

Discrete Optimization

# Solving the pallet loading problem

Gustavo H.A. Martins<sup>a</sup>, Robert F. Dell<sup>b,\*</sup>

<sup>a</sup> Center for Naval Systems Analyses, Brazilian Navy, Rio de Janeiro, RJ 20000, Brazil

<sup>b</sup> Operations Research Department, Naval Postgraduate School, Monterey, CA 93943, USA

Received 30 December 2005; accepted 13 November 2006

Available online 8 January 2007

## Abstract

This paper presents new bounds, heuristics, and an exact algorithm for the Pallet Loading Problem (PLP). PLP maximizes the number of boxes placed on a rectangular pallet. All boxes have identical rectangular dimensions and, when placed, must be located completely within the pallet. Boxes may be rotated 90° so long as they are placed with edges parallel to the pallet's edges. The set of all PLP instances with an area ratio (pallet area divided by box area) less than 101 boxes can be represented by 3,080,730 equivalent classes. Our G5-heuristic finds optimal solutions to 3,073,724 of these 3,080,730 classes and in the remaining 7006 classes only differs from the best known bound by one box. We develop three other heuristics that solve another 54 instances. Finally, we solve the 6952 remaining classes with our exact HVZ algorithm. Only a subset of these classes has been solved previously.

© 2006 Elsevier B.V. All rights reserved.

*Keywords:* Packing; Pallet loading; Combinatorial optimization

We represent each instance of a Pallet Loading Problem (PLP) with a quadruplet  $(X, Y, a, b)$ . We have a rectangular pallet with length  $X$  and width  $Y$  ( $X \geq Y$ ), and a rectangular box with length  $a$  and width  $b$  ( $a \geq b$ ). Boxes may be rotated 90° so long as they are placed with edges parallel to the pallet's edges, i.e., the placement must be orthogonal. We assume, without loss of generality, that  $X, Y, a, b$  are positive integers (Bischoff and Dowsland, 1982). We also assume that at least one box can be placed in the pallet ( $X \geq a$  and  $Y \geq b$ ).

The PLP is encountered when trying to maximize the number of identical boxes with dimensions  $a$

and  $b$ , placed in a pallet with dimensions  $X$  and  $Y$  where each box has a “this side up” restriction [e.g., Bischoff and Dowsland (1982)]. Even without such a restriction, operational considerations may dictate the use of vertical layers with the same height. Considerations regarding stability and safety of the boxes imply the use of orthogonal placement [e.g., Dowsland (1987a), Nelissen (1995), and Young-Gun and Maing-Kyu (2001)]. PLP is also present in some cutting stock and floor design settings.

Dowsland (1984) establishes that PLP instances can be divided into classes with the same optimal placement pattern. Martins and Dell (2007) define the Minimum Size Instance (MSI) of an equivalence class of PLP, show that every class has one and only

\* Corresponding author.

E-mail address: [dell@nps.edu](mailto:dell@nps.edu) (R.F. Dell).

one MSI, and then enumerate all 3,080,730 equivalence classes with an area ratio (pallet area divided by box area) smaller than 101 boxes. The complete set of instances can be accessed at <http://www.palletloading.org>. This paper presents new bounds, heuristics, and an exact algorithm for PLP and demonstrates their use by solving the MSI from all 3,080,730 equivalence classes. Only a subset of these classes has been solved previously.

## 1. Partition, H-box, V-box, and block

Given an instance  $(X, Y, a, b)$  of PLP,  $N(X, Y, a, b)$  is the number of boxes in an optimal placement and  $W(M, X, Y, a, b) \equiv X * Y - M * a * b$  is the waste encountered for  $M$  boxes where  $M \leq \lfloor (X * Y) / (a * b) \rfloor$ . Also  $A_x \equiv \lfloor X/a \rfloor$ ,  $A_y \equiv \lfloor Y/a \rfloor$ ,  $B_x \equiv \lfloor X/b \rfloor$ , and  $B_y \equiv \lfloor Y/b \rfloor$ .

Let  $(n, m)$  denote an ordered pair of non-negative integers satisfying  $n * a + m * b \leq S$  for a pallet dimension  $S$ , ( $X$  or  $Y$ ). Then, the ordered pair  $(n, m)$  is called a *feasible partition* of  $S$ . If  $n$  and  $m$  also satisfy  $0 \leq S - n * a - m * b < b$  then  $(n, m)$  is called an *efficient partition* of  $S$  (Bischoff and Dowsland, 1982). Dowsland (1984) shows that if two instances of PLP possess the same set of efficient partitions for both the pallet width and length, then both instances share the same set of optimal placements. If  $n$  and  $m$  satisfy  $n * a + m * b = S$ , then  $(n, m)$  is called a *perfect partition* of  $S$  (Dowsland, 1984). The set of all perfect partitions of pallet dimension  $S$  is denoted  $P(S, a, b)$ .

We define an *H-box* (*V-box*) as a box with its largest dimension oriented horizontally (vertically). We also define a *block* as a rectangular subset of a placement such that no box is only partially included in the rectangle. In other words, a block partitions the boxes of a placement into two groups, those inside and those outside the block. If all boxes in a block have the same orientation (V-boxes or H-boxes), then the block is called a *homogeneous block* (Scheithauer and Terno, 1996). A homogeneous block of V-boxes (H-boxes) is a *V-block* (*H-block*).

A *hollow block* or a *diagonal block* (Fig. 1) contains boxes in one orientation, across one diagonal of the block, surrounded by boxes in the other orientation. Each homogeneous block placed along the diagonal (away from the diagonal) of the pattern is called a *diagonal element block*, or *diagonal element*

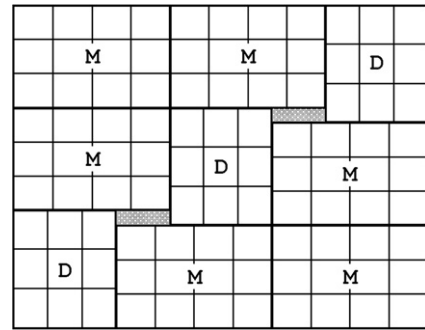


Fig. 1. Feasible placement for instance (85, 66, 8, 7). Diagonal elements are identified with the letter D. Main elements are identified with the letter M.

(*main element blocks*, or *main elements*). An unused region is called a *hole*.

## 2. Selected PLP heuristics

One-, two-, four-, and five-block heuristics (e.g., Nelissen, 1993) are among the simplest to implement. The *one-block heuristic* places all boxes with the same orientation; a constant-time computation determines the best orientation. All  $n$ -block heuristics select a size for the first block, then for the second block, until the last block is placed.

