcal{M}$, we have $E(\widehat{T}\langle Z_1 \rangle) \cap E(\widehat{T}\langle Z_2 \rangle) = \emptyset$ since a partition $\mathcal{M}$ of M output from the algorithm is a union of collections $\mathcal{C}_j$, $j = 1, 2, \ldots, f$. This proves (iv).

Before proving the property (v), we can show the following lemma (the proof is omitted due to space limination).

Lemma 5 *Let $(\mathcal{M}, \mathcal{B}, \mathcal{T}')$ be a tuple obtained from a binary tree $\widehat{T}$ by algorithm* TREECOVER. *Then for each edge $e = (x, y) \in E(\widehat{T})$, where $y \in Ch_{\widehat{T}}(x)$, we have*

(i) *For $\mathcal{T}'_1 = \{\widehat{T}\langle Z \cup \{t_Z\} \rangle \in \mathcal{T}' \mid Z \in \mathcal{M}, Z \cap V(\widehat{T}_y) \neq \emptyset \neq Z \cap (V(\widehat{T}) - V(\widehat{T}_y))\}$, it holds $|\mathcal{T}'_1| \leq 1$;*

(ii) *$|\{\widehat{T}\langle Z \cup \{t_Z\} \rangle \in \mathcal{T}' \mid Z \in \mathcal{M}, Z \subseteq V(\widehat{T}) - V(\widehat{T}_y), t_Z \in V(\widehat{T}_y)\}| \leq \lambda - 1$; and*

(iii) *$|\{\widehat{T}\langle Z \cup \{t_Z\} \rangle \in \mathcal{T}' \mid Z \in \mathcal{M}, Z \subseteq V(\widehat{T}_y), t_Z \in V(\widehat{T}) - V(\widehat{T}_y)\}| \leq \lambda - |\mathcal{T}'_1|$.* □

We are ready to prove property (v) in Lemma 4. Let $e = (x, y)$ be an arbitrary edge of $\widehat{T}$, where $y \in Ch_{\widehat{T}}(x)$. Let all edges of $\widehat{T}\langle Z \cup t_Z \rangle \in \mathcal{T}'$, $Z \in \mathcal{M}$, be directed toward t_Z, and let $\mathcal{T}'_1$ be as defined in Lemma 5. The number of trees in $\mathcal{T}'$ passing through e toward y is at most the sum of the number of trees in $\mathcal{T}'_1$ and trees in $\{\widehat{T}\langle Z \cup \{t_Z\} \rangle \in \mathcal{T}' \mid Z \in \mathcal{M}, Z \subseteq V(\widehat{T}) - V(\widehat{T}_y), t_Z \in V(\widehat{T}_y)\}$. Similarly, the number of trees in $\mathcal{T}'$ passing through e toward x is at most the sum of the number of subsets in $\mathcal{T}'_1$ and trees in $\{\widehat{T}\langle Z \cup \{t_Z\} \rangle \in \mathcal{T}' \mid Z \in \mathcal{M}, Z \subseteq V(\widehat{T}_y), t_Z \in V(\widehat{T}) - V(\widehat{T}_y)\}$. Hence Lemma 5(i)-(iii) completes the proof of (v). □

4 Approximation Algorithm to CTR

This section presents an approximation algorithm for an instance $I = (G, w, s, M, q, \kappa, \lambda)$ of the CTR problem based on results on tree covers in the previous section. The basic idea of the algorithm is to compute an approximate Steiner tree T in $(G, w, M \cup \{s\})$, find a tree cover $\mathcal{T}'$ of the tree T such that $|\{T' \in \mathcal{T}' \mid$

$e \in E(T')\}| \leq \lambda$ for each $e \in E(T)$, and finally connect each tree in $\mathcal{T}'$ to s in order to get a tree-routings $\mathcal{T}$ in the instance I.

Algorithm APPROXCTR
Input: An instance $I = (G, w, s, M, q, \kappa, \lambda)$ of the CTR.
Output: A solution $(\mathcal{M}, \mathcal{T})$ to I.

Step 1. Compute a ρ_{ST}-approximate solution T to the Steiner tree problem in G that spans $M \cup \{s\}$ and then regard T as a tree rooted at s.
Define a function $d : M \longrightarrow R^+$ by setting

$$d(t) := d_{(G,w)}(s,t), \quad t \in M.$$

Step 2. Apply Lemma 4 to $(T, w, s, M, q, d, \kappa, \lambda)$ to get a partition $\mathcal{M} = \cup_{1 \leq j \leq f} \mathcal{C}_j$ of M, a set $\mathcal{B} = \{t_1, t_2, \ldots, t_f\}$ of hub vertices, where $t_Z = t_j$ for each $Z \in \mathcal{C}_j$, $j = 1, 2, \ldots, f$, and a set $\mathcal{T}' = \{T\langle Z \cup \{t_Z\}\rangle \mid Z \in \mathcal{M}\}$ of subtrees of T that satisfy Conditions (i)-(v) of the lemma.

