

# **JPACKING: PROGRAMA PARA DISTRIBUIÇÃO OTIMIZADA DE POLÍGONOS EM UM PLANO BIDIMENSIONAL UTILIZANDO ALGORITMOS GENÉTICOS**

Aluno: Rodrigo D'avila

Orientador: Prof. Daniel Theisges  
dos Santos, Mestre

# Roteiro

- **Introdução**
  - Objetivos
- **Fundamentação Teórica**
  - Requisitos
  - Trabalhos Correlatos
- **Desenvolvimento**
  - Especificação
  - Operacionalidade
- **Resultados**
  - Discussões
  - Comparação
- **Conclusões**
  - Sugestões

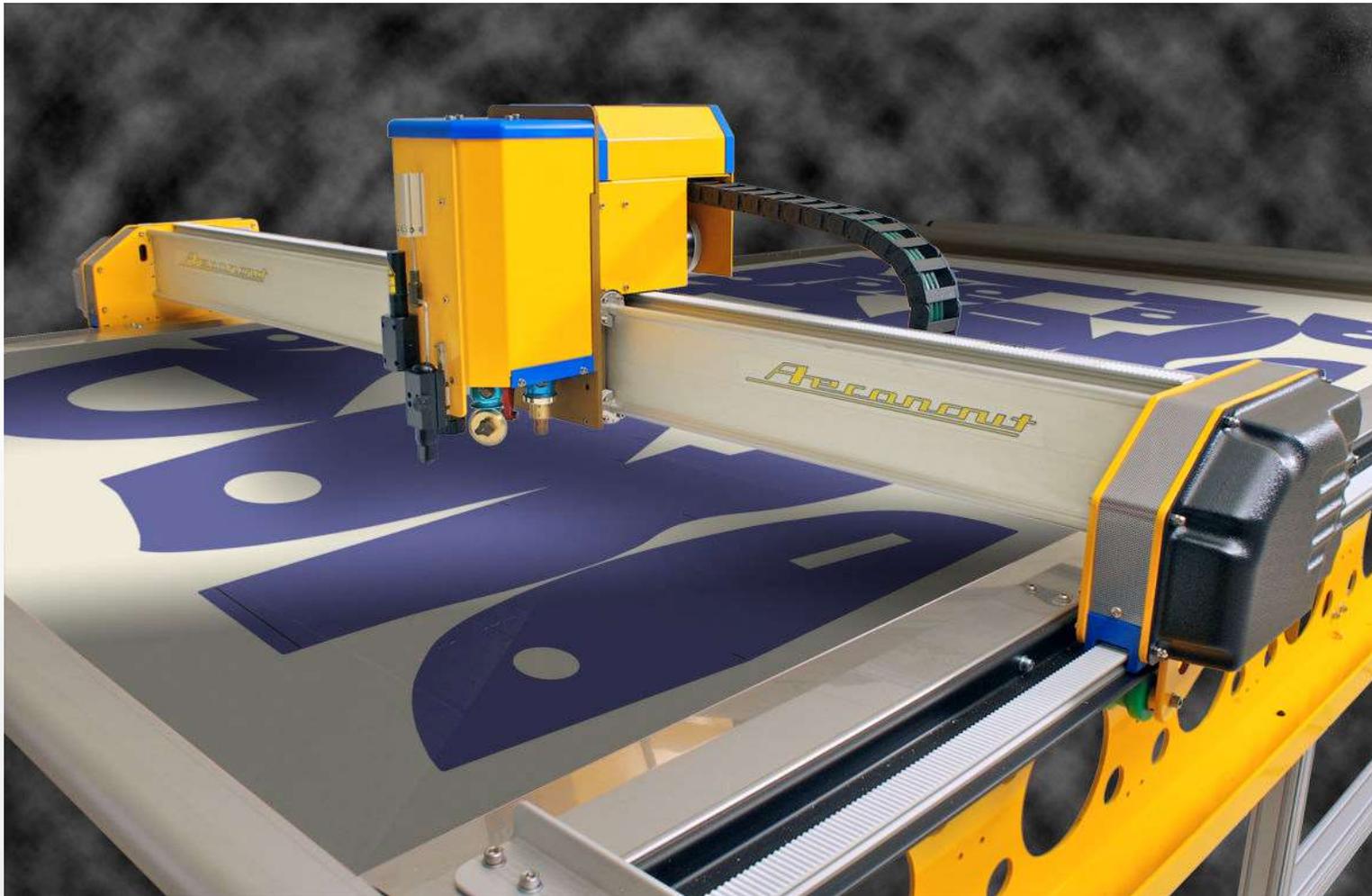
# Roteiro

- **Introdução**
  - Objetivos
- **Fundamentação Teórica**
  - Requisitos
  - Trabalhos Correlatos
- **Desenvolvimento**
  - Especificação
  - Operacionalidade
- **Resultados**
  - Discussões
  - Comparação
- **Conclusões**
  - Sugestões

# Introdução

- A distribuição otimizada de **polígonos irregulares** em um plano bidimensional é um problema frequente em:
  - Manufatura de peças de roupa, tecido, couro, até metal e madeira
- Evitar o problema de desperdício de matéria-prima.

# Introdução



# Introdução

- Trata-se de um problema NP-Difícil.
- Não existe uma resposta determinística em um tempo polinomial.
- Mas pode ser dividido em dois passos principais:
  - Estratégia de encaixe.
  - Ordem de empacotamento.

# Objetivos

- **Objetivo Geral:**

- implementar um software capaz retornar uma solução completa para o problema de disposição de polígonos em uma superfície plana bidimensional

- **Objetivos específicos:**

1. Implementar o algoritmo de Bottom-left fill com NFP;
2. Implementar o Algoritmo Genético com operador de Crossover, Mutação, Seleção e Fitness;
3. Integrar os algoritmos (1) e (2) em um programa com saída e entrada em SVG.

# Roteiro

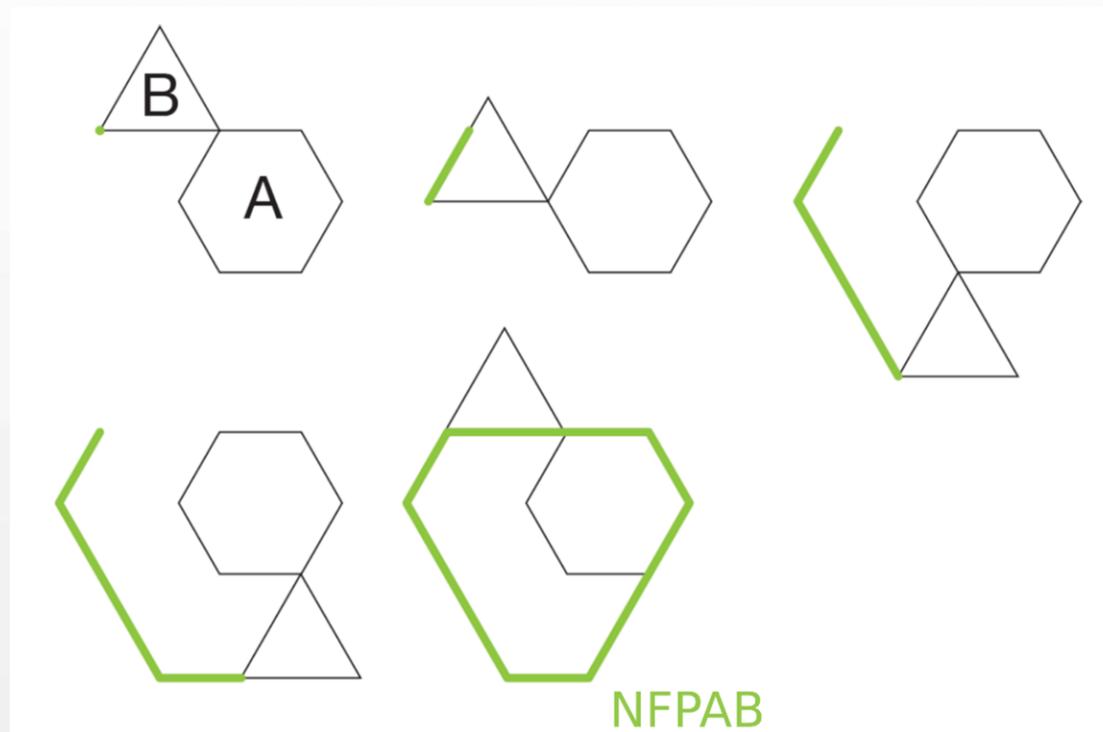
- **Introdução**
  - Objetivos
- **Fundamentação Teórica**
  - Requisitos
  - Trabalhos Correlatos
- **Desenvolvimento**
  - Especificação
  - Operacionalidade
- **Resultados**
  - Discussões
  - Comparação
- **Conclusões**
  - Sugestões

# Fundamentação

- Estratégia de encaixe
  - Algoritmo de No-Fit Polygon (NFP).
  - Algoritmo de Bottom-left fill.
- Ordem de empacotamento.
  - Algoritmo Genético

# Algoritmo de NFP

- Polígonos convexos e não convexos.
- Técnica orbital.

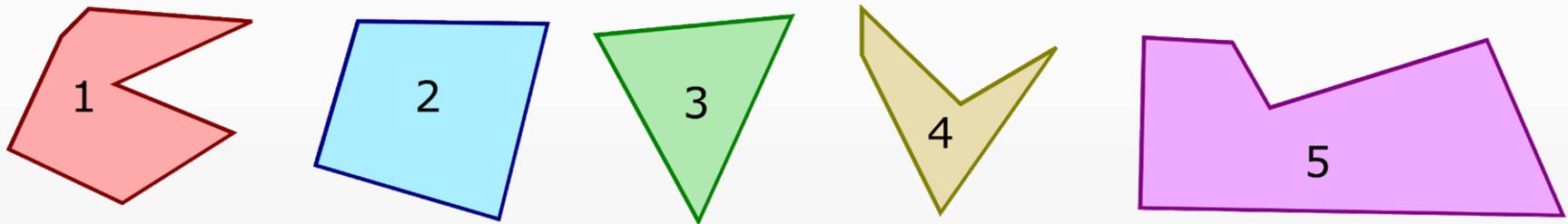


# Algoritmo de bottom-left fill

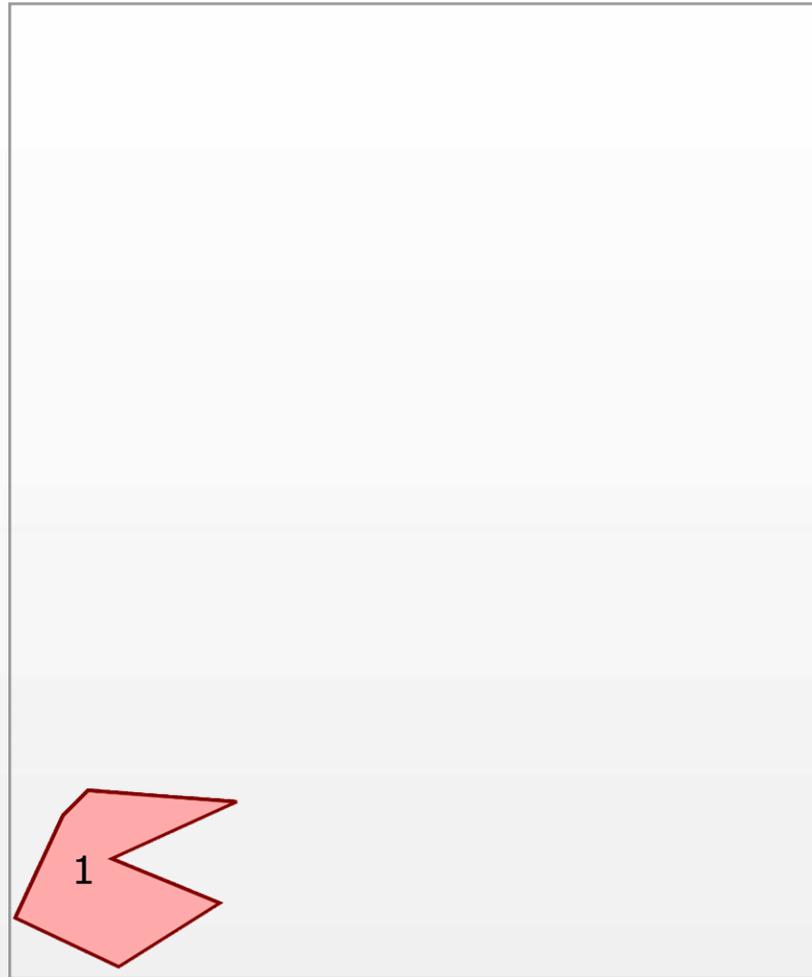
- Estratégia de encaixe
- Obedece uma ordem de inserção.
- Checagem de colisão com o NFP.
- Respeita
  - número de rotações
  - altura da forma

