OR Spectrum (2018) 40:997–1027
https://doi.org/10.1007/s00291-018-0524-4

REGULAR ARTICLE

Loading constraints for a multi-compartment vehicle routing problem

Manuel Ostermeier[2] · Sara Martins[1] · Pedro Amorim[1] · Alexander Hübner[2]

Received: 25 January 2018 / Accepted: 30 May 2018 / Published online: 29 June 2018

Abstract

Multi-compartment vehicles (MCVs) can deliver several product segments jointly. Separate compartments are necessary as each product segment has its own specific characteristics and segments cannot be mixed during transportation. The size and position of the compartments can be adjusted for each tour with the use of flexible compartments. However, this requires that the compartments can be accessed for loading/unloading. The layout of the compartments is defined by the customer and segment sequence, and it needs to be organized in a way that no blocking occurs during loading/unloading processes. Routing and loading layouts are interdependent for MCVs. This paper addresses such loading/unloading issues raised in the distribution planning when using MCVs with flexible compartments, loading from the rear, and standardized transportation units. The problem can therefore be described as a two-dimensional loading and multi-compartment vehicle routing problem (2L-MCVRP). We address the problem of obtaining feasible MCV loading with minimal routing, loading and unloading costs. We define the loading problem that configures the compartment setup. Consequently, we develop a branch-and-cut (B&C) algorithm as an exact approach and extend a large neighborhood search (LNS) as a heuristic approach. In both cases, we use the loading model in order to verify the feasibility of the tours and to assess the problem as a routing and loading problem. The loading model dictates the cuts to be performed in the B&C, and it is used as a repair mechanism in the LNS. Numerical studies show that the heuristic reaches the optimal solution for small instances and can be applied efficiently to larger problems. Additionally, further tests on large instances enable us to derive general rules regarding the influence of loading constraints. Our results were validated in a case study with a European retailer. We identified that loading constraints matter even for small instances. Feasible loading can often be achieved only through minor changes to the routing solution and therefore with limited additional costs. Further, the importance to integrate loading constraints grows as the problem size increases, especially when a heterogeneous mix of segments is ordered.

Keywords Loading constraints · Multi-temperature logistics · Vehicle routing · Large neighborhood search · Branch-and-cut

Extended author information available on the last page of the article

1 Introduction

This paper addresses loading/unloading issues raised in distribution planning when using multi-compartment vehicles (MCVs) with flexible compartments. The main applications of MCVs are waste collection (e.g., Henke et al. 2015), fuel distribution (e.g., Derigs et al. 2011) and food distribution (e.g., Hübner and Ostermeier 2018). The focus of this paper is on the retail grocery distribution, where efficient logistics are essential due to narrow margins, the requirements for higher product availability and the extensive regulations (e.g., product traceability).

Retailers need to differentiate temperature requirements when transporting grocery products (e.g., frozen, fresh, ambient). In former times, each product segment was delivered separately, i.e., each group of products with a particular temperature requirement was distributed on single-compartment vehicles (SCVs). However, over the last decade an increasing number of retailers have been using the relatively new technology of MCVs (Klingler et al. 2016). MCVs are technically able to split flexibly their loading area into separate compartments, where each one of them is adjusted to a particular temperature, meeting the specific product requirements. The number and size of compartments of each MCV are not predefined. The number of used compartments can vary between one and five. An MCV can be adjusted for each tour separately without any loss of capacity. Additionally, the position of the compartments can be freely chosen on the vehicle according to the orders assigned. These features of an MCV make it possible to perform joint deliveries of product segments on the same vehicle.

To benefit from using MCVs, retailers need to overcome additional challenges. Stores usually order products from different segments at the same time, but retailers organize the distribution centers (DC) by temperature zone. Hence, a truck needs to pick up the orders separately for each segment at the DC gate for the specific temperature in order to consolidate customer orders for different segments. The trucks are loaded with standardized and identically sized transportation units. The loading process is different from waste or glass waste collection, where the products are loaded from the top. In grocery distribution, the loading of MCVs is carried out from the rear of the vehicle. Figure 1 illustrates a possible loading layout of an MCV tour with four different product segments. Two horizontal compartment walls (from rear door to front) and three vertical walls (between segments 1 and 2, 2 and 3, and 2 and 4) divide the loading area. When a delivery with orders from multiple compartments is performed, certain compartments have to be accessed at the same customer stop in order to unload all his orders. However, the access to a compartment may be blocked by other compartments due to the rear loading. For example, compartment 1 and its orders can only be accessed if the corresponding parts of compartments 2, 3 and 4 have been at least partially emptied previously. Therefore, if the loading/unloading operations are ignored during route planning, different orders can only be unloaded after the other loaded orders have been rearranged.

The problem described can be classified as a two-dimensional loading multi-compartment vehicle routing problem (2L-MCVRP). More specifically, it defines the route and compartment sizes, assigns orders to compartments, and determines how compartments and orders should be placed in the truck in order to obtain feasible tours

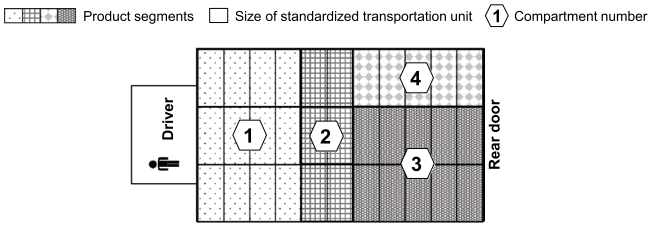

Fig. 1 Loading layout of one MCV with four product segments (illustrative example)

at the lowest possible cost. Since most of the literature on MCVs focuses on waste and petrol applications, where each compartment can be accessed individually (e.g., from the top or by separate filler), this is—to the best of our knowledge—the first paper that studies the loading problem of MCVs. The contributions of this paper are along three dimensions. Firstly, we identify the MCVRP with loading constraints considering rear loading and standardized transportation units. Secondly, we develop two solution approaches for solving the problem based on an exact approach (branch-and-cut (B&C)) and a heuristic approach (an extension of a large neighborhood search (LNS) framework). Both approaches use the development of a multi-compartment packing problem (MCPP) that identifies tours with infeasible loading situations, ensuring the route plan of feasible tours. Finally, we identify the impact of loading constraints in MCVRPs by means of numerical studies based on retail data and generalize the findings with randomly generated data.

The remainder of this paper is organized as follows: In Sect. 2, we elaborate the problem context and discuss the related literature. The mathematical formulation of the 2L-MCVRP is given in Sect. 3, followed by the proposed solution approaches presented in Sect. 4. In Sect. 5, the influence of loading constraints is evaluated by means of numerical experiments. Finally, the findings are summarized in Sect. 6.

2 Loading problem of MCVs

The distribution process using MCVs needs to be distinguished from the distribution that is made with standard vehicles, as different loading/unloading processes need to be considered. In this section, we first detail the distribution processes with MCVs and its associated costs before defining the actual loading issues. Further, we discuss the related literature to conclude this section.

2.1 Distribution process with MCVs and associated costs

Product segments are stored in the DC by temperature zone. For that reason, an MCV needs to pick up pallets from multiple segments (see left-hand side of Fig. 2, where four shipping gates are approached) at different shipping gates. As a result, the loading costs depend on the number of product segments and hence on the number of compartments used on the vehicle. On the one hand, the more the segments there are, the more expensive the loading costs will be as there are setup costs related to traveling between

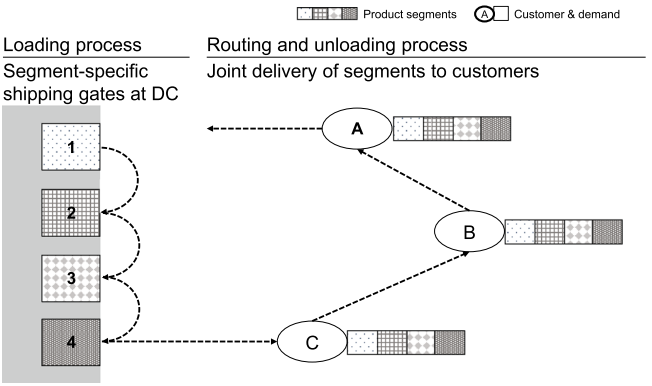

Fig. 2 Loading, routing and unloading processes with an MCV

gates to load different segments. On the other, customer orders from different segments can be combined, which implies the reduction in the number of stops at stores (see the right-hand side of Fig. 2, where each customer receives more than one segment) and the related unloading costs (=stop costs) (see Hübner and Ostermeier 2017).

Sequencing orders in the loading area of an SCV is trivial for our problem application. As we are focusing on grocery distribution and the transportation units are of identical size and cannot be stacked, the loading of orders onto an SCV needs to follow the opposite sequence of the route so that the orders of the first customer are at the rear of the truck. MCVs need to collect customer orders from each segment with different compartments respecting the tour sequence. Orders are not intended to be reorganized during the tour (e.g., moving orders from store B during a stop at store A) due to strict legal regulations. These regulations state that products have to be stored under appropriate temperature during transportation. Moreover, short time windows are given for the unloading process, and retail outlets have limited space and significantly higher unloading costs if orders are reorganized at customer stops. In the following, we describe the requirements of loading/unloading procedures to obtain a feasible MCV loading.

(1) Requirements for collecting orders from one segment All orders of one segment that will be transported on the same vehicle have to be loaded at the same time (i.e., no other segment can be loaded in between), as otherwise it would be necessary to approach a shipping gate for the same segment several times. This is impractical as it would lead to an unmanageable planning situation, longer waiting times at the DC and a significant increase in setup costs for loading. Figure 3 illustrates four different loading options for an MCV with four compartments (vehicle (V) 1–4). The loading area is divided into three aisles, and a compartment can be within and across an aisle. The example V1 shows a feasible solution where two segments share the middle aisle, but are separated by a compartment wall. Two compartments for the same segment can be set up one after another in different aisles of an MCV, as shown in example V2. This is viable as the compartments can be loaded one after the other at the same shipping gate. In contrast, example V3 shows the possibility of splitting one compartment for

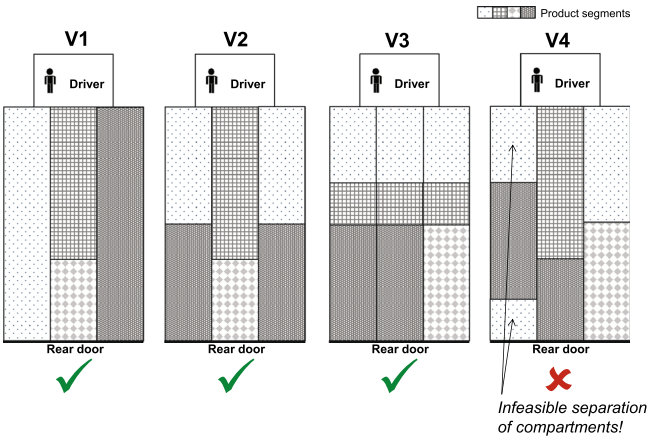

Fig. 3 Vehicle loading layout with multiple compartments (illustrative example)

the same segment crosswise, i.e., into different aisles, whereas example V4 illustrates that one compartment cannot be split in the same aisle. A loading layout as given in V4 would require the repeated approach of a shipping gate. When a vehicle leaves the DC the layout of the loading area is fixed and cannot be changed.

