Approximation Algorithms for Mixed, Windy, and Capacitated Arc Routing Problems

René van Bevern[1], Christian Komusiewicz[2], and Manuel Sorge[2]

1 **Novosibirsk State University, Novosibirsk, Russia,** `rvb@nsu.ru`
2 **Institut für Softwaretechnik und Theoretische Informatik, TU Berlin, Germany**
`{christian.komusiewicz,manuel.sorge}@tu-berlin.de`

Abstract

We show that any $\alpha(n)$-approximation algorithm for the n-vertex metric asymmetric Traveling Salesperson problem yields $O(\alpha(C))$-approximation algorithms for various mixed, windy, and capacitated arc routing problems. Herein, C is the number of weakly-connected components in the subgraph induced by the positive-demand arcs, a number that can be expected to be small in applications. In conjunction with known results, we derive constant-factor approximations if $C \in O(\log n)$ and $O\big(\log C / \log\log C\big)$-approximations in general.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems, G.1.6 Optimization, G.2.1 Combinatorics, G.2.2 Graph Theory, I.2.8 Problem Solving, Control Methods, and Search

Keywords and phrases vehicle routing, transportation, Rural Postman, Chinese Postman, NP-hard problem, parameterized algorithm, combinatorial optimization

Digital Object Identifier 10.4230/OASIcs.ATMOS.2015.130

1 Introduction

Golden and Wong [16] introduced the Capacitated Arc Routing problem in order to model the search for minimum-cost routes for vehicles of equal capacity that are initially located in a vehicle depot and have to serve all "customer" demands. Applications of Capacitated Arc Routing include snow plowing, waste collection, meter reading, and newspaper delivery [7]. Herein, the customer demands require that roads of a road network are served. The road network is modeled as a graph whose edges represent roads and whose vertices can be thought of as road intersections. The customer demands are modeled as positive integers assigned to edges of this network. Moreover, each edge has a travel cost.

Capacitated Arc Routing Problem (CARP)
Instance: An undirected graph $G = (V, E)$, a *depot* vertex $v_0 \in V$, travel costs $c\colon E \to \mathbb{N} \cup \{0\}$, edge demands $d\colon E \to \mathbb{N} \cup \{0\}$, and a vehicle capacity Q.
Task: Find a set W of closed walks in G, each corresponding to the route of one vehicle and passing through the depot vertex v_0, and a serving function $s\colon W \to 2^E$ such that

- $\sum_{w \in W} c(w)$ is minimized, where $c(w) := \sum_{i=1}^{\ell} c(e_i)$ for a walk $w = (e_1, e_2, \ldots, e_\ell) \in E^\ell$,
- each closed walk $w \in W$ *serves* a subset $s(w)$ of edges of w and $\sum_{e \in s(w)} d(e) \leq Q$,
- each edge e with $d(e) > 0$ is served by exactly one walk in W.

Note that vehicle routes may traverse each vertex or edge of the input graph multiple times. Well-known special cases of CARP are the NP-hard Rural Postman Problem [21], where the vehicle capacity is unbounded and hence, the goal is to find a shortest possible route for one vehicle that visits all positive-demand edges, and the polynomial-time solvable Chinese Postman Problem [9, 10], where additionally all edges have positive demand.

15th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS'15).
Editors: Giuseppe F. Italiano and Marie Schmidt; pp. 130–143

OpenAccess Series in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Mixed and windy problem variants. CARP is polynomial-time constant-factor approximable [4, 20, 25]. It is natural to study the approximability of generalizations of CARP on directed, mixed, and windy graphs. This is also noted in a recent survey on the computational complexity of arc routing problems by van Bevern, Niedermeier, Sorge, and Weller [5, Challenge 5]. Herein, a mixed graph may contain directed arcs in addition to undirected edges in order to model one-way roads or the requirement of servicing a road in a *specific* direction or in *both* directions. In a windy graph, the cost for traversing an undirected edge $\{u, v\}$ in the direction from u to v may be different from the cost for traversing it in the opposite direction. (This models sloped roads, for example.) In this work, we present approximation algorithms for mixed and windy variants of CARP. To formally state the problem, we need some terminology related to mixed graphs.

▶ **Definition 1** (Walks in mixed and windy graphs)**.** A *mixed graph* is a triple $G = (V, E, A)$, where V is a set of *vertices*, $E \subseteq \{\{u, v\} \mid u, v \in V\}$ is a set of *(undirected) edges*, $A \subseteq V \times V$ is a set of *(directed) arcs* (that might contain loops), and no pair of vertices has an arc *and* and an edge between them. The *head* of an arc $(u, v) \in V \times V$ is v, its *tail* is u.

A *walk in* G is a sequence $w = (a_1, a_2, \ldots, a_\ell)$ such that, for each $a_i = (u, v)$, $1 \leq i \leq \ell$, we have $(u, v) \in A$ or $\{u, v\} \in E$ and such that the tail of a_i is the head of a_{i-1} for $1 < i \leq \ell$. If (u, v) occurs in w, then we say that w *traverses* the arc $(u, v) \in A$ or the edge $\{u, v\} \in E$, respectively. If the tail of a_1 is the head of a_ℓ, then we call w a *closed walk*.

If $c\colon V \times V \to \mathbb{N} \cup \{0, \infty\}$ is the *travel cost* between vertices of G, the *cost of a walk* $w = (a_1, \ldots, a_\ell)$ is $c(w) := \sum_{i=1}^{\ell} c(a_i)$. The *cost of a set* W *of walks* is $c(W) := \sum_{w \in W} c(w)$.

Formally, we present approximation algorithms for the following problem.

Mixed and Windy Capacitated Arc Routing Problem (MWCARP)
Instance: A mixed graph $G = (V, E, A)$, a depot vertex $v_0 \in V$, travel costs $c\colon V \times V \to \mathbb{N} \cup \{0, \infty\}$, demands $d\colon E \cup A \to \mathbb{N} \cup \{0\}$, and a vehicle capacity Q.
Task: Find a minimum-cost set W of closed walks in G, each passing through the depot vertex v_0, and a serving function $s\colon W \to 2^{E \cup A}$ such that

- each $w \in W$ *serves* a subset $s(w)$ of the edges and arcs it traverses and $\sum_{e \in s(w)} d(e) \leq Q$,
- each edge or arc e with $d(e) > 0$ is served by exactly one walk in W.

For brevity, we use the term "arc" to refer to both edges and arcs. Besides studying the approximability of MWCARP, we also consider the following special case:

Mixed and Windy Rural Postman Problem (MWRPP)
Instance: A mixed graph $G = (V, E, A)$ with travel costs $c : V \times V \to \mathbb{N} \cup \{0, \infty\}$ and a set $R \subseteq E \cup A$ of *required arcs*.
Task: Find a minimum-cost closed walk in G traversing all arcs in R.

If, moreover, $E = \emptyset$, then we obtain the Directed Rural Postman Problem (DRPP).

