Mixed Integer Programs to Improve Solutions of Vehicle Routing Problems with Intra-route Constraints

Simen T. Vadseth[1*], Henrik Andersson[1†], Jean-François Cordeau[2†], Magnus Stålhane[1†]

[1]Department of Industrial Economics and Technology Management, Norwegian University of Science and Technology, Norway

[2]Department of Logistics and Operations Management, HEC Montréal, Canada

*Corresponding author(s). E-mail(s): simen.t.vadseth@ntnu.no;
†These authors contributed equally to this work.

Abstract

Many variants of the vehicle routing problem (VRP) pose significant computational challenges in logistics optimization, and improvement heuristics have emerged as effective tools for refining solutions found by local search methods and meta-heuristics. This paper introduces exact route modifying improvement models (RMIM) that aim to assemble high-quality solutions by selecting routes from a pool while allowing modifications to be made to the selected routes. We evaluate these models on vehicle routing problems with intra-route constraints, including the multi-trip VRP (MTVRP), the pickup and delivery problem with time windows (PDPTW), and the VRP with time windows (VRPTW). Our proposed matheuristic approach achieves best known solutions for most benchmark instances for the MTVRP, and the RMIMs improve many existing solutions for both the PDPTW and the VRPTW. These findings showcase the value of using RMIMs to enhance solutions to different types of VRPs.

Keywords: Vehicle Routing Problem, Matheuristic, MTVRP, PDPTW, VRPTW

1 Introduction

Vehicle routing problems are common in practice but also serve as important testbeds in operations research. These complex optimization problems usually consist of a fleet of vehicles based at one or several depots that must serve the demands of customers. The most studied vehicle routing problem, the capacitated vehicle routing problem (CVRP), considers a single depot that serves a set of customers with a capacitated fleet of vehicles in a single time period (Laporte, 2009). However, richer vehicle routing problems that include, for example, time windows, pickups and deliveries, multiple time periods, or inventory management have also been extensively studied (Toth and Vigo, 2014). Vehicle routing problems are known to be NP-hard (Lenstra and Kan, 1981), which means that they can take an impractically long time to solve to optimality using exact algorithms. As a result, heuristic algorithms have been developed to solve these problems approximately. Heuristics are faster than exact algorithms and can find good-quality solutions in a shorter amount of computing time (Cordeau et al., 2005).

One specific type of heuristics, namely improvement heuristics, has proven to be an important and efficient tool to solve vehicle routing problems. Improvement heuristics are algorithms that aim to improve an existing solution by iteratively modifying it. They are widely used to solve problems in various domains, such as manufacturing, transportation, healthcare, and supply chain management. Traditionally, these heuristics use local search techniques to improve a given solution. Local search algorithms start with an initial solution and make incremental changes until an optimal or good solution is found. These algorithms can either follow a straight path, where only modifications that improve the solutions are accepted, or a path where temporary deteriorations of the objective function value are accepted. Descent or hill climbing heuristics are examples of the former. They always stop in a local optimum and are therefore very dependent on the initial solution. The latter group can offset these limitations by accepting worsening solutions (Van Breedam, 1995). Tabu search, simulated annealing, and adaptive large neighborhood search algorithms are examples of these types of improvement heuristics and have been used with great success on several vehicle routing problems (Van Breedam, 1995; Pisinger and Ropke, 2007; Cordeau and Laporte, 2005). Thus, improvement heuristics are flexible, as they can be scaled to solve problems of varying complexity. Moreover, they can be customized to meet the specific constraints of individual problems.

In recent years, the performance of CPUs and mathematical programming solvers has largely increased. Consequently, combining mathematical programming techniques with heuristic methods has become more common to solve optimization problems more effectively. These solution methods are usually referred to as *matheuristics.* Matheuristics often have the heuristic advantage of solving problems quickly. In addition, they have the ability to handle a larger number of constraints and variables as well as higher complexity compared to traditional heuristics. Consequently, several improvement matheuristics have been developed. For routing problems, Archetti and Speranza (2014) define three types of matheuristics in their survey paper, where improvement matheuristics is one of them. One of the most popular and successful improvement approaches is collecting routes in a route pool during the optimization process and using a set partitioning formulation at different intervals to select a feasible subset of routes. This approach has been successfully used by several researchers, including Archetti et al. (2017) for the inventory routing problem (IRP), Alvarenga et al. (2007) for the vehicle routing problem with time windows (VRPTW), and Subramanian et al. (2013) for several vehicle routing variants such as the CVRP, the Asymmetric VRP, the Open VRP and the Multi-depot VRP among others.

The major drawback of the set-partitioning approach is that if the route pool does not contain the correct routes, the approach will not produce good solutions. This holds even if the pool includes most of the optimal routes or routes very similar to the optimal ones. To deal with this drawback, several researchers have proposed set-partitioning-inspired formulations that select routes from a route pool; however, smaller and larger modifications of the routes are allowed. We refer to these improvement formulations as *Route Modifying Improvement Models* (RMIMs). Typically, the costs of these route modifications are calculated approximately (Russell, 2017; Solyalı and Süral, 2017; Manousakis et al., 2022). However, to solve the production routing problem (PRP), Vadseth et al. (2023) developed an exact RMIM where each modification is calculated exactly in terms of cost with respect to the original problem and any solution to the RMIM is a feasible solution to the original problem. In this RMIM, routes are selected from a route pool. Still, single customer nodes can be both inserted or removed from each route, and their *multi-start route improving* *matheuristic for the production routing problem* was able to improve the state-of-the-art and find several new best known solutions for the PRP. This methodology was further developed in Skålnes et al. (2023), where the authors successfully solved the IRP, the split delivery vehicle routing problem (SDVRP), and the CVRP. In this version, longer sequences of customers can be inserted or removed from a route.

As described above, improvement heuristics can be very useful in solving vehicle routing problems. The exact RMIMs used by Vadseth et al. (2023) and Skålnes et al. (2023), are, for example, integral parts of some of the current state-of-the-art solution methods for the IRP, PRP, and SDVRP. Vehicle routing problems usually involve high costs and investments, and a small improvement to the objective function can result in significant savings. A natural question then, which is also the research question of this paper, is whether the RMIM proposed in Skålnes et al. (2023) is efficient for solving other types of vehicle routing problems. Moreover, can they be used to improve solutions found by other solution methods?

A common characteristic for the problems where exact RMIMs have been successful (PRP, IRP, and SDVRP) is that each customer can be visited multiple times, by different vehicles, and the quantities delivered to the customer on these visits need to meet the total demand. These types of constraints are often referred to as *inter-route* constraints since they involve multiple vehicle routes. Conversely, *intra-route* constraints limit what constitutes a feasible route. In the case of the PRP, IRP, and SDVRP, the only intra-route constraint is the vehicle capacity which limits the total quantity delivered by a route, and is thus dependent on the set of customers visited by a route, but not on the sequence in which they are visited.

Sequence-dependent intra-route constraints include time windows, pairing and precedence relationships, load onboard the vehicle (if there are both pickups and deliveries made at the customers), among others. Such intra-route constraints present a substantial challenge for RMIMs as they complicate the potential modifications. The insertion of a customer into a route may depend on multiple factors, and the most cost-effective insertion position may not be feasible without several other route adjustments. Notably, the potential of RMIMs for vehicle routing problems with a higher degree of intra-route constraints has not been thoroughly explored in the published literature. Therefore, this study aims to contribute to filling the gap by formulating and evaluating RMIMs for vehicle routing problems where intra-route constraints play a significant role.

We look at problems where time is an important parameter. More specifically, we address the multi-trip vehicle routing problem (MTVRP), the pickup and delivery problem with time windows (PDPTW), and the VRPTW. These are all well-studied vehicle routing problems with several intra-route constraints and well-established benchmark instances. We propose a novel matheuristic for the MTVRP that finds the best known solution for 76 of 104 benchmark instances, 28 of which are new to the literature. Further, we demonstrate that our proposed RMIMs can improve the best known solutions for both the PDPTW and VRPTW by warm-starting with the current best solution. The RMIMs are able to improve 203 benchmark instances for the PDPTW and 20 for the VRPTW. This is an interesting finding, since multiple researchers armed with different solution methods have thoroughly examined these problems, and we demonstrate that RMIMs can improve solutions where existing methods have reached a local optimum or otherwise converged. Consequently, it is clear that these MIPs complement the search space of traditional methods. Therefore, our research shows that the vehicle routing community should look closer into RMIMs and that they are a valuable tool for finding better solutions to many types of vehicle routing problems.

The rest of the paper is organized as follows: In Section 2, we describe a general version of the RMIM presented in Skålnes et al. (2023) that serves as the basis for the problem-specific RMIMs presented later on. In Section 3, we introduce an RMIM for the MTVRP and a novel algorithm for solving the problem. Section 4 expands the RMIM to the VRPTW and PDPTW, and our computational results are provided in Section 5. Lastly, Section 6 provides our concluding remarks.

