Otimização do Layout de Estaleiros: Um Estudo Comparativo

Frederico Gomes Pires Azzolini

Sumário

Introdução

- Problema universal: otimizar a produção
- Ou seja: Gastar menos e produzir mais

- Como?
- Depende

Introdução - Estaleiros

- Produtos de grande porte
  - Muito consumo de materiais
  - Alto custo de manejo e transporte
- Tendem a ser construídos sem nenhum processo sistemático

Introdução - Exemplos

- Minimizar o tempo do ciclo de produção utilizando Algoritmo Genético, Azadivar e Wang (2000)
- Maximizar os requerimentos de distância, adjacência e proporção de área entre os departamentos, Aiello et al. (2006)

Introdução

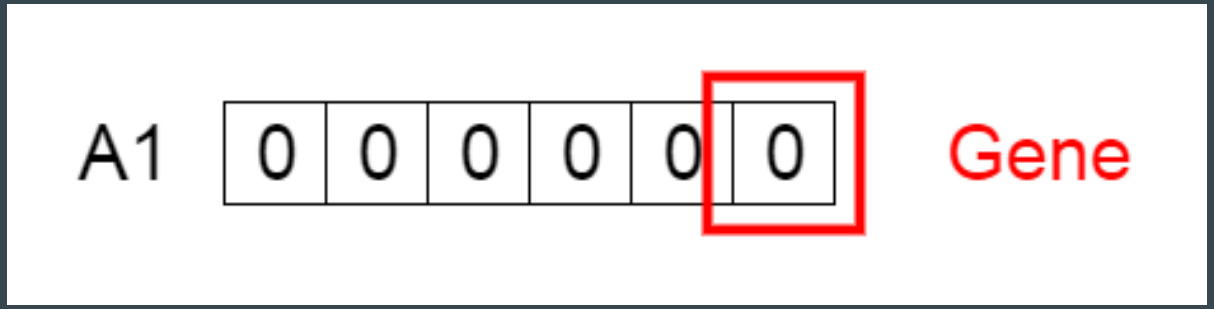
- Choi, Kim e Chung (2017)
- Sistema de duas etapas
  - Algoritmo Genético para determinar melhor topologia
  - Heurística de crescimento para determinar melhores dimensões para os departamentos

Fundamentação Teórica - Algoritmo Genético

- Inicialmente propostos por Holland (1992)
- Equilibram dois objetivos conflitantes (Zuben, 2000)
  - Exploração do espaço em busca de soluções
  - Aproveitamento das melhores soluções encontradas

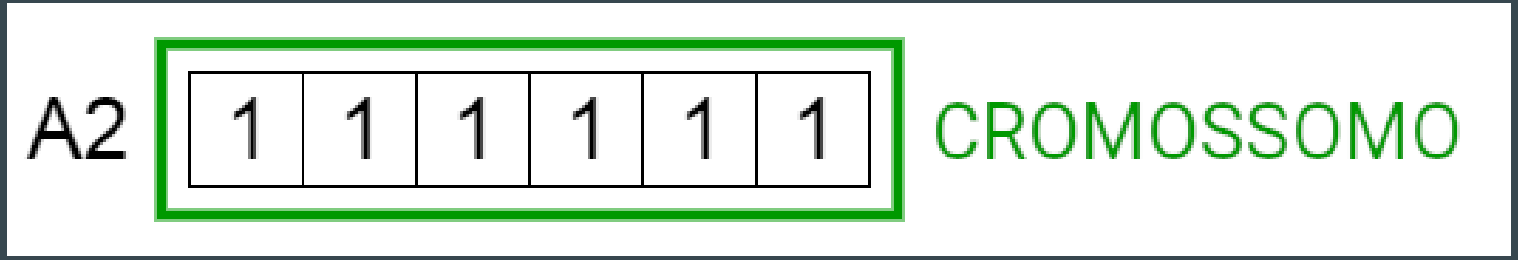
Fundamentação Teórica - Algoritmo Genético