Relation to metric asymmetric TSP. In the development of approximation algorithms for MWCARP, one has to be aware of the fact that, even for DRPP, there cannot be approximation algorithms better than those for the following strongly related variant of TSP:

Metric Asymmetric Traveling Salesperson Problem ($\triangle$-ATSP)
Instance: A set V of vertices and travel costs $c\colon V \times V \to \mathbb{N} \cup \{0\}$ satisfying the triangle inequality $c(u, v) \leq c(u, w) + c(w, v)$ for all $u, v, w \in V$.
Task: Find a minimum-cost cycle that visits every vertex in V exactly once.

Given a $\triangle$-ATSP instance, one obtains an equivalent DRPP instance simply by adding a zero-cost loop to each vertex and by adding these loops to the set R of required arcs. This leads to the following observation.

▶ **Observation 2.** *Any α-approximation for* DRPP *yields an α-approximation for* $\triangle$-ATSP.

The constant-factor approximability of $\triangle$-ATSP is a long-standing open problem, in contrast to the *symmetric* metric TSP, where the cost of an arc does not depend on its direction. Symmetric metric TSP admits the famous 3/2-approximation by Christofides [6] and Serdyukov [23]. For $\triangle$-ATSP, however, the relatively recent $O(\log n/\log\log n)$-approximation by Asadpour, Goemans, Mądry, Gharan, and Saberi [2] is the first asymptotic improvement over the $O(\log n)$-approximation by Frieze, Galbiati, and Maffioli [15] from 1982.

Our contribution. As discussed above, any α-approximation for DRPP yields an α-approximation for $\triangle$-ATSP. Our contribution is the following theorem for the converse direction.

▶ **Theorem 3.** *If n-vertex* $\triangle$-ATSP *is $\alpha(n)$-approximable in $t(n)$ time, then*
(i) *n-vertex* DRPP *is $(\alpha(C)+1)$-approximable in $O(t(C)+n^3\log n)$ time,*
(ii) *n-vertex* MWRPP *is $(\alpha(C)+3)$-approximable in $O(t(C)+n^3\log n)$ time, and*
(iii) *n-vertex* MWCARP *is $O(\alpha(C+1))$-approximable in $O(t(C+1)+n^3\log n)$ time,*
where C is the number of weakly connected components in the subgraph induced by the positive-demand arcs and edges.

The theorem shows that, although MWCARP is generally not easier to approximate than $\triangle$-ATSP, the approximation quality of MWCARP depends mainly on the number C of weakly connected components in the subgraph induced by positive-demand arcs. There are applications where C is small, which is also exploited in exact exponential-time algorithms for DRPP [12, 17, 24]. For example, the company Berliner Stadtreinigungsbetriebe provided us with instances arising in snow plowing in Berlin, in which the required arcs induce a subgraph with only three or four weakly connected components.

A consequence of Theoem 3 is the following corollary, which follows from the exact $O(2^n n^2)$ time algorithm for n-vertex $\triangle$-ATSP by Bell [3], Held, and Karp [19]:

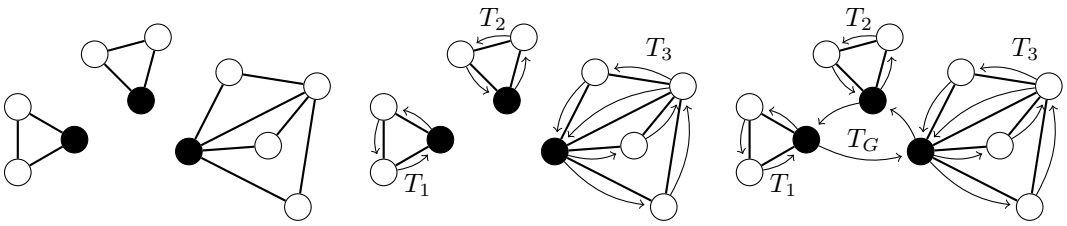
▶ **Corollary 4.** MWCARP *is constant-factor approximable in $O(2^C C^2+n^3\log n)$ time and thus in polynomial time for $C\in O(\log n)$.*

For perspective on Corollary 4, recall that finding a polynomial-time constant-factor approximation for MWCARP in general would, via Observation 2, answer a question open since 1982 [15]. Computing *optimal* solutions of MWCARP is NP-hard even if $C=1$ [5].

2 Preliminaries

Although DRPP, MWRPP, and MWCARP are problems on mixed graphs as defined in Definition 1, in some of our proofs we use more general mixed *multigraphs* $G=(V,E,A)$ with a set $V=:V(G)$ of *vertices*, a multiset $E=:E(G)$ over $\{\{u,v\}\mid u,v\in V\}$ of *(undirected) edges*, a multiset $A=:A(G)$ over $V\times V$ of *(directed) arcs* that may contain self-loops, and *travel costs* $c\colon V\times V\to\mathbb{N}\cup\{0,\infty\}$. If $E=\emptyset$, then G is a *directed multigraph*.

From Definition 1, recall the definition of walks in mixed graphs. An *Euler tour for G* is a closed walk that traverses each arc and each edge of G as often as it is present in G. A graph is *Eulerian* if it allows for an Euler tour. Let $w=(a_1,a_2,\ldots,a_\ell)$ be a walk. The *starting point of w* is the tail of a_1, the *end point of w* is the head of a_ℓ. A *segment* of w



(a) Input: Only required arcs R shown, vertices in V_R are black.

(b) Compute Euler tours T_i for each conn. component of $G[R]$.

(c) Add closed walk T_G to get a feasible solution T.

Figure 1 Steps of Algorithm 1 executed to construct a feasible solution for DRPP when all connected components of $G[R]$ are Eulerian.

is a consecutive subsequence of w. Two segments $w_1 = (a_i, \ldots, a_j)$ and $w_2 = (a_{i'}, \ldots, a_{j'})$ of a walk w are *non-overlapping* if $j < i'$ or $j' < i$. Note that two segments of w might be non-overlapping yet share arcs if w contains an arc several times. The *distance* $\mathrm{dist}_G(u, v)$ from vertex u to vertex v of G is the minimum cost of a walk from u to v in G.

The *underlying undirected (multi)graph* of G is obtained by replacing all directed arcs by undirected edges. Two vertices u, v of G are *(weakly) connected* if there is a walk starting in u and ending in v in the underlying undirected graph of G. A *(weakly) connected component* of G is a maximal subgraph of G in which all vertices are mutually (weakly) connected.

For a multiset $R \subseteq V \times V$ of arcs, $G[R]$ is the directed multigraph consisting of the arcs in R and their incident vertices of G. We say that $G[R]$ is the graph *induced by the arcs in* R. For a walk $w = (a_1, \ldots, a_\ell)$ in G, $G[w]$ is the directed multigraph consisting of the arcs $a_1, \ldots, a_\ell$ and their incident vertices, where $G[w]$ contains each arc with the multiplicity it occurs in w. Note that $G[R]$ and $G[w]$ might contain arcs with a higher multiplicity than G and, therefore, are not necessarily sub(multi)graphs of G. Finally, the cost of a multiset R is $c(R) := \sum_{a \in R} \nu(a)c(a)$, where $\nu(a)$ is the multiplicity of a in R.

