Pallet building and truck loading strategies for an inter-depot transportation problem

M.T. Alonso[1], R. Alvarez-Valdes[2], F. Parreño[1], and J.M. Tamarit[2]

[1] Department of Math, University of Castilla-La Mancha, Albacete (Spain)
mariateresa.alonso@uclm.es; francisco.parreno@uclm.es
[2] Department of Statistics and Operational research, University of Valencia, Valencia (Spain), ramon.alvarez@uv.es; jose.tamarit@uv.es

Abstract. The mission of logistics is basically to get the right goods or services to the right place, at the right time, and in the desired conditions, while making the greatest contribution to the company [1].
Everyday a distribution company has to decide how to put goods onto pallets according to the customers' orders and then how to efficiently distribute these pallets among the number of trucks so as to reduce the trucks needed to supply the customers. The first problem is known as the *pallet loading problem*, whereas the second problem is the *container loading problem*. But, we solve the problem in one phase, building and placing pallets at the same time. For each position in the truck a pallet is built, tailored for that position according to the constraints. At each iteration, a randomized constructive procedure builds a solution for loading all the products.

1 Introduction

The problem begins when a customer has to receive a set of different products, $j \in \{1, \dots J\}$, to meet daily orders. The days are represented by $d \in \{1 \dots D\}$, and each order line is composed of a product type j and the amount of it for that day a_{jd}. The product is defined by its dimensions (x_j, y_j, z_j) in millimetres, its weight w_j in kilograms, and its possible rotations (rx_j, ry_j, rz_j), with $rx_j = 1$ if the dimension x_j of the product can be upward.

All the products have a pre-defined placement in layers. A layer is an arrangement of products of the same type in rows and columns, defined by its dimensions (l_j, w_j, h_j) in millimeters, its weight q_j in kilograms, and the number of units in the layer lu_j. The quantity of layers per day is $n_{jd} = a_{jd}/lu_j$, whereas the total quantity of layers is $n_j = \sum_{d \in D} a_{jd}/lu_j$. The number of layers of the same product that can be stacked is ls_j and the stackability identification is $ids_j \in \{1, \dots, 4\}$, which represents the density of the product, denser products having an identification of 1 and less dense ones receiving a 4. One layer can be placed on top of another if the layer below has a stackability identification smaller than or equal to that of the layer above.

The layers are stacked to build pallets. Here a pallet provides a loading space above its base, made of wood, metal, or plastic. Each pallet has a base

whose dimensions are (l^p, w^p, h^p) millimetres, a weight, q^p, and it can be rotated. Typically the space in horizontal directions is limited by the horizontal size of the pallet, though some overhang in both directions can be allowed. In the vertical direction the height of the loaded pallet is limited to half the height of the truck, in order to place two pallets, one on top of the other.

The layer composition of each type of product has been previously decided, without holes in the surfaces, so we have to stack layers to build pallets. Three different types of pallets can be built:

- *Stock pallet*, where all the layers are of the same product. In this case the number of layers composing the pallet is fixed, due to stackability reasons, and therefore the pallet height is also fixed.
- *Case pallet*, which combines layers of different products.

The company wants to make as many stock pallets as possible. Only when not all the layers of a product fit onto stocks pallets are the remaining layers used to form case pallets.

Apart from the pallet composition, other constraints should be considered in this phase, pallet building:

- **Orientation constraint**: layers can be rotated, and all placements must be parallel to the sides of the pallet.
- **Support constraint**: A layer must be supported at a certain percentage, 75% to be precise, by the pallet surface, or the layer on which it rests.
- **Priority constraint**: A pallet can only contain items of orders of the same due date.
- **Stackability constraint**: The pallet height is limited to half the truck height. Therefore, at most two pallets can be placed in each stack. However, not any two pallets can be placed in the same stack, as the selection of the two pallets depends on certain properties such as their weight, with lighter pallets being placed on top of heavier pallets. Each pallet belongs to a stackability group, based on the items loaded on it. Denser pallets have a lower identification and less dense ones a higher identification, so that one pallet can be placed on the top of another if its identification is greater than or equal to the identification of the pallet below.

