

Programação Matemática II

Capítulo 2: Programação Linear Inteira

Raquel Bernardino

rbernardino@iseg.ulisboa.pt
Gabinete 511 Quelhas 6

Sumário

Motivação

Introdução

Formulações

Algoritmos para programação inteira

Geração de desigualdades válidas

Sumário

Motivação

Introdução

Formulações

Algoritmos para programação inteira

Geração de desigualdades válidas

Motivação

- Todos os dias estamos perante problemas de Investigação Operacional, como por exemplo:
  - Determinar o caminho rápido para vir para o ISEG.
  - Decidir onde fazer as compras para a semana (distância versus preços).

- Para formular estes problemas muitas vezes precisamos de variáveis inteiras.

Exemplos de aplicação da Investigação Operacional



(a) 2017



(b) 2019



(c) 2020



(d) 2023

Figura 1: Prémios aplicação inovadoras da IO da revista científica *European Journal of Operations Research*.

Sumário

Motivação

Introdução

Formulações

Algoritmos para programação inteira

Geração de desigualdades válidas

Introdução

Um problema de programação linear inteira mista (PLIM) é um problema de programação linear em que algumas variáveis têm obrigatoriamente valores inteiros. $\Longrightarrow$ Perda da propriedade da divisibilidade da PL.

- Num problema de programação linear inteira (PLI) todas as variáveis têm obrigatoriamente valores inteiros.
- Num problema de programação linear binária todas as variáveis têm obrigatoriamente valor binário, isto é, valor 0 ou 1.

Um problema de otimização combinatória (POC) é um PLIM cuja região admissível (RA) é um conjunto finito.

- A solução ótima é um subconjunto de um conjunto finito - a RA.

Introdução

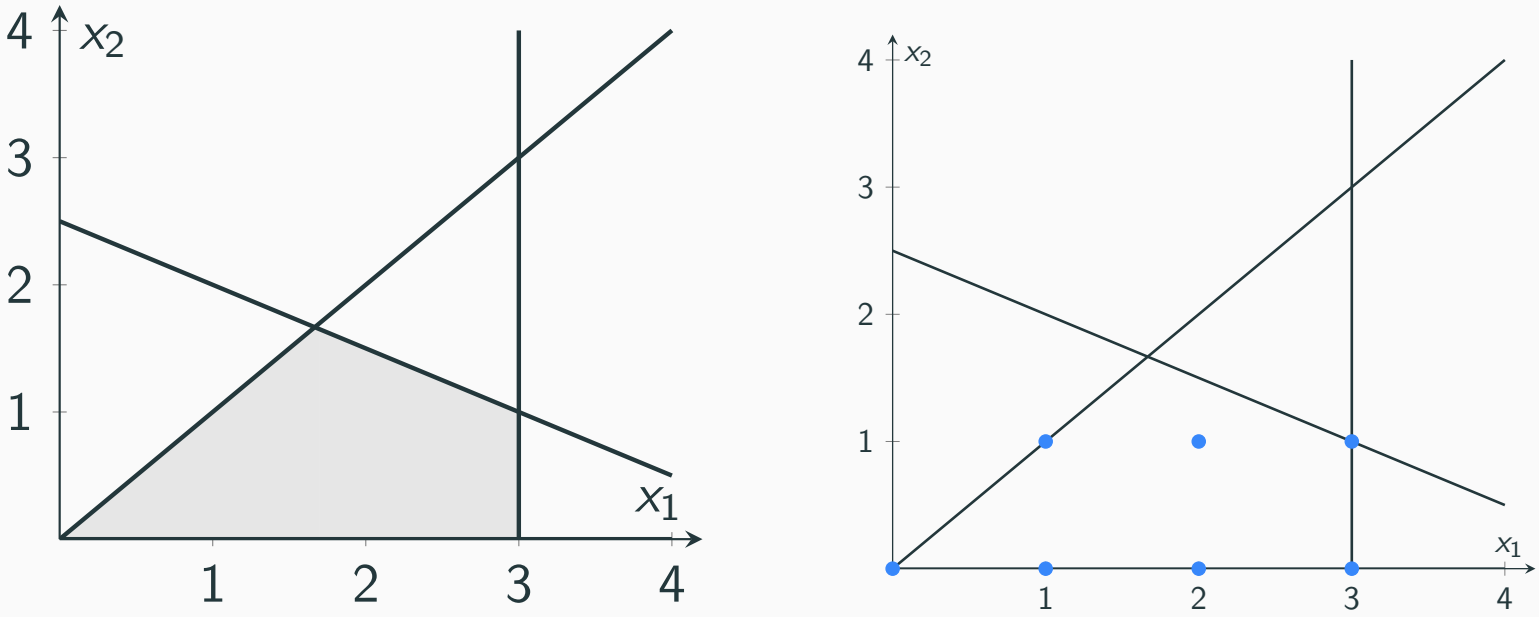


Figura 2: RA de um PL versus RA de um POC.

Introdução

Exemplos de problemas de otimização combinatória são:

- o problema da afetação;
- o problema dos transportes;
- o problema do saco-mochila;
- o problema de *lot-sizing*;
- o problema do caixeiro viajante; ou
- o problema do roteamento de veículos.

Introdução

Como podemos resolver um problema de otimização combinatória?

- Como a RA é um conjunto finito, podemos calcular todas as soluções admissíveis (SA) e determinar a melhor. $\Longrightarrow$ Enumeração!

- A enumeração da RA de um POC pode ser um processo muito demorado.

Que algoritmos podemos usar para determinar a SO de um POC?

- Algoritmo de *branch-and-bound*
- Algoritmo de cortes de Gomory

Sumário

Motivação

Introdução

Formulações

- Problemas clássicos
- Formulação de condições específicas

Algoritmos para programação inteira

Geração de desigualdades válidas

Problema do Saco-Mochila

Dado:

- um conjunto de itens $\mathcal{I}$, onde o item $i \in \mathcal{I}$ tem peso p_i e valor v_i; e
- um saco-mochila com capacidade P;

o Problema do Saco-Mochila consiste em determinar que itens devem ser incluídos no saco-mochila de forma a maximizar o valor total sem exceder a sua capacidade.

Problema do Saco-Mochila: uma instância

Considere-se um saco-mochila com capacidade 10 e um conjunto de seis itens com as seguintes caraterísticas:

Item (i)	1	2	3	4	5	6
Peso (p_i)	7	2	2	4	6	8
Valor (v_i)	10	3	5	5	8	12

Conseguem dar-me exemplos de soluções admissíveis?

Problema do Saco-Mochila: formulação

- Variáveis:

$$x_i = \begin{cases} 1, & \text{se o item } i \in \mathcal{I} \text{ é escolhido} \\ 0, & \text{caso contrário} \end{cases}$$

- Formulação:

$$\max \sum_{i \in \mathcal{I}} v_i x_i$$

$$\text{sujeito a: } \sum_{i \in \mathcal{I}} p_i x_i \leq P$$

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

Problema do Saco-Mochila: exemplos de restrições lógicas

- Um dos artigos 2 ou 3 deve ser escolhido
  - $x_2 + x_3 \geq 1$
- Podemos escolher no máximo 3 artigos
  - $x_1 + x_2 + x_3 + x_4 + x_5 + x_6 \leq 3$
- Se o artigo 2 é escolhido, então o artigo 4 também deve ser escolhido
  - $x_2 \leq x_4$
- Se o artigo 1 é escolhido, então o artigo 3 não pode ser escolhido
  - $x_1 + x_3 \leq 1$
- Se o artigo 1 é escolhido, deveremos escolher mais outro artigo
  - $x_1 \leq x_2 + x_3 + x_4 + x_5 + x_6$
- Se o artigo 1 é escolhido, deveremos escolher também os artigos 3 e 4
  - $x_1 \leq x_3$ e $x_1 \leq x_4$
- Se o artigo 2 é escolhido, não podemos escolher o artigo 3 nem o 4
  - $x_2 + x_3 \leq 1$ e $x_2 + x_4 \leq 1$

Problema do Lot-Sizing

Dado:

- uma linha de produção de um único produto;
- um horizonte temporal com um conjunto T de períodos;
- a procura d_t do produto no período $t \in T$;
- o custo de produção p_t de uma unidade do produto no período $t \in T$;
- o custo fixo de produzir f_t o produto no período $t \in T$; e
- o custo de armazenar h_t uma unidade do produto durante um período $t \in T$;

o Problema do Lot-Sizing consiste em determinar quantas unidades produzir em cada período de forma a satisfazer a sua procura, tendo em conta que a procura pode ser satisfeita por unidades produzidas ou armazenadas, de forma a minimizar o custo total, que é composto pelos custos unitários, de produção e de armazenamento.

Problema do Lot-Sizing: uma instância

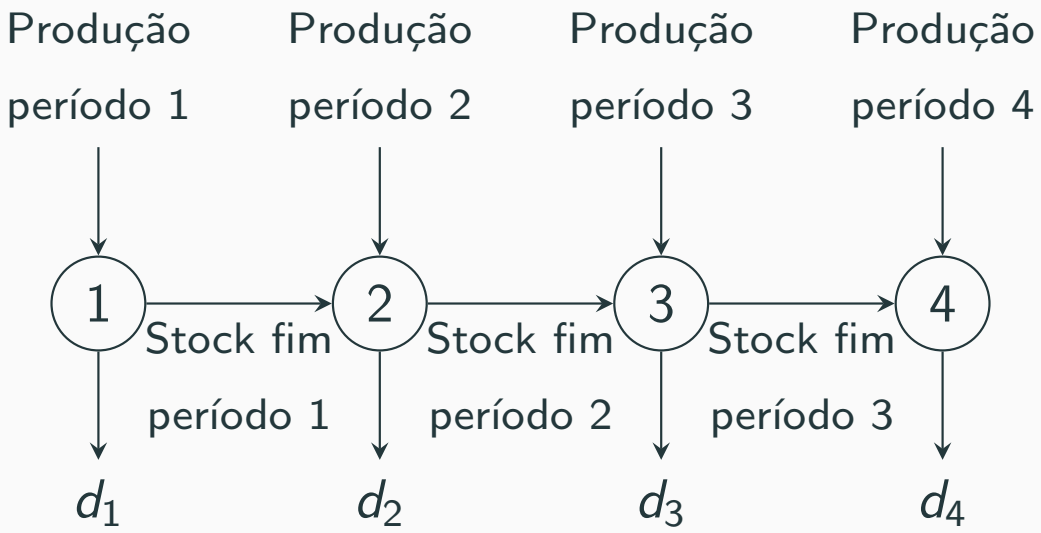
Considere-se um horizonte temporal com $T = \{1, 2, 3, 4\}$ períodos. Em cada período, o custo unitário de produção é 1 u.m., o custo fixo de produção é 10 u.m. e o custo de armazenamento é 1 u.m. por unidade e por período, ou seja, $p_t = h_t = 1$ e $f_t = 10, \forall t \in T$. A procura de cada período é apresentada na tabela seguinte:

Período (t)	1	2	3	4
Procura (d_t)	7	6	8	4

Conseguem dar-me exemplos de soluções admissíveis?

Problema do Lot-Sizing: uma instância

- Representação esquemática:



Problema do Lot-Sizing: formulação

- Variáveis:

  x_t : quantidade produzida no período $t \in T$

  s_t : quantidade em stock no fim do período $t \in T \cup \{0\}$

  $$y_t = \begin{cases} 1, & \text{se há produção no período } t \in T \\ 0, & \text{caso contrário} \end{cases}$$

- Formulação:

$$\begin{aligned} \min \quad & \sum_{t=1}^{T} (f_t y_t + p_t x_t + h_t s_t) & \\ & s_{t-1} + x_t = d_t + s_t & t \in T \\ & s_0 = 0 & \\ & x_t \leq M_t y_t & t \in T \\ & x_t \geq 0,\ s_t \geq 0,\ y_t \in \{0,1\} & t \in T \end{aligned}$$

  - M_t é uma constante com valor suficientemente grande

Problema do Lot-Sizing: exemplos de restrições de limites de variáveis

- Se há produção no período t, a linha de produção tem que funcionar
  - $x_t \leq M_t y_t$ M_t é um majorante da quantidade produzida no período t
- Se a capacidade de produção no período t é C_t
  - $x_t \leq C_t y_t$
- Se há produção no período t, é necessário produzir pelo menos L_t unidades
  - $x_t \geq L_t y_t$ e $x_t \leq M_t y_t$

Problema do Caixeiro Viajante

Dado:

- um conjunto de cidades N; e
- custos de deslocação entre cada par de cidades c_{ij}, $i, j \in N : i \neq j$,

o Problema do Caixeiro Viajante consiste em determinar a rota de menor custo que passe por todas as cidades uma e uma só vez.

