INTERNATIONAL TRANSACTIONS IN OPERATIONAL RESEARCH

WILEY

INTERNATIONAL TRANSACTIONS IN OPERATIONAL RESEARCH

Intl. Trans. in Op. Res. 00 (2026) 1–27
DOI: 10.1111/itor.70246

Random-key optimization for 2D irregular packing with reusable area evaluation

Felipe S. C. Roberto[a,*], Victor U. Pugliese[a], Sanderson L. Gonzaga de Oliveira[a], Oseias Ferreira[b], Antonio A. Chaves[a] and Fabio A. Faria[a,c]

[a] *Universidade Federal de São Paulo, São José dos Campos/SP, Brazil*
[b] *EMBRAER S.A., São José dos Campos/SP, Brazil*
[c] *Instituto Superior Técnico, Universidade de Lisboa, Lisboa, Portugal*
E-mail: felipe.silvestre@unifesp.br[Roberto]

Received 30 January 2026; received in revised form 31 July 2026; accepted 2 August 2026

Abstract

The diverse constraints of industrial applications lead to variants of two-dimensional (2D) irregular packing problems that require tailored solution methods. This paper addresses a real-world industrial challenge by proposing a new problem definition, the maximum reusable contiguous area problem (MRCAP), and a novel metric, the maximum contiguous area, to measure and maximize the contiguous unused area in a layout, thereby facilitating the reuse of remnant material. This study proposes an approach focused on optimizing placement policies. We develop a decoder, implemented within a new version of the random-key optimizer (RKO) framework, that dynamically assigns the best placement rule from an 11-heuristic portfolio. We validate our methodology on established literature benchmarks. Among the 15 benchmark 2D Irregular Knapsack Problem instances evaluated, RKO matched the best-performing existing algorithm on 11 and achieved better solutions on the remaining 4. These results suggest that RKO is competitive with, and potentially superior to, this algorithm. RKO's results indicate that current benchmarks no longer adequately represent the problem, motivating the introduction of extended benchmark instances. RKO also outperforms leading 2D irregular strip packing problem (SPP) methods relying on constructive sequence search. Finally, comparing our method against the top SPP algorithm, which minimizes layout overlap, on real-world MRCAP instances shows that RKO yields superior remnant quality in almost all problem cases. This demonstrates that minimizing layout width in SPP does not ensure effective remnant valorization, highlighting the distinction between these two optimization objectives.

Keywords: 2D irregular packing; cutting and packing; irregular knapsack problem; placement policy optimization; nesting optimization; layout optimization; packing heuristics; sustainable manufacturing; material utilization; no-fit polygon

*Corresponding author.

1. Introduction

The two-dimensional (2D) irregular cutting and packing problems are a classical family of $\mathcal{NP}$-hard optimization problems with critical relevance across a wide range of industrial applications, including metallurgical, textile, aerospace, and automotive (Garey and Johnson, 1979; Wäscher et al., 2007; Guo et al., 2022). The main objective of these problems is to minimize material waste when packing small items into containers. However, each industry presents unique constraints that require algorithms to be specifically adapted. This necessity drives the development of new methods to meet these diverse requirements, as addressed in this study in collaboration with our aerospace partner.

Two of the most classical packing problems are the 2D irregular knapsack problem (KP) and the 2D irregular strip packing problem (SPP). Practitioners apply each problem in its respective context. The KP involves a set of items I and a finite cutting area S with specified height H and width W; the objective is to maximize the utilization of S by selecting and packing a subset of items from I. In contrast, the SPP has a similar definition; however, it requires packing all items from I into a cutting area with a single open dimension. Therefore, the container S has a fixed height H, and the objective is to find the minimum width W required to pack all items (Wäscher et al., 2007).

While these problems are practical and widely studied in the literature, real-world applications often require modifications to meet specific constraints. Engineers frequently use composite materials in aerospace and automotive applications because these materials offer a remarkable strength-to-weight ratio. A composite is an engineered material that combines a reinforcement (carbon fiber) with a matrix (epoxy), resulting in a material exhibiting high strength and stiffness (Bandaru et al., 2024). Given the importance of this material, which is expensive to produce, the recycling of its remnants is highly challenging. Even if the material is ground down or melted, it loses its unique structural properties. Therefore, the primary way to reuse the remnants is in other applications, such as creating tools and jigs used on the production line, rather than for their original purpose (Bandaru et al., 2024). However, if a remnant is sufficiently large, it can be re-cut and used for its original objective, even in a manual cutting process. We introduce this real-world problem as the maximum reusable contiguous area problem (MRCAP) evaluated by the maximum contiguous area (MCA) metric.

In the literature, the closest problem to this objective is the SPP, when applied to a large bin in which remnants will remain. One expects that the method of solving SPP organizes the layout to consolidate the unused area on the right side of the bin. However, this result is only an indirect outcome of the SPP's objective and does not consider alternative layout configurations. This limitation implies that even a state-of-the-art algorithm, such as the new advances in the area by Gardeyn et al. (2025), is not optimized for this specific objective when compared with an algorithm designed to optimize the MCA for MRCAP. Figure 1 highlights the objective difference: MRCAP (a) actively searches for the most valuable remnant shape, regardless of its form or location, whereas SPP (b) often yields rectangular remnants on the right side due to width minimization.

This study investigates the limitations of minimizing the layout width as an approach for maximizing remnant valorization. To address this limitation, we employ the random-key optimizer (RKO) framework (Chaves et al., 2025). This method separates the metaheuristic search from the

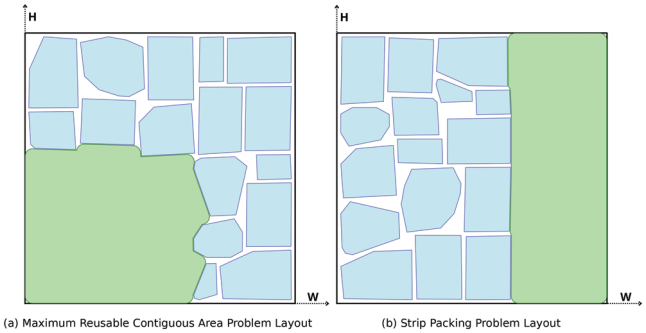

Fig. 1. Conceptual comparison of layout objectives.

problem constraints using a decoder. In this context, the decoder is a deterministic algorithm that translates a vector of random keys into a feasible packing layout. We propose a decoder based on a portfolio of placement rules to solve the classical KP, SPP, and MRCAP.

To bridge this gap, this paper proposes reframing the optimization objective from width minimization to remnant value maximization. The main contributions of this paper are as follows:

1. We formally define the MRCAP, a novel 2D irregular packing definition driven by a critical industrial need: maximizing the value of remnant material for subsequent re-use.
2. We introduce the MCA and its variant, the maximum contiguous convex area (MCCA), as novel metrics for the problem. These metrics quantify the remnant material's quality and usability by measuring the largest contiguous area above a minimum thickness threshold (via morphological opening).
3. We develop a decoder for 2D irregular packing problems that leverages a portfolio of 11 distinct placement heuristics. We validated this decoder on three problem variants: the classical KP, the SPP, and the MRCAP.
4. We introduce and make publicly available a comprehensive industrial dataset, comprising 14 instances for MRCAP and five new real-world instances specifically for KP, along with a proposed extension of classical benchmarks to foster future research.
5. We introduce a new version of the proven architecture of the existing C++ RKO (Chaves et al., 2025). Specifically, we develop a comprehensive Python framework that serves as a highly versatile metaheuristic platform easily adaptable to new combinatorial problems through a custom decoder implementation.

We organize the remaining sections as follows. Section 2 reviews the relevant literature on 2D irregular packing heuristics. Section 3 details the geometric tools and the portfolio of placement rules used in our approach. Section 4 formally defines MRCAP and the proposed metrics. Section 5 describes the RKO framework and the implementation of the specific decoder. Section 6 presents the computational experiments that validate the methodology on classical benchmarks and analyze the industrial case study. Finally, Section 7 offers conclusions and directions for future research.

2. Related work and challenges in two-dimensional irregular packing

The academic effort to solve cutting and packing problems is extensive, dating back to the seminal work of Gilmore and Gomory (1961). For irregular shapes, the pioneering concept of "shape envelope" (Art, 1966) shaped the field. This concept evolved into the formal no-fit polygon (NFP) (Adamowicz and Albano, 1976) and inner-fit polygon (IFP) (Gomes and Oliveira, 2002). These foundational geometric tools underpin the feasible placement regions used in modern constructive algorithms. However, formulating nesting as an exact mathematical model remains a significant challenge due to the logical complexity of non-overlap constraints (Carravilla et al., 2003). Leao et al. (2020) highlighted that, while mixed-integer programming and branch-and-cut approaches continue to advance (Lastra-Díaz and Ortuño, 2024; Cherri et al., 2026), their application in large-scale industrial scenarios is often hindered by high computational costs. Consequently, metaheuristics remain the most practical solution for real-world variants. In this context, we review KP and SPP heuristics to contextualize our contributions.

2.1. Heuristics for the 2D irregular knapsack problem

For years, the 2D irregular KP received less attention than its rectangular counterpart due to its geometric complexity. This situation changed with the work of Del Valle et al. (2012), who introduced a greedy randomized adaptive search procedure (GRASP) based heuristic and benchmarked it on the now-classical EURO special interest group on cutting and packing (ESICUP) instances, establishing a new performance baseline. This approach focuses on the KP involving irregular items, where the objective is to maximize the total packed area by selecting an optimal subset of items. Building on this work, Kierkosz and Łuczak (2019) later improved upon these results with a fast and effective one-pass heuristic that uses a fitting function based on changes in the NFP area to guide item placement, establishing the current state of the art for this specific problem variant.

