A mixed-integer linear model for the Multiple Heterogeneous Knapsack Problem with realistic container loading constraints and bins' priority

Igor Deplano[a,*], Charly Lersteau[a] and Trung Thanh Nguyen[a,*]

[a]*Liverpool Logistics, Offshore and Marine Research Institute,*

Department of Maritime and Mechanical Engineering,

Liverpool John Moores University, Liverpool L3 3AF, United Kingdom.

E-mail: igor.deplano@gmail.com; C.L.Lersteau@ljmu.ac.uk; T.T.Nguyen@ljmu.ac.uk

Received DD MMMM YYYY; received in revised form DD MMMM YYYY; accepted DD MMMM YYYY

Abstract

We propose a mixed-integer linear model that solves the Multiple Heterogeneous Knapsack Problem, minimising the wasted space of the bins, taking into account their priority, and considering the following constraints: non overlapping, boundaries and positioning (both constrained and free), rotations (around the vertical axis), orthogonal displacement, weight limit, static stability, weight distribution in a pyramidal region and load bearing considering items' arbitrary centre of mass. In the experiments, the priority parameter is set to prioritize smaller bins, but the model is generic to cover other cases. We study the trade-off of adding more constraints to make the problem more realistic and the complexity of finding a solution. We introduce new metrics that facilitate the comparison of datasets used in experiments. These metrics allow the export of theoretical results in the industry. We also propose a constructive heuristic named Weight First Best Fit to handle large scale instances in a reasonable time.

Keywords: Linear programming, Multiple Heterogeneous Knapsack Problem, Container loading problem, Physical Internet, Packing, Heuristic

1. Introduction

The cutting and packing class of problems (Wäscher et al., 2007) has been studied in the field of Operational Research since the 1940s because of their wide range of applications in areas such as logistics, scheduling, and manufacturing.

* Corresponding authors(e-mail: igor.deplano@gmail.com, T.T.Nguyen@ljmu.ac.uk).

The aim of this paper is to develop a mixed integer linear programming model (MIP) for the Multi-Heterogeneous Knapsack Problem (MHKP), considering the Container Loading Problem (CLP) constraints including stability, weight limits, weight distribution and load bearing constraints (Bortfeldt and Wäscher, 2013), and minimising the wasted space of bins with higher priority. We want to extend the state of the art considering items with an arbitrary centre of mass and restricting the acceptable bins' global centre of mass in a pyramidal region. This model can be used as a base model for further research on heuristics for both the knapsack packing and bin packing problems. We study the impact of the CLP constraints on the complexity of the solution space. The pyramidal region of choice is a trade-off between the need for a packing solution and achieving desirable conditions for safe container handling operations such as lifting and truck transport.

The MHKP consists of packing a strongly heterogeneous set of box-shaped items into a weakly heterogeneous set of box-shaped bins such that the wasted space is minimized or equivalently the bins' occupied space is maximized (Bortfeldt and Wäscher, 2013). The item profit is equal to the item volume, which differs from the general Knapsack problem where items with the same volume may have different profits. Items must be placed inside the borders of the bin without overlapping each other and without floating in mid air. We consider orthogonal rotation of items over the vertical axis.

A key difference between knapsack problems and bin packing problems is that the former considers profit maximization whereas the latter focuses on the minimization of the number of bins used (Martello and Toth, 1990; Kellerer et al., 2004). The solution spaces of these problems overlap when these conditions are met: the bins are homogeneous, the profit in the knapsack is the occupied volume, and all the items can be placed in the available set of bins. If we are considering heterogeneous bins, the solution spaces of these problems overlap when the availability of each type of bin is limited and there is a filling preference, e.g. based on the packing efficiency. Without limits and preference, the bin type with the biggest volume will dominate all the other bin types in the bin packing problem.

We provide a general model that assigns a priority parameter on each bin. Without any loss of generality, we focus our experiments on a case study where smaller bins are preferred. This is done by assigning each bin, a priority equal to the inverse of the bin's volume. Such assignment produces a minimization of the wasted space while prioritizing smaller bins over bigger ones. This is because companies often prefer to deal with smaller bins as they are less expensive. Giving priority to filling small bins (rather than large bins) is very important in some industries, e.g. fast-moving consumer goods. An example is food delivery companies like Deliveroo and Uber Eats, where it is necessary to prioritise smaller boxes that can be carried by bicycles over larger boxes that have to be carried by cars/vans/trucks. This is so that the food can be delivered to every part of the area in the shortest time. Also it is cheaper and more environmental friendly. Similarly, for last-mile delivery companies such as Amazon Logistics, Amazon Flex, myHermes, DPDLocal and the like, priority will be given to smaller bins since they can fit into small vans and cars owned by the couriers. Other examples can be found in last-mile delivery for pharmaceutical, consumer electronic, personal care, household care products, branded and packaged food, spirits and tobacco. The similar prioritisation also underlies the principle of the Physical Internet, which is considered the future of logistics. In the Physical Internet initiative (Montreuil, 2012, 2011), priority is given to packing goods into small, modular boxes if possible. Then thanks to modularisation the smaller boxes can be attached together to be carried in bigger containers if needed. We exploit the work done so far in the Physical Internet by using their containers' dimensions as a proxy for dealing with real-world instances. In Sallez et al. (2016), Physical Internet Containers or π-containers are divided into three main categories: T-container, H-container, and P-container. Table 1 illustrates their planned dimensions: T-containers aim to replace 20 feet and 40 feet containers, H-containers are designed to fit in a pallet pallet sized surface, and P-containers have various dimensions aiming to directly contain goods. Containers inside the same category can be combined into groups and locked together (composition) while a container of a smaller category can be encapsulated in a container of a larger category (encapsulation). The locking mechanism is still under development but considering the available options (Landschützer et al., 2015), it is reasonable to assume that items will be placed over a grid (positioning will be restricted to certain points).

Table 1: Table of dimensions of π-containers (container types marked with * have ten dimension modifiers: 100% | 90% | ... | 20% | 10%)

Type	length	width	height
T-container	1.2m, 2.4m, 3.6m, 4.8m, 6m, 12m	1.2m, 2.4m	2.4m, 1.2m
H-container *	1.2m	1.2m	2.4m
P-container *	1.2m	1.2m	2.4m

It is important to underline that even if the problem instances arise from the Physical Internet Containers, the problem can be adapted to any common containers or box types by restricting the bins to standard dimensions, e.g. 20 feet and 40 feet for ISO-standard containers. From now on, we will adopt the common bin packing vocabulary: T-containers will be called bins and H-containers will be called items.

The first contribution of this paper is an MIP Model for the MHKP considering some CLP constraints, and minimising the wasted space of the higher prioritized bins. The main novelty of the model is the arbitrary items' centre of mass (CoM) and the acceptable global bins' CoM pyramidal shape. This creates a model that, for the first time, covers both the pallet-loading and the container-loading instances with or without interlocking mechanisms.

The second contribution is a study of the impact of the CLP constraints on the complexity of the solution space.

The third contribution is the introduction of new metrics to compare the datasets used for the experimental part. An effort in this direction is important because faster algorithms are often specialized algorithms that efficiently handle only particular cases of the problem (Wolpert and Macready, 1997). Having such metrics will also improve the export of theoretical results in the industry.

The last contribution is the introduction of a simple deterministic heuristic that we name Weight First Best Fit(WFBF). WFBF is an on-line heuristic that fills the available bins with one item at a time, sequentially, until the available bins or the available items are exhausted. The algorithm is capable of handling any problem size in a reasonable time.

This paper is organized as follows: in section 2 we will investigate the previous work done on CLP encountered in the logistics field. In section 3 we will discuss the model in detail. In section 4 we will discuss the heuristic WFBF and its limitations. Section 5 will describe the dataset used, and in section 6 we will show and discuss the computational experiments used to validate the model. The paper will end with conclusions and discussions of future research.

2. Literature review

Our literature review focuses on mathematical linear models for the three dimensional bin packing problem (3DBPP) or three dimensional knapsack packing problem (3DKPP) that tackle some CLP constraints such as weight distribution, load-bearing, and stability. Readers interested in approximated methods are referred to Zhao et al. (2016); Coffman Jr et al. (2013); Christensen et al. (2017).

The first application-oriented literature review on cutting and packing problems was published by Sweeney and Paternoster (1992), including papers written since 1940. Coffman Jr et al. (2004) published a bibliography on the bin packing problem. Martello and Toth (1990) and successively Kellerer et al. (2004) made a detailed analysis of the broad class of knapsack problems. Dyckhoff (1990) made the first categorization of the cutting and packing problems, further improved by Wäscher et al. (2007).

According to Bortfeldt and Wäscher (2013), the most frequently used constraints in the CLP can be categorized as follows: weight distribution, loading priorities, orientation, stacking or load-bearing; cargo-related: complete-shipment, allocation, positioning; load-related: stability, complexity patterns. For a state-of-the-art review of the CLP algorithms that also considers heuristics and metaheuristics, readers are referred to the work of Zhao et al. (2016).

The geometrical constraints are common to any packing problem (Crainic et al., 2012). They are about positioning items inside bins ensuring that they will not overlap and that they will not exceed the bin's dimensions. Positioning can be modelled in absolute terms (in a Cartesian system where the origin is in one vertex of the bin) or in relative terms (the position is relative to other items inside the bin) (Bortfeldt and Wäscher, 2013). Considering only the underlying geometrical constraints (orthogonal packing inside the bins), BPP remains an NP-Hard problem in a strong sense. Delorme et al. (2016) recently published a survey on exact methods focusing on the one-dimensional version of the problem, while for approximation methods to solve the multiple-dimensional version, the most recent surveys are by Christensen et al. (2017, 2016).

Regarding the CLP constraints, weight distribution refers to displacing items inside the bin in order to obtain an overall centre of mass inside a safe region or to minimize the distance from a reference point, which is usually placed in the centre of the container. The centre of mass position is considered important (Bortfeldt and Wäscher, 2013) because it can affect the safety of operations during lifting with cranes, vehicle stability and asymmetric tyre wear.

Trivella and Pisinger (2016) modelled this problem as an MIP model for the multidimensional case, not considering rotations, nor load-bearing and assuming a homogeneous density of rectangular shaped boxes. This implies that a bin's centre of mass falls in the centre of the box. The model is too hard to solve even for small instances. Thus, the authors proposed a multi-level local search heuristic that exploits the Fekete-Schepers interval graphs representation (Fekete et al., 2007; Fekete and Schepers, 2004).

Paquay et al. (2016) tackled a multiple bin size bin packing problem (MBSBPP) considering the following constraints: geometrical (non overlapping, boundaries, rotations and positioning), orthogonal displacement, weight limit, orientation, special shapes of the containers, stability, weight distribution and fragility (this is a special case of load-bearing, where items cannot bear any load). Even in this paper, the mass of items is considered homogeneous.

Load-bearing is the limit of weight that can be stacked over a box. Often it is represented as "do not place more than n boxes" or "do not place box i over box j". Junqueira et al. (2012) tackled the loading of a single bin considering vertical stability (the ability of boxes to not fall/move along the vertical axis) and horizontal stability (the capacity of boxes to not move along the axes that form the horizontal plane). Their load-bearing is constrained inside a 100% vertical stability, which implies that no gaps between items are allowed. Jin et al. (2003) modelled the 3DBPP with stability, heterogeneous bins, and rotations. The large-scale case is solved with an algorithm based on Tabu-Search for the assignment phase and a sub-volume heuristic for the packing phase. Hifi et al. (2010) modelled a 3DBPP for packing identical bins without considering additional constraints. They proposed a set of lower bound inequalities based on analogies with the parallel-machine scheduling problems. Ceschia and Schaerf (2013) developed two algorithms, one based on Tabu-Search and the other on Simulated Annealing. They considered rotations, load-bearing, multi-drop, weight limits and stability (full-support). De Queiroz and Miyazawa (2013) tackled the oriented two-dimensional strip packing problem (2SP), considering load balancing, load-bearing, and a multi-drop constraint.