Palestra de William Cook sobre o problema do caixeiro viajante:
`https://www.youtube.com/watch?v=5VjphFYQKj8&t=55s` [Euro 2019, Dublin]

Site com um *solver* para o caixeiro viajante:
`https://www.math.uwaterloo.ca/tsp/index.html`

Problema do Caixeiro Viajante: uma instância

Considere-se um conjunto de cinco cidades $\{1, 2, 3, 4, 5\}$ com os seguintes custos de deslocação entre as mesmas:

	1	2	3	4	5
1	-	3	5	19	2
2	3	-	14	15	11
3	5	14	-	20	1
4	19	15	20	-	9
5	2	11	1	9	-

Conseguem dar-me exemplos de soluções admissíveis?

Problema do Caixeiro Viajante: formulação

- Variáveis:

$$x_{ij} = \begin{cases} 1, \text{se o arco } (i,j) \in A \text{ é atravessado pelo caixeiro viajante} \\ 0, \text{caso contrário} \end{cases}$$

u_i : posição que o nodo $i \in N \setminus \{1\}$ ocupa na rota.

- Formulação:

$$\min \sum_{(i,j)\in A} c_{ij} x_{ij}$$

sujeito a:

$$\sum_{i\in N} x_{ji} = 1 \qquad \forall j \in N \qquad \text{(um arco sai de j)}$$

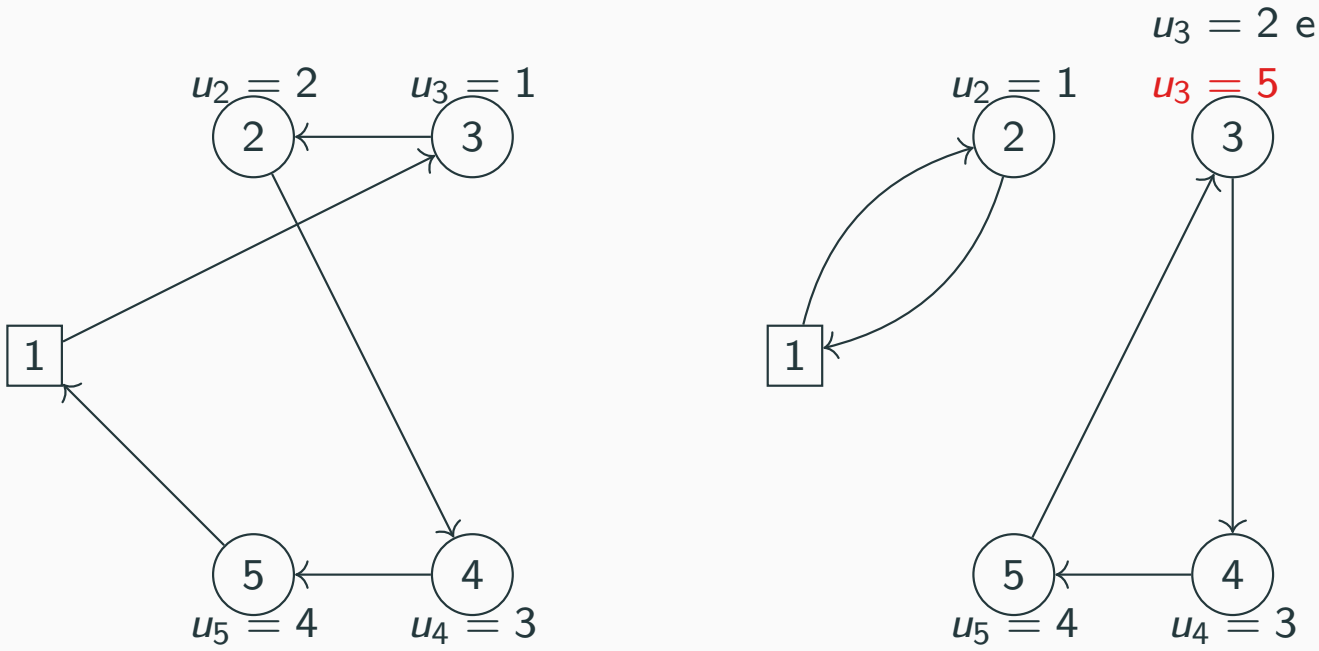
$$\sum_{i\in N} x_{ij} = 1 \qquad \forall j \in N \qquad \text{(um arco entra de j)}$$

$$u_i + (|N|-1)x_{ij} \leq u_j + |N| - 2 \qquad \forall (i,j) \in A : i,j \neq 1 \qquad \text{(posição j é a do i mais 1, se vamos de i para j)}$$

$$1 \leq u_i \leq |N| - 1 \qquad \forall i \in N \setminus \{1\} \qquad \text{(limites variáveis u)}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in A$$

Problema do Caixeiro Viajante: formulação



(a) Solução admissível TSP.

(b) Solução não admissível TSP.

Figura 3: Como funciona a formulação para o caixeiro viajante.

Disjunção de restrições

- Já vimos como formular algumas condições específicas:
  - Restrições lógicas
    - Exemplos: itens incompatíveis e limite ao itens escolhidos.
  - Limites de quantidades
    - Exemplos: quantidades mínimas e quantidades máximas.

Resta-nos ver como podemos formular a disjunção de restrições.

Disjunção de restrições: exemplo

Considere-se as seguintes restrições:

- A: $3x_1 + 4x_2 \leq 5$
- B: $5x_1 + 2x_2 \leq 2$

Uma desta restrições tem de ser satisfeita (a restrição A ou a B é satisfeita).

- Variáveis:

$$y = \begin{cases} 1 & \text{se a restrição A é satisfeita} \\ 0 & \text{se a restrição B é satisfeita} \end{cases}$$

- Formulação:

$$3x_1 + 4x_2 \leq 5 + M(1 - y)$$
$$5x_1 + 2x_2 \leq 2 + My$$
$$y \in \{0, 1\}$$

- M é uma constante com valor suficientemente grande

Disjunção de restrições: caso geral

Considere-se as seguintes restrições:

- A^i: $a^i x \leq b_i$, com $i = 1, \ldots, m$

Pelo menos k destas restrições têm de ser satisfeitas.

- Variáveis:

$$y_i = \begin{cases} 1 & \text{se a restrição } A^i \text{ é satisfeita, com i} = 1, \ldots, \text{m} \\ 0 & \text{caso contrário} \end{cases}$$

- Formulação:

$$a^i x \leq b_i + M(1 - y_i), \qquad i = 1, \ldots, m$$

$$\sum_{i=1}^{m} y_i \geq k$$

$$y_i \in \{0, 1\}, \qquad i = 1, \ldots, m$$

- M é uma constante com valor suficientemente grande

Sumário

Motivação

Introdução

Formulações

Algoritmos para programação inteira

- Algoritmo de branch-and-bound
- Algoritmo de planos de corte de Gomory

Geração de desigualdades válidas

Conceitos base: relaxação

Definição: Relaxação

Um problema

$$(RP) \qquad z^R = \min\{f(x) : x \in T \subseteq \mathbb{R}^n\}$$

é uma *relaxação* de

$$(IP) \qquad z = \min\{c(x) : x \in X \subseteq \mathbb{R}^n\}$$

se:

(i) $X \subseteq T$; e

(ii) $f(x) \leq c(x), \forall x \in X.$

Maximização: $f(x) \geq c(x), \quad \forall x \in X$

Conceitos base: relaxação

Maximização: $z^R \geq z$

Proposição

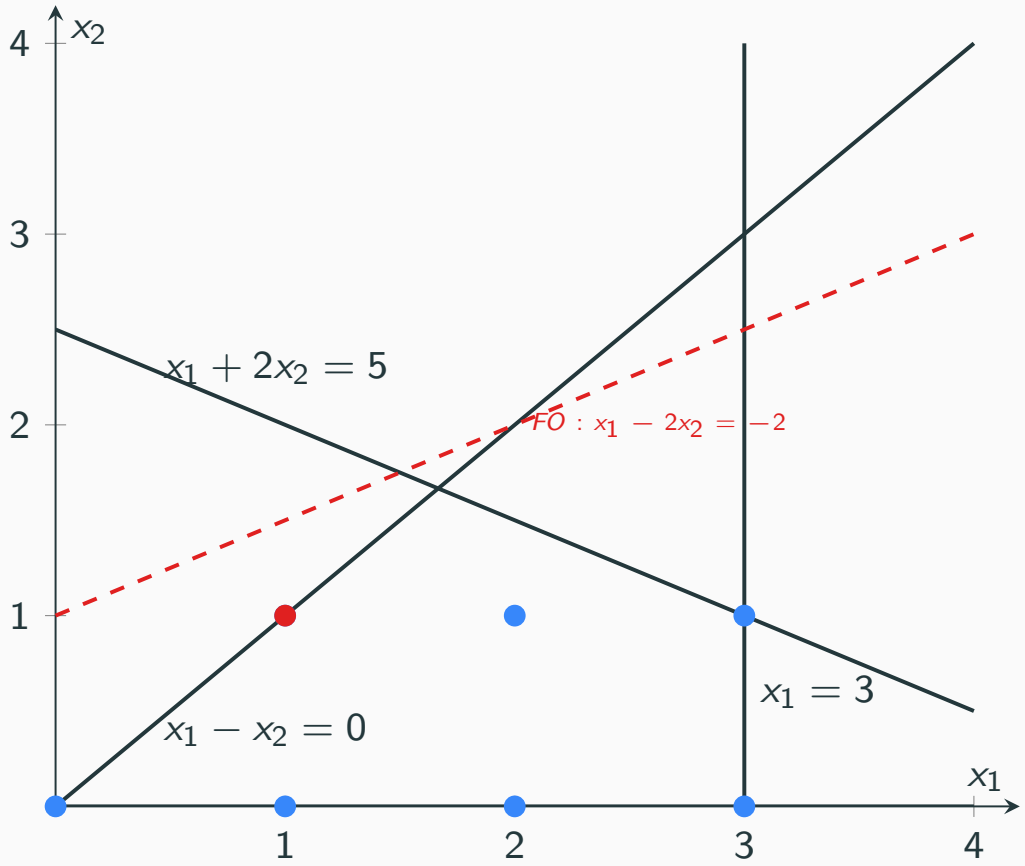
Se RP é uma relaxação de IP, então $z^R \leq z$.

Demonstração.

