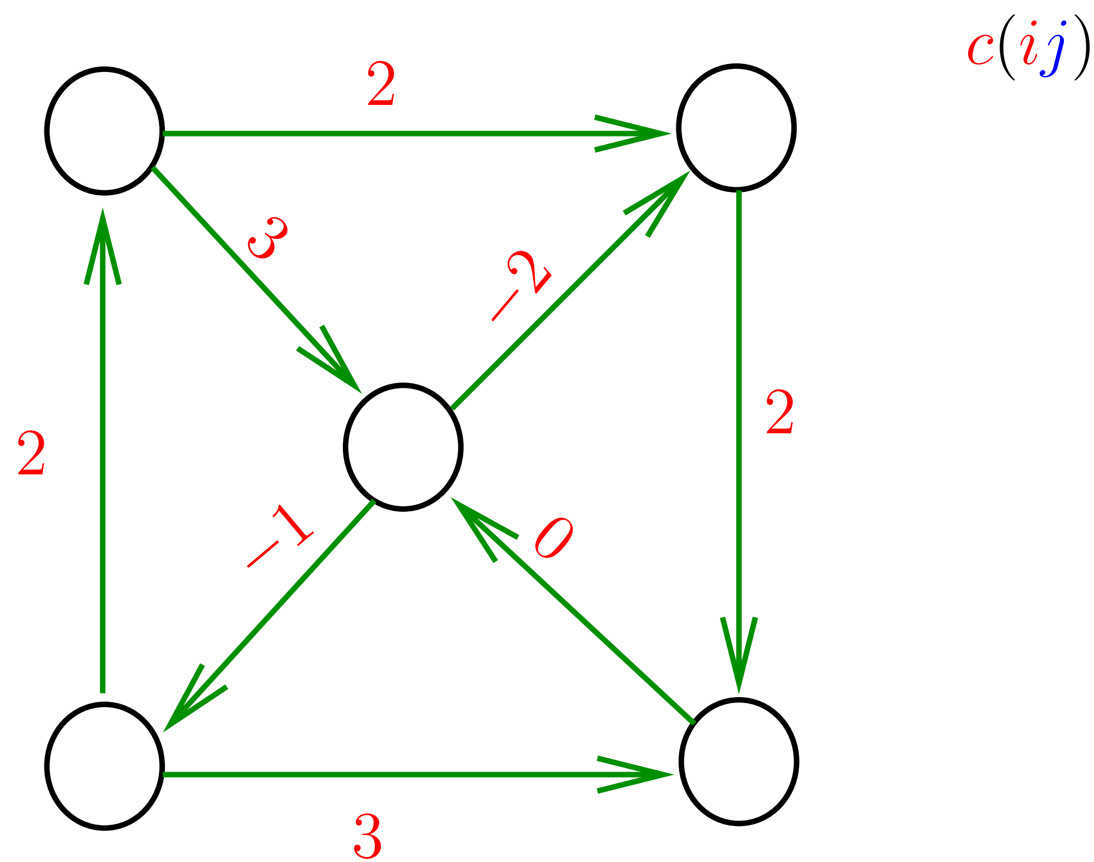
Melhores momentos

AULA PASSADA

Custo médio de um ciclo

Dada uma rede (N, A, c) com função-custo c, o custo médio de um ciclo O é o número

$$\mu(O) = \frac{c(O)}{|O|} \; .$$

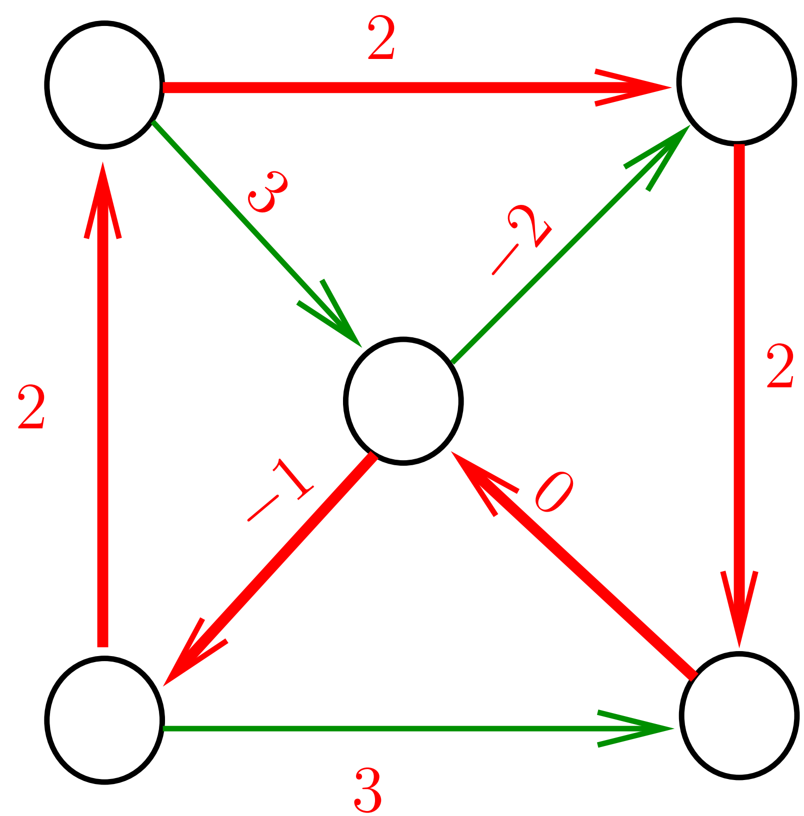


Custo médio de um ciclo

Dada uma rede (N, A, c) com função-custo c, o custo médio de um ciclo O é o número

$$\mu(O) = \frac{c(O)}{|O|}\,.$$

$c(ij)$

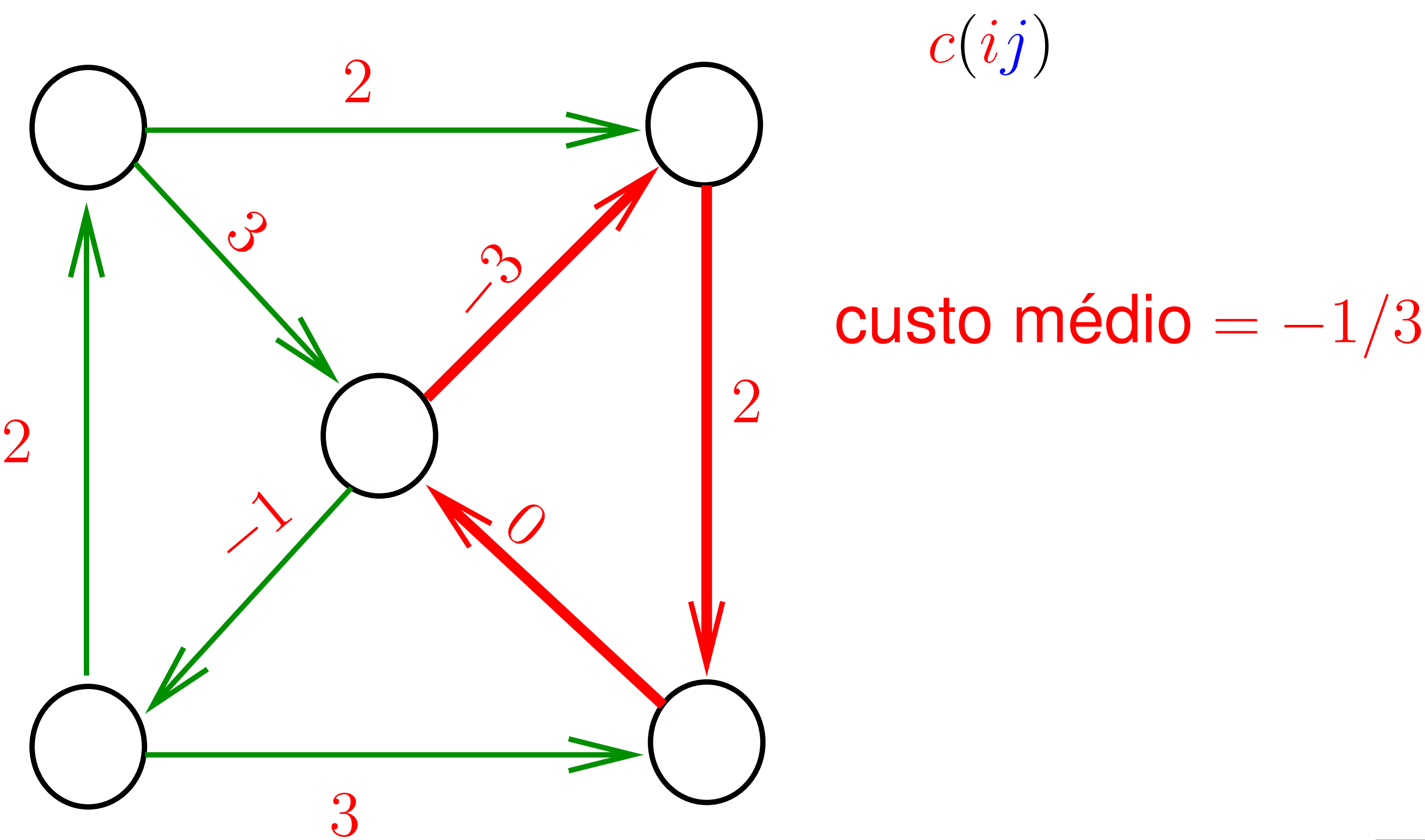


custo médio $= 1$

Custo médio de um ciclo

Dada uma rede (N, A, c) com função-custo c, o custo médio de um ciclo O é o número

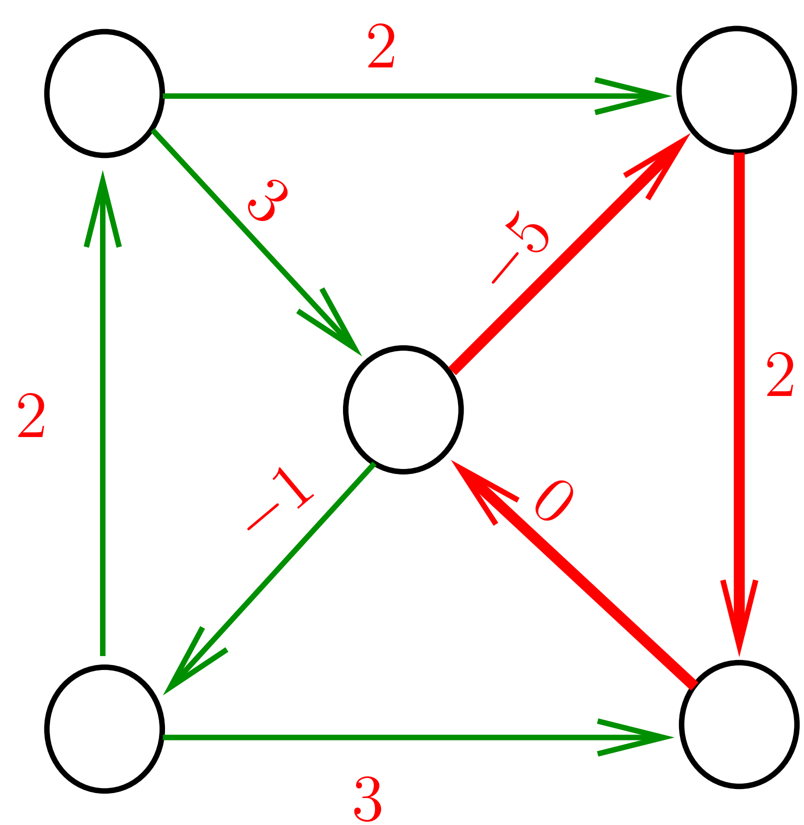
$$\mu(O) = \frac{c(O)}{|O|} .$$



Problema do custo médio mínimo

Dada uma rede (N, A, c) encontrar um ciclo de custo médio mínimo.

$$\mu_*(c) := \min\left\{\frac{c(O)}{|O|} : O \text{ é um ciclo}\right\}$$



$c(ij)$

$\mu_*(c) = -1$

Algoritmo Min-Mean-Cycle-Value

MIN-MEAN-CYCLE-DYN-PROG (N, A, c)

$\mu \leftarrow$ MIN-MEAN-CYCLE-VALUE(N, A, c)

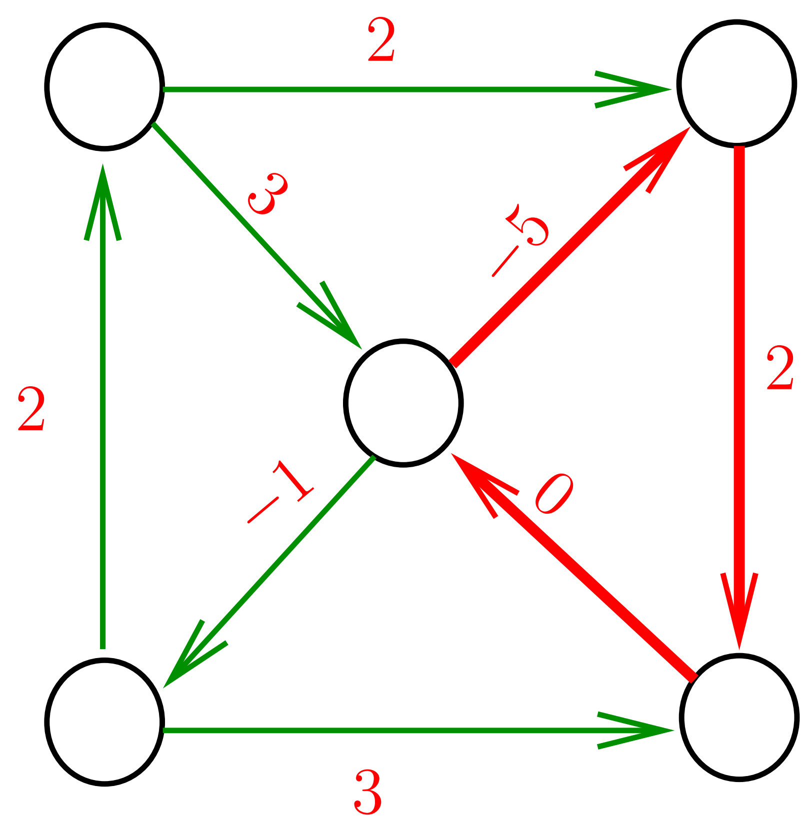
$\langle O, y\rangle \leftarrow$ CICLO-NEGATIVO$(N, A, c - \mu)$

