A hybrid algorithm for the vehicle routing problem with three-dimensional loading constraints and mixed backhauls

Henriette Koch[1] · Maximilian Schlögell[1] · Andreas Bortfeldt[1]

Abstract

In this paper, a variant of the vehicle routing problem with mixed backhauls (VRPMB) is presented, i.e. goods have to be delivered from a central depot to linehaul customers, and, at the same time, goods have to be picked up from backhaul customers and brought to the depot. Both types of customers can be visited in mixed sequences. The goods to be delivered or picked up are three-dimensional (cuboid) items. Hence, in addition to a routing plan, a feasible packing plan for each tour has to be provided considering a number of loading constraints. The resulting problem is the vehicle routing problem with three-dimensional loading constraints and mixed backhauls (3L-VRPMB). The simultaneous transport of linehaul and backhaul items presents a particular challenge of the problem. We consider two different loading variants in order to avoid any reloading during the tour: (i) rear loading with separate linehaul and backhaul sections and (ii) loading at a long side. In order to solve the problem, we propose a hybrid metaheuristic consisting of a reactive tabu search for the routing problem and different packing heuristics for the loading problem. Numerical experiments are reported with benchmark instances from the literature for the one-dimensional VRPMB to examine the performance of the routing algorithm and with newly generated instances for the 3L-VRPMB.

Keywords Vehicle routing · Backhauls · Tabu search · Packing

1 Introduction

In 2010, the average empty running rate—i.e. the share of trucks driving without transporting any goods—in the European Union amounted to 24% de Angelis (2011). This occurs, for example, if vehicles return empty from their deliveries. By incorporating the pickup of goods (*backhauling*) during the tours into the logistics system, empty runs can be reduced which subsequently leads to a reduction in travelled distances, fuel consumption and CO_2 emission. Therefore, vehicle routing problems (VRPs) with backhauls also gain increasing attention in research.

While backhaul problems can be modelled in different variants (cf e.g. Parragh et al. (2008); Irnich et al. (2014)), this paper will be focused on the (VRPMB). In this problem variant, goods have to be either delivered to customers (*line-*

haul) or picked up from them (*backhaul*). The sequence of linehaul and backhaul customers within a tour can be chosen arbitrarily.

Moreover, we aim to provide a more realistic modelling of the transportation of (bulky) goods which are of a size that cannot be neglected to ensure feasibility when planning the routes. Therefore, the transported goods are assumed to be three-dimensional (3D) cuboid items. Each solution of the problem must, thus, be equipped with a feasible packing plan per route. A particular challenge of the problem is to transport linehaul and backhaul items simultaneously on the same vehicle. In order to avoid any reloading during a tour, two different loading approaches are considered: (i) loading from the rear side with horizontal separation of the loading space into a delivery section and a pickup section and (ii) loading from one long side. The side from which items are loaded and unloaded is subsequently called loading side. The resulting problem belongs to the group of VRPs with three-dimensional loading constraints (3LVRPs) which was introduced by Gendreau et al. (2006).

We propose a hybrid algorithm for solving the three-dimensional VRPMB. The underlying routing problem is

✉ Henriette Koch
   henriette.koch@ovgu.de

[1] Department of Management Science,
   Otto-von-Guericke-University Magdeburg,
   39106 Magdeburg, Germany

solved with a reactive tabu search (RTS) based on the approach of Nagy et al. (2013). In order to solve the packing subproblem, different packing heuristics have been implemented which can be chosen alternatively. They were tested and compared concerning their performance.

The remainder of this paper is organized as follows: a detailed problem description is presented in Sect. 2. In Sect. 3, an overview of the relevant literature is given. The proposed RTS and the packing heuristics are described in Sect. 4. In Sect. 5, the experiment setup is described, and the results are presented and analysed. Finally, the paper concludes with a summary and an outlook to future research in Sect. 6.

2 Problem description

Let $G = (N, E)$ be a weighted, directed graph with the node set $N = \{0, 1, \ldots, n\}$, where node 0 represents the depot and the nodes $1, \ldots, n$ represent the n customer locations, and the edge set $E = \{(i, j) : i, j \in N, i \neq j\}$. The customers are divided into l linehaul customers and b backhaul customers, i.e. $N = \{0, 1, \ldots, n\} = \{0, 1, \ldots, l, l + 1, \ldots, l + b\}$. Furthermore, let c_{ij} be the cost corresponding to edge $(i, j) \in E$.

A set $I_i = \{1, \ldots, m_i\}$ of m_i cuboid items (boxes) is assigned to each customer i ($i \in N \setminus \{0\}$) which must be either delivered to them (linehaul) or picked up from them (backhaul). Each item I_{ik} ($i \in N \setminus \{0\}, k \in I_i$) has a known length l_{ik}, width w_{ik}, height h_{ik} and weight d_{ik} and is assigned with a fragility flag f_{ik} indicating whether it is fragile ($f_{ik} = 1$) or not ($f_{ik} = 0$). The entire cargo, given by the set of boxes of all n customers, might be strongly heterogeneous (nearly as many box types as boxes), weakly heterogeneous (many boxes, few box types) or even homogeneous. v_{max} identical vehicles are available with a given weight capacity D and a three-dimensional cuboid loading space of length L, width W and height H. All vehicles can be loaded and unloaded either from the rear or from one long side (see below).

A solution for the problem must contain information about the allocation of customers to routes, the customer sequences of the routes and the corresponding packing plans.

A packing plan P contains placements for one or more items. It is feasible if it fulfils the following conditions: (P1) all items lie entirely within their loading space, (P2) any two items which are placed simultaneously in one loading space must not overlap, and (P3) all items must be placed orthogonally to the loading space edges. Moreover, the following additional packing constraints must be adhered to [cf. Gendreau et al. (2006)]:

(PC1) **Fixed vertical orientation** The items can be rotated by 90° on the horizontal plane, but the height dimensions are fixed.

(PC2) **Vertical stability** Each item must be supported by a given percentage α by the top faces of other items or the container floor (see Sect. 5.1).

(PC3) **Fragility** A non-fragile item cannot be placed on top of a fragile item, whereas fragile items can be placed on top of any other item.

(PC4) **LIFO** The last-in, first-out constraint requires that the items are loaded and unloaded solely by straight movements towards the loading side. Therefore, it must be ensured that the (un-)loading is not blocked by items that are delivered later or have already been picked up.

Let l_1 and l_2 be two linehaul customers and b_1 and b_2 two backhaul customers. Assuming l_1 precedes l_2 in a given route, no item of l_2 must be positioned between the loading side and any item of customer l_1 or above such item. Analogously, if b_1 precedes b_2 in a route, no item of b_1 can be positioned between the loading side and any item of customer b_2 or above such item. Furthermore, if b_1 precedes l_1 in a route, no item of b_1 may be placed between the loading side and any item of l_1 or above such item. In addition, no item of l_1 can be placed between the loading side and any item of b_1 or above such item.

The orientation constraint (PC1) often occurs if technical devices, e.g. household appliances, are to be transported. Constraint (PC2) is also called static stability constraint and prevents boxes from falling down onto the container floor (see Bortfeldt and Wäscher (2013)). By the fragility constraint (PC3), the stacking of boxes is restricted because of their limited load bearing strength. The stability and fragility constraint occur in different variants; see, for example, Paquay et al. (2017, p. 1583).

The LIFO policy (PC4) implies that the reloading of any item during the route is forbidden. Therefore, this constraint is particularly challenging considering that linehaul and backhaul items are transported simultaneously. Two alternative loading approaches are applied here in order to avoid any reloading effort.

In the first variant, double-decker vehicles are used. These vehicles are rear-loaded (the loading side is the rear side), and the loading space is separated horizontally so that two separate compartments are available for each type (linehaul or backhaul). This way, the LIFO constraint must not be considered w.r.t. a mixture of linehaul and backhaul items. It is assumed that both compartments are of the same size. In the
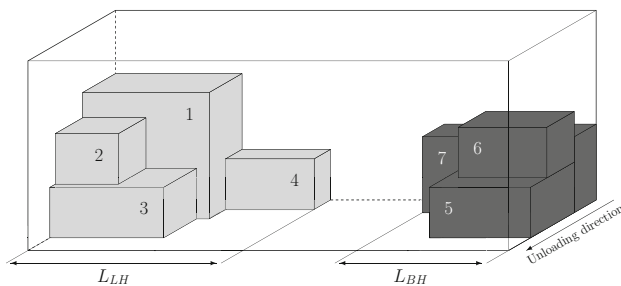
Fig. 1 Side loading

following, this variant will be referred to as loading space partition (LSP).

Secondly, side loading (SL) is applied for which so-called tautliners are used. These vehicles can only be loaded and unloaded from the side (the loading side is one long side). An example is illustrated in Fig. 1. By loading linehaul (light grey) and backhaul (dark grey) items from opposing sides (cabin and rear side), space is created for backhaul items when linehaul items are unloaded. L_{LH} and L_{BH} represent the loading lengths (i.e. maximum front edge) of all linehaul and backhaul items, respectively, which are currently in the loading space. In order to avoid any overlapping, the sum of both lengths must not exceed L. The LIFO constraint must be considered as above ensuring that the (un-)loading is not blocked. In the example in Fig. 1, items 2 and 3 must not be delivered after item 1. Moreover, the constraint must also be considered along the length axis ensuring that the unloading of linehaul items successively creates space for backhaul items. Therefore, item 4 may not be delivered after item 1.

Since we assume uniform vehicles, only one of the loading approaches can be applied for a problem instance, i.e. all vehicles in any generated solution have to be loaded either by the LSP approach or by the SL approach.

A feasible route R is a sequence of locations $(0, i_1, \ldots, i_{n_r}, 0)$ which fulfils the following conditions: (R1) it starts and ends at the depot, (R2) it comprises each customer $i \in R \backslash \{0\}$ exactly once, (R3) the total weight of all items transported simultaneously does not exceed the vehicle weight capacity D, and (R4) a feasible packing plan P_L exists for all linehaul customers in R at the beginning of the tour and a feasible packing plan P_B exists for all backhaul customers at the end of the tour.

Let v be the number of used vehicles in a solution. Assuming each vehicle travels exactly one tour, a solution consists of a set of v triples $(R_t, P_{t,L}, P_{t,B})$ containing a route R_t for each vehicle t $(t = 1, \ldots, v)$ and the corresponding packing plans $P_{t,L}$ and $P_{t,B}$. A solution is feasible if (S1) all routes R_t and packing plans $P_{t,L}, P_{t,B}$ $(t = 1, \ldots, v)$ are feasible, (S2) each packing plan $P_{t,L}$ $(P_{t,B})$ contains all of the respective linehaul (backhaul) items (and no others) of all customers visited in R_t $(t = 1, \ldots, v)$, (S3) each customer $i \in N \backslash \{0\}$

is assigned to exactly one route, and (S4) the number of used vehicles v does not exceed the number of available vehicles v_{max}. Moreover, a feasible solution for the problem with SL approach must also adhere to the restriction (S5) that the linehaul and backhaul items that are transported *simultaneously* at any given moment in a route R_t $(t = 1, \ldots, v)$ do not overlap, i.e. the sum of the lengths L_{LH} and L_{BH} must never exceed L (see above).

A feasible solution is to be found that minimizes the total travel distance (TTD). The problem can be classified as a 3L-VRP with mixed backhauls (3L-VRPMB).

3 Literature review

The 3L-VRPMB has so far only been considered once, namely by Reil et al. Reil et al. (2018). The following literature review is focused on the one-dimensional VRPMB and problem variants of the 3L-VRP. In addition, some relevant studies about the VRP with two-dimensional loading constraints are shortly mentioned.

