An overview on container loading problems.

Pedro Henrique Centenaro [a]

[a] Department of Control, Automation and Computation Engineering, Federal University of Santa Catarina, Blumenau, Santa Catarina, Brazil, pedro.centenaro@grad.ufsc.br

Abstract. The logistic process of loading cargo into containers encompasses all industries, being a crucial step to improve the delivery of goods. In order to obtain competitive advantages, companies seek to maximize the total volume or profit of the loaded items, while also minimizing the number of containers needed to deliver products. Moreover, the loading process may be complicated by constraints such as weight limits and box orientations. Though it is mathematically possible to model container loading problems (CLPs), solving them to optimality is practically impossible, due to their combinatorial nature. Thus, most of the research on this topic has focused on heuristics that produce good solutions in an acceptable amount of time. In light of these facts, this overview investigates the types of CLPs that have been proposed and considered by researchers, and the constraints that have been tackled the most. We also investigate a few heuristics in greater detail, based on their importance and ability to illustrate the different techniques that can be used to solve CLPs. We observe that most researchers have focused on solving CLPs with a wide variety of boxes and few container types. Furthermore, the average number of constraints considered in papers has changed little in three decades. This indicates the existence of many gaps in the literature. In conclusion, we hypothesize that the almost stagnant exploration of CLPs may be explained by the authors' focus on improving the efficiency of container loading based on previous works that are already biased towards certain problem configurations. We conclude that society will benefit from efforts to close research gaps and make CLP software more accessible to laypersons.

Keywords. Container loading problems, heuristics, mathematical optimization, decision support systems, literature overview.

1. Introduction

Container loading problems (CLPs) are combinatorial optimization problems in which the goal is to orthogonally load boxes into one or more containers, subject to a set of contextual constraints, e.g. weight or volume limits, compatibility and stackability of boxes. Typically, when dealing with a single container, the objective is to minimize the unused volume. For multiple containers (MCLP), the goal is usually to minimize the number of used containers [1]. Since they are generalizations of the knapsack problem (KP), (M)CLPs are NP-hard, which means polynomial-time algorithms for solving them are unknown and unlikely to be found [2]. Regardless, the speed with which computers can solve such problems to near-optimality means the study of optimization techniques is a necessity in nowaday's competitive landscape.

This overview on (M)CLPs is structured as follows: Section 2 provides more details on the methods and goals used to investigate the subject. Section 3 explains the main concepts used to classify (M)CLPs, as well as the types of constraints that such problems may include that have so far been identified in the literature. The section closes with a brief view into the state and trends of (M)CLP research, and what gaps exist in the field. Section 4 gives an overview on exact methods for solving (M)CLPs and identifies the niches in which they are useful. Section 5 presents a number of heuristics in accordance with their categorization in the literature. After mentioning heuristics of higher complexity, the section comes to a close by discussing how heuristics can be compared. Section 6 introduces decision support systems, programs that have the potential to approximate researchers and the needs of the industry. Finally, we present our concluding remarks in section 7, based on the development of the previous sections.

2. Research methods

This overview is based on two fronts: macro and micro. The macro is the categorization of (M)CLPs, particularly that of Bortfeldt et al. [3] and Fanslau et al. [4]. The micro is part of a larger research project that requires the reading of specific papers detailing the techniques proposed by authors over the years. We used the macro to contextualize (M)CLPs, the trends and state of research, and the micro to expand upon topics where examples are deemed important. Special attention was dedicated to heuristics, which have been the main focus of researchers' attention for decades.

The goals of this research are: (i) to characterize (M)CLPs in a way that can be understood by laypersons; (ii) to identify the terminology used by researchers to differentiate problem types and constraints; (iii) to describe a few of the predominant methods proposed by researchers to solve (M)CLPs; (iv) to analyze trends and identify gaps in the literature. Additionally, we hope that the conclusions drawn from the development of this overview will be useful for the furtherance of our, and the readers', research projects.

