

Melhores momentos

AULA PASSADA

Ford e Fulkerson

Método dos caminhos de incremento

PASSO DE INCREMENTO: encontre um caminho de incremento para o fluxo corrente. Incremente o valor do fluxo “enviando a maior quantidade possível de fluxo através do caminho”.

Note que o método não especifica como encontrar o pseudo-caminho de incremento.

Edmonds e Karp

Método dos incrementos através de caminhos “curtos”

PASSO DE INCREMENTO: encontre na **rede residual** (do **fluxo corrente**) um caminho de **comprimento mínimo**.

Incremente o valor do fluxo “enviando a **maior quantidade possível** de fluxo possível através desse caminho”.

Resumo

Invariante: no início de cada iteração temos um st -pseudofluxo $\tilde{x}$ que respeita as capacidades.

Algoritmo	número de passos de incremento	consumo de tempo
FORD-FULKERSON	$O(nU)$	$O(nmU)$
MAX-CAPACITY	$\leq 2m(1 + \lfloor \lg U \rfloor)$ $O(m \lg U)$	$O(n^2 m \lg U)$
CAPACITY-SCALING	$\leq 2m(1 + \lfloor \lg U \rfloor)$ $O(m \lg U)$	$O(m^2 \lg U)$
EDMONDS-KARP	nm	$O(nm^2)$

Estamos supondo que $n = O(m)$ (grafo conexo)

AULA 19

Dinits

PF 16.1, 16.2, 16.3, 16.4

Dinits

EDMONDS-KARP envia fluxos através de caminhos curtos, mas **calcula** um 1-potencial $(s, *)$ -ótimo em cada iteração.

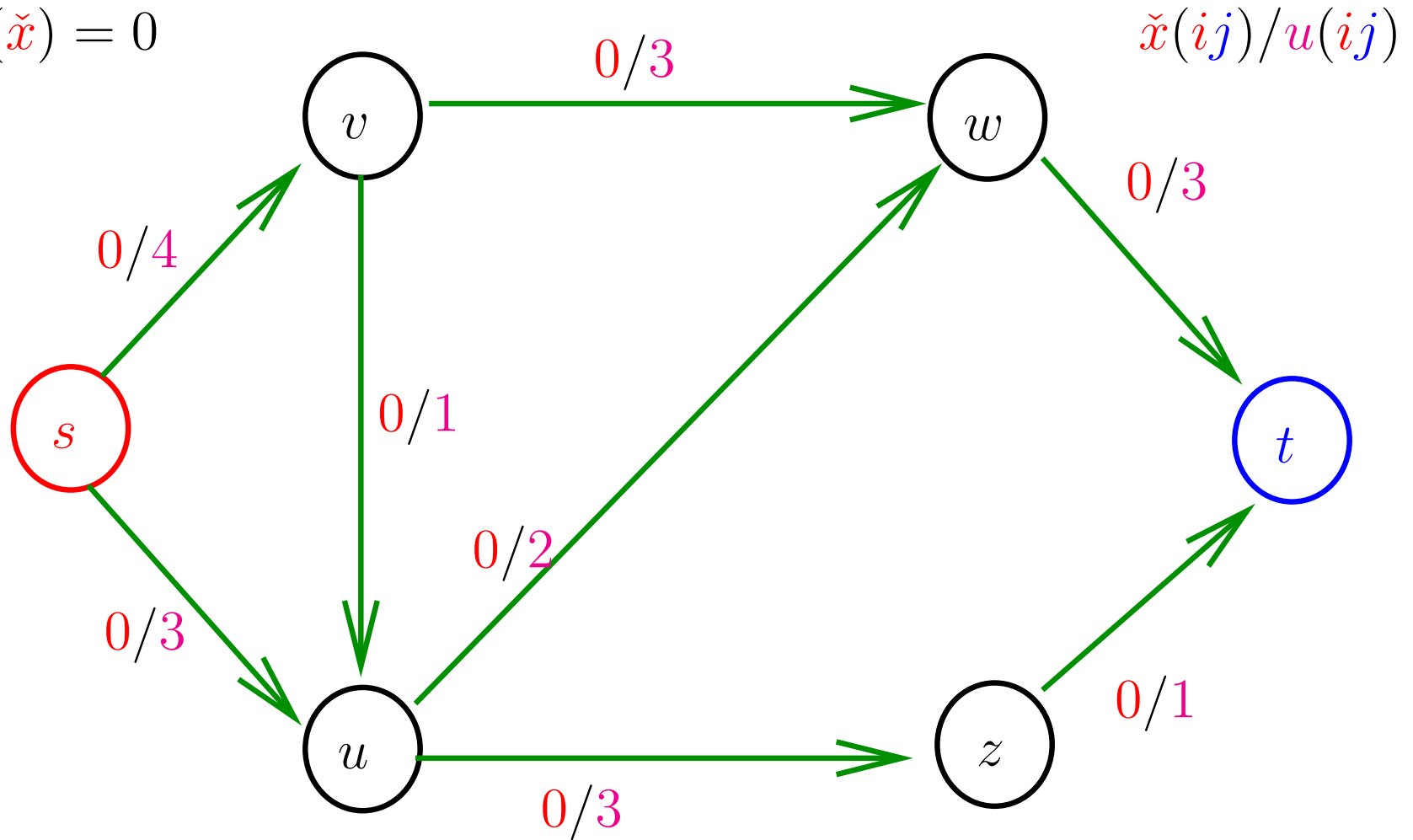
EDMONDS-KARP descarta os resultados da busca em largura.

Idéia: reutilizar o resultado de cada busca em largura o máximo possível.

Em cada iteração o algoritmo usa uma “**rede em camadas**” (= *layered network*) formada pelos arcos que estão em algum caminho mínimo com origem em s (ou com destino t , dependendo do gosto do freguês).

Rede 1

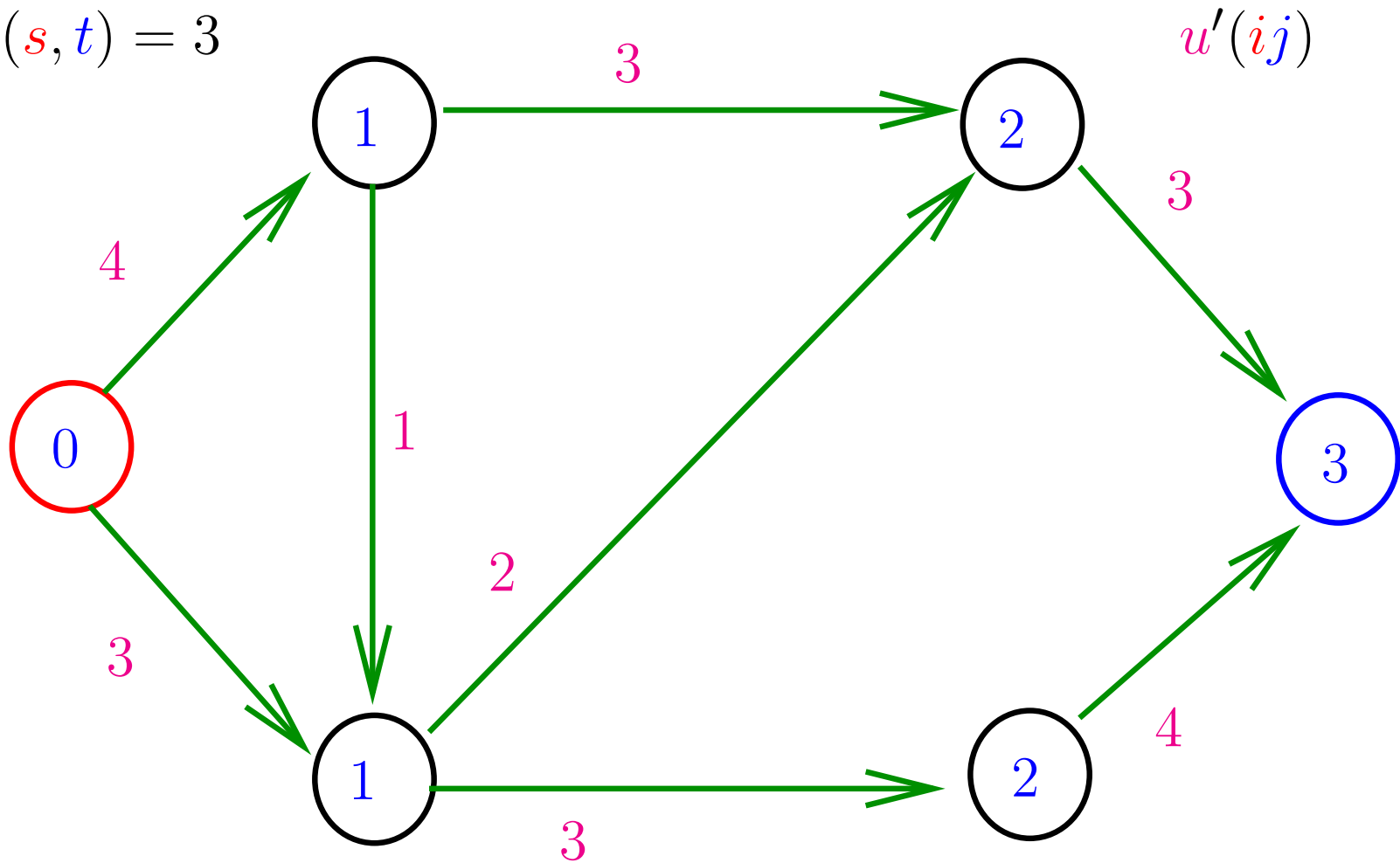
$$\text{val}(\tilde{x}) = 0$$



Arcos ausentes têm **capacidade** nula.