# Algoritmo de bottom-left fill

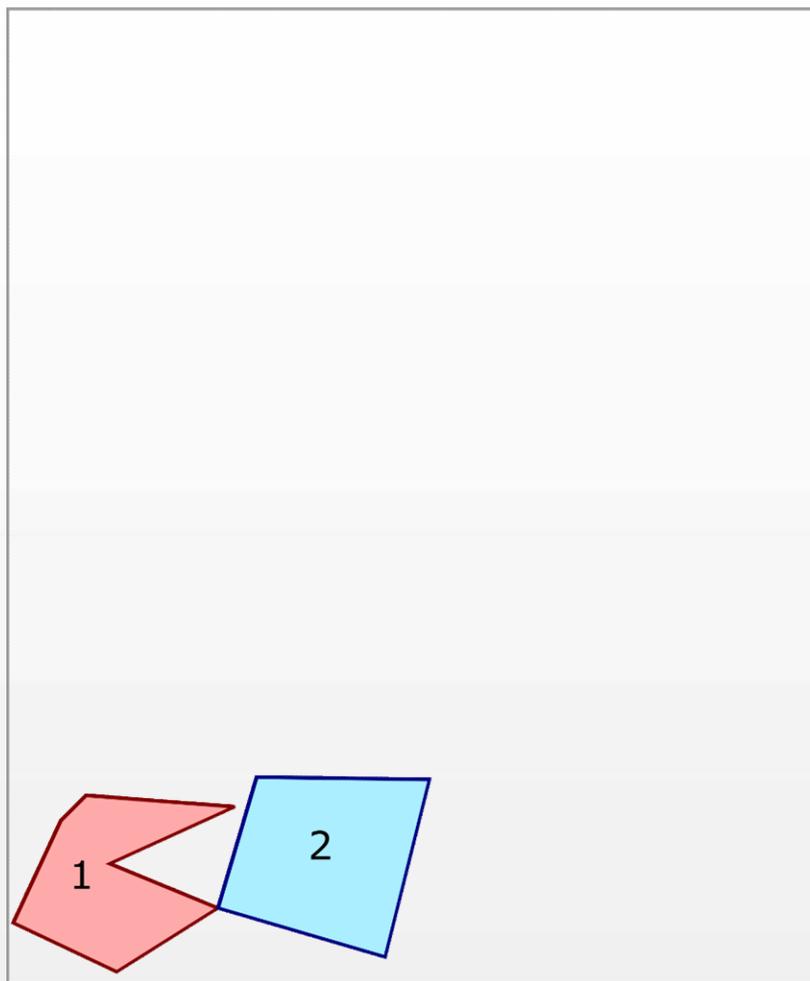
- Ordem de empacotamento



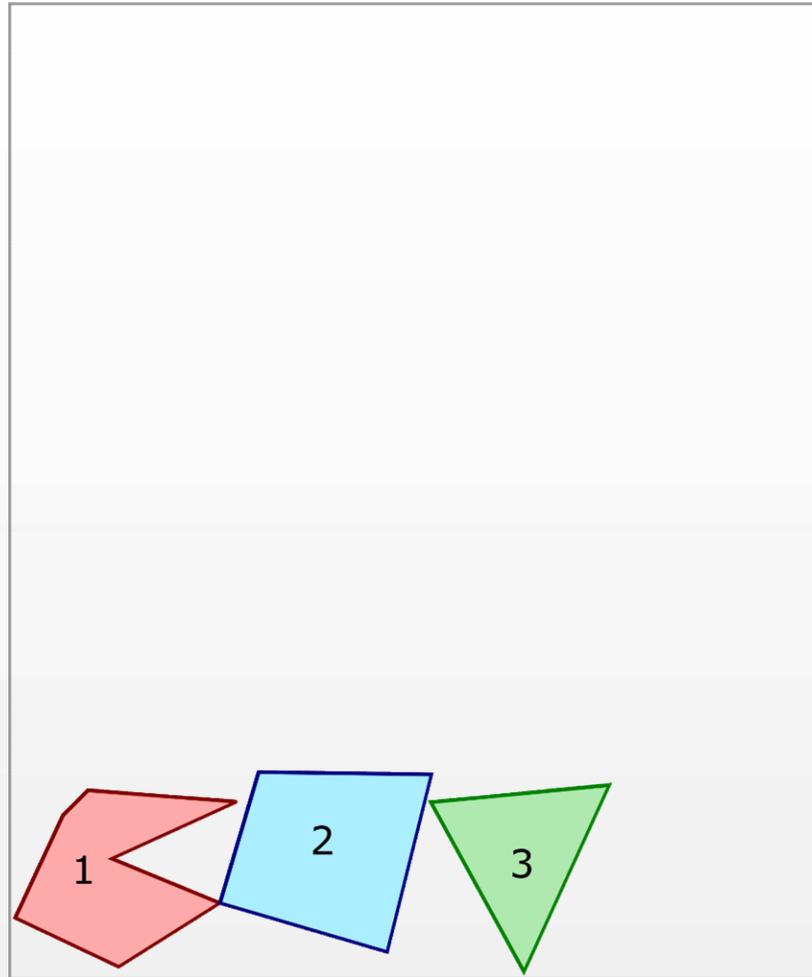
# Algoritmo de bottom-left fill



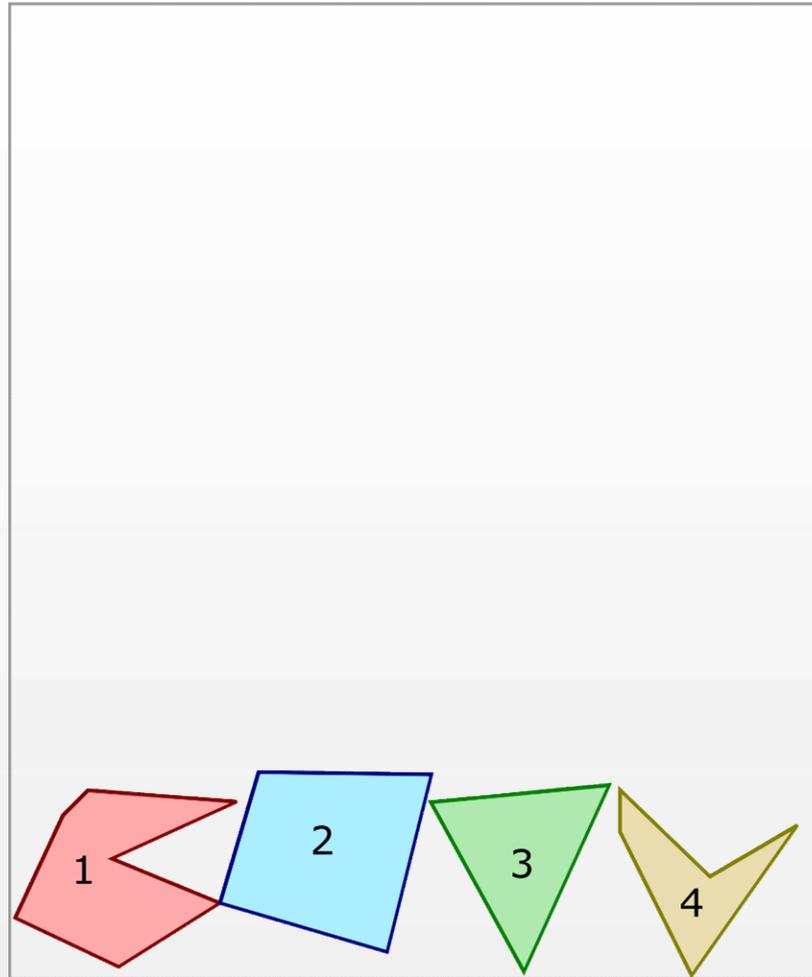
# Algoritmo de bottom-left fill



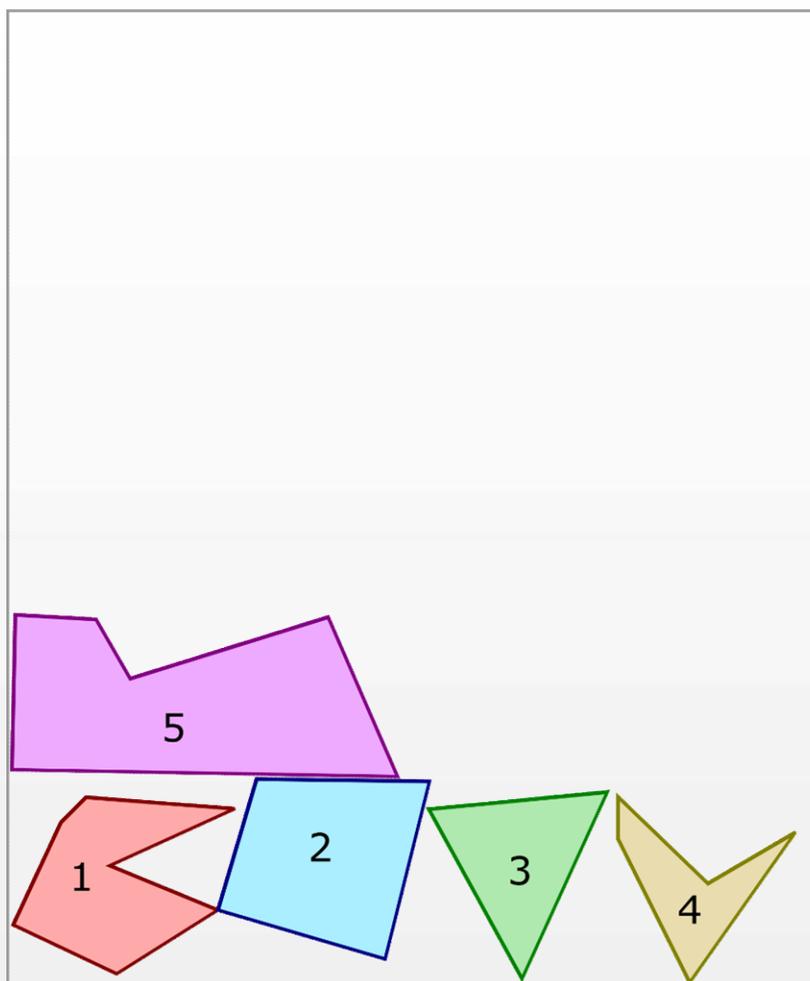
# Algoritmo de bottom-left fill



# Algoritmo de bottom-left fill

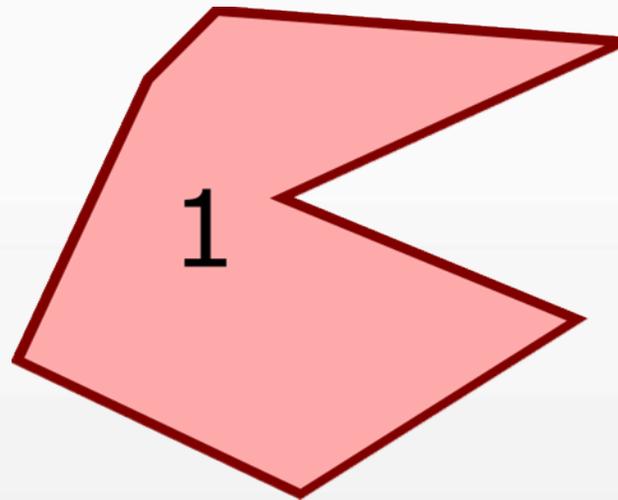


# Algoritmo de bottom-left fill



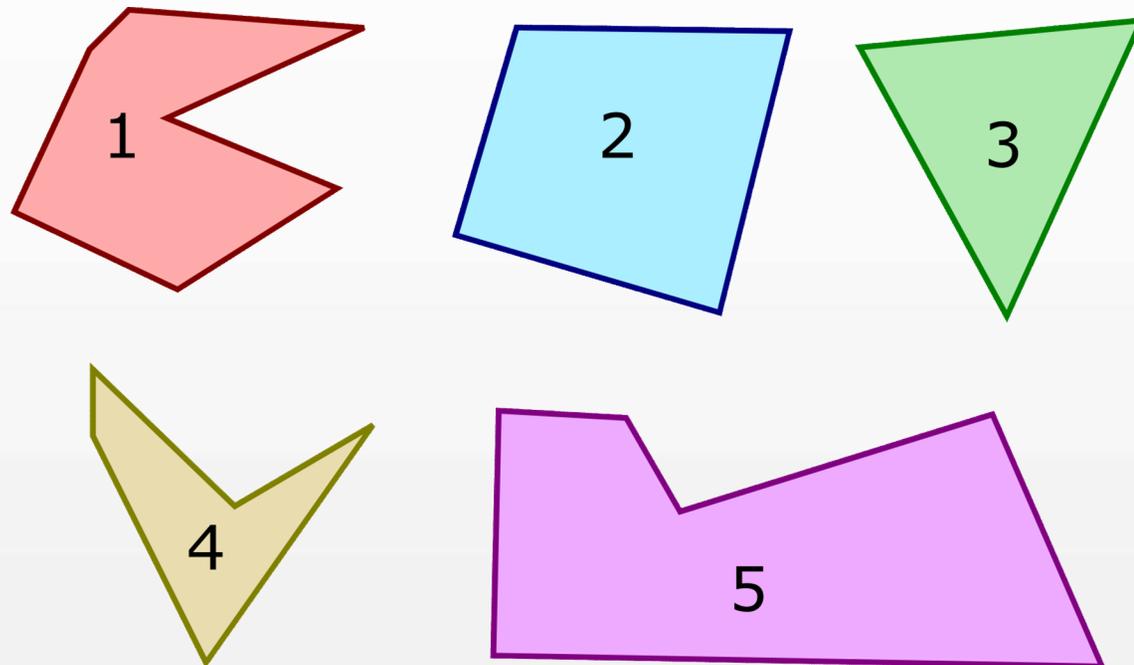
# Algoritmo Genético

- Gene



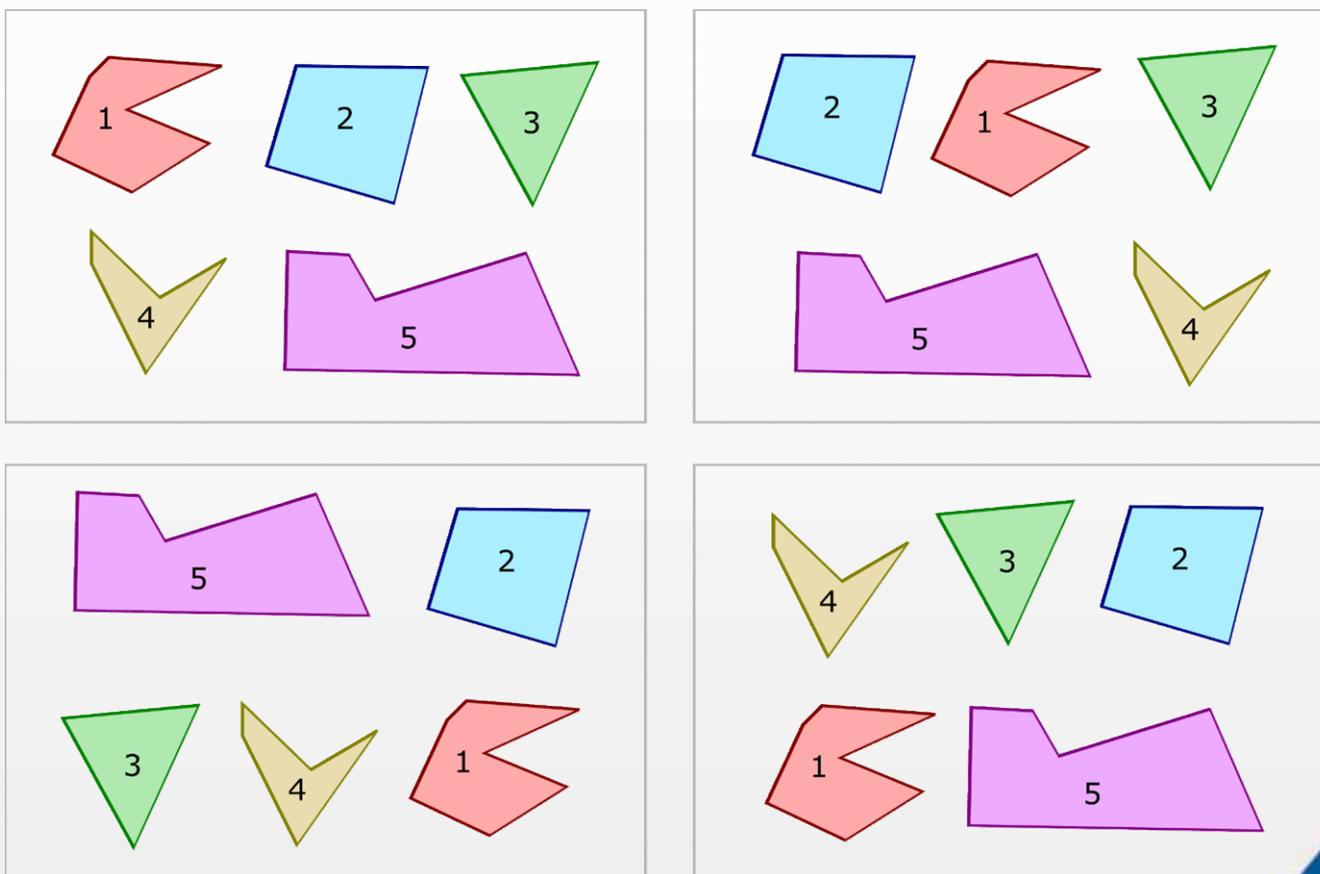
# Algoritmo Genético

- Indivíduo



