Better Approximation Ratios for the Single-Vehicle Scheduling Problems on Tree/Cycle Networks

Yuanxiao Wu and Xiwen Lu(✉)

East China University of Science and Technology, Shanghai, China
yxwu0212@163.com, xwlu@ecust.edu.cn

Abstract. We investigate the single vehicle scheduling problems based on tree/cycle networks. Each customer, assumed as a vertex on the given network, has a release time and a service time requirements. The single vehicle starts from the depot and aims to serve all the customers. The objective of the problem is to find the relatively optimal routing schedule so as to minimize the makespan. We provide a $\frac{16}{9}$-approximation algorithm and a $\frac{48}{25}$-approximation algorithm for the tour-version and the path-version of single vehicle scheduling problem on a tree, respectively. For the tour-version of single vehicle scheduling problem on a cycle, we present a $\frac{5}{3}$-approximation algorithm.

Keywords: Vehicle · Routing · Scheduling · Network · Approximation algorithm

1 Introduction

The single vehicle scheduling problem (SVSP) consists of a set of customers situated at different vertices on a given network and a single vehicle initially located at a fixed depot. Each customer has a release time before which it cannot be served, and a service time which the vehicle has to spend in serving the customer. The vehicle, required to serve all the customers, takes a travel time when it travels from one customer to another. The completion time of a customer is defined as the time by which it has been served completely, while the completion time of the vehicle means the time by which it has served all the customers and returned to its initial location. A permutation of the customers, which implies the customer service order and can specify the routing of the vehicle, is considered as a schedule for the problem. The problem aims to find a schedule to minimize the makespan. We distinguish two versions. In the first one, which is called tour-version, the makespan is defined as the completion time of the vehicle. In the other one, which is known as path-version, the makespan means the completion time of the last served customer. For convenience, when the network is restricted to a line (resp. tree, cycle), we denote the single vehicle scheduling problem by L-SVSP (resp. T-SVSP, C-SVSP). When the service time of each

X. Gao et al. (Eds.): COCOA 2017, Part I, LNCS 10627, pp. 313–323, 2017.
https://doi.org/10.1007/978-3-319-71150-8_27

customer is zero, SVSP is known as single vehicle routing problem (SVRP) in some paper. Thus we denote SVRP on a line, tree and cycle by L-SVRP, T-SVRP and C-SVRP, respectively.

There are plenty of results on VRP and VSP. [1] showed that both the tour-version and the path version of L-SVSP are ordinarily NP-hard. [2] presented polynomial time algorithms for both versions of L-SVRP. [3] provided a $\frac{3}{2}$-approximation algorithm for the tour-version of L-SVSP in the case that the vehicle initial locates at an extreme vertex, and [4] gave a $\frac{5}{3}$-approximation algorithm for the counterpart in which the initial location of the vehicle is arbitrary. [5,6] presented a $\frac{3}{2}$-approximation algorithm for the tour-version of L-SVSP and a $\frac{5}{3}$-approximation algorithm for the path-version of L-SVSP with no constraint on the initial location of the vehicle, and [6] also provided examples to show that the performance ratios are tight. When it comes to multi-vehicle scheduling problem (MVSP), [7] provided a 2-approximation algorithm for the path-version of the general L-MVSP in which the initial location of each vehicle is arbitrary. [8] showed that both the tour-version and path-version of L-MVRP can be solved in polynomial time.

It has been shown in [9] that both the tour-version and the path-version of T-SVRP (and hence T-SVSP) are ordinarily NP-hard. For the tour-version of T-SVSP, they showed that the problem can be exactly solved in $O(n \log n)$ time if adding a constraint that the vehicle has to process all tasks in a depth-first manner, and the $O(n \log n)$ time algorithm can be considered as a 2-approximation algorithm for the T-SVSP without the depth-first constraint. [10] ultimately proved that both versions of T-SVSP are strongly NP-hard. For the tour-version, they provided a $O(n^b)$ time DP algorithm where b is the number of leaves. [5] introduced an improved $\frac{11}{6}$-approximation algorithm for the tour-version of T-SVSP. In their algorithm, they partitioned the set of customers into two subsets. The vehicle first serves part of customers in one subset, and then serves all the remaining customers. They also presented a $\frac{9}{5}$-approximation algorithm with a similar method for the tour-version of C-SVSP. [11] provided a $\frac{9}{5}$-approximation algorithm and a $\frac{27}{14}$-approximation algorithm for the tour-version and the path-version of T-SVSP, respectively. For both tour-version and path-version of C-SVSP, they provided a $\frac{12}{7}$-approximation algorithm. [12] presented polynomial time approximation schemes for both versions of SVSP on a tree with a constant number of leaves. [13] presented a 3-approximation algorithm for MVSP on a tree and a $(5 - \frac{2}{m})$-approximation algorithm for that on a general network.

