Local search techniques for a routing-packing problem

Sara Ceschia[a,*], Andrea Schaerf[a], Thomas Stützle[b]

[a]*DIEGM, Università degli Studi di Udine*
Via delle Scienze 206, I-33100, Udine, Italy
[b]*IRIDIA, Université Libre de Bruxelles (ULB)*
Av. F. Roosevelt 50, CP 194/6, B-1050 Brussels, Belgium

Abstract

We propose a complex real-world problem in logistics that integrates routing and packing aspects. It can be seen as an extension of the *Three-Dimensional Loading Capacitated Vehicle Routing Problem* (3L-CVRP) introduced by Gendreau et al. (2006). The 3L-CVRP consists in finding a set of routes that satisfies the demand of all customers, minimizes the total routing cost, and guarantees a packing of items that is feasible according to loading constraints. Our problem formulation includes additional constraints in relation to the stability of the cargo, to the fragility of items, and to the loading and unloading policy. In addition, it considers the possibility of *split deliveries*, so that each customer can be visited more than once. We propose a local search approach that considers the overall problem in a single stage. It is based on a composite strategy that interleaves simulated annealing with large-neighborhood search. We test our solver on 13 real-world instances provided by our industrial partner, which are very diverse in size and features. In addition, we compare our solver on benchmarks from the literature of the 3L-CVRP showing that our solver performs well compared to other approaches proposed in the literature.

Keywords: Routing, packing, loading constraints, local search

*Corresponding author

Email addresses: sara.ceschia@uniud.it (Sara Ceschia), schaerf@uniud.it (Andrea Schaerf), stuetzle@ulb.ac.be (Thomas Stützle)

1. Introduction

In different industrial fields, the competitiveness on lead times has pushed companies to advanced computer systems for designing and scheduling integrated logistics. This trend has increased the attention of the academic community to optimization problems that arise from real-world logistic applications. In line with this trend, Gendreau et al. (2006) have introduced the *Three-Dimensional Loading Capacitated Vehicle Routing Problem* (3L-CVRP), which can be considered as a combination of two optimization problems: the *Capacitated Vehicle Routing Problem* (VRP) and the *Three-Dimensional Bin Packing Problem* (3BP). In short, the 3L-CVRP consists in finding a set of routes that satisfies the demand of all customers, minimizes the total routing cost and guarantees a feasible packing of items.

Our project includes many real world features, some of which are described in our previous work on VRP (Ceschia et al., 2011) and *Container Loading* (CLP) (Ceschia & Schaerf, 2013) in isolation, but have never been considered together.

Even though our problem is more complicated than 3L-CVRP, we are also interested in the 3L-CVRP as in this way we can compare our approach to the results of a number of other algorithms specifically designed for the 3L-CVRP on a set of benchmark instances.

As our problem and the 3L-CVRP are a combination of two strongly NP-hard problems, they are very difficult to solve. Hence, it appears to be appropriate to tackle them by using metaheuristic techniques for solving large instances in reasonable computation time (Hoos & Stützle, 2005). In this paper, we propose a local search approach based on a combination of simulated annealing (SA) and large-neighborhood search, which considers the overall problem in one single stage, by including neighborhood relations that modify both the routes and the container loading. The algorithm uses a solution representation based on sequences of items that makes it suitable to deal with split deliveries. It applies three neighborhood operators to modify the current candidate solution. The algorithm is automatically tuned using a new implementation of *Iterated F-Race* (Birattari et al., 2010) by López-Ibáñez et al. (2011).

We test the algorithm on a set of real-world instances, provided by our industrial partner beanTech (`http://www.beantech.it`) and we analyze the relation between performance and instance properties. In addition, we test our algorithm on 3L-CVRP benchmark instances from the literature. In fact,

our solver obtains solutions of better quality than two other algorithms proposed in the literature for the 3L-CVRP. Two more recent algorithms for the 3L-CVRP, however, reach better performance than our solver. Nevertheless, we should consider that our algorithm actually solves a more general problem and that the other algorithms were proposed and fine-tuned specifically for tackling the 3L-CVRP. All instances and best results are available at `http://www.diegm.uniud.it/ceschia/index.php?page=vrclp` for future comparisons.

2. Problem description

We introduce the problem in stages. Starting from the 3L-CVRP (Section 2.1), we describe progressively the new features in order to be able to fully define the complete problem that we deal with (Section 2.2).

2.1. 3L-CVRP Formulation

The 3L-CVRP formulation is provided by Gendreau et al. (2006). We report it here in order to make the paper self-contained and facilitate comparisons to the model we are proposing. The main entities involved in the problem are:

Clients: It is given a set of customers $\mathcal{V} = \{0, 1, \ldots, n\}$, each corresponding to a vertex of the graph $G = (\mathcal{V}, \mathcal{E})$. The depot is treated as a special customer and it is identified with vertex 0. Each edge (i, j) of the graph has an associated cost c_{ij}, which is the cost of traveling from customer i to customer j.

Items: Each customer $i \in \mathcal{V} \setminus \{0\}$ requires a supply of m_i items whose total weight is d_i. The demand of customer i is made up of *items*, that is, three-dimensional rectangular boxes each one having width w_{ik}, height h_{ik} and length l_{ik} ($k = 1, \ldots, m_i$). The total volume needed by customer i is $s_i = \sum_{k=1}^{m_i} w_{ik} h_{ik} l_{ik}$. In addition, a flag f_{ik} is associated to each item, such that if $f_{ik} = 1$ the corresponding item is fragile. The total number of items for all customers is $M = \sum_{i=1}^{n} m_i$.

Vehicles: The delivery of items is performed by a fleet $\mathcal{F} = \{1, \ldots, v\}$ of identical vehicles, each one with weight capacity D. The three-dimensional rectangular loading space has width W, height H, and length L, so the total loading volume available for each vehicle is $S = W \cdot H \cdot L$.

The solution of the 3L-CVRP calls for the determination of a set of routes that minimizes the total transportation cost and satisfies all the following routing and packing constraints:

1. the number of routes must be at most equal to the size of the fleet, i.e. one route per vehicle;
2. each route must start and end at the depot;
3. each customer must be visited exactly once;
4. the demands of all customers must be satisfied;
5. for each route the total demand weight must not exceed the weight capacity of the vehicle;
6. items must be stowed completely in the vehicle;
7. no two items can overlap;
8. the loading must be orthogonal, i.e. the edges of an item must lie parallel to the edges of the vehicle.

Furthermore, for each route the loading must be feasible according to these additional conditions:

Fixed Vertical Orientation (C1): Items can be rotated by 90° on the *width-length* plane, keeping fixed the vertical orientation.

Fragility (C2): Non-fragile items cannot be placed on the top of fragile ones.

Minimum Supporting Area (C3): The base of each item must be supported by other items or by the vehicle's floor at least by a minimum supporting area, which is proportional to the bottom side of the item.

LIFO Policy (C4): The loading order is the inverse of the customers' visit order. In such a way it is possible to unload items of a customer without moving items belonging to other customers that will be visited later. The unloading procedure is performed through straight movements parallel to the length and height plane.

2.2. Complete problem

We now describe the formulation of the complex real-world problem that we tackled, which extends the 3L-CVRP problem in several directions.

2.2.1. Weakly heterogeneous cargo and heterogeneous fleet

Firstly, the 3L-CVRP does not consider the possibility of having a cargo composed by large groups of items of the same dimensions and a heterogeneous fleet. Indeed in the 3L-CVRP benchmarks, there are only a few large items, all different from each other. However, in many real cases, there is a large number of identical items to be loaded. As examples, we refer to the real-world instances described by Ceschia & Schaerf (2013) and those used by Moura & Oliveira (2008). In addition, the fleet may be composed by vehicles that are not identical. This leads to the definition of new problem entities:

Item types: Each item i (with $i = 1, \ldots, M$) belongs to an item type j (with $j = 1, \ldots, it$), which is characterized by: dimensions (w_j, l_j, h_j), allowed rotations (uw_j, ul_j, uh_j), number of items of this type (μ_j), weight (wg_j) and bearable weight for each face (bw_j, bl_j, bh_j).

The allowed rotations are denoted by the binary-valued parameters uw_j, ul_j, uh_j, which are 1 if the corresponding side can be positioned upright, and 0 otherwise. The bearing weight is the maximum load, expressed in units of weight per area, which may be placed on the top surface of the item when the corresponding edge is placed upright. Its role in the problem formulation will be explained in Section 2.2.2.

Vehicle types: Each vehicle i (with $i, \ldots, v$) belongs to a vehicle type j (with $j = 1, \ldots, vt$), which is characterized by dimensions (W_j, L_j, H_j), number of vehicles of this type (ν_j), weight capacity (D_j) and possible fixed cost of use (C_j).

In case of weakly heterogeneous cargo, an efficient loading can be usually obtained by grouping items of the same type in homogeneous blocks (Eley, 2002; Fanslau & Bortfeldt, 2010). Nevertheless, in our model items belonging to different customers can be of the same type, thus a block homogeneous with respect to the item type could be heterogeneous with respect to deliveries (i.e. customers). Therefore the packing strategy has to be modified in order to take into account different deliveries for a single block, so as to not violate the C4 constraint.