Requirements for a feasible (2) customer and (3) segment sequence From the unloading problem of an MCV arises the question of whether all orders that need to be unloaded at a certain stop can be accessed without any obstacles. Figure 4, in which tour data are given on the left-hand side, illustrates possible sequencing issues for a tour example. This example considers three customers (A, B and C, represented by circles), and each customer orders from a specific segment (represented by rectangles) with different order sizes (represented by the size of the rectangle). Let us assume that the customer sequence with the minimum transportation costs is *B*–*C*–*A*. Three loading layouts are presented for the given customer orders. The first two layouts (V1 and V2) are infeasible because the orders of customer A would block the unloading orders of customer C, who needs to be served before A. The only layout that would allow feasible unloading is the third (V3). However, this would require approaching the shipping gate of one segment twice. As a consequence, no feasible loading can be achieved for the tour *B*–*C*–*A*. To sum up, constraints related to collecting orders from one segment, sequencing of customers and sequencing of segments have to be considered for the loading of MCVs.

Note that the backhaul of empty transportation units (pallets or roll-cages) is not considered in this study, as we found that this is not an issue for our partners in practice. In operational practice, the driver decides how many units he returns from each store depending on available truck space. As empty pallets or roll-cages can be stacked, and thus require only a fraction of the space of full ones, their return is not a bottleneck.

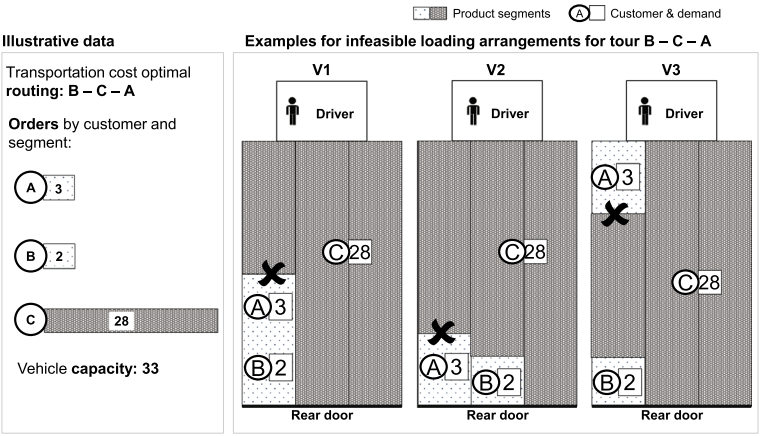

Fig. 4 Illustrative example for infeasible loading/unloading

2.2 Related literature

The 2L-MCVRP is a variant of the capacitated vehicle routing problem (CVRP). Based on the problem characteristics, there are two relevant streams within the vehicle routing problem (VRP) literature. On the one hand, as MCVs are a central aspect in our routing decisions, our work clearly extends the MCVRP. On the other, the problem concerning loading of MCVs that we are facing is related to the VRP with loading constraints.

Literature related to MCVRP While there is a wide range of publications dealing with the CVRP and its various extensions [see general overviews at Golden et al. (2008) and Toth and Vigo (2014)], there is only a small body of MCVRP literature. We are considering MCVs with flexible compartments, whose number and size can be adjusted. The MCVRP extends the CVRP by adding constraints for the building and loading of different compartments on each vehicle. Besides the delivery of groceries, there are several areas of application for MCVs, such as fuel distribution or waste collection. However, most publications in these areas consider the use of fixed compartments [e.g., Muyldermans and Pang (2010) for waste collection and Avella et al. (2004) for fuel distribution], instead of vehicles where compartments can be adjusted flexibly.

Derigs et al. (2011) are the first to present a comprehensive problem formulation for the MCVRP with flexible compartments. They formulate a general model for the application of their MCVRP for food and fuel distribution. The authors employ a solver suite combining different heuristic approaches to solve the MCVRP. This involves construction heuristics (e.g., sweep algorithm) as well as a combination of local search and metaheuristics. In addition, Derigs et al. (2011) identify a very efficient LNS with a combination of Shaw removal and regret insertion operators. Henke et al. (2015) consider the use of MCVs for the collection of glass waste. They allow flexible compartments, but restrict their sizes to predefined values. They also allow the use of more product categories than compartments. For their MCVRP, they apply a variable neighborhood search. A similar problem setting is taken into account by Koch et al.

(2016), who propose a genetic algorithm. Recently, Henke et al. (2017) have presented a B&C algorithm for the MCVRP in glass collection. Hübner and Ostermeier (2018) extend the MCVRP by considering process costs related to the use of MCVs in grocery distribution. They show that costs for loading/unloading have to be taken into account to achieve a more realistic evaluation of total costs due to the additional operations that are required when using MCVs. Ostermeier and Hübner (2018) analyze the settings under which MCVs, SCVs or a mixed fleet is efficient.

Literature related to VRP with loading constraints Different loading constraints have to be faced in many routing problems. The VRP with loading constraints is a combination of the CVRP and the loading problem. A general overview of loading problems can be found in Wäscher et al. (2007) and Bortfeldt and Wäscher (2013). A review on the combination of routing and loading problems was introduced by Iori and Martello (2010) and more recently by Pollaris et al. (2014). Côté et al. (2017) examine the value of integrating the loading decisions instead of solving routing and loading sequentially. Grocery retailers use non-stackable, standardized transportation units for the transport of goods from the DC to outlets. They typically use either pallets (e.g., discounters) or roll-cages (e.g., supermarkets). Therefore, the loading of MCVs in grocery distribution considers the assignment of two-dimensional objects of identical size (i.e., pallets or roll-cages) to identical containers (i.e., the loading area of identical vehicles). Consequently, we focus on the literature concerning the two-dimensional loading CVRP (2L-CVRP) that also considers sequence-based loading. The 2L-CVRP is first addressed by Iori et al. (2007). They propose a B&C algorithm for routing and an integrated branch-and-bound (B&B) framework to check loading feasibility. Similarly, Côté et al. (2014) develop a B&C algorithm that uses a one-dimensional contiguous bin packing problem to identify and eliminate infeasible loading. Besides these exact methods, most publications focus on heuristic solution approaches for the 2L-CVRP. Attanasio et al. (2007) solve a simplified integer linear program within a cutting plane framework for their variant of the 2L-CVRP, extended by multiple time windows for each day. Gendreau et al. (2008) present a tabu search (TS) for the 2L-CVRP. In their approach, they combine heuristics, lower bounds and a truncated B&B procedure to solve the loading problem. A guided TS is used by Zachariadis et al. (2009). Further, Zachariadis et al. (2013) propose a metaheuristic that uses a local search procedure for diversification. Fuellerer et al. (2009) address the 2L-CVRP where a rotation of items of 90 degrees is allowed. They apply an ant colony optimization to solve the respective problem. Finally, Pollaris et al. (2016) develop a special case of the 2L-CVRP, the CVRP with sequence-based pallet loading and axle weight constraints. They present a mixed integer linear program for their problem formulation and compare it to a model without axle weight restrictions.

2.3 Summary

Sequential loading of different segments, joint unloading of customer orders, the use of standardized transportation units, rear loading and tailored compartment layouts constitute a relevant but also complex decision problem for retailers that use MCVs.

Moreover, there has been no literature so far that addresses both two-dimensional loading and MCVRPs. In our problem, we combine the MCVRP and the 2L-CVRP formulation and thus leverage both streams in the literature. Iori et al. (2007) and Côté et al. (2014) inspired our sequential approach to tackle both routing and loading problems. However, in contrast to classical 2L-CVRP, in our case the orders of one customer can be put on the same tour as well as on different ones. A new cut structure is proposed to cope with this characteristic. We use an LNS framework for the routing problem as this showed very good results in various applications of Derigs et al. (2011) and Hübner and Ostermeier (2018) for MCVRPs. Moreover, we leverage Côté et al. (2014) to develop a B&C algorithm for the 2L-MCVRP to eliminate infeasible loading.

3 Two-dimensional loading MCVRP

The described loading requirements are incorporated into an MCVRP model to eliminate infeasible loading situations for the overall routing problem. The 2L-MCVRP can be defined as follows. For an undirected, weighted graph $G = (N, E)$ a set of vertices $N = \{0, 1, \ldots, n\}$ is given, representing the location of the DC ($\{0\}$) and the locations of n customers. The connection between different locations is represented by the set of edges $E = \{(i, j) : i, j \in N\}$, where each edge (i, j) is associated with traveling costs t_{ij}. It is assumed that all traveling costs satisfy the triangle inequality and each tour starts and ends at the DC. Let V be the set of vehicles available for transportation at the DC. The number of vehicles available is assumed to be sufficiently large to fulfill customer demand and consists of identical vehicles with capacity Q. The loading area of each vehicle can be split into a limited number of compartments, determined by the set of compartments C. Further, the number of compartments used on each vehicle is denoted by $k \in K$, with $K = \{1, \ldots, |C|\}$. Following Hübner and Ostermeier (2018), we consider an MCVRP with loading/unloading costs. The loading costs lc_k depend on the number of segment-specific shipping gates approached and are related to the number k of compartments used. The unloading cost ulc is incurred every time a customer is visited.

Consider that O is the set of orders to be delivered on a given day and q_m the quantity of each order $m \in O$ ($q_m > 0$). Let P be the set of product segments available for distribution. Further, $D_i \subseteq O$ is the subset of orders from customer $i \in N$ and $W_p \subseteq O$ is the subset of orders for a segment $p \in P$. Finally, to eliminate tours with infeasible loading, a set Ω of such tours is defined, where $\omega \in \Omega$ represents a tour that violates the loading. Furthermore, let $E_\omega \subseteq E$ be the subset composed by the sequence of edges that are traversed in the infeasible tour $\omega \in \Omega$. Moreover, let $O_\omega \subseteq O$ be the subset of orders considered in the infeasible loading tour $\omega \in \Omega$. An infeasible tour is therefore defined by the customer sequence, i.e., the edges traversed (E_ω) and the orders assigned (O_ω), which also implicitly defines the segments assigned. In this way, the set Ω includes all tours that violate the loading constraints presented in Sect. 2.

