Approximation Algorithms for Min-Max Cycle Cover Problems

Wenzheng Xu, Weifa Liang, *Senior Member, IEEE,* and Xiaola Lin

Abstract—As a fundamental optimization problem, the vehicle routing problem has wide application backgrounds and has been paid lots of attentions in past decades. In this paper we study its applications in data gathering and wireless energy charging for wireless sensor networks, by devising improved approximation algorithms for it and its variants. The key ingredients in the algorithm design include exploiting the combinatorial properties of the problems and making use of tree decomposition and minimum weighted maximum matching techniques. Specifically, given a metric complete graph G and an integer $k > 0$, we consider rootless, uncapacitated rooted, and capacitated rooted min-max cycle cover problems in G with an aim to find k rootless (or rooted) edge-disjoint cycles covering the vertices in V such that the maximum cycle weight among the k cycles is minimized. For each of the mentioned problems, we develop an improved approximate solution. That is, for the rootless min-max cycle cover problem, we develop a $(5\frac{1}{3}+\epsilon)$-approximation algorithm; for the uncapacitated rooted min-max cycle cover problem, we devise a $(6\frac{1}{3}+\epsilon)$-approximation algorithm; and for the capacitated rooted min-max cycle cover problem, we propose a $(7+\epsilon)$-approximation algorithm. These algorithms improve the best existing approximation ratios of the corresponding problems $6+\epsilon$, $7+\epsilon$, and $13+\epsilon$, respectively, where ϵ is a constant with $0 < \epsilon < 1$. We finally evaluate the performance of the proposed algorithms through experimental simulations. Experimental results show that the actual approximation ratios delivered by the proposed algorithms are always no more than 2, much better than their analytical counterparts.

Index Terms—Wireless sensor networks; data gathering; mobile sinks; vehicle routing problem; min-max cycle cover; tree decomposition; approximation algorithms; combinatorial optimization.

1 Introduction

This paper studies a fundamental optimization problem in operations research and computer science communities – the vehicle routing problem, which has wide application backgrounds and has been paid lots of attentions in past decades [1], [2], [3], [4], [5], [6], [9], [10], [11], [15], [16], [23], [30], [31]. We here are motivated by its two application scenarios in wireless sensor networks: one is data gathering, another is wireless recharging of sensors.

We start from the application scenario one. In traditional wireless sensor networks (WSNs), sensors are powered by energy-limited batteries, and there is a stationary sink. All sensed data from sensors will be relayed to the stationary sink directly or through multihop relays for further processing. Since sensors near to the sink have to relay more data for others, they usually deplete their battery energy much faster. Such imbalanced energy consumptions among the sensors will shorten the network lifetime significantly. To prolong the network lifetime by minimizing the energy consumptions of sensors, a mobile sink instead of a stationary sink has been employed to travel around the vicinities of sensors periodically so that sensors can upload their sensed data to the mobile sink [20], [21], [29]. Although sink mobility can improve various network performance, it also results in data delivery delay due to the slow mechanical movement of the mobile sink, where the data delivery delay means the time duration of sensing data from its generation to its collection by the sink. Therefore, it is desirable to employ multiple mobile sinks so that the traveling distance of each mobile sink can be significantly shortened and more ‘fresh’ data (data with less delivery delay) can be collected on time [17], [19], [22], [34]. One fundamental optimization problem related to this is that, given k mobile sinks located at one or multiple depots in a large scale WSN, which are used to collaboratively collect sensed data from the sensors, how to find a traveling trajectory for each of the k mobile sinks such that the longest traveling time among them is minimized. If each of the k mobile sinks can upload its collected data to one of its nearby depots, the problem then is to find k rootless close traveling tours such that the maximum traveling time among the k tours is minimized [34]. Otherwise, each mobile sink has to return and upload its collected data to its own depot [17]. It thus requires that each of the k close tours contains a root (a depot) in this case. Moreover, since there are a limited number of available mobile sinks at each depot, the number of traveling tours allocated to the mobile sinks at each depot thus is restricted. Under this constraint, the optimization problem then is to find capacitated rooted close traveling tours for the k mobile sinks such that the maximum traveling time among the k tours is minimized.

We then deal with the second application scenario. We consider a wireless rechargeable sensor network in which each sensor can be recharged periodically to

- *W. Xu and W. Liang are with the Research School of Computer Science, The Australian National University, Canberra, ACT 0200, Australia. E-mails: wenzheng.xu@anu.edu.au, wliang@cs.anu.edu.au*
- *X. Lin is with the School of Information Science and Technology, Sun Yat-Sen University, Guangzhou, 51006, China. Email: linxl@mail.sysu.edu.cn*

avoid its energy expiration. To do so, we can employ one or multiple mobile chargers to traverse within the network and charge the sensors [8], [13], [18], [24], [26], [27], [28], [32], [33]. One fundamental question related to mobile chargers' charging tours scheduling is that given $k \geq 1$ mobile chargers, how to schedule and find charging tours for the k mobile chargers such that none of sensors in the network expires. As the mobile chargers themselves need to be recharged at depot(s) when they finish their charging tours, each charging tour of a mobile charger is a close tour including its depot. One typical optimization objective of this tour finding and scheduling problem is to minimize the maximum traveling distance among the k mobile chargers, thus minimizing the charging duration per tour. Also, the number of depots and the number of mobile chargers allocated to each depot may have restrictions, under this constraint, the problem can be casted as rootless (rooted), capacitated (uncapacitated) close tour problem.

In general, the vehicle routing problem in a wireless sensor network is to dispatch k mobile vehicles from a single depot (or multiple depots) to serve all sensors in the network such that the latest completion time among the k mobile vehicles is minimized, with or without the number of vehicle capacity constraint at each depot. The vehicle routing problem in a metric graph is equivalent to covering all vertices in the graph with k cycles such that the maximum cycle weight is minimized, where *a cycle weight* is the weighted sum of the edges in the cycle. We refer to this as *the min-max cycle cover problem*. In more general setting, there is also a weight (or handling time) on each vertex, a cycle weight is the weighted sum of edges and vertices in the cycle. We note that this general case can be reduced to the special case where only the edges have weights by transforming a vertex-weighted and edge-weighted graph into another edge-weighted graph [31]. Therefore, in the rest of this paper, we only consider edge-weighted graphs. The min-max cycle cover problem is NP-hard by reducing from the classical Traveling Salesman Problem (TSP) [25]. Thus, in this paper we will focus on devising approximation algorithms that achieve constant approximation ratios for the min-max cycle cover problem and its variants.

1.1 Related Work

In terms of data gathering in wireless sensor networks, Zhao *et al.* [34] studied the problem of finding traveling trajectories of multiple mobile collectors such that the maximum data gathering time among the mobile collectors is minimized. They proposed a heuristic algorithm for the problem, where the data gathering time of a traveling trajectory assigned to a mobile collector consists of the moving time of the mobile collector and the data uploading time of sensors in the trajectory. Kim *et al.* [17] considered the k traveling salesperson with neighborhood problem, which aims to find k close moving trajectories for the k mobile collectors such that the length of the longest trajectory is minimized, subject to that each trajectory contains the base station, where one mobile collector only needs to move to the communication range of a sensor in order to collect the accumulated data in that sensor. They also developed an approximation algorithm for the problem. Liang *et al.* [19] considered the similar k trajectory finding problem by exploring the combinatorial property of the problem and proposing a fast heuristic solution.

In general, the k trajectory finding and scheduling problem in wireless sensor networks can be abstracted as the min-max cycle cover problem or its variants. Thus, in the rest of this paper we focus on devising improved approximate solution to the min-max cycle cover problem, and a closely related problem to the min-max cycle cover problem is the min-max k-tree cover problem, since minimum spanning trees are constant factor approximations to traveling salesman tours [10]. *The min-max k-tree cover problem* is to find k edge-disjoint trees covering all vertices in a graph such that the maximum tree weight is minimized. Specifically, for the rootless min-max tree cover problem, Even *et al.* [10] and Arkin *et al.* [2] devised $(4+\epsilon)$-approximation algorithms independently by adopting different algorithmic techniques. Khani *et al.* [16] later improved the approximation ratio to $3+\epsilon$, which implies that there is a $(6+\epsilon)$-approximation algorithm for the rootless min-max cycle cover problem. Even *et al.* [10] also presented a $(4+\epsilon)$-approximation algorithm for the capacitated rooted min-max tree cover problem, assuming there are k roots with each having a unit capacity, which leads to an $(8+\epsilon)$-approximation algorithm for the capacitated rooted min-max cycle cover problem, where ϵ is given constant with $0 < \epsilon < 1$.

There are other studies on the min-max cycle cover problem without the use of min-max tree covers. For example, for the single-rooted min-max k-cycle cover problem, Frederickson *et al.* [11] proposed a $(1+e-1/k)$-approximation algorithm, where e is the best approximation ratio for the classic TSP problem. With multiple roots, Xu *et al.* [31] recently achieved a $(7+\epsilon)$-approximation ratio for the uncapacitated rooted min-max cycle cover problem for a vertex weighted metric graph, and they also presented $(7+\epsilon)$-approximation and $(13+\epsilon)$-approximation algorithms for the capacitated rooted min-max cycle cover problem with the overall capacity being equal to or larger than k, respectively.

It must be mentioned that although our design technique for the rootless min-max cycle cover problem is inspired by the work on the rootless min-max tree cover problem [16], they are essentially different. That is, given a graph G, assume that B^*_{tree} (or B^*) is the value of the optimal solution to the rootless min-max tree (or cycle) cover problem. The algorithm in [16] is based on a key observation that there is at most one edge with weight greater than $B^*_{tree}/2$ in any tree in an optimal min-max tree cover of G. We however observe that there are no edges with weight greater than $B^*/2$ and there are at most two edges with weight greater than $B^*/3$ in any cycle of an optimal min-max cycle cover (see Lemma 3). Therefore, at most two connected components can be obtained by removing the edges with weights greater than $B^*/3$ from any cycle in the optimal solution. In addition, unlike an existing tree decomposition technique that only ensures the upper bound on the weighted sum of the edges in each decomposed subtree (see Lemma 1), our tree decomposition technique for the capacitated rooted min-max cycle cover problem enables providing such a tree decomposition that the weighted sum of the edges in each decomposed subtree is bounded by not only an upper bound but also a lower bound (see Lemma 11).

1.2 Contributions

In this paper, we deal with the vehicle routing problem and its variants in wireless sensor networks by devising improved approximate solutions. The main contributions of this paper are as follows.

We first develop a $(5\frac{1}{3}+\epsilon)$-approximation algorithm and a $(6\frac{1}{3}+\epsilon)$-approximation algorithm for the rootless and uncapacitated rooted min-max cycle cover problems, respectively, which improve their existing approximation ratios $6+\epsilon$ and $7+\epsilon$, while keeping the same time complexity as the algorithms in [16]. We then devise a $(7+\epsilon)$-approximation algorithm for the capacitated rooted min-max cycle cover problem, which improves its existing approximation ratio of $13+\epsilon$ but with less running time [31]. Specifically, the time complexity of the algorithm in [31] is $O(n^{2.5}\log n(\log n+\log\frac{1}{\epsilon}))$ while ours is $O(n^{2.5}(\log n+\log\frac{1}{\epsilon}))$, where n is the number of vertices in graph G and ϵ is a constant with $0<\epsilon<1$. We finally evaluate the performance of the proposed algorithms through experimental simulations. Experimental results show that the actual approximation ratios delivered by the proposed algorithms are always no more than 2, much better than their analytical counterparts.

The rest of the paper is organized as follows. Section 2 introduces preliminaries. Section 3, 4, and 5 propose approximation algorithms for the three mentioned problems respectively. We finally evaluate the performance of the proposed algorithms through simulations in Section 6, and conclude our discussion in Section 7.

2 Preliminaries

In this section, we first introduce several notions and notations. We then provide the precise problem definitions. We finally introduce two important techniques: the tree decomposition technique and the transformation technique from a tree cover to a cycle cover.

