Expert Systems With Applications 137 (2019) 471–492

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

A hybrid approach for a multi-compartment container loading problem

Rodolfo Ranck Júnior[a], Horacio Hideki Yanasse[b], Reinaldo Morabito[c], Leonardo Junqueira[d,*]

[a] Programa de Pós-Graduação em Computação Aplicada, Instituto Nacional de Pesquisas Espaciais, Avenida dos Astronautas, 1758, 12227-010, Jardim da Granja, São José dos Campos, SP, Brazil
[b] Instituto de Ciência e Tecnologia, Universidade Federal de São Paulo, Avenida Cesare Mansueto Giulio Lattes, 1201, 12247-014, Eugênio de Melo, São José dos Campos, SP, Brazil
[c] Departamento de Engenharia de Produção, Universidade Federal de São Carlos, Rodovia Washington Luís, km 235 - SP-310, São Carlos, SP 13565-905, Brazil
[d] Departamento de Engenharia de Produção, Escola Politécnica, Universidade de São Paulo, Avenida Prof. Luciano Gualberto, 1380, São Paulo, Butantã, SP 05508-010, Brazil

ARTICLE INFO

Article history:
Received 16 May 2018
Revised 8 July 2019
Accepted 9 July 2019
Available online 10 July 2019

Keywords:
Container Loading Problems
Hybrid Approach
Multiple Compartments
Beverage Distribution

ABSTRACT

In this paper, we address a real problem of packing boxes into a multi-compartment container, in which the boxes must be delivered to customers in a predefined route. This problem arises, for example, in the distribution of beverages (packed in "boxes") by trucks whose container has multiple compartments. The objective is to find a feasible packing plan that minimizes the handling of boxes inside the container along the whole truck route. The following practical constraints must be met: orientation of the boxes, cargo stability, load bearing strength of the boxes and load balancing. To the best of our knowledge, this is the first study focusing on a vehicle packing problem with multi-compartment containers in the context of beverage distribution. To solve this problem, we present a hybrid approach which consists of a heuristic method based on the generation of horizontal layers and on the solution of mixed integer linear programming models. Computational tests were performed with this approach and a large variety of instances based on real data from a soft drink company. The results show that the approach is able to find feasible solutions for all instances considered under all constraints, contrary to what was observed in practice with the manual procedures available in the company. In practice, if a feasible solution is not obtained, it is necessary to change the predefined route plan and/or to consider an additional new delivery route and/or to use a solution that violates some of the constraints involved. In this sense, the proposed solution approach has good potential for being embedded into existing expert and intelligent systems for supporting the decision making process.

1. Introduction

Freight transportation is an important activity in several sectors, especially for companies that need to collect different types of products from their suppliers and/or that need to deliver different types of products to their customers (e.g., *milk run*). Together with the activities of warehousing and handling, freight transportation permeates the costliest sectors in logistics, and therefore it must be performed efficiently, with the products being collected/delivered on time, within the desired conditions and with reduced costs. The costs of freight transportation can be related to the delivery routes (e.g., Gendreau, Iori, Laporte & Martello, 2006), to the production planning and lot sizing (e.g., Norden & Velde, 2005), and to the container loading (e.g., Chen, Lee & Shen, 1995; Terno, Scheithauer, Sommerweiss & Riehme, 2000).

In this paper, we investigate a real Container Loading Problem arising in the beverage industry. This problem was motivated by a case study carried out in a soft drink company in Brazil, but similar loading problems are also found in other beverage companies. This company is a large producer that maintains a high degree of international brand recognition, and it bottles very well-known soft drinks in different countries. In the present problem, the company needs to deliver its products daily to a set of customers, and it has available a given type of vehicle/truck whose container has multiple compartments (some of them of different sizes). This type of container is useful, e.g., when one aims at increasing the cargo stability or separation. The truck must then perform a delivery route known in advance (determined by the vehicle routing system

* Corresponding author.
E-mail addresses: junqueira@usp.br, leo_junqueira@yahoo.com (L. Junqueira).

https://doi.org/10.1016/j.eswa.2019.07.020
0957-4174/

of the company), and all the boxes required by the customers in the route must be packed inside the container. When packing the cargo, several practical constraints must be met, such as the orientation of the boxes, the cargo stability, the load bearing strength of the boxes and the load balancing. The main objective is to find a feasible packing plan with all the boxes required by the customers in the predefined route, and, if this solution is found, then to minimize the handling of boxes inside the container along the truck route.

It is important to note that the aforementioned practical constraints must be met along the whole truck route, and not only at its departure from the depot (see, e.g., Junqueira, Oliveira, Carravilla & Morabito, 2013). Since this problem involves "loss of mass" along the delivery route, it must be ensured that the boxes required by a customer, once dropped-off, neither cause cargo instability to the boxes required by the remaining customers nor cause the center of gravity of the remaining cargo to lay on an unsafe position.

This problem of loading boxes in a multi-compartment vehicle differs from most Container Loading Problems treated in the literature. It can also be seen as a problem in which multiple containers or pallets are available for packing the boxes (see, e.g., Chan, Bhagwat, Kumar, Tiwari & Lam, 2006; Chen et al., 1995; Eley, 2003; Alonso, Alvarez-Valdes, Iori & Parreño, 2019). However, it considers several objects (compartments) in a same vehicle respecting, for example, the load balancing and/or multiple destinations of the boxes. It is also related to the Master Bay Plan Problem, which consists in determining stowage plans for containers in a ship (see, e.g., Pacino & Jensen, 2012; Sciomachen & Tanfani, 2003, 2007; Araújo, Chaves, Salles Neto & Azevedo, 2016). In that problem, the containers, the ship and its bays can be viewed as the items, the object and its compartments of the present problem, respectively. However, these two problems also have major differences, as discussed in Section 2.

According to Wäscher, Haussner and Schumann (2007), this problem can be seen as an extended three-dimensional packing problem, once it extends the concept of packing/cutting items in objects, for example, due to the minimization of the handling of items inside objects along a route. However, the associated problem of how to pack boxes (beverage packaging) inside objects (compartments of the vehicle) does not fit perfectly the typology proposed by the authors, once it does not address the maximization of the outputs (selection of items to be packed in the objects) or the minimization of the inputs (selection of objects to pack the items), but all the items and objects must be used for the packing, while satisfying the related constraints.

In the day to day of the beverage company, it is hard to obtain in practice feasible solutions for this problem with the manual procedures that the company has available, which usually entail a lot of handling of boxes inside the container due to difficulties with the arrangement of all boxes inside the compartments along the vehicle route, besides respecting other problem constraints. If a feasible solution is not obtained, it is necessary to change the predefined route plan and/or to consider an additional new delivery route and/or to use a solution that violates some of the constraints involved. In this sense, the development of solution approaches for being embedded into existing expert and intelligent systems may support the decision making process.

We present in this paper a hybrid solution approach that consists of a heuristic method based on the generation of horizontal layers and on the solution of mixed integer linear programming models. The used formulations contemplate several practical packing considerations, as the aforementioned ones. The proposed solution approach is novel in combining exact and heuristic methods to tackle a real container loading variant for the first time, while at the same time inspired by the way that the beverage company that motivated this study packs the boxes inside the container, and we believe that it can be directly used, or straightforwardly modified to be used, in other related distribution settings.

To the best of our knowledge, this is the first study focusing on a vehicle packing problem with multi-compartment containers in the context of beverage distribution, and it contributes to the field by describing in detail what the practical problem is so that future researchers may use this problem instead of a stylized one to motivate their research. The paper also provides practical guidelines in the development and management of expert and intelligent systems for loading (unloading) products into (from) a multi-compartment container along the route of its truck. These guidelines are based on the design, development and testing of computational procedures based on operations research techniques and inspired on a real-life case study of the problem.

This work is organized as follows. In Section 2, the related literature is briefly reviewed. In Section 3, the problem addressed in this paper is properly defined. In Section 4, the hybrid solution approach to the problem is presented. In Section 5, the results of some computational tests with the proposed solution approach and a large variety of instances based on real data provided by a beverage company are analyzed and discussed. Finally, in Section 6, the concluding remarks and some perspectives for future research are presented.

2. Related literature

Packing smaller items into larger objects is a frequently encountered problem when dealing with cargo transportation, and it is normally a complex task when efficiency is longed-for. The arrangement of items inside an object (i.e., a packing pattern) must meet at least two basic constraints (the so-called geometrical constraints): the items are packed completely inside the object and the items do not overlap each other. The Container Loading Problem is a three-dimensional packing problem in which the boxes (items) need to be placed inside containers (objects) respecting one or more criteria. According to Bortfeldt and Wäscher (2013), with few exceptions, most studies consider rectangular items that can only be placed orthogonally into also rectangular containers.

Realistic Container Loading Problems normally take into consideration other practical aspects, such as the orientation of the boxes, the weight distribution within the container, the multiple destinations, the cargo stability and the load bearing strength of the boxes, among others (Bischoff & Ratcliff, 1995). Examples of studies that tackle Container Loading Problems with some of these practical considerations include, e.g., Bischoff (2006), Ceschia and Schaerf (2013) and Ramos, Oliveira, Gonçalves and Lopes (2016), Davies and Bischoff (1999), Junqueira, Morabito and Yamashita (2012a), Junqueira, Morabito and Yamashita (2012b), Silva, Soma and Maculan (2003). A compilation and discussion of studies that addressed several practical constraints can be found in Bortfeldt and Wäscher (2013).

Container Loading Problems generalize the well-known Knapsack Problem, and therefore they are also NP-Hard. In general, only small practical instances of these problems can be solved with exact algorithms (e.g., Hifi, 2004; Lai, Xue & Xu, 1998; Martello, Pisinger & Vigo, 2000). Due to their inherent complexity, most solution methods for Container Loading Problems are heuristics, which can be classified into constructive heuristics (e.g., Araújo & Armentano, 2007; Bischoff, Janetz & Ratcliff, 1995; George & Robinson, 1980), metaheuristics (e.g., Gehring & Bortfeldt, 1997; Gonçalves & Resende, 2012; Moura & Oliveira, 2005; Parreño, Alvarez-Valdes, Oliveira & Tamarit, 2010; Ramos, Silva & Oliveira, 2018) and tree-search heuristics (e.g., Araya & Riff, 2014; Eley, 2002; Fanslau & Bortfeldt, 2010; Pisinger, 2002). There is also a class of approximate algorithms that are able to ensure a given

worst-case performance, for example, in what concerns the objective function of the problem (e.g., Hifi, 2002; Miyazawa & Wakabayashi, 1999). A recent comparative review of algorithms for Container Loading Problems can be found in Zhao, Bennell, Bektaş and Dowsland (2016).

According to Pisinger (2002), these heuristics can also be classified into four fundamental types related to how the boxes are arranged inside the container:

(a) *Wall-building*. The boxes are arranged in horizontal (e.g., Bischoff et al., 1995; Terno et al., 2000) or vertical (e.g., Davies & Bischoff, 1999; George & Robinson, 1980) layers;
(b) *Stack-building*. Boxes are arranged in vertical columns (e.g., Gehring & Bortfeldt, 1997; Haessler & Talbot, 1990);
(c) *Guillotine Cutting*. Boxes are arranged in such a way that they could be entirely separated by means of a series of guillotine cuts (e.g., Morabito & Arenales, 1994);
(d) *Cuboid Arrangement*. Boxes of the same type and with the same orientation are arranged in blocks (e.g., Bortfeldt, Gehring & Mack, 2003; Eley, 2002).

The arrangement of boxes in layers is frequently employed when solving Container Loading Problems. This type of arrangement makes it easy to load the boxes, once it enables a less complex packing pattern. In the case where the layers are horizontal, they also favor the cargo stability (Bischoff & Ratcliff, 1995). When packing horizontal layers, in some cases it is possible to separate them with thin sheets (dividers) of wood or plastic in order to increase the cargo stability and/or separate the boxes.

In the Combinatorial Optimization literature, studies that address multi-compartment containers/vehicles are frequently related to Capacitated Vehicle Routing Problems, in which routes need to be defined respecting the capacities of the vehicle compartments (see, e.g., Henke, Speranza & Wäscher, 2015; Lahyani, Coelho, Khemakhem, Laporte & Semet, 2015). To the best of our knowledge, studies addressing these problems, although they consider capacitated compartments (e.g., with relation to the weight or volume), they do not contemplate geometrical constraints taking into account more than one spatial dimension, and therefore they are not considered Container Loading Problems (see, e.g., Avella, Boccia & Sforza, 2004; Derigs et al., 2011).

Another related problem, the Master Bay Plan Problem, which consists in determining stowage plans for containers in a ship, can also be seen as a Container Loading Problem (see, e.g., Pacino & Jensen, 2012; Sciomachen & Tanfani, 2003, 2007; Araújo et al., 2016). In this problem, the containers, the ship and its bays can be understood as items, object and compartments, respectively. The following characteristics, which are typical of this problem, differ from the ones addressed in this study: (a) the height of a packing pattern is not limited by physical walls, but only by load balancing and/or stability constraints; (b) the containers are standardized, they have the smaller sides with the same dimension and frequently the same height. Therefore, the variety of containers with relation to their spatial dimensions is low (typically there are only two or three container types); (c) due to the characteristics of ship and containers, the containers are arranged in independent stacks with fixed orientation. Therefore, the resulting packing patterns are guillotine along the height, and, due to the container sizes, they are frequently also guillotine along the other two dimensions; (d) the larger side of a container is relatively large and typically occupies between 50% and 100% of the smaller side of the bay; (e) given the metallic structure of the containers, there is no concern with damages of a container due to the pressure applied by other containers placed above it.

As mentioned before, the problem of loading boxes in a multi-compartment vehicle can also be seen as a Container Loading Problem in which multiple containers or pallets are available for packing the boxes. To the best of our knowledge, unlike the problem addressed in this study, these problems do not consider several objects in a same vehicle respecting, for example, the load balancing and/or multiple destinations of the boxes (see, e.g., Chen et al., 1995; Chan et al., 2006; Eley, 2003).

