European Journal of Operational Research 224 (2013) 486–496

Contents lists available at SciVerse ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

A reduction approach for solving the rectangle packing area minimization problem

Andreas Bortfeldt *

University in Hagen, Department of Information Systems, Profilstrasse 8, 58084 Hagen, Germany

ARTICLE INFO

Article history:
Received 29 October 2011
Accepted 12 August 2012
Available online 26 August 2012

Keywords:
Packing
Rectangle packing area minimization problem
Floor planning
Open dimension problem
MCNC
GSRC

ABSTRACT

In the rectangle packing area minimization problem (RPAMP) we are given a set of rectangles with known dimensions. We have to determine an arrangement of all rectangles, without overlapping, inside an enveloping rectangle of minimum area. The paper presents a generic approach for solving the RPAMP that is based on two algorithms, one for the 2D Knapsack Problem (KP), and the other for the 2D Strip Packing Problem (SPP). In this way, solving an instance of the RPAMP is reduced to solving multiple SPP and KP instances. A fast constructive heuristic is used as SPP algorithm while the KP algorithm is instantiated by a tree search method and a genetic algorithm alternatively. All these SPP and KP methods have been published previously. Finally, the best variants of the resulting RPAMP heuristics are combined within one procedure. The guillotine cutting condition is always observed as an additional constraint. The approach was tested on 15 well-known RPAMP instances (above all MCNC and GSRC instances) and new best solutions were obtained for 10 instances. The computational effort remains acceptable. Moreover, 24 new benchmark instances are introduced and promising results are reported.

1. Introduction

The paper deals with the two-dimensional (2D) rectangle packing area minimization problem (RPAMP) that is also called rectangle packing problem. For a set of n rectangles (or items) with length l_i and width w_i $(i = 1, \ldots, n)$ we have to determine a feasible arrangement of all n rectangles inside a larger rectangle (called container) with length l_C, width w_C and minimum area $A = l_C * w_C$.

An arrangement is said to be feasible when no items overlap and all items are placed completely within the container and parallel to the container edges. There are no further restrictions imposed on the item stock, so the stock may be homogeneous (one single type of rectangle), weakly heterogeneous (a few rectangle types, many rectangles per type) or strongly heterogeneous (many rectangle types, a few rectangles per type). Two rectangles belong to the same type if they are congruent.

According to the typology of cutting and packing problems (C&P) by Wäscher et al. (2007) the RPAMP is referred to as open dimension problem (ODP) or more precisely as a 2D rectangular open dimension problem with two variable dimensions.

The RPAMP has to be dealt with in different industries and in many practical C&P scenarios. To date the most important application domain of the RPAMP is the VLSI design (see Section 2). However, the RPAMP also occurs as packing problem in facility layout and in newspaper page layout, e.g. in the arrangement of advertisements. As a cutting problem the RPAMP may, for example, occur in the metal, glass or wood industries if fixed sets of rectangular pieces that are stipulated in customer orders are to be cut from larger plates or boards.

The following constraints play an important role in two-dimensional C&P scenarios (cf., e.g., Lodi et al., 1999) and are combined here with the RPAMP:

(C1) Orientation constraint: For a defined subset of the item stock the permitted orientation variants are restricted. For each rectangle one of the two possible spatial orientations may be prohibited.
(C2) Guillotine cutting constraint: It is required that all items of a packing plan can be reproduced by guillotine cuts, i.e. by edge-to-edge cuts that run parallel to the edges of the container.

Both the orientation and the guillotine cutting constraint often have to be observed in cutting problems, due to the nature of material surfaces (e.g. as a result of rolling) or the cutting technology. The orientation constraint also occurs in packing, for example in the arrangement of advertisements.

Lodi et al. (1999) introduced the following schema of subtypes for the 2D Bin Packing Problem that is latterly also used for the 2D Strip Packing Problem. Following this schema and considering the constraints (C1) and (C2), the following four subtypes (variants) can be distinguished:

– *OF:* orientation of all pieces is fixed (O) and guillotine cutting is not required (F).

* Tel.: +49 2331 987 4433; fax: +49 2331 987 4447.
E-mail address: andreas.bortfeldt@fernuni-hagen.de

0377-2217/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.ejor.2012.08.006

- *RF:* pieces may be rotated by 90° (R) and guillotine cutting is not required (F).
- *OG:* orientation of all pieces is fixed (O) and guillotine cutting is required (G).
- *RG:* pieces may be rotated by 90° (R) and guillotine cutting is required (G).

Note that a feasible solution with regard to subtype OG is also feasible for the other subtypes while a feasible solution with regard to subtype RG and OF, respectively, is also feasible in terms of subtype RF. Since the constraints (C1) and (C2) may also occur in RPAMP applications it is obvious to use the above schema also for this kind of 2D C&P problem.

In this paper we investigate in which way effective heuristics for solving the RPAMP can be designed by means of algorithms for related 2D C&P problems, namely the 2D Strip Packing Problem (SPP) and the 2D Knapsack Problem (KP). In the 2D-SPP and 2D-KP we are also given a set of rectangles with fixed dimensions. In the 2D-SPP we look for a feasible arrangement of all rectangles within a container of fixed width and minimum length. In the 2D-KP we search for a feasible arrangement of a subset of the given rectangles with maximum total area that fits into a container with a given width and length.

A generic schema for solving the RPAMP is presented here, based on an algorithm for the 2D-SPP as well as an algorithm for the 2D-KP. In this way, solving an instance of the RPAMP is reduced to solving a series of SPP instances and a series of KP instances. Hence, the RPAMP schema is called reduction approach. The generic schema is then transferred into a set of completely specified RPAMP solution methods. For this the SPP algorithm and the KP algorithm are instantiated by well-known, earlier published methods. At the end, some of the best RPAMP method variants are combined within one procedure to utilize their complementary strengths. Experimental results for well-known RPAMP instances (MCNC, GSRC, instances by Imahori et al. (2005)) will show that the combined procedure achieves a high solution quality. The guillotine cutting constraint (C2) is observed in any case. Finally, 24 new RPAMP benchmark instances are introduced and results of the combined procedure are reported.

In the following section we give a literature overview concerning the RPAMP. In Section 3 we describe the reduction approach and the resulting combined procedure. Section 4 will report on numerical experiments, while some conclusions are drawn in Section 5.

2. Literature review

The design of VLSI chips can be considered the main application area of the RPAMP and most publications on this matter relate to this domain. An important step in chip design is the placement of a set of rectangular circuits (also called modules) on a rectangular chip. The modules are to be connected by wires according to the chip design. There are different goals that are optimized alternatively or simultaneously. Most important are the minimization of the area of the rectangular chip and the minimization of the total wire length. A better utilization of the ship area helps saving material and costs while reducing the total wire length can increase the operational speed of chips.

Multiple domain-specific constraints occur in ship design. Placement constraints (see Tang and Wong, 2001; Pisinger, 2007) are imposed on the placement of specific modules and they can, e.g., fix the position of a module. For symmetry and regularity constraints the reader is referred to Drakidis et al. (2006) and Chen et al. (2011b).

In the chip design domain, the RPAMP and related problems are mostly called Floor Planning Problem (FPP). Two main variants of the FPP are investigated. In the first FPP variant the rectangular modules have fixed dimensions and in the second variant only the areas of the modules are fixed and the lengths and widths of the modules can vary to some degree. Here we do not consider the FPP variant with variable-sized modules (see, e.g. Valenzuela and Wang, 2002; Adya and Markov, 2003) and we will pay little attention to technical details and special constraints of the FPP with fixed modules.

In Table 1 an extensive sample of papers for solving the RPAMP and FPP with hard modules, respectively, is listed. In column 1 authors and year of the publications are indicated. In column 2 the considered objectives are listed; if both area and wire length appear, then mostly a simultaneous optimization of these goals can be carried out. Column 3 includes the additional constraints (if any). In column 4 the types of approaches are given (abbreviations are explained at the end of Table 1). In column 5 additional characteristics of the approaches are specified; since most of the solution methods work with coded solutions, in many cases the used representation of solutions is shown here. Column 6 indicates whether results of a pure area minimization (i.e. without wire length optimization) were published.