1.1 Truck loading

Once the pallets are built, they have to be placed onto trucks, and a given set of trucks $k \in \{1 \dots K\}$ is available. We deal with an homogeneous fleet, all of the same type, two axles trucks. For each type of truck k, we know its dimensions (L, W, H), the weight it can bear on the front axle, Q_1, and on the rear axle, Q_2, and the maximum weight Q. The distances of the axles from the front part of the truck are δ_1 and δ_2. For one type of truck to another can vary any parameter a little bit, as lenght, width, weight and so on. Moreover, there are restrictions on the maximum weights on the axles. The truck has two axles that carry the

load, with the front axle at position δ_1 and the rear axle at position δ_2, where δ_1 and δ_2 are measured from the front of the loading space of the truck.

For security reasons the load has to be well spread on the truck and must be distributed between the two axles, not exceeding the maximum weight they can bear. If Cog_x is the x-coordinate of the center of gravity of the load inside the truck and g the total weight of the load, the weights supported by the front axle (g_1) and the rear axle (g_2) are determined by equations (1) and (2).

$$g_1 = \frac{g(Cog_x - \delta_1)}{\delta_2 - \delta_1} \tag{1}$$

$$g_2 = \frac{g(\delta_2 - Cog_x)}{\delta_2 - \delta_1} \tag{2}$$

In this phase of truck loading, some other constraints related to how to place the pallets should be considered:

- **Priority constraint**: The due dates of the orders must be respected, meaning that if truck x contains orders for day d, then trucks $x+1$, $x+2$, etc. cannot contain orders of days before d. If the items of day d do not completely fill a truck, the items of the day $d+1$ can be used to fill up the truck. If with these items the truck is not still completely filled, then items of day $d+2$ can be used, and so on, until the truck is filled or there are no items left.
- **Weight constraint**: The weight capacity Q of the truck cannot be exceeded.
- **Axle weight constraint**: The axles of the truck have weight limits Q_1 and Q_2; the weights g_1 and g_2 in equations (1, 2) should not exceed Q_1 and Q_2, respectively. Usually, $Q_1 + Q_2 \geq Q$.
- **Stability constraint**: For stability reasons, the pallets have to be placed touching at least one side of the truck.

The problem corresponds to the inter-depot transportation of a distribution company, where the main objective is to minimize the total number of trucks needed for all the orders and, in case of solutions with the same number of trucks, one is better than the other if the last truck is less filled.

We tackle two problems: putting products onto pallets and pallets into trucks, but both have the same characteristics. According to the typology proposed by Wäscher et al. [2], they can be classified as three-dimensional problems. The kind of assignment is input minimization, a set of small items is to be assigned to a set of large objects whose quantity is large enough to accommodate all the small items. The assortment of the small items is weakly heterogeneous, the small items can be grouped in relatively few classes with respect to the total number of items. The assortment of the large objects is only one large object with fixed dimensions, pallet and truck, and the shape of the small items is regular. Both problems can be classified as a Single Stock Size Cutting Stock problem (SSSCSP).

2 Literature review

The problem that we study represents an important category of loading problems, namely the problem of loading products on pallets and placing pallets into trucks. In particular, the loading of layers on pallets and pallets into trucks, because the layer composition is given.

The constraints that are relevant in this problem are largely the ones mentioned previously: weight and weight distribution, stackability, and allocation of items. In the container loading problem these constraints have been considered in many previous studies. According to the survey by Bortfeldt and Wäscher [3], at least 13.9% of the container loading literature deals with weight limit, while weight distribution is considered by 12.1% of the papers, keeping the centre of gravity in the geometrical mid-point of the container or not exceeding a certain distance. The stackability constraint is dealt with by 15.2% of the papers.

Here, we mention some studies directly related to the problem being tackled in this problem:

- **Priority constraint**: The due dates of orders must be respected. Bischoff and Ratcliff [4] and Ren et al. [5] work with product priorities, trying to accommodate first the more desirable products.
- **Stackability constraint**: Preventing a product type from being placed on top of another type has been considered by Scheithauer and Terno [6] and constraints such as "Stack no more than X height" are encountered in Bischoff and Ratcliff [4] and in Lin et al. [14].
- **Support constraint**: Many studies([13], [15], [16], [17], [18]) work with full support, and some others ([19],[20]) with partial support.
- **Weight constraint**: Gehring and Bortfeldt [12], Bortfeldt et al. [21], Terno et al. [22] and Egeblad et al. [23] include among their constraints a maximum weight that cannot be exceeded.
- **Axle weight constraint**: Bischoff and Marriott [25] keep the centre of gravity in the geometrical mid-point of the container. A certain flexibility, not exceeding a certain distance from the geometrical mid-point, is allowed in Bortfeldt and Gehring [13] and Gehring and Bortfeldt [12].

