A Comparative Review of 3D Container Loading Algorithms

Xiaozhou Zhao[1], Julia A. Bennell, Tolga Bektaş

Southampton Management School
and
Centre for Operational Research, Management Science and Information Systems (CORMSIS)
University of Southampton, Southampton, SO17 1BJ, UK

Kath Dowsland

Gower Optimal Algorithms, UK

Abstract

Three-dimensional cutting and packing problems have a range of important applications and are of particular relevance to the transportation of cargo in the form of container loading problems. Recent years has seen a marked increase in the number of papers examining a variant of the container loading problem ranging from largely theoretical to implementations that focus on meeting the many critical real world constraints. In this paper we review the literature focusing on the solution methodologies employed by researchers, with the aim of providing insight into some of the critical algorithmic design issues. In addition we provide an extensive comparison of algorithm performance across the benchmark literature.

Keywords: cutting and packing, container loading, review

1. Introduction

The three-dimensional (3D) packing problem is a natural generalization of the classical one- and two-dimensional problems. The most commonly cited application for such problems is the transportation of goods that may be packed directly into containers or trucks or first packed on pallets. With a few exceptions, the literature focuses on items contained within three-dimensional boxes. Even with this restriction, it is clear that the relevance and scope of application is high. Despite its important industrial and commercial applications, historically, publications on this problem area are relatively few in comparison to its one- and two-dimensional counterparts. However, recent years has seen rapid growth in research and publications on this problem, including new research avenues that examine integrating packing with routing and scheduling.

Arranging boxes into a container, truck or pallet is one of the more complex packing problems with respect to real world constraints. While aiming to produce a packing arrangement that makes the best use of resources, an underestimate of the number of containers required to transport certain goods may

[1]Corresponding author. E-mail: X.Zhao@soton.ac.uk

be very costly in terms of delay or carrying extra containers that are under-utilized, resulting in wasted resource. In addition to the box sizes, there is a host of constraints associated with weight distribution, stacking, stability and support that need to be respected when determining the number of containers. Further, clients may constrain consignments to be carried together, or delivery vehicles need to unload at several locations. Bortfeldt and Wäscher (2013) provide a comprehensive review of such constraints.

The aim of this paper is to review the literature on 3D container loading. Here, the term *container loading* is used in its broadest sense, in a similar way to Bortfeldt and Wäscher (2013), who recently provided a review that specifically focuses on the inclusion of authors of various constraints encountered in practice but gives relatively little attention to the solution approaches. Our paper is a complementary review to that of Bortfeldt and Wäscher (2013) in that it focuses on the design and implementation of solution methodologies for solving these problems. We also provide an experimental comparison of different algorithms on benchmark data sets to identify the state-of-the-art in solution methods in the area. In order to contain the size of the review, we have excluded pallet loading, irregular objects and do not explicity review the open dimension problem although some papers addressing this problem are mentioned.

The rest of the paper is structured as follows. The next section provides a detailed description of the problem types reviewed in the paper, following the typology of Wäscher et al. (2007). Section 3 briefly discusses the constraints met in practice. Sections 4 and 5 discuss specific aspects of the solution approaches, these are the placement heuristics (how a layout is constructed) and the improvement heuristics (how to search for better solutions) respectively. Section 6 presents the research that examines the use of exact methods. Section 7 gives an overview of how the literature is split between different problem varients. Experimental results, along with benchmark data sets are analyzed in section 8. Conclusions are given in Section 9.

2. Problem Variants

According to Wäscher et al.(2007), cutting and packing problems can be grouped by dimensionality, assortment of large items, assortment of small items and the objective. Here we are only concerned with three dimensional items that are cuboid. We will call the large items containers and the small items boxes. Boxes may be *identical*, *weakly heterogeneous* (many boxes but a few box types) or *strongly heterogeneous* (few boxes and many box types). If there is more than one container, these can be identical, weakly or strongly heterogeneous. They provide the following problem typology.

If the objective is one of *input minimization*, then the aim is to pack all the boxes in as few containers as possible. Combining the classes of large and small item assortments gives six unique problems as follows,

- Single Stock-Size Cutting Stock Problem (SSSCSP) if the containers are identical and boxes are

weakly heterogeneous,

- Single Bin-Size Bin Packing Problem (SBSBBP) if the containers are identical and the boxes are strongly heterogeneous,
- Multiple Stock-Size Cutting Stock Problem (MSSCSP) if the containers and the boxes are weakly heterogeneous ,
- Multiple Bin-Size Bin Packing Problem (MBSBPP) if the containers are weakly heterogeneous and boxes are strongly heterogeneous,
- Residual Cutting Stock Problem (RCSP) if the containers are strongly heterogeneous and boxes are weakly heterogeneous,
- Residual Bin Packing Problem (RBPP) if the containers and the boxes are strongly heterogeneous,

If the objective is one of *output maximization*, then the aim is to pack a subset of boxes that give the highest value into a fixed set of containers. Here there may be a single container or multiple containers. Seven unique problems can be defined as follows:

- Identical Item Packing Problem (IIPP) if there is a single container and the boxes are identical
- Single Large Object Placement Problem (SLOPP) if there is single container and weakly heterogeneous boxes,
- Single Knapsack Problem (SKSP) if there is a single container and the boxes are strongly heterogeneous,
- Multiple Identical Large Object Placement Problem (MILOPP) if there are multiple identical containers and weakly heterogeneous boxes,
- Multiple Heterogeneous Large Object Placement Problem (MHLOPP) if the containers are weakly or strongly heterogeneous and weakly heterogeneous boxes,
- Multiple Identical Knapsack Problem (MIKSP) if there are multiple identical containers and strongly heterogeneous boxes,
- Multiple Heterogeneous Knapsack Problem (MHKSP) if the containers are weakly or strongly heterogeneous and strongly heterogeneous boxes.

All the above problems assume the containers have fixed dimensions. There is one further case where two dimensions are fixed and one, either the length or height, is variable. Clearly, this is a single container

input minimization problem, defined by Wäscher et al.(2007) as the Open Dimension Problem. We do not explicitly review this problem type here, some example papers are Allen et al. (2011), Bortfeldt and Mack (2007), Faina (2000), Fujiyoshi et al. (2009), He et al. (2012), Lai et al. (1998), Li and Cheng (1992), Miyazawa and Wakabayashi (1997, 1999), and Yeung and Tang (2005).

3. Problem Constraints

There are three basic constraints that are fundamental to the problem and observed by all research papers. These are stated as follows:

- boxes may only be placed with their edges parallel to the walls of the container,
- all boxes must be placed within the container,
- boxes may not intersect each other.

In many cases only solutions that result in the entire base of the box being fully support are acceptable, others allow a small amount of overhang. In cases where the centre of gravity of a box must be supported, it is assumed to be at the same position as its geometrical centre.

Beyond these basic constraints are a multitude of practical considerations. These are largely associated with box orientation, stability, stacking, weight distribution, weight capacity, multi-drop, prioritization and bearing strength of boxes. These are comprehensively discussed by Bortfeldt and Wäscher (2013), who divide them into constraints in relation to the container, items, cargo, positioning and load. We only briefly describe them here.

There are six unique orientations for a box. Many authors allow boxes to freely rotate, for example, Gehring and Bortfeldt, (1997), Wang et al (2008), Egeblad and Pisinger (2009). Perhaps more realistically is the case when the vertical orientation is fixed but boxes may rotate horzontally, for example, Heassler and Talbot (1990), Chien and Deng (2004). While a few papers (Scheithauer, 1992; Morabito and Arenales, 1994) disallow any rotation among boxes.

Load stability is an important consideration in practice, yet is inconsistently dealt with in the literature. If a load is unstable it can lead to cargo damage and difficulty loading and unloading. Use of straps and filling gaps with airbags between boxes and between boxes and the container wall is an industry practice, but costly and less desirable. Load stability is most commonly achieved through guaranteeing full support for the bottom of the boxes (Gehring and Bortfeldt, 1997). Others relax this constraint and allow partial support by defining a maximum overhang (Parreño et al., 2008; Parreño et al., 2010). Several papers include further constraints such as requiring at least one side to be attached to another box. Two measurements for evaluating cargo stability are proposed by Bischoff and Ratcliff (1995).

Stacking constraints refer to restrictions on how boxes are placed on top of each other as a result of the bearing strength of the boxes. Each box has a maximum weight per unit of area it can support that may vary depending on the box orientation. Also, Bischoff and Ratcliff (1995) argue that the load bearing strength of a box is primarily provided by its side walls. Therefore, in some cases it would be feasible to place an identical box directly on the top of the existing box, but damage may be caused if a box half the size with half the weight is placed centrally on top. Limits on the height a heavy box can be placed in the container may also be associated with stability.

The weight capacity and how the weight is distributed across the container are both important practical constraints for shipping and handling the already loaded container. The ideal weight distribution is measured if a container's centre of gravity is close to the geometrical mid-point of its floor. An unevenly distributed weight may cause difficulties for certain handling operations. In the case of loading a vehicle, the weight on the axels is the critical measure. Further, containers may not exceed a total weight, and in some scenarios, weight capacity, rather than the limited available loading space, is the binding constraint.

Multi-drop is a situation in which a container will be unloaded in a number of different terminals (Bischoff and Ratcliff, 1995; Lai et al., 1998; Ren et al., 2011). Boxes with different consignments are packed separately as an effective strategy to deal with this situation. Therefore, the arrangement of packing patterns is better to be designed in a way that unloading and re-loading a large part of cargo at every destination is avoided.

The above mentioned constraints, which have direct impact on the packing patterns, have to be taken into consideration within the single container heuristic. Eley (2003) defines two other constraints that impact the distribution of boxes among the different containers. Separation of boxes refers to the situation that boxes of two different types must be stored in separate containers. In practice, foodstuff and perfumery articles are required to be transported separately. Complete shipment refers to the situation that shipment should include all other boxes belonging to a certain sub-set.

4. Placement Heuristics

In any heuristic solution approach, there needs to be a mechanism to decide how to put the boxes in the container, whether this is simply to generate an initial solution or as an integral part of the approach. Here we call this the placement heuristic, also commonly known as the construction heuristic. For container loading there are a wide range of approaches. When the mix of boxes are weakly heterogeneous then the most common are wall building and layer building, whereas in the strongly heterogeneous case, boxes will be placed one at a time.

Under wall and layer building schemes, boxes of the same type are arranged in rows or columns to fill one side or the floor of the empty space. A list of empty spaces is created for all the feasible placement

positions. Once an empty space accommodates a wall or layer, new spaces are generated. Generally, once a wall or layer is built, the remaining space is treated as a reduced-sized container. Both approaches mimic manual packing by attempting to create flat packing faces.

Along with the decision of how to pack the boxes comes the decision of which box type to pack next. Common approaches include pre-determined ordering and dynamic ordering. The pre-determined ordering is achieved by employing some sort criteria, such as box volume, number of boxes, base area or a certain dimension of a box. Dynamic sorting tends to be based on the usable space remaining after the next placement, volume of remaining boxes of the same type and free floor space.

The following sections discuss the placement heuristics including the sequencing decisions. Many of these papers may employ improvement strategies in addition, these are discussed in the following section.

4.1. Wall-building

The basic wall-building scheme fills the container with a number of walls across the depth of the container. While the walls are created sequentially, issues around weight distribution can be dealt with by swapping, interchanging or taking the mirror image, provided the boxes do not interlock.

The earliest example of wall-building is by George and Robinson (1980). They design a two-section heuristic based on a real industry example of 800 boxes with no more than 20 types. Given the next empty space, initially the whole container, they select the first box of each wall so that each wall has a size which is not too deep or too shallow using the following ranking criteria. First, select the box whose smallest dimension is the largest among all candidates. In case there is a tie in the first criterion, select the box type with the highest quantity. If there is still a tie, select the box type with the longest largest dimension. Once at least one of a certain box type has been packed, that box type is open. Open box types are given preference where the type with largest quantity has the highest ranking. If the length of unfilled container is too short, usually below a certain defined amount, the rest packing no longer follows the wall-building rules.

Based on the George-Robinson framework, Bischoff and Marriott (1990) compare 14 different heuristics, combined by six ranking rules and three filling methods developed by others, with different ranking criteria. Without a clear winner, they proposed a hybrid version where all 14 heuristics are executed and compared to select the best solution.