2 A general exact route modifying improvement model for routing problems

The general RMIM introduced in this section builds on the work presented in Vadseth et al. (2023) and Skålnes et al. (2023). Here, we present the sets, variables, functions, and constraints needed to formulate the RMIM independent of the specific routing problem. The presented RMIM can be used to solve routing problems with a set of customers $\mathcal{N}$, indexed by i, served by a fleet of vehicles $\mathcal{K}$, indexed by k, starting and ending their routes at a single depot denoted by node 0. The routing problems can be modeled on a graph $G = (\mathcal{N}_0, \mathcal{A})$, where $\mathcal{N}_0 = \mathcal{N} \cup \{0\}$ is the set of nodes and $\mathcal{A} = \{(i,j) \in \mathcal{N}_0 \times \mathcal{N}_0 \mid i \neq j\}$ is the set of arcs connecting the nodes in the graph. The parameter C_{ij} is defined as the cost of traversing arc (i,j).

The RMIM is a modified version of a route-based model for a routing problem where a binary variable $\lambda_r \in \{0,1\}$ indicates whether route r in the set of routes $\mathcal{R}$ is used or not. Here, a route is defined as a Hamiltonian cycle that visits a subset of customers in $\mathcal{N}$, starting and ending at the depot. Each route r has an associated parameter C_r^T defined as the transportation cost of the route. The vehicle routing problem can only be solved by combining the different routes in $\mathcal{R}$. However, unlike in a pure set-partitioning setting, the routes in $\mathcal{R}$ can be modified by inserting or removing customers in the proposed RMIM. All modifications to the original routes are calculated exactly in the objective function. This is true even in cases where multiple changes are made simultaneously.

To describe the RMIM, we introduce the following notation. The customers visited in route $r \in \mathcal{R}$ are represented by the set $\mathcal{N}_r$. We also introduce a set of clusters $\mathcal{C}$. Here, a cluster $c \in \mathcal{C}$ is an unordered subset of customers $c \subseteq \mathcal{N}$ that can be added to a predefined position in a route. Usually, it is not feasible to insert all nodes into every route; hence, we need to introduce the set $\mathcal{C}_r$ which denotes the clusters that can be inserted into route r. For example, if a node can only be visited once by the same route, we define $\mathcal{C}_r = \{c \in \mathcal{C} | c \cap \mathcal{N}_r = \emptyset\}$. The binary variable z_{cr} is also introduced, which takes value 1 if cluster $c \in \mathcal{C}_r$ is inserted into route $r \in \mathcal{R}$, and 0 otherwise. As mentioned above, cluster c is always inserted in a predefined position, $p_r^*(c)$, in route r. How this position is determined depends on the specific problem. Often, this can be the position that increases the total cost of the route the least. If this is the case, $p_r^*(c)$ is calculated as:

$$p_r^*(c) = \underset{p \in 1,\ldots,|\mathcal{N}_r|+1}{\arg\min} \{C^{SP}(i_r(p-1), c, i_r(p)) - C_{i_r(p-1), i_r(p)}\}, \qquad r \in \mathcal{R}, c \in \mathcal{C}_r. \tag{1}$$

Here, the function $C^{SP}(i_{start}, c, i_{end})$ gives the cost of the shortest path starting at node i_{start}, visiting all nodes in c, and ending at node i_{end}. Furthermore, the node placed in position p in route r is returned by the function $i_r(p)$. Note that for all routes $r \in \mathcal{R}$, the depot has both the first position 0 and the last position $|\mathcal{N}_r|+1$. The cost, C^I_{cr}, of adding cluster $c \in \mathcal{C}_r$ to route $r \in \mathcal{R}$ is calculated as:

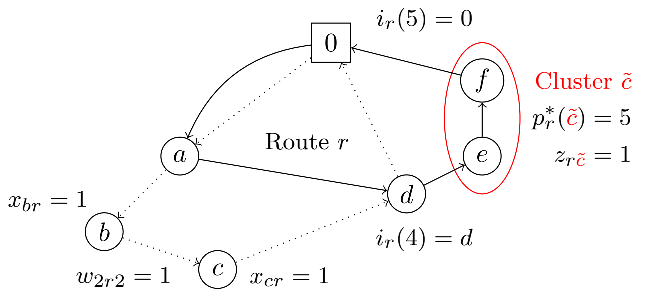
$$C^I_{cr} = C^{SP}(i_r(p^*_r(c)-1), c, i_r(p^*_r(c))) - C_{i_r(p^*_r(c)-1), i_r(p^*_r(c))}, \qquad r \in \mathcal{R}, c \in \mathcal{C}_r. \tag{2}$$

As seen above, the nodes in a cluster get an order when the cluster is coupled with a route and a specific position.

In addition, we introduce the binary variable $w_{pr\theta}$, which takes the value 1 if $\theta \in \Theta$ consecutive nodes starting from position $p \in \mathcal{P}^\theta_r$ are removed from route $r \in \mathcal{R}$. The set Θ is defined as $\Theta = \{1, \ldots, M\}$, where the parameter M denotes the maximum number of consecutive customers that can be removed. Let the set $\mathcal{P}_r$ denote all positions in route r. Moreover, the set $\mathcal{P}^\theta_r \subset \mathcal{P}_r = \{1, ..., |\mathcal{N}_r|+1\}$ denotes the positions, p, in route r where θ consecutive nodes starting from p can be removed. The cost savings of removing θ nodes starting from position p in route r is calculated as:

$$C^R_{\theta pr} = \sum_{p'=p}^{p+\theta} C_{i_r(p'-1), i_r(p')} - C_{i_r(p-1), i_r(p+\theta)}, \qquad r \in \mathcal{R}, \theta \in \Theta, p \in \mathcal{P}^\theta_r. \tag{3}$$

The introduced notation, including $i_r(p)$, $p^*_r(i)$, C^I_{cr} and $w_{pr\theta}$ is illustrated in Figure 1. Here, a route r, marked with dotted arrows, is modified by inserting and removing customers, and the resulting route is marked with solid arrows.



$$\begin{aligned} C^T_r &= C_{0a} + C_{ab} + C_{bc} + C_{cd} + C_{d0} \\ C^I_{\tilde{c}r} &= C^{SP}(i_r(4), \tilde{c}, i_r(5)) - C_{i_r(4), i_r(5)} \\ &= C_{de} + C_{ef} + C_{f0} - C_{d0} \\ C^R_{2r2} &= C_{ab} + C_{bc} + C_{cd} - C_{ad} \end{aligned}$$

$$\begin{aligned} 0 \to a \to d \to e \to f \to 0 : C^T_r + C^I_{\tilde{c}r} - C^R_{2r2} \\ = C_{0a} + C_{ad} + C_{de} + C_{ef} + C_{f0} \end{aligned}$$

Fig. 1 A route consisting of the depot and nodes a, b, c, and d is changed to a route consisting of the depot and nodes a, d, e, and f by inserting a cluster and removing two consecutive nodes. The cost of the route is then changed from $C_{0a} + C_{ab} + C_{bc} + C_{cd} + C_{d0}$ to $C_{0a} + C_{ad} + C_{de} + C_{ef} + C_{f0}$.

The objective function for the general RMIM can be formulated in the following way:

$$\min \sum_{r \in \mathcal{R}} C^T_r \lambda_r - \sum_{r \in \mathcal{R}} \sum_{\theta \in \Theta} \sum_{p \in P^\theta_r} C^R_{\theta pr} w_{pr\theta} + \sum_{r \in \mathcal{R}} \sum_{c \in \mathcal{C}_r} C^I_{cr} z_{cr}. \tag{4}$$

To keep the calculations of route modifications correct, we must impose some additional limitations. To ease readability, we introduce the binary variable x_{ir}, which takes the value 1 if node i is removed from route r (node i must be a part of route r), and 0 otherwise. The additional constraints are defined as follows:

$$\sum_{\theta \in \Theta} \sum_{p'=p-\theta+1}^{p} w_{p'r\theta} = x_{i_r(p)r}, \qquad r \in \mathcal{R}, p \in \mathcal{P}_r \setminus |\mathcal{N}_r| - 1, \tag{5}$$

$$\sum_{c \in \mathcal{C}_r : p^*_r(c) = p} z_{cr} \leq \lambda_r, \qquad r \in \mathcal{R}, p \in \mathcal{P}_r, \tag{6}$$

$$x_{i_r(p^*_r(c)-1)r} + x_{i_r(p^*_r(c))r} \leq 2(\lambda_r - z_{cr}) \qquad r \in \mathcal{R}, c \in \mathcal{C}_r, \tag{7}$$

$$x_{i_r(p-1)r} + x_{i_r(p+\theta)r} \leq 2(\lambda_r - w_{pr\theta}) \qquad \theta \in \Theta, r \in \mathcal{R}, p \in P^\theta_r. \tag{8}$$

Constraints (5) link the x_{ir} variables with the $w_{pr\theta}$ variables and state that a customer cannot be removed if one of the corresponding $w_{pr\theta}$ variables is not set to 1. Constraints (6) make sure that only one cluster can be inserted in position p in route r. Additionally, constraints (7) ensure that the nodes before and after the position where cluster c can be placed in route r cannot be removed if cluster c is added to the route. Constraints (8) are similar, but involve removed nodes. Here, it is stated that we cannot remove the nodes placed before or after a sequence of consecutive nodes if the nodes are removed. The last three constraints are needed since the cost calculations of removing or adding nodes include the nodes before or after the added/removed nodes. Hence, without these constraints, the cost calculations would not be correct and, instead, the model would approximate the costs of modifying the routes. Please note that in constraints (7) and (8), the variable x_{0r} is not defined (or equivalently its value is set equal to zero).

