Escola de Engenharia Mauá

ECM511 –Pesquisa Operacional e ~Métodos de Otimização
Prof. Joyce M Zampirolli
joyce.zampirolli@maua.br

Problemas de Programação Inteira (PPI) Método Branch and Bound

Março/2016

Ex. Rádio Totó

Na rádio Totó Ternura, o programa "Movimento Operário" é dedicado aos amantes da música de ópera. Para a próxima hora foram programados reclames de 15, 16, 20, 25, 30, 35, 40 e 50 segundos, a serem veiculados em blocos de, no máximo, 1 minuto. Formule um problema de programação inteira que possa ser utilizado para minimizar o número de blocos comerciais veiculados na próxima hora. *Dica*: serão necessários, no máximo, 8 blocos comerciais (por quê?).

Ex. Rádio Totó

1. Variáveis de decisão

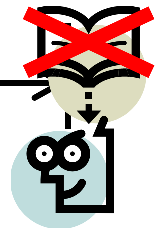
$$c_{ij} = \begin{cases} 1, \text{ se o comercial } i \ (i = 1,\ldots,8) \text{ vai ao ar no bloco } j \ (j = 1,\ldots,8) \\ 0, \text{ em caso contrário} \end{cases}$$

Bloco 1: uma possível solução

Comercial 1 15 segundos	Comercial 2 16 segundos	Comercial 3 20 segundos	9 seg.

$$b_j = \begin{cases} 1, \text{ se o bloco } j \ (j = 1,\ldots,8) \text{ for veiculado} \\ 0 \text{ em caso contrário} \end{cases}$$

60 segundos



2. Função objetivo

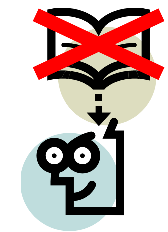
Minimizar o número de blocos veiculados

$$\min \quad z = \sum_{j=1}^{8} b_j$$

Por favor, anote!

Neste exemplo: $c_{11} = 1;\ c_{21} = 1;\ c_{31} = 1;\ b_1 = 1$

Ex. Rádio Totó



3. Restrições

3.1. Veiculação obrigatória dos comerciais

$$\text{Comercial 1:}\quad c_{11} + c_{12} + c_{13} + \boxed{\cdots} + c_{17} + c_{18} = 1$$

$$\text{Comercial 2:}\quad c_{21} + c_{22} + c_{23} + \boxed{\cdots} + c_{27} + c_{28} = 1$$

$$\boxed{\vdots}$$

NÃO FAÇA ISSO!

Aqui pode!

$$\sum_{j=1}^{8} c_{ij} = 1, \ \underline{\text{para } i = 1,2,\ldots,8}$$

São 8 equações, uma para cada comercial i...

...em que são somados 8 termos, relativos a cada bloco j

Ex. Rádio Totó

3. Restrições

3.1. Veiculação obrigatória dos comerciais

$$\sum_{j=1}^{8} c_{ij}$$

Importante:

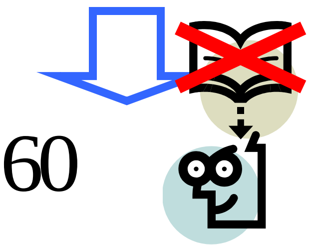
$$\sum_{i=1}^{n} v_i \leq 3 \qquad \neq \qquad v_i \leq 3 \text{ , para } i = 1,2,\ldots,n$$

Uma soma de 8 termos.

Por favor, anote!

Ex. Rádio Totó

3.2. Duração máxima de cada bloco



$$\text{Bloco 1: } 15\,c_{11} + 16\,c_{21} + 20\,c_{31} + \boxed{\cdots} + 50\,c_{81} \leq 60$$

$$\text{Bloco 2: } 15\,c_{12} + 16\,c_{22} + 20\,c_{32} + \boxed{\cdots} + 50\,c_{82} \leq 60\,b_2$$

$$\boxed{\vdots}$$

NÃO!

$$15\,c_{1j} + 16\,c_{2j} + \boxed{\cdots} + 50\,c_{8j} \leq 60\,b_j \text{ , para } j = 1,2,\ldots,8$$

NÃO!!

São 8 inequações, uma para cada bloco j

Ex. Rádio Totó

3.2. Duração máxima de cada bloco

Anote, por favor!

NÃO faça isso!

$$15\,c_{1\,j} + 16\,c_{2\,j} + \boxed{\cdots} + 50\,c$$

São valores constantes e conhecidos.

Logo, NÃO SÃO VARIÁVEIS DE DECISÃO!

Opção: $$\sum_{i=1}^{8} D_i c_{ij} \leq 60\,b_j$$

Sendo: D_i = duração (em segundos) do comercial i ($i = 1,2,\ldots,8$)

São PARÂMETROS do modelo.

3.3. Variáveis binárias

$$c_{ij}\,,\,b_j = \{0,1\}\text{, para } i = 1,2,\ldots,8 \text{ e para } j = 1,2,\ldots,8$$

Ex. Set-covering

1) (*Set-covering*) O condado de Miserê é formado por seis cidades que desejam determinar onde devem ser construídas bases de combate a incêndios. A norma de segurança local estabelece que cada cidade deve estar "coberta" por ao menos uma base de atendimento, ou seja, deve ser alcançada pela base mais próxima em até 15 minutos. Os tempos de deslocamento entre as cidades estão representados na Tabela 3.4

Tabela 3.4 - Tempos de deslocamento (em minutos) entre as cidades

De	**Para**					
	Cidade 1	**Cidade 2**	**Cidade 3**	**Cidade 4**	**Cidade 5**	**Cidade 6**
Cidade 1	0	10	20	30	30	20
Cidade 2	10	0	25	35	20	10
Cidade 3	20	25	0	15	30	20
Cidade 4	30	35	15	0	15	25
Cidade 5	30	20	30	15	0	14
Cidade 6	20	10	20	25	14	0

Modele um PPI que minimize o número de bases instaladas no condado.

