Gutenberg School of Management and Economics

& Research Unit “Interdisciplinary Public Policy”

Discussion Paper Series

Exact Algorithms for the Multi-Compartment Vehicle Routing Problem with Flexible Compartment Sizes

Katrin Heßler

April 3, 2020

Discussion paper number 2007

Johannes Gutenberg University Mainz
Gutenberg School of Management and Economics
Jakob-Welder-Weg 9
55128 Mainz
Germany
wiwi.uni-mainz.de

Contact Details:

Katrin Heßler
Logistikmanagement
Johannes Gutenberg University Mainz
Jakob-Welder-Weg 9
55128 Mainz
Germany

khessler@uni-mainz.de

Exact Algorithms for the Multi-Compartment Vehicle Routing Problem with Flexible Compartment Sizes

Katrin Heßler*[,a]

[a] *Chair of Logistics Management, Gutenberg School of Management and Economics, Johannes Gutenberg University Mainz, Jakob-Welder-Weg 9, D-55128 Mainz, Germany.*

Abstract

The multi-compartment vehicle routing problem with flexible compartment sizes is a variant of the classical vehicle routing problem in which customers demand different product types and the vehicle capacity can be separated into different compartments each dedicated to a specific product type. The size of each compartment is not fixed beforehand but the number of compartments is limited. We consider two variants for dividing the vehicle capacity: On the one hand the vehicle capacity can be discretely divided into compartments and on the other hand compartment sizes can be chosen arbitrarily. The objective is to minimize the total distance of all vehicle routes such that all customer demands are met and vehicle capacities are respected. Modifying a branch-and-cut algorithm based on a three-index formulation for the discrete problem variant from the literature, we introduce an exact solution approach that is tailored to the continuous problem variant. Moreover, we propose two other exact solution approaches, namely a branch-and-cut algorithm based on a two-index formulation and a branch-price-and-cut algorithm based on a route-indexed formulation, that can tackle both packing restrictions with mild adaptions and can be combined into an effective two-stage approach. Extensive computational tests have been conducted to compare the different algorithms. For the continuous variant, we can solve instances with up to 50 customers to optimality and for the discrete variant, several previously open instances can now be solved to proven optimality. Moreover, we analyse the cost savings of using continuously flexible compartment sizes instead of discretely flexible compartment sizes.

Key words: routing, branch-price-and-cut, multi-compartment

1. Introduction

Multi-compartment vehicle routing problems (MCVRP) are variants of the classical *capacitated vehicle-routing problem* (CVRP, Toth and Vigo 2014) in which several product types must be transported separately. The transportation of products in separated zones is necessary for various real-world problems, e.g., the transportation of dangerous goods, liquid or bulk products, as well as the transportation of food products in different temperature zones. Instead of using one type of vehicle for each product type, it is often beneficial to collect or deliver several product types combined in one vehicle (Muyldermans and Pang 2010). Various different multi-compartment vehicle configurations can be presumed, e.g., the size of separated zones can be fixed or flexible, the assignment of product types to compartments can be preset or arbitrary, and there can exist different (in)compatibilities between two different product types or a compartment and a product type (Pollaris *et al.* 2014).

The paper at hand considers MCVRPs with flexible compartment sizes in which different product types are incompatible with each other such that they must be transported in separate compartments. The

*Corresponding author.
Email address: `khessler@uni-mainz.de` (Katrin Heßler)

assignment of product types to compartments is preset. Two different problem variants are investigated: On the one hand we consider the *multi-compartment vehicle routing problem with continuously flexible compartment sizes* (MCVRP-CFCS, Koch *et al.* 2016) in which compartment sizes can be set arbitrarily within the limits of the vehicle capacity. A practical application is, in particular, the distribution of food (Derigs *et al.* 2010; Hübner and Ostermeier 2019). On the other hand, we consider the *multi-compartment vehicle routing problem with discretely flexible compartment sizes* (MCVRP-DFCS, Henke *et al.* 2015) in which compartment sizes can only be set according to pre-defined, equally spaced positions. Practical applications are amongst others the shipment of bulk products (Fagerholt and Christiansen 2000) and the collection of glass waste (Henke *et al.* 2015). In the following, we refer collectively to the MCVRP-CFCS and MCVRP-DFCS as *multi-compartment vehicle routing problems with flexible compartment sizes* (MCVRP-FCS).

The MCVRP-CFCS and MCVRP-DFCS are both a generalization of the CVRP (Toth and Vigo 2014) and, therefore, NP-hard. Moreover, the MCVRP-CFCS is a restriction of the *commodity-constrained split delivery vehicle routing problem* (C-SDVRP, Archetti *et al.* 2016; Gschwind *et al.* 2019) in which customer demands are composed of different commodities but no product types exist such that all commodities can be transported together in one zone. If the limit on the number of compartments in the MCVRP-CFCS is greater or equal to the number of product types then all different product types can be transported together on one vehicle and both the MCVRP-CFCS and the C-SDVRP are equivalent.

In the literature, several variants of the MCVRP with heuristic and exact solution approaches have been discussed. Pollaris *et al.* (2014) present an overview of vehicle routing problems with loading constraints including a summary of MCVRP literature. Henke (2017) provides a recent review and extended classifications for the MCVRP. In the following, we first give a short overview of publications about MCVRPs with fixed compartment sizes and focus afterwards on literature about MCVRPs with flexible compartment sizes.

Fixed compartments. Numerous MCVRP publications with fixed compartment sizes deal with the distribution of liquid products. In particular, different petrol replenishment problems are studied. The specialty of petrol distribution is that typically the content of each compartment can only be delivered to one customer because vehicles are not equipped with debit meters. Brown and Graves (1981) present an automated real-time dispatch system. Avella *et al.* (2004) provide a branch-and-price algorithm and Cornillier *et al.* (2008) formulate a set-partitioning problem that can solve instances with a small set of petrol stations optimally. Recent technology allows us to equip vehicles with debit meters so that the content of a compartment can be split between several deliveries and customers may allow different vehicles to fill the same tank. Using this fact, Coelho and Laporte (2015) present a classification scheme that distinguishes between split and unsplit compartments and tanks. They propose specialized models for particular versions of the problem and a branch-and-cut algorithm applicable to all variants. A variant of the MCVRP that includes time windows is solved by Benantar *et al.* (2016) with a tabu search algorithm. Other MCVRP variants with liquid products are the collection of olive oil (Lahyani *et al.* 2015), solved by a branch-and-cut algorithm, and the collection of raw milk (Caramia and Guerriero 2010), solved by the combination of two mathematical formulations and a local search algorithm.

Routing logistics literature on other goods than liquid products is also rich. Muyldermans and Pang (2010) introduce a local search algorithm for a waste collection problem and compare separate collection for each waste type with co-collection of different waste types. An ant colony algorithm is proposed by Reed *et al.* (2014) that solves a waste collection problem in which the location of the depot site is separated from the vehicle depot. Fallahi *et al.* (2008) suggest a memetic algorithm and a tabu search for an animal food distribution problem with sanitary rules that recommend to always use the same compartment for one species. Similar sanitary rules are defined in the livestock collection problem in which animals from farms are collected for slaughter at a slaughterhouse. Oppen and Løkketangen (2008) present a tabu search approach and Oppen *et al.* (2010) introduce an exact column-generation based solution approach. A grocery distribution problem is presented by Ostermeier *et al.* (2018) that includes the decision of using cost-different single-compartment or multi-compartment vehicles. The problem is solved by a large neighborhood search. An MCVRP with time windows and three time planning periods arising in a city logistics problem is proposed and solved by an adaptive large neighborhood search by Eshtehadi *et al.* (2020). Mirzaei and Wøhlk (2017) compare two MCVRP variants that allow either only single or multiple visits to the same customer. Both variants are solved exactly by a branch-and-price algorithm. A variable neighborhood search for the selective MCVRP with time windows is proposed by Melechovský (2013). In this variant profits are dedicated to customers and product types and the aim is to maximize the total profit. Other MCVRP variants consider stochastic instead of deterministic demands (Mendoza *et al.* 2010; 2011; Goodson 2015).

Flexible compartments. Little attention has been paid to MCVRP with flexible compartment sizes. Fagerholt and Christiansen (2000) introduce a bulk ship scheduling problem with a flexible cargo hold that can be partitioned discretely into several smaller holds. The problem is solved by a set partitioning approach consisting of two phases for the scheduling and allocation problem. Chajakis and Guignard (2003) propose a model for the distribution to convenience stores and develop approximation schemes based on Lagrangean relaxation. The packing is constrained by two independent dimensions (weight and volume), and apart from transportation also cooling costs of each compartment for non-ambient temperature items are considered. An MCVRP with loading and unloading costs that occurs in grocery distribution is introduced by Hübner and Ostermeier (2019). In this variant, using multi-compartment vehicles saves transportation costs but increases (un)loading costs because more than one shipping gate has to be approached at the warehouse. They present a large-neighborhood search with specialized removal and reinsert operators. Ostermeier *et al.* (2018) include loading constraints to the problem, develop a branch-and-cut algorithm, and extend the large neighborhood search of Hübner and Ostermeier. Derigs *et al.* (2010) consider the MCVRP with fixed and flexible compartment sizes and introduce a solver suite consisting of construction heuristics, improvement heuristics, and metaheuristics. In both variants, products are not dedicated to compartments but incompatibility relations between products and compartments as well as two products are considered. Compartment sizes can be set arbitrarily in the variant with flexible compartment sizes. They do not consider the discrete version. Pirkwieser *et al.* (2012) extend this problem by using a measure to distinguish packings and aiming to use solutions with a denser packing. They present a variable neighborhood search with a new neighborhood structure.

Henke *et al.* (2015) introduce the MCVRP-DFCS that occurs in the context of glass waste collection. A model formulation is proposed that can solve problem instances with up to 10 locations to proven optimality. Moreover, they provide a variable neighborhood search that finds good quality solutions. Later on, Henke *et al.* (2018) suggest a branch-and-cut algorithm for the MCVRP-DFCS. Their algorithm can solve instances with up to 50 locations to proven optimality within two hours. The model formulation is also used for the MCVRP-CFCS variant by setting the unit compartment size to one. We later compare against their results. Koch *et al.* (2016) introduce the MCVRP-CFCS and present a heuristic approach that is based on different genetic algorithms for the CVRP from the literature. The algorithm can find an optimal solution for the majority of instances with up to 50 locations within one second (Henke 2018). The cost saving of using continuously flexible compartments instead of discrete ones is also investigated.

The contributions of the paper at hand are the following. We introduce a three-index formulation tailored to solve the MCVRP-CFCS exactly. Moreover, we introduce a two-index formulation and a route-based formulation suited for column generation for both the MCVRP-CFCS and MCVRP-DFCS. Both algorithms can solve the two problem variants with mild adaptions and are combined to an effective two-stage approach. To compare the algorithms, extensive numerical experiments have been conducted on instances from the literature. For the MCVRP-CFCS, the experiments demonstrate good performance for instances with up to 50 customers. For the MCVRP-DFCS, several new instances can be solved to proven optimality for the first time compared to results from the literature.

The remainder of the paper is organized as follows. In the next section, we formally define the MCVRP-CFCS and MCVRP-DFCS. Subsequently, three exact solution approaches are presented. At first, a branch-and-cut algorithm based on a three-index and separation procedures are introduced in Sections 3. We do the same in Section 4 with a branch-and-cut algorithm based on a two-index formulation. Afterwards, a branch-price-and-cut algorithm including details on the generation of route variables, stabilization techniques, valid inequalities, and branching is presented in Section 5. In Section 6, we conduct numerical experiments to compare the different algorithms and compare total costs of the MCVRP-CFCS and MCVRP-DFCS. Conclusions are drawn in Section 7.

2. Problem Definition

We formally define the MCVRP-CFCS and MCVRP-DFCS as follows. Let $N = \{1, \dots, n\}$ be the set of *customers* and $P = \{1, \dots, \rho\}$ the set of *product types*. The *demand* of customer $i \in N$ for product type $p \in P$ is denoted by d_{ip}. The set of product types $P_i = \{p \in P : d_{ip} > 0\}$ delivered to customer $i \in N$ may contain any and all product types P, i.e., $P_i \subseteq P$ for all $i \in N$.

A maximum of m homogeneous *vehicles* $F = \{1, \dots, m\}$ is available for delivery. Each vehicle can be separated into a limited number of C *compartments*. Note that the number of product types demanded by customer i can exceed the number of compartments, i.e., $|P_i| > C$ is possible such that at least two vehicles are needed to serve customer i. For the MCVRP-CFCS, the compartment sizes can be set arbitrarily. For the MCVRP-DFCS, the vehicle capacity can be separated into compartments such that each compartment size is a multiple of *unit size* q^{unit}.

Let $G(V, E)$ be a complete undirected graph with vertex set $V = N \cup \{0\}$ and edge set E with $i < j$ for all $\{i, j\} \in E$. Vertex 0 represents the *depot* and routing *costs* between two nodes $\{i, j\} \in E$ are given by c_{ij}. A *route* $r = \{i_0, \dots, i_s, i_{s+1}\}$ delivering products $S_{i_k} \subseteq P_{i_k}$, $k \in \{1, \dots, s\}$, is feasible if

(i) it is a cycle passing through the depot, i.e., $i_0 = i_{s+1} = \{0\}$;
(ii) all customers $i_1, \dots, i_s$ are different;
(iii) the number of compartments is respected, i.e., $\left|\bigcup_{k=1}^{s} S_{i_k}\right| \le C$; and
(iv) capacity constraints hold, i.e., for continuously flexible compartment sizes

$$\sum_{k=1}^{s} \sum_{p \in S_{i_k}} d_{i_k p} \le Q, \tag{1a}$$

or for discretely flexible compartment sizes

$$\sum_{p \in P} \left\lceil \sum_{\substack{k \in \{1,\dots,s\}, \\ p \cap S_{i_k} \neq \emptyset}} d_{i_k p} \right\rceil_{q^{\text{unit}}} \le Q, \tag{1b}$$

where $\lceil . \rceil_{q^{\text{unit}}}$ denotes the rounding up value according to the unit compartment size q^{unit}. Regardless of compartment division, the task is to determine a cost-minimal set of at most m feasible routes such that all customer demands are met.

The formulations that we introduce in the following rely on different graphs. For the sake of clarity, we already define most of these graphs in this section. A summary of all graphs is depicted in Table 1. Graph $\bar{G}(\bar{V}, \bar{E})$ is derived from graph $G(V, E)$ by duplicating each customer node $i \in N$ for all product types $p \in P_i$ yielding a new customer set $\bar{N}$. The new graph $\bar{G}$ consists of $\left|\bar{V}\right| = 1 + \sum_{i \in N} |P_i|$ vertices. For each vertex $k \in \bar{N}$, let $f_c(k) \in N$ denote the corresponding customer, $f_p(k) \in P$ the corresponding product type, and $f_d(k) \in P$ the corresponding demand, respectively. Moreover, let $\bar{E}$ be the corresponding edge set such that $\bar{G}(\bar{V}, \bar{E})$ results in a complete undirected graph. Both graphs G and $\bar{G}$ can also be converted into directed graphs G^d and $\bar{G}^d$, respectively, by duplicating each edge between customers into two reversed arcs and adding a second depot node $n + 1$. The start depot 0 is connected to all customer nodes by outgoing arcs and the end depot $n + 1$ is connected to all customer nodes by incoming arcs. Let A and $\bar{A}$ denote the arc sets, respectively.

Table 1: Overview of graphs.

graph	(un)directed	vertex set	customer set	edge/arc set	depot vertices	number of vertices of customer i
G	undirected	V	N	E	0	1
G^d	directed	$V \cup \{n+1\}$	N	A	$0, n+1$	1
$\bar{G}$	undirected	$\bar{V}$	$\bar{N}$	$\bar{E}$	0	$\lvert P_i \rvert$
$\bar{G}^d$	directed	$\bar{V} \cup \{n+1\}$	$\bar{N}$	$\bar{A}$	$0, n+1$	$\lvert P_i \rvert$

3. Branch-and-cut algorithm (three-index formulation)

Henke *et al.* (2018) suggest a branch-and-cut algorithm for the MCVRP-DFCS based on a three-index formulation. Their model handles discrete compartment size by variables y^I_{pf}, with $p \in P$ and $f \in F$, that indicate the size of the compartment of vehicle f for product type p in the number of basic unit sizes q^{unit}. To compare the total cost of both (continuous and discrete) problem variants, they suggest setting $q^{\text{unit}} = 1$ for the continuous variant. Instead, we propose a model for the MCVRP-CFCS that does not use the basic unit compartment size. Note that in this section we only present the solution approach for the MCVRP-CFCS. For the MCVRP-DFCS, we refer to (Henke *et al.* 2018).