In this paper, we consider both the tour-version and the path-version of T-SVSP and the tour-version of C-SVSP. For each problem, we propose an approximation algorithm and prove its performance ratio. Our algorithms are improvements on those in [11].

The rest of this paper is structured as follows. In Sect. 2, we introduce the formulation of T-SVSP and some notations. In Sect. 3, we present approximation algorithms for two versions of T-SVSP and analyse the performance ratio. In Sect. 4, we describe an approximation algorithm for the tour-version of C-VSP and prove the performance ratio. Finally we give some concluding remarks in Sect. 5.

2 Problem Formulation and Preliminaries

The SVSP discussed in this paper is mathematically defined as follows. Let $G = (V \cup \{0\}, E)$ be an undirected network where $V = \{1, 2, ..., n\}$ is a vertex set and E is a set of edges. An edge $e \in E$ is an unordered pair (j, k) of two vertices in V, where j, k are called the endpoints of e. The travel time of the vehicle is $t_{j,k} \geq 0$, which is the time to traverse edge $e = (j, k)$ from j to k, and $t_{k,j} = t_{j,k}$. When $(j, k) \notin E$, $t_{k,j}$ denotes the travel time for the vehicle travelling along the shortest path from j to k. There is a unique customer i at each vertex $i \in V$. Unless ambiguity would result, we do not distinguish between vertex and customer. There is a vehicle initially located at the depot 0 to serve all the costumers. Each customer i is associated with a release time r_i and a service time p_i. It means that the vehicle cannot start serving customer i before r_i, and needs p_i time units to finish its service. The vehicle arriving at a vertex i before r_i either waits until r_i to serve the customer v_i, or moves to other vertices without serving v_i if it is more advantageous (in this case, the vehicle has to come back to i later to serve customer i). A routing schedule of the vehicle is specified by a sequence $\pi = (\pi(1), \pi(2), ..., \pi(n))$ of customers to be served, i.e. the vehicle travels along a shortest path from 0 to $\pi(1)$ in G, taking the travel time of the length of the path, waits until $r_{\pi(1)}$ if the arrival time is before $t = r_{\pi(1)}$ and serve the customer $\pi(1)$. After serving $\pi(1)$, it immediately moves to $\pi(2)$, waits until $r_{\pi(2)}$ if the arrival time is before $t = r_{\pi(2)}$ and serve the customer $\pi(2)$, and so on. In the following, for any feasible schedule π, we always assume $\pi(0) = \pi(n+1) = 0$. Let $C_{[i]}(\pi)$ denote the service completion time of customer $\pi(i)$ in π, and set $C_{[0]}(\pi) = 0$. Then, $C_{[i]}(\pi)$ equals to $\max\{r_{\pi(i)}, C_{[i-1]}(\pi) + t_{\pi(i-1),\pi(i)}\} + p_{\pi(i)}$ for all $i = 1, 2, ..., n$. The makespan of π is donoted by $C^{tour}_{max}(\pi)$ in the tour-version, and $C^{path}_{max}(\pi)$ in the path-version. Then, $C^{tour}_{max}(\pi) = C_{[n]}(\pi) + t_{0,\pi(n)}$, $C^{path}_{max}(\pi) = C_{[n]}(\pi)$.

We now introduce some notations to be used throughout the article. Let

$$\begin{aligned} t_{max} &= max_{1 \leq i \leq n} t_{0,i} \\ L &= \textstyle\sum_{(i,j) \in E} t_{i,j} \\ r_{max} &= max_{1 \leq i \leq n} r_i \\ P &= \textstyle\sum_{i=1}^{n} p_i \end{aligned} \tag{1}$$

and for $0 \leq t \leq r_{max}$,

$$\begin{aligned} V(t) &= \{i \in V \mid r_i \geq t\} \\ P(t) &= \textstyle\sum_{i \in V(t)} p_i \\ V'(t) &= \{i \in V \mid r_i > t\} \\ P'(t) &= \textstyle\sum_{i \in V'(t)} p_i \end{aligned} \tag{2}$$

Note that $P(t)$ and $P'(t)$ are piecewise constant functions of t, and different only at $r_j (j = 1, 2, ..., n)$.

