Approximation Algorithms for Min-max Capacitated Path Covers*

Zhou Xu **Liang Xu**†

Department of Logistics and Maritime Studies,
Faculty of Business, The Hong Kong Polytechnic University
Email: {lgtzx,08900198r}@polyu.edu.hk

Abstract

This paper presents the first approximation algorithms and the first inapproximability results for min-max path cover problems, where a capacity constraint restricts the number of customers that can be serviced by every trip of the paths in the cover. Depending on different applications, every path in the cover may either be restricted to contain only one trip, or be allowed to contain multiple trips but with a return to the depot between every two consecutive trips. We develop a 5-approximation algorithm for the problem with multiple trips allowed, and a $(7+\epsilon)$-approximation algorithm for any $\epsilon > 0$ for the problem with single trips only. For both problems, we show that unless $\mathbf{NP} = \mathbf{P}$, it is impossible to achieve any performance ratios less than 3/2.

Keywords: approximation algorithm, capacity constraint, min-max, path cover, single/multiple depots

1 Introduction

Due to several large-scale emergencies that have occurred recently, routing for relief efforts has drawn great attention. In this case, the latest service completion time becomes critical. Such a new strategic goal on the routing of vehicles can be formulated into a min-max path cover problem. Depending on different applications, vehicles may either be allowed to take multiple trips with replenishment at the depot between every two consecutive trips [4], or be restricted to have one trip only [5], with each trip restricted to serve a limited number of customers due to the capacity of vehicles.

This motivates the following two problems to be studied in this paper, the Min-max Capacitated Path Cover Problem with Multiple Trips (CPCPMT), and the Min-max Capacitated Path Cover Problem with Single Trips (CPCPST). The problem instance of them can be represented by a tuple (G, r, J, w, h, k, Q), where $G = (V, E)$ is a complete undirected graph, and $r \in V$ denotes the depot of vehicles, and $J = V \setminus \{r\}$ represents a set of customers. The function w forms a metric, giving a non-negative edge weight to each edge in E, to indicate the edge traveling time. The function h gives a non-negative vertex weight $h(v)$ for each vertex $v \in V$, to represent the customer service handling time, where $h(r)$ is defined to be zero without loss of generality. Integers k, and Q represent the number of identical vehicles, and the capacity of each vehicle, respectively, such that every trip of the k vehicles can only service at most Q customers.

For any subgraph H of G, we use $V(H)$, $J(H)$, and $E(H)$ to represent the vertex set, the customer set, and the edge set of H, respectively, and use $w(H)$, and $h(H)$ to denote the total edge weight, and the total vertex weight of H, respectively. Thus, for any set of paths, denoted by $\mathcal{P}$, its latest service completion time can be represented by the maximum total edge and vertex weights of any path in $\mathcal{P}$, which are defined as the cost of $\mathcal{P}$, and denoted by:

$$cost(\mathcal{P}) = \max_{P \in \mathcal{P}} \{w(P) + h(P)\}.$$

Given any instance $\mathcal{I} = (G, r, J, w, h, k, Q)$, both the min-max CPCPMT and the min-max CPCPST are to minimize $cost(\mathcal{P})$ of a set $\mathcal{P}$ of k paths, denoted by $\mathcal{P} = \{P_1, P_2, ..., P_k\}$, such that all the customers in J are serviced by $\mathcal{P}$, while each path in $\mathcal{P}$ starts from the depot r, and contains trips that service at most Q customers each. In the min-max CPCPST, each path in $\mathcal{P}$ contains only one trip. Therefore, we assume that $kQ \geq |J|$ in order to guarantee that the instance has at least one feasible solution. In the min-max CPCPMT, however, each path can contain multiple trips, but needs to return to the depot r between every two consecutive trips.

Throughout this paper, we use $\mathcal{P}^* = \{P_1^*, \ldots, P_k^*\}$ to denote an optimum solution to $\mathcal{I}$, and use $opt(\mathcal{I})$ to denote $cost(\mathcal{P}^*)$, where $\mathcal{I}$ is any instance of the min-max CPCPMT or the min-max CPCPST.

Related work. In vehicle routing literature, most of the work studied problems to determine tours (rather than paths), and to minimize the total traveling time (rather than to minimize the latest service completion time) [10, 6], for which heuristic algorithms were developed but with no guarantees of any constant performance ratios [14, 3, 4]. Altinkemer [1], however, developed an approximation algorithm that achieves a performance ratio of 5/2 for a problem aiming to minimize the total traveling time of a tour for a single vehicle with multiple trips allowed.

The most related work in literature is Campbell and Venenbussche [5], which studied some special cases of the min-max CPCPST, but ignored the vertex weights. The focus of their work is on evaluating the worst case performance, under the min-max objective, for solutions that minimizes the total traveling time. From their results, one can derive a 2-approximation algorithm, and a 4-approximation algorithm, respectively, for special cases with $k = 1$ and $Q = \infty$, and with $Q = \infty$, respectively.

Other related literature studied the min-max uncapacitated tour (or tree) cover problems, where feasible solutions are a set of k tours (or trees) [2, 13, 8].

*This research was supported by PolyU Grant A-SA24.

†Corresponding author

For example, Frederickson [9] proposed a $(5/2-1/k)$-approximation algorithm for a k-Traveling Salesmen Problem (k-TSP), to determine a set of tours to service all customers under the min-max objective, which is equivalent to a min-max tour cover problem.

As far as we know, no approximation algorithms with constant performance ratios are known for the min-max CPCPMT and the min-max CPCPST. Moreover, inapproximability results are unknown for the two problems, and even for their related problems in literature.

Our results. Our main results and their significance are the following:

1. We have developed a 5-approximation algorithm for the min-max CPCPMT, and a $(7+\epsilon)$-approximation algorithm for any $\epsilon > 0$ for the min-max CPCPST, which are their first constant ratio approximation algorithms. (See Section 4 and Section 5.)
2. We have derived the first inapproximability results for the min-max CPCPMT and the min-max CPCPST, by showing that it is impossible to achieve any performance ratios less than 3/2 in polynomial time, unless **NP** = **P**. (See Section 2.)
3. Based on our work on the min-max CPCPST and the min-max CPCPMT, we jave developed approximation algorithms and inapproximability results for the extensions in which vehicles start their trips from multiple depots. (See Section 6.)