Recall graph $G(V, E)$ defined in Section 2. The new model relies on four types of variables. First of all, the symmetric formulation has non-negative integer routing variables x_{ijf} for all edges $\{i, j\} \in E$ and vehicles $f \in F$. Binary delivery variables u_{ipf} indicate whether the demand of product type $p \in P$ at customer $i \in N$ is served by vehicle $f \in F$. The coupling between routing and delivery variables is ensured variables z_{if} that specify if node $i \in V$ is visited by vehicle $f \in F$. Additionally, to handle the maximal allowed number of compartments per vehicle, we introduce binary variables y_{pf} indicating whether the vehicle $f \in F$ delivers product type $p \in P$. The new formulation is:

$$\min \sum_{\{i,j\} \in E} \sum_{f \in F} c_{ij} x_{ijf} \tag{2a}$$

$$\text{subject to} \quad \sum_{f \in F} u_{ipf} = 1 \qquad \forall i \in N, p \in P, d_{ip} > 0 \tag{2b}$$

$$u_{ijf} \leq z_{if} \qquad \forall i \in N, p \in P, f \in F \tag{2c}$$

$$z_{if} \leq z_{0f} \qquad \forall i \in N, f \in F \tag{2d}$$

$$\sum_{j \in N} x_{0jf} \leq 2m \tag{2e}$$

$$\sum_{j \in V, \{i,j\} \in E} x_{ijf} + \sum_{j \in V, \{j,i\} \in E} x_{jif} = z_{if} \qquad \forall i \in V, f \in F \tag{2f}$$

$$\sum_{i \in N} u_{ipf} \leq n y_{pf} \qquad \forall p \in P, f \in F \tag{2g}$$

$$\sum_{p \in P} y_{pf} \leq C \qquad \forall f \in F \tag{2h}$$

$$\sum_{i \in N} \sum_{p \in P} d_{ip} u_{ipf} \leq Q \qquad \forall f \in F \tag{2i}$$

$$\sum_{\{i,j\} \in \delta(S)} x_{ijf} \geq 2\sigma(S) \qquad \forall f \in F, S \subseteq N, S \neq \emptyset \tag{2j}$$

$$x_{ijf} \in \{0, 1\} \qquad \forall \{i, j\} \in E, i \neq 0, f \in F \tag{2k}$$

$$x_{0jf} \in \{0, 1, 2\} \qquad \forall j \in N, f \in F \tag{2l}$$

$$u_{ipf} \in \{0,1\} \qquad \forall i \in N, p \in P, f \in F \tag{2m}$$

$$z_{if} \in \{0,1\} \qquad \forall i \in V, f \in F \tag{2n}$$

$$y_{pf} \in \{0,1\} \qquad \forall p \in P, f \in F. \tag{2o}$$

The objective function (2a) minimizes routing costs. Equalities (2b) ensure that each supply is delivered by exactly one vehicle. The coupling between u- and z-variables is established by constraints (2c). Constraints (2d) ensure that a vehicle only visits customers if the depot is included in the tour and (2e) restricts the number of vehicles. The float constraints are established by (2f). The coupling between u- and y-variables is guaranteed by constraints (2g). Constraints (2h) and (2i) limit the number of compartments and the capacity per vehicle, respectively. Constraints (2j), known as capacity cuts, ensure both solution connectivity and packing feasibility according to (iii) and (1). In these constraints, $\delta(S)$ is the set of edges with exactly one endpoint in S and $\sigma(S)$ denotes the minimum number of vehicles needed to serve S. Already for the classical CVRP, it is difficult to calculate $\sigma(S)$ because an (NP-hard) one-dimensional bin packing problem with items $k \in S$, weights $f_d(k)$, and bin capacity Q must be solved. Therefore, it is usual to replace $\sigma(S)$ by a lower bound of a simple relaxation. For the MCVRP-CFCS, one such bound that calculates the minimum of vehicles needed to serve S according to the number of compartments and the vehicle capacity is

$$\max\left\{\left\lceil \frac{|f_p(S)|}{C} \right\rceil, \left\lceil \frac{f_d(S)}{Q} \right\rceil\right\}, \tag{3a}$$

where $f_p(S)$ is the set of product types and $f_d(S)$ the sum of the demands of all vertices in S. For the MCVRP-DFCS, we can bound $\sigma(S)$ from below by

$$\max\left\{\left\lceil \frac{|f_p(S)|}{C} \right\rceil, \left\lceil \frac{1}{Q} \sum_{p \in P} \left\lceil \sum_{k \in S, p = f_p(k)} f_d(k) \right\rceil_{q^{\text{unit}}} \right\rceil\right\}. \tag{3b}$$

Here, the second argument additionally takes into account discrete compartment sizes. Finally, variable domains are defined by (2k)-(2o).

3.1. Valid inequalities

Additional symmetry breaking constraints are added to formulation (2) to avoid equivalent feasible solutions that can occur if the same tour is assigned to different vehicles. Preliminary experiments showed that ordering tours in decreasing order of their total cost is most beneficial for the MCVRP-DFCS (Henke *et al.* 2018). Therefore, we also add the following symmetry breaking constraints to formulation (2) for the MCVRP-CFCS.

$$\sum_{\{i,j\} \in E} c_{ij} x_{ij,f+1} \leq \sum_{\{i,j\} \in E} c_{ij} x_{ijf} \qquad \forall f \in F \setminus \{|F|\} \tag{4}$$

3.2. Separation procedure

Simply solving (2) by using a MIP solver is not advisable because the number of capacity cuts is exponential in $|V|$. In this section we describe how these constraints can be added dynamically utilizing a separation procedure.

For both integer and fractional solutions, we apply two different procedures, namely subtour-elimination constraints and exact capacity cuts. Note that an inequality is violated if the difference between the left-hand and right-hand side is greater than a given threshold $\epsilon = 10^{-4}$.

Subtour-elimination constraints. For each vehicle $f \in F$, we find subtours as follows. Let $\bar{x}_{ijf}$ be a solution to the LP for vehicle $f \in F$ and $G^s(V, E^s)$ be the support graph. To determine subtours, we call Algorithm 1 on support graph $G^s(V, E^s)$ with edge set $E^s = \{\{i,j\} \in E : \bar{x}_{ijf} > 0\}$. Irrespective of whether or not subset S is a real subtour, all found violated cuts are added to the model. Note that contrary to (Henke *et al.* 2018), we allow fractional solutions for this procedure.

Capacity cuts. As proposed by Henke *et al.* (2018), capacity cuts are additionally separated. We call Algorithm 1 on a combined support graph $G^s(V, E^s)$ for all vehicles where edge set $E^s = \{\{i,j\} \in E : \bar{x}_{ij} = \sum_{f \in F} \bar{x}_{ijf} > 0, i \neq 0\}$. Connected components S are determined and all violated cuts are added.

Algorithm 1: Violated cut generator

input : support graph $G(V, E)$ with edge set $E = \{\{i,j\} \in E : \bar{x}_{ij} > 0\}$
output: violated cuts

```
Determine connected components S of G via the efficient union-find algorithm of Tarjan (1979);
foreach connected component S do
    Set S ← S \ {0};
    Calculate the flow f_0S between the depot 0 and S;
    Calculate σ(S) according to (3a) or (3b), respectively;
    if f_0S < 2σ(S) then
        A violated cut for subset S is found;
```

---

4. Branch-and-cut algorithm (two-index formulation)

The two-index formulation relies on the undirected graph $\bar{G}(\bar{V}, \bar{E})$ defined in Section 2. Recall that for each node $k \in \bar{N}$, the functions $f_c(k) \in N$, $f_p(k) \in P$, and $f_d(k) \in P$ respectively denote the corresponding customer, product type, and demand. Travel costs between the same customer are set to 0, i.e., $c_{kl} = 0$ for all $\{k,l\} \in \bar{E}$ with $f_c(k) = f_c(l)$. For $k \in \bar{V}$, let $\delta(k)$ be the set of all edges incident to k. Our formulation is based on the classical symmetric formulation of Laporte *et al.* (1985) that is already successfully applied to other vehicle routing problems (VRP) with difficult packing restrictions, e.g. the VRP with two-dimensional loading constraints (Iori *et al.* 2007). We use binary routing variables x_{kl} indicating whether a vehicle traverses edge $\{k,l\} \in \bar{E}$. The two-index formulation is:

$$
\begin{aligned}
\min \quad & \sum_{\{k,l\} \in \bar{E}} c_{kl} x_{kl} && && \text{(5a)}\\
\text{subject to} \quad & \sum_{\{k,l\} \in \delta(k)} x_{kl} = 2 && \forall k \in \bar{N} && \text{(5b)}\\
& \sum_{\{0,l\} \in \delta(0)} x_{0l} = 2y && && \text{(5c)}\\
& \sum_{\{k,l\} \in \delta(S)} x_{kl} \geq 2\sigma(S) && \forall S \subseteq \bar{N}, S \neq \emptyset && \text{(5d)}\\
& x_{kl} \in \{0,1\} && \forall \{k,l\} \in \bar{E} \setminus \delta(0) && \text{(5e)}\\
& x_{0l} \in \{0,1,2\} && \forall \{0,l\} \in \delta(0) && \text{(5f)}\\
& \left\lceil \frac{\sum_{k \in \bar{V}} f_d(k)}{Q} \right\rceil \leq y \leq m \text{ and integer.} && && \text{(5g)}
\end{aligned}
$$

The objective (5a) minimizes travel costs. Constraints (5b) impose that each node is visited once and constraint (5c) restricts the number of vehicles leaving from and returning to the depot. Constraints (5d), known as capacity cuts, ensure both solution connectivity and packing feasibility according to (iii) and (1). Again, $\delta(S)$ is the set of edges with exactly one endpoint in S and $\sigma(S)$ denotes the minimum number of vehicles needed to serve S. We bound $\sigma(S)$ from below by (3a) or (3b). The domains of routing and vehicle number variables are given by (5e)-(5f) and (5g), respectively.

The disadvantage of formulation (5) is that on the one hand symmetry can occur between two solutions when tours are identical but the sequence of packing product types for a customer is different and on the other hand it cannot be solved by directly using a MIP solver because it contains the large-size family of constraints (5d). In the following, we introduce symmetry breaking constraints as well as other valid inequalities and describe how constraints (5d) can be added dynamically using separation procedures.

4.1. Valid inequalities

Formulation (5) can be further strengthened by employing valid inequalities. We introduce one class of symmetry breaking constraints and two classes of logical inequalities.

Consider a customer with (at least) three product types k, l, and s supplied by one vehicle (see Figure 1a). Then the solution $x_{kl} = x_{ls} = 1$ is equivalent to $x_{ks} = x_{ls} = 1$. To forbid the latter and ensure that products belonging to the same customer are collected in an increasingly manner, we introduce the class of symmetry breaking constraints

$$x_{ks} + x_{ls} \leq 1 \qquad \forall k, l, s \in \bar{V}, f_c(k) = f_c(l) = f_c(s). \tag{6a}$$

Moreover, it is possible to calculate an upper bound on the flow within a customer. An example is illustrated in Figure 1b. Consider a customer i demanding p_i product types. We can divide the vertices belonging to customer i into groups of size C, e.g. nodes 1 and 2 in Figure 1b are one group. The number of edges within one group is at most $C-1$. The $p_i \mod C$ leftover vertices not assigned to a group (node 5 in Figure 1b) can be connected by at most $\max\{0, (p_i \mod C) - 1\}$ edges. Hence, the flow between vertices of customer i is at most

$$\text{maxflow}(i) = \left\lceil \frac{p_i}{C} \right\rceil (C-1) + \max\{0, (p_i \mod C) - 1\}.$$

Therefore, valid inequalities are

$$\sum_{\substack{\{k,l\} \in \bar{E}, \\ f_c(k) = f_c(l) = i}} x_{kl} \leq \text{maxflow}(i) \qquad \forall i \in N. \tag{6b}$$

If the number of compartments is $C = 2$ then the flow from a vertex of a customer to other vertices of the same customer is at most 1. This is especially essential for customers with many product types. Therefore, we can employ the second class of valid inequalities

$$\sum_{\substack{\{k,l\} \in \delta(l), \\ f_c(k) = f_c(l)}} x_{kl} + \sum_{\substack{\{l,s\} \in \delta(l), \\ f_c(l) = f_c(s)}} x_{ls} \leq 1 \qquad \forall l \in \bar{V}. \tag{6c}$$

(a) Equivalent solutions $x_{kl} = x_{ls} = 1$ (solid lines) and $x_{ks} = x_{ls} = 1$ (dashed lines).

(b) Solution with a maximum number of edges for a customer with five product types and a limited number of compartments $C = 2$.

Figure 1: Examples to illustrate inequalities (6a) and (6b). In both cases, all vertices belong to one customer and only edges between vertices of this customer are considered.

4.2. Separation procedure

Again, formulation (5) contains a large-sized family of constraints because the number of capacity cuts is exponential in $|\bar{V}|$. Similar to the separation procedure described in Section 3.2, subtour-elimination constraints, and capacity cuts are added dynamically to the model as follows.

Subtour-elimination constraints. Let $\bar{x}_{kl}$ be an integer or fractional solution to the LP and $\bar{G}^s(\bar{V}, \bar{E}^s)$ be the support graph with edge set $\bar{E}^s = \{\{k,l\} \in \bar{E} : \bar{x}_{kl} > 0\}$. Subtours are determined by utilizing Algorithm 1. Analogous to Section 3.2, irrespective of whether or not subset S is a real subtour, all found violated cuts are added to the model.

Capacity cuts. First, we apply a heuristic procedure that also relies on the support graph $\bar{G}^s$ with edge set $\bar{E}^s$. The algorithm tries to find a subset S of small size that is connected and consists of many different product types. The pseudocode is depicted in Algorithm 2. Starting with a randomly chosen vertex $k \in \bar{N}$, the set S is enlarged by adding connected vertices on the support graph $\bar{G}^s$ with preferably new product types. Set S is further enlarged until either no connected vertex exists or a violated cut $f_{0S} < 2\sigma(S)$ is found. The algorithm is restarted with a new non-considered randomly chosen vertex $k \in \bar{N} \setminus U$ (set U contains already considered vertices) until all vertices are processed.

Second, if no violated cut is found by the heuristic procedure, we apply Algorithm 1 for the support graph $G^s(\bar{V}, \bar{E}^s)$ and edge set $\bar{E}^s = \{\{k,l\} \in \bar{E} : \bar{x}_{kl} > 0, k \neq 0\}$.

Algorithm 2: Heuristic capacity cut

input : graph $\bar{G}^s(\bar{V}, \bar{E}^s)$
output: sets to check S

Set $S = U = \varnothing$;
while $U \neq \bar{V} \setminus \{0\}$ **do**
  if $S = \varnothing$ **then**
    Choose randomly a vertex $k \in \bar{N} \setminus U$ and set $S \leftarrow S \cup \{k\}$ and $U \leftarrow U \cup \{k\}$;
  else if *Vertices connected to S exist* **then**
    Choose randomly a vertex $k \in \bar{N} \setminus U$ connected to S (if possible with $f_p(k) \notin f_p(S)$) and set $S \leftarrow S \cup \{k\}$ and $U \leftarrow U \cup \{k\}$;
  else
    $S = \varnothing$;
  Check S regarding $f_{0S} < 2\sigma(S)$;

5. Branch-price-and-cut algorithm

To solve both the MCVRP-CFCS and MCVRP-DFCS with a column-based solution approach, we propose a set-partitioning formulation. Since the MCVRP-CFCS is a restriction of the C-SDVRP, we can adapt the model of Archetti *et al.* (2015). The new formulation is based on the directed graph $G^d(V \cup \{n+1\}, A)$ (cf. Section 2). Each vehicle route starts and ends at the depot vertices 0 and $n+1$, respectively. A feasible route is an elementary 0-$(n+1)$-path that respects the number of compartments and capacity constraints (cf. (i)-(1) in Section 2). Let Ω be the set of feasible routes and $c^r = \sum_{(i,j)\in A(r)} c_{ij}$ the cost of route $r \in \Omega$, where $A(r) \subset A$ is the set of arcs traversed by route r. The formulation uses binary route variables λ^r, $r \in \Omega$, that indicate whether a route is performed. The non-negative integer variables ψ and z_i model the number of used vehicles and the number of times customer $i \in N$ is visited, respectively. The flow over arc $(i,j) \in A$ is modeled by non-negative integer variables x_{ij}. Moreover, let $X = \{P' \subseteq P : |P'| = C\}$ be the set of all feasible packing combinations of different product types. For example, an instance with three product types and a maximum of $C = 2$ compartments results in three feasible packing combinations $X = \{\{1,2\},\{2,3\},\{1,3\}\}$. Moreover, let χ_L be the number of routes packed with compartment combination $L \in X$. The formulation is as follows:

$$\min \quad \sum_{r\in\Omega} c^r \lambda^r \tag{7a}$$

$$\text{subject to} \quad \sum_{r\in\Omega} a_{ip}^r \lambda^r = 1 \qquad \forall i \in N, p \in P_i \tag{7b}$$

$$\sum_{r\in\Omega} \lambda^r - \psi = 0 \tag{7c}$$

$$\left\lceil \frac{\sum_{i\in N}\sum_{p\in P_i} d_{ip}}{Q} \right\rceil \le \psi \le m \text{ and integer} \tag{7d}$$

$$\lambda^r \in \{0,1\} \qquad \forall r \in \Omega \tag{7e}$$

$$\sum_{r\in\Omega} g_L^r \lambda^r - \chi_L = 0 \qquad \forall L \in X \tag{7f}$$

$$0 \le \chi_L \le m \text{ and integer} \qquad \forall L \in X \tag{7g}$$

$$\sum_{r\in\Omega} e_i^r \lambda^r - z_i = 0 \qquad \forall i \in N \tag{7h}$$

$$1 \le z_i \le \min\{|P_i|, m\} \text{ and integer} \qquad \forall i \in N \tag{7i}$$

$$\sum_{r\in\Omega} b_{ij}^r \lambda^r - x_{ij} = 0 \qquad \forall (i,j) \in A \tag{7j}$$

$$0 \le x_{ij} \le \min\{|P_i|, |P_j|, m\} \text{ and integer} \qquad \forall (i,j) \in A. \tag{7k}$$

The objective function (7a) minimizes routing costs. Equalities (7b) ensure that each supply is delivered by exactly one route. In these constraints, the binary coefficient $a_{ip}^r = 1$ if product $p \in P_i$ is delivered to customer $i \in N$ by route r. Constraint (7c) models the number of vehicles and constraints (7d) and (7e) define the domains for vehicle number variable ψ and route variables λ^r. Constraints (7f)-(7k) are redundant but might be added for branching and/or to ensure integer solutions. More precisely, constraints (7f)-(7g) count the number of routes that are packed with compartment combination $L \in X$. Here, the coefficients g_L^r indicate if route r is packed with compartment combination $L \in X$. Moreover, constraints (7h)-(7k) restrict the number of times customer $i \in N$ is visited and arc $(i,j) \in A$ is traversed. In these constraints, the binary coefficients e_i^r and b_{ij}^r indicate if customer $i \in N$ is visited and arc $(i,j) \in A$ is traversed by route r, respectively.