Se x^* é uma solução ótima de IP, então $x \in X$ e $z = c(x^*)$. Como RP é uma relaxação de IP sabemos que $X \subseteq T$ e $f(x^*) \leq c(x^*)$. Logo, $z^R \leq f(x^*) \leq c(x^*) \leq z$. □

Exemplo

$$(IP) \equiv \min z = x_1 - 2x_2$$
$$s.a: x_1 - x_2 \geq 0$$
$$x_1 + 2x_2 \leq 5$$
$$x_1 \leq 3$$
$$x_1, x_2 \geq 0 \text{ e inteiros}$$

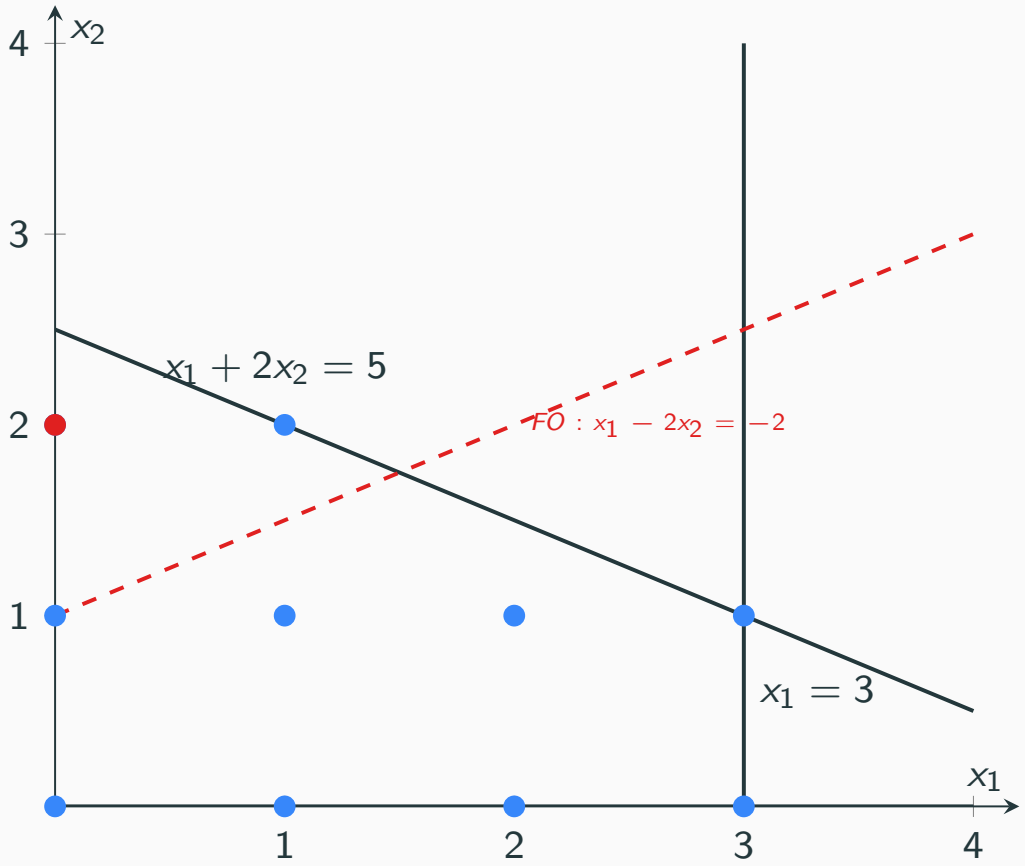


$$v(IP) = c(1,1) = 1 \times 1 - 2 \times 1 = -1$$

Exemplo

$$(RL_1) \equiv \min z = x_1 - 2x_2$$
$$s.a: x_1 + 2x_2 \leq 5$$
$$x_1 \leq 3$$
$$x_1, x_2 \geq 0 \text{ e inteiros}$$

Removida a restrição $x_1 - x_2 \geq 0$.



$$v(RL_1) = c(0,2) = 1 \times 0 - 2 \times 2 = -4$$

Conceitos base: relaxação

Propriedade

(i) Se a relaxação RP é impossível, então o problema original IP é impossível.

(ii) Seja x^* a SO da relaxação RP e $f(x) = c(x), \forall x \in X$. Se $x^* \in X$, então x^* é a SO do problema IP.

Demonstração.

(i) Se RP é impossível, então $T = \emptyset$. Assim, como $X \subseteq T$ temos $X = \emptyset$ e podemos concluir que o problema original IP também é impossível.

(ii) Como x^* a SO da relaxação RP e $x^* \in X$, temos $z \leq c(x^*) = f(x^*) = z^R$. Como RP é uma relaxação de IP temos $z^R \leq z$. Assim, $z = z^R$ e x^* é uma SO de IP, pois $z = c(x^*)$.

□

Conceitos base: relaxação linear

A relaxação linear de um PLIM é obtida removendo as restrições de integralidade.

Definição: Relaxação linear

Dado o problema

$$(IP) \qquad z = \min\{c(x) : x \in X \subseteq \mathbb{Z}^n\},$$

a sua *relaxação linear* é

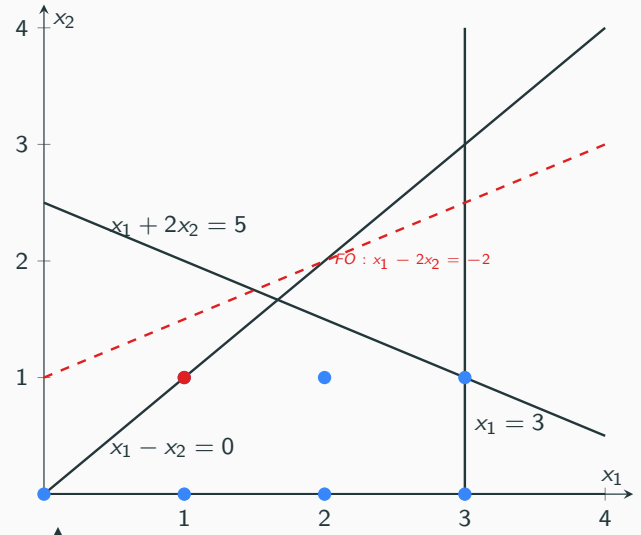
$$(PLR) \qquad z^{LR} = \min\{c(x) : x \in X \subseteq \mathbb{R}^n\}.$$

Para avaliar a qualidade do valor da relaxação linear podemos usar o gap.

$$gap = \frac{z - z^{LR}}{z}.$$

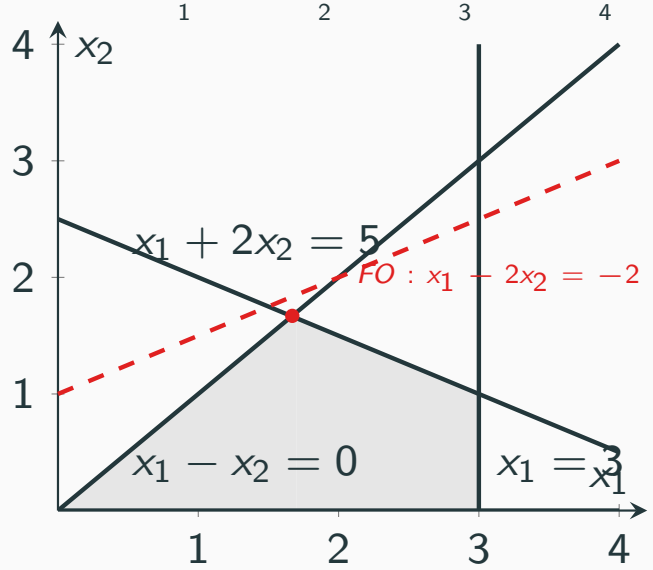
Exemplo

$$(IP) \equiv z = \min x_1 - 2x_2$$
$$s.a: x_1 - x_2 \geq 0$$
$$x_1 + 2x_2 \leq 5$$
$$x_1 \leq 3$$
$$x_1, x_2 \geq 0 \text{ e inteiros}$$



$$(RL_2) \equiv z = \min x_1 - 2x_2$$
$$s.a: x_1 - x_2 \geq 0$$
$$x_1 + 2x_2 \leq 5$$
$$x_1 \leq 3$$
$$x_1, x_2 \geq 0 \text{ e inteiros}$$

$$gap = \frac{-1-(-5/3)}{|-1|}^{1} = \frac{2}{3} \implies 66.67\%.$$



[1] Usamos o módulo no denominador quando o valor ótimo é negativo.

Algoritmos de enumeração

Seja

$$(PLI) \equiv z = \min\{c(x) : x \in S\}.$$

Ideia geral: decompor S em conjuntos "mais pequenos" e resolver o PLI nesses conjuntos.

Proposição

Seja $S = S_1 \cup S_2 \cup \ldots \cup S_{|K|}$ uma decomposição de S e seja $z^k = \min\{c(x) : x \in S_k\}, k \in K$. Então,

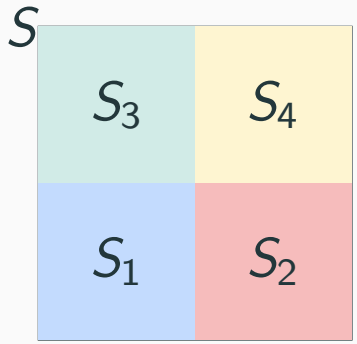
$$z = \min_{k \in K}\{z^k\}.$$

Algoritmos de enumeração

- Caso S_k não seja de resolução mais fácil pode ser decomposto, isto é,

$$S_k = S_{k_1} \cup S_{k_2} \cup \ldots.$$

- Este processo pode ser repetido até todos os subproblemas serem resolvidos. $\Longrightarrow$ Pesquisa em árvore.
- Como decompor S em subconjuntos $S_1, \ldots, S_k$?
  - Os conjuntos $S_1, \ldots, S_k$ devem ser uma partição de S.



  - A forma mais simples é particionar S em dois subconjuntos S_1 e S_2.

Algoritmos de enumeração

Variáveis binárias

Seja x_i uma variável binária, isto é, $x_i \in \{0, 1\}$.

O conjunto S pode ser particionado em:

- $S_1 = \{x \in S : x_i = 0\}$
- $S_2 = \{x \in S : x_i = 1\}$

Variáveis inteiras

Seja x_i uma variável com valor fracionário, isto é, $x_i = \alpha \notin \mathbb{Z}$.

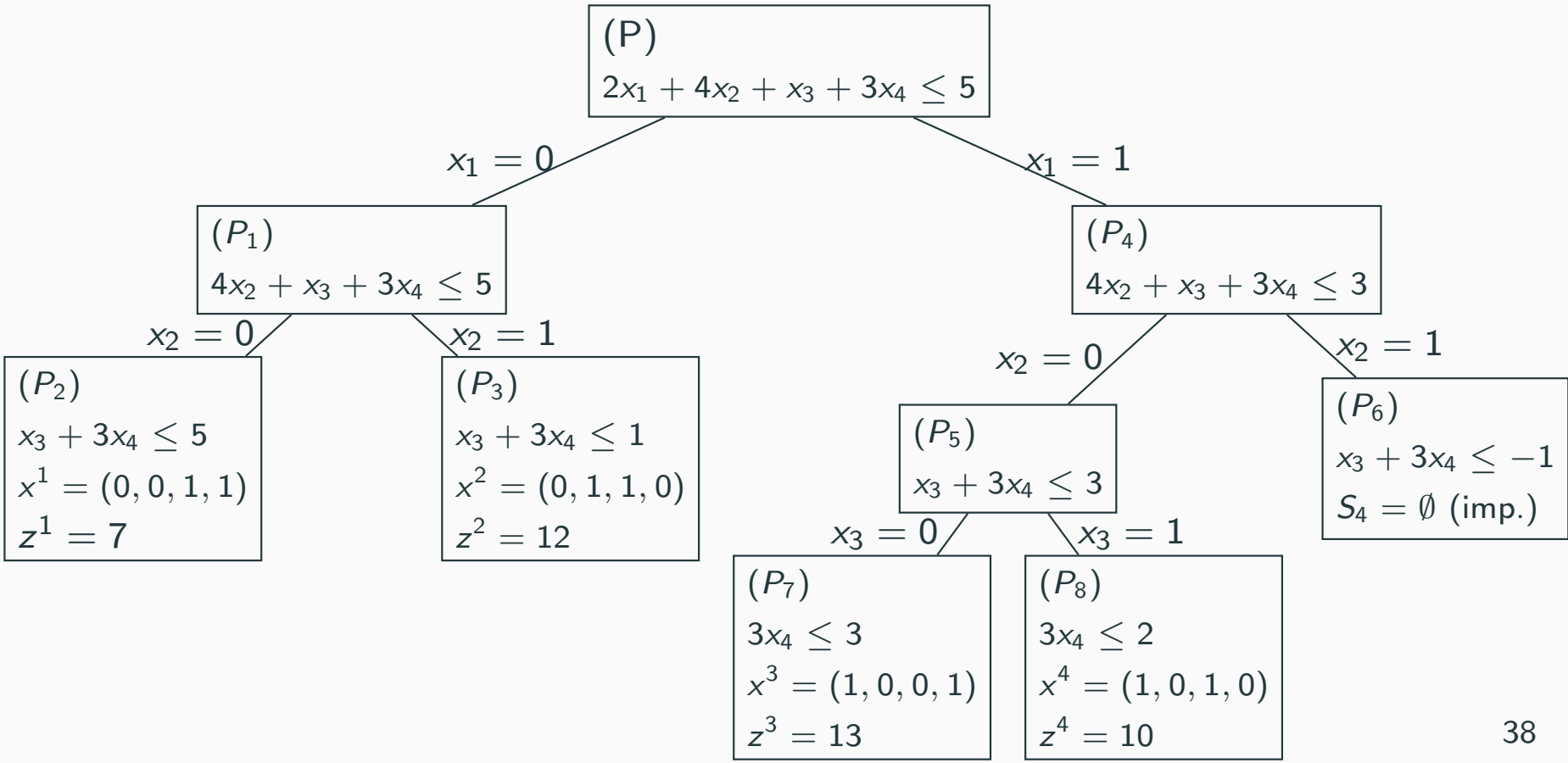
O conjunto S pode ser particionado em:

- $S_1 = \{x \in S : x_i \leq \lfloor \alpha \rfloor\}$
- $S_2 = \{x \in S : x_i \geq \lceil \alpha \rceil\}$

Podemos ir decompondo os conjuntos sucessivamente. $\Longrightarrow$ Árvore de enumeração.