Algorithms, developed in this paper, are based on a novel use of a tour splitting procedure in Section 2, which extends the work of Frederickson [9] by taking vertex weight into consideration.

2 Inapproximability Results

The following theorem states an inapproximability bound of 3/2 for the min-max CPCPST:

Theorem 1. *Unless* **NP** = **P**, *there is no polynomial-time* $(3/2-\epsilon)$*-approximation algorithm for the min-max CPCPST, for any* $\epsilon > 0$*, even if* $h(v) = 0$ *for all* $v \in V$.

Proof. Suppose there exists a $(3/2-\epsilon)$-approximation algorithm for the min-max CPCPST with $\epsilon > 0$. We are going to show that this algorithm can be used to solve the 3DM in polynomial time, defined as follows, unless **NP** $\neq$ **P**.

Given any 3DM instance, consider a CPCPST instance $\mathcal{I} = (G, r, J, w, h, k, Q)$, which is defined as follows. Let $G = (V, E)$ be a complete undirected graph on V, where $V = \{r\} \cup W \cup X \cup Y \cup \bigcup_{i=1}^{m} \{q_{i,j} : 1 \le j \le 6\}$, and r is the depot, and $J = V \setminus \{r\}$ is the customer set. Let $\widetilde{E} = \bigcup_{i=1}^{m} \widetilde{E}_i$, where for each element $M_i = (x, y, z) \in \mathcal{M}$, the edge subset $\widetilde{E}_i$ consists of the 16 edges shown in Figure 1. For each $\widetilde{E}_i$, set $w(q_{i,1}, q_{i,2}) = w(q_{i,4}, q_{i,5}) = w(q_{i,3}, q_{i,6}) = 0$, and set the edge weight of remainders in $\widetilde{E}_i$ to be 1. For each edge $(u, v) \in E \setminus \widetilde{E}$, define $w(u, v)$ as the total edge weight of the shortest path from u to v in the subgraph $(V, \widetilde{E})$. It can be verified that w forms a metric. Moreover, we set each vertex weight to be 0, and let $k = 2m + n$ and $Q = 3$.

Define $\mathcal{F}(4, 2)$ as the set of paths P of G such that $|V(P)| = 4$ and $w(P) \le 2$, and that P starts from the depot r. Consider each path $P \in \mathcal{F}(4, 2)$. We can denote P by $(rv_1v_2v_3)$, where v_j for $1 \le j \le 3$ are three different customers in J. Notice $|E(P)| = 3$, and $w(r, v) \ge 1$ for any $v \in J$. To satisfy $w(P) \le 2$, $w(r, v_1)$ must be 1, because as shown in Figure 1, G has no adjacent edges with edge weight both equal to 0. For each $1 \le i \le m$, consider the following eight paths:

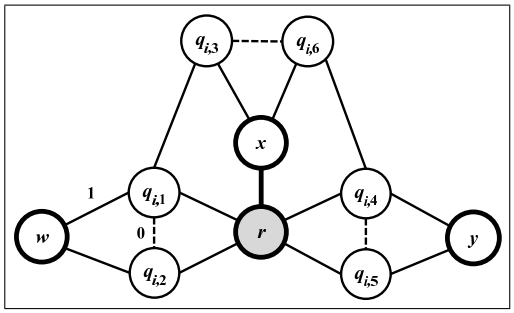

Figure 1: A component for each tuple $M_i = (w, x, y)$ used in transforming any 3DM instance to a min-max CPCPST instance with $h(v) = 0$ for all $v \in V$.

$$P_{i,1} = (rq_{i,1}q_{i,2}w), \quad P_{i,2} = (rq_{i,2}q_{i,1}w),$$
$$P_{i,3} = (rxq_{i,3}q_{i,6}), \quad P_{i,4} = (rxq_{i,6}q_{i,3}),$$
$$P_{i,5} = (rq_{i,4}q_{i,5}y), \quad P_{i,6} = (rq_{i,5}q_{i,4}y),$$
$$P_{i,7} = (rq_{i,1}q_{i,2}q_{i,3}), \quad P_{i,8} = (rq_{i,4}q_{i,5}q_{i,6}).$$

It can be verified $\{P_{i,j} : 1 \le i \le m, 1 \le j \le 8\}$ is a subset of $\mathcal{F}(4, 2)$. Moreover, $\{P_{i,1}, P_{i,2} : 1 \le i \le m\}$ consists of all paths in $\mathcal{F}(4, 2)$ that contain w, $\{P_{i,3}, P_{i,4} : 1 \le i \le m\}$ consists of all paths in $\mathcal{F}(4, 2)$ that contain x, and $\{P_{i,5}, P_{i,6} : 1 \le i \le m\}$ consists of all paths in $\mathcal{F}(4, 2)$ that contain y.

We are now going to show that the 3DM instance has an exact matching if and only if the $(3/2-\epsilon)$-approximation algorithm for the min-max CPCPST returns a feasible solution to $\mathcal{I}$ with cost at most 2.

On one hand, if the 3DM instance has an exact matching $\mathcal{M}'$, we can construct $\mathcal{P}$ by including paths $P_{i,2j}$ for $1 \le j \le 3$, for all $M_i \in \mathcal{M}'$, and including paths $P_{i,j}$ for $7 \le j \le 8$, for all $M_i \in \mathcal{M} \setminus \mathcal{M}'$. Thus, $|\mathcal{P}| = 3|\mathcal{M}'| + 2(|\mathcal{M}| - |\mathcal{M}'|) = k$.

It can be verified that $\mathcal{P}$ covers all vertices in V, and $cost(\mathcal{P}) = 2$. Thus the $(3/2-\epsilon)$-approximation algorithm must return a feasible solution to $\mathcal{I}$ with cost at most 2.

On the other hand, if the $(3/2-\epsilon)$-approximation algorithm returns a feasible solution $\mathcal{P}$, to the min-max CPCPST instance $\mathcal{I}$, with $cost(\mathcal{P}) \le 2$, then we construct $\mathcal{M}'$ by including all elements $M_i \in \mathcal{M}$ that have $P_{i,3} \in \mathcal{P}$ or $P_{i,4} \in \mathcal{P}$, where $1 \le i \le m$.

