Heurísticas matemáticas (*matheuristics*)

Elisangela Martins de Sá

CEFET

1º semestre, 2025

Sumário

Sumário

Métodos heurísticos

Definição

Métodos heurísticos: são algoritmos que buscam retornar uma solução de qualidade em **um tempo adequado** sem garantir a otimalidade da solução encontrada.

- Métodos construtivos: algoritmos que constroem soluções iniciais.
- Métodos de refinamento: algoritmos que buscam melhorar uma dada solução.
  - Métodos de busca local: algoritmos que buscam melhorar uma dada solução realizando uma busca em sua vizinhança.

Conveniência

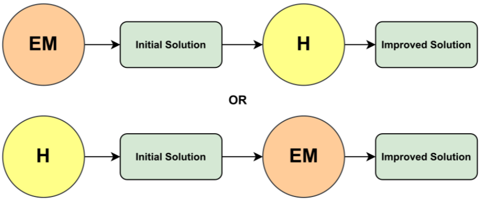
- Podem ser usadas para encontrar boas soluções em tempo razoável.
- Podem ser combinadas com métodos exatos para acelerar sua convergência.

Log do solver CPLEX

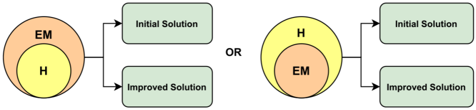
```
        Nodes                                          Cuts/
   Node  Left    Objective  IInf  Best Integer    Best Bound    ItCnt

*     0+    0                         12.0000        0.0000
*     0+    0                          6.0000        0.0000
      0     0       4.6267    10       6.0000        4.6267       47
      0     0       4.6267    11       6.0000       Cuts: 9       53
      0     0       4.6267    11       6.0000      Cuts: 12       61
*     0+    0                          5.0000        4.6267
      0     0       cutoff             5.0000        5.0000       61
Elapsed time = 0.05 sec. (3.78 ticks, tree = 0.01 MB,
solutions = 3)
Clique cuts applied:  2
Cover cuts applied:  2
Flow cuts applied:  1
Zero-half cuts applied:  1
Gomory fractional cuts applied:  1
Root node processing (before b&c):
  Real time             =    0.05 sec. (3.78 ticks)
Total (root+branch&cut) =    0.05 sec. (3.78 ticks)
```

Combinação de heurísticas (H) e métodos exatos (EM) [Ngoo et al., 2024]

Integração flexível (*loose*)



Integração forte (*tight*)



Métodos heurísticos

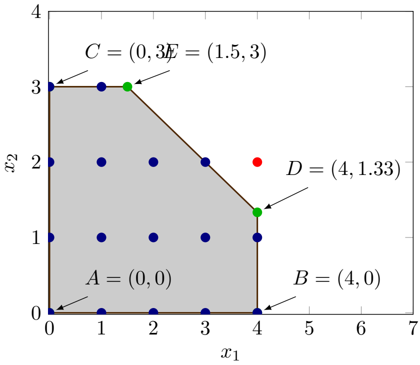
Definição

Matheurísticas ou heurísticas matemáticas (MIP heuristics): são procedimentos para encontrar boas soluções para um problema usando modelos de programação matemática [Ngoo et al., 2024].

Relaxação: exemplo

Programa Linear Inteiro

$$\begin{aligned} \max\ & 150x_1 + 200x_2 \\ \text{s. a}\ & x_1 \leq 4 \\ & x_2 \leq 3 \\ & 2x_1 + 3x_2 \leq 12 \\ & x_1, x_2 \in \mathbb{Z}_+ \end{aligned}$$



Relaxação: exemplo

Exemplo:

- Considere o seguinte problema da mochila

$$\begin{aligned} \max\ & 3x_1 + 14x_2 + 18x_3 \\ \text{s. a}\ & 3x_1 + 5x_2 + 6x_3 \leq 10 \\ & x_1, x_2, x_3 \in \{0, 1\} \end{aligned}$$

Solução relaxada

- Solução: $\bar{x} = (0;\ 0,8;\ 1)$

Problema da mochila com adição de cortes de cobertura

- Considere o seguinte problema da mochila após adição de cortes de cobertura

$$\max\ 11x_1 + 5x_2 + 18x_3 + 7x_4 + 8x_5 \quad (1)$$
$$\text{s. a}\ \ 5x_1 + 6x_2 + 6x_3 + 4x_4 + 3x_5 \leq 12 \quad (2)$$
$$x_1 + x_3 + x_5 \leq 2 \quad (3)$$
$$x_3 + x_4 + x_5 \leq 2 \quad (4)$$
$$x_1, x_2, x_3, x_4, x_5 \in \{0, 1\} \quad (5)$$

- Solução da relaxação linear: $\bar{x} = [0,5; 0; 1; 0,5; 0,5]$
  - $\bar{x}_1 = 0,5$; $\bar{x}_2 = 0$; $\bar{x}_3 = 1$; $\bar{x}_4 = 0,5$; $\bar{x}_5 = 0,5$

Problema do caixeiro viajante

Relaxação linear

- Função objetivo: 9
- Solução x_{ij}

O\D	1	2	3	4	5
1				1	
2			0.4		0.6
3		1			
4	0.6				0.4
5	0.4		0.6		

Problema do caixeiro viajante

Relaxação linear

- Função objetivo: 9
- Solução x_{ij}

O\D	1	2	3	4	5
1				1	
2			0.4		0.6
3		1			
4	0.6				0.4
5	0.4		0.6		

Dados de entrada

- Depósito: nó 1
- Matriz de custo

O\D	1	2	3	4	5
1	0	5	2	2	6
2	4	0	1	4	4
3	2	1	0	1	5
4	1	1	5	0	2
5	3	5	1	5	0

Formulação: Problema bin-packing

$$\min \sum_{j=1}^{m} y_j \tag{6}$$

$$\sum_{j=1}^{m} x_{ij} = 1 \qquad \forall i \in \{1, \ldots, n\} \tag{7}$$

$$\sum_{i=1}^{n} w_i x_{ij} \leq C y_j \qquad \forall j \in \{1, \ldots, m\} \tag{8}$$

$$y_j \in \{0, 1\}, x_{ij} \in \{0, 1\} \tag{9}$$

Problema bin-packing relaxado

Considere que cada caixa tem capacidade $C = 12$ e 7 caixas candidatas

Objetos	1	2	3	4	5	6	7	Total
w_i	3	5	7	6	2	4	7	34

Solução do problema relaxado (valores não nulos)

Caixas		1	2	3	4	5
y_j		0.17	1	0.58	0.08	1
x_{ij}	1					1
	2	0.4				0.6
	3			1		
	4		0.83		0.17	
	5					1
	6					1
	7		1			

Sumário

Relaxation Enforced Neighbourhood Search (RENS)

Notação

- Seja x o vetor de variáveis de decição
- Seja I o conjunto de índices das variáveis inteiras

Procedimento RENS

1. Resolver a relaxação linear do problema original (PL)
   - obter uma solução relaxada $\bar{x}$
   - encontrar $F \subset I$, tal que $\bar{x}_j \notin \mathbb{Z}$, para todo $j \in F$
2. Realizar os arredondamentos, correspondentes a x_j para $j \in F$, usando a solução do subproblema reduzido $P(\bar{x})$
   - $P(\bar{x})$ resulta da fixação das variáveis x_j em $\bar{x}_j$ para $j \in I \setminus F$.

Exemplo 1: Problema Bin-packing

$$\min \sum_{j=1}^{m} y_j \tag{10}$$

$$\sum_{j=1}^{m} x_{ij} = 1 \qquad \forall i \in \{1, \ldots, n\} \tag{11}$$

$$\sum_{i=1}^{n} w_i x_{ij} \leq C y_j \qquad \forall j \in \{1, \ldots, m\} \tag{12}$$

$$y_j \in \{0, 1\}, x_{ij} \in \{0, 1\} \tag{13}$$

Problema bin-packing relaxado

Considere que cada caixa tem capacidade $C = 12$

Objetos	1	2	3	4	5	6	7	Total
w_i	3	5	7	6	2	4	7	34

Solução do problema relaxado

Caixas		1	2	3	4	5
y_j		0.17	1	0.58	0.08	1
x_{ij}	1					1
	2	0.4				0.6
	3			1		
	4		0.83		0.17	
	5					1
	6					1
	7		1			

- $I_y = \{1, 2, 3, 4, 5, 6, 7\}$
- $I_x = \{(i, j)\ \forall i, j = 1, \ldots, 7\}$
- $F_y = ?$
- $F_x = ?$

Exemplo 1 - Problema reduzido

Ilustração das variáveis do problema reduzido

Caixas	2	7 4	3	4	1 5 6 2		

Exemplo 1 - Problema reduzido

Ilustração das variáveis do problema reduzido

Caixas	2	7 4	3	4	1 5 6 2		

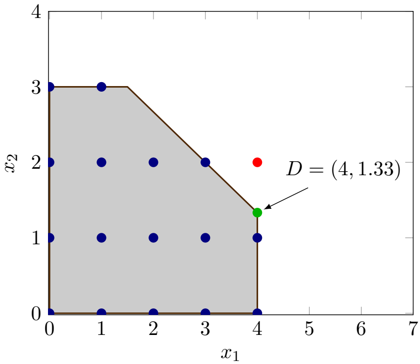
Solução do problema reduzido

Caixas	2	7	3	4	1 5 6		

Relaxação: exemplo

Programa Linear Inteiro

$$
\begin{aligned}
\max\ & 150x_1 + 200x_2 \\
\text{s. a}\ & x_1 \leq 4 \\
& x_2 \leq 3 \\
& 2x_1 + 3x_2 \leq 12 \\
& x_1, x_2 \in \mathbb{Z}_+
\end{aligned}
$$

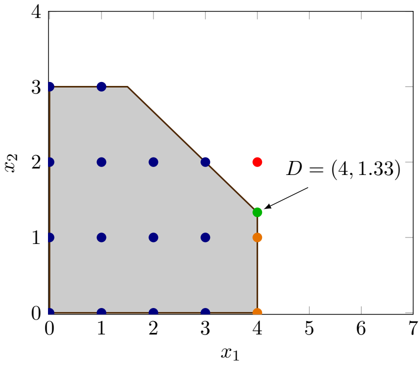


Determine o problema reduzido para a solução ótima da relaxação linear D e determine o gap entre a solução ótima e a solução encontrada via RENS.