3 Rural Postman

In this section, we present our approximation algorithms for DRPP and MWRPP, thus proving Theorem 3(i) and (ii). We first present, in Section 3.1, an algorithm for the special case of DRPP where the required arcs induce a subgraph with Eulerian connected components. Sections 3.2 and 3.3 subsequently generalize this algorithm to DRPP and MWRPP by adding to the set of required arcs an arc set of low weight so that the required arcs induce a graph with Eulerian connected components.

3.1 Special Case: Required arcs induce Eulerian components

To turn $\alpha(n)$-approximations for n-vertex $\triangle$-ATSP into $(\alpha(C) + 1)$-approximations for this special case of DRPP, we use Algorithm 1. Figure 1 illustrates its two main steps.

In fact, to solve this special case of DRPP, we will not exploit that Algorithm 1 and the following lemma allow R to be a *multi*set and that they allow V_R, the set of vertices incident with arcs of R, to contain more than one vertex of each connected component of $G[R]$. This will become relevant in Section 3.2, when we plug in Algorithm 1 to solve DRPP in general.

▶ **Lemma 5.** *Let G be a directed graph with travel costs c and R be a multiset of arcs of G such that $G[R]$ consists of C Eulerian connected components, let $V_R \subseteq V(G[R])$ be a vertex*

Algorithm 1: Algorithm for the proof of Lemma 5.

Input: A directed graph G with travel costs c, a multiset R of arcs of G such that $G[R]$ consists of C Eulerian connected components, and a set $V_R \subseteq V(G[R])$ containing at least one vertex of each connected component of $G[R]$.

Output: A closed walk traversing all arcs in R.

1 **for** $i = 1, \ldots, C$ **do**
2 $\quad v_i \leftarrow$ any vertex of V_R in component i of $G[R]$;
3 $\quad T_i \leftarrow$ Euler tour of connected component i of $G[R]$ starting and ending in v_i;
4 $(V_R, c') \leftarrow \triangle$-ATSP instance on the vertices V_R, where $c'(v_i, v_j) := \mathrm{dist}_G(v_i, v_j)$;
5 $T_{V_R} \leftarrow \alpha(|V_R|)$-approximate $\triangle$-ATSP solution for (V_R, c');
6 $T_G \leftarrow$ closed walk for G obtained by replacing each arc (v_i, v_j) on T_{V_R} by a shortest path from v_i to v_j in G;
7 $T \leftarrow$ closed walk obtained by following T_G and taking a detour T_i whenever reaching a vertex v_i;
8 **return** T;

set containing at least one vertex of each connected component of $G[R]$, and let $\tilde{T}$ be any closed walk containing the vertices V_R.

If n-vertex $\triangle$-ATSP is $\alpha(n)$-approximable in $t(n)$ time, then Algorithm 1 applied to (G, c, R) and V_R returns a closed walk of cost at most $c(R) + \alpha(|V_R|) \cdot c(\tilde{T})$ in $O(t(n) + n^3)$ time that traverses all arcs of R.

Proof. We first show that the closed walk T returned by Algorithm 1 visits all arcs in R. Since the $\triangle$-ATSP solution T_{V_R} constructed in line 5 visits all vertices V_R, in particular $v_1, \ldots, v_C$, so does the closed walk T_G constructed in line 6. Thus, for each vertex v_i, $1 \leq i \leq C$, T takes Euler tour T_i through the connected component i of $G[R]$ and, thus, visits all arcs in R.

We analyze the cost $c(T)$. The closed walk T is composed of the Euler tours T_i computed in line 3 and the closed walk T_G computed in line 6. Hence, $c(T) = c(T_G) + \sum_{i=1}^{C} c(T_i)$. Since each T_i is an Euler tour for some connected component i of $G[R]$, each T_i visits each arc of component i as often as it is contained in R. Consequently, $\sum_{i=1}^{C} c(T_i) = c(R)$.

It remains to analyze $c(T_G)$. Observe first that the distances in TSP instance (V_R, c') correspond to shortest paths in G and thus fulfill the triangle inequality. We have $c(T_G) = c'(T_{V_R})$ by construction of the $\triangle$-ATSP instance (V_R, c') in line 4 and by construction of T_G from T_{V_R} in line 6. Let $\tilde{T}$ be any closed walk containing V_R and let $T^*_{V_R}$ be an optimal solution for the $\triangle$-ATSP instance (V_R, c'). If we consider the closed walk $\tilde{T}_{V_R}$ that visits the vertices V_R of the $\triangle$-ATSP instance (V_R, c') in the same order as $\tilde{T}$, we get $c'(T^*_{V_R}) \leq c'(\tilde{T}_{V_R}) \leq c(\tilde{T})$. Since the closed walk T_{V_R} computed in line 5 is an $\alpha(|V_R|)$-approximate solution to the $\triangle$-ATSP instance (V_R, c'), it finally follows that $c(T_G) = c'(T_{V_R}) \leq \alpha(|V_R|) \cdot c'(T^*_{V_R}) \leq \alpha(|V_R|) \cdot c(\tilde{T})$.

Regarding the running time, observe that the instance (V_R, c') in line 4 can be constructed in $O(n^3)$ time using the Floyd-Warshall all-pair shortest path algorithm [11], which dominates all other steps of the algorithm except for, possibly, line 5. ◀

Lemma 5 proves Theorem 3(i) for DRPP instances $I = (G, c, R)$ when $G[R]$ consists of Eulerian connected components: pick V_R to contain exactly one vertex of each of the C connected components of $G[R]$. Since an optimal solution T^* for I visits the vertices V_R and satisfies $c(R) \leq c(T^*)$, Algorithm 1 yields a solution of cost at most $c(T^*) + \alpha(C) \cdot c(T^*)$.

Algorithm 2: Algorithm for the proof of Lemma 8.

Input: A DRPP instance $I = (G, c, R)$ such that $G[R]$ has C connected components and a set V_R of vertices, one of each connected component of $G[R]$.
Output: A feasible solution for I.

```
f ← minimum-cost flow for the UMCF instance (G, balance_{G[R]}, c);
foreach a ∈ A(G) do add arc a with multiplicity f(a) to (initially empty) multiset R*;
T ← closed walk computed by Algorithm 1 applied to (G, c, R ⊎ R*) and V_R;
return T;
```

3.2 Directed Rural Postman

In the previous section, we proved Theorem 3(i) for the special case of DRPP when $G[R]$ consists of Eulerian connected components. We will now reduce DRPP to this special case in order to prove Theorem 3(i) for the general DRPP. To this end, observe that a feasible solution T for a DRPP instance (G, c, R) enters each vertex v of G as often as it leaves. Thus, if we consider the multigraph $G[T]$ on the vertex set $V(G)$ that contains each arc of G with same multiplicity as T, then $G[T]$ is a supermultigraph of $G[R]$ in which every vertex is *balanced* [8, 24]:

▶ **Definition 6** (Balance)**.** By $\text{balance}_G(v) := \text{indeg}_G(v) - \text{outdeg}_G(v)$, we denote the *balance* of a vertex v of a graph G. We call a vertex v *balanced* if $\text{balance}_G(v) = 0$.