Solution methods for solving the FPP with fixed modules are mostly tested by means of two well-known sets of benchmark instances, namely the MCNC (Microelectronics Center of North Carolina) and GSRC (Gigascale Systems Research Center) instances. All these instances can be downloaded from http://vlsicad.eecs.umich.edu/BK. Here, for each method the solution quality, i.e. the best achieved density, is listed for the two most important MCNC benchmarks, namely Ami33 and Ami49, in columns 7 and 8 if available (see Table 3 for further information). The density is calculated as quotient of the total area of all rectangles and the area of the found enveloping rectangle in percent.

Virtually all contributions listed in Table 1 deal with subtype RF, i.e. the constraints (C1) and (C2) are not taken into account. The floor planning variant with guillotine cutting constraint is called slicing variant and the opposite variant is termed non-slicing variant. The non-slicing variant is preferred in the floor planning literature since the related solution space is larger this allowing for a higher utilization and/or smaller total wire length in general.

The RPAMP is NP-hard (cf. Christofides and Whitlock, 1977). Up to now, exact approaches can cope with instances with 30 pieces at most (cf. Chan and Markov, 2004; Korf et al., 2010). Thus, mainly heuristic and meta-heuristic procedures were suggested that do not ensure optimal solutions. In the majority of papers simulated annealing and genetic algorithms were proposed. Much research has been done to find out effective and efficient representations of solutions that are needed to easily manipulate and vary solutions within meta-heuristic approaches. As the results for the instances Ami33 and Ami49 illustrate, the level of solution quality has been considerable improved in the last 15 years. The most successful RPAMP methods will be included in the numerical comparison in Section 4.

3. Reduction approach and resulting RPAMP solution methods

Below a heuristic for solving the RPAMP, abbreviated as AMRH (as area minimization reduction heuristic), is introduced. AMRH makes use of an algorithm SPA for solving the 2D-SPP and an algorithm KA for the 2D-KP and it is generic since both algorithms can be chosen arbitrarily. Based on AMRH eight RPAMP solution methods are completely specified. Finally, the two most promising RPAMP methods are combined to a single procedure.

Table 1
Solution methods for the RPAMP and FPP with fixed modules (sample).

Authors/source	Objectives	Additional constraints	Approach type	Additional characteristics of approach	Pure area minimization	Density Ami33 (%)	Density Ami49 (%)
Murata et al. (1996)	Area, wire length	–	SA	Sequence pair (SP) represent.	–	–	–
Nakatake et al. (1996)	Area	–	SA	Bounded sliceline grid (BSG) represent.	Yes	–	–
Guo et al. (1999)	Area, wire length	–	C/GH	O-tree represent, deterministic	Yes	92.5	94.3
Chang et al. (2000)	Area, wire length	–	SA	B*-tree represent.	Yes	91.0	96.3
Hong et al. (2000)	Area, wire length	–	SA	Corner block list represent.	Yes	96.3	91.9
Pang et al. (2000)	Area, wire length	–	RS	O-tree represent.	Yes	93.1	93.9
Lin and Chang (2001)	Area, wire length	–	SA	Transitive closure graph (TCG) represent.	Yes	96.3	96.4
Tang and Wong (2001)	Area, wire length	Placement constraints	SA	Sequence pair represent.	Yes	95.9	97.1
Yao et al. (2001)	–	–	–	Comparison of multiple floorplan represent.	–	–	–
Zhou et al. (2001)	Area, wire length	–	SA	Extended corner block list represent.	Yes	97.0	96.6
Lin et al. (2002)	Area	–	SA	generalized Polish expression represent.	Yes	98.0	97.2
Wu et al. (2002)	Area	–	C/GH	Less flexibility first approach, deterministic	Yes	97.0	96.0
Lin and Chang (2002)	Area, wire length	–	SA	Transitive closure graph (TCG) represent.	Yes	97.5	97.4
Zhuang et al. (2002)	Area	–	SA	Q-sequence represent.	Yes	96.8	96.5
Adya and Markov (2003)	Area, wire length	–	SA	Sequence pair represent.	Yes	97.1	95.1
Lin and Chang (2003)	Area, wire length	–	SA	Transitive closure graph-based (TCG) represent.	Yes	96.3	96.4
Young et al. (2003)	Area, wire length	–	SA	Twin binary sequences (TBS) represent.	Yes	96.7	96.1
Chan and Markov (2004)	Area	–	B&B	Exact multi-level B&B, subtypes RF, RG considered	Yes	92.5	92.8
Dong et al. (2004)	Area, wire length	–	QP, C/GH	Less flexibility first principle, deterministic method	No	–	–
Zhou and Wang (2004)	Area	–	SA	Adjacent constraint graph (ACG) represent.	Yes	96.3	96.0
Imahori et al. (2005)	Area	–	LS	Designed for more abstract problem, SP represent.	Yes	–	97.4
Chen et al. (2005)	Area	–	SA	Segment list (SL) represent.	Yes	95.5	95.9
Wu and Chan (2005)	Area	–	C/GH	Improved least flexibility first approach, deterministic	Yes	98.2	97.8
Tang and Sebastian (2005)	Area, wire length	–	GA	O-tree represent.	Yes	94.8	94.5
Drakidis et al. (2006)	Area	Symmetry	GA	Observed, sequence pair represent.	Yes	96.3	94.9
Kimura and Ida (2006)	Area, wire length	–	GA	sequence pair represent.	No	–	–
Sun et al. (2006)	Area	–	PSO	B*-tree represent.	Yes	90.3	86.4
Pisinger (2007)	Area	Placement	SA	Sequence pair represent, semi-normalized placements	Yes	99.0	98.5
Chatterjee and Manikas (2007)	Area	–	GA	Sequence pair represent.	Yes	87.6	90.1
Chen and Huang (2007)	Area	–	IH	Candidate corner occupying action approach, advanced placement evaluation	Yes	98.6	98.3
Fernando and Katkoori (2008)	Area, wire length	–	GA	Sequence pair represent.	No	–	–
Clautiaux et al. (2008)	Area	–	CP	Exact approach	Yes	–	–
Rahim et al. (2008)	Area	–	GA	Binary tree represent.	Yes	96.3	–
Beltran-Cano et al. (2009)	Area	–	GRASP/VNS	Problem reduction to 2D strip packing	Yes	97.0	97.5
Lin et al. (2009)	Area, wire length	Symmetry	SA	B*-tree represent.	Yes	94.0	96.2
Chen et al. (2010)	Area, wire length	–	PSO	Integer coding represent.	Yes	89.6	91.1
Korf et al. (2010)	Area	Orientation	CS	Two exact approaches	Yes	–	–
Janiak et al. (2010)	Area	–	LS	Sequence pair represent.	Yes	90.3	96.1
Maag et al. (2010)	Area	Orientation, only subtype OF	NLO	Temporary relaxation of non-overlapping constraint	Yes	–	–
Li et al. (2010)	Area	–	LS	AP-TCG represent.	Yes	98.1	98.3
Chen et al. (2011a)	Area	–	SA	B*-tree represent.	Yes	–	–
Chen et al. (2011b)	Area, wire length	Regularity	SA	Sequence pair represent.	Yes	95.4	93.8

Key: GA: genetic algorithm; C/GH: constructive/greedy heuristic, SA: simulated annealing, GRASP: greedy randomized adaptive search procedure, VNS: variable neighborhood search, PSO: particle swarm optimization, LS: local search, IH: improvement heuristic, B&B: branch and bound, RS: random-based search, QP: quadratic programming, QS: constraint satisfaction, CP: constraint programming, NLO: nonlinear optimization.

3.1. Generic reduction heuristic AMRH

The generic reduction heuristic AMRH is based on the following ideas:

- To tackle a given RPAMP instance an initial solution is generated by algorithm SPA. Then a series of 2D-KP instances is solved by means of the algorithm KA. The set of rectangles of all 2D-KP instances (denoted by R) is given by the RPAMP instance while the container dimensions are varied such that the container areas form a monotonically decreasing sequence. A 2D-KP instance is said to be solved successfully by algorithm KA if all given rectangles were packed and the solution is then also called successful. According to the specification of the containers each successful solution of a 2D-KP instance represents a new best solution for the RPAMP instance, too.
- An ordered list of potential container widths for 2D-KP instances is provided in advance by a dedicated heuristic. Each of these container widths is generally tried several times with different container lengths. The next container width from the list is only tried if the incumbent best solution cannot be improved any longer using the current width.
- Given the width of a 2D-KP instance, the length is then calculated such that a successful solution of the next 2D-KP instance is a new best RPAMP solution at the same time. At the beginning of the search the density of the incumbent best RPAMP solution is generally relatively low and consequently the container lengths are determined such that the increments of the densities of consecutive 2D-KP instances are relatively large. At the end of the search only small density increments can normally be achieved and the container lengths are chosen accordingly.