Following the work of Del Valle et al. (2012), Silveira and Xavier (2022) further explored the constructive approach by proposing a heuristic based on genetic algorithms to optimize the item insertion order, integrating a grouping heuristic that aimed at minimizing the convex hull to create compact layouts. This study also investigated how to integrate the strip packing heuristic as a "compactor" to optimize knapsack solutions. Additionally, Roberto et al. (2025) proposed the greedy column generation (GCG) algorithm, a competitive constructive approach tailored to 2D irregular packing problems. This collective effort has revealed that heuristic methods, especially those that employ geometric properties such as the NFP to determine placement, are fundamental to solving the 2D Irregular KP efficiently.

2.2. Heuristics for the 2D irregular strip packing problem

In the context of the SPP, Elkeran (2013) established a performance benchmark in the literature. His hybrid approach joined Guided Cuckoo Search and pairwise clustering to produce high-quality solutions. More recently, Gardeyn et al. (2025) established a new performance baseline across most benchmark instances with their open-source heuristic, Sparrow. This success highlights the potential for iterative improvement heuristics that operate on infeasible solutions. Solving the

2D irregular SPP has driven the field to develop two distinct algorithmic schools (Elkeran, 2013; Gardeyn et al., 2025).

School 1: Searching over layout (Overlap Minimization). This school currently represents the state-of-the-art (SOTA). Methods in this category start with a complete, often initially infeasible, solution and iteratively apply local modifications to resolve temporary item collisions, effectively converting the optimization problem into a sequence of feasibility problems (Gardeyn et al., 2025). This approach involves: (i) shrinking the container width, (ii) tolerating and penalizing overlaps, and (iii) using penetration depth or alternative geometric metrics to guide the search toward feasibility for the shorter strip (Elkeran, 2013; Imamichi et al., 2009). Algorithms belonging to this dominant school include 2DNest (Egeblad et al., 2007), which employs a fast neighborhood search; iterated local search based on a quasi-Newton method (ILSQN) (Imamichi et al., 2009) and extended local search (ELS) (Leung et al., 2012), based on nonlinear programming; guided cuckoo search (GCS) (Elkeran, 2013), a hybrid approach; raster overlap minimization algorithm (ROMA) (Sato et al., 2019), which utilizes a raster penetration map; and Sparrow (Gardeyn et al., 2025), the current SOTA open-source implementation.

School 2: Searching over the sequence (Constructive). This approach builds feasible solutions incrementally by selecting items based on an insertion order and a placement rule. The main advantage lies in maintaining a valid layout at every step. However, previous research identifies a limitation in this school: a reliance on optimizing the insertion sequence, which often yields suboptimal results for complex instances (Gardeyn et al., 2025). Key methods in this school include TOPOS (Oliveira et al., 2000), the 2-Exchange Heuristic (Gomes and Oliveira, 2002), adaptations such as GCG (Roberto et al., 2025), and evolutionary algorithms focused on sequence optimization (Pinheiro et al., 2016; Amaro Júnior et al., 2017).

This work contributes to School 2 (Constructive) by shifting the optimization focus from the insertion sequence to the placement policy via the proposed decoder. This approach provides the flexibility required to address the KP, SPP, and the proposed MRCAP within a unified framework.

3. Geometric methods

We now focus on the fundamental components required to build valid solutions. This section details the geometric tools used to guarantee feasibility, specifically the NFP and IFP, and introduces the portfolio of placement rules that guides our constructive decoder.

To construct a valid layout, a packing algorithm relies on computational geometry tools to ensure two conditions: the pieces must not overlap, and the cutting area must entirely contain the pieces. This section describes the fundamental methods that form the basis of our decoder, the NFP and the IFP, and how their combination allows us to define the region of feasible placements for any piece.

3.1. Geometric feasibility using no-fit and inner-fit polygons

In irregular packing, the NFP is the central geometric construct for collision avoidance. The NFP between two pieces, A and B, defines the forbidden region for A's reference point relative to B's

position. By placing A's reference point outside of the NFP(A, B), we guarantee that the two pieces do not overlap. One can extend this principle to all pieces already packed.

The NFP is the most widely used method to prevent overlaps. Researchers developed many techniques to compute this important geometric tool. Art (1966) introduced the first method for this objective. Albano and Sapuppo (1980) formalized it and named it NFP. Subsequent work by Bennell and Oliveira (2008) demonstrated how to compute this method. In this work, we first compute all NFPs for every pair of pieces in a pre-processing step. During the execution of the algorithm, we simply translate the corresponding NFP(A, B) to the position of piece B. To compute a single NFP(A, B), we decompose pieces A and B into convex parts, calculate the NFP for each pair of parts using the Minkowski Sum, and then take the union of all resulting partial NFPs, a standard technique for handling non-convex polygons. When the algorithm has already packed more than one piece B_i, and it is necessary to place the piece A, the algorithm translates all relevant NFPs [NFP(A, B_i)] into the positions of each B_i and takes the union of them to define the total forbidden region for A.

The IFP is another tool to ensure feasible layouts. This method describes a relationship between a piece A and the cutting area S. This relation returns a region of space where the algorithm can pack the piece A on the cutting area without extending S. The algorithm meets the condition if the algorithm places the reference point of A inside the IFP(A, S). This technique ensures that all vertices of A will be within the cutting area.

3.2. Placement rules

Both the NFP and IFP techniques are notably relevant to ensure feasible solutions, forming a critical component of algorithms that search exclusively within the viable space. By combining these tools, we can algebraically define the Feasible(A) region (Sato et al., 2012), which is the exact set of all valid reference points for a piece A, as shown in Equation (1). The challenge, then, becomes how to select the "best" point within this potentially complex region.

$$\text{Feasible}(A) = \text{IFP}(A, S) \setminus \bigcup_i \text{NFP}(A, B_i) \tag{1}$$

This difference determines the region of the cutting area where positioning the reference point of piece A guarantees that piece A is within the cutting area and does not overlap with any pieces already packed. Furthermore, it is possible to analyze these positions by considering only the vertices of Feasible(A). It ensures that all candidate positions are feasible and promising, because selecting only the vertices of the feasible region yields positions that either lie on the bin's boundary or are in contact with other pieces, thereby producing a more compact layout. Figure 2 shows how these search spaces are built step by step. Figure 2a highlights the IFP with a green dashed boundary, where the surrounding red area represents invalid regions where item A would fall outside the bin. Figure 2b shows the combined NFP, where the red shaded area marks the forbidden overlap regions where item A would collide with already packed items. Finally, Fig. 2c isolates the final feasible region. This green area maps all positions for the reference point of item A that guarantee the piece remains within the cutting area without overlapping any other pieces.

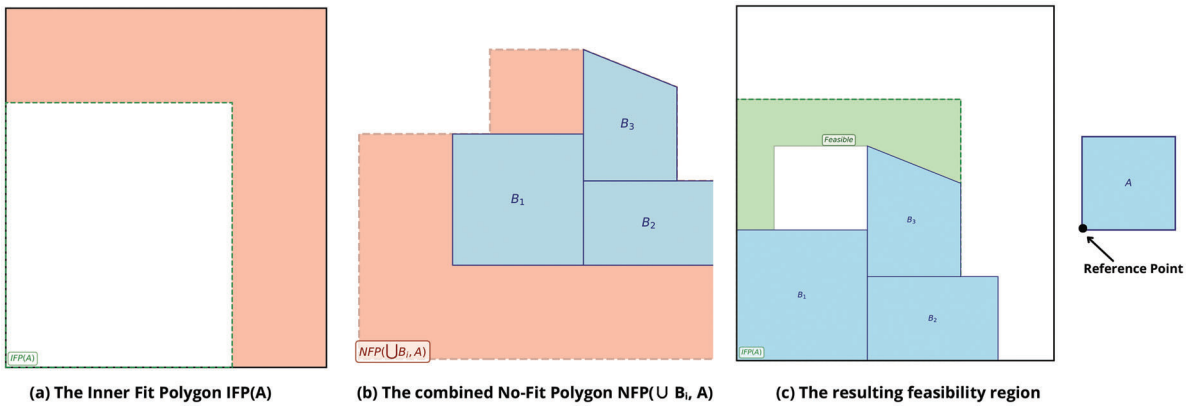

Fig. 2. Construction of the feasible placement region using IFP and NFP.

However, even when considering only the vertices of the feasible region, there can still be many positions, and choosing only one is not a trivial task. In 2D packing problems, it is usual to have a placement rule, such as the prominent bottom-left (BL) heuristic. As researchers designed algorithms, they developed different placement rules to explore diverse layouts. Besides BL, the left-bottom (LB) is another well-known rule; unlike BL, which first finds the bottom-most position and then searches for the left-most, LB does the opposite. Both strategies are employed, for instance, in the decoder proposed by Gonçalves and Resende (2011). In addition to these, Del Valle et al. (2012) used an extended search rule that searches for the position that minimizes the area of the resulting layout's rectangular enclosure to promote compactness. For regular packing, Gonçalves and Resende (2012) created two placement rules that search for positions that maximize the distance to the upper-right corner of the bin. More recently, Chen et al. (2025) advanced this concept in the context of rectangular packing by proposing a corner-occupying placement strategy, which restricts candidate positions to the corners formed by the bin's boundaries and previously packed items. Finally, Pinheiro et al. (2016) used a portfolio of three placement rules in his random-key genetic algorithm, two of which relied on the BL concept and the other on the NFP itself, for irregular packing.

While previous works, such as Gonçalves and Resende (2011) and Pinheiro et al. (2016), have employed small sets of placement rules, these rules stem from different design rationales and lack a unified derivation method. In this paper, we propose a systematic method to derive a portfolio of placement rules directly from the vertices of the Feasible(A) polygon. We organize our portfolio of 11 placement rules into two strategic groups: eight *corner-based rules* designed to build the layout's boundaries, and three *center-based rules* designed to densify the layout by filling internal voids.

