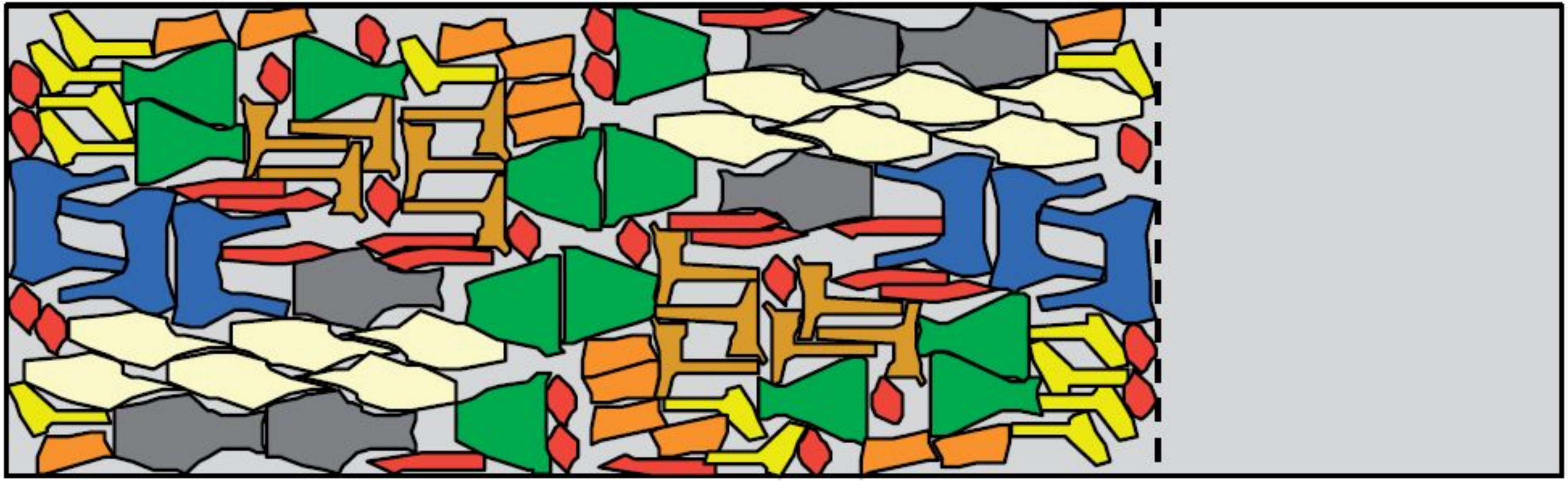
The geometry of nesting problems: A tutorial

1. Intro. RL

Introduction



Problem definition

- Nesting problem
    - where more than one piece of irregular shape must be place in a configuration with the other piece(s) in order to optimise an objective
- Irregular shapes : simple polygons, and in some cases, polygons that may contain holes.
-

Cutting and Packing Problem

- Dyckhoff(1990) : classification of cutting and packing problems
  - four fundamental criteria
    - Dimensionality : one-dimensional, two-dimensional, three-dimensional and n-dimensional
    - objective of the assignment,
      - minimising waste
      - maximising the profit gained from those pieces.
    - large objects : the stock sheets
    - small items : the pieces that are partition

Pixel/raster method

- Raster methods
  - divide the continuous stock sheet into discrete areas
  - reducing the geometric information to coding the data by a grid represented by a matrix
- Oliveira and Ferreira (1993) : 0 (empty space), 1 (existence of piece), greater than 1 (overlap)