The heuristic proposed by Scheithauer and Terno (1996), the G4-heuristic, recursively applies a four-block heuristic. There is no *a priori* limitation on the number of recursive calls. The heuristic proposed by Morabito and Morales (1998), referred to here as M&M, uses 1st-order cuts and generates 1st-order patterns. A 1st-order pattern is a pattern generated by successive guillotine or 1st-order cuts. A *guillotine cut* divides a piece of larger stock in two smaller pieces with a cut from one side to the other [e.g., Gilmore and Gomory (1965), Christofides and Whitlock (1977), Wang (1983), and Oliveira and Ferreira (1990)]. A cut is 1st-order if it produces five new rectangles arranged in such a way as not to form a guillotine cutting pattern (Arenales and Morabito, 1995).

Nelissen (1993) describes a family of *diagonal heuristics*, which generate diagonal block patterns and proposes the *angle heuristic* and the *recursive angle heuristic*. These heuristics place blocks of boxes, with a given orientation, along the diagonal of the pallet, surrounded by boxes with the opposite orientation. The main difference between the diago-

nal heuristics and the angle heuristics is that the placement obtained with these latter procedures may not be symmetric, and holes can have different sizes. The diagonal heuristic restricts each diagonal block to have a width of only one box; the angle heuristic is recursive and has longer run time than the diagonal heuristic.

### 3. The hollow block heuristic

We propose the *Hollow Block* (HB) heuristic. It has more flexibility than the diagonal block heuristic because it allows the diagonal elements to have more than one box across the width, and achieves shorter run times than the angle heuristic because it only looks for patterns generated by perfect partitions of the length and width. It only considers blocks that cover the whole pallet, with area wasted only by holes.

The HB heuristic selects compatible pairs of perfect partitions for both length and width, and uses these partitions to define the dimensions of the elements of the hollow block. Let  $(i, j) \in P(X, a, b)$ , for  $i > 0$  and  $j > 0$ , and  $(f, g) \in P(Y, a, b)$ , for  $f > 0$  and  $g > 0$ , be the selected perfect partitions. If  $i * a \leq j * b$  and  $g * b \leq f * a$ , the diagonal elements are formed by H-boxes, and the main elements formed of V-boxes. If  $j * b \leq i * a$  and  $f * a \leq g * b$ , the diagonal elements are formed by V-boxes, and the main elements formed of H-boxes. Otherwise, there can be only four blocks, two with H-boxes, and two with V-boxes. After defining the composition of the diagonal elements, the heuristic determines if it is feasible to place main elements with the necessary dimensions to generate the hollow block.

If there are  $d$  diagonal elements in a placement, then there are  $d * (d - 1)$  main elements in the placement. Ordering the diagonal element from left to right, with 1 being the leftmost block and  $d$  the rightmost, each diagonal block  $i$  has  $d - i$  main elements to the right, and  $d - i$  above, or  $\sum_{i=1}^d 2 * (d - i) = d * (d - 1)$  main elements. If the diagonal element is formed by V-boxes, as in Fig. 1, it is feasible to create the main elements if  $i \equiv 0 \pmod{d - 1}$  and  $g \equiv 0 \pmod{d - 1}$ . To find  $d$ , the HB heuristic tries integers from two up to  $GCD(i, g) + 1$ , the greatest common divisor of  $i$  and  $g$  plus one. If the diagonal element is formed by H-boxes, the same considerations are valid for

$j$  and  $f$ . In the example in Fig. 1,  $i = 8$ ,  $g = 6$ , and  $d = 3$ .

The pseudocode for the HB heuristic is

```

Best_Solution ← 0
For each  $(i, j) \in P(X, a, b)$ ,  $i > 0$ ,  $j > 0$ 
  For each  $(f, g) \in P(Y, a, b)$ ,  $f > 0$ ,  $g > 0$ 
    If  $i * a \leq j * b$  and  $g * b \leq f * a$ ,
      diagonal block is H-block
      For  $d$  from 2 to  $GCD(j, f) + 1$ 
         $S \leftarrow d * (i * g + j * f / (d - 1))$ 
        If  $S > Best\_Solution$  then update
           $Best\_Solution$ 
      If  $j * b \leq i * a$  and  $f * a \leq g * b$ ,
        diagonal block is V-block
        For  $d$  from 2 to  $GCD(i, g) + 1$ 
           $S \leftarrow d * (f * j + i * g / (d - 1))$ 
          If  $S > Best\_Solution$  then update
             $Best\_Solution$ 
      Otherwise, the solution has only four
        blocks

```

### 4. Easily computed bounds

An easily computed and intuitive bound is the *Area Ratio (AR) Bound*,  $\lfloor (X * Y) / (a * b) \rfloor$ . Smith and De Cani (1980) and Dowsland (1985) report the AR bound is equal to the optimal solution in about 15% of randomly generated test instances.

Another intuitive bound is  $\lfloor X/b \rfloor * \lfloor Y/b \rfloor$ , the maximum number of boxes in a (horizontal) row multiplied by the maximum number of boxes in a (vertical) column. If  $b$  is relatively close to  $a$ , then this *Maximum Product (MP) Bound* can outperform the AR bound. For example, the instance (23, 23, 5, 4) has an AR bound  $\lfloor (23 * 23) / (5 * 4) \rfloor = 26$  and a MP bound of  $\lfloor 23/4 \rfloor * \lfloor 23/4 \rfloor = 25$ .

Selecting an instance in the equivalence class that minimizes the AR bound may provide a tighter bound. Proposed by Dowsland (1984), this produces exact bounds in at least 92% of her 5000 randomly generated test cases. We call this bound the *Minimum Area Ratio (MAR) Bound*.

Barnes (1979) proposes a bound based on boxes being represented by patterns of  $a \times 1$  or  $b \times 1$  boxes. Because the solution to problems with unit width is easily obtained (Barnett and Kynch, 1967), Barnes computes the wasted area obtained when placing only  $a \times 1$  or only  $b \times 1$  boxes to produce an upper bound on the number of boxes placed.

Letchford and Amaral (2001) presents a more complete discussion on bounds for PLP, including dominance results and additional bounds such as one based on linear programming found in Isermann (1987).