2.2.2. Load bearing strength

The definition of item fragility and the corresponding constraint C2 is rather simplistic. It needs to be reformulated by means of introducing the

concept of maximum supported weight (that could be proportional to the weight of the item). Indeed, Figure 1 highlights a doubtful situation: box 1, which is not fragile, is supported at least for the 75% of its base area by box 3, which is not fragile too. Nevertheless, box 2 is as high as box 3 and it is pushed near box 3 and under box 1. This placement generates a fragility violation because box 2 is fragile and box 1 is not, although it could reasonably be expected that box 1 is mainly supported by box 3, independently of box 2.

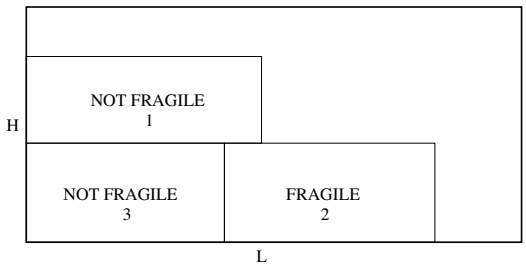


Figure 1: Violation of the Fragility constraint.

Therefore, the qualitative notion of fragility is replaced by the quantitative one of bearing strength, which leads us to replace the C2 constraint with the following one:

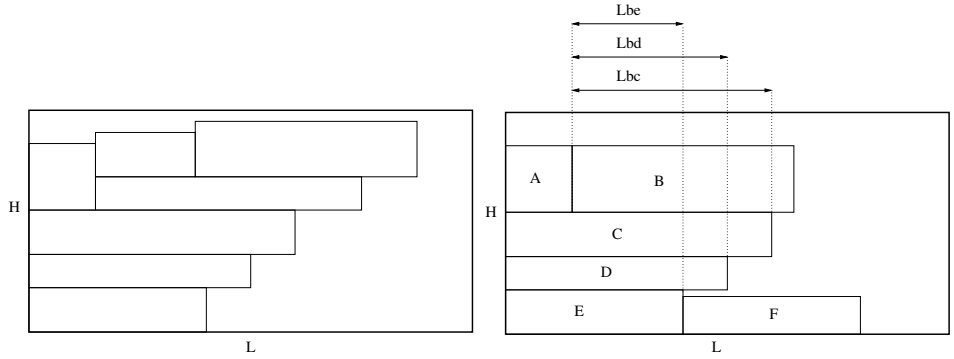
Load Bearing Strength (C5): There is a maximum weight per unit area that a box can uphold depending on its type and its vertical orientation.

2.2.3. Cargo stability

The proposed definition of stability based only on the minimum supporting area between one item and the underlying one can lead to skewed item stacks that are actually unstable (see Figure 2a). Therefore, the notion of stability has been reformulated and the constraint C3 was changed to:

Robust Stability (C6): A minimum supporting area has to be guaranteed for all items below the current one in the stack.

This means that for each item, the constraint C3 is applied not only to the underlying item, but also to all items below it.



(a) Skewed item stack (b) The Robust Stability constraint

Figure 2: Reformulation of the notion of stability of the cargo.

Figure 2b highlights the difference between the standard Minimum Supporting Area (C3) constraint and the Robust Stability (C6) constraint. Assuming that all items in Figure 2b have the same width, the supporting area depends only on the length, so the label Lij marks the part of the item j that supports item i. The C3 constraint establishes that the stability of box B has to be calculated considering only box C, which is immediately below it. In contrast, the C6 constraint requires that the stability of the box B is computed taking into account items E and D, too. In this case, L_{be} is too short, thus, box B is not stable for C6, while according to C3, L_{bc} is long enough to guarantee stability (L_{bd} and L_{be} are not considered).

2.2.4. The LIFO constraint

The LIFO constraint, as formulated in the 3L-CVRP, is not realistic in the sense that it does not capture correctly the physical constraints that allow the human operator or machine to unload an item without having to move the others.

In detail, the LIFO constraint proposed by Gendreau et al. (2006) establishes that any item of a customer visited later than the current one, must not be placed above a box of the current customer or between it and the rear of the vehicle. A variant of this formulation has already been proposed by Tarantilis et al. (2009) leading to the problem called Manual 3L-CVRP (or M3L-CVRP) in order to capture situations where boxes are manually unloaded. In this case, boxes are not necessarily elevated before pulling them

out of the vehicle, thus it is possible to place an item of a subsequent customer above (without contact) a box of the current customer. Figure 3 shows an example of loading that is feasible for the M3L-CVRP and infeasible for the original 3L-CVRP; in fact, the tour represented below the loading area indicates that the first customer to be visited is customer 1, whose box has above it a box of a customer that is later in the tour, customer 2. This situation would incur a violation of the classical LIFO constraint, but it is correct for the M3L-CVRP because a human operator can simply slide out the box without touching the one above.

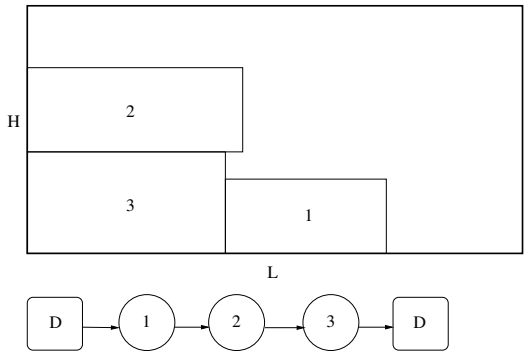


Figure 3: Manual 3L-CVRP: a human operator can simply slide out the box 1 without touching box 2.

The LIFO constraint requires to be further refined in order to deal with real cases. In fact, Figure 4 draws a situation that is considered feasible for both 3L-CVRP and M3L-CVRP, but in reality it is totally unworkable. The designed tour establishes that customer 1 is the first one to be visited, therefore box 1 needs to be unloaded before the others. However, both a human operator and a forklift are not able to unload it without moving the others, since they are at the rear door of the vehicle and box 1 is too far away at the bottom side. In this case, any subsequent placement of the box in a position closer to the rear door would incur in a fragility or stability violation.

We, thus, introduce a new loading constraint:

Reachability (C7): An item is considered reachable if the distance between it and a human operator or a forklift is less than or equal to a fixed length λ. It is supposed that the operator is placed as close as possible

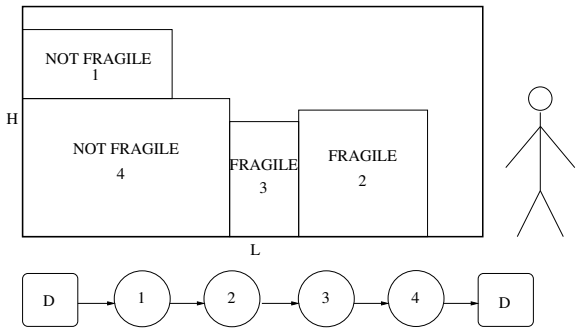


Figure 4: Situation impracticable for unloading box 1

to items inside the vehicle, i.e. the position with the minimum length with respect to the current loading.

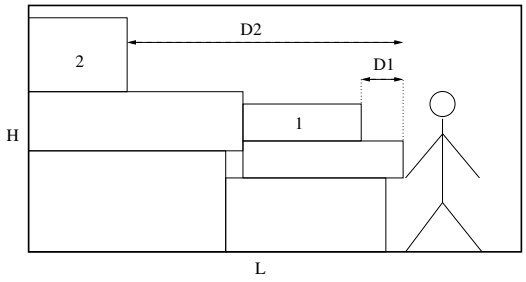


Figure 5: The Reachability constraint

Figure 5 shows the position of an operator inside a vehicle and how distances are computed (D_j is the distance between the operator and item j). In this case, both items 1 and 2 could be unloaded without moving other items, thus this loading does not violate the standard LIFO constraint (C4). However, due to the physical limitations of the operator, only item 1 can be unload, while box 2 is unreachable.

In case of manual unloading λ is set to 50 cm (approximately the length of a human arm), otherwise it is set to a higher value, typically 100 cm. This way, our solver is implicitly able to manage cases where both constraints about reachability and manual unloading are enforced, and cases where only the reachability constraint is applied.

2.2.5. Split deliveries

Finally, as already mentioned, it is possible that a customer has a demand that is greater than the vehicle capacity; in this case, the demand needs to be split and be delivered by more than one vehicle (we do not allow that a customer is visited more than once by the same vehicle).

As it will be clear in Section 4, this is a significant change that leads to a different search space and more complex neighborhood operators. In fact, this routing problem falls into the category of CVRP *with Split Deliveries* (SDVRP), which uses approaches that are quite different from those of the classical CVRP.

3. Related Work

For a recent and comprehensive review on routing problems with loading constraints, the reader should refer to Iori & Martello (2010); we review here only articles that deal with the three dimensional case.