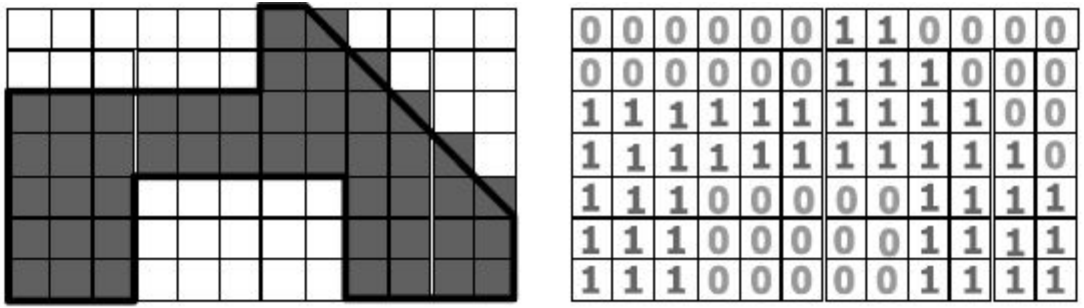


Fig. 2. The 0–1 raster representation for irregular pieces.

Pixel/raster method

- Segenreich and Braga (1986) : 3 (interior)

$$
\begin{bmatrix}
0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0\\
0&1&3&1&0&0&0&0&0\\
1&3&3&3&1&1&1&1&1\\
1&3&3&3&3&3&3&3&1\\
0&1&1&1&1&1&1&3&1\\
0&0&0&0&0&0&1&1&1
\end{bmatrix}
+
\begin{bmatrix}
0&0&0&0&1&1&1&1&1\\
0&0&0&0&1&3&3&3&1\\
0&0&0&0&1&3&3&3&1\\
0&0&0&0&1&3&3&3&1\\
0&0&0&0&1&3&3&3&1\\
0&0&0&0&1&3&3&3&1\\
0&0&0&0&1&3&3&3&1\\
0&0&0&0&1&1&1&1&1
\end{bmatrix}
=
\begin{bmatrix}
0&0&0&0&1&1&1&1&1\\
0&0&0&0&1&3&3&3&1\\
0&0&1&0&1&3&3&3&1\\
0&1&3&1&1&3&3&3&1\\
1&3&3&3&2&4&4&4&2\\
1&3&3&3&4&6&6&6&2\\
0&1&1&1&2&4&4&6&2\\
0&0&0&0&1&1&2&2&2
\end{bmatrix}
$$

Fig. 3. A non-Boolean raster representation for irregular pieces.

Pixel/raster method

- Babu and Babu (2001)

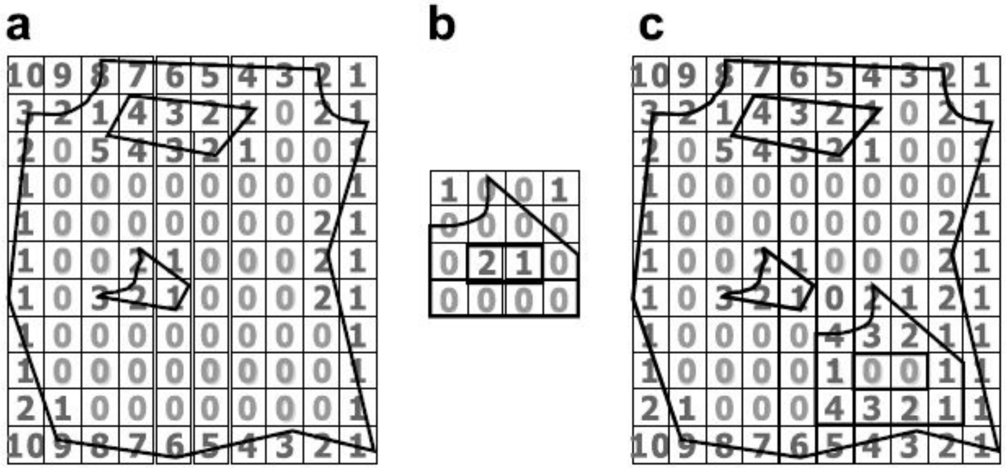


Fig. 4. Raster method proposed by Babu and Babu (2001): (a) stock sheet with defects, (b) piece and (c) stock sheet with a piece placed.

