# Introdução a Programação Linear

Flávio Keidi Miyazawa

UNIVERSIDADE DE CAMPINAS - UNICAMP
INSTITUTO DE COMPUTAÇÃO - IC

Campinas, 2002-2016

# Sumário I

# Programação Linear

Programação linear

- dá a resolução exata para muitos problemas
- dá a resolução aproximada para muitos problemas
- faz parte dos principais métodos para obter soluções ótimas
- faz parte dos principais métodos para obter soluções aproximadas
- dá excelentes delimitantes para soluções ótimas
- pode ser executada muito rapidamente
- há diversos programas livres e comerciais

**Obs.:** Algumas teorias sobre a estrutura dos programas lineares serão apresentadas para se entender os resultados das reduções, mas o foco da aula será nas reduções para Programas Lineares.

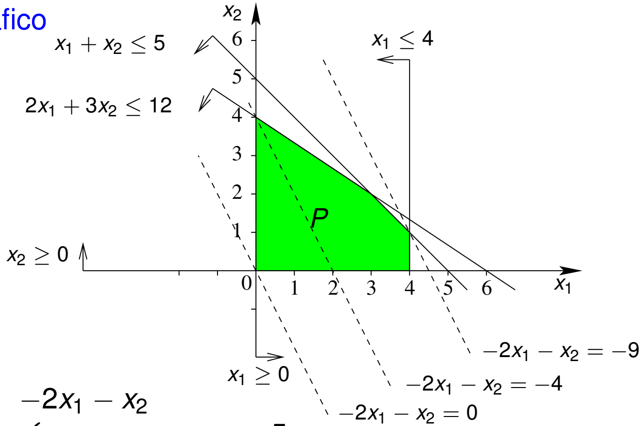
# Programação Linear

**Problema PL:** Dados matriz $A = (a_{ij}) \in \mathbb{Q}^{n \times m}$, vetores $c = (c_i) \in \mathbb{Q}^n$ e $b = (b_i) \in \mathbb{Q}^m$, encontrar vetor $x = (x_i) \in \mathbb{Q}^m$ (se existir) que

$$
\begin{array}{ll}
\text{minimize} & c_1x_1 + c_2x_2 + \cdots + c_mx_m \\
\text{sujeito a} & \left\{ \begin{array}{rcl}
a_{11}x_1 + a_{12}x_2 + \cdots + a_{1m}x_n & \leq & b_1 \\
a_{21}x_1 + a_{22}x_2 + \cdots + a_{2m}x_n & \geq & b_2 \\
\vdots & \vdots & \vdots \\
a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nm}x_n & = & b_n \\
x_i \in \mathbb{Q} & &
\end{array} \right.
\end{array}
$$

**Teorema:** *PL pode ser resolvido em tempo polinomial.*

## Exemplo gráfico




$$\text{minimize} \quad -2x_1 - x_2$$

$$\text{sujeito a} \quad \begin{cases} x_1 + x_2 \leq 5 \\ 2x_1 + 3x_2 \leq 12 \\ x_1 \leq 4 \\ x_1 \geq 0 \\ x_2 \geq 0 \end{cases}$$

Sol. ótima: $x_1 = 4$ e $x_2 = 1$
Valor da sol. ótima = $-9$

**Teorema:** *Uma solução que é vértice do PL pode ser encontrada em tempo polinomial.*

Algoritmos polinomiais para resolver PL:

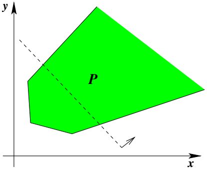
- Algoritmo dos elipsóides (Khachiyan'79) e
- Método dos pontos interiores (Karmarkar'84).

Algoritmos exponenciais para resolver PL:

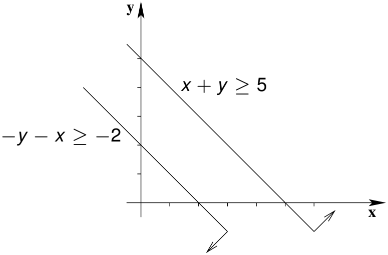
- Método simplex (Dantzig'47) (tempo médio polinomial).

# Nem sempre encontramos uma solução ótima

- Sistema ilimitado




- Sistema inviável



# Resolvedores de Sistemas Lineares

## Programas comerciais

- CPLEX / IBM: `www.ilog.com/products/cplex/`
- XPRESS: `www.dashoptimization.com/`
- GUROBI:: `http://www.gurobi.com/`

## Programas livres

- SCIP: `http://scip.zib.de/`
- COIN-CLP: `www.coin-or.org/Clp/`
- SoPlex / Zib: `www.zib.de/Optimization/Software/Soplex`
- GLPK / Gnu: `www.gnu.org/software/glpk/glpk.html`
- LEMON / Coin-OR: `http://www.coin-or.org/`
  Biblioteca de grafos usando solver GLPK ou COIN-CLP

# Trecho de implementação usando LEMON

```
#include <lemon/lp.h> // g++ ex_lp1.cpp -lemon -lglpk -o ex_lp1
using namespace lemon;
using namespace std;
int main()
{ Lp lp;                 / Declarando/criando um LP (inicialmente v
  Lp::Col x1 = lp.addCol();  // Adicionando duas variáveis
  Lp::Col x2 = lp.addCol();
  lp.addRow(   x1 +    x2 <= 5 );   // Adicionando restrições
  lp.addRow( 2*x1 + 3*x2  <= 12);
  // Definindo os limitantes superior e inferior de cada variável
  lp.colLowerBound( x1, 0 );   lp.colUpperBound( x1, 4 );
  lp.colLowerBound( x2, 0 );
  // Definindo a função objetivo e se de minimização ou maximizaçã
  lp.obj( -2*x1 - x2 );   lp.min();
  lp.solve();        // Resolvendo o sistema
  if (lp.primalType() == Lp::OPTIMAL) {// Imprimindo valor das var
    cout << "Valor da funcao objetivo: " << lp.primal() << endl;
    cout << "x1 = " << lp.primal(x1) << endl;
    cout << "x2 = " << lp.primal(x2) << endl;
  } else {cout << "Nao encontrou solucao otima." << endl;}
  return 0;}
```

## Problema da Dieta

São dados:

- $m$ alimentos: $ali_1, \ldots, ali_m$
- $n$ nutrientes: $nut_1, \ldots, nut_n$
- preço $p_j$ de cada alimento $ali_j$
- recomendações diárias $r_i$ de cada nutriente $nut_i$, para uma dieta balanceada
- quantidade $q_{ij}$ do nutriente $nut_i$ no alimento $ali_j$,

Objetivo: Encontrar uma dieta mais barata respeitando as recomendações nutricionais

## Exemplo: Dieta I

*Nutrientes, com recomendações diárias mínima e máxima*

| Nutriente | mínimo | máxima |
|---|---|---|
| $nut_1$ = cálcio | 800 | 1200 |
| $nut_2$ = ferro | 10 | 18 |

*Alimentos, com preço e composição nutricional:*

| Alimento | preço | cálcio | ferro |
|---|---|---|---|
| $ali_1$ = carne de boi (kg) | 20.0 | 110.00 | 29.00 |
| $ali_2$ = feijão cozido (kg) | 6.0 | 170.00 | 15.00 |
| $ali_3$ = leite desnatado (lt) | 2.0 | 1150.00 | 0.00 |

## Formulação Linear

### Variáveis

- $x_1$ quantidade de *carne de boi* a comprar
- $x_2$ quantidade de *feijão cozido* a comprar e
- $x_3$ quantidade de *leite desnatado* a comprar.

### Função objetivo

- Minimizar o preço pago por todos os alimentos. I.e.,

$$\text{minimize } 20{,}0x_1 + 6{,}0x_2 + 2{,}0x_3$$

### Restrições

- quantidades $x_1$, $x_2$ e $x_3$ devem ser valores válidos e satisfazer restrições nutricionais

## Restrições nutricionais

- Quantidade de cálcio consumido dentro das recomendações diárias

$$\begin{aligned} 110x_1 + 170x_2 + 1150x_3 &\geq 800 \\ 110x_1 + 170x_2 + 1150x_3 &\leq 1200 \end{aligned}$$

- Quantidade de ferro consumido dentro das recomendações diárias

$$\begin{aligned} 29x_1 + 15x_2 &\geq 10 \\ 29x_1 + 15x_2 &\leq 18 \end{aligned}$$

## Restrições de não negatividade

- Nenhuma quantidade pode ser negativa

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

## Formulação Linear

$$
\begin{array}{ll}
\text{minimize} & 20{,}0x_1 + 6{,}0x_2 + 2{,}0x_3 \\
\text{sujeito a} & \left\{
\begin{array}{rrrcl}
110x_1 + & 170x_2 + & 1150x_3 & \geq & 800 \\
110x_1 + & 170x_2 + & 1150x_3 & \leq & 1200 \\
29x_1 + & 15x_2 & & \geq & 10 \\
29x_1 + & 15x_2 & & \leq & 18 \\
x_1 \geq 0 & x_2 \geq 0 & x_3 \geq 0 & &
\end{array}
\right.
\end{array}
$$

Resolvendo-se:

Obtemos uma dieta de 5,194 reais com

$x_1 = 0$ kg. de carne de boi,

$x_2 = 0.666$ kg. de feijão cozido e

$x_3 = 0.597$ lt. de leite desnatado.

A quantidade diária de cálcio nesta dieta é 800 e de ferro é 10.