Since $G[T]$ is a supergraph of $G[R]$ in which all vertices are balanced and since a directed connected multigraph is Eulerian if and only if all its vertices are balanced, we immediately obtain the below observation. Herein and in the following, for two (multi-)sets X and Y, $X \uplus Y$ is the multiset obtained by adding the multiplicities of each element in X and Y.

▶ **Observation 7.** *Let T be a feasible solution for a* DRPP *instance (G, c, R) such that $G[R]$ has C connected components and let R^* be a minimum-cost multiset of arcs of G such that every vertex in $G[R \uplus R^*]$ is balanced. Then, $c(R \uplus R^*) \leq c(T)$ and $G[R \uplus R^*]$ consists of at most C Eulerian connected components.*

Algorithm 2 computes an $(\alpha(C) + 1)$-approximation for a DRPP instance (G, c, R) by first computing a minimum-cost arc multiset R^* such that $G[R \uplus R^*]$ contains only balanced vertices and then applying Algorithm 1 to $(G, c, R \uplus R^*)$. To find R^*, we use a folklore reduction [8, 10, 13] to the Uncapacitated Minimum-Cost Flow problem:

Uncapacitated Minimum-Cost Flow (UMCF)
Instance: A directed graph $G = (V, A)$ with *supply* $s\colon V \to \mathbb{Z}$ and *costs* $c\colon A \to \mathbb{N} \cup \{0\}$.
Task: Find a *flow* $f\colon A \to \mathbb{N} \cup \{0\}$ minimizing $\sum_{a \in A} c(a)f(a)$ such that, for each $v \in V$,

$$\sum_{(v,w) \in A} f(v, w) - \sum_{(w,v) \in A} f(w, v) = s(v). \tag{FC}$$

Equation (FC) is known as the *flow conservation* constraint: for every vertex v with $s(v) = 0$, there are as many units of flow entering the node as leaving it. Nodes v with $s(v) > 0$ "produce" $s(v)$ units of flow, whereas nodes v with $s(v) < 0$ "consume" $s(v)$ units of flow. UMCF is solvable in $O(n^3 \log n)$ time [1, Theorem 10.34].

▶ **Lemma 8.** *Let $I := (G, c, R)$ be a* DRPP *instance such that $G[R]$ has C connected components and let V_R be a vertex set containing exactly one vertex of each connected component of $G[R]$. Moreover, consider two closed walks in G:*

- *let $\tilde{T}$ be any closed walk containing the vertices V_R and*
- *let $\hat{T}$ be any feasible solution for I.*

If n-vertex $\triangle$-ATSP is $\alpha(n)$-approximable in $t(n)$ time, then Algorithm 2 applied to I and V_R returns a feasible solution of cost at most $c(\hat{T}) + \alpha(C) \cdot c(\tilde{T})$ in $O(t(n) + n^3 \log n)$ time.

Proof. Observe that Algorithm 2 in line 2 indeed computes a minimum-cost arc set R^* such that all vertices in $G[R \uplus R^*]$ are balanced (we provide details in Appendix A).

We use the optimality of R^* to give an upper bound on the cost of the closed walk T computed in line 3. Since V_R contains exactly one vertex of each connected component of $G[R]$, it contains at least one vertex of each connected component of $G[R \uplus R^*]$. Therefore, Algorithm 1 is applicable to $(G, c, R \uplus R^*)$ and, by Lemma 5, yields a closed walk in G traversing all arcs in $R \uplus R^*$ and having cost at most $c(R \uplus R^*) + \alpha(|V_R|) \cdot c(\tilde{T})$. This is a feasible solution for (G, c, R) and, since by Observation 7, we have $c(R \uplus R^*) \leq c(\hat{T})$, it follows that this feasible solution has cost at most $c(\hat{T}) + \alpha(C) \cdot c(\tilde{T})$.

Finally, the running time of Algorithm 2 follows from the fact that the minimum-cost flow in line 1 is computable in $O(n^3 \log n)$ time [1, Theorem 10.34] and that Algorithm 1 runs in $O(n^3 + t(C))$ time (Lemma 5). ◀

Proof of Theorem 3(i). Let (G, c, R) be an instance of DRPP and let V_R be a set of vertices containing exactly one vertex of each connected component of $G[R]$. An optimal solution T^* for I contains all arcs in R and all vertices in V_R and hence, by Lemma 8, Algorithm 2 computes a feasible solution T with $c(T) \leq c(T^*) + \alpha(C) \cdot c(T^*)$ for I. ◀

3.3 Mixed and Windy Rural Postman

In the previous section, we presented Algorithm 2 for DRPP in order to prove Theorem 3(i). We now show how to apply Algorithm 2 to MWRPP in order to prove Theorem 3(ii).

To this end, we replace each undirected edge $\{u, v\}$ in an MWRPP instance by two directed arcs (u, v) and (v, u), where we force the undirected *required* edges of the MWRPP instance to be traversed in the cheaper direction:

▶ **Lemma 9.** *Let $I := (G, c, R)$ be an* MWRPP *instance and let $I' := (G', c, R')$ be the* DRPP *instance obtained from I as follows:*

- *G' is obtained by replacing each edge $\{u, v\}$ of G by two arcs (u, v) and (v, u),*
- *R' is obtained from R by replacing each edge $\{u, v\} \in R$ by an arc (u, v) if $c(u, v) \leq c(v, u)$ and by (v, u) otherwise.*

Then, each feasible solution for I' is a feasible solution of the same cost for I and, for each feasible solution T for I, there is a feasible solution T' for I' with $c(T') < 3c(T)$.

We prove Lemma 9 in Appendix B. Using Lemma 9, it is easy to prove Theorem 3(ii).

Proof of Theorem 3(ii). Given an MWRPP instance $I = (G, c, R)$, compute a DRPP instance $I' := (G', c, R')$ as described in Lemma 9. This can be done in linear time.

Let V_R be a set of vertices containing exactly one vertex of each connected component of $G'[R']$ and let T^* be an optimal solution for I. Observe that T^* is not necessarily a feasible solution for I', since it might serve required arcs of I' in the wrong direction. Yet T^* is a closed walk in G' visiting all vertices of V_R. Moreover, by Lemma 9, I' has a feasible solution T' with $c(T') \leq 3c(T^*)$.

Thus, applying Algorithm 2 to I' and V_R yields a feasible solution T of cost at most $c(T') + \alpha(C) \cdot c(T^*) \leq 3c(T^*) + \alpha(C) \cdot c(T^*)$ due to Lemma 8. Finally, T is also a feasible solution for I by Lemma 9. ◀

Algorithm 3: Algorithm for the proof of Proposition 12.