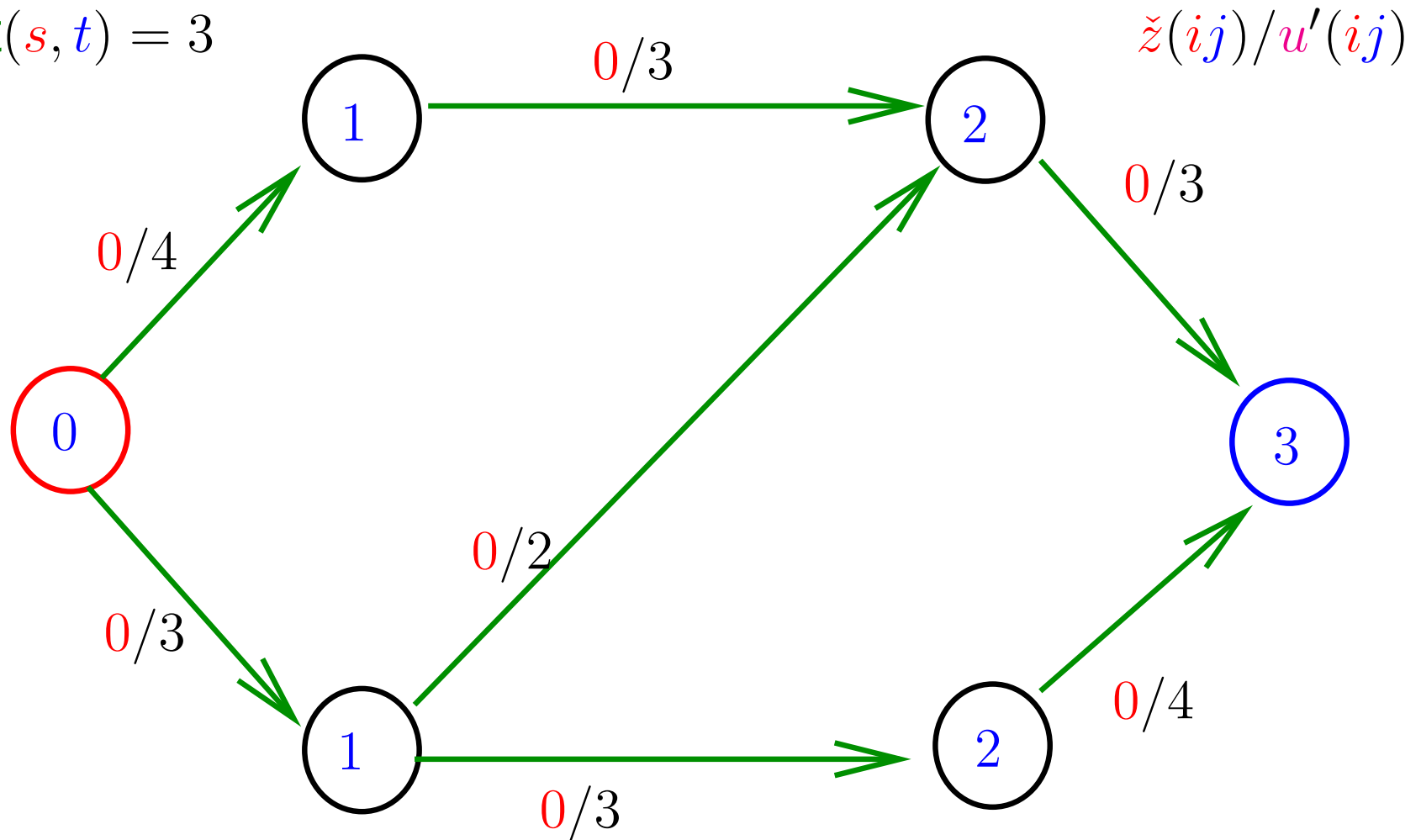
Rede residual 1

$\text{dist}(s, t) = 3$



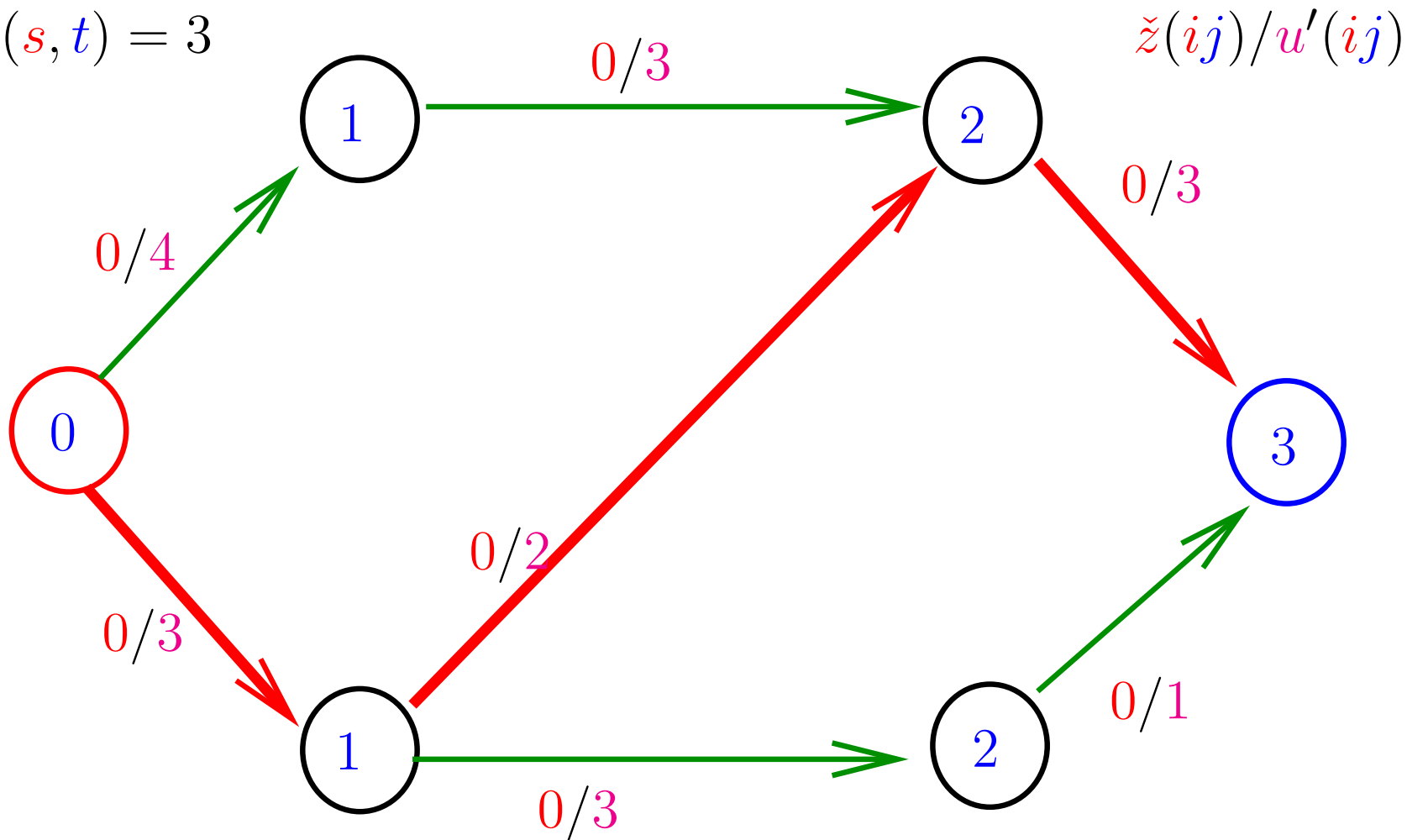
Rede em camadas 1

$\text{dist}(s, t) = 3$



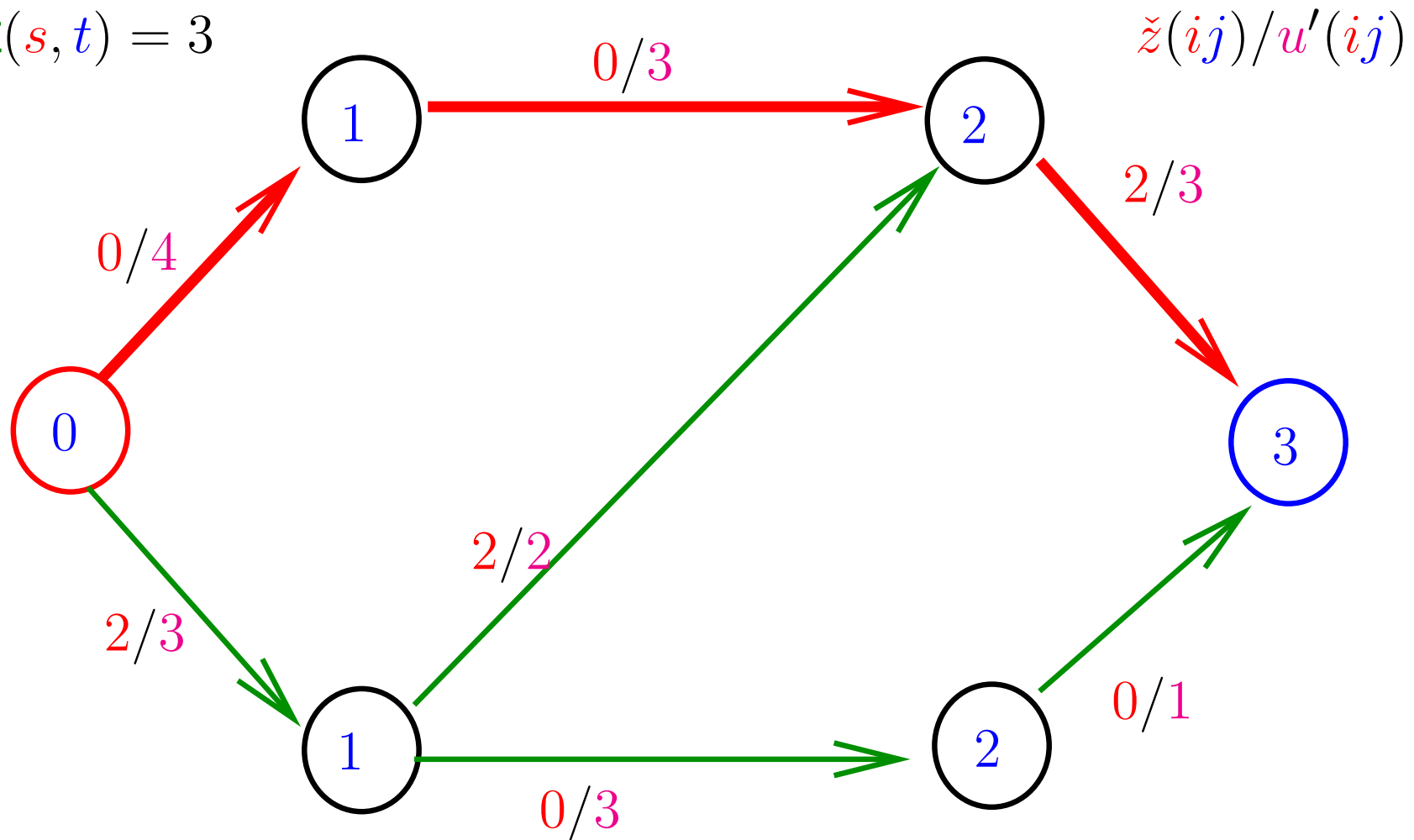
Caminho de incremento 1

$$\text{dist}(s, t) = 3$$



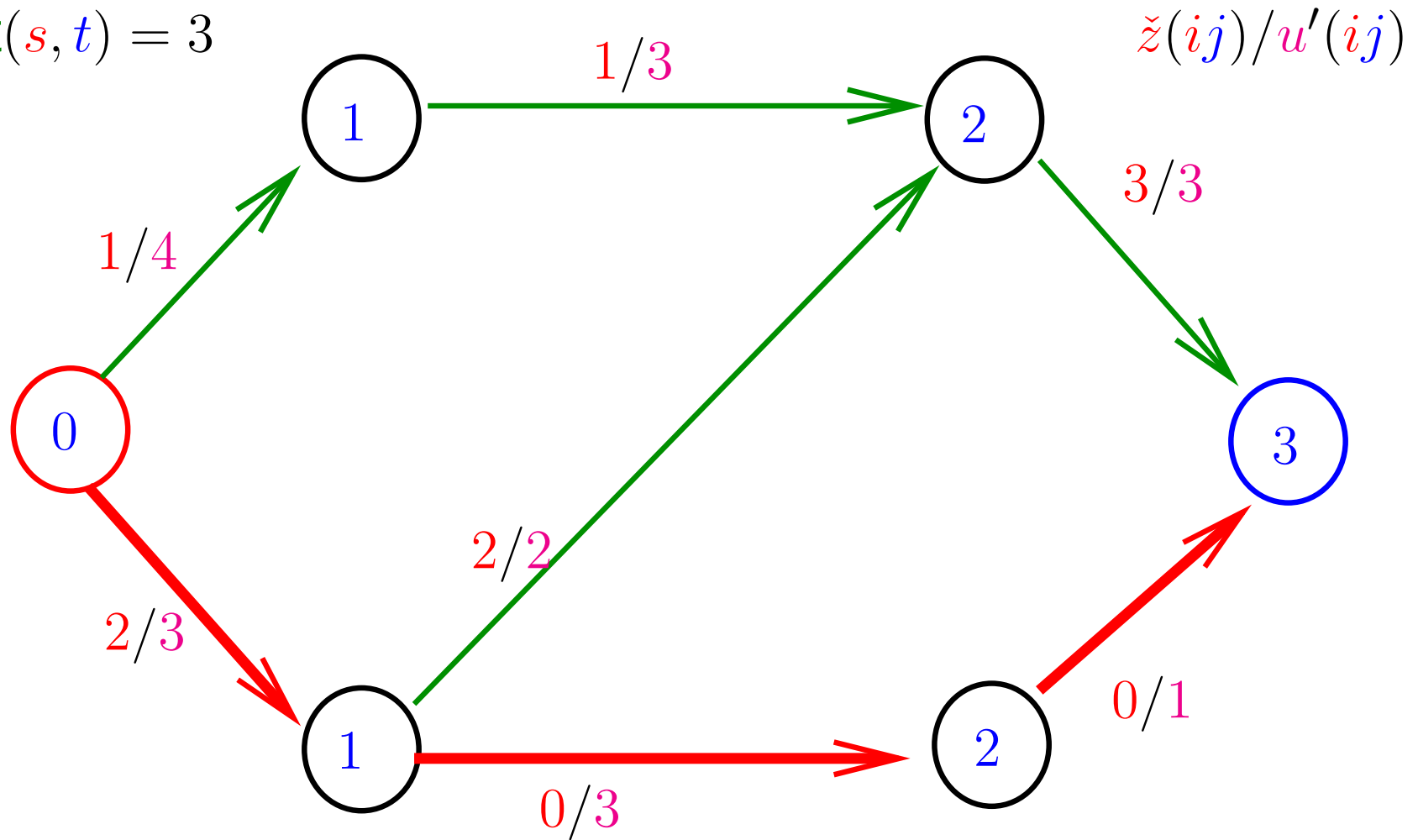
Caminho de incremento 2

$\text{dist}(s, t) = 3$



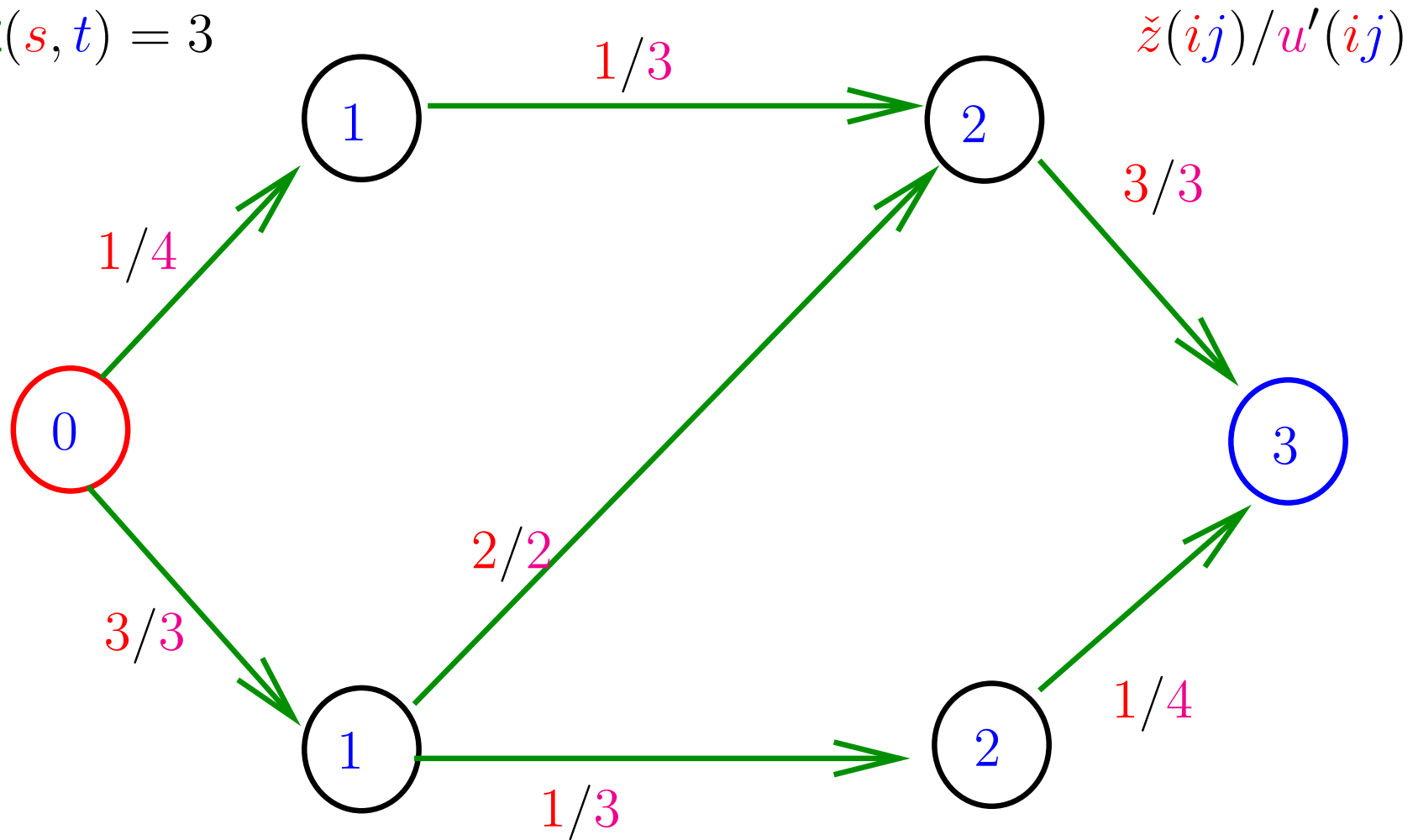
Caminho de incremento 3

$\text{dist}(s, t) = 3$



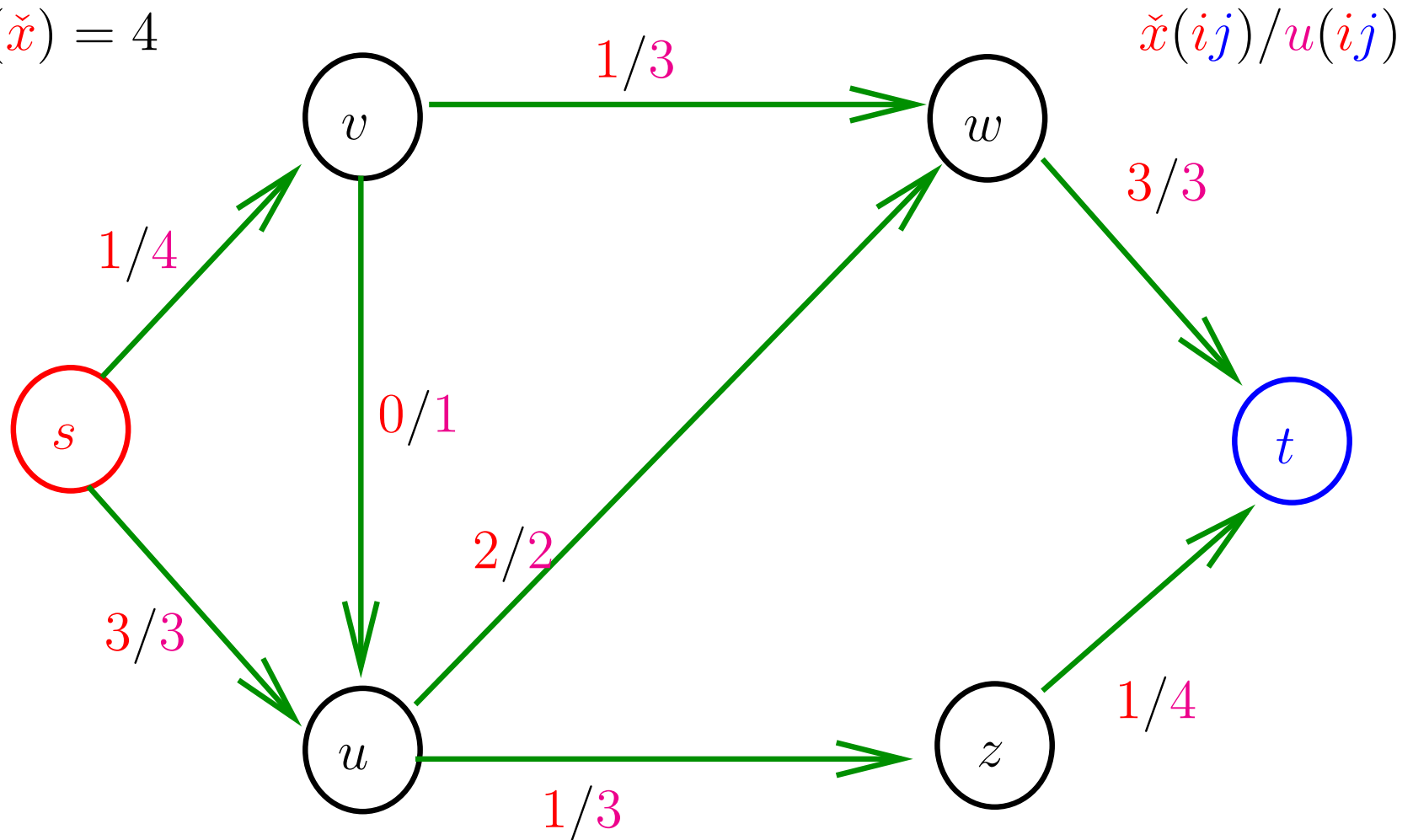
Fluxo bloqueador 1

$\text{dist}(s, t) = 3$



Rede 2

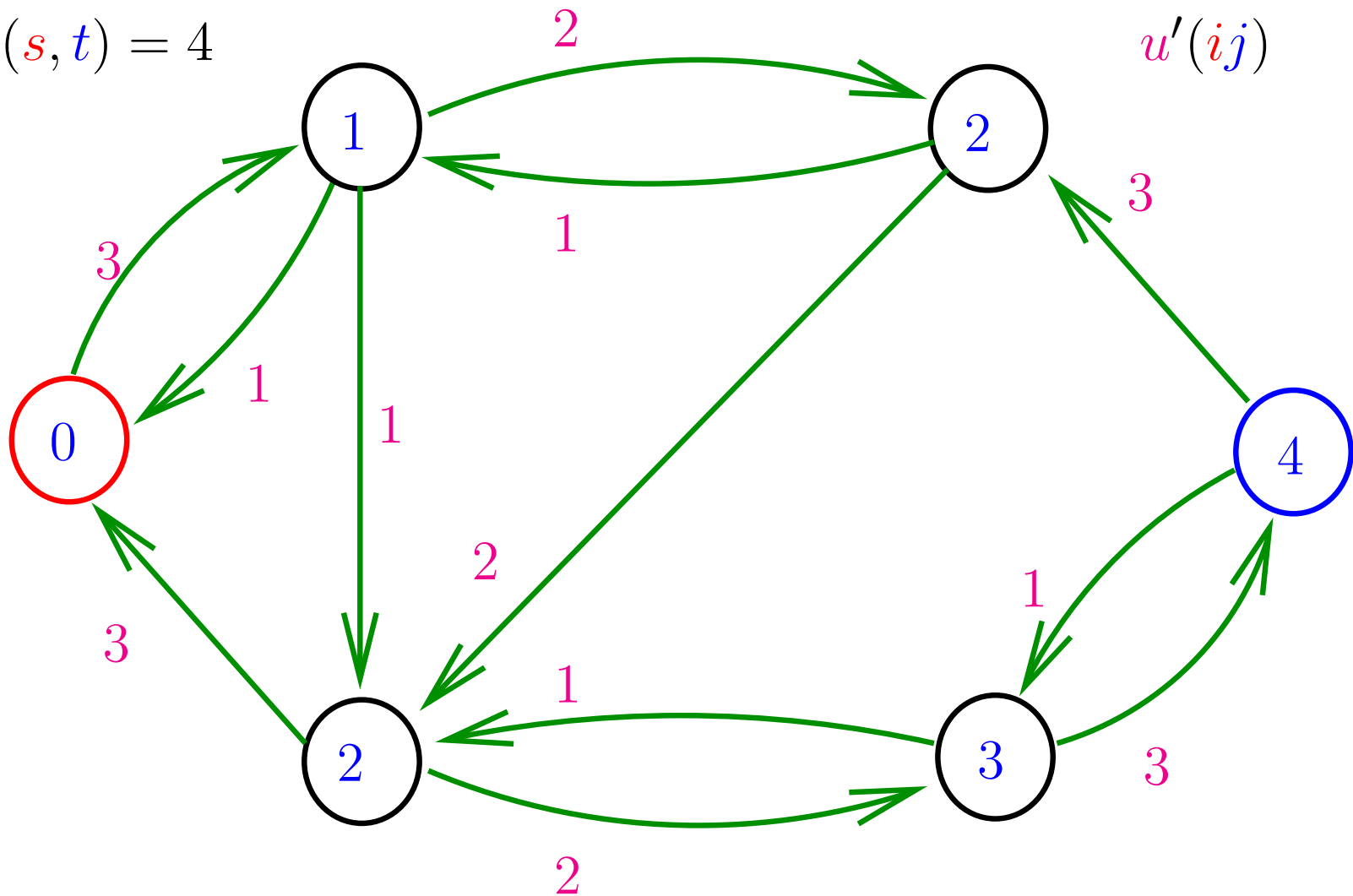
$$\text{val}(\tilde{x}) = 4$$



Arcos ausentes têm **capacidade** nula.

Rede residual 2

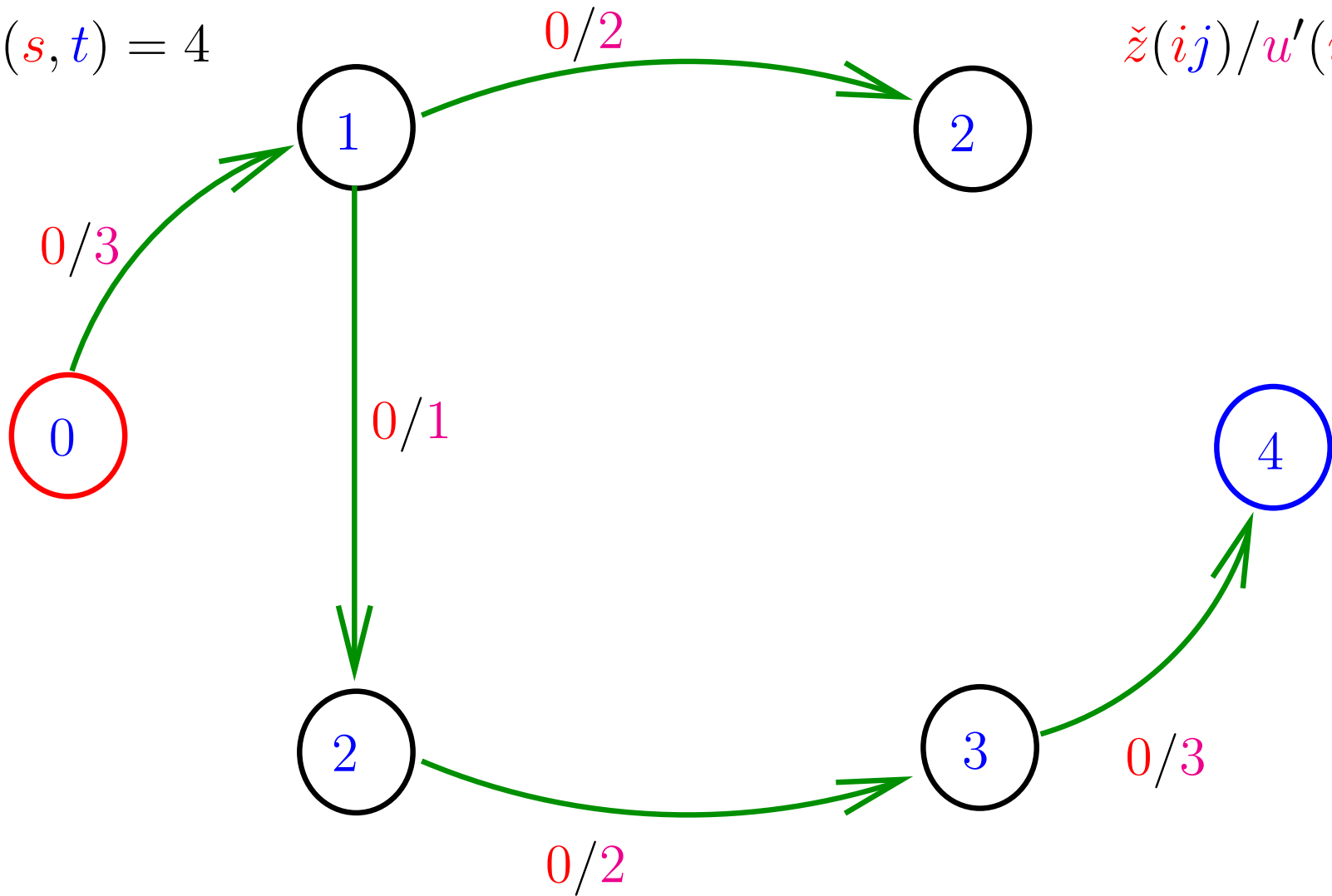
$$\text{dist}(s, t) = 4$$



Rede em camadas 2

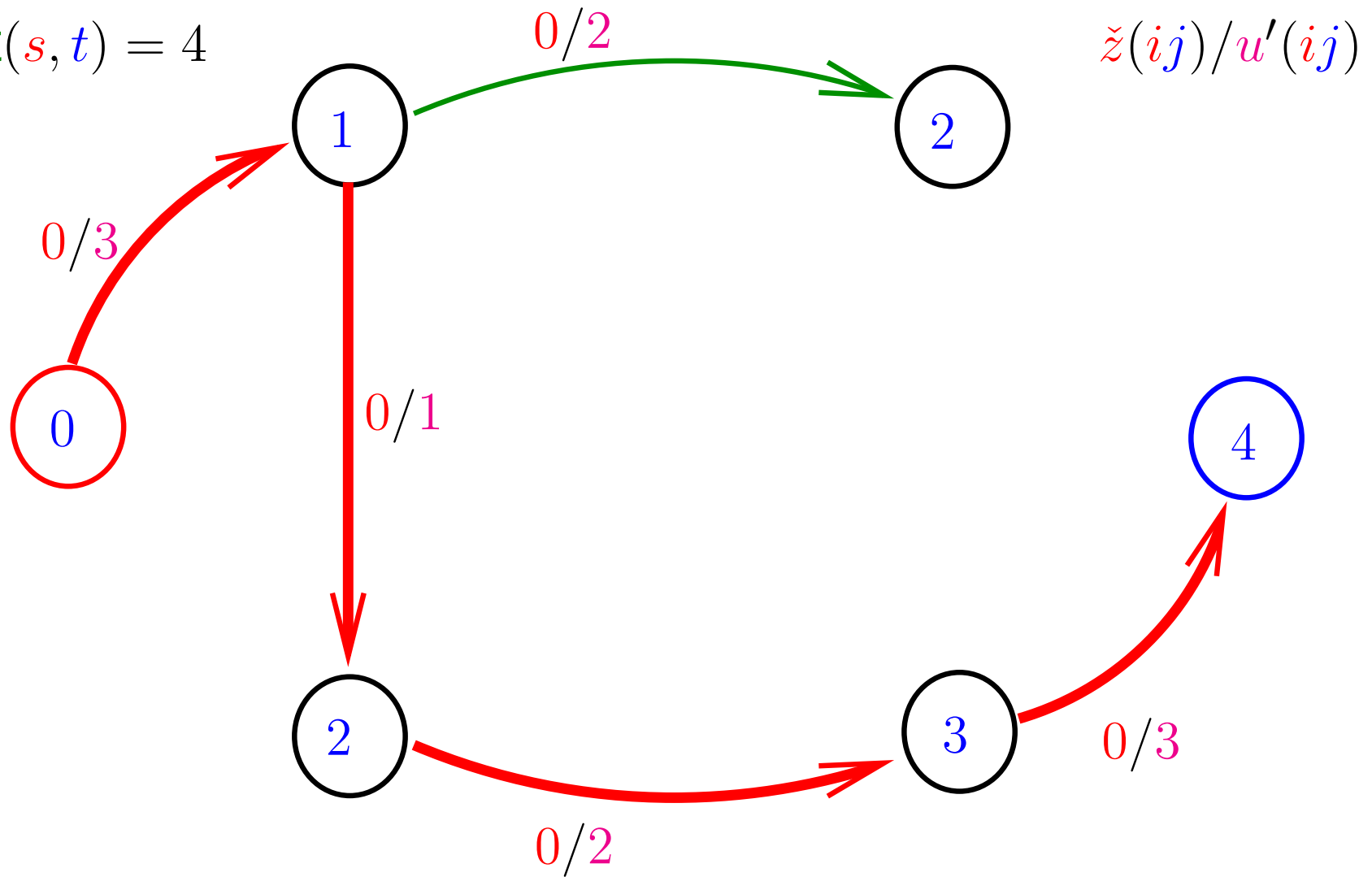
$$\text{dist}(s, t) = 4$$

$$\tilde{z}(ij)/u'(ij)$$



Caminho de incremento 4

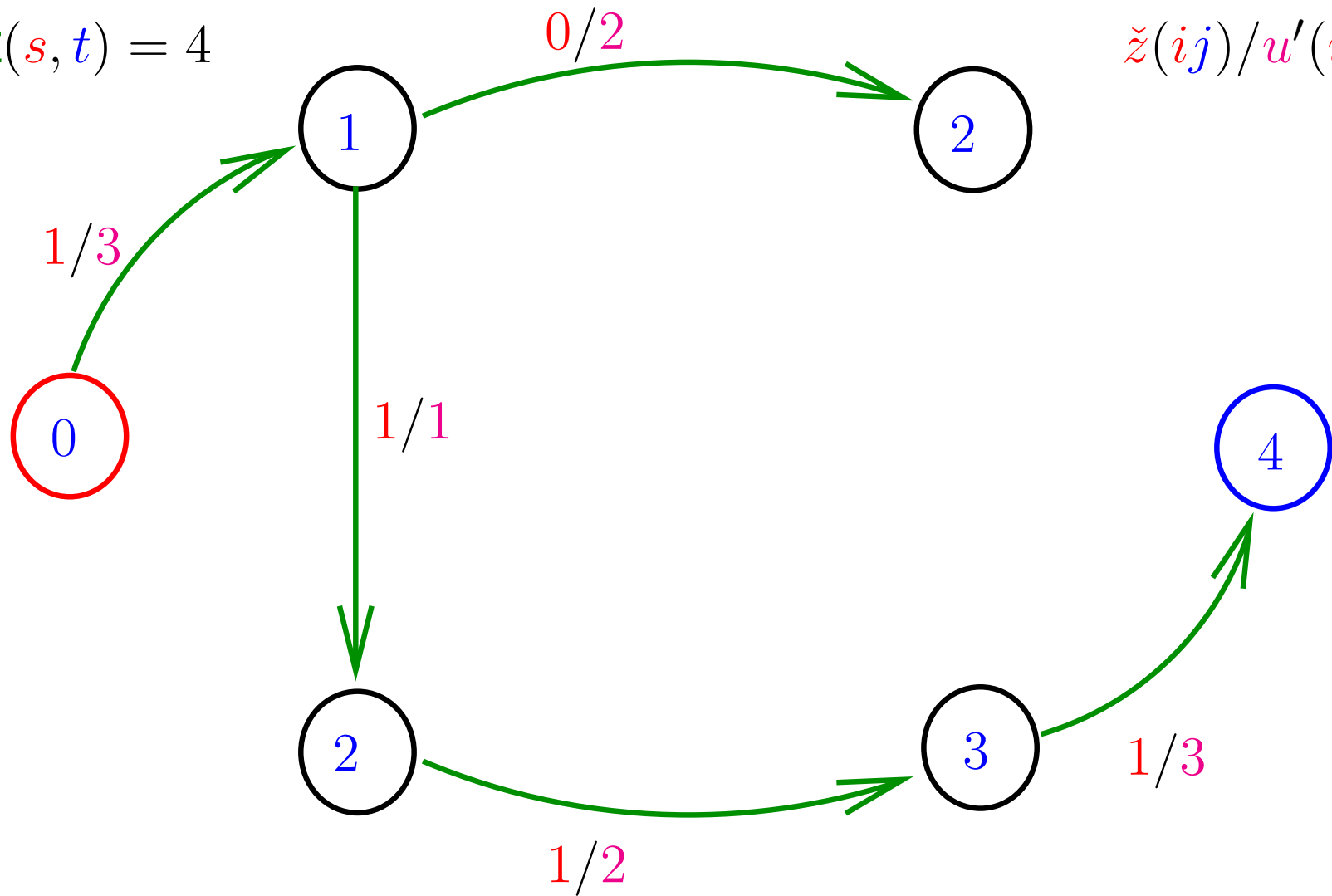
$\text{dist}(s, t) = 4$



Fluxo bloqueador 2

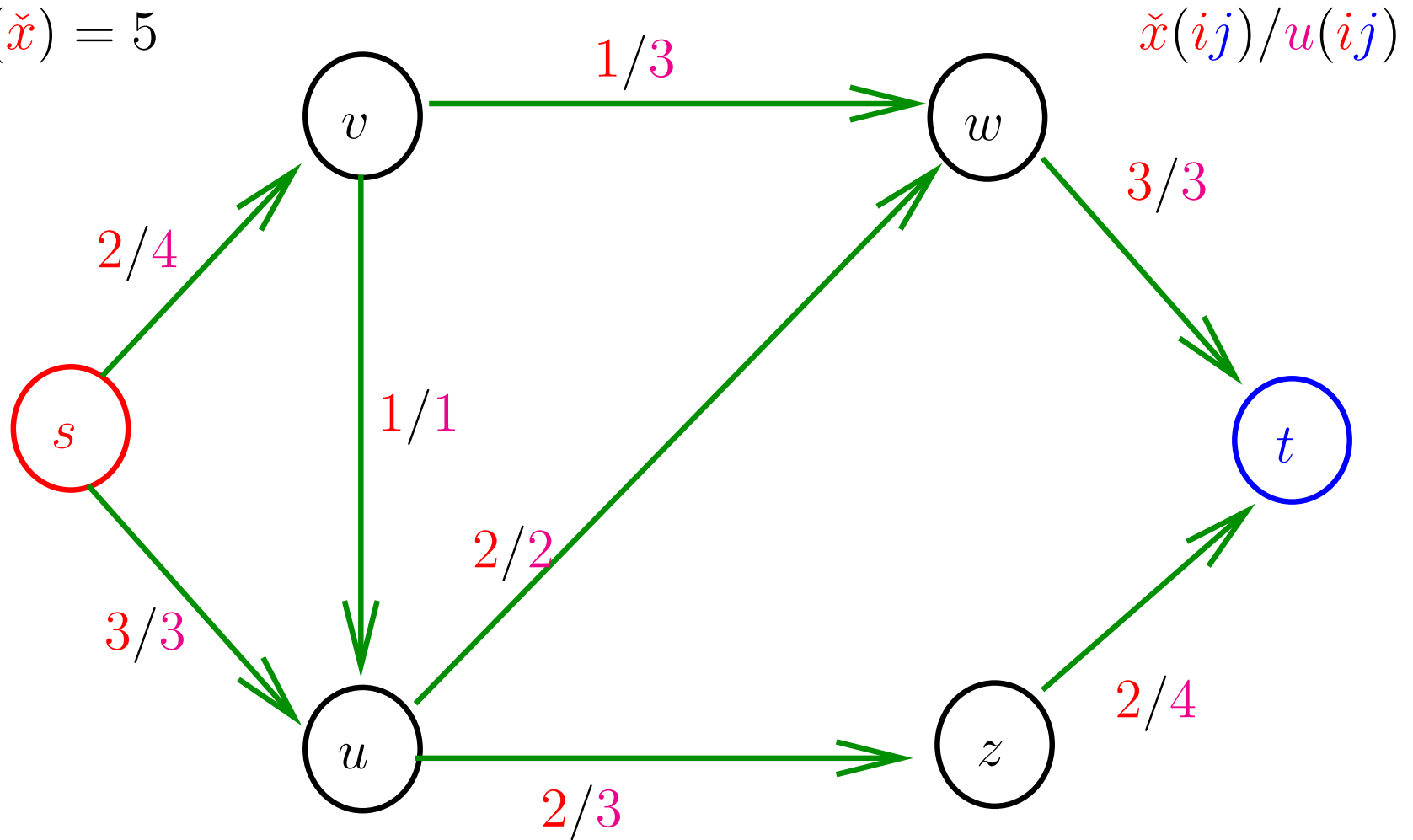
$\text{dist}(s, t) = 4$

$\tilde{z}(ij)/u'(ij)$



Rede 3

$$\text{val}(\tilde{x}) = 5$$

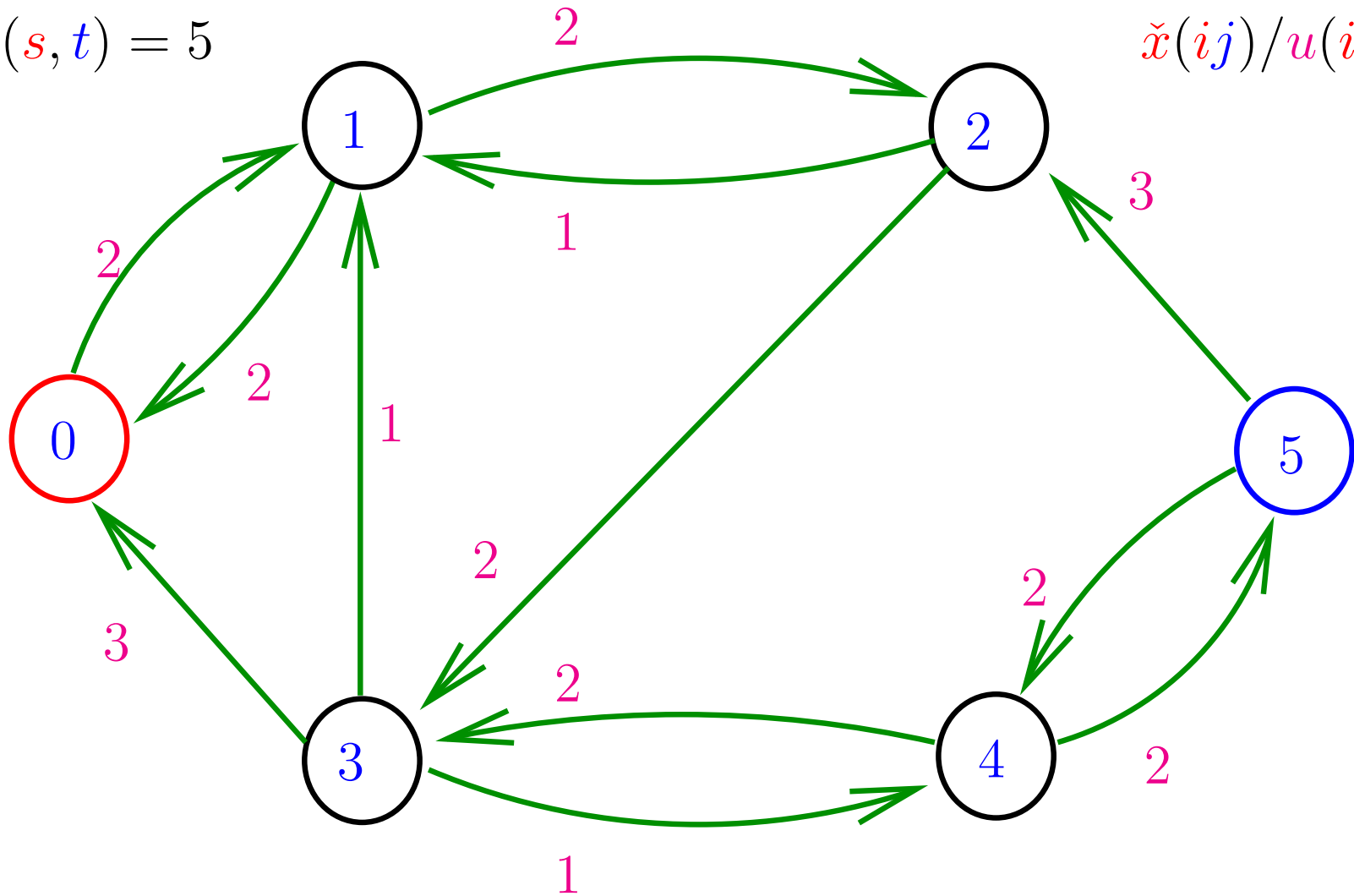


Arcos ausentes têm **capacidade** nula.

