Sebastian Reil, Andreas Bortfeldt, Lars Mönch

Heuristics for vehicle routing problems with backhauls, time windows, and 3D loading constraints

Mathematik und Informatik

Informatik-Berichte
374 - 10/2016

INFORMATIK
BERICHTE

374 – 10/2016

HEURISTICS FOR VEHICLE ROUTING PROBLEMS WITH BACKHAULS, TIME WINDOWS, AND 3D LOADING CONSTRAINTS

Sebastian Reil, Andreas Bortfeldt, Lars Mönch

FernUniversität in Hagen

Fakultät für Mathematik und Informatik
D-58084 Hagen

HEURISTICS FOR VEHICLE ROUTING PROBLEMS WITH BACKHAULS, TIME WINDOWS, AND 3D LOADING CONSTRAINTS

Sebastian Reil[1], Andreas Bortfeldt[2], Lars Mönch[1*]

[1]**Department of Mathematics and Computer Science**

University of Hagen

58097 Hagen, Germany

[2]**Department of Management Science**

Otto von Guericke University

39106 Magdeburg, Germany

Abstract

In this paper, we consider vehicle routing problems with backhauls and time windows (VRPBTW). Different backhaul variants are studied, namely clustered backhauls (CB), mixed linehauls and backhauls, and variants with simultaneous delivery and pickup and with divisible delivery and pickup. Three dimensional loading constraints are assumed. A two-phase approach following the principle packing first, routing second is proposed. In the first phase, the packing of goods is carried out by solving a 3D strip packing problem for each customer using tabu search. The resulting VRPTW instance is solved in the second phase using first a multi-start evolutionary strategy to minimize the number of vehicles while again tabu search is applied to minimize the total travel distance. We show that the various backhaul types can be incorporated into this framework. For the backhaul variants different from CB, unloading and reloading efforts are taken into account. Moreover, side loading and a separation of the loading space into separate compartments for goods of linehaul and backhaul customers are proposed. Computational results for benchmark instances and new randomly generated problem instances are presented that demonstrate that the heuristics determine high-quality solutions in a short amount of computing time. The unloading and reloading strategies outperform the strategies based on two separate compartments.

Keywords: Transportation, Packing, Vehicle routing with backhauls and time windows, Three dimensional loading constraints, Integrated routing and packing problem

1 Introduction

Vehicle routing problems (VRPs) are important in many real-world applications. Recently, there is a trend to consider more real-world constraints in problem formulations (cf. Toth and Vigo 2014). This leads to the notation of rich VRPs (cf. Lahyani *et al.* 2015 and Caceres-Cruz *et al.* 2015). Important attributes of rich VRPs are backhaul customers, i.e., goods have to be picked up at the customer locations and have to be transported to the depot (cf. Caceres-Cruz *et al.* 2015). Another important

[*] Corresponding author, E-mail: Lars.Moench@fernuni-hagen.de

class of attributes is given by loading constraints, i.e., more complex loading strategies for the boxes with the customer goods are considered in addition to the fairly simple weight or volume constraints of the vehicles in conventional VRPs (cf. Lahyani *et al.* 2015 and Pollaris *et al.* 2015). Time windows are important in many real-world applications. This class of constraints is extensively considered in the literature (cf., for instance, Bräysy and Gendreau 2005a, 2005b).

VRPs with backhaul and time window constraints are discussed to some extent in the literature (cf., for instance, Reimann and Ulrich, 2006, Parragh *et al.* 2008, Küçükoğlu and Öztürk 2015 amongst others). However, this is not true for the combination of backhauls and loading constraints. We are only aware of the paper by Bortfeldt *et al.* (2015) where a VRP with clustered backhauls and three dimensional loading constraints is studied and the paper by Pinto *et al.* (2015) where a VRP with mixed linehauls and backhauls and two dimensional loading constraints is investigated. Time windows are not considered in these two papers. More generally, the combination of 3D loading constraints and time windows is only rarely discussed in the literature. Therefore, there is a need to consider VRPs where backhauls, loading constraints, and time windows are combined. For instance, global moving companies have to deal with such problems because they have to deliver house moving items to the customers. At the same time they might collect packaging material and bring it back to the depot. Time windows are also important in this application scenario.

In the present paper, we extend the approach proposed by Bortfeldt and Homberger (2013) for the VRPTW with 3D loading constraints (3L-VRPTW) by considering various types of backhauls. It turns out that the packing first, routing second (P1R2) heuristic from Bortfeldt and Homberger (2013) can be applied for situations with backhauls too. The P1R2 heuristic is based on the idea that in the packing stage the boxes of each customer are packed in a separate segment of the loading space by solving a 3D strip packing problem (3D-SPP) for each customer. A loading length arises for the boxes of each customer. In the second stage, the corresponding routing problem is solved where the sum of the loading lengths of the customers that are assigned to a single vehicle can not exceed the loading space length of the vehicle. In the present paper, we consider variants where we take into account the unloading and reloading effort caused by mixed sequences of linehaul and backhaul customers. We study also situations where the loading space of each vehicle is either divided into two vertical or horizontal compartments of the same size, namely one for boxes of linehaul and one for boxes of backhaul customers. Moreover, we investigate side loading in addition to rear-loading strategies.

The paper is organized as follows. The problem is described in Section 2. This includes a discussion of related work. The proposed heuristics are presented in Section 3. The results of computational experiments are described in Section 4. Finally, conclusions and future research directions are discussed in Section 5.

2 Problem Formulation and Related Work

We start by formulating the problem in Subsection 2.1. We then discuss related work in Subsection 2.2.

2.1 Problem Setting

In this subsection, we describe the considered problems in detail. We look at the following four VRPB variants in the present paper that differ in the way how backhaul customers are dealt with:

1. **VRP with Clustered Backhauls (VRPCB)**: Each customer is either a linehaul or a backhaul customer. In each route, linehaul customers are visited before backhaul customers if any.
2. **VRP with Mixed Linehauls and Backhauls (VRPMB)**: Each customer is either a linehaul or a backhaul customer. However, in contrast to the VRPCB, linehaul and backhaul customers are allowed in an arbitrary, i.e. mixed, sequence.
3. **VRP with Simultaneous Delivery and Pickup (VRPSDP)**: Each customer is at the same time linehaul and backhaul customer. Only a single visit of each customer is allowed.
4. **VRP with Divisible Delivery and Pickup (VRPDDP)**: Each customer is at the same time linehaul and backhaul customer. However, in contrast to the VRPSDP, delivery and pickup at a customer location can be performed in a single visit or in two separate visits. Note that the vehicles used for the two visits of the same customer might be different in the present paper whereas the same vehicle is sometimes assumed for the two visits in the literature (cf., for instance, Irnich *et al.* 2014).

Next, we describe the assumptions made in the present paper.

- A vehicle fleet is considered that consists of an unlimited number of vehicles with an identical rectangular loading space with length L, width W, and height H. Each vehicle moves at a constant speed of 1 length unit/ 1 time unit. We differentiate between rear-loaded vehicles that have only a single compartment and rear-loaded vehicles with a single compartment for linehaul customers and with a single compartment for backhaul customers, respectively (cf. Wassan and Nagy 2014 for possible vehicle designs in different VRPB settings). The two compartments have the same size. They are obtained by a horizontal or a vertical partition of the loading space. We refer to the first situation as double-deck loading, whereas the latter situation is called widthwise load partition. Double-deck loading and widthwise load compartments are depicted in Figure 1.

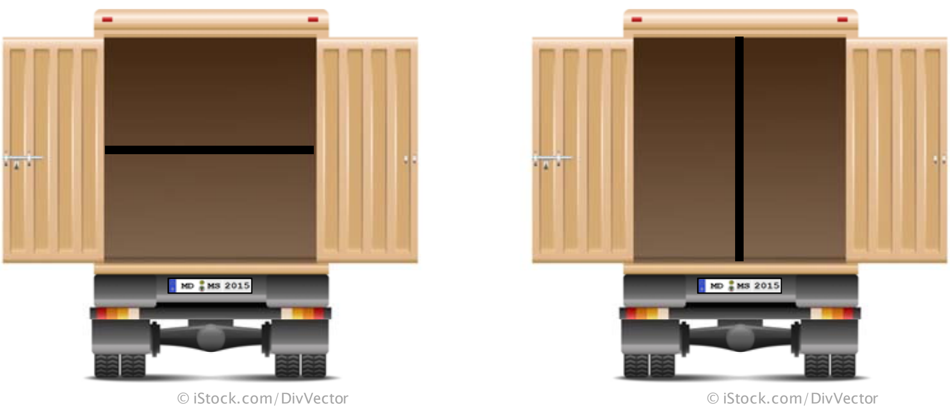

Figure 1: Double-deck loading and widthwise load compartments

We will also consider the case where a side loading of the vehicles is performed (cf. Wade and Salhi 2002). In this situation, only a single compartment is required. The loading of a vehicle with linehaul boxes ends at the front, i.e., the boxes of the linehaul customer to be served last on the route are next to the cabin. At the same time, backhaul customers are loaded starting from the rear, i.e., the boxes of the first backhaul customer are located next to the door of the loading space of the vehicle.

- A set of $n+1$ nodes, denoted by $V := \{0,1,\ldots,n\}$ is used to model the depot and the customer sites. The node 0 refers to the single depot, while the n customers are represented by the nodes $1,\ldots,n$. Let E denote the set of all undirected edges (i,j) that connect all pairs of nodes from V. The resulting graph is $G := (V,E)$. A distance $c_{ij} > 0,\ 0 \le i < j \le n,$ is assigned to each edge (i,j).
- On the one hand, linehaul customer i has to be delivered with a set of m_i boxes $I_{ik}, k = 1,\ldots,m_i$. Each box is a rectangular packing piece. The boxes for linehaul customers are initially located at the depot. On the other hand, backhaul customer j is equipped with m_j boxes $I_{jk}, k = 1,\ldots,m_j$, that are initially located at the corresponding customer location. The boxes associated with backhaul customers have to be picked up at the customer location and have to be transported to the depot. Furthermore, we assume that each box $I_{lk}, l = 1,\ldots,n, k = 1,\ldots,m_l$, has a length l_{lk}, a width w_{lk}, and a height h_{lk}. Note that these settings lead to a volume of s_{lk} for box I_{lk}.
- The loading space of each vehicle is embedded into the first octant of a Cartesian coordinate system in such a manner that the length, the width, and the height of the loading space are parallel to the three axes of the coordinate system. The placement of a box I_{lk} is fully described by the coordinate triplet (x_{lk}, y_{lk}, z_{lk}) that represents the corner of the box that is closest to the origin of the coordinate system. The spatial orientation of a box is described by a one-to-one mapping of the three box dimensions and the three coordinate directions. An orientation index o_{lk} indicates which of the possible spatial orientations is selected for box I_{lk}. A packing plan P for a loading space consists in general of several box placements and is feasible if the following three constraints hold:

  (P1) Each placed box lies completely within the loading space. This means that the common scalar capacity of a vehicle in conventional VRPs is replaced by a 3D rectangular loading space.

  (P2) Any two boxes that are placed in the same truck loading space do not overlap.

  (P3) Each placed box lies parallel to the surface areas of the loading space.

  A loading space with placed boxes that are feasible is exemplified in Figure 2.

Next, we discuss the route-related constraints that are common for all the four VRPB variants:

(R1) Each route starts and ends at the depot.