3.1 Vehicle routing problem with mixed backhauls

The (one-dimensional) VRPMB has been studied intensively in the past decades. Straightforward heuristic solution approaches were primarily used in the beginning. They include, for example, Savings and insertions heuristics (Golden et al. 1985; Casco et al. 1988), or cluster-first-route-second heuristics (Halse 1992). In the recent past, the trend shifted towards the use of metaheuristics. One of the first metaheuristics for the VRPMB was presented by Wade and Salhi (2004) who suggested an ant colony optimization (ACO) approach. A hybrid approach presented in Crispim and Brandão (2005) consists of a tabu search (TS) and a variable neighbourhood descent heuristic. An adaptive large neighbourhood search (ALNS) for a great variety of VRPs with backhauls has been described in Ropke and Pisinger (2006). More recently, a reactive tabu search (RTS) to solve the VRPMB was proposed in Nagy et al. (2013) and serves as the base for the solution approach presented in this paper. In addition, a problem variant is also considered in Nagy et al. (2013) where a mixture of linehaul and backhaul customers in a tour is only allowed if a given percentage of the vehicle capacity is available. Hence, if the percentage equals 100%, the VRP with clustered backhauls (VRPCB) is considered, i.e. all linehaul customers have to be visited before the backhaul customers within a tour. Further recent approaches include ACO (Wassan et al. 2013), adaptive local search (Avci and Topaloglu 2015) and evolutionary algorithms (García-Nájera et al. 2015).

3.2 Vehicle routing problems with loading constraints

The capacitated VRP (CVRP) with three-dimensional loading constraints (3L-CVRP) was first presented by Gendreau et al. (2006), who also introduced the above-mentioned constraints regarding the packing subproblem. Subsequently, the problem was studied by various researchers (e.g. Tarantilis et al. (2009); Fuellerer et al. (2010); Bortfeldt (2012)). The 3L-VRP with time windows was first dealt with by Moura (2008) and Moura and Oliveira (2009). They consider two objective criteria. Namely, the TTD and the number of tours as common in research regarding VRPs with time windows (VRPTW). Furthermore, the maximization of the utilized volume as another optimization objective is considered in Moura (2008). The 3L-CVRP with a heterogeneous vehicle fleet is addressed in Wei et al. (2014).

The routing problem is usually tackled with a metaheuristic approach, e.g., genetic algorithm (Moura 2008; Miao et al. 2012), TS (Gendreau et al. 2006; Tarantilis et al. 2009; Wang et al. 2010; Ma et al. 2011; Wisniewski 2011; Zhu et al. 2012; Tao and Wang 2015) or ACO (Fuellerer et al. 2010). Since solving the packing problem requires comparatively much computing time as the packing procedure is called very frequently, the packing problem is often solved by applying simple construction heuristics, e.g. based on bottom-left and touching area heuristics. More complex packing approaches are, for example, applied by Bortfeldt (2012) (tree search) or Zhang et al. (2015) (local-search-based approach). Furthermore, an exact approach for solving the routing subproblem and a GRASP algorithm for solving the packing problem for the obtained routes is used in Escobar-Falcón et al. (2016).

So far, the underlying VRP was mainly assumed to be a CVRP or VRPTW. Variants with pickup and delivery have not been studied intensely yet. The 3L-VRP with clustered backhauls (3L-VRPCB) is approached in Bortfeldt et al. (2015) where ALNS and variable neighbourhood search are used for the routing problem and a tree search procedure is applied to the packing problem. The pickup and delivery problem with three-dimensional loading constraints is studied in Bartók and Imreh (2011) and Männel and Bortfeldt (2016). Here, goods are transported from a loading location to an unloading location (that are not the depot).

Different variants of 3L-VRPs with backhauls and time windows, among them the 3L-VRPMB, are tackled in Reil et al. (2018). The authors propose a two-phase approach following the principle "packing first, routing second". That is, first for each customer a truck segment is filled by the customer's boxes by means of a TS packing procedure. In the second phase, the remaining routing task is done with an evolutionary strategy and a TS. The heuristic is able to cope with large instances with up to 1000 customers and 50,000 boxes.

In addition, VRPs with backhauls have been studied considering two-dimensional (2D) loading constraints, e.g. Dominguez et al. (2015) (clustered backhauls), Pinto et al. (2015) (mixed backhauls) or Zachariadis et al. (2016) (simultaneous delivery and pickup). In Pinto et al. (2015) and Zachariadis et al. (2016), variants with simultaneous transportation of linehaul and backhaul items are considered where rearrangements are not allowed.

A detailed overview of the literature on VRP with two- and three-dimensional loading constraints is provided in Pollaris et al. (2015). Packing problems, related constraints and algorithms that are partially of relevance for 3L-VRPs are reviewed in Bortfeldt and Wäscher (2013).

4 Hybrid solution approach

Being a generalization of the CVRP, the 3L-VRPMB is also an NP-hard optimization problem (cf. e.g. Toth and Vigo (2014)). In order to find high-quality solutions within reasonable computing time, a metaheuristic framework is applied to solve the routing problem. As in many previous works, the packing problem is tackled with construction heuristics. In the following subsections, both parts of the solution procedure will be described.

4.1 Reactive tabu search

The routing problem is solved with a reactive tabu search (RTS) based on the work of Nagy et al. (2013). The rough outline of the procedure is depicted in Fig. 2. It starts with the initialization of the search (initial solution, tabu list). In each iteration, a neighbour of the current solution s is generated by applying a selected move m_s. Usually, in tabu search algorithms the best non-tabu move is used or a tabu move if it satisfies an aspiration criterion. However, we work with a candidate list CL here, consisting of n_{CL} ($n_{CL} > 1$) moves. In general, the move m_s to be applied to the current solution s is chosen at random from the candidate list. Tabued moves held in the tabu list are expressed in terms of a customer i and a tour t so that i must not be inserted into t for a number of iterations given by the tabu tenure tt.

Reactive elements are included in the tabu list management changing the tabu tenure based on the search progress. In each iteration, a reinitialization of the tabu search can also be triggered. Otherwise, a local optimization procedure is applied to the current solution s after the application of the move m_s. The components of the RTS are described in detail in the following subsections.

```
 1: procedure REACTIVE TABU SEARCH(in: instance data,
    parameters, out: best solution s_best)
 2:    initialize tabu search
 3:    s := s_best := s_init ▷ initialize current solution s and
       best found solution s_best
 4:    while stopping criterion is not met do
 5:       determine move m_s for current solution s
 6:       s := s ⊕ m_s                ▷ realize move m_s
 7:       if f(s) < f(s_best) then
 8:          s_best := s         ▷ update best found solution
 9:       end if
10:       update tabu list TL
11:       if no reinitialization then
12:          apply local optimization to s
13:       end if
14:    end while
15: end procedure
```

Fig. 2 Reactive tabu search

4.1.1 Initialization of the search

Two different construction heuristics are applied alternatively to generate the initial solution in order to test their impact on the performance of the RTS.

On the one hand, we use the modified Sweep heuristic as in Nagy et al. (2013). This heuristic extends the classical Sweep heuristic of Gillett and Miller (1974) by leaving the 20% of the customers that are closest to the depot out of the procedure to form single-stop tours. In doing so, poor quality solutions should be avoided and these customers should be left to the RTS algorithm to find the best-fitting routes [see Wassan et al. (2008); Nagy et al. (2013)].

In addition, the Savings heuristic of Clarke and Wright (1964) is also applied in order to construct initial solutions. In this case, all customers are included into the construction process. Initially, they form single-stop tours and are successively merged according to the Savings criterion until no further merging is possible.

At the very beginning of the solution procedure, the tabu search is initialized with an empty tabu list (TL := ∅), and a tabu tenure $tt := tt_{init}$, $count_{ors} := 0$ and $ma := 0$. The variables $count_{ors}$ and ma serve for the reactive operations (see below).

4.1.2 Neighbourhood structures and moves

The two inter-route move types used in Nagy et al. (2013) are applied here. *Shift* moves remove one customer from one tour and reinsert the customer into another (which can also be an empty tour). *Swap* moves remove two customers from different tours and reinsert them into the tours of the respective other customer. A move consists of one (*Shift*) or two (*Swap*) customer movements. Each customer movement is characterized by a customer to be moved, a source tour, a target tour and a target position. Hence, the swapped customers are

not necessarily inserted into the previous (source) position of the respective other customer but can be inserted into any position. Analogously, any position can be considered for the *Shift* moves. For example, shifting customer i from tour t_1 into tour t_2 at position p_1, and shifting customer i from tour t_1 into tour t_2 at position p_2 are two different moves. This approach is contrary to the original approach of Nagy et al. (2013) who always insert a customer into its best position in the target tour.

Furthermore, in Nagy et al. (2013), the whole neighbourhood is evaluated, i.e. each customer and each target tour are considered for the *Shift* moves and each pair of customers in different tours for the *Swap* moves. In contrast, we aim to omit potentially unpromising moves in order to save computing time while maintaining the solution quality. As relevant criterion, we use the distance $\Delta(t_1, t_2)$ between two tours t_1 and t_2 which is calculated using the minimum and maximum x- and y-values of the coordinates of the customers included in the respective tours:

$$\Delta(t_1, t_2) = \left(\max\left[\min_{i \in R'_{t_1}} x_i, \min_{i \in R'_{t_2}} x_i \right] - \min\left[\max_{i \in R'_{t_1}} x_i, \max_{i \in R'_{t_2}} x_i \right] \right)$$
$$+ \left(\max\left[\min_{i \in R'_{t_1}} y_i, \min_{i \in R'_{t_2}} y_i \right] - \min\left[\max_{i \in R'_{t_1}} y_i, \max_{i \in R'_{t_2}} y_i \right] \right).$$

Here, R'_{t_i} stands for the set of customers served in tour t_i ($i = 1, 2$) excluding the depot ($R'_{t_i} = R_{t_i} \setminus \{0\}$ [see Sect. 2]). Negative distance values indicate two intersecting tours and offer more potential for improvement than pairs of tours that are further away from each other. Only tp_{max} of all tour pairs with the smallest distances $\Delta(t_1, t_2)$ are considered. tp_{max} is a predefined parameter.

4.1.3 Determining a move for the current solution

In each iteration, a move m_s is determined to construct a new solution from the current one according to $s := s \oplus m_s$. In order to determine m_s, in a first step, a candidate list CL is generated as depicted in Fig. 3 (lines 7 to 22).

In the beginning, the set of all moves M for solution s is generated. All of these moves are then examined. In the end, CL contains up to n_{CL} moves m that lead to feasible solutions $s' := s \oplus m$. The feasibility check has the following aspects:

- all tours of a feasible solution s' must not exceed the vehicle weight capacity D and volume capacity V ($V = L \cdot W \cdot H$),
- feasible packing plans $P_{t,L}$ and $P_{t,B}$ must exist for all tours t of s'. This includes (S5) in the case of SL.

CL contains the best non-tabu moves leading to feasible solutions. In addition, a tabu move m that yields a new best

solution $s' := s \oplus m$ with $f(s') < f(s_{\text{best}})$ is also accepted for CL, i.e. aspiration by objective is applied.

When the candidate list CL is completely generated, the move m_s is determined as follows:

(1) If CL is empty (including only dummy moves, see Fig. 3), the tabued move with the shortest remaining tabu tenure is chosen as m_s, i.e. aspiration by default is used.
(2) If CL is not empty and the solution $s' := s \oplus m_{\text{best}}$ (see Fig. 3, line 26) is a new best solution, m_s is set to m_{best}; clearly, the best move m_{best} is the move for which $f(s \oplus m)$ gets minimal.
(3) If CL is not empty and there is no new best solution, the move m_s is selected randomly from CL.

Hence, by not necessarily selecting the best neighbour of s, a diversification mechanism is introduced into the search.

