# Desenvolvendo AEs

Conceitos e Aplicações

# Sumário

- Principais aspectos
- Representação e Operadores Clássicos.
- Fitness
- Outras Representações, Operadores e Fitness
- Seleção
- Outros Operadores

## Principais aspectos

1. Definir a representação da solução
   a. Codificação
   b. Decodificação
2. Definindo a função de fitness
   a. Factibilidade vs Infactibilidade
   b. Mono objetivo vs Multi-objetivo

## Aspectos principais

3. Operadores
    a. Inicialização
    b. Seleção para Reprodução
    c. Crossover
    d. Mutação
    e. Seleção para Sobrevivência.
    f. Migração, genocídio, etc.
4. Critério de parada.
5. Avaliando o AE implementado
    a. Avaliando o  fitness médio
    b. Determinando o melhor indivíduo
    c. Ajustando os parâmetros (Tunning) do AE

# Representação e Operadores Clássicos

- Representação
  - Definir qual será a representação do indivíduo é uma das primeiras decisões que se deve tomar no desenvolvimento de um AG.
  - Codificação e Decodificação
    - Codificação direta
    - Codificação indireta ⇛ Decodificação

# Representação e Operadores Clássicos

- Representação Binária

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |

Representação Binária (8bits)

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |

Representação Binária (16 bits)

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |

Representação Binária (32 bits)

# Representação e Operadores Clássicos

- Representação Binária

cromossomo com 2 variáveis

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | $x_1$ |
| 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | $x_2$ |

cromossomo com 3 variáveis

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | $x_1$ |
| 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | $x_2$ |
| 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | $x_3$ |

# Representação e Operadores Clássicos

- Representação Binária
  - Duas importantes questões na representação de números reais em binários são:
    - Intervalo de domínio de cada uma das variáveis
    - Precisão desejada

# Representação e Operadores Clássicos

- Representação Binária

Exemplo - Decodificação

$$f(x) = x\,\text{sen}(10\pi x) + 1 \qquad -1 \le x \le 2\,.$$

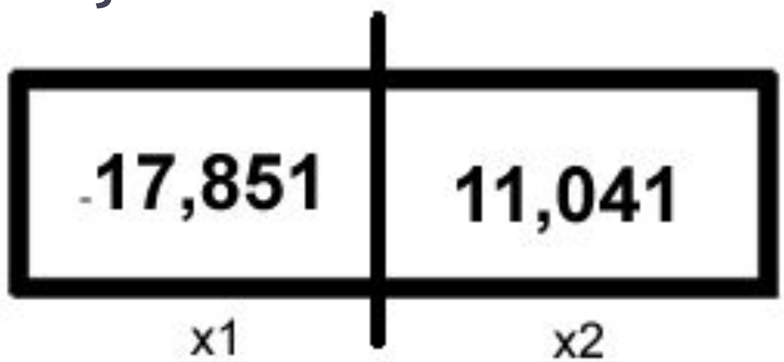
$$s_1 = 1000101110110101000111$$

$$b_{10} = (1000101110110101000111)_2 = 2288967$$

$$x = \min + (\max - \min)\frac{b_{10}}{2^l - 1} \qquad x_1 = -1 + (2+1)\frac{2.288.967}{\left(2^{22} - 1\right)} = 0{,}637197$$

# Representação e Operadores Clássicos

- Representação Real




Cromossomo Real

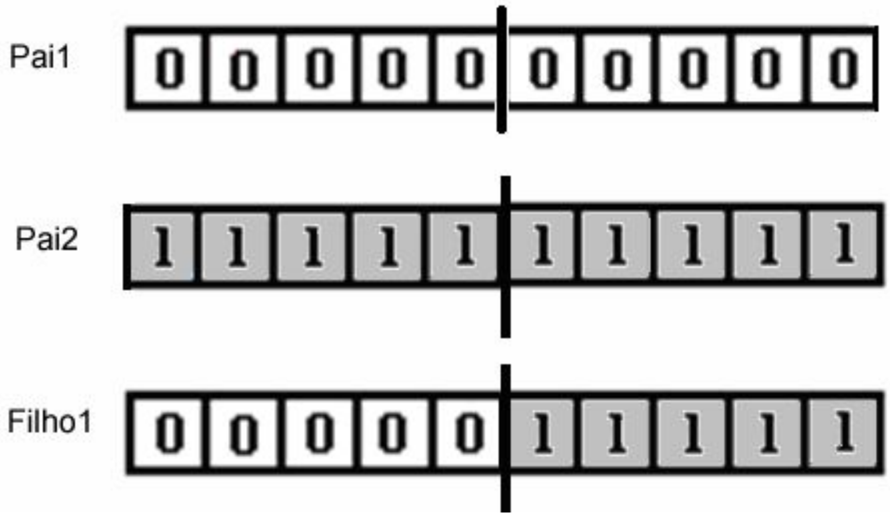
# Representação e Operadores Clássicos

## Representação Binária x Real

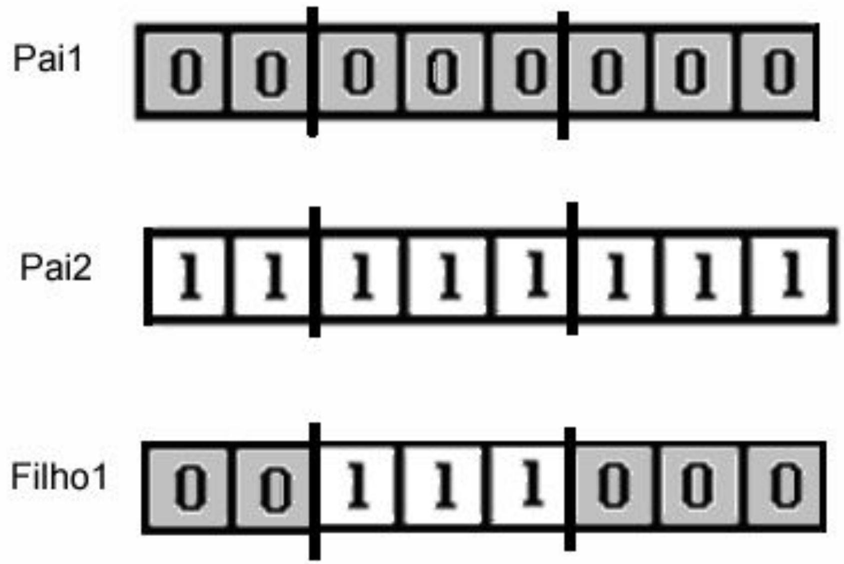
- Binária
  - ↑ Tradicional e fácil de utilizar
  - ↓ Cromossomos longos para representar parâmetros contínuos com boa precisão.
  - ↓ Longas cadeias podem levar a uma convergência lenta do método.
  - ↓ Não uniformidade dos operadores. Por exemplo, mutação nos primeiro bits tem mais impacto que nos últimos.
- Real
  - ↑ Cadeias menores
  - ↑ Compreendida mais naturalmente.
  - ↑ Facilidade para criar novos operadores.

# Operadores Clássicos

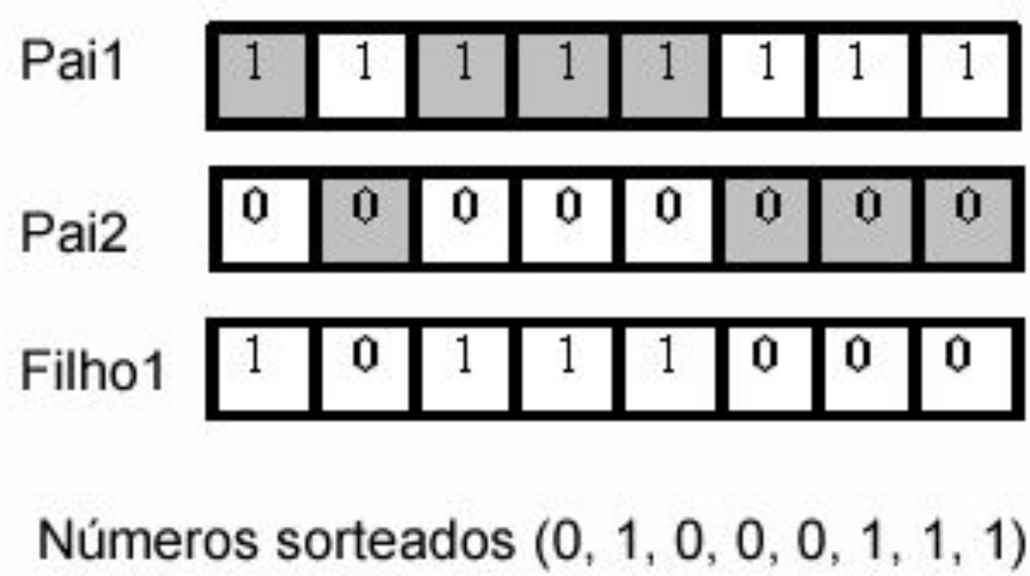
# Crossover de um ponto
# Representação Binária



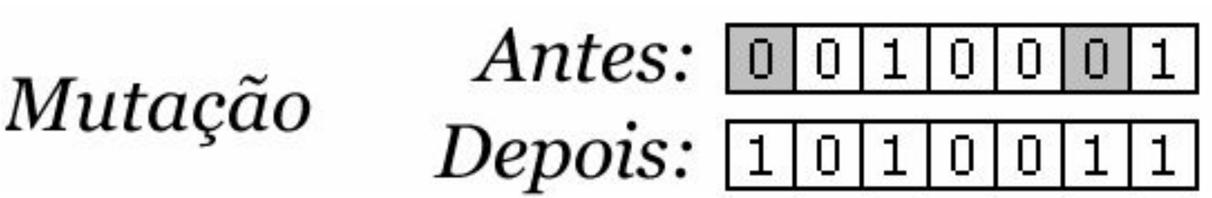
# Crossover de dois pontos Representação Binária



# Crossover uniforme
# Representação Binária



# Mutação
Representação Binária



# Crossovers
# Representação Real

Pais

$\mathbf{p}_1 = (p_{11}, p_{12}, \ldots, p_{1l})$

$\mathbf{p}_2 = (p_{21}, p_{22}, \ldots, p_{2l})$

Filho

$\mathbf{c} = (c_1, c_2, \ldots, c_l).$

**Crossover média** (Davis, 1991) $\mathbf{c} = (\mathbf{p}_1 + \mathbf{p}_2)/2$

**Crossover média geométrica**, $c_i = \sqrt{p_{1i} p_{2i}}$

# Crossovers - Representação Real

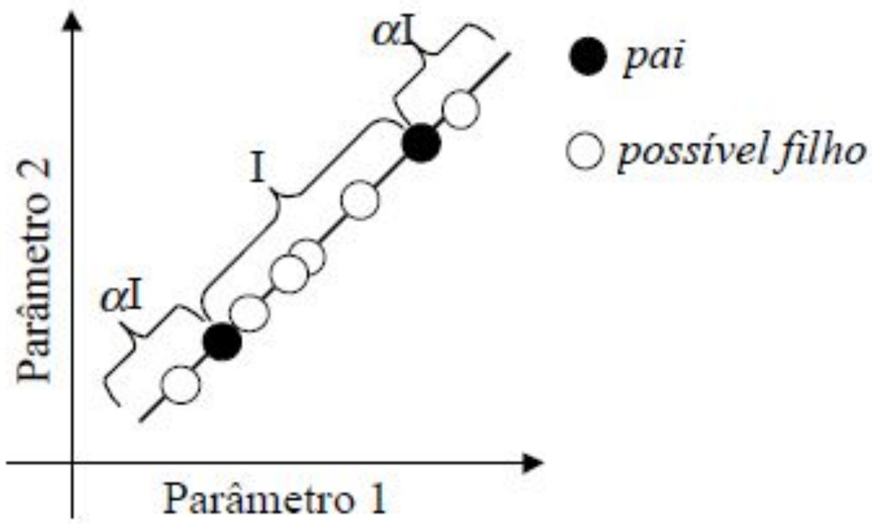
**Crossover *BLX-* $\alpha$**

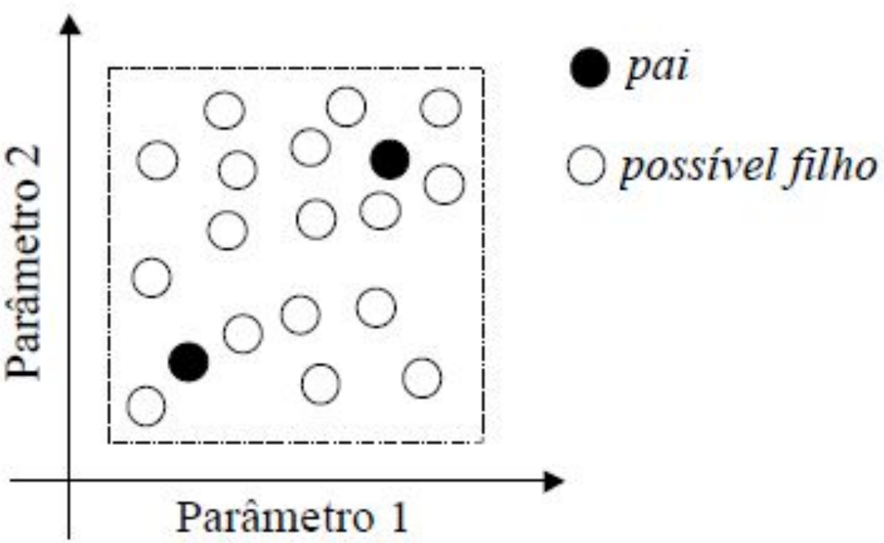
(Eshelman e Shaffer, 1993)

$$c = p_1 + \beta(p_2 - p_1) \qquad \beta \in U(-\alpha, 1+\alpha).$$

$p_1 = (30{,}173;\ 85{,}342)$ $\quad c_1 = 30{,}173 + 1{,}262(75{,}989 - 30{,}173) = 87{,}993$

$p_2 = (75{,}989;\ 10{,}162)$ $\quad c_2 = 85{,}342 + 1{,}262(10{,}162 - 85{,}342) = -9{,}535$



# Crossovers - Representação Real

**Crossover linear** (Wright, 1991):

$$\mathbf{c}_1 = 0{,}5\mathbf{p}_1 + 0{,}5\mathbf{p}_2$$

$$\mathbf{c}_2 = 1{,}5\mathbf{p}_1 - 0{,}5\mathbf{p}_2$$

$$\mathbf{c}_3 = -0{,}5\mathbf{p}_1 + 1{,}5\mathbf{p}_2$$

**Crossover aritmético** (Michalewicz, 1994)

$$\mathbf{c}_1 = \beta\mathbf{p}_1 + (1-\beta)\mathbf{p}_2$$

$$\mathbf{c}_2 = (1-\beta)\mathbf{p}_1 + \beta\mathbf{p}_2$$

$$\beta \in U(0,1)$$

**Crossover heurístico** (Michalewicz, 1994) $\mathbf{c} = \mathbf{p}_1 + r(\mathbf{p}_1 - \mathbf{p}_2)$, onde $f(\mathbf{p}_1) > f(\mathbf{p}_2)$

$$r \in U(0,1)$$

# Mutação
# Representação Real

**Mutação uniforme:**

$$c_i = \begin{cases} U(a_i, b_i), & \text{se } i = j \\ p_i & \text{caso contrário} \end{cases}$$

**Mutação gaussiana:**

$$c_i = \begin{cases} N(p_i, \sigma), & \text{se } i = j \\ p_i & \text{caso contrário} \end{cases}$$

**Mutação *creep*:**

# Mutação
# Representação Real

**Mutação limite** (Michalewicz, 1994) $c_i = \begin{cases} a_i & \text{se } r < 0{,}5 \text{ e } i = j \\ b_i & \text{se } r \geq 0{,}5 \text{ e } i = j \\ p_i & \text{caso contrário} \end{cases}$ $\quad r \in U(0,1)$

**Mutação não-uniforme** (Michalewicz, 1994):

$$c_i = \begin{cases} p_i + (b_i - p_i) f(G) & \text{se } r_1 < 0{,}5 \text{ e } i = j \\ p_i - (p_i - a_i) f(G) & \text{se } r_1 \geq 0{,}5 \text{ e } i = j \\ p_i & \text{caso contrário} \end{cases}$$

$$f(G) = \left( r_2 \left( 1 - \frac{G}{G_{\max}} \right) \right)^b$$

$r_1$ e $r_2 \in U(0,1)$

$b = 6$

# Fitness

- Medida de performance também conhecida como função objetivo (*objective function*), função de qualidade (*quality function*) ou função de avaliação (*evaluation function*).
- Utilizada como base para seleção (reprodução ou sobrevivência).