Rede residual 3

$\text{dist}(s, t) = 5$

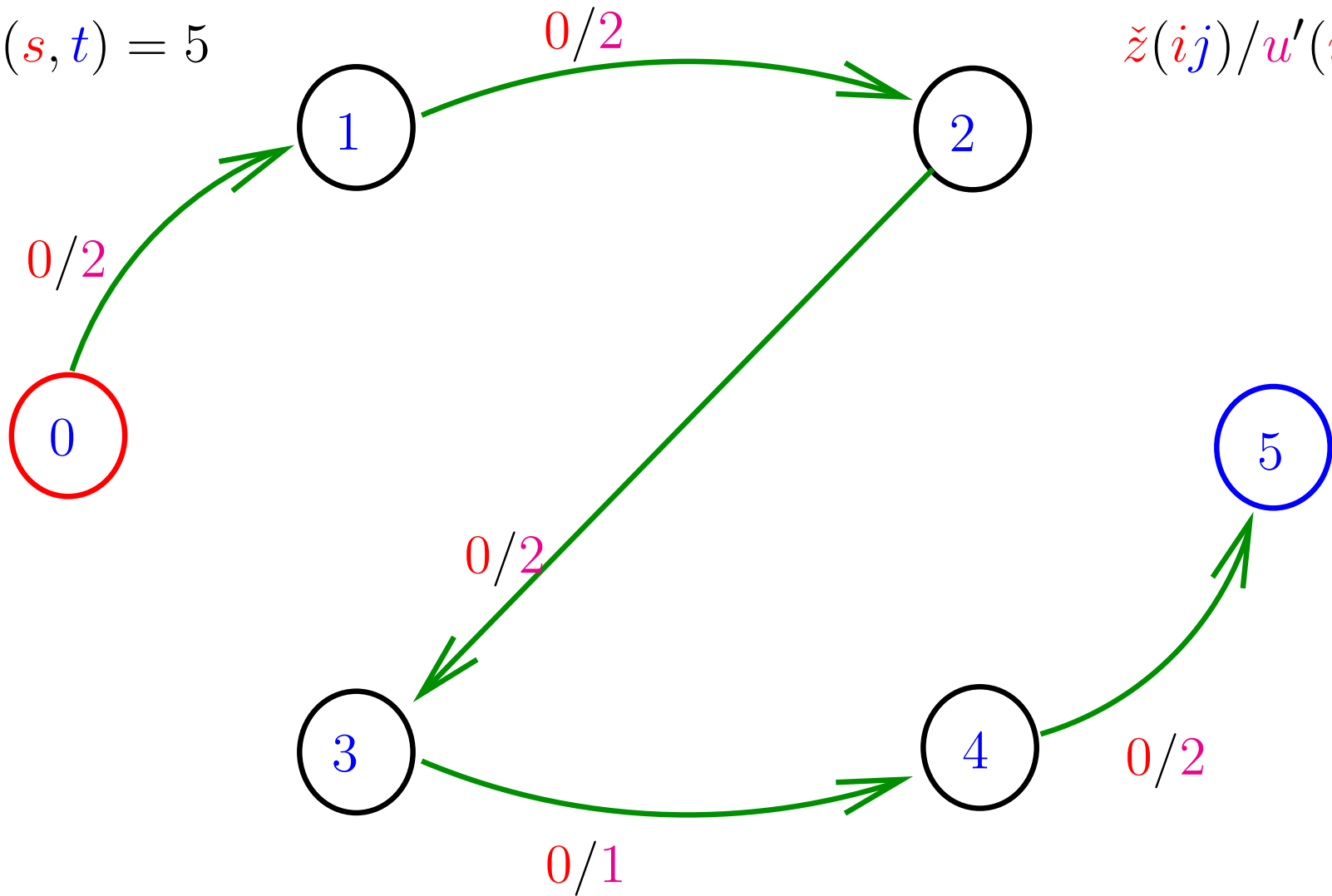
$\check{x}(ij)/u(ij)$



Rede em camadas 3

$$\text{dist}(s, t) = 5$$

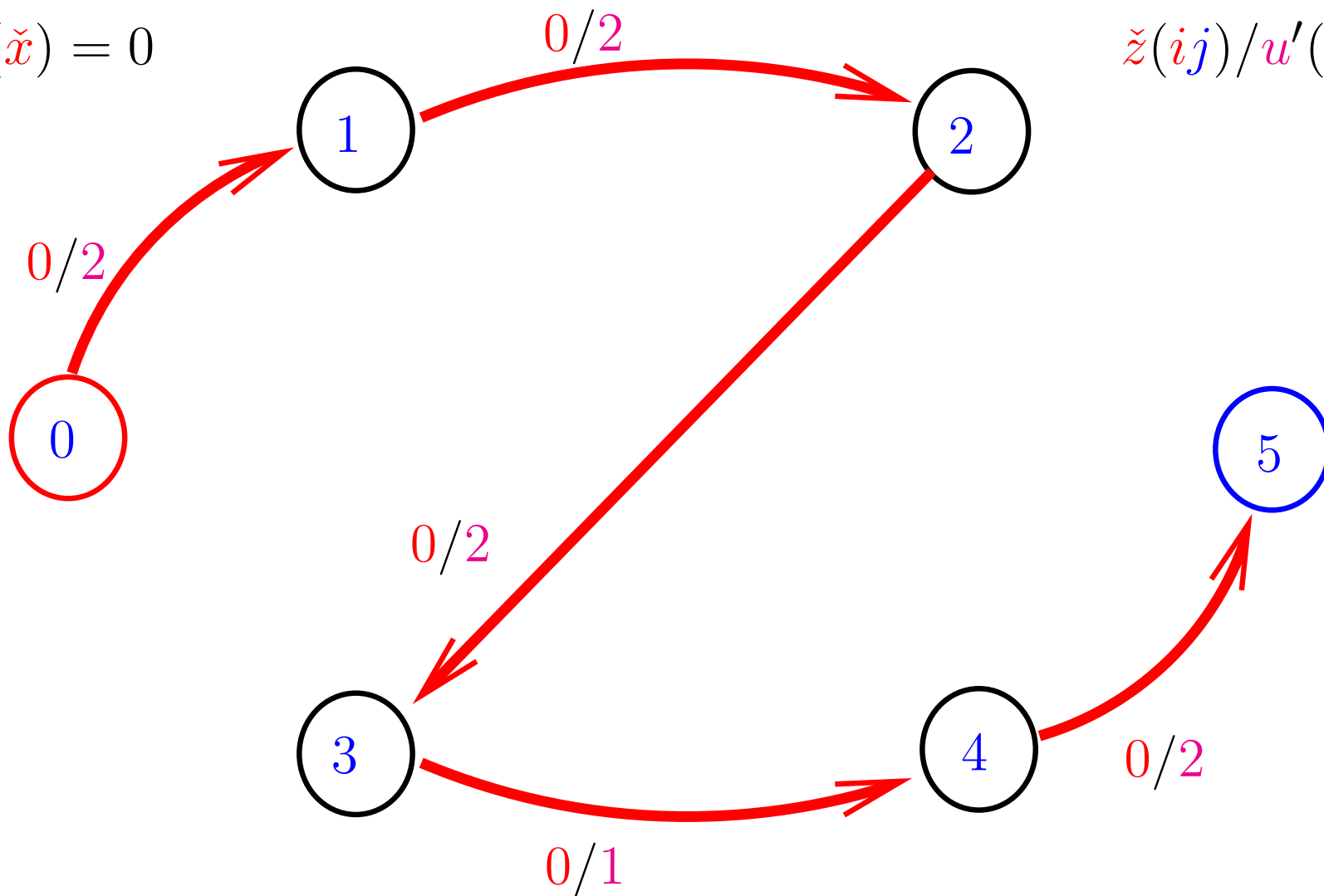
$$\tilde{z}(ij)/u'(ij)$$



Caminho de incremento 5

$$\text{val}(\check{x}) = 0$$

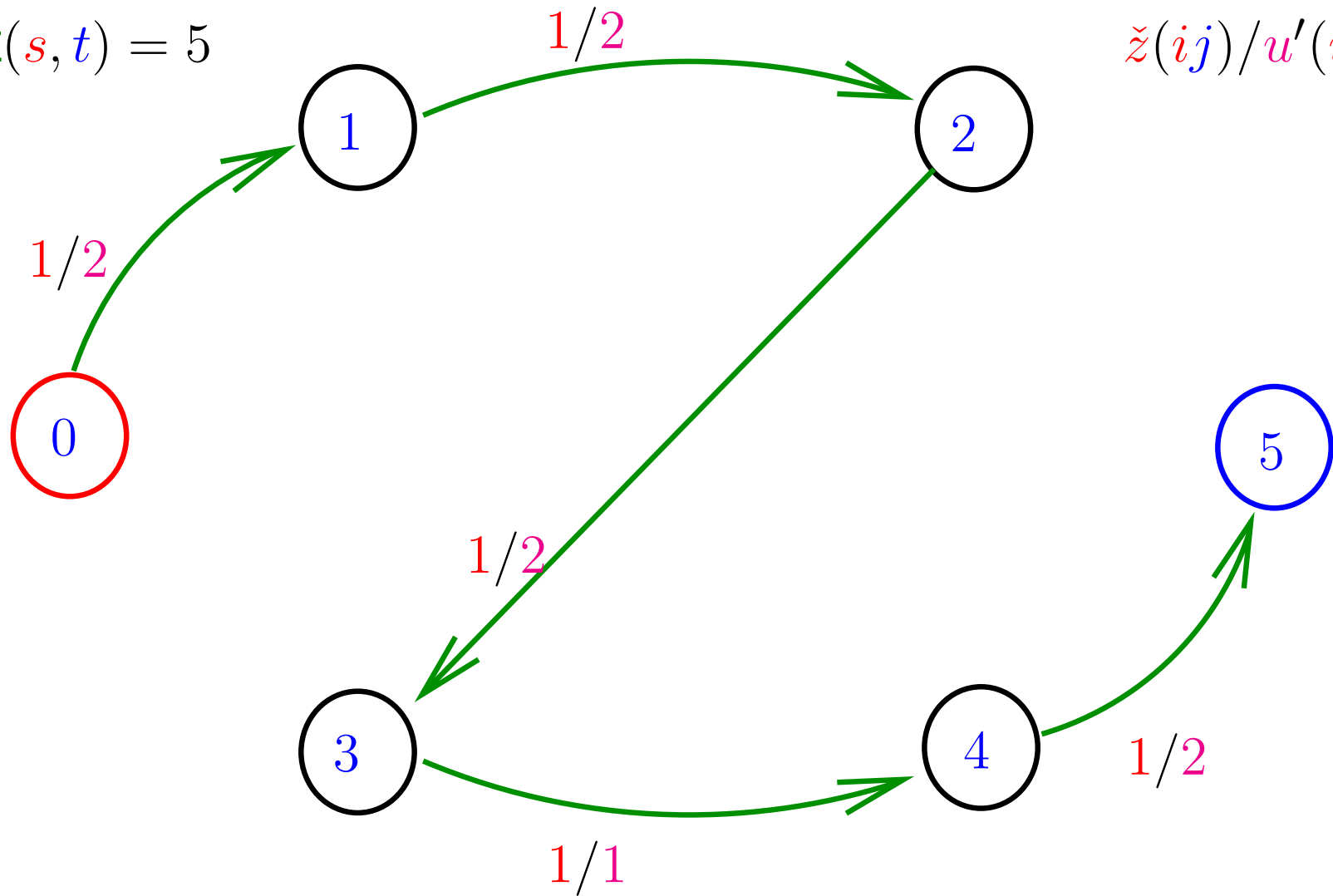
$$\check{z}(ij)/u'(ij)$$



Fluxo bloqueador 3

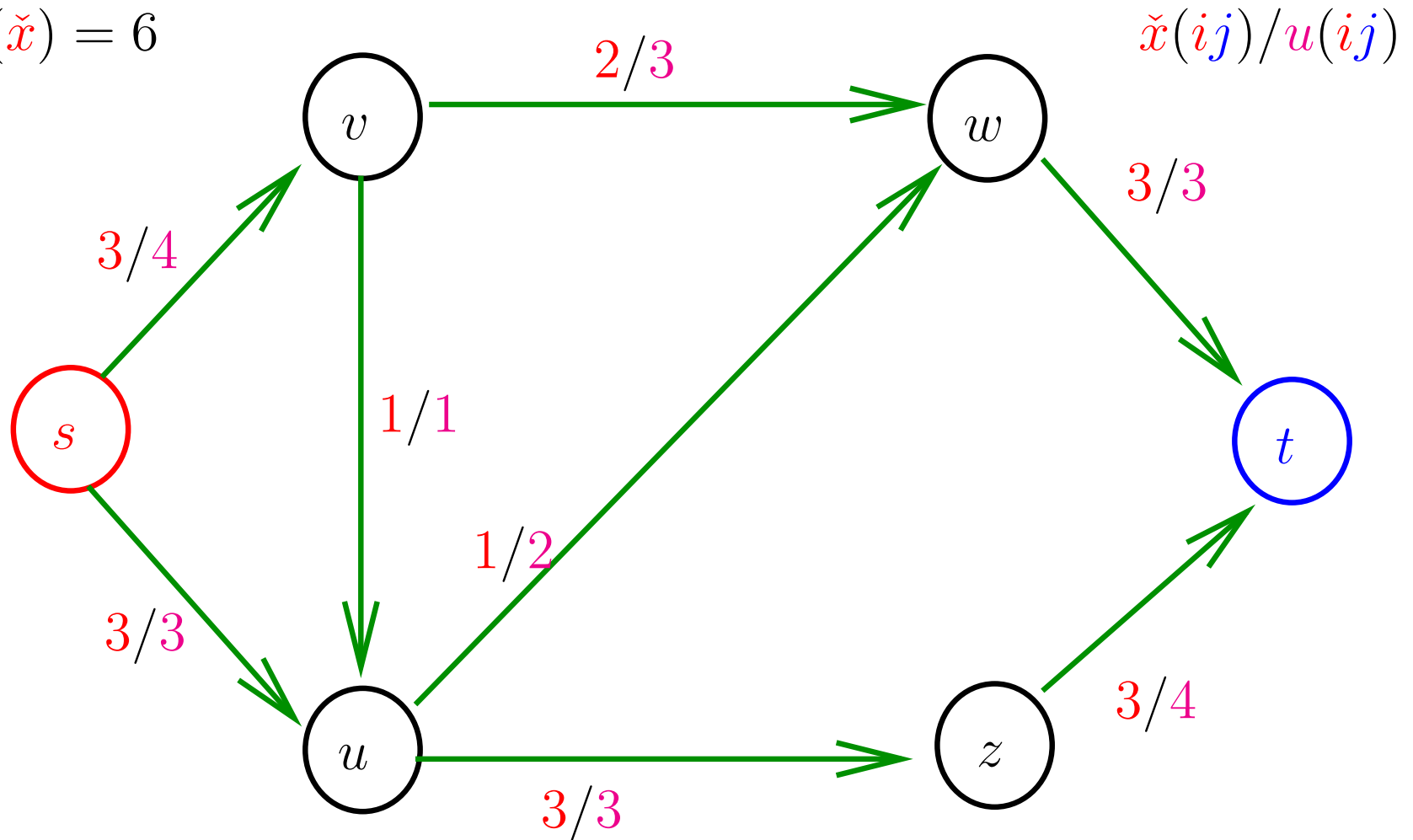
$\text{dist}(s, t) = 5$

$\tilde{z}(ij)/u'(ij)$



Rede 4

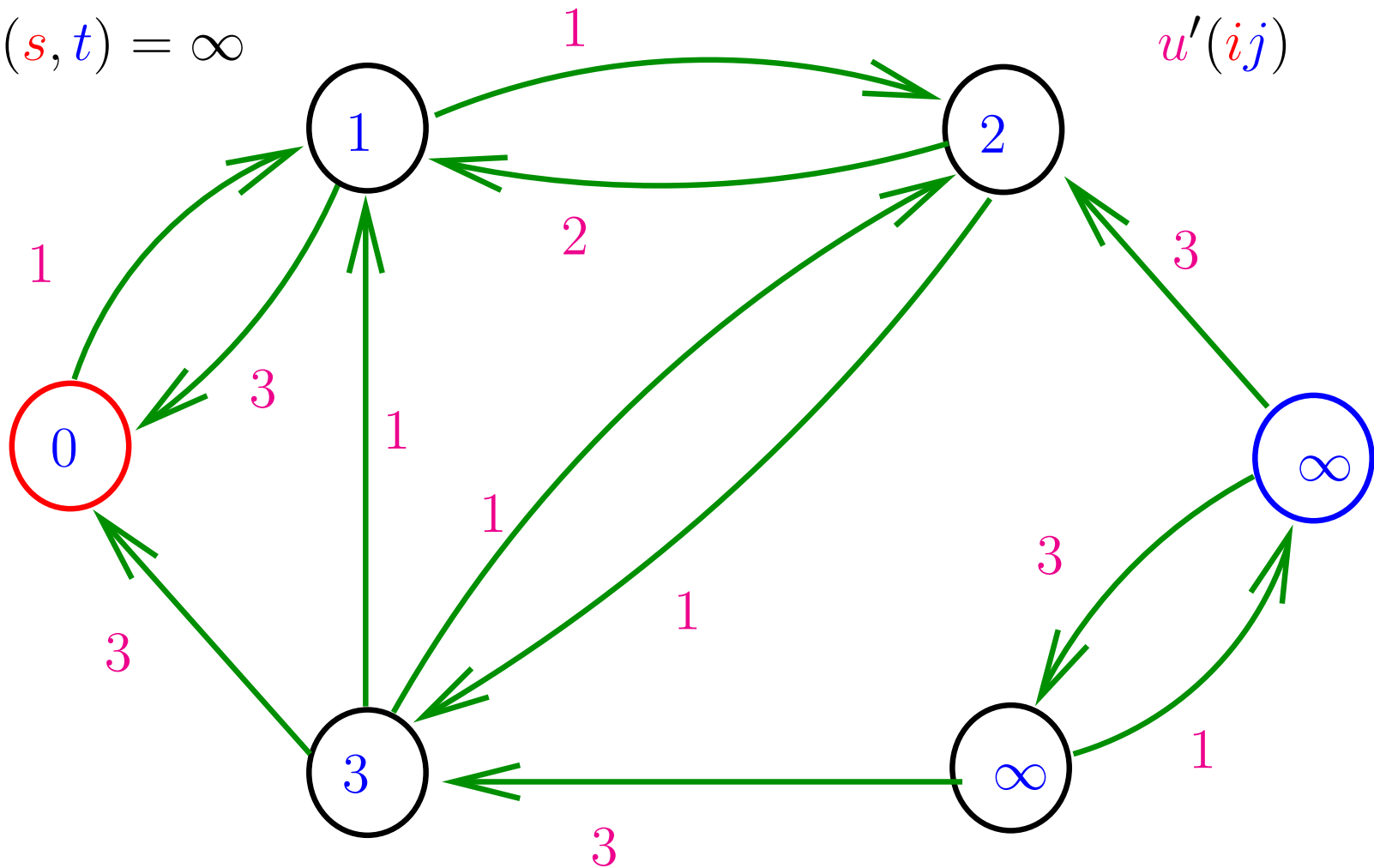
$$\text{val}(\tilde{x}) = 6$$



Arcos ausentes têm **capacidade** nula.

