Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 53-2 (2020) 10006–10010

A Hybrid Genetic Algorithm for Pallet Loading in Real-World Applications

Gabriele Ancora [*] Gianluca Palli [*] Claudio Melchiorri [*]

[*] *DEI - University of Bologna, Bologna, Italy (e-mail: {gabriele.ancora2, gianluca.palli, claudio.melchiorri}@unibo.it).*

Abstract: This paper addresses the so called "Distributor's Pallet Packing Problem" in a real industrial scenario. The main goal is to develop an algorithm for loading heterogeneous rectangular boxes on a bin, minimizing some objective functions and also satisfying geometric, stability and fragility constraints. The algorithm must be able to provide, in a reasonable time, the spatial coordinates of the vertices of the placed boxes and also the optimal boxes input sequence. Since this type of combinatorial problem is classified as NP-hard, classical optimization techniques are not suitable. For these reasons, a metaheuristic approach has been developed in order to reduce burden complexity. In particular, a genetic algorithm hybridized with an innovative heuristic technique has been used. The validity and the performance of this algorithm have been tested on several packing instances (orders) provided by an industrial company. The paper is intended as a preliminary study for future developments in the area of industrial container loading problems.

Keywords: genetic algorithm, metaheuristic techniques, NP-hard problems, optimization, packing problems

1. INTRODUCTION

The problem addressed in this paper derives from a real-world application, that is to pack on a bin a sequence of boxes of heterogeneous size, shape and weight, in order to optimize their arrangement. Moreover, some constraints must be satisfied, in terms of geometry, stability and fragility. Typical dimension of the boxes ranges from $7 \times 15 \times 30$ to $75 \times 75 \times 3$ cm, with weights ranging from 5 to 40 kg. The number of boxes on a single bin can vary from 1 to 80, and typically 15 bins per hour should be prepared. This problem belongs to the family of container loading problems. Using the typology defined by Wäscher et al. (2007), the considered problem is of type 3/V/O/M. It is already treated in the literature using different mathematical approaches. We can divide them in two main categories: exact and heuristics/metaheuristic approaches. Due to NP-hardness of the problem (Yaman et al. (2008)), exact optimization techniques (e.g., Bischoff et al. (1995), Al-Shayea (2011), Junqueira et al. (2013)) can not be used in real-world applications because, in general, they involve a very large number of boxes. We choose to tackle this problem by using a metaheuristic framework because of its capacity to try to escape from local minima, exploring more thoroughly the solution space in a very reasonable time. In particular, a hybrid genetic algorithm is used. Without a metaheuristic framework, heuristic techniques (e.g., Bischoff et al. (1990)) are not the most suitable choice, because they are often too greedy, and they usually get trapped in a local optimum, often very far from the global optimum solution. Since we want to approach this problem from a practical packing point of view, we need to take into account several real-world-application constraints, in terms of geometry, stability and fragility. Genetic algorithm approaches have already been treated in the literature (e.g., Wu et al. (2010), Li et al. (2014)) using two different chromosome structures. In this work we choose the one in which only the input boxes sequence is coded (i.e., fixed) because it allows, on average, to find better solutions in a reasonable time with respect to the other in which also the orientation of the boxes is coded. The latter structure, in fact, may require a large number of chromosomes for each generation in order to try to find a good solution, having no degrees of freedom with respect to the boxes input sequence and orientations. Regarding the evaluation process of the chromosome, a new efficient heuristic procedure has been developed and discussed. This paper is organized as follows: in Section 2 a summary of hybrid genetic algorithms is presented, focusing on their adaptation to our problem and introducing the new heuristic procedure. In Section 3 simulation results are presented and discussed, reporting data of a real-world packing application, while in Section 4 some final remarks and plans for future activities are reported.

2. HYBRID GENETIC ALGORITHM

Genetic Algorithms (GAs) are iterative optimization procedures that repeatedly apply GA operators (such as selection, crossover, and mutation) to a group of solutions until some criterion of convergence has been satisfied. In a GA, a search point (solution) is a setting in the search space with dimensions n and it is coded into a string, $x = (x_1, .., x_n)$, which is analogous to a chromosome in biological systems. The string/chromosome is composed of n characters, $x_1, ..., x_n$, which are analogous to the n genes. A set of chromosomes (or individuals) is called *population*. Each iterative step in which a new population is obtained is called *generation*. A GA hybridized with a heuristic is called *Hybrid Genetic Algorithm* (HGA). A basic HGA procedure has the following steps.