We consider a complete graph $G=(V,E)$, and an edge weight function: $w: E\mapsto\mathbb{Z}^+$, and the edge weights satisfy the triangle inequality. The vertex set and the edge set of a graph G are referred to as $V(G)$ and $E(G)$ respectively, and $|V(G)|$ and $|E(G)|$ are referred to as the number of vertices and edges in G. For a weighted graph G, $w(G)$ is defined as $\sum_{e\in E(G)}w(e)$. A graph G is a *multi-graph* if there are multiple edges between a pair of vertices or there is a self loop at a vertex in the graph.

2.1 Problem definitions

Definition 1: Given a complete graph $G=(V,E)$, a metric edge weight function $w: E\mapsto\mathbb{Z}^+$ and a positive integer k, the *rootless min-max cycle cover problem* in G is to find k edge-disjoint cycles $C_1,C_2,\ldots,C_k$ covering all vertices in V, i.e., $\cup_{i=1}^{k}V(C_i)=V$ and $E(C_i)\cap E(C_j)=\emptyset$ if $i\neq j$, such that the maximum cycle weight, $\max_{i=1}^{k}\{w(C_i)\}$, is minimized.

Definition 2: Given a complete graph $G=(V,E)$, a depot set $D\subset V$, a metric edge weight function $w: E\mapsto\mathbb{Z}^+$, and a positive integer k, the *uncapacitated rooted min-max cycle cover problem* in G is to find k edge-disjoint cycles $C_1,C_2,\ldots,C_k$ covering all vertices in $V-D$ such that each cycle contains exactly one depot in D and the maximum cycle weight is minimized.
Notice that a depot can be included by multiple cycles in this problem definition.

Definition 3: Given a complete graph $G=(V,E)$, a depot set $D\subset V$, a metric edge weight function $w: E\mapsto\mathbb{Z}^+$, a positive integer k, and a constraint function $f: D\mapsto\mathbb{Z}^+$ that satisfies $\sum_{r\in D}f(r)\geq k$, the *capacitated rooted min-max cycle cover problem* in G is to find k edge-disjoint cycles $C_1,C_2,\ldots,C_k$ covering all vertices in $V-D$ such that each depot $r\in D$ is contained by at most $f(r)$ cycles, each cycle contains exactly one depot from D and the maximum cycle weight is minimized.

2.2 A paradigm of tree decomposition

A widely-used tree decomposition technique [16], [10] is to decompose a large tree (in terms of tree weight) into several edge-disjoint smaller subtrees by bounding the tree weight. For the sake of completeness, we state the tree decomposition by the following lemma.

Lemma 1: [16], [10] Given a tree T with weight $w(T)$, assume that each edge in T has weight no more than β and $w(T)\geq 2\beta$. Then, tree T can be decomposed into x edge-disjoint subtrees $T_1,\ldots,T_x$ such that $w(T_i)<2\beta$ for each i with $1\leq i\leq x$, and $\frac{\sum_{i=1}^{x}w(T_i)}{x}\geq\beta$, where $\beta>0$ and $2\leq x\leq\lfloor\frac{w(T)}{\beta}\rfloor$.

Proof: Trees with weight in the interval $[\beta,2\beta)$ can be split away from T until the weight of the leftover tree is less than 2β. Suppose that the split trees are $T_1,T_2,\ldots,T_x$ with $x\geq 2$. From the construction, we know that $w(T_i)\in[\beta,2\beta)$ for i with $1\leq i\leq x-1$. The only tree that may have weight less than β is T_x. Note that prior to splitting T_{x-1}, the weight of the remaining tree is at least 2β, therefore, the average weight of T_{x-1} and T_x is no less than β. Thus, the average weight of all T_is is at least β. Therefore, x cannot be greater than $\lfloor\frac{w(T)}{\beta}\rfloor$. □

2.3 A cycle cover derived from a tree cover

We introduce a popular technique that transforms a k-tree cover of a graph G into a k-cycle cover of G, and state this transformation in the following lemma.

Lemma 2: Given a metric complete graph $G = (V, E; w)$, a positive integer k, and a k-tree cover $\mathcal{T} = \{T_1, \ldots, T_k\}$ of G with $\max_{T_i \in \mathcal{T}}\{w(T_i)\} \leq \alpha B^*$, $\mathcal{T}$ can be transformed into an edge-disjoint k-cycle cover $\mathcal{C} = \{C_1, \ldots, C_k\}$ of G such that $\max_{C_i \in \mathcal{C}}\{w(C_i)\} \leq 2\alpha B^*$, where α is a constant greater than 1 and B^* is the value of the optimal solution to the min-max k-cycle cover problem in G.

Proof: For each tree T_i in $\mathcal{T}$, a Eulerian tour with the weight no more than $2\alpha B^*$ is obtained by doubling the edges in T_i, then a cycle C_i can be derived from this tour by shortcutting repeated vertices in the tour. As the edge weights meet the triangle inequality, we have $w(C_i) \leq 2w(T_i)$, for all i with $1 \leq i \leq k$. A cycle cover $\mathcal{C}$ then is found with $\max_{C_i \in \mathcal{C}}\{w(C_i)\} \leq 2\alpha B^*$. □

3 ALGORITHM FOR THE ROOTLESS MIN-MAX CYCLE COVER PROBLEM

In this section, we deal with the rootless min-max cycle cover problem in $G = (V, E)$ by devising a $(5\frac{1}{3} + \epsilon)$-approximation algorithm. We start with the following lemma, which will be the cornerstone of the proposed algorithm for the problem.

Lemma 3: Given a graph $G = (V, E)$, a metric edge weight function $w : E \mapsto \mathbb{Z}^+$, assume that C is a cycle in G with $w(C) \leq B$. Then, (i) for any edge $e \in E(C)$, $w(e) \leq B/2$. (ii) There are no more than two edges in $E(C)$ with weights greater than $B/3$.

Proof: Suppose that cycle C contains at least three vertices. Otherwise, the claims are straightforward and easily verified. We start with Case (i). Suppose that there is an edge $e = (u, v) \in E(C)$ with $w(e) > B/2$. Apart from a path P_1 consisting of a single edge e only in C, there is another vertex-disjoint path P_2 between u and v in C. The length of path P_2 is $w(C) - w(P_1) < B - B/2 = B/2$. On the other hand, following the triangle inequality of the edge weights in G, we have $w(e) \leq w(P_2) < B/2$, which contradicts the assumption. Therefore, $w(e) \leq B/2$ for each edge in C.

We then show Case (ii). As we assume that C contains at least three vertices, then C contains at least three edges as well. Now, suppose that there are at least three edges in C with weight greater than $B/3$. Then, $w(C)$ is larger than $3 \cdot B/3 = B$, which contradicts the assumption. □

In the following we will devise an algorithm for the problem. Let $OPT = \{C_1^*, C_2^*, \ldots, C_k^*\}$ denote an optimal solution to the rootless min-max cycle cover problem in G and B^* the maximum cycle weight in OPT, i.e., $B^* = \max_{i=1}^{k}\{w(C_i^*)\}$. Assume there is a guess B of B^* with $B \geq B^*$. The proposed algorithm is to find a k-tree cover $\mathcal{T} = \{T_1, T_2, \ldots, T_k\}$ of G covering all vertices in G with $\max_{T_i \in \mathcal{T}}\{w(T_i)\} \leq \frac{8}{3}B$ first, and then construct a cycle cover from the tree cover by Lemma 2. As a result, a k-cycle cover of G is obtained, and the maximum weight among the cycles is no more than $2 \cdot \frac{8}{3}B = \frac{16}{3}B$.

3.1 Algorithm overview

The basic idea of the proposed algorithm is as follows. A subgraph G' of graph G is obtained by removing all edges with $w(e) > B/3$. Assume that G' contains $l + h$ connected components $CC_1, \ldots, CC_l, CC_{l+1}, \ldots, CC_{l+h}$. Let T_i be an MST of CC_i for all i with $1 \leq i \leq l+h$. The $l+h$ connected components of G' can be further classified into light connected components and heavy connected components, where a connected component CC_i is referred to as a *light connected component* if $w(T_i) < B$; otherwise, it is referred to as a *heavy connected component*. Assume that G' contains l light connected components and h heavy connected components.

The general strategy adopted for the problem is to merge the MSTs first, using the edges with weight no more than $B/2$ to reduce the number of trees. Then, it is followed by the tree decomposition with bounding the weight of each decomposed tree within $\frac{8}{3}B$ such that the number of decomposed trees is no more than k. We distinguish the rest of our discussions into three cases: Case one: G' does not contain any heavy connected components, i.e., $h = 0$. Case two: G' does not contain any light connected components, i.e., $l = 0$. And Case three: G' contains both light and heavy connected components, i.e., $l \neq 0$ and $h \neq 0$. For each of these three cases we show how to find k trees covering all vertices in G such that the maximum tree weight is no more than $\frac{8}{3}B$.

3.2 Case one: G' does not contain any heavy connected components

We start with Case 1: there are no heavy connected components in G'. We construct a tree cover $\mathcal{T}$ by merging some of the l trees using the edges with weight no greater than $B/2$ so that the number of resulting trees is no more than k as follows.

As G' contains l light connected components only, an auxiliary graph $H = (X, E_X)$ is constructed as follows. Each vertex v_i in X corresponds to a light connected component CC_i, $1 \leq i \leq l$. There is an edge between two vertices v_i and v_j if and only if there is an edge in G between the vertices in CC_i and CC_j with weight no more than $B/2$ and $i \neq j$, for all i and j with $1 \leq i, j \leq l$. Let M be a maximum matching of graph H. Then, a tree cover $\mathcal{T}$ of G can be found based on M as follows.

Initially $\mathcal{T} = \emptyset$. Then, for each pair of matched vertices v_i and v_j in M, a resulting tree $T_{i,j}$ is obtained by adding a cheapest edge e (with weight no more than $B/2$) between the MST T_i of CC_i to the MST T_j of CC_j, which is then added to $\mathcal{T}$. It is obvious that $w(T_{i,j}) \leq \frac{5}{2}B$ as $w(T_i) < B$, $w(T_j) < B$, and $w(e) \leq \frac{B}{2}$. For each non-matched vertex v_i in H, add the MST T_i of CC_i to $\mathcal{T}$ directly. Clearly, the weight of each tree in $\mathcal{T}$ is no more than $\frac{5}{2}B$. The rest is to show that $|\mathcal{T}| \leq k$. Notice that $\mathcal{T}$ contains $|M| + |X - V(M)| = |M| + l - 2|M| = l - |M|$ trees. In the following, we show that $l - |M| \leq k$.

Consider the optimal solution OPT of the problem. We remove all edges with weight greater than $B/3$

This is the author's version of an article that has been published in this journal. Changes were made to this version by the publisher prior to publication.
The final version of record is available at http://dx.doi.org/10.1109/TC.2013.2295609

from OPT. Following Lemma 3, there are at most two edges in each cycle C_i^* in OPT with weight larger than $B/3$. Thus, no more than two connected components $C_{i,1}^*$ and $C_{i,2}^*$ will be the results after the removal of all edges with weight larger than $B/3$ from C_i^*. The cycles in OPT thus are classified into three different categories: light cycles, heavy cycles, and bad cycles, where a cycle C_i^* is a *light cycle* (or *heavy cycle*) if the connected components obtained by removing all the edges with weight greater than $B/3$ are contained in light connected components (or heavy components) CC_x and CC_y of G', where $1 \leq x, y \leq l$ (or $l+1 \leq x, y \leq l+h$). Otherwise, C_i^* is a *bad cycle*, which means that one of the two connected components $C_{i,1}^*$ and $C_{i,2}^*$ is in a light connected component CC_j and the other is in a heavy component $CC_{j'}$ of G' with $j \neq j'$. Assume that OPT contains k_l^* light cycles, k_h^* heavy cycles, and k_b^* bad cycles, then $|OPT| = k_l^* + k_h^* + k_b^* = k$.