Rede residual 4

$$\text{dist}(s, t) = \infty$$

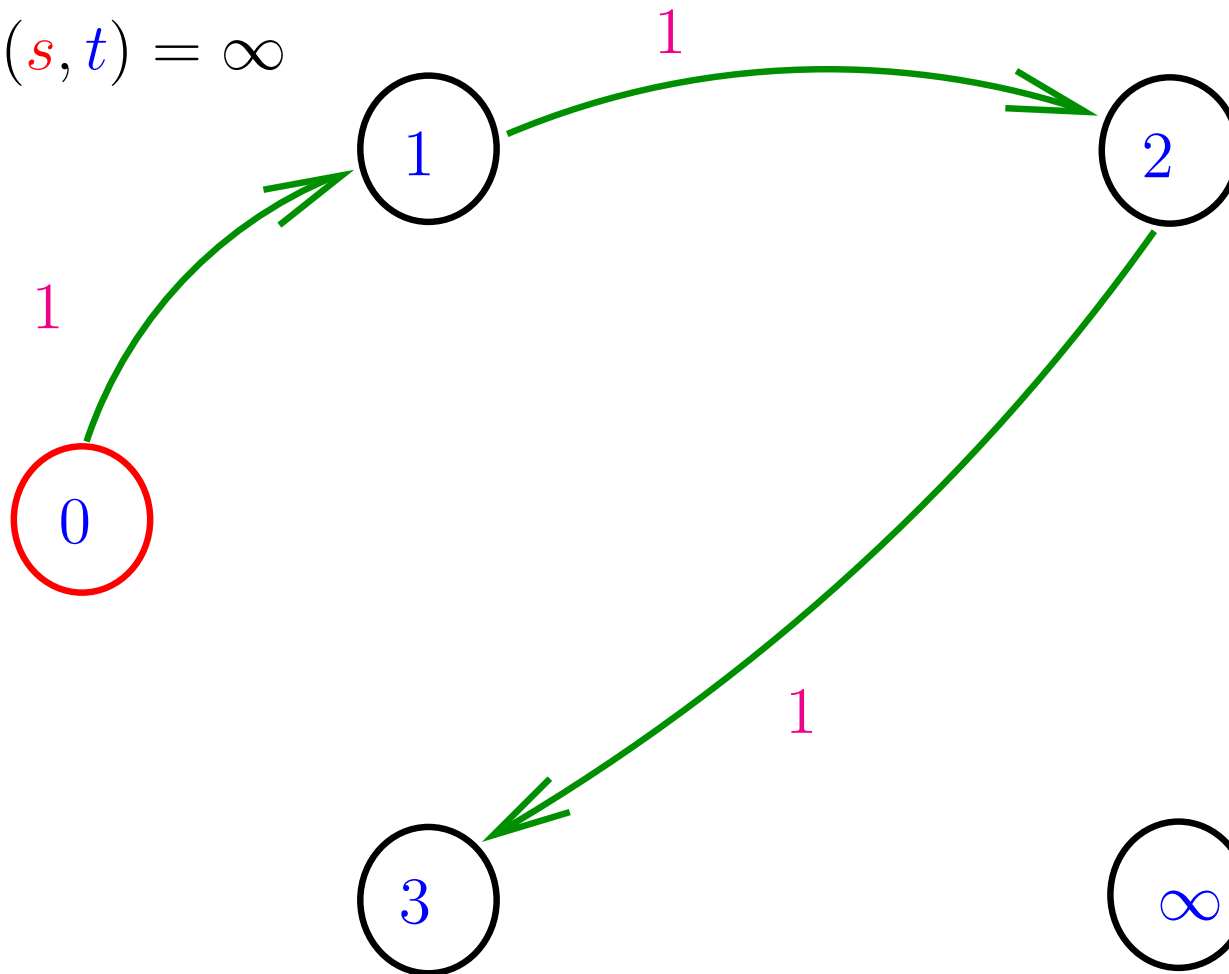


$$u'(ij)$$

Rede em camadas 4

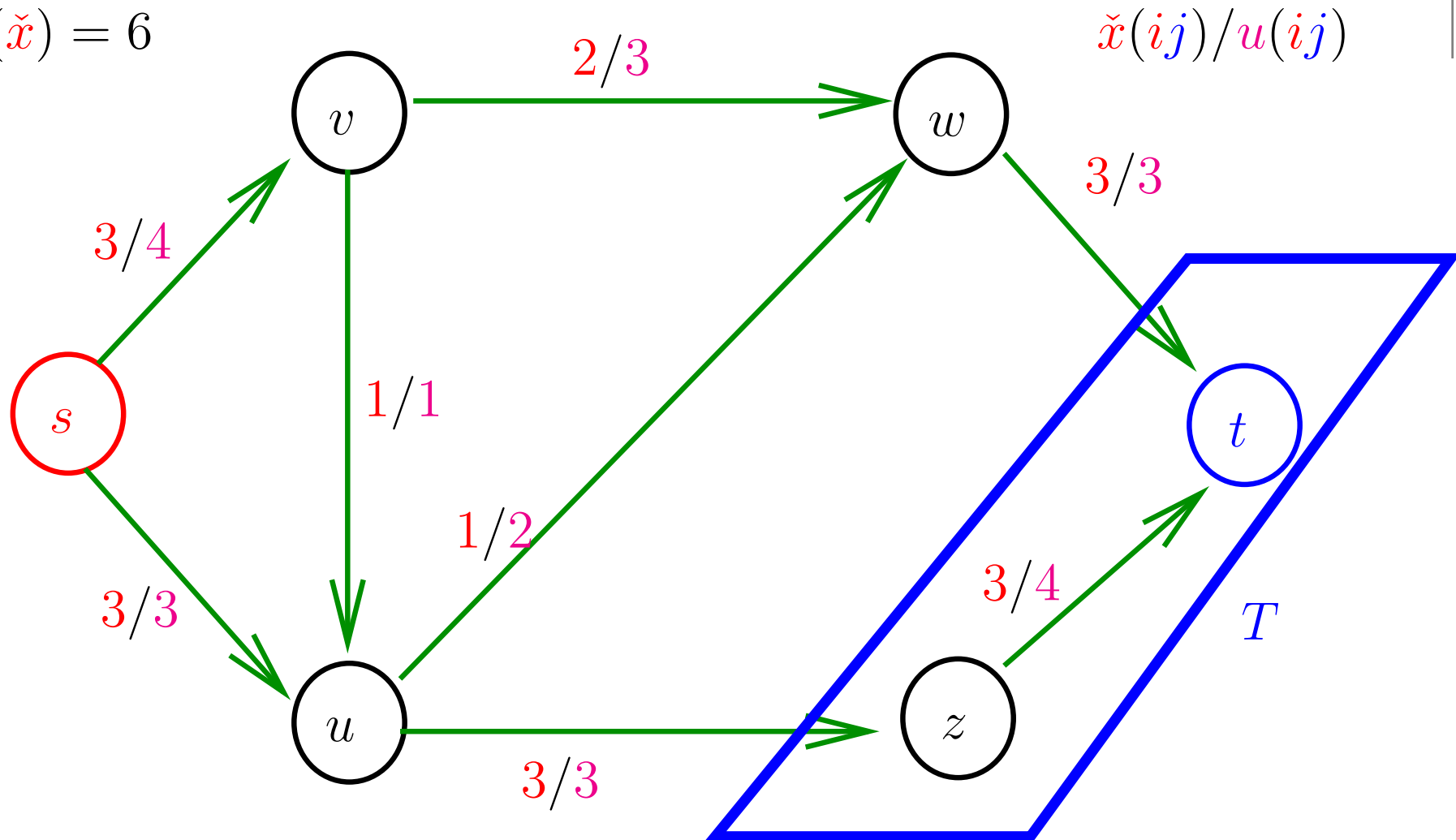
$$\text{dist}(s, t) = \infty$$

$$u'(ij)$$



Fluxo máximo e corte mínimo

$$\text{val}(\check{x}) = 6$$



Algoritmo de Dinits

Método dos “fluxos bloqueadores”

PASSO BLOQUEADOR:

encontre um **fluxo bloqueador** $\tilde{z}$ na “rede em camadas” **do fluxo corrente** $\tilde{x}$.

Intere novamente com $\tilde{x} + \tilde{z}$ no papel de $\tilde{x}$.

Na descrição do algoritmo de Dinits que veremos a rede em camadas será representada **implicitamente** por um 1-potencial $(s, *)$ -ótimo no grafo residual $(N, A_{\tilde{x}})$.

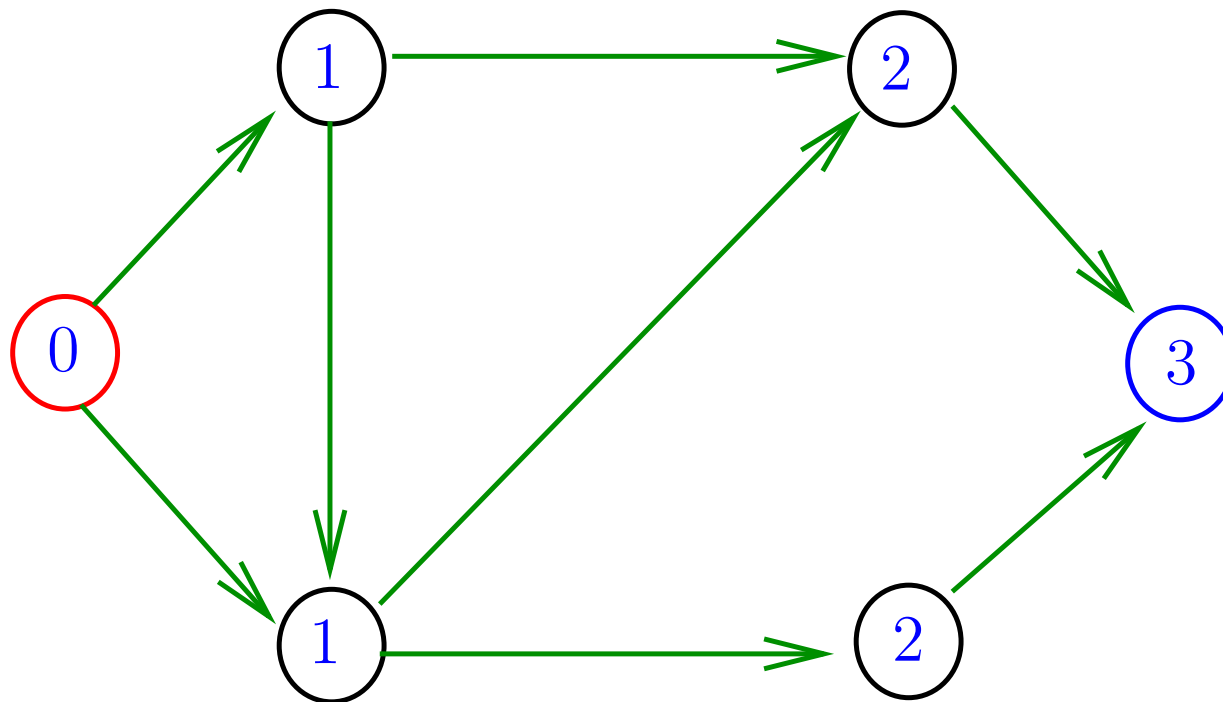
Observação: pode-se usar um 1-potencial $(*, t)$ -ótimo ao invés de $(s, *)$ -ótimo.

Potencial ótimo

Dado um nó s , um 1-potencial y é $(s, *)$ -ótimo se:

- existe um caminho P de s a i tal que $|P| = y(i) - y(s)$ ou
- não existe um caminho de s a i e $y(i) - y(s) \geq n$.

Um arco ij é **justo** se $y(j) - y(i) = 1$.



Potencial ótimo

POTENCIAL-ÓTIMO (N, A, s)

```
1  para cada  $i$  em  $N$  faça
2       $y(i) \leftarrow n$ 
3   $y(s) \leftarrow 0$ 
4   $L \leftarrow \langle s \rangle$   $\triangleright L$  funciona como uma fila
5  enquanto  $L \neq \langle \rangle$  faça
6      retire o primeiro elemento, digamos  $i$ , de  $L$ 
7      para cada arco  $ij$  em  $A(i)$  faça  $\triangleright ij$  entra em  $j$ 
8          se  $y(j) > y(i) + 1$   $\triangleright y(j) = n$ 
9              então  $y(j) \leftarrow y(i) + 1$ 
10             acrescente  $j$  ao final de  $L$ 
11  devolva  $y$ 
```

Consumo de tempo: $O(n + m)$

Algoritmo de Dinits

DINITs (N, A, u, s, t)

1 $\tilde{x} \leftarrow 0$

2 **repita**

3 $A_{\tilde{x}} \leftarrow \{ij \in A : \tilde{x}(ij) < u(ij)\}$

4 $y \leftarrow \text{POTENCIAL-ÓTIMO}(N, A_{\tilde{x}}, t)$

5 **se** $y(t) - y(s) < n$ **então**

6 $\tilde{x} \leftarrow \text{PASSO-BLOQUEADOR}(N, A_{\tilde{x}}, s, t, \tilde{x}, y)$

7 **até que** $y(t) - y(s) \geq n$

8 $x \leftarrow \text{FLUXO}(\tilde{x})$

9 $T \leftarrow \{j \in N : y(j) - y(s) \geq n\}$

10 **devolva** x e T

Invariantes

No início de cada iteração das linhas 3–6 vale que:

(i0) $\tilde{x}$ é um st -pseudofluxo que respeita u

(i1) y é um 1-potencial $(s, *)$ -ótimo no de $(N, A_{\tilde{x}})$

Número de iterações

Digamos que uma **fase** do algoritmo é uma execução das linhas 3–6.

Fato. O número de fases é $\leq n - 1$.

Demonstração (rascunho):

- $(N, A_k) :=$ rede residual construída da fase k (linha 3);
- $y_k :=$ 1-potencial $(s, *)$ -ótimo obtido na fase k (linha 4).
Podemos supor que $y_k(s) = 0$.

Para $k = 1, \dots, r - 1$ ($r =$ número de fases) vale que:

$$(\star) \ y_{k+1}(t) > y_k(t).$$

Isto implica que o número de fases é $\leq n - 1$.

Lembra da observação que ajudou?

Para cada arco ij em A_{k+1} vale que

$$(2A) \quad y_k(j) - y_k(i) \leq 1 \text{ se } ij \in A_k;$$

$$(2B) \quad y_k(j) - y_k(i) = -1 \text{ se } ij \notin A_k;$$

Demonstração de (2A) (rascunho).

Segue do fato de y_k ser um 1-potencial.

Demonstração de (2B) (rascunho).

Se $ij \notin A_k$, então $ji \in A_k$ e ji pertence a um caminho de incremento P utilizado na linha 6 pelo algoritmo **PASSO-BLOQUEADOR** durante a fase k .

Logo, $y_k(i) - y_k(j) = 1 \Rightarrow y_k(j) - y_k(i) = -1$.

Rascunho da demonstração de (★)

Seja $P = \langle s = i_0, i_1, i_2, \dots, i_{q-1}, i_q = t \rangle$ um st -caminho mínimo em (N, A_{k+1}) .

Temos que

$$\begin{aligned} y_{k+1}(t) &= |P| \\ &\geq (y_k(t) - y_k(i_{q-1})) + \dots + (y_k(i_1) - y_k(s)) \\ &= y_k(t) - y_k(s) \\ &= y_k(t), \end{aligned}$$

onde a desigualdade é devida a (2A) e (2B).

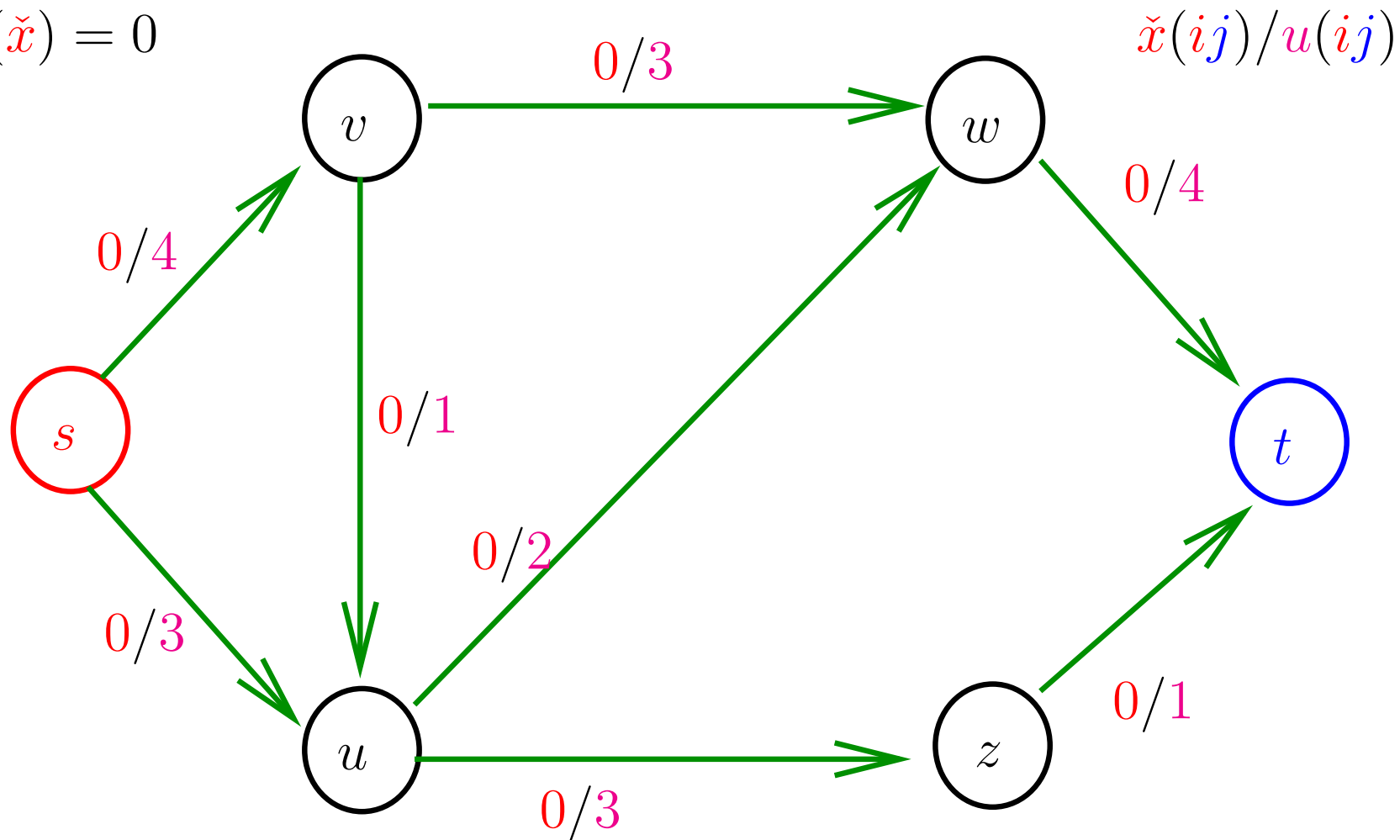
Como pelo menos um arco em cada st -caminho mínimo em (N, A_k) foi saturado no passo bloqueador e não está em (N, A_{k+1}) , então $y_{k+1}(t) > y_k(t)$ (por (2B)).

Conclusão

O número de fases do algoritmo **DINITS** é $\leq n - 1$.

Rede

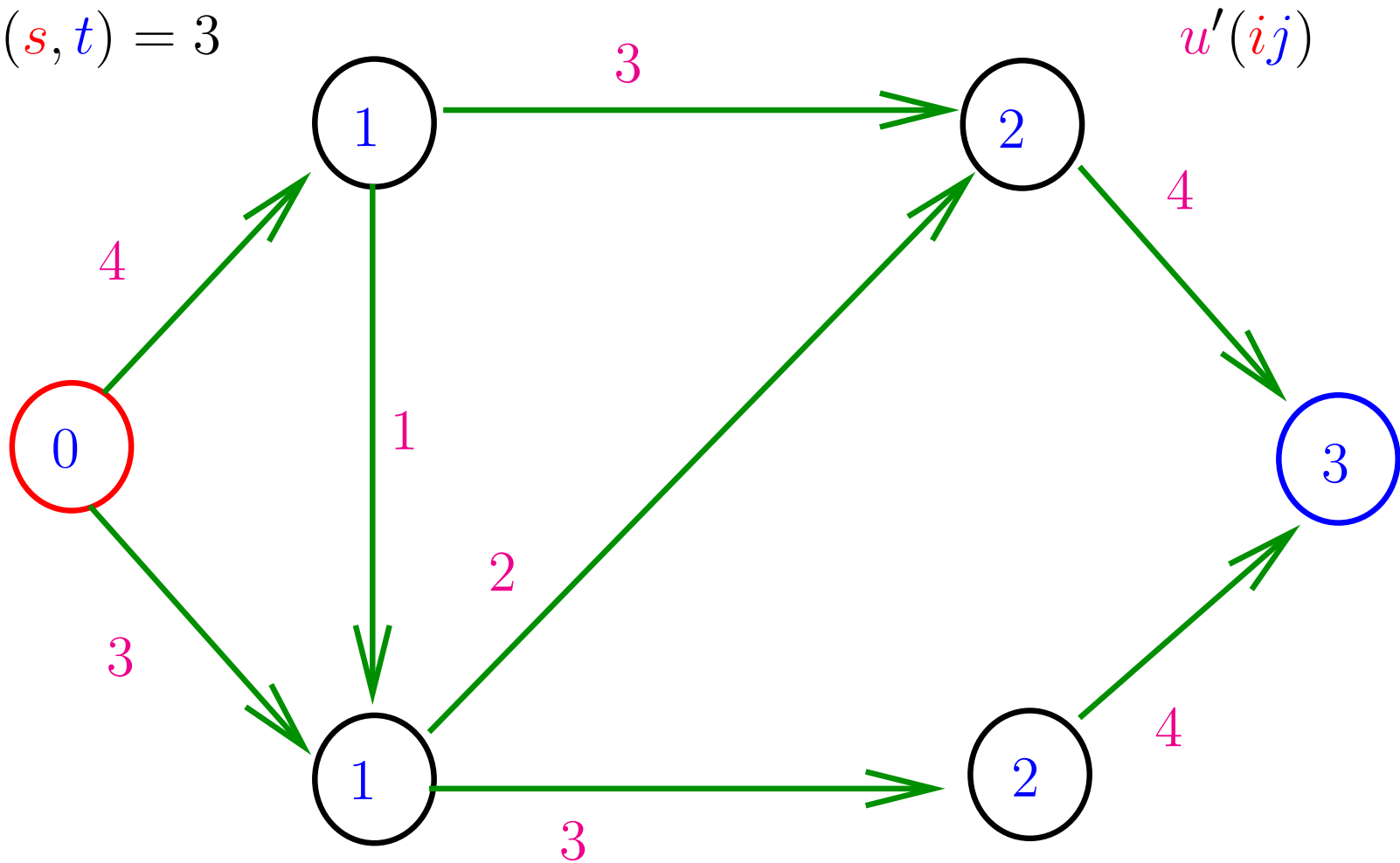
$$\text{val}(\tilde{x}) = 0$$



Arcos ausentes têm **capacidade** nula.