3. An overview on problems

There are many types of (M)CLPs and constraints to consider. This section's overview is based on the 2012 literature review by Bortfeldt et al. [3], who analyzed 163 papers on (M)CLPs, made available between 1980 and 2011.

3.1 Problem types

The generic description of CLPs allows for many different interpretations of the problem. We first detail heterogeneity. We say that the set of items considered in a problem is *weakly heterogeneous* (WH) if the set of items is large and it is possible to split them into a small number of categories. Otherwise, the items are *strongly heterogeneous* (SH). The same categorization is valid for containers in MCLPs. When the number of containers available is enough to load every item in the problem, we have an *input value minimization problem*; that is, the goal is to minimize the number of containers used. Elsewise, the objective is to maximize the value or volume of the loaded cargo, and we have an *input value maximization problem*.

3.2 Constraint types

Bortfeldt et al. [3] consider 10 types of constraints CLP heuristics may implement. *Weight limits* impose a threshold on the total weight of the items within the container, while *weight distribution constraints* attempt to homogenize the sum of weights over areas of the container floor. *Loading priorities* arise when a subset of items must be loaded for a solution to be valid, while *orientation constraints* limit the number of rotations that may be applied to the boxes. *Stacking/load-bearing constraints* require that all boxes be able to handle the cumulative weight of the boxes stacked on top, which is further complicated by the fact that different box orientations can endure different weights. In *complete-shipment* heuristics, if a certain item is loaded into the container, then all items in the same category must also be loaded. *Allocation constraints* are imposed in MCLPs when items in the same category should be put in the same container, or when incompatible item types (e.g. fragile and heavy objects) should go in different containers. *Positioning constraints* require that certain items be placed in specific spaces within the container. For instance, in some cases it might be desirable to keep heavy objects closer to the container door, to facilitate the unloading process. *Stability constraints* are explicitly imposed to guarantee that items aren't damaged during transportation. An item is vertically stable if it has no risk of falling to the ground or on top of other boxes. Usually constraints demand that boxes be partially or completely covered underneath by the top of other boxes, or that at least their center of gravity is covered. On the other hand, an item is horizontally stable if its horizontal shifts are insignificant as the container moves. Heuristics often attempt to guarantee this stability by loading boxes adjacent to each other, or to a container wall. Finally, *complexity constraints* aim to make packings simple enough that humans can understand and arrange them quickly.

3.3 State of research

Of the papers reviewed by Bortfeldt et al. [3], 51.5% consider minimization problems. Notably, most efforts have gone towards solving either MCLPs with identical containers or the CLP with SH items. Meanwhile, the CLP with WH items and the MCLP with WH items and containers have received very little attention from researchers. On the other hand, 58.9% of papers consider maximization problems, with significant attention being given to CLPs with a large container. However, almost no research has been done on MCLPs. Furthermore, with regards to item types, 93.9% of papers deal exclusively with cuboids.

On the topic of constraints, figure 1 compares the growth in the number of papers published and the average number of constraints considered per paper in every quinquennium (and the 2010 to 2011 period).

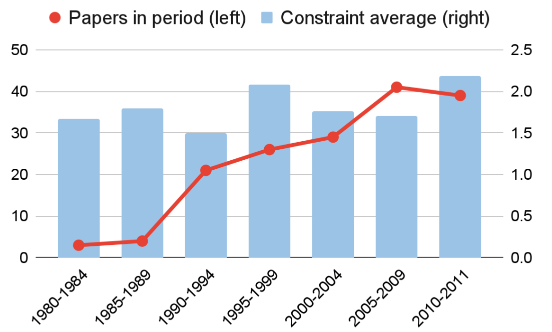

Fig. 1 - Constraint averages and papers published on (M)CLPs per period.

The notable increase in the number of papers is accompanied by an inexpressive variation of the constraint average over the years. Figure 2 presents the findings by Bortfeldt et al. [3] on the number of constraints considered by all the papers analyzed.