Pixel/raster method

- Advantage : simple
- Disadvantage : memory intensive, inaccurate

Direct trigonometry and th D function

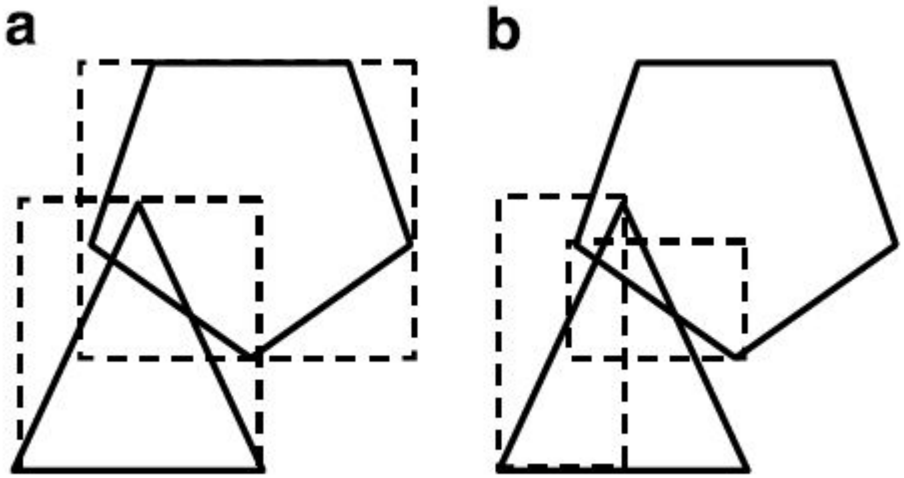


Fig. 5. (a) If two polygons overlap, then the enclosing rectangles of the pieces must overlap. (b) If two edges intersect then the enclosing rectangles of the edges must intersect.

Direct trigonometry and th D function

Test1 : bounding boxes overlap?
 No - a
 Yes - test2
Test2 : edge's bounding box overlap?
 No - b
 Yes - test3
Test3 : edge intersection?
 No for all - test4
 Yes - one pair of edges intersection
  Overlap (c)
Test4 : vertex inside the other polygon
 No for both - do not overlap (e)
 Yes for one - overlap (d)

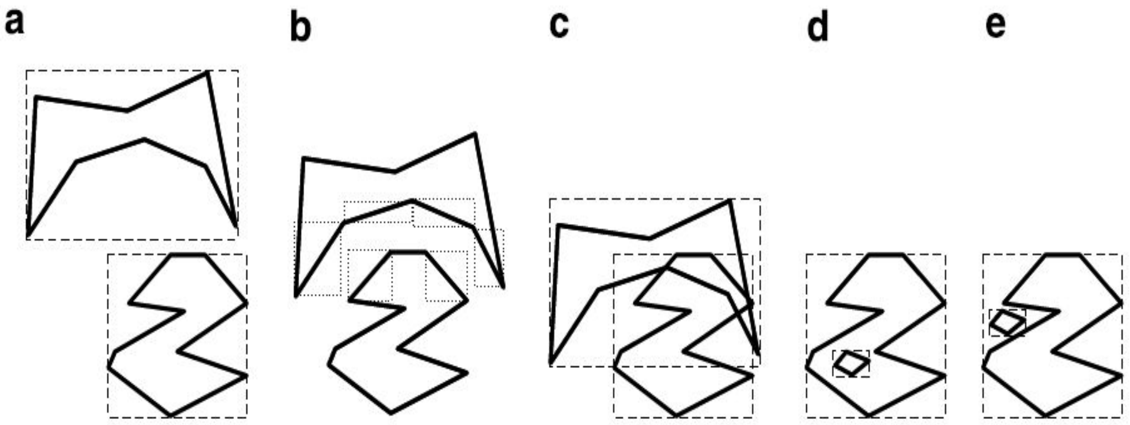
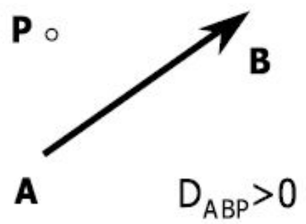


Fig. 6. Examples of the relative position of two polygons identified by tests 1 to 4.

