A 9-fold Partition Heuristic for Packing Boxes into a Container

Lauro Lins [1] *Sóstenes Lins* [2] *Reinaldo Morabito* [3]

[1]Departamento de Informática da UFPE
50740-540 Recife, Brazil
ldl@di.ufpe.br

[2]Departamento de Matemática da UFPE
Cidade Universitária
50740-540 Recife, PE, Brazil
sostenes@dmat.ufpe.br

[2]Departamento de Engenharia de Produção da UFSCar
Caixa Postal 676
13565-905 São Carlos, SP, Brazil
morabito@power.ufscar.br

Abstract

In this paper we propose a simple recursive uniform algorithm for the problem of packing boxes into a container. We are particularly concerned with the special case where all boxes are identical. The algorithm has as input a fixed triad (an object which generalizes a plane graph) and can be specialized to recover a previous result in dimension two by Morabito and Morales. The study has practical applications for the problems of pallet and container loading where the number of rectangles and boxes to be loaded is small.

Keywords: 3-D packing, combinatorial optimization, pallet/container loading

1 Introduction

In this article we present a simple recursive algorithm for the problem of packing boxes into a larger box, henceforth referred to as a *container*. The algorithm is based on the presentation of feasible packing as feasible depth assignments in a *triad* (an object which generalizes pairs of dual graphs). To each such assignment, there corresponds a partition of the containers into specific subcontainers. Each subcontainer

can be recursively partitioned into further subcontainers, according to the specifications in the triad. A subcontainer which is not partitioned is loaded homogeneously with the boxes. The algorithm performs a search over all feasible depth assignments of the problem to find the most valuable depth assignment, i.e., the one which induces a maximum number of packed boxes. To organize and simplify the search, the procedure explores recursion techniques, according to the algorithm of Morabito and Morales [14] proposed for the 2-D case. The procedure could be easily specialized to pack rectangles into a larger rectangle (2-D case) or bars into a larger bar (1-D case).

The *3-D packing problem* (3PP) has various practical applications. In particular, in the loading of products (packed into boxes) over pallets, or inside containers or safe-trucks. Such problems appear in the logistic activities of transporting and storing goods or supplies and, depending on the scale of the supply/distribution chain, a small increase in the volume of products loaded over a pallet, or inside a container, can result in substantial savings.

Several studies treating packing problems have been reported. See, for instance the surveys of Dowsland and Dowsland [8], Dyckhoff and Finke [9], Sweeney and Paternoster [18], Bischoff and Waescher [5]. Other relevant references are Abdou and Yang [1], Morabito and Arenales [13], Bischoff and Ratcliff [4], Nelissen [15], Scheithauer and Terno [17], and Miyazawa and Wakabayashi [12].

In the present study we were particularly interested in the case of boxes of equal size. However, the algorithm can easily be adapted to the more general case, where there is a certain number of boxes and containers with distinct sizes. These generalizations can be strengthened to handle demand requirements. In this case, the method treated here plays the role of a column generator to a simplex approach. These matters will be treated elsewhere.

Here, we also assumed that the boxes, available in large quantity, are to be orthogonally arranged inside the container (i.e. with their faces parallel to those of the container) and that no vertical orientation for the box loading is fixed. This assumption provides, in general, six nonequivalent ways to position a box. The algorithm can be easily adapted to the case in which there is a fixed vertical orientation. In this case we have at most two nonequivalent ways to position the box.

3PP is an NP-hard problem and can be formulated as an integer linear programming, for example, extending the (0,1)-model in Beasley [3], originally proposed for the 2-D case. Or applying the (0,1)-model in Tsai et al. [19] based on disjunctive restrictions. Exact methods of branch-and-bound type exploring bounds coming from the surrogate and Lagrangean relaxation can be defined following [3] and [19].

Alternatively, Dowsland [7] presented an interesting approach for the 2-D packing, which can also be extended to treat 3PP. Basically, the approach consists of finding

the maximum stable set of a particular finite graph where the nodes correspond to the possible positioning of the boxes inside the container. Two nodes are the ends of an edge if the box positions which they represent overlap.

However, because of the size of practical problems, both the (0,1)-models mentioned above and Dowsland's graph approach are generally too big to be computationally treated. In this way, most methods found in the literature are heuristics, such as those listed in the previous paragraphs.

The recursive procedure treated here, namely 9-fold partition, is also heuristic. However, it deterministically finds the optimum over a well-defined class of patterns, which will contain an optimum in the vast majority of the cases.