Exemplo

$$(P) \equiv \max\ 8x_1 + 10x_2 + 2x_3 + 5x_4 \quad \text{Problema saco-mochila}$$
$$\text{s.a: } 2x_1 + 4x_2 + x_3 + 3x_4 \leq 5$$
$$x_1, x_2, x_3, x_4 \in \{0, 1\}$$



Algoritmos de enumeração

- A cada nodo k está associado um problema P_k e um conjunto S_k, que é a RA de P_k.
  - Se P_k é obtido de P_j através de uma ramificação diz-se que k é filho de j.
  - Se P_k é obtido de P_j através de uma sequência de ramificações diz-se que k é descendente de j.
  - Temos $v(P_j) \leq v(P_k)$, para todo o descendente k de j.
    - Relação válida para problemas de minimização.
  - Temos $v(P_j) \geq v(P_k)$, para todo o descendente k de j.
    - Relação válida para problemas de maximização.

Nota: Cada vez que ramificamos estamos a adicionar restrições e, consequentemente, a piorar o valor da solução ótima dos subproblemas.

Algoritmos de enumeração

- Como se relaciona z com os limites nos valores de z^k?

Proposição

Seja:

- $S = S_1 \cup S_2 \cup \ldots \cup S_K$ uma decomposição de S;
- $z^k = \min\{c(x) : x \in S_k\}, k = 1, \ldots, K$;
- $\overline{z}^k$ um limite superior para z^k; e
- $\underline{z}^k$ um limite inferior para z^k.

Então, $\overline{z} = \min_{k=1,\ldots,K}\{\overline{z}^k\}$ é um limite superior para z e $\underline{z} = \min_{k=1,\ldots,K}\{\underline{z}^k\}$ é um limite inferior para z.

Maximização: $\overline{z} = \max_{k=1,\ldots,K} \{\overline{z}^k\}$

$\underline{z} = \max_{k=1,\ldots,K} \{\underline{z}^k\}$

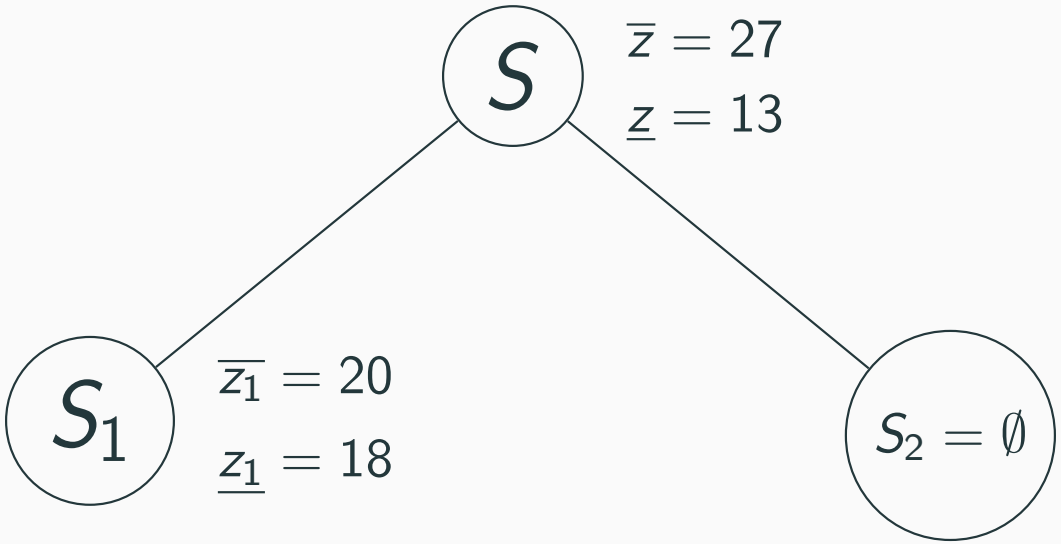
Algoritmos de enumeração

- Podemos usar a proposição anterior para reduzir o tamanho da árvore de enumeração.

- Um nodo k pode ser cancelado (não ramificado) em três situações:

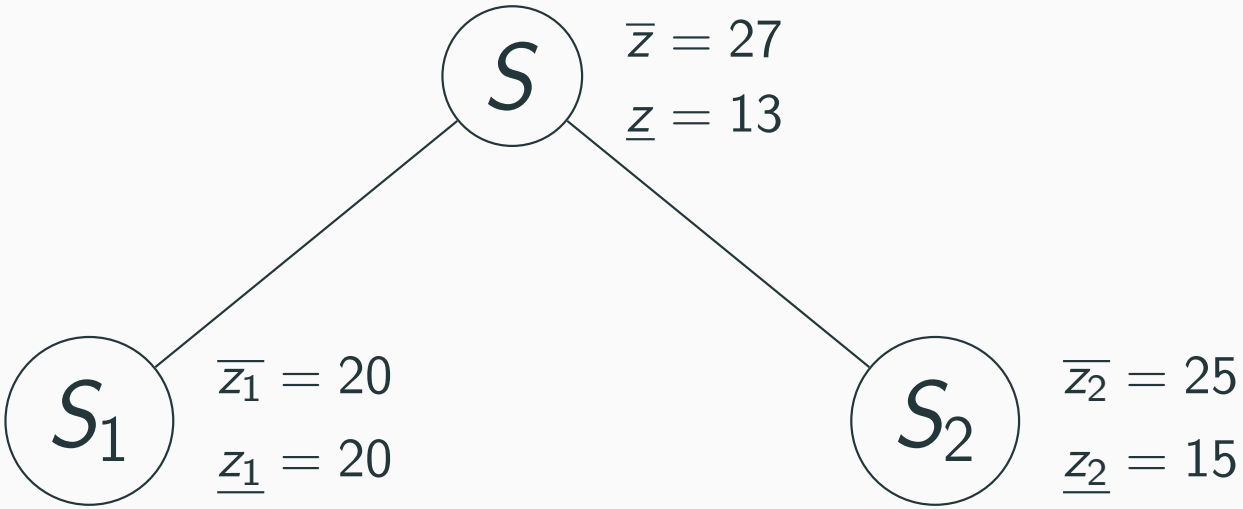
  1. o problema é impossível, $S_k = \emptyset$;
  2. a solução obtida é admissível para o problema original P; ou
  3. é possível provar que P_k não contém a solução ótima (ou soluções melhores que a melhor solução conhecida para P).

Problema de minimização



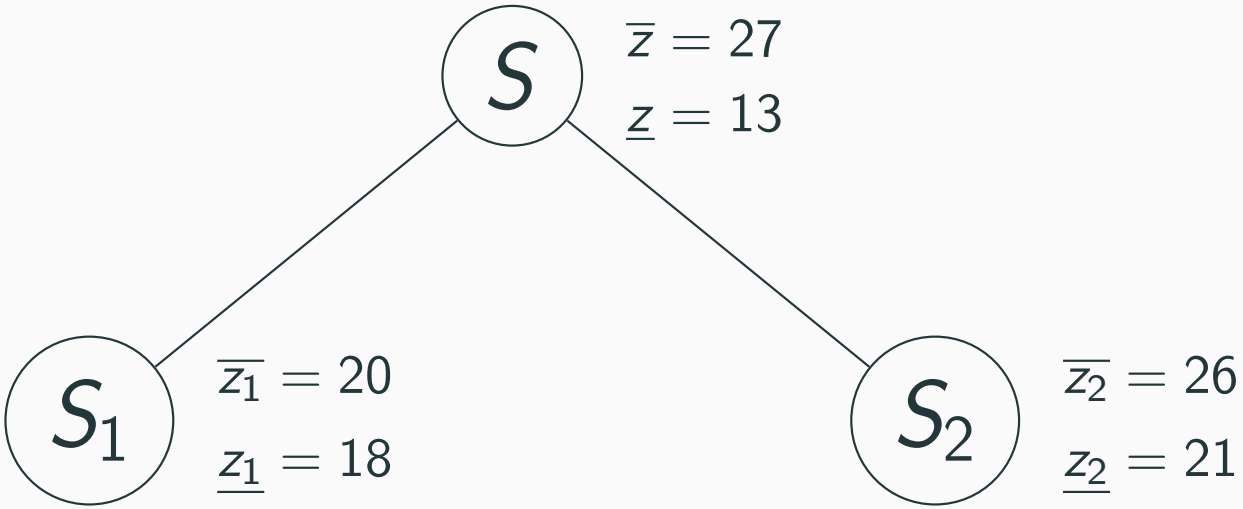
O problema P_2 é impossível, não é necessário ramificá-lo. $\Longrightarrow$ Corte por impossibilidade.

Problema de minimização



Como o limite superior e inferior de P_1 são iguais ($\overline{z}_1 = \underline{z}_1 = 20$) obtivemos o seu valor ótimo e por isso não é necessário ramificá-lo. $\Longrightarrow$ Corte por otimalidade.

Problema de minimização



O valor ótimo de P_2 é pelo menos 21, enquanto o valor ótimo de S_1 é no máximo 20. Assim, como a partir de P_2 vamos sempre obter soluções piores que as de P_1 podemos não explorá-lo mais. $\Longrightarrow$ Corte por limite.

Algoritmo de branch-and-bound

- É um algoritmo de enumeração!
- Como resolver os subproblemas?
  - É resolvida a sua relaxação linear.
- Notação:

$(P) \equiv z = \min\{c(x) : x \in X\}$

$(P_k) \equiv z_k = \min\{c(x) : x \in X_k\}$

L Lista de problemas por resolver

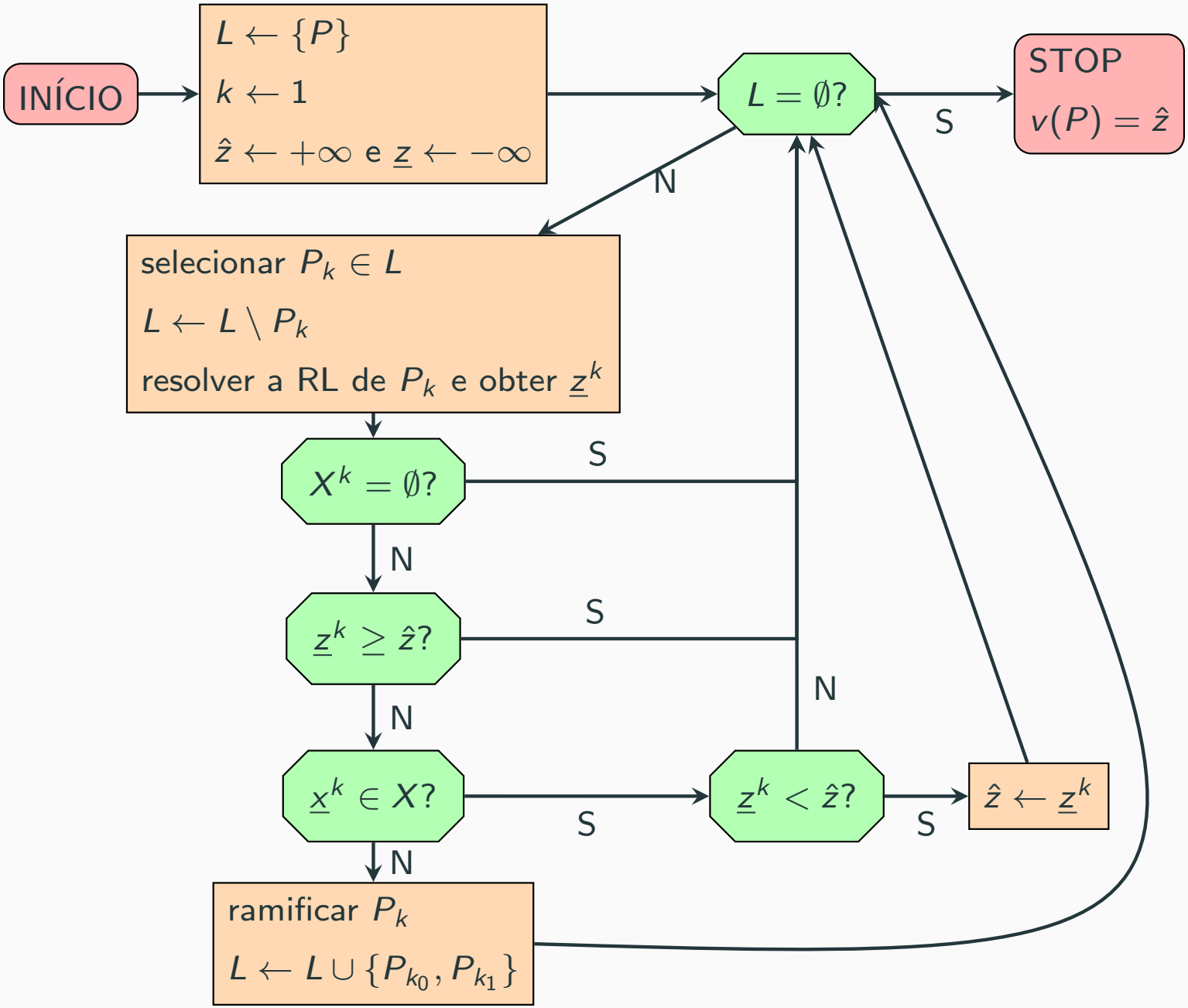
$\hat{z}$ valor da melhor SA conhecida - incumbente.