Rede residual

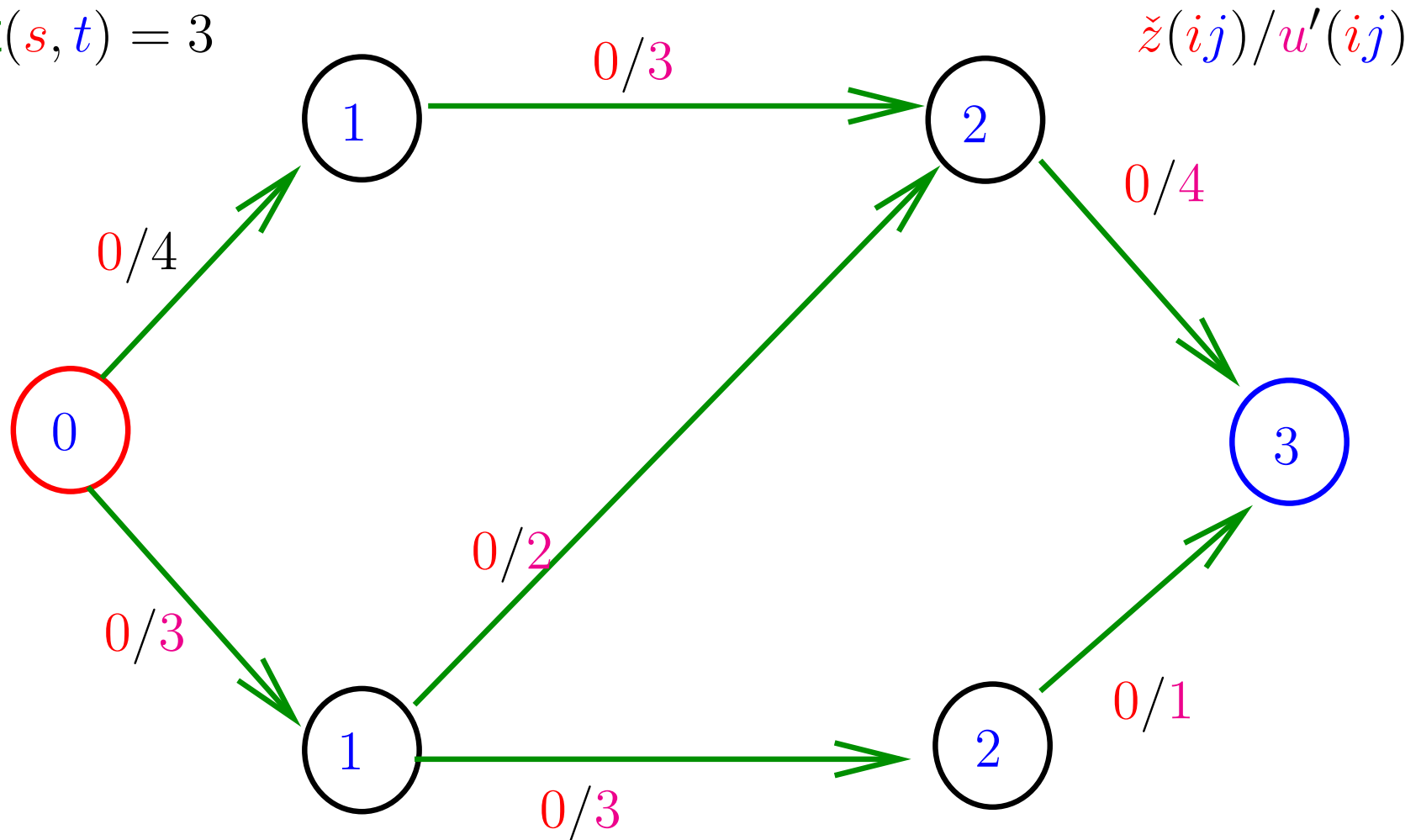
$$\text{dist}(s, t) = 3$$



$$u'(ij)$$

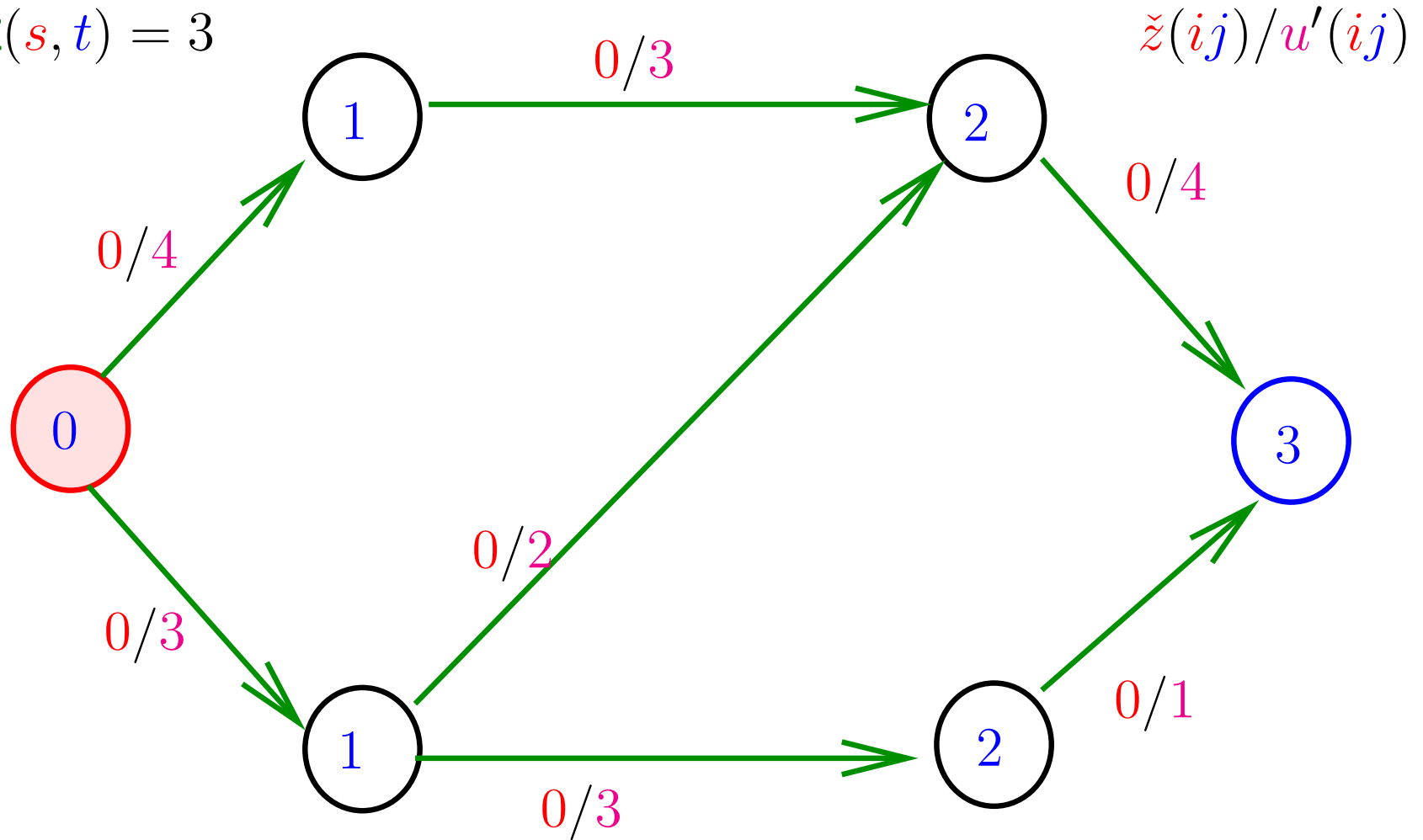
Rede em camadas

$\text{dist}(s, t) = 3$



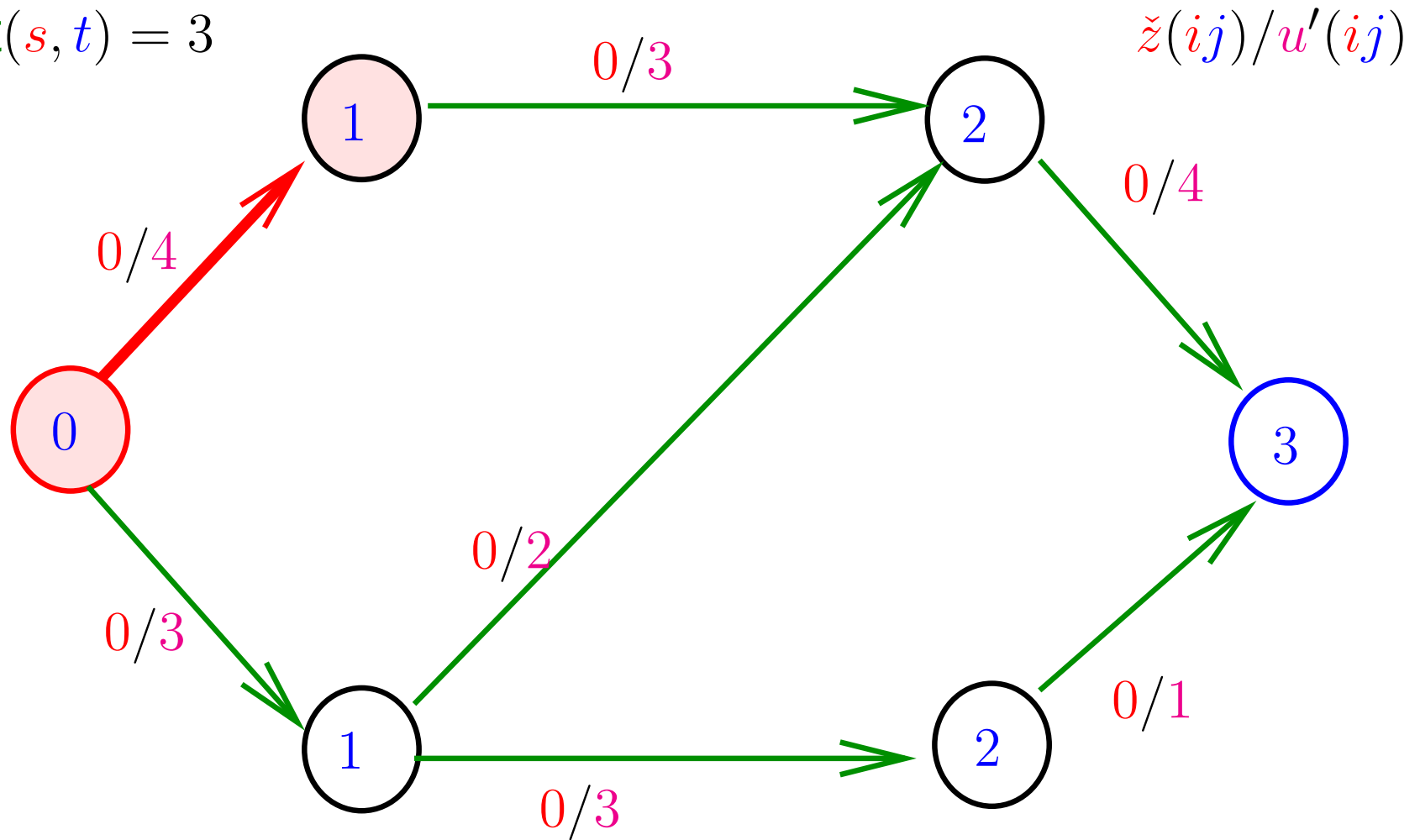
Passo bloqueador (0)

$\text{dist}(s, t) = 3$



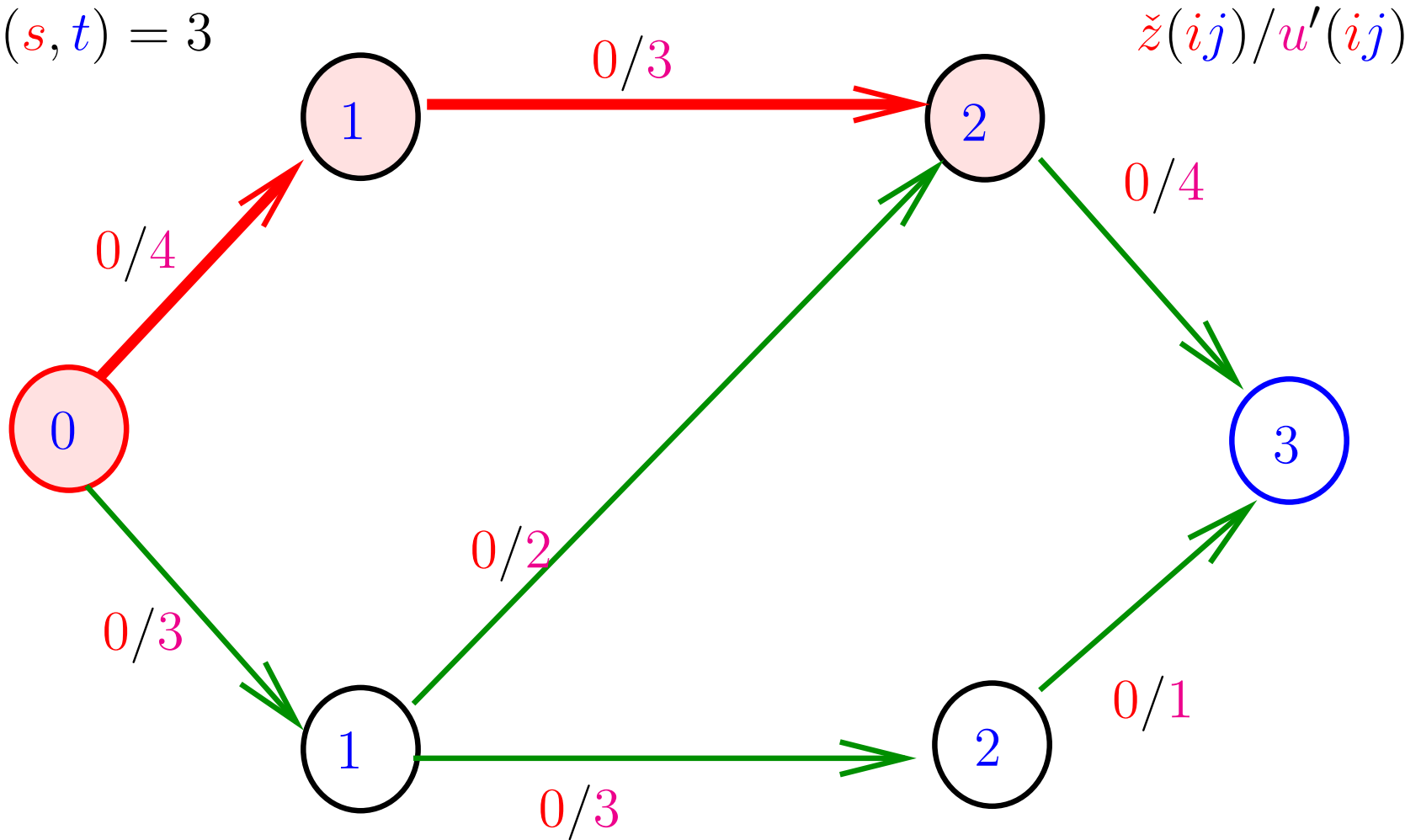
Passo bloqueador (1)

$\text{dist}(s, t) = 3$



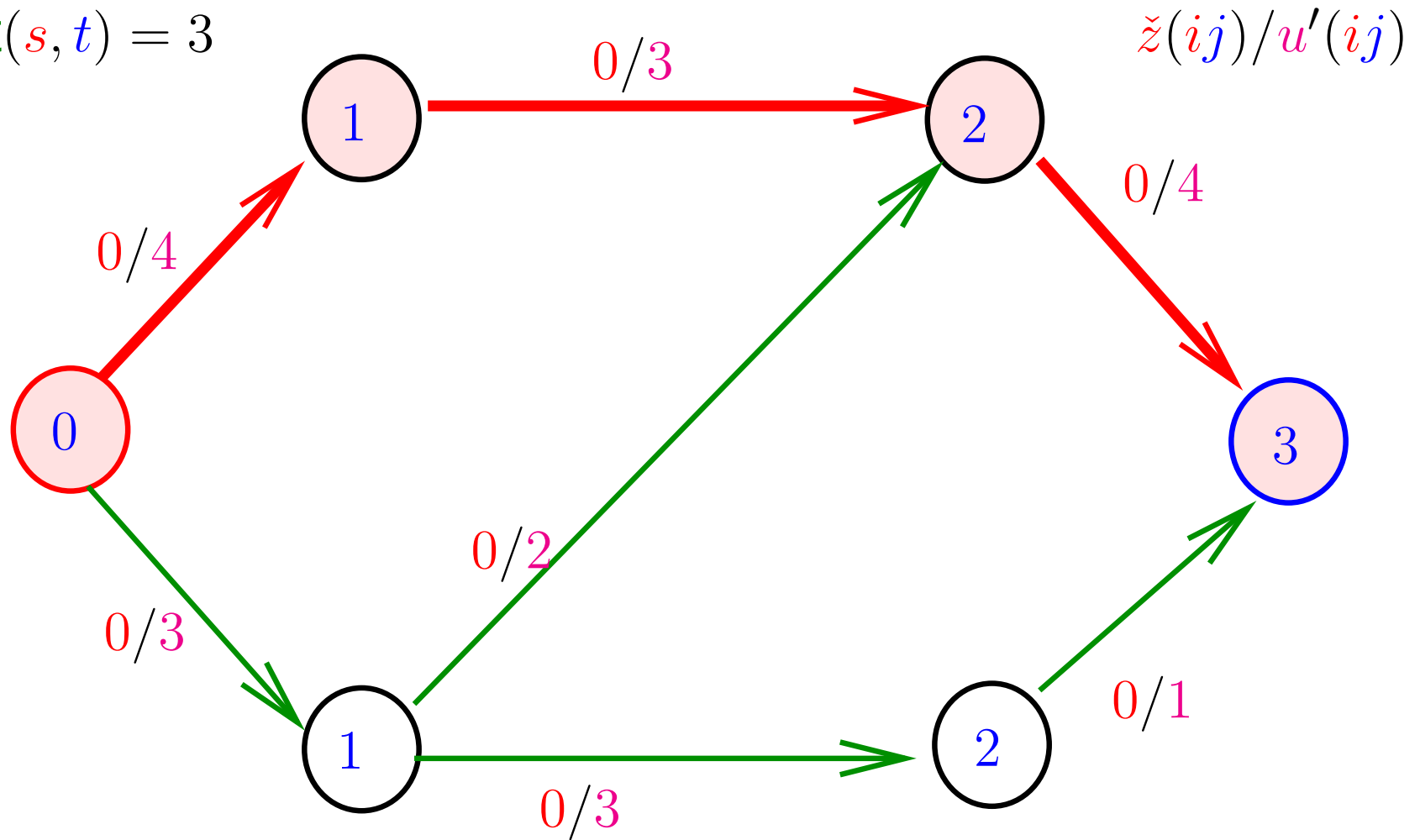
Passo bloqueador (2)

$\text{dist}(s, t) = 3$



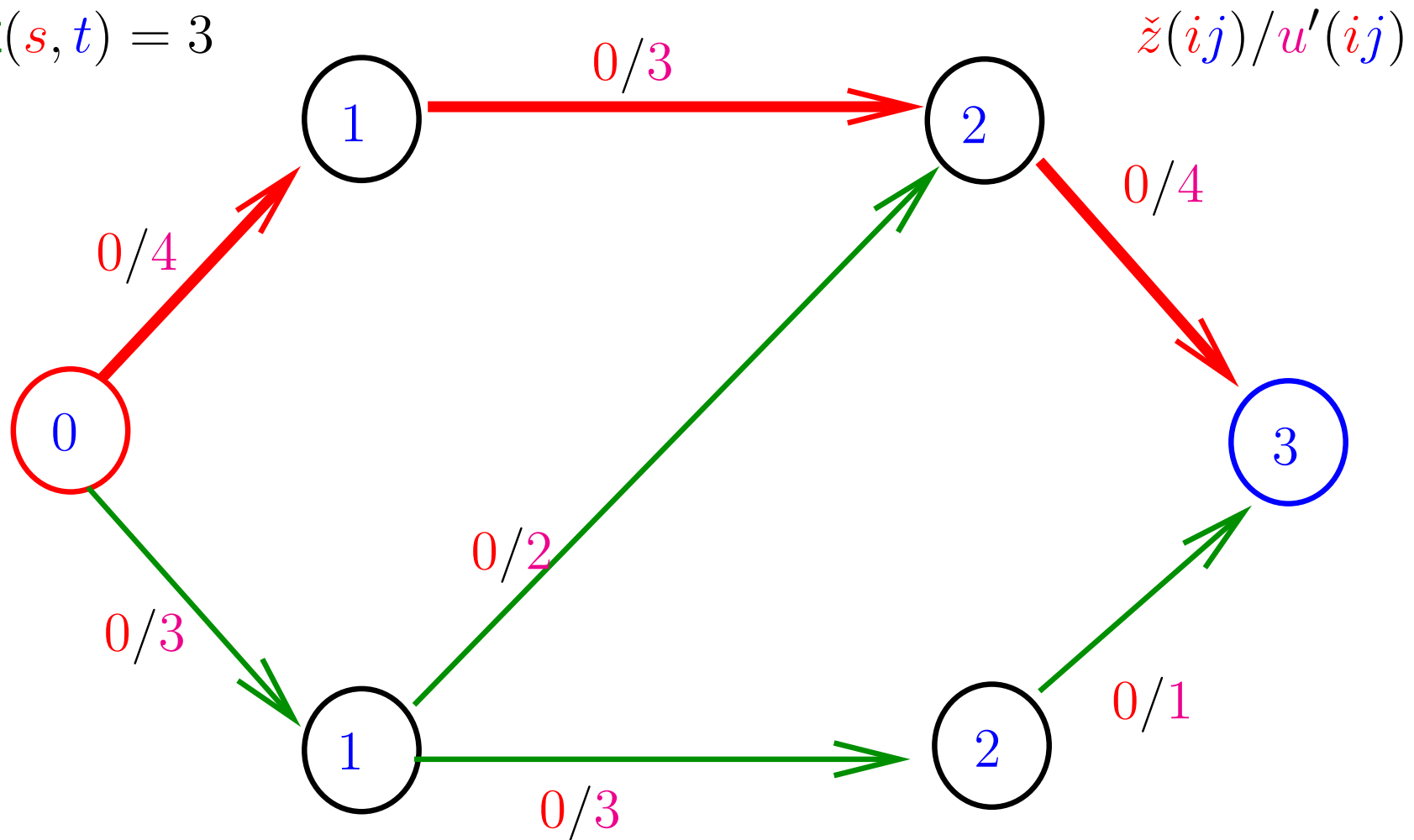
Passo bloqueador (3)

$\text{dist}(s, t) = 3$



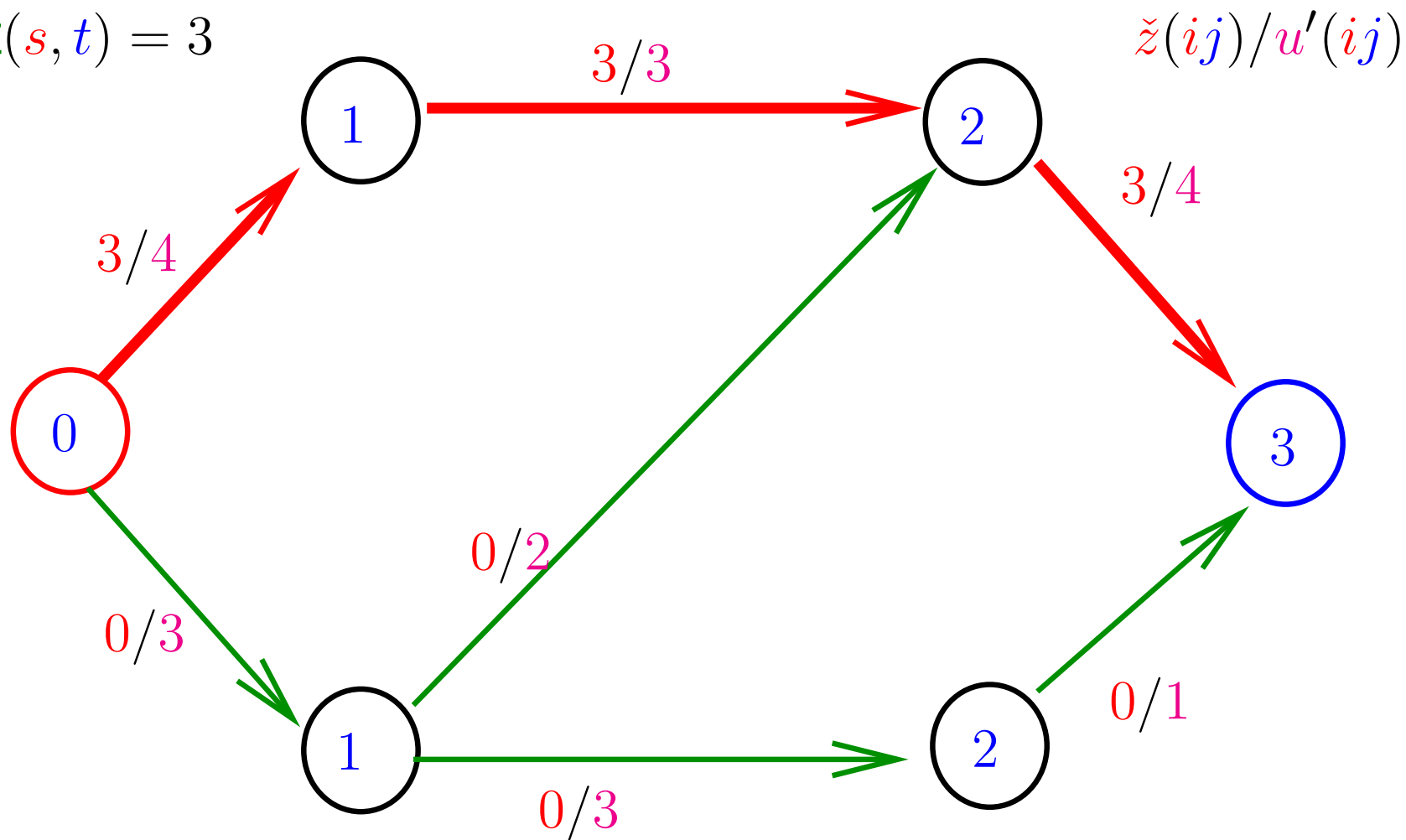
Caminho de incremento

$\text{dist}(s, t) = 3$



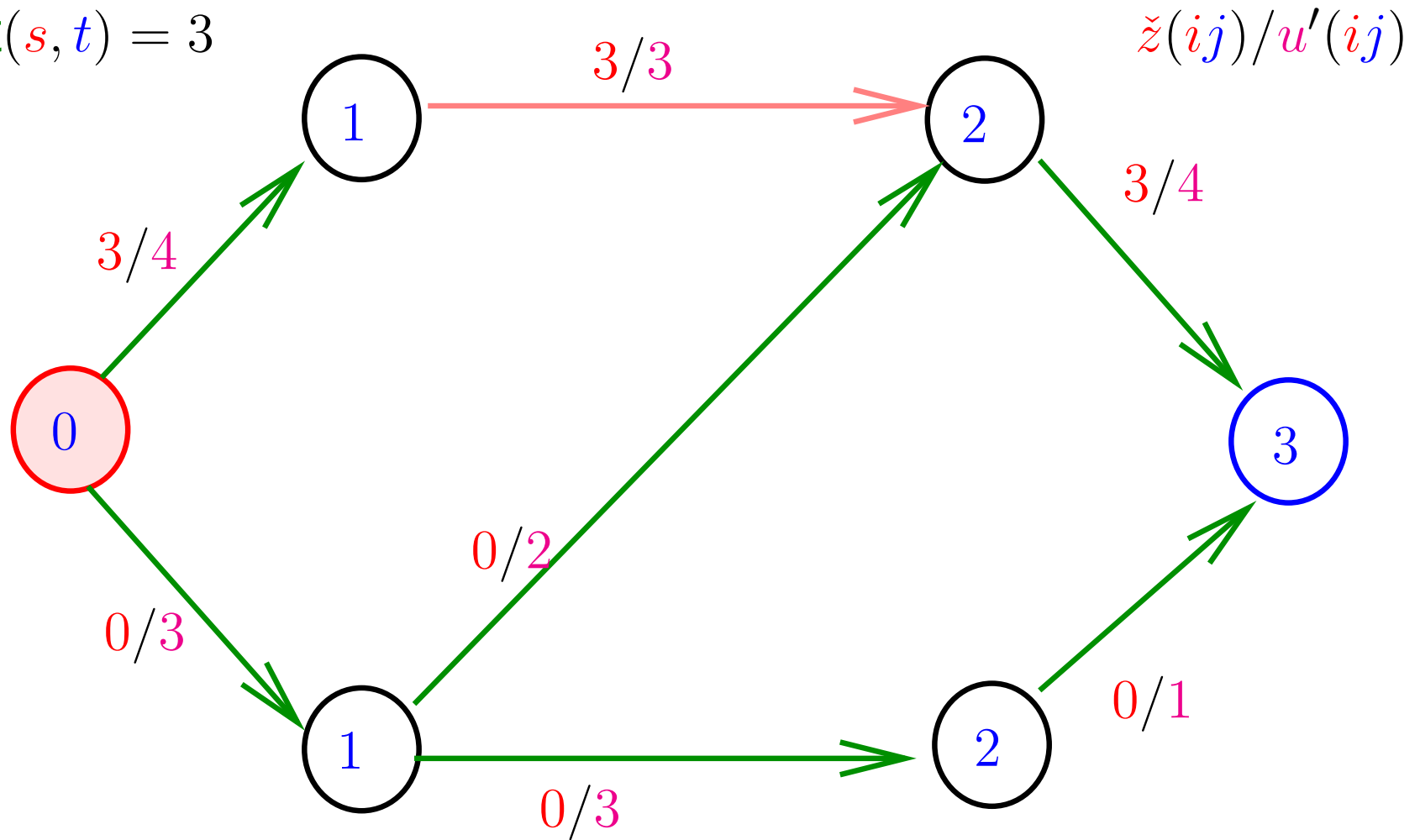
Incremente fluxo

$\text{dist}(s, t) = 3$



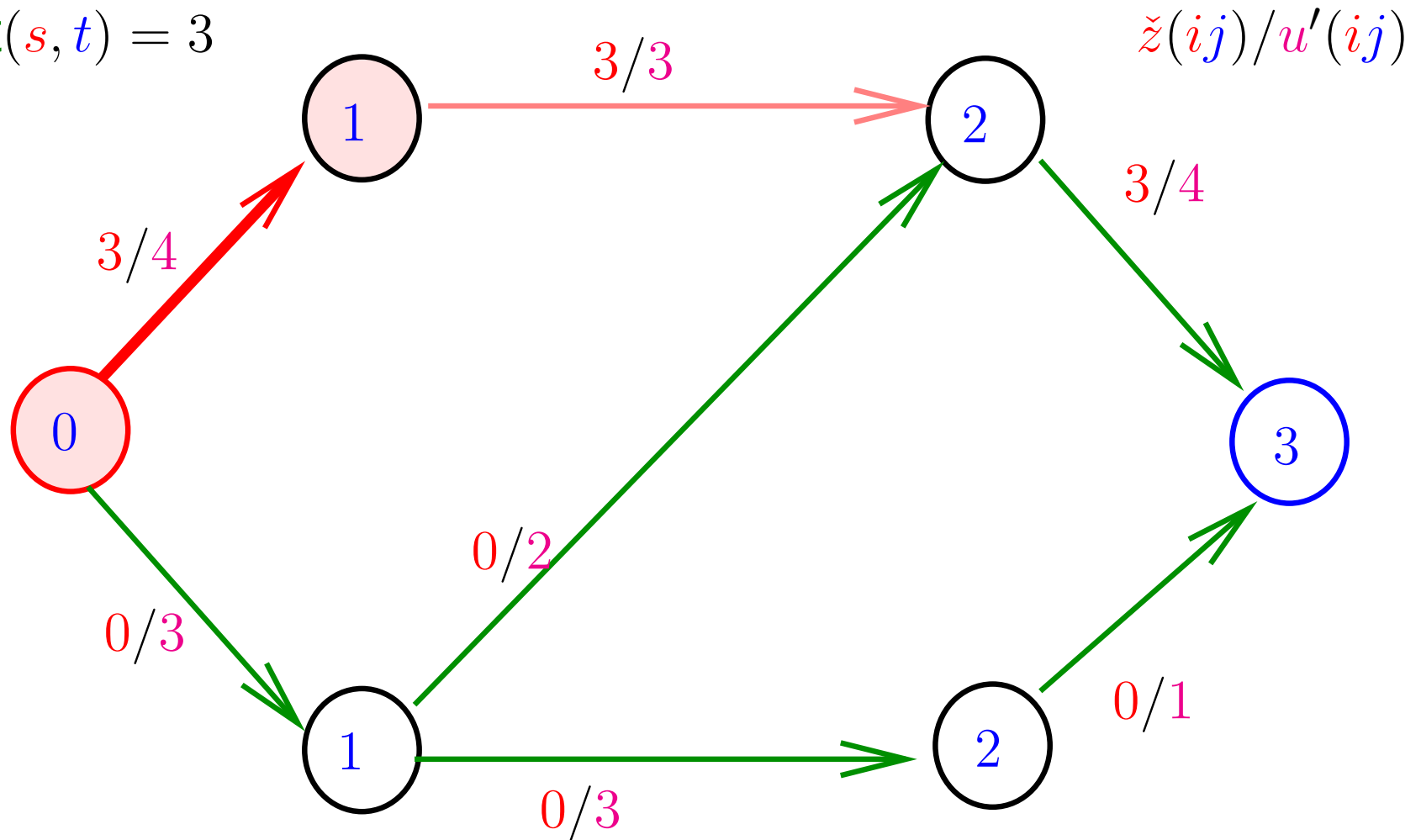
Passo bloqueador (4)