The given notation and constraints are always needed to formulate the RMIM in addition to the problem-specific constraints. However, some of the variables might be adjusted to handle relevant features of a specific problem.

3 The multi-trip vehicle routing problem

The multi-trip vehicle routing problem (MTVRP) is an extension of the CVRP. This is a single-period problem where each customer $i \in \mathcal{N}$ has a demand, D_i, that the depot must serve. Each vehicle $k \in \mathcal{K}$ starts and ends its routes at the depot and has a loading capacity of Q. Unlike the CVRP, in the MTVRP, each vehicle $k \in \mathcal{K}$ has a time limit, T^{max}, limiting its total driving time. Furthermore, vehicle k can make multiple trips (routes) as long as the combined driving time of all the trips transversed by k is at most T^{max}. The cost of using arc (i, j) is commonly set equal to the driving time, i.e., $C_{ij} = T_{ij}$ for every arc. For more details on the MTVRP, we refer to the survey paper by Cattaruzza et al. (2018).

3.1 The RMIM for the MTVRP

To formulate an RMIM for the MTVRP, we must add an index k to route variables λ_r to specify which vehicle k uses route r. In addition, we must introduce the variable t_{rk}, which is the total time vehicle k uses on route r. This variable is introduced to avoid adding the k index on variables z_{cr}, x_{ir}, and $w_{pr\theta}$. The parameter T^T_r denotes the original travel time of route r, while T^I_{cr} and $T^R_{pr\theta}$ denote the change in travel time when inserting cluster c or removing θ consecutive nodes from route r. All other notation remains the same as is described for the general version in Section 2. Furthermore, $p^*_r(c)$ is defined as in Equation (1), C^I_{cr} is defined as in Equation (2), and $\mathcal{C}_r = \{c \in \mathcal{C} | c \cap \mathcal{N}_r = \emptyset\}$ since the only limitation we have regarding insertions is that each customer can only be visited once. Lastly, we introduce the demand of cluster c as $D^C_c = \sum_{i \in c} D_i$. We can then define the RMIM for the MTVRP in the following way:

$$\min \sum_{r \in \mathcal{R}} \sum_{k \in \mathcal{K}} C^T_r \lambda_{rk} - \sum_{r \in \mathcal{R}} \sum_{\theta \in \Theta} \sum_{p \in P^\theta_r} C^R_{\theta pr} w_{pr\theta} + \sum_{r \in \mathcal{R}} \sum_{c \in \mathcal{C}_r} C^I_{cr} z_{cr} \tag{9}$$

$$
\begin{aligned}
&\text{Constraints (5),} && \\
&\sum_{c\in\mathcal{C}_r:p_r^*(c)=p} z_{cr} + x_{i_r(p)r} \le \sum_{k\in\mathcal{K}} \lambda_{rk}, && r\in\mathcal{R}, p\in\mathcal{P}_r, \quad (10)\\
&x_{i_r(p_r^*(c)-1)r} + x_{i_r(p_r^*(c))r} \le 2(\sum_{k\in\mathcal{K}} \lambda_{rk} - z_{cr}) && r\in\mathcal{R}, c\in\mathcal{C}_r, \quad (11)\\
&x_{i_r(p-1)r} + x_{i_r(p+\theta)r} \le 2(\sum_{k\in\mathcal{K}} \lambda_{rk} - w_{pr\theta}) && \theta\in\Theta, r\in\mathcal{R}, p\in P_r^\theta. \quad (12)\\
&\sum_{r\in\mathcal{R}:i\in\mathcal{N}_r} (\sum_{k\in\mathcal{K}} \lambda_{rk} - x_{ir}) + \sum_{r\in\mathcal{R}} \sum_{c\in\mathcal{C}_r:i\in c} z_{cr} = 1, && i\in\mathcal{N}, \quad (13)\\
&\sum_{i\in\mathcal{N}_r} D_i(\lambda_{rk} - x_{ir}) + \sum_{c\in\mathcal{C}_r} D_c^C z_{cr} \le Q, && r\in\mathcal{R}, k\in\mathcal{K}, \quad (14)\\
&\sum_{r\in\mathcal{R}} t_{rk} \le T^{max}, && k\in\mathcal{K}, \quad (15)\\
&T_r^T - \sum_{\theta\in\Theta} \sum_{p\in\mathcal{P}_r^\Theta} T_{pr\theta}^R w_{pr\theta} + \sum_{c\in\mathcal{C}_r} T_{cr}^I z_{cr} \le t_{rk} + T^{max}(1-\lambda_{rk}), && r\in\mathcal{R}, k\in\mathcal{K}, \quad (16)\\
&\sum_{k\in\mathcal{K}} \lambda_{rk} \le 1, && r\in\mathcal{R}, \quad (17)\\
&\lambda_{rk}\in\{0,1\}, && r\in\mathcal{R}, k\in\mathcal{K}, \quad (18)\\
&z_{cr}\in\{0,1\}, && c\in\mathcal{C}_r, r\in\mathcal{R}, \quad (19)\\
&x_{ir}\in\{0,1\}, && i\in\mathcal{N}_r, r\in\mathcal{R}, \quad (20)\\
&w_{pr\theta}\in\{0,1\}, && \theta\in\Theta, r\in\mathcal{R}, p\in\mathcal{P}_r^\theta, \quad (21)\\
&t_{rk}\ge 0, && r\in\mathcal{R}, k\in\mathcal{K}. \quad (22)
\end{aligned}
$$

The objective function (9) minimizes transportation costs. Constraints (10), (11), and (12) are adaptions of constraints (6), (7), and (8) accounting for the k index in the variables λ, while constraints (13) state that a customer can only be visited once. Furthermore, constraints (14) make sure that the vehicle capacity is respected. Constraints (15) state that the combined driving times of the routes a vehicle traverses cannot exceed the time limit, while constraints (16) control the time a vehicle spends on each route. Constraints (17) ensure that a route can only be used once. Finally, constraints (18) - (22) state the variable domains.

3.2 A novel matheuristic for the MTVRP

The MIP described above is small enough to be solved multiple times given that the route set $\mathcal{R}$ is not too large. The standard benchmark set of instances for the MTVRP, introduced by Taillard et al. (1996), allows infeasible solutions (the vehicle driver is allowed to have overtime). Here, each unit of overtime is multiplied by 2 to account for the associated costs. However, a solution without overtime is always considered superior to one with overtime for these instances, even if the transportation costs are higher. The RMIM can be updated to handle this set of instances by introducing the overtime variable $o_k \ge 0$, which represents the overtime induced by vehicle k. We must also add the term $\sum_{k\in\mathcal{K}} 2o_k$ to the objective function. In addition, constraints (18) must be rewritten to:

$$\sum_{r \in R} t_{rk} \le T^{max} + o_k, \qquad k \in \mathcal{K}. \tag{23}$$

Lastly, T^{max} is no longer a valid big-M constant in constraints (16). Hence, the constraints must be rewritten to:

$$T_r^T \lambda_{rk} - \sum_{\theta \in \Theta} \sum_{p \in \mathcal{P}_r^\Theta} T_{pr\theta}^R w_{pr\theta} + \sum_{c \in \mathcal{C}_r} T_{cr}^I z_{cr} \le t_{rk} + \sum_{c \in \mathcal{C}_r} T_{cr}^I (1 - \lambda_{rk}), \qquad r \in \mathcal{R}, k \in \mathcal{K}. \tag{24}$$

By making these changes, we can introduce the following simple yet efficient matheuristic. First, we solve the MTVRP instance as a CVRP instance. The routes given from the CVRP solution form the route set $\mathcal{R}$, which is later used in the RMIM. By assigning the routes in $\mathcal{R}$ to the different vehicles in $\mathcal{K}$ we already have a feasible solution to the original MTVRP instance (although it might include overtime). This connection has been exploited in the heuristics proposed by Taillard et al. (1996) and Petch and Salhi (2003), where several CVRP solutions are created before a bin packing problem is solved to assign routes to vehicles. In our matheuristic, we instead solve the RMIM to assign routes to each vehicle. It can also make improvements to the routes in $\mathcal{R}$. The RMIM is solved for several iterations, where we update $\mathcal{R}$ between each iteration and set it equal to the routes of the current solution. In each iteration, we warm-start the MIP with the current solution. The matheuristic is summarized in Algorithm 1.