Since the set Ω of feasible routes and, accordingly, the number of columns in formulation (7) is very big, we perform a branch-price-and-cut (BPC) algorithm to solve the problem. For this purpose, we start with the linear relaxation of (7) over a subset $\Omega' \subset \Omega$. This so-called restricted master problem (RMP) is solved by column generation (Desaulniers *et al.* 2005). Similar to the C-SDVRP, the subproblem can be formulated as a variant of the shortest-path problem with resource constraints (SPPRC, Irnich and Desaulniers 2005). To reach integrality this column generation process is embedded in a branch-and-bound algorithm.

In the following, we describe different components of the algorithm, namely how to solve the subproblem, stabilization techniques by the help of dual inequalities, valid inequalities to strengthen the lower bound, the branching procedure, and further acceleration techniques.

5.1. *Pricing problem formulation*

Instead of solving one subproblem at each column generation iteration, we divide the subproblem into several pricing problems and solve each of these pricing problems separately. To reduce the difficulty of packing constraints according to compartments, we consider $|X|$ pricing problems, i.e., one pricing problem for each feasible compartment combination $L \in X$, where $L \subseteq P$ denotes the set of considered product types. Recall that for example, an instance with three product types and a maximum of $C = 2$ compartments results in three pricing problems $X = \{\{1,2\},\{2,3\},\{1,3\}\}$.

Let π_{ip}, σ, ν_L, μ_i, and ρ_{ij} be the dual variables associated with constraints (7b), (7c), (7f), (7h), and (7j), respectively. Reconsider the directed graph $\bar{G}^d(\bar{V} \cup \{n+1\}, \bar{A})$ defined in Section 2. Let

$$\bar{c}_{kl} = c_{kl} - \frac{1}{2}(\pi_{f_c(k)f_p(k)} + \pi_{f_c(l)f_p(l)}) - \frac{1}{2}(\mu_{f_c(k)} + \mu_{f_c(l)}) - \rho_{f_c(k)f_c(l)}$$

be the modified travel cost over arc $(k,l) \in \bar{A}$. Then, the pricing problem for $L \in X$ can be formulated as follows:

$$
\begin{aligned}
\min \quad & \sum_{(k,l)\in\bar{A}} \bar{c}_{kl} x_{kl} - \sigma - \nu_L && && \text{(8a)}\\
\text{subject to} \quad & \sum_{(k,l)\in\bar{A}} x_{kl} - \sum_{(l,s)\in\bar{A}} x_{ls} = 0 && \forall l \in \bar{N} && \text{(8b)}\\
& \sum_{l\in\bar{V}} x_{0l} = 1 && && \text{(8c)}\\
& \sum_{k\in\bar{V}} x_{k,m+1} = 1 && && \text{(8d)}\\
& \sum_{(k,l)\in\bar{A}} x_{kl} + \sum_{(l,s)\in\bar{A}} x_{ls} = 0 && \forall l, f_p(l) \in P \setminus L && \text{(8e)}\\
& \sum_{(k,l)\in\delta(S)} x_{kl} \geq 2\sigma(S) && \forall S \subseteq \bar{N}, S \neq \emptyset && \text{(8f)}\\
& x_{kl} \in \{0,1\} && \forall (k,l) \in \bar{A}. && \text{(8g)}
\end{aligned}
$$

The objective (8a) minimizes the reduced cost of the route and float conservation is ensured by constraints (8b). Constraints (8c) and (8d) impose that exactly one vehicle leaves and enters the depot, respectively. All arcs that should not be considered in the pricing problem for $L \in X$ are set to 0 in constraints (8e). Capacity constraints (8f) ensure connectivity and packing feasibility according to (1). Note that (iii) holds true by construction because the number of used compartments is already limited by constraints (8e). The domain of variables x_{kl} is given by (8g).

5.2. SPPRC formulation for the pricing problem

To solve the pricing problem for $L \in X$, we formulate it as an SPPRC over an undirected multi-graph. For this purpose, the depot node 0 and all customer nodes $i \in N$ are duplicated into two copies $0'$ and $0''$ as well as i' and i'', respectively. Each arc $(i,j) \in A$ results in two *routing edges* $\{i', j''\}$ and $\{i'', j'\}$. To model deliveries to customer i, there are parallel *delivery edges* between i' and i'' for each feasible packing combination $S_i \subseteq P_i$ with $S_i \subseteq L$, denoted as $\{i', i''\}^{S_i}$. Figure 2 shows an example of two pricing problems for an instance with three customers.

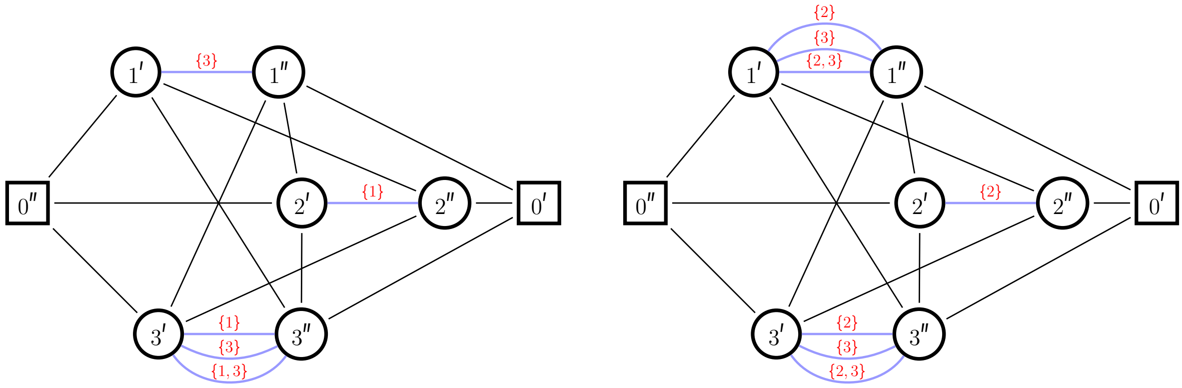

Figure 2: Two SPPRC pricing networks with three customers $N = \{1,2,3\}$ and product type sets $P_1 = \{2,3\}, P_2 = \{1,2\}$ and $P_3 = \{1,2,3\}$ for an instance with $C = 2$ and $\rho = 3$ yielding three separate pricing problem $L_1 = \{1,2\}$, $L_2 = \{1,3\}$, and $L_3 = \{2,3\}$. The left picture illustrates the network for L_2 and the right one for L_3. Note that packing combinations including product types $p = 2$ and $p = 1$ are not feasible for L_2 and L_3, respectively.

A route is a $0''$-$0'$-path alternating between vertices $V' = \{0'\} \cup \{i' : i \in N\}$ and $V'' = \{0''\} \cup \{i'' : i \in N\}$. The reduced cost can be defined as

$$\tilde{c}_{i',j''} = \tilde{c}_{i'',j'} = c_{ij} - (\mu_i + \mu_j + \rho_{ij} + \rho_{ji})/2 \qquad \forall (i,j) \in A \tag{9a}$$

$$\tilde{c}^{S_i}_{i',i''} = -\sum_{p \in S_i} \pi_{ip} \qquad \forall i \in N, S_i \subseteq P_i, S_i \subseteq L \tag{9b}$$

with $\mu_0 = \sigma + \nu_L$. All benchmark instances are symmetric, therefore, the multi-graph has also a symmetric reduced-cost structure.

The demand is modeled differently for both problem variants. For the MCVRP-CFCS, we set the demand

$$d^{S_i}_{i',i''} = \sum_{p \in S_i} d_{ip} \tag{10}$$

for all delivery edges and $d_{i'j''} = d_{i''j'} = 0$ on routing edges $\{i', j''\}$ and $\{i'', j'\}$. A $0''$-$0'$-path represents a feasible route if the accumulated demand does not exceed the vehicle capacity Q.

For the MCVRP-DFCS, we consider instead a demand vector $\boldsymbol{d}$ of dimension $|L|$ as resource with

$$\left(\boldsymbol{d}^{S_i}_{i',i''}\right)_p = \begin{cases} d_{ip} & \text{if } p \in S_i, \\ 0 & \text{otherwise,} \end{cases} \qquad p \in L, \tag{11}$$

for delivery edges and $\boldsymbol{d} = \boldsymbol{0}$ for routing edges $\{i', j''\}$ and $\{i'', j'\}$. A $0''$-$0'$-path with accumulated demand vector $\boldsymbol{d}$ represents a feasible route if

$$\sum_{p \in L} \lceil \boldsymbol{d}_p \rceil_{q^{\text{unit}}} \le Q. \tag{12}$$

The solution approach of the pricing problems is split into two phases. First, we pre-compute Pareto-optimal deliveries for each customer $i \in N$. Second, the pricing problem is solved via an SPPRC on the reduced SPPRC multi-graph only containing Pareto-optimal deliveries.

Pareto-optimal deliveries. Since the number of product types per pricing problem does not exceed ten for all benchmark instances (see Section 6.1), the number of Pareto-optimal deliveries can be determined by enumeration. The definition of Pareto-optimality differs for both problem variants. For the MCVRP-CFCS, an edge $\{i', i''\}^{S_i^1}$ is not Pareto-optimal and can be excluded if an edge $\{i', i''\}^{S_i^2}$ exists with

$$\tilde{c}_{i',i''}^{S_i^2} \leq \tilde{c}_{i',i''}^{S_i^1} \quad \text{and} \quad d_{i',i''}^{S_i^2} < d_{i',i''}^{S_i^1}. \tag{13}$$

For the MCVRP-DFCS, additionally $S_i^2 \subseteq S_i^1$ must hold. Note that the Pareto-reduction must be repeated in every column generation iteration because dual prices change in each iteration.

SPPRC over the reduced multi-graph. To solve the SPPRC on the reduced multi-graph, we use the following resources: (i) accumulated reduced cost according to (9); (ii) accumulated demand (vector) according to (10) or (11), respectively; and (iii) visit indicators for each customer $i \in N$ that are increased when one of the edges $\{i', i''\}^{S_i}$ is traversed. At the beginning, all resources are set to 0 and labels are propagated alternating between vertex sets V' and V'', i.e., in monodirectional forward labeling, a vertex i' is only propagated towards the same customer vertex i'' and vertices $i'' \in V''$ are only propagated towards a different customer vertex $j' \in V'$ with $i \neq j$. Labels are feasible if the (sum of vector entries of the) demand does not exceed Q and if the visit indicator does not exceed 1. Note that for the MCVRP-DFCS, it does not suffice to compare the accumulated demand for dominance but the demand vector must be taken into account component-by-component.

It is possible to use an implicit bidirectional labeling approach because the SPPRC is completely symmetric such that forward and backward propagation produces identical partial paths. Thereby, the computational effort can be reduced by only propagating in one direction and combining these partial paths in a merge procedure. This technique has already been applied in (Bode and Irnich 2012; Goeke *et al.* 2019; Gschwind *et al.* 2019).

5.3. Stabilization and dual inequalities

To stabilize the column generation process, dual inequalities (DIs) can be added to the dual model to the linear relaxation of (7). Let D^* be the set of optimal solutions to the dual model to the linear relaxation of (7). According to (Amor *et al.* 2006), a *dual-optimal inequality* (DOI) is defined as a DI of the form $\boldsymbol{t}^T\pi \leq t$ with $\boldsymbol{t} \in \mathbb{Z}^m$ and $t \in \mathbb{Z}$ if $D^* \subseteq \{\pi : \boldsymbol{t}^T\pi \leq t\}$. Moreover, a set of DIs $\boldsymbol{Q}^T\pi \leq \boldsymbol{q}$ with $\boldsymbol{Q} \in \mathbb{Z}^{m\times n}$ and $\boldsymbol{q} \in \mathbb{Z}^n$ comprises *deep dual-optimal inequalities* (DDOIs) if $D^* \cap \{\pi : \boldsymbol{Q}^T\pi \leq \boldsymbol{q}\} \neq \emptyset$. A general introduction to the use of DIs for the stabilization of the column generation process can be found in (Amor *et al.* 2006; Gschwind and Irnich 2016).

DIs are in general not necessarily DOIs or DDOIs for both the MCVRP-CFCS and MCVRP-DFCS. Nevertheless, it is beneficial to add DI columns at the beginning to the RMP to stabilize the column-generation process at the risk of a possible over-stabilization. The addition of DIs and possible over-stabilization resolved with a recovery procedure are explained in more detail in the following.

Static addition of dual inequalities. For each customer $i \in N$ and product pair $p, q \in P$ with $d_{ip} \leq d_{iq}$, the DIs columns corresponding to the *pair inequalities* (PI) $\pi_{ip} \leq \pi_{iq}$ are added to the initial RMP. Since the number of product types $|P_i|$ per customer i is low (less than ten for all benchmark instances) and rather many PIs are eliminated because of over-stabilization (see the paragraph below), we decided to add all PIs per customer instead of typically used *ranking inequalities* $\pi_{ip_1} \leq \pi_{ip_2} \leq \cdots \leq \pi_{ip_{|P_i|}}$ with $d_{ip_1} \leq d_{ip_2} \leq \cdots \leq d_{ip_{|P_i|}}$ (Amor *et al.* 2006). To avoid a high number of recovery procedure iterations, we do not add further DIs of the form $\pi_{ip} \leq \sum_{p\in S} \pi_{ip}$ with $S \subseteq P_i$, so-called *subset inequalities*, that strongly influence the compartment composition of the solution routes. Moreover, we do not identify violated DIs during the pricing approach and add them dynamically to the master problem (7).

Over-stabilization and recovery procedure. Note that in general PIs are neither DOIs nor DDOIs such that all dual-optimal solutions are cut-off. This possible over-stabilization can be purged by a recovery procedure proposed in (Gschwind and Irnich 2016). Given the RMP solution, this procedure tries to build a pure route-columns solution. If this is not possible, i.e. a DI column corresponding to $\pi_{ip} \leq \pi_{iq}$ with a positive value exists that is not compatible with any route column, then the RMP is over-stabilized. In this case, the recovery procedure eliminates all PIs $\pi_{i\bar{p}} \leq \pi_{iq}$ with $\bar{p} \in P_i$ from the RMP. Afterwards, the column generation process restarts and iterates until a pure route-columns solution exists. Note that a DI column is classified incompatible with a route column if either the resulting route column exceeds the number of compartments C or a product type is delivered twice.

5.4. *Valid inequalities and cutting strategy*

Three classes of valid inequalities are added to the RMP during the solution process. On the one hand we add two families of non-robust cuts, namely *subset-row inequalities* (SR inequalities) (Jepsen *et al.* 2008) for subsets of cardinality three and *strong-degree constraints* (SD constraints) (Contardo *et al.* 2014). Subset-row inequalities for subsets of cardinality three ensure for elementary routes that at most one route that fulfills at least two of three tasks is part of a feasible solution. Strong-degree constraints ensure that a demand d_{ip} with $i \in N$ and $p \in P$ is served by at least one elementary or non-elementary route. The definition of these non-robust cuts including the impact on DIs is the same for both MCVRP-FCS variants as for the C-SDVRP. Therefore, we refer to (Gschwind *et al.* 2019) for a detailed description. On the other hand, we add the family of robust capacity cuts (Fukasawa *et al.* 2005) that are described in detail in the following.

Let $S \subseteq N$, $S \neq \varnothing$, be a customer subset and let $\delta^-(S)$ denote the arcs of the digraph $G = (V, A)$ with $i \notin S$ and $j \in S$. Then, we can formulate the *capacity cut* (CC)

$$\sum_{r\in\Omega}\left(\sum_{(i,j)\in\delta^-(S)} b^r_{ij}\right)\lambda^r \geq \max\left\{\left\lceil \frac{\sum_{i\in S}\sum_{p\in P_i} d_{ip}}{Q}\right\rceil, \left\lceil \frac{|\{p \in P_i : i \in S\}|}{C}\right\rceil\right\} \tag{14}$$

with corresponding dual price γ_S. The right-hand side does not only consider the vehicle capacity Q but also the available number of compartments C. These cuts are robust because the value $\gamma_S/2$ can be distributed symmetrically on the edges (i', j'') and (i'', j') for all $(i, j) \in \delta^-(S)$ of the undirected SPPRC pricing network.

Overall cutting strategy. The cut-generation strategy depends on the MCVRP-FCS variant and the underlying instance. Since the number of compartments C is typically more restrictive than the vehicle capacity Q, SR inequalities and SD constraints are less effective compared to the C-SDVRP. Moreover, both cutting types influence the Pareto-reduction and are therefore not used at all or only up to level three in the branch-and-bound tree (for details see Section 6.3). Of course, SD constraints are additionally added deeper in the tree if needed to guarantee elementary routes for the completeness of the branching rule (cf. Section 5.5).

In contrary, CCs are very effective for both MCVRP-FCS variants. Therefore, the following CCs are already added at the beginning to the initial RMP. For each customer $i \in N$ with $|P_i| > C$, we add a capacity cut for subset $S = \{i\}$ if the right-hand side of (14) is at least 2. Moreover, let $r_i(j)$ be a ranking function ordering the neighbors of i by travel cost, i.e. $r_i(j_1) = 1, r_i(j_2) = 2, \ldots$ for ordered travel costs $c_{i,j_1} \leq c_{i,j_2} \leq \ldots$ for $j_1, j_2, \dots \in N$. For each $i \in N$, we add all CCs for customer subsets $S = \{i, j\} \subseteq N$ with $P_i \cup P_j \neq P_i \cap P_j$, $|P_i \cup P_j| > C$, and minimal ranking function $r_i(j)$. Additionally, for instances with three or more available vehicles, we sort for each customer $i \in N$ the neighbors $j_1, j_2, \dots \in N$ according to the ranking function, i.e. $r_i(j_1) < r_i(j_2) < \ldots$, and add a CC for the smallest subset $S = \{i, j_1, j_2, \ldots\}$ with the right-hand side of (14) equal to 3.