4.1.4 Tabu list management

The tabu list contains all customer movements that are currently tabu, i.e. the information which customers must not be inserted into which tours, as well as the iteration number in which the respective moves are allowed again. The tabu tenure tt determines how long a movement is set tabu. If a *Shift* move removing customer i from tour t was applied in the current iteration, any move that inserts i into t would be tabu for tt iterations. If a *Swap* move was applied, two movements must be set tabu—one for each customer-tour combination affected: if customer i_1 from t_1 was swapped with customer i_2 from tour t_2, then any move inserting i_1 into t_1 *or* i_2 into t_2 is not allowed for the next tt iterations.

Three operations to adapt the search can be carried out within the RTS, namely increasing and decreasing of the tabu tenure tt, and reinitialization of the search. These operations are described in detail in Table 1 (see also Nagy et al. (2013)).

The parameter $GapMax$ is handled as a constant in Nagy et al. (2013). The results could be improved, though, by taking the size of the instance into account. If an instance contains many customers, there are more possibilities for the algorithm to move around a local optimum and to avoid tabued moves. Therefore, we determine this value as $GapMax = \lambda_{gap} \cdot \sqrt{n}$. In addition, a maximum tabu tenure $tt_{\max} = \lambda_{tt} \cdot n$ is considered. λ_{gap} and λ_{tt} are predefined parameters.

4.1.5 Local optimization

At the end of an iteration, two post-optimization procedures are utilized. They are applied only to the tours affected in the current iteration. The first routine is a sequence of intra-route shifts. That is, a customer is moved to another position within the tour if this shift leads to a TTD reduction. This is done iteratively for all customers in the tour until no further improvement is achieved.

In the second routine, the visiting sequence of the customers in a tour is reversed if this reduces the maximum load of a route. In Nagy et al. (2013), the utilization of the capacity is taken into account for this purpose since they consider the one-dimensional case. Here, the maximum loading length needed for linehaul and backhaul items that are simultaneously transported (cf. Fig. 1) is reduced if possible. By this means, an additional customer might be visited in the given route.

4.1.6 Tour number reduction

The heuristics for the construction of the initial solution can lead to solutions with more than $v_{\max}$ tours. In order to guide the search towards solutions that do not violate the maximum tour number constraint, objective function values of solutions with more than $v_{\max}$ tours are penalized. A penalty term $(p \cdot c_{\max} \cdot \max(0, v - v_{\max}))$ is added to the TTD. p is a fixed parameter, $c_{\max}$ is the maximum distance between any two customers $(c_{\max} = \max_{(i,j) \in E} c_{ij})$, and v is the number of vehicles used in the respective solution. The factors $c_{\max}$ and p serve to ensure a sufficiently large penalty.

Furthermore, in connection with the *Shift* move, a customer can only be moved into an empty tour if less than $v_{\max}$ tours are used in the current solution.

4.1.7 Stopping criteria

The algorithm has to run for at least $iter_{\min}$ iterations. However, the search may continue beyond $iter_{\min}$ iterations if the last improvement was less than $iter_{\text{no\_impr}}$ ago. Then, it stops after $iter_{\text{no\_impr}}$ iterations without improvement.

4.2 Packing heuristics

Three different variants of the deepest-bottom-left-fill (DBLF) heuristic have been integrated into the RTS. It was originally developed for 2D packing (Baker et al. 1980; Hopper 2000) and extended to 3D packing (Karabulut and Inceoglu 2005). The DBLF variants are described here for the 3D container loading problem (3DCLP). In the 3D-CLP, a subset of a set of 3D (cuboid) items has to be packed into one container of fixed dimensions such that the filling rate is maximized. Below, we adapt these heuristics to ensure the feasibility of a route w.r.t. the packing subproblem within the 3L-VRPMB.

4.2.1 DBLF heuristics for the 3D container loading problem

The first variant of the DBLF heuristic that we consider is based on the DBLF implementation presented by Karabulut and Inceoglu (2005) (and by Hopper (2000) for the 2D case).

```
 1: procedure DETERMINE_MOVE(in: current solution s, best solution s_best, instance data, parameters, out: move m_s)
 2:     initialize set of all moves M
 3:     m_d := dummy move for initialization of CL with f(s ⊕ m_d) = ∞
 4:     m_worst := m_d                                                                    ▷ initialize worst move in CL
 5:     m_TabuShort := m_d, δ(m_TabuShort) := ∞                              ▷ init. move with shortest remaining tabu tenure δ
 6:     CL is initialized with n_CL dummy moves m_d
 7:     for all moves m ∈ M do
 8:         s' := s ⊕ m                                                                    ▷ generate neighbour s' of s
 9:         if (m is not tabu and f(s') < f(s ⊕ m_worst)
10:             or f(s') < f(s_best) then                                                       ▷ aspiration by objective
11:             if s' is feasible then
12:                 CL := CL ∪ {m}, CL := CL \ {m_worst}                              ▷ current move replaces m_worst
13:                 determine new m_worst
14:             end if
15:         else
16:             if m is tabu then
17:                 if δ(m) < δ(m_TabuShort) then                                                ▷ aspiration by default
18:                     if s' is feasible then m_TabuShort := m end if
19:                 end if
20:             end if
21:         end if
22:     end for
23:     if CL contains only dummy moves then
24:         m_s := m_TabuShort
25:     else
26:         determine best move m_best ∈ CL                                       ▷ f(s ⊕ m_best) ≤ f(s ⊕ m), ∀ m ∈ CL
27:         if f(s ⊕ m_best) < f(s_best) then m_s := m_best
28:         else select m_s ∈ CL, m_s ≠ m_d at random                                     ▷ without dummy moves
29:         end if
30:     end if
31: end procedure
```

Fig. 3 Determination of move m_s

Table 1 RTS updating

RTS operation	Triggering event/condition	Actions
1. Increasing tabu tenure	(i) Currently generated solution $s(it)$ is copy of old solution $s(it\_old)$ and	– tt := min(tt · ϕ_{inc}, tt$_{max}$) (tt is increased by a factor $\phi_{inc} > 1$, but at most up to tt$_{max}$)
	(ii) CopyGap := $it - it\_old <$ GapMax, i.e. a copy was generated after too few iterations.	– $ma := 0.1 CopyGap + 0.9 ma$
2. Decreasing tabu tenure	(i) Currently generated solution $s(it)$ is not a copy of any older solution and	– tt := max(tt · ϕ_{dec}, 1) (tt is decreased by a factor $\phi_{dec} < 1$, but at most down to 1)
	(ii) No. of iterations since last tt change is greater than moving average (ma) of $CopyGap$ over all iterations since last (re-)initialization	
3. Reinitialization of the search	– For each generated solution $s(it)$, $\eta(s)$ is defined as the number of iterations it_old for which $s(it) = s(it\_old)$, i.e. $s(it)$ is a copy of a former solution	– TL := ∅
	– If $\eta(s)$ exceeds the limit η_{max}, the counter for often-repeated solutions count$_{ors}$ is increased by one	– tt, count$_{ors}$, and ma are set back to their initial values (see above)
	– A reinitialization of the search is triggered if count$_{ors}$ exceeds the threshold count$_{ors,max}$	– an old solution is selected at random to be the new initial solution

In the approach of Karabulut and Inceoglu (2005), items are packed according to a given sequence. If an item cannot be placed feasibly in a container, it is skipped. Moreover, the spatial orientation is provisionally assumed to be fixed. The priorities for the placement are to position the items as far as possible to the back, (then) to the bottom and (then) to the left of the loading space. In implementations of the DBL heuristic, the final placement is often found using a sliding technique. On the contrary, the *Fill* method allows to fill gaps by keeping track of all possible placement positions and places each item in the deepest, bottom-most, left-most available position. The different approaches are depicted in Fig. 4. (For the sake of simplicity, the figures in this section show 2D problems). In the case of the *Fill* approach (Fig. 4a), the potential positions are defined by the top-left-back, bottom-right-back, and bottom-left-front corners of already placed items. The positions are sorted based on the deepest-bottom-left priority which is indicated by the numbers 1-5 in the example figure. It is successively tested whether the placement in the respective positions would be feasible. The tests terminate as soon as a feasible position was identified or all positions have been tested. The position that is nearest to the bottom and (then) nearest to the left and where the placement would be feasible is position 2. Hence, the gap could be filled which is not possible by sliding the item from the top-right corner (Fig. 4b). After a successful placement test for a given position, an item is (if possible) further moved starting from that position as far as possible towards the back, the bottom and the left. In the example in Fig. 4a, the final position of the item would be position $2'$.

The second implementation variant extends the placement test of the heuristic presented above and is subsequently called DBLF$^+$. A position where an item cannot be placed feasibly is not considered any further in the DBLF approach. On the contrary, sliding an item into *one* direction is considered during the placement in the variant DBLF$^+$ (regardless of whether a feasible placement is already possible without the sliding). That way, a feasible placement could result from an otherwise infeasible position. The DBLF priority is applied to the sliding, too. That is, first, sliding towards the back, then towards the bottom and then towards the left is tested. If sliding is possible towards one direction, the other ones are not regarded any more. As before, further movements towards the back, bottom and left can be conducted *after* the placement test was successful.

An example is illustrated in Fig. 5. Testing the placement in position 2 according to the DBLF approach would lead to an infeasible placement (Fig. 5b). The item would overlap with other items. Thus, the next position would be tested now. However, considering sliding for position 2 would lead to a placement further to the bottom which is feasible (Fig. 5c).

The third variant is a combination of the original DBLF implementation and DBLF$^+$ in which the sliding of DBLF$^+$

(a) bottom-left-fill (BLF)

(b) bottom-left (BL)
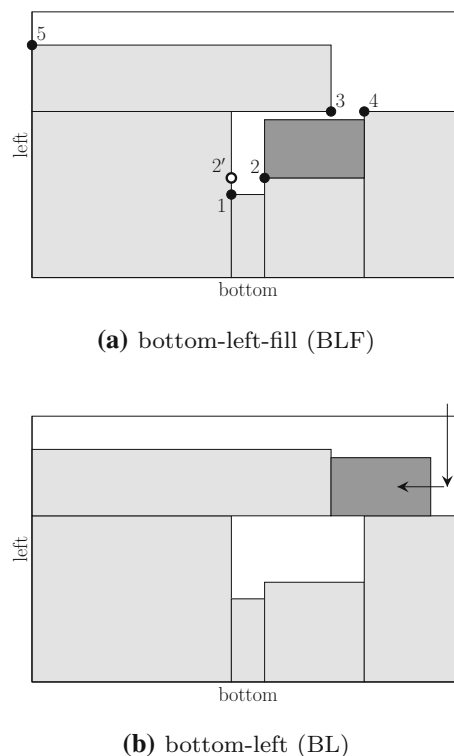
Fig. 4 Comparison between BLF and BL approach

is only used once the original DBLF procedure cannot find a feasible placement for an item. The variant is subsequently called DBLF-Comb and is outlined in Fig. 6. The functions *placement_DBLF* and *placement_DBLF$^+$* represent the respective procedure in which the feasibility of a placement is tested.

Furthermore, variants of the TA heuristic [e.g. Lodi et al. (1999)] were also implemented and tested but led to worse results than the DBLF variants. Therefore, they will not be described in further detail here.

4.2.2 Adaption of the DBLF heuristics to the 3L-VRPMB

In order to apply the three packing heuristics to the packing subproblem described in Sect. 2, the following modifications have been made:

– In the 3L-VRPMB, each item has two possible spatial orientations. Therefore, the following placement rule is applied: if placing an item in a given position with the first orientation fails, the same position is tested again with the second orientation.
– Only those placements are accepted where the geometrical constraints [(P1)–(P3)] as well as the packing constraints *vertical stability*, *fragility* and *LIFO* are satisfied.