Junqueira et al. (2011) proposed a mixed integer programming model for the *container loading problem with multi-drop constraints*, where the delivery route is given and the task is to pack items into a container respecting the unloading sequence. Their model also reckoned with the constraint related to the reachability of a box by a forklift truck or by a worker's arm, and computational results for different values of the λ parameter were carried out.

Doerner et al. (2007) introduced the *multi-pile vehicle routing problem* (MP-VRP), which stems from a company that delivers timber products. In this case, products that have the same dimensions are grouped together in pallets, which can a have fixed width, but length and height variable within certain values. The objective is to find a loading that respects the sequential constraints and minimizes the routing cost. The problem has been solved through tabu search and ant colony optimization (ACO) (Doerner et al., 2007), and variable neighborhood search and branch-and-cut (Tricoire et al., 2009).

Another practical problem that integrates routing and packing aspects is the *pallet-packing vehicle routing problem* (PPVRP) (Zachariadis et al., 2012). The key difference between the PP-VRP and the other models is that items must first be assigned and packed into pallets (Bischoff & Ratcliff, 1995), which are then loaded in vehicles. Zachariadis et al. developed a solution technique based on tabu search to deal with the routing aspects,

whereas the packing heuristic is related to the one described in Tarantilis et al. (2009). In order to allow to easily unload items, all the demand of a customer must be stacked in the same pallet and the LIFO constraint is not imposed.

The literature on the 3L-CVRP tackles this problem usually by means of a hierarchical solution approach where the master problem, the routing problem is solved by a metaheuristic technique, while the slave problem, the packing problem, is delegated to fast and simple packing heuristics, such as the *bottom left algorithm* (Baker et al., 1980) and the *touching perimeter algorithm* (Lodi et al., 1999).

In the solution technique presented by Gendreau et al. (2006), an outer tabu search algorithm works on the routing problem moving customers from different routes, whereas an inner tabu search routine deals with the packing problem swapping items and iteratively invoking the loading heuristic. The search space includes infeasible states, in which there are violations of the capacity of vehicles (exceeding weight) or the loading space of the vehicle is not sufficient to contain all items.

Tarantilis et al. (2009) implemented a hybrid tabu search-guided local search algorithm. The tabu search explores the routing part, while a bundle of six different packing heuristics are iteratively reapplied until a feasible loading is obtained. If no feasible loading is achieved, the items of the sequence are re-sorted for a maximum of three times (keeping fixed the customer sequence). In addition, as already mentioned, they introduced a new problem version, called *Manual 3L-CVRP*, which was described above.

The ant colony optimization (ACO) approach described by Fuellerer et al. (2010) draws on some of the features of the *Saving-based* ACO (Reimann et al., 2004), which was successfully used to solve the standard CVRP. To test the feasibility of a route as for loading constraints, two greedy packing heuristics are repeatedly applied and, if necessary, swaps among items of the same customer are performed.

Recently, Wang et al. (2010) developed a two phase tabu search algorithm. In the first phase, they consider both hard and soft constraints, allowing the search process to go through infeasible states as in (Gendreau et al., 2006); in the second phase, they apply five different neighborhood operators (*2-opt*, *2-swap*, *move*, *crossover*, *splitting*), and only feasible solutions are generated and evaluated. The classical *bottom left algorithm* has been adapted to perform effectively in the case of the supporting area constraint.

Finally, Bortfeldt (2012) presented a hybrid approach that uses a tabu

search algorithm for routing and a tree search algorithm for packing. In the tabu search algorithm, he distinguishes two phases by applying different neighborhood structures and different quality measures: during the first phase, the aim is to reduce the number of vehicles, while in the second phase the goal is to reduce the total travel distance. The loading of boxes is carried out by means of a tree search procedure that uses a complex system for ranking boxes not already packed and for filtering the potential placement points, which are generated using the *extreme point-based heuristic* of Crainic et al. (2008).

Moura & Oliveira (2008) studied a different integrated vehicle routing and container loading problem. Their problem takes into account several additional real world features, such as customers' time windows, service times, strongly and weakly heterogeneous cargos, cargo's orientation, LIFO constraint and load stability. The objective function is a weighted sum of three components: the number of vehicles, the total travel time, and the wasted space in vehicles. They propose two solution approaches: in the first one, called *sequential method*, they solve simultaneously the routing and the packing problem. In this case, two constraints (the LIFO constraint and the one that states that each customer belongs to exactly one route) are relaxed, thus the problem turns out to be a Split Delivery VRP. The second solution approach follows the *hierarchical method* in the sense that it first solves the routing problem and then tries to pack the items in the vehicles. Moreover, in the hierarchical method all constraints are considered. They extensively discuss the interdependency between the routing and the packing problem in different cases (number of customers per route, heterogeneity of cargo, density of goods). Their conclusion is that in case of routes with many customers, the routing aspects dominate the loading ones; on the other hand, when the demand of a customer is large so that it fills a big part of the vehicle, the loading problem becomes important. Finally,an integrated solution approach is more appropriate when the number of customers per route is small and the demand is weakly heterogeneous.

To our knowledge, the work by Moura & Oliveira (2008) is the only one that considers the possibility of splitting the customer's demand in a routing-packing problem context. On the other hand, there is a large body of literature on the approaches used to solve the VRP with Split Deliveries (SDVRP). The SDVRP was introduced by Dror & Trudeau (1989) who solved it by a heuristic algorithm, showing substantial savings obtained by allowing split deliveries. In a subsequent work, Dror et al. (1994) studied an

integer linear programming (ILP) formulation and proposed a branch and bound algorithm. Subsequently, Archetti et al. have extensively analyzed the problem (Archetti et al., 2006a, 2008a, 2011) and solved (Archetti et al., 2006b, 2008b) it by using a tabu search (TS) algorithm and a hybrid heuristics approach, respectively.

Our problem formulation considers some additional issues that have been already introduced in the literature of *three dimensional packing* and *container loading* problems. Ratcliff & Bischoff (1998), Davies (2000), Bischoff (2006), and Junqueira et al. (2012) have studied the case of boxes with different load bearing strengths; Bischoff & Ratcliff (1995), Davies & Bischoff (1999), Bortfeldt & Gehring (2001), Eley (2002), de Castro Silva et al. (2003), and Junqueira et al. (2012) dealt with the problem of ensuring the stability of the cargo in a static and/or dynamic environment.

4. Solution Technique

In this section, we present our solution technique, which is based on local search. Firstly, we define the basic elements of local search, i.e. the search space, the cost function, the initial solution, and the neighborhood relations; then we briefly describe the metaheuristic technique we used, namely a combination of simulated annealing and large neighborhood search.

4.1. Search space

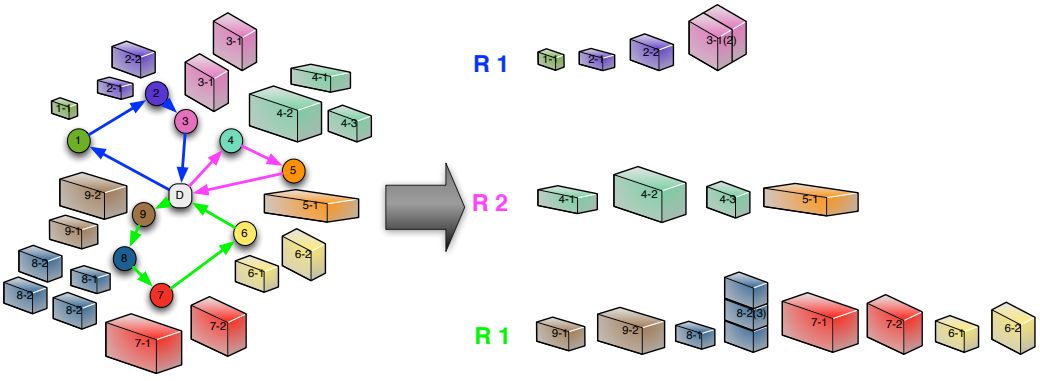


Figure 6: The search space consist of sequences of blocks of boxes.

Differently than the approaches to the 3L-CVRP described in Section 3 (Gendreau et al., 2006; Tarantilis et al., 2009; Fuellerer et al., 2010; Wang et al., 2010; Bortfeldt, 2012), we design a one-stage local search solver, which works on the items-vehicles space rather than on the customers-vehicles one. The solver is responsible for

- generating the customers' routes,
- setting the order in which the items of each sequence will be loaded, and
- selecting the loading heuristic that will be used to determine the actual loading of items in the vehicle.

A state in the search space is a set of sequences, one for each vehicle, where each element of a sequence is an item or a block of items of the same type (see Figure 6 for an illustration). Items belonging to the same customer are consecutive, in such a way each items' sequence corresponds to a unique customers' sequence.

4.2. Cost function

The cost function is a weighted sum of the objective function and the *distance to feasibility*. In fact, the solver can accept a move that leads to an infeasible state, although violations of hard constraints are penalized by multiplying their degree of violation with a suitably high weight.