5.5. *Branching*

In the following, we briefly summarize the six-level branching strategy that is similar to the one applied in (Archetti *et al.* 2015; Gschwind *et al.* 2019). At the first level, we branch on the number of vehicles and at the second level, we branch on the number of routes that are packed with compartment combination $L \in X$ (see constraints (7f)-(7g)). At the third level, we branch on the number of visits to each customer. Note that infeasible subsets P_i are eliminated from the customer network if possible. At the fourth level, we branch on the edge flow. Again, edges can be eliminated from the customer network for zero-flow decisions. At level five and six, we use Ryan-Foster-branching for supplies at the same customer and different customers, respectively. Up to level six the branching scheme is not complete because non-elementary routes can still be part of the solution. To guarantee elementary routes, we separate SD constraints if all other values considered at branching levels one to six are integer. For an explanation for the completeness of the branching scheme and the impact of branching on DIs, we refer to (Gschwind *et al.* 2019; p. 97).

To improve the dual bound as fast as possible, we use a best-bound first tree exploration strategy. The branching variable is selected as the one with the fractional part closest to 0.5.

5.6. Acceleration techniques

To relax the elementary SPPRC, we use the ng-path relaxation proposed by Baldacci *et al.* (2011) that prohibits cycles in a pre-defined neighborhood of vertex i but allows cycles over vertex j if j is not in the neighborhood of i. For larger neighborhood sizes, fewer cycles are possible but the computational effort increases on average. In our case, a good tradeoff between neighborhood size and computational effort is obtained with a neighborhood size of ten.

Moreover, the SPPRC is solved first heuristically on several reduced SPPRC network at each iteration to accelerate the column generation process. We consider two types of reduction techniques. The first one reduces the size of the customer network according to delivery edges. Considering Pareto-optimal deliveries, we only use the best or three best product combinations for each customer $i \in N$, i.e. $S_i^* = \arg\min_{S_i \subseteq P_i} \tilde{c}_{i',i''}^{S_i}$ or $S_i^* = \{S_{i_1}, S_{i_2}, S_{i_3}\}$ with $\tilde{c}_{i',i''}^{S_{i1}}, \tilde{c}_{i',i''}^{S_{i2}}, \tilde{c}_{i',i''}^{S_{i3}}$ minimal, respectively. Let $D^{del} = 1, 3$ denote this relaxation and $D^{del} = S^*$ all Pareto-optimal deliveries, respectively. The second type of reduction technique reduces the size of the customer network according to routing edges. We limit the number of edges D^{adj} adjacent to a customer by 2, 5, and 10. Additionally, we only consider edges between customers and the depot as well as edges between customers belonging to the pre-calculated TSP-tour over all vertices. Let D^{adj} = TSP denote this relaxation. Combining both reduction techniques and considering different pricing problems, the overall pricing strategy is depicted in Algorithm 3.

Algorithm 3: Heuristic pricing strategy

input : dual prices for the SPPRC network
output: negative reduced cost columns or information that no such column exists

for $D^{del} \in \{1, 3, S^*\}$ **do**
  for $D^{adj} \in \{2, TSP, 5, 10, |n|\}$ **do**
    for *randomly sorted* $L \in X$ **do**
      Solve pricing problem L for the reduced SPPRC network with delivery edges D^{del} and routing edges D^{adj};
      if *at least one negative reduced cost column is found* **then**
        return columns;
return information that no negative reduced cost column exist;

6. Computational results

In this section, we first give an overview of the benchmark instances and then describe details of the implementation. After presenting an overview of pretests and the computational setup, the section closes with detailed results and a comparison between the algorithms and total costs for both MCVRP-FCS variants.

6.1. Benchmark instances

In total, we consider three sets of `small(H15)`, `mid-size(H18)`, and `large(H15)` benchmark instances. All instances are characterized by three parameters: the number of product types ρ, the number of available compartments per vehicle C, and a supply parameter s that denotes if the total number of supplies is small ($s = 1$), medium ($s = 2$), or large ($s = 3$). Note that the classification into `small(H15)`, `mid-size(H18)`, and `large(H15)` only depends on the number of vertices $|V|$ and not on other parameters (especially not on the supply parameter s). The same set of benchmark instances can be used for both problem variants. For the MCVRP-DFCS, the unit compartment size is set to $q^{\text{unit}} = 0.1\,Q$, i.e., the vehicle is divided into 10 basic compartment units. Note that the vehicle capacity Q is divisible by ten for all benchmark instances. If the number of product types equals the number of compartments, i.e., $\rho = C$, then the number of compartments does not restrict the feasible region and the MCVRP-CFCS is actually a C-SDVRP.

The first set of `mid-size(H18)` instances is proposed in (Henke *et al.* 2018) and consists of 675 instances with 10 to 50 vertices. The number of product types is $\rho = 3, 4$, the maximal number of compartments is $C = 2, 3, 4$, and the supply parameter is $s = 1, 2, 3$. The instances are constructed in such a way that the number of vehicles $m = 2, 3$ is relatively constant.

The second and third set of benchmark instances are introduced in (Henke *et al.* 2015). For these instances, the number of product types $\rho = 3, 6, 9$ and the maximal number of compartments $C = 2, 3, 4, 6, 7, 9$ are larger compared to the first set of instances while the supply parameter is again $s = 1, 2, 3$. Moreover, the number of vehicles is not relatively constant but is higher for instances with more customers and total demand. Originally, the second set contained 1350 instances with 10 vertices. Because the instances are small and rather easy to solve, we only use a subset of 135 `small(H15)` instances that consists of 5 (instead of 50) instances for each ρ-C-s-combination. The third set of 27 `large(H15)` instances with 50 vertices contains one instance for each ρ-C-s-combination.

6.2. Details of the implementation

The branch-and-cut algorithms are implemented in C++ using CPLEX 12.10.0 with Concert Technology. For the branch-price-and-cut algorithm, the RMP is also solved utilizing CPLEX at each column-generation iteration. Moreover, CPLEX is used as a primal MIP-based heuristic solver after the solution of each branch-and-bound node using all generated but DI columns. All algorithms are compiled into 64-bit single-thread code with Microsoft Visual Studio 2015. The computational study is carried out on a 64-bit Microsoft Windows 10 computer with an Intel® Core™ i7-5930k CPU clocked at 3.5 GHz and 64 GB of RAM. For the separation procedure of the branch-and-cut algorithms, generic callbacks are used for both user and lazy cuts. According to (Henke *et al.* 2018), computation times are limited to a maximum of 7200 seconds (2 hours). Apart from the number of threads and the time limit, CPLEX's default values are kept for all parameters.

6.3. Pretests and computational setup

In this section, we specify the solution approaches that are compared for both problem variants. Pretests showed that it is beneficial to combine two of the three solution approaches (see details below). Table 2 shows an overview of all solution approaches that are explained in detail in the following.

First of all, the branch-price-and-cut algorithm proposed in Section 5 is called `BaP`. Moreover, we refer to the branch-and-cut algorithms based on the three-index and two-index formulation as `ThreeIndex` (for the continuous variant), `ThreeIndexDiscrete` (for the discrete variant) and `TwoIndex` (for both variants), respectively. Henke *et al.* (2018) propose to solve the MCVRP-CFCS with the three-index formulation for the MCVRP-DFCD and unit size $q^{\text{unit}} = 1$. We also refer to this version as `ThreeIndexDiscrete`.

For the branch-price-and-cut algorithm, pretests showed that the following settings are beneficial. As stated in Section 5.4, some CCs are already added at the beginning to the initial RMP and the cut-generation of SR inequalities and SD constraints depends on the underlying instance and problem variant. For the MCVRP-CFCS, SR inequalities and SD constraints affect the Pareto-reduction and have a strong impact on computation times unless the supply parameter is $s = 1$. Therefore, we do not use SR inequalities at all and SD constraints within the first levels of the branch-and-bound tree for the MCVRP-CFCS for instances

Table 2: Overview of solution approaches.

name	MCVRP-CFCS	MCVRP-DFCS
`ThreeIndex`	branch-and-cut of the three-index formulation (see Section 3)	
`ThreeIndexDiscrete`	branch-and-cut of Henke *et al.* (2018) with $q^{\text{unit}} = 1$ (see Section 3)	branch-and-cut of Henke *et al.* (2018) (see Section 3)
`TwoIndex`	branch-and-cut of the two-index formulation (see Section 4)	
`BaP`	branch-price-and-cut (see Section 5)	
`BaP+ThreeIndex`	two-stage-approach combining `BaP` and `ThreeIndex`	
`BaP+TwoIndex`	two-stage-approach combining `BaP` and `TwoIndex`	

with supply parameter $s = 2, 3$, respectively. SD constraints are only separated if all values at all branching levels are integer. For the MCVRP-DFCS, the Pareto-reduction is less effective and SR inequalities and SD constraints are used up to level three in the branch-and-bound tree.

The computation time of `ThreeIndex` mainly depends on the number of vertices. Instead, the computational performance of both `TwoIndex` and the `BaP` is instance-specific and does not follow obvious rules. Moreover, the solution times are discrepant for these algorithms for some instances as depicted in Table 3. Note that the entries of all result tables have the following meaning:

#opt: number of instances solved to proven optimality within 2 hours (7200 seconds);
time $\bar{T}$: average computation time in seconds; unsolved instances are taken into account with the time limit TL of 2 hours (7200 seconds);
gap: $100 \cdot (UB - LB)/LB$, i.e., the gap in percent;
No.: instance number.

To take advantage of the obvious discrepancy in the computation times, we combine both algorithms by first solving the problem with `BaP`. If no optimal solution is found after 60 seconds or 1000 generated columns, the MIP solver is called to find a good feasible solution and the problem is solved by `TwoIndex`. Using the feasible solution as upper bound for the branch-and-cut algorithm reduces the size of the branch-and-bound tree and, therefore, yields better computational performance. We refer to this variant as `BaP+TwoIndex`. To test the influence of using upper bounds for the branch-and-cut algorithm of the three-index formulation depicted in Section 3, we also consider the variant `BaP+ThreeIndex` in which the `BaP` is analogously interrupted after 60 seconds or 1000 generated columns.

Table 3: Instances for the MCVRP-CFCS with contrary computation times for the `TwoIndex` and `BaP` approach and summarized results for `mid-size(H18)` instances with $|V| = 10, 15$.

Instances					`TwoIndex`			`BaP`			`BaP+TwoIndex`		
$\|V\|$	ρ	C	s	No.	#opt	time $\bar{T}$	gap	#opt	time $\bar{T}$	gap	#opt	time $\bar{T}$	gap
10	4	2	3	1	1	516.8	**0.0**	1	30.8	**0.0**	1	**24.1**	**0.0**
10	4	2	3	3	1	1736.5	**0.0**	1	**91.0**	**0.0**	1	706.7	**0.0**
15	3	2	2	3	1	**1.9**	**0.0**	1	529.6	**0.0**	1	62.0	**0.0**
15	3	3	2	4	1	**0.1**	**0.0**	1	1119.9	**0.0**	1	63.5	**0.0**
Total					4	563.9	**0.0**	4	442.8	**0.0**	4	**214.1**	**0.0**
Total ($\|V\| = 10$)					**75**	46.0	**0.0**	**75**	**7.4**	**0.0**	**75**	24.0	**0.0**
Total ($\|V\| = 15$)					**69**	791.4	**0.3**	63	1653.8	13.4	**69**	**681.0**	**0.3**

6.4. *Results for the MCVRP-CFCS*

To compare the algorithms, we first consider `mid-size(H18)` benchmark instances with a lower number of product types ρ compared to the `small(H15)` and `large(H15)` benchmark instances. As mentioned before, we combine the `BaP` approach with both the `ThreeIndex` and `TwoIndex` approach. The results clustered according to the number of supplies are summarized in Table 4. Overall, the `ThreeIndex` approach can solve most of the instances to proven optimality and has the lowest average computation time. The `ThreeIndexDiscrete` is slightly inferior but can solve one more instance with supply parameter $s = 1$ exactly and is on average a bit faster for instances with supply parameter $s = 2$. Using the `BaP` approach for upper bounds up to 60 seconds does not speed up the `ThreeIndex` approach on average. The `BaP+TwoIndex` approach is altogether inferior but is almost 50 % faster for instances with supply parameter $s = 1$.

Table 4: MCVRP-CFCS results for `mid-size(H18)` instances clustered according to the number of supplies.

Instances		ThreeIndex			BaP+ThreeIndex			BaP+TwoIndex			ThreeIndexDiscrete (Henke *et al.* 2018)		
s	#inst	#opt	time $\bar{T}$	gap	#opt	time $\bar{T}$	gap	#opt	time $\bar{T}$	gap	#opt	time $\bar{T}$	gap
1	225	210	602.4	**0.5**	210	597.2	**0.5**	**217**	**348.8**	0.6	211	632.6	**0.5**
2	225	**209**	914.9	**0.3**	208	904.4	**0.3**	153	2373.1	17.0	205	**890.8**	**0.3**
3	225	**209**	**811.9**	**0.5**	208	857.3	**0.5**	123	3346.0	36.0	207	861.9	**0.5**
Total	675	**628**	**776.4**	**0.4**	626	786.3	**0.4**	493	2022.6	17.9	623	795.1	**0.4**

Given an instance with unknown supply parameter s, it is very simple to classify the instance according to s. Therefore, we suggest applying the `BaP+TwoIndex` approach for instances with $s = 1$ and the `ThreeIndex` approach for instances with $s = 2, 3$, respectively. In the following, we refer to this combined approach as `ThreeIndex/BaP+TwoIndex`. The results clustered according to the number of vertices are summarized in Table 5. Excluding results of `ThreeIndex/BaP+TwoIndex`, the `ThreeIndex` approach performs best and is superior in almost all clusters. The computation time of `BaP+TwoIndex` is on average considerably slower. Nevertheless, combing both approaches yields seven more optimally solved instances and reduces the average computation time by around 80 seconds compared to the `ThreeIndex` approach. In total, the `ThreeIndex/BaP+TwoIndex` approach can solve 635 of 675 instances to proven optimality.

Table 5: MCVRP-CFCS results for `mid-size(H18)` instances clustered according to the number of vertices.

Instances		ThreeIndex			BaP+ThreeIndex			BaP+TwoIndex			ThreeIndexDiscrete (Henke *et al.* 2018)			ThreeIndex/ BaP+TwoIndex		
$\lvert V\rvert$	#inst	#opt	time $\bar{T}$	gap	#opt	time $\bar{T}$	gap	#opt	time $\bar{T}$	gap	#opt	time $\bar{T}$	gap	#opt	time $\bar{T}$	gap
10	75	**75**	0.9	**0.0**	**75**	3.4	**0.0**	**75**	24.0	**0.0**	**75**	1.0	**0.0**	**75**	**0.8**	**0.0**
15	75	**75**	4.0	**0.0**	**75**	32.4	**0.0**	69	681.0	0.3	**75**	5.2	**0.0**	**75**	**3.8**	**0.0**
20	75	**75**	28.8	**0.0**	**75**	57.3	**0.0**	61	1486.9	8.5	**75**	**22.6**	**0.0**	**75**	28.0	**0.0**
25	75	**75**	107.7	**0.0**	**75**	132.1	**0.0**	55	2052.6	14.3	**75**	147.5	**0.0**	**75**	**84.9**	**0.0**
30	75	**75**	214.0	**0.0**	**75**	209.4	**0.0**	51	2361.2	23.9	**75**	280.1	**0.0**	**75**	**192.8**	**0.0**
35	75	**72**	845.4	**0.1**	**72**	757.5	**0.1**	49	2587.5	23.9	**72**	**658.7**	**0.1**	**72**	824.3	**0.1**
40	75	**66**	1407.5	0.7	65	1423.8	**0.6**	45	2933.4	30.1	**66**	**1338.3**	**0.6**	65	1428.4	1.7
45	75	58	2205.3	1.4	58	2193.5	1.5	45	2924.0	28.4	55	2294.2	1.4	**61**	**1949.0**	**1.0**
50	75	57	2174.2	1.8	56	2267.3	1.7	43	3153.3	31.4	55	2408.4	1.6	**62**	**1715.0**	**1.4**
Total	675	628	776.4	**0.4**	626	786.3	**0.4**	493	2022.6	17.9	623	795.1	**0.4**	**635**	**691.9**	0.5

The so-far best-performing algorithms `ThreeIndex`, `BaP+TwoIndex`, and `ThreeIndex/BaP+TwoIndex` are also tested for `small(H15)` and `large(H15)` instances, both with a higher number of product types compared to the `mid-size(H18)` instances. Additionally, we test the `BaP` approach for these instances because symmetry issues are more relevant for the `TwoIndex` approach for larger ρ. Results clustered according to the number of product types and the supply parameter are summarized in Table 6. Note that we only report the gap if an upper bound is found.

For the `small(H15)` instances with only 10 vertices, the `ThreeIndex` approach can solve all instances with an average computation time of 37.2 seconds to proven optimality. The other approaches, `BaP` and `BaP+TwoIndex`, only perform appropriately for instances with supply parameter $s = 1$. Also the `ThreeIndex/BaP+TwoIndex` approach can solve one instance less with an average computation of about 50 seconds more compared to the `ThreeIndex` approach.