# Trecho de implementação usando LEMON

```
int main() {
 Lp lp;  // Declarando um LP (inicialmente vazio)
 typedef Lp::Col LPvar;
 // Adicionando três variáveis
 LPvar x1=lp.addCol(); LPvar x2=lp.addCol(); LPvar x3=lp.addCol();
 // Definindo a função objetivo e direção de minimização
 lp.obj( 20.0*x1 + 6.0*x2 + 2.0*x3 );   lp.min();
 lp.addRow( 800 <= 110*x1 + 170*x2 + 1150*x3 <= 1200 );
 lp.addRow(  10 <=  29*x1 +  15*x2           <= 18 );
 lp.colLowerBound( x1, 0 );    // restrição x1 >= 0
 lp.colLowerBound( x2, 0 );    // restrição x2 >= 0
 lp.colLowerBound( x3, 0 );    // restrição x3 >= 0
 lp.solve();   // Resolvendo o sistema
 if (lp.primalType() == Lp::OPTIMAL) {  // Imprimindo o valor das
   cout << "Valor da funcao objetivo: " << lp.primal() << endl;
   cout << "x1 = " << lp.primal(x1) << endl;
   cout << "x2 = " << lp.primal(x2) << endl;
   cout << "x3 = " << lp.primal(x3) << endl;
 } else {cout << "Nao encontrou solucao otima." << endl;}
 return 0;
}
```

## Exemplo: Dieta II

*Nutrientes, com recomendações diárias mínima e máxima*

| *Nutriente* | *mínimo* | *máxima* |
|---|---|---|
| $nut_1$ = *fósforo* | *800* | *1200* |
| $nut_2$ = *vitamina C* | *60* | *90* |

*Alimentos, com preço e composição nutricional:*

| *Alimento* | *preço* | *fósforo* | *vitamina C* |
|---|---|---|---|
| $ali_1$ = *carne de boi (kg)* | *20.0* | *1800.00* | *0.00* |
| $ali_2$ = *feijão cozido (kg)* | *6.0* | *490.00* | *10.00* |

### Variáveis

- $x_1$ quantidade de *carne de boi* a comprar
- $x_2$ quantidade de *feijão cozido* a comprar

## Formulação Linear

$$
\begin{array}{ll}
\text{minimize} & 20{,}0x_1 + 6{,}0x_2 \\
\text{sujeito a} & \left\{
\begin{array}{rrcl}
1800x_1 + & 490x_2 & \geq & 800 \\
1800x_1 + & 490x_2 & \leq & 1200 \\
 & 10x_2 & \geq & 60 \\
 & 10x_2 & \leq & 90 \\
x_1 \geq 0 & x_2 \geq 0 & &
\end{array}
\right.
\end{array}
$$

Este sistema é inviável. De fato:

- O único a ter vitamina C é o feijão (e bem pouco)
- Por $10x_2 \geq 60$ temos que $x_2 \geq 6$
  I.e., para suprir necessidade de vitamina C, precisamos ingerir pelo menos 6 kg. de feijão por dia!!
- 6kg. de feijão contém $6 \star 490 = 2940$mg de fósforo
  acima do limite diário permitido de 1200mg de fósforo.

## Otimização de Portfolio

- Temos 100.000 reais para investir em ações
- As ações selecionadas e a porcentagem de retorno esperado em 1 ano são:

| Empresa | Retorno (em %) |
|---|---|
| $emp_1$ = Petrobrás (petróleo/estatal) | 9.0% |
| $emp_2$ = Vale do Rio Doce (siderurgia) | 10.2% |
| $emp_3$ = Votorantim (siderurgia) | 6.5% |
| $emp_4$ = Texaco (petróleo) | 9.5% |
| $emp_5$ = Sanasa (água/estatal) | 8.5% |

A recomendação dos especialistas é a seguinte:

- Recomenda se investir pelo menos 25% e no máximo 55% em empresas estatais.
- Petrobrás e Texaco são empresas do mesmo setor (petróleo). Recomenda-se que o investimento nas duas não passe de 55 %.
- Vale do Rio Doce e Votorantim são do mesmo setor (siderurgia) recomenda que o investimento nas duas não passe de 45 %.
- Apesar da Vale do Rio Doce ter a maior taxa de retorno, há boatos que ela pode estar maquiando faturamento. Recomenda-se que a quantidade de investimento nela não passe de 60% do total de investimento feito em empresas de siderurgia.

# Formulação Linear

## Variáveis

- $x_1$ quantidade de investimento na *Petrobrás*
- $x_2$ quantidade de investimento na *Vale do Rio Doce*
- $x_3$ quantidade de investimento na *Votorantim*
- $x_4$ quantidade de investimento na *Texaco*
- $x_5$ quantidade de investimento na *Sanasa*

## Função objetivo

- Maximizar o lucro esperado,

$$\text{maximize } 0{,}090x_1 + 0{,}102x_2 + 0{,}065x_3 + 0{,}095x_4 + 0{,}085x_5$$

## Restrições

- quantidades $x_1, \ldots, x_5$ devem ser valores válidos e devem satisfazer recomendações dos especialistas

## Restrições impostas por especialistas

- Recomenda se investir pelo menos 25% e no máximo 55% em empresas estatais.

$$\begin{aligned} x_1 + x_5 &\geq 25000 \\ x_1 + x_5 &\leq 55000 \end{aligned}$$

- Petrobrás e Texaco são empresas do mesmo setor (petróleo). Recomenda-se que o investimento nas duas não passe de 55 %.

$$x_1 + x_4 \leq 55000$$

- Vale do Rio Doce e Votorantim são do mesmo setor (siderurgia) recomenda que o investimento nas duas não passe de 45 %.

$$x_2 + x_3 \leq 45000$$

- Apesar da Vale do Rio Doce ter a maior taxa de retorno, há boatos que ela pode estar maquiando faturamento. Recomenda-se que a quantidade de investimento nela não passe de 60% do total de investimento feito em empresas de siderurgia.

$$x_2 \leq 0.6(x_2 + x_3)$$

$$\text{I.e.,} \quad -0.4x_2 + 0.6x_3 \geq 0$$

**Demais Restrições**

- Total investido é 100.000

$$x_1 + x_2 + x_3 + x_4 + x_5 = 100000$$

- Nenhuma quantidade pode ser negativa

$$x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0, \quad x_4 \geq 0, \quad x_5 \geq 0$$

# Formulação Linear

$$
\begin{array}{ll}
\text{maximize} & 0{,}090x_1 + 0{,}102x_2 + 0{,}065x_3 + 0{,}095x_4 + 0{,}085x_5 \\
\text{sujeito a} & \left\{
\begin{array}{rrrrrcl}
x_1+ & x_2+ & x_3+ & x_4+ & x_5 & = & 100000 \\
x_1+ & & & & x_5 & \geq & 25000 \\
x_1+ & & & & x_5 & \leq & 55000 \\
x_1+ & & & x_4 & & \leq & 55000 \\
& x_2+ & x_3 & & & \leq & 45000 \\
& -.4x_2+ & .6x_3 & & & \geq & 0 \\
x_1 \geq 0 & x_2 \geq 0 & x_3 \geq 0 & x_4 \geq 0 & x_5 \geq 0 & &
\end{array}
\right.
\end{array}
$$

Resolvendo-se:

Obtemos uma lucro estimado de 9094 reais investindo

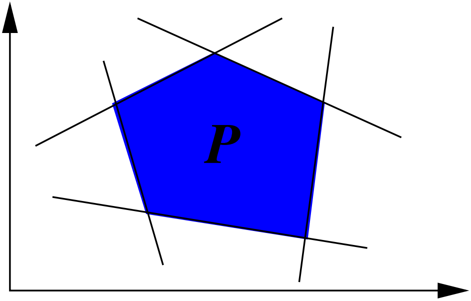
$x_1 = 0$ na Petrobrás, $x_2 = 12000$ na Vale do Rio Doce,
$x_3 = 8000$ na Votorantim, $x_4 = 55000$ na Texaco e
$x_5 = 25000$ na Sanasa.

# ALGUMAS DEFINIÇÕES

**Def.:** Chamamos o conjunto de pontos $P$,

$$P := \left\{ x \in \mathbb{Q}^n : \begin{array}{ccc} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1m}x_n & \leq & b_1 \\ \vdots & \vdots & \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nm}x_n & \geq & b_n \end{array} \right\}$$

como sendo um poliedro.



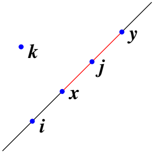
**Def.:** Dado conjunto de pontos $S$, dizemos que $y$ é uma combinação convexa dos pontos de $S$ se $y = \alpha_1 y_1 + \alpha_2 y_2 + \cdots + \alpha_n y_n$, onde $y_i \in S$, $\alpha_i \geq 0$ e $\sum_i \alpha_i = 1$

**Exemplo:** *Os pontos que são combinação convexa de dois pontos $x$ e $y$ são:*

$$\text{conv}(\{x, y\}) := \{\alpha_1 x + \alpha_2 y : \ \alpha_1 \geq 0, \ \alpha_2 \geq 0, \ \alpha_1 + \alpha_2 = 1\}$$

*substituindo $\alpha_2 = 1 - \alpha_1$, temos*

$$\text{conv}(\{x, y\}) := \{\alpha x + (1 - \alpha) y : \ 0 \leq \alpha \leq 1\}$$



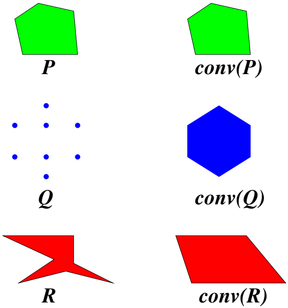

*$j$ é combinação convexa de $x$ e $y$, mas $i$ e $k$ não são.*

Exercício: Mostre que se $x, y \in conv(S)$ e $z \in conv(x, y)$, então temos também que $z \in conv(S)$.

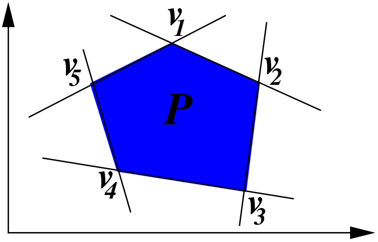
**Def.:** *O* *fecho convexo* *de* $S$*, denotado por* $conv(S)$ *é o conjunto dos pontos que são combinação convexa dos pontos de* $S$*.*

**Exemplo:** Dado dois pontos $x, y \in \mathbb{R}^n$, qualquer ponto que esteja no segmento de reta que liga $x$ a $y$ está em $conv(\{x, y\})$.

**Exemplo:**



**Def.:** Os vértices ou pontos extremais de $P$ são os pontos de $P$ que não podem ser escritos como combinação convexa de outros pontos de $P$




Vértices de $P$: $v_1, v_2, v_3, v_4, v_5$

Note que um vértice do poliedro é um ponto único que satisfaz um determinado conjunto de restrições na igualdade.

# Determinantes e Resolução de Sistemas Lineares

*Determinante por Laplace*

Dada matriz $A$ de ordem $n$ e coluna $j$,

$$|A| = \sum_{i=1}^{n} a_{ij} \cdot (-1)^{i+j} |A^{i,j}|,$$

onde $A^{i,j}$ é a matriz $A$ removendo a linha $i$ e coluna $j$

*Resolução de Sistemas Lineares por Cramer*

Dado matriz quadrada $A$ e vetor $b$, encontrar $x$ tal que $Ax = b$:

$$x_i = \frac{|A_b^j|}{|A|},$$

onde $A_b^j$ é a matriz $A$ com a coluna $j$ trocada pelo vetor $b$.

Def.: Submatriz de $A$ é uma matriz obtida de $A$ removendo linhas e colunas.

