Nesting irregular polygons within a polygon

Rei Kaçani Anas Assel

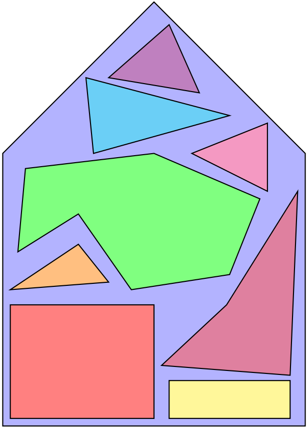
12 Jan 2024

Table of Contents

Table of Contents

Introduction [1]

The nesting problem generally refers to the problem of placing a number of shapes within the bounds of some material such that no pair of shapes overlap. The objective is to minimize the size of the material which is equivalent to maximizing the utilization of the material.



[1] Nielsen [2007]

Real world applications of nesting problem [1,2]

Significance of the nesting problem:

- **Industry:** fitting irregular materials (metal, wood, fabric) within a larger container reduces waste and production costs: shipping, aeronautics, woodworking, and footwear etc ...
- **Efficient storage and transportation:** by effective nesting in containers or trucks
- **Architecture, Engineering, and Design**: for example by nesting columns or panels efficiently, structural integrity is improved and construction costs are reduced; important in design for creating visually appealing compositions

[1] Nielsen [2007]

[2] Bennell and Oliveira [2008]

Table of Contents

Polygon

Polygonal curve (Polygonzug) [3]

A sequence of finetly many vertices connected by straight line segments such that each segment (except for the first) starts at the end of the previous segment.

Polygon [3]

A polygon with vertices $p_0, p_1, p_2 \dots p_n$ for $n \in N$ with $n \geq 3$ is a polygonal curve such that $p_0 = p_n$.

A polygon is called **simple** if no point of the plane belongs to more than two edges of the polygon and the only points which belong to precisely two edges are the vertices. [4]

[3] Held [2023]

[4] Shermer [1989]

Polygon [3]

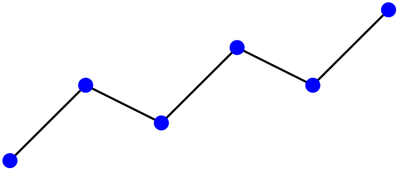


Figure: Polygonal Curve

[3] Held [2023]

Polygon [3]

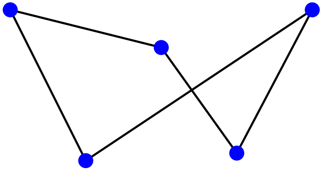


Figure: Non-Simple Polygon

[3] Held [2023]

Polygon [3]

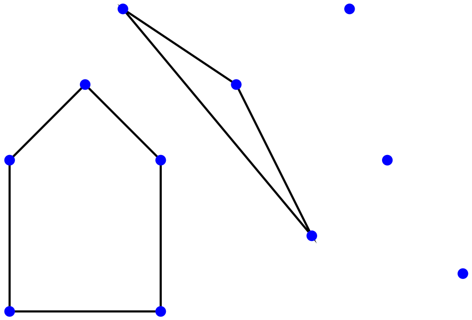


Figure: Planar Straight Line Graph

[3] Held [2023]

Polygon [3]

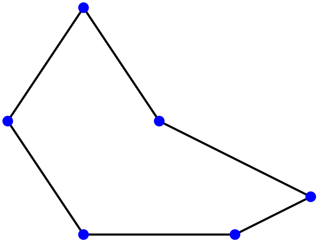


Figure: Simple Polygon

[3] Held [2023]

Irregular Polygons

Properties:

- polygons that have unequal sides and unequal angles
- can be convex or concave

Convex [3]

Let S be the set of points of polygon P. $\overline{pq}$ denotes the straight line segment between p and q. Polygon P ist convex if $\forall p, q \in S, \overline{pq} \subset S$

- can be orthogonal (all of its sides are either horizontal or vertical)

Examples:

rectangle, trapezium, kite, scalane triangle and other polygons with different number of unequal sides and angles.