# Fitness

- Geralmente, busca-se maximizar o valor do *fitness*, porém pode ser implementada de forma trivial como um valor a ser minimizado:
  - 1/fitness
  - -fitness
  - N-fitness

# Fitness

- A determinação de uma função de *fitness* é uma tarefa difícila e muito importante.
- As restrições relacionadas ao problema podem ser incorpordadas na função de fitness via penalização.
  - Penaliza-se as restrições violadas.

# Fitness

- Porém, as restrições podem ser tratadas em AEs de outras formas:
  - Explicitamente com operadores de seleção que descartam codificações (indivíduos) infactíveis.
  - Explicitamente com operadores de reparos que atuam sobre codificações (indivíduos) infactíveis.
  - Implicitamente com codificações que permitem apenas a representação de soluções factíveis.

# Outras Representações e seus Operadores

# Outras Representações e seus Operadores

*International Journal of Production Research*
Vol. 47, No. 11, 1 June 2009, 3097–3119

Taylor & Francis
Taylor & Francis Group

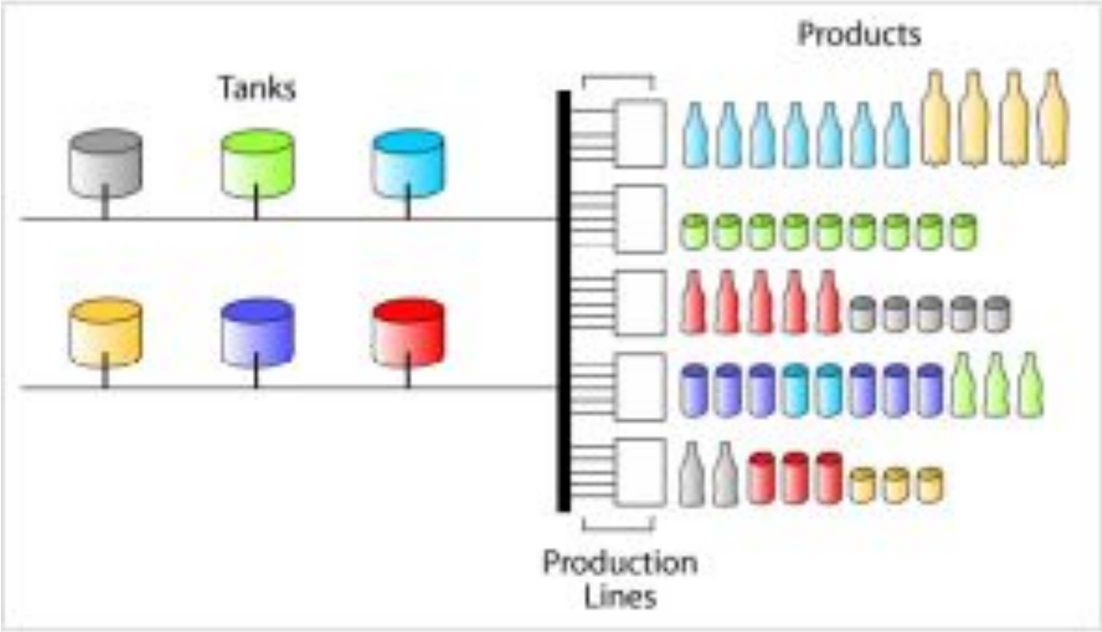
**Multi-population genetic algorithm to solve the synchronized and integrated two-level lot sizing and scheduling problem**

C.F.M. Toledo[a], P.M. França[b], R. Morabito[c] and A. Kimms[d]*

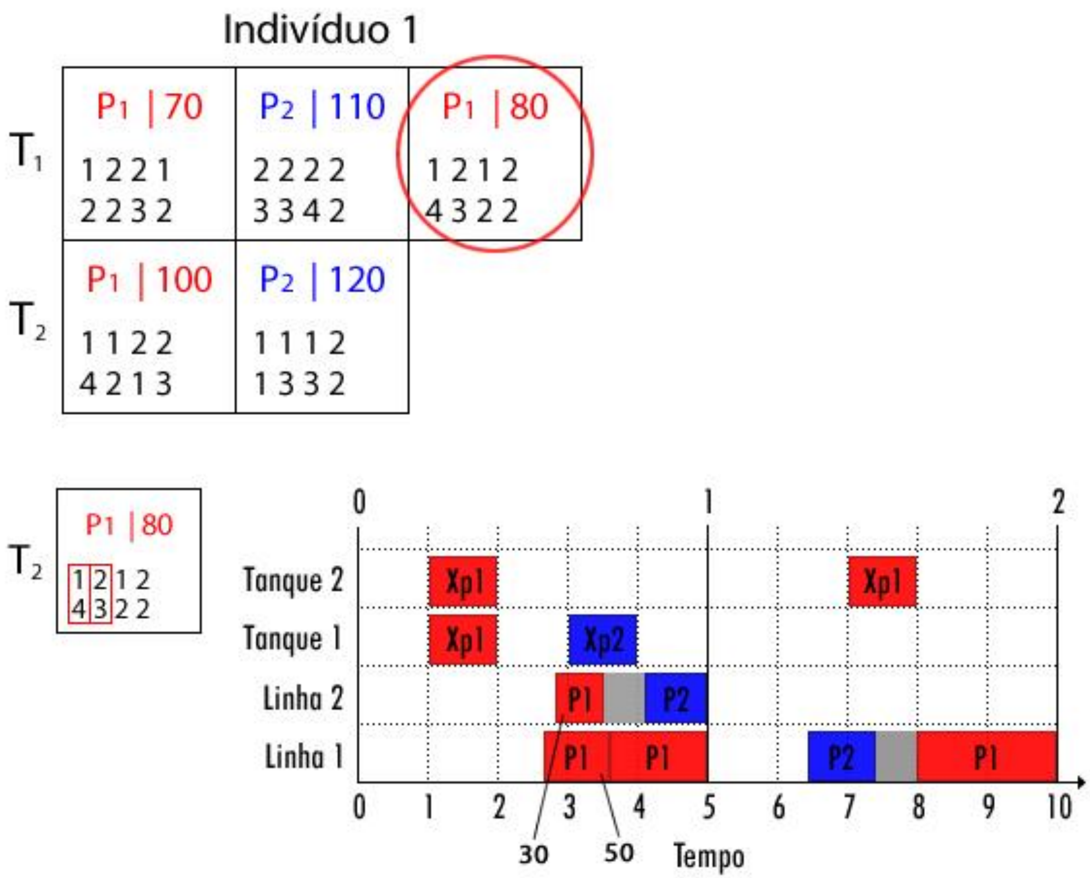
[a]*Departamento de Ciência da Computação, Universidade Federal de Lavras, C.P. 3037, 37200-000, Lavras, MG, Brazil;* [b]*Departamento de Matemática, Estatística e Computação, Universidade Estadual Paulista, Faculdade de Ciências e Tecnologia, Rua Roberto Simonsen, 305 19060-900, Presidente Prudente, SP, Brazil;* [c]*Departamento de Engenharia de Produção, Universidade Federal de São Carlos, C.P. 676, 13565-905, São Carlos, SP, Brazil;* [d]*Department of Technology and Operations Management, University of Duisburg-Essen, 47048 Duisburg, Germany*

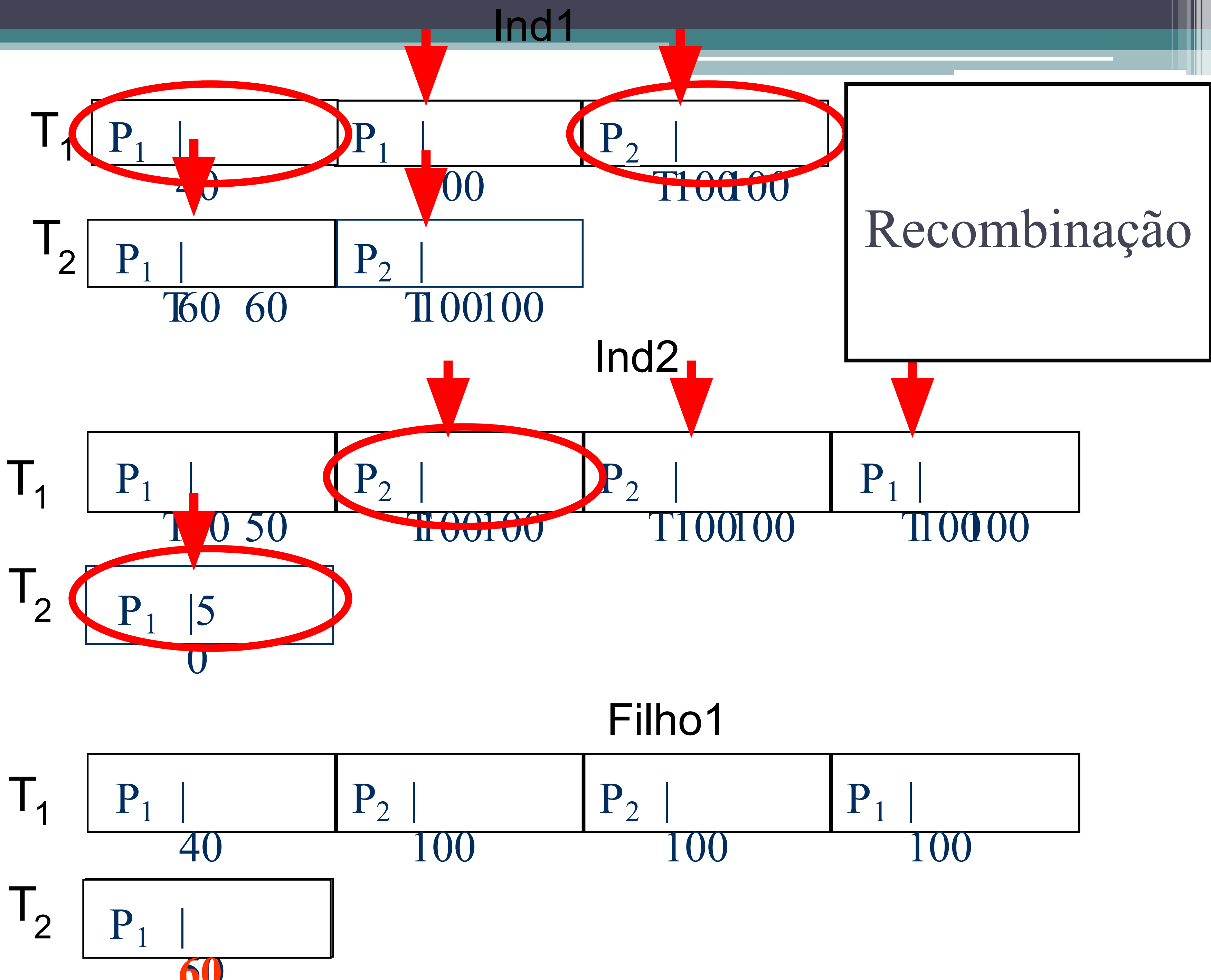
# Outras Representações e seus Operadores

## Problema Industrial



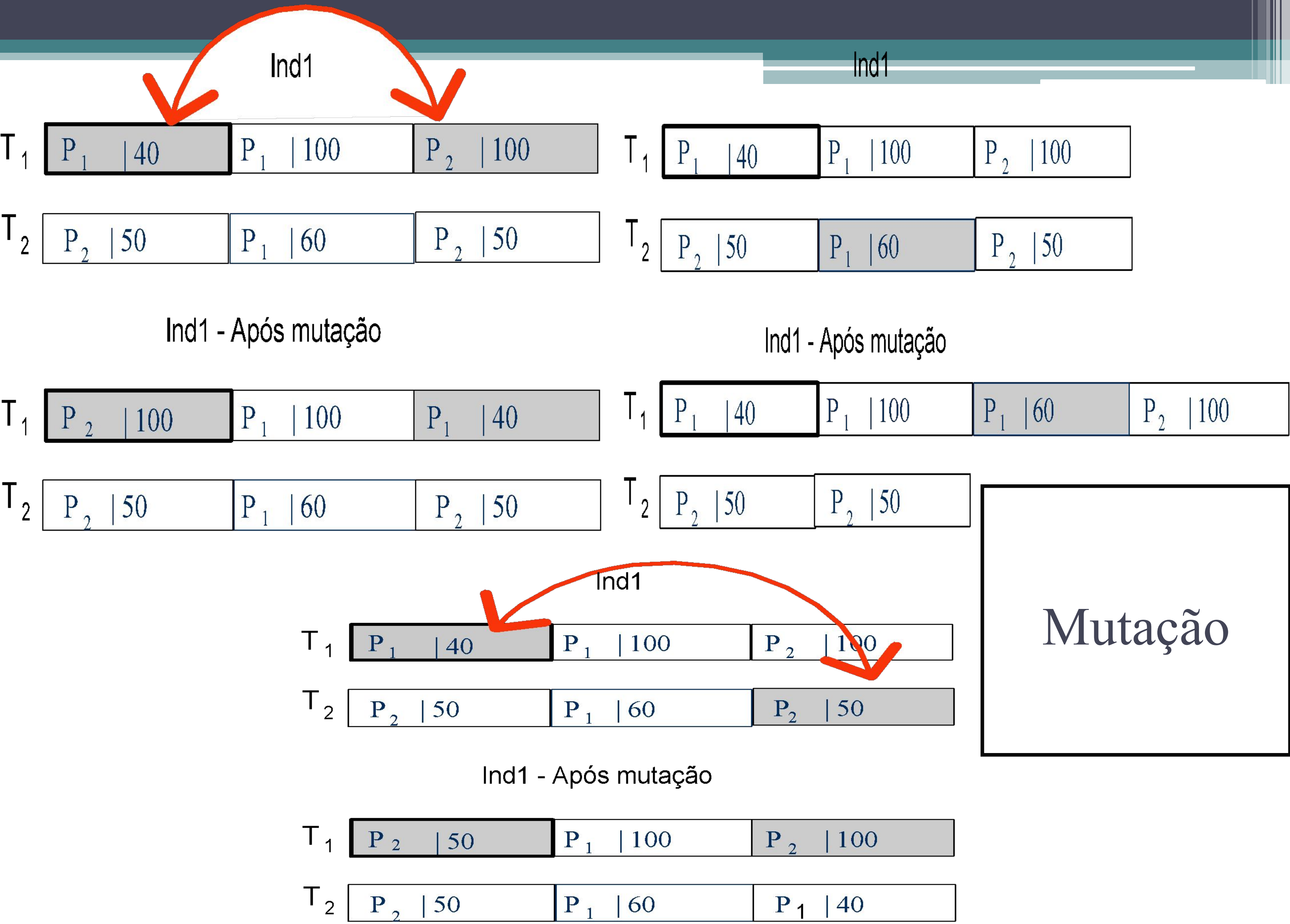
# Codificação e Decodificação





Ind1
T1
P1 |
P1 |
P2 |
T2
P1 |
P2 |
Recombinação
Ind2
T1
P1 |
P2 |
P2 |
P1 |
T2
P1 |5
0
Filho1
T1
P1 |
40
P2 |
100
P2 |
100
P1 |
100
T2
P1 |
60