(a) Placed items with potential positions

(b) Without consideration of sliding

(c) With consideration of sliding

Fig. 5 Comparison between BLF and BL approach

- In order to facilitate finding a feasible packing plan for a given route, the following sequence of items is chosen: the first items to be placed are the ones of the last (first) customers to be visited in the case of lineha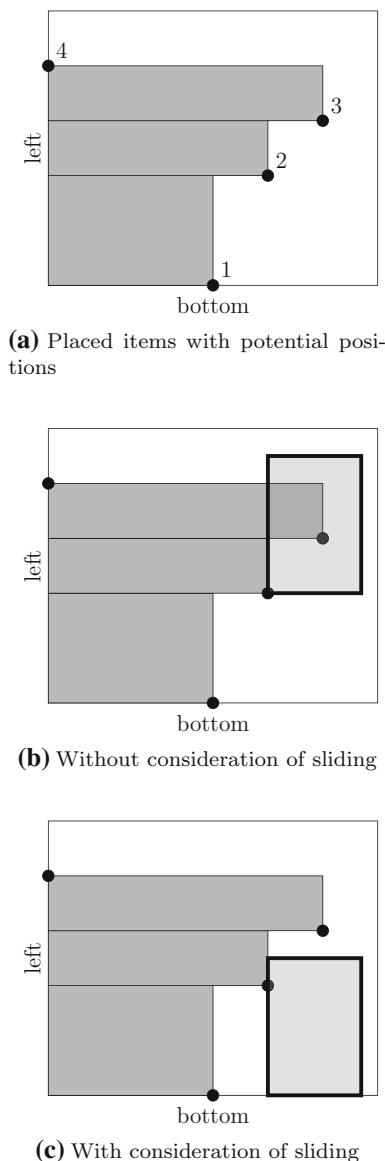ul (backhaul) customers. The items of one customer are sorted by the item fragility (non-fragile items first), breaking ties by non-increasing volume, breaking ties by non-increasing length and breaking ties by non-increasing width.
- The packing problem to be solved is the orthogonal packing problem (OPP), i.e. the objective is not to maximize the filling rate but to load *all* items belonging to a route feasibly into the loading space. Thus, the procedure is aborted as soon as one item cannot be placed feasibly in any available position.

```
 1: procedure DBLF-COMB(in: instance data, sequence
     of items I, out: packing plan)
 2:    initialize sorted set of positions P := {(0, 0, 0)}
 3:    for i := 1 to |I| do
 4:       current item item := I(i)
 5:       placed := false  ▷ binary variable stating whether
          an item could be placed feasibly or not
 6:       for p := 1 to |P| do
 7:          if placement_DBLF(item, p) is feasible then
 8:             placed := true, p' := p, break
 9:          end if
10:       end for
11:       if placed = false then
12:          for p := 1 to |P| do
13:             if placement_DBLF⁺(item, p) is feasible then
14:                placed := true, p' := p, break
15:             end if
16:          end for
17:       end if
18:       if placed = true then
19:          place item in P(p') (with sliding if necessary)
20:          move item as far as possible towards the back,
             bottom, left
21:          update P
22:       end if
23:    end for
24: end procedure
```

Fig. 6 DBLF-Comb

- For the SL approach, the heuristics were modified. The LIFO restriction as described above must be considered along the width axis of the loading space in order to avoid rearrangements when (un-)loading items. Moreover, it should also be considered along the length axis in order to guarantee that the unloading of linehaul items gradually decreases L_{LH} and, thus, creates space for the backhaul items (see Fig. 1). However, simply extending the restriction in this manner resulted in packing patterns as in Fig. 7a. Let the visiting sequence here be {4, 3, 2, 1}. Items of neither customer 3 nor 4 can be placed behind item I_{22} (from the view of the unloading side). In order to avoid such gaps, the placement priorities have been adjusted for the SL: with first priority, an item is to be placed as close as possible towards the origin of the loading space, where the distance is defined as the sum of length and width coordinate of a given placement position. Subsequently, the DBLF rule is applied again, i.e. ties are broken by non-increasing length coordinate, then by height coordinate and then by width coordinate. This rule results in patterns like in Fig. 7b where items tend to be stacked first and then to be arranged towards the side from which they are loaded (cabin or rear side).
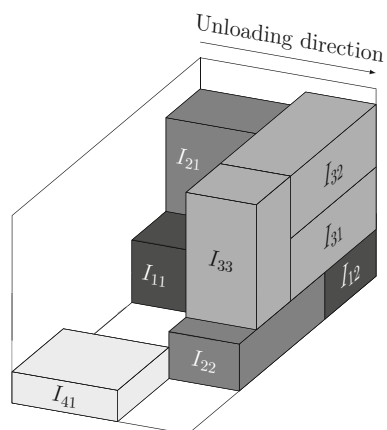
(a) Modified LIFO restriction
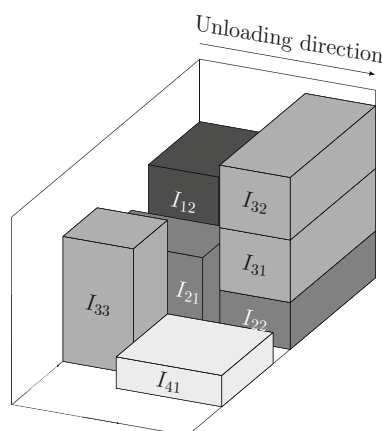
(b) Modified placement rule

Fig. 7 Example packing patterns for side loading

4.3 Integration of routing and packing

The packing procedure is executed in order to check whether the items transported on a tour can be packed feasibly according to the above-formulated packing constraints. In the course of a packing check, (generally) two packing plans—one for the linehaul items and one for the backhaul items of a tour—are determined. Moreover, in the case of the SL approach, the procedure tests whether linehaul and backhaul items would overlap at any stop of the route according to the generated packing plans (see restriction (S5), Sect. 2).

A packing check is made when a route is tested for feasibility (see Fig. 3). Feasibility tests are conducted whenever a new solution is generated, i.e. during the generation of the initial solution, during the generation of neighbour solutions and in connection with the local optimization. Firstly, it is checked whether the items transported simultaneously exceed the weight and volume capacity of the vehicle at any

stop of the tour. If this is not the case, the packing procedure is called. Note that not every neighbour of the current solutions is tested for feasibility during the TS. Only if the corresponding move can potentially be added to the candidate list, the routes affected by the move are tested. That way, the efficiency is increased significantly (cf. Fig. 3, lines 11,18).

In the course of the packing procedure, the maximum loading length of a packing plan is determined. This is needed (i) for the tests regarding the SL approach, and (ii) for the second routine of the local optimization (reverse operator).

The application of the packing procedure is usually computationally very expensive. Therefore, a cache is used which stores routes that have already been tested for packing feasibility. The cache is organized as a map with a maximum size of 1,000,000 elements. The oldest route in the map is removed if the size is exhausted.

5 Computational experiments

5.1 Benchmark instances

5.1.1 One-dimensional instances

In order to make sure the implemented routing approach is working properly, it was applied to one-dimensional (1D) benchmark instances and the results were compared to the ones obtained by recently published methods for the VRPMB. Three instance sets were used. Two of them, namely GJB89 and TV97, were originally proposed for the VRPCB by Goetschalckx and Jacobs-Blecha (1989) and Toth and Vigo (1997), respectively. Later these sets were used as VRPMB instances in Halse (1992) and Wade (2002) by relaxing the delivery-before-pickups assumption. At the same time, the "fixed fleet" restriction of the original instances, i.e. constraint (S4) in our problem formulation, was dropped. This specification of the VRPMB instance sets GJB89 and TV97 has been maintained in all further VRPMB papers (see Wassan et al. (2008), p. 154 and Wassan et al. (2013)), and we will also observe it. As a consequence, the GJB89 set for the VRPMB includes only 46 instances (not 62 as for the VRPCB) while the TV97 set has 33 instances as in the original version. In the set GJB89, the number of customers varies between 25 and 150 customer locations and there are 50%, 66% or 80% linehaul customers. The instances of the set TV97 have 21 to 100 customer locations and the shares of linehaul customers are the same as in set GJB89. Finally, the third set (SN99) was generated by Salhi and Nagy (1999) for the VRPMB. This set consists of 42 instances with 50–199 customer locations with 10%, 25% and 50% backhaul customers. Again, there is no constraint in the set SN99 with regard to the number of vehicles (routes). Note that we con-

sider only those 21 instances that do not include tour length limits and drop times.

5.1.2 Three-dimensional instances

New 3L-VRPMB instances were generated which cover different features of the problem. The number of customers is set to 20, 60 and 100, and the share of linehaul customers to 50% and 80%. The locations of the customers were determined randomly. Moreover, we keep the total number fixed to 200 items so that the number of items per customer depends on the number of customers in an instance (5–15 for $n = 20$, 2–5 for $n = 60$, 1–3 for $n = 100$).

We consider different levels of heterogeneity with respect to the items, that is, instances were generated with 3, 10 or 100 different item types. The edge dimensions and weights of each item type were generated randomly. There are instances with large items and instances with small items. The length and width of large items were uniformly randomly generated in the intervals $[0.2L, 0.6L]$ and $[0.2W, 0.6W]$, respectively (cf. Gendreau et al. (2006)). The height of an item was determined in the interval $[0.2H, 0.5H]$. The factor 0.5 was chosen for the upper limit of the interval to ensure that the items can be placed inside the partitioned loading space (cf. Sect. 2). For instances with 20 customers, only small items are considered, since all large items of one customer would nearly completely fill the loading space due to the comparatively large number of items per customer. The length (respectively, width and height) of small items is generated in the interval $[0.1L, 0.3L]$ (respectively, $[0.1W, 0.3W]$ and $[0.1H, 0.3H]$). In addition, instances with mixed item size were also generated which have ca. 50% small items and 50% large items. Any instance with mixed items results by merging an instance with small items and an instance with large items where both instances share the same characteristics (except item size).

The weight (in weight units (WU)) of an item type was randomly generated between $0.001[\frac{WU}{VU}] \cdot hboxvol[VU]$ and $0.01 [\frac{WU}{VU}] \cdot vol[VU]$, where vol is the item's volume (in volume units (VU)). 20%, but at least one, of the item types is fragile. The loading space dimensions are $L = 60$, $W = 25$, $H = 30$ and the weight capacity $D = 200$. The percentage α in the stability constraint (PC2) is set to 75%.

For the 3D instances, constraint (S4) on the available number of vehicles v_{max} is in force. v_{max} is determined by applying the Savings heuristic to the instances. The resulting number of routes is used as v_{max}. This way, feasible solutions regarding the vehicle constraint (S4) are ensured.

Taking into account the variation of the customer number, percentage of linehaul customers, item size and heterogeneity of items, there are 42 feature combinations. For each combination, 10 instances were generated randomly resulting in

Table 2 Parameter settings

Parameter	Value	Parameter	Value
$count_{max}$	6	p	4
$iter_{min}$	1500	η_{max}	3
$iter_{no\_impr}$	200	λ_{tls}	4
n_{CL}	3	λ_{gap}	50
tls_{init}	1	ϕ_{inc}	1.1
tp_{max}	30%	ϕ_{dec}	1.1

420 3L-VRPMB instances, among them 120 instances contain a mix of small and large items.

The 3L-VRPMB instances as well as our best solutions can be found online at http://www.mansci.ovgu.de/Forschung/Materialien.html. In addition, the well-known 27 3L-CVRP instances introduced in Gendreau et al. (2006) were used for a numerical comparison.

5.2 Parameter setting

The parameter settings were mostly adapted from Nagy et al. (2013). The values for the newly introduced parameters were determined in pretests with a set of test instances taken from the three sets for the VRPMB. Their settings are given in Table 2.

5.3 Computational results

The hybrid algorithm was implemented in C++, and the experiments were conducted on a Haswell system with up to 3.2 GHz and 16 GB RAM per core.

