Approximation Algorithms for the Min-Max Cycle Cover Problem With Neighborhoods

Lijia Deng, Wenzheng Xu, *Member, IEEE*, Weifa Liang, *Senior Member, IEEE*, Jian Peng, Yingjie Zhou, *Member, IEEE*, Lei Duan, *Member, IEEE*, and Sajal K. Das, *Fellow, IEEE*

***Abstract*— In this paper we study the min-max cycle cover problem with neighborhoods, which is to find a given number of K cycles to collaboratively visit n Points of Interest (POIs) in a 2D space such that the length of the longest cycle among the K cycles is minimized. The problem arises from many applications, including employing mobile sinks to collect sensor data in wireless sensor networks (WSNs), dispatching charging vehicles to recharge sensors in rechargeable sensor networks, scheduling Unmanned Aerial Vehicles (UAVs) to monitor disaster areas, etc. For example, consider the application of employing multiple mobile sinks to collect sensor data in WSNs. If some mobile sink has a long data collection tour while the other mobile sinks have short tours, this incurs a long data collection latency of the sensors in the long tour. Existing studies assumed that one vehicle needs to move to the location of a POI to serve it. We however assume that the vehicle is able to serve the POI as long as the vehicle is within the neighborhood area of the POI. One such an example is that a mobile sink in a WSN can receive data from a sensor if it is within the transmission range of the sensor (e.g., within 50 meters). It can be seen that the ignorance of neighborhoods will incur a longer traveling length. On the other hand, most existing studies only took into account the vehicle traveling time but ignore the POI service time. Consequently, although the length of some vehicle tour is short, the total amount of time consumed by a vehicle in the tour is prohibitively long, due to many POIs in the tour. In this paper we first study the min-max cycle cover problem with neighborhoods, by incorporating both neighborhoods and POI service time into consideration. We then propose novel approximation algorithms for the problem, by exploring the combinatorial properties of the problem. We finally evaluate the proposed algorithms via experimental simulations. Experimental results show that the proposed algorithms are promising. Especially, the maximum tour times by the proposed algorithms are only about from 80% to 90% of that by existing algorithms.**

***Index Terms*— Min-max cycle cover problem with neighborhoods, mobile data collection in WSNs, mobile charging scheduling in WSNs, multiple UAVs scheduling, approximation algorithms, traveling salesman problem with neighborhoods.**

I. Introduction

In THIS paper we study the min-max cycle cover problem with neighborhoods, which is to find a given number of K cycles to collaboratively visit n Points of Interest (POIs) in a two-dimensional Euclidean space [1], [5], [31], [34], [35], such that the length of the longest cycle among the K cycles is minimized. In other words, we need to balance the workloads of the K cycles, without incurring long cycles. The problem arises from many applications [10], [20], [26], [30]–[32], [34], [35], [37]. We here briefly introduce its two potential applications: one is to employ mobile sinks to collect sensory data in sensor networks; the other is to schedule Unmanned Aerial Vehicles (UAVs) to monitor a disaster region.

We start with the first application of the problem, that is to employ mobile sinks to collect sensory data in wireless sensor networks (WSNs). WSNs have wide applications in smart cities, data centers, precision agriculture, smart grids, and so on. A fundamental problem in WSNs is how to efficiently collect valuable sensing data from sensors. Since sensors may be sparsely deployed at some strategic locations and may not be able to communicate with each other directly. An effective solution to the data collection problem in WSNs is to dispatch multiple mobile sinks to collect sensing data from sensors [16], [39], [40]. Fig. 1(a) shows that $K = 2$ mobile sinks are employed to collect sensing data along their tours C_1 and C_2, respectively. A mobile sink will collect the data from the sensors in its data collection tour one by one. After finishing the data collection, the sink will return the base station (BS) and download the collected data to the BS. It can be seen that if the data collection tour of some mobile sink is very long, (e.g., tour C_2 in Fig. 1(a)), there will be a long time delay before the mobile sink returns to the BS, where the total amount of time spent by the mobile sink in its tour is dominated by its traveling time [31]. A long data collection tour incurs a long data collection latency. However, it is desirable for the BS to receive as much 'fresh' data as possible. For example, in a sensor network for monitoring the water quality of a reservoir, where a large area of the reservoir may be contaminated in a short period [22]. Therefore, the lengths among different data collection tours must be balanced so that sensing data can be collected by the BS as soon as possible. Fig. 1(b) demonstrates two balanced tours.

Manuscript received June 3, 2019; revised February 22, 2020 and April 27, 2020; accepted May 31, 2020; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor G. Zussman. Date of publication June 16, 2020; date of current version August 18, 2020. The work of Wenzheng Xu was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61602330 and in part by the Sichuan Science and Technology Program under Grant 2018GZDZX0010, Grant 2018GZ0094, Grant 2018GZ0093, and Grant 2017GZDZX0003. The work of Weifa Liang was supported by the Australian Research Council under the Discovery Project Scheme under Grant DP200101985. The work of Yingjie Zhou was supported by the NSFC under Grant 61801315. The work of Sajal K. Das was supported in part by the NSF under Grant CCF-1725755, Grant CNS-1818942, and Grant CNS-1545050. *(Corresponding author: Wenzheng Xu.)*

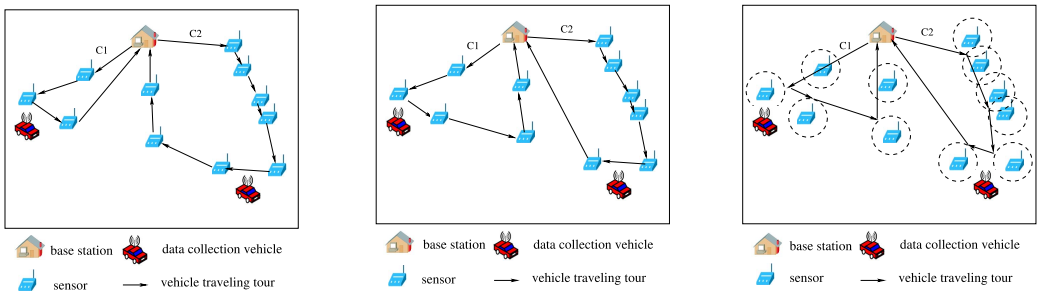
Lijia Deng, Wenzheng Xu, Jian Peng, Yingjie Zhou, and Lei Duan are with the College of Computer Science, Sichuan University, Chengdu 610065, China (e-mail: denglijia542@163.com; wenzheng.xu3@gmail.com; jianpeng@scu.edu.cn; yjzhou09@gmail.com; leiduan@scu.edu.cn).

Weifa Liang is with the Research School of Computer Science, The Australian National University, Canberra, ACT 2601, Australia (e-mail: wliang@cs.anu.edu.au).

Sajal K. Das is with the Department of Computer Science, Missouri University of Science and Technology, Rolla, MO 65409 USA (e-mail: sdas@mst.edu).

This article has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Digital Object Identifier 10.1109/TNET.2020.2999630



(a) Unbalanced tours C_1 and C_2 of two mobile sinks.

(b) More balanced tours of the two mobile sinks.

(c) The data collection tours of two vehicles when considering the service neighborhoods.

Fig. 1. The comparison between unbalanced and balanced tour plannings for data collections in WSNs.

We then focus on another application of the min-max cycle cover problem with neighborhoods in scheduling multiple Unmanned Aerial Vehicles (UAVs) to monitor disaster areas. Lightweight drones, such as a DJI phantom 4 Pro, are widely used in aerial photography, precision agriculture, disaster rescue, etc [2], [7], [17], [19], [25], [27]. For example, when a disaster (such as an earthquake or a flooding) occurs, the most critical issue is to save the people in danger. However, transportation and communication infrastructures in the disaster zone may have been destroyed already. In this scenario, UAVs can be used for disaster rescuing, by dispatching them to fly over the disaster area, taking photos or videos with the cameras on-board, and transmitting the valuable information (i.e., photos or videos) to a nearby rescue station [12], [38] for human decision making. It is desirable to collect the information of POIs in the disaster area as early as possible, since rescue operations must be performed very quickly and efficiently [19]. Therefore, multiple UAVs should be scheduled to monitor the POIs in a balanced manner.

Notice that there are also many other applications of the min-max cycle cover problem with neighborhoods, such as dispatching charging vehicles to recharge sensors in rechargeable sensor networks [20], [26], [28]–[32], the employment of autonomous robots for space exploration, undersea work, marine monitoring, tactile object recognition, facility inspection, waste managements, etc [18], [33], [35].

Although the min-max cycle cover problem without neighborhoods has attracted a lot of attentions [3], [13], [35], [36], existing studies did not consider its variants. For example, one is that they did not consider the neighborhood areas of POIs; The other is that they did not take the service time of POIs into account. However, these are critical in practical applications.

We notice that most of existing studies did not consider the neighborhood areas of POIs. They usually assumed that a vehicle (e.g., a mobile sink, a charging vehicle or a UAV) has to move to the location of a POI to serve it. However, in many applications, the vehicle is able to serve the POI, even when the vehicle stays at a location in the neighborhood area of the POI. For example, a mobile sink can collect the data of a sensor, when the sink is within its communication range (e.g., within 50 meters of the sensor [24]). The ignorance of neighborhood areas will incur longer traveling lengths of vehicles. For example, Fig. 1(b) shows the data collection tours of two mobile sinks found by existing studies, where the service neighborhoods of sensors are ignored. In contrast, a mobile sink is able to receive the data of a sensor when the sink is within its transmission range. Fig. 1(c) demonstrates their data collection tours when considering the neighborhoods of sensors, where a dotted circle represents the communication area of a sensor [5], [6]. It is clear that the traveling length of each mobile sink in Fig. 1(c) is much shorter than that in Fig. 1(b). Shorter data collection tours can reduce not only the data collection latency but also the energy consumption of vehicles on traveling.

On the other hand, most existing studies ignored the service time of POIs. The total time spent by a vehicle in its tour consists of its traveling time along the tour and service time on the POIs in the tour. Although existing studies can find tours for different UAVs with more or less equal traveling lengths, some tour may contain many POIs than others to be served, and such the service time on POIs sometimes is comparative to the vehicle traveling time. For example, in the application for dispatching charging vehicles to recharge sensors, it takes substantial amount of time for a vehicle to recharge a sensor (from 30 minutes to 2 hours) [26], [30]. Therefore, the total amount of time consumed of one tour by existing studies may be much longer than that of other tours.

In this paper we study a novel min-max cycle cover problem with neighborhoods, by incorporating both service neighborhoods and POI service time into consideration. Specifically, given a positive integer K, the problem is to find K tours to serve POIs, such that the longest tour time among the K tours is minimized, where a tour time is composed of the vehicle traveling time and the service time of POIs in the tour, and the vehicle can serve a POI when it is at a location in the neighborhood of the POI.

We consider three different variants of the min-max cycle cover problem with neighborhoods: (1) A rootless variant, i.e., each tour does not contain a depot; (2) A single-rooted variant, i.e., every tour contains a single depot; and (3) a multi-rooted variant, i.e., different tours must contain different depots. We will also propose novel approximation algorithms for each of the three variants.

The novelties of the proposed algorithms for the max-min cycle cover problem with neighborhoods are as follows. For the rootless variant, we upper bound the total time of a tour that visits the neighborhoods in several 'nearby' optimal tours, see Lemma 2 in Section 3.3, where the word 'nearby' means that the nearest traveling time between two optimal tours is no greater than a threshold. For the single-rooted variant, we show the relationship between the longest time of serving a single node and the value of the optimal solution,

see Lemma 5 in Section 4.2. Finally, for the multi-rooted variant, we design a novel tree merging technique so that different rootless tours can be matched to different roots, see Section 5.2 and Lemma 6 in Section 5.3.