# Mutação


Ind1
T 1
P 1 | 40
P 1 | 100
P 2 | 100
T 2
P 2 | 50
P 1 | 60
P 2 | 50
Ind1 - Após mutação
T 1
P 2 | 100
P 1 | 100
P 1 | 40
T 2
P 2 | 50
P 1 | 60
P 2 | 50
Ind1
T 1
P 1 | 40
P 1 | 100
P 2 | 100
T 2
P 2 | 50
P 1 | 60
P 2 | 50
Ind1 - Após mutação
T 1
P 1 | 40
P 1 | 100
P 1 | 60
P 2 | 100
T 2
P 2 | 50
P 2 | 50
Ind1
T 1
P 1 | 40
P 1 | 100
P 2 | 100
T 2
P 2 | 50
P 1 | 60
P 2 | 50
Ind1 - Após mutação
T 1
P 2 | 50
P 1 | 100
P 2 | 100
T 2
P 2 | 50
P 1 | 60
P 1 | 40

# Fitness - Função Objetivo

$$\sum_{i=1}^{J}\sum_{j=1}^{J}\sum_{l=1}^{L}\sum_{s=1}^{S} s_{ijl}\, z_{ijl} + \sum_{j=1}^{J}\sum_{t=1}^{T} h_j\, I_{jt} + \sum_{j=1}^{J}\sum_{l=1}^{L}\sum_{s=1}^{T.S} v_{jl}\, q_{jls} +$$

$$\sum_{i=1}^{\overline{J}}\sum_{j=1}^{\overline{J}}\sum_{k=1}^{\overline{L}}\sum_{s=1}^{\overline{S}^{(1)}} \overline{s}_{ijl}\, \overline{z}_{ijl} + \sum_{j=1}^{\overline{J}}\sum_{k=1}^{\overline{L}}\sum_{t=1}^{\overline{T}} \overline{h}_j\, \overline{I}_{jk,t.T^m} + \sum_{j=1}^{\overline{J}}\sum_{k=1}^{\overline{L}}\sum_{s=1}^{T.\overline{S}} \overline{v}_{jl}\, \overline{q}_{jks} +$$

$$M \sum_{j=1}^{J} q_j^0 \quad \mathbf{(1)}$$

- Variáveis
- $q_{jls} \geq 0$: quantidade do produto j produzida em l no lote s.
- $q_{jks} \geq 0$: quantidade do xarope j armazenada no tanque k e lote s.
- $q^0_j$: demanda não atendida do produto j.

# Outras Representações e seus Operadores

Computers & Operations Research 40 (2013) 910–919

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

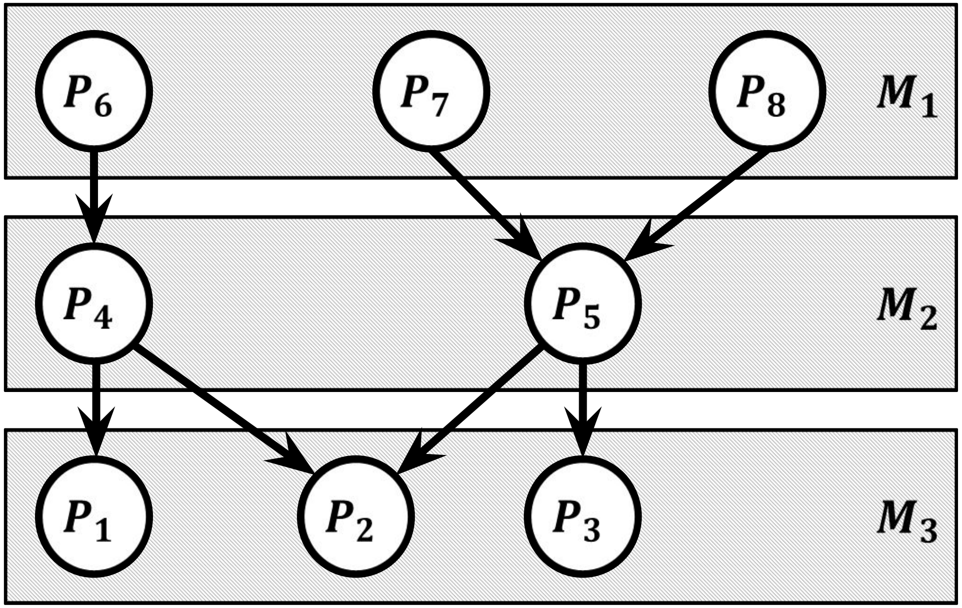
journal homepage: www.elsevier.com/locate/caor


## A hybrid multi-population genetic algorithm applied to solve the multi-level capacitated lot sizing problem with backlogging

Claudio Fabiano Motta Toledo [a,*], Renato Resende Ribeiro de Oliveira [b], Paulo Morelato França [c]

[a] University of São Paulo, Institute of Mathematics and Computer Science, Brazil
[b] Federal University of Lavras, Department of Computer Science, Brazil
[c] UNESP - Department of Mathematics and Computing, Brazil

# The Multi-Level Capacitated Lot Sizing Problem (MLCLSP)



# Formulação Matemática

Fitness - Função Objetivo ►

$$\min \sum_{j=1}^{J} \sum_{t=1}^{T} (bc_j\, b_{jt} + h_j\, i_{jt}) \qquad (1)$$

$$i_{j(t-1)} + b_{jt} + x_{jt} = i_{jt} + b_{j(t-1)} + D_{jt} \quad \forall j,t \mid j \in \Delta \qquad (2)$$

$$i_{j(t-1)} + x_{jt} = i_{jt} + \sum_{k \in \delta_j} r_{jk}\, x_{kt} \quad \forall j,t \mid j \notin \Delta \qquad (3)$$

$$x_{jt} \leq y_{jt}\, B_{jt} \quad \forall j,t \qquad (4)$$

$$y_{jt} \leq w_{ft} \quad \forall j,t,f \mid p_{jf} = 1 \qquad (5)$$

$$\sum_{j=1}^{J} a_{mj}\, x_{jt} + \sum_{f=1}^{F} st_{mf}\, w_{ft} \leq C_{mt} \quad \forall m,t \qquad (6)$$

$$x_{jt}, i_{jt}, b_{jt} \geq 0 \quad y_{jt}, w_{ft} \in \{0,1\} \qquad (7)$$

# Formulação Matemática

Fitness - Função Objetivo ►

$$\min \sum_{j=1}^{J} \sum_{t=1}^{T} (bc_j \, b_{jt} + h_j \, i_{jt}) \quad (1)$$

External demand ►

$$i_{j(t-1)} + b_{jt} + x_{jt} = i_{jt} + b_{j(t-1)} + D_{jt} \quad \forall j, t \mid j \in \Delta \quad (2)$$

$$i_{j(t-1)} + x_{jt} = i_{jt} + \sum_{k \in \delta_j} r_{jk} \, x_{kt} \quad \forall j, t \mid j \notin \Delta \quad (3)$$

$$x_{jt} \leq y_{jt} \, B_{jt} \quad \forall j, t \quad (4)$$

$$y_{jt} \leq w_{ft} \quad \forall j, t, f \mid p_{jf} = 1 \quad (5)$$

$$\sum_{j=1}^{J} a_{mj} \, x_{jt} + \sum_{f=1}^{F} st_{mf} \, w_{ft} \leq C_{mt} \quad \forall m, t \quad (6)$$

$$x_{jt}, i_{jt}, b_{jt} \geq 0 \quad y_{jt}, w_{ft} \in \{0, 1\} \quad (7)$$

# Formulação Matemática

Fitness - Função Objetivo ► $$\min \sum_{j=1}^{J} \sum_{t=1}^{T} (bc_j\, b_{jt} + h_j\, i_{jt}) \quad (1)$$

External demand ► $$i_{j(t-1)} + b_{jt} + x_{jt} = i_{jt} + b_{j(t-1)} + D_{jt} \quad \forall j, t \mid j \in \Delta \quad (2)$$

Internal demand ► $$i_{j(t-1)} + x_{jt} = i_{jt} + \sum_{k \in \delta_j} r_{jk}\, x_{kt} \quad \forall j, t \mid j \notin \Delta \quad (3)$$

$$x_{jt} \leq y_{jt}\, B_{jt} \quad \forall j, t \quad (4)$$

$$y_{jt} \leq w_{ft} \quad \forall j, t, f \mid p_{jf} = 1 \quad (5)$$

$$\sum_{j=1}^{J} a_{mj}\, x_{jt} + \sum_{f=1}^{F} st_{mf}\, w_{ft} \leq C_{mt} \quad \forall m, t \quad (6)$$

$$x_{jt}, i_{jt}, b_{jt} \geq 0 \quad y_{jt}, w_{ft} \in \{0, 1\} \quad (7)$$

# Formulação Matemática

Fitness - Função Objetivo ►
$$\min \sum_{j=1}^{J} \sum_{t=1}^{T} (bc_j \, b_{jt} + h_j \, i_{jt}) \quad (1)$$

External demand ►
$$i_{j(t-1)} + b_{jt} + x_{jt} = i_{jt} + b_{j(t-1)} + D_{jt} \quad \forall j, t \mid j \in \Delta \quad (2)$$

Internal demand ►
$$i_{j(t-1)} + x_{jt} = i_{jt} + \sum_{k \in \delta_j} r_{jk} \, x_{kt} \quad \forall j, t \mid j \notin \Delta \quad (3)$$

Production allowance ►
$$x_{jt} \leq y_{jt} \, B_{jt} \quad \forall j, t \quad (4)$$
$$y_{jt} \leq w_{ft} \quad \forall j, t, f \mid p_{jf} = 1 \quad (5)$$

$$\sum_{j=1}^{J} a_{mj} \, x_{jt} + \sum_{f=1}^{F} st_{mf} \, w_{ft} \leq C_{mt} \quad \forall m, t \quad (6)$$

$$x_{jt}, i_{jt}, b_{jt} \geq 0 \quad y_{jt}, w_{ft} \in \{0, 1\} \quad (7)$$

# Formulação Matemática

Fitness - Função Objetivo ► $$\min \sum_{j=1}^{J} \sum_{t=1}^{T} (bc_j \, b_{jt} + h_j \, i_{jt}) \quad (1)$$

External demand ► $$i_{j(t-1)} + b_{jt} + x_{jt} = i_{jt} + b_{j(t-1)} + D_{jt} \quad \forall j,t \mid j \in \Delta \quad (2)$$

Internal demand ► $$i_{j(t-1)} + x_{jt} = i_{jt} + \sum_{k \in \delta_j} r_{jk} \, x_{kt} \quad \forall j,t \mid j \notin \Delta \quad (3)$$

Production allowance ►
$$x_{jt} \leq y_{jt} \, B_{jt} \quad \forall j,t \quad (4)$$
$$y_{jt} \leq w_{ft} \quad \forall j,t,f \mid p_{jf} = 1 \quad (5)$$

Capacity limit ► $$\sum_{j=1}^{J} a_{mj} \, x_{jt} + \sum_{f=1}^{F} st_{mf} \, w_{ft} \leq C_{mt} \quad \forall m,t \quad (6)$$

$$x_{jt}, i_{jt}, b_{jt} \geq 0 \quad y_{jt}, w_{ft} \in \{0,1\} \quad (7)$$

# Formulação Matemática

Fitness - Função Objetivo ▶ $$\min \sum_{j=1}^{J} \sum_{t=1}^{T} (bc_j\, b_{jt} + h_j\, i_{jt}) \quad (1)$$

External demand ▶ $$i_{j(t-1)} + b_{jt} + x_{jt} = i_{jt} + b_{j(t-1)} + D_{jt} \quad \forall j,t \mid j \in \Delta \quad (2)$$

Internal demand ▶ $$i_{j(t-1)} + x_{jt} = i_{jt} + \sum_{k \in \delta_j} r_{jk}\, x_{kt} \quad \forall j,t \mid j \notin \Delta \quad (3)$$

Production allowance ▶ $$x_{jt} \leq y_{jt}\, B_{jt} \quad \forall j,t \quad (4)$$

$$y_{jt} \leq w_{ft} \quad \forall j,t,f \mid p_{jf} = 1 \quad (5)$$

Capacity limit ▶ $$\sum_{j=1}^{J} a_{mj}\, x_{jt} + \sum_{f=1}^{F} st_{mf}\, w_{ft} \leq C_{mt} \quad \forall m,t \quad (6)$$