We generalize the classic corner-seeking logic to all four corners of the container and complement it with center-based strategies to target internal voids. All 11 rules derive from the same set of candidate points [vertices of Feasible(A)] by varying the sorting criteria. The primary intuition is to provide the algorithm with a balanced yet versatile portfolio, in which all rules are initially equally available for selection. By maintaining this portfolio, the optimizer can autonomously determine the ideal placement strategy for each piece, adapting its choice to the current layout configuration. Table 1 details the portfolio, where corner rules apply a lexicographical sort (primary then

Table 1
Eleven placement rules defined by vertex-sorting criteria

Rule name	Sorting criterion
Corner-based strategies	
Bottom-Left (BL)	Minimize y, then minimize x
Left-Bottom (LB)	Minimize x, then minimize y
Bottom-Right (BR)	Minimize y, then maximize x
Right-Bottom (RB)	Maximize x, then minimize y
Upper-Left (UL)	Maximize y, then minimize x
Left-Upper (LU)	Minimize x, then maximize y
Upper-Right (UR)	Maximize y, then maximize x
Right-Upper (RU)	Maximize x, then maximize y
Center-based strategies	
Nearest to Bin Center (NBC)	Minimize distance to bin geometric center
Nearest to Feasible Center (NFC)	Minimize distance to Feasible Region centroid
Nearest to Layout Centroid (NLC)	Minimize distance to centroid of packed items

(a) Corner-Based Strategies (b) Center-Based Strategies

Fig. 3. Overview of the decoder's placement strategy portfolio.

secondary coordinates) and center rules minimize the Euclidean distance to specific anchor points.

Figure 3 illustrates some rules from our portfolio. Figure 3a shows several corner-based rules. It is possible to see that even similar rules, such as BL and LB, can find different positions because their sorting criteria are processed differently. Figure 3b presents the NFC rule, which searches for free space in the bin, and the NLC rule, which seeks to cluster with the already-packed pieces.

By incorporating this portfolio of 11 distinct heuristics, the RKO gains the capability to dynamically switch between expanding the layout's boundaries and densifying its core. This diverse portfolio is central to the philosophy behind our decoder's design. In traditional sequence-based approaches, the primary optimization task is to find the optimal insertion order of the pieces. These methods often rely on a single fixed placement rule. Our approach reframes this problem. We contend that if the placement strategy is robust and versatile enough to place any piece in an advantageous position regardless of the layout's current state, the insertion sequence becomes less critical. Hence, the optimization task changes. Instead of prioritizing the search for an optimal sequence, the decoder focuses on assigning the best placement rule to each piece within a promis-

ing, pre-sorted sequence. RKO, therefore, is not merely evolving an order. It develops a dynamic assignment policy that maps each piece to the most effective placement strategy from our portfolio to construct dense layouts.

4. The maximum reusable contiguous area problem

We employ the geometric foundation and placement rules established in the previous section to address a specific industrial challenge. This section formally defines the MRCAP and presents novel metrics to quantify the value of the remnant material.

The 2D irregular KP aims to maximize the utilized area without considering the layout's format, whereas the SPP focuses on minimizing width. To the best of our knowledge, neither of these classical formulations directly optimizes the layout to maximize the quality of the remnant material. The SPP is the closest problem in the literature. When applied to a bin where a remnant is known to exist, its objective function tends to position all pieces to the left, resulting in the consolidated unused area as a rectangle on the right side. However, this rectangular remnant is merely a consequence of width minimization, not the objective itself. A consequence of this approach is that the SPP does not directly explore alternative remnant formats, such as a large consolidated block in the center of the bin, since its objective is not aligned with remnant quality maximization. This distinction motivates the development of new metrics designed to guide algorithms in identifying these alternative, high-quality layouts.

Real-world industrial challenges drive this necessity, particularly in the handling of advanced materials such as composites. Unlike metals or polymers, which can be melted down and reformed into new sheets, analysts cannot easily reprocess composite materials without losing their essential structural properties. In this case, the primary method of reuse is a "re-cut" process, which is economically viable if the remnant is contiguous, sufficiently large, and has a usable geometry.

Furthermore, real-world production demand is often volatile. It is common to process cutting orders for several items smaller than the bin area, making a large remnant certain rather than possible. In such scenarios, organizing the layout to ensure that this inevitable remnant is highly usable becomes a primary economic objective.

4.1. Problem definition

We now define the problem. Consider a set I that contains items the method needs to pack. In the classical KP, a bin S has fixed dimensions W and H, and the objective is to maximize area utilization by selecting and packing a subset of I. In the SPP, the bin has a fixed height H but a variable width, and the objective is to pack all items in I while minimizing the required width. The MRCAP differs from both formulations. Given a bin S with fixed dimensions W and H, where S is sufficiently large to contain all items in I, otherwise, no feasible solution exists, the objective is to pack all items such that each item is positioned within its $Feasible$ region, satisfying both non-overlap and containment constraints defined by the NFP and IFP, respectively, while maximizing the contiguous usable remnant R. One obtains the remnant from the geometric difference of the

bin S and the layout L, formed by the packed pieces I. The challenge lies in quantifying the value of the remnant R.

Following the model structure by Gardeyn et al. (2025), we introduce the following notation:

- $i \in I$: an item from the set I to be packed by the method.
- P_i: the polygon representing the shape of item i.
- S: the stationary container (bin) has a fixed width W and height H.
- u_i: a rigid transformation vector (x, y, θ) for item i.
- $\mathcal{F}_i \subseteq \mathbb{R}^2 \times [0, 2\pi)$: set of valid transformations for item i, defined by a position (x, y) and rotation θ.
- $P_i(u_i)$: the region occupied by item i after applying transformation u_i.
- L: the layout defined by the union of all positioned items: $\bigcup_{i \in I} P_i(u_i)$.
- R: the remnant defined as the geometric difference $S \setminus L$.
- $f(R)$: a function quantifying the utility of the remnant (MCA or MCCA).

We can now formulate the MRCAP more formally:

$$\text{maximize} \quad f(R) \tag{2}$$

$$\text{subject to} \quad P_i(u_i) \subseteq S, \qquad \forall i \in I, \tag{3}$$

$$P_i(u_i) \cap P_j(u_j) = \emptyset, \qquad \forall i, j \in I, i \neq j, \tag{4}$$

$$u_i \in \mathcal{F}_i, \qquad \forall i \in I. \tag{5}$$

Constraint (3) requires that every item lies entirely within the bin S, while Constraint (4) guarantees that no two items overlap. Constraint (5) ensures that the placement respects instance-specific limitations, such as allowable discrete rotations. By enforcing these conditions for all $i \in I$, the method packs the entire input set within the bin S without overlap.

To situate MRCAP within the standard literature, we refer to the typology of cutting and packing problems proposed by Wäscher et al. (2007). One can classify SPP as an *open dimension problem (ODP)* due to its variable-width objective, whereas KP falls under the *single large object placement problem (SLOPP)* category, characterized by item selection. The MRCAP shares the *input minimization* requirement of the ODP, as the method must pack all items while operating within the fixed-dimension boundaries of an SLOPP. Consequently, according to Wäscher's criteria, MRCAP constitutes an *input minimization problem on a single large object*. Unlike the classical ODP, which minimizes the utilized width, the MRCAP objective is to determine a packing pattern that maximizes the contiguous area (MCA) of the fixed remnant R.

4.2. Remnant quality metrics

This section introduces the metrics developed to quantify the utility of the residual space. The total remnant area is constant for any valid packing of the set of items. Therefore, maximizing usability requires analyzing the geometry of the remnant rather than its magnitude. We introduce two metrics to guide this optimization: the MCA and the MCCA.

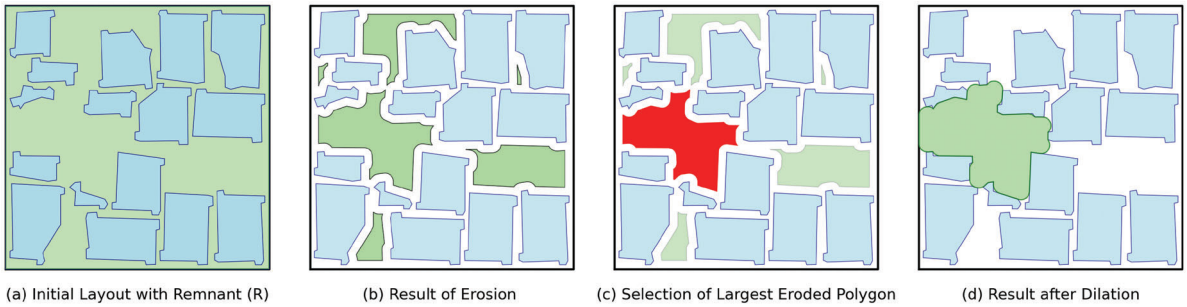

Fig. 4. Overview of the morphological opening process.

4.2.1. *The maximum contiguous area*

While one can easily calculate the total remnant area, computing the usable contiguous remnant is not a simple task, especially considering industrial constraints. In our real-world experiments, a key constraint requires that the pieces have a security margin (m) between them to prevent potential cutting mistakes. This margin ensures the physical separation of pieces, making the entire remnant region R inherently contiguous. However, not all of this contiguous area is practically usable. To filter out thin regions that arise solely from the security margin, the approach applies the morphological opening with a minimum thickness parameter e. In our experiments, we adopt the heuristic $e = 2m$, which ensures that the gap between two adjacent pieces, each separated by a margin m, is never considered usable area. This approach retains only regions that are substantially larger than the operational margin as candidate remnants.

Let $Opening(R, e)$ denote the result of applying a morphological opening process to the initial remnant region R. This process involves first using an erosion operation with distance e to R, which shrinks the remnant polygons and eliminates parts thinner than e, potentially creating multiple disjoint polygons. Then, a dilation operation with distance e is applied, yielding a reconstruction of the substantial parts that closely matches their original dimensions. This sequence effectively filters the remnant R to retain only areas with a minimum thickness of e. Figure 4 illustrates this process.