For the formulation of the 2L-MCVRP, we introduce the decision variable x_{mvc} that determines the assignment of order m to compartment c on vehicle v, and b_{ijv} that determines the travel sequence between customers i and j of vehicle v.

x_{mvc} $\begin{cases} = 1, & \text{if order } m \text{ is assigned to compartment } c \text{ on vehicle } v \\ = 0, & \text{otherwise} \end{cases}$

b_{ijv} $\begin{cases} = 1, & \text{if vehicle } v \text{ is traveling from customer } i \text{ to } j \\ = 0, & \text{otherwise.} \end{cases}$

Further auxiliary variables are introduced as denoted below.

a_{vc} $\begin{cases} = 1, & \text{if compartment } c \text{ on vehicle } v \text{ is active} \\ = 0, & \text{otherwise} \end{cases}$

e_{vk} $\begin{cases} = 1, & \text{if vehicle } v \text{ has } k \text{ active compartments} \\ = 0, & \text{otherwise} \end{cases}$

f_v represents the number of customer stops for vehicle v

$u_{iv} = h,\ h \in \{1, \ldots, n\}$ represents position h of customer i on vehicle v.

The mathematical model can be formulated as follows:

$$Minimize \sum_{v \in V} \left(\sum_{k \in K} lc_k \cdot e_{vk} + \sum_{i \in N} \sum_{j \in N} t_{ij} \cdot b_{ijv} + ulc \cdot f_v \right) \tag{1}$$

Subject to

$$\sum_{j \in N \setminus \{0\}} b_{0jv} \leq 1 \qquad v \in V \tag{2}$$

$$\sum_{i \in N} b_{igv} = \sum_{j \in N} b_{gjv} \qquad v \in V,\ g \in N \tag{3}$$

$$u_{iv} - u_{jv} + (n+1) \cdot b_{ijv} \leq n \qquad v \in V,\ i \in N,\ j \in N \setminus \{0\} \tag{4}$$

$$u_{0v} = 1 \qquad v \in V \tag{5}$$

$$\sum_{m \in O} \sum_{c \in C} q_m \cdot x_{mvc} \leq Q \qquad v \in V \tag{6}$$

$$\sum_{v \in V} \sum_{c \in C} x_{mvc} = 1 \qquad m \in O \tag{7}$$

$$\sum_{m \in D_j} \sum_{c \in C} x_{mvc} \leq |O| \cdot \sum_{i \in N} b_{ijv} \qquad v \in V,\ j \in N \setminus \{0\} \tag{8}$$

$$\sum_{m \in W_p} x_{mvc} \leq |O| \cdot (1 - \sum_{r \in W_q} x_{rvc}) \qquad v \in V,\ c \in C,\ p, q \in P : p \neq q \tag{9}$$

$$\sum_{m \in O} x_{mvc} \leq |O| \cdot a_{vc} \qquad c \in C,\ v \in V \tag{10}$$

$$\sum_{c \in C} a_{vc} = \sum_{k \in K} k \cdot e_{vk} \qquad v \in V \tag{11}$$

$$\sum_{k \in K} e_{vk} \leq 1 \qquad v \in V \tag{12}$$

$$f_v \geq \sum_{i \in N} \sum_{j \in N \setminus \{0\}} b_{ijv} \qquad v \in V \tag{13}$$

$$\sum_{m \in O_\omega} \sum_{c \in C} x_{mvc} + \sum_{(i,j) \in E_\omega} b_{ijv} \leq |E_\omega| + |O_\omega| - 1 \quad \omega \in \Omega,\ v \in V \tag{14}$$

$$x_{mvc}, b_{ijv}, a_{vc}, e_{vk} \in \{0,1\}; \quad f_v, u_{iv} \in \mathbb{Z}_o^+ \qquad m \in O, v \in V,\ c \in C,\ i, j \in N,\ k \in K. \tag{15}$$

The objective function (1) minimizes the total cost of all tours and consists of three parts: (i) loading, (ii) transportation and (iii) unloading costs. Constraints (2) and (3) represent routing constraints, guaranteeing that each vehicle departs at most once from the depot and that every vehicle that visits a location also leaves it again. Constraints (4) and (5) are used to eliminate subtours. The former determines the position of each location within the tour, while the latter ensures that the depot is the first in the sequence of locations. The overall quantity of the orders assigned to a vehicle cannot exceed the vehicle capacity as stated by Constraint (6). According to Constraint (7), each order can only be assigned to one compartment and one vehicle. Additionally, an order can only be assigned to a vehicle if the associated customer is visited on the tour (Constraint (8)). Constraint (9) ensures that orders from different segments are not assigned to the same compartment. To model the use of compartments, Constraint (10) ensures that a compartment is set to active if an order is assigned to it. Constraints (11) and (12) control the value for the number of compartments used. For this, e_{vk} has to be activated for the correct number of compartments and it can only be activated once for each vehicle. In addition, Constraint (13) ensures that the number of stops (variable f_v) is equal to the number of stores visited by each vehicle. Constraint (14) ensures that the infeasible tours in terms of loading are eliminated from the search space. This is guaranteed by excluding all infeasible combinations of orders (O_ω) and customer sequences (E_ω) of infeasible tours $\omega \in \Omega$. In Sect. 4.1, we present the loading model that identifies whether a tour is feasible or not. This is how we address loading issues in our solution approach. Lastly, the domains of the decision variables are defined by Constraint (15).

4 Solution approaches

The 2L-MCVRP is an extension of the CVRP, and therefore NP-hard (Toth and Vigo 2014). We provide an exact and heuristic approach to solve the problem. Following Iori et al. (2007) and Côté et al. (2014), we sequentially tackle the routing and loading problem. The loading requirements, described in Sect. 2, are included in both approaches using a specialized packing problem. Please note that the terms loading and packing are considered equivalent in the formulation of our loading problem.

In Sect. 4.1, we first present a B&C algorithm that iteratively solves the routing and loading problem by generating cuts for infeasible tours at each integer node. However, this exact method is only capable of solving small instances. Therefore, we further extend the LNS framework of Derigs et al. (2011) and Hübner and Ostermeier (2018) to solve problem sizes relevant in practice by incorporating a repair mechanism for infeasible loading tours.

4.1 Exact approach for the 2L-MCVRP

A B&C algorithm is proposed to solve the 2L-MCVRP as presented in Fig. 5. This exact approach is used to get a benchmark for the heuristic in terms of solution quality. In the first step, the MCVRP is solved without loading constraints by using the B&B framework provided by CPLEX, i.e., Constraint (14) of the 2L-MCVRP is not considered. As soon as a feasible solution for the MCVRP, and therefore an integer node is reached, loading feasibility is checked in Step (a). For this purpose, we introduce an integer programming (IP) formulation that is solved for each tour to find a feasible loading. Depending on the outcome of Step (a), there are two possible cases for Step (b). In the event of no loading violations, we obtain a feasible solution and continue with the branching. If no feasible loading can be found for a tour, the corresponding tour is declared infeasible and therefore added to the set Ω. This means that a cut is added to the 2L-MCVRP model.

In the following, we describe more precisely Steps (a) and (b) since they represent the main contribution of our B&C.

(a) Checking loading of tours The adding of cuts for our problem is based on a special packing problem for MCVs. As we consider the transportation of identical transportation units, i.e., pallets or roll-cages (henceforth denoted as *items*), the loading area and its capacity can be divided into a certain set of items (see the left-hand side of Fig. 6). To position each single item on the vehicle, the loading area is divided into $|X| \cdot |Y|$ possible spots. Here, $x \in X$ represents the rows available on the loading area and $y \in Y$ the different positions within a row. The orientation of these items is considered to be fixed in order to be able to move them in the required manner (e.g., pallets can be accessed using pallet trucks).

For the tour planning, we assign orders to compartments of vehicles. Each order $m \in O$ has a fixed quantity q_m that expresses the number of loading items. The goal is that each item is exactly assigned to just one position (x, y) on the vehicle (see the right-hand side of Fig. 6).

The loading of each tour is always feasible in terms of available capacity, which means that the capacity of all orders assigned does not exceed the vehicle capacity (see Constraint (6)). However, the crucial point is to ensure that all the items loaded can

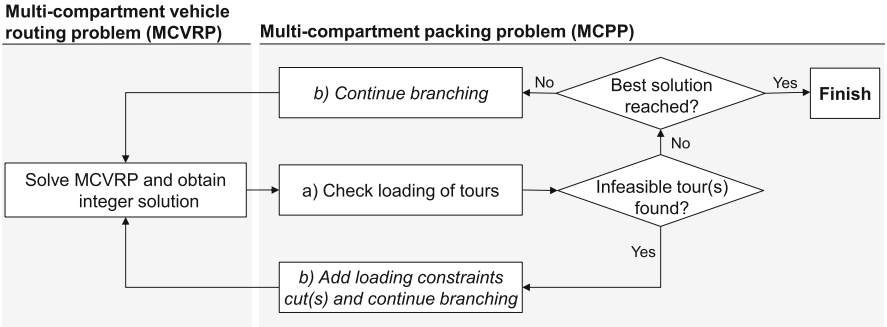

Fig. 5 Scheme of exact approach for 2L-MCVRP with branch-and-cut

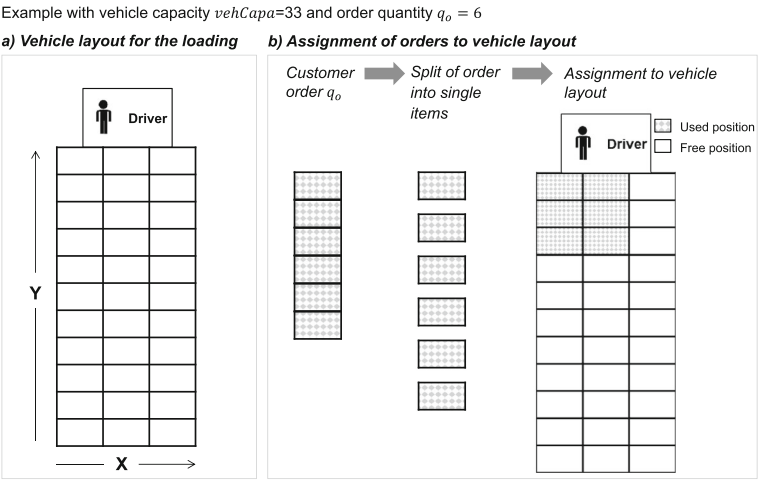

Fig. 6 Loading layout for the multi-compartment packing problem (MCPP) (illustrative example)

be accessed without rearrangement, according to the loading requirements described in Sect. 2.