(1) Define an objective/fitness function, and set the GA operators (such as population size, parent/offspring ratio, selection method, number of crossovers, and mutation rate);
(2) Generate the initial population in a random way or using heuristics;
(3) Evaluate the objective function for each individual (chromosome or solution) in the initial population;
(4) Generate an offspring population by using GA operators (such as selection/mating, crossover, and mutation);
(5) Evaluate the objective function of each individual in the offspring population using a heuristic procedure;
(6) Decide which individuals to include in the next population;
(7) If a stopping criterion is satisfied, then the procedure is halted. Otherwise, go to step 4.

In our case, n indicates the number of boxes to be palletized and $x_k \in \{1, ..., n\}$ represents the k-th box placed. Therefore, a chromosome $x = (x_1, ..., x_n)$ is an input boxes sequence. As a direct consequence of this choice, the genetic algorithm will run only on this information (using the classical genetic operators such as selection, mutation and crossover), allowing boxes orientations and anchor points to be free optimization variables of the evaluation process. Moreover, in the special case of identical items, this choice allows us to use only one chromosome and one generation to find a suitable solution. In this way, the algorithm will turn into a non-genetic version, reducing calculation time even more.

2.1 Initial population

The chromosomes of the first population are heuristically generated based on the following observations.

- The larger products, in most cases, should be packed first, so that the less bulky boxes can be placed into the small remaining spaces.
- In some cases, it is more convenient to insert boxes starting from the less voluminous ones, for example when many small and a few large boxes need to be palletized, so that the large boxes can be laid on the layers formed by the small ones.

Therefore, to be able to cover as many cases as possible, we divide the first population into three subsets: the main subset is the one composed of those chromosomes created in according to the first observation. The chromosomes belonging to the second subset will be created according to the second observation while those in the third subset will be randomly created. The number of chromosomes belonging to each subset is a parameter of the algorithm.

2.2 Evaluation process

The chromosome evaluation process is performed only if the input sequence coded in the chromosome is feasible, i.e., if it is a generic permutation of the sequence $(1, 2, ...n)$. In this case, for each box to place, a constrained minimization problem has to be solved, with respect to both anchor points and box orientations that represent the only free variables of the problem. Regarding orientations, since the boxes are assumed to rotate only orthogonally and since any of the four vertices of the lower face of the box can coincide with the candidate anchor point, there are 24 possible orientations (2 orientations along x-axis, 2 orientations along y-axis and 2 orientation along z-axis, for each vertex of the lower face). Notice that, in case of fragile box, some orientations may not be recommended. This is taken into account using a suitable fitness function. Regarding anchor points, they are generated using the following heuristic procedure.

A new heuristic procedure From an analysis of heuristics proposed in the literature, one of the most performing is the one based on the concept of corner points (Wu et al. (2010)). The proposed heuristic can be seen as its extension, allowing the algorithm to explore many more configurations within a reasonable computational time. Initially, when the bin is still empty, the only candidate anchor point for the first box will be the point at the centre of the bin. Once a box has been palletized (using both optimal anchor point and optimal orientation according to the fitness function), the algorithm will add the eight vertices of the placed box to the set of candidate anchor points for the following box in the input sequence. The chosen anchor points is not removed from the set, if it results usable again. Given a box to place, a feasibility check is required for all the possible pairs (anchor point, orientation) and, only among all the feasible pairs, the one that minimizes the fitness function is chosen.

Feasibility check A pair (anchor point, orientation) is considered feasible if the related box satisfies these constraints:

- Geometric constraints:
  - · The box must lie completely on the pallet;
  - · There must be no overlap between the placed boxes;
  - · The height of the boxes must not exceed the maximum bin height threshold;
- Stability constraint:
  - · At least s% and v vertices ($v \in \{1, 2, 3, 4\}$) of the lower surface of the box must lie on the pallet wood or on other boxes. Choosing $s \geq 70$ and $v \geq 3$ ensures that its center of mass will always lie in the convex hull of all the contact points (Paquay et al. (2016)).

If all the pairs are infeasible, the fitness value of the chromosome is set to $+\infty$.

Fitness function In order to be able to choose the optimal pair (anchor point, orientation) when the k-th box of the sequence has to be placed ($k \in \{1, ..., n\}$), a suitable fitness function f is needed. It can be write as sum of weighted functions:

$$f_k = w_1 f_{1k} + w_2 f_{2k} - w_3 f_{3k},$$

where f_i are normalized fitness functions and w_i are positive weights ($i \in \{1, ..., 3\}$). In particular:

- f_{1k} is the maximum height of the boxes once box k is placed;
- f_{2k} is the height of the center of mass of the boxes once box k is placed;
- f_{3k} is the moment of inertia of the placed boxes with respect to z-axis passing through the centre of the pallet, once box k is placed;

Once all the boxes have been placed, a global fitness is required for the chromosome selection process. Moreover, a new normalized and positive-weighted fitness function is introduced. It takes into account the boxes fragility constraints:

$$f_4 = 1 - \frac{b}{n},$$

where b indicates the number of palletized boxes using their preferred orientations. Then, the entire fitness is:

$$f = f_n + w_4 f_4.$$

2.3 Genetic Operators

Selection A modified version of the Tournament Method for selection is adopted. Once all the chromosomes have been evaluated, n_{rs} chromosomes are randomly selected and, among these, the best chromosome is chosen for the next generation. This process is repeated until the 80% of the new population is created. Then, the best chromosome of the entire population is selected and replicated to complete the new population. In order to preserve the best chromosome, the crossover and mutation processes will not act on some of its clones.