5.3.1 One-dimensional instances

The RTS in combination with the Savings procedure for the construction of the initial solutions will be referred to as RTS_Sav in the following. The combination with the Sweep heuristic shall be called RTS_Sweep. Moreover, we consider the combination of RTS_Sav and RTS_Sweep, denoted as RTS_Comb, where first RTS_Sav and then RTS_Sweep are applied each with the same parameter setting. Due to some random components in the algorithm, five runs for each variant of the RTS were conducted for each instance. The results for the 1D instances are summarized in Tables 3 and 4. Note that only 45 GJB89 instances are considered here as one of the compared methods (Wassan et al. (2008)) does not provide results for one instance.

For each instance set, the obtained solutions are compared to the best solutions of Nagy et al. (2013) (NWS13) and the best known solutions from the literature (BKS). Furthermore, an analogue comparison is done with a second well-perform-

ing VRPMB method; in the case of the instance sets GJB89 and TV97, the RTS by Wassan et al. (2008) (WNA08) is taken while the ALNS by Ropke and Pisinger (2006) is used for the (reduced) set SN99.

In the first column of Table 3, the instance set and the RTS variant are indicated. The columns Dev-best (2–4) contain for each compared method and BKS the average percentage deviation of the best found TTD per instance from the best TTD of the compared method and the BKS, respectively. The columns Dev-avg (5–7) contain for each compared method and BKS the average percentage deviation of the average obtained TTDs over five runs per instance from the best TTD of the compared method and the BKS, respectively. For each instance set and each compared method (including our RTS), the number of used vehicles over all instances per set is calculated. Note that in the BKS case the number of used vehicles is calculated using the solutions with best TTD values. The columns Dec-veh (8–10) contain for each compared method and BKS the difference of the sums of used vehicles over all instances of a set between our RTS and the compared method and the BKS, respectively. Negative values indicate lower numbers of used vehicle in the solutions provided by the RTS.

In the first two columns of Table 4, the instance sets and related numbers of instances are given. In the third column, the numbers of new best solutions provided by the proposed RTS are indicated while the last three columns show the average run times per instance (in seconds).

The comparison of best solutions in terms of total travel distance can be summed up as follows where only our combined RTS is considered. Compared to the method provided in Nagy et al. (2013) (NWS13), an improvement of -1.21% could be reached by our RTS over all 99 instances (note that negative values indicate that our results are better than results of a compared method). Taking the sets GJB89 and TV97, our RTS is slightly better than the method proposed in Wassan et al. (2008) (WNA08) with a mean improvement of -0.5%. Among the compared methods, only the ALNS presented in Ropke and Pisinger (2006) (RP06) is clearly dominating our method, for instance set SN99 with an improvement of 1.59 %.

The number of used vehicles (or routes) represents a second optimization criterion that is often taken into account in vehicle routing problems. Our combined RTS achieves mostly the same numbers of vehicles as the compared methods. Significant differences only occur compared to the ALNS proposed in Ropke and Pisinger (2006) (the numbers of vehicles used by the ALNS of Ropke and Pisinger (2006) were picked up from Nagy et al. (2013), p. 212). In total, 80 vehicles less are used in the solutions of RTS_Comb.

The deviations of our best solutions from the BKS (see Table 3, column 4) indicate RTS_Sav produces better results than RTS_Sweep. The difference amounts to 0.66%-points

over all 99 instances. Applying the Savings heuristic instead of or combined with the Sweep heuristic did enable us to improve the results obtained by Nagy et al. (2013).

However, Table 4 makes clear that this improvement was only possible on the cost of higher run times. Note that Nagy et al. (2013) state that less than a minute was needed for the majority of the instances. Nevertheless, the run times for all variants of our RTS seem to be acceptable. Finally, the numbers of new best solutions in Table 4 prove that the RTS performs relatively well for the well-known VRPMB instance sets.

5.3.2 Three-dimensional instances

The test of the hybrid algorithm deals with three different aspects. First, as the final hybrid algorithm must be equipped with a packing heuristic, the available packing heuristics were initially tested and compared. Second, the impact of the initial solution construction heuristic as well as of the loading approach is studied. Third, a numerical comparison with the 3L-VRPMB heuristic by Reil et al. (2018) is conducted. The parameter settings determined with the one-dimensional instances were also applied to the 3D case. Third, a numerical comparison with the 3L-VRPMB heuristic by Reil et al. (2018) as well as the 3L-CVRP heuristics by Gendreau et al. (2006) and Zhang et al. (2015) is conducted.

Determination of the best packing heuristic

For this purpose, a test set of 30 out of 420 instances was formed. The instances were chosen at random from the instances with small and large items so that all remaining feature attributes (number of customers, number of item types, etc.) are combined.

To determine the best packing heuristic, we compare six variants of the hybrid algorithm with the Savings heuristic which differ regarding the used packing heuristic and the loading approach. Each of the available packing heuristics (DBLF, DBLF+ and DBLF-Comb) is combined with each loading approach (loading space partition (LSP) and side loading (SL)). Each of the selected instances was run five times with each of the six variants. In Table 5, the obtained TTD results for each of the variants are compared to the best found solutions over all six hybrid algorithm variants. Per variant, the table contains the average (Dev-avg) and maximum (Dev-max) percentage deviations of the obtained TTDs from the best found ones.

The best results were obtained with the SL approach and the DBLF heuristic. However, in combination with LSP, DBLF-Comb produced better results than the plain DBLF heuristic. Hence, the respective results were examined in more detail with the conclusion that DBLF seems to work better for instances with small items whereas the combination dominates in the case of large items. Therefore, in the final tests, the packing heuristic is chosen depending on the

Table 3 RTS results for 1D instances (A)

| Set/variant | Total travel distance | | | | | | No. of used vehicles | | |
	Dev-best (%)			Dev-avg (%)			Diff-veh		
GJB89	**NWS13**	**WNA08**	**BKS**	**NWS13**	**WNA08**	**BKS**	**NWS13**	**WNA08**	**BKS**
RTS_Sav	− 0.30	− 0.19	0.51	0.18	0.29	0.99	1	1	4
RTS_Sweep	0.26	0.37	1.08	1.22	1.33	2.05	0	0	3
RTS_Comb	− 0.67	− 0.56	0.14	0.70	0.81	1.52	0	0	3
TV97	**NWS13**	**WNA08**	**BKS**	**NWS13**	**WNA08**	**BKS**	**NWS13**	**WNA08**	**BKS**
RTS_Sav	− 0.47	0.06	0.45	0.23	0.77	1.16	− 2	− 2	− 2
RTS_Sweep	− 0.60	− 0.07	0.32	0.01	0.55	0.94	0	0	0
RTS_Comb	− 0.93	− 0.41	− 0.02	0.12	0.66	1.05	0	0	0
SN99	**NWS13**	**RP06**	**BKS**	**NWS13**	**RP06**	**BKS**	**NWS13**	**RP06**	**BKS**
RTS_Sav	− 2.24	2.18	2.25	− 1.75	2.68	2.76	− 2	− 82	− 17
RTS_Sweep	− 0.34	4.27	4.34	0.25	4.89	4.97	0	− 80	− 15
RTS_Comb	− 2.80	1.59	1.67	-0.75	3.79	3.86	0	− 80	− 15

Table 4 RTS results for 1D instances (B)

| Set | # | New best solutions (TTD) | Run time (s) | | |
			RTS_Sav	*RTS_Sweep*	*RTS_Comb*
GJB89	45	18	188.0	215.8	403.8
TV97	33	12	108.4	89.6	198.0
SN99	21	0	491.6	523.8	1015.4
Total	99	30	223.7	236.8	460.5

Table 5 Comparison of packing heuristics (TTD)

| Packing heuristic | Dev-avg(%) | | Dev-max(%) | |
	LSP	SL	LSP	SL
DBLF	18.7	1.3	44.8	6.7
DBLF+	18.1	5.0	42.3	10.8
DBLF-Comb	17.0	2.6	42.3	8.5

item sizes: DBLF for small items and DBLF-Comb for large items. DBLF is also used in later experiments for instances with mixed items.

Impact of initial solution heuristic and loading approach

To study these issues, experiments were conducted with all 420 instances each of which was solved five times by four variants of the hybrid algorithm. These variants result by using both loading approaches combined with both initial solution heuristics. The combined variant of the hybrid algorithm is not applied here (as in the 1D case) since the run time effort would be presumably too high.

The results are presented first in Tables 6 and 7. They contain in the first three columns a description of the different instance groups. Then, for each of the four variants of the hybrid algorithm mean values of average and best obtained TTDs over five runs and of related numbers of used vehicles are indicated.

Reading the last lines of the tables, one can realize that the Savings heuristic performs better than the Sweep heuristic and the SL approach dominates the LSP approach. Nevertheless, a more detailed presentation regarding the impact of the initial solution heuristic and loading approach should be given in the next two tables.

The results regarding the initial solution construction heuristics are summarized in Table 8. The results obtained with *RTS_Sav* and the respective loading approaches serve as a benchmark here. The column *Dev-avg* contains the average percentage deviations of the TTDs obtained with *RTS_Sweep* from the average benchmark TTDs aggregated over both loading approaches.

The results support the findings from the 1D tests (Table 3) indicating that *RTS_Sav* slightly outperforms *RTS_Sweep*. The TTDs obtained with *RTS_Sweep* deviate on average 0.77% from the TTDs obtained with *RTS_Sav*.

Table 9 provides a comparison of the two different loading approaches. Here, the results obtained with LSP were used as benchmark for each initial solution heuristic. In the table, the average deviations (Dev-avg) of the SL solutions (w.r.t. the TTDs) from the LSP solutions (aggregated over both initial solution heuristics) and the average numbers of used vehi-

Table 6 Results for the hybrid algorithm with the Savings heuristic and both loading approaches

Instances			Sav/LSP			Sav/SL		
n	Items	LH (%)	TTD-avg	TTD-best	v	TTD-avg	TTD-best	v
20	Small	50	367.04	366.64	2.7	346.04	343.90	2.0
		80	401.18	400.43	3.6	353.63	353.18	2.5
60	Small	50	549.95	544.88	2.9	545.37	540.87	2.6
		80	566.99	563.76	3.4	537.60	533.50	2.7
	Mixed	50	1,077.01	1,066.51	11.9	816.10	807.62	7.7
		80	1,334.40	1,325.68	19.3	847.65	834.50	9.7
	Large	50	1,573.87	1,562.13	20.7	1,101.09	1,092.30	11.9
		80	2,022.44	2,011.67	31.3	1,298.59	1,285.89	15.9
100	Small	50	685.88	680.90	2.9	687.96	683.54	2.8
		80	702.45	698.03	3.6	685.00	678.90	2.9
	Mixed	50	1,090.95	1,081.76	10.7	931.25	917.19	7.2
		80	1,444.22	1,437.70	19.6	1,027.11	1,016.23	10.0
	Large	50	1,585.90	1,575.95	18.7	1,250.00	1,229.68	12.3
		80	2,108.14	2,099.73	30.8	1,443.52	1,426.37	16.1
Total			1,107.08	1,100.32	12.7	846.24	837.16	6.1

Table 7 Results for the hybrid algorithm with the Sweep heuristic and both loading approaches

Instances			Sweep/LSP			Sweep/SL		
n	Items	LH (%)	TTD-avg	TTD-best	v	TTD-avg	TTD-best	v
20	small	50	367.78	366.44	2.8	345.64	344.12	2.1
		80	401.24	400.83	3.6	353.87	353.24	2.4
60	Small	50	551.42	545.58	2.8	546.09	540.00	2.5
		80	566.95	560.60	3.6	542.49	536.93	2.7
	Mixed	50	1,078.80	1,068.71	12.3	851.58	836.66	7.6
		80	1,338.70	1,329.14	18.3	959.12	944.74	10.1
	Large	50	1,569.51	1,559.37	20.6	1,112.69	1,103.81	12.1
		80	2,019.70	2,010.20	31.4	1,309.97	1,297.87	16.5
100	Small	50	693.47	683.28	2.7	692.96	683.51	2.8
		80	708.73	701.24	3.5	692.43	683.84	2.8
	Mixed	50	1,113.41	1,100.76	10.6	959.65	942.53	8.6
		80	1,465.85	1,453.70	19.2	1,109.79	1,093.22	11.4
	Large	50	1,592.64	1,578.37	19.3	1,275.86	1,246.34	12.5
		80	2,118.04	2,105.98	30.4	1,465.13	1,443.13	16.3
Total			1,112.46	1,103.75	12.2	872.66	860.71	6.4

cles of the best solutions (v-best) are reported. Furthermore, the average computing times (in seconds) are listed in the columns "Run times (s)".