- Gene



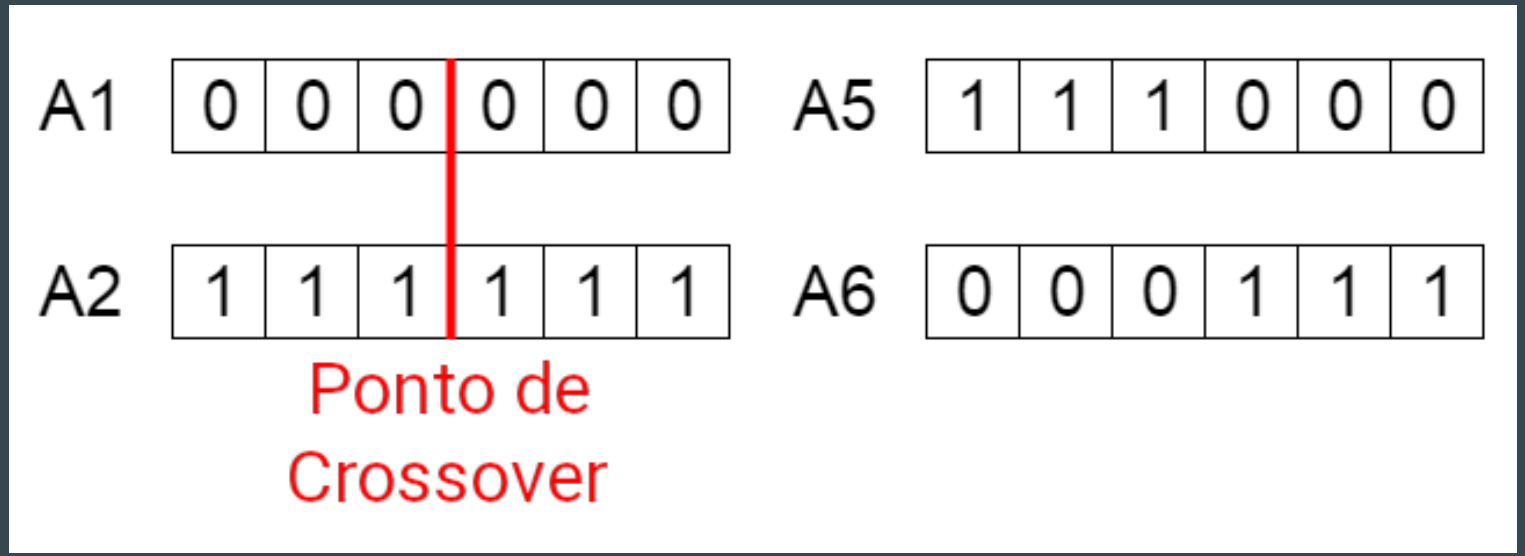
Fundamentação Teórica - Algoritmo Genético

- Cromossomo



Fundamentação Teórica - Algoritmo Genético

- Crossover (ou Recombinação)



