Programação Inteira

Análise de sistemas

Maria M. Gamboa

2º Semestre de 2023. 16/10/2023

Programação inteira

- Otimização discreta
- Otimização inteira
- Programação combinatória
- Programação discreta

Algumas (ou todas) as variáveis pertencem a um conjunto discreto tipicamente um subconjunto dos inteiros.

Formulação problemas

ETE's a construir

Considere uma região com baixa densidade populacional média, com rede de esgoto precária, que somente junta efluentes de n pequenos setores e entrega cada um sem tratar no curso d'água local. Cada um dos setores atende uma população diferente, p_i.

Por convenio com uma universidade foram elaborados projetos de n pequenas ETE's (uma para cada setor), com estimativas de custo c_i.

Foi obtida verba especial de V para projetos de saneamento, e você deve indicar quais ETE's construir para que a maior quantidade de pessoas seja atendida.

Formulação problemas

ETE's a construir

$$\text{maximizar: } f(x) = \sum_{i=1}^{n} p_i x_i$$

$$\text{s. a: } \sum_{i=1}^{n} c_i x_i \leq V$$

$$x \in \mathbb{B}_+^n$$

Formulação problemas

ETE's a construir

$$\text{maximizar: } f(x) = \sum_{i=1}^{n} p_i x_i$$

$$\text{s. a: } \sum_{i=1}^{n} c_i x_i \leq V$$

$$x \in \mathbb{B}_{+}^{n}$$

Problema da mochila

Formulação problemas

ETE's a construir

$$
\begin{aligned}
\text{maximizar: } & f(x) = \sum_{i=1}^{n} p_i x_i \\
\text{s. a: } & \sum_{i=1}^{n} c_i x_i \leq V \\
& x \in \mathbb{B}_+^n
\end{aligned}
$$

Problema da mochila
Existem variações de maior complexidade como mochila multidimensional, mochila inteira, múltiplas mochilas...

Formulação problemas

Atendimento de ocorrências na rede

Uma companhia de água está recebendo grande número de reclamações por vazamentos e outras ocorrências na rede, e o atendimento está sendo muito demorado.

Em reunião sobre o tema é levantado que muito tempo é gasto indo a atender a ocorrência e voltando à companhia para sair de novo, às vezes ao mesmo setor que já foram.

Surge então a ideia de juntar todas as n ocorrências ativas no inicio do dia e elaborar uma rota para que a equipe atenda todas com menos perda de tempo por deslocamentos.

Como a pessoa com maior formação, você deve elaborar as rotas.

Formulação problemas

Atendimento de ocorrências na rede

Primeira ideia:

$$x_{i,j} = 1 \text{ Se vai de } i \text{ para } j \text{ ; 0 se não}$$

$$\text{minimizar: } f(x) = \sum_{i=1}^{n} \sum_{j>i}^{n} t_{ij} x_{ij}$$

$$\text{s. a: } \sum_{j<i} x_{ji} + \sum_{j>i} x_{ij} = 2 \quad i = 1, 2...n$$

$$x \in \mathbb{B}_{+}^{n(n-1)/2}$$

Formulação problemas

Atendimento de ocorrências na rede

Primeira ideia:

$$x_{i,j} = 1 \text{ Se vai de } i \text{ para } j\text{ ; 0 se não}$$

$$\text{minimizar: } f(x) = \sum_{i=1}^{n} \sum_{j>i}^{n} t_{ij} x_{ij}$$

$$\text{s. a: } \sum_{j<i} x_{ji} + \sum_{j>i} x_{ij} = 2 \quad i = 1, 2 \ldots n$$

$$x \in \mathbb{B}_{+}^{n(n-1)/2}$$

Pode gerar subrotas desconexas

Formulação problemas

Atendimento de ocorrências na rede

Formulação completa:

S: subrota

$$
\begin{aligned}
\text{minimizar: } & f(x) = \sum_{i=1}^{n} \sum_{j>i}^{n} t_{ij} x_{ij} \\
\text{s. a: } & \sum_{j<i} x_{ij} + \sum_{j>i} x_{ij} = 2 \\
& \sum_{i \in S} \sum_{j \notin S} x_{ij} + \sum_{i \notin S} \sum_{j \in S} x_{ij} \geq 2 \quad S \subset N \\
& x \in \mathbb{B}_{+}^{n(n-1)/2}
\end{aligned}
$$

Formulação problemas

Atendimento de ocorrências na rede

Formulação completa:

$$
\begin{aligned}
S&: \text{ subrota} \\
\text{minimizar: } & f(x) = \sum_{i=1}^{n} \sum_{j>i}^{n} t_{ij} x_{ij} \\
\text{s. a: } & \sum_{j<i} x_{ij} + \sum_{j>i} x_{ij} = 2 \\
& \sum_{i \in S} \sum_{j \notin S} x_{ij} + \sum_{i \notin S} \sum_{j \in S} x_{ij} \geq 2 \quad S \subset N \\
& x \in \mathbb{B}_{+}^{n(n-1)/2}
\end{aligned}
$$

Problema do caixeiro-viajante

Variações: vários caixeiros viajantes, caixeiro-viajante e aquisição, e lucro, etc...