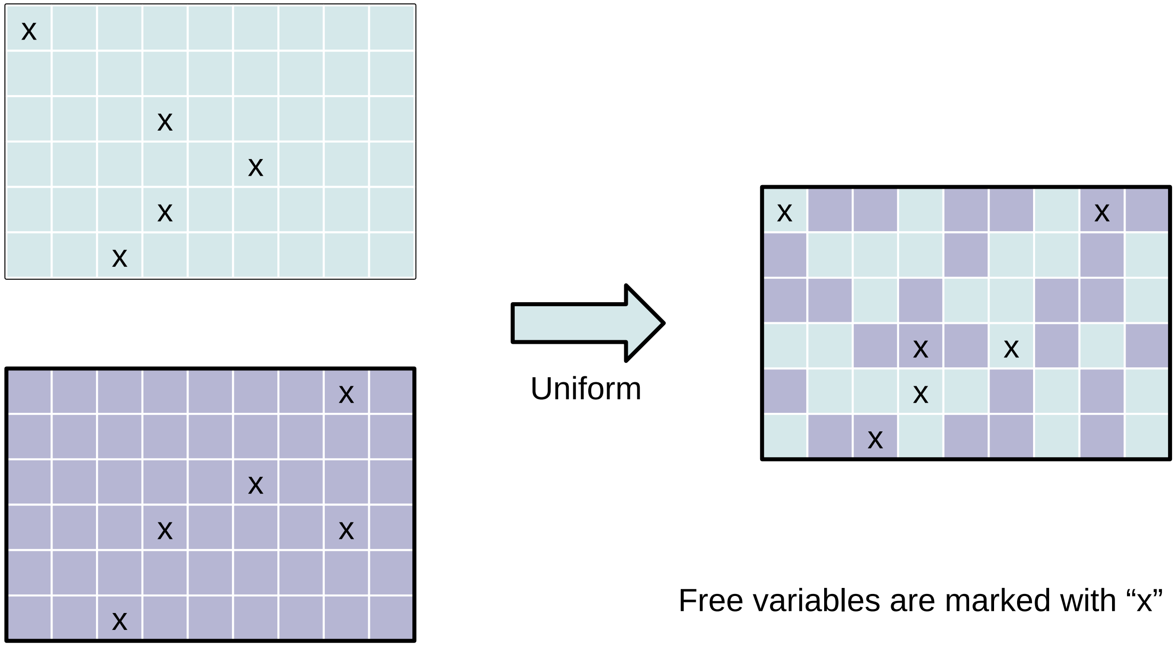
Variable domains ▶ $$x_{jt}, i_{jt}, b_{jt} \geq 0 \quad y_{jt}, w_{ft} \in \{0,1\} \quad (7)$$

# Representação

- Matriz Binária F x T

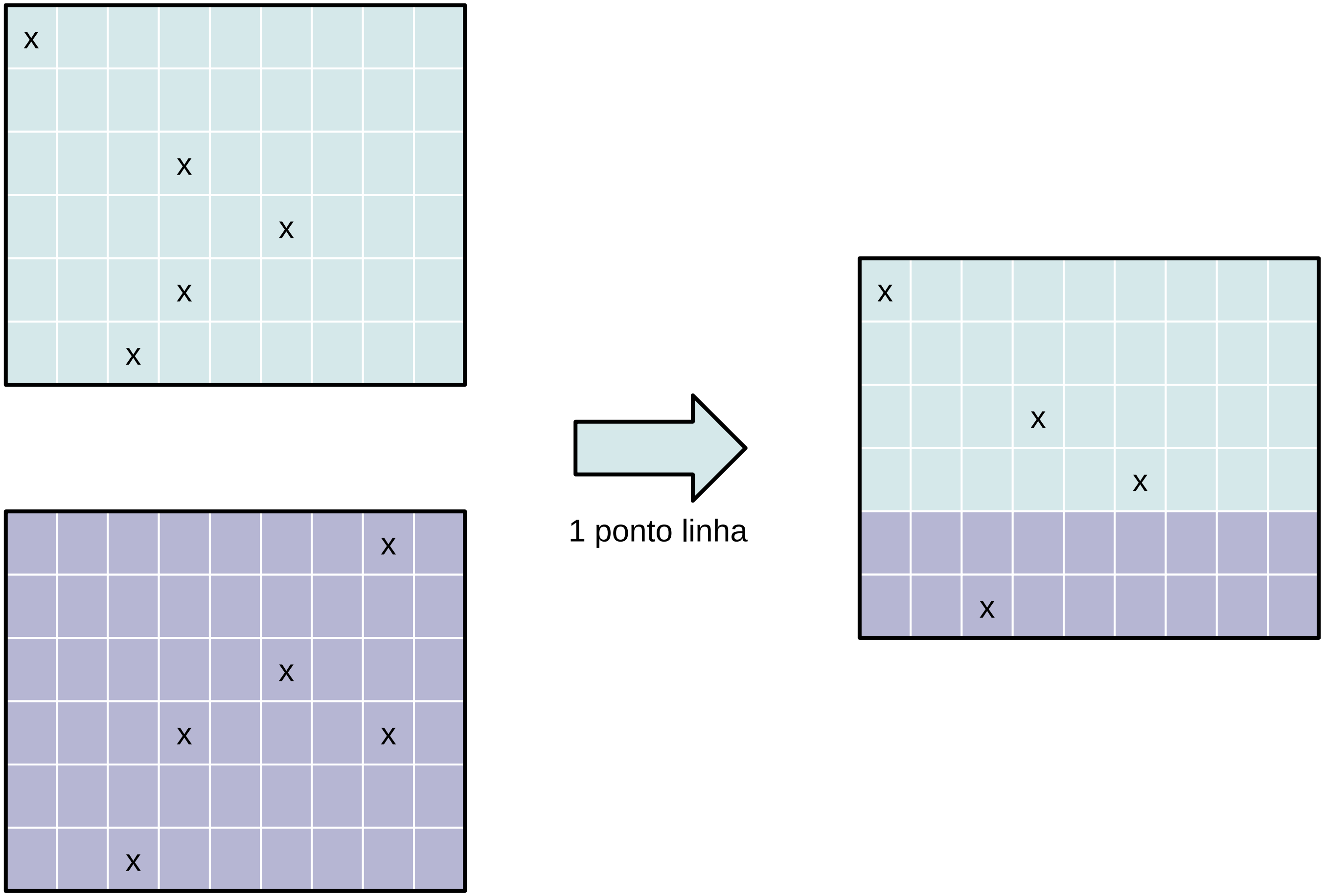
| | $T_1$ | $T_2$ | $T_3$ | $T_4$ | $T_5$ | $T_6$ | $T_7$ | $T_8$ | $T_9$ |
|---|---|---|---|---|---|---|---|---|---|
| $F_1$ | **[1]** | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 |
| $F_2$ | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| $F_3$ | 1 | 1 | 0 | **[0]** | 0 | 0 | 0 | 0 | 1 |
| $F_4$ | 1 | 0 | 0 | 1 | 0 | **[0]** | 1 | 0 | 0 |
| $F_5$ | 1 | 0 | 0 | **[1]** | 1 | 1 | 0 | 0 | 1 |
| $F_6$ | 1 | 0 | **[0]** | 1 | 0 | 0 | 0 | 0 | 1 |

# Crossover

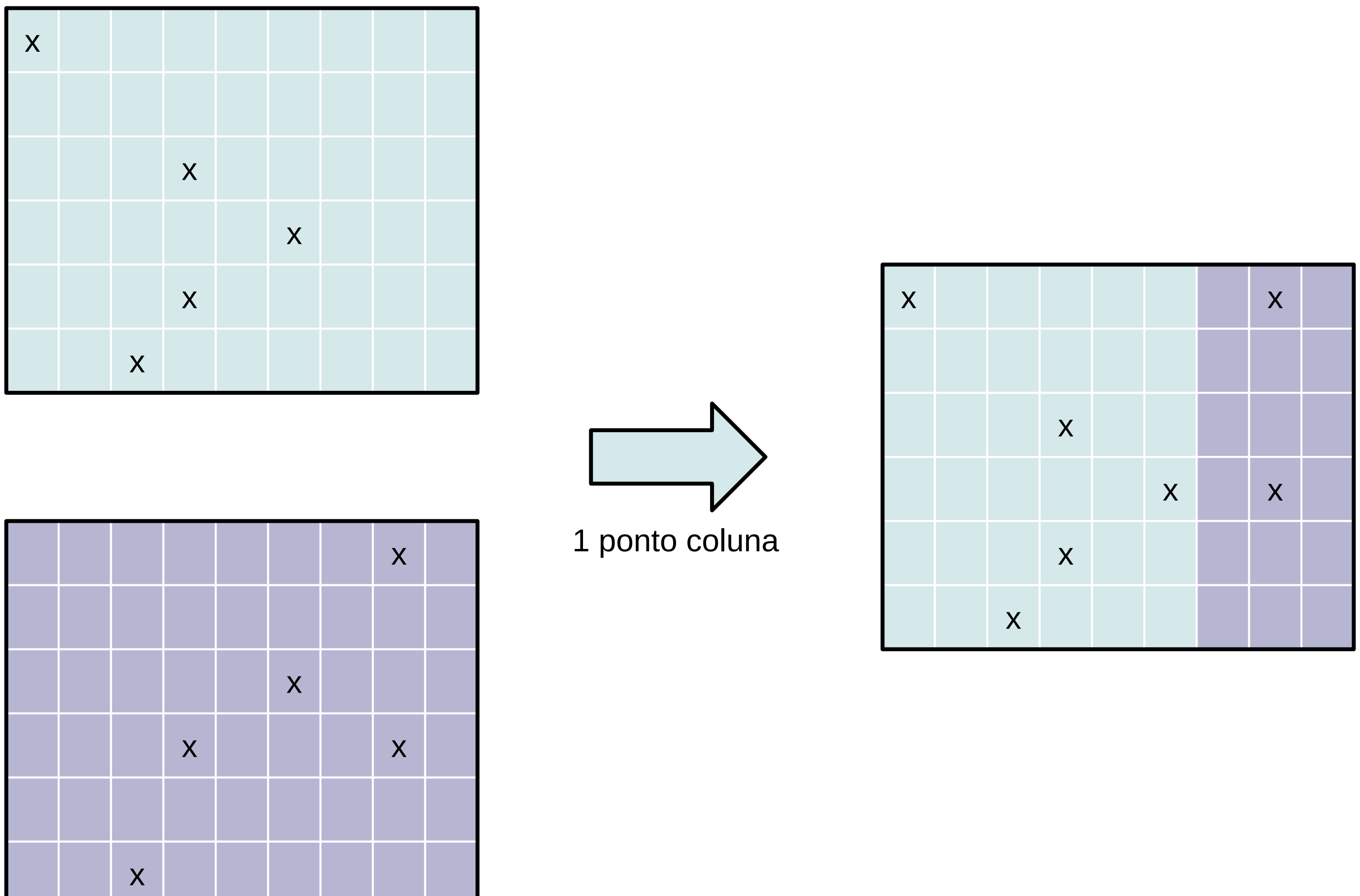



Free variables are marked with “x”

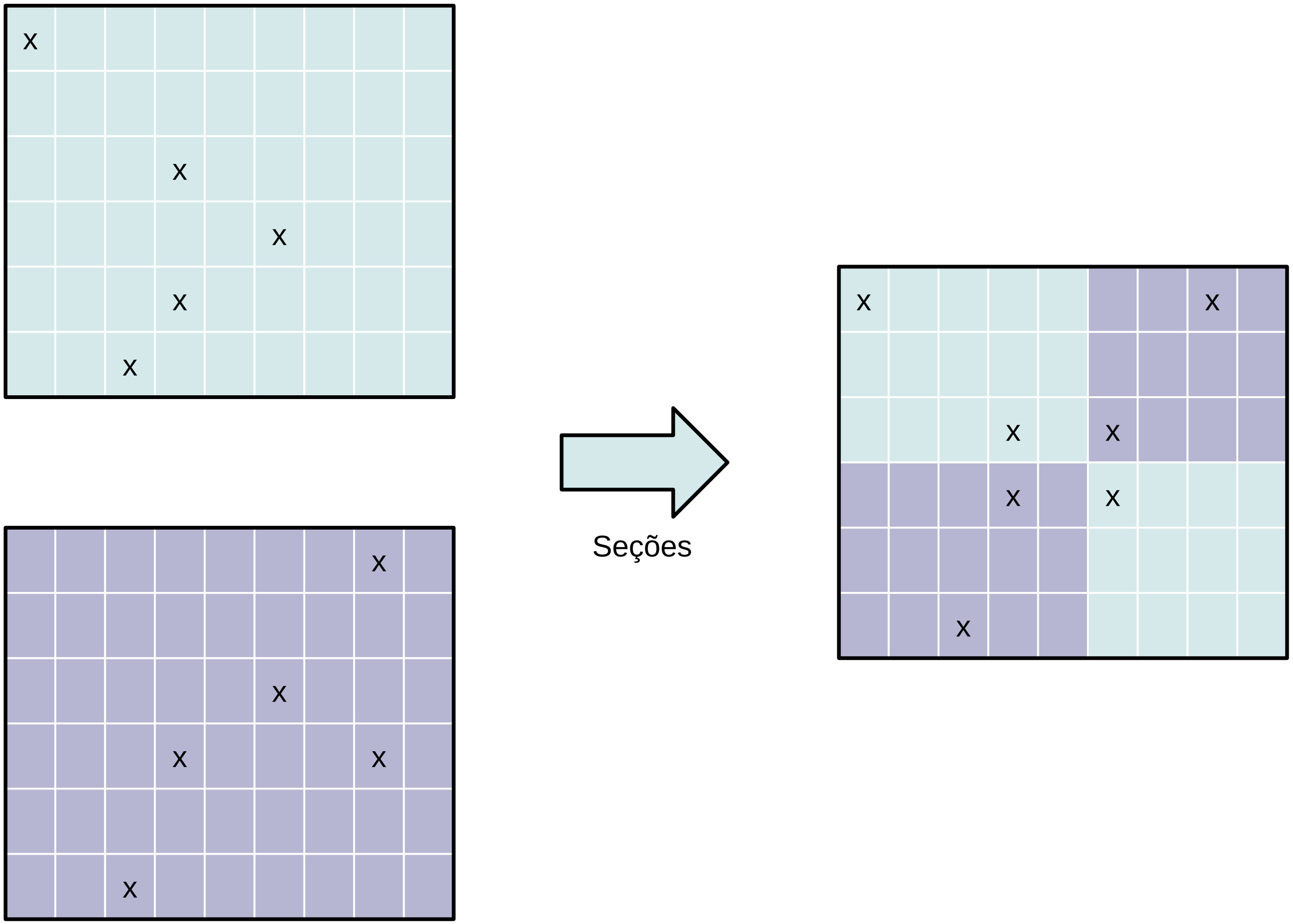
# Crossover



# Crossover



# Crossover



# Mutação

| | $T_1$ | $T_2$ | $T_3$ | $T_4$ | $T_5$ | $T_6$ | $T_7$ | $T_8$ | $T_9$ |
|---|---|---|---|---|---|---|---|---|---|
| $F_1$ | [1] | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 |
| $F_2$ | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| $F_3$ | 1 | 1 | 0 | [0] | 0 | 0 | 0 | 0 | 1 |
| $F_4$ | 1 | 0 | 0 | 1 | 0 | [0] | 1 | 0 | 0 |
| $F_5$ | 1 | 0 | 0 | [1] | 1 | 1 | 0 | 0 | 1 |
| $F_6$ | 1 | 0 | [0] | 1 | 0 | 0 | 0 | 0 | 1 |

Inversão

| | $T_1$ | $T_2$ | $T_3$ | $T_4$ | $T_5$ | $T_6$ | $T_7$ | $T_8$ | $T_9$ |
|---|---|---|---|---|---|---|---|---|---|
| $F_1$ | [1] | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 |
| $F_2$ | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| $F_3$ | 1 | 1 | 0 | [0] | 0 | 0 | 0 | 0 | 1 |
| $F_4$ | 1 | 0 | 0 | 1 | 0 | [0] | 1 | 0 | 0 |
| $F_5$ | 1 | 0 | 0 | [1] | 1 | 1 | 0 | 0 | 1 |
| $F_6$ | 1 | 0 | [0] | 1 | 0 | 0 | 0 | 0 | 1 |