The above review shows that currently there is no exact method for solving the MHKP with the following constraints despite these being very common in real scenarios: non overlapping, boundaries and positioning (both constrained and free), rotations (around vertical axis), orthogonal displacement, weight limit, static stability, feasible weight distribution in a pyramidal region and load-bearing considering items' arbitrary centre of mass. This research aims to fill this gap by proposing a mixed-integer linear model that solves MHKP while prioritising bins, and considering all the aforementioned constraints. Having such a model will help further research on heuristics and approximations models.

3. Problem definition

Definition 1 (Item)**.** An item $i \in I$ is a parallelepiped object with length l_i, width w_i and height h_i. The possible dimensions are combinations of a predefined set of dimensions (in this paper, without any loss of generality, we take the dimensions specified in Table 1 as an example for experimental purposes). Each item has its load bearing limit Λ_i, which is the maximal weight that the item can bear on its top surface. Moreover, each item has a centre of mass expressed in coordinates κ_i^x, κ_i^y, κ_i^z.

Definition 2 (Bin)**.** A bin $j \in J$ is a parallelepiped object with length L_j, width W_j and height H_j. The possible dimensions are combinations of the ones given in Table 1. Every bin has weight limit Ω_j, which represent the maximal weight that can be allocated inside, and a priority $p_j > 0$, which represents the priority, or penalty, applied to the bin in the selection process.

Definition 3 (Bin's Centre of Mass)**.** The feasible global centre of mass of bin j is described by the coordinates of its vertex, ϱ_j, and the ratio of its pyramidal base, ξ_j. The pyramid base is proportional to the bin's size, Fig 1 shows an example. The ideal CoM is placed in the centre of the of the bin's bottom face. This position maximises the safety for crane lifting operations, truck transport and eventually manual handling. The choice of the pyramidal region is a trade-off between the ideal CoM and the need to have feasible solutions. Restricting the base of the pyramid and moving the vertex will result in having a feasible region nearer to the ideal one.

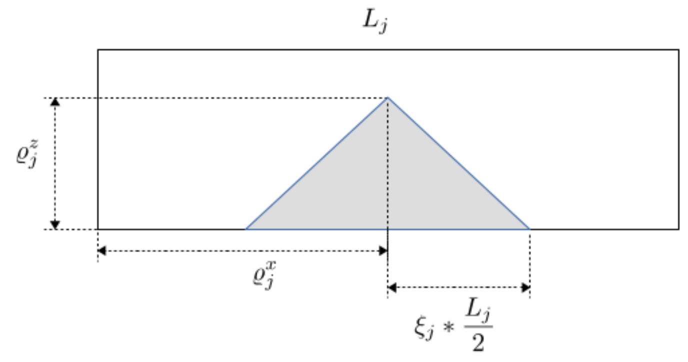

Fig. 1: Description of the feasible centre of mass pyramidal space

Definition 4 (Grid-based positioning)**.** The positions inside the bin along the plane parallel to the ground are restricted by a grid. Each grid cell is a square with β indicating the length of the side. β is the greatest common divisor of all items sides. This choice ensures item alignment. Items' positions are aligned on the grid. Note that this grid-based positioning can be changed to free positioning by setting $\beta = 1$. Fig. 2 illustrates the concept.

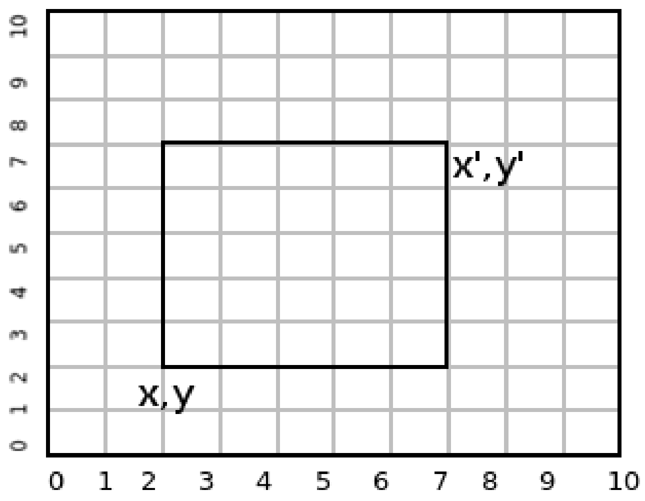

Fig. 2: Description of the feasible position along the x/y plane inside the bin

Definition 5 (Rotation of items)**.** Every item can rotate orthogonally around its vertical axis. The number of feasible rotations is two. Table 2 describes possible orientations.

Table 2: Table of orientations

Orientation Type	Parallel to x-axis	Parallel to y-axis	Parallel to z-axis
1	Length	Width	Height
2	Width	Length	Height

Definition 6 (Reflection)**.** The reflection operator is a π radiant rotation that affects only the item centre of mass. Fig 3 shows an example.

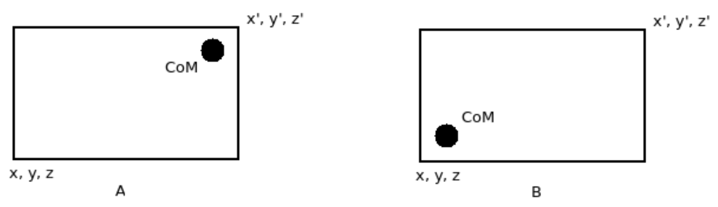

Fig. 3: A shows an item and its CoM, B shows the item's CoM after the reflection is applied. Note that the operator affects only the CoM position.

Definition 7 (Coordinate systems)**.** The proposed model uses two coordinate systems. Fig 4 shows the description. The first coordinate system has as its origin the bin's front-bottom-left corner, and is used to describe the positions inside the bin. The second coordinate system has as its origin the item's front-bottom-left corner and is used for defining the position of the CoM in an item.

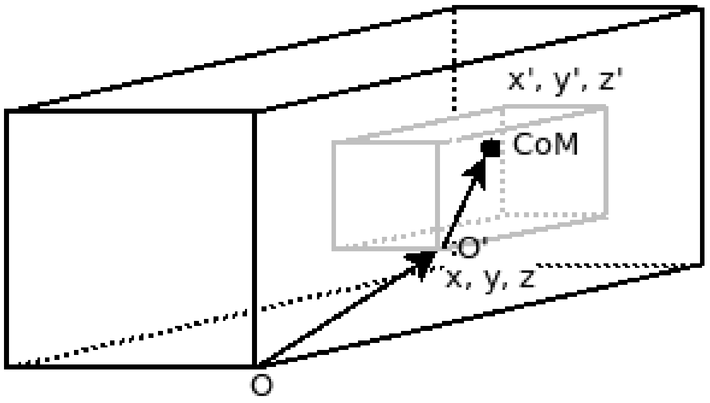

Fig. 4: Description of the coordinate system

Table 3: Model overview

min	wasted space of bins with higher priority
st:	
	each item must be assigned to at most one bin
	assigned items can rotate orthogonally along the vertical axis
	assigned items must be positioned on a grid
	assigned items must be placed inside bins limits
	assigned items must not overlap
	assigned items have to be placed on the ground or one or more items, they cannot float in middle air
	the sum of the items' weight assigned to a bin must be lower than or equal to the bin's supported weight limit
	for each assigned item, the stacked weight over it must be lower than its load bearing limit
	the global centre of mass of every bin must fall inside a pyramidal region

One of the aims of this paper is to give a mathematical model for the problem in Table 3. We developed the model considering items with an arbitrary centre of mass (CoM) and restricting the acceptable bins' global CoM in a pyramidal region. Items can rotate orthogonally around the vertical axis. This implies that there are two feasible rotations for every item. The overview is shown in Table 2. The rotation applied to an item affects the position of its CoM. Moreover, as we are considering arbitrary CoM, the model has to recalculate the CoM

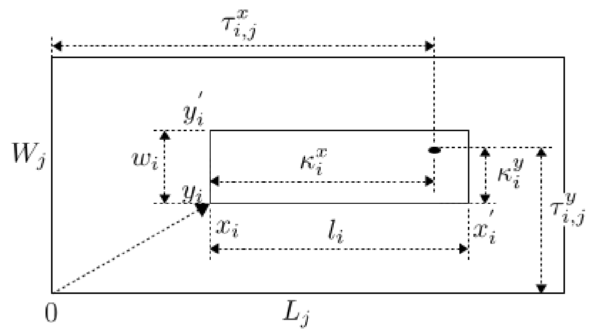

Fig. 5: Parameters and variables representation in two dimensions. The larger parallelepiped is the bin, the internal one is the item. For the sake of simplicity, this figure does not present rotations.

considering an additional operation, which is a π radiant rotation that we call reflection, that affects only the CoM of the item. Items are placed orthogonally, that is, items' sides are aligned with bin's sides.

In Table 4 we list all the model's parameters, Fig 5 shows a two-dimensional representation of the parameters and some variables. The variables of the model are presented in Table 5.

The rotation modelling is inspired by the work of Lin et al. (2014). $\varphi_{o,i} \in \{0,1\}$ is a binary variable that equals 1 if the orientation of item $i \in I$ follows orientation type $o \in O = \{1,\ldots,r\}$, where r is the number of the feasible rotations for an item (i.e 2). $\varphi_{o,i}$ will be 0 otherwise. The orientation types are defined in Table 2. We are considering only 2 orthogonal rotations in the plane x/y because it is a common practice to avoid any rotation of the z-axis. For example, in the MODULUSCHA project (MODULUSHCA, 2012), an H-container has a male/female mechanism on the top/bottom surface that limits the feasible rotations to only two along the ground plane. Moreover, items and bins are regular parallelepipeds and since we are aiming for optimal configuration, considering non-orthogonal rotations will only increase the complexity of the problem without improving solutions.

Items are aligned in a grid where each cell is a square with β indicating the length of the side. Aligned positioning is a common practice, examples being pallet positioning into ISO-standard containers, boxes inside a pallet or the locking mechanism (Landschützer et al., 2015) in the Physical Internet containers. Furthermore, this practice reduces the feasible search space and can be disabled by assigning $\beta = 1$. It is preferable that the solution includes the items' absolute positioning using one of the bin's corners as the origin of the Cartesian system. We define $\beta * x_i$, $\beta * y_i$, z_i, the coordinates of the front-bottom-left item corner.

3.1. The model

The model's objective (1) is to minimize the wasted space of the bins having higher priority.