**Teorema:** *Todo vértice $y$ de um poliedro, definido por uma matriz $A$, é determinado por uma submatriz não singular $A'$ de $A$ tal que $A' \cdot y = b'$, onde $b'$ contém os correspondentes elementos de $b$ das linhas de $A'$.*

Def.: Se todo vértice de um poliedro tem apenas componentes inteiras, então o poliedro é dito ser inteiro.

**Corolário:** *Se $P$ é um poliedro definido por matriz $A$ e vetor $b$ (e.g., $Ax \leq b$), tal que toda submatriz de $A$ tem determinante em $\{-1, 0, +1\}$ e $b$ é inteiro, então todo vértice do poliedro é inteiro.*

**Def.:** *Uma matriz $A$ é dita ser totalmente unimodular (TU) se para qualquer submatriz quadrada $A'$ de $A$ (selecionando linhas e colunas de $A$) temos $det(A') \in \{-1, 0, +1\}$.*

**Teorema:** *Se $A$ é TU, então para todo vetor inteiro $b$ temos que $P = \{x : Ax \leq b\}$ é inteiro.*

**Lema:** *Se A e B são matrizes TU, então as seguintes matrizes também são TU:*

1. *Matriz obtida de A removendo uma linha (coluna)*
2. *Matriz obtida de A duplicando uma linha (coluna)*
3. *Matriz obtida de A trocando duas linhas (colunas)*
4. $A^T$ *e* $-A$
5. $(\, A \mid I \,)$ *e* $\begin{pmatrix} A \\ I \end{pmatrix}$
6. $(\, A \mid -A \,)$

*Prova.* Itens 1., 2., e 3., seguem da propriedade de determinante e de matriz TU.
4. Determinante da transposta é igual ao da original e multiplicar linha por -1 apenas troca sinal do determinante.
5. Basta aplicar regra de Laplace nos elementos da submatriz de $I$.
6. Duplicar colunas e trocar sinal da coluna mantém TU. □

**Teorema:** *Se A é TU e b e u são vetores inteiros, então os seguintes poliedros são inteiros:*

1. $\{x : Ax \geq b\}$
2. $\{x : Ax \leq b,\ x \geq 0\}$
3. $\{x : Ax = b,\ x \geq 0\}$
4. $\{x : Ax = b,\ 0 \leq x \leq u\}$

*Prova.* Note que o poliedro $\{x : Ax = b\}$ é dado por $\{x : Ax \leq b,\ Ax \geq b\}$.

No item 4., basta ver que a matriz $\begin{pmatrix} A \\ -A \\ I \\ -I \end{pmatrix}$ é TU. □

**Lema:** *Se A é a matriz de incidência de grafo orientado, então A é TU.*

*Prova.* Por indução na ordem da submatriz. A base é direta.
Seja $B$ uma submatriz quadrada de $A$.

1. Se há uma coluna $z$ só com elementos nulos: Neste caso o determinante de $B$ é 0.
2. Se há coluna $z$ com exatamente um elemento não nulo $t \in \{-1, +1\}$: Neste caso, o determinante é $+t$ ou $-t$ vezes a submatriz removendo a linha e coluna de $t$ (regra de Laplace).
3. Caso contrário, toda coluna tem $-1$ e $+1$. Como a soma de todas as linhas é nula, o determinante é nulo.

□

De maneira geral, vale que:
**Lema:** *Se A é matriz formada por elementos em* $\{-1, 0, 1\}$*, e cada coluna tem no máximo uma ocorrência de -1 e no máximo uma ocorrência de 1, então A é TU.*
*Prova.* Exercício. □

**Lema:** *Se $A$ é a matriz de incidência de um grafo bipartido não-orientado $G = (X, Y, E)$, então $A$ é TU.*

*Prova.* Multiplique por -1 todas as linhas de $A$ correspondentes aos vértices de $Y$. Note que isto só muda o sinal dos subdeterminantes.

Isto nos dá uma matriz de incidência de um grafo orientado e a prova segue pelo lema anterior. $\square$

Exercícios:

- Mostre que se $A$ é a matriz de incidência de um grafo orientado e $B$ uma matriz obtida a partir de $A$ transformando alguns elementos não nulos em 0, então $B$ também é TU.
- Seja $G = (N, A)$ um grafo orientado com conjunto de nós $N$ e arcos $A$ e dois nós distintos $s$ e $t$. Apresente um poliedro definido por variável $x \in [0, 1]^{|A|}$ indexada em $A$, com número polinomial de restrições, e conjunto de vértices $\mathcal{V}$, tal que, se $\hat{x} \in \mathcal{V}$, então $\hat{x}$ representa os caminhos de $s$ a $t$, com $\hat{x}_a = 0$ para todo $a \in \delta^-(s) \cup \delta^+(t)$. Isto é, são caminhos sem arcos entrando em $s$ ou saindo de $t$.

# Fluxo de Custo Mínimo

Considere um sistema de produção de bens onde há centros consumidores e produtores.

- Todos os bens produzidos devem ser todos consumidos.
- Os bens produzidos chegam até os consumidores através de rotas.
- Cada rota tem uma capacidade máxima de escoamento (direcionada
- Cada rota tem seu custo para transportar cada unidade do bem.
- Objetivo: Transportar os bens dos produtores para os consumidores minimizando custo para transportar os bens.

Aplicações importantes na Computação:

- Transferência de dados em rede de computadores, deteção de "gargalos" da rede na transferência.
- Projeto de vias de tráfego no planejamento urbano.
- Deteção da capacidade de transmissão entre pontos de redes de telecomunicações.

## Notação:

Seja $G = (V, E)$ um grafo não orientado.

- Dado $S \subseteq V$, denotamos por $\delta(S)$ o conjunto de arestas com um extremo em $S$ e outro em $V \setminus S$.
- Dado $v \in V$, denotamos por $\delta(v)$ o conjunto $\delta(\{v\})$.

Seja $D = (V, E)$ um grafo orientado.

- Dado $S \subseteq V$, denotamos por $\delta^+(S)$ o conjunto de arcos com início em $S$ e fim (ponta) em $V \setminus S$.
- Dado $v \in V$, denotamos por $\delta^+(v)$ o conjunto $\delta^+(\{v\})$.
- Dado $S \subseteq V$, denotamos por $\delta^-(S)$ o conjunto de arcos com fim (ponta) em $S$ e início em $V \setminus S$.
- Dado $v \in V$, denotamos por $\delta^-(v)$ o conjunto $\delta^-(\{v\})$.

# Definições:

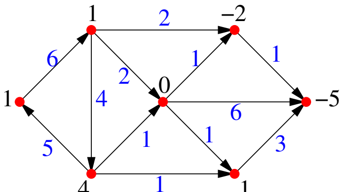
Seja $G = (V, E)$ um grafo orientado com função de capacidades nas arestas $c : E \to \mathbb{Q}^+$, demandas $b : V \to \mathbb{Q}$ e custos nas arestas $w : E \to \mathbb{Q}^+$.

**Def.:** *Dado um vértice $i \in V$, se $b_i < 0$ dizemos que $i$ é um consumidor e se $b_i > 0$ dizemos que $i$ é um produtor.*

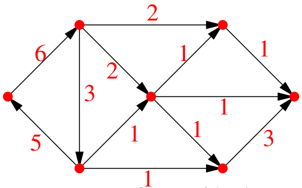
**Def.:** Dizemos que $x : E \to \mathbb{Q}^+$ é um fluxo em $G$ se

$$\sum_{e \in \delta^+(v)} x_e - \sum_{e \in \delta^-(v)} x_e = b_v, \quad \forall v \in V, \qquad e \qquad 0 \leq x_e \leq c_e, \quad \forall e \in E$$

**Ex.:**




G(V,E,c,d)




Um fluxo viável

**Def.:** Dado um fluxo $x : E \to \mathbb{Q}^+$ (respeitando $c$ e $b$) definimos o custo do fluxo $x$ como sendo $\sum_{e \in E} w_e x_e$.

**Problema** FLUXO DE CUSTO MÍNIMO: Dados um grafo orientado $G = (V, E)$, capacidades $c : E \to \mathbb{Q}^+$, demandas $b : V \to \mathbb{Q}^+$ e uma função de custo nas arestas $w : E \to \mathbb{Q}^+$, encontrar um fluxo $x : E \to \mathbb{Q}^+$ de custo mínimo.

**Encontrar** $x$ **tal que**

$$\min \sum_{e \in E} w_e x_e$$

$$\sum_{e \in \delta^+(v)} x_e - \sum_{e \in \delta^-(v)} x_e = b_v \quad \forall v \in V$$

$$0 \leq x_e \leq c_e \quad \forall e \in E$$

**Teorema:** *Se as capacidades nas arestas e as demandas dos vértices são inteiros (i.e., $c : E \rightarrow \mathbb{Z}^+$ e $b : V \rightarrow \mathbb{Z}^+$) então os vértices do poliedro do fluxo são inteiros.*

*Prova.* Segue do fato que a matriz de incidência de um grafo orientado é TU. □

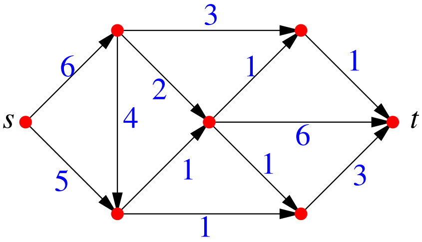
## Subproblemas

- Fluxo máximo de um vértice $s$ a um vértice $t$ ($st$-fluxo)
- Problema do corte de capacidade mínima
- Problema do caminho mínimo (com pesos não negativos)
- Emparelhamento de peso máximo em grafos bipartidos

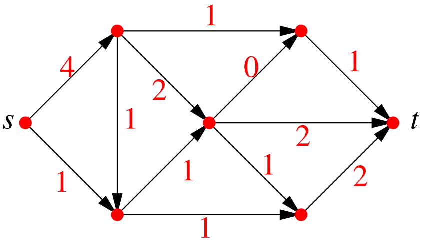
## Fluxo Máximo de $s$ para $t$ ($st$-fluxo máximo)

- Não temos custo para transportar os bens.
- Cada rota tem uma capacidade máxima de escoamento (direcionada).
- Só temos um produtor (vértice $s$) de produção arbitrariamente grande.
- Só temos um consumidor (vértice $t$) de consumo arbitrariamente grande.
- Nós internos apenas repassam bens.
- Objetivo: Maximizar o transporte de bens de $s$ para $t$, respeitando restrições de capacidade do fluxo.

**Exemplo:**




G(V,E) e capacidades




fluxo de valor 5

Seja $G = (V, E)$ um grafo orientado, capacidades $c : E \to \mathbb{Q}^+$ e vértices $s$ e $t$.