Input: An MWCARP instance $I = (G, v_0, c, d, Q)$ such that $(v_0, v_0) \in R_d$ and such that $G[R_d]$ has C connected components.
Output: A feasible solution for I.

```
/* Compute a base tour containing all demand arcs and the depot      */
```
1 $I' \leftarrow$ MWRPP instance $I' := (G, c, R_d)$;
2 $T \leftarrow \beta(C)$-approximate MWRPP tour for I' starting and ending in v_0;

```
/* Split the base tour into one tour for each vehicle                */
```
3 $(W, s) \leftarrow$ a feasible splitting of T;
4 **foreach** $w \in W$ **do**
5 close w by adding shortest paths from v_0 to s and from t to v_0 in G, where s, t are the start and endpoints of w, respectively;
6 **return** (W, s);

4 Capacitated Arc Routing

Our approximation algorithm for MWCARP uses the fact that joining all vehicle tours of a solution gives an MWRPP tour traversing all positive-demand arcs and the depot. Thus, in order to approximate MWCARP, the idea is to first compute an approximate MWRPP tour and then split it into subtours, each of which can be served by a vehicle of capacity Q. Then we close each subtour by shortest paths via the depot. This algorithm is inspired by the CARP algorithms of Jansen [20] and Wøhlk [25] and the algorithm of Frederickson, Hecht, and Kim [14] for (undirected) k-person minimax routing problems. Our analysis, however, is necessarily different, since we cannot use arcs and edges in backwards direction.

▶ **Definition 10** (Demand arc). For a demand function $d\colon E(G) \cup A(G) \to \mathbb{N} \cup \{0\}$ we define $R_d := \{a \in E(G) \cup A(G) \mid d(a) > 0\}$ to be the set of *demand arcs*.

We will construct an MWCARP solution from a *feasible splitting* of an MWRPP tour T.

▶ **Definition 11** (Feasible splitting). For an MWCARP instance $I = (G, v_0, c, d, Q)$, let T be a closed walk containing all arcs in R_d and $W = (w_1, \dots, w_\ell)$ be a tuple of segments of T. In the following, we refer by W to both the tuple and the set of walks it contains.

Furthermore, consider a serving function $s\colon W \to 2^{R_d}$ that assigns to each walk the set of arcs in R_d it serves. We call (W, s) a *feasible splitting of* T if the following conditions hold:

1. the walks in W are mutually non-overlapping segments of T,
2. when concatenating the walks in W in order, one obtains a subsequence of T,
3. each $w_i \in W$ begins and ends with an arc in $s(w_i)$,
4. $\{s(w_i) \mid w_i \in W\}$ is a partition of R_d, and
5. for each $w_i \in W$, we have $\sum_{e \in s(w_i)} d(e) \leq Q$ and, if $i < \ell$, then $\sum_{e \in s(w_i)} d(e) + d(a) > Q$, where a is the first arc served by w_{i+1}.

A feasible splitting of a given closed walk T as above can be computed in linear time using a greedy strategy (we refer to Appendix C for details).

The algorithm. Algorithm 3 constructs an MWCARP solution from an approximate MWRPP solution T containing all demand arcs and the depot v_0. In order to ensure that T contains v_0, Algorithm 3 assumes that the input graph has a demand loop (v_0, v_0): if this loop is not present, one can add it with zero cost. Note that, while this does not change

the cost of an optimal solution, it might increase the number of connected components in the subgraph induced by demand arcs by one. To compute an MWCARP solution from T, Algorithm 3 first computes a feasible splitting (W, s) of T. To each walk $w_i \in W$, it then adds a shortest path from the end of w_i to the start of w_i via the depot. It is not hard to check that Algorithm 3 indeed outputs a feasible solution by using the properties of feasible splittings and the fact that T contains all demand arcs. The remainder of this section is devoted to the analysis of the solution, thus proving the following proposition and, consequently, Theorem 3(iii).

▶ **Proposition 12.** *Let $I = (G, v_0, c, d, Q)$ be an* MWCARP *instance and let I' be the instance obtained from I by adding a zero-cost demand arc (v_0, v_0) if it is not present.*

If MWRPP *is $\beta(C)$-approximable in $t(n)$ time, then Algorithm 3 applied to I' computes a $(8\beta(C+1)+3)$-approximation for I in $O(t(n)+n^3)$ time. Herein, C is the number of connected components in $G[R_d]$.*

The following lemma follows from the observation that the concatenation of all vehicle tours in any MWCARP solution yields an MWRPP tour containing all demand arcs and the depot. It is proven in Appendix D.

▶ **Lemma 13.** *Let $I = (G, v_0, c, d, Q)$ be an* MWCARP *instance with $(v_0, v_0) \in R_d$ and an optimal solution (W^*, s^*). The closed walk T and its feasible splitting (W, s) computed in lines 2 and 3 of Algorithm 3 satisfy $c(W) \leq c(T) \leq \beta(C)c(W^*)$, where C is the number of connected components in $G[R_d]$.*

It remains to analyze the length of the shortest paths from v_0 to $w_i \in W$ and from w_i to v_0 added in line 5 of Algorithm 3. We bound their lengths in the lengths of an auxiliary walk $A(w_i)$ from v_0 to w_i and of an auxiliary walk $Z(w_i)$ from w_i to v_0. The auxiliary walks $A(w_i)$ and $Z(w_i)$ consist of arcs of W, whose total cost is bounded by Lemma 13, and of arcs of an optimal solution (W^*, s^*). We show that, in total, the walks $A(w_i)$ and $Z(w_i)$ for *all* $w_i \in W$ use each subwalk of W and W^* at most a constant number of times. For this, we group the walks in W into consecutive pairs, for each of which we will be able to charge the cost of the auxiliary walks to a distinct vehicle tour of the optimal solution.

▶ **Definition 14** (Consecutive pairing)**.** For a feasible splitting (W, s) with $W = (w_1, \ldots, w_\ell)$, we call $W^2 := \{(w_{2i-1}, w_{2i}) \mid i \in \{1, \ldots, \lfloor \ell/2 \rfloor\}\}$ a *consecutive pairing*.

We can now show, by applying Hall's theorem [18], that each pair traverses an arc from a *distinct* tour of an optimal solution (Appendix E).

▶ **Lemma 15.** *Let $I = (G, v_0, c, d, Q)$ be an* MWCARP *instance with an optimal solution (W^*, s^*) and let W^2 be a consecutive pairing of some feasible splitting (W, s). Then, there is an injective map $\phi\colon W^2 \to W^*, (w_i, w_{i+1}) \mapsto w^*$ such that $(s(w_i) \cup s(w_{i+1})) \cap s^*(w^*) \neq \emptyset$.*

In the following, we fix an arbitrary arc in $(s(w_i) \cup s(w_{i+1})) \cap s^*(w^*)$ and call it the *pivot arc* of (w_i, w_{i+1}). Informally, the auxiliary walks for each w_i are constructed as follows. To get from the endpoint of w_i to v_0, walk along the closed walk T until traversing the first pivot arc a. To get from the head of a to v_0, walk along w^*, which is the walk of W^* containing a. To get from v_0 to w_i, take the same approach, that is, walk backwards on T from the start point of w_i until traversing a pivot arc and then follow the tour of W^* containing a. The formal definition of the auxiliary walks $A(w)$ and $Z(w)$ is given below (see also Figure 2).