[3] Held [2023]

Table of Contents

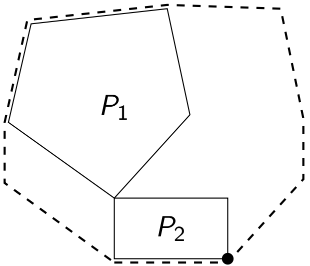
No-fit polygon concept (NFP) as a heuristic nesting algorithm [5]

NFP determines all arrangements that two arbitrary polygons may take such that the polygons touch but do not overlap.
Goal: find an optimal arrangement of objects inside a bigger object to make space usage maximal

[5] Burke and Kendall [1999]

No-fit polygon concept (NFP) as a heuristic nesting algorithm

Procedure: P_1 and P_2 are polygons that do not rotate.
P_1 remains stationary while P_2 orbits around P_1, staying in contact with it but never intersecting it. The reference point (filled circle) of P_2 becomes the boundary of NFP.
So the NFP of pieces P_1 and P_2, denoted by NFP_{P_1,P_2}, is the region in which the reference point of polygon P_2 cannot be placed because it would overlap polygon P_1.



NFP in nesting problem

Convex hull [3]

The convex hull of a set of points is the smallest convex set that contains the points.

- Polygon P_1 is the stationary polygon
- The reference point of P_2 is placed on each vertex of the NFP_{P_1,P_2} and for each position the convex hull for the two polygons is calculated.
- Convex hull that has the minimum area is returned as the best packing of the two polygons.
- This larger polygon now becomes the stationary polygon and the next polygon is used as the orbiting polygon. This process is repeated until all polygons have been processed.

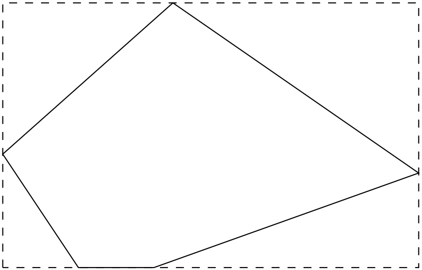
[3] Held [2023]

Bottom-left strategy [6]

The small pieces are sorted in a decreasing order according to their surface size and are placed as far to the left as possible.

Notations:

- reference point of a piece is the bottom-left corner of the enclosing rectangle of the shape



(0,0) Reference point

- (x_i, y_i) and (x_j, y_j) are respectively the reference points for the fixed piece i and the moving piece j

[6]Dowsland et al. [2001]

Bottom-left strategy

- E is the set of edges from the set of $NFP_{i,j}$
- (x_m^*, y_m^*) is the leftmost point on edge-m, such that $(x_m^*, y_m^*) \notin interior(NFP_{k,j})$ for any fixed piece k, where k=$1 \ldots i-1, k \neq j$.
- w_j is the width of the enclosing rectangle of piece j
- W is the total width of the container
- M is larger than the maximum packing length

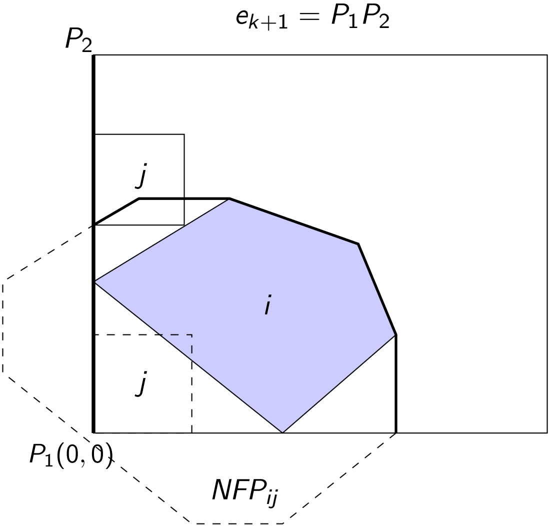
,

Bottom-left strategy

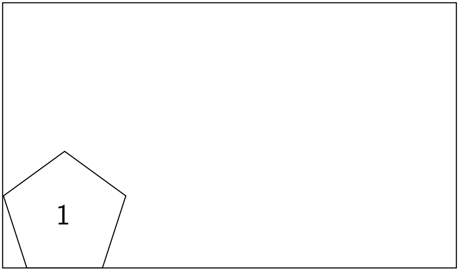
Procedure

1. Place the reference point of first piece i at (0,0). Set j=2 for the second piece
2. Set $(x_j, y_j) = (\infty, \infty)$
3. Let $E = e_1, e_2, \dots, e_k$ be the list of NFP edges from $NFP_{i,j}$
4. Let e_{k+1} be the left edge corresponding to the left edge of the sheet. Let $E = E \cup e_{k+1}$
5. Clip the edges in E against the region $(0, 0, M, W - w_i)$
6. For $m = 1 \dots k + 1$ find (x^*_m, y^*_m);
   set $(x_j, y_j) = (x^*_{m\ min}, y^*_{m\ min}) : x^*_{m\ min} \leq x^*_m \ \forall e_m \in E$ (in case of a tie in values of x^*_m, choose the one with minimum y^*_m)
7. Set j=j+1. If $j \leq n$ goto Step 2 , else STOP.