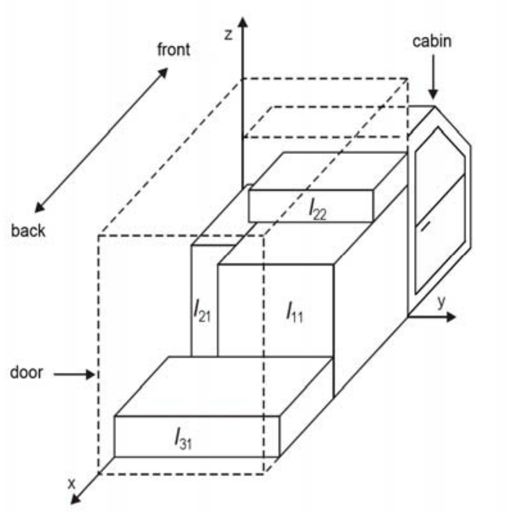

Figure 2: Example loading space with placed boxes

(R2) Each customer is visited exactly once in one route to deliver goods (if any) and each customer is visited once in one route to pick up goods (if any). Hence, in the VRPDDPTW variant a customer can be visited twice in one route or once in two routes or once in one route. In the remaining VRPB variants each customer is visited once in one route throughout.

(R3) Each vehicle performs exactly one trip, i.e., the number of routes is equal to the number of vehicles.

(R4) A delivery time and/or pickup time window $[e_i, l_i]$ with $e_i < l_i$ and a service time $s_i \geq 0$ is associated with each linehaul/backhaul customer, respectively. Note that the regular service time can be increased if unloading and reloading efforts are necessary. On the one hand, each vehicle has to wait until e_i at the location of customer i to begin with the delivery/pickup service, respectively. On the other hand, it is not allowed that the vehicle arrives later than l_i at the location of customer i. There is an additional time window $[e_0, l_0]$ for the depot that limits the total route duration. Note that for the VRPDDPTW two time windows are associated with each customer since up to two visits are possible for each customer. The corresponding time windows for customer i associated with delivery and pickup activities are denoted by $[e_i^L, l_i^L]$ and $[e_i^B, l_i^B]$, respectively. Note that this differentiation is not required for the remaining three VRPB variants.

The following additional loading constraints have to be ensured:

(L1) Weight constraint: Each box I_{ik} has a weight $d_{ik} > 0$. For each point of time, the total weight of all boxes cannot be larger than the maximum load weight D that is the same for all vehicles.

(L2) Orientation constraint: Up to five spatial orientations are forbidden for certain boxes. Often one or two box dimensions are excluded as the height dimension, while horizontal 90° rotations are allowed (height constraint). If the height dimension is given for all boxes, this constraint is called this-way-up constraint.

(L3) Support constraint: A certain percentage a of the base area of a box has to be supported by other boxes if the box is not placed on the floor of the loading space. When we denote by b and c the horizontal dimensions of the box above the floor, respectively, a portion of the base area of size abc has to be placed on other boxes.

(L4) Stacking constraint: Boxes are classified into fragile and non-fragile boxes. When a box is fragile only other fragile boxes might be placed on top of it, while both fragile and non-fragile boxes might be stacked on non-fragile boxes.

(L5) LIFO constraint for the 3L-VRPCBTW and two-compartment situations assuming rear-loaded vehicles:

Linehaul part: If customer i is visited it has to be ensured that it is possible to unload all of his boxes exclusively using movements parallel to the longitudinal axis of the loading space or compartment. Therefore, it is not allowed to place any box belonging to another customer that is visited later than customer i over a box of customer i or between a box of customer i and the rear.

Backhaul part: If customer i is visited it has to be ensured that it is possible to load all of his boxes exclusively using movements parallel to the longitudinal axis of the loading space or compartment. Therefore, it is not allowed to place any box belonging to another customer that is visited before customer i over a box of customer i or between a box of customer i and the rear.

A feasible solution of a 3L-VRPB instance fulfills the constraints (P1)-(P3), (R1)-(R5), (L1)-(L4) and the constraints that are related to the visit of backhaul customers. Depending on the problem variant and the properties of the fleet, constraint (L5) must be additionally ensured. Given a 3L-VRPB instance, we are interested in determining a feasible solution that minimizes the number of vehicles (nv) as the primary objective, while minimizing the total travel distance (ttd) serves as a secondary objective. The four considered problem classes are abbreviated by 3L-VRPCBTW, 3L-VRPMBTW, 3L-VRPSDPTW, and 3L-VRPDDPTW, respectively. The considered problem classes in combination with the possible loading strategies are summarized in Table 1.

Table 1: Considered problem variants

Loading strategy	**Single compartment**		**Two compartments**	
	Rear-loaded	**Side loading**	**Double-deck**	**Widthwise**
3L-VRPB variant				
3L-VRPTW	X	-	-	-
3L-VRPCBTW	X	-	-	-
3L-VRPMBTW	X	X	X	X
3L-VRPSDPTW	X	X	X	X
3L-VRPDDPTW	X	X	X	X

For the sake of completeness, we also perform computational experiments for the 3L-VRPB variants without time windows and for the different VRPBTW variants.

2.2 Discussion of Related Work

We discuss related work with respect to 3L-CVRP and 3L-VRPTW as well as with respect to 3L-VRPB variants. Gendreau *et al.* (2006) introduce the 3L-CVRP. Most of the constraints described in Subsection 2.1 are already included in this problem formulation. A two-phase tabu search algorithm is taken. The outer tabu search determines routes, while the inner tabu search solves a 3D-SPP for a given customer sequence. Various heuristics are proposed for the 3L-CVRP (cf. Pollaris *et al.* 2015 for a recent survey). For instance, an ant colony optimization approach is discussed by Fuellerer *et al.* (2010), whereas Ruan *et al.* (2013) design a bee algorithm for the 3L-CVRP. Tabu search hybridized with guided local search is presented by Tarantilis *et al.* (2009). Some of the best performing heuristics are based on tabu search-type algorithms. Here, we refer for instance to Wisniewski *et al.* (2011), Bortfeldt (2012), Zhu *et al.* (2012), and Tao and Wang (2015).

The 3L-VRPTW is only rarely discussed in the literature. We are only aware of the following three papers. Moura (2008) presents a multi-objective genetic algorithm for the 3L-VRPTW, while Moura and Oliveira (2009) propose different constructive heuristics for the same problem. The method proposed by Bortfeldt and Homberger (2013) consists of an evolutionary strategy and two tabu search procedures. A local search heuristic for the pallet packing VRPTW, considered as a variant of the 3L-VRPTW, is discussed by Zachariades *et al.* (2012). The goods of the customers are first packed on identical pallets that are then loaded onto vehicles. The proposed heuristic does not take into account the LIFO condition.

Next, we look at papers that deal with 3L-VRPB variants and related problems with loading constraints. The 3L-VRPCB has only been tackled by Bortfeldt *et al.* (2015). They propose a Large Neighborhood Search (LNS) and a Variable Neighborhood Search (VNS) scheme. The VRPMB with two dimensional loading constraints (2L-VRPMB) is discussed by Pinto *et al.* (2015). An insertion heuristic is proposed. Zachariades *et al.* (2016) describe a local search approach for the 2L-VRPSDP and the 2L-VRPCB. In addition, a bi-directional 2L-VRP is considered where linehaul and backhaul customers are served by different routes. Likewise, the VRP with pickup and delivery (PD) and 2D or 3D loading constraints is only researched in three papers. Malapert *et al.* (2008) deal with the 2L-VRPPD. Bartók and Imreh (2011) describe a heuristic for solving the 3L-VRPPD. However, it neglects the important LIFO constraint. Männel and Bortfeldt (2016) discuss several 3L-VRPPD variants. Hybrid approaches based on LNS and tree search heuristics are proposed. We are not aware of any paper that deals with the 3L-VRP, backhauls, and time windows at the same time.

The most pertinent research for the present paper is Bortfeldt and Homberger (2013) with the above mentioned P1R2 approach. We extend this approach to allow for considering various types of backhaul customers. In addition, we look at different strategies to deal with unloading and reloading efforts that are a result of the simultaneous occurrence of linehaul and backhaul customers. Finally, we propose new benchmark instances for the four considered 3L-VRPBTW classes.

3 Heuristic Approaches

In this section, we start by discussing the overall framework of the P1R2 scheme in Subsection 3.1. Modifications of the original P1R2 heuristic are also presented in Subsection 3.1. We discuss how the framework can be tailored to the four 3L-VRPB variants in Subsection 3.2. The resulting heuristic is called Backhaul P1R2 (BP1R2) in the rest of this paper.

3.1 Overall Framework

3.1.1 Two-stage Decomposition Heuristic

For the sake of completeness, we start by recalling the main ingredients of the P1R2 heuristic for the 3L-VRPTW. The main idea of the P1R2 heuristic consists in decomposing the solution process into two stages. The first stage is devoted to packing, while the second stage deals with routing.

We start by discussing the packing stage. Here, the boxes of each customer are packed in a separate segment of the corresponding loading space. A 3D-SPP instance for customer i, $i=1,\dots,n$, is fully defined by the width W, the height H, and the set of boxes associated with the customer. The (minimal) loading length ll_i and the packing plans for the boxes of each customer i are stored. Loading-related constraints have to be ensured when the 3D-SPP instance is solved.

In the second stage, routes are calculated for the VRPTW instance that is derived from the corresponding 3L-VRPTW instance. It is important to note that only those routes R are feasible that fulfill the loading length constraint $\sum_{i \in R} ll_i \le L$. We note that this constraint for the routes plays the role of the capacity constraint in the derived VRPTW instance. In addition, other conventional VRPTW constraints have to be fulfilled to obtain feasible routes. Customer combinations are an additional feature of the approach. A customer combination is a set of customers with low utilized segments whose boxes are grouped together in one segment to increase the segment utilization and to reduce the loading length. When the second stage is completed, a solution is given by taking the routes R and arranging the corresponding packing plans.

3.1.2 Integrated Algorithms

The 3D-SPP subproblems resulting from the first stage are solved using the tabu search algorithm proposed by Bortfeldt and Gehring (1999). This tabu search algorithm still provides high-quality solutions (cf. Bortfeldt and Mack 2007). We refer to Bortfeldt and Homberger (2013) for a more detailed algorithmic description of the first stage of the P1R2 scheme.

The two-phase approach by Homberger and Gehring (2005) for solving the VRPTW is used in the second stage of the P1R2 scheme. The number of vehicles nv is minimized in the first phase applying an evolutionary strategy while the total travel distance ttd is minimized in the second phase of the approach using a tabu search approach.

The (μ,λ) evolutionary strategy used in the first phase generates offspring using neighborhood structures that are based on the well-known Or-opt, 2-opt*, and 1-interchange operators (cf. Bräysy and Gendreau 2005a for a description of these operators). The three neighborhood structures are randomly selected per descendant. The initial generation is determined using the modified savings heuristic from Homberger and Gehring (2005). A lexicographical evaluation function is applied that guides the search towards a direction that reduces the number of vehicles. A heuristic that removes customers from the route with the smallest number of customers is another important ingredient of the evolutionary strategy. The evolutionary strategy terminates when a given maximum computing time or a lower bound for the number of vehicles is reached. The first lower bound is given as the ratio of the sum of the minimum load lengths of all customers and the length of the loading space L. A similar lower bound is considered that is based on the weight constraint (L1).

