RESEARCH AND DEVELOPMENT OF ALGORITHMS FOR SOLVING THE TWO-DIMENSIONAL IRREGULAR CUTTING STOCK PROBLEM

PRESENTER: Maxim Poryvai, Bachelor student

RESEARCH SUPERVISOR: Andrei Chupakhin, Research fellow

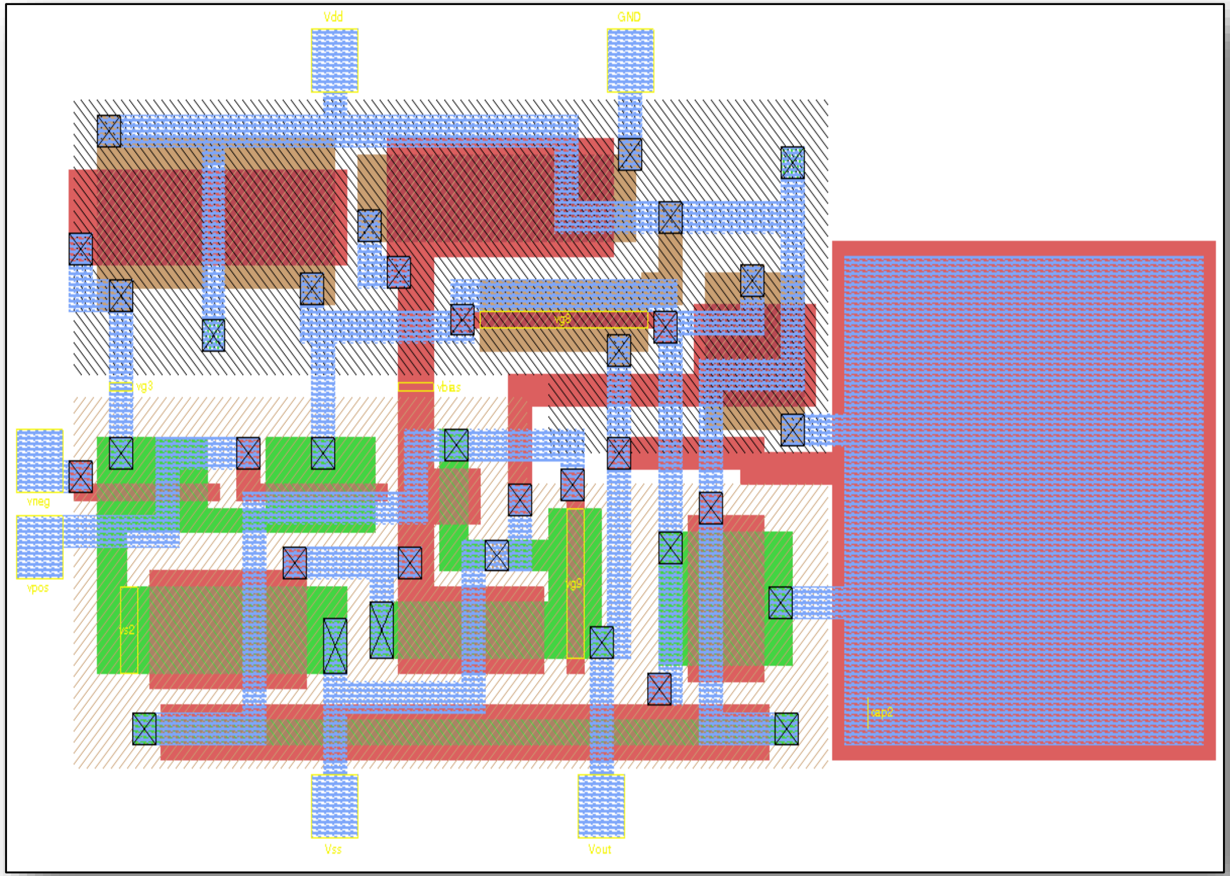
Lomonosov Moscow State University

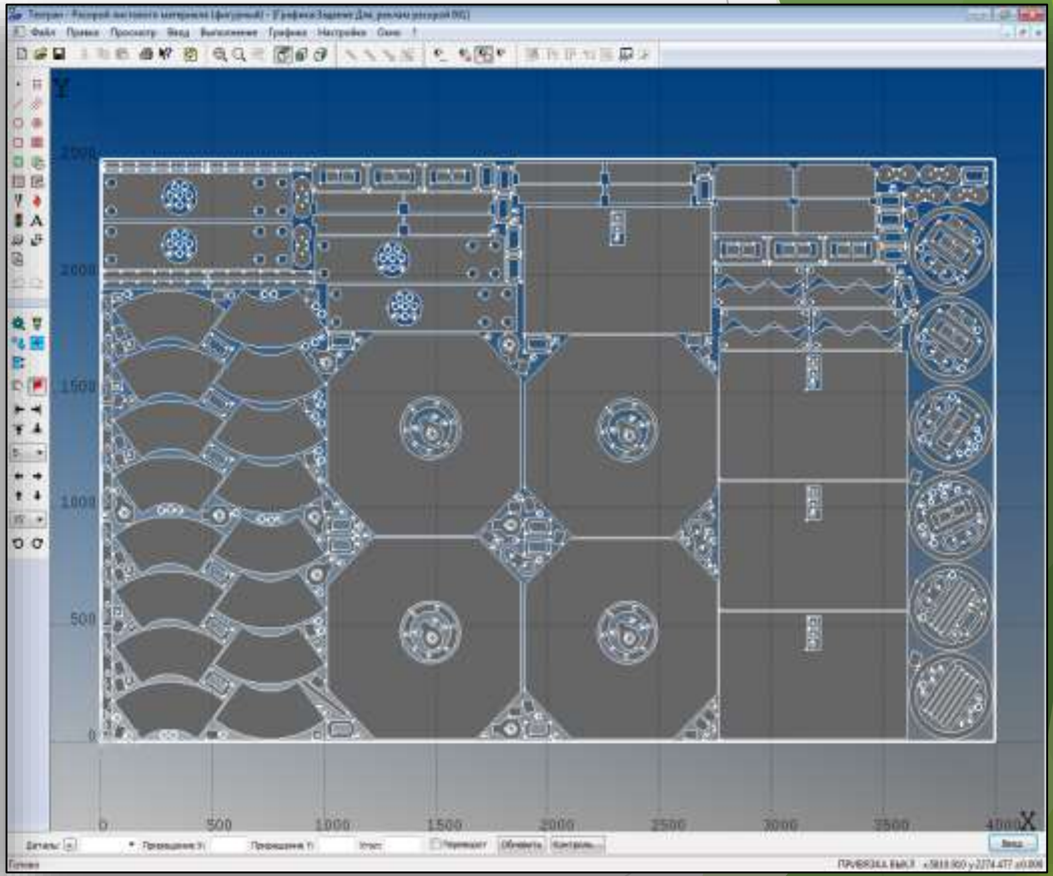
Faculty of Computational Mathematics and Cybernetics

