Approximating the Generalized Capacitated Tree-routing Problem

Ehab Morsy, Hiroshi Nagamochi

Department of Applied Mathematics and Physics
Graduate School of Informatics
Kyoto University
Yoshida Honmachi, Sakyo, Kyoto 606-8501, Japan
{ehab,nag}@amp.i.kyoto-u.ac.jp

Abstract

In this paper, we introduce the *generalized capacitated tree-routing problem* (GCTR), which is described as follows. Given a connected graph $G = (V, E)$ with a sink $s \in V$ and a set $M \subseteq V - \{s\}$ of terminals with a nonnegative demand $q(v)$, $v \in M$, we wish to find a collection $\mathcal{T} = \{T_1, T_2, \ldots, T_\ell\}$ of trees rooted at s to send all the demands to s, where the total demand collected by each tree T_i is bounded from above by a demand capacity $\kappa > 0$. Let $\lambda > 0$ denote a bulk capacity of an edge, and each edge $e \in E$ has an installation cost $w(e) \geq 0$ per bulk capacity; each edge e is allowed to have capacity $k\lambda$ for any integer k, which installation incurs cost $kw(e)$. To establish a tree routing T_i, each edge e contained in T_i requires $\alpha+\beta q'$ amount of capacity for the total demand q' that passes through edge e along T_i and prescribed constants $\alpha, \beta \geq 0$, where α means a fixed amount used to separate the inside of the routing T_i from the outside while term $\beta q'$ means the net capacity proportional to q'. The objective of GCTR is to find a collection $\mathcal{T}$ of trees that minimizes the total installation cost of edges. Then GCTR is a new generalization of the several known multicast problems in networks with edge/demand capacities. In this paper, we prove that GCTR is $(2\lceil\lambda/(\alpha + \beta\kappa)\rceil/\lfloor\lambda/(\alpha + \beta\kappa)\rfloor + \rho_{\rm ST})$-approximable if $\lambda \geq \alpha + \beta\kappa$ holds, where $\rho_{\rm ST}$ is any approximation ratio achievable for the Steiner tree problem.

Key words: Approximation Algorithm, Graph Algorithm, Routing Problems, Network Optimization, Tree Cover.

1 Introduction

In this paper, we introduce the *generalized capacitated tree-routing problem* (GCTR), which is described as follows. Given a connected graph $G = (V, E)$ with a demand capacity $\kappa > 0$, a bulk edge capacity $\lambda > 0$, a sink $s \in V$, and a set $M \subseteq V - \{s\}$ of terminals with a nonnegative demand $q(v)$, $v \in M$, we wish to find a collection

* Technical report 2008-001, March 21, 2008

$\mathcal{T} = \{T_1, T_2, \ldots, T_\ell\}$ of trees rooted at s to send all the demands to s, where the total demand in the set Z_i of terminals assigned to tree T_i does not exceed the demand capacity κ. Each edge $e \in E$ has an installation cost $w(e) \geq 0$ per bulk capacity; each edge e is allowed to have capacity $k\lambda$ for any integer k, which requires installation cost $kw(e)$. To establish a tree routing T_i through an edge e, we assume that e needs to have capacity at least

$$\alpha + \beta \sum_{v \in Z_i \cap D_{T_i}(v_i^e)} q(v)$$

for prescribed coefficients $\alpha, \beta \geq 0$, where v_i^e is the tail of e in T_i and $D_{T_i}(v_i^e)$ denotes the set of descendants of v_i^e in T_i including v_i^e; α means a fixed amount used to separate the inside and outside of the routing T_i while term $\beta \sum_{v \in Z_i \cap D_{T_i}(v_i^e)} q(v)$ means the net capacity proportional to the amount $\sum_{v \in Z_i \cap D_{T_i}(v_i^e)} q(v)$ of demands that passes through edge e along T_i. Hence, given a set $\mathcal{T} = \{T_1, T_2, \ldots, T_\ell\}$ of trees, each edge e needs to have capacity $k_e\lambda$ for the least integer k_e such that

$$\sum_{T_i \in \mathcal{T} : T_i \text{ contains } e} [\alpha + \beta \sum_{v \in Z_i \cap D_{T_i}(v_i^e)} q(v)] \leq k_e \lambda,$$

and the total installation cost of edges incurred by $\mathcal{T}$ is given as $\sum_{e \in E} k_e w(e)$, where $k_e = 0$ if no $T_i \in \mathcal{T}$ contains e. The objective of GCTR is to find a set $\mathcal{T}$ of trees that minimizes the total installation cost of edges. We formally state GCTR 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.

Generalized Capacitated Tree-Routing Problem (GCTR):
Input: A graph G, an edge weight function $w : E(G) \to R^+$, a sink $s \in V(G)$, a set $M \subseteq V(G) - \{s\}$ of terminals, a demand function $q : M \to R^+$, a demand capacity $\kappa > 0$, an edge capacity $\lambda > 0$, and prescribed constants $\alpha, \beta \geq 0$.
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 the total installation cost of $\mathcal{T}$, that is,

$$\sum_{e \in E(G)} \lceil \sum_{T_i : e \in E(T_i)} (\alpha + \beta \sum_{v \in Z_i \cap D_{T_i}(v_i^e)} q(v))/\lambda \rceil w(e),$$

where v_i^e is the tail of e in T_i and $D_{T_i}(v_i^e)$ denotes the set of descendants of v_i^e in T_i including v_i^e.

We have a variant of GCTR if it is allowed to purchase edge capacity in any required quantity. In this model, for each edge e of the underlying network, we assign capacity of $\lambda_e = \alpha|\mathcal{T}'| + \beta \sum_{T_i \in \mathcal{T}'} \sum_{v \in Z_i \cap D_{T_i}(v_i^e)} q(v)$ on e, where $\mathcal{T}'$ is the set of trees containing e. That is, the total cost of the constructed trees equals $\sum_{e \in E} \lambda_e w(e)$. We call this variant of GCTR, *the fractional generalized capacitated tree-routing problem* (FGCTR).

We easily see that GCTR and FGCTR contain two classical NP-hard problems, the *Steiner tree problem* and the *bin packing problem* [6]. We see that GCTR with an edge weighted graph G, $\alpha = \lambda = 1$, and $\beta = 0$ is equivalent to the Steiner tree problem in G when $\kappa \geq \sum_{v \in M} q(v)$, and is equivalent to the bin packing problem with bin size κ when G is a complete graph, $w(e) = 1$ for all edges e incident to s and $w(e) = 0$ otherwise. We see that FGCTR also has a similar relationship with the Steiner tree problem and the bin packing problem. The Steiner tree problem is known to be NP-hard even if it has Euclidean or rectilinear costs [5]. A series of approximation algorithms for the Steiner tree problem have been developed over the last two decades [3,8,10,16,17,19,21]. The best known approximation factor for the Steiner tree problem is 1.55 [17].

The characteristic of GCTR and FGCTR is their routing capacity which is a linear combination of the number of trees and the total amount of demands that pass through an edge. Such a general form of capacity constraint can be found in some applications.