Relaxação: exemplo

Programa Linear Inteiro

$$\begin{aligned} \max \ & 150x_1 + 200x_2 \\ \text{s. a} \ & x_1 \leq 4 \\ & x_2 \leq 3 \\ & 2x_1 + 3x_2 \leq 12 \\ & x_1, x_2 \in \mathbb{Z}_+ \end{aligned}$$



Determine o problema reduzido para a solução ótima da relaxação linear D e determine o gap entre a solução ótima e a solução encontrada via RENS.

Relaxation Enforced Neighbourhood Search (RENS)

Notação

- Problema MIP ou PI

$$(MIP \text{ ou } PI) \max_{x \in \mathbb{R}^n_+} \{c'x : Ax \leq b, x_j \in \mathbb{Z}, j \in I\}$$

- Relaxação linear

$$(PL) \max_{x \in \mathbb{R}^n_+} \{c'x : Ax \leq b, j \in I\}$$

- Seja $\bar{x}$ a solução do (PL) e $F = \{j \in I : \bar{x}_j \notin \mathbb{Z}\}$
- Problema reduzido

$$P(\bar{x}) : \max_{x \in \mathbb{R}^n_+} \left\{c'x : Ax \leq b, x_j = \bar{x}_j \text{ para } j \in I \backslash F, x_j \in \mathbb{Z} \text{ para } j \in F\right\}$$

Relaxation Induced Neighbourhood Search (RINS)

Notação

- $\tilde{x}$ é uma solução incumbente dada
- $\bar{x}$ é a solução do (PL)
- $J = \{j \in I : \tilde{x}_j = \bar{x}_j\}$
- Problema reduzido

$$P(\overline{x}, J) : \max_{x \in \mathbb{R}^n_+} \{cx : Ax \leq b, x_j = \overline{x}_j \text{ para } j \in J, x_j \in \mathbb{Z}, \forall j \in I \setminus J\}$$

Relaxation Induced Neighbourhood Search (RINS)

Notação

- $\tilde{x}$ é uma solução incumbente dada
- $\bar{x}$ é a solução do (PL)
- $J = \{j \in I : \tilde{x}_j = \bar{x}_j\}$
- Problema reduzido

$$P(\overline{x}, J) : \max_{x \in \mathbb{R}^n_+} \{cx : Ax \leq b, x_j = \overline{x}_j \text{ para } j \in J, x_j \in \mathbb{Z}, \forall j \in I \setminus J\}$$

Procedimento RINS

1. Resolver o problema (PL) e obter $\bar{x}$.
2. Realizar os arredondamentos em relação a $\overline{x}$ usando a solução do subproblema reduzido associado a $\overline{x}$ e J.

Relaxation Induced Neighbourhood Search (RINS)

Exemplo: Considere que cada caixa tem capacidade $C = 12$

Objetos	1	2	3	4	5	6	7
w_i	3	5	7	6	2	4	7

Solução incumbente:

Caixas		1	2	3	4	5
y_j		1	1	1	1	0
x_{ij}	1	1				
	2	1				
	3		1			
	4			1		
	5	1				
	6			1		
	7				1	

Solução da relaxação:

Caixas		1	2	3	4	5
y_j		1	0.8	1	0	0
x_{ij}	1	1				
	2		0.8	0.2		
	3		0.4	0.6		
	4	0.5	0.5			
	5	1				
	6	1				
	7			1		

Relaxation Induced Neighbourhood Search (RINS)

Definição do conjunto J

- $I_y \setminus J_y = \{2, 4\}$
- $I_x \setminus J_x = \{(2,1), (2,2), (2,3), (3,2), (3,3), (4,1), (4,2), (4,3), (6,1), (6,3), (7,3), (7,4)\}$

Exemplo: Considere que cada caixa tem capacidade $C = 12$

i	1	2	3	4	5	6	7
w_i	3	5	7	6	2	4	7

$$
\begin{aligned}
P(\overline{x}, J) : \min\ & y_2 + y_4 + 2 \\
& x_{11} = x_{51} = 1, \quad x_{21} + x_{22} + x_{23} = 1 \\
& x_{32} + x_{33} = 1, \quad x_{41} + x_{42} = 1 \\
& x_{61} + x_{63} = 1, \quad x_{73} + x_{74} = 1 \\
& 5x_{21} + 6x_{41} + 4x_{61} \leq 7 \\
& 5x_{22} + 7x_{32} + 6x_{42} \leq 12y_2 \\
& 5x_{23} + 7x_{33} + 4x_{63} + 7x_{73} \leq 12 \\
& 7x_{74} \leq 12y_4 \\
& y_j \in \{0, 1\}, x_{ij} \in \{0, 1\}
\end{aligned}
$$

Sumário

Exemplo - Localização de instalações

Problema de localização de instalações com 3 instalações candidatas e 4 clientes.

- y_j indica se a instalação j será aberta ou não
- x_{ij} indica se o cliente i será alocado a instalação j ou não
- Objetivo: minimizar o custo total de instalação e de transporte

Solução relaxação linear

y_j	0.25	0.5	0.25
x_{ij}	1	0	0
	0	1	0
	0	1	0
	0	0	1

Exemplo - Localização de instalações

Iteração 1

- Integralizando apenas os índices referentes a candidata 1
- Função objetivo: 28956.525

y_j	0	0.5	0.5
x_{ij}	0	0	1
	0	1	0
	0	1	0
	0	0	1

Exemplo - Localização de instalações

Iteração 2

- Integralizando os índices referentes a candidata 2
- Função objetivo: 32403.95

y_j	0	0	1
x_{ij}	0	0	1
	0	0	1
	0	0	1
	0	0	1

Exemplo - Localização de instalações

Iteração 2

- Integralizando os índices referentes a candidata 2
- Função objetivo: 32403.95

y_j	0	0	1
x_{ij}	0	0	1
	0	0	1
	0	0	1
	0	0	1

- Solução ótima - função objetivo: 31230.275

y	1	0	0
x	1	0	0
	1	0	0
	1	0	0
	1	0	0

Heurística *Relax and Fix*:Lógica do algoritmo

1. Passo 1: Particionamento das variáveis de decisão inteiras

x	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$

Heurística *Relax and Fix*:Lógica do algoritmo

1. Passo 1: Particionamento das variáveis de decisão inteiras

$$x \; \begin{array}{|ccc|ccc|ccc|} \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{Z} \\ \hline \end{array}$$

2. Passo 2
   - Relaxação

$$x \; \begin{array}{|ccc|ccc|ccc|} \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{R} & \mathbb{R} & \mathbb{R} & \mathbb{R} & \mathbb{R} & \mathbb{R} \\ \hline \end{array}$$

Heurística *Relax and Fix*:Lógica do algoritmo

1. Passo 1: Particionamento das variáveis de decisão inteiras

$$x \quad \begin{array}{|ccc|ccc|ccc|} \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{Z} \end{array}$$

2. Passo 2
   - Relaxação

$$x \quad \begin{array}{|ccc|ccc|ccc|} \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{R} & \mathbb{R} & \mathbb{R} & \mathbb{R} & \mathbb{R} & \mathbb{R} \end{array}$$

   - Solução do problema resultante

$$\tilde{x}^1 \quad \begin{array}{|ccc|ccc|ccc|} \tilde{x}_1^1 & \tilde{x}_2^1 & \tilde{x}_3^1 & \tilde{x}_4^1 & \tilde{x}_5^1 & \tilde{x}_6^1 & \tilde{x}_7^1 & \tilde{x}_8^1 & \tilde{x}_9^1 \end{array}$$

Heurística *Relax and Fix*: Lógica do algoritmo

1. Passo 2
   - Relaxação

$$x \quad \begin{array}{|ccc|ccc|ccc|} \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{R} & \mathbb{R} & \mathbb{R} & \mathbb{R} & \mathbb{R} & \mathbb{R} \end{array}$$

   - Solução do problema resultante

$$\tilde{x}^1 \quad \begin{array}{|ccc|ccc|ccc|} \tilde{x}_1^1 & \tilde{x}_2^1 & \tilde{x}_3^1 & \tilde{x}_4^1 & \tilde{x}_5^1 & \tilde{x}_6^1 & \tilde{x}_7^1 & \tilde{x}_8^1 & \tilde{x}_9^1 \end{array}$$

2. Passo 3
   - Relaxação e fixação

$$x \quad \begin{array}{|ccc|ccc|ccc|} \tilde{x}_1^1 & \tilde{x}_2^1 & \tilde{x}_3^1 & \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{R} & \mathbb{R} & \mathbb{R} \end{array}$$

Heurística *Relax and Fix*:Lógica do algoritmo

1. Passo 2
   - Relaxação

x									
	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$

   - Solução do problema resultante

$\tilde{x}^1$									
	$\tilde{x}_1^1$	$\tilde{x}_2^1$	$\tilde{x}_3^1$	$\tilde{x}_4^1$	$\tilde{x}_5^1$	$\tilde{x}_6^1$	$\tilde{x}_7^1$	$\tilde{x}_8^1$	$\tilde{x}_9^1$

2. Passo 3
   - Relaxação e fixação

x									
	$\tilde{x}_1^1$	$\tilde{x}_2^1$	$\tilde{x}_3^1$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$

   - Solução do problema resultante

$\tilde{x}^2$									
	$\tilde{x}_1^1$	$\tilde{x}_2^1$	$\tilde{x}_3^1$	$\tilde{x}_4^2$	$\tilde{x}_5^2$	$\tilde{x}_6^2$	$\tilde{x}_7^2$	$\tilde{x}_8^2$	$\tilde{x}_9^2$

Heurística *Relax and Fix*:Lógica do algoritmo

1. Passo 2
   - Relaxação

x	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$

   - Solução do problema resultante

$\tilde{x}^1$	$\tilde{x}_1^1$	$\tilde{x}_2^1$	$\tilde{x}_3^1$	$\tilde{x}_4^1$	$\tilde{x}_5^1$	$\tilde{x}_6^1$	$\tilde{x}_7^1$	$\tilde{x}_8^1$	$\tilde{x}_9^1$

2. Passo 3
   - Relaxação e fixação

x	$\tilde{x}_1^1$	$\tilde{x}_2^1$	$\tilde{x}_3^1$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$