The main contributions of this paper can be summarized as follows. Different from most existing studies that neither considered the service neighborhoods nor the service time of POIs, in this paper we study the min-max cycle cover problem with neighborhoods by incorporating these two into consideration. To the best of our knowledge, we propose the first approximation algorithms for both rootless and single-rooted min-max cycle cover problems with neighborhoods, delivering solutions with the longest tour times being $(27+\epsilon)\cdot OPT_{rootless}+c_1\cdot r$ and $7.75\cdot OPT_s+c_2\cdot r$, respectively, where ϵ is a given constant with $0<\epsilon\leq 1$, $OPT_{rootless}$ and OPT_s are the optimal solutions to the rootless and single-rooted min-max cycle cover problems, c_1 and c_2 are positive constants, $r=\frac{R}{\eta}$, R is the neighborhood radius of each POI, and η is the vehicle travel speed. We also devise an improved approximation algorithm for the multi-rooted min-max cycle cover problem with neighborhoods, delivering an approximate solution with the longest tour time $(28+\epsilon)\cdot OPT_m+c_3\cdot r$ while the best result for the problem so far is $(369+\epsilon)\cdot OPT_m+c_3'\cdot r$ [16], where OPT_m is the optimal solution to the problem, c_3 and c_3' are two positive constants with $c_3\leq c_3'$, ϵ is a given constant with $0<\epsilon\leq 1$. We finally evaluate the proposed algorithms via experimental simulations. Experimental results show that the proposed algorithms are very promising. Especially, the longest tour times delivered by the proposed algorithms are only around 80% to 90% of those by existing algorithms.

The rest of the paper is organized as follows. Section II introduces preliminaries. Section III, IV, and V propose approximation algorithms for the three variants of the min-max cycle cover problems with neighborhoods, respectively. Section VI evaluates the proposed algorithms through experimental simulations. Section VII reviews related work, and Section VIII concludes the paper.

II. Preliminaries

A. System Model

Let V be a set of n Points of Interest (POIs) in a two-dimensional Euclidean space, i.e., $V=\{v_1,v_2,\ldots,v_n\}$. Denote by (x_i,y_i) the coordinate of POI v_i with $1\leq i\leq n$. The *neighborhood area* of a POI v_i is represented by a disk $D(v_i)$ that centers at v_i with a given radius R, where R is a given non-negative number. That is, $D(v_i)$ is the set of points (including POI v) with the coordinate (x,y) such that $(x-x_i)^2+(y-y_i)^2\leq R^2$, i.e., $D(v_i)=\{v\mid (x-x_i)^2+(y-y_i)^2\leq R^2$, and v is a point with the coordinate $(x,y)\}$. Notice that any two disks $D(v_i)$ and $D(v_j)$ may overlap with each other.

To serve the POIs in V, we assume that there are K vehicles located at K depots $s_1,s_2,\ldots,s_K$, where vehicle k is at depot s_k with $1\leq k\leq K$. Notice that in some applications, multiple vehicles may be located at a single depot. In this case, the depot can be treated as multiple 'virtual' identical depots and assume that there is only one vehicle located at a 'virtual' depot [31].

We assume that a vehicle can serve a POI v_i if it stays at any location in the neighborhood area $D(v_i)$ of v_i, and it takes $h(v_i)$ time to serve v_i with $h(v_i)\geq 0$. On the other hand, since the vehicle has to travel to the neighborhood areas of POIs, we also assume that it takes $l(p_i,p_j)$ time from locations p_i to p_j, where locations p_i and p_j are in disks $D(v_i)$ and $D(v_j)$ of POIs v_i and v_j, respectively. To serve the POIs in V by the K vehicles, we need to partition set V into K disjoint subsets $V_1,V_2,\ldots,V_K$, and each vehicle serves the POIs in V_k, where $1\leq k\leq K$. Let $n_k=|V_k|$, then $\sum_{k=1}^{K}n_k=|V|=n$.

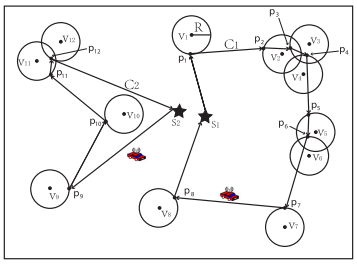


Fig. 2. An example of the network model, where there are 12 POIs $v_1,v_2,\ldots,v_{12}$ in a 2D space, and the neighborhood of each POI v_i is a disk $D(v_i)$ that centers at v_i with a given radius R.

The *service tour* C_k of a vehicle for serving POIs in V_k is defined as follows. Assume that the vehicle will serve POIs $v_1,v_2,\ldots,v_{n_k}$ in V_k one by one. The vehicle starts from depot s_k, and moves to a location p_1 in the neighborhood area $D(v_1)$ to serve POI v_1, then travels from p_1 to a location p_2 in $D(v_2)$ to serve v_2, and so on. It finally returns depot s_k after serving v_{n_k} at p_{n_k}. Therefore, the service tour is $C_k=s_k\rightarrow p_1\rightarrow p_2\rightarrow\cdots\rightarrow p_{n_k}\rightarrow s_k$, where location p_i is in the neighborhood $D(v_i)$ of POI v_i with $1\leq i\leq n_k$. For example, Fig. 2 shows the service tours of $K=2$ vehicles.

Denote by η the traveling speed of each vehicle. Then, the traveling time of a vehicle between any two locations p_i and p_j is $l(p_i,p_j)=\frac{d(p_i,p_j)}{\eta}$, where $d(p_i,p_j)$ is the Euclidean distance between p_i and p_j. Let $r=\frac{R}{\eta}$, where R is the radius of each disk.

It can be seen that the total time $w(C_k)$ by a vehicle in tour C_k consists of its service time for POIs in V_k and its traveling time along tour C_k, which are $\sum_{i=1}^{n_k}h(v_i)$ and $\sum_{i=0}^{n_k}l(p_i,p_{i+1})$, respectively, i.e., $w(C_k)=\sum_{i=1}^{n_k}h(v_i)+\sum_{i=0}^{n_k}l(p_i,p_{i+1})$, where $h(v_i)$ is the time spent by the vehicle for serving v_i and $l(p_i,p_{i+1})$ is the traveling time taken by the vehicle from locations p_i to p_{i+1} and $p_0=p_{n_k+1}=s_k$.

B. Problem Definitions

In a large scale network, there are many POIs to be served. If only one vehicle is employed, the service waiting time at some POIs will be prohibitively long. Therefore, it is desirable to employ multiple vehicles to serve the POIs, such that the service time can be significantly shortened.

Given K vehicles, a very important problem is how to assign n POIs to the K vehicles in a balanced way, such that the longest tour time among the K vehicle tours is minimized. Otherwise, some vehicles still have many POIs to be served, resulting in long waiting times for some POIs, while the other vehicles have only a few POIs to be served. The problem is termed as the *min-max cycle cover problem with neighborhoods*.

We consider three different variants of the problem. (1) A rootless variant, i.e., each tour does not contain a depot; (2) A single-rooted variant, i.e., every tour contains the single depot; and (3) a multi-rooted variant, i.e., different tours must contain different depots.

The rationale behind the three variants is that, an algorithm for the rootless variant will be served as a subroutine for the algorithm for the multi-rooted variant. On the other hand, when the K vehicles are co-located at the same depot (i.e., the single-rooted variant), we can explore special combinatorial properties of this variant to devise an improved approximation algorithm than that for the multi-rooted variant.

We start by defining the problem without depots. Given n POIs $v_1, v_2, \ldots, v_n$ in V and K vehicles in a 2D space, and the travel speed η of each vehicle, denote by $D(v_i)$ the neighborhood of each v_i in V with a given radius R. *The rootless min-max cycle cover problem with neighborhoods* is to find K closed tours $C_1, C_2, \ldots, C_K$, such that the maximum total consumed time among the K tours, i.e., $\max_{i=1}^{K}\{w(C_i)\}$, is minimized, subject to the constraint that one of the K vehicles must visit at least one location in the neighborhood $D(v_i)$ of each v_i in V, i.e., $D(v_i) \cap (\cup_{k=1}^{K} V(C_k)) \neq \emptyset$ for each v_i in V. That is, our optimization objective is to

$$\min\{\max_{i=1}^{K}\{w(C_i)\}\}, \tag{1}$$

$$\text{subject to } D(v_i) \cap (\cup_{k=1}^{K} V(C_k)) \neq \emptyset, \quad \forall v_i \in V \tag{2}$$

The single-rooted min-max cycle cover problem with neighborhoods can be defined similarly, and the only difference from the rootless case of the problem is that all the K tours must contain the same depot s.

We finally extend the single depot case to the multiple depot case, by considering the problem with K vehicles located at K different depots $s_1, s_2, \ldots, s_K$. We term this variant as *the multi-rooted min-max cycle cover problem with neighborhoods*.

III. Approximation Algorithm for the Rootless Min-Max Cycle Cover Problem With Neighborhoods

In this section, we propose a novel approximation algorithm for the rootless min-max cycle cover problem with neighborhoods.

A. Algorithm Framework

Given n POIs $v_1, v_2, \ldots, v_n$ in a 2D space, a radius R, and K vehicles, we assume that tours $C_1^*, C_2^*, \ldots, C_K^*$ form an optimal solution to the rootless min-max cycle cover problem with neighborhoods. Also, let OPT be the optimal value of the optimal solution, i.e., $OPT = \max_{k=1}^{K}\{w(C_k^*)\}$.

The basic idea of the proposed algorithm is that, given an upper bound B on the optimal value OPT (i.e., $B \geq OPT$), the algorithm will find a solution with no more than K tours such that the maximum tour time among the K tours is no more than $27B + 108r$, where $r = \frac{R}{\eta}$, R is the radius of each disk and η is the travel speed of each vehicle. Then, the algorithm can find a lower bound OPT' on OPT through binary search, in the end the algorithm can find K tours with the maximum tour time $27 \cdot OPT' + 108r$.

Specifically, we start by estimating an upper bound OPT_{ub} on OPT. We find an approximate tour C that contains each POI in V. Let $OPT_{ub} = h(V) + l(C)$, where $h(V)$ is the time spent by the vehicle for serving all POIs in V and $l(C)$ is the traveling time taken by a vehicle for visiting POIs in V along C. It can be seen that $OPT \leq OPT_{ub}$.

Given a guess B of OPT (B may be larger or less than OPT), assume that there is an algorithm for finding tours such that the consumed time of each tour is no larger than $27B + 108r$ (which will be shown later), the algorithm for the rootless min-max cycle cover problem with neighborhoods is presented in `Algorithm 1`.

Algorithm 1 Approximation Algorithm for the Rootless Min-Max Cycle Cover Problem With Neighborhoods (**approAlgNoRoots**)

Input: n POIs with their coordinates, the service time $h(v_i)$ of each POI v_i, the radius R of each disk, K vehicles, vehicle travel speed η, and an error threshold ϵ with $0 < \epsilon \leq 1$

Output: K tours $C_1, C_2, \ldots, C_K$ such that their maximum tour time is $27 \cdot OPT + 108r$

```
Let lb ← 0 and ub ← OPT_ub = h(V) + l(C);
Let δ ← ϵ/27;
while lb(1 + δ) < ub do
    Let B ← (lb+ub)/2; /* a guess of OPT. */
    Invoke Algorithm 2 to find, say K', tours C_1, C_2, ..., C_K' with the maximum weight 27B + 108r;
    if K' ≤ K then
        ub ← B; /* the guess B of OPT is too large, reduce the upper bound ub to B. */
    else
        lb ← B; /* the guess B of OPT is too small, increase the lower bound lb to B. */
    end if
end while
Let OPT' = ub; /* find a lower bound OPT' on OPT*/
Invoke Algorithm 2 to find K tours C_1, C_2, ..., C_K with their maximum weight 27 · OPT' + 108r;
```

B. Algorithm

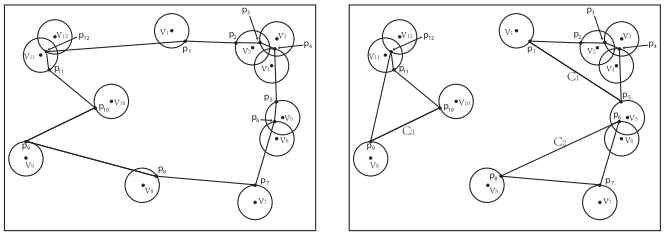
Given a guess B of OPT, we here find a set of tours $C_1, C_2, \ldots, C_{K'}$ such that the maximum tour time among the K' tours is no more than $27B + 108r$. We later show that the number of tours K' is no more than K if $B \geq OPT$.

The algorithm consists of three steps. Step one: partition the set V of POIs into disjoint subsets; Step two: obtain an approximate shortest tour for visiting the neighborhoods of POIs in each subset; and Step three: split long tours into short tours with the resulting tour time being no larger than $27B + 108r$.

