WORKING PAPER SERIES

Hybrid Algorithms for the Vehicle Routing Problem with Pickup and Delivery and Two-dimensional Loading Constraints

Dirk Männel

Working Paper No. 10/2017

Impressum (§ 5 TMG)

Herausgeber:
Otto-von-Guericke-Universität Magdeburg
Fakultät für Wirtschaftswissenschaft
Der Dekan

Verantwortlich für diese Ausgabe:
Dirk Männel
Otto-von-Guericke-Universität Magdeburg
Fakultät für Wirtschaftswissenschaft
Postfach 4120
39016 Magdeburg
Germany

http://www.fww.ovgu.de/femm

Bezug über den Herausgeber
ISSN 1615-4274

Hybrid Algorithms for the Vehicle Routing Problem with Pickup and Delivery and Two-dimensional Loading Constraints

Dirk Männel
Otto von Guericke University, Universitätsplatz 2, 39106 Magdeburg, Germany
dirk.maennel@gmx.de
Phone: 0049 2054 8549220

Abstract

We extend the classical Pickup and Delivery Problem (PDP) to an integrated routing and two-dimensional loading problem, called PDP with two-dimensional loading constraints (2L-PDP). A set of routes of minimum total length has to be determined such that each request is transported from a loading site to the corresponding unloading site. Each request consists of a given set of 2D rectangular items with a certain weight. The vehicles have a weight capacity and a rectangular two-dimensional loading area. All loading and unloading operations must be done exclusively by movements parallel to the longitudinal axis of the loading area of a vehicle and without moving items of other requests. Furthermore, each item must not be moved after loading and before unloading.

The problem is of interest for the transport of rectangular-shaped items that cannot be stacked one on top of the other because of their weight, fragility or large dimensions. The 2L-PDP also generalizes the well-known Capacitated Vehicle Routing Problem with Two-dimensional Loading Constraints (2L-CVRP), in which the demand of each customer is to be transported from the depot to the customer's unloading site.

This paper proposes two hybrid algorithms for solving the 2L-PDP and each one consists of a routing and a packing procedure. Within both approaches, the routing procedure modifies a well-known large neighborhood search for the one-dimensional PDP and the packing procedure uses six different constructive heuristics for packing the items. Computational experiments were carried out using 60 newly proposed 2L-PDP benchmark instances with up to 150 requests.

Key words: Transportation, vehicle routing, packing, pickup and delivery.

1 Introduction

Vehicle routing problems widely arise in transportation logistics if companies are interested in optimizing their routes. Therefore problems like the classical capacitated vehicle routing problem (CVRP) and the classical pickup and delivery problem (PDP) have been investigated in the literature for many years. However the classical modeling does not consider constraints occurring in real world settings regarding the feasibility of the loading. To ensure that calculated routes can actually be implemented, a two-dimensional (2D) or three-dimensional (3D) modeling of cargo and loading spaces is indispensable in many situations. Therefore in the last ten years a good deal of research has been done on integrated routing problems with 2D or 3D loading constraints. Several packing constraints, e.g. concerning stacking of goods, can only be considered if customer demands are viewed as sets of 3D items. At the same time, often any reloading effort should be avoided that is any temporary or permanent repositioning or rotation of items after loading and before unloading. There are different practical reasons to forbid reloading of goods during a pickup and delivery route. Absence of manpower, tight working time, lack of equipment and shortage of space at customer sites are some of them. Moreover, the goods might be fragile, extra heavy or even hazardous. A 2D modeling instead of a 3D modeling is sufficient if the goods to be transported can be considered as rectangular items that cannot be stacked due to their weight, dimensions or fragility. Such issues arise in industries where large-sized items have to be transported, e.g. furniture, mechanical components and household appliances.

In the following, we consider the pickup and delivery problem with two-dimensional loading constraints (2L-PDP). As in the classical PDP, a number of requests have to be transported from a pickup point to a delivery point by means of homogeneous vehicles. However, in the 2L-PDP the demands consist of sets of 2D items to be placed on 2D loading areas of the vehicles. All vehicles are assumed as rear-loaded, i.e. the goods are loaded and unloaded at the rear exclusively by movements in length direction of the vehicle, while moving them in width direction is not permitted in the loading or unloading operation. Moreover, we assume that any reloading of items after loading and before unloading is not allowed.

Two hybrid algorithms for solving the 2L-PDP are proposed that consist of a routing and a packing procedure. Within both approaches, the routing procedure modifies a well-known large neighborhood search for the one-dimensional PDP and the packing procedure uses six different constructive heuristics for packing the items. Computational experiments are carried out using 60 newly proposed 2L-PDP benchmark instances with up to 150 requests.

The rest of the paper is organized as follows: In Section 2, the relevant literature is reviewed, and the problem is formulated in Section 3. Two solution approaches are described in Section 4. Computational experiments are reported in Section 5. Conclusions are drawn and an outlook to further research is given in Section 6.

2 Related work

Up to now the 2L-PDP was only considered by Malapert et al. (2008). They proposed a constraint programming approach for the loading aspects of the problem but did not report any numerical results. Therefore, we will focus on recent papers on the classical PDP with paired pickup and delivery points and on VRPs with 2D and 3D loading constraints. We refer the reader to Toth and Vigo (2014) for a comprehensive survey on vehicle routing. Although this paper only covers two-dimensional loading constraints, we want to consider papers on vehicle routing problems with 3D loading constraints in the literature review, too. Recent surveys of integrated vehicle routing problems with 2D and 3D loading constraints were published by Iori and Martello (2010, 2013) and Pollaris et al. (2015).

Following the classification schema by Parragh et al. (2008), the classical PDP is characterized by paired pickup and delivery points, i.e. each request is associated with a special pickup and a special delivery point. Moreover, the PDP deals with the transportation of goods and persons. In case of passenger transportation, there are often special constraints and objectives concerning the inconvenience of passengers. This problem category is known as dial-a-ride problems. A further distinction can be made with regard to the number of available vehicles. We will consider only the multi-vehicle case, while the single-vehicle case, representing an immediate extension of the Traveling Salesman Problem (TSP), is not considered here. Furthermore, several papers deal with PDPs with multiple depots or a heterogeneous fleet where a certain request can only be served with a subset of the available vehicles (see below).

The problem formulation for the classical PDP and the PDP with time windows (PDPTW) can be found, e.g. in Parragh et al. (2008) and in Toth and Vigo (2014). Most of the published solution methods are surveyed by Berbeglia et al. (2007) and Parragh et al. (2008). As the PDP generalizes the TSP it is NP-hard. Thus most papers have proposed heuristics and especially metaheuristics for solving the PDP. For an introduction in metaheuristic approaches, we refer the reader to Gendreau and Potvin (2010).

The classical PDP with time windows and multiple vehicles was first solved by Nanry and Barnes (2000) with a reactive tabu search approach. Mostly the minimization of the number of needed vehicles is used as first optimization criterion while the minimization of the total travel distance is the second criterion. A tabu embedded simulated annealing approach has been developed by Li and Lim (2001). These authors also have introduced the widely used Li-and-Lim set of benchmark instances for the PDPTW. Pankratz (2005) proposed a grouping genetic algorithm for the PDP while Lu and Dessouky (2006) have developed an ingenious construction heuristic. Ropke and Pisinger (2006) presented an adaptive large neighborhood search algorithm for the PDPTW which covers also multiple depots and heterogeneous fleets. A two-stage hybrid algorithm was presented by Bent and van Hentenryck (2006). The first phase uses simulated annealing to decrease the number of vehicles needed. The second phase consists of a large neighborhood search algorithm in order to reduce total travel cost. Nagata and Kobayashi (2010) introduced a very effective guided ejection search algorithm to reduce the number of vehicles needed. The minimization of the total travel distance was not considered in their approach. Outstanding results, especially for larger instances, were achieved through the neighborhood search methods by Bent and van Hentenryck (2006), by Ropke and Pisinger (2006) and by Nagata and Kobayashi (2010).

In the capacitated vehicle routing problem with 2D loading constraints (2L-CVRP) the requests consist of 2D rectangular items to be transported. The vehicles have a rectangular loading area where the items must be placed without overlapping. Furthermore, some additional constraints like LIFO constraint (Last In, First Out) and orientation constraint (see Section 3) are to be taken into account. Several metaheuristic methods for solving the 2L-CVRP were published, e.g. by Gendreau et al. (2008), Fuellerer et al. (2009), Zachariadis et al. (2009), Duhamel et al. (2011) and Wei et al. (2015). Iori et al. (2007) proposed an exact solution approach for the 2L-CVRP. Extensions of the 2L-CVRP were also considered in the literature, e.g. the 2L-VRP with time windows (Khebbache-Hadji et al., 2013), the 2L-VRP with heterogeneous fleet (Leung et al., 2013), the 2L-VRP with backhauls

(Dominguez et al., 2016) and the 2L-VRP with simultaneous pickup and delivery (Zachariadis et al., 2016).

In vehicle routing problems with three dimensional loading constraints, the items are stackable 3D rectangular boxes which must be placed inside the 3D loading space of a vehicle. Additional constraints are to observe in the 3D case, e.g. the stacking constraint (non-fragile boxes must not be placed above fragile boxes) and the support constraint (at least a given percentage of the base area of a box must be supported by other boxes if the box is not placed on the floor of a loading space). The 3L-CVRP was introduced and first solved by Gendreau et al. (2006). Further papers on 3L-CVRP were published, for example, by Tarantilis et al. (2009), Fuellerer et al. (2010), Wang et al. (2010), Wisniewski et al. (2011), Bortfeldt (2012), Zhu et al. (2012), Ruan et al. (2013), Wei et al. (2014), Zhang et al. (2015) and Tao and Wang (2015). Moura and Oliveira (2009) have first introduced and solved the VRP with time windows and 3D loading constraints (3L-VRPTW). A hybrid algorithm for solving the 3L-VRPTW was published by Bortfeldt and Homberger (2013). Zachariadis et al. (2012) consider a 3L-VRP with time windows where boxes are stacked on pallets, which in turn are loaded in vehicles. The 3L-VRP with backhauls was introduced by Bortfeldt et al. (2015). The problem was solved with an algorithm including a neighbourhood search algorithm for routing and a tree search algorithm for packing boxes. The 3L-VRP with pickup and delivery was solved with a similar algorithm by Männel and Bortfeldt (2016, 2017). The 3L-VRP with pickup and deliveries was already considered by Bartók and Imreh (2011) in a simpler fashion. These authors neglected the LIFO constraint and did not provide any numerical results.

3 Problem definition

Now the 2L-PDP is described more formally. There are given n requests each consisting of a pickup point i, a delivery point $n+i$ and a set I_i of goods that are to be transported from i to $n+i$ ($i = 1,\dots,n$). There are v_{max} identical vehicles, originally located at the single depot (denoted by 0), with a rectangular loading area with length L and width W and maximum weight capacity D. Let $V = \{0,1,\dots,n,n+1,\dots,2n\}$ be the set of all nodes, i.e. pickup and delivery points including the depot. Let E be a set of undirected edges (i,j) that connect all node pairs ($0 \le i, j \le 2n$, $i \ne j$) and let $G = (V, E)$ be the resulting graph. Let travel costs c_{ij} ($c_{ij} \ge 0$) be assigned to each edge (i,j) and let the travel costs be symmetric, i.e. $c_{ij} = c_{ji}$ ($0 \le i, j \le 2n$, $i \ne j$). The sets I_i include m_i rectangular items I_{ik} and item I_{ik} has the length l_{ik} and the width w_{ik} ($i = 1,\dots,n$, $k = 1,\dots,m_i$). m is the sum $\sum m_i$ ($i = 1,\dots,n$) and denotes the total number of items.

The loading area of each vehicle is embedded in the first quadrant of a Cartesian coordinate system in such a way that the length and width of the loading area lie parallel to the x and y axis, respectively. The placement of item I_{ik} in a loading area is given by the coordinates x_{ik} and y_{ik} of the corner of the item closest to the origin of the coordinate system; in addition, a binary variable o_{ik} indicates which of the possible orientations of item I_{ik} is selected ($i = 1,...,n$, $k = 1,...,m_i$). o_{ik}=0 means that the item is placed with its length parallel to the x-axis, while o_{ik}=1 indicates that the item is rotated by 90° and its length is parallel to the y-axis.

A packing plan P for a loading area comprises one or more placements and is regarded as feasible if the following conditions hold:

(FP1) each placed item lies completely within the loading area,

(FP2) any two items that are placed on the same truck loading area do not overlap,

(FP3) each placed items lies with its edges parallel to the edges of the loading area.

Figure 1 shows a loading area with placed items. Each vehicle is loaded and unloaded at the rear and empty at the beginning of a route.

A feasible route R is a sequence of $2p+2$ nodes ($p \geq 1$) that starts and ends at the depot. R should include the pickup and delivery points of p different (among the n given) requests and each pickup point must precede the delivery point of the same request. A solution of the 2L-PDP is a set of v sequences (R_l, $P_{l,1},\ldots,P_{l,2p_l}$), where R_l is a route and $P_{l,q}$ is a packing plan ($l = 1,\ldots,v$, $q = 1,\ldots,2p_l$, p_l denotes the number of requests of route l). $P_{l,q}$ represents the packing plan of route l after having visited its $(q+1)th$ node, i.e. after some items were loaded or unloaded at the $(q+1)th$ node of route l.

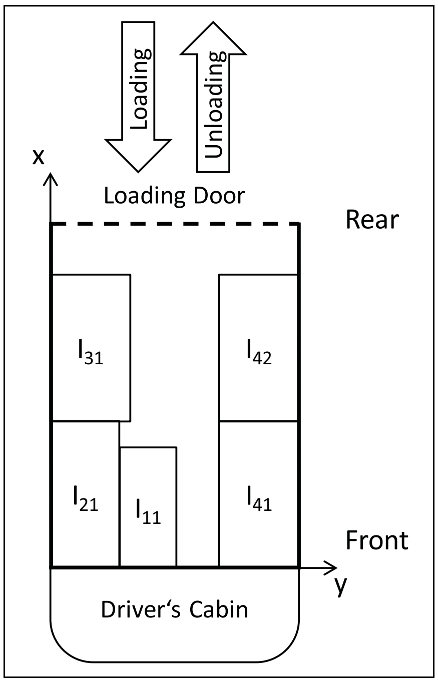

Figure 1: A loading area with placed items.

To be feasible, a solution must fulfill the following six conditions:

(F1) each route R_l starts and ends at the depot and contains at least one pickup and one delivery point ($l = 1,\ldots,v$),

(F2) each pickup point and each delivery point must occur exactly once in exactly one route,

(F3) the pickup point and the delivery point of each request lie in the same route,

(F4) each the pickup point occurs in its route before the corresponding delivery point, (F5) all packing plans $P_{l,q}$ are feasible ($l = 1,\ldots,v$, $q = 1,\ldots,2p_l$), i.e. fulfill conditions (FP1) – (FP3),

(F6) the packing plan $P_{l,q}$ for a route R_l and its $(q+1)th$ node contains exactly the placements for those items which are to be loaded but not (yet) to be unloaded at the first $q+1$ nodes of the route.