Ex. Set-covering

1. Variáveis de decisão

$$b_i = \begin{cases} 1, \text{ se uma base for construída na cidade } i \ (i = 1,\ldots,6) \\ 0 \ \text{ em caso contrário} \end{cases}$$

2. Função objetivo

Minimizar o número de bases instaladas no condado

$$\min \ z = \sum_{i=1}^{6} b_i$$

Ex. Set-covering

3. Restrições

3.1. Cobertura mínima

Cidade 1: $b_1 + b_2 \geq 1$

Cidade 2: $b_1 + b_2 + b_6 \geq 1$

Cidade 3: $b_3 + b_4 \geq 1$

Cidade 4: $b_3 + b_4 + b_5 \geq 1$

Cidade 5: $b_4 + b_5 + b_6 \geq 1$

Cidade 6: $b_2 + b_5 + b_6 \geq 1$

3.2. Variáveis binárias

$b_i = \{0,1\}$, para $i = 1,\ldots,6$

Da cidade	Para a cidade					
	1	2	3	4	5	6
1	$\boxed{0}$	$\boxed{10}$	20	30	30	20
2	$\boxed{10}$	$\boxed{0}$	25	35	20	$\boxed{10}$
3	20	25	$\boxed{0}$	$\boxed{15}$	30	20
4	30	35	$\boxed{15}$	$\boxed{0}$	$\boxed{15}$	25
5	30	20	30	$\boxed{15}$	$\boxed{0}$	$\boxed{14}$
6	20	$\boxed{10}$	20	25	$\boxed{14}$	$\boxed{0}$

Tem que colocar restrições de não negatividade e de variáveis inteiras?

Ex. Máquinas e Tarefas (alocação)

1) Em uma planta industrial existem cinco máquinas que devem executar cinco tarefas. O tempo requerido para a execução de cada tarefa depende da máquina utilizada, como indicado na Tabela 3.6.

Tabela 3.6 - Tempos de operação e de *set-up*

	Tarefa					**Tempo de *set-up***
	1	**2**	**3**	**4**	**5**	
Máquina 1	42	70	93	×	×	30
Máquina 2	×	85	45	×	×	40
Máquina 3	58	×	×	37	×	50
Máquina 4	58	×	55	×	38	60
Máquina 5	×	60	×	54	×	20

Se uma máquina for utilizada, existe um tempo de *set-up* associado, indicado na Tabela 3.6. Formule um modelo de programação inteira que minimize o tempo total (operação e *set-up*) gasto na execução das cinco tarefas.

Ex. mquinas e tarefas

1. Variáveis de decisão

$$x_{ij} = \begin{cases} 1, \text{ se a máq. } i \ (i = 1,\ldots,5) \text{ executa a tarefa } j \ (j = 1,\ldots,5) \\ 0 \quad \text{em caso contrário} \end{cases}$$

$$y_i = \begin{cases} 1, \text{ se a máquina } i \ (i = 1,\ldots,5) \text{ for utilizada} \\ 0 \quad \text{em caso contrário} \end{cases}$$

2. Função objetivo

Minimizar o tempo de execução (operação + *set-up*)

$$\min z = 42x_{11} + 70x_{12} + \ldots + 30y_1 + \ldots + 20y_5$$

NÃO FAÇA ISSO!

Continua...

Notação simplificada

Ao invés de "...", use a seguinte notação:

$$\min \quad z = \sum_{i=1}^{5}\sum_{j=1}^{5} \boldsymbol{T_{ij}} x_{ij} + \sum_{i=1}^{5} \boldsymbol{TS_i} y_i$$

Parâmetros do modelo
($\neq$ variáveis)

sendo: T_{ij} = tempo que a máquina i (i = 1,...,5) gasta para executar a tarefa j (j = 1,...,5)

TS_i = tempo de *set-up* da máquina i (i = 1,...,5)

Ainda não acabou: quanto vale T_{14}?

Notação simplificada

2. Função objetivo

Minimizar o tempo de execução (operação + *set-up*)

$$\min \quad z = \sum_{i=1}^{5} \sum_{j=1}^{5} T_{ij} x_{ij} + \sum_{i=1}^{5} TS_i \, y_i$$

sendo: T_{ij} = tempo que a máquina i ($i = 1,\ldots,5$) gasta para executar a tarefa j ($j = 1,\ldots,5$)

$T_{ij} = \boldsymbol{M}$ (nº computacionalmente alto) para os os pares máquina/tarefa indefinidos.

TS_i = tempo de *set-up* da máquina i ($i = 1,\ldots,5$)

Agora sim!

Ex. máquinas e tarefas

3. Restrições

$T_{21} = T_{51} = M$

3.1. Toda tarefa deve ser executada

$$\text{Tarefa 1: } x_{11} + x_{21} + x_{31} + x_{41} + x_{51} = 1$$

$$\text{Tarefa 2: } x_{12} + x_{22} + x_{32} + x_{42} + x_{52} = 1$$

$$\vdots$$

5 equações, uma para cada **tarefa *j***

Notação simplificada:

$$\boxed{\sum_{i=1}^{5} x_{ij} = 1, \text{ para } j = 1,\ldots,5}$$

São 5 equações, uma para cada tarefa *j*

Ex. máquinas e tarefas

3. Restrições

$T_{14} = T_{15} = \text{M}$

3.2. Utilização das máquinas

$$\text{Máquina 1: } x_{11} + x_{12} + x_{13} + x_{14} + x_{15} \leq \mathbf{5} y_1$$

$$\text{Máquina 2: } x_{21} + x_{22} + x_{23} + x_{24} + x_{25} \leq \mathbf{5} y_2$$

$$\vdots$$

5 inequações, uma para cada **máquina *i***

Notação simplificada:

$$\boxed{\sum_{j=1}^{5} x_{ij} \leq 5 y_i \text{ , para } i = 1,\ldots,5}$$

São 5 inequações, uma para cada máquina *i*

3.3. Variáveis binárias...

$$x_{ij}, y_i = \{0,1\}, \text{ para } i = 1,2,\ldots,5 \text{ e para } j = 1,2,\ldots,5$$