Algorithm 1 A matheuristic for MTVRP

```
 1: Input: MTVRP instance
 2: $Sol_{best} = \infty$
 3: $Sol_{CVRP}$ = CVRPSolver(MTVRP instance)
 4: $\mathcal{R} = getRoutes(Sol_{CVRP})$
 5: for $it$ iterations do
 6:     $Sol$ = RMIM($\mathcal{R}$)
 7:     if $Sol < Sol_{best}$ then
 8:         $\mathcal{R} = getRoutes(Sol)$
 9:         $Sol_{best} = Sol$
10:     else
11:         Break
12:     end if
13: end for
14: Return $Sol_{best}$
```

Here, Sol_{best} is the current best solution and Sol_{CVRP} is the solution produced by the CVRP solver. This work uses the genetic algorithm presented in Vidal et al. (2012) and the open-source implementation described in Vidal (2022). The function *getRoutes* returns the routes associated with a solution to either the CVRP or the MTVRP.

To populate the set of clusters $\mathcal{C}$, in this work we have used a k-means algorithm (Ahmed et al., 2020) to produce clusters with cardinality greater than 1. Here, a k-value $= \lfloor |\mathcal{N}|/2 \rfloor$ determines the number of clusters. If we define the clusters produced by the k-means algorithm as $\mathcal{C}^k$, we can set $\mathcal{C} = \left\{\bigcup_{i \in \mathcal{N}} \{i\} \bigcup c \in \mathcal{C}^k : |c| \leq H\right\}$. The parameter H is the largest cluster size we want to include in the model, and it is set to 4 in this work.

4 Vehicle routing problems with time windows

In this section, we analyze two other routing problems, namely, the vehicle routing problem with time windows (VRPTW) and the pickup and delivery problem with time windows (PDPTW). Both the VRPTW and the PDPTW are extensions of the CVRP where a single depot, denoted by node 0, must serve a set of nodes $\mathcal{N}$ in a single period with a fleet of vehicles, $\mathcal{K}$, where each vehicle $k \in \mathcal{K}$ has a capacity Q and can perform one route. Similar to the CVRP and MTVRP, both the VRPTW and PDPTW can be modeled on a graph $G = (\mathcal{N}_0, \mathcal{A})$. However, each node $i \in \mathcal{N}_0$ has a time window $[A_i, B_i]$ in which the node must be served. A_i is the earliest start of service time customer i and B_i is the latest. In addition, each customer i has a service time S_i that dictates the time needed to serve the customer. In the VRPTW, each customer i has a demand, D_i, that must be served. This is similar to the CVRP. In the PDPTW, however, the customers, $i \in \mathcal{N}$, are split into two disjoint subsets $\mathcal{N}^P$ and $\mathcal{N}^D$, where $\mathcal{N}^P \cup \mathcal{N}^D = \mathcal{N}$. Here, $\mathcal{N}^P = \{1, ..., n\}$ denotes pickup nodes and $\mathcal{N}^D = \{n+1, ..., 2n\}$ denotes delivery nodes. A pickup node, $i \in \mathcal{N}^{\mathcal{P}}$, has a given demand D_i that must be delivered to a corresponding delivery node $(i+n) \in \mathcal{N}^{\mathcal{D}}$ with demand $D_{(i+n)} = -D_i$. Both node i and node $(i+n)$ must be served by the same vehicle since we do not allow transshipment and node i must be served before node $(i+n)$. For more details on the VRPTW and PDPTW, we refer to Toth and Vigo (2014).

4.1 The RMIM for the VRPTW

Due to the time windows, we must introduce some new notation to formulate an RMIM for the VRPTW. First, for readability, we introduce the binary variable u_{pr}, which takes the value 1 if a node is not inserted or removed from position p in route r, and 0 otherwise. Second, we must introduce the variable t_{pr} which denotes the time a vehicle arrives at position p in route r. To account for inserted clusters, we must introduce some new time windows. If cluster c is inserted into route r, the time window $[A_{cr}, B_{cr}]$ represents the time window within which node $i = i_r(p_r^*(c) - 1)$ located just before c on route r must be served. When visiting nodes in a cluster, we may encounter waiting times since the vehicle can arrive before the time windows for one or multiple nodes start. We can minimize the waiting time when setting the time windows and thus set the traveling time through a cluster equal to a constant. The time window $[A_{cr}, B_{cr}]$ for inserting cluster c in route r is defined in Algorithm 2, where $c_r(p)$ denotes the node placed in position $p \in 1, ..., |c|$ in cluster c when inserted in route r.

Further, we must introduce additional notation to update the time spent when arriving at different positions in route r. For the sake of readability, we introduce the parameter T^s_{ijr} which denotes the travel time between nodes i and j, including the service time of node i in route r, i.e., $T^s_{ijr} = S_i + T_{ij}$. This parameter is needed if neither node i nor node j is removed from route r or if the nodes located between nodes i and j are removed. However, if cluster c is inserted in route r we need to keep track of the time spent between arriving at the node in front of c $(i_r(p_r^*(c) - 1))$ and arriving at the node located after c $(i_r(p_r^*(c)))$. This relationship is denoted by the parameter T^c_{cjr}, which includes the service time of the node

Algorithm 2 Time window for customer located just before cluster c in route r

```
1: $A_{cr} = A_{i_r(p^*_r(c)-1)}$
2: $B_{cr} = B_{i_r(p^*_r(c)-1)}$
3: $time = S_{i_r(p^*_r(c)-1)} + T_{i_r(p^*_r(c)-1)c_r(1)}$
4: for $j = 1$ to $|c|$ do
5:     if $A_{cr} + time \geq B_{c_r(j)}$ then
6:         $A_{cr} = B_{cr} = 0$
7:         Break
8:         *Stop as it is infeasible to insert cluster c*
9:     end if
10:    if $B_{cr} + time \leq A_{c_r(j)}$ then
11:        $A_{cr} = B_{cr}$
12:        Break
13:        *Stop as we have forced waiting time*
14:    end if
15:    $A_{cr} = \max\{A_{cr}, A_{c_r(j)} - time\}$
16:    $B_{cr} = \min\{B_{cr}, B_{c_r(j)} - time\}$
17:    if $j \neq |c|$ then
18:        $time += S_{c_r(j)} + T_{c_r(j)c_r(j+1)}$
19:    end if
20: end for
```

in front of c, the traveling time needed to visit all nodes in cluster c, the service time for all nodes in c, and the traveling time from the last node in c to node j. Please note that this relationship is a constant due to the definition of the time window $[A_{cr}, B_{cr}]$. Parameter T^c_{cjr} is defined in Algorithm 3:

Algorithm 3 Travel times including service time when inserting cluster c in route r

```
1: $T^c_{cjr} = A_{cr}$
2: $T^c_{cjr} += S_{i_r(p^*_r(c)-1)}$
3: $T^c_{cjr} += T_{i_r(p^*_r(c)-1)c_r(1)}$
4: for $j = 1$ to $|c|$ do
5:     $T^c_{cjr} = \max\{T^c_{cjr}, A_{c_r(j)}\}$
6:     $T^c_{cjr} += S_{c_r(j)}$
7:     if $j \neq |c|$ then
8:         $T^c_{cjr} += T_{c_r(j)c_r(j+1)}$
9:     end if
10: end for
11: $T^c_{cjr} += T_{c(|c|)i}$
12: $T^c_{cjr} -= A_{cr}$
```

The predefined position where we can insert nodes and the order in which the nodes are visited in a cluster is complicated by time windows. For the CVRP and MTVRP, the order is determined by the shortest path function $C^{SP}(i_{start}, c, i_{end})$ defined in Section 2. However, the shortest path might not be feasible in the VRPTW due to time window constraints. Section 4.3 describes how the insertion position of cluster c in route r, $p_r^*(c)$, and the insertion cost, C^I_{cr}, are defined for the VRPTW. The RMIM for the VRPTW can be defined in the following way:

$$\min \sum_{r\in\mathcal{R}} C_r^T \lambda_r - \sum_{r\in\mathcal{R}} \sum_{\theta\in\Theta} \sum_{p\in P_r^\theta} C^R_{\theta pr} w_{pr\theta} + \sum_{r\in\mathcal{R}} \sum_{c\in\mathcal{C}_r} C^I_{cr} z_{cr} \tag{25}$$