In addition, the following routing and packing constraints are to be satisfied:

(C1) LIFO *constraint for pickup points*: A packed item i of a certain request is said to be in unloading position if there is no packed item i' of another request placed between i and the rear of the vehicle. If the $(q+1)th$ node of route l is a pickup point, then all items to be loaded there must be in unloading position in the packing plan $P_{l,q}$, i.e. after loading ($l = 1,\ldots,v$, $q = 1,\ldots,2p_l$).

(C2) LIFO *constraint for delivery points*: If the $(q+1)th$ node is a delivery point, then all items to be unloaded there must be in unloading position in the packing plan $P_{l,q-1}$, i.e. before unloading ($l = 1,\ldots,v$, $q = 1,\ldots,2p_l$). Both LIFO constraints ensure that all items of a given request can be loaded or unloaded exclusively by movements parallel to the longitudinal axis of the loading area of a vehicle and without moving items of other requests.

(C3) *Reloading ban*: Each item I_{ik} of request i must not be moved after loading and before unloading ($i = 1,\ldots,n$, $k = 1,\ldots,m_i$). If the item I_{ik} is loaded at the $(q+1)th$ node and unloaded at the $(q'+1)th$ node of route l, its placement (x_{ik}, y_{ik}, o_{ik}) must be the same in the packing plans $P_{l,q}$, $P_{l,q+1},\ldots$, $P_{l,q'-1}$ ($i = 1,\ldots,n$, $k = 1,\ldots,m_i$, $l = 1,\ldots,v$, $1 \le q < q' \le 2p_l$).

(C4) *Weight constraint*: Each item I_{ik} has a positive weight d_{ik} ($i = 1,\ldots,n$, $k = 1,\ldots,m_i$) and the total weight of all items in a packing plan $P_{l,q}$ must not exceed a maximum weight capacity D ($l = 1,\ldots,v$, $q = 1,\ldots,2p_l$).

(C5) *Route length constraint*: The total distance of a route must not exceed a specified maximum d_{max}. This constraint can also be understood as a route duration constraint if the vehicle velocity is set to a constant.

(C6) *Route number constraint*: The number of routes v must not exceed the number of vehicles v_{max}.

Finally, the 2L-PDP consists of determining a feasible solution that meets the constraints (C1) – (C6) and minimizes the total travel distance of all routes.

The LIFO constraint for delivery points (C2) is well-known from the 2L-CVRP. At a delivery point, the LIFO constraint requires that between an item A to be unloaded and the rear of the vehicle no item B is situated that needs to be unloaded later. Otherwise item B has to be reloaded before item A can be unloaded by a pure movement in length direction. In Figure 1, the item I_{31} is in unloading position while the items I_{11} and I_{21} are not in unloading position because of the blocking item I_{31}. Both items I_{41} and I_{42} are in unloading position because they belong to the same request.

As also pickup points occur in a pickup and delivery route, a LIFO constraint to exclude reloading of goods at pickup points (C1) has to be included. At a pickup point the constraint (C1) requires that between the position of an item A just loaded and the rear of the vehicle no item B is situated that was loaded at an earlier pickup point. Again, otherwise a reloading of item B would be inevitable.

It is an essential feature of 2L-PDP that the LIFO constraints for delivery and pickup points are not sufficient to rule out any reloading effort. Furthermore, the reloading ban constraint (C3) has to be required to rule out any reloading effort. Figure 2 shows a simple example of a route and with three requests and one item per request. There exist feasible packings plans for both nodes P2 (items of request 1 and 2 loaded) and P3 (items of request 1 and 3 loaded). The packing plans fulfill both LIFO constraints (C1) and (C2), but the reloading ban (C3) is violated because box I_{11} is rotated in the second packing plan. In the example it is obviously impossible to implement the shown route without reloading box I_{11}. Hence the LIFO constraints (C1) and (C2) alone are not sufficient to rule out any reloading effort and the reloading ban constraint (C3) turns out to be necessary.

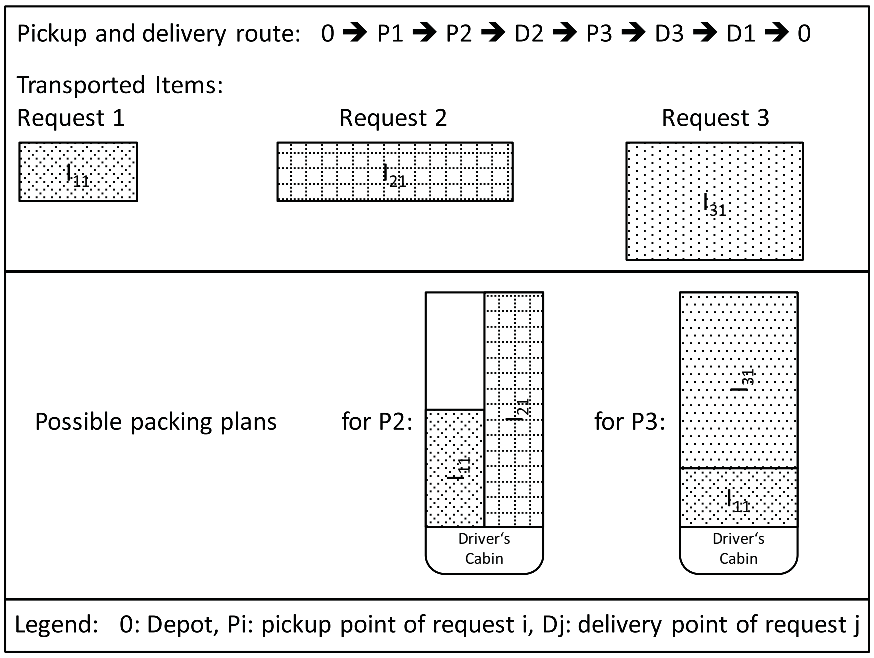
Pickup and delivery route: 0 → P1 → P2 → D2 → P3 → D3 → D1 → 0
Transported Items:
Request 1
Request 2
Request 3
I11
I21
I31
Possible packing plans
for P2:
for P3:
Driver's Cabin
Legend: 0: Depot, Pi: pickup point of request i, Dj: delivery point of request j

Figure 2: Packing plans with reloading for a pickup-delivery-route.

Moreover in this paper a second variant of the 2L-PDP is considered where the so-called orientation constraint (C7) is added: each placed item must lie on the loading area with its length edge parallel to the x-axis of the coordinate system (no rotation allowed). The original variant without the constraint (C7) is called in the following “Rotate” variant while the second variant is called “NoRotate” variant. In the NoRotate variant all orientation variables o_{ik} must be equal to zero in a feasible solution.

4 Two solution approaches

In this section, two solution approaches for solving the 2L-PDP are proposed. Each solution approach is a hybrid algorithm and consists of two nested procedures. The outer procedure is the routing procedure, and the packing procedure is the inner procedure. The routing procedure is basically the same for both approaches and is designed as large neighborhood search. The two approaches differ in the manner how packing checks are made and how the reloading ban constraint (C3) is taken into account.

In the first approach, the reloading ban is ensured by the routing procedure as the solution space is restricted by an additional routing constraint, the so-called independent partial route (IPR) condition. With this approach, called "Independent Partial Routes" (or IPR), it is possible to use conventional packing heuristic for packing checks. In this paper, six well-known constructive packing heuristics for the two dimensional container loading problem are integrated in a packing procedure in order to check whether a certain set of items can be packed on the loading area or not. These packing heuristics are widely used in the literature on the 2L-CVRP (see Gendreau et al., 2008, and Zachariades et al., 2009).

In the second approach ("Simultaneous Packing") the reloading ban constraint is observed by a new type of packing procedure which is able to construct a series of interrelated packing plans (see below). So the IPR condition is not needed in the second approach and this leads to a large extension of the explored solution space. Therefore, a noticeable improvement of the solution quality is expected with the Simultaneous Packing approach. However, with the larger solution space to be explored, a rising CPU-time consumption is expected, too. The main properties of both solution approaches are outlined in Table 1.

Table 1: Main properties of the two solution approaches.

Property	**Approach 1** **Independent Partial Routes**	**Approach 2** **Simultaneous Packing**
Reloading ban constraint (C3) observed by	Routing procedure	Packing procedure
Expected total travel distance	Higher	Lower
Expected CPU-power consumption	Lower	Higher

This section is organized as follows. In subsection one, the routing procedure is outlined. The second subsection presents the IPR solution approach where the 2L-PDP is solved using a straightforward packing procedure for the two-dimensional container loading problem. The implementation of the packing procedure itself is explained in subsection three. Finally, the subsection four presents the novell simultaneous packing procedure and its integration into the routing procedure.

4.1 Routing procedure

The routing procedure is derived from the (adaptive) large neighborhood search (LNS) heuristic for solving the PDP with time windows by Ropke and Pisinger (2006). In this paper a similar implementation of the routing procedure is used like in Männel and Bortfeldt (2016) before, thus the routing procedure is described in this paper only in a short fashion.

The LNS heuristics uses the „fix and optimize"-principle to construct new solutions and the neighborhood structure is defined implicitly by several removal and insert operators (heuristics). To get a new solution, first a removal operator destroys a part of the current solution, which means that some requests will be removed from their routes. Subsequently, an insert operator reinserts the removed requests at certain positions of certain routes to get a new feasible solution. The routing procedure is shown in Algorithm 1. After constructing the initial solution an iterative neighborhood search is carried out until a given time limit is exceeded. The number ξ of requests to be removed and reinserted in the solution is selected randomly within each iteration. Among the four available removal and the three available insert heuristics, one removal and one insert heuristic are selected randomly per iteration. The next solution is generated by the selected heuristics according to $s_{next} := Ih(Rh(s_{curr}, \xi))$. If s_{next} passes the acceptance test, it becomes the new current solution s_{curr}, and the best solution s_{best} is updated if s_{next} realizes a better objective function value. Otherwise, the initial solution of the next iteration s_{curr} remains unchanged. For the acceptance tests, the well-known simulated annealing rule with a geometric cooling scheme is used. The selection probabilities for the removal and insertion heuristics are fix. In the following Table 2, the available heuristics are shown.

```
2l_pdp_lns (in: problem data, parameters, out: best solution $s_{best}$)
    construct initial solution $s_{curr}$ and set $s_{best} := s_{curr}$
    while stopping criterion is not met do
            select number of requests to be removed $\xi$
            select removal heuristic $Rh$ and insertion heuristic $Ih$
            determine next solution: $s_{next} := Ih(Rh(s_{curr}, \xi))$
            check acceptance of $s_{next}$
            if $s_{next}$ is accepted then
                $s_{curr} := s_{next}$
                if $f(s_{curr}) < f(s_{best})$ then $s_{best} := s_{curr}$
    return $s_{best}$
```

Algorithm 1: LNS-based routing algorithm for the 2L-PDP.

Table 2: Removal and insertion heuristics of the LNS heuristic for 2L-PDP.

Heuristic	Description
Random removal Rh_R	Removes iteratively requests that are selected at random.
Shaw removal Rh_S	Removes iteratively requests that are related in terms of location and weight.
Worst removal Rh_W	Removes iteratively a request whose removal leads to the largest cost (total travel distance) reduction.
Tour removal Rh_T	Removes all requests from a randomly chosen route. If less than ξ requests are removed in this way, further requests will be removed with Shaw removal.
Greedy insertion Ih_G	Inserts iteratively requests into the solution such that the increase of the cost function is minimal.
Regret-2 insertion Ih_{R2}	Inserts iteratively requests into the solution such that the gap in the cost function between inserting the request into its best and its second best route is maximal.
Regret-3 insertion Ih_{R3}	Inserts iteratively requests into the solution such that the sum of two gaps in the cost function is maximal. The first gap results from inserting the request into its best and its second best route, while the second gap results from inserting the request into its best and its third best route.

4.2 IPR solution approach

In the IPR solutions approach, we want to use a conventional packing procedure for the two dimensional container loading problem to solve to the 2L-PDP. For an arbitrarily chosen route, the packing procedure has to deliver feasible packing plans for each node in the route. Each packing plan must contain exactly the items loaded on the vehicle when it is leaving the corresponding node and, furthermore, fulfill the conditions (FP1) – (FP3) and the packing related constraints (C1) – (C3) and (C7) (if necessary). In the IPR solution approach, we want to keep the packing effort low by adding a further routing constraint. This IPR constraint allows us to not to do packing checks for all nodes of a route but to restrict the packing checks to a few selected nodes within the route only. The packing plan for the other nodes can be derived from the packing plans for the selected nodes.

Definition 1: For a given node x in a 2L-PDP route, the corresponding request sequence $rs(x)$ is defined as follows. $rs(x)$ contains exactly the requests which are loaded and not yet unloaded when the vehicle is leaves the node x. The order of the requests in $rs(x)$ is given by the order of the corresponding pickup nodes within the route.

Example 1: Considering the route $0 \rightarrow P1 \rightarrow P2 \rightarrow D1 \rightarrow P3 \rightarrow P4 \rightarrow D3 \rightarrow D2 \rightarrow D4 \rightarrow 0$, the corresponding request sequences of the nodes $P4$ and $D3$ are (2, 3, 4) and (2, 4), respectively.

Definition 2:

(i) We consider a sequence $(i_1,\dots,i_s,i_{s+1},\dots,i_{2s})$ of $2s$ nodes (s > 0). This sequence is called “IPR block”, if its first s elements are pickup points and its last s elements are the *corresponding* delivery points and if, furthermore, the delivery points lie in inverse order of their corresponding pickup points. More formally, it should hold the following three conditions:

- $i_p \neq i_q$ for each $p, q \in (1,\dots,s)$ with $p \neq q$
- $1 \leq i_p \leq n$ for each $p \in (1,\dots,s)$
- $i_{2s-p+1} = i_p + n$ for each $p \in (1,\dots,s)$

(ii) A 2L-PDP route is called “IPR route” if it consists of one or more IPR blocks (plus the depot at the beginning and the end of the route).

(iii) We say that a solution of the 2L-PDP fulfills the IPR constraint if all contained routes are IPR routes.

Obviously a 2L-PDP route is an IPR route if and only if the two following conditions hold:

(1) if the vehicle visits a delivery point, then all delivery points for all items on the loading area will be visited before another pickup take place and

(2) all delivery points lie in inverse order of their corresponding pickup points, i.e. if i and j are two arbitrarily chosen requests from the route and P_i lies before P_j, then D_i must lie behind D_j in the route.