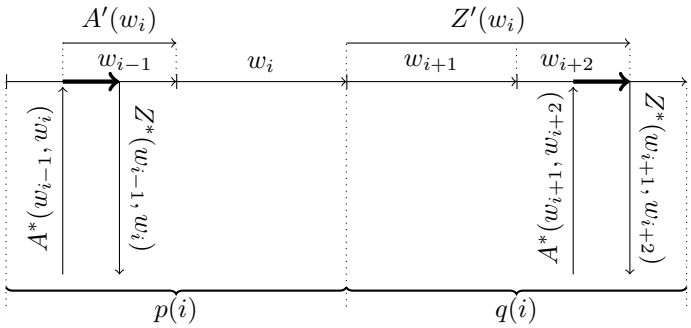


Figure 2 Illustration of Definition 16. Dotted lines are ancillary lines. Thin arrows are walks. The braces along the bottom show a consecutive pairing of walks $w_{i-1}, \dots, w_{i+2}$. Bold arcs are pivot arcs. Here, $p(i)$ is exactly the pair that contains w_i and $q(i)$ is the next pair.

▶ **Definition 16** (Auxiliary walks). Let $I = (G, v_0, c, d, Q)$ be an MWCARP instance, (W^*, s^*) be an optimal solution, and W^2 be a consecutive pairing of some feasible splitting (W, s) of a closed walk T containing all arcs R_d and v_0, where $W = (w_1, \dots, w_\ell)$.

Let $\phi\colon W^2 \to W^*$ be an injective map as in Lemma 15 and for each pair $(w, w') \in W^2$ let
$\boldsymbol{A^*(w_i, w_{i+1})}$ be a subwalk of $\phi(w_i, w_{i+1})$ from v_0 to the tail of the pivot arc of (w_i, w_{i+1}),
$\boldsymbol{Z^*(w_i, w_{i+1})}$ be a subwalk of $\phi(w_i, w_{i+1})$ from the head of the pivot arc of (w_i, w_{i+1}) to v_0.
For each walk $w_i \in W$ with $i \geq 3$ (that is, w_i is not in the first pair of W^2), let
$\boldsymbol{p(i)}$ be the index of the pair whose pivot arc is traversed first when walking T backwards starting from the starting point of w_i,
$\boldsymbol{A'(w_i)}$ be the subwalk of T starting at the end point of $A^*(w_{2p(i)}, w_{2p(i)+1})$ and ending at the start point of w_i, and
$\boldsymbol{A(w_i)}$ be the walk from v_0 to the start point of w_i following first $A^*(w_{2p(i)}, w_{2p(i)+1})$ and then $A'(w_i)$.
For each walk $w_i \in W$ with $i \leq \ell - 3$ (that is, w_i is not in the last pair of W^2, where w_ℓ might not be in any pair if ℓ is odd), let
$\boldsymbol{q(i)}$ be the index of the pair whose pivot arc is traversed first when following T starting from the end point of w_i,
$\boldsymbol{Z'(w_i)}$ be the subwalk of T starting at the end point of w_i and ending at the start point of $Z^*(w_{2q(i)}, w_{2q(i)+1})$, and, finally, let
$\boldsymbol{Z(w_i)}$ be the walk from the end point of w_i to v_0 following $Z'(w_i)$ and $Z^*(w_{2q(i)}, w_{2q(i)+1})$.
We are now ready to prove Proposition 12, which also concludes our proof of Theorem 3.

Proof of Proposition 12. Let $I = (G, v_0, c, d, Q)$ be an MWRPP instance and (W^*, s^*) be an optimal solution. If there is no demand arc (v_0, v_0) in I, then we add it with zero cost in order to make Algorithm 3 applicable. This clearly does not change the cost of an optimal solution but may increase the number of connected components of $G[R_d]$ to $C + 1$.

In lines 2 and 3, Algorithm 3 computes a tour T and its feasible splitting (W, s). Denote $W = (w_1, \dots, w_\ell)$. The solution returned by Algorithm 3 consists, for each $1 \leq i \leq \ell$, of a tour starting in v_0, following a shortest path to the starting point of w_i, then w_i, and a shortest path back to v_0.

For $i \geq 3$, the shortest path from v_0 to the starting point of w_i has length at most $c(A(w_i))$. For $i \leq \ell - 3$, the shortest path from the end point of w_i to v_0 has length at most $c(Z(w_i))$. This amounts to $\sum_{i=3}^{\ell} c(A(w_i)) + \sum_{i=1}^{\ell-3} c(Z(w_i))$. To bound the costs of the shortest paths added for $i \in \{1, 2, \ell-2, \ell-1, \ell\}$, observe the following. For each $i \in \{1, 2\}$, the shortest paths

from v_0 to the start point of w_i and from the end point of $w_{\ell-i}$ to v_0 together have length at most $c(T)$. The shortest path from the end point of w_ℓ to v_0 has length at most $c(T) - c(W)$. Thus, the solution returned by Algorithm 3 has cost at most

$$\begin{aligned}
&\sum_{i=1}^{\ell} c(w_i) + \sum_{i=3}^{\ell} c(A(w_i)) + \sum_{i=1}^{\ell-3} c(Z(w_i)) + 3c(T) - c(W) \\
&= \sum_{i=3}^{\ell} c(A(w_i)) + \sum_{i=1}^{\ell-3} c(Z(w_i)) + 3c(T) \\
&= 3c(T) + \\
&\quad + \sum_{i=3}^{\ell} c(A^*(w_{2p(i)}, w_{2p(i)+1})) + \sum_{i=1}^{\ell-3} c(Z^*(w_{2q(i)}, w_{2q(i)+1})) + \qquad \text{(S1)} \\
&\quad + \sum_{i=3}^{\ell} c(A'(w_i)) + \sum_{i=1}^{\ell-3} c(Z'(w_i)). \qquad \text{(S2)}
\end{aligned}$$

Observe that, for a fixed i, one has $p(i) = p(j)$ only for $j \leq i + 2$ and $q(i) = q(j)$ only for $j \geq i - 2$. Moreover, by Lemma 15 and Definition 16, for $i \neq j$, $A^*(w_i, w_{i+1})$ and $A^*(w_j, w_{j+1})$ are subwalks of distinct walks of W^*. Similarly, $Z^*(w_i, w_{i+1})$ and $Z^*(w_j, w_{j+1})$ are subwalks of distinct walks of W^* if $i \neq j$. Hence, sum (S1) counts every arc of W^* at most three times and is therefore bounded from above by $3c(W^*)$. Moreover, for a walk w_i, let $\mathcal{A}_i$ be the set of walks w_j such that any arc a of w_i is contained in $A'(w_j)$ and let $\mathcal{Z}_i$ be the set of walks such that any arc a of w_i is contained in $Z'(w_j)$. Observe that $A'(w_j)$ and $Z'(w_j)$ cannot completely contain two walks of the same pair of the consecutive pairing W^2 of W since, by Lemma 15, each pair has a pivot arc and $A'(w_j)$ and $Z'(w_j)$ both stop after traversing a pivot arc. Hence, the walks in $\mathcal{A}_i \cup \mathcal{Z}_i$ can be from at most three pairs of W^2: the pair containing w_i and the two neighboring pairs. Finally, observe that w_i itself is not contained in $\mathcal{A}_i \cup \mathcal{Z}_i$. Thus, $\mathcal{A}_i \cup \mathcal{Z}_i$ contains at most five walks (Figure 3 in the appendix shows such a worst-case example). Therefore, sum (S2) counts every arc of W at most five times and is bounded from above by $5c(W)$.