Department for computer systems and automation

2022

MOTIVATION



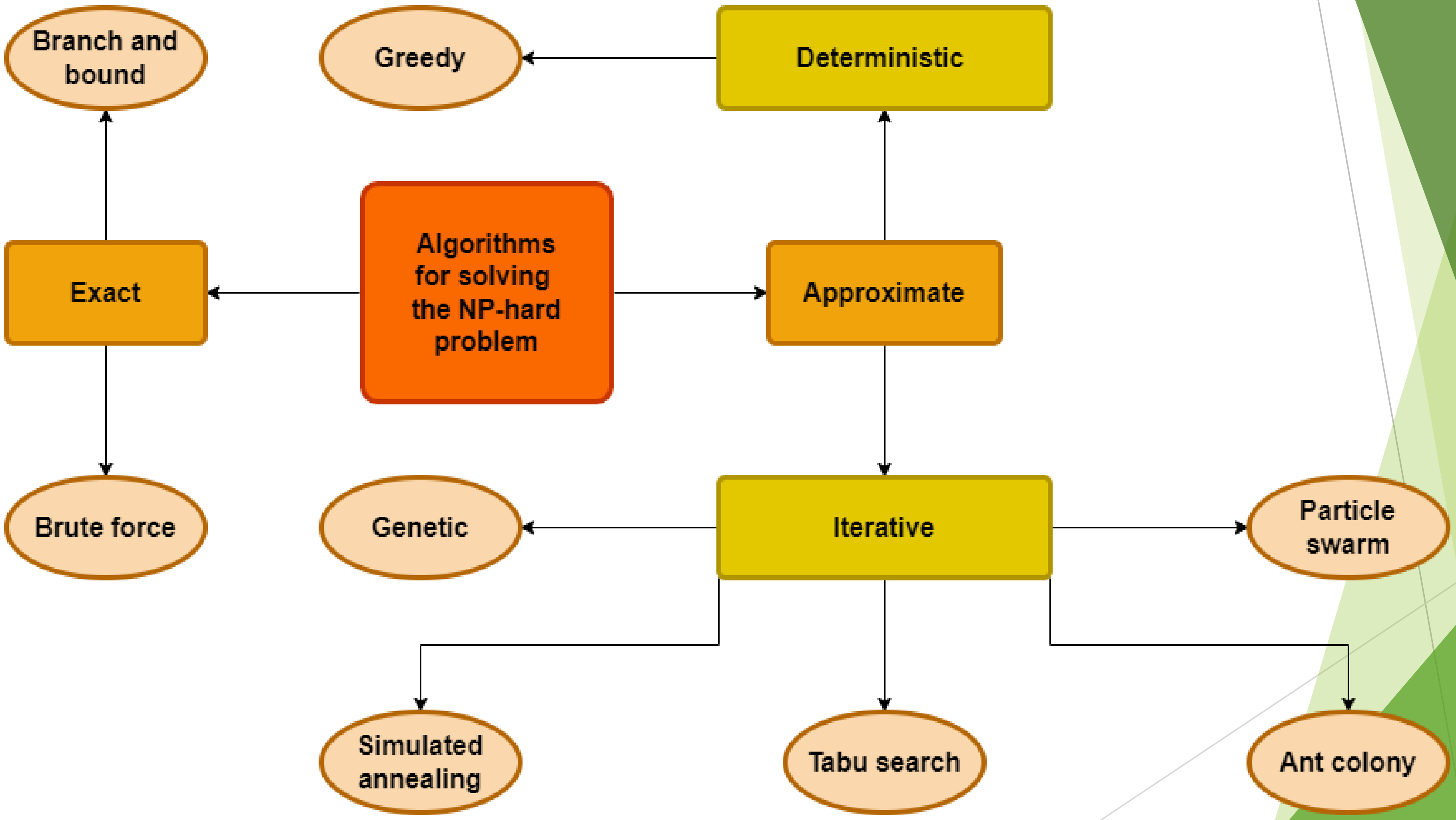


PURPOSE OF RESEARCH

Creation of a **library containing implementations of various algorithms** in a unified style, with a unified class hierarchy

- Conducting an **overview of the subject area**
- **Development of a methodology** for research and comparison of algorithm implementations
- **Experimental study** of algorithm implementations based on the developed methodology

CLASSIFICATION OF ALGORITHMS



NOTATIONS

1. $Q = \{q_i, i = 1, \ldots, M, \ldots\}$ — rectangular sheets with the same size
2. $S = \{s_i, i = 1, \ldots, n\}$ — figures (polygons)
3. Each polygon s_i is given by the sequence of its vertices $s_{ij} = (x_{ij}, y_{ij}), j = 1, \ldots, k_i$
4. M — the number of sheets used to place a given set of figures S
5. d_i — convex hull for the figures placed on q_i
6. e_i — envelope (minimal bounding rectangle) for the figures placed on q_i
7. $ar(x)$ — area of x, $x \in S \cup Q \cup \{d_i, i = 1, \ldots, M\} \cup \{e_i, i = 1, \ldots, M\}$
8. If the figure s_j is placed on the sheet q_i, we will write $s_j \in q_i$

9. $$FF_a = \frac{\sum_{i=1}^{M-1} \{(1 - \sum_{s_j \in q_i} ar(s_j))/ar(q_i)\} + (1 - \sum_{s_j \in t_M} ar(s_j))/ar(e_M)}{M}$$