The pseudo-code of heuristic AMRH is shown in Algorithm 1. At first an initial solution is generated and the best solution s_{best} is set accordingly. A vector W_C with nw different container widths is then provided by a special heuristic and further variables (irk, new_width, iw) are set to initial values. In the subsequent loop solutions to the given RPAMP instance are calculated using the nw provided width values. In each cycle the following steps are done:

(1) If the flag *new_width* is set then the next value in vector W_C is taken as current width w_C, otherwise the old width value is kept.
(2) The current container length l_C is determined by means of the current width w_C, the incumbent best solution s_{best} and the increment rank ikr. For each of the possible values $1,2,\ldots,max\_irk$ of irk a desired increment $I(irk)$ of the density is stipulated and the values $I(irk)$ form an increasing sequence (max_irk, $I(irk)$, $irk = 1,\ldots,max\_irk$, are constant parameters, see Section 3.2). If $irk \geqslant 1$ the container length is chosen such that for the specified width w_C the density increment $I(irk)$ compared to the best solution so far will be implemented if the next 2D-KP instance (w_C, l_C, R) is solved successfully. For $irk = 0$ the container length l_C is defined such that the smallest possible increment of density will result.
(3) The next 2D-KP instance (w_C, l_C, R) is then solved by algorithm KA. If the instance was solved successfully, the best RPAMP solution is updated. Here the smallest enveloping rectangle for the new best packing plan is recalculated since this can yield to a further improvement. The flag *new_width* is set to false, i.e. the current width is kept for the next cycle. If the increment rank irk is zero it is increased again to 1 in order to allow for a greater reduction of length l_C (a further increasing of irk turned out to be not useful).
(4) If the instance was not solved successfully two cases are distinguished. If the increment rank irk is still greater than zero it is reduced by one and the flag *new_width* is set false. Thus, in the next cycle a 2D-KP instance with same width but slightly enlarged length is to be solved, i.e. the procedure looks for a new best solution with a smaller density increment than before. For $irk = 0$ the flag *new_width* is set true and the width index iw is incremented; hence, a new width is tried in the next iteration.

Algorithm 1. Generic reduction heuristic AMRH

Algorithm AMRH (in: number n and set R of rectangles $(l_i, w_i, i = 1,\ldots,n)$, out: best solution s_{best})
// initialize search
generate initial solution s_0 by algorithm SPA and set $s_{best} := s_0$;
provide appropriate container widths $W_C(iw)$, $iw = 1,\ldots,nw$;
set increment rank $irk := max\_irk$;
set flag *new_width* := true and width index $iw := 1$;
// perform search with provided width values
while $iw \leqslant nw$ **do**
  if *new_width* **then** set current width $w_C := W_C(iw)$; **endif**;
  provide current container length l_C;
  generate solution s_{new} for 2D-KP instance (w_C, l_C, R) by algorithm KA;
  if all rectangles packed **then**
    $s_{best} := s_{new}$; *new_width* := false;
    if $irk = 0$ **then** $irk := 1$; **endif**;
  else // not all rectangles packed
    if $irk > 0$ **then** $irk :=$ irk $- 1$; *new_width* := false;
    else *new_width* := true; $iw := iw + 1$; **endif**;
  endif;
endwhile;

Two parts of the heuristic need further explanations.

3.1.1. Generating the initial solution

At first, an interval $[w_{min}, w_{max}]$ of reasonable width values is determined. w_{min} is set to the maximum of the smaller dimensions of all rectangles. w_{max} is set to $\lfloor area(R)^{1/2} * \alpha \rfloor$ where $area(R)$ stands for the total area of all rectangles and $\lfloor a \rfloor$ is the greatest integer not greater than a. Note that we can assume $w_C \leqslant l_C$ since an orientation constraint (C1) is not present here. For an optimal density of 100% it would be suffice to consider width values not greater than $\lfloor area(R)^{1/2} \rfloor$. As the density of the optimal solution is often lower a relatively large value is chosen for w_{max} by means of a parameter $\alpha > 1$ (see Section 3.2). For each width $w_C \in [w_{min}, w_{max}]$ the corresponding 2D-SPP instance (w_C, R) is solved by means of the algorithm SPA and the best solution is taken as initial RPAMP solution.

3.1.2. Generating width values

The vector W_C of promising width values can be generated in three variants WV1 to WV3. The number nw of width values is limited by the parameter ub_nw in any case and the width values are always taken from the interval $[w_{min}, w_{max}]$ defined above.

In *variant WV1* equidistant width values are selected in $[w_{min}, w_{max}]$ according to

$$dw = \lfloor (w_{max} - w_{min})/ub\_nw \rfloor, \quad nw = ub\_nw,$$
$$WC(iw) = w_{min} + (iw - 1)^{*} dw, \quad iw = 1,\ldots,nw.$$

In *variant WV2* we assume that width values leading to the best 2D-SPP solutions (generated by algorithm SPA) are also appropriate width values regarding the entire search within heuristic AMRH. Therefore, all solutions for the 2D-SPP instances

RPAMP instance:

$n = 6$

$r_1 = (10,2)$, $r_2 = (10,3)$, $r_3 = (10,4)$,

$r_4 = (10,5)$, $r_5 = (10,6)$, $r_6 = (10,7)$

wC	*nc*	*width combinations*
8	*2*	*2+6, 3+5*
9	*3*	*2+7, 3+6, 4+5*

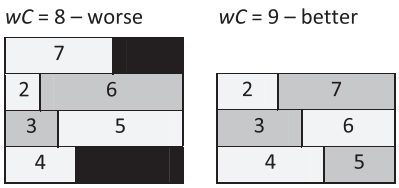

Fig. 1. Width combinations and packing arrangements.

(w_C,R), $w_C \in [w_{min},w_{max}]$ are sorted by their densities in descending order. The first *ub_nw* width values in this sequence form the vector W_C.

In *variant WV3* we believe that those width values will lead to high quality RPAMP solutions that can be combined most frequently by side dimensions of the given rectangles. Promising widths are generated by means of width combinations. A width combination for a width value w_C is defined as a set of rectangle dimensions (l_i or w_i, $i \in \{1,\ldots,n\}$) summing up to w_C. The frequency $nc(w_C)$ of width w_C is given by the number of all width combinations for the value w_C, i.e. the number of all width combinations in which the sum of the item dimensions equals w_C. The number of the item dimensions per combination is given by a parameter *maxitems*.

To implement variant WV3 a dynamic programming procedure of pseudo-polynomial complexity serves to generate basically *all* width combinations for all width values $w_C \in [w_{min},w_{max}]$ (see Section 3.1.1) with exactly *maxitems* item dimensions. The procedure is shown in Appendix A. Then the width values w_C are sorted by their frequencies $nc(w_C)$ in decreasing order and the first *ub_nw* width values are offered in vector W_C. The search can be further diversified by several search phases with different *maxitems* values. This aspect is considered in Section 3.2.

Variant WV3 is illustrated by a small RPAMP instance with $n = 6$ rectangles (see Fig. 1). In the table all width combinations are listed for the values $w_C = 8$ and $w_C = 9$, respectively. The best possible RPAMP solutions for both container widths are shown below. For $w_C = 8$ there are only two width combinations and the pattern is worse (black areas represent waste) and for $w_C = 9$ there are three width combinations and the pattern is better, i.e. the enveloping area is smaller and the pattern does not include any waste (note that the identical vertical dimensions of all items were compressed in the figure).

3.2. Specification of solution methods and their combination

In order to fully specify solution methods based on the generic procedure AMRH at first some packing algorithms for the 2D-SPP and the 2D-KP are selected from the literature. Algorithms for the Strip Packing Problem in two and three dimensions were recently proposed, e.g., by Bortfeldt (2006), Bortfeldt and Mack (2007), Alvarez-Valdes et al. (2008), Belov et al. (2008), Allen et al. (2011) and Leung et al. (2011). Algorithms for the Knapsack Problem in two and three dimensions were developed in the last years, e.g., by Beasley (2004), Gonçalves and Resende (2006), Alvarez-Valdes et al. (2007), Morabito and Pureza (2007), Baldacci and Boschetti (2007), Hadjiconstantinou and Iori (2007), Cui and Zhang (2007), de Araujo and Armentano (2007), Egeblad and Pisinger (2009), Bortfeldt and Winter (2009), Parreño et al. (2010), Fanslau and Bortfeldt (2010) and Gonçalves and Resende (2012).