## 5. Performance of selected heuristics and easily computed bounds

Martins and Dell (2007) show the set of all PLP instances with an area ratio (pallet area divided by box area) less than 101 boxes can be represented by 3,080,730 equivalent classes. For each equivalence class, we compute the MP, MAR, and Barnes bound. We then sequentially apply the one-block, two-block, HB, and five-block heuristics; stopping if a placement is found that equals the best bound. Table 1 shows the results. The first column is the maximum number of boxes that can be placed in a class instance as given by the area ratio. The second column lists the total number of classes. The next four columns show the number, and cumulative percentage, of classes with an optimal placement computed by a heuristic and verified with the MP, MAR, or Barnes bound. The last column shows the number of classes with the best upper bound larger than the best placement generated. For example, the first row shows that among the 662 distinct equivalence classes with up to 10 boxes, the one-block heuristic optimally solved instances in 469 classes (70.8%), the two-block solved an additional 166 instances, totaling 635 classes (95.9%) solved with these two heuristics. The HB and five-block heuristics solved instances in the 27 remaining classes.

Considering instances with an area ratio less than 101 boxes, the one- and two-block heuristics combined provide optimal solutions for 86.2% of the equivalence classes. Adding the results of the other two heuristics, HB and five-block, these heuristics solve 95.0% of the instances. These percentages do not include the cases where a better bound could prove optimality.

## 6. Bounds based on the existence of single perfect partitions

Bounds in this section are applicable when, given an instance  $(X, Y, a, b)$ , there is only one perfect partition in the width (or length) of the pallet, and placing  $M \geq N(X, Y, a, b)$  boxes produces wasted area,  $W(M, X, Y, a, b)$ , smaller than the corresponding dimension  $Y$  (or  $X$ ) of the pallet.

The first case considered is when this single perfect partition consists of only V-boxes or H-boxes; we call this partition a *Homogeneous Perfect Partition*. Given a Homogeneous Perfect Partition, we can derive a bound using a smaller related instance.

We prove the case for the width, with H-boxes.

**Theorem 1.** *Let  $(X, Y, a, b)$  be an instance of PLP, with the only perfect partition in  $Y$  is given by  $B_y * b = Y$ ,  $N(X, Y, a, b) \leq M \leq \lfloor (X * Y) / (a * b) \rfloor$ , and  $W(M, X, Y, a, b) < X + a$ . Then  $N(X + a, Y, a, b) \leq B_y + M$ .*

**Proof.** When comparing pallets for instances  $(X + a, Y, a, b)$  and  $(X, Y, a, b)$ , the larger has an added area of  $a * Y$  or  $a * b * B_y$  (because  $B_y * b = Y$ ). This added area permits the addition of at least  $B_y$  boxes. We show that no more than  $B_y$  can be added.

The pallet for instance  $(X + a, Y, a, b)$  can be partitioned into  $X + a$  columns of unit length, each of which is composed of  $Y$  unit squares, as illustrated by the dashed lines in Fig. 2. In a normal placement, every unit square on the pallet is either completely covered or uncovered by a box. We first observe that for an optimal solution  $M = N(X, Y, a, b)$  there must be at least one unit column with zero waste because  $W(M, X, Y, a, b) < X + a$  and therefore  $W(M', X + a, Y, a, b) < X + a$  for  $M' \geq M + B_y$ . Any such unit column with zero waste must be covered with H-boxes because this corresponds to the only perfect  $Y$ -partition.

If there is a group of H-boxes completely aligned in an optimal solution, as in Fig. 2a, it is easy to see that we can detach the block formed by this group,

Table 1  
Absolute (and cumulative percentage) performance of simple heuristics

Number of boxes	Number of classes	One-block	Two-block	HB	Five-block	Unsolved classes
10	662	469 (70.8)	166 (95.9)	19 (98.8)	8 (100.0)	0 (0.0)
20	7309	4646 (63.6)	2156 (93.1)	236 (96.3)	203 (99.1)	68 (0.9)
50	216,095	128,204 (59.3)	63,322 (88.6)	4455 (90.7)	13,291 (96.8)	6823 (3.2)
100	3,080,730	1,812,852 (58.9)	840,019 (86.2)	45,405 (87.7)	225,929 (95.0)	156,527 (5.0)

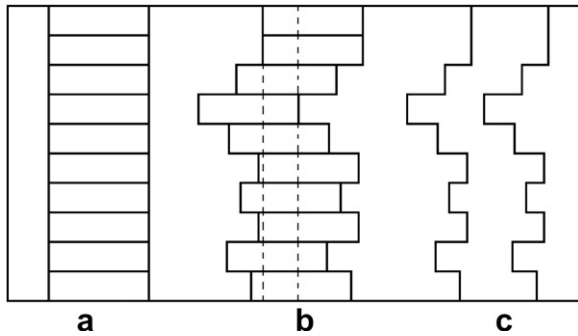


Fig. 2. Examples of groups of H-boxes covering a perfect partition used in the proof of Theorem 1. Here  $B_y = 10$  and the dashed lines show a column with width 1.

which has  $B_y$  boxes, and reconnect whatever is on either side of the block. The reduced instance has solution  $N(X, Y, a, b)$ , the combined solution is  $B_y + N(X, Y, a, b)$ .

If the group of H-boxes is present in an optimum solution in a more arbitrary shape, as in Fig. 2b, then we can transform this solution to another, with the boxes forming a homogeneous block. Given an arbitrary group of  $B_y$  H-boxes, such that the horizontal coordinates of the all boxes in this group overlap in at least one unit of length, we perform the transformation by initially removing these  $B_y$  H-boxes from the pallet. This operation partitions the boxes in the pallet in two sets: those to the left of the removed block, the *left* set, and those to the right, the *right* set. In the next step of the transformation, all the boxes of the right set are pushed to the left, until they touch a box from the other set. Because of the existence of a homogeneous perfect partition, H-boxes from the right set can be pushed by a distance of exactly  $a$  units to the left forming together with the left set a reduced pallet of length  $X - a$ . This creates an empty region at the right edge of the pallet with length of  $a$  units, where the removed group of H-boxes can be placed, after being aligned. Fig. 2c shows the puzzle-like result of removing the group of H-boxes. The reduced instance also has solution  $N(X, Y, a, b)$ , the combined solution is  $B_y + N(X, Y, a, b)$ .

If we only have an upper bound  $M \geq N(X, Y, a, b)$  with  $W(M, X, Y, a, b) < X + a$ . We know if  $M' = N(X + a, Y, a, b)$  is the optimal number of boxes then  $M' = B_y + N(X, Y, a, b) \leq B_y + M$ . Therefore,  $N(X + a, Y, a, b) \leq B_y + M$ .  $\square$

**Corollary 1.** Let  $(X, Y, a, b)$  be an instance of PLP, with  $M \geq N(X, Y, a, b)$  and  $W(M, X, Y, a, b) < X + j * a$ . If the only perfect partition in  $Y$  is given by  $B_y * b = Y$ , then  $N(X + j * a, Y, a, b) \leq j * B_y + M$ .