Moura and Oliveira (2005) adopt the placement heuristic of George and Robinson (1980) with two modifications. The first modification arises from new space creation. While George and Robinson's heuristic assumes the width of original space as the width of the new created height space, Moura and Oliveira (2005) restrict the space to the area above the packed boxes, citing issue with stability in the former approach. The second modification restricts the width of new wall to be no longer than that of the previous

wall, resulting in the amalgamation of unpacked spaces into larger more useful spaces and increasing cargo stability.

Instead of using a fixed sorting criteria, Chien and Wu (1998) and Pisinger (2002) determine the arrangement of boxes by a tree-search algorithm and use the wall approach first proposed by George and Robinson (1980). In Pisinger's (2002) extension of wall building approach, a tree search is designed to find the set of wall depths and strip widths, which can be either vertically or horizontally oriented, to make up the wall. Firstly different wall depths are selected at each branching node, followed by the strip width. Only a fixed number of sub-nodes are considered for each branching node and back tracking is permitted. Several different wall depths are considered according to a total of 27 ranking criteria at each branching node.

Bischoff and Ratcliff (1995) consider the container loading with where cargo has multiple destinations. Their approach packs each consignment in sequence starting with those assigned to the final destination. Given a consignment, the next box packed and its orientation is selected in order to maximize the remaining usable space. In the case of a tie, they select the box space combination with the smallest length protrusion. Two further tie-breakers are to choose the box with largest volume and then to choose the space with shortest width. Here the wall building strategy sequentially adds individual columns or a container width instead of complete walls.

Gehring et al. (1990) pack boxes into a single container of known dimensions using a wall-building approach. The length (thickness) of each wall is decided by the length of the 'Layer Determining Box' (LDB) which is the first, usually largest, box placed in that wall. Two non-overlapping spaces are generated in that section once the LDB is placed. The first space stays next to the LDB with the same height. The second space is above LDB and the first place and is across the container width. Each box is allowed to stay within a wall rather than project into adjacent walls. Davies and Bischoff (1999) adopt the concept of LDB for generating a block which consists of a fixed number of walls rather than a single vertical wall. Boxes are allowed to bridge the adjacent walls but not two separate blocks. The ranking criteria for selecting box types and orientations are based on Bischoff and Ratcliff (1995). When it is impossible to place another LDB, the current wall is considered as the last wall and its wall depth is extended to the end of the container rather than the length of the LDB. Bortfeldt and Gehring (2001) also use the wall construction heuristic based on LDB as part of a hybrid genetic algorithm.

Bortfeldt and Gehring (1998) and Bortfeldt et al. (2003) construct cuboid configurations of a single box depth, which are effectively walls. In a 1-arrangement, the cuboid consists of as many of one box type as can feasibly fit along the width and height. A 2-arrangements places two 1-arrangement cuboids next to or one on top of the other. All empty spaces are stored in a list, and the space with smallest volume is always packed first. Two available standards are applied alternatively to evaluate the local arrangements

within a packing space. The first standard is that the overall volume of the boxes placed in the space should be maximized. The second standard contains double criteria of smallest possible loss volume and largest possible maximum effective volume.

Chien and Deng (2004) describe a container packing support system to determine and visualize, in a step-by-step manner or continuously, the container packing pattern that consists of the packing orientation of each box and corresponding location within a fixed dimensional container. Boxes are packed into vertical strips which later are combined into walls. The spatial matrix representation method is adopted when searching and merging the empty spaces.

4.2. Layer-building

Layer building is also a common placement heuristic, although fewer researchers adopt this approach. The packing arrangement is constructed by first placing boxes on the bottom of the container to create the base layer. Once this layer is full, the next layer is constructed on top of the base layer, and so on until no further layers can be contained within the height of the container. It is important to note that some paper describe layers that, according to our terminology, are walls.

The approach of Bischoff and Ratcliff (1995) is inspired by pallet loading. Patterns are built from the container floor upwards in the form of layer building. Each layer contains no more than two different types of boxes, selecting the box type and orientation according to the utilization of the loading surface on which a layer is built. Their motivation for this approach was in tackling stability resulting from significant height differences or gaps exist between adjacent walls of boxes found in other approaches. Lim et al. (2012) follow the same layer building approach as part of an iterative heuristic that focuses on the issue of packing awkward box types. A certain boxes packing priority is revised after each iteration to reflect how "awkward" it was to pack. As a result awkward boxes are packed earlier.

Ratcliff and Bischoff (1998) build on the previous work and design a container loading algorithm which allows for load bearing constraints. Their heuristic approach iteratively packs layers of boxes one at a time from the container floor upwards. Each layer contains no more than two rectangular blocks. Boxes in a single block are of the same types and all have the same orientation. At each step, the algorithm checks the inequality between the weight of the chosen block and the load limit on the loading surface. Boxes with a low load bearing ability can not be packed in the early stages of the procedure. Therefore, the opportunities for future placements on higher layers are examined at each step.

Loh and Nee (1992) propose a layer building construction heuristic. Weakly heterogeneous boxes are grouped according to height and then groups are sorted tallest to shortest. Within each height group, they are sorted by descending base area. Boxes are packed starting at the back of the container, in order, in the first large enough space. Lodi et al. (2002) following a similar sorting strategy, however, group boxes if

they are within a certain hight tolerance. If the tallest box in the group has height h, then all boxes with height between h and αh are also members of that group. α is set between 0 and 1. Like Loh and Nee (1992), boxes are sorted by non-increasing base area within groups. A two dimensional packing heuristic creates the layer by evaluating and scoring candidate positions. After generating layers that accommodate all the pieces, they solve a one dimensional bin packing problem to pack layer heights within a finite bin height. They call the heuristic Height first - Area second (HA) and combine it with tabu search.

4.3. Other placement heuristics

Blocks are homogeneous and each block consists of only identical boxes. Packing container with homogeneous blocks results in several advantages. First of all, it is easy to arrange and requires less loading time. Re-sorting cargo is avoided after unloading as boxes of same type are stowed closely to one another. The constraint of load bearing strength is less problematic when identical boxes are stacked. The block structure provides extra stability as identical boxes packed in rectangular shape cannot easily slip. Liu et al. (2011a) develope a hybrid tabu search approach in which both a tabu search and a block-building heuristic run iteratively. Box types are sorted in decreasing volume to generate the initial solution. Blocks are made up of the same box type in a configuration to best fit the available space. In some cases another box type is added to improve the fit. Ren et al. (2011) adopt a block placement strategy to tackle shipment priority within a single container. They define five block evaluation functions and use a tree search strategy branching on the best block placement for each evaluation function. Wang et al. (2008) also use a tree structure to represent the container and branch on the mutually exclusive sub-spaces left after placing a block of identical boxes. Zhang et al. (2012) propose a block-building heuristic under which multiple types of boxes can be bound in one single block. A multi-layer search algorithm is designed for block selection.

Lai and Chan (1997) propose the maximal space strategy. Maximal space intervals are the set of largest cuboid spaces that entirely cover the unpacked volume of the container. Note these spaces may overlap. After a box is placed, these are generated and stored in a list. Any intervals that are not sufficiently large to contain a box or that are entirely contained within other space intervals are removed from the list. The next box is packed in the closest space to the bottom left corner of the container that is large enough to accommodate the box. Many efficient algorithms adopt the maximal space approach. For example, Parreño et al. (2008; 2010) design a constructive heuristic that can place columns, rows, walls or layers of boxes of the same type. The main idea is to identify maximal spaces and then fill them with the best configuration of identical boxes according to one of two criteria. In each iteration the maximal space with the minimum distance to a corner of the container is filled where space volume is used to break ties. This heuristic is combined with GRASP in Parreño et al. (2008) and Variable Neighborhood Search in Parreño et al. (2010). Other recent papers generating competitive results on benchmark dataset embrace this idea

(Gonçalves and Resende, 2012; Zhu and Lim, 2012; Zhu et al., 2012b; Araya and Riff, 2014).

Gehring and Bortfeldt (1997) pack boxes into towers or stacks when dealing with strongly heterogeneous boxes. Towers have a base box directly placed on the container floor. Each of the rest of boxes within that arrangement is placed on another box in a way that its bottom area is entirely supported from below. A greedy algorithm is used to minimize the spaces wasted above the base boxes. In the second step, tower bases are arranged to cover the floor of the container using as a two-dimensional packing problem.

Eley (2002) solves single container loading problem with boxes in block arrangements. The proposed greedy heuristic sorts boxes by volume in the non-increasing order. Each combination of box orientation and empty space is investigated by certain criteria, and the most appropriate empty space is chosen for a box. The main criterion is that after packing a box the sum of the volume of spaces where no remaining boxes can be packed is minimized. In the case of a tie, preference is given to the empty space closest to the lower back left corner of the container. The solutions are further improved through considering different box loading sequences via a tree search where branching is carried out for different box types and box orientations.

Lim et al. (2003) sort the boxes by evaluating a pairwise priority according to the ratio of volume and base area. They propose a new construction approach called multi-faced buildup algorithm where boxes can be placed on any wall in the container as each and every wall can be used as base or floor. Once boxes are placed on the wall of the container to construct the first base, the other boxes will be placed on top of these boxes.

Ngoi et al. (1994) avoid the dependency of the layout on the order of the boxes by evaluating every potential placement location for each box. In order to improve the efficiency of their approach, they use a spatial representation technique. A single 3D matrix (or the spatial matrix), in the form of layers of 2D matrices is used to represent the positions and dimensions of all the packed boxes and empty spaces. Chua et al. (1998) and Chien et al. (2009) also use the spatial representation techniques to model the box packing process and to track both the packed and unpacked space in a single container. Compared to that of Ngoi et al. (1994), their algorithm allows users to pre-assign positions for certain boxes. Spacial matrices are also used by Bischoff (2006) who develops a new construction heuristic to specifically consider load bearing strength. The approach guarantees all boxes are directly placed on the container floor or have their whole base in direct contact with other boxes. The constructive heuristic packs a single box in each iteration and uses a scoring system made up of five criteria to choose between placement alternatives. The available loading surfaces are stored and updated in two matrices, one records the height of surfaces and the other records their corresponding load bearing ability.

The above packing heuristics model available spaces. Martello et al. (2000) and Crainic et al. (2008) both follow an alternative strategy of identifying placement points, the former calling it corner points, the

latter extreme points. These become the candidate placement positions for placing the unpacked boxes. Consider the packing profile over the 3D surface (if overhang not permitted), these points arise from contact between the face of two boxes or a box and the container and are located at the point where three edges coincide to create a fully concave corner. Martello et al. (2000) approach first distributes boxes between bins before determining the specific position of each box. Each bin is constructed using a tree search where each node is a partial solution and there is a child node for each feasible corner point for the next box, where boxes are packed in non-increasing order of volume. Crainic et al. (2008) simultaneously assigns boxes to containers and packs using the well known best fit decreasing heuristic.

A similar placement strategy is proposed by Huang and He (2009a) and Huang and He (2009b) to pack boxes into a single container. In their papers they call it caving degree where boxes packed into a corner or even a cave, an empty hollow-like space surrounded by many boxes, whenever possible. The caving degree approach stores all available corners into a list and evaluates all combinations of corners, types of boxes and allowed orientations selecting the best combination of corner, box and orientation based on ranking rules. He and Huang (2010) suggest a modification to the original approach in the effort of decreasing the searching space. He and Huang (2011) make further improvement to the modified approach including adding a local search phase.

Han et al. (1986) construct an L-shape pattern by placing boxes along the base and one vertical edge of the container. George (1992) designs a heuristic to solve the same problem of identical boxes. The heuristic runs iteratively and each iteration creates a layer. The reorientation of the container is allowed and layers can be built against any side wall or even floor rather than only the end wall suggested by common approaches.

Lins et al. (2002) provide a directed graph representation of a packing arrangement that builds on a successful application to the two-dimensional case. Three directed graphs are needed to represent the x, y and z edges of each box within the container. They claim that degeneracy and symmetry lead to a sufficient reduction in problem size to allow for enumeration. Experimental results are for identical boxes with six orientations and a largest problem size of 394 boxes.

Clearly there are many diverse ways of creating arrangements of boxes to load a container in terms of placement rules and static and dynamic ordering. While there is no definitive answer as to the best approach, sorting by volume is a very common characteristic. Chien and Wu (1999) proposed the integration of space allocation and ranking in a decision support system, although did not provide any experimental evidence.

