J Glob Optim (2009) 43:277–297
DOI 10.1007/s10898-007-9271-9

Column enumeration based decomposition techniques for a class of non-convex MINLP problems

Steffen Rebennack · Josef Kallrath · Panos M. Pardalos

Received: 15 August 2007 / Accepted: 9 December 2007 / Published online: 28 December 2007

Abstract We propose a decomposition algorithm for a special class of nonconvex mixed integer nonlinear programming problems which have an assignment constraint. If the assignment decisions are decoupled from the remaining constraints of the optimization problem, we propose to use a column enumeration approach. The master problem is a partitioning problem whose objective function coefficients are computed via subproblems. These problems can be linear, mixed integer linear, (non-)convex nonlinear, or mixed integer nonlinear. However, the important property of the subproblems is that we can compute their exact global optimum quickly. The proposed technique will be illustrated solving a cutting problem with optimum nonlinear programming subproblems.

Keywords MINLP · Column enumeration · Decomposition · Packing

1 Introduction

Problems assigning projects to departments, activities to units, delivery orders to vehicles, or packing geometric objects to given areas are well known real world problems. Modeling these problems mathematically can easily lead to nonconvex mixed integer nonlinear programming (MINLP) problems. Solving them is in general very challenging as they belong to the 'harder' cases of the $\mathcal{NP}-hard$ problems.

Column enumeration is a special variant of column generation and is applicable when a small number of columns is sufficient. This is, for instance, the case in real world cutting

S. Rebennack (✉) · P. M. Pardalos
Center of Applied Optimization, University of Florida, Gainesville, FL 32611, USA
e-mail: steffen@ufl.edu

J. Kallrath
Department of Astronony, University of Florida, Gainesville, FL 32611, USA
e-mail: kallrath@astro.ufl.edu

P. M. Pardalos
e-mail: pardalos@ufl.edu

stock problems when it is known that the optimal solution have only a small amount of trimloss. This, usually, eliminates most of the pattern. Column enumeration naturally leads to a type of selecting columns or partitioning models. A collection of illustrative examples contained in Schrage (2006, Sect. 11.7) covers several problems of grouping, matching, covering, partitioning, and packing in which a set of given objects has to be grouped into subsets to maximize or minimize some objective function. Discussion of column enumeration in the context of column generation can be found in Wilhelm (2001). Despite the limitations with respect to the number of columns, column enumeration has some advantages:

- no pricing problem — in contrast to column generation,
- easily applicable to mixed integer programming (MIP) problems,
- column enumeration is relatively easy to implement.

Decomposition methods for mathematical programming have been widely studied in the past and used especially to solve large-scale optimization problems. A survey can be found in the book edited by Pardalos and Resende (2002, Chap. 7). The decomposition approach we develop, is a generalization of the one by Kallrath (2004, Chap. 4.3.3) who solves an online-version of a vehicle routing problem. In this application, a column is a set of orders. The subproblem is to route and schedule the orders assigned to vehicles. For solving the routing (sequencing) and scheduling problem she developed a tailor-made solution algorithm. In this article, we apply such an approach to the packing or cutting problem described by Kallrath (2008) where the columns are subsets of the objects to be packed into given rectangular plates of known size. The subproblems are nonconvex nonlinear programming (NLP) problems. However, it is possible to solve them to global optimality in short time.

The structure of this article is as follows. A detailed problem description is provided in Sect. 2. The solution approach is discussed in Sect. 3. Generalizations of the decomposition approach are provided in Sect. 4. In Sect. 5, we describe the column concept and dominance rules to be applied to a problem in which a set of objects, circles, need to be assigned to a set of given rectangular plates. In addition, some numerical experiments and results are included. Section 6 summarizes this work and gives an outlook on generalizations and future research enhancements. In the Appendix, we discuss some analytical solutions for cutting up to three circles from a rectangular plate.

2 Problem description

The nonconvex MINLP problems we want to solve are assignment problems in which a set $\mathcal{I}$ of n objects should be allocated to a set of resources $\mathcal{R}$. Each object $i \in \mathcal{I}$ needs to be allocated to exactly one resource $r \in \mathcal{R}$. Resource r can cover several objects. The allocation decisions are represented by binary variables α_{ir} being 1, if object i is allocated to a resource r and 0 otherwise. We assume the following four structural properties.

Ass. 1: The cost for assigning objects to resources are additive with respect to the resources.
Ass. 2: There are no cost involved if resource r is not used.
Ass. 3: The assignment constraints are decoupled from the remaining constraints.
Ass. 4: The constraints associated with resource r are satisfied in case resource r is not assigned any object.

Examples for a problem meeting this framework are the vehicle routing problem in which orders are allocated to vehicles, or the problem of assigning and cutting geometric objects from given steel plates.

Let the optimization problem contain the continuous variables $x \in \mathbb{R}^{n_c}$, the integer variables $y \in \mathbb{Z}^{n_g}$ and the binary variables $\alpha \in \{0,1\}^n$ representing the allocation decisions. We define vector $\alpha_{.r} := (\alpha_{1r}, \alpha_{2r}, \ldots, \alpha_{nr})$. Assumption Ass. 1 gives us that the total cost to minimize is the sum over the resources r of a cost function f depending on the variables x, y and $\alpha_{.r}$. Hence, Ass. 1 states that the objective function is separable with respect to $\alpha_{.r}$. We get from Ass. 2 that $\alpha_{.r} = \vec{0}$ implies

$$f(x, y, \vec{0}) = 0. \tag{1}$$

Let $g(x, y, \alpha_{.r}) = \vec{0}$ and $h(x, y, \alpha_{.r}) \geq \vec{0}$ be the constraint functions. Recognize that with Ass. 3, the constraint functions g and h depend only on the assignment of the objects to a particular resource r, but not on all the resource; i.e., f and g only depend on $\alpha_{.r}$ instead of α. Ass. 4 can then be expressed in formula as: $\alpha_{.r} = \vec{0}$ implies $g(x, y, \alpha_{.r}) = \vec{0}$ and $h(x, y, \alpha_{.r}) \geq \vec{0}$. Finally, the optimization problem can be written as

$$F := \min \sum_r f(x, y, \alpha_{.r}) \tag{2}$$

subject to

$$\sum_r \alpha_{ir} = 1; \quad \forall i \tag{3}$$

and

$$g(x, y, \alpha_{.r}) = \vec{0}; \quad \forall r \tag{4}$$

$$h(x, y, \alpha_{.r}) \geq \vec{0}; \quad \forall r, \tag{5}$$

as well as the domain constraints

$$x \in \mathbb{R}^{n_c}, \quad y \in \mathbb{Z}^{n_g}, \quad \alpha \in \{0,1\}^n. \tag{6}$$

The objective function and the constraint functions g and h are problem-specific and can be of any type—except for the structural assumptions we make. Hence, in most cases this problem is nonconvex. If there are many resources r to consider, then the problem becomes quickly very large and difficult to solve to global optimality.

3 Solution approach

If the assignment decisions are decoupled from the remaining constraints of the optimization problem (Ass. 3 of Sect. 2), we propose to use a column enumeration approach. A column c is any index set $\mathcal{I}_c \subseteq \mathcal{I}$. The set $\mathcal{C}$ is the power set of $\mathcal{I}$ and contains $2^n - 1$ columns, not counting the empty set. As the number of columns grows exponentially in n, we can use a complete enumeration only for small values of n, i.e., smaller than 10 or 15. We obtain a decomposed problem with respect to the columns c. The master problem is a partitioning problem whose objective function coefficients are computed in several subproblems. Such a subproblem can be a linear programming (LP), mixed integer linear programming (MILP), NLP, or MINLP problem. In addition to the four structural properties of Sect. 2, the subproblems have to meet the following important properties.