Suppose that we wish to find a minimum number of trucks to carry given n items $v_1, v_2, \ldots, v_n$, where each item v_i has size $q(v_i)$ and weight $\beta q(v_i)$, where β is a specific gravity. We also have bins; the weight of a bin is α and the capacity of a bin is κ. Items are first put into several bins, and then the bins are assigned to trucks under capacity constraints. That is, we can put items in a bin B so that the total size $\sum_{v_i \in B} q(v_i)$ of the items does not exceed the bin capacity κ, where the weight of the bin B is given by a linear combination $\alpha + \beta \sum_{v_i \in B} q(v_i)$. We can load packed bins into a truck as long as the total weight of these packed bins does not exceed the truck capacity λ. The objective is to find assignments of items to bins and packed bins to trucks such that the number of required trucks is minimized. This problem can be described as GCTR.

Suppose that a petroleum corporation wishes to construct a network of pipelines to collect raw oil from several locations to a set of storage stations (to be specified among all locations), each of which has a specified demand capacity, and then send the oil from these storage stations to a specified major refinery. Moreover, for the sake of efficiency, the corporation staff wants to construct a set of trees that spans all locations, each of which contains a storage station. A single pipe type with a specified bulk capacity is available. For each edge of the underlying pipe network, it is allowed to install either zero or an integer number of pipes, where each pipe has a nonnegative construction cost. A part of pipe capacity is used to protect the internal surface of the pipe, while the rest of the pipe capacity needs to be proportional to the amount of oil that goes through the pipe. Therefore, the required amount of capacity of edge is given as a linear combination of the number of trees that and the total demand pass through the edge. The goal of the corporation is to construct the cheapest possible set of feasible tree-routings so that the demands of all locations can be routed simultaneously to the refinery without violating the capacity constraint.

Another application can be found in a video delivery system in a computer network. We are given a graph $G = (V, E)$ with a set V of nodes, a set E of links, a cost function $w : E \to R^+$, and a link bandwidth $\lambda > 0$. We have a service center $s \in V$ and a set $M \subseteq V$ of clients (terminals) with demands $q : M \to R^+$. The service center s actually consists of a large number of servers, each can serve at most κ demands from clients that are assigned to it. Notice that, if we use IP multicast (see [22] for the detail), then for each server and its clients, the routing subgraph connecting them must be a tree. Suppose we can install as many links as we can. Then the problem is to find an assignment of clients to servers that minimizes the total link installation cost without violating the capacity of every server and the bandwidth of every link, where the latter is considered as a linear combination of the traffic due to the routing (the number of servers using the link) and the data communication (the total data going through the link).

Similar routing problems in which the objective function is a linear combination of two or more optimization requirements have been studied before [1,2,20]. For example, given a *lattice graph* with an edge capacity and a vertex cost function, *the global routing problem in VLSI design* asks to construct a set of trees that spans a given set of *nets* (subsets of the vertex set) under an edge capacity constraint. Terlaky et al. [20] have studied a problem of minimizing an objective function which is defined as a linear combination of the total edge cost and the total number of bends of all trees, where a bend at a vertex corresponds a via in VLSI design, which leads to extra cost in manufacturing.

We here observe that our new problem formulation, GCTR, includes several important routing problems as its special cases.

Firstly, GCTR is closely related to the *capacitated network design problem* (CND), which has received a number of attentions in the recent study [7,13,18]. 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 $M \subseteq V(G) - \{s\}$ 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 the sum of weights of edges to be installed, that is,

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

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

Salman et al. [18] designed a 7-approximation algorithm for CND by using approximate shortest path trees defined in [11] to route demands to the sink. Afterwards Hassin et al. [7] gave a $(2 + \rho_{\mathrm{ST}})$-approximation algorithm, where ρ_{ST} is any approximation ratio achievable for the Steiner tree problem. By designing 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 GCTR and CND are equivalent in the case where $\alpha = 0$, $\beta = 1$, and $\kappa = \lambda$.

The second special case of GCTR 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 $s \in V(G)$, a set $M \subseteq V(G) - \{s\}$ of terminals, a demand function $q : M \to R^+$, and a demand 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 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 CMTR is equivalent to GCTR with $\alpha = 1$, $\beta = 0$, and $\lambda = 1$. CMTR also has received a number of attentions in the recent study [4,9,12,14]. For CMTR with a general demand, a $(2 + \rho_{\rm ST})$-approximation algorithm is known [9]. If $q(v) = 1$ for all $v \in M$, and κ is a positive integer in an instance of CMTR, then we call the problem of such instances the *unit demand case* of CMTR. For the unit demand case of CMTR, Cai et al. [4] gave a $(2 + \rho_{\rm ST})$-approximation algorithm, and Morsy and Nagamochi [14] recently proposed a $(3/2 + (4/3)\rho_{\rm ST})$-approximation algorithm.

Finally, we observe that GCTR generalizes the *capacitated tree-routing problem* (CTR) proposed recently in [15]. The problem can be formulated as follows.

Capacitated Tree-Routing Problem (CTR):
Input: A graph G, an edge weight function $w : E(G) \rightarrow R^+$, a sink $s \in V(G)$, a set $M \subseteq V(G) - \{s\}$ of terminals, a demand function $q : M \rightarrow R^+$, a demand 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 the sum of weights of edges to be installed under the edge capacity constraint, that is,

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

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

Note that CMTR (resp., CND) is equivalent to CTR in the case where $\lambda = 1$ (resp., $\kappa = 1$ and $q(v) = 1$ for every $v \in M$). On the other hand, CTR is equivalent to GCTR with $\alpha = 1$ and $\beta = 0$. Thus, the integer edge capacity in CTR represents the number of trees allowed to contain a copy of the edge. Recently, Morsy and Nagamochi [15] designed a $(2 + \rho_{\rm ST})$-approximation algorithm for CTR.

As observed above, GCTR is a considerably general model for routing problems. In this paper, we prove that GCTR admits a $(2\lceil\lambda/(\alpha+\beta\kappa)\rceil/\lfloor\lambda/(\alpha+\beta\kappa)\rfloor + \rho_{\rm ST})$-approximation algorithm if $\lambda \geq \alpha + \beta\kappa$ holds. The high-level description of the proposed algorithm resembles our algorithm for CTR problem [15], but we need to derive a new lower bound to the problem. Namely, given an instance $I = (G, w, s, M, q, \alpha, \beta, \kappa, \lambda)$ of GCTR, the main idea of our algorithm is to compute an integer capacity λ' depending on λ, κ, α, and β and then find a feasible tree-routings solution to the instance $I' = (G, w, s, M, q, \kappa, \lambda')$ of CTR. Here such capacity λ' is chosen so that this set of tree-routings is a feasible solution to the original GCTR instance I.

Table 1
Approximation algorithms for CND, CMTR, CTR, and GCTR problems, where $\theta = \lceil \lambda/(\alpha + \beta\kappa) \rceil / \lfloor \lambda/(\alpha + \beta\kappa) \rfloor$.