Table 4: List of parameters

Name	Description
J	Set of bins' indexes
I	Set of items indexes
O	Set of available rotations
n	Cardinality of J
m	Cardinality of I
L_j	Length of bin j
W_j	Width of bin j
H_j	Height of bin j
p_j	priority of bin j, $p_j > 0$
L	Max length of bins in J
W	Max width of bins in J
H	Max height of bins in J
l_i	Length of item i
w_i	Width of item i
h_i	Height of item i
ω_i	Weight of item i
Ω_j	Weight limit of bin j
Λ_i	Load-bearing limit of item i
ϱ_j	Vertex of the pyramid for bin j
κ_i	centre of mass vector for item i
κ_i^s	With $s \in \{x, y, z\}$ coordinate s of the centre of mass for item i
ξ_j	Radius in the base of the pyramid for bin j
β	Dimension of the side of the square cell grid

Table 5: List of variables

Name	Description
x_i, y_i	Position of the item i in the grid, $\beta \cdot x_i$, $\beta \cdot y_i$, are the respective coordinates of the front-bottom-left edge
z_i	Coordinate of the front-bottom-left edge of the item i
$x_i^{'}, y_i^{'}, z_i^{'}$	Coordinate of the back-top-right edge of the item i.
$\varphi_{o,i}$	1 if the applied rotation for the item i is $o \in O$, otherwise 0
$S_{i,k}$	1 if item i and item k are assigned to the same bin, otherwise 0
Z_j	1 if bin j is considered in the solution, otherwise 0
$C_{i,j}$	1 if item i is assigned to bin j, otherwise 0
$x_{i,k}^p, y_{i,k}^p, z_{i,k}^p$	1 if item i is placed after item k along the coordinate x, y and z, which means respectively $x_k^{'} \leq x_i$, $y_k^{'} \leq y_i$, $z_k^{'} \leq z_i$, otherwise 0
$\pi_{i,k,p}$	$p \in \{0, 1, 2, 3\}$, 1 if the corner p of the item k is on the top surface of the item i, otherwise 0
$\theta_{i,m}$	1 if item i is placed on the ground of the bin, otherwise 0
$\theta_{i,k}$	1 if item i is placed on the top of item k, otherwise 0
R_i	1 if reflection is applied on item i, otherwise 0
$\upsilon_i^x, \upsilon_i^y$	Coordinate of the centre of mass of the item i after applying the item rotation and reflection, the reference is the origin of item i
$\tau_{i,j}^x, \tau_{i,j}^y, \tau_{i,j}^z$	Coordinate of the centre of mass for the item i using as reference the origin of the bin j
$\lambda_{i,k}$	1 if item k must be considered for the sum of the weights that are placed over the item i, otherwise 0

$$\text{minimize} \qquad \sum_{j\in J} p_j \cdot (L_j \cdot W_j \cdot H_j - \sum_{i\in I}(l_i \cdot w_i \cdot h_i \cdot C_{i,j})) \tag{1}$$

In the case study in this paper, without any loss of generality, we use the inverse of the bin's volume as the priority (2). This is equivalent to giving filling priority to smaller bins. Other possible values may be: $p_j = 1$, to minimize the global wasted space, a standard objective in the packing problems; where $p_j = c, c \in \mathbb{N}$ is an arbitrary value that shapes a preference of the filling sequence, i.e. shaping a priority on the filling of trucks, containers, or companies.

$$p_j = \frac{1}{(L_j \cdot W_j \cdot H_j)} \tag{2}$$

The objective (1), using the priority defined in (2), is equivalent to maximizing the packing efficiency. The proof for equivalence is trivial and is shown in Appendix A. Note that, using rational representation, the objective satisfies our requirements without having to add other penalties. There are other ways to model the objective to satisfy this requirement. For example, giving unused bins penalty zero would be a good approach if we added a specific constraint to force all items to be allocated.

The model is subject to the following constraints.

$$C_{i,j} \leq Z_j \qquad \forall i \in I, j \in J \tag{3a}$$

$$\sum_{j\in J} C_{i,j} \leq 1 \qquad \forall i \in I \tag{3b}$$

$$\sum_{i\in I} C_{i,j} \geq Z_j \qquad \forall j \in J \tag{3c}$$

$$\sum_{o\in O} \varphi_{o,i} = \sum_{j\in J} C_{i,j} \qquad \forall i \in I \tag{3d}$$

Our problem considers selecting a subset of items and bins. The total allocation of items is not mandatory because there may be a set I of items such that, given a set J of bins, there exists no valid solution with total allocation. Constraint (3a) forces to 0 every $C_{i,j}$ if the bin j has not been considered in the solution. Constraint (3b) ensures that every item is assigned to at most one bin, while constraint (3c) states that at least one item must be placed in an opened bin. Bins without at least one item inside are not considered a valid solution. Constraint (3d) forces only one rotation if the item is assigned, O are the available rotations, shown in Table 2.

$$\sum_{i \in I} C_{i,j} \cdot \omega i \leq \Omega_j \qquad \forall j \in J \tag{4}$$

$$\sum_{i \in I} C_{i,j} \cdot l_i \cdot w_i \cdot h_i \leq L_j \cdot W_j \cdot H_j \qquad \forall j \in J \tag{5}$$

Constraint (4) and (5) ensure that the content of each bin do not exceed a weight limit and a volume limit, respectively.

The positioning and non-overlapping constraints are based on the ones proposed by Paquay et al. (2016) adapted with the rotation style found in Lin et al. (2014). The main difference between them is that Paquay et al. (2016) adopted one boolean variable per side orientation, while Lin et al. (2014) adopted one variable per rotation. In detail, constraints (6a)-(6c) define the borders of the items along the axes, taking into account the possible rotation.

$$x_i^{'} - \beta \cdot x_i = l_i \cdot \varphi_{0,i} + w_i \cdot \varphi_{1,i} \qquad \forall i \in I \tag{6a}$$

$$y_i^{'} - \beta \cdot y_i = l_i \cdot \varphi_{1,i} + w_i \cdot \varphi_{0,i} \qquad \forall i \in I \tag{6b}$$

$$z_i^{'} - z_i = h_i \qquad \forall i \in I \tag{6c}$$

Constraints (7a)-(7c) ensure that the shape of each item is contained inside the borders of the selected bin. We precompute $L = \max_{j \in J} L_j$, $W = \max_{j \in J} W_j$, $H = \max_{j \in J} H_j$ as the maximum dimensions available for the considered set of bins. L, W and H are used as big M constraints.

$$x_i^{'} \leq \sum_{j \in J} C_{i,j} \cdot W_j + (1 - Z_j) \cdot W \qquad \forall i \in I \tag{7a}$$

$$y_i^{'} \leq \sum_{j \in J} C_{i,j} \cdot L_j + (1 - Z_j) \cdot L \qquad \forall i \in I \tag{7b}$$

$$z_i^{'} \leq \sum_{j \in J} C_{i,j} \cdot H_j + (1 - Z_j) \cdot H \qquad \forall i \in I \tag{7c}$$

Constraints (8a)-(8d) deal with the case when two items i,k are allocated to the same bin. The constraints enforce that $S_{i,k}$ is equal to 1 if and only if item i is in the same container of item k. $S_{i,k}$ is useful in other parts of the model.

$$S_{i,k} \leq \sum_{j \in J} C_{i,j} \qquad \forall j \in J, \forall i,k \in I | i \neq k \tag{8a}$$

$$S_{i,k} \leq \sum_{j \in J} C_{k,j} \qquad \forall j \in J, \forall i,k \in I | i \neq k \tag{8b}$$

$$S_{i,k} \geq C_{k,j} + C_{i,j} - 1 \qquad \forall j \in J, \forall i,k \in I | i \neq k \tag{8c}$$

$$S_{i,k} + C_{i,j} + C_{k,l} \leq 2 \qquad \forall j,l \in J | l \neq j, \forall i,k \in I | i \neq k \tag{8d}$$

The non overlapping constraints are basically the same as in Paquay et al. (2016) with minor modifications: we utilized $S_{i,k}$ on the right part of constraint (9a) and stated clearly the coordinates considering the position in the grid. Constraint (9a) states that two items do not overlap if the boundaries along at least one dimension do not overlap. The overlapping boundaries are characterized by the constraints (9b)-(9d).

$$x^p_{i,k} + y^p_{i,k} + z^p_{i,k} + x^p_{k,i} + y^p_{k,i} + z^p_{k,i} \geq S_{i,k} \qquad \forall i,k \in I | i \neq k \tag{9a}$$

$$\begin{aligned} x'_k &\leq \beta \cdot x_i + (1 - x^p_{i,k}) \cdot W \\ \beta \cdot x_i + 1 &\leq x'_k + x^p_{i,k} \cdot W \end{aligned} \qquad \forall i,k \in I | i \neq k \tag{9b}$$

$$\begin{aligned} y'_k &\leq \beta \cdot y_i + (1 - y^p_{i,k}) \cdot L \\ \beta \cdot y_i + 1 &\leq y'_k + y^p_{i,k} \cdot L \end{aligned} \qquad \forall i,k \in I | i \neq k \tag{9c}$$

$$z'_k \leq z_i + (1 - z^p_{i,k}) \cdot H \qquad \forall i,k \in I | i \neq k \tag{9d}$$

$$z^p_{i,k} = 0, x^p_{i,k} = 0, y^p_{i,k} = 0 \qquad \forall i,k \in I | i = k \tag{9e}$$

The stability is defined by constraining every item to lie on the bin's floor (10a) or on at least one other item (10b)-(10p). Constraint (10q) excludes the possibility of having one item both simultaneously lying over another one $\theta_{i,k} = 1$ and on the ground ($\theta_{i,m} = 1$). $\pi_{i,k,l}, l \in \{0,1,2,3\}$ represents which corner of item k is under item i.

Constraints (10b) and (10c) define one of the conditions for item i to be stable over an item k is $z_i = z'_k$, i.e. the top surface of item k is at the same height as the bottom surface of item i. The criteria for defining whether an item is on top of another in the x/y plane are defined respectively in (10d) - (10g) for the x axis and in (10h) - (10k) for the y axis. The constraints (10l) - (10o) enforce that if item k is on top of item i, at least two bottom corners of k should be on the top surface of i to maintain stability. (10p) states that two items can be considered stable only if they are in the same container.