   - Solução do problema resultante

$\tilde{x}^2$	$\tilde{x}_1^1$	$\tilde{x}_2^1$	$\tilde{x}_3^1$	$\tilde{x}_4^2$	$\tilde{x}_5^2$	$\tilde{x}_6^2$	$\tilde{x}_7^2$	$\tilde{x}_8^2$	$\tilde{x}_9^2$

3. Passo 4
   - Relaxação e fixação

x	$\tilde{x}_1^1$	$\tilde{x}_2^1$	$\tilde{x}_3^1$	$\tilde{x}_4^2$	$\tilde{x}_5^2$	$\tilde{x}_6^2$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$

Heurística *Relax and Fix*:Lógica do algoritmo

1. Passo 2
   - Relaxação

x	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$

   - Solução do problema resultante

$\tilde{x}^1$	$\tilde{x}_1^1$	$\tilde{x}_2^1$	$\tilde{x}_3^1$	$\tilde{x}_4^1$	$\tilde{x}_5^1$	$\tilde{x}_6^1$	$\tilde{x}_7^1$	$\tilde{x}_8^1$	$\tilde{x}_9^1$

2. Passo 3
   - Relaxação e fixação

x	$\tilde{x}_1^1$	$\tilde{x}_2^1$	$\tilde{x}_3^1$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{R}$	$\mathbb{R}$	$\mathbb{R}$

   - Solução do problema resultante

$\tilde{x}^2$	$\tilde{x}_1^1$	$\tilde{x}_2^1$	$\tilde{x}_3^1$	$\tilde{x}_4^2$	$\tilde{x}_5^2$	$\tilde{x}_6^2$	$\tilde{x}_7^2$	$\tilde{x}_8^2$	$\tilde{x}_9^2$

3. Passo 4
   - Relaxação e fixação

x	$\tilde{x}_1^1$	$\tilde{x}_2^1$	$\tilde{x}_3^1$	$\tilde{x}_4^2$	$\tilde{x}_5^2$	$\tilde{x}_6^2$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$

   - Solução do problema resultante

$\tilde{x}^3$	$\tilde{x}_1^1$	$\tilde{x}_2^1$	$\tilde{x}_3^1$	$\tilde{x}_4^2$	$\tilde{x}_5^2$	$\tilde{x}_6^2$	$\tilde{x}_7^3$	$\tilde{x}_8^3$	$\tilde{x}_9^3$

Relax and Fix no problema bin packing - Resultados

Problema relaxado - função objetivo: 2.83

y	0.47222	0.47222	0.47222	0.47222	0.47222	0.47222
x	0.16667	0.16667	0.16667	0.16667	0.16667	0.16667
	0.16667	0.16667	0.16667	0.16667	0.16667	0.16667
	0.16667	0.16667	0.16667	0.16667	0.16667	0.16667
	0.16667	0.16667	0.16667	0.16667	0.16667	0.16667
	0.16667	0.16667	0.16667	0.16667	0.16667	0.16667
	0.16667	0.16667	0.16667	0.16667	0.16667	0.16667
	0.16667	0.16667	0.16667	0.16667	0.16667	0.16667

Relax and Fix no problema bin packing - Resultados

- Iteração 1
- Integralizando índices referentes às caixa 1 e 2
- Função objetivo: 2.833333333

y	0	0	0.70833	0.70833	0.70833	0.70833
x	0	0	0.25	0.25	0.25	0.25
	0	0	0.25	0.25	0.25	0.25
	0	0	0.25	0.25	0.25	0.25
	0	0	0.25	0.25	0.25	0.25
	0	0	0.25	0.25	0.25	0.25
	0	0	0.25	0.25	0.25	0.25
	0	0	0.25	0.25	0.25	0.25

Relax and Fix no problema bin packing - Resultados

- Iteração 2
- Integralizando índices referentes às caixa 3 e 4
- Função objetivo: 2.833333333

y	0	0	1	1	0.41667	0.41667
x	0	0	0	0	0.5	0.5
	0	0	0	1	0	0
	0	0	0	0	0.5	0.5
	0	0	1	0	0	0
	0	0	1	0	0	0
	0	0	1	0	0	0
	0	0	0	1	0	0

Relax and Fix no problema bin packing - Resultados

- Iteração 3
- Integralizando índices referentes às caixa 5 e 6
- Função objetivo: 3

y	0	0	1	1	1	0
x	0	0	0	0	1	0
	0	0	0	1	0	0
	0	0	0	0	1	0
	0	0	1	0	0	0
	0	0	1	0	0	0
	0	0	1	0	0	0
	0	0	0	1	0	0

Heurística Relax and Fix

Algorithm 1 Heurística Relax and Fix

Entradas: $MIP(P \cap \mathbb{Z}^{|I|} \times \mathbb{R}^{n-|I|})$,
Partições I_t para $t \in \{1, .., T\}$

Saída : x^*

for $t \in \{1, .., T\}$ **do**
  for $j \in I$ **do**
    if $j \in I_t$ **then**
      Faça $x_j \in \mathbb{Z}$
    end
    else
      Faça $x_j \in \mathbb{R}$ *para* $j \in I_k, \forall k = t+1, \ldots, T$
    end
  end
  $\bar{x} \leftarrow$ *Solução ótima do* $MIP(P)$ *resultante*
  $P = P \cap \{x : x_j = \bar{x}_j, \forall j \in I_t\}$
end
$x^* \leftarrow \bar{x}$
Retorne x^*

Particionamento para problema de Localização de instalações

Proposta de particionamento

- Seja $J = \{1, \ldots, n\}$ o conjunto das instalações candidatas
- Seja NP o número de partes que serão criadas e TP o tamanho de cada parte, assumindo partes homogêneas.
- Particionamento: Distribuir os elementos de J em sequência para cada uma das NP partes.
  - Se j pertence à parte p, então as variáveis y_j e x_{ij} pertenceriam a esta partição para todo i.

Exemplo:

- $J = \{1, 2, 3, 4, 5, 6\}$
- Se $NP = 2$, então $TP = 3$
  - Parte $1 = \{1, 2, 3\} : y_1,\ y_2,\ y_3,\ x_{i1},\ x_{i2},\ x_{i3}, \forall i$
  - Parte $2 = \{4, 5, 6\} : y_4,\ y_5,\ y_6,\ x_{i4},\ x_{i5},\ x_{i6}, \forall i$

Particionamento para problema de Localização de instalações

Proposta de particionamento

- Seja $J = \{1, \ldots, n\}$ o conjunto das instalações candidatas
- Seja NP o número de partes que serão criadas e TP o tamanho de cada parte, assumindo partes homogêneas.
- Particionamento: Distribuir os elementos de J em sequência para cada uma das NP partes.
  - Se j pertence à parte p, então as variáveis y_j e x_{ij} pertenceriam a esta partição para todo i.

Exemplo:

- $J = \{1, 2, 3, 4, 5, 6\}$
- Se $NP = 3$, então $TP = 2$
  - Parte $1 = \{1, 2\} : y_1,\ y_2,\ x_{i1},\ x_{i2}, \forall i$
  - Parte $2 = \{3, 4\} : y_3,\ y_4,\ x_{i3},\ x_{i4}, \forall i$
  - Parte $3 = \{5, 6\} : y_5,\ y_6,\ x_{i5},\ x_{i6}, \forall i$

Relax and Fix - Localização de facilidades - 100 instalações e 5 partes

```
                 Iteração 1
Valor da solução relaxada 50.6255
Facilidades instaladas:
                 Iteração 2
Valor da solução relaxada 56.3516
Facilidades instaladas:
                 Iteração 3
Valor da solução relaxada 63.97
Facilidades instaladas:
                 Iteração 4
Valor da solução relaxada 87.5696
Facilidades instaladas:
                 Iteração 5
Valor da solução relaxada 183.788
Facilidades instaladas: 82 85 89 96
Tempo 20.21
```

Relax and Fix - Localização de facilidades - 100 instalações e 10 partes

```
                 Iteração 1
Valor da solução relaxada 49.3027
Facilidades instaladas:
                 Iteração 2
Valor da solução relaxada 50.6255
Facilidades instaladas:
                Iteração 3
Valor da solução relaxada 53.1623
Facilidades instaladas:
                 Iteração 4
Valor da solução relaxada 56.3516
Facilidades instaladas:
                 Iteração 5
Valor da solução relaxada 58.4608
Facilidades instaladas:
Tempo 1.96
```

Relax and Fix - Localização de facilidades - 100 instalações e 5 partes

```
                 Iteração 6
Valor da solução relaxada 63.97
Facilidades instaladas:
                 Iteração 7
Valor da solução relaxada 74.4656
Facilidades instaladas:
                 Iteração 8
Valor da solução relaxada 87.5696
Facilidades instaladas:
                 Iteração 9
Valor da solução relaxada 126.478
Facilidades instaladas: 82
                 Iteração 10
Valor da solução relaxada 200.009
Facilidades instaladas: 82 94 95 96
Tempo 7.53
```

Resultados - Relax and fix

Table 1: Comparação dos resultados de três particionamentos do Relax-and-fix: 5 partes, 10 partes e 20 partes com o resultado usando apenas o CPLEX limitado a 2 minutos.