Example 2: The route $0 \rightarrow P1 \rightarrow P2 \rightarrow D2 \rightarrow P3 \rightarrow D3 \rightarrow D1 \rightarrow 0$ is not an IPR route because the vehicle does not become empty after visiting delivery point $D2$ and before visiting pickup point $P3$. The route $0 \rightarrow P3 \rightarrow P4 \rightarrow P5 \rightarrow D4 \rightarrow D5 \rightarrow D3 \rightarrow 0$ is not an IPR route because the delivery points $D4$ and $D5$ do not lie in inverse order of their corresponding pickup points. The route $0 \rightarrow P1 \rightarrow P2 \rightarrow D2 \rightarrow D1 \rightarrow P3 \rightarrow P4 \rightarrow P5 \rightarrow D5 \rightarrow D4 \rightarrow D3 \rightarrow P6 \rightarrow D6 \rightarrow 0$ is an IPR route consisting of three IPR blocks. The pickup points $P2$, $P5$ and $P6$ are called “last pickup” points because they are the last in the row of consecutive pickup points and are followed by a delivery point. Obviously, each IPR block contains exactly one last pickup point.

Definition 3: We consider an arbitrarily chosen pickup node from a 2L-PDP route with the corresponding request sequence $rs = (i_1,...,i_s)$ ($s > 0$, $1 \leq i_p \leq n$ for each $p \in (1,...,s)$). We say that a packing plan for the request sequence rs fulfills the *cumulative LIFO constraint* (CLC) if for each p and q with $1 \leq p < q \leq s$ no item of request i_p lies between an item of i_q and the rear of the vehicle, i.e. if no loading operation of an *later* loaded item of i_q is blocked by an *earlier* loaded item of i_p.

Proposition 1: Let be given an IPR route consisting of one or more IPR blocks and let exist packing plans observing conditions (FP1) – (FP3), (C7) (if necessary) and (CLC) for the request sequence of each last pickup point of each IPR block. Then feasible packing plans in terms of 2L-PDP exist for all nodes in the route which fulfill:

(i) for pickup points the conditions (FP1) – (FP3), the orientation constraint (C7) (if necessary) and the LIFO constraint for pickup points (C1),

(ii) for delivery points the conditions (FP1) – (FP3), the orientation constraint (C7) (if necessary) and the LIFO constraint for delivery points (C2) and

(iii) observe (collectively) the reloading ban constraint (C3).

Proof:

(i) In the following, the given packing plans for the last pickup points will be called “master plans”. Each master plan contains placements for all requests belonging to its corresponding IPR block. Thus for each pickup node in the route, a packing plan can be derived by simply removing placements for items not yet loaded from the corresponding master plan. Obviously, the derived plans will contain the correct items and fulfill the conditions (FP1) – (FP3) together with constraint (C7) (if necessary). Because the cumulative LIFO constraint (CLC) is stronger than the LIFO constraint for pickup points (C1) all derived plans fulfill also the constraint (C1) too.

(ii) As in the proof of (i) for each delivery node, a packing plan can derived by simply removing placements for items already unloaded from the corresponding master plan. Obviously, these derived plans again will contain the correct items and fulfill the conditions (FP1) – (FP3) together with constraint (C7) (if necessary). Finally we want to prove the LIFO constraint for delivery points (C2) indirectly and assume that (C2) would not hold. In this case would exist requests *A* and *B* belonging to the same IPR block such that delivery point D_A would lie before D_B in the route and an the unloading operation of an item *a* of request *A* would be blocked by an item *b* of request *B*. Because of the structure of the IPR blocks, the pickup point P_A would lie behind the pickup point P_B, i.e. both items *a* and *b* would be also contained in the packing plan of P_A. Since all packing plans for nodes of the same IPR block are derived from one master plan, the items *a* and *b* would hold the same positions in all plans. Hence the item *b* would block the loading operation of item *a* at pickup point P_A, which would lead to a violation of constraint (C1).

(iii) If an IPR route is given, each item must only be stowed in packing plans of one IPR block. As shown in (i) and (ii), all packing plans for the other pickup and delivery points of an IPR block will be derived from the packing plan for the last pickup point by simply removing items. Hence, the positions of all items that occur in multiple packing plans remain unchanged. □

The outcome of *Proposition 1* is, that in case of IPR routes, the cumulative LIFO constraint (CLC) is sufficient that constraints (C1) – (C3) hold. Furthermore, we show in the following *Proposition 2* that constraints (C1) and (C3) are sufficient for the (CLC) constraint to hold, i.e. in case of IPR routes (CLC) and (C1) – (C3) are equivalent. So the inclusion of the (CLC) constraint neither restricts the search space additionally, nor leads to a loss of solution quality in case of the IPR solution approach.

Proposition 2: Let be given a 2L-PDP route and let packing plans exist for all pickup points in the route. Let all these plans fulfill the LIFO constraint for delivery points (C1) and the reloading ban constraint (C3). Then all these packing plans also fulfill the cumulative LIFO constraint (CLC).

Proof: We assume that the (CLC) constraint would not hold. Then would exist requests *A*, *B* and *C* (with pickup point P_A lying before P_B and P_B before P_C) and items *a* (of *A*) and *b* (of *B*) such that item *a* would lie between item *b* and the rear of the vehicle in the packing plan for pickup point P_C. Because of the reloading ban constraint (C3), these items would hold the same placements in the packing plan for the earlier pickup point P_B too, i.e. in this packing plan item *a* would also lie between item *b* and the rear of the vehicle. Thus the LIFO constraint for pickup points (C1) would be violated at pickup point P_B. □

Finally, in this section, it is to discuss how the packing procedure will be integrated into the routing procedure. In general, the LNS heuristic removes some requests (i.e. pairs of a pickup and a delivery point) of the route and reinserts some new requests. In case of the 2L-PDP, it is impossible that a route loses their "packability" by removing requests from the route. Thus, packing checks are integrated in insertion heuristics exclusively (and not in removal heuristics). The integration of the packing procedure into the insertions heuristics takes place according to two principles. First, one-dimensional checks are made before 2D packing checks are carried out. Second, all possible insertions are first evaluated and sorted by cost *before* the "expensive" packing checks are made. By this technique, called "evaluating first, packing second", the packing effort is kept low since the packing checks can be aborted each time after a few (2D-)feasible insertions have been detected. The packing checks are made “on demand”, i.e. the packing check for a certain insert possibility is not made until all other insertion possibilities for the same request and the same route with lower insertion costs have been checked in terms of packing. Whenever for a certain pair of request and route a packable insertion possibility is found, then all other insertions possibilities for this pair of request and route with higher insertion costs can be neglected from further packing checks. In this context, “packable insertion possibility” means that the route which would result from the implementation of the insert possibility is feasible in terms of packing. For more details about the integration of the packing checks into the insertion heuristics, the reader is referred to Männel and Bortfeldt (2016).

4.3 Packing procedure for IPR solution approach

In the last section, it was shown that in case of restriction to IPR routes a conventional packing procedure is sufficient to ensure the existence of feasible packing plans for the 2L-PDP. The packing procedure must be applied only to the corresponding request sequences of the last pickup points of the routes. It has to deliver packing plans which fulfill the conditions (FP1) – (FP3), the constraint (C7) (if necessary) and the cumulative LIFO constraint (CLC). As described before, then it is ensured that packing plans for all nodes satisfying the constraints (C1) – (C3) can be derived.

In this section, the packing procedure *packing_check_rs* is introduced and it is described how the procedure performs the packing check for a certain request sequence $rs = (i_1,\ldots,i_s)$. The procedure is similar to Zachariades et al. 2009. The packing procedure uses six constructive heuristics $H_1 – H_6$ and five orderings $Ord_1 – Ord_5$ for the items shown in Table 3 and 4, respectively. To observe the cumulative LIFO constraint (CLC), the items are ordered corresponding to the position of their request in the request sequence as primary criterion. The packing procedure is shown in Algorithm 2.

Table 3: Heuristics used in the packing procedure.

Heuristic	Description	Main idea for choosing the item and the allocation point
H_1	Bottom-Left Fill (Chazelle 1983)	Minimize the allocation points x-coordinate first and y-coordinate second
H_2	Left-Bottom Fill (Chazelle 1983)	Minimize the allocation points y-coordinate first and x-coordinate second
H_3	Touching Perimeter (Lodi et al. 1999)	Maximize the sum of the items common edges with other items and the loading area edges
H_4	Touching Perimeter No Walls (Lodi et al. 1999)	Maximize the sum of the items common edges with other items
H_5	Min Area heuristic (Zachariadis et al. 2009)	Minimize the size of the allocation points corresponding rectangular surface
H_6	LBFH (Lowest Reference Line Best Fit heuristic) (Leung et al. 2011)	Uses predictive strategy with changing the placing order of items, the best fitting item for the lowest rectangular space will be chosen

Table 4: Characteristics of the orderings used in the packing procedure.

Ordering	First criterion	Second criterion	Third criterion
Ord_1	position of request in *rs*	area l*w (descending)	longer side max(l,w) (descending)
Ord_2	position of request in *rs*	width w (descending)	length l (descending)
Ord_3	position of request in *rs*	length l (descending)	width w (descending)
Ord_4	position of request in *rs*	ratio of longer to shorter side max(l,w) / min(l,w) (descending)	area l*w (descending)
Ord_5	position of request in *rs*	ratio of longer to shorter side max(l,w) / min(l,w) (ascending)	area l*w (descending)

```
packing_check_rs (in: request sequence rs, out: boolean result)
    if cache.contains(rs) then                      // if rs was already checked => take result from cache
        return cache.get-result(rs)
    for u := 1 to 5 do
        is := build-item-sequence(rs, $Ord_u$)        // build item sequence for rs using ordering $Ord_u$
        for v := 1 to 6 do
            if heuristic $H_v$ can pack item sequence is then
                cache.set-result(rs, true)          // save positive result for rs in cache and return
                return true
    cache.set-result(rs, false)                     // all (u, v)-pairs were tried without success
    return false                                    // save negative result for rs in cache and return
```

Algorithm 2: Packing procedure *packing_check_rs*.

The procedure *packing_check_rs* takes a request sequence as input and returns a boolean value (true or false) indicating whether a feasible packing plan was found or not. First, the procedure checks if the request sequence is contained in the packing cache, which means that it was already checked. In this case the result is taken from the cache and the procedure terminates. Otherwise two nested loops are executed, the outer loop iterates over the orderings and the inner loop iterates over the heuristics. In each iteration of the outer loop, first the item sequence for the current ordering is built and then up to six heuristics are applied. The procedure terminates with result “true” if one heuristic can pack the item sequence *is*, otherwise the procedure returns “false” after all 30 combinations of heuristics and orderings were tried without success. In both cases, the result is saved in the packing cache before leaving the procedure. The usage of the packing cache provides a large speedup of the algorithm because the retrieval of the check result from the cache is 100 to 1000 times faster than the repetition of the packing check with the six heuristics and five orderings.

In the following, the packing heuristic H_1 (Bottom-Left Fill) is explained in detail. Central component of this heuristic is *posList*, a set of so-called allocation points (positions where the lower left corner of new items can be placed). Initially, *posList* contains only the position (0, 0). The items will be placed “item by item”, respecting the item sequence which was created by the ordering in advance. Each time after an item was placed, the set *posList* will be updated, no more usable allocation points will be removed and new allocation points will be added. The new allocation points are so-called extreme points (see Crainic et al. 2008) generated by projection of the upper left corner of the placed item into –y direction and of the lower right corner into –x direction. The allocation points emerge where the projections “hit” the sides of already placed items or the loading area. Thereby, each projection can create more than one new allocation points. In Figure 3 is shown a loading area with some placed items. The allocation points are marked as bold circles.

In the Bottom-Left Fill heuristic the position for placing an item is selected from *posList* as follows. All allocations points in *posList* are checked if the considered item can be placed feasible at this allocation point, i.e. without overlapping or violating the cumulative LIFO loading constraint. Amongst all feasible allocation points, the one with the lowest x-coordinate is selected, ties are broken by the lowest y-coordinate. Thus, the Bottom-Left Fill heuristic tends to generate packing plans consisting of strips parallel to the y-axis. If for one item no feasible placement can be found, then the heuristic terminates without success, otherwise the heuristic terminates successfully when all items are placed. In case of the original 2L-PDP problem variant (Rotate), each allocation point is considered twice, one time for placing the item in original orientation and a second time for a placing it in rotated orientation, while in the second problem variant (NoRotate) only the original orientation is considered.

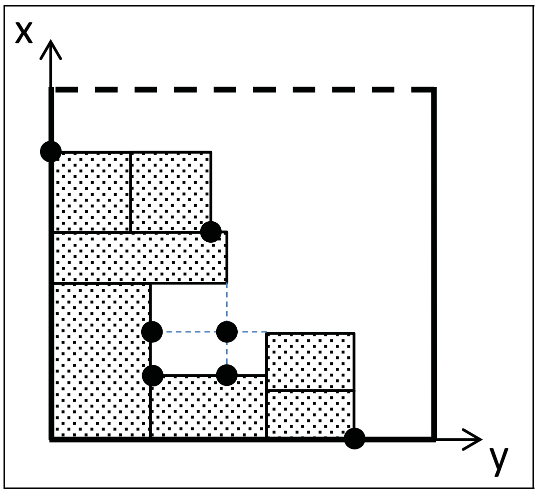

Figure 3: Allocation points marked as bold circles.

The Left-Bottom Fill heuristic H_2 works like the heuristic H_1 with the only difference, that amongst all feasible placements the allocation point with the lowest y-coordinate will be selected, ties are broken by the lowest x-coordinate. Thus, the Left-Bottom Fill heuristic tends to build packing plans made up by strips parallel to the x-axis. This approach can be useful if an extra long item must be loaded late.

In case of the Touching Perimeter heuristic H_3 for each feasible allocation point in *posList*, the total touching perimeter value of the inserted item is calculated. The touching perimeter is evaluated as the sum of the common edges of the inserted item with the edges of the already inserted items and the edges of the loading area. The item is placed at that allocation point which reaches the maximal touching perimeter value. The Touching Perimeter heuristic tends to initially place the items at the edges of the loading area and later fill the inner parts of it.

The Touching Perimeter No Walls heuristic H_4 uses the same principle like H_3, but the touching perimeter is calculated only considering common edges of the item to place with already placed items, common edges with the loading area are not taken into account. Thus, this heuristic tends to fill the inner part of the loading area earlier and cover its edges later.

In case of the Min Area heuristic H_5, the size of its corresponding rectangular surface area is calculated for each feasible allocation point. The loading position selected is the one yielding the minimum surface area. Figure 4 shows an example of a loading area with two arranged items (dotted) and three possible allocation points. The corresponding rectangular surface areas are shown as squared. The main goal of the heuristic is achieving a high degree of utilization of the vehicle's loading areas.