**Def.:** *Dizemos que $x : E \to \mathbb{Q}^+$ é um st-fluxo se*

$$\sum_{e \in \delta^+(v)} x_e - \sum_{e \in \delta^-(v)} x_e = 0 \quad \forall v \in V \setminus \{s, t\}$$

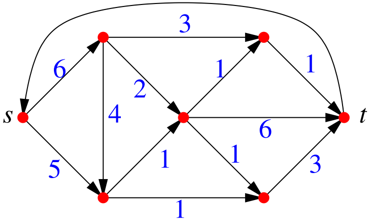
$$0 \leq x_e \leq c_e \quad \forall e \in E$$

**Def.:** *Dado um st-fluxo $x$ para $G = (V, E, c, s, t)$, definimos o valor do fluxo $x$ como sendo $x(\delta^+(s)) - x(\delta^-(s))$.*

**Problema** *st*-FLUXO MÁXIMO Dado um grafo orientado $G = (V, E)$, capacidades $c : E \to \mathbb{Q}^+$ e vértices $s$ e $t$, encontrar um fluxo de $s$ para $t$ de valor máximo.

# Formulação Linear

- Para colocar no formato do problema de fluxo de custo mínimo, adicione uma aresta $t \to s$.



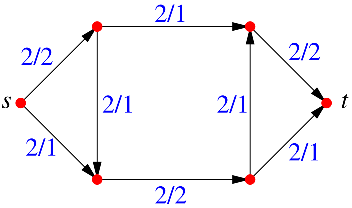

G(V,E) e capacidades

$$
\begin{array}{ll}
\text{maximize} & x_{ts} \\
\text{sujeito a} & \left\{ \begin{array}{rcl} \sum_{e \in \delta^+(v)} x_e - \sum_{e \in \delta^-(v)} x_e & = & 0 \quad \forall v \in V \\ 0 \;\leq\; x_e & \leq & c_e \quad \forall e \in E. \end{array} \right.
\end{array}
$$

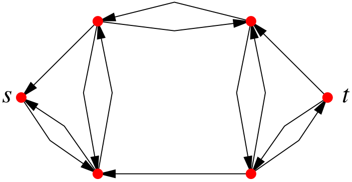
**Corolário:** *Se as capacidades $c_e$ são inteiras, então os vértices do poliedro do fluxo são inteiros.*

# Fluxo Máximo por Caminhos Aumentadores

**Grafo Residual** Dado um grafo orientado $G = (V, E)$, capacidades $c : E \to \mathbb{Q}^+$, vértices $s$ e $t$ e fluxo $f$ de $s$ para $t$, o grafo residual é o grafo $G_f = (V, A)$ onde $(u, v) \in A$ sse $(u, v) \in E$ e $f(u, v) < c(u, v)$ ou $(v, u) \in E$ e $f(u, v) > 0$.




Cap/Fluxo (de valor 3)




Grafo residual

# Fluxo Máximo por Caminhos Aumentadores

**Def.:** *Dado um fluxo* $f$*, um caminho aumentador de* $s$ *a* $t$ *é uma sequência de vértices e arestas* $P = (v_1, e_1, v_2, e_2, \ldots, v_{k-1}, e_{k-1}, v_k)$ *onde*

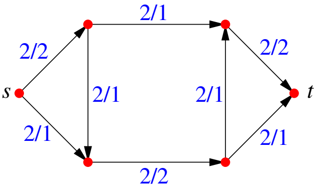
- $s = v_1$ *e* $t = v_k$ *(o caminho vai de* $s$ *a* $t$*)*
- $e_i = (v_i, v_{i+1})$ *ou* $e_i = (v_{i+1}, v_i)$
- *Se* $e_i = (v_i, v_{i+1})$ *então* $f(e) < c(e)$
  *(é possível mandar mais fluxo pela aresta* $e$*, aumentando o fluxo que chega em* $v_{i+1}$*)*
- *Se* $e_i = (v_{i+1}, v_i)$ *então* $f(e) > 0$
  *(é possível diminuir o fluxo pela aresta* $e$*, aumentando o fluxo que chega em* $v_{i+1}$*)*

Caminho aumentador dado por um caminho no grafo residual

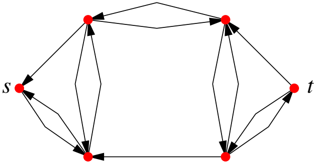
# Fluxo Máximo por Caminhos Aumentadores

Idéia: Se existir caminho aumentador de $s$ para $t$, podemos aumentar o fluxo atual.

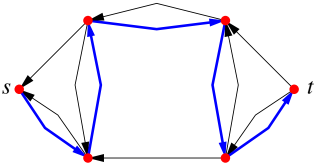
**Exemplo:** *[de Caminho aumentador]*



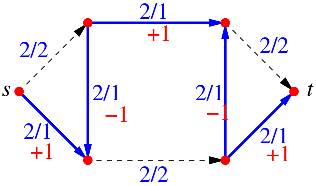

Cap/Fluxo (de valor 3)




Grafo residual




Caminho aumentador

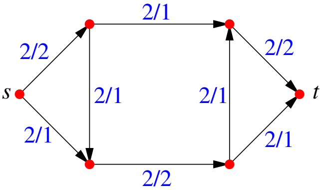



Caminho aumentador de valor 1

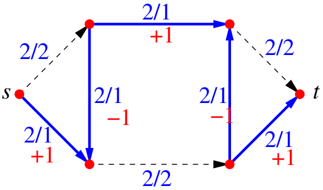
# Fluxo Máximo por Caminhos Aumentadores

Idéia: Se existir caminho aumentador de $s$ para $t$, podemos aumentar o fluxo atual.

**Exemplo:** *[de Caminho aumentador]*




Cap/Fluxo (de valor 3)




Caminho aumentador de valor 1

Dado caminho aumentador $(s = v_1, e_1, v_2, e_2, \ldots, e_{k-1}, v_k = t)$, defina $\Delta(e_i)$ como

$$\Delta(e_i) = \begin{cases} c(e_i) - f(e_i) & \text{se } e_i = (v_i, v_{i+1}) \\ f(e_i) & \text{se } e_i = (v_{i+1}, v_i) \end{cases}$$

O valor do caminho aumentador é o valor $\min\{\Delta(e_i) : 1 \leq i < k\}$.

# Fluxo Máximo por Caminhos Aumentadores

Dado fluxo $f$, como econtrar um caminho aumentador ?

- Use um algoritmo de busca em largura ou profundidade no grafo residual
- Considerando que cada vértice enxerga os arcos que saem e os que entram nele
- Partindo de $s$, se a busca atingir $t$ obtenha o valor do caminho aumentador

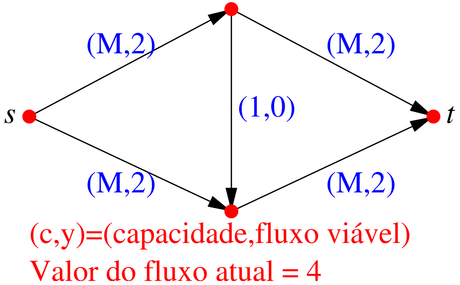
## Ford-Fulkerson

Faça $f(e) \leftarrow 0$ para todo $e \in E$.
Enquanto existir caminho aumentador $P$ faça
    Seja $P = (v_1, e_1, v_2, e_2, \ldots, v_{k-1}, e_{k-1}, v_k)$
    Seja $\Delta(P)$ o valor do caminho aumentador $P$
    Faça $f(e_i) \leftarrow \begin{cases} f(e_i) + \Delta(P) & \text{se } e_i = (v_i, v_{i+1}) \\ f(e_i) - \Delta(P) & \text{se } e_i = (v_{i+1}, v_i) \end{cases}$
    para $i = 1, \ldots, k-1$
devolva $f$

Resolvemos o Problema do Fluxo Máximo, usando várias resoluções do Problema da Busca em Grafos (Largura ou Profundidade).

**Proposição:** *O algoritmo de Ford-Fulkerson não é de tempo polinomial.*

Considere o grafo seguinte, com capacidade $M$ nas arestas, exceto na aresta do meio, que tem capacidade 1:




A cada iteração, use o caminho aumentador que usa a aresta de capacidade 1. Valor do fluxo máximo: $2M$.

Número de iterações: $O(M)$.

# Estratégia Edmonds-Karp

## Algoritmo de Edmonds-Karp

- Refinamento do algoritmo Ford-Fulkerson
- Caminhos aumentadores obtidos por busca em largura no grafo residual

**Teorema:** *O algoritmo de Edmonds-Karp, obtém um fluxo de valor máximo em tempo $O(VE^2)$.*

# *st*-Corte Mínimo

Seja $G = (V, E)$ grafo orientado, capacidades $c : E \to \mathbb{Q}^+$ e vértices $s, t$

**Def.:** Um $st$-corte é um conjunto $S \subseteq V$ tal que $s \in S$ e $t \in \overline{S}$ (onde $\overline{S} := V \setminus S$). Também é denotado pelo conjunto de arestas $\delta(S)$.

**Def.:** Definimos a capacidade de um $st$-corte $S$, como sendo a soma

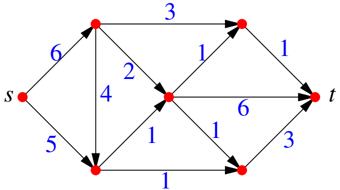
$$c(\delta(S)) := \sum_{e \in \delta(S))} c_e.$$

**Problema** *st*-CORTE MÍNIMO:
Dado grafo orientado $G = (V, E)$, capacidades $c : E \to \mathbb{Q}^+$ e vértices $s, t \in V$, encontrar um $st$-corte de capacidade mínima.