Ex. Milk Run

1) (*Milk Run*) Quatro caminhões estão disponíveis para a distribuição de leite em cinco fábricas de alimentos. A capacidade e o custo diário operacional de cada caminhão são dados na Tabela 3.5. A demanda em cada fábrica deve ser atendida por apenas um caminhão, mas um caminhão pode realizar entregas para mais de uma fábrica. A demanda diária de cada fábrica é dada por: fábrica 1, 100 galões; fábrica 2, 200 galões; fábrica 3, 300 galões; fábrica 4, 500 galões; fábrica 5, 800 galões.

Tabela 3.5 - Capacidade e custo de operação dos caminhões

	Capacidade (galões)	Custo operacional diário
Caminhão 1	400	$ 45,00
Caminhão 2	500	$ 50,00
Caminhão 3	600	$ 55,00
Caminhão 4	1.100	$ 60,00

Formule um PPI que possa ser utilizado para minimizar o custo diário de atendimento das demandas das cinco fábricas.

Ex. Milk Run

1. Variáveis de decisão

$$c_{ij} = \begin{cases} 1, \text{ se o cam. } i \ (i = 1,\ldots,4) \text{ atende a fábrica } j \ (j = 1,\ldots,5) \\ 0 \ \text{ em caso contrário} \end{cases}$$

$$t_i = \begin{cases} 1, \text{ se o caminhão } i \ (i = 1,\ldots,4) \text{ for utilizado} \\ 0 \ \text{ em caso contrário} \end{cases}$$

2. Função objetivo

Minimizar o custo total diário de operação

$$\min \ z = 45\, t_1 + 50\, t_2 + 55\, t_3 + 60\, t_4$$

Ex. Milk Run

3. Restrições

3.1. Atendimento das fábricas

$$\text{Fábrica 1: } c_{11} + c_{21} + c_{31} + c_{41} = 1$$

$$\text{Fábrica 2: } c_{12} + c_{22} + c_{32} + c_{42} = 1$$

$$\vdots$$

5 equações, uma para cada fábrica *j*

$$\boxed{\sum_{i=1}^{4} c_{ij} = 1 \ , \ \text{para } j = 1,\ldots,5}$$

Ex.: Milk Run

3.3. Variáveis binárias...

$$c_{ij}, t_i = \{0,1\}, \text{ para } i = 1,\ldots,4 \text{ e para } j = 1,\ldots,5$$

1: $100c_{11} + 200c_{12} + 300c_{13} + 500c_{14} + 800c_{15} \leq 400t_1$

2: $100c_{21} + 200c_{22} + 300c_{23} + 500c_{24} + 800c_{25} \leq 500t_2$

3: $100c_{31} + 200c_{32} + 300c_{33} + 500c_{34} + 800c_{35} \leq 600t_3$

4: $100c_{41} + 200c_{42} + 300c_{43} + 500c_{44} + 800c_{45} \leq 1.100t_4$

$$\boxed{\sum_{j=1}^{5} DF_j \, c_{ij} \leq CAP_i \, t_i \, , \text{ para } i = 1,\ldots,4}$$

Parâmetros do modelo (≠ variáveis)

Opção:

sendo: DF_j = demanda da fábrica j ($j = 1,\ldots,5$)

CAP_i = capacidade do caminhão i ($i = 1,\ldots,4$)

Algoritmo Branch-and-Bound

Exemplo:

$$\max z = 8x_1 + 5x_2$$

$$\text{suj. a:} \quad x_1 + x_2 \leq 6$$

$$9x_1 + 5x_2 \leq 45$$

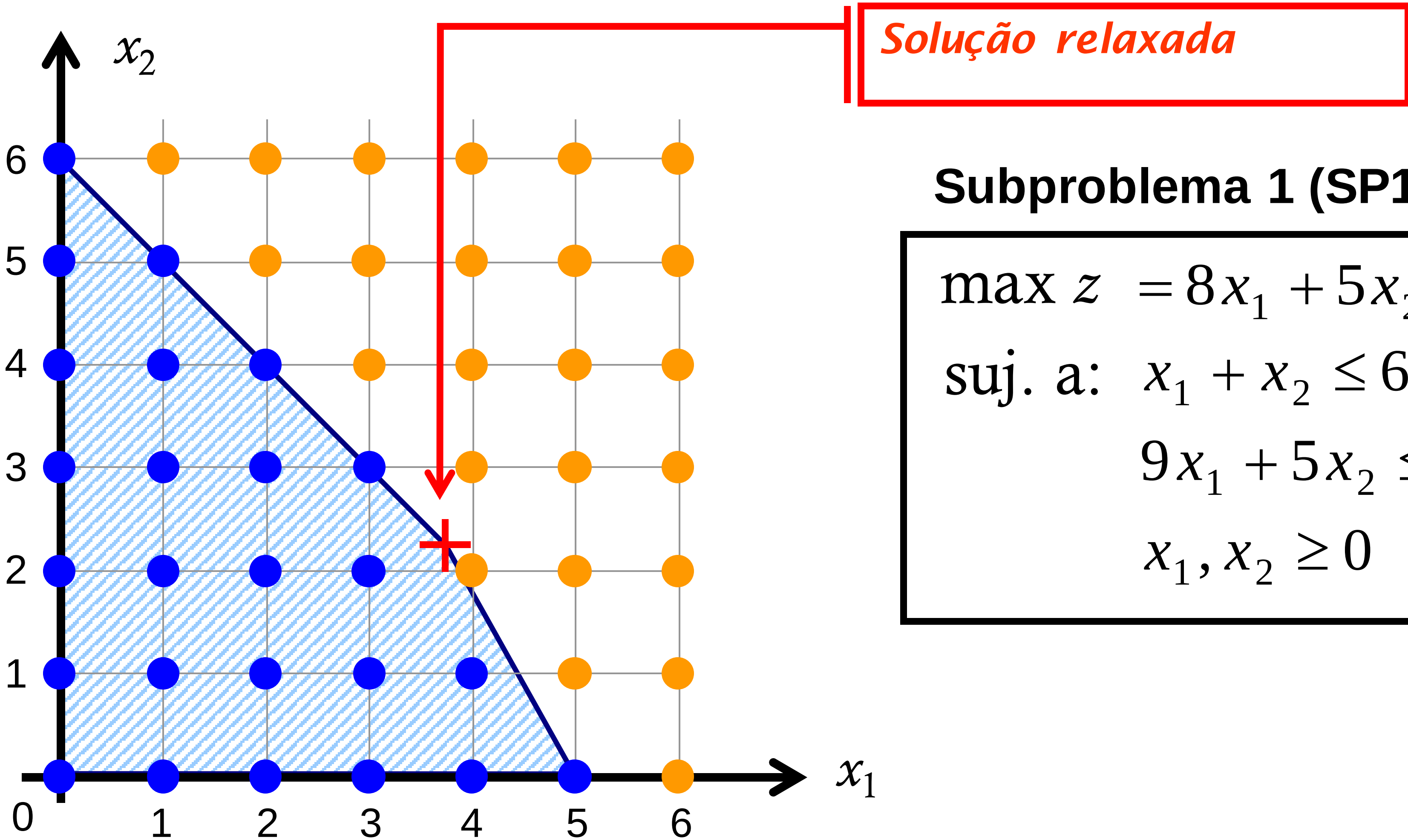
$$x_1, x_2 \geq 0$$

$$\boxed{x_1, x_2 : \text{inteiros}}$$

Inicialmente, resolve-se o problema ***sem*** *a restrição que exige* ***solução inteira****, obtendo-se uma* ***solução relaxada****.*

Este **modelo relaxado** é um dos (vários) que terão que ser resolvidos e será chamado de **Subproblema 1**

Algoritmo Branch-and-Bound



Subproblema 1 (SP1)

$$\begin{aligned} \max z &= 8x_1 + 5x_2 \\ \text{suj. a:} \quad & x_1 + x_2 \leq 6 \\ & 9x_1 + 5x_2 \leq 45 \\ & x_1, x_2 \geq 0 \end{aligned}$$

Soluções inteiras inviáveis

Soluções inteiras viáveis

Algoritmo Branch-and-Bound

Resolvendo os subproblemas no LPSolve IDE:

1. … P), termi… om ";"
2. … esolver c…
3. … r os resu…

```
max:  8x1+5x2;
x1+x2<=6;
9x1+5x2<=45;
```

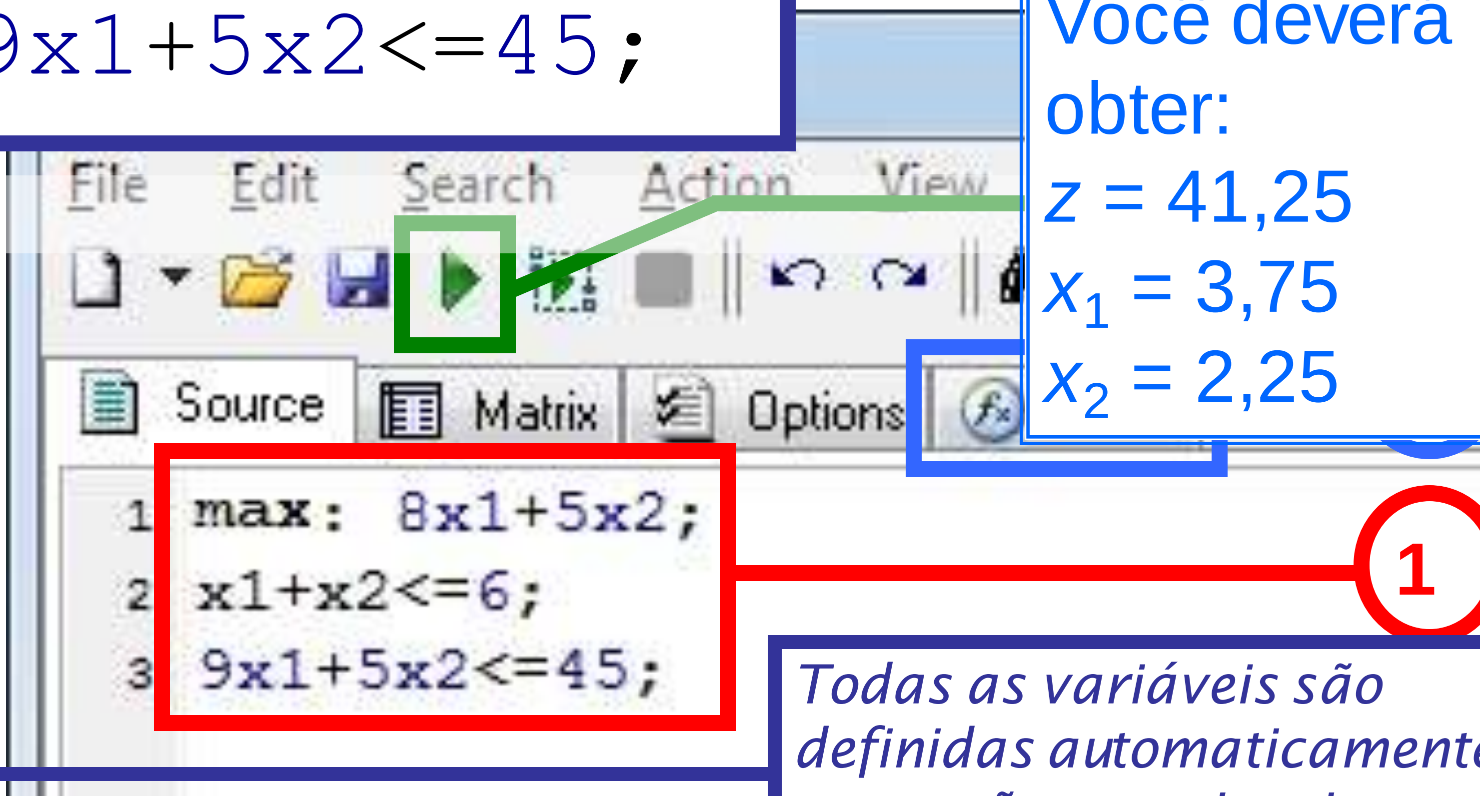
OK, agora RESOLVA O SP1!