$\text{dist}(s, t) = 3$



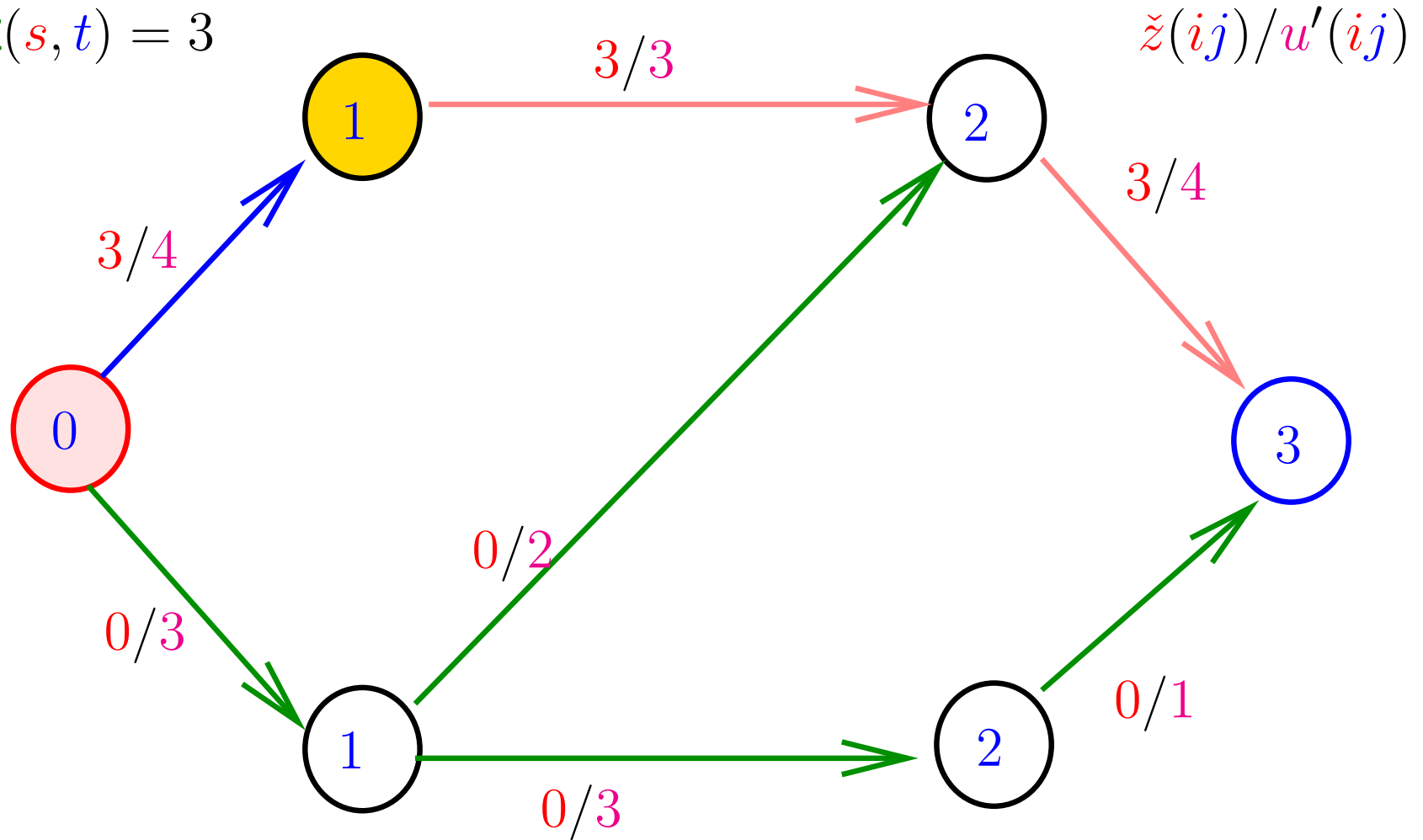
Passo bloqueador (5)

$\text{dist}(s, t) = 3$



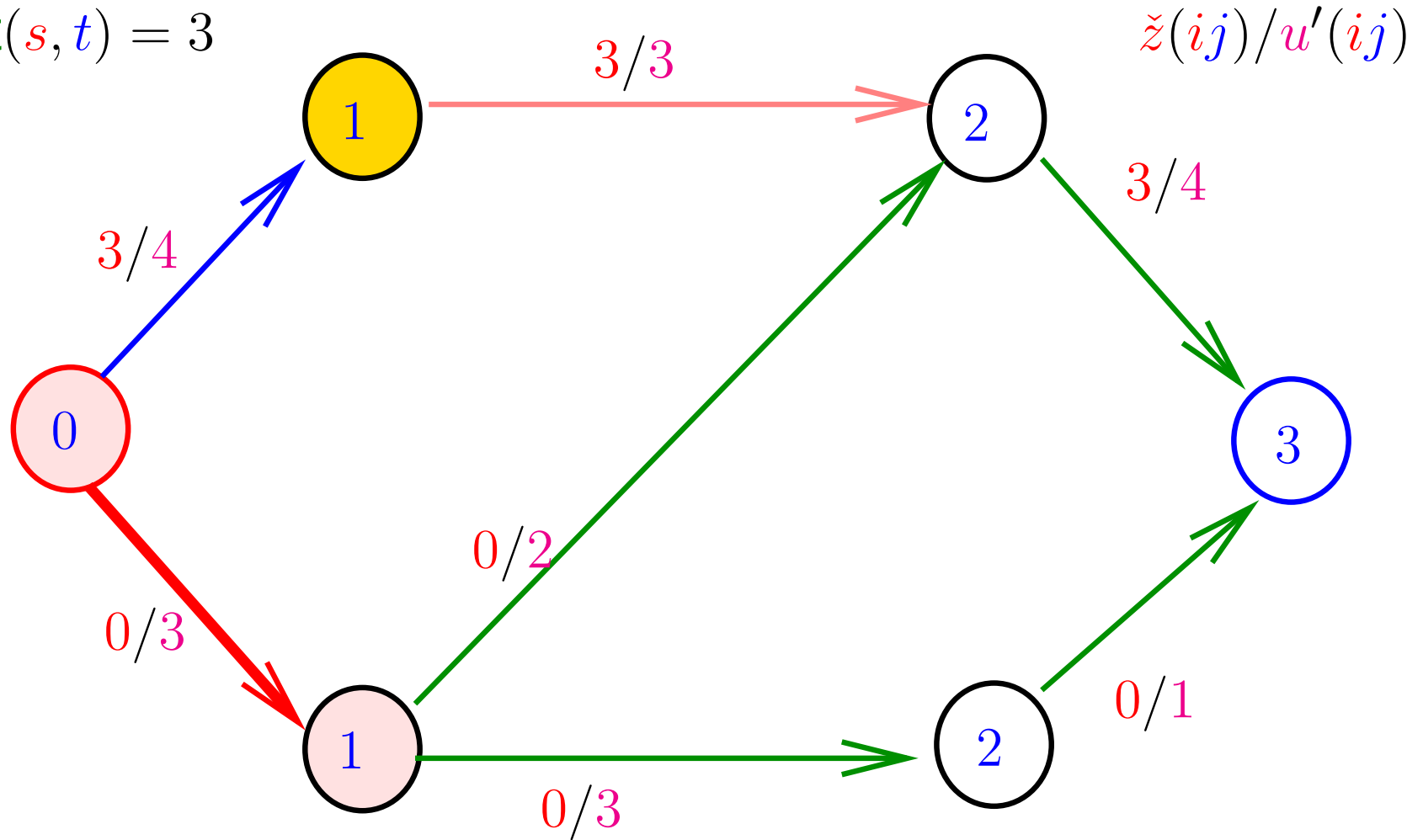
Passo bloqueador (6)

$\text{dist}(s, t) = 3$



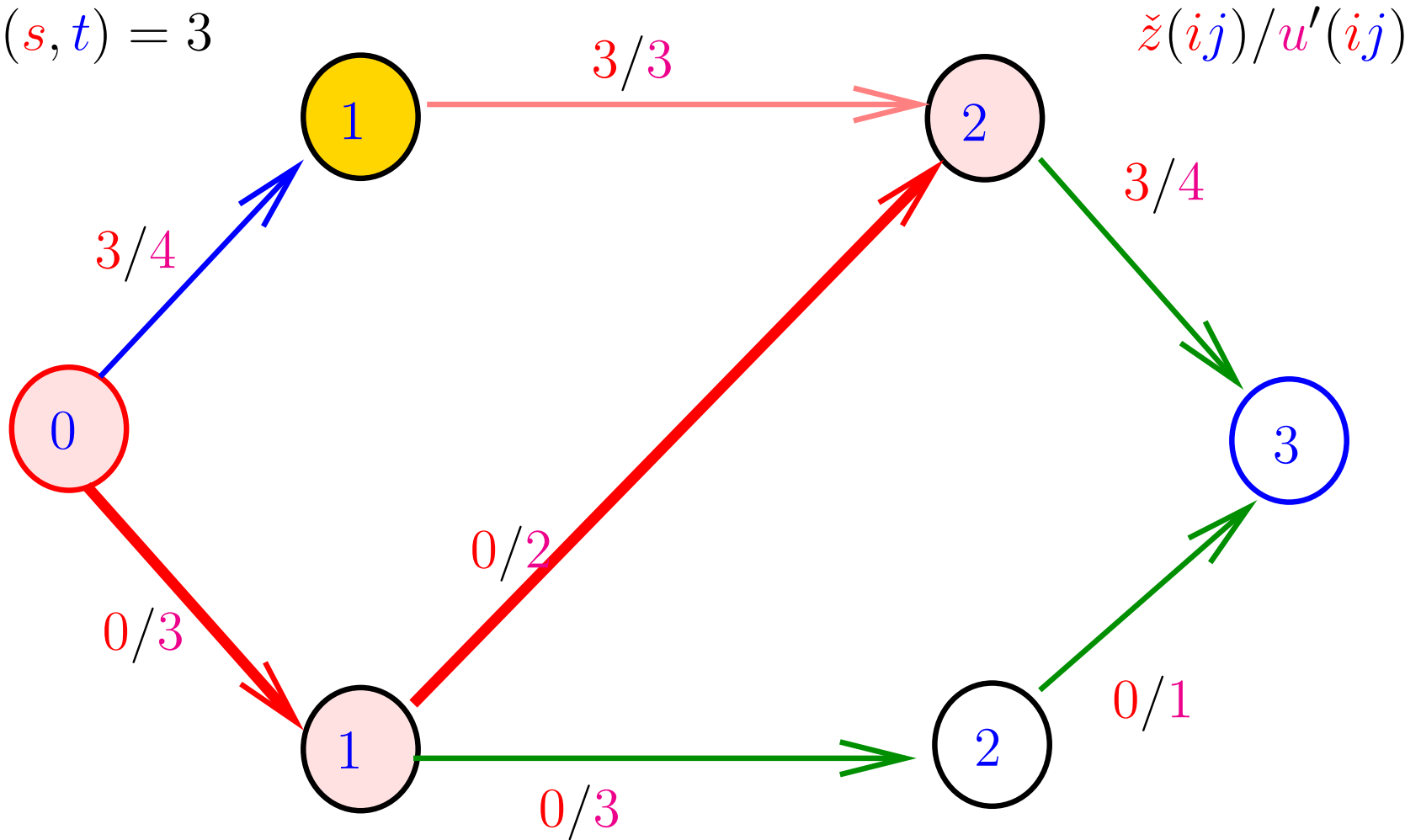
Passo bloqueador (7)

$\text{dist}(s, t) = 3$



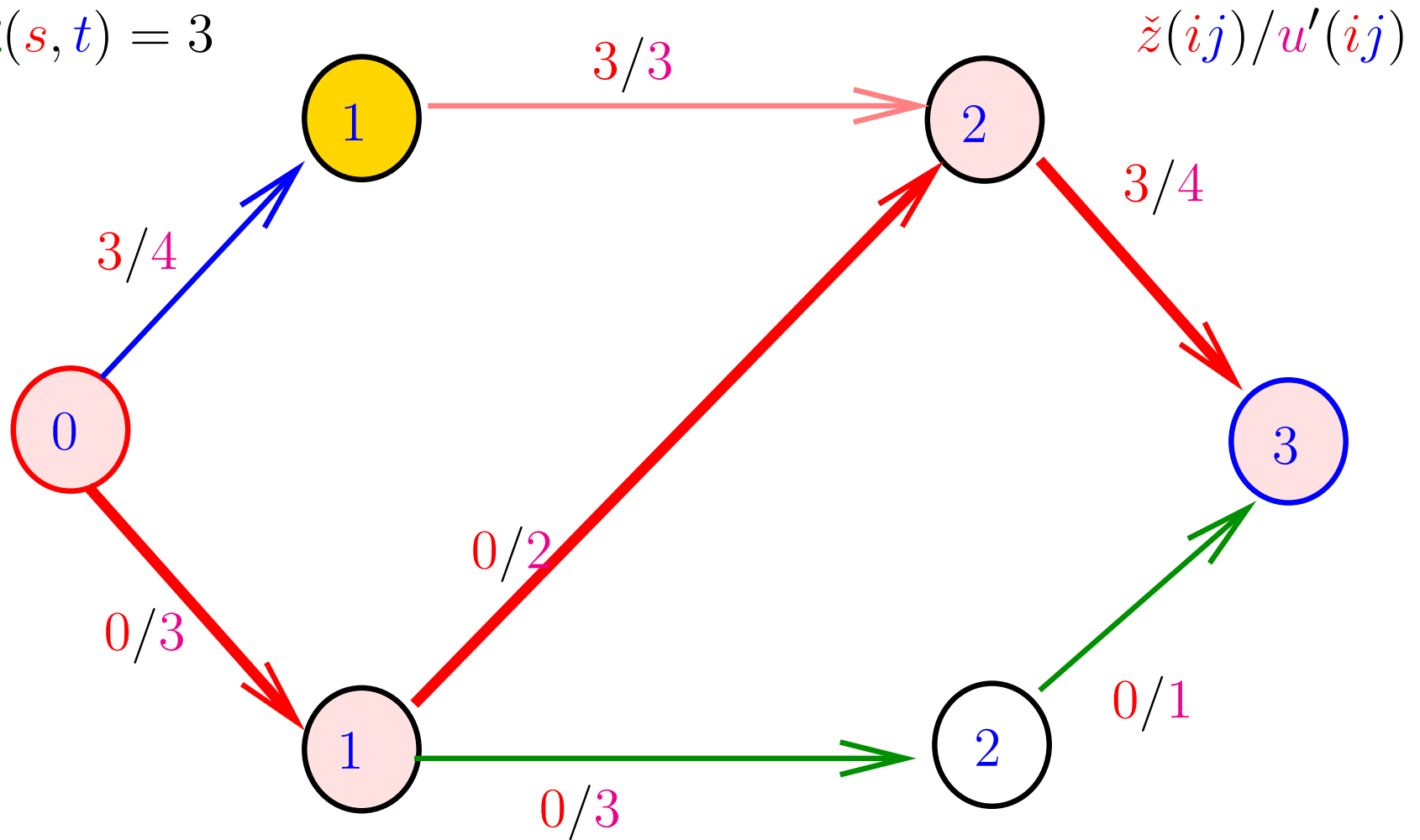
Passo bloqueador (8)

$\text{dist}(s, t) = 3$



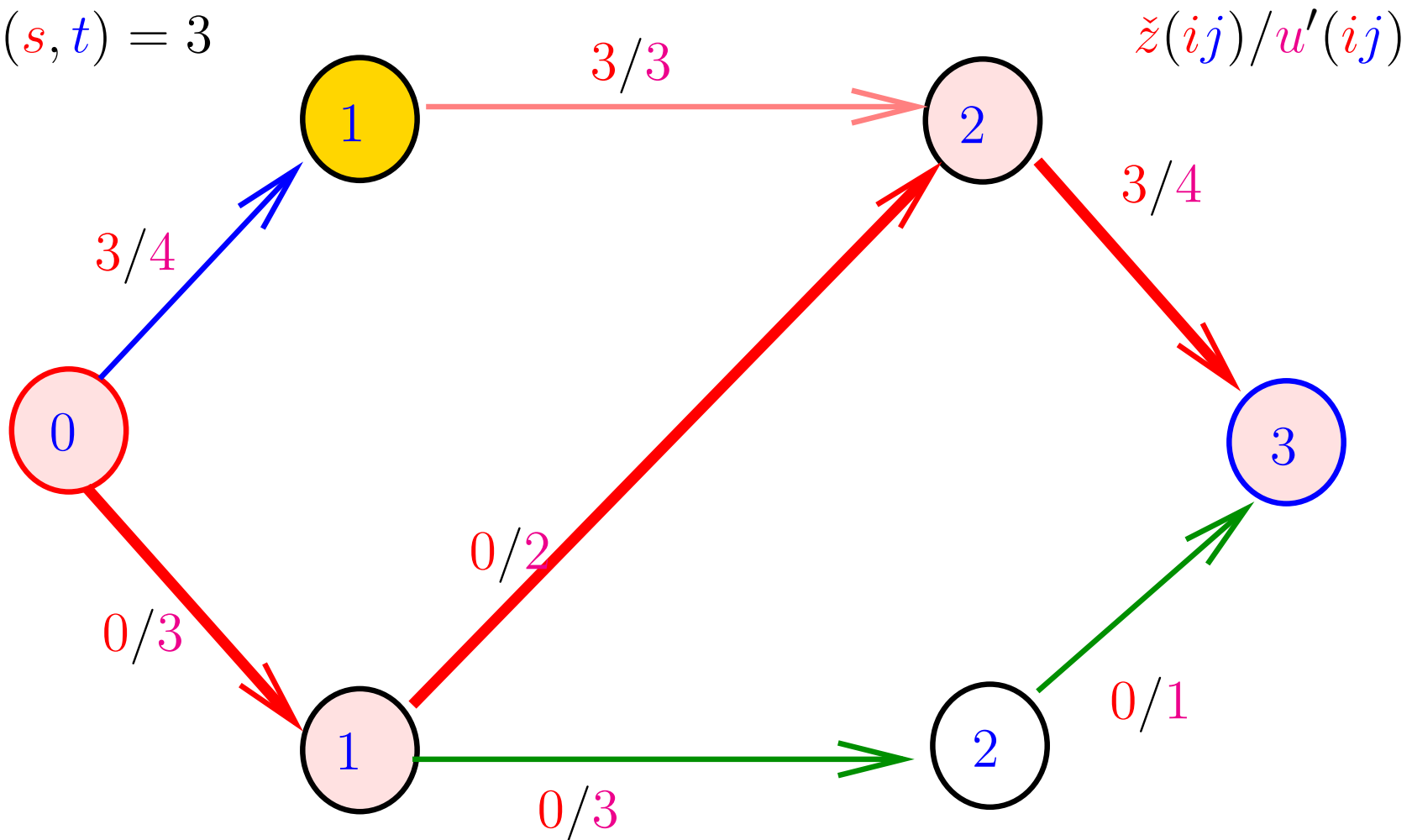
Passo bloqueador (9)