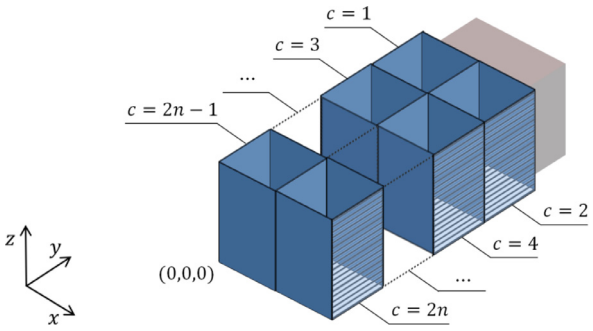


Fig. 1. Example of a multi-compartment container.

3. Problem definition

Consider a vehicle that has a set of compartments which are available to cargo packing. These compartments can be directly accessed through doors on the vehicle sides, and they are arranged in contiguous rows. The compartments have all the same length and width, but they can have different heights due to the axles and wheels of the truck. There are two rows of compartments along the smaller side of the vehicle, while along the larger side the number of rows can vary according to their types. Fig. 1 shows a scheme of a multi-compartment container. For the sake of simplicity, in this figure all the compartments have equal sizes.

All the compartments and boxes are rectangular hexahedrons (cuboids). The boxes are rigid, they have different dimensions and they must be orthogonally arranged inside the vehicle compartments without necessarily obeying a specific pattern and/or in horizontal layers. Each layer has already defined an optimal packing pattern for boxes of a same type (i.e., with the same dimensions, weight and load bearing strength). The layers are separated from each other (or from other unbound items) by rigid sheets that have the same length and width of the compartment.

The vehicle is loaded in a logistic cell (origin) and its route is predefined by a sequence of stops (destinations) that it must follow to serve the customers. This route is previously determined by a delivery routing system of the company (in the studied company, there may be hundreds of vehicles). The demands of all customers are known in advance. All the boxes required by the customers in the predefined route must be packed inside the vehicle and the following constraints must be met:

C1: *Boxes orientation*. The boxes have fixed vertical orientation;
C2: *Cargo stability*. The boxes must be stabilized along the whole route, either by other boxes or by the container/compartment floor and walls. A layer offers support to the boxes placed above it, provided that there is a minimum (predefined) number of boxes in this layer;
C3: *Load bearing strength of the boxes*. In any stacking, a box placed below other boxes must have enough resistance to support the pressure that is applied over its top face;
C4: *Load balancing*. The vehicle must be well-balanced along the whole route, with relation to both axes x and y, i.e., the real position of the center of gravity of the cargo must not be far away from the ideal position of the center of gravity of the vehicle. Unlike constraints C1-C3, which are hard constraints and must be always met, constraints C4 are soft, i.e., a not

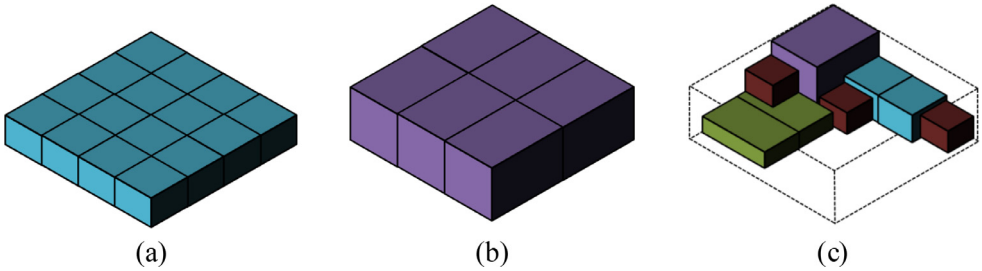


Fig. 2. Examples of horizontal layers.

Fig. 3. Example of packing pattern for one of the container compartments.

so well-balanced vehicle is not prevented from being used to serve the customers.

In this study, our objective is to minimize the handling of boxes inside the vehicle compartments along its route among all feasible solutions that pack all boxes into the vehicle. A box from stop k' (i.e., that must be unloaded at stop k') needs to be relocated in a stop k only if it is placed above boxes from stop k, with $k' > k$. As constraints C4 are soft, the load balancing deviations are also penalized in the objective function. According to Wäscher et al. (2007), this problem can be seen as an extended three-dimensional packing problem, once it extends the concept of packing/cutting items in objects, for example, due to the minimization of the handling of items inside objects along a route. However, the associated problem of how to pack boxes inside objects does not fit perfectly the typology proposed by the authors, once it does not address the maximization of the outputs (selection of items to be packed in the objects) or the minimization of the inputs (selection of objects to pack the items), but all the items and objects must be used for the packing, while satisfying the related constraints.

4. Proposed solution approach

The proposed solution approach is a hybrid method with a heuristic that aims at packing the boxes in horizontal layers, separated by physical dividers which are often used in the distribution of beverages to protect the loaded cargo and help provide vertical stability, with the help of simplified mathematical models of the problem. The following types of horizontal layers are considered: *complete*, which already have a predefined packing pattern, are made up of boxes of the same type and there are no stacked boxes; *incomplete*, which do not have a predefined packing pattern and are made up of boxes that are not in complete layers (*residual* boxes). The length and width of a complete or incomplete layer are the same of a compartment. The height of a complete layer is defined by the box type it contains, while the height of an incomplete layer is given by the highest position of one of the boxes that it contains. Fig. 2 shows examples of complete (patterns a and b) and incomplete (pattern c) layers.

Fig. 3 shows on the left an example of packing pattern for a container compartment with the layers shown in Fig. 2 (the yellow sheets represent the physical dividers), while on the right it shows a real packing pattern using horizontal layers for packing beverages in a compartment. Note that a complete layer always allows the placement of a divider above it, and therefore it can provide vertical stability for the boxes placed above it.

The heuristic presented here is inspired by the way that the beverage company that motivated this study packs the boxes inside the container, i.e., placing them in complete layers whenever possible. It assumes that:

A1: The relocation of boxes at a stop consists of removing and temporarily placing them in a zone outside the vehicle. Immediately after all boxes from this stop are unloaded from the vehicle, the relocated boxes are loaded back;

A2: The relocation of boxes is penalized by a function of the number and weight of the relocated boxes. Additionally, as it was observed in practice, minimizing the handling of boxes is considered more important than minimizing the load balancing deviations;

A3: The position of the boxes placed inside the vehicle (i.e., their front-bottom-left corners), defined with relation to the origin of a Cartesian coordinate system (see Fig. 1) before the vehicle performs its delivery route:

- cannot be changed along the route with relation to both axes x and y;
- can only be changed along the route with relation to axis z, in the case in which all boxes of a given layer, placed below, are unloaded from the vehicle. In this case, the boxes above this layer in the same compartment are all moved down until they are supported by another layer or by the compartment floor.

A4: Any box can always be reached, even though this implies in relocating other boxes;

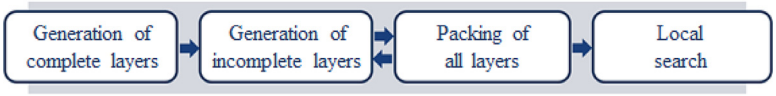


Fig. 4. Basic steps of the proposed heuristic.

A5: The position of the center of gravity of a layer is located at the geometric center of this layer;
A6: The container, the boxes and the dividers placed between layers are all rigid bodies;
A7: The weight above a layer is homogeneously distributed among the boxes on this layer;
A8: The load bearing strength of a box is homogeneously distributed over the area of its top face (i.e., each box supports the same admissible pressure in each point of its top face);
A9: The density of each box is homogeneous;
A10: The bottom faces of all boxes must be completely supported by other boxes, layers or the compartment floor (i.e., full support);
A11: For stability reasons, as it was observed in practice, incomplete layers cannot be placed below any complete layer in the same compartment;
A12: The walls of the compartments, as well as the dividers placed between layers, have negligible thicknesses;
A13: The boxes on a complete layer cannot be stacked.

In the following, the steps that compose the proposed heuristic are explained in detail. For reasons of organization, the definitions and notation used are presented in Appendix A, the pseudocodes are presented in Appendix B, and the mathematical formulations are presented in Appendix C.

The proposed heuristic can be divided into four basic steps, which are shown in Fig. 4.

4.1. Basic step 1

4.1.1. Generating the complete layers

Initially, given the available number of boxes of each type, we try to generate the largest possible number of complete layers. To this end, boxes of the same type are assigned to layers in inverse order of the vehicle stops, while they can be packed in these layers. Fig. 8 (see Appendix B) describes the procedure *gComp*() that is used for generating the complete layers. Thereby, the residual boxes, that can only be placed on the top of the compartments due to stability reasons (see assumption A11 at the beginning of Section 4), tend to be those ones required by the initial stops. Note that a complete layer already has a predefined packing pattern and, therefore, it is not necessary to define the position and orientation of the boxes while generating them.

4.2. Basic step 2

4.2.1. Generating the incomplete layers

The incomplete layers are generated using a constructive procedure respecting the practical constraints of the problem. Fig. 9 (see Appendix B) describes the procedure *gInc*() that is used to generate the incomplete layers. First, the residual boxes are sorted in a list in inverse order of the vehicle stops. As a first tiebreaker, we use the height of the boxes in decreasing order, and, if necessary, as a second tiebreaker, we use the load bearing strength of the boxes in decreasing order.

Following the defined order for the residual boxes, we try to sequentially pack them in a same incomplete layer. If after this attempt there are still not packed residual boxes, a new incomplete layer is generated and the process is repeated with this new layer and with the residual boxes still not packed. To try to define the position and the orientation of the residual boxes in an incomplete layer, the *First Fit* (FF) and *Best Fit* (BF) heuristics (described ahead) are used. Fig. 10 (see Appendix B) describes the procedure *pRes*() that is used to try to define the orientation and position of the residual boxes inside the incomplete layers.

The incomplete layers must have their heights limited to the heights of the compartments (that can be different). To this end, for every new incomplete layer that is generated, a new compartment is chosen as reference to limit the height of this layer. The compartments are chosen by their heights in increasing order, and, with this, we seek to allow that the first generated layers fit in a larger variety of compartments, thus increasing the chances of finding a feasible and efficient packing.

Two attempts of packing the residual boxes are performed. In one of the attempts the position and orientation of the residual boxes in the incomplete layers are defined using the FF heuristic, while in the other attempt the BF heuristic is used. If both attempts lead to a feasible solution (i.e., all residual boxes are packed in the incomplete layers), as a tiebreaker, we choose the solution whose sum of the heights of the incomplete layers is smaller.

4.2.2. First fit *and* best fit *heuristics for packing the residual boxes*

Initially, only position (0,0,0) of an incomplete layer is available and each residual box placed on the layer defines up to nine positions available for packing the next residual boxes (see Fig. 6 in Appendix A). Every packed box also makes unavailable the positions it occupies.

In order to try to define the position and orientation that the next residual box is packed in an incomplete layer, two heuristics are used:

(a) *First Fit* (FF). The available positions in the incomplete layer are kept in a list in lexicographic order along axes *z*, *y* and *x*, in this order. To try to pack the next residual box, we select the first position in this list and we try to place the box in it. If no constraint is violated, the box is placed with its current orientation in the selected position. Otherwise, we try again to place the box in the selected position, but with its other possible orientation (i.e., after horizontally rotating the box by 90°). If the box has not yet been packed, we select the next position in the list of available positions, if any, and the procedure is repeated (i.e., we try again to pack the box). Note that this heuristic is a variation of the *BackLeftLow* and *LeftBackLow* heuristics of Tarantilis, Zachariadis and Kiranoudis (2009) adapted to the present problem. With this heuristic, we try to pack each residual box first in the lower positions along the height of the incomplete layer, aiming at minimizing the final height of this layer;
(b) *Best Fit* (BF). The available positions in the incomplete layer are kept in a list in any order. We try to pack the next residual box in every available position with its two possible orientations. If any, we select the feasible combination (position and orientation) that maximizes the contact area between the faces of the current box with the faces of the boxes already packed (i.e., the current box is packed with this orientation). To favor the packing of boxes in the lower positions along the height of the incomplete layer, aiming at minimizing the final height of this layer, the contact area between the top and bottom faces of the boxes is not accounted for selecting the best feasible combination. In the case of a tie between the feasible combinations, we select the combination with the lowest position along the height

of the incomplete layer. Note that this heuristic is a variation of the *MaxTouchingAreaW, MaxTouchingAreaNoWallsW, MaxTouchingAreaL* and *MaxTouchingAreaNoWallsL* heuristics of Tarantilis et al. (2009) adapted to the present problem. With this heuristic, we try to favor a more compact packing of the residual boxes, and, at the same time, the placement of boxes in the lower positions along the height of the incomplete layer.

4.2.3. Avoiding the generation of layers that do not lead to a feasible solution

A layer may need to be on the top of some compartment (i.e., with no other layer above it) by at least one of the following reasons:

(a) *Stability*. The layer does not have an enough number of boxes and/or it is incomplete;
(b) *Fragility*. The load bearing strength of the boxes on the layer is not enough.

If in any moment of the route the number of layers that need to be on the top of some compartment is larger than the number of compartments, then a feasible solution cannot be obtained. To avoid this situation, during the generation of the incomplete layers and after the generation of the complete layers, we perform a procedure that aims at identifying and excluding the excess of layers that need to be on the top of some compartment. Fig. 11 (see Appendix B) describes the procedure *excL*(k) that is used for identifying and excluding the excess of layers that need to be on the top of some compartment at a stop k. Note that these removed boxes, as they are no longer packed in any layer, they are therefore residual boxes (by definition). We then try to pack these boxes in the incomplete layers left. As an incomplete layer can only be on the top of some compartment, this procedure prioritizes the exclusion of layers with lower weight.

4.3. Basic step 3

4.3.1. Packing all layers

After generating all layers (complete and incomplete), we try to pack them in the compartments by solving Model 1 (see Appendix C), a Mixed Integer Linear Programming Problem (MILPP) that considers all problem constraints and objectives. Note that the geometric arrangement of the complete and incomplete layers inside a compartment can be seen as a one-dimensional packing problem (with additional loading constraints). The objective function of Model 1 aims at minimizing the penalties for the handling of boxes along the route and the penalties for the load balancing deviations. A feasible solution for Model 1 consists in packing all layers while respecting the geometrical constraints, cargo stability constraints, and load bearing constraints. If a feasible solution for Model 1 is not obtained, we perform an iterative procedure in search for a feasible solution. At each iteration of this iterative procedure:

(a) We exclude a complete layer which can support less weight above it along the route (i.e., which has the smallest load bearing strength indicator), since it is possibly the hardest to combine with the remaining layers;
(b) We try to pack the boxes of this excluded layer in the incomplete layers (generating again the incomplete layers);
(c) With the new set of layers, we try to solve Model 1.

