Programação Inteira I - Introdução

Alexandre Checoli Choueiri

28/01/2024

Motivação

Os programas **lineares**

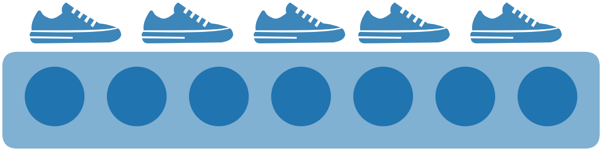
Até o momento trabalhamos com a chamada **programação linear - PL**. Uma das premissas da PL é a de que as variáveis do problema $x \in \mathbb{R}^{+}$, ou seja, podem assumir valores fracionários.

Em muitos casos isso não é um problema, por exemplo:

Motivação

1 - Variáveis com números inteiros

Considere uma linha de produção de sapatos:



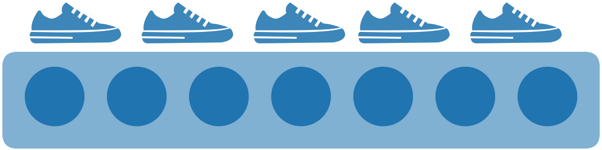
Supondo que a solução ótima de um PL tenha solução para a produção de 480.5 sapatos em um dia, a um custo de R$90. **O custo total desa solução é de:**

$$Z = 480.5 \cdot 90 = R\$43.245,00$$

Motivação

1 - Variáveis com números inteiros

Considere uma linha de produção de sapatos:



Como não podemos produzir 480.5 sapatos, podemos usar uma técnica de **arredondamento**, podemos produzir 481 sapatos, com isso temos o custo:

$$\boxed{Z = 481 \cdot 90 = R\$43.290,00}$$

Motivação

1 - Variáveis com números inteiros

Note que a diferença entre os custos é ínfima, somente

$$\boxed{\Delta Z = R\$43.290,00 - R\$43.245,00 = 45}$$

A diferença é ínfima, apenas 0.103%. Com essa diferença, podemos considerar **aceitável** a técnica de arredondamento.

Motivação

1 - Variáveis com números inteiros

Considere a produção de navios...

Os navios podem levar até 36 meses para serem construídos, com custos altíssimos.

> Suponha ainda que a solução de um PL para o planejamento de construção de navios para os próximos 5 anos foi de 7.8 un.

E agora? Arredondamos para cima (8) ou para baixo (7)?

Motivação

1 - Variáveis com números inteiros

Arredondar ou não arredondar?

Nesse caso, como os custos unitários e tempo de produção são muito elevados, qualquer erro, por menor que seja, causa um impacto muito grande no planejamento final. Esse é um exemplo em que não podemos usar o arredondamento! Precisamos saber qual a **solução ótima inteira**.

Motivação

2 - Variáveis binárias

Variáveis binárias

No entanto, a maior vantagem da programação inteira está no **uso de variáveis binárias** (variáveis que só podem assumir valores 0 ou 1). Essas variáveis possibilitam a criação de muitos novos modelos, pois representam decisão do tipo "sim" e "não". Veremos exemplos na parte de modelagem.

Motivação

2 - Variáveis binárias

Variáveis binárias

No entanto, a maior vantagem da programação inteira está no **uso de variáveis binárias** (variáveis que só podem assumir valores 0 ou 1). Essas variáveis possibilitam a criação de muitos novos modelos, pois representam decisão do tipo "sim"e "não". Veremos exemplos na parte de modelagem.

$$x = \begin{cases} 1: & \text{Se a máquina é utilizada} \\ 0: & \text{Caso contrário} \end{cases}$$

Variáveis binárias

No entanto, a maior vantagem da programação inteira está no **uso de variáveis binárias** (variáveis que só podem assumir valores 0 ou 1). Essas variáveis possibilitam a criação de muitos novos modelos, pois representam decisão do tipo "sim"e "não". Veremos exemplos na parte de modelagem.

$$x = \begin{cases} 1: & \text{Se a máquina é utilizada} \\ 0: & \text{Caso contrário} \end{cases}$$

$$x_i = \begin{cases} 1: & \text{Se o investimento } i \text{ é feito} \\ 0: & \text{Caso contrário} \end{cases}$$

Variáveis binárias

No entanto, a maior vantagem da programação inteira está no **uso de variáveis binárias** (variáveis que só podem assumir valores 0 ou 1). Essas variáveis possibilitam a criação de muitos novos modelos, pois representam decisão do tipo "sim" e "não". Veremos exemplos na parte de modelagem.

$$x = \begin{cases} 1: & \text{Se a máquina é utilizada} \\ 0: & \text{Caso contrário} \end{cases}$$

$$x_i = \begin{cases} 1: & \text{Se o investimento } i \text{ é feito} \\ 0: & \text{Caso contrário} \end{cases}$$

$$x_{ij} = \begin{cases} 1: & \text{Se o caminho entre o ponto } i \text{ e } j \text{ é percorrido} \\ 0: & \text{Caso contrário} \end{cases}$$

Formalização e regiões factíveis

Formalização e regiões factíveis

Agora formalizaremos as possibilidades de programas inteiros (PI) que existem (**podemos ter modelos com variáveis inteiras e reais na mesma formulação**). Além disso, precisamos entender como é a região factível dessas novas formulações, pois elas estão diretamente relacionadas ao método de resolução dos problemas (método *Branch and Bound*).

Formalização e regiões factíveis

RELEMBRANDO

Um modelo de **programação linear (PL)** admite valores reais não negativos para as suas variáveis, e assume a forma genérica:

$$
\begin{aligned}
\text{min z} = \mathbf{c}^T\mathbf{x} \\
\mathbf{A}\mathbf{x} = \mathbf{b} \\
\mathbf{x} \in \mathbb{R}^n_+
\end{aligned}
$$

1. **A** é uma matriz mxn chamada matriz dos coeficientes.
2. $\mathbf{c}^T$ é o vetor dos custos.
3. $\mathbf{b}^T$ é o vetor dos termos independentes ou de recursos.