$A' \leftarrow \{ij : y(j) - y(i) = c(ij) - \mu\}$

$\langle O', y\rangle \leftarrow$ DAG(N, A')

devolva O'

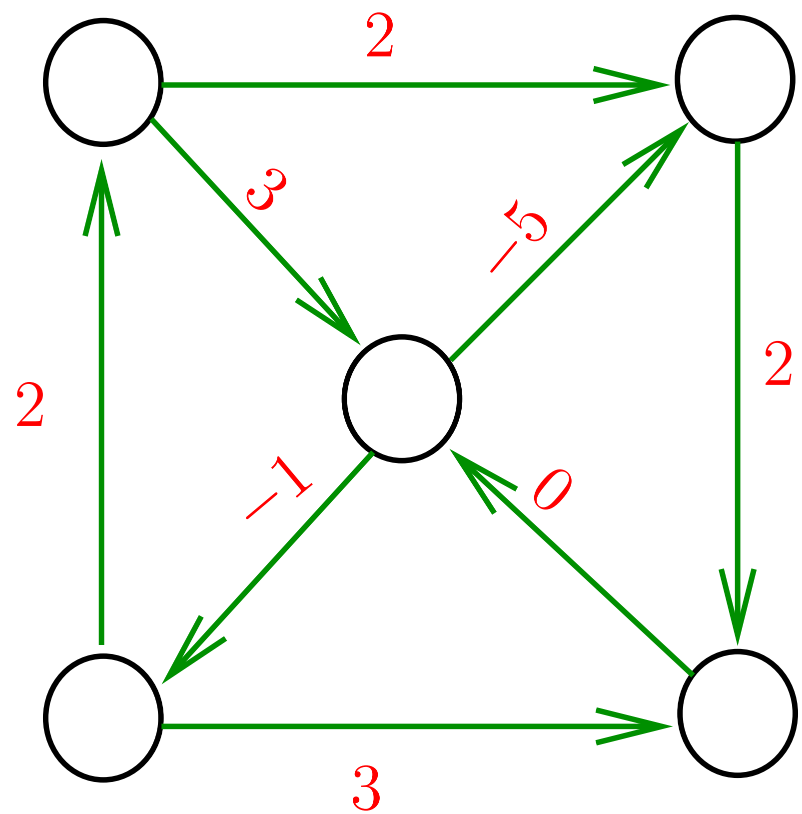
Simulação



$c(ij)$

$\mu_*(c) = -1$

Simulação

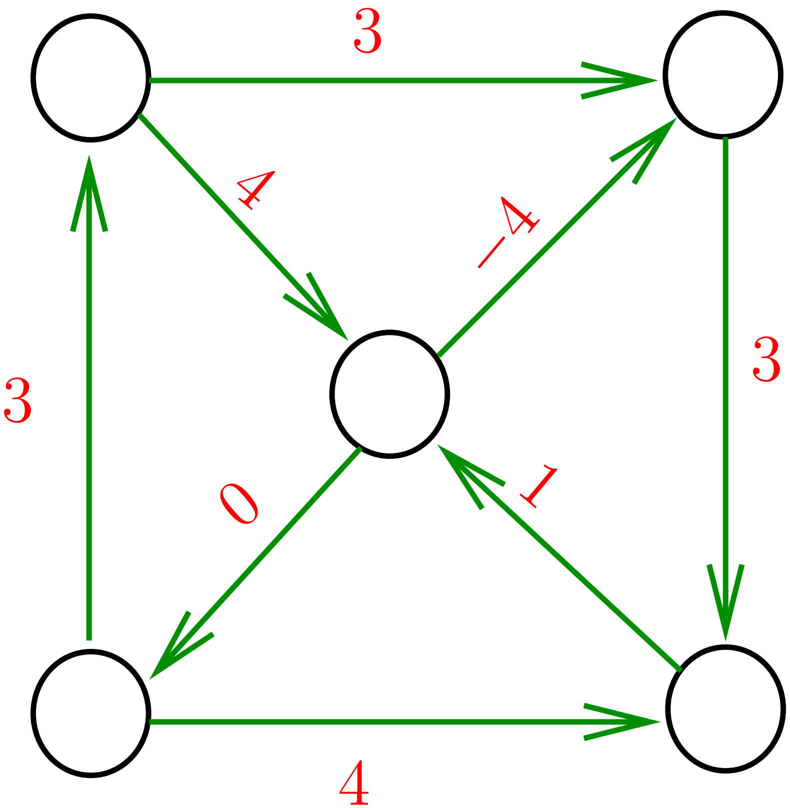


$c(ij)$

$\mu_*(c) = -1$

1 $\quad \mu \leftarrow -1$

Simulação



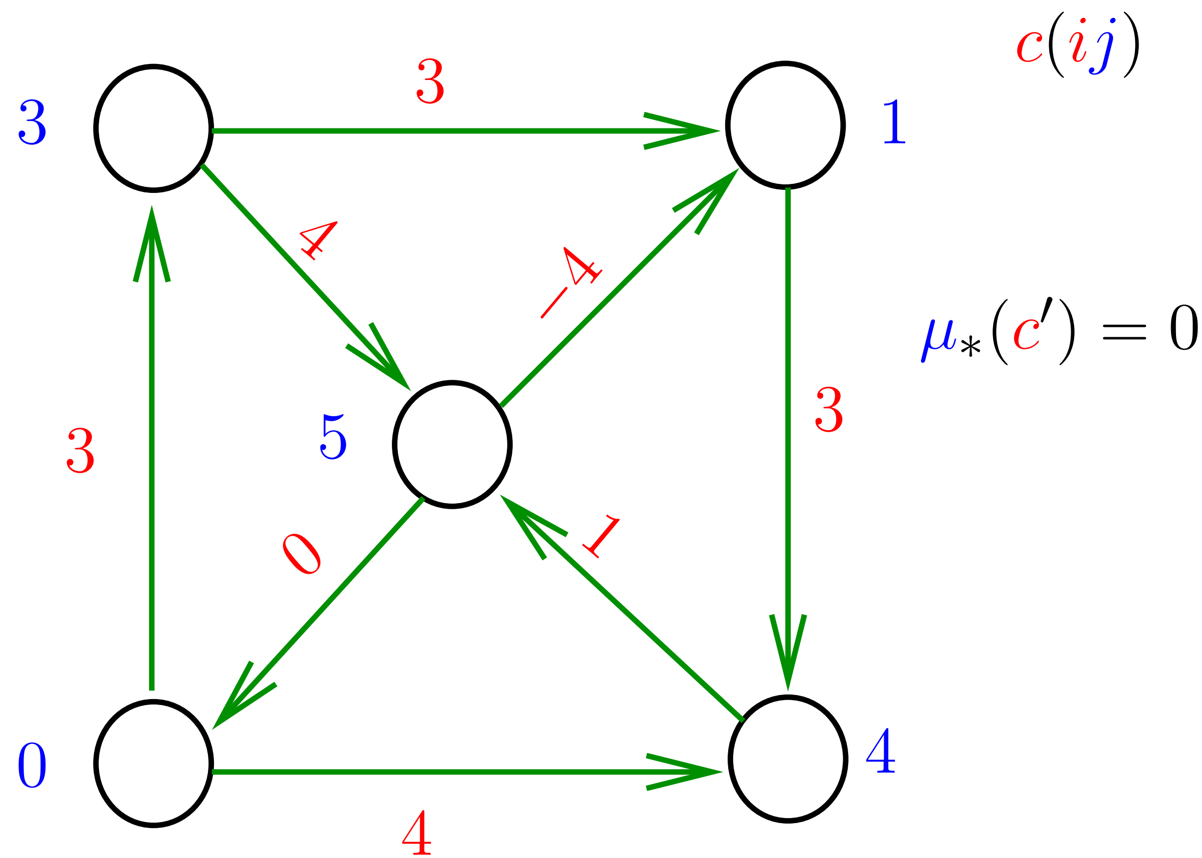
$c(ij)$

$\mu_*(c') = 0$

2 **para cada** ij **faça**

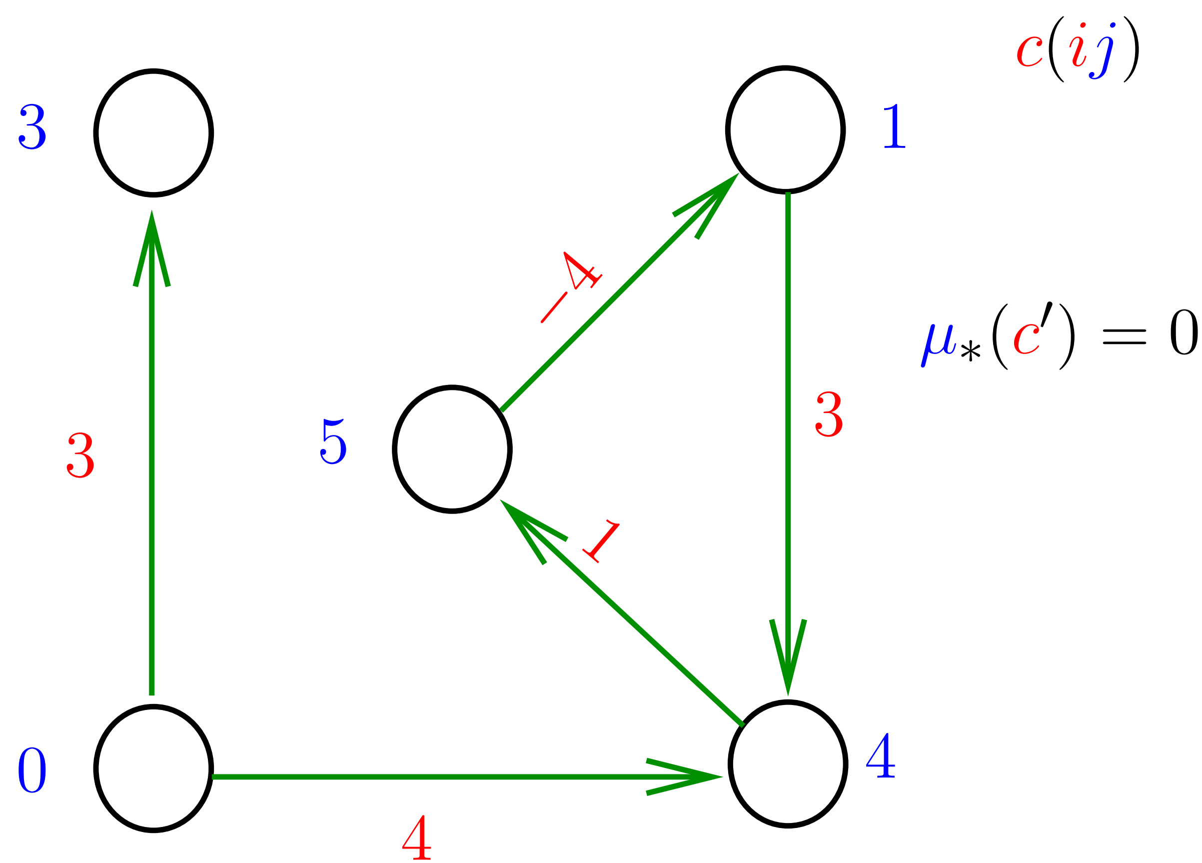
2 $\quad c'(ij) \leftarrow c(ij) - \mu$

Simulação



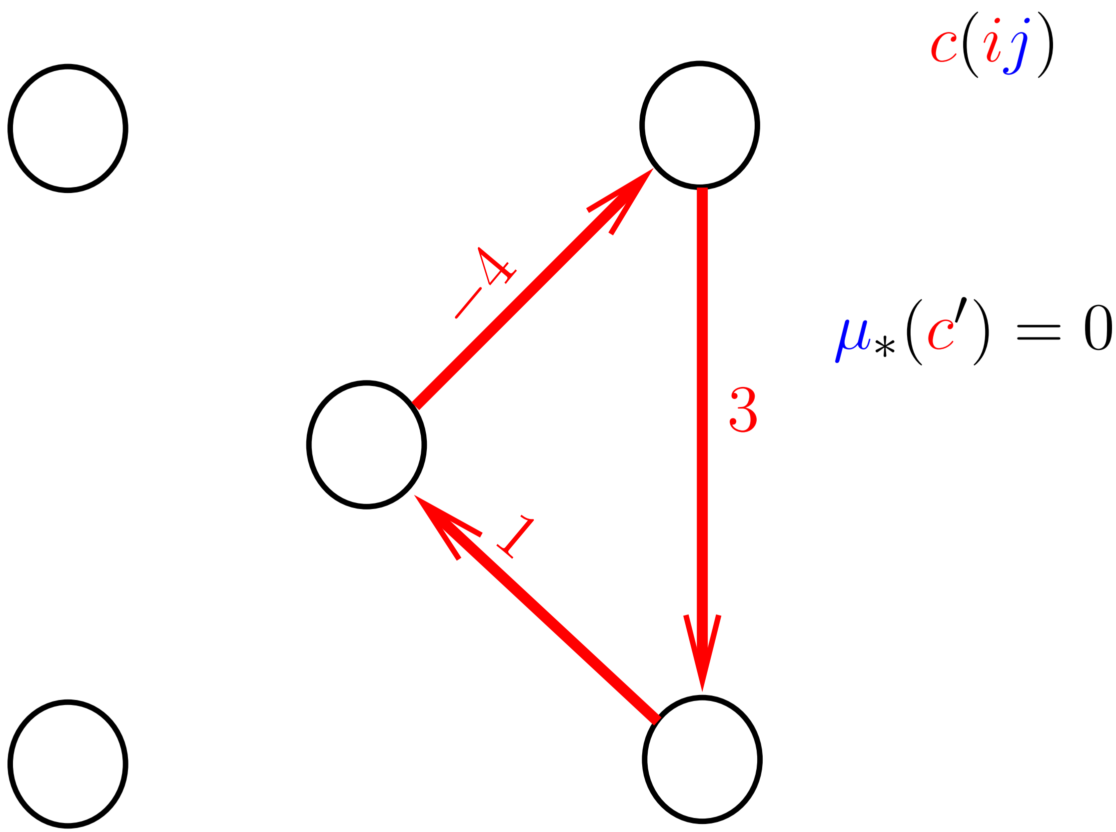
2 $\langle O, y\rangle \leftarrow$ CICLO-NEGATIVO(N, A, c')