4.4. Basic step 4

4.4.1. Local search

If a feasible solution for the problem is obtained, we try to improve this solution with a local search procedure, which consists in seeking for an alternative packing for the residual boxes. First, all residual boxes are removed from the solution obtained, and we try to pack them back again on the top of the compartments using an iterative procedure. Fig. 12 (see Appendix B) describes the procedure *bLocal*() that is used to perform a local search on the first solution obtained by the heuristic. At each iteration of this procedure the following steps are performed:

(a) We try to assign the residual boxes to compartments by solving Model 2 (see Appendix C), another MILPP. The objective function of Model 2 aims at minimizing the penalties for the handling of residual boxes along the route and the penalties for the load balancing deviations. Note that this objective function assumes that the residual boxes are all packed in the compartments to which they are assigned. A feasible solution to Model 2 consists in packing all residual boxes in the compartments while respecting the geometrical constraints, vertical support constraints and a limitation on the total weight of the residual boxes assigned to a compartment.
(b) We try to sequentially pack the residual boxes in their respective compartments with a procedure similar to the one used for generating the initial incomplete layers (see Fig. 9). Fig. 13 (see Appendix B) describes the procedure *gInc2*() that is used to try to generate again the incomplete layers, according to the assignment of residual boxes to compartments that was defined by the procedure *bLocal*(). Again, the residual boxes are sorted in a list in inverse order of the vehicle stops. As a first tiebreaker, we use the height of the boxes in decreasing order, and, if necessary, as a second tiebreaker, we use the load bearing strength of the boxes in decreasing order. Following the defined order for the residual boxes, they are sequentially packed in their respective compartments, provided that they respect the packing constraints. Two attempts for packing the residual boxes are performed: in the first the position and orientation of the residual boxes in the incomplete layers are defined using the FF heuristic, while in the second the BF heuristic is used. Each attempt generates a partial solution (i.e., a subset of packed residual boxes) and we choose the solution whose sum of the volumes of the packed residual boxes is larger. Fig. 14 (see Appendix B) describes the procedure *pRes2*() that is used to try to define the orientation and position of the residual boxes inside the incomplete layers, according to the assignment of residual boxes to compartments that was defined by the procedure *bLocal*();
(c) The residual boxes assigned to a compartment that can be packed are fixed in it. To ensure the convergence of the procedure, if a box of a given type cannot be packed in a compartment, then the residual boxes of the same type are forbidden from being assigned to this compartment in the next iterations of the procedure.

5. Computational tests

In the following the generation of the test instances, the computational results and their analysis are presented.

5.1. Information of containers and boxes

The characteristics of containers and boxes used in the computational experiments are presented in the following. This information was obtained from data provided by the beverage company that motivated this study. The *relative volume* of boxes is the equivalent volume metric employed by the beverage company. For a box of type $i \in M$, the relative volume is not necessarily related to the real volume occupied by a box of this type (i.e., $l_i w_i h_i$), but it is given by $v_i^{rel} = 100/maxQ_i$, where $maxQ_i$ is the maximum number

Table 1
Information of the containers used in the computational experiments.

Container	Compartments ($c \in C$)										
	V^{rel}	1	2	3	4	5	6	7	8	9	10
2 compartments	200	*St*	*St*	–	–	–	–	–	–	–	–
4 compartments	400	*St*	*St*	*St*	*St*	–	–	–	–	–	–
6 compartments	600	*St*	*St*	*St*	*St*	*St*	*St*	–	–	–	–
8 compartments	760	*St*	*St*	*St*	*St*	*Lo*	*Lo*	*St*	*St*	–	–
10 compartments	960	*St*	*St*	*St*	*St*	*St*	*St*	*Lo*	*Lo*	*St*	*St*

Table 2
Dimensions of the compartment types used in the computational experiments.

Compartment Type	Length (cm)	Width (cm)	Height (cm)
Standard (*St*)	100	120	178
Lowered (*Lo*)	100	120	148

of boxes of type i that the company places above a pallet with this box type only for storage purposes. Each container also has a given maximum capacity defined for the *total relative volume* of boxes placed in it, which is given by V^{rel}, and we use this information in some of the instance classes to limit the volume of boxes that can be packed in the container (described ahead).

In total, we considered 5 container types. The capacities and the number of compartments available in each one of them are presented in Table 1. In this table, *St* and *Lo*, respectively, refer to the compartment types *standard* and *lowered*, while the symbol "– " indicates that the respective compartment is not present. Note that containers with eight and ten compartments have *lowered* compartments due to the axles and wheels of the truck. Note also that containers with two and four compartments are not used by the beverage company, although they are also found in practice. In the experiments, however, they are useful to evaluate the sensitivity of the proposed hybrid approach.

Table 2 presents the dimensions of the two types of compartments considered.

Note that it is assumed that the walls of the compartments have negligible thicknesses (see assumption A12 at the beginning of Section 4). Hence, $L = 2L^{comp}$ and $W = (|C|/2)W^{comp}$. In total, we considered 32 box types. The diverse characteristics of each one of them are presented in Table 3.

5.2. Data for the instances generation

To evaluate the performance of the proposed hybrid approach, 21 instance classes were considered, and for each class 10 instances were generated as described in the following.

5.2.1. Instances of Class 1

They correspond to instances 1R,...,10R. Instance 1R is a practical example obtained with data provided by the beverage company, while instances 2R,...,10R are generated from instance 1R by randomly choosing the order in which the customers (stops) are visited by the vehicle. With the instances of this class, the objective is to evaluate the performance of the proposed hybrid approach against a realistic case, and also to evaluate possible impacts on the solution of the problem with changes in the boxes delivery order.

As it was admitted, the first stop of the vehicle is destination 1, the second stop of the vehicle is destination 2, and so on. To keep this definition, instances 2R,...,10R are generated by exchanging the demands of the stops of instance 1R (see procedure *gInst1*() described in Fig. 15 of Appendix B). Note that this exchange is equivalent to changing the sequence in which the customers of instance 1R are visited by the vehicle. Instance 1R has a container with 8 compartments and 18 stops, and the demands of each customer (stop) are presented in Table 4.

Table 3
Data of the box types used in the computational experiments.

$i \in M$	l_i (cm)	w_i (cm)	h_i (cm)	P_i (kg)	σ_i (kg/cm^2)	v_i^{rel}
1	13	20	12	2.31	0.10	0.21
2	14	20	13	2.19	0.09	0.21
3	20	27	13	4.61	0.10	0.38
4	5	16	13	0.90	0.07	0.11
5	11	16	14	1.60	0.09	0.18
6	4	16	15	0.68	0.09	0.07
7	13	13	16	2.06	0.10	0.19
8	19	28	17	5.70	0.03	1.09
9	28	42	18	23.06	0.06	2.78
10	13	16	19	2.00	0.09	0.17
11	17	24	20	5.00	0.06	0.69
12	13	17	20	2.97	0.08	0.29
13	16	24	21	6.70	0.07	0.67
14	19	26	22	6.60	0.08	0.60
15	20	27	23	6.53	0.07	0.65
16	12	20	23	3.20	0.07	0.37
17	13	19	24	3.40	0.09	0.30
18	12	18	24	3.20	0.07	0.33
19	14	21	24	3.95	0.07	0.42
20	14	20	25	3.40	0.07	0.30
21	15	26	26	6.98	0.13	0.71
22	31	42	27	15.01	0.07	1.43
23	33	48	29	15.01	0.05	2.38
24	28	32	30	15.00	0.08	1.11
25	35	51	32	29.52	0.08	2.38
26	18	28	33	10.00	0.06	1.14
27	17	27	33	9.90	0.06	1.14
28	16	27	34	5.10	0.04	1.04
29	21	31	35	12.90	0.08	1.00
30	31	41	36	22.87	0.07	2.00
31	24	35	37	16.00	0.06	1.67
32	20	31	38	14.45	0.07	1.25
Min.	4	13	12	0.68	0.03	0.07
Max.	35	51	38	29.52	0.13	2.78
Avg.	18.09	26.53	23.94	8.21	0.08	0.85
Std. Dev.	7.60	9.76	7.91	7.28	0.02	0.72

5.2.2. Instances of Classes 2 to 21

They correspond to instances 1A,...,200A. These instances are randomly generated by varying the container types, the number of stops, the box types and their demands (see procedure *gInst2*() described in Fig. 16 of Appendix B). With the instances of these classes, the objective is to evaluate the performance of the proposed hybrid approach against cases not covered by the instances of Class 1.

In order to define an instance with this procedure, initially we try to generate the demand of some box type for each different stop of the vehicle. All generated demand is accepted (i.e., added to the instance) if its relative volume plus the relative volume of the demands already accepted are not larger than a given tolerance. Otherwise, we accept the largest possible parcel of the generated demand and the instance generation is finished. After trying to define the initial demand for each stop, the demand generation process is repeated for the same instance while it is possible.

Table 4
Data of the demands of the box types for each customer of instance 1R.

$k \in K$	$i \in M$												
	1	3	7	12	13	16	19	22	25	28	29	30	31
1	1	1	0	0	0	0	1	0	2	1	1	0	0
2	0	0	0	0	0	0	5	0	0	0	24	0	0
3	0	0	0	2	0	0	0	0	0	0	3	0	0
4	10	0	0	0	0	5	0	0	0	0	3	0	0
5	0	4	0	0	1	0	0	1	2	2	1	0	0
6	0	0	0	0	1	0	0	0	0	0	0	0	0
7	6	0	0	0	0	0	0	0	2	0	9	0	0
8	0	62	6	109	0	0	0	0	3	0	24	49	0
9	0	12	0	0	0	0	0	2	2	0	0	0	0
10	8	23	22	3	0	0	2	0	6	4	30	0	0
11	0	0	0	0	7	0	0	0	0	0	7	0	0
12	5	0	0	0	0	0	2	0	0	0	18	0	0
13	2	0	0	0	0	0	0	18	0	0	0	0	0
14	4	33	0	4	0	0	5	0	3	5	19	17	10
15	10	0	0	0	0	0	2	0	0	0	38	0	0
16	20	13	0	26	0	0	6	0	10	0	22	0	0
17	0	2	0	0	0	0	0	1	4	0	1	0	0
18	2	0	0	0	0	4	2	2	9	1	4	0	0

Table 5
Data used for the generation of problem instances of Classes 2 to 21.

Class	Instances	k^{mx}	v^{mx}	Container
2	1A,...,10A	3	0.8	2 compartments
3	11A,...,20A	3	0.9	2 compartments
4	21A,...,30A	4	0.8	2 compartments
5	31A,...,40A	4	0.9	2 compartments
6	41A,...,50A	6	0.8	4 compartments
7	51A,...,60A	6	0.9	4 compartments
8	61A,...,70A	8	0.8	4 compartments
9	71A,...,80A	8	0.9	4 compartments
10	81A,...,90A	9	0.8	6 compartments
11	91A,...,100A	9	0.9	6 compartments
12	101A,...,110A	12	0.8	6 compartments
13	111A,...,120A	12	0.9	6 compartments
14	121A,...,130A	12	0.8	8 compartments
15	131A,...,140A	12	0.9	8 compartments
16	141A,...,150A	16	0.8	8 compartments
17	151A,...,160A	16	0.9	8 compartments
18	161A,...,170A	15	0.8	10 compartments
19	171A,...,180A	15	0.9	10 compartments
20	181A,...,190A	20	0.8	10 compartments
21	191A,...,200A	20	0.9	10 compartments

Table 5 presents specific data used for the generation of the problem instances of Classes 2 to 21, where k^{mx} is the maximum number of stops that can be assigned, and v^{mx} is a parameter that constraints the total relative volume of boxes, $v^{mx} \in \mathbb{R}$, $0 \leq v^{mx} \leq 1$. The maximum number of box types that can be assigned is given by $n^{mx} = 2|C|$, and the largest demand of a box of type $i \in M$ that can be generated at once is given by $b_i^{mx} = \lceil 0.1V^{rel}/v_i^{rel} \rceil$.

5.3. Results generation

Models 1 and 2 were solved using software IBM ILOG CPLEX (version 12.5.1) with the default parameters. The computational time spent to solve any instance of Model 1 was limited to 1200 s (20 min). The implementation codes were written in C#, compiled in a 64-bit platform, and performed on a Intel Core i7-2860QM at 3.6 GHz, with 8GB DDR3 SDRAM. The random numbers generator used the method *System.Random* from the library of classes Microsoft .NET *Framework* 4. Consider the following notation used in the remaining of this section:

cZ: total value of the penalties for the handling of boxes;
cB: total value of the penalties for the load balancing deviations. Note that, according to expression (15) (see Appendix C), *cZ* has priority over *cB* (see also assumption A2 at the beginning of Section 4);
aGap: absolute gap. This value is given by the absolute difference between the smallest upper bound and the largest lower bound found;
rGap: relative gap. This value is given by the division of *aGap* by the smallest upper bound found;
t^{P1},t^{P2}: respectively the computational times, in seconds, spent with the solution of Models 1 and 2;
t^{est},t^{def}: respectively the computational times, in seconds, spent for generating the incomplete layers before and after the solution of Model 1 (i.e., respectively running the procedures described in Figs. 9 and 13);
t^{Δ}: total computational time, in seconds;
nPos: total number of available positions in the container for placing boxes and/or layers. In Model 1 this number is given by $\sum_{c \in C}(\mu_c + 1)$;
nBox: total number of required boxes;
nP1, *nP2*: respectively the number of times that we tried to solve Models 1 and 2 for a same instance;
nEx: number of layers identified as exceeding, and, therefore, excluded from the solution (see the procedure described in Fig. 11);
%res: percentage of residual boxes with relation to the total number of required boxes;
$\%BF^{est}$: percentage of times that the BF heuristic is chosen for packing the residual boxes in incomplete layers before solving Model 1. In the remaining times, the FF heuristic is chosen for this purpose (see the procedure described in Fig. 9);
$\%BF^{def}$: percentage of times that the BF heuristic is chosen for packing the residual boxes in incomplete layers after solving Model 2. In the remaining times, the FF heuristic is chosen for this purpose (see the procedure described in Fig. 13);
%P1L: percentage of instances in which the solution obtained after solving Model 1 could be improved with the local search procedure (see the procedure described in Fig. 12).