Ass. 5: The submodel contains only constraints connected to the objects contained in column c. We call this the composition requirement.

Ass. 6: We can compute the exact global optimum of the submodel in short time.

The master problem is the partitioning model

$$F_M := \min \sum_{cr} F_{cr}\delta_{cr} \tag{7}$$

subject to ensuring that each object i is exactly contained in one resource, where the indicator function $I_c(i) = 1$ if column c contains object i and 0 otherwise:

$$\sum_{cr|I_c(i)=1} \delta_{cr} = 1; \quad \forall i, \tag{8}$$

The following constraints guarantee that each resource r is used at most for one column c,

$$\sum_{c} \delta_{cr} \leq 1; \quad \forall r. \tag{9}$$

Finally, there are the domain constraints for the binary variable δ

$$\delta_{cr} \in \{0, 1\}; \quad \forall c, r. \tag{10}$$

Note that (8) is the counterpart of (3) in the original problem.

For each column and resource combination, the coefficients F_{cr} are computed via subproblems. For each subproblem, the column is fix, defining

$$\beta_i = \begin{cases} 1, & \text{if } i \in \mathcal{I}_c \\ 0, & \text{otherwise} \end{cases}; \quad \forall i.$$

The subproblems P_{cr} are then defined as

$$P_{cr}: \quad \overline{F}_{cr} = \min f(x, y, \beta) \tag{11}$$

subject to

$$g(x, y, \beta) = 0 \tag{12}$$

$$h(x, y, \beta) \geq 0, \tag{13}$$

and

$$x \in \mathbb{R}^{n_c}, \quad y \in \mathbb{Z}^{n_g}. \tag{14}$$

Finally, this allows us to define the objective function coefficients

$$F_{cr} = \begin{cases} \overline{F}_{cr}, & \text{if problem (7) to (10) is feasible} \\ M, & \text{otherwise} \end{cases}; \quad \forall cr, \tag{15}$$

for the master problem, where M is a number sufficiently large. However, for computational purposes, one would fix the corresponding δ_{cr} variable to 0 and assign an arbitrary (finite) number as its coefficient F_{cr}. We call the master problem infeasible, if $F_M = M$.

Theorem 3.1 *Problem (2)–(6) is equivalent to the master problem (7)–(10) with the coefficients defined in (15) and obtained from the subproblems (11)–(14).*

Proof We show that there is a one-to-one correspondence of the solutions to the original problem (2)–(6) and the master problem (7)–(10); which means that if one of the problems is feasible and has a bounded optimal solution, then we can derive a solution to the other problem with the same objective function value. Furthermore, the original problem has a bounded optimal solution if and only if the master problem also has one.

Assume that the original problem defined by (2) is feasible and let $(\widehat{x}, \widehat{y}, \widehat{\alpha})$ be a finite optimal solution. Now, define

$$\delta_{cr} := \begin{cases} 1, & \text{if } \widehat{\alpha}_{ir} = 1 \forall i \in \mathcal{I}_c \\ 0, & \text{otherwise} \end{cases}, \quad \forall c, r. \tag{16}$$

This definition implies

$$\sum_r \widehat{\alpha}_{ir} = \sum_{cr|I_c(i)=1} \delta_{cr}.$$

Notice that $\sum_i \widehat{\alpha}_{ir}$ is the number of objects assigned to resource r. In addition, this defines the unique column $\overline{c}$, associated with resource r, as $\overline{c} = \cup_{i|\widehat{\alpha}_{ir}=1}\{i\}$ ensuring that δ satisfies constraint (9). Hence, δ is a feasible solution to the master problem. If $\widehat{\alpha}_{\cdot r} = \vec{0}$, then we get from the definition (16) that $\delta_{cr} = 0$ for all c, meaning that resource r is not assigned any object. As we have $M \cdot 0 = 0$, we get that $\sum_c F_{cr}\delta_{cr} = 0$. Otherwise, there exists exactly one $\overline{c}$ such that $\delta_{\overline{c}r} = 1$ and $\sum_c F_{cr}\delta_{cr} = F_{\overline{c}r}$. (11) implies that $\overline{F}_{\overline{c}r} = \min f(x, y, \widehat{\alpha}_{\cdot r})$ where variables (x, y) satisfy constraints (12)–(14). From the composition requirement of the constraints g and h, Ass. 5, we get that $(x, y) := (\widehat{x}, \widehat{y})$ is a global optimal solution to problem P_{cr}. Hence, we have that $F_{\overline{c}r} = \overline{F}_{\overline{c}r}$. With Ass. 2, this leads to $\sum_{cr} F_{cr}\delta_{cr} = \sum_r f(\widehat{x}, \widehat{y}, \widehat{\alpha}_{\cdot r})$. All this yields now to

$$F \geq F_M.$$

Vice versa, let $\widehat{\delta}$ be a finite optimal solution to the master problem. Define

$$\alpha_{ir} := \begin{cases} 1, & \text{if } \exists c : \widehat{\delta}_{cr} = 1 \wedge I_c(i) = 1 \\ 0, & \text{otherwise} \end{cases}, \quad \forall i, r.$$

With this definition, constraint (8) implies that α satisfies constraint (3). With assumption Ass. 4 and Ass. 5, constraints (8) and (9), the $\widehat{x}$ and $\widehat{y}$ solutions to the subproblem P_{cr} can be glued together in an obvious way to (x, y), satisfying constraints (4)–(6). Now, if there exists no $\overline{c}$ such that $\delta_{\overline{c}r} = 1$, then $\sum_c F_{cr}\delta_{cr} = 0$. This equals $f(x, y, \alpha_{\cdot r})$ according to Ass. 2. Otherwise, if there exists such a $\overline{c}$, then $\sum_c F_{cr}\delta_{cr} = F_{\overline{c}r} = \overline{F}_{\overline{c}r} = f(x, y, \alpha_{\cdot r})$. This shows that

$$F \leq F_M.$$

With the construction of the solutions above, we get that the original problem is feasible with a finite optimal solution, if and only if the master problem is feasible. □

Depending on functions f, g, and h, the subproblem P_{cr} can be a LP, MILP, NLP or MINLP problem. Again we mention that the important property of the subproblem is that we can compute its global minimum quickly.

Note that the master problem consists of an exponential number of binary variables. More precisely, the master problem (7)–(10) has $\left(2^{|\mathcal{I}|} - 1\right)|\mathcal{R}|$ binary variables and $|\mathcal{I}| + |\mathcal{R}|$ constraints.

The generation of the columns is problem-specific. The sequence in which to solve them is also problem-specific and allows us to apply dominance rules and feasibility implications. We illustrate this in detail for a cutting problem in Sect. 5.

4 Generalizations

In this section, we provide various generalizations classified into those which do not lead to a modified column generation schemes or submodels, and those which do.

4.1 Unchanged subproblems

The problem can be generalized or abstracted from the original meaning to treat any problem with binary variables α_{ir} and the assignment constraints (3)

$$\sum_r \alpha_{ir} = 1; \quad \forall i$$

as long as the subproblem only involves the objects contained in the column and no other ones, i.e., the assignment decisions can be decoupled from the rest of the problem.

A first way of generalization is obviously to replace (3) by

$$\sum_r \alpha_{ir} \circledast B_i; \quad \forall i, \tag{17}$$

where $B_i \in \mathbb{N}$ and the relation $\circledast \in \{=, \leq, \geq\}$, implies that object i has to be assigned to exactly B_i resources, to at most B_i resources, or to at least B_i resources. In that case, (8) is replaced by

$$\sum_{cr|I_c(i)=1} \delta_{cr} \circledast B_i; \quad \forall i. \tag{18}$$

Note that as object i could now be contained in several columns, the dominance rules could be affected and need to be revised.