To illustrate this, we present the results obtained by the procedure for a non-trivial example. An analysis of the performance of the procedure for solving a number of randomly generated examples, as well as examples derived from a case study of a Brazilian distribution center, is being compiled and will be reported shortly.

2 The 9-Fold Partition

The partition of a rectangle into smaller rectangles can be specified by a *duet* (G_x, G_y). The two graphs G_x and G_y have a $1-1$ correspondence between their edges. They are also equipped with *depth functions* on the vertices. The position and dimensions of a subcontainer S are specified by a pair of corresponding edges labelled e_S in the following way: the starting point is (x_1, y_1), where x_1 is the end of e_S in G_x with the smallest depth and y_1 is the end of e_S in G_y with the smallest depth. The dimension of the subcontainer is $(x_2 - x_1) \times (y_2 - y_1)$, where x_2 and y_2 are the depths of the other ends of e_S in G_x and G_y, respectively. The graphs G_x and G_y of a duet are called *profiles of the duet.*

Below we present the duet which corresponds to the canonical partition of the Morabito and Morales [14] recursion.

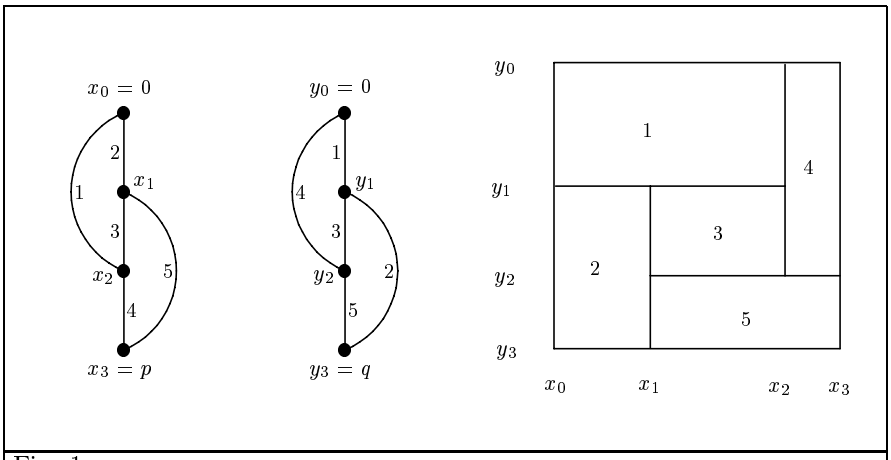

Fig. 1: A duet and its 5-fold rectangle partition: basis for the Morabito-Morales recursion

What is important in this choice of this 5-fold partition, is that it is the simplest non-guillotine pattern which, by means of recursions applied to the subrectangles, can produce amazingly rich patterns (see Fig. 4) attaining the optimum in the vast majority of cases. Each level of the recursion is a nested loop involving 4 variables, which correspond to the values of the depth assignment of the intermediate vertices in the duet: x_1, y_1, x_2, y_2. The process can be seen as a depth-first search where each edge labeled i in the j-th profile of a duet is recursively replaced by the whole j-profile. For instance, Fig 2 depicts the replacement of edges previously labeled 1 (see Fig 1) in the first and second profiles by, respectively, the first and second profiles.

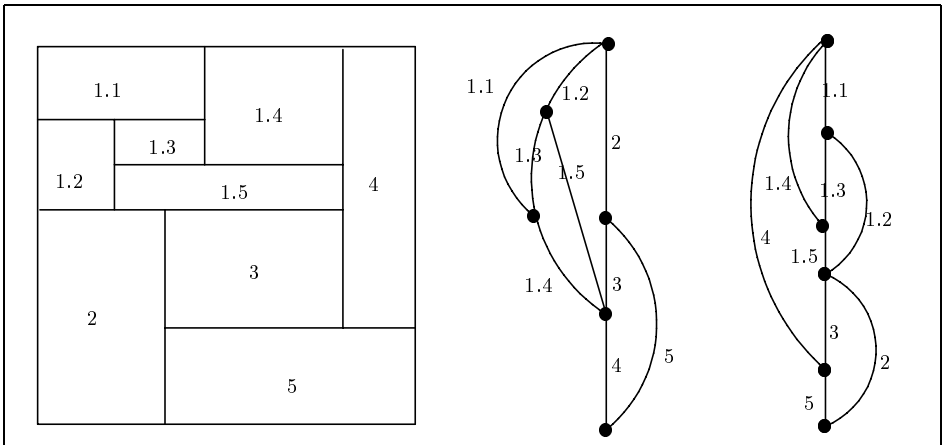

Fig. 2: Recursive replacements of edges labeled 1 by profiles

The central idea of this paper is to define the 3-D counterpart of this partition. We call it the *canonical 9-fold partition* and it is displayed below. Starting with sub-