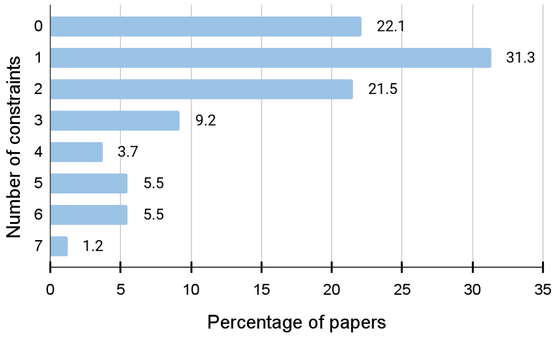

Fig. 2 - Percentage of papers implementing each number of constraints from 1980 to 2011.

This chart gives us the insight that more than half the papers presented in 31 years consider at most one constraint type. According to Bortfeldt et al. [3], the majority (70.6%) of papers take orientation constraints into account, followed by stability constraints (37.4%) and no constraints at all (22.1%). This data, in conjunction with the data on problem types, seems to indicate that most problems tackle specific iterations of the (M)CLP. It is worth noting that, in many cases, orientation constraints are very easy to implement – a few boolean variables describing which rotations are allowed for a certain item should be enough for most heuristics. Thus, it makes sense for this type of constraint to be implemented so often.

4. Exact approaches

Given a mathematical model that adequately assesses all the constraints needed to represent an optimization problem, it is possible to arrive at an optimal solution – hence the name "exact approach". Egeblad et al. [5] presented a relatively simple CLP model that maximizes the profit of the selected boxes. The coordinates and dimensions of box i are represented by tuples (x_i, y_i, z_i) and (w_i, h_i, d_i), and the container has width W, height H and depth D. The objective (expression (1)) is to maximize the profit of a single container, with p_i denoting the profit of box i and s_i being a binary variable that defines whether box i has been selected (1) or not (0).

$$\max \sum_{i=1}^{n} p_i s_i \qquad (1)$$

This model constrains the boxes' coordinates to the container's space. For example, for box i, constraints (2) through (4) must be respected.

$$0 \leq x_i \leq W - w_i \qquad (2)$$
$$0 \leq y_i \leq H - h_i \qquad (3)$$
$$0 \leq z_i \leq D - d_i \qquad (4)$$

To ensure that selected boxes do not overlap, binary variables are introduced to dictate whether one box is to the left (ℓ_{ij}), right (r_{ij}), over (o_{ij}), under (u_{ij}), behind (b_{ij}) or in front (f_{ij}) of the others. For example, if box i must be to the left of j, then ℓ_{ij} is 1, and constraint (5) is valid.

$$x_i - x_j + W\ell_{ij} \leq W - w_i \qquad (5)$$

An earlier model by Chen et al. [6] includes binary variables representing item rotations and considers multiple containers. The objective becomes the minimization of wasted space across the selected containers.

Though applicable to problems with up to a few dozen boxes, exact models are not viable for solving large-scale CLP problems. Because they need binary variables to work, these models are solved through branch-and-bound methods. Such methods nullify all integrality (binary) constraints, and instead require that the affected variables be non-negative real numbers, a process known as relaxation. The problem is then divided into multiple subproblems that are solved until a purely integer solution is found and determined to be optimal [7]. Although contemporary methods like branch-and-cut improve this process, they are too slow, due to the sheer number of subproblems generated. Egeblad et al. [5] remark that their model's binary variables further complicate this issue, since they are tied to big M constraints which, when relaxed, may increase the number of subproblems to solve. The same is true for the model by Chen et al [6]. Despite the shortcomings of exact approaches to CLPs, they can aid heuristics by solving simplified packing problems that provide useful upper and lower bounds to solutions [5,8]. For instance, a CLP with n items may be turned into a KP as indicated in expressions (6) through (8).

$$\max \sum_{i=1}^{n} w_i h_i d_i s_i \qquad (6)$$
$$\text{s.t.} \sum_{i=1}^{n} w_i h_i d_i s_i \leq WHD, \qquad (7)$$
$$s_i \in \{0, 1\}. \qquad (8)$$

The objective (6) is to maximize the total volume of selected items, constrained by the fact that this volume must not exceed the container's (7). Since volume is treated as a one-dimensional property by this model, it stands to follow that an optimal solution to this KP is an upper bound to the three-dimensional model by Egeblad et al [5].