This result follows directly if we apply recursion to the previous result.

The cases of homogeneous perfect  $Y$ -partitions in  $a$ , and both cases of  $X$ -partitions, follow the same line of reasoning. The sizes of pallets to which this bound applies are limited by  $Y \leq X < a * b$ , because for larger sizes if there is a homogeneous perfect partition given by  $m * b$ , then there is another perfect partition,  $b * a + (m - a) * b$ .

The procedure to rearrange boxes adopted in the proof of Theorem 1 only applies to homogeneous partitions that are perfect. When considering other homogeneous efficient partitions, there is at least one unit strip that is not covered with a box. In this case, the considerations associated with Fig. 2 do not apply.

We refer to this bounding procedure as the *Single Homogeneous Perfect Partition (SHPP) Bound*. Although the requirements to apply this bound are restrictive, Table 2 shows that more than 50% of the instances not bounded by other elementary procedures (as reported in Table 1) are bounded by this new simple bound. The first three columns are from Table 1. The next four columns contain the number of classes with an optimal placement found by the heuristic and verified by the SHPP bound. The last column contains the number of remaining classes without a proven optimal solution.

A different bound based on single perfect partitions can be computed, when  $W(M, X, Y, a, b) \leq b$  or  $W(M, X, Y, a, b) \leq a$ , with  $M \geq N(X, Y, a, b)$ , using some results from Nelissen (1995). Nelissen defines  $lx_{ij}$  as a lower bound on the number of unit

Table 2  
Number of instances with an open result in Table 1 bounded by the SHPP Bound

Number of boxes	Number of classes	Unsolved from Table 1	One-block	Two-block	Hollow-block	Five-block	Unsolved classes
20	7309	68	20	29	0	2	17
50	216,095	6823	1874	1823	0	371	2755
100	3,080,730	156,527	37,350	35,808	0	7726	75,643

rows in any optimal solution where boxes in each such unit row correspond to  $X$ -partition  $(i, j)$ , and  $ly_{f,g}$  as the lower bound on the number of unit columns in any optimal solution where boxes in each such unit column correspond to  $Y$ -partition  $(f, g)$ . For any optimal solution, he shows if

- $X - ly_{f,g} < a$ , then the number of H-boxes is a multiple of  $g$ ,
- $X - ly_{f,g} < b$ , then the number of V-boxes is a multiple of  $f$ ,
- $Y - lx_{i,j} < a$ , then the number of V-boxes is a multiple of  $j$ , and if
- $Y - lx_{i,j} < b$ , then the number of H-boxes is a multiple of  $i$ .

**Theorem 2.** *Let  $(X, Y, a, b)$  be an instance of PLP, with  $N(X, Y, a, b) \leq M \leq \lfloor (X * Y) / (a * b) \rfloor$ ,  $W(M, X, Y, a, b) < b$ , and assume that the only perfect  $Y$ -partition is  $n * a + m * b = Y$ . Then the number of H-boxes in the solution is a multiple of  $m$ , and the number of V-boxes is a multiple of  $n$ .*

**Proof.** Because  $(n, m)$  is the only perfect  $Y$ -partition and  $W(M, X, Y, a, b) < b$ , there are more than  $X - b$  unit columns where boxes in these unit columns correspond to  $Y$ -partition  $(n, m)$ . Otherwise, more than  $b$  unit columns, each with at least one unit of waste, would be present, and the total waste would be larger than  $W(M, X, Y, a, b)$ . Then  $ly_{n,m} > X - b$ ,  $X - ly_{n,m} < b$ , the number of H-boxes is a multiple of  $m$ , and the number of V-boxes is a multiple of  $n$ , as shown by Nelissen.  $\square$

The same results can be shown for partitions of  $X$ .

One example of application of this bound is the class of instance  $(14, 13, 4, 3)$ . The area ratio bound is 15, with 2 units of waste. Because  $W(15, 14, 13, 4, 3) = 2 < b = 3$ , and there is only one perfect  $X$ -partition,  $(2, 2)$ , the number of H-boxes, and V-boxes, is even. The bound is reduced to 14, an even number, and the solution obtained by the two-block heuristic is now known to be optimal.

We refer to the bound generated by this procedure as the *Single Perfect Partition (SPP) Bound*. Table 3 presents updated results using the SPP bound. The first three columns are from Table 2. The next four columns contain the number of classes with an optimal placement found by the heuristic and verified by the SPP bound. The last column contains the number of remaining classes without a proven optimal solution.

### 7. Bounds based on relaxed classes

We extend the use of equivalence classes by using relaxed and restricted classes [e.g., Letchford and Amaral (2001)]. Let  $(X, Y, a, b)$  and  $(Z, W, c, d)$  be instances of PLP, with the set of feasible partitions given by  $F(X, a, b)$ ,  $F(Y, a, b)$ ,  $F(Z, c, d)$ , and  $F(W, c, d)$ . If  $F(X, a, b) \subset F(Z, c, d)$  and  $F(Y, a, b) \subset F(W, c, d)$ , we define  $(Z, W, c, d)$  to be a *relaxed class* of  $(X, Y, a, b)$ , and represent it  $(Z, W, c, d) \succ (X, Y, a, b)$ , and define  $(X, Y, a, b)$  to be a *restricted class* of  $(Z, W, c, d)$ ,  $(X, Y, a, b) \prec (Z, W, c, d)$ .

If  $(X, Y, a, b) \prec (Z, W, c, d)$ , then  $N(X, Y, a, b) \leq N(Z, W, c, d)$  and we refer to a bound determined from a relaxed instance as a *Relaxed Class (RC) Bound*. The time required to find relaxed classes is proportional to the number of efficient partitions and to the time required to find the MSI of a class (Martins and Dell, 2007). The procedure increases each efficient partition by one unit, of  $a$  or  $b$ , depending on the case, and computes the MSI of the resulting relaxed class, if feasible.

An example of where the RC bound applies is instance  $(26, 19, 7, 3)$ . The best bound for this instance used in Table 3 is 23. However, this can be reduced to 22 because  $(26, 19, 7, 3) \prec (18, 13, 5, 2)$  and  $N(18, 13, 5, 2) \leq 22$  using Barnes bound. Table 4 presents updated results using the RC bound. The first three columns are from Table 3. The next four columns contain the number of classes with an optimal placement found by the heuristic and verified by the RC bound. The last column contains the number of remaining classes without a proven optimal solution.