Fundamentação Teórica - Algoritmo Genético

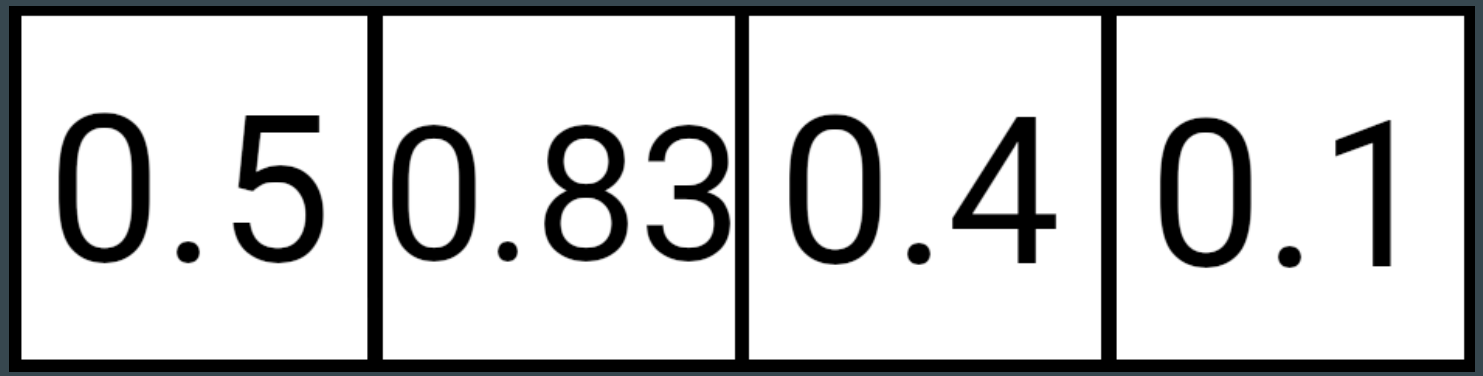
- Mutação

Fundamentação Teórica - BRKGA

- Biased Random-Key Genetic Algorithm citado inicialmente por Gonçalves e Resende (2011)
- Utiliza soluções codificadas
  - Cromossomo -> Decoder -> Solução Viável