Você deverá obter:

z = 41,25

x_1 = 3,75

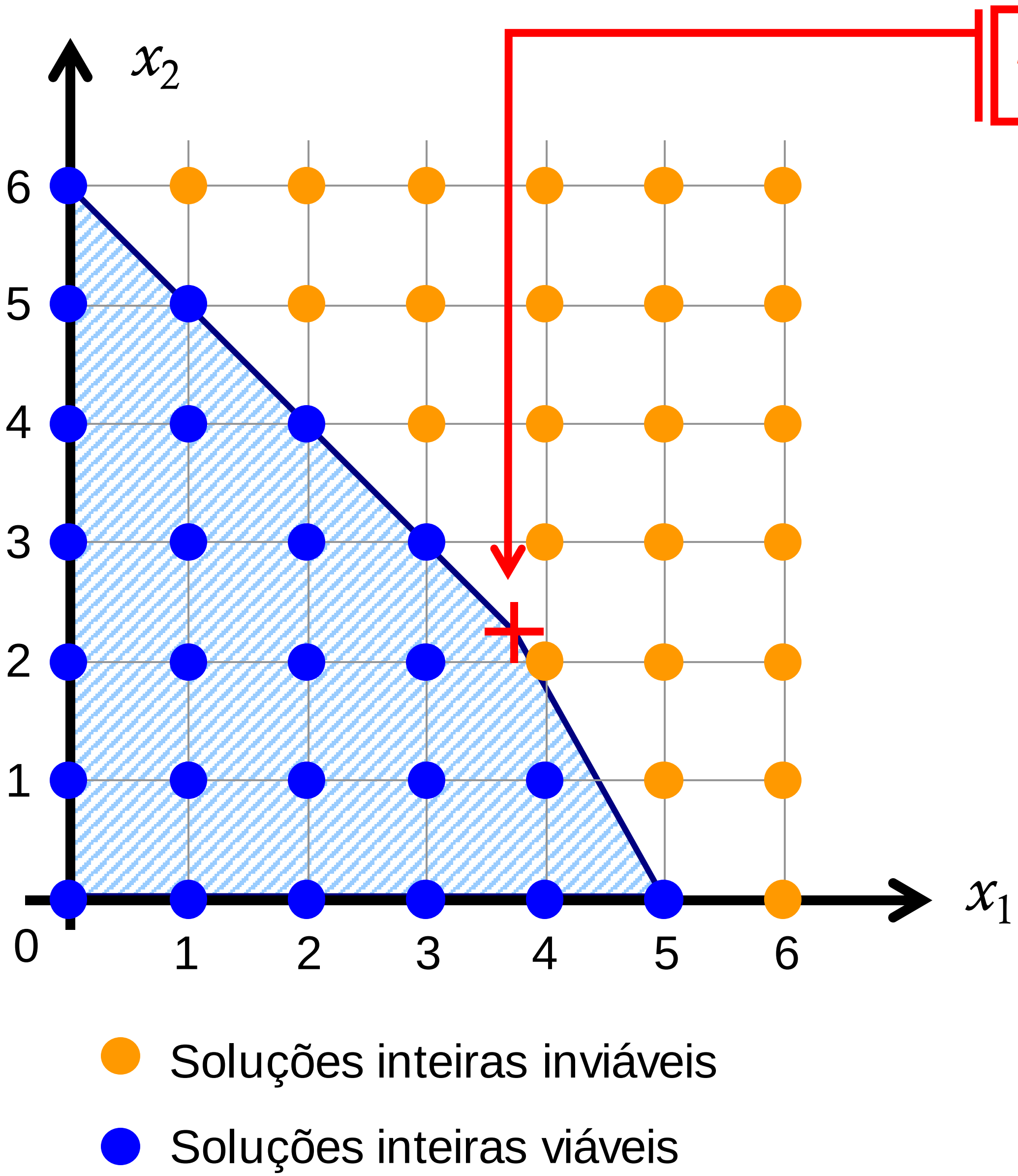
x_2 = 2,25



1

Todas as variáveis são definidas automaticamente como não negativas!