1) Step One: Partition POI Set V Into Disjoint Subsets: Given any two disks $D(v_i)$ and $D(v_j)$, denote by $l(D(v_i), D(v_j))$ the minimum traveling time between points in disks $D(v_i)$ and $D(v_j)$. That is, $l(D(v_i), D(v_j)) = \frac{d(v_i,v_j)-2R}{\eta}$ if disks $D(v_i)$ and $D(v_j)$ do not overlap with each other, where $d(v_i, v_j)$ is the Euclidean distance between POIs v_i and v_j, R is the radius of each disk and η is the travel speed of the vehicle. Otherwise ($D(v_i)$ and $D(v_j)$ overlap with each other), $l(D(v_i), D(v_j)) = 0$.

An auxiliary graph $G = (V, E)$ is first constructed, where $V = \{v_1, v_2, \ldots, v_n\}$, and there is an edge (v_i, v_j) in E if the minimum traveling time between disks $D(v_i)$ and $D(v_j)$ is no more than $\frac{B}{2}$, i.e., $l(D(v_i), D(v_j)) \leq \frac{B}{2}$, where $1 \leq i, j \leq n$.

Assume that there are T connected components $CC_1, CC_2, \ldots, CC_T$ in G, where $T \geq 1$. Denote by $V_1, V_2, \ldots, V_T$ the sets of POIs in these T connected components, respectively. Then, $\sum_{t=1}^{T} |V_t| = |V|$. It can be



(a) An approximate shortest tour C'_t for the TSPN problem, where tour $C'_t = p_1 \rightarrow p_2 \rightarrow \cdots \rightarrow p_{12} \rightarrow p_1$, p_i is a location in disk $D(v_i)$, and $1 \leq i \leq 12$

(b) tour C'_t is split into 3 subtours C_1, C_2 and C_3

Fig. 3. An illustration of the proposed algorithm for the rootless min-max cycle cover problem with neighborhoods.

seen that, for any two disks $D(v_i)$ and $D(v_j)$ that centers at POIs v_i and v_j, respectively, the minimum traveling time between them is strictly longer than $\frac{B}{2}$ if POIs v_i and v_j are in two different connected components of G.

2) Step Two: Obtain an Approximate Shortest Tour C'_t for Visiting the Neighborhoods of POIs in Each Subset V_t: For each connected component CC_t, let $V_t = \{v_1, v_2, \ldots, v_{n_t}\}$ be the set of POIs in CC_t, where $n_t = |V_t|$ and $1 \leq t \leq T$. Recall that (x_i, y_i) is the coordinate of a POI v_i in V_t and $D(v_i)$ is its neighborhood area. The algorithm then finds an approximate shortest tour C'_t that visits the disks centered at POIs in V_t by applying an algorithm from [5], such that $w(C'_t) \leq 6.75 \cdot w(C^*_t) + 20.4r$, where C^*_t is a shortest tour. Let $C'_t = p_1 \rightarrow p_2 \rightarrow \cdots \rightarrow p_{n_t} \rightarrow p_1$, where p_i is a location in the neighborhood $D(v_i)$. For example, Fig. 3(a) shows a tour C'_t for visiting the neighborhoods of 12 POIs. As a result, the algorithm obtains T tours $C'_1, C'_2, \ldots, C'_T$ for the T connected components.

3) Step Three: Split Long Tours Into Short Tours: Having the T tours $C'_1, C'_2, \ldots, C'_T$, if none of them has a tour time $w(C'_t)$ larger than $27B+108r$, it is done. Otherwise, we further partition those long tours into short tours that the weight of each short tour is no greater than $27B + 108r$ as follows.

For each tour $C'_t = p_1 \rightarrow p_2 \rightarrow \cdots \rightarrow p_{n_t} \rightarrow p_1$, we define the weight of each edge (p_i, p_{i+1}) in C'_t as $w'(p_i, p_{i+1}) = \frac{h(v_i)+h(v_{i+1})}{2} + l(p_i, p_{i+1})$. It can be seen that

$$w'(C'_t) = \sum_{1 \leq i \leq n_t} \left(\frac{h(v_i) + h(v_{i+1})}{2} + l(p_i, p_{i+1})\right) = \sum_{1 \leq i \leq n_t} h(v_i) + \sum_{1 \leq i \leq n_t} l(p_i, p_{i+1}) = w(C'_t). \quad (3)$$

Let path $P = p_1 \rightarrow p_2 \rightarrow \ldots \rightarrow p_{n_t}$ derived from C'_t directly. Then, $w'(P) \leq w'(C'_t)$. Assume that we have split k paths $P_1, P_2, \ldots, P_k$ from P and the first l_k locations $p_1, p_2, \ldots, p_{l_k}$ in P are contained in the k paths. Initially, $k = 0$ and $l_k = 0$. The $(k+1)$th path P_{k+1} is $p_{l_k+1} \rightarrow p_{l_k+2} \rightarrow \cdots \rightarrow p_{l_{k+1}}$, where the weighted sum $w'(P_{k+1})$ of the edges in P_{k+1} is no more than $13.5B + 54r$, while the weighted sum $w'(P_{k+1} \rightarrow p_{l_{k+1}+1})$ of the edges in path $P_{k+1} \rightarrow p_{l_{k+1}+1}$ is strictly larger than $13.5B + 54r$. The splitting procedure continues until all location nodes in P are split. Assume that K' paths are obtained from splitting the T tours $C'_1, C'_2, \ldots, C'_T$, where $K' \geq T$.

For each obtained path P_k, a tour C_k is constructed from P_k, by connecting the two end-points of P_k, where $1 \leq k \leq K'$. It can be seen that the weight $w(C_k)$ of tour C_k is no more than $2w'(P_k) \leq 27B + 108r$. Fig. 3(b) shows that three tours C_1, C_2, and C_3 are split from the tour C'_t in Fig. 3(a). Notice that although disk $D(v_5)$ is visited by both tours C_1 and C_2 in Fig. 3(b), POI v_5 will be served by the vehicle in C_1, but not by the vehicle in C_2.

Given any guess B of OPT, the detailed algorithm for finding, say K', tours in G with each tour time being no larger than $27B + 108r$ is given in `Algorithm 2`.

Algorithm 2 Algorithm for Finding Tours With Each Tour Time Being No Larger Than $27B + 108r$

Input: n POIs $v_1, v_2, \ldots, v_n$ and a guess B of OPT

Output: K' tours $C_1, C_2, \ldots, C_{K'}$ with each tour time being no larger than $27B + 108r$

1: Construct an auxiliary graph $G = (V, E)$, where there is an edge (v_i, v_j) in E if the minimum traveling time between disks $D(v_i)$ and $D(v_j)$ is no more than $\frac{B}{2}$;
2: Assume that there are T connected components $CC_1, CC_2, \ldots, CC_T$ in G. Denote by $V_1, V_2, \ldots, V_T$ the node sets of the T connected components, respectively;
3: Let $\mathcal{P} = \emptyset$; /* the set of split paths. */
4: **for** $t \leftarrow 1$ to T **do**
5: Find an approximate shortest tour $C'_t = p_1 \rightarrow p_2 \rightarrow \cdots \rightarrow p_{n_t} \rightarrow q_1$ that visits the neighborhood of each POI in V_t, by applying an algorithm from [5];
6: Split tour C'_t into short subpaths so that the weighted sum of the edges in each subpath is no greater than $13.5B + 54r$, and add these subpaths to $\mathcal{P}$;
7: **end for**
8: For each P_k in $\mathcal{P}$, obtain a closed tour C_k from P_k by connecting the two end points of P_k; Let $\mathcal{C} = \{C_1, C_2, \ldots C_{K'}\}$, where $K' = |\mathcal{P}|$.

C. Algorithm Analysis

The key to analyze the approximation ratio of the proposed algorithm is to show that `Algorithm 2` can find no more than K tours with the maximum weight among the tours no greater than $27B + 108r$ if $B \geq OPT$.

Recall that $C^*_1, C^*_2, \ldots, C^*_K$ form an optimal solution to the problem, and $OPT = \max_{i=1}^{K}\{w(C^*_i)\}$. To this end, we first show that the POIs served by each optimal tour C^*_i must be contained in a single connected component CC_t at `Step 2` of `Algorithm 2` if $B \geq OPT$, where $1 \leq t \leq T$. Then, for each connected component CC_t, assume that there are k^*_t optimal tours contained in CC_t, where $k^*_t \geq 1$ with $1 \leq t \leq T$. It is clear that, $\sum_{t=1}^{T} k^*_t = K$. We finally prove that the number k_t of delivered tours from CC_t in `Algorithm 2` is no more than k^*_t, i.e., $k_t \leq k^*_t$. Then, the number of delivered tours by `Algorithm 2` is no more than K, i.e., $\sum_{t=1}^{T} k_t \leq \sum_{t=1}^{T} k^*_t = K$ if $B \geq OPT$.

We first show that the POIs served by each optimal tour C^*_i must be contained in a single connected component CC_t.

Lemma 1: The POIs served by each C^*_i among the optimal K tours must be contained in a single connected component CC_t, where $1 \leq t \leq T$.

Proof: The proof in contained in Section 1 of the supplementary materials file. □

For each CC_t, assume that there are k_t^* of the optimal K tours contained in CC_t. It can be seen that the k_t^* optimal tours are close to each other, which means that the minimum traveling time between two of the k^* optimal tours is no greater than $\frac{B}{2}$. Then, $\sum_{t=1}^{T} k_t^* = K$. Also, denote by C_t^* the optimal tour for visiting disks that are served in the k_t^* optimal tours. We bound the total consumed time in C_t^* by the following lemma.

Lemma 2: The total consumed time $w(C_t^*)$ of C_t^* is no more than $(2k_t^* - 1)B + 8(k_t^* - 1)r$ if $B \geq OPT$, where $r = \frac{R}{\eta}$, R is the radius of each disk and η is the traveling speed of each vehicle.

Proof: Recall that there are k_t^* optimal tours in CC_t. For the sake of convenience, assume that the k_t^* optimal tours are $C_1^*, C_2^*, \ldots, C_{k_t^*}^*$. For each C_i^*, it can be seen that the minimum traveling time between any two disks visited in C_i^* is no more than $\frac{B}{2}$, by Lemma 1. On the other hand, following the construction of connected component CC_t, there is at least one disk $D(u)$ visited by C_i^*, and another disk $D(v)$ visited by another tour C_j^* with $j \neq i$ and $1 \leq i, j \leq k_t^*$, such that the minimum traveling time $l(D(u), D(v))$ between them is no greater than $\frac{B}{2}$. For example, Fig. 4(a) shows that there are $k_t^* = 3$ optimal tours in a connected component CC_t and Fig. 4(b) demonstrates that the minimum traveling time $l(D(u_1), D(v_1))$ between disk $D(u_1)$ in C_1^* and disk $D(v_1)$ in C_2^* is no more than $\frac{B}{2}$. Similarly, $l(D(u_2), D(v_2)) \leq \frac{B}{2}$. Also, assume that line segment uv intersects the two circles centered at POIs u and v at two points a_u and b_v, respectively. For example, Fig. 4(b) shows that line segment u_1v_1 intersects the circles centered at POIs u_1 and v_1 at points a_1 and b_1, respectively. It can be seen that traveling time $l(a_u, b_v)$ between points a_u and b_v is equal to $l(D(u), D(v))$, i.e., $l(a_u, b_v) = l(D(u), D(v))$, if disks $D(u)$ and $D(v)$ do not overlap with each other; otherwise (disks $D(u)$ and $D(v)$ overlap), $l(D(u), D(v)) = 0 \leq l(a_u, a_v)$.

Assume that the optimal tour C_i^* serves POI u at location p_u in $D(u)$ while C_j^* serves POI v at location p_v in $D(v)$. We distinguish it into two cases: (i) Disks $D(u)$ and $D(v)$ overlap with each other, and (ii) $D(u)$ and $D(v)$ do not overlap. We show that the traveling time $l(p_u, p_v)$ between locations p_u and p_v is no more than $\frac{B}{2} + 4r$ as follows.