We allow two kinds of unfeasibility: *exceeding weight* (H1) and *unloaded volume* (H2). The first one represents the total weight that exceeds each vehicle's capacity and the second one represents the total volume of items that do not fit into the vehicles. As already proposed by Gendreau et al. (2006), we set the value of the weight of H1 in dependency of the size of the instance and we fix it to $\texttt{HW} \cdot 100 \cdot \bar{c}/D$, where $\bar{c}$ denotes the average edge distance and HW is a multiplicative constant. Conversely, the weight of H2 is set to HW. Preliminary experiments have shown that the best choice for HW is 3, thus, this value has been used during all experimentation. All other hard constraints are enforced by construction.

4.3. Initial solution

The initial solution is generated at random. Firstly, we randomly select a customer, whose is then assigned to a single random vehicle. Then, we

iteratively select at random a box type and we create a block of homogeneous items. We append it, with a random rotation, to the sequence of blocks of the selected vehicle. After that, we rebuild from each sequence of blocks the corresponding sequence of customers.

4.4. Neighborhood relations

We implemented the following three neighborhood relations:

Move Customer & Change Strategy (MCCS): This neighborhood is defined by the removal of a customer from a route and its insertion in another one in a specific position. As a consequence, all the blocks of items belonging to the selected customer are removed from the old sequence and then inserted in the new one, keeping fixed their relative position and their orientation. In addition, the loading strategy of the new route can be changed. A MCCS move is identified by six attributes $\langle c, or, op, nr, np, st\rangle$ where c represents a customer, or and op the old route and the old position in the old route, and np and nr, the new route and new position in the new route, respectively. Lastly, st is the loading strategy selected for the new route. If the customer is not moved (i.e., if $or = nr \wedge op = np$), an MCCS move performs only a change of strategy.

Swap Customers (SC): This neighborhood is defined by exchanging a customer with another one. A move m of type SC is identified by six attributes $\langle c_1, c_2, r_1, r_2, p_1, p_2\rangle$ where c_1 and c_2 are customers, r_1 and r_2 are routes, p_1 and p_2 are the positions of the customers in the corresponding route. Blocks of items belonging to the customers involved in the move are swapped keeping fixed their reciprocal position and their orientation. If r_1 is the same as r_2, the move is an intra-route exchange.

Move & Rotate Block (MRB): This neighborhood is defined by the removal of a block of homogeneous items from the blocks' sequence of a route and its insertion, possibly with a new orientation, into another route. A move of MRB type is represented by 7 attributes $\langle bt, or, op, nr, np, r, q\rangle$, where bt is the box type identifier, or and op are the old sequence and the old position in the old sequence, np and nr, the new sequence and new position in the new sequence, r is the rotation and q is the number of boxes of the block that are involved in the move. A block can be completely removed from the old route, or

just partially. In the latter case, the original block is divided in two: the past that remains in the old route, and a new block composed by q boxes, which is inserted in the new route.

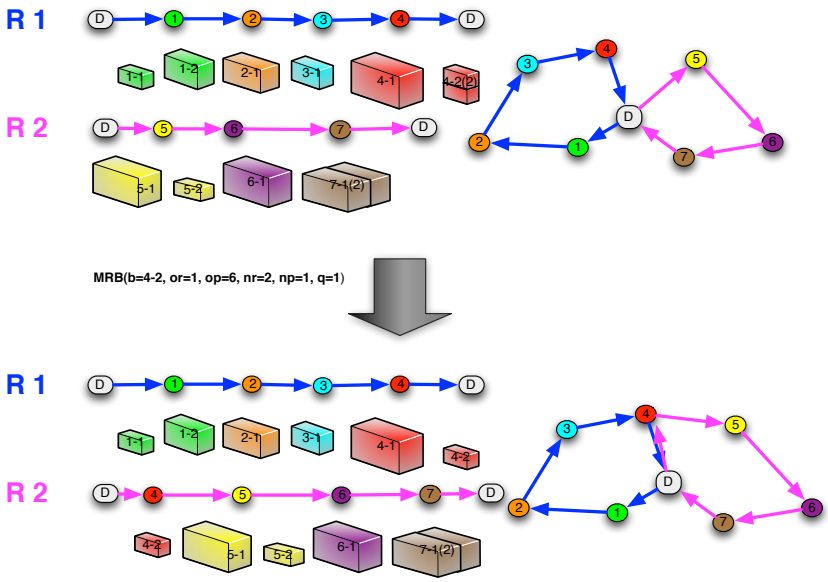


Figure 7: An example of an MRB move that leads to a split delivery.

Note that if the new route is different from the old one, an MRB move split of the demand. In Figure 7, a single item of customer 4 is moved from route 1 to route 2, thus the demand is split and the customer is visited twice.

Once a move is performed, a packing heuristic is called for the blocks' sequences involved in the move. The packing heuristic takes care of all the constraints about the loading and returns the total volume loaded. If some items do not fit in the vehicle, their volume contributes to the cost component H2. The packing heuristic does not modify the blocks' sequence, thus, the C4 constraint is always guaranteed.

4.5. Loading heuristics

We have implemented nine loading heuristics. Eight of these are variants of the extension to the three-dimensional case of the *bottom left algorithm* (Baker et al., 1980) and the *touching perimeter algorithm* (Lodi et al., 1999). These heuristics individuate the *normal position* (Christofides & Whitlock,

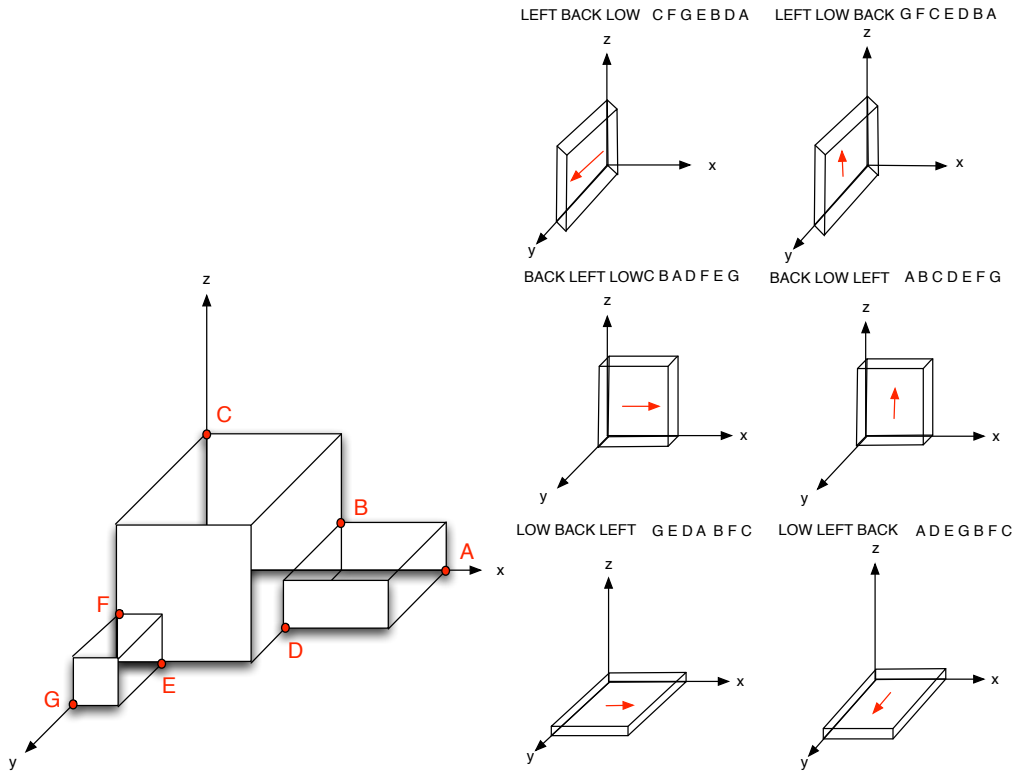
1977), i.e., with the item's bottom edge touching either the floor of the vehicle or the top edge of another item, and its left edge touching either the left edge of the vehicle or the right edge of another item. The detection of normal positions in the three-dimensional case is not trivial (see Martello et al., 2000; Crainic et al., 2008; Martello et al., 2007); for this task we applied the procedure proposed by Martello et al. (2000) with some modifications that take into account the C3 constraint.

For each item, the choice of the packing position among all the possible normal positions in a vehicle depends on the specific strategy. To give an intuition of how each strategy works, in Figure 8a we introduce an easy example of loading with only three items: all the normal positions are labeled with capital letters. The difference between strategies is in the order in which the normal positions are tested. In this example there are seven candidate normal positions (A, B, C, D, E, F, G) where it is possible to place a new item. If we select the Back Low Left strategy, the first position that will be tested is A; if it this placement is feasible (all loading constraints are satisfied), the box is placed with its bottom left corner in A. Otherwise, the box is rotated and the we check if the A position is now feasible. In case of infeasibility, the next normal position (B) is tested with the box in the original orientation, and so on.