For $0 \leq t \leq r_{max}$, let $v(t)$ and $v'(t)$ denote the farthest vertices from vertex 0 in $V(t)$ and $V'(t)$, respectively. We also define

θ: the shortest travelling tour over $V \cup \{0\}$

$\delta'_0(t)$: the shortest travelling path over $V'(t) \cup \{0\}$ starting from the depot 0.

When $G = (V \cup \{0\}, E)$ is a tree, for any $U \subset V \cup \{0\}$, we define the spanning subtree on U as the smallest subtree of G which contains all the vertices in U. Furthermore, for $0 \leq t \leq r_{max}$, we let $T(t)$, $T'(t)$, $\widehat{T}(t)$ and $\widehat{T}'(t)$ denote the spanning subtree on the vertex sets $V(t)$, $V'(t)$, $V(t) \cup \{0\}$ and $V'(t) \cup \{0\}$, respectively.

In the following, a symbol denoting a subgraph of G also be considered as its length, i.e. if G' is a subgraph of G, then $G' = \sum_{(j,k)\in G'} t_{j,k}$.

3 SVSP on Tree Network

In this section we consider T-SVSP. Without loss of generality, we consider the depot 0 as the root of the tree. A $\frac{16}{9}$-approximation algorithm for the tour-version is presented in Sect. 3.1, and a $\frac{48}{25}$-approximation algorithm for the path-version is provided in Sect. 3.2.

3.1 Tour-Version of T-SVSP

Karuno et al. [3] provided a $\frac{11}{6}$-approximation algorithm for the tour-version of T-SVSP. They first gave several different candidate schedules and then chose the best one as the approximation solution. Yu and Liu [6] proved that the tour version of T-SVSP has an r-approximation algorithm if the tour version of T-SVRP has. Then they presented a $\frac{9}{5}$-approximation algorithm for the tour-version of T-SVRP. We will adopt a similar approach and show the approximation ratio can be reduced to $\frac{16}{9}$.

Algorithm 1 for the tour-version of T-SVRP

Step 1. We define σ as a tour on the tree in which the vehicle starts out from the depot 0 and visits $v(0)$ first, and then visits other customers in a depth-first sequence, and finally returns to the depot. Construct a schedule π_1 such that the service order of the customers is the same as the visiting order of the customers in σ.

Step 2. Let x denote the point that is $L + \frac{t_{max}}{2}$ time units away from the depot 0 along σ. Let $t^* = max\{2L, r_{max}\}$. Construct a schedule π_2 such that the vehicle first waits at vertex 0 for $t^* - L - \frac{t_{max}}{2}$ time units, and then travels along σ without serving any customer until it arrives at $v(0)$, and then travels along σ from vertex $v(0)$ to point x to serve the customers in $V \backslash V'(t^* - L + \frac{t_{max}}{2})$, and then travels to vertex 0 to serve all the remaining customers in a depth-first sequence.

Step 3. Construct a schedule π_3 such that the vehicle first waits at vertex 0 for $t^* - L + \frac{t_{max}}{2}$ time units, and then travels reverse of σ to point x to serve the customers in $V \backslash V'(t^* - L + \frac{t_{max}}{2})$, and then travels to vertex 0 to serve all the remaining customers in a depth-first sequence.

Step 4. Choose the best one among π_1,π_2 and π_3 as the approximation solution π.

In the proof of the following theorem, we first propose several upper bounds on the makespan of π_1, π_2 and π_3, and then prove that Algorithm 1 is a $\frac{16}{9}$-approximation algorithm to the tour-version of T-SVSP. Let π^* denote the optimal schedule for the tour-version of T-SVRP.

Theorem 1. *Algorithm 1 is a $\frac{16}{9}$-approximation algorithm to the tour-version of T-SVRP.*

Proof. The proof will be presented in three steps.

Step 1: We prove an upper bound on $C^{tour}_{max}(\pi_1)$.