For `large(H15)` instances, the performance of the algorithms is different. The `ThreeIndex` approach cannot solve any of the instances and cannot even find a feasible solution. The reason for the poor performance is most likely that the number of vehicles is on average three times higher for `large(H15)` instances compared to `small(H15)` and `mid-size(H18)` instances and, therefore, symmetry issues outweigh. Note that the number of available vehicles for `mid-size(H18)` (`large(H15)`) instances is on average 2.7 (8.4). The `BaP` approach can at least solve four instances to proven optimality and the `BaP+TwoIndex` approach performs best with six instances solved to proven optimality. In total, we can find the optimal solution for 6 of 27 `large(H15)` instances.

Summarized we advise using the `ThreeIndex` approach for instances with a low number of vertices $|V|$. For `mid-size(H18)` and `large(H15)` instances, we recommend solving the instance with the `ThreeIndex/BaP+TwoIndex` approach. Combining all results, we can solve all 135 `small(H15)` instances, 643 of 658 `mid-size(H18)` instances and 6 of 27 `large(H15)` instances. Instance-by-instance results are listed in the Online Appendix.

Table 6: MCVRP-CFCS results for `small(H15)` and `large(H15)` instances clustered according to the number of product types and the supply parameter.

Instance					ThreeIndex			BaP			BaP+TwoIndex			ThreeIndex/BaP+TwoIndex		
class	$\lvert V\rvert$	ρ	s	#inst	#opt	time $\bar{T}$	gap	#opt	time $\bar{T}$	gap	#opt	time $\bar{T}$	gap			
small	10	3	1	10	**10**	0.2	**0.0**	**10**	**0.1**	**0.0**	**10**	0.2	**0.0**	**10**	0.2	**0.0**
			2	10	**10**	**0.4**	**0.0**	**10**	0.9	**0.0**	**10**	0.9	**0.0**	**10**	**0.4**	**0.0**
			3	10	**10**	**0.5**	**0.0**	**10**	3.0	**0.0**	**10**	2.8	**0.0**	**10**	**0.5**	**0.0**
		6	1	15	**15**	**0.4**	**0.0**	**15**	1.2	**0.0**	**15**	1.2	**0.0**	**15**	1.2	**0.0**
			2	15	**15**	**1.1**	**0.0**	**15**	40.5	**0.0**	12	1477.2	0.8	**15**	**1.1**	**0.0**
			3	15	**15**	**2.2**	**0.0**	14	531.0	4.3	12	1515.1	6.7	**15**	**2.2**	**0.0**
		9	1	20	**20**	**12.3**	**0.0**	**20**	12.9	**0.0**	19	367.8	< 0.1	19	367.8	< 0.1
			2	20	**20**	**85.6**	**0.0**	18	1287.2	0.3	9	4028.5	13.8	**20**	**85.6**	**0.0**
			3	20	**20**	**149.9**	**0.0**	11	4317.9	2.1	5	5404.6	34.7	**20**	**149.9**	**0.0**
Total($\lvert V\rvert = 10$)				135	**135**	**37.2**	**0.0**	123	896.2	0.8	102	1784.9	8.0	134	90.0	**0.0**
large	50	3	1	2	0	*TL*		1	3615.6	< 0.1	**2**	**278.3**	**0.0**	**2**	**278.3**	**0.0**
			2	2	0	*TL*		0	*TL*	40.3	0	*TL*	15.5	0	*TL*	
			3	2	0	*TL*		0	*TL*		0	*TL*	50.7	0	*TL*	
		6	1	3	0	*TL*		**2**	4203.4	**1.0**	**2**	**2496.4**	29.3	**2**	**2496.4**	29.3
			2	3	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
			3	3	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
		9	1	4	0	*TL*		1	6983.1		**2**	**4131.1**		**2**	**4131.1**	
			2	4	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
			3	4	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
Total($\lvert V\rvert = 50$)				27	0	*TL*		4	6569.4		**6**	**5710.0**		**6**	**5710.0**	

6.5. Results for the MCVRP-DFCS

For this problem variant, we also consider the `mid-size(H18)` benchmark instances with a lower number of product types ρ first. The results clustered according to the number of supplies are summarized in Table 7. The performance of the algorithms is similar to the MCVRP-CFCS variant. Again, using the `BaP` approach for upper bounds up to 60 seconds does not speed up the `ThreeIndexDiscrete` approach on average. Overall, the `ThreeIndexDiscrete` approach performs best but the `BaP+TwoIndex` is superior for instances with supply parameter $s = 1$. Due to the different performance of the algorithms for different supply parameters s, we also consider the `ThreeIndexDiscrete/BaP+TwoIndex` approach.

Table 7: MCVRP-DFCS results for `mid-size(H18)` instances clustered according to the number of supplies.

Instances		`ThreeIndexDiscrete` (Henke *et al.* 2018)			`BaP+ThreeIndexDiscrete`			`BaP+TwoIndex`		
s	#inst	#opt	time $\bar{T}$	gap	#opt	time $\bar{T}$	gap	#opt	time $\bar{T}$	gap
1	225	208	704.9	**0.6**	208	719.7	**0.6**	**214**	**562.3**	0.7
2	225	**206**	**1025.1**	**0.3**	**206**	1061.1	0.4	152	2586.2	20.4
3	225	**208**	**871.1**	**0.5**	207	941.2	0.6	114	3634.6	39.0
Total	675	**622**	**867.0**	**0.5**	621	907.3	**0.5**	480	2261.0	20.0

Results clustered according to the number of vertices can be found in Table 8. Excluding results of `ThreeIndexDiscrete/BaP+TwoIndex`, the `ThreeIndexDiscrete` approach is the best performing single-stage approach with 622 of 675 optimally solved instances and an average computation time of 867.0 seconds. For this problem variant, the `ThreeIndexDiscrete/BaP+TwoIndex` approach can solve 628 of 675 instances to proven optimality. Compared to the MCVRP-CFCS variant, we can solve 7 instances less and the average computation time increases by around 120 seconds (2 minutes).

Table 8: MCVRP-DFCS results for `mid-size(H18)` instances clustered according to the number of vertices.

Instances		`ThreeIndexDiscrete` (Henke *et al.* 2018)			`BaP+ThreeIndexDiscrete`			`BaP+TwoIndex`			`ThreeIndexDiscrete/` `BaP+TwoIndex`		
$\lvert V\rvert$	#inst	#opt	time $\bar{T}$	gap	#opt	time $\bar{T}$	gap	#opt	time $\bar{T}$	gap	#opt	time $\bar{T}$	gap
10	75	**75**	1.2	**0.0**	**75**	19.4	**0.0**	74	193.1	< 0.1	**75**	**1.1**	**0.0**
15	75	**75**	**5.5**	**0.0**	**75**	43.3	**0.0**	68	833.6	2.8	**75**	6.7	**0.0**
20	75	**75**	**37.7**	**0.0**	**75**	78.3	**0.0**	65	1280.3	8.0	**75**	41.5	**0.0**
25	75	**74**	**190.8**	**0.0**	**74**	238.4	**0.0**	53	2395.6	17.2	**74**	274.2	**0.0**
30	75	**75**	**367.8**	**0.0**	74	379.4	**0.0**	49	2778.6	26.8	**75**	488.5	**0.0**
35	75	**72**	**928.2**	**0.2**	72	969.5	**0.2**	46	3016.3	27.9	71	1022.8	**0.2**
40	75	**65**	1365.5	**0.7**	**65**	1428.6	**0.7**	44	2996.4	32.0	**65**	**1332.4**	1.7
45	75	57	2336.8	1.3	57	2394.9	1.4	41	3484.1	30.3	**58**	**2168.3**	**1.2**
50	75	54	2569.6	2.2	54	2614.0	2.2	40	3371.0	35.3	**60**	**2039.6**	**1.5**
Total	675	622	867.0	**0.5**	621	907.3	**0.5**	480	2261.0	20.0	**628**	**819.5**	**0.5**

Results for `small(H15)` and `large(H15)` instances are depicted in Table 9. Again, the performance of the algorithms is similar to the MCVRP-CFCS variant. For `small(H15)` instances, the `ThreeIndexDiscrete` approach can solve all instances to proven optimality and is very fast with an average computation time of 36.5 seconds compared to the other algorithms. The `large(H15)` instances are very hard to solve for this problem variant. Only one instance can be solved by the `BaP+TwoIndex` approach. Contrary to the MCVRP-CFCS variant, the `BaP` approach performs best with two instances solved to proven optimality.

Overall, we recommend using the `ThreeIndexDiscrete` approach for `small(H15)` instances with a low number vertices $|V|$ and the `ThreeIndexDiscrete/BaP+TwoIndex` approach for `mid-size(H18)` instances. For `large(H15)` instances, the `BaP` and `BaP+TwoIndex` approach perform best, most likely, because the number of available vehicles has again a high impact on symmetry issues of the `ThreeIndex` approach. Combining all results, we can solve all 135 `small(H15)` instances, 638 of 658 `mid-size(H18)` instances and 2 of 27 `large(H15)` instances. Note that instance-by-instance results are listed in the Online Appendix.

Table 9: MCVRP-DFCS results for `small(H15)` and `large(H15)` instances clustered according to the number of product types and the supply parameter.

Instance					`ThreeIndexDiscrete` (Henke *et al.* 2018)			`BaP`			`BaP+TwoIndex`			`ThreeIndexDiscrete`/ `BaP+TwoIndex`		
class	$\lvert V\rvert$	ρ	s	#inst	#opt	time $\bar{T}$	gap	#opt	time $\bar{T}$	gap	#opt	time $\bar{T}$	gap			
`small`	10	3	1	10	**10**	0.2	**0.0**	**10**	**0.1**	**0.0**	**10**	0.2	**0.0**	**10**	0.2	**0.0**
			2	10	**10**	**0.3**	**0.0**	**10**	1.2	**0.0**	**10**	1.2	**0.0**	**10**	**0.3**	**0.0**
			3	10	**10**	**0.5**	**0.0**	**10**	23.7	**0.0**	**10**	9.5	**0.0**	**10**	**0.5**	**0.0**
		6	1	15	**15**	**0.3**	**0.0**	**15**	2.3	**0.0**	**15**	2.3	**0.0**	**15**	2.3	**0.0**
			2	15	**15**	**1.1**	**0.0**	10	2607.2	1.0	12	1506.2	0.8	**15**	**1.1**	**0.0**
			3	15	**15**	**3.1**	**0.0**	6	4965.2	60.1	9	4199.9	31.3	**15**	**3.1**	**0.0**
		9	1	20	**20**	**16.6**	**0.0**	19	381.1	0.0	**20**	103.5	**0.0**	**20**	103.5	**0.0**
			2	20	**20**	**66.8**	**0.0**	8	4892.3	50.6	7	4871.7	28.4	**20**	**66.8**	**0.0**
			3	20	**20**	**159.2**	**0.0**	0	*TL*	71.0	1	6930.9	73.7	**20**	**159.2**	**0.0**
Total($\lvert V\rvert$ = 10)				135	**135**	**36.5**	**0.0**	88	2691.4	24.8	94	2398.9	18.7	**135**	49.6	**0.0**
`large`	50	3	1	2	0	*TL*		**1**	**3946.0**		**1**	4402.6		**1**	4402.6	
			2	2	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
			3	2	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
		6	1	3	0	*TL*		**1**	**7095.8**		0	*TL*		0	*TL*	
			2	3	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
			3	3	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
		9	1	4	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
			2	4	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
			3	4	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
Total($\lvert V\rvert$ = 50)				27	0	*TL*		**2**	**6947.4**		1	6992.8		1	6992.8	

6.6. *Cost comparison between MCVRP-CFCS and MCVRP-DFCS*

In this section, we compare the total cost of continuously flexible compartment sizes with the total cost of discretely flexible compartment sizes. We only consider instances that are optimally solved for both MCVRP-FCS variants. Note that `large(H15)` instances are not taken into account because only a few instances are solved to proven optimality. Nevertheless, we observe that the total costs differ for 7 `large(H15)` instances and so far no `large(H15)` instance with identical total costs for both MCVRP-FCS variants is known. Table 10 displays the total cost comparison clustered according to the number of vertices, Table 11 clustered according to the number of supplies, and Table 12 clustered according to the number of product types. The table entries have the following meaning:

- **set:** benchmark instance set;
- **#div:** number of instances with different total cost, i.e., number of instances with $z_{con} \neq z_{dis}$;
- **div(%):** percentage of instances with different total cost, i.e., $100 \cdot$ #div/#inst;
- $\bar{z}_{con}$**:** average total cost of the MCVRP-CFCS variant;
- $\bar{z}_{dis}$**:** average total cost of the MCVRP-DFCS variant;
- **red(%):** the average reduction of the total cost in percent, i.e., the average of $100 \cdot (z_{dis} - z_{con})/z_{dis}$.

According to Table 10, the number of instances with different total cost does not depend on the number of vertices but cost savings are on average higher for instances with a lower number of vertices.

Table 11 shows that the supply parameter s impacts the cost savings of continuously flexible compartment sizes compared to discretely flexible compartment sizes. Cost savings are on average higher for instances with a smaller supply parameter.

Table 12 shows that cost savings are higher for instances with a higher number of product types. Moreover, the number of instances with different total cost increases for instances with a higher number of product types.

Table 10: Total cost comparison of `mid-size(H18)` instances clusterd according to the number of vertices.

$\lvert V\rvert$	#inst	#div	div(%)	$\bar{z}_{con}$	$\bar{z}_{dis}$	red(%)
10	75	43	57.3	433.8	447.6	3.1
15	75	47	62.7	508.2	518.6	2.0
20	75	45	60.0	557.9	566.2	1.5
25	75	51	68.0	617.6	626.3	1.4
30	75	50	66.7	654.6	663.0	1.3
35	73	55	75.3	694.1	702.8	1.2
40	66	45	68.2	722.6	730.5	1.1
45	60	41	68.3	724.9	733.2	1.1
50	59	37	62.7	753.2	759.4	0.8
Total	633	414	65.4	622.8	631.8	1.4

Table 11: Total cost comparison of `small(H15)` and `mid-size(H18)` instances clusterd according to the number of supplies.

set	$\lvert V\rvert$	s	#inst	#div	div(%)	$\bar{z}_{con}$	$\bar{z}_{dis}$	red(%)
`small(H15)`	10	1	45	21	46.7	423.9	432.4	2.0
	10	2	45	18	40.0	553.4	562.5	1.6
	10	3	45	10	22.2	658.3	666.1	1.2
Total			135	49	36.3	545.2	553.6	1.5
`mid-size(H18)`	10–50	1	220	144	65.5	545.4	555.2	1.8
	10–50	2	204	163	79.9	614.3	625.0	1.7
	10–50	3	209	107	51.2	712.4	719.1	0.9
Total			633	414	65.4	622.8	631.8	1.4

Table 12: Total cost comparison of `small(H15)` and `mid-size(H18)` instances clusterd according to the number of product types.

set	$\lvert V\rvert$	ρ	#inst	#div	div(%)	$\bar{z}_{con}$	$\bar{z}_{dis}$	red(%)
`small(H15)`	10	3	30	8	26.7	412.2	414.6	0.6
	10	6	45	14	31.1	523.6	528.4	0.9
	10	9	60	27	45.0	627.8	642.1	2.2
Total			135	49	36.3	545.2	553.6	1.5
`mid-size(H18)`	10–50	3	262	172	65.6	608.7	615.8	1.2
	10–50	4	371	242	65.2	632.7	643.1	1.6
Total			633	414	65.4	622.8	631.8	1.4

7. Conclusion

In this paper, we provided three new exact solution approaches for the multi-compartment vehicle routing problem with continuous flexible compartment sizes and two new exact solution approaches for the multi-compartment vehicle routing problem with discrete flexible compartment sizes. Computational tests have been conducted on benchmark instances from the literature. We identified that the performance of the algorithms depends on the instance parameters. For the MCVRP-CFCS and MCVRP-DFCS, a branch-and-cut algorithm based on a three-index formulation performs best for `small(H15)` instances with a low number of vertices. A combined algorithm of this branch-and-cut algorithm for instances with high supplies per customer and a two-stage approach consisting of a branch-price-and-cut and a branch-and-cut algorithm of a two-index formulation turned out to perform best for `mid-size(H18)` instances with a low number of vehicles. For the former type (MCVRP-CFCS), the algorithms can solve all `small(H15)` instances and `mid-size(H18)` instances with up to 30 vertices and over 80% of the `mid-size(H18)` instances with 50 vertices to optimality within two hours. Moreover, the two-stage approach consisting of the branch-price-and-cut and the branch-and-cut algorithm of the two-index formulation can solve 6 of 27 `large(H15)` instances. For the latter type (MCVRP-DFCS), the algorithms deliver new provably optimal solutions for 16 `mid-size(H18)` instances and 2 `large(H15)` instances. A comparison between the total costs of both variants shows that the savings potential of using continuously flexible compartment sizes instead of discretely flexible compartment sizes depends on average on the number of vertices, the number of supplies, and the number of product types.

References

Amor, H. B., Desrosiers, J., and de Carvalho, J. M. V. (2006). Dual-optimal inequalities for stabilized column generation. *Operations Research*, **54**(3), 454–463.

Archetti, C., Bianchessi, N., and Speranza, M. G. (2015). A branch-price-and-cut algorithm for the commodity constrained split delivery vehicle routing problem. *Computers & Operations Research*, **64**, 1–10.

Archetti, C., Campbell, A. M., and Speranza, M. G. (2016). Multicommodity vs. single-commodity routing. *Transportation Science*, **50**(2), 461–472.

Avella, P., Boccia, M., and Sforza, A. (2004). Solving a fuel delivery problem by heuristic and exact approaches. *European Journal of Operational Research*, **152**(1), 170–179.

Baldacci, R., Mingozzi, A., and Roberti, R. (2011). New route relaxation and pricing strategies for the vehicle routing problem. *Operations Research*, **59**(5), 1269–1283.

Benantar, A., Ouafi, R., and Boukachour, J. (2016). A petrol station replenishment problem: new variant and formulation. *Logistics Research*, **9**(1).