D-function

- Test3 : D-function

$$D_{ABP} = ((X_A - X_B)(Y_A - Y_P) - (Y_A - Y_B)(X_A - X_P))$$



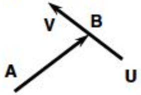
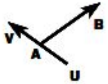
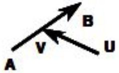
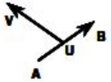
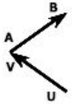
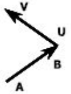
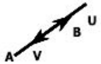
$D_{ABP} > 0$: p is in left side of AB

$D_{ABP} < 0$: p is in right side of AB

$D_{ABP} = 0$: p is on the line of AB

D–function

Table 1
D-functions analysis of the relative position of two oriented edges

a	$D_{ABU} \neq 0 \wedge D_{ABV} \neq 0 \wedge D_{ABU} \neq D_{ABV} \wedge D_{UVA} \neq 0 \wedge D_{UVB} \neq 0 \wedge D_{UVA} \neq D_{UVB}$	
b	$D_{ABU} \neq 0 \wedge D_{ABV} \neq 0 \wedge D_{ABU} \neq D_{ABV} \wedge D_{UVA} \neq 0 \wedge D_{UVB} = 0$ Additionally, if $D_{UVA} < 0$ then A is on the right side of the oriented edge UV and if $D_{UVA} > 0$ then A is on the left side of the oriented edge UV	
c	$D_{ABU} \neq 0 \wedge D_{ABV} \neq 0 \wedge D_{ABU} \neq D_{ABV} \wedge D_{UVA} = 0 \wedge D_{UVB} \neq 0$ Additionally, if $D_{UVB} < 0$ then B is on the right side of the oriented edge UV and if $D_{UVB} > 0$ then B is on the left side of the oriented edge UV	
d	$D_{ABU} \neq 0 \wedge D_{ABV} = 0 \wedge D_{UVA} \neq 0 \wedge D_{UVB} \neq 0 \wedge D_{UVA} \neq D_{UVB}$ Additionally, if $D_{ABU} < 0$ then U is on the right side of the oriented edge AB and if $D_{ABU} > 0$ then U is on the left side of the oriented edge AB	
e	$D_{ABU} = 0 \wedge D_{ABV} \neq 0 \wedge D_{UVA} \neq 0 \wedge D_{UVB} \neq 0 \wedge D_{UVA} \neq D_{UVB}$ Additionally, if $D_{ABV} < 0$ then V is on the right side of the oriented edge AB and if $D_{ABV} > 0$ then V is on the left side of the oriented edge AB	
f	$D_{ABU} \neq 0 \wedge D_{ABV} = 0 \wedge D_{UVA} \neq 0 \wedge D_{UVB} = 0$ Additionally, if $D_{ABU} < 0$ then U is on the right side of the oriented edge AB and if $D_{ABU} > 0$ then U is on the left side of the oriented edge AB	
g	$D_{ABU} \neq 0 \wedge D_{ABV} = 0 \wedge D_{UVA} = 0 \wedge D_{UVB} \neq 0$ Additionally, if $D_{ABU} < 0$ then U is on the right side of the oriented edge AB and if $D_{ABU} > 0$ then U is on the left side of the oriented edge AB	
h	$D_{ABU} = 0 \wedge D_{ABV} \neq 0 \wedge D_{UVA} \neq 0 \wedge D_{UVB} = 0$ Additionally, if $D_{ABV} < 0$ then V is on the right side of the oriented edge AB and if $D_{ABV} > 0$ then V is on the left side of the oriented edge AB	
i	$D_{ABU} = 0 \wedge D_{ABV} \neq 0 \wedge D_{UVA} = 0 \wedge D_{UVB} \neq 0$ Additionally, if $D_{ABV} < 0$ then V is on the right side of the oriented edge AB and if $D_{ABV} > 0$ then V is on the left side of the oriented edge AB	
j	$D_{ABU} = 0 \wedge D_{ABV} = 0$	