The tabu search heuristic of the second phase is again based on the three neighborhood structures of the first phase. They are randomly chosen per iteration. The tabu list is used to store the link between nodes, while a solution is set tabu if at least one of its links can be found in the tabu list. The number of vehicles obtained from the first phase is not increased in the second phase where we aim for reducing the ttd value. The tabu search scheme is able to produce high-quality solutions (cf. Nagata *et al.* 2010), at least with respect to minimizing the number of vehicles. We refer to Bortfeldt and Homberger (2013) for more details of the second stage algorithm of P1R2.

3.1.3 Improvement of P1R2

We propose a fairly straightforward modification of P1R2. However, it improves the performance of the heuristic to a large extent as we will see in Section 4 when we present the results of computational experiments. The first modification is based on the observation that the (μ,λ) evolutionary strategy and the tabu search algorithm often improve the results considerably at the beginning of the search process but after a while the improvement process stagnates. Therefore, we perform restarts of the second stage of the P1R2 approach after a certain amount of time. Let nr be the number of restarting iterations. The maximum computing time per instance ct is equally distributed among the different iterations by giving each restarting iteration an amount of computing time that is equal to ct / nr. The evolutionary strategy is initialized in iteration $iter \geq 2$ with the best solution found in previous iterations. The proposed restarting strategy is also applied within the BP1R2 approach.

3.2 Tailoring the Framework for the Different Backhaul Variants

3.2.1 Modification of the First Stage

Modifications are required because of the different 3L-VRPB variants and the possibility to have more than one compartment, i.e., boxes of each customer are packed in a separate segment of the corresponding compartment of the loading space. Up to two 3D-SPP instances, namely one for linehaul and one for backhaul boxes, have to be solved for each customer depending on the considered 3L-VRPB variant. Note that either the width W or the height H of the loading space is divided in half if two compartments are available. The loading lengths and the packing plans for linehaul boxes (if any) and for backhaul boxes (if any) of each customer are stored. We denote the loading length for each customer i by ll_i if the customer has either linehaul or backhaul boxes, i.e. in the case of the 3L-VRPCBTW and 3L-VRPMBTW. If customer i has linehaul and backhaul boxes at the same time, i.e. for the 3L-VRPSDPTW and the 3L-VRPDDPTW, we denote the corresponding loading length by ll_i^L and ll_i^B, respectively. Note that the constraints (L1)-(L4) have to be ensured by the first stage. The modified first stage is summarized in the following algorithm outline in pseudo code:

BP1R2 (First Stage)(IN: customer set, OUT: packing plans, loading lengths**)**

FOR each customer DO

IF (customer is either a linehaul or a backhaul customer) DO

- solve the 3D-SPP instance associated with the customer
- store the resulting packing plan and the loading length ll_i

ELSE

- solve a separate 3D-SPP instance for linehaul and backhaul boxes for the customer
- store the resulting packing plans and the loading lengths ll_i^L and ll_i^B

ENDIF

ENDFOR

END.

3.2.2 Modifications of the Second Stage

Modifications of the second stage are required because of backhaul customers and because of two compartments, namely the loading length constraint

(LL1) $$\sum_{i \in R} ll_i \le L \tag{1}$$

for each route R of an 3L-VRPTW instance has to be changed to deal with backhauls. We start by discussing the situation that the loading space is not divided into two compartments. In the case of the 3L-VRPCBTW, we have to ensure that

(LL2) $$\max\left\{ \sum_{i \in R \cap LC} ll_i , \sum_{i \in R \cap BC} ll_i , \right\} \le L , \tag{2}$$

where we denote the set of linehaul customers by LC and the set of backhaul customers by BC. Next, we discuss the 3L-VRPMBTW. Here, we have to formulate the loading length constraint for each customer i of a route R as follows:

(LL3) $$\sum_{j \in L(R,i)} ll_j + \sum_{k \in B(R,i)} ll_k \le L , \tag{3}$$

where $L(R,i)$ is the set of linehaul customers from R that are not unloaded until or at the location of customer i. Similar, $B(R,i)$ is the set of backhaul customers from R that are loaded before or at the location of customer i. In the case of the depot 0, constraints (LL3) reads as (LL2). Note that it is enough to check (LL3) for the depot and for such backhaul customers, where at least one successor is a linehaul customer. We continue by considering the 3L-VRPSDPTW case. In this situation, constraint (LL3) has to be fulfilled for each customer i of a route R. However, in contrast to the 3L-VRPMBTW case $j \in L(R,i) \cap B(R,i)$ is possible. Finally, constraint (LL3) has to be fulfilled in the 3L-VRPDDP case too. However, in contrast to the previous cases, the customers have to be replaced by pairs of customers and route positions since up to two visits are possible for each customer.

Next, we discuss the two-compartment situation. For the 3L-VRPMBTW, constraint (LL2) is correct since each customer is either linehaul or backhaul. For the 3L-VRPSDPTW and 3L-VRPDDPTW variants we have to ensure the following constraint that is similar to (LL2):

(LL4) $$\max\left\{\sum_{i\in R} ll_i^L, \sum_{i\in R} ll_i^B\right\} \le L. \quad (4)$$

The side loading strategy can be considered as a rear-loading strategy. Therefore, the constraints (LL2) and (LL3) have to be enforced for the related 3L-VRPBTW variants. Moreover, the constraints (R1)-(R5) have to be ensured. Note that customer combinations (see Subsection 3.1.1) are not considered in BP1R2 in contrast to the 3L-VRPTW case since it is likely that a grouping of linehaul and backhaul boxes in a single segment leads to additional unloading and reloading efforts.

Both the (μ,λ) evolutionary strategy and the tabu seach algorithm randomly choose a neighborhood structure (see Subsection 3.1.2) for each move. A new solution is determined from a corresponding neighborhood of the incumbent solution. Only feasible moves will be considered in this situation. These feasibility checks are different for the four 3L-VRPBTW variants and will be described in separate subsections below. Note that 3L-VRPBTW variant-specific feasibility checks will also occur in the modified savings heuristic.

3.2.3 Modifications of the Second Stage for 3L-VRPCBTW

Linehaul and backhaul customers are clustered in the 3L-VRPCBTW. In addition, all linehaul customers have to be delivered before any backhaul customer on each route. Therefore, we introduce the binary indicator

$$ct_i := \begin{cases} 0, \text{if customer } i \text{ is a linehaul customer} \\ 1, \text{ otherwise} \end{cases} \quad (5)$$

for each customer i to model these precedence constraints. Note that a similar idea is used in Vidal *et al.* (2014) to consider VRPCB-type problems by using a very large artificial distance between backhaul and linehaul customers. Therefore, the precedence constraints between linehaul and backhaul customers will be ensured.

For an arbitrary route $R := (0,\ldots,i,j,\ldots,0)$ with customers i and j that are visited in a consecutive manner we have to ensure that

$$ct_i \le ct_j \quad (6)$$

is valid whenever a move is applied in the evolutionary strategy or in the tabu search algorithm. In addition, constraints (6) have to be checked whenever subroutes are merged in the modified savings algorithm. Note that the resulting checks can be performed in $O(1)$ time since only the affected customer nodes have to be considered.

We also have to change the lower bound for the number of vehicles that is used in the evolutionary strategy as a stopping criterion. The following expression is used:

$$nv_{min} = \left\lceil \max\left\{\frac{1}{L}\sum_{i\in LC} ll_i, \frac{1}{L}\sum_{i\in BC} ll_i, \frac{1}{D}\sum_{i\in LC} d_i, \frac{1}{D}\sum_{i\in BC} d_i\right\}\right\rceil, \quad (7)$$

where we abbreviate by $d_i = \sum_{k=1}^{m_i} d_{ik}$ the weight of the m_i boxes that belong to customer i. Differentiating between linehaul and backhaul customers is required because they are clustered.

Because of the clustering of linehaul and backhaul customers and the LIFO constraint, unloading and reloading effects do not occur.

3.2.4 Modifications of the Second Stage for 3L-VRPMBTW

In this situation, each customer is either a linehaul or a backhaul customer. No precedence constraints have to be ensured for the visits of the customers. If a rear-loaded vehicle is considered and the loading space is not divided into two compartments, it is likely that unloading and reloading efforts occur since backhaul boxes are placed infront of linehaul boxes during the trip. A simple single-vehicle example is shown in Figure 3. After unloading the boxes that belong to customer 1, the boxes of backhaul customer 2 are loaded infront of the boxes of linehaul customer 4. Hence, the LIFO constraint is not fulfilled anymore.

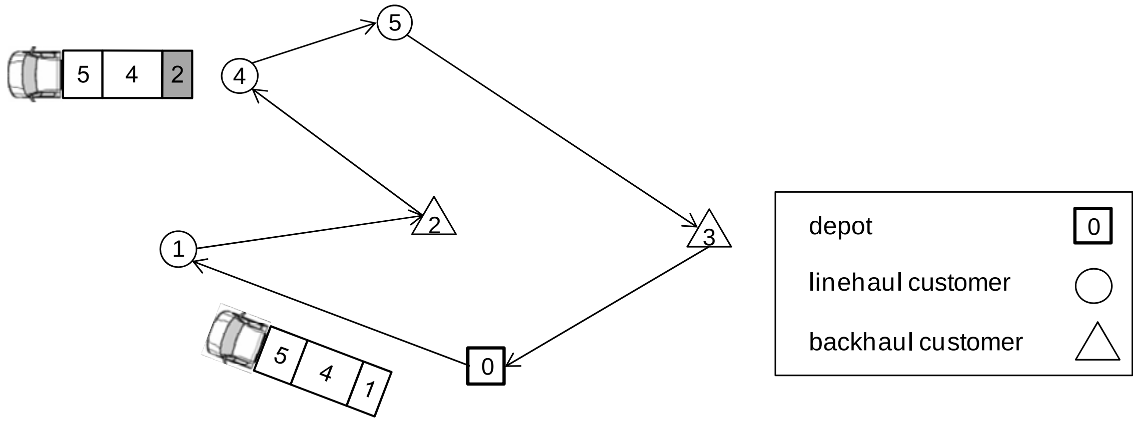

Figure 3: Simple example for unloading and reloading efforts

The additional effort that is caused by unloading and reloading activities is measured based on the weights of the affected boxes, i.e. the boxes that have to be unloaded and reloaded. However, the volume of the boxes or a combination of weight and volume might be taken into account too.

Only backhaul boxes are subject of unloading and reloading activities since we assume rear-loaded vehicles. This approach is reasonable because backhaul items are often empty packaging or waste that are less sensitive with respect to damages. But other unloading and reloading strategies seem to be possible too. The unloading and reloading effort leads to increasing service times. Therefore, we use a factor α that has the dimension of time divided by weight, i.e., $\alpha = 1$ means that one weight unit of reloading effort costs one time unit. The procedure to compute the unloading and reloading effort eff for a given linehaul customer k can be described in pseudo code notation as follows:

CalculateAdditionalLoadingEffort(IN: route R, customer k, factor α, OUT: effort eff **)**

$eff = 0$; $start = k$;

WHILE ($pred(start) \neq 0$) DO

$start = pred(start)$; //determine first customer of route R

ENDWHILE

WHILE ($start \neq k$) DO // at least one predecessor customer has to be backhaul customer to

// cause unloading and reloading efforts

IF ($ct_{start} = 1$) THEN

$eff = eff + 2d_{start}$; // add unloading and reloading effort

ENDIF

$start = succ(start)$;

ENDWHILE

$eff = \alpha\, eff$; // determine additional service time

END.