We tried to solve all instances of the 21 classes (C1,...,C21) with the proposed hybrid approach and Table 6 presents a summary of the obtained results. Due to limitations of space, we only present the average values for each class, which were rounded to two decimal digits. The values for *cZ*, *cB* and $|N''|$ refer to the best solution obtained for each instance, while the values for *rGap* refer to Model 1. Note that the values in the last line of this table correspond to an average of average values. The table also presents the detailed results for instance 1R.

5.4. Comments on the results

At least one solution was obtained by the proposed approach for all 210 instances evaluated in this section. In 205 out of these 210 instances, a solution could be obtained by the local search procedure described in Fig. 12 (local search). In 117 out of these 205 instances, the solution obtained with this procedure was better than the initial solution (obtained with Model 1). In 107 out of these 117 instances, the solution obtained was better only in what concerns the load balancing deviations (it was the same in what concerns the handling of boxes), while in the remaining 10 instances it was also better in what concerns the handling of boxes. The lower frequency of instances in which the penalties with the handling of boxes are decreased by the local search procedure can be explained by the fact that the boxes that are relocated, in most cases, are in complete layers, which have their positions already defined in the initial solution. Note that this fact is related to the heuristic strategy of defining the residual boxes as the ones from the first stops and also of initially trying to pack them in a smaller number of layers.

It can also be observed that the performance of the local search procedure tends to be better with instances with more compartments (see *%P1L*). With a larger number of compartments there are also more possibilities to assign boxes to compartments with Model 2 (used by the local search procedure), which increases the chances of a feasible solution to be found with this search. Note also that, in these cases, the average number of times in which the residual boxes are assigned with Model 2 also increase (see *nP*2). One possible reason for this is that Model 2 preferably seeks to assign residual boxes to compartments closer to the position of the geometric center of the container (in order to reduce the penalties with the load balancing deviations), and, therefore, in fewer compartments. As in these cases there is a larger number of residual boxes (in absolute terms), there is also a larger number of residual boxes assigned to compartments, which reduces the chances of these boxes to be packed at once, thus requiring new assignments (i.e., new solutions of Model 2).

In the experiments with instances of Class 1, as expected, we noted the influence of the delivery sequence in the objective function of the problem. The results indicate that either the handling of boxes or the load balancing can play a relevant role in the definition of routes of lower costs, in the case of a possible integrated approach with vehicle routing problems (see, e.g., Junqueira & Morabito, 2015; Hokama, Miyazawa & Xavier, 2016).

On average, for instances of Classes 2 to 21 with a same number of compartments, the penalties with the handling of boxes and the load balancing deviations always increase with the increase of the demands (i.e., with the increase of the value of v^{mx} used in the experiments). These penalties also usually increase with the number of vehicle stops. In particular, we observed a large increase of these penalties for Classes 17, 19 and 21. The solutions for some of the instances of these classes exceeded 1000 units of relocated boxes along the route.

The iterative procedure applied in the cases in which a feasible solution could not be obtained (see lines 3 to 15 in Fig. 7) needed to be run in only one of the evaluated instances (one instance of Class 4, causing an increase in the value of *nP*1). However, for this single instance it proved to be effective, being able to find a feasible solution.

The procedure used for identifying and excluding the excess of layers that need to be on the top of some compartment (see Fig. 11) needed to be run in only two of the evaluated instances (one instance of Class 6 and one instance of Class 8, see *nEx*). In total, three complete layers were excluded and their boxes were placed on incomplete layers with this procedure. In these two evaluated cases, the procedure proved to be effective, i.e., it allowed the generation of a feasible set of complete and incomplete layers that could later be all packed inside the container.

With relation to the FF and BF heuristics used for packing the residual boxes in the incomplete layers, we note that both were effective for generating the solutions. On average, for the incomplete layers generated with the procedure described in Fig. 9, both heuristics were chosen in an approximate number of times (see $\%BF^{est}$), while for the case in which the incomplete layers were generated with the procedure described in Fig. 13, the BF heuristic was always chosen in more than 62% of times (see $\%BF^{def}$). We note that the BF heuristic is chosen as the tiebreaker criterion in both procedures. For the case in which the incomplete layers were generated with the procedure described in Fig. 9, the BF heuristic is chosen for breaking a tie in 10% of times, while for the case in which the incomplete layers were generated with the procedure described in Fig. 13, the BF heuristic is chosen for breaking a tie in approximately 70% of times.

In the times in which the FF heuristic was chosen (i.e., in the times in which the BF heuristic was not chosen), it was due to the fact that it was able to generate a more compact packing (in the case of the procedure described in Fig. 9) or that it was able to pack a larger volume of boxes in a given iteration (in the case of the procedure described in Fig. 13), when compared to the BF heuristic. Note that the FF heuristic packs each residual box always in lower positions on a layer. As the boxes (of a same stop) being packed are sorted by their heights, the packing surface generated above them tends to be relatively flat, which facilitates the packing of the next boxes. On the other hand, the BF heuristic, when trying to place each box in a position that maximizes the contact area between the lateral faces of this box with the lateral faces of other boxes, although it favors a compact packing, it can generate a more fragmented packing surface than the FF heuristic, which in some cases makes it difficult the packing of the next boxes.

In what concerns the computational times, t^{P1} increase a lot with the increase in the number of compartments. This growth is not evident among the classes with more than four compartments (Class 1, and Classes 10 to 21), because in almost all instances of these classes the time limit defined for the solution of Model 1 was reached. Note that, for the instances with more compartments, the number of complete layers and the number of possible positions in the compartments are larger. These values, together with the number of vehicle stops, influenced the size of Model 1. In the case of t^{P2}, they usually increase with the increase in the number of compartments and residual boxes, but they are always low (on average, less than 1 s for almost all cases). t^{est} and t^{def} also increase with the increase in the number of residual boxes and they are always low (on average, they are always less than 2 and 4 s, respectively).

In 76 out of the 210 evaluated instances, the relative gaps (*rGap*) of the solutions obtained with Model 1 are equal to 0, i.e., these solutions were proven optimal by CPLEX. Note that these solutions are optimal for the set of complete and incomplete layers made available, which does not mean that these solutions cannot be improved with the procedure described in Fig. 12, once it modifies the incomplete layers already defined in previous steps of the proposed approach. In fact, it could be observed in some instances of classes with up to four compartments (Classes 2 to 9). Except for

Table 6
Summary of the results obtained with the proposed hybrid approach for instance 1R and for all instances of Classes 1 to 21.

	cZ	cB	t^{P1}	t^{P2}	t^{est}	t^{def}	t^{Δ}	$\|N'\|$	$\|N''\|$	$\%res$	nEx	$nP1$	$nP2$	$\%BF^{est}$	$\%BF^{def}$	$nPos$	$nBox$	P_1^{Δ}	$rGap$	$\%P1L$
1R	120.18	10,525.57	1200.89	0.62	1.39	3.68	1219.60	32.00	3.00	21.15	0.00	1.00	3.00	–	–	72.00	780.00	7358.98	1.00	–
C1	436.49	17,056.10	1200.68	0.64	1.32	3.07	1217.74	32.00	3.60	21.15	0.00	1.00	3.14	70.00	81.82	72.00	780.00	7358.98	1.00	40.00
C2	34.03	131.03	0.25	0.02	0.21	0.34	0.89	7.40	1.50	17.52	0.00	1.00	1.00	40.00	100.00	13.20	408.60	1535.96	0.00	50.00
C3	58.28	144.45	2.81	0.02	0.47	0.67	4.09	10.60	1.40	17.71	0.00	1.00	1.00	50.00	100.00	16.40	764.40	1805.34	0.00	40.00
C4	66.62	203.43	0.32	0.02	0.24	0.36	1.05	7.30	1.30	17.67	0.00	1.10	1.00	30.00	90.00	13.20	408.60	1535.96	0.00	30.00
C5	81.72	228.97	1.41	0.02	0.42	0.66	2.67	10.60	1.20	17.71	0.00	1.00	1.00	40.00	90.00	16.40	764.40	1805.34	0.00	20.00
C6	16.90	775.88	58.98	0.13	0.66	1.34	61.67	14.70	2.30	20.02	0.20	1.00	1.70	20.00	70.59	30.00	643.40	3028.79	0.00	20.00
C7	22.24	750.35	703.99	0.11	0.73	1.39	707.67	18.30	1.90	14.89	0.00	1.00	1.40	40.00	64.29	33.20	812.40	3584.54	0.10	50.00
C8	13.07	1025.70	48.28	0.17	0.73	1.34	51.43	14.90	2.60	19.02	0.10	1.00	1.50	50.00	86.67	30.00	643.40	3028.79	0.00	70.00
C9	26.87	1419.59	709.01	0.41	0.60	1.08	794.52	18.30	2.00	14.89	0.00	1.00	1.30	30.00	76.92	33.20	812.40	3584.54	0.37	60.00
C10	0.00	389.32	1200.66	0.21	0.74	1.44	1207.49	24.80	2.50	14.41	0.00	1.00	1.60	20.00	62.50	52.20	1039.00	4629.48	0.88	40.00
C11	88.01	1711.33	1099.30	0.21	0.76	1.50	1105.77	24.80	3.30	15.95	0.00	1.00	1.70	70.00	64.71	49.80	1031.30	5122.72	0.89	50.00
C12	2.47	1272.09	1201.39	0.29	0.66	1.46	1209.19	24.80	2.60	14.41	0.00	1.00	1.80	20.00	77.78	52.20	1039.00	4629.48	0.94	40.00
C13	22.39	2351.16	1201.51	0.29	0.72	1.47	1209.49	24.80	3.10	15.95	0.00	1.00	1.70	30.00	76.47	49.80	1031.30	5122.72	0.93	70.00
C14	0.00	582.08	1200.88	0.57	0.69	1.59	1210.96	30.10	4.60	13.32	0.00	1.00	1.60	40.00	81.25	64.20	1180.50	5914.39	0.99	70.00
C15	56.32	3564.29	1200.67	0.59	0.94	2.36	1213.97	35.90	3.70	12.70	0.00	1.00	2.40	50.00	87.50	73.00	1179.00	6808.12	0.99	60.00
C16	3.80	1652.58	1201.26	0.58	0.71	1.53	1213.78	30.10	3.70	13.32	0.00	1.00	1.50	50.00	86.67	64.20	1180.50	5914.39	0.97	80.00
C17	389.54	6489.75	1200.93	0.54	0.94	2.28	1218.34	35.90	3.70	12.70	0.00	1.00	2.50	40.00	64.00	73.00	1179.00	6808.12	1.00	70.00
C18	45.99	4514.83	1200.93	0.73	0.84	2.19	1222.15	42.30	4.60	9.95	0.00	1.00	2.00	30.00	80.00	91.20	2023.10	7814.03	1.00	90.00
C19	734.18	9767.12	1200.53	0.94	1.15	2.67	1228.36	44.40	4.40	12.20	0.00	1.00	2.22	40.00	90.00	93.80	2078.40	8691.06	1.00	70.00
C20	54.18	6886.02	1201.25	1.01	0.86	2.15	1230.47	42.40	5.00	9.92	0.00	1.00	1.90	20.00	78.95	91.10	2023.10	7814.03	0.99	90.00
C21	995.66	13,834.27	1200.99	0.89	1.22	3.07	1235.75	44.40	3.90	12.20	0.00	1.00	2.22	60.00	85.00	93.80	2078.40	8691.06	1.00	60.00
Avg.	149.94	3559.54	815.10	0.40	0.74	1.62	826.07	25.66	3.00	15.12	0.01	1.00	1.72	40.00	80.72	52.67	1157.15	5010.85	0.62	55.71

one single instance with six compartments, these 76 instances are all of classes with up to four compartments. For the remaining instances, the relative gaps are almost always equal to 1, or close to this value, which means that nothing (or very little) can be stated about the quality of these solutions, in relative terms (see the definition of *rGap*). However, in 149 out of the 210 instances, we note that the absolute gaps (*aGap*) of the solutions obtained are smaller than $\omega^z[\omega^{zq} + Min_{i \in M}(\omega^{zp} P_i)]$, i.e., the smallest unit of penalty for the handling of boxes. It means that these solutions are "optimal for the handling of boxes" (no unit of penalty for the handling of boxes can be further diminished from them).

The solutions obtained with the proposed approach were not compared to the solutions employed in practice, since they were not made available by the beverage company. During visits to this company, its logistics manager reaffirmed the difficulty in finding feasible vehicle loading plans that, besides packing all boxes of the route in the vehicle compartments, also meet all remaining constraints of the problem. The logistics manager found the solutions produced by the proposed approach potentially good to be used in practice.

6. Conclusion

In this paper, we presented a hybrid approach for the solution of a three-dimensional packing problem that arises in real situations, as it is the case of the beverage distribution using trucks with multi-compartment containers. The approach is based on the generation of horizontal layers and in the solution of smaller packing and assignment problems. The used formulations contemplate several practical packing considerations and they allowed that a large variety of instances based on real data could be solved. Several constraints must be met along the vehicle route, which can make it harder the solution of instances of larger sizes (i.e., with larger number of layers, stops, compartments and possible positions inside the compartments).

As it is usual in real situations, it was assumed that the vehicle route is known in advance (provided by a vehicle routing system) and that all required boxes must be packed inside the vehicle. In practice, if a feasible solution is not obtained, it is necessary to change the predefined route plan and/or to consider an additional new delivery route and/or to use a solution that violates some of the constraints involved. Computational tests were performed with this approach and a large variety of instances based on real data from a soft drink company that motivated this study, and the results show that the proposed solution approach was able to find feasible solutions for all considered test instances, which is hard to achieve in practice with the manual procedures that the company has available, due to difficulties with the arrangement of all boxes inside the compartments along the vehicle route, besides respecting other problem constraints. We also note that the decision makers from the beverage company found that the proposed solution approach has good potential for improving the manual procedures that the company has available, which usually entail a lot of handling of boxes inside the container along the truck route.

We note that this is the first study focusing on a vehicle packing problem with multi-compartment containers in the context of beverage distribution, and we believe that the proposed models and heuristic can be useful to motivate future researches that address and extend this problem. In particular, the proposed solution approach has good potential for being embedded into existing expert and intelligent systems for supporting the decision making process.