Let $MCA_{abs}(R)$ be the area of the largest single polygon within the resulting set $Opening(R, e)$. Since the morphological opening may produce multiple disjoint polygons, the method iterates over all resulting components and selects the one with the largest area. By selecting only the largest polygon, the metric incentivizes the algorithm to consolidate the unused area into a single contiguous region rather than fragmenting it across the bin. This design choice addresses a key characteristic of the problem: the future demand that future cutting operations will extract from this remnant is unknown at the time of the original cutting operation. A single large contiguous area accommodates more future item configurations than the same total area split into smaller disjoint regions, as each fragment, though above a minimum area threshold, has reduced dimensions that restrict the size of cuttable items. If instead the metric considered the total area of all qualifying polygons, the optimizer could distribute items across multiple disjoint material islands, thereby reducing the available options for subsequent reprocessing operations. Maximizing the largest polygon thus preserves the continuity of the unused area, providing a larger contiguous region available for future use. The $MCA_{abs}(R)$ represents the absolute usable contiguous area.

While $MCA_{abs}(R)$ quantifies the absolute size, to provide a more intuitive measure of remnant consolidation, the metric the optimizer aims to maximize is the MCA, defined as the ratio presented in Equation (6):

$$\text{MCA}(R) = \frac{MCA_{abs}(R)}{\text{Area}(R)}. \tag{6}$$

This ratio represents the proportion of the total unused area, Area(R), consolidated into the largest usable contiguous block. For a given MRCAP instance, the total remnant area Area(R) is constant. The reason is that the problem determines a fixed bin S, and the layout L must contain all specified pieces, making Area(L) constant as well[Area(R) = Area(S) − Area(L)]. Therefore, maximizing the ratio MCA(R) in Equation (6) is mathematically equivalent to maximizing the absolute usable area $MCA_{abs}(R)$, since the denominator Area(R) is constant for the instance.

4.2.2. The maximum contiguous convex area

The MCA metric identifies a contiguous region with thickness e; however, it does not characterize the geometric shape of this area. If, within the context of the cutting process, the analyst can easily reuse any area larger than e, the MCA is the appropriate metric. However, if the area must be contiguous and the shape must resemble a convex polygon, we developed another metric, the MCCA.

To encourage the area to be more convex, we adapted the MCA metric to guide the optimizer's search towards more convex areas. To compute the MCCA, the process begins similarly, applying a morphological opening to the remnant polygon R with parameter e to obtain the usable area $MCA_{abs}(R)$. However, we now penalize this area based on the extent to which the $MCA_{abs}(R)$ represents its own convex hull. The convex hull of a polygon is the smallest convex polygon that includes all the vertices of the original polygon. If the ratio of $MCA_{abs}(R)$ to its convex hull area is close to 1, this ratio implies that the format is similar to a convex polygon. It forces the metric, and thus the optimizer, to search for large, convex, contiguous remnants. Equation (7) presents the MCCA:

$$\text{MCCA}(R) = \frac{MCA_{abs}(R)}{\text{Area}(R)} \times \frac{MCA_{abs}(R)}{\text{Area}(\text{ConvexHull}(P_{MCA_{abs}}))}, \tag{7}$$

where $P_{MCA_{abs}}$ is the largest single polygon identified within $Opening(R, e)$.

Consequently, the MCCA serves primarily as a guidance metric during the optimization runtime to promote convex shapes. Upon completion of the search, we report the final solution quality using the MCA ratio (Equation (6)). This metric provides an accurate quantitative assessment of the usable remnant's proportion, reflecting the actual amount of contiguous material available for reuse.

Figure 5 illustrates the difference between the metric objectives. MCA aims to maximize the size of any remnant thicker than e. MCCA promotes (via the convexity ratio) more regular, convex shapes. In Fig. 5b, the dashed line represents the convex hull of the largest contiguous area, and the gray regions indicate the difference between the convex hull and the actual remnant geometry. The convexity ratio, defined as the proportion of the convex hull covered by the remnant, is 91.7% in this example, meaning the MCCA penalty is small. Both metrics target the same industrial need,

Fig. 5. Conceptual comparison between the MCA and MCCA objectives.

consolidating unused material into a single large region for subsequent reuse, but differ in how they balance area and geometric regularity. The next section discusses the implications of this choice.

4.3. *The MRCAP as a decision-support framework*

A key characteristic of the MRCAP is that the exact future use of the remnant rarely becomes known at the time of the original cutting operation. This lack of information introduces a significant element of demand uncertainty into the packing process (Hadj Salem et al., 2023). The choice between MCA and MCCA depends on the level of knowledge about future demand. If the professional knows that future items are smaller than the thickness threshold e, or that their shapes are compatible with irregular remnant geometries, MCA is the appropriate metric, as it maximizes the available contiguous area. However, when future demand is entirely unknown, MCCA provides a more conservative alternative by favoring convex remnant shapes, which tend to reduce waste for a broader range of possible item configurations. In practice, this choice is context-dependent and ultimately relies on the judgment of the domain expert. To support this decision, the RKO framework maintains a pool of diverse elite solutions, enabling practitioners to compare different layouts and select the one that best matches their operational needs.

This capability has significant economic implications. Composite materials are costly ($\approx$ \$199/m^2) and problematic to recycle without structural loss (Bandaru et al., 2024). A composite 3D structure forms by stacking multiple 2D layers (plies), each of which requires an independent cutting operation on a separate sheet of material. Consequently, the waste, or the savings, from each 2D layout is multiplied across all layers of the structure. For a representative industrial scenario with 20 layers per structure, 2 m^2 layouts, 30% nominal waste, and an annual volume of 1000 structures, the total material cost, accounting for waste, exceeds \$2.3 million per year. Under these parameters, each percentage point of improvement in remnant usability translates to approximately \$23,880 in annual savings. This compounding cost structure motivates the development of metrics and algorithms that directly optimize remnant quality.

5. A random-key optimizer for 2D irregular packing

Having defined the problems and metrics, this section presents the computational framework developed to solve them. We describe the adaptation of RKO to handle the geometric constraints of the KP, SPP, and MRCAP. We organized the following subsections to present the framework's architecture, the specific solution representation based on a portfolio of placement rules, and the constructive decoder that generates feasible layouts.

RKO (Chaves et al., 2025) is a stochastic local search framework that decouples the optimization engine from problem-specific constraints by leveraging a random-key decoder scheme. RKO encodes solutions as vectors of random keys in $[0, 1)^n$, which a problem-specific decoder maps to feasible solutions and evaluates. This standardized representation allows the framework to address diverse optimization problems by simply implementing the corresponding decoder.

Functionally, RKO executes multiple distinct metaheuristics concurrently in parallel, allowing them to exchange information through a shared global pool of elite solutions. Furthermore, to optimize performance during the search, the framework employs online parameter control via *Q*-Learning (Watkins and Dayan, 1992), enabling each metaheuristic to adapt its configuration dynamically at runtime.

5.1. The RKO decoder for 2D irregular packing

This section details the specific decoders implemented for the KP, SPP, and the novel MRCAP, highlighting the framework's adaptability. To adapt the RKO framework to 2D irregular packing problems, the decoder must bridge the gap between the continuous random-key space and the geometric constraints of the physical layout. This subsection details the three essential components of our decoder implementation. First, the solution representation maps random keys to placement rules and rotations. The layout construction process then determines the solution, followed by evaluation of the objective function, which assesses packing quality based on the specific problem variant (KP, SPP, or MRCAP).

5.1.1. Solution representation

The design of the solution representation is a critical first step, as this random-key vector is the input the decoder uses to construct a complex and feasible layout. We designed the encoding to be versatile, supporting all three problems (KP, SPP, and MRCAP) within a unified structure.

One of the most common representations for 2D packing problems is to fix a single placement rule and search for different insertion orders of the pieces. A more complex decoder for these problems involves searching over both a sequence of pieces and a portfolio of placement rules, as presented by Gonçalves and Resende (2011), who optimized the insertion order and which placement rule to use (BL or LB). Pinheiro et al. (2016) employed a similar technique, but their placement rules consisted of three variants of the BL rule. We built the decoders on the hypothesis that if the placement strategy is robust enough to place any piece in an advantageous position (as our 11-rule portfolio is), the insertion order becomes less critical. Therefore, the search focuses on optimizing the placement rule for each piece.

We defined the main structure of our solution representation with this hypothesis. First, the list of pieces is pre-sorted by area, from largest to smallest, fixing the insertion order. Empirical evaluations indicated that adverse sequences, such as ascending area or perimeter, prematurely scatter small items, frequently causing the algorithm to fail in finding feasible layouts. Random orderings also yielded strictly suboptimal results. While co-evolving both the insertion sequence and the placement policy could theoretically expand the solution space, initial tests showed that it delays convergence. We acknowledge that fixing the sequence restricts the set of reachable layouts; however, within the same computational budget, the policy-based search outperformed sequence-based alternatives. Therefore, fixing the sequence allows the metaheuristic to focus its computational budget entirely on policy selection within the time limit.

For a set with n pieces, the decoders for KP and MRCAP receive a vector of $2n$ random keys, while the decoder for SPP receives $2n + 1$. The first n keys determine which placement rule from our portfolio to use for each piece in the sequence. The second n keys define the rotation to use for each piece. All decoders share the same structure; however, SPP requires an additional key. This final key represents a shrink factor that the method applies iteratively to reduce the container's width, aligning the search with the SPP's objective of minimizing width. Given this solution representation, the following section details how the method transforms this vector of random keys into a layout.

5.1.2. Layout construction process

After defining the solution representation, it is necessary to transform this vector of random keys into a layout. This construction process is fundamental and is applied identically across all three problems: KP, SPP, and MRCAP. As established in our representation, the method already fixed the piece order placement (pre-sorted by area). Therefore, the decoder's task is to execute a deterministic process: it iterates through this fixed sequence and for each piece applies the specific placement rule and rotation dictated by the corresponding random keys.