Here we have chosen three packing algorithms that are briefly characterized in Table 2. All of them show a good or excellent performance. Moreover, these algorithms observe the guillotine cutting constraint (C2) and it is a special topic of this paper to propose and examine RPAMP methods that satisfy this constraint. For each algorithm Table 2 also includes the parameter setting that is used here. For a detailed description of the algorithms the reader is referred to the mentioned papers.

Now eight RPAMP solution methods AMRH*i* ($i = 1,\ldots,8$) are derived from the generic procedure AMRH by means of the selected packing algorithms and the variants WV1 to WV3 for generating container widths. The methods are specified in Table 3 and organized in two groups of four methods each. In the first group 2D-KP instances are solved by means of the tree search algorithm CLTRS and in the second group by the genetic algorithm CBGAS; 2D-SPP instances are always calculated by the heuristic BFDH*. The width generation variant WV3 is applied with different maximum numbers of items per width combination as shown in the third column of Table 3. Further parameters of AMRH are set in a uniform fashion for all eight methods. The maximum number of tried width values *nw_ub* is always set to 150. There are five different increment ranks (i.e. $max\_irk = 5$) and the vector of increments is specified according to $I = (0.001, 0.005, 0.020, 0.050, 0.100)$. The parameter α for determining the maximal width w_{max} is set to 1.2. This allows for calculating an optimal solution in a square with a very low density of only 69.44%. Finally, an additional finishing criterion is introduced, i.e. the search is aborted if the density of the incumbent best solution reaches 99%. By these parameter settings, found by experiments, the eight RPAMP methods are completely specified.

All methods observe the guillotine cutting constraint (C2) while the orientation constraint (C1) is not taken into account here. Hence the methods are able to solve RPAMP instances of the subtypes RG and RF. All of them work in a deterministic fashion since BFDH* and CLTRS are deterministic algorithms, while in the genetic algorithm CBGAS the seed value of random numbers is fixed here (see Table 2).

The eight RPAMP methods were evaluated by means of preliminary numerical experiments. Each method was applied (once) to the 11 MCNC and GSCR instances and the four instances by Imahori et al. (2005). In column 4 of Table 3 the average density over the 15 instances is indicated. In columns 5 and 6 the numbers of shared and sole best values are listed that result if the methods are compared to each other. In column 7 the mean total run times (in seconds) are presented. Note that the time for running the dynamic programming procedure in the variants AMRH*i* ($i = 3,4,7,8$) is always included in the total run times in Table 3 as well as in the following tables.

The methods AMRH5 to AMRH8 with integrated genetic algorithm CBGAS perform generally better than the methods AMRH1 to AMRH4 that are based on the tree search algorithm CLTRS. Nevertheless, the latter methods show a potential to generate high quality solutions for smaller instances. The more sophisticated and "expensive" width generation variant WV3 proves its worth: 25 shared and six sole best values are originated by WV3 while the corresponding figures for the other width variants are 8 and 1, respectively, for WV1 and 3 and 0, respectively, for WV2. Width

Table 2
Selected packing algorithms.

2D-SPP algorithm BFDH[*]
Source: Bortfeldt (2006) (see p. 825f)
Type of approach: constructive heuristic
Structure of packing plans: layer structure, i.e. items are packed in consecutive disjoint layers
Used parameter setting: no parameters

2D-KP algorithm CLTRS
Source: Fanslau and Bortfeldt (2010)
Type of approach: tree search algorithm
Structure of packing plans: block structure, i.e. items are grouped in blocks of rectangular shape
Used parameter setting:
- First search phase is removed, only second phase with general blocks takes place
- Max. number of blocks *max_bl*: 10,000, min. space utilization per block *min_fr*: 100%
- Min. search width *min_ns*: 3, time limit per call: 7 s (2.61 GHz, see Section 4)

2D-KP algorithm CBGAS
Source: Bortfeldt and Gehring (2001) and Bortfeldt (2006)
Type of approach: genetic algorithm
Structure of packing plans: layer structure, i.e. items are packed in consecutive disjoint layers
Used parameter setting:
- In general the *standard parameter setting* of the GA (see Bortfeldt, 2006, p. 824, Table 4) is applied
- There are four exceptions: (1) The number of generations is set to *ngen* = 100, if the number of pieces is not greater than 100, and *ngen* = 20 otherwise. (2) Only one run is executed per KP instance (*nruns* = 1) and the *first* start distribution for the probabilities of the value combinations of layer parameters is used. (3) A reduction of instances is not carried out. (4) The seed value of random numbers is fixed to 1

generation variant WV1 performs well if it is combined with the genetic algorithm CBGAS while WV2 turns out to be a disappointing approach.

Finally, to diversify the search a combined procedure, abbreviated as AMRHC, is introduced as follows: AMRHC is composed by the two method variants AMRH4 and AMRH8. These variants can be considered the most successful RPAMP methods in terms of density and best values. Thus, AMRHC aims at utilizing the complementary strengths of the best RPAMP methods specified before.

For smaller instances with less than 100 rectangles the two method variants are run in the indicated order; for larger instances with 100 or more pieces the variant with integrated genetic algorithm CBGAS is executed first and the variant with integrated tree search algorithm CLTRS is run afterwards. In this way it is ensured that for smaller as well as for larger instances the method variant is called first that yields particular good results for the respective instance group. All parameter settings and rules defined before remain in force. In particular, the entire search is aborted after a density of 99% has been reached.

4. Final computational experiments

All RPAMP methods were implemented in C and all tests were run on AMD Athlon PCs (2.61 GHz, 2.00 GB RAM, Windows XP); time limits relate to this PC type. In the final computational experiments the combined procedure AMRHC is to be evaluated. First AMRHC is compared to RPAMP methods by other authors. Afterwards 24 new RPAMP benchmark instances are introduced and AMRHC results are reported.

4.1. Comparison with solution methods from literature

The combined heuristic AMRHC is compared to multiple RPAMP solution methods from the literature using the MCNC and GSRC benchmarks and the instances by Imahori et al. (2005). We have selected the methods proposed by Chan and Markov (2004), Imahori et al. (2005), Wu and Chan (2005), Kimura and Ida (2006), Chen and Huang (2007), Pisinger (2007), Fernando and Katkoori (2008) and Li et al. (2010). Moreover, a result for instance Rp100 obtained by the method of Nakatake et al. (1996) is also considered. To the best of our knowledge, these are the strongest solution procedures so far. Thus, it is certain that each currently known best density value for one of the 15 used instances is generated by at least one compared solution procedure. However, it must be stated that for the methods by Kimura and Ida (2006) and by Fernando and Katkoori (2008), respectively, only results could be presented for a simultaneous optimization of area and wire length while for all other methods the quoted results were achieved for a mere area minimization.

The results of the comparison are presented in the following two tables. In the first three columns of Table 4a the used instances are characterized by name, number and total area of pieces in μm^2. The instances Apte to Ami49 belong to set MCNC, the instances N10 to N300 belong to set GSRC while the instances Rp100 to Pcb500 were introduced by Imahori et al. (2005). The following eight columns of Table 4a are dedicated to the selected methods. Each one includes the *best* densities *d* (in%) achieved by the appropriate method for a subset of the 15 instances. As an exception Imahori et al. (2005) present only average values over 10 runs per instance. In column 12 the best density values *so far* are collected and these values are set in italic in columns 4–9. Columns 13 and 14 include the densities *d* for the combined heuristic AMRHC and for method variant AMRH8 with an additional time limit of 60 seconds for the total computation time. Note that no such time limit was imposed in Section 3.2. New best densities yielded by AMRHC are set in bold and in italic; densities generated by AMRH8 that are better than previous best values in column 12 are set in italic.

Table 3
Specification of RPAMP solution methods and results for 15 benchmark instances.

Name	2D-KP algorithm	Variant of width generation	Mean density in (%)	No. of shared best values	No. of sole best values	Mean total run time (s)
AMRH1	CLTRS	WV1	94.34	1	0	1152
AMRH2	CLTRS	WV2	97.79	1	0	1211
AMRH3	CLTRS	WV3, maxitems = 4	97.97	3	0	1167
AMRH4	CLTRS	WV3, maxitems = 6	97.82	6	3	1139
AMRH5	CBGAS	WV1	98.17	7	1	392
AMRH6	CBGAS	WV2	97.85	2	0	1183
AMRH7	CBGAS	WV3, maxitems = 3	98.02	7	0	247
AMRH8	CBGAS	WV3, maxitems = 4	98.23	9	3	261

Table 4a
Comparison with other solution methods (solution quality).