# Mutação

| | $T_1$ | $T_2$ | $T_3$ | $T_4$ | $T_5$ | $T_6$ | $T_7$ | $T_8$ | $T_9$ |
|---|---|---|---|---|---|---|---|---|---|
| $F_1$ | [1] | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 |
| $F_2$ | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| $F_3$ | 1 | 1 | 0 | [0] | 0 | 0 | 0 | 0 | 1 |
| $F_4$ | 1 | 0 | 0 | 1 | 0 | [0] | 1 | 0 | 0 |
| $F_5$ | 1 | 0 | 0 | [1] | 1 | 1 | 0 | 0 | 1 |
| $F_6$ | 1 | 0 | [0] | 1 | 0 | 0 | 0 | 0 | 1 |

Troca na Coluna

| | $T_1$ | $T_2$ | $T_3$ | $T_4$ | $T_5$ | $T_6$ | $T_7$ | $T_8$ | $T_9$ |
|---|---|---|---|---|---|---|---|---|---|
| $F_1$ | [1] | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| $F_2$ | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| $F_3$ | 1 | 1 | 0 | [0] | 0 | 0 | 0 | 0 | 1 |
| $F_4$ | 1 | 0 | 0 | 1 | 0 | [1] | 1 | 0 | 0 |
| $F_5$ | 1 | 0 | 0 | [1] | 1 | 1 | 0 | 0 | 1 |
| $F_6$ | 1 | 0 | [0] | 1 | 0 | 0 | 0 | 0 | 1 |

# Mutação

| | $T_1$ | $T_2$ | $T_3$ | $T_4$ | $T_5$ | $T_6$ | $T_7$ | $T_8$ | $T_9$ |
|---|---|---|---|---|---|---|---|---|---|
| $F_1$ | **[1]** | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 |
| $F_2$ | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| $F_3$ | 1 | 1 | 0 | **[0]** | 0 | 0 | 0 | 0 | 1 |
| $F_4$ | 1 | 0 | 0 | 1 | 0 | **[0]** | 1 | 0 | 0 |
| $F_5$ | 1 | 0 | 0 | **[1]** | 1 | 1 | 0 | 0 | 1 |
| $F_6$ | 1 | 0 | **[0]** | 1 | 0 | 0 | 0 | 0 | 1 |

Troca na linha

| | $T_1$ | $T_2$ | $T_3$ | $T_4$ | $T_5$ | $T_6$ | $T_7$ | $T_8$ | $T_9$ |
|---|---|---|---|---|---|---|---|---|---|
| $F_1$ | **[1]** | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 |
| $F_2$ | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| $F_3$ | 1 | 1 | 0 | **[0]** | 0 | 0 | 0 | 0 | 1 |
| $F_4$ | 1 | 1 | 0 | 1 | 0 | **[0]** | 0 | 0 | 0 |
| $F_5$ | 1 | 0 | 0 | **[1]** | 1 | 1 | 0 | 0 | 1 |
| $F_6$ | 1 | 0 | **[0]** | 1 | 0 | 0 | 0 | 0 | 1 |

# Outras Representações e seus Operadores

ELSEVIER

Computers & Operations Research

Volume 48, August 2014, Pages 40–52

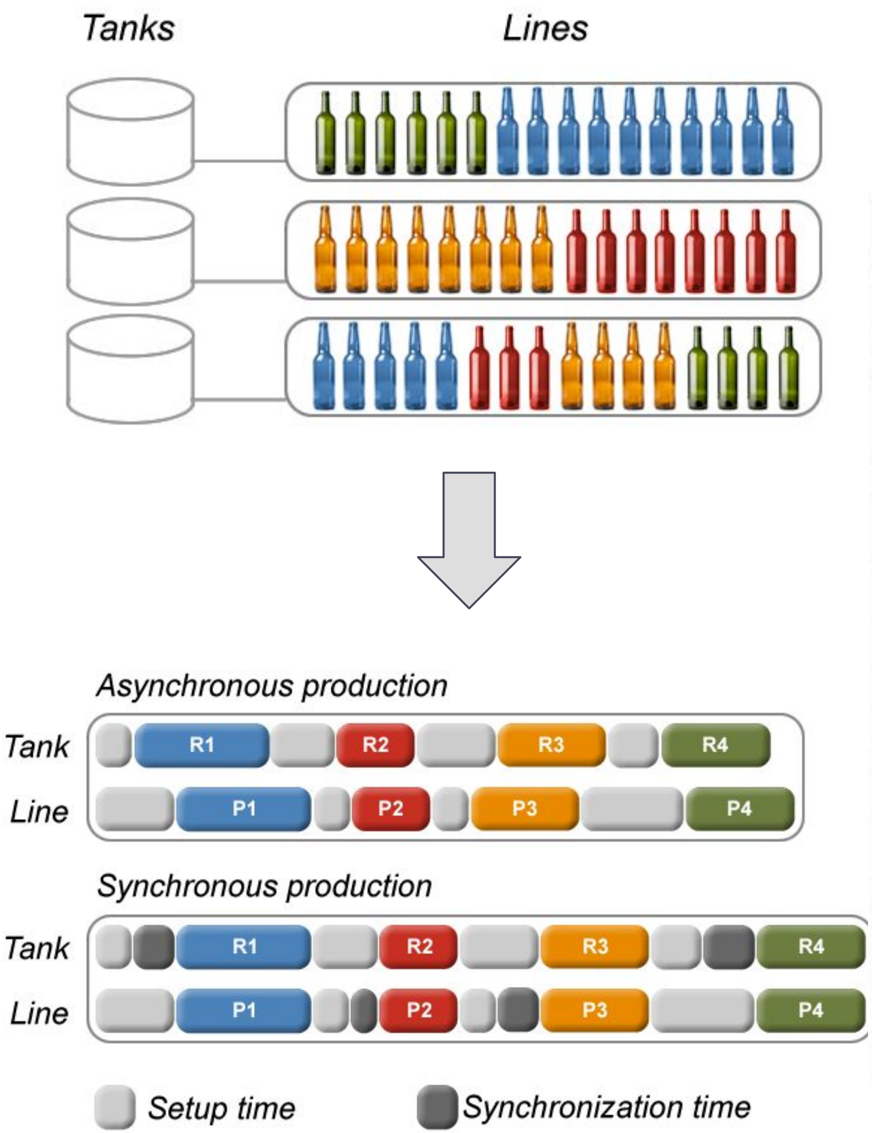

## A genetic algorithm/mathematical programming approach to solve a two-level soft drink production problem

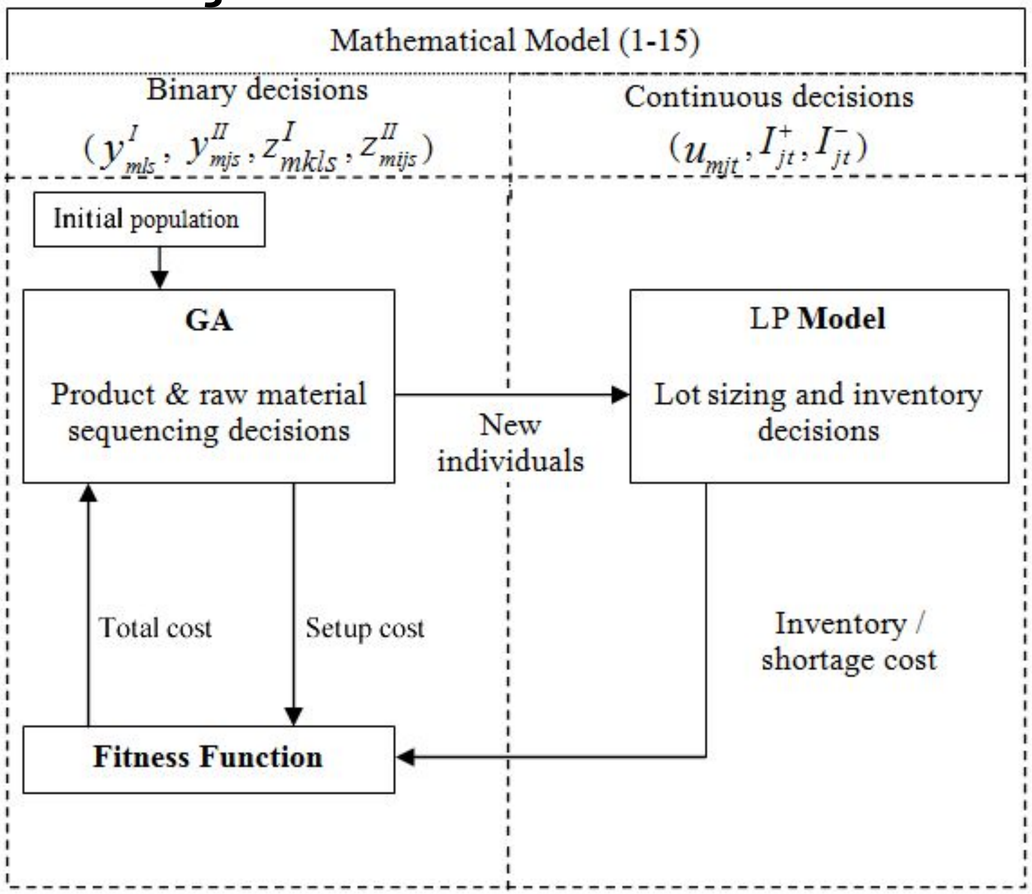
Claudio Fabiano Motta Toledo[a], Lucas de Oliveira[b], Rodrigo de Freitas Pereira[a], Paulo Morelato França[c], Reinaldo Morabito[d]

Show more

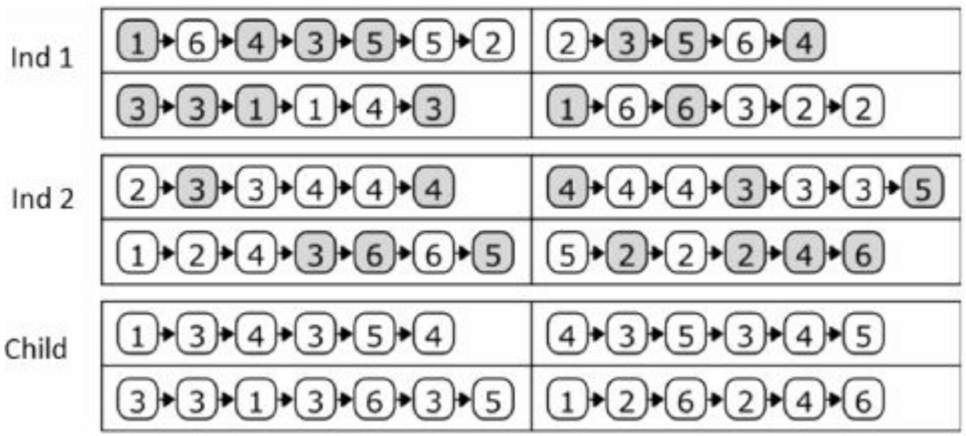
https://doi.org/10.1016/j.cor.2014.02.012

# Problema Industrial e Solução Híbrida



# Codificação, Decodificação e Fitness




**evaluateFitness**(*Individual*)
**begin**

*//Decoding procedure*
totalSetupCosts = calculateSetupCosts(*Individual*).
updatedParameters(Individual, $ns_{mjt}$ , $W_{mt}$ ).
*//LP Model is solved using Cplex*
totalInventoryPlusShortageCosts = solveLPModel( $ns_{mjt}$ , $W_{mt}$ )
fitnessValue = totalSetupCosts + totalInventoryPlusShortageCosts

return fitnessValue.
**end**

# Codificação, Decodificação e Fitness

Calculado solucionando modelo PL

Minimize

$$\sum_{j=1}^{J}\sum_{t=1}^{T}(h_j I_{jt}^{+} + g_j I_{jt}^{-}) + \sum_{s=1}^{N}\sum_{m=1}^{M}\sum_{k\in\beta_m}\sum_{l\in\beta_m} s_{kl}^{I} z_{mkls}^{I}$$

$$+ \sum_{s=1}^{N}\sum_{m=1}^{M}\sum_{i\in\alpha_m}\sum_{j\in\alpha_m} s_{ij}^{II} z_{mijs}^{II}$$

# Crossover e Mutação

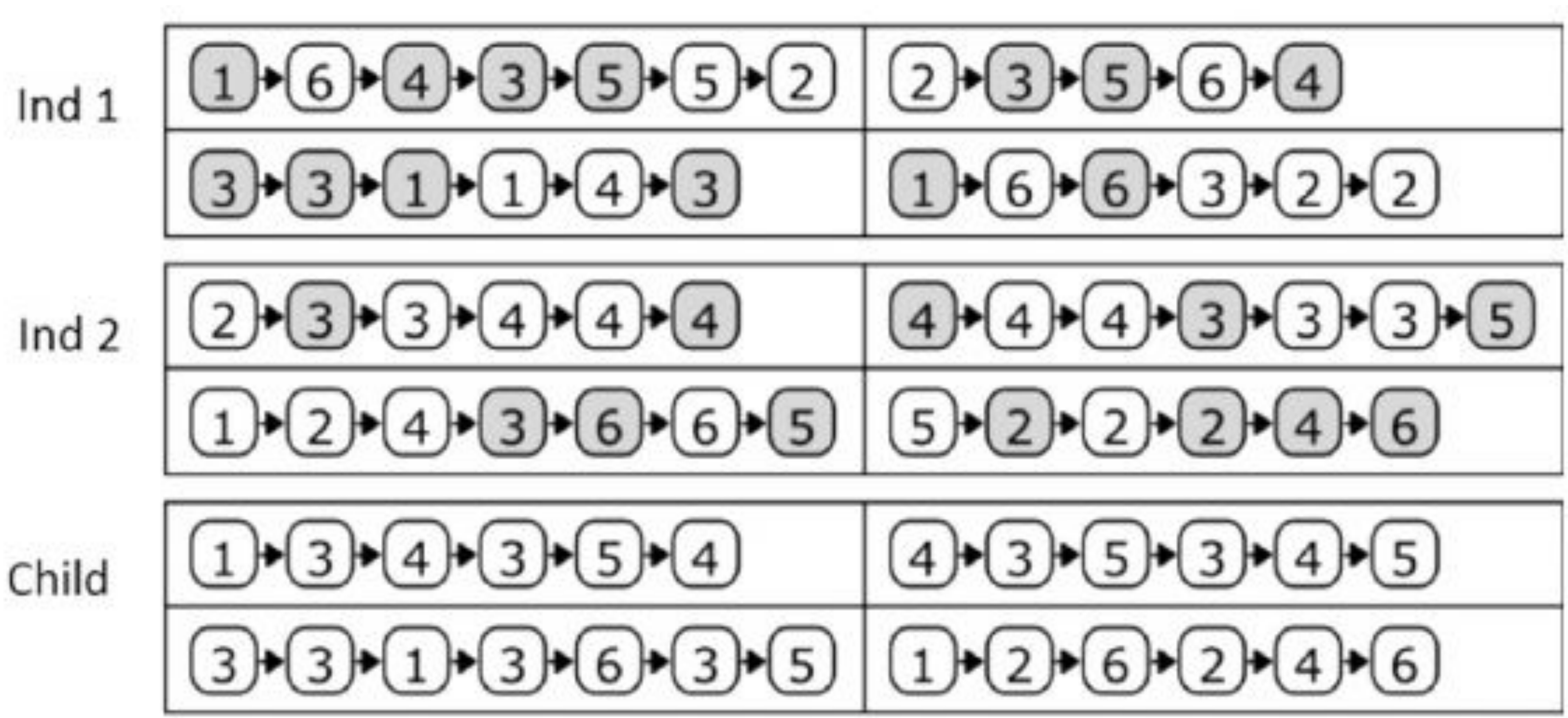



**Fig. 8.** Uniform crossover operator.

# Outras Representações e seus Operadores

International Journal on Artificial Intelligence Tools

< Previous Article

Volume 26, Issue 01, February 2017

Next Article >

Add to Favorites | Download to Citation Manager | Citation Alert

PDF (2,454 KB) | PDF Plus (1,583 KB)