containers $1, 4, 5, 7$ and 8 in the left-hand picture, we add subcontainers 3 and 9 as shown in the middle and, finally, add subcontainers 2 and 6 to obtain the complete pattern on the right.

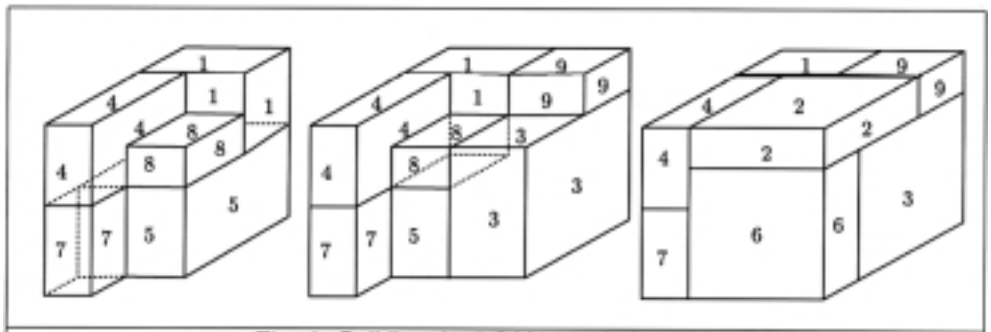

Fig. 3: Building the 9-fold canonical partition

The canonical 9-fold partition can be represented by a *triad* (G_x, G_y, G_z), which generalizes the duet in an obvious way. Fig 4 shows the triad representation of the 9-fold canonical partition, given by its three profiles.

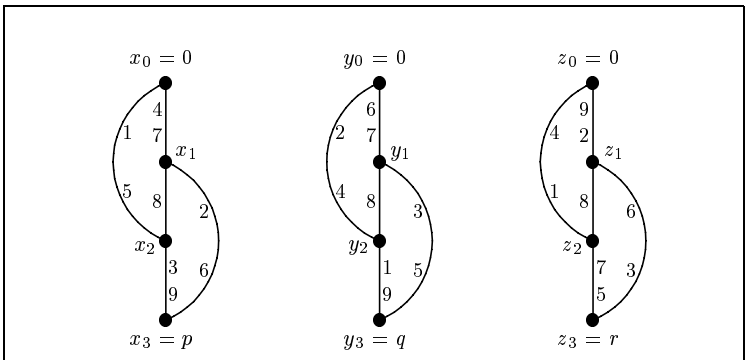

Fig. 4: Triad T_0 corresponding to the canonical 9-fold partition

Note that like subrectangle 3 in Fig 1, subcontainer 8 is totally internal. This can be seen in the triad above. This triad, which is the smallest non-trivial 3-D non-guillotine pattern, is symmetrical with respect to x, y, z, and produces patterns that surpass the layers-knapsack approach for the 3-D case, as we will see in the next section. Similar to the 2-D case, the basic cost factor in each call of the recursive procedure is a nested loop in, this time, *six* variables, corresponding to the values

of the depth assignment of the intermediate vertices of the triad: $x_1, y_1, z_1, x_2, y_2, z_2$. The increase from 4 to 6 makes a considerable difference due to the intrinsic exponential search involved and therefore accuracy is extremely important in the design of an algorithm with an acceptable running time.

3 An Example

We produced an example to show that a recursive algorithm based on the 9-fold partition is superior to the layers-knapsack approach for the 3-D case. Consider the problem of packing $(9 \times 6 \times 5)$ boxes into a $(61 \times 44 \times 50)$-container.

3.1 Solution Based on 2-D Approaches

3.1.1 Fixed Orientation

If the vertical orientation of the boxes is fixed, then the optimal solution is depicted in Fig. 5, consisting of $\lfloor 50/9 \rfloor$ layers each with 89 boxes positioned as shown in the figure, yielding a total of 445 boxes ($\lfloor z \rfloor$ denotes the largest integer less than or equal to z). This example is adapted from [14] and comes from an optimal solution at recursivity level $n = 3$ of the recursive 2-D Morabito-Morales algorithm.