The layout is constructed by iterating through the n pre-sorted pieces. Operating under the solution representation that partitions the $2n$ random-key vector into a first block of n placement rules and a second block of n rotations, the decoder processes these components dynamically during runtime. Specifically, during the ith iteration of the loop, the decoder processes the ith piece in the sequence, using the ith rule key and the $(n + i)$th rotation key to determine its placement.

To transform a continuous random key (a value in [0, 1)) into a discrete choice, we use a simple interval mapping. For placement rules, the method associates the key with a number in the interval [0, 10] (representing our 11 rules). The method divides the [0, 1) range into 11 equal sub-intervals, and the key's value determines which rule is selected. The framework applies the same mapping process to the rotation keys, dividing the interval according to the number of allowable rotations defined for the instance (typically a subset of $\{0^\circ, 90^\circ, 180^\circ, 270^\circ\}$). Computationally, the method simplifies this interval selection: the index is found by multiplying the key by the number of choices and taking the integer part. For the 11 placement rules, the key is multiplied by 11. Since the key belongs to the interval [0, 1), the product is guaranteed to be in the interval [0, 11), ensuring the resulting integer part falls within the valid index range $\{0, 1, \ldots, 10\}$.

Therefore, the complete process for each piece is as follows: first, the decoder determines the piece's assigned rotation and placement rule from the keys. Second, the approach rotates the piece accordingly, and the decoder computes its $Feasible(A)$ region as explained in Equation (1). Third,

it selects the single vertex from this feasible region that satisfies the decoded placement rule (e.g., the vertex with the minimum y and then the minimum x for the "BL" rule). Finally, the vertices in the piece are translated to this chosen position, placing them in the layout. This process is repeated for all n pieces until the configuration is complete.

Finally, the SPP decoder employs an additional key, the shrink factor s, to dynamically minimize the layout's width (W). Before the construction process begins, the method uses this key to calculate a new, smaller target width. To stabilize the search and avoid overly aggressive reduction attempts, we limit the maximum possible reduction to 1% of the incumbent width W. The key $s \in [0, 1)$ scales the magnitude of this reduction, yielding a new target width W_i calculated as $W_i = W - (s \times (W \times 0.01))$. The decoder then attempts to build the layout within this W_i. If the process successfully packs all pieces, this W_i becomes the new incumbent width. If construction fails, the process discards W_i, and W remains unchanged for the next decoding attempt.

5.1.3. Calculation of the objective function

As detailed in the previous sections, the layout construction process for the KP and MRCAP is identical. The critical difference between them lies in how the method evaluates the final layouts. The KP is a subset selection problem that aims to maximize bin utilization, defined as the ratio of the total packed area to the bin area.

The MRCAP, conversely, is not a selection problem and focuses on maximizing the quality of the remnant material. This feature implies a fundamental constraint for MRCAP: the method must pack all items from the set I. This constraint requires the bin to be sufficiently large to contain all items while maximizing the remaining unused capacity. Therefore, while the KP objective is to maximize area utilization using a subset of I, the MRCAP objective is to maximize the remnant quality, using the MCA and MCCA metrics after packing all items in I.

Finally, the objective for the SPP, to minimize the width W, is handled dynamically during the construction phase via the shrink factor key, as detailed in the previous section. With these distinct objective functions, we configured the decoder to transform a vector of random keys into a specific, evaluated solution for each problem.

5.2. Implementation overview

As mentioned, we implemented a Python-based version of RKO, inspired by the C++ version by Chaves et al. (2025). In our implementation, eight metaheuristics are available: simulated annealing, biased random-key genetic algorithm (BRKGA), variable neighborhood search, iterated local search, large neighborhood search, genetic algorithm, particle swarm optimization, and multi-start. This study implemented all of them according to the original logic of the C++ version.

The primary divergence from the original implementation lies in the decoder architecture. Our version embeds the decoder within a problem-specific class, which the main RKO class receives as input. This problem class must define the complete decoder logic via two methods: a decode method that maps a vector of random keys to a problem solution, and a cost method that takes the resulting solution and computes its objective value. The value returned by the cost method then guides RKO. This accessible architecture leverages Python's position as one of the most widely used

programming languages (TIOBE Software, 2025). Its intuitive syntax reduces the barrier to entry, allowing practitioners outside the specific field of optimization to easily implement custom decoders and apply these techniques to real-world challenges. The Python RKO framework is available in this repository: https://github.com/RKO-solver/RKO_Python.

6. Computational experiments and results

This section presents the computational evaluation of the proposed RKO framework and the MRCAP formulation. We organized the experiment into three parts: first, we detail the experimental design and the datasets used, including classical benchmarks and new industrial instances. Second, we validate the methodology on established KP and SPP to assess the performance of the proposed decoder. Finally, we present a comparative analysis of real-world industrial instances, focusing exclusively on contrasting the objectives of the SPP and MRCAP. This comparison evaluates the remnant quality of the SPP's natural outcome, a rectangular block on the right side, resulting from width minimization, against layouts explicitly optimized for remnant consolidation by the proposed MRCAP formulation. In contrast, the proposed approach explores alternative layouts suitable for this specific goal.

6.1. Experimental methodology

Experiments were conducted on an Intel Core i7-9700K (16 GB RAM) using a Python 3.13 implementation (Shapely library). The source code is available at https://github.com/FelipeSilvestre04/MRCAP. We report the mean of 10 runs evaluating area utilization (KP), width minimization (SPP), and MCA (MRCAP).

Table 2 details the two dataset categories. The first comprises 15 classical ESICUP instances used for KP (Del Valle et al., 2012; Kierkosz and Łuczak, 2019) and SPP, as used from 2000 (Oliveira et al., 2000) to 2025 (Gardeyn et al., 2025). To overcome limitations related to subset selection in the KP, we also evaluate an extended version of these instances, as detailed in Section 6.2.1. For the SPP, this study determined initial widths via the GCG algorithm (Roberto et al., 2025). The second set introduces 14 real-world industrial instances. Unlike the classical set, these instances require oversized sheets, necessitating the objective of MRCAP. Additionally, we combined instances from the real-world set to create extended KP benchmarks with higher item counts (Table 4).

6.2. Validation on classical benchmark problems

Before addressing the industrial challenges of MRCAP, we evaluate the proposed decoder architecture against literature baselines. In this subsection, we apply the RKO framework to the KP and the SPP to validate our approach on these classical variants of 2D irregular packing. These experiments aim to assess the decoder's performance relative to SOTA methods and establish the geometric validity required for the subsequent remnant-focused problem.

Table 2
Characteristics of all benchmark instances, divided into classical instances and new real-world industrial instances, where m denotes the number of piece types, and n is the total number of pieces.

Instance	m	n	Height	W_{fixed}	W_{init} (SPP)	Rotations (°)
Classical benchmark instances						
fu	11	12	38.00	34.00	34.00	{0, 90, 180, 270}
jackobs1	25	25	40.00	13.00	11.50	{0, 90, 180, 270}
jackobs2	25	25	70.00	28.20	28.20	{0, 90, 180, 270}
shapes0	4	43	40.00	63.00	63.00	{0}
shapes1	4	43	40.00	59.00	65.00	{0, 180}
shapes2	7	28	15.00	27.30	27.30	{0, 180}
dighe1	16	16	100.00	138.14	102.14	{0}
dighe2	10	10	100.00	134.05	134.05	{0}
albano	8	24	4900.00	10122.63	10122.63	{0, 180}
dagli	10	30	60.00	65.60	60.60	{0, 180}
mao	9	20	2550.00	2058.60	1858.60	{0, 90, 180, 270}
marques	8	24	104.00	83.60	80.60	{0, 90, 180, 270}
shirts	8	99	40.00	63.13	63.13	{0, 180}
swim	10	48	5752.00	6568.00	6568.00	{0, 180}
trousers	17	64	79.00	245.75	245.75	{0, 180}
New real-world industrial instances						
ED-1	19	19	1420.56	1420.56	—	{0, 90, 180, 270}
ED-2	16	16	1423.10	1423.10	—	{0, 90, 180, 270}
ED-3	16	16	1475.33	1475.33	—	{0, 90, 180, 270}
ED-4	16	16	1526.03	1526.03	—	{0, 90, 180, 270}
ED-5	17	17	1386.10	1386.10	—	{0, 90, 180, 270}
ED-6	16	16	1794.85	1794.85	—	{0, 90, 180, 270}
ED-7	18	18	893.40	893.40	—	{0, 90, 180, 270}
ED-8	20	20	1809.96	1809.96	—	{0, 90, 180, 270}
ED-9	19	19	1850.60	1850.60	—	{0, 90, 180, 270}
ED-10	17	17	1745.04	1745.04	—	{0, 90, 180, 270}
ED-11	18	18	1628.73	1628.73	—	{0, 90, 180, 270}
ED-12	17	17	1527.69	1527.69	—	{0, 90, 180, 270}
ED-13	16	16	1499.25	1499.25	—	{0, 90, 180, 270}
ED-14	16	16	1618.86	1618.86	—	{0, 90, 180, 270}

6.2.1. Knapsack problem

Our first experiment evaluates the RKO framework on the 2D irregular KP. The evaluation is twofold: first, we conduct an initial comparison to validate our specific decoder design against traditional structures; second, we compare our validated RKO against the published SOTA.

A core hypothesis of our methodology is that for complex positioning problems, optimizing a rich portfolio of placement rules over a fixed insertion sequence (pre-sorted by area) is more effective than optimizing the insertion sequence over a fixed, simple placement rule. All experiments in this section use the bin dimensions presented in Table 2 (columns W_{fixed} and Height).

To validate this premise, we compared three decoder variants within the same RKO framework:

- D1 (this work): Fixed insertion order (by area), optimizing rotations and the 11 placement rules.