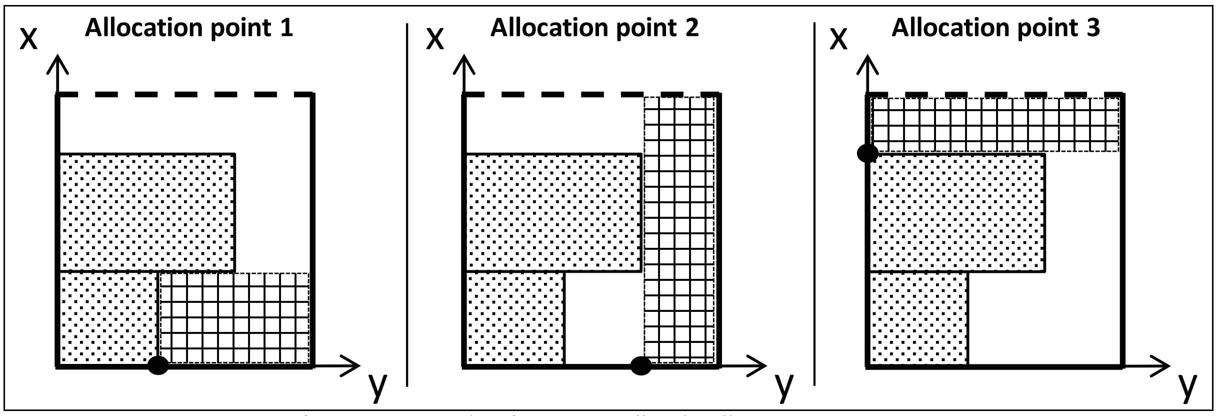

Figure 4: Example of corresponding loading areas.

The last heuristic H_6 is the LBFH heuristic (Lowest Reference Line Best Fit heuristic). The LBFH heuristic determines in each iteration first the lowest non-occupied rectangular space (with minimum x-coordinate). Then all not yet placed items of the currently processed request are tested whether they fit into the considered space. Fitting items furthermore score "fitness points" if they fill out the complete width of the considered space or if their upper edge reaches the same x-value like the adjacent items placed on the left or right of the considered space. Finally, the item with the best fitness value gets placed. Ties are broken by the items position in the sequence *is*. The main goal of the LBFH heuristic is to make the best use of the available space and reduce waste. To do so the heuristic uses a predictive strategy and changes the placing order of items belonging to the same request. For more details, the reader is referred to Leung et al. (2011).

The order of the six heuristics H_1 to H_6 is chosen so that the most simple heuristics (with smallest computational effort) Bottom-Left Fill and Left-Bottom Fill are tried first in the packing procedure *packing_check_rs*. If they fail to construct a feasible packing plan, they are followed by the more complex heuristics Touching Perimeter, Touching Perimeter No Wall, Min Area and LBFH.

4.4 Simultaneous Packing approach

In section 4.2, we introduced the IPR solution approach which heavily restricts the solution space by incorporating two additional requirements, namely (1) and (2), to the routes to satisfy the packing related constraints (C1) – (C3) and (C7) (if necessary). Now we want to introduce the Simultaneous Packing approach which drops the additional requirement (1) and allows the algorithm to explore a much larger solution space.

Definition 4:

(i) A 2L-PDP route is called "LIFO route" if the condition holds that each two delivery points in the route lie in inverse order of their corresponding pickup points. More formally, if i and j are two arbitrarily chosen requests from the route and P_i lies before P_j, then D_i must lie behind D_j in the route.

(ii) A partial route of a LIFO route is called "LIFO route block" (LRB) if the vehicle is empty when arriving at the first node and empty when leaving the last node of the partial route and if, furthermore, the vehicle does not become empty within the partial route.

(iii) We say that a solution of the 2L-PDP fulfills the LIFO route condition if all of its contained routes are LIFO routes.

Example 3: The route $0 \rightarrow P1 \rightarrow P2 \rightarrow D2 \rightarrow P3 \rightarrow P4 \rightarrow D4 \rightarrow P5 \rightarrow D5 \rightarrow D3 \rightarrow D1 \rightarrow P6 \rightarrow P7 \rightarrow D7 \rightarrow D6 \rightarrow 0$ is a LIFO route consisting of two LIFO route blocks. The first LRB starts at $P1$ and ends at $D1$, while the second LRB starts at $P6$ and ends at $D6$. The first LRB contains three last pickup points $P2$, $P4$ and $P5$ with the corresponding request sequences (1, 2), (1, 3, 4) and (1, 3, 5). The second LRB contains only one last pickup point $P7$ with the request sequence (6, 7).

In the Simultaneous Packing approach, we want to continue applying the idea of restricting the packing checks to the last pickup points within the route. The packing plans for the other nodes should be derived from the packing plans for the last pickup points. Thus it is to investigate under which additional circumstances the existence of packing plans for the last pickups in a route can ensure that feasible packing plans for all nodes of the route exist.

Proposition 3: Let be given a LIFO route consisting of one or more LIFO route blocks and let exist packing plans observing conditions (FP1) – (FP3), (C7) (if necessary) and (CLC) for the request sequences of all last pickup points of the route. Furthermore, let these packing plans fulfill the reloading ban constraint (C3), i.e. each item should hold the same placement in all plans containing this item. Then feasible packing plans in terms of 2L-PDP exist for all nodes in the route which fulfill:

(i) for pickup points the conditions (FP1) – (FP3), the orientation constraint (C7) (if necessary) and the LIFO constraint (C1),

(ii) for delivery points the conditions (FP1) – (FP3), the orientation constraint (C7) (if necessary) and the LIFO constraint (C2) and

(iii) observe (collectively) the reloading ban constraint (C3).

Proof:

(i) Each pickup point which is not a last pickup is followed by a sequence of one or more consecutive pickup points. This sequence of pickup points ends with a last pickup point which is followed by a delivery point. We call this last pickup point "corresponding last pickup point" of the original pickup point. Thus for each pickup point a packing plan can be derived by simply removing placements for items not yet loaded from the packing plan of the corresponding last pickup point. Obviously, the derived plans will contain the correct items and fulfill the conditions (FP1) – (FP3) together with constraint (C7) (if necessary). Since the cumulative LIFO constraint (CLC) is stronger than the LIFO constraint for pickup points (C1), all derived plans fulfill the constraint (C1), too.

(ii) For each delivery point in a LIFO route, the "corresponding last pickup point" is found as follows. We consider all pickup points lying in the route before the regarded delivery point and choose the last of them as corresponding last pickup point. Obviously, for each delivery point exists at least one pickup point lying in the route before it. Furthermore, the so selected pickup point is a last pickup point because it directly precedes the regarded delivery point or there are only delivery points located between them. Thus for each delivery point, a packing plan can be derived by simply removing placements for items already unloaded from the packing plan of its corresponding last pickup point. The derived packing plans will contain the correct items and fulfill the conditions (FP1) – (FP3) together with constraint (C7) (if necessary). Finally, the proof of the LIFO constraint for delivery points (C2) can take place indirectly like the proof of *Proposition* 1 because in case of LIFO routes, it is still ensured that delivery points lie in invers order of their corresponding pickup points. Thus a violation of constraint (C2) would result in a violation of constraint (C1) at a pickup point.

(iii) The packing plans for all nodes will be generated by simply removing placements from the packing plans of the last pickup points. Since these original packing plans of the last pickup points observe reloading ban constraint (C3), the derived plans will fulfill it too. □

In section 4.2, it was shown that in case of IPR routes the inclusion of the cumulative LIFO constraint (CLC) does not lead to an additional restriction of the search space. In case of LIFO routes, it holds the same. The proof is not presented here because it is almost identical to the proof of *Proposition* 2.

The outcome of *Proposition* 3 is that the concept of packing checks from the IPR approach (only performing packing checks for the request sequences of the last pickup points) can be taken over to the Simultaneous Packing approach. However, the packing procedure now must additionally ensure the reloading ban constraint for the last pickup points. This check must be made per LIFO route block because only last pickup points contained in the same LIFO route block have common requests and common items. Thus for last pickup points which do not belong to the same LRB, the reloading ban constraint is observed automatically.

Example 3 (continued): To check the LRB $P1 \rightarrow P2 \rightarrow D2 \rightarrow P3 \rightarrow P4 \rightarrow D4 \rightarrow P5 \rightarrow D5 \rightarrow D3 \rightarrow D1$ in terms of packing for the 2L-PDP the following steps must be performed:

- finding packing plans for the three request sequences (1, 2), (1, 3, 4), (1, 3, 5),
- checking if the items of request 1 hold the same placements in all three plans,
- checking if the items of request 3 hold the same placements in the second and third plan.

To check the whole LIFO route from *Example* 3 in terms of packing for the 2L-PDP, it furthermore has to be checked (independently) if a feasible packing plan for the request sequence (6, 7) exists. Since the second LRB $P6 \rightarrow P7 \rightarrow D7 \rightarrow D6$ contains only one last pickup point, there are no interdependencies between packing plans of the second LRB to observe. The packing plans for the individual points in the route can be derived as follows from the plans of the last pickup points:

- for $P1$, $D2$ from the plan of $P2$ (request sequence (1, 2)),
- for $P3$, $D4$ from the plan of P4 (request sequence (1, 3, 4)),
- for $D5$, $D3$, $D1$ from the plan of $P5$ (request sequence (1, 3, 5)),
- for $P6$, $D7$, $D6$ from the plan of $P7$ (request sequence (6, 7)).

Proposition 4: Let rs_1 and rs_2 be two request sequences with $p+q$ and $p+s$ requests, respectively (p, q, $s > 0$). Furthermore, let the first p requests of both request sequences be identically: $rs_1 = (i_1,...,i_p,i_{p+1},...,i_{p+q})$, $rs_2 = (i_1,...,i_p,i_{p+q+1},...,i_{p+q+s})$. If v and u ($v \in \{1,...,6\}$, $u \in \{1,...,5\}$) exist such that the heuristic H_v can pack both request sequences using the ordering Ord_u, then the resulting packing plans for both request sequences fulfill the reloading ban constraint, i.e. the items of requests $\{i_1,...,i_p\}$ hold the same placements in both plans.

Proof: Let j be the total number of items belonging to the first p identical requests $\{i_1,...,i_p\}$ of the request sequences rs_1 and rs_2. Then the corresponding item sequences is_1 and is_2 contain j identical items at the beginning (after getting ordered by Ord_u). When the heuristic H_v constructs the packing plan for both item sequences, obviously the first j identical items are getting the same placements because the heuristics do not look forward, i.e. to determine the placements for the first j items, only the properties (length and width) of these items are taken into account. The heuristics do not take into account the number, length or width of further items in is_1 and is_2 holding positions greater than j. Together with the fact that H_1–H_6 do not contain any stochastic components, this ensures identical placements for the first j items in both packing plans. □

With *Proposition* 4 it becomes clear, that to check a LRB in terms of packing for the 2L-PDP, it is sufficient to find a certain heuristic H_v and a certain ordering Ord_u such that the pair (H_v, Ord_u) can pack successfully the request sequences of all last pickup points of the considered LRB. Now the packing procedure *packing_check_rs* for the IPR routes can be enhanced to the procedure *packing_check_lrb* which checks a LRB in terms of packing for the 2L-PDP.

In Algorithm 3, the packing procedure *packing_check_lrb* is shown in detail. The procedure uses the cache *cache-lrb* to speed up the search. The procedure checks first if the input *lrb* is contained in *cache-lrb*. In this case, the result is taken from the cache and the procedure terminates. Otherwise, the procedure builds up the array *rs-arr* of all request sequences of last pickup points of *lrb*. Then the procedure tries to find a (u, v)-pair for which H_v can construct packing plans for all elements of *rs-arr* using the ordering Ord_u. For this purpose, the procedure uses three nested loops over the orderings, the heuristics and the item sequences. Each time a new ordering will be processed, the item sequence array *is-arr* will be built up. It contains the ordered item sequences for the request sequences of *rs-arr*. To speed up the packing procedure, this step will be done before entering the loop over the heuristics. In the most inner loop, the elements of *is-arr* are getting packing checked with heuristic H_v. The boolean variable *ok* indicates whether all item sequences could be packed successfully using the (H_v, Ord_u)-pair. In case of *ok=true* the procedure ends with *result*=true because packing plans fulfilling the reloading ban constraint for all last pickup points of *lrb* were found. Otherwise *ok=false* signals that at least one item sequence could not be packed with heuristic H_v using ordering Ord_u. In this case, the procedure continues with the next heuristic or the next ordering. If there are no further (H_v, Ord_u)-pairs remaining, the procedure ends with *result*=false. Before leaving the procedure, the overall packing result for *lrb* is saved in *cache-lrb*.

```
packing_check_lrb (in: lifo route block lrb, out: boolean result)
    if cache-lrb.contains(lrb) then
        return cache-lrb.get-result(lrb)
    rs-arr := build-request-sequences(lrb)          // build request sequences for all last pickup points in lrb
    for u := 1 to 5 do                                // loop over u (orderings)
        is-arr := { }                                 // allocate empty array for item sequences
        for j := 1 to sizeof(rs-arr) do
            is-arr[j] := build-item-sequence(rs-arr[j], $Ord_u$)   // build corresponding item sequences for request seq.
        for v := 1 to 6 do                            // loop over v (heuristics)
            ok := true
            for j := 1 to sizeof(is-arr) do           // loop over j (item sequences)
                if heuristic $H_v$ can not pack item sequence is-arr[j] then
                    ok := false                       // set ok to false to neglect (u, v)-pair
                    break inner for loop              // break for loop over index j
            if ok = true then
                cache-lrb.set-result(lrb, true)       // $H_v$ + $Ord_u$ have checked all request-seq. with success
                return true                           // save positive result in cache-lrb and return
    cache-lrb.set-result(lrb, false)                  // all (u, v)-pairs were tried without success
    return false                                      // save negative result in cache-lrb and return
```

Algorithm 3: Packing procedure *packing_check_lrb*.

Finally, it should be mentioned that the procedure *packing_check_lrb* does not save packing plans during the search process for already checked LRBs, because this would consume too much memory. In the *cache-lrb* (beside the LRB itself and its checking result) only the values u and v are saved, i.e. the numbers of the heuristic H_v and the ordering Ord_u which were able to construct feasible packing plans for all last pickup points of the considered LRB. When the stopping criterion in the LNS routing procedure is met and the search is getting aborted, the packing plans for all last pickup points in the best solution are reconstructed by the appropriate packing heuristic H_v in combination with the appropriate ordering Ord_u.

5 Computational experiments

The section is organized as follows. In the first part we test our solution approach against well-known 2L-CVRP instances to check whether we can reach the solution quality of the best so far existing algorithms for the 2L-CVRP. Since the 2L-CVRP is a special case of the 2L-PDP, where all pickups take place in the depot and after performing one delivery, no other pickup may follow in the route, our solutions approach can be used to solve 2L-CVRP instances, too. We use a modified version of the first hybrid algorithm (IPR) to meet the special requirements of the 2L-CVRP. In the 2L-CVRP there is to construct only one packing plan for each route with no LIFO constraint for pickup points (C1) and no reloading ban constraint (C3) to observe. Thus the two additional requirements to the routes (see section 4.2) can be dropped in this test, while LIFO constraint of the 2L-CVRP is ensured by the packing procedure.