$$z_i \leq H \cdot (1 - \theta_{i,m}) \qquad \forall i \in I \tag{10a}$$

$$z_i \leq H \cdot (1 - \theta_{i,k}) + z_k^{'} \qquad \forall i, k \in I | i \neq k \tag{10b}$$

$$z_i \geq H \cdot (\theta_{i,k} - 1) + z_k^{'} \qquad \forall i, k \in I | i \neq k \tag{10c}$$

$$\beta \cdot x_i + W \cdot (\theta_{i,k} - 1) \leq \beta \cdot x_k + (1 - \pi_{i,k,0}) \cdot W \qquad \forall i, k \in I | i \neq k \tag{10d}$$

$$\beta \cdot x_k + W \cdot (\pi_{i,k,0} - 1) \leq x_i^{'} + (1 - \theta_{i,k}) \cdot W \qquad \forall i, k \in I | i \neq k \tag{10e}$$

$$\beta \cdot x_i + W \cdot (\theta_{i,k} - 1) \leq x_k^{'} + (1 - \pi_{i,k,1}) \cdot W \qquad \forall i, k \in I | i \neq k \tag{10f}$$

$$x_k^{'} + W \cdot (\pi_{i,k,1} - 1) \leq x_i^{'} + (1 - \theta_{i,k}) \cdot W \qquad \forall i, k \in I | i \neq k \tag{10g}$$

$$\beta \cdot y_i + L \cdot (\theta_{i,k} - 1) \leq \beta \cdot y_k + (1 - \pi_{i,k,2}) \cdot L \qquad \forall i, k \in I | i \neq k \tag{10h}$$

$$\beta \cdot y_k + L \cdot (\pi_{i,k,2} - 1) \leq y_i^{'} + (1 - \theta_{i,k}) \cdot L \qquad \forall i, k \in I | i \neq k \tag{10i}$$

$$\beta \cdot y_i + L \cdot (\theta_{i,k} - 1) \leq y_k^{'} + (1 - \pi_{i,k,3}) \cdot L \qquad \forall i, k \in I | i \neq k \tag{10j}$$

$$y_k^{'} + L \cdot (\pi_{i,k,3} - 1) \leq y_i^{'} + (1 - \theta_{i,k}) \cdot L \qquad \forall i, k \in I | i \neq k \tag{10k}$$

$$\pi_{i,k,0} + \pi_{i,k,1} \geq \theta_{i,k} \qquad \forall i, k \in I | i \neq k \tag{10l}$$

$$\pi_{i,k,2} + \pi_{i,k,3} \geq \theta_{i,k} \qquad \forall i, k \in I | i \neq k \tag{10m}$$

$$\pi_{i,k,0} + \pi_{i,k,1} \leq \theta_{i,k} \cdot 2 \qquad \forall i, k \in I | i \neq k \tag{10n}$$

$$\pi_{i,k,2} + \pi_{i,k,3} \leq \theta_{i,k} \cdot 2 \qquad \forall i, k \in I | i \neq k \tag{10o}$$

$$\theta_{i,k} \leq S_{i,k} \qquad \forall i, k \in I | i \neq k \tag{10p}$$

$$\sum_{k \in I^+} \theta_{i,k} \geq \sum_{j \in J} C_{i,j} \qquad \forall i \in I \tag{10q}$$

$$\theta_{i,k} = 0, \pi_{i,k,l} = 0 \qquad \forall i, k \in I | i = k, \forall l \in 0, 1, 2, 3 \tag{10r}$$

We are dealing with items with an arbitrary centre of mass and this condition affects both load-bearing and the bins' global CoM. Before proceeding further, we have to model how the centre of mass is affected by the item rotations and reflection. $\kappa^i = (\kappa_i^x, \kappa_i^y, \kappa_i^z)$ represents the actual centre of mass for the item $i \in I$, where the dimensions are relative to the front-bottom-left corner. We use $\upsilon_i^x, \upsilon_i^y$ to represent the CoM point after applying rotations and reflection. Note that CoM along the z-axis is not affected by rotation or reflection. However, if z-axis rotations (i.e. 6-way rotations) are considered, it is trivial to extend the model here to cover such a case.

In the group of constraints (11) we state the centre of mass considering reflection and rotation. (11a)-(11d) define the x coordinate, R_i is a decision variable that determines which group of equations must be satisfied. R_i is equal to 1 if there is reflection, 0 otherwise. The first group, (11a)-(11b) represent the case of no reflection, while (11c)-(11d) represent the case with reflection. The rotations follow the same schema adopted before in the constraints group (6).

$$\upsilon_i^x \leq \kappa_i^x \cdot \varphi_{0,i} + \kappa_i^y \cdot \varphi_{1,i} + R_i \cdot W \qquad \forall i \in I \tag{11a}$$

$$\upsilon_i^x \geq \kappa_i^x \cdot \varphi_{0,i} + \kappa_i^y \cdot \varphi_{1,i} - R_i \cdot W \qquad \forall i \in I \tag{11b}$$

$$\upsilon_i^x \leq x_i^{'} - (\kappa_i^x \cdot \varphi_{0,i} + \kappa_i^y \cdot \varphi_{1,i}) + (1 - R_i) \cdot W \qquad \forall i \in I \tag{11c}$$

$$\upsilon_i^x \geq x_i^{'} - (\kappa_i^x \cdot \varphi_{0,i} + \kappa_i^y \cdot \varphi_{1,i}) - (1 - R_i) \cdot W \qquad \forall i \in I \tag{11d}$$

$$\upsilon_i^y \leq \kappa_i^y \cdot \varphi_{0,i} + \kappa_i^x \cdot \varphi_{1,i} + R_i \cdot L \qquad \forall i \in I \tag{11e}$$

$$\upsilon_i^y \geq \kappa_i^y \cdot \varphi_{0,i} + \kappa_i^x \cdot \varphi_{1,i} - R_i \cdot L \qquad \forall i \in I \tag{11f}$$

$$\upsilon_i^y \leq y_i^{'} - (\kappa_i^y \cdot \varphi_{0,i} + \kappa_i^x \cdot \varphi_{1,i}) + (1 - R_i) \cdot L \qquad \forall i \in I \tag{11g}$$

$$\upsilon_i^y \geq y_i^{'} - (\kappa_i^y \cdot \varphi_{0,i} + \kappa_i^x \cdot \varphi_{1,i}) - (1 - R_i) \cdot L \qquad \forall i \in I \tag{11h}$$

Now we are ready to introduce the constraints that restrict the bins CoM. We define a pyramidal safe region in which the centre of mass has to lie. This region is defined by the vertex ϱ_j, and has a base radius ξ_j. A possible solution could be given by setting the vertex coordinates in the bin's centre, respectively $\varrho_j^z = \frac{1}{2} \cdot H_j$, $\varrho_j^x = \frac{1}{2} \cdot L_j$ and $\varrho_j^y = \frac{1}{2} \cdot W_j$.

$\tau_{i,j}^x$, $\tau_{i,j}^y$, $\tau_{i,j}^z$ represent the coordinates of the CoM of item i considering as coordinate reference system the origin of the bin j. Constraints (12a)-(12c) represent the reference coordinate for υ_i^x, (12d)-(12f) for υ_i^y and (12g)-(12i) remap κ_i^z, because we do not have rotations over the z axis.

The next group of constraints defines the pyramidal region. The linearization exploits the style in Paquay et al. (2016), defining the feasible region for each bin. The principal differences are the region shape and the introduction of the asymmetry in the items CoM. (12j) and (12k) define the pyramid limits along the x axis, (12l) and (12m) for the y axis, while (12n) defines the constraint on the actual height of the CoM that must fall under the region previously defined. ξ_j represents the radius in the base of the pyramid for bin j. We pre-compute $\xi_j = \alpha \cdot L_j$ if $W_j > L_j$ or $\xi_j = \alpha \cdot W_j$ otherwise. α is a parameter to define the base of the pyramid, $\alpha \in (0, 1]$. By default, we suggest $\alpha = 0.8$ which means 80% of the lower dimension.

$$\tau^x_{i,j} \leq W_j \cdot C_{i,j} \qquad \forall i \in I, j \in J \tag{12a}$$

$$\tau^x_{i,j} \leq \upsilon^x_i + \beta \cdot x_i \qquad \forall i \in I, j \in J \tag{12b}$$

$$\tau^x_{i,j} \geq \upsilon^x_i + \beta \cdot x_i - W_j \cdot (1 - C_{i,j}) \qquad \forall i \in I, j \in J \tag{12c}$$

$$\tau^y_{i,j} \leq L_j \cdot C_{i,j} \qquad \forall i \in I, j \in J \tag{12d}$$

$$\tau^y_{i,j} \leq \upsilon^y_i + \beta \cdot y_i \qquad \forall i \in I, j \in J \tag{12e}$$

$$\tau^y_{i,j} \geq \upsilon^y_i + \beta \cdot y_i - L_j \cdot (1 - C_{i,j}) \qquad \forall i \in I, j \in J \tag{12f}$$

$$\tau^z_{i,j} \leq H_j \cdot C_{i,j} \qquad \forall i \in I, j \in J \tag{12g}$$

$$\tau^z_{i,j} \leq \kappa^z_i + z_i \qquad \forall i \in I, j \in J \tag{12h}$$

$$\tau^z_{i,j} \geq \kappa^z_i + z_i - H_j \cdot (1 - C_{i,j}) \qquad \forall i \in I, j \in J \tag{12i}$$

$$\sum_{i \in I} \tau^x_{i,j} \cdot \omega_i \leq \varrho^x_j \cdot (\sum_{p \in I} \omega_p \cdot C_{p,j}) + \frac{\varrho^z_j \cdot (\sum_{p \in I} \omega_p \cdot C_{p,j}) - \sum_{i \in I} \tau^z_{i,j} \cdot \omega_i}{\varrho^z_j} \cdot \xi_j \qquad \forall j \in J \tag{12j}$$

$$\sum_{i \in I} \tau^x_{i,j} \cdot \omega_i \geq \varrho^x_j \cdot (\sum_{p \in I} \omega_p \cdot C_{p,j}) - \frac{\varrho^z_j \cdot (\sum_{p \in I} \omega_p \cdot C_{p,j}) - \sum_{i \in I} \tau^z_{i,j} \cdot \omega_i}{\varrho^z_j} \cdot \xi_j \qquad \forall j \in J \tag{12k}$$

$$\sum_{i \in I} \tau^y_{i,j} \cdot \omega_i \leq \varrho^y_j \cdot (\sum_{p \in I} \omega_p \cdot C_{p,j}) + \frac{\varrho^z_j \cdot (\sum_{p \in I} \omega_p \cdot C_{p,j}) - \sum_{i \in I} \tau^z_{i,j} \cdot \omega_i}{\varrho^z_j} \cdot \xi_j \qquad \forall j \in J \tag{12l}$$

$$\sum_{i \in I} \tau^y_{i,j} \cdot \omega_i \geq \varrho^y_j \cdot (\sum_{p \in I} \omega_p \cdot C_{p,j}) - \frac{\varrho^z_j \cdot (\sum_{p \in I} \omega_p \cdot C_{p,j}) - \sum_{i \in I} \tau^z_{i,j} \cdot \omega_i}{\varrho^z_j} \cdot \xi_j \qquad \forall j \in J \tag{12m}$$

$$\sum_{i \in I} \tau^z_{i,j} \cdot \omega_i \leq \varrho^z_j \cdot \sum_{p \in I} \omega_p \cdot C_{p,j} \qquad \forall j \in J \tag{12n}$$

The last part of the model concerns load-bearing. We assume that every item i is not perfectly rigid, thus, can resist the pressure of no more than a specific load, Λ_i. We know the centre of mass, κ^i, but how the mass is distributed inside i remains unknown. The weight that an item is bearing is the sum of the items' weights inside the cuboid space over it. Constraints (13a) and (13b) restrict the z coordinate of items placed over item i. Constraints (13c)-(13f) limit the items having their centre of mass over item i, note that those equations follow the same style used for stability, exploiting the previously defined centre of mass. Constraint (13g) ensures that the total weight of items placed over item i does not exceed Λ_i.

$$z_i^{'} \leq z_k + (1 - \lambda_{i,k}) \cdot H \qquad \forall i,k \in I | i \neq k \tag{13a}$$

$$\lambda_{i,k} \cdot H + z_i^{'} \geq z_k + 1 \qquad \forall i,k \in I | i \neq k \tag{13b}$$

$$(\lambda_{i,k} - 1) \cdot W + \beta \cdot x_i \leq \sum_{j \in J} \tau_{k,j}^x \qquad \forall i,k \in I | i \neq k \tag{13c}$$

$$\sum_{j \in J} \tau_{k,j}^x \leq x_i^{'} + (1 - \lambda_{i,k}) \cdot W \qquad \forall i,k \in I | i \neq k \tag{13d}$$

$$(\lambda_{i,k} - 1) \cdot L + \beta \cdot y_i \leq \sum_{j \in J} \tau_{k,j}^y \qquad \forall i,k \in I | i \neq k \tag{13e}$$

$$\sum_{j \in J} \tau_{k,j}^y \leq y_i^{'} + (1 - \lambda_{i,k}) \cdot L \qquad \forall i,k \in I | i \neq k \tag{13f}$$

$$\sum_{k \in I} \lambda_{i,k} \cdot \omega_k \leq \Lambda_i \qquad \forall i \in I \tag{13g}$$

$$x_i, y_i, z_i, x_i^{'}, y_i^{'}, z_i^{'}, \tau_{i,j}^x, \tau_{i,j}^y, \tau_{i,j}^z, \upsilon_i^x, \upsilon_i^y \in \mathbb{N}, \qquad \forall i \in I, j \in J \tag{14a}$$

$$S_{i,k}, Z_j, C_{i,j}, x_{i,k}^p, y_{i,k}^p, z_{i,k}^p, \theta_{i,m}, R_i, \lambda_{i,k} \in \{0,1\}, \qquad \forall i,k \in I, j \in J, m \in I \bigcup \{m+1\} \tag{14b}$$

$$\pi_{i,k,p}, \varphi_{o,i} \in \{0,1\}, \qquad \forall i,k \in I, p \in \{0,1,2,3\}, o \in O \tag{14c}$$

3.2. Model splitting and sizes

From the previous model we extract three submodels called V0, V1, V2, with each including a subset of the constraints. Let n be the number of items and m be the number of bins.