Problem		unit demands $q \equiv 1$	general demands $q \geq 0$
CND	$\alpha = 0, \beta = 1$, $\kappa = \lambda \in R^+$	$1 + \rho_{\mathrm{ST}}$ [7]	$2 + \rho_{\mathrm{ST}}$ [7]
CMTR	$\alpha = 1, \beta = 0$, $\lambda = 1, \kappa \in R^+$	$2 + \rho_{\mathrm{ST}}$ [4], $3/2 + (4/3)\rho_{\mathrm{ST}}$ [14]	$2 + \rho_{\mathrm{ST}}$ [9]
CTR	$\alpha = 1, \beta = 0$ $\lambda, \kappa \in R^+$	$2 + \rho_{\mathrm{ST}}$ [15]	$2 + \rho_{\mathrm{ST}}$ [15]
GCTR	$\alpha, \beta, \kappa, \lambda \in R^+$ with $\lambda \geq \alpha + \beta\kappa$	$2\theta + \rho_{\mathrm{ST}}$ [this paper]	$2\theta + \rho_{\mathrm{ST}}$ [this paper]

We can show that, with a slight modification, the approximation algorithm proposed for GCTR delivers a $(\alpha + \beta\kappa)(2 + \rho_{\mathrm{ST}})$-approximate solution to FGCTR (the details is omitted due to space limitation).

Table 1 shows a summary of the recent approximation algorithms for CND, CMTR, CTR, and GCTR.

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

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 GCTR. The first lower bound is based on the Steiner tree problem.

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

Proof. Consider an optimal solution $(\mathcal{M}^*, \mathcal{T}^*)$ to GCTR instance I. Let $E^* = \cup_{T' \in \mathcal{T}^*} E(T')$ $(\subseteq E(G))$, i.e., the set of all edges used in the optimal solution. Then the edge set E^* 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 GCTR solution. Hence the minimum cost of a Steiner tree to $(G, w, M \cup \{s\})$ is no more than the optimal value to GCTR instance I. $\Box$

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

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

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

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

Proof. Consider an optimal solution $(\mathcal{M}^* = \{Z_1, \ldots, Z_p\}, \mathcal{T}^* = \{T_1, \ldots, T_p\})$ to GCTR instance I. For each edge $e \in E(T_i)$, $i = 1, 2, \ldots, p$, we assume that $e = (u_i^e, v_i^e)$, where $v_i^e \in Ch_{T_i}(u_i^e)$. Let *opt*$(I)$ denote the optimal value of GCTR instance I. Then we have

$$\begin{aligned}
opt(I) &= \sum_{e \in E(G)} \left\lceil [\alpha|\{T_i \mid e \in E(T_i)\}| + \beta \sum_{T_i : e \in E(T_i)} q(Z_i \cap D_{T_i}(v_i^e))]/\lambda \right\rceil w(e) \\
&\geq \sum_{e \in E(G)} w(e)[\alpha|\{T_i \mid e \in E(T_i)\}| + \beta \sum_{T_i : e \in E(T_i)} q(Z_i \cap D_{T_i}(v_i^e))]/\lambda \\
&= (\alpha/\lambda) \sum_{e \in E(G)} |\{T_i \mid e \in E(T_i)\}| w(e) \\
&\quad + (\beta/\lambda) \sum_{e \in E(G)} w(e) \sum_{T_i : e \in E(T_i)} q(Z_i \cap D_{T_i}(v_i^e)) \\
&= (\alpha/\lambda) \sum_{T_i \in \mathcal{T}^*} w(T_i) + (\beta/\lambda) \sum_{T_i \in \mathcal{T}^*} \sum_{e \in E(T_i)} q(Z_i \cap D_{T_i}(v_i^e)) w(e). \qquad (1)
\end{aligned}$$

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

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

since $w(T_i) \geq d_{(G,w)}(s, v)$ for all $v \in V(T_i)$. On the other hand, for each tree $T_i \in \mathcal{T}^*$, we have

$$\sum_{e\in E(T_i)} q(Z_i \cap D_{T_i}(v_i^e))w(e) = \sum_{v\in Z_i} q(v)d_{(T_i,w)}(s,v) \geq \sum_{v\in Z_i} q(v)d_{(G,w)}(s,v). \qquad (3)$$

Hence by summing (2) and (3) overall trees in $\mathcal{T}^*$ and substituting in (1), we conclude that

$$(\alpha+\beta\kappa)/(\lambda\kappa)\sum_{v\in M} q(v)d_{(G,w)}(s,v) \leq opt(I),$$

which completes the proof. $\square$

3 Tree Cover

This section is devoted to present some results on the existence of tree covers, based on which we design our approximation algorithm to GCTR in the next section.

We first review a basic result on tree covers.

Lemma 3 [9] *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 demand capacity κ with $\kappa \geq max\{q(v) \mid v \in M\}$, 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 $E(T_1) \cap E(T_2) = \emptyset$ for all distinct trees $T_1, T_2 \in \mathcal{T}$.*

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

The following corollary is an immediate consequence of the particular construction of a partition $\mathcal{Z}$ in Lemma 3.

Corollary 4 [15] *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 $Z \in \mathcal{Z}_2$.*
(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 $E(T\langle Z_0 \cup Z'\rangle) \cap E(T\langle Z\rangle) = \emptyset$ for all $Z \in \mathcal{Z} - \{Z_0, Z'\}$.* $\square$

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 wish 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 $Z \in \mathcal{M}$, we classify a partition $\mathcal{M}$ of M into disjoint collections $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_f$ and then choose hub vertices t_Z, $Z \in \mathcal{M}$, such that $t_Z = t_j \in \{\mathrm{argmin}_{t \in Z \in \mathcal{C}_j} d(t)\}$ for each $Z \in \mathcal{C}_j$, $j \leq f-1$, and $t_Z = s$ for each $Z \in \mathcal{C}_f$, as shown in the next lemma.

Lemma 5 *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 demand capacity κ with $\kappa \geq max\{q(v) \mid v \in M\}$, an edge capacity $\lambda > 0$, and prescribed constants $\alpha, \beta \geq 0$ with $\lambda \geq \alpha + \beta\kappa$, there exist a partition $\mathcal{M} = \cup_{1 \leq j \leq f} \mathcal{C}_j$ of M, and a set $\mathcal{B} = \{t_j \in \{\mathrm{argmin}_{t \in Z \in \mathcal{C}_j} d(t)\} \mid j \leq f-1\} \cup \{t_f = s\}$ of hub vertices such that:*

(i) *$|\mathcal{C}_j| \leq \lfloor \lambda/(\alpha + \beta\kappa) \rfloor$ for all $j = 1, 2, \ldots, f$;*
(ii) *$q(Z) \leq \kappa$ for all $Z \in \mathcal{M}$;*
(iii) *$\sum_{Z \in \mathcal{C}_j} q(Z) \geq \lfloor \lambda/(\alpha + \beta\kappa) \rfloor (\kappa/2)$ 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 $\lfloor \lambda/(\alpha + \beta\kappa) \rfloor$.*

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

To prove Lemma 5, we can assume without loss of generality that in a given tree T, (i) all terminals are leaves, i.e., $M = L(T)$, by introducing a new edge of weight zero for each non-leaf terminal, 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 replicating internal vertices of degree more than 3, so that copies of the same vertex are connected with zero-weight edges.