An interesting line of research would be to develop an effective validation of the proposed hybrid approach using it in the company on a daily basis to better evaluate the advantages and disadvantages of the loading plans generated by the approach in comparison with the ones used in the company. Some extension proposals also include:

(a) Enhance the proposed solution approach, e.g., trying to initially solve Model 1 with a fast heuristic. We note that for some instances, particularly those with a larger number of layers and compartments, the solutions obtained relatively high penalties with the handling of boxes and the load balancing deviations. Additionally, for many of these instances it was neither possible to prove the optimality of the solutions obtained with Model 1, nor to obtain smaller relative gaps (within the defined 20 min time limit).
(b) Allow the possibility of relocating boxes, at each stop, to a different position inside the vehicle, for example, to other compartments. This relocation could be advantageous to strategically reduce the handling of boxes and to obtain feasible solutions to instances whose solutions would be previously infeasible. We remind that, in this study, it was considered the relocation of a box in only one special case, that is when all boxes on the layer below it are unloaded (see assumption A3 at the beginning of Section 4);
(c) Evaluate other objectives that can be related to reducing the boxes unloading time along the route, as, for example, trying to keep boxes from a same stop in fewer compartments and/or in closer compartments;
(d) Combine the three-dimensional container loading problem of this study with vehicle routing problems, so that the vehicle routing and cargo loading decisions are considered simultaneously.

Declaration of interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper

Credit authorship contribution statement

Rodolfo Ranck Júnior: Methodology, Software, Validation, Formal analysis, Data curation. **Horacio Hideki Yanasse:** Conceptualization, Project administration. **Reinaldo Morabito:** Conceptualization, Supervision. **Leonardo Junqueira:** Investigation, Writing - original draft, Writing - review & editing, Visualization.

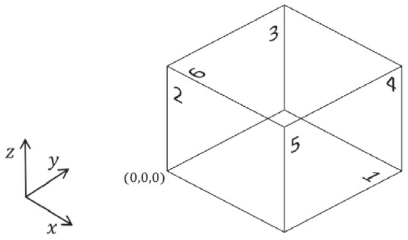
Acknowledgements

The authors would like to thank the two anonymous reviewers for their useful comments and suggestions, and the Brazilian soft drink company for the valuable collaboration with this study. This research was partially supported by CAPES, CNPq and FAPESP (Grant 14/16906-1).

Appendix A. Definitions and Notation

To indicate the position of items and objects, we use as reference the origin of the axes *x, y* and *z* that are adjacent to the faces 1, 2 and 3 of the cuboid shown in Fig. 5. This point is referred to as front-bottom-left corner (FBLC). Note that the container, the compartments, the layers and the boxes have all their own FBLC.

The notation employed in the two next sections follow a same format, $name^{specification}_{index\ 1,\ldots,index\ N}$, in which the name may come with some emphasis (e.g., an upper dash). The specification is used to differentiate notation with a same name. In the case of lists, to indicate a specific element, we employ the format $list\ name^{specification}_{index\ 1,\ldots,index\ N}[position\ in\ the\ list]$. Consider the following notation:



Face	Nomenclature
1	Bottom face
2	Front face
3	Left face
4	Back face
5	Right face
6	Top face

Fig. 5. Reference cuboid for items and objects.

Basic Sets:

K: set of vehicle stops;
C: set of container compartments;
N': set of complete layers;
N'': set of incomplete layers;
N: set of all layers, i.e., $N = N' \cup N''$;
M: set of required box types;
M': set of residual boxes.

Indexes:

k, k': a stop, $k, k' \in K$;
c: a compartment, $c \in C$;
j, j': a layer, $j, j' \in N$;
i: a box type, $i \in M$;
f: a box of any type and stop, $f \in M'$;
s, s': positions in a compartment for placing a layer $j \in N$;
p, p': possible positions along axis x;
q, q': possible positions along axis y;
r, r': possible positions along axis z.

Parameters:

$\mathcal{M}$: sufficiently large positive number;
L, W, H: length, width and height of the container, respectively along axes x, y and z;
L^{comp}: length of any compartment along axis x;
W^{comp}: width of any compartment along axis y;
H_c^{comp}: height of a compartment $c \in C$ along axis z;
p_c, q_c: position of a compartment $c \in C$ inside the container, respectively along axes x and y;
l_i, w_i, h_i: length, width and height of a box of type $i \in M$, respectively along axes x, y and z;
$\bar{h}_j$: height of a layer $j \in N$;
$\phi(f)$: type of a box $f \in M'$;
k_j^{max}: last stop in which boxes on layer $j \in N$ are unloaded;
b_i: number of residual boxes of type $i \in M$;
b_{ik}: number of boxes of type $i \in M$ required by stop $k \in K$;
P_i: weight of a box of type $i \in M$;
P_k^{Δ}: total weight of the cargo when only boxes from stops $k' \geq k \in K$ are inside the vehicle;
P_{jk}: weight of a layer $j \in N$ when only boxes from stops $k' \geq k \in K$ are inside the vehicle;
P_{ck}^{max}: maximum weight that a stack of complete layers, if any, placed in a compartment $c \in C$, can support above it when only boxes from stops $k' \geq k \in K$ are inside the vehicle. If there are no complete layers in compartment c, then $P_{ck}^{max} = \mathcal{M}$;
Q_{jk}: number of boxes on a layer $j \in N$ when only boxes from stops $k' \geq k \in K$ are inside the vehicle;
$Q_{fc}^{<}$: number of stops previous to the stop of the residual box $f \in M'$ in which the boxes on complete layers are unloaded from compartment $c \in C$. Therefore, it defines the number of times that box f will need to be relocated if it is packed in compartment c;
$\bar{\sigma}_j$: maximum supported pressure by any point on the top face of a box on the complete layer $j \in N'$;
$\bar{\psi}_j$: minimum number of boxes that the complete layer $j \in N'$ needs to have so that it can provide vertical stability for the boxes placed above it. In all the computational experiments (see Section 5.3) the value $\bar{\psi}_j = 4$, $j \in N'$ was used, that is, for any stop $k \in K$, four boxes on a complete layer $j \in N'$ (e.g., one at each corner) are enough to provide vertical stability to boxes placed above this layer;
π_{jk}: sum of all areas of the top faces of boxes placed on the complete layer $j \in N'$ when only boxes from stops $k' \geq k \in K$ are inside the vehicle;
R_j: indicator of the load bearing strength of the complete layer $j \in N'$ along the route. The higher this value, the more resistant to stacking a layer j is. It is assumed that this indicator for a given stop $k \in K$, when only boxes from stops $k' \geq k \in K$ are inside the vehicle, is given by: $\bar{\sigma}_j \pi_{jk} / P_k^{\Delta}$, if $Q_{jk} \geq \bar{\psi}_j$, and it is equal to 0 otherwise. From this relation, note that if layer j has $1, \ldots, \bar{\psi}_j - 1$ boxes, the referred indicator is equal to zero, once no box can be placed above it (for stability reasons). Note also that this indicator is higher the lower the total weight inside the vehicle for stop k (P_k^{Δ}) and, with this, we seek to reflect the load bearing strength of layer j against the other layers inside the container, i.e., its relative load bearing strength. As one aims an indicator for the whole route, we propose that: $R_j = \sum_{\{k \in K | Q_{jk} \geq \bar{\psi}_j\}} (\bar{\sigma}_j \pi_{jk} / P_k^{\Delta}) / k_j^{\max}$, $j \in N'$, i.e., R_j is given by the average of the load bearing strength indicators of the complete layer j for all stops from which it has boxes;
isBF: it is equal to 1 if the *Best Fit* (BF) heuristic for packing residual boxes is activated, and it is equal to 0 if the *First Fit* (FF) heuristic for packing residual boxes is activated;
α: parameter related to the desired vertical stability, $\alpha \in \mathbb{R}$, $0 \leq \alpha \leq 1$. In one extreme, $\alpha = 1$ indicates that the bottom face of any box must be 100% supported by the top faces of other boxes or by the object floor. In the other extreme, $\alpha = 0$ indicates that there is no concern about the vertical stability of the boxes. In all the computational experiments (see Section 5.3) the value $\alpha = 1$ was used, that is, the minimum vertical support must be 100% (i.e., full support). Note that, as $\alpha = 1$, a residual box cannot be placed above other residual boxes from previous stops (see procedure *vStab*(*f, p, q, r, j*) ahead) and, therefore, residual boxes can always be unloaded without relocating any other box;
β, γ: parameters related to the desired horizontal stability, $\beta \in \mathbb{R}$, $0 \leq \beta \leq 1$; $\gamma \in \mathbb{R}$, $0 \leq \gamma \leq 1$. In one extreme, $\beta = 1$ ($\gamma = 1$) indicates that the left (front) face of any

box must be 100% supported by the right (back) faces of other boxes or by the object left (front) wall. In the other extreme, $\beta = 0$ ($\gamma = 0$) indicates that there is no concern about the horizontal stability of the boxes. In all the computational experiments (see Section 5.3) the values $\beta = \gamma = 0$ were used, that is, no minimum horizontal support is required;

e_{jk}: it is equal to 1 if the complete layer $j \in N'$ has boxes from stop $k \in K$, and it is equal to 0 otherwise;

$e^{>}_{jk}$: it is equal to 1 if the complete layer $j \in N'$ has boxes from stops $k' > k \in K$, and it is equal to 0 otherwise;

$e^{\geq}_{fk}$: it is equal to 1 if the residual box $f \in M'$ is from a stop $k' \geq k \in K$, and it is equal to 0 otherwise;

$\rho_{jj'}$: it is equal to 1 if the complete layer $j \in N'$ has 0 or at least $\bar{\psi}_j$ boxes, when only boxes from stops $k' \geq k \in K$ are inside the vehicle, for $k = 1, \ldots, k^{max}_{j'}$. In this case, we say that layer j vertically stabilizes layer j'. Otherwise, it is equal to 0;

ρ'_{fc}: it is equal to 1 if there are no complete layers in compartment $c \in C$, or if any complete layer $j \in N'$ packed in compartment c has 0 or at least $\bar{\psi}_j$ boxes, when only boxes from stops $k' \geq k \in K$ are inside the vehicle, for $k = 1, \ldots, k^*$, in which k^* is the stop of the residual box $f \in M'$. In these cases, we say that compartment c provides vertical stability to box f. Otherwise, it is equal to 0;

g^{vx}, g^{vy}: ideal position of the center of gravity of the cargo inside the vehicle, respectively along axes x and y. This center of gravity is defined using the Cartesian coordinate system for the container, whose position (0,0,0) is located at the FBLC of the container. In all the computational experiments (see Section 5.3) the values $g^{vx} = L/2$ and $g^{vy} = W/2$ were used in Models 1 and 2 (see Appendix C), that is, the ideal position for the center of gravity of the vehicle is defined as the position of the geometric center of its container;

τ^x_{jk}, τ^y_{jk}: position of the center of gravity of layer $j \in N$, when only boxes from stops $k' \geq k \in K$ are inside the vehicle, respectively along axes x and y. This center of gravity is defined using the Cartesian coordinate system for the own layer, whose position (0,0,0) is located at the FBLC of the layer. These parameters are known in advance, before solving Model 1;

$\varepsilon^x, \varepsilon^y$: tolerance values for the load balancing deviations, respectively along axes x and y, such that $\varepsilon^x \geq 0$, $\varepsilon^y \geq 0$. In all the computational experiments (see Section 5.3) the values $\varepsilon^x = P^{\Delta}_1 / |C| \cdot 0.02 \cdot L$ and $\varepsilon^y = P^{\Delta}_1 / |C| \cdot 0.02 \cdot W$ were used in Models 1 and 2, that is, the tolerance for the load balancing deviations, respectively along axes x and y, is given by the average weight of the boxes in a compartment, when all required boxes are in the container, multiplied by 2% of the container sizes along axes x and y;

ω^{dev}, ω^z: relative weights, respectively, for the load balancing deviations and for the handling of boxes;

ω^{zp}, ω^{zq}: relative weights, respectively, for the weight and for the number of relocated boxes. In all the computational experiments (see Section 5.3) the values $\omega^{zp} = 0.2$ and $\omega^{zq} = 0.8$ were used in Models 1 and 2, that is, the penalty for the handling of boxes is given by 20% of the weight and 80% of the number of these boxes. Additionally, the weights for the handling of residual boxes along the route and for the load balancing deviations in the objective function are equal, that is, $\omega^{dev} = \omega^z = 1$ in Models 1 and 2;

μ_c: upper bound on the number of complete layers in a compartment $c \in C$;

μ'_c: maximum number of complete layers whose sum of heights is not larger than the height of a compartment $c \in C$;

μ''_c: maximum number of complete layers that can be stacked above the complete layer with largest load bearing strength plus one unit;

ζ: parameter that if $\zeta = 1$, μ_c is an upper bound on the number of complete layers in a compartment $c \in C$; if $\zeta < 1$, then it is not guaranteed that μ_c is still an upper bound and, therefore, it can make infeasible a solution to the problem. Note that the size of the problem grows with the value of ζ. In all the computational experiments (see Section 5.3) the value $\zeta = 0.7$ was used. This value was obtained after computational experiments with several instances of Model 1. For the computational experiments, $\zeta = 0.7$ was chosen for being the smallest value that did not imply significant changes in the solution quality, when compared to $\zeta = 1$;

y'_{jsc}: values of variables y_{jsc}, $j \in N$; $c \in C$; $s \in S_c$, obtained after solving Model 1.