A second step of generalization is to consider constraints of the type

$$\sum_r A_{ir}\alpha_{ir} \circledast B_i; \quad \forall i. \tag{19}$$

Let us explain the meaning of the coefficients A_{ir} in (19) by considering only the case $\sum_r A_{ir}\alpha_{ir} \leq B_i$. If object i is allocated to resource r a certain cost A_{ir} has to be paid. The total cost is not allowed to exceed the budget, B_i, of object i. The column enumeration counterpart of (19) is

$$\sum_{cr|I_c(i)=1} A_{ir}\delta_{cr} \circledast B_i; \quad \forall i. \tag{20}$$

A third generalization possibility is to consider (17) and/or (19) simultaneously with (3). This leads to the simultaneous presence of (8) and (18) and/or (20). Obviously, this complicates solving the partitioning model. Note that the subproblems are not affected at all by these generalizations.

4.2 Subproblem modification and generation of columns

Another step is to generalize variables α_{ir} to be positive integer instead of binary. Let us call the new variables z_{ir} and we get

$$\sum_r z_{ir} \circledast B_i; \quad \forall i. \tag{21}$$

In this case, the number of columns to consider increases, as now object i is assigned z_{ir} times to resource r. Hence, we get for each possible value of the variable additional columns. Recognize that the number of additional columns is again exponential in the number of objects in $\mathcal{I}$. More precisely, it is $2^{(n-1)}z_{ir} - 1$, as object i has to be contained in all previous columns 2, 3, up to z_{ir} times. The constraint for the master problem generalizes to

$$\sum_{crj|I_{cij}=1} j\delta_{cr} \circledast B_i; \quad \forall i, \tag{22}$$

where the indicator function I_{cij} is defined as

$$I_{cij} = \begin{cases} 1, & \text{if } i \text{ is contained exactly } j \text{ times in column } c \\ 0, & \text{otherwise} \end{cases}; \quad \forall c, i, j.$$

The index j can be interpreted as the multiplicity of object i contained in column c.

Similar to (19), we obtain the following generalization

$$\sum_{r} A_{ir} z_{ir} \circledast B_i; \quad \forall i, \tag{23}$$

with $A_{ir} \geq 0$. The master problem reads in this case as

$$\sum_{crj|I_{cij}=1} jA_{ir}\delta_{cr} \circledast B_i; \quad \forall i. \tag{24}$$

5 Cutting circles from several given rectangles

This problem is a subproblem of those described by Josef Kallrath (2008). A set of circles should be packed into a set of given stocked rectangles of known width and length. The objective is to select some of the available rectangles in order to minimize trimloss. In a second step the objects should be arranged in the rectangles in such a way that a remaining rectangle of maximum size is obtained; this is, however, not the focus of this paper. We see in Sect. 5.1 that this cutting/allocation problem does meet the structure of our general framework. It serves us as a basis for numerical experiments.

Before we apply our approach we review some of the relevant literature. The problem falls into the class of two-dimensional cutting or packing problems of regular objects. It comes close to the Dyckhoff (1990) classification 2/V/D/F; i.e., two-dimensional, V = kind of assignment: a selection of objects and all items, D = assortment of large objects: different figures, and F = assortment of small items: few items of different figures. Related problems are strip packing (given length, width to be minimized) and bin packing (given width, length to be minimized) into rectangles.

There are several publications treating the problem of fitting different-sized circles into rectangles of given size. Fraser and George (1994) discuss packing circles of the same size in a container of fixed dimensions. George et al. (1995) formulated a mixed integer non-linear programming problem for packing different-sized pipes into a rectangular container which is equivalent to packing unequal circles into rectangles. They also addressed the problem of how to allocate pipes to various containers in a shipment in order to minimize the number of containers. They developed a number of heuristic procedures including a genetic algorithm for approximately solving this problem. Because of the container-shipping background of their problem, they also discussed the stability of packing solutions in their paper. Stoyan and Yaskov (1998) discussed and developed exact and approximate algorithms to

compute the global optimum of placing either rectangles or circles into a given rectangle. This paper is very much recommended to the reader because it contains useful analytical results and also reviews many results obtained by Russian and Ukrainian researchers among them V. L. Rvachev. Stoyan and Yaskov (2004) extended their approach to strip packing of circles into one rectangle of fixed width and length to be minimized. Huang et al. (2005) developed a greedy algorithm for packing unequal circles into given rectangles. The problem of packing circles into given rectangles has been shown to be $\mathcal{NP}-hard$; cf. Lenstra and Rinnooy Kan (1979). Although already 15 years old and mostly on packing into given rectangles, it is still inspiring to read the invited review article by Dowsland and Dowsland (1992) on packing problems.

5.1 Modeling

Let us formulate the above problem of cutting circles, $i \in \mathcal{I}$ with $|\mathcal{I}| \geq 2$, from a set of rectangles, $r \in \mathcal{R}$, as a MINLP problem. We denote by R_i the radius of circle i, W_r is the width of rectangle r and L_r is its length. The objective is to minimize the total trimloss, represented in (25), while assigning each of the circles to exactly one rectangular plate, see (26). The cutting process is subject to non-overlapping constraints (28) where variable x_i is the center of circle i in two dimensions, i.e., $d = \{1, 2\}$. The center of the circles have to meet the boundary constraints (29), (30), and (31).

$$\min \sum_r y_r \left(L_r W_r - \pi \sum_i \alpha_{ir} R_i^2 \right) \tag{25}$$

$$s.t. \sum_r \alpha_{ir} = 1; \quad \forall i \tag{26}$$

$$y_r \geq \alpha_{ir}; \quad \forall i \tag{27}$$

$$\sum_d \left(x_{id} - x_{jd} \right)^2 \geq \alpha_{ir} \cdot \alpha_{jr} \left(R_i + R_j \right)^2; \quad \forall i, j, r; \quad i \neq j \tag{28}$$

$$\alpha_{ir} \left(x_{id} - R_i \right) \geq 0; \quad \forall i, d, r \tag{29}$$

$$\alpha_{ir} \left(x_{i1} + R_i \right) \leq W_r; \quad \forall i, r \tag{30}$$

$$\alpha_{ir} \left(x_{i2} + R_i \right) \leq L_r; \quad \forall i, r \tag{31}$$

$$x_i \in \mathbb{R}_+^d; \quad \forall i \tag{32}$$

$$y_r \in \{0, 1\}; \quad \forall r \tag{33}$$

$$\alpha_{ir} \in \{0, 1\}; \quad \forall i, r \tag{34}$$

Next, let us define what a column represents in this application. It is a fixed subset of the set of all circles. The master problem is then given by (7)–(10). Now, let us consider the subproblem in this case. Once a column c and a resource r, which is in this case a rectangular plate, are fixed, the subproblem to be solved becomes the following cutting problem.

$$\overline{F}_{cr} = \min \; L_r W_r - \pi \sum_{i \in c} R_i^2 \tag{35}$$

$$s.t. \sum_d \left(x_{id} - x_{jd} \right)^2 \geq (R_i + R_j)^2; \quad \forall i, j \in c; \quad i \neq j \tag{36}$$

$$\left(x_{id} - R_i \right) \geq 0; \quad \forall i \in c, d \tag{37}$$

$$(x_{i1} + R_i) \leq W_r; \quad \forall i \in c \tag{38}$$

$$(x_{i2} + R_i) \leq L_r; \quad \forall i \in c \tag{39}$$

$$x_i \in \mathbb{R}^d_+; \quad \forall i \tag{40}$$

The objective function value is the trimloss for column c assigned to resource plate r and is therefore known. As in the original formulation (25)–(34), we have the non-overlapping constraints (36). Recognize that if column c contains only one circle, then this set of constraints is empty. In addition, there are the boundary constraints for the center of the circles (37), (38), and (39). Notice that the integer variable y for each sub problem is fix to value 1 and hence does not appear in formulation (35)–(40).