Thus, Algorithm 3 returns a solution of cost $3c(T)+5c(W)+3c(W^*)$ which, by Lemma 13, is at most $8c(T) + 3c(W^*) \leq 8\beta(C+1)c(W^*) + 3c(W^*) \leq (8\beta(C+1)+3)c(W^*)$. ◀

5 Conclusion

With the exception of MWCARP, we expect our algorithms to yield good heuristics. In particular, the $\triangle$-ATSP instances should be sufficiently small to allow for the computation of optimal solutions. For MWCARP, a better approach than the presented one could be to compute an MWRPP tour and then compute an *optimal* splitting of this tour into vehicle tours. Our analysis gives a worst-case bound for this approach. We conclude with an open question: can the $(\alpha(C) + 3)$-approximation for MWRPP in Theorem 3(ii) be improved to an $(\alpha(C) + 3/2)$-approximation using the 3/2-approximation for Mixed Chinese Postman given by Raghavachari and Veerasamy [22]?

Acknowledgments. We thank Sepp Hartung, Iyad Kanj, and André Nichterlein for fruitful discussions. This research was initiated during a research retreat of the algorithms and complexity group of TU Berlin, held in Rothenburg/Oberlausitz, Germany, in March 2015, while René van Bevern was with TU Berlin under support of the DFG, project DAPA (NI 369/12). Manuel Sorge was supported by the DFG, project DAPA (NI 369/12).

References

1 Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows—Theory, Algorithms and Applications*. Prentice Hall, 1993.

2 Arash Asadpour, Michel X. Goemans, Aleksander Mądry, Shayan Oveis Gharan, and Amin Saberi. An $O(\log n/\log\log n)$-approximation algorithm for the asymmetric traveling salesman problem. In *Proc. SODA'10*, pages 379–389. SIAM, 2010.

3 Richard Bellman. Dynamic programming treatment of the Travelling Salesman Problem. *J. ACM*, 9(1):61–63, 1962.

4 René van Bevern, Sepp Hartung, André Nichterlein, and Manuel Sorge. Constant-factor approximations for capacitated arc routing without triangle inequality. *Oper. Res. Lett.*, 42(4):290–292, 2014.

5 René van Bevern, Rolf Niedermeier, Manuel Sorge, and Mathias Weller. Complexity of arc routing problems. In *Arc Routing: Problems, Methods, and Applications*. SIAM, 2014.

6 Nicos Christofides. Worst case analysis of a new heuristic for the traveling salesman problem. Management Science Research Rept. 388, Carnegie-Mellon University, 1976.

7 Ángel Corberán and Gilbert Laporte, editors. *Arc Routing: Problems, Methods, and Applications*. SIAM, 2014.

8 Frederic Dorn, Hannes Moser, Rolf Niedermeier, and Mathias Weller. Efficient algorithms for Eulerian Extension and Rural Postman. *SIAM J. Discrete Math.*, 27(1):75–94, 2013.

9 Jack Edmonds. The Chinese postman problem. *Oper. Res.*, pages B73–B77, 1975. Suppl. 1.

10 Jack Edmonds and Ellis L. Johnson. Matching, Euler tours and the Chinese postman. *Math. Program.*, 5:88–124, 1973.

11 Robert W. Floyd. Algorithm 97: Shortest path. *Commun. ACM*, 5(6):345, June 1962.

12 Greg N. Frederickson. *Approximation Algorithms for NP-hard Routing Problems*. PhD thesis, Faculty of the Graduate School of the University of Maryland, 1977.

13 Greg N. Frederickson. Approximation algorithms for some postman problems. *J. ACM*, 26(3):538–554, 1979.

14 Greg N. Frederickson, Matthew S. Hecht, and Chul E. Kim. Approximation algorithms for some routing problems. *SIAM J. Comput.*, 7(2):178–193, 1978.

15 A. M. Frieze, G. Galbiati, and F. Maffioli. On the worst-case performance of some algorithms for the asymmetric traveling salesman problem. *Networks*, 12(1):23–39, 1982.

16 Bruce L. Golden and Richard T. Wong. Capacitated arc routing problems. *Networks*, 11(3):305–315, 1981.

17 Gregory Gutin, Magnus Wahlström, and Anders Yeo. Parameterized Rural Postman and Conjoining Bipartite Matching problems. Available on arXiv:1308.2599v4, 2014.

18 P. Hall. On representatives of subsets. *J. London Math. Soc.*, 10:26–30, 1935.

19 Michael Held and Richard M. Karp. A dynamic programming approach to sequencing problems. *J. SIAM*, 10(1):196–210, 1962.

20 Klaus Jansen. An approximation algorithm for the general routing problem. *Inform. Process. Lett.*, 41(6):333–339, 1992.

21 J. K. Lenstra and A. H. G. Rinnooy Kan. On general routing problems. *Networks*, 6(3):273–280, 1976.

22 Balaji Raghavachari and Jeyakesavan Veerasamy. A 3/2-approximation algorithm for the Mixed Postman Problem. *SIAM J. Discrete Math.*, 12(4):425–433, 1999.

23 A. I. Serdyukov. O nekotorykh ekstremal'nykh obkhodakh v grafakh. *Upravlyayemyye sistemy*, 17:76–79, 1978. [On some extremal by-passes in graphs. *zbMATH 0475.90080*].

24 Manuel Sorge, René van Bevern, Rolf Niedermeier, and Mathias Weller. A new view on Rural Postman based on Eulerian Extension and Matching. *J. Discrete Alg.*, 16:12–33, 2012.

25 Sanne Wøhlk. An approximation algorithm for the Capacitated Arc Routing Problem. *The Open Operational Research Journal*, 2:8–12, 2008.

A Omitted details in the proof of Lemma 8

Proof. To complete the proof of Theorem 8, we prove that Algorithm 2 in line 2 indeed computes a minimum-cost arc set R^* such that all vertices in $G[R \uplus R^*]$ are balanced. This follows from the one-to-one correspondence between arc multisets R' such that $G[R \uplus R']$ has only balanced vertices and flows f for the UMCF instance $I' := (G, \text{balance}_{G[R]}, c)$:

1. For each vertex v with $\text{balance}_{G[R]}(v) = 0$, R' has to contain as many incident in-arcs as out-arcs so that $\text{balance}_{G[R \uplus R']}(v) = 0$. Likewise, by (FC), in any feasible flow for I', as many units of flow enter v as leave v.
2. Each vertex v with $\text{balance}_{G[R]}(v) > 0$ has $\text{balance}_{G[R]}(v)$ more incident in-arcs than out-arcs in $G[R]$ and, thus, in order for $\text{balance}_{G[R \uplus R']}(v) = 0$ to hold, R' has to contain $\text{balance}_{G[R]}(v)$ more out-arcs than in-arcs incident to v. Likewise, by (FC), in any feasible flow for I', there are $\text{balance}_{G[R]}(v)$ more units of flow leaving v than entering v.
3. For each vertex v with $\text{balance}_{G[R]}(v) < 0$, analogous arguments apply.