Jesimar da Silva Arantes et al, *Int. J. Artif. Intell. Tools* **26**, 1760008 (2017) [30 pages]
https://doi.org/10.1142/S0218213017600089

## Heuristic and Genetic Algorithm Approaches for UAV Path Planning under Critical Situation

Jesimar da Silva Arantes[1]

Márcio da Silva Arantes[1]

Claudio Fabiano Motta Toledo[1,†]

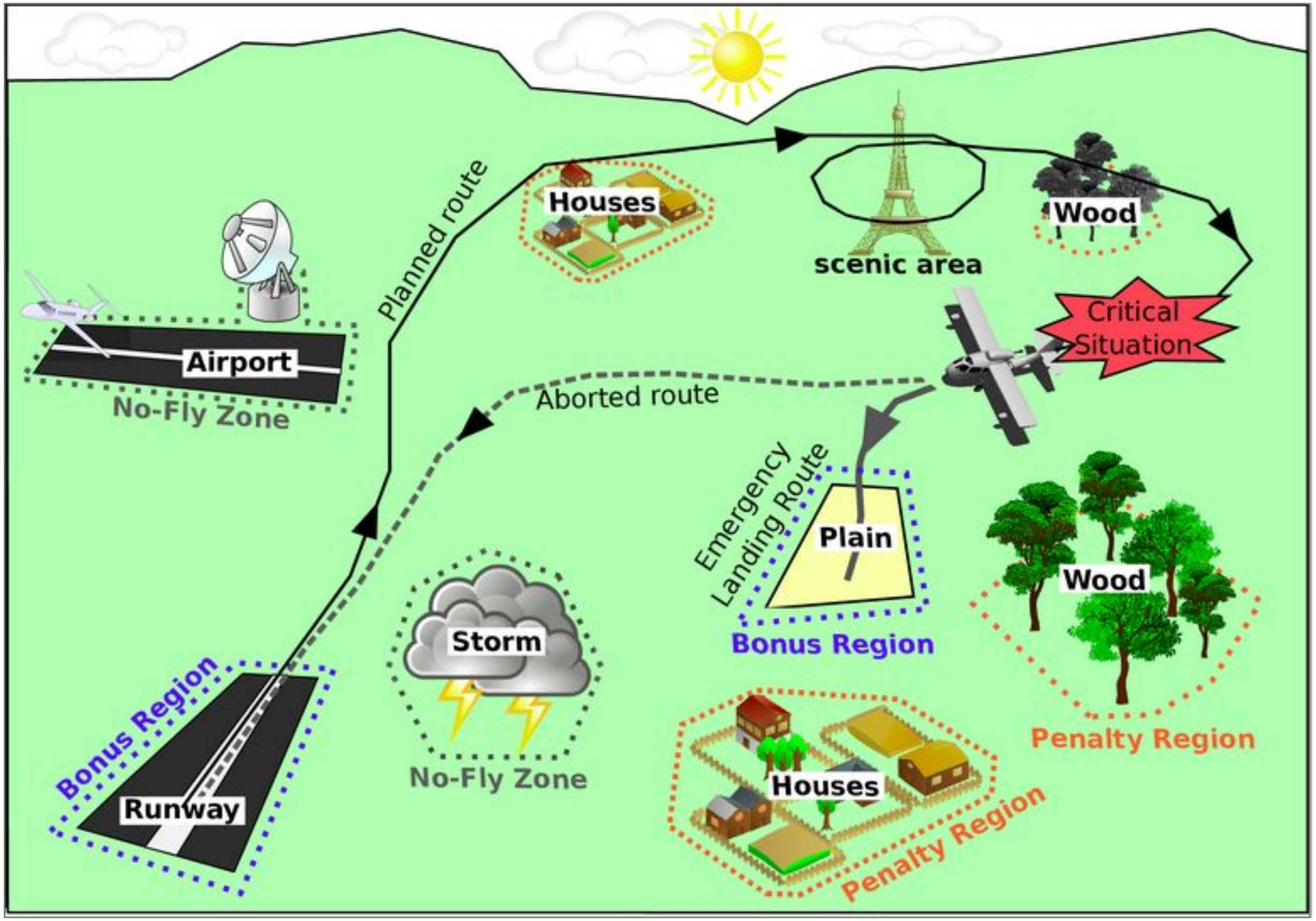
Onofre Trindade Júnior[1]

Brian Charles Williams[2]

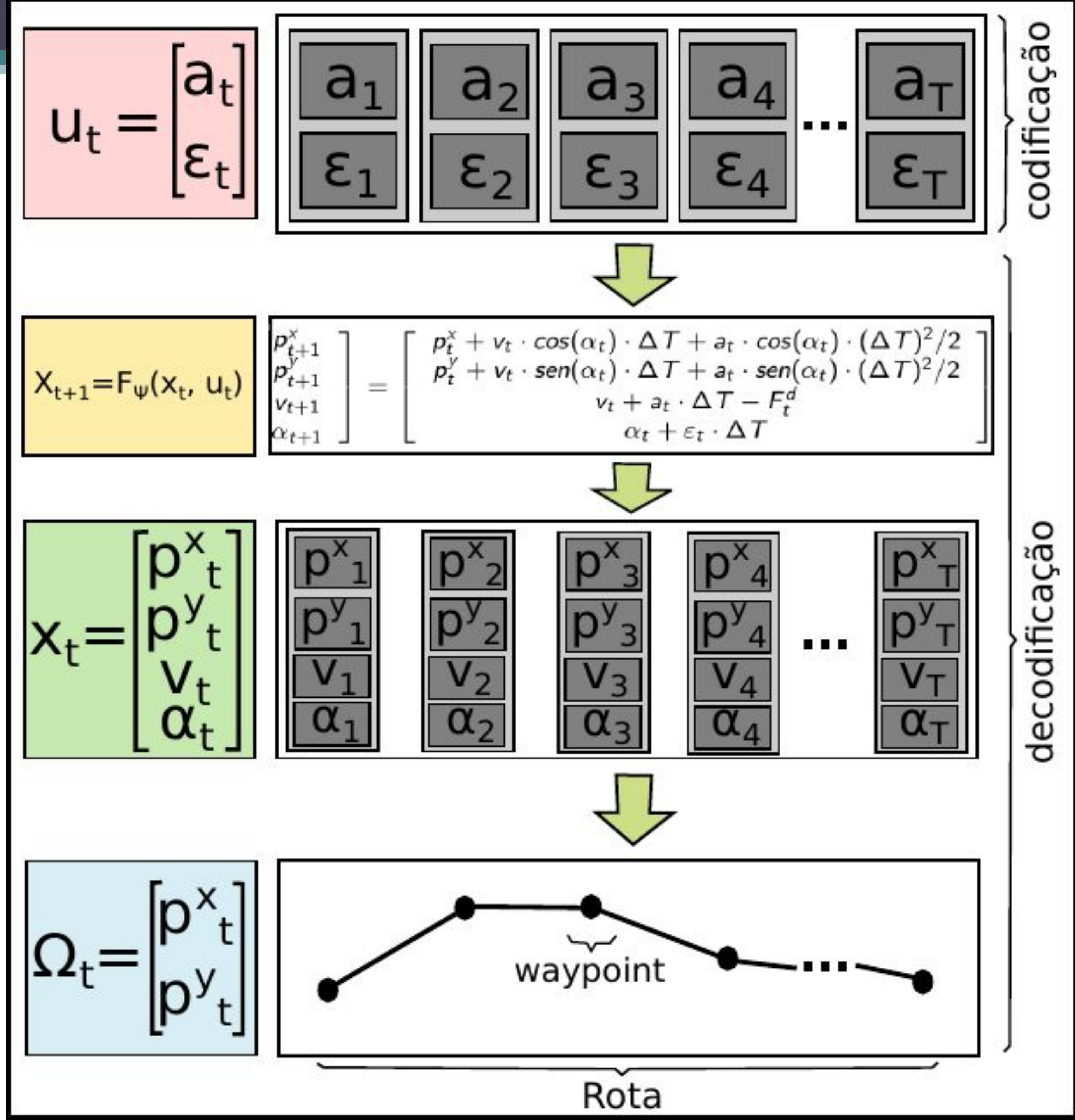
[1]University of São Paulo, USP, São Carlos, São Paulo, Brazil

[2]Massachusetts Institute of Technology, MIT, Cambridge, USA

# Planejamento de Rota para Pouso de Emergência com Alocação de Risco



# Codificação e Decodificação



# Operadores

- Crossovers:
  - Média aritmética
  - Média geométrica
  - BLX-α were applied.

- Mutação:
  - Uniforme
  - Limite
  - Creep

# Fitness

$$fitness = f_{Landing_{\phi_b}} + f_{Landing_{\phi_p}} + f_{Flight_{\phi_n}} + f_{Curves} + f_{DistUAV_{\phi_b}} + f_{Violated_T} + f_{\psi_b}$$

$$f_{Landing_{\phi_b}} = -C_{\phi_b} \cdot \sum_{i=1}^{|\phi_b|} (P(x_K \in Z^i_{\phi_b}))$$ Pouso em região bonificadora

$$f_{Landing_{\phi_p}} = C_{\phi_p} \cdot \sum_{i=1}^{|\phi_p|} (P(x_K \in Z^i_{\phi_p}))$$ Pouso em região penalizadora

# Fitness

Pouso em região NFZ

$$f_{Flight_{\phi_n}} = C_{\phi_n} \cdot max(0, 1 - \Delta - P\left(\bigwedge_{t=0}^{K} \bigwedge_{i=1}^{|\phi_n|} x_t \notin Z^i_{\phi_n}\right))$$

$$f_{Curves} = \frac{1}{|\varepsilon_{max}|} \cdot \sum_{t=0}^{K} \|u_t\| \cdot |\varepsilon_t|$$

Curvas abruptas

$$f_{DistUAV_{\phi_b}} = shortestDist(\overline{x}_K, Z_{\phi_b})$$

Distâncias longas

# Fitness

$$f_{Violated_T} = \begin{cases} C_{\phi_b} \ , v_K - v_{min} > 0 \\ 0 \ , \text{otherwise} \end{cases}$$

Alcança região bonifcadora sem pousar

$$f_\psi = \begin{cases} C_{\phi_b} \cdot 2^{\frac{(K-T)}{10}} \ , \psi = \psi_b \\ 0 \ , \text{otherwise} \end{cases}$$

Privilegia rotas com reduzido número de waypoints (falhas na bateria)

# Outras Representações e seus Operadores

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCIAIG.2017.2766218, IEEE Transactions on Computational Intelligence and AI in Games

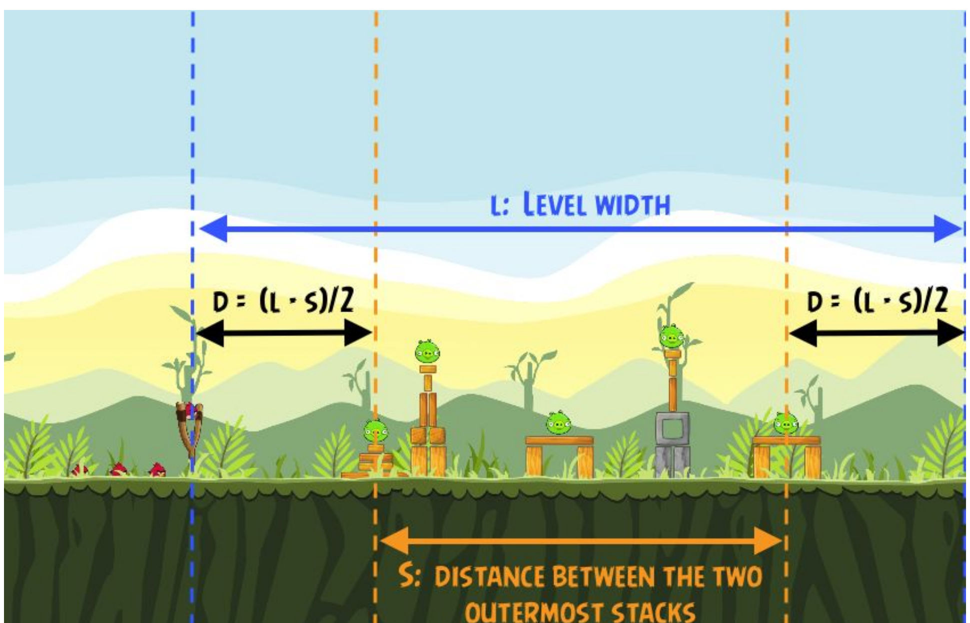


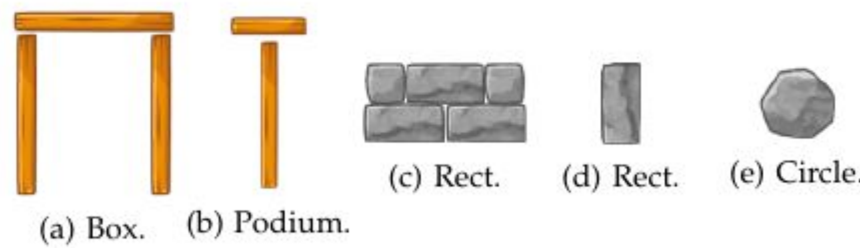
## Tanager: A Generator of Feasible and Engaging Levels for Angry Birds

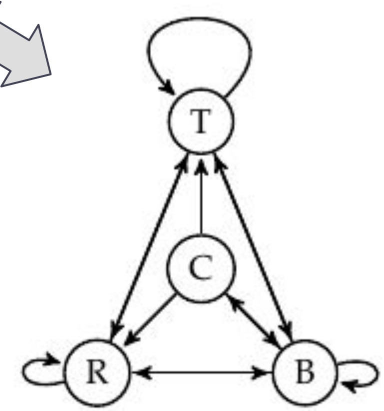
Lucas N. Ferreira and Claudio Fabiano Motta Toledo