I.	5		10		20		CPX
	F.O	Tim[s]	F.O	Tim[s]	F.O	Tim[s]	F.O
a	**183.79**	20.21	200.01	8.54	203.51	**6.39**	308.36
b	**139.68**	6.63	145.47	**4.32**	147.80	5.57	151.32
c	**123.33**	8.09	124.68	**4.94**	130.54	5.57	152.66

Heurística *Fix and Optimize* - Lógica do algoritmo

1. Passo 1: Particione o vetor de variáveis de variáveis de decisão inteiras
   - Vetor de variáveis de decisão

     $$x \quad \begin{array}{|ccc|ccc|ccc|} \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \mathbb{Z} \\ \hline \end{array}$$

   - Solução incumbente inicial

     $$\tilde{x}^0 \quad \begin{array}{|ccc|ccc|ccc|} \tilde{x}_1^0 & \tilde{x}_2^0 & \tilde{x}_3^0 & \tilde{x}_4^0 & \tilde{x}_5^0 & \tilde{x}_6^0 & \tilde{x}_7^0 & \tilde{x}_8^0 & \tilde{x}_9^0 \\ \hline \end{array}$$

Heurística *Fix and Optimize* - Lógica do algoritmo

1. Passo 1: Particione o vetor de variáveis de variáveis de decisão inteiras
   - Vetor de variáveis de decisão

x	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$

   - Solução incumbente inicial

$\tilde{x}^0$	$\tilde{x}_1^0$	$\tilde{x}_2^0$	$\tilde{x}_3^0$	$\tilde{x}_4^0$	$\tilde{x}_5^0$	$\tilde{x}_6^0$	$\tilde{x}_7^0$	$\tilde{x}_8^0$	$\tilde{x}_9^0$

2. Passo 2
   - Fixar e otimizar (Parte 1)

x	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\tilde{x}_4^0$	$\tilde{x}_5^0$	$\tilde{x}_6^0$	$\tilde{x}_7^0$	$\tilde{x}_8^0$	$\tilde{x}_9^0$

Heurística *Fix and Optimize* - Lógica do algoritmo

1. Passo 1: Particione o vetor de variáveis de variáveis de decisão inteiras
    - Vetor de variáveis de decisão

x	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$

    - Solução incumbente inicial

$\tilde{x}^0$	$\tilde{x}_1^0$	$\tilde{x}_2^0$	$\tilde{x}_3^0$	$\tilde{x}_4^0$	$\tilde{x}_5^0$	$\tilde{x}_6^0$	$\tilde{x}_7^0$	$\tilde{x}_8^0$	$\tilde{x}_9^0$

2. Passo 2
    - Fixar e otimizar (Parte 1)

x	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\tilde{x}_4^0$	$\tilde{x}_5^0$	$\tilde{x}_6^0$	$\tilde{x}_7^0$	$\tilde{x}_8^0$	$\tilde{x}_9^0$

    - Solução do problema resultante

$\tilde{x}^1$	$\tilde{x}_1^1$	$\tilde{x}_2^1$	$\tilde{x}_3^1$	$\tilde{x}_4^0$	$\tilde{x}_5^0$	$\tilde{x}_6^0$	$\tilde{x}_7^0$	$\tilde{x}_8^0$	$\tilde{x}_9^0$

Heurística *Fix and Optimize* - Lógica do algoritmo

1. Passo 2
   - Fixar e otimizar (Parte 1)

     $$x \quad \begin{array}{|ccc|ccc|ccc|} \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \tilde{x}_4^0 & \tilde{x}_5^0 & \tilde{x}_6^0 & \tilde{x}_7^0 & \tilde{x}_8^0 & \tilde{x}_9^0 \end{array}$$

   - Solução do problema resultante

     $$\tilde{x}^1 \quad \begin{array}{|ccc|ccc|ccc|} \tilde{x}_1^1 & \tilde{x}_2^1 & \tilde{x}_3^1 & \tilde{x}_4^0 & \tilde{x}_5^0 & \tilde{x}_6^0 & \tilde{x}_7^0 & \tilde{x}_8^0 & \tilde{x}_9^0 \end{array}$$

2. Passo 3
   - Fixar e otimizar (Parte 2)

     $$x \quad \begin{array}{|ccc|ccc|ccc|} \tilde{x}_1^1 & \tilde{x}_2^1 & \tilde{x}_3^1 & \mathbb{Z} & \mathbb{Z} & \mathbb{Z} & \tilde{x}_7^0 & \tilde{x}_8^0 & \tilde{x}_9^0 \end{array}$$

Heurística *Fix and Optimize* - Lógica do algoritmo

1. Passo 2
   - Fixar e otimizar (Parte 1)

x	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\tilde{x}_4^0$	$\tilde{x}_5^0$	$\tilde{x}_6^0$	$\tilde{x}_7^0$	$\tilde{x}_8^0$	$\tilde{x}_9^0$

   - Solução do problema resultante

$\tilde{x}^1$	$\tilde{x}_1^1$	$\tilde{x}_2^1$	$\tilde{x}_3^1$	$\tilde{x}_4^0$	$\tilde{x}_5^0$	$\tilde{x}_6^0$	$\tilde{x}_7^0$	$\tilde{x}_8^0$	$\tilde{x}_9^0$

2. Passo 3
   - Fixar e otimizar (Parte 2)

x	$\tilde{x}_1^1$	$\tilde{x}_2^1$	$\tilde{x}_3^1$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\tilde{x}_7^0$	$\tilde{x}_8^0$	$\tilde{x}_9^0$

   - Solução do problema resultante

$\tilde{x}^2$	$\tilde{x}_1^1$	$\tilde{x}_2^1$	$\tilde{x}_3^1$	$\tilde{x}_4^2$	$\tilde{x}_5^2$	$\tilde{x}_6^2$	$\tilde{x}_7^0$	$\tilde{x}_8^0$	$\tilde{x}_9^0$

Heurística *Fix and Optimize* - Lógica do algoritmo

1. Passo 3
   - Fixar e otimizar (Parte 2)

x	$\tilde{x}_1^1$	$\tilde{x}_2^1$	$\tilde{x}_3^1$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$	$\tilde{x}_7^0$	$\tilde{x}_8^0$	$\tilde{x}_9^0$

   - Solução do problema resultante

$\tilde{x}^2$	$\tilde{x}_1^1$	$\tilde{x}_2^1$	$\tilde{x}_3^1$	$\tilde{x}_4^2$	$\tilde{x}_5^2$	$\tilde{x}_6^2$	$\tilde{x}_7^0$	$\tilde{x}_8^0$	$\tilde{x}_9^0$

2. Passo 4
   - Fixar e otimizar (Parte 3)

x	$\tilde{x}_1^1$	$\tilde{x}_2^1$	$\tilde{x}_3^1$	$\tilde{x}_4^2$	$\tilde{x}_5^2$	$\tilde{x}_6^2$	$\mathbb{Z}$	$\mathbb{Z}$	$\mathbb{Z}$

   - Solução do problema resultante

$\tilde{x}^3$	$\tilde{x}_1^1$	$\tilde{x}_2^1$	$\tilde{x}_3^1$	$\tilde{x}_4^2$	$\tilde{x}_5^2$	$\tilde{x}_6^2$	$\tilde{x}_7^3$	$\tilde{x}_8^3$	$\tilde{x}_9^3$

Heurística Fix and Optimize

Algorithm 2 Heurística com uma iteração de Fix and Optimize

Entradas: $MIP(P \cap \mathbb{Z}^{|I|} \times \mathbb{R}^{n-|I|}), \tilde{x}$
Partições I_t para $t \in \{1, .., T\}$
Saída : x^*
$\tilde{P} = P \cap \{x : x_j = \tilde{x}_j, \forall j \in I\}$
for $t \in \{1, .., T\}$ **do**
  if $j \in I_t$ **then**
    $\tilde{P} \leftarrow \tilde{P} \setminus \{x_j = \tilde{x}_j\}$
  end
  $\tilde{x} \leftarrow$ *Solução ótima do* $MIP(\tilde{P})$
  for $j \in I_t$ **do**
    $\tilde{P} \leftarrow \tilde{P} \cap \{x_j = \tilde{x}_j\}$
  end
end
$x^* \leftarrow \tilde{x}$
Retorne x^*

Observação

- O algoritmo representa apenas uma iteração de Fix and Optimize.

Fix and optimize (adaptado)

```
Loop principal Fix and Optimize (82 94 95 96)
                 Iteração 1: 1-10
Valor da solução incumbente 200.009
Facilidades instaladas: 82 94 95 96
                 Iteração 2: 11-20
Valor da solução incumbente 193.58
Facilidades instaladas: 11 15 94 96
                 Iteração 3: 21-30
Valor da solução incumbente 189.672
Facilidades instaladas: 15 20 21 26
                 Iteração 4: 31-40
Valor da solução incumbente 184.655
Facilidades instaladas: 15 20 32 35
                 Iteração 5: 41-50
Valor da solução incumbente 184.655
Facilidades instaladas: 15 20 32 35
```

Fix and optimize (adaptado)

```
                 Iteração 6: 51-60
Valor da solução incumbente 180.396
Facilidades instaladas: 35 53 54 58
                 Iteração 7: 61-70
Valor da solução incumbente 180.15
Facilidades instaladas: 53 54 58 69
                 Iteração 8: 71-80
Valor da solução incumbente 176.862
Facilidades instaladas: 53 58 69 78
                 Iteração 9: 81-90
Valor da solução incumbente 176.581
Facilidades instaladas: 58 78 82 85
                 Iteração 10: 91-100
Valor da solução incumbente 176.581
Facilidades instaladas: 58 78 82 85
```

Fix and optimize (adaptado)- Localização de instalação - capa.txt

```
                   Iteração 11: 1-10
Valor da solução incumbente 176.581
Facilidades instaladas: 58 78 82 85

                   Iteração 12: 11-20
Valor da solução incumbente 175.925
Facilidades instaladas: 15 58 78 85

                  Iteração 13: 21-30
Parte = { 21 22 23 24 25 26 27 28 29 30 15 58 78 85}
Valor da solução incumbente 175.925
Facilidades instaladas: 15 58 78 85
```

Resultados - fix and optimize

Table 2: Comparação dos resultados de três particionamentos: 5 partes, 10 partes e 20 partes com o resultado usando apenas o CPLEX limitado a 2 minutos.

I	RF		FixOpt - 1 iter		FixOpt		CPX	z^*
I	F.O	T[s]	F.O	T[s]	F.O	T[s]	F.O	F.O
a	200.01	8.5	177.22	66.9	171.61	120.0	308.36	171.56
b	145.47	4.3	139.53	16.2	132.45	120.0	151.32	129.79
c	124.68	4.9	120.24	13.2	115.38	120.0	152.66	115.06

Sumário

Definições: Hamming distance

Definição

Distância de Hamming (Hamming distance) entre dois vetores binários fornece o número de posições em que estes vetores se diferem.