Derived Subsets:

S_c: set of vertical positions in which a complete layer can be placed in a compartment $c \in C$. In a compartment, the lower-valued positions have lower height in this compartment (i.e., a layer placed in position s is below another layer placed in position $s' > s$ in the same compartment). This set is defined as:

$$S_c = \left\{ s \mid \; s \in \mathbb{N},\; 0 < s \leq \mu_c \; \right\}, c \in C$$

S_{jc}: set of vertical positions in which any layer $j \in N$ can be placed in a compartment $c \in C$. As the incomplete layers cannot be placed below the complete layers, but only in the last position of a compartment, this set is defined as:

$$S_{jc} = \left\{ s \mid \begin{array}{ll} s \in S_c, & if \; j \in N' \\ s = \mu_c + 1, & if \; j \in N'' \end{array} \right\}, \; j \in N; c \in C$$

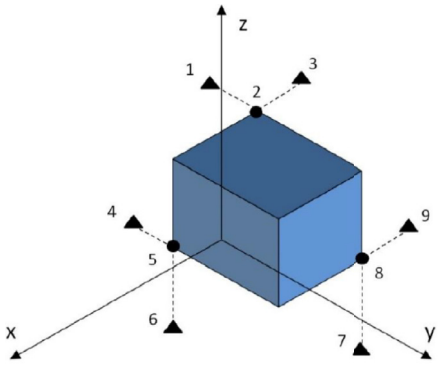
Lists:

$list^{comp}$: list that has only the container compartments sorted by their heights in increasing order;

$list^{pos}_j$: list of the available positions for packing residual boxes in the incomplete layer $j \in N''$. The positions in this list are kept in lexicographic order along axes z, y and x, in this order. Initially, there is no available position in this list;

$list^{res}$: list of all residual boxes $f \in M'$ sorted in inverse order of the vehicle stops. In the case of a tie, the boxes from the same stop are sorted by their heights in decreasing order. If the tie persists, the boxes from the same stop and with the same height are sorted by their load bearing strengths in decreasing order;

$list^{top}_k$: list of all generated layers that, when only boxes from stops $k' \geq k \in K$ are inside the vehicle, need to be on the top of some compartment by at least one of the following reasons: (a) stability: layer j has $1, \ldots, \bar{\psi}_j - 1$ boxes and/or it is incomplete; (b) fragility: layer j cannot be placed below any other layer due to the load bearing strength of its boxes. This list is sorted by the weights of the layers in increasing order.



Point	Coordinate
1	$(p, 0, r + h_i)$
2	$(p, q, r + h_i)$
3	$(0, q, r + h_i)$
4	$(p + l_i, 0, r)$
5	$(p + l_i, q, r)$
6	$(p + l_i, q, 0)$
7	$(p, q + w_i, 0)$
8	$(p, q + w_i, r)$
9	$(0, q + w_i, r)$

Fig. 6. Possible positions available for placing a residual box.

Auxiliary Procedures:

vOver(f, p, q, r, j): procedure that verifies if, when placing the residual box $f \in M'$ in position (p, q, r) of the incomplete layer $j \in N''$, box f does not overlap another box already packed and it is completely inside the incomplete layer $j \in N''$;

vStack(f, p, q, r, j): procedure that verifies if, when placing the residual box $f \in M'$ in position (p, q, r) of the incomplete layer $j \in N''$, the maximum supported pressure by any point (p', q', r') on the top faces of boxes already packed in the incomplete layer $j \in N''$ is not exceeded by the pressure applied by boxes placed above point (p', q', r');

vStab(f, p, q, r, j): being $i = \phi(f)$ and $k \in K$ the stop of the residual box $f \in M'$, this procedure verifies if, when placing box f in position (p, q, r) of the incomplete layer $j \in N''$: the area of the bottom face of box f in direct contact with the top faces of boxes from stops $k' \geq k$, or with the floor of the incomplete layer $j \in N''$, is not smaller than $\alpha(l_i w_i)$; the area of the left face of box f in direct contact with the right faces of boxes from stops $k' \geq k$, or with the left wall of the incomplete layer $j \in N''$, is not smaller than $\beta(h_i w_i)$; the area of the front face of box f in direct contact with the back faces of boxes from stops $k' \geq k$, or with the front wall of the incomplete layer $j \in N''$, is not smaller than $\gamma(l_i h_i)$.

The domains of the coordinates p, q and r in an incomplete layer $j \in N''$ (respectively with relation to axes x, y and z), candidates for some residual box to be placed in with its FBLC, are given by what are known as *extreme points* (Crainic, Perboli & Tadei, 2008). Once a residual box is placed on the layer it defines up to nine positions available for packing the next residual boxes. Fig. 6 shows these positions.

Appendix B. Related Pseudocodes

```
1.  Begin main()
2.      Generate the complete layers with procedure gComp() (Figure 8);
3.      Repeat
4.          Repeat
5.              Generate the incomplete layers with procedure gInc() (Figure 9);
6.              If some residual box was not packed in the incomplete layers
7.                  End main()
8.              Identify and exclude the possible excess of layers that need to be on the top of some
                compartment with procedure excL(k) (Figure 11), run for each k = 1, ..., |K|;
9.          While some layer is excluded with procedure excL(k);
10.         Try to pack all layers j by exactly solving Model 1 (Appendix C);
11.         If a feasible solution with Model 1 was not obtained
12.             If there are no more complete layers
13.                 End main()
14.             Remove the boxes on the complete layer j with the lowest R_j and exclude this layer;
15.     While a feasible solution with Model 1 is not obtained;
16.     Store a copy of the solution obtained (solution 1);
17.     Remove all boxes on the incomplete layers of the solution obtained and exclude these layers;
18.     Try again to pack the residual boxes in the obtained solution with procedure bLocal() (Figure 12);
19.     Evaluate the objective function value of the new feasible solution obtained in step 18 (solution 2), if
        any, with Model 1;
20.     Return the best of the two solutions obtained;
21. End main()
```

Fig. 7. Heuristic main procedure.

```
1.  Begin gComp()
2.    Define the number of generated layers equal to 0;
3.    For k = |K|, ... ,1
4.      For i ∈ M
5.        For f = 1, ... , b_ik
6.          If there is a layer j with boxes of type i and with available space (i.e., a layer with a
            positive number of boxes of type i smaller than the maximum number of boxes of this type
            that it can have)
7.            Assign the box f of type i and stop k to layer j;
8.          Otherwise
9.            Generate an empty layer and assign the box f of type i and stop k to it;
10.       End For
11.     End For
12.   End For
13.   Remove all boxes on layers that are not full of boxes (i.e., that are not complete layers) and exclude
      these layers;
14. End gComp()
```

Fig. 8. Procedure *gComp*() for generating the complete layers.

```
1.  Begin gInc()
2.    Define two auxiliary variables (sets) initially empty: N''(1) and N''(2);
3.    Activate the FF heuristic;
4.    Try to pack the residual boxes in incomplete layers with procedure pRes() (Figure 10);
5.    Store a copy of the generated incomplete layers in set N''(1);
6.    Activate the BF heuristic;
7.    Try again to pack the residual boxes in incomplete layers with procedure pRes();
8.    Store a copy of the generated incomplete layers in set N''(2);
9.    If not all residual boxes could be packed in the layers of set N''(1), or in the layers of set N''(2)
10.     Return Error;
11.   Else If all residual boxes could be packed in the layers of set N''(1), but not all of them could be
      packed in the layers of set N''(2)
12.     Keep only the incomplete layers obtained with the FF heuristic (the FF heuristic is chosen):
        N'' = N''(1);
13.   Else If all residual boxes could be packed in the layers of set N''(2), but not all of them could be
      packed in the layers of set N''(1)
14.     Keep only the incomplete layers obtained with the BF heuristic (the BF heuristic is chosen):
        N'' = N''(2);
15.   Otherwise all residual boxes could be packed either in the layers of set N''(1) or in the layers of set
      N''(2)
16.     If the total height of the incomplete layers of set N''(1) is smaller than the total height of the
        incomplete layers of set N''(2)
17.       Keep only the incomplete layers obtained with the FF heuristic (the FF heuristic is chosen):
          N'' = N''(1);
18.     Otherwise
19.       Keep only the incomplete layers obtained with the BF heuristic (the BF heuristic is chosen):
          N'' = N''(2);
20. End gInc()
```

Fig. 9. Procedure *gInc*() for generating the incomplete layers.

1. **Begin *pRes*()**
2. $j = 0$;
3. Remove the boxes on all incomplete layers j and exclude these layers;
4. **Repeat**
5. $j = j + 1$;
6. Generate a new incomplete layer (incomplete layer j);
7. Include position (0,0,0) in the initially empty $list_j^{pos}$;
8. $c = list^{comp}[j]$;
9. Define the limits for the length, width and height for packing the residual boxes in the incomplete layer j, respectively, as L^{comp}, W^{comp} and H_c^{comp};
10. **For** $countF = 1, \ldots, |list^{res}|$
11. **If** the box $list^{res}[countF]$ is already packed
12. Go to step 37 (next box);
13. $bestVal = -1$;
14. **For** $countPos = 1, \ldots, |list_j^{pos}|$
15. **For** $rot = 0$ to 1
16. $f = list^{res}[countF]$;
17. $(p, q, r) = list_j^{pos}[countPos]$;
18. Define the orientation of the residual box f by rotating it horizontally by 90º;
19. Verify if the residual box f can be placed, with its defined orientation, in position (p, q, r) of the incomplete layer j with procedures $vOver(f, p, q, r, j)$, $vStack(f, p, q, r, j)$ and $vStab(f, p, q, r, j)$ (Appendix A);
20. **If** the residual box f can be placed, with its defined orientation, in position (p, q, r) of the incomplete layer j
21. **If** $isBF = 1$
22. Assign to val the total contact area between the lateral faces (i.e., front, right, left and back) of the residual box f with the remaining boxes already packed in the incomplete layer j. To do this calculation, it is considered that the residual box f is packed, with its defined orientation, in position (p, q, r) of the incomplete layer j;
23. **If** $val > bestVal$
24. $bestVal = val$;
25. $bestPos = (p, q, r)$;
26. Store the orientation of box f in $bestO$;
27. **Else If**
28. Go to step 34;
29. **End For**
30. **End For**
31. **If** the residual box f could be placed in some position of the incomplete layer j
32. $(p, q, r) = bestPos$;
33. Define the orientation of box f according to $bestO$;
34. Place the residual box f, with its defined orientation, in position (p, q, r) of the incomplete layer j;
35. Add to $list_j^{pos}$ all the positions (see Figure 6 in Appendix A) not yet added to this list. To this end, it is considered that $i = \varphi(f)$;
36. Remove from $list_j^{pos}$ all the positions that were occupied by the residual box f;
37. **End For**
38. **While** there are boxes not packed in $list^{res}$ and $j < |C|$;
39. **End *pRes*()**

Fig. 10. Procedure *pRes*() for packing the residual boxes.

1. **Begin *excL(k)***
2. Remove from $list_k^{top}$ (i.e., the identified exceeding layers) all boxes on the $Max\{|list_k^{top}| - |C|, 0\}$ first complete layers and exclude these layers;
3. **End *excL(k)***

Fig. 11. Procedure *excL*(k) for identifying and excluding the excess of layers that need to be on the top of some compartment at a stop k.

1. **Begin *bLocal*()**
2. **While** there are residual boxes not packed
3. Try to assign the not yet packed residual boxes to compartments by solving Model 2 (Appendix C);
4. **If** a feasible solution with Model 2 could not be obtained
5. **End *bLocal*()**
6. Try to pack the residual boxes in their respective compartments (to which these boxes were assigned) with procedure *gInc2*() (Figure 13);
7. **For** $c \in C$
8. **For** $f \in M'$
9. **If** the residual box f was packed in compartment c
10. Fix the variable λ_{fc} to 1 (i.e., $\lambda_{fc} = 1$) (Appendix C);
11. **Otherwise**
12. Fix the variable λ_{fc} to 0 (i.e., $\lambda_{fc} = 0$) if some box of the same type of the residual box f could not be packed in compartment c;
13. **End For**
14. **End For**
15. **End While**
16. **End *bLocal*()**

Fig. 12. Procedure *bLocal*() for seeking an alternative packing for the residual boxes.

1. **Begin *gInc2*()**
2. Define the following auxiliary variables (lists and sets) initially empty: $list^{pos(*)}$, $list^{pos(1)}$, $list^{pos(2)}$, $N''^{(*)}$, $N''^{(1)}$ and $N''^{(2)}$;
3. Store a copy of the initial solution obtained for the incomplete layers: $list^{pos(*)} = list^{pos}$ and $N''^{(*)} = N''$;
4. Activate the FF heuristic;
5. Try to pack the residual boxes in incomplete layers with procedure $pRes2()$ (Figure 14);
6. Store a copy of the partial solution obtained for the incomplete layers: $list^{pos(1)} = list^{pos}$ and $N''^{(1)} = N''$;
7. Restore the initial solution obtained for the incomplete layers: $list^{pos} = list^{pos(*)}$ and $N'' = N''^{(*)}$;
8. Activate the BF heuristic;
9. Try again to pack the residual boxes in incomplete layers with procedure $pRes2()$;
10. Store a copy of the partial solution obtained for the incomplete layers: $list^{pos(2)} = list^{pos}$ and $N''^{(2)} = N''$;
11. **If** the total volume of the residual boxes packed in the incomplete layers of set $N''^{(1)}$ is larger than the total volume of the residual boxes packed in the incomplete layers of set $N''^{(2)}$
12. Keep only the partial solution obtained with the FF heuristic (the FF heuristic is chosen): $N'' = N''^{(1)}$ and $list^{pos} = list^{pos(1)}$;
13. **Otherwise**
14. Keep only the partial solution obtained with the BF heuristic (the BF heuristic is chosen): $N'' = N''^{(2)}$ and $list^{pos} = list^{pos(2)}$;
15. **End *gInc2*()**

Fig. 13. Procedure *gInc2*() for generating the incomplete layers.

Begin *pRes2*()
  For $countF = 1, \ldots, |list^{res}|$;
    If the box $list^{res}[countF]$ is already packed
      Go to step 35 (next box);
    $bestVal = -1$;
    $f = list^{res}[countF]$;
    Set c equal to the compartment to which the residual box f is assigned;
    If there is not yet an incomplete layer placed in compartment c
      Generate a new incomplete layer (incomplete layer c). Note that, without loss of generality, the index of an incomplete layer was defined as the same index of the compartment where it is packed (in this case the incomplete layers also store the information of the compartment where they are placed);
      Place the incomplete layer c in the last position (i.e., the top) of compartment c;
      Define the limits for the length and width for packing the residual boxes in the incomplete layer c, respectively, as L^{comp} and W^{comp};
      Define the limit for the height for packing the residual boxes in the incomplete layer c as H_c^{comp} minus the sum of the heights of the complete layers that possibly are packed in this compartment;
      Include position (0,0,0) in the initially empty $list_c^{pos}$;
    For $countPos = 1, \ldots, |list_c^{pos}|$
      For $rot = 0$ to 1
        $(p, q, r) = list_c^{pos}[countPos]$;
        Define the orientation of the residual box f by rotating it horizontally by 90º;
        If the residual box f can be placed, with its defined orientation, in position (p, q, r) of the incomplete layer c
          If $isBF = 1$
            Assign to val the total contact area between the lateral faces (i.e., front, right, left and back) of the residual box f with the remaining boxes already packed in the incomplete layer c. To do this calculation, it is considered that the residual box f is packed, with its defined orientation, in position (p, q, r) of the incomplete layer c;
            If $val > bestVal$
              $bestVal = val$;
              $bestPos = (p, q, r)$;
              Store the orientation of box f in $bestO$;
            Else If
              Go to step 32;
      End For
    End For
    If the residual box f could be placed in some position of the incomplete layer c
      $(p, q, r) = bestPos$;
      Define the orientation of box f according to $bestO$;
      Place the residual box f, with its defined orientation, in position (p, q, r) of the incomplete layer c;
      Add to $list_c^{pos}$ all the positions (see Figure 6 in Appendix A) not yet added to this list. To this end, it is considered that $i = \varphi(f)$;
      Remove from $list_c^{pos}$ all the positions that were occupied by the residual box f;
  End For
End *pRes2*()

Fig. 14. Procedure *pRes2*() for packing the residual boxes.