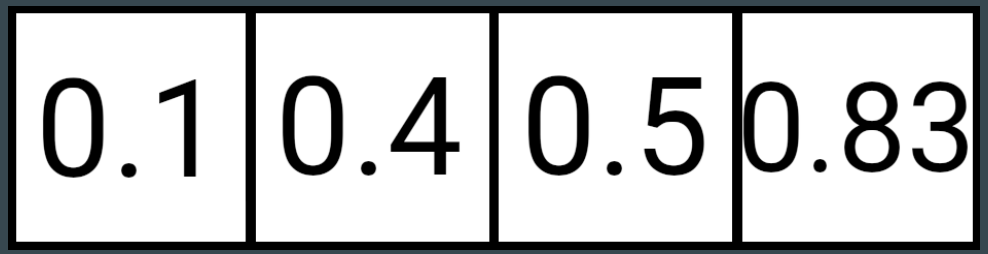
Fundamentação Teórica - BRKGA

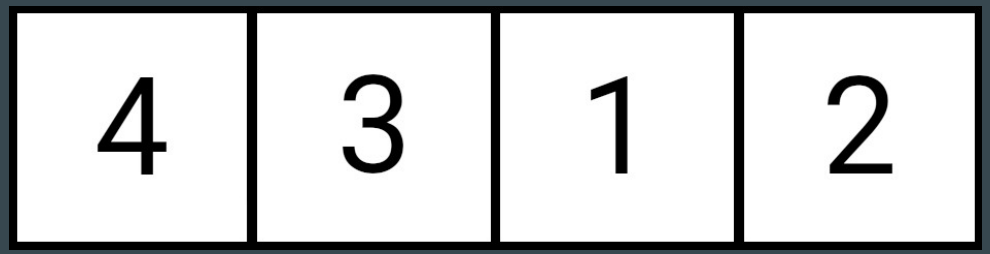
- Cromossomo



Fundamentação Teórica - BRKGA

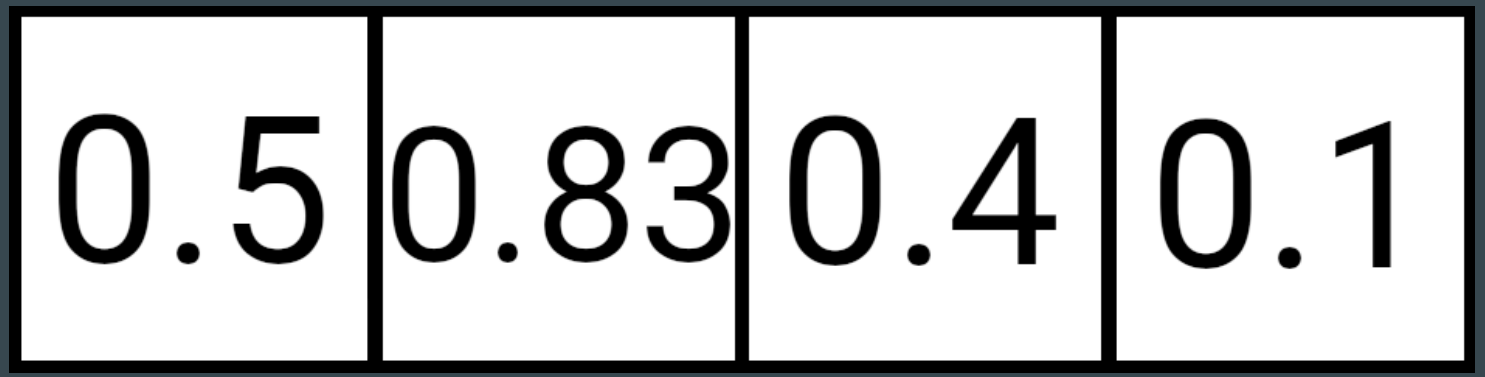
- Decoder



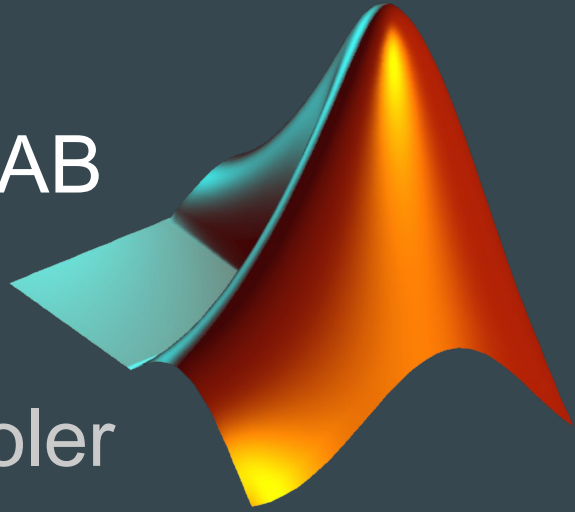


Fundamentação Teórica - BRKGA

- Mutação



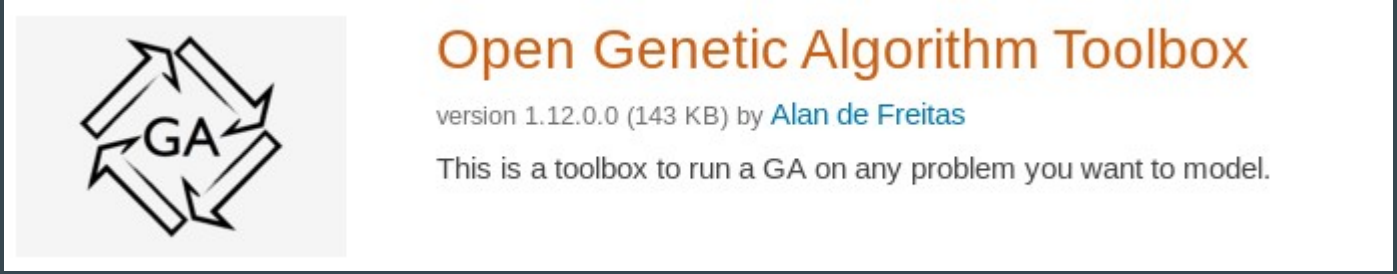
Ferramentas e Tecnologias - MATLAB



- Matrix Laboratory
- Criado nos anos 70 por Cleve Moler
- Utiliza linguagem própria (M-code), assim como C++, C# e Java

Ferramentas e Tecnologias - GA_framework

- Estrutura genérica para Algoritmos Genéticos
- Altamente adaptável



Método - Função Objetivo

- Mesma utilizada por Choi, Kim e Chung (2017)

$$Mininize \left[\sum_{i \in D} \sum_{j \in D} f_{i,j}^{tp} \cdot c_{i,j}^{tp} \cdot d_{i,j}^{tv} \right]$$