- Distância de Hamming

$$\delta(x^a, x^b) = \sum_{j \in I} |x_j^a - x_j^b| = \sum_{j \in I} [x_j^a(1 - x_j^b) + x_j^b(1 - x_j^a)]$$

Exemplo:

a) $x^a = [1, 0, 1, 0, 0], \; x^b = [0, 0, 1, 0, 1]$
b) $x^a = [1, 0, 1, 0, 0], \; x^b = [0, 1, 0, 1, 1]$
c) $x^a = [1, 0, 1, 0, 0], \; x^b = [1, 0, 1, 0, 0]$
d) $x^a = [1, 0, 1, 1, 0], \; x^b = [1, 1, 1, 1, 0]$

Definição de estrutura de vizinhança

Definição

- Vizinhança $\mathcal{N}^k(\tilde{x}) = \{x \in X : \delta(x, \tilde{x}) \leq k\}$

Exemplo:

1) Seja $x^a = [1, 0, 1]$, determine:
   a) $\mathcal{N}^1(x^a) =$
   b) $\mathcal{N}^2(x^a) =$

Exemplo: Problema da mochila

j	Peso	Util.
1	6	10
2	4	3
3	12	12
4	5	4
5	1	1

- Conjunto de soluções viáveis X
- Solução incumbente: $\tilde{x} = (0, 0, 1, 0, 0)$

Exemplo: Problema da mochila

j	Peso	Util.
1	6	10
2	4	3
3	12	12
4	5	4
5	1	1

- Conjunto de soluções viáveis X
- Solução incumbente: $\tilde{x} = (0, 0, 1, 0, 0)$
- Existe uma solução melhor que $\tilde{x}$ em $\mathcal{N}^1(\tilde{x})$?

Exemplo: Problema da mochila

j	Peso	Util.
1	6	10
2	4	3
3	12	12
4	5	4
5	1	1

- Conjunto de soluções viáveis X
- Solução incumbente: $\tilde{x} = (0, 0, 1, 0, 0)$
- Existe uma solução melhor que $\tilde{x}$ em $\mathcal{N}^1(\tilde{x})$?
- Existe uma solução melhor que $\tilde{x}$ em $\mathcal{N}^2(\tilde{x})$?

Exemplo: Problema da mochila

j	Peso	Util.
1	6	10
2	4	3
3	12	12
4	5	4
5	1	1

- Conjunto de soluções viáveis X
- Solução incumbente: $\tilde{x} = (0, 0, 1, 0, 0)$
- Existe uma solução melhor que $\tilde{x}$ em $\mathcal{N}^1(\tilde{x})$?
- Existe uma solução melhor que $\tilde{x}$ em $\mathcal{N}^2(\tilde{x})$?
- Existe uma solução melhor que $\tilde{x}$ em $\mathcal{N}^3(\tilde{x})$?

Exemplo: Problema da mochila

j	Peso	Util.
1	6	10
2	4	3
3	12	12
4	5	4
5	1	1

- Conjunto de soluções viáveis X
- Solução incumbente: $\tilde{x} = (0, 0, 1, 0, 0)$
- Existe uma solução melhor que $\tilde{x}$ em $\mathcal{N}^1(\tilde{x})$?
- Existe uma solução melhor que $\tilde{x}$ em $\mathcal{N}^2(\tilde{x})$?
- Existe uma solução melhor que $\tilde{x}$ em $\mathcal{N}^3(\tilde{x})$?
- Defina $\mathcal{N}^k(\tilde{x})$ usando programação linear.

Definição de estrutura de vizinhança

1) (Problema bin packing) Considere que cada caixa tem capacidade $C = 12$

Objetos	1	2	3	4
w_i	3	5	7	6

Solução:

$$
\begin{array}{rcccc}
\bar{y} = & 1 & 1 & 1 & 1 \\
\bar{x} = & 1 & 0 & 0 & 0 \\
 & 0 & 1 & 0 & 0 \\
 & 0 & 0 & 1 & 0 \\
 & 0 & 0 & 0 & 1
\end{array}
$$

determine, se existir, alguma solução diferente da solução corrente em $\mathcal{N}^1(\bar{y}, \bar{x}), \mathcal{N}^2(\bar{y}, \bar{x})$ e $\mathcal{N}^3(\bar{y}, \bar{x})$.

Definição de estrutura de vizinhança

1) (Problema bin packing) Considere que cada caixa tem capacidade $C = 12$

Objetos	1	2	3	4
w_i	3	5	7	6

Solução:

$$
\begin{array}{lcccc}
\bar{y} = & 1 & 1 & 1 & 1 \\
\bar{x} = & 1 & 0 & 0 & 0 \\
 & 0 & 1 & 0 & 0 \\
 & 0 & 0 & 1 & 0 \\
 & 0 & 0 & 0 & 1
\end{array}
$$

determine, se existir, alguma solução diferente da solução corrente em $\mathcal{N}^1(\bar{y}, \bar{x}), \mathcal{N}^2(\bar{y}, \bar{x})$ e $\mathcal{N}^3(\bar{y}, \bar{x})$.

Definição

- Vizinhança $\mathcal{N}^k(\bar{y}) = \{(y, x) \in YX : \delta((y, x), (\bar{y}, \bar{x})) \leq k\}$

Definição de estrutura de vizinhança

1) (Problema bin packing) Considere que cada caixa tem capacidade $C = 12$

Objetos	1	2	3	4
w_i	3	5	7	6

Solução:

$$\begin{array}{lcccc} \bar{y} = & 1 & 1 & 1 & 1 \\ \bar{x} = & 1 & 0 & 0 & 0 \\ & 0 & 1 & 0 & 0 \\ & 0 & 0 & 1 & 0 \\ & 0 & 0 & 0 & 1 \end{array}$$

Considere a seguinte estrutura de vizinhança

- Vizinhança $\mathcal{N}^k(\bar{y}) = \{(y, x) \in YX : \delta(y, \bar{y}) \leq k\}$ ou seja

$$\mathcal{N}^k(\bar{y}) = \{(y, x) \in YX : (1-y_1)+(1-y_2)+(1-y_3)+(1-y_4) \leq k\} \tag{1}$$

determine, se existir, alguma solução melhor que a solução corrente em $\mathcal{N}^1(\bar{y})$

Exemplo 2: Problema da mochila

Problema inicial

$$\max \quad 2x_1 + 2x_2 + 3x_3 + 4x_4 + 6x_5 + 5x_6 + 8x_7 + 7x_8 \tag{1}$$
$$s.a \; 4x_1 + 15x_2 + 7x_3 + 9x_4 + 8x_5 + 10x_6 + 9x_7 + 11x_8 \leq 32 \tag{2}$$
$$x_j \in \{0, 1\} \tag{3}$$

- Solução inicial $x^0 = (1, 1, 1, 0, 0, 0, 0, 0)$
- Defina o subproblema a melhor na vizinhança

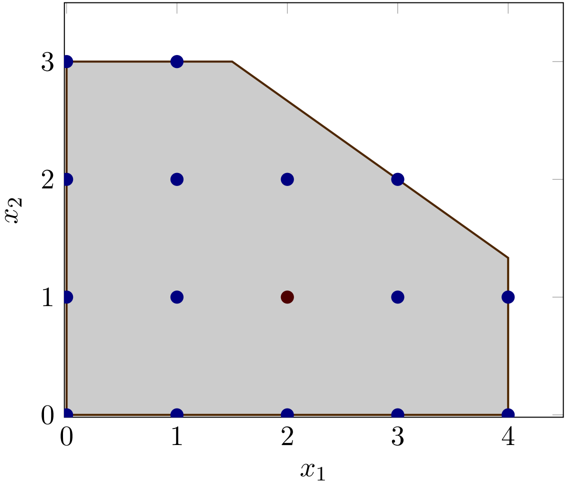
$$\mathcal{N}^2(x^0)$$

Definição de estrutura de vizinhança

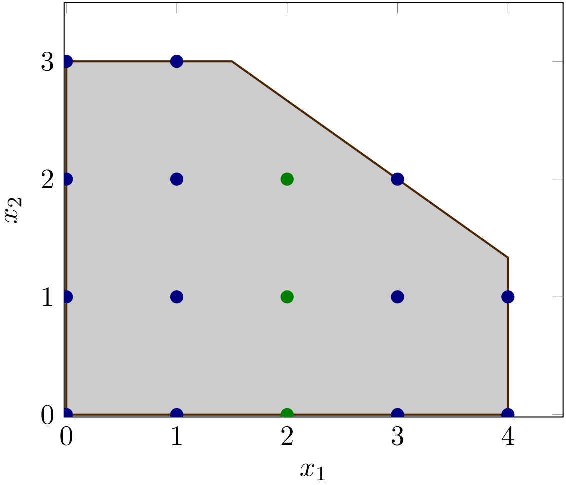
Tipos de fixação

- *Hard fixing:* fixação de variáveis em um valor
- *Soft fixing:* adição do pseudo-corte $\delta(x, \tilde{x}) \leq k$, que fixa o valor de algumas variáveis de forma menos específica

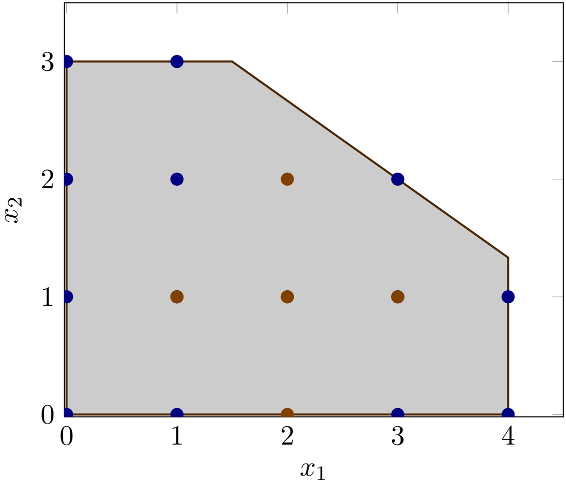
Hard fixing e Soft Fixing



Hard fixing e Soft Fixing



Hard fixing e Soft Fixing



Local branching [Fischetti and Lodi, 2003]

$$(MIP) \min \{c'x : x \in X, x_j \in \{0,1\}, j \in I\}$$

Pressupostos:

- Estrutura de vizinhança $\mathcal{N}^k(.) \subset X$
- $X = (X \cap \mathcal{N}^k(\bar{x})) \cup (X \setminus \mathcal{N}^k(\bar{x}))$

Local branching [Fischetti and Lodi, 2003]

$$(MIP) \min \{c'x : x \in X, x_j \in \{0,1\}, j \in I\}$$

Pressupostos:

- Estrutura de vizinhança $\mathcal{N}^k(.) \subset X$
- $X = (X \cap \mathcal{N}^k(\bar{x})) \cup (X \setminus \mathcal{N}^k(\bar{x}))$

Algoritmo Local Branching

- Passo 1: Inicialização
  - Solução inicial viável $\bar{x}^0$, distância k.
  - $Y \leftarrow X$
  - $h \leftarrow 0$