4.4. Sequencing

Sequencing refers to the ordering of the boxes and can be critical to the effectiveness of the packing heuristic. Modifications to the placement heuristics often involve altering the box sequence. Here we summarize the most common box sequences described in the literature. We categorize sequences into two groups: static and dynamic. *Static sequencing* refers to fixed ranking of boxes or box types before packing takes place. *Dynamic sequencing* refers to the criteria that decides the next box or space to be packed during the packing process. Twelve static sequencing rules and five dynamic sequencing rules are identified through the review. Table 1 provides a tabulated listing of the sequences found in the literature. The primary ranking criteria for a particular paper is stated as 1, and the first and second tie breaking criteria are indicated as 2, 3, and onwards. It is worth to mention there are two slight exceptions. In Xue and Lai (1997), three criteria are picked and all combinations with a third-tier tie are tried . There is no clear winner of 6 combinations. Among Crainic et al.'s (2008) proposed sequencing rules, two combinations generating best results are remained in the table. However, the first priority of both combinations is slightly different from the ones we defined in this paper. Take the first combination, clustered area and height, as an example. Clustered area means separating base areas into clusters defined by certain intervals. Boxes are then assigned to clusters according to their base areas. Clusters are ordered by certain rule, and boxes within the same cluster are ordered by height.

		Static												Dynamic						
Year	Article	SA	SB	SC	SD	SE	SF	SG	SH	SI	SJ	SK	SL	DA	DB	DC	DD	DE	Placement	Container
1980	George and Robinson, 1980				1	2	3												Wall	Single
1990	Gehring et al., 1990												1						Wall	Single
1992	Loh and Nee, 1992		1							2									layer	Single
1993	Lin et al., 1993											1	2						layer	Single
1994	Hemminki, 1994						2						1						Wall	Single
1995	Bischoff and Ratcliff, 1995												3	1			2		Wall	Single
1997	Xue and Lai, 1997			1		1		1												Multiple Identical
1998	Bortfeldt and Gehring, 1998		1																block	Single
1998	Ratcliff and Bischoff, 1998													1					Layer	Single
1999	Chien and Wu, 1999					1							1						Other Placements	Single
1999	Davies and Bischoff, 1999												3	1			2		Wall	Single
2000	Martello et al., 2000							1					1						Corner/Extreme Points	Single
2001	Bortfeldt and Gehring, 2001							2		3			1						Wall	Single
2002	Gehring and Bortfeldt, 2002												1					2	Wall	Single
2002	Lodi et al., 2002			2						1									Layer	Single
2002	Eley, 2002												1						Block	Single
2003	Eley, 2003															1			Block	Single
2004	Chien and Deng, 2004	1		2				3	4	5									Wall	Single
2005	Lim et al., 2005							2			3		1						Other Placements	Single
2005	Moura and Oliveira, 2005													1					Wall	Single
2006	Takahara, 2006							2	3	4			1						Wall	Multiple Identical
2008	Crainic et al., 2008			1						2									Corner/Extreme Points	Single
	Crainic et al., 2008			2						1										
2008	Parreno et al., 2008					2													Maximal Space	Single
2008	Takahara, 2008												1						Wall	Multiple Multi-sized
2009	Crainic et al., 2009												1							Multiple Identical
2010	Almeida and Figueiredo, 2010							3	4	5			1							Multiple Identical
2010	Fanslau and Bortfeldt, 2010														1				Block	Single
2010	Wu et al., 2010												1						Corner/Extreme Points	Single
2011	Che et al., 2011													1						Multiple Identical
2011	Dereli and Das, 2011							2	1										Wall	Single
2011	Liu et al., 2011												1						Block	Single
2012	Kang et al., 2012							4	2	3			1						Other Placements	Single
2012	Zhang et al., 2012													1					Block	Single

SA: base dimensions which are the depth and width of the bottom of a box; SB: aggregate base area which is the sum of the base areas of an arrangement; SC: base area of a box; SD: size of the smallest dimension;

SE: the quantity of boxes in the box type; SF: the length of the largest dimension; SG: the depth of a box; SH: the width of a box; SI: the height of a box;

SJ: the perimeter which is the sum of the boxes' 12 dimensions; SK: the weight of a box; SL: the volume of a box;

DA: potential volume utilization or total volume use within the next free space to be packed; DB: the volume of a block which consists of boxes of same type with same orientation; DC: the volume of remaining boxes of the same type;

DD: lengthwise protrusion which is the sum of the coordinate from the packing point and the length of the box to be packed; DE: bottom area of the left free space.

Table 1: Sequencing Table

5. Improvement Heuristics

While the placement heuristics will provide a fast, and in many cases reasonable quality, solution, broadening the search space using an improvement heuristic can usually provide a significant gain. In general, these work with one or more complete solutions and make neighborhood moves to find better solutions. These vary in sophistication from simple neighborhood structures and acceptance criteria that only accept improving moves, to complex and varying neighborhoods and acceptance criteria that can find many local optima including many implementations of the standard metaheuristics. In this section we review how these techniques have been employed for 3D container loading and how they build on the placement heuristics described in the previous section.

Many of the placement heuristics are dependant on a sorted order, or permutation, of boxes. This representation is often used in a Genetic Algorithm (GA) implementation. Its popularity may be due to the many genetic operators designed to deal with permutation representations in the GA literature.

One of the earliest examples is Hemminki (1994) who uses a GA to determine the widths of the walls. Each chromosome consists of a string of integers. Each integer indicates the packing strategy used in corresponding wall, and all these walls are combined into a whole container. Parent chromosomes are selected randomly and the selection chance of each chromosome is directly linked to its fitness. Crossover or mutation is applied to two parents in order to generate two child chromosomes. Best combinations of walls will be decided through the evolving process of the generations.

The GA of Gehring and Bortfeldt (1997) solves a two-dimensional subproblem. As discussed earlier, boxes are arranged into towers. The tower bases form the two-dimensional packing problem. A chromosome represents the solution by a placement vector, which indicates the sequence of the placements of the tower bases and the two possible orientations of each tower base. Each chromosome is transformed into a solution by identifying an order list of placement corners and placing the boxes in the first feasible placement corner. The population are ranked according to fitness value and the ranking provides the selection probabilities. Since the chromosomes in this cases are permutations of tower bases, a permutation preserving operators are applied to generate descendants representing feasible solutions. Crossover and mutation are alternated randomly. A similar framework for the GA is implemented by Bortfeldt and Gehring (2001). However, in this work the GA is representing layers (walls) of boxes that are constructed using a basic heuristic based on the LDB. The basic heuristic is used to generate starting solutions and new layers for offspring via crossover and mutation. The chromosome with the best stowage plan finally undergoes resequencing of the layers in order to determine the best weight distribution. Gehring and Bortfeldt (2002) try to improve the solution quality through the application of a parallel genetic algorithm. The new model is implemented in a local PC network and each GA instance is assigned to a separate workstation.

Wu et al. (2010) propose two different approaches to load boxes into a single bin. The second approach is a genetic algorithm. Each bit of the chromosome has two parts, the box and its rotation angle numbered from one to six. Initially all the GA chromosome encode the boxes in non-increasing volume order with a randomly assigned rotation. Reproduction uses roulette wheel selection, a one point crossover with repair mechanisms and a mutation that randomly swaps boxes or changes rotations. The best solution is copied to the next generation.

Gonçalves and Resende (2012, 2013) also use a permutation structure but avoid the need to correct descendent chromosomes by using a biased random key strategy. The strategy considers a permutation of boxes and randomly assign a real number to each box. Sorting the boxes by the random number provides a new permutations. The biased random key approach applies the genetic operator to the random keys rather than the boxes, hence ensuring any offspring is feasible. Their implementation doubles the length of the chromosome to include a placement heuristic for each box.

Dereli and Das (2011) propose an alternative population heuristic that has many similarities to GA but is inspired by foraging behavior of honey bees. The bees algorithm works with a population of solutions and evaluates their fitness, where promising solutions are selected and undergo a neighborhood search phase. The best solutions and some newly generated solution go on to the next population. Layouts are constructed using LDB placement heuristic hence the focus of the search algorithm is to find the alternative widths of layers for a better overall performance through enabling or disabling the rotation of the boxes.

Tabu Search (TS) is one of the most popular metaheuristics in combinatorial optimisation and this is also reflected in how often it has been adopted in 3D packing. Bortfeldt and Gehring (1998)and Bortfeldt et al. (2003) employ tabu search in combination with their wall building heuristic for loading a weakly heterogeneous boxes into a single container. A complete solution is first generated through a basic heuristic. The tabu search algorithm operates on the encoded problem in the form of a packing sequence and fillable packing spaces. Neighborhood moves involve deviating from the fillable space order. The tabu list holds the complete packing sequence of the selected neighbor. Parallelization of the tabu search algorithm is later applied to improve the solution quality (Bortfeldt et al., 2003). The general approach in these papers is built on by Mack et al., (2004), who apply a simulated annealing (SA) algorithm as an alternative to the TS while maintaining the basic heuristic. SA is later combined with TS to form a hybrid search system. Efforts are taken to integrate the advantages of both meta-heuristics and avoid their weakness. Therefore, SA is pre-processed to determine a good starting solution for one or more TS iterations. The parallelization of SA and the hybrid search is introduced in the end.

Liu et al. (2011) also perform a search over the assignment of boxes to spaces. Instead of altering the free space assigned to a box, the search alters the box packing order using a swap neighborhood. The tabu list length varies with the size of the problem.

An intuitive way to combine TS and a placement heuristic when solving the multi-container problem is to use the TS to assign boxes to containers and then use the placement heuristic to determine the layout within the container. This is the approach of Jin et al. (2003), who design several neighborhood moves with the aim of eliminating a target bin. Crainic et al. (2009) suggest a two-level tabu search. The first-level tabu search, similar to Jin et al. (2003), focuses on assigning boxes to bins without deciding their placement. The neighborhood focuses on changing bin assignment through swapping and reassigning boxes between containers. The second-level tabu search works to arrange the boxes in each container using an extreme point construction heuristic and searching over the interval graph proposed by Fekete and Schepers (1997, 2004). Each pair of boxes in the same bin will follow a certain approach to form seven feasible combinations.

Instead of assigning boxes to containers Lodi et al. (2002) use tabu search to assign boxes to two dimensional layers. Their unified tabu search approach has been successfully applied in two dimensional bin packing. The tabu search governs the boxes in each layer and attempts to reduce the total number of layers. The neighborhood moves are designed to empty a specific target layer that is considered the weakest or most empty. A single box from the target is combined with all boxes in k other bins. The neighborhood includes different values of k as well as different boxes from the target layer. The tabu list holds move penalties determined by the outcome of the move.

Guided local search (GLS) has many similarities to TS in terms of using memory to guide the search. Farøe et al. (2003) tackle the SBSBBP using guided local search (GLS). The paper is novel in that it permits overlap between boxes within the containers. Given a fixed number of containers, the aim is to find a feasible arrangement of boxes, once found, one container is eliminated and the boxes are distributed over the remaining containers and the search begins again. A neighborhood move arises from moving a single box along one coordinate axes within a single container or moving a box to the same co-ordinate position in another container. Features that guide the local search away from local optima are defined by the volume of overlap between box i and box j and the total volume of i and j. Due to the large neighborhood, fast local search is used to find local minima.

Apart from Mack et al. (2004), SA has not been a common choice of meta-heuristic. Further examples are Lai and Chan (1997) and Faina (2000), both examine the open dimension problem. In the case of Lai and Chan (1997), neighborhood moves arise from swapping two boxes in the packing order and the search is governed by a fairly standard cooling schedule. Results are reported for the 2D problem but not the 3D problem. Faina (2000) also performs neighborhood moves over the sequence of the boxes. A clear disadvantage to this approach is the computational cost of reconstructing solutions after a move. Faina (2000) reports that beyond 64 boxes the computational effort of improving the solution is unfavorable.

While GA, TS and SA are the classic trio of meta-heuristics, Greedy Randomized Adaptive Search

Procedure (GRASP) and Variable Neighborhood Search (VNS), are popular predecessors. GRASP is designed to use a greedy solution construction heuristic and perturb characteristics of the construction process by making selections randomly from a restricted candidate list rather than the most greedy choice. Hence, it has clear advantages for combining with a placement heuristic for container loading.

Lai et al. (1998) consider the open dimension problem where boxes are grouped into collections of customer orders and packed as separate sub-lengths of the container. Hence they argue that they can solved each subproblem of minimizing the packing length of each customer order. They model the problem as a graph and show that the optimal soluton is when the maximum clique is equal to the number of the boxes. They solve the maximum clique problem exactly and the maximal clique problem using GRASP.