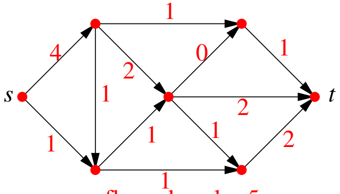
**Aplicações importantes na Computação:** Projeto de redes de conectividade, Detecção de congestionamentos em redes de conectividade, Classificação de dados (data mining), Clusterização, particionamento de circuitos VLSI, etc

Claramente, um *st*-corte limita superiormente o valor de um fluxo máximo. De fato, vale que:

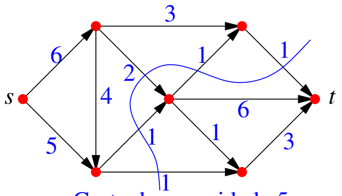
**Teorema:** *O valor de um st-fluxo máximo de s para t é igual à capacidade de um st-corte mínimo.*




G(V,E,c)




fluxo de valor 5




Corte de capacidade 5

# Redução do Corte Mínimo para Fluxo Máximo

**Def.:** *Um st-corte é um conjunto $S \subset V$ tal que $s \in S$ e $t \notin S$.*
**Def.:** *Definimos a capacidade de um st-corte $S$, como sendo a soma*

$$c(\delta^+(S)) := \sum_{e \in \delta^+(S)} c_e.$$

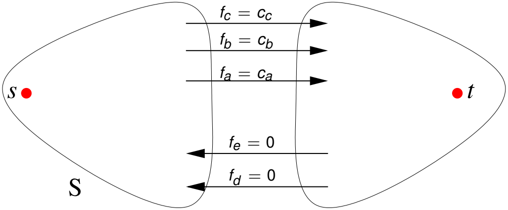
**Problema:** *[do st-Corte Mínimo] Dado um grafo orientado $G = (V, E)$, capacidades $c : E \rightarrow \mathbb{Q}^+$ e vértices $s$ e $t$, encontrar um st-corte de capacidade mínima.*

# Redução do Corte Mínimo para Fluxo Máximo

**Teorema:** *O algoritmo de Ford-Fulkerson, obtém um fluxo de valor máximo.*

*Prova.* Exercício. Dicas:

- Considere a última busca realizada para tentar obter um caminho aumentador.
- Esta busca partiu de $s$, mas não atingiu $t$.
- Seja $S$ os vértices atingidos pela busca.
- Mostre que $S$ é um corte com capacidade igual ao valor do fluxo encontrado, com a seguinte cara:




□

# Fluxo Máximo $\times$ Corte Mínimo

Do teorema anterior, conclua o seguinte teorema:

**Teorema:** *[do Fluxo Máximo-Corte Mínimo] O valor do fluxo máximo de $s$ para $t$ é igual à capacidade do menor corte separando $s$ e $t$.*

*Prova.* Exercício. Dica: Use a prova que o Algoritmo Ford-Fulkerson obtém fluxo com valor igual ao de um corte. $\square$

**Exercício:** *Reduza o problema do $st$-Corte Mínimo para o problema do Fluxo de Valor Máximo.*

## Exercício: Apresente algoritmos de tempo polinomial:

**Problema:** *[Caminhos Disjuntos em Grafos orientados] Dado grafo orientado e vértices $s$ e $t$, encontrar o maior número de $st$-caminhos orientados disjuntos nos arcos.*

**Problema:** *[Corte Separador de $s$ e $t$ em grafos orientados] Dado grafo orientado e vértices $s$ e $t$, encontrar o menor número de arcos cuja remoção desconecta todos os $st$-caminhos.*

**Problema:** *[Caminhos Disjuntos em Grafos não-orientados] Dado grafo não-orientado e vértices $s$ e $t$, encontrar o maior número de $st$-caminhos disjuntos nas arestas.*

**Problema:** *[Corte Separador de $s$ e $t$ em grafos não-orientados] Dado grafo não-orientado e vértices $s$ e $t$, encontrar o menor número de arestas cuja remoção desconecta todos os $st$-caminhos.*

# Caminhos Disjuntos e Teoremas de Menger

## Exercícios: Apresente algoritmos de tempo polinomial:

**Problema:** *[Caminhos Disjuntos nos Vértices em Grafos orientados] Dado grafo orientado e vértices $s$ e $t$, encontrar o maior número de $st$-caminhos orientados disjuntos nos vértices.*

**Problema:** *[Corte de Vértices que separa $s$ e $t$ em grafos orientados] Dado grafo orientado e vértices $s$ e $t$, encontrar o menor número de vértces cuja remoção desconecta todos os $st$-caminhos.*

**Problema:** *[Caminhos Disjuntos nos Vértices em Grafos não-orientados] Dado grafo não-orientado e vértices $s$ e $t$, encontrar o maior número de $st$-caminhos disjuntos nos vértices.*

**Problema:** *[Corte de Vértices que separa $s$ e $t$ em grafos não-orientados] Dado grafo não-orientado e vértices $s$ e $t$, encontrar o menor número de vértices cuja remoção desconecta todos os $st$-caminhos.*

Exercícios:

- Faça um programa que, dado um *st*-fluxo de valor máximo, encontra um *st*-corte mínimo em tempo linear.
- Dado um grafo *não orientado*, com capacidades nas arestas, indicando a capacidade de fluxo que pode ir de um extremo ao outro de uma aresta, em qualquer uma das direções. O problema do fluxo máximo de um vértice $s$ a um vértice $t$ é encontrar um fluxo que sai de $s$ e chega em $t$, respeitando a capacidade nas arestas e a lei de conservação do fluxo em cada vértice interno. Reduza este problema ao problema de encontrar um *st*-fluxo máximo em grafo orientado.
- Dado um grafo não orientado com capacidades nas arestas, e dois vértices $s$ e $t$, resolva o problema de encontrar um corte de capacidade mínima separando $s$ e $t$.

# Caminhos Disjuntos e Teoremas de Menger

## Exercício: Apresente algoritmos de tempo polinomial

**Def.:** *Um grafo conexo não orientado $G$ é $k$-aresta conexo (resp. $k$-vértice conexo) se a remoção de menos que $k$ arestas (resp. vértices) quaisquer nos dá um grafo conexo.*

**Problema:** *Mostre que um grafo não orientado é $k$-aresta conexo (resp. $k$-vértice conexo) se e somente se há $k$ caminhos aresta disjuntos (resp. vértice disjuntos) entre quaisquer par de vértices.*

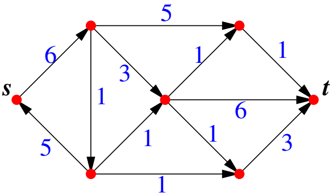
**Problema:** *Decida se um grafo não orientado é $k$-aresta conexo (resp. $k$-vértice conexo), caso contrário, apresente conjunto com menos que $k$ arestas (resp. vértices) cuja remoção nos dá um grafo desconexo.*

**Problema:** *Formule a versão orientada do problema acima e proponha um algoritmo que o resolva em tempo polinomial.*

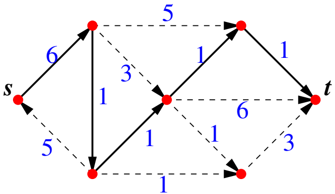
Um excelente livro sobre problemas em fluxo é dado em Ahuja, Magnanti e Orlin; Network Flows: Theory, Algorithms and Applications

# Problema do Caminho Mínimo

**Problema** CAMINHO MÍNIMO: Dado um grafo orientado $G = (V, E)$, custos nas arestas $w : E \to \mathbb{Q}^+$ e vértices $s$ e $t$, encontrar um caminho de $s$ para $t$ de custo mínimo.



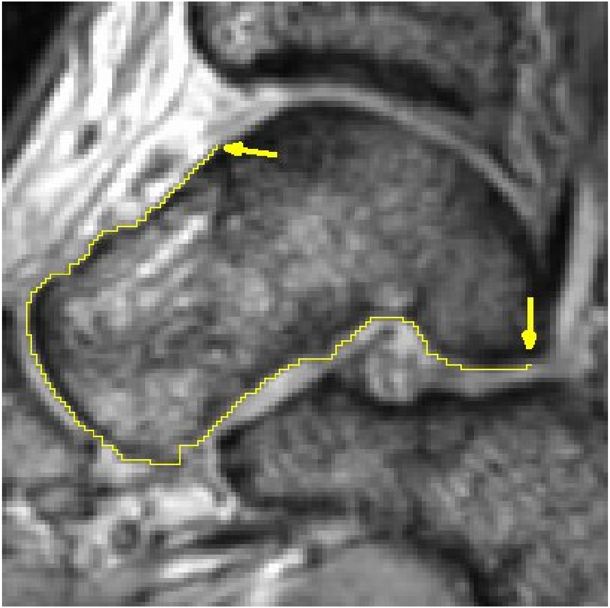
G(V,E,w)



Um caminho minimo de s para t

**Aplicações:** Determinação de rotas de custo mínimo, segmentação de imagens, Escolha de centro distribuidor, reconhecimento de fala, etc

# Exemplo de aplicação em segmentação de imagens



# Formulação para o Problema do Caminho Mínimo

Considere a seguinte formulação:

$$
(P) \quad \begin{array}{ll}
\text{minimize} & \sum_{e\in E} w_e x_e \\
\text{sujeito a} & \left\{ \begin{array}{rcll}
\sum_{e\in\delta^+(v)} x_e - \sum_{e\in\delta^-(v)} x_e & = & 0 & \forall v \in V \setminus \{s,t\} \\
\sum_{e\in\delta^+(s)} x_e - \sum_{e\in\delta^-(s)} x_e & = & 1 & \\
\sum_{e\in\delta^+(t)} x_e - \sum_{e\in\delta^-(t)} x_e & = & -1 & \\
0 \quad \leq \quad x_e & \leq & 1 & \forall e \in E.
\end{array} \right.
\end{array}
$$

- Formulação é caso particular do problema do fluxo de custo mínimo

**Corolário:** *Se $x$ é um ponto extremal ótimo de $(P)$, então $x$ é inteiro. I.e., $x_e \in \{0,1\} \forall e \in E$.*

**Teorema:** *Se $x$ é um ponto extremal ótimo de $(P)$, então as arestas $e \in E$ onde $x_e = 1$ formam um caminho mínimo ligando $s$ a $t$.*

# Caminhos Disjuntos de Custo Mínimo

**Exercícios:**

1. Considere a seguinte formulação para o problema do caminho mínimo:

$$
(P) \quad
\begin{array}{ll}
\min & \sum_{e \in E} w_e x_e \\
\text{s.a.} & \left\{
\begin{array}{rcll}
\sum_{e \in \delta^+(v)} x_e - \sum_{e \in \delta^-(v)} x_e & = & 0 & \forall v \in V \setminus \{s, t\} \\
\sum_{e \in \delta^+(s)} x_e & = & 1 & \\
\sum_{e \in \delta^-(t)} x_e & = & 1 & \\
0 \leq x_e & \leq & 1 & \forall e \in E \setminus (\delta^-(s) \cup \delta^+(t)) \\
0 \leq x_e & \leq & 0 & \forall e \in \delta^-(s) \cup \delta^+(t)
\end{array}
\right.
\end{array}
$$

   Mostre que o poliedro associado a esta formulação é inteiro.

2. Se $x$ é um ponto extremal ótimo de $(P)$, então as arestas $e \in E$ onde $x_e = 1$ formam um caminho mínimo ligando $s$ a $t$.