As the objective function (35) is not directly dependent on the variables x_i, the above problem becomes a decision problem whether column c can be packed into the rectangular plate r or not. This can be solved, for instance, by fixing the width of the rectangular plate and calculating the minimal length. For this, one can use the solution approach proposed by Kallrath (2008).

With all the discussions above, we are now able to recognize that the cutting problem formulated as constraints (25)–(34) is a nonconvex MINLP and meets the special structure of the proposed approach; i.e., it satisfies the six assumptions Ass. 1–Ass. 6:

- The objective function (25) is a sum over the resources; hence the cost are additive with respect to the resources.
- There are no cost involved if resource r is not used; i.e., variable y_r will be 0 in any optimal solution of problem (25)–(34).
- The assignment decisions are decoupled from the remaining constraints, i.e., they do not appear in the problem formulation (35)–(40),
- If no object is assigned to rectangular plate r, then variable $\alpha_{\cdot r} = \vec{0}$ implying that constraints (28)–(31) are satisfied. By feasibility, also constraint (27) is satisfied.
- The composition requirement is satisfied, as formulation (35)–(40) involves only variables associated with objects of column c.
- The sub problems are nonconvex because of the reverse-convex constraints (36). However, they can be solved to global optimality quickly for a small number of objects, see Kallrath (2008).

We point out that formulation (25)–(34) is stated only for illustration purposes and is not considered to be used as an optimization formulation, because it can be strengthened in many ways.

For this particular cutting application, one can exploit information about the columns and the rectangular plates. First of all, the decision problem for some columns and resources can be decided very quickly; for instance if the area of the circles to pack is larger than the area of the rectangular plate, or if one of the circles is too large to fit into the plate at all. We want to discuss here two of such approaches. In the first approach the analytical solutions for the case of two and three circles discussed in Appendices A.2 and A.3, respectively, are included. In the second approach, we consider some dominance rules which we discuss in the following subsection.

5.2 Dominance rules

Due to the structure of the optimization problem it is possible to derive the following feasibility and infeasibility implications.

1. *Column dominance*: If the column c is feasible for a given rectangle, all sub-columns $c' \subset c$ are feasible as well and there is no need to solve the sub problem for the sub-columns c', as we can calculate the trimloss for this column resource combination without optimization.
2. *Rectangle dominance*: If column c is feasible for a rectangle r_1 with length l and width w it is also feasible for a larger rectangle r_2 with length and width (L, W) with $L \geq l$ and $W \geq w$ $(r_1 \subseteq r_2)$. One could think that the column selection variable δ_{cr_2} can be fixed to zero because the waste associated with r_2 for that column and resource is larger than the waste for r_1, but it is not true. The reason is that each rectangle can be used at most once and an optimal solution could be "forced" to use a rectangle with larger waste.
3. *Implied infeasibility*: If column c is infeasible for a rectangle then each super-column of $c' \supset c$ is also infeasible for this rectangle.
4. *Object dominance: feasibility*: If a column c is feasible and contains object o, then all objects which are "smaller" than o can take the place of o. Hence, the corresponding columns are also feasible. More generally, all combinations of objects, which can be packed without overlap into object o, can take the place of o. The resulting columns are also feasible.
5. *Object dominance: infeasibility*: If a column c is infeasible and contains object o, then all columns containing instead of object o one which is "greater" than o are also infeasible. More generally, if object $\hat{o}$ is not contained in column c, then all combinations of objects from column c which can be packed without overlap into object $\hat{o}$ can be replaced by $\hat{o}$ and the resulting column is also infeasible.
6. *Identical objects*: If the problem instance contains objects which are mathematically identical, then there is no need to consider all possible subsets of them combined with the other objects. It is enough to take only the number of objects into account instead of its numbering. So for m identical objects, you have only m combinations instead of $2^m - 1$. This brakes the symmetry of the variables α_{ir} for identical objects.

5.3 Numerical experiments and results

In this section, we discuss some implementation issues and computational results for the cutting problem introduced above. We start with a short review of software packages, which guarantee to provide global optimal solutions. A brief discussion of the solvers including stochastic methods can be found for instance in a survey paper by Pintér (1996a). A more detailed overview of deterministic methods is given by Liberti (2006, Chap. 8). The αBB method by Floudas et al. is a Branch-And-Bound algorithm using quadratic underestimations, (1995, 2000,1997). GAMS currently supports three global optimization solvers. The Branch-And-Reduce Optimization Navigator (BARON) by Sahinidis et al. (1995, 1996). In addition, there is the Lipschitz Global Optimization (LGO) solver by Pintér, (1996b), which also supports trigonometric functions.The third solver LINDOGlobal is provided by Lindo Systems, Inc. and uses Branch-And-Cut techniques (2004).

Following Kallrath (2004, Chap. 4.3.3) the power set of $\mathcal{I}$ is enumerated by generating the binary representation of k, $k = 1, \ldots, 2^n$ with $n = |\mathcal{I}|$. The problem is modeled in GAMS, version 22.2, and the NLP problems are solved with BARON. The computations are executed on a Pentium Intel Centrino Duo 2.00 GHz; 1.00 GB RAM with Windows XP platform. The structure of the program is as follows. After a column is selected, the decision problem for all rectangles is solved. First, some analytical methods are applied to decide on feasibility. We discuss this in greater detail with the Tables 5 and 6. If none of these methods

Table 1 Rectangle area is uniformly distributed in [1.70, 3.00], [3.00, 6.00]

# cir.	# rect.	# prob.	# feasible	# solved	# res.	waste	CPU (sec.)
8	20	5,100	690	115	3	9.882	27.06
8	50	12,750	2,024	322	3	9.210	66.55
8	100	25,500	4,589	580	2	8.399	207.33
8	250	63,750	12,706	1,104	2	7.466	387.63
8	500	127,500	26,172	1,651	2	7.153	687.59
8	1,000	255,000	51,510	2,242	2	7.050	930.81
8	5,000	1,275,000	250,821	4,499	2	6.704	2,304.08
8	10,000	2,550,000	499,956	5,790	2	6.704	4,152.20

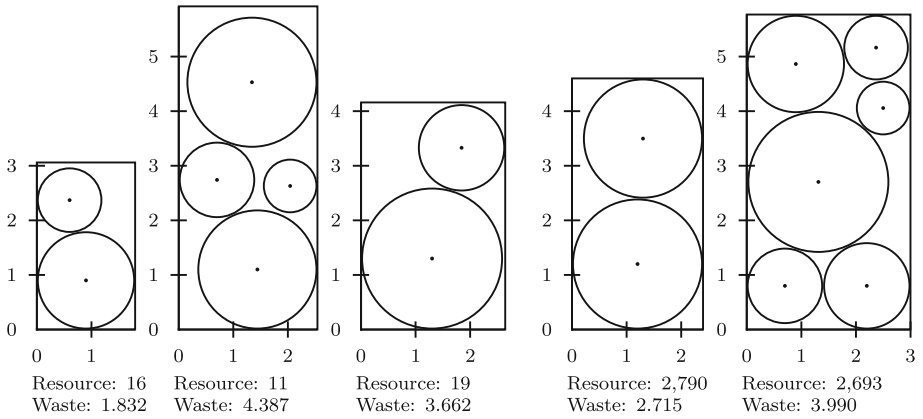
computed a solution, BARON is called to solve problem (35)–(40). After the solution is obtained, several feasibility and infeasibility implications are used to obtain solutions for other column-rectangle combinations.