Moura and Oliveira (2005) sort the available box types by the volume utilization given the next empty space. The basic greedy strategy chooses the first box type. Under GRASP, they select a random box from a candidate list of the best candidates controlled by a threshold parameter set between zero and one. When set to one the selection is purely random and when set to zero it is purely greedy. The constructed solution is then improved by a first found improve local-search where the neighborhood unpacks the tail of the sequence then repacks greedily forbidding the first box in the tail to be returned to its previous position. A similar implementation of GRASP is used by Parreño et al. (2008) and combined with the maximal-space placement heuristic. This time candidates join the restricted list according to rank (i.e. being in the top $100a\%$, where the parameter a is between zero and one) rather than meeting a quality threshold. Initially, a is randomly chosen from a set of possible values. As the search progresses probabilities are attached to each possible value according to its past success.

Following the successful implementation of GRASP, the same authors combine their maximal-space heuristic with VNS algorithm (Parreño et al., 2010). The authors propose five types of neighborhood move for the descent phase of the VNS and an additional neighborhood move for the shaking phase. These are layer reduction that removes rows or columns from a layer and attempts to fill the empty space with other boxes, column insertion adds a new column in one of the empty spaces, box insertion is similar to column insertion but only adds a single new box, and the last two moves are to empty an already packed region and to fill it using best-volume for one neighborhood and best-fit for the other. The shaking neighborhood eliminates between 10% and 30% of the boxes in the current solution and refilling with the best-volume criterion.

While the meta-heuristic approaches described above seek to be less greedy or at least include phases that allow a more diverse search of the solution space, other authors simply perform a greedy local search many times. For example Takahara (2006, 2008) perform local search over the permutation of pieces and piece orientation via swap moves. Multiple starting solutions provide a broader search of the solution space. Another approach is to make the greedy choice after looking ahead a few steps, thus reducing

the short-term focus of the decision. Zhu and Lim (2012) describe a look-ahead tree search that decides the next placement on the basis of the best node of an expanded tree of depth d where the m best child nodes are expanded at each node. A predecessor of this paper, Eley (2002) uses a restricted tree search to construct a solution. The tree is searched breadth first and a limit is imposed on the number of child nodes explored. In addition, to avoid a dominant solution path, if two nodes at the same depth have the same evaluation function and the item is packed, then one of the nodes is removed. They call the tree search the pilot method.

6. Exact Methods

Compared to the number of heuristics available, the number of exact algorithms in 3D container loading are limited. One reason behind this limitation is the difficulty in representing possible patterns or practical packing constraints. Even if this can be done (and we provide pointers to the literature below where it is), another difficulty is the solution of the resulting formulation, which is often large-scale due to the number of containers and boxes. Nevertheless, encouraging progress has been made along this line of research for which a review is presented below.

Mohanty et al. (1994) describe an algorithm for the (3D) BPP, where the aim is to pack boxes of different types, each with a prespecified number, length, width and height, into a set of containers of different sizes, with the objective of maximizing the total value of packed boxes. The problem is originally formulated as a multidimensional knapsack problem, including an integer variable that is defined with respect to each container type packed with respect to a given pattern. As the number of such patterns is excessively large, the complexity of solving the formulation to optimality is a challenge. For this reason, the authors describe a column generation procedure where the pricing problem involves solving an integer program, in the form of a general fractional knapsack problem with side constraints, which itself is difficult to solve. Instead, the authors propose heuristics to solve it. Although the overall solution approach is heuristic in nature, it is interesting that elements of exact solution methods (i.e., mathematical formulations) are integrated within the approach. In a relevant study, Eley (2003) looks at a similar problem setting, possibly with some side constraints, and an objective function that minimizes the number of bins used using a two-stage column generation algorithm. In the first stage, a limited number of patterns (columns) are generated *a priori* using a heuristic, which are then fed into an integer program that is solved to optimality in the second stage. The algorithm has the advantage of being able to incorporate a number of practical constraints, such as box orientation or load stability. The computational results that the author presents on benchmark problems suggests that the integer program does not require more than five seconds of solution time, and the algorithm itself outperforms those by Ivancic et al. (1989) and Bortfeldt (2000).

Given the success of column generation, it is not surprising to see that Zhu et al. (2012a) have also described a similar algorithm for the Multiple Bin-Size BPP, where the aim is to minimize the cost of containers used. The problem also incorporates some side constraints, namely full-support and partial-support for boxes and that all boxes are packed orthogonally. The pricing problem here is a single container loading problem. Instead of generating the actual columns, which is costly, the authors instead resort to a strategy where a reasonable approximation of the columns, named prototype columns, are generated. Compared with the algorithm of Che et al. (2011a), the proposed method finds solutions that improve the average gap from the optimal solutions by about 50%.

An alternative approach to mathematical models where variables are defined with respect to each possible pattern (or packing) has been to use formulations where variables are defined with respect to placement of individual items. Such an approach avoids the problem of dealing with a large set of patterns, and instead relies on a discretization of the container space and an explicit representation of decisions on whether items are placed in certain coordinates or not. Chen et al. (1995) present such a model in the form of a mixed integer linear programming formulation for the general 3D container loading problem, with an objective to minimize the unused space of the containers used. The model is used to solve a sample instance of three heterogeneous containers and six boxes using LINGO, with a solution time of 15 minutes. The authors also report results on another sample instances with a single container and six boxes, this time to minimize the required length of the container. A polyhedral study of an improved version of this formulation is presented by Padberg (2000). The study provides some facet results and valid inequalities, with the recommendation that they are used within a branch-and-cut algorithm to solve the problem. Allen et al. (2012) provide empirical evidence that the computational reach of this formulation is only about 20 boxes using modern solvers. The authors present an alternative formulation that is based on space-index variables, which has better performance than that of the Chen et al. (1995) and Padberg (2000), both with respect to the size of the instance and the computational time required to solve to optimality. Wu et al. (2010)builds on the work of Chen et al. (1995) for the open dimension problem.

For a mixed integer programming formulation of the BPP with identical bins, see Hifi et al. (2010), who also present some valid inequalities and computational results. Another model based on the Cartesian coordinate system is presented by Junqueira et al. (2012a) for the 3D container loading problem, where constraints representing vertical and horizontal stability and load bearing strength of the boxes are explicitly represented in the formulation. Computational results presented by the authors show that by using CPLEX 11.0 with default parameters, the formulation is able to solve randomly generated instances with four types and up to 20 boxes, and between 10 and 100 containers, in most cases to optimality. The difficulty of finding an optimal solution increases with the number containers as well as with the similarity of dimensions of the boxes. An extension of this model to the variation of the problem with multi-drop

constraints is presented in Junqueira et al. (2012b).

Lim et al. (2013) present a heuristic algorithm for a problem named the single container 3D loading problem with axle-weight constraints, which arises due to the weight restriction for trucks which carry containers when they are offloaded from ships. This additional restriction introduces nonlinearities in a possible mathematical programming formulation of the problem. The authors describe a heuristic procedure, but make use of integer linear programming formulations, either mixed or binary, within the algorithm to improve the packing.

Finally, we mention the work by Hifi (2002) presents approximation algorithms for container loading problem. The algorithms are based on the idea of solving a series of knapsack problems to form stacks, and then putting the stacks into the container by solving unbounded knapsack problem.

7. Implementation by Problem Type

In this section we categorise the papers according to the problem types they address. Bortfeldt and Wäscher (2013) provide a useful table that identifies each paper by its problem type. Here we go further and identify the features of the methodologies employed for each of these problems.

7.1. Single Container Loading

Among the papers we have reviewed, twenty-five tackle SLOPP. The type and sophistication of the methodologies vary greatly. Thirteen of the paper only employ a placement heuristic, and of these thirteen seven use a static ranking and a placement criteria (Bischoff and Marriott, 1990; Chien and Deng, 2004; Chien et al., 2009; Christensen and Rousøe, 2009; George and Robinson, 1980; Loh and Nee, 1992; Ren et al., 2011). The remaining six use dynamic selection of the next box or space (Bischoff and Ratcliff, 1995; Davies and Bischoff, 1999; Ratcliff and Bischoff, 1998; Wang et al., 2008), or a tree search to make the exploration less greedy (Lins et al., 2002; Morabito and Arenales, 1994). Eleven papers augment the placement heuristic with an improvement heuristic, or design an improvement heuristic that directly tackles the problem. Of these two use genetic algorithm (Gonçalves and Resende, 2012; Kang et al., 2012), three use tabu search (Bortfeldt et al., 2003; Liu et al., 2012a; Mack et al., 2004) and one uses simulated annealing (Mack et al., 2004), with the rest adopting various local search techniques (Bischoff, 2006; Burke et al., 2012; Eley, 2002; Hifi, 2002; Moura and Oliveira, 2005; Parreño et al., 2008, 2010). Finally Junqueira et al. (2012b) is the only paper we reviewed that developes exact methods for SLOPP.

Thirty-three papers aim to solve SKP. The methodologies differ largely in terms of type and sophistication. Ten of the paper only employ a placement heuristic, and of these ten five use a static ranking and a placement criteria (Chien and Wu, 1999; Gehring et al., 1990; Haessler and Talbot, 1990; Lim et al., 2003; Liu et al., 2011b). The remaining five use dynamic selection of the next box or space (Burke

et al., 2012; Lim and Zhang, 2005; Ngoi et al., 1994; Scheithauer, 1992), or a tree search to make the exploration less greedy (Chien and Wu, 1998). Nineteen papers augment the placement heuristic with an improvement heuristic, or design an improvement heuristic that directly tackles the problem. Of these nineteen eight use genetic algorithm (Bortfeldt and Gehring, 2001; Gehring and Bortfeldt, 1997, 2002; Gonçalves and Resende, 2012; Hasni and Sabri, 2013; Liang et al., 2007; Lin et al., 1993; Yeh et al., 2003), one uses tabu search (Liu et al., 2011a) and one uses simulated annealing (Egeblad and Pisinger, 2009), with the rest adopting various local search techniques (Fanslau and Bortfeldt, 2010; He and Huang, 2011, 2012; Huang and He, 2009a, 2009b; Lim et al., 2005; Parreño et al., 2008, 2010; Pisinger, 2002). Finally the following four papers develope exact methods for SKP, Fekete et al. (2007), Junqueira et al. (2012a, 2012b), Padberg (2002).

7.2. The Multiple Container Packing Problem

To load multiple containers, three strategies are usually suggested: sequential strategy, pre-assignment strategy and simultaneous strategy (Eley, 2003). Under a sequential strategy (de Castro Silva et al., 2003; Thapatsuwan et al., 2012; Wu et al., 2010; Xue and Lai, 1997), containers are filled one after the other. A new container is opened once the current containers can not store any of the available boxes. However, one of the disadvantage of this strategy is that containers filled last are at risk of poor volume utilization ratios since large or awkwardly formed boxes are often left to the end. The pre-assignment strategy first distributes boxes among different containers. No actual packing occurs at the distribution stage. A heuristic is then applied to load assigned boxes into each container. Boxes not be packed in their designated containers are re-packed to other containers under additional mechanisms. Jin et al. (2003) is a typical example of adopting this strategy. The simultaneous strategy adopted by Chen et al. (1995), de Almeida and Figueiredo (2010), and Soak et al. (2008) attempts to assign and pack boxes into more than one container at a time. It is surprising to learn that the sequential strategy outperformed the simultaneous strategy in Eley's 2002 approach. A few exceptions are Che et al. (2011a), Crainic et al. (2009) and Farøe et al. (2003) who all apply mixed strategies. Che et al. (2011a) firstly pack boxes into packing patterns and then load one container at a time based on these patterns. Crainic et al. (2009) build initial solution under a sequential strategy but apply improvement phase using pre-assignment strategy. Farøe et al. (2003) construct an initial solution by firstly generating a number of walls before combining them into whole bins.