Formulação problemas

Coleta de material reciclável

As cooperativas de reciclagem são grandes alternativas para o melhoramento da gestão de resíduos sólidos, além de ter alto impacto social positivo.

A cooperativa do município tem somente um caminhão para fazer a coleta de material reciclável, e se não consegue garantir uma frequência de atendimento pelo menos nas principais ruas da cidade pode ter seu funcionamento suspenso.

Até agora a rota é decidida pela experiência do motorista, e não está conseguindo atender todas as ruas, de forma que pedem sua ajuda para melhorar a coleta.

Formulação problemas

Coleta de material reciclável

$$
\begin{aligned}
\text{minimizar:}\ & f(x) = \sum_{i,j \in E} t_{ij} x_{ij} \\
\text{s. a:}\ & \sum_{\{j:(i,j) \in E\}} x_{ij} - \sum_{\{j:(j,i) \in E\}} x_{ji} = 0 \\
& x_{ij} \geq 1 \, \forall (i,j) \in E \\
& x \in \mathbb{Z}_{+}^{E}
\end{aligned}
$$

Formulação problemas

Coleta de material reciclável

$$
\begin{aligned}
\text{minimizar: } & f(x) = \sum_{i,j \in E} t_{ij} x_{ij} \\
\text{s. a: } & \sum_{\{j:(i,j)\in E\}} x_{ij} - \sum_{\{j:(j,i)\in E\}} x_{ji} = 0 \\
& x_{ij} \geq 1 \, \forall (i,j) \in E \\
& x \in \mathbb{Z}_+^E
\end{aligned}
$$

Pontes de Königsberg - Problema do carteiro chinês

Programação inteira

Programação linear inteira PI
Variáveis de decisão inteiras

$$\begin{aligned} \text{maximizar:} \;\; & f(x) = cx \\ \text{s. a:} \;\; & Ax \leq b \\ & x \in \mathbb{Z}^n_+ \end{aligned}$$

Programação linear inteira mixta PIM

Variáveis inteiras e variáveis reais

$$\begin{aligned} \text{maximizar: } & f(x) = cx + dy \\ \text{s. a: } & Ax + Dy \leq b \\ & x \in \mathbb{R}^n_+ \quad , \quad y \in \mathbb{Z}^p_+ \end{aligned}$$

Programação inteira

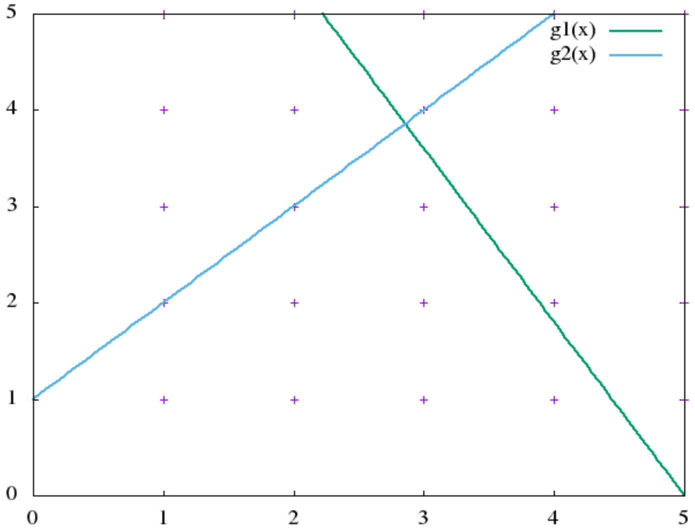
Programa binária
Variáveis de decisão binarias

$$\begin{aligned} \text{maximizar: } & f(x) = cx \\ \text{s. a: } & Ax \leq b \\ & x \in \mathbb{B}_+^n \end{aligned}$$

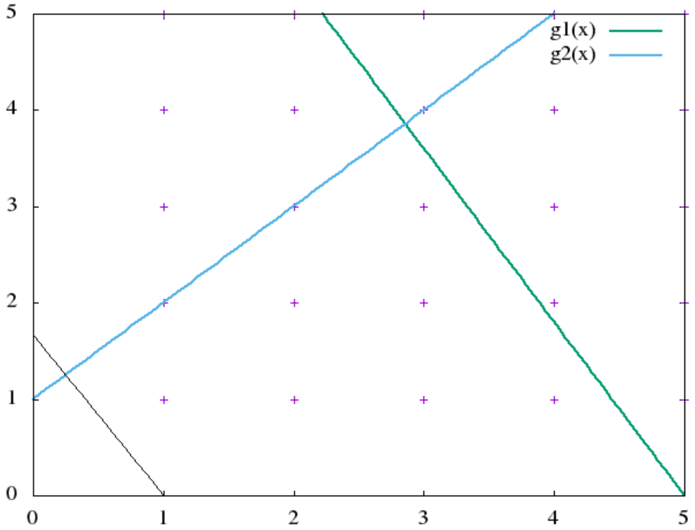
Exemplo PI:

$$
\begin{aligned}
\text{maximizar} f(X) &= 10x_1 + 6x_2 \\
\text{s. a: } 9x_1 + 5x_2 &\leq 45 \\
-4x_1 + 5x_2 &\leq 5 \\
x &\in \mathbb{Z}_+^2
\end{aligned}
$$