# Caminhos Disjuntos de Custo Mínimo

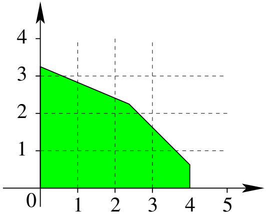
Exercícios:

1. Considere um grafo não-orientado $G = (V, E, w)$, com pesos nas arestas $w : E \to \mathbb{Q}^+$, vértices $s$ e $t$ e inteiro positivo $k$. Projete um algoritmo de tempo polinomial que encontra $k$ caminhos disjuntos nas arestas, $P_1, \ldots, P_k$, que ligam o vértice $s$ ao vértice $t$, e cujo peso total das arestas é mínimo; ou, se for o caso, escreva que não existem estes $k$ caminhos.
2. Descreva o problema do exercício anterior, mas para grafos orientados. Projete um algoritmo de tempo polinomial para resolvê-lo.
3. Descreva os problemas dos dois exercícios anteriores, mas para caminhos disjuntos nos vértices internos (os caminhos só se interseptam nos vértices $s$ e $t$).

**Def.:** *Dado um poliedro $P$, definimos seu fecho inteiro, $P_I$, como sendo o fecho convexo dos vetores inteiros de $P$. I.e.,*

$$P_I := conv\{x \in P : \ x \text{ é inteiro}\}$$

**Exemplo:** *Exemplo de poliedro $P$, conjunto $I$ dos pontos inteiros em $P$ e o fecho convexo de $I$.*



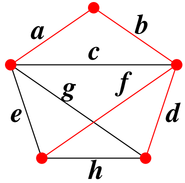

Poliedro $P$     Pontos inteiros $I$     $P_I := conv(I)$

**Def.:** *Dado um conjunto finito e ordenado $E$, dizemos que $\chi^A := (\chi_e^A : e \in E)$ é o vetor de incidência de $A$, $A \subseteq E$; onde $\chi_e^A = 1$ se $e \in A$ e $0$ caso contrário.*

**Exemplo:** Se $E = (a, b, c, d, e, f, g)$ e $A = \{a, c, e, f\}$ e $\chi^A$ é um vetor de incidência de $A$ em $E$

$$\chi^A = (1, 0, 1, 0, 1, 1, 0).$$

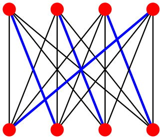
**Exemplo:** Seja $G = (V, E)$ o grafo abaixo com ordenação das arestas $E = (a, b, c, d, e, f, g, h)$ e $T$ o subgrafo definido pelas arestas em vermelho.




Então o vetor de incidência das arestas de $T$ em $E$ é $\chi^T = (1, 1, 0, 1, 0, 1, 0)$.

# Emparelhamento em Grafos Bipartidos

**Def.:** Dado um grafo $G = (V, E)$, dizemos que $M \subseteq E$ é um emparelhamento de $G$ se $M$ não tem arestas com extremos em comum.



**Problema** EMPARELHAMENTO-BIPARTIDO: Dados um grafo bipartido $G = (V, E)$, e custos nas arestas $c : E \to \mathbb{Z}$, encontrar emparelhamento $M \subseteq E$ que maximize $c(M)$.

**Aplicações:** Encontrar atribuição de candidatos para vagas maximizando aptidão total, atribuição de motoristas de veículos, formação de equipes (eg. com um chefe e subordinados), etc.

**Exercício:** *Reduza o problema de encontrar um Emparelhamento de Cardinalidade Máxima em grafos bipartidos ao Problema do Fluxo Máximo. Mostre que a aplicação do Algoritmo Ford-Fulkerson nos dá um algoritmo de tempo polinomial para este problema.*

# Formulação para Emparelhamento em Grafos Bipartidos

## Formulação em Programação Linear Inteira

$$
\begin{array}{ll}
\text{maximize} & \displaystyle\sum_{e \in E} c_e x_e \\
\text{sujeito a} & \left\{
\begin{array}{rcll}
\displaystyle\sum_{e \in \delta(v)} x_e & \leq & 1 & \forall v \in V \\
0 \leq x_e & \leq & 1 & \forall e \in E \\
x_e \ \text{inteiro} & & & \forall e \in E
\end{array}
\right.
\end{array}
$$

Note que a formulação acima só não é um programa linear, devido a restrição de integralidade em $x_e$.

# Relaxação Linear

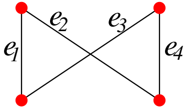
$$
(LP_{Emp}) \qquad \begin{array}{ll} \text{maximize} & \displaystyle\sum_{e \in E} c_e x_e \\ \text{sujeito a} & \left\{ \begin{array}{rcccll} & \displaystyle\sum_{e \in \delta(v)} x_e & \leq & 1 & \forall v \in V \\ 0 & \leq & x_e & \leq & 1 & \forall e \in E \end{array} \right. \end{array}
$$

**Teorema:** *Os vértices do poliedro $LP_{Emp}$ são inteiros.*

*Prova.* Exercício. □

# Exemplo:

Considere o seguinte grafo bipartido com custos unitários:




## Programa Linear

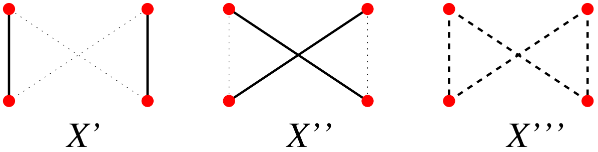
$$
\begin{array}{ll}
\text{maximize} & x_{e_1} + x_{e_2} + x_{e_3} + x_{e_4} \\
\text{sujeito a} & \left\{ \begin{array}{l}
x_{e_1} + x_{e_2} \leq 1, \\
x_{e_3} + x_{e_4} \leq 1, \\
x_{e_1} + x_{e_3} \leq 1, \\
x_{e_2} + x_{e_4} \leq 1, \\
0 \leq x_{e_1} \leq 1, \\
0 \leq x_{e_2} \leq 1, \\
0 \leq x_{e_3} \leq 1, \\
0 \leq x_{e_4} \leq 1
\end{array} \right.
\end{array}
$$

# Variáveis do programa linear correspondente:

$$X = (x_{e_1}, x_{e_2}, x_{e_3}, x_{e_4})$$

Os seguintes vetores são soluções ótimas do programa linear:

$$X' = (1,0,0,1) \quad X'' = (0,1,1,0) \quad X''' = (\frac{1}{2},\frac{1}{2},\frac{1}{2},\frac{1}{2})$$




- $X'$ e $X''$ são vértices do poliedro do emparelhamento
- $X'''$ é solução ótima, mas é combinação convexa de $X'$ e $X''$ ($X''' = \frac{1}{2}X' + \frac{1}{2}X''$). I.e., $X'''$ não é vértice de $P_{Emp}$.

# Problema do Transporte

**Problema** PROBLEMA DO TRANSPORTE: São dados um conjunto $A$ de fornecedores e um conjunto $B$ de consumidores, onde $A = \{1, \ldots, m\}$ e $B = \{1, \ldots, n\}$, todos relativos a um mesmo material. O fornecedor $i \in A$ produz $a_i$ unidades e o consumidor $j \in B$ consome $b_j$ unidades. O custo para transportar uma unidade do fornecedor $i$ para o consumidor $j$ é de $c_{ij}$, para todo $i \in A$ e $j \in B$. Considere que $T = \sum_i a_i = \sum_j b_j$. O problema consiste em determinar a forma mais barata para transportar $T$ unidades dos fornecedores para os consumidores, satisfazendo a produção e consumo dos produtores e consumidores.

**Aplicações:** Transporte de materiais, abastecimento (energia, água,...), transmissão de dados/broadcast

# Problema do Transporte

**Exercício:** Resolva o Problema do Transporte.

**Exercício:** Resolva uma variação do Problema do Transporte (capacitado nas arestas) supondo que o fornecimento $x_{ij}$ é no máximo um valor $u_{ij}$, dado na entrada, para todo $i \in A$ e $j \in B$.

**Exercício:** Resolva uma variação do Problema do Transporte supondo que cada aresta tem uma capacidade máxima, $\sum_{i \in A} a_i \geq \sum_{j \in B} b_j$ (não necessariamente valendo na igualdade) devemos atender todos os consumidores, mas os fornecedores só precisam respeitar capacidade de produção (não necessariamente, a produção de um fornecedor $i$ precisa ser consumida).

**Exercício:** $b$-matching: Dados grafo bipartido $G = (A, B, E)$ com função de peso nas arestas $w : E \to \mathbb{Q}$, e função limitadora de grau $b : A \cup B \to Z^+$, encontrar um conjunto de arestas $M$ tal que o número de arestas de $M$ incidentes a um vértice $i$ é no máximo $b_i$ e o peso total das arestas de $M$ é máximo.

# Exercícios

**Exercício:** Dados grafo bipartido $G = (V, E)$ com função de peso nos vértices $w : V \to \mathbb{Q}$, encontrar um conjunto de vértices $I \subseteq V$ que não tem arestas incidentes em comum (conjunto independente). Apresente uma formulação em programação linear inteira para resolver este problema, cuja relaxação apresente apenas vértices inteiros. Com isso, projete um algoritmo de tempo polinomial para encontrar um conjunto independente em $G$ de peso máximo.

**Exercício:** Dados grafo bipartido $G = (V, E)$ com função de peso nos vértices $w : V \to \mathbb{Q}$, encontrar um conjunto de vértices $C \subseteq V$ tal que toda aresta $e \in E$ tem pelo menos um dos extremos em $C$ (cobertura por vértices). Faça como no exercício anterior e projete um algoritmo de tempo polinomial para encontrar uma cobertura por vértices em $G$ de peso mínimo. Outra maneira de resolver o problema da cobertura por vértices, é reduzindo este problema para o anterior. Mostre como isto pode ser feito.

# Exercícios

**Exercício:** Dados grafo bipartido $G = (V, E)$ com função de peso nos vértices $w : V \to \mathbb{Q}$, encontrar um conjunto de arestas $F \subseteq E$ tal que para todo vértice $v \in V$, há pelo menos uma aresta de $F$ incidente em $v$ (cobertura por arestas). Faça como no exercício anterior e projete um algoritmo de tempo polinomial para encontrar uma cobertura por vértices em $G$ de peso mínimo.