Begin *gInst1*()
  For $aux = 2, \ldots, 10$
    Define an initially empty instance (instance $auxR$);
    Define the container of instance $auxR$ as the container of instance 1R;
    $k' = 1$;
    While there are stops from instance 1R not chosen for the generation of instance $auxR$
      Randomly select a stop k from instance 1R not yet chosen for the generation of instance $auxR$;
      Define the demand of stop k' of instance $auxR$ as the demand of stop k of instance 1R;
      $k' = k' + 1$;
    End While
  End For
End *gInst1*()

Fig. 15. Procedure *gInst*1() for the generation of the problem instances 2R,...,10R of Class 1.

1. **Begin** ***gInst2***()
2. **For** $aux = 1, \ldots, 200$
3. Define an initially empty instance (instance $auxA$);
4. Define the following parameters for generating instance $auxA$: k^{mx}, v^{mx} and the container, according to Table 5; n^{mx} and b_i^{mx} (see Section 5.2);
5. $totalVol = 0$; $continue = 1$;
6. Define two auxiliary lists L^s and L^t initially empty;
7. Randomly select n^{mx} different box types from Table 3 and store them in list L^t;
8. **While** $continue = 1$
9. **If** list L^s is empty
10. Store in list L^s the stops $1, \ldots, k^{mx}$;
11. Randomly select a stop k from list L^s and remove it from this list;
12. Randomly select a box type i from list L^t;
13. Randomly select an integer value ξ between $[1, \ldots, b_i^{mx}]$;
14. **If** $totalVol + \left(\xi v_i^{rel}\right) \leq v^{mx}V^{rel}$
15. $totalVol = totalVol + \left(\xi v_i^{rel}\right)$;
16. Add ξ units to the demand of the box of type i and stop k of instance $auxA$;
17. **Otherwise**
18. Add $\left\lfloor (v^{mx}V^{rel} - totalVol)/v_i^{rel} \right\rfloor$ units to the demand of the box of type i and stop k of instance $auxA$;
19. $continue = 0$;
20. **End While**
21. **End For**
22. **End** ***gInst2***()

Fig. 16. Procedure *gInst2*() for the generation of the problem instances of Classes 2 to 21.

Appendix C. Mathematical Formulations

The geometric arrangement of the complete and incomplete layers inside a compartment can be seen as a one-dimensional packing problem (with additional loading constraints). For this reason, note that it is not necessary to know the *absolute position* of each layer in the compartment, but only the order in which they are placed in this compartment, i.e., their *relative positions*, which are then considered in the formulations that follow. Note that the relative positions for placing a layer in a compartment start in 1, and not in 0, as in the case of the absolute positions.

Model 1 (i.e., expressions (1)–(16)) in the following aims at packing all boxes in the vehicle compartments considering all constraints and the problem objective function. However, Model 1 represents a simplified version of the problem, once the boxes must be all packed in the compartments by means of complete and incomplete layers, which were already generated in previous steps of the heuristic. Consider the following decision variables to this formulation:

Variables:

g_k^x, g_k^y : free (auxiliary) decision variables that define the real position of the center of gravity of the cargo, when only boxes from stops $k' \geq k \in K$ are inside the vehicle, respectively along axes x and y. This center of gravity is defined using the Cartesian coordinate system for the container, whose position (0,0,0) is located at the FBLC of the container;

y_{jsc}: binary decision variable that is equal to 1 if layer $j \in N$ is placed in position $s \in S_c$ of compartment $c \in C$, and it is equal to 0 otherwise;

$\bar{z}_{jk}$: binary decision variable that is equal to 1 if layer $j \in N$ has boxes from stops $k' > k \in K$ and it is placed above another layer that has boxes from stop k. In this case, these boxes from stops k' will need to be relocated at stop k. Otherwise, it is equal to 0;

θ_k^x, θ_k^y : non-negative decision variables that define the vehicle load balancing deviations (i.e., unbalancing), when only boxes from stops $k' \geq k \in K$ are inside the vehicle, respectively along axes x and y.

Model 1

Formulation:

$$Min\ \omega^z \sum_{k \in K \setminus \{|K|\}} \sum_{j \in N} \left(\omega^{zq} Q_{jk+1} + \omega^{zp} P_{jk+1}\right) \bar{z}_{jk} + \omega^{dev} \sum_{k \in K} \left(\theta_k^x + \theta_k^y\right) \tag{1}$$

$$\sum_{c \in C} \sum_{s \in S_{jc}} y_{jsc} = 1 \qquad j \in N \tag{2}$$

$$\sum_{j \in N} \sum_{s \in S_{jc}} \overline{h}_j y_{jsc} \leq H_c \qquad c \in C \tag{3}$$

$$\sum_{j \in N'} y_{jsc} \leq 1 \qquad c \in C; s \in S_c \tag{4}$$

$$\sum_{j \in N''} y_{jsc} \leq 1 \qquad c \in C; s = \mu_c + 1 \tag{5}$$

$$g_k^x = \left(\sum_{j \in N} \sum_{c \in C} \sum_{s \in S_{jc}} P_{jk} \left(p_c + \tau_{jk}^x\right) y_{jsc} \right) \Big/ P_k^{\Delta} \qquad k \in K \tag{6}$$

$$g_k^y = \left(\sum_{j \in N} \sum_{c \in C} \sum_{s \in S_{jc}} P_{jk} \left(q_c + \tau_{jk}^y\right) y_{jsc} \right) \Big/ P_k^{\Delta} \qquad k \in K \tag{7}$$

$$\begin{matrix} \theta_k^x \geq P_k^{\Delta}\left(g_k^x - g^{\nu x}\right) - \varepsilon^x \\ \theta_k^x \geq P_k^{\Delta}\left(g^{\nu x} - g_k^x\right) - \varepsilon^x \end{matrix} \qquad k \in K \tag{8}$$

$$\begin{matrix} \theta_k^y \geq P_k^{\Delta}\left(g_k^y - g^{\nu y}\right) - \varepsilon^y \\ \theta_k^y \geq P_k^{\Delta}\left(g^{\nu y} - g_k^y\right) - \varepsilon^y \end{matrix} \qquad k \in K \tag{9}$$

$$\sum_{\left\{\begin{matrix} s' \in S_c | \\ s' > s \end{matrix}\right\}} y_{j's'c} \leq \sum_{j \in N'} \rho_{jj'} y_{jsc} \qquad j' \in N'; c \in C; s \in S_c \setminus \{\mu_c\} \tag{10}$$

$$y_{j's'c} \le \sum_{j \in N'} \rho_{jj'}\, y_{jsc} + \left(1 - \sum_{j \in N'} y_{jsc}\right) \qquad j' \in N''; c \in C; s \in S_c; \; s' = \mu_c + 1 \tag{11}$$

$$\mathcal{M}\left(1 - y_{j'sc}\right) + \bar{\sigma}_{j'} y_{j'sc} \ge \left(\sum_{j \in N} \sum_{\left\{ \begin{array}{l} s' \in S_{jc} | \\ s' > s \end{array} \right\}} P_{jk} y_{js'c} \right) \Big/ \pi_{j'k} \qquad k \in K; j' \in N'; c \in C; s \in S_c; \pi_{j'k} \ne 0 \tag{12}$$

$$\sum_{\left\{ \begin{array}{l} s' \in S_{j'c} | \\ s' > s \end{array} \right\}} e^{>}_{j'k} y_{j's'c} - \bar{z}_{j'k} \le 1 - \sum_{j \in N'} e_{jk} y_{jsc} \qquad k \in K \backslash \{|K|\}; j' \in N; c \in C; s \in S_c \tag{13}$$

$$\begin{array}{ll} \theta^x_k,\ \theta^y_k \ge 0 & k \in K \\ y_{jsc} \in \{0,1\} & j \in N; c \in C; s \in S_c \\ \bar{z}_{jk} \in \{0,1\} & j \in N; k \in K \setminus \{|K|\} \end{array} \tag{14}$$

The *objective function* (1) aims at minimizing the penalties for the handling of boxes along the route (first parcel) and the penalties for the load balancing deviations (second parcel). Note that in this expression Q_{jk+1} and P_{jk+1} respectively correspond to the number and weight of boxes on layer $j \in N$ that will be relocated at stop $k \in K$ if $\bar{z}_{jk} = 1$.

Constraints (2) ensure that all layers $j \in N$ are packed in the compartments. Note that these constraints also ensure that a layer $j \in N$ is only placed in positions $s \in S_c$, $c \in C$, assigned to it. Constraints (3) ensure that the height of a layer stack placed in a compartment $c \in C$ is not larger than the height of this compartment. Constraints (4)–(5) ensure that no more than one layer $j \in N$ occupies any position in a compartment $c \in C$. Note that this condition is sufficient to avoid that the layers overlap each other inside a compartment.

Constraints (6)–(7) define the auxiliary variables g^x_k and g^y_k, respectively. Note in these constraints that $(p^c + \tau^x_{jk})$ and $(q^c + \tau^y_{jk})$ correspond to the position of the center of gravity of a layer $j \in N$, respectively along axes x and y. Constraints (8)–(9) define the load balancing deviations along axes x and y. These load balancing deviations, when only boxes from stops $k' \ge k \in K$ are inside the vehicle, are given by the absolute distance between the real position of the center of gravity of the cargo and the ideal position of the center of gravity of the vehicle, weighted by the cargo weight inside the vehicle. These deviations are accounted for if they are larger than the tolerances ε^x and ε^y, respectively, along axes x and y. We note that these constraints are only valid together with the objective function (1), that penalizes these deviations. Note also that, to hold the linearity of this formulation, an absolute value function is defined, without loss of generality, with two linear inequalities.

Constraints (10) ensure that a complete layer $j' \in N'$ can only be placed in a position $s' \in S_c | s' > s$ of compartment $c \in C$ (i.e., $y_{j's'c} = 1$), if in any position $s \in S_c$ below it in this compartment there is another complete layer $j \in N'$ that can vertically stabilize it (i.e., $\rho_{jj'} y_{jsc} = 1$). Note that these constraints never prevent a complete layer j' to be placed in position $s' = 1$ of compartment c, because in this case layer j' is vertically stabilized by the compartment floor. Note also that these constraints do not allow a complete layer j' to be placed above a position that is not occupied by any other layer (i.e., generating a "hole" in the compartment). Constraints (11) ensure that an incomplete layer $j' \in N''$ can only be placed in the position $s' = \mu_c + 1$ of compartment $c \in C$, if in each position $s \in S_c$ below it in this compartment: there is a complete layer $j \in N'$ that can vertically stabilize it (i.e., $\rho_{jj'} y_{jsc} = 1$); or there is no layer (i.e., $\sum_{j \in N'} y_{jsc} = 0$). Note that it is allowed to place an incomplete layer $j' \in N''$ above a position that is not occupied by any other layer, once it was defined that the incomplete layers can only be placed in the last position $(\mu_c + 1)$ of some compartment $c \in C$ (so they are never below another layer).

Constraints (12) ensure that, when only boxes from stops $k' \ge k \in K$ are inside the vehicle, the pressure applied over any point on the top face of a box on the complete layer $j' \in N'$ placed in position $s \in S_c$ of compartment $c \in C$ is not larger than the maximum pressure than can be supported by that point (i.e., $\bar{\sigma}_{j'} y_{j'sc}$). This pressure is given by the weight of layers $j \in N$ placed in positions $s' \in S_c$ above the complete layer j' in the same compartment $c \in C$, divided by the sum of the areas of the top faces of boxes placed on the complete layer j'. Note that if the complete layer j' is not in the evaluated position (i.e., $y_{j'sc} = 0$), these constraints are redundant. We also note that here were considered assumptions A7 and A8 at the beginning of Section 4.

Constraints (13) ensure that if a layer $j' \in N$ that has boxes from stops $k' > k \in K$ is placed in a position $s' \in S_{j'c}$ (i.e., $e^{>}_{j'k} y_{j's'c} = 1$) above a complete layer $j \in N'$ that has boxes from stop k placed in a position $s \in S_c$ (i.e., $e_{jk} y_{jsc} = 1$) of the same compartment $c \in C$, then the boxes on layer j' from stops k' will need to be relocated at stop k (i.e., $\bar{z}_{j'k} = 1$). Note that this handling of boxes is penalized in the objective function (1). Constraints (14) define the domain of the decision variables. Note that variables $\bar{z}_{jk}$ are not defined to $k = |K|$, because in the last vehicle stop the boxes from previous stops will have already been unloaded, and therefore it will no longer be necessary to relocate boxes.