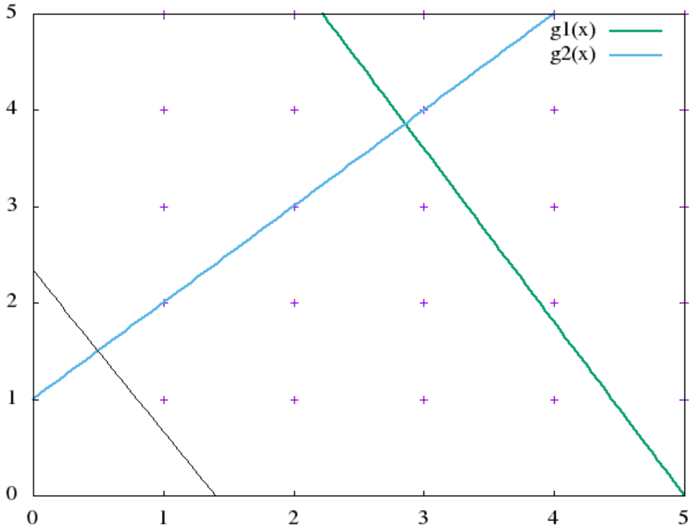
Exemplo PI:



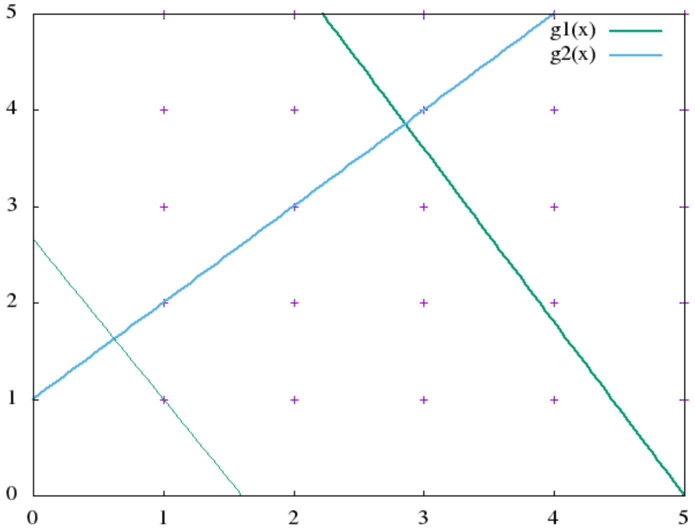
Exemplo PI:



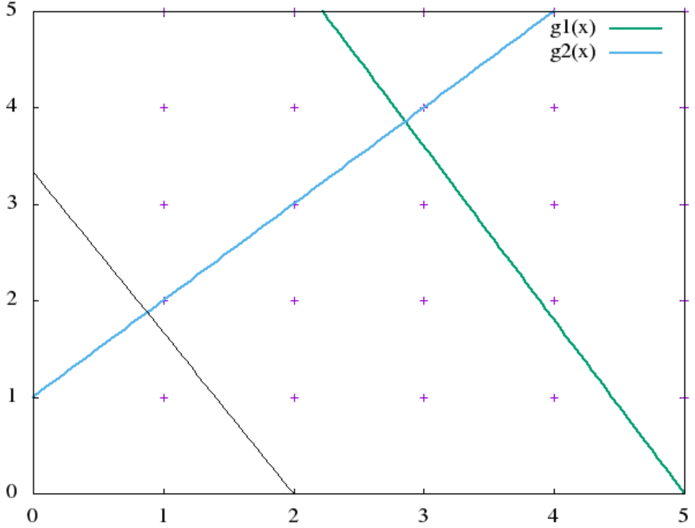
Exemplo PI:



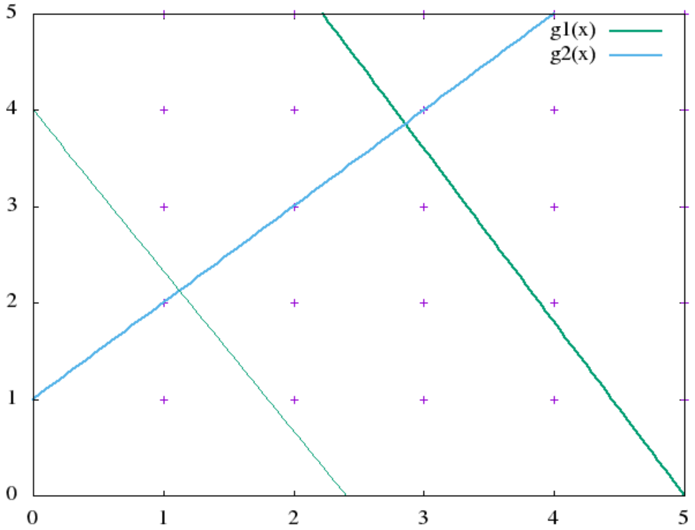
Exemplo PI:



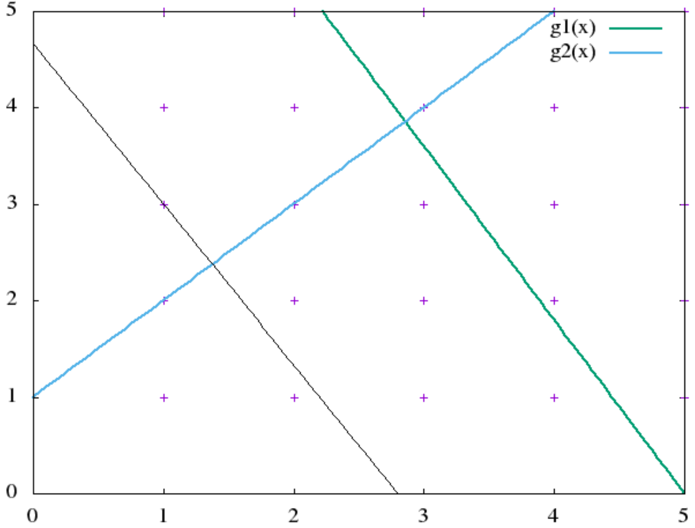
Exemplo PI:



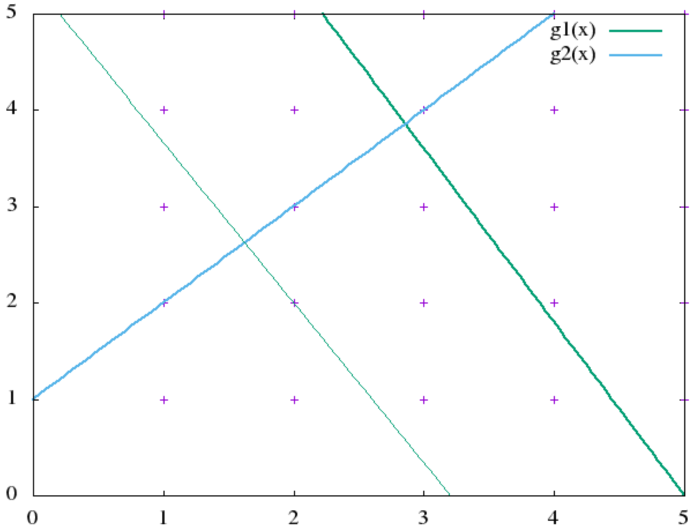
Exemplo PI:



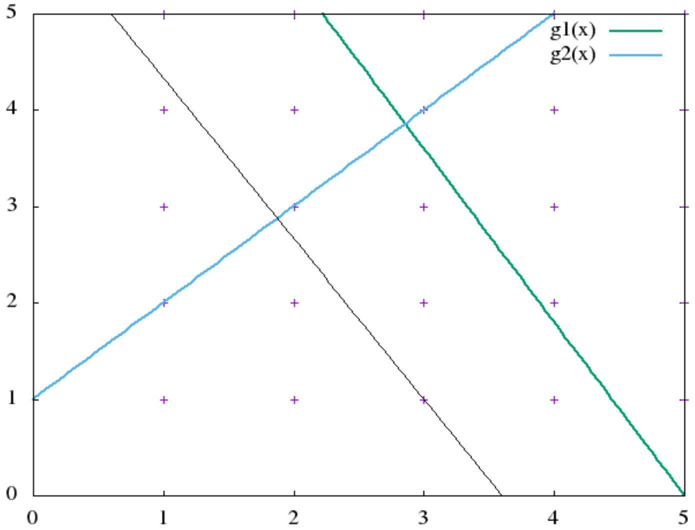
Exemplo PI:



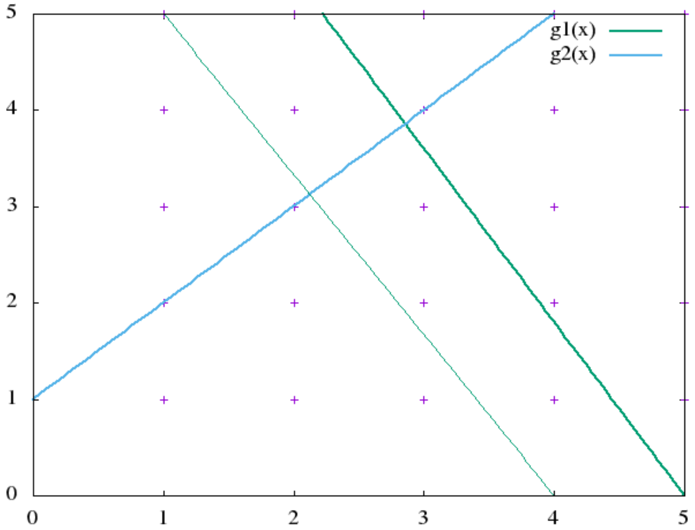
Exemplo PI:



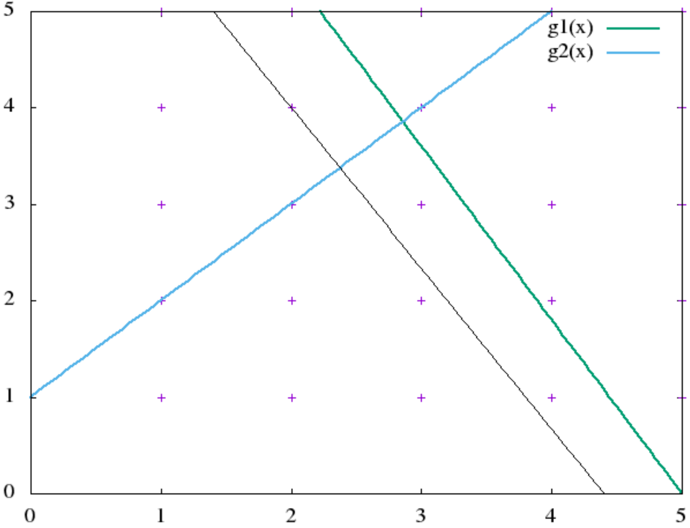
Exemplo PI:



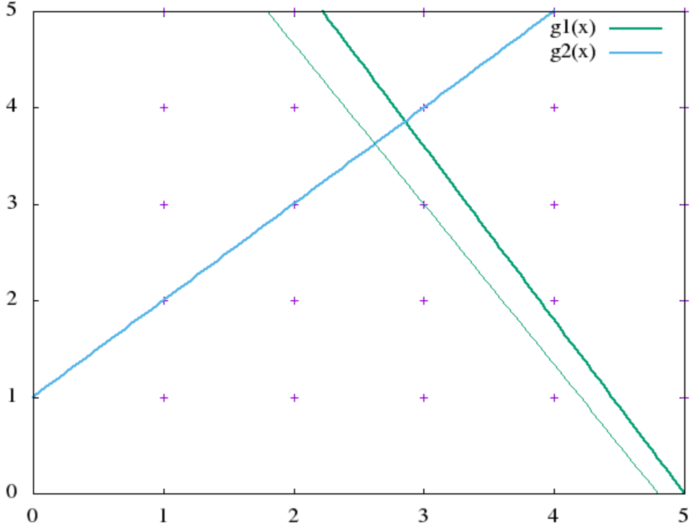
Exemplo PI:



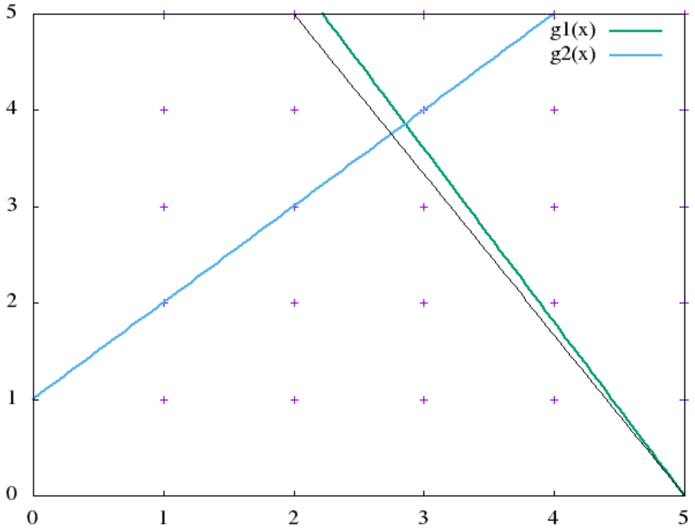
Exemplo PI:



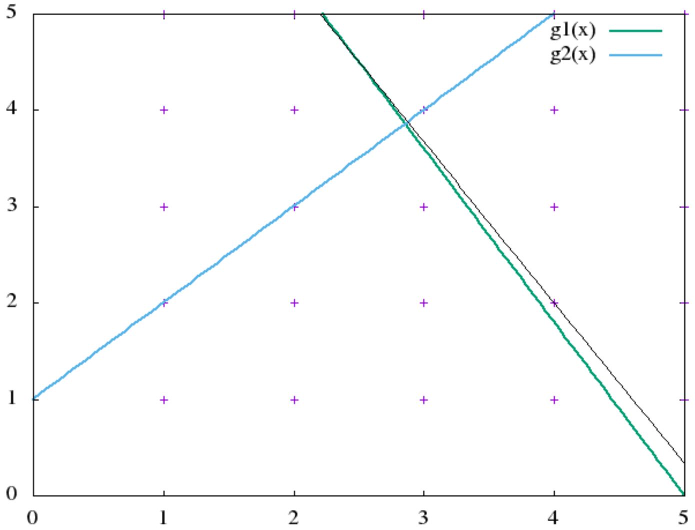
Exemplo PI:



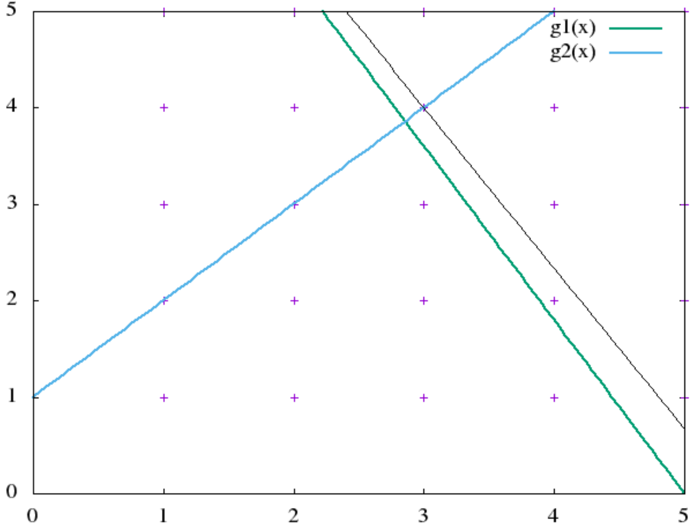
Exemplo PI:



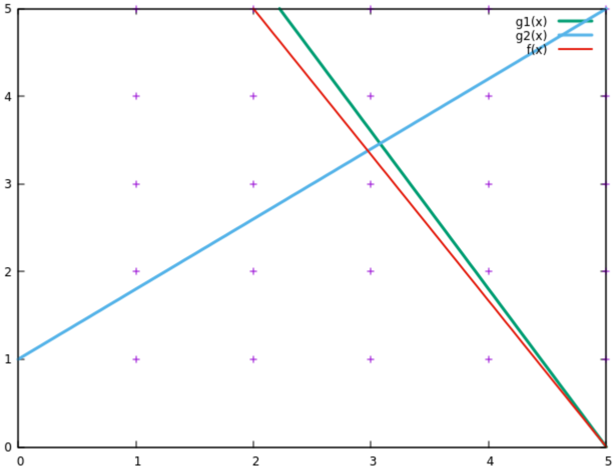
Exemplo PI:



Exemplo PI:



Exemplo PI:



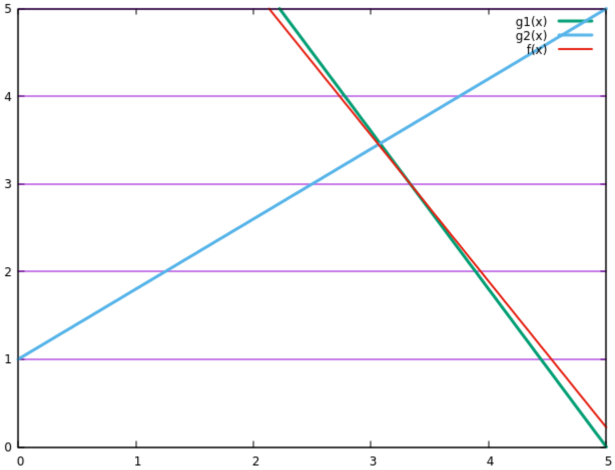
Ótimo (máximo): f(5.0 , 0.0) = 50.0

Exemplo PIM:

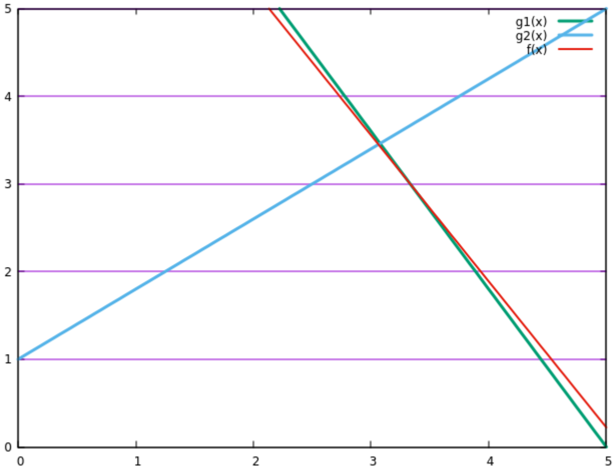
Se relaxarmos a condição de integralidade de x_1:

$$\begin{aligned} \text{maximizar} f(X) &= 10x_1 + 6x_2 \\ \text{s. a: } 9x_1 + 5x_2 &\leq 45 \\ -4x_1 + 5x_2 &\leq 5 \\ x_1 \in \mathbb{R}^1_+ \quad , \quad & x_2 \in \mathbb{Z}^1_+ \end{aligned}$$