Consider now Table 1. In this test, the number of circles and its radii are fixed. The number of circles can be found in the first column, # cir., and it is in all cases eight. The radii are 0.5, 0.6, 0.7, 0.8, 0.9, 1.1, 1.2, and 1.3, respectively. For each instance, a set of rectangular plates is randomly generated with a C++ program using the *rand()* function. The width and length of the rectangles are uniformly chosen from the interval [170, 300] and [300, 600], respectively, with a discretization of 0.01. All rectangular plates are different. The number of rectangular plates can be found in the second column labeled with # rect. In the next columns, there are the number of problems to be solved, # prob., the number of feasible problems, i.e., the decision problem (35) – (40) is positive, and the actual number of problems solved with BARON. All other problems could be solved analytically, heuristically or with the dominance rules. The last three columns show the number of resources used in a globally optimal solution, # res., the total trimloss or waste, waste, and finally the CPU time in seconds.

Table 1 shows that the number of NLPs to solve increases linearly while the number of problems increase exponentially as well as the number of feasible problems. Obviously, the number of used resource plates decreases with the increase of the number of plates and with this also the total trimloss. We realize that the CPU time also increases exponentially, which is surprising at the first glance as the number of NLPs being solved increases only linearly. However, checking the solution times for each column-rectangle combination shows that the exponential increase in the running time results from the increase in time spent for each problem solved. Loosely speaking, the implications for other columns and rectangles work better with more rectangles, leaving the more difficult problems to the solver.

Figure 1 shows an optimal solution for two instances considered in Table 1. The case of 20 different rectangular plates is given in Fig. 1a. We clearly see that the length of resource, i.e., rectangular plate, 11 and 16 could be smaller and still fit the circles. The solution for 10,000 rectangular plates, shown in Fig. 1b, gives the reason why the waste for the two instances of 5,000 and 10,000 rectangles of Table 1 are equal. As the solution for the case of 10,000 resources uses only rectangles with number 2,693 and 2,790, there are optimal solutions for these two cases which are identical.

What we could not see in Table 1 is the variation of the CPU time dependent on the number of rectangular plates. It should be intuitively clear that the variation of the CPU time should decrease when increasing the number of rectangular plates. Numerical tests confirm this. The differences in the running times are extremely large for the case of 20 rectangular plates. In order to be able to interpret the following computational results meaningfully, we



(a) 20 rectangles **(b)** 10,000 rectangles

Fig. 1 Optimal solution for different numbers of rectangular plates

Table 2 250 rectangles with different random seed number

# cir.	# rect.	seed	# feasible	# solved	# res.	waste	CPU (s)
8	250	0	12,706	1,104	2	7.466	387.63
8	250	48,632	11,951	1,054	2	7.462	434.38
8	250	54,284	13,019	1,134	2	7.830	471.53
8	250	72,356	10,804	970	2	7.685	333.47
8	250	98,623	11,520	1,040	2	7.921	417.63
8	250	287,614	11,812	1,009	2	7.551	358.58
8	250	356,218	11,913	1,074	2	7.891	389.41
8	250	7,893,492	12,927	1,108	2	7.685	479.77
8	250	17,234,506	12,847	1,148	2	6.919	405.05
8	250	23,678,257	12,666	1,058	2	7.715	321.81

The rectangle area is uniformly distributed in [1.70, 3.00], [3.00, 6.00]

need a more or less "stable" CPU running time for different problem instances. Therefore consider Table 2. The number of different rectangular plates is fixed to 250 and different rectangular plates are generated with the help of different seed numbers for the random generator provided by C++. Recognize that the computational results of row one in Table 2 and of row four in Table 1 are from the same run. Furthermore, we see that the differences in the last five columns are quite small, especially the CPU times are very similar. Therefore, for the following tests, we fix the number of rectangular plates to 250 and consider always, except for Table 3, the same combination provided by the seed number zero, resulting to the values of the first row.

Next, let us look at the dependence of the sizes of the rectangular plates to the computational results. Those are provided in Table 3. Again, we fix the number of circles to eight as well as the number of rectangular plates to 250. We vary only the maximal length of the rectangular plates, shown in column three, max L. From these results we recognize that the number of problems to be solved increases with the increase of the allowed length of the rectangular plates and with this also the CPU time. The reason is that the dominance rules do not work so well when the rectangular plates are more different. In addition, as more circles potentially fit into one particular rectangular plate, the CPU time increases, see also

Table 3 Different rectangle sizes

# cir.	min L	max L	# solved	# res.	area	waste	CPU (s)
8	3.00	3.25	197	4	31.61	9.968	30.99
8	3.00	3.50	258	4	31.21	9.565	40.02
8	3.00	4.00	498	3	30.09	8.443	81.11
8	3.00	4.50	687	3	29.40	7.754	125.24
8	3.00	5.00	848	3	29.41	7.761	223.25
8	3.00	5.50	955	2	29.18	7.537	362.50
8	3.00	6.00	1,104	2	29.11	7.466	387.63
8	3.00	6.50	1,147	2	29.14	7.495	532.06
8	3.00	7.00	1,201	2	28.98	7.333	805.81
8	3.00	7.50	1,213	2	29.13	7.489	1, 166.14

Table 4 Different number of circles

# cir.	# rect.	# prob.	# feasible	# solved	# res.	area	waste	CPU (s)
8	250	63,750	12,706	1,104	2	29.11	7.466	387.63
9	250	127,750	17,635	1,875	3	33.62	8.843	646.28
10	250	255,750	24,460	2,942	3	37.25	9.627	1,237.34
11	250	511,750	30,371	3,877	3	41.71	10.626	1,761.76
12	250	1,023,750	42,882	5,924	4	44.95	11.574	3,606.41
13	250	2,047,750	48,544	7,061	4	50.63	13.122	5,957.99

Kallrath (2008). An indicator that more circles fit into the plates is given in column # res. The column named area gives the total area of the used rectangular plates. We realize that the trimlosses for these circles are quite huge, approximately 30%. For the next test, we fix the upper bound of the rectangular plates to 6.00 and use the rectangles generated for the instance of row seven.

We study now the influence of the number of circles to the computational results. As we use the column enumeration technique as a solution method, the number of circles should greatly influence the running time. As the number of circles increases, the number of columns grows exponentially with base 2. Consider now Table 4. We realize that the number of problems indeed increases exponentially. Interestingly, the number of feasible problems as well as the number of problems to be solved with BARON increases only linearly. However, the CPU time increases exponentially. The reason is that it is computationally more expensive to solve the decision problem (35)–(40) for a larger amount of circles; or more precisely, the running time grows exponentially with the number of circles. Figure 2 shows a global optimal solution for the last instance of Table 4, having 13 circles and 250 rectangles.

Let us now have a look at the impact of the analytical solutions and the dominance rules implemented. Let us mention that it is empirically efficient to go through the columns randomly in order to exploit many dominances. The motivation is that there are some column-rectangle combinations which are computationally "hard" in the sense that they can only be solved by none (or very few) dominance rules. On the other hand, such column-rectangle combinations allow many implications for other columns.