We prove Lemma 5 by showing that the next algorithm actually delivers a desired tuple

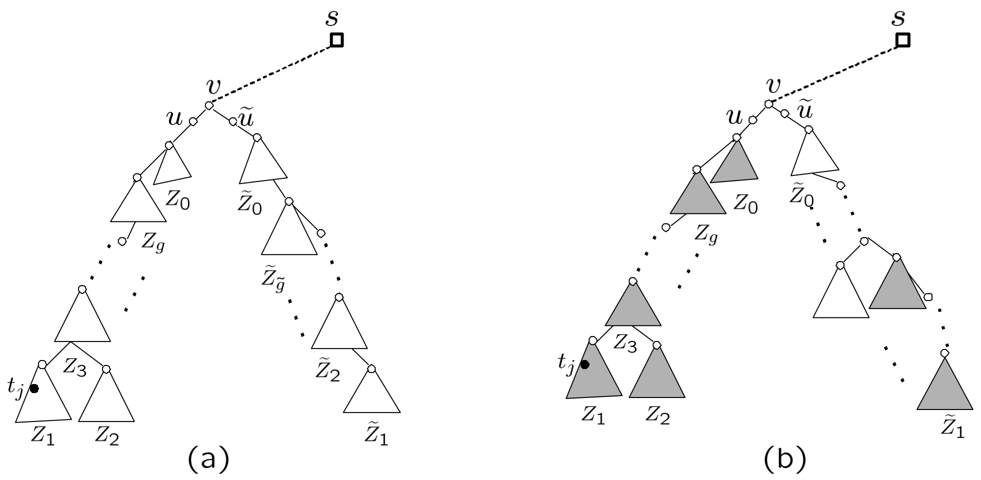

Fig. 1. Illustration of the case of $|\mathcal{Z}_2| = g + \tilde{g} \geq \lambda'$ in an iteration of algorithm TreeCover; (a) Line 10 identifies a terminal $t_j \in V(T_v)$ with the minimum vertex weight d, where $t_j \in V(T_u)$ in this figure; (b) Line 17 or 19 constructs $\mathcal{C}_j$ that contains 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 $\mathcal{C}_j$. Line 26 then removes all the terminals in $\cup_{Z \in \mathcal{C}_j} Z$ from the terminal set M, and hence no vertices of $V(T_u)$ will be chosen as hub vertices in the subsequent iterations.

$(\mathcal{M}, \mathcal{B}, \mathcal{T}')$. 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 $M = L(\widehat{T})$, a demand function $q : M \to R^+$, a vertex weight function $d : M \to R^+$, a demand capacity κ with $\kappa \geq max\{q(v) \mid v \in M\}$, an edge capacity $\lambda > 0$, and prescribed constants $\alpha, \beta \geq 0$ with $\lambda \geq \alpha + \beta\kappa$.
Output: A tuple $(\mathcal{M}, \mathcal{B}, \mathcal{T}')$ that satisfies Conditions (i)-(v) in Lemma 5.

```
1  $\mathcal{T}' := \emptyset$; $T := \widehat{T}$; $j := 0$; $\lambda' = \lfloor \lambda/(\alpha + \beta\kappa) \rfloor$;
2  while The current $T$ has a vertex $v$ with $q(V(T_v) \cap M) \geq \kappa\lambda'/2$ do
3    $j := j + 1$; Choose such $v$ with the maximum depth in $T$;
4    if $v \in L(T)$ then
5      $Z := \{v\}$; $\mathcal{C}_j := \{Z\}$; $t_j := t_Z := v$; $\mathcal{T}' := \mathcal{T}' \cup \{T\langle Z \cup \{t_Z\}\rangle\}$;
6    else   /* $|Ch_T(v)| = 2$ by the choice of $v$ */
7      Denote $Ch_T(v) = \{u, \tilde{u}\}$ and $Z_v = V(T_v) \cap M$;
8      Find a partition $\mathcal{Z}_1 \cup \mathcal{Z}_2$ of $Z_v$ by applying Lemma 3 with $(T_v, w, v, Z_v, q, \kappa)$;
       /* $|\mathcal{Z}_1| \leq |Ch_T(v)| = 2$ from Lemma 3(i) */
9      Denote $\mathcal{Z}_1 = \{Z_0, \widetilde{Z}_0\}$ (possibly $Z_0 = \emptyset$ or $\widetilde{Z}_0 = \emptyset$) 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);   /* $g, \tilde{g} < \lambda'$ since $q(V(T_u) \cap M), q(V(T_{\tilde{u}}) \cap M) <$
       $\kappa\lambda'/2$ and $q(Z) \geq \kappa/2$ for every $Z \in \mathcal{Z}_2$ */
10     Choose $t_j \in \{\mathrm{argmin}_{t \in M \cap V(T_v)} d(t)\}$, where we assume $t_j \in V(T_u)$ w.l.o.g;
11     if $g + \tilde{g} < \lambda'$ then
12       $\mathcal{C}_j := \{Z_0 \cup \widetilde{Z}_0, Z_1, \ldots, Z_g, \widetilde{Z}_1, \ldots, \widetilde{Z}_{\tilde{g}}\}$   /* $|\mathcal{C}_j| = g + \tilde{g} + 1 \leq \lambda'$. */
         /* $E(T\langle Z\rangle) \cap E(T\langle Z_0 \cup \widetilde{Z}_0\rangle) = \emptyset$ for all $Z \in \mathcal{Z}_2$, by Corollary 4(i) */
13     else  /* $g + \tilde{g} \geq \lambda'$ */
14       Let $Z_b \in \{Z_1, \ldots, Z_g\}$ be a subset such that $E(T\langle Z\rangle) \cap E(T\langle Z_0 \cup Z_b\rangle)$
         $= \emptyset$ for all $Z \in \mathcal{Z} - \{Z_0, Z_b\}$;  /* Such $Z_b$ exists by Corollary 4(ii) and
         $g > 0$ (any $Z_b \in \{Z_1, \ldots, Z_g\}$ will do if $Z_0 = \emptyset$) */
15       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;
16       if $q(Z_0 \cup Z_b) \leq \kappa$ then
17         $\mathcal{C}_j := \{Z_1, \ldots, Z_{b-1}, Z_0 \cup Z_b, Z_{b+1}, \ldots, Z_g\} \cup \{\widetilde{Z}_1, \ldots, \widetilde{Z}_{\lambda'-g}\}$ /* $|\mathcal{C}_j| = \lambda'$ */
18       else  /* $q(Z_0 \cup Z_b) > \kappa$ and $g < \lambda' - 1$ since $q(V(T_u) \cap M) < \kappa\lambda'/2$ */
19         $\mathcal{C}_j := \{Z_0, Z_1, Z_2, \ldots, Z_g\} \cup \{\widetilde{Z}_1, \ldots, \widetilde{Z}_{\lambda'-g-1}\}$   /* $|\mathcal{C}_j| = \lambda'$ */
20       end if
21     end if;
22     for each $Z \in \mathcal{C}_j$ do
23       $t_Z := t_j$; $\mathcal{T}' := \mathcal{T}' \cup \{T\langle Z \cup \{t_Z\}\rangle\}$
24     end for
25   end if;
26   $M := M - \cup_{Z \in \mathcal{C}_j} Z$; $T := T\langle M \cup \{s\}\rangle$
     /* $t_j \notin V(T)$ */
27 end while;
```

```
       /* $q(M) < \kappa\lambda'/2$ */
28  $f := j+1$; $t_f := s$;
29  if $M = \emptyset$ then
30      $\mathcal{C}_f := \emptyset$
31  else
32      Find a partition $\mathcal{Z}_1 \cup \mathcal{Z}_2$ of $M$ by applying Lemma 3 with $(T, w, s, M,$
        $q, \kappa)$, where $\mathcal{Z}_1 = \{Z_0, \widetilde{Z}_0\}$;   /* $|\mathcal{Z}_2| < \lambda'$ since $q(Z) \geq \kappa/2$, $Z \in \mathcal{Z}_2$ */
33      $\mathcal{C}_f := \{Z_0 \cup \widetilde{Z}_0\} \cup \mathcal{Z}_2$;   /* $|\mathcal{C}_f| = |\mathcal{Z}_2| + 1 \leq \lambda'$. */
34      for each $Z \in \mathcal{C}_f$ do
35        $t_Z := s$; $\mathcal{T}' := \mathcal{T}' \cup \{T\langle Z \cup \{t_Z\}\rangle\}$
36      end for
37  end if;
38  $\mathcal{M} := \cup_{1 \leq j \leq f} \mathcal{C}_j$; $\mathcal{B} := \{t_j \mid 1 \leq j \leq f\}$.
```