NFP : nofit polygon

-

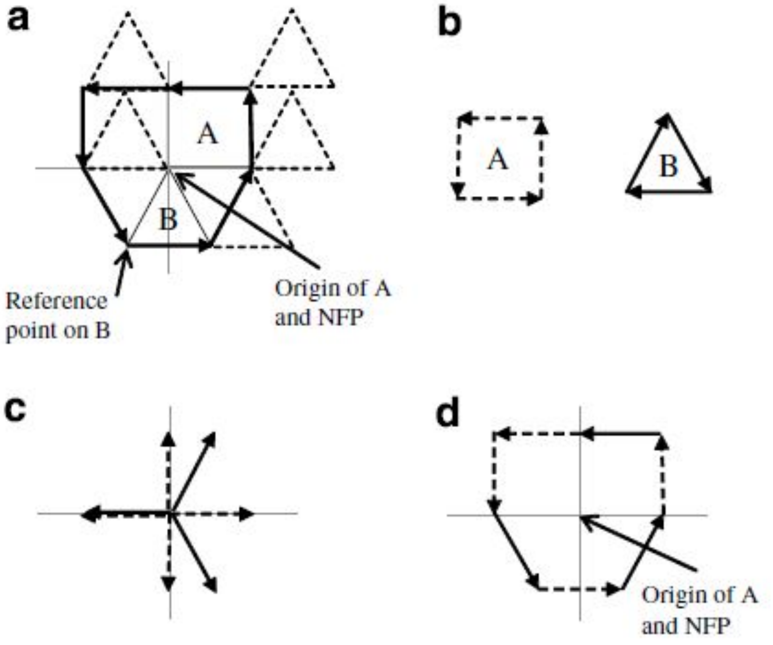


Fig. 9. (a) Tracing movement of polygon *B* around *A* to form the NFP, (b) the orientation of *A* and *B* and (c) the order of the slopes of *A* and *B*, (d) the NFP.