In π_1, the vehicle either waits at some customers for their release or doesn't wait at any vertex. If the vehicle waits at some customers, assume customer j is the last customer where the vehicle waits. Because the first served customer is $v(0)$, the travel time of the vehicle before it arriving at j is at least $t_{0,v(0)}+t_{v(0),j}$. The total travel time of the vehicle in π_1 is $2L$. Then, the travel time of the vehicle after it serves customer j is no more than $2L - t_{0,v(0)} - t_{v(0),j}$. Thus,

$$\begin{aligned} C^{tour}_{max}(\pi_1) &\leq r_j + 2L - t_{0,v(0)} - t_{v(0),j} \\ &= r_j + t_{0,j} + 2L - t_{0,v(0)} - t_{v(0),j} - t_{0,j} \\ &\leq 2C^{tour}_{max}(\pi^*) - 2t_{max} \end{aligned} \tag{3}$$

where the last equality follows $r_j + t_{0,j} \leq C^{tour}_{max}(\pi^*)$, $2L \leq C^{tour}_{max}(\pi^*)$ and $t_{max} = t_{0,v(0)} \leq t_{v(0),j} + t_{0,j}$.

If the vehicle doesn't wait at any vertex, then

$$\begin{aligned} C^{tour}_{max}(\pi_1) &= 2L \\ &= C^{tour}_{max}(\pi^*) \\ &\leq 2C^{tour}_{max}(\pi^*) - 2t_{max} \end{aligned} \tag{4}$$

Combing the above two cases, we conclude that

$$C^{tour}_{max}(\pi_1) \leq 2C^{tour}_{max}(\pi^*) - 2t_{max}. \tag{5}$$

Step 2: We show an upper bound on $min\{C^{tour}_{max}(\pi_2), C^{tour}_{max}(\pi_3)\}$.

We first consider the total waiting time of the vehicle in π_2 and π_3. In both π_2 and π_3, the time of the vehicle arriving at point x is t^*. Since $t^* \geq r_{max}$, the vehicle does not wait at any customer in the later process. Thus, the total waiting time of the vehicle in π_2 is $t^* - L - \frac{t_{max}}{2}$, and that in π_3 is $t^* - L + \frac{t_{max}}{2}$. The total waiting time of the vehicle in π_2 and π_3 is $2t^* - 2L$.

Now we turns to provide an upper bound of the total travel time of the vehicle in π_2 and π_3. Let T_1 and T_2 be subtrees visited in π_2 and π_3, respectively. It is easy to see that the travel times of the vehicle in π_2 and π_3 are no more than $2L + 2((T_1 \backslash T_2) \cap \widehat{T}'(t^* - L + \frac{t_{max}}{2}))$ and $2L + 2((T_2 \backslash T_1) \cap \widehat{T}'(t^* - L + \frac{t_{max}}{2}))$, respectively. Since $2((T_1 \backslash T_2) \cap \widehat{T}'(t^* - L + \frac{t_{max}}{2})) + 2((T_2 \backslash T_1) \cap \widehat{T}'(t^* - L + \frac{t_{max}}{2})) \leq 2\widehat{T}'(t^* - L + \frac{t_{max}}{2})$, the total travel time of the vehicle in π_2 and π_3 is at most $4L + 2\widehat{T}'(t^* - L + \frac{t_{max}}{2})$, which is no more than $4L + 2\widehat{T}'(L + \frac{t_{max}}{2})$ for $t^* - L \geq L$.

Combing the above two discussion,

$$\begin{aligned} C_{max}^{tour}(\pi_2) + C_{max}^{tour}(\pi_3) &\leq 2t^* - 2L + 4L + 2\widehat{T}'(L + \tfrac{t_{max}}{2}) \\ &= 2t^* + 2L + 2\widehat{T}'(L + \tfrac{t_{max}}{2}) \end{aligned} \tag{6}$$

A straightforward conclusion of the inequality above is showed as follows.

$$\begin{aligned} &\min\{C_{max}^{tour}(\pi_2), C_{max}^{tour}(\pi_3)\} \\ &\leq \tfrac{1}{2}(C_{max}^{tour}(\pi_2) + C_{max}^{tour}(\pi_3)) \\ &\leq t^* + L + \widehat{T}'(L + \tfrac{t_{max}}{2}) \\ &\leq t^* + L + (C_{max}^{tour}(\pi^*) - L - \tfrac{t_{max}}{2} + t_{0,v(L+\frac{t_{max}}{2})})/2 \\ &\leq \tfrac{7}{4} C_{max}^{tour}(\pi^*) + \tfrac{t_{max}}{4} \end{aligned} \tag{7}$$

where the second inequality follows $C_{max}^{tour}(\pi^*) \geq t + 2\widehat{T}(t) - t_{0,v(t)}$ and $\widehat{T}'(t) \leq \widehat{T}(t)$, the last inequality follows $t_{0,v(L+\frac{t_{max}}{2})} \leq t_{max}$ and $C_{max}^{tour}(\pi^*) \geq t^* \geq 2L$.