However, there are not many papers that address the issues of pallet and truck loading together. The next references place products on pallets and pallets into trucks.

Morabito et al. [1] deal with the same problem but in 2D because the products cannot be stacked. The set of products have dimensions (l, w, h), where h represents the maximum height for loading. The problem consists in how to load the maximum number of products on a pallet of dimensions (L, W). The problem is solved by using the approach proposed by Morabito and Morales [11]. The approach consists in a heuristic algorithm that divides the rectangle into 5 blocks and in each block the products are placed in a fixed orientation. Finally, when the pallets are built, they use the same approach to load the pallets into the trucks.

Haessler and Talbot [10] describe a heuristic for loading customers' orders, developing load patterns for trucks and rail shipments. The products have low density and for that reason the approach is based on loading by volume rather than by weight. To deal with axle weight constraints, stacks are sequenced by alternating the heaviest and lightest stacks.

Takahara [9] deals with the problem of loading a set of items P, in a set of multiple bins, containers and pallets, S. A loading sequence for the items is chosen, for instance $\sigma = p_2, p_3, p_1, p_5, ..., p_n$, and this determines the order in which the items are inserted into the bins. The sequence is selected by a metaheuristic method based on a neighbourhood search. A selector determines the sequence of the bins, for instance $\mu = s_2, s_1, s_3, ..., s_m$. When a bin is selected the first item is loaded in the bin, placing it into the first space in which it fits. If the item does not fit into the bin, the next bin is selected. There is an observer that determines when to exchange the sequence of the items with a neighbour sequence and when the choice of the bin is changed from following the sequence to being randomly chosen, depending on the quality of the solutions.

Lim et al. [8] deal with the single container problem with axle weight constraints. They propose an integrated heuristic solution approach that combines a GRASP wall-building algorithm with linear integer programming models. They first apply a customized wall-building heuristic based on the GRASP by Moura and Oliveira [7], including special considerations for box weight and density. Then they use an integrated approach to handle the weight requirements. If the container load limit is exceeded, they unload the necessary number of boxes by iteratively solving an integer programming model to meet the requirement. If the axle weight limit is exceeded, they take two steps iteratively until the limit is satisfied: the first step consists in interchanging positions of the walls and the second step consists in solving an integer programming model to unload boxes and apply the first step one more time to improve the container balance as well as to force a feasible weight distribution.

3 The problem approach

We can consider the problem as two subproblems. The first subproblem would be building the pallets, where the input will be the list of orders and the output will be a pallet list with all the products to send. The second problem would be how to load the pallets onto the trucks, where the input will be the list of pallets provided by the first problem and the output will be the number of trucks that are necessary to load the whole list of pallets from the first problem.

The disadvantage of this approach is that the two problems are independent and one does not take into account the features and constraints of the other. In particular, in the first problem the pallets are built without any information about the trucks. Therefore, a good solution for the pallet loading phase is not necessarily a good starting point for the truck loading phase.

Our proposal is to keep a constant flow of information between the two problems, taking into account the constraints and the features of both. In the truck

loading phase, we gather information about the features of the pallet, whereas in the building pallet phase, we collect the information about the placement of the pallet in the truck. With both pieces of information, we build the best pallet for a specific position in the truck using the remaining products.

To solve this problem, we propose a multi-start algorithm with a constructive phase in which a solution is built by adding a pallet at each step according to the information collected from the truck. In the next sections, the algorithm is described in detail.

3.1 Constructive phase

The constructive algorithm builds a solution by means of an iterative process in two steps, the first one consists in finding a space in the truck to place the pallet and the second one in building the pallet and placing it onto the truck.

Step 1. Selecting a place The first step is to select a position in the truck to place the next pallet. If there are no oversize layers forming oversize pallets, each pallet would have a predefined position in the truck because all the pallets would have the same dimensions. But if there are some oversize pallets in which some layers are larger than the base, the position of the pallet cannot be predefined, because it depends on the pallet dimensions. That is the reason why we have to select a place to put the pallet at each step.