5. Heuristics

Heuristics are computational methods that attempt to produce solutions to problems in an adequate amount of time, which often means a suboptimal result is achieved. Humans frequently rely on heuristics for problem-solving [9], and natural processes are often taken as inspiration to compute problems of high complexity [10]. Fanslau et al. [4]

divide CLP heuristics in the literature into three categories, which we explore in the following subsections.

5.1 Conventional heuristics

Heuristics classified as conventional are specifically designed to deal with the CLP. Constructive heuristics build a solution from nothing, while improvement heuristics start with a solution and modify it in the hopes of finding a better configuration. There are "hybrid" heuristics that implement both of these procedures. According to Bortfeldt et al. [3], the wall-building method (WB) proposed by George et al. [11] is the earliest constructive heuristic proposed for the CLP. In the WB, a single container is repeatedly sliced into spaces that share its width and height, but with reduced depth. Layers are then filled with boxes, following a series of procedures that attempt to minimize wasted space. The layer-building method (LB) by Bischoff et al. [12] attempts to fill as much base area as possible with pairs of item types, a procedure that is applied from the ground up. The block-building method by Eley [13] sorts boxes from most to least voluminous and then places them in the container, attempting to minimize unfillable volumes. This results in the formation of blocks of items of the same type, which is helpful for packing. Figure 3 compares the spaces to fill in each iteration of WB, LB, and BB. While WB is oriented by depth, LB is oriented by height. BB considers filling all the available spaces adjacent to the current load configuration.

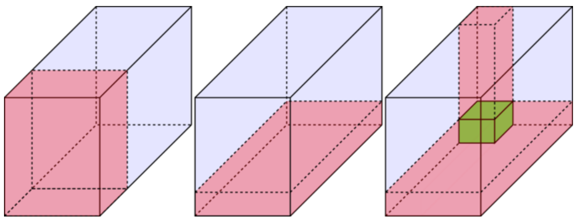

Fig. 3 - Comparison between WB, LB and BB spaces (in red), respectively.

5.2 Metaheuristics

Metaheuristics are more abstract heuristics that can be applied to a wide variety of problems – that is, they are not specialized. These methods are frequently based on natural processes, like simulated annealing (SA), used by Egeblad et al. [5] to solve CLPs. Firstly, boxes are enumerated from 1 to n, and three sequences of boxes (A, B and C) are generated. The position of a box in one such sequence reveals details about its placement relative to the others. For example, if box i comes before j in sequence A, that means i is to the left, above or in front of j. By checking the boxes' positions in the other sequences, an exact conclusion about their relative placements is reached. In order to improve the initial solution, the SA randomly shuffles the positions of items in each sequence and evaluates the results. If the shuffling led to a better solution, then it is accepted as the current solution. Otherwise, a worse solution may still be accepted, as a measure to avoid local optima. This procedure is probabilistic, with the odds of acceptance decreasing as the number of accepted solutions increases.

Genetic algorithms (GA) are another popular type of metaheuristic used to solve CLPs. For instance, Gonçalves et al. [14] propose a method that encodes packing sequences and orientations in chromosomes. In a problem with M boxes, each chromosome has $2M$ genes, with the first M genes being used to describe the order in which to pack boxes, and the last M genes defining their orientation. To convert the genes into packings, a procedure similar to the BB is used. In every iteration of the method, the best-evaluated solutions are randomly combined with other solutions in the population by swapping genes from their chromosomes. The resulting variety leads to convergence in the long run. To avoid local optima, a number of solutions suffer mutations from one population to the next. Furthermore, multiple populations are managed every iteration, with the two best solutions across all populations being inserted in all populations.