Not surprisingly, much better results can be obtained with the SL approach. The respective TTDs are on average about 16% lower than the TTDs obtained with the LSP and less vehicles are needed. These results could be expected as the SL approach allows making use of the whole loading space, whereas half of it is left empty at least at the very beginning and the very end of a tour if the loading space is separated. However, differences can be observed among the different instance classes. The SL approach seems to be particularly

beneficial if the items are large (as smaller items can be placed more easily also within a smaller loading space), and if the share of linehaul and backhaul items is unequal. Moreover, the differences decrease with an increasing number of customers, i.e. with a decreasing number of items per customer.

Moreover, the hybrid algorithm with the LSP requires less computing time than the variant with the SL approach. Two reasons can be identified. Firstly, checking feasibility for the SL approach takes longer because the two generated packing plans need to be compared for each backhaul stop of a tour to avoid any overlapping of linehaul and backhaul items. Secondly, the SL approach allows to form longer tours which

Table 8 Results for 3D instances: Initial solution construction heuristics

Instances		Dev-avg (%)]
n	Items	RTS_Sweep from RTS_Sav
20	small	0.05
60	Small	0.36
	Mixed	0.57
	Large	0.13
100	Small	0.98
	Mixed	2.10
	Large	1.20
Total		0.77

also require a higher packing effort. Independently of the loading approach, it can be observed that the computing times increase with the size of the underlying VRP, i.e. with the number of customers, and that the computing times depend on the item sizes. Instances with large items can be solved much faster than instances with small items. In the case of small items, much more customers can be merged into one tour (resulting in more items per tour) which requires higher efforts for the packing checks.

Numerical comparison with the 3L-VRPMB heuristic by Reil et al. Reil et al. (2018)

For the numerical comparison, 128 of the 3L-VRPMB instances introduced above were selected among the 420 instances. Thirty-two instances were chosen at random from the instances with 20 customers. As mentioned above, those contain only instances with small items. Forty-eight instances were chosen at random for instances with 60 and with 100

customers. More instances were selected for those instance groups as they contain instances with small, mixed and large items.

The heuristic by Reil et al. (2018) (RBM18) is able to work with different loading approaches and strategies. In this comparison, the side loading approach is taken which turned out to be the strongest available loading approach for the 3L-VRPMB(TW) (see Reil et al. (2018), p. 14). The observed constraints for packing and routing are similar to the RTS heuristic; in particular, a supporting rate of 75% in constraint (PC2) is required for the heuristic by Reil et al. (2018), too. The RTS heuristic is applied with the Savings heuristic and with both loading approaches. Since only one run of the procedure of Reil et al. (2018) was performed on each selected instance, we randomly chose the result of one run (out of five) of the RTS on each one of those instances for comparison purposes.

Tables 10 and 11 contain the results of the comparison which are again organized by instance groups. In the first two columns of Table 10 and 11, the instance groups are characterized. In the following nine columns of Table 10, for each of the compared methods, average values per instance group are given for the total travel distance, the number of used vehicles and the run time (in seconds). In the last four columns of Table 11, the minimum and average deviations per instance group of the total travel distances gained by the heuristic of Reil et al. (2018) to the ones yielded by RTS(LSP) and RTS(SL) are listed. Negative values indicate better solutions obtained by Reil et al. (2018).

Both hybrid RTS variants (LSP and SL) perform considerably better than the heuristic by Reil et al. (2018) for the given set of instances in terms of total travel distance as well

Table 9 Results for 3D instances: Loading approaches

Instances			Dev-avg(%)	v-best		Run time (s)	
n	Items	LH(%)	Of SL from LSP	LSP	SL	LSP	SL
20	Small	50	− 5.39	2.8	2.1	8.1	22.3
		80	− 11.37	3.6	2.5	6.0	23.6
60	Small	50	− 0.84	2.9	2.6	246.9	286.7
		80	− 4.57	3.5	2.7	150.9	247.6
	Mixed	50	− 20.61	12.1	8.0	18.8	31.9
		80	− 28.33	17.9	9.9	14.9	27.8
	Large	50	− 28.99	20.3	11.8	10.7	15.9
		80	− 35.32	30.5	16.1	12.5	12.8
100	Small	50	0.13	2.9	2.8	1, 778.2	1, 719.0
		80	− 2.36	3.6	2.9	1, 088.1	1, 606.4
	Mixed	50	− 13.70	11.0	8.2	117.7	214.9
		80	− 25.49	17.3	10.9	66.6	123.2
	Large	50	− 20.33	19.0	13.0	43.7	71.3
		80	− 30.88	29.2	16.6	32.3	53.9
Total			− 16.19	12.6	7.9	256.8	318.4

Table 10 Results for 3D instances: Comparison with Reil et al. (2018) (A)

Instances		RBM18			RTS (LSP)			RTS (SL)		
n	Items	TTD	v	Run time (s)	TTD	v	Run time (s)	TTD	v	Run time (s)
20	Small	462.9	4.6	481.8	385.2	3.2	31.0	351.9	2.3	90.7
60	Small	813.2	8.2	482.4	568.2	3.4	399.8	543.3	2.8	731.4
	Mixed	1, 851.2	26.8	723.4	1, 140.6	13.6	26.9	871.2	8.6	46.4
	Large	2, 501.8	38.5	963.4	1, 753.0	24.4	23.1	1, 189.7	13.8	26.1
100	Small	994.7	9.9	483.2	688.4	3.1	2, 314.5	685.8	2.8	3, 015.8
	Mixed	2, 088.8	28.0	545.6	1, 360.5	15.9	153.9	1, 050.2	10.2	167.1
	Large	3, 441.9	49.7	866.1	2, 068.5	27.3	70.5	1, 498.7	16.8	63.3
Total		1, 519.8	20.4	619.5	1, 011.4	11.2	433.9	798.8	7.1	598.1

Table 11 Results for 3D instances: Comparison with Reil et al. (2018) (B)

Instances		Deviation of RBM18 from RTS (%)			
n	Items	Min (LSP)	Min (SL)	Avg (LSP)	Avg (SL)
20	Small	− 4.70	18.04	20.80	31.53
60	Small	27.11	31.64	43.18	49.89
	Mixed	21.80	58.91	64.03	112.81
	Large	− 14.74	45.02	44.84	110.02
100	Small	20.42	21.61	44.47	45.12
	Mixed	5.18	32.98	52.44	96.86
	Large	− 0.38	84.61	70.08	130.68
Total		6.74	37.38	44.48	74.05

as number of used vehicles while the running times are comparable. In the heuristic of Reil et al. (2018), each customer of a route has a special compartment, i.e. segment of the loading space, that is filled exclusively by the boxes of this customer. In the instances used here, very often a customer has only few and/or small boxes. In this case, the volume of a customer compartment cannot be utilized very well. Hence, more vehicles (or routes) and also longer travel distances are needed for transporting all the goods. It is our belief that for differently constructed instances the compared heuristic could perform better. This is proven by those instances in which the heuristic by Reil et al. (2018) achieved better results than the RTS (see Table 11, column 3). On the other hand, our instances do reflect situations where individuals make purchases in department stores consisting of few or small things and therefore make sense. We can conclude that—as in other vehicle routing areas—several heuristics with different abilities are necessary to cover the full range of requirements.

Numerical experiments on the instances of Gendreau et al. Gendreau et al. (2006)

Finally, the RTS was applied to the well-known 3L-CVRP instances of Gendreau et al. (2006). In this regard, its results are compared to those of Gendreau et al. (2006) and Zhang et al. (2015). The latter have developed an evolutionary local search which proved to be one of the best approaches for the 3L-CVRP to date.

Table 12 presents the results of these experiments. Instance information is provided in the first columns [name, number of customers (n), total number of items (m), maximum number of vehicles (v_{max})]. The RTS heuristic was applied with both the Savings and Sweep heuristic and with both loading approaches. For the rear loading approach, the loading space separation was omitted as a 3L-CVRP instance does not contain any backhaul customers.

In Table 12, the TTD and number of routes (v) of the best solutions among all runs are listed. In addition, the average run times are given in seconds (t (s)). For the comparison with the approaches from the literature, the best TTDs and average run times of Gendreau et al. (2006) and Zhang et al. (2015) are presented as well as the deviations of the TTDs provided by the RTS from the TTDs of the comparison procedures.

As can be seen, our approach achieves slightly better solutions than the ones of Gendreau et al. (2006). The local search of Zhang et al. (2015) provides considerably better results. The RTS does outperform the approach of Zhang et al. (2015) in one case (E016-05m). However, it must be noted that the RTS did not achieve to find a solution where the vehicle number restriction is satisfied. The found solution requires one vehicle more than allowed. As mentioned above, the only mechanism to handle this restriction is the penalization of additional tours. For the presented instances, this approach seems to be insufficient as the maximum number of vehicles is often violated by the best found solutions.

An advantage of our approach is, though, that moderate solutions are provided rather fast requiring on average only about 9 seconds of run time.

Table 12 Results for instances of Gendreau et al. (2006) and comparison with Gendreau et al. (2006) and Zhang et al. (2015)