Other constraints that have to be taken into account when selecting the place are the center of gravity and the axle weight, because the position of the pallet affects the weight that each axle will bear, and can move the center of gravity away from the geometric center of the truck.

Since the weight has to be balanced between the two axles, we have decided to divide the truck into two parts, front and rear, starting from the center, as in Figure 1, in which white corresponds to front and blue to back. By placing a pallet on a different side each time, we can control the balance of the weight and keep the center of gravity approximately in the center of the truck.

Fig. 1. Truck overview, divided into two parts

The center of gravity and the weight each axle will bear are calculated with Equations (1) and (2). If the weight that axle 1 can bear $(Q_1 - g_1)$ is greater than that of axle 2 $(Q_2 - g_2)$, the next space is selected at the front; otherwise, it is chosen at the back.

Starting at the center of the truck and going in the chosen direction, towards the front or back, we look for a space big enough to place a pallet. We search the floor of the truck first and, if there is no space left, we look at the top of the pallets already loaded. When the space is found its features are extracted: its dimensions (S_x, S_y, S_z), the maximum weight that can be put into this position W_s, and the stackability identification of the lower pallet, ids_s, if the space is above another pallet.

With these features of the space, we go on to the next step.

Step 2. Building a pallet for this place In this step, a pallet is built that is tailored to the space selected, taking into account its characteristics. When building a pallet, we always use the predefined layers.

The layers are ordered by day, first days first, and by density, the denser ones first. We take the first layer of the list and check whether there are enough layers of this product to build a complete pallet, in other words a stock pallet. If this is the case, we build a stock pallet. Otherwise, these layers are added to the pallet and we continue searching for layers that cannot form a stock pallet by themselves and add them to the pallet, building a case pallet with layers of different products.

One layer is selected if it satisfies several conditions. First, it fits into the selected space. Second, the weight of the pallet, including this layer, does not exceed the maximum allowed weight for this space. Third, the height of the pallet does not exceed half the height of the truck.

The pallet built in that way is defined by its dimension, (l^p, w^p, h^p), the number of layers nl^p, the weight q^p, and the stackability identification, which will be the minimum identification of all the layers making up the pallet.

The pallet is placed into the selected space, touching the nearest wall of the truck, in such a way that the potential gap is left in the middle of the truck, where it can be used by pallets on the other side and also by pallets on top, always satisfying a 75% area of support.

When the pallet is inserted into the solution, a new cycle begins, calculating the new center of gravity, the load bearing of the axles, the direction to be chosen, and updating the list of layers.

When the first level on the floor of the truck is completed, all the pallets are pushed to the front of the truck, if the weight axle restriction allow it. The way in which we place the pallets, starting from the center, may leave two gaps, one at the front and another at the back of the truck. By pushing the pallets to the front of the truck we join the gaps at the end, with the possibility of placing another pallet in that gap. This process is performed before starting the placement at the second level.

The constructive phase continues until all the layers have been placed or until no more pallets can be placed on the truck because one of the constraints, limiting the volume, the weight, or the number of pallets, has been reached. If there are still pallets to be sent, a new truck is opened and the process continues.

4 Randomized constructive algorithm

In order to improve the solution, our proposal is a randomized constructive algorithm using the constructive procedure described in the previous section and adding a random selection mechanism. In the constructive phase the solution is built step by step by adding one element to a partial solution, but in order to select the element to add, a greedy randomized strategy is used to provide diversity to the process.

The randomization procedure for selecting the layer to be packed is based on the strategy introduced by Resende and Werneck [24], in which a restricted candidate list, RCL, is built by randomly selecting a fraction γ ($0 \leq \gamma \leq 1$) of the elements to be placed. Then, the first element of RCL in the ordered list is selected.

It is difficult to determine the value of γ that gives the best average results. The reactive strategy, proposed by Prais and Ribeiro (2000), is to let the algorithm find the best value of γ from among a small set of allowed values. The parameter γ is initially taken at random from a set of discrete values $\mathcal{D} = \{\gamma_1, \gamma_2, ..., \gamma_m\}$, but after a certain number of iterations, the relative quality of the solutions obtained with each value of γ is taken into account and the probability of values consistently producing better solutions increases. In this way, the randomization procedure reacts to the results obtained in the iterative process, tuning the parameter to the most suitable values for each instance. Initially we take 9 values, from 0.1 to 0.9.