We now construct another auxiliary graph $H' = (X, E'_X)$ based on the k_l^* light cycles in OPT as follows.

Each vertex $v_i \in X$ corresponds to a light connected component CC_i of G'. There is a self loop edge on vertex v_i if there exists a cycle $C_x^* \in OPT$ such that $CC_{x,1}^*$ is contained in CC_i of G' (there is no $CC_{x,2}^*$). There is an edge $(v_i, v_j) \in E'_X$ if there is a cycle $C_x^* \in OPT$ such that $CC_{x,1}^*$ and $CC_{x,2}^*$ are in connected components CC_i and CC_j, respectively. Clearly, it is easy to verify that $|E'_X| = k_l^* \leq k$. Let M' be a maximum matching of graph H'. A tree cover $\mathcal{T}'$ based on the maximum matching M' in H' then can be constructed, using the similar approach as we did for the tree cover $\mathcal{T}$ based on the maximum matching M in H. Thus, $\mathcal{T}'$ contains $l - |M'|$ trees.

We claim that $l - |M| \leq k$ by the following lemma.

Lemma 4: Given the constructed graph $H = (X, E_X)$ and $H' = (X, E'_X)$, let M and M' be the maximum matchings in H and H' respectively, we have $l - |M| \leq l - |M'|$ and $l - |M'| \leq k$, then $l - |M| \leq k$.

Proof: We start by showing that $l - |M| \leq l - |M'|$. Note that if there is an edge in H' between two different vertices v_i and v_j in X, then there is a cycle C_x^* in OPT such that the two connected components $CC_{x,1}^*$ and $CC_{x,2}^*$ derived from C_x^* are contained in connected components CC_i and CC_j of G', respectively. By Lemma 3, CC_i and CC_j can be connected with an edge with weight no greater than $B/2$. Therefore, there must have an edge in H between vertices v_i and v_j, too. As M is a maximum matching in H, we have $|M| \geq |M'|$, i.e., $l - |M| \leq l - |M'|$.

We then show that $l - |M'| \leq k$. Following our assumption that there are only l light connected components in G', each vertex in H' is adjacent to at least one edge in E'_X (a self-loop edge is counted as an edge in E'_X too) by the construction of H'. Then, each vertex in $X - V(M')$ is adjacent to at least an edge in $E'_X - M'$, and no two distinct vertices in $X - V(M')$ are connected by an edge in $E'_X - M'$ as the matching M' is the maximum one. Thus, $|X - V(M')| \leq |E'_X| - |M'|$. Therefore, $\mathcal{T}'$ contains $l - |M'| = |X - V(M')| + |M'| \leq |E'_X| - |M'| + |M'| = |E'_X| = k_l^* \leq k$ trees. The lemma then follows. □

3.3 Case two: G' does not contain any light connected components

We then deal with Case 2: there are no light connected components in G'. In this case $l = 0$ and G' contains only h heavy connected components. A tree cover $\mathcal{T}$ of G can be constructed as follows.

For the MST T_i of each connected component CC_i, if $w(T_i) \geq \frac{8}{3}B$, T_i can be decomposed into several subtrees such that the weight of each subtree is no more than 2β by Lemma 1, where $\beta = \frac{4}{3}B$. These subtrees are then added to $\mathcal{T}$; otherwise ($B \leq w(T_i) < \frac{8}{3}B$), T_i is added to $\mathcal{T}$ directly, $1 \leq i \leq h$. We claim that the tree cover $\mathcal{T}$ of G contains no more than k trees.

Lemma 5: Given a metric graph $G = (V, E)$, assume that $B \geq B^*$, let G' be a subgraph of G after the removal of all edges with weight greater than $B/3$. Assume that each connected component of G' is a heavy connected component. Then, the constructed tree cover $\mathcal{T}$ by the above approach is a k-tree cover of G, and the maximum tree weight is no more than $\frac{8}{3}B$.

Proof: We give a lower bound on the weighted sum of trees in $\mathcal{T}$ first. Assume that there are x MSTs of the h heavy connected components with weight in the interval $[B, \frac{8}{3}B)$, where $0 \leq x \leq h$. Then, each of the rest $h - x$ MSTs has weight at least $\frac{8}{3}B$. Following the construction of $\mathcal{T}$, the x MSTs are directly put into $\mathcal{T}$, and each of the $h - x$ MSTs are decomposed into subtrees and the average weight of the decomposed subtrees is at least $\frac{4}{3}B$ by Lemma 1. Then, the $h - x$ MSTs are decomposed into $|\mathcal{T}| - x$ subtrees in $\mathcal{T}$ with average tree weight no less than $\frac{4}{3}B$. Thus, we have

$$\begin{aligned} w(\mathcal{T}) &\geq (|\mathcal{T}| - x) \cdot \frac{4}{3}B + x \cdot B \\ &\geq |\mathcal{T}|\frac{4}{3}B - \frac{h}{3}B \quad \text{since } x \leq h. \end{aligned} \tag{1}$$

We then estimate the upper bound of $\sum_{i=1}^{h} w(T_i)$, using the optimal solution OPT. Assume that $OPT = \{C_1^*, C_2^*, \ldots, C_k^*\}$. By removing all edges with weight greater than $B/3$ from OPT, each cycle C_i^* can be partitioned into either one connected component $CC_{i,1}^*$ if none or one edge is removed from it, or two connected components $CC_{i,1}^*$ and $C_{i,2}^*$ if two edges are removed from it. Following our assumption that there are only h heavy connected components in G' after removing all edges with weight greater than $B/3$ from G, then, each $CC_{i,1}^*$ and/or $CC_{i,2}^*$ are in one of h heavy connected components. Thus, all k cycles in OPT are heavy cycles, i.e., $k_l^* = k_b^* = 0$ and $k_h^* = k$. Suppose the removal of edges with weight greater than $B/3$ results in p heavy cycles that have two connected components and q cycles have only one connected components, where $p + q = k_h^* = k$. Thus, there are $2p + q$ connected components derived from OPT after the removal of edges with weight greater than $B/3$, and the weighted sum of these connected components is no more than $pB/3 + qB$, since each of the p cycles has been removed two edges with weight greater than $B/3$ and the weight of each cycle is no more than B^* with $B \geq B^*$. We can merge these $2p + q$ connected components into h connected components by adding exactly $2p+q-h$ edges with weight no more than $B/3$ in G', as G' contains h heavy connected components. Since the weighted sum of the MSTs of these h connected components is the minimum weighted sum of a forest of h trees spanning all vertices in G, then,

$$\begin{aligned} \sum_{i=1}^{h} w(T_i) &\leq pB/3 + qB + (2p+q-h)B/3 \\ &\leq \frac{4}{3}kB - \frac{h}{3}B \ \text{ as } p+q = k_h^* \leq k. \end{aligned} \quad (2)$$

Since $w(\mathcal{T}) = \sum_{i=1}^{h} w(T_i)$, we have $|\mathcal{T}| \leq k$ by combining Eq. (1) and Eq. (2). □

3.4 Case three: G' contains both light and heavy connected components

We now deal with Case three. We assume that G' contains l light connected components $CC_1, CC_2, \ldots, CC_l$ and h heavy connected components $CC_{l+1}, CC_{l+2}, \ldots, CC_{l+h}$, i.e., $l \neq 0$ and $h \neq 0$. For each light connected component CC_i of G', denote by $w_{min}(CC_i)$ the minimum edge weight $w(e)$ between the vertices in CC_i and its *nearest heavy connected component* CC_j with $l+1 \leq j \leq l+h$ if there is one edge in G with weight no greater than $B/2$, i.e., $w(e) = \min_{e'=(u,v)\in E}\{w(e') \mid u \in CC_i, v \in CC_{j'}, w(e') \leq B/2, l+1 \leq j' \leq l+h\}$. Otherwise, $w_{min}(CC_i) = \infty$. Define $A(CC_i) = w(T_i) + w_{min}(CC_i)$, $1 \leq i \leq l$. The general strategy for this case is to reduce it to cases one and two, respectively. For the sake of convenience, in the following we initially assume that the k_l^* light cycles in OPT are given, under this assumption we show that there is a k-tree cover of G. We later show how to find a k-tree cover of G by removing the assumption.

3.4.1 The k_l^* light cycles in OPT are given

Recall that the k cycles in OPT have been classified into k_l^* light cycles, k_h^* heavy cycles, and k_b^* bad cycles, where $k_l^* + k_h^* + k_b^* = k$. Given the k_l^* light cycles in OPT, we first construct the auxiliary multi-graph $H' = (X, E'_X)$ as we did in Case one, where X is the set of vertices corresponding to the l light connected components of G'. The edge set E'_X is defined by the k_l^* light cycles in OPT. The multi-graph H' may contain vertices without any adjacent edges including self-loops, we term these vertices as the *isolated vertices*, which correspond to the light connected components of G' that contain only the vertices from bad cycles. Clearly, $|E'_X| = k_l^*$. Let M' be a maximum matching of H'. Then, each vertex in X is either matched with another vertex or unmatched at all. Let a^* be the number of unmatched isolated vertices and b^* the number of unmatched vertices that have adjacent edges including self-loops. Clearly $0 \leq a^*, b^* \leq l$ and $|M'| = (l - a^* - b^*)/2$.

We then construct an auxiliary weighted graph $H'_{a^*,b^*} = (Y, E^*_Y)$ based on the maximum matching M' of H' as follows. Y contains l *regular vertices*, corresponding to the l light connected components $CC_1, \ldots, CC_l$ of G', a^* *heavy vertices* representing the a^* light connected components will be merged to at most a^* heavy connected components of G', and b^* *null vertices* which imply the MSTs of these light connected components that will be in the k-tree cover of G. We refer to a heavy connected component CC_j that has been enlarged by merging one or multiple light connected components into it as the *updated heavy connected component* CC'_j, $l+1 \leq j \leq l+h$. There is an edge in E^*_Y between two regular vertices v_i and v_j if there are two connected components $CC^*_{x,1}$ and $CC^*_{x,2}$ derived from a light cycle $C^*_x \in OPT$ after the removal of all edges with weight greater than $B/3$ from it, and $CC^*_{x,1}$ and $CC^*_{x,2}$ are in two light connected components CC_i and CC_j of G' respectively. The weight of this edge is zero. There is an edge in E^*_Y between a regular vertex v_i and each of the a^* heavy vertices if $A(CC_i) \neq \infty$, and the weight of this edge is $A(CC_i)$. There is an edge in E^*_Y between every null vertex and every regular vertex with weight zero.

It is easily shown that a minimum weighted perfect matching in H'_{a^*,b^*} can be found based on the maximum matching M' in H'. Let $M(H'_{a^*,b^*})$ be the minimum weighted perfect matching. Then, $M(H'_{a^*,b^*}) = \{(v_i, v_j) \mid (v_i, v_j) \in M'\} \cup \{(v_i, a\ null\ vetrex) \mid\ v_i$ is an unmatched vertex in H' incident to at least an edge in $H'\} \cup \{(v_i, a\ heavy\ vertex)\ \mid v_i$ is an unmatched isolated vertex in H' and $A(CC_i) \neq \infty\}$.

A k-tree cover of G, $\mathcal{T}^*$, then can be found based on $M(H'_{a^*,b^*})$, where $\mathcal{T}^* = \mathcal{T}^*(S_1) \cup \mathcal{T}^*(S_2)$, S_1 consists of all light connected components that are either matched with another light connected components in M' or unmatched but incident to at least one edge in H', and S_2 consists of all updated heavy connected components.

$\mathcal{T}^*(S_1)$ consists of MSTs formed by each pair of matched light connected components in M' or the MST of a light connected component that is matched with a null vertex in $M(H'_{a^*,b^*})$. It is easy to see that the maximum tree weight among the trees in $\mathcal{T}^*(S_1)$ is no more than $\frac{5}{2}B$. It can be shown that the number of trees in $\mathcal{T}^*(S_1)$ is no more than k_l^* through a reduction to Case one as follows. A subgraph H'' of H' is obtained by the removal of all isolated vertices from graph H'. Following the similar argument in Case one, we have $|\mathcal{T}^*(S_1)| = b^* + \frac{l-a^*-b^*}{2} \leq k_l^*$.