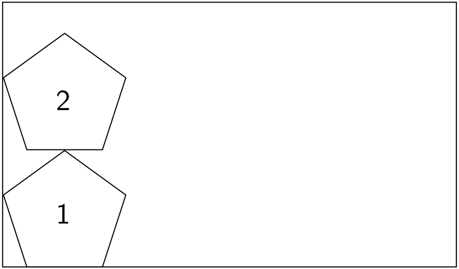
Bottom-left strategy



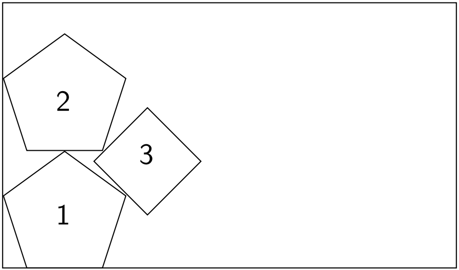
Bottom-left strategy



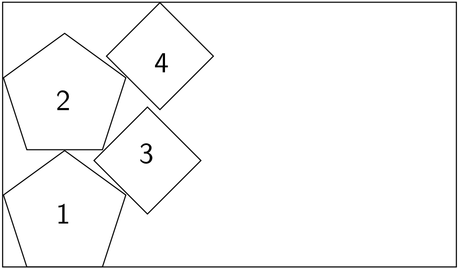
Bottom-left strategy



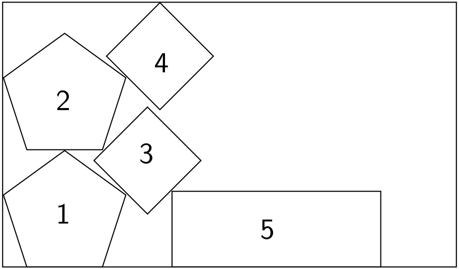
Bottom-left strategy



Bottom-left strategy



Bottom-left strategy



A constructive algorithm: The TOPOS Algorithm [7]

- In each iteration, characterized by the current partial solution, all the different pieces still available are placed in all admissible orientations, generating several new alternative partial solutions.
- The best new partial solution (which corresponds to minimal area) is selected to be the current partial solution for the next iteration.
- The algorithm stops when all pieces are placed

Notations:

P(i,j) - Piece of type i with an orientation j

PS - Partial solution

S - Set of pieces to place

n - number of different pieces

m - number of admissible orientations

waste – the difference between the area of the rectangular enclosure of PS and the area of all pieces already placed, including the one that is under evaluation

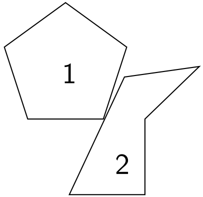
[7] Oliveira et al. [2000]

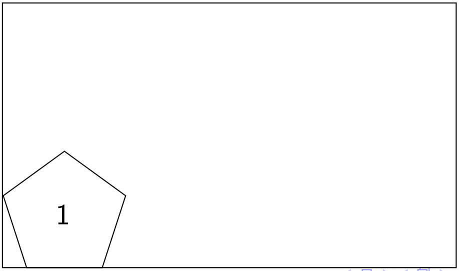
A constructive algorithm: The TOPOS Algorithm

```
PS ← {}
while S ≠ {} do                                        ▷ until no more pieces to place
    for i = 1 to n do                                  ▷ iterate over each piece type i
        if pieces_available_of_type(i) = TRUE then
            for j = 1 to m do                          ▷ each orientation of the current piece type i
                NFP ← no_fit_polygon(PS, P(i, j))
                placement_point ← area_minimization(PS, P(i, j), NFP)
                waste ← waste_evaluation_function(PS, P(i, j), placement_point)
                if waste < best_value then
                    best_value ← waste
                    best_piece ← i
                    best_orientation ← j
                    best_placement_point ← placement_point
                end if
            end for
            S ← S − {best_piece}
            PS ← join(PS, P(best_piece, best_orientation), best_placement_point)
        end if
    end for
```