Instance	No. of pieces	Total area of pieces (μm^2)	Chan and Markov (2004) *d* (%)	Imahori et al. (2005) *d* (%)	Wu and Chan (2005) *d* (%)	Kimura and Ida (2006) *d* (%)	Chen and Huang (2007) *d* (%)	Pisinger (2007) *d* (%)	Fernando and Katkoori (2008) *d* (%)	Li et al. (2010) *d* (%)	Best so far *d* (%)	AMRHC *d* (%)	AMRH8 *d* (%)
Apte	9	465,61,628	*99.23* (99.23)[a]	–	–	–	99.23	*99.23*	–	99.23	99.23	99.23	99.23
Xerox	10	193,50,296	*97.75* (96.67)[a]	–	–	–	97.68	97.71	–	97.58	97.75	96.33	96.33
Hp	11	88,30,584	*98.70* (97.77)[a]	–	–	–	98.67	*98.70*	–	98.70	98.70	97.77	97.77
Ami33	33	11,56,449	92.52	–	98.25	74.29	98.84	*99.01*	96.37	98.09	99.01	99.01	98.27
Ami49	49	354,45,424	92.84	97.37	97.81	96.77	98.27	*98.48*	93.75	98.30	98.48	***98.58***	97.27
N10	10	221,679	–	–	–	–	–	–	*95.00*	–	95.00	***95.62***	*95.62*
N30	30	208,591	–	–	–	*94.44*	–	–	92.74	–	94.44	***97.79***	*96.87*
N50	50	198,579	–	–	–	91.99	–	–	91.51	*98.50*	98.50	98.44	98.09
N100	100	179,501	93.38	–	*99.85*	–	98.57	–	87.24	97.71	99.85	98.72	98.48
N200	200	175,696	91.97	–	95.91	–	*98.64*	–	81.63	96.82	98.64	***99.09***	*99.09*
N300	300	273,170	91.97	–	96.52	–	*98.75*	–	81.20	96.16	98.75	***99.03***	*99.03*
Rp100	100	205,056	–	96.78	–	–	–	–	–	–	97.08[b]	***99.06***	*99.06*
Pcb146	146	786,63,600	–	*96.71*	–	–	–	–	–	–	96.71	***98.85***	*98.28*
Rp200	200	410,112	–	*96.30*	–	–	–	–	–	–	96.30	***99.11***	*99.11*
Pcb500	500	13,58,165	–	*96.28*	–	–	–	–	–	–	96.28	***99.07***	*97.71*
Avg.	–	–	–	–	–	–	–	–	–	–	**97.65**	**98.38**	**98.00**

[a] Density in brackets is optimum for subtype RG.
[b] Density obtained by method of Nakatake et al. (1996), see Imahori et al. (2005).

In the last line average values are indicated only if results are available for all 15 instances. In the column of Chan and Markov (2004) their results for subtype RG (with guillotine constraint) and for the instances Apte, Xerox and Hp are also given. Note that all results for these instances indicated by Chan and Markov (2004) are optimum values.

Table 4b is similar to Table 4a and includes total run times for the eight comparison methods and for AMRHC and AMRH8. All run times *t* are given in seconds of the respective processor. At the end of Table 4b the mean run times over the calculated instances and the cycle frequencies (in GHz) of the used processors are indicated. Moreover, for each approach it is said whether it is stochastic and the performed number of runs per instance is given.

The results in Tables 4a and 4b are commented as follows:

- The combined solution method AMRHC achieves a high solution quality. New best solutions were found for 9 of the calculated 15 instances. Additionally, for instances Apte and Ami33 the known optimum/best value of subtype RF was reached while for Hp the optimum value of subtype RG has been met. The best known values were missed only for the instances Xerox, N50 and N100. The average value of the best known solutions was improved by 0.7% points.
- Solution method AMRHC does observe the guillotine cutting constraint (C2) while all compared methods do not. Thus, it has been shown experimentally that the substantial reduction of the solution space by constraint (C2) does not unavoidable lead to a reduction of solution quality. All new best solutions for the instances with 100 items or more have been generated by a method variant with integrated genetic algorithm CBGAS. Hence the solution space is further reduced in that only packing plans with a layer structure can be constructed (cf. Table 2). For large instances even this stricter reduction of the solution space allows for high quality solutions that are better than the best currently known solutions for subtype RF, i.e. without any restriction of the solution space.
- However, the computing times of the combined method AMRHC are admittedly higher than those of some of the compared methods and a stronger processor was used for the calculations, too. For example, Wu and Chan (2005) as well as Chen

Table 4b
Comparison with other solution methods (run time).

Instance	No. of pieces	Chan and Markov (2004) *t* (s)	Imahori et al. (2005) *t* (s)	Wu and Chan (2005) *t* (s)	Kimura and Ida (2006) *t* (s)	Chen and Huang (2007) *t* (s)	Pisinger (2007) *t* (s)	Fernando and Katkoori (2008) *t* (s)	Li et al. (2010) *t* (s)	AMRHC *t* (s)	AMRH8 (60 seconds) *t* (s)
Apte	9	2.38	–	–	–	0.01	0.58	–	0.40	1.00	1.00
Xerox	10	9812.00	–	–	–	0.01	0.68	–	0.70	1308.00	61.00
Hp	11	891.00	–	–	–	0.02	0.75	–	0.60	1286.00	59.00
Ami33	33	1.73	–	10.00	1800.00	2.01	1359.00	640.00	4.70	116.00	61.00
Ami49	49	3.01	100.00	77.00	1800.00	56.61	3004.00	1384.00	6.70	1752.00	62.00
N10	10	–	–	–	–	–	–	181.00	–	1301.00	61.00
N30	30	–	–	–	1800.00	–	–	?	–	1543.00	62.00
N50	50	–	–	–	1800.00	–	–	?	7.10	1711.00	63.00
N100	100	5.62	–	1.00	–	8.22	–	?	25.60	1915.00	61.00
N200	200	7.09	–	6.00	–	9.70	–	?	124.00	37.00	37.00
N300	300	11.04	–	17.00	–	37.23	–	4789.00	215.00	39.00	40.00
Rp100	100	–	200.00	–	–	–	–	–	–	59.00	59.00
Pcb146	146	–	300.00	–	–	–	–	–	–	2250.00	66.00
Rp200	200	–	400.00	–	–	–	–	–	–	14.00	14.00
Pcb500	500	–	1000.00	–	–	–	–	–	–	554.00	62.00
Avg.	–	1341.73	400.00	22.20	1800.00	14.23	873.00	1748.50	42.81	926.00	51.20
Cycle frequ. (GHz)		1.2	1.0	1.8	?	2.4	2.4	0.2	1.83	2.61	2.61
Stochastic approach		No	Yes	No	Yes	No	Yes	Yes	Yes	No	No
No. of runs		1	10	1	40	1	20	5	?	1	1

and Huang (2007) report on maximal computing times of approximately one minute. On the other hand there are other methods that also spend a large amount of time to achieve a top solution quality (e.g., the method by Pisinger (2007)) and the best densities of the stochastic approaches were sometimes calculated by means of a considerable number of runs per instance.

- Concerning the computational effort it can further be argued that an average total computing time of approximately 15 min may be regarded as moderate and tolerable in many applications (e.g., in industrial research and development). Moreover, it is a straightforward task to parallelize the method variants of the combined method by a local area network or a multi-core processor this leading to a drastic reduction of needed computing time. At last we refer to the results of the method variant AMRH8 with the imposed time limit of one minute. Again, for 8 of 15 instances new best solutions were achieved and the average density of the best known solutions has been improved slightly. These results were calculated in 51 s on average being a relatively short computation time.

In Figs. 2 and 3 packing plans for the instances Ami33 and Ami49 are plotted. The plan for Ami33 has a width of 693 units, a length of 1680 units and the density amounts to 99.33% being the best density that has been ever obtained. This plan has been calculated by method variant AMRH3 using a special configuration and within 7835 s. Consequently, the result does not appear in the above tables but it shows the strength of the developed heuristics.

The packing plan for Ami49 with a density of 98.58% was found by method variant AMRH4 (cf. Table 3), has width 4592 and length 7860.