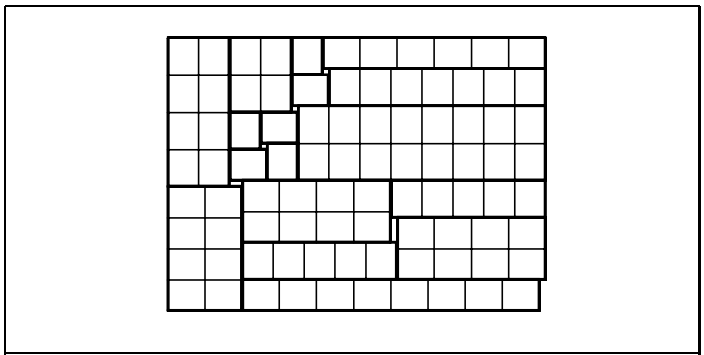

Fig. 5: $\lfloor 50/9 \rfloor$ layers with 89 $(9 \times 6 \times 5)$-boxes: 445 boxes

3.1.2 Non Fixed Orientation

If the vertical direction of the boxes is no longer fixed, then the loading can be improved. Let us consider what we get by applying a layers-knapsack approach to the $(61 \times 44 \times 50)$-container and $(9 \times 6 \times 5)$-boxes example. We have three possible directions to stack the layers.

By using the left-to-right direction to stack the boxes, the three solutions for the 2-D problems using the Morabito-Morales method are :

- 1. (44×50)-rectangle and (6×5)-faces: 73 boxes (optimal solution - level 3)
- 2. (44×50)-rectangle and (5×9)-faces: 48 boxes (optimal solution - level 1)
- 3. (44×50)-rectangle and (6×9)-faces: 37 boxes (optimal solution - level 1)

Thus the associated knapsack problem is

$$\max\ 73x_1 + 48x_2 + 37x_3$$

$$\text{subject to } 9x_1 + 6x_2 + 5x_3 \leq 61$$

$$x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, \text{integer}$$

where x_1, x_2, x_3 correspond respectively to the number of layers using the patterns of the 2-D problems 1, 2, 3, as above.

The optimal solution for this knapsack problem is $x_1 = 0, x_2 = 10, x_3 = 0$. Therefore, we obtain a solution with $10 \times 48 = 480$ boxes (recall that the solution with fixed vertical orientation loaded only 445 boxes).

By using the back-to-front direction to stack the boxes the three solutions for the 2-D problems are :

- 4. (61×50)-rectangle and (6×5)-faces: 101 boxes (optimal solution - level 1)
- 5. (61×50)-rectangle and (5×9)-faces: 67 boxes (optimal solution - level 1)
- 6. (61×50)-rectangle and (6×9)-faces: 53 boxes (optimal solution - level 1)

Thus the associated knapsack problem is

$$\max\ 101x_1 + 67x_2 + 53x_3$$

$$\text{subject to } 9x_1 + 6x_2 + 5x_3 \leq 44$$

$$x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, \text{integer}$$

where x_1, x_2, x_3 correspond respectively to the numer of layers using the patterns of the 2-D problems 4, 5, 6, as above.

The optimal solution for this knapsack problem is $x_1 = 1, x_2 = 5, x_3 = 1$. Therefore, we obtain a solution with $1 \times 101 + 5 \times 67 + 1 \times 53 = 489$ boxes, which is better than the previous one with 480 boxes.

Finally, we stack the boxes in the top-to-bottom direction. The three solutions for the 2-D problems are :

- 7. (61×44)-rectangle and (6×5)-faces: 89 boxes (optimal solution - level 3)
- 8. (61×44)-rectangle and (5×9)-faces: 58 boxes (normal completion - level 3)
- 9. (61×44)-rectangle and (6×9)-faces: 46 boxes (optimal solution - level 1)

In order to check if the 2-D solution obtained for problem 8 is optimal, we solved the LP-relaxation of the 0-1 model for this problem proposed by Beasley [3] (see section 1). The optimal relaxed solution is less than 59 and, therefore, the solution above (58 boxes) is indeed optimal.

The associated knapsack problem is

$$\max 89x_1 + 58x_2 + 46x_3$$

$$\text{subject to } 9x_1 + 6x_2 + 5x_3 \leq 50$$

$$x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, \text{integer}$$

where x_1, x_2, x_3 correspond respectively to the number of layers using the patterns of the 2-D problems 7, 8, 9, as above.