10. $$FF_b = \frac{\sum_{i=1}^{M} (1 - \sum_{s_j \in q_i} ar(s_j))/ar(d_i)}{M}$$

MATHEMATICAL FORMULATION

Given

1. Q, a — sheet length, b — sheet width
2. S, s_{ij} $(i = 1, \dots, n; j = 1, \dots, k_i)$

Constraints

1. $\forall i \in \{1, \dots, n\}$: s_i has no holes
2. $\forall i \in \{1, \dots, n\}$: s_i can be rotated by angles $\alpha = 90t$ $(t \in \{0, 1, 2, 3\})$ degrees using the following transformation:
   $\forall j \in \{1, \dots, k_i\}$:
   $(x_{ij}, y_{ij}) \to (x_{ij} \cdot \cos\alpha - y_{ij} \cdot \sin\alpha, x_{ij} \cdot \sin\alpha + y_{ij} \cdot \cos\alpha)$
3. $\forall i \in \{1, \dots, n\}$: s_i can be mirrored by the following transformation:
   $\forall j \in \{1, \dots, k_i\} : (x_{ij}, y_{ij}) \to (x_{ij}, -y_{ij})$

Find

$$FF = 1 - (A \cdot FF_a + B \cdot FF_b) \to max \quad (1)$$

GREEDY ALGORITHM

Pseudocode:

```
sort the figures in descending area order or in random order (set on startup)
while number of remaining figures > 0:
    take the next figure
    apply all possible transformations to it
    set the value of the maximum of the fitness function to -1
    for each transformed variant of current figure:
        for each of the current sheets:
            calculate the value of the fitness function for the lowest left possible figure placement
    if it was not possible to place the figure anywhere:
        take a new sheet
        place the figure in the bottom-left corner of a new sheet
    else:
        apply the change with the maximum value of the fitness function
    remove a figure from the list of unplaced figures
```

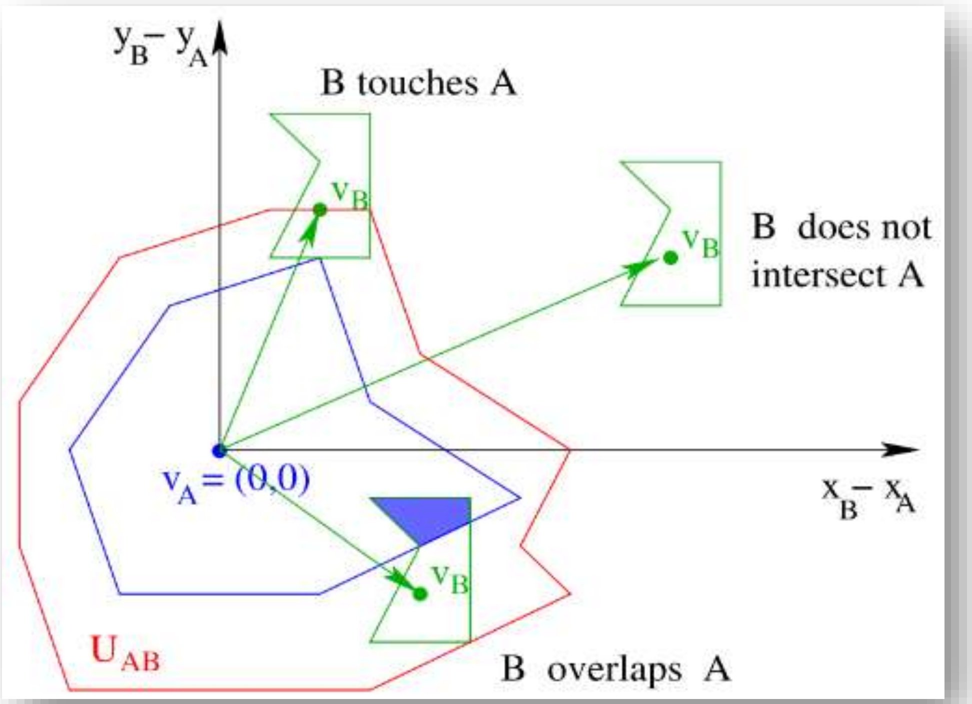
SIMULATED ANNEALING ALGORITHM

Pseudocode:

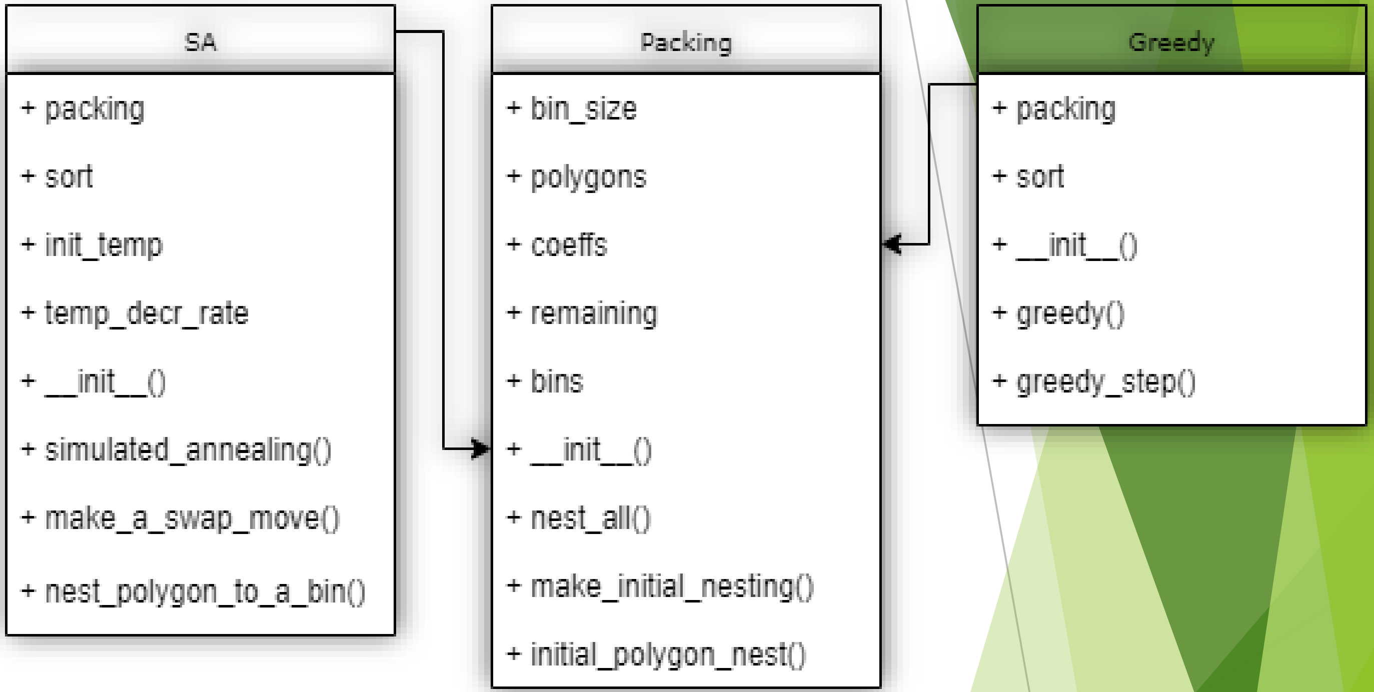
generate the initial placement of the figures (sorting is random or in descending area order; set at startup; differs from the greedy algorithm in that the shapes are not transformed and are each time placed in the lowest left allowed place only on the last of the current sheets)

```
calculate the value of the "energy of the system" E = 100000 * (1 – FF) after the initial placement
while temperature > 0:
    try to make a change (take a random figure from each of two random sheets and change the sheets for these figures: on the new sheet, the placement for both figures is lowest left possible)
    if the attempt is successful:
        calculate the difference between the values of the "energies" after the change and before it
        if (difference < 0) OR (random number from [0, 1] < exp(- difference / temperature)):
            apply the change
            update the "system energy" value
            temperature = temperature – step
        else:
            temperature = temperature – step * 0.1
    else:
        temperature = temperature – step * 0.1
```