Here, $pred(v)$ provides the customer that is the direct predecessor of customer v on route R. The direct successor of customer v on route R is denoted by $succ(v)$. Different variants of the CalculateAdditionalLoadingEffort procedure are used in the BP1R2 scheme for 3L-VRPMBTW depending on the neighborhood structures.

The additional unloading and reloading time has to be considered when the time window constraints (R5) are checked if route changes occur due to moves. We show an example for the required feasibility checks when inserting customer k between customer i and j in a route. The two checks to be performed are:

$$dt_i + c_{ik} \leq l_k , \tag{8}$$

$$\max(dt_i + c_{ik}, e_k) + s_k + eff_k + c_{kj} \leq at_j , \tag{9}$$

where dt_i is the earliest possible departure time of the vehicle at the location of customer i, and at_j is the latest possible arrival time at the location of customer j. Constraint (8) ensures that the inserted customer k can be reached from customer i within the time window for customer k whereas constraint (9) models the fact that the latest possible arrival time of the vehicle at the location of customer j is not smaller than the sum of the start of the service, the service time, and the unloading and reloading time at the location of customer k. Note that the latest possible arrival time can be different from the latest point of time for service since the additional unloading and reloading effort at the location of customer j caused by including customer k has to be taken into account.

The unloading and reloading effort can be avoided when separate compartments for linehaul and backhaul boxes are used. However, this approach has the disadvantage that only one half of the overall loading space is reserved for linehaul and backhaul boxes, respectively. This might lead to a poor utilization of the compartments. Increasing nv values might be a consequence.

Side loading of the vehicles is another considered strategy in this paper. It turns out that this approach leads to a loading situation where the segments of the cluster with backhaul boxes are mirrored compared to the single compartment situation. Therefore, the side loading strategy can be considered as a rear-loading strategy with a single compartment where the time needed for unloading and reloading activities is zero. We refer to side loading in the computational results by using the notation $\alpha = 0.0$. A small size example is depicted in Figure 4. It consists of a route that contains eight customers, namely the four linehaul customers $L_1, \ldots, L_4$ and the four backhaul customers $B_1, \ldots, B_4$. The sequence of customer visits is $0 - L_1 - B_1 - L_2 - L_3 - B_2 - L_4 - B_3 - B_4 - 0$.

a) **rear-loaded vehicle**, route 0- L_1 - B_1 - L_2 - L_3 - B_2 - L_4 - B_3 - B_4 –0

depot 0:	front	L_4	L_3	L_2	L_1	empty	Rear
after L_1:	front	L_4	L_3	L_2	empty		Rear
after B_1:	front	L_4	L_3	L_2	B_1	empty	Rear
after L_2:	front	L_4	L_3	B_1	empty		Rear
after L_3:	front	L_4	B_1	empty			Rear
after B_2:	front	L_4	B_1	B_2	empty		Rear
after L_4:	front	B_1	B_2	empty			Rear
after B_3:	front	B_1	B_2	B_3	empty		Rear
after B_4:	front	B_1	B_2	B_3	B_4	empty	Rear

b) **vehicle with side loading**, route 0- L_1 - B_1 - L_2 - L_3 - B_2 - L_4 - B_3 - B_4 –0

depot 0:	front	L_4	L_3	L_2	L_1	empty	Rear
after L_1:	front	L_4	L_3	L_2	empty		Rear
after B_1:	front	L_4	L_3	L_2	empty	B_1	Rear
after L_2:	front	L_4	L_3	Empty		B_1	Rear
after L_3:	front	L_4	Empty			B_1	Rear
after B_2:	front	L_4	Empty		B_2	B_1	Rear
after L_4:	front	Empty			B_2	B_1	Rear
after B_3:	front	empty		B_3	B_2	B_1	Rear
after B_4:	front	empty	B_4	B_3	B_2	B_1	Rear

Figure 4: Example for side loading

The situation for a rear-loaded vehicle is shown in Figure 4 a). The unloading and reloading effort is indicated by dark gray-coloured backhaul customers. Figure 4 b) depicts the situation for a vehicle with side loading. We clearly see the mirrored clusters with backhaul boxes in Figure 4 a) and b).

3.2.5 Modifications of the Second Stage for 3L-VRPSDPTW

Since each customer in a 3L-VRPSDPTW instance has simultaneously linehaul and backhaul boxes, we have to solve a separate 3D-SPP instance for the linehaul and backhaul boxes of each customer, respectively. Therefore, we obtain minimum loading lengths ll_i^L and ll_i^B for each customer. In the case of rear-loaded vehicles with a single loading space, all linehaul boxes that belong to a customer will be delivered before all backhaul boxes of this customer will be picked up. Unloading and reloading effort arises for each customer on a route in the case of rear-loaded vehicles with a single loading space except for the customer that will be visited right after the depot. The corresponding effort can be computed, in principle, based on the CalculateAdditionalLoadingEffort procedure described in Subsection 3.2.4. However, the second while loop in this procedure has to be slightly modified in the sense that the check for backhaul customers is not required anymore and that only the weight of the backhaul boxes has to be considered to determine the unloading and reloading effort.

3.2.6 Modifications of the Second Stage for 3L-VRPDDPTW

In the 3L-VRPDDPTW, again each customer owns simultaneously linehaul and backhaul boxes. However, in contrast to the 3L-VRPSDPTW case, two visits are possible for each customer to deal with the delivery and pickup activities. It is possible to reformulate this problem into a 3L-VRPMBTW by doubling the number of considered customer nodes. This means that for each customer $i, 1 \le i \le n$, of the original instance an additional customer node $i+n$ for backhaul boxes will be considered. The original customer node exclusively represents the linehaul boxes of customer i.

Two cases can be differentiated. In the first case, customer i is not the direct successor or predecessor of customer $i+n$ in a route R. Two visits of the original customer i are necessary in this situation. In the second case, customer i and $i+n$ are adjacent in a route R, i.e., only a single visit of the original customer $i, 1 \le i \le n$ is required. Again we assume that the linehaul boxes belonging to customer i will be delivered before the corresponding backhaul boxes are picked up to avoid unnecessary unloading and reloading efforts. The two cases are again reflected by the procedure CalculateAdditionalLoadingEffort.

4 Computational Experiments

In this section, we start by describing the design of experiments in Subsection 4.1. The parameterization of the heuristics and implementation issues are discussed in Subsection 4.2. We then present the details of the computational experiments in Subsection 4.3.

4.1 Design of Experiments

In this subsection, we describe the problem instance classes used within the computational experiments. We consider the following four main instance classes:

1. Problem instances for the VRPCBTW, the VRPMBTW, and the VRPSDPTW available in the literature are used to assess the performance of the proposed heuristics in the case that no 3D

loading constraints are given. Note that to the best of our knowledge, instances for the VRPDDPTW are not described in the literature. The computational experiments for this instance class are performed to demonstrate that we use a routing heuristic that is able to determine high-quality solutions.

2. Already existing problem instances for the 3L-VRPTW are taken to demonstrate that the modifications of the P1R2 approach are beneficial with respect to the results presented by Bortfeldt and Homberger (2013). Note that only those results from Bortfeldt and Homberger (2013) are used for comparison that are obtained by the P1R2 variant that does not support customer combinations.
3. New problem instances for the 3L-VRPCBTW, 3L-VRPMBTW, 3L-VRPSDPTW, and 3L-VRPDDPTW are generated.
4. For the sake of completeness, problem instances for the 3L-VRPCB, 3L-VRPMB, 3L-VRPSDP, and 3L-VRPDDP are considered. These results complement the results for the Class 3 instances.

The problem instances for Class 1 and Class 2 are taken from Gelinas *et al.* (1995), Wang and Chen (2012), Moura and Oliveira (2009), and Bortfeldt and Homberger (2013). Due to space limitations, we do not describe the instances in detail. The main features are summarized in Table 2.

Table 2: Description of the problem instances (Class 1 and 2)

Class	Origin	Characteristics
1a: VRPCBTW 1b: VRPMBTW	Gelinas *et al.* (1995)	3×15 instances with 25, 50, and 100 customers, respectively
		based on Solomon (1987) instances
1c: VRPSDPTW	Wang and Chen (2012)	56 instances with 100 customers, organized in six different groups
		based on Solomon (1987) instances
2a: 3L-VRPTW	Moura and Oliveira (2009)	46 instances with 25 customers
		based on VRPTW instances of Solomon (1987) (R1 and R2) and container loading problem instances (CLP) of Bischoff and Ratcliff (1995)
		weakly heterogeneous box sets, between 1010 and 1550 boxes per instance
		instance groups: GI-I1: based on R1 instances, tight time windows, GI-I2: based on R2 instances, wide time windows, GII-I1: 1050 boxes, i.e. on average 42 boxes per customer, GII-I2: 1550 boxes, i.e. on average 62 boxes per customer
2b: 3L-VRPTW	Bortfeldt and Hom-berger (2013)	120 instances, number of customers ranging from 100 to 1000
		based on VRPTW instances of Solomon (1987) (R1 and R2) and Gehring and Homberger (1999) (R1 and R2) and CLP instances of Bischoff and Ratcliff (1995) (BR2 and BR4)
		weakly heterogeneous box sets, between 5000 and 50,000 boxes per instance
		instance groups: six groups with 20 instances per group, the groups are given by the pairs (100,5000), (200,10000), (400,20000), (600,30000), (800,40000), (1000,50000) where the first component is the number of customers and the second one the number of boxes, for each group systematic combination of the R1, R2, BR2, and BR4 sets are used to obtain tight and wide time windows and varying number of box types and number of boxes per customer

Because there are no publicly available instances for Class 3 and Class 4, new instances are generated as follows. Seven groups of instances are considered for Class 3. Two instances are randomly taken from each of the four instance groups found in Moura and Oliveira (2009). Therefore, eight customer sets with corresponding boxes are included in the first group. Each customer set contains 25 customers. The number of boxes for the original GI instances is changed from 1050 to 1025 boxes to allow for a reasonable loading space separation, while the number of boxes for the GII instances is changed from 1550 to 1410 boxes for the same reason. For each of the remaining six groups, four instances are randomly selected from the corresponding instance groups proposed by Bortfeldt and Homberger (2013). 24 customer sets with corresponding boxes are obtained by this procedure. The largest problem instances contain 1000 customers. As proposed by Goetschalckx and Jacobs-Blecha (1989), a linehaul portion of 50%, 66%, and 80% is chosen for each customer set. The linehaul and backhaul attributes are randomly assigned to the customers while the correct linehaul portion is ensured. The boxes of each instance are equipped with a random weight measured in tons taken from $U[0.02,0.2]$ where $U[a,b]$ denotes a uniform distribution over the interval $[a,b]$. The maximum load weight of $D = 24$ tons is used for each vehicle. Overall, we obtain 96 new problem instances that can be used for the four 3L-VRPBTW variants. The main features of the Class 3 instances are summarized in Table 3. The instances are available from VRP Instances (2016).