5.3 Tree search methods

Tree search methods perform incomplete or graph searches for solutions to the CLP. Eley [13] proposed one such procedure to improve solutions obtained by their BB method. In this procedure, different packing orders and orientations are considered per tree node. Pisinger [15] proposed a heuristic that fills containers layer by layer, solving KPs to determine box positions. A tree search algorithm is then employed to check if variations of the layers' depths and the box columns' widths result in better solutions.

5.4 Highly-constrained heuristics

The approaches presented so far are relatively simple. As such, they often serve as building blocks for complex heuristics, like the one proposed by Egeblad et al. [16] for loading furniture of different sizes into a single container, which implements 7 of the 10 constraints identified by Bortfeldt et al [3]. The heuristic begins by building placement templates for larger items in a way that attempts to ensure their stability. Then, it builds "quad-walls" with 4 templates determined by tree search. Afterwards, a greedy heuristic is used to place the most voluminous medium-sized items in the container. Finally, the placements of small items are determined by a slightly modified version of the tree search WB heuristic by Pisinger [15]. Another example of a highly-constrained heuristic is the one by Gendreau et al. [17], which employs metaheuristics to solve problems combining MCLPs with capacitated vehicle routing problems.

5.5 Comparisons between heuristics

As the great number of (M)CLPs implies, the performance of heuristics depends largely on the problem they are applied to. The WB alone proves this point: George et al. [11] proposed two procedures for selecting which boxes to fill spaces with. Later, Bischoff et al. [18] compared the procedures and realized that one consistently outperformed the other for their test set. Furthermore, they showed that the better WB procedure was greater for solving problems with SH sets of items, while by combining the WB with another heuristic [12], the method performed better for WH sets. Test sets are also used to compare different heuristics, with the most popular being listed by Bortfeldt et al [3].

6. Decision support systems

Decision support systems (DSSs) are tools that provide an interface for the analysis and visualization of solutions. While commercial DSSs for (M)CLPs exist aplenty, DSSs are not the main focus of academic research. In this section, we describe a few of the DSSs that have been proposed in the literature. A 2004 paper by Chien et al. [19] attempts to improve a company's loading process with a graphical user interface (GUI). Users are allowed to select between two container types frequently used by the company. Then, they need only choose the file containing the problem's data for the program to compute a solution using a WB approach. A three-dimensional view of the container with the boxes is produced, and the loading process can be visualized step by step. A similar DSS was proposed in 2010 by Dereli et al. [20]. However, in order to build walls of items, this DSS uses a bee colony metaheuristic with five parameters that are up to the user to determine. Thus, this solution might not be appropriate for laypersons. In 2022, Pachón et al. [21] introduced a DSS that uses a GRASP metaheuristic [22] to load cargo compliant with orientation, stacking, weight limits, and stability constraints. This implementation also handles multi-drop, a special case of positioning constraint that groups together items to be delivered to the same client. Furthermore, cuboids and irregular shapes, including cylinders, can be loaded together.

7. Conclusions

Despite their seemingly simple nature, (M)CLPs are full of intricacies that, from a purely abstract perspective, can be ignored. Perhaps this serves as a hypothesis for the overarching theme of deficits in the literature identified by Bortfeldt et al [3]. While initially authors focused on developing intuitive methods for loading cargo (section 5.1), in recent decades the interest has shifted to robust methods that improve upon groundwork heuristics or introduce stochastic elements to optimize solutions as much as possible (sections 5.2 and 5.3). Hence, most of the interest lies in improving cargo-loading methods that disregard most practical constraints, which could explain why so few constraints are considered to this day. This could also elucidate why most papers consider SH sets of items with either a single container or a WH set of containers – because there is a larger pool of heuristics and popular test sets for comparison.

It is interesting to note that conventional heuristics, though limited, serve as building blocks for most (M)CLP methods. For instance, the WB by George et al. [11] is so simple that adapting it to support multiple containers and orientation constraints is trivial. Furthermore, even though methods like the ones proposed by Pisinger [15] and Egeblad et al. [16] completely redefine the WB process, they still draw inspiration from its layer-building approach. Therefore, it seems reasonable to infer that the more robust a method becomes, either in its search for solution improvements or the number of constraints it can handle, the greater its integration of previous concepts and procedures.