Método - Algoritmo Genético

- Criação de Indivíduos
  - Vetor Ordenado sem Repetições
  - Função Recursiva
  - Busca manter o fator aleatório sem desrespeitar as restrições

```
function [ solution ] = estaleiro_generate_random( problem, ~ )
  array = randperm(problem.n_var, problem.n_var);
  it = 0;
  align_departments = [];
  solution = zeros(1, problem.n_var);
  if not(isempty(problem.constraint))
    for k=1:length(problem.constraint)
      array(array == problem.constraint(k, 1)) = [];
      array(array == problem.constraint(k, 2)) = [];
    end
  end
  if not(isempty(problem.fixedPos))
    for k = 1:length(problem.fixedPos)
      array(array == problem.fixedPos(k).dept) = [];
      position = ((problem.fixedPos(k).i - 1) * problem.width) + problem.fixedPos(k).j;
      solution(position) = problem.fixedPos(k).dept;
    end
  end
  if not(isempty(problem.constraint))
    for k=1:length(problem.constraint)
      if isempty(align_departments)
        align_departments = [align_departments problem.constraint(k, 1)];
        align_departments = [align_departments problem.constraint(k, 2)];
      else
        if not(ismember(problem.constraint(k, 1), align_departments))
          align_departments = [align_departments problem.constraint(k, 1)];
        end
        if not(ismember(problem.constraint(k, 2), align_departments))
          align_departments = [align_departments problem.constraint(k, 2)];
        end
      end
    end
    [success, solution] = generate(align_departments, problem, solution);
  end
  for i = 1:length(solution)
    if solution(i) == 0
      solution(i) = array(1);
      array(1) = [];
    end
  end
end
```

Método - Algoritmo Genético

- Crossover
  - Crossover em Ciclo
  - Respeitar a ordenação e impedir repetições

```
function child = estaleiro_crossover_cycle(parents,~, ~, population)
    n_parents = length(parents);
    randparents = randperm(n_parents);
    parent1 = population.ind{parents(randparents(1))};
    parent2 = population.ind{parents(randparents(2))};
    child = zeros(1,length(parent1));
    pos_parent1 = randi(n_parents);
    aux = []; first = [];
    first = parent1(pos_parent1);
      while true
        child(pos_parent1) = parent1(pos_parent1);
        if parent1(pos_parent1) == first
          break;
        end
        for i=1:length(parent1)
          if parent1(i) == parent2(pos_parent1)
            pos_parent1 = i;
          end
        end
      end
      for i=1:length(child)
        parent2(parent2 == child(i)) = [];
      end
      for i=1:length(child)
        if child(i) == 0
          child(i) = parent2(1);
          parent2(1) = [];
        end
      end
  end
```

Método - Algoritmo Genético

- Mutação
  - Mutação BitFlop
  - Respeitar a ordenação e impedir repetições

```
function x = estaleiro_mutation_bitflop(x, problem,~, ~)
    pos = [];
    pos(1) = randi(length(x));
    pos(2) = randi(length(x));
    if not(isempty(problem.fixedPos))
      ok = false;
      while not(ok)
        ok = true;
        for k = 1:length(problem.fixedPos)
          if problem.fixedPos(k).dept == x(pos(1))
            pos(1) = randi(length(x));
            ok = false;
          elseif problem.fixedPos(k).dept == x(pos(2))
            pos(2) = randi(length(x));
            ok = false;
          end
        end
      end
    end
    aux = x(pos(1));
    x(pos(1)) = x(pos(2));
    x(pos(2)) = aux;
  end
```

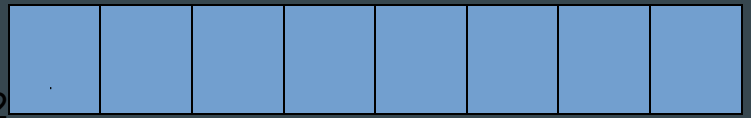
Método - BRKGA

- Criação de Indivíduos
  - Departamento livre:
    - Valor aleatório
  - Departamento com restrição de adjacência:
    - Calcular peso adequado para posição desejada
  - Departamento com restrição de posição:
    - Valor = -1