In the second part, 60 new benchmark instances with up to 150 requests and 433 items are introduced. In the third part, which is the main part of this section, the two new solution approaches "Independent Partial Routes" and "Simultaneous Packing" are tested against the new instances. The test results will be compared with two lesser constrained problem variants, namely the "Unrestricted" and "One Dimensional" (1D). In the Unrestricted variant, we assume that at each node any reloading can be made without any cost or time consumption, hence both LIFO constraints (C1) and (C2) and the reloading ban constraint (C3) will be neglected in this variant. In the 1D variant, we drop furthermore the requirement to construct feasible packing plans, hence it is only required that the total area of all items loaded on the vehicle does not exceed the total area of the loading area, without considering if a feasible packing exists. Thus, the 1D variant is identical to the classical PDP with two scalar capacity conditions (weight and area). For both 1D variant and Unrestricted variant, we use a modified version of the IPR hybrid algorithm where the two additional requirements to the routes are dropped. In the 1D variant no packing checks will be done at all, while in the Unrestricted variant there will be constructed feasible packing plans for all nodes in a route. These packing plans do not need to meet the LIFO constraint for pickup points, thus the first criterion in all orderings (see section 4.1) is neglected in this variant.

Both hybrid algorithms are implemented in Java programming language using Eclipse IDE. All the experiments have been conducted on a PC with Intel Core i7-6700K (4.0 GHz, 32 GB RAM) running Windows 10 operating system. The identical chosen parameter setting for the routing procedure LNS of both hybrid algorithms is shown in Table 5. All parameter values were determined based on limited computational experiments using a trial and error strategy.

Table 5: Parameter setting for the LNS routing procedure.

Parameter	Description	Value
r_{min}	lower bound for no. of removed customers	$0.04 \cdot n$
r_{max}	upper bound for no. of removed customers	$0.4 \cdot n$
w	start temperature control parameter	0.005
c	rate of geometrical cooling	0.9999
$p(Rh_R)$, $p(Rh_S)$	probability of Random / Shaw removal	0.3, 0.4
$p(Rh_W)$, $p(Rh_T)$	probability of Worst / Tour removal	0.1, 0.2
$p(Ih_G)$, $p(Ih_{R2})$, $p(Ih_{R3})$	probability of Greedy / Regret-2 / Regret-3 insert	0.1, 0.6, 0.3
w_{r1}, w_{r2}	weights of relatedness formula for Shaw removal	9, 2

5.1 Computational results for 2L-CVRP

Gendreau et al. (2008) have introduced 36 CVRP problems containing up to 255 customers. For each of these CVRP problems, they created five 2L-CVRP instances by considering different classes of item characteristics (Class 1–5). Thereby, the instances of Class 1 are pure CVRP problems, whereas the 144 instances of Classes 2–5 are "real" 2L-CVRP problems. For these instances, each item belongs to one of three possible shape categories with equal probability, while the number of items demanded by a customer is determined as random value from a given interval. The details of the items characteristics are shown in Table 6. The average number of items per customer rises from 1.5 for instances of Class 2 up to 3 items per customer for instances of Class 5.

Table 6: The item characteristics for 2L-CVRP instances of Class 2–5.

Class	Item-Number m_i	Vertical		Homogeneous		Horizontal	
		Length	Width	Length	Width	Length	Width
2	[1, 2]	[0.4L, 0.9L]	[0.1W, 0.2W]	[0.2L, 0.5L]	[0.2W, 0.5W]	[0.1L, 0.2L]	[0.4W, 0.9W]
3	[1, 3]	[0.3L, 0.8L]	[0.1W, 0.2W]	[0.2L, 0.4L]	[0.2W, 0.4W]	[0.1L, 0.2L]	[0.3W, 0.8W]
4	[1, 4]	[0.2L, 0.7L]	[0.1W, 0.2W]	[0.1L, 0.4L]	[0.1W, 0.4W]	[0.1L, 0.2L]	[0.2W, 0.7W]
5	[1, 5]	[0.1L, 0.6L]	[0.1W, 0.2W]	[0.1L, 0.3L]	[0.1W, 0.3W]	[0.1L, 0.2L]	[0.1W, 0.6W]

Our hybrid algorithm, in the following denoted with "LNS+6CH", was tested five times against each of the 144 instances considering the problem variant with LIFO constraint and fixed orientation ("NoRotate") allowing up to one hour CPU time. The results are first averaged over the five runs and then averaged over the Classes 2–5 for each of the 36 problems. In Table 7, the results for the LNS+6CH hybrid algorithm are compared to those obtained by

- ACO with 3 hours CPU time (Fuellerer et al., 2009), Pentium IV 3.2 GHz,
- EGTS + LBFH (Leung et al., 2011), Intel Core 2 Duo 2.0 GHz,
- PRMP (Zachariadis et al., 2013), Intel Core 2 Duo E6600 2.4 GHz,
- VNS (Wei et al., 2015), Intel Xeon E5430 with a 2.66 GHz.

Table 7: Results for 2L-CVRP (averaged over Classes 2–5).

Prob-lem	ACO			EGTS + LFBH			PRMP			VNS			LNS+6CH		
	ttd-avg	rt-avg	gap %	ttd-avg	rt-avg	gap %	ttd-avg	rt-avg	gap %	ttd-avg	rt-avg	gap %	ttd-avg	rt-avg	gap %
1	295.10	8.9	2.79	303.40	3.2	5.68	**287.09**	1.4	0.00	287.52	4.5	0.15	298.01	5.6	3.80
2	345.28	0.7	0.31	345.23	1.3	0.30	**344.21**	1.0	0.00	**344.21**	0.6	0.00	345.23	5.0	0.30
3	383.12	3.7	0.45	387.89	5.0	1.70	**381.40**	1.3	0.00	381.43	1.5	0.01	384.58	5.0	0.83
4	441.11	3.6	0.26	443.25	2.4	0.75	**439.97**	1.6	0.00	440.27	1.0	0.07	442.14	5.0	0.49
5	383.21	15.7	0.21	387.60	4.6	1.36	**382.39**	2.6	0.00	**382.39**	4.2	0.00	386.80	6.0	1.15
6	500.76	5.2	0.26	502.25	3.5	0.55	**499.48**	5.6	0.00	**499.48**	1.7	0.00	502.88	5.0	0.68
7	705.64	14.3	0.49	715.54	8.3	1.90	702.27	5.3	0.01	**702.18**	12.3	0.00	711.31	5.4	1.30
8	713.33	19.9	2.26	716.36	8.3	2.69	699.55	7.0	0.28	**697.58**	21.2	0.00	713.12	7.4	2.23
9	616.69	5.8	0.28	621.23	4.3	1.02	615.93	6.2	0.16	**614.95**	3.7	0.00	615.94	5.0	0.16
10	701.65	61.6	2.01	731.69	23.3	6.38	688.63	55.0	0.12	**687.80**	115.8	0.00	711.32	20.5	3.42
11	736.53	71.5	1.57	762.83	38.0	5.20	725.83	75.3	0.10	**725.11**	54.5	0.00	748.82	23.8	3.27
12	617.07	8.6	0.41	622.35	7.9	1.27	615.23	7.1	0.12	**614.52**	7.5	0.00	620.41	5.1	0.96
13	2598.46	76.3	1.75	2647.88	30.8	3.69	2554.93	119.6	0.05	**2553.76**	53.5	0.00	2617.70	25.4	2.50
14	1050.48	202.3	1.93	1075.04	49.0	4.31	**1030.61**	637.1	0.00	1033.12	416.5	0.24	1054.56	67.4	2.32
15	1223.03	172.5	3.09	1223.19	65.8	3.10	1193.88	68.2	0.63	**1186.38**	298.5	0.00	1210.15	57.5	2.00
16	701.72	9.4	0.10	703.74	13.0	0.39	**701.01**	14.2	0.00	**701.01**	3.4	0.00	704.75	5.0	0.53
17	865.56	7.5	0.17	869.93	16.1	0.68	865.33	40.9	0.15	**864.06**	4.4	0.00	864.57	5.0	0.06
18	1073.34	362.1	1.52	1096.57	67.8	3.72	1061.29	95.1	0.38	**1057.27**	396.5	0.00	1078.03	120.7	1.96
19	779.29	149.8	2.05	798.20	61.7	4.53	767.13	188.3	0.46	**763.62**	297.5	0.00	779.68	49.9	2.10
20	544.79	1322.6	2.34	559.17	232.5	5.05	535.89	1660.9	0.67	**532.31**	921.8	0.00	544.18	440.9	2.23
21	1061.44	1284.6	2.63	1084.98	179.0	4.91	1043.12	420.2	0.86	**1034.25**	1003.0	0.00	1055.95	428.2	2.10
22	1086.84	904.9	2.45	1113.64	152.3	4.97	1068.35	524.3	0.71	**1060.87**	1107.8	0.00	1085.37	301.6	2.31
23	1100.68	1490.3	2.96	1130.13	215.3	5.72	1080.59	519.5	1.09	**1068.99**	953.0	0.00	1094.16	496.8	2.35
24	1158.06	389.6	2.03	1177.28	132.7	3.72	1143.88	1064.3	0.78	**1135.05**	841.3	0.00	1157.30	129.9	1.96
25	1428.74	3007.9	3.07	1470.11	373.2	6.06	1403.33	2319.5	1.24	**1386.16**	1306.0	0.00	1423.94	1002.6	2.73
26	1427.91	4379.0	4.98	1431.32	499.0	5.23	1374.49	1491.2	1.05	**1360.19**	1240.3	0.00	1387.02	1459.4	1.97
27	1400.46	1898.3	2.79	1445.64	371.4	6.10	1378.13	4163.8	1.15	**1362.50**	1242.3	0.00	1396.34	632.8	2.48
28	2734.60	10800.8	2.70	2808.10	979.8	5.46	2677.71	8640.1	0.57	**2662.59**	2423.3	0.00	2683.39	3600.0	0.78
29	2361.32	10800.9	4.80	2396.78	1150.4	6.37	2273.25	5484.3	0.89	**2253.26**	2672.8	0.00	2309.20	3600.0	2.48
30	1906.16	10800.7	4.10	1983.48	1699.0	8.32	1858.69	4676.9	1.51	**1831.09**	2502.0	0.00	1876.39	3600.0	2.47
31	2431.13	10800.8	4.18	2497.25	4368.2	7.02	2370.77	5845.4	1.59	**2333.55**	2760.8	0.00	2390.12	3600.0	2.42
32	2400.06	10800.6	4.96	2438.65	2445.8	6.65	2332.28	9433.2	2.00	**2286.62**	2664.0	0.00	2337.85	3600.0	2.24
33	2467.61	10800.6	4.72	2543.24	2053.7	7.93	2404.52	5662.5	2.04	**2356.37**	2614.5	0.00	2413.23	3600.0	2.41
34	1272.29	10800.7	5.84	1276.27	3443.5	6.17	1231.90	13141.8	2.48	**1202.10**	2825.8	0.00	1241.73	3600.0	3.30
35	1612.42	10800.7	10.00	1606.38	4560.8	9.59	1500.97	8989.6	2.40	**1465.77**	3053.0	0.00	1593.32	3600.0	8.70
36	1846.66	10800.8	5.39	1850.50	3667.1	5.61	1774.94	10059.6	1.30	**1752.16**	3282.5	0.00	1891.36	3600.0	7.94
Avg			**2.55**			**4.28**			**0.69**			**0.01**			**2.25**

The allowed CPU time for LNS+6CH is set to one third of the CPU time consumed by the ACO algorithm of Fuellerer et al. (2009) but not less than 5 seconds. For each algorithm, the average total travel distance and average computation time (averaged over Classes 2–5) are provided. For each problem, the minimum average total travel distance value is marked in bold. The gap in percent is calculated as (*avg-ttd / min-avg-ttd* – 1)*100%.

The results show that LNS+6CH performs better than the "older" algorithms ACO (0.30%) and EGTS + LFBH (2.03%) but cannot reach the solution quality of the "newer" algorithms PRMP (1.56%) and VNS (2.24%). Nevertheless, these gaps are small and the computation times used for LNS+6CH are comparable to those of the other algorithms, so LNS+6CH can be considered as "state of the art" procedure for solving the 2L-CVRP.

5.2 Benchmark instances for 2L-PDP

The 60 new 2L-PDP instances were generated based on the 2L-CVRP instances by Gendreau et al. (2008). First 20 2L-CVRP instances with 25 to 150 customers were selected as shown in Table 8.

Table 8: 2L-CVRP instances used to create new 2L-PDP benchmark instances.

Average item count	Customer count				
per request	25	50	75	100	150
1.5	09-2	19-2	21-2	25-2	30-2
2.0	09-3	19-3	21-3	25-3	30-3
2.5	09-4	19-4	21-4	25-4	30-4
3.0	09-5	19-5	21-5	25-5	30-5

For each of the selected 2L-CVRP instances, three 2L-PDP instances were created with different characteristics regarding the distribution of pickup and delivery points of the requests. In the first variant "Random", the sites are uniformly distributed in a rectangular section of the plane, while they are clustered in the other variants. In the second variant "Mixed clusters", individual clusters may contain pickup as well as delivery points, while only sites of one sort can occur in an individual cluster of the third variant "Pure clusters". The three types of 2L-PDP instances are denoted by the suffixes "-Rnd", "-Mix" and "-Pur", e.g. the instances constructed based on 09-2 are denoted with 09-2-Rnd, 09-2-Mix and 09-2-Pur.

As in the original 2L-CVRP instances, each four 2L-PDP instances for the same problem number and the same distribution variant (e.g. 09-2-Rnd to 09-5-Rnd) share the same node set. The item sets, the requests weights and the dimension of the loading areas ($L = 40$, $W = 20$) were taken over from the original instances without any change (see Table 9). The vehicles weight capacity was slightly adapted

to ensure that the weight constraint (C4) does not become redundant but a good utilization of the loading area is still possible and the packing task is not too easy. The maximum route length was defined so that best solutions found by the hybrid algorithms contain a reasonable number of routes (from 3 to 15 routes depending on the size of the instance). The characteristics of the 2L-PDP instances are summarized in Table 9. The instances are offered at the website
http://www.mansci.ovgu.de/Forschung/Materialien.html.

Table 9: Overview of the new 2L-PDP benchmark instances.

Parameter	Value	Remark
Total number of instances	60	
Request number per instance	25 / 50 / 75 / 100 / 150	12 instances each
Node number per instance	50 / 100 / 150 / 200 / 300	12 instances each
Average number of items per request	1.5 / 2.0 / 2.5 / 3.0	15 instances each
Distribution variants of pickup / delivery points	Random / Pure Cluster / Mixed Cluster	20 instances each

5.3 Computational results for the 2L-PDP

The detailed results for the 2L-PDP instances regarding total travel distance (*ttd*) are presented in Table 10 and 11. The structure for both tables is identical, Table 10 covers the “Rotate” variant where 90° rotations of the items are allowed, while Table 11 corresponds to the “NoRotate” variant with the additional constraint (C7).

Table 10: Results (travel distances) for different variants of 2L-PDP (“Rotate” variant).