Bode, C. and Irnich, S. (2012). Cut-first branch-and-price-second for the capacitated arc-routing problem. *Operations Research*, **60**(5), 1167–1182.

Brown, G. G. and Graves, G. W. (1981). Real-time dispatch of petroleum tank trucks. *Management Science*, **27**(1), 19–32.

Caramia, M. and Guerriero, F. (2010). A milk collection problem with incompatibility constraints. *Interfaces*, **40**(2), 130–143.

Chajakis, E. D. and Guignard, M. (2003). Scheduling deliveries in vehicles with multiple compartments. *Journal of Global Optimization*, **26**(1), 43–78.

Coelho, L. C. and Laporte, G. (2015). Classification, models and exact algorithms for multi-compartment delivery problems. *European Journal of Operational Research*, **242**(3), 854–864.

Contardo, C., Cordeau, J.-F., and Gendron, B. (2014). An exact algorithm based on cut-and-column generation for the capacitated location-routing problem. *INFORMS Journal on Computing*, **26**(1), 88–102.

Cornillier, F., Boctor, F. F., Laporte, G., and Renaud, J. (2008). An exact algorithm for the petrol station replenishment problem. *Journal of the Operational Research Society*, **59**(5), 607–615.

Derigs, U., Gottlieb, J., Kalkoff, J., Piesche, M., Rothlauf, F., and Vogel, U. (2010). Vehicle routing with compartments: applications, modelling and heuristics. *OR Spectrum*, **33**(4), 885–914.

Desaulniers, G., Desrosiers, J., and Solomon, M. M., editors (2005). *Column Generation*. GERAD 25th anniversary series. Springer Science+Business Media Inc, Boston, MA.

Eshtehadi, R., Demir, E., and Huang, Y. (2020). Solving the vehicle routing problem with multi-compartment vehicles for city logistics. *Computers & Operations Research*, **115**, 104859.

Fagerholt, K. and Christiansen, M. (2000). A combined ship scheduling and allocation problem. *Journal of the Operational Research Society*, **51**(7), 834–842.

Fallahi, A. E., Prins, C., and Calvo, R. W. (2008). A memetic algorithm and a tabu search for the multi-compartment vehicle routing problem. *Computers & Operations Research*, **35**(5), 1725–1741.

Fukasawa, R., Longo, H., Lysgaard, J., de Aragão, M. P., Reis, M., Uchoa, E., and Werneck, R. F. (2005). Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Mathematical Programming*, **106**(3), 491–511.

Goeke, D., Gschwind, T., and Schneider, M. (2019). Upper and lower bounds for the vehicle-routing problem with private fleet and common carrier. *Discrete Applied Mathematics*, **264**, 43–61.

Goodson, J. C. (2015). A priori policy evaluation and cyclic-order-based simulated annealing for the multi-compartment vehicle routing problem with stochastic demands. *European Journal of Operational Research*, **241**(2), 361–369.

Gschwind, T. and Irnich, S. (2016). Dual inequalities for stabilized column generation revisited. *INFORMS Journal on Computing*, **28**(1), 175–194.

Gschwind, T., Bianchessi, N., and Irnich, S. (2019). Stabilized branch-price-and-cut for the commodity-constrained split delivery vehicle routing problem. *European Journal of Operational Research*, **278**(1), 91–104.

Henke, T. (2017). Multi-compartment vehicle routing problems a review and an extended classification. Technical report, Otto-von-Guericke University Magdeburg, Faculty of Economics and Management, Magdeburg, Germany.

Henke, T. (2018). *Multi-compartment vehicle routing problems in the context of glass waste collection*. Ph.D. thesis, Otto-von-Guericke University Magdeburg, Magdeburg, Germany.

Henke, T., Speranza, M. G., and Wäscher, G. (2015). The multi-compartment vehicle routing problem with flexible compartment sizes. *European Journal of Operational Research*, **246**(3), 730–743.

Henke, T., Speranza, M. G., and Wäscher, G. (2018). A branch-and-cut algorithm for the multi-compartment vehicle routing problem with flexible compartment sizes. *Annals of Operations Research*, **275**(2), 321–338.

Hübner, A. and Ostermeier, M. (2019). A multi-compartment vehicle routing problem with loading and unloading costs. *Transportation Science*, **53**(1), 282–300.

Iori, M., Salazar-González, J.-J., and Vigo, D. (2007). An exact approach for the vehicle routing problem with two-dimensional loading constraints. *Transportation Science*, **41**(2), 253–264.

Irnich, S. and Desaulniers, G. (2005). Shortest path problems with resource constraints. In G. Desaulniers, J. Desrosiers, and M. M. Solomon, editors, *Column Generation*, pages 33–65. Springer US, Boston, MA.

Jepsen, M., Petersen, B., Spoorendonk, S., and Pisinger, D. (2008). Subset-row inequalities applied to the vehicle-routing problem with time windows. *Operations Research*, **56**(2), 497–511.

Koch, H., Henke, T., and Wäscher, G. (2016). A Genetic Algorithm for the Multi-Compartment Vehicle Routing Problem with Flexible Compartment Sizes. FEMM Working Papers 160004, Otto-von-Guericke University Magdeburg, Faculty of Economics and Management.

Lahyani, R., Coelho, L. C., Khemakhem, M., Laporte, G., and Semet, F. (2015). A multi-compartment vehicle routing problem arising in the collection of olive oil in tunisia. *Omega*, **51**, 1–10.

Laporte, G., Nobert, Y., and Desrochers, M. (1985). Optimal routing under capacity and distance restrictions. *Operations Research*, **33**(5), 1050–1073.

Melechovskỳ, J. (2013). A variable neighborhood search for the selective multi-compartment vehicle routing problem with time windows. *Lecture notes in management science*, **5**, 159–166.

Mendoza, J. E., Castanier, B., Guéret, C., Medaglia, A. L., and Velasco, N. (2010). A memetic algorithm for the multi-compartment vehicle routing problem with stochastic demands. *Computers & Operations Research*, **37**(11), 1886–1898.

Mendoza, J. E., Castanier, B., Guéret, C., Medaglia, A. L., and Velasco, N. (2011). Constructive heuristics for the multicompartment vehicle routing problem with stochastic demands. *Transportation Science*, **45**(3), 346–363.

Mirzaei, S. and Wøhlk, S. (2017). Erratum to: A branch-and-price algorithm for two multi-compartment vehicle routing problems. *EURO Journal on Transportation and Logistics*, **6**(2), 185–218.

Muyldermans, L. and Pang, G. (2010). On the benefits of co-collection: Experiments with a multi-compartment vehicle routing algorithm. *European Journal of Operational Research*, **206**(1), 93–103.

Oppen, J. and Løkketangen, A. (2008). A tabu search approach for the livestock collection problem. *Computers & Operations Research*, **35**(10), 3213–3229.

Oppen, J., Løkketangen, A., and Desrosiers, J. (2010). Solving a rich vehicle routing and inventory problem using column generation. *Computers & Operations Research*, **37**(7), 1308–1317.

Ostermeier, M., Martins, S., Amorim, P., and Hübner, A. (2018). Loading constraints for a multi-compartment vehicle routing problem. *OR Spectrum*, **40**(4), 997–1027.

Pirkwieser, S., Raidl, G. R., and Gottlieb, J. (2012). Improved packing and routing of vehicles with compartments. In *Computer Aided Systems Theory – EUROCAST 2011*, pages 392–399. Springer Berlin Heidelberg.

Pollaris, H., Braekers, K., Caris, A., Janssens, G. K., and Limbourg, S. (2014). Vehicle routing problems with loading constraints: state-of-the-art and future directions. *OR Spectrum*, **37**(2), 297–330.

Reed, M., Yiannakou, A., and Evering, R. (2014). An ant colony algorithm for the multi-compartment vehicle routing problem. *Applied Soft Computing*, **15**, 169–176.

Tarjan, R. E. (1979). A class of algorithms which require nonlinear time to maintain disjoint sets. *Journal of Computer and System Sciences*, **18**(2), 110–127.

Toth, P. and Vigo, D., editors (2014). *Vehicle Routing*, volume 18 of *MOS-SIAM Series on Optimization*. Society for Industrial and Applied Mathematics, Philadelphia, PA.

Online Appendix

In this Appendix, we present instance-by-instance results. The entries in the Tables 13–15 have the following meaning:

$\lvert V\rvert$:	number of nodes;
ρ:	number of product types;
C:	number of compartments;
s:	supply parameter;
No.:	instance number;
UB:	upper bound; bold if $LB = UB$, i.e., optimality is proven;
LB:	lower bound; bold ditto;
div:	marked if divergent optimal objective values of MCVRP-CFCS and MCVRP-DFCS.

Table 13 displays the results for the `small(H15)` instances, Table 14 for the `mid-size(H18)` instances, and Table 15 for the `large(H15)` instances.

Table 13: Detailed results for the `small(H15)` instances.

Instance					MCVRP-CFCS		MCVRP-DFCS		
$\|V\|$	ρ	C	s	No.	UB	LB	UB	LB	div
10	3	2	1	1	339	339	339	339	
10	3	2	1	2	371	371	371	371	
10	3	2	1	3	355	355	355	355	
10	3	2	1	4	348	348	348	348	
10	3	2	1	5	374	374	376	376	×
10	3	2	2	1	375	375	375	375	
10	3	2	2	2	517	517	517	517	
10	3	2	2	3	477	477	478	478	×
10	3	2	2	4	478	478	478	478	
10	3	2	2	5	496	496	496	496	
10	3	2	3	1	609	609	609	609	
10	3	2	3	2	623	623	623	623	
10	3	2	3	3	614	614	614	614	
10	3	2	3	4	630	630	630	630	
10	3	2	3	5	579	579	579	579	
10	3	3	1	1	342	342	353	353	×
10	3	3	1	2	338	338	350	350	×
10	3	3	1	3	273	273	297	297	×
10	3	3	1	4	355	355	355	355	
10	3	3	1	5	329	329	329	329	
10	3	3	2	1	358	358	358	358	
10	3	3	2	2	408	408	408	408	
10	3	3	2	3	333	333	339	339	×
10	3	3	2	4	338	338	338	338	
10	3	3	2	5	353	353	367	367	×
10	3	3	3	1	413	413	413	413	
10	3	3	3	2	306	306	306	306	
10	3	3	3	3	401	401	403	403	×
10	3	3	3	4	295	295	295	295	
10	3	3	3	5	340	340	340	340	
10	6	2	1	1	485	485	485	485	
10	6	2	1	2	560	560	560	560	
10	6	2	1	3	629	629	629	629	
10	6	2	1	4	509	509	509	509	
10	6	2	1	5	579	579	579	579	
10	6	2	2	1	754	754	754	754	
10	6	2	2	2	813	813	813	813	
10	6	2	2	3	912	912	912	912	
10	6	2	2	4	880	880	880	880	
10	6	2	2	5	611	611	611	611	
10	6	2	3	1	1062	1062	1062	1062	
10	6	2	3	2	912	912	912	912	
10	6	2	3	3	1045	1045	1045	1045	
10	6	2	3	4	1053	1053	1053	1053	
10	6	2	3	5	835	835	835	835	
10	6	4	1	1	320	320	335	335	×
10	6	4	1	2	391	391	391	391	
10	6	4	1	3	285	285	285	285	
10	6	4	1	4	389	389	391	391	×
10	6	4	1	5	370	370	392	392	×
10	6	4	2	1	429	429	429	429	
10	6	4	2	2	532	532	532	532	
10	6	4	2	3	455	455	455	455	
10	6	4	2	4	499	499	501	501	×
10	6	4	2	5	381	381	381	381	
10	6	4	3	1	593	593	593	593	
10	6	4	3	2	697	697	697	697	
10	6	4	3	3	565	565	565	565	
10	6	4	3	4	489	489	489	489	
10	6	4	3	5	543	543	543	543	
10	6	6	1	1	396	396	406	406	×
10	6	6	1	2	318	318	318	318	
10	6	6	1	3	305	305	309	309	×
10	6	6	1	4	305	305	321	321	×
10	6	6	1	5	384	384	415	415	×
10	6	6	2	1	329	329	342	342	×
10	6	6	2	2	297	297	335	335	×
10	6	6	2	3	346	346	362	362	×
10	6	6	2	4	331	331	331	331	
10	6	6	2	5	333	333	358	358	×
10	6	6	3	1	304	304	304	304	
10	6	6	3	2	381	381	381	381	
10	6	6	3	3	338	338	338	338	
10	6	6	3	4	313	313	328	328	×
10	6	6	3	5	304	304	311	311	×
10	9	2	1	1	675	675	675	675	
10	9	2	1	2	567	567	567	567	
10	9	2	1	3	888	888	888	888	
10	9	2	1	4	781	781	781	781	
10	9	2	1	5	822	822	822	822	
10	9	2	2	1	1111	1111	1111	1111	
10	9	2	2	2	940	940	940	940	
10	9	2	2	3	1178	1178	1178	1178	
10	9	2	2	4	1134	1134	1134	1134	
10	9	2	2	5	1003	1003	1003	1003	
10	9	2	3	1	1146	1146	1146	1146	
10	9	2	3	2	1399	1399	1399	1399	
10	9	2	3	3	1418	1418	1418	1418	
10	9	2	3	4	1442	1442	1442	1442	
10	9	2	3	5	1395	1395	1395	1395	
10	9	4	1	1	431	431	431	431	
10	9	4	1	2	272	272	272	272	
10	9	4	1	3	414	414	425	425	×
10	9	4	1	4	395	395	395	395	
10	9	4	1	5	620	620	620	620	
10	9	4	2	1	705	705	705	705	
10	9	4	2	2	542	542	542	542	
10	9	4	2	3	688	688	688	688	
10	9	4	2	4	651	651	651	651	
10	9	4	2	5	586	586	588	588	×
10	9	4	3	1	893	893	893	893	
10	9	4	3	2	643	643	643	643	
10	9	4	3	3	831	831	831	831	
10	9	4	3	4	842	842	842	842	
10	9	4	3	5	856	856	856	856	
10	9	7	1	1	307	307	319	319	×
10	9	7	1	2	377	377	382	382	×
10	9	7	1	3	385	385	400	400	×
10	9	7	1	4	389	389	389	389	
10	9	7	1	5	310	310	346	346	×
10	9	7	2	1	564	564	569	569	×
10	9	7	2	2	390	390	399	399	×
10	9	7	2	3	475	475	486	486	×
10	9	7	2	4	470	470	470	470	
10	9	7	2	5	488	488	496	496	×
10	9	7	3	1	543	543	543	543	
10	9	7	3	2	493	493	493	493	
10	9	7	3	3	593	593	593	593	
10	9	7	3	4	575	575	577	577	×
10	9	7	3	5	490	490	497	497	×
10	9	9	1	1	351	351	359	359	×
10	9	9	1	2	351	351	394	394	×
10	9	9	1	3	388	388	389	389	×
10	9	9	1	4	349	349	413	413	×
10	9	9	1	5	355	355	393	393	×
10	9	9	2	1	371	371	398	398	×
10	9	9	2	2	358	358	435	435	×
10	9	9	2	3	443	443	488	488	×
10	9	9	2	4	389	389	406	406	×
10	9	9	2	5	381	381	474	474	×
10	9	9	3	1	352	352	434	434	×
10	9	9	3	2	325	325	383	383	×
10	9	9	3	3	360	360	443	443	×
10	9	9	3	4	396	396	448	448	×
10	9	9	3	5	384	384	430	430	×

Continued on next column

Table 14: Detailed results for the `mid-size(H18)` instances.