Algoritmo Branch-and-Bound

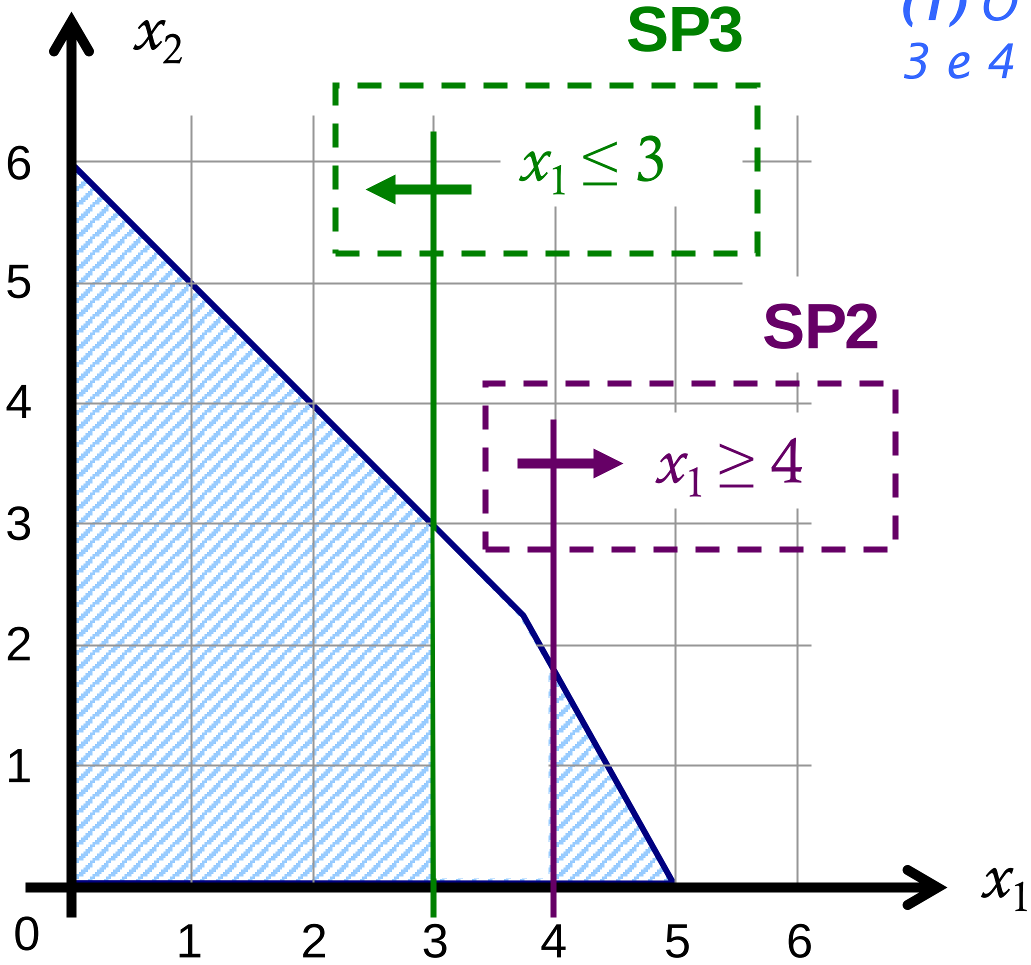


Subproblema 1 (SP1)

$$\begin{aligned} \max z \; &= 8x_1 + 5x_2 \\ \text{suj. a:} \quad & x_1 + x_2 \le 6 \\ & 9x_1 + 5x_2 \le 45 \\ & x_1, x_2 \ge 0 \end{aligned}$$

Como a solução do SP1 ***não é inteira*** *(ou seja, há variáveis de decisão com valores fracionários), acrescentam-se restrições a* ***uma*** *das variáveis (por exemplo, x_1).*

Algoritmo Branch-and-Bound



*(1) O valor de x_1 **não pode** estar entre 3 e 4 (daí, a criação do SP2 e do SP3).*

Subproblema 1 (SP1)

$$
\begin{aligned}
\max z \; &= 8x_1 + 5x_2 \\
\text{suj. a:} \; & x_1 + x_2 \leq 6 \\
& 9x_1 + 5x_2 \leq 45 \\
& x_1, x_2 \geq 0
\end{aligned}
$$

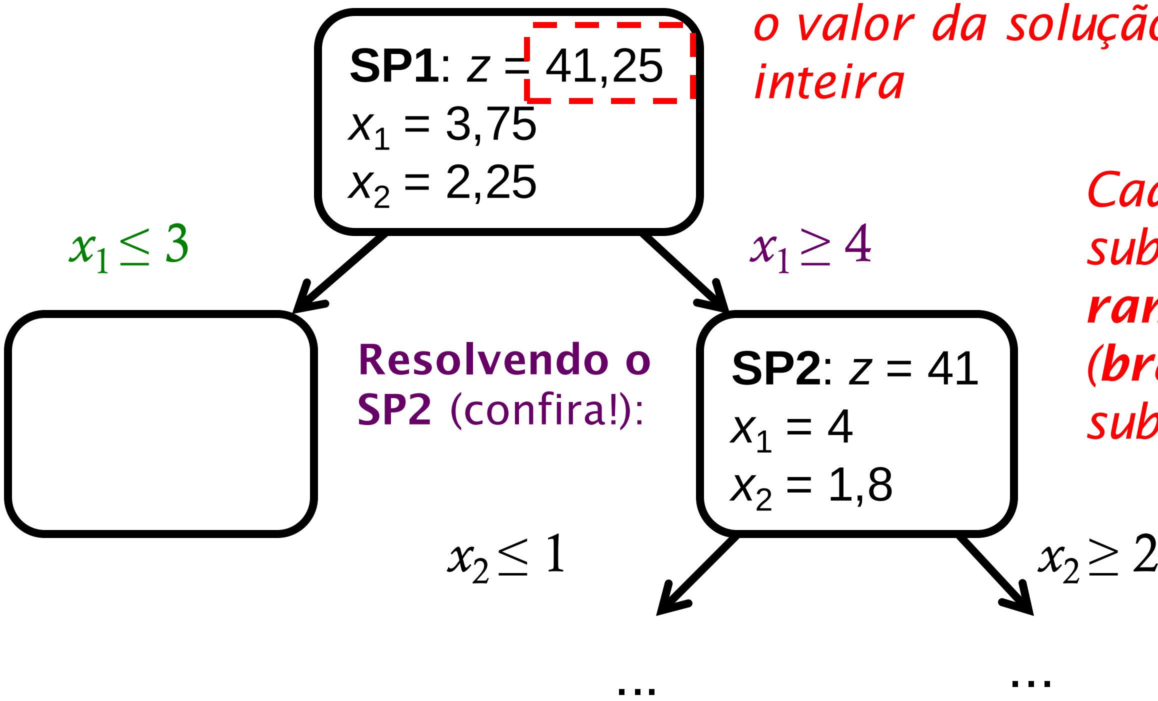
(2) Sim, eu poderia ter acrescentado restrições para a variável x_2.