The submodel V0 contains constraints (1) - (10r) and considers only stability. The size of the model is $9n^2 + nm + m + 9n$ variables and $n^2m^2 - nm^2 - n^2m + 2nm + 25n^2 + 3m - 15n$ constraints.

The submodel V1 includes constraints (1) - (12n) and considers stability and the distribution of the CoM. The size of the model is $9n^2 + 4nm + m + 12n$ variables and $n^2m^2 - nm^2 - n^2m + 11nm + 25n^2 + 8m - 7n$ constraints.

The submodel V2 includes constraints (1) - (13g) and considers stability, distribution of the CoM and load bearing. The size of the model is $10n^2 + 4nm + m + 12n$ variables and $n^2m^2 - nm^2 - n^2m + 11nm + 31n^2 + 8m - 12n$ constraints.

Table 6: Size of models

	constraints	variables
V0	$n^2m^2 - nm^2 - n^2m + 2nm + 25n^2 + 3m - 15n$	$9n^2 + nm + m + 9n$
V1-V0	$9nm + 5m + 8n$	$3nm + 3n$
V2-V0	$6n^2 + 9nm + 5m + 3n$	$n^2 + 3n + 3nm$
V2-V1	$6n^2 - 6n + n$	n^2

4. Heuristic Weight First Best Fit(WFBF)

In this section we show a simple deterministic heuristic that we name Weight First Best Fit(WFBF). WFBF is an on-line heuristic that fills the available bins with one item at a time, sequentially, until the available bins or the available items are exhausted. WFBF includes a multi-start procedure in case the available bins have the same priority but different dimensions. The assumptions of the heuristic is that all the bins have priority $p_j < 1$ or $p_j \geq 1$. The procedure randomises the order of the bins in each unique priority set and tests if there are better solutions. WFBF belongs to the category of constructive methods (Zhu et al., 2012) and it is detailed in Algorithms 1 and 2.

WFBF assumes, without any loss of generality, that it is given as an input the list of bins (*bins*), the list of items (*items*) and the maximum repetitions (*maxLimit*). The list of items is sorted by weight in descending order, discarding the items that do not fit in any available bin. The sorting of the list of items may even be internalised in the algorithm as can be trivially calculated by a few lines of code.

The list of bins is sorted with the function $randomisedOrderWithPriority(bins)$. $randomisedOrderWithPriority$ takes as input the list of bins, and returns the list of bins sorted by "randomised priority ascending", if $p_j < 1$, or in "randomised priority descending", if $p_j \geq 1$. The steps of $randomisedOrderWithPriority$ are the following: 1) partition the list of bins in classes of bins with the same priority 2) for each class, shuffle the list of bins in the class 3) flatten the classes in priority order. 4) return the flattened array.

$randomisedOrderWithPriority$ is necessary for the multi-starting procedure to explore different combinations when there are bins in the same class that have the same priority but respectively different shapes. If a class is composed by bins with the same shapes then there is no benefit in shuffling. An example of possible $randomisedOrderWithPriority$ outputs are shown in Fig 6, given as input any array a,b or c, the output can be any array between a,b or c.

$isRandomizable(bins)$ evaluates if the solution may benefit from $randomisedOrderWithPriority$. The procedure is simple, it returns true if for at least one class (in the list of classes of bins with the same priority) there are

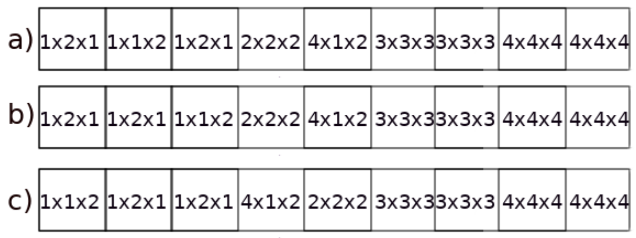

Fig. 6: Example of $randomisedOrderWithPriority$ outputs. The bins are sorted by priority, inside each priority class, the order is random. If the bins' shapes are exactly the same, there will not be any difference between every output.

at least two elements and their shapes are different, false otherwise. This function is used to avoid unnecessary repetitions in the case where the list of bins has every priority class with one element only.

$isBetterSolution(solution1, solution2)$ returns true if $solution1$ opens fewer bins then $solution2$. If $solution2$ is empty or not set, then returns true. The function returns false otherwise.

Let $solution$ be the set of final solutions. For each non-empty bin j, let $solution_j$ be the set of items assigned to the bin j with their configuration. The configuration for each item is its position p, its rotation o and a flag r to determine if the reflection is applied.

If the bin list $isRandomizable$ then repeat for the $maxLimit$ the following filling procedure.

The algorithm fills the next empty bin j if there are still available items. A procedure enumerates all the possible positions in the space of the bin j. This procedure is trivial, and the cardinality of $position_j$ is $\frac{H_j \cdot W_j \cdot L_j}{\beta}$. For each available item i, in every combination of possible positions p, rotations o and reflection r, the algorithm computes a ranking function (Eq:15).

The ranking function ranks the quality of the item configuration, the smaller the rank, the better the configuration. The function promotes configurations that are near the floor of the bin and near the vertical axis placed in the vertex of the bin's feasible centre of mass space. The variables z_i and $\tau_i^{x,y}$ are explained in Table 5, respectively. They are the coordinate z of item i and the position of the CoM of item i in the plane orthogonal to the bin's floor. $\theta_j^{x,y}$ is listed in Table 4 and is the position of the vertex of the feasible CoM region of the bin j.

$$rank \Leftarrow (z_i + \left\| \tau_i^{x,y} - \theta_j^{x,y} \right\|_2)^2 \tag{15}$$

WFBF makes use of three functions, named $isNotOverlapping(solution_j, item_{i,p,o,r})$, $isLoadBearingValid(solution_j, item_{i,p,o,r})$ and $isCDMValid(solution_j, item_{i,p,o,r})$. Their names describe their function, and can be implemented in a straightforward way from the model's constraints.

$isNotOverlapping(solution_j, item_{i,p,o,r})$ tests if the configuration of item i does not overlap any other item already placed inside the bin j

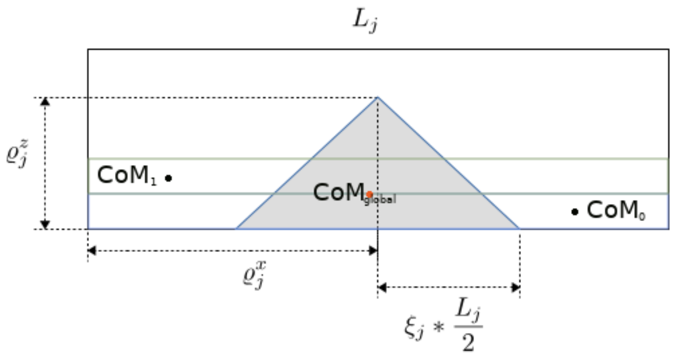

Fig. 7: An example that WFBF cannot solve, despite there being a feasible solution. The feasible solution is represented in the figure, with CoM_{global} in the feasible region.

$isLoadBearingValid(solution_j, item_{i,p,o,r})$ tests if adding the item i with the configuration p, o, r breaks some already placed items inside bin j under the CoM of item i.

$isCDMValid(solution_j, item_{i,p,o,r})$ tests if the global CoM in bin j is inside the admissible region.

The *best* rank is initialized to infinite. If the *rank* of a new item is strictly better than the rank of the current *best* (first best fit) and the placement of this new item does not violate the filling constraints, update the best item with the new item.

After all configurations have been tested, if there is no best (i.e. there is no feasible configuration), the item i is placed in a list of discarded items, *discarded*. Items in this list will be considered for the next bin. If a best configuration is found, the $item_{i,p,o,r}$ is appended into $solution_j$, and the variable $feasiblePositions$ is updated by removing the ones that now are overlapping with the $item_{i,p,o,r}$. The last procedure helps to speed up the algorithm.

Since the items are initially sorted by weight in descending order, WFBF may not guarantee a global optimal solution. WFBF computational complexity in the worst case is $O(m \cdot n \cdot q)$ where m is the number of bins, n is the number of items and q is the number of possible positions. The worst case scenario belongs to the class of cases that WFBF cannot handle: consider one bin j with dimensions W_j,L_j,H_j, and at least two items i,k with dimensions $w_k = w_i = W_j$,$l_k = l_i = L_j$,h_k, h_i. If the CoM of the items i,k are outside the feasible CoM region of the bin j, WFBF will not be able to find any feasible packing solution, while, in fact there exists one. The example is shown in Fig 7, the feasible solution is represented in the figure, with CoM_{global} in the feasible region.