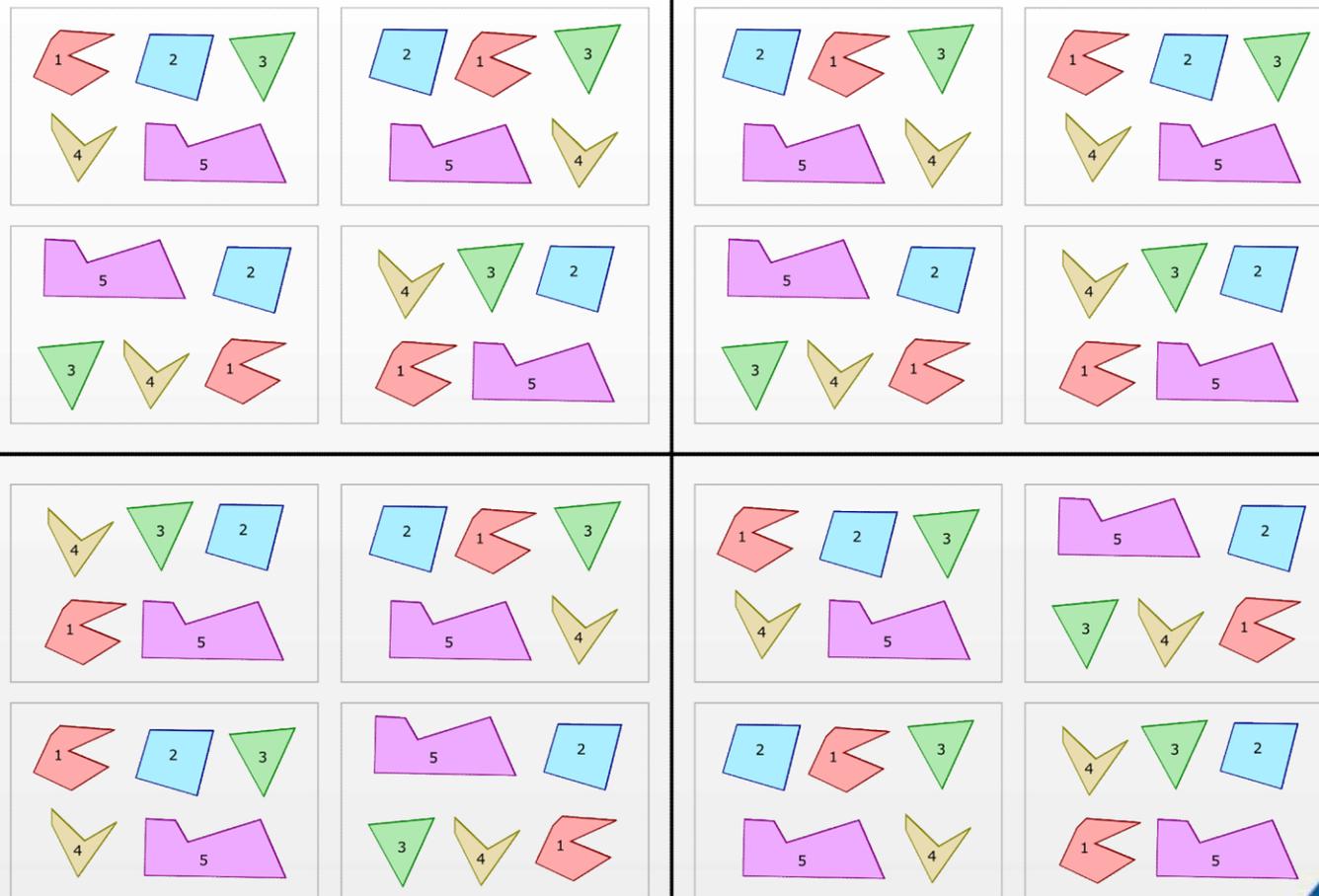
# Algoritmo Genético

- População



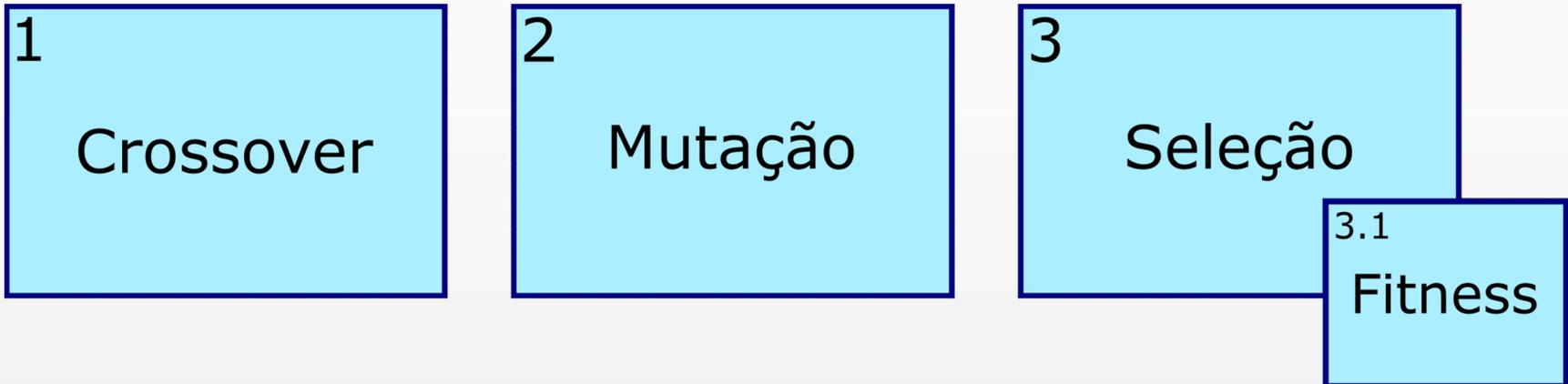
# Algoritmo Genético

- Geração



# Algoritmo Genético

- Operadores



# Crossover

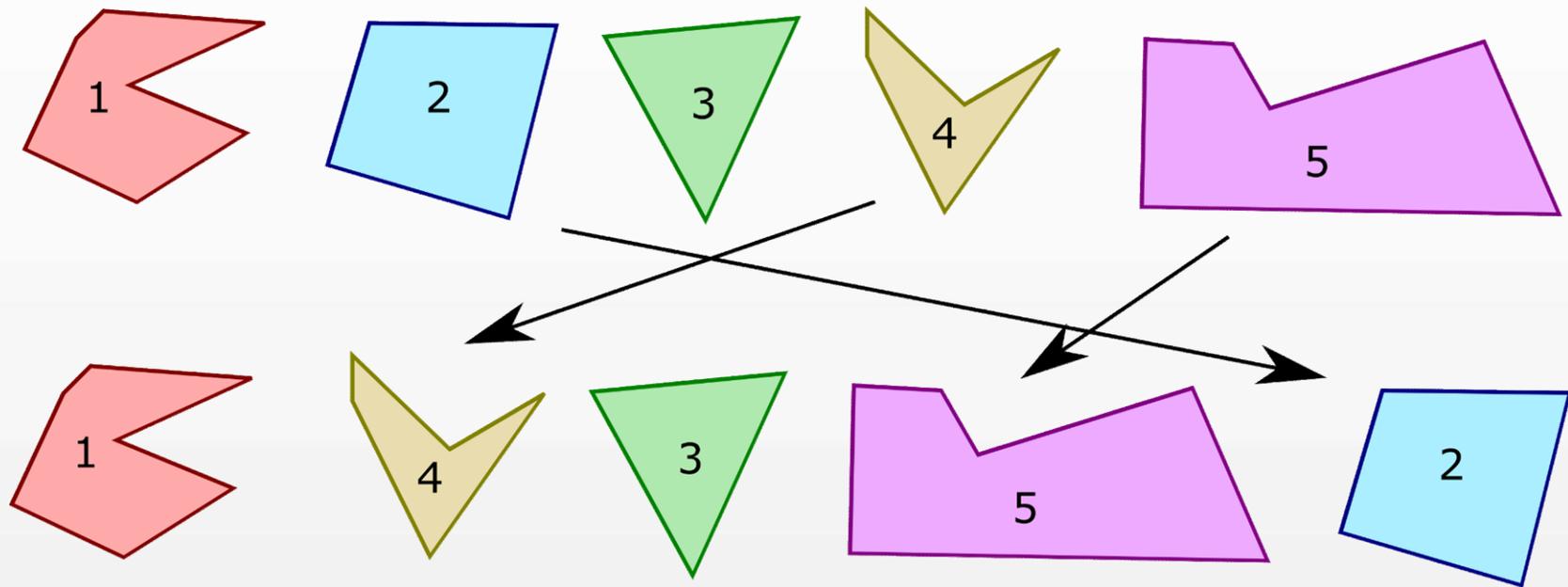
- Partial Matched Crossover (PMX).
- Permutação.
- Indivíduos inválidos.

Pontos fixos

a)	Pai A:	4	8	7	3	6	5	1	0	9	2
	Pai B:	3	1	4	2	7	9	0	8	6	5
b)	Pai A:	4	8	7	2	7	9	1	0	9	2
	Pai B:	3	1	4	3	6	5	0	8	6	5
c)	Pai A:	4	8	7	2	7	9	1	0	9	2
	Pai B:	3	1	4	3	6	5	0	8	6	5
d)	Filho A:	4	8	6	2	7	9	1	0	5	3
	Filho B:	2	1	4	3	6	5	0	8	7	9

# Mutação

- Swap Mutator