For the strategies based on the *bottom left algorithm* (Back Left Low, Back Low Left, Low Back Left, Low Left Back, Left Low Back, Left Back Low), Figure 8b describes the order of selection of candidate positions and the direction of loading (vertical walls, horizontal layers, from the left side, from the back ...). The feasibility of a position is evaluated with the current orientation; if it is not feasible, a rotation on the width-length plane is applied to the item.

The Area and Area No Walls strategies derive from the *touching perimeter algorithm*: the first one selects the position with highest score, defined as the percentage of the item's area that touches either the vehicle and other items already packed; the second one does not consider the wall of vehicles in the score. Each position is evaluated only for the current orientation because the candidate positions are ordered for decreasing score, which is computed considering the current orientation.

The other packing heuristic (Wall Building) is based on the *wall building approach* introduced by George & Robinson (1980) for the *Container Loading Problem*. The idea is to fill the vehicle loading space in a number of layers across the depth of it. Every layer is divided in vertical stacks and each stack is then packed by consecutively inserting items, keeping fixed the loading



(a) Normal positions (b) Order of selection of candidate positions

Figure 8: The Back Left Low, Back Low Left, Low Back Left, Low Left Back, Left Low Back, Left Back Low strategies.

order of the blocks. The depth of a layer is set to be the depth of the first item that is inserted in that layer. For all the following items, if they do not fit in that depth, they are moved to a new layer.

4.6. Metaheuristics

We use a sequential solving approach that alternates a simulated annealing algorithm and a large-neighborhood search, called *intensifier*. The algorithm control is illustrated in the style of the *Generalized Local Search Machines* (GLSM) proposed by Hoos (1998). Nodes represent search strategies and arrows represent transitions of the control from one search strategy to another. Details of the strategies are explained next. Figure 9 illustrates the control of the algorithm. `F` denotes the value of the cost function of the best solution found in the current state, whereas `F_best` refers to the best solution found so far in the overall solution process. Each component starts from the best solution of the previous one, and the overall process makes a fixed number R of rounds.

In the SA component, at each iteration a random neighbor is selected. A move is performed if it is an improving one. Otherwise, a move is accepted with probability $e^{-\Delta F/T}$, where T is a time-decreasing parameter called temperature and ΔF is the cost of a move. At each temperature level, a number σ_N of neighbors of the current solution are sampled. The value of T is modified using a geometric schedule, i.e., $T' = \alpha \cdot T$, in which the parameter α, $0 < \alpha < 1$, is called the cooling rate. The search starts at temperature T_0 and stops when it reaches T_{min}.

Starting from the second time the SA component is executed, the initial temperature is not set to T_0 but to a lower value, in order to avoid to "destroy" completely the previous solution. We therefore define a new parameter, called ρ, such that the initial temperature of all execution of SA but the first is set to T_0/ρ. The SA component uses as neighborhood the union of MCCS, MRB, and SC moves. The random selection prescribed by SA is performed selecting first the neighborhood to be used, and then the specific move.

In order to improve the effectiveness of the SA component, it is useful to set the weight of the hard constraint violation `HW` to a relatively low value, given that the selection relies on ΔF. As a consequence, it is possible that the whole process terminates with a solution that has violations of hard constraints even though a feasible solution could be reached. For this reason, we introduce a new component in the process, called *REPAIR* in Figure 9,

whose aim is to eliminate violations. In the *REPAIR* state, HW is set to a particularly high value and a simple hill climbing (HC) algorithm is invoked.

We also apply the special-purpose operator, the intensifier, that uses as neighborhood a composition of the neighborhoods MCCS and MRB. At each iteration, the neighborhood composed by one move of MCCS and one of MRB is explored and the first improving move is selected. In order to reduce the size of the composite neighborhood, we consider only pairs of moves that concern the same customer.

As shown in Figure 9, the intensifier is launched only when SA gives no improvement in the cost function. This choice is motivated by the fact that the intensifier is computationally expensive and therefore it should be launched only when SA is clearly stuck. Once it has been launched, the intensifier is invoked iteratively as long as it finds improvements in the cost function.

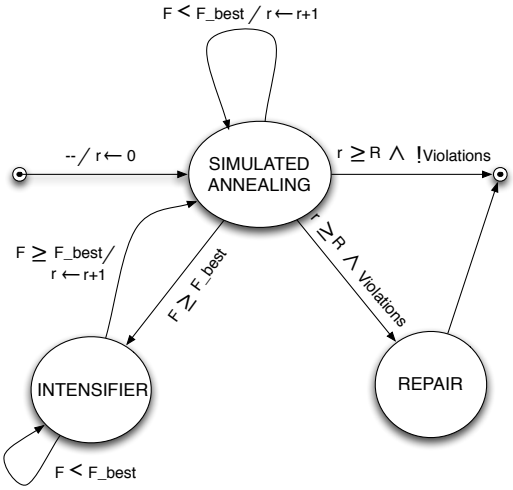


Figure 9: A GLSM representation of the control flow of our algorithm. See the text for more details.

5. Experimental analysis

In this section, we first describe the setting of the parameters used in the experiments. Then we give a description of our new instances and we show our results. Lastly, we compare our approach with the best ones in

the literature on the 3L-CVRP, using the instances introduced by Gendreau et al. (2006) as benchmarks.

5.1. Software Tools and Environment

The software is written in *C++*, it uses the framework EasyLocal++ (Di Gaspero & Schaerf, 2003), and it is compiled using the GNU C/C++ compiler, v. 4.4.3. All experiments have been performed on a 2.66Ghz quad-core PC with 4 GB RAM, running Ubuntu Linux x86_64 (rel. 10.04). For the automatic tuning of the solver we use the *irace* package, provided by López-Ibáñez et al. (2011).

5.2. Parameter Settings

The SA algorithm has five parameters to tune: start temperature T_0, stop temperature T_{min}, cooling rate α, the number of neighbors sampled at each temperature σ_N, and restart temperature ratio ρ. Moreover, given that the solver uses different neighborhoods, we add two parameters γ_{MCCS} and γ_{MRB}, which are the probability of drawing a move of type MCCS and MRB, respectively. The probability of drawing a move of type SC is set to $\gamma_{\mathsf{SC}} = 1 - \gamma_{\mathsf{MCCS}} - \gamma_{\mathsf{MRB}}$.

We decide to tune $\delta = T_0/T_{min}$ instead of T_{min}, which turned out to provide a better distribution of the configurations than using T_{min} directly. In addition, in order to give a similar amount of time to the same instance for each parameter configuration, we let the parameters δ and α vary, and we compute σ_N in such a way to have for the SA component exactly the same number of SA iterations I for each benchmark. In detail, the number of neighbors sampled for each temperature is $\sigma_N = -I/\log_\alpha(\delta)$. For each instance, being v the number of vehicles and n the number of customers, the total number of iterations granted is $I = 10^4 \times v \times n$. The full solver stops when it has performed four rounds or one *idle* round, i.e., a round without an improvement of the best result.

Preliminary experiments show that α is not significant. This is not surprising, because in our setting σ_N is a function of the other parameters. Therefore, α only determines the entity of the single step in the temperature and not the actual slope of the cooling trajectory, which is determined by δ. We therefore set α to the fixed value 0.9999. We also experimented with different values for the probabilities $\gamma_{\mathsf{MCCS}}, \gamma_{\mathsf{MRB}}$, but there was no strong evidence in favor of some values. We therefore set $\gamma_{\mathsf{MCCS}} = 0.4$, and γ_{MRB} and γ_{SC} to 0.3.

For the benchmark instances, we set a tuning budget for I/F-Race of 4000 experiments. We use the following domains for the parameter settings: $T_0 \in [10, 10^5]$, $\delta \in [10^2, 10^6]$ and $\rho \in [10^{-1}, 10^3]$, and the set of training instances consist of instances 1–11 published in (Gendreau et al., 2006). For our new instances, we keep the same parameter domains and we use as training instances the set {SD-CSS2, SD-CSS3, SD-CSS4, SD-CSS7, SD-CSS12}, which are the fastest to be solved.

The outcome of the I/F-Race procedure is that the best configurations are the following: $T_0 = 12746.634$, $\delta = 28990$ so that ($T_{min} = 439.680$, and $\rho = 210$ for the 3L-CVRP benchmarks, and $T_0 = 297143$, $\delta = 88298$ (resulting in $T_{min} = 3.36$), and $\rho = 810$ for our instances. The results for this configuration are presented in Table 2 and Table 5 in comparison with previous work.

5.3. New instances

Table 1 shows the features of each instance in terms of number of customers (n), box types (bt), total number of boxes (M), vehicle types (vt) and total number of vehicles (v). The terms φ and θ indicate in percentage the ratio between the total volume of boxes and the total volume of available vehicles and the ratio between the average customer demand (in m^3) and the average volume of vehicles, respectively.

The instances exhibit a high variability in terms of the number of customers, the number of box types and the number of boxes. In particular, it is worth noticing that the number of boxes in seven of the thirteen instances is greater than one thousand. The terms φ and θ give an intuition of how difficult is the packing subproblem and of the number of nodes per route, respectively.