Constraints (5) – (8),

$$\sum_{r\in\mathcal{R}:i\in\mathcal{N}_r} (\lambda_r - x_{ir}) + \sum_{r\in\mathcal{R}} \sum_{c\in\mathcal{C}_r:i\in c} z_{cr} = 1, \qquad i \in \mathcal{N}, \tag{26}$$

$$\sum_{i\in\mathcal{N}_r} D_i(\lambda_r - x_{ir}) + \sum_{c\in\mathcal{C}_r} D^C_c z_{cr} \leq Q\lambda_r, \qquad r \in \mathcal{R}, \tag{27}$$

$$\sum_{c\in\mathcal{C}_r:p_r^*(c)=p} z_{cr} + x_{i_r(p)r} + u_{pr} = \lambda_r, \qquad r \in \mathcal{R}, p \in \mathcal{P}_r, \tag{28}$$

$$t_{(p-1)r} + T^s_{i_r(p-1)i_r(p)r} u_{pr} + \sum_{c\in\mathcal{C}_r:p_r^*(c)=p} T^c_{ci_r(p)r} z_{cr} \leq t_p, \qquad r \in \mathcal{R}, p \in \mathcal{P}_r, \tag{29}$$

$$+ \sum_{\theta\in\Theta} (T^s_{i_r(p-\theta-1)i_r(p)r} - T^s_{i_r(p-1)i_r(p)r}) w_{(p-\theta)r\theta}$$

$$\sum_{c\in\mathcal{C}_r:p_r^*(c)=p+1} (B_{cr} - A_{i_r(p)}) z_{cr} + A_{i_r(p)}(\lambda_r - x_{x_{i_r(p)r}}) \leq t_{pr}, \qquad r \in \mathcal{R}, p \in \mathcal{P}_r \setminus |\mathcal{N}_r| - 1, \tag{30}$$

$$\sum_{c\in\mathcal{C}_r:p_r^*(c)=p+1} (A_{cr} - B_{i_r(p)}) z_{cr} + B_0 x_{i_r(p)r} \geq t_{pr}, \qquad r \in \mathcal{R}, p \in \mathcal{P}_r \setminus |\mathcal{N}_r| - 1, \tag{31}$$

$$+ B_{i_r(p)}(\lambda_r - x_{i_r(p)r})$$

$$A_0(\lambda_r - \sum_{c\in\mathcal{C}_r:p_r^*(c)=1} z_{cr}) + \sum_{c\in\mathcal{C}_r:p_r^*(c)=1} A_{cr} z_{cr} \leq t_{0r}, \qquad r \in \mathcal{R}, \tag{32}$$

$$B_0 \geq t_{(|\mathcal{N}_r|+1)r}, \qquad r \in \mathcal{R}, \tag{33}$$

$$\lambda_r \in \{0,1\}, \qquad r \in \mathcal{R}, \tag{34}$$

$$x_{ir} \in \{0,1\}, \qquad r \in \mathcal{R}, i \in \mathcal{N}_r, \tag{35}$$

$$z_{cr} \in \{0,1\}, \qquad r \in \mathcal{R}, c \in \mathcal{C}_r, \tag{36}$$

$$w_{pr\theta} \in \{0,1\}, \qquad \theta \in \Theta, r \in \mathcal{R}, p \in \mathcal{P}_r^\theta, \tag{37}$$

$$u_{pr} \in \{0,1\}, \qquad r \in \mathcal{R}, p \in \mathcal{P}_r. \tag{38}$$

The objective function (25) minimizes transportation costs, while constraints (26) state that all customers must be served. Constraints (27) ensure that the vehicle capacity is respected. Moreover, constraints (28) make sure that a node is inserted or removed or that nothing happens at a position p in route r. Constraints (29) ensure time management, while constraints (30) and (31) state the time windows. In addition, (32) and (33) state the time windows for the depot. Constraints (34) - (38) define the variable domains.

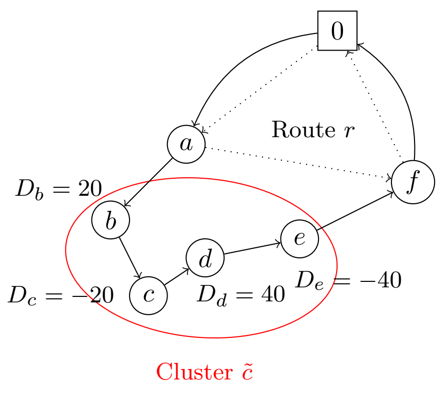
4.2 The RMIM for the PDPTW

To formulate the RMIM for the PDPTW we use most of the same notation as for the VRPTW. However, we need to introduce the variable q_{pr} that denotes the quantity onboard a vehicle after visiting position p in route r. Additionally, we must introduce the parameter F_{cr}, which is the largest additional load a vehicle carries while visiting cluster c in route r compared to the load onboard after visiting the node before c. F_{cr} is defined in Algorithm 4:

Algorithm 4 The largest additional load carried while visiting cluster c in route r

```
1: $load = 0$
2: $F_{cr} = 0$
3: for $j = 1$ to $|c|$ do
4:     $load$ += $D_{c_r(j)}$
5:     if $load > F_{cr}$ then
6:         $F_{cr} = load$
7:     end if
8: end for
```

Parameter F_{cr} is needed to ensure that the vehicle capacity is not breached while serving one of the customers in cluster c. The calculation of F_{cr} is illustrated in Figure 2:



Node pairs: (a,f), (b,c), (d,e)

Step 0: $load = 0$ and $F_{\tilde{c}r} = 0$

Step 1: $load = 20$ and $F_{\tilde{c}r} = 20$

Step 2: $load = 0$

Step 3: $load = 40$ and $F_{\tilde{c}r} = 40$

Step 4: $load = 0$

Fig. 2 A route consisting of the depot and nodes a, and f is changed to a route consisting of the depot and nodes a, b, c, d, e, and f by inserting a cluster. The additional load the vehicle carries while serving the node is shown to be 40.

Section 4.3 describes how the insertion position of cluster c in route r, $p_r^*(c)$, and the insertion cost, C_{cr}^I, are defined for the PDPTW. The RMIM for the PDPTW can be defined in the following way:

$$\min \sum_{r \in \mathcal{R}} C_r^T \lambda_r - \sum_{r \in \mathcal{R}} \sum_{\theta \in \Theta} \sum_{p \in P_r^\theta} C_{\theta pr}^R w_{pr\theta} + \sum_{r \in \mathcal{R}} \sum_{c \in \mathcal{C}_r} C_{cr}^I z_{cr} \tag{39}$$

Constraints $(5)-(8), (26), (28)-(33)$,

$$x_{ir} = x_{(i+n)r}, \qquad i \in \mathcal{N}^P, r \in \mathcal{R}, \tag{40}$$

$$\sum_{c \in \mathcal{C}_r : i \in c} z_{cr} = \sum_{c \in \mathcal{C}_r : (i+n) \in c} z_{cr}, \qquad i \in \mathcal{N}^P, r \in \mathcal{R}, \tag{41}$$

$$q_{(p-1)r} + D_{i_r(p)} u_{pr} + \sum_{c \in \mathcal{C}_r : p^*_r(c) = p} z_{cr}(D^C_c + D_{i_r(p)}) \leq q_{pr}, \qquad r \in \mathcal{R}, p \in \mathcal{P}_r, \tag{42}$$

$$q_{pr} + \sum_{c \in \mathcal{C}_r : p^*_r(c) = p+1} z_{cr} F_{cr} \leq Q \lambda_r \qquad r \in \mathcal{R}, p \in \mathcal{P}_r \setminus |\mathcal{N}_r| - 1, \tag{43}$$

$$q_{pr} \geq 0, \qquad r \in \mathcal{R}, p \in \mathcal{P}_r \setminus |\mathcal{N}_r| - 1, \tag{44}$$

$$q_{pr} = 0, \qquad r \in \mathcal{R}, p \in \{0, |\mathcal{N}_r| - 1\}, \tag{45}$$

$$\lambda_r \in \{0,1\}, \qquad r \in \mathcal{R}, \tag{46}$$

$$x_{ir} \in \{0,1\}, \qquad r \in \mathcal{R}, i \in \mathcal{N}_r, \tag{47}$$

$$z_{cr} \in \{0,1\}, \qquad r \in \mathcal{R}, c \in \mathcal{C}_r, \tag{48}$$

$$w_{pr\theta} \in \{0,1\}, \qquad \theta \in \Theta, r \in \mathcal{R}, p \in \mathcal{P}^{\theta}_r. \tag{49}$$

The objective function (39) minimizes transportation costs. Furthermore, constraints (40) and (41) state that the same vehicle must serve a pickup and delivery pair. Constraints (42) are load balancing constraints, while constraints (43) - (45) ensure that the vehicle capacity is respected. Constraints (46) - (49) state the variable domains. The precendence between node i and node $i + n$ is taken care of by the node order in cluster c and by only accepting routes with the correct precedencee order in route set $\mathcal{R}$.