$\mathcal{T}^*(S_2)$ is constructed as follows. Assume that there is a matched edge in $M(H'_{a^*,b^*})$ between a regular vertex v_i (or light connected component CC_i) and a heavy vertex. Let CC_j be the nearest heavy connected component of CC_i. Merging CC_i to CC_j results in an updated heavy connected component CC'_j. Let T'_j be the MST of CC'_j. Then, $w(T'_j) = w(T_j) + A(CC_i)$, $l+1 \leq j \leq l+h$. Now, the problem becomes Case two, where there are h updated heavy connected components. A set of trees with weight no more than $\frac{8}{3}B$ can be found through applying the tree decomposition on each T'_j for all j with $l+1 \le j \le l+h$. We then show that $|\mathcal{T}^*(S_2)| \le k^*_b + k^*_h$, thus there is a tree cover of G with no more than $|\mathcal{T}^*| = |\mathcal{T}^*(S_1)| + |\mathcal{T}^*(S_2)| \le k^*_l + (k^*_b + k^*_h) = k$ trees.

Let us define some notations first. Recall that each bad cycle $C^*_i \in OPT$ has been divided into two connected components $CC^*_{i,1}$ and $CC^*_{i,2}$ after the removal of two edges with weight greater than $B/3$ from it, and one of them is in a light connected component CC_x and the other is in a heavy connected component CC_y of G' with $1 \le x \le l$ and $l+1 \le y \le l+h$. For the sake of discussion convenience, we further assume that $CC^*_{i,1}$ is the connected component that is contained in a light connected component CC_x of G'. Define the *excess weight* of each bad cycle C^*_i, $w_{excess}(C^*_i)$, as the weight of the connected component $CC^*_{i,1}$ of C^*_i in the light connected component plus the smaller edge weight of the two removed edges $e_{i,1}$ and $e_{i,2}$. For example, assume that $CC^*_{i,1}$ is the connected component of C^*_i in the light connected component of G' and $e_{i,1}$ and $e_{i,2}$ are the two removed edges from C^*_i with $w(e_{i,1}) \le w(e_{i,2})$. Then,

$$w_{excess}(C^*_i) = w(CC^*_{i,1}) + w(e_{i,1}). \tag{3}$$

Denote by w_{excess} the sum of excess weights of the k^*_b bad cycles in OPT.

We show $|\mathcal{T}^*(S_2)| \le k^*_b + k^*_h$ by the following lemma.

Lemma 6: Let T'_j be the MST of CC'_j for all j with $l+1 \le j \le l+h$. Notice that if no light connected component is merged to CC_j, CC'_j is the original CC_j itself. Let $\mathcal{T}^*(S_2)$ be the set of trees after applying the tree decomposition on each T'_j for all j with $l+1 \le j \le l+h$ where $\beta = \frac{4}{3}B$. Then, $|\mathcal{T}^*(S_2)| \le k^*_b + k^*_h$.

Proof: In the following we show that there are no more than $k^*_b + k^*_h$ trees with bounding weights by decomposing each trees T'_j for every j with $l+1 \le j \le l+h$.

Using a similar argument as we did to obtain inequality (1) in Case two, we have,

$$\sum_{j=l+1}^{l+h} w(T'_j) \ge |\mathcal{T}^*(S_2)|\frac{4}{3}B - \frac{h}{3}B. \tag{4}$$

Then, we show that

$$\sum_{j=l+1}^{l+h} w(T'_j) \le (k^*_b + k^*_h)\frac{4}{3}B - \frac{h}{3}B. \tag{5}$$

Inequalities (4) and (5) imply that $|\mathcal{T}^*(S_2)| \le (k^*_b + k^*_h)$.

We show that inequality (5) holds due to $\sum_{j=l+1}^{l+h} w(T_j) \le (k^*_b + k^*_h)\frac{4}{3}B - \frac{h}{3}B - w_{excess}$ and $w(M(H'_{a^*,b^*})) = \sum_{j=l+1}^{l+h}(w(T'_j) - w(T_j)) \le w_{excess}$.

With a similar argument as we did to obtain inequality (2) in Case two, it can be shown that

$$\sum_{j=l+1}^{l+h} w(T_j) \le (k^*_b + k^*_h)\frac{4}{3}B - \frac{h}{3}B - w_{excess}, \tag{6}$$

where T_j is the MST of CC_j, omitted.

We now show that $w(M(H'_{a^*,b^*})) = \sum_{j=l+1}^{l+h}(w(T'_j) - w(T_j)) \le w_{excess}$ as follows.

Following the construction of H'_{a^*,b^*}, $w(M(H'_{a^*,b^*}))$ is the sum of all $A(CC_i)$ of light connected components in G' that isolated vertices in H' correspond to, where $1 \le i \le l$. We show that, for each light connected component CC_i that an isolated vertex in H' corresponds to, $A(CC_i)$ is no more than the sum of excess weight of the bad cycles that derived connected components by the removal of edges with weight greater than $B/3$ are contained in CC_i. Then, the inequality holds.

For each light connected component CC_i that an isolated vertex v_i in H' corresponds to, assume that CC_i contains t connected components derived from t bad cycles $C^*_{i_1}, \ldots, C^*_{i_t}$ with $1 \le i_j \le k$ and $1 \le j \le t$. Denote by $CC^*_{i_1,1}, CC^*_{i_2,1}, \ldots, CC^*_{i_t,1}$ the t connected components contained in CC_i and $e_{i_1,1}, \ldots, e_{i_t,1}$ the t smaller weight edges among the $2t$ removed edges with weight greater than $B/3$ from the t bad cycles. Then, $B/3 < w(e_{i_j,1}) \le B/2$, $1 \le j \le t$. Let e be the cheapest edge by merging CC_i to its nearest heavy connected component. As $A(CC_i) = w(T_i) + w(e)$ and $\sum_{j=1}^{t} w_{excess}(C^*_{i_j}) = \sum_{j=1}^{t} w(CC^*_{i_j,1}) + \sum_{j=1}^{t} w(e_{i_j,1})$, we then show that

$$w(T_i) + w(e) \;\le\; \sum_{j=1}^{t} w(CC^*_{i_j,1}) + \sum_{j=1}^{t} w(e_{i_j,1}). \tag{7}$$

By the definition of edge e, we have that $w(e) \le \min\{w(e_{i_j,1}) \mid 1 \le j \le t\}$. Assume that $w(e_{i_t,1}) = \min_{j=1}^{t} w(e_{i_j,1})$, then, $w(e) \le w(e_{i_t,1})$. We then show that $w(T_i) \le \sum_{j=1}^{t} w(CC^*_{i_j,1}) + \sum_{j=1}^{t-1} w(e_{i_j,1})$. We notice that the t connected components $CC^*_{i_j,1}$ in CC_i with $1 \le j \le t$ can become a single connected subgraph spanning all vertices in CC_i, by adding extra $t-1$ edges with weight no greater than $B/3$. Then, $w(T_i) \le \sum_{j=1}^{t} w(CC^*_{i_j,1}) + (t-1)B/3 \le \sum_{j=1}^{t} w(CC^*_{i_j,1}) + \sum_{j=1}^{t-1} w(e_{i_j,1})$ as T_i is an MST of CC_i and $B/3 < w(e_{i_j,1}) \le B/2$ for j with $1 \le j \le t$. We then conclude that $\sum_{j=l+1}^{l+h} w(T'_j) \le \frac{4}{3}(k^*_b + k^*_h)B - \frac{h}{3}B$, which implies that there are no more than $k^*_b + k^*_h$ decomposed trees with the maximum tree weight no greater than $\frac{8}{3}B$ according to Lemma 1. □

3.4.2 Without the knowledge of the k^*_l light cycles

The above approximate solution obtained is based on an important assumption, that is, the k^*_l light cycles in OPT are given. Having the k^*_l light cycles, an auxiliary graph H' then is constructed and a maximum matching M' of H' is found. The values of a^* and b^* are then obtained through M'. A weighted auxiliary graph H'_{a^*,b^*} based on a^* and b^* is constructed and a minimum weighted perfect matching $M(H'_{a^*,b^*})$ in H'_{a^*,b^*} must exist. A k-tree cover $\mathcal{T}^*$ of G finally is derived based on $M(H'_{a^*,b^*})$. In the following we show how to find the approximation solution to the problem without knowing the k^*_l light cycles in OPT.

As a^* and b^* are non-negative integers in the interval $[0, l]$ and $a^*+b^* \leq l$, there are in total $\sum_{a=0}^{l}(l-a) = \frac{l(l+1)}{2}$ possible pairs of values of a and b. a^* and b^* must be one of these $\frac{l(l+1)}{2}$ pairs. In each pair with a and b, we construct another auxiliary weighted graph $H_{a,b}$ without the knowledge of OPT, which is constructed later. We show that when $a = a^*$ and $b = b^*$, H'_{a^*,b^*} is a spanning subgraph of H_{a^*,b^*}. Thus, the minimum weighted perfect matching $M(H'_{a^*,b^*})$ in H'_{a^*,b^*} is one perfect matching in H_{a^*,b^*}. Then, instead of using $M(H'_{a^*,b^*})$ of H'_{a^*,b^*} to find $\mathcal{T}^*$, we can use the minimum weighted perfect matching $M(H_{a^*,b^*})$ of H_{a^*,b^*} to find a k-tree cover of G, $\mathcal{T}$. The difference between H'_{a^*,b^*} and H_{a^*,b^*} is that the former can be constructed if the k_l^* light cycles are given while the later does not need this knowledge.

The weighted auxiliary graph $H_{a,b} = (Y, E_Y)$ is constructed as follows. Y contains l regular vertices v_i, $1 \leq i \leq l$, corresponding to the l light connected components of G', a heavy vertices, and b null vertices. There is an edge in E_Y between two regular vertices v_i and v_j if there is an edge in G with weight no greater than $B/2$ between the vertices in two light connected components CC_i and CC_j, and the weight of the edge is zero for any i and j with $1 \leq i, j \leq l$. There is an edge in E_Y between every null vertex and every regular vertex and its weight is zero. There is an edge in E_Y between a regular vertex v_i and every heavy vertex if light connected component CC_i has a finite value $A(CC_i)$ and the weight of the edge is $A(CC_i)$ for some i with $1 \leq i \leq l$.

If there is a minimum weighted perfect matching in $H_{a,b}$, let $M(H_{a,b})$ be the minimum weighted perfect matching. A k-tree cover $\mathcal{T}_{a,b}$ of G can be constructed based on $M(H_{a,b})$ as follows.

Initially, $\mathcal{T}_{a,b} = \emptyset$. For each matched edge $(x, y) \in M(H_{a,b})$, assume that x must be a regular vertex with $x = v_i$. Then, if y is a null vertex, the MST T_i of CC_i is added to $\mathcal{T}_{a,b}$. If $y = v_j$ is a regular vertex with $i \neq j$, an MST $T_{i,j}$ is obtained by joining T_i of CC_i and T_j of CC_j with a cheapest edge between them. $T_{i,j}$ is added to $\mathcal{T}_{a,b}$. Otherwise (y must be a heavy vertex), let CC_j be the nearest heavy connected component of CC_i, i.e., $A(CC_i)$ is equal to the minimum edge weight of an edge e between the vertices in CC_i and CC_j plus $w(T_i)$. CC_i will be merged to CC_j. Let CC'_j be the updated heavy connected component and T'_j the MST of CC'_j. Then, $T'_j = T_i \cup T_j \cup \{e\}$ and $w(T'_j) = w(T_j) + A(CC_i)$. For each updated heavy connected component CC'_j (notice that a CC'_j may be the results by merging multiple light connected components). If $w(T'_j) < \frac{8}{3}B$, T'_j is added to $\mathcal{T}_{a,b}$ directly. Otherwise, apply tree decomposition on T'_j to split away subtrees from T'_j by Lemma 1 with $\beta = \frac{4}{3}B$. Add these split subtrees into $\mathcal{T}_{a,b}$.