The *Gini index* (Gini, 1921) is a measure of the heterogeneity of boxes from the relative frequencies associated with a box type. A value of 0 expresses that all the boxes are equal, whereas a value of 100 % that all items are different. We highlight this aspect because different levels of box heterogeneity need different packing strategies in order to perform the loading effectively.

5.4. Experimental results

We applied our solver to different problem formulations: the 3L-CVRP formulation, described in Section 2.1, the complete problem formulation, described in Section 2.2, and the complete problem formulation without split

Table 1: Features of the new real-world instances

Instance	n	bt	M	vt	v	φ	θ	Gini index
SD-CSS1	11	36	254	1	5	45.17	18.82	98.23%
SD-CSS2	25	15	350	1	13	59.57	29.78	88.29%
SD-CSS3	33	9	285	1	26	41.55	31.77	76.86%
SD-CSS4	37	13	312	1	12	56.58	17.86	89.49%
SD-CSS5	41	47	7035	2	13	5.10	1.57	95.39%
SD-CSS6	43	97	8060	1	35	28.53	22.70	93.21%
SD-CSS7	45	14	284	2	10	62.57	13.60	88.44%
SD-CSS8	48	70	3275	3	36	37.12	27.27	97.19%
SD-CSS9	56	45	1725	1	23	48.10	19.41	97.44%
SD-CSS10	60	29	1840	1	20	23.30	7.64	88.26%
SD-CSS11	92	34	3790	2	13	43.79	6.12	88.67%
SD-CSS12	129	10	745	1	50	53.67	20.64	97.34%
SD-CSS13	129	63	2880	1	35	34.21	9.21	96.53%

deliveries. The results are shown in Table 2. Average running times vary between 300 and 10000 seconds (which is set as timeout) depending on the average number of boxes for vehicle.

For some instances, it has been impossible to find a feasible solution, without split delivery. For such cases, we report the level of violation of the H2 constraint (the H1 constraint is never violated). In order to give a measure of the quality of a solution that is real-valued, the cost z is expressed in total km traveled and the value of H2 violations in m^3 of boxes that were not possible to load in the vehicles.

Comparing the 3L-CVRP formulation and the complete one without split deliveries, it is clear that the first one is able to obtain solutions with lower cost. Thus, we can conclude that the new packing constraints imposes in the complete formulation, which are related to the notions of fragility, stability and reachability have a no negligible impact on the solution process.

For the remainder of this section we focus on the results of the complete formulation. Looking at the results for the complete formulation with split deliveries, we notice that for instance SD-CSS3 this formulation is the only one able to obtain a feasible solution. A possible explanation is that this instance has the highest θ value, which is the ratio between the average customer demand and the average volume of vehicles, and the solution of

Table 2: Results on our instances with different constraints (H2 in m^3 and z in km).

	3L-CVRP formulation		No Split Delivery		Split Delivery	
Instance	H2	z	H2	z	H2	z
SD-CSS1	0	570.86	0	584.84	0	642.09
SD-CSS2	0	1203.32	0	1209.05	0	1316.78
SD-CSS3	51.22	–	51.22	–	0	1664.50
SD-CSS4	0	1139.86	0	1182.10	0	1260.75
SD-CSS5	0	1183.67	1.09	–	0	2618.52
SD-CSS6	0	1993.98	3.91	–	0	3667.62
SD-CSS7	0	1180.90	0	1254.88	0	1565.44
SD-CSS8	0	2318.31	15.96	–	0	4957.96
SD-CSS9	0	1772.48	0	1889.00	0	3260.29
SD-CSS10	0	1294.59	0	1785.94	0	2792.31
SD-CSS11	0	2690.05	0	3307.03	0	4379.04
SD-CSS12	0	3480.73	0	3517.69	0	3643.32
SD-CSS13	0	2806.02	0	3589.73	0	6627.97

this case might require to split the demand of customers to different vehicles.

For most instances, the use of the larger space with split deliveries allows us to obtain a feasible solution (SD-CSS3, SD-CSS5, SD-CSS6, SD-CSS8). On the other hand, when the instance is large in terms of the number of boxes and vehicles, exploring a larger space is not effective, resulting in solutions of worse quality. Our experimental results also suggest to use an adaptive strategy in the complete formulation that switches between the possibility of using or not split delivery. One may start by forbidding split deliveries. If feasible solutions without split deliveries are found, these are typically of a better quality w.r.t. the total distance traveled. If feasible solutions are not obtained relatively quickly, for example, after the first round through the simulated annealing and intensifier phases, we may switch to the formulation considering split deliveries to increase the chance to find feasible solutions.

5.5. Analysis of the routing and packing components

In the previous section, we have shown the performance of our solver on different formulations of the integrated routing and packing problem; we now examine the importance of the routing component and the packing component taken in isolation. In order to do that, the new instances have been

converted into CLP and VRP instances, by removing data and constraints related to customers and to boxes, respectively.

In detail, for the packing component we solve the multiple container loading problem (MCLP) as defined by Ivancic et al. (1989), where the goal is to load all the boxes into a minimum number of containers, and only constraints about the rotations (C1) and the supporting area (C3) are considered. We thus remove the association between boxes and clients, such that boxes can be loaded in any order without respecting the LIFO constraint (C4), and we do not take into account the possible route lengths that would be implied from the particular packing. Consequently, the solver initially generates a solution by creating one single block for each box type and randomly assigning it to a vehicle. The search space is then explored using only the MRB neighborhood, given that we fix the packing strategy to the Wall Building approach, which is the most effective for CLPs, and the neighborhoods that involve customers have no meaning.

In Table 3, we report the results obtained by our solver on the corresponding MCLP, in terms of average volume utilization (φ') of the vehicles used, whose number is reported as v'. Comparing these results with those on the 3L-CVRP formulation, we see that often the volume utilization increases strongly; in one exceptional case, on instance SD-CSS5, by almost five times. Recall that the routing component has influence on the packing especially through the LIFO constraint that ensures that all boxes of a customer can be unloaded without moving any of the boxes of other customers. Hence, dropping these particular constraints does allow in many cases for a much denser packing. As can be seen, the MCLP formulation runs, in some cases, orders of magnitude faster than the algorithm tackling the full version (remember that the algorithm is stopped if four rounds or one idle round has been done). Note also that running times are quite different from instance to instance and they are related mainly to the number of box types.

For the VRP, we remove the packing component by considering only the constraints related to the total volume of the boxes and their total weight. In other words, the instance is treated as a standard CVRP instance where in addition to the usual capacity constraint due to the weight, we consider also a capacity constraint with respect to the total volume. In this case, the solver uses only the SC and MC neighborhoods.

Table 4 presents the performance of the solver on the vehicle routing problem in isolation. Also in this case, the average saving, here in terms of distance traveled, is substantial; it is equal to 43.17% with a reduction on

Table 3: Performance of the algorithm solving the container loading problem in isolation

	3L-CVRP formulation			CLP formulation			$\Delta_{\varphi'}$
Instance	v'	φ'	sec	v'	φ'	sec	
SD-CSS1	5	0.45	356.57	4	0.56	16.10	25.00%
SD-CSS2	13	0.60	5033.87	13	0.60	0.33	0.00%
SD-CSS3	23	–	–	22	0.49	0.79	–
SD-CSS4	12	0.57	5184.83	11	0.62	3.98	9.09%
SD-CSS5	12	0.06	10000.00	2	0.33	228.53	497.24%
SD-CSS6	32	0.29	5043.56	19	0.53	2749.62	84.21%
SD-CSS7	10	0.63	6125.72	10	0.63	1.10	0.00%
SD-CSS8	36	0.37	4401.13	21	0.64	1769.38	71.68%
SD-CSS9	23	0.48	1734.35	17	0.65	1079.92	35.29%
SD-CSS10	18	0.26	7978.60	7	0.67	1023.35	157.17%
SD-CSS11	13	0.44	1553.93	9	0.63	513.38	44.03%
SD-CSS12	48	0.56	1763.91	37	0.73	17.58	29.74%
SD-CSS13	31	0.39	4664.27	19	0.63	1328.95	63.17%
	276	0.42	4486.73	191	0.59	671.77	84.72%

the number of vehicles of more than 50%. This shows that the constraints on capacity and total volume alone are quite loose, so that the VRP constraints alone would make it possible to deliver the goods to customers by less vehicles, leading also to a strong reduction in overall route length.

Thus, we can deduce that the constraint about the weight capacity is not tight, whereas the dimensions of each item and its volume occupation have a strong impact and greatly reduce the possibility of grouping together many customers in the same route.

5.6. Comparison with related work

Our solver has been tested also on the 27 instances proposed by Gendreau et al. (2006), which have been derived from CVRP instances (Toth & Vigo, 2002) by specifying the customer demand in the form of rectangular three-dimensional items. A detailed description of the instances' features can be found in the work by Gendreau et al. (2006). For all benchmarks, the supporting area factor is set to 0.75, as in previous works.