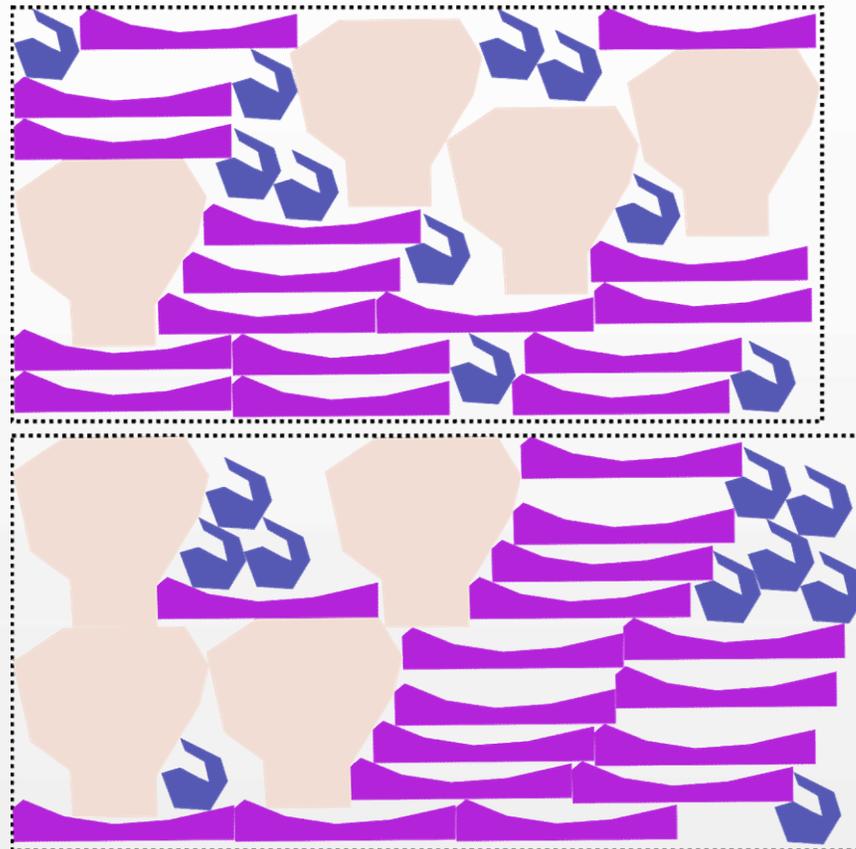


# Seleção e Fitness

- Recebe uma ordem de empacotamento
- Executa o Bottom-left fill, com NFP
- Avalia o indivíduo pela ocupação

# Seleção e Fitness

- Seleção e Fitness



# Entrada e Saída

- Arquivo SVG
- Editores vetoriais
- Edição manual
- Browsers
- Retroalimentação

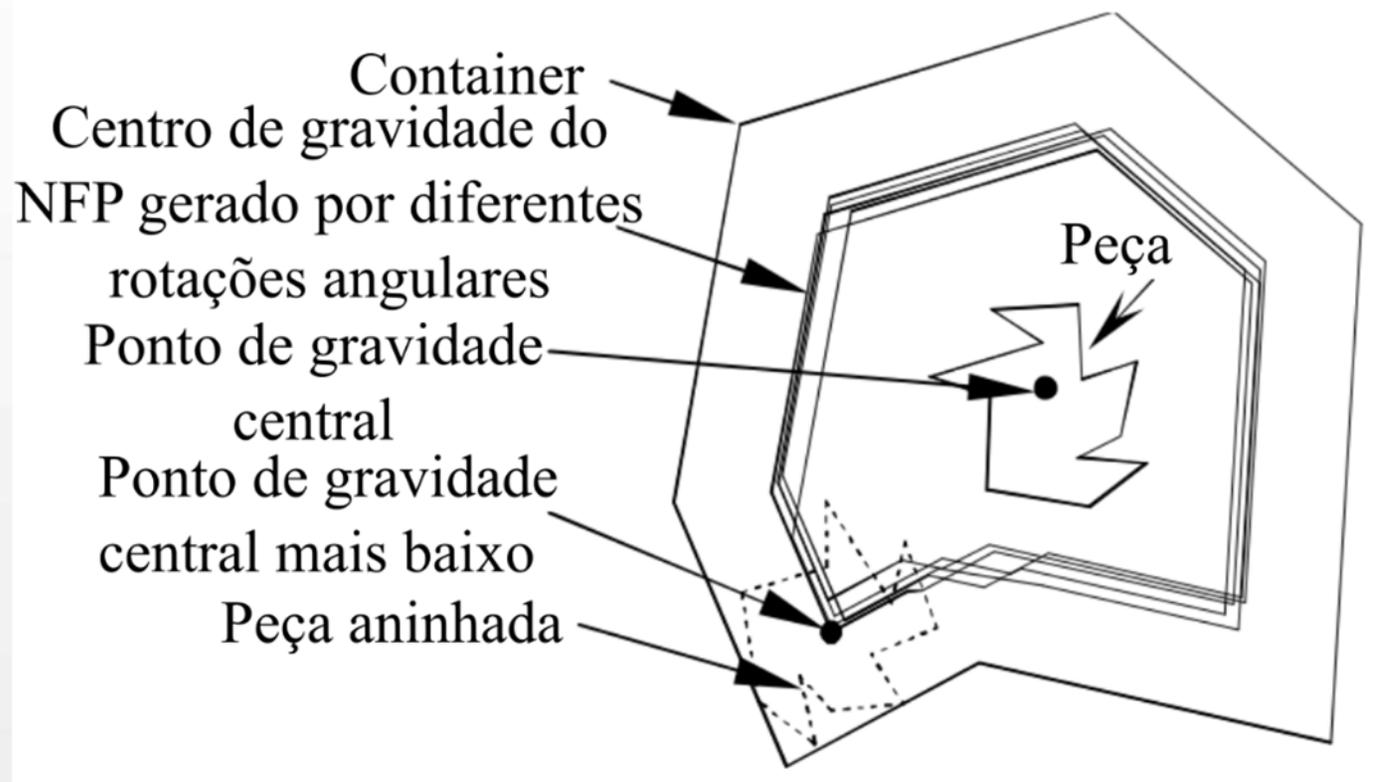


# Trabalhos Correlatos

- 3 trabalhos correlatos:
  - Haiming, Jiong e Xinsheng (2006)
  - Junior, Pinheiro e Saraiva (2013)
  - Brandt (2011)

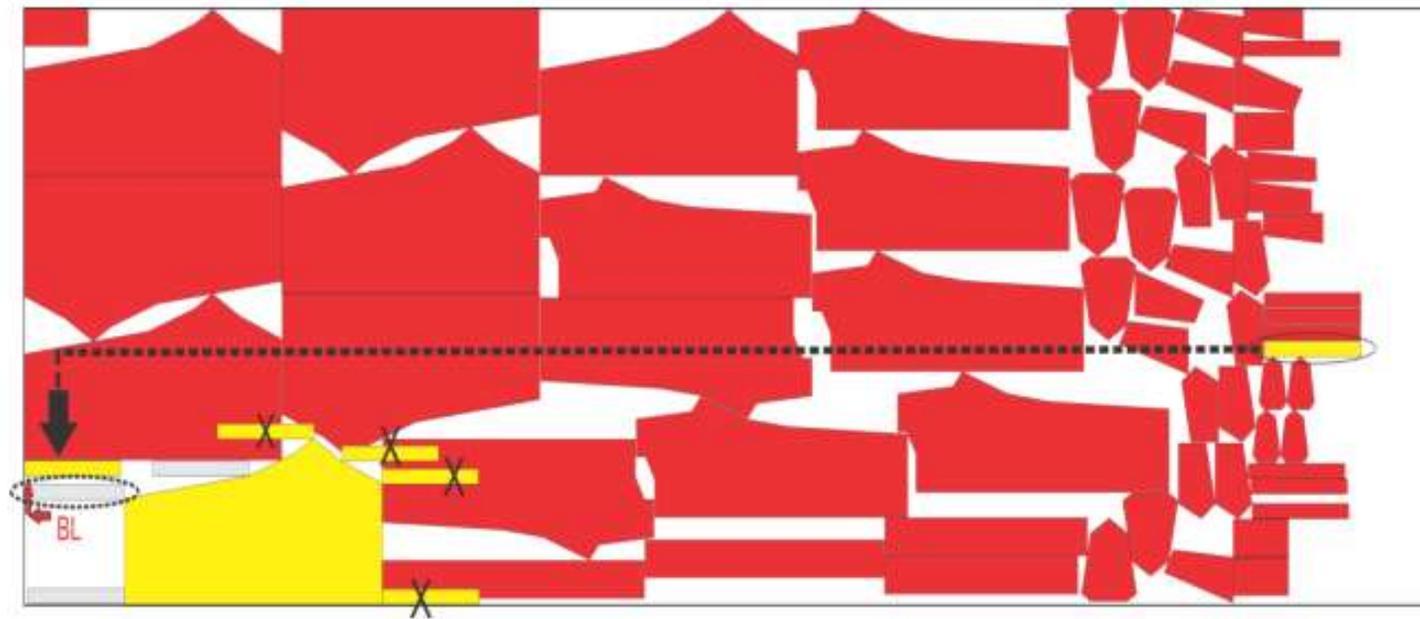
# Haiming, Jiong e Xinsheng (2006)