4.2. New RPAMP benchmark instances and first results

For a further evaluation of the combined heuristic AMRHC 24 new RPAMP benchmark instances were used that are provided at www.fernuni-hagen.de/evis (look for file 24_rpamp_instances.zip).

Sets of benchmark instances for C&P problems should include a broad spectrum of instances with different characteristics in order to ensure a thorough evaluation of heuristics or exact solution methods. Relevant characteristics of 2D C&P problems were investigated and utilized by, e.g., Wang and Valenzuela (2001) and Bortfeldt and Gehring (2006) and some of them are also applied here. The new RPAMP instances were generated at random by means of the following characteristics and definitions:

- Number of rectangles n. 12 instances were generated with 50 rectangles and 12 instances have 200 rectangles.
- Maximum aspect ratio of all rectangles of an instance $\rho = \max\{\rho(i), i = 1,\ldots,n\}$. Here $\rho(i)$ is the aspect ratio of rectangle i calculated as $d_{max}(i)/d_{min}(i)$ where $d_{min}(i)$ denotes the smaller and $d_{max}(i)$ the greater dimension of rectangle i ($i = 1,\ldots,n$). Hence ρ represents an upper bound of the proportion of the sides of each rectangle and it guarantees so to speak that rectangles do not degenerate into line segments. 12 instances were generated with $\rho = 2$ and the other 12 instances with $\rho = 5$.
- Heterogeneity of the rectangles $v = n/nt$ where nt stands for the number of rectangle types. For 12 instances the heterogeneity v was set to 1, i.e. the instances are strongly heterogeneous (see Section 1). The other 12 instances are weakly heterogeneous since $v = 10$ was chosen, i.e. there are 10 rectangles per type.
- Share of small rectangles μ (in%). An instance has μ percent small and $(100 - \mu)$ percent large rectangles. The dimensions of small rectangles are chosen at random in the interval $[d^s_{min}, d^s_{max}]$ while the dimensions of large rectangles are taken randomly from the interval $[d^l_{min}, d^l_{max}]$. The interval bounds are defined as follows: $d^s_{min} = A^{1/2}/(2 * n^{1/2} * \rho^{1/2})$, $d^s_{max} = (A^{1/2} * \rho^{1/2})/(2 * n^{1/2})$, $d^l_{min} = (2 * A^{1/2})/(n^{1/2} * \rho^{1/2})$ and $d^l_{max} = (2 * A^{1/2} * \rho^{1/2})/n^{1/2}$. Here A stands for the specified total area of all rectangles. The interval bounds observe the maximum aspect ratio ρ and they assure a considerable difference in terms of area between small and large rectangles. The 24 instances are organized in three groups of eight instances each and the share of small rectangles μ was set to 70% for the first, 50% for the second and 30% for the third group. The total area of all rectangles A was specified by 200,000 area units for all instances.

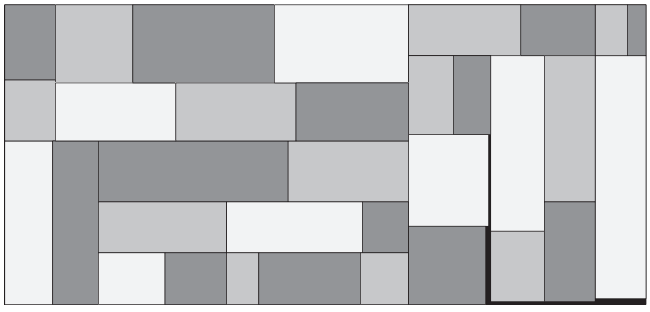

Fig. 2. Packing plan for instance Ami33 (*key:* black areas represent waste).

Fig. 3. Packing plan for instance Ami49 (*key:* black areas represent waste).

In Table 5 the 24 instances are numbered (columns 1 and 8) and the values of the characteristics n, ρ, v and μ are listed (columns 2–5 and 9–12). The combined heuristic AMRHC was applied to the 24 instances using the same configuration and test bed as in Section 4.1. However, the search is no longer aborted after a density of 99% (or higher) has been reached. The achieved densities (in %) are presented in columns 6 and 13 while the run times (in 2.61 gigahertz seconds) can be found in columns 7 and 14 of Table 5.

The results can be summarized as follows:

- AMRHC achieves a good solution quality for all instances and the smallest observed density amounts to 97.74%. A high mean density of 99.23% was reached for the large instances with 200 rectangles while the average density for the smaller instances with 50 rectangles amounts to 98.91% and is only little worse.
- For the 24 new instances a constantly high solution quality could be obtained without any reconfiguration of the heuristic AMRHC.
- The relatively long run times are due to the fact that 150 different width values are tried per instance and no time limit was set for the total run time in the interest of a high solution quality. However, the detailed results show that slightly worse densities can be obtained in much shorter run times.

Table 5
Definitions and results for 24 new RPAMP benchmark instances.

No.	n	ρ	v	μ (%)	d (%)	t (s)	No.	n	ρ	v	μ (%)	d (%)	t (s)
1	50	2	1	70	98.65	2501	13	200	2	1	70	99.44	2936
2	50	2	1	30	97.74	1553	14	200	2	1	30	99.26	4019
3	50	2	1	50	98.60	1697	15	200	2	1	50	99.39	3771
4	50	2	10	70	99.70	1335	16	200	2	10	70	99.23	2321
5	50	2	10	30	99.27	1420	17	200	2	10	30	99.20	1507
6	50	2	10	50	99.51	1410	18	200	2	10	50	99.01	1731
7	50	5	1	70	98.73	2972	19	200	5	1	70	99.61	9948
8	50	5	1	30	97.79	1556	20	200	5	1	30	99.13	3363
9	50	5	1	50	98.51	1785	21	200	5	1	50	99.50	6151
10	50	5	10	70	99.64	1362	22	200	5	10	70	99.47	2300
11	50	5	10	30	99.19	1410	23	200	5	10	30	98.75	1755
12	50	5	10	50	99.56	1434	24	200	5	10	50	98.72	1886
Avg	–	–	–	–	98.91	1702.9	–	–	–	–	–	99.23	3474.0

- Taking the results for the smaller instances it seems that a high density can be easier reached for weakly heterogeneous item sets. Note that all six smaller instances for which a 99.x% density was calculated belong to the instance group with heterogeneity $v = 10$. However, this guess has to be verified by additional experiments. A significant impact of the other characteristics (maximum aspect ratio and share of small rectangles) cannot be observed.

5. Conclusions

This paper presents a generic procedure for the RPAMP that is based on solution methods for the 2D Knapsack Problem and the 2D Strip Packing Problem. By this approach tackling an RPAMP instance is reduced to solving multiple 2D-SPP and 2D-KP instances. Eight fully defined RPAMP solution methods were derived from the generic procedure. Four of them integrate a tree search algorithm from the literature for solving the 2D-KP instances while the other four methods make use of a well-known genetic algorithm for the same purpose. The eight methods also differ regarding the applied strategy for defining container width values of the 2D-KP instances that are to be solved in the course of the search. The two most promising methods were selected for a combined procedure denoted by AMRHC.

AMRHC was tested by means of 15 well-known RPAMP benchmark instances that come from the sets MCNC and GSRC or were proposed by Imahori et al. (2005). For 10 instances new best solutions could be obtained. The run times can be regarded as acceptable, at least for many applications. The combined procedure observes the guillotine cutting constraint while the competing RPAMP methods from the literature do not satisfy this constraint, i.e. better solutions were obtained in a smaller solution space.

In addition, 24 new RPAMP instances with 50 or 200 rectangles were introduced. For these instances AMRHC has achieved a constantly top solution quality with an average density of more than 99%.

The 24 new RPAMP instances are provided in the Web to stimulate experiments with other solution methods. In a future version the procedure AMRHC should have learned to cope with further relevant constraints.

Appendix A

The combinatorial problem we deal with here may be called Interval Subset Sum Problem (ISSP): we are given n ($n > 1$) positive integers w_i, an interval $[w_{min}, w_{max}]$ (w_{min}, w_{max} positive integers) and a positive integer p_{max}. Each p-tuple $(w_{i_1}, w_{i_2}, \ldots, w_{i_p})$ with $i_k < i_{(k+1)}$ ($k = 1, \ldots, p-1$, $1 \leqslant i_1$, $i_p \leqslant n$) is called a selection and p is called its length ($p \geqslant 1$, integer). A selection $s = (w_{i_1}, w_{i_2}, \ldots, w_{i_p})$ is said to have sum w if $w_{i_1} + w_{i_2} + \cdots + w_{i_p} = w$. For each value $w \in [w_{min}, w_{max}]$ and each length $p \in [1, p_{max}]$ we have to determine the number $N^p(w)$ of all different selections of length p and sum w. Moreover, we have to determine the number $N(w)$ of all different selections with sum w ($w \in [w_{min}, w_{max}]$) and a length $p \leqslant p_{max}$. One can assume without loss of generality that $w_i \leqslant w_{max}$ and $w_1 + w_2 + \cdots + w_n \geqslant w_{min}$ ($i = 1, \ldots, n$).