NFP : nofit polygon

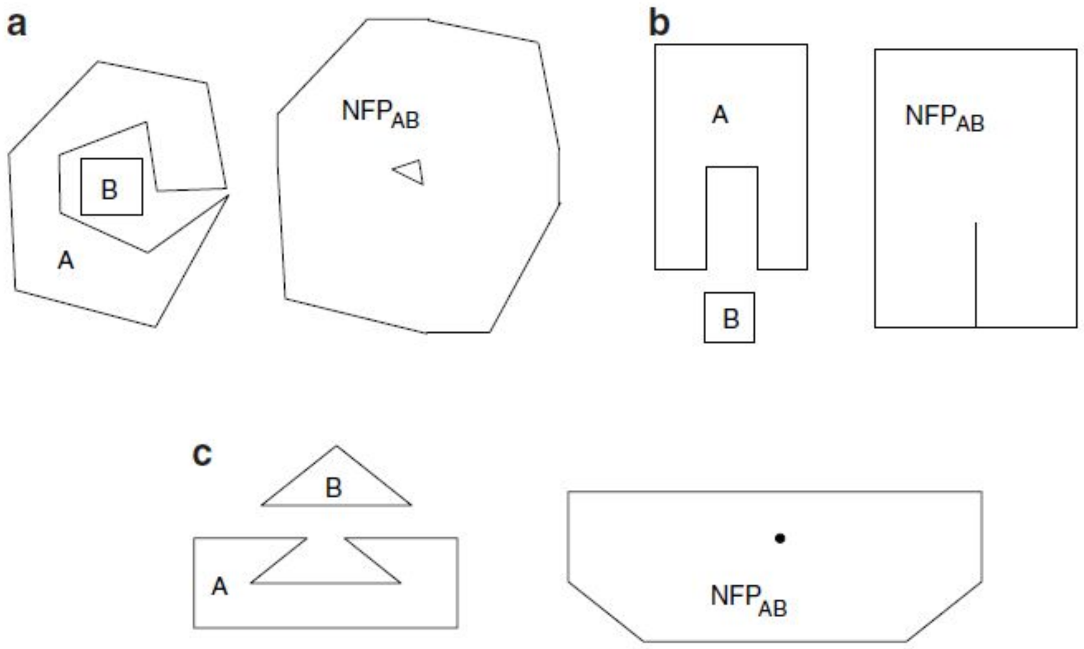


Fig. 10. Combinations of polygons the generate multiply-connected NFPs. (a) NFP where *B* can fit, with space, inside concavity but cannot slide into the concavity. (b) NFP where *B* slides into concavity in one direction only and (c) NFP where *B* can fit at a single point inside concavity but cannot slied into the concavity.

Sliding Algorithm

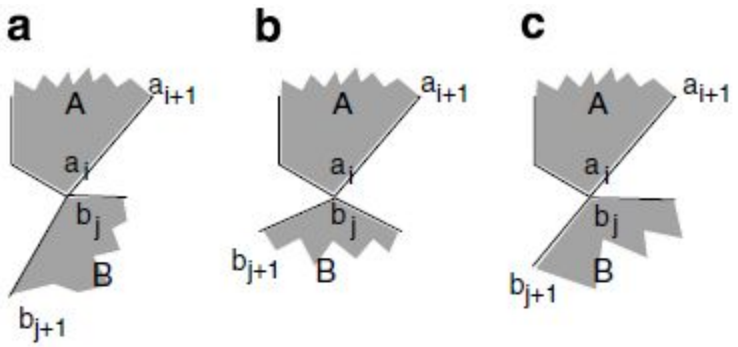


Fig. 11. Scenarios for edge vertex sliding combination.

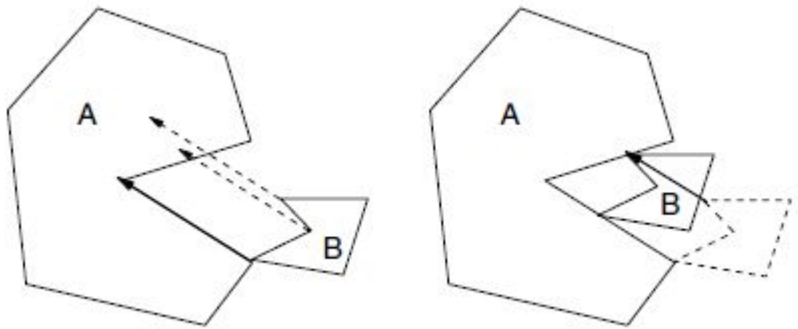
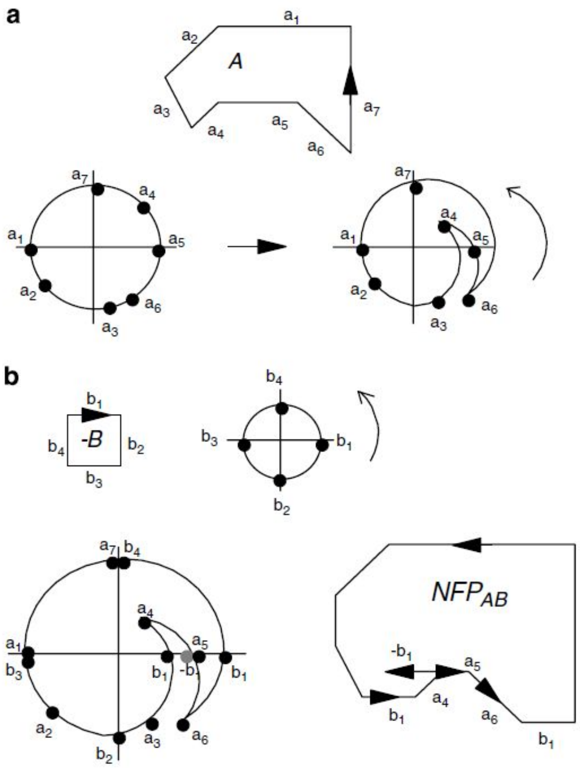


Fig. 12. Sliding edge of *A* projected from vertices of *B* to find minimum intersection distance.