Formalização e regiões factíveis

Exemplo: Considere o seguinte modelo:

$$\begin{aligned} \text{max z} = {} & 5x_1 - x_2 \\ & 7x_1 - 5x_2 \leq 13 \\ & 3x_1 + 2x_2 \leq 17 \\ & x \in \mathbb{R}^2_+ \end{aligned}$$

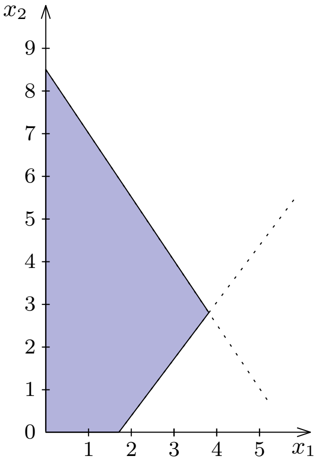
Formalização e regiões factíveis

Exemplo: Considere o seguinte modelo:

$$\begin{aligned} \text{max z} = {} & 5x_1 - x_2 \\ & 7x_1 - 5x_2 \leq 13 \\ & 3x_1 + 2x_2 \leq 17 \\ & \mathbf{x} \in \mathbb{R}^{\mathbf{2}}_{+} \end{aligned}$$

Note que pelo domínio das variáveis, estamos tratando de um problema de programação linear.

Formalização e regiões factíveis



E a sua **região factível** é dada pela área definida pela intersecção das restrições.

Formalização e regiões factíveis

Já um modelo de **programação inteira (PI)** admite valores inteiros não negativos para as suas variáveis, e assume a forma genérica:

$$
\begin{aligned}
\min \mathsf{z} = \mathbf{c}^T \mathbf{x} \\
\mathbf{A}\mathbf{x} = \mathbf{b} \\
\mathbf{x} \in \mathbb{Z}_+^n
\end{aligned}
$$

1. **A** é uma matriz mxn chamada matriz dos coeficientes.
2. $\mathbf{c}^T$ é o vetor dos custos.
3. $\mathbf{b}^T$ é o vetor dos termos independentes ou de recursos.

Formalização e regiões factíveis

Exemplo: Considere o seguinte modelo:

$$\begin{aligned} \text{max z} = {} & 5x_1 - x_2 \\ & 7x_1 - 5x_2 \leq 13 \\ & 3x_1 + 2x_2 \leq 17 \\ & x \in \mathbb{Z}_+^2 \end{aligned}$$

[1] O nome programação inteira "esconde" que a mesma também é linear, o correto seria "programação linear inteira", mas para diferenciar mais do termo PL, convenciona-se usar somente PI

Formalização e regiões factíveis

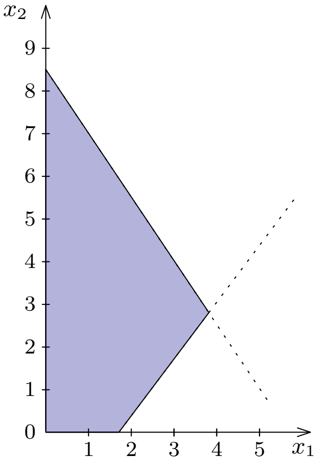
Exemplo: Considere o seguinte modelo:

$$\begin{aligned} \text{max z} = {} & 5x_1 - x_2 \\ & 7x_1 - 5x_2 \leq 13 \\ & 3x_1 + 2x_2 \leq 17 \\ & \mathbf{x} \in \mathbb{Z}_+^{\mathbf{2}} \end{aligned}$$

Note que pelo domínio das variáveis, estamos tratando de um problema de programação inteira[1].

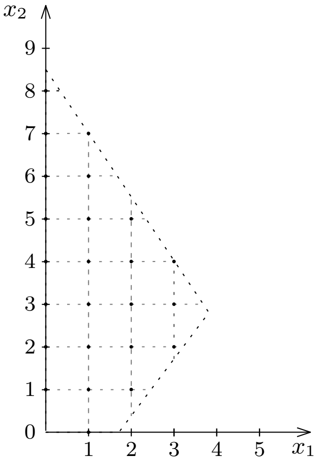
[1] O nome programação inteira "esconde" que a mesma também é linear, o correto seria "programação linear inteira", mas para diferenciar mais do termo PL, convenciona-se usar somente PI

Formalização e regiões factíveis



E agora, qual é a região factível do PI?

Formalização e regiões factíveis



Somente o **conjunto de pontos inteiros** que estão dentro da região delimitada pelas restrições.

Formalização e regiões factíveis

Um modelo de **programação inteira mista (PIM)** admite tanto valores inteiros quanto reais não negativos para as suas variáveis, e assume a forma genérica:

$$\begin{aligned} \min \text{z} = \mathbf{c}^T\mathbf{x} + \mathbf{d}^T\mathbf{y} \\ \mathbf{Ax} + \mathbf{Dy} = \mathbf{b} \\ \mathbf{x} \in \mathbb{R}_+^n, \mathbf{y} \in \mathbb{Z}_+^p \end{aligned}$$

1. **A** é uma matriz mxn e **D** uma matriz mxp.
2. $\mathbf{c}^T$ é um vetor $1xn$ e $\mathbf{y}^T$ um vetor $1xp$.
3. $\mathbf{b}^T$ é o vetor dos termos independentes ou de recursos.

Formalização e regiões factíveis

Exemplo: Considere o seguinte modelo:

$$
\begin{aligned}
\text{max z} = \; & 5x_1 - x_2 \\
& 7x_1 - 5x_2 && \leq 13 \\
& 3x_1 + 2x_2 && \leq 17 \\
& x_1 \in \mathbb{Z}_+, x_2 \in \mathbb{R}_+
\end{aligned}
$$

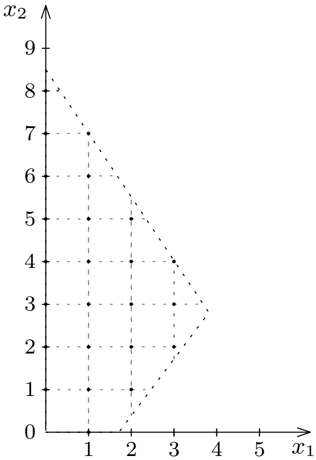
Formalização e regiões factíveis

Exemplo: Considere o seguinte modelo:

$$
\begin{aligned}
\max\ z = {} & 5x_1 - x_2 \\
& 7x_1 - 5x_2 \quad \leq 13 \\
& 3x_1 + 2x_2 \quad \leq 17 \\
& \mathbf{x_1} \in \mathbb{Z}_+, \mathbf{x_2} \in \mathbb{R}_+
\end{aligned}
$$

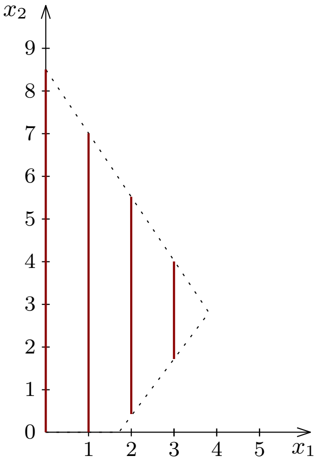
Note que pelo domínio das variáveis, estamos tratando de um problema de programação inteira mista (PIM), pois x_1 é inteiro e x_2 pertence aos reais.