5. Dataset

In most studies in the literature, given a fixed dimension for the bin, items dimensions are generated randomly. For example, Junqueira et al. (2012) considered a random dataset made from cubic bins of different size and items

Algorithm 1 Heuristic WFBF, part 1

```
Require: $bins$
Require: $maxLimit$
Require: $items$ sorted by weight in descending order, removing the items that do not fit in any available bin
1: $bestSolution \Leftarrow \emptyset$
2: $repeatLimit \Leftarrow 1$
3: $repeat \Leftarrow 0$
4: if $isRandomizable(bins)$ then
5:    $repeatLimit \Leftarrow maxLimit$
6: end if
7: for $repeat < repeatLimit$ do
8:    $solution \Leftarrow \emptyset$
9:    $itemsleft \Leftarrow items$
10:   $bins \Leftarrow randomisedOrderWithPriority(bins)$
11:   while $j \in bins$ do
12:      $discarded \Leftarrow \emptyset$
13:      if $itemsleft \neq \emptyset$ then
14:         $solution_j \Leftarrow \emptyset$
15:         $positions_j \Leftarrow$ enumerate all the possible positions available inside the bin $j$
16:         for $i \in itemsleft$ do
17:            $best \Leftarrow \infty$
18:            $winning \Leftarrow \emptyset$
19:            for all $item_{i,p,o,r}$, $p \in positions_j$, $o \in O$, $r \in \{0,1\}$ reflection do
20:               $rank \Leftarrow (z_i + \left\| \tau_i^{x,y} - \theta_j^{x,y} \right\|_2)^2$
21:               if $rank < best$ AND isNotOverlapping($solution_j$,$item_{i,p,o,r}$) AND isLoadBearingValid($solution_j$,$item_{i,p,o,r}$) AND isCDMValid($solution_j$,$item_{i,p,o,r}$) then
22:                  $best \Leftarrow rank$
23:                  $winning \Leftarrow item_{i,p,o,r}$
24:               end if
25:            end for
```

that were ranging from 10% to 50% or 25% to 75% of the bin size. This procedure, although widely used, could lead to results that do not entirely match with the characteristics of real-world scenarios. A dataset would be ideal if it is useful for both academic and industrial fields. Thus, we decided to exploit the work made by Landschützer et al. (2015) for the Physical Internet challenge (Montreuil, 2012). We built our test instances by randomly picking

Algorithm 2 Heuristic WFBF, part 2

```
26:            if $winning \neq \emptyset$ then
27:                $feasiblePositions \Leftarrow$ remove the overlapping winning positions from $feasiblePositions$
28:                $solution_j \Leftarrow solution_j \bigcup \{item_{i,p,o,r}\}$
29:            else
30:                $discarded \Leftarrow discarded \bigcup \{i\}$
31:            end if
32:        end for
33:        $itemsleft \Leftarrow discarded$
34:      end if
35:    end while
36:    $repeat \Leftarrow repeat + 1$
37:    if $isBetterSolution(solution, bestSolution)$ then
38:      $bestSolution \Leftarrow solution$
39:    end if
40:  end for
41:  return $bestSolution$
```

containers and items from the types defined in the full theoretical set of Physical Internet containers. The set consists of 24 types of bins and 440 types of items. Note that since the PI modular containers are designed to be compatible with existing ISO units, the tested instances should be applicable for current scenarios where ISO loading units (e.g. 40, 20, 10 ft ISO containers) are also used.

The other features, such as load-bearing for items and weight limits for the bins, have been selected to be as realistic as possible. The bin's weight limit is proportional to its volume and the weight limits for actual 40' inter-modal containers (Eq: 16), with the exception of the 20' container which follows the actual standard. The same philosophy has been adopted for defining the maximal weight for the items (Eq: 17).

$$\Omega_j = \frac{W_j \cdot L_j \cdot H_j}{40'volume} \cdot 40'weight_{Limit} \tag{16}$$

$$\Omega_i = \frac{w_i \cdot l_i \cdot h_i}{maxitemvolume} \cdot \frac{20'weight_{Limit} \cdot maxitemvolume}{20'volume} \tag{17}$$

The items' centre of mass has been generated in a random point under $\frac{3}{5}$ of the height and between 20% and 80% of the other dimensions.

The items' load-bearing in real-world boxes varies considerably depending on the material used and construction geometries. Thus, we opt for simplicity and set the load bearing to be 3 times the maximal weight of the item. This is a parameter that can be changed by users if necessary.

$$\Lambda_i = 3 \cdot \Omega_i \tag{18}$$

The item weight has been generated randomly with a uniform distribution between 30% and 100% of the items' maximal weight.

Every experiment is composed of n bins and m items. The procedure for generating a test instance is to firstly randomly pick n bins and then randomly pick m items such that, for every item there exists at least one bin that can fit it. There is no loss of generality because this procedure is equivalent to filtering unfitting items before the optimization task. All the instances follow the same item distribution. The set of available types of items is ordered by increasing volume in such a way that every instance is made of 70% of items after the median (higher volume), and 30% before (lower volume).

Moreover, we introduce some new metrics to better evaluate the results of the experiments. The first one is inspired by the eccentricity of conics, which will measure the distance of an item/bin from being a cube. Thus, given l_i, w_i, h_i as the dimensions of the item/bin i, the eccentricity of i is defined by Eq: 19.

$$\varepsilon_i = \max\{\left|\frac{l_i - w_i}{l_i + w_i}\right|, \left|\frac{h_i - w_i}{h_i + w_i}\right|, \left|\frac{l_i - h_i}{l_i + h_i}\right|\} \tag{19}$$

The second one, α_i in Eq:20, measures the asymmetry of the centre of mass. α_i is defined as the Euclidean distance from the actual centre of mass to the centroid (geometric centre) C^i of the item.

$$\alpha_i = \|C^i, CA^i\|_2 \tag{20}$$

The third one is the relative volume, Eq: 21. This metric is calculated as the volume of item i divided by the volume of the bin j.

$$VolRel_{i,j} = \frac{h_i \cdot w_i \cdot l_i}{H_j \cdot W_j \cdot L_j} \tag{21}$$

6. Experiments and discussion

The first aim of the experiments is to test the model behaviour by varying the number of items and bins. We also investigate the impact of the constraints on the difficulty of finding a solution. Each experiment terminates with a final script that validates the solution found. We tested the three models listed in Section 3.2. Briefly, V0 considers only stability, V1 considers stability and CoM distribution and V2 considers stability, CoM distribution, and load-bearing. The second aim is to have a comparison metric to test the goodness of the proposed heuristic. Having the results from the model we can know how far the heuristic is from finding the optimal solution.

All versions have been implemented in OPL and solved using IBM CPLEX® 12.7. The heuristic has been implemented in Python 3.6. The machine used to perform the experiments was an Intel® Core™ i7-4790 CPU @ 3.60GHz, 16GB RAM, with Windows 7 Enterprise 64-bit. It is worth noting that, as mentioned before, in all the following experiments, and without affecting the generality of the model, we use a priority equal to the inverse of the bin's volume (2), which is equivalent to giving filling priority to smaller bins.

As reported by many authors, the complexity of the problem depends obviously on the number of items, the number of bins and their dimensions. The first two can be explained by looking at how the number of variables and the number of constraints grows according to the number of items and bins, as shown in Table 6. The item/bin dimensions involved affect the number of feasible positions in the solution space. Thus, considering large spaces increases the difficulty.

In each experiment we reported the gap calculated by the solver, the formula is reported in Eq: 22. We applied the same formula to calculate the gap in the heuristic results.

$$gap = \frac{|bestbound - bestinteger|}{\frac{1}{10^{10}} + |bestinteger|} \tag{22}$$

In the first group of experiments, we focus on packing 10 random items in the smallest bin, with dimensions $120 \times 120 \times 120$. In Table 7 we report the average features values of the solutions found for all the 1,000 instances solved with V0, V1. and V2. The *dataset* column represents the features of the dataset. $\mu_{\bullet}$ indicates average while $\sigma^2_{\bullet}$ is the variance. ω indicates the weights, LB is the load bearing limit, $VolRel$ is the relative volume $(\frac{volumeitem}{volumebin})$, ε is the previously defined eccentricity, α is the asymmetry, *objective* value is the result of minimizing the wasted space (Eq. 1), *elapsedtime* is the time elapsed from the starting point before the pre-solving phase to the solution or the time limit. V0 is not affected by the weight but it packs fewer items than V1 because it chooses items with bigger volume than V1 (μ_{VolRel}). The degradation of the objective between V0 and V1 is 0.12% and the time to achieve a solution increases by about 30%. V2, instead, is about 15 times more complex to solve than V1. V2 chooses items with higher load-bearing limit (μ_{LB}) which will also have higher volume (μ_{VolRel}) and be slightly less eccentric than V1 but with higher μ_{α}. We hypothesize that the complexity is due to the combination of the following factors: higher volume items imply a higher centre of mass along the H axis and this brings a lower feasible solution space[1]. Similarly, the higher the eccentricity, the higher the α. Those factors combined with the restricted base will force the solver to explore more nodes before finding a solution.

The second group of experiments has been made to test the behaviour of the model using only one bin. We ran

[1]The higher the height in the pyramid, the less space available.

Table 7: 10 item 1 bin, average values over 1,000 random instances, time in milli-seconds

feature	V0	V1	V2	dataset
available items	10	10	10	10
packed items	9.813	9.817	9.386	-
μ_{ω}	23.312	23.010	24.500	22.657
σ^2_{ω}	1167.43	1170.52	1265.41	1111.56
μ_{LB}	107.38	105.69	112.78	103.97
σ^2_{LB}	21698.31	21797.74	23466.60	20633.55
μ_{VolRel}	0.05005	0.04926	0.05259	0.04843
σ^2_{VolRel}	0.00479	0.00481	0.00519	0.00455
μ_{ε}	0.5587	0.5589	0.5557	0.5604
σ^2_{ε}	0.0357	0.0358	0.0355	0.0359
μ_{α}	11.884	11.877	12.120	11.882
σ^2_{α}	81.871	81.919	82.789	82.305
$\mu_{objective}$	0.5316	0.5328	0.5407	-
$\mu_{elapsedtime}$	330.16	432.86	7019.56	-

experiments with instances having 18 to 200 items, each experiment had a time limit of 60 minutes. In Table 8 and Table 9 we report the results, with time and objective value, for V0, V1 and V2. Trying to solve more than 350 items made our computer run out of memory. With 130 items or more, the solver spent most of the time in the pre-solving phase and very few iterations in the solving one. In almost every case, V2 final objective is about 100% worse than V1. In some instances, for example, with 100 items, we can see that V2 obtains a better objective than V1 within the time limit. This may be due to a different choice in the nodes to explore.

The third group of experiments aims to show the behaviour of the model with multiple bins. We tested with 2, 3 and 5 bins and number of items ranging from 70 to 200. Table 10 shows the computational time and the objective value for V0, V1, and V2 according to the number of items and the number of bins. Compared to the results for one bin, the overall performance degraded dramatically when the number of bins is two or more. Due to the increased number of bins, for instances with more than 70 items, the solver spent most of its time on the pre-solving phase and very few iterations on the solving phase. The fourth group of experiments compare the heuristic WFBF with the results of V2. The comparison has been made using the three groups of problem instances used previously. Table 11 shows the experiments using 1 bin, from 18 to 90 items. WFBF is capable of finding on some instances the optimal solution, while in general it achieved better results using much less time.

Table 12 shows experiments using 1 bin, from 100 to 200 items. As expected, WFBF significantly outperformed

Table 8: Experiments using 1 bin, from 18 to 90 items. Time is measured in seconds (* is when the solver has hit the time limit of 60 minutes).

instance		V 0			V 1			V 2		
item	bin	time	objective	gap%	time	objective	gap%	time	objective	gap%
18	1	9.84	0.3090	0	14.38	0.3090	0	3600*	0.4407	29.87
19	1	2.72	0.5098	0	15.89	0.5098	0	3600*	0.5409	0.62
20	1	1.34	0.6912	0	1.404	0.6912	0	3.681	0.6912	0
21	1	9.06	0.2935	0	22.62	0.2935	0	3600*	0.3754	21.80
22	1	4.74	0.8211	0	5.61	0.8211	0	15.965	0.8211	0
23	1	9.12	0.3325	0	27.22	0.3325	0	3600*	0.5452	39.02
24	1	5.662	0.8965	0	0.936	0.8965	0	65.542	0.8965	0
25	1	3600*	0.0394	100	3600*	0.0394	100	3600*	0.0668	100
26	1	2.574	0.8664	0	6.16	0.8664	0	42.933	0.8664	0
27	1	3600*	0.1172	100	3600*	0.1538	100	3600*	0.3513	99.92
28	1	9.375	0.6917	0	34.42	0.6917	0	3600*	0.7218	4.17
29	1	9.475	0.7164	0	16.03	0.7164	0	256	0.7164	0
30	1	10.202	0.7842	0	11.31	0.7842	0	84.68	0.7842	0
35	1	45.831	0.5476	0	87.42	0.5476	0	3600*	0.6194	11.59
40	1	59.9	0.719	0	111.8	0.719	0	3600*	0.7545	4.70
45	1	3600*	0.2271	1.96	3600*	0.2937	24.20	3600*	0.3412	34.74
50	1	1813.27	0.4771	0	1234.49	0.4771	0	3600*	0.5462	12.65
55	1	3600*	0.0312	100	3600*	0.0430	100	3600*	0.3027	100
60	1	3600*	0.5633	29.75	3600*	0.5682	30.36	3600*	0.6117	35.31
65	1	1142.86	0.5414	0	3236.7	0.5414	0	3600*	0.6615	22.16
70	1	3600*	0.2394	100	3600*	0.2310	100	3600*	0.3841	100
80	1	3600*	0.0622	100	3600*	0.0902	100	3600*	0.3636	100
90	1	3600*	0.325	100	3600*	0.3364	100	3600*	0.4847	100

the solver, the filling rates are in average greater than 90%, in some cases achieve 100%. This is reasonable since having more items also improves the probability of having items that fit together.