Table 3  
Number of instances with an open result in Table 2 bounded by the SPP bound

Number of boxes	Total number of classes	Unsolved from Table 2	One-block	Two-block	Hollow-block	Five-block	Unsolved classes
20	7309	17	0	1	0	2	14
50	216,095	2755	0	20	1	28	2706
100	3,080,730	75,739	0	128	19	234	75,358

Table 4  
Number of instances with an open result in Table 3 bounded by the RC bound

Number of boxes	Total number of classes	Unsolved from Table 3	One-block	Two-block	Hollow-block	Five-block	Unsolved classes
20	7309	14	0	1	0	0	13
50	216,095	2706	115	589	2	218	1782
100	3,080,730	75,358	4298	16,260	61	8107	46,632

## 8. Bound based on similarity of classes

A bound from a restricted class may also be helpful in bounding a relaxed class. Consider the instance (37, 30, 8, 3). The best bound for this instance in Table 4 is 46 and the best bound for its restricted instance (24, 19 5 2),  $(24, 19, 5, 2) \prec (37, 30, 8, 3)$ , is 45 by the AR bound. The perfect  $Y$ -partition (0, 10) is feasible only in the relaxed instance; otherwise they share the same set of feasible partitions. Restricted to using the (0, 10) partition,  $N(37, 30, 8, 3) = \max\{N(37 - 8 * i, 30, 8, 3) + 10 * i, i = 1, 2, 3, 4\} = 45$  (using the result of Theorem 1). Thus  $N(37, 30, 8, 3) \leq 45$  because when the  $Y$ -partition (0, 10) is used, the best result is 45 and if it is not used, the best result is also 45.

We apply this when having two classes that differ only in a homogeneous perfect partition and refer to this bound as the *Combined Perfect Partition and Restricted Class (CPPRC) Bound*. Table 5 updates results using this new bound. The first three columns are from Table 4. The next four columns contain the number of classes with an optimal placement found by the given heuristic and verified by the CPPRC bound. The last column contains the number of remaining classes without a proven optimal solution.

## 9. The G5-heuristic

The G5-heuristic, like the M&M Heuristic, looks for 1st-order patterns, but applies the HB heuristic to identify partial solutions of a hollow block pattern. The hollow block pattern, as shown in Fig. 1, is a G4-pattern, and also a 1st-order pattern.

We implement the G5-heuristic with only one level of recursion.

There are four main loops in the algorithm, sequentially assigning the dimensions of the four blocks in corners of the placement. The fifth, or central, block has its dimensions defined by the other blocks as in the five-block heuristic. Each of these five blocks has a placement computed using the hollow block, one-, two-, and five-block heuristics, or the G5-heuristic, but with no additional recursive calls.

The wasted area within each block is also computed. At each step of the algorithm, after defining the dimensions of a block, the cumulative sum of wasted areas is computed, preventing the exploration of patterns producing too much waste.

In order to avoid looking at symmetric patterns, the block with the most boxes placed, with the exception of the central block, is defined to be the first block at the lower left corner. Whenever a pattern is produced with another block having a larger number of boxes, the execution of the algorithm moves to the next pattern. This differs from the G4 and M&M heuristics.

As in the G4-heuristic, we keep track of all partial patterns produced. Therefore, the solution to the placement of a block with given dimensions is stored when it is first computed, and is used again whenever a block with the same dimensions is encountered later in the procedure.

Morabito and Morales (1998) present their heuristic run times on a Pentium 100 MHz personal computer. Scheithauer has an implementation of the G4-heuristic available for download from the Internet (CADAP, 2002). We run both G4 and G5

Table 5  
Number of instances with an open result in Table 4 bounded by the CPPRC bound

Number of boxes	Total number of classes	Unsolved from Table 4	One-block	Two-block	Hollow-block	Five-block	Unsolved classes
20	7309	13	0	0	0	1	12
50	216,095	1782	0	24	3	72	1683
100	3,080,730	46,632	0	289	21	770	45,552

heuristics on the same set of problems using a Pentium 133 MHz personal computer, so the run times, presented in Table 6, can be compared with those reported by Morabito and Morales. As described by Morabito and Morales, problems D1 and D2 are from Dowsland (1984), N1 to N5 from Nelissen (1993), and ST1 to ST5 from Scheithauer and Terno (1996).

Considering the instances presented in Table 6, the G5-heuristic often performed slower than the G4-heuristic, and faster than the M&M heuristic although the results from the M&M were recorded on a slower computer. When comparing only the G5-heuristic and the G4-heuristic, on a larger set of instances, the G5-heuristic was faster for instances where the bounds were tighter, taking on average 43% of the time required by the G4-heuristic. But when the best solution provided by the heuristics differed from the best bound, the G5-heuristic was slower, taking on average twice the time required by the G4-heuristic.

Scheithauer (2000) provides a list with 206 instances, from a larger set called *Cover II*, by Nelissen (1993), containing instances satisfying constraints  $1 \leq X/Y \leq 2$ ,  $1 \leq a/b \leq 4$ , and  $51 \leq (X * Y)/(a * b) < 101$ . These 206 instances are not solved to optimality by Nelissen’s heuristic. Scheithauer (2000) reports that the G4-heuristic is able to solve 167 of these instances. Lins et al. (2003) report solving all instances from the Cover II set; taking a few hundred seconds for some difficult instances. The G5-heuristic solves the same 167 instances solved by the G4-heuristic, and solves two additional instances, (121, 120, 16, 9), and (107, 65, 10, 7).

Table 6

Run times for a Pentium 100 MHz computer for M&M and a Pentium 133 MHz personal computer for G4 and G5, when solving a selected set of instances

ID	Instance	MSI	N	Run time (second)		
				M&M	G4	G5
D1	(22, 16, 5, 3)	(22, 16, 5, 3)	23	0.10	0.00	0.00
D2	(86, 82, 15, 11)	(23, 22, 4, 3)	42	0.50	0.05	0.06
N1	(43, 26, 7, 3)	(43, 26, 7, 3)	52 <sup>a</sup>	1.60	0.06	0.50
N2	(87, 47, 7, 6)	(87, 47, 7, 6)	97	46.30	1.21	0.60
N3	(153, 100, 24, 7)	(109, 71, 17, 5)	90	0.10	1.21	0.01
N4	(42, 39, 9, 4)	(42, 39, 9, 4)	45	2.00	0.11	0.25
N5	(124, 81, 21, 10)	(64, 41, 11, 5)	47	5.50	0.16	0.11
ST1	(40, 25, 7, 3)	(40, 25, 7, 3)	47	2.10	0.11	0.19
ST2	(52, 33, 9, 4)	(52, 33, 9, 4)	47	3.10	0.11	0.19
ST3	(57, 44, 12, 5)	(57, 44, 12, 5)	41	0.90	0.11	0.13
ST4	(56, 52, 12, 5)	(56, 52, 12, 5)	48	2.20	0.11	0.21
ST5	(300, 200, 21, 19)	(127, 85, 9, 8)	149	0.10	9.23	0.01

<sup>a</sup> Indicates that the solution is not optimal.