Formalização e regiões factíveis



E agora, qual é a região factível do **PIM**?

Formalização e regiões factíveis



Somente os **segmentos de reta** que estão dentro da região delimitada pelas restrições.

Formalização e regiões factíveis

Um modelo de **programação binaria (PB)** admite somente valores 0 ou 1 para suas variáveis.

$$
\begin{aligned}
\text{min z} &= \mathbf{c}^T\mathbf{x} \\
\mathbf{A}\mathbf{x} &= \mathbf{b} \\
\mathbf{x} &\in \mathbb{B}^n_+
\end{aligned}
$$

Em que:

1. **A** é uma matriz mxn chamada matriz dos coeficientes.
2. $\mathbf{c}^T$ é o vetor dos custos.
3. $\mathbf{b}^T$ é o vetor dos termos independentes ou de recursos.
4. $\mathbb{B}^n$ representa o espaço dos vetores com n componentes binárias (0,1).

Formalização e regiões factíveis

Exemplo: Considere o seguinte modelo:

$$
\begin{aligned}
\text{max z} = {} & 5x_1 - x_2 \\
& 7x_1 - 5x_2 \leq 13 \\
& 3x_1 + 2x_2 \leq 17 \\
& x \in \mathbb{B}^2
\end{aligned}
$$

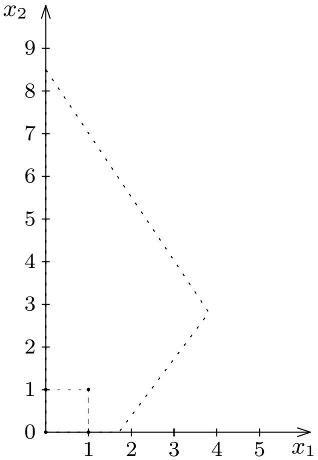
Formalização e regiões factíveis

Exemplo: Considere o seguinte modelo:

$$
\begin{aligned}
\text{max z} = {} & 5x_1 - x_2 \\
& 7x_1 - 5x_2 \leq 13 \\
& 3x_1 + 2x_2 \leq 17 \\
& \mathbf{x} \in \mathbb{B}^{\mathbf{2}}
\end{aligned}
$$

Note que pelo domínio das variáveis, estamos tratando de um problema de programação binária, pois x_1 e x_2 pertencem ao conjunto $\{0,1\}$.

Formalização e regiões factíveis



Somente os **pontos 0-1** que estão dentro da região delimitada pelas restrições.

Formalização e regiões factíveis

Conclusão

Quando falamos em programação inteira existem 3 possibilidades de modelos/formulações:

1. **PI: Programação inteira** - todas as variáveis são inteiras.
2. **PIM: Programação inteira mista** - existe uma mistura de variáveis reais e inteiras.
3. **PB: Programação binária** - As variáveis só podem assumir valores 0 ou 1.

Formalização e regiões factíveis

Conclusão

Quando falamos em programação inteira existem 3 possibilidades de modelos/formulações:

1. **PI: Programação inteira** - todas as variáveis são inteiras.
2. **PIM: Programação inteira mista** - existe uma mistura de variáveis reais e inteiras.
3. **PB: Programação binária** - As variáveis só podem assumir valores 0 ou 1.

ATENÇÃO: Embora os termos programação **linear** e programação **inteira** possam dar a entender que os programas inteiros não são lineares, isso não é verdade! Poderíamos chamá-los de **programas inteiros lineares**. É só uma questão de convenção.

Modelagem inteira e binária

Modelagem inteira e binária

Agora podemos entender alguns problemas que **só podem ser modelados usando a programação inteira** (a maioria dos problemas reais entram nesta categoria). Outros modelos serão vistos na continuação da disciplina (Pesquisa Operacional II).

Algumas das aplicações incluem:

1. Roteirização de veículos
2. Sequenciamento de ordens de produção
3. Carregamento de contêineres/paletes
4. Determinação de escalas de horários
5. Dimensionamento de frota

Modelagem inteira e binária

O problema da mochila

Considere que você irá acampar em uma montanha. Você vai levar uma mochila de capacidade 150kg, e deve escolher um subconjunto de itens para levar na viagem, de forma que o peso total da mochila não seja excedido. Sabendo ainda que cada item possui uma pontuação, valorando o quão útil ele é para a viagem, o seu objetivo é maximizar a soma da pontuação dos itens levados. Considere os itens abaixo, com a relação de pontuação e peso para cada um:

Considere que você irá acampar em uma montanha. Você vai levar uma mochila de capacidade 150kg, e deve escolher um subconjunto de itens para levar na viagem, de forma que o peso total da mochila não seja excedido. Sabendo ainda que cada item possui uma pontuação, valorando o quão útil ele é para a viagem, o seu objetivo é maximizar a soma da pontuação dos itens levados. Considere os itens abaixo, com a relação de pontuação e peso para cada um:

	Ipod	**Abobrinha**	$\mathbf{H_2O}$	**Canivete**	**Carne**	**Arroz**	**Aveia**	**PS4**
Valor	10	8	5	15	25	17	8	30
Peso	50	55	60	45	15	25	35	25

Como fica o modelo matemático para o **problema da mochila**?

Modelagem inteira e binária

O problema da mochila

Modelagem inteira e binária

O problema da mochila

Sejam as variáveis:

$$x_i = \begin{cases} 1 : \text{Se o item } i \text{ é levado na mochila} \\ 0 : \text{c.c} \end{cases}$$

Sejam as variáveis:

$$x_i = \begin{cases} 1 : \text{Se o item } i \text{ é levado na mochila} \\ 0 : \text{c.c} \end{cases}$$

$$\begin{aligned} \text{max z} = \quad & 10x_1 + 8x_2 + 5x_3 + 15x_4 + 25x_5 + 17x_6 + 8x_7 + 30x_8 \\ \text{Sujeito à} \quad & 50x_1 + 55x_2 + 60x_3 + 45x_4 + 15x_5 + 25x_6 + 35x_7 + 25x_8 \leq 150 \\ & x_i \in \{0, 1\}, i \in \{1, ..., 8\} \end{aligned}$$

Importante

Precisamos conseguir escrever os modelos matemáticos de forma genérica, para qualquer conjunto de dados. Por exemplo, se existirem mais 2 itens a serem escolhidos, a **natureza** da função objetivo e das restrições **permanece a mesma** (pois o problema continua sendo o mesmo!). Para escrever os modelos de forma genérica usamos parâmetros, conjuntos e a notação de somatório.