Local branching

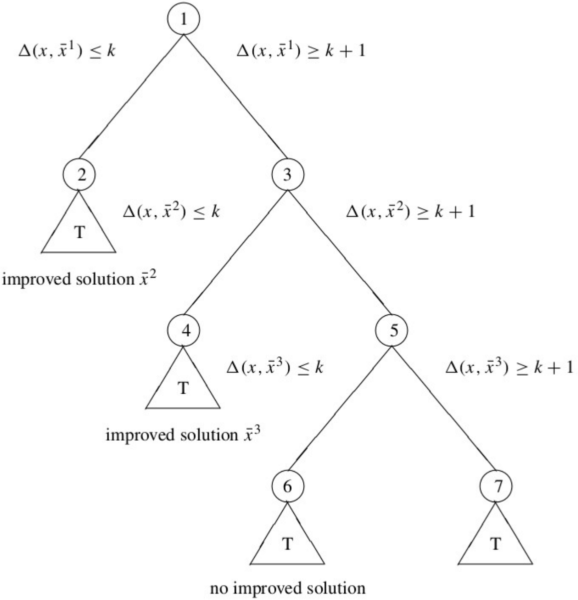
Algoritmo Local Branching Básico ($\bar{x}^0$, k, X)

- Passo 1: Inicialização
  - $Y \leftarrow X$
  - $h \leftarrow 0$
- Passo 2: Resolva o $MIP(Y \cap \mathcal{N}^k(\bar{x}^h))$
- Passo 3: Se a solução ótima $x' \in Y \cap \mathcal{N}^k(\bar{x}^h)$ é melhor que $\bar{x}^h$, então
  - $Y = Y \setminus \mathcal{N}^k(\bar{x}^h)$.
  - $h \leftarrow h + 1$
  - $\bar{x}^h \leftarrow \bar{x}'$
  - Se critério de parada não tiver sido atingido, volte ao Passo 2

  Caso contrário

  - $Y = Y \setminus \mathcal{N}^k(\bar{x}^h)$
  - Resolva o problema $MIP(Y)$ e obtenha sua solução ótima x''
  - Pare
- Retorne a melhor solução entre $\bar{x}^h$ e x''

Local branching básico



Critério de parada

- Parar após todos os problemas serem resolvidos (Método Exato)
- Parar após um dado número de iterações (Matheurística)
- Impor um tempo limite de resolução (Matheurística)

Local branching com tempo limite

Algorithm 3 LocalBranching($\bar{x}, k^0$, tempo limite)

$k \leftarrow k_0$, $Y \leftarrow X$, $x^* \leftarrow \bar{x}$;
enquanto *Critério de parada não atendido* **faça**
  Resolver $MIP(Y \cap \mathcal{N}^k(\tilde{x}), \text{tempo limite})$
  caso *Nenhuma solução viável obtida* **faça** $k \leftarrow k - \lceil \frac{k}{2} \rceil$;
  caso *Solução ótima x' encontrada* **faça**
    $Y = Y \setminus \mathcal{N}^k(\tilde{x})$
    $\tilde{x} \leftarrow x'$ e $k \leftarrow k_0$
    se $c'x' < c'x^*$ **então** $x^* \leftarrow x'$;
  fim
  caso *Solução viável encontrada* **faça**
    $Y = Y \setminus \mathcal{N}^0(\tilde{x})$ e $\tilde{x} \leftarrow x'$
  fim
  caso $Y = \emptyset$ **faça** $Y = Y \setminus \mathcal{N}^k(\tilde{x})$ e aplique uma diversificação;
fim
Retorne x^*

Exemplo 2: Problema da mochila

Problema inicial

$$
\begin{aligned}
\max \;\; & 2x_1 + 2x_2 + 3x_3 + 4x_4 + 6x_5 + 5x_6 + 8x_7 + 7x_8 && (1)\\
s.a \;\; & 4x_1 + 15x_2 + 7x_3 + 9x_4 + 8x_5 + 10x_6 + 9x_7 + 11x_8 \leq 32 && (2)\\
& x_j \in \{0, 1\} && (3)
\end{aligned}
$$

Considere a solução inicial $x^0 = (1, 0, 0, 1, 0, 1, 1, 0)$ com valor de f.o. $z^0 = 19$

Exemplo 2: Problema da mochila

Problema inicial

$$\max \quad 2x_1 + 2x_2 + 3x_3 + 4x_4 + 6x_5 + 5x_6 + 8x_7 + 7x_8 \tag{1}$$
$$s.a \; 4x_1 + 15x_2 + 7x_3 + 9x_4 + 8x_5 + 10x_6 + 9x_7 + 11x_8 \leq 32 \tag{2}$$
$$x_j \in \{0, 1\} \tag{3}$$

Considere a solução inicial $x^0 = (1, 0, 0, 1, 0, 1, 1, 0)$ com valor de f.o. $z^0 = 19$

Iteração 1: $k = 3$

$$\begin{aligned} \max \quad & 2x_1 + 2x_2 + 3x_3 + 4x_4 + 6x_5 + 5x_6 + 8x_7 + 7x_8 \\ s.a \; & 4x_1 + 15x_2 + 7x_3 + 9x_4 + 8x_5 + 10x_6 + 9x_7 + 11x_8 \leq 32 \\ & (1 - x_1) + x_2 + x_3 + (1 - x_4) + x_5 + (1 - x_6) + (1 - x_7) + x_8 \leq 3 \\ & x_j \in \{0, 1\} \end{aligned}$$

Solução $x' = (1, 0, 0, 0, 1, 1, 1, 0)$ e $z' = 21$

Exemplo 2: Problema da mochila

Iteração 1: $k = 3$

$$\begin{aligned} \max \quad & 2x_1 + 2x_2 + 3x_3 + 4x_4 + 6x_5 + 5x_6 + 8x_7 + 7x_8 \\ s.a \quad & 4x_1 + 15x_2 + 7x_3 + 9x_4 + 8x_5 + 10x_6 + 9x_7 + 11x_8 \leq 32 \\ & (1 - x_1) + x_2 + x_3 + (1 - x_4) + x_5 + (1 - x_6) + (1 - x_7) + x_8 \leq 3 \\ & x_j \in \{0, 1\} \end{aligned}$$

Solução $x' = (1, 0, 0, 0, 1, 1, 1, 0)$ e $z' = 21$

Iteração 2: $k = 3$

$$\begin{aligned} \max \quad & 2x_1 + 2x_2 + 3x_3 + 4x_4 + 6x_5 + 5x_6 + 8x_7 + 7x_8 \\ s.a \quad & 4x_1 + 15x_2 + 7x_3 + 9x_4 + 8x_5 + 10x_6 + 9x_7 + 11x_8 \leq 32 \\ & (1 - x_1) + x_2 + x_3 + (1 - x_4) + x_5 + (1 - x_6) + (1 - x_7) + x_8 \geq 4 \\ & (1 - x_1) + x_2 + x_3 + x_4 + (1 - x_5) + (1 - x_6) + (1 - x_7) + x_8 \leq 3 \\ & x_j \in \{0, 1\} \end{aligned}$$

Solução $x' = (1, 0, 0, 0, 1, 0, 1, 1)$ e $z' = 23$

Exemplo 2: Problema da mochila

Iteração 2: $k = 3$

$$
\begin{aligned}
\max\ & 2x_1 + 2x_2 + 3x_3 + 4x_4 + 6x_5 + 5x_6 + 8x_7 + 7x_8 \\
s.a\ & 4x_1 + 15x_2 + 7x_3 + 9x_4 + 8x_5 + 10x_6 + 9x_7 + 11x_8 \leq 32 \\
& (1 - x_1) + x_2 + x_3 + (1 - x_4) + x_5 + (1 - x_6) + (1 - x_7) + x_8 \geq 4 \\
& (1 - x_1) + x_2 + x_3 + x_4 + (1 - x_5) + (1 - x_6) + (1 - x_7) + x_8 \leq 3 \\
& x_j \in \{0, 1\}
\end{aligned}
$$

Solução $x' = (1, 0, 0, 0, 1, 0, 1, 1)$ e $z' = 23$

Iteração 3: $k = 3$

$$
\begin{aligned}
\max\ & 2x_1 + 2x_2 + 3x_3 + 4x_4 + 6x_5 + 5x_6 + 8x_7 + 7x_8 \\
s.a\ & 4x_1 + 15x_2 + 7x_3 + 9x_4 + 8x_5 + 10x_6 + 9x_7 + 11x_8 \leq 32 \\
& (1 - x_1) + x_2 + x_3 + (1 - x_4) + x_5 + (1 - x_6) + (1 - x_7) + x_8 \geq 4 \\
& (1 - x_1) + x_2 + x_3 + x_4 + (1 - x_5) + (1 - x_6) + (1 - x_7) + x_8 \geq 4 \\
& (1 - x_1) + x_2 + x_3 + x_4 + (1 - x_5) + x_6 + (1 - x_7) + (1 - x_8) \leq 3 \\
& x_j \in \{0, 1\}
\end{aligned}
$$

Exemplo 1: Local branching aplicado ao bin packing

(Problema bin packing) Considere que cada caixa tem capacidade $C = 12$

Objetos	1	2	3	4
w_i	3	5	7	6

- Solução inicial

$$
\begin{array}{lcccc}
\bar{y} = & 1 & 1 & 1 & 1 \\
\bar{x} = & 1 & 0 & 0 & 0 \\
 & 0 & 1 & 0 & 0 \\
 & 0 & 0 & 1 & 0 \\
 & 0 & 0 & 0 & 1
\end{array}
$$

- Vizinhança
  $\mathcal{N}^k(\bar{y}) = \{(y, x) \in YX : \delta(y, \bar{y}) \leq k\}$

Exemplo 1: Local branching aplicado ao bin packing

(Problema bin packing) Considere que cada caixa tem capacidade $C = 12$

Objetos	1	2	3	4
w_i	3	5	7	6

- Solução inicial

$$\begin{array}{rcccc} \bar{y} = & 1 & 1 & 1 & 1 \\ \bar{x} = & 1 & 0 & 0 & 0 \\ & 0 & 1 & 0 & 0 \\ & 0 & 0 & 1 & 0 \\ & 0 & 0 & 0 & 1 \end{array}$$

- Vizinhança
$\mathcal{N}^k(\bar{y}) = \{(y,x) \in YX : \delta(y,\bar{y}) \leq k\}$