Table 3
Performance comparison for the 2D irregular knapsack problem (area utilization % and time). Bold values indicate the best result for each instance.

	RKO (This work)		One-Pass (Kierkosz and Łuczak, 2019)		GRASP (Del Valle et al., 2012)		Max. solution (All *n* pieces)
Instance	Avg. Util (%)	Time (seconds)	Avg. Util (%)	Time (seconds)	Avg. Util (%)	Time (seconds)	Area %
fu	**83.82**	5.3	**83.82**	1.0	**83.82**	21.8	**83.82**
jackobs1	**75.38**	0.8	**75.38**	21.5	**75.38**	8.3	**75.38**
jackobs2	**68.44**	0.9	**68.44**	27.5	**68.44**	565.7	**68.44**
shapes0	**63.33**	243.4	60.95	0.7	60.16	1552.5	**63.33**
shapes1	**67.63**	293.5	**67.63**	2.7	64.24	3891.6	**67.63**
shapes2	**79.12**	131.7	76.68	1.7	72.89	1048.1	**79.12**
dighe1	**72.40**	1.8	**72.40**	0.4	**72.40**	10.7	**72.40**
dighe2	**74.60**	1.9	70.42	0.1	**74.60**	0.1	**74.60**
albano	**86.06**	231.1	**86.06**	2.8	80.38	961.6	**86.06**
dagli	**77.31**	2.4	**77.31**	3.5	75.86	1106.4	**77.31**
mao	**71.60**	3.5	**71.60**	15.2	**71.60**	120.4	**71.60**
marques	**82.74**	10.2	**82.74**	7.3	**82.74**	217.8	**82.74**
shirts	**85.54**	239.7	84.82	23.3	77.02	14317.1	**85.54**
swim	**67.35**	167.2	**67.35**	178.8	64.27	39781.4	**67.35**
trousers	**88.63**	340.9	**88.63**	74.5	78.66	5796.6	**88.63**
Wins	**15**		11		7		—

- D2 (Gonçalves and Resende, 2011): Optimizing insertion order, rotations, and a choice of two placement rules (BL, LB).
- D3 (Jakobs, 1996): Optimizing insertion order and rotations, using a single fixed BL.

Preliminary experiments demonstrated that our policy-based decoder (D1) outperformed sequence-based variants. Detailed comparative results are available in the Supporting Information provided in the online repository. While all decoders successfully packed all items on the simpler instances, our proposed D1 decoder achieved better area utilization on the most geometrically complex instances (`shapes0`, `shapes1`, `shapes2`, `shirts`, `trousers`). This result strongly validates our hypothesis that the RKO's search is more effective when focused on evolving the placement policy rather than the insertion sequence. Therefore, we use the D1 decoder (hereafter referred to simply as "RKO") for all subsequent experiments.

Having validated our decoder design, we compared its performance against established SOTA. Following the experiments of Del Valle et al. (2012) and Kierkosz and Łuczak (2019), we used the bin dimensions mentioned in Table 2. Table 3 shows this comparison.

Table 3 reveals two key findings. First, our RKO framework establishes new state-of-the-art results for these KP instances, achieving the best-known solution across all 15 problem cases and distinguishing the proposed approach from prior methods (Del Valle et al., 2012; Kierkosz and Łuczak, 2019). Second, RKO is the first study to pack all *n* pieces across all 15 instances. The complete visual layouts of the resulting packing configurations for all instances are available in the online repository. Specifically, RKO matched the best-performing existing algorithm, the one-pass heuristic (Kierkosz and Łuczak, 2019), on 11 instances and obtained better solutions on the remain-

Table 4
Characteristics of the proposed KP benchmark set, modified classical instances, the new real-world instances, and the baseline results obtained by RKO.

Instance	m	n	Width	Height	Rotations (°)	Best %	Avg. %	Avg. Pcs	Avg. Time	Limit
Modified classical instances										
fu	11	12	34.00	28.96	{0, 90, 180, 270}	93.24	92.25	9.00	258.7	600
jackobs1	22	25	13.00	27.41	{0, 90, 180, 270}	92.61	91.86	18.88	329.5	600
jackobs2	22	25	28.20	43.55	{0, 90, 180, 270}	85.17	84.42	16.50	284.9	600
shapes0	4	43	63.00	23.03	{0}	66.72	66.31	23.00	254.0	1200
shapes1	4	43	59.00	24.59	{0, 180}	69.47	68.30	22.38	474.3	1200
shapes2	7	28	27.30	10.79	{0, 180}	82.16	81.73	19.75	562.2	1200
dighe1	16	16	138.14	65.82	{0}	78.45	78.45	11.00	13.9	600
dighe2	10	10	134.05	67.82	{0}	75.04	75.04	8.00	2.8	600
albano	8	24	10122.63	3833.58	{0, 180}	87.64	87.43	14.50	559.9	1200
dagli	10	30	65.60	42.17	{0, 180}	87.76	87.61	19.38	353.9	1200
mao	9	20	2058.60	1659.82	{0, 90, 180, 270}	84.68	84.61	17.00	410.0	1200
marques	8	24	83.60	78.23	{0, 90, 180, 270}	89.85	89.80	16.33	932.6	1200
shirts	8	99	63.13	31.11	{0, 180}	88.51	88.01	74.38	1049.1	1200
swim	10	48	6568.00	3521.79	{0, 180}	75.88	75.50	26.83	1014.3	1200
trousers	17	64	245.75	63.65	{0, 180}	90.91	90.32	44.38	1038.6	1200
New real-world instances										
ED-C1	14	25	1178.33	1571.11	{0, 90, 180, 270}	84.60	82.92	15.60	754.6	1200
ED-C2	13	25	1461.78	1949.04	{0, 90, 180, 270}	86.10	85.25	16.00	728.1	1200
ED-C3	15	50	2280.85	1282.98	{0, 90, 180, 270}	77.75	76.87	30.00	690.5	1200
ED-C4	11	118	3078.33	1923.95	{0, 90, 180, 270}	82.08	81.89	49.78	1056.7	1200
ED-C5	12	96	5826.81	3641.76	{0, 90, 180, 270}	81.84	81.39	63.44	1130.3	1200

ing four problem cases. These results suggest that RKO is at least competitive with, and potentially superior to, the one-pass heuristic on the test set. While this validates the positioning efficacy of our decoder, it also reveals that these benchmarks no longer require subset selection, which is the principal characteristic that differentiates the KP from other packing problems (Wäscher et al., 2007). When Del Valle et al. (2012) originally formed this set, the placement heuristics available at the time could not pack all items, effectively preserving this challenge. However, improvements in positioning algorithms have eliminated this requirement, reducing these instances to pure positioning problems.

To restore the subset selection requirement, we propose an extended set of these classical instances based on a standardized reduction rule: we reduced the container's height until the total item area reached 110% of the bin's area, explicitly verifying that each item remains geometrically feasible within the reduced dimensions. In addition, we introduce five new real-world industrial instances characterized by natural material scarcity. Table 4 presents the characteristics of this reinforced KP dataset and the baseline results obtained by our RKO.

To complement the modified classical benchmarks, we used data from our aeronautical partner to generate instances ($ED - C_1$ to $ED - C_5$). Since the original real-world instances possess MRCAP characteristics, specifically a low item count relative to an oversized bin, they required adaptation to serve as valid KP benchmarks. We combined multiple original orders to create large-scale instances and adjusted the container dimensions so that the total item area amounted to 110%

Table 5
Average area utilization (%) for the 2D irregular SPP. Results are grouped by methodology: School 1 (Overlap Minimization) and School 2 (Constructive). Bold values indicate the best result for each instance.

Instance	School 1 (Overlap Minimization)					School 2 (Constructive)		
	Sparrow (Gardeyn, 25)	ROMA (Sato, 19)	GCS (Elkeran, 13)	CFREFP (Sato, 12)	ELS (Leung, 12)	RKO (This work)	μ-BRKGA (Amaro, 17)	BLF (Burke, 06)
fu	**92.24**	91.95	90.68	89.54	90.00	91.04	88.34	86.90
jackobs1	**89.09**	**89.09**	88.90	88.61	88.35	**89.09**	78.79	82.60
jackobs2	**84.77**	83.56	81.14	80.18	80.97	80.65	77.90	74.80
shapes0	68.60	**68.73**	67.26	65.31	66.85	62.32	63.50	60.50
shapes1	75.69	**75.86**	73.79	71.71	74.24	66.25	65.98	66.50
shapes2	**84.68**	83.02	82.40	80.81	82.55	83.00	—	—
dighe1	—	—	**100.00**	**100.00**	91.61	**100.00**	**100.00**	77.40
dighe2	—	—	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	79.40
albano	89.39	**89.47**	87.47	86.35	87.38	86.86	84.69	84.60
dagli	**89.26**	87.50	87.06	86.88	86.27	86.06	—	—
mao	**86.14**	83.76	82.93	81.14	82.57	83.12	—	—
marques	**90.93**	89.97	89.40	87.71	88.32	88.08	84.31	86.50
shirts	**89.66**	87.62	87.59	87.18	87.20	86.02	—	—
swim	**78.26**	74.29	74.49	70.82	74.10	71.48	70.78	68.40
trousers	**91.73**	90.48	89.02	87.73	88.29	88.75	88.67	88.50
Wins	10	4	2	2	1	3	2	0

of the bin's area. This constraint ensures that these real-world instances retain the subset-selection characteristics of the KP. The corresponding packing figures and the instance data files are accessible in the online repository.

6.2.2. Strip packing problem

Following the validation on KP, we applied the same comparative analysis to the 2D irregular SPP. This step is crucial for validating our decoder's capability in a classical width-minimization context. We again compared our proposed D1 decoder against the D2 decoder proposed by Gonçalves and Resende (2011) and the traditional D3 decoder. Our policy-based approach outperformed sequence-based variants in 13 of 15 instances. The remaining two cases resulted in ties solely because all methods reached the optimal solution. Detailed comparative results are available in the Supporting Information provided in the online repository.