Minkovski sums

$$S = A \oplus B = \{a + b | a \in A, b \in B\}.$$

$$S = A \oplus B = \bigcup_{b \in B} A_b.$$



A reinforcement learning algorithm for the 2D-rectangular strip packing problem

- The 2D packing problem : the cutting and packing problems
  - manufacturing industries
  - many metaheuristics have been proposed and applied on the packing problem
- the approach combined with machine learning serves as a novel paradigm for solving the combinatorial optimization problem
- the machine learning approaches have very limited literature reports on the appliance of the packing problem.
- propose a reinforcement learning method for the 2D-rectangular strip packing problem.
  - the sequence of the items and the layout is constructed piece by piece.
  - use the lowest centroid placement rule for the piece placement
  - a Q-learning based sequence optimization is applied

A reinforcement learning algorithm for the 2D-rectangular strip packing problem

- The 2D rectangular packing problem : the two-dimensional regular type of the C&P problem
  - NP-hard combinatorial optimization problems
    - exact algorithms can provide optimal solutions
    - take huge amount of computational effort.
  - Since meta-heuristics are able to produce satisfactory solutions within reasonable time, over these years, many meta-heuristics have been applied to solve this problem [12],
    - tabu search,
    - simulated annealing
    - genetic algorithms.

A reinforcement learning algorithm for the 2D-rectangular strip packing problem

- Problem description
  - Minimize L’

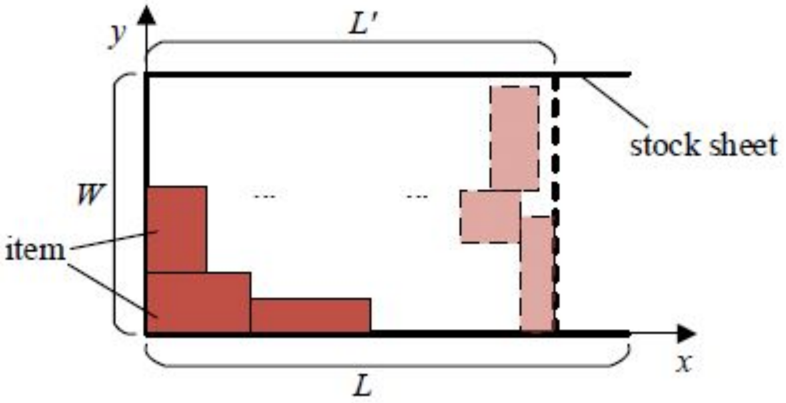


Figure 1. Rectangular strip packing problem

A reinforcement learning algorithm for the 2D-rectangular strip packing problem

- Algorithm
  - Lowest centroid placement rule

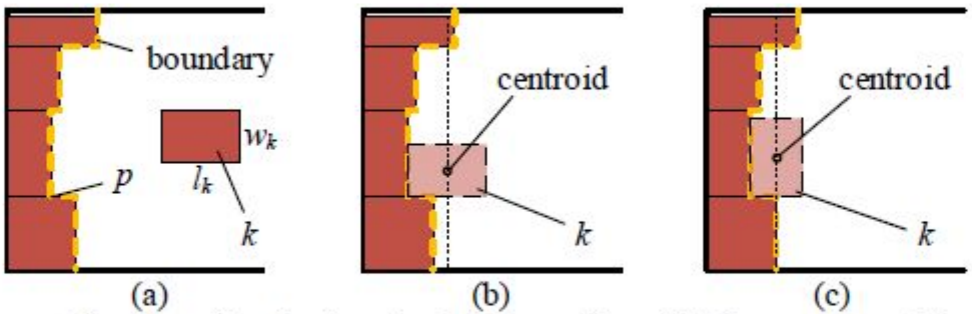


Figure 2. Placement decision of each piece during the packing. (a) The current packing state, the piece to be placed is denoted by k. (b) Piece k takes a rotation of 0°. (c) Piece k takes a rotation of 90°.