It is easily show that the maximum tree weight of the trees in $\mathcal{T}_{a,b}$ is no more than $\frac{8}{3}B$. We then show that when $a = a^*$ and $b = b^*$, $|\mathcal{T}_{a^*,b^*}| \leq k$ as follows.

By the definitions of H_{a^*,b^*} and H'_{a^*,b^*}, we know that H'_{a^*,b^*} is a spanning subgraph of H_{a^*,b^*}. Thus, there must be a perfect matching in H_{a^*,b^*} as there is a perfect matching in H'_{a^*,b^*}.

We note that there are l regular vertices in H_{a^*,b^*}. Within the perfect matching $M(H_{a^*,b^*})$, a^* regular vertices are matched to the a^* heavy vertices, b^* regular vertices are matched to b^* null vertices, and the rest of regular vertices match themselves, i.e., $|M(H_{a^*,b^*})| = a^*+b^*+(l-a^*-b^*)/2$. The number of trees obtained from the matched edges between the regular vertices and a regular vertex and a null vertex is $b^*+(l-a^*-b^*)/2 \leq k_l^*$ with the maximum tree weight $\frac{5}{2}B$. We also notice that the weight of the minimum weighted matching $M(H_{a^*,b^*})$ in H_{a^*,b^*} is no more than that of the minimum weighted perfect matching $M(H'_{a^*,b^*})$ in H'_{a^*,b^*}, so $w(M(H_{a^*,b^*})) \leq w(M(H'_{a^*,b^*})) \leq w_{excess}$. Then, the weighted sum of the MSTs of all updated heavy connected components is $\sum_{j=l+1}^{l+h} w(T'_j) = \sum_{j=l+1}^{l+h} w(T_j) + w(M(H_{a^*,b^*})) \leq \frac{4}{3}(k_b^*+k_h^*)B - \frac{h}{3}B$ by combining inequality (6). Apply the tree decomposition to each T'_j for all j with $l+1 \leq j \leq l+h$, no more than $k_b^*+k_h^*$ trees with the maximum tree weight $\frac{8}{3}B$ can be derived by combining Inequality (4). Thus, the number of trees covering all vertices in G is no more than $k_l^* + (k_b^* + k_h^*) = k$.

3.5 Algorithm

The detailed algorithm is described in `Algorithm` 1.

Step 2 of `Algorithm` 1 is explained by the following lemma.

Lemma 7: In `Algorithm` 1, if $l + h \geq 8k$, there is no k-tree cover of G with maximum tree weight $\frac{8}{3}B$.

Proof: We show the claim by contradiction. Suppose that there is a k-tree cover with the maximum tree weight $\frac{8}{3}B$ when $l + h \geq 8k$. We assume that graph $H_{a,b}$ from which the tree cover $\mathcal{T}$ is derived contains a heavy vertices and b null vertices. For each light connected component merged to a heavy component with an edge e at Step 13, we have $w(e) > B/3$, and for the MST T_j of a heavy connected component CC_j, we have $w(T_j) \geq B$ for all j with $l+1 \leq j \leq l+h$. We only consider the trees in $\mathcal{T}_{heavy}$ by decomposing the MSTs of updated heavy connected components at Step 14 of the algorithm. Then,

$$w(\mathcal{T}_{heavy}) = \sum_{j=l+1}^{l+h} w(T'_j) \ \geq \ h \cdot B + a \cdot B/3. \qquad (8)$$

On the other hand, $\mathcal{T}_{heavy}$ contains no more than $k - (l - a - b)/2 - b$ trees with the maximum tree weight no greater than $\frac{8}{3}B$. Therefore,

$$w(\mathcal{T}_{heavy}) \leq (k - (l - a - b)/2 - b) \cdot 8B/3. \qquad (9)$$

Combining inequities (8) and (9), we have

$$8k \ > \ (l + h) + 3(l - a) + 4b + 2h \ \geq \ l + h, \qquad (10)$$

the last inequality holds due to $l \geq a$. □

We thus have the following Lemma.

Lemma 8: If $B \geq B^*$, `Algorithm` 1 will deliver a k-tree cover of G with the maximum tree weight $\frac{8}{3}B$.

Algorithm 1 `Rootless-Tree-Cover`

Input: $G = (V, E)$, a metric $w : E \mapsto \mathbb{Z}^+$, k, and B
Output: a set $\mathcal{T}$ of trees which is a k-tree cover covering all vertices in G with the maximum tree weight $\frac{8}{3}B$

Let $CC_1, \ldots, CC_{l+h}$ be the l light and h heavy connected components of G after the removal of all edges with weight greater than $B/3$;
if $l + h \geq 8k$ **then**
  return "B is too low"; EXIT;
end if
for $a \leftarrow 0$ to l **do**
  for $b \leftarrow 0$ to $l - a$ **do**
    $\mathcal{T}_{a,b} \leftarrow \emptyset$; /* the k-tree cover set*/
    Construct a graph $H_{a,b}$ that contains l regular vertices, a heavy vertices, and b null vertices;
    Find a minimum weighted perfect matching $M(H_{a,b})$ in $H_{a,b}$;
    if there is such a perfect matching in $H_{a,b}$ **then**
      For each matched edge between two regular vertices v_i and v_j, an MST $T_{i,j}$ is obtained by joining the MST T_i of CC_i and the MST $_j$ of CC_j through a cheapest edge between them, and add $T_{i,j}$ to $\mathcal{T}_{a,b}$;
      For each regular vertex v_i that is matched to a null vertex, add the MST T_i of CC_i to $\mathcal{T}_{a,b}$;
      For each regular vertex v_i that is matched to a heavy vertex, the light connected component CC_i is merged to its nearest heavy connected component CC_j, denote by CC'_j the updated heavy connected component and T'_j be the MST of CC'_j;
      For the MST T'_j of each updated heavy connected component, if $w(T'_j) < \frac{8}{3}B$, add T'_j to $\mathcal{T}_{a,b}$; otherwise, T'_j is decomposed into several subtrees, such that the weight of each split away subtree is no more than $\frac{8}{3}B$ by Lemma 1, where $\beta = \frac{4}{3}B$, add these subtrees to $\mathcal{T}_{a,b}$;
      if $|\mathcal{T}_{a,b}| \leq k$ **then**
        return $\mathcal{T}_{a,b}$
      end if
    end if
  end for
end for
return "B is too low"

3.6 Strongly polynomial approximation

The proposed approximation algorithm above is a polynomial algorithm only when the maximum weight of edges in G is bounded by a polynomial of 2^n. Thus, the optimal value B^* can be found within a polynomial number of iterations by performing binary search on the interval $[0, n \cdot w_m]$, where $n = |V|$, $m = |E|$, and $w_m = \max_{e \in E}\{w(e)\}$. Otherwise, the proposed algorithm is a pseudo-polynomial algorithm because its time complexity depends on the maximum value of edge weights. In the following, for a given constant $0 < \epsilon < 1$, we show that the number of guesses B of the value of B^* is a polynomial of n by adopting a technique in [10].

Assume that the edge weights in G are sorted in increasing order, denote by $w_1 \leq w_2 \leq \cdots \leq w_m$. It is obvious that $B^* \leq n \cdot w_m$. Using different guesses B of B^*, `Algorithm` 1 proceeds iteratively until a feasible solution is found. Specifically, the initial guess B of B^* is $B = w_m$. When $B = w_m$, if `Algorithm` 1 returns that "B is too low", which means that this guess is too small, B^* must be in the interval $[w_m, n \cdot w_m]$. The algorithm then guesses the next B by binary search until a guess $(1 + \frac{\epsilon}{16/3})B$ of B^* is found such that `Algorithm` 1 delivers a solution with the maximum cycle weight no greater than $(\frac{16}{3} + \epsilon)B$. That is, the search interval now is narrowed down in the interval $[B, (1 + \frac{\epsilon}{16/3})B]$. Clearly $B < B^*$, and the upper bound of B^* is $(1 + \frac{\epsilon}{16/3})B$. Thus, $(\frac{16}{3} + \epsilon)B \leq (\frac{16}{3} + \epsilon)B^*$, and the number of iterations (by binary search) is bounded by $\lceil \log(\frac{n \cdot w_m}{16\epsilon \cdot w_m/3}) \rceil = O(\log n + \log \frac{1}{\epsilon})$. Otherwise, we try the next B with value no greater than the current value w_m to see whether `Algorithm` 1 delivers a solution, too. The next B thus is one of the m possible values $w_1, \cdots, w_m$ which can be found using binary search in the sequence of edge weights until an index i of an edge weight is found such that `Algorithm` 1 returns "B is too low" when $B = w_i$, while `Algorithm` 1 returns a k-tree cover of G with the maximum tree weight $\frac{8}{3} \cdot w_{i+1}$ when $B = w_{i+1}$, where $i \in [1, m-1]$.

If $w_{i+1} \leq \frac{n^2}{\epsilon} \cdot w_i$, then the number of iterations for searching a proper B in the interval $[w_i, w_{i+1}]$ is strongly polynomial with an approximation ratio of $5\frac{1}{3} + \epsilon$ as discussed in the above. Otherwise, denote by $w' = \frac{n^2}{\epsilon} \cdot w_i$. If `Algorithm` 1 can deliver a k-tree cover with the maximum tree weight $\frac{8}{3}w'$ when $B = w'$. It then performs the binary search in the interval $[w_i, w']$ to find a better B through a series of iterations by binary search. Thus, the algorithm is strongly polynomial with an approximation ratio of $5\frac{1}{3} + \epsilon$. Otherwise (`Algorithm` 1 returns that "B is too low" when $B = w'$), then $B^* \geq w_{i+1}$, which is shown in the following. Suppose that $B^* < w_{i+1}$, then the cycles in OPT contain only the edges with weight no greater than w_i. Thus, the maximum cycle weight among the cycles in OPT is at most $n \cdot w_i \leq n^2 \cdot w_i = \epsilon \cdot w' < \epsilon \cdot B^* < B^*$, since $w' < B^*$ and $0 < \epsilon < 1$. This contradicts the definition of B^*. Note that `Algorithm` 1 can find a solution with the maximum cycle weight $\frac{16}{3} \cdot w_{i+1}$, then, $\frac{16}{3} \cdot w_{i+1} < \frac{16}{3}B^*$. We thus have the following theorem.

Theorem 1: Given a metric complete graph $G = (V, E)$ and a positive integer k, there is a $(5\frac{1}{3}+\epsilon)$-approximation algorithm for the rootless min-max cycle cover problem in G, which takes $O((n^2k^2+k^5)(\log n+\log \frac{1}{\epsilon}))$ time, where $n = |V|$ and ϵ is a constant with $0 < \epsilon < 1$.

Proof: Combining Lemmas 2 and 8 and the above discussions, the approximation ratio of the proposed algorithm is straightforward, omitted.

The rest is to analyze the time complexity of the proposed algorithm. The number of iterations of the binary search for the optimal B^* is at most $O(\log n + \log \frac{1}{\epsilon})$ by the above discussion. In each iteration, `Algorithm` 1 is invoked and its time complexity is analyzed as follows.