The results from the SPP decoder comparison are conclusive and reinforce our findings from the KP analysis. Our D1 architecture, which optimizes a diverse placement policy, outperforms the traditional sequence-optimizing decoders (D2 and D3). This result supports our core hypothesis: the search is more effective when focused on optimizing the placement policy (i.e., how the method places a piece) rather than primarily depending on the insertion sequence, that is, the rule governing when the algorithm places a piece.

Having validated our decoder's performance, we selected it as our definitive RKO architecture. The next step was to benchmark its performance against established SPP algorithms from the literature. Table 5 reports the average area utilization (%) over 10 runs for each method, computed as

the total item area divided by the used strip area ($H \times W_{best}$).

The comparative analysis of the SPP, presented in Table 5, should be interpreted in the context of the two algorithmic schools discussed in Section 2. As expected, School 1 (Overlap Minimization) dominates, with specialized methods such as Sparrow (Gardeyn et al., 2025) and ROMA (Sato et al., 2019), achieving the highest area utilization. Although our RKO, belonging to School 2 (Constructive), does not match the state-of-the-art methods from School 1, it achieves the best performance among constructive approaches in School 2, outperforming previous methods such as μ-BRKGA (Amaro Júnior et al., 2017).

This improvement stems from shifting the optimization target from the insertion sequence to the placement policy. Crucially, while the specialized algorithms of School 1 tightly target the strip-width minimization objective, RKO's constructive architecture is objective-agnostic: the same decoder framework can optimize for different objectives by simply changing the evaluation function. This flexibility is what enables its application to the MRCAP, where the goal shifts from minimizing layout width to maximizing the contiguous reusable area.

6.3. MRCAP versus SPP on industrial instances

After validating our RKO framework on classical benchmarks, this final experiment analyzes the central hypothesis of our paper: that maximizing the value of the remnant (MRCAP objective) is fundamentally different from minimizing the layout width (SPP objective). This situation implies that if the industrial goal is to obtain a high-value remnant, the algorithm must be structured to solve MRCAP directly.

6.3.1. Experimental setup

To perform this evaluation, we compare the objectives directly. We contrast the RKO's results, tailored to optimize for MRCAP, with those from the state-of-the-art SPP algorithm, Sparrow (Gardeyn et al., 2025). We select Sparrow solely because it defines the performance ceiling for the SPP, not to benchmark RKO against it. We use Sparrow to solve the SPP on our real-world instances and evaluate the resulting layouts from both approaches using our proposed metrics. Since the SPP minimizes width, it naturally consolidates items toward one side of the bin, typically producing a rectangular remnant attached to the opposite border. While this is often a practical and effective outcome, the SPP objective does not incentivize systematic evaluation of alternative remnant configurations that may yield a higher contiguous area.

It raises the methodological question of whether an SPP algorithm could be adapted to optimize for MRCAP. We argue that this adaptation is non-trivial. Algorithms, such as Sparrow, operate by iteratively shrinking the bin, tolerating temporary overlaps, and minimizing penetration depth to guide the search toward feasibility. We define MRCAP in a fixed overcapacity bin such that the total item area is strictly less than the container. In such an environment, achieving zero overlap is straightforward, and the overlap-minimization mechanism loses its driving gradient. Moreover, Sparrow's high-performance geometry engine is deeply coupled to collision detection and overlap quantification, whereas the MCA metric depends on entirely different geometric operations, morphological opening, polygon erosion, and dilation, applied to the negative space. Integrating such operations would require substantial modifications, effectively resulting in a different algorithm.

It is important to acknowledge that the SPP already provides an implicit form of remnant valorization well-suited to its standard use case: contiguous rolls of material, such as textiles or metal coils. In this context, minimizing width directly minimizes material consumption, and the resulting rectangular remnant, which spans the full height of the roll, offers an ideal geometry for subsequent use. This outcome closely aligns with what the MCCA metric values: a large, convex, and geometrically regular remnant. However, when manufacturers supply the material as fixed-dimension sheets of high-value material, as in composite manufacturing, the rectangular remnant on the border is not necessarily the largest achievable contiguous area. In such scenarios, the MRCAP formulation explicitly optimizes for remnant consolidation, exploring alternative configurations that the SPP objective does not incentivize. Although specific adaptations of SPP solvers are conceivable, such as introducing artificial items representing a desired remnant shape, these require prior knowledge of the target geometry and multiple solver executions, whereas MRCAP discovers the remnant configuration autonomously.

Given these considerations, the most rigorous comparison is to evaluate the objectives themselves rather than attempting to retrofit one algorithm to the other. We, therefore, compare the outcomes of the best available SPP solver [Sparrow(SPP)] and our flexible-objective framework configured for MRCAP [RKO(MRCAP)], evaluating both under the industrially relevant metric: the MCA.

6.3.2. Computational results

The economic discussion provides the definitive context for our main experiment. We analyze the layouts generated by Sparrow, optimizing for the SPP, against our RKO framework for the MRCAP objective. This study evaluates both layouts using the MCA ratio (Equation (6)), which quantifies the proportion of unused area consolidated into the largest contiguous region. Table 6 reports results under two configurations. In the MCA columns, RKO optimizes directly for MCA. In the MCCA columns, RKO optimizes for MCCA. Sparrow optimizes for SPP in both cases. This study evaluates all final layouts using MCA, since it directly represents how much of the unused area remains consolidated into a single contiguous region. All values are the average of 10 runs.

Table 6 presents the results of this comparison. The results indicate that the choice of objective function has a stronger impact on remnant value than the choice of algorithm. Although Sparrow (Gardeyn et al., 2025) is currently considered the leading method for the SPP, RKO optimized for the MRCAP objective yielded higher MCA values.

Focusing on the primary MCA metric, the layouts generated under the MRCAP objective exhibited superior remnant quality compared to those constrained by the SPP formulation in 13 out of 14 industrial instances. This finding challenges the assumption that minimizing layout width is a substitute for maximizing remnant value. The SPP objective does not incentivize the optimizer to explore layouts that consolidate the remnant more effectively, as conceptually illustrated in Fig. 1. The magnitude of this difference is not trivial, with RKO(MRCAP) delivering substantial increases in remnant consolidation on several instances, such as ED-4 (+11.50%) and ED-6 (+12.16%). For the single instance, ED-7, the SPP approach yielded a better result (−4.26%). This case highlights that a right-sided rectangular remnant is a good solution, but it is not the only one, and in 13 of 14 cases, it was not the best.

The MCCA metric, favoring convex formats, confirms this. Even under this stricter metric, optimizing for remnant utility improved MCCA in 10 of 14 instances. Thus, simple rectangles are

Table 6
Comparison of MCA ratio (%) on 14 industrial instances using a 1200-second limit. RKO (MRCAP) maximizes the contiguous reusable area (MCA/MCCA), while Sparrow (SPP) minimizes width. Time-to-best reported in seconds. Bold values indicate the best result for each instance.

Metric	MCA					MCCA				
Instance	RKO (MRCAP)	Time (seconds)	Sparrow (SPP)	Time (seconds)	Diff.	RKO (MRCAP)	Time (seconds)	Sparrow (SPP)	Time (seconds)	Diff.
ED-1	**47.03**	721.2	46.78	1185.6	+0.25	**47.37**	709.4	46.78	1185.6	+0.59
ED-2	**51.94**	721.8	46.23	1187.5	+5.71	**47.62**	552.1	46.23	1187.5	+1.39
ED-3	**47.45**	1186.2	44.64	1167.8	+2.81	**48.22**	739.5	44.64	1167.8	+3.58
ED-4	**69.98**	810.2	58.48	1188.5	+11.50	**58.74**	657.6	58.48	1188.5	+0.26
ED-5	**46.03**	504.0	45.07	1195.9	+0.96	**45.64**	535.8	45.07	1195.9	+0.57
ED-6	**85.06**	704.9	72.90	1192.0	+12.16	**73.64**	653.0	72.90	1192.0	+0.74
ED-7	37.95	330.6	**42.21**	1188.2	−4.26	40.64	653.4	**42.21**	1188.2	−1.57
ED-8	**75.69**	990.9	67.39	1195.9	+8.30	**77.57**	797.9	67.39	1195.9	+0.18
ED-9	**78.56**	587.3	69.90	1195.4	+8.66	**70.00**	690.5	69.90	1195.4	+0.10
ED-10	**75.08**	733.1	66.15	1095.0	+8.93	65.03	688.2	**66.15**	1095.0	−1.12
ED-11	**78.52**	655.3	67.25	1156.2	+11.27	**67.90**	781.0	67.25	1156.2	+0.65
ED-12	**74.50**	556.9	64.63	1187.7	+9.87	**64.87**	442.0	64.63	1187.7	+0.24
ED-13	**71.52**	787.9	63.01	1190.9	+8.51	61.53	732.9	**63.01**	1190.9	−1.48
ED-14	**76.48**	709.8	66.58	1196.6	+9.90	66.35	674.1	**66.58**	1196.6	−0.23
Tot.Wins/Diff.	13	—	1	—	+94.57	10	—	4	—	+3.90

insufficient. While humans prefer visual simplicity, automated machines prioritize usable area. Consequently, RKO finds layouts that are larger (MCA) and often retain a usable, convex form (MCCA).

Additionally, an analysis of placement rule usage frequencies provides further insight into the decoder's behavior. Similar to the balanced portfolio utilization observed in the KP and SPP experiments, the MCA optimization distributed the selection among the 11 rules relatively uniformly, with an average usage frequency of 9.1% per rule. A similar pattern appeared in the MCCA optimization, except for the nearest to feasible center (NFC) rule, which the optimizer selected in only 2.2% of placements within high-quality solutions.