A constructive algorithm: The TOPOS Algorithm

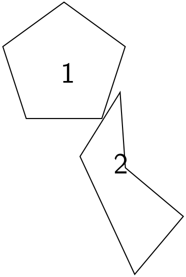
$NFP_{1,2}$ at different orientations of piece 2 is calculated:

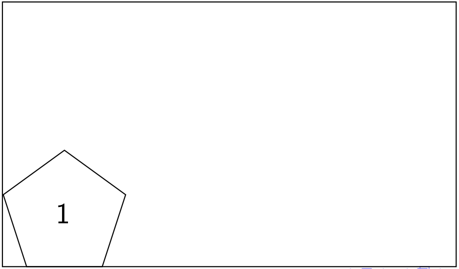




A constructive algorithm: The TOPOS Algorithm

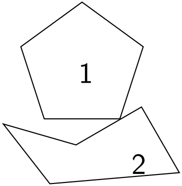
$NFP_{1,2}$ at different orientations of piece 2 is calculated:

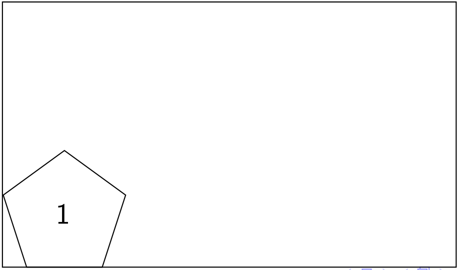




A constructive algorithm: The TOPOS Algorithm

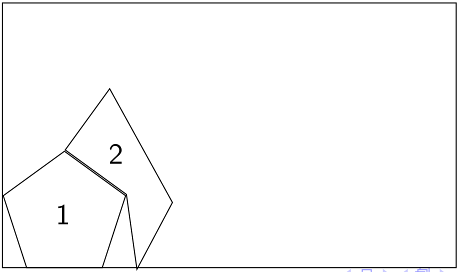
$NFP_{1,2}$ at different orientations of piece 2 is calculated:





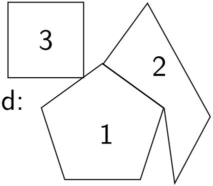
A constructive algorithm: The TOPOS Algorithm

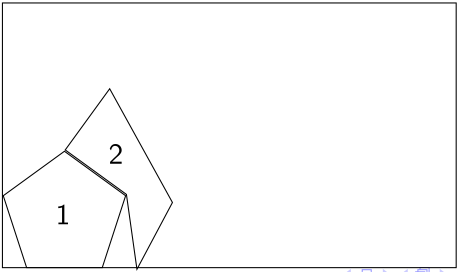
Best position and orientation of piece 2, $PS = \{1, 2\}$



A constructive algorithm: The TOPOS Algorithm

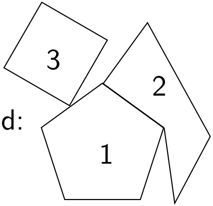
$NFP_{PS,3}$ at different orientations of 3 is calculated:

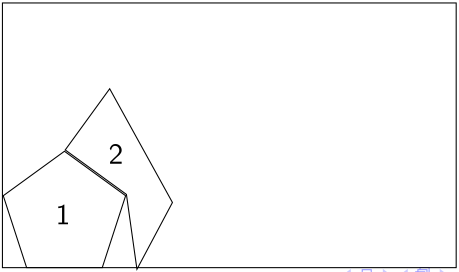




A constructive algorithm: The TOPOS Algorithm

$NFP_{PS,3}$ at different orientations of 3 is calculated:





A constructive algorithm: The TOPOS Algorithm

Best position and orientation of piece 3 and $PS = \{1, 2, 3\}$

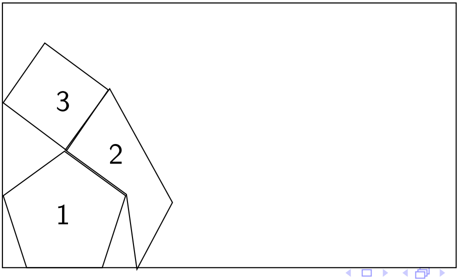


Table of Contents

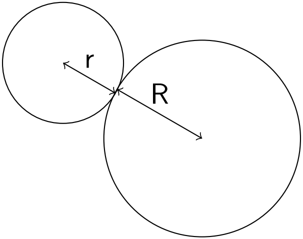
ϕ Function [8]

$\rightarrow$describes the interaction between two geometrical objects in such a way that the positions of the objects are the input and a real value is the output.