Table 13 shows the experiments with multiple bins. The results show that the heuristic clearly outperforms the exact model both in terms of objective value and computational time. An increase in the number of items

Table 9: Experiments using 1 bin, from 100 to 200 items. Time is measured in seconds(* is when the solver has hit the time limit of 60 minutes).

instance		V 0			V 1			V 2		
item	bin	time	objective	gap%	time	objective	gap%	time	objective	gap%
100	1	3600*	0.1833	100	3600*	0.6	100	3600*	0.5666	100
110	1	3600*	0.7291	100	3600*	0.4265	100	3600*	0.5759	100
120	1	3600*	0.0962	100	3600*	0.3555	100	3600*	0.3333	100
130	1	3600*	0.3083	100	3600*	0.3412	100	3600*	0.4458	100
140	1	3600*	0.9111	100	3600*	0.6495	100	3600*	0.5937	100
150	1	21.45	0	0	3600*	0.2305	100	3600*	0.2777	100
160	1	3600*	0.8411	100	3600*	0.5533	100	3600*	0.5650	100
170	1	28.06	0	0	3600*	0.0444	100	3600*	0.1666	100
180	1	3600*	0.3611	100	3600*	0.7777	100	3600*	0.4049	100
190	1	3600*	0.8333	100	3600*	0.8266	100	3600*	0.9333	100
200	1	3600*	0.8819	100	3600*	0.5944	100	3600*	0.6208	100

and bins does increase the computational time and the gap% of the heuristic in certain cases, but this increase appears to be linear. This demonstrates the capability of the heuristic in dealing with larger instances.

The advantage of the heuristic, as demonstrated above, is the fast computational time and better quality than what the exact model can offer within the given time limit. The heuristic does however have its limitations due to its sequential filling procedure (one bin at a time and giving priority to smaller volume bins). According to this procedure, the first bin will be the one with lowest volume. It is filled having all the items available. The next bin, will have at least the same volume, however can only be filled using the discarded items from the previous bins. These choices have the drawback that every discarded item is going to be placed in a bin which will have at least the same volume as the previous one. Since we are assuming to allocate every item, it may lead to under-filling the last bin, since it is more likely to be also the one with the biggest volume. As an example, considering the experiment with 5 bins and 160 items, the first bin is filled with 4 items and has a 100% filling rate, the second bin is filled with 66 items at a filling rate of 92.44%, the third bin is filled with 66 items at a filling rate of 17% and the fourth one is filled with 24 items at a filling rate of 15%. The volume of the fourth bin is double the volume of the third bin. Other examples of this behaviour are reported in the next group of experiments.

The last group of experiments is to measure the costs/benefits of using a randomised order to discover new configurations. The instances in these experiments consider multiple bins such that for each distinct volume there

Table 10: Experiments using multiple bins. * time limit exceeded. The objective function is minimising Eq:1. We report the value of the objective and the number of bins that has been opened(in parentheses). In the last cases (5 bins, from 130 to 200 items), the solver was not able to finish the pre-solving phase before the time limit.

instance		V 0			V 1			V 2		
item	bin	time	objective (used bins)	gap%	time	objective (used bins)	gap%	time	objective (used bins)	gap%
70	2	3600*	1.1022(2)	18.23	3600*	1.3634(1)	30.24	3600*	1.4066(1)	32.39
100	2	3600*	1.8705(2)	31.24	3600*	1.5491(2)	16.97	3600*	1.5696(2)	18.06
130	2	3600*	1.8006(2)	72.33	3600*	1.9791(1)	74.83	3600*	1.9791(1)	74.81
160	2	3600*	1.8888(1)	55.48	3600*	1.9305(1)	56.44	3600*	1.7489(1)	51.91
200	2	3600*	1.5(2)	254.30	3600*	1.2222(2)	289.36	3600*	1.3333(2)	273.58
70	3	3600*	2.2336(3)	9.84	3600*	2.4269(1)	15.19	3600*	2.4486(1)	15.94
100	3	3600*	2.3031(1)	49.45	3600*	2.2376(1)	47.97	3600*	2.2302(1)	47.79
130	3	3600*	2.5999(3)	76.95	3600*	2.6311(2)	70.98	3600*	2.6930(1)	71.64
160	3	3600*	2.6666(1)	97.50	3600*	2.4763(3)	97.31	3600*	2.6969(1)	97.53
200	3	3600*	2.8333(2)	61.09	3600*	2.8353(2)	61.12	3600*	3(0)	91.69
70	5	3600*	3.8309(5)	39.09	3600*	4.4104(2)	47.09	3600*	4.3152(3)	45.92
100	5	3600*	4.5870(2)	28.47	3600*	4.2220(2)	22.28	3600*	4.2713(2)	23.18
130	5	3600*	5(0)	63.45	3600*	4.565(3)	60.06	3600*	4.6055(2)	60.41
160	5	3600*	5(0)	86.98	3600*	5(0)	85.67	3600*	5(0)	85.67
200	5	3600*	5(0)	81.36	3600*	5(0)	77.48	3600*	4.6811(2)	75.84

are at least two bins with different dimensions. We compared the contribution of $randomisedOrderWithPriority$ to a simple sorting by volume ascending. The *repeatLimit* is set at 10 iterations for each experiment, in both configurations.

Table 14 and Table 15 compare the behaviour of the heuristic using simple sorting to that of the heuristic using $randomisedOrderWithPriority$.

In these tables we show also the details for each bin filled. The *opening sequence* is the progression number of the container that has been filled. It is useful to highlight cases where a container has been skipped because there were no fitting item. *items packed* is the number of items that has been allocated inside the bin. *filling rate* is the percentage of the bin's total volume filled by the items. For the majority of the instances CPLEX did not report a lower bound in the time limit of one hour, thus we could not report the gap. In some instances, i.e. Table 15 5 bin with 160 to 400 items, WFBF skips a bin because there is no remaining fitting item: the height of the bin is lower than the height of the remaining items. For each instance the objective is highlighted in bold if there is a clear winner between *randomisedOrderWithPriority* and simple sorting, while the cases where the objectives are equal or worse are not highlighted. *randomisedOrderWithPriority* was able to improve the solutions in two cases. Since the simple ordering case is included in the possibles output, we can state that *randomisedOrderWithPriority* is better than having a simple ordering, and the benefit is dependent on the execution of a sufficient number of repetitions.

7. Conclusions

This research proposes a mixed-integer linear model that solves the Multiple Heterogeneous Knapsack Problem, while minimising the wasted space of the higher priority bins, and considering the following constraints: non overlapping, boundaries and positioning (both constrained and free), rotations (around z-axis), orthogonal displacement, weight limit, static stability, weight distribution in a pyramidal region and load bearing considering items' arbitrary centre of mass.

We focused our experiments on a case study of minimizing the wasted space while prioritizing smaller bins over bigger ones, but the model is generic and thus can be applied to case with other priorities.

This model can obtain routinely mathematically proved optimal solutions in small sized problems. For medium sized instances it can only achieve sub-optimal solutions. Despite that, it remains a useful tool for studying the problem. It provides a benchmark to compare the quality of the solutions found by heuristics and approximations models.

We studied the trade-off of adding more constraints to make the problem more realistic and the complexity of finding a solution. The model has been derived in submodels V0 (only stability), V1 (stability and centre of mass distribution), V2 (stability, centre of mass distribution and load-bearing). The experiments showed that, even with very small instances, the computational time for finding the optimal solution considering the load bearing constraint was 16.2 times longer than without considering it.

We introduced new metrics that aim to facilitate the comparison and analysis of different knapsack instances. An effort in this direction is important because faster algorithms are often specialized algorithms that only efficiently handle particular cases of the problem. Often decision-makers are not operational research specialists, so having such metrics improves the export of theoretical results to the industry.

We designed a simple deterministic heuristic to deal with large-scale instances in reasonable computational time and memory. Weight First Best Fit is an on-line heuristic that fills the available bins one at a time, sequentially, until the available bins or the available items are exhausted. WFBF significantly outperforms the exact model in almost all instances. We also highlighted some limits of the algorithm.

Table 11: Heuristic compared to V2. Experiments using 1 bin, from 18 to 90 items. Time is measured in seconds (* is when the solver has hit the time limit of 60 minutes).

instance		Heuristic			V 2		
item	bin	time	objective	gap%	time	objective	gap%
18	1	4.1864	**0.3090**	0.02	3600*	0.4407	29.87
19	1	8.0681	0.7015	23.37	3600*	**0.5409**	0.62
20	1	13.3015	**0.6912**	0	3.681	**0.6912**	0
21	1	4.5546	**0.3269**	10.20	3600*	0.3754	21.80
22	1	27.0961	**0.8211**	0	15.965	**0.8211**	0
23	1	6.2495	**0.4075**	18.41	3600*	0.5452	39.02
24	1	31.3101	**0.8965**	0	65.542	**0.8965**	0
25	1	1.3735	**0.0622**	100	3600*	0.0668	100
26	1	30.9466	**0.8664**	0	42.933	**0.8664**	0
27	1	2.5830	**0.235**	99.88	3600*	0.3513	99.92
28	1	19.0795	**0.69173**	0	3600*	0.7218	4.17
29	1	19.7695	**0.7164**	0	256	**0.7164**	0
30	1	36.5915	**0.7842**	0	84.68	**0.7842**	0
35	1	31.4864	**0.54769**	0.01	3600*	0.6194	11.59
40	1	43.3107	**0.7190**	0.01	3600*	0.7545	4.70
45	1	19.7262	**0.2226**	0.03	3600*	0.3412	34.74
50	1	35.9081	**0.4825**	1.12	3600*	0.5462	12.65
55	1	7.0416	**0.1032**	100	3600*	0.3027	100
60	1	45.3121	**0.3957**	0	3600*	0.6117	35.31
65	1	54.0963	**0.5414**	4.89	3600*	0.6615	22.16
70	1	10.9763	**0.14807**	100	3600*	0.3841	100
80	1	2.3786	**0.0094**	100	3600*	0.3636	100
90	1	15.6290	**0.0474**	100	3600*	0.4847	100

We have also added a top level search that tests different orderings of the bins. The search is useful when there are bins with the same priority and different dimensions.

For future work, we will analyse the behaviour of the model and heuristic using different combinations of priorities. Further research is necessary to design new algorithms to overcome the limits of the proposed heuristic.