We have tested our randomized constructive algorithm on the 77 test instances. Each instance was run for 1500 iterations, updating the parameters every 500 iterations. The tests were performed on a PC Intel Core i3-2100 (3.1Ghz, 4GB).

5 Results

For comparing results, a lower bound of the number of trucks has been calculated. There are three lower bounds, by weight, by volume and by position. The first one is the number of trucks by weight and it is the total weight of products demanded divided by the maximum weight of a truck. The second one is the number of trucks by volume, the total volume demanded of products divided by the volume of a truck. And the third one is the sum of the total height of all layers demanded divided by the height a truck, this is the number of piles you can build, and if it is divided by the number of positions in a truck $|I| = \lfloor \frac{L}{w^p} \rfloor * \lfloor \frac{W}{l^p} \rfloor$, you obtain the number of trucks per positions. Getting the maximum of the three you obtain a lower bound of the number of trucks, as equation 3.

$$LB = max \left\{ \left\lceil \frac{\sum_{j \in J} n_j * q_j}{Q} \right\rceil, \left\lceil \frac{\sum_{j \in J} l_j * w_j * h_j * n_j}{L * W * H} \right\rceil, \left\lceil \frac{\frac{\sum_{j \in J} h_j * n_j}{H}}{I} \right\rceil \right\} \quad (3)$$

Summing the lower bounds of all the instances, it is necesary at least 511 trucks for supplying the total demand. At Table 1 are showed the results. The first column is the number of trucks needed for supply all the instances.
With the deterministic constructive algorithm version, you need 581 trucks for supplying the demand of all instances and it is made 23518 pallets. Each running time is 0.02 second in overall per instance.
With the randomized version it is necesary 562 trucks and it is built 24442 pallets. Each instance runs for 35,29 second on average. The number of trucks needed for all instances has been reduced in 19 trucks.

	Trucks	Pallets	Time
Bound	511		
Deterministic	581	23518	0,02
Randomized	562	24442	35,29

Table 1. Comparative results

At Table 2 is shown the number of instances that achieve the lower bound with the deterministic version and the randomized version, at the first column. The number of instances that get a gap of one truck in both versions, at column two. The instances with a gap of three trucks at third column.

	Lower bound	+1 truck	+2 trucks	+3 trucks
Deterministic	23	38	16	0
Randomized	30	43	4	0

Table 2. Distance with the lower bound

6 Conclusions and future work

We developed a randomized constructive algorithm based on a joint approach concerning the two subproblems involved: to put products onto pallets and to load the pallets into trucks. The algorithm has been run on 77 instances obtaining a reduction of the number of trucks from the deterministic version to the randomized version.

These results could be improved adding an improved phase when the constructive algorithm is finished. The improvements are:

– Filling a percentage of the container. From the constructive solution, a percentage of the pallets on each truck is extracted. Then, the resulting gaps are filled with new pallets composed of the layers of the extracted pallets and those loaded on other posterior trucks ordered by decreasing volume.

Note that we respect the priority order related to the days, so the layers corresponding to first day will be placed first, and so on.

- Swapping + insertion. This improvement is for those trucks that have been closed because one axle has reached the maximum weight and we try to obtain better results by swapping pallets between consecutive trucks. The idea is to move heavy pallets from the first truck (named A) to the next truck (named B) so that the axles of truck A support lower weights, allowing us to load more pallets. We evaluate the remaining pallets on truck B to check if one of them can be inserted into a gap on truck A. We only consider this improvement if we find successful candidates for both movements, swapping and insertion.