Instance				1D	Unrestricted			Simultaneous Packing			Independent Partial Routes		
type	req. n	items m	CPU sec	avg-ttd	avg-ttd	gap %	reloading effort	avg-ttd	gap %	reloading effort	avg-ttd	gap %	reloading effort
09-2	25	40	30	786.63	823.83	4.70	156.62	962.75	22.97	-	1007.82	28.41	-
09-3	25	61	30	812.82	849.68	4.58	158.31	971.18	19.95	-	999.83	23.25	-
09-4	25	63	30	801.42	845.59	5.37	145.94	995.88	24.76	-	1017.64	27.33	-
09-5	25	91	30	772.90	785.15	1.54	185.65	900.69	16.77	-	975.16	25.96	-
19-2	50	82	60	1215.46	1283.19	5.53	175.56	1527.66	26.03	-	1595.75	31.31	-
19-3	50	103	60	1257.50	1299.03	3.30	183.71	1548.52	23.43	-	1603.02	27.67	-
19-4	50	134	60	1277.66	1339.93	4.91	178.30	1580.34	23.70	-	1655.35	29.44	-
19-5	50	157	60	1113.60	1136.04	1.99	268.14	1410.24	26.86	-	1495.82	33.92	-
21-2	75	114	120	1659.17	1745.65	5.45	201.10	2086.47	25.66	-	2113.70	27.40	-
21-3	75	164	120	1845.22	1952.08	5.72	182.00	2236.64	20.96	-	2259.65	22.28	-
21-4	75	168	120	1683.96	1735.30	2.98	191.06	2098.60	24.43	-	2126.25	26.06	-
21-5	75	202	120	1560.33	1596.65	2.39	258.28	1970.56	26.13	-	2024.57	29.64	-
25-2	100	157	300	2254.38	2398.23	6.39	176.77	2878.46	27.57	-	2944.73	30.60	-
25-3	100	212	300	2258.33	2348.21	3.97	204.74	2844.93	25.84	-	2909.35	28.81	-
25-4	100	254	300	2274.79	2350.94	3.36	190.97	2842.27	24.92	-	2913.71	28.19	-
25-5	100	311	300	2009.49	2043.85	1.71	271.30	2654.88	31.81	-	2736.86	36.05	-
30-2	150	225	900	3018.56	3169.07	5.00	191.66	3844.96	27.32	-	3900.22	29.20	-

type	req. n	items m	CPU sec	1D avg-ttd	Unrestricted avg-ttd	gap %	reloading effort	Simultaneous Packing avg-ttd	gap %	reloading effort	Independent Partial Routes avg-ttd	gap %	reloading effort
30-3	150	298	900	3182.12	3313.98	4.15	190.92	3958.25	24.37	-	4027.43	26.60	-
30-4	150	366	900	3144.45	3251.56	3.41	194.16	3913.52	24.43	-	3953.07	25.74	-
30-5	150	433	900	2772.60	2821.62	1.72	274.09	3540.44	27.61	-	3613.64	30.34	-
Average						**3.91**	**198.96**		**24.78**			**28.41**	

Table 11: Results (travel distances) for different variants of 2L-PDP (“NoRotate” variant).

Instance				**1D**	**Unrestricted**			**Simultaneous Packing**			**Independent Partial Routes**		
type	**req. n**	**items m**	**CPU sec**	**avg-ttd**	**avg-ttd**	**gap %**	**reloading effort**	**avg-ttd**	**gap %**	**reloading effort**	**avg-ttd**	**gap %**	**reloading effort**
09-2	25	40	30	786.63	888.97	12.77	127.93	1015.79	29.61	-	1049.79	33.72	-
09-3	25	61	30	812.82	894.75	9.79	153.85	1021.14	25.96	-	1060.53	30.56	-
09-4	25	63	30	801.42	892.24	10.93	106.92	1013.19	26.90	-	1027.84	28.63	-
09-5	25	91	30	772.90	785.12	1.59	180.01	901.34	16.85	-	979.45	26.51	-
19-2	50	82	60	1215.46	1340.42	10.41	179.32	1594.17	31.72	-	1651.59	36.14	-
19-3	50	103	60	1257.50	1347.99	7.27	180.90	1609.69	28.31	-	1653.40	31.92	-
19-4	50	134	60	1277.66	1371.68	7.40	162.01	1628.44	27.71	-	1684.76	31.88	-
19-5	50	157	60	1113.60	1152.01	3.43	231.37	1437.16	29.42	-	1534.23	37.52	-
21-2	75	114	120	1659.17	1810.71	9.27	203.86	2172.99	30.97	-	2219.16	33.90	-
21-3	75	164	120	1845.22	1993.00	7.97	177.08	2311.73	25.08	-	2338.55	26.57	-
21-4	75	168	120	1683.96	1771.38	5.12	170.97	2140.80	26.98	-	2172.33	28.86	-
21-5	75	202	120	1560.33	1632.27	4.70	229.42	2004.40	28.31	-	2043.14	30.89	-
25-2	100	157	300	2254.38	2526.46	12.11	170.78	3005.40	33.20	-	3088.52	36.98	-
25-3	100	212	300	2258.33	2443.66	8.19	192.45	2963.90	31.14	-	3025.99	33.99	-
25-4	100	254	300	2274.79	2406.40	5.78	175.17	2929.75	28.80	-	2988.75	31.53	-
25-5	100	311	300	2009.49	2075.81	3.31	233.85	2683.32	33.21	-	2770.52	37.70	-
30-2	150	225	900	3018.56	3315.24	9.88	171.31	3990.33	32.15	-	4028.47	33.45	-
30-3	150	298	900	3182.12	3434.56	7.94	177.21	4094.93	28.69	-	4148.72	30.42	-
30-4	150	366	900	3144.45	3344.72	6.38	185.50	4048.24	28.72	-	4084.07	29.93	-
30-5	150	433	900	2772.60	2865.08	3.29	243.84	3602.58	29.87	-	3668.91	32.38	-
Average						**7.38**	**182.69**		**28.68**			**32.17**	

In the leftmost column of both tables, the instance types are listed. The next three columns show the number of requests, the number of items and the allowed CPU time, which varies from 30 to 900 seconds depending on the size of the instance (apart from that, the allowed computation time for the 1D variant is set to only 20% of the normal value). The fifth column shows the total travel distances for the 1D variant for which only the weight constraint (C4), and the routing constraints (C5) and (C6) are considered and no packing check will be done (only the total item area will still be checked). In the following nine columns, the total travel distances, the gaps and the reloading quantities are indicated for the Unrestricted variant and also for the two hybrid algorithms (Simultaneous Packing / Independent Partial Routes) for the original problem variant. In the Unrestricted, variant the constraints (C1) – (C3) are omitted while the others constraints, especially the need to find valid packing plans, are still in force. In this variant reloading effort at each pickup or delivery point can occur, i.e. temporary or permanent changes of placements of items which do not belong to the loaded / unloaded request may happen. In the Simultaneous Packing and Independent Partial Routes solution approach all constraints are in force as described in Chapter 3, where all reloading effort is ruled out by the constraints (C1) – (C3). In the Simultaneous Packing approach, the reloading ban constraint is enforced by a new type of packing procedure, while in the Independent Partial Routes approach the reloading ban constraint (C3) is enforced by an additional routing condition instead of the simultaneous packing checks which reduces the numerical effort but restricts the search space more. All presented total travel distances are mean values over five runs. To keep the tables compact the results are averaged, furthermore, over all instances of the same type, e.g. "09-2" stands for the three 2L-PDP instances which are derived from the original 2L-CVRP instance 09-2. The corresponding gaps are calculated as $(ttd - ttd_{1D}) / ttd_{1D} * 100$ (%). The reloading effort is given as percentage of the total item area (= sum of the area of all items in the instance). If an item is reloaded, say, at three nodes in the route, then the area of the item is counted three times. Thus it may occur that the reloading effort exceeds 100%. In the last lines of Tables 10 and 11, the gap values of the 2L-PDP variants are averaged over the 60 instances. Detailed results for each single instance are presented in Tables 13 and 14 of appendix A.

Summarizing the results for the "Rotate" problem variant, we can state that the travel distances increase significantly increases if the 2L-PDP instances are solved instead of the corresponding 1D-PDP instances. For the Unrestricted variant, the total travel distances grow on average by 3.91% compared to the 1D case. For the original problem variant, the mean gap is even higher and amounts to 24.78% (Simultaneous Packing approach) and 28.41% (Independent Partial Routes approach), respectively. For the Unrestricted variant arises a reloading effort of 198.96% on average, which means that each item was reloaded (on average) nearly two times during its route, while for the two new hybrid algorithms for the original problem variant no reloading effort occurs by definition. So we come to the conclusion that avoiding any reloading effort leads to increase of the travel costs of approximately 20% or the other way round, we can save approximately 20% of the travel costs if we are willing to pay this in form of the additional reloading effort. The comparison between the two new hybrid algorithms shows that the more complex Simultaneous Packing approach performs 2.83% (124.78% to 128.41%) better than the simpler Independent Partial Routes approach if no reloading is allowed. This result coincides with the expectation formulated in section 4 (see Table 1).

For the "NoRotate" problem variant, the results regarding total travel distance show gaps which are approximately 4% points larger than in the "Rotate" variant. This result is plausible because the packing task without the possibility to rotate items is more difficult to solve. This leads to an additional restriction of the solution space and an increase of the best objective function value. In case of omitting the constraints (C1) – (C3), the "NoRotate" problem variant shows a smaller reloading effort (182.69% to 198.96%) because the longer routes lead to generally "less occupied" loading areas and to less situations where reloading effort can occur. Furthermore, the results of the "NoRotate" variant confirm the conclusions we made in the previous paragraph.

In Table 12, the average computing times to find the best solution and the average total number of iterations executed are shown for the two new hybrid algorithms. Again the results are averaged over all instances of the same type, while the detailed results are presented in Table 15 of Appendix A. The times are given as absolute values and as percentages of the allowed computing time per instance. In the last column the ratio of executed iterations of both hybrid algorithms is shown (*iterations*$_{SP}$ / *iterations*$_{IPR}$). All values are averaged over five runs. The results show that the simpler Independent Partial Routes approach only needs 44.78% of the computing time on average to find the best solution while the Simultaneous Packing approach needs 48.63% to find the best solution. The comparison of total executed iterations shows that the Simultaneous Packing approach can execute only approximately 40% of the iterations of the other approach in the same computing time. This shows that the Simultaneous Packing approach is more expensive in terms of CPU usage than the Independent Partial Routes approach because of the more complex packing algorithm. On the other hand, there may be still potential for further improvements with the Simultaneous Packing approach if more CPU time would be allowed (especially for the instances with 150 requests). Again, this result coincide with the expectation formulated in section 4 (see Table 1).

Table 12: Total iteration numbers and computing times to find the best solution (“Rotate” variant).

Instance				Independent Partial Routes			Simultaneous Packing			Ratio iterations
type	req. n	items m	CPU sec	Runtime to best	Runtime to best in %	Total iterations	Runtime to best	Runtime to best in %	Total iterations	
09-2	25	40	30	3.93	13.10	743407.67	3.38	11.27	316799.00	0.43
09-3	25	61	30	5.57	18.58	760754.67	4.41	14.69	325637.33	0.43
09-4	25	63	30	5.04	16.79	737302.00	4.85	16.16	322605.67	0.44
09-5	25	91	30	4.69	15.63	744638.00	1.75	5.83	281710.67	0.38
19-2	50	82	60	20.80	34.67	318347.67	18.74	31.23	121372.67	0.38
19-3	50	103	60	16.93	28.21	319906.33	24.49	40.82	125209.33	0.39
19-4	50	134	60	19.87	33.12	312806.33	23.58	39.29	124830.00	0.40
19-5	50	157	60	12.88	21.46	267537.00	17.10	28.49	95587.33	0.36
21-2	75	114	120	56.77	47.31	228759.33	69.26	57.72	85333.33	0.37
21-3	75	164	120	70.48	58.73	250279.67	70.77	58.98	101774.67	0.40
21-4	75	168	120	66.60	55.50	204657.00	73.52	61.26	71585.67	0.35
21-5	75	202	120	59.60	49.67	178960.33	71.32	59.43	63674.33	0.35
25-2	100	157	300	177.60	59.20	357548.33	195.71	65.24	141081.33	0.39
25-3	100	212	300	179.42	59.81	347215.00	198.79	66.26	141376.00	0.41
25-4	100	254	300	184.13	61.38	330252.00	160.52	53.51	137278.00	0.42
25-5	100	311	300	169.64	56.55	265729.67	181.00	60.33	103366.33	0.39
30-2	150	225	900	626.13	69.57	389412.33	661.43	73.49	152337.67	0.39
30-3	150	298	900	630.90	70.10	378192.67	725.95	80.66	163377.33	0.43
30-4	150	366	900	634.83	70.54	354757.67	717.30	79.70	149151.33	0.42

30-5	150	433	900	500.48	55.61	286010.67	614.65	68.29	111248.33	0.39
Average					**44.78**			**48.63**		**0.40**

6 Conclusions and future work

In this paper, the vehicle routing problem with pickup and delivery (PDP) has been extended to an integrated vehicle routing and loading problem with 2D rectangular items to be transported in homogeneous vehicles on a rectangular 2D loading area (2L-PDP). In the problem formulation, we focused on the question under which conditions any reloading effort, i.e. any movement of items after loading and before unloading, can be avoided. It turned out that the LIFO constraints for pickup and delivery points are not sufficient. Instead, the new reloading ban constraint was required to rule out any reloading effort.

Two solution approaches implemented as hybrid algorithms consisting of a routing and a packing procedure were proposed to tackle the 2L-PDP. In the first solution approach (Independent Partial Routes), a large neighborhood search procedure for routing is combined with a packing procedure using six well-known constructive packing heuristics. To ensure the LIFO constraint at delivery points and the reloading ban constraint the search space must be restricted to routes which are fulfilling two additional requirements (1) and (2) (see Section 4.2). In the second more complex solution approach (Simultaneous Packing), basically the same routing procedure is combined with a new type of packing procedure which is able to construct a series of interrelated packing plans fulfilling the reloading ban constraint (see Section 4.4). Therefore, in the second approach the additional requirement (1) to the routes can be dropped.

The hybrid algorithms were tested with the well-known 2L-CVRP instances by Gendreau et al. (2008) and reached a good solution quality compared to the best 2L-CVRP solution methods available. For testing the hybrid 2L-PDP algorithms, 60 2L-PDP instances with up to 150 requests and up to 433 items were introduced. The results for the 2L-PDP variants are plausible in that the second approach performs nearly 3% better than the first solution approach on average. Neglecting LIFO and reloading ban constraints (Unrestricted variant) would lead to a reduction of around 20% of the total travel distance. Put differently, ruling out any reloading has to be paid by a 20% increase of travel distance.

In future research, a packing procedure based on the second solution approach should be developed, which is able to observe the LIFO constraint for delivery points, too. This would allow to drop also the additional requirement (2) so that a further improvement of the solution quality could be expected.