To solve the ISSP a dynamic programming procedure is proposed that is based on the additional assumption $w_i < w_{i+1}$, $i = 1, \ldots, n-1$, i.e., we will constrain our procedure to the case that all integers w_i are different.

Let $N^{p,i}(w)$ denote the number of all selections s of length p and sum w with the additional attribute that the last value of s is w_i ($1 \leqslant i \leqslant n$, $1 \leqslant w \leqslant w_{max}$, $1 \leqslant p \leqslant p_{max}$). $N^{p,i}(w)$ is set to 0 if $w \leqslant 0$; note that $N^{p,i}(w) = 0$ if $i < p$. Now we have obvious equations for selections of length $p = 1$:

$$N^{1,i}(w) = 1, \quad \text{if } w = w_i \quad \text{for an } i \in \{1, \ldots, n\} \quad \text{and} \quad 0 \text{ otherwise}, \; w \in [1, w_{max}]. \tag{A1}$$

We observe that each selection s' with sum w, length $p+1$ and with last value w_i can be obtained from exactly one selection s with sum $w - w_i$ and length p. This selection s results from s' by removing the last value (or summand) w_i. Hence we have the following recursion:

$$N^{p+1,i}(w) = \sum_{j<i} N^{p,j}(w - w_i), \quad i \in \{1, \ldots, n\}, \quad p \in \{1, \ldots, p_{max} - 1\}, \quad w \in [1, w_{max}]. \tag{A2}$$

By means of relations (A1) and (A2) we are obviously able to calculate all numbers $N^{p,i}(w)$ ($1 \leqslant i \leqslant n$, $w_{min} \leqslant w \leqslant w_{max}$, $1 \leqslant p \leqslant p_{max}$). Finally, the numbers to be determined result by the obvious equations $N^p(w) = \sum_{i=1}^{n} N^{p,i}(w)$, ($w_{min} \leqslant w \leqslant w_{max}$, $1 \leqslant p \leqslant p_{max}$) and $N(w) = \sum_{p=1}^{p_{max}} N^p(w)$, ($w_{min} \leqslant w \leqslant w_{max}$).

The calculation of the recursions (A2) constitutes the most time-consuming part of the procedure. It is easily seen that this calculation has a worst case complexity of $O(w_{max} n^2)$. Hence the dynamic programming procedure for the ISSP has pseudo-polynomial complexity. The procedure can be easily extended in such a way that all different selections for all sum values w of an interval $[w_{min}, w_{max}]$ with exactly p summands (or at most p summands) are output.

Finally, we comment on the application of the procedure. First, the experiments showed that better results can be obtained if the width values w of an interval $[w_{min}, w_{max}]$ are evaluated by the numbers of selections with *exactly* p_{max} summands (instead by the numbers of selections with at most p_{max} summands). Second, the procedure was applied without modification to instances where some of the rectangle dimensions appear several times.

Judging by the results this should not have had a significant negative impact. An adaptation of the procedure to the weakly heterogeneous case is left to further research. Third, the procedure needed only about 5 s for the largest calculated benchmark instance with 1000 lengths (instance Pcb500).

References

Adya, S.N., Markov, I.L., 2003. Fixed-outline floorplanning: enabling hierarchical design. IEEE Transactions on VLSI 11, 1120–1135.

Allen, S.D., Burke, E.K., Kendall, G., 2011. A hybrid placement strategy for the three-dimensional strip packing problem. European Journal of Operational Research 209, 219–227.

Alvarez-Valdes, R., Parreño, F., Tamarit, J.M., 2007. A Tabu search algorithm for two-dimensional non-guillotine cutting problems. European Journal of Operational Research 183, 1167–1182.

Alvarez-Valdes, R., Parreño, F., Tamarit, J.M., 2008. Reactive GRASP for the strip packing problem. Computers and Operations Research 35, 1065–1083.

Baldacci, R., Boschetti, M.A., 2007. A cutting-plane approach for the two-dimensional orthogonal non-guillotine cutting problem. European Journal of Operational Research 183, 1136–1149.

Beasley, J.E., 2004. A population heuristic for constrained two-dimensional non-guillotine cutting. European Journal of Operational Research 156, 601–627.

Belov, G., Scheithauer, G., Mukhacheva, E.A., 2008. One-dimensional heuristics adapted for two-dimensional rectangular strip packing. Journal of the Operational Research Society 59, 823–832.

Beltran-Cano, D., Melian-Batista, B., Marcos Moreno-Vega, J., 2009. Solving the rectangle packing problem by an iterative hybrid heuristic. In: Moreno-Diaz, R., et al. (Eds.), EUROCAST 2009, LNCS 5717, pp. 673–680.

Bortfeldt, A., 2006. A genetic algorithm for the two-dimensional strip-packing problem. European Journal of Operational Research 172, 814–837.

Bortfeldt, A., Gehring, H., 2001. A hybrid genetic algorithm for the container loading problem. European Journal of Operational Research 131, 143–161.

Bortfeldt, A., Gehring, H., 2006. New large benchmark instances for the two-dimensional strip packing problem with rectangular pieces. In: Proc. of the 39th Annual Hawaii International Conference on System Sciences 2006, HICSS '06, vol. 2, p. 30b.

Bortfeldt, A., Mack, D., 2007. A heuristic for the three-dimensional strip packing problem. European Journal of Operational Research 183, 1267–1279.

Bortfeldt, A., Winter, T., 2009. A genetic algorithm for the two-dimensional knapsack problem with rectangular pieces. International Transactions in Operational Research 16, 685–713.

Chan, H.H., Markov, I.L., 2004. Practical slicing and non-slicing block-packing without simulated annealing. In: Proc. of the ACM Great Lakes Symposium on VLSI, pp. 282–287.

Chang, Y.-C., Chang, Y.-W., Wu, G.-M., Wu, S.-W., 2000. B*-Trees: a new representation for non-slicing floorplans. In: Proc. of ACM/IEEE Design Automation Conference, pp. 458–464.

Chatterjee, D., Manikas, T.W., 2007. A genetic algorithm for non-slicing floorplan representation. In: Proc. of the National Conference on Intelligent Systems, Hyderabad.

Chen, M., Huang, W., 2007. A two-level search algorithm for 2D rectangular packing problem. Computers & Industrial Engineering 53, 123–136.

Chen, S., Dong, S., Hong, X., 2005. A new interconnect-aware floorplan representation and its application to floorplanning targeting buffer planning. In: Proc. of 48th Midwest Symposium on Circuits and Systems, pp. 1071–1074.

Chen, J., Guo, W., Chen, G., 2010. A PSO-based intelligent decision algorithm for VLSI floorplanning. Soft Computing 14, 1329–1337.

Chen, J., Zhu, W., Ali, M.M., 2011a. A hybrid simulated annealing algorithm for nonslicing VLSI floorplanning. IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews 41, 544–553.

Chen, X., Hu, J., Xu, N., 2011b. Regularity-constraint floorplanning for multi-core processors. In: Proc. of the 2011 International Symposium on Physical design (ISPD '11), ACM, New York, 2011.

Christofides, N., Whitlock, C., 1977. An algorithm for two-dimensional cutting problems. Operations Research 25, 30–44.

Clautiaux, F., Jouglet, A., Carlier, J., Moukrim, A., 2008. A new constraint programming approach for the orthogonal packing problem. Computers and Operations Research 35, 944–959.

Cui, Y., Zhang, X., 2007. Two-stage general block patterns for the two-dimensional cutting problem. Computers and Operations Research 34, 2882–2893.

De Araujo, O.C.B., Armentano, V.A., 2007. A multi-start random constructive heuristic for the container loading problem. Pesquisa Operacional 27, 311–331.

Dong, S., Yang, Z., Hong, X., Wu, Y., 2004. Module placement based on quadratic programming and rectangle packing using less flexibility first principle. In: Proc. of the 2004 International Symposium on Circuits and Systems, pp. 61–64.