- NFP, low gravity center
- Algoritmo Genético



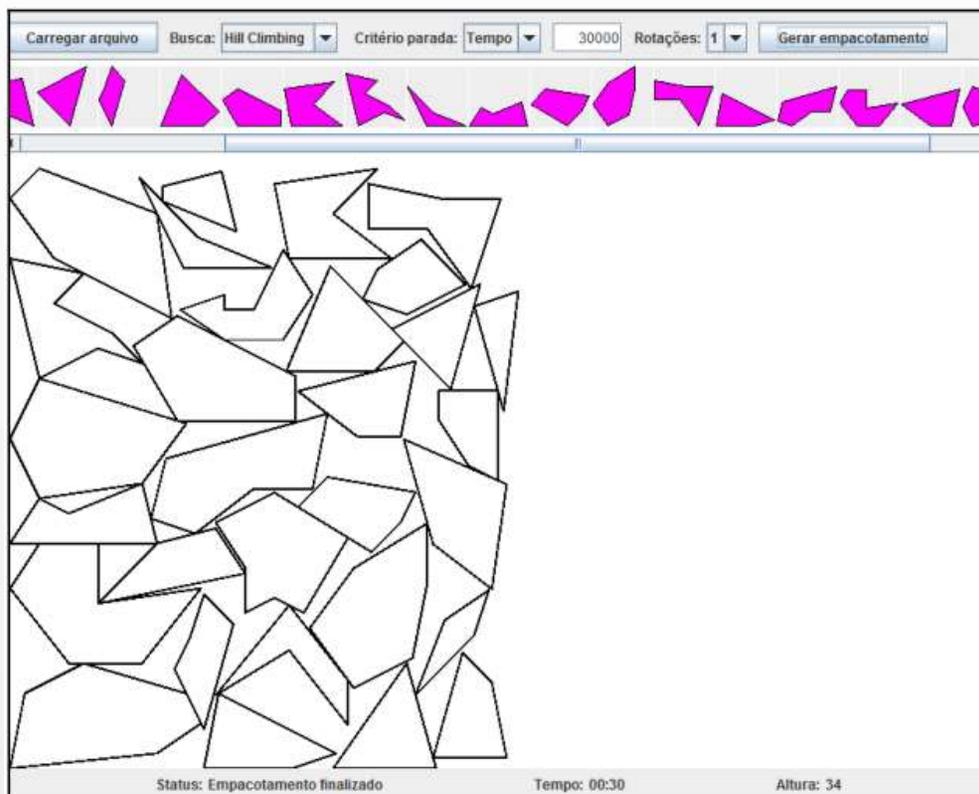
# Junior, Pinheiro e Saraiva (2013)

- NFP, bottom-left fill
- Algoritmo Genético
- Encolhimento



# Brandt (2011)

- NFP, bottom-left fill
- Hill Climbing e Tabu Search



# Roteiro

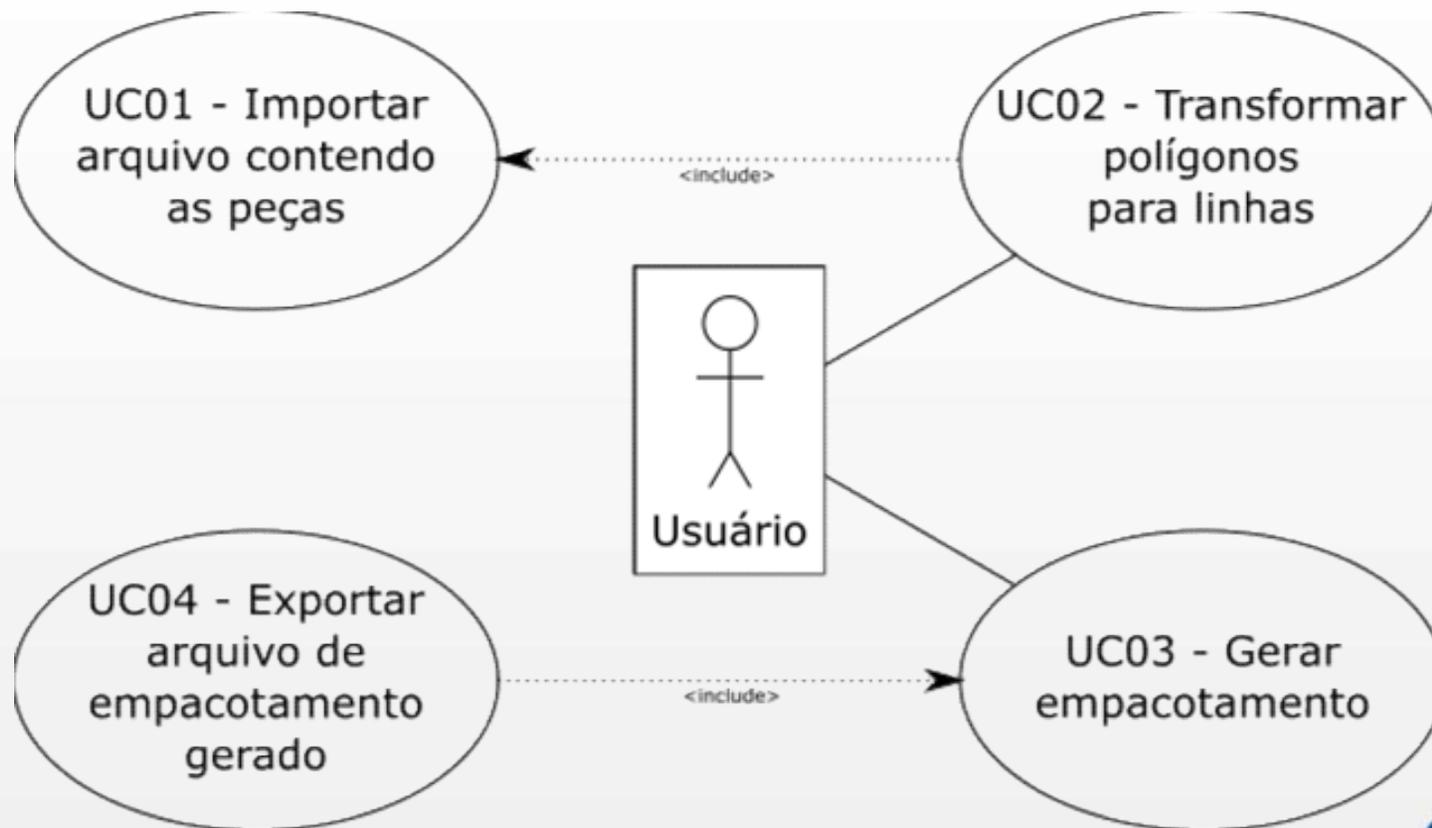
- **Introdução**
  - Objetivos
- **Fundamentação Teórica**
  - Requisitos
  - Trabalhos Correlatos
- **Desenvolvimento**
  - Especificação
  - Operacionalidade
- **Resultados**
  - Discussões
  - Comparação
- **Conclusões**
  - Sugestões

# Requisitos

- RF01 - Parâmetros de linha de comando
- RF02 - Obedecer altura do material
- RF03 - Informações sobre altura e tempo
- RF04 - Não sobrepor polígonos (falha biblioteca)
- RF05 - Bottom-left fill com NFP
- RF06 - Exportar/importar no formato SVG
- RF07 - Parar por tempo
- RNF01 - Obter resultado em menos de 20 min.

# Especificação

- Casos de uso



# Especificação

- Atividades principais

0 Transformação segmentos de reta / Multiplicação

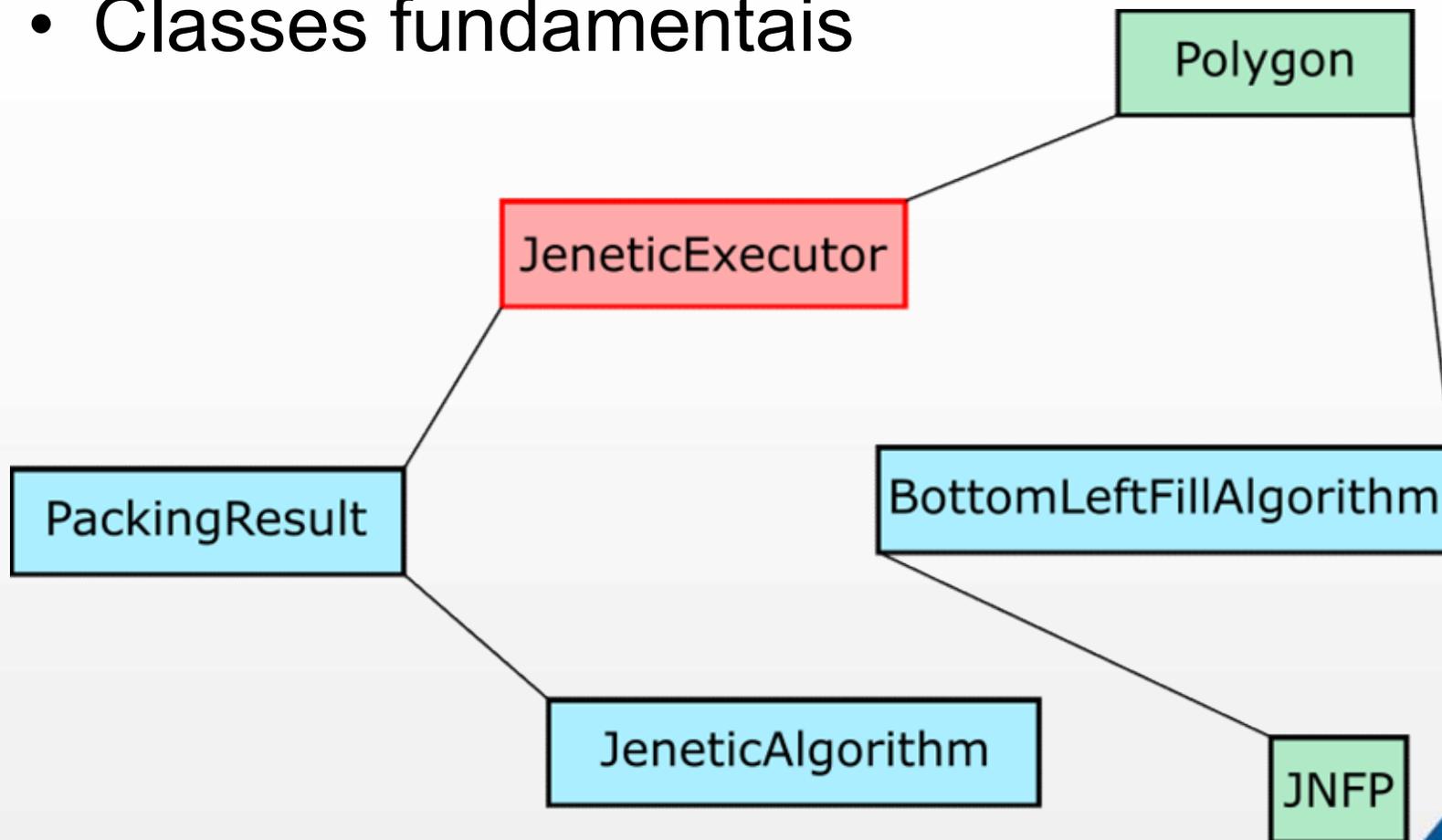
1 Leitura  
do  
SVG  
e  
Extração  
dos  
Polygonos