This behavior aligns with the geometric constraints imposed by the MCCA objective. Since MCCA penalizes non-convex remnants, placements generated by the NFC rule tend to place pieces near the center of the contiguous unused area, fragmenting the remnant and reducing its convexity. Consequently, the RKO tends to avoid this rule when optimizing for convex remnant structures.

7. Conclusion

This paper addresses the divergence between classical cutting and packing objectives and the requirements of advanced manufacturing applications involving reusable remnant materials. Our results show that minimizing strip width, as in the classical SPP, does not necessarily maximize remnant usability.

To address this gap, we introduce the MRCAP and the MCA metric. MCA defines a morphological opening process that incorporates a minimum material thickness requirement for subsequent reuse steps. We develop an RKO decoder to evaluate this formulation. The proposed decoder utilizes a portfolio of 11 placement rules applied to a fixed pre-sorted sequence. This study evaluated the framework on three problem classes. For the KP, RKO matches the reported performance of the existing best-performing method on 11 instances and surpasses it on the remaining 4. Also, RKO produced complete packing for all 15 classical benchmark instances, motivating the introduction of extended instances that reinstate the need for subset-constrained selection. For the SPP, the proposed approach achieved competitive results while evaluating the effect of placement policy assignment independently from sequence optimization.

The main computational experiments support the proposed formulation. Compared with an SPP-based approach across 14 industrial instances, RKO optimized for MRCAP achieved higher MCA values in 13 cases, with remnant consolidation gains peaking at +12.16% in instance ED-6. The proposed method demonstrates the practical relevance of optimizing remnant usability in industrial cutting processes involving high-value composite materials. The RKO framework and the 14 industrial instances are made publicly available to support future research on remnant-oriented packing problems.

Future work will extend this methodology toward integrated production planning settings. One direction is a multi-objective formulation that jointly determines which items to pack and how to arrange them to maximize the contiguous reusable area. Another extension is the sequential reuse of irregular remnants as input regions for subsequent packing operations under anticipated future demand. This setting would allow the evaluation of remnant reuse across multiple cutting stages under demand uncertainty.

Acknowledgments

The authors acknowledge financial support from CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico) through the PIBIC scholarship awarded to the first author and the Industrial-Academic Doctoral Scholarship awarded to the second author in partnership with Embraer. The third author acknowledges CNPq for financial support with grant 402599/2023-3. The authors also thank the anonymous referees for their careful reading, insightful comments, and suggestions.

The Article Processing Charge for the publication of this research was funded by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) (ROR identifier: 00x0ma614).

References

Adamowicz, M., Albano, A., 1976. Nesting two-dimensional shapes in rectangular modules. *Computer-Aided Design* 8, 1, 27–33.

Albano, A., Sapuppo, G., 1980. Optimal allocation of two-dimensional irregular shapes using heuristic search methods. *IEEE Transactions on Systems, Man, and Cybernetics* 10, 5, 242–248.

Amaro Júnior, B., Pinheiro, P.R., Coelho, P.V., 2017. A parallel biased random-key genetic algorithm with multiple populations applied to irregular strip packing problems. *Mathematical Problems in Engineering* 2017, 1–11.

Art, R.C., Jr., 1966. An approach to the two-dimensional irregular cutting stock problem. Technical Report 36.Y08, IBM Cambridge Scientific Center, Cambridge, MA.

Bandaru, A.K., Anderson, T., Weaver, P.M., 2024. Recycling CF/PEEK offcut waste from laser assisted tape placement: influence of overlaps and gaps. *Composites Part A* 180, 108104.

Bennell, J.A., Oliveira, J.F., 2008. The geometry of nesting problems: a tutorial. *European Journal of Operational Research* 184, 2, 397–415.

Carravilla, M.A., Ribeiro, C., Oliveira, J.F., 2003. Solving nesting problems with non-convex polygons by constraint logic programming. *International Transactions in Operational Research* 10, 6, 651–663.

Chaves, A.A., Resende, M.G.C., Schuetz, M.J.A., Brubaker, J.K., Katzgraber, H.G., de Arruda, E.F., Silva, R.M.A., 2025. A random-key optimizer for combinatorial optimization. *Journal of Heuristics* 31, 32.

Chen, M., Peng, X., Tang, X., 2025. Filtered beam search algorithm for the two-dimensional rectangular packing problem. *International Transactions in Operational Research* 32, 6, 3729–3755.

Cherri, L.H., Cherri, A.C., Silva, E.F., Oliveira, J.F., 2026. A branch-and-cut algorithm for nesting problems with guillotine constraints. *European Journal of Operational Research* 335, 1, 50–66.

Del Valle, A.M., de Queiroz, T.A., Miyazawa, F.K., Xavier, E.C., 2012. Heuristics for two-dimensional knapsack and cutting stock problems with items of irregular shape. *Expert Systems with Applications* 39, 16, 12589–12598.

Egeblad, J., Nielsen, B.K., Odgaard, A., 2007. Fast neighborhood search for two- and three-dimensional nesting problems. *European Journal of Operational Research* 183, 3, 1249–1266.

Elkeran, A., 2013. A new approach for sheet nesting problem using guided cuckoo search and pairwise clustering. *European Journal of Operational Research* 231, 3, 757–769.

Gardeyn, J., Vanden Berghe, G., Wauters, T., 2025. An open-source heuristic to reboot 2D nesting research. Preprint. https://doi.org/10.48550/arXiv.2509.13329.

Garey, M.R., Johnson, D.S., 1979. *Computers and Intractability: A Guide to the Theory of NP-completeness*. W. H. Freeman, New York.

Gilmore, P., Gomory, R.E., 1961. A linear programming approach to the cutting stock problem. *Operations Research* 9, 6, 849–859.

Gomes, A.M., Oliveira, J.F., 2002. A 2-exchange heuristic for nesting problems. *European Journal of Operational Research* 141, 2, 359–370.

Gonçalves, J.F., Resende, M.G.C., 2011. A parallel multi-population genetic algorithm for a constrained two-dimensional orthogonal packing problem. *Journal of Combinatorial Optimization* 22, 2, 180–201.

Gonçalves, J.F., Resende, M.G.C., 2012. A biased random-key genetic algorithm for a 2D and 3D bin packing problem. Technical Report. AT&T Labs Research, Florham Park, NJ.

Guo, B., Zhang, Y., Hu, J., Li, J., Wu, F., Peng, Q., Zhang, Q., 2022. Two-dimensional irregular packing problems: a review. *Frontiers in Mechanical Engineering* 8, 833635.

Hadj Salem, K., Silva, E., Oliveira, J.F., 2023. Cutting and packing problems under uncertainty: literature review and classification framework. *International Transactions in Operational Research* 30, 6, 3329–3360.

Imamichi, T., Yagiura, M., Nagamochi, H., 2009. An iterated local search algorithm based on nonlinear programming for the irregular strip packing problem. *Discrete Optimization* 6, 4, 345–361.

Jakobs, S., 1996. On genetic algorithms for the packing of polygons. *European Journal of Operational Research* 88, 1, 165–181.

Kierkosz, I., Łuczak, M., 2019. A one-pass heuristic for nesting problems. *Operations Research and Decisions* 29, 1, 37–59.

Lastra-Díaz, J.J., Ortuño, M.T., 2024. Mixed-integer programming models for irregular strip packing based on vertical slices and feasibility cuts. *European Journal of Operational Research* 313, 1, 69–91.

Leao, A.S., Silva, E., Oliveira, J.F., 2020. Mathematical models for nesting problems: a structured review. *European Journal of Operational Research* 280, 1, 1–16.

Leung, S.C.H., Lin, Y., Zhang, D., 2012. Extended local search algorithm based on nonlinear programming for two-dimensional irregular strip packing problem. *Computers & Operations Research* 39, 3, 678–686.

Oliveira, J.F., Gomes, A.M., Ferreira, J.S., 2000. TOPOS - a new constructive algorithm for nesting problems. *OR Spektrum* 22, 2, 263–284.

Pinheiro, P.R., Amaro Júnior, B., Saraiva, R.D., 2016. A random-key genetic algorithm for solving the nesting problem. *International Journal of Computer Integrated Manufacturing* 29, 11, 1159–1165.

Roberto, F.S.C., Pugliese, V.U., Gonzaga de Oliveira, S.L., Ferreira, O.F., Faria, F.A., 2025. Greedy column generation: a novel constructive heuristic for 2D knapsack and strip packing. Paper was presented at the Anais do LVII Simpósio Brasileiro de Pesquisa Operacional (SBPO 2025), Gramado, RS.

Sato, A.K., Martins, T.C., Gomes, A.M., Tsuzuki, M.S.G., 2019. Raster penetration map applied to the irregular packing problem. *European Journal of Operational Research* 279, 3, 657–671.

Sato, A.K., Martins, T.C., Tsuzuki, M.S.G., 2012. An algorithm for the strip packing problem using collision free region and exact fitting placement. *Computer-Aided Design* 44, 8, 766–777.

Silveira, T., Xavier, E.C., 2022. Irregular packing problems: heuristic algorithms and evaluation of no-fit polygon generators. Master's thesis, Federal University of Alfenas, Alfenas, Minas Gerais, Brazil. Available online in April 2022. https://www.bcc.unifal-mg.edu.br/~tiago/files/Mestrado__Irregular_Packing_Problems__Heuristic_Algorithms_and_Evaluation_of_NFP_generators.pdf

TIOBE Software, 2025. Tiobe index for January 2025. Accessed: 20 January 2025. https://www.tiobe.com/tiobe-index/

Wäscher, G., Haußner, H., Schumann, H., 2007. An improved typology of cutting and packing problems. *European Journal of Operational Research* 183, 3, 1109–1130.

Watkins, C.J.C.H., Dayan, P., 1992. Technical note: Q-learning. *Machine Learning* 8, 3, 279–292.