Modelagem inteira e binária

O problema da mochila

Relembrando alguns termos:

$$\sum_{i=1}^{5} x_i =$$

$$\sum_{i=1}^{5} x_i a_i =$$

Relembrando alguns termos:

$$\sum_{i=1}^{5} x_i = x_1 + x_2 + x_3 + x_4 + x_5$$

$$\sum_{i=1}^{5} x_i a_i =$$

Relembrando alguns termos:

$$\sum_{i=1}^{5} x_i = x_1 + x_2 + x_3 + x_4 + x_5$$

$$\sum_{i=1}^{5} x_i a_i = x_1 a_1 + x_2 a_2 + x_3 a_3 + x_4 a_4 + x_5 a_5$$

ATENÇÃO: Quando temos um somatório, basta expandirmos ele horizontalmente.

$$\sum_{i=1}^{2} x_{ij} \leq 0 \quad , j = 1, 2, 3.$$

Modelagem inteira e binária

O problema da mochila

$$\sum_{i=1}^{2} x_{ij} \leq 0 \quad , j = 1, 2, 3.$$

$$x_{11} + x_{21} \leq 0$$
$$x_{12} + x_{22} \leq 0$$
$$x_{13} + x_{23} \leq 0$$

ATENÇÃO: Neste caso existe um somatório que deve ser expandido para cada linha do conjunto $j = 1, 2, 3$, assim temos uma variação tanto em i (nos somatórios), quanto em j (cada restrição representa um novo j).

Voltando para o **modelo genérico da mochila**, temos:

Conjuntos:

I: Conjunto de itens.

Parâmetros:

w_i: Peso do item $i, \forall i \in I$.

v_i: Valor do item $i, \forall i \in I$.

C : Capacidade da mochila.

Variáveis:

$$x_i = \begin{cases} 1 : \text{Se o item } i \text{ é levado na mochila} \\ 0 : \text{c.c} \end{cases}$$

$x_i \in \{0, 1\} \forall i \in I$

Modelagem inteira e binária

O problema da mochila

O modelo fica então:

$$\text{max z} = \sum_{i \in I} x_i v_i$$

sa:

$$\sum_{i \in I} x_i w_i \leq C$$

$$x_i \in \{0, 1\}, i \in I$$

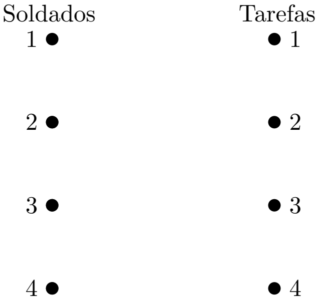
A função objetivo maximiza a soma dos valores de todos os itens que são levados. A única restrição do modelo garante que a capacidade da mochila não é excedida pelo peso dos itens levados.

Considere agora que você é responsável pela designação de 4 tarefas à 4 soldados de um exército. Você possui o histórico de notas de cada um desses soldados para cada uma das tarefas (notas de 0 a 10). Sabendo que cada soldado deve executar somente uma tarefa, e todas as tarefas devem ser executadas, bem como a tabela de notas ao lado. Qual é o modelo de programação inteira para o **problema de designação**?

	AT1	AT2	AT3	AT4
	3	2	3	7
	4	3	3	5
	1	7	3	3
	3	5	1	3

Modelagem inteira e binária

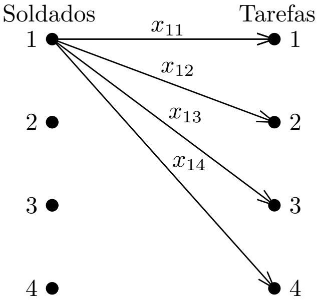
O problema de designação



Como no problema do transporte, podemos entender o problema da designação por um grafo. **Quais devem ser as variáveis para esse problema**?

Modelagem inteira e binária

O problema de designação



Podemos criar uma variável para cada combinação soldado-atividade, gerando 16 variáveis.

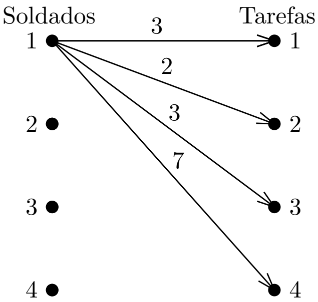
Variáveis:

$$x_{ij} = \begin{cases} 1 : \text{Se o soldado } i \text{ executa a atividade } j \\ 0 : \text{c.c} \end{cases}$$

$\forall i, j \in (1, ..., 4)$

Modelagem inteira e binária

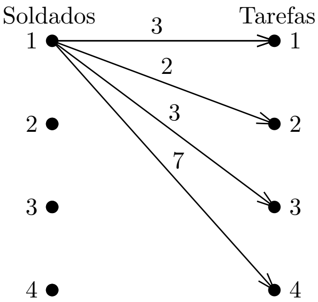
O problema de designação



A função objetivo deve maximizar a soma das aptidões dos soldados i que desempenham as tarefas j.

Modelagem inteira e binária

O problema de designação

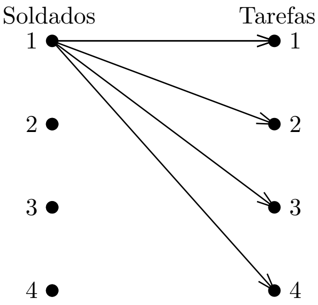


A função objetivo deve maximizar a soma das aptidões dos soldados i que desempenham as tarefas j.

$$\boxed{\max z = 3x_{11} + 2x_{12} + 3x_{13} + ... + 1x_{43} + 3x_{44}}$$

Modelagem inteira e binária

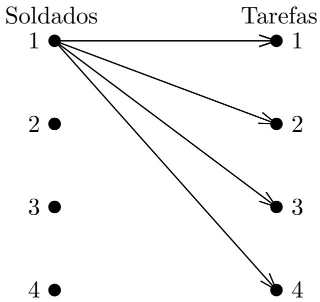
O problema de designação



A primeira restrição diz que todo **soldado deve executar uma única tarefa**. Para o soldado $i = 1$ temos:

Modelagem inteira e binária

O problema de designação

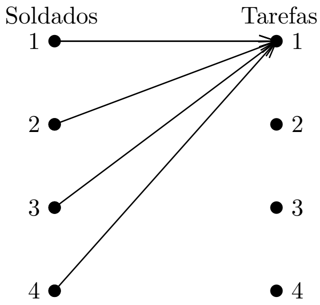


A primeira restrição diz que todo **soldado deve executar uma única tarefa**. Para o soldado $i = 1$ temos:

$$\boxed{x_{11} + x_{12} + x_{13} + x_{14} = 1}$$

Modelagem inteira e binária

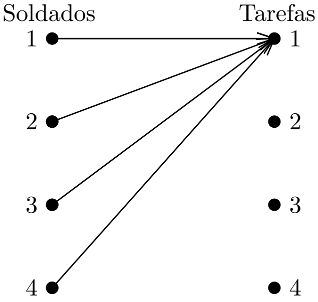
O problema de designação



Ainda, **toda tarefa deve ser executada por um soldado**. Para a tarefa $j = 1$ temos:

Modelagem inteira e binária

O problema de designação



Ainda, **toda tarefa deve ser executada por um soldado**. Para a tarefa $j = 1$ temos:

$$\boxed{x_{11} + x_{21} + x_{31} + x_{41} = 1}$$

Modelagem inteira e binária

O problema de designação

$$
\begin{aligned}
\text{max z} = 3x_{11} + 2x_{12} + 3x_{13} + \ldots + 1x_{43} + 3x_{44} \\
x_{11} + x_{12} + x_{13} + x_{14} = 1 \\
x_{21} + x_{22} + x_{23} + x_{24} = 1 \\
x_{31} + x_{32} + x_{33} + x_{34} = 1 \\
x_{41} + x_{42} + x_{43} + x_{44} = 1 \\
x_{11} + x_{21} + x_{31} + x_{41} = 1 \\
x_{12} + x_{22} + x_{32} + x_{42} = 1 \\
x_{13} + x_{23} + x_{33} + x_{43} = 1 \\
x_{14} + x_{24} + x_{34} + x_{44} = 1 \\
x_{ij} \in \{0, 1\}, \forall i, \forall j \in (1, 2, 3, 4)
\end{aligned}
$$

O modelo genérico **para o problema da designação**, fica então:

Conjuntos:

I: Conjunto de recursos (soldados) e atividades (tarefas).

Parâmetros:

a_{ij}: Valor do recurso i para a tarefa j $\forall i, j \in I$.

Variáveis:

$$x_{ij} = \begin{cases} 1 : \text{Se o soldado } i \text{ executa a atividade } j \\ 0 : \text{c.c} \end{cases}$$

O modelo fica então:

$$\text{max z} = \sum_{i,j \in I} x_{ij} a_{ij}$$

sa:

$$\sum_{j \in I} x_{ij} = 1, \forall i \in I$$

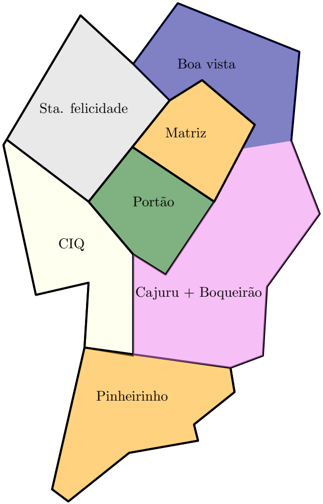
$$\sum_{i \in I} x_{ij} = 1, \forall j \in I$$

$$x_{ij}, \forall i, \forall j \in I$$

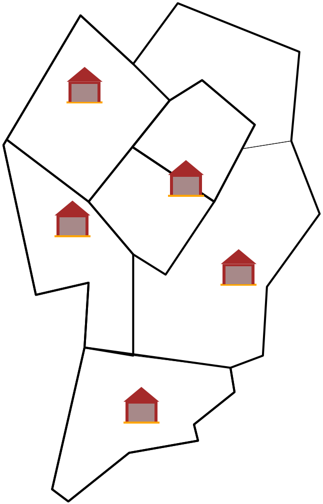
Modelagem inteira e binária

O problema da cobertura de conjuntos

Considere o conjunto de bairros de Curitiba (não todos).



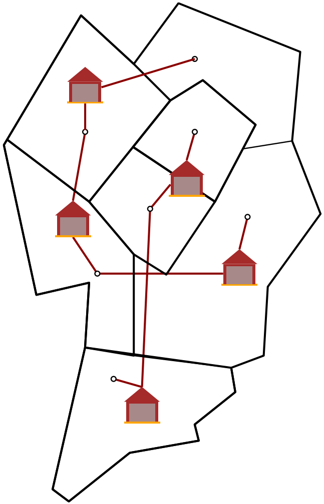
A prefeitura vai construir **novas estações contra-incêndios**. Existem locais possíveis para a construção dessa estações. Existe um **custo** para a construção de cada estação (*se a mesma for utilizada*).



Modelagem inteira e binária

O problema da cobertura de conjuntos

E pelas localizações das estações, sabemos **quais bairros que ela pode atender**. A restrição é de que o tempo-resposta de chegada ao local deve ser inferior a 8 minutos. A imagem ao lado mostra **quais estações podem atender quais bairros**.



A informação de quais estações podem atender quais bairros pode ser resumida por uma **matriz de adjacências**, com valores iguais a 1, se a estação da linha atende o bairro da coluna, e 0 caso contrário. Considere a matriz de adjacência abaixo, bem como os custos de instalação de cada estação (não é uma representação das imagens anteriores).

	Bairros						
Estações	**1**	**2**	**3**	**4**	**5**	**6**	**Custo**
1	1	0	0	0	1	0	500
2	0	1	0	0	0	1	600
3	0	0	0	1	0	1	450
4	1	0	1	0	0	0	800

A informação de quais estações podem atender quais bairros pode ser resumida por uma **matriz de adjacências**, com valores iguais a 1, se a estação da linha atende o bairro da coluna, e 0 caso contrário. Considere a matriz de adjacência abaixo, bem como os custos de instalação de cada estação (não é uma representação das imagens anteriores).

	Bairros						
Estações	**1**	**2**	**3**	**4**	**5**	**6**	**Custo**
1	1	0	0	0	1	0	500
2	0	1	0	0	0	1	600
3	0	0	0	1	0	1	450
4	1	0	1	0	0	0	800

Quais estações devem ser construídas de forma que **todos os bairros sejam atendidos por pelo menos uma delas**, e os **custos** totais de construção sejam **minimizados**?