Table 7

Number of instances from Table 5 without a previously proven optimal solution, solved using the G5-heuristic

Number of boxes	Total number of classes	Unsolved from Table 5	Solved with G5-Heuristic	Unsolved classes
20	7309	12	10	2
50	216,095	1683	1449	234
100	3,080,730	45,552	38,546	7006

These two instances have solutions with 1st-order patterns that are not G4-patterns.

Table 7 presents the number of instances not solved with the simple heuristics mentioned previously with a proven optimal solution obtained with the G5-heuristic. The first three columns are from Table 5. The next column contains the number of classes with an optimal placement found by the G5-heuristic. The last column contains the number of remaining classes without a proven optimal solution.

The solution generated by the G5-heuristic differs from the best known bound by at most one box. As verified by our HVZ exact algorithm (Section 11), the G5-heuristic generates optimal solutions to all instances of PLP with an area ratio less than 51 boxes, and all but 59 of the instances with an area ratio less than 101 boxes.

### 10. Higher order non-guillotine heuristics

The *Higher-Order Non-Guillotine (HONG)* heuristics look for arrangement similar to the optimal

eight-block arrangement found by Morabito and Morales (1998) for instance (43, 26, 7, 3). The HONG heuristics divide the pallet into at most eight blocks, distributed as shown in Fig. 3. As can be seen, none is 1st-order. The pattern in Fig. 3a is a vertical pattern, because it is possible to draw a vertical line crossing four blocks. A horizontal pattern, where a horizontal line crosses four blocks, is shown in (b). The last pattern, (c), is named a central pattern.

HONG is an extension of the G5-heuristic, with up to five additional loops, for a total of nine nested loops. Each of these eight blocks can be placed using the non-recursive G5-heuristic. We implement three versions of the heuristic, one for each pattern. Table 8 lists some instances of the COVER II set reported unsolved by Scheithauer (2000), solved using the HONG heuristics. The run times shown correspond to the time required by a Pentium III 600 MHz computer, using the version of the heuristic that obtains the optimal solution for each instance.

In addition to instances in Table 8, the HONG heuristic is able to solve a few other instances not included in the COVER II set. Table 9 presents the results of the application of the HONG heuristic to the instances with open results from Table 7. Only 6952 classes remain unsolved of which 234 have 50 boxes or less.

### 11. An exact algorithm for PLP – the HVZ algorithm

Dowland (1987b) proposes an exact algorithm for PLP. While the Guillotine PLP is solvable in  $O(\log_2 X * \log_2^2 a)$  time (Tarnowski et al., 1994), no similar result has been found for the non-guillotine case, with some authors questioning even if PLP is in NP [e.g., Nelissen (1993), Young-Gun and Maing-Kyu (2001), and Martins (2003)].

We use a coordinate system with origin at the lower left corner of the pallet with the left lower corner of box  $i$ , when placed in the pallet, represented by  $(x_i, y_i)$ . In the coding, an H-box is represented by the letter H, and a V-box by the letter V. Each box is placed in the feasible position that yields the minimum sum of coordinates, i.e.,  $x_i + y_i$ . In case of ties, the position minimizing  $y_i$  is selected. A position is feasible if the box does not overlap with another box placed previously, and is completely placed within the pallet.

Fig. 4 presents a feasible placement for instance (27, 18, 7, 4), with 16 boxes. The corresponding string is HVHVVVHVVVVVVHVH.

There are feasible placements that cannot be represented by this coding scheme. A simple example is instance (7, 7, 5, 2), and the placement depicted in Fig. 5. The corresponding string would be HVVH, but when trying to place the third box, the proce-

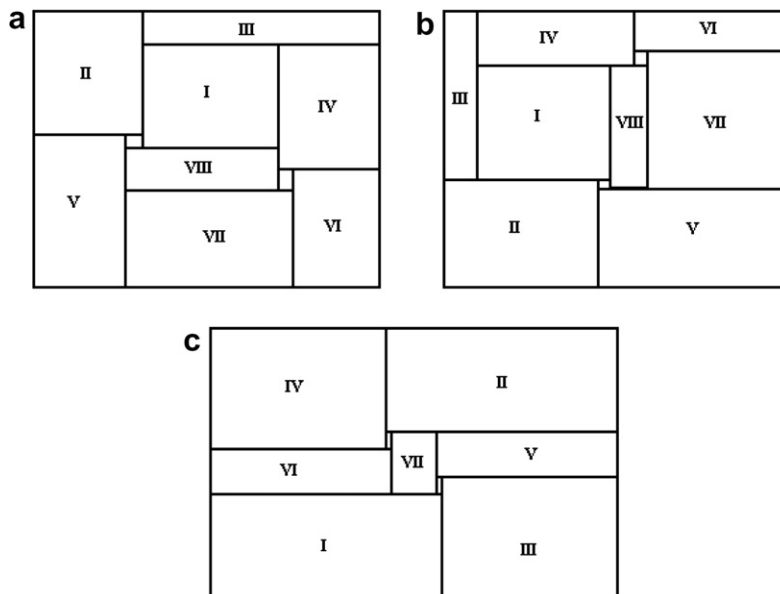


Fig. 3. Non-1st-order patterns explored by the HONG heuristic. The pattern in (a) is a vertical pattern; (b) shows a horizontal pattern; and (c) is a central pattern.