Método - BRKGA

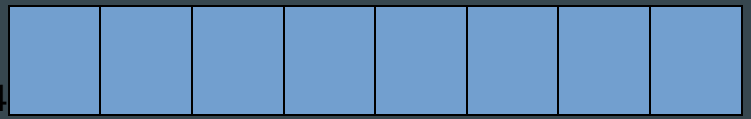
1

2



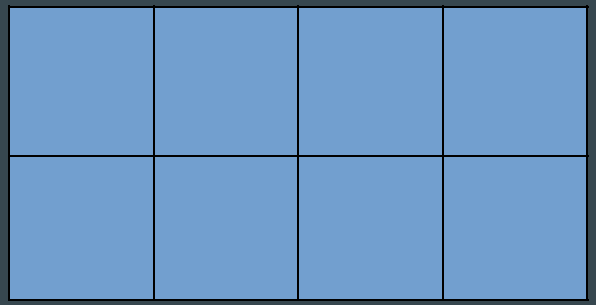
3

4



5

6



Método - BRKGA

1

	0,1		0,4		0,8	0,9	

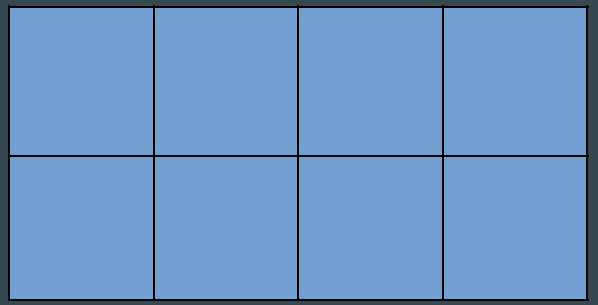
2

3

4

5

6



Método - BRKGA

1

	0,1	-1	0,4		0,8	0,9	-1

2

3

			8				3

4

5

6

			8
			3

Método - BRKGA

1

	0,1	-1	0,4		0,8	0,9	-1

2

3

			8				3

4

5

6

5	1		8
			3

Método - BRKGA

1

	0,1	-1	0,4		0,8	0,9	-1

2

3

5	1		8				3

4

5

6

5	1		8
			3

Método - BRKGA

1

	0,1	-1	0,4		0,8	0,9	-1

2

3

5	1	2	8	4	6	7	3

4

5

6

5	1		8
			3

Método - BRKGA

1

	0,1	-1	0,4		0,8	0,9	-1

2

3

5	1	2	8	4	6	7	3

4

5

6

5	1	2	8
4	6	7	3

Método - BRKGA

1

0,075	0,1	-1	0,4	0,05	0,8	0,9	-1

2

5	1	2	8	4	6	7	3

3

5	1	2	8
4	6	7	3

4

Método - BRKGA

- Crossover
  - Crossover Uniforme Viciado
  - Com alterações de modo a respeitar as restrições

Método - BRKGA

0,05	0,1	-1	0,4	0,075	0,8	0,9	-1

0,765	0,93	-1	0,6	0,847	0,45	0,1	-1

Método - BRKGA

0,05	0,1	-1	0,4	0,075	0,8	0,9	-1

0,765	0,93	-1	0,6	0,847	0,45	0,1	-1

0,075	0,1	-1	0,4	0,075	0,8	0,9	-1

0,765	0,93	-1	0,6	0,847	0,45	0,1	-1

Método - BRKGA

0,05	0,1	-1	0,4	0,075	0,8	0,9	-1
0,765	0,93	-1	0,6	0,847	0,45	0,1	-1
0,075	0,1	-1	0,4	0,075	0,8	0,9	-1
0,765	0,93	-1	0,6	0,847	0,45	0,1	-1
0,765	0,1	-1	0,6	0,847	0,45	0,9	-1