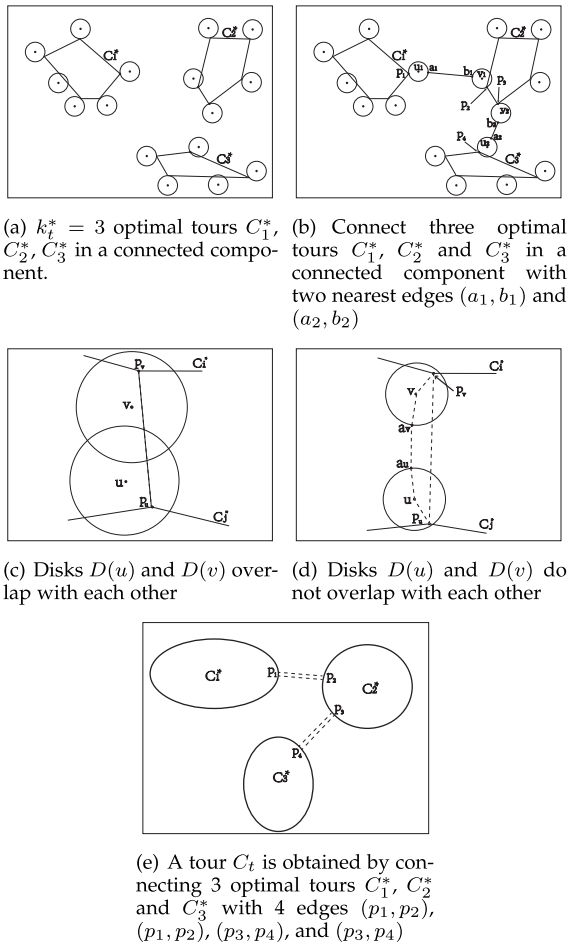
Case (i) where disks $D(u)$ and $D(v)$ overlap with each other. It can be seen that the traveling time $l(p_u, p_v)$ between locations p_u and p_v is no more than $4r$, see Fig. 4(c), i.e.,

$$l(p_u, p_v) \leq 4r \leq 4r + \frac{B}{2}, \quad \text{as } B \geq 0. \tag{4}$$

Case (ii) where $D(u)$ and $D(v)$ do not overlap with each other, see Fig. 4(d). The traveling time $l(p_u, p_v)$ between p_u and p_v is upper bounded by

$$\begin{aligned} &l(p_u, p_v) \\ &\quad \leq l(p_u, u) + l(u, a_u) + l(a_u, a_v) + l(a_v, v) + l(v, p_v), \\ &\qquad \text{by the triangle inequality, e.g., see Fig. 4(d),} \\ &\quad \leq r + l(u, a_u) + l(a_u, a_v) + l(a_v, v) + r, \\ &\qquad \text{as } p_u \text{ is in } D(u) \text{ and } p_v \text{ is in } D(v), \\ &\quad = r + r + l(a_u, a_v) + r + r, \\ &\quad \leq \frac{B}{2} + 4r, \text{ as } l(a_u, a_v) = l(D(u), D(v)) \leq \frac{B}{2} \end{aligned} \tag{5}$$

By combining Eq.(4) and Eq.(5), we have $l(p_u, p_v) \leq 4r + \frac{B}{2}$.



(a) $k_t^* = 3$ optimal tours C_1^*, C_2^*, C_3^* in a connected component.

(b) Connect three optimal tours C_1^*, C_2^* and C_3^* in a connected component with two nearest edges (a_1, b_1) and (a_2, b_2)

(c) Disks $D(u)$ and $D(v)$ overlap with each other

(d) Disks $D(u)$ and $D(v)$ do not overlap with each other

(e) A tour C_t is obtained by connecting 3 optimal tours C_1^*, C_2^* and C_3^* with 4 edges (p_1, p_2), (p_1, p_2), (p_3, p_4), and (p_3, p_4)

Fig. 4. An illustration of Lemma 2.

A single tour C_t that visits the disks centered at POIs in CC_t can be obtained, by adding $2(k_t^* - 1)$ edges with the weight of each edge is no more than $\frac{B}{2} + 4r$, as the degree of each location in C_t is a positive even number. For example, Fig. 4(e) illustrates a tour C_t by connecting $k_t^* = 3$ optimal tours with $2(3-1) = 4$ edges.

The weight of tour C_t is no more than

$$\begin{aligned} w(C_t) &\leq \sum_{i=1}^{k_t^*} w(C_i^*) + 2(k_t^* - 1)(\frac{B}{2} + 4r) \\ &\leq k_t^* \cdot OPT + 2(k_t^* - 1)(\frac{B}{2} + 4r), \\ &= (2k_t^* - 1)B + 8(k_t^* - 1)r, \text{ as } OPT \leq B. \end{aligned} \tag{6}$$

Recall that C_t^* is an optimal tour for visiting the disks centered at POIs in CC_t. We have that $w(C_t^*) \leq w(C_t) \leq (2k_t^* - 1)B + 8(k_t^* - 1)r$. The lemma then follows. □

Lemma 3: Algorithm 2 can find no more than K tours with the maximum weight among the K tours being $27B + 108r$ if $B \geq OPT$.

Proof: Recall that there are $k_1^*, k_2^*, \ldots, k_T^*$ optimal tours in the T connected components, respectively. We show that the number k_t of delivered tours from each CC_t by Algorithm 2 is no more than k_t^*. Then, $\sum_{t=1}^{T} k_t \leq \sum_{t=1}^{T} k_t^* = K$. For each CC_t, Algorithm 2 can find an approximate tour C_t' for visiting the disks centered at POIs in CC_t, such that $w(C_t') \leq 6.75 \cdot w(C_t^*) + 20.4r$, by following a

recent work [5], where C_t^* is the optimal tour. Then, the weight of path P obtained by `Algorithm 2` is

$$\begin{aligned} w'(P) &\le w'(C_t') = w(C_t'), \\ &\le 6.75 \cdot w(C_t^*) + 20.4r \\ &\le 6.75((2k_t^* - 1)B + (8k_t^* - 8)r) + 20.4r \\ &\le (13.5B + 54r)k_t^*, \text{ as } B \ge 0 \text{ and } r \ge 0. \end{aligned} \quad (7)$$

It can be seen that the number k_t of split paths from path P by `Algorithm 2` is no more than $\lceil \frac{w'(P)}{13.5B+54r} \rceil \le \lceil \frac{(13.5B+54r)k_t^*}{13.5B+54r} \rceil \le k_t^*$. Therefore, the number of tours delivered by `Algorithm 2` is no more than $\sum_{t=1}^{T} k_t \le \sum_{t=1}^{T} k_t^* = K$, as each tour is constructed from a split path by connecting its two end-points.

Finally, the weight of each tour C_i by `Algorithm 2` is no more than $2(13.5B+54r) = 27B+108r$. The lemma then follows. □

Theorem 1: Given n POIs with their coordinates, the service time $h(v_i)$ of each POI v_i, the radius R of each disk, K vehicles and the vehicle travel speed η, there is an approximation algorithm, i.e., `Algorithm 1`, for the rootless min-max cycle cover problem with neighborhoods, which finds no more than K tours with the maximum tour weight among the tours being no larger than $(27 + \epsilon) \cdot OPT + 108r$, where ϵ is a given constant with $0 \le \epsilon \le 1$, OPT is the value of an optimal solution and $r = \frac{R}{\eta}$.

Proof: Following Lemma 3, `Algorithm 2` can find no more than K tours with the maximum tour weight no more than $27\ B + 108r$ if $B \ge OPT$. Recall that `Algorithm 1` finds an estimate OPT' on the optimal value OPT by a binary search. When the **while** loop in `Algorithm 1` terminates, we have $ub \le (1+\delta)lb$, which means that we find at least $K+1$ tours with the guess $B\ (= lb)$ on the optimal solution, while we find no more than K tours when $B = ub$. Following Lemma 3, we know that $lb < OPT$; otherwise ($lb \ge OPT$), we can find no more than K tours with the guess $B = lb$. Following `Step 12` of `Algorithm 1`, we know that $OPT' = ub \le (1+\delta)lb < (1+\delta)OPT$. Therefore, the maximum tour weight among the found tours at `Step 13` of `Algorithm 1` is no more than $27 \cdot OPT' + 108r \le 27(1+\delta)OPT + 108r = (27 + \epsilon) \cdot OPT + 108r$, where $\delta = \frac{\epsilon}{27}$. The theorem then follows. □

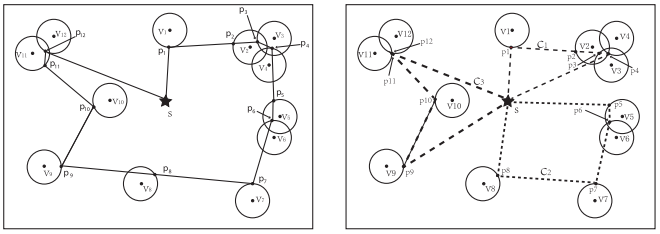
IV. Approximation Algorithm for the Single-Rooted Min-Max Cycle Cover Problem With Neighborhoods

In this section, we devise a novel approximation algorithm for the single-rooted min-max cycle cover problem with neighborhoods in a 2D space.

A. Algorithm

The basic idea of the proposed algorithm is that it first finds a single approximate tour C' for the Traveling Salesman Problem with Neighborhoods (TSPN), which is to find a shortest tour, such that each disk centered at a node with a given radius is visited by the tour. The algorithm then partitions tour C into K balanced, disjoint subtours $C_1, C_2, \ldots, C_K$.

Unlike the rootless min-max cycle cover problem with neighborhoods, the single-rooted problem considered in this section assumes that each found tour must contain a single depot s.



(a) An approximate tour C' for the TSPN problem, where tour $C' = s \to p_1 \to p_2 \to \cdots \to p_{12} \to s$, p_i is a location in disk $D(v_i)$, and $1 \le i \le 12$

(b) Split tour C' into $K = 3$ $s-$rooted subtours C_1, C_2, C_3

Fig. 5. An illustration of the proposed algorithm for the single-rooted min-max cycle cover problem with neighborhoods.

1) Step One: Obtain an Approximate Tour C': We first obtain an approximate tour $C' = s \to p_1 \to p_2 \to \cdots \to p_n \to s$ that visits depot s and the neighborhood of each disk $D(v_i)$, by applying an algorithm from [5], see Fig. 5(a), where p_i is a location in disk $D(v_i)$.

2) Step Two: Split the Long Tour Into Short Tours: We split tour C' into K subtours $C_1, C_2, \ldots, C_K$ with each rooted at depot s. Notice that the split procedure here is totally different from the one in the previous section (see `Algorithm 2` in `Section III-B.3`). Recall that the total time $w(C')$ spent by a vehicle in tour C' consists of its total service time for the n POIs $v_1, v_2, \ldots, v_n$ and its traveling time in tour C', which are $\sum_{i=1}^{n} h(v_i)$ and $\sum_{i=0}^{n} l(p_i, p_{i+1})$, respectively. Then $w(C') = \sum_{i=1}^{n} h(v_i) + \sum_{i=0}^{n} l(p_i, p_{i+1})$.

We define the weight of edge (p_i, p_{i+1}) as $w'(p_i, p_{i+1}) = \frac{h(v_i)+h(v_{i+1})}{2} + l(p_i, p_{i+1})$. It can be seen that $w'(C') = w(C')$ by Eq.(3).

On the other hand, denote by c_{max} the longest time for serving only one POI in tour C, i.e, $c_{max} = \max_{p_i \in P}\{h(v_i) + 2l(p_i, s)\}$, where $P = \{p_1, p_2, \ldots, p_n\}$, and POI v_i is served at location p_i.

We split tour C' into K single-rooted tour $C_1, C_2, \ldots, C_K$ by the edge weight $w'(p_i, p_{i+1})$ and c_{max} as follows. We first find $K-1$ special locations $p_{l_1}, p_{l_2}, \ldots, p_{l_{K-1}}$ in tour C', where p_{l_i} is the last location along tour C such that the total weight of the path from s to p_{l_i} along C' is no more than $w_i = \frac{i}{K}(w'(C') - c_{max}) + \frac{c_{max}}{2}$, but the total weight of the path from s to p_{l_i+1} is strictly larger than w_i, where $1 \le i \le K-1$.

We construct subtours $C_1, C_2, \ldots, C_K$ with the $K-1$ locations $p_{l_1}, p_{l_2}, \ldots, p_{l_{K-1}}$, where C_1 is obtained by connecting the last location p_{l_1} of path $s \to p_1 \to p_2 \to \cdots \to p_{l_1}$ to depot s, i.e., $C_1 = s \to p_1 \to p_2 \to \cdots \to p_{l_1} \to s$; C_i $(2 \le i \le K-1)$ is derived from adding path $p_{l_{i-1}+1} \to \cdots \to p_{l_i}$ with two edges $(s, p_{l_{i-1}+1})$ and (p_{l_i}, s), i.e., $C_i = s \to p_{l_{i-1}+1} \to \cdots \to p_{l_i} \to s$; and the last subtour C_K is obtained by adding path $p_{l_{K-1}+1} \to \cdots \to p_n \to s$ with one edge $(s, p_{l_{K-1}+1})$, i.e., $C_K = s \to p_{l_{K-1}+1} \to \cdots \to p_n \to s$. For example, Fig. 5(b) shows that tour C' is spit into $K = 3$ subtours C_1, C_2, and C_3.