Instance					MCVRP-CFCS		MCVRP-DFCS		
$\lvert V\rvert$	ρ	C	s	No.	UB	LB	UB	LB	div
10	3	2	1	1	357	357	357	357	
10	3	2	1	2	356	356	408	408	×
10	3	2	1	3	392	392	392	392	
10	3	2	1	4	381	381	388	388	×
10	3	2	1	5	332	332	334	334	×
10	3	2	2	1	517	517	517	517	
10	3	2	2	2	625	625	634	634	×
10	3	2	2	3	509	509	532	532	×
10	3	2	2	4	427	427	432	432	×
10	3	2	2	5	444	444	453	453	×
10	3	2	3	1	724	724	724	724	
10	3	2	3	2	631	631	631	631	
10	3	2	3	3	674	674	674	674	
10	3	2	3	4	542	542	542	542	
10	3	2	3	5	547	547	566	566	×
10	3	3	1	1	349	349	351	351	×
10	3	3	1	2	297	297	297	297	
10	3	3	1	3	444	444	444	444	
10	3	3	1	4	331	331	340	340	×
10	3	3	1	5	347	347	448	448	×
10	3	3	2	1	347	347	441	441	×
10	3	3	2	2	409	409	430	430	×
10	3	3	2	3	379	379	387	387	×
10	3	3	2	4	301	301	313	313	×
10	3	3	2	5	456	456	466	466	×
10	3	3	3	1	349	349	354	354	×
10	3	3	3	2	356	356	379	379	×
10	3	3	3	3	384	384	394	394	×
10	3	3	3	4	269	269	269	269	
10	3	3	3	5	318	318	338	338	×
10	4	2	1	1	474	474	474	474	
10	4	2	1	2	412	412	412	412	
10	4	2	1	3	451	451	464	464	×
10	4	2	1	4	411	411	411	411	
10	4	2	1	5	444	444	444	444	
10	4	2	2	1	411	411	411	411	
10	4	2	2	2	640	640	640	640	
10	4	2	2	3	477	477	477	477	
10	4	2	2	4	516	516	518	518	×
10	4	2	2	5	602	602	620	620	×
10	4	2	3	1	728	728	728	728	
10	4	2	3	2	545	545	545	545	
10	4	2	3	3	591	591	591	591	
10	4	2	3	4	640	640	640	640	
10	4	2	3	5	639	639	639	639	
10	4	3	1	1	306	306	306	306	
10	4	3	1	2	370	370	404	404	×
10	4	3	1	3	391	391	391	391	
10	4	3	1	4	418	418	418	418	
10	4	3	1	5	411	411	426	426	×
10	4	3	2	1	452	452	488	488	×
10	4	3	2	2	328	328	328	328	
10	4	3	2	3	381	381	382	382	×
10	4	3	2	4	483	483	488	488	×
10	4	3	2	5	359	359	411	411	×
10	4	3	3	1	438	438	441	441	×
10	4	3	3	2	496	496	496	496	
10	4	3	3	3	534	534	536	536	×
10	4	3	3	4	575	575	575	575	
10	4	3	3	5	618	618	618	618	
10	4	4	1	1	328	328	328	328	
10	4	4	1	2	321	321	349	349	×
10	4	4	1	3	389	389	412	412	×
10	4	4	1	4	392	392	406	406	×
10	4	4	1	5	437	437	437	437	
10	4	4	2	1	351	351	402	402	×
10	4	4	2	2	294	294	375	375	×
10	4	4	2	3	377	377	393	393	×
10	4	4	2	4	265	265	303	303	×
10	4	4	2	5	322	322	393	393	×
10	4	4	3	1	367	367	380	380	×
10	4	4	3	2	380	380	424	424	×
10	4	4	3	3	330	330	342	342	×
10	4	4	3	4	378	378	378	378	
10	4	4	3	5	270	270	292	292	×
15	3	2	1	1	484	484	486	486	×
15	3	2	1	2	360	360	366	366	×
15	3	2	1	3	450	450	450	450	
15	3	2	1	4	543	543	562	562	×
15	3	2	1	5	548	548	548	548	×

Continued on next column/page

Instance					MCVRP-CFCS		MCVRP-DFCS		
$\lvert V\rvert$	ρ	C	s	No.	UB	LB	UB	LB	div
15	3	2	2	1	495	495	496	496	×
15	3	2	2	2	630	630	630	630	
15	3	2	2	3	570	570	574	574	×
15	3	2	2	4	597	597	608	608	×
15	3	2	2	5	607	607	611	611	×
15	3	2	3	1	689	689	706	706	×
15	3	2	3	2	742	742	742	742	
15	3	2	3	3	607	607	607	607	
15	3	2	3	4	668	668	673	673	×
15	3	2	3	5	619	619	619	619	
15	3	3	1	1	467	467	474	474	×
15	3	3	1	2	449	449	455	455	×
15	3	3	1	3	362	362	362	362	
15	3	3	1	4	341	341	341	341	×
15	3	3	1	5	438	438	438	438	
15	3	3	2	1	451	451	483	483	×
15	3	3	2	2	393	393	398	398	×
15	3	3	2	3	372	372	419	419	×
15	3	3	2	4	402	402	408	408	×
15	3	3	2	5	408	408	412	412	×
15	3	3	3	1	356	356	370	370	×
15	3	3	3	2	458	458	458	458	
15	3	3	3	3	450	450	453	453	×
15	3	3	3	4	426	426	429	429	×
15	3	3	3	5	423	423	459	459	×
15	4	2	1	1	526	526	526	526	
15	4	2	1	2	413	413	413	413	
15	4	2	1	3	477	477	477	477	
15	4	2	1	4	587	587	616	616	×
15	4	2	1	5	571	571	571	571	
15	4	2	2	1	614	614	614	614	
15	4	2	2	2	562	562	566	566	×
15	4	2	2	3	661	661	661	661	
15	4	2	2	4	615	615	615	615	
15	4	2	2	5	726	726	726	726	
15	4	2	3	1	806	806	806	806	
15	4	2	3	2	641	641	644	644	×
15	4	2	3	3	695	695	696	696	×
15	4	2	3	4	821	821	821	821	
15	4	2	3	5	650	650	650	650	
15	4	3	1	1	490	490	502	502	×
15	4	3	1	2	516	516	516	516	
15	4	3	1	3	394	394	395	395	×
15	4	3	1	4	387	387	400	400	×
15	4	3	1	5	420	420	420	420	
15	4	3	2	1	510	510	517	517	×
15	4	3	2	2	477	477	490	490	×
15	4	3	2	3	489	489	497	497	×
15	4	3	2	4	433	433	502	502	×
15	4	3	2	5	458	458	462	462	×
15	4	3	3	1	741	741	741	741	
15	4	3	3	2	530	530	545	545	×
15	4	3	3	3	642	642	642	642	
15	4	3	3	4	430	430	430	430	
15	4	3	3	5	647	647	647	647	
15	4	4	1	1	360	360	381	381	×
15	4	4	1	2	435	435	508	508	×
15	4	4	1	3	446	446	481	481	×
15	4	4	1	4	448	448	448	448	
15	4	4	1	5	373	373	375	375	×
15	4	4	2	1	440	440	462	462	×
15	4	4	2	2	432	432	476	476	×
15	4	4	2	3	411	411	411	411	
15	4	4	2	4	409	409	437	437	×
15	4	4	2	5	412	412	437	437	×
15	4	4	3	1	415	415	445	445	×
15	4	4	3	2	432	432	432	432	
15	4	4	3	3	391	391	404	404	×
15	4	4	3	4	485	485	539	539	×
15	4	4	3	5	493	493	506	506	×
20	3	2	1	1	494	494	494	494	
20	3	2	1	2	448	448	448	448	
20	3	2	1	3	576	576	576	576	
20	3	2	1	4	475	475	476	476	×
20	3	2	1	5	523	523	530	530	×
20	3	2	2	1	633	633	634	634	×
20	3	2	2	2	593	593	593	593	
20	3	2	2	3	678	678	682	682	×
20	3	2	2	4	533	533	533	533	×
20	3	2	2	5	586	586	588	588	×
20	3	2	3	1	763	763	763	763	
20	3	2	3	2	641	641	641	641	

Continued on next column/page

Instance					MCVRP-CFCS		MCVRP-DFCS		
$\|V\|$	ρ	C	s	No.	UB	LB	UB	LB	div
20	3	2	3	3	869	869	869	869	
20	3	2	3	4	878	878	878	878	
20	3	2	3	5	790	790	793	793	×
20	3	3	1	1	463	463	463	463	
20	3	3	1	2	464	464	466	466	×
20	3	3	1	3	416	416	425	425	×
20	3	3	1	4	401	401	401	401	
20	3	3	1	5	427	427	446	446	×
20	3	3	2	1	435	435	443	443	×
20	3	3	2	2	424	424	430	430	×
20	3	3	2	3	472	472	493	493	×
20	3	3	2	4	400	400	400	400	
20	3	3	2	5	434	434	466	466	×
20	3	3	3	1	477	477	477	477	
20	3	3	3	2	468	468	479	479	×
20	3	3	3	3	376	376	394	394	×
20	3	3	3	4	466	466	467	467	×
20	3	3	3	5	419	419	419	419	
20	4	2	1	1	565	565	565	565	
20	4	2	1	2	576	576	586	586	×
20	4	2	1	3	485	485	485	485	
20	4	2	1	4	488	488	488	488	
20	4	2	1	5	640	640	640	640	
20	4	2	2	1	717	717	717	717	
20	4	2	2	2	712	712	712	712	
20	4	2	2	3	627	627	628	628	×
20	4	2	2	4	613	613	620	620	×
20	4	2	2	5	689	689	689	689	
20	4	2	3	1	822	822	823	823	×
20	4	2	3	2	688	688	688	688	
20	4	2	3	3	833	833	833	833	
20	4	2	3	4	896	896	896	896	
20	4	2	3	5	812	812	812	812	
20	4	3	1	1	513	513	519	519	×
20	4	3	1	2	403	403	447	447	×
20	4	3	1	3	472	472	472	472	
20	4	3	1	4	481	481	481	481	
20	4	3	1	5	443	443	452	452	×
20	4	3	2	1	521	521	525	525	×
20	4	3	2	2	527	527	528	528	×
20	4	3	2	3	576	576	582	582	×
20	4	3	2	4	608	608	640	640	×
20	4	3	2	5	514	514	514	514	
20	4	3	3	1	614	614	614	614	
20	4	3	3	2	758	758	771	771	×
20	4	3	3	3	745	745	759	759	×
20	4	3	3	4	893	893	893	893	
20	4	3	3	5	675	675	675	675	×
20	4	4	1	1	458	458	490	490	×
20	4	4	1	2	468	468	484	484	×
20	4	4	1	3	453	453	453	453	
20	4	4	1	4	508	508	515	515	×
20	4	4	1	5	429	429	548	548	×
20	4	4	2	1	449	449	449	449	×
20	4	4	2	2	437	437	487	487	×
20	4	4	2	3	523	523	542	542	×
20	4	4	2	4	429	429	438	438	×
20	4	4	2	5	442	442	445	445	×
20	4	4	3	1	445	445	459	459	×
20	4	4	3	2	468	468	486	486	×
20	4	4	3	3	482	482	503	503	×
20	4	4	3	4	443	443	462	462	×
20	4	4	3	5	481	481	486	486	×
25	3	2	1	1	592	592	592	592	
25	3	2	1	2	586	586	586	586	
25	3	2	1	3	488	488	488	488	
25	3	2	1	4	594	594	596	596	×
25	3	2	1	5	512	512	517	517	×
25	3	2	2	1	637	637	638	638	×
25	3	2	2	2	733	733	734	734	×
25	3	2	2	3	732	732	732	732	
25	3	2	2	4	735	735	735	735	
25	3	2	2	5	576	576	586	586	×
25	3	2	3	1	793	793	793	793	
25	3	2	3	2	817	817	817	817	×
25	3	2	3	3	886	886	886	886	
25	3	2	3	4	896	896	924	924	×
25	3	2	3	5	888	888	888	888	
25	3	3	1	1	494	494	495	495	×
25	3	3	1	2	474	474	558	558	×
25	3	3	1	3	475	475	476	476	×
25	3	3	1	4	472	472	486	486	×
25	3	3	1	5	497	497	508	508	×
25	3	3	2	1	502	502	514	514	×
25	3	3	2	2	490	490	508	508	×
25	3	3	2	3	534	534	549	549	×
25	3	3	2	4	443	443	445	445	×
25	3	3	2	5	489	489	502	502	×
25	3	3	3	1	436	436	436	436	
25	3	3	3	2	507	507	521	521	×
25	3	3	3	3	474	474	475	475	×
25	3	3	3	4	455	455	471	471	×
25	3	3	3	5	516	516	531	531	×
25	4	2	1	1	616	616	616	616	
25	4	2	1	2	724	724	724	724	
25	4	2	1	3	613	613	613	613	
25	4	2	1	4	556	556	556	556	
25	4	2	1	5	608	608	614	614	×
25	4	2	2	1	773	773	773	773	
25	4	2	2	2	744	744	746	746	×
25	4	2	2	3	835	835	844	844	×
25	4	2	2	4	937	937	942	942	×
25	4	2	2	5	881	881	884	884	×
25	4	2	3	1	919	919	927	927	×
25	4	2	3	2	885	885	885	885	
25	4	2	3	3	861	861	861	861	
25	4	2	3	4	798	798	798	798	
25	4	2	3	5	896	896	896	896	
25	4	3	1	1	544	544	545	545	×
25	4	3	1	2	492	492	523	523	×
25	4	3	1	3	545	545	545	545	
25	4	3	1	4	518	518	521	521	×
25	4	3	1	5	543	543	543	543	
25	4	3	2	1	558	558	568	568	×
25	4	3	2	2	677	677	687	687	×
25	4	3	2	3	603	603	625	625	×
25	4	3	2	4	623	623	623	623	
25	4	3	2	5	465	465	504	504	×
25	4	3	3	1	812	812	825	825	×
25	4	3	3	2	827	827	827	827	×
25	4	3	3	3	794	794	800	800	×
25	4	3	3	4	774	774	774	774	
25	4	3	3	5	789	789	789	789	
25	4	4	1	1	483	483	496	496	×
25	4	4	1	2	460	460	472	472	×
25	4	4	1	3	510	510	534	534	×
25	4	4	1	4	434	434	499	499	×
25	4	4	1	5	496	496	511	511	×
25	4	4	2	1	513	513	520	520	×
25	4	4	2	2	477	477	483	483	×
25	4	4	2	3	521	521	521	521	
25	4	4	2	4	432	432	436	436	×
25	4	4	2	5	533	533	540	540	×
25	4	4	3	1	512	512	543	543	×
25	4	4	3	2	492	492	502	502	×
25	4	4	3	3	554	554	561	561	×
25	4	4	3	4	521	521	528	528	×
25	4	4	3	5	446	446	458	458	×
30	3	2	1	1	553	553	564	564	×
30	3	2	1	2	560	560	560	560	
30	3	2	1	3	589	589	605	605	×
30	3	2	1	4	534	534	534	534	
30	3	2	1	5	616	616	620	620	×
30	3	2	2	1	717	717	717	717	×
30	3	2	2	2	774	774	778	778	×
30	3	2	2	3	750	750	750	750	
30	3	2	2	4	755	755	758	758	×
30	3	2	2	5	758	758	758	758	
30	3	2	3	1	939	939	939	939	
30	3	2	3	2	935	935	935	935	
30	3	2	3	3	897	897	898	898	×
30	3	2	3	4	909	909	910	910	×
30	3	2	3	5	955	955	955	955	
30	3	3	1	1	538	538	549	549	×
30	3	3	1	2	500	500	522	522	×
30	3	3	1	3	528	528	528	528	
30	3	3	1	4	532	532	540	540	×
30	3	3	1	5	440	440	457	457	×
30	3	3	2	1	482	482	482	482	×
30	3	3	2	2	597	597	608	608	×
30	3	3	2	3	541	541	571	571	×
30	3	3	2	4	473	473	487	487	×
30	3	3	2	5	513	513	523	523	×
30	3	3	3	1	445	445	445	445	
30	3	3	3	2	554	554	562	562	×
30	3	3	3	3	494	494	519	519	×
30	3	3	3	4	484	484	499	499	×
30	3	3	3	5	565	565	565	565	
30	4	2	1	1	535	535	555	555	×
30	4	2	1	2	557	557	558	558	×
30	4	2	1	3	703	703	703	703	
30	4	2	1	4	640	640	640	640	