Step 3. For each edge $e = (u, v)$ of T, $v \in Ch_T(u)$, with $|\{T' \in \mathcal{T}' \mid e \in E(T')\}| > \lambda$,
Define $\mathcal{C}_{in}(e) := \{Z \in \mathcal{M} \mid Z \subseteq V(T) - V(T_v), t_Z \in V(T_v)\}$ and $\mathcal{C}_{out}(e) := \{Z \in \mathcal{M} \mid Z \subseteq V(T_v), t_Z \in V(T) - V(T_v)\}$.

while $|\{T' \in \mathcal{T}' \mid e \in E(T')\}| > \lambda$ **do**
Choose two arbitrary subsets $Z \in \mathcal{C}_{in}(e)$ and $Z' \in \mathcal{C}_{out}(e)$, where $Z \in \mathcal{C}_j$ and $Z' \in \mathcal{C}_{j'}$, $1 \leq j, j' \leq f$.
Let $\mathcal{C}_j := (\mathcal{C}_j - \{Z\}) \cup \{Z'\}$, $\mathcal{C}_{j'} := (\mathcal{C}_{j'} - \{Z'\}) \cup \{Z\}$, $t_Z := t_{j'}$, and $t_{Z'} := t_j$.
Let $\mathcal{C}_{in}(e) := \mathcal{C}_{in}(e) - \{Z\}$ and $\mathcal{C}_{out}(e) := \mathcal{C}_{out}(e) - \{Z'\}$.
Let $\mathcal{T}' := (\mathcal{T}' - \{T\langle Z \cup \{t_j\}\rangle, T\langle Z' \cup \{t_{j'}\}\rangle\}) \cup \{T\langle Z \cup \{t_Z\}\rangle, T\langle Z' \cup \{t_{Z'}\}\rangle\}$.

Step 4. For each $j = 1, 2, \ldots, f-1$, choose a shortest path $SP(s, t_j)$ between s and t_j in (G, w) and join t_j to s by installing a copy of each edge in $SP(s, t_j)$.
Let $\mathcal{T} := \{T\langle Z \cup \{t_Z\}\rangle + SP(s, t_Z) \mid Z \in \mathcal{M}\}$ and output $(\mathcal{M}, \mathcal{T})$.

The idea of Step 3 in APPROXCTR is originated from a procedure of swapping paths in the algorithm for the CND due to Hassin et al. [2].

We state the following lemma without proof due to space limitation.

Lemma 6 *Let $\mathcal{T}'$ be output by algorithm* APPROXCTR *applied to an instance $I = (G, w, s, M, q, \kappa, \lambda)$ of the* CTR *problem. Then for any edge e of the Steiner tree T, we have $|\{T\langle Z \cup \{t_Z\}\rangle \in \mathcal{T}' \mid Z \in \mathcal{M}, e \in E(T\langle Z \cup \{t_Z\}\rangle)\}| \leq \lambda$.* □

Next we show the feasibility and compute the approximation factor of the approximate solution $(\mathcal{M}, \mathcal{T})$ output from algorithm APPROXCTR.

Theorem 1 *For an instance $I = (G, w, s, M, q, \kappa, \lambda)$ of the* CTR, *algorithm* APPROXCTR *delivers a $(2+\rho_{\mathrm{ST}})$-approximate solution $(\mathcal{M}, \mathcal{T})$, where ρ_{ST} is the approximation ratio of solution T to the Steiner tree problem.*