Método - BRKGA

- Decoder
  - Algoritmo de ordenação crescente simples

1

0,765	0,1	-1	0,6	0,847	0,45	0,9	-1

2

0,1	0,45	-1	0,6	0,765	0,847	0,9	-1

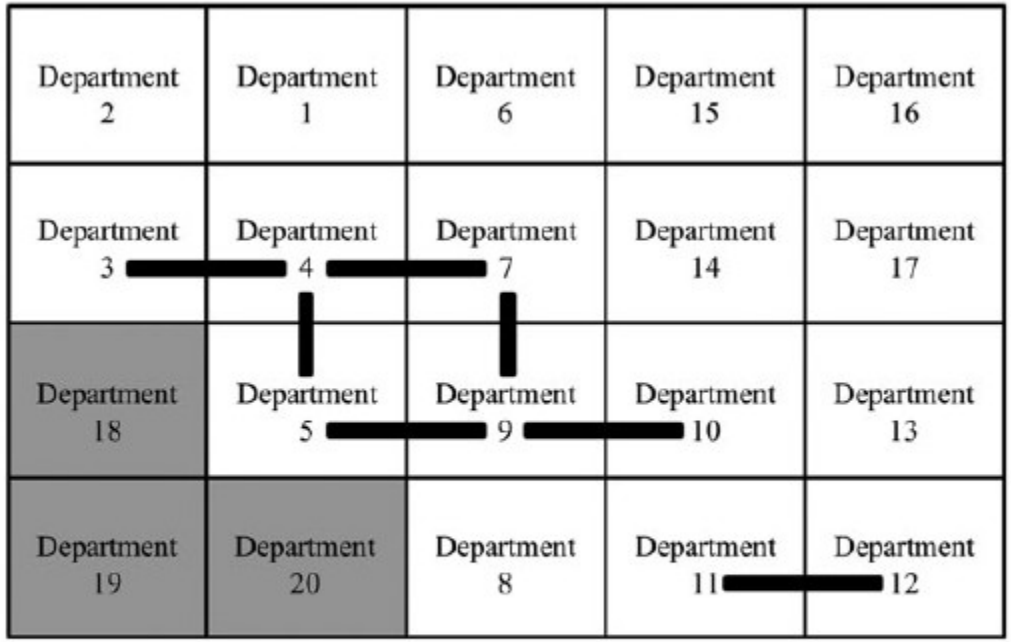
2	6	3	4	1	5	7	8

Método - Testes

- Cada código foi executado 1600 vezes
  - 500 indivíduos (400x)
  - 1000 indivíduos (400x)
  - 5000 indivíduos (400x)
  - 10000 indivíduos(400x)

Método - Testes

- Informações armazenadas:
  - Probabilidade de Crossover
  - Probabilidade de Mutação
  - Tempo Médio de Execução
  - Melhor Valor Obtido



19	20	8	11	12	18	5	9	10	13	3	4	7	14	17	2	1	6	15	16

Valor ótimo: 11.819 unidades

Resultados - GA

Tamanho da População	Probabilidade de *Crossover*	Probabilidade de Mutação	Tempo Médio de Execução (s)	Melhor Valor (u.m.)
500	0,9	0,05	1.922,8	11.077
1000	0,9	0,05	3.723,2	10.947
5000	0,9	0,05	16.364	10.947
10000	0,9	0,05	37.808	10.947

Resultados - BRKGA

Tamanho da População	Probabilidade de *Crossover*	Probabilidade de Mutação	Tempo Médio de Execução (s)	Melhor Valor (u.m.)
500	0,9	0,05	1.989,4	11.882
1000	0,9	0,05	2.030	10.947
5000	0,9	0,05	24.607	10.947
10000	0,9	0,05	29.066	10.947

Resultado - GA (1000 indivíduos)

1	12	14	15	16
2	11	10	8	17
18	13	9	5	6
19	20	7	4	3

Resultado - BRKGA (1000 indivíduos)

1	12	14	15	16
6	11	10	8	17
18	13	9	5	2
19	20	7	4	3

Conclusão