The corresponding *multi-compartment packing problem* (MCPP) can be classified as an *identical item packing problem* (IIPP) (see Wäscher et al. 2007). It involves the assignment of identical items to exactly one large object (i.e., one vehicle) while maximizing the number of orders loaded. In our case, this requires the assignment of all orders while considering customer and segment sequence. If all orders cannot be assigned, the loading is not feasible on the corresponding tour.

For the mathematical formulation of the MCPP, we introduce additional sets and parameters to the 2L-MCVRP formulation in Sect. 3. Firstly, to take into account the customer sequence that the vehicle has to follow during the tour, the set $G_i \subseteq N$ contains the predecessors of customer $i \in N$. Predecessors of customer i are all customers visited earlier in the tour. Secondly, in order to make segments comparable in the sense of sequencing, we assign a number to all product segments in $P^* = \{1, \ldots, |P|\}$. As presented in the 2L-MCVRP (see Sect. 3), the subsets D_i and W_p are used to represent all orders from a customer $i \in N$ and all orders from a segment $p \in P$, respectively. Lastly, M is used as a sufficiently large number to regulate the setting of our decision variables. We introduce the following decision variables for the formulation of the MCPP:

$$\alpha_{mxy} \begin{cases} = 1, & \text{if an item of order } m \text{ has been assigned to position } (x, y) : \ x \in X, \ y \in Y \\ = 0, & \text{otherwise} \end{cases}$$

$$\beta_{pq} \begin{cases} = 1, & \text{if segment } p \in P \text{ is loaded after } q \in P \text{ in the segment sequence} \\ = 0, & \text{otherwise.} \end{cases}$$

As we want to define a feasible loading, all orders assigned to the vehicle have to be loaded. Therefore, the MCPP does not require an objective function, but has to respect the following constraints.

$$\sum_{x \in X} \sum_{y \in Y} \alpha_{mxy} = q_m \qquad m \in O \tag{16}$$

$$\sum_{m \in O} \alpha_{mxy} \leq 1 \qquad x \in X,\ y \in Y \tag{17}$$

$$\beta_{pq} + \beta_{qp} = 1 \qquad p, q \in P^* :\ p > q \tag{18}$$

$$\beta_{pq} + \beta_{qf} \leq 1 + \beta_{pf} \qquad p, q, f \in P^* :\ p \neq q \neq f \tag{19}$$

$$y \cdot \alpha_{mxy} \leq y' \cdot \alpha_{rxy'} + M(2 - \alpha_{rxy'} - \beta_{pq}) \qquad p, q \in P^*,\ p \neq q,\ m \in W_p,\ r \in W_q,\ x \in X,\ y, y' \in Y \tag{20}$$

$$y \cdot \alpha_{mxy} \leq y' \cdot \alpha_{rxy'} + M(1 - \alpha_{rxy'}) \qquad i \in N,\ j \in G_i,\ m \in D_j,\ r \in D_i,\ x \in X,\ y, y' \in Y \tag{21}$$

$$\alpha_{mxy}, \beta_{pq} \in \{0, 1\} \qquad m \in O,\ x \in X,\ y \in Y,\ p, q \in P^*. \tag{22}$$

Constraint (16) ensures that each order assigned to the vehicle has all items loaded. This is ensured by setting the sum of all assigned items of an order m equal to the order quantity q_m. Only one item can be assigned to each position in the vehicle, which is guaranteed by Constraint (17). The requirements for customer and segment sequence, as described in Sect. 2, are represented by Constraints (18) to (21). First, Constraints (18) and (19) define the segment sequence, i.e., in which sequence the segment-specific shipping gates should be visited. Second, Constraint (20) ensures that the defined sequence of segments is not violated during the loading and thus the orders of each segment are jointly collected. This means that if two segments are assigned to the same aisle, the first segment in the sequence and thus the corresponding orders are loaded onto the vehicle first. Hence, the second segment has a position closer to the rear door (see Fig. 4). The customer sequence is ensured by Constraint (21). Variable domains are defined by Constraint (22).

(b) Adding cuts or continuing branching If the loading is not feasible, cuts of type (14) are added to the 2L-MCVRP model (1)–(15) and branching goes on. As previously described, these cuts take into account the special characteristics of our problem setting. In contrast to classical VRPs with loading constraints, in our case the orders of one customer can be put on the same tour as well as on different ones. This is illustrated in Fig. 7, where the data for the simplified example are given on the left-hand side and possible loading solutions on the right-hand side.

The example shows the solution for a tour with three customers (A, B and C) and the corresponding orders denoted by the customer and the order number for different segments (e.g., A1). The best MCVRP solution found for this tour is the customer sequence A–B–C, which is displayed in the upper part (i). However, this is an infeasible tour in terms of loading, because order C2 is blocking order B1 and customer B is supplied before C. If we add a cut considering only the customer sequence, all possible order combinations for the sequence A–B–C would also be cut. However, as displayed in the part (ii) of Fig. 7, the same customer sequence can result in a feasible tour if other orders from the same customers are assigned to the vehicle. Just by exchanging the segments assigned, a feasible loading is possible for the tour. This means that not

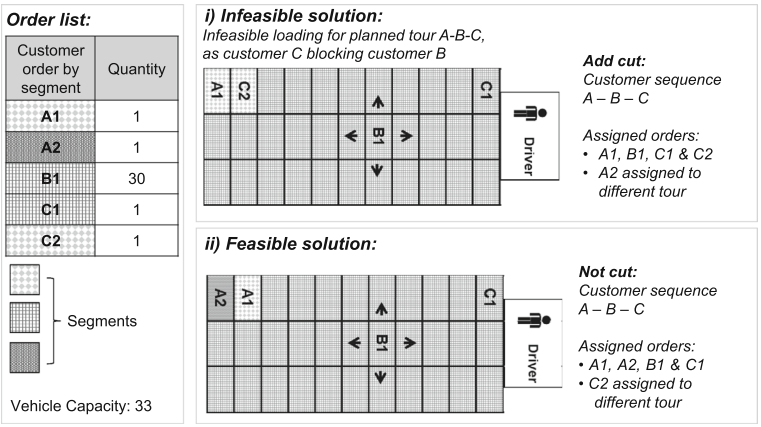

Fig. 7 Example for added cuts: customers and orders define cut

only the customer sequence needs to be considered, but also which orders of each customer are assigned. Therefore, two tours with the same customer sequence but different order structures might differ in terms of loading feasibility. After introducing the cuts, the branching procedure continues until the best solution is reached (and proved optimal).

4.2 Heuristic approach for the 2L-MCVRP

We extend a large neighborhood search (LNS) framework by incorporating loading constraints. This involves the inclusion of a repair mechanism that checks the tours loading feasibility and repairs it, if no feasible loading is possible. The solution approach is divided into two phases, and its overall structure is given in Fig. 8.

First, an LNS is applied solely to the MCVRP, which is denoted as standard LNS (sLNS), and provides a solution for the MCVRP. Afterward, the final solution is checked for infeasibilities. If the sLNS solution shows tours with infeasible loading, the search continues solving the 2L-MCVRP. At this stage, a repair mechanism is incorporated into the previous LNS framework, now denoted as repair LNS (rLNS). Both LNS variants are similar; however, the rLNS calls the repair mechanism each time a new best candidate solution is found. If an infeasible loading is detected in one tour, it is repaired and the resulting solution is checked again by the acceptance criteria. The rLNS only saves the best solutions that satisfy the loading constraints. The main features of the sLNS framework used are described in Sect. 4.2.1, followed by the description of the repair mechanism used in the rLNS (Sect. 4.2.2).

4.2.1 Standard LNS (sLNS) for the MCVRP

We used the LNS framework proposed by Derigs et al. (2011) and Hübner and Ostermeier (2018) as the basis for our sLNS heuristic. The sLNS starts by generating a solution using the parallel savings algorithm presented by Clarke and Wright (1964).

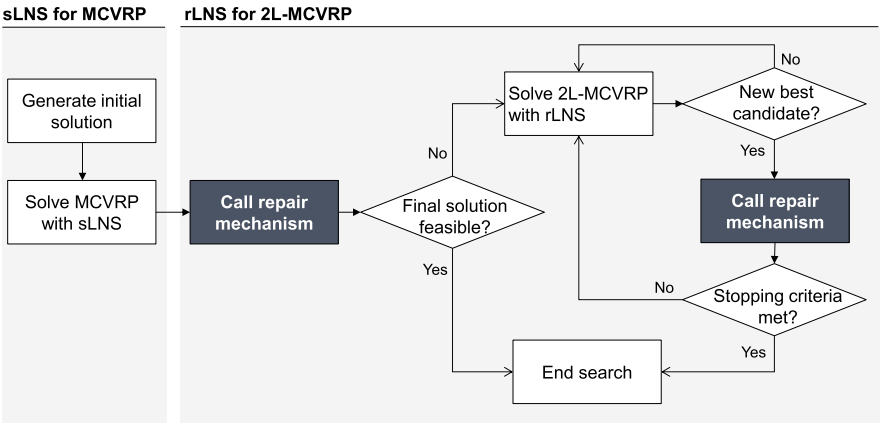

Fig. 8 Scheme of rLNS approach

It operates by building single customer tours for each given order and then combines tours according to the savings in traveling distance.

Algorithm 1 outlines the sLNS framework. From the initial solution we move on by applying the Shaw removal (Shaw 1997) as destroy operator to remove orders from the incumbent solution. In the following step, the orders removed are reinserted into tours using the regret-k insertion as repair operator. It was shown in previous works on MCVRPs that these operators are able to efficiently generate good solutions (Derigs et al. 2011). At the end, a new solution is defined, and if it fulfills the defined acceptance criterion, it replaces the incumbent solution. The procedure is repeated until a final criterion is met. In the subsequent paragraphs a further description of the procedure is presented, specifically the operators and acceptance criteria used.

Shaw Removal The Shaw removal was introduced by Shaw (1997) and is based on a similarity measure R_{ml} between two randomly selected orders m and l (either from the same customer or from different customers) with $m, l \in O$. A predefined number r of orders has to be removed from all orders O. In the following, we divide the set of orders O into removed orders (O^-) and assigned orders (O^+), such that $O^+, O^- \subseteq O$, $O^+ \cup O^- = O$ and $O^+ \cap O^- = \emptyset$.