Proof. Since $\mathcal{M} = \cup_{1 \leq j \leq f} \mathcal{C}_j$, Lemma 4(ii) implies that $q(Z) \leq \kappa$ for all $Z \in \mathcal{M}$. That is, $(\mathcal{M}, \mathcal{T})$ satisfies the routing capacity constraint on each tree. Now we show that $\mathcal{T}$ satisfies the edge capacity constraint, that is, $|\{T' \in \mathcal{T} \mid e \in E(T')\}| \leq \lambda h_T(e)$ for any $e \in E(G)$. Note that each tree in $\mathcal{T}$ is a tree $T\langle Z \cup \{t_Z\}\rangle \in \mathcal{T}'$, $Z \in \mathcal{M}$, plus the shortest path $SP(s, t_Z)$ between s and t_Z in (G, w). By Lemma 6, installing one copy on each edge of the Steiner tree T implies that $|\{T\langle Z \cup \{t_Z\}\rangle \in \mathcal{T}' \mid Z \in \mathcal{M}, e \in E(T\langle Z \cup \{t_Z\}\rangle)\}| \leq \lambda$ for any $e \in E(T)$. On the other hand, each collection $\mathcal{C}_j$, $j \leq f$, contains at most λ subsets of $\mathcal{M}$, all of which are assigned to a common hub vertex t_j. Hence it is enough to install one copy of each edge in a shortest path $SP(s, t_j)$ between s and t_j in (G, w), $j \leq f-1$ ($t_f = s$), in order to get a feasible set $\mathcal{T}$ of tree-routings. This implies that the number of trees in $\mathcal{T}$ passing through a copy of each installed edge on the network is at most λ. Thereby $(\mathcal{M}, \mathcal{T})$ is feasible to I and the total weight of the installed edges on the network is bounded by

$$w(T) + \sum_{1 \leq j \leq f-1} d(t_j).$$

For a minimum Steiner tree T^* that spans $M \cup \{s\}$, we have $w(T) \leq \rho_{ST} \cdot w(T^*)$ and $w(T^*) \leq opt(I)$ by Lemma 1, where $opt(I)$ denotes the weight of an optimal solution to the CTR. Hence $w(T) \leq \rho_{ST} \cdot opt(I)$ holds. To prove the theorem, it suffices to show that

$$\sum_{1 \leq j \leq f-1} d(t_j) \leq 2opt(I). \tag{3}$$

Consider a collection $\mathcal{C}_j$, $j \leq f-1$ obtained by applying Lemma 4 to $(T, w, s, M, q, d, \kappa, \lambda)$ in Step 2. Note that even if some subsets of $\mathcal{C}_j$ are applied by swapping in Step 3, the hub vertex of the updated collection remains unchanged. That is, the set $\mathcal{B}$ of hub vertices computed in Step 2 does not change throughout the algorithm. Hence Lemma 4(iii) implies that

$$\sum_{t \in Z \in \mathcal{C}_j} q(t)d(t) \geq d(t_j) \sum_{t \in Z \in \mathcal{C}_j} q(t) \geq (\kappa\lambda/2)d(t_j). \tag{4}$$

By summing inequality (4) overall $\mathcal{C}_j$'s (computed in Step 2), $j \leq f-1$, we have

$$\frac{1}{2} \sum_{1 \leq j \leq f-1} d(t_j) \leq \sum_{1 \leq j \leq f-1} \sum_{t \in Z \in \mathcal{C}_j} (q(t)/(\kappa\lambda))d(t)$$

$$\leq \sum_{t \in M} (q(t)/(\kappa\lambda))d(t).$$

Hence Lemma 2 completes the proof of (3). □

5 Conclusion

In this paper, we have studied the capacitated tree-routing problem (CTR), a new routing problem formulation under a multi-tree model which unifies several important routing problems such as the capacitated network design problem (CND) and the capacitated multicast tree routing problem (CMTR). We have proved that the CTR is $(2+\rho_{ST})$-approximable based on some new results on tree covers, where ρ_{ST} is any approximation factor achievable for the Steiner tree problem. Future work may include design of approximation algorithms for further extensions of our tree-routing model.

References

[1] Z. Cai, G.-H Lin, and G. Xue, Improved approximation algorithms for the capacitated multicast routing problem, LNCS 3595, (2005) 136–145.

[2] R. Hassin, R. Ravi, and F. S. Salman, Approximation algorithms for a capacitated network design problem, Algorithmica, 38, (2004) 417–431.

[3] R. Jothi and B. Raghavachari, Approximation algorithms for the capacitated minimum spanning tree problem and its variants in network design, In proceedings of ICALP 2004, LNCS 3142, (2004) 805–818.

[4] Y. Mansour and D. Peleg, An approximation algorithm for minimum-cost network design, Tech. Report Cs94-22, The Weizman Institute of Science, Rehovot, (1994); also presented at the DIMACS Workshop on Robust Communication Network, (1998).

[5] E. Morsy and H. Nagamochi, An improved approximation algorithm for capacitated multicast routings in networks, In Proceedings of International Symposium on Scheduling 2006, Tokyo, Japan, (2006) 12–17.

[6] G. Robins and A. Z. Zelikovsky, Improved Steiner tree approximation in graphs, In Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms-SODA'2000, (2000) 770–779.

[7] F. S. Salman, J. Cheriyan, R. Ravi, and S. Subramanian, Approximating the single-sink link-installation problem in network design, SIAM J. Optim., 11, (2000) 595–610.