Caso não tenha sido determinada nenhuma SA, $\hat{z} = +\infty$.

$\underline{z}^k$ minorante de z^k.

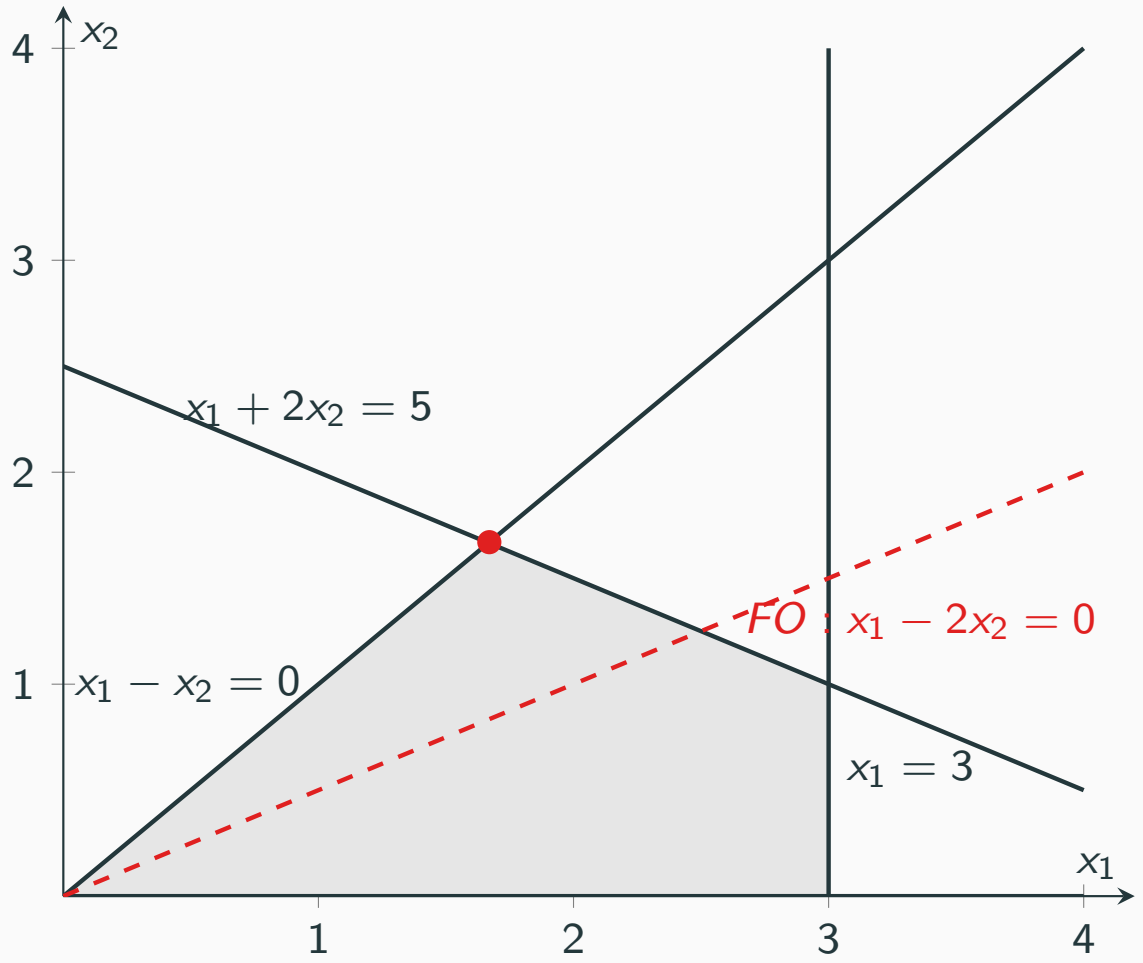
Temos $\underline{z}^k = -\infty$ se P_k impossível e $\underline{z}^k = z^k$ caso contrário.

Algoritmo de branch-and-bound



Exemplo

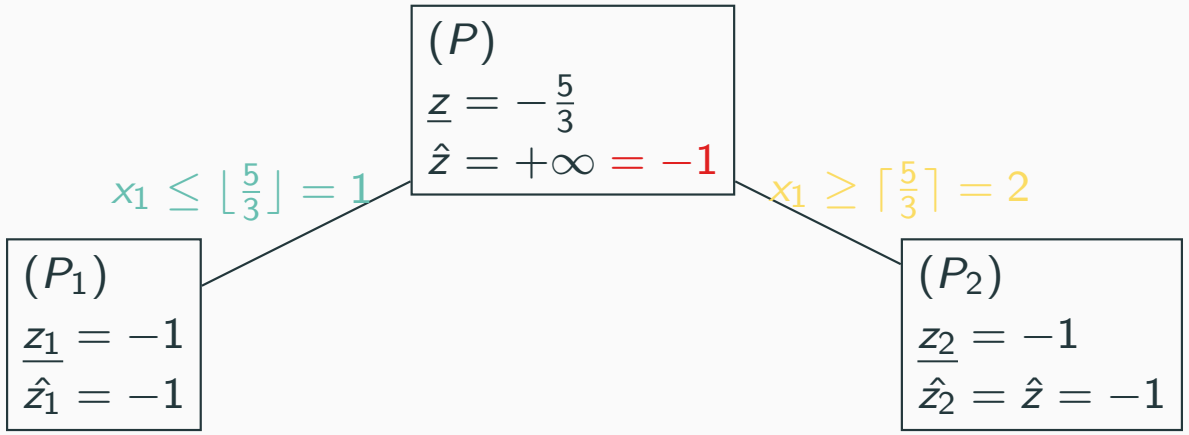
$$
\begin{aligned}
(P) \equiv \min z = & x_1 - 2x_2 \\
s.a: & x_1 - x_2 \geq 0 \\
& x_1 + 2x_2 \leq 5 \\
& x_1 \leq 3 \\
& x_1, x_2 \geq 0 \text{ e inteiros}
\end{aligned}
$$

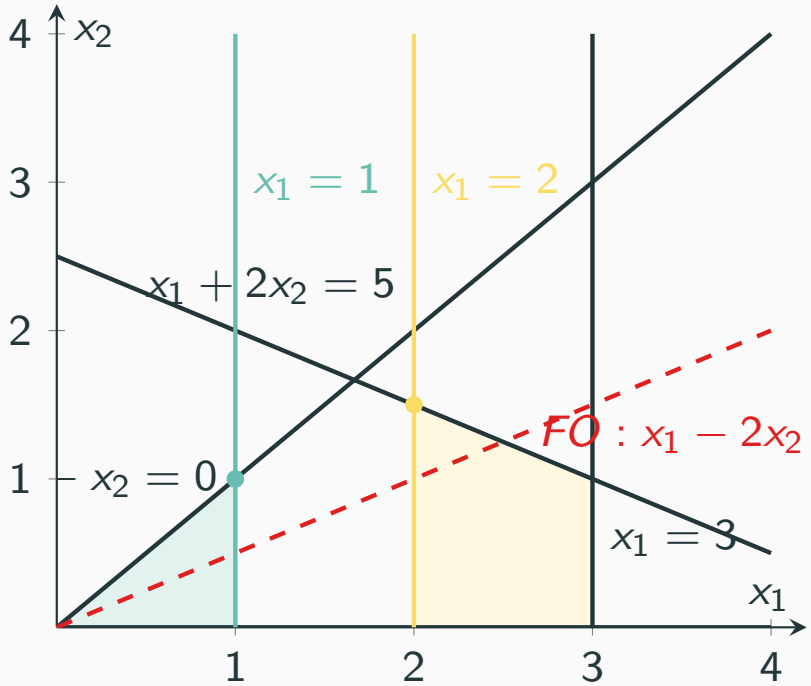


Resolvendo a relaxação linear:

$x_{RL} = (\frac{5}{3}, \frac{5}{3})$ e $\underline{z} = \frac{5}{3} - 2 \times \frac{5}{3} = -\frac{5}{3}$

Exemplo

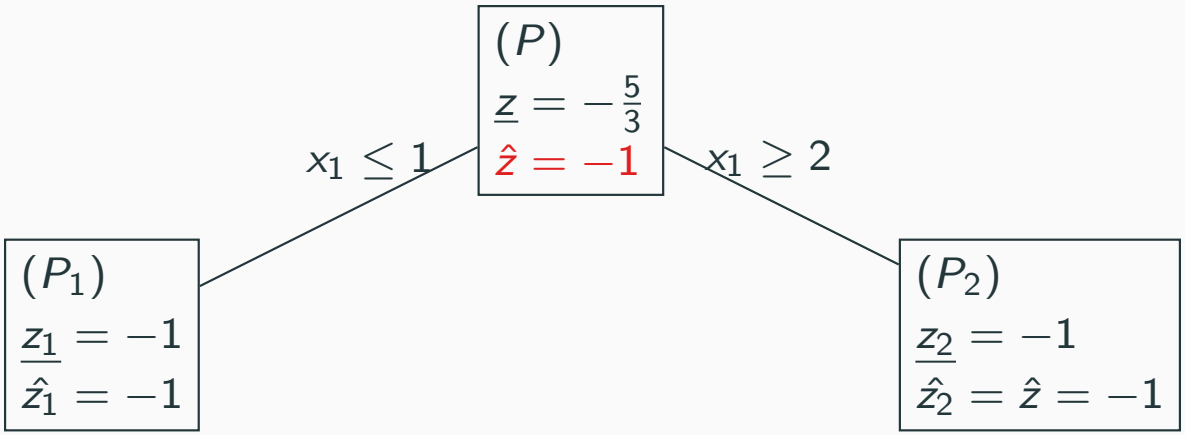




- $(P_1) : x_1^* = (1, 1)$ e $\underline{z_1} = \overline{z_1} = -1$
- $(P_2) : x_2^* = (2, \frac{3}{2})$ e $\underline{z_2} = -1$

■ Não é preciso ramificar mais a partir de P_1 pois já encontramos uma SA. Podemos atualizar o majorante.

Exemplo



Como o minorante de P_2 é igual a $\hat{z}$ podemos cancelar o nodo pois não vai conter a SO. Todas as soluções de descendentes de P_2 vão ter valor pior ou igual que -1.

Algoritmo de branch-and-bound

Estratégias de ramificação e pesquisa

- Como ramificar?
  - Escolher a variável fracionária com valor α mais próximo de $\lfloor \alpha \rfloor + \frac{1}{2}$ na relaxação linear (variável "mais fracionária").
  - Escolher a variável fracionária com o índice mais baixo.
- Como selecionar o subproblema?
  - Pesquisa em profundidade.
    Escolher um dos últimos problemas criados. $\Longrightarrow$ Descer rapidamente na árvore de pesquisa de modo a encontrar rapidamente uma solução admissível.
  - Seleção do melhor nodo.
    Escolher o problema em aberto com maior valor de $\underline{z}^k$.

Algoritmo de branch-and-bound

- Estratégias de redução do número de nodos da árvore
  - Utilizar heurísticas para determinar soluções admissíveis (majorantes).
  - Se os coeficientes da função objetivo são inteiros, z só toma valores inteiros para soluções admissíveis. Neste caso os minorantes $\underline{z}^k$ podem ser substituídos por $\lceil \underline{z}^k \rceil$.
  - Tentar melhorar a qualidade dos minorantes.