Now we prove that the tuple $(\mathcal{M}, \mathcal{B}, \mathcal{T}')$ output from algorithm TreeCover satisfies Conditions (i)-(v) in Lemma 5.

(i) Clearly, $|\mathcal{C}_j| = 1 \leq \lambda'$ for any collection $\mathcal{C}_j$ computed in line 5. Consider a collection $\mathcal{C}_j$ computed in line 12. 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 17 or 19, 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 32 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 33. This proves (i).

(ii) For a collection $\mathcal{C}_j$ computed in line 5, $q(Z) \leq \kappa$, $Z \in \mathcal{C}_j$, 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 8 by applying Lemma 3 to $(T_v, w, v, Z_v, q, \kappa)$. Lemma 3(ii)-(iii) implies that $q(Z_0 \cup \widetilde{Z}_0) < \kappa$ and $q(Z) \leq \kappa$ for all $Z \in \mathcal{Z}_2$. Furthermore, for a collection $\mathcal{C}_j$ computed in line 17, we have $q(Z_0 \cup Z_b) \leq \kappa$. Hence each subset Z added to $\mathcal{C}_j$ in line 12, 17, or 19 has demand at most κ. Lemma 3(ii)-(iii) implies also that each subset of $\mathcal{C}_f$ computed in line 33 has demand at most κ. This proves (ii).

(iii) This condition holds for a collection $\mathcal{C}_j$ computed in line 5 since $q(v) = q(V(T_v) \cap M) \geq \kappa\lambda'/2$. Consider a collection $\mathcal{C}_j$ computed in line 12. 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 8 is a partition of Z_v and $q(Z_v) \geq \kappa\lambda'/2$ by using the condition in line 2. For a collection $\mathcal{C}_j$ computed in line 17, 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 19, 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 $Z_1, Z_2 \in \mathcal{C}_j$, where T is the current tree during the jth iteration. Moreover, since any collection computed in line 5, 12, or 33 contains all subsets in a partition $\mathcal{Z}_1 \cup \mathcal{Z}_2$ of Z_v computed in line 8 and by the assumption in line 15 used in constructing $\mathcal{C}_j$ in line 17 or 19, we conclude that $E(T\langle Z\rangle) \cap E(T\langle (M - \cup_{Z \in \mathcal{C}_j} Z) \cup \{s\}\rangle) = \emptyset$ 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.

Lemma 6 *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 $\mathcal{M}_1(e) = \{Z \in \mathcal{M} \mid e \in E(\widehat{T}\langle Z\rangle)\}$, it holds $|\mathcal{M}_1(e)| \leq 1$;*
(ii) *$|\{Z \in \mathcal{M} \mid Z \subseteq V(\widehat{T}) - V(\widehat{T}_y), t_Z \in V(\widehat{T}_y)\}| \leq \lambda' - 1$; and*
(iii) *$|\{Z \in \mathcal{M} \mid Z \subseteq V(\widehat{T}_y), t_Z \in V(\widehat{T}) - V(\widehat{T}_y)\}| \leq \lambda' - |\mathcal{M}_1(e)|$.*

Proof. (i) By Lemma 5(iv), we have $E(\widehat{T}\langle Z_1\rangle) \cap E(\widehat{T}\langle Z_2\rangle) = \emptyset$ for all distinct subsets $Z_1, Z_2 \in \mathcal{M}$. This means that there exists at most one subset $Z \in \mathcal{M}$ such that $e \in E(\widehat{T}\langle Z\rangle)$ and consequently $|\mathcal{M}_1(e)| \leq 1$, which proves (i).

Note that throughout processing of any subtree T_v (for a vertex v chosen in line 3), the algorithm does not assign any subset in $\{Z \in \mathcal{M} \mid Z \subseteq V(\widehat{T}) - V(\widehat{T}_v)\}$ (resp., $\{Z \in \mathcal{M} \mid Z \subseteq V(\widehat{T}_v)\}$) to a hub vertex in $V(\widehat{T}_v)$ (resp., $V(\widehat{T}) - V(\widehat{T}_v)$). This implies that when $y \notin D_{\widehat{T}}(v) - \{v\}$, none of the subsets in $\{Z \in \mathcal{M} \mid Z \subseteq V(\widehat{T}) - V(\widehat{T}_y)\}$ (resp., $\{Z \in \mathcal{M} \mid Z \subseteq V(\widehat{T}_y)\}$) is assigned to a hub vertex in $V(\widehat{T}_y)$ (resp., $V(\widehat{T}) - V(\widehat{T}_y)$). Then it is sufficient to prove properties (ii) and (iii) for the subtree $T = \widehat{T}\langle(M - \cup_{i<j}(\cup_{Z \in \mathcal{C}_i} Z)) \cup \{s\}\rangle$, where the vertex v chosen in line 3 of the jth iteration be such that $y \in D_T(v) - \{v\}$.