# Dualidade em Programação Linear

Considere o seguinte PL:

$$
\begin{array}{ll}
\text{minimize} & c_1x_1 + c_2x_2 + c_3x_3 \\
\text{sujeito a} & \left\{
\begin{array}{lcl}
a_{11}x_1 + a_{12}x_2 + a_{13}x_3 & \geq & b_1 \\
a_{21}x_1 + a_{22}x_2 + a_{23}x_3 & = & b_2 \\
a_{31}x_1 + a_{32}x_2 + a_{33}x_3 & \leq & b_3 \\
x_1 \geq 0, \qquad\qquad x_3 \leq 0 & &
\end{array}
\right.
\end{array}
$$

Vamos delimitar o valor ótimo do LP:

Multiplique as restrições por $y_1 \geq 0$, $y_2$ e $y_3 \leq 0$:

$$
\begin{array}{lcl}
y_1(a_{11}x_1 + a_{12}x_2 + a_{13}x_3) & \geq & y_1b_1 \\
y_2(a_{21}x_1 + a_{22}x_2 + a_{23}x_3) & = & y_2b_2 \\
y_3(a_{31}x_1 + a_{32}x_2 + a_{33}x_3) & \geq & y_3b_3
\end{array}
$$

Somando estas inequações obtemos:

$$
\begin{array}{rcl}
y_1(a_{11}x_1 + a_{12}x_2 + a_{13}x_3) & \geq & y_1 b_1 \\
y_2(a_{21}x_1 + a_{22}x_2 + a_{23}x_3) & = & y_2 b_2 \\
y_3(a_{31}x_1 + a_{32}x_2 + a_{33}x_3) & \geq & y_3 b_3 \\
\hline
(y_1 a_{11} + y_2 a_{21} + y_3 a_{31})x_1 + & & \\
(y_1 a_{12} + y_2 a_{22} + y_3 a_{32})x_2 + & & \\
(y_1 a_{13} + y_2 a_{23} + y_3 a_{33})x_3 & \geq & (y_1 b_1 + y_2 b_2 + y_3 b_3) = yb
\end{array}
$$

Comparando com a função objetivo $cx = c_1x_1 + c_2x_2 + c_3x_3$, se

$$c_1x_1 \geq (y_1 a_{11} + y_2 a_{21} + y_3 a_{31})x_1$$

$$c_2x_2 = (y_1 a_{12} + y_2 a_{22} + y_3 a_{32})x_2$$

$$c_3x_3 \geq (y_1 a_{13} + y_2 a_{23} + y_3 a_{33})x_3$$

Nestas condições, temos $cx \geq yb$ e portanto $yb$ é limitante de $cx$

Como $x_1 \geq 0$ e $x_3 \leq 0$, podemos simplificar as condições para ter $cx \geq yb$ como:

$$c_1 \geq y_1 a_{11} + y_2 a_{21} + y_3 a_{31}$$

$$c_2 = y_1 a_{12} + y_2 a_{22} + y_3 a_{32}$$

$$c_3 \leq y_1 a_{13} + y_2 a_{23} + y_3 a_{33}$$

Naturalmente, queremos dentre todos os valores de $y$, o que melhor delimita $cx \geq yb$, i.e., com $yb$ máximo. Assim, obtemos o seguinte problema dual:

$$\begin{array}{ll} \text{maximize} & y_1 b_1 + y_2 b_2 + y_3 b_3 \\ \text{sujeito a} & \left\{ \begin{array}{l} y_1 a_{11} + y_2 a_{21} + y_3 a_{31} \leq c_1 \\ y_1 a_{12} + y_2 a_{22} + y_3 a_{32} = c_2 \\ y_1 a_{13} + y_2 a_{23} + y_3 a_{33} \geq c_3 \\ y_1 \geq 0, \qquad\qquad y_3 \leq 0 \end{array} \right. \end{array}$$

Convenção para nomear estes sistemas:

**Problema Primal**

$$
\begin{array}{ll}
\text{minimize} & c_1 x_1 + c_2 x_2 + c_3 x_3 \\
\text{sujeito a} & \left\{
\begin{array}{lcl}
a_{11}x_1 + a_{12}x_2 + a_{13}x_3 & \geq & b_1 \\
a_{21}x_1 + a_{22}x_2 + a_{23}x_3 & = & b_2 \\
a_{31}x_1 + a_{32}x_2 + a_{33}x_3 & \leq & b_3 \\
x_1 \geq 0, \qquad\qquad x_3 \leq 0 & &
\end{array}
\right.
\end{array}
$$

**Problema Dual**

$$
\begin{array}{ll}
\text{maximize} & y_1 b_1 + y_2 b_2 + y_3 b_3 \\
\text{sujeito a} & \left\{
\begin{array}{l}
y_1 a_{11} + y_2 a_{21} + y_3 a_{31} \leq c_1 \\
y_1 a_{12} + y_2 a_{22} + y_3 a_{32} = c_2 \\
y_1 a_{13} + y_2 a_{23} + y_3 a_{33} \geq c_3 \\
y_1 \geq 0, \qquad\qquad y_3 \leq 0
\end{array}
\right.
\end{array}
$$

**Exercício:** *Faça exatamente a mesma análise feita anteriormente para se obter um dual (como limitante de um programa linear), mas em vez de partir de um problema de minimização, comece com um problema de maximização, e obtenha seu dual como um problema de minimização.*

**Exercício:** *Faça o programa dual da formulacao relaxada dos seguintes problemas:*

- *Problema da Cobertura de Vertices*
- *Problema da Cobertura por Conjuntos*
- *Problema de Localizacao de Facilidades*
- *Problema de Steiner*

# Problemas Primal e Dual

## Problema Primal

$$
(P) \qquad
\begin{array}{lrcll}
\text{minimize} & c\,x & & & \\
\text{sujeito a} & (Ax)_i & \geq & b_i & \text{para cada } i \text{ em } M_1\,, \\
 & (Ax)_i & = & b_i & \text{para cada } i \text{ em } M_2\,, \\
 & (Ax)_i & \leq & b_i & \text{para cada } i \text{ em } M_3\,, \\
 & x_j & \geq & 0 & \text{para cada } j \text{ em } N_1\,, \\
 & x_j & \leq & 0 & \text{para cada } j \text{ em } N_3\,.
\end{array}
$$

## Problema Dual

$$
(D) \qquad
\begin{array}{lrcll}
\text{maximize} & y\,b & & & \\
\text{sujeito a} & (yA)_j & \leq & c_j & \text{para cada } j \text{ em } N_1\,, \\
 & (yA)_j & = & c_j & \text{para cada } j \text{ em } N_2\,, \\
 & (yA)_j & \geq & c_j & \text{para cada } j \text{ em } N_3\,, \\
 & y_i & \geq & 0 & \text{para cada } i \text{ em } M_1\,, \\
 & y_i & \leq & 0 & \text{para cada } i \text{ em } M_3\,.
\end{array}
$$

**Lema:** *Seja $(P)$ um programa primal de minimização e $(D)$ seu problema dual de maximização, então $cx \geq yb$ para todo $x \in P$ e $y \in D$.*

## Folgas Complementares

Dois vetores $x$ e $y$, indexados por $M$ e $N$ respectivamente, têm folgas complementares se,

$x_j = 0 \quad \text{ou} \quad (yA)_j = c_j \quad \forall j \in N_1 \cup N_3$ (folgas complementares primais)
e
$y_i = 0 \quad \text{ou} \quad (Ax)_i = b_i \quad \forall i \in M_1 \cup M_3$ (folgas complementares duais).

**Lema:** *(das folgas complementares) Se $(P)$ é um programa linear e $(D)$ seu programa dual, $x$ e $y$ soluções viáveis de $(P)$ e $(D)$ então*

$$(cx = yb) \quad \Leftrightarrow \quad (x \text{ e } y \text{ satisfazem folgas complementares})$$

**Teorema:** *(da dualidade) Se $(P)$ é um programa linear de minimização e $(D)$ seu programa dual (de maximização), então vale exatamente uma das possibilidades:*

1. $(P)$ *e* $(D)$ *são viáveis e* $\text{OPT-LP}(P)$=$\text{OPT-LP}(D)$.
2. $(P)$ *é viável e* $(D)$ *é inviável e* $\text{OPT-LP}(P) = -\infty$.
3. $(P)$ *é inviável e* $(D)$ *é viável e* $\text{OPT-LP}(D) = \infty$.
4. $(P)$ *e* $(D)$ *são inviáveis.*

**Lema:** *(de Farkas) Exatamente um dos programas restritos têm solução:*

$$(RP)\quad \begin{array}{rcll} \exists & x & \in & \mathbb{Q}^N \\ (Ax)_i & \geq & b_i & \forall i \in M, \\ x_i & \geq & 0 & \forall j \in N. \end{array} \qquad (RD)\quad \begin{array}{rcll} \exists & y & \in & \mathbb{Q}^M \\ yb & > & 0 & \\ (yA)_j & \leq & 0 & \forall j \in N, \\ y_i & \geq & 0 & \forall i \in M. \end{array}$$

*Prova.*
Considere o programa linear $(P)$ e seu dual $(D)$:

$$(P)\quad \begin{array}{rrcll} \min & 0\,x & & & \\ & (Ax)_i & \geq & b_i & \forall i \in M, \\ & x_i & \geq & 0 & \forall j \in N. \end{array} \qquad (D)\quad \begin{array}{rrcll} \max & yb & & & \\ & (yA)_j & \leq & 0 & \forall j \in N, \\ & y_i & \geq & 0 & \forall i \in M. \end{array}$$

Note que $(P)$ é viável se e só se $(RP)$ é víavel e o programa $(D)$ é sempre viável pois $y = 0$ satisfaz as restrições.

Como $(D)$ é viável, apenas as alternativas 1. ou 3. do Lema da Dualidade podem ocorrer.

Caso 1: $(P)$ é viável e OPT-LP$(P)$=OPT-LP$(D)$.

$(P)$ viável $\Rightarrow$ $(RP)$ é viável.

$$\Big(\forall y \in (D),\ yb \leq \text{OPT-LP}(D) = \text{OPT-LP}(P) = 0\Big) \Rightarrow (RD) \text{ é inviável.}$$

Caso 3: OPT-LP$(D) = \infty$ e $(P)$ é inviável.