To prove that $\mathcal{M}'$ is an exact matching for the 3DM instance, we can first verify that each path $P \in \mathcal{P}$ contains exactly 3 customer vertices. Thus, $\mathcal{P} \subseteq \mathcal{F}(4, 2)$, and no two different paths in $\mathcal{P}$ can share the same vertex in J. Furthermore, we can verify that $|M'| = n$ and no two elements of M' agree in any coordinate by contradiction.

Therefore, M' is an exact matching, which completes the proof. □

Based on the proof of Theorem 1, we can also obtain the following inapproximability result for the min-max CPCPMT.

Theorem 2. *Unless* **NP** = **P**, *there is no polynomial-time* $(3/2-\epsilon)$*-approximation algorithm for the min-max CPCPMT, for any* $\epsilon > 0$*, even if* $h(v) = 0$ *for all* $v \in V$.

3 Tour Splitting

In order to develop approximation algorithms for the min-max CPCPST and the min-max CPCPMT, we devise a tour splitting procedure in Algorithm 1. Given a tour C, we first revise the edge weight of each $(u, v) \in E(C)$ in Step 1 as

$$w'(u, v) = w(u, v) + h(u) + h(v)$$

to include the vertex weight approximately. For each segment S of C, let $w'(S)$ denote the total revised edge weight of S. Algorithm 1 then splits the tour into segments in Step 2, such that the total revised edge weight of each segment are bounded by a given positive number B.

Algorithm 1 (Tour Splitting Procedure)**.**
Input: a tour C in which no customer in J appears more than once, a positive number B, and a start point s.
Output: a set $\mathcal{S}$ of segments of C.

1. For each edge $(u, v) \in E(C)$, revise the edge weight as $w'(u, v) = w(u, v) + h(u) + h(v)$.
2. From the start point s of C, relabel all the vertices along C in a clockwise direction by v_1, v_2, ..., and $v_{|V(C)|}$, in which $v_1 = v_{|V(C)|} = s$. Set $\pi(0) = 0$ and $t = 0$.
3. Repeat the following, until the revised total edge weight of the segment $(v_{\pi(t)+1} v_{\pi(t)+2} \dots v_{|V(C)|})$ of C is less than or equal to B.
   (a) Increase t by 1.
   (b) Let $v_{\pi(t)}$ denote the last vertex along C in a clockwise direction from $v_{\pi(t-1)+1}$, such that $w'(v_{\pi(t-1)+1} v_{\pi(t-1)+2} \dots v_{\pi(t)}) \le B$.
4. Set $\pi(t) = |V(C)|$, and so $v_{\pi(t)} = v_{|V(C)|}$. Split C into t segments by setting $S_i = (v_{\pi(i-1)+1} v_{\pi(i-1)+2} \dots v_{\pi(i)})$, for $1 \le i \le t$. Return $\mathcal{S} = \{S_i, 1 \le i \le t\}$.

Algorithm 1 runs in polynomial time, and holds the following two properties.

Firstly, notice that no customer in J appears in C more than once, and $h(r) = 0$. By the definition of w', as shown in Step 1 of Algorithm 1, we know:

$$w'(C) = w(C) + 2h(C). \tag{1}$$

Secondly, for the segment set $\mathcal{S}$ returned by Algorithm 1, its cost (under w and h), denoted by $cost(\mathcal{S}) = \max_{1 \le i \le |\mathcal{S}|} \{w(S_i) + h(S_i)\}$, is bounded by B, due to the following Lemma 1.

Lemma 1. *Algorithm 1 splits C into a set $\mathcal{S}$ of segments, such that:*

1. *$cost(\mathcal{S}) \le B$, and*
2. *$|\mathcal{S}| \le \lceil [w(C) + 2h(C)]/B \rceil$.*

Proof. Firstly, according to Step 3 of Algorithm 1, $w'(S_i) \le B$ must be satisfied for $1 \le i \le |\mathcal{S}|$. Notice that $S_i = (v_{\pi(i-1)+1} v_{\pi(i-1)+2} \dots v_{\pi(i)})$, which implies $w'(S_i) = w(S_i) + 2h(S_i) - h(v_{\pi(i-1)+1}) - h(v_{\pi(i)})$. Since $h(S_i) \ge h(v_{\pi(i-1)+1}) + h(v_{\pi(i)})$, we obtain $w(S_i) + h(S_i) \le w'(S_i)$. Thus, $cost(\mathcal{S}) \le B$.

Secondly, according to Step 3(b) of Algorithm 1, $B < w'(S_i) + w'(v_{\pi(i)+1}, v_{\pi(i)})$ must be satisfied for $1 \le i \le |\mathcal{S}| - 1$. Hence, $(|\mathcal{S}| - 1)B < w'(C)$. By (1), since $|\mathcal{S}|$ is an integer, we obtain $|\mathcal{S}| \le \lceil [w(C) + 2h(C)]/B \rceil$. □

4 Approximation Algorithm for the Min-max CPCPMT

In this section, we present a 5-approximation algorithm for the min-max CPCPMT, in which every path of the vehicles is allowed to contain multiple trips, but needs to return to the depot between every two consecutive trips.

Algorithm 2 (CPCPMT)**.**
Input: an instance $\mathcal{I} = (G, r, J, w, h, k, Q)$.
Output: a set $\mathcal{P}$ of k paths.

1. (a) Find a minimum spanning tree T^* of G. Obtain a tour C' that consists of all customers in J, by doubling each edge of T^*, finding an Eulerian cycle, and short-cutting the depot r and any duplications of vertices in the cycle. Set the start point s as r. Relabel the customer vertices of C' in a clockwise direction by v_1, v_2, ..., and $v_{|J|}$.
   (b) For each iteration i, where $i = 1, 2, ..., Q$, split the portion of the tour from v_{i+1} to $v_{\lfloor (|J|-i)/Q \rfloor Q+i}$ into segments that consist of Q customers each. It may form two additional segments, i.e., the first segment that contains vertices v_1 to v_i, and the last segment that may contain vertices $v_{\lfloor (|J|-i)/Q \rfloor Q+i+1}$ to $v_{|J|}$. Connect the two endpoints of each segment to the depot r. This leads to a new tour, denoted by C_i, which contains multiple trips with each trip starting from the depot r and consisting of at most Q customers.
   (c) Among all the tours C_i obtained in Step 1(b) for $1 \le i \le Q$, let C denote one with the minimum total edge and vertex weight.
2. Apply the Algorithm 1 on the tour C and a bound B, to split C into a set $\mathcal{S}$ of segments, where B is set as follows.
$$B = 2|J|\bar{d}/(Qk) + 2[w(T^*) + h(G)]/k, \tag{2}$$
$$\bar{d} = \sum_{v \in J} w(r, v)/|J|. \tag{3}$$
3. For each segment in $\mathcal{S}$, connect one of its endpoints to the depot r, which forms a set $\mathcal{P}$ of paths. If $|\mathcal{S}| < k$, then add $k - |\mathcal{S}|$ empty paths to $\mathcal{P}$.
4. Return $\mathcal{P}$.