Table 8  
Selected instances from the COVER II set solved to optimality by the HONG heuristics

Instance	Number of boxes	Run time (second)
(43, 26, 7, 3)	53	0.50
(49, 28, 8, 3)	57	0.99
(61, 35, 10, 3)	71	2.63
(61, 38, 10, 3)	77	3.66
(67, 37, 11, 3)	75	3.36
(67, 40, 11, 3)	81	7.51
(141, 119, 21, 8)	99	88.17
(93, 46, 13, 4)	82	15.26
(63, 44, 8, 5)	69	3.80
(57, 34, 7, 4)	69	2.64
(106, 59, 13, 5)	96	17.84
(141, 71, 13, 8)	96	18.13
(74, 73, 13, 5)	82	30.45
(74, 49, 11, 4)	82	0.28
(127, 121, 23, 7)	95	1.20
(106, 59, 13, 5)	96	8.68
(76, 74, 13, 5)	86	345.54
(106, 100, 16, 7)	94	11.26
(83, 82, 11, 7)	88	3.48
(104, 69, 12, 7)	85	2.10
(103, 86, 11, 8)	100	15.05
(104, 71, 11, 7)	95	20.47
(75, 51, 8, 5)	95	6.91
(108, 71, 11, 7)	99	10.61
(78, 51, 8, 5)	99	4.54
(61, 38, 6, 5)	77	0.86
(108, 65, 10, 7)	100	2.44
(164, 83, 14, 11)	88	3.38
(105, 53, 9, 7)	88	2.83
(57, 34, 7, 4)	69	1.70
(122, 86, 16, 7)	93	1.97

Table 9  
Number of instances, from Table 7, without a proven optimal solution, solved using the HONG heuristics

Number of boxes	Total number of classes	Unsolved classes from Table 7	Solved by HONG Heuristic	Unsolved classes
20	7309	2	0	2
50	216,095	234	0	234
100	3,080,730	7006	54	6952

sure would place it at position (2, 2), not at position (5, 0).

We include Z, for zero, representing the position of the left lower corner of a wasted rectangular area. In the example above, the placement is now coded by HVZVH. The Z label occupies the position (2, 2). The next available position is (5, 0), where the next V-box is placed. The only position left to place an H-box is (2, 5).

The wasted rectangle has length and width at least one unit long, but may have larger dimensions,

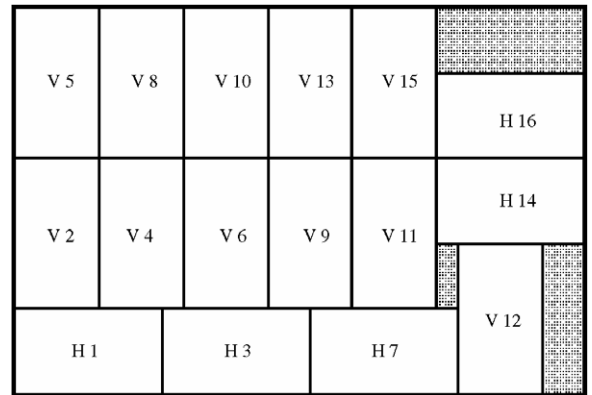


Fig. 4. Feasible placement for instance (27, 18, 7, 4). The placement can be represented by the binary string HVHVVVVHV-VVVVHVH, with H indicating an H-box and V a V-box. The number shown in each box corresponds to the placement order.

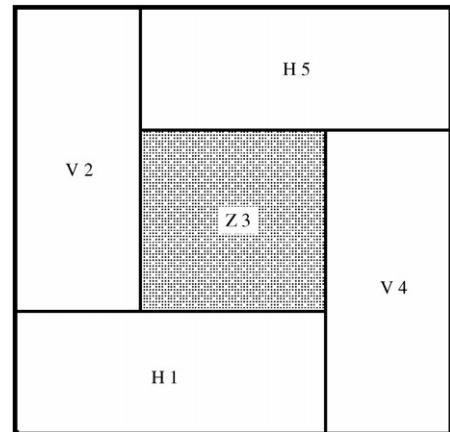


Fig. 5. Feasible placement for instance (7, 7, 5, 2).

depending on the use of normal placement patterns. In this case, the wasted area covers the region up to the next positions that could be used to place a box in a normal placement pattern, i.e., positions with the coordinates corresponding to integral combinations of *a* and *b*. In the example, the wasted area has dimensions 2 × 2, because positions (3, 2), (2, 3) and (3, 3) cannot be obtained as combinations of 5 and 2.

As the amount of wasted area is registered whenever a Z label is included in the string, we can use this coding in a depth-first search, fathoming the branches of the search tree presenting too much wasted area.

Each node of the search tree corresponds to visiting the point with minimum sum of coordinate values. The HVZ algorithm selects between placing an H-box, a V-box, and marking the region as wasted.

If a box is placed, then all coordinate points covered by the box are labeled as used. If marked as wasted, only one coordinate point is labeled, and the area of the wasted region is added to the total wasted area,  $TW$ . The next node examined corresponds to the next unlabeled coordinate point. If, at any given step of the algorithm, the maximum allowed amount of wasted area,  $MW$ , is surpassed ( $TW > MW$ ), then the algorithm backtracks. The algorithm terminates if a feasible placement of  $N$  boxes is obtained, or if all possible labels are explored.

The set of coordinate points is defined in the initialization of the algorithm, and corresponds to the cartesian product of the sets of feasible partitions of the box length and width. In our implementation of the HVZ algorithm, we use two arrays to represent these coordinate points:

- A two-dimensional array, with the first dimension corresponding to feasible partitions of the length, and the second to feasible partitions of the width. This array is used when labeling the coordinate point covered by a box; and
- A one-dimensional array, with pointers to the coordinate points, ordered by ascending sum of coordinate values, and then by ascending width. This array is used to identify the next unmarked point.

Some experimentation with HVZ reveals some strategies performed at each node that can yield shorter run times:

- If the number of boxes loaded is less than  $\lfloor N/2 \rfloor$ , and the total wasted area is already larger than

half of the maximum allowed,  $TW > MW/2$ , then the algorithm backtracks. This is justified by the symmetry of the placements. In this case, the complement of the packing pattern contains at least  $N/2$  boxes and at most  $MW/2$  wasted area, and can be obtained at a later step of the algorithm.

- After placing a box, verify that there is enough space between the box and the borders of the pallet to place another box. If not, mark the region as wasted.
- If an optimal placement is formed with smaller blocks placed together, and if each of these blocks can be placed in different ways, then we have several similar, or equivalent, solutions. In order to avoid searching for solutions on patterns similar to others already identified as not optimal, we verify if the box being placed completes a block. If this is the case, we check to see if a block with the same dimensions has already been investigated, if the placement of this block is the same, and, if so, we backtrack.

## 12. HVZ results

We use the HVZ algorithm to solve the MSI from the remaining 6952 classes. Most of these instances solve in a few minutes but 226 require more than an hour and 23 require more than 24 hours, using a Pentium III 600 MHz computer. The HVZ confirmed the heuristics found an optimal solution to all but five instances, (74, 46, 7, 5), (86, 52, 9, 5), (95, 92, 11, 8), (172, 66, 19, 7), and (178, 60, 16, 7).