Exemplo PIM:



Exemplo PIM:



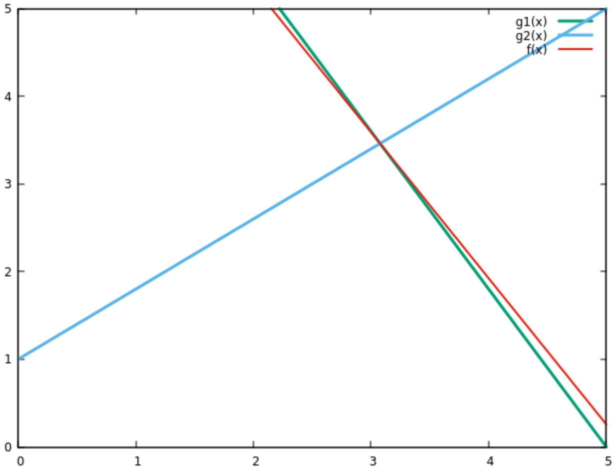
Ótimo (máximo): f(3.333 , 3) = 51.333

Exemplo PL:

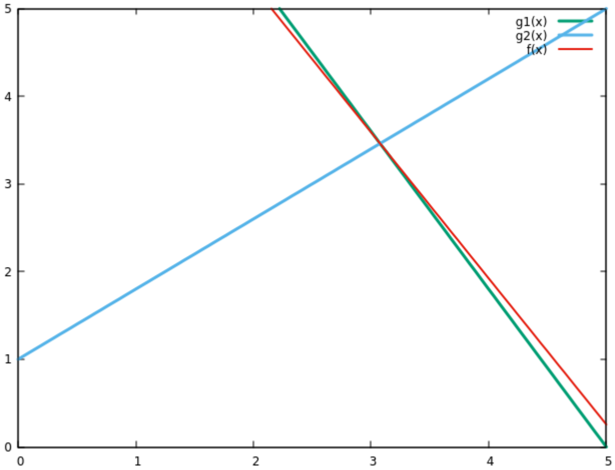
Se relaxarmos as duas condições de integralidade:

$$\begin{aligned} \text{maximizar} f(X) &= 10x_1 + 6x_2 \\ \text{s. a: } 9x_1 + 5x_2 &\leq 45 \\ -4x_1 + 5x_2 &\leq 5 \\ x_1, x_2 &\in \mathbb{R}^1_+ \end{aligned}$$

Exemplo PL:



Exemplo PL:



Ótimo (máximo): f(3.077, 3.462) = 51.538

Relaxação Linear

A relaxação linear do problema de PI leva a PIM, e no fim a PL.

- Conjunto soluções factíveis da PI: S_{PI}
- Conjunto soluções factíveis da PIM, relaxando alguns: S_{PIM}
- Conjunto soluções factíveis da PL, relaxando todos: S_{PL}

Relaxação Linear

A relaxação linear do problema de PI leva a PIM, e no fim a PL.

- Conjunto soluções factíveis da PI: S_{PI}
- Conjunto soluções factíveis da PIM, relaxando alguns: S_{PIM}
- Conjunto soluções factíveis da PL, relaxando todos: S_{PL}

$$S_{PI} \subset S_{PIM} \subset S_{PL}$$

Relaxação Linear

A relaxação linear do problema de PI leva a PIM, e no fim a PL.

- Conjunto soluções factíveis da PI: S_{PI}
- Conjunto soluções factíveis da PIM, relaxando alguns: S_{PIM}
- Conjunto soluções factíveis da PL, relaxando todos: S_{PL}

$$S_{PI} \subset S_{PIM} \subset S_{PL}$$

$$fmax_{PI} \leq fmax_{PIM} \leq fmax_{PL}$$

Resolução do problema de programação inteira

- Tem finitas soluções factíveis. Mas podem ser muitas
- Tem menos soluções factíveis que PL. Mas é mais difícil de resolver.

Resolução do problema de programação inteira

- Tem finitas soluções factíveis. Mas podem ser muitas
- Tem menos soluções factíveis que PL. Mas é mais difícil de resolver.

Tem finitas soluções factíveis $\rightarrow$ Enumeração

Resolução do problema de programação inteira

- Tem finitas soluções factíveis. Mas podem ser muitas
- Tem menos soluções factíveis que PL. Mas é mais difícil de resolver.

Tem finitas soluções factíveis $\rightarrow$ Enumeração

Enumeração completa?

Resolução do problema de programação inteira

- Tem finitas soluções factíveis. Mas podem ser muitas
- Tem menos soluções factíveis que PL. Mas é mais difícil de resolver.