2  
**Algoritmo Genético**

3 Escrita  
do  
SVG  
com  
melhor  
resultado

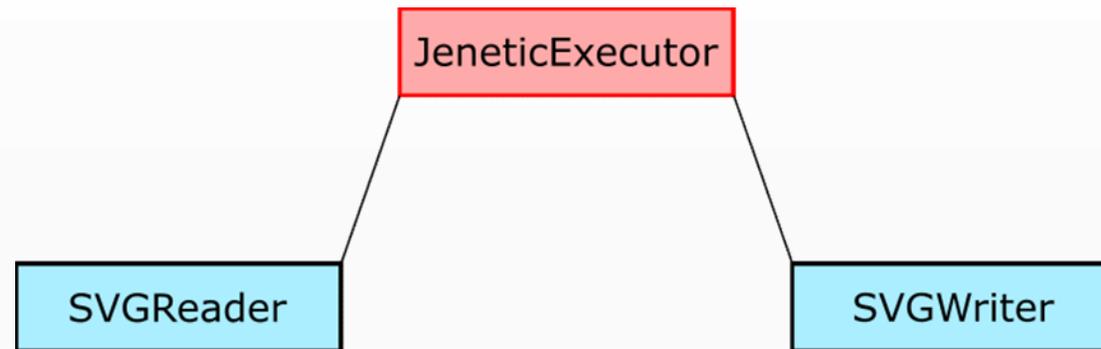
# Especificação

- Classes fundamentais

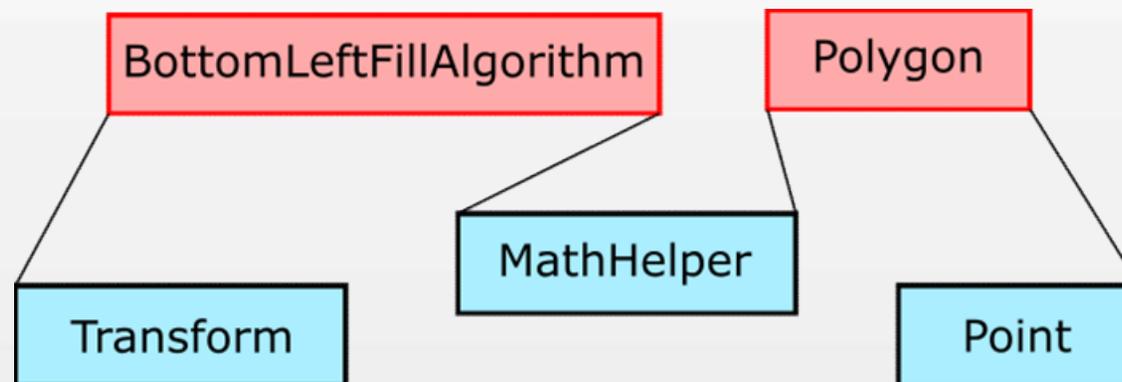


# Especificação

- Classes leitura e escrita do arquivo SVG



- Classes utilitárias



# Desenvolvimento

- **Linguagens/IDE**
  - Eclipse
  - Java 8
  - Python 3.5
- **Brandt (2011)**
  - Funções matemáticas
  - Bottom-left fill
- **JNFP**
  - No-fit Polygon
- **Jenetics**
  - Algoritmo Genético
- **svg.path**
  - Manipulação SVG

0 Transformação segmentos de reta / Multiplicação

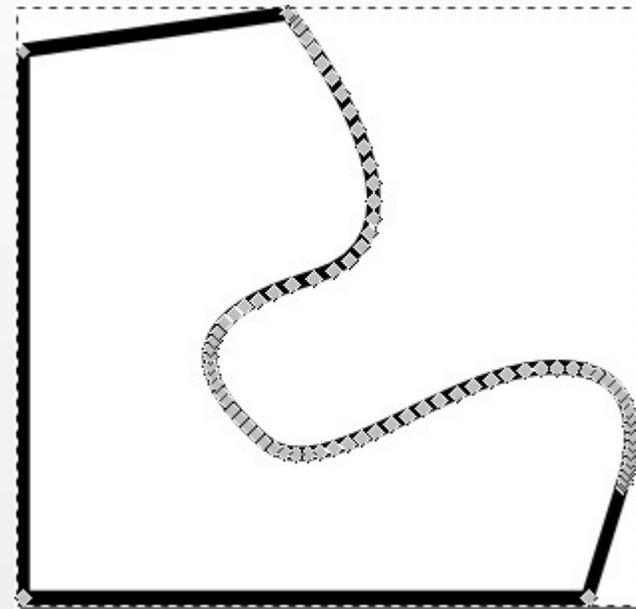
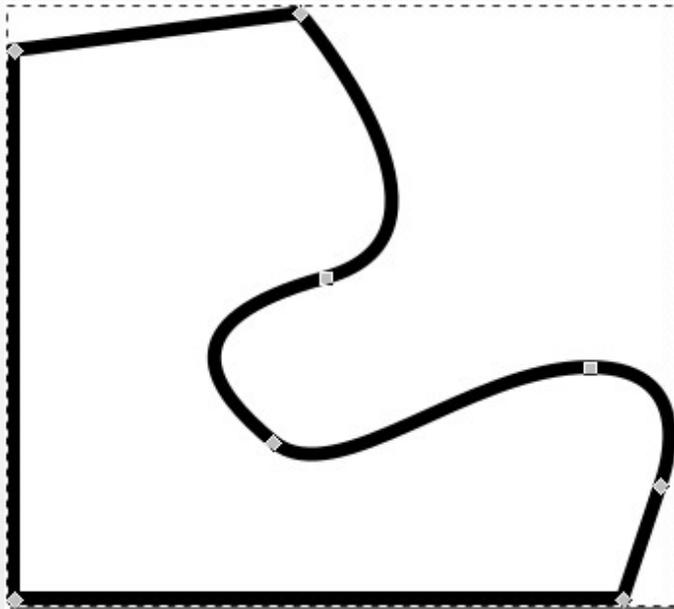
1 Leitura  
do  
SVG  
e  
Extração  
dos  
Polygonos

2  
**Algoritmo Genético**

3 Escrita  
do  
SVG  
com  
melhor  
resultado

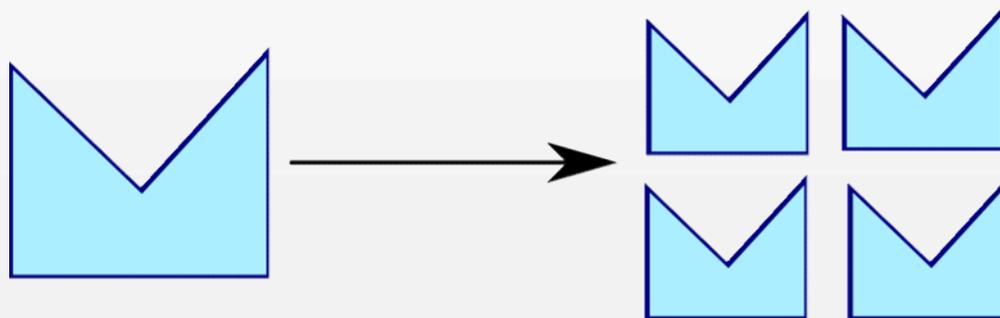
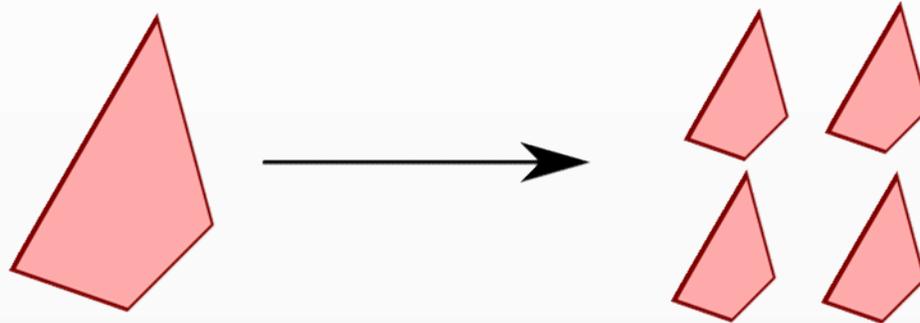
# Segmentos de reta

- Script Python para conversão
- Transforma (curvas, verticais, horizontais) para linha.



# Multiplicação

- Multiplica todos as tags <path>



0 Transformação segmentos de reta / Multiplicação

1 Leitura  
do  
SVG  
e  
Extração  
dos  
Polygonos

2  
**Algoritmo Genético**

3  
Escrita  
do  
SVG  
com  
melhor  
resultado

# Leitura e Extração

- Leitura de todas tags <path>
- Limpeza dos comandos de desenho
- Somente as coordenadas X e Y

```
d="M 147.879,31.2569 L 147.879,41.8171 L  
159.651,41.8171 L 160.369,39.662 L  
160.369,39.662 L 160.376,39.6453 L  
160.393,39.5976 L 160.418,39.5223 L  
160.448,39.4229 L 160.479,39.3028 L  
160.509,39.1654 L 160.534,39.0143 L  
160.551,38.8528 L 160.557,38.6844 L  
160.549,38.5126 L 160.523,38.3407 L  
160.478,38.1724 L Z" />
```

0 Transformação segmentos de reta / Multiplicação

1 Leitura  
do  
SVG  
e  
Extração  
dos  
Polygonos

2  
**Algoritmo Genético**

3 Escrita  
do  
SVG  
com  
melhor  
resultado

# Algoritmo Genético

- Otimização
  - Idade máxima
  - Tamanho da população
- Alteradores
  - SwapMutator
  - PartiallyMatchedCrossover
- Limitadores
  - Melhor estável
  - Por tempo
  - Quantidade de gerações

0 Transformação segmentos de reta / Multiplicação

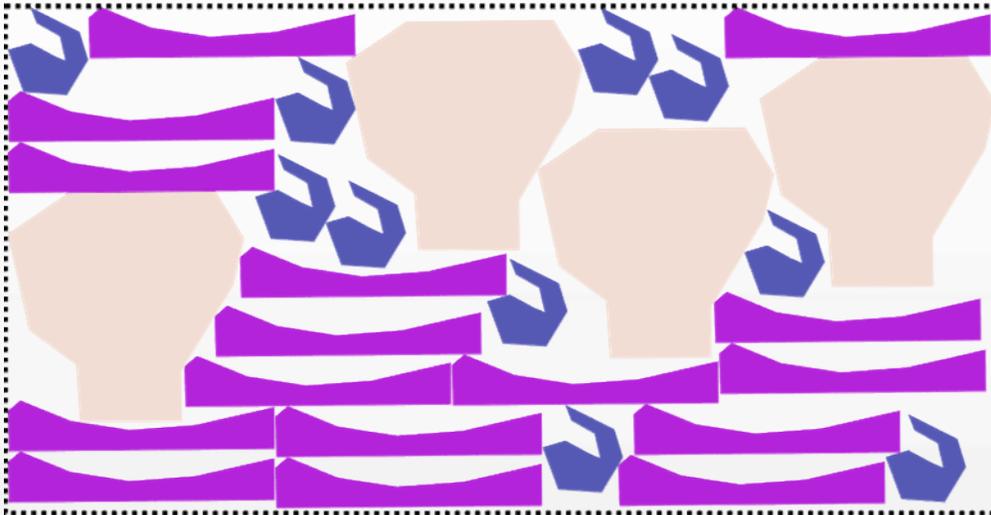
1 Leitura  
do  
SVG  
e  
Extração  
dos  
Polygonos