The formulation can be further strengthened by realizing that the load onboard a vehicle must be at least equal to D_i if node i is a pickup node, and not greater than $Q - (-D_i)$ if node i is a delivery node. We can then update constraints (43) and (44):

$$q_{pr} + \sum_{c \in \mathcal{C}_r : p^*_r(c) = p+1} z_{cr} F_{cr} \leq Q \lambda_r + \qquad r \in \mathcal{R}, p \in \mathcal{P}_r \setminus |\mathcal{N}_r| - 1, \tag{50}$$

$$(\lambda_r - x_{i(p)} - \sum_{c \in \mathcal{C}_r : p^*_r(c) = p+1} z_{cr}) \min\{0, D_{i(p)}\},$$

$$q_{pr} \geq \max\{0, D_{i(p)}\}(\lambda_r - x_{i(p)}), \qquad r \in \mathcal{R}, p \in \mathcal{P}_r \setminus |\mathcal{N}_r| - 1. \tag{51}$$

4.3 The RMIM for the VRPTW and PDPTW in an algorithmic setting

For the VRPTW and PDPTW, we cannot use the definitions for the insertion position of cluster c in route r, $p^*_r(c)$ and the insertion cost, C^I_{cr}, described in Section 2. If we did, most of our insertions would be infeasible due to time window constraints. For example, the cheapest position to insert a cluster could be between two nodes with time windows that end before we can even serve some of the nodes in c. Therefore, we must consider time windows when determining $p^*_r(c)$ and the visiting order of nodes in c for the VRPTW and PDPTW. We achieve this by updating Equation (1):

$$p^*_r(c) = \underset{p \in 1,\ldots,|\mathcal{N}_r|+1}{\arg\min} \{C^{SPTW}(i_r(p-1), c, i_r(p)) - C_{i_r(p-1), i_r(p)}\}, \qquad r \in \mathcal{R}, c \in \mathcal{C}_r. \tag{52}$$

Here, the algorithm $C^{SPTW}(i_{start}, c, i_{end})$ returns the shortest path starting at node i_{start}, visiting all nodes in c, and ending at node i_{end} while respecting the time windows for all nodes in route r and cluster c. If it is impossible to add cluster c between nodes i_{start} and i_{end} without making the route infeasible, given that everything else stays the same, the algorithm returns an error. If all positions $p \in 1, ..., |\mathcal{N}_r| + 1$ return an error, it is not possible to insert cluster c into route r, and the upper bound for z_{cr} will thus be 0 and C_r will not include cluster c. The cost, C^I_{cr}, of adding cluster $c \in \mathcal{C}$ to route $r \in \mathcal{R}$ can, for the VRPTW and PDPTW, be calculated as:

$$C^I_{cr} = C^{SPTW}(i_r(p^*_r(c) - 1), c, i_r(p^*_r(c))) - C_{i_r(p^*_r(c)-1), i_r(p^*_r(c))}, \qquad r \in \mathcal{R}, c \in \mathcal{C}_r. \tag{53}$$

Although $C^{SPTW}(i_r(p - 1), c, i_r(p))$ deals with the time windows, it is a rather limiting approach since a better position might be feasible if we remove something else in the route. Yet, it is very complex to determine a priori which nodes we can remove to make the insertion feasible. To deal with this problem, we use the following approach. For all routes in route set $\mathcal{R}$, we make several duplicates where we remove a single customer in the VRPTW or a single node pair in the PDPTW. For example, route 0-1-2-3-0 in the VRPTW will have the following three duplicates: 0-2-3-0, 0-1-3-0, and 0-1-2-0. Although this comes at the expense of a larger route set $\mathcal{R}$, this approach opens up additional insertions that would not be possible otherwise.

To populate clusters in the VRPTW, we use the same method as for the MTVRP. For the PDPTW, we use single-node clusters and clusters consisting of a single node pair i and $i + n$. If a single-node cluster is added to a route, the single-node cluster containing the associated delivery or pickup node must also be added to the same route as described in constraints (40)-(41).

5 Computational study

In this section, we present our computational results. We have tested the proposed matheuristic on a set of benchmark instances for the MTVRP and warm-started the RMIM for the PDPTW and VRPTW with the best known solutions found in the literature for benchmark instances for the PDPTW and VRPTW. To avoid spending an excessively long time closing the dual gap when solving MIPs, we have set a time limit on all RMIMs. For simplicity, this time limit is set to 1,800 seconds for all instances of the MTVRP, 3,600 seconds for all instances of the VRPTW, and 3,600 seconds and 10,800 seconds, respectively, for instances with under or over 1,000 customers for the PDPTW. All computational experiments were run on a 12-core Intel E5-2670v3 processor clocked at 2.3 GHz and 64 GB of RAM. All algorithms are coded in C++, and the commercial solver Gurobi 9.5.1 has been used. Detailed solution files are publicly available at http://axiomresearchproject.com/

5.1 MTVRP

The well-established set of benchmark instances for the MTVRP was introduced by Taillard et al. (1996). They are constructed from the CMT1-CMT5 and CMT11-CMT12 instances proposed by Christofides et al. (1979) and instances F11-F12 proposed by Fisher (1994) for the CVRP. Here, all distances are Euclidean. Several MTVRP instances were constructed from each of the 12 CVRP instances, with many different $|\mathcal{K}|$ and two different versions of $\mathcal{T}^{max}$. For version 1, $\mathcal{T}^{max} = [1.05z^*/|\mathcal{K}|]$, and $\mathcal{T}^{max} = [1.1z^*/|\mathcal{K}|]$ for version 2. Here, z^* is the solution value found by Rochat and Taillard (1995) for the original CVRP instance. This results in a total of 104 benchmark instances. The distances are not rounded to integers, as in the original CVRP instances.

The benchmark instances proposed in Taillard et al. (1996) allow overtime. This means that $\mathcal{T}^{max}$ can be breached; however, this results in overtime which is penalized by a factor of 2. Please note that a solution without overtime is always considered superior to one with overtime, even if the total costs are higher. This is handled by adding an additional large cost to the overtime variable in this work. All best known solutions to the benchmark instances were summarized in Cattaruzza et al. (2018), and, to the best of our knowledge, no published work has reported improved solutions since then. Thus, we compare our solutions with the best known solutions and not with individual solution methods. The comparison is summarized in Table 1.

Here, "Best Known" refers to the best known solution value reported in Cattaruzza et al. (2018). If the solution is optimal, it is denoted by an asterisk. "Overtime" indicates whether the solution includes overtime or not. Please note that a solution can be both optimal and include overtime. In this case, there are no feasible solutions without overtime. The column "This paper" refers to the solution value found by our method, while the "Overtime" column to the right denotes whether our solution includes overtime. "RMIM" refers to how much better our final solution is than the solution found using a CVRP and bin packing solver. In the "This paper" column, best known solutions are indicated in **bold**. We can find the best known solution for 76 of 104 instances. Here, 28 of them are new to the literature. We can see that the majority of the best known solutions, 73 of 76 instances, were not improved by the RMIM. This means that the best known solution consists purely of the routes created by solving the instance as a CVRP. For the solutions that were improved by the RMIM, the average improvement was 1.69% for version 1 and 1.32% for version 2. Note that the improvement can be negative if the RMIM turns a solution with overtime into a solution without.

5.2 VRPTW

For the VRPTW, there exist two well-known sets of benchmark instances. The first one was proposed by Solomon (1987) and consists of 25, 50, or 100 customers. The second set was proposed by Gehring and Homberger (1999) and consists of 200, 400, 600, 800, or 1,000 customers. Both sets use Euclidean distances, and the nodes in each instance are clustered (C), randomly distributed (R), or a mix (RC). In addition, previous works have used both a hierarchical objective where the primary goal is to reduce the number of vehicles, and a pure cost objective when solving the benchmark instances. We have used the latter, and an overview of all best known solutions for this version can be found in the CVRPLib (Uchoa et al., 2017) webpage (http://vrp.atd-lab.inf.puc-rio.br/index.php/en/). All instances in Solomon (1987) have been solved to optimality, while 96 of 300 solutions are proven optimal for the benchmark instances of Gehring and Homberger (1999).

By warm-starting the RMIM for the VRPTW, we improved 20 of the 204 instances not solved to proven optimality. This translates to 9% of the instances. As seen in Table 2, several of the improvements are rather marginal in terms of solution value. Considering that the VRPTW has received much attention and numerous solution methods exist, these are still very good numbers. Only one of the 20 improved instances has clustered nodes, and 13 have randomly distributed nodes. This indicates that the RMIM works better on non-clustered instances or that other solution methods perform better on clustered instances, and a larger number of optimal solutions have been found for these instances. A possible explanation for the former is that the clusters used in the RMIM need to be larger to be efficient on clustered instances and that replacing and inserting subparts of each cluster is not fruitful.