Step 3: We prove the correctness of Theorem 1.

Combining the conclusion of the above two steps,

$$\begin{aligned} &\min\{C_{max}^{tour}(\pi_1), C_{max}^{tour}(\pi_2), C_{max}^{tour}(\pi_3)\} \\ &\leq \min\{2C_{max}^{tour}(\pi^*) - 2t_{max}, \tfrac{7}{4} C_{max}^{tour}(\pi^*) + \tfrac{t_{max}}{4}\} \\ &\leq \tfrac{16}{9} C_{max}^{tour}(\pi^*) \end{aligned} \tag{8}$$

This completes the proof of Theorem 1. □

[11] showed that, in linear time, each instance $\mathcal{I}$ of the tour-version of T-SVSP can be transformed into an instance $\mathcal{I}'$ of the tour-version of T-SVRP such that any r-approximation schedule π' of $\mathcal{I}'$ also can be transformed into an r-approximation schedule of $\mathcal{I}$. Therefore, we design the following $\frac{16}{9}$-approximation algorithm for the tour-version of T-SVSP.

Algorithm 2 for the tour-version of T-SVSP

Step 1. Given an instance $\mathcal{I} = (T = (V \cup \{0\}, E), r, t, p)$ of T-SVSP, construct an auxiliary instance $\mathcal{I}'(T' = (V' \cup \{0\}, E'), r', t')$ of T-SVRP as follows. $V' = V \cup \bigcup_{i=1}^{n} n+i$, $E' = E \cup \bigcup_{i=1}^{n} (i, n+i)$, $t'_{j,k} = t_{j,k}$ for each edge $(j,k) \in E$, $t'_{i,n+i} = \frac{p_i}{2}$, $r'_i = r_i$ and $r'_{n+i} = r_i + \frac{p_i}{2}$ for each $i \in V$.

Step 2. Call Algorithm 1 to solve auxiliary instance $\mathcal{I}'$ of T-SVRP and obtain a schedule π'.

Step 3. Construct a schedule π such that customer i is served before customer j if customer $n+i$ is served before customer $n+j$ in π' for each pair $i, j \in V$.

Theorem 2. *Algorithm 2 is a $\frac{16}{9}$-approximation algorithm to the tour-version of T-SVSP.*

Proof. Theorem 2 is a direct inference of Theorem 1. □

3.2 Path-Version of T-SVSP

When it comes to the path-version of T-SVSP, notice that the gap between the makespan of a schedule for the tour-version of T-SVSP and that of the same schedule for the path-version of T-VSP is no more than t_{max}. Therefore, the approximation algorithm for the tour-version of T-VSP can also be used to solve the path-version of T-SVSP.

Algorithm 3 for the path-version of T-SVSP

Step 1. Call Algorithm 2 to obtain a schedule π_1 for the corresponding tour-version of T-SVSP.

Step 2. Let $P_{0,v(0)}$ indicate the unique path between the vertices 0 and $v(0)$ in the tree G. It is easy to see that $t_{0,v(0)} = t_{max}$. We assume that there are $m+2$ vertices $0, 1, 2, ..., m+1 = v(0)$ on path $P_{0,v(0)}$. Then, deleting the edges in $P_{0,v(0)}$, we obtain $m+2$ subtrees, which can be described as $T^i (0 \leq i \leq m+1)$ such that T^i is connected with vertex i. Let V^i denote the vertex set of T^i. Solving the auxiliary L-SVRP on the path $P_{0,v(0)}$, where the release time of vertex i is redefined as $max\{r_j \mid j \in V^i\}$, by the dynamic programming algorithm of [2], we obtain a service order of $V^i s$. Then, serving the customers in each V^i in an arbitrary depth-first order, we obtain a schedule π_2.

Step 3. Choose the best one between π_1 and π_2 as the approximate solution π.

In the proof of the following theorem, we first propose several upper bounds on the makespan of π_1 and π_2, and then prove that Algorithm 3 is a $\frac{48}{25}$-approximation algorithm to the path-version of T-SVSP. Let π' and π^* denote the optimal schedule for the tour-version of T-SVSP and the path-version of T-SVSP, respectively.

Theorem 3. *Algorithm 3 is a $\frac{48}{25}$-approximation algorithm to the path-version of T-SVSP.*

Proof. The proof will be presented in three steps.