IMPLEMENTATION



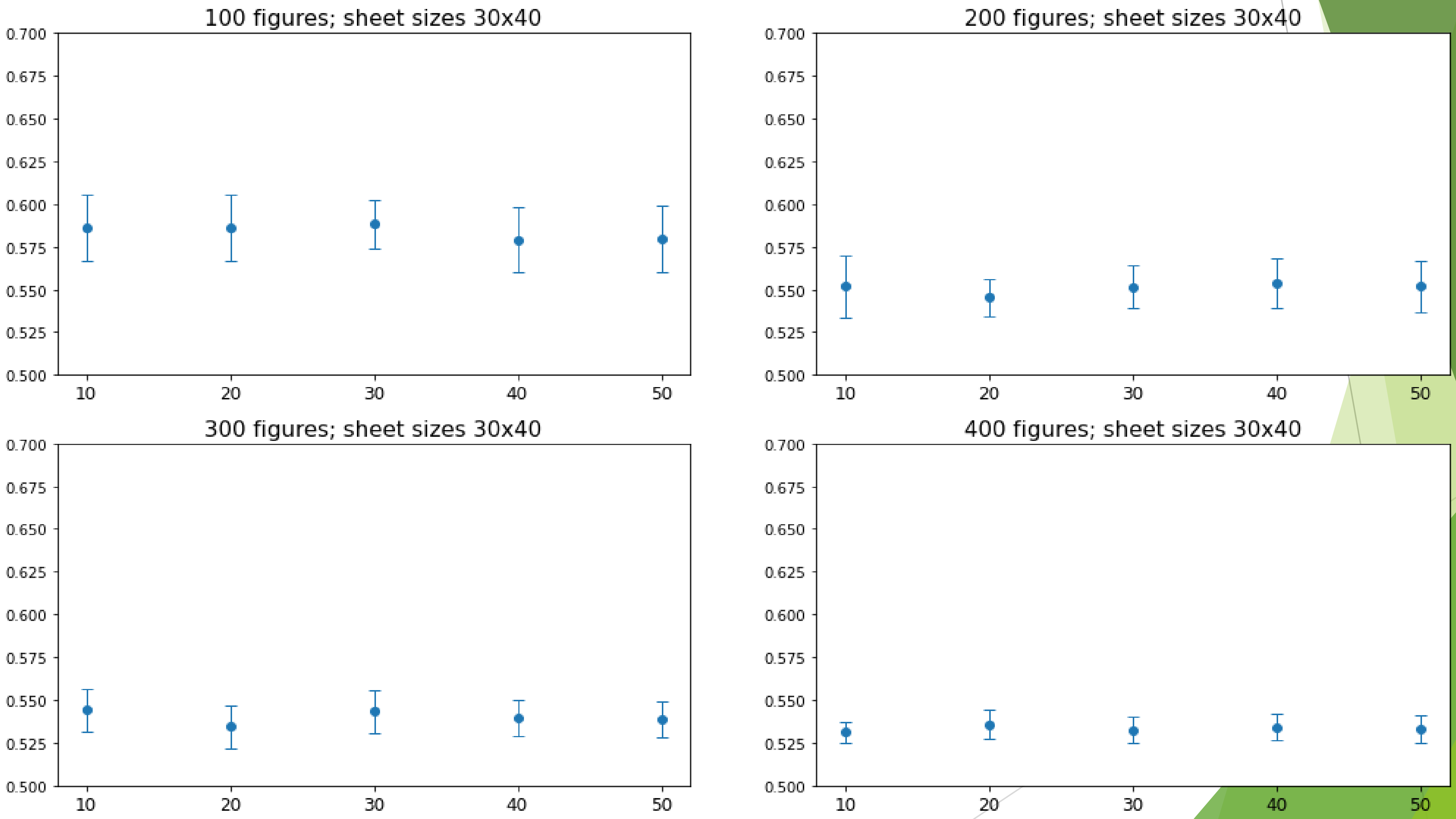
No-Fit Polygon (NFP)



Class diagram

METHODOLOGY

Mean and standard deviation of the fitness function values depending on the number of runs