■ O algoritmo de branch-and-bound pode ser utilizado como uma heurística.

$\Longrightarrow$ Por exemplo, parando o algoritmo quando uma SA for encontrada.

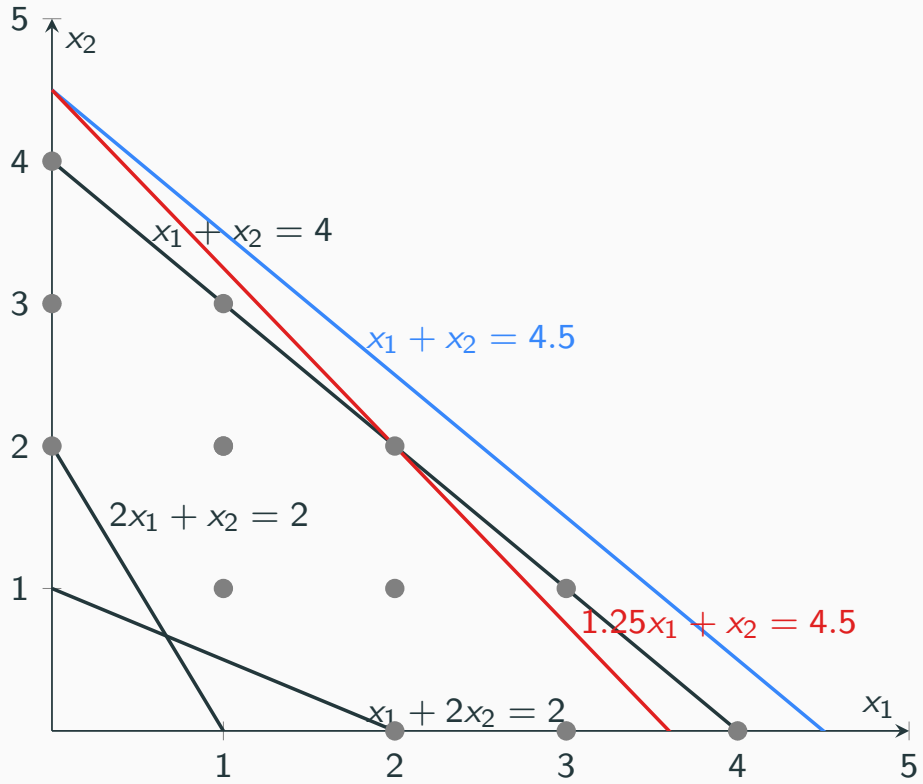
Algoritmo de planos de corte de Gomory

Definição

Seja $X \subseteq \mathbb{R}^n$. Uma *desigualdade válida* para X é uma desigualdade $\pi_1 x_1 + \ldots + \pi_n x_n \leq \pi_0$ ($\pi x \leq \pi_0$) que é satisfeita por todos os pontos de X.

Exemplo

Seja $X = \{(x_1, x_2) \in \mathbb{Z}^2 : x_1 + x_2 \leq 4, x_1 + 2x_2 \geq 2, 2x_1 + x_2 \geq 2 \text{ e } x_1, x_2 \geq 0\}$. $\Longrightarrow$ Pontos cinzentos no gráfico.



A desigualdade $x_1 + x_2 \leq 4.5$ é uma desigualdade válida para X.

A desigualdade $1.25x_1 + x_2 \leq 4.5$ não é uma desigualdade válida para X. $\Longrightarrow$ Os pontos $(3, 1)$ e $(4, 0)$ pertencem a X e não satisfazem a desigualdade.

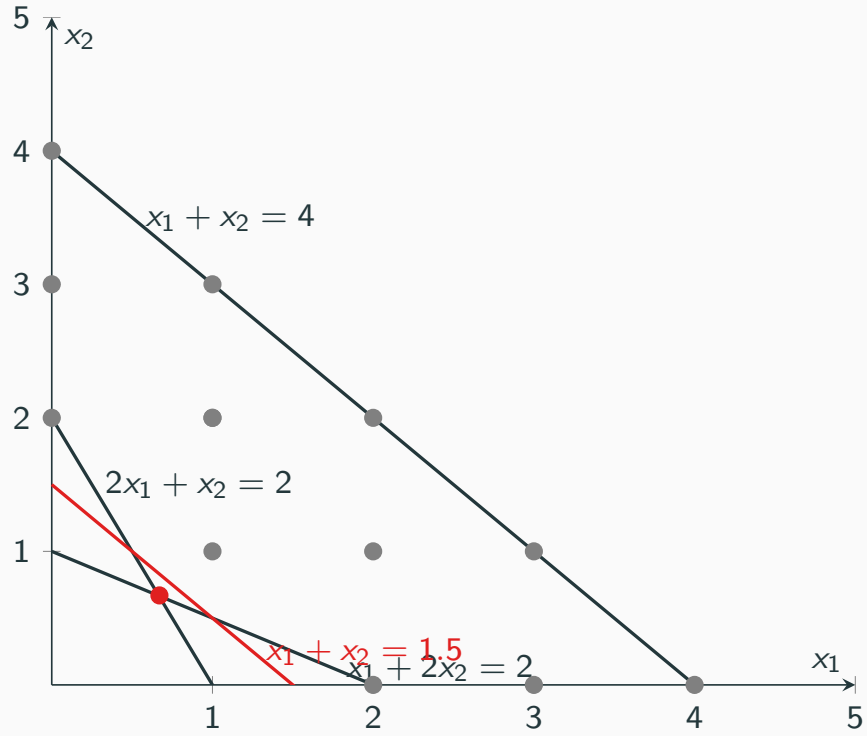
Algoritmo de planos de corte de Gomory

Definição

Sejam $X \subseteq \mathbb{R}^n$ e $x^* \notin X$. Uma desigualdade válida para X não satisfeita por x^*, diz-se um *plano de corte*.

Exemplo

Seja $X = \{(x_1, x_2) \in \mathbb{Z}^2 : x_1 + x_2 \leq 4, x_1 + 2x_2 \geq 2, 2x_1 + x_2 \geq 2 \text{ e } x_1, x_2 \geq 0\}$. $\Longrightarrow$
Pontos cinzentos no gráfico.



A desigualdade $x_1 + x_2 \geq 1.5$ é um plano de corte para $x^* = (\frac{2}{3}, \frac{2}{3})$, que é a SO da relaxação linear do problema

$$\min\{3x_1 + 2x_2 : (x_1, x_2) \in X\}.$$

- Esta é a ideia geral do algoritmo de planos de corte de Gomory.

Algoritmo de planos de corte de Gomory

Seja

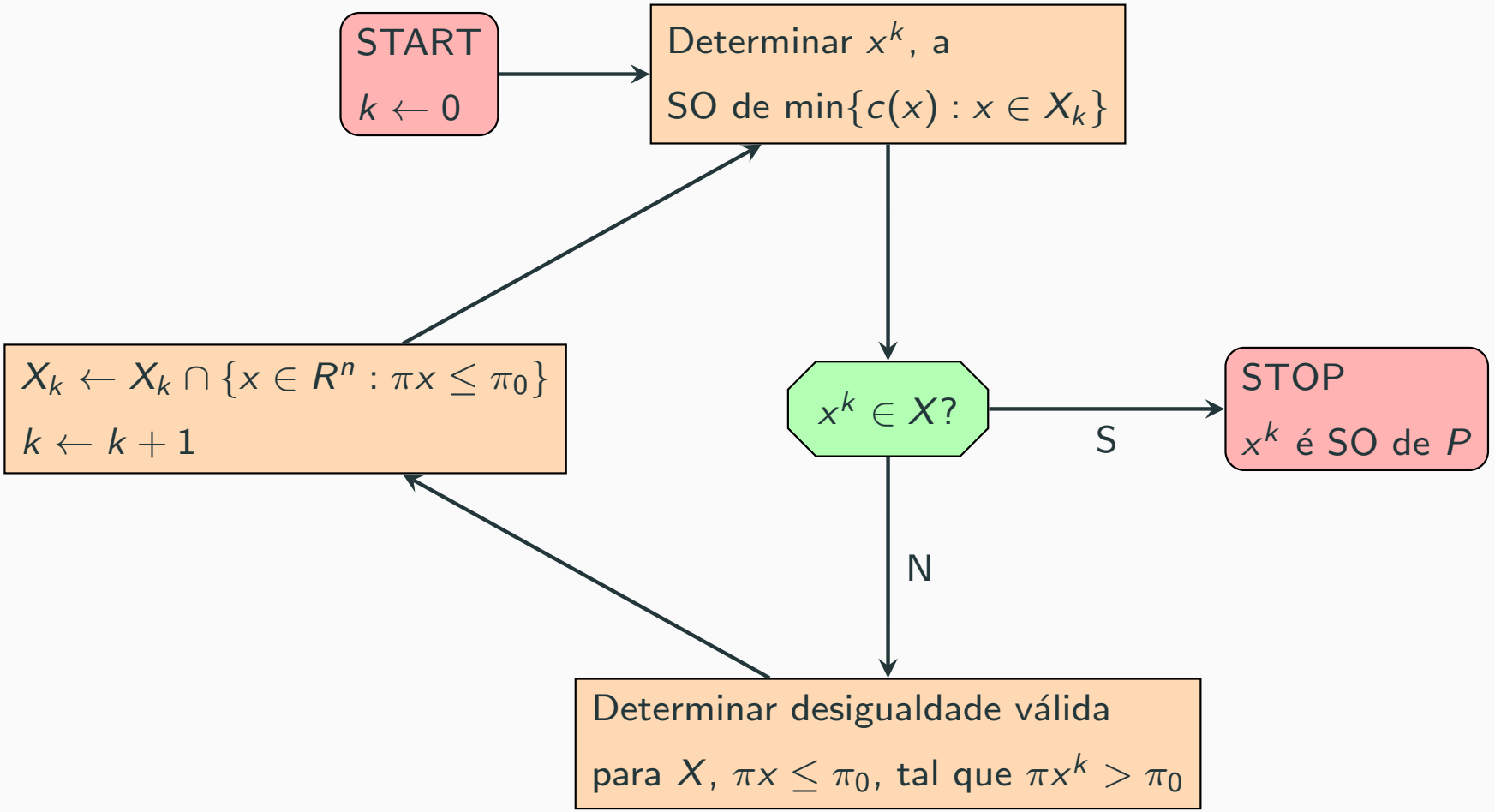
$$(P) \equiv \min\{c(x) : x \in X\},$$

com $X = \{x \in R^n : Ax \geq b$ e x_j inteiro, para $j \in J\}$ sendo J um subconjunto de $\{1, ..., n\}$.

Ideia geral: resolver uma sucessão de relaxações lineares até à solução obtida estar em X. Cada relaxação linear é obtida da anterior acrescentando-lhe planos de corte.

Algoritmo de planos de corte de Gomory

Seja $\overline{X}_0 = \{x \in R^n : Ax \geq b\}$ (X sem as restrições de integralidade).



Algoritmo de planos de corte de Gomory

Como determinar planos de corte para a SO da relaxação linear?

- Procedimento de geração de cortes de Gomory, que é válido para qualquer PLIM

Procedimento de geração de cortes de Gomory

Seja

$$(P) \equiv z = \min\{c(x) : Ax \geq b,\ x \geq 0 \text{ e inteiros}\}.$$

Passo 1: Resolver a relaxação linear de P.

Passo 2: Escrever o sistema $Ax \geq b$ na forma aumentada.

Passo 3: Rescrever o sistema em função das VNB da SO, isto é, $x_i + \sum_{j \in VNB} \bar{a}_{ij} x_j = \bar{b}_i$ para toda a VB x_i.