The first order m removed is chosen randomly from the set O. Further orders are gradually removed according to a procedure based on the similarity measure R_{ml}. This similarity index expresses the similarity of two orders m and l, with $m \in O^-$ and $l \in O^+$, and is calculated using Eq. (23). The higher the calculated value of R_{ml} is, the less similar the two orders are. Derigs et al. (2011) propose a modified similarity measure to include the MCVRP particularities. The measure is given below and combines traveling cost, order segment and order quantity.

$$\frac{1}{R_{ml}} := \phi \cdot \frac{\text{cost}_{ml}}{\text{cost}_{\max}} + \omega \cdot \text{segment}_{ml} + \psi \cdot \frac{|\text{quantity}(m) - \text{quantity}(l)|}{\text{quantity}_{\max}}. \quad (23)$$

Algorithm 1 standard Large Neighborhood Search (sLNS)

Require: (initial solution S, set remove parameter r, set regret parameter k)
1: $S_{best} := S$
2: Shaw-Removal $SR(r)$
3: Regret-Insertion $RI(k)$
4: Set allowed deviation according Record-To-Record Travel $RRT()$
5: **while** improving = **true do**
6: $S' := S$
7: Remove r orders from S' using $SR(r)$
8: Reinsert removed orders into S' using $RI(k)$
9: **if** $ObjectFunction(S') < ObjectFunction(S_{best})$ **then**
10: $S_{best} = S'$
11: $S = S'$
12: Reset number of unsuccessful runs to zero
13: **else if** $ObjectFunction(S') \leq ObjectFunction(S_{best})$ plus accepted deviation D **then**
14: $S := S'$
15: Increase number of unsuccessful runs
16: **else**
17: Increase number of unsuccessful runs
18: Continue with original solution S
19: **end if**
20: **if** Number of unsuccessful runs = limit **then**
21: Set improvement **false**
22: **else if** Number of unsuccessful runs = reset border **then**
23: Remove high number of orders from S using $SR(r)$
24: Reinsert removed orders into S using $RI(k)$
25: **end if**
26: **end while**
27: **return** S_{best}

The parameters $\text{cost}_{\max}$ and $\text{quantity}_{\max}$ indicate the maximum traveling cost and order quantity over all orders considered. Furthermore, segment_{ml} is set to 1 if two orders are from different segments and cannot be allocated to the same compartment, and 0 otherwise.

After the first order m has been randomly selected, the order's similarity is calculated. In the removal process, the similarity measure is combined with a randomization step in order not to choose the most similar order, but a random order among the similar orders instead. Once the similarity index is calculated, the orders are ranked according to similarity in descending order. Afterward, the next orders for removal are selected based on the random number $z \in [0, 1)$ and a parameter α. The parameter α can be seen as a parameter for diversification. If α is chosen to be 1, the similarity is not taken into account and the choice of orders is completely random. Thus, for the selection of a new order, it is not the order with the highest similarity that is chosen, but the one that can be found $z^{\alpha} \cdot 100$ percent down the similarity ranking. After r orders are removed, the algorithm continues with the reinsertion phase.

Regret-k Insertion Repair is done with the regret-k insertion operator of Ropke and Pisinger (2006). It is less myopic than a greedy approach as it considers information on the postponed insertion of orders. More precisely, it takes into account the k best insertions instead of only the best one. Based on the cost difference between these k

options and the best option, a regret value is calculated using Eq. (24). Let $\text{totalcost}^m_{v_u}$ denote the cost of adding a selected order $m \in O$ at its best position on the uth best tour. The cost calculation is based on the objective function (1).

$$\text{regret}^m_k := \sum_{u=2}^{k} \left(\text{totalcost}^m_{v_u} - \text{totalcost}^m_{v_1}\right). \tag{24}$$

The order m with the highest regret value is selected and inserted into its best tour. This means that the order selection not only is based on the actual state, but also considers what could be lost later if an order is not immediately inserted at its best position. After the order m with the highest regret is inserted into O^+, the regret value is recalculated for each order s on the removal list O^-, as the insertion options might have changed. The insertion procedure is thus iterated until all orders removed, $m \in O^-$, are allocated to a new position in the tour planning, and hence $O^- = \emptyset$ and $O^+ = O$.

Record-To-Record Travel The search operators of the sLNS are embedded into an RRT framework as presented in Dueck (1993). During the search, a new solution is only accepted if it provides an improvement or if it lies within a predefined deviation from the best known solution so far. Furthermore, the following procedures are integrated in the search for the regulation of the application runtime. First, every time the number of unsuccessful iterations reaches a reset border, the next iteration of the sLNS is run with a high value (e.g., one half of the order list) for the number of orders removed (r). This allows to create a completely different neighborhood, thus aiming to avoid local minima. Second, a limit is set for the maximum number of consecutive fruitless iterations to terminate the algorithm.

4.2.2 LNS with repair mechanism (rLNS) for the 2L-MCVRP

The incorporation of a repair mechanism for loading constraints is the central aspect of our extension of the MCVRP. The rLNS framework starts by calling the repair mechanism to check the loading feasibility of the sLNS solution obtained. If the sLNS solution provides feasible tours, the search ends. Otherwise, the repair mechanism changes the tours in order to make them feasible and the rLNS continues solving the 2L-MCVRP (see Fig. 8). This step is required as in contrast to the exact approach, no cuts can be used during the heuristic search.

The rLNS framework is similar to the sLNS (see Algorithm 1). However, each time the algorithm finds a new best candidate solution (line 9 of Algorithm 1), instead of saving it as the best solution, it calls the repair mechanism to check its feasibility. If the solution is declared feasible it is saved as best and the search continues. Otherwise, the solution is repaired and checked again by the acceptance criteria. Note that by repairing the solution we are changing some decisions that could result in an increase in the solution cost. An overview of the repair mechanism is presented in Fig. 9, followed by a description of each step.

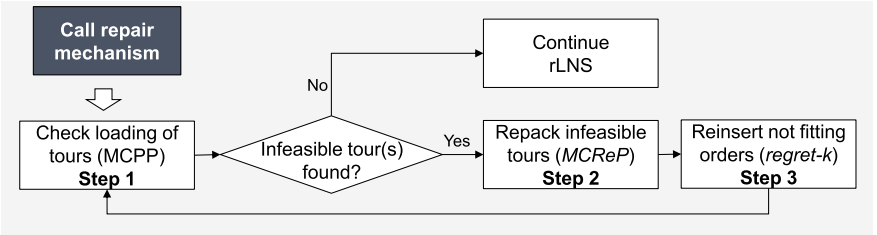

Fig. 9 Procedure of repair mechanism within rLNS

Step 1: First, the repair mechanism applies the MCPP model, presented in Sect. 4.1, to each tour of the solution to assess its loading feasibility. If an infeasible tour is found, a repacking phase is applied to remove the orders that do not completely fit on the truck without violating the loading constraints. This procedure is carried out for all tours in the current solution until every tour has been checked for feasibility.

Step 2: The repacking phase uses a model denoted as *multi-compartment repacking problem* (MCReP). The MCReP is based on two modifications of the MCPP model. Instead of aiming to find a feasible loading for a tour, the MCReP tries to maximize the number of items loaded (from the orders assigned to the tour) while satisfying the loading constraints. To achieve this, the MCPP model is altered by adding an objective function and modifying Constraint (16). The MCReP is defined as follows:

$$\text{Maximize} \sum_{m \in P} \sum_{x \in X} \sum_{y \in Y} \alpha_{mxy} \tag{25}$$

subject to: (17) − (21)

$$\sum_{x \in X} \sum_{y \in Y} \alpha_{mxy} \leq q_m \quad m \in P \tag{26}$$

$$\alpha_{mxy}, \beta_{hz} \in \{0, 1\} \quad m \in P,\ x \in X,\ y \in Y,\ h, z \in S. \tag{27}$$

The new Constraint (26) allows the assignment of a number of items lower than the quantity of each order. This constraint together with the objective function (25) results in a model that aims to maximize the number of items loaded without violating the loading constraints. Therefore, we obtain a solution with feasible loading for the considered tour, but also a solution that does not completely assign all orders, as the loading of all orders is not feasible.

Step 3: By the end of the repacking phase, every order that caused non-feasible loading for a given tour will have been added to the list of orders removed (O^-). Based on this list, the regret-k insertion is applied to find the best option for the removed orders. To prevent cycling, a tour with removed orders is set tabu for the corresponding orders in the reinsertion phase. As the reinsertion causes changes to the structure of affected tours, the feasibility has to be assessed again. Therefore, the MCPP is applied to all modified tours within the reinsertion process and, if a tour is

found to be infeasible, the repacking is applied again, as described above. This process is repeated, until a feasible loading has been found for all tours within the solution. Note that a new tour is created if it is the lowest total cost option or if all available tours are set tabu for an order. The algorithmic structure of the repair mechanism is presented in Algorithm 2.

Algorithm 2 Repair mechanism

```
Require: Routing solution S
MCPP model approach as given in 4.1
Regret-Insertion I(k)
ST = Number of Tours in S
for i = 1 to ST do
    Apply MCPP to tour i
    if Loading of tour i is infeasible then
        Start repacking phase
        Remove order(s) that did not fit on tour
        Set tour i tabu for removed order(s)
    end if
end for
Reinsert removed orders into S using I(k)
while Not all tours feasible do
    for All changed tours do
        Apply MCPP to selected tour
        if Loading of selected tour is infeasible then
            Start repacking phase
            Remove order(s) that did not fit on tour
            Set tour tabu for removed order(s)
            if All tours are set tabu for one order then
                Create new tour for this order
            end if
        end if
    end for
    Reinsert removed orders into S using I(k)
end while
return S
```

5 Numerical experiments

In this section, we present numerical experiments to evaluate our solution approaches and grasp the importance of including loading constraints in the MCVRP. As a basis for our experiments, we present our data generation structure in Sect. 5.1. An overview of all tests present in Sects. 5.2 to 5.4 is given in Table 1.

In Sect. 5.2, we use small size instances and compare the results of the 2L-MCVRP with the results of the MCVRP (without loading constraints) obtained by the corresponding exact approaches. Furthermore, we investigate the differences between the heuristic and the B&C approaches. This test aims to identify the efficiency of the heuristic in terms of solution quality and runtime. Additionally, the results of the LNS with the repair mechanism (rLNS) are compared to the sLNS solution with an ex post evaluation of loading feasibilities. This is achieved by applying the repair mech-

Table 1 Overview of numerical tests

Sections	Model	Solution approaches	Data applied	Test purpose
5.2.1	MCVRP and 2L-MCVRP	MIP Solver and B&C	Small randomly generated data	Impact of loading constraints for small instances
5.2.2	2L-MCVRP	B&C, rLNS, sLNS_ExPost	Small randomly generated data	Effectiveness of heuristic approach
5.3	2L-MCVRP	rLNS, sLNS_ExPost	Large randomly generated data	Efficiency of heuristic and impact of loading constraints for larger instances
5.4	2L-MCVRP	rLNS, sLNS_ExPost	Case study	Impact of loading constraints in practice

anism only to the final solution of the sLNS framework. We denote this approach as sLNS_ExPost. In Sect. 5.3, we consider tests with larger instances. Here, we test the rLNS as well as the sLNS_ExPost to obtain further insights regarding runtime and the influence of different data structures on loading constraints. In Sect. 5.4, we apply the methodology to a real-life case example of a major European grocery retailer.