Instance				Results RTS				Comparison with Gendreau et al. (2006)			Comparison with Zhang et al. (2015)		
Name	n	m	v_{max}	TTD-best	v	t (s)		TTD	t (s)	Dev (%)	TTD	t (s)	Dev (%)
E016-03m	15	32	4	325.08	5	0.1		316.32	129.5	2.77	302.02	3.2	7.64
E016-05m	15	26	5	332.30	6	0.0		350.58	5.3	−5.21	334.96	0.2	−0.79
E021-04m	20	37	4	407.72	5	1.0		447.73	461.1	−8.94	385.53	365.6	5.76
E021-06m	20	36	6	449.03	7	0.1		448.48	181.1	0.12	437.19	20.3	2.71
E022-04g	21	45	6	508.20	6	1.0		464.24	75.8	9.47	443.17	151.9	14.67
E022-06m	21	40	6	502.51	7	1.0		504.46	1,167.9	−0.39	501.06	3.4	0.29
E023-03g	22	46	6	829.02	6	1.3		831.66	181.1	−0.32	771.07	19.5	7.52
E023-05s	22	43	6	894.58	7	1.1		871.77	156.1	2.62	813.13	325.2	10.02
E026-08m	25	50	8	667.29	9	1.1		666.10	1,468.5	0.18	630.13	6.3	5.90
E030-03g	29	62	8	922.69	8	3.3		911.16	714.0	1.27	824.69	473.5	11.88
E030-04s	29	58	8	868.70	8	2.6		819.36	396.4	6.02	776.19	220.5	11.92
E031-09h	30	63	9	631.70	10	1.5		651.58	268.1	−3.05	610.23	21.6	3.52
E033-03n	32	61	8	2,873.23	7	4.0		2,928.34	1,639.1	−1.88	2,656.72	495.0	8.15
E033-04g	32	72	9	1,507.94	8	7.1		1,559.64	3,451.6	−3.32	1,369.35	1,079.4	10.12
E033-05s	32	68	9	1,486.29	8	6.1		1,452.34	2,327.4	2.34	1,338.35	1,295.0	11.05
E036-11h	35	63	11	719.53	12	2.1		707.85	2,550.3	1.65	698.61	6.0	2.99
E041-14h	40	79	14	878.83	15	2.6		920.87	2,142.5	−4.57	866.40	17.3	1.43
E045-04f	44	94	11	1,409.86	12	6.9		1,400.52	1,452.9	0.67	1,223.64	1,104.4	15.22
E051-05e	50	99	12	882.01	12	9.1		871.29	1,822.3	1.23	744.33	1,494.9	18.50
E072-04f	71	147	18	684.43	18	12.4		732.12	790.0	−6.51	570.82	2,441.6	19.90
E076-07s	75	155	17	1,219.07	17	22.4		1,275.20	2,370.3	−4.40	1,063.20	2,510.8	14.66
E076-08s	75	146	18	1,338.57	18	20.8		1,277.94	1,611.3	4.74	1,140.54	2,689.2	17.36
E076-10e	75	150	17	1,281.67	18	21.2		1,258.16	6,725.6	1.87	1,094.33	2,887.9	17.12
E076-14s	75	143	16	1,256.14	17	18.7		1,307.09	6,619.3	−3.90	1,101.67	2,282.7	14.02
E101-08e	100	193	22	1,623.32	23	34.2		1,570.72	5,630.9	3.35	1,359.25	2,846.1	19.43
E101-10c	100	199	26	1,865.45	27	30.3		1,847.95	4,123.7	0.95	1,545.67	2,958.3	20.69
E101-14s	100	198	23	1,740.88	25	25.7		1,747.52	7,127.2	−0.38	1,479.73	3,065.5	17.65
				1,040.97		8.8		1,042.26	2,058.9	−0.13	928.96	1,066.1	10.72

6 Conclusions

Transporting linehaul and backhaul goods in the same tours can be a useful mean to reduce empty running trucks, travelled distances, fuel consumption and, in consequence, to reduce costs. Although the integration of backhauls into the VRP was studied frequently in the research, the transported goods and available capacities were mostly considered to be one-dimensional. Applying solutions obtained by solving such problems could turn out to be infeasible when the transported goods are bulky. Therefore, we present a 3L-VRPMB which includes not only the backhauls into the VRP but also three-dimensional loading constraints so that, for example, load stability or loading sequences can be considered. In order to solve it, a hybrid algorithm consisting of a reactive tabu search (which was originally developed for the one-dimensional VRPMB) combined with a packing construction heuristic was implemented.

The procedure was tested for both one-dimensional and three-dimensional test instances. The one-dimensional tests indicate that the implemented RTS is comparable to the original approach. Moreover, a second heuristic for constructing initial solutions was applied which led to further improvements of the results.

New instances were generated for the 3L-VRPMB and different packing construction heuristics were implemented and compared. The final hybrid algorithm combined the RTS with variants of the deepest-bottom-left-fill approach. Since linehaul and backhaul customers can be visited in any sequence in the VRPMB, linehaul and backhaul items are (partly) transported simultaneously. In the hybrid approach, this is realized by two different loading approaches which ensure that any reloading during a tour is avoided. They include rear-loaded vehicles with horizontally separated loading spaces into linehaul and backhaul sections, and side-loaded vehicles. The best results—in terms of both total travel distance and number of tours—were obtained with the side loading approach.

A comparison has been conducted with the 3L-VRPMB heuristic by Reil et al. (2018). In this comparison, our hybrid algorithm performed considerably better than the method of Reil et al. (2018) in terms of total travel distance and number of routes. This may have been caused partially by the use of benchmark instances for which the method of Reil et al. (2018) is less suitable. In addition, the hybrid RTS heuristic was compared to further heuristics presented in the literature using the 3L-CVRP instances of Gendreau et al. (2006). In terms of solution quality, the RTS is comparable to the approach of Gendreau et al. (2006). The local search of Zhang et al. (2015) performs significantly better. Yet, our approach is able to provide moderate solutions in rather small run times.

The mixture of linehaul and backhaul items within a route leads to the application of the presented loading approaches. Both face limitations which could be tackled further in future research regarding the 3L-VRPMB. For example, the side loading approach could provide more flexibility if a mixture of linehaul and backhaul items along the length axis of the loading space was allowed. This would make packing more complicated but might lead to better utilization of the loading space and, thus, better solutions. Future studies on the 3L-VRPMB and other VRPs with backhauls should also integrate further and more complicated packing constraints, e.g. axle weight constraints, to cope with even more difficult real-world situations.

Appendix

Following, the detailed results of the one-dimensional tests are presented. Tables 13, 14 and 15 state the instance characteristics in the first columns. The columns "BKS" contains the TTD and the number of vehicles (v) of the best known solution from the literature. The next columns contain the best found solutions (TTD-best) of our RTS and the number of vehicles corresponding to the best solution (v-best). The last column states whether the best solution was obtained in connection with the Sweep (*Sw*) heuristic, the Saving (*Sav*) heuristic, or both.

Table 13 Results RTS for GJB89 instance set

Instance			BKS		RTS-Comb		
Name	n	LH (%)	TTD	v	TTD-best	v-best	init
A1	25	80	223,085	8	223,088	8	both
A2	25	80	169,497	5	169,500	5	both
A3/A4	25	80	142,032	3	142,034	3	both
B1	30	66	232,430	7	232,436	7	both
B2	30	66	179,194	5	179,194	4	Sav
B3	30	66	145,699	3	145,702	3	both
C1	40	50	237,100	7	237,110	7	both
C2	40	50	196,833	5	**196,683**	5	Sw
C3/C4	40	50	164,794	3	164,891	3	both
D1/D2	38	80	307,109	11	307,110	11	both
D3	38	80	220,700	7	220,751	7	both
D4	38	80	182,496	5	182,928	5	both
E1	45	66	220,742	7	220,742	7	both
E2	45	66	190,084	4	**190,049**	4	Sav
E3	45	66	182,804	4	**181,941**	4	Sav
F1/F2	60	50	243,599	6	244,353	6	Sav
F3	60	50	212,296	4	212,296	4	Sw
F4	60	50	200,964	4	**198,709**	4	Sav
G1	57	80	297,707	10	**297,656**	10	Sav
G2	57	80	234,653	6	**234,101**	6	Sw
G3/G4	57	80	213,757	5	**212,748**	5	both
G5	57	80	202,610	4	**200,521**	4	Sav
G6	57	80	188,823	3	**188,696**	3	Sw
H1	68	66	235,269	6	236,427	6	Sav
H2	68	66	214,908	5	**213,732**	5	Sav
H3/H4	68	66	202,971	4	204,794	4	both
H5/H6	68	66	201,896	4	**196,446**	4	Sav
I1	90	50	320,703	9	**320,217**	10	Sav
I2	90	50	272,621	7	276,519	7	Sav
I3/I4/I5	90	50	238,245	5	**237,662**	5	Sw
J1	94	80	330,235	10	**324,265**	10	Sav
J2	94	80	292,698	8	294,004	8	Sav
J3	94	80	249,931	6	255,195	6	Sav
J4	94	80	257,895	6	275,311	7	Sav
K1	113	66	352,253	10	352,729	10	Sw
K2/K3	113	66	317,004	8	317,274	8	Sav
K4	113	66	294,848	7	**293,621**	7	Sw
L1	150	50	394,414	10	395,803	11	Sav
L2/L3	150	50	360,018	8	365,189	9	Sw
L4/L5	150	50	337,620	7	**335,186**	7	Sav
M1/M2	125	80	360,897	10	366,426	10	Sav
M3	125	80	335,486	9	336,753	9	Sw
M4	125	80	300,225	7	305,428	7	Sav
N1/N2	150	66	370,690	10	**365,724**	10	Sav
N3/N4	150	66	349,516	9	352,064	9	Sav
N5/N6	150	66	319,811	7	**315,475**	7	Sav

Bold values indicate a better solution than the best known solution

Table 14 Results RTS for TV97 instance set

Instance			BKS		RTS-comb		
Name	n	LH (%)	TTD	v	TTD-best	v-best	init
EIL22_2	21	50	324	3	326	3	Both
EIL22_3	21	60	341	3	342	3	Both
EIL22_5	21	80	341	3	342	3	Both
EIL23_2	22	50	526	2	527	2	Both
EIL23_3	22	60	526	2	527	2	Both
EIL23_5	22	80	514	2	514	2	Both
EIL30_2	29	50	417	2	419	2	Both
EIL30_3	29	60	475	3	484	3	Sav
EIL30_5	29	80	475	3	478	3	Sw
EIL33_2	32	50	680	3	681	3	Sw
EIL33_3	32	60	680	3	681	3	Both
EIL33_5	32	80	686	3	687	3	Sw
EIL51_2	50	50	466	3	**465**	3	Sw
EIL51_3	50	60	491	4	491	4	Sw
EIL51_5	50	80	497	4	497	4	Sav
EIL76a_2	75	50	670	6	**666**	6	Sav
EIL76a_3	75	60	719	7	**704**	7	Sav
EIL76a_5	75	80	759	8	760	9	Sav
EIL76b_2	75	50	768	8	**757**	8	Sav
EIL76b_3	75	60	826	10	831	10	Sav
EIL76b_5	75	80	904	12	**897**	12	Sw
EIL76c_2	75	50	629	5	**626**	5	Sw
EIL76c_3	75	60	663	6	**660**	6	Sw
EIL76c_5	75	80	697	7	**695**	7	Sav
EIL76d_2	75	50	608	4	**602**	4	Sw
EIL76d_3	75	60	627	5	627	5	Sw
EIL76d_5	75	80	653	6	**646**	6	Sav
EIL101a_2	100	50	730	5	739	4	Sw
EIL101a_3	100	60	754	6	756	6	Sw
EIL101a_5	100	80	789	7	795	7	Sav
EIL101b_2	100	50	861	8	**854**	8	Sw
EIL101b_3	100	60	923	10	931	10	Sw
EIL101b_5	100	80	969	11	**966**	11	Sav

Bold values indicate a better solution than the best known solution

Table 15 Results RTS for SN99 instance set

Instance			BKS		RTS-comb		
Name	n	$LH(\%)$	TTD	v	TTD-best	v-best	init
CMT01H	50	50	462	3	466	3	Sw
CMT01Q	50	75	490	4	490	4	Sw
CMT01T	50	90	520	5	520	5	Sw
CMT02H	75	50	661	6	666	6	Sw
CMT02Q	75	75	732	8	733	8	Sw
CMT02T	75	90	783	9	785	9	Sw
CMT03H	100	50	701	5	723	5	Sw
CMT03Q	100	75	747	6	754	6	Sav
CMT03T	100	90	798	7	802	7	Sav
CMT04H	150	50	829	7	870	7	Sav
CMT04Q	150	75	915	9	932	9	Sav
CMT04T	150	90	993	11	1023	11	Sav
CMT05H	199	50	983	20	1041	10	Sav
CMT05Q	199	75	1118	20	1157	13	Sav
CMT05T	199	90	1278	15	1280	16	Sw
CMT11H	120	50	818	4	847	4	Sav
CMT11Q	120	75	939	6	944	6	Sav
CMT11T	120	90	999	7	1007	7	Sav
CMT12H	100	50	629	5	638	5	Sw
CMT12Q	100	75	729	7	749	8	Sav
CMT12T	100	90	788	9	788	9	Sav

References

Avci, M., & Topaloglu, S. (2015). An adaptive local search algorithm for vehicle routing problem with simultaneous and mixed pickups and deliveries. *Computers & Industrial Engineering*, *83*, 15–29. https://doi.org/10.1016/j.cie.2015.02.002.