Drakidis, A., Mack, R.J., Massara, R.E., 2006. Packing-based VLSI module placement using genetic algorithm with sequence-pair representation. IEEE Proceedings: Circuits Devices and Systems 153, 545–551.

Egeblad, J., Pisinger, D., 2009. Heuristic approaches for the two- and three-dimensional knapsack packing problem. Computers and Operations Research 36, 1026–1049.

Fanslau, T., Bortfeldt, A., 2010. A tree search algorithm for solving the container loading problem. INFORMS Journal on Computing 22, 222–235.

Fernando, P., Katkoori, S., 2008. An elitist non-dominated sorting based genetic algorithm for simultaneous area and wirelength minimization in VLSI floorplanning. In: Proc. of 21st International Conference on VLSI Design, Hyderabad, pp. 337–342.

Gonçalves, J.F., Resende, M.G.C., 2006. A Hybrid Heuristic for the Constrained Two-dimensional Non-guillotine Orthogonal Cutting Problem. AT&T Labs Research Technical report TD-&UNQN6.

Gonçalves, J.F., Resende, M.G.C., 2012. A parallel multi-population biased random-key genetic algorithm for a container loading problem. Computers and Operations Research 39, 179–190.

Guo, P.N., Cheng, C.K., Yoshimura, T., 1999. An O-tree representation of nonslicing floorplans and its applications. In: Proc. of ACM/IEEE Design Automation Conference, pp. 268–273.

Hadjiconstantinou, E., Iori, M., 2007. A hybrid genetic algorithm for the two-dimensional single large object placement problem. European Journal of Operational Research 183, 1150–1166.

Hong, X.L., Huang, G., Cai, Y.C., 2000. Corner block list: an effective and efficient topological representation of non-slicing floorplan. In: Proc. of IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp. 8–12.

Imahori, S., Yagiura, M., Ibaraki, T., 2005. Improved local search algorithms for the rectangle packing problem with general spatial costs. European Journal of Operational Research 167, 48–67.

Janiak, A., Kozik, A., Lichtenstein, M., 2010. New perspectives in VLSI design automation: deterministic packing by sequence pair. Annals of Operations Research 179, 35–56.

Kimura, Y., Ida, K., 2006. Improved genetic algorithm for VLSI floorplan design with non-slicing structure. Computers and Industrial Engineering 50, 528–540.

Korf, R.E., Moffitt, M.D., Pollack, M.E., 2010. Optimal rectangle packing. Annals of Operations Research 179, 261–295.

Leung, S.C.H., Zhang, D., Sim, K.M., 2011. A two-stage intelligent search algorithm for two dimensional strip packing problem. European Journal of Operational Research 215, 57–69.

Li, Yiming., Li, Yi., Zhou, M., 2010. Area optimization in floorplanning using AP-TCG. Journal of Convergence Information Technology 5, 216–222.

Lin, J.M., Chang, Y.W., 2001. TCG: a transitive closure graph-based representation for non-slicing floorplans. In: Proc. of the Design Automation Conference, pp. 764–769.

Lin, J.-M., Chang, Y.-W., 2002. TCG-S: Orthogonal coupling of P*-admissible representations for general floorplans. In: Proc. of the Design Automation Conference, pp. 842–847.

Lin, J.-M., Chang, Y.-W., 2003. TCG: a transitive closure graph-based representation for general floorplans. IEEE Transactions on Very Large Scale Integration Systems XX (Y), 2003.

Lin, C.-T., Chen, D.-S., Wang, Y.-W., 2002. GPE: a new representation for VLSI floorplan problem. In: Proc. of the IEEE International Conference on Computer Design: VLSI in Computers and Processors (ICCD'02), pp. 42–44.

Lin, P.-H., Chang, Y.-W., Lin, S.-C., 2009. Analog placement based on symmetry-island formulation. IEEE Transactions on CAD of Integrated Circuits and Systems 28, 791–804.

Lodi, A., Martello, S., Vigo, D., 1999. Heuristic and metaheuristic approaches for a class of two-dimensional bin packing problems. INFORMS Journal on Computing 11, 345–357.

Maag, V., Berger, M., Winterfeld, A., Küfer, K.-H., 2010. A novel non-linear approach to minimal area rectangular packing. Annals of Operations Research 179, 243–260.

Morabito, R., Pureza, V., 2007. Geração de padrões de cortes bidimensionais guilhotinados restritos via programação dinâmica e busca em grafo-e/ou. Produção 17(1), 033–051.

Murata, H., Fujiyoshi, K., Nakatake, S., Kajitani, Y., 1996. VLSI module placement based on rectangle-packing by the sequence-pair. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 15, 1518–1524.

Nakatake, S., Murata, H., Fujiyoshi, K., Kajitani, Y., 1996. Block placement on BSG-structure and IC layout application. In: Proc. of International Conference on Computer Aided Design, pp. 484–490.

Pang, Y., Cheng, C.K., Yoshimura, T., 2000. An enhanced perturbing algorithm for floorplan design using the O-tree representation. In: Proc. of International Symposium on Physical Design, pp. 168–173.

Parreño, F., Alvarez-Valdez, R., Oliveira, J.F., Tamarit, J.M., 2010. Neighborhood structures for the container loading problem: a VNS implementation. Journal of Heuristics 16, 1–22.

Pisinger, D., 2007. Denser packings obtained in O(nloglogn) time. INFORMS Journal on Computing 19, 395–405.

Rahim, H.A., Rahman, A.A.A., Ahmad, R.B., Arifin, W.N.F.W., Ahmad, M.I., 2008. The performance study of two genetic algorithm approaches for VLSI macro-cell layout area optimization. In: Proc. of Second Asia International Conference on Modelling and Simulation, Kuala Lumpur, pp. 207–212.

Sun, T.-Y., Hsieh, S.-T., Wang, H.-M., Lin, C.-W., 2006. Floorplanning based on particle swarm optimization. In: Proc. of the Emerging VLSI Technologies and Architectures, Karlsruhe, 2006.

Tang, M., Sebastian, A., 2005. A genetic algorithm for VLSI floorplanning using O-tree representation. In: Rothlauf, F., et al. (Eds.), EvoWorkshops 2005, LNCS 3449, Springer, Berlin, pp. 215–224.

Tang, X., Wong, D.F., 2001. FAST-SP: a fast algorithm for block placement based sequence pair. In: Proc. of the Design Automation Conference, pp. 521–526.

Valenzuela, C.L., Wang, P.Y., 2002. VLSI placement and area optimization using a genetic algorithm to breed normalized postfix expressions. IEEE Transactions on Evolutionary Computation 6, 390–401.

Wang, P.Y., Valenzuela, C.L., 2001. Data set generation for rectangular placement problems. European Journal of Operational Research 134, 378–391.
Wäscher, G., Haußner, H., Schumann, H., 2007. An improved typology of cutting and packing problems. European Journal of Operational Research 183, 1109–1130.
Wu, Y.-L., Chan, C.-K., 2005. On Improved Least Flexibility First Heuristics Superior for Packing and Stock Cutting Problems. LNCS 3777. Springer, Berlin, pp. 70–81.
Wu, Y.-L., Huang, W., Lau, S.-C., Wong, C.K., Young, G.H., 2002. An effective quasi-human based heuristic for solving the rectangle packing problem. European Journal of Operational Research 141, 341–358.
Yao, B., Chen, H., Cheng, C.K., Graham, R., 2001. Revisiting floorplan representations. In: Proc. of the International Symposium on Physical Design, pp. 138–143.
Young, E.F.Y., Chu, C.C.N., Sgen, Z.C., 2003. Twin binary sequences: a non-redundant, representation for general non-slicing floorplan. IEEE Transactions on CAD 22, 457–469.
Zhou, H., Wang, J., 2004. ACG – Adjacent constraint graph for general floorplans. In: Proc. of the IEEE International Conference on Computer Design, San Jose, pp. 572–575.
Zhou, S., Dong, S., Hong, X., Cai, Y., Cheng, C.K., Gu, J., 2001. ECBL, an extended corner block list with solution space including optimum placement. In: Proc. of Int. Symposium on Physical Design, pp. 156–161.
Zhuang, C., Sakanushi, K., Jin, L., Kajitani, Y., 2002. An enhanced Q-sequence augmented with empty-room-insertion and parenthesis trees. Proceedings of Design, Automation and Test in Europe 2002, 61–68.