2  
**Algoritmo Genético**

3 Escrita  
do  
SVG  
com  
melhor  
resultado

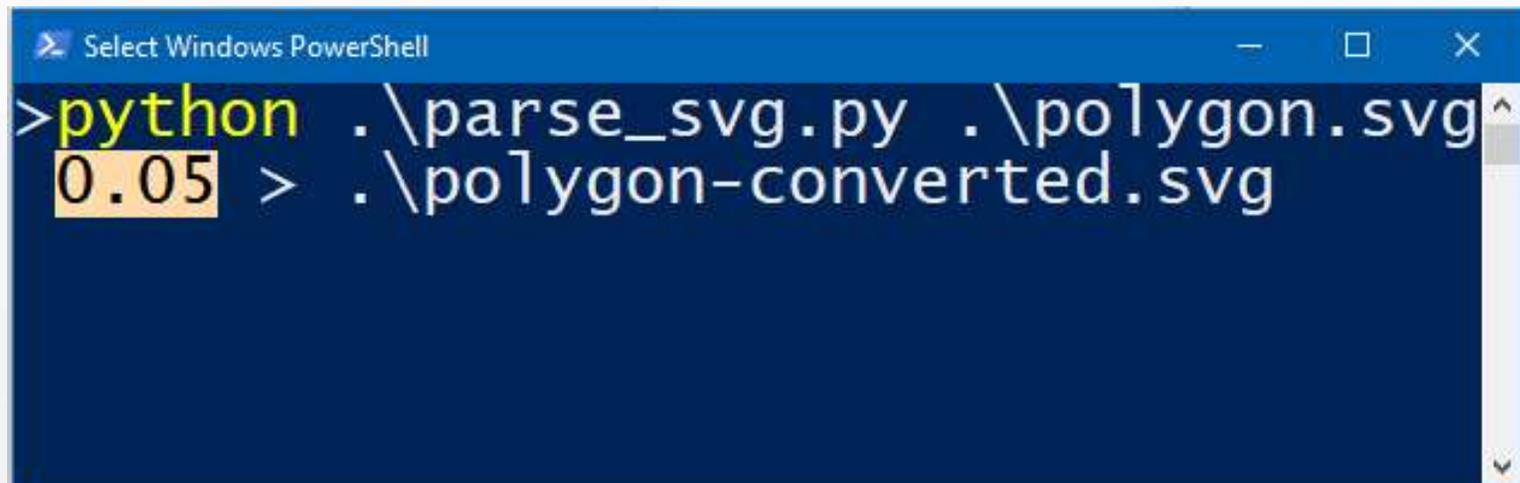
# Exportação

- Transforma o melhor resultado para SVG



# Operacionalidade

- Conversão para segmento de reta
  1. SVG-ENTRADA
  2. FATOR-PONTOS



```
Select Windows PowerShell
> python .\parse_svg.py .\polygon.svg
0.05 > .\polygon-converted.svg
```

# Operacionalidade

- Execução do empacotamento
  1. SVG-ENTRADA
  2. ALTURA
  3. ROTAÇÕES
  4. GERAÇÕES
  5. POPULAÇÃO
  6. TEMPO

```
Windows PowerShell
> java -cp "packing.jar;jenetics.jar;cc.jar"
JeneticExecutor alma4 100 1 200 200 1200000
```

# Roteiro

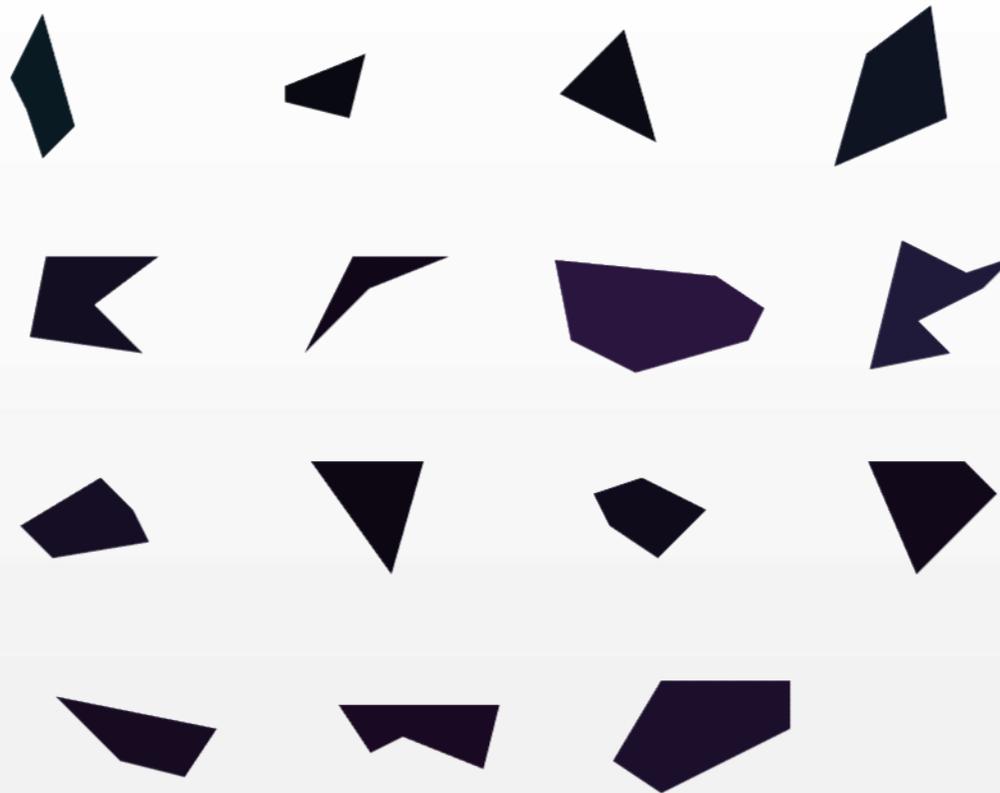
- **Introdução**
  - Objetivos
- **Fundamentação Teórica**
  - Requisitos
  - Trabalhos Correlatos
- **Desenvolvimento**
  - Especificação
  - Operacionalidade
- **Resultados**
  - Discussões
  - Comparação
- **Conclusões**
  - Sugestões

# Resultados

- Foram realizados 6 testes
- Datasets diferentes para o Teste 6
- Mostrar 3 testes principais
  1. **Gerações e populações**
  2. **Comparação Genético, Hill Climbing e Tabu Search**
  3. **Avaliação resultados JPacking e SVGNest**

# Datasets

poly1a - 15 polígonos



# Datasets

poly3b - 45 polígonos

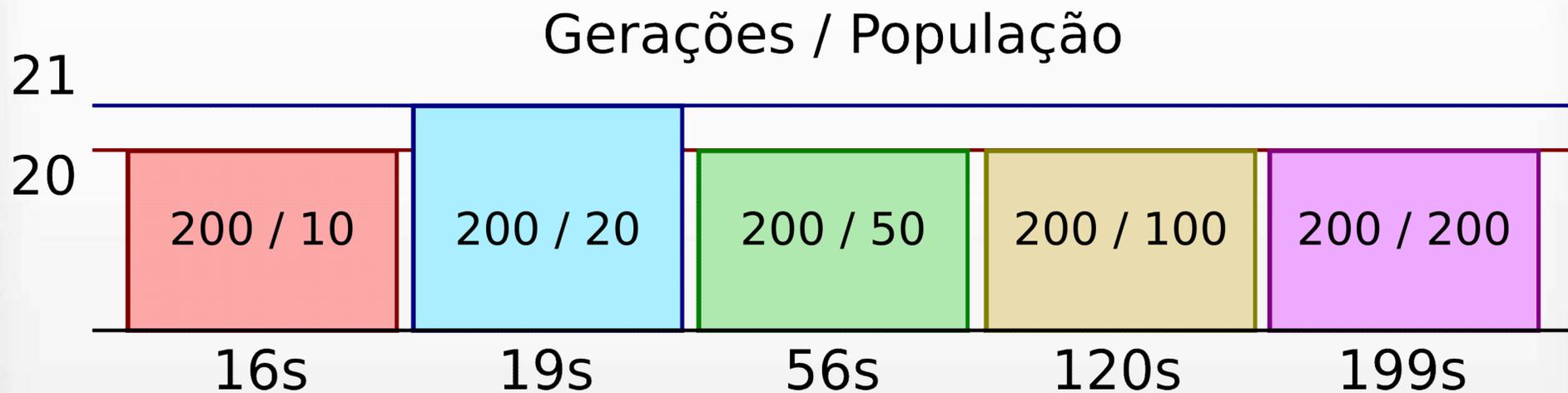


# Teste 1

- Gerações e populações
  - Mais gerações e mais população ajudam
  - Mais geração e população = demora
  - Calibração

# Teste 1

- Dataset poly3b



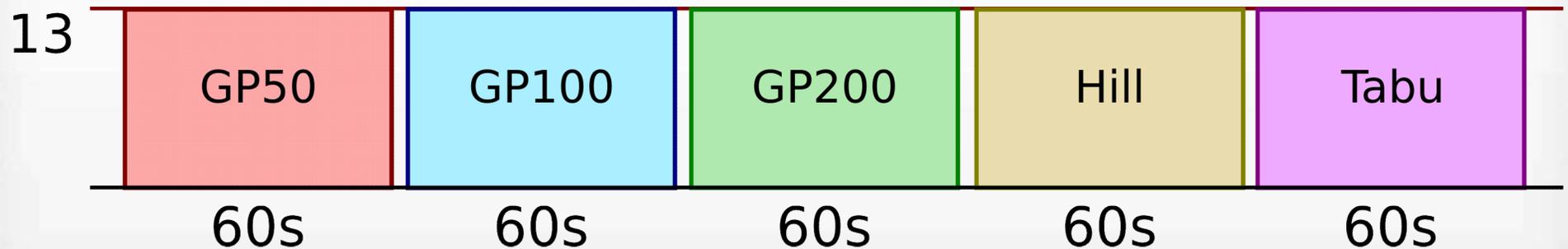
# Testes 2

- Comparação Genético, Hill Climbing e Tabu Search
  - Hill Climbing e Tabu Search foram melhores (**por muito pouco**)
  - Algoritmo Genético foi capaz igualar em alguns datasets
  - Calibração