$(P)$ inviável $\Rightarrow$ $(RP)$ é inviável.

$$\Big(\text{OPT-LP}(D) = \infty\Big) \Rightarrow \Big(\exists y \in (D) :\ yb > 0\Big) \Rightarrow (RD) \text{ é viável.} \qquad \square$$

# Método Primal-Dual

- Baseado em Folgas Complementares e Problemas de Viabilidade
- Muitas vezes produz algoritmos combinatórios

# Método Primal-Dual Clássico

$$(P)\quad \begin{array}{lrcll} \min & cx & & & \\ & (Ax)_i & \geq & b_i & \forall i \in M\,, \\ & x_i & \geq & 0 & \forall j \in N\,. \end{array} \qquad (D)\quad \begin{array}{lrcll} \max & yb & & & \\ & (yA)_j & \leq & c_j & \forall j \in N\,, \\ & y_i & \geq & 0 & \forall i \in M\,. \end{array}$$

## Folgas Complementares

Se $\hat{x}$ e $\hat{y}$ são soluções ótimas de $(P)$ e $(D)$, então

$\hat{x}_j = 0$ ou $(\hat{y}A)_j = c_j$ (folgas complementares primais)

$\hat{y}_i = 0$ ou $(A\hat{x})_i = b_i$ (folgas complementares duais)

## Estratégia:

Dado vetor $y$ viável de (D),

- obter $x$ viável de $(P)$ satisfazendo folgas complementares com $y$
  **ou**
- obter $y'$ tq. $y'' \leftarrow y + y'$ é viável em $(D)$ e $y''b > yb$;
  repita processo com $y''$

## Folgas Complementares

Se $\hat{x}$ e $\hat{y}$ são soluções ótimas de $(P)$ e $(D)$, então

$$\hat{x}_j = 0 \quad \text{ou} \quad (\hat{y}A)_j = c_j \qquad \text{(folgas complementares primais)}$$
$$\hat{y}_i = 0 \quad \text{ou} \quad (Ax)_i = b_i \qquad \text{(folgas complementares duais)}$$

Dado vetor $y$ viável de (D),

$$I(y) := \{i \in M : \ y_i = 0\} \quad \text{e} \quad J(y) := \{j \in N : \ (yA)_j = c_j\} \, .$$

$$\text{(Viabil.)} \quad \begin{array}{rcll} (Ax)_i & \geq & b_i & \forall i \in M \, , \\ x_i & \geq & 0 & \forall j \in N \, . \end{array} \; + \; \text{(F.C.)} \quad \begin{array}{rcll} (Ax)_i & = & b_i & \forall i \in M \setminus I(y) \, , \\ x_j & = & 0 & \forall j \in N \setminus J(y) \, . \end{array}$$

---

Problema Restrito Primal

$$(RP) \quad \begin{array}{rcll} (Ax)_i & \geq & b_i & \forall i \in I(y) \, , \\ (Ax)_i & = & b_i & \forall i \in M \setminus I(y) \, , \\ x_j & \geq & 0 & \forall j \in J(y) \, , \\ x_j & = & 0 & \forall j \in N \setminus J(y) \, . \end{array}$$

Pelo Lema de Farkas, ($RP$) é viável ou ($RD$) é viável, não ambos (exercício).

$$
(RP)\quad \begin{array}{rcll} (Ax)_i & \geq & b_i & \forall i \in I(y)\,, \\ (Ax)_i & = & b_i & \forall i \in M \setminus I(y)\,, \\ x_j & \geq & 0 & \forall j \in J(y)\,, \\ x_j & = & 0 & \forall j \in N \setminus J(y)\,. \end{array} \oplus \quad (RD)\quad \begin{array}{rcll} y'b & > & 0 & \\ (y'A)_j & \leq & 0 & \forall j \in J(y)\,, \\ y'_i & \geq & 0 & \forall i \in I(y)\,. \end{array}
$$

$$I(y) := \{i \in M : \ y_i = 0\} \quad \text{e} \quad J(y) := \{j \in N : \ (yA)_j = c_j\}\,.$$

$$(D) \quad \begin{array}{rrcll} \max & yb & & & \\ & (yA)_j & \le & c_j & \forall j \in N\,, \\ & y_i & \ge & 0 & \forall i \in M\,. \end{array}$$

$$(RD) \quad \begin{array}{rcll} y'b & > & 0 & \\ (y'A)_j & \le & 0 & \forall j \in J(y)\,, \\ y'_i & \ge & 0 & \forall i \in I(y)\,. \end{array}$$

**Lema:** *Se $y$ é viável para $(D)$ e $y'$ é viável para $(RD)$, então, existe $\theta > 0$ tal que $y'' \leftarrow y + \theta\, y'$ é também é viável para $(D)$.*

*Prova.* Exercício. □

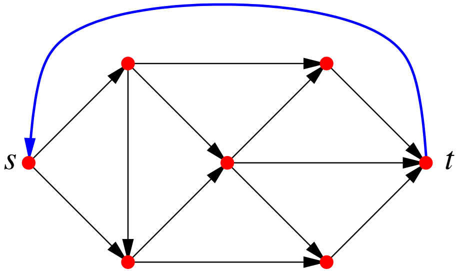
**Método** PRIMAL-DUAL $(A, b, c)$

seja $y$ um vetor viável de $(D)$
enquanto $\mathrm{RP}(A, b, y)$ não tem solução faça
  seja $y'$ uma solução do $\mathrm{RD}(A, b, y)$
  se $y + \theta y'$ em $(D)$ para todo $\theta$ positivo
    então devolva $y'$
    senão seja $\theta$ máximo tal que $y + \theta y'$ é viável em $(D)$
      $y \leftarrow y + \theta y'$
seja $x$ uma solução do $\mathrm{RP}(A, b, y)$
devolva $x$ e $y$

# Problema do Fluxo Máximo

**Problema** FLUXO-MÁXIMO: Dado um grafo orientado $G = (N, A)$, capacidades $c : A \to \mathbb{N}$ e vértices $s$ e $t$, encontrar um fluxo de valor máximo de $s$ para $t$.

Simplificação: adicionar aresta $(t, s)$ sem restrição de capacidade.




Objetivo: Encontrar fluxo que respeita conservação de fluxo em todos os vértices e maximiza fluxo na aresta $(t, s)$.

Por conveniência, vamos chamar a formulação do fluxo de Dual.

$$
(D) \quad \begin{array}{lrcll}
\max & y_{ts} & & & \\
& \displaystyle\sum_{e\in\delta^+(i)} y_e - \sum_{e\in\delta^-(i)} y_e & = & 0 & \forall i \in N, \\
& y_{ij} & \leq & c_{ij} & \forall ij \in A, \\
& y_{ij} & \geq & 0 & \forall ij \in A.
\end{array}
$$

Primal: encontrar $x = x' \| x''$, $x'$ indexado por $N$ e $x''$ indexado por $A$ que

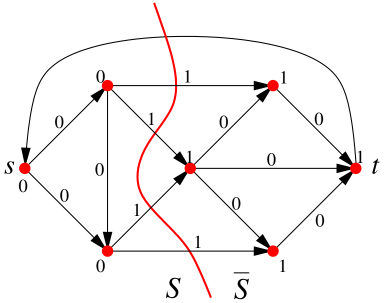
$$
(P) \quad \begin{array}{lrcll}
\min & \displaystyle\sum_{ij\in A} c_{ij} x''_{ij} & & & \\
& x'_t - x'_s & \geq & 1 & , \\
& x'_i - x'_j + x''_{ij} & \geq & 0 & \forall ij \in A, \\
& x''_{ij} & \geq & 0 & \forall ij \in A.
\end{array}
$$

## Sobre o programa *(P)*

$$(P)\quad \begin{array}{lrcl} \min & \displaystyle\sum_{ij\in A} c_{ij}\, x''_{ij} & & \\ & x'_t - x'_s & \geq & 1 \quad , \\ & x'_i - x'_j + x''_{ij} & \geq & 0 \quad \forall ij \in A, \\ & x''_{ij} & \geq & 0 \quad \forall ij \in A. \end{array}$$

Dado um corte $S \subset N$ que separa $s$ de $t$ (i.e. $s \in S$ e $t \notin S$), defina

$$x'_i = \begin{cases} 0 & \forall i \in S \\ 1 & \forall i \in N \setminus S \end{cases} \qquad x''_{ij} = \begin{cases} 1 & \forall ij \in \delta^+(S) \\ 0 & \forall ij \in A \setminus \delta^+(S) \end{cases}$$




Temos $x = x' \| x''$ viável para $(P)$ definindo um corte com capacidade dada pela função objetivo.

## Problemas Restritos

Dado $y$, fluxo viável seja

$$I := \{ij \in A : \; y_{ij} = 0\} \qquad \text{e} \qquad J := \{ij \in A : \; y_{ij} = c_{ij}\} .$$

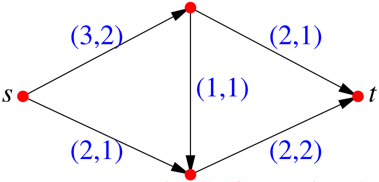
Restrito Primal: encontrar $x = x' \| x''$ viável em $(D)$ satisfazendo F.C.

$$(RP) \quad \begin{array}{rcll} x'_t - x'_s & \geq & 1 & , \\ x'_i - x'_j + x''_{ij} & = & 0 & \forall ij \in A \setminus I, \\ x'_i - x'_j + x''_{ij} & \geq & 0 & \forall ij \in I, \\ x''_{ij} & \geq & 0 & \forall ij \in J, \\ x''_{ij} & = & 0 & \forall ij \in A \setminus J. \end{array}$$

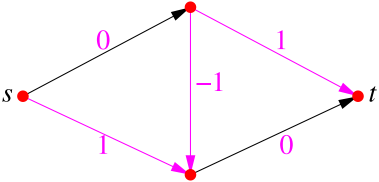
Restrito Dual: Se $(RP)$ é inviável, pelo Lema de Farkas, $(RD)$ é viável

$$(RD) \quad \begin{array}{rcll} y_{ts} & > & 0 & , \\ \displaystyle\sum_{e \in \delta^+(i)} y_e - \sum_{e \in \delta^-(i)} y_e & = & 0 & \forall i \in N, \\ y_{ij} & \leq & 0 & \forall ij \in J, \\ y_{ij} & \geq & 0 & \forall ij \in I. \end{array}$$

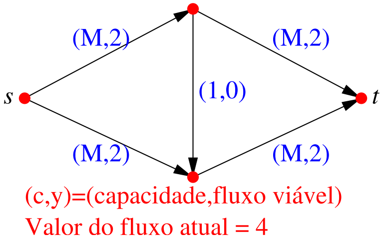
($RD$) viável $\Rightarrow \exists$ caminho aumentador $P = (y')$ de $s$ para $t$, representando um fluxo adicional.




(c,y)=(capacidade,fluxo viável)
Valor do fluxo inicial = 3




Caminho aumentador
Fluxo no caminho = 1




($RD$) inviável $\Rightarrow$ ($RP$) é viável

Seja $S := \{v \in N :$ existe caminho aumentador de $s$ para $v\}$ e

$$x'_i = \begin{cases} 0 & \forall i \in S \\ 1 & \forall i \in N \setminus S \end{cases} \qquad x''_{ij} = \begin{cases} 1 & \forall ij \in \delta^+(S) \\ 0 & \forall ij \in A \setminus \delta^+(S) \end{cases}$$

Temos que $x = x' \| x''$ é viável em ($RP$)