Table 3: Description of the Class 3 problem instances

Class	**Characteristics**
3: 3L-VRPBTW	96 instances for the 3L-VRPCBTW/3L-VRPMBTW and for the 3L-VRPSDPTW/ 3L-VRPDDPTW, the number of customers per instance is ranging from 25 to 1000
	based on the instances of Moura and Oliveira (2009) and Bortfeldt and Homberger (2013)
	weakly heterogeneous box sets, between 1025 and 50,000 boxes per instance
	instance groups: G1: two instances of each instance group of Moura and Oliveira (2009), i.e. a total of eight instances, G2-7: four instances of the six instance groups of Bortfeldt and Homberger (2013), i.e. a total of 24 instances, 100, 200, 400, 600, 800, and 1000 customers are included in the six groups

Note that we have only a single time window for each customer, i.e. $\left[e_i^L, l_i^L\right] = \left[e_i^B, l_i^B\right]$, in our VRPDDPTW problem instances. Class 4 instances are directly obtained from Class 3 instances by neglecting the time windows.

The number of vehicles, the total travel distance, and the total unloading and reloading effort are the performance measures of interest. They are abbreviated by nv, ttd, and tre, respectively. Three independent replications are performed for each heuristic with stochastic ingredients to obtain statistical reasonable results. The average value of the corresponding performance measure values is taken.

4.2 Parameter Setting for the Heuristics and Implementation Issues

All the parameters for the different ingredients of the P1R2 approach except the amount of allowed computing time are chosen as in the original papers (cf. Bortfeld and Gehring 1999 for the 3D-SPP and Homberger and Gehring 2005 for the VRPBTW variants). The maximum amount of computing time per instance is taken from Bortfeldt and Homberger (2013). This means that for each of the two phases of the second stage of the BP1R2 heuristic a computing time of 30 seconds per instance with 25 customers and a computing time of 240 seconds per instance with more than 25 customers are allowed. A single restart of the second stage of the P1R2 approach and of the BP1R2 approach, respectively, is performed. The maximum computing time per instance is equally distributed among the two restarting iterations. The search of the evolutionary strategy is expedited by using the lower bounds based on the load length and the weight (see Subsection 3.1.2) or, if appropriate, the lower bound (7). Moreover, the tabu search algorithm for routing decisions terminates when 500,000 consecutive iterations do not lead to an improvement. Note that up to 3,000,000 tabu search iterations are possible within 240 seconds. Moreover, for all Class 1 instances a maximum computing time of 30 seconds per phase is allowed.

An unlimited amount of computing time is possible for the first stage, i.e. the packing procedure. However, the boxes that belong to a single customer require even in the case of instances with 50,000 boxes less than one second of computing time. This means that the required computing time for packing even for large-size instances with 1000 customers and 50,000 boxes is typically smaller than 300 seconds.

The support constraint factor $a = 100\%$ is selected within the computational experiments. In addition, the factor α to compute the unloading and reloading effort is taken from $\alpha \in \{0,1,2,5\}$. Note that $\alpha = 0$ corresponds to vehicles with side loading, i.e., no unloading and reloading effort occurs in this situation.

The heuristics presented in this paper are coded in the C++ programming language. All the computational experiments are conducted on a computer with Intel Xeon CPU E5-2620 2.0 GHz processor and 32 GB of RAM. Since the proposed heuristic is single threaded only a single core is used.

4.3 Computational Results

4.3.1 Results for Class 1 Instances

We start by presenting results for Class 1a instances. Note that we make sure that each route contains at least one linehaul customer. This restriction is important to ensure that our results are comparable with the results from the literature. The absolute values for the two performance measures are shown in Table 4. In addition, the table contains the results of a comparison with results obtained by Vidal *et al.* (2014) for problem instances with 100 customers using the hybrid genetic algorithm (HGA) from the unified solution framework for multi-attribute vehicle routing problems. The best result out of ten independent runs of the genetic algorithm for each problem instance is shown. We report the deviation that is given by

$$dev(M) := \left(1 - M(\text{HGA})/M(\text{BP1R2})\right)100\%\,, \qquad (10)$$

where $M \in \{nv, ttd\}$ is the performance measure of interest. The argument of M indicates whether the performance measure value is computed for solutions obtained by the HGA or by the BP1R2 approach.

Table 4: Computational results for Class 1a instances

Instances	25 customers		50 customer		100 customers		Vidal *et al.* (2014), 100 customers			
	nv	*ttd*	*nv*	*ttd*	*nv*	*ttd*	*nv*	*ttd*	*dev*(*nv*)	*dev*(*ttd*)
BHR101A	9	644.8	13	1149.7	22	1822.2	22	1818.9	0.00	0.18
BHR101B	10	727.6	16	1255.3	23	1959.5	23	1959.5	0.00	0.00
BHR101C	9	693.5	16	1242.6	24	1939.1	24	1939.1	0.00	0.00
BHR102A	7	564.6	11	989.6	19	1655.1	19	1653.2	0.00	0.11
BHR102B	9	629.1	15	1109.5	22	1753.9	22	1750.7	0.00	0.18
BHR102C	8	586.3	15	1115.2	22	1775.8	22	1775.8	0.00	0.00
BHR103A	5	505.3	8	847.9	15	1387.1	15	1385.4	0.00	0.12
BHR103B	5	546.1	9	906.9	15	1401.5	15	1390.3	0.00	0.80
BHR103C	5	497.0	10	896.3	17	1468.5	17	1456.5	0.00	0.82
BHR104A	4	463.4	6	697.3	11	1111.5	10	1204.6	9.09	-8.37
BHR104B	5	484.6	7	764.0	11	1178.8	11	1154.8	0.00	2.03
BHR104C	4	456.5	7	749.1	11	1232.6	11	1190.2	0.00	3.44
BHR105A	6	574.0	9	1102.9	15	1579.1	15	1560.1	0.00	1.20
BHR105B	7	633.3	10	1057.3	16	1583.3	16	1583.3	0.00	0.00
BHR105C	6	623.6	10	1075.2	16	1720.6	16	1709.7	0.00	0.64
Total	**99**	**8629.7**	**162**	**14958.7**	**259**	**23568.6**	**258**	**23532.0**	**-**	**-**
Average	**-**	**-**	**-**	**-**	**-**	**-**			**0.39**	**0.16**

We see from Table 4 that the HGA outperforms BP1R2 on average only by 0.39% for the nv performance measure and by 0.16% with respect to ttd. Note that the computing time per instance is around four minutes in Vidal *et al.* (2014) whereas the computing time per instance is around 60 seconds for the proposed BP1R2 approach.

Next, we present computational results for Class 1b instances. Again, first absolute values are presented while we also compare the results for problem instances with 100 customers with results described by Küçükoğlu and Öztürk (2015). They propose a heuristic that hybridizes simulated annealing with tabu search. Their objective function is only ttd. The corresponding computational results are presented in Table 5. The deviation from the results of Küçükoğlu and Öztürk (2015) is computed based on

$$dev(M) := \left(M(\text{BP1R2})/M(\text{HH}) - 1\right)100\% \,, \tag{11}$$

where $M(\text{HH})$ is the performance measure value for the hybrid heuristic and where $M \in \{nv, ttd\}$ is the performance measure of interest.

The results in Table 5 demonstrate that the results by Küçükoğlu and Öztürk (2015) are slightly outperformed by the proposed BP1R2 heuristic with respect to ttd. It is remarkable that we obtain some new best results since our primary objective function is nv and not ttd. These results are marked in bold. Note that the computing time of HH per instance is between one and two minutes. Overall, we conclude that the second stage of the BP1R2 heuristic, namely the modified two-phase approach by Homberger and Gehring (2005), leads to high-quality solutions that are competitive with state-of-the-art approaches described in the literature.

Table 5: Computational results for Class 1b instances

Instances	25 customers		50 customers		100 customers		Küçükoğlu and Öztürk (2015), 100 customers			
	nv	*ttd*	*nv*	*ttd*	*nv*	*ttd*	*nv*	*ttd*	*dev*(*nv*)	*dev*(*ttd*)
BHR101A	8	618.3	11	1100.7	19	1653.6	21	1652.2	-9.52	0.09
BHR101B	8	618.3	11	1100.7	19	1662.7	21	1653.2	-9.52	0.57
BHR101C	8	618.3	11	1100.7	19	1666.3	20	1655.6	-5.00	0.64
BHR102A	7	548.1	10	927.5	17.7	1483.9	18	1473.6	-1.85	0.70
BHR102B	7	553.7	10	927.5	17	1491.8	19	1487.0	-10.53	0.33
BHR102C	7	548.4	10	923.7	17	1497.0	18	1478.0	-5.56	1.29
BHR103A	4	473.4	8	784.6	14	**1213.6**	14	1225.5	0.00	**-0.97**
BHR103B	4	473.4	8	784.6	14	**1216.8**	15	1228.1	-6.67	**-0.92**
BHR103C	4	473.4	8	793.8	14	**1213.6**	15	1224.6	-6.67	**-0.90**
BHR104A	4	418.0	6	634.1	10	**994.8**	12	1014.3	-16.67	**-1.92**
BHR104B	4	418.0	6	636.4	10	**995.4**	11	1025.5	-9.09	**-2.93**
BHR104C	4	418.0	6	631.5	10	**992.4**	12	1019.0	-16.67	**-2.61**
BHR105A	5	569.3	8	1014.4	14	1400.6	16	1377.0	-12.50	1.71
BHR105B	5	567.1	8.3	991.7	14	1408.9	15	1379.9	-6.67	2.10
BHR105C	5	556.7	8	1013.2	14	**1379.1**	16	1379.4	-12.50	**-0.02**
Total	**84**	**7872.4**	**129.3**	**13365.1**	**222.7**	**20270.6**	**243**	**20282.2**	**-**	**-**
Average	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-8.37**	**-0.06**

Next, we present computational results for Class 1c instances. The absolute values for the two performance measures are shown in Table 6. Here, the sum of the performance measure values of all instances that form a group is shown. In addition, the table contains the results of a comparison with the results obtained by Wang *et al.* (2015) where a parallel simulated annealing algorithm including an insertion-based heuristic is proposed. Here, again the deviation (11) is used where the hybrid algorithm is replaced by the simulated annealing algorithm. The results by Wang *et al.* (2015) are slightly outperformed by the proposed heuristic with respect to the number of vehicles.

Table 6: Computational results for Class 1c instances

Instances	Wang *et al.* (2015)		BP1R2		Wang *et al.* (2015)	
	nv	*ttd*	*nv*	*ttd*	*dev*(*nv*)	*dev*(*ttd*)
Rdp1xx	151	1202.4	**147.7**	1229.1	**-2.21**	2.22
Rdp2xx	32	972.1	32.7	976.0	2.08	0.40
Cdp1xx	94	960.4	**91.0**	983.0	**-3.19**	2.36
Cdp2xx	24	591.7	24.0	591.8	0.00	0.02
RCdp1xx	98	1415.6	**95.3**	1441.5	**-2.72**	1.83
RCdp2xx	27	1162.3	**26.3**	1204.1	**-2.47**	3.60
Total	**426**	**1055.7**	**417.0**	**1075.6**	**-**	**-**
Average	**-**	**-**	**-**	**-**	**-2.11**	**1.88**

4.3.2 Results for Class 2 Instances

Computational results for Class 2a instances are shown in Table 7. We report the sum of the corresponding performance measure values over all instances of a group. Moreover, averaged values over all instances of an instance group based on the relative measure (11) are reported if appropriate. Here, the HH approach is replaced by the algorithms used for comparison, i.e. the algorithm from Moura and Oliveira (2009) and from Bortfeldt and Homberger (2013), respectively.