# Codificação e Decodificação

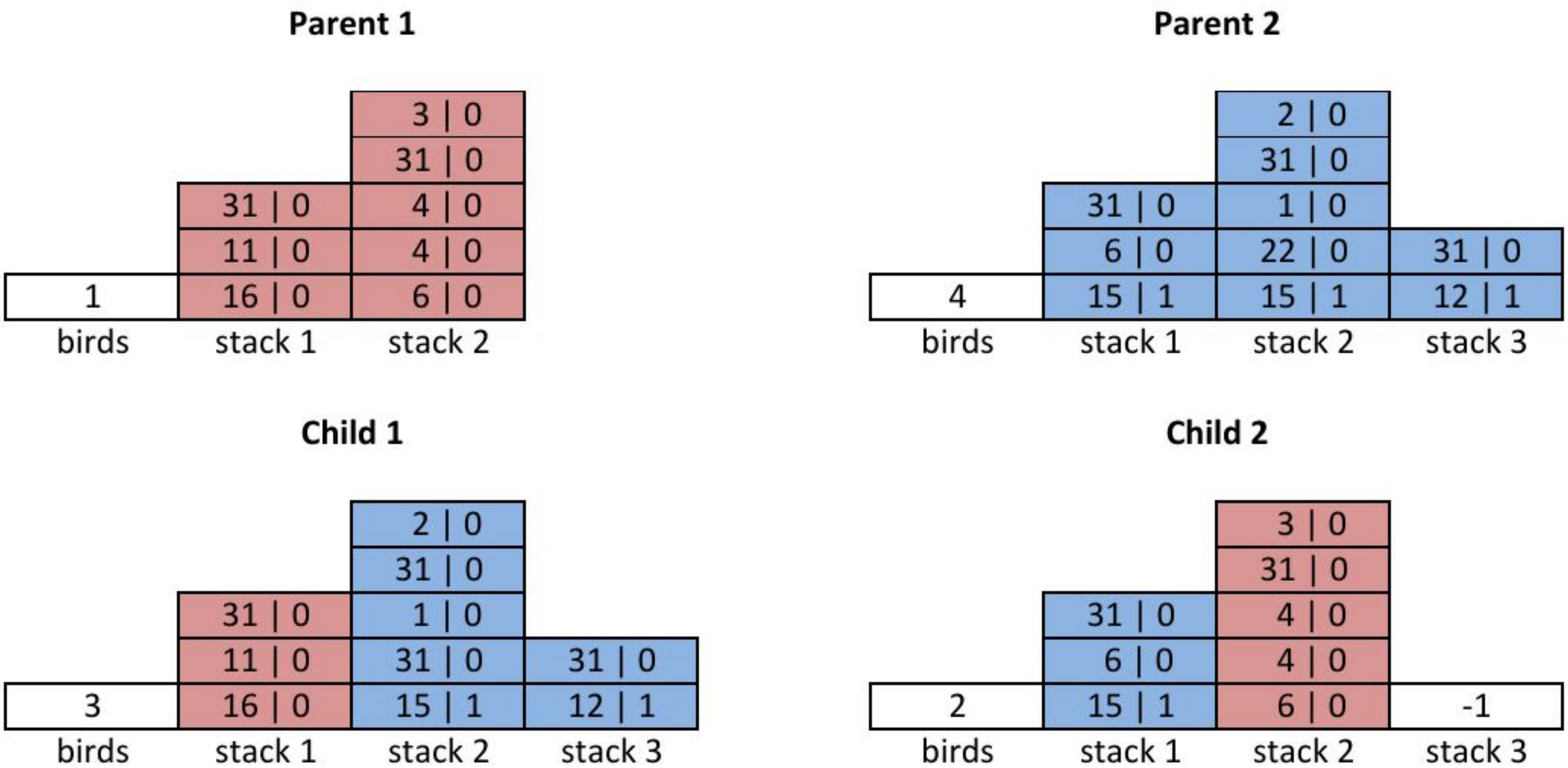
<table>
<tr><td></td><td></td><td>31 | 0</td><td></td><td></td><td></td><td>31 | 0</td><td></td><td></td></tr>
<tr><td></td><td>31 | 0</td><td>13 | 0</td><td></td><td></td><td></td><td>13 | 0</td><td></td><td></td></tr>
<tr><td></td><td>13 | 0</td><td>28 | 0</td><td></td><td></td><td></td><td>28 | 0</td><td></td><td></td></tr>
<tr><td></td><td>12 | 0</td><td>29 | 1</td><td></td><td></td><td></td><td>29 | 0</td><td></td><td></td></tr>
<tr><td></td><td>13 | 1</td><td>10 | 1</td><td></td><td>31| 0</td><td></td><td>15 | 0</td><td></td><td>31| 0</td></tr>
<tr><td>4</td><td>11 | 0</td><td>24 | 1</td><td>-1</td><td>18| 0</td><td>-1</td><td>25 | 1</td><td>-1</td><td>18| 0</td></tr>
<tr><td>birds</td><td>stack 1</td><td>stack 2</td><td>stack 3</td><td>stack 4</td><td>stack 5</td><td>stack 6</td><td>stack 7</td><td>stack 8</td></tr>
</table>



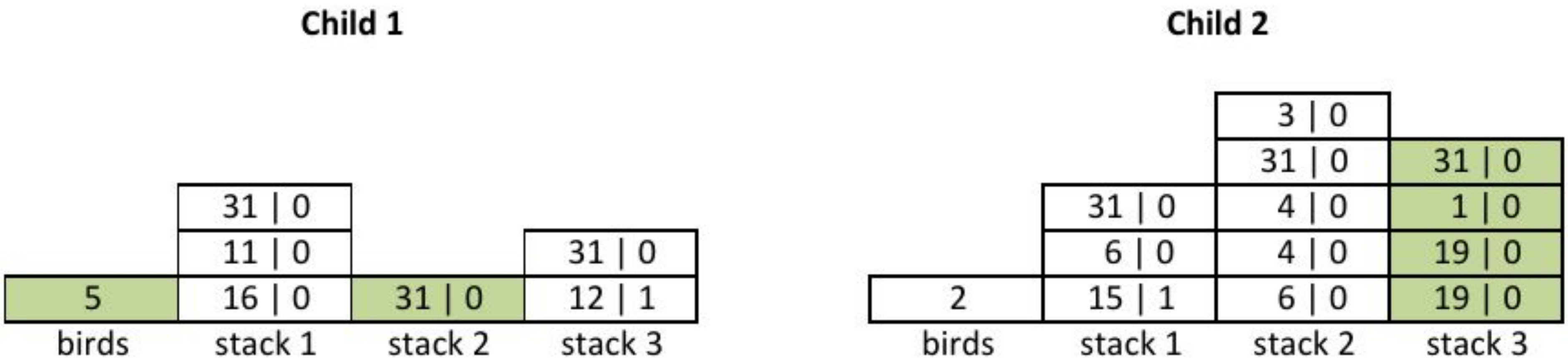



# Crossover


Parent 1
3 | 0
31 | 0
31 | 0
4 | 0
11 | 0
4 | 0
1
16 | 0
6 | 0
birds
stack 1
stack 2
Parent 2
2 | 0
31 | 0
31 | 0
1 | 0
6 | 0
22 | 0
31 | 0
4
15 | 1
15 | 1
12 | 1
birds
stack 1
stack 2
stack 3
Child 1
2 | 0
31 | 0
31 | 0
1 | 0
11 | 0
31 | 0
31 | 0
3
16 | 0
15 | 1
12 | 1
birds
stack 1
stack 2
stack 3
Child 2
3 | 0
31 | 0
31 | 0
4 | 0
6 | 0
4 | 0
2
15 | 1
6 | 0
-1
birds
stack 1
stack 2
stack 3

# Mutação


Child 1
31 | 0
11 | 0
31 | 0
5
16 | 0
31 | 0
12 | 1
birds
stack 1
stack 2
stack 3
Child 2
3 | 0
31 | 0
31 | 0
31 | 0
4 | 0
1 | 0
6 | 0
4 | 0
19 | 0
2
15 | 1
6 | 0
19 | 0
birds
stack 1
stack 2
stack 3

# Fitness

$$fitness(x) = ||\lfloor b_n * B \rfloor - B_u| + ||\lfloor l_n * L \rfloor - L_b| + p_f + s$$

- $B$: quantidade máxima de pássaros definida pelo game designer.
- $B_u$ : quantidade de pássaros utilizada durante simulação.
- $0 \leq b_n \leq 1$: valor definido pelo AE.
- $L$:quantidade máxima de blocos definida pelo game designer.
- $L_b$: quantidade de blocos no início da simulação.
- $p_f$: número de porcos ao final da simulação.
- $s$: estabilidade dos blocos.

# Outras Representações e seus Operadores

Applied Soft Computing

Volume 46, September 2016, Pages 778–791


## An approach based on hybrid genetic algorithm applied to image denoising problem

Jonatas Lopes de Paiva[a], Claudio F.M. Toledo[a], Helio Pedrini[b]

Show more

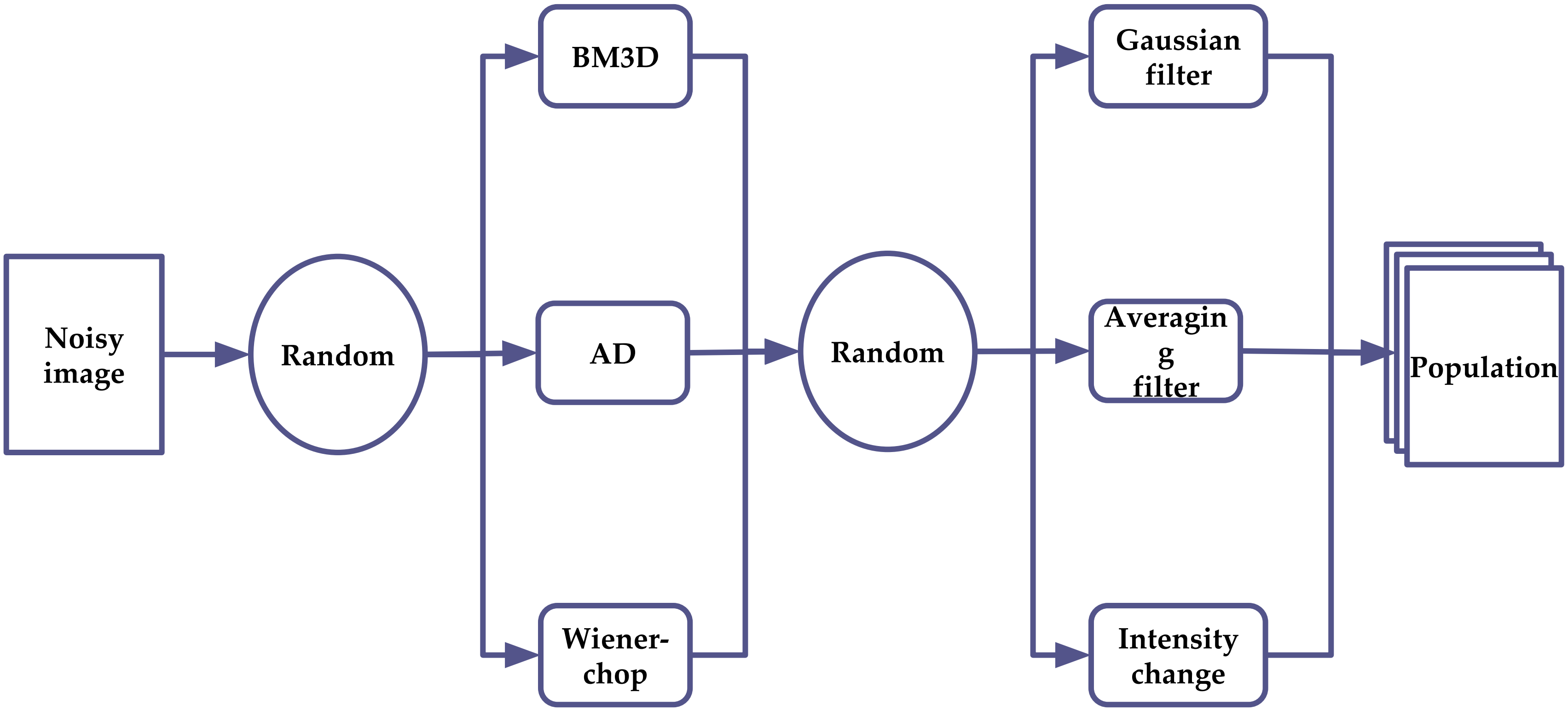
https://doi.org/10.1016/j.asoc.2015.09.013

Get rights and content

# Codificação

| | | |
|---|---|---|
| 238 | 225 | 214 |
| 240 | 233 | 228 |
| 234 | 239 | 255 |

# Inicialização



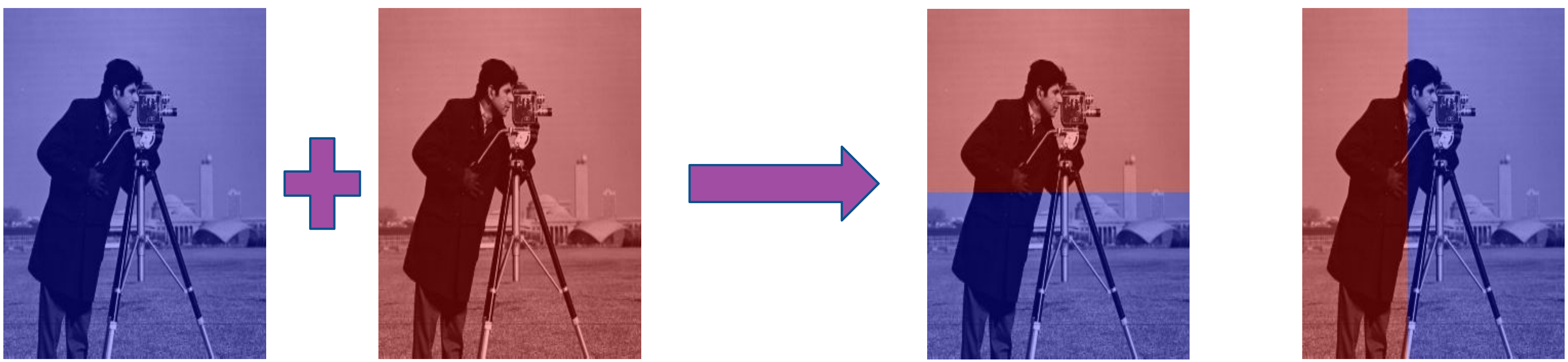
# Fitness

$$fitness(I) = \left(\sum_{\Omega} \sqrt{1 + \beta^2 |\nabla I|^2}\right) + \frac{\lambda}{2}(I - I_0)^2$$

Coeficiente de variação total

Termo de fidelização

# Crossover



Parent 1 | Parent 2 | One-point row | One-point column

# Crossover

| | | |
|---|---|---|
| 238 | 225 | 214 |
| 240 | 233 | 228 |
| 234 | 239 | 255 |

Parent 1

| | | |
|---|---|---|
| 250 | 214 | 222 |
| 241 | 229 | 248 |
| 199 | 216 | 212 |