The algorithm for the single-rooted min-max cycle cover problem with neighborhoods is referred to as Algorithm `approAlgOneRoot`.

B. Algorithm Analysis

In the following, we first prove that the consumed time of each obtained subtour C_i $(1 \le i \le K)$ is no more than

$\frac{1}{K}(w(C')-c_{max})+c_{max}$. We then bound the weight of c_{max}. We finally analyze the approximation ratio of the proposed algorithm.

Lemma 4: Algorithm `approAlgOneRoot` can find a solution with no more than K subtours and the consumed time of each subtour C_i does not exceed $\frac{1}{K}(w(C')-c_{max})+c_{max}$, where $c_{max} = \max_{p_i \in P}\{h(v_i) + 2 \cdot l(p_i, s)\}$, $h(v_i)$ is the service time of POI v_i, p_i is the location where v_i is served by a vehicle in the neighborhood of v_i, $l(p_i, s)$ is the traveling time between p_i and the root s, and $1 \leq i \leq K$.

Proof: The proof is contained in Section 2 of the supplementary materials file. □

We bound the weight of c_{max} by the following lemma.

Lemma 5: Assume that tours $C_1^*, C_2^*, \ldots, C_K^*$ form an optimal solution to the single-rooted min-max cycle cover problem with neighborhoods. Let OPT be the optimal value of the problem, i.e., $OPT = \max_{i=1}^{K}\{w(C_i^*)\}$. Then, we have $c_{max} \leq OPT+4r$, where $c_{max} = \max_{p_i \in P}\{h(v_i)+2l(p_i, s)\}$ with $P = \{p_1, p_2, \ldots, p_n\}$, p_i is in disk $D(v_i)$, $r = \frac{R}{\eta}$, R is the radius of each disk and η is the travel speed of each vehicle.

Proof: Recall that c_{max} is the longest time for serving any POI v_i in V, where v_i is served at a location p_i in disk $D(v_i)$. Let p_i be such a location with $h(v_i)+2l(p_i, s) = c_{max}$. Denote by p_i^* the service location for POI v_i in disk $D(v_i)$ in the optimal solution. Then, the traveling time between p_i and p_i^* is no larger than $2r$, as both p_i and p_i^* are in disk $D(v_i)$. We then have

$$\begin{aligned} c_{max} &= h(v_i) + 2l(s, p_i), \\ &\leq h(v_i) + 2(l(s, p_i^*) + l(p_i^*, p_i)), \\ &\leq h(v_i) + 2(l(s, p_i^*) + 2r), \quad \text{as } l(p_i^*, p_i) \leq 2r, \\ &\leq OPT + 4r, \text{ as } OPT \geq h(v_i) + 2l(s, p_i^*). \end{aligned} \tag{8}$$

Therefore, we have $c_{max} \leq OPT + 4r$. □

Theorem 2: Given n POIs $v_1, v_2, \ldots, v_n$ in a 2D space, a radius R, K vehicles located at a depot s initially, and the travel speed η, there is an approximation algorithm for the single-rooted min-max cycle cover problem with neighborhoods, and the maximum tour time in its solution is no more than $7.75 \cdot OPT + 20.4r$, where OPT is the optimal value for the problem and $r = \frac{R}{\eta}$.

Proof: Recall that, we have $w(C') \leq 6.75w(C^*) + 20.4r$ by the work in [5], where $w(C')$ and $w(C^*)$ are the total weight of the delivered tour C' and the optimal value of tour C^* for the problem, respectively.

Also, recall that OPT is the optimal value for the single-rooted min-max cycle cover problem with neighborhoods, and $C_1^*, C_2^*, \ldots, C_K^*$ form an optimal solution to the problem. Then, $OPT = \max_{i=1}^{K}\{w(C_i^*)\}$.

Since each tour C_i^* is rooted at depot s, we can construct a single tour C_N that covers all neighborhoods $D(v_1), D(v_2), \ldots, D(v_n)$ and s from the K optimal tours $C_1^*, C_2^*, \ldots, C_K^*$, and the total consumed time in tour C_N is no more than $w(C_N) \leq \sum_{i=1}^{K} w(C_i^*) \leq K \cdot OPT$. On the other hand, since C^* is the optimal solution to the TSPN problem, we know that

$$w(C^*) \leq w(C_N) \leq K \cdot OPT. \tag{9}$$

In the following, we bound the total consumed time in each split subtour by Algorithm `approAlgOneRoot`.

$$\begin{aligned} &\max_{i=1}^{K}\{w(C_i)\} \\ &\leq \frac{1}{K}w(C') + (1 - \frac{1}{K})c_{max}, \text{ by Lemma 4,} \\ &\leq \frac{1}{K}(6.75w(C^*) + 20.4r) + (1 - \frac{1}{K})c_{max} \\ &\leq \frac{1}{K}(6.75w(C^*) + 20.4r) + (1 - \frac{1}{K})(OPT + 4r), \\ &\qquad \text{as } c_{max} \leq OPT + 4r \text{ by Lemma 5,} \\ &= \frac{6.75}{K}w(C^*) + \frac{20.4r}{K} + (1 - \frac{1}{K})OPT + 4r - \frac{4r}{K}, \\ &\leq \frac{6.75}{K} \cdot K \cdot OPT + \frac{16.4r}{K} + (1 - \frac{1}{K})OPT + 4r, \text{ by Eq.(9)} \\ &\leq 7.75 \cdot OPT + 20.4r, \quad \text{as } K \geq 1. \end{aligned} \tag{10}$$

The theorem then follows. □

V. Approximation Algorithm for the Multi-Rooted Min-Max Cycle Cover Problem With Neighborhoods

In this section, we deal with the multi-rooted min-max cycle cover problem with neighborhoods, which is to find K tours $C_1, C_2, \ldots, C_K$ to collaboratively visit the neighborhoods $D(v_1), D(v_2), \ldots, D(v_n)$ of n POIs $v_1, v_2, \ldots, v_n$ in a 2D space, such that the longest consumed time among the K tours, i.e., $\max_{i=1}^{K}\{w(C_i)\}$, is minimized, subject to that tour C_i must contain depot s_i, where the consumed time of tour C_i consists of the time for serving the POIs and the traveling time spent by a vehicle in the tour and $1 \leq i \leq K$.

A. The Basic Idea

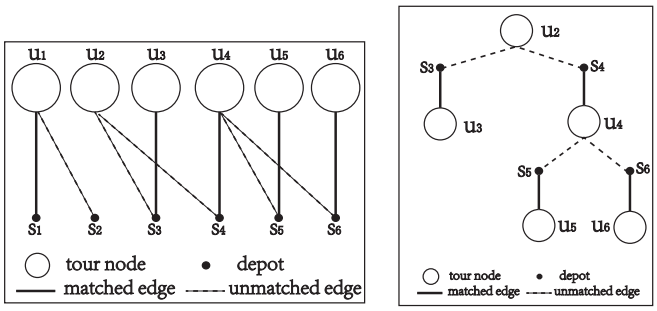
Assume that tours $C_1^*, C_2^*, \ldots, C_K^*$ form an optimal solution to the problem, where tour C_i^* contains depot s_i with $1 \leq i \leq K$. Denote by OPT_m the optimal value of the problem, i.e., $OPT_m = \max_{i=1}^{K}\{w(C_i^*)\}$.

The basic idea of the proposed algorithm is that, given an upper bound B on OPT_m, it first finds K' ($\leq K$) rootless tours that only visit disks $D(v_1), D(v_2), \ldots, D(v_n)$ of the n POIs, which means that each tour does not necessarily contain any depot. It then checks whether there is a matching between the tours and the depots. If yes, it obtains K' rooted tours, by connecting each tour to its matched depot; otherwise, it merges some rootless tours in a novel way, such that less than K' resulting tours are obtained. This merge procedure continues until each tour is matched to a depot.

B. Algorithm

Recall that OPT is the optimal value of the optimal solution for the *rootless* min-max cycle cover problem with neighborhoods. It can be seen that the optimal value OPT_m for the multi-rooted version of the problem is no less than OPT, i.e., $OPT_m \geq OPT$, since any feasible solution $\mathcal{C}' = \{C_1', C_2', \ldots, C_K'\}$ can be obtained for the rootless problem, where C_i' is constructed by shortcutting the depot s_i in C_i^*.

Given an upper bound B on OPT_m, in the following we show that there are no more than K rooted tours covering all POIs such that the maximum tour weight among the tours is no more than $28B + 112r$.



(a) An example of the constructed bipartite graph G_b with $K' = 6$ tours and $K = 6$ depots

(b) A maximal u_2-rooted tree T_A in G_b, where the path from u_2 to each leaf is an alternating path

Fig. 6. An illustration of the proposed algorithm for the multi-rooted min-max cycle cover problem with neighborhoods.

We first find K' ($\leq K$) rootless tours $C_1, C_2, \ldots, C_{K'}$ with the maximum tour weight being no more than $27B + 108r$, by invoking `Algorithm 2`, since $B \geq OPT_m \geq OPT$. Denote by $\mathcal{C}$ the set of these K' tours, i.e., $\mathcal{C} = \{C_1, C_2, \ldots, C_{K'}\}$.

For each rootless tour C_i, let $C_i = p_1 \rightarrow p_2 \rightarrow \cdots \rightarrow p_{n_i} \rightarrow p_1$, where p_k in C_i is the service location for some POI, and $1 \leq k \leq n_i$. Denote by $l(C_i, s_j)$ the *minimum traveling time between tour C_i and depot s_j*, i.e., $l(C_i, s_j) = \min_{k=1}^{n_i}\{\frac{d(p_k, s_j)}{\eta}\}$, where $d(p_k, s_j)$ is the Euclidean distance between location p_k and depot s_j, and η is the traveling speed of a vehicle.

We then construct a bipartite graph $G_b = (U \cup S, E_b)$ from the K' tours and the K depots, see Fig. 6(a), where each node $u_i \in U$ represents a tour C_i with $1 \leq i \leq K'$ (u_i is referred to as a *tour node*), and S is the set of depots, i.e., $S = \{s_1, s_2, \ldots, s_K\}$. There is an edge (u_i, s_j) in E_b between a tour node u_i and a depot s_j if the minimum traveling time $l(C_i, s_j)$ between them is no larger than $\frac{B}{2}+2r$, i.e., $l(C_i, s_j) \leq \frac{B}{2} + 2r$.

A maximum matching M in graph G_b can be found. We distinguish our discussion into two cases. Case (i): each tour node u_i (representing tour C_i) is matched to a depot s_j in M; and Case (ii): there is at least one node u_i that has not been matched with any depot.

For Case (i) where each node u_i (or tour C_i) is matched to a depot s_j, we can obtain K' ($\leq K$) rooted tours $C_1^r, C_2^r, \ldots, C_{K'}^r$ as follows. Recall that $C_i = p_1 \rightarrow p_2 \rightarrow \cdots \rightarrow p_{n_i} \rightarrow p_1$. For the sake of convenience, assume that p_1 is the nearest location to depot s_j among the n_i locations in C_i, i.e., $p_1 = \arg\min_{k=1}^{n_i}\{d(p_k, s_j)\}$. Then, $l(p_1, s_j) \leq \frac{B}{2} + 2r$, since tour C_i is matched to s_j. We obtain a rooted tour C_i^r from C_i, by replacing the last edge (p_{n_i}, p_1) in C_i with two edges (p_{n_i}, s_j) and (s_j, p_1), i.e., $C_i^r = p_1 \rightarrow p_2 \rightarrow \cdots \rightarrow p_{n_i} \rightarrow s_j \rightarrow p_1$. We later show that the weight of tour C_i^r is no more than $28B + 112r$, i.e., $w(C_i^r) \leq 28B + 112r$.

For Case (ii) where there is at least one tour node u_i in G_b that does not match to any depot. We merge some of the K' rootless tours to obtain less than K' resulting tours. The detailed merge procedure is as follows.