Variáveis:

$$x_i = \begin{cases} 1 : \text{Se a estação } i \text{ é construída} \\ 0 : \text{c.c} \end{cases}$$

$\forall i \in (1, ..., 4)$

	Bairros						
Estações	**1**	**2**	**3**	**4**	**5**	**6**	**Custo**
1	1	0	0	0	1	0	500
2	0	1	0	0	0	1	600
3	0	0	0	1	0	1	450
4	1	0	1	0	0	0	800

A função objetivo deve minimizar os custos de construção, portanto:

	Bairros						
Estações	**1**	**2**	**3**	**4**	**5**	**6**	**Custo**
1	1	0	0	0	1	0	500
2	0	1	0	0	0	1	600
3	0	0	0	1	0	1	450
4	1	0	1	0	0	0	800

A função objetivo deve minimizar os custos de construção, portanto:

$$\text{min z} = 500x_1 + 600x_2 + 450x_3 + 800x_4$$

Modelagem inteira e binária

O problema da cobertura de conjuntos

	Bairros						
Estações	**1**	**2**	**3**	**4**	**5**	**6**	**Custo**
1	1	0	0	0	1	0	500
2	0	1	0	0	0	1	600
3	0	0	0	1	0	1	450
4	1	0	1	0	0	0	800

As únicas restrições são de cobertura, ou seja, considerando o bairro 1: **pelo menos** uma das 2 estações capazes de atendê-lo deve ser construída, assim, temos:

	Bairros						
Estações	**1**	**2**	**3**	**4**	**5**	**6**	**Custo**
1	1	0	0	0	1	0	500
2	0	1	0	0	0	1	600
3	0	0	0	1	0	1	450
4	1	0	1	0	0	0	800

As únicas restrições são de cobertura, ou seja, considerando o bairro 1: **pelo menos** uma das 2 estações capazes de atendê-lo deve ser construída, assim, temos:

$$\boxed{x_1 + x_4 \geq 1}$$

Fazendo isso para **todos os bairros**, temos:

	Bairros						
Estações	**1**	**2**	**3**	**4**	**5**	**6**	**Custo**
1	1	0	0	0	1	0	500
2	0	1	0	0	0	1	600
3	0	0	0	1	0	1	450
4	1	0	1	0	0	0	800

Bairros							
Estações	**1**	**2**	**3**	**4**	**5**	**6**	**Custo**
1	1	0	0	0	1	0	500
2	0	1	0	0	0	1	600
3	0	0	0	1	0	1	450
4	1	0	1	0	0	0	800

Fazendo isso para **todos os bairros**, temos:

$$\begin{aligned} x_1 + x_4 &\geq 1 \\ x_2 &\geq 1 \\ x_4 &\geq 1 \\ x_1 &\geq 1 \\ x_2 + x_3 &\geq 1 \end{aligned}$$

O modelo genérico para o **problema da cobertura de conjuntos** fica então:

Conjuntos:

I: Conjunto de estações.

J: Conjunto de bairros.

Parâmetros:

c_i: Custo de abrir a estação i, $\forall i \in I$.

a_{ij}: 1 se a estação i atende o bairro j, 0 c.c, $\forall j \in J, \forall i \in I$.

Variáveis:

$$x_i = \begin{cases} 1 : \text{Se a estação } i \text{ é construída} \\ 0 : \text{c.c} \end{cases} \qquad \forall i \in I$$

Modelagem inteira e binária

O problema da cobertura de conjuntos

O modelo fica então:

$$\text{min z} = \sum_{i \in I} x_i c_i$$

sa:

$$\sum_{i \in I} a_{ij} x_i \geq 1, \forall j \in J$$

$$x_i \in \{0, 1\}, \forall i \in I$$

OBS: Se a restrição for de igualdade (ou seja, cada região deve ser atendida **exatamente** por uma estação), o problema é chamado de **partição de conjuntos**.

$$\text{min z} = \sum_{i \in I} x_i c_i$$

sa:

$$\sum_{i \in I} a_{ij} x_i = 1, \forall j \in J$$

$$x_i \in \{0, 1\}, \forall i \in I$$

Possibilidades da modelagem binária

Possibilidades da modelagem binária

Modelagem de situações complexas

Vimos alguns modelos simples que usam variáveis inteiras/binárias, no entanto, as variáveis binárias podem ser usadas para modelar situações mais complexas, tais como:

1. Relações lógicas
2. Big M
   2.1 Implicações "se então"
   2.2 Custo fixo
   2.3 Restrição ativa ou inativa
   2.4 Restrições disjuntivas

Vamos usar um exemplo para verificar as possibilidades de modelagem de restrições lógicas.

Considere que você está interessado em escolher um conjunto de investimentos dentre 7 possibilidades $\{1, ..., 7\}$. Sem considerar a função objetivo, ou mesmo os retornos esperados para os investimentos, e sabendo que as variáveis de decisão são:

Possibilidades da modelagem binária

Relações lógicas

Vamos usar um exemplo para verificar as possibilidades de modelagem de restrições lógicas.

> Considere que você está interessado em escolher um conjunto de investimentos dentre 7 possibilidades $\{1, ..., 7\}$. Sem considerar a função objetivo, ou mesmo os retornos esperados para os investimentos, e sabendo que as variáveis de decisão são:

Variáveis:

$$x_i = \begin{cases} 1 : \text{Se o investimento } i \text{ é escolhido} \\ 0 : \text{c.c} \end{cases} \qquad x_i \in \{0, 1\} \forall i \in 1, .., 7$$

Modele as seguintes restrições.

Possibilidades da modelagem binária

Relações lógicas

I - Você não pode investir em todos eles

II - Você deve investir em pelo menos um deles

III - Você pode realizar no máximo 3 investimentos

IV - O investimento 1 ou (incluso) o investimento 2 deve ser selecionado

Possibilidades da modelagem binária

Relações lógicas

I - Você não pode investir em todos eles

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 \leq 7$$

II - Você deve investir em pelo menos um deles

III - Você pode realizar no máximo 3 investimentos

IV - O investimento 1 ou (incluso) o investimento 2 deve ser selecionado

I - Você não pode investir em todos eles

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 \leq 7$$

II - Você deve investir em pelo menos um deles

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 \geq 1$$

III - Você pode realizar no máximo 3 investimentos

IV - O investimento 1 ou (incluso) o investimento 2 deve ser selecionado

I - Você não pode investir em todos eles

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 \leq 7$$

II - Você deve investir em pelo menos um deles

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 \geq 1$$

III - Você pode realizar no máximo 3 investimentos

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 \leq 3$$

IV - O investimento 1 ou (incluso) o investimento 2 deve ser selecionado

I - Você não pode investir em todos eles

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 \leq 7$$

II - Você deve investir em pelo menos um deles

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 \geq 1$$

III - Você pode realizar no máximo 3 investimentos

$$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 \leq 3$$

IV - O investimento 1 ou (incluso) o investimento 2 deve ser selecionado

$$x_1 + x_2 \geq 1$$

Big M

Existem muitas situações que só podem ser modeladas com restrições que exigem um conhecimento prévio dos valores máximos (ou mínimos) que uma desigualdade pode atingir. Por padrão esse número é sempre representado por **M** (ou seja, big-M). Como se trata de um recurso de modelagem (combinação de variáveis e restrições que modelam uma situação), o funcionamento dessas restrições pode não ser tão intuitivo quanto as que vimos até agora, de forma que é preciso pensar um pouco e nos convencermos de que elas funcionam!