Simulação



3 $\quad A' \leftarrow \{ij : y(j) - y(i) = c'(ij)\}$

Simulação



4 $\quad \langle O', y \rangle \leftarrow$ DAG(N, A')

Consumo de tempo

O consumo de tempo do algoritmo MIN-MEAN-CYCLE-DYN-PROG é $\mathrm{O}(nm)$ mais o consumo de tempo do algoritmo MIN-MEAN-CYCLE-VALUE.

AULA 22

Ciclo de Custo Médio Mínimo

PF 25.2

Algoritmo de Karp

$d[k, j]$:= custo mínimo de um passeio com exatamente k arcos e tem ponta final j.

Logo, $d[0, j] = 0$ e

$$d[k+1, j] = \min(d[k, i] + c(ij) : ij \in A)$$

para $k = 0, 1, \ldots$

Teorema. O custo médio mínimo de um ciclo é

$$\min_j \max \left\{ \frac{d[n, j] - d[k, j]}{n - k} : 0 \leq k \leq n - 1 \right\}$$

Demonstração (esboço). Podemos supor que $\mu_*(c) = 0$.

Demonstração (1)

Parte 1. Para todo j existe k tal que

$$d[n,j] - d[k,j] \geq 0 = \mu_*(c) \; .$$

Demonstração (esboço) Seja

$$P = \langle i_0, i_1, \ldots, i_r = i, \ldots, i = i_t, \ldots, i_n = j \rangle$$

passeio tal que $c(P) = d[n,j]$.

O passei P pode ser decomposto em um passeio com, digamos, k arcos

$$P' = \langle i_0, \ldots, i_r = i = i_t, \ldots, i_n = j \rangle$$

e um ciclo

$$O' = \langle i = i_r, \ldots, i = i_t \rangle \; .$$

Demonstração (2)

Temos que

$$
\begin{aligned}
d[n,j] - d[k,j &\geq d[n,j] - c(P') \\
&= c(O') \\
&\geq 0 = \mu_*(c)
\end{aligned}
$$

Portanto, $\min\max\ldots \geq 0 = \mu_*(c)$.

Demonstração (3)

Parte 2. Existe j tal que para todo k vale que

$$d[n,j] - d[k,j] \leq \mu_*(c) \ .$$

Demonstração (esboço) Seja $O = \langle i = i_0, i_1, \ldots, i_t = i \rangle$ um ciclo de custo 0.

Seja r tal que

$$d[r,i] = \min\{d[k,i] : k = 0,1,\ldots\}$$

Podemos supor que $n - t \leq r < n$.

Tome $j := i_{n-r}$ e sejam

$P := \langle i = i_0, i_1, \ldots, j \rangle$ e $Q := \langle j, \ldots, i_t = i \rangle$.

Demonstração (4)

Para todo k vale que

$$\begin{aligned} d[k,j] + c(Q) &\geq d[k+|Q|, i_t] \\ &\geq d[r,i] \\ &\geq d[n,j] - c(P) \end{aligned}$$

Logo, para todo k tem-se que

$$d[n,j] - d[k,j] \leq c(P) + c(Q) = c(O) = 0.$$

Portanto, $\min\max\ldots \leq 0 = \mu_*(c)$.

Algoritmo Min-Mean-Cycle-Value

MIN-MEAN-CYCLE-VALUE (N, A, c)

para cada i em N **faça**
  $d[0, i] \leftarrow 0$
para $k \leftarrow 0$ **até** $n-1$ **faça**
  para cada j em N **faça**
    $d[k+1, j] \leftarrow \infty$
    para cada ij em A **faça**
      $d[k+1, j] \leftarrow \min\{d[k+1, j], d[k, i] + c(ij)\}$
para cada j em N **faça**
  $m[j] \leftarrow \max\left\{\frac{d[n,j]-d[k,j]}{n-k} : k \in [0\,.\,.\,n-1]\right\}$
$\mu \leftarrow \min\{m[j] : j \in N\}$
devolva μ

Consumo de tempo

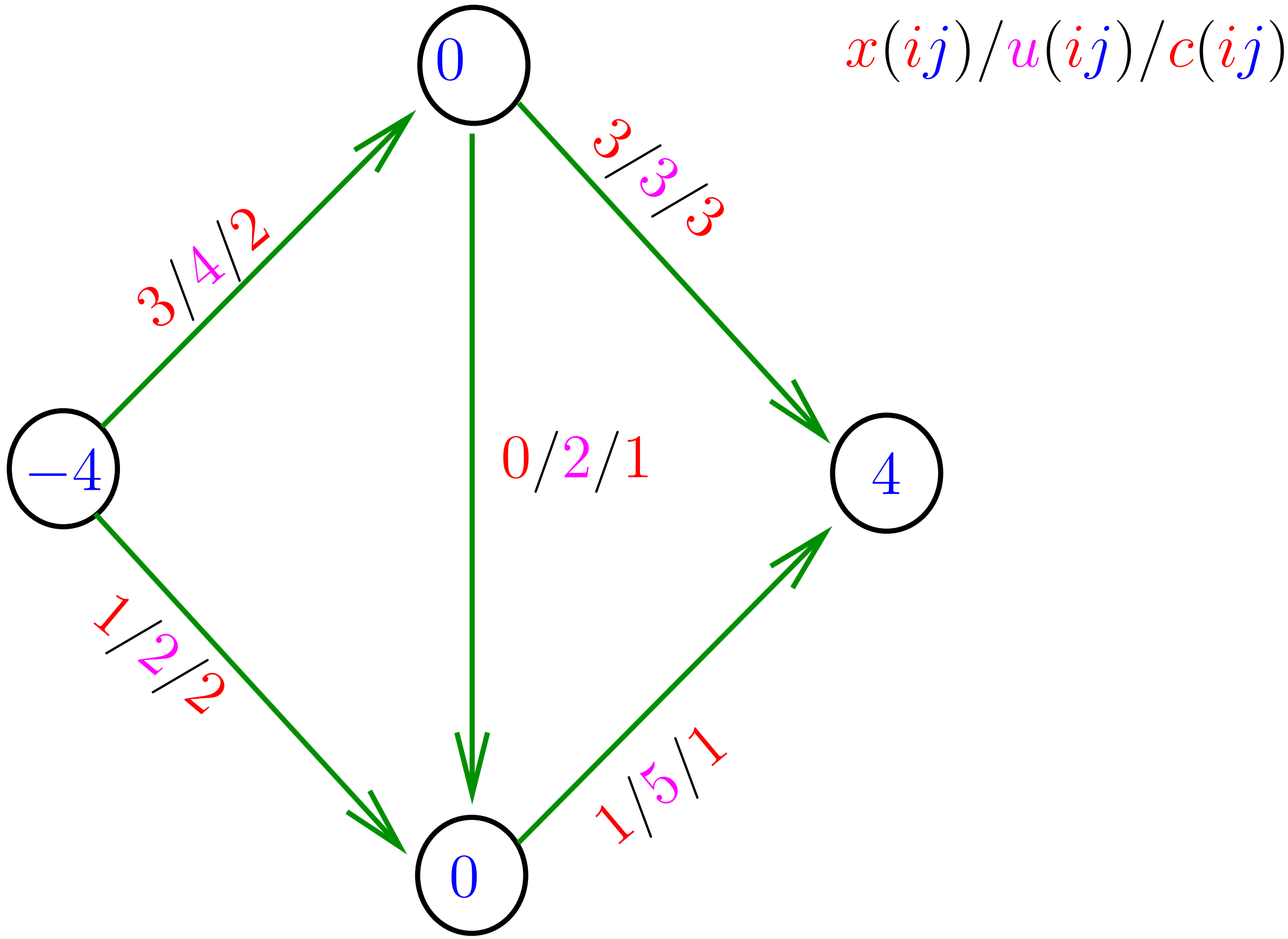
O consumo de tempo do algoritmo
MIN-MEAN-CYCLE-VALUE é $\mathrm{O}(nm)$.

O consumo de tempo do algoritmo
MIN-MEAN-CYCLE-DYN-PROG é $\mathrm{O}(nm)$.