5.1 Overview of test instances

In order to analyze the impact of loading constraints in the MCVRP, we use randomly generated problem instances. These instances can be found on http://www1.ku.de/wwf/pw/forschung/. We leverage our data generation on the data structures that we obtained from our case study (see Sect. 5.4). For both small- and large-sized problems, we define the number of customers, the number of orders per customers, i.e., the number of different segments each customer orders, and the order quantity.

Small instances As the problem becomes very hard to solve exactly with an increasing number of customers and/or orders, we generate small instances. Please note that for the MCVRP the main drivers for complexity are the number of orders and related segments, and not the number of customers as in classical VRP formulations. In the first type of instance (=Scenario 1) there are four customers and four segments and each customer orders all segments. This results in a total of 16 orders. The second type of instances (=Scenario 2) comprises ten customers with each customer ordering only one out of four available segments. For these small problems, the distance matrix is randomly generated with distances between any two customers set at between 7 and 80 km. The distances between the customers and the DC are set at between 90 and 120 km to create a clustered group of customers.

Large instances The larger instances are created for 25, 50, 75 and 100 customers with two or four segments. As not all customers have demand for all segments, it is also possible that not all available segments are ordered. Therefore, different sets of instances are generated, changing the number of segments ordered. The distance

Table 2 Range of order quantity per segment (in TU)

Segment	Minimum quantity	Maximum quantity
1	1	5
2	1	10
3	5	20
4	10	25

Table 3 Applied costs for loading MCV

# Segments to be loaded	1	2	3	4
Loading (CU/shipping gate)	2.70	5.57	8.27	10.97

design in Solomon (1983) for the VRP with 100 customers is used for all tests and has been adjusted to our problem instances. We use the instances related to clustered customers (C-type) and uniformly distributed customers (R-type).

In food distribution, the mix of orders between customers and segments is very heterogeneous, as the analysis of our case study has shown (see Sect. 5.4). Using this information, the quantity of each order depends on the segment in question and varies between a given range. The ranges used are presented in Table 2. For instance, segment 4 could comprise the ambient assortment with larger order sizes and segment 1 could be seen as deep-frozen where orders are usually small. The segments associated with each customer are chosen randomly depending on the number of orders a customer submits. For instance, in the case where each customer orders one out of four segments, one segment is selected randomly to generate the orders. The quantity of orders is defined according to the segments in question.

Homogeneous MCVs with a total capacity of 33 transportation units (TU) are used for all tests. The loading/unloading costs used in the experiments have been derived from our case study. The loading costs depend on the number of segments to be loaded at the DC and are presented in Table 3. Unloading costs increase with every customer stop and are set to 2.20 currency units (CU). The transportation costs are based on the travel distances between any two locations.

The computational results were obtained on a 3.60 GHz PC with 16 GB memory. The implementation of the B&C algorithm has been done in C++ and the LNS algorithm in Java. The setting of the heuristic-specific parameters comes from previous applications of an LNS to the MCVRPs, since they performed very well on different instances related to our case (Hübner and Ostermeier 2018; Derigs et al. 2011). Accordingly, the weights for the calculation of the similarity measure R_{ml} were set to $\phi = \omega = 0.4$ and $\psi = 0.2$ and $\alpha = 4$ for the Shaw removal. This choice of weights is based on the higher influence of distance costs and product segments compared to order size. The number of items removed r is chosen randomly using a uniform distribution dependent on the respective problem size as given in Table 4. The regret parameter has been set at $k = 2$. Furthermore, the termination limit is 1000 iterations and the limit for a solution reset is 500 iterations for all LNS applications. The maximum deviation D allowed for the RRT equals 0.4% for all tests.

Table 4 Number of removed orders (r)

# Customer	Minimum	Maximum
10	2	5
25	2	10
50	5	15
75	5	20
100	5	30

5.2 Comparison between exact and heuristic approaches for small instances

We present two comparisons concerning small problems. First, in Sect. 5.2.1, we compare the exact solutions of the MCVRP (without loading constraints) with the exact solutions for the 2L-MCVRP provided by our B&C algorithm. This experiment aims to understand the cost implications of adding loading constraints. Then, the results obtained are compared to the results obtained with the heuristic approaches, in Sect. 5.2.2, to validate the efficiency of the heuristics. These comparisons are based on ten different instances for each scenario. The B&C approach was concluded after two hours as it became very hard to prove optimality. However, the search ends with an average solution gap of 0.01%. For the heuristic approaches, 50 repetitions are applied to create a sample of results for comparison. As the heuristic provides stable results with an average variation coefficient (standard deviation/mean) below 1%, the comparisons are made based on the average result achieved for each instance.

5.2.1 Comparison of exact approaches for 2L-MCVRP versus MCVRP

This experiment uses the B&B of the CPLEX MIP solver and the B&C algorithm to solve the MCVRP and the 2L-MCVRP, respectively. The comparisons are based on Scenario 1 and Scenario 2 from the small instances, as introduced above. The cost deviations between the 2L-MCVRP solution and the MCVRP solution achieved for each scenario are presented in Fig. 10. The results show that the cost deviation of considering loading constraints is at the most 0.5% for Scenario 1 and between 1.2 and 2.3% for Scenario 2. Even though this is a rather small improvement, it shows that even for small instances, the loading constraints have an impact on the routing decisions, particularly for Scenario 2. Here, the impact on costs is higher than in Scenario 1, as ten different customers are considered with only one order each. Any move of an order is equal to the move of a customer to a different tour and has a high influence on the routing, as only ten customers are available. To sum up, tours with feasible loading can be achieved with a small increase in the costs and thus would be preferable to a solution that would require rearrangement of the orders. In most cases, the exchange of two orders or the reassignment of a single order is sufficient to obtain feasible tours.

5.2.2 Comparison of exact approaches versus heuristics for 2L-MCVRP

This second numerical experiment aims to analyze the efficiency of the heuristic approaches. It compares the solutions achieved by the rLNS and the sLNS_ExPost

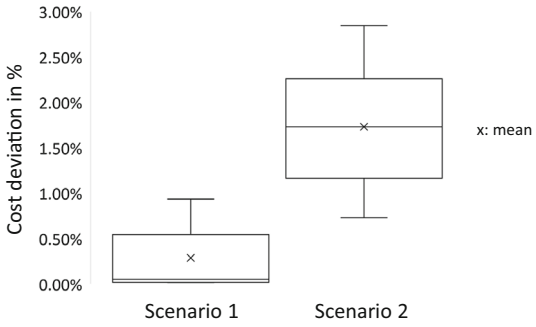

Fig. 10 Comparison of the results for 2L-MCVRP versus MCVRP with exact approaches, *10 instances per scenario*

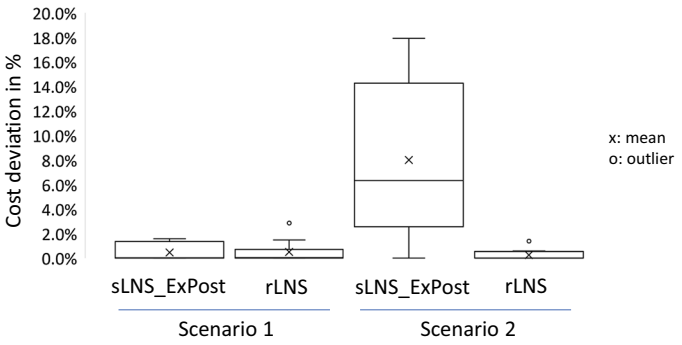

Fig. 11 Comparison of two heuristics versus exact approach for 2L-MCVRP, *10 instances per scenario*

with the solutions from B&C for the 2L-MCVRP. Figure 11 shows the cost deviations of both heuristic approaches to the B&C for the same two scenarios (with 4 and 10 customers) used previously.

The results indicate that the rLNS usually achieves the optimal solutions of the 2L-MCVRP for both scenarios. The sLNS_ExPost approach also achieves solutions with an average deviation close to 0% for Scenario 1. For Scenario 2, where only 1 out of 4 segments are ordered, the sLNS_ExPost provides solutions with an average deviation of 8%. In this scenario, merely moving orders to different tours is equal to moving customers and therefore has a higher impact on costs.

5.3 Analysis of loading constraints for larger instances

Subsequent to our tests with small instances, we analyze the impact of loading constraints for larger instances. Firstly, we analyze the runtime of the heuristic for increasing problem sizes. Secondly, we present tests for different problem settings to examine the impact of loading constraints. For these tests with larger instances, the results of 20 instances for each of the given scenarios are compared. As in the exact approach analysis, 50 repetitions of the heuristic approaches were applied to each instance to create a sample of results for comparison. As the solution approaches also provided stable results for the larger instances with an average variation coefficient

Table 5 Runtime for increasing problem sizes (min), *20 instances with 50 runs*

#Customers	#Segments ordered	sLNS_ExPost runtime			rLNS runtime		
		Average	Maximum	StDev	Average	Maximum	StDev
25	1 out of 4	0.10	0.94	0.10	4.54	27.30	2.88
	4 out of 4	0.06	0.64	0.07	0.41	16.80	0.78
50	1 out of 4	0.12	1.12	0.12	6.16	61.62	7.61
	4 out of 4	0.21	2.18	0.17	1.21	29.65	1.77
75	1 out of 4	0.17	1.03	0.12	16.01	158.50	22.90
	4 out of 4	0.34	1.56	0.18	1.27	37.40	2.25
100	1 out of 4	0.54	0.80	0.09	7.12	66.90	8.60
	4 out of 4	0.17	1.36	0.17	1.43	11.80	1.44

below 1%, the comparisons are made based on the average result achieved for each instance.