It takes $O(n^2)$ time to obtain a subgraph G' of G by removing the edges with weight greater than $B/3$. Finding the MSTs of all connected components in G' takes $O(n^2)$ time. Within `Algorithm` 1, there are no more than $\frac{l(l+1)}{2} = O(l^2)$ tree covers $\mathcal{T}_{a,b}$ of G to be constructed. For each tree cover $\mathcal{T}_{a,b}$, the auxiliary graph $H_{a,b}$ can be constructed in time $O(n^2)$ as the edges in $H_{a,b}$ can be determined by the number of edges in G. It is also known that $H_{a,b}$ contains no more than $2l$ vertices, it takes $O((2l)^3) = O(l^3)$ time to finding a minimum weighted perfect matching by applying an algorithm in [12]. The construction of tree cover $\mathcal{T}_{a,b}$ takes $O(n)$ time as the tree decomposition can be implemented by depth-first search on the MSTs of updated heavy connected components. Thus, `Algorithm` 1 takes $O(n^2) + O(l^2)(O(n^2)+O(l^3)+O(n)) = O(n^2l^2+l^5) = O(n^2k^2+k^5)$ as $l \le l+h \le 8k$ by Lemma 7. Thus, the time complexity of the proposed algorithm is $O((n^2k^2+k^5)(\log n+\log \frac{1}{\epsilon}))$. □

4 ALGORITHM FOR THE UNCAPACITATED ROOTED MIN-MAX CYCLE COVER PROBLEM

In this section, we focus on the uncapacitated rooted min-max cycle cover problem, for which we devise a $(6\frac{1}{3} + \epsilon)$-approximation algorithm as follows.

4.1 Algorithm

The proposed algorithm for this problem is to find no more than k trees covering all vertices in $V - D$ with each tree having exactly one depot from D, such that the maximum tree weight is no greater than $\frac{19}{6}B$ when $B \ge B_r^*$. To this end, `Algorithm` 1 for the rootless min-max tree cover problem will be invoked. That is, it first finds k trees covering all vertices in $V - D$ with the maximum tree weight $\frac{8}{3}B$. It then connects each found tree to its nearest depot by an edge with weight at most $B/2$. The detailed algorithm is described in `Algorithm` 2.

Algorithm 2 `Uncapacitated-Rooted-Tree-Cover`

Input: $G = (V, E)$, a metric $w : E \mapsto \mathbb{Z}^+$, k, a depot set D ($D \subset V$), and B

Output: a set $\mathcal{S}$ which is a rooted k-tree cover with maximum tree weight at most $\frac{19}{6}B$.

1: `Call Algorithm` 1 with inputs $G - D = G(V - D, E - \{(r, v) \mid (r, v) \in E, r \in D, v \in V - D\})$, w, k, and B.
2: **if** `Algorithm` 1 delivers a k-tree cover $\mathcal{T}$ of graph $G - D$ with maximum tree weight $\frac{8}{3}B$ **then**
3: $\quad\mathcal{S} \leftarrow \emptyset$;
4: $\quad$For each $T_i \in \mathcal{T}$, connect T_i to its nearest depot r_i, and add the resulting tree rooted at r_i into $\mathcal{S}$;
5: $\quad$**return** $\mathcal{S}$;
6: **else**
7: $\quad$**return** "B is too low";
8: **end if**

4.2 Algorithm analysis

The correctness and approximation ratio of the proposed algorithm are guaranteed by the following lemmas.

Assuming that OPT_r is an optimal solution to the problem and B_r^* is the optimal value. Define $w(v, D) = \min_{r\in D}\{w(v, r)\}$ as the minimum distance between each vertex v in $V - D$ and the depot set D. Then, there is an important property of an optimal (uncapacitated or capacitated) rooted min-max cycle cover of G: the maximum value of $w(v, D)$ for all vertices in $V - D$ is no greater than $B_r^*/2$.

Lemma 9: $\max_{v\in V-D}\{w(v, D)\} \le B_r^*/2$.

Proof: Let v_{max} be the vertex in $V - D$ such that $w(v_{max}, D) = \max_{v\in V-D}\{w(v, D)\}$. Assume that vertex v_{max} is in the cycle $C^* \in OPT_r$ and r_{C^*} is the depot of C^*. Then, $w(v_{max}, r_{C^*}) \le w(C^*)/2 \le B_r^*/2$ by Lemma 3. Following the definition of $w(v_{max}, D)$, we have $w(v_{max}, D) \le w(v_{max}, r_{C^*}) \le B_r^*/2$. □

Lemma 10: If $B \ge B_r^*$, `Algorithm` 2 will deliver an uncapacitated rooted k-tree cover of G with the maximum tree weight $\frac{19}{6}B$.

Proof: We first argue that `Algorithm` 1 delivers a k-tree cover of $G - D$ with the maximum tree weight $\frac{8}{3}B$ at Step 1 of `Algorithm` 2. Given the OPT_r, the depot vertices in the cycles of OPT_r can be shortcut, thus, a feasible k-cycle cover $\mathcal{C}$ of graph $G - D$ then can be obtained, and $\max_{C_i^*\in\mathcal{C}} w(C_i^*) \le B_r^*$. Let OPT be an optimal solution to the rootless min-max cycle cover problem in graph $G-D$ with the optimal value B^*, then, $B^* \le \max_{C_i^*\in\mathcal{C}} w(C_i^*) \le B_r^*$. Therefore, $B \ge B_r^* \ge B^*$. Thus, `Algorithm` 1 can find a k-tree cover $\mathcal{T}$ of graph $G - D$ with the maximum tree weight $\frac{8}{3}B$ by Lemma 8.

We then show that the minimum distance between each found tree and its nearest depot is no more than $B/2$ as $\min_{v\in T_i, r\in D}\{w(v, r)\} = \min_{v\in T_i}\{w(v, D)\} \le \max_{v\in V-D}\{w(v, D)\} \le B_r^*/2 \le B/2$. Then, the weight of each tree in $\mathcal{S}$ is no more than $\frac{8}{3}B + \frac{B}{2} = \frac{19}{6}B$. □

We thus have the following theorem.

Theorem 2: Given a metric complete graph $G = (V, E; w)$, a depot set $D \subset V$, and a positive integer k, there is a $(6\frac{1}{3} + \epsilon)$-approximation algorithm for the uncapacitated rooted min-max cycle cover problem in G, which takes $O((n^2k^2 + k^5)(\log n + \log \frac{1}{\epsilon}))$ time, where $n = |V|$ and ϵ is a given constant with $0 < \epsilon < 1$.

Proof: Following Lemmas 2, 10, and the similar analysis in the previous section, the analysis of the approximation ratio and the time complexity of the proposed algorithm is straightforward, omitted. □

5 ALGORITHM FOR THE CAPACITATED ROOTED MIN-MAX CYCLE COVER PROBLEM

In this section we devise a $(7+\epsilon)$-approximation algorithm for the capacitated rooted min-max cycle cover problem, in which each depot d in D has a maximum serving capacity on the number of vehicles it can serve. This general problem can be reduced to one special case of the problem, that is, each depot can only serve one vehicle and there are at least k depots in total [31], because each depot r with $f(r)$ serving capacity can be treated as $f(r)$ 'virtual' depots with a unit serving capacity, and the $f(r)$ virtual depots are located at the same location. Therefore, in the rest we only consider this special case of the problem.

We start with the following crucial lemma.

Lemma 11: Given a tree T with weight $w(T)$, assume that each edge in T has weight no more than B and $w(T) \geq 3B$, then T can be decomposed into x edge-disjoint trees $T_1, \ldots, T_x$ such that $\frac{3}{2}B \leq w(T_i) < 3B$ for each i with $1 \leq i \leq x-1$ and $B \leq w(T_x) < 3B$, where $B > 0$ and $2 \leq x \leq \lfloor \frac{w(T)}{3B/2} \rfloor$.

Proof: By Lemma 1, when β is set as $\frac{3}{2}B$, T can be decomposed into x edge-disjoint trees $T'_1, \ldots, T'_x$ with $\frac{3}{2}B \leq w(T'_i) < 3B$ for each i with $1 \leq i \leq x-1$, $w(T'_x) < 3B$, and $w(T'_{x-1}) + w(T'_x) \geq 3B$, where $2 \leq x \leq \lfloor \frac{w(T)}{3B/2} \rfloor$.

We now construct T_i from T'_i as follows. If $w(T'_i) \geq B$, $T_i = T^{'}_i$ for all i with $i = 1, \ldots, x$; otherwise $T_i = T^{'}_i$ for all i with $i = 1, \ldots, x-2$. The rest is to construct the last two trees T_{x-1} and T_x. Following the construction of $T^{'}_{x-1}$ and $T^{'}_x$, their union $T'_{x-1,x} = T^{'}_{x-1} \cup T^{'}_x$ is connected and the weight of $T'_{x-1,x}$ is within $3B \leq w(T'_{x-1,x}) < 4B$. We then split off a subtree $T^{''}_{x-1}$ from $T'_{x-1,x}$ with the bounded weight in the interval $[B, 2B)$. Denote by $T^{''}_x$ the leftover tree. Now, if $\frac{3}{2}B \leq w(T^{''}_{x-1}) \leq 2B$, then $w(T^{''}_x) = w(T'_{x-1,x}) - w(T^{''}_{x-1}) \geq 3B - 2B = B$ and $w(T^{''}_x) < 4B - \frac{3}{2}B < 3B$. For this case, $T_{x-1} = T^{''}_{x-1}$ and $T_x = T^{''}_x$. Otherwise $(B \leq w(T^{''}_{x-1}) < \frac{3}{2}B)$, then $w(T^{''}_x) \geq 3B - \frac{3}{2}B = \frac{3}{2}B$ and $w(T^{''}_x) < 4B - B = 3B$. $T_{x-1} = T^{''}_x$ and $T_x = T^{''}_{x-1}$. The lemma then follows. □

5.1 Algorithm

Let OPT_r be an optimal solution to the capacitated rooted cycle cover problem in G with the optimal value B^*_r. Assume that $B \geq B^*_r$. The idea of the proposed algorithm is to find no more than k trees covering all vertices in $V - D$ with the maximum tree weight $3B$ first, by applying the tree decomposition technique. It then connects the found trees to the depots in set D through a maximum matching in an auxiliary bipartite graph while ensuring that the shortest distance between the vertices in each tree and its matched depot is no more than $B/2$. The detailed algorithm is given in `Algorithm` 3.

5.2 Algorithm analysis

To show the correctness of `Algorithm` 3, we have:

Algorithm 3 `Capacitated-Rooted-Tree-Cover`

Input: $G = (V, E)$, a metric $w : E \mapsto \mathbb{Z}^+$, k, D ($D \subset V, |D| \geq k$), and B

Output: A rooted k-tree cover $\mathcal{S}$ of G in which each tree has a distinct root in D with the maximum tree weight $\frac{7}{2}B$.

1: A subgraph of $G - D$ is then obtained by removing all edges in $G - D$ with the weight greater than $B/2$. Assume that the subgraph contains y components: $CC_1, \ldots, CC_y$. Let T_i be the MST of CC_i for all i with $1 \leq i \leq y$;
2: $\mathcal{T} \leftarrow \emptyset$; $\mathcal{S} \leftarrow \emptyset$;
3: For each T_i, if $w(T_i) < 3B$, add T_i to $\mathcal{T}$; otherwise, T_i is decomposed into x_i edge-disjoint subtrees by Lemma 11, add the x_i subtrees into $\mathcal{T}$;
4: **if** $|\mathcal{T}| \leq k$ **then**
5:   Construct a bipartite graph $H = (\mathcal{T}, D, E')$. There is an edge in E' between a tree vertex $T \in \mathcal{T}$ and a depot $r \in D$ if there is an edge in G between a vertex in T and r with weight no more than $B/2$;
6:   Find a maximum matching $M(H)$ in H;
7:   **if** each tree vertex T is matched in $M(H)$ **then**
8:     For each tree $T \in \mathcal{T}$, connect T to its matched depot r with the cheapest edge between them and add the resulting tree rooted at r to $\mathcal{S}$;
9:     **return** $\mathcal{S}$
10:   **else**
11:     **return** "B is too low".
12:   **end if**
13: **else**
14:   **return** "B is too low".
15: **end if**

Lemma 12: If $B \geq B^*_r$, `Algorithm` 3 will find a k-tree cover in which each tree contains a distinct depot in D and no depot is contained by more than one tree, such that the maximum tree weight is $\frac{7}{2}B$.

Proof: Given an optimal solution OPT_r of the problem, a set OPT of segments (or lines) that do not contain any depots can be derived as follows.