The optimal solution for this knapsack problem is $x_1 = 5, x_2 = 0, x_3 = 1$. Therefore, we obtain a solution with $5 \times 89 + 1 \times 46 = 491$ boxes, which is the best solution based on the 2-D approaches above. Observe that a trivial upper bound for the number of boxes is $\lfloor 61 \times 44 \times 50/9 \times 6 \times 5 \rfloor = 497$.

3.2 The 9-fold Partition Approach

In this section we show that the 3-D recursive algorithm based on the 9-fold partition can produce better solutions than the 2-D approaches for such an example.

3.2.1 Level 1

The 3-D solution obtained at level $n = 1$ of the recursion already loads 490 boxes in a complicated non-guillotine pattern involving 8 subcontainers, as shown below (Fig. 6). This solution required a microcomputer runtime of 5 minutes.

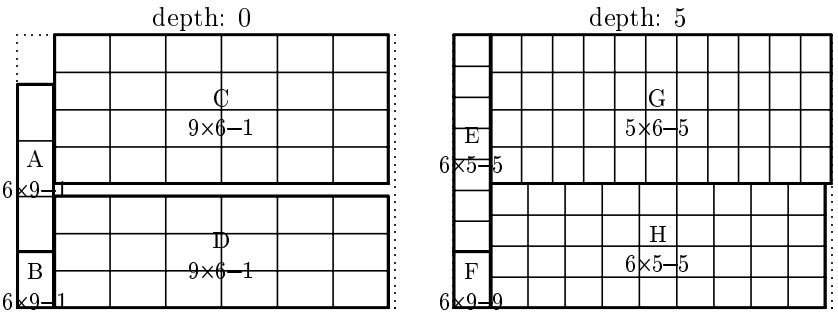

Fig. 6: 3-D solution: level 1 — 490 boxes — 8 Subcontainers

In Fig 6 the 490 boxes are packed using 8 subcontainers as follows:

- Subcontainer A: 1-stack of (1×3)-arrangement of $(6 \times 9 \times 5)$-boxes: 3 boxes;
- Subcontainer B: 1-stack of (1×1)-arrangement of $(6 \times 9 \times 5)$-boxes: 1 box;
- Subcontainer C: 1-stack of (6×4)-arrangement of $(9 \times 6 \times 5)$-boxes: 24 boxes;
- Subcontainer D: 1-stack of (6×3)-arrangement of $(9 \times 6 \times 5)$-boxes: 18 boxes;
- Subcontainer E: 5-stack of (1×7)-arrangement of $(6 \times 5 \times 9)$-boxes: 35 boxes;
- Subcontainer F: 9-stack of (1×1)-arrangement of $(6 \times 9 \times 5)$-boxes: 9 boxes;
- Subcontainer G: 5-stack of (11×4)-arrangement of $(5 \times 6 \times 9)$-boxes: 220 boxes;
- Subcontainer H: 5-stack of (9×4)-arrangement of $(6 \times 5 \times 9)$-boxes: 180 boxes.

Note in Fig. 6 that the first four blocks start at depth 0 and the last four at depth 5.

3.2.2 Higher Levels

Let us see what happens at higher levels, for example, at level 2:

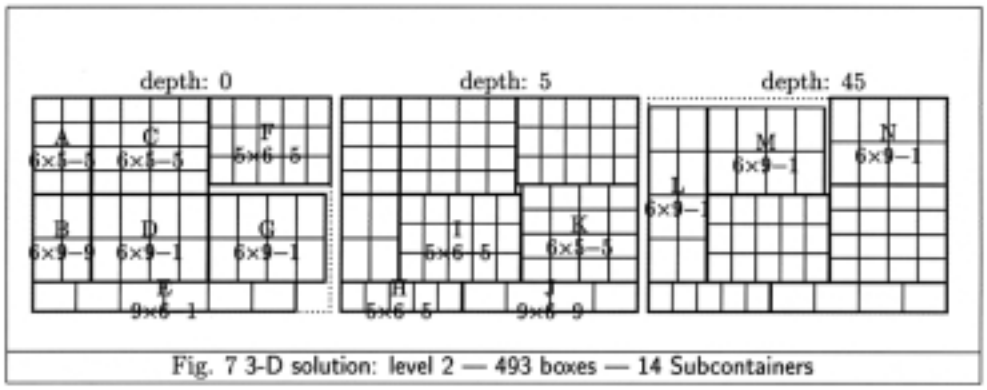

Fig. 7 3-D solution: level 2 — 493 boxes — 14 Subcontainers

Fig. 7 shows that the packing pattern obtained at level 2 (493 boxes) is better than the best pattern found by the 2-D approaches (491 boxes). Let us also see what happens at level 3:

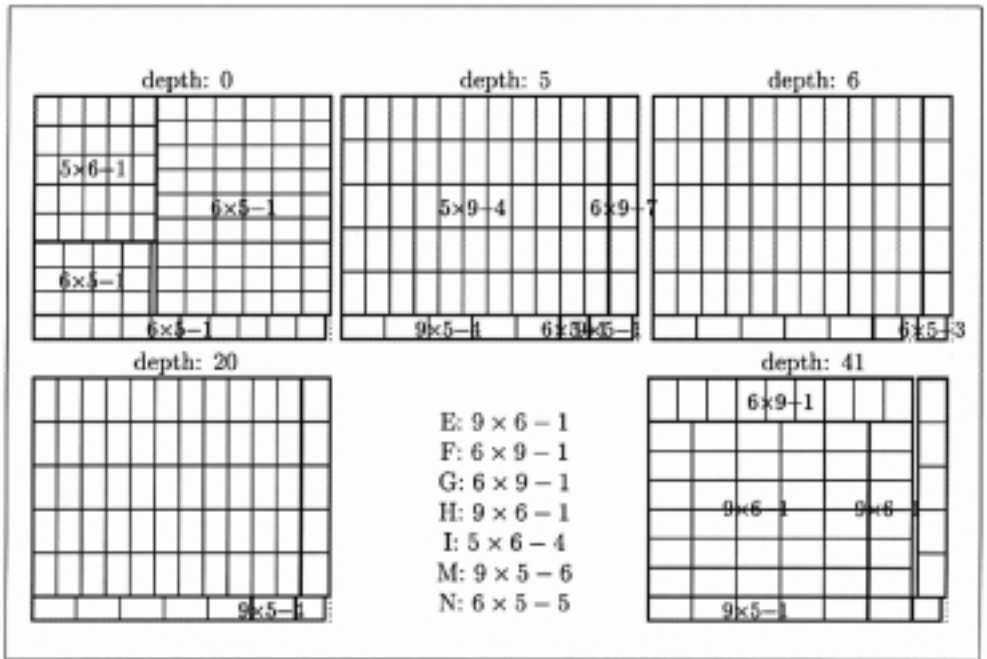

Fig. 8 3-D solution: level 3 — 494 boxes — 19 Subcontainers — Interrupted

The number of boxes put in the subcontainers A, ..., S are

A=5	B=5	C=20	D=10	E=4	F=1	G=3	H=1	I=8	J=30
K=220	L=12	M=6	N=35	O=54	P=35	Q=25	R=11	S=9,	

for a total of 494 boxes.

Limiting the runtime in one day, the best 3-D pattern obtained at level 3 (Fig. 8) loaded 4 boxes more than the pattern found at level 1 (Fig. 6), and 1 box more than the pattern at level 2 (Fig. 7). It loaded 3 boxes more than the best solution found by the 2-D approaches.

In order to check for the optimality of this solution, we solved the LP-relaxation of the 0-1 model in Beasley [3] for three different 2-D problems:

- 1. (50×44)-rectangle and (5×6), (5×9) and (6×9) faces

- 2. (61×44)-rectangle and (5×6), (5×9) and (6×9) faces
- 3. (61×50)-rectangle and (5×6), (5×9) and (6×9) faces

The minimum upper bound derived from the LP-solutions for these three problems was 496 boxes, that is, a gap of only two boxes relative to the solution of Fig. 8. Running the Morabito-Morales 2D heuristic for the first of these problems, at level 20 provided a lower bound of 10 for the wasted area of a 44×50 slice. Accepting this as optimal we found that the minimum wasted volume was $61 \times 10 = 610$. Since

$$\lfloor (61 \times 50 \times 44 - 610)/(9 \times 6 \times 5) \rfloor = 494,$$

this would imply that our solution in Fig. 8 is optimal.