Thus, from a multiset R' of arcs such that $G[R \uplus R']$ is balanced, we get a feasible flow f for I' by setting $f(v, w)$ to the multiplicity of the arc (v, w) in R'. From a feasible flow f for I', we get a multiset R' of arcs such that $G[R \uplus R']$ is balanced by adding to R' each arc (v, w) with multiplicity $f(v, w)$. We conclude that the arc multiset R^* computed in line 2 is such that $G[R \uplus R^*]$ is balanced. Moreover, it is a minimum-cost such set, since a set of lower cost would yield a flow cheaper than the optimum flow f computed in line 1. ◀

B Proof of Lemma 9

Proof. It is obvious that each feasible solution T' for I' is a feasible solution for I, since each required edge of I is served by T' in at least one direction. Moreover, the cost functions in I and I' are the same.

Now, for the opposite direction, let T be a feasible solution for I. We obtain a feasible solution T' of I' as follows:

1. For each arc (u, v) or non-required edge $\{u, v\}$ traversed by T in direction from u to v, T' traverses arc (u, v).
2. For each required edge $\{u, v\}$ traversed by T from u to v such that $c(u, v) \leq c(v, u)$, T' traverses (u, v).
3. For each required edge $\{u, v\}$ traversed by T from u to v such that $c(u, v) > c(v, u)$, T' traverses $(u, v), (v, u)$, and again (u, v).

The closed walk T' is indeed a feasible solution to I': in (2), note that $(u, v) \in R'$ and that it is served by T'. In (3), in contrast, $(v, u) \in R'$, which is also served by T'.

To compute the cost of T', observe that only (3) increases the cost of T' compared to T: instead of $c(u, v)$, which is paid by T for traversing $\{u, v\}$ in the direction from u to v, the closed walk T' pays $c(u, v) + c(v, u) + c(u, v) < 3c(u, v)$ since $c(v, u) < c(u, v)$. ◀

C Obtaining feasible splittings

Given an MWCARP instance $I = (G, v_0, c, d, Q)$, a feasible splitting (W, s) of a closed walk T that traverses all arcs in R_d can be computed in linear time as follows. We assume that each arc has demand at most Q since otherwise I has no feasible solution. Now, traverse T, successively defining subwalks $w \in W$ and the corresponding sets $s(w)$ one at a time. The traversal starts with the first arc $a \in R_d$ of T and by creating a subwalk w consisting only of a and $s(w) = \{a\}$. On discovery of a still unserved arc $a \in R_d \setminus (\bigcup_{w' \in W} s(w'))$ do the

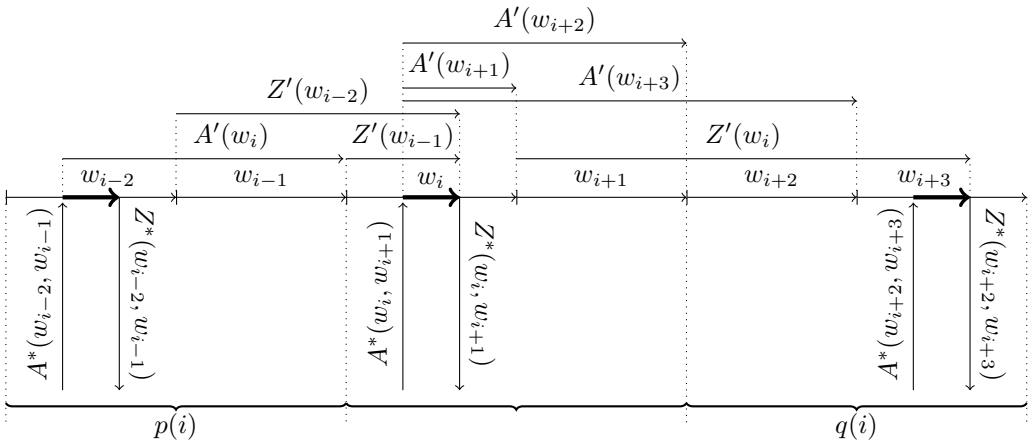


Figure 3 Illustration of the situation in which a maximum number of five different walks in W traverse the same pivot arc (the bold arc of w_i) in their respective auxiliary walks.

following. If $\sum_{e \in s(w)} d(e) + d(a) \leq Q$, then add a to $s(w)$ and append to w the subwalk of T that was traversed since discovery of the previous unserved arc in R_d. Otherwise, mark w and $s(w)$ as finished, start a new tour $w \in W$ with a as the first arc, set $s(w) = \{a\}$, and continue the traversal of T. If no arc a is found, then stop. It is not hard to verify that indeed, (W, s) is a feasible splitting.

D Proof of Lemma 13

Proof. Consider an optimal solution (W^*, s^*) to I. The closed walks in W^* visit all arcs in R_d. Concatenating them to a closed walk T^* gives a feasible solution for the MWRPP instance $I' = (G, c, R_d)$ in line 1 of Algorithm 3. Moreover, $c(T^*) = c(W^*)$. Thus, we have $c(T) \leq \beta(C)c'(T^*)$ in line 2. Moreover, by Condition 1 of Definition 11, one has $c(W) \leq c(T)$. This finally implies $c(W) \leq c(T) \leq \beta(C)c(T^*) = \beta(C)c(W^*)$ in line 3. ◀

E Proof of Lemma 15

Proof. Define an undirected bipartite graph B with the partite sets W^2 and W^*. A pair $(w, w') \in W^2$ and a closed walk $w^* \in W^*$ are adjacent in B if $(s(w) \cup s(w')) \cap s^*(w^*) \neq \emptyset$. We prove that B allows for a matching that matches each vertex of W^2 to some vertex in W^*. To this end, by Hall's theorem [18], it suffices to prove that, for all subsets $S \subseteq W^2$, it holds that $|N_B(S)| \geq |S|$, where $N_B(S) := \bigcup_{v \in S} N_B(v)$ and $N_B(v)$ is the set of neighbors of a vertex v in B. Observe that, by Condition 5 of Definition 11 of feasible splittings, for each pair $(w, w') \in W^2$ we have $d(s(w) \cup s(w')) \geq Q$. Since the pairs serve pairwise disjoint sets of demand arcs (Condition 4 of feasible splittings), the pairs in S serve a total demand of at least $Q \cdot |S|$ in the closed walks $N_B(S) \subseteq W^*$. Since each closed walk in $N_B(S)$ serves demand at most Q, the set $N_B(S)$ is at least as large as S, as required. ◀