We consider first the feasibility implications in Table 5. For each column and rectangular combination, the algorithm first applies some analytical solutions. It checks two obvious cases: If the column exists only of one circle or if all circles in the column can be aligned next to each other horizontally or vertically. The number of detected feasibilities is listed in

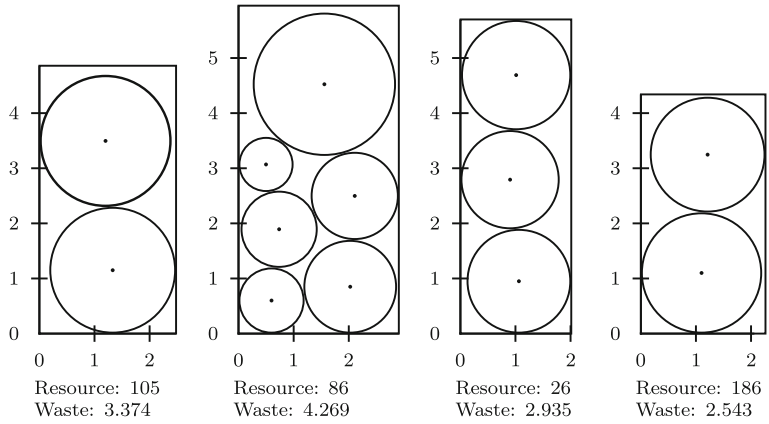


Fig. 2 Optimal solution for an instance with 13 circles and 250 rectangles

Table 5 Feasibility implications

# cir.	# feas.	# solver	# triv.	# circles	# box	# rect.	# sub col.	# cutting
8	12,706	450	41	211	24	1,806	2,594	7,580
9	17,635	715	80	177	16	2,355	2,959	11,333
10	24,460	1,124	93	177	17	2,966	3,474	16,607
11	30,371	1,487	87	301	15	3,241	4,857	20,383
12	42,882	2,257	123	347	16	4,052	4,813	31,274
13	48,544	2,722	152	374	18	4,177	6,961	34,140

the 4th column with label # triv. The next step is only applied when the number of circles in the column is two or three. If this is true, some analytical investigations are exploited, stated in Appendices A.2 and A.3, respectively. The number of detected cuttings with this method is given in column labeled # circles. As a last method before calling the BARON solver, a heuristic to pack the circles is performed. The circles are replaced by squares with length equal to the double radius. In this way, the circles are the incircles of the squares. The resulting boxes are packed heuristically by arranging them in different orders. The number of valid cuttings found is given in column # box. If none of the analytical methods or the heuristic described finds a solution, the NLP problem (35)–(40) is formulated and solved with BARON. Afterwards, the following dominance rules for the feasibility case are implemented. First, the feasibility implication for all other rectangular plates are exploited, see # rect. Then, all sub columns are considered for all rectangular plates (# sub col). And finally, if no other implications apply, then the dominance rule 4 of Sect. 5.2 is used (# cutting). In column three, the number of feasible problems resulting from NLP solutions is given. The computational results stated in Table 5 show that the dominance rules detect a huge fraction of all feasible problems. In particular, the object dominance, provided in the last column, finds more than half of all feasible column-rectangle combinations.

Next, let us have a closer look at the infeasibility implications given in Table 6. We start with checking for the trivial infeasibilities: if the largest circle of the column fits into the rectangle or if the sum of the area of the circles is larger than the area of the rectangle. The number of these infeasibilities is given in column # triv. Next, infeasibilities resulting from the analytical solutions of two circles are tested. They are provided in column # 2 circles. After

Table 6 Infeasibility implications

# cir.	#infeas.	# solver	# triv.	# 2 circles	# rect.	# super col.	# cutting
8	51,044	654	127	111	1,918	23,713	24,521
9	110,115	1,160	211	177	2,888	46,094	59,585
10	231,290	1,816	310	206	4,729	38,446	185,783
11	481,379	2,390	295	232	4,814	104,842	368,806
12	980,868	3,667	421	275	6,884	235,330	734,291
13	1,999,206	4,339	453	350	7,825	102,834	1,883,405

the detected infeasibility from the described methods, or from the solver, the dominance rules are exploited. Afterwards, the implied infeasibilities with respect to rectangular plates for this particular column are tested and the number of detected infeasibilities with this method is provided in column # rect. Next, all super columns and rectangle combinations are checked and the results are listed in column # super col. Last, the infeasibility of object dominance introduced in Sect. 5.2 is used and stated in the last column. From the results given in Table 6, we recognize that the dominance rules detect many infeasibilities and especially the super columns and the object dominance take a large fraction.

6 Conclusions

We have discussed a general approach to decompose a certain class of nonconvex mixed integer programming problems. We started by an assignment structure which we generalized and formulated the column enumeration approach. For a special cutting problem we demonstrated the efficiency of this approach; we assigned circles to up to 10,000 rectangles available on stock.

Future research could focus on applying this solution technique to a variety of other assignment problems. For instance, the cutting of not only circles but in addition rectangles, polygons or convex objects, i.e., ellipses, into given rectangular plates.

Acknowledgements The authors would like to thank Hanif D. Sherali (Virginia Tech) for suggesting the paper of W. E. Wilhelm. In addition, we want to thank two unknown referees for their constructive comments and suggestions. Research was partially supported by NSF and Airforce Grants.

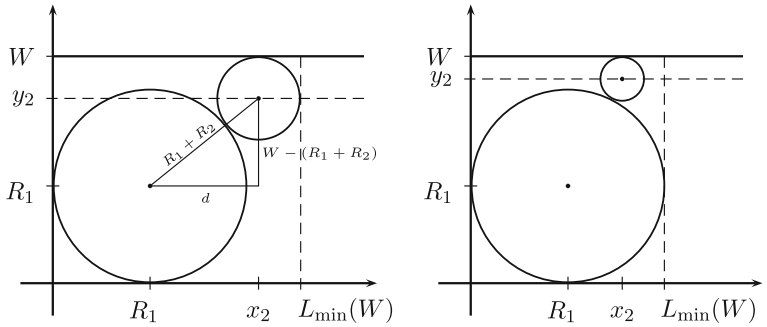
Appendix

A Implications from analytic solutions

In this appendix, we discuss the packing of one, two or three circles. For the case of one or two circles, we give a complete solution for all cases. For the packing of three circles we provide sufficient and necessary conditions for the length and width of the rectangles for some particular cases. For this appendix, let the radii satisfy the condition $R_1 \geq R_2 \geq R_3$.

A.1 One circle

The case of one circle is obvious. Feasibility is possible only for $W \geq 2R_1$ and $L \geq 2R_1$. Thus, the problem is infeasible, if and only if $\min\{W, L\} < 2R_1$.



(a) $2R_1 \le W \le 2(R_1 + R_2)$ **(b)** $W \ge \left(\sqrt{R_1} + \sqrt{R_2}\right)^2$

Fig. 3 Optimum placement of two circles by given width W

A.2 Two circles

A first test is, of course, to check feasibility for the two circles individually as described in Appendix A.1. This implies that the problem is infeasible if $\min\{W, L\} < 2\max\{R_1, R_2\} = 2R_1$.

Let us define the critical radius by

$$R_2^* = R_1\left(\sqrt{2} - 1\right)^2. \tag{41}$$

If $R_2 \le R_2^*$ then the problem is feasible, iff $\min\{W, L\} \ge 2\max\{R_1, R_2\} = 2R_1$. This result comes from a geometric argument, placing the two circles into the square of length $2R_1$.

Theorem A.1 provides sufficient and necessary conditions based on the following question. For given width W, what is the minimum length $L_{\min}(W)$ for obtaining feasibility?