Algoritmo Min-Mean-Klein

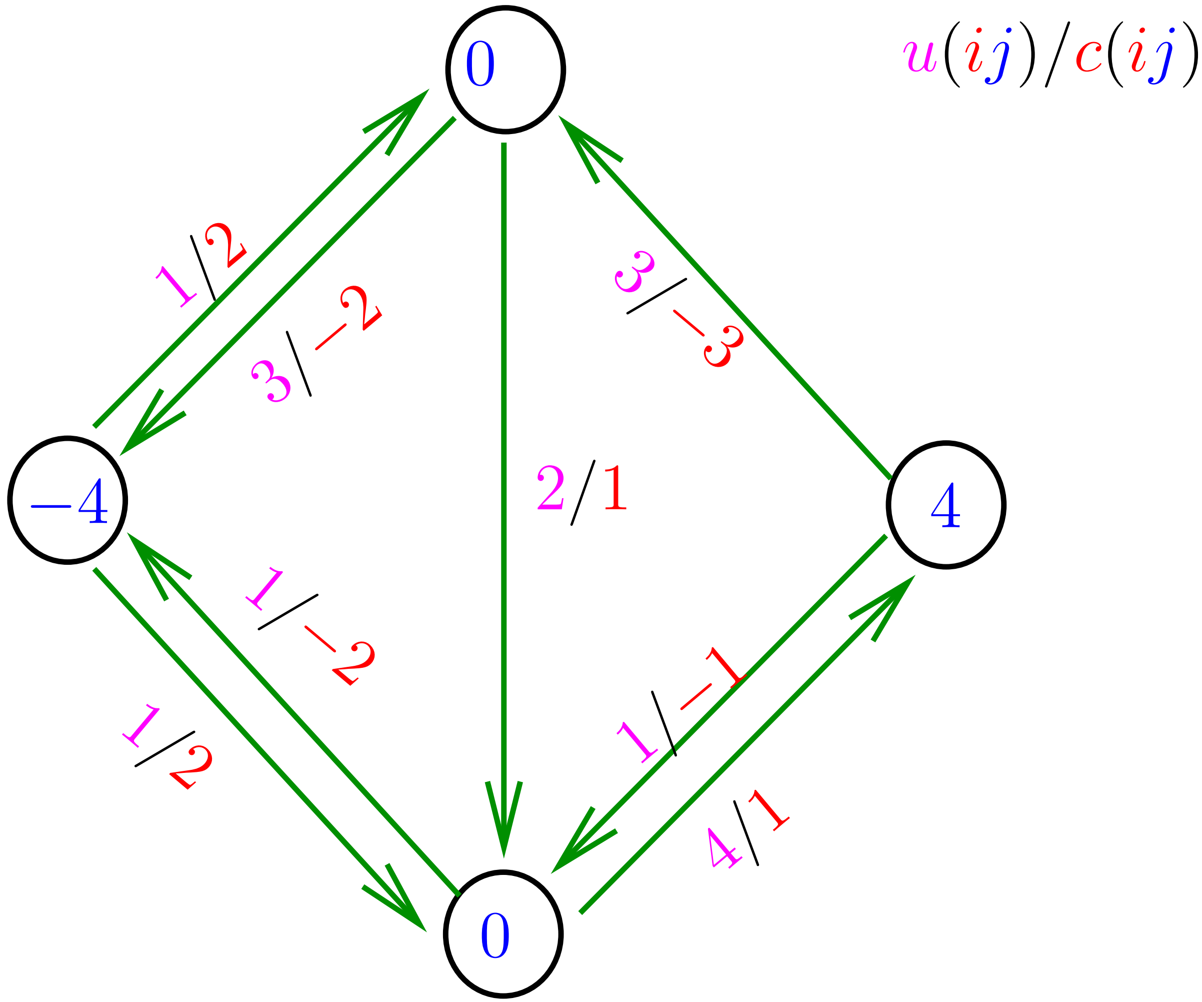
PF 25.3, 25.4

Rede e fluxo viável

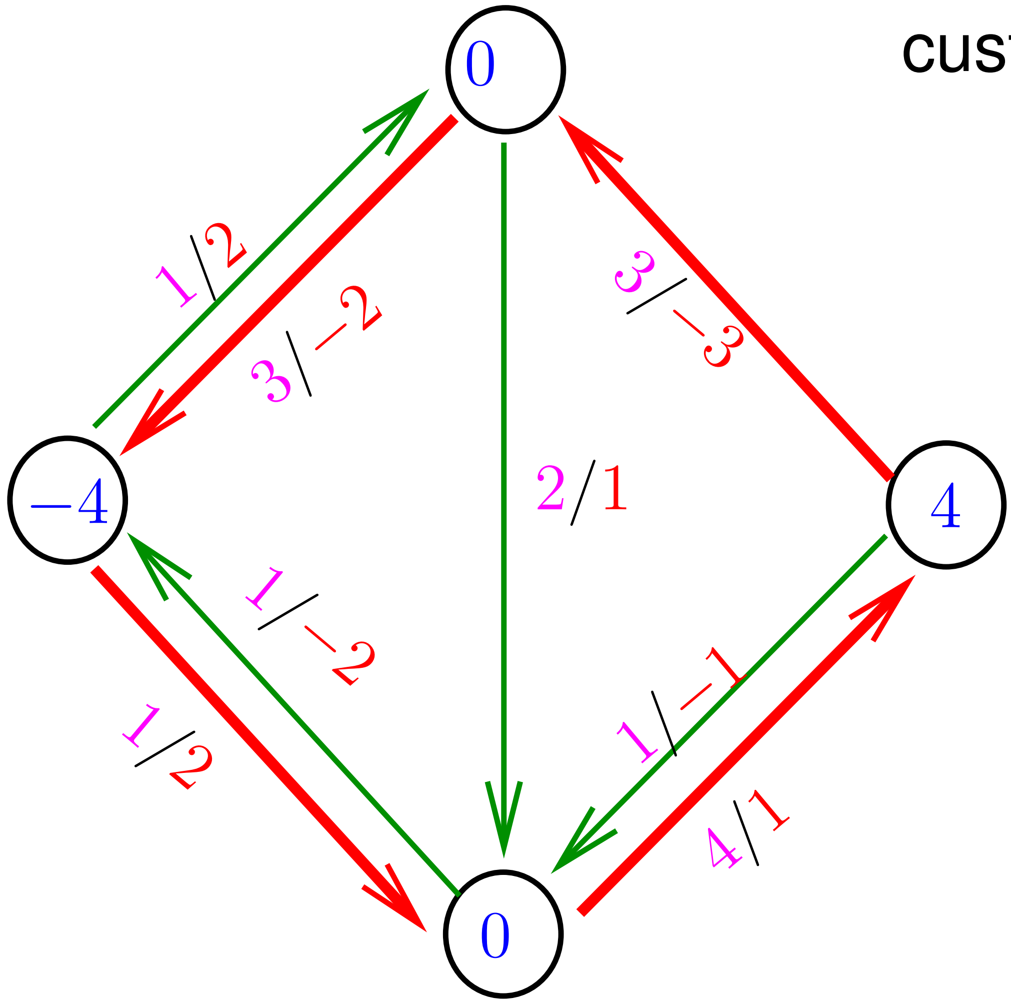


Custo $= 3 \times 2 + 1 \times 2 + 0 \times 1 + 3 \times 3 + 1 \times 1 = 18$

Rede residual

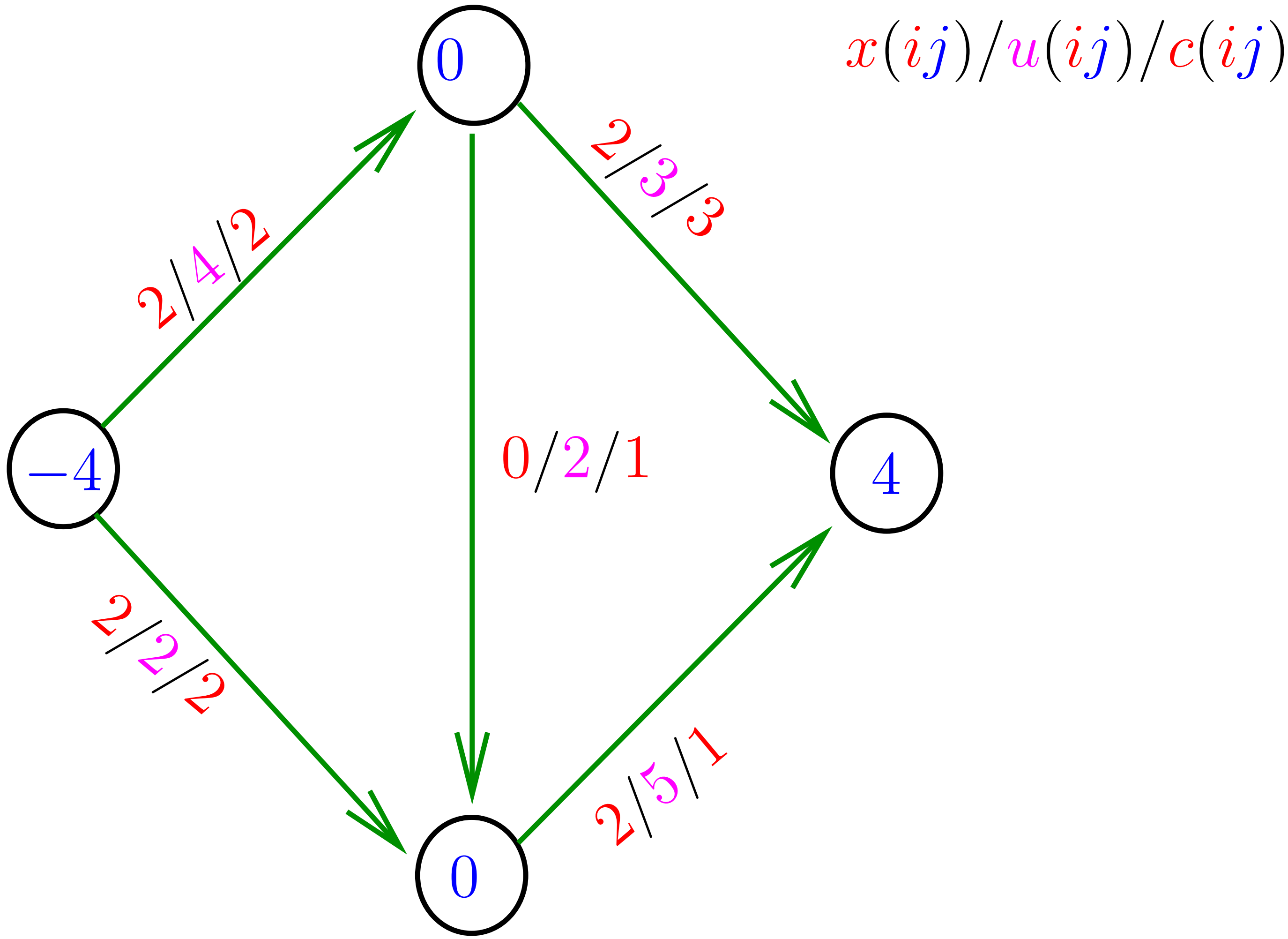


Ciclo de custo médio mínimo



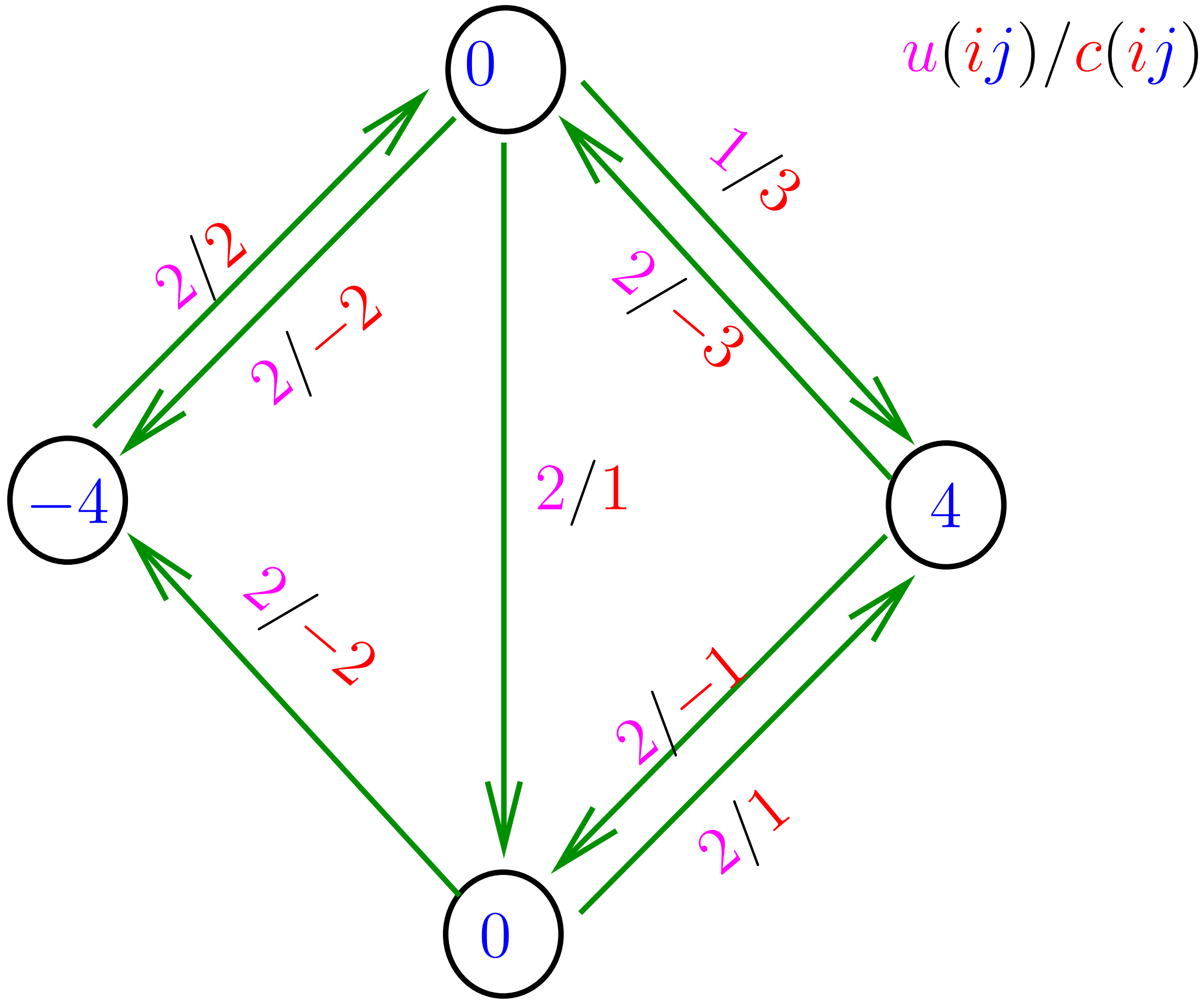
custo médio $= -1/2$

Rede e fluxo viável

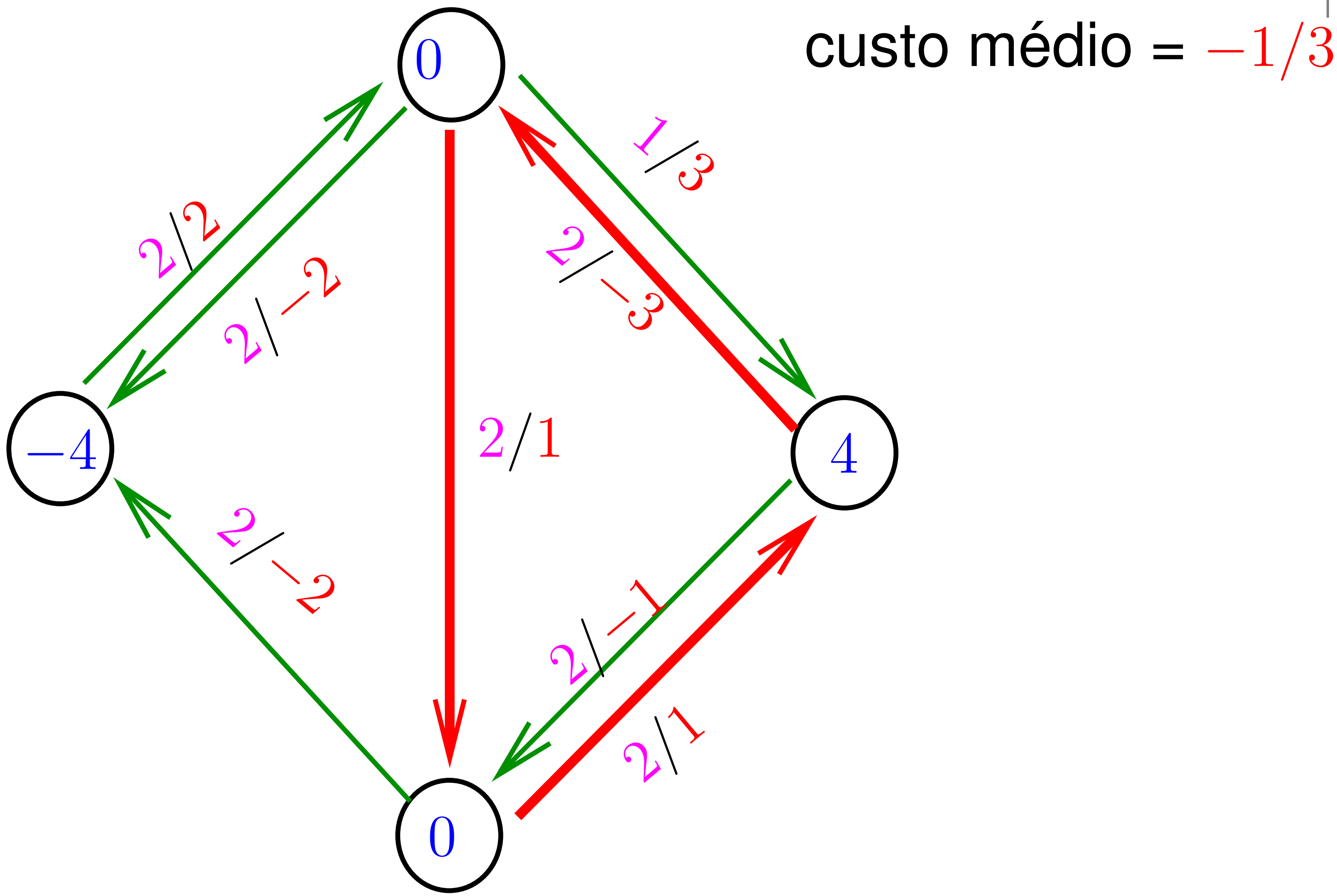