Step 1. We prove an upper bound on $C^{path}_{max}(\pi_1)$.

Let j denote the last customer served by the vehicle in π^*. It is easy to see that $C^{tour}_{max}(\pi^*) - C^{path}_{max}(\pi^*) \leq t_{0,j}$, and $t_{0,j} \leq t_{max}$. Then, we obtain $C^{tour}_{max}(\pi^*) \leq C^{path}_{max}(\pi^*) + t_{max}$. Thus,

$$\begin{aligned} C^{path}_{max}(\pi_1) &\leq C^{tour}_{max}(\pi_1) \leq \tfrac{16}{9} C^{tour}_{max}(\pi') \\ &\leq \tfrac{16}{9} C^{tour}_{max}(\pi^*) \\ &\leq \tfrac{16}{9} C^{path}_{max}(\pi^*) + \tfrac{16}{9} t_{max} \end{aligned} \tag{9}$$

Step 2. We show an upper bound on $C^{path}_{max}(\pi_2)$.

It is easy to see that the optimal makespan of the auxiliary L-SVRP is a lower bound of $C^{path}_{max}(\pi^*)$. Compared with the optimal makespan of L-SVRP,

the makespan of π_2 increases at most $2(L - t_{max}) + P$ time units for travelling the subtrees $T^0, T^1, ..., T^{m+1}$ and serving all the customers. Then, we obtain

$$\begin{aligned} C^{path}_{max}(\pi_2) &\le C^{path}_{max}(\pi^*) + 2(L - t_{max}) + P \\ &\le 2C^{path}_{max}(\pi^*) - t_{max} \end{aligned} \tag{10}$$

Step 3. We prove the correctness of Theorem 3.

$$\begin{aligned} &\min\{C^{path}_{max}(\pi_1), C^{path}_{max}(\pi_2)\} \\ \le\ &\min\{\tfrac{16}{9}C^{path}_{max}(\pi^*) + \tfrac{16}{9}t_{max}, 2C^{path}_{max}(\pi^*) - t_{max}\} \\ \le\ &\tfrac{48}{25}C^{path}_{max}(\pi^*) \end{aligned} \tag{11}$$

This completes the proof of Theorem 3. □

4 SVSP on Cycle Network

We now turn to the tour version of C-SVSP. Let $G = (V \cup \{0\})$ be a cycle, where the vertices in $V \cup \{0\}$ are numbered increasingly in the counterclockwise order. In the following of this section, we consider vertex $n + 1$ as vertex 0.

Recall the definitions of $L, \theta, \delta'_0(t)$ in Sect. 2. We assume that there is no edge $(i, i + 1)$ such that $t_{i,i+1} \ge \frac{1}{2}L$. Otherwise there exists an optimal schedule which never goes through the edge $(i, i+1)$, thus the tour-version of C-SVSP can be considered as a tour-version of L-SVSP. Since [6] showed a $\frac{3}{2}$-approximation algorithm for the tour-version of L-SVSP, we focus on the situation satisfying $t_{i,i+1} < \frac{1}{2}L$ for all $i = 0, 1, ..., n$, which implies $\theta = L$. Suppose that $V'(t) = \{i_1, i_2, ..., i_k\}$ with $i_1 < i_2 < ... < i_k$, and $i_0 = i_{k+1} = 0$. Then $\delta'_0(t)$ is the shortest one among the following paths:

(i) for $0 \le j \le k - 1$, the paths first from i_0 to i_j in the counterclockwise direction, and then from i_j to i_{j+1} in the clockwise direction;
(ii) for $2 \le j \le k + 1$, the paths first from i_0 to i_j in the clockwise direction, and then from i_j to i_{j-1} in the counterclockwise direction.

Then, we provide a $\frac{5}{3}$-approximation algorithm for the tour-version of C-SVSP as follows.

Algorithm 4 for the tour-version of C-SVSP

Step 1. Solve the corresponding C-SVRP by the dynamic programming algorithm of [10] to generate a schedule π_1.
Step 2. Find $t^*(0 \le t^* \le r_{max})$ such that $P'(t^*) + \delta'_0(t^*) \le 2t^* \le P(t^*) + \delta_0(t^*)$. Partition the customers into $V \backslash V'(t^*)$ and $V'(t^*)$. Let $v_{\delta'_0(t^*)}$ denote the other end point differing from vertex 0 in path $\delta'_0(t^*)$. Construct π_2 in which the vehicle first travels to $v_{\delta'_0(t^*)}$ and waits until t^*(if necessary), then goes through the cycle to serve all the customers in $V \backslash V'(t^*)$ before it arrives $v_{\delta'_0(t^*)}$ again, and travels along $\delta'_0(t^*)$ to serve the customers in $V'(t^*)$.
Step 3. Choose the best one among π_1 and π_2 as the approximate solution π.