		Version 1					Version 2				
Instance	|K|	Best known	Overtime	This paper	Overtime	RMIM	Best known	Overtime	This paper	Overtime	RMIM
CMT1	1	524.61*	No	**524.61**	No	0.00 %	524.61*	No	**524.61**	No	0.00 %
	2	533.00*	No	546.68	No	3.87 %	529.85*	No	**529.85**	No	2.00 %
	3	569.54	Yes	574.57	Yes	4.37 %	552.68	No	557.51	Yes	1.98 %
	4	564.07	Yes	618.20	Yes	4.01 %	546.29*	No	605.85	Yes	4.14 %
CMT2	1	835.26*	No	**835.26**	No	0.00 %	835.26*	No	**835.26**	No	0.00 %
	2	835.26*	No	**835.26**	No	0.00 %	835.26*	No	**835.26**	No	0.00 %
	3	835.26*	No	**835.26**	No	0.00 %	835.26*	No	**835.26**	No	0.00 %
	4	835.26*	No	**835.26**	No	0.00 %	835.26*	No	**835.26**	No	0.00 %
	5	835.80*	No	838.18	No	0.04 %	835.26*	No	**835.26**	No	0.00 %
	6	858.58	No	886.93	Yes	3.38 %	839.22*	No	862.00	No	3.14 %
	7	866.58	Yes	928.36	Yes	0.04 %	844.70	No	904.13	Yes	0.06 %
CMT3	1	826.14*	No	**826.14**	No	0.00 %	826.14	No	**826.14**	No	0.00 %
	2	826.14*	No	**826.14**	No	0.00 %	826.14	No	**826.14**	No	0.00 %
	3	826.14*	No	**826.14**	No	0.00 %	826.14	No	**826.14**	No	0.00 %
	4	829.45	No	832.22	Yes	0.00 %	826.14	No	**826.14**	No	0.00 %
	5	832.89	No	930.34	Yes	0.61 %	832.34	No	867.53	No	3.62 %
	6	836.22	No	893.79	Yes	0.91 %	834.35	No	866.42	No	2.22 %
CMT4	1	1031.00	No	**1028.42**	No	0.00 %	1031.07	No	**1028.42**	No	0.00 %
	2	1031.07	No	**1028.42**	No	0.00 %	1030.45	No	**1028.42**	No	0.00 %
	3	1028.42	No	**1028.42**	No	0.00 %	1031.59	No	**1028.42**	No	0.00 %
	4	1031.10	No	**1028.42**	No	0.00 %	1031.07	No	**1028.42**	No	0.00 %
	5	1031.07	No	**1028.42**	No	0.00 %	1030.86	No	**1028.42**	No	0.00 %
	6	1034.61	No	1051.23	No	-0.16 %	1030.45	No	1031.07	No	-0.14 %
	7	1068.59	No	1076.55	Yes	0.16 %	1036.08	No	1071.52	No	-3.31 %
	8	1056.54	No	1084.10	Yes	3.76 %	1044.32	No	1065.13	No	2.33 %
CMT5	1	1302.43	No	**1291.45**	No	0.02 %	1299.86	No	**1291.45**	No	0.02 %
	2	1302.15	No	**1291.45**	No	0.00 %	1305.35	No	**1291.45**	No	0.00 %
	3	1301.29	No	**1291.45**	No	0.00 %	1301.03	No	**1291.45**	No	0.00 %
	4	1304.78	No	**1291.45**	No	0.00 %	1303.65	No	**1291.45**	No	0.00 %
	5	1300.02	No	**1291.45**	No	0.00 %	1300.62	No	**1291.45**	No	0.00 %
	6	1303.37	No	**1291.45**	No	0.00 %	1306.17	No	**1291.45**	No	0.00 %
	7	1309.40	No	**1291.45**	No	0.00 %	1031.54	No	**1291.45**	No	0.00 %
	8	1303.91	No	**1291.45**	No	0.00 %	1308.78	No	**1291.45**	No	0.00 %
	9	1307.93	No	**1304.24**	No	0.12 %	1307.25	No	**1291.45**	No	0.00 %
	10	1323.01	No	1314.10	Yes	0.00 %	1308.81	No	**1291.45**	No	0.00 %
CMT11	1	1042.11*	No	**1042.11**	No	0.00 %	1042.11*	No	**1042.11**	No	0.00 %
	2	1042.11*	No	**1042.11**	No	0.00 %	1042.11*	No	**1042.11**	No	0.00 %
	3	1042.11*	No	**1042.11**	No	0.00 %	1042.11*	No	**1042.11**	No	0.00 %
	4	1078.64	No	1055.09	Yes	0.00 %	1042.11*	No	**1042.11**	No	0.00 %
	5	1042.11*	No	**1042.11**	No	0.00 %	1042.11*	No	**1042.11**	No	0.00 %
CMT12	1	819.56*	No	**819.56**	No	0.00 %	819.56*	No	**819.56**	No	0.00 %
	2	819.56*	No	**819.56**	No	0.00 %	819.56*	No	**819.56**	No	0.00 %
	3	819.56*	No	**819.56**	No	0.00 %	819.56*	No	**819.56**	No	0.00 %
	4	819.56*	No	**819.56**	No	0.00 %	819.56*	No	**819.56**	No	0.00 %
	5	845.56	No	836.80	Yes	0.00 %	824.78*	No	820.78	Yes	0.00 %
	6	845.48	Yes	**845.48**	Yes	0.00 %	823.14*	No	824.16	No	-0.05 %
F11	1	241.97	No	**241.97**	No	0.00 %	241.97	No	**241.97**	No	0.00 %
	2	250.85	No	249.97	Yes	0.00 %	241.97	No	**241.97**	No	0.00 %
	3	256.93	Yes	259.47	Yes	0.82 %	254.07	No	**254.07**	No	-0.20 %
F12	1	1162.96	No	**1162.96**	No	0.00 %	1162.96	No	**1162.96**	No	0.00 %
	2	1162.96	No	**1162.96**	No	0.00 %	1162.96	No	**1162.96**	No	0.00 %
	3	1162.96	No	**1162.96**	No	0.00 %	1162.96	No	**1162.96**	No	0.00 %

Table 1 Results for MTVRP instances.

Instance	Best known solution	Our solution
R1_6_4	15721.4	15720.8
R1_8_3	29304.5	29301.2
R1_8_5	33494.2	33494.0
R1_8_10	30918.4	30918.3
R2_8_1	24968.8	24963.8
R2_8_5	22798.2	22795.6
R2_8_8	12611.7	12611.6
C1_10_8	41652.1	41648.0
R1_10_1	53046.5	53026.1
R1_10_2	48263.1	48261.6
R1_10_3	44677.1	44673.3
R1_10_6	46930.3	46928.2
R2_10_4	17811.5	17811.4
R2_10_8	17403.8	17403.7
RC1_10_1	45790.8	45790.7
RC1_10_3	42122.0	42121.9
RC1_10_6	44903.6	44898.2
RC1_10_7	44417.1	44409.0
RC1_10_9	43858.1	43858.0
RC2_10_4	15657.0	15654.7

Table 2 Solution costs for the 20 new best known VRPTW solutions

5.3 PDPTW

Several sets of benchmark instances exist for the PDPTW, and four of them are well known. The benchmark sets proposed by Ropke et al. (2007) and Ropke and Cordeau (2009) are tailored towards exact methods and all instances have been solved to optimality. However, two sets where there are still open instances exist. The first set was proposed by Li and Lim (2001) and consists of six groups of instances with 100, 200, 400, 600, 800, or 1,000 customers. There are 354 instances in total, and the distances are Euclidean. In addition, the instances in each group belong to one of six classes: $LC1$, $LC2$, $LR1$, $LR2$, $LRC1$, and $LRC2$. The customers in LC are clustered, the customers in LR are randomly distributed and the customers in LRC are partially clustered and randomly distributed. The -1 instances have a short scheduling horizon, while the -2 instances have a longer one. The second set of benchmark instances was proposed by Sartori and Buriol (2020). Here, the instances were generated using real-world data for addresses and travel times. The set consists of 300 instances and can be split into 12 groups with between 100 and 5,000 customers. Both sets of instances have a hierarchical objective: 1) Minimize the number of vehicles first and 2) Minimize total distance second.

For the first set of instances, an overview of the best known solutions can be found at https://www.sintef.no/projectweb/top/pdptw/. Recent contributors to this overview include Christiaens and Vanden Berghe (2020), Sartori and Buriol (2020), and several unpublished works. We were not able to improve any solutions from this set by warm-starting the RMIM with the best known solution.