Custo $= 2 \times 2 + 2 \times 2 + 0 \times 1 + 2 \times 3 + 2 \times 1 = 16$

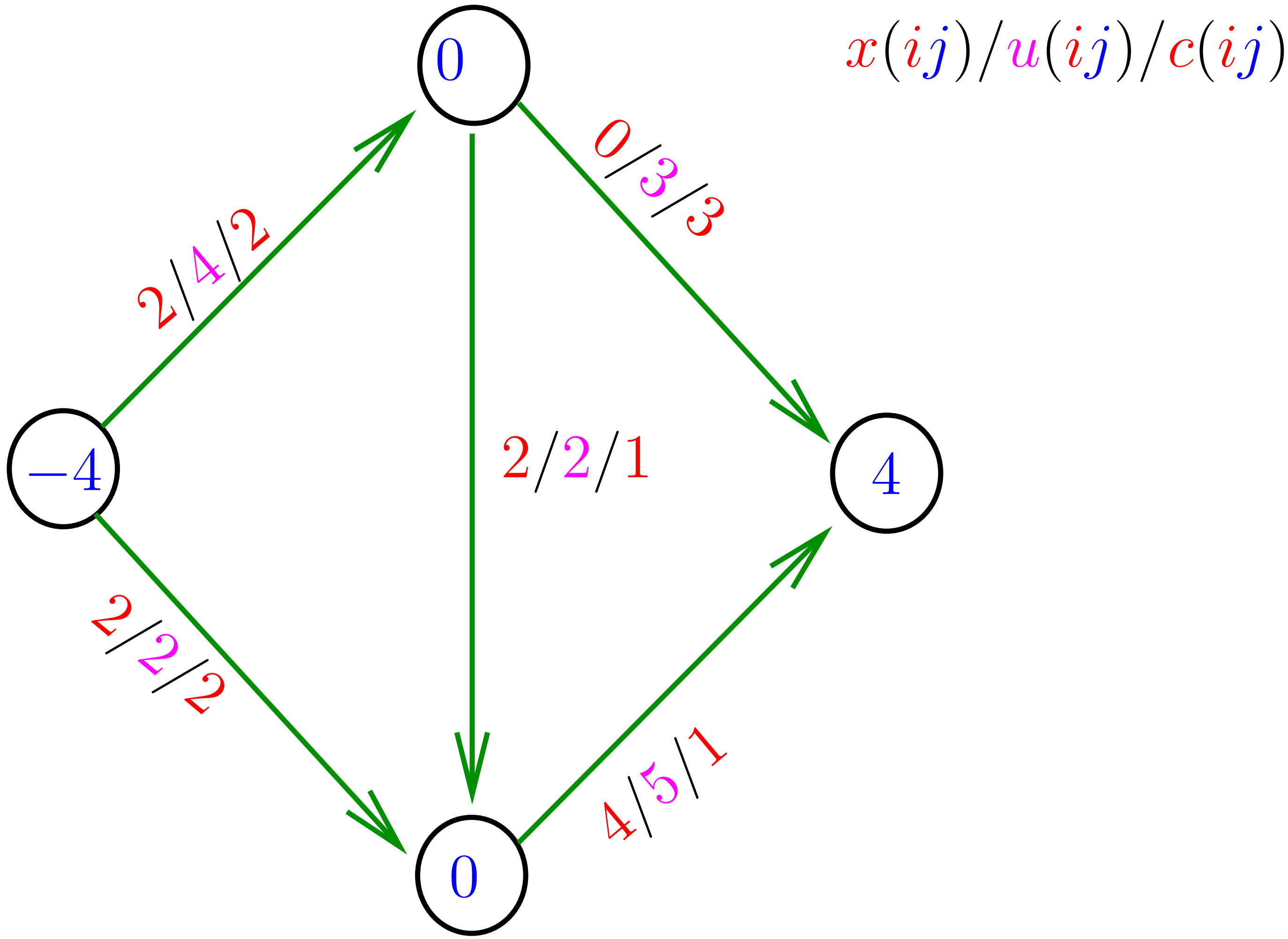
Rede residual



Ciclo de custo médio mínimo

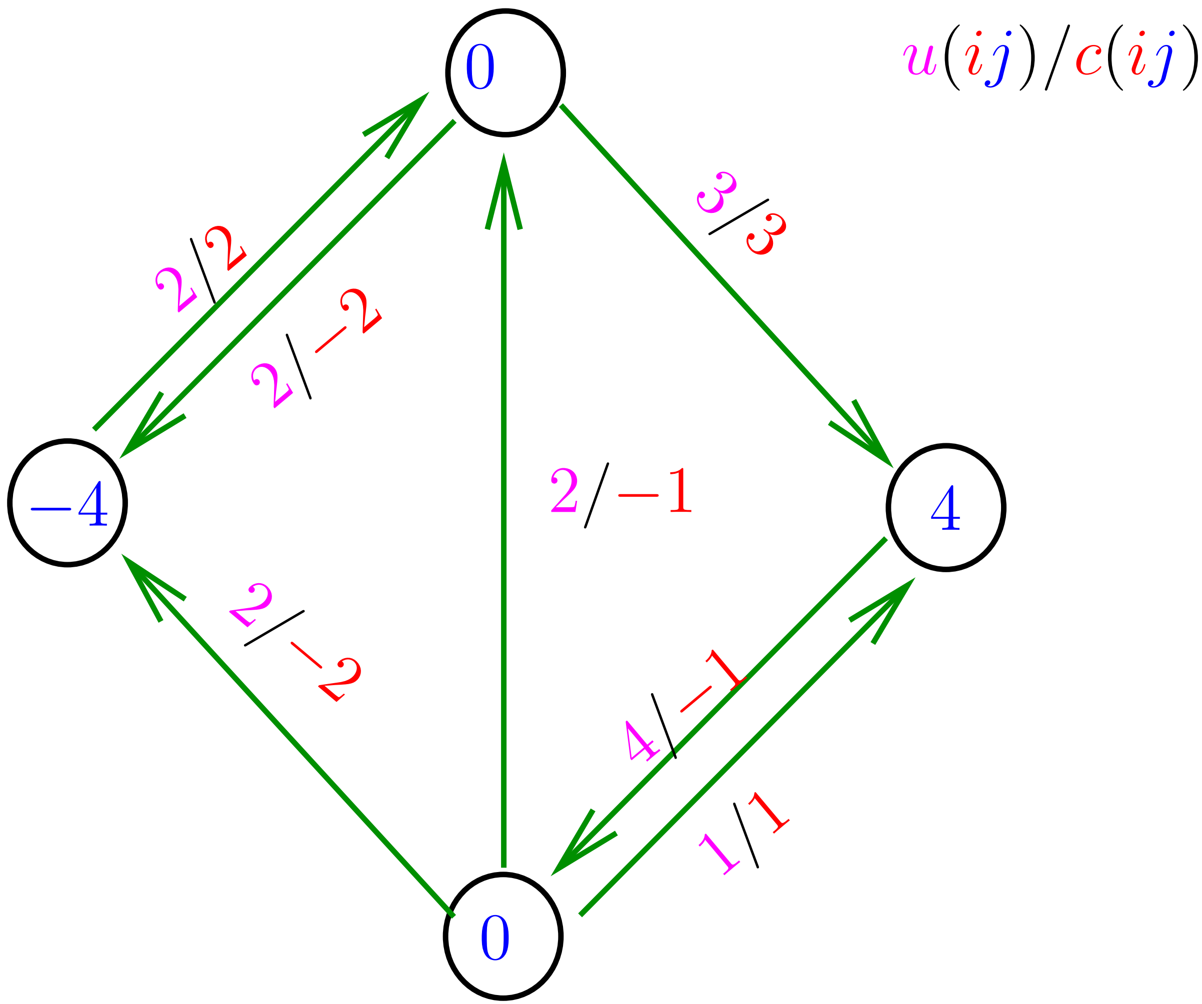


Rede e fluxo viável

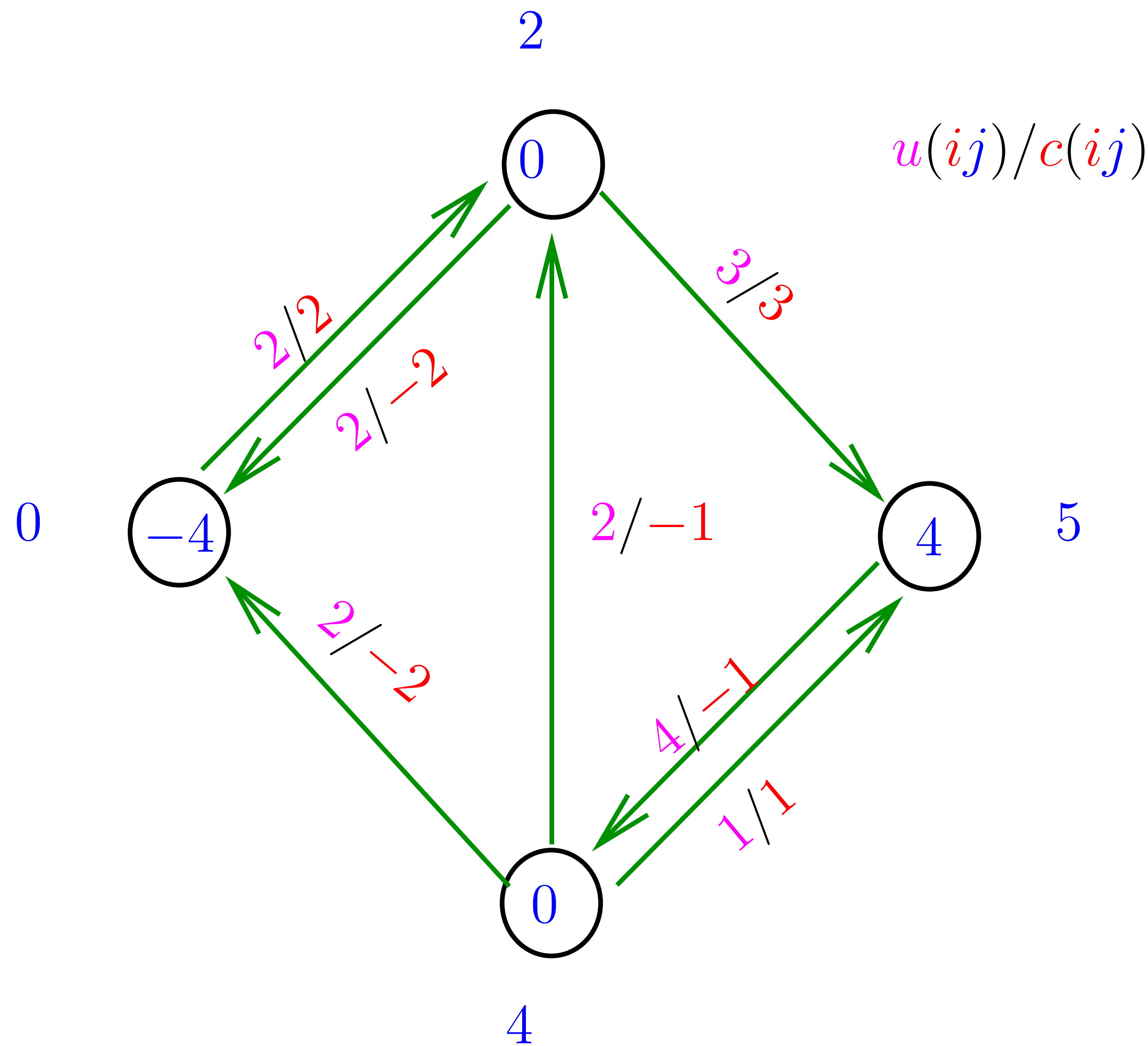


Custo $= 2 \times 2 + 2 \times 2 + 2 \times 1 + 0 \times 3 + 4 \times 1 = 14$

Rede residual



Potencial



Folgas complementares

Seja x o fluxo encontrado.