$\text{dist}(s, t) = 3$



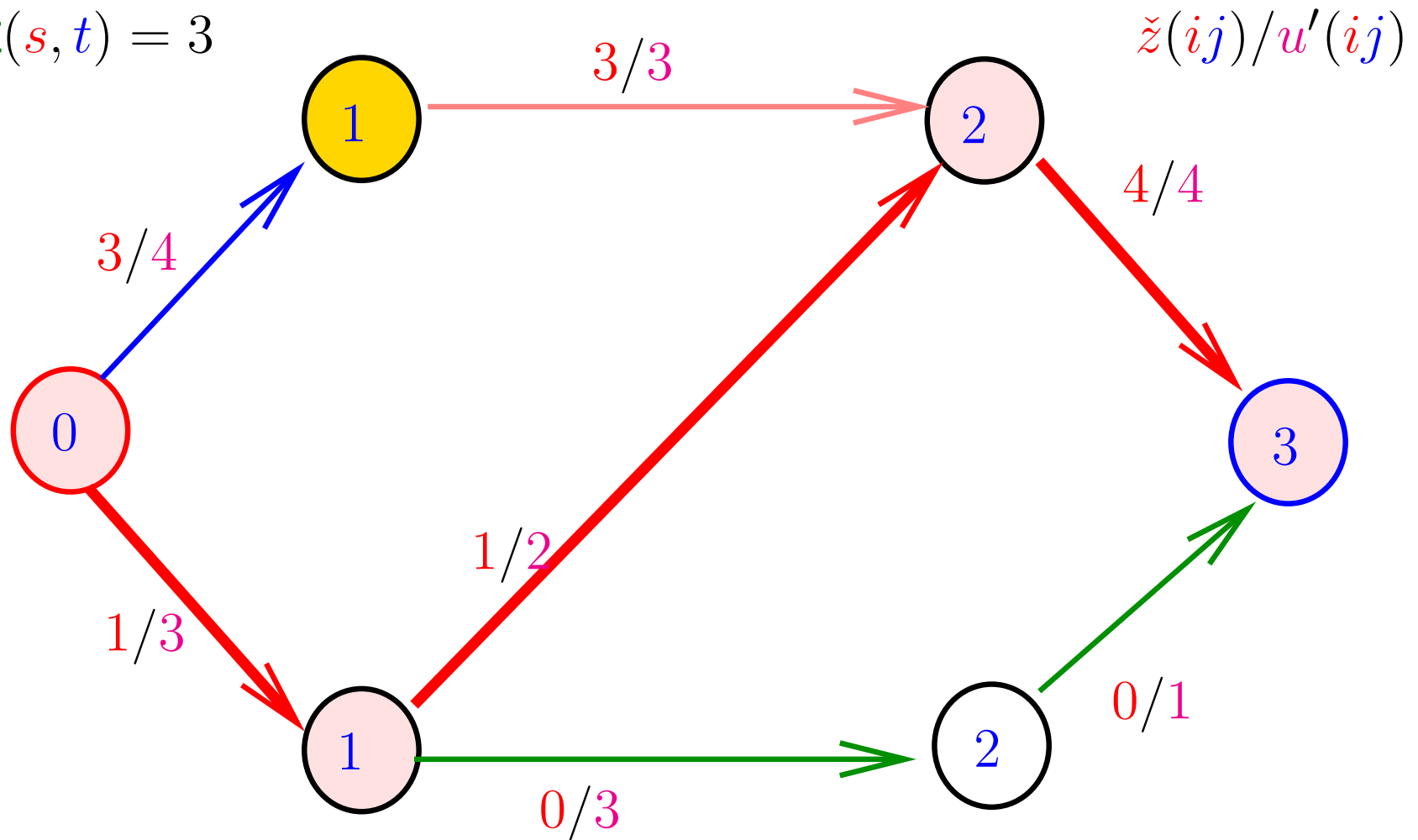
Caminho de incremento

$$\text{dist}(s, t) = 3$$



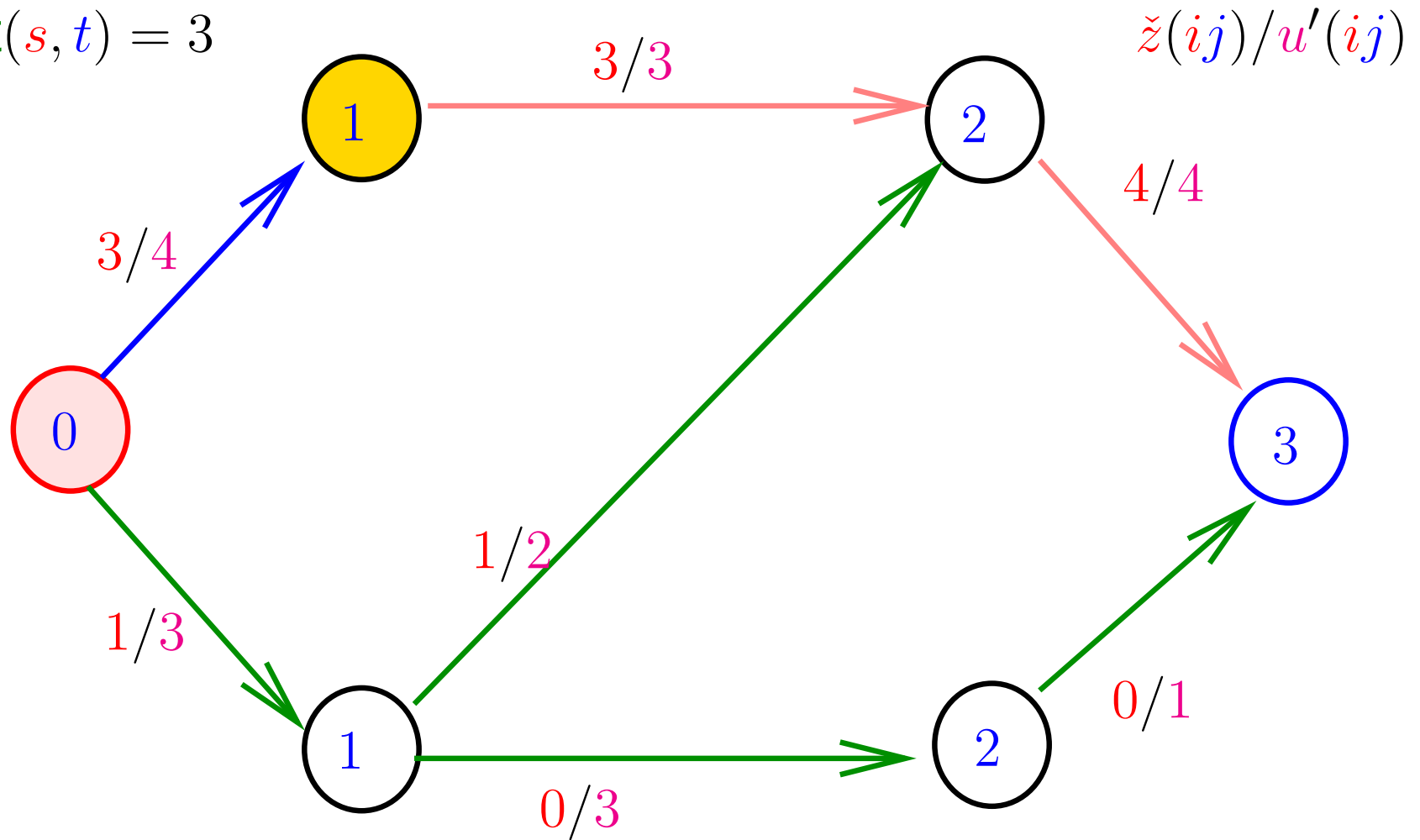
Incremente fluxo

$\text{dist}(s, t) = 3$



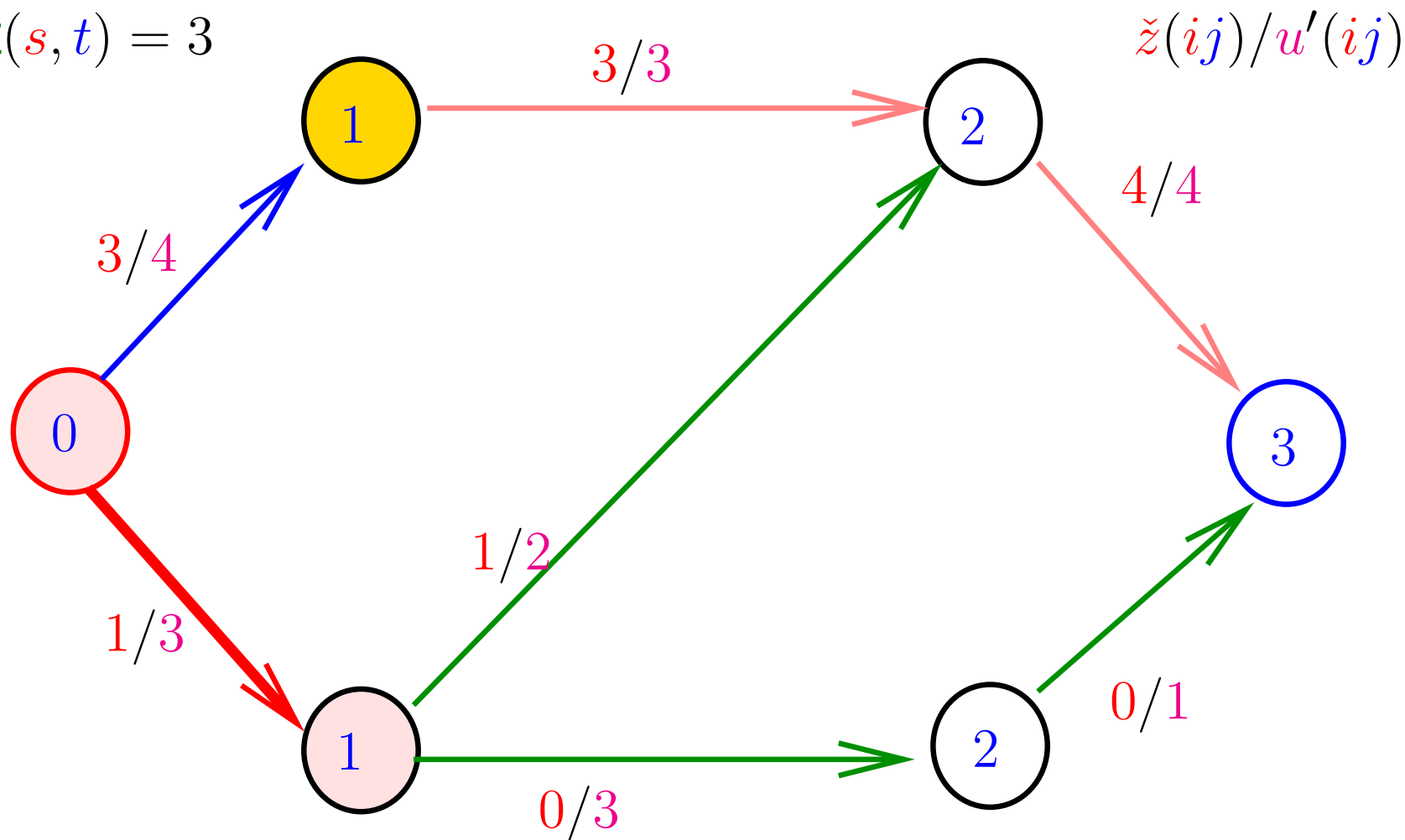
Passo bloqueador (10)

$\text{dist}(s, t) = 3$



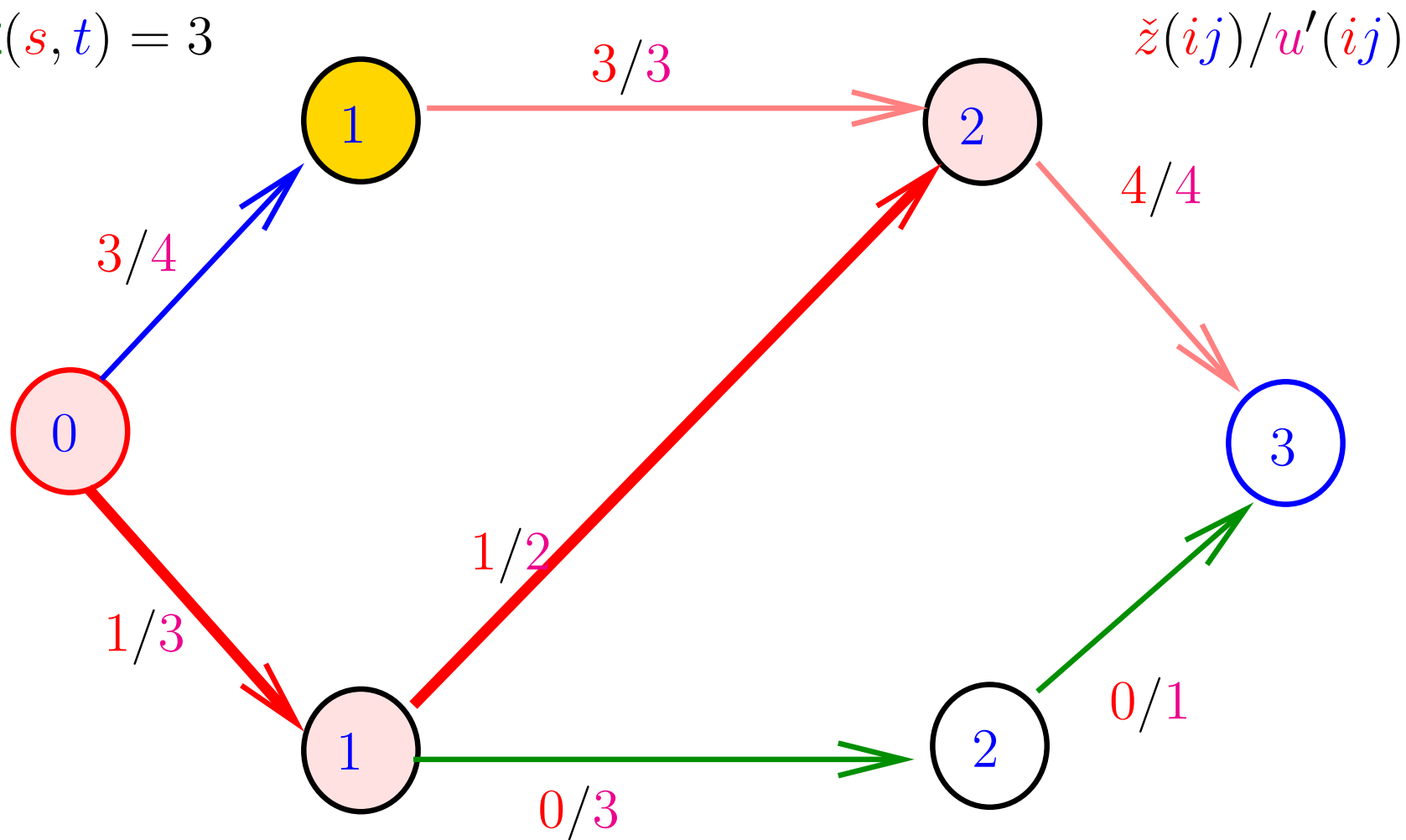
Passo bloqueador (11)

$\text{dist}(s, t) = 3$



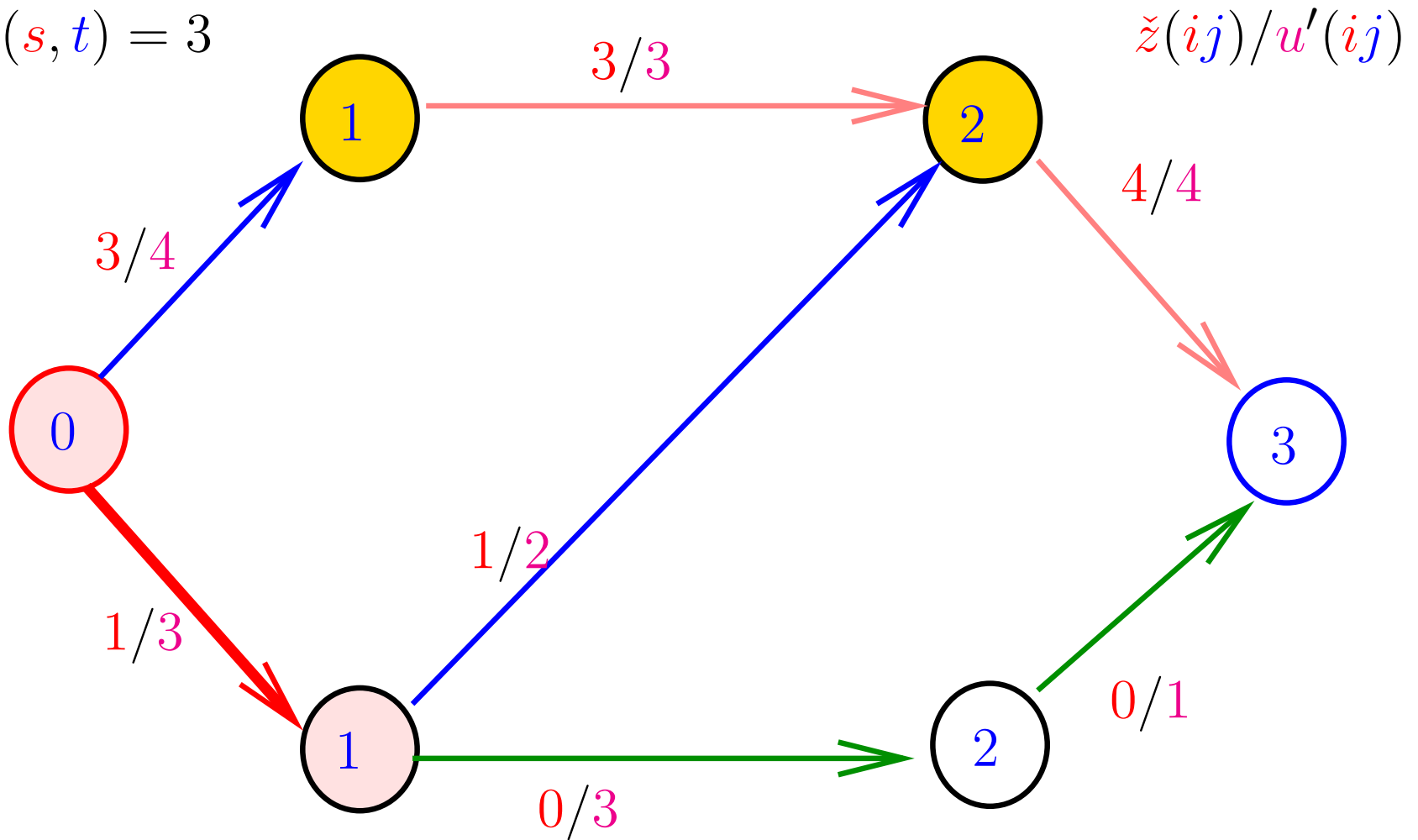
Passo bloqueador (12)

$\text{dist}(s, t) = 3$



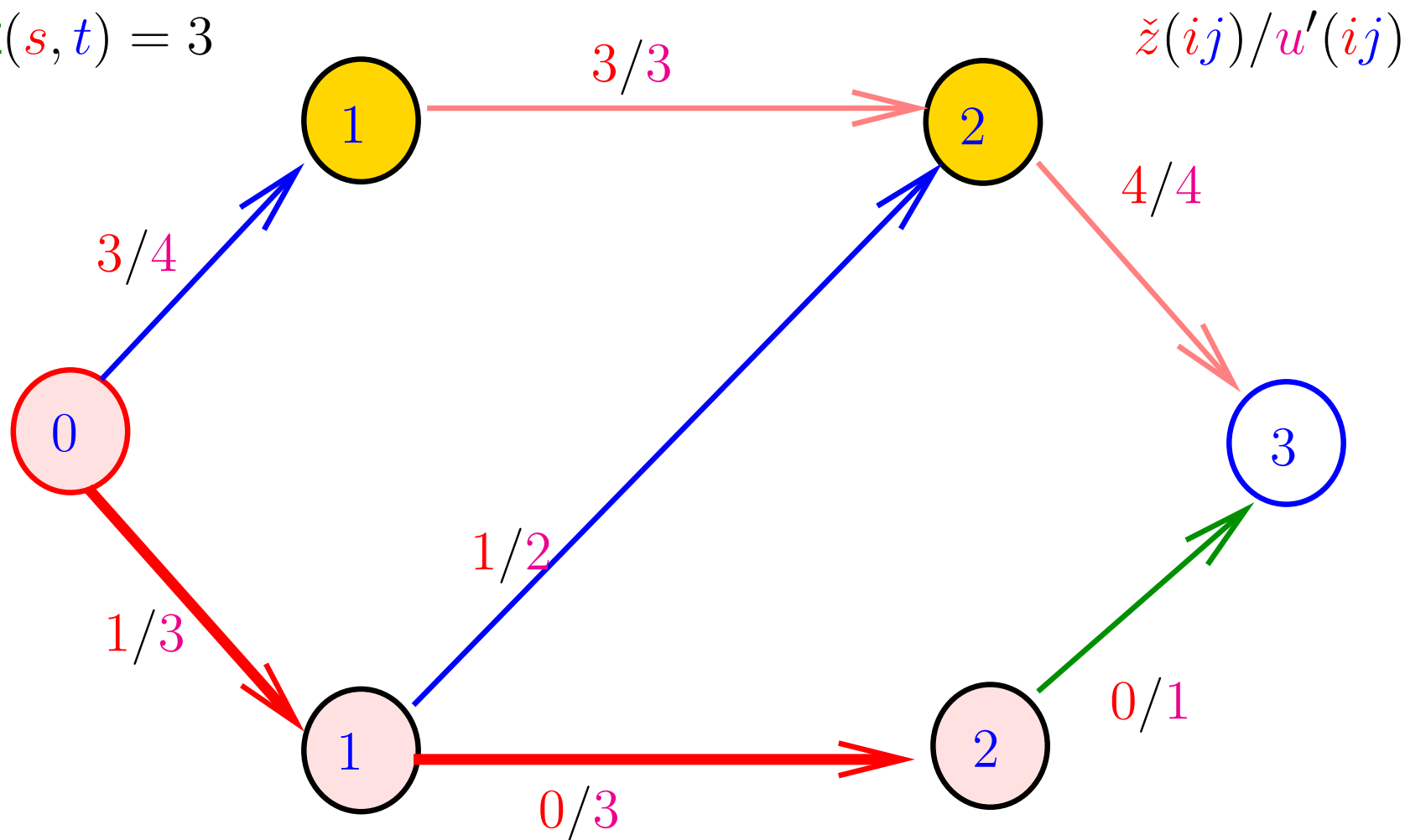
Passo bloqueador (13)

$\text{dist}(s, t) = 3$



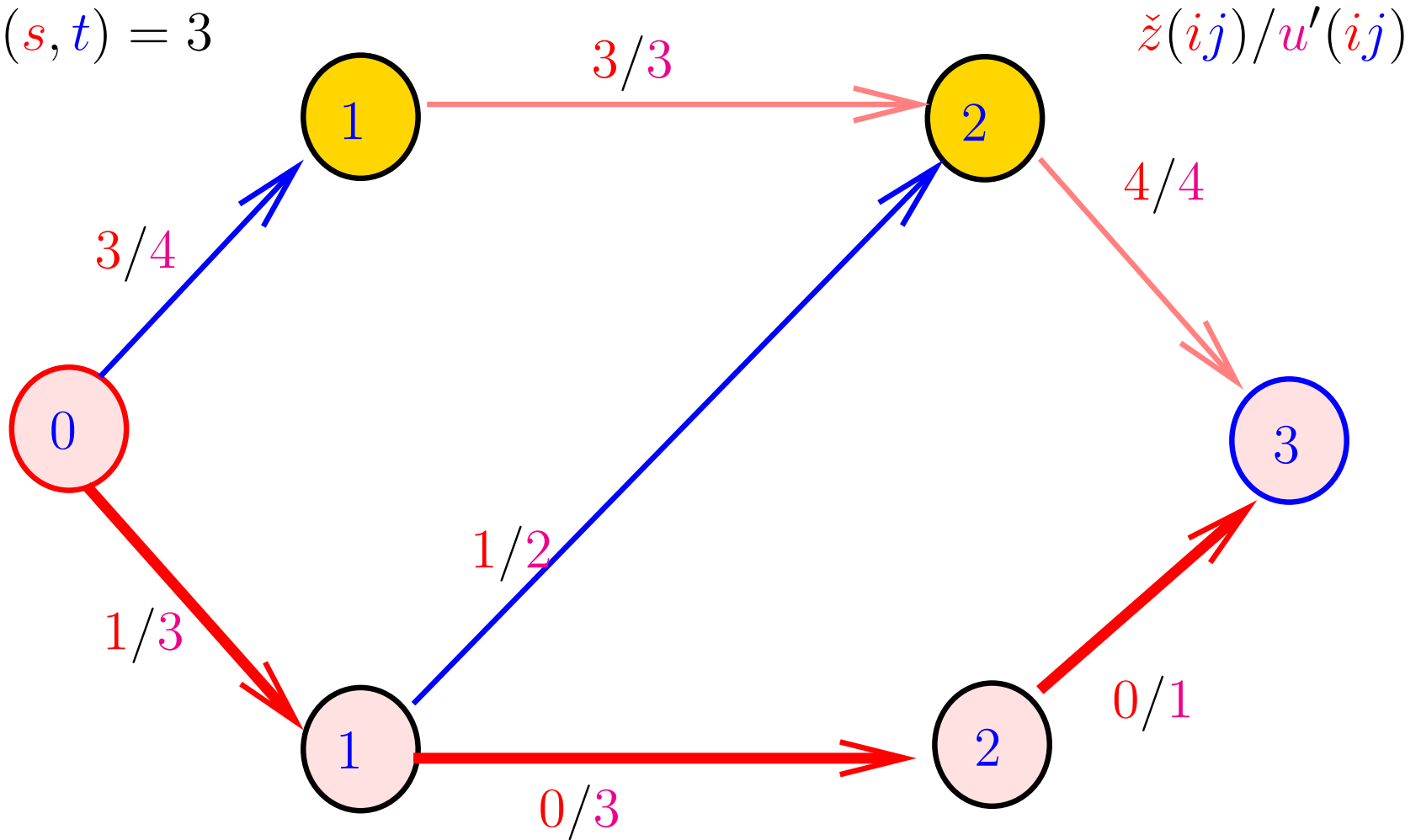
Passo bloqueador (14)

$\text{dist}(s, t) = 3$



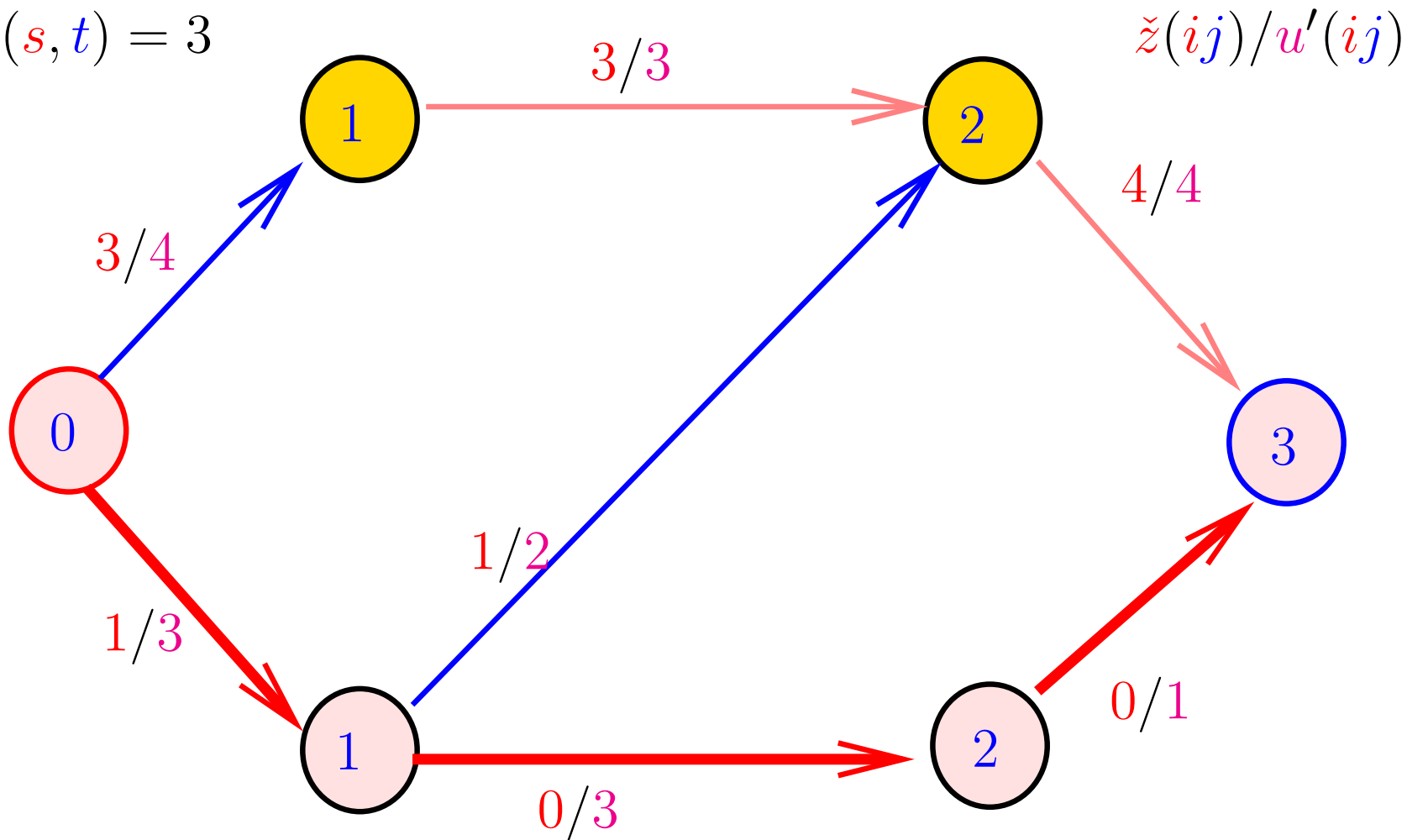
Passo bloqueador (15)