Table 7: Computational results for Class 2a instances

Instances	Moura and Oliveira (2009)		Bortfeldt and Homberger (2013)		BP1R2		Moura and Oliveira (2009)		Bortfeldt and Homberger (2013)	
	nv	*ttd*	*nv*	*ttd*	*nv*	*ttd*	*dev*(*nv*)	*dev*(*ttd*)	*dev*(*nv*)	*dev*(*ttd*)
GI-I1	79	891.2	63	543.1	62	546.5	-21.52	-38.67	-1.59	0.63
GI-I2	54	1715.4	44	551.2	44	527.6	-18.52	-69.24	0.00	-4.28
GII-I1	91	1251.5	75	604.4	75	579.6	-17.58	-53.69	0.00	-4.10
GII-I2	75	2907.9	66	557.2	66	541.1	-12.00	-81.39	0.00	-2.90
Total	**299**	**1664.5**	**248**	**564.4**	**247**	**549.3**	**-**	**-**	**-**	**-**
Average	**-**	**-**	**-**	**-**	**-**	**-**	**-17.39**	**-67.00**	**-0.40**	**-2.67**

We see from Table 7 that the results obtained by Moura and Oliveira (2009) and by Bortfeldt and Homberger (2013) are outperformed by the proposed BP1R2 heuristic. It turns out that the restarting strategy described in Subsection 3.1.3 is worthwhile.

Computational results for Class 2b instances are presented in Table 8. We see again that the BP1R2 approach outperforms the original P1R2 approach by Bortfeldt and Homberger (2013) with respect to both performance measures. While the improvement for nv with 0.03% is fairly small, the ttd values are improved by 12.79% on average. The largest improvements are obtained for instances with a large number of customers.

Table 8: Computational results for Class 2b instances

#customers	Bortfeldt and Homberger (2013)		BP1R2		Bortfeldt and Homberger (2013)	
	nv	*ttd*	*nv*	*ttd*	*dev*(*nv*)	*dev*(*ttd*)
100	303	1363.9	299.7	1322.0	-1.10	-3.07
200	564	5445.4	564.0	4965.3	0.00	-8.82
400	1120	13647.0	1120.0	12336.0	0.00	-9.61
600	1680	32587.1	1680.0	28087.7	0.00	-13.81
800	2238	56664.3	2238.0	49275.7	0.00	-13.04
1000	2801	87396.5	2801.3	75908.8	0.01	-13.14
Total	**8706**	**32850.7**	**8703.0**	**28649.2**	**-**	**-**
Average	**-**	**-**	**-**	**-**	**-0.03**	**-12.79**

4.3.3 Results for the Class 3 Instances

We start by presenting results for the 3L-VRPCBTW in Table 9. Both the absolute performance measure values and the corresponding values relative to the initial solution found by the modified savings heuristic are reported using expression (11).

Table 9: Computational results for 3L-VRPCBTW instances

#customers	*nv*	*ttd*	*dev*(*nv*)	*dev*(*ttd*)
25	111.0	545.9	-53.36	-21.58
100	213.7	1949.6	-54.73	-23.97
200	269.7	5306.4	-67.03	-41.96
400	484.3	10341.1	-63.69	-47.26
600	834.3	22327.6	-52.10	-43.08
800	1118.7	52602.6	-63.48	-44.79
1000	1749.3	84560.7	-55.93	-43.83
Total	**4781.0**	**22272.5**	**-**	**-**
Average	**-**	**-**	**-58.91**	**-43.92**

We see from Table 9 that the initial solution is strongly improved by the metaheuristic approach. Improvements up to 67% and 47% for nv and ttd, respectively, are possible.

Next, we discuss computational results for the 3L-VRPMBTW instances. The corresponding results are shown in Table 10.

Table 10: Computational results for 3L-VRPMBTW instances

#customers	$\alpha = 0$			$\alpha = 1$			$\alpha = 2$		
	nv	*ttd*	*tre*	*nv*	*ttd*	*tre*	*nv*	*ttd*	*tre*
25	99.0	481.1	0.0	100.0	479.0	54.4	100.3	491.7	102.2
100	190.0	1669.8	0.0	192.7	1688.1	264.1	194.3	1706.6	501.7
200	235.7	4619.2	0.0	237.0	4670.6	844.2	240.3	4684.3	1583.1
400	431.3	9636.1	0.0	432.7	9701.4	1824.3	434.3	9753.5	3487.8
600	765.7	22368.4	0.0	766.3	22258.5	2614.9	767.0	22173.1	5105.0
800	1009.3	50881.5	0.0	1012.7	51219.5	4017.8	1017.7	51207.4	7893.3
1000	1558.3	83011.7	0.0	1564.3	82799.3	4242.5	1567.7	83461.0	8321.5
Total	**4289.3**	**21643.6**	**0.0**	**4305.7**	**21661.9**	**1739.6**	**4321.7**	**21746.2**	**3387.1**
	$\alpha = 5$			double-deck			widthwise		
	nv	*ttd*	*tre*	*nv*	*ttd*	*tre*	*nv*	*ttd*	*tre*
25	103.7	508.4	181.5	151.0	568.9	0.0	154.0	584.5	0.0
100	197.7	1731.1	1020.7	311.0	1986.6	0.0	317.0	2053.8	0.0
200	246.3	4829.1	3268.9	439.0	7229.4	0.0	474.0	7146.0	0.0
400	443.0	9850.5	7226.0	764.0	14879.3	0.0	806.0	16519.4	0.0
600	770.0	22102.6	11323.1	999.3	29378.0	0.0	1010.3	27393.8	0.0
800	1032.3	50917.5	18538.7	1888.0	83399.9	0.0	1965.3	84575.0	0.0
1000	1584.7	83201.3	19733.6	2818.7	145019.7	0.0	2919.7	144513.4	0.0
Total	**4377.7**	**21706.1**	**7684.3**	**7371.0**	**35378.8**	**0.0**	**7646.3**	**35421.3**	**0.0**

The obtained results are depicted in Figure 5 relative to the results obtained for side loading, i.e. for $\alpha = 0$.

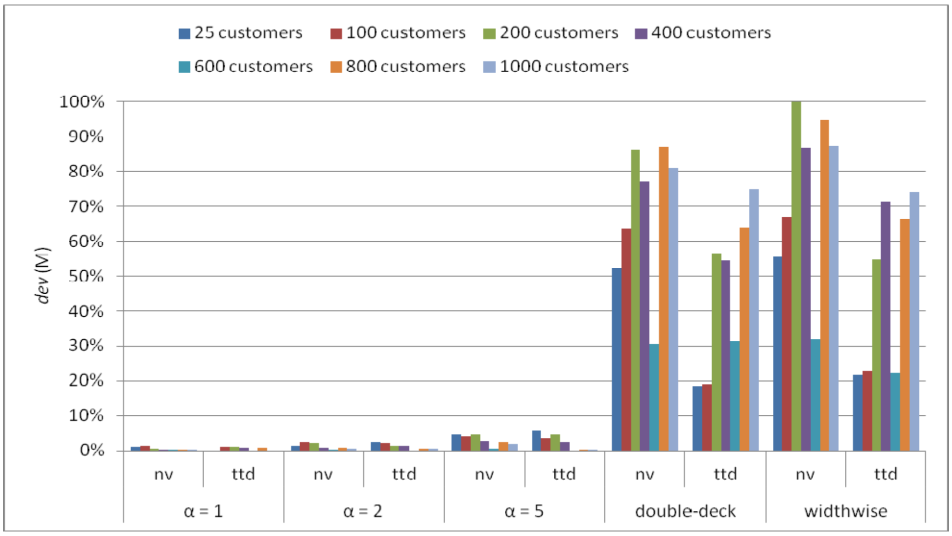

Figure 5: Results for the 3L-VRPMBTW instances relative to the setting $\alpha = 0$

We see from Table 10 and Figure 5 that increasing values for α lead to larger nv values. Compared to the corresponding results for 3L-VRPCBTW instances, we observe that improvements up to 10.28% for nv are obtained. Improvements up to 2.82% are possible for the ttd measure. These improvements are expected since the 3L-VRPMBTW offers much more room for improvement compared to the 3L-VRPCBTW with its precedence constraints. The expensive unloading and reloading activities can be avoided by considering two separate compartments (cf. Figure 1). The obtained results for this setting, however, show that a double-deck loading outperforms the setting with widthwise load compartments. But even in the double-deck loading situation around 68% more vehicles are required compared to the worst case setting with $\alpha = 5$. The *ttd* value is around 63% larger in the case of double-deck loading. Overall, considering separate compartments for linehaul and backhaul boxes seems to be only reasonable if delays due to unloading and reloading efforts have to be avoided.

Results for the 3L-VRPSDPTW instances are shown in Table 11. The corresponding results relative to the results obtained for $\alpha = 0$ are presented in Figure 6.

Table 11: Computational results for 3L-VRPSDPTW instances

#customers	$\alpha = 0$			$\alpha = 1$			$\alpha = 2$		
	nv	*ttd*	*tre*	*nv*	*ttd*	*tre*	*nv*	*ttd*	*tre*
25	94.0	461.6	0.0	96.0	471.1	99.2	102.0	467.2	187.6
100	196.0	1546.1	0.0	204.3	1607.4	453.4	212.0	1667.1	855.1
200	258.0	4231.2	0.0	260.3	4356.1	1420.3	266.0	4514.0	2742.7
400	545.0	10476.8	0.0	545.0	10579.8	2855.0	547.7	10724.9	5559.8
600	574.0	16455.2	0.0	574.7	16599.9	5746.0	579.7	16466.1	11244.6
800	1061.3	48158.4	0.0	1061.3	48723.9	6104.7	1064.0	48888.7	12108.3
1000	1614.0	81396.7	0.0	1614.3	82915.5	6916.5	1617.7	82480.4	13738.9
Total	**4342.3**	**20398.5**	**0.0**	**4356.0**	**20715.6**	**2961.8**	**4389.0**	**20709.4**	**5828.1**
	$\alpha = 5$			double-deck			widthwise		
	nv	*ttd*	*tre*	*nv*	*ttd*	*tre*	*nv*	*ttd*	*tre*
25	110.0	499.0	410.5	150.0	558.1	0.0	154.0	558.1	0.0
100	238.0	1834.9	1875.5	314.0	1948.9	0.0	322.3	1981.3	0.0
200	288.3	4611.2	6083.8	495.7	5906.3	0.0	540.3	6268.4	0.0
400	576.3	10734.9	12449.1	837.3	13458.8	0.0	888.7	13696.4	0.0
600	610.7	17036.7	25677.3	1019.3	23496.3	0.0	1009.7	23639.7	0.0
800	1087.7	49824.5	28976.9	2082.7	64608.9	0.0	2171.7	66848.1	0.0
1000	1643.3	81232.3	32912.1	3115.3	115172.4	0.0	3228.0	118263.3	0.0
Total	**4554.3**	**20784.1**	**13599.5**	**8014.3**	**28213.5**	**0.0**	**8314.7**	**28976.7**	**0.0**