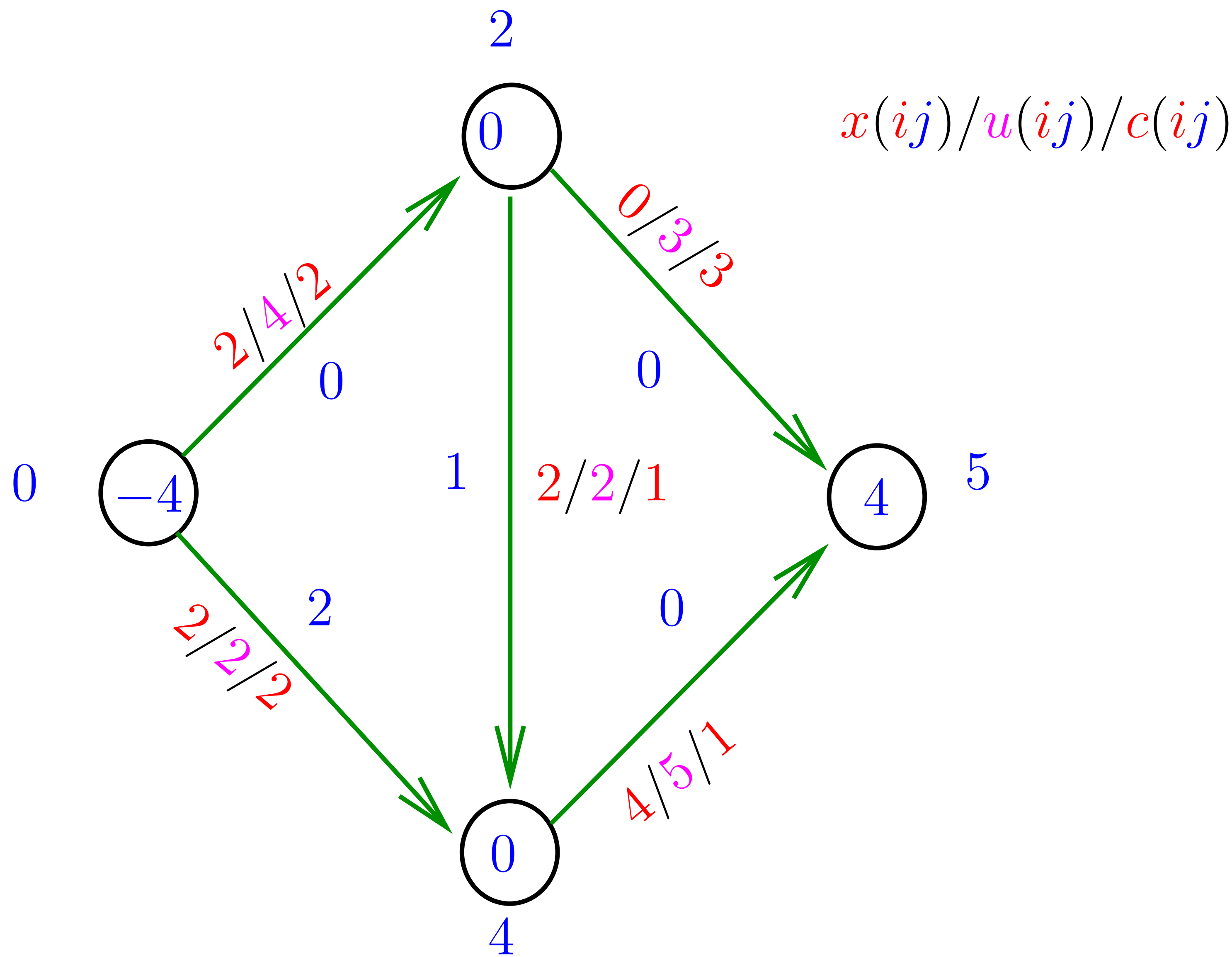
Seja y o potencial encontrado.

Temos as folgas de x e y são complementares já que,

$$y(j) - y(i) < c(ij) \Rightarrow x(ij) = 0 \quad \text{e}$$
$$y(j) - y(i) > c(ij) \Rightarrow x(ij) = u(ij) \ .$$

Conclusão: x é um fluxo viável de custo mínimo.

Solução dual viável



$$yb - wu = 4 \times 5 - 2 \times 2 - 1 \times 2 = 14$$

Algoritmo de Min-Mean-Klein

MIN-MEAN-KLEIN (N, A, u, b, c)

$\langle x, T\rangle \leftarrow$ FLUXO-VIÁVEL (N, A, u, b)
se x não está definido
  então devolva T

$\check{x} \leftarrow$ PSEUDOFLUXO(x)

repita

  $A_x \leftarrow \{ij \in A : \check{x}(ij) < u(ij)\}$
  $\langle O, y\rangle \leftarrow$ CICLO-NEGATIVO(N, A_x, c)
  se O está definido
    então $O \leftarrow$ MIN-MEAN-CYCLE(N, A_x, c)
      $\check{x} \leftarrow$ BARATEIE-FLUXO$(\check{x}, O)$

até que O não está definido

$x \leftarrow$ FLUXO$(\check{x})$
devolva x e y

Barateie fluxo

BARATEIE-FLUXO $(\check{x}, O)$

```
δ ← min{u(ij) − x(ij) : ij é arco de O}
para cada arco ij em O faça
        x̌(ij) ← x̌(ij) + δ
        x̌(ji) ← x̌(ji) − δ
devolva x̌
```

Invariantes

No início de cada iteração do bloco de linhas 6–9 valem as seguintes invariantes:

(i0) x é inteiro;

(i1) x é satisfaz b;

(i2) x respeita u.

onde $x :=$ FLUXO$(\check{x})$.

Observação

O algoritmo EDMONDS-KARP para encontra um fluxo máximo pode ser "visto" como uma aplicação do algoritmo MIN-MEAN-KLEIN.

EDMONDS-KARP (N, A, u, s, t)

$A' \leftarrow A \cup \{ts\}$
para cada ij em A' **faça**
    $c(ij) \leftarrow 1 \qquad u'(ij) \leftarrow u(ij)$
para cada i em N **faça**
    $b(i) \leftarrow 0$
$c(ts) \leftarrow -\infty \qquad u'(ts) \leftarrow \infty$
$b(s) \leftarrow -$*valor* $\qquad b(t) \leftarrow$ *valor*
$\langle x, y \rangle \leftarrow$ MIN-MEAN-KLEIN(N, A', u', b, c)
devolva x e y

Número de iterações

- x_k := fluxo viável no início da iteração k (linha 5)
- (N, A_k) := rede residual construída da iteração k (linha 6);
- O_k := ciclo de custo médio mínimo obtido na iteração k (linha 9);
- α_k := $-$ custo médio mínimo de um ciclo (N, A_k, c) ($= -$ custo médio de O_k);
- α_k é o menor valor para o qual existe uma função potencial y_k tal que

$$y_k(j) - y_k(i) \leq c(ij) + \alpha_k$$

para cada arco ij em (N, A_k, c);

Número de iterações

$r :=$ número de iterações

Fato 1. Para $k = 1, \ldots, r$, existe ij em O_k tal que

$$ij \notin O_h \ .$$

para todo $h \geq k + 2nm\lceil \ln n \rceil$.

Fato 2. Para $k = 1, \ldots, r - 1$ vale que

$$\alpha_{k+1} \leq \alpha_k \ .$$

Fato 3. Para $k = 1, \ldots, r - m - 1$ vale que

$$\alpha_{k+m} \leq (1 - 1/n)\alpha_k \ .$$

Conclusões

O algoritmo MIN-MEAN-KLEIN faz não mais que $4nm^2\lceil \ln n \rceil$ iterações.

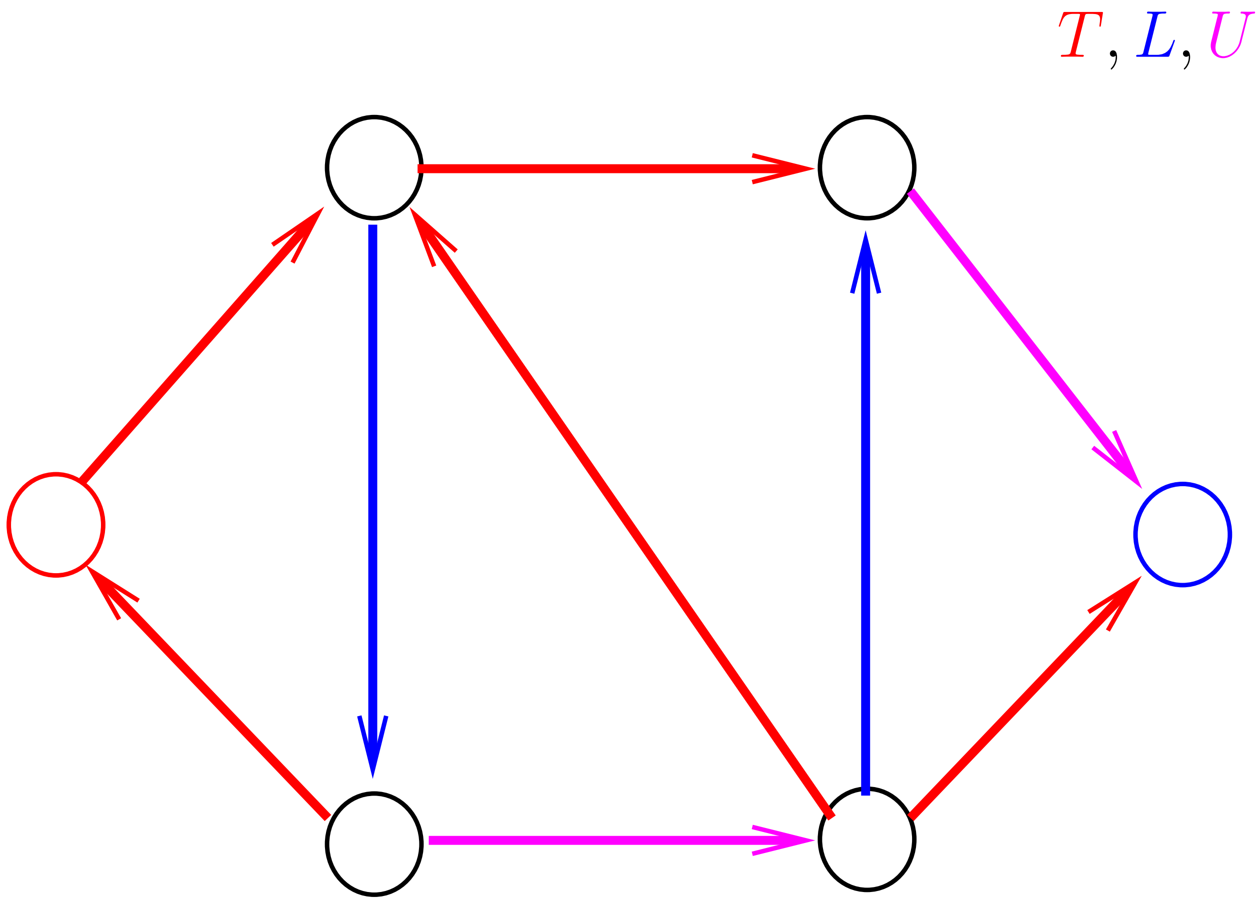
O consumo de tempo do algoritmo MIN-MEAN-KLEIN é $\mathrm{O}(n^2 m^3 \lg n)$.

Este consumo de tempo é fortemente polinomial.

Simplex para redes

Estruturas arbóreas

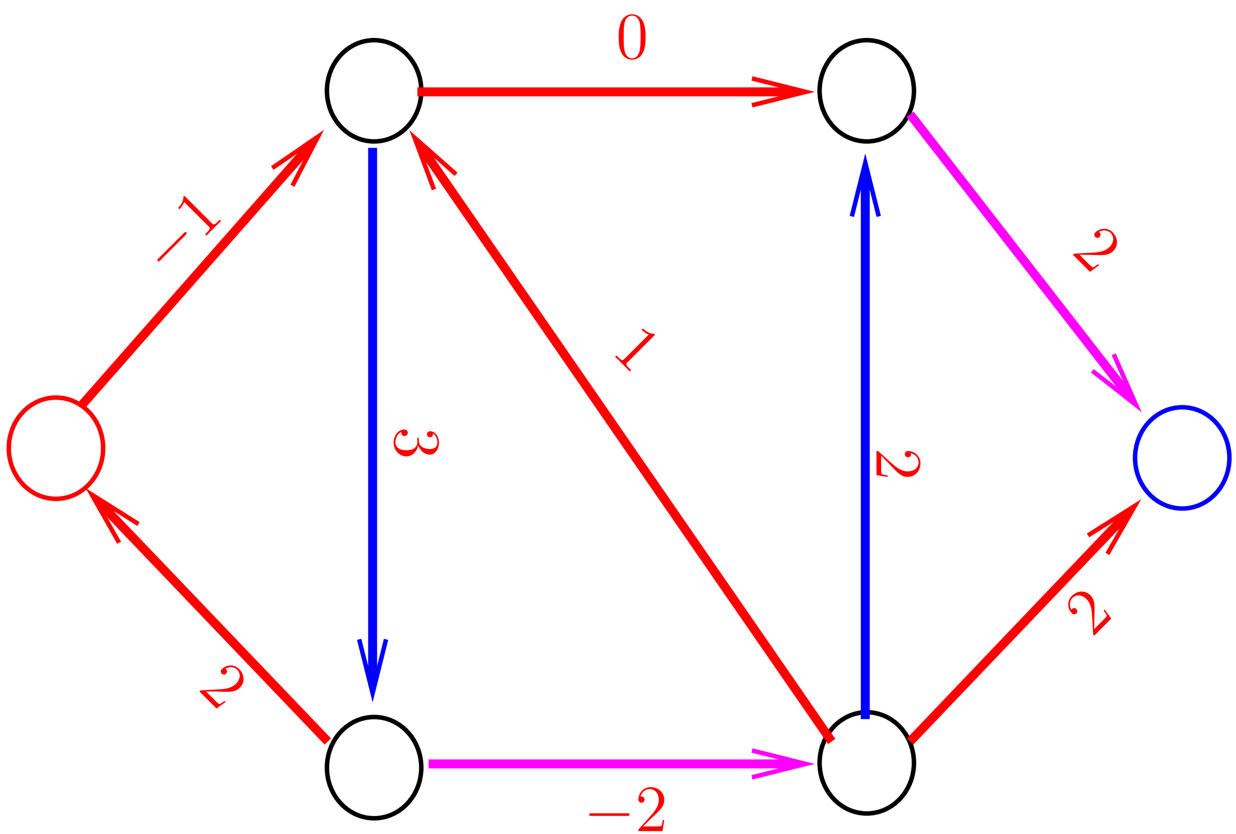
Uma estrutura arbórea num grafo (N, A) é uma tripartição T, L, U de A tal que (N, T) é uma árvore.