(ii) Consider the first moment when a vertex in $V(T_y)$ is assigned to the hub vertex of a subset in $\mathcal{M}_2(e) = \{Z \in \mathcal{M} \mid Z \subseteq V(T) - V(T_y)\}$ during the execution of TreeCover. Let v be the vertex such that the tree T_v with $y \in D_T(v) - \{v\}$ is being processed in the jth iteration of the algorithm. The algorithm first chooses a vertex $t_j \in V(T_v)$ in line 10, where $t_j \in V(T_u)$ is assumed without loss of generality, and then constructs a collection $\mathcal{C}_j$ such that all terminals in $V(T_u)$ are contained in $\mathcal{C}_j$ ($V(T_u) \cap M \subseteq \cup_{Z \in \mathcal{C}_j} Z$) and all subsets of $\mathcal{C}_j$ are assigned to a hub vertex t_j (see Fig. 1). This implies that $y \in V(T_u)$ and $t_j \in V(T_y)$. Moreover, once a set of subsets in $\mathcal{M}_2(e)$ is assigned to a hub vertex in T_y in an iteration of the algorithm, none of the vertices of T_y will become a hub vertex in the subsequent iterations since all terminals in $\mathcal{C}_j$ (and hence in $V(T_u)$) will be removed from the terminal set in the next iterations (see line 26). Therefore, all subsets in $\mathcal{M}_2(e)$ assigned to a hub vertex in $V(T_y)$ are assigned to t_j. On the other hand, the number of subsets assigned to t_j (which equals $|\mathcal{C}_j|$) is the sum of the number of subsets in $\mathcal{M}_2(e)$ which are assigned to t_j and subsets in $\{Z \in \mathcal{M} \mid Z \cap V(T_y) \neq \emptyset\}$. There exists at least one subset in the latter set since $t_j \in V(T_y)$ ($V(T_y) \cap M \neq \emptyset$). Hence the number of subsets in $\mathcal{M}_2(e)$ which are assigned to t_j is at most $\lambda' - 1$ since $|\mathcal{C}_j| \leq \lambda'$. This proves (ii).

(iii) Consider the first moment when a vertex in $V(T) - V(T_y)$ is assigned to the hub vertex of a subset in $\mathcal{M}_3(e) = \{Z \in \mathcal{M} \mid Z \subseteq V(T_y)\}$ during the execution of TreeCover. Let v be the vertex such that the tree T_v with $y \in D_T(v) - \{v\}$ is being processed in the jth iteration of the algorithm. Note that, for $Ch(v) = \{u, \tilde{u}\}$, we have $q(V(T_u) \cap M)/(\kappa/2), q(V(T_{\tilde{u}}) \cap M)/(\kappa/2) < \lambda'$ since otherwise $q(V(T_u) \cap M) \geq \kappa\lambda'/2$

or $q(V(T_{\tilde{u}}) \cap M) \geq \kappa\lambda'/2$ would violate the choice of v. Thus, for a partition $\mathcal{Z}_1 \cup \mathcal{Z}_2$ of Z_v computed and described in lines 8 and 9, the maximum number of possible subsets in $\{Z \in \mathcal{Z}_2 \mid Z \subseteq V(T_u)\}$ (resp., $\{Z \in \mathcal{Z}_2 \mid Z \subseteq V(T_{\tilde{u}})\}$) is less than λ'. Moreover, we have $|\{Z \in \mathcal{Z}_1 \mid Z \subseteq V(T_u)\}|, |\{Z \in \mathcal{Z}_1 \mid Z \subseteq V(T_{\tilde{u}})\}| \leq 1$, by Lemma 3(i). Hence, $|\{Z \in \mathcal{M} \mid Z \cap V(T_u) \neq \emptyset\}|, |\{Z \in \mathcal{M} \mid Z \cap V(T_{\tilde{u}}) \neq \emptyset\}| \leq \lambda'$ hold (since $\mathcal{M} = \cup_{1 \leq j \leq f} \mathcal{C}_j$). This implies that $|\{Z \in \mathcal{M} \mid Z \cap V(T_y) \neq \emptyset\}| \leq \lambda'$ since $y \in D_T(u)$ or $y \in D_T(\tilde{u})$. That is, the number of subsets in $\mathcal{M}_1(e) \cup \mathcal{M}_3(e)$ is at most λ' and hence the number of subsets in $\mathcal{M}_3(e)$ is at most $\lambda' - |\mathcal{M}_1(e)|$ since $\mathcal{M}_1(e) \cap \mathcal{M}_3(e) = \emptyset$. This proves (iii). □

We are ready to prove property (v) in Lemma 5. 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 $\mathcal{M}_1(e)$ be as defined in Lemma 6. The number of trees in $\mathcal{T}'$ passing through e toward y is at most the sum of the number of trees in $\{\widehat{T}\langle Z \cup \{t_Z\}\rangle \in \mathcal{T}' \mid Z \in \mathcal{M}_1(e)\}$ 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 trees in $\{\widehat{T}\langle Z \cup \{t_Z\}\rangle \in \mathcal{T}' \mid Z \in \mathcal{M}_1(e)\}$ 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 6(i)-(iii) completes the proof of (v). □

Before describing the algorithm, we discuss the following lemma.

Lemma 7 *Let $(\mathcal{M}, \mathcal{B}, \mathcal{T}')$ be a tuple obtained by applying Lemma 5 to $(T, w, s, M, q, d, \alpha, \beta, \kappa, \lambda)$. Then we can find a new partition $\mathcal{C}'_1, \mathcal{C}'_2, \ldots, \mathcal{C}'_f$ of $\mathcal{M}$ by swapping subsets between $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_f$, so that each collection $\mathcal{C}'_j$ contains at most $\lfloor \lambda/(\alpha + \beta\kappa)\rfloor$ subsets from $\mathcal{M}$, all of which are assigned to the hub vertex t_j, $j = 1, 2, \ldots, f$, and for $\mathcal{T}'' = \{T\langle Z \cup \{t_Z\}\rangle \mid Z \in \mathcal{M}\}$, it holds $|\{T' \in \mathcal{T}'' \mid e \in E(T')\}| \leq \lfloor \lambda/(\alpha + \beta\kappa)\rfloor$ for any edge $e \in E(T)$.*