Seven papers address SSSCSP. Among them, Bischoff and Ratcliff (1995) and Ivancic et al. (1989) only propose placement heuristic, while Che et al. (2011a, 2011b), Eley (2002) and Kang et al. (2010) design improvement heuristic. Sciomachen and Tanfani (2007) is an exceptional paper which determines container stowage plans in a ship by comparing its own problem to SSSCSP in 3D packing. We reviewed fourteen SBSBPP papers. Most paper in this category only focus on the construction of placement heuristic

(Amossen and Pisinger, 2010; Burke et al., 2012; Crainic et al., 2008; de Castro Silva et al., 2003; Epstein and Levy, 2010; Lim and Zhang, 2005; Martello et al., 2000; Miyazawa and Wakabayashi, 2009). Among those proposed improvement heuristic, tabu search is surprisingly chosen by most papers (Crainic et al., 2009; Jin et al., 2003; Lodi et al., 2002, 2004), while Farøe et al. (2003) propose a guided local search. Hifi et al. (2010) is the only paper to use exact methods although these are embedded within a broader heuristic, while Boschetti (2004) suggests a new lower bound. Among MSSCSP papers, Ivancic et al. (1989) only give a placement heuristic while others (Brunetta and Gregoire, 2005; Che et al., 2011a, 2011b; Eley, 2003) all design improvement heuristic. All four MBSBPP papers (Brunetta and Gregoire, 2005; Ceschia and Schaerf, 2011; de Almeida and Figueiredo, 2010; Jin et al., 2003) contain improvement heuristic. Chen et al. (1995) claim their paper can solve all problem types including RCSP. However, no algorithm is proposed to tackle specifically RCSP. Two papers (de Almeida and Figueiredo, 2010; Jin et al., 2003) in our collection solve RBPP. Similar to RCSP, Hifi's (2002) algorithm is capable of solving MILOPP but it is a problem type remaining almost untackled. MIKP is the third problem type remaining unexplored. Eley (2003) and Mohanty et al. (1994) are the only two papers we found solving MHLOPP. The paper of Ceschia and Schaerf (2011) also solves MHKP.

8. Experimental Results

One aim of this paper is to provide a comparison of the various algorithms described above based on their performance on benchmark data sets, in an attempt to identify the state-of-the-art algorithms for this class of problems. To our knowledge, five classes of benchmark data sets exist for 3D loading problems. These are:

1. Ivancic et al. (1989): This data set contains 47 instances of weakly heterogeneous boxes. Only one container size is assigned to each instance, but different container sizes are assigned across different instances. The number of containers should be kept as small as possible as is typical in the Single Stock-Size Cutting Stock Problem (SSSCSP).
2. Loh and Nee (1992): This data set contains weakly heterogeneous boxes and a single container in each of its 15 problem instances. Boxes may be left over if the single container is full. This dataset is designed for the Single Large Object Placement Problem (SLOPP).
3. Bischoff and Ratcliff (1995): This set is for the Single Large Object Placement Problem (SLOPP) which packs weakly heterogeneous boxes into a single container. There are seven classes in total with each class including 100 instances.
4. Davies and Bischoff (1999): This set pertains to the Single Knapsack Problem (SKP) which packs strongly heterogeneous boxes into a single container. There are eight classes in total with each class

including 100 problem instances. This set is usually merged with Bischoff and Ratcliff (1995) to become a so-called BR1-15 dataset.

5. Martello et al. (2002): This set is for a typical Single Bin-Size Bin Packing Problem (SBSBPP) which packs strongly heterogeneous boxes into a minimum number of identical containers. There are eight classes of problem that are randomly generated according to certain parameters, but the second and the third classes are not included in the result tables due to their similarity to the first class.

In the rest of the section we provide comparisons of all algorithms published in papers listed in Table 2, which also presents the running environment of each algorithm. Then, we present comparison results for each data set. In particular, Table 3 is for the data set by Ivancic et al. (1989) and Table 4 is for the data set by Loh and Nee (1992). Tables 5 and 6 jointly present the data sets by Bischoff and Ratcliff (1995) and Davies and Bischoff (1999). Finally, Table 7 presents the comparison results for the instances by Martello et al. (2002).

Table 2 lists all the papers whose algorithms we compare in this section. Papers are listed by year of publication and we assign a code that contains the first letter of the author's name and year of publication. In cases where more than one method is presented in the same paper, extra letters, usually short for the methods (and often assigned by the authors themselves), are added to the end. For example, two methods are presented in Lodi et al. (2002). We therefore include both methods in the table under names of L2002-HA and L2002-TS. Some authors present two sets of results, one that takes account of box stability (S), and another where stability is ignored (U). In these cases we add letter S for when box stability is included, and letter U when it is not. An example would be Fanslau and Bortfeldt (2010) as we differentiate them as FB2010S and FB2010U. Gehring and Bortfeldt (1997), as well as Bortfeldt and Gehring (1998), re-ran their algorithm after the publication of their original papers using a more powerful PC. In this case the new codes GB1997N and BG1998N are assigned. The last two algorithms are separated in the table because they are lower bound generators. In Table 3 the number of box types and total number of boxes are provided for each of the 47 instances. For the 13 methods compared, results are presented in term of the number of containers used. In some cases average running times are reported. A lower bound is also presented in the last column of the table. Table 4 contains 15 problem instances and the number of boxes is given for each instance. Overall, the results of 17 algorithms are presented. While 16 results are shown by the percentage utilisation rate in the columns of U%, the original Loh and Nee (1992) paper uses packing density (D%) to present their results. Instead of using the original three container dimensions, the maximum used container dimensions are chosen. The number of boxes leftover (BL), running time in seconds indicated as rt(s) and stopping time in seconds are presented for each method, if they are reported by the original authors. Data set Bischoff and Ratcliff (1995) (reported as BR1-7) and Davies and Bischoff

(1999) (generally reported as BR8-15) are combined making it a total of 15 data sets, namely BR1-15. Tables 5 only includes those results taking stability into consideration while Tables 6 consists of results without taking stability into account. Numbers of box types are given to each class for both tables. The results are presented as percentage utilisation. Running times and stopping time are also presented if they are available. It is worth pointing out that in Table 5 although all algorithms attempt to take stability into account, the levels of stability are still different and difficult to identify. While some algorithm have the base of all boxes fully supported, others permit a certain percentage of the base to be supported. Therefore, inferior results don't necessarily mean the algorithm is less competitive. Bin dimension and number of items are presented for each class in Table 7. Results are in the form of the number of bins used, and stopping time is presented if available. A lower bound is also presented in the last column of this table. All tables show clear improvements on results along the time. Partly benefiting from the development of computer technology, it proves that more competitive algorithms are being developed through all these years.

Year	Paper	Code	Running Environment
1989	Ivancic et al., 1989	IMM1989	IBM PC System 2, model 50
1990	Bischoff and Marriott, 1990	BM1990	(not stated)
1992	Loh and Nee, 1992	LN1992	HP9000 Series 300 workstation with the CPU rated at 5 MIPS
1994	Ngoi et al., 1994	N1994	80386 processor with 33 MHz clock speed
1995	Bischoff and Ratcliff, 1995	BR1995a	(not stated)
1995	Bischoff and Ratcliff, 1995	BR1995b	(not stated)
1995	Bischoff and Ratcliff, 1995	BR1995C	(not stated)
1997	Gehring and Bortfeldt, 1997	GB1997	Pentium/130 MHz PC
1997	Gehring and Bortfeldt, 1997	GB1997N	Pentium PC with a frequency of 400 MHz
1998	Bortfeldt and Gehring, 1998	BG1998	Pentium PC with a frequency of 200 MHz
1998	Bortfeldt and Gehring, 1998	BG1998N	Pentium PC with a frequency of 400 MHz
1998	Chua et al., 1998	C1998	DOS platform; IBM PC
2000	Bortfeldt, 2000	B2000	Pentium PC (200 MHz)
2000	Martello et al., 2000	MPV2000	HP9000/C160 160 MHz
2000	Martello et al., 2000	MPV2000-BS	HP9000/C160 160 MHz
2000	Martello et al., 2000	MPV2000-SPack	HP9000/C160 160 MHz
2001	Bortfeldt and Gehring, 2001	BG2001	Pentium PC with a frequency of 400 MHz
2002	Eley, 2002	E2002-seq	Pentium C with 200 MHz clock and 32MB memory
2002	Eley, 2002	E2002-sim	Pentium C with 200 MHz clock and 32MB memory
2002	Eley, 2002	E2002	Pentium C with 200 MHz clock and 32MB memory
2002	Gehring and Bortfeldt, 2002	GB2002	slave on Pentium PC, 400 MHz; master on 486 PC, 33 MHz
2002	Lodi et al., 2002	L2002-HA	Digital 500 workstation with a 500 MHz CPU
2002	Lodi et al., 2002	L2002-TS	Digital Alpha 533MHz
2003	Bortfeldt et al., 2003	B2003	Pentium with a frequency of 2 GHz
2003	Bortfeldt et al., 2003	B2003PS	Pentium with a frequency of 2 GHz
2003	Eley, 2003	E2003	Petium II PC with a 266 MHz clock and 192MB memory
2003	Farøe et al., 2003	F2003	Digital 500 workstation with a 500 MHz CPU
2003	Lim et al., 2003	L2003	alpha 600 MHz machine
2004	Mack et al., 2004	M2004	several Intel Pentium computers with a frequency of 2GHz combined in a LAN
2005	Lim et al., 2005	L2005	(not stated)
2005	Lim and Zhang, 2005	LZ2005	2.40 GHz Pentium 4 PC with 128 MB memory limit
2005	Moura and Oliveira, 2005	MO2005	Pentium IV at 2.4GHz with 480Mb RAM
2006	Bischoff, 2006	B2006	1.7 GHz Pentium 4 computer with 256 Mb of RAM
2006	Takahara, 2006	T2006	Pentium4 PC with 2.8GHz and 1GB memory
2008	Takahara, 2008	T2008	Xeon PC with 3.0GHz and 3GB memory
2008	Parreño et al., 2008	P2008	Pentium Mobile at 1,500MHz with 512MB RAM
2008	Wang et al., 2008	W2008	Pentium PC
2008	Crainic et al., 2008	C2008	Pentium4 2000 MHz CPU
2009	Crainic et al., 2009	C2009	Pentium4 2000 Mhz CPU
2010	Fanslau and Bortfeldt, 2010	FB2010S	AMD Athlon processor (64 FX60, 2.6GHz) with 2,048MB RAM
2010	Fanslau and Bortfeldt, 2010	FB2010U	AMD Athlon processor (64 FX60, 2.6GHz) with 2,048MB RAM
2010	He and Huang, 2010	HH2010	Intel(R) Xeon(R) 2.33 GHz and 1 GB RAM
2010	Parreño et al., 2010	P2010	Pentium Mobile @ 1,500MHz with 512MB RAM
2011	Che et al., 2011a	C2011	Intel Xeon E5430 CPU @ 2.66 GHz, 8GB RAM
2011	Dereli and Das, 2011	DD2011	(not stated)
2011	He and Huang, 2011	HH2011	Intel(R) Xeon(R) 2.33 GHz
2011	Liu et al., 2011a	L2011	Intel Centrino Duo CPU 1.66GHz and RAM 1GB
2011	Ren et al., 2011	R2011	Intel Core 2 U9300 PC (1.2GHz, 2GB RAM)
2012	Gonçalves and Resende, 2012	GR2012S	AMD 2.2 GHz Opteron 6-core CPU
2012	Gonçalves and Resende, 2012	GR2012U	AMD 2.2 GHz Opteron 6-core CPU
2012	Lim et al., 2012	L2012	2.40GHz Pentium 4 pc with 512MB memory limit
2012	Zhang et al., 2012	Z2012S	Intel(R) Xeon(R) X5460 @ 3.16GHz
2012	Zhang et al., 2012	Z2012U	Intel(R) Xeon(R) X5460 @ 3.16GHz
2012	Zhu and Lim, 2012	ZL2012S	Intel Xeon E5520 Quad-Core CPUs @ 2.27GHz with 8G RAM
2012	Zhu and Lim, 2012	ZL2012U	Intel Xeon E5520 Quad-Core CPUs @ 2.27GHz with 8G RAM
2012	Zhu et al., 2012a	ZE2012aS	Intel Xeon E5520 CPU @ 2.26GHz, 8GB RAM
2012	Zhu et al., 2012a	ZE2012aU	Intel Xeon E5520 CPU @ 2.26GHz, 8GB RAM
2012	Zhu et al., 2012b	ZE2012b	Intel Xeon E5520 Quad-Core CPUs @ 2.27 GHz
2014	Araya and Riff, 2014	AR2014S	Intel Xeon @ 2.20GHz and 8GB RAM with 2 quad-processors
2014	Araya and Riff, 2014	AR2014U	Intel Xeon @ 2.20GHz and 8GB RAM with 2 quad-processors
2002	lower bound (Eley, 2002)	lbE2002	Pentium C with 200 MHz clock and 32MB memory
2004	lower bound (Boschetti, 2004)	lbB2004	Pentium III Intel 933 MHz