Estruturas arbóreas e potenciais (1)

Suponha que T, L, U é uma estrutura arbórea de (N, A, c). Dizemos que uma função potencial y é determinada por T, c se, para cada ij em T,

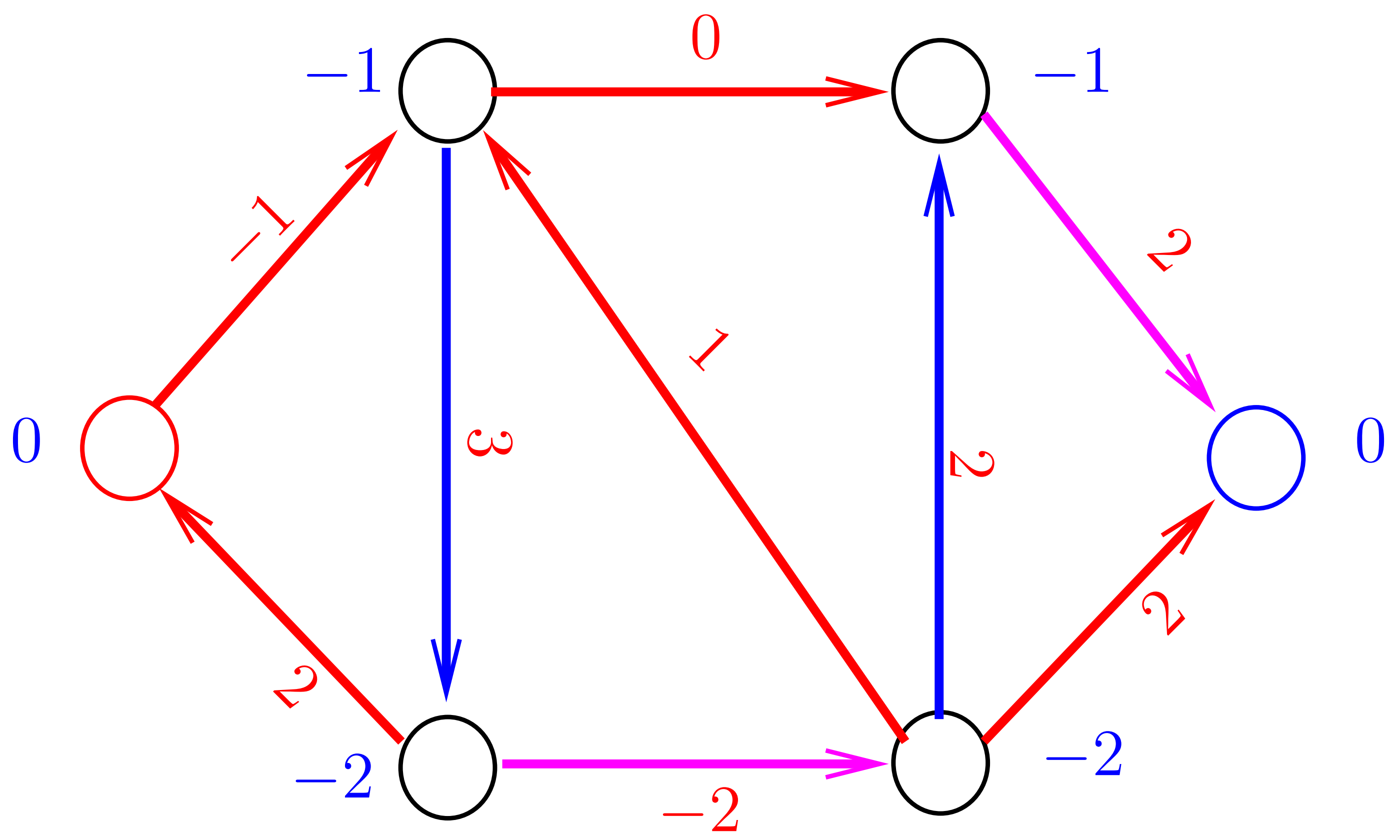
$$y(j) - y(i) = c(ij) \ .$$



Estruturas arbóreas e potenciais (1)

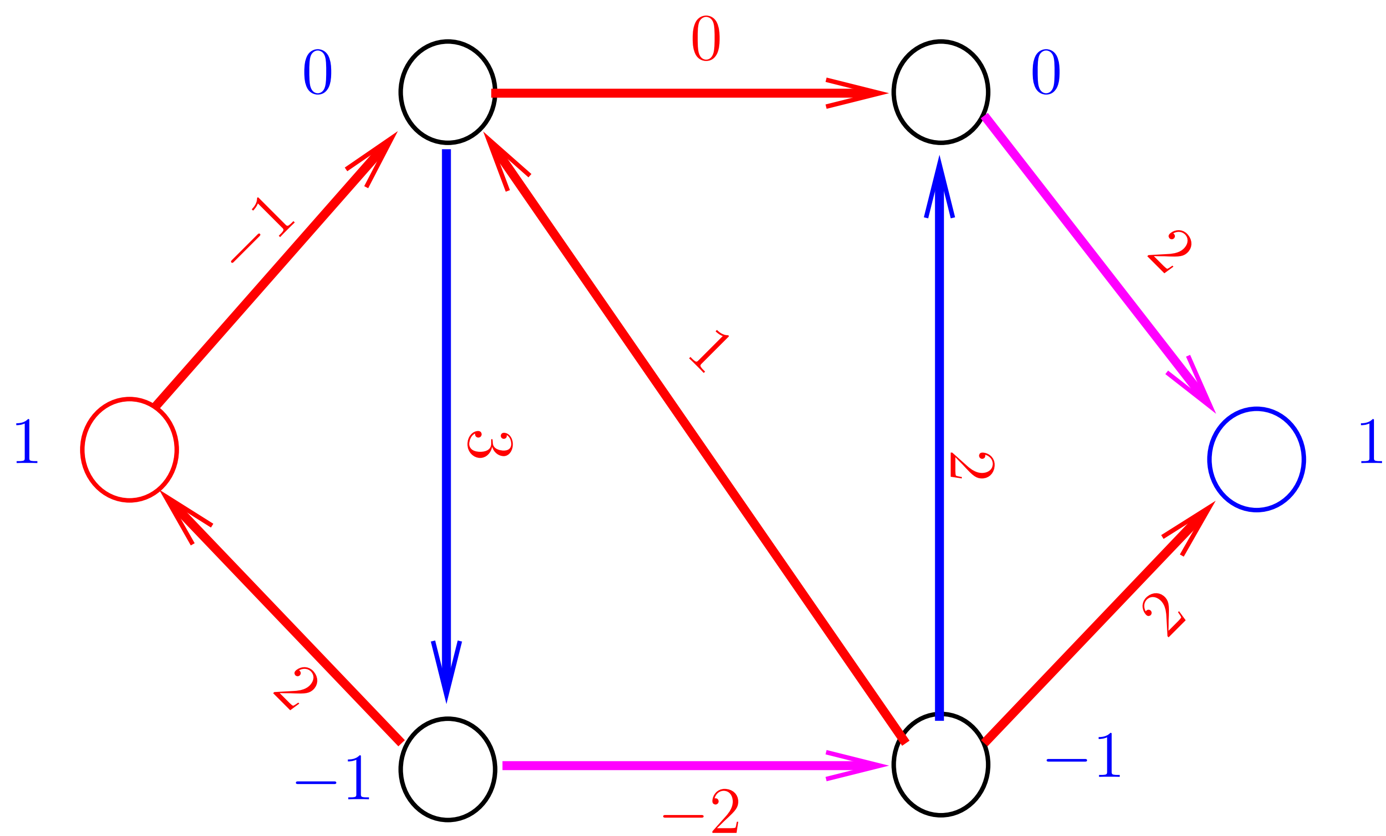
Suponha que T, L, U é uma estrutura arbórea de (N, A, c).
Dizemos que uma função potencial y é determinada por T, c se, para cada ij em T,

$$y(j) - y(i) = c(ij) \ .$$



Estruturas arbóreas e potenciais (2)

Quantas funções potenciais determinadas por T, c existem? Em geral muitas! Essencialmente uma!



Estruturas arbóreas e fluxos (1)

Suponha que x é um fluxo que respeita u.

Uma estrutura arbórea T, L, U está associada ao fluxo x se

$$\begin{aligned} ij \in L \quad &\Rightarrow x(ij) = 0 \\ ij \in U \quad &\Rightarrow x(ij) = u(ij) \end{aligned}$$

Logo, todo arco ij tal que $0 < x(ij) < u(ij)$ está em T.

Não é verdade que existe uma estrutura arbórea associada a qualquer fluxo que respita u.

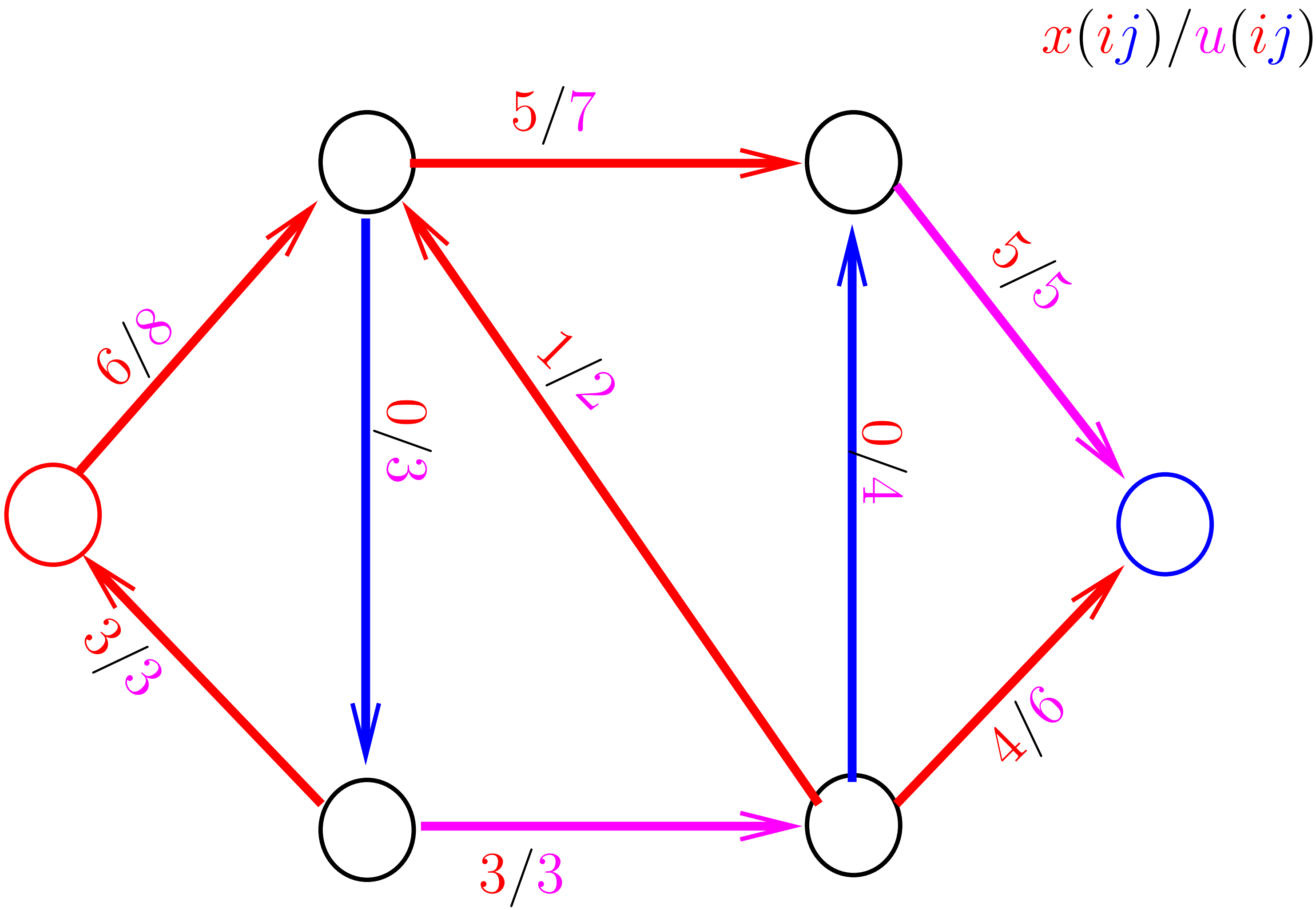
Um arco ij tal que $0 < x(ij) < u(ij)$ é dito livre.