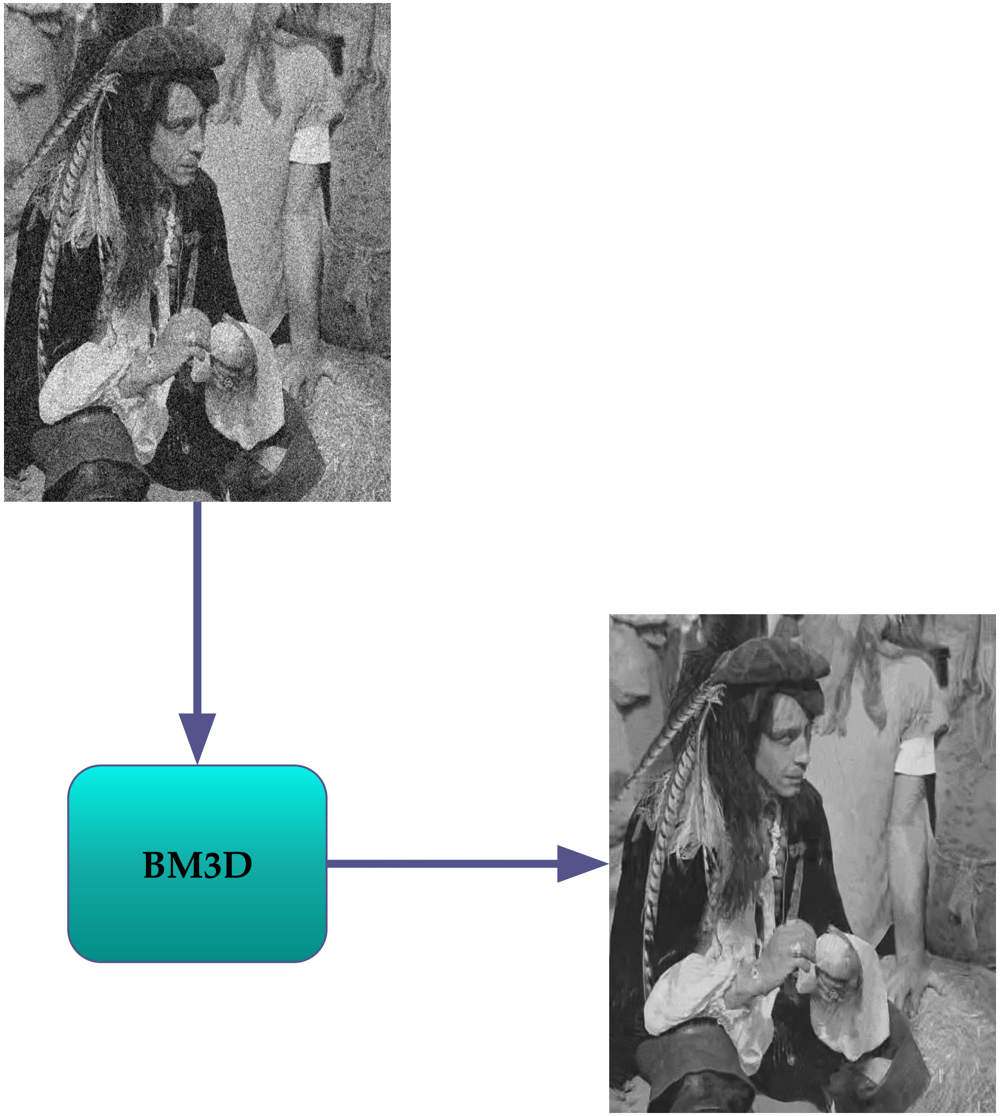
Parent 2

| | | |
|---|---|---|
| 238 | 214 | 214 |
| 241 | 229 | 228 |
| 234 | 239 | 212 |

Child

# Mutação

- Um entre três métodos de recuperação de ruídos é escolhido aleatoriamente cada vez que o operador de mutação é aplicado
  - *AD*
  - *Wiener-chop*
  - *BM3D*



# Seleção

- **Torneio**
  - Proposto por Goldberg e Deb em Goldberg and Deb (1991).
  - Seleciona aleatoriamente um número fixo de indivíduos Q e o melhor entre esses indivíduos é escolhido para cruzamento.
- **Truncamento**
  - Truncamento foi proposto por Mühlenbein e Schlierkamp-Voose em Mühlenbein and Schlierkamp-Voosen (1993).
  - Um número $M \geq N$ de indivíduos é gerado e os melhores N indivíduos são selecionados para formar a próxima população.

# Seleção - Torneio

| disputa | indivíduo | cromossomo | *fitness* | vencedor |
|---|---|---|---|---|
| 1 | 1 | 110111 | 5 | 1 |
| | 4 | 011001 | 3 | |
| 2 | 6 | 010100 | 2 | 8 |
| | 8 | 111011 | 5 | |
| 3 | 9 | 101101 | 4 | 9 |
| | 11 | 010001 | 2 | |
| 4 | 13 | 111000 | 3 | 13 |
| | 15 | 000001 | 1 | |

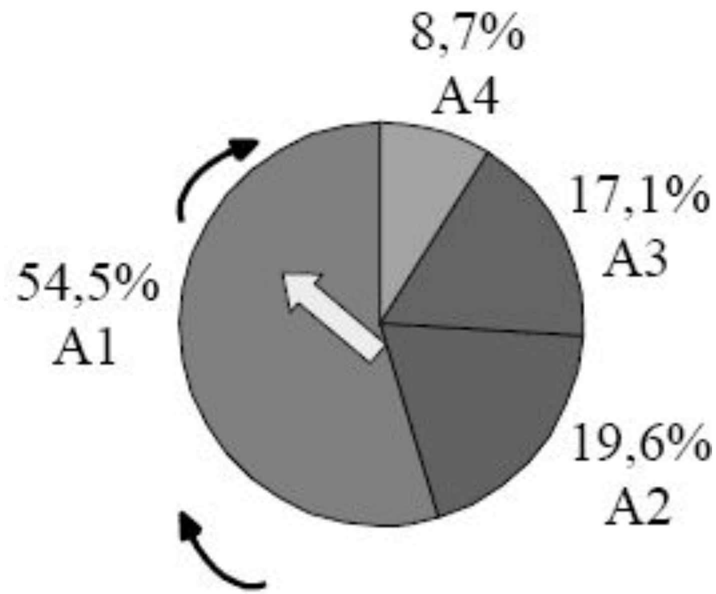
# Seleção - Torneio

- Exemplo de Algoritmo para Torneio usando Mating-Pool

1. indAtual=i=1
2. Enquanto ( indAtual≤μ)  faça
    a. Selecione k indivíduos aleatóriamente, com ou sem repetições
    b. Selecione o indivíduo com melhor valor de fitness
    c. i recebe o índice do melhor indivíduo selecionado
    d. mating-pool[indAtual]=parents[i]
    e. indAtual = indAtual + 1
3. fimEnquanto

# Seleção - Roleta

- **Mais chance para os mais aptos**




- **Problemas com valores negativos**

$$f_i = 2(N - i)/(N - 1)$$

# Seleção -Roleta

- Exemplo de Algoritmo para Roleta usando Mating-Pool

indAtual=1
total=$\sum_{i=1}^{\mu} f(i)$
**Enquanto** (indAtual ≤ μ ) **faça**
  i=1
  subTotal = 0
  selecione r∈[0,total]
  **Enquanto** (subTotal < r ) **faça**
    i=1
    subTotal = subTotal + f(i);
  **fimEnquanto**
  matingPool[indAtual] = parents[i];
  ind_atual = indAtual +1;
**fimEnquanto**

# Seleção- Roleta

## Algoritmo 2

indAtual=1
**Para** todo i=1 até μ **faça**
$\alpha[i] = \sum_{j=1}^{i} \mathrm{P}sel(j)$
**Enquanto** (indAtual ≤ μ ) **faça**
selecione r∈[0,1]
i=1
**Enquanto** (α[i] < r) **faça**
i = i+1;
**fimEnquanto**
matingPool[indAtual] = parents[i];
ind_atual = indAtual +1;
**fimEnquanto**

# Universal Sampling Algorithm

**Para** todo i=1 até μ **faça**

$\alpha[i] = \sum_{j=1}^{i} \text{P}sel(j)$

indAtual=i=1

selecione r∈[0,1/μ]

**Enquanto** (indAtual ≤ μ ) **faça**

**Enquanto** (r<α[i]) **faça**

matingPool[indAtual] = parents[i];

r = r + 1/μ

indAtual = indAtual +1

**fimEnquanto**

i = i +1;

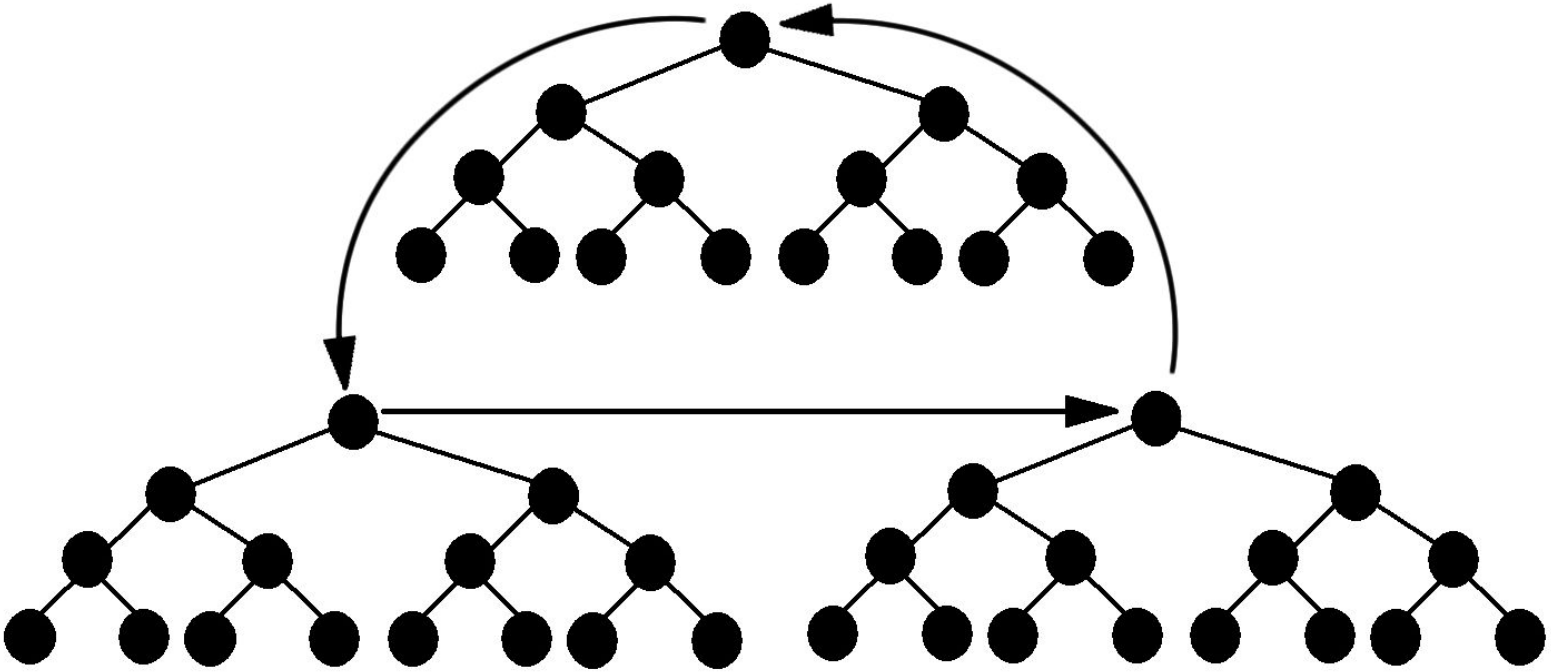
**fimEnquanto**

# Outros Operadores

- Migração
- Difusão
- Modelo farmer/worker

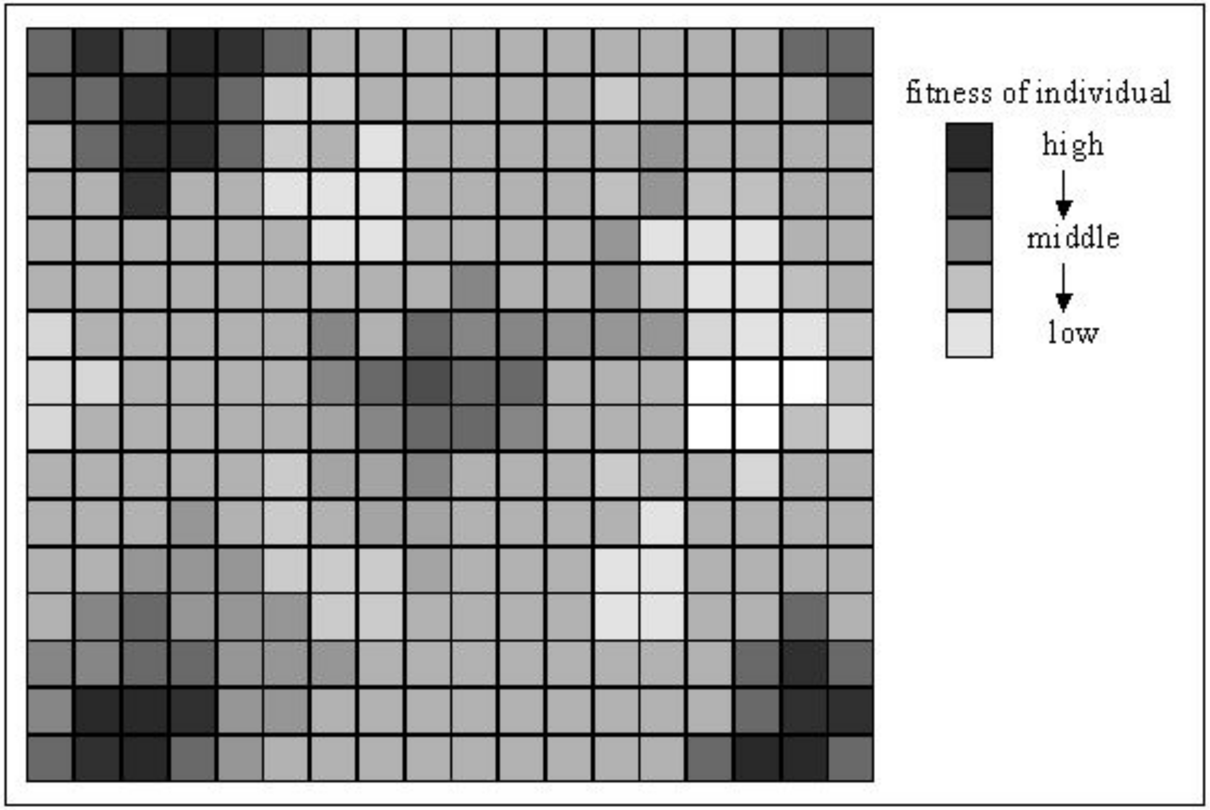
# Outros Operadores

- Migração



# Outros Operadores

- Modelo de Difusão



# Outros Operadores

- Modelo worker/farmer