5.3.1 Runtime

The problem complexity and requirements for the heuristic approach increase quickly for larger problems. Therefore, we decided to focus on the runtime for the rLNS and the sLNS_ExPost for 25, 50, 75 and 100 customers. The C-type distance matrix is considered for the tests performed, since this is more related to retail practice, where outlets are located in populated areas. These areas are clusters. In all tests, four segments are available. However, as in the tests with the exact approach, we consider two different scenarios for all order structures, where either all four segments are ordered by each customer, or only one segment per customer. The average, maximum and standard deviation of the computational time (runtime) required to solve each instance of each scenario are summarized in Table 5. Since there are 20 instances for each scenario and each instance runs 50 times, the results presented are based on 1000 ($20 \cdot 50$) individual runs.

Our tests show that the number of segments ordered is an important driver for the computational time required to solve the 2L-MCVRP. We see that for the same number of customers, changing the segments ordered from the 4 out of 4 case to the 1 out of 4 case leads to an increase in the rLNS runtime. All runtime indicators increase. A higher number of customers does not have an influence on runtime. This can be attributed to the number of infeasible solutions achieved during the search. An increasing number of customers influences the routing decision, but it is the mix of orders that indicates the number of compartments to be used, and it has a greater impact on loading feasibility. This aspect will be analyzed in more detail in Sect. 5.3.2. The bottleneck for the rLNS is the repair mechanism and therefore the call of the MCPP and the MCReP models. As more infeasible solutions are found during the search, the more repacking phases are required and the runtime of the rLNS increases. This can be found, for example, in the tests with 75 customers and 1 out of 4 segments ordered, which have higher runtimes than the remaining scenarios. As the runtime is highly driven by infeasibility tests, even though fewer customers are considered, three out of

the 20 instances tested had a high number of infeasible tour combinations. Analysis of the sLNS_ExPost results shows that it is possible to solve all scenarios in less than three minutes, as the repair mechanism is only required once.

5.3.2 Scenario analysis for solution quality

In this subsection, we analyze the impact of loading constraints for different large data settings. This also enables us to gain further insight into the runtime of the suggested approaches. This analysis is based on the randomly generated instances with 25 and 100 customers. We would like to note that even in the case of 25 customers we are considering up to 100 orders. In contrast to other VRPs, the order number is an essential parameter for defining the problem size. Three different scenarios are considered for the number of segments ordered: the 1 out of 4 and 4 out of 4 scenarios (Scenarios 1 & 2), previously used, and an additional scenario with only two segments available, and both ordered by all customers (2 out of 2, Scenario 3). We emphasize the case with four segments as usually retailers are confronted with this situation. We further examine changes in customer distribution (matrix type) and loading costs (costs) in each scenario. The matrix types comprise clustered (C) and uniformly distributed (R) customers. Normal loading costs (NC) are used as given in Table 3 and also lower costs (LC) with a 50% reduction (i.e., LC = 50% NC). This change in loading costs can lead to changes in the compartment setting of MCVs (see Hübner and Ostermeier 2018), i.e., the lower the loading costs, the higher the tendency to use more compartments. Besides the number of customers, all settings and scenarios were the same for the cases with 25 and 100 customers. In this numerical experiment, we run the sLNS for the MCVRP (without loading constraints), and if the final solution reached shows infeasible tours, we run the sLNS_ExPost and the rLNS. Table 6 summarizes the result for 25 customers, and Table 7 for 100 customers. We indicate the number of instances (out of the 20) with infeasible tours for the best final solution achieved by the sLNS for the MCVRP for each scenario (see column 3 in Tables 6 and 7). We further compare the average cost of the final solutions provided by the rLNS and the sLNS_ExPost as well as the maximum improvement achieved by the rLNS. Note that the rLNS never provides a worse solution than the sLNS_ExPost as it continues the search from the sLNS_ExPost solution (see Fig. 8). The runtime required to solve each instance with the rLNS is also presented. The runtime of the sLNS_ExPost was always below one minute for all scenarios with 25 customers and was below four minutes for the scenarios with 100 customers.

Scenario 1 In the first scenario, up to 25% of instances with 25 customers lead to infeasible solutions (see Table 6). For 100 customers (see Table 7), this value amounts to over 50%. The results show that the rLNS can improve the ex post application of loading constraints from 1% up to 3%, for the case of 25 customers. When we increase the number of customers, the average cost deviation between the two approaches is close to zero. However, the best solutions found by the rLNS can reach an improvement of up to 3.4%. The runtime of the rLNS for the 25- and 100-customer cases shows a moderate increase considering that the number of orders increases from 100 (25-customer case) to 400 (100-customer case).

Table 6 Test overview for instances with 25 customers, *20 instances per scenario*

# Segments ordered	Matrix type[1]	Loading costs[2]	Infeasible instances[3] (%)	Average cost deviation (%)	Maximum improvement[4] (%)	rLNS runtime (min)	
						Average	Maximum
Scenario 1	C	NC	25	−1.32	−2.81	1.17	16.80
	C	LC	25	−1.04	−2.65	0.92	10.46
(4 out of 4)	R	NC	10	−1.11	−1.65	0.62	2.88
Scenario 2	C	NC	35	−3.62	−4.30	3.62	27.30
	C	LC	15	−4.79	−8.44	1.75	6.99
(1 out of 4)	R	NC	20	−2.23	−2.66	2.44	16.64
Scenario 3	C	NC	0	–	–	–	−−
	C	LC	0	–	–	–	−−
(2 out of 2)	R	NC	0	–	–	–	−−

[1] According to Solomon (1983), *C* clustered customers, *R* uniformly distributed
[2] Loading cost types: *NC* normal costs (see also Table 3), *LC* lower costs, reduced by 50% compared to NC
[3] Percentage of instances with sLNS solutions that contain at least one infeasible tour (i.e., violate loading constraints)
[4] Representing the maximum cost improvement of rLNS versus sLNS_ExPost

Table 7 Test overview for instances with 100 customers, *20 instances per scenario*

# Segments ordered	Matrix type[1]	Loading costs[2]	Infeasible instances[3] (%)	Average cost deviation (%)	Maximum improvement[4] (%)	rLNS runtime (min)	
						Average	Maximum
Scenario 1	C	NC	50	−0.28	−3.38	1.27	11.80
	C	LC	35	−0.27	−2.94	1.88	27.77
(4 out of 4)	R	NC	55	−0.36	−3.29	2.60	37.33
Scenario 2	C	NC	100	−1.98	−9.71	7.12	66.90
	C	LC	80	−1.74	−10.57	8.70	101.80
(1 out of 4)	R	NC	95	−1.44	−7.51	11.15	85.15
Scenario 3	C	NC	25	−0.22	−3.23	0.57	9.81
	C	LC	30	−0.22	−3.73	0.47	9.88
(2 out of 2)	R	NC	30	−0.22	−2.80	0.76	16.20

[1] According to Solomon (1983), *C* clustered customers, *R* uniformly distributed
[2] Loading cost types: *NC* normal costs (see also Table 3), *LC* lower costs, reduced by 50% compared to NC
[3] Percentage of sLNS solutions that contain at least one infeasible tour (i.e., violate loading constraints)
[4] Representing the maximum cost improvement of rLNS versus sLNS_ExPost

Scenario 2 The data structure with 1 out of 4 segments ordered shows a higher impact on the loading feasibility of tours. For 25 customers, the results indicate that up to 35% of the solutions are infeasible and, for 100 customers, the value reaches 100%. This increase in infeasible solutions impacts both runtime and solution deviation from the sLNS_ExPost. The larger the number of feasibility checks required, the higher the runtime, with some instances requiring more than one hour to be solved. On the other hand, this increase in runtime is accompanied by better solutions. The

average cost deviation increases to 3% (maximum 8%) for the 25-customer cases. The average cost deviation for the 100-customer cases is still small (below 2%), but reaches improvements of up to 10.57%.

Scenario 3 It is not surprising that in our last scenario with two segments the impact of loading constraints is smaller compared to the other scenarios. The reduction in available segments also reduces the options for the compartment setup on a vehicle and therefore also for loading at the DC. In the tests with 25 customers, this setting results in feasible solutions for all instances. However, when analyzing the 100-customer cases, 30% of the instances were found to be infeasible, with up to 3% of cost improvement with the rLNS. These results show that we cannot neglect loading constraints even with only two segments.

Summary In our tests we compare the performance of the rLNS and the sLNS_ExPost for different large data settings. This enabled the analysis of solution quality, runtime efficiency and impact of loading constraints. As shown in different scenarios, the rLNS performs well in terms of solution quality. The approach is able to improve the solutions of the sLNS_ExPost, achieving maximum improvements between 3 and 10%. However, the improvements achieved by the rLNS come with greater computational effort. The average runtime of the rLNS can increase on average to 11 minutes (with a maximum of more than one hour), whereas the sLNS_ExPost has a maximum runtime of 4 minutes. The main driver for these differences in solution quality and runtime is the frequency of infeasible solutions within the search. With an increasing number of infeasible solutions, the rLNS provides a higher solution improvement, but also consumes more computational time. The scenarios considered show that the number of available segments together with the order structure of customers is the most influential driver for loading issues. Other factors such as the geographic positioning of customers and costs do not have an impact on runtime and solution quality. With a larger number of segments, the number of infeasible tours increases. If only two segments are available, fewer solutions will contain infeasible tours. However, the occurrence of infeasible tours increases for four segments, especially if customers only order 1 out of 4 segments, with up to 100% of the problems having an infeasible solution.

5.4 Case study

To conclude the numerical experiments, we applied our solution approach to a case study with a major European grocery retailer. For this analysis, we use data from one specific DC. The data set includes an example in which 4 different segments are available for delivery during a week. As before, the distribution is carried out by identical vehicles with a capacity of 33 TU. The longest distance from the DC to a customer is about 300 km. The considered DC supplies around 100 customers spread in an area of 54.000 km^2. The example week comprises seven delivery days and more than 2,000 customer orders. The complete information regarding our case study is presented in Table 8.

Table 8 Case data regarding a week of delivery

Day	# of orders	Ø order size	Std. dev. order size	Ø number of segments per customer
1	261	6	6	2.7
2	330	10	8	3.4
3	353	10	8	3.7
4	335	10	8	3.5
5	357	11	9	3.7
6	338	11	9	3.5
7	202	11	5	2.1

By solving the 2L-MCVRP with the rLNS and the sLNS_ExPost for each delivery day, we obtain seven different instances. The summary of our findings is displayed in Table 9. In all but the last day, infeasible tours have been found with the sLNS. This is due to the small number of segments per customer (on average only 2 out of 4 segments, see Table 8). On day 7, most of the customers only receive two of the segments. Hence, it is similar to a 2 out of 2 structure, as all customers receive the same two segments.