In the proof of the following theorem, we first propose several upper bounds on the makespan of π_1 and π_2, and then prove that Algorithm 4 is a $\frac{5}{3}$-approximation algorithm to the tour-version of T-SVSP. Let π^* denote the optimal schedule for the tour-version of C-SVSP.

Theorem 4. *Algorithm 4 is an $\frac{5}{3}$-approximation algorithm to the tour-version of C-SVSP.*

Proof. The proof will be presented in three steps.

Step 1. We show that $C_{max}^{tour}(\pi_1) \leq C_{max}^{tour}(\pi^*) + P$.

The optimal makespan of the C-SVRP is a lower bound of $C_{max}^{tour}(\pi^*)$. As a solution to C-SVSP, π_1 increases at most P time units in makespan than as an optimal solution to C-SVRP. Then, $C_{max}^{tour}(\pi_1) \leq C_{max}^{tour}(\pi^*) + P$.

Step 2. We prove that $C_{max}^{tour}(\pi_2) \leq max\{\frac{7}{3}C_{max}^{tour}(\pi^*) - P, \frac{5}{2}C_{max}^{tour}(\pi^*) - \frac{3}{2}P, \frac{5}{3}C_{max}^{tour}(\pi^*)\}$.

The vehicle does not wait at any customer in $V \backslash V'(t^*)$, because it starts out from vertex $v_{\delta_0'(t^*)}$ at or later than time t^*. If the vehicle does not wait at any customer in $V'(t^*)$, we have

$$\begin{aligned} C_{max}^{tour}(\pi_2) &\leq max\{t^*, t_{0,v_{\delta_0'(t^*)}}\} + \theta + \delta_0'(t^*) + P \\ &\leq max\{\tfrac{1}{3}C_{max}^{tour}(\pi^*), \tfrac{1}{2}\theta\} + 2\theta + P \\ &\leq max\{\tfrac{7}{3}C_{max}^{tour}(\pi^*) - P, \tfrac{5}{2}C_{max}^{tour}(\pi^*) - \tfrac{3}{2}P\} \end{aligned} \tag{12}$$

where the second inequality follows from $C_{max}^{tour}(\pi^*) \geq t^* + \delta_0(t^*) + P(t^*) \geq 3t^*$, $t_{0,v'(t^*)} \leq \frac{1}{2}\theta$ and $\delta_0'(t^*) \leq \theta$, and the last inequality follows from $C_{max}^{tour}(\pi^*) \geq \theta + P$.

If the vehicle waits at some customer in $V'(t^*)$, let k be the last customer where the vehicle waits. Then we have

$$\begin{aligned} C_{max}^{tour}(\pi_2) &\leq r_k + P'(t^*) + \delta_0'(t^*) \\ &\leq \tfrac{5}{3}C_{max}^{tour}(\pi^*) \end{aligned} \tag{13}$$

where the last inequality follows from $C_{max}^{tour}(\pi^*) \geq r_i + t_{0,i},\ for\ any\ i \in V$ and $C_{max}^{tour}(\pi^*) \geq t^* + \delta_0(t^*) + P(t^*) \geq \frac{3}{2}(\delta_0'(t^*) + P'(t^*))$.

Step 3. We prove the minimum one between the makespans of π_1 and π_2 is at most $\frac{5}{3}C_{max}^{tour}(\pi^*)$.

$$\begin{aligned} &\min\{C_{max}^{tour}(\pi_1), C_{max}^{tour}(\pi_2)\} \\ &\leq \min\{C_{max}^{tour}(\pi^*) + P, max\{\tfrac{7}{3}C_{max}^{tour}(\pi^*) - P, \tfrac{5}{2}C_{max}^{tour}(\pi^*) - \tfrac{3}{2}P, \tfrac{5}{3}C_{max}^{tour}(\pi^*)\}\} \\ &= \tfrac{5}{3}C_{max}^{tour}(\pi^*) \end{aligned} \tag{14}$$

This completes the proof. □

5 Conclusions

In this paper, we consider the single-vehicle scheduling problems on a tree and a cycle, and all of these problems are known to be NP-hard. For the tour-version

and the path-version of T-SVSP, we present a $\frac{16}{9}$-approximation algorithm and a $\frac{48}{25}$-approximation algorithm, respectively. We also consider the tour-version of single-vehicle scheduling problem on a cycle, and give a $\frac{5}{3}$-approximation algorithm. Our algorithms improve the previous best results in the literature. The main idea used in the improved algorithms is to utilize the structure characteristics of tree and cycle to antedate the department time of the vehicle.