# Testes 2

- Dataset poly1a execução 60 segundos

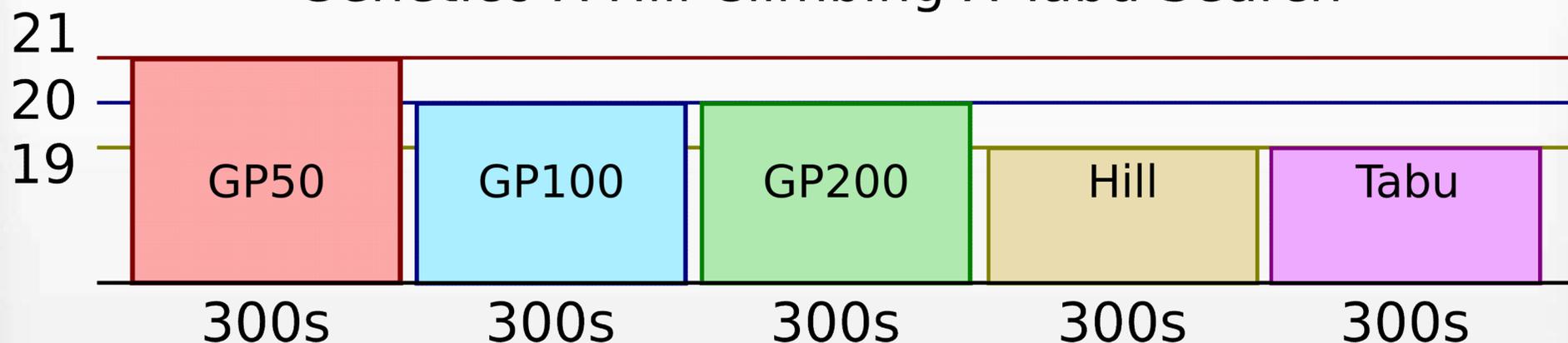
Genético X Hill Climbing X Tabu Search



# Testes 2

- Dataset poly3b execução 300 segundos

Genético X Hill Climbing X Tabu Search

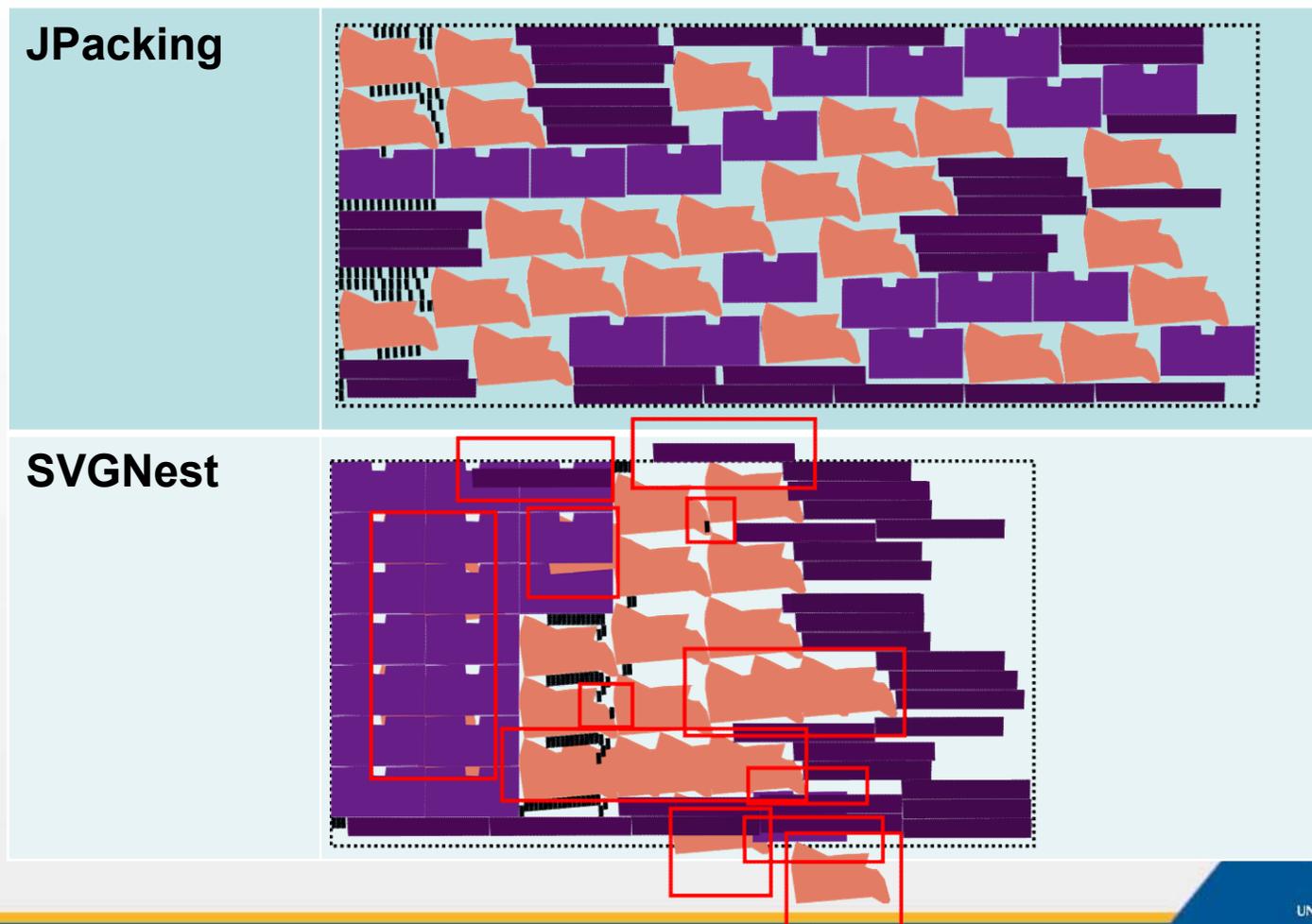


# Testes 3

- Comparação entre o JPacking e o programa SVGNest
  - Open Source,
  - Web
  - manipula SVG
- Localizado em: [svgnest.com](http://svgnest.com)

# Teste 3

- Comparação JPacking e SVGNest (alma1)



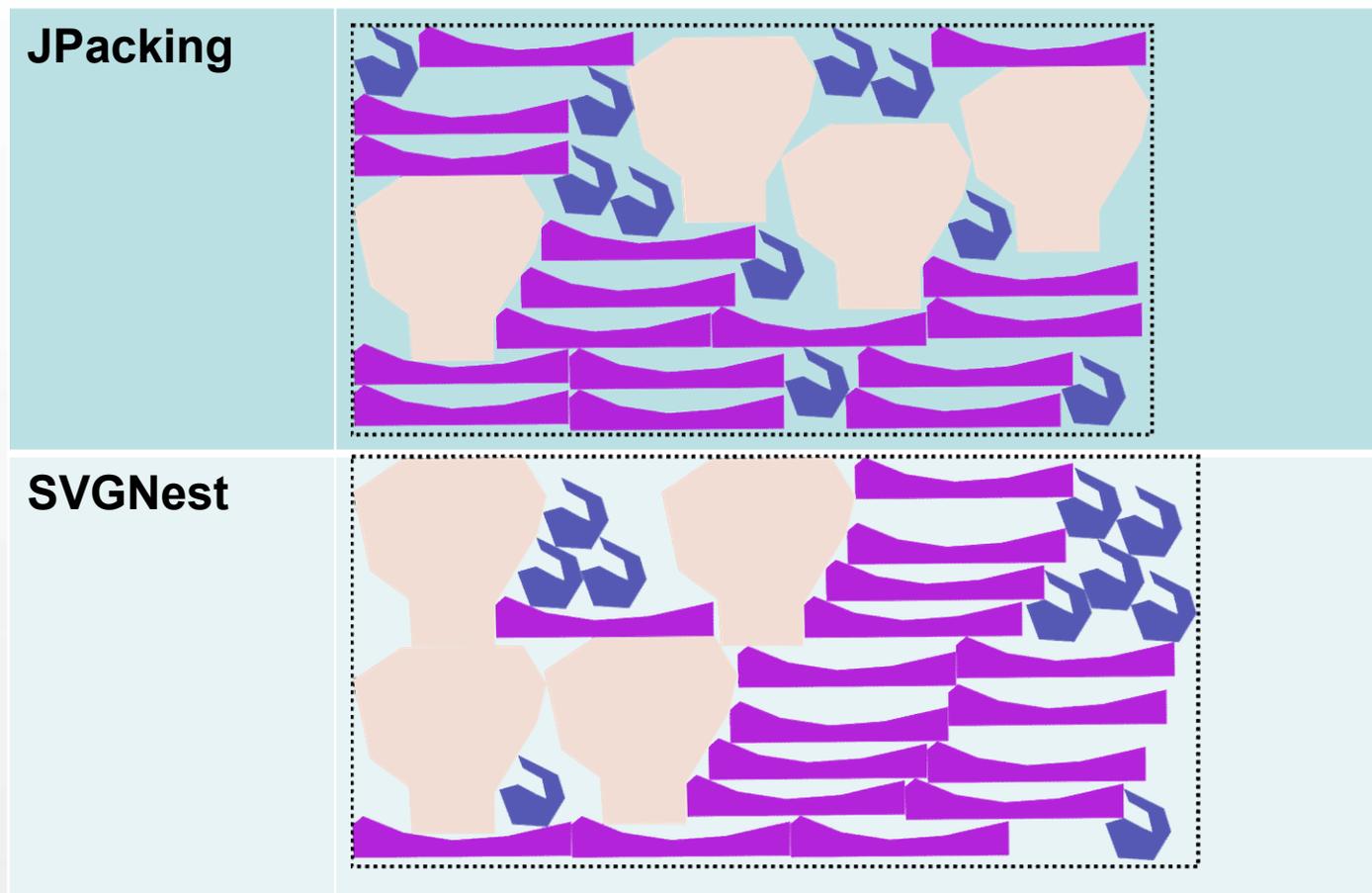
# Teste 3

- Comparação JPacking e SVGNest (alma4)



# Teste 3

- Comparação JPacking e SVGNest (roupas2)



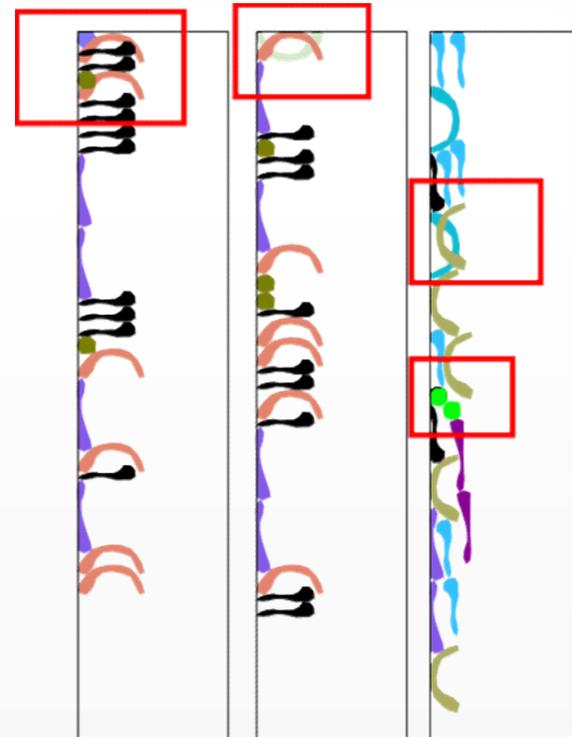
# Comparação

Funcionalidade	Haiming, Jiong e Xinsheng (2006)	Junior, Pinheiro e Saraiva (2013)	Brandt (2011)	Trabalho atual
Estratégia de encaixe	NFP, low gravity center	NFP, Bottom-left fill	NFP, Bottom-left fill	NFP, Bottom-left fill
Algoritmo	Genético	Genético	Hill Climbing e Tabu Search	Genético
Técnica adicional	-	Encolhimento	-	-
Interface Gráfica	-	-	Sim	Não

# Comparação

Funcionalidade	Haiming, Jiong e Xinsheng (2006)	Junior, Pinheiro e Saraiva (2013)	Brandt (2011)	Trabalho atual
Interface linha de comando	-	-	Não	Sim
Alteração manual	-	-	Sim	Sim
Arquivo de entrada	-	-	XML	SVG
Arquivo de saída	não especificado	não especificado	Não tem	SVG

# Discussões



- Erros de empacotamento
  - Foi investigado
  - Verificado ordem dos vértices
  - Não foi descoberto
  - Problemas JNFP
- Datasets maiores (> 150 peças)
  - Jenetics limite de 43330 genes por individuo
  - Bottom-left fill lento, problema do deslocamento

# Roteiro

- **Introdução**
  - Objetivos
- **Fundamentação Teórica**
  - Requisitos
  - Trabalhos Correlatos
- **Desenvolvimento**
  - Especificação
  - Operacionalidade
- **Resultados**
  - Discussões
  - Comparação
- **Conclusões**
  - Sugestões

# Conclusões

- O JPacking é capaz de realizar o empacotamento
- Não existem parâmetros fixos, testes e calibração ajudam
- O Hill Climbing e o Tabu Search são bons algoritmos, contudo, Genético permite maior calibração
- JPacking não é totalmente perfeito.
  - Largura infinita
  - Erros de empacotamento

# Sugestões

1. Reimplementar o algoritmo de Bottom-left fill
2. Algoritmo para avaliar a rotação
3. Espaço entre os polígonos
4. Explorar limitações de formato de forma
5. Plugin para o Inkscape

# DEMONSTRAÇÃO

# **JPACKING: PROGRAMA PARA DISTRIBUIÇÃO OTIMIZADA DE POLÍGONOS EM UM PLANO BIDIMENSIONAL UTILIZANDO ALGORITMOS GENÉTICOS**

Aluno: Rodrigo D'avila

Orientador: Prof. Daniel Theisges  
dos Santos, Mestre