$\text{dist}(s, t) = 3$



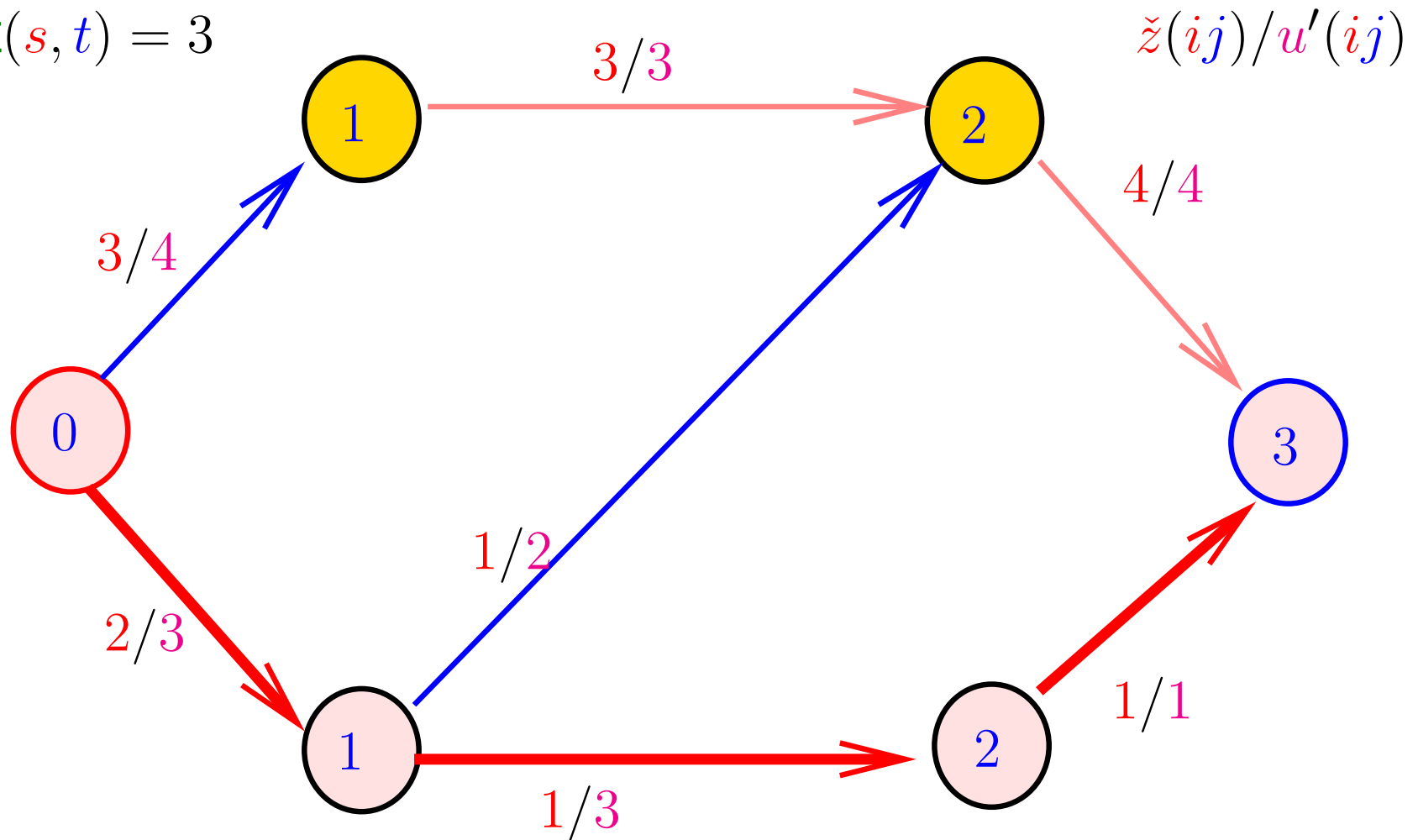
Caminho de incremento

$$\text{dist}(s, t) = 3$$



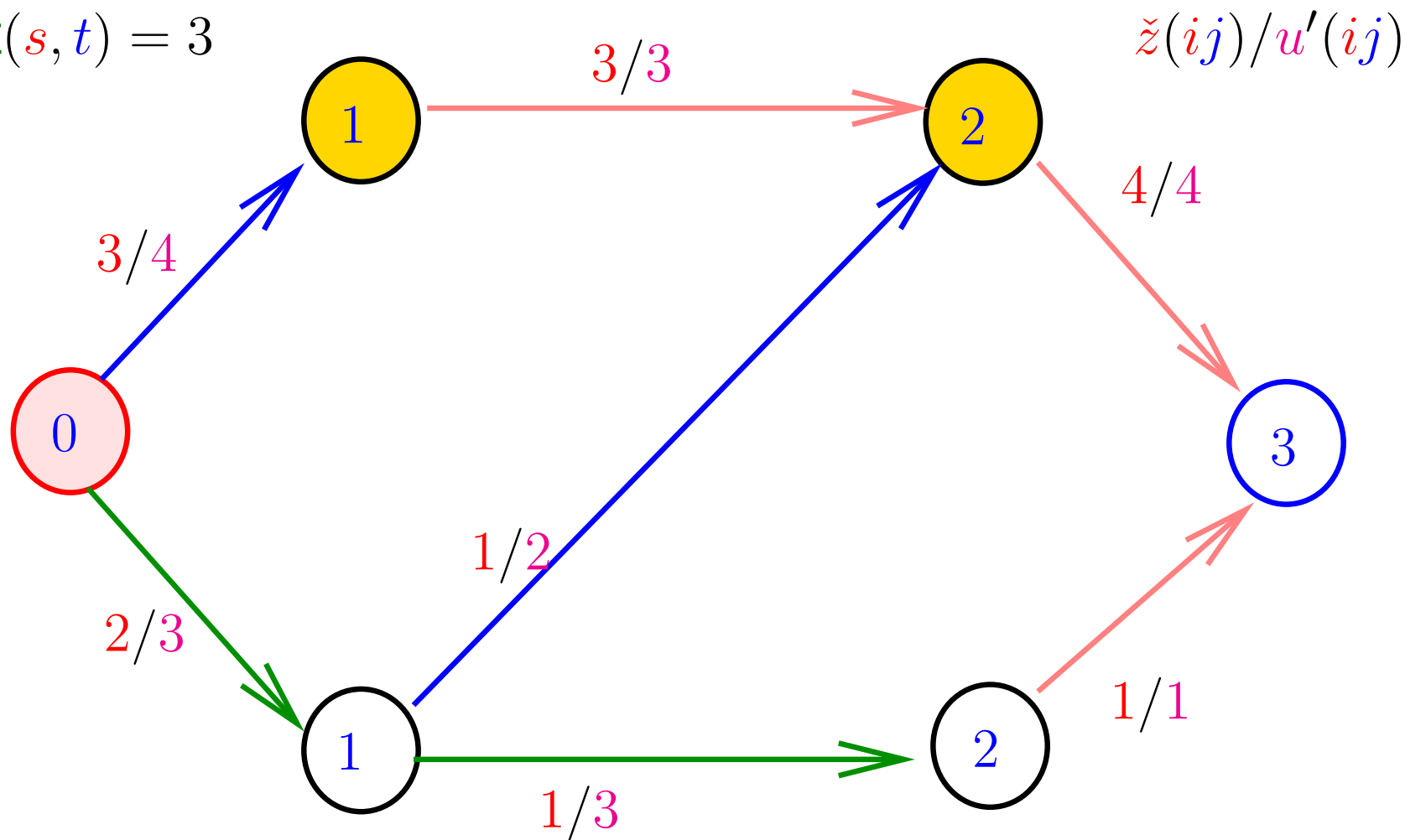
Incremente fluxo

$\text{dist}(s, t) = 3$



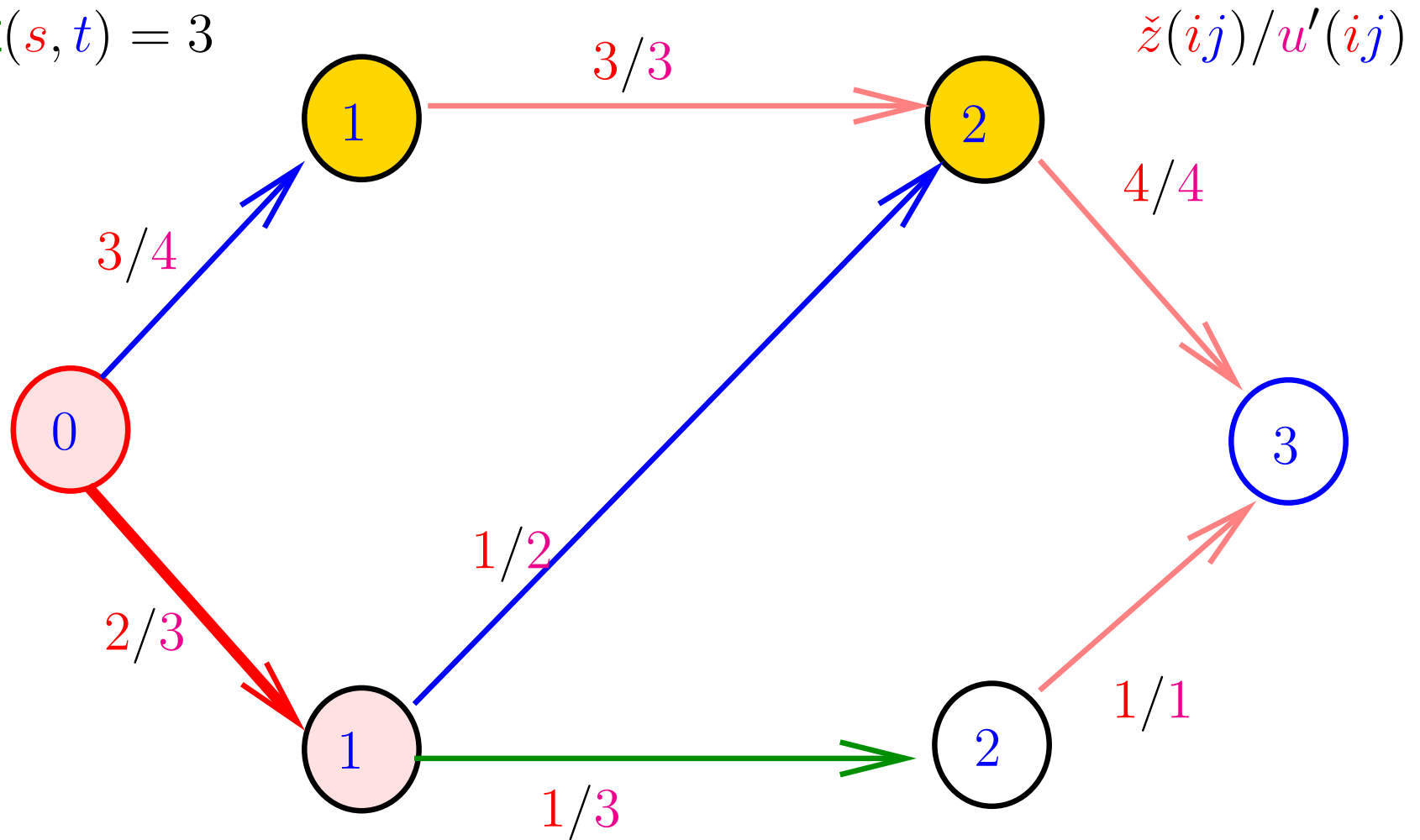
Passo bloqueador (16)

$\text{dist}(s, t) = 3$



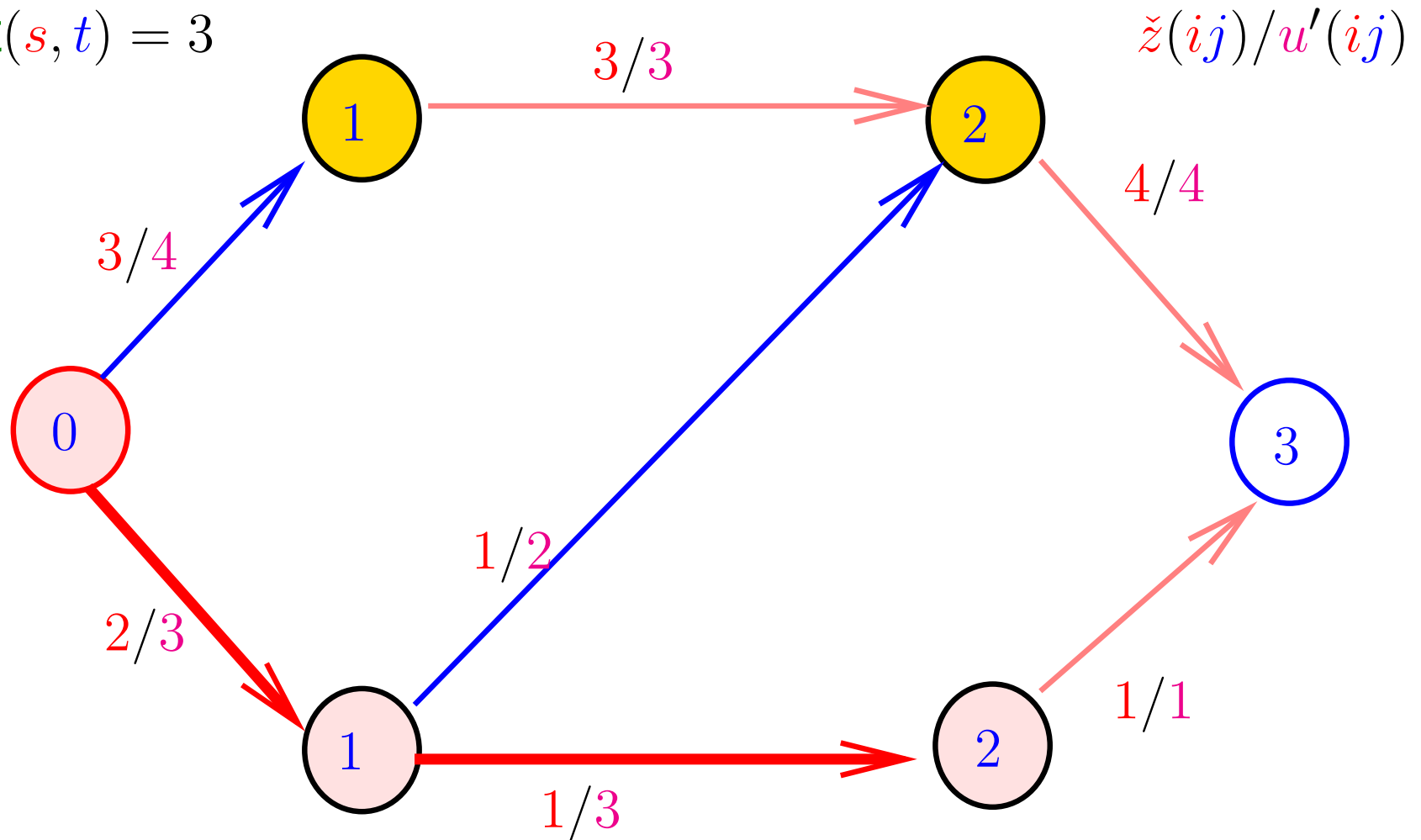
Passo bloqueador (17)

$\text{dist}(s, t) = 3$



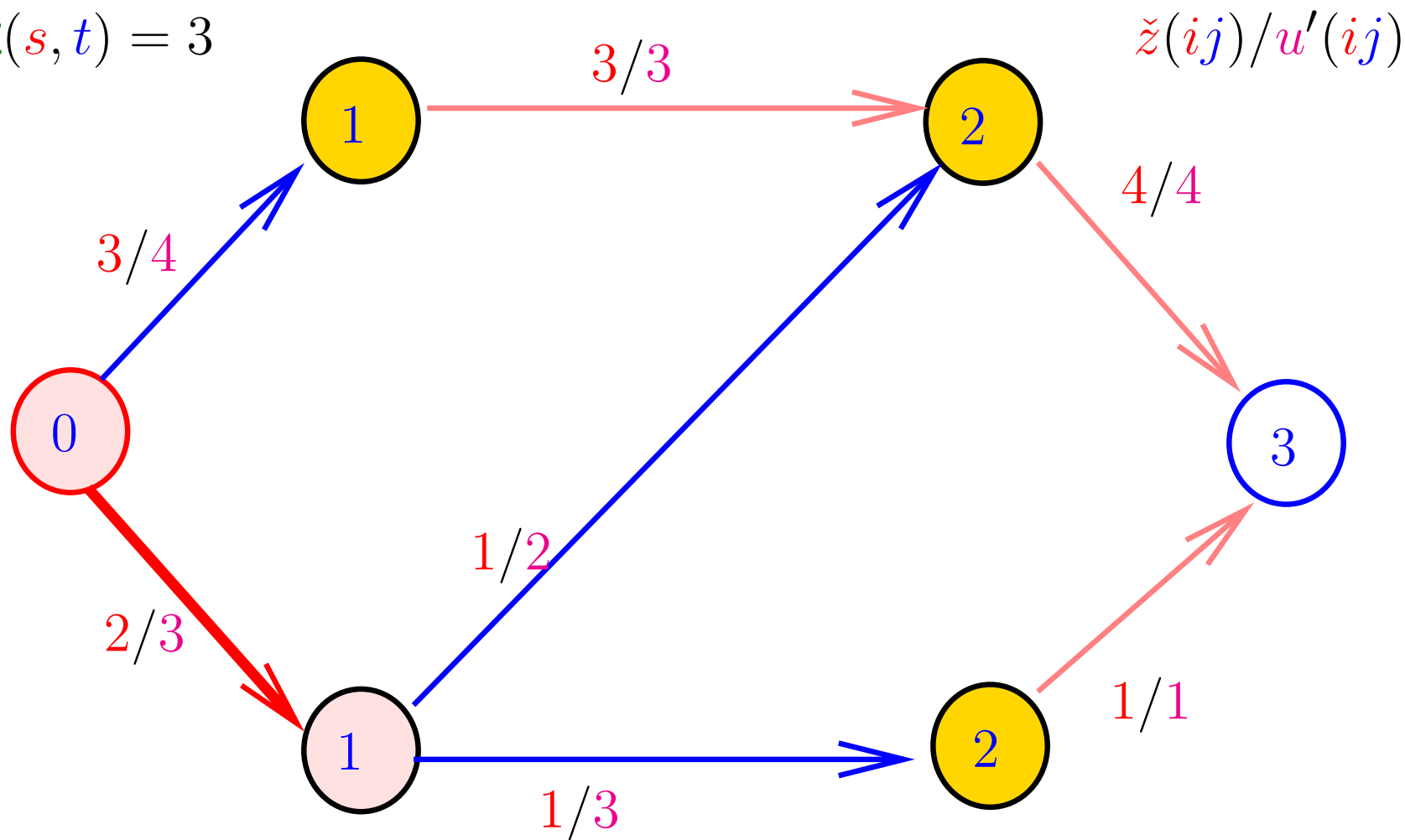
Passo bloqueador (18)

$\text{dist}(s, t) = 3$



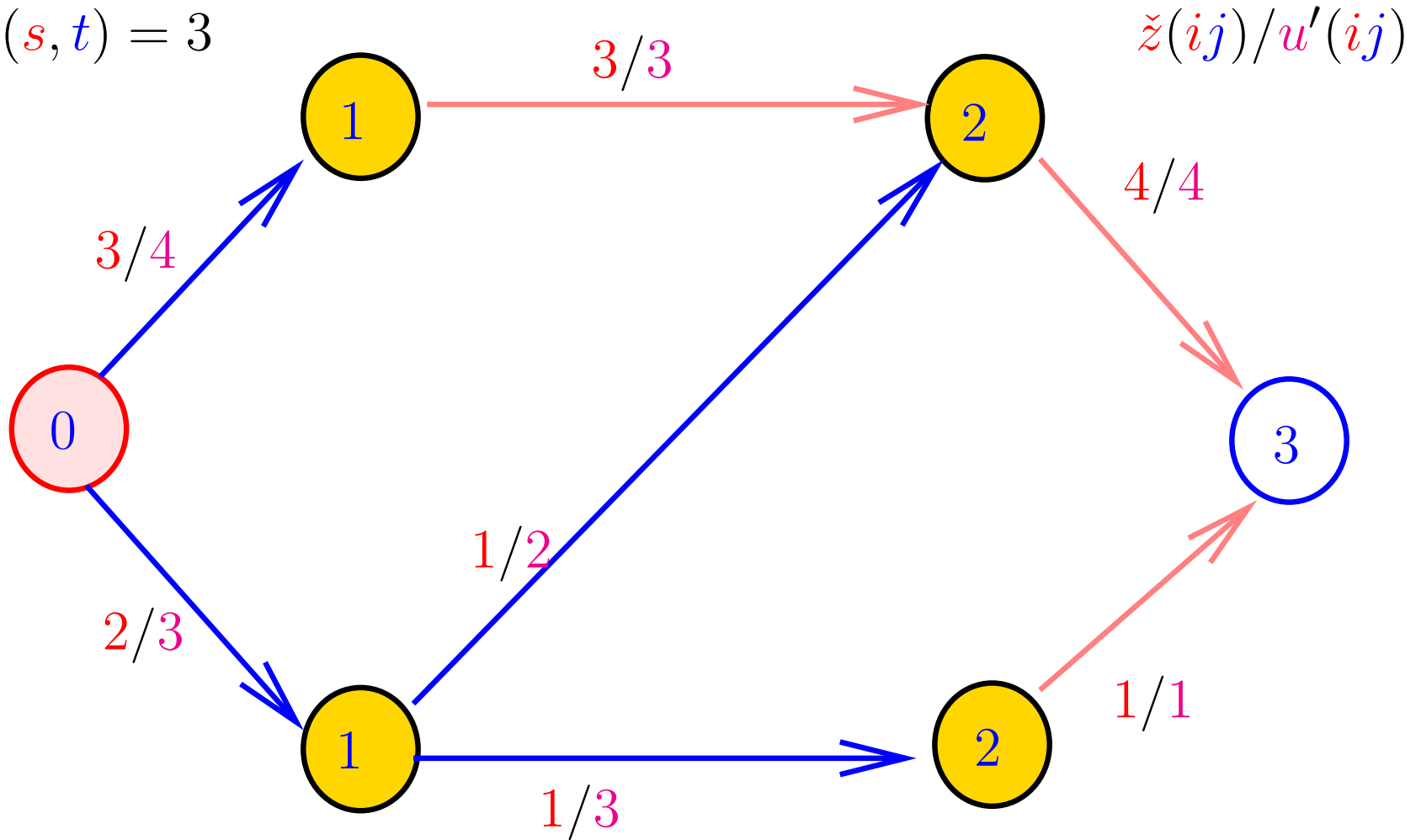
Passo bloqueador (19)

$\text{dist}(s, t) = 3$



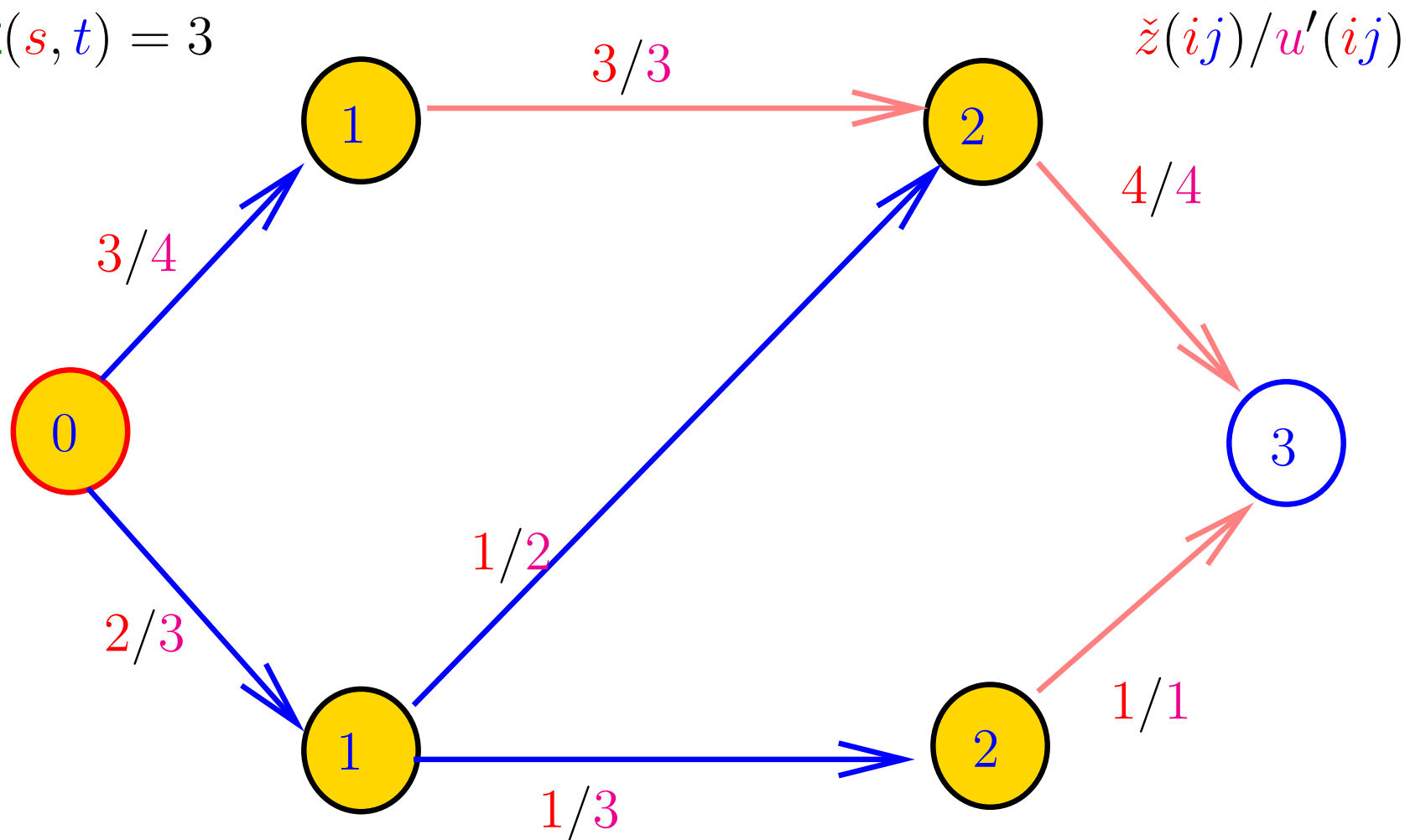
Passo bloqueador (