Crossover and mutation In the literature, crossover and mutation processes are conducted on each newly generated offspring with constant probability P_c and P_m, respectively. In this work, these probabilities are assumed to be generation-varying decreasing functions. By doing so, the algorithm tries to explore the solution spaces in a very exhaustive way at the begin and then it tries to converge to the optimal direction. The crossover is a one-point version: a random location is selected for two parents and the two parts after the crossover point of the two parents are switched over to form two children (Wu et al. (2010)). If the newly formed children are not feasible due to some boxes appear in the children twice while some do not appear at all the chromosome is considered infeasible and its fitness values is set to $+\infty$. Regarding mutation, for each chromosome, two positions are randomly selected and the elements on these positions are swapped.

2.4 Algorithm Output

As mentioned before, the algorithm will provide as output the best input sequence founded during the genetic process and also the set of the three-dimensional coordinates of the vertices of the placed boxes, with respect to a fixed reference frame.

3. SIMULATIONS

The above algorithm is coded in Python 3.7, running on a 2.6 GHz Intel Core i7 processor with 16 GB RAM.

3.1 Algorithm parameters

The algorithm parameters are set as follows.

- Number of chromosomes = 30;
- Number of generations = 8;
- Number of chromosomes randomly selected in the selection process: $n_{rs} = 3$;
- Crossover probability: $P_c(g) = [0.3, 0.28, 0.25, 0.23, 0.2, 0.18, 0.15, 0.1]$;
- Mutation probability: $P_m(g) = [0.7, 0.65, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1]$;
- Fitness weights: $[w_1, w_2, w_3, w_4] = [0.8, 0.8, 0.1, 0.5]$.

3.2 Input instances and graphical results

The 32 instances used for the HGA represent real orders provided from an industrial company. They are reported in Table 1. Some of the most relevant solutions are shown in Figure 1 - 6.

Table 1. Instances

Instance	N. of boxes	Identical boxes	Simulation time
1	1	Yes	42 ms
2	3	Yes	130 ms
3	4	Yes	150 ms
4	5	Yes	160 ms
5	6	Yes	0.26 s
6	7	Yes	0.34 s
7	8	Yes	0.48 s
8	21	Yes	5.4 s
9	26	Yes	9.1 s
10	29	Yes	52 s
11	30	Yes	54 s
12	58	Yes	4.2 min
13	71	Yes	6.8 min
14	75	Yes	7.9 min
15	2	No	1.3 s
16	4	No	7.1 s
17	5	No	11.2 s
18	6	No	14.5 s
19	7	No	21.3 s
20	8	No	32,8 s
21	9	No	33.2 s
22	11	No	34.2 s
23	12	No	1.3 min
24	13	No	1.4 min
25	14	No	1.6 min
26	15	No	2.2 min
27	17	No	3.6 min
28	18	No	3.9 min
29	20	No	4,2 min
30	23	No	7.3 min
31	26	No	8.8 min
32	58	No	10.2 min

4. CONCLUSION AND FURTHER WORKS

The bin packing problem is very common in industrial companies. To solve it, a genetic algorithm coupled with a new performing heuristic has been used, taking into account geometric, stability and fragility constraints. One of the most important features of our algorithm is its capability and efficiency to provide as output, in addition to the spatial coordinates of the vertices of the placed boxes, also the optimal boxes input sequence. From an analysis of the simulation results, the proposed HGA proves to be very efficient and flexible for all the input instances. Possible future works could concern in the investigation of some machine learning techniques for learning some algorithm parameters, like fitness functions weights, increasing performance and quality of the solutions even more. Furthermore, additional practical constraints can be also added to the model.

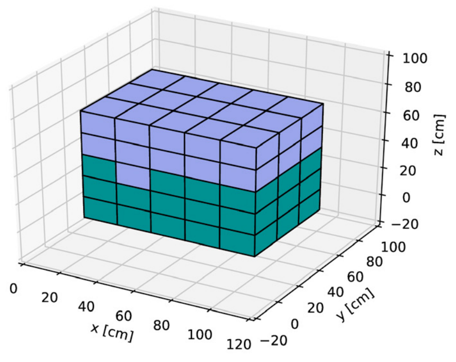

Fig. 1. Optimal solution related to instance 14.