For the second set of instances, an overview of the best known solutions can be found at https://github.com/cssartori/pdptw-instances/tree/master/solutions. Here, 246 of 300 solutions provided by the original authors have been improved by three different unpublished works. By warm-starting the RMIM with the best known solutions, we improved 203 of 300 instances. Two of the solutions use fewer vehicles than the previous best known solutions did. The results for the other 201 instances are summarized in Table 3. Here, "# of NBKs" refers to the number of new best known solutions, "Avg. units" refers to the average improvement of the objective function in terms of units, while "Avg. [%]" refers to the average improvement in terms of percentage. As we can clearly see, the RMIM is the method with the most best known solutions. The number of customers in the instances we improved was in the range of 200 to 5,000, and all instances except one had more than 200 customers. We improved all instances in the range of 1500-2,500 and noticed that for several of the largest instances we spent almost all of the allocated time in the root node. Hence, we might have found additional improvements if we had increased the time limit for the largest instances. The reason we were more successful on the second set is likely that fewer researchers have worked on this benchmark set, and fewer optimal solutions have already been found. Also, it seems that the RMIM works better on instances where the distances are integer numbers. To adjust for the hierarchical objective, we added an additional large cost to each route.

Customers	# of NBKs	Avg. units	Avg. [%]
100	0	–	–
200	1	3.0	0.09
400	6	2.5	0.09
600	16	9.8	0.23
800	21	19.7	0.23
1000	23	21.7	0.24
1500	24	25.5	0.25
2000	25	67.0	0.33
2500	25	49.7	0.22
3000	23	102.0	0.28
4000	21	60.7	0.15
5000	16	10.1	0.03
Total/Avg.	201	41.8	0.22

Table 3 Average improvements among new best known PDPTW solutions with unchanged vehicle count.

6 Concluding remarks

This paper presents new exact RMIMs for the PDPTW and VRPTW and a novel matheuristic for the MTVRP. Improvement heuristics are important in solving vehicle routing problems, and RMIMs have been shown to be efficient on the SDVRP, IRP, and PRP. These are vehicle routing problems with a low degree of intra-route constraints. Our computational studies indicate that RMIMs can be a valuable tool for solving vehicle routing problems with a larger degree of intra-route constraints as well. The MTVRP, PDPTW, and VRPTW are all problems where time plays a significant part and thus include several intra-route constraints. Our proposed matheuristic for the MTVRP is able to find the best known solution for 76 of 104 benchmark instances where 28 of them are new to the literature. For the VRPTW and PDPTW, we find 20 and 203 new best known solutions, respectively, by warm-starting the RMIMs with the best known solutions from the literature. These are very good results considering that the VRPTW and PDPTW have received a lot of attention, and numerous solution methods exist for both problems.

The findings of our study confirm that RMIMs are an interesting research direction and that they can complement existing solution methods. As demonstrated in the computational study, they can find improvements that other neighborhood operators have failed to find. In addition, we have demonstrated that they can be formulated for problems with a high degree of intra-route constraints. However, they are slow to solve and suffer from the presence of a great number of constraints and weak big-M constants. Hence, future research should look at how they can be simplified and whether simpler operators that explore the same neighborhoods as the RMIMs can be designed.

Acknowledgments. We want to thank the Norwegian Research Council for funding the research and our collaborators in the AXIOM project [grant number: 263031] for their valuable feedback.

References

Ahmed, M., Seraj, R., and Islam, S. M. S. (2020). The k-means algorithm: A comprehensive survey and performance evaluation. *Electronics*, 9(8):1295.

Alvarenga, G. B., Mateus, G. R., and De Tomi, G. (2007). A genetic and set partitioning two-phase approach for the vehicle routing problem with time windows. *Computers & Operations Research*, 34(6):1561–1584.

Archetti, C., Boland, N., and Speranza, M. G. (2017). A matheuristic for the multivehicle inventory routing problem. *INFORMS Journal on Computing*, 29(3):377–387.

Archetti, C. and Speranza, M. G. (2014). A survey on matheuristics for routing problems. *EURO Journal on Computational Optimization*, 2(4):223–246.

Cattaruzza, D., Absi, N., and Feillet, D. (2018). Vehicle routing problems with multiple trips. *Annals of Operations Research*, 271:127–159.

Christiaens, J. and Vanden Berghe, G. (2020). Slack induction by string removals for vehicle routing problems. *Transportation Science*, 54(2):417–433.

Christofides, N., Mingozzi, A., and Toth, P. (1979). The Vehicle Routing Problem. In Toth, P., editor, *Combinatorial Optimization*, pages 315–338. John Wiley & Sons Ltd, Chichester.

Cordeau, J.-F., Gendreau, M., Hertz, A., Laporte, G., and Sormany, J.-S. (2005). New heuristics for the vehicle routing problem. In Langevin, A. and Riopel, D., editors, *Logistics Systems: Design and Optimization*, pages 279–297. Springer US, Boston, MA.

Cordeau, J.-F. and Laporte, G. (2005). Tabu search heuristics for the vehicle routing problem. In Sharda, R., Voß, S., Rego, C., and Alidaee, B., editors, *Metaheuristic Optimization via Memory and Evolution: Tabu Search and Scatter Search*, pages 145–163. Springer US, Boston, MA.

Fisher, M. L. (1994). Optimal solution of vehicle routing problems using minimum k-trees. *Operations Research*, 42(4):626–642.

Gehring, H. and Homberger, J. (1999). A parallel hybrid evolutionary metaheuristic for the vehicle routing problem with time windows. In *Proceedings of EUROGEN99*, volume 2, pages 57–64, Berlin. Springer.

Laporte, G. (2009). Fifty years of vehicle routing. *Transportation Science*, 43(4):408–416.

Lenstra, J. K. and Kan, A. R. (1981). Complexity of vehicle routing and scheduling problems. *Networks*, 11(2):221–227.

Li, H. and Lim, A. (2001). A metaheuristic for the pickup and delivery problem with time windows. In *Proceedings 13th IEEE International Conference on Tools with Artificial Intelligence. ICTAI 2001*, pages 160–167. IEEE.

Manousakis, E. G., Kasapidis, G. A., Kiranoudis, C. T., and Zachariadis, E. E. (2022). An infeasible space exploring matheuristic for the production routing problem. *European Journal of Operational Research*, 298(2):478–495.

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

Pisinger, D. and Ropke, S. (2007). A general heuristic for vehicle routing problems. *Computers & Operations Research*, 34(8):2403–2435.

Rochat, Y. and Taillard, É. D. (1995). Probabilistic diversification and intensification in local search for vehicle routing. *Journal of Heuristics*, 1:147–167.

Ropke, S. and Cordeau, J.-F. (2009). Branch and cut and price for the pickup and delivery problem with time windows. *Transportation Science*, 43(3):267–286.

Ropke, S., Cordeau, J.-F., and Laporte, G. (2007). Models and branch-and-cut algorithms for pickup and delivery problems with time windows. *Networks: An International Journal*, 49(4):258–272.

Russell, R. A. (2017). Mathematical programming heuristics for the production routing problem. *International Journal of Production Economics*, 193:40–49.

Sartori, C. S. and Buriol, L. S. (2020). A study on the pickup and delivery problem with time windows: Matheuristics and new instances. *Computers & Operations Research*, 124:105065.

Skålnes, J., Vadseth, S. T., Andersson, H., and Stålhane, M. (2023). A branch-and-cut embedded matheuristic for the inventory routing problem. *Computers & Operations Research*, 159:106353.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, 35(2):254–265.

Solyalı, O. and Süral, H. (2017). A multi-phase heuristic for the production routing problem. *Computers & Operations Research*, 87:114–124.

Subramanian, A., Uchoa, E., and Ochi, L. S. (2013). A hybrid algorithm for a class of vehicle routing problems. *Computers & Operations Research*, 40(10):2519–2531.

Taillard, E. D., Laporte, G., and Gendreau, M. (1996). Vehicle routeing with multiple use of vehicles. *Journal of the Operational Research Society*, 47(8):1065–1070.

Toth, P. and Vigo, D. (2014). *Vehicle Routing: Problems, Methods, and Applications.* SIAM.

Uchoa, E., Pecin, D., Pessoa, A., Poggi, M., Vidal, T., and Subramanian, A. (2017). New benchmark instances for the capacitated vehicle routing problem. *European Journal of Operational Research*, 257(3):845–858.

Vadseth, S. T., Andersson, H., Stålhane, M., and Chitsaz, M. (2023). A multi-start route improving matheuristic for the production routeing problem. *International Journal of Production Research*, 0(0):1–22.

Van Breedam, A. (1995). Improvement heuristics for the vehicle routing problem based on simulated annealing. *European Journal of Operational Research*, 86(3):480–490.

Vidal, T. (2022). Hybrid genetic search for the cvrp: Open-source implementation and swap* neighborhood. *Computers & Operations Research*, 140:105643.

Vidal, T., Crainic, T. G., Gendreau, M., Lahrichi, N., and Rei, W. (2012). A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. *Operations Research*, 60(3):611–624.