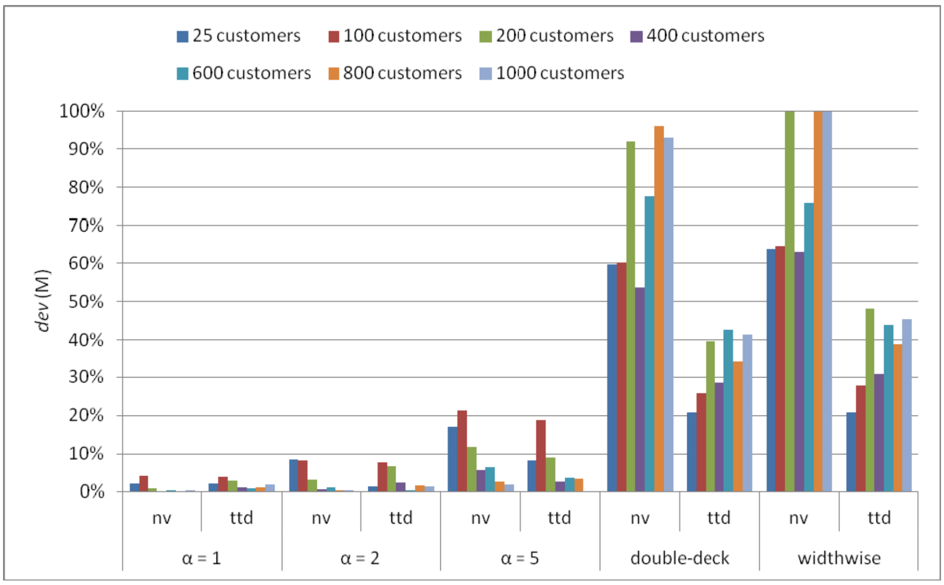

Figure 6: Results for the 3L-VRPSDPTW instances relative to the setting $\alpha = 0$

We see that similar to the results for the 3L-VRPMBTW increasing α values lead to larger nv values. Even the ttd values slightly increase in this situation more often than in the 3L-VRPMBTW case. Similar results as for the 3L-VRPMBTW case are obtained for the setting with two compartments. Again, the setting with widthwise load compartments is outperformed by the setting with double-deck loading. Around 76% more vehicles are needed compared to the setting $\alpha = 5$.

The computational results obtained for the 3L-VRPDDPTW instances are summarized in Table 12. All the results relative to the results for $\alpha = 0$ are shown in Figure 7.

Table 12: Computational results for 3L-VRPDDPTW instances

#customers	$\alpha = 0$			$\alpha = 1$			$\alpha = 2$		
	nv	*ttd*	*tre*	*nv*	*ttd*	*tre*	*nv*	*ttd*	*tre*
25	94.0	461.0	0.0	96.0	470.6	98.6	102.0	466.4	187.9
100	196.0	1545.7	0.0	204.3	1609.9	453.2	212.0	1669.0	854.7
200	258.0	4232.4	0.0	260.3	4349.2	1418.9	266.0	4513.8	2741.5
400	545.0	10469.8	0.0	545.0	10584.2	2845.1	547.7	10703.9	5559.7
600	574.0	16522.9	0.0	575.0	16565.8	5736.4	579.7	16464.3	11234.7
800	1061.3	48119.0	0.0	1061.0	49004.5	6106.2	1065.0	48376.3	12083.2
1000	1614.0	81285.9	0.0	1614.3	82725.5	6923.1	1618.0	81851.4	13747.7
Total	**4342.3**	**20387.2**	**0.0**	**4356.0**	**20722.5**	**2960.0**	**4390.3**	**20564.0**	**5824.7**
	$\alpha = 5$			double-deck			widthwise		
	nv	*ttd*	*tre*	*nv*	*ttd*	*tre*	*nv*	*ttd*	*tre*
25	110.0	496.9	405.0	150.0	556.8	0.0	153.0	576.1	0.0
100	237.7	1841.4	1864.3	314.3	1940.8	0.0	322.3	1977.8	0.0
200	287.0	4674.2	5991.8	493.7	6072.3	0.0	539.7	6318.9	0.0
400	568.3	11692.3	11887.5	837.0	13639.6	0.0	887.7	13952.2	0.0
600	607.3	17272.9	25142.6	1018.0	24045.2	0.0	1008.7	23855.5	0.0
800	1087.3	50081.8	28939.6	2080.7	65829.1	0.0	2171.3	66750.3	0.0
1000	1643.7	81313.7	32959.9	3115.3	115437.5	0.0	3226.0	118386.3	0.0
Total	**4541.3**	**20983.8**	**13449.5**	**8009.0**	**28509.7**	**0.0**	**8308.7**	**29049.1**	**0.0**

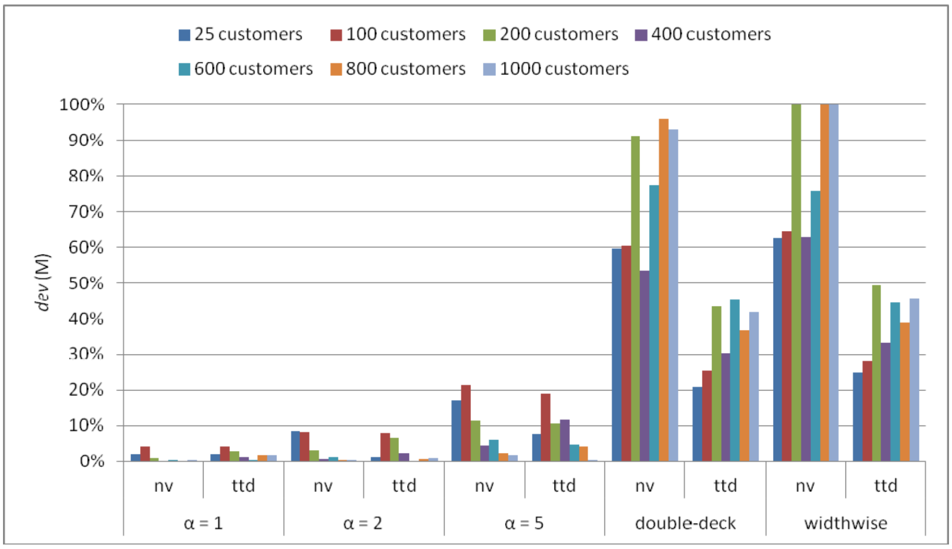

Figure 7: Results for the 3L-VRPDDPTW instances relative to the setting $\alpha = 0$

We see from Table 12 and Table 11 that only minor improvement with respect to the nv and to the ttd measures can be obtained for 3L-VRPDDPTW instances compared to 3L-VRPSDPTW instances. This effect is caused by the fact that only a single time window per customer is considered in the generated problem instances. The behavior of the setting with two compartments is similar to the 3L-VRPMBTW and the 3L-VRPSDPTW case.

4.3.4. Results for the Class 4 Instances

Because we do not have time windows and we consider the ttd measure, the different unloading and reloading strategies do not lead to different results. We present the obtained computational results for the four different 3L-VRPB variants in Table 13. The corresponding columns are labeled by CB, MB, SDP, and DDP to indicate the results for the 3L-VRPCB, 3L-VRPMB, 3L-VRPSDP, and 3L-VRPDDP. Moreover, we present the two deviations

$$dev(M) := \left(M(\text{3L - VRPMB})/M(\text{3L - VRPCB}) - 1\right)100\% \tag{12}$$

and

$$dev(M) := \left(M(\text{3L - VRPDDP})/M(\text{3L - VRPSDP}) - 1\right)100\% \tag{13}$$

to compare related 3L-VRPB variants. The corresponding columns are denoted by MB/CB (%) and DDP/SDP (%), respectively.

Table 13: Computational results for Class 4 instances

# cust.	CB		MB		MB/CB (%)		SDP		DDP		DDP/SDP(%)	
	nv	*ttd*	*nv*	*ttd*	*nv*	*ttd*	*nv*	*ttd*	*nv*	*ttd*	*nv*	*ttd*
25	86.0	444.5	86.0	402.4	0.00	-9.48	84.0	384.8	84.0	384.6	0.00	-0.04
100	151.0	1185.4	151.0	1105.5	0.00	-6.74	165.0	1101.1	165.0	1101.1	0.00	0.00
200	226.0	3821.1	226.0	3563.6	0.00	-6.74	257.0	3350.5	257.0	3350.1	0.00	-0.01
400	422.7	8860.8	422.7	8406.8	0.00	-5.12	545.0	9023.0	544.0	9638.5	-0.18	6.82
600	752.3	21798.0	751.0	22477.0	-0.18	3.11	573.0	15147.5	573.0	15145.9	0.00	-0.01
800	972.7	39280.3	972.0	40306.4	-0.07	2.61	1059.0	35231.5	1056.3	44519.3	-0.25	26.36
1000	1463.0	69441.0	1463.0	69312.2	0.00	-0.19	1601.3	65966.2	1598.3	76076.5	-0.19	15.33
Tot.	**4073.7**	**18159.5**	**4071.7**	**18247.0**	**-**	**-**	**4284.3**	**16323.7**	**4277.7**	**18825.1**	**-**	**-**
Avg.	-	-	-	-	**-0.05**	**0.48**	-	-	-	-	**-0.16**	**15.32**

We see from Table 13 and the tables for 3L-VRPBTW instances that as expected the results for the 3L-VRPB variants outperform the corresponding results for Class 3 instances. This is mainly due to the missing time window constraints for Class 4 instances. We also find in Table 13 that the number of vehicles is slightly lower for the 3L-VRPMB when compared to the 3L-VRPCB. This improvement is obtained by slightly increasing ttd values. Moreover the nv values are slightly improved for the 3L-VRPDDP compared to the 3L-VRPSDP. These reductions are obtained at the expense of increased ttd values for the 3L-VRPDDP instances.

5 Conclusions and Future Research

In this paper, we discussed four different 3L-VRPBTW variants. Modifications of the P1R2 heuristic are proposed that allow for dealing with backhaul customers. Unloading and reloading of boxes are necessary for some of the 3L-VRPBTW variants that increase the service time. In addition to conventional rear-loaded vehicles with a single loading space, we consider vehicles with separate compartments for linehaul and backhaul customers. Side loading of the vehicles is also studied. The performance of the proposed heuristics was investigated by conducting computational experiments with benchmark instances from the literature and with new randomly generated problem instances. It turned out that high-quality solutions can be computed in a short amount of time. The unloading and reloading strategies outperform the strategies based on separate compartments for boxes of linehaul and backhaul customers. The side loading strategy outperforms the remaining strategies.

There are several directions for future research. First of all, we believe that it is worthwhile to relax the assumption that one or two separate segments are assigned to each customer to load the corresponding boxes. For the 3L-VRPCBTW it seems possible to directly extend the LNS and the VNS approach from Bortfeldt *et al.* (2015) to this situation. For the remaining three 3L-VRPBTW variants it seems necessary to work with separate compartments for linehaul and backhaul boxes. Besides the LNS and the VNS approaches, we are interested to extend the unified solution framework by Vidal *et al.* (2014) towards integrated routing and loading decisions. The hybrid genetic algorithm proposed by Vidal *et al.* (2014) is one of the best performing solution approaches for many VRP classes.

Another direction for future research consists in looking at dynamic and stochastic versions of 3L-VRPBTW variants. It seems to be possible to extend the simulation framework proposed by Sprenger and Mönch (2012) to the present situation to allow for an execution of routes in an uncertain environment. Moreover, we expect that the sampling approach proposed by von der Linde and Mönch (2015) can be used for 3L-VRPBTW variants too.

References

Bartók, T.; Imreh, C. (2011): Pickup and Delivery Vehicle Routing with Multidimensional Loading Constraints. *Acta Cybernetica*, 20, 17-33.