To prove the correctness of Algorithm 2, we present a following upper bound on the total edge and vertex weight of the tour C obtained by Step 1.

Lemma 2. *In Algorithm 2, the tour C obtained by Step* 1 *satisfies:*

$$w(C) + h(C) \le 2|J|\bar{d}/Q + 2w(T^*) + h(G). \tag{4}$$

Proof. From Step 1(a) of Algorithm 2, we know $w(C') \le 2w(T^*)$ by the triangle inequality of w. When Step 1(b) constructs the Q tours C_i for $1 \le i \le Q$, each customer $v \in J$ has been connected to the depot r for at most twice, and each edge of C has appeared in the Q tours at most Q times. Thus, by summing up all the total edge and vertex weight of the Q tours, and by the definition of $\bar{d}$ in (3), we obtain:

$$\sum_{i=1}^{Q} [w(C_i) + h(C_i)] \le 2|J|\bar{d} + 2Qw(T^*) + Qh(G).$$

Since $w(C) + h(C) \leq w(C_i) + h(C_i)$ for $1 \leq i \leq Q$, we have:

$$Q[w(C) + h(C)] \leq 2|J|\bar{d} + 2Qw(T^*) + Qh(G),$$

which leads to (4). □

Based on Lemma 2, we can prove the correctness of Algorithm 2 as follows.

Lemma 3. *For any min-max CPCPMT instance $\mathcal{I}$, Algorithm 2 returns a feasible solution to $\mathcal{I}$ in polynomial time.*

Proof. The polynomial time complexity is easy to be verified. We prove as follows that the path set $\mathcal{P}$ returned by Algorithm 2 is a feasible solution to $\mathcal{I}$.

From Lemma 1, we have $|\mathcal{S}| \leq \lceil [w(C) + 2h(C)]/B \rceil$. Thus, according to the value of B chosen in (2), and by Lemma 2 and $h(C) \leq h(G)$, we obtain $|\mathcal{S}| \leq k$, which implies $|\mathcal{P}| = k$ due to Step 3. Due to Step 3 of Algorithm 2, each path in $\mathcal{P}$ starts from the depot r. Notice that the tour C obtained in Step 1 of Algorithm 2 covers all the customers in J, and so does $\mathcal{P}$, because paths in $\mathcal{P}$ contains all segments split from C in Step 2. Moreover, notice that C contains multiple trips with each trip containing at most Q customers, and so does each segment split from C. This implies that every trip in paths in $\mathcal{P}$ contains at most Q customers. Thus, $\mathcal{P}$ satisfies the capacity constraint, and is a feasible solution to the min-max CPCPMT instance $\mathcal{I}$. □

To analyze the performance ratio of Algorithm 2, we derive as follows a lower bound on $opt(\mathcal{I})$ for any given instance $\mathcal{I}$.

Lemma 4. *For any min-max CPCPMT instance $\mathcal{I}$, $opt(\mathcal{I})$ satisfies*

$$opt(\mathcal{I}) \geq \max\{\frac{|J|\bar{d}}{Qk}, \frac{w(T^*) + h(G)}{k}\}. \quad (5)$$

Proof. Consider an optimal solution $\mathcal{P}^* = \{P_1^*, \ldots, P_k^*\}$ to $\mathcal{I}$. For each path P_i^*, where $1 \leq i \leq k$, let $y(i)$ denote the number of its trips. For its j-th trip, where $1 \leq j \leq y(i)$, let v_j^* denote the customer with largest edge weight from the depot r in the trip. Thus, we have

$$\sum_{j=1}^{y(i)} w(r, v_j^*) \leq w(P_i^*) + h(P_i^*), \text{ for } 1 \leq i \leq k.$$

Since each trip can service at most Q customers,

$$[\sum_{v \in J(P_i^*)} w(r, v)]/Q \leq \sum_{j=1}^{y(i)} w(r, v_j^*), \text{ for } 1 \leq i \leq k.$$

Thus, noticing $\sum_{i=1}^{k} \sum_{v \in J(P_i^*)} w(r, v) = |J|\bar{d}$ by (3), we obtain

$$\begin{aligned} |J|\bar{d}/Q &\leq \sum_{i=1}^{k} \sum_{j=1}^{y(i)} w(r, v_j^*) \leq \sum_{i=1}^{k} [w(P_i^*) + h(P_i^*)] \\ &\leq k \cdot opt(\mathcal{I}). \end{aligned} \quad (6)$$

Moreover, notice that $\bigcup P_i^*$ spans all the vertices of G, implying

$$w(T^*) + h(G) \leq \sum_{i=1}^{k} [w(P_i^*) + h(P_i^*)] \leq k \cdot opt(\mathcal{I}).$$

Together with (6), this leads to (5). □

Hence, we can establish the performance ratio of Algorithm 2.

Theorem 3. *Algorithm 2 achieves a performance ratio of 5 in polynomial time for the min-max CPCPMT.*

Proof. Consider any min-max CPCPMT instance $\mathcal{I}$. By Lemma 3, Algorithm 2 returns a feasible solution $\mathcal{P}$ to $\mathcal{I}$ in polynomial time. For the segment set $\mathcal{S}$ constructed in Step 2 of Algorithm 2, from Lemma 1 and (2), we know $cost(\mathcal{S}) \leq 2|J|\bar{d}/(Qk) + 2[w(T^*) + h(G)]/k$. Thus, according to Lemma 4, we have $cost(\mathcal{S}) \leq 4opt(\mathcal{I})$. Notice that each path in $\mathcal{P}$ is constructed by connecting a segment in $\mathcal{S}$ to r, in Step 3 of Algorithm 2. Due to the triangle inequality of w, $w(r, v) \leq opt(\mathcal{I})$ for each vertex $v \in V$, which implies $cost(\mathcal{P}) \leq 5opt(\mathcal{I})$. Thus, the performance ratio of Algorithm 2 is 5. □

5 Approximation Algorithm for the Min-max CPCPST

In this section, we present a $(7 + \epsilon)$-approximation algorithm for any $\epsilon > 0$ for the min-max CPCPST, in which every path of the vehicles is forced to contain only one trip.