Big M

Existem muitas situações que só podem ser modeladas com restrições que exigem um conhecimento prévio dos valores máximos (ou mínimos) que uma desigualdade pode atingir. Por padrão esse número é sempre representado por **M** (ou seja, big-M). Como se trata de um recurso de modelagem (combinação de variáveis e restrições que modelam uma situação), o funcionamento dessas restrições pode não ser tão intuitivo quanto as que vimos até agora, de forma que é preciso pensar um pouco e nos convencermos de que elas funcionam!

Utilizamos esse recurso para modelar:

1. Implicações do tipo "se então"
2. Custo fixo
3. Restrição ativa ou inativa
4. Restrições disjuntivas

Considere o mesmo exemplo dos 7 investimentos usado na explicação das relações lógicas. Como poderíamos modelar o seguinte caso:

O investimento 4 só pode ser escolhido se o 2 também for

Considere o mesmo exemplo dos 7 investimentos usado na explicação das relações lógicas. Como poderíamos modelar o seguinte caso:

O investimento 4 só pode ser escolhido se o 2 também for

Para entender melhor os comportamentos que queremos modelar, um recurso é o de **atribuir valores para as variáveis envolvidas** e analisar quais combinações são factíveis e quais não são. As variáveis envolvidas são x_4 e x_2, sendo que cada uma só pode assumir valores 0 ou 1. Isso implica nas seguintes combinações de valores:

O investimento 4 só pode ser escolhido se o 2 também for

x_4	x_2	Factível?

Vamos analisar todas as combinações de valores possíveis.

O investimento 4 só pode ser escolhido se o 2 também for

x_4	x_2	Factível?
0	0	✓

Nenhum dos dois investimentos ser selecionado, é uma possibilidade factível, ou seja, **se 2 não for escolhido 4 não precisa ser escolhido**.

O investimento 4 só pode ser escolhido se o 2 também for

x_4	x_2	Factível?
0	0	✓
0	1	✓

Da mesma forma, se 2 for selecionado 4 não precisa ser também.

O investimento 4 só pode ser escolhido se o 2 também for

x_4	x_2	Factível?
0	0	✓
0	1	✓
1	0	✗

No entanto, se 2 não for selecionado **4 também não pode ser selecionado**, fazendo a combinação acima infactível.

Big M

Implicações do tipo - se então

O investimento 4 só pode ser escolhido se o 2 também for

x_4	x_2	Factível?
0	0	✓
0	1	✓
1	0	✗
1	1	✓

Finalmente, ambos serem selecionados é uma possibilidade factível.

Big M

Implicações do tipo - se então

Precisamos então desdobrar todas as combinações da tabela em uma (ou mais) equação/inequação linear.

x_4	x_2	Factível?
0	0	✓
0	1	✓
1	0	✗
1	1	✓

$$?? + ?? + ?? \leq \geq = ??$$

Precisamos então desdobrar todas as combinações da tabela em uma (ou mais) equação/inequação linear.

x_4	x_2	Factível?
0	0	✓
0	1	✓
1	0	✗
1	1	✓

$$?? + ?? + ?? \leq \geq = ??$$

A restrição que modela a situação fica:

$$x_4 \leq x_2$$

Big M

Implicações do tipo - se então

Vamos analisar o funcionamento da restrição considerando a tabela de possibilidade:

x_4	x_2	$x_4 \leq x_2$	Factível
0	0	$0 \leq 0$	✓
0	1	$0 \leq 1$	✓
1	0	$1 \leq 0$	✗
1	1	$1 \leq 1$	✓

Note que a restrição **modela todas as situações da tabela**. Estamos modelando a situação do tipo "se-então" (se x_4 então x_2).

Big M

Implicações do tipo - se então

De **forma geral**, podemos modelar a situação genérica de que, se uma das variáveis

$$x_1 + x_2 + ... + x_k$$

tiver valor maior do que zero, implica que uma variável binária y deve ser 1, como:

$$\boxed{x_1 + x_2 + ... + x_k \leq My}$$

Em que $\mathbf{M}$ é um limitante superior para os valores possíveis de $x_1 + x_2 + ... + x_k$.

Big M

Implicações do tipo - se então

Note que no exemplo anterior, temos que $\mathbf{M} = 1$:

$$x_4 \leq x_2$$

Big M

Implicações do tipo - se então

Note que no exemplo anterior, temos que $\mathbf{M} = 1$:

$$x_4 \leq x_2$$

O maior valor que x_4 pode assumir é 1. Escrevendo a restrição na forma genérica:

$$x_4 \leq 1x_2$$

Big M

Implicações do tipo - se então

Note que no exemplo anterior, temos que $\mathbf{M} = 1$:

$$x_4 \leq x_2$$

O maior valor que x_4 pode assumir é 1. Escrevendo a restrição na forma genérica:

$$x_4 \leq 1x_2$$

Os investimentos 1, 2 e 3 só podem ser escolhidos se o 4 também o for

Note que temos uma situação de se-então, de forma que podemos recorrer a inequação genérica:

Os investimentos 1, 2 e 3 só podem ser escolhidos se o 4 também o for

$$x_1 + x_2 + x_3 \leq \mathbf{M} x_4$$

Os investimentos 1, 2 e 3 só podem ser escolhidos se o 4 também o for

$$x_1 + x_2 + x_3 \leq \mathbf{M}x_4$$

Basta descobrirmos o valor do limitante $\mathbf{M}$. Como todas as variáveis são binárias, temos que o máximo valor possível para o lado esquerdo da inequação é 3, de forma que podemos modelar a situação como:

$$\boxed{x_1 + x_2 + x_3 \leq 3x_4}$$

Os investimentos 1, 2 e 3 só podem ser escolhidos se o 4 também o for

$$x_1 + x_2 + x_3 \leq \mathbf{M}x_4$$

Basta descobrirmos o valor do limitante $\mathbf{M}$. Como todas as variáveis são binárias, temos que o máximo valor possível para o lado esquerdo da inequação é 3, de forma que podemos modelar a situação como:

$$\boxed{x_1 + x_2 + x_3 \leq 3x_4}$$

A **mecânica** da restrição é que, se o lado esquerdo assume algum valor (qualquer uma das variáveis ≥ 0) x_4 é obrigado a assumir valor para manter a inequação verdadeira.