There are some issues that are still unsolved. We would like to know whether the approximation bounds obtained in this paper are tight. As a natural extension of this paper, researchers may study SVSP on a general network. If the general network satisfies the triangle inequality, it is straightforward to design a $\frac{5}{2}$-approximation algorithm. A better approximation bound is desirable.

Acknowledgement. The authors would like to thank the associated editor and the anonymous referees for their constructive comments and kind suggestions. This research was supported by the National Natural Science Foundation of China under Grant No. 11371137.

References

1. Tsitsiklis, J.N.: Special cases of traveling salesman and repairman problems with time windows. Networks **22**, 263–282 (1992). https://doi.org/10.1002/net.3230220305
2. Psaraftis, H.N., Solomon, M.M., Magnanti, T.L., Kim, T.-U.: Routing and scheduling on a shoreline with release times. Manage. Sci. **36**, 212–223 (1990). https://doi.org/10.1287/mnsc.36.2.212
3. Karuno, Y., Nagamochi, H., Ibaraki, T.: Better approximation ratios for the single-vehicle scheduling problems on line-shaped networks. Networks **39**(4), 203–209 (2002). https://doi.org/10.1002/net.10028
4. Gaur, D.R., Gupta, A., Krishnamurti, R.: A $\frac{5}{3}$-approximation algorithm for scheduling vehicles on a path with release and handling times. Inform. Process. Lett. **86**, 87–91 (2003). https://doi.org/10.1016/S0020-0190(02)00474-X
5. Bhattacharya, B., Carmi, P., Hu, Y., Shi, Q.: Single vehicle scheduling problems on Path/Tree/Cycle Networks with release and handling times. In: Hong, S.-H., Nagamochi, H., Fukunaga, T. (eds.) ISAAC 2008. LNCS, vol. 5369, pp. 800–811. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-92182-0_70
6. Yu, W., Liu, Z.: Single vehicle scheduling problems with release and service times on a line. Networks **57**, 128–134 (2011). https://doi.org/10.1002/net.20393
7. Karuno, Y., Nagamochi, H.: 2-approximation algorithms for the multi-vehicle scheduling prbolem on a path with release and handling times. Discrete Appl. Math. **129**, 433–447 (2003). https://doi.org/10.1016/S0166-218X(02)00596-6
8. Yu, W., Liu, Z.: Vehicle routing problems on a line-shaped network with release time constraints. Oper. Res. Lett. **37**, 85–88 (2009). https://doi.org/10.1016/j.orl.2008.10.006
9. Karuno, Y., Nagamochi, H., Ibaraki, T.: Vehicle scheduling on a tree with release and handling times. In: Ng, K.W., Raghavan, P., Balasubramanian, N.V., Chin, F.Y.L. (eds.) ISAAC 1993. LNCS, vol. 762, pp. 486–495. Springer, Heidelberg (1993). https://doi.org/10.1007/3-540-57568-5_280

10. Nagamochi, H., Mochizuki, K., Ibaraki, T.: Complexity of the single vehicle scheduling problem on graphs. Inform. Syst. Oper. Res. **35**, 256–276 (1997). https://doi.org/10.1080/03155986.1997.11732334
11. Bao, X., Liu, Z.: Approximation algorithms for single vehicle scheduling problems with release and service times on a tree or cycle. Theoret. Comput. Sci. **434**, 1–10 (2012). https://doi.org/10.1016/j.tcs.2012.01.046
12. Augustine, J.E., Seiden, S.S.: Linear time approximation schemes for vehicle scheduling problems. Theoret. Comput. Sci. **324**, 147–160 (2004). https://doi.org/10.1016/j.tcs.2004.05.013
13. Bhattacharya, B., Hu, Y.: Approximation algorithms for the multi-vehicle scheduling problem. In: Cheong, O., Chwa, K.-Y., Park, K. (eds.) ISAAC 2010. LNCS, vol. 6507, pp. 192–205. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-17514-5_17