Define $OPT \triangleq \{T^*_i \mid T^*_i = C^*_i - \{(r_i, x_i), (r_i, y_i)\}$ for all $C^*_i \in OPT_r, (r_i, x_i), (r_i, y_i) \in C^*_i\}$. That is, T^*_i is a segment of C^*_i by the removal of the depot r_i and its two adjacent edges from C^*_i. Then, $w(T^*_i) \leq B^*_r \leq B$.

Assume that the subgraph of $G - D$ obtained after the removal of all edges with weight greater than $B/2$ contains y connected components $CC_1, \ldots, CC_y$. Following Lemma 3, no edges in the segments of OPT will be removed at Step 1, these segments in OPT then are partitioned into y classes, depending on in which connected components of a subgraph they are contained and the subgraph is induced from $G - D$ by removing all edges with weight greater than $B/2$. Denote by OPT_i the set of segments contained by connected component CC_i for each i with $1 \leq i \leq y$. For the segments in OPT_i, denote by D^*_i the set of depots in their corresponding cycles in OPT_r and D^* as the union of all D_i^*. Clearly, $D_i^* \neq \emptyset$, $D_i^* \cap D_j^* = \emptyset$ if $i \neq j$, $\cup_{i=1}^{y} D_i^* = D^* \subseteq D$, $\sum_{i=1}^{y} |D_i^*| = |D^*| = k$, and $|OPT_i| = |D_i^*|$, $1 \leq i \leq y$.

In the following we show that (i) the number of trees in $\mathcal{T}$ obtained at Step 3 is no more than k; and (ii) each tree in $\mathcal{T}$ is matched to a different depot in D at Step 6.

We first show case (i): $|\mathcal{T}| \leq k$. Assume that $\mathcal{T} = \cup_{i=1}^{y} \mathcal{T}_i$ and $\mathcal{T}_i \cap \mathcal{T}_j = \emptyset$ if $i \neq j$, where $\mathcal{T}_i$ is the set of trees obtained by decomposing T_i at Step 3 for all i with $1 \leq i \leq y$. To this end, we only need to show that $|\mathcal{T}_i| \leq |D_i^*|$ for all i with $1 \leq i \leq y$. For each MST T_i of CC_i, if $w(T_i) < 3B$, then $|\mathcal{T}_i| = 1 \leq |D_i^*|$. Otherwise,

$$w(T_i) \geq (\frac{3}{2}|\mathcal{T}_i| - \frac{1}{2})B, \text{ by Lemma 11.} \quad (11)$$

The rest is to estimate an upper bound on $w(T_i)$ as follows. Through adding $|OPT_i| - 1$ edges in CC_i with weight no greater than $B/2$ to connect different segments of OPT_i, a connected component that spans all vertices in connected component CC_i can be obtained. Thus,

$$\begin{aligned} w(T_i) &\leq |OPT_i| \cdot B + (|OPT_i| - 1) \cdot B/2 \\ &= (\frac{3}{2}|OPT_i| - \frac{1}{2})B = (\frac{3}{2}|D_i^*| - \frac{1}{2})B. \quad (12) \end{aligned}$$

Combining inequalities (11) and (12), we have that $|\mathcal{T}_i| \leq |D_i^*|$. Thus, $|\mathcal{T}| = \sum_{i=1}^{y} |\mathcal{T}_i| \leq \sum_{i=1}^{y} |D_i^*| = k$.

We then show case (ii): each tree can be matched by a different depot in set D. As $D_i^* \cap D_j^* = \emptyset$ if $i \neq j$, we only need to show that each tree in $\mathcal{T}_i$ will match a depot in set D_i^* for all i with $1 \leq i \leq y$.

If $|\mathcal{T}_i| = 1$ which implies that there is only one tree in it, then there must have an edge with weight no greater than $B/2$ that connects the tree and a depot in D_i^* as $|D_i^*| \geq 1$. Otherwise ($|\mathcal{T}_i| \geq 2$), we have $w(T_i) \geq 3B$ by Lemma 11. Then, there must exist such a matching from the trees in $\mathcal{T}_i$ to depots in D_i^*, which is guaranteed by the Hall's Theorem [7] which says that for each subset $\mathcal{A}$ of $\mathcal{T}_i$, the neighbor set $N(\mathcal{A})$ of $\mathcal{A}$ satisfies $|N(\mathcal{A})| \geq |\mathcal{A}|$.

Consider any subset $\mathcal{A}$ of $\mathcal{T}_i$, its neighbor set $N(\mathcal{A})$ is a subset of D_i^* that the shortest distance from a vertex in a tree in $\mathcal{A}$ to a depot in the subset of D_i^* is no more than $B/2$. Let $OPT_i^*(\mathcal{A})$ denote the subset of segments in OPT_i that have non-empty intersections of the vertices in the segments and the trees in $\mathcal{A}$. Namely, $T_i^* \in OPT_i^*(\mathcal{A})$ if and only if there is a tree T in $\mathcal{A}$ such that $V(T) \cap V(T_i^*) \neq \emptyset$. Then, $|N(\mathcal{A})| \geq |OPT_i^*(\mathcal{A})|$ and $|OPT_i^*(\mathcal{A})| \geq |\mathcal{A}|$, which are shown as follows.

We start with that $|N(\mathcal{A})| \geq |OPT_i^*(\mathcal{A})|$. Since $B \geq B_r^*$ and the distance between each vertex and its depot in OPT_r is at most $B/2$, there is an edge in the constructed auxiliary graph $H = (\mathcal{T}, D, E')$ between a tree $T \in \mathcal{A}$ and a depot $r \in D_i^*$ if T intersects a segment T_l^* of the cycle C_l^* with depot at r. Hence, $|N(\mathcal{A})| \geq |OPT_i^*(\mathcal{A})|$.

We then show that $|OPT_i^*(\mathcal{A})| \geq |\mathcal{A}|$. Recall that each edge in a tree of $\mathcal{A}$ is an edge of the MST T_i of connected component CC_i. A subgraph G_i' of CC_i is obtained by removing all edges in each subtree of T_i in $\mathcal{A}$ and adding all edges of segments in $OPT_i^*(\mathcal{A})$. Then, G_i' will become a connected subgraph by adding no more than $(|OPT_i^*(\mathcal{A})| - 1)$ edges in CC_i with weight no greater than $B/2$, since CC_i itself is a connected component which contains only the edges with weight no greater than $\frac{B}{2}$. Denote by G_i'' the connected subgraph and $E_{\mathcal{A}}$ the set of no more than $(|OPT_i^*(\mathcal{A})| - 1)$ added edges. Then, $|E_{\mathcal{A}}| \leq |OPT_i^*(\mathcal{A})| - 1$. As T_i is an MST of CC_i,

$$w(G_i'') \geq w(T_i). \quad (13)$$

Meanwhile,

$$\begin{aligned} w(G_i'') &= w(G_i') + w(E_{\mathcal{A}}) \\ &= w(T_i) - w(\mathcal{A}) + w(OPT_i^*(\mathcal{A})) + w(E_{\mathcal{A}}). \quad (14) \end{aligned}$$

Combining inequalities (13) and (14) we have

$$w(OPT_i^*(\mathcal{A})) + w(E_{\mathcal{A}}) \geq w(\mathcal{A}). \quad (15)$$

According to Lemma 11, $w(\mathcal{A}) \geq (\frac{3}{2}|\mathcal{A}| - \frac{1}{2})B$, since $\mathcal{A}$ is a subset of $\mathcal{T}_i$. On the other hand,

$$\begin{aligned} w(OPT_i^*(\mathcal{A})) + w(E_{\mathcal{A}}) &\leq |OPT_i^*(\mathcal{A})| \cdot B \\ &\quad + (|OPT_i^*(\mathcal{A})| - 1) \cdot \frac{B}{2} \\ &= (\frac{3}{2}|OPT_i^*(\mathcal{A})| - \frac{1}{2})B. \quad (16) \end{aligned}$$

Therefore, $|OPT_i^*(\mathcal{A})| \geq |\mathcal{A}|$ by inequalities (15) and (16).

In conclusion, each tree in $\mathcal{T}_i$ can be matched to a different depot in D_i^*, $1 \leq i \leq y$. Thus, each tree in $\mathcal{T} = \cup_{i=1}^{y} \mathcal{T}_i$ can be matched to a different depot in $D^* = \cup_{i=1}^{y} D_i^* \subseteq D$. It is also clear that the maximum tree weight of trees in $\mathcal{S}$ is no more than $\frac{7}{2}B$. □

Theorem 3: Given a metric complete graph $G = (V, E)$, a depot set $D \subset V$, an integer k, and a constraint $f : D \mapsto \mathbb{Z}^+$ ($\sum_{r \in D} f(r) \geq k$), there is a $(7 + \epsilon)$-approximation algorithm for the capacitated rooted min-max cycle cover problem in G, which takes $O(n^{2.5}(\log n + \log \frac{1}{\epsilon}))$ time, where $n = |V|$ and ϵ is a constant with $0 < \epsilon < 1$.

Proof: Following Lemmas 2 and 12, the approximation ratio of the proposed algorithm can be easily derived, omitted. We now analyze its time complexity.

Let $n = |V|$. With the similar argument as we did in Theorem 1, the number of iterations for finding the optimal B_r^* is bounded by $O(\log n + \log \frac{1}{\epsilon})$. The rest is to analyze the time complexity of `Algorithm` 3 as follows.

The removal of the edges with weights greater than $B/2$ from $G - D$ takes $O(n^2)$ time, while finding the MSTs of the y connected components in the resulting graph takes $O(n^2)$ time too. Finding a tree cover of $G - D$ takes $O(n)$ time, by decomposing the y MSTs. The construction of the bipartite graph $H = (\mathcal{T}, D, E')$ requires $O(n^2)$ time. As H contains no more than $k + |D| = O(n)$ vertices, the minimum weighted maximum matching in H can be found in time $O(n^{2.5})$ by applying an algorithm in [14]. Then, each tree in $\mathcal{T}$ is connected to its matched depot in D, which takes $O(n^2)$ time. Thus, the time complexity of `Algorithm` 3 is $O(n^2) + O(n^2) + O(n) + O(n^2) + O(n^{2.5}) + O(n^2) = O(n^{2.5})$, which means that the running time of the proposed algorithm is $O(n^{2.5}(\log n + \log \frac{1}{\epsilon}))$. □

6 Performance Evaluation

In this section, we evaluate the performance of the proposed algorithms through experimental simulations. We also investigate the impact of important parameters including network size n and the number of cycles k on the algorithm performance.

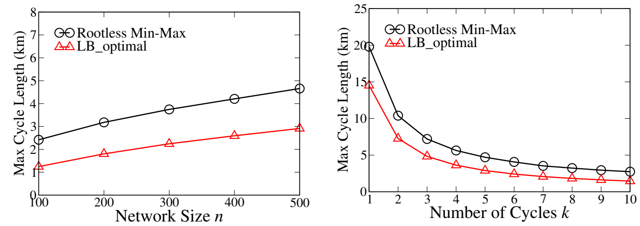
6.1 Simulation environment

We consider a network G consisting of 100 to 500 vertices randomly deployed in a $1,000m \times 1,000m$ square region. In the default setting, the number of cycles k varies from 1 to 10. For the (uncapacitated or capacitated) rooted min-max cycle cover problem, we assume that there are 10 depots randomly deployed in the region with each being contained by at most one cycle in the capacitated rooted min-max cycle cover problem. Each value in figures is the mean of the results by applying the mentioned algorithms to 50 different network topologies of the same network size.