Finally, it is worth noting that commercial DSSs are built to accommodate as many constraints as possible while staying relatively easy to use. Academia seems to have much to learn from these tools: The DSS by Pachón et al. [21] is an example of software partially based on the industry's standards, with a simple interface and the implementation of many more constraints than the average found in the literature.

In conclusion, society has the most to gain from the following concerted efforts by researchers: to explore problem types that have mostly been ignored in the literature; to implement robust methods capable of solving problems with more constraints than the average in the last decades; and to seek ways to integrate these advancements in DSSs that the layperson can use.

8. References

[1] Lim A, Zhang X. The container loading problem. *Proceedings of the 2005 ACM symposium on Applied computing*. 2005; 913-917.

[2] Erickson J. *Algorithms*. 2019. 472.

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

[4] Fanslau T, Bortfeldt A. A tree search algorithm for solving the container loading problem. *INFORMS Journal on Computing*. 2010; 22(2):222-35.

[5] Egeblad J, Pisinger D. Heuristic approaches for the two- and three-dimensional knapsack packing problem. *Computers & Operations Research*. 2009; 36(4):1026-1049.

[6] Chen CS, Lee SM, Shen QS. An analytical model for the container loading problem. *European Journal of Operational Research*. 1995; 80(1):68-76.

[7] Chen DS, Batson RG, Dang Y. *Applied integer*

programming: modeling and solution. John Wiley & Sons, Hoboken; 2010. 490.

[8] Santos LF, Iwayama RS, Cavalcanti LB, Turi LM, Morais FE, Mormilho G, Cunha CB. A variable neighborhood search algorithm for the bin packing problem with compatible categories. *Expert Systems with Applications*. 2019; 124:209-225.

[9] Muraski C, Bossaerts P. How Humans Solve Complex Problems: The Case of the Knapsack Problem. *Scientific Reports*. 2016; 6(34851):1-10.

[10] Yu B, Yang ZZ, Yao B. An improved ant colony optimization for vehicle routing problem. *European Journal of Operational Research*. 2009; 196(1):171-6.

[11] George JA, Robinson DF. A heuristic for packing boxes into a container. *Computers & Operations Research*. 1980; 7(3):147-56.

[12] Bischoff EE, Janetz F, Ratcliff MS. Loading pallets with non-identical items. *European Journal of Operational Research*. 1995; 84(3):681-92.

[13] Eley M. Solving container loading problems by block arrangement. *European Journal of Operational Research*. 2002; 141(2):393-409.

[14] Gonçalves JF, Resende MG. A parallel multi-population biased random-key genetic algorithm for a container loading problem. *Computers & Operations Research*. 2012; 39(2):179-90.

[15] Pisinger D. Heuristics for the container loading problem. *European Journal of Operational Research*. 2002; 141(2):382-92.

[16] Egeblad J, Garavelli C, Lisi S, Pisinger D. Heuristics for container loading of furniture. *European Journal of Operational Research*. 2010 Feb 1;200(3):881-92.

[17] Gendreau M, Iori M, Laporte G, Martello S. A tabu search algorithm for a routing and container loading problem. *Transportation Science*. 2006; 40(3):342-50.

[18] Bischoff EE, Marriott MD. A comparative evaluation of heuristics for container loading. *European Journal of Operational Research*. 1990; 44(2):267-76.

[19] Chien CF, Deng JF. A container packing support system for determining and visualizing container packing patterns. *Decision Support Systems*. 2004; 37(1):23-34.

[20] Dereli T, Daş GS. Development of a decision support system for solving container loading problems. *Transport*. 2010; 25(2):138-47.

[21] Pachón JC, Martínez-Franco J, Álvarez-Martínez D. SIC: An intelligent packing system with industry-grade features. *SoftwareX*. 2022; 20:101241.

[22] Martínez DA, Alvarez-Valdes R, Parreño F. A grasp algorithm for the container loading problem with multi-drop constraints. *Pesquisa Operacional*. 2015; 35:1-24.