Theorem A.1 *For given width W of a rectangle, the two circles with radii* $R_1 \ge R_2 \ge 0$ *fit into the rectangle without overlap, if and only if*

$$L \ge \begin{cases} 2R_1, & \text{if } W \ge \left(\sqrt{R_1} + \sqrt{R_2}\right)^2 \\ R_1 + R_2 + \sqrt{2W(R_1 + R_2) - W^2}, & \text{if } 2R_1 \le W < \left(\sqrt{R_1} + \sqrt{R_2}\right)^2 \end{cases} \tag{42}$$

Proof Let (x_1, y_1) and (x_2, y_2) be the coordinates of the circles with radius R_1 and R_2, respectively and the rectangle's width side W be parallel to the x-axis; as illustrated in Fig. 3. Without loss of generality, we can assume that $x_1 = y_1 = R_1$. With this, it is optimal to place the smaller circle along the parallel line to the x-axis with $y = W - R_2$ and "move" it until it touches either the large circle or the y-axis. Let us first consider the case when $W \le 2(R_1 + R_2)$ which is illustrated in Fig. 3a. Obviously, we get for the minimum length of the rectangle

$$L_{\min}(W) \ge 2R_1,$$

and in addition the bound

$$L_{\min}(W) \ge R_1 + d + R_2.$$

With the Pythagorean law, we can solve for d and get

$$\begin{aligned} d &= \sqrt{(R_1 + R_2)^2 - (W - (R_1 + R_2))^2} \\ &= \sqrt{2W(R_1 + R_2) - W^2}. \end{aligned}$$

Let us now calculate the critical value W_c for the width W for which both circles fit into the rectangle with length $L = 2R_1$. Such a case is shown in Fig. 3b. This can be done by solving the quadratic equation

$$\sqrt{2W_c(R_1 + R_2) - W_c^2} + R_2 = R_1,$$

which gives

$$W_c = \left(\sqrt{R_1} \pm \sqrt{R_2}\right)^2.$$

As we need that $W \geq 2R_1$ we get

$$W_c = \left(\sqrt{R_1} + \sqrt{R_2}\right)^2. \tag{43}$$

Therefore, the two circles fit in a rectangle with length $L = 2R_1$, if for its width holds $W_c \leq W \leq 2(R_1 + R_2)$. As the case $W > 2(R_1 + R_2)$ is obvious, we get the proposed result. □

From the proof above, we get coordinates for an optimum placement of the two circles dependent on the width W

$$\begin{aligned} (x_1, y_1) &= (R_1, R_1), \\ (x_2, y_2) &= \begin{cases} (R_2, R_1 + 2\sqrt{R_1 R_2}), & \text{if } W \geq 2(R_1 + R_2) \\ (R_1 + \sqrt{2W(R_1 + R_2) - W^2}, W - R_2), & \text{if } 2R_1 \leq W < 2(R_1 + R_2) \end{cases} \end{aligned}$$

According to the problem, the width W of the rectangle is given and we could derive an optimum length $L_{\min}(W)$. Considering W as a variable and minimizing the product $L_{\min}(W) \cdot W$ yields a rectangle with minimum area which contains both circles.

A.3 Three Circles

A first test to check is whether the two largest circles can be packed into the given rectangle of width W and length L using the criteria described in Appendix A.2. If this is not possible, we have proven the infeasibility of packing these three circles into this rectangle. Otherwise, we know $L_{\min}(W)$ for packing the two largest circles from Theorem A.1.

Without loss of generality, we can assume that $W \leq L$. Obviously, $W \geq 2R_1$. Let us label the three circles with C_1, C_2, and C_3. To avoid notational confusion, we distinguish in the following the cases of the minimal rectangular length with respect to circles C_1 and C_2 only and with respect to all three circles. The first case is labeled with $L_{\min,2}(W)$, indicating that only the largest two circles are considered. In the following, we discuss two cases for feasible packings. Case II gives hint how to derive a complete solution theory for the optimal packing of three circles with arbitrary radii.

Case I: All three circles fit into the rectangle with $W = L = 2R_1$.

For this case, we use Eq. 43 with value $W_c = 2R_1$ to derive a critical value R_2^*, resulting in

$$R_2^* = \left(\sqrt{2R_1} - \sqrt{R_1}\right)^2,$$

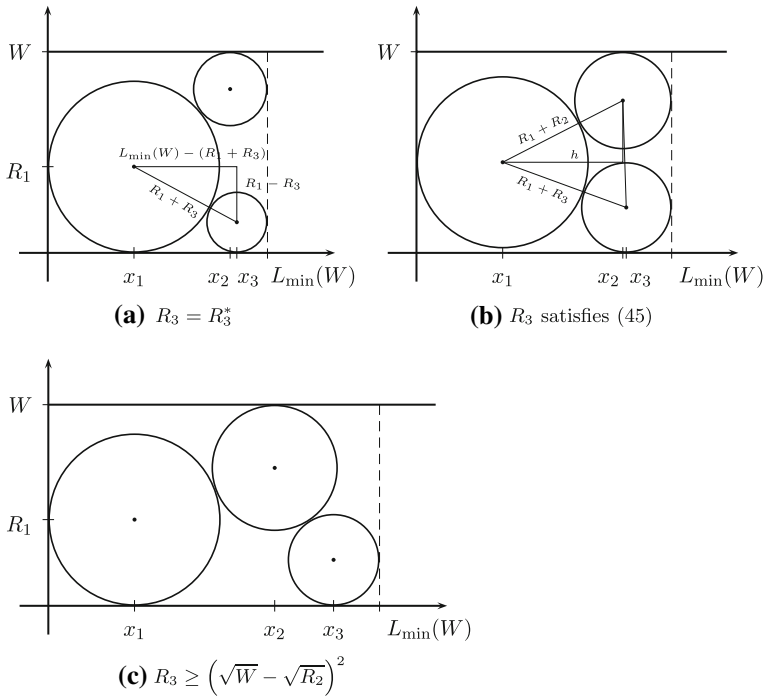


Fig. 4 Optimum placement of three circles satisfying $x_1 \leq x_2 \leq x_3$

which gives the condition

$$R_2 \leq (3 - 2\sqrt{2})R_1$$

for the circle C_2 to fit into this rectangle. The third circle C_3 fits also in this rectangle, as $R_3 \leq R_2$. Coordinates for an optimal packing of these circles are, for instance,

$$\begin{aligned}(x_1, y_1) &= (R_1, R_1),\\ (x_2, y_2) &= (R_2, R_2),\\ (x_3, y_3) &= (R_3, 2R_1 - R_3).\end{aligned}$$

From this point on, we assume $R_2 \geq (3 - 2\sqrt{2})R_1$.

Case II: $x_1 \leq x_2 \leq x_3$.

We divide this case in three sub-cases and provide justification for this approach at the end. These cases are illustrated in Fig. 4.

II.1: Corresponding to Fig. 4a.