Baker, B. S., Coffman, J. E. G., & Rivest, R. L. (1980). Orthogonal packings in two dimensions. *SIAM Journal on Computing*, *9*(4), 846–855. https://doi.org/10.1137/0209064.

Bartók, T., & Imreh, C. (2011). Pickup and delivery vehicle routing with multidimensional loading constraints. *Acta Cybernetica*, *20*(1), 17–33. https://doi.org/10.14232/actacyb.20.1.2011.3.

Bortfeldt, A. (2012). A hybrid algorithm for the capacitated vehicle routing problem with three-dimensional loading constraints. *Computers & Operations Research*, *39*(9), 2248–2257. https://doi.org/10.1016/j.cor.2011.11.008.

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*(1), 82–96. https://doi.org/10.1016/j.ejor.2014.12.001.

Bortfeldt, A., & Wäscher, G. (2013). Constraints in container loading—A state-of-the-art review. *European Journal of Operational Research*, *229*(1), 1–20. https://doi.org/10.1016/j.ejor.2012.12.006.

Casco, D. O., Golden, B. L., & Wasil, E. A. (1988). Vehicle routing, studies in management science and systems. In B. L. Golden (Ed.), *Vehicle routing with backhauls: Models, algorithms, and case studies* (pp. 127–147). Amsterdam: North-Holland.

Clarke, G., & Wright, J. W. (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, *12*(4), 568–581.

Crispim, J., & Brandão, J. (2005). Metaheuristics applied to mixed and simultaneous extensions of vehicle routing problems with backhauls. *The Journal of the Operational Research Society*, *56*(11), 1296–1302.

de Angelis, L. (2011). A fall in average vehicle loads: Average loads, distances and empty running in road freight transport-2010. Statistics in Focus 63.

Dominguez O, Guimarans D, Juan AA (2015) A hybrid heuristic for the 2l-vrp with clustered backhauls. In *Proceedings of the XVI Conferencia de la Asociación Española para la Inteligencia Artificial (CAEPIA)*.

Escobar-Falcón LM, Álvarez-Martínez D, Granada-Echeverri M, Escobar JW, Romero-Lázaro RA (2016) A matheuristic algorithm for the three-dimensional loading capacitated vehicle routing problem (3l-cvrp). Revista Facultad de Ingeniería Universidad de Antioquia (Vol. 78, pp. 9–20). https://doi.org/10.17533/udea.redin.n78a02.

Fuellerer, G., Doerner, K. F., Hartl, R. F., & Iori, M. (2010). Metaheuristics for vehicle routing problems with three-dimensional loading constraints. *European Journal of Operational Research*, *201*(3), 751–759. https://doi.org/10.1016/j.ejor.2009.03.046.

García-Nájera, A., Bullinaria, J. A., & Gutiérrez-Andrade, M. A. (2015). An evolutionary approach for multi-objective vehicle routing problems with backhauls. *Computers & Industrial Engineering*, *81*, 90–108. https://doi.org/10.1016/j.cie.2014.12.029.

Gendreau, M., Iori, M., Laporte, G., & Martello, S. (2006). A tabu search algorithm for a routing and container loading problem. *Transportation Science*, *40*(3), 342–350. https://doi.org/10.1287/trsc.1050.0145.

Gillett, B. E., & Miller, L. R. (1974). A heuristic algorithm for the vehicle-dispatch problem. *Operations Research*, *22*(2), 340–349. https://doi.org/10.1287/opre.22.2.340.

Goetschalckx, M., & Jacobs-Blecha, C. (1989). The vehicle routing problem with backhauls. *European Journal of Operational Research*, *42*(1), 39–51. https://doi.org/10.1016/0377-2217(89)90057-X.

Golden, B. L., Baker, E., Alfaro, J., Schaffer, J. (1985). The vehicle routing problem with backhauling: Two approaches. In: R. D. Hammesfahr (Eds.), Proceedings of the twenty-first annual meeting of the S.E. TIMS, Myrtle Beach, SC, USA.

Halse K (1992) Modeling and solving complex vehicle routing problems. Ph. d. thesis, Technical University of Denmark, Lyngby.

Hopper E (2000) Two-dimensional packing utilising evolutionary algorithms and other meta-heuristic methods. Ph.D. thesis, University of Wales, Cardiff.

Irnich, S., Toth, P., & Vigo, D. (2014). The family of vehicle routing problems. In P. Toth & D. Vigo (Eds.), *Vehicle routing, MOS-SIAM series on optimization* (pp. 1–33). Philadelphia: SIAM.

Karabulut K, Inceoglu MM (2005) A hybrid genetic algorithm for packing in 3d with deepest bottom left with fill method. In: Yakhno T (ed) Advances in Information Systems, Lecture Notes in Computer Science (Vol. 3261, pp. 441–450). Springer, Berlin. https://doi.org/10.1007/978-3-540-30198-1_45.

Lodi, A., Martello, S., & Vigo, D. (1999). Heuristic and metaheuristic approaches for a class of two-dimensional bin packing problems. *INFORMS Journal on Computing*, *11*(4), 345–357. https://doi.org/10.1287/ijoc.11.4.345.

Ma Hw, Zhu W, Xu S (2011) Research on the algorithm for 3l-cvrp with considering the utilization rate of vehicles. In: Chen R (ed) Intelligent Computing and Information Science, Communications in Computer and Information Science (Vol. 134, pp. 621–629). Springer, Berlin. https://doi.org/10.1007/978-3-642-18129-0_94.

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. https://doi.org/10.1016/j.ejor.2016.04.016.

Miao, L., Ruan, Q., Woghiren, K., & Ruo, Q. (2012). A hybrid genetic algorithm for the vehicle routing problem with three-dimensional loading constraints. *RAIRO—Operations Research*, *46*(1), 63–82. https://doi.org/10.1051/ro/2012008.

Moura A (2008) A multi-objective genetic algorithm for the vehicle routing with time windows and loading problem. In: Bortfeldt A, Homberger J, Kopfer H, Pankratz G, Strangmeier R (eds) Intelligent Decision Support, Gabler Edition Wissenschaft, Betriebswirtschaftlicher Verlag Dr. Th. Gabler/GWV Fachverlage GmbH Wiesbaden, Wiesbaden (pp. 187–201). https://doi.org/10.1007/978-3-8349-9777-7_11.

Moura, A., & Oliveira, J. F. (2009). An integrated approach to the vehicle routing and container loading problems. *OR Spektrum*, *31*(4), 775–800. https://doi.org/10.1007/s00291-008-0129-4.

Nagy, G., Wassan, N. A., & Salhi, S. (2013). The vehicle routing problem with restricted mixing of deliveries and pickups. *Journal of Scheduling*, *16*(2), 199–213. https://doi.org/10.1007/s10951-012-0296-7.

Paquay, C., Limbourg, S., Schyns, M., & Oliveira, J. F. (2017). Mip-based constructive heuristics for the three-dimensional bin packing problem with transportation constraints. *International Journal of Production Research*, *56*(4), 1581–1592. https://doi.org/10.1080/00207543.2017.1355577.

Parragh, S. N., Doerner, K. F., & Hartl, R. F. (2008). A survey on pickup and delivery problems. *Journal für Betriebswirtschaft*, *58*(1), 21–51. https://doi.org/10.1007/s11301-008-0033-7.

Pinto, T., Alves, C., De, C., & Moura, A. (2015). An insertion heuristic for the capacitated vehicle routing problem with loading con-

straints and mixed linehauls and backhauls. *FME Transaction*, *43*(4), 311–318. https://doi.org/10.5937/fmet1504311P.

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 Spektrum*, *37*(2), 297–330. https://doi.org/10.1007/s00291-014-0386-3.

Reil, S., Bortfeldt, A., & Mönch, L. (2018). Heuristics for vehicle routing problems with backhauls, time windows, and 3D loading constraints. *European Journal of Operational Research*, *266*(3), 877–894. https://doi.org/10.1016/j.ejor.2017.10.029.

Ropke, S., & Pisinger, D. (2006). A unified heuristic for a large class of vehicle routing problems with backhauls. *European Journal of Operational Research*, *171*(3), 750–775. https://doi.org/10.1016/j.ejor.2004.09.004.

Salhi, S., & Nagy, G. (1999). A cluster insertion heuristic for single and multiple depot vehicle routing problems with backhauling. *The Journal of the Operational Research Society*, *50*(10), 1034. https://doi.org/10.2307/3009928.

Tao, Y., & Wang, F. (2015). An effective tabu search approach with improved loading algorithms for the 3L-CVRP. *Computers & Operations Research*, *55*, 127–140. https://doi.org/10.1016/j.cor.2013.10.017.

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*(2), 255–271. https://doi.org/10.1109/TITS.2009.2020187.

Toth, P., & Vigo, D. (1997). An exact algorithm for the vehicle routing problem with backhauls. *Transportation Science*, *31*(4), 372–385. https://doi.org/10.1287/trsc.31.4.372.

Toth, P., & Vigo, D. (Eds.). (2014). *Vehicle routing: Problems, methods, and applications, MOS-SIAM series on optimization*, (2nd ed., Vol. 18). Philadelphia: SIAM.

Wade AC (2002) Constructive and ant system heuristics for a class of vehicle routing problem with backhauls. Ph.D. thesis, University of Birmingham.

Wade, A., & Salhi, S. (2004). An ant system algorithm for the mixed vehicle routing problem with backhauls. *Metaheuristics: Computer decision-making* (pp. 699–719). Boston: Springer. https://doi.org/10.1007/978-1-4757-4137-7_33.

Wang L, Guo S, Chen S, Zhu W, Lim A (2010) Two natural heuristics for 3d packing with practical loading constraints. In: Zhang BT, Orgun MA (eds) PRICAI 2010, Lecture notes in computer science Lecture notes in artificial intelligence (Vol. 6230, pp. 256–267). Springer, Berlin. https://doi.org/10.1007/978-3-642-15246-7_25

Wassan, N. A., Nagy, G., & Ahmadi, S. (2008). A heuristic method for the vehicle routing problem with mixed deliveries and pickups. *Journal of Scheduling*, *11*(2), 149–161. https://doi.org/10.1007/s10951-008-0055-y.

Wassan, N. A., Salhi, S., Nagy, G., Wassan, N., & Wade, A. C. (2013). Solving the mixed backhauling vehicle routing problem with ants. *International Journal of Energy Optimization and Engineering*, *2*(2), 62–77. https://doi.org/10.4018/ijeoe.2013040105.

Wei, L., Zhang, Z., & Lim, A. (2014). An adaptive variable neighborhood search for a heterogeneous fleet vehicle routing problem with three-dimensional loading constraints. *IEEE Computational Intelligence Magazine*, *9*(4), 18–30. https://doi.org/10.1109/MCI.2014.2350933.

Wisniewski MA, Ritt M, Buriol LS (2011) A tabu search algorithm for the capacitated vehicle routing problem with three-dimensional loading constraints. In *XLIII Simposio Brasilero de Pesquisa Operacional*.

Zachariadis, E. E., Tarantilis, C. D., & Kiranoudis, C. T. (2016). The vehicle routing problem with simultaneous pick-ups and deliveries and two-dimensional loading constraints. *European Journal of*

Operational Research, *251*(2), 369–386. https://doi.org/10.1016/j.ejor.2015.11.018.

Zhang, Z., Wei, L., & Lim, A. (2015). An evolutionary local search for the capacitated vehicle routing problem minimizing fuel consumption under three-dimensional loading constraints. *Transportation Research Part B: Methodological*, *82*, 20–35. https://doi.org/10.1016/j.trb.2015.10.001.

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*(9), 2178–2195. https://doi.org/10.1016/j.cor.2011.11.001.