We first find a maximal u_i-rooted tree T_A in G_b, so that each path starting at node u_i and ending at any leaf node in T_A is an *augmenting path*, where an augmenting path is such a path that consists of unmatched edges and matched edges alternatively [11]. It can be seen that tree T_A can be found, by applying Depth-First Search (DFS) starting from node u_i. Denote by $T_A = (U_A \cup S_A, E_A)$ the found tree, where U_A and S_A are the sets of tour nodes and depots in T_A, respectively, and E_A is the set of edges in T_A. For example, Fig. 6(b) shows a found tree T_A.

Tree T_A has the following two important properties: (i) The number of tour nodes is one more than the number of depots in tree T_A (i.e., $|U_A| = |S_A| + 1$) and $|S_A| \geq 1$. For example, the numbers of tour nodes and depots in Fig. 6(b) are 5 and 4, respectively. (ii) Assume that the tour nodes in U_A represent n'_U tours $C_1, C_2, \ldots, C_{n'_U}$, where $n'_U = |U_A|$. For each disk $D(v)$ visited by one of the n'_U tours, assume that $D(v)$ is visited by an optimal tour C_k^* that contains depot s_k, where $D(v)$ is the disk that centers at POI v. Then, depot s_k must be contained in set S_A if $B \geq OPT_m$.

The aforementioned two properties imply that the disks visited by the n'_U tours $C_1, C_2, \ldots, C_{n'_U}$ are visited by only $n'_S = |S_A|$ ($= n'_U - 1$ optimal rooted tours for the problem. We then can find no more than n'_S rootless tours $C'_1, C'_2, \ldots, C'_{n'_S}$ with each having a weight no more than $27B + 108r$, such that the visited disks by the n'_U tours $C_1, C_2, \ldots, C_{n'_U}$ are also visited by the n'_S tours, by invoking `Algorithm 2`, due to $B \geq OPT_m \geq OPT$. We then replace the n'_U tours in $\mathcal{C}$ with the newly n'_S tours, i.e., let $\mathcal{C}' = (\mathcal{C} \setminus \{C_1, C_2, \ldots, C_{n'_U}\}) \cup \{C'_1, C'_2, \ldots, C'_{n'_S}\}$. It can be seen that the number of tours in $\mathcal{C}'$ is one less than the number in $\mathcal{C}$, i.e., $|\mathcal{C}'| = |\mathcal{C}| - 1$.

Similar to the construction of bipartite graph G_b, we then construct another bipartite graph $G' = (U' \cup S, E')$ from the tours in $\mathcal{C}'$ and the K depots and see whether each tour in $\mathcal{C}'$ can be matched to a depot in S. If yes, done. Otherwise, the tour merging procedure continues until each tour is matched to a depot.

The algorithm for the multi-rooted min-max cycle cover problem with neighborhoods is referred to as Algorithm `approAlgMultiRoots`.

C. Algorithm Analysis

The key to the analysis of the approximation ratio of the proposed algorithm is to show that the solution delivered by the algorithm contains no more than K rooted tours, and the maximum tour weight among the tours is no more than $28B + 112r$ if $B \geq OPT_m$.

We first show important properties of tree T_A.

Lemma 6: Assume that a tour node u_i is not matched to any depot in the maximum matching M of graph $G_b = (U \cup S, E_b)$. Construct a maximal u_i-rooted tree $T_A = (U_A \cup S_A, E_A)$ in G_b, so that each path from node u_i to any leaf node in tree T_A is an augmenting path, where $U_A \subseteq U$, $S_A \subseteq S$, and $E_A \subseteq E_b$. We have that

(i) each leaf node in T_A is matched in M;
(ii) each leaf node in T_A is a tour node, not a depot node;
(iii) there is at least one depot in set S_A, i.e., $|S_A| \geq 1$.
(iv) The number of tour nodes is one more than the number of depots in tree T_A, i.e., $|U_A| = |S_A| + 1$.
(v) Assume that the tour nodes in U_A represent n'_U tours $C_1, C_2, \ldots, C_{n'_U}$, where $n'_U = |U_A|$. For each disk $D(v)$ visited by one of the n'_U tours, assume that $D(v)$ is visited by an optimal tour C_k^* that contains depot s_k, where $D(v)$ is the disk that centers at POI v. Then, depot s_k must be contained in set S_A if $B \geq OPT_m$.

Proof: It can be seen that the first three claims holds, see Fig. 6, and their proofs can be found in Section 3 of the supplementary materials file. In the following we show the rest claims.

We now show claim (iv) that the number of tour nodes is one more than the number of depots in tree T_A, i.e., $|U_A| = |S_A|+1$. In fact, we show a stronger claim. That is, for any u_i-rooted subtree $T'_A = (U'_A \cup S'_A, E'_A)$ of tree T_A, such that each leaf in subtree T'_A is a tour node, we have $|U'_A| = |S'_A| + 1$. We show the claim by an induction on the number of tour nodes in U'_A as follows.

It is obvious that the claim is true when subtree T'_A consists of only a single node u_i, since u_i is tour node and $|U'_A| = 1$ and $|S'_A| = 0$.

We assume that the claim holds when there are n'_U tour nodes in a subtree of T_A. We consider the case where there are n'_U+1 tour nodes in a subtree $T'_A = (U'_A \cup S'_A, E'_A)$ of tree T_A, where each leaf in T'_A is a tour node and $|U'_A| = n'_U + 1$. There must be a leaf node u in T'_A, such that it is matched in M, since each tour node except u_i in T_A is matched and there are at least two tour nodes in T'_A. It can be seen that node u is matched to its parent node s_u in T'_A, where s_u is a depot. Let u' be the parent of s_u in tree T'_A, where u' is a tour node. We construct a graph $T'' = (U''_A \cup S''_A, E''_A)$ from tree T'_A, by removing both nodes u and s_u, and edges (u, s_u) and (s_u, u') from T'_A. It can be seen that graph T''_A is a tree, since node u is a leaf and s_u has only one child in tree T'_A, due to the fact that s_u can be matched to only one node, i.e., node u. Following the assumption that claim (iv) holds for a subtree T''_A of T'_A, we know that $|U''_A| = |S''_A| + 1$. Since $U'_A = U''_A \cup \{u\}$ and $S'_A = S''_A \cup \{s_u\}$, we have $|U'_A| = |U''_A| + 1 = |S''_A| + 1 + 1 = |S'_A|+1$. Claim (iv) then follows, since T_A is a subtree of itself.

We finally prove claim (v) that depot s_k must be in S_A by contradiction. Suppose that s_k is not in S_A. Assume that disk $D(v)$ is visited by a tour C of the n'_U tours, and a node u in tree $T_A = (U_A \cup S_A, E_A)$ represents tour C. Since disk $D(v)$ is visited by an optimal tour C^*_k that contains depot s_k, the minimum traveling time $l(D(v), s_k)$ between disk $D(v)$ and depot s_k must be no more than $\frac{B}{2}$, as $B \geq OPT_m \geq w(C^*_k)$. Then, the minimum traveling time $l(C, s_k)$ between tour C and depot s_k is no larger than $\frac{B}{2} + 2r$ between node u (representing tour C) and depot s_k in graph G_b. We distinguish our discussion into three cases. Case (1): depot s_k is not matched to any node in M; Case (2): depot s_k is matched to a tour node u' in G_b, but u' is not in tree T_A, i.e., $u' \in U \setminus U_A$; and Case (3): depot s_k is matched to a tour node u' in G_b, and u' is in tree T_A, i.e., $u' \in U_A$. In the following, we show that none of the three cases is possible.

We start with Case (1) by contradiction. Suppose that case (1) is possible. Consider the alternating path $P_{u_i,u}$ from u_i to u in tree T_A. It can be seen that node u is matched to its parent node in T_A, and the parent is the last second node in path $P_{u_i,u}$. Then, path $P_{u_i,u} \rightarrow s_k$ in G is also an augmenting path and both the first node u_i and the last node s_k are not matched. A matching M' can be constructed from path $P_{u_i,u} \rightarrow s_k$ and M such that the number of matched edges in M' is one more than the number in M (i.e., $|M'| = |M| + 1$), by following Lemma 1 in [11]. This however contradicts that M is the maximum matching. Thus, Case (1) is impossible.

We then prove Case (2) that depot s_k is matched to a tour node u' in $U \setminus U_A$ is impossible. A tree T'_A from T_A can be obtained, by adding both edges (u, s_k) and (s_k, u'). Consider the augmenting path $P_{u_i,u}$ from u_i to u in tree T_A. It can be seen that path $P_{u_i,u} \rightarrow s_k \rightarrow u'$ is also an augmenting path. T_A is a proper subtree of T'_A, which contradicts that T_A is the maximal u_i-rooted tree in G such that the path from u_i to each leaf is an augmenting path. Therefore, Case (2) is impossible.

We finally show Case (3) that depot s_k is matched to a tour node u' in U_A is impossible. Following the construction of tree T_A. It can be seen that each tour node u_j except the tree root u_i is matched to a depot $s_{k'}$ in tree T_A. Then, tour node u' is matched to both depots $s_{k'}$ and s_k in M, where $s_{k'}$ is in tree T_A while s_k is not in T_A. This however contradicts the definition of a matching, in which tour node u' can be matched to no more than one depot. Therefore, Case (3) is also impossible.

By combining the above discussions, the assumption that s_k is not in S_A is false. Claim (v) then follows. □

Lemma 7: Algorithm `approAlgMultiRoots` can deliver no more than K rooted tours such that the maximum tour weight is $28B + 112r$, if $B \geq OPT_m$.

Proof: Since $B \geq OPT_m \geq OPT$, the proposed algorithm can find $K'(\leq K)$ rootless tours by invoking `Algorithm` 2, and let $\mathcal{C} = \{C_1, C_2, \ldots, C_{K'}\}$ be the set of the K' tours, where OPT is the optimal value for the rootless version of the problem. If each tour in $\mathcal{C}$ is matched to a depot we then have a set of no more than K rooted tours in $\mathcal{C}^r$. Otherwise (at least one tour C_i in $\mathcal{C}$ is not matched), following Lemma 6, the disks visited by the n'_U tours $C_1, C_2, \ldots, C_{n'_U}$ are visited by only $n'_S = |S_A|$ $(= n'_U - 1)$ optimal rooted tours for the problem. Then, the proposed algorithm will find no more than $|S_A|$ rootless tours, and the updated rootless tours in $\mathcal{C}$ obtained is no more than $K' - 1$. If each resulting rootless tour in $\mathcal{C}$ is matched to a depot, done. Otherwise, we continue to obtain another set of updated rootless tours and the number of tours is decreased by one. Notice that the algorithm must stop, as the number of tours in $\mathcal{C}$ decreases by one at least after each time. Therefore, the proposed algorithm obtains no more than K rooted tours if $B \geq OPT_m$.

It can be seen that the weight of each rooted tour C^r_i is no greater than $27B + 108r + 2 \cdot (\frac{B}{2} + 2r) \leq 28B + 112r$. □

Theorem 3: Given n POIs with their coordinates, the service time $h(v_i)$ of each POI v_i, the radius R of each disk, K vehicles located at K depots $s_1, s_2, \ldots, s_K$, respectively, and the travel speed η of each vehicle, there is an approximation algorithm for the multi-rooted min-max cycle cover problem with neighborhoods, which finds an approximate solution with no more than K rooted tours with the maximum tour weight among the tours no larger than $(28+\epsilon) \cdot OPT_m + 112r$, where ϵ is given constant with $0 < \epsilon \leq 1$, OPT_m is the value of the optimal solution and $r = \frac{R}{\eta}$.

Proof: Its proof is similar to the one in Theorem 1, omitted. □

VI. PERFORMANCE EVALUATION

A. Simulation Environment

We consider the application of the min-max cycle cover problem with neighborhoods in scheduling multiple mobile sinks to collect sensing data in WSNs. We assume that the network consists of from 100 to 500 sensors that are deployed in a $1\ km \times 1\ km$ square area [30]. We also assume that the number of mobile sinks K varies from 1 to 5 [31]. To collect sensing data from a sensor by a mobile sink, the mobile sink needs to move to a location within the transmission range

R of the sensor, e.g., $R = 50\ m$ [24]. The traveling speed of each mobile sink is $\eta = 1\ m/s$. Also, the service time $h(v)$ for collecting data from a sensor v is randomly drawn from an interval $[0,\ 30]$ seconds. For the single-rooted min-max cycle cover problem with neighborhoods, the depot is located at the center of the monitoring area. On the other hand, for the multi-rooted min-max cycle cover problem with neighborhoods, the K depots are randomly deployed in the network.