*(3) O acréscimo de restrições tende a reduzir o 'espaço de busca' de soluções. Assim, as soluções do **SP2** e do **SP3 não podem ser melhores** do que a do **SP1**.*

Algoritmo Branch-and-Bound

Árvore de subproblemas:

*Este valor estabelece um **limite superior** (**bound**) para o valor da solução ótima inteira*



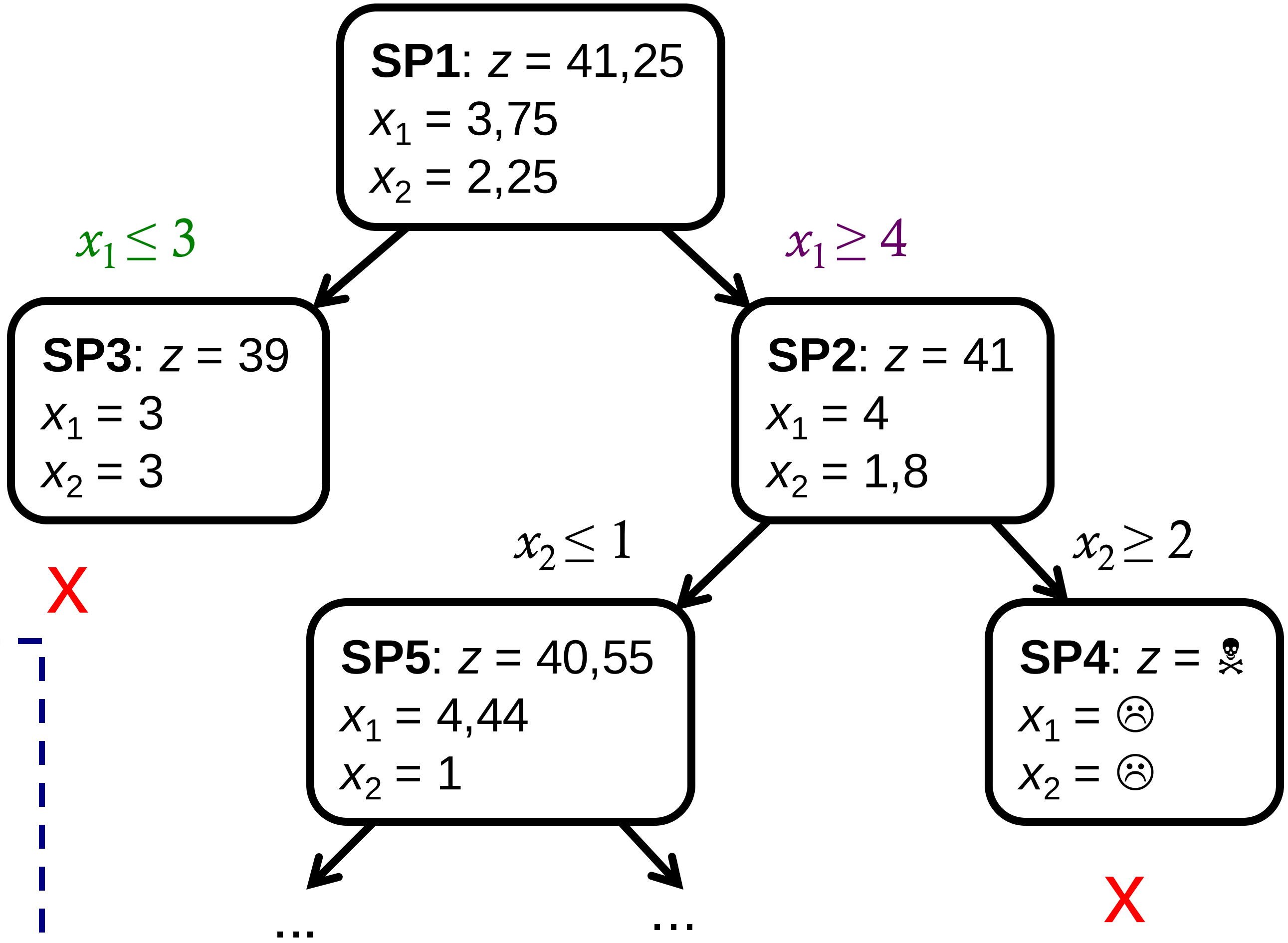
*Cada subproblema é **ramificado** (**branch**) em dois subproblemas*