For the 3L-CVRP, the MRB neighborhood is restricted to exclude moves that lead to split deliveries. This is done by allowing to reinsert an item

Table 4: Performance of the algorithm solving the vehicle routing problem in isolation

	3L-CVRP formulation			VRP formulation			Δ_z
Instance	v'	z	sec	v'	z	sec	
SD-CSS1	5	570.86	356.57	3	366.94	14.66	-35.72%
SD-CSS2	13	1203.32	5033.87	8	775.24	20.43	-35.57%
SD-CSS3	23	–	–	12	960.07	26.83	–
SD-CSS4	12	1139.86	5184.83	7	784.52	25.06	-31.17%
SD-CSS5	12	1183.67	10000.00	2	511.77	58.29	-56.76%
SD-CSS6	32	1993.98	5043.56	11	1065.94	59.88	-46.54%
SD-CSS7	10	1180.90	6125.72	7	737.08	61.70	-37.58%
SD-CSS8	36	2318.31	4401.13	15	1273.60	68.33	-45.06%
SD-CSS9	23	1772.48	1734.35	12	1070.31	79.90	-39.62%
SD-CSS10	18	1294.59	7978.60	5	736.98	80.44	-43.07%
SD-CSS11	13	2690.05	1553.93	6	875.77	134.25	-67.44%
SD-CSS12	48	3480.73	1763.91	28	2530.35	187.59	-27.30%
SD-CSS13	31	2806.02	4664.27	13	1340.82	171.94	-52.22%
	276	1802.90	11986.73	130	1002.26	76.10	-43.17%

only in a position where it is adjacent to another one belonging to the same customer. Thus, the MRB move can only modify the reciprocal position of items of the same customer and change their orientation.

Table 5 summarizes the features of the instances and shows the computational results. The best result is shown in bold face. For instances by Gendreau et al. (2006), in Table 5 we compare our solver with the reported results of the TS by Gendreau et al. (2006), the guided TS by Tarantilis et al. (2009), the ACO by Fuellerer et al. (2010) and the Hybrid TS by Bortfeldt (2012). We do not report the results by Wang et al. (2010) because they use a larger number of vehicles, thus they are not comparable.

Since TS is deterministic, it was invoked only once for each instance and this value is reported. For the guided TS we have only the cost of the best solutions available. ACO, Hybrid TS and our SA are invoked 10 times with different random seeds on each run, and the average and the minimum cost are reported.

The outcome is that our solver improves on the original results by Gendreau et al. (2006) and those of Tarantilis et al. (2009). It performs worse,

Table 5: Computational results for 3L-CVRP on Gendreau et al. (2006) instances (time is in seconds).

I	n	M	v	Gendreau et al.		Tarantilis et al.		Fuellerer et al.			Bortfeldt			Us		
				z	sec	z_{min}	sec	z_{avg}	z_{min}	sec	z_{avg}	z_{min}	sec	z_{avg}	z_{min}	sec
1	15	32	4	316.32	129.5	321.47	13.2	305.35	304.13	11.2	**302.02**	302.02	72.3	305.1	302.03	100.62
2	15	26	5	350.58	5.3	334.96	11.5	**334.96**	334.96	0.1	**334.96**	334.96	0.9	334.97	334.97	55.19
3	20	37	4	447.73	461.1	430.95	540.6	409.79	399.68	88.5	**401.44**	392.63	182	439.35	414.51	213.82
4	20	36	6	448.48	181.1	458.04	323.5	440.68	440.68	3.9	**437.54**	437.19	16.1	447.43	440.69	139.39
5	21	45	6	464.24	75.8	465.79	99.6	453.19	450.93	22.7	**451.03**	443.61	182.6	460.13	450.16	179.59
6	21	40	6	504.46	1167.9	507.96	1212.4	501.47	498.32	17.5	**498.38**	498.16	23.6	507.24	498.33	165.36
7	22	46	6	831.66	181.1	796.61	364.8	797.47	792.13	51.4	772.49	769.68	133.1	**755.79**	752.75	225.08
8	22	43	8	871.77	156.1	880.93	230	**820.67**	820.67	56.2	821.35	810.89	139.1	822.04	820.22	193.22
9	25	50	8	666.1	1468.5	642.22	982.2	**635.5**	635.5	15.3	645.81	630.13	24.3	654.79	630.14	216.01
10	29	62	8	911.16	714	884.74	1308.4	841.12	840.75	241.2	**827.29**	820.35	175.1	856.79	841.43	427.44
11	29	58	8	819.36	396.4	873.43	522.5	821.04	818.87	172.4	**815.62**	803.61	136.4	828.71	805.93	383.93
12	30	63	9	651.58	268.1	624.24	294.6	**629.07**	626.37	46.2	630.46	614.59	14	637.89	621.58	393.54
13	32	61	8	2928.34	1639.1	2799.74	2193.1	2739.8	2739.8	235.4	**2694.81**	2645.95	268.4	2747.96	2718.34	460.28
14	32	72	9	1559.64	3451.6	1504.44	4581.3	1472.26	1466.84	623.8	**1413.59**	1368.42	311.6	1494.01	1436.51	574.77
15	32	68	9	1452.34	2327.4	1415.42	2528.3	1405.48	1367.58	621	**1355.5**	1341.14	311.5	1410.01	1369.48	556.86
16	35	63	11	707.85	2550.3	698.61	4256.5	**698.92**	698.92	12.8	705.05	698.61	3.4	702.34	698.63	442.43
17	40	79	14	920.87	2142.5	872.79	2096	**870.33**	868.59	11.8	917.96	866.4	2.5	882.65	866.42	549.87
18	44	94	11	1400.52	1452.9	1296.59	2275.2	1261.07	1255.64	2122.2	**1228.98**	1207.72	309.5	1272.62	1238.71	1103.97
19	50	99	12	871.29	1822.3	818.68	2509	781.29	777.18	614.3	**753.87**	741.74	416.5	817.77	786.13	1239.86
20	71	147	18	732.12	790	641.57	1940.9	611.26	604.28	3762.3	**596.42**	587.95	427	625.15	612.42	2977.67
21	75	155	17	1275.2	2370.3	1159.72	2823.4	1124.55	1110.09	5140	**1107**	1090.22	443.4	1160.77	1133.68	3143.04
22	75	146	18	1277.94	1611.3	1245.35	2685.6	1197.43	1194.18	2233.6	**1171.49**	1147.8	423.5	1220.57	1188.51	2752.97
23	75	150	17	1258.16	6725.6	1231.92	4659.1	1171.77	1158.51	3693.4	**1135.46**	1130.54	425.8	1191.7	1149.33	3030.6
24	75	143	16	1307.09	6619.3	1201.96	4854.1	1148.7	1136.8	1762.8	**1128.82**	1116.13	411.1	1194.01	1162.37	2934.6
25	100	193	22	1570.72	5630.9	1457.96	5725.8	1436.32	1429.64	8619.7	**1428.8**	1407.36	453	1477.09	1455.3	5978.76
26	100	199	26	1847.95	4123.7	1711.93	6283.1	**1616.99**	1611.78	6651.2	1625.31	1600.35	430.6	1662.95	1627	5720.78
27	100	198	23	1747.52	7127.2	1646.44	9915.7	1573.5	1560.7	10325.8	**1550.85**	1529.86	435	1628.01	1587.04	5110.95
Avg				1042.26	2058.86	997.20	2415.94	966.67	960.87	1746.54	**953.79**	938.44	228.60	982.88	960.84	1454.47

however, than the ACO (Fuellerer et al., 2010) and the Hybrid TS (Bortfeldt, 2012) approaches although in one case (instance 7) is able to find the best known result. However, all the techniques reported in Table 5 are specifically designed and tuned for the case of the 3L-CVRP and for these instances, whereas our technique solves the more general problem that deals with split deliveries, weakly heterogeneous cargos, and stability constraints as described below.

6. Conclusion and Future Work

In this paper we have addressed a complex problem, that combines routing and packing issues, and considers several features that arise in real-world situations. This problem extends and enriches the *Three-Dimensional Loading Capacitated Vehicle Routing Problem* (Gendreau et al., 2006) by redefining some constraints with respect to the notion of stability, loading and unloading policy and the heterogeneity of the cargo. We also considered the possibility of managing a heterogeneous fleet and to split the demand of a customer in more vehicles.

We have presented a local search approach based on simulated annealing and large-neighborhood search that solves the integrated problem in one single stage. In fact, a solution is represented as sequences of items making suitable the use of neighborhood operators that modify both routes and arrangements in vehicles.

The solution approach has been tested on 13 new real-world instances, that exhibit great diversity in size and features and the effect of introducing split deliveries has been analyzed and discussed. All instances are available at `http://www..uniud.it/ceschia/index.php?page=vrclp`.

In addition, the solver has been tested on benchmark instances of (Gendreau et al., 2006) for the 3L-CVRP. Although our solver is not designed for that problem, it reaches better or equal solutions than two algorithms specifically designed for the 3L-CVRP; but it is inferior to two other recent metaheuristics algorithms for the 3L-CRVP.