References

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

Bent, R; van Hentenryck, P (2006): A two-stage hybrid algorithm for pickup and delivery vehicle routing problems with time windows. *Computers & Operations Research*, 33:875–893.

Berbeglia, G; Cordeau, JF; Gribkovskaia, I; Laporte, G (2007): Static pickup and delivery problems: A classification scheme and survey. *Top*, 15:1–31.

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

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

Bortfeldt, A; Hahn, T; Männel, D; Mönch, L (2015): Hybrid algorithms for the vehicle routing problem with clustered backhauls and 3D loading constraints. *European Journal of Operational Research*, 243:82–96.

Chazelle, B (1983): The bottom-left bin packing heuristic: An efficient implementation. *IEEE Transactions on Computers*, C-32:697–707.

Crainic, T; Perboli, G; Tadei, R (2008): Extreme Point-based Heuristics for Three-dimensional Bin Packing. *INFORMS Journal on Computing*, 20:368–384.

Dominguez, O; Guimarans, D; Juan, AA; De La Nuez, I (2016): A Biased-Randomised Large Neighbourhood Search for the two-dimensional Vehicle Routing Problem with Backhauls. *European Journal of Operational Research*, 255:442-462.

Duhamel, C; Lacomme, P; Quilliot, A; Toussaint, H (2011): A multi-start evolutionary local search for the two-dimensional loading capacitated vehicle routing problem. *Computers & Operations Research*, 38:617–640.

Fuellerer, G; Doerner, KF; Hartl, RF; Iori, M (2009): Ant colony optimization for the two-dimensional loading vehicle routing problem. *Computers & Operations Research*, 36:655–673.

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

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

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

Gendreau, M; Potvin, JY (2010): Handbook of Metaheuristics, second edition. Springer, New York.

Iori, M; Salazar Gonzalez, JJ; Vigo, D (2007): An exact approach for the vehicle routing problem with two-dimensional loading constraints. *Transportation Science*, 41:253–264.

Iori, M; Martello, S (2010): Routing Problems with Loading Constraints. *Top*, 18:4–27.

Iori, M; Martello, S (2013): An Annotated Bibliography of Combined Routing and Loading Problems. *Yugoslav Journal of Operations Research*, 23(3):311–326.

Khebbache-Hadji, S; Prins, C; Yalaoui, A; Reghioui, M (2013): Heuristics and memetic algorithm for the two-dimensional loading capacitated vehicle routing problem with time windows. *Central European Journal of Operations Research,* 21:307–336.

Leung, SCH; Zhou, X; Zhang, D; Zheng, J (2011): Extended guided tabu search and a new packing algorithm for the two-dimensional loading vehicle routing problem. *Computers & Operations Research*, 38:205–215.

Leung, SCH; Zhang, Z; Zhang, D; Hua, X; Lim, MK (2013): A meta-heuristic algorithm for heterogeneous fleet vehicle routing problems with two-dimensional loading constraints. *European Journal of Operational Research*, 225:199–210.

Li, H; Lim, A (2001): A metaheuristic for the pickup and delivery problem with time windows. 13th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'01). IEEE Computer Society, Los Alamitos, CA, 333–340.

Lodi, A; Martello, S; Vigo, D (1999): Heuristic and Metaheuristic Approaches for a Class of Two-Dimensional Bin Packing Problems. *INFORMS Journal on Computing*, 11:345–357.

Lu, Q; Dessouky, MM (2006): A new insertion-based construction heuristic for solving the pickup and delivery problem with time windows. *European Journal of Operational Research*, 175:672–687.

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

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

Männel, D; Bortfeldt, A (2017): Solving the Pickup and Delivery Problem with Three-dimensional Loading Constraints and Reloading Ban. *European Journal of Operational Research*, in press.

Moura, A; Oliveira, JF (2009): An Integrated Approach to Vehicle Routing and Container Loading Problems. *Operations Research Spectrum* 31:775-800.

Nagata, Y; Kobayashi, S (2010): A Memetic Algorithm for the Pickup and Delivery Problem with Time Windows Using Selective Route Exchange Crossover. *Parallel Problem Solving from Nature, PPSN XI*, R. Schaefer et al. (eds.), 536-545, Springer: Berlin.

Nanry, WP; Barnes, W (2000): Solving the pickup and delivery problem with time windows using reactive tabu search. *Transportation Research Part B: Methodological*, 34:107–121.

Pankratz, G (2005): A grouping algorithm for the pickup and delivery problem with time windows. *OR Spectrum*, 27:21-41.

Parragh, SN; Doerner, KF; Hartl, RF (2008): A Survey on Pickup and Delivery Problems. Part II: Transportation between pickup and delivery locations. *Journal für Betriebswirtschaft*, 58:81–117.

Pollaris, H; Braekers, K; Caris, A; Janssens, G; Limbourg, S (2015): Vehicle routing problems with loading constraints: state-of-the-art and future directions. *OR Spectrum*, 37:297–330.

Ropke, S; Pisinger, D (2006): An Adaptive Large Neighborhood Search for the Pickup and Delivery Problem with Time Windows. *Transportation Science*, 40:455–472.

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

Tao, Y; Wang, F (2015): An effective tabu search approach with improved loading algorithms for the 3L-CVRP. *Computers & Operations Research*, 55:127–140.

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

Toth, P; Vigo, D (2014): Vehicle Routing: Problems, Methods, and Applications, second edition. MOS-SIAM series on optimization, Philadelphia.

Wang, L; Guo, S; Chen, S; Zhu, W; Lim, A (2010): Two Natural Heuristics for 3D Packing with Practical Loading Constraints. *Computer Science,* Vol. 6230, 256–267.

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, 18–30.

Wei, L; Zhang, Z; Zhang, D; Lim, A (2015): A variable neighborhood search for the capacitated vehicle routing problem with two-dimensional loading constraints. *European Journal of Operational Research*, 243:798–814.

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

Zachariadis, EE; Tarantilis, CD, Kiranoudis, CT (2009): A Guided Tabu Search for the Vehicle Routing Problem with two-dimensional loading constraints. *European Journal of Operational Research*, 195:729–743.

Zachariadis, EE; Tarantilis, CD; Kiranoudis, CT (2012): The pallet-packing vehicle routing problem. *Transportation Science*, 46:341–358.

Zachariadis, EE; Tarantilis, CD, Kiranoudis, CT (2013): Integrated distribution and loading planning via a compact metaheuristic algorithm. *European Journal of Operational Research*, 228:56–71.

Zachariadis, EE; Tarantilis, CD, Kiranoudis, CT (2016): The Vehicle Routing Problem with Simultaneous Pick-ups and Deliveries and Two-Dimensional Loading Constraints. *European Journal of Operational Research*, 251:369-386.

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*, 82:20–35.

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

Appendix A

Table 13: Results (travel distances) for different variants of 2L-PDP ("Rotate" variant, complete results).

Instance				1D	Unrestricted			Simultaneous Packing			Independent Partial Routes		
name	req. n	items m	CPU sec	avg-ttd	avg-ttd	gap %	reloading effort	avg-ttd	gap %	reloading effort	avg-ttd	gap %	reloading effort
09-2-Rnd	25	40	30	704.24	733.44	4.15	220.87	944.26	34.08	–	944.43	34.11	–
09-2-Mix	25	40	30	828.46	885.71	6.91	154.00	1025.23	23.75	–	1098.42	32.59	–
09-2-Pur	25	40	30	827.21	852.33	3.04	95.01	918.75	11.07	–	980.59	18.54	–
09-3-Rnd	25	61	30	727.29	755.03	3.81	167.18	944.26	29.83	–	949.75	30.59	–
09-3-Mix	25	61	30	881.69	892.55	1.23	167.03	1034.77	17.36	–	1115.25	26.49	–
09-3-Pur	25	61	30	829.47	901.47	8.68	140.71	934.50	12.66	–	934.50	12.66	–
09-4-Rnd	25	63	30	709.10	730.40	3.00	174.54	945.28	33.31	–	945.28	33.31	–
09-4-Mix	25	63	30	847.03	905.10	6.86	145.93	1045.78	23.46	–	1111.18	31.19	–
09-4-Pur	25	63	30	848.14	901.27	6.26	117.35	996.58	17.50	–	996.46	17.49	–
09-5-Rnd	25	91	30	706.07	709.09	0.43	205.65	869.40	23.13	–	869.40	23.13	–
09-5-Mix	25	91	30	821.05	833.10	1.47	155.85	955.09	16.33	–	1098.32	33.77	–
09-5-Pur	25	91	30	791.58	813.27	2.74	195.45	877.58	10.86	–	957.75	20.99	–
19-2-Rnd	50	82	60	1123.94	1212.26	7.86	197.88	1590.74	41.53	–	1594.11	41.83	–
19-2-Mix	50	82	60	1434.45	1517.19	5.77	181.75	1733.38	20.84	–	1865.05	30.02	–
19-2-Pur	50	82	60	1087.98	1120.11	2.95	147.06	1258.88	15.71	–	1328.09	22.07	–
19-3-Rnd	50	103	60	1188.44	1224.59	3.04	227.07	1613.54	35.77	–	1612.66	35.70	–
19-3-Mix	50	103	60	1484.52	1534.46	3.36	191.25	1748.10	17.76	–	1839.38	23.90	–
19-3-Pur	50	103	60	1099.54	1138.03	3.50	132.82	1283.92	16.77	–	1357.02	23.42	–
19-4-Rnd	50	134	60	1213.39	1246.89	2.76	200.90	1633.25	34.60	–	1641.95	35.32	–
19-4-Mix	50	134	60	1480.96	1554.61	4.97	177.80	1794.54	21.17	–	1917.66	29.49	–
19-4-Pur	50	134	60	1138.63	1218.30	7.00	156.21	1313.23	15.33	–	1406.45	23.52	–
19-5-Rnd	50	157	60	1060.30	1079.21	1.78	327.44	1544.18	45.64	–	1544.28	45.65	–
19-5-Mix	50	157	60	1304.16	1334.05	2.29	271.72	1562.40	19.80	–	1761.23	35.05	–
19-5-Pur	50	157	60	976.34	994.85	1.90	205.26	1124.14	15.14	–	1181.94	21.06	–
21-2-Rnd	75	114	120	1767.76	1828.34	3.43	225.93	2338.90	32.31	–	2340.32	32.39	–
21-2-Mix	75	114	120	1799.79	1877.38	4.31	231.19	2172.83	20.73	–	2209.18	22.75	–
21-2-Pur	75	114	120	1409.95	1531.23	8.60	146.17	1747.68	23.95	–	1791.59	27.07	–
21-3-Rnd	75	164	120	1971.75	2053.05	4.12	195.62	2465.74	25.05	–	2464.34	24.98	–
21-3-Mix	75	164	120	1968.64	2134.77	8.44	191.60	2374.29	20.61	–	2407.09	22.27	–
21-3-Pur	75	164	120	1595.27	1668.42	4.59	158.80	1869.91	17.22	–	1907.51	19.57	–
21-4-Rnd	75	168	120	1743.98	1817.47	4.21	207.66	2341.08	34.24	–	2323.37	33.22	–
21-4-Mix	75	168	120	1815.20	1871.71	3.11	209.61	2174.64	19.80	–	2244.16	23.63	–
21-4-Pur	75	168	120	1492.70	1516.72	1.61	155.89	1780.08	19.25	–	1811.23	21.34	–
21-5-Rnd	75	202	120	1630.90	1659.14	1.73	280.62	2237.01	37.16	–	2250.07	37.96	–
21-5-Mix	75	202	120	1728.91	1765.75	2.13	262.50	2067.03	19.56	–	2154.49	24.62	–
21-5-Pur	75	202	120	1321.19	1365.06	3.32	231.73	1607.64	21.68	–	1669.15	26.34	–
25-2-Rnd	100	157	300	2417.89	2559.98	5.88	200.45	3104.05	28.38	–	3106.46	28.48	–
25-2-Mix	100	157	300	2223.29	2380.69	7.08	199.64	2948.81	32.63	–	3081.46	38.60	–
25-2-Pur	100	157	300	2121.95	2254.02	6.22	130.20	2582.53	21.71	–	2646.27	24.71	–
25-3-Rnd	100	212	300	2423.59	2528.47	4.33	206.07	3105.41	28.13	–	3098.09	27.83	–
25-3-Mix	100	212	300	2196.52	2274.95	3.57	242.74	2906.68	32.33	–	3045.32	38.64	–
25-3-Pur	100	212	300	2154.88	2241.20	4.01	165.41	2522.70	17.07	–	2584.63	19.94	–
25-4-Rnd	100	254	300	2471.34	2545.58	3.00	191.98	3088.07	24.95	–	3095.72	25.26	–
25-4-Mix	100	254	300	2186.82	2269.58	3.78	231.34	2886.40	31.99	–	3035.13	38.79	–

25-4-Pur	100	254	300	2166.21	2237.66	3.30	149.58	2552.36	17.83	–	2610.30	20.50	–
25-5-Rnd	100	311	300	2146.53	2185.33	1.81	269.22	2946.79	37.28	–	2929.42	36.47	–
25-5-Mix	100	311	300	1981.35	2013.81	1.64	308.35	2733.71	37.97	–	2889.13	45.82	–
25-5-Pur	100	311	300	1900.61	1932.41	1.67	236.32	2284.15	20.18	–	2392.03	25.86	–
30-2-Rnd	150	225	900	3212.91	3357.04	4.49	192.59	4125.77	28.41	–	4127.65	28.47	–
30-2-Mix	150	225	900	2941.75	3085.51	4.89	246.16	3888.60	32.19	–	3989.23	35.61	–
30-2-Pur	150	225	900	2901.00	3064.66	5.64	136.23	3520.50	21.35	–	3583.79	23.54	–
30-3-Rnd	150	298	900	3319.29	3458.27	4.19	195.97	4218.74	27.10	–	4224.18	27.26	–
30-3-Mix	150	298	900	3085.68	3237.63	4.92	232.76	3999.59	29.62	–	4158.31	34.76	–
30-3-Pur	150	298	900	3141.39	3246.05	3.33	144.04	3656.44	16.40	–	3699.78	17.78	–
30-4-Rnd	150	366	900	3323.61	3432.26	3.27	206.47	4189.11	26.04	–	4181.14	25.80	–
30-4-Mix	150	366	900	3038.70	3126.83	2.90	229.36	3910.82	28.70	–	4021.36	32.34	–
30-4-Pur	150	366	900	3071.03	3195.58	4.06	146.64	3640.62	18.55	–	3656.72	19.07	–
30-5-Rnd	150	433	900	2955.31	3051.51	3.26	275.64	3889.38	31.61	–	3892.39	31.71	–
30-5-Mix	150	433	900	2660.34	2700.66	1.52	322.03	3577.25	34.47	–	3731.25	40.25	–
30-5-Pur	150	433	900	2702.15	2712.68	0.39	224.58	3154.68	16.75	–	3217.27	19.06	–
Average						3.91	198.96		24.78			28.41	

Table 14: Results (travel distances) for different variants of 2L-PDP ("NoRotate" variant, complete results).