Com o conhecimento de como modelar situações do tipo **se-então**, conseguimos avançar para a modelagem de **custos fixos**. Custos fixos surgem nas situações em que, quando um recurso é utilizado, independentemente da quantidade ou tempo, existe um custo fixo associado. Em contrapartida, se nenhuma unidade do recurso é utilizada não existe custo fixo.

Vamos entender por meio de um exemplo.

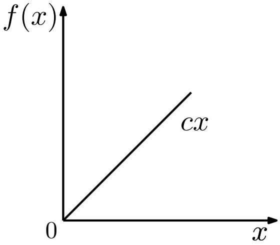
Big M

Custos fixos

Considere uma fábrica que produz um produto x. Cada unidade produzida de x gera um custo de c. Considerando somente essa situação, podemos modelar a função custo como fazemos usualmente:

Considere uma fábrica que produz um produto x. Cada unidade produzida de x gera um custo de c. Considerando somente essa situação, podemos modelar a função custo como fazemos usualmente:

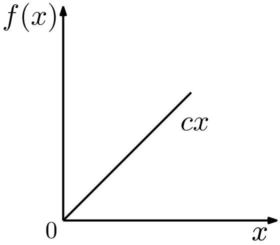
$$\boxed{f(x) = cx}$$



Big M
Custos fixos

Considere uma fábrica que produz um produto x. Cada unidade produzida de x gera um custo de c. Considerando somente essa situação, podemos modelar a função custo como fazemos usualmente:

$$\boxed{f(x) = cx}$$



Imagine agora que a produção de x implica no **aluguel de uma máquina** a custo S. Independentemente da quantidade que for produzida de x (> 0), o custo S é constante.

Note que **não podemos** modelar essa função simplesmente por:

$$f(x) = cx + S$$

Note que **não podemos** modelar essa função simplesmente por:

$$\boxed{f(x) = cx + S}$$

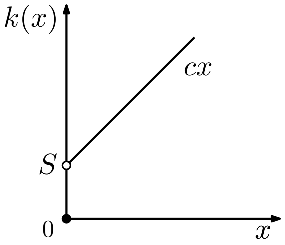
Nesse caso, mesmo quando nenhuma quantidade do produto é produzida, ainda teríamos o custo de aluguel da máquina S.

Essa nova função é descrita por:

$$k(x) = \begin{cases} 0 & \text{se } x = 0 \\ cx + S & \text{se } x > 0 \end{cases}$$

Essa nova função é descrita por:

$$
k(x) = \begin{cases} 0 & \text{se } x = 0 \\ cx + S & \text{se } x > 0 \end{cases}
$$



Como modelar linearmente essa situação? Implicitamente a essa situação, podemos enxergar um caso de "se-então", visto anteriormente.

Big M

Custos fixos

Vamos observar somente a função $k(x)$ quando $x > 0$:

$$k(x) = cx + S$$

Da forma como está, a função não desempenha o esperado. Quando $x = 0$ a parcela cx é anulada automaticamente (correto), porém o custo fixo S não.

Big M

Custos fixos

Vamos observar somente a função $k(x)$ quando $x > 0$:

$$k(x) = cx + S$$

Da forma como está, a função não desempenha o esperado. Quando $x = 0$ a parcela cx é anulada automaticamente (correto), porém o custo fixo S não. Essa dicotomia de anular ou não S pode ser representada por uma nova **variável binária**, alocada multiplicando o próprio s. Seja y essa variável:

$$k(x) = cx + yS$$

Quando $y = 0$ o custo s é anulado. Agora só precisamos criar uma restrição que ativa y sempre que $x > 0$, ou seja: se $x > 0$ então $y = 1$ (uma situação se-então!).

Big M

Custos fixos

Então, para modelar o custo fixo na função $k(x)$, basta adicionarmos a variável binária y e uma nova restrição do tipo "se-então" que ativa y quando $x > 0$:

$$k(x) = cx + yS$$
$$x \leq My$$

Big M

Custos fixos

Então, para modelar o custo fixo na função $k(x)$, basta adicionarmos a variável binária y e uma nova restrição do tipo "se-então" que ativa y quando $x > 0$:

$$k(x) = cx + yS$$
$$x \leq My$$

ATENÇÃO: Lembre quais as possibilidades que a restrição $x \leq My$ permite para as variáveis:

Big M

Custos fixos

Então, para modelar o custo fixo na função $k(x)$, basta adicionarmos a variável binária y e uma nova restrição do tipo "se-então" que ativa y quando $x > 0$:

$$k(x) = cx + yS$$
$$x \leq My$$

ATENÇÃO: Lembre quais as possibilidades que a restrição $x \leq My$ permite para as variáveis:

x	y	Factível?
0	0	✓
0	1	✓
1	0	✗
1	1	✓

Big M

Custos fixos

Então, para modelar o custo fixo na função $k(x)$, basta adicionarmos a variável binária y e uma nova restrição do tipo "se-então" que ativa y quando $x > 0$:

$$k(x) = cx + yS$$
$$x \leq My$$

ATENÇÃO: Lembre quais as possibilidades que a restrição $x \leq My$ permite para as variáveis:

x	y	Factível?
0	0	✓
0	1	✓
1	0	✗
1	1	✓

Note que é possível que y tenha valor e x não. Esse comportamento não modela o custo fixo. **Por quê então ele funciona da forma que criamos?**

Como essas situações usualmente envolvem um **custo** propriamente dito, a função objetivo será de minimização, sendo que o conjunto completo fica:

$$\begin{aligned} \text{min } k &= cx + yS \\ x &\leq My \\ x &\in \mathbb{Z}^+, y \in \{0, 1\} \end{aligned}$$

Como essas situações usualmente envolvem um **custo** propriamente dito, a função objetivo será de minimização, sendo que o conjunto completo fica:

$$\begin{aligned} \text{min } k &= cx + yS \\ x &\leq My \\ x &\in \mathbb{Z}^+, y \in \{0,1\} \end{aligned}$$

A **mecânica** da restrição deve ser analisada em conjunto com a função objetivo: embora y possa ser 1 quando $x = 0$, como a função objetivo deve ser minimizada, a solução ótima sempre vai optar por zerar y.

Implementação no GUSEK

Implementação no GUSEK

- Podemos implementar facilmente modelos com variáveis inteiras/binárias nos arquivos .lp do GUSEK. O vídeo *1.1 Solver GUSEK* da página de treinamentos mostra todas as possibilidades (a partir do modelo 2).
- Existem linguagens de programação matemática para automatizar a criação de modelos maiores, uma dessas linguagens é o MathProg. Para um treinamento na ferramenta, acesse o tutorial Mathprog, video 1.2 na página de treinamentos da disciplina.