Table 12: Heuristic compared to V2. Experiments using 1 bin, from 100 to 200 items. Time is measured in seconds(* is when the solver has hit the time limit of 60 minutes).

instance		Heuristic			V 2		
item	bin	time	objective	gap%	time	objective	gap%
100	1	8.8989	**0.0707**	100	3600*	0.5666	100
110	1	70.1088	**0.1455**	100	3600*	0.5759	100
120	1	3.3416	**0.0681**	100	3600*	0.3333	100
130	1	10.8469	**0.0263**	100	3600*	0.4458	100
140	1	38.3794	**0.0916**	100	3600*	0.5937	100
150	1	1.3438	**0.0**	100	3600*	0.2777	100
160	1	46.1770	**0.1043**	100	3600*	0.5650	100
170	1	0.3227	**0.0**	100	3600*	0.1666	100
180	1	2.7800	**0.0**	100	3600*	0.4049	100
190	1	12.4953	**0.0619**	100	3600*	0.9333	100
200	1	24.0775	**0.0428**	100	3600*	0.6208	100

Acknowledgement

This research is supported by an LJMU PhD scholarship, an NRCP-funded project no NRCP1617-6-125 managed by the Royal Academy of Engineering, and an RSSB-funded project no COF-INP-05. We would like to thank Dr Kathryn Gow for her proof reading. We would like to thank to the anonymous reviewers, and the Associate Editor that helped with their comments to significantly improve the paper.

Bibliography

References

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

Ceschia, S. and Schaerf, A. (2013). Local search for a multi-drop multi-container loading problem. *Journal of Heuristics*, pages 1–20.

Christensen, H. I., Khan, A., Pokutta, S., and Tetali, P. (2016). Multidimensional bin packing and other related problems: A survey.

Christensen, H. I., Khan, A., Pokutta, S., and Tetali, P. (2017). Approximation and online algorithms for multidimensional bin packing: A survey. *Computer Science Review*, 24:63–79.

Table 13: Heuristic compared to V2. Experiments using multiple bins. * time limit exceeded

instance		Heuristic			V 2		
item	bin	time	objective (used bins)	gap%	time	objective (used bins)	gap%
70	2	103.2236	**1.0228(2)**	7.02	3600*	1.4066(1)	32.39
100	2	197.5630	**1.2862(2)**	0.01	3600*	1.5696(2)	18.06
130	2	111.9377	**0.6241(2)**	20.12	3600*	1.9791(1)	74.81
160	2	181.0182	**0.9430(2)**	10.81	3600*	1.7489(1)	51.91
200	2	68.4242	**0.0751(2)**	96.76	3600*	1.3333(2)	273.58
70	3	210.6892	**2.1218(3)**	2.99	3600*	2.4486(1)	15.94
100	3	374.9229	**1.9325(3)**	39.75	3600*	2.2302(1)	47.79
130	3	262.8290	**1.6446(3)**	53.56	3600*	2.6930(1)	71.64
160	3	195.6397	**1.0881(3)**	93.88	3600*	2.6969(1)	97.53
200	3	455.4567	**1.6725(2)**	85.09	3600*	3(0)	91.69
70	5	25.8132	**3.4386(2)**	32.13	3600*	4.3152(3)	45.92
100	5	47.9942	**3.3819(2)**	2.98	3600*	4.2713(2)	23.18
130	5	212.4729	**3.0586(3)**	40.39	3600*	4.6055(2)	60.41
160	5	327.8276	**2.7441(4)**	73.89	3600*	5(0)	85.67
200	5	433.9891	**2.7896(3)**	59.46	3600*	4.6811(2)	75.84

Coffman Jr, E. G., Csirik, J., Galambos, G., Martello, S., and Vigo, D. (2013). Bin packing approximation algorithms: survey and classification. In *Handbook of combinatorial optimization*, pages 455–531. Springer.

Coffman Jr, E. G., Csirik, J., Johnson, D. S., and Woeginger, G. J. (2004). An introduction to bin packing. *Bibliographie. Siehe www. inf. u-szeged. hu/˜ csirik.*

Crainic, T. G., Perboli, G., and Tadei, R. (2012). Recent advances in multi-dimensional packing problems. In *New technologies-trends, innovations and research*. InTech.

de Queiroz, T. A. and Miyazawa, F. K. (2013). Two-dimensional strip packing problem with load balancing, load bearing and multi-drop constraints. *International Journal of Production Economics*, 145(2):511–530.

Delorme, M., Iori, M., and Martello, S. (2016). Bin packing and cutting stock problems: Mathematical models and exact algorithms. *European Journal of Operational Research*, 255(1):1–20.

Dyckhoff, H. (1990). A typology of cutting and packing problems. *European Journal of Operational Research*, 44(2):145–159.

Fekete, S. P. and Schepers, J. (2004). A combinatorial characterization of higher-dimensional orthogonal packing. *Mathematics of Operations Research*, 29(2):353–368.

Fekete, S. P., Schepers, J., and Van der Veen, J. C. (2007). An exact algorithm for higher-dimensional orthogonal packing. *Operations*

Table 14: Heuristic. Experiments using multiple bins; the bins for each experiment have the same volume and different dimensions. * time limit exceeded.

instance		Heuristic without $randomisedOrderWithPriority$						Heuristic with $randomisedOrderWithPriority$					
item	bin	time	objective (used bins)	gap%	opening sequence	items packed	filling rate%	time	objective (used bins)	gap%	opening sequence	items packed	filling rate%
70	2	48.56	1.3579(2)	34,78				75.13	**1.3097(2)**	13.29			
					1	68	64.20				1	60	52.44
					2	0	0.0				2	10	16.57
100	2	204.68	1.3453(2)	3.29				207.41	1.3453(2)	3.29			
					1	86	30.79				1	86	30.79
					2	13	34.66				2	13	34.66
130	2	108.71	1.0497(2)	76,50				140.88	**0.9792(2)**	0.0			
					1	107	88.04				1	107	76.14
					2	17	6.98				2	23	25.93
160	3	436.29	1.6169(2)	-				496.747	1.6169(2)	-			
					1	18	99.54				1	18	99.54
					2	142	38.75				2	142	38.75
200	3	463.34	**0.3824(1)**	-				528.69	1.3824(2)	-			
					1	200	61.75				1	196	60.03
											2	4	1.72

Research, 55(3):569–587.

Hifi, M., Kacem, I., Nègre, S., and Wu, L. (2010). A linear programming approach for the three-dimensional bin-packing problem. *Electronic Notes in Discrete Mathematics*, 36:993–1000.

Jin, Z., Ito, T., and Ohno, K. (2003). The three-dimensional bin packing problem and its practical algorithm. *JSME International Journal Series C Mechanical Systems, Machine Elements and Manufacturing*, 46(1):60–66.

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

Kellerer, H., Pferschy, U., and Pisinger, D. (2004). Introduction to np-completeness of knapsack problems. In *Knapsack problems*, pages 483–493. Springer.

Landschützer, C., Ehrentraut, F., and Jodin, D. (2015). Containers for the physical internet: requirements and engineering design related to fmcg logistics. *Logistics Research*, 8(1):1–22.

Lin, Y.-H., Meller, R. D., Ellis, K. P., Thomas, L. M., and Lombardi, B. J. (2014). A decomposition-based approach for the selection

Table 15: Heuristic. Experiments using multiple bins; the bins for each experiment have the same volume and different dimensions. * time limit exceeded.

instance		Heuristic without $randomisedOrderWithPriority$						Heuristic with $randomisedOrderWithPriority$					
item	bin	time	objective (used bins)	gap%	opening sequence	items packed	filling rate%	time	objective (used bins)	gap%	opening sequence	items packed	filling rate%
160	5	421.56	2.4695(3)	-				859.03	2.4695(3)	-			
					1	158	52.38				1	135	33.73
					3	2	0.66				3	25	19.31
200	5	506.95	2.4033(3)	-				885.82	2.4033(3)	-			
					1	199	59.33				1	177	39.77
					3	1	0.33				3	23	19.88
400	5	1475.59	**1.8067(3)**	-				1377.57	2.8067(4)	-			
					1	237	90.55				1	323	75.41
					2	134	14.46				2	76	40.58
					3	29	14.30				4	1	3.33
800	5	2721.43	0.61345(3)	-				2756.12	0.61345(3)	-			
					1	142	97.60				1	81	98.24
					2	424	83.50				2	403	87.53
					3	234	57.54				3	316	52.87

of standardized modular containers. *International Journal of Production Research*, 52(15):4660–4672.

Martello, S. and Toth, P. (1990). *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley & Sons, Inc., New York, NY, USA.

MODULUSHCA (2012). The modulushca project.

Montreuil, B. (2011). Toward a physical internet: meeting the global logistics sustainability grand challenge. *Logistics Research*, 3(2-3):71–87.

Montreuil, B. (2012). Physical internet manifesto.

Paquay, C., Schyns, M., and Limbourg, S. (2016). A mixed integer programming formulation for the three-dimensional bin packing problem deriving from an air cargo application. *International Transactions in Operational Research*, 23(1-2):187–213.

Sallez, Y., Pan, S., Montreuil, B., Berger, T., and Ballot, E. (2016). On the activeness of intelligent physical internet containers. *Computers in Industry*, 81:96–104.

Sweeney, P. E. and Paternoster, E. R. (1992). Cutting and packing problems: a categorized, application-orientated research bibliography. *Journal of the Operational Research Society*, pages 691–706.

Trivella, A. and Pisinger, D. (2016). The load-balanced multi-dimensional bin-packing problem. *Computers & Operations Research*, 74:152–164.

Wäscher, G., Haußner, H., and Schumann, H. (2007). An improved typology of cutting and packing problems. *European journal of operational research*, 183(3):1109–1130.

Wolpert, D. H. and Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82.

Zhao, X., Bennell, J. A., Bektaş, T., and Dowsland, K. (2016). A comparative review of 3d container loading algorithms. *International Transactions in Operational Research*, 23(1-2):287–320.

Zhu, W., Oon, W.-C., Lim, A., and Weng, Y. (2012). The six elements to block-building approaches for the single container loading problem. *Applied Intelligence*, 37(3):431–445.

Appendices

A. Appendix A

Minimizing the wasted space giving priority to smaller bins is equivalent to maximizing the packing efficiency (1)-(4).

$$\text{minimize} \quad \sum_{j\in J} \frac{(L_j \cdot W_j \cdot H_j - \sum_{i\in I}(l_i \cdot w_i \cdot h_i \cdot C_{i,j}))}{(L_j \cdot W_j \cdot H_j)} = \tag{1}$$

$$\sum_{j\in J} \frac{L_j \cdot W_j \cdot H_j}{L_j \cdot W_j \cdot H_j} - \sum_{j\in J} \frac{\sum_{i\in I}(l_i \cdot w_i \cdot h_i \cdot C_{i,j})}{L_j \cdot W_j \cdot H_j} = \tag{2}$$

$$n - \sum_{j\in J} \frac{\sum_{i\in I}(l_i \cdot w_i \cdot h_i \cdot C_{i,j})}{L_j \cdot W_j \cdot H_j} \Leftrightarrow \tag{3}$$

$$\text{maximize} \quad \sum_{j\in J} \frac{\sum_{i\in I}(l_i \cdot w_i \cdot h_i \cdot C_{i,j})}{L_j \cdot W_j \cdot H_j} \tag{4}$$

Where $\frac{l_i \cdot w_i \cdot h_i}{L_j \cdot W_j \cdot H_j}$ is the packing efficiency.