To compare the performance of the proposed algorithms `approAlgNoRoots`, `approAlgOneRoot`, and `approAlgMultiRoots` for rootless, single-rooted, and multi-rooted min-max cycle cover problems with neighborhoods, we consider existing algorithms as follows.

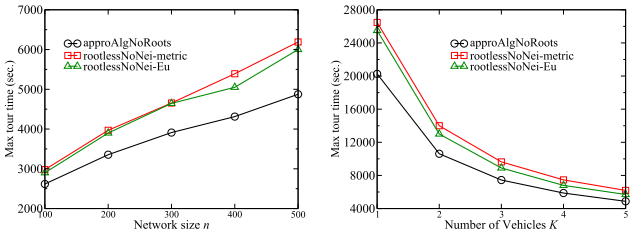
First, for the rootless min-max cycle cover problem with neighborhoods, we compare our algorithm against algorithms `rootlessNoNei-metric` and `rootlessNoNei-Eu` in [35], which deliver $5 + \epsilon$ and $4 + \epsilon$ approximate solutions to the rootless min-max cycle cover problems without neighborhoods in a generic metric space and a Euclidean space, respectively, where ϵ is a given constant with $0 < \epsilon \leq 1$.

Second, for the single-rooted min-max cycle cover problem with neighborhoods, we compare our algorithm against two existing algorithms: Algorithm `singleNoNei` that does not consider neighborhoods in [9]; and Algorithm `singleNei` in [16], which considers service neighborhoods and the longest tour time is $(369+\epsilon)\cdot OPT_s + c\cdot r$, where OPT_s is the optimal value of the problem and c is constant.

Finally, for the multi-rooted min-max cycle cover problem with neighborhoods, we compare our algorithm against the following three existing algorithms. The first two are Algorithm `multiNoNei-metric` in [31] and Algorithm `multiNoNei-Eu` in [35], which deliver $7 + \epsilon$ and $5 + \epsilon$ approximate solutions to the problem without neighborhoods in a metric space and a Euclidean space, respectively. The other is Algorithm `multiNei` in [16], which takes service neighborhoods into account, and the longest tour time is $369 \cdot OPT_m + c \cdot r$, where OPT_m is the optimal value of the problem.

B. Algorithm Performance

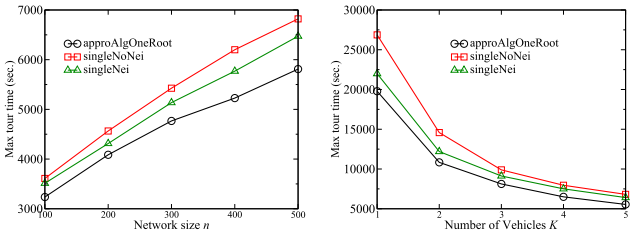
We first evaluate the algorithm performance for the rootless min-max cycle cover problem with neighborhoods. Fig. 7(a) shows the performance of different algorithms by varying the network size n from 100 to 500, while keeping the number of vehicles K at 5, from which it can be seen that the maximum tour time by each algorithm increases with the growth of the network size n, as more POIs need to be visited. Fig. 7(a) also demonstrates that the maximum tour time by the proposed algorithm `approAlgNoRoots` is only from 80% to 90% of that by algorithm `rootlessNoNei-Eu`. For example, the maximum tour times by algorithms `approAlgNoRoots`, `rootlessNoNei-metric` and `rootlessNoNei-Eu` are about 4,900, 6,200, and 6,000, seconds, respectively, when the network size $n = 500$. On the other hand, Fig. 7(b) plots the performance of the three comparison algorithms by varying the number of vehicles K from 1 to 5, when $n = 500$, from which it can be seen that the maximum tour time by each of the three algorithms decreases very quickly when more vehicles are deployed. Fig. 7(b) also shows that the maximum tour time by the proposed algorithm `approAlgNoRoots` is about 80% of those by the other two algorithms.



(a) Maximum tour times by different algorithms by varying the network size n from 100 to 500, when $K = 5$

(b) Maximum tour times by different algorithms by varying the number of vehicles K from 1 to 5, when $n = 500$

Fig. 7. Performance of different algorithms for the rootless min-max cycle cover problem with neighborhoods.



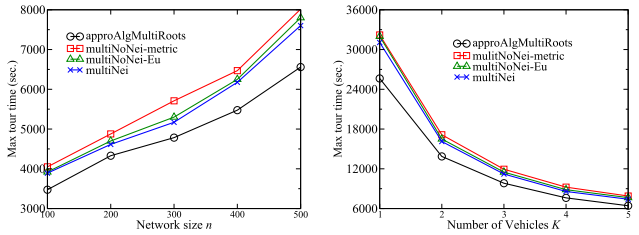
(a) Maximum tour times by different algorithms by varying the network size n from 100 to 500, when $K = 5$

(b) Maximum tour times by different algorithms by varying the number of vehicles K from 1 to 5, when $n = 500$

Fig. 8. Performance of different algorithms for the single-rooted min-max cycle cover problem with neighborhoods.

We then study the performance of different algorithms for the single-rooted min-max cycle cover problem with neighborhoods. Fig. 8(a) shows that the maximum tour time by the proposed algorithm `approAlgOneRoot` is much shorter than those by algorithms `singleNoNei` and `singleNei`. For example, the maximum tour times by algorithms `approAlgOneRoot`, `singleNoNei` and `singleNei` are 5,800, 6,800, and 6,500 seconds, respectively when $n = 500$. The rationale behind the algorithm performance is that, algorithm `singleNoNei` does not take the service neighborhoods of POIs into account. On the other hand, although algorithm `singleNei` considers such neighborhoods, it assumes that the K vehicles are initially located at different depots [16]. In contrast, the proposed algorithm `approAlgOneRoot` explores the combinatorial properties of the optimization problem when the K vehicles are co-located at a single depot. On the other hand, Fig. 8(b) plots that the maximum tour time in the solution delivered by algorithm `approAlgOneRoot`, which is smaller than that by the other two algorithms when the number of vehicles K increases from 1 to 5.

We finally investigate the performance of different algorithms for the multi-rooted min-max cycle cover problem with neighborhoods. Fig. 9(a) demonstrates that the maximum tour time in the solution delivered by Algorithm `approAlgMultiRoots` is about 85%, 88%, and 89% of those by algorithms `multiNoNei-metric`, `multiNoNei-Eu`, and `multiNei`, respectively, where Algorithm `multiNei` takes POI neighborhoods into consideration, while algorithms `multiNoNei-metric` and `multiNoNei-Eu` do not. Fig. 9(b) also shows that the maximum tour time by Algorithm `approAlgMultiRoots` is no more than 84% of those by the other two algorithms.



(a) Maximum tour times by different algorithms by varying the network size n from 100 to 500, when $K = 5$

(b) Maximum tour times by different algorithms by varying the number of vehicles K from 1 to 5, when $n = 500$

Fig. 9. Performance of different algorithms for the multi-rooted min-max cycle cover problem with neighborhoods.

VII. Related Work

The min-max cycle cover problem has attracted a lot of attentions in the past, due to its wide applications. We notice that most existing studies did not consider the neighborhoods of POIs [5], [6], [8], [9], [14]–[17], [19], [21], [23], [31], [34], [35]. For example, for the rootless min-max cycle cover problem, Xu *et al.* [31] assumed that one vehicle needs to move to the location of a sensor to recharge it, and they studied the problem of dispatching K charging vehicles to recharge energy-critical sensors such that the length of the longest charging tour is minimized. They proposed a $5\frac{1}{3}$-approximation algorithm. Yu and Liu [35] later improved the approximation ratio to 5 for the problem in general metric graphs, and to $(4 + \epsilon)$ in the Euclidean space, where ϵ is a given constant with $0 < \epsilon \leq 1$. For the single-rooted min-max cycle cover problem, Frederickson *et al.* [9] devised a 2.5-approximation algorithm. For the multi-rooted min-max cycle cover problem, Xu *et al.* [31] designed a 7-approximation algorithm, and Yu and Liu [34] studied the relationship between rootless and multi-rooted min-max cover problems. Since a tree can be transformed to a closed tour in a metric graph such that the weight of the closed tour is no more than twice the tree weight, the min-max tree cover problem has also been well studied, which is to find K trees to cover all nodes such that the weight of the heaviest tree is minimized. Even *et al.* [8] proposed a 4-approximation algorithm. Khani and Salavatipour, *et al.* [14] later improved the approximation ratio to 3.

There are extensive studies on the Traveling Salesman Problem with Neighborhoods (TSPN) in a 2D space, which is to find a shortest tour, such that each disk centered at a node with a given radius is visited by the tour. The studies of the TSPN problem can be further categorized by different constraints, such as whether any two disks are disjoint with each other; and whether the radii of different disks are identical. For the TSPN under the constraints that all disks have identical radii and their coverage are not overlapping with each other, for which Dumitrescu and Mitchell [4] proposed a Polynomial Time Approximation Scheme (PTAS). For the TSPN under the constraints that any two disks are disjoint but with different radii, Mitchell [23] devised a constant approximation algorithm for it. For the TSPN under the constraints that two disks are allowed to overlap with each other and the radii of different disks are equal, Dumitrescu and Mitchell [4] proposed a 7.62-approximation algorithm and later they further improved the ratio to 6.75 [5]. And finally for the TSPN under the constraints that two disks may overlap with each other but their radii are different, Dumitrescu *et al.* [6] recently developed a constant approximation algorithm for it. Furthermore, Ma *et al.* [21] investigated the use of a single charging vehicle to charge multiple sensors simultaneously, such that the vehicle traveling distance is minimized. On the other hand, Liang *et al.* [19] recognized that the photos taken by a UAV at different location have redundant information. They studied how to dispatch an energy-limited UAV to monitor a disaster area, such that the non-redundant information collected in its flying tour is maximized.

There are a few studies on the multi-rooted min-max cycle cover problem with neighborhoods. For example, Kim *et al.* [15], [16] proposed an approximation algorithm for the problem by ignoring the service times of POIs, where the longest tour length is $(369 + \epsilon) \cdot OPT_m + c'_3 \cdot r$, where c'_3 is a positive constant. They also considered the use of K heterogeneous drones to monitor POIs, where both departure times and flying speeds of different drones are different. They proposed constant approximation algorithms [17]. Different from their studies, the proposed algorithm for the multi-rooted min-max cycle problem with neighborhoods in this paper considers not only the vehicle traveling time but also the POI service time. Furthermore, the maximum tour time delivered by the proposed algorithm is much shorter than that in [15], [16], which is only $(28 + \epsilon) \cdot OPT_m + c_3 \cdot r$, while the one in [15], [16] is $(369 + \epsilon) \cdot OPT_m + c'_3 \cdot r$ with $c_3 \leq c'_3$, i.e., $(28 + \epsilon) \cdot OPT_m + c_3 \cdot r \ll (369 + \epsilon) \cdot OPT_m + c'_3 \cdot r$.

VIII. Conclusion

In this paper we studied the min-max cycle cover problem with neighborhoods, by incorporating both neighborhoods and POI service time into consideration. We also proposed novel approximation algorithms for the problem, by exploring its combinatorial properties. We finally evaluated the proposed algorithms via experimental simulations. Experimental results showed that the proposed algorithms are very promising. Especially, the longest tour times in the solutions delivered by the proposed algorithms are only about from 80% to 90% of those by existing algorithms.

Acknowledgment

The authors would like to thank the three anonymous referees and the associate editor for their expertise comments and constructive suggestions, which have helped them to improve the quality and presentation of the article greatly.

References

[1] E. M. Arkin and R. Hassin, "Approximation algorithms for the geometric covering salesman problem," *Discrete Appl. Math.*, vol. 55, no. 3, pp. 197–218, Dec. 1994.

[2] C. A. B. Baker, S. Ramchurn, W. T. Teacy, and N. R. Jennings, "Planning search and rescue missions for UAV teams," in *Proc. Eur. Conf. Artif. Intell. (ECAI)*, 2017, pp. 1777–1782.

[3] L. Bertazzi, B. Golden, and X. Wang, "Min–max vs. Min–sum vehicle routing: A worst-case analysis," *Eur. J. Oper. Res.*, vol. 240, no. 2, pp. 372–381, Jan. 2015.