References

1. R. Morabito and S.R. Morales and J.A. Widmer, Loading optimization of palletized products on trucks, Transportation Research Part E: Logistics and Transportation Review vol 36, number 4,pp.285-296, 2000.
2. G. Wäscher and H. Haussner and H. Schumann, An improved typology of cutting and packing problems, European Journal of Operational Research, vol. 183, number 3, pp.1109 - 1130, 2007.
3. A. Bortfeldt and G. Wäscher, Constraints in container loading A state-of-the-art review, European Journal of Operational Research vol. 229, number 1, pp. 1-20, 2013
4. E.E. Bischoff and M.S.W. Ratcliff, Issues in the development of approaches to container loading, Omega vol. 23, number 4, pp. 377-390, 1995.
5. J. Ren and Y. Tian and T. Sawaragi, A tree search method for the container loading problem with shipment priority , European Journal of Operational Research , vol 214, number 3, pp. 526-535, 2011.
6. G. Scheithauer and J. Terno, A heuristic approach for solving the multi-pallet packing problem, technical report, Dresden university, 1996.
7. A. Moura and J.F. Oliveira, A GRASP approach to the container-loading problem, IEEE Intelligent Systems, vol. 20, number 4, pp. 50-57, 2005.
8. A. Lim and H. Ma and C. Qiu and W. Zhu, The single container loading problem with axle weight constraints , International Journal of Production Economics , vol. 144, number 1, pp.358-369, 2013.
9. S. Takahara, Loading problem in multiple containers and pallets using strategic search method, Modeling decisions for artificial intelligence, vol. 3558, pp. 448-456, 2005
10. R.W. Haessler and F.B. Talbot, Load planning for shipments of low density products, European Journal of Operational Research, vol. 44, number 2, pp.289 - 299, 1990.
11. R. Morabito and S. Morales, A simple and effective recursive procedure for the manufacturer's pallet loading problem, The Journal of the Operational Research Society, vol. 49, number 8, pp. 819-828, 1998.
12. H. Gehring and A. Bortfeldt, A Genetic algorithm for solving the container loading problem, International Transactions in Operational Research, vol.4, number 5-6, pp.401–418, 1997.

13. A. Bortfeldt and H. Gehring, A hybrid genetic algorithm for the container loading problem , European Journal of Operational Research , vol. 131, number 1, pp. 143-161,2001.
14. J-L. Lin and B. Foote and S. Pulat and C. Chir-Ho and J.Y. Cheung, Hybrid genetic algorithm for container packing in three dimensions, Proceedings of the Ninth Conference on Artificial Intelligence for Applications, pp.353-359, 1993
15. O.C.B. Araujo and V.A. Armentano, A multi-start random constructive heuristic for the container loading problem , Pesquisa Operacional , vol. 27, pp.311-331, 2007.
16. M. Eley, Solving container loading problems by block arrangement, European Journal of Operational Research , vol. 141, number 2, pp.393-409, 2002.
17. T. Fanslau and A. Bortfeldt , A Tree Search algorithm for solving the container loading problem , INFORMS Journal on Computing ,vol. 22, number 2, pp. 222-235, 2010.
18. B.K.A. Ngoi and M.L. Tay and E.S. Chua , Applying spatial representation techniques to the container packing problem , International Journal of Production Research , vol 32, number 1, pp. 111-123, 1994.
19. Z. Jin and K. Ohno and J. Du , An efficient approach for the three-dimensional container packing problem with practical constraints , Asia-Pacific Journal of Operational Research , vol. 21, number 3, pp. 279-295,2004.
20. L. Junqueira and R. Morabito and D.S. Yamashita, Three-dimensional container loading models with cargo stability and load bearing constraints , Computers and Operations Research , vol. 39, number 1, pp. 74-85, 2012.
21. A. Bortfeldt and H. Gehring and D. Mack, A parallel tabu search algorithm for solving the container loading problem , Parallel Computing , vol."29 , number 5, pp.641-662 , 2003.
22. J. Terno and G. Scheithauer and U. Sommerweiß and J. Riehme, An efficient approach for the multi-pallet loading problem , European Journal of Operational Research , vol. 123, number 2, pp. 372-381, 2000.
23. J. Egeblad and C. Garavelli and S. Lisi and D. Pisinger, Heuristics for container loading of furniture , European Journal of Operational Research , vol. 200, number 3, pp. 881-892, 2010.
24. M.G.C. Resende and R.F. Werneck , A hybrid heuristic for the p-median problem , Journal of Heuristics, vol. 10, number 1, pp.59-88, 2004.
25. E.E. Bischoff and M.D. Marriott, A comparative evaluation of heuristics for container loading, European Journal of Operational Research, vol. 44, number 2, pp. 267 - 276, 1990.