The approximation algorithm relies on Algorithm 3. Given any $\lambda > 0$, and any instance $\mathcal{I}$ of the min-max CPCPST, Algorithm 3 either returns "λ is too small" (implying $\lambda < opt(\mathcal{I})$), or returns a feasible solution to $\mathcal{I}$ with cost at most 7λ, in polynomial time, (as shown in Lemma 5). Thus, since $opt(\mathcal{I})$ is bounded by the interval $[\max_{v \in J} w(r, v), 2w(G) + h(G)]$, we can apply a binary search to obtain a close value λ, such that Algorithm 3 will return a feasible solution with cost at most 7λ, and that for any $\epsilon > 0$, Algorithm 3 will return and guarantee that $(\lambda - \epsilon)$ is too low. Hence, a polynomial time $(7 + \epsilon)$-approximation algorithm can be obtained.

Algorithm 3 (CPCPST)**.**
Input: an instance $I = (G, r, J, w, h, k, Q)$, and $\lambda > 0$.
Output: "λ is too small", or a set $\mathcal{P}$ of paths.

1. If $\max_{v \in J} w(r, v) > \lambda$, return "$\lambda$ is too small".
2. Find a minimum spanning tree T^* of G. Obtain a tour C that consists of all customers in J, by doubling each edge of T^*, finding an Eulerian cycle, and short-cutting the depot r and any duplications of vertices in the cycle.
3. Apply Algorithm 1 on the tour C and a bound B, to split the tour C into a set $\mathcal{S}$ of segments, where $B = 2\lambda$. By Lemma 1, each segment in $cost(\mathcal{S}) \leq 2\lambda$. If $|\mathcal{S}| > k$, return "λ is too small". Otherwise, if $|\mathcal{S}| < k$, then add $(k - |\mathcal{S}|)$ empty segments to $\mathcal{S}$.
4. Set the path set $\mathcal{P} = \emptyset$. For each iteration i, where $i = 1, 2, ..., k - 1$,
   (a) Find S_{min}, and S_{max}, which denote the segments in $\mathcal{S}$ that contain the least, and the most number of customers, respectively.
   (b) If $|V(S_{max})| \leq Q$, then, stop the iteration, and go to Step 5.
   (c) Otherwise, $|V(S_{max})| > Q$, and then, from one of its endpoints, relabel the vertices along S_{max} by v_1, v_2, ..., $v_{|V(S_{max})|}$. Notice that $|V(S_{min})| < Q$, (as shown in the proof of Lemma 5.) Let $j = Q - |V(S_{min})|$, which satisfies $1 \leq j \leq |V(S_{max})|$. Connect one endpoint of S_{min} to the segment $(v_1 \ldots v_j)$ of S_{max} through v_1, and then, connect the

other endpoint of S_{min} to the depot r, which forms a path P_i. Add P_i to the path set $\mathcal{P}$. Replace S_{max} in $\mathcal{S}$ with its remaining segment $(v_{j+1}v_{j+2}\dots v_{|V(S_{max})|})$. Remove S_{min} from $\mathcal{S}$.

5. For each segment remaining in $\mathcal{S}$, connect one of its endpoints to the depot r, to form a path, which is then added to the path set $\mathcal{P}$.

6. Return $\mathcal{P}$.

The correctness of Algorithm 3 is shown in Lemma 5 as follows.

Lemma 5. *Given any min-max CPCPST instance $\mathcal{I}$, and given any $\lambda > 0$, if Algorithm 3 returns "λ is too small", then $\lambda < opt(\mathcal{I})$; otherwise, Algorithm 3 returns a feasible solution $\mathcal{P}$ to $\mathcal{I}$ with $cost(\mathcal{P}) \leq 7\lambda$.*

Proof. If Algorithm 3 returns "λ is too small" in Step 1, then there exists at least one vertex $v \in J$ with $w(r, v) > \lambda$. Due to the triangle inequality of w, we have $w(r, v) \leq opt(\mathcal{I})$, which implies $\lambda < opt(\mathcal{I})$

If Algorithm 3 returns "λ is too small" in Step 3, then by Lemma 1, $k < \lceil [w(C) + 2h(G)]/(2\lambda) \rceil$. According to Step 2 of Algorithm 3, and by the triangle inequality of w, we have $w(C) \leq 2w(T^*)$, which implies $k < \lceil [w(T^*) + h(G)]/\lambda \rceil$. To show $\lambda < opt(\mathcal{I})$, consider an optimal solution $\mathcal{P}^* = \{P_i^* : 1 \leq i \leq k\}$ to $\mathcal{I}$. Since $\bigcup_{i=1}^{k} P_i^*$ spans all the vertices of G, we have $w(T^*) + h(G) \leq \sum_{i=1}^{k} [w(P_i^*) + h(P_i^*)]$, which is less than or equal to $k \cdot opt(\mathcal{I})$. Thus, we obtain $\lambda < opt(\mathcal{I})$.

Otherwise, Algorithm 3 returns a path set $\mathcal{P}$. According to Steps 3 and 4, we can see that for each iteration in Step 4, it always satisfies that $|\mathcal{P}| + |\mathcal{S}| = k$, and that each path in $\mathcal{P}$ contains exact Q customers. Thus, in Step 3(c), where $|V(S_{max})| > Q$, it must hold that $|V(S_{min})| < Q$, because $kQ \geq |J|$.

Let us now verify the capacity constraint for $\mathcal{P}$. When Step 4 of Algorithm 3 stops its iterations, it can be seen that each path P_i in $\mathcal{P}$ satisfies $|V(P_i)| = Q$. Furthermore, we can show that every segment remained in $\mathcal{S}$ contains at most Q customers.