Table 2: Lists of Papers included in the Results Comparison

No.	Box Types	Total Boxes No.	IMM1989	BR1995b	B2000	E2002-seq	E2002-sim	E2003	LZ2005	T2006	T2008	C2011	L2012	ZE2012aU	ZE2012aS	lbE2002
1	2	70	26	27	25	27	26	25	25	25	25	25	25	25	25	19
2	2	70	11	11	10	11	10	10	10	10	10	10	10	10	10	7
3	4	180	20	26	20	21	22	20	19	20	20	19	19	19	19	19
4	4	180	27	27	28	29	30	26	26	26	26	26	26	26	26	26
5	4	180	65	59	51	55	51	51	51	51	51	51	51	51	51	46
6	3	103	10	10	10	10	10	10	10	10	10	10	10	10	10	10
7	3	103	16	16	16	16	16	16	16	16	16	16	16	16	16	16
8	3	103	5	4	4	4	4	4	4	4	4	4	4	4	4	4
9	2	110	19	19	19	19	19	19	19	19	19	19	19	19	19	16
10	2	110	55	55	55	55	55	55	55	55	55	55	55	55	55	37
11	2	110	18	25	18	17	18	17	16	16	16	16	16	16	17	14
12	3	95	55	55	53	53	53	53	53	53	53	53	53	53	53	45
13	3	95	27	27	25	25	25	25	25	25	25	25	25	25	25	20
14	3	95	28	28	28	27	27	27	27	27	27	27	27	27	27	27
15	3	95	11	15	11	12	12	11	11	11	11	11	11	11	11	11
16	3	95	34	29	26	28	26	26	26	26	26	26	26	26	26	21
17	3	95	8	10	7	8	7	7	7	7	7	7	7	7	7	7
18	3	47	3	2	2	1	1	2	2	2	2	2	2	2	2	1
19	3	47	3	3	3	2	2	3	3	3	3	3	3	3	3	1
20	3	47	5	5	5	2	2	5	5	5	5	5	5	5	5	2
21	5	95	24	26	21	24	26	20	20	20	20	20	20	20	20	17
22	5	95	10	11	9	9	9	8	9	9	9	8	9	8	8	8
23	5	95	21	22	20	21	21	20	20	20	20	19	20	19	20	17
24	4	72	6	7	6	6	6	6	5	5	5	5	5	5	5	5
25	4	72	6	5	5	6	5	5	5	5	5	5	5	5	5	4
26	4	72	3	4	3	3	3	3	3	3	3	3	3	3	3	3
27	3	95	5	5	5	5	5	5	5	5	5	5	5	4	4	4
28	3	95	10	12	10	11	10	10	9	10	10	10	9	10	10	9
29	4	118	18	23	17	18	18	17	17	17	17	17	17	17	17	15
30	4	118	24	26	22	22	23	22	22	22	22	22	22	22	22	18
31	4	118	13	14	13	13	14	13	12	13	13	12	12	12	12	11
32	3	90	5	4	4	4	4	4	4	4	4	4	4	4	4	4
33	3	90	5	5	5	5	5	5	4	5	5	4	4	4	4	4
34	3	90	9	8	8	5	9	8	8	8	8	8	8	8	8	5
35	2	84	3	3	2	2	2	2	2	2	2	2	2	2	2	2
36	2	84	18	14	14	18	14	14	14	14	14	14	14	14	14	10
37	3	102	26	23	23	26	23	23	23	23	23	23	23	23	23	12
38	3	102	50	45	45	46	45	45	45	45	45	45	45	45	45	25
39	3	102	16	18	15	15	15	15	15	15	15	15	15	15	15	12
40	4	85	9	11	9	9	9	8	9	9	9	8	9	8	8	7
41	4	85	16	17	15	16	15	15	15	15	15	15	15	15	15	14
42	3	90	4	5	4	4	4	4	4	4	4	4	4	4	4	4
43	3	90	3	3	3	3	3	3	3	3	3	3	3	3	3	3
44	3	90	4	4	3	4	4	4	3	3	3	3	3	3	3	3
45	4	99	3	3	3	3	3	3	3	3	3	3	3	3	3	2
46	4	99	2	2	2	2	2	2	2	2	2	2	2	2	2	2
47	4	99	4	4	3	3	3	3	3	3	3	3	3	3	3	3
Total			763	777	705	725	716	699	694	698	698	692	694	691	693	572
Avg. Time (s)								<30	6.43	<2		45.94	6.43	246.2	116.7	

Table 3: Ivancic et al., 1989

		LN1992		BM1990	N1994		BR1995b		GB1997		BG1998		C1998	BG2001		E2002		B2003	L2005	MO2005			W2008		HH2010	DD2011	R2011			L2012
Dataset	Box No.	BL	D%	U%	BL	U%	BL	U%	BL	U%	BL	U%	U%	BL	U%	BL	U%	U%	U%	rt(s)	BL	U%	BL	U%	U%	U%	rt(s)	BL	U%	U%
LN01	100	0	78.12	62.5	0	62.5	0	62.5	0	62.5	0	62.5	62.5	0	62.5	0	62.5	n/a	62.5	28	0	62.5	0	62.5	62.5	62.5	7	0	62.5	62.5
LN02	200	32	76.77	90	54	80.7	35	90	39	89.5	28	96.6	76.02	51	89.8	53	90.8		80.4	45	19	92.6	35	90.7	97.9	86.3	332	25	97.9	96.4
LN03	200	0	69.46	53.4	0	53.4	0	53.4	0	53.4	0	53.4	53.43	0	53.4	0	53.4		53.4	105	0	53.4	0	53.4	53.4	53.4	18	0	53.4	53.4
LN04	100	0	77.57	55	0	55	0	55	0	55	0	55	54.96	0	55	0	55		55	54	0	55	0	55	55	55	55	0	55	55
LN05	120	1	85.79	77.2	0	77.2	0	77.2	0	77.2	0	77.2	77.19	0	77.2	0	77.2		76.7	16	0	77.2	0	77.2	77.2	77.2	32	0	77.2	77.2
LN06	200	45	88.55	83.1	48	88.7	77	83.1	32	91.1	49	91.2	79.51	45	92.4	44	87.9		84.8	34	28	91.7	37	92.9	96.7	89.2	167	21	96.3	93.5
LN07	200	21	78.17	78.7	10	81.8	18	78.7	7	83.3	0	84.7	72.14	0	84.7	0	84.7		77	56	0	84.7	0	84.7	84.7	83.2	29	0	84.7	84.7
LN08	130	7	67.58	59.4	0	59.4	0	59.4	0	59.4	0	59.4	59.42	0	59.4	0	59.4		59.4	20	0	59.4	0	59.4	59.4	59.4	6	0	59.4	59.4
LN09	200	0	84.22	61.9	0	61.9	0	61.9	0	61.9	0	61.9	61.89	0	61.9	0	61.9		61.9	78	0	61.9	0	61.9	61.9	61.9	14	0	61.9	61.9
LN10	250	0	70.1	67.3	0	67.3	0	67.3	0	67.3	0	67.3	67.29	0	67.3	0	67.3		67.3	54	0	67.3	0	67.3	67.3	67.3	24	0	67.3	67.3
LN11	100	0	65.44	62.2	0	62.2	0	62.2	0	62.2	0	62.2	62.16	0	62.2	0	62.2		62.2	34	0	62.2	0	62.2	62.2	62.2	3	0	62.2	62.2
LN12	120	0	79.33	78.5	0	78.5	0	78.5	0	78.5	0	78.5	71	0	78.5	0	78.5		69.5	57	0	78.5	0	78.5	78.5	78.5	43	0	78.5	78.5
LN13	130	15	77.03	78.1	2	84.1	20	78.1	0	85.6	4	84.3	84.14	0	85.6	0	85.6		73.3	68	0	85.6	0	85.6	85.6	85.6	61	0	85.6	84.9
LN14	120	0	69.09	62.8	0	62.8	0	62.8	0	62.8	0	62.8	62.81	0	62.8	0	62.8		62.8	54	0	62.8	0	62.8	62.8	62.8	10	0	62.8	62.8
LN15	250	0	73.56	59.5	0	59.5	0	59.5	0	59.5	0	59.5	59.46	0	59.5	0	59.5		59.5	42	0	59.5	0	59.5	59.5	59.5	19	0	59.5	59.5
Average			74.19	68.64		69		68.6		69.9		70.4	66.93		70.15		69.91	70.9	67	49.7		70.3		70.2	71	69.55			70.93	70.6
Avg. Time (s)																								<20			51			39.88
Stopping Time (s)										500		500						600												

Table 4: Loh and Nee, 1992

		BG1998N	BG2001	E2002	GB2002	MO2005		B2006		FB2010S	L2011	R2011		GR2012S	Z2012S	ZE2012aS	ZL2012S	AR2014S
Dataset	Box Types	U%	U%	U%	U%	rt(s)	U%	rt(s)	U%	U%	U%	rt(s)	U%	U%	U%	U%	U%	U%
BR1	3	92.63	87.81	n/a	88.10	8	89.07	>26.4	90.57	94.51	88.14	114	93.90	94.34	94.43	93.57	94.40	94.50
BR2	5	92.70	89.40		89.56	12	90.43		90.84	94.73	89.52	145	94.54	94.88	94.87	93.87	94.85	95.03
BR3	8	92.31	90.48		90.77	25	90.86		91.43	94.74	90.53	213	94.35	95.05	95.06	94.14	95.10	95.17
BR4	10	91.62	90.63		91.03	28	90.42		92.21	94.41	90.75	308	94.08	94.75	94.89	93.86	94.81	94.97
BR5	12	90.86	90.73		91.23	40	89.57		91.25	94.13	90.79	413	94.17	94.58	94.68	93.51	94.52	94.80
BR6	15	90.04	90.72		91.28	59	89.71		91.04	93.85	90.74	500	93.48	94.39	94.53	93.39	94.33	94.65
BR7	20	88.63	90.65		91.04	64	88.05	>321	90.81	93.20	90.07	663	92.82	93.74	93.96	92.68	93.59	94.09
BR8	30	87.11	89.73		90.26	71	86.13			92.26	88.89			92.65	93.27		92.65	93.15
BR9	40	85.76	89.06		89.50	85	85.08			91.48	88.51			91.90	92.60		92.11	92.53
BR10	50	84.73	88.40		88.73	89	84.21			90.86	87.76			91.28	92.05		91.60	92.04
BR11	60	83.55	87.53		87.87	94	83.98			90.11	87.06			90.39	91.46		90.64	91.40
BR12	70	82.79	86.94		87.18	98	83.64			89.51	86.97			89.81	90.91		90.35	90.92
BR13	80	82.29	86.25		86.70	110	83.54			88.98	86.90			89.27	90.43		89.69	90.51
BR14	90	81.33	85.55		85.81	121	83.25			88.26	86.40			88.57	89.80		89.07	89.93
BR15	100	80.85	85.23		85.48	128	83.21			87.57	86.23			87.96	89.24		88.36	89.33
Average 1-7		91.26	90.06	88.75	90.43		89.73		91.16	94.20	90.08		93.91	94.53	94.63	93.57	94.51	94.74
Average 8-15		83.55	87.34		87.69		84.13			89.88	87.34			90.23	91.22		90.56	91.22
Average		87.15	88.61		89		86.74			91.90	88.62			92.24	92.81		92.40	92.87
Avg. Time 1-7 (s)						33.71				320		337						
Avg. Time (s)						68.8				320				232			150.55	
Stopping Time (s)		500			250					60+180					500		150	150

Table 5: Bischoff and Ratcliff (1995) and Davies and Bischoff (1999) with Stability Consideration