Example: ϕ Function for circles

$$\phi(x_1, y_1; x_2, y_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} - (r + R)$$

$$\begin{cases} \textit{pieces overlap} & \text{if } \phi(x_1, y_1; x_2, y_2) < 0 \\ \textit{pieces touch each other} & \text{if } \phi(x_1, y_1; x_2, y_2) = 0 \\ \textit{pieces are separated} & \text{if } \phi(x_1, y_1; x_2, y_2) > 0 \end{cases}$$



[8] Timmerman [2013]

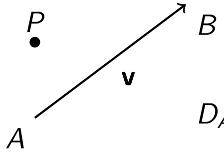
Direct trigonometry/ The D-function [8]

Direct trigonometry uses the polygons directly. It consists of tests for calculating if two lines intersect.
D-function is used to calculate overlap between two polygons using trigonometry.

$$D_{A,P,B} = (X_A - X_B) * (Y_A - Y_P) - (Y_A - Y_B) * (X_A - X_P) \qquad (1)$$

where A and B stand for the beginning and ending points of the line and P a point in space.

$$\begin{cases} P \ on \ the \ right \ of \ AB & \text{if } D_{A,P,B} < 0 \\ P \ on \ AB & \text{if } D_{A,P,B} = 0 \\ P \ on \ the \ left \ of \ AB & \text{if } D_{A,P,B} > 0 \end{cases}$$



$D_{A,B,P} > 0$

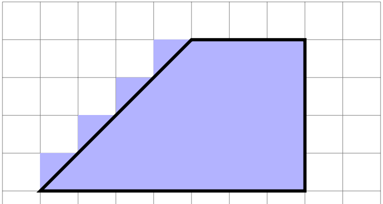
[8] Timmerman [2013]

Pixel/Raster method [8]

Representation: existing space(1), empty space(0), overlapping space(>1)

0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	0	0
0	0	0	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	0	0
0	1	1	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0

Pixel representation



Raster representation

[8] Timmerman [2013]

Conclusion

- When placing polygons inside another polygon is necessary to *minimize the gaps* between actual nested polygons, so more polygons can be nested and the *utilization of space or material is maximized*
- used in industry, design, architecture, transportation etc
- No-Fit Polygon (NFP), Bottom Left Strategy, constructive algorithms
- Pixel/Raster method and D function make nesting process more optimal

Thank you for your attention!

References I

Julia A. Bennell and Jose F. Oliveira. The geometry of nesting problems: A tutorial. *European Journal of Operational Research*, 184(2):397–415, Jan 2008.

Edmund Burke and Graham Kendall. Applying ant algorithms and the no fit polygon to the nesting problem. In *Australasian Joint Conference on Artificial Intelligence*, volume 1747, page 453–464, Sydney, Australia, Dec 1999.

Kathryn A. Dowsland, Subodh Vaid, and William B. Dowsland. An algorithm for polygon placement using a bottom-left strategy. *European Journal of Operational Research*, 141:371–381, May 2001.

Martin Held. Computational geometry. pages 32–35,101–102, Aug 2023.

References II

Benny K. Nielsen. An efficient solution method for relaxed variants of the nesting problem. In Joachim Gudmundsson and C. Barry Jay, editors, *Theory of Computing 2007. Proceedings of the Thirteenth Computing: The Australasian Theory Symposium (CATS2007). January 30 - Febuary 2, 2007, Ballarat, Victoria, Australia, Proceedings*, volume 65 of *CRPIT*, pages 123–130. Australian Computer Society, 2007.

Jose Fernando Oliveira, A. Miguel Gomes, and Jose Soeiro Ferreira. Topos – a new constructive algorithm for nesting problems. pages 263–284, Jan 2000.

Thomas C. Shermer. Visibility properties of polygons. page 18, Jun 1989.

Mattijs Timmerman. Optimization methods for nesting problems. page 49, 2013.