Passo 4: O corte de Gomory é obtido através do seguinte procedimento:

$$x_i + \sum_{j \in VNB} \bar{a}_{ij} x_j = \bar{b}_i \implies x_i + \sum_{j \in VNB} \lfloor \bar{a}_{ij} \rfloor x_j \leq \bar{b}_i \implies x_i + \sum_{j \in VNB} \lfloor \bar{a}_{ij} \rfloor x_j \leq \lfloor \bar{b}_i \rfloor$$

$$\implies \sum_{j \in VNB} (\bar{a}_{ij} - \lfloor \bar{a}_{ij} \rfloor) x_j \geq \bar{b}_i - \lfloor \bar{b}_i \rfloor$$

Algoritmo de planos de corte de Gomory

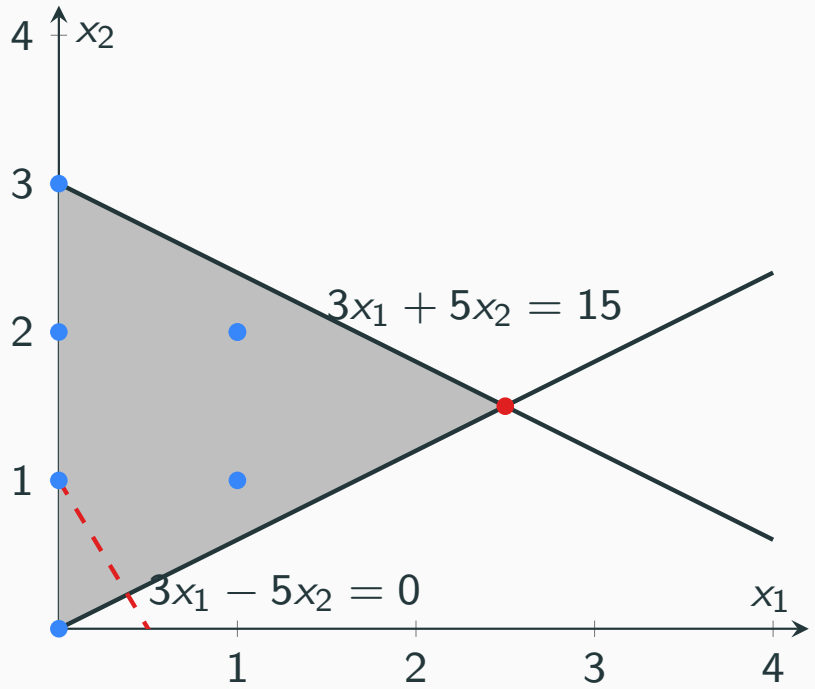
Definição

O *corte de Gomory* é $\sum_{j \in VNB} (\bar{a}_{ij} - \lfloor \bar{a}_{ij} \rfloor) x_j \geq \bar{b}_i - \lfloor \bar{b}_i \rfloor$.

Exemplo

$$
\begin{aligned}
(P) \equiv \max z = {} & 2x_1 + x_2 \\
\text{s.a: } & 3x_1 - 5x_2 \leq 0 \\
& 3x_1 + 5x_2 \leq 15 \\
& x_1, x_2 \geq 0 \text{ e inteiros}
\end{aligned}
$$

- Resolvendo a relaxação linear de P obtém-se $x^0 = (\frac{5}{2}, \frac{3}{2})$.



Exemplo

- Escrevendo o sistema na forma aumentada obtemos:

$$\begin{cases} 3x_1 - 5x_2 + x_3 = 0 \\ 3x_1 + 5x_2 + x_4 = 15 \end{cases}$$

- A solução na forma aumentada é $x^0 = (\frac{5}{2}, \frac{3}{2}, 0, 0)$ e tem como VBs x_1 e x_2.

Exemplo

- Reescrevendo o sistema em função das VB obtemos:

$$\begin{cases} 3x_1 - 5x_2 + x_3 = 0 \\ 3x_1 + 5x_2 + x_4 = 15 \end{cases} \iff \begin{cases} x_1 = \frac{5}{3}x_2 - \frac{1}{3}x_3 \\ 3(\frac{5}{3}x_2 - \frac{1}{3}x_3) + 5x_2 + x_4 = 15 \end{cases}$$

$$\iff \begin{cases} - \\ 5x_2 - x_3 + 5x_2 + x_4 = 15 \end{cases} \iff \begin{cases} x_1 = \frac{5}{3}(\frac{15}{10} + \frac{1}{10}x_3 - \frac{1}{10}x_4) - \frac{1}{3}x_3 \\ x_2 - \frac{1}{10}x_3 + \frac{1}{10}x_4 = \frac{15}{10} \end{cases}$$

$$\iff \begin{cases} x_1 = \frac{5}{2} - \frac{1}{6}x_3 - \frac{1}{6}x_4 \\ x_2 - \frac{1}{10}x_3 + \frac{1}{10}x_4 = \frac{15}{10} \end{cases} \iff \begin{cases} x_1 + \frac{1}{6}x_3 + \frac{1}{6}x_4 = \frac{5}{2} \\ x_2 - \frac{1}{10}x_3 + \frac{1}{10}x_4 = \frac{3}{2} \end{cases}$$

Como ambos os $\bar{b}_i$ são fracionários podemos escolher qualquer restrição para gerar um corte de Gomory. $\Longrightarrow$ Vamos escolher a linha do x_1.

$$(\frac{1}{6} - 0)x_3 + (\frac{1}{6} - 0)x_4 \geq (\frac{5}{2} - 2) \iff \frac{1}{6}x_3 + \frac{1}{6}x_4 \geq \frac{1}{2}$$

Exemplo

- Temos que escrever o corte $\frac{1}{6}x_3 + \frac{1}{6}x_4 \geq \frac{1}{2}$ em função de x_1 e x_2 para o conseguirmos representar no gráfico.
- Da forma aumentada de P tiramos que:

$$\begin{cases} 3x_1 - 5x_2 + x_3 = 0 \iff x_3 = -3x_1 + 5x_2 \\ 3x_1 + 5x_2 + x_4 = 15 \iff x_4 = 15 - 3x_1 - 5x_2 \end{cases}$$

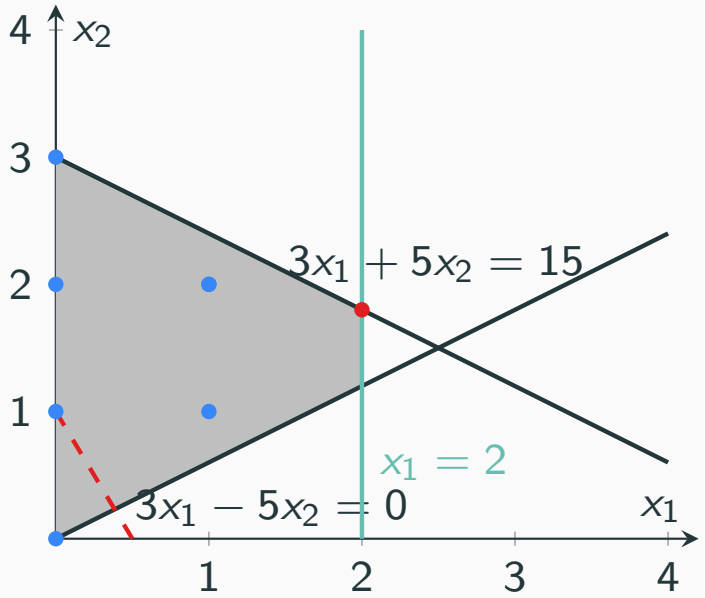
- Logo,

$$\frac{1}{6}x_3 + \frac{1}{6}x_4 \geq \frac{1}{2} \iff \frac{1}{6}(-3x_1 + 5x_2) + \frac{1}{6}(15 - 3x_1 - 5x_2) \geq \frac{1}{2}$$
$$\iff -\frac{3}{6}x_1 + \frac{5}{6}x_2 + \frac{15}{6} - \frac{3}{6}x_1 - \frac{5}{6}x_2 \geq \frac{1}{2} \iff -x_1 \geq -\frac{12}{6} \iff x_1 \leq 2$$

Exemplo

$$
\begin{aligned}
(P^1) \equiv \max z &= 2x_1 + x_2 \\
\text{s.a: } & 3x_1 - 5x_2 \leq 0 \\
& 3x_1 + 5x_2 \leq 15 \\
& x1 \leq 2 \\
& x_1, x_2 \geq 0 \text{ e inteiros}
\end{aligned}
$$

- Resolvendo a relaxação linear de P^1 obtém-se $x^1 = (2, \frac{9}{5})$.



Exemplo

- Escrevendo o sistema na forma aumentada obtemos:

$$\begin{cases} 3x_1 - 5x_2 + x_3 = 0 \\ 3x_1 + 5x_2 + x_4 = 15 \\ x_1 + x_5 = 2 \end{cases}$$

- A solução na forma aumentada é $x^1 = (2, \frac{9}{5}, 3, 0, 0)$ e tem como VBs x_1, x_2 e x_3.

Exemplo

- Reescrevendo o sistema em função das VB obtemos:

$$\begin{cases} 3x_1 - 5x_2 + x_3 = 0 \\ 3x_1 + 5x_2 + x_4 = 15 \\ x_1 + x_5 = 2 \end{cases} \iff \begin{cases} - \\ x_2 = 3 - \frac{3}{5}x_1 - \frac{1}{5}x_4 \\ x_1 = 2 - x_5 \end{cases} \iff \begin{cases} - \\ x_2 = \frac{9}{5} + \frac{3}{5}x_5 - \frac{1}{5}x_4 \\ - \end{cases}$$

$$\iff \begin{cases} 3(2 - x_5) - 5(\frac{9}{5} + \frac{3}{5}x_5 - \frac{1}{5}x_4) + x_3 = 0 \\ - \\ - \end{cases} \iff \begin{cases} x_3 + x_4 - 6x_5 = 3 \\ x_2 + \frac{1}{5}x_4 - \frac{3}{5}x_5 = \frac{9}{5} \\ x_1 + x_5 = 2 \end{cases}$$

Vamos escolher a restrição associada a x_2 para gerar um corte de Gomory porque é a única com um $\bar{b}_i$ fracionário.

$$(\frac{1}{5} - 0)x_4 + (-\frac{3}{5} - (-1))x_5 \geq (\frac{9}{5} - 1) \iff \frac{1}{5}x_4 + \frac{2}{5}x_5 \geq \frac{4}{5}$$

Exemplo

- Temos que escrever o corte $\frac{1}{5}x_4 + \frac{2}{5}x_5 \geq \frac{4}{5}$ em função de x_1 e x_2 para o conseguirmos representar no gráfico.
- Da forma aumentada de P^1 tiramos que:

$$\begin{cases} x_1 + 5x_2 + x_4 = 15 \iff x_4 = 15 - 3x_1 - 5x_2 \\ x_1 + x_5 = 2 \iff x_5 = 2 - x_1 \end{cases}$$

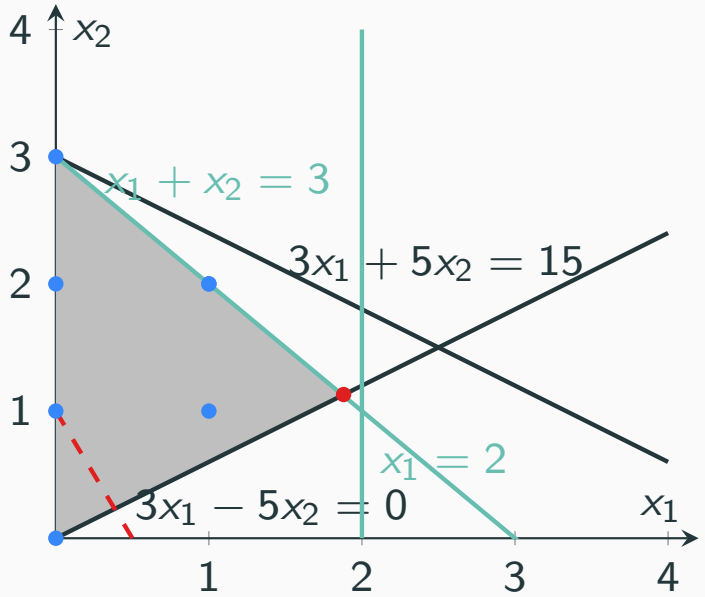
- Logo,

$$\frac{1}{5}x_4 + \frac{2}{5}x_5 \geq \frac{4}{5} \iff \frac{1}{5}(15 - 3x_1 - 5x_2) + \frac{2}{5}(2 - x_1) \geq \frac{4}{5}$$
$$\iff x_1 + x_2 \leq 3$$

Exemplo

$$
\begin{aligned}
(P^2) \equiv \max z = {} & 2x_1 + x_2 \\
\text{s.a: } & 3x_1 - 5x_2 \leq 0 \\
& 3x_1 + 5x_2 \leq 15 \\
& x1 \leq 2 \\
& x_1 + x_2 \leq 3 \\
& x_1, x_2 \geq 0 \text{ e inteiros}
\end{aligned}
$$

- Resolvendo a relaxação linear de P^1 obtém-se $x^2 = (\frac{15}{8}, \frac{9}{8})$.