Iter 1 $k = 1$

Solução:

$$\begin{array}{rcccc} \tilde{y} = & 1 & 1 & 1 & 0 \\ \tilde{x} = & 1 & 0 & 0 & 0 \\ & 0 & 1 & 0 & 0 \\ & 0 & 0 & 1 & 0 \\ & 0 & 1 & 0 & 0 \end{array}$$

Exemplo 1: Local branching aplicado ao bin packing

(Problema bin packing) Considere que cada caixa tem capacidade $C = 12$

Objetos	1	2	3	4
w_i	3	5	7	6

- Vizinhança
  $\mathcal{N}^k(\bar{y}) = \{(y, x) \in YX : \delta(y, \bar{y}) \leq k\}$

Iter 1 $k = 1$

Solução:

$$\begin{array}{lcccc} \tilde{y} = & 1 & 1 & 1 & 0 \\ \tilde{x} = & 1 & 0 & 0 & 0 \\ & 0 & 1 & 0 & 0 \\ & 0 & 0 & 1 & 0 \\ & 0 & 1 & 0 & 0 \end{array}$$

Iter 2 $k = 1$

Solução:

$$\begin{array}{lcccc} \tilde{y} = & 1 & 1 & 0 & 0 \\ \tilde{x} = & 1 & 0 & 0 & 0 \\ & 0 & 1 & 0 & 0 \\ & 1 & 0 & 0 & 0 \\ & 0 & 1 & 0 & 0 \end{array}$$

Implementação Local branching

- Busca em uma vizinhança $\mathcal{N}^k(\tilde{x})$ pode ser definida adicionando o pseudo-corte $\delta(x, \tilde{x}) \leq k$, ou seja,

$$\sum_{j \in I} [(1 - \tilde{x}_j) x_j + \tilde{x}_j (1 - x_j)] \leq k$$

- O novo conjunto de solução $Y = Y \setminus \mathcal{N}^k(\tilde{x})$ pode ser representado eliminando $\delta(x, \tilde{x}) \leq k$ e adicionando o pseudo-corte $\delta(x, \tilde{x}) \geq k + 1$, ou seja,

$$\sum_{j \in I} [(1 - \tilde{x}_j) x_j + \tilde{x}_j (1 - x_j)] \geq k + 1$$

Implementação Local branching

- Sugestão de implementação no concert:

```
IloExpr branch(env);
for(int j = 0; j < n; j++){
  branch += ( 1 - x[j])*vx[j]  + x[j]*(1 - vx[j]);
}
IloExtractable left_branching = mod.add(branch <= k);
```

depois

```
mod.remove(left_branching);
mod.add(branch >= k + 1);
branch.end();
```

Sumário

Heurísticas matemáticas: revisão RENS e RINS

Problema da mochila

$$\max \quad 2x_1 + 2x_2 + 3x_3 + 4x_4 + 6x_5 + 5x_6 + 8x_7 + 7x_8 \tag{1}$$

$$s.a \; 4x_1 + 15x_2 + 7x_3 + 9x_4 + 8x_5 + 10x_6 + 9x_7 + 11x_8 \leq 31 \tag{2}$$

$$x_j \in \{0, 1\} \tag{3}$$

Com base na solução da relaxação linear $x = (0.75, 0, 0, 0, 1, 0, 1, 1)$

- Defina o subproblema reduzido RENS.
- Defina o subproblema reduzido RINS utilizando a solução incumbente $\tilde{x} = (1, 1, 1, 0, 0, 0, 0, 0)$

Heurísticas matemáticas: revisão Relax-and-Fix

- Partição= $\{\{1,2,3\},\{4,5,6\},\{7,8\}\}$
- Problema da mochila

$$\begin{aligned} \max\ & 2x_1 + 2x_2 + 3x_3 + 4x_4 + 6x_5 + 5x_6 + 8x_7 + 7x_8 \\ s.a\ & 4x_1 + 15x_2 + 7x_3 + 9x_4 + 8x_5 + 10x_6 + 9x_7 + 11x_8 \leq 31 \\ & x_j \in \{0,1\} \end{aligned}$$

Heurísticas matemáticas: revisão Relax-and-Fix

- Partição= $\{\{1,2,3\},\{4,5,6\},\{7,8\}\}$
- Problema da mochila

$$
\begin{aligned}
\max \quad & 2x_1 + 2x_2 + 3x_3 + 4x_4 + 6x_5 + 5x_6 + 8x_7 + 7x_8 \\
s.a \quad & 4x_1 + 15x_2 + 7x_3 + 9x_4 + 8x_5 + 10x_6 + 9x_7 + 11x_8 \leq 31 \\
& x_j \in \{0,1\}
\end{aligned}
$$

- Iteração 1:
  - Problema SP_1

$$
\begin{aligned}
\max \quad & 2x_1 + 2x_2 + 3x_3 + 4x_4 + 6x_5 + 5x_6 + 8x_7 + 7x_8 && (1) \\
s.a \quad & 4x_1 + 15x_2 + 7x_3 + 9x_4 + 8x_5 + 10x_6 + 9x_7 + 11x_8 \leq 31 && (2) \\
& x_1 \in \{0,1\}, x_2 \in \{0,1\}, x_3 \in \{0,1\} && (3) \\
& x_4 \in [0,1], x_5 \in [0,1], x_6 \in [0,1], x_7 \in [0,1], x_8 \in [0,1] && (4)
\end{aligned}
$$

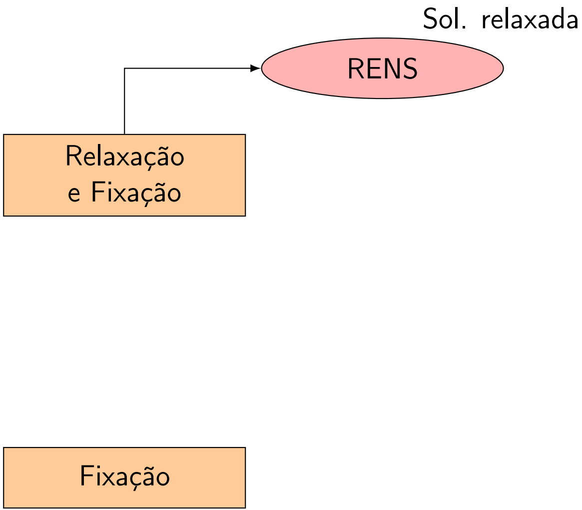
  - Solução $\bar{x} = (0,0,0,0,1,0.3,1,1)$