Bischoff, E. E.; Ratcliff, M. S. W. (1995): Issues in the Development of Approaches to Container Loading. *Omega*, 23, 377–390.

Bortfeldt, A. (2012): A Hybrid Algorithm for the Capacitated Vehicle Routing Problem with Three-Dimensional Loading Constraints. *Computers & Operations Research*, 39, 2248-2257.

Bortfeldt, A.; Gehring, H. (1999): Two Metaheuristics for Strip Packing Problems. *Proceedings of the Fifth International Conference of the Decision Sciences Institute*, Despotis, D. K., Zopounidis, C. (eds.), 1153-1156.

Bortfeldt, A.; Homberger, J. (2013): Packing First, Routing Second – A Heuristic for the Vehicle Routing and Loading Problem. *Computers & Operations Research*, 40, 873-885.

Bortfeldt, A.; Hahn, T.; Männel, D.; Mönch, L. (2015): Hybrid Algorithms for the Vehicle Routing Problem with Clustered Backhauls and 3D Loading Constraints. *European Journal of Operational Research,* 243, 82-96.

Bortfeldt, A.; Mack, D. (2007): A Heuristic for the Three-dimensional Strip Packing Problem. *European Journal of Operational Research,* 183(3), 1267-1279.

Bräysy, O.; Gendreau, M. (2005a): Vehicle Routing Problem with Time Windows, Part I: Route Construction and Local Search Algorithms. *Transportation Science*, 39(1), 104-118.

Bräysy, O.; Gendreau, M. (2005b): Vehicle Routing Problem with Time Windows, Part II: Metaheuristics. *Transportation Science*, 39(1), 119-139.

Caceres-Cruz, J.; Arias, P.; Guimarans, D.; Riera, D.; Juan, A. A. (2015): Rich Vehicle Routing Problem: Survey. *ACM Computing Surveys*, 47(2).

Fuellerer, G.; Doerner, K. F.; Hartl, R.; Iori, M. (2010): Metaheuristics for Vehicle Routing Problems with Three-dimensional Loading Constraints. *European Journal of Operational Research*, 201, 751–759.

Gelinas, S.; Desrochers, M.; Desrosiers, J.; Solomon, M. M. (1995): A New Branching Strategy for Time Constrained Routing Problems with Application to Backhauling. *Annals of Operations Research*, 61, 91-109.

Gendreau, M.; Iori, M.; Laporte, G.; Martello, S. (2006): A Tabu Search Algorithm for a Routing and Container Loading Problem. *Transportation Science*, 40, 342–350.

Gehring, H.; Homberger, J. (1999): A Parallel Hybrid Evolutionary Metaheuristic for the Vehicle Routing Problem with Time Windows. *Proceedings of EUROGEN99-Short Course on Evolutionary Algorithms in Engineering and Computer Science*, Miettinen, K., Mäkelä, M., Toivanen, J. (eds.) Reports of the Department of Mathematical Information Technology, Series A. Collections, No. A 2/1999, University of Jyväskylä, 57–64.

Goetschalckx, M.; Jacobs-Blecha, C. (1989): The Vehicle Routing Problem with Backhauls. *European Journal of Operational Research*, 42, 39-51.

Homberger, J.; Gehring, H. (2005): A Two-phase Hybrid Metaheuristic for the Vehicle Routing Problem with Time Windows. *European Journal of Operational Research*, 162(1), 220-238.

Irnich, S.; Toth, P.; Vigo, D. (2014): The Family of Vehicle Routing Prolems. Chapter 1 in Toth, P., Vigo, D. (eds.): *Vehicle Routing: Problems, Methods, and Applications.* SIAM Monographs on Discrete Mathematics and Applications, 2nd edition, Philadelphia, 1-33.

Küçükoğlu, I.; Öztürk, N. (2015): An Advanced Hybrid Meta-heuristic Algorithm for the Vehicle Routing Problem with Backhauls and Time Windows. *Computers & Industrial Engineering*, 86, 60-68.

Lahyani, R.; Khemakhem, M.; Semet, F. (2015): Rich Vehicle Routing Problems: From a Taxonomy to a Definition. *European Journal of Operational Research*, 241, 1-14.

Malapert, A.; Guéret, C.; Jussien, N.; Langevin, A.; Rousseau, L.-M. (2008): Two-dimensional Pickup and Delivery Routing Problem with Loading Constraints. *Proceedings of the First CPAIOR Workshop on Bin Packing and Placement Constraints (BPPC'08), Paris, France.*

Männel, D.; Bortfeldt, A. (2016): A Hybrid Algorithm for the Vehicle Routing Problem with Pickup and Delivery and Three-dimensional Loading Constraints. *European Journal of Operational Research*, 254(3), 840-858.

Moura, A. (2008): A Multi-objective Genetic Algorithm for the Vehicle Routing Problem with Time Windows and Loading. Bortfeldt, A., Homberger, J., Kopfer, H., Pankratz, G., Strangmeier, R. (eds.): *Intelligent Decision Support: Current Challenges and Approaches*, Gabler, Wiesbaden, 187-201.

Moura, A.; Olievera, J. F. (2009): An Integrated Approach to the Vehicle Routing and Container Loading Problems. *OR Spectrum*, 31, 775-800.

Nagata, Y.; Bräysy, O.; Dullaert, W. (2010): A Penalty-based Edge Assembly Memetic Algorithm for the Vehicle Routing Problem with Time Windows. *Computers & Operations Research*, 37(4), 724–737.

Parragh, S. N.; Doerner, K. F.; Hartl, R. F. (2008): A Survey on Pickup and Devivery Problems. Part I: Transportation between Customers and Depot. *Journal für Betriebswirtschaft*, 58, 21-51.

Pinto, T.; Alves, C.; de Carvalho, J. V.; Moura, A. (2015): An Insertion Heuristic for the Capacitated Vehicle Routing Problem with Loading Constraints and Mixed Linehauls and Backhauls. *FME Transactions*, 43(4), 311-318.

Pollaris, H.; Braekers, K.; Caris, A.; Janssens, G. K.; Limbourg, S. (2015): Vehicle Routing Problems with Loading Constraints: State-of-the-art and Future Directions. *OR Spectrum*, 37, 297-330.

Reimann, M.; Ulrich, H. (2006): Comparing Backhauling Strategies in Vehicle Routing Using Ant Colony Optimization. *Central European Journal of Operations Research*, 14(2), 105-123.

Ruan, Q.; Zhang, Z.; Miao, L.; Shen, H. (2013): A Hybrid Approach for the Vehicle Routing Problem with Three-dimensional Loading Constraints. *Computers & Operations Research*, 40, 1579-1589.

Solomon, M. M. (1987): Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints. *Operations Research*, 35, 254–265.

Sprenger, R.; Mönch, L. (2012): A Methodology to Solve Large-Scale Cooperative Transportation Planning Problems. *European Journal of Operational Research*, 223, 626–636.

Tao, Y.; Wang, F. (2015): An efficient Tabu Search Approach with Improved Loading Algorithms for the 3L-CVRP. *Computers & Operations Research*, 55, 127-140.

Tarantilis, C. D.; Zachariadis, E. E.; Kiranoudis, C. T. (2009): A Hybrid Metaheuristic Algorithm for the Integrated Vehicle Routing and Three-dimensional Container-loading Problem. *IEEE Transactions on Intelligent Transportation Systems*, 10, 255–271.

Toth, P.; Vigo, D. (eds.) (2014): *Vehicle Routing: Problems, Methods, and Applications*. SIAM Monographs on Discrete Mathematics and Applications, 2nd edition, Philadelphia.

Vidal, T.; Crainic, T. G.; Gendreau, M.; Prins, C. (2014): A Unified Solution Framework for Multi-attribute Vehicle Routing Problems. *European Journal of Operations Research*, 234(3), 658-673.

von der Linde, R.; Mönch, L. (2015): A Sampling Approach to Solve the Vehicle Routing Problem with Time Windows and Stochastic Travel Times. *Proceedings of the 6th IESM Conference*, 740-747.

VRP Instances (2016): http://p2schedgen.fernuni-hagen.de/index.php?id=266. Accessed 2016-/9/16.

Wade, A. C.; Salhi, S. (2002): An Investigation into a New Class of Vehicle Routing Problem with Backhauls. *Omega*, 30, 479-487.

Wang, H.-F.; Chen Y.-Y. (2012): A Genetic Algorithm for the Simultaneous Delivery and Pickup Problems with Time Window. *Computers & Industrial Engineering*, 62(1), 84-95.

Wang, C.; Mu, D.; Zhao, F.; Sutherland, J. W. (2015): A Parallel Simulated Annealing Method for the Vehicle Routing Problem with Simultaneous Pickup-delivery and Time Windows. *Computers & Industrial Engineering*, 83, 111-122.

Wassan, N. A.; Nagy, G. (2014): Vehicle Routing Problems with Deliveries and Pickups: Modeling Issues and Meta-heuristic Solution Approaches. *International Journal of Transportation*, 2(1), 95-110.

Wisniewski, M.; Ritt, M.; Buriol, L. S. (2011): A Tabu Algorithm for the Capacitated Vehicle Routing Problem with Three-dimensional Loading Constraints. *Anais do XLIII Simpósio Brasileiro de Pesquisa Operacional*, 1502-1511.

Zachariadis, E. E.; Tarantilis, C. D.; Kiranoudis, C. T. (2012): The Pallet-Packing Vehicle Routing Problem. *Transportation Science*, 46, 341–358.

Zachariadis, E. E.; Tarantilis, C. D.; Kiranoudis, C. T. (2016): Vehicle Routing Strategies for Pick-up and Delivery Service Under Two Dimensional Loading Constraints. *Operational Research*. Accepted for Publication.

Zhu, W.; Qin, H.; Lim, A.; Wang, L. (2012): A Two-stage Tabu Search Algorithm with Enhanced Packing Heuristics for the 3L-CVRP and M3L-CVRP. *Computers & Operations Research*, 39, 2178-2195.

Verzeichnis der zuletzt erschienenen Informatik-Berichte

[364] Güting, R.H., Behr, T., Düntgen, C.:
Book Chapter: Trajectory Databases; 5/2012

[365] Paul, A., Rettinger, R., Weihrauch, K.:
CCA 2012 Ninth International Conference on Computability and Complexity in Analysis (extended abstracts), 6/2012

[366] Lu, J., Güting, R.H.:
Simple and Efficient Coupling of a Hadoop With a Database Engine, 10/2012

[367] Hoyrup, M., Ko, K., Rettinger, R., Zhong, N.:
CCA 2013 Tenth International Conference on Computability and Complexity in Analysis (extended abstracts), 7/2013

[368] Beierle, C., Kern-Isberner, G.:
4th Workshop on Dynamics of Knowledge and Belief (DKB-2013), 9/2013

[369] Güting, R.H., Valdés, F., Damiani, M.L.:
Symbolic Trajectories, 12/2013

[370] Bortfeldt, A., Hahn, T., Männel, D., Mönch, L.:
Metaheuristics for the Vehicle Routing Problem with Clustered Backhauls and 3D Loading Constraints, 8/2014

[371] Güting, R. H., Nidzwetzki, J. K.:
DISTRIBUTED SECONDO: An extensible highly available and scalable database management system, 5/2016

[372] M. Kulaš:
A practical view on substitutions, 7/2016

[373] Fabio Valdés, Ralf Hartmut Güting:
Index-supported Pattern Matching on Tuples of Time-dependent Values, 7/2016