Proof. For each edge $e = (x, y) \in E(T)$, $y \in Ch(x)$, with $|\{T' \in \mathcal{T}' \mid e \in E(T')\}| > \lfloor \lambda/(\alpha + \beta\kappa)\rfloor$, we proceed as follows. Define $\mathcal{M}_{in}(e) := \{Z \in \mathcal{M} \mid Z \subseteq V(T) - V(T_y), t_Z \in V(T_y)\}$ and $\mathcal{M}_{out}(e) := \{Z \in \mathcal{M} \mid Z \subseteq V(T_y), t_Z \in V(T) - V(T_y)\}$. Lemma 6 implies that

$$|\mathcal{M}_{in}(e)|, |\mathcal{M}_{out}(e)| \leq \lfloor \lambda/(\alpha + \beta\kappa)\rfloor - |\mathcal{M}_1(e)|,$$

where $\mathcal{M}_1(e) = \{Z \in \mathcal{M} \mid e \in E(T\langle Z\rangle)\}$. On the other hand, by the construction of $\mathcal{T}'$, we conclude that the number of trees in $\mathcal{T}'$ passing through e in both directions equals $|\mathcal{M}_{in}(e)| + |\mathcal{M}_{out}(e)| + |\mathcal{M}_1(e)|$. Therefore, $\mathcal{M}_{in}(e) \neq \emptyset$ and $\mathcal{M}_{out}(e) \neq \emptyset$ in the case where the total number of trees in $\mathcal{T}'$ passing through e exceeds $\lfloor \lambda/(\alpha + \beta\kappa)\rfloor$. In this case, we swap an arbitrary subset $Z \in \mathcal{M}_{in}(e)$ with another subset $Z' \in \mathcal{M}_{out}(e)$, where we assume that Z and Z' belong to collections $\mathcal{C}_j$ and $\mathcal{C}_{j'}$, respectively, and then reassign the hub vertices of Z and Z' such that $t_Z = t_{j'}$ and $t_{Z'} = t_j$. As a result, $\mathcal{M}_{in}(e)$, $\mathcal{M}_{out}(e)$, $\mathcal{C}_j$, $\mathcal{C}_{j'}$, and $\mathcal{T}'$ are updated so that $\mathcal{M}_{in}(e) := \mathcal{M}_{in}(e) - \{Z\}$, $\mathcal{M}_{out}(e) := \mathcal{M}_{out}(e) - \{Z'\}$, $\mathcal{C}_j := (\mathcal{C}_j - \{Z\}) \cup \{Z'\}$, $\mathcal{C}_{j'} := (\mathcal{C}_{j'} - \{Z'\}) \cup \{Z\}$, and $\mathcal{T}' := (\mathcal{T}' - \{T\langle Z \cup \{t_j\}\rangle, T\langle Z' \cup \{t_{j'}\}\rangle\}) \cup \{T\langle Z \cup \{t_Z\}\rangle, T\langle Z' \cup \{t_{Z'}\}\rangle\}$. This swapping operation decreases the number of trees in $\mathcal{T}'$ passing through each edge in $E(T\langle Z \cup \{t_j\}\rangle) \cap E(T\langle Z' \cup \{t_{j'}\}\rangle)$ (which includes e), where t_j and $t_{j'}$ were the previous hub vertices of Z and Z', respectively, and hence $|\mathcal{M}_{in}(e)|, |\mathcal{M}_{out}(e)| \leq \lfloor \lambda/(\alpha + \beta\kappa)\rfloor -$

$|\mathcal{M}_1(e)|$ still holds. Note that, the number of trees in $\mathcal{T}'$ passing through each of the remaining edges of T never increases. This swapping process is repeated as long as the number of trees in the current $\mathcal{T}'$ passing through e exceeds $\lfloor \lambda/(\alpha+\beta\kappa) \rfloor$.

Thus $\mathcal{C}'_j := \mathcal{C}_j$, $j = 1, 2, \ldots, f$, and $\mathcal{T}'' := \mathcal{T}'$ satisfy conditions of the lemma. $\square$

The idea of the proof of Lemma 7 is originated from a procedure of swapping paths in the algorithm for CND problem due to Hassin et al. [7].

4 Approximation Algorithm to GCTR

This section presents an approximation algorithm for an instance $I = (G, w, s, M, q, \alpha, \beta, \kappa, \lambda)$ of GCTR problem based on results on tree covers in the previous section. Our algorithm begins by computing an approximate Steiner tree T in $(G, w, M \cup \{s\})$. We then find a tree cover $\mathcal{T}''$ of the tree T such that, for each $e \in E(T)$, $|\{T' \in \mathcal{T}'' \mid e \in E(T')\}| \le \lfloor \lambda/(\alpha+\beta\kappa) \rfloor$ and hence $\sum_{T' \in \mathcal{T}'' : e \in E(T')} (\alpha + \beta q(D_{T'}(v^e) \cap M)) \le (\alpha+\beta\kappa)|\{T' \in \mathcal{T}'' \mid e \in E(T')\}| \le (\alpha+\beta\kappa)\lfloor \lambda/(\alpha+\beta\kappa) \rfloor \le \lambda$, where $e = (u^e, v^e) \in E(T')$ with $v^e \in Ch_{T'}(u^e)$. Finally, we connect each tree in $\mathcal{T}''$ to s in order to get a tree-routings $\mathcal{T}$ in the instance I.

Algorithm ApproxGCTR
Input: An instance $I = (G, w, s, M, q, \alpha, \beta, \kappa, \lambda)$ of GCTR.
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 $d : M \to R^+$ by setting

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

Step 2. Apply Lemma 5 to $(T, w, s, M, q, d, \alpha, \beta, \kappa, \lambda)$ to get a partition $\mathcal{M} = \cup_{1 \le j \le f} \mathcal{C}_j$ 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. Apply Lemma 7 to the tuple $(\mathcal{M}, \mathcal{B}, \mathcal{T}')$ output from Step 2 to get a new partition $\mathcal{C}'_1, \mathcal{C}'_2, \ldots, \mathcal{C}'_f$ of $\mathcal{M}$ and a set $\mathcal{T}'' = \{T\langle Z \cup \{t_Z\}\rangle \mid Z \in \mathcal{M}\}$ of subtrees of T that satisfy the conditions of the lemma.
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 $\mathcal{T} := \{T_Z = T\langle Z \cup \{t_Z\}\rangle + SP(s, t_Z) \mid Z \in \mathcal{M}\}$ and output $(\mathcal{M}, \mathcal{T})$.

Now we show the feasibility and analyze the approximation factor of the approximate solution $(\mathcal{M}, \mathcal{T})$ output by algorithm ApproxGCTR.

Theorem 8 *For an instance $I = (G, w, s, M, q, \alpha, \beta, \kappa, \lambda)$ of* GCTR*, algorithm* ApproxGCTR *delivers a $(2\lceil \lambda/(\alpha+\beta\kappa) \rceil / \lfloor \lambda/(\alpha+\beta\kappa) \rfloor + \rho_{\mathrm{ST}})$-approximate solution $(\mathcal{M}, \mathcal{T})$, where ρ_{ST} is the approximation ratio of solution T to the Steiner tree problem.*

Proof. Lemma 5(ii) implies that $(\mathcal{M}, \mathcal{T})$ satisfies the demand capacity constraint on each tree.