***Importante**: solução inteira não significa que a função objetivo (z) deve ter, obrigatoriamente, um valor inteiro!*

Algoritmo Branch-and-Bound

Tabela dos limites da solução:

	LI	LS
1	**-∞**	**41,25**
2	**-∞**	**41,25**
3	**39**	**41**
4	**39**	**41**
5	**39**	**40,55**
6		
7		



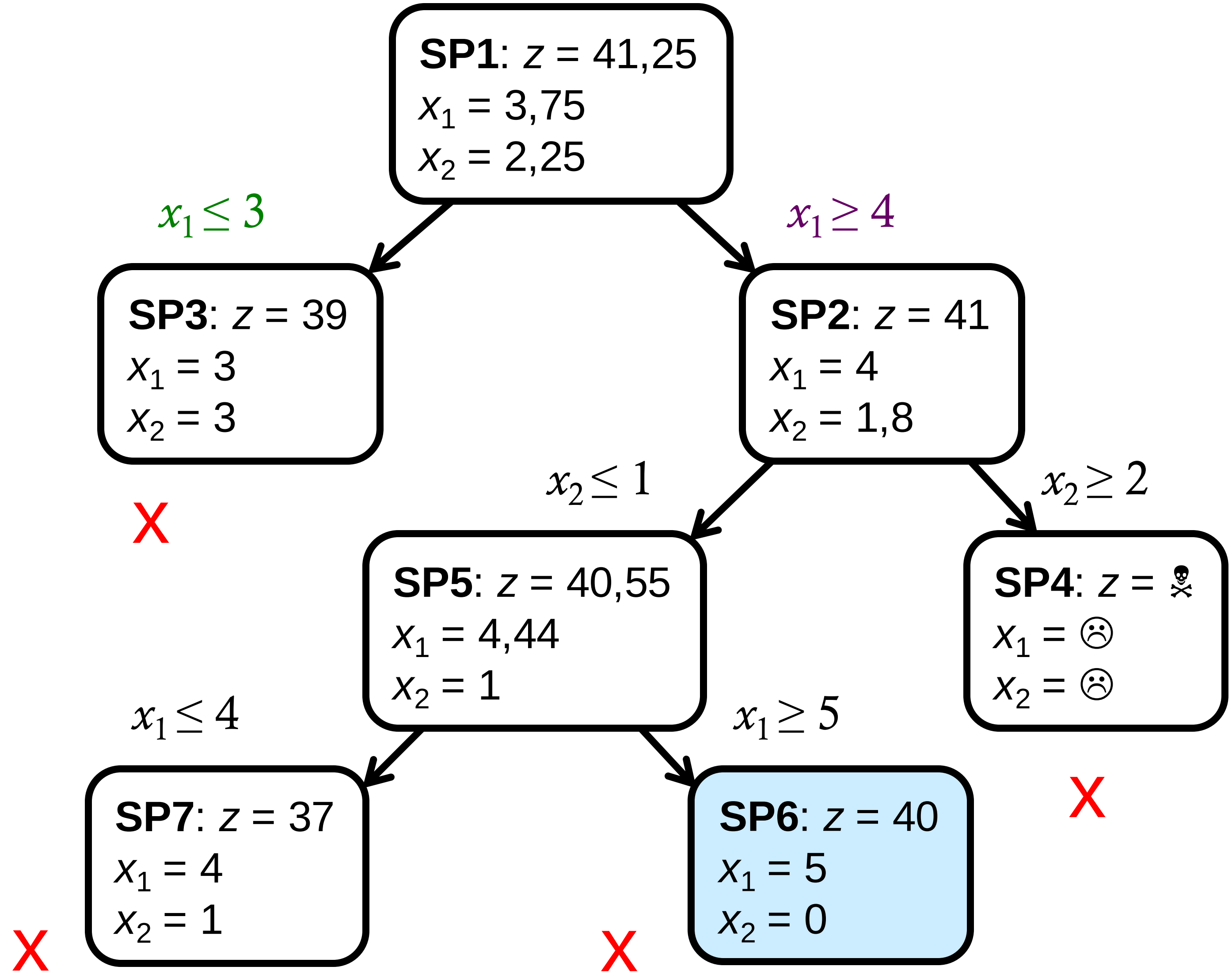
Neste ponto, já se sabe que z está entre 39 e 40,55. Se a diferença entre estes valores estiver dentro de uma **margem de tolerância** aceitável, então o algoritmo Branch-and-Bound pode parar!*

Algoritmo Branch-and-Bound

Tabela dos limites da solução:

	LI	LS
1	**-∞**	**41,25**
2	**-∞**	**41,25**
3	**39**	**41**
4	**39**	**41**
5	**39**	**40,55**
6	**40**	**40,55**
7	**40**	**40**

SOLUÇÃO ÓTIMA!



Branch-and-Bound: ex. p/ discussão

$$\max \ z = x_1 + 3x_2 + 4x_3$$

$$\text{suj. a:} \quad x_1 + 3x_2 + 2x_3 \leq 25$$

$$2x_1 + 4x_2 - 2x_3 \geq 30$$

$$3x_1 + x_2 + 5x_3 \leq 28$$

$$x_1, x_2, x_3 \geq 0$$

$$x_1, x_2, x_3 : \text{inteiros}$$

Regras para as ramificações:

- **Entre os subproblemas:** maior valor de z.

- **Entre as variáveis:** menor índice.

Branch-and-Bound: ex. p/ discussão

	L.I.	L.S.
1	**-∞**	**27,87**
2	**-∞**	**27,87**
3	**-∞**	**27,86**
4	**-∞**	**27,86**
5	**20**	**27,66**
6	**27**	**27,66**
7	**27**	**27,57**
8	**27**	**27,57**
9	**27**	**27**