		BR1995C	GB1997N	B2003		B2003par		L2003	M2004	L2005	P2008		FB2010U	HH2010	P2010	DD2011		HH2011	L2012	GR2012U	Z2012U	ZE2012aU	ZE2012b	ZL2012U	AR2014U
Dataset	Box Types	U%	U%	rt(s)	U%	rt(s)	U%	U%	U%	U%	rt(s)	U%	U%	U%	U%	rt(s)	U%	U%	U%	U%	U%	U%	U%	U%	U%	U%
BR1	3	83.37	86.77	3	93.23	36	93.52	88.70	93.70	87.40	1.27	93.27	95.05	92.32	94.93	1.16	83.41	92.92	91.60	95.28	94.92	94.16	95.54	95.59	95.69
BR2	5	83.57	88.12	10	93.27	48	93.77	88.17	94.30	88.70	2.32	93.38	95.43	92.71	95.19	2.54	84.60	93.93	91.99	95.90	95.48	94.50	95.98	96.13	96.24
BR3	8	83.59	88.87	31	92.86	97	93.58	87.52	94.54	89.30	4.62	93.39	95.47	93.25	94.99	5.14	85.42	93.71	92.30	96.13	95.69	94.71	96.08	96.30	96.49
BR4	10	84.16	88.68	48	92.40	138	93.05	87.58	94.27	89.70	6.52	93.16	95.18	93.15	94.71	7.66	85.19	93.68	92.36	96.01	95.53	94.55	95.94	96.15	96.31
BR5	12	83.89	88.78	65	91.61	179	92.34	87.30	93.83	89.70	8.58	92.89	95.00	92.85	94.33	10.38	85.11	93.73	91.90	95.84	95.44	94.20	95.74	95.98	96.18
BR6	15	82.92	88.53	46	90.86	150	91.72	86.86	93.34	89.70	12.23	92.62	94.79	92.88	94.04	16.66	84.69	93.63	91.50	95.72	95.38	94.15	95.61	95.81	96.05
BR7	20	82.14	88.36	60	89.65	198	90.55	87.15	92.50	89.40	19.25	91.86	94.24	92.66	93.53	29.54	83.99	93.14	91.01	95.29	95.00	93.74	95.14	95.36	95.77
BR8	30	80.10	87.52								38.2	91.02	93.70	91.99	92.78	82.94		92.92		94.76	94.66		94.63	94.80	95.33
BR9	40	78.03	86.46								63.1	90.46	93.44	91.76	92.19	160.77		92.49		94.34	94.30		94.29	94.53	95.07
BR10	50	76.53	85.53								97.08	89.87	93.09	91.41	91.92	298.95		92.24		93.86	94.11		94.05	94.35	94.97
BR11	60	75.08	84.82								136.5	89.36	92.81	91.10	91.46	497.79		91.91		93.60	93.87		93.78	94.14	94.80
BR12	70	74.37	84.25								183.21	89.03	92.73	90.75	91.20	861.37		91.83		93.22	93.67		93.67	94.10	94.64
BR13	80	73.56	83.67								239.8	88.56	92.46	90.33	91.11	1775.79		91.56		92.99	93.45		93.54	93.86	94.59
BR14	90	73.37	82.99								307.62	88.46	92.40	90.04	90.64	2218.17		91.30		92.68	93.34		93.36	93.83	94.49
BR15	100	73.38	82.47								394.66	88.36	92.40	89.51	90.38	3531.71		91.02		92.46	93.14		93.32	93.78	94.37
Average 1-7		83.38	88.30		92.00		92.70	87.60	93.78	89.13		92.94	95.00	92.83	94.53		84.63	93.53	91.81	95.74	95.35	94.29	95.72	95.90	96.11
Average 8-15		75.55	84.71									89.39	92.88	90.86	91.46			91.91		93.49	93.82		93.83	94.17	94.78
Average		79.20	86.39									91.05	93.90	91.78	92.89			92.67		94.54	94.53		94.71	94.98	95.40
Avg. Time 1-7 (s)				38		121			222		7.83		319		28	10.44			706.59						
Avg. Time (s)											101		320	4661	296	633.37				147				503.45	
Stopping Time (s)			500								5000 iter		60+180								500		500	500	500

Table 6: Bischoff and Ratcliff (1995) and Davies and Bischoff (1999) without Stability Consideration

9. Conclusion

In this paper we have focused our review on the design and implementation of solution methodologies for solving 3D container loading problem with an experimental comparison on the performances of various algorithms on benchmark data sets. We have reviewed 113 papers that cover all variants of 3D container loading and address a wide spectrum of combinatorial optimisation methodologies, which we review according to placement heuristics, improvement heuristics and exact methods. We have attempted to provide an insightful review that describes the main ideas clearly. These sections do not differentiate between problem types, so we provide an overview of these papers by problem variant. Our examination of the literature has highlighted three possible fruitful avenues for research. First we found that there are significantly fewer papers examining the multiple container packing problem when compared to the single container loading problem. Further, of the papers looking at multiple containers, very few consider heterogeneous container

Class	Bin Dimensions	Item No.	MPV2000	MPV2000-BS	MPV2000-SPack	L2002-HA	L2002-TS	F2003	C2008	C2009	lbB2004
1	100*100	50	13.6	13.5	15.3	13.9	13.4	13.4	13.7	13.4	12.9
		100	27.3	29.5	27.4	27.6	26.6	26.7	27.2	26.7	25.6
		150	38.2	38	40.4	38.1	36.7	37	37.7	37	35.8
		200	52.3	52.3	55.6	52.7	51.2	51.2	51.9	51.1	49.7
Class Total			131.4	133.3	138.7	132.3	127.9	128.3	130.5	128.2	124
4	100*100	50	29.4	29.4	29.8	29.4	29.4	29.4	29.4	29.4	29
		100	59.1	59	60	59	59	59	59	58.9	58.5
		150	87.2	87.3	87.9	86.9	86.8	86.8	86.8	86.8	86.4
		200	119.5	119.3	120.3	119	118.8	119	118.8	118.8	118.3
Class Total			295.2	295	298	294.3	294	294.2	294	293.9	292.2
5	100*100	50	9.2	9.1	10.2	8.5	8.4	8.3	8.4	8.3	7.6
		100	17.5	17	17.6	15.1	15	15.1	15.1	15.2	14
		150	24	23.7	24	21.4	20.4	20.2	21	20.1	18.8
		200	31.8	31.7	31.7	28.6	27.6	27.2	28.1	27.4	26
Class Total			82.5	81.5	83.5	73.6	71.4	70.8	72.6	71	66.4
6	10*10	50	9.8	11	11.2	10.5	9.9	9.8	10.1	9.8	9.4
		100	19.4	22.3	24.5	20	19.1	19.1	19.6	19.1	18.4
		150	29.6	32.4	35	30.6	29.4	29.4	29.9	29.2	28.5
		200	38.2	40.8	42.3	39.1	37.7	37.7	38.5	37.7	36.7
Class Total			97	106.5	113	100.2	96.1	96	98.1	95.8	93
7	40*40	50	8.2	8.2	9.3	8	7.5	7.4	7.5	7.4	6.8
		100	15.3	13.9	15.3	13.3	12.5	12.3	13.2	12.3	11.5
		150	19.7	18.1	20.1	17.2	16.1	15.8	17	15.8	14.4
		200	28.1	28	28.7	25.2	23.9	23.5	25.1	23.5	22.7
Class Total			71.3	68.2	73.4	63.7	60	59	62.8	59	55.4
8	100*100	50	10.1	9.9	11.3	9.9	9.3	9.2	9.4	9.2	8.7
		100	20.2	20.2	21.7	19.9	18.9	18.9	19.5	18.8	18.4
		150	27.3	26.8	28.3	25.7	24.1	23.9	25.2	23.9	22.5
		200	34.9	34	35	31.6	30.3	29.9	31.3	30	28.2
Class Total			92.5	90.9	96.3	87.1	82.6	81.9	85.4	81.9	77.8
Total			769.9	775.4	802.9	751.2	732	730.2	743.4	729.8	708.8
Stopping Time (s)			1000				1000	1000			

Table 7: Martello et al. 2000

problem types. Another observation is that often papers that tackle the multi container problem do so by simply extending their models for the single container loading problem. A second research issue surrounds the consideration of real world constraints. Only a small proportion of papers comprehensively address this, while most other papers focus on a few specific constraints. Even within those papers discussing real world constraints, not all critical constraints are taken into consideration. A consistent set of real world constraints would benefit the research community and make adoption by practitioners more likely. This links with the final research issue regarding the quantity and quality of benchmark datasets. As Bortfeldt and Wäscher (2013) point out, the current decade-old data sets might not be challenging anymore. Realistic and challenging data sets with clear real world constraints would help in moving this research area forward.

References

Allen, S.D., Burke, E.K., Kendall, G., 2011. A hybrid placement strategy for the three-dimensional strip packing problem. European Journal of Operational Research 209, 219-227.

Allen, S.D., Burke, E.K., Mareček, J., 2012. A space-indexed formulation of packing boxes into a larger box. Operations Research Letters 40, 20-24.

Amossen, R.R., Pisinger, D., 2010. Multi-dimensional bin packing problems with guillotine constraints. Computers & Operations Research 37, 1999-2006.

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

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

Bischoff, E.E., Marriott, M.D., 1990. A comparative evaluation of heuristics for container loading. European Journal of Operational Research 44, 267-276.

Bischoff, E.E., Ratcliff, M.S.W., 1995. Issues in the development of approaches to container loading. Omega 23, 377-390.

Bortfeldt, A., 2000. Eine Heuristik für Multiple Containerladeprobleme. OR Spektrum 22, 239-261.

Bortfeldt, A., Gehring, H., 1998. Applying Tabu Search to Container Loading Problems. Fernuniversität Hagen - Lehrstuhl für Wirtschaftsinformatik, Prof. Gehring - Publikationen; In: Operations Research Proceedings, 1997, Springer Verlag, Berlin u.a. 1998, S. 533-538.

Bortfeldt, A., Gehring, H., 2001. A hybrid genetic algorithm for the container loading problem. European Journal of Operational Research 131, 143-161.

Bortfeldt, A., Gehring, H., Mack, D., 2003. A parallel tabu search algorithm for solving the container loading problem. Parallel Computing 29, 641-662.

Bortfeldt, A., Mack, D., 2007. A heuristic for the three-dimensional strip packing problem. European Journal of Operational Research 183, 1267-1279.

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

Boschetti, M.A., 2004. New lower bounds for the three-dimensional finite bin packing problem. Discrete Applied Mathematics 140, 241-258.

Brunetta, L., Grégoire, P., 2005. A General Purpose Algorithm for Three-Dimensional Packing. INFORMS Journal on Computing 17, 328-338.

Burke, E.K., Hyde, M.R., Kendall, G., Woodward, J., 2012. Automating the packing heuristic design process with genetic programming. Evolutionary Computation 20, 63-89.

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

Che, C.H., Huang, W., Lim, A., Zhu, W., 2011a. The multiple container loading cost minimization problem. European Journal of Operational Research 214, 501-511.

Che, C.H., Huang, W., Lim, A., Zhu, W., 2011b. A Heuristic for the Multiple Container Loading Cost Minimization Problem, in: Mehrotra, K.G., Mohan, C.K., Oh, J.C., Varshney, P.K., Ali, M. (Eds.), Modern Approaches in Applied Intelligence, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 276-285.

Chen, C.S., Lee, S.M., Shen, Q.S., 1995. An analytical model for the container loading problem. European Journal of Operational Research 80, 68-76.

Chien, C.-F., Deng, J.-F., 2004. A container packing support system for determining and visualizing container packing patterns. Decision Support Systems 37, 23-34.

Chien, C.-F., Lee, C.-Y., Huang, Y.-C., Wu, W.-T., 2009. An efficient computational procedure for determining the container-loading pattern. Computers & Industrial Engineering 56, 965-978.

Chien, C.-F., Wu, W.-T., 1998. A recursive computational procedure for container loading. Computers & Industrial Engineering 35, 319-322.

Chien, C.-F., Wu, W.-T., 1999. A framework of modularized heuristics for determining the container loading patterns. Computers & Industrial Engineering 37, 339-342.

Christensen, S.G., Rousøe, D.M., 2009. Container loading with multi-drop constraints. International Transactions in Operational Research 16, 727-743.

Chua, C.K., Narayanan, V., Loh, J., 1998. Constraint-based spatial representation technique for the container packing problem. Integrated Manufacturing Systems 9, 23-33.

Crainic, T.G., Perboli, G., Tadei, R., 2008. Extreme Point-Based Heuristics for Three-Dimensional Bin Packing. INFORMS Journal on Computing 20, 368-384.

Crainic, T.G., Perboli, G., Tadei, R., 2009. TS2PACK: A two-level tabu search for the three-dimensional bin packing problem. European Journal of Operational Research 195, 744-760.

Davies, A.P., Bischoff, E.E., 1999. Weight distribution considerations in container loading. European Journal of Operational Research 114, 509-527.

de Almeida, A., Figueiredo, M.B., 2010. A particular approach for the Three-dimensional Packing Problem with additional constraints. Computers & Operations Research 37, 1968-1976.

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

Dereli, T., Das, G.S., 2011. A hybrid bee(s) algorithm for solving container loading problems. Applied Soft Computing 11, 2854-2862.

Egeblad, J., Pisinger, D., 2009. Heuristic approaches for the two- and three-dimensional knapsack packing problem. Computers & Operations Research 36, 1026-1049.

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

Eley, M., 2003. A bottleneck assignment approach to the multiple container loading problem. OR Spectrum 25, 45-60.

Epstein, L., Levy, M., 2010. Dynamic multi-dimensional bin packing. Journal of Discrete Algorithms 8, 356-372.

Faina, L., 2000. A global optimization algorithm for the three-dimensional packing problem. European Journal of Operational Research 126, 340-354.