Now we show that $\mathcal{T}$ satisfies the edge capacity constraint. Let $\mathcal{M} = \cup_{1 \le j \le f} \mathcal{C}'_j$ and $\mathcal{T}''$ be output from Step 3 of algorithm ApproxGCTR. Note that each tree $T_Z \in \mathcal{T}$ is a tree $T\langle Z \cup \{t_Z\}\rangle \in \mathcal{T}''$ plus the shortest path $SP(s, t_Z)$ between s and t_Z in (G, w). By Lemma 7, $|\{T' \in \mathcal{T}'' \mid e \in E(T')\}| \le \lfloor \lambda/(\alpha + \beta\kappa) \rfloor$ for any $e \in E(T)$. On the other hand, each collection $\mathcal{C}'_j$, $j \le f$, contains at most $\lfloor \lambda/(\alpha + \beta\kappa) \rfloor$ subsets from $\mathcal{M}$, all of which are assigned to a common hub vertex t_j. Thus, by installing one copy of each edge of the Steiner tree T and each edge in a shortest path $SP(s, t_j)$ between s and t_j in (G, w), $j \le f - 1$ $(t_f = s)$, we get a set $\mathcal{T}$ of tree-routings such that $|\{T_Z \in \mathcal{T} \mid e \in E(T_Z)\}| \le k_e \lfloor \lambda/(\alpha + \beta\kappa) \rfloor$ for any $e \in E(G)$, where k_e is the number of installed copies of e. Consequently, for any $e \in E(G)$, we observe that

$$\begin{aligned}\sum_{T_Z \in \mathcal{T}: e \in E(T_Z)} (\alpha + \beta q(D_{T_Z}(v^e) \cap Z)) &\le \sum_{T_Z \in \mathcal{T}: e \in E(T_Z)} (\alpha + \beta q(Z)) \\ &\le (\alpha + \beta\kappa) k_e \lfloor \lambda/(\alpha + \beta\kappa) \rfloor \le k_e \lambda,\end{aligned}$$

where $e = (u^e, v^e) \in E(T_Z)$ with $v^e \in Ch_{T_Z}(u^e)$. Thereby $(\mathcal{M}, \mathcal{T})$ is feasible to I and the total weight of the installed edges on the network is bounded by

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

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

$$\sum_{1 \le j \le f-1} d(t_j) \le 2\lceil \lambda/(\alpha + \beta\kappa) \rceil / \lfloor \lambda/(\alpha + \beta\kappa) \rfloor opt(I). \tag{4}$$

Consider a collection $\mathcal{C}_j$, $j \le f - 1$ obtained by applying Lemma 5 to $(T, w, s, M, q, d, \alpha, \beta, \kappa, \lambda)$ 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 new collection $\mathcal{C}'_j$ remains unchanged. That is, the set $\mathcal{B}$ of hub vertices computed in Step 2 remains unchanged throughout the algorithm. The choice of t_j and Lemma 5(iii) imply that

$$\sum_{t \in Z \in \mathcal{C}_j} q(t) d(t) \ge d(t_j) \sum_{t \in Z \in \mathcal{C}_j} q(t) \ge \lfloor \lambda/(\alpha + \beta\kappa) \rfloor (\kappa/2) d(t_j). \tag{5}$$

By summing inequality (5) overall $\mathcal{C}_j$'s, $j \le f - 1$, we have

$$\begin{aligned}(\alpha + \beta\kappa) \lfloor \lambda/(\alpha + \beta\kappa) \rfloor / (2\lambda) \sum_{1 \le j \le f-1} d(t_j) &\le (\alpha + \beta\kappa)/(\kappa\lambda) \sum_{1 \le j \le f-1} \sum_{t \in Z \in \mathcal{C}_j} q(t) d(t) \\ &\le (\alpha + \beta\kappa)/(\kappa\lambda) \sum_{t \in M} q(t) d(t).\end{aligned}$$

Hence Lemma 2 completes the proof of (4). $\square$

5 Conclusion

In this paper, we have studied the generalized capacitated tree-routing problem (GCTR), a new routing problem formulation under a multi-tree model with a general routing capacity, which unifies several important routing problems such as the capacitated network design problem (CND), the capacitated multicast tree routing problem (CMTR), and the capacitated tree-routing problem (CTR). We have proved that GCTR with $\lambda \geq \alpha + \beta\kappa$ is $(2\lceil \lambda/(\alpha+\beta\kappa)\rceil / \lfloor \lambda/(\alpha+\beta\kappa)\rfloor + \rho_{\text{ST}})$-approximable based on a new lower bound to the problem and 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 GCTR in the case of $\lambda < \alpha+\beta\kappa$. Also, it will be interested to obtain a better approximation algorithm for the fractional generalized capacitated tree-routing problem (FGCTR). We remark that GCTR with a very small λ compared with $\alpha + \beta\kappa$ is closely related with FGCTR.

References

[1] L. Behjat, New modeling and optimization techniques for the global routing problem, Ph.D. Thesis, University of Waterloo (2002).

[2] L. Behjat, A. Vannelli, W. Rosehart Integer linear programming models for global routing, Informs Journal on Computing, 18 (2) (2002) 137-150.

[3] P. Berman, V. Ramaiyer, Improved approximations for the Steiner tree problem, J. Algorithms, 17 (1994) 381-408.

[4] Z. Cai, G.-H Lin, G. Xue, Improved approximation algorithms for the capacitated multicast routing problem, In Proceedings of COCOON 2005, LNCS 3595 (2005) 136-145.

[5] M. R. Garey, D. S. Johnson, The rectilinear Steiner tree problem is NP-complete, SIAM J. Appl. Math., 32 (1977) 826-843.

[6] M. R. Garey and D. S. Johnson, Computers and Intractability, a Guide to the Theory of NP-completeness, Freeman, San Francisco 1978.

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

[8] S. Hougardy, H. J. Prömmel, A 1.598 approximation algorithm for the Steiner problem in graphs, In Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms, (1999) 448-453.

[9] R. Jothi, 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.

[10] M. Karpinsky, A. Zelikovsky, New approximation algorithms for the Steiner tree problem, J. Combin. Optim., 1 (1997) 47-65.

[11] S. Khuller, B. Raghavachari, N. N. Young, Balancing minimum spanning and shortest path trees, Algorithmica, 14 (1993) 305-322.

[12] G.-H. Lin, An improved approximation algorithm for multicast k-tree routing, Journal of Combinatorial Optimization, 9 (2004) 349-356.

[13] Y. Mansour, 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).

[14] E. Morsy, H. Nagamochi, An improved approximation algorithm for capacitated multicast routings in networks, Theoritical Computer Sceince, 390 (2008) 81-91.

[15] E. Morsy, H. Nagamochi, Approximating capacitated tree-routings in networks, In Proceedings of the 4th Annual Conference on Theory and Applications of Models of Computation (TAMC07), LNCS 4484 (2007) 342-353.

[16] H. J. Prömmel, A. Steger, RNC-approximation algorithms for the Steiner problem, In Proceedings of the 14th Annual Symposium on Theoritical Aspects of Computer Science, (1997) 559-570.

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

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

[19] H. Takahashi, A. Matsuyama, An approximate solution for the Steiner problem in graphs, Math. Japon., 24 (1980) 573-577.

[20] T Terlaky, A. Vannelli, H. Zhang, On routing in VLSI design and communication networks, ISAAC 2005, LNCS 3827 (2005) 1051-1060.

[21] A. Zelikovsky, An 11/6-approximation algorithm for the network Steiner problem, Algorithmica, 9 (1993) 463-470.

[22] L. Zhao, H. Yamamoto, Multisource receiver-driven layered multicast, In Proceedings of IEEE TENCON 2005 (2005), 1325-1328.