Tem finitas soluções factíveis $\rightarrow$ Enumeração

Enumeração completa?

- Computador pessoal: 10^1teraFLOPS; Computação distribuida: 10^2teraFLOPS; Supercomputadores: 10^2petaFLOPS

Resolução do problema de programação inteira

- Tem finitas soluções factíveis. Mas podem ser muitas
- Tem menos soluções factíveis que PL. Mas é mais difícil de resolver.

Tem finitas soluções factíveis $\rightarrow$ Enumeração

Enumeração completa?

- Computador pessoal: 10^1teraFLOPS; Computação distribuida: 10^2teraFLOPS; Supercomputadores: 10^2petaFLOPS
- Problema do caixeiro viajante. Número de rotas $= (n-1)!$

Resolução do problema de programação inteira

- Tem finitas soluções factíveis. Mas podem ser muitas
- Tem menos soluções factíveis que PL. Mas é mais difícil de resolver.

Tem finitas soluções factíveis $\rightarrow$ Enumeração

Enumeração completa?

- Computador pessoal: 10^1teraFLOPS; Computação distribuida: 10^2teraFLOPS; Supercomputadores: 10^2petaFLOPS
- Problema do caixeiro viajante. Número de rotas $= (n-1)!$
- Crescimento do tempo para o número de cidades é exponencial!

Resolução do problema de programação inteira

Enumeração completa? Torna-se inviável!

Resolução do problema de programação inteira

Enumeração completa? Torna-se inviável!

Melhor: Enumeração implícita: Considerar subconjuntos de soluções, descartando alguns. Repetir implicitamente.
Estrategia de dividir para conquistar.

Resolução do problema de programação inteira

Enumeração completa? Torna-se inviável!

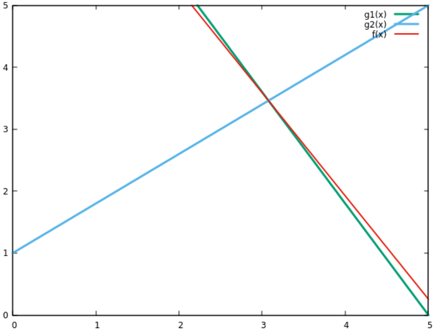
Melhor: Enumeração implícita: Considerar subconjuntos de soluções, descartando alguns. Repetir implicitamente.
Estrategia de dividir para conquistar.

Método: Branch and bound. Baseado em árvores de busca. Doig, Land (1960) e Dakin (1965)

Branch and bound

Exemplo

$$\begin{aligned} &\max\ f(X) = 10x_1 + 6x_2 \\ &\text{s. a: } 9x_1 + 5x_2 \leq 45 \\ &\qquad -4x_1 + 5x_2 \leq 5 \\ &\qquad x \in \mathbb{Z}_+^2 \end{aligned}$$



Branch and bound

- Relaxar inicialmente : Fazer relaxação linear e encontrar o ótimo com PL (simplex). Problema inicial com n restrições.

Branch and bound

- Relaxar inicialmente
- Ramificar : Considerando uma das variáveis, criar novas restrições com o inteiro menor e o maior ao valor ótimo, e com cada uma delas um novo problema. Obtêm dois problemas com n+1 restrições.

Branch and bound

- Relaxar inicialmente
- Ramificar
- Limitar : Resolver cada um dos novos problemas de PL.

Branch and bound

- Relaxar inicialmente
- Ramificar
- Limitar
- Eliminar : Desconsiderar os subproblemas que:
  *Não tem solução factível
  *A solução ótima tem valor de f. objetivo pior que outro obtido, e tem alguma variável que não é inteira.
  *A solução ótima tem todas as variáveis inteiras. Se o valor da f. objetivo for melhor que qualquer obtido, salvar ele.

Branch and bound

- Relaxar inicialmente
- Ramificar
- Limitar
- Eliminar
- Ramificar : Somente no problema não desconsiderado. Com o resultado do novo problema 'ótimo', fazer nova 'ramificação'. Considerando uma outra variável e os correspondentes inteiros maior e menor, criar dois novos problemas com n+2 restrições.

Branch and bound

- Relaxar inicialmente
- Ramificar
- Limitar
- Eliminar
- Ramificar
- Parar : Repetir até ter analisado todo o espaço e ter uma única solução ótima (se existir). Ou, critério tolerância

Branch and bound

- Relaxar inicialmente
- Ramificar
- Limitar
- Eliminar
- Ramificar
- Parar

Variáveis binárias: A partição do problema segundo x_i é utilizando $x_i = 0$ e $x_i = 1$ nos novos problemas, respectivamente.
Seleção de nós pode ser sequencial

Branch and bound

Essa ideia geral é adaptada para tipos específicos de problema, para melhorar desempenho.

Seleção do problema (nó) a ser ramificado:

- A priori: Busca em profundidade (mais usada. Mais tempo). Busca lateral.
- Adaptativo: Maior limitante superior (menor tempo, maior memória).