The improvement ratio of the rLNS is similar to the results for Scenario 1 in Table 7 (100-customer case). The average cost deviation between the solution approaches is below 1%, with the highest improvement reaching 4%. The average and maximum runtimes are higher for the large real problems (Table 9) compared to the simulated problems (Table 7). The sLNS_ExPost provided solutions in less than 1 minute. As previously mentioned, the rLNS logic uses the solution of the sLNS_ExPost as a starting point for the rLNS search (see Fig. 8). Therefore, retailers can use the sLNS_ExPost to generate tours with feasible loading and decide if they want to continue with the rLNS search, knowing that it requires a higher computational effort. Furthermore, the sLNS solution cost for the MCVRP can be compared to one of the sLNS_ExPost, i.e., decide whether the computational effort of the rLNS is worthwhile after repairing the infeasibilities, and depending on the cost increase. Most of the infeasible tours

Table 9 Summary of case study results

Day	Average cost deviation (%)	Maximum improvement (%)	rLNS runtime (min)	
			Average	Maximum
1	−0.77	−4.46	5.29	50.50
2	−0.15	−0.94	1.06	3.51
3	−0.31	−2.78	3.71	18.51
4	−0.86	−4.09	2.31	11.34
5	−0.33	−1.70	3.85	18.12
6	−0.52	−1.85	6.58	20.53
7	–	–	–	–

found could be repaired by exchanging one or two orders. However, it is important to note that even though the number of orders to be moved can be reduced, the impact of rearranging them instead of changing the tour can be high. For instance, if a truck is unloaded in a street instead of inside a shipping gate with controlled temperature, the unloading and reloading of the blocking orders could lead to degradation of the products. Altogether, our case study shows that loading constraints are a relevant issue in retail practice and their consideration in the distribution plan can reduce the costs by up to 4.46%. Additionally, we show the appearance of loading issues on almost all delivery days.

6 Conclusion

The focus of this work was to define and examine the problem of loading constraints that occurs in grocery distribution with MCVs and to develop a solution approach capable of solving the arising MCVRP variant. We therefore presented a detailed problem description to highlight the loading issues of flexible MCVs. We showed that even in the simplest problems, unloading issues can arise when delivering goods to a customer, if the corresponding loading of a vehicle is not taken into account during the loading process. The loading of MCVs has never been studied in the literature. Therefore, we developed a packing problem called multi-compartment packing problem that defines how the vehicle should be loaded in order to respect the loading constraints, if such is possible.

We introduced an extension for the MCVRP with flexible compartments, the 2L-MCVRP, that extends both the literature on MCVRP and two-dimensional loading. For the 2L-MCVRP, we proposed a B&C algorithm to exactly solve the problem. The solution approach makes use of the packing model developed to solve the MCVs loading problem and adds cuts to infeasible tours. We also adapted the LNS framework proposed by Hübner and Ostermeier (2018) to consider loading constraints. For this, the packing problem introduced is used to check the loading feasibility of tours and a modified packing model was created to repair the infeasible tours and thus guide the search for feasible solutions.

The numerical experiments showed that i) loading constraints matter even for small instances, ii) feasible loading can often be achieved by only minor changes to the routing solution and therefore with limited additional costs, and iii) the number of infeasible solutions goes up as the problem sizes increase, especially when a heterogeneous mix of segments is ordered. The numerical experiments also showed that there is a small deviation between the average results achieved with the rLNS and the ex post check of loading constraints (sLNS_ExPost), and that the former requires greater computational effort. For the most complex problems, the sLNS_ExPost approach can therefore be used to generate good feasible solutions in a short running time. Nevertheless, if the solution obtained results in a high cost increase, the rLNS approach can be used to improve it.

The MCVRP variant with flexible compartments is still a very novel research area. The literature is very limited as most publications address fixed compartment sizes. As a consequence, there are various possibilities for extending variants of the MCVRP

with loading constraints. For a start, the packing problem proposed could be adapted to other types of MCVs with less flexibility. Following classical VRP formulations, the MCVRP with loading constraints could also be extended to account for backhauls or pickup and delivery problems. Additionally, there is a lack of literature concerning MCVRPs across multiple periods. The delivery with MCVs might impact delivery frequencies and therefore also delivery patterns (Holzapfel et al. 2016). Usually, delivery patterns are defined by retailers for their stores. Using multiple compartments opens up new possibilities for defining the corresponding delivery patterns. Lastly, the scope of the research could be extended to include the impact on store operations or inventory costs (c.f., Gaur and Fisher 2004; Hübner and Ostermeier 2017).

Acknowledgements The research of the first author is funded by the German Ministry of Education and Research and the Hanns Seidel Foundation. The initialization of this international research cooperation was financially supported by the Bavarian Research Alliance. Further, this work is financed by the ERDF—European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation—COMPETE 2020 Programme within project POCI-01-0145-FEDER-006961, and by National Funds through the FCT—Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) as part of project UID/EEA/50014/2013. The second author was also supported by Grant SFRH/BD/102013/2014 from FCT.

References

Attanasio A, Fuduli A, Ghiani G, Triki C (2007) Integrated shipment dispatching and packing problems: a case study. J Math Model Algorithms 6(1):77–85

Avella P, Boccia M, Sforza A (2004) Solving a fuel delivery problem by heuristic and exact approaches. Eur J Oper Res 152(1):170–179

Bortfeldt A, Wäscher G (2013) Constraints in container loading a state-of-the-art review. Eur J Oper Res 229(1):1–20

Clarke G, Wright JW (1964) Scheduling of vehicles from a central depot to a number of delivery points. Oper Res 12(4):568–581

Côté J-F, Gendreau M, Potvin J-Y (2014) An exact algorithm for the two-dimensional orthogonal packing problem with unloading constraints. Oper Res 62(5):1126–1141

Côté J-F, Guastaroba G, Speranza MG (2017) The value of integrating loading and routing. Eur J Oper Res 257(1):89–105

Derigs U, Gottlieb J, Kalkoff J, Piesche M, Rothlauf F, Vogel U (2011) Vehicle routing with compartments: applications, modelling and heuristics. OR Spectr 33:885–914

Dueck G (1993) New optimization heuristics. J Comput Phys 104(1):86–92

Fuellerer G, Doerner KF, Hartl RF, Iori M (2009) Ant colony optimization for the two-dimensional loading vehicle routing problem. Comput Oper Res 36(3):655–673

Gaur V, Fisher M (2004) A periodic inventory routing problem at a supermarket chain. Oper Res 52(6):813–822

Gendreau M, Iori M, Laporte G, Martello S (2008) A tabu search heuristic for the vehicle routing problem with two-dimensional loading constraints. Networks 51(1):4–18

Golden BL, Raghavan S, Wasil EA (2008) The vehicle routing problem: latest advances and new challenges. operations research/computer science interfaces series. Springer, Berlin

Henke T, Speranza MG, Wäscher G (2015) The multi-compartment vehicle routing problem with flexible compartment sizes. Eur J Oper Res 246(3):730–743

Henke T, Speranza MG, Wäscher G (2017) A branch-and-cut algorithm for the multi-compartment vehicle routing problem with flexible compartment sizes. Working Paper No. 04/2017, Otto-von-Guericke-University Magdeburg

Holzapfel A, Hübner A, Kuhn H, Sternbeck M (2016) Delivery pattern and transportation planning in grocery retailing. Eur J Oper Res 252:54–68

Hübner A, Ostermeier M (2017) Compartmentalized trucks cut the cost of grocery distribution. Supply Chain Management Review
Hübner A, Ostermeier M (2018) A multi-compartment vehicle routing problem with loading and unloading costs. Transp Sci. https://doi.org/10.1287/trsc.2017.0775
Iori M, Martello S (2010) Routing problems with loading constraints. TOP 18(1):4–27
Iori M, Salazar-González J-J, Vigo D (2007) An exact approach for the vehicle routing problem with two-dimensional loading constraints. Transp Sci 41(2):253–264
Klingler R, Hübner A, Kempcke T (2016) End-to-end supply chain management in grocery retailing. European Retail Institute, Cologne, Germany
Koch H, Henke T, Wäscher G (2016) A genetic algorithm for the multi-compartment vehicle routing problem with flexible compartment sizes. Working Paper No. 04/2016, Otto-von-Guericke-University Magdeburg
Muyldermans L, Pang G (2010) On the benefits of co-collection: experiments with a multi-compartment vehicle routing algorithm. Eur J Oper Res 206(1):93–103
Ostermeier M, Hübner A (2018) Vehicle selection for a multi-compartment vehicle routing problem. Eur J Oper Res. https://doi.org/10.1016/j.ejor.2018.01.059
Pollaris H, Braekers K, Caris A, Janssens GK, Limbourg S (2014) Vehicle routing problems with loading constraints: state-of-the-art and future directions. OR Spectr 37(2):297–330
Pollaris H, Braekers K, Caris A, Janssens GK, Limbourg S (2016) Capacitated vehicle routing problem with sequence-based pallet loading and axle weight constraints. EURO J Transp Logist 5(2):231–255
Ropke S, Pisinger D (2006) An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. Transp Sci 40(4):455–472
Shaw P (1997) A new local search algorithm providing high quality solutions to vehicle routing problems. APES Group, Dept of Computer Science, University of Strathclyde, Glasgow, Scotland, UK
Solomon M (1983) Vehicle routing and scheduling with time window constraints: Models and algorithms. Technical report, College of Business Admin., Northeastern University, USA
Toth P, Vigo D (2014) Vehicle Routing: Problems, Methods, and Applications, 2nd edn. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics
Wäscher G, Haußner H, Schumann H (2007) An improved typology of cutting and packing problems. Eur J Oper Res 183(3):1109–1130
Zachariadis EE, Tarantilis CD, Kiranoudis CT (2009) A guided tabu search for the vehicle routing problem with two-dimensional loading constraints. Eur J Oper Res 195(3):729–743
Zachariadis EE, Tarantilis CD, Kiranoudis CT (2013) Integrated distribution and loading planning via a compact metaheuristic algorithm. Eur J Oper Res 228(1):56–71

Affiliations

Manuel Ostermeier[2] · Sara Martins[1] · Pedro Amorim[1] · Alexander Hübner[2]

✉ Alexander Hübner
alexander.huebner@tum.de

Manuel Ostermeier
manuel.ostermeier@tum.de

Sara Martins
sara.martins@fe.up.pt

Pedro Amorim
pamorim@fe.up.pt

[1] INESC TEC, Faculty of Engineering, University of Porto, Rua Dr. Roberto Frias, s/n, 4600-001 Porto, Portugal

[2] Technical University of Munich, Campus Straubing, Supply and Value Chain Management, Schulgasse 22, 94315 Straubing, Germany