Moreover, by Step 5, each path in $\mathcal{P}$ must start from the depot r. By Step 2, $\mathcal{P}$ must service all the customers in J. Notice that in each iteration in Step 4, it always satisfies that $|\mathcal{P}| + |\mathcal{S}| = k$. Thus, the resulting path set $\mathcal{P}$ after Step 5 satisfies $|\mathcal{P}| \leq k$. Therefore, $\mathcal{P}$ is a feasible solution to $\mathcal{I}$.

Finally, we are going to prove that the cost of $\mathcal{P}$ is at most 7λ. By Lemma 1, the cost of $\mathcal{S}$, constructed in Step 3, is not greater than 2λ. Thus, in each iteration of Step 4, we have total edge and vertex weight of any portion of S_{max} is not greater than 2λ. Since weight of the edge that connects S_{min} and $(v_1 ... v_j)$ in Step 4(c) is not greater than 2λ, and weight of the edge that connects S_{min} to r in Step 4(c) is not greater than λ. Thus, for every path constructed in Step 4, its total edge and vertex weight are not greater than 7λ. Hence, $cost(\mathcal{P}) \leq 7\lambda$. The proof is completed. □

From Lemma 5, and according to the arguments in beginning of this section, we can directly obtain a $(7 + \epsilon)$-approximation algorithm, for any $\epsilon > 0$, by a binary search, which establishes the following theorem.

Theorem 4. *For any $\epsilon > 0$, there exists a polynomial time $(7+\epsilon)$-approximation algorithm for the min-max CPCPST.*

6 Extensions for Multiple Depots

In this section, we develop approximation algorithms and inapproximability results for the extensions of the min-max CPCPMT and the min-max CPCPST, in which vehicles can start their trips from multiple depots. Our study contributes to a growing body of literature on the multiple depot vehicle routing problems, such as [11, 7, 12], which, however, mainly focus on minimizing the total traveling time rather than on optimizing the min-max objective.

In the situation with multiple depots, a problem instance can be represented by $\mathcal{I} = (G, D, J, w, h, k, Q)$, where $G = (V, E)$ is a given complete undirected graph, $D \subseteq V$ denotes the set of depots, $J = V \setminus D$ denotes the customer set, and w, h, k, and Q are edge weight, vertex weight, the vehicle number, and the vehicle capacity. We can assume $h(d) = 0$ for each depot $d \in D$ without loss of generality.

Thus, both the min-max CPCPMT with multiple depots (CPCPMT-MD in short), and the min-max CPCPST with multiple depots (CPCPST-MD in short) are to minimize $cost(\mathcal{P})$ of a set $\mathcal{P}$ of k paths, denoted by $\mathcal{P} = \{P_1, P_2, ..., P_k\}$, such that all the customers in J are serviced by $\mathcal{P}$, while each path in $\mathcal{P}$ starts from a depot in D, and contains trips that service at most Q customers each. In the min-max CPCPST-MD, each path in $\mathcal{P}$ is restricted to contain only one trip. We thus assume $kQ \geq |J|$ to ensure the feasibility of the given instance. In the min-max CPCPMT-MD, each path is allowed to contain multiple trips, but between every two consecutive trips, the path needs to return to a depot in D, which can be different from the depot where the path begins.

6.1 Inapproximability Results for Multiple Depot Extensions

In the following, we prove the inapproximability results for the min-max CPCPST-MD, and the min-max CPCPMT-MD, respectively.

Theorem 5. *Unless* **NP** = **P**, *there is no polynomial-time $(2 - \epsilon)$-approximation algorithm for the min-max CPCPST-MD, for any $\epsilon > 0$, even if $h(v) = 0$ for all $v \in V$.*

Proof. Based on a similar reduction from 3DM to an instance of the min-max CPCPST-MD, we can verify this theorem.

□

Based on the proof of Theorem 5, we can further derive the following inapproximability result for the min-max CPCPMT-MD.

Theorem 6. *Unless* **NP** = **P**, *there is no polynomial-time $(2 - \epsilon)$-approximation algorithm for the min-max CPCPMT-MD, for any $\epsilon > 0$, even if $h(v) = 0$ for all $v \in V$.*

6.2 Approximation Algorithm for the Min-max CPCPMT-MD

In the following, we develop a $(7 + \epsilon)$-approximation algorithm for the min-max CPCPMT-MD, which relies on Algorithm 4. Given any $\lambda > 0$ and any instance $\mathcal{I}$ of the min-max CPCPMT-MD, Algorithm 4 either returns that "λ is too small" (implying $\lambda < opt(\mathcal{I})$) or returns a feasible solution $\mathcal{P}$ to $\mathcal{I}$ with cost at most 7λ.

Considering an induced graph $G(J)$, let $E(\lambda)$ denote the subset of edges in $G(J)$ with edge weight less than or equal to λ. Let $G_j(\lambda)$ for $1 \leq j \leq m(\lambda)$ denote each of the $m(\lambda)$ connected components of the

subgraph $(J, E(\lambda))$. Since D and J are disjoint, $G_j(\lambda)$ contains no depots in D.

Algorithm 4 (CPCPMT-MD)**.**
Input: Instance $\mathcal{I} = (G, D, J, w, h, k, Q)$ and $\lambda > 0$
Output: "λ is too small" or a path set $\mathcal{P}$.

1. If there exists $v \in J$ such that $\min_{d\in D} w(v, d) > \lambda$, then return "$\lambda$ is too small".
2. For each connected component $G_j(\lambda)$ of the subgraph $(J, E(\lambda))$, where $E(\lambda)$ denote the subset of edges in G with edge weight less than or equal to λ, and $1 \le j \le m(\lambda)$, do the followings:
   (a) Find a minimum spanning tree T_j^* of $G_j(\lambda)$, and construct a tour, denoted by C_j', in $G_j(\lambda)$ by doubling each edge of T_j^*, finding an Euclidian cycle, and short-cutting any duplications of vertices in the cycle.
   (b) For each iteration i, where $i = 1, 2, ..., Q$, do the followings:
      i. Follow Step 1(b) of Algorithm 2 to split the tour C_j', from v_{i+1} and in a clockwise direction, into segments with each containing at most Q customers.
      ii. For each two consecutive segments σ and σ', split from C_j' by Step 2(b).i, let (v, v') denote the edge in C_i' that joins the endpoints of σ and σ', and then connect both v and v' to the nearest depot of v in D. This leads to a new tour, denoted by $C_{j,i}$, that contains multiple trips with each trip starting from a depot in D, and consisting of at most Q customers.
   (c) Among all the tours $C_{j,i}$ obtained in Step 2(b), let C_j be the one with minimum total edge and vertex weight.
   (d) Apply Algorithm 1 on C_j and $B = 6\lambda$ to split C_j into a set $\mathcal{S}_j$ of segments.
3. If $\sum_{j=1}^{m(\lambda)} |\mathcal{S}_j| > k$, returns "$\lambda$ is too small".
4. For $1 \le j \le m(\lambda)$, connect an endpoint of each segment in $\mathcal{S}_j$ to its nearest depot in D, which forms a set $\mathcal{P}$ of paths. If $|\mathcal{S}| < k$, then add $k - |\mathcal{S}|$ empty paths to $\mathcal{P}$.
5. Return $\mathcal{P}$.