Heurísticas matemáticas

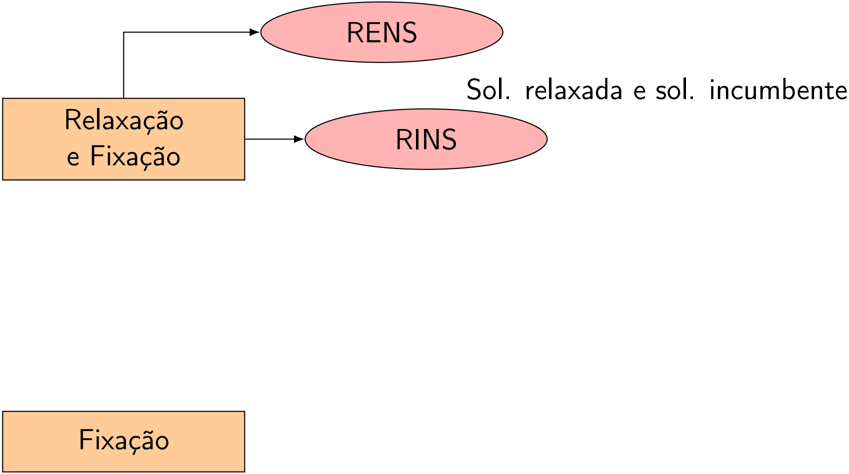
Relaxação
e Fixação

Fixação

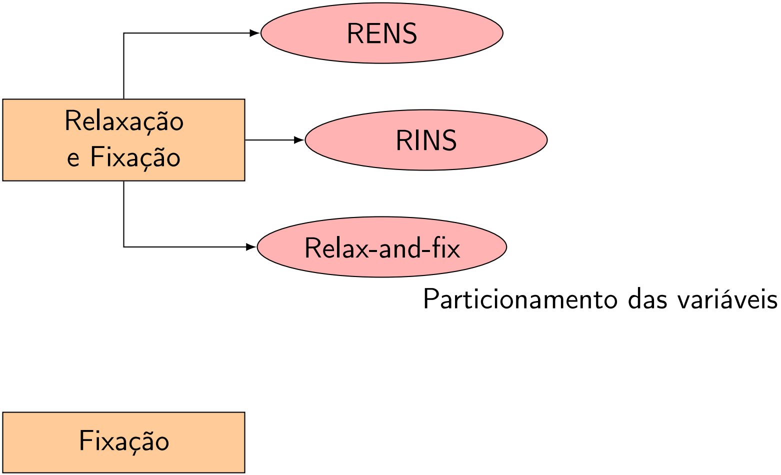
Heurísticas matemáticas



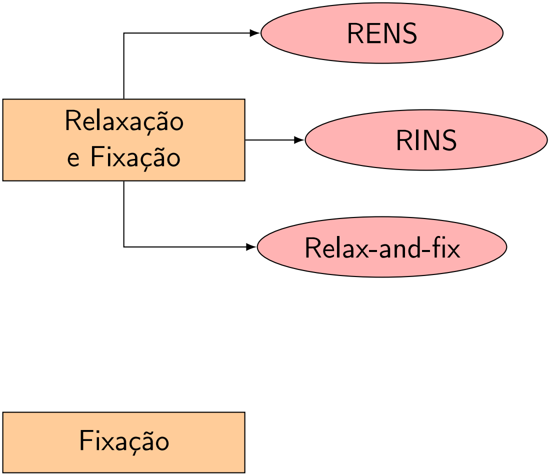
Heurísticas matemáticas



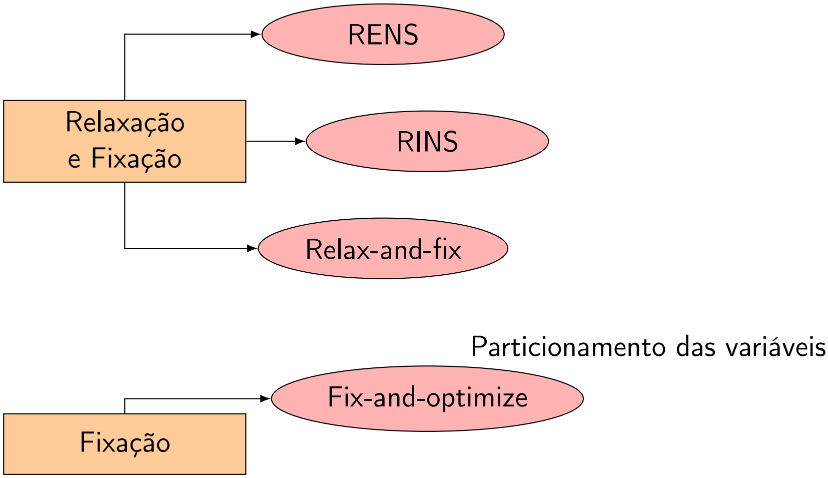
Heurísticas matemáticas



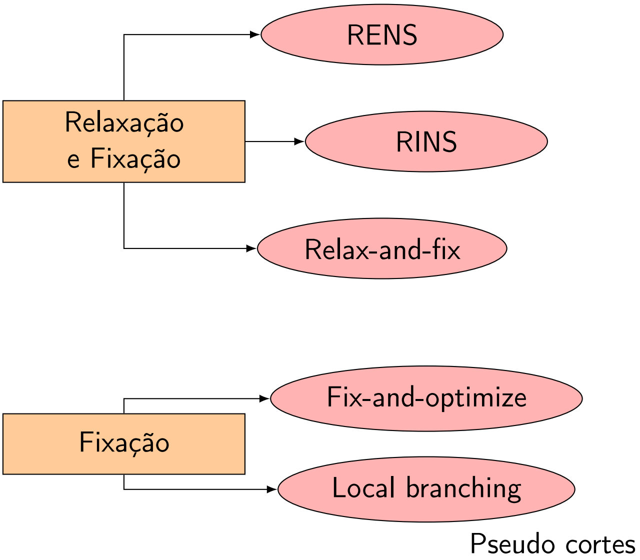
Heurísticas matemáticas



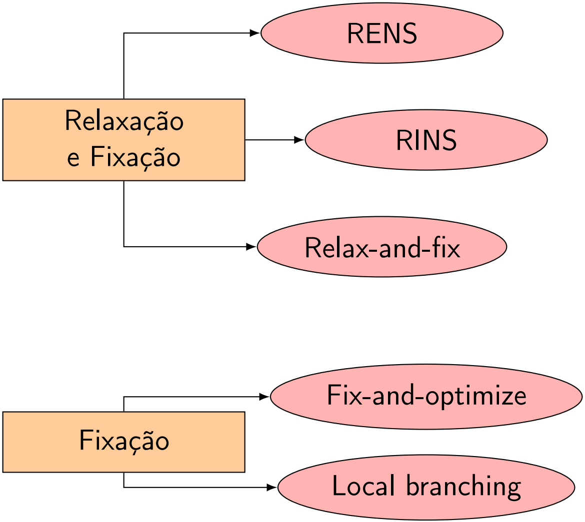
Heurísticas matemáticas



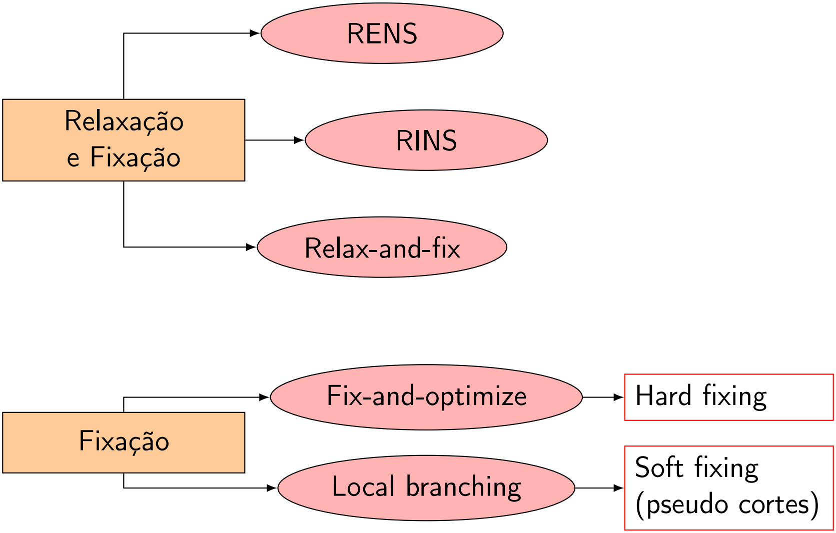
Heurísticas matemáticas



Heurísticas matemáticas



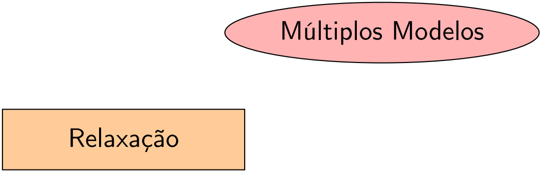
Heurísticas matemáticas



Heurísticas matemática: Relaxação

Relaxação

Heurísticas matemática: Relaxação



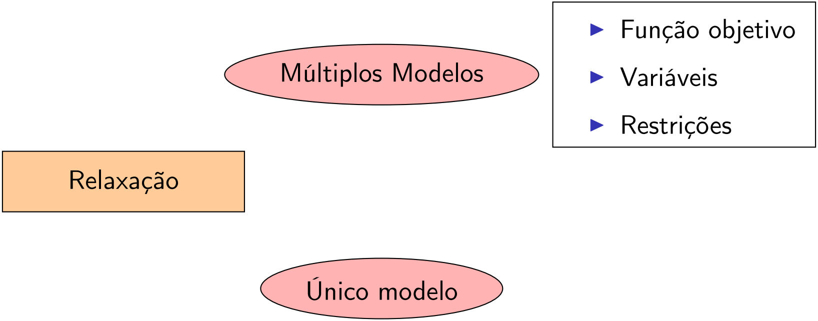
Heurísticas matemática: Relaxação

Múltiplos Modelos

- Função objetivo
- Variáveis
- Restrições

Relaxação

Heurísticas matemática: Relaxação



Heurísticas matemática: Relaxação

Relaxação

Múltiplos Modelos

- Função objetivo
- Variáveis
- Restrições

Único modelo

Conversão de tipo

- IloConversion

Heurísticas matemática: Relaxação

Relaxação

Múltiplos Modelos

- Função objetivo
- Variáveis
- Restrições

Único modelo

Conversão de tipo

- IloConversion

IloConversion - Concert

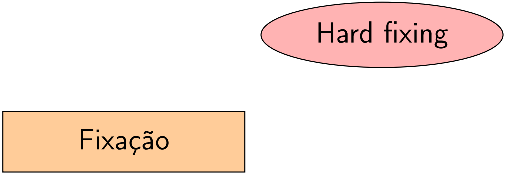
- IloConversion(IloEnv, IloNumVar, IloNumVar::Type)

```
IloNumVar x(env, 0, 1, ILOBOOL);
IloConversion convx(env, x, ILOFLOAT);
mod.add(convx); // Relaxa
mod.remove(convx); // Retira a relaxação
```

Heurísticas matemática: Fixação

Fixação

Heurísticas matemática: Fixação



Heurísticas matemática: Fixação

Fixação

Hard fixing

- Adição de uma restrição $x_j = \bar{x}_j$
- Mudança de bounds $\bar{x}_j \leq x_j \leq \bar{x}_j$

Heurísticas matemática: Fixação

Fixação

Hard fixing

- Adição de uma restrição $x_j = \bar{x}_j$
- Mudança de bounds $\bar{x}_j \leq x_j \leq \bar{x}_j$

Mudança de limitante inferior (LB) e limitante superior (UB)

- Concert: setBounds(IloNum lb, IloNum ub)

```
IloNumVar x(env, 0, 1, ILOBOOL);
x.setBounds(1,1);
```

Heurísticas matemática: Fixação

Hard fixing

Fixação

Soft fixing

- Adição de uma restrição $x_j = \bar{x}_j$
- Mudança de bounds $\bar{x}_j \leq x_j \leq \bar{x}_j$

Heurísticas matemática: Fixação

Fixação

Hard fixing

Soft fixing

- Adição de uma restrição $x_j = \bar{x}_j$
- Mudança de bounds $\bar{x}_j \leq x_j \leq \bar{x}_j$

Heurísticas matemática: Fixação

Fixação

Hard fixing

- Adição de uma restrição $x_j = \bar{x}_j$
- Mudança de bounds $\bar{x}_j \leq x_j \leq \bar{x}_j$

Soft fixing

Pseudo cortes

- $\delta(x, \tilde{x}) \leq k$

Distância de hamming

$$\delta(x, \tilde{x}) = \sum_j |x_j - \tilde{x}_j| = \sum_j [x_j(1 - \tilde{x}_j) + \tilde{x}_j(1 - x_j)]$$

Heurísticas matemática: Fixação

Fixação

Hard fixing

- Adição de uma restrição $x_j = \bar{x}_j$
- Mudança de bounds $\bar{x}_j \leq x_j \leq \bar{x}_j$

Soft fixing

Pseudo cortes

- $\delta(x, \tilde{x}) \leq k$

Distância de hamming

$$\delta(x, \tilde{x}) = \sum_j |x_j - \tilde{x}_j| = \sum_j [x_j(1 - \tilde{x}_j) + \tilde{x}_j(1 - x_j)]$$

Pseudocortes

$$\delta(x,\tilde{x}) = \sum_j |x_j - \tilde{x}_j| = \sum_j [x_j(1-\tilde{x}_j) + \tilde{x}_j(1-x_j)]$$

Código

```
IloExpr branch(env);
for(int j = 0; j < n; j++){
  branch += y[j] * ( 1 - ytil[j]) + ytil[j] * (1 - y[j]);
}
IloExtractable left_branching = mod.add( branch <= k);
.
.
mod.remove(left_branching);
mod.add( branch >= k + 1);
```

Referências

Matteo Fischetti and Andrea Lodi. Local branching. *Mathematical programming*, 98:23–47, 2003.

Chong Man Ngoo, Say Leng Goh, Nasser R Sabar, Mohd Hanafi Ahmad Hijazi, Graham Kendall, et al. A survey of mat-heuristics for combinatorial optimisation problems: variants, trends and opportunities. *Applied Soft Computing*, page 111947, 2024.