Exemplo

- Escrevendo o sistema na forma aumentada obtemos:

$$\begin{cases} 3x_1 - 5x_2 + x_3 = 0 \\ 3x_1 + 5x_2 + x_4 = 15 \\ x_1 + x_5 = 2 \\ x_1 + x_2 + x_6 = 3 \end{cases}$$

- A solução na forma aumentada é $x^2 = (\frac{15}{8}, \frac{9}{8}, 0, \frac{30}{8}, \frac{1}{8}, 0)$ e tem como VBs x_1, x_2, x_4 e x_5.

Exemplo

- Reescrevendo o sistema em função das VB obtemos:

$$\begin{cases} 3x_1 - 5x_2 + x_3 = 0 \\ 3x_1 + 5x_2 + x_4 = 15 \\ x_1 + x_5 = 2 \\ x_1 + x_2 + x_6 = 3 \end{cases} \iff \ldots \iff \begin{cases} x_1 + \frac{1}{8}x_3 + \frac{5}{8}x_6 = \frac{15}{8} \\ \frac{2}{8}x_3 + x_4 - \frac{30}{8}x_6 = \frac{30}{8} \\ -\frac{1}{8}x_3 + x_5 - \frac{5}{8}x_6 = \frac{1}{8} \\ x_2 - \frac{1}{8}x_3 + \frac{3}{8}x_6 = \frac{9}{8} \end{cases}$$

Como todos os $\bar{b}_i$ são fracionários podemos escolher qualquer restrição para gerar um corte de Gomory. $\Longrightarrow$ Vamos escolher a linha do x_1.

$$(\frac{1}{8} - 0)x_3 + (\frac{5}{8} - 0)x_6 \geq (\frac{15}{8} - 1) \iff \frac{1}{8}x_3 + \frac{5}{8}x_6 \geq \frac{7}{8}$$

Exemplo

- Temos que escrever o corte $\frac{1}{8}x_3 + \frac{5}{8}x_6 \geq \frac{7}{8}$ em função de x_1 e x_2 para o conseguirmos representar no gráfico.
- Da forma aumentada de P^2 tiramos que:

$$\begin{cases} 3x_1 - 5x_2 + x_3 = 0 \iff x_3 = -3x_1 + 5x_2 \\ x_1 + +x_2 + x_6 = 3 \iff x_6 = 3 - x_1 - x_2 \end{cases}$$

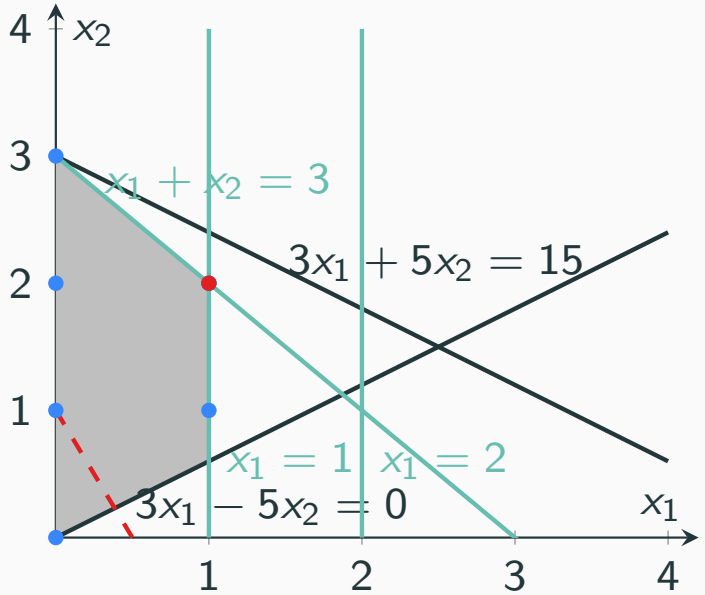
- Logo,

$$\frac{1}{8}x_3 + \frac{5}{8}x_6 \geq \frac{7}{8} \iff \frac{1}{8}(-3x_1 + 5x_2) + \frac{5}{8}(3 - x_1 - x_2) \geq \frac{7}{8}$$
$$\iff x_1 \leq 1$$

Exemplo

$$
\begin{aligned}
(P^3) \equiv \max z = {} & 2x_1 + x_2 \\
\text{s.a: } & 3x_1 - 5x_2 \leq 0 \\
& 3x_1 + 5x_2 \leq 15 \\
& x1 \leq 2,\ x_1 + x_2 \leq 3 \\
& x_1 \leq 1 \\
& x_1, x_2 \geq 0 \text{ e inteiros}
\end{aligned}
$$

- Resolvendo a relaxação linear de P^3 obtém-se $x^3 = (1, 2)$. Como x^3 é inteira é a SO de P. Logo, $z = 2 \times 1 + 2 = 4$. $\Longrightarrow$ *FIM.*



Sumário

Motivação

Introdução

Formulações

Algoritmos para programação inteira

Geração de desigualdades válidas

Desigualdades válidas

Def.: O invólucro convexo de um conj. X é o menor conj. q contém todos os pontos de X.

- Uma forma de tornar os algoritmos apresentados mais eficientes é "aproximando" a RA do PLR com o invólucro convexo do problema original.
  $\Longrightarrow$ Feito através da adição de desigualdades válidas, que cortem soluções fracionárias.

- Existem várias formas de obter desigualdades válidas para um problema.
  - Algoritmos de geração de desigualdades válidas. $\Longrightarrow$ Algoritmo de Chavátal-Gomory.
  - Desigualdades válidas baseadas nas especificidades do problema. $\Longrightarrow$ Famílias de desigualdades válidas.
    - Exemplo: Desigualdades de cobertura para o problema do saco-mochila.

Algoritmo de Chvátal-Gomory

Procedimento de Chvátal-Gomory para construir desigualdades válidas

Seja $X = P \cap \mathbb{Z}^n$, onde $P = \{x \in \mathbb{R}^n : Ax \leq b, x \geq 0\}$, A é uma matriz $m \times n$ com colunas $\{a_1, \ldots, a_n\}$ e $u \in \mathbb{R}^m_+$:

(i) A desigualdade

$$\sum_{j=1}^{n} ua_j x_j \leq ub$$

é válida para X ($u \geq 0$ e $\sum_{j=1}^{n} a_j x_j \leq b$).

(ii) A desigualdade

$$\sum_{j=1}^{n} \lfloor ua_j \rfloor x_j \leq ub$$

é válida para X ($u \geq 0$).

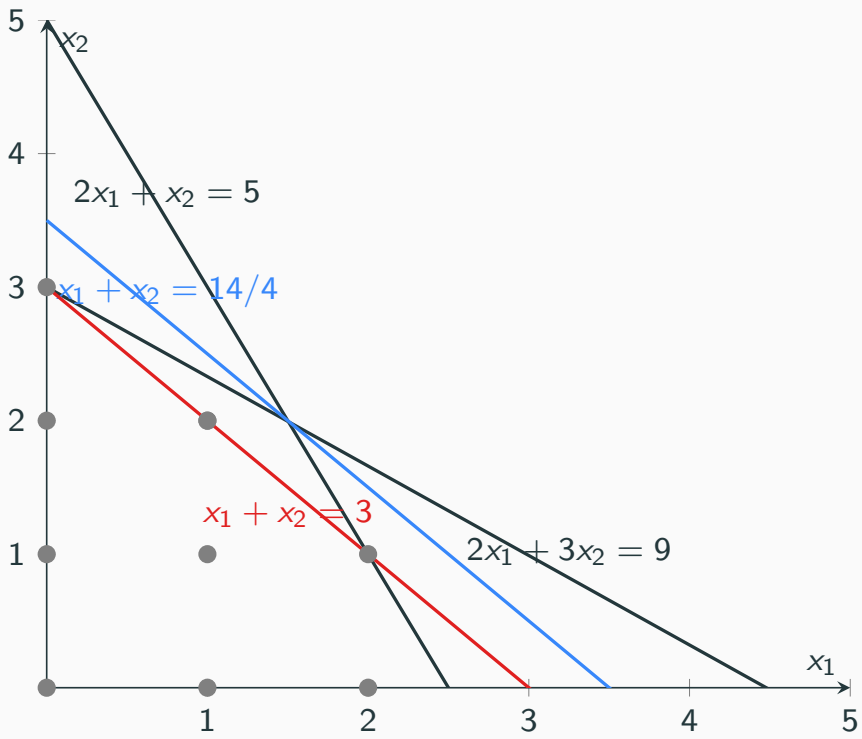
(iii) A desigualdade

$$\sum_{j=1}^{n} \lfloor ua_j \rfloor x_j \leq \lfloor ub \rfloor$$

é válida para X (x é inteiro e, consequentemente, $\sum_{j=1}^{n} \lfloor ua_j \rfloor x_j$ é inteiro).

Exemplo

Seja $X = \{(x_1, x_2) \in \mathbb{Z}^2 : 2x_1 + 3x_2 \leq 9, 2x_1 + x_2 \leq 5 \text{ e } x_1, x_2 \geq 0\}$.



Considere-se o seguinte vetor de coeficientes $u = (\frac{1}{4}, \frac{1}{4})$.

$$\begin{array}{lll} (1) & 2x_1 + 3x_2 \leq 9 & (\times \frac{1}{4}) \\ (2) & 2x_1 + x_2 \leq 5 & (\times \frac{1}{4}) \quad + \\ \hline & x_1 + x_2 \leq \frac{14}{4} & \end{array}$$

Aplicando o procedimento de Chvátal-Gomory obtemos $x_1 + x_2 \leq \lfloor \frac{14}{4} \rfloor \iff x_1 + x_2 \leq 3$.

Desigualdades válidas

Teorema

Qualquer desigualdade válida para X pode ser obtida aplicando sucessivamente o procedimento de Chvátal-Gomory (começando com o sistema $Ax \leq b$, $x \geq 0$ e acrescentando-lhe desigualdades que vão sendo obtidas).

Desigualdades baseadas nas especificidades do problema

Considere-se o caso geral do problema do saco-mochila

$$\max\{c(x) : x \in X\},$$

com

$$X = \{x \in \{0,1\}^n : \sum_{i=1}^{n} a_i x_i \leq b\}.$$

Definição

Uma *cobertura* é um subconjunto de artigos $S \subseteq N$ que não cabem todos na mochila, isto é,

$$\sum_{i \in S} a_i > b.$$

Definição

Sendo S uma cobertura, uma desigualdade válida - *desigualdade cobertura* - para o problema do saco-mochila é

$$\sum_{i \in S} x_i \leq |S| - 1.$$

Exemplo

$$\begin{aligned} \max z &= 32x_1 + 20x_2 + 19x_3 + 13x_4 + 11x_5 + 8x_6 \\ \text{s.a: } & 12x_1 + 9x_2 + 7x_3 + 5x_4 + 5x_5 + 3x_6 \leq 14 \\ & x_i \in \{0,1\},\ i = 1,\ldots,6 \end{aligned}$$

- Resolvendo (Solver) a relaxação linear do problema apresentado obtemos $z^{PLR} = 37.(6)$. $\Longrightarrow$ Quando temos variáveis binárias é necessário adicionar as restrições $x_i \leq 1$ à relaxação linear.

- Desigualdades de cobertura para a instância apresentada são, por exemplo:

1. $x_1 + x_2 \leq 1$
2. $x_1 + x_3 \leq 1$
3. $x_1 + x_6 \leq 1$
4. $x_2 + x_3 + x_4 \leq 2$
5. $x_2 + x_3 \leq 1$

Adicionando estas desigualdades válidas à relaxação linear obtemos $z^{PLR^+} = 37.4$.

Notas sobre programação linear inteira

- Muitas aplicações práticas requerem o uso de variáveis inteiras na sua modelação.
- Algoritmos para resolver problemas de programação linear inteira:
  - Branch-and-bound: Decompõe o problema em subproblemas mais fáceis de resolver.
  - Cortes de Gomory: Gera cortes para cortar a solução ótima da relaxação linear.
- A adição de desigualdades válidas permite-nos aproximar o valor da relaxação linear do valor ótimo do problema original.

Referências bibliográficas



Hillier, F. S. & Lieberman, G. J. (2021). *Introduction to Operations Research* Mc Graw-Hill, New York, USA, 11th edition.



Wolsey, L. A. (1998). *Integer programming*. John Wiley & Sons