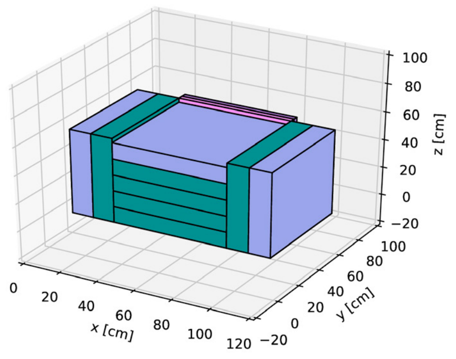

Fig. 2. Optimal solution related to instance 22.

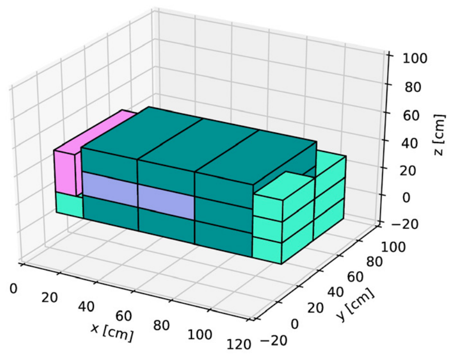

Fig. 3. Optimal solution related to instance 27.

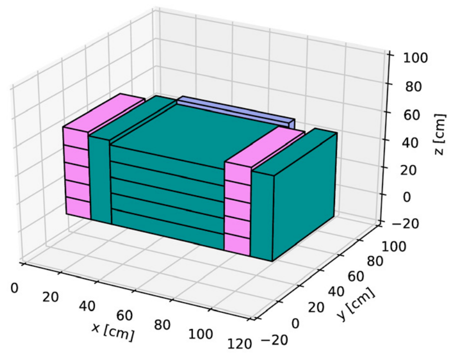

Fig. 4. Optimal solution related to instance 28.

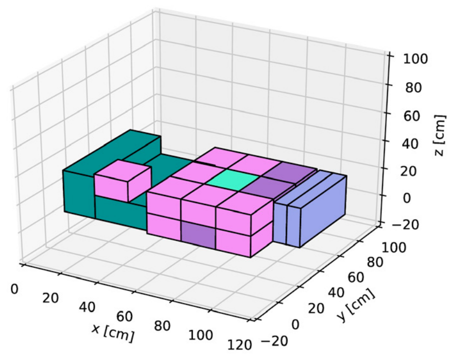

Fig. 5. Optimal solution related to instance 30.

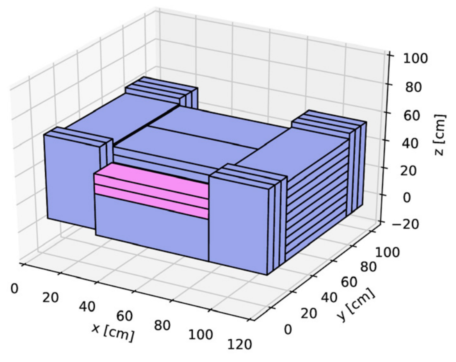

Fig. 6. Optimal solution related to instance 32.

REFERENCES

G. Wascher, H. Haubner, H. Schumann An improved typology of cutting and packing problems *EJOR*, 183: 1109–1130, 2007.

E.E. Bischoff, F. Janetz, M.S.W. Ratcliff Loading pallets with non-identical items *EJOR*, 84:681–692, 1995.

E.E. Bischoff, M.D. Marriott A comparative evaluation of heuristics for container loading *EJOR*, 44:267–276, 1990.

A. Lodi, S. Martello, D. Vigo Heuristic algorithms for the three-dimensional bin packing problem *EJOR*, 141: 410–420, 2002.

Y. Wu, W. Li, M. Goh, R. De Souza Three Dimensional Bin Packing Problem with Variable Bin Height *EJOR*, 202:347–355, 2010.

X. Li, Z. Zhao, K. Zhang A genetic algorithm for the three-dimensional bin problem with heterogeneous bins *Industrial and Systems Engineering Research Conference*, 2014.

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

L. Junqueira, R. Morabito, D.S. Yamashita, H.H. Yannasse Optimization models for the three-dimensional container loading problem with practical constraints *Modeling and Optimization in Space Engineering, Springer*, 271–293, 2013.

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

A.M., Al-Shayea Solving the three-dimensional palet-paking problem using mixed 0–1 model *Journal of Service Science and Management*, 4:513–522, 2011.

H. Yaman, A. Şen Manufacturers mixed pallet design problem *European Journal of Operational Research*, 186: 826–840, 2008.