Consider any instance $\mathcal{I}$ of the min-max CPCPMT-MD. To prove the correctness of Algorithm 4, consider each connected component $G_j(\lambda)$ of the subgraph $(J, E(\lambda))$, where $1 \le j \le m(\lambda)$. Let $\mathcal{P}_j^*$ denote the subset of paths in the optimum path cover $\mathcal{P}^*$ with each path containing at least one vertex of $G_j(\lambda)$. Let k_j^* denote the number of paths in $\mathcal{P}_j^*$. We next derive as follows a lower bound on $opt(\mathcal{I})$.

Lemma 6. *For each connected component $G_j(\lambda)$ of the subgraph $(J, E(\lambda))$, where $1 \le j \le m(\lambda)$, define*

$$\bar{d}_j = \frac{\sum_{v\in J(G_j(\lambda))} \min_{d\in D} w(v, d)}{|J(G_j(\lambda))|}. \quad (7)$$

If $opt(\mathcal{I}) \le \lambda$, then

$$opt(\mathcal{I}) \ge \frac{|J(G_j(\lambda))|\bar{d}_j}{Qk_j^*}. \quad (8)$$

Proof. We can follow the proof of Lemma 4 for each $G_j(\lambda)$. More specifically, we can replace G with $G_j(\lambda)$, J with $J(G_j(\lambda))$, $\mathcal{P}^*$ with $\mathcal{P}_j^*$, $\bar{d}$ with $\bar{d}_j$ defined in (7), k with k_j^*, and $w(r, v)$ with $\min_{d\in D} w(d, v)$ in the proof of of Lemma 4, to obtain

$$|J(G_j(\lambda))|\bar{d}_j/Q \quad \le \quad k_j^* \cdot opt(\mathcal{I}).$$

which leads to (8) directly. □

Thus, the correctness of Algorithm 4 can be proved as follows.

Lemma 7. *Given any min-max CPCPMT-MD instance $\mathcal{I}$, and given any $\lambda > 0$, if Algorithm 4 returns "λ is too small", then $\lambda < opt(\mathcal{I})$; otherwise, Algorithm 4 returns a feasible solution $\mathcal{P}$ to $\mathcal{I}$ with $cost(\mathcal{P}) \le 7\lambda$.*

Proof. If Algorithm 4 returns "λ is too small" in Step 1, there exists at least a vertex $v \in J$ such that $\min_{d\in D} w(v, d) > \lambda$. Due to the triangle inequality of w, we can see $\lambda < opt(\mathcal{I})$.

Otherwise, if Algorithm 4 returns "λ is too small" in Step 3, then $\sum_{j=1}^{m(\lambda)} |\mathcal{S}_j| > k$. By contradiction, suppose $opt(\mathcal{I}) \le \lambda$. Consider each connected component $G_j(\lambda)$ in Step 2, where $1 \le j \le m(\lambda)$. Notice that by eliminating depots in each path in $\mathcal{P}_j^*$ by shortcuts, and adding at most $k_j^* - 1$ edges with weight not greater than λ, one can obtain a tree that spans all vertices in $G_j(\lambda)$, implying that in Step 2(a),

$$w(T_j^*) + h(T_j^*) \le \sum_{P\in\mathcal{P}_j^*} [w(P) + h(P)] + k_j^*\lambda \le 2k_j^*\lambda \quad (9)$$

In Step 2(b).ii, for each two consecutive segments σ and σ', which are split from C_j' in Step 2(b).i, consider the edge (v, v') in C_j' that joins σ and σ', and let $d(v)$ denote the nearest depot of v in D. By the triangle inequality, we have $w(d(v), v') + w(d(v), v) \le w(v, v') + 2w(d(v), v)$. Thus, according to Step 2(b), and noticing $w(C_j') \le 2w(T_j^*)$, by following an argument similarly to the proof of Lemma 4, we have:

$$w(C_j) + 2h(C_j) \le 2|J(G_j)|\bar{d}_j/Q + 2[w(T_j^*) + h(T_j^*)].$$

Thus, if $opt(\mathcal{I}) \le \lambda$, then by Lemma 6, equation (9), Lemma 1, and equation (1), we obtain

$$|\mathcal{S}_j| = \lceil \frac{2|J(G_j)|\bar{d}_j/Q + 2[w(T_j^*) + h(T_j^*)]}{6\lambda} \rceil \le k_j^*,$$

which implies $\sum_{i=1}^{m(\lambda)} |\mathcal{S}_j| \le \sum_{i=1}^{m} k_j^* = k$, leading to the contradiction. Hence, $\lambda < opt(\mathcal{I})$.

Otherwise, Algorithm 4 returns a set $\mathcal{P}$ of paths in Step 5. Similarly to the proof of Lemma 3, it can be verified that $\mathcal{P}$ is a feasible solution to the given instance $\mathcal{I}$ of the min-max CPCPMT-MD. Due to $B = 6\lambda$ and Lemma 1, we have $cost(\mathcal{S}_j) \le 6\lambda$ for each $\mathcal{S}_j$ obtained in Step 2(d). According to Step 1, each edge that joins a vertex to its nearest depot in D must have a weight not greater than λ in Step 4. Hence, $cost(\mathcal{P}) \le 7\lambda$, which completes the proof. □

Similarly to Section 5, from Lemma 7 and Algorithm 4, we can obtain a $(7+\epsilon)$-approximation for any $\epsilon > 0$, by binary search. Thus, the following theorem is established.