It is considered that minimizing the handling of boxes has larger priority with relation to minimizing the load balancing, as it was stated in assumption A2 at the beginning of Section 4. Therefore, the penalized load balancing deviations, $\omega^{dev} \sum_{k \in K} (\theta^x_k + \theta^y_k)$, are defined in such a way that they are smaller than the smallest penalized relocated unit, $\omega^z[\omega^{zq} + Min_{i \in M}(\omega^{zp} P_i)]$. Note that the upper bounds on variables θ^x_k and θ^y_k are respectively given by $P^{\Delta}_k L - \varepsilon^x$ and $P^{\Delta}_k W - \varepsilon^y$. Therefore, the relative weight for the load balancing deviations is defined as:

$$\omega^{dev} = \omega^z[\omega^{zq} + Min_{i \in M}(\omega^{zp} P_i)] \Big/ \left[1 + \sum_{k \in K} \left(P^{\Delta}_k L - \varepsilon^x + P^{\Delta}_k W - \varepsilon^y\right)\right] \tag{15}$$

The bound μ_c is given by the following relation that uses the smallest of two known upper bounds (μ'_c and μ''_c) on the number of complete layers in a compartment $c \in C$:

$$\mu_c = |N'|/|C| + \zeta\left[Min\left(\mu'_c, \mu''_c\right) - |N'|/|C|\right] \qquad c \in C \tag{16}$$

Model 2 (i.e., expressions (8)–(9), (17)–(25)) in the following is used by the local search procedure of the heuristic (see Fig. 12 in Appendix B) to assign the residual boxes to the container compartments, where the complete layers are already packed, aiming at considering all constraints and the problem objective function. Consider the following decision variables to this formulation:

Variables:

λ_{fc}: binary decision variable that is equal to 1 if box $f \in M'$ is assigned to compartment $c \in C$, and it is equal to 0 otherwise.

Model 2

Formulation:

$$Min\ \omega^z \sum_{f \in M'} \sum_{c \in C} Q^{\le}_{fc} \left(\omega^{zq} + \omega^{zp} P_{\varphi(f)}\right) \lambda_{fc} + \ \omega^{dev} \sum_{k \in K} \left(\theta^x_k + \theta^y_k\right) \tag{17}$$

$$\theta_k^x \geq P_k^{\Delta}\left(g_k^x - g^{vx}\right) - \varepsilon^x \quad k \in K \qquad (8)$$
$$\theta_k^x \geq P_k^{\Delta}\left(g^{vx} - g_k^x\right) - \varepsilon^x$$

$$\theta_k^y \geq P_k^{\Delta}\left(g_k^y - g^{vy}\right) - \varepsilon^y \quad k \in K \qquad (9)$$
$$\theta_k^y \geq P_k^{\Delta}\left(g^{vy} - g_k^y\right) - \varepsilon^y$$

$$\sum_{c \in C} \lambda_{fc} = 1 \quad f \in M' \qquad (18)$$

$$\sum_{f \in M'} l_{\varphi(f)} w_{\varphi(f)} h_{\varphi(f)} \lambda_{fc} \leq L^{comp} W^{comp}\left(H_c^{comp} - \sum_{j \in N'} \sum_{s \in S_c} \bar{h}_j y'_{jsc}\right) \quad c \in C \qquad (19)$$

$$h_{\varphi(f)} \lambda_{fc} \leq \left(H_c^{comp} - \sum_{j \in N'} \sum_{s \in S_c} \bar{h}_j y'_{jsc}\right) \quad f \in M'; c \in C \qquad (20)$$

$$\lambda_{fc} \leq \rho'_{fc} \quad f \in M'; c \in C \qquad (21)$$

$$\sum_{f \in M'} P_{\varphi(f)} e^{\geq}_{fk} \lambda_{fc} \leq P_{ck}^{\max} \quad c \in C; k \in K \qquad (22)$$

$$g_k^x = \left(\begin{array}{c} \sum_{f \in M'} \sum_{c \in C} e^{\geq}_{fk} P_{\varphi(f)} (p_c + L^{comp}/2) \lambda_{fc} + \\ \sum_{j \in N'} \sum_{c \in C} \sum_{s \in S_c} P_{jk}\left(p_c + \tau_{jk}^x\right) y'_{jsc} \end{array}\right) \Big/ P_k^{\Delta} \quad k \in K \qquad (23)$$

$$g_k^y = \left(\begin{array}{c} \sum_{f \in M'} \sum_{c \in C} e^{\geq}_{fk} P_{\varphi(f)} (q_c + W^{comp}/2) \lambda_{fc} + \\ \sum_{j \in N'} \sum_{c \in C} \sum_{s \in S_c} P_{jk}\left(q_c + \tau_{jk}^y\right) y'_{jsc} \end{array}\right) \Big/ P_k^{\Delta} \quad k \in K \qquad (24)$$

$$\theta_k^x,\ \theta_k^y \geq 0 \quad k \in K$$
$$\lambda_{fc} \in \{0, 1\} \quad f \in M'; c \in C \qquad (25)$$

The *objective function* (17) aims at minimizing the penalties for the handling of residual boxes along the route (first parcel) and the penalties for the load balancing deviations (second parcel). Note that this objective function assumes that the residual boxes are all packed in the compartments to which they are assigned. Constraints (18) ensure that all residual boxes are assigned to compartments $c \in C$. Constraints (19) ensure that the total volume of residual boxes $f \in M'$ assigned to compartment $c \in C$ is not larger than the available volume in this compartment, given by the total volume of the compartment minus the volume occupied by complete layers that may already be packed in it. Constraints (20) ensure that a residual box $f \in M'$ is not assigned to a compartment $c \in C$ with available height smaller than the box height. This available height is given by the height of compartment c minus the total height of the stack of complete layers that may already be packed in it. Constraints (21) ensure that a residual box $f \in M'$ is only assigned to a compartment $c \in C$ that provides vertical support to it (i.e., $\rho'_{fc} = 1$). Constraints (22) limit to P_{ck}^{max}, for any stop $k \in K$, the total weight of the residual boxes assigned to a compartment $c \in C$. Constraints (8)–(9) and (23)–(24) have the same role as constraints (6)–(9) previously defined. The center of gravity of the set of residual boxes assigned to a compartment is estimated in the geometric center of this compartment. Constraints (25) define the domain of the decision variables.

References

Alonso, M. T., Alvarez-Valdes, R., Iori, M., & Parreño, F. (2019). Mathematical models for multi container loading problems with practical constraints. *Computers and Industrial Engineering, 127*, 722–733.

Araújo, O. C. B., & Armentano, V. A. (2007). A multi-start random constructive heuristic for the container loading problem. *Pesquisa Operacional, 27*(2), 311–331.

Araújo, E. J., Chaves, A. A., Salles Neto, L. L., & Azevedo, A. T. (2016). Pareto clustering search applied for 3D container ship loading plan problem. *Expert Systems with Applications, 44*, 50–57.

Araya, I., & Riff, M.-C. (2014). A beam search approach to the container loading problem. *Computers and Operations Research, 43*, 100–107.

Avella, P., Boccia, M., & Sforza, A. (2004). Solving a fuel delivery problem by heuristic and exact approaches. *European Journal of Operational Research, 152*(1), 170–179.

Bischoff, E. E. (2006). Three-dimensional packing of items with limited load bearing strength. *European Journal of Operational Research, 168*(3), 952–966.

Bischoff, E. E., & Ratcliff, M. S. W. (1995). Issues in the development of approaches to container loading. Omega, *23*(4), 377–390.

Bischoff, E. E., Janetz, F., & Ratcliff, M. S. W. (1995). Loading pallets with non-identical items. *European Journal of Operational Research, 84*(3), 681–692.

Bortfeldt, A., & Wäscher, G. (2013). Constraints in container loading – a state-of-the-art review. *European Journal of Operational Research, 229*(1), 1–20.

Bortfeldt, A., Gehring, H., & Mack, D. (2003). A parallel tabu search algorithm for solving the container loading problem. *Parallel Computing, 29*(5), 641–662.

Ceschia, S., & Schaerf, A. (2013). Local search for a multi-drop multi-container loading problem. *Journal of Heuristics, 19*(2), 275–294.

Chan, F. T. S., Bhagwat, R., Kumar, N., Tiwari, M. K., & Lam, P. (2006). Development of a decision support system for air-cargo pallets loading problem: A case of study. *Expert Systems with Applications, 31*(3), 472–485.

Chen, C. S., Lee, S. M., & Shen, Q. S. (1995). An analytical model for the container loading problem. *European Journal of Operational Research, 80*(1), 68–76.

Crainic, T. G., Perboli, G., & Tadei, R. (2008). Extreme point-based heuristics for three-dimensional bin packing. *INFORMS Journal on Computing, 20*(3), 368–384.

Davies, A. P., & Bischoff, E. E. (1999). Weight distribution considerations in container loading. *European Journal of Operational Research, 114*(3), 509–527.

Derigs, U., Gottlieb, J., Kalkoff, J., Piesche, M., Rothlauf, F., & Vogel, U. (2011). Vehicle routing with compartments: Applications, modelling and heuristics. *OR Spectrum, 33*(4), 885–914.

Eley, M. (2002). Solving container loading problems by block arrangements. *European Journal of Operational Research, 141*(2), 393–409.

Eley, M. (2003). A bottleneck assignment approach to the multiple container loading problem. *OR Spectrum, 25*(1), 45–60.

Fanslau, M., & Bortfeldt, A. (2010). A tree search algorithm for solving the container loading problem. *INFORMS Journal on Computing, 22*(2), 222–235.

Gehring, H., & Bortfeldt, A. (1997). A genetic algorithm for solving the container loading problem. *International Transactions in Operational Research, 4*(5–6), 401–418.

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

George, J. A., & Robinson, D. F. (1980). A heuristic for packing boxes into a container. *Computers and Operations Research, 7*(3), 147–156.

Gonçalves, J. F., & Resende, M. G. C. (2012). A parallel multi-population biased random-key genetic algorithm for a container loading problem. *Computers and Operations Research, 39*(2), 179–190.

Haessler, R. W., & Talbot, F. B. (1990). Load planning for shipments of low density products. *European Journal of Operational Research, 44*(2), 289–299.

Henke, T., Speranza, M. G., & Wäscher, G. (2015). The multi-compartment vehicle routing problem with flexible compartment sizes. *European Journal of Operational Research, 246*(3), 730–743.

Hifi, M. (2002). Approximate algorithms for the container loading problem. *International Transactions in Operational Research, 9*(6), 747–774.

Hifi, M. (2004). Exact algorithms for unconstrained three-dimensional cutting problems: A comparative study. *Computers and Operations Research, 31*(5), 657–674.

Hokama, P., Miyazawa, F. K., & Xavier, E. C. (2016). A branch-and-cut approach for the vehicle routing problem with loading constraints. *Expert Systems with Applications, 47*, 1–13.

Junqueira, L., Morabito, R., & Yamashita, D. S. (2012a). Three-dimensional container loading models with cargo stability and load bearing constraints. *Computers and Operations Research, 39*(1), 74–85.

Junqueira, L., Morabito, R., & Yamashita, D. S. (2012b). MIP-based approaches for the container loading problem with multi-drop constraints. *Annals of Operations Research, 199*(1), 51–75.

Junqueira, L., Oliveira, J. F., Carravilla, M. A., & Morabito, R. (2013). An optimization model for the vehicle routing problem with practical three-dimensional loading constraints. *International Transactions in Operational Research, 20*(5), 645–666.

Junqueira, L., & Morabito, R. (2015). Heuristic algorithms for a three-dimensional loading capacitated vehicle routing problem in a carrier. *Computers and Industrial Engineering, 88*, 110–130.

Lahyani, R., Coelho, L. C., Khemakhem, M., Laporte, G., & Semet, F. (2015). A multi-compartment vehicle routing problem arising in the collection of olive oil in Tunisia. Omega, *51*, 1–10.

Lai, K. K., Xue, J., & Xu, B. (1998). Container packing in a multi-customer delivering operation. *Computers and Industrial Engineering, 35*(1–2), 323–326.
Martello, S., Pisinger, D., & Vigo, D. (2000). The three-dimensional bin packing problem. *Operations Research, 48*(2), 256–267.
Miyazawa, F. K., & Wakabayashi, Y. (1999). Approximation algorithms for the orthogonal Z-oriented three-dimensional packing problem. *SIAM Journal on Computing, 29*(3), 1008–1029.
Moura, A., & Oliveira, J. F. (2005). A GRASP approach to the container-loading problem. *IEEE Intelligent Systems, 4*(20), 50–57.
Morabito, R., & Arenales, M. (1994). An And/Or-graph approach to the container loading problem. *International Transactions in Operational Research, 1*(1), 59–73.
Norden, L., & Velde, S. (2005). Multi-product lot-sizing with a transportation capacity reservation contract. *European Journal of Operational Research, 165*(1), 127–138.
Pacino, D., & Jensen, R. M. (2012). Constraint-based local search for container stowage slot planning. *Lecture Notes in Engineering and Computer Science, 2*, 1467–1472.
Parreño, F., Alvarez-Valdes, R., Oliveira, J. F., & Tamarit, J. M. (2010). Neighborhood structures for the container loading problem: A VNS implementation. *Journal of Heuristics, 16*(1), 1–22.
Pisinger, D. (2002). Heuristics for the container loading problem. *European Journal of Operational Research, 141*(2), 382–392.
Ramos, A. G., Oliveira, J. F., Gonçalves, J. F., & Lopes, M. P. (2016). A container loading algorithm with static mechanical equilibrium stability constraints. *Transportation Research Part B: Methodological, 91*, 565–581.
Ramos, A. G., Silva, E., & Oliveira, J. F. (2018). A new load balance methodology for container loading problem in road transportation. *European Journal of Operational Research, 266*(3), 1140–1152.
Sciomachen, A., & Tanfani, E. (2003). The master bay plan problem: A solution method based on its connection to the three-dimensional bin packing problem. *IMA Journal of Management Mathematics, 14*(3), 251–269.
Sciomachen, A., & Tanfani, E. (2007). A 3D-BPP approach for optimising stowage plans and terminal productivity. *European Journal of Operational Research, 183*(3), 1433–1446.
Silva, J. L. C., 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*(2), 141–153.
Tarantilis, C. D., Zachariadis, E. E., & Kiranoudis, C. T. (2009). A hybrid metaheuristic algorithm for the integrated vehicle routing and three-dimensional container-loading problem. *IEEE Transactions on Intelligent Transportation Systems, 10*(2), 255–271.
Terno, J., Scheithauer, G., Sommerweiss, U., & Riehme, J. (2000). An efficient approach for the multi-pallet loading problem. *European Journal of Operational Research, 123*(2), 372–381.
Wäscher, G., Haussner, H., & Schumann, H. (2007). An improved typology of cutting and packing problems. *European Journal of Operational Research, 183*(3), 1109–1130.
Zhao, X., Bennell, J. A., Bektaş, T., & Dowsland, K. (2016). A comparative review of 3D container loading algorithms. *International Transactions in Operational Research, 23*(1–2), 287–320.