4 Conclusion and Further Developments

The example of the previous section suggests that the recursive 9-fold approach introduced in this paper is interesting and merits further consideration. Implementation aspects and details to improve the running time are currently being developed. Triad representation has revealed itself as an important tool to deal with symmetry and isomorphism issues (Herz [10]).

Consideration of these issues has so far reduced the running time by a factor of five. Despite this, the running time for the relatively difficult example of the previous section is still prohibitively high in practice: five minutes for level 1, several hours for level 2 and almost one day for the first solution for level 3 (interrupted before termination). This leaves a gap of only 2 boxes relative to the upper bound generated by the LP-relaxation of a 0-1 formulation for the problem.

It is of the utmost importance to decrease these running times, since the solutions with the layers-knapsack approach were obtained instantaneously. An effective way to improve the running time of the algorithm, which has not been implemented yet, is to use the best solution of the layers-knapsack method to provide a good initial lower bound. Indeed, the effort required to solve the 9 associated 2-D problems, plus the 3 knapsack problems, should be more than compensated by presenting a lower bound which will most likely curtail the implicit enumeration considerably. Another technique involves an adaptative thickening of the discretization. We believe that an appropriate calibration of various heuristic techniques will make the running times acceptable for most problems from practical situations.

5 Acknowledgements

The second author acknowledges the financial support of Pronex (proc. 107/97 — Complexity of discrete structures), of UFPE, of CNPq (proc. 30.1103/80) and of

Finep. The third author has received financial support of Protem of CNPq (proc. 680082/95-6 — Cutting and packing), of UFSCar, of CNPq (proc. 522973/95-7) and of Fapesp (proc. 9522-0 and 97/13930-1).

References

[1] G. Abdou and M. Yang (1994). A systematic approach for the three-dimensional palletization problem. *Int.J.Prod.Res.* 32(10), 2381-2394.

[2] F. Barnes (1979). Packing the maximum number of *mxn* tiles in a large *pxq* rectangle. *Discrete Mathematics* 26, 93-100.

[3] J. Beasley (1985). An exact two-dimensional non-guillotine tree search procedure. *Oper.Res.* 33, 49-64.

[4] E. Bischoff and M. Ratcliff (1995). Loading multiple pallets. *J.Opl.Res.Soc.* 46, 1322-1336.

[5] E. Bischoff and G. Waescher (eds.) (1995). Special issue on cutting and packing. *Eur.J.Opl.Res.* 84(3), 503-712.

[6] T. Cormen, C. Leiserson and R. Rivest. Introduction to Algorithms. McGraw-Hill (1990).

[7] K. Dowsland (1987). An exact algorithm for the pallet loading problem. *Eur.J.Opl.Res.* 31, 78-84.

[8] K. Dowsland and W. Dowsland (1992). Packing problems. *Eur.J.Opl.Res.* 56, 2-14.

[9] H. Dyckhoff and U. Finke (1992). *Cutting and packing in production and distribution: Typology and bibliography, Springler-Verlag Co, Heildelberg.*

[10] J. Herz (1972). Recursive computational procedure for two dimensional stock cutting. *IBM J.Res.Develop.* 16, 462-469.

[11] S. Lins (1998). Triads: a generalization of planar duality on graphs. *In preparation*

[12] F.K. Miyazawa and Y. Wakabayashi (1997). An algorithm for the three-dimensional packing problem with asymptotic performance analysis. Algorithmica 18(1), 122-144.

[13] R. Morabito and M. Arenales (1994). An AND/OR-graph approach to the container loading problem. *Int.Trans.Opl.Res.* 1(1), 59-73.

[14] R. Morabito and S. Morales (1998). A simple and effective recursive procedure for the manufacturer's pallet loading problem. Journal of the Operational Research Society, 49, 819-828.

[15] J. Nelissen (1995). How to use structural constraints to compute an upper bound for the pallet loading problem. *Eur.J.Opl.Res.* 84, 662-680.

[16] A. Nijenhuis and H. Wilf. Combinatorial Algorithms Academic Press (1978).

[17] G. Scheithauer and J. Terno (1996). The G4-heuristic for the pallet loading problem. *J.Opl.Res.Soc.* 47, 511-522.

[18] P. Sweeney and E. Paternoster (1992). Cutting and packing problems: A categorized, application-oriented research bibliography. *J.Opl.Res.Soc.* 43, 691-706.

[19] R. Tsai, E. Malstrom and W. Kuo (1993). Three dimensional palletization of mixed box sizes. *IEE Trans.* 25(4), 64-75.