Um fluxo é acíclico se o conjunto de seus arcos livres é acíclico.

Um fluxo é degenerado se algum arco em T não é livre.

Estruturas arbóreas e fluxos (2)

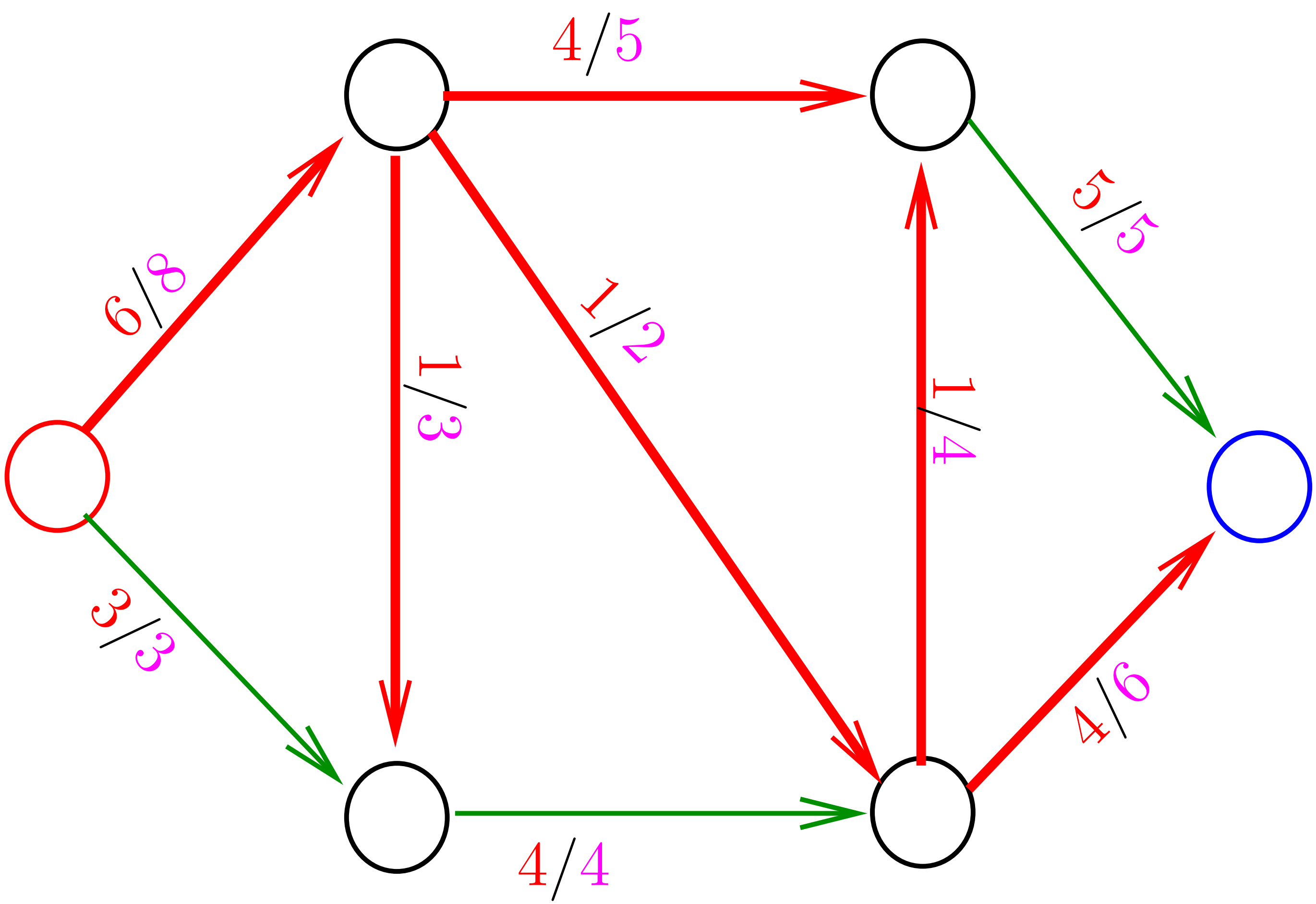
Eis um exemplo de uma estrutura arbórea T, L, U associada a um fluxo x.



Estruturas arbóreas e fluxos (3)

Eis um exemplo de fluxo que não possui uma estrutura arbórea T, L, U associada.

$x(ij)/u(ij)$



Estruturas arbóreas e fluxos (4)

Fato. Se x é um fluxo viável, então existe um fluxo viável acíclico x' tal que $cx' \leq cx$. A demonstração é algorítmica.

FLUXO-ACÍCLICO (N, A, u, c, x)

repita
    $A' \leftarrow \{ij \in A : 0 < x(ij) < u(ij)\}$
    $O \leftarrow$ PSEUDO-CICLO(N, A')
    se O está definido
        se $c(O) > 0$
            então $O \leftarrow$ "pseudo-ciclo reverso de O"
        $x \leftarrow$ BARATEIE-FLUXO(x, O)
até que O não está definido
devolva x

Barateie fluxo (esboço)

Recebe um fluxo x que respeita u e um pseudo-ciclo de custo não-negativo O e **devolve** o fluxo x após enviar "δ unidades de fluxo através de O".

BARATEIE-FLUXO (x, O)

$\delta_1 \leftarrow \min\{x(ij) : ij$ é arco de $\overleftarrow{O}\}$

$\delta_2 \leftarrow \min\{u(ij) - x(ij) : ij$ é arco de $\overrightarrow{O}\}$

$\delta \leftarrow \min\{\delta_1, \delta_2\}$

para cada arco ij em O **faça**

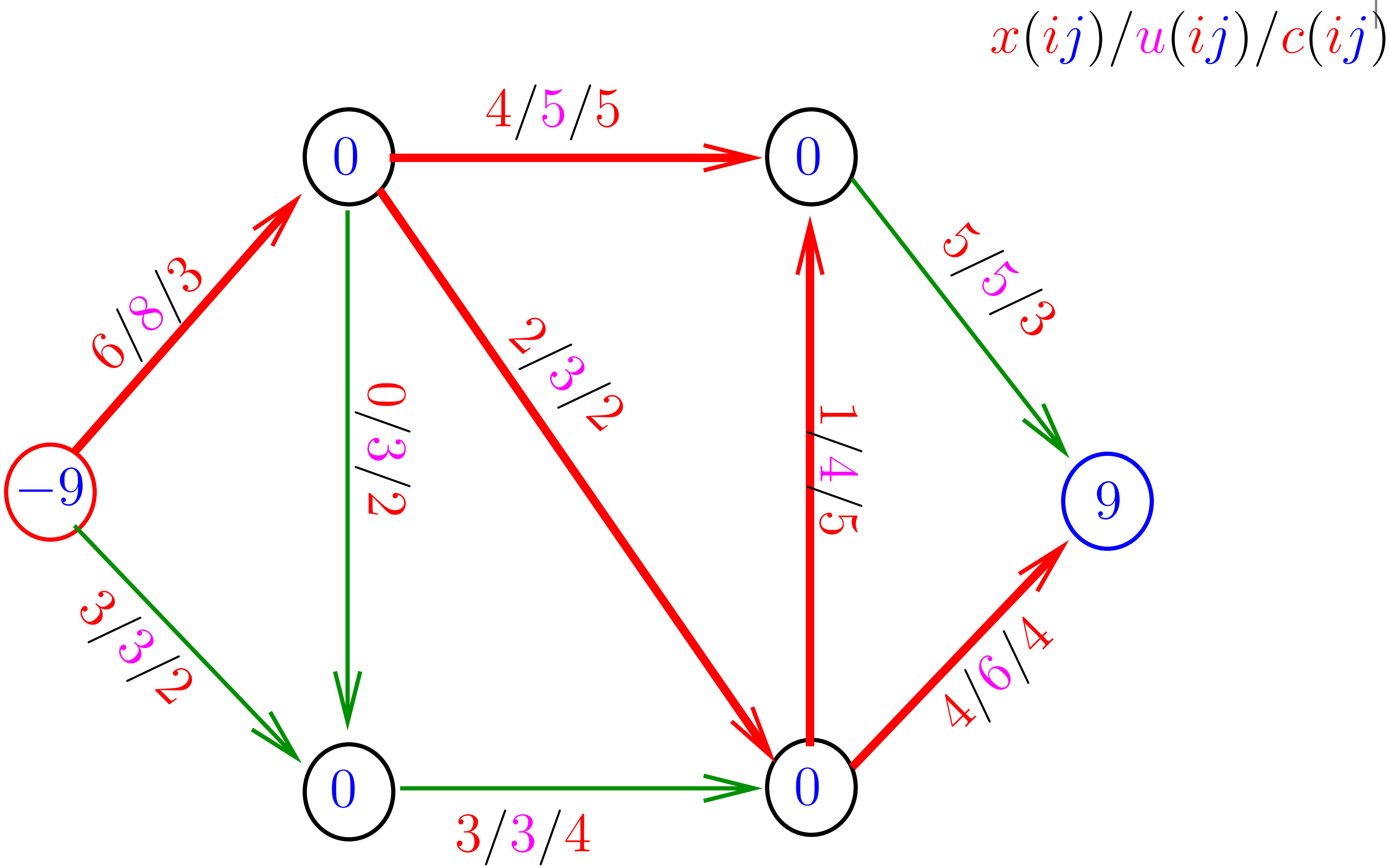
  se $ij \in \overrightarrow{O}$

    então $x(ij) \leftarrow x(ij) + \delta$

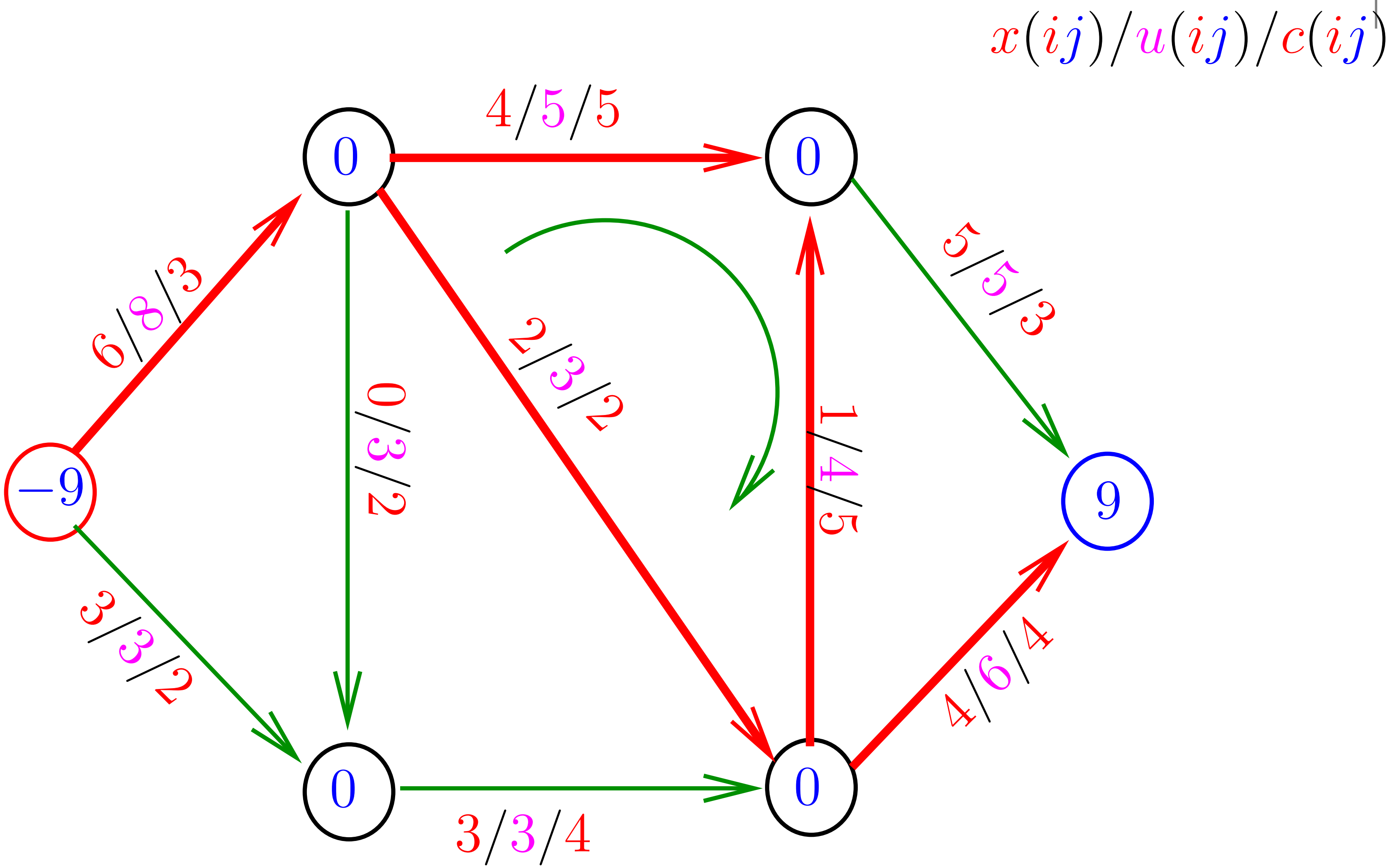
    senão $x(ij) \leftarrow x(ij) - \delta$

devolva x

Exemplo



Exemplo



Exemplo