Table 10  
Run times, on a Pentium III 600 MHz, obtained with the HVZ algorithm and the G5-heuristic for instances from Table 6

ID	Instance	MSI	N	Run time (second)	
				G5-Heuristic	Exact algorithm
D1	(22, 16, 5, 3)	(22, 16, 5, 3)	23	0.00	0.00
D2	(86, 82, 15, 11)	(23, 22, 4, 3)	42	0.06	0.00
N1	(43, 26, 7, 3)	(43, 26, 7, 3)	53	0.05	0.17
N2	(87, 47, 7, 6)	(87, 47, 7, 6)	97	0.33	4,092.61
N3	(153, 100, 24, 7)	(109, 71, 17, 5)	90	0.00	0.05
N4	(42, 39, 9, 4)	(42, 39, 9, 4)	45	0.11	0.55
N5	(124, 81, 21, 10)	(64, 41, 11, 5)	47	0.05	0.93
ST1	(40, 25, 7, 3)	(40, 25, 7, 3)	47	0.00	0.16
ST2	(52, 33, 9, 4)	(52, 33, 9, 4)	47	0.06	0.44
ST3	(57, 44, 12, 5)	(57, 44, 12, 5)	41	0.00	0.06
ST4	(56, 52, 12, 5)	(56, 52, 12, 5)	48	0.05	2.42
ST5	(300, 200, 21, 19)	(127, 85, 9, 8)	149	0.00	2,247.83

Table 10 shows some sample HVZ and G5-heuristic solution times for the same instances listed in Table 6.

### 13. Complexity of the HVZ algorithm

Using a HVZ string to code any placement requires at most  $N + W$  characters, where  $N$  is the number of boxes placed and  $W$  is the wasted area in an optimal placement. Each of the  $N$  characters corresponding to placed boxes can take two different values, with  $2^N$  ways to assign these values. If  $W$  characters are necessary to represent wasted areas, then there are  $\binom{N+W}{W}$  ways of selecting the position of these characters in the string.

It takes  $O((N + W)^2)$  to verify if the pattern coded by a given string is feasible. As the number of boxes placed and amount of wasted area gets larger, the computational effort increases exponentially.

### 14. Conclusions

We have presented both heuristic and exact algorithms for the PLP, and employ these algorithms in concert with a set of bounding procedures, to solve to optimality all instances of PLP with an area ratio less than 101 boxes. Only a subset of these classes has been previously considered and, of those previously considered, an optimal solution had been unknown for several instances.

### References

- Arenales, M., Morabito, R., 1995. An AND/OR-graph approach to the solution of two-dimensional non-guillotine cutting problems. *European Journal of Operational Research* 84, 599–617.
- Barnes, F.W., 1979. Packing the maximum number of  $m \times n$  tiles in a large  $p \times q$  rectangle. *Discrete Mathematics* 26, 93–100.
- Barnett, S., Kynch, G.J., 1967. Exact solution of a simple cutting problem. *Operations Research* 15, 1051–1056.
- Bischoff, E.E., Dowsland, W.B., 1982. An application of the micro to product design and distribution. *Journal of the Operational Research Society* 33, 271–281.
- CADAP, 2002. “CADAP – Cutting and Packing at Dresden University of Technology.” Available from: <http://www.math.tu-dresden.de/~capad/capad.html>.
- Christofides, N., Whitlock, C., 1977. An algorithm for two-dimensional cutting problems. *Operations Research* 25, 30–45.
- Dowsland, K.A., 1984. The three-dimensional pallet chart: An analysis of the factors affecting the set of feasible layouts for a class of two-dimensional packing problems. *Journal of the Operational Research Society* 35, 895–905.
- Dowsland, K.A., 1985. Determining an upper bound for a class of rectangular packing problems. *Computers and Operations Research* 12, 201–205.
- Dowsland, K.A., 1987a. A combined data-base and algorithmic approach to the pallet-loading. *Journal of the Operational Research Society* 38, 341–345.
- Dowsland, K.A., 1987b. An exact algorithm for the pallet loading problem. *European Journal of Operational Research* 31, 78–84.
- Gilmore, P.C., Gomory, R.E., 1965. Multistage cutting stock problems of two and more dimensions. *Operations Research* 13, 94–120.
- Isermann, H., 1987. Ein Planungssystem zur Optimierung der Palettenbeladung mit kongruenten rechteckigen Versandgebänden. *EUR Spectrum* 9, 235–249.
- Letchford, A.N., Amaral, A., 2001. Analysis of upper bounds for the pallet loading problem. *European Journal of Operational Research* 132, 582–593.
- Lins, L., Lins, S., Morabito, R., 2003. An  $L$ -approach for packing  $(l, w)$ -rectangles into rectangular and  $L$ -shaped pieces. *Journal of the Operational Research Society* 54, 777–789.
- Martins, G., 2003. Packing in two and three dimensions, Ph.D. Dissertation, Naval Postgraduate School, Monterey, California, June.
- Martins, G., Dell, R., 2007. The minimum size instance of a pallet loading problem equivalence class. *European Journal of Operational Research* 179, 17–26.
- Morabito, R., Morales, S., 1998. A simple and effective recursive procedure for the manufacturer’s pallet loading problem. *Journal of the Operational Research Society* 49, 819–828.
- Nelissen, J., 1993. New Approaches to the Pallet Loading Problem, Working paper, RWTH Aachen <ftp://ftp.informatik.rwth-aachen.de/pub/reports/pallet.ps.Z>.
- Nelissen, J., 1995. How to use structural constraints to compute an upper bound for the pallet loading problem. *European Journal of Operational Research* 84, 662–680.
- Oliveira, J., Ferreira, J., 1990. An improved version of Wang’s algorithm for two-dimensional cutting problems. *European Journal of Operational Research* 44, 256–266.
- Scheithauer, G., 2000. Personal communication.
- Scheithauer, G., Terno, J., 1996. The G4-Heuristic for the pallet loading problem. *Journal of the Operational Research Society* 47, 511–522.
- Smith, A., De Cani, P., 1980. An algorithm to optimize the layout of boxes in pallets. *Journal of the Operational Research Society* 31, 573–578.
- Tarnowski, A.G., Terno, J., Scheithauer, G., 1994. A polynomial time algorithm for the guillotine pallet loading problem. *Information Systems and Operational Research* 32, 275–287.
- Wang, P.Y., 1983. Two algorithms for constrained two-dimensional cutting stock problems. *Operations Research* 31, 573–586.
- Young-Gun, G., Maing-Kyu, K., 2001. A fast algorithm for two-dimensional pallet loading problems of large size. *European Journal of Operational Research* 134, 193–202.