To evaluate the performance of the proposed algorithms, we use the lower bounds of the maximum cycle length as approximations of the optimal costs. Specifically, for the rootless min-max cycle cover problem, the lower bound on the maximum cycle length is $LB\_optimal = \sum_{i=1}^{k} w(T_i)/k$, where subtrees $T_1, T_2, \cdots, T_k$ are obtained by removing the $k-1$ largest-weight edges from a minimum spanning tree in graph G. For the (uncapacitated or capacitated) rooted min-max cycle cover problem, the lower bound is $LB\_optimal = w(T')/k$, where tree T' is a minimum spanning tree in a graph G', and G' is derived from graph G by contracting all depots in D into a single vertex r_D. That is, we remove all depots in D and their adjacent edges from G first, and we then introduce a new vertex r_D and there is an edge between each vertex $v \in V - D$ and r_D with edge weight being the minimum edge weight between vertex v and any depot r in D, i.e., $w(v, r_D) = \min_{r \in D}\{w(v, r)\}$.

6.2 Performance evaluation of proposed algorithms

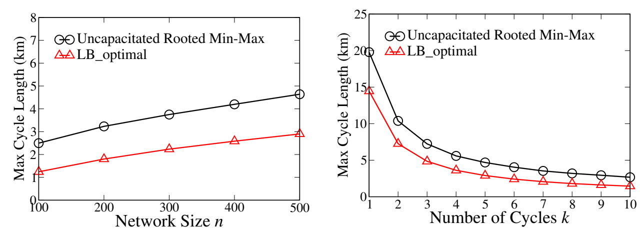
We first investigate the performance of algorithm `Rootless Min-Max Cycle Cover` for the rootless min-max cycle cover problem. Fig. 1 (a) plots the performance curves of algorithm `Rootless Min-Max Cycle Cover` and the lower bound on the maximum cycle length, by varying network size n while fixing the number of cycles k at 5, from which it can be seen that the delivered solution is fractional of the optimal. In detail, the maximum cycle length obtained by algorithm `Rootless Min-Max Cycle Cover` is around from 1.6 to 2 times of the lower bound of the optimal one, which is far less than its analytical approximation ratio $5\frac{1}{3} + \epsilon$. By varying the number of cycles k from 1 to 10 while fixing network size n at 500, Fig 1 (b) clearly shows that with the growth of the number of cycles k, the maximum cycle length in the solution delivered by algorithm `Rootless Min-Max Cycle Cover` will decrease, and its actual value is around from 1.4 to 2 times of the lower bound on the optimal one. This indicates that the estimate on its theoretical approximation ratio $5\frac{1}{3} + \epsilon$ is very conservative.



(a) The max cycle length when $k = 5$. (b) The max cycle length when $n = 500$

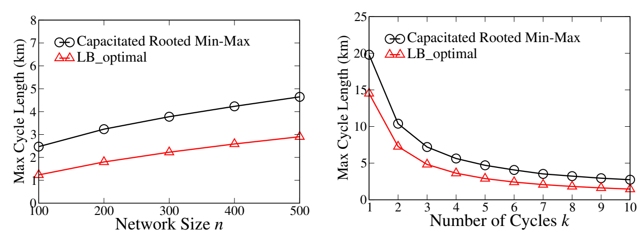
Fig. 1. The performance of algorithm `Rootless Min-Max Cycle Cover`.

We then study the performance of the proposed algorithms for uncapacitated and capacitated rooted min-max cycle cover problems in Fig. 2 and 3, respectively. It can be seen from both figures that the performance of the algorithms for both problems have the similar behaviors as that in Fig. 1. The ratios of the maximum cycle length delivered by the proposed algorithms to their lower bounds on the optimal ones are much less than their theoretical approximation ratios $6\frac{1}{3} + \epsilon$ and $7 + \epsilon$, respectively.



(a) The max cycle length when $k = 5$. (b) The max cycle length when $n = 500$.

Fig. 2. The performance of algorithm `Uncapacitated Rooted Min-Max Cycle Cover`.



(a) The max cycle length when $k = 5$. (b) The max cycle length when $n = 500$.

Fig. 3. The performance of algorithm `Capacitated Rooted Min-Max Cycle Cover`.

7 Conclusions

In this paper we have dealt with a fundamental optimization problem - the vehicle routing problem and its variants. We have devised approximation algorithms with constant approximation ratios, by exploiting the combinatorial property of the problems and employing the tree decomposition and maximum matching techniques. We finally evaluate the performance of the proposed algorithms through experimental simulations. Experimental results demonstrate that the proposed algorithms are very efficient and promising. The empirical approximation ratios are no more than 2, which are far less than their analytical counterparts $5\frac{1}{3}+\epsilon$, $6\frac{1}{3}+\epsilon$ and $7+\epsilon$, respectively.

References

[1] D. Applegate, W. Cook, S. Dash, and A. Rohe. Solution of a Min-Max Vehicle Routing Problem. *INFORM Journal on Computing*, Vol.14, pp.132-143, 2002.
[2] E. M. Arkin, R. Hassin, and A. Levin. Approximations for minimum and min-max vehicle routing problems. *Journal of Algorithms*, Vol.59, pp.1-18, 2006.
[3] I. Averbakh and O. Berman. $(p-1)/(p+1)$-approximate algorithms for p-traveling salesmen problems on a tree with minmax objective. *Discrete Applied Mathematics*, Vol.75, pp.201-216, 1997.
[4] A. M. Campbell, D. Vandenbussche, and W. Hermann. Routing for relief efforts. *Journal of Transportation Science*, Vol.42, pp.127-145, 2008.
[5] J. Carlsson, D. Ge, A. Subramaniam, and Y. Ye. Solving min-max multi-depot vehicle routing problem. *Proc. of the FIELDS Workshop on Global Optimization*, 2007.
[6] J. Desrosiers, Y. Dumas, M. M. Solomon, and F. Soumis. Time constrained routing and scheduling. *Handbooks in Operations Research and Management Science*, Elsevier, Vol.8, pp. 35-139, 1995.
[7] R. Diestel. *Graph Theory.* Springer-Verlag, 2000.
[8] M. Erol-Kantarci and H. T. Mouftah. Suresense: sustainable wireless rechargeable sensor networks for the smart grid. *IEEE Wireless Communications*, Vol.19, pp.30-36, 2012.
[9] G. Even, N. Garg, J. Könemann, R. Ravi, and A. Sinha. Covering graphs using trees and stars. *Proc. of APPROX*, LNCS, Vol.2764, pp.24-35, 2003.
[10] G. Even, N. Garg, J. Könemann, R. Ravi, and A. Sinha. Min-max tree covers of graphs. *Operations Research Letters*, Vol.32, pp.309-315, 2004.
[11] G. N. Frederickson, M. S. Hecht, and C. E. Kim. Approximation algorithms for some routing problems. *Proc. of FOCS*, IEEE, 1976.
[12] H. N. Gabow. Data structures for weighted matching and nearest common ancestors with linking. *Proc. of SODA*, ACM, 1990.
[13] S. Guo, C. Wang, and Y. Yang. Mobile data gathering with wireless energy replenishment in rechargeable sensor networks. *Proc. of INFOCOM*, IEEE, 2013.
[14] J. E. Hopcroft and R. M. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing*, Vol.2, pp.225-231, 1973.
[15] R. Jothi and B. Raghavachari. Approximating k-traveling repairman problem with repairtimes. *Journal of Discrete Algorithms*, Elsevier, Vol.5, pp.293-303, 2007.
[16] M. R. Khani and M. R. Salavatipour. Improved approximation algorithms for the min-max tree cover and bounded tree cover problems. *Proc. of APPROX*, LNCS, Vol.6845, pp.302-314, 2011.
[17] D. Kim, B.H. Abay, R.N. Uma, W. Wu, W. Wang, and A. O. Tokuta. Minimizing data collection latency in wireless sensor network with multiple mobile elements. *Proc. of INFOCOM*, IEEE, 2012.
[18] Z. Li, Y. Peng, W. Zhang, and D. Qiao. J-RoC: a joint routing and charging scheme to prolong sensor network lifetime. *Proc. of ICNP*, IEEE, 2011.
[19] W. Liang and J. Luo. Network lifetime maximization in sensor networks with multiple mobile sinks. *Proc. of LCN*, IEEE, 2011.
[20] W. Liang, J. Luo and X. Xu. Prolonging network lifetime via a controlled mobile sink in wireless sensor networks. *Proc. of Globecom*, IEEE, 2010.
[21] W. Liang, P. Schweitzer, and Z. Xu. Approximation algorithms for capacitated minimum spanning forest problems in wireless sensor networks with a mobile sink. *IEEE Trans. Computers*, Vol.62, pp.1932-1944, 2013.
[22] M. Ma, Y. Yang. Data gathering in wireless sensor networks with mobile collectors. *Proc. of IPDPS*, IEEE, 2008.
[23] H. Nagamochi and K. Okada. Approximating the minmax rooted-tree cover in a tree. *Information Processing Letters*, Vol.104, pp.173-178, 2007.
[24] Y. Shi, L. Xie, Y. T. Hou, and H. D. Sherali. On renewable Sensor Networks with wireless energy transfer. *Proc. of INFOCOM*, IEEE, 2011.
[25] V. V. Vazirani. *Approximation Algorithms*. Springer, 2001.
[26] C. Wang, J. Li, F. Ye, and Y. Yang. Multi-vehicle coordination for wireless energy replenishment in sensor networks. *Proc. of IPDPS*, IEEE, 2013.
[27] L. Xie, Y. Shi, Y. T. Hou, and H. D. Sherali. Making sensor networks immortal: an energy-renewal approach with wireless power transfer. *IEEE/ACM Trans. Netw.*, Vol.20, pp.1748-1761, 2012.
[28] L. Xie, Y. Shi, Y. T. Hou, W. Lou, H. D. Sherali, and S. F. Midkiff. On renewable sensor networks with wireless energy transfer: the multi-node case. *Proc. of SECON*, IEEE, 2012.
[29] Z. Xu, W. Liang, and Y. Xu. Network lifetime maximization in delay-tolerant sensor networks with a mobile sink. *Proc. of DCOSS*, IEEE, 2012.
[30] Z. Xu, L. Xu, and C. Li. Approximation results for min-max path cover problems in vehicle routing. *Journal of Naval Research Logistics*, Vol.57, pp.728-748, 2010.
[31] Z. Xu, D. Xu, and W. Zhu. Approximation results for a min-max location-routing problem. *Discrete Applied Mathematics*, Vol.160, pp.306-320, 2012.
[32] S. Zhang, J. Wu, and S. Lu. Collaborative mobile charging for sensor networks. *Proc. of MASS*, IEEE, 2012.
[33] M. Zhao, J. Li, and Y. Yang. Joint mobile energy replenishment and data gathering in wireless rechargeable sensor networks. *Proc. of 23rd Intl Teletraffic Congress* (ITC), IEEE, 2011.
[34] M. Zhao, M. Ma, and Y. Yang. Efficient data gathering with mobile collectors and space-division multiple access technique in wireless sensor networks. *IEEE Trans. Computers*, Vol.60, pp.400-417, 2011.

Wenzheng Xu received the ME and BSc degrees in computer science from Sun Yat-Sen University, Guangzhou, China, in 2010 and 2008, respectively. He is currently a Ph.D candidate at Sun Yat-Sen University and a visiting student at the Australian National University. His research interests include routing algorithms and protocols design for wireless ad hoc and sensor networks, approximation algorithms, combinatorial optimization, and graph theory.

Weifa Liang (M'99–SM'01) received the PhD degree from the Australian National University in 1998, the ME degree from the University of Science and Technology of China in 1989, and the BSc degree from Wuhan University, China in 1984, all in computer science. He is currently an Associate Professor in the Research School of Computer Science at the Australian National University. His research interests include design and analysis of energy-efficient routing protocols for wireless ad hoc and sensor networks, cloud computing, design and analysis of parallel and distributed algorithms, approximation algorithms, and graph theory. He is a senior member of the IEEE.

Xiaola Lin received the MSc and BSc degrees in Computer Science from Peking University, Beijing, China in 1985 and 1982 respectively, and the Ph.D degree in Computer Science from Michigan State University in the States in 1992. He is currently a full professor in the Department of Computer Science at Sun Yat-Sen University, Guangzhou, China. His research interests include parallel and distributed computing and computer networks.