A reinforcement learning algorithm for the 2D-rectangular strip packing problem

- Algorithm
  - Q-learning for sequence optimization

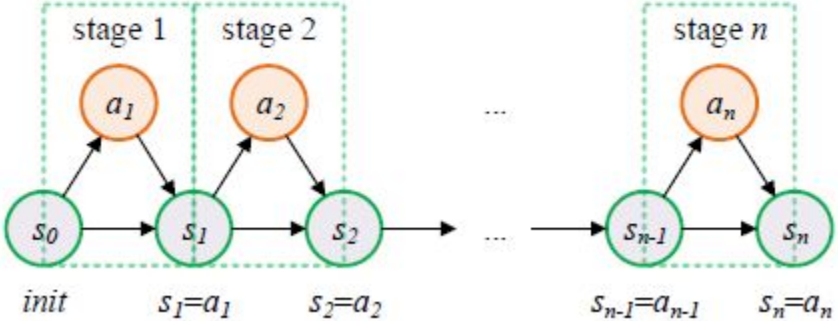


Figure 3. Markov Decision Process for the n stages packing

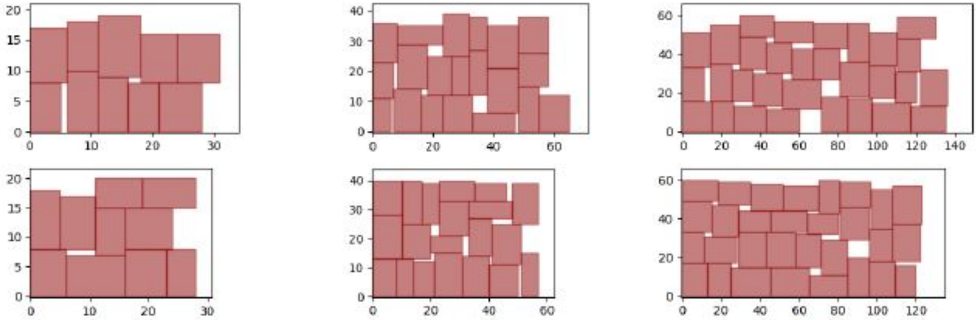
Algorithm 1. Q-learning for rectangular packing

Initialize Q table as a matrix of 0
Initialize S_{opt}
for $t = 1$ to m **do**:
    Initialize s_0
        for $i = 1$ to n **do**:
            Choose a_i at s_{i-1} according to ε-greedy policy
            Take a_i, enter stage i, $s_i = a_i$
            if $i = n$ **then** $r_i = C/L'$
            else $r_i = 0$
            Update Q(s_{i-1}, a_i)
        end for
    Update S_{opt}
end for
Output S_{opt}

A reinforcement learning algorithm for the 2D-rectangular strip packing problem

- Computational experiments

Condition 1: $W = 20$, $w_i \in [5, 7]$, $l_i \in [8, 10]$, $n = 10$
Condition 2: $W = 40$, $w_i \in [6, 10]$, $l_i \in [11, 15]$, $n = 20$
Condition 3: $W = 60$, $w_i \in [11, 15]$, $l_i \in [16, 20]$, $n = 30$



(a) (b) (c)

Figure 4. Layout comparison between the stochastic sequence and the optimized sequence under the three conditions. (a) Condition 1. (b) Condition 2. (c) Condition 3