[4] A. Dumitrescu and J. S. B. Mitchell, "Approximation algorithms for TSP with neighborhoods in the plane," *J. Algorithms*, vol. 48, no. 1, pp. 135–159, Aug. 2003.

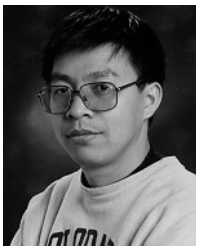
[5] A. Dumitrescu and C. D. Tóth, "The traveling salesman problem for lines, balls, and planes," *ACM Trans. Algorithms*, vol. 12, no. 3, pp. 1–29, Jun. 2016.

[6] A. Dumitrescu and C. D. Tóth, "Constant-factor approximation for TSP with disks," in *A Journey Through Discrete Mathematics*. Cham, Switzerland: Springer, 2017, pp. 375–390.

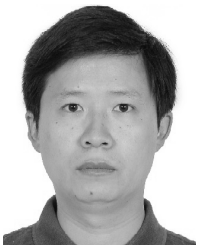
[7] M. Erdelj, E. Natalizio, K. R. Chowdhury, and I. F. Akyildiz, "Help from the sky: Leveraging UAVs for disaster management," *IEEE Pervasive Comput.*, vol. 16, no. 1, pp. 24–32, Jan. 2017.
[8] G. Even, N. Garg, J. Könemann, R. Ravi, and A. Sinha, "Min–max tree covers of graphs," *Oper. Res. Lett.*, vol. 32, no. 4, pp. 309–315, Jul. 2004.
[9] G. N. Frederickson, M. S. Hecht, and C. E. Kim, "Approximation algorithms for some routing problems," in *Proc. 17th Annu. Found. Comput. Sci. (FOCS)*, Oct. 1976, pp. 216–227.
[10] L. Gupta, R. Jain, and G. Vaszkun, "Survey of important issues in UAV communication networks," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 1123–1152, 2nd Quart., 2016.
[11] J. E. Hopcroft and R. M. Karp, "An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs," *SIAM J. Comput.*, vol. 2, no. 4, pp. 225–231, 1973.
[12] S. Jeong, O. Simeone, and J. Kang, "Mobile edge computing via a UAV-mounted cloudlet: Optimization of bit allocation and path planning," *IEEE Trans. Veh. Technol.*, vol. 67, no. 3, pp. 2049–2063, Mar. 2018.
[13] M. Khachay and K. Neznakhina, "Polynomial time solvable subclass of the generalized traveling salesman problem on grid clusters," in *Proc. Int. Conf. Anal. Images Social Netw. Texts*. Cham, Switzerland: Springer, 2017, pp. 346–355.
[14] M. R. Khani and M. R. Salavatipour, "Improved approximation algorithms for the min-max tree cover and bounded tree cover problems," *Algorithmica*, vol. 69, no. 2, pp. 443–460, Jun. 2014.
[15] 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," in *Proc. 31th IEEE Int. Conf. Comput. Commun.*, Mar. 2012, pp. 504–512.
[16] D. Kim, R. N. Uma, B. H. Abay, W. Wu, W. Wang, and A. O. Tokuta, "Minimum latency multiple data MULE trajectory planning in wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 13, no. 4, pp. 838–851, Apr. 2014.
[17] D. Kim, L. Xue, D. Li, Y. Zhu, W. Wang, and A. O. Tokuta, "On theoretical trajectory planning of multiple drones to minimize latency in search-and-reconnaissance operations," *IEEE Trans. Mobile Comput.*, vol. 16, no. 11, pp. 3156–3166, Nov. 2017.
[18] J. C. Latombe, *Robot Motion Planning*. New York, NY, USA: Springer, 2012.
[19] Y. Liang *et al.*, "Nonredundant information collection in rescue applications via an energy-constrained UAV," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2945–2958, Apr. 2019.
[20] W. Liang, Z. Xu, W. Xu, J. Shi, G. Mao, and S. K. Das, "Approximation algorithms for charging reward maximization in rechargeable sensor networks via a mobile charger," *IEEE/ACM Trans. Netw.*, vol. 25, no. 5, pp. 3161–3174, Oct. 2017.
[21] Y. Ma, W. Liang, and W. Xu, "Charging utility maximization in wireless rechargeable sensor networks by charging multiple sensors simultaneously," *IEEE/ACM Trans. Netw.*, vol. 26, no. 4, pp. 1591–1604, Aug. 2018.
[22] J. Ma, F. Meng, Y. Zhou, Y. Wang, and P. Shi, "Intelligent water pollution source identification and localization in wireless sensor networks," in *Proc. 2nd IEEE Adv. Inf. Manage., Communicates, Electron. Autom. Control Conf. (IMCEC)*, May 2018, pp. 1300–1305.
[23] J. S. B. Mitchell, "A constant-factor approximation algorithm for TSP with pairwise-disjoint connected neighborhoods in the plane," in *Proc. Annu. Symp. Comput. Geometry (SoCG)*, 2010, pp. 183–191.
[24] X. Ren, W. Liang, and W. Xu, "Data collection maximization in renewable sensor networks via time-slot scheduling," *IEEE Trans. Comput.*, vol. 64, no. 7, pp. 1870–1883, Jul. 2015.
[25] X. Wang, A. Chowdhery, and M. Chiang, "Networked drone cameras for sports streaming," in *Proc. IEEE 37th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jun. 2017, pp. 308–318.
[26] C. Wang, J. Li, F. Ye, and Y. Yang, "A mobile data gathering framework for wireless rechargeable sensor networks with vehicle movement costs and capacity constraints," *IEEE Trans. Comput.*, vol. 65, no. 8, pp. 2411–2427, Aug. 2016.
[27] Q. Wu, L. Liu, and R. Zhang, "Fundamental trade-offs in communication and trajectory design for UAV-enabled wireless network," *IEEE Wireless Commun.*, vol. 26, no. 1, pp. 36–44, Feb. 2019.
[28] W. Xu, W. Liang, H. Kan, Y. Xu, and X. Zhang, "Minimizing the longest charge delay of multiple mobile chargers for wireless rechargeable sensor networks by charging multiple sensors simultaneously," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2019, pp. 881–890.
[29] W. Xu, W. Liang, X. Jia, H. Kan, Y. Xu, and X. Zhang, "Minimizing the maximum charging delay of multiple mobile chargers under the multi-node energy charging scheme," *IEEE Trans. Mobile Comput.*, early access, Feb. 14, 2020, doi: 10.1109/TMC.2020.2973979.
[30] W. Xu, W. Liang, X. Jia, Z. Xu, Z. Li, and Y. Liu, "Maximizing sensor lifetime with the minimal service cost of a mobile charger in wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 17, no. 11, pp. 2564–2577, Nov. 2018.
[31] W. Xu, W. Liang, and X. Lin, "Approximation algorithms for min-max cycle cover problems," *IEEE Trans. Comput.*, vol. 64, no. 3, pp. 600–613, Mar. 2015.
[32] W. Xu, W. Liang, X. Lin, and G. Mao, "Efficient scheduling of multiple mobile chargers for wireless sensor networks," *IEEE Trans. Veh. Technol.*, vol. 65, no. 9, pp. 7670–7683, Sep. 2016.
[33] Y. Yan and Y. Mostofi, "Co-optimization of communication and motion planning of a robotic operation under resource constraints and in fading environments," *IEEE Trans. Wireless Commun.*, vol. 12, no. 4, pp. 1562–1572, Apr. 2013.
[34] W. Yu and Z. Liu, "Better approximability results for min–max tree/cycle/path cover problems," *J. Combinat. Optim.*, vol. 37, no. 2, pp. 563–578, Feb. 2019.
[35] W. Yu and Z. Liu, "Improved approximation algorithms for some min-max and minimum cycle cover problems," *Theor. Comput. Sci.*, vol. 654, pp. 45–48, Nov. 2016.
[36] W. Yu, Z. Liu, and X. Bao, "New approximation algorithms for the minimum cycle cover problem," in *Proc. Int. Workshop Frontiers Algorithmics*. Cham, Switzerland: Springer, 2018, pp. 81–95.
[37] Y. Zeng and R. Zhang, "Energy-efficient UAV communication with trajectory optimization," *IEEE Trans. Wireless Commun.*, vol. 16, no. 6, pp. 3747–3760, Jun. 2017.
[38] C. Zhan, Y. Zeng, and R. Zhang, "Energy-efficient data collection in UAV enabled wireless sensor network," *IEEE Wireless Commun. Lett.*, vol. 7, no. 3, pp. 328–331, Jun. 2018.
[39] Y. Zhang, S. He, and J. Chen, "Near optimal data gathering in rechargeable sensor networks with a mobile sink," *IEEE Trans. Mobile Comput.*, vol. 16, no. 6, pp. 1718–1729, Jun. 2017.
[40] R. Zhang, J. Peng, W. Xu, W. Liang, Z. Li, and T. Wang, "Utility maximization of temporally correlated sensing data in energy harvesting sensor networks," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 5411–5422, Jun. 2019.

Lijia Deng received the B.Sc. degree in computer science from Hainan University, China, in 2017. He is currently pursuing the master's degree in computer science with Sichuan University. His research interests include wireless sensor networks, algorithm design and analysis, and approximation algorithms.

Wenzheng Xu (Member, IEEE) received the B.Sc., M.E., and Ph.D. degrees in computer science from Sun Yat-sen University, Guangzhou, China, in 2008, 2010, and 2015, respectively. He was a Visitor with the Australian National University and the Chinese University of Hong Kong. He is currently an Associate Professor with Sichuan University. His research interests include wireless ad hoc and sensor networks, mobile computing, approximation algorithms, combinatorial optimization, online social networks, and graph theory.



Weifa Liang (Senior Member, IEEE) received the B.Sc. degree from Wuhan University, China, in 1984, the M.E. degree from the University of Science and Technology of China in 1989, and the Ph.D. degree from Australian National University in 1998, all in computer science. He is currently a Full Professor with the Research School of Computer Science, Australian National University. His research interests include the design and analysis of energy-efficient routing protocols for wireless ad hoc and sensor networks, mobile edge computing and cloud computing, software-defined networking, online social networks, design and analysis of parallel and distributed algorithms, approximation algorithms, combinatorial optimization, and graph theory.



Jian Peng received the B.A. and Ph.D. degrees from the University of Electronic Science and Technology of China (UESTC) in 1992 and 2004, respectively. He is currently a Professor with the College of Computer Science, Sichuan University. His current research interests include wireless sensor networks, big data, and cloud computing.

Yingjie Zhou (Member, IEEE) received the Ph.D. degree from the School of Communication and Information Engineering, University of Electronic Science and Technology of China (UESTC), China, in 2013. He was a Visiting Scholar with the Department of Electrical Engineering, Columbia University, New York. He is currently an Assistant Professor with the College of Computer Science, Sichuan University (SCU), China. His current research interests include network measurement, behavioral data analysis, resource allocation, and neural networks.

Lei Duan (Member, IEEE) received the B.Sc. and Ph.D. degrees in computer science from Sichuan University in 2003 and 2008, respectively. He was a Visiting Ph.D. Student with the Department of Computer Science and Engineering, Wright State University, from 2007 to 2008, and a Visiting Scholar with the School of Computing Science, Simon Fraser University, from 2012 to 2013. He is currently a Professor with the School of Computer Science, Sichuan University. His research interests include data mining, knowledge management, evolutionary computation, bioinformatics, and health-informatics.

Sajal K. Das (Fellow, IEEE) is currently the Chair of Computer Science Department and the Daniel St. Clair Endowed Chair with the Missouri University of Science and Technology, USA. His current research interests include the theory and practice of wireless sensor networks, big data, cyber-physical systems, smart healthcare, distributed and cloud computing, security and privacy, biological and social networks, applied graph theory, and game theory. He directed numerous funded projects in these areas totaling over $15M and published extensively with more than 600 research articles in high-quality journals and refereed conference proceedings. He serves as the founding Editor-in-Chief for the *Pervasive and Mobile Computing Journal* and an Associate Editor for the IEEE TRANSACTIONS ON MOBILE COMPUTING, *ACM Transactions on Sensor Networks*, and so on. He is a Co-Founder of the IEEE PerCom, the IEEE WoWMoM, and ICDCN conferences, and served on numerous conference committees as a general chair, a program chair, or a program committee member.