Theorem 7. *For any $\epsilon > 0$, there exists a polynomial time $(7+\epsilon)$-approximation algorithm for the min-max CPCPMT-MD.*

6.3 Approximation Algorithm for the Min-max CPCPST-MD

For the min-max CPCPST-MD, we develop as follows a bi-criteria approximation algorithm. For any $\epsilon > 0$, it achieves a bi-criteria performance ratio of $(5+\epsilon, 2)$, which, as a weaker notion of approximation guarantees, implies that the path cover returned has total edge and vertex weight not greater than $(5+\epsilon)opt(\mathcal{I})$, and has the total number of paths contained not greater than $2k$.

Algorithm 5 (CPCPST-MD)**.**
Input: Instance $\mathcal{I} = (G, D, J, w, h, k, Q)$ and $\lambda \geq 0$
Output: "λ is too small" or a path set $\mathcal{P}$.

1. If there exists $v \in J$ such that $\min_{d \in D} w(v, d) > \lambda$, then return "$\lambda$ is too small".
2. For each connected component $G_j(\lambda)$ of the subgraph $(J, E(\lambda))$, where $E(\lambda)$ denote the subset of edges in G with edge weight less than or equal to λ, and $1 \leq j \leq m(\lambda)$, do the following:
   (a) Find a minimum spanning tree T_j^* of $G_j(\lambda)$, and construct a tour, denoted by C_j', in $G_j(\lambda)$ by doubling each edge of T_j^*, finding an Euclidian cycle, and short-cutting any duplications of vertices in the cycle.
   (b) Revise Algorithm 1 to split C_j' into a set $\mathcal{S}_j$ of segments, where the Step 3(b) of Algorithm 1 is revised to find the last vertex $v_{\pi(t)}$ such that the segment $w'(v_{\pi(t-1)+1}, \ldots, v_{\pi(t)})$ has total revised edge weight less than or equal to $B = 4\lambda$, and contains at most Q customers, i.e., $w'(v_{\pi(t-1)+1}, \ldots, v_{\pi(t)}) \leq B$ and $|J(v_{\pi(t-1)+1}, \ldots, v_{\pi(t)})| \leq Q$.
3. If $\sum_{j=1}^{m(\lambda)} |\mathcal{S}_j| > 2k$, returns "$\lambda$ is too small".
4. For each segment in $\mathcal{S}_j$, and for $1 \leq j \leq m(\lambda)$, connect any endpoint of it to the nearest depot in D, which forms a set $\mathcal{P}$ of paths. Return $\mathcal{P}$.

By following an argument similarly to the proof of Lemma 7, we can prove Lemma 8 for the correctness of Algorithm 5.

Lemma 8. *If Algorithm 5 returns "λ is too small", then $\lambda < opt(\mathcal{I})$; otherwise Algorithm 5 returns a path cover $\mathcal{P}$ with $cost(\mathcal{P}) \leq 5\lambda$ and $|\mathcal{P}| \leq 2k$.*

Similarly to Section 5, from Lemma 8 and Algorithm 5, we can obtain a $(5+\epsilon, 2)$-approximation for any $\epsilon > 0$, by binary search. Thus, the following theorem is established for the min-max CPCPST-MD.

Theorem 8. *For any $\epsilon > 0$, there exists a polynomial time bi-criteria $(5+\epsilon, 2)$-approximation algorithm for the min-max CPCPST-MD.*

7 Concluding Remarks

In this paper, we have presented the first constant ratio approximation algorithms, for the min-max CPCPMT and the min-max CPCPST, achieving performance ratios of 5 and $(7+\epsilon)$, respectively. We have proved that both of the problems cannot be approximated with any performance ratios less than 3/2 unless $\mathbf{NP} = \mathbf{P}$. The analysis and techniques developed in this work establishes a basis for future research on other min-max vehicle routing problems, such as the min-max capacitated tour cover problem.

References

[1] K. Altinkemer and B. Gavish. Technical Note–Heuristics for Delivery Problems with Constant Error Guarantees. *Transportation Science*, 24(4):294–297, 1990.

[2] I. Berman and O. Averbakh. (p-1)/(p+1)-approximate algorithms for p-travelling salesmen problems on a tree with minmax objective. *Discrete Applied Mathematics*, 75:201–216, 1997.

[3] J. Brandao and A. Mercer. A tabu search algorithm for the multi-trip vehicle routing and scheduling problem. *European Journal of Operational Research*, 100(1):180–191, 1997.

[4] JCS Brandao and A. Mercer. The multi-trip vehicle routing problem. *Journal of the Operational research society*, pages 799–805, 1998.

[5] A. M. Campbell, D. Vandenbussche, and W. Hermann. Routing for relief efforts. *Transportation Science*, 42(2):127–145, 2008.

[6] M. Charikar, S. Khuller, and B. Raghavachari. Algorithms for capacitated vehicle routing. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 349–358. ACM New York, NY, USA, 1998.

[7] J.F. Cordeau, M. Gendreau, and G. Laporte. A tabu search heuristic for periodic and multi-depot vehicle routing problems. *Networks*, 30(2), 1997.

[8] G. Even, N. Garg, J. Könemann, R. Ravi, and A. Sinha. Minmax tree covers of graphs. *Operations Research Letters*, 32(4):309–315, 2004.

[9] G.N. Frederickson, M.S. Hecht, and C.E. Kim. Approximation algorithms for some routing problems. *SIAM Journal on Computing*, 7(2), 1978.

[10] M. Haimovich and A.H.G.R. Kan. Bounds and heuristics for capacitated routing problems. *Mathematics of Operations Research*, pages 527–542, 1985.

[11] A. Lim and F. Wang. Multi-depot vehicle routing problem: A one-stage approach. *IEEE transactions on Automation Science and Engineering*, 2(4):397–402, 2005.

[12] W. Malik, S. Rathinam, and S. Darbha. An approximation algorithm for a symmetric generalized multiple depot, multiple travelling salesman problem. *Operations Research Letters*, 35(6):747–753, 2007.

[13] H. Nagamochi and K. Okada. Approximating the minmax rooted-tree cover in a tree. *Information Processing Letters*, 104:173–178, 2007.

[14] RJ Petch and S. Salhi. A multi-phase constructive heuristic for the vehicle routing problem with multiple trips. *Discrete Applied Mathematics*, 133(1-3):69–92, 2003.