Theorem A.1 gives us for the minimal length of the rectangle for a given width W and the two circles C_1 and C_2 the value $L_{\min,2}(W) = R_1 + R_2 + \sqrt{2W(R_1 + R_2) - W^2}$. Now, let us calculate the critical radius R_3^* for C_3 to fit into this rectangle. This implies that $L_{\min}(W) = L_{\min,2}(W)$. As $x_1 \leq x_2 \leq x_3$, there is always the possibility to place, in an optimal packing, circle C_3 as indicated in Fig. 4a. Using the relation given in the figure, we get a quadratic inequality for R_3^* with the solutions

$$R_3^* \gtrless R_1 + L_{\min,2}(W) \pm 2\sqrt{R_1 L_{\min,2}(W)}.$$

As $R_3^* < R_1$, we are only interested in the second solution. In addition, we have to make sure that the two circles C_2 and C_3 can be placed "above" each other. This provides another upper bound

$$R_3^* \leq \left(\sqrt{W} - \sqrt{R_2}\right)^2,$$

where we used once again Eq. 43 for circles C_2 and C_3. The minimum of these two bounds provides, finally, the critical radius

$$R_3^* = \min\left\{R_1 + L_{\min,2}(W) - 2\sqrt{R_1 L_{\min,2}(W)}, \quad \left(\sqrt{W} - \sqrt{R_2}\right)^2\right\}. \tag{44}$$

The coordinates of the three circles are given by, for instance,

$$\begin{aligned}(x_1, y_1) &= (R_1, R_1),\\ (x_2, y_2) &= (R_1 + \sqrt{2W(R_1 + R_2) - W^2}, W - R_2),\\ (x_3, y_3) &= (R_1 + R_2 + \sqrt{2W(R_1 + R_2) - W^2} - R_3, R_3).\end{aligned}$$

II.2: Corresponding to Fig. 4b.
In this case, radius R_3 has to satisfy the bounds

$$R_1 + L_{\min,2}(W) - 2\sqrt{R_1 L_{\min,2}(W)} \leq R_3 \leq \left(\sqrt{W} - \sqrt{R_2}\right)^2. \tag{45}$$

According to Fig. 4b, we derive an expression for

$$L_{\min} = R_1 + h + R_2,$$

which is dependent on h. The value of h is the (positive) solution of

$$(h + R_2 - R_3)^2 + \left(W - R_2 - R_3 - \sqrt{(R_1 + R_2)^2 - h^2}\right)^2 = (R_1 + R_3)^2. \tag{46}$$

Recognize, that Eq. 46 is quadratic in variable h. The coordinates of the circles are, for instance,

$$\begin{aligned}(x_1, y_1) &= (R_1, W - \sqrt{(R_1 + R_2)^2 - h^2} - R_2),\\ (x_2, y_2) &= (L_{\min}(W) - R_2, W - R_2),\\ (x_3, y_3) &= (L_{\min}(W) - R_3, R_3).\end{aligned}$$

II.3: Corresponding to Fig. 4c.
For this case we have that

$$R_3 \geq \left(\sqrt{W} - \sqrt{R_2}\right)^2. \tag{47}$$

This enables us to calculate

$$\begin{aligned}L_{\min} &= R_1 + \sqrt{(R_1 + R_2)^2 - (W - R_1 - R_2)^2} + \sqrt{(R_2 + R_3)^2 - (W - R_2 - R_3)^2} + R_3\\ &= R_1 + \sqrt{W(2R_1 + 2R_2 - W)} + \sqrt{W(2R_2 + 2R_3 - W)} + R_3.\end{aligned}$$

Recognize that $W \leq 2(R_2 + R_3) \leq 2(R_1 + R_2)$, because of inequality (47) and $2\sqrt{R_2}\sqrt{R_3} \leq R_2 + R_3$ holds for all positive R_2 and R_3. For the coordinates of the three circles we get, for instance,

$$(x_1, y_1) = (R_1, R_1),$$
$$(x_2, y_2) = (R_1 + \sqrt{2W(R_1 + R_2) - W^2}, W - R_2),$$
$$(x_3, y_3) = (L_{\min}(W) - R_3, R_3).$$

These three sub-cases provide an optimal packing of three circles with arbitrary radii for the special case that the coordinates for the three circles satisfy $x_1 \leq x_2 \leq x_3$. This is true, as we systematically consider optimal packings for different radii of circle three in the sub-cases discussed above. Using this technique for all six perturbations of the x-coordinates of the circles, one can derive a complete solution for the packing of three circles. Exploiting symmetry arguments helps to reduce the number of sub-cases to be discussed.

References

Adjiman, C.S., Androulakis, I.P., Floudas, C.A.: Global optimization of MINLP problems in process synthesis and design. Comput. Chem. Eng. **21**(Suppl. S), S445–S450 (1997)

Adjiman, C.S., Androulakis, I.P., Floudas, C.A.: Global optimization of mixed-integer nonlinear problems. AICHE J. **46**(9), 1769–1797 (2000)

Androulakis, I.P., Maranas, C.D., Floudas, C.A.: αBB: A global optimization method for general constrained nonconvex problems. J. Glob. Optim. **7**(4), 337–363 (1995)

Dowsland, K.A., Dowsland, W.B.: Packing Problems. Euro. J. Oper. Res. **56**, 2–14 (1992)

Dyckhoff, H.: A Typology of Cutting and Packing Problems. Euro. J. Oper. Res. **44**, 145–159 (1990)

Fraser, H.J., George, J.A.: Integrated Container Loading Software for Pulp and Paper Industry. Euro. J. Oper. Res. **77**, 466–474 (1994)

Floudas, C.A.: Deterministic Global Optimization: Theory, Algorithms and Applications, vol. 37 of Nonconvex Optimization and Its Applications, pp. 309–554. Kluwer Academic Publishers (2000)

George, J.A., George, J.M., Lamar, B.W.: Packing Different-sized Circles into a Rectangular Container. Euro. J. Oper. Res. **84**, 693–712 (1995)

Huang, W.Q., Li, Y., Akeb, H., Li, C.M.: Greedy Algorithms for Packing Unequal Circles into a Rectangular Container. J. Oper. Res. Soc. **56**(5), 539–548 (2005)

Kallrath, J.: Online Storage Systems and Transportation Problems with Applications: Optimization Models and Mathematical Solutions, vol. 91 of Applied Optimization, pp. 92–104. Kluwer Academic Publishers, Norwell, MA (2004)

Kallrath, J.: Cutting circles and polygons from area-minimizing rectangles. J. Glob. Optim (2008). doi: 10.1007/s10898-007-9274-6

Lindo Systems: Lindo API: User's Manual. Lindo Systems, Inc., Chicago (2004)

Lenstra, J.K., Rinnooy Kan, A.H.G.: Complexity of Packing, Covering, and Partitioning Problems. In: Schrijver, A. (ed.) Packing and Covering in Combinatorics, pp. 275–291.Mathematisch Centrum, Amsterdam (1979)

Liberti, L., Maculan, N. (eds.): Global Optimization: From Theory to Implementation, vol. 84 of Nonconvex Optimization and Its Applications, pp. 223–232. Springer (2006)

Pintér, J.D.: Continuous global optimization software: A brief review. Optima, **52**, 1–8 (1996a). See also http://plato.la.asu.edu/gom.html

Pintér J.D. Global Optimization in Action: Continuous and Lipschitz Optimization: Algorithms, Implementations and Applications, vol. 6 of Nonconvex Optimization and Its Applications. Kluwer Academic Publishers (1996b)

Pardalos P.M., Resende M.G.C. (eds.): Handbook of Applied Optimization pp. 337–351. Oxford University Press (2002)

Ryoo, H.S., Sahinidis, N.V.: Global optimization of non-convex NLPs and MINLPs with application in process design. Comput. Chem. Eng. **19**(5), 551–566 (1995)

Ryoo, H.S., Sahinidis, N.V.: A branch-and-reduce approach to global optimization. J. Glo. Optim. **8**(2), 107–138 (1996)

Sahinidis, N.V.: BARON: A general purpose global optimization software package. J. Glob. Optim. **8**(2), 201–205 (1996)

Schrage, L.: Optimization Modeling with LINGO. LINDO Systems, Inc., Chicago, IL (2006)

Stoyan, Y.G., Yaskov, G.N.: Mathematical model and solution method of optimization problem of placement of rectangles and circles taking into account special constraints. Int. Trans. Oper. Res. **5**(1), 45–57 (1998)

Stoyan, Y.G., Yaskov, G.N.: A mathematical model and a solution method for the problem of placing various-sized circles into a strip. Euro. J. Oper. Res. **156**, 590–600 (2004)
Wilhelm, W. E.: A technical review of column generation in integer programming. Optim. Eng. **2**, 159–200 (2001)