For the future work, we want to design a more sophisticated packing heuristic able to effectively pack large number of items while satisfying complex constraints that come up in practice. In addition, we plan to study the relationship between the features of the instances and the solution approach in order to automatically apply the most suitable one.

The long-term goal of the project is the design of an application that could be used in practice by potential clients, either interactively or in batch mode. The solver should be flexible enough to be able to adapt automatically to different problem formulations. In particular, it shall be able to select constraints and cost components at run-time based on the analysis of the specific instance. The proposed solver will be the algorithmic core of the application, which needs to be complemented by a graphical front-end and a persistency managing tool.

Acknowledgements

We thank our industrial partner beanTech for providing us the real-world data and for bringing to our attention this interesting problem. Sara Ceschia and Andrea Schaerf acknowledge support from Google Inc. under the Google Focused Grant Program on "Mathematical Optimization and Combinatorial Optimization in Europe". Thomas Stützle acknowledges support from the Belgian F.R.S.-FNRS, of which he is a Research Associate.

Archetti, C., Feillet, D., Gendreau, M., & Speranza, M. G. (2011). Complexity of the VRP and SDVRP. *Transportation Research Part C: Emerging Technologies*, *19*, 741–750.

Archetti, C., Savelsbergh, M. W. P., & Speranza, M. G. (2006a). Worst-case analysis for split delivery vehicle routing problems. *Transportation Science*, *40*, 226–234.

Archetti, C., Savelsbergh, M. W. P., & Speranza, M. G. (2008a). To split or not to split: That is the question. *Transportation Research Part E: Logistics and Transportation Review*, *44*, 114–123.

Archetti, C., Speranza, M. G., & Hertz, A. (2006b). A tabu search algorithm for the split delivery vehicle routing problem. *Transportation Science*, *40*, 64–73.

Archetti, C., Speranza, M. G., & Savelsbergh, M. W. P. (2008b). An optimization-based heuristic for the split delivery vehicle routing problem. *Transportation Science*, *42*, 22–31.

Baker, B. S., Coffman, Jr., E. G., & Rivest, R. L. (1980). Orthogonal packings in two dimensions. *SIAM Journal on Computing*, *9*, 846.

Birattari, M., Yuan, Z., Balaprakash, P., & Stützle, T. (2010). F-race and iterated F-race: An overview. In T. Bartz-Beielstein, M. Chiarandini, L. Paquete, & M. Preuss (Eds.), *Experimental Methods for the Analysis of Optimization Algorithms* (pp. 311–336). Berlin: Springer.

Bischoff, E. E. (2006). Three-dimensional packing of items with limited load bearing strength. *Journal of Operational Research*, *168*, 952–966.

Bischoff, E. E., & Ratcliff, M. S. W. (1995). Issues in the development of approaches to container loading. *Omega*, *23*, 377–390.

Bortfeldt, A. (2012). A hybrid algorithm for the capacitated vehicle routing problem with three-dimensional loading constraints. *Computers and Operations Research*, *39*, 2248–2257.

Bortfeldt, A., & Gehring, H. (2001). A hybrid genetic algorithm for the container loading problem. *European Journal of Operational Research*, *131*, 143–161.

Ceschia, S., Di Gaspero, L., & Schaerf, A. (2011). Tabu search techniques for the heterogeneous vehicle routing problem with time windows and carrier-dependent costs. *Journal of Scheduling*, *14*, 601–615.

Ceschia, S., & Schaerf, A. (2013). Local search for a multi-drop multi-container loading problem. *Journal of Heuristics*, *19*, 275–294.

Christofides, N., & Whitlock, C. (1977). An algorithm for two-dimensional cutting problems. *Operations Research*, *25*, 30–44.

Crainic, T. G., Perboli, G., & Tadei, R. (2008). Extreme point-based heuristics for three-dimensional bin packing. *INFORMS Journal on Computing*, *20*, 368–384.

Davies, A. (2000). *Approaches to the container loading problem*. Ph.D. thesis University of Wales Swansea.

Davies, A. P., & Bischoff, E. E. (1999). Weight distribution considerations in container loading. *European Journal of Operational Research*, *114*, 509–527.

de Castro Silva, J. L., Soma, N. Y., & Maculan, N. (2003). A greedy search for the three-dimensional bin packing problem: the packing static stability case. *International Transactions in Operational Research*, *10*, 141–153.

Di Gaspero, L., & Schaerf, A. (2003). EasyLocal++: An object-oriented framework for flexible design of local search algorithms. *Software-Practice and Experience*, *33*, 733–765.

Doerner, K. F., Fuellerer, G., Hartl, R. F., Gronalt, M., & Iori, M. (2007). Metaheuristics for the vehicle routing problem with loading constraints. *Networks*, *49*, 294–307.

Dror, M., Laporte, G., & Trudeau, P. (1994). Vehicle routing with split deliveries. *Discrete Applied Mathematics*, *50*, 239 – 254.

Dror, M., & Trudeau, P. (1989). Savings by split delivery routing. *Transportation Science*, *23*, 141.

Eley, M. (2002). Solving container loading problems by block arrangement. *European Journal of Operational Research*, *141*, 393–409.

Fanslau, T., & Bortfeldt, A. (2010). A tree search algorithm for solving the container loading problem. *INFORMS Journal of Computing*, *22*, 222–235.

Fuellerer, G., Doerner, K. F., Hartl, R. F., & Iori, M. (2010). Metaheuristics for vehicle routing problems with three-dimensional loading constraints. *European Journal of Operational Research*, *201*, 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.

George, J. A., & Robinson, D. F. (1980). A heuristic for packing boxes into a container. *Computer & Operations Research*, *7*, 147–156.

Gini, C. (1921). Measurement of inequality of incomes. *The Economic Journal*, *31*, pp. 124–126.

Hoos, H. H. (1998). *Stochastic local search-methods, models, applications*. Ph.D. thesis Darmstadt University of Technology / Germany.

Hoos, H. H., & Stützle, T. (2005). *Stochastic Local Search — Foundations and Applications*. San Francisco, CA (USA): Morgan Kaufmann Publishers.

Iori, M., & Martello, S. (2010). Routing problems with loading constraints. *Top*, *18*, 4–27.

Ivancic, N., Mathur, K., & Mohanty, B. (1989). An integer programming based heuristic approach to the three-dimensional packing problem. *Journal of Manufacturing and Operations Management*, *2*, 268–298.

Junqueira, L., Morabito, R., & Sato Yamashita, D. (2011). MIP-based approaches for the container loading problem with multi-drop constraints. *Annals of Operations Research*, *199*, 51–75.

Junqueira, L., Morabito, R., & Sato Yamashita, D. (2012). Three-dimensional container loading models with cargo stability and load bearing constraints. *Computers & Operations Research*, *39*, 74–85.

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.

López-Ibáñez, M., Dubois-Lacoste, J., Stützle, T., & Birattari, M. (2011). *The `irace` Package, Iterated Race for Automatic Algorithm Configuration*. Technical Report TR/IRIDIA/2011-004 IRIDIA, Université Libre de Bruxelles Belgium.

Martello, S., Pisinger, D., & Vigo, D. (2000). The three-dimensional bin packing problem. *Operations Research*, *48*, 256–267.

Martello, S., Pisinger, D., Vigo, D., Boef, E. D., & Korst, J. (2007). Algorithm 864. *ACM Transactions on Mathematical Software*, *33*.

Moura, A., & Oliveira, J. F. (2008). An integrated approach to the vehicle routing and container loading problems. *OR Spectrum*, *31*, 775–800.

Ratcliff, M., & Bischoff, E. (1998). Allowing for weight considerations in container loading. *OR Spectrum*, *20*, 65–71.

Reimann, M., Doerner, K. F., & Hartl, R. F. (2004). D-ants: Savings based ants divide and conquer the vehicle routing problem. *Computers & Operations Research*, *31*, 563–591.

Tarantilis, C., Zachariadis, E. E., & Kiranoudis, C. T. (2009). A hybrid metaheuristic algorothm for the integrated vehicle routing and three-dimensional container-loading problem. *IEEE Transactions on Intelligent Transportation Systems*, *10*, 1524–9050.

Toth, P., & Vigo, D. (Eds.) (2002). *The Vehicle Routing Problem*. Monographs on Discrete Mathematics and Applications. Philadelpia, PA (USA): SIAM.

Tricoire, F., Doerner, K. F., Hartl, R. F., & Iori, M. (2009). Heuristic and exact algorithms for the multi-pile vehicle routing problem. *OR Spectrum*, *33*, 931–959.

Wang, L., Guo, S., Chen, S., Zhu, W., & Lim, A. (2010). Two natural heuristics for 3d packing with practical loading constraints. *PRICAI 2010: Trends in Artificial*, *6230*, 256–267.

Zachariadis, E. E., Tarantilis, C. D., & Kiranoudis, C. T. (2012). The pallet-packing vehicle routing problem. *Transportation Science*, *46*, 341–358.