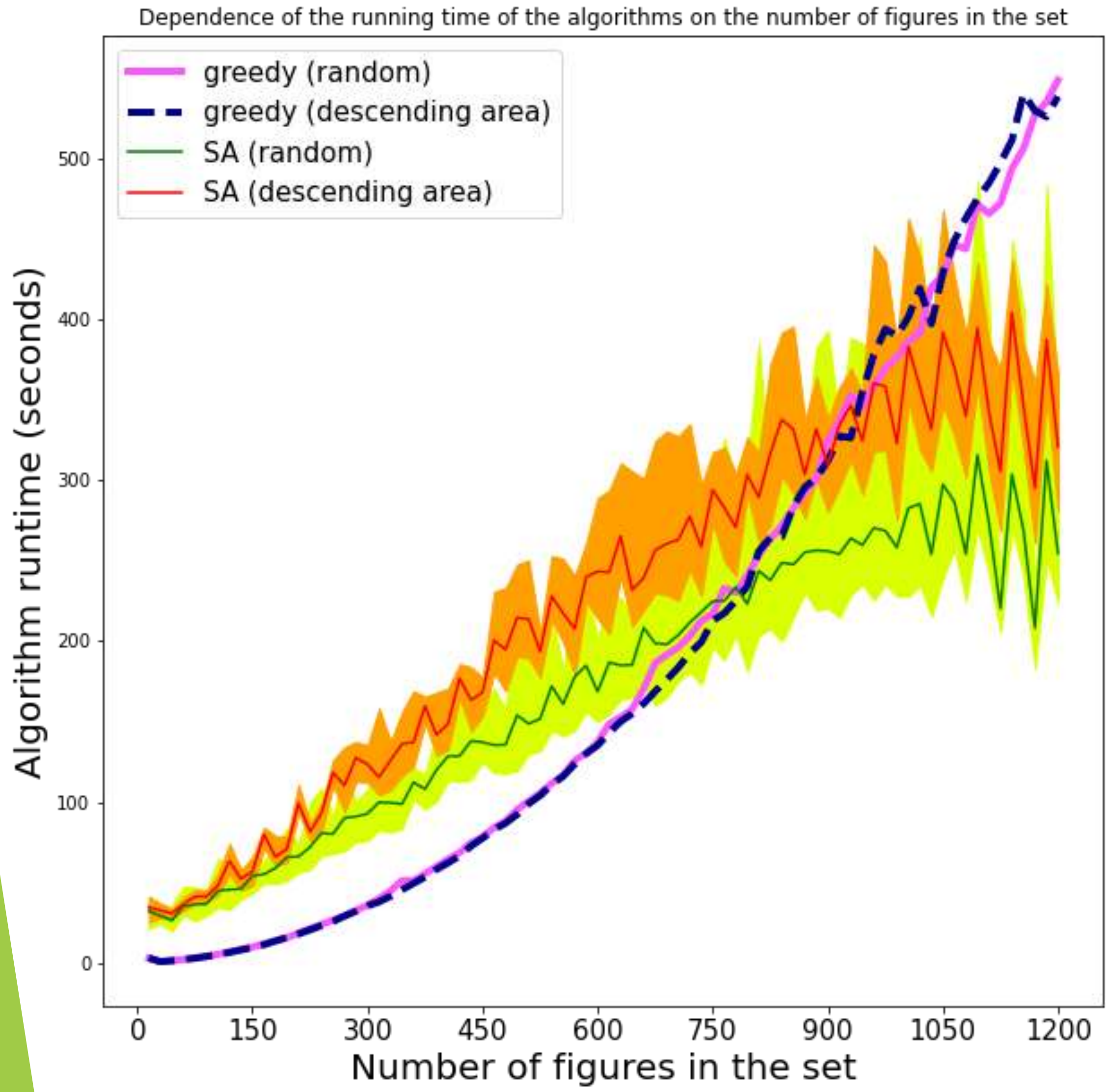


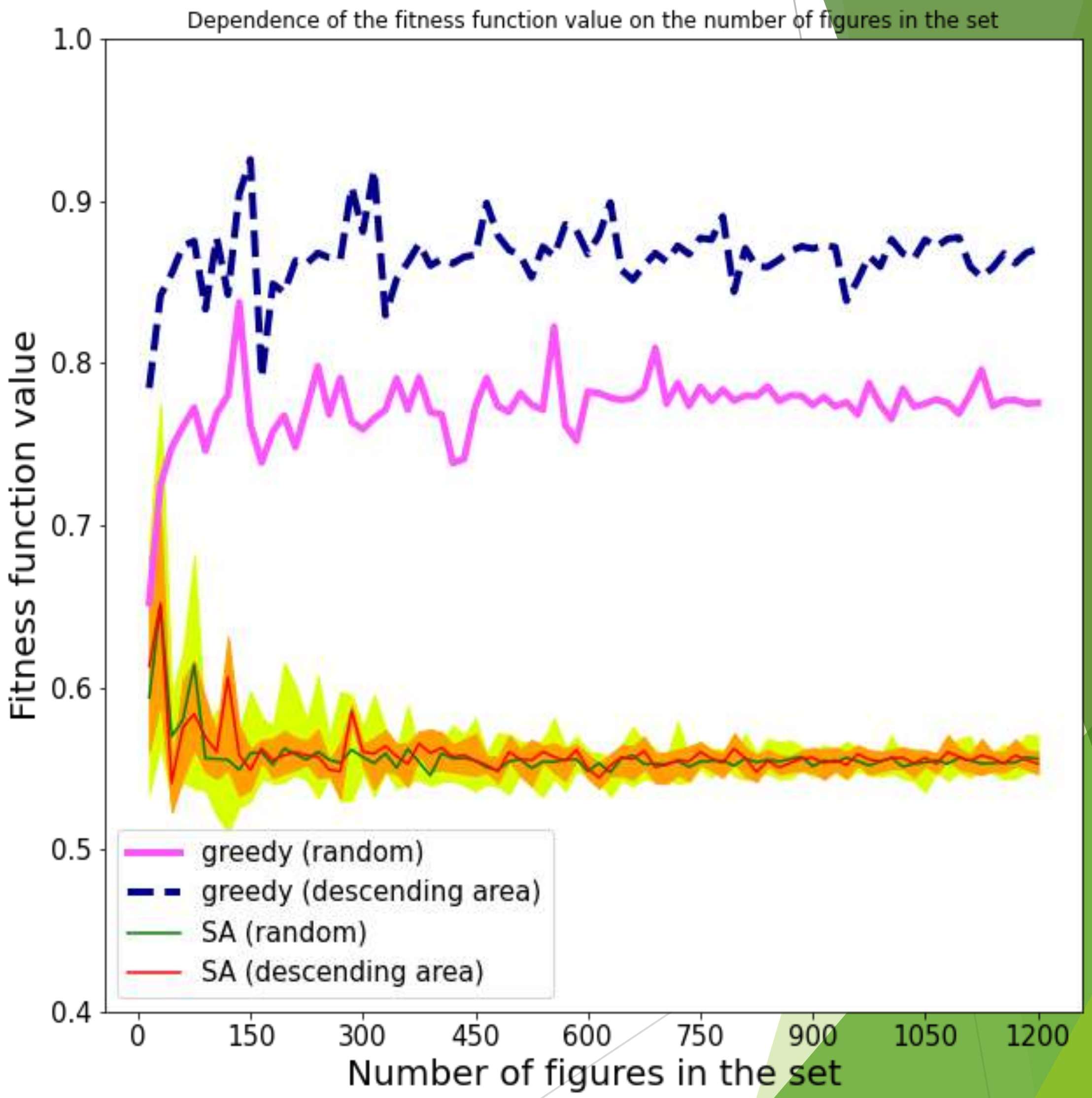
EXPERIMENTAL STUDY

The following parameters for the generator function were used for the experiments:

- sizes of sheets: **30x40**;
- algorithm: **greedy, simulated annealing (initial temperature: 1200, temperature decrease rate: 1)**;
- types of figures: **rectangle (5x4)**, **equilateral triangle (side length: 10)**, **isosceles trapezium (bases: 9 and 3; height: 3)**;
- initial number of each figure: **5**;
- final number of each figure: **400**;
- number increase step of each figure: **5**;
- figures sorting type: **random, in descending area order**.

RESULTS


Dependence of the running time of the algorithms on the number of figures in the set
greedy (random)
greedy (descending area)
SA (random)
SA (descending area)
Algorithm runtime (seconds)
Number of figures in the set
0
100
200
300
400
500
0
150
300
450
600
750
900
1050
1200


Dependence of the fitness function value on the number of figures in the set
greedy (random)
greedy (descending area)
SA (random)
SA (descending area)
Fitness function value
Number of figures in the set
0.4
0.5
0.6
0.7
0.8
0.9
1.0
0
150
300
450
600
750
900
1050
1200

FUTURE WORK

Expand the review of the algorithms by considering newer papers presenting modifications of classical algorithms as well as possibly new approaches to solving the cutting problem

Implement other algorithms to solve the problem and add the implementations to the library

Modify and optimize algorithm implementations to improve quality and reduce runtime

After completing the Python code, rewrite the time-critical parts in C++, which will also speed up the algorithms implementations

Thank you for your attention

+7(977)351-62-74

- Phone

mvporyvaj@lvk.cs.msu.ru

- E-mail

https://github.com/maxporyvay/2D-irregular-stock-cutting-problem-multi-algorithm-solution

- Project repository