Continued on next column/page

Continued on next column/page

Instance					MCVRP-CFCS		MCVRP-DFCS		
$\|V\|$	ρ	C	s	No.	UB	LB	UB	LB	div
30	4	2	1	5	693	693	693	693	
30	4	2	2	1	890	890	893	893	×
30	4	2	2	2	905	905	905	905	×
30	4	2	2	3	847	847	853	853	×
30	4	2	2	4	870	870	870	870	
30	4	2	2	5	799	799	799	799	×
30	4	2	3	1	967	967	967	967	
30	4	2	3	2	899	899	899	899	
30	4	2	3	3	1032	1032	1032	1032	
30	4	2	3	4	1017	1017	1017	1017	
30	4	2	3	5	954	954	954	954	
30	4	3	1	1	585	585	593	593	×
30	4	3	1	2	512	512	522	522	×
30	4	3	1	3	564	564	590	590	×
30	4	3	1	4	573	573	599	599	×
30	4	3	1	5	549	549	552	552	×
30	4	3	2	1	756	756	758	758	×
30	4	3	2	2	699	699	724	724	×
30	4	3	2	3	554	554	573	573	×
30	4	3	2	4	634	634	651	651	×
30	4	3	2	5	663	663	670	670	×
30	4	3	3	1	768	768	768	768	
30	4	3	3	2	830	830	830	830	
30	4	3	3	3	761	761	761	761	
30	4	3	3	4	878	878	878	878	
30	4	3	3	5	841	841	841	841	
30	4	4	1	1	501	501	517	517	×
30	4	4	1	2	502	502	539	539	×
30	4	4	1	3	543	543	567	567	×
30	4	4	1	4	524	524	539	539	×
30	4	4	1	5	534	534	552	552	×
30	4	4	2	1	509	509	531	531	×
30	4	4	2	2	490	490	500	500	×
30	4	4	2	3	489	489	492	492	×
30	4	4	2	4	540	540	540	540	
30	4	4	2	5	557	557	567	567	×
30	4	4	3	1	549	549	577	577	×
30	4	4	3	2	482	482	509	509	×
30	4	4	3	3	529	529	539	539	×
30	4	4	3	4	451	451	474	474	×
30	4	4	3	5	497	497	501	501	×
35	3	2	1	1	622	622	628	628	×
35	3	2	1	2	640	640	647	647	×
35	3	2	1	3	671	671	674	674	×
35	3	2	1	4	560	560	560	560	
35	3	2	1	5	552	552	555	555	×
35	3	2	2	1	742	742	742	742	
35	3	2	2	2	783	783	784	784	×
35	3	2	2	3	831	831	836	836	×
35	3	2	2	4	833	833	854	854	×
35	3	2	2	5	793	793	794	794	×
35	3	2	3	1	1073	1073	1073	1073	×
35	3	2	3	2	963	963	963	963	
35	3	2	3	3	1030	1030	1035	1035	×
35	3	2	3	4	1055	1055	1055	1055	
35	3	2	3	5	938	938	938	938	
35	3	3	1	1	555	555	565	565	×
35	3	3	1	2	540	540	548	548	×
35	3	3	1	3	557	557	564	564	×
35	3	3	1	4	528	528	557	557	×
35	3	3	1	5	528	528	550	550	×
35	3	3	2	1	533	533	543	543	×
35	3	3	2	2	545	545	548	548	×
35	3	3	2	3	572	572	572	572	
35	3	3	2	4	494	494	506	506	×
35	3	3	2	5	600	600	602	602	×
35	3	3	3	1	532	532	574	574	×
35	3	3	3	2	502	502	502	502	
35	3	3	3	3	520	520	521	521	×
35	3	3	3	4	542	542	555	555	×
35	3	3	3	5	530	530	533	533	×
35	4	2	1	1	665	665	669	669	×
35	4	2	1	2	686	686	686	686	
35	4	2	1	3	755	755	755	755	
35	4	2	1	4	688	688	688	688	
35	4	2	1	5	687	687	691	691	×
35	4	2	2	1	1058	1058	1058	1058	
35	4	2	2	2	856	856	856	856	
35	4	2	2	3	907	907	910	910	×
35	4	2	2	4	987	987	996	996	×
35	4	2	2	5	1054	1037	1056	1037	
35	4	2	3	1	1049	1049	1049	1049	×
35	4	2	3	2	954	954	954	954	
35	4	2	3	3	1049	1049	1057	1057	×
35	4	2	3	4	1089	1089	1089	1089	
35	4	2	3	5	991	991	1000	1000	×
35	4	3	1	1	583	583	601	601	×
35	4	3	1	2	624	624	636	636	×
35	4	3	1	3	549	549	571	571	×
35	4	3	1	4	556	556	560	560	×
35	4	3	1	5	595	595	595	595	
35	4	3	2	1	733	733	734	734	×
35	4	3	2	2	694	694	731	731	×
35	4	3	2	3	757	757	760	760	×
35	4	3	2	4	732	732	737	737	×
35	4	3	2	5	703	703	720	720	×
35	4	3	3	1	915	915	918	915	
35	4	3	3	2	930	930	930	930	
35	4	3	3	3	925	925	925	925	
35	4	3	3	4	904	904	939	939	×
35	4	3	3	5	861	861	861	861	
35	4	4	1	1	533	533	549	549	×
35	4	4	1	2	515	515	523	523	×
35	4	4	1	3	499	499	503	503	×
35	4	4	1	4	526	526	564	564	×
35	4	4	1	5	475	475	505	505	×
35	4	4	2	1	605	605	607	607	×
35	4	4	2	2	577	577	590	590	×
35	4	4	2	3	535	535	557	557	×
35	4	4	2	4	538	538	550	550	×
35	4	4	2	5	578	578	584	584	×
35	4	4	3	1	536	536	574	574	×
35	4	4	3	2	516	516	518	518	×
35	4	4	3	3	566	566	570	570	×
35	4	4	3	4	470	470	471	471	×
35	4	4	3	5	466	466	500	500	×
40	3	2	1	1	726	726	729	729	×
40	3	2	1	2	662	662	670	670	×
40	3	2	1	3	598	598	601	601	×
40	3	2	1	4	630	630	632	632	×
40	3	2	1	5	598	598	601	601	×
40	3	2	2	1	853	853	856	853	
40	3	2	2	2	842	842	842	842	
40	3	2	2	3	829	829	832	832	×
40	3	2	2	4	836	836	837	837	×
40	3	2	2	5	819	819	833	833	×
40	3	2	3	1	1054	1054	1054	1054	
40	3	2	3	2	996	996	996	996	
40	3	2	3	3	1049	1049	1049	1049	
40	3	2	3	4	1078	1078	1078	1078	
40	3	2	3	5	1074	1074	1074	1074	
40	3	3	1	1	540	540	544	544	×
40	3	3	1	2	606	606	621	621	×
40	3	3	1	3	615	615	615	615	
40	3	3	1	4	555	555	555	555	
40	3	3	1	5	504	504	516	516	×
40	3	3	2	1	620	620	625	625	×
40	3	3	2	2	629	629	633	633	×
40	3	3	2	3	594	594	602	602	×
40	3	3	2	4	590	590	591	591	×
40	3	3	2	5	585	585	586	586	×
40	3	3	3	1	558	558	560	560	×
40	3	3	3	2	604	604	608	608	×
40	3	3	3	3	599	599	607	607	×
40	3	3	3	4	624	624	624	624	
40	3	3	3	5	565	565	565	565	
40	4	2	1	1	798	746	799	746	
40	4	2	1	2	729	729	732	732	×
40	4	2	1	3	813	813	813	813	
40	4	2	1	4	757	757	774	774	×
40	4	2	1	5	729	729	732	732	×
40	4	2	2	1	944	944	944	944	
40	4	2	2	2	931	931	938	938	×
40	4	2	2	3	1039	1039	1043	1043	×
40	4	2	2	4	990	980	1015	980	
40	4	2	2	5	965	965	965	965	
40	4	2	3	1	1095	1095	1095	1095	
40	4	2	3	2	1090	1090	1111	1111	×
40	4	2	3	3	1079	1079	1086	1086	×
40	4	2	3	4	1053	1053	1053	1053	
40	4	2	3	5	1169	1169	1169	1169	
40	4	3	1	1	576	576	576	576	
40	4	3	1	2	688	688	688	688	
40	4	3	1	3	559	559	606	606	×
40	4	3	1	4	598	598	598	598	×
40	4	3	1	5	609	609	617	617	×
40	4	3	2	1	695	695	695	695	
40	4	3	2	2	685	685	711	711	×
40	4	3	2	3	772	772	774	774	×
40	4	3	2	4	770	757	774	774	×
40	4	3	2	5	825	798	848	810	
40	4	3	3	1	952	952	970	952	

Continued on next column/page

Instance					MCVRP-CFCS		MCVRP-DFCS		
$\|V\|$	ρ	C	s	No.	UB	LB	UB	LB	div
40	4	3	3	2	927	912	930	912	
40	4	3	3	3	**946**	**946**	**946**	**946**	
40	4	3	3	4	874	853	874	853	
40	4	3	3	5	969	950	992	958	
40	4	4	1	1	**568**	**568**	**584**	**584**	×
40	4	4	1	2	**626**	**626**	**641**	**641**	×
40	4	4	1	3	**526**	**526**	**551**	**551**	×
40	4	4	1	4	**603**	**603**	**607**	**607**	×
40	4	4	1	5	**597**	**597**	**622**	**622**	×
40	4	4	2	1	**561**	**561**	**633**	**633**	×
40	4	4	2	2	**542**	**542**	**580**	**580**	×
40	4	4	2	3	**569**	**569**	**577**	**577**	×
40	4	4	2	4	**576**	**576**	**585**	**585**	×
40	4	4	2	5	**604**	**604**	**610**	**610**	×
40	4	4	3	1	**542**	**542**	**558**	**558**	×
40	4	4	3	2	**594**	**594**	**618**	**618**	×
40	4	4	3	3	**547**	**547**	**547**	**547**	
40	4	4	3	4	**581**	**581**	**595**	**595**	×
40	4	4	3	5	**554**	**554**	**561**	**561**	×
45	3	2	1	1	**749**	**749**	**749**	**749**	
45	3	2	1	2	**725**	**725**	**725**	**725**	
45	3	2	1	3	**718**	**718**	**722**	**722**	×
45	3	2	1	4	**632**	**632**	**634**	**634**	×
45	3	2	1	5	**722**	**722**	**723**	**723**	×
45	3	2	2	1	936	919	958	922	
45	3	2	2	2	907	889	917	889	
45	3	2	2	3	**803**	**803**	**806**	**806**	×
45	3	2	2	4	**860**	**860**	**861**	**861**	×
45	3	2	2	5	**836**	**836**	**846**	**846**	×
45	3	2	3	1	**1113**	**1113**	**1113**	**1113**	
45	3	2	3	2	**1063**	**1063**	**1063**	**1063**	
45	3	2	3	3	**1130**	**1130**	**1130**	**1130**	
45	3	2	3	4	**982**	**982**	**982**	**982**	
45	3	2	3	5	1240	1235	1240	1238	
45	3	3	1	1	**577**	**577**	**581**	**581**	×
45	3	3	1	2	**574**	**574**	**606**	**606**	×
45	3	3	1	3	**578**	**578**	**578**	**578**	
45	3	3	1	4	**552**	**552**	**556**	**556**	×
45	3	3	1	5	**571**	**571**	**578**	**578**	×
45	3	3	2	1	**599**	**599**	**604**	**604**	×
45	3	3	2	2	**609**	**609**	**609**	**609**	
45	3	3	2	3	**577**	**577**	**577**	**577**	
45	3	3	2	4	**639**	**639**	**639**	**639**	
45	3	3	2	5	**575**	**575**	**581**	**581**	×
45	3	3	3	1	**603**	**603**	**614**	**614**	×
45	3	3	3	2	**559**	**559**	**561**	**561**	×
45	3	3	3	3	**606**	**606**	**615**	**615**	×
45	3	3	3	4	**605**	**605**	**618**	**618**	×
45	3	3	3	5	**586**	**586**	**604**	**604**	×
45	4	2	1	1	**809**	**809**	819	809	
45	4	2	1	2	**759**	**759**	**766**	**766**	×
45	4	2	1	3	**637**	**637**	**637**	**637**	
45	4	2	1	4	835	781	835	790	
45	4	2	1	5	**732**	**732**	**732**	**732**	
45	4	2	2	1	1075	1054	**1075**	**1075**	
45	4	2	2	2	989	980	989	980	
45	4	2	2	3	993	961	996	975	
45	4	2	2	4	1030	991	**1030**	**1030**	
45	4	2	2	5	**1040**	**1040**	**1041**	**1041**	×
45	4	2	3	1	**1105**	**1105**	**1113**	**1113**	×
45	4	2	3	2	**1169**	**1169**	**1169**	**1169**	
45	4	2	3	3	**1089**	**1089**	**1089**	**1089**	
45	4	2	3	4	**1199**	**1199**	**1214**	**1214**	×
45	4	2	3	5	**1108**	**1108**	**1108**	**1108**	
45	4	3	1	1	**609**	**609**	**616**	**616**	×
45	4	3	1	2	**676**	**676**	**676**	**676**	
45	4	3	1	3	**596**	**596**	**607**	**607**	×
45	4	3	1	4	**635**	**635**	**664**	**664**	×
45	4	3	1	5	**614**	**614**	**624**	**624**	×
45	4	3	2	1	**777**	**777**	**800**	**800**	×
45	4	3	2	2	**777**	**777**	**779**	**779**	×
45	4	3	2	3	**777**	**777**	800	786	×
45	4	3	2	4	**805**	**805**	**805**	**805**	
45	4	3	2	5	**802**	**802**	809	802	
45	4	3	3	1	1034	1011	1056	1028	
45	4	3	3	2	**918**	**918**	**931**	**931**	×
45	4	3	3	3	1071	1011	1100	1011	
45	4	3	3	4	1031	978	1031	981	
45	4	3	3	5	1021	957	1033	957	
45	4	4	1	1	**567**	**567**	**587**	**587**	×
45	4	4	1	2	**599**	**599**	**628**	**628**	×
45	4	4	1	3	**623**	**623**	**629**	**629**	×
45	4	4	1	4	**594**	**594**	**594**	**594**	
45	4	4	1	5	**621**	**621**	**621**	**621**	
45	4	4	2	1	**607**	**607**	**652**	**652**	×
45	4	4	2	2	**585**	**585**	**592**	**592**	×
45	4	4	2	3	**605**	**605**	**626**	**626**	×
45	4	4	2	4	**671**	**671**	**676**	**676**	×
45	4	4	2	5	**598**	**598**	**606**	**606**	×
45	4	4	3	1	**628**	**628**	**640**	**640**	×
45	4	4	3	2	**590**	**590**	**626**	**626**	×
45	4	4	3	3	**610**	**610**	**627**	**627**	×
45	4	4	3	4	**633**	**633**	**637**	**637**	×
45	4	4	3	5	**574**	**574**	**603**	**603**	×
50	3	2	1	1	**687**	**687**	**693**	**693**	×
50	3	2	1	2	**685**	**685**	**685**	**685**	
50	3	2	1	3	**793**	**793**	**793**	**793**	
50	3	2	1	4	**682**	**682**	**694**	**694**	×
50	3	2	1	5	**770**	**770**	**770**	**770**	
50	3	2	2	1	940	915	983	915	
50	3	2	2	2	**900**	**900**	908	900	
50	3	2	2	3	**899**	**899**	**899**	**899**	×
50	3	2	2	4	**929**	**929**	**931**	**931**	×
50	3	2	2	5	**872**	**872**	**873**	**873**	×
50	3	2	3	1	**1258**	**1258**	**1258**	**1258**	
50	3	2	3	2	1216	1198	1250	1203	
50	3	2	3	3	**1078**	**1078**	**1078**	**1078**	
50	3	2	3	4	1199	1164	1199	1173	
50	3	2	3	5	**1094**	**1094**	**1094**	**1094**	
50	3	3	1	1	**613**	**613**	**643**	**643**	×
50	3	3	1	2	**601**	**601**	**605**	**605**	×
50	3	3	1	3	**609**	**609**	**621**	**621**	×
50	3	3	1	4	**627**	**627**	**627**	**627**	
50	3	3	1	5	**664**	**664**	**667**	**667**	×
50	3	3	2	1	**619**	**619**	**633**	**633**	×
50	3	3	2	2	**626**	**626**	**641**	**641**	×
50	3	3	2	3	**632**	**632**	**632**	**632**	
50	3	3	2	4	**699**	**699**	**707**	**707**	×
50	3	3	2	5	**641**	**641**	**641**	**641**	
50	3	3	3	1	**650**	**650**	**650**	**650**	
50	3	3	3	2	**665**	**665**	**665**	**665**	
50	3	3	3	3	**680**	**680**	**680**	**680**	
50	3	3	3	4	**608**	**608**	**608**	**608**	
50	3	3	3	5	**651**	**651**	**654**	**654**	×
50	4	2	1	1	830	821	**830**	**830**	
50	4	2	1	2	**862**	**862**	**862**	**862**	
50	4	2	1	3	791	728	828	734	
50	4	2	1	4	**727**	**727**	**727**	**727**	
50	4	2	1	5	**758**	**758**	**758**	**758**	
50	4	2	2	1	975	956	979	956	
50	4	2	2	2	**1079**	**1079**	1088	1079	
50	4	2	2	3	**1034**	**1034**	**1034**	**1034**	
50	4	2	2	4	1076	1065	1096	1065	
50	4	2	2	5	1062	1059	1062	1059	
50	4	2	3	1	**1290**	**1290**	**1293**	**1293**	×
50	4	2	3	2	**1182**	**1182**	**1182**	**1182**	
50	4	2	3	3	1070	1062	**1072**	**1072**	×
50	4	2	3	4	**1145**	**1145**	**1145**	**1145**	
50	4	2	3	5	**1154**	**1154**	**1156**	**1156**	×
50	4	3	1	1	**652**	**652**	**665**	**665**	×
50	4	3	1	2	**617**	**617**	**617**	**617**	
50	4	3	1	3	**725**	**725**	**728**	**728**	×
50	4	3	1	4	**614**	**614**	**618**	**618**	×
50	4	3	1	5	**695**	**695**	**700**	**700**	×
50	4	3	2	1	**804**	**804**	805	804	
50	4	3	2	2	**830**	**830**	884	834	×
50	4	3	2	3	**840**	**840**	**841**	**841**	×
50	4	3	2	4	**763**	**763**	**770**	**770**	×
50	4	3	2	5	**768**	**768**	**779**	**779**	×
50	4	3	3	1	**1030**	**1030**	**1030**	**1030**	
50	4	3	3	2	**1098**	**1098**	**1098**	**1098**	
50	4	3	3	3	1069	1012	1147	1037	
50	4	3	3	4	1018	968	1062	991	
50	4	3	3	5	1077	996	1077	999	
50	4	4	1	1	**668**	**668**	**673**	**673**	×
50	4	4	1	2	**602**	**602**	**613**	**613**	×
50	4	4	1	3	**659**	**659**	**659**	**659**	×
50	4	4	1	4	**570**	**570**	**581**	**581**	×
50	4	4	1	5	**625**	**625**	**626**	**626**	×
50	4	4	2	1	**625**	**625**	**627**	**627**	×
50	4	4	2	2	**585**	**585**	**605**	**605**	×
50	4	4	2	3	**593**	**593**	**621**	**621**	×
50	4	4	2	4	**584**	**584**	**599**	**599**	×
50	4	4	2	5	**609**	**609**	**642**	**642**	×
50	4	4	3	1	**615**	**615**	**629**	**629**	×
50	4	4	3	2	**615**	**615**	**630**	**630**	×
50	4	4	3	3	**607**	**607**	**617**	**617**	×
50	4	4	3	4	**604**	**604**	**619**	**619**	×
50	4	4	3	5	**592**	**592**	**615**	**615**	×

Continued on next column/page

Table 15: Detailed results for the `large(H15)` instances.

Instance				MCVRP-CFCS		MCVRP-DFCS		
$\lvert V \rvert$	ρ	C	s	UB	LB	UB	LB	div
50	3	2	1	**1000**	**1000**	**1022**	**1022**	×
50	3	2	2	1486	1036	2537	1036	
50	3	2	3		1340		1343	
50	3	3	1	**1028**	**1028**	2227	1036	×
50	3	3	2	1017	1013		1020	×
50	3	3	3	954	917		944	
50	6	2	1	1324	1310	**1324**	**1324**	
50	6	2	2		1485		1534	
50	6	2	3		1820		1821	
50	6	4	1	**917**	**917**	2369	945	×
50	6	4	2		1116		1176	
50	6	4	3		1242		1260	
50	6	6	1	**972**	**972**		1012	×
50	6	6	2	1057	910		981	
50	6	6	3		857		897	
50	9	2	1	2493	1513	2777	1513	
50	9	2	2		1889		1889	
50	9	2	3		2440		2440	
50	9	4	1	3353	1108		1139	
50	9	4	2		1041		1041	
50	9	4	3		1397		1402	
50	9	7	1	**951**	**951**		1008	×
50	9	7	2		980		988	
50	9	7	3		1126		1148	
50	9	9	1	**963**	**963**		1013	×
50	9	9	2		966		1068	
50	9	9	3		914		1014	