Fanslau, T., Bortfeldt, A., 2010. A Tree Search Algorithm for Solving the Container Loading Problem. INFORMS Journal on Computing 22, 222-235.

Farøe, O., Pisinger, D., Zachariasen, M., 2003. Guided Local Search for the Three-Dimensional Bin-Packing Problem. INFORMS Journal on Computing 15, 267-283.

Fekete, S.P., Schepers, J., 1997. A new exact algorithm for general orthogonal d-dimensional knapsack problems, in: Burkard, R., Woeginger, G. (Eds.), Algorithms ESA 97, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 144-156.

Fekete, S.P., Schepers, J., 2004. A Combinatorial Characterization of Higher-Dimensional Orthogonal Packing. Mathematics of Operations Research 29, 353-368.

Fekete, S.P., Schepers, J., Veen, J.C. van der, 2007. An Exact Algorithm for Higher-Dimensional Orthogonal Packing. Operations Research 55, 569-587.

Fujiyoshi, K., Kawai, H., Ishihara, K., 2009. A Tree Based Novel Representation for 3D-Block Packing. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 28, 759-764.

Gehring, H., Bortfeldt, A., 1997. A Genetic Algorithm for Solving the Container Loading Problem. International Transactions in Operational Research 4, 401-418.

Gehring, H., Bortfeldt, A., 2002. A Parallel Genetic Algorithm for Solving the Container Loading Problem. International Transactions in Operational Research 9, 497-511.

Gehring, H., Menschner, K., Meyer, M., 1990. A computer-based heuristic for packing pooled shipment containers. European Journal of Operational Research 44, 277-288.

George, J.A., 1992. A Method for Solving Container Packing for a Single Size of Box. The Journal of the Operational Research Society 43, 307-312.

George, J.A., Robinson, D.F., 1980. A heuristic for packing boxes into a container. Computers & Operations Research 7, 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 & Operations Research 39, 179-190.

Gonçalves, J.F., Resende, M.G.C., 2013. A biased random key genetic algorithm for 2D and 3D bin packing problems. International Journal of Production Economics 145, 500-510.

Haessler, R.W., Talbot, F.B., 1990. Load planning for shipments of low density products. European Journal of Operational Research 44, 289-299.

Han, C.P., Knott, K., Egbelu, P.J., 1986. A heuristic approach to the three dimensional cargo loading problem. Computers & Industrial Engineering 11, 109-113.

Hasni, H., Sabri, H., 2013. On a Hybrid Genetic Algorithm for Solving the Container Loading Problem with no Orientation Constraints. Journal of Mathematical Modelling and Algorithms in Operations Research 12, 67-84.

He, K., Huang, W., 2010. A caving degree based flake arrangement approach for the container loading problem. Computers & Industrial Engineering 59, 344-351.

He, K., Huang, W., 2011. An efficient placement heuristic for three-dimensional rectangular packing. Computers & Operations Research 38, 227-233.

He, Y., Wu, Y., de Souza, R., 2012. A global search framework for practical three-dimensional packing with variable carton orientations. Computers & Operations Research 39, 2395-2414.

Hemminki, J., 1994. Container loading with variable strategies in each layer. University of Turku, Institute for Applied Mathematics.

Hifi, M., 2002. Approximate algorithms for the container loading problem. International Transactions in Operational Research 9, 747-774.

Hifi, M., Kacem, I., Négre, S., Wu, L., 2010. A Linear Programming Approach for the Three-Dimensional Bin-Packing Problem. Electronic Notes in Discrete Mathematics 36, 993-1000.

Huang, W., He, K., 2009a. A caving degree approach for the single container loading problem. European Journal of Operational Research 196, 93-101.

Huang, W., He, K., 2009b. A new heuristic algorithm for cuboids packing with no orientation constraints. Computers & Operations Research 36, 425-432.

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

Jin, Z., Ito, T., 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, 60-66.

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

Junqueira, L., Morabito, R., Sato Yamashita, D., 2012b. MIP-based approaches for the container loading problem with multi-drop constraints. Annals of Operations Research 199, 51-75.

Kang, K., Moon, I., Wang, H., 2012. A hybrid genetic algorithm with a new packing strategy for the three-dimensional bin packing problem. Applied Mathematics and Computation 219, 1287-1299.

Kang, M.-K., Jang, C.-S., Yoon, K.-S., 2010. Heuristics with a new block strategy for the single and multiple containers loading problems. Journal of the Operational Research Society 61, 95-107.

Lai, K.K., Chan, J.W.M., 1997. Developing a simulated annealing algorithm for the cutting stock problem. Computers & Industrial Engineering 32, 115-127.

Lai, K.K., Xue, J., Xu, B., 1998. Container packing in a multi-customer delivering operation. Computers & Industrial Engineering 35, 323-326.

Liang, S.-C., Lee, C.-Y., Huang, S.-W., 2007. A hybrid meta-heuristic for the container loading problem. Communications of the International Information Management Association 7, 73-84.

Lim, A., Ma, H., Xu, J., Zhang, X., 2012. An iterated construction approach with dynamic prioritization for solving the container loading problems. Expert Systems with Applications 39, 4292-4305.

Lim, A., Rodrigues, B., Wang, Y., 2003. A multi-faced buildup algorithm for three-dimensional packing problems. Omega 31, 471-481.

Lim, A., Rodrigues, B., Yang, Y., 2005. 3-D Container Packing Heuristics. Applied Intelligence 22, 125-134.

Lim, A., Zhang, X., 2005. The container loading problem, in: Proceedings of the 2005 ACM Symposium on Applied Computing, SAC 05. ACM, New York, NY, USA, pp. 913-917.

Lin, J.-L., Foote, B., Pulat, S., Chang, C.-H., Cheung, J.Y., 1993. Hybrid genetic algorithm for container packing in three dimensions, in: Proceedings of the Ninth Conference on Artificial Intelligence for Applications. IEEE Computer Society Press, pp. 353-359.

Lins, L., Lins, S., Morabito, R., 2002. An n-tet graph approach for non-guillotine packings of n-dimensional boxes into an n-container. European Journal of Operational Research 141, 421-439.

Liu, J., Yue, Y., Dong, Z., Maple, C., Keech, M., 2011a. A novel hybrid tabu search approach to container loading. Computers & Operations Research 38, 797-807.

Liu, W.-Y., Lin, C.-C., Yu, C.-S., 2011b. On the Three-Dimensional Container Packing Problem Under Home Delivery Service. Asia-Pacific Journal of Operational Research 28, 601-621.

Lodi, A., Martello, S., Vigo, D., 2002. Heuristic algorithms for the three-dimensional bin packing problem. European Journal of Operational Research 141, 410-420.

Lodi, A., Martello, S., Vigo, D., 2004. TSpack: A Unified Tabu Search Code for Multi-Dimensional Bin Packing Problems. Annals of Operations Research 131, 203-213.

Loh, T., Nee, A., 1992. A packing algorithm for hexahedral boxes, in: Proceedings of the Conference of Industrial Automation. Singapore, pp. 115-126.

Mack, D., Bortfeldt, A., Gehring, H., 2004. A parallel hybrid local search algorithm for the container loading problem. International Transactions in Operational Research 11, 511-533.

Martello, S., Pisinger, D., Vigo, D., 2000. The Three-Dimensional Bin Packing Problem. Operations Research 48, 256-267.

Miyazawa, F.K., Wakabayashi, Y., 1997. An algorithm for the three-dimensional packing problem with asymptotic performance analysis. Algorithmica 18, 122-144.

Miyazawa, F.K., Wakabayashi, Y., 1999. Approximation Algorithms for the Orthogonal Z-Oriented Three-Dimensional Packing Problem. SIAM Journal on Computing 29, 1008-1029.

Miyazawa, F.K., Wakabayashi, Y., 2009. Three-dimensional packings with rotations. Computers & Operations Research 36, 2801-2815.

Mohanty, B.B., Mathur, K., Ivancic, N.J., 1994. Value considerations in three-dimensional packing A heuristic procedure using the fractional knapsack problem. European Journal of Operational Research 74, 143-151.

Morabito, R., Arenales, M., 1994. An AND/OR-graph approach to the container loading problem. International Transactions in Operational Research 1, 59-73.

Moura, A., Oliveira, J.F., 2005. A GRASP approach to the container-loading problem. IEEE Intelligent Systems 20, 50-57.

Ngoi, B.K.A., Tay, M.L., Chua, E.S., 1994. Applying spatial representation techniques to the container packing problem. International Journal of Production Research 32, 111-123.

Padberg, M., 2000. Packing small boxes into a big box. Mathematical Methods of Operations Research 52, 1-21.

Parreño, F., Alvarez-Valdes, R., Oliveira, J., Tamarit, J., 2010. Neighborhood structures for the container loading problem: a VNS implementation. Journal of Heuristics 16, 1-22.

Parreño, F., Alvarez-Valdes, R., Tamarit, J.M., Oliveira, J.F., 2008. A Maximal-Space Algorithm for the Container Loading Problem. INFORMS Journal on Computing 20, 412-422.

Pisinger, D., 2002. Heuristics for the container loading problem. European Journal of Operational Research 141, 382-392.

Ratcliff, M.S.W., Bischoff, E.E., 1998. Allowing for weight considerations in container loading. OR Spektrum 20, 65-71.

Ren, J., Tian, Y., Sawaragi, T., 2011. A tree search method for the container loading problem with shipment priority. European Journal of Operational Research 214, 526-535.

Scheithauer, G., 1991. A three-dimensional bin packing algorithm. Journal of Information Processing and Cybernetics 27, 263-271.

Scheithauer, G., 1992. Algorithms for the Container Loading Problem, in: Gaul, W., Bachem, A., Habenicht, W. (Eds.). Presented at the Operations Research Proceedings 1991, Springer Berlin Heidelberg, pp. 445-452.

Sciomachen, A., Tanfani, E., 2007. A 3D-BPP approach for optimising stowage plans and terminal productivity. European Journal of Operational Research 183, 1433-1446.

Soak, S.-M., Lee, S.-W., Yeo, G.-T., Jeon, M.-G., 2008. An effective evolutionary algorithm for the multiple container packing problem. Progress in Natural Science 18, 337-344.

Takahara, S., 2006. A Simple Meta-heuristic Approach for the Multiple Container Loading Problem, in: IEEE International Conference on Systems, Man and Cybernetics, 2006. SMC 06. pp. 2328-2333.

Takahara, S., 2008. A Multi-start Local Search Approach to the Multiple Container Loading Problem, in: Bednorz, W. (Ed.), Advances in Greedy Algorithms. IN-TECH, pp. 55-68.

Thapatsuwan, P., Pongcharoen, P., Hicks, C., Chainate, W., 2012. Development of a stochastic optimisation tool for solving the multiple container packing problems. International Journal of Production Economics 140, 737-748.

Wang, Z., Li, K.W., Levy, J.K., 2008. A heuristic for the container loading problem: A tertiary-tree-based dynamic space decomposition approach. European Journal of Operational Research 191, 86-99.

Wäscher, G., Haußner, H., Schumann, H., 2007. An improved typology of cutting and packing problems. European Journal of Operational Research 183, 1109-1130.

Wu, Y., Li, W., Goh, M., de Souza, R., 2010. Three-dimensional bin packing problem with variable bin height. European Journal of Operational Research 202, 347-355.

Xue, J., Lai, K.K., 1997. Effective methods for a container packing operation. Mathematical and Computer Modelling 25, 75-84.

Yeh, J.M., Lin, Y.C., Yi, S., 2003. Applying genetic algorithms and neural networks to the container loading problem. Journal of Information and Optimization Sciences 24, 423-443.

Yeung, L.H.W., Tang, W.K.S., 2005. A hybrid genetic approach for container loading in logistics industry. IEEE Transactions on Industrial Electronics 52, 617-627.

Zhang, D., Peng, Y., Leung, S.C.H., 2012. A heuristic block-loading algorithm based on multi-layer search for the container loading problem. Computers & Operations Research 39, 2267-2276.

Zhu, W., Huang, W., Lim, A., 2012a. A prototype column generation strategy for the multiple container loading problem. European Journal of Operational Research 223, 27-39.

Zhu, W., Lim, A., 2012. A new iterative-doubling Greedy-Lookahead algorithm for the single container loading problem. European Journal of Operational Research 222, 408-417.

Zhu, W., Oon, W.-C., Lim, A., Weng, Y., 2012b. The six elements to block-building approaches for the single container loading problem. Applied Intelligence 37, 431-445.