Instance				**1D**	**Unrestricted**			**Simultaneous Packing**			**Independent Partial Routes**		
name	**req. n**	**items m**	**CPU sec**	**avg-ttd**	**avg-ttd**	**gap %**	**reloading effort**	**avg-ttd**	**gap %**	**reloading effort**	**avg-ttd**	**gap %**	**reloading effort**
09-2-Rnd	25	40	30	704.24	762.44	8.27	136.43	977.09	38.74	–	977.09	38.74	–
09-2-Mix	25	40	30	828.46	951.79	14.89	148.06	1065.49	28.61	–	1107.33	33.66	–
09-2-Pur	25	40	30	827.21	952.69	15.17	99.29	1004.77	21.47	–	1064.96	28.74	–
09-3-Rnd	25	61	30	727.29	752.41	3.45	162.73	963.25	32.44	–	963.25	32.44	–
09-3-Mix	25	61	30	881.69	977.87	10.91	161.75	1086.04	23.18	–	1147.82	30.18	–
09-3-Pur	25	61	30	829.47	953.98	15.01	137.09	1014.14	22.26	–	1070.50	29.06	–
09-4-Rnd	25	63	30	709.10	737.21	3.96	152.77	958.60	35.19	–	958.60	35.19	–
09-4-Mix	25	63	30	847.03	989.50	16.82	87.39	1074.08	26.81	–	1111.18	31.19	–
09-4-Pur	25	63	30	848.14	950.02	12.01	80.60	1006.88	18.72	–	1013.73	19.52	–
09-5-Rnd	25	91	30	706.07	716.79	1.52	189.33	869.40	23.13	–	869.40	23.13	–
09-5-Mix	25	91	30	821.05	823.11	0.25	188.69	957.04	16.56	–	1098.42	33.78	–
09-5-Pur	25	91	30	791.58	815.46	3.02	162.02	877.58	10.86	–	970.54	22.61	–
19-2-Rnd	50	82	60	1123.94	1256.20	11.77	198.62	1651.75	46.96	–	1657.00	47.43	–
19-2-Mix	50	82	60	1434.45	1561.96	8.89	195.70	1782.44	24.26	–	1897.36	32.27	–
19-2-Pur	50	82	60	1087.98	1203.09	10.58	143.66	1348.33	23.93	–	1400.42	28.72	–
19-3-Rnd	50	103	60	1188.44	1258.80	5.92	201.45	1650.19	38.85	–	1651.61	38.97	–
19-3-Mix	50	103	60	1484.52	1586.18	6.85	186.91	1824.13	22.88	–	1870.79	26.02	–
19-3-Pur	50	103	60	1099.54	1198.98	9.04	154.32	1354.76	23.21	–	1437.80	30.76	–
19-4-Rnd	50	134	60	1213.39	1275.84	5.15	186.71	1659.92	36.80	–	1666.77	37.36	–
19-4-Mix	50	134	60	1480.96	1591.69	7.48	173.25	1819.19	22.84	–	1932.04	30.46	–
19-4-Pur	50	134	60	1138.63	1247.50	9.56	126.07	1406.21	23.50	–	1455.46	27.83	–
19-5-Rnd	50	157	60	1060.30	1093.82	3.16	276.44	1563.10	47.42	–	1567.80	47.86	–
19-5-Mix	50	157	60	1304.16	1352.91	3.74	228.17	1578.70	21.05	–	1792.71	37.46	–

19-5-Pur	50	157	60	976.34	1009.30	3.38	189.50	1169.67	19.80	–	1242.20	27.23	–
21-2-Rnd	75	114	120	1767.76	1914.73	8.31	219.04	2427.36	37.31	–	2450.61	38.63	–
21-2-Mix	75	114	120	1799.79	1951.98	8.46	226.31	2251.38	25.09	–	2297.14	27.63	–
21-2-Pur	75	114	120	1409.95	1565.43	11.03	166.22	1840.23	30.52	–	1909.73	35.45	–
21-3-Rnd	75	164	120	1971.75	2089.50	5.97	189.03	2519.17	27.76	–	2521.73	27.89	–
21-3-Mix	75	164	120	1968.64	2177.43	10.61	190.37	2468.48	25.39	–	2514.22	27.71	–
21-3-Pur	75	164	120	1595.27	1712.05	7.32	151.83	1947.55	22.08	–	1979.70	24.10	–
21-4-Rnd	75	168	120	1743.98	1842.86	5.67	186.36	2380.33	36.49	–	2378.51	36.38	–
21-4-Mix	75	168	120	1815.20	1919.86	5.77	192.46	2211.61	21.84	–	2271.51	25.14	–
21-4-Pur	75	168	120	1492.70	1551.43	3.93	134.09	1830.46	22.63	–	1866.97	25.07	–
21-5-Rnd	75	202	120	1630.90	1693.52	3.84	242.49	2266.97	39.00	–	2269.90	39.18	–
21-5-Mix	75	202	120	1728.91	1803.61	4.32	248.00	2107.93	21.92	–	2164.75	25.21	–
21-5-Pur	75	202	120	1321.19	1399.67	5.94	197.78	1638.30	24.00	–	1694.79	28.28	–
25-2-Rnd	100	157	300	2417.89	2678.67	10.79	177.22	3248.80	34.36	–	3260.42	34.85	–
25-2-Mix	100	157	300	2223.29	2516.00	13.17	189.28	3048.13	37.10	–	3215.23	44.62	–
25-2-Pur	100	157	300	2121.95	2384.71	12.38	145.85	2719.28	28.15	–	2789.91	31.48	–
25-3-Rnd	100	212	300	2423.59	2631.36	8.57	209.89	3224.03	33.03	–	3215.21	32.66	–
25-3-Mix	100	212	300	2196.52	2368.35	7.82	212.11	2991.48	36.19	–	3130.54	42.52	–
25-3-Pur	100	212	300	2154.88	2331.26	8.19	155.34	2676.20	24.19	–	2732.23	26.79	–
25-4-Rnd	100	254	300	2471.34	2620.74	6.05	176.39	3165.64	28.09	–	3155.20	27.67	–
25-4-Mix	100	254	300	2186.82	2289.26	4.68	211.66	2956.18	35.18	–	3090.57	41.33	–
25-4-Pur	100	254	300	2166.21	2309.21	6.60	137.48	2667.44	23.14	–	2720.49	25.59	–
25-5-Rnd	100	311	300	2146.53	2210.69	2.99	237.91	2976.51	38.67	–	2983.29	38.98	–
25-5-Mix	100	311	300	1981.35	2052.81	3.61	274.93	2783.52	40.49	–	2908.70	46.80	–
25-5-Pur	100	311	300	1900.61	1963.95	3.33	188.71	2289.93	20.48	–	2419.58	27.31	–
30-2-Rnd	150	225	900	3212.91	3475.64	8.18	175.50	4265.40	32.76	–	4257.73	32.52	–
30-2-Mix	150	225	900	2941.75	3261.25	10.86	206.96	4051.89	37.74	–	4151.40	41.12	–
30-2-Pur	150	225	900	2901.00	3208.83	10.61	131.48	3653.69	25.95	–	3676.27	26.72	–
30-3-Rnd	150	298	900	3319.29	3575.52	7.72	181.68	4320.93	30.18	–	4329.80	30.44	–
30-3-Mix	150	298	900	3085.68	3365.50	9.07	207.16	4138.23	34.11	–	4252.08	37.80	–
30-3-Pur	150	298	900	3141.39	3362.66	7.04	142.79	3825.63	21.78	–	3864.27	23.01	–
30-4-Rnd	150	366	900	3323.61	3525.60	6.08	197.54	4324.26	30.11	–	4291.85	29.13	–
30-4-Mix	150	366	900	3038.70	3219.18	5.94	217.04	4042.58	33.04	–	4161.23	36.94	–
30-4-Pur	150	366	900	3071.03	3289.39	7.11	141.92	3777.89	23.02	–	3799.13	23.71	–
30-5-Rnd	150	433	900	2955.31	3092.13	4.63	247.94	3931.38	33.03	–	3909.83	32.30	–
30-5-Mix	150	433	900	2660.34	2732.56	2.71	299.78	3637.44	36.73	–	3809.21	43.19	–
30-5-Pur	150	433	900	2702.15	2770.56	2.53	183.80	3238.91	19.86	–	3287.70	21.67	–
Average						**7.38**	**182.69**		**28.68**			**32.17**	

Table 15: Total iteration numbers and computing times to find the best solution ("Rotate" variant, complete results).

Instance				**Independent Partial Routes**			**Simultaneous Packing**			**Ratio iterations**
name	**req. n**	**items m**	**CPU sec**	**Runtime to best**	**Runtime to best in %**	**Total iterations**	**Runtime to best**	**Runtime to best in %**	**Total iterations**	
09-2-Rnd	25	40	30	3.14	10.47	677510	4.13	13.77	301744	0.45

09-2-Mix	25	40	30	1.79	5.97	859628	1.87	6.23	343621	0.40
09-2-Pur	25	40	30	6.86	22.87	693085	4.14	13.80	305032	0.44
09-3-Rnd	25	61	30	7.66	25.53	676485	3.46	11.53	296549	0.44
09-3-Mix	25	61	30	3.42	11.40	870142	1.94	6.47	356203	0.41
09-3-Pur	25	61	30	5.64	18.80	735637	7.82	26.07	324160	0.44
09-4-Rnd	25	63	30	3.69	12.30	658966	3.80	12.67	293943	0.45
09-4-Mix	25	63	30	1.50	5.00	858187	5.32	17.73	358785	0.42
09-4-Pur	25	63	30	9.92	33.07	694753	5.42	18.07	315089	0.45
09-5-Rnd	25	91	30	2.50	8.33	660934	0.93	3.10	256483	0.39
09-5-Mix	25	91	30	7.54	25.13	869122	2.54	8.47	335365	0.39
09-5-Pur	25	91	30	4.03	13.43	703858	1.78	5.93	253284	0.36
19-2-Rnd	50	82	60	18.15	30.25	301668	27.04	45.07	121480	0.40
19-2-Mix	50	82	60	29.48	49.13	377188	7.55	12.58	147204	0.39
19-2-Pur	50	82	60	14.77	24.62	276187	21.63	36.05	95434	0.35
19-3-Rnd	50	103	60	25.65	42.75	307649	26.88	44.80	124157	0.40
19-3-Mix	50	103	60	9.89	16.48	365401	10.74	17.90	154127	0.42
19-3-Pur	50	103	60	15.24	25.40	286669	35.86	59.77	97344	0.34
19-4-Rnd	50	134	60	18.80	31.33	291448	16.73	27.88	123299	0.42
19-4-Mix	50	134	60	22.12	36.87	369146	29.31	48.85	150056	0.41
19-4-Pur	50	134	60	18.70	31.17	277825	24.69	41.15	101135	0.36
19-5-Rnd	50	157	60	12.59	20.98	227995	16.64	27.73	98035	0.43
19-5-Mix	50	157	60	9.83	16.38	342502	23.82	39.70	113064	0.33
19-5-Pur	50	157	60	16.21	27.02	232114	10.83	18.05	75663	0.33
21-2-Rnd	75	114	120	55.08	45.90	226599	62.27	51.89	99188	0.44
21-2-Mix	75	114	120	55.68	46.40	269231	69.27	57.73	94611	0.35
21-2-Pur	75	114	120	59.56	49.63	190448	76.24	63.53	62201	0.33
21-3-Rnd	75	164	120	57.04	47.53	265594	58.88	49.07	119277	0.45
21-3-Mix	75	164	120	74.63	62.19	276408	75.12	62.60	113066	0.41
21-3-Pur	75	164	120	79.77	66.48	208837	78.32	65.27	72981	0.35
21-4-Rnd	75	168	120	36.28	30.23	220580	59.64	49.70	89828	0.41
21-4-Mix	75	168	120	72.46	60.38	232932	80.12	66.77	75745	0.33
21-4-Pur	75	168	120	91.05	75.88	160459	80.79	67.33	49184	0.31
21-5-Rnd	75	202	120	80.05	66.71	179722	52.74	43.95	82425	0.46
21-5-Mix	75	202	120	40.37	33.64	205508	92.29	76.91	69756	0.34
21-5-Pur	75	202	120	58.39	48.66	151651	68.93	57.44	38842	0.26
25-2-Rnd	100	157	300	166.90	55.63	360736	211.72	70.57	153021	0.42
25-2-Mix	100	157	300	186.16	62.05	396084	188.72	62.91	150189	0.38
25-2-Pur	100	157	300	179.73	59.91	315825	186.68	62.23	120034	0.38
25-3-Rnd	100	212	300	211.22	70.41	346793	225.58	75.19	151316	0.44
25-3-Mix	100	212	300	128.54	42.85	381004	189.64	63.21	148064	0.39
25-3-Pur	100	212	300	198.51	66.17	313848	181.15	60.38	124748	0.40
25-4-Rnd	100	254	300	163.23	54.41	329294	141.33	47.11	151164	0.46
25-4-Mix	100	254	300	139.60	46.53	372671	184.21	61.40	144303	0.39
25-4-Pur	100	254	300	249.57	83.19	288791	156.02	52.01	116367	0.40
25-5-Rnd	100	311	300	182.83	60.94	243853	221.73	73.91	105327	0.43
25-5-Mix	100	311	300	112.10	37.37	338184	135.55	45.18	122609	0.36
25-5-Pur	100	311	300	214.00	71.33	215152	185.72	61.91	82163	0.38
30-2-Rnd	150	225	900	588.85	65.43	387306	748.72	83.19	169808	0.44

30-2-Mix	150	225	900	639.49	71.05	438919	481.07	53.45	160988	0.37
30-2-Pur	150	225	900	650.04	72.23	342012	754.51	83.83	126217	0.37
30-3-Rnd	150	298	900	761.26	84.58	363763	773.34	85.93	175194	0.48
30-3-Mix	150	298	900	528.54	58.73	428842	666.94	74.10	168398	0.39
30-3-Pur	150	298	900	602.89	66.99	341973	737.58	81.95	146540	0.43
30-4-Rnd	150	366	900	626.66	69.63	340395	821.90	91.32	159951	0.47
30-4-Mix	150	366	900	513.93	57.10	416217	539.58	59.95	164887	0.40
30-4-Pur	150	366	900	763.90	84.88	307661	790.42	87.82	122616	0.40
30-5-Rnd	150	433	900	591.70	65.74	270265	525.86	58.43	127891	0.47
30-5-Mix	150	433	900	357.22	39.69	362924	581.38	64.60	122701	0.34
30-5-Pur	150	433	900	552.52	61.39	224843	736.71	81.86	83153	0.37
Average					**44.78**			**48.63**		**0.40**

Otto von Guericke University Magdeburg
Faculty of Economics and Management
P.O. Box 4120 | 39016 Magdeburg | Germany

Tel.: +49 (0) 3 91 / 67-1 85 84
Fax: +49 (0) 3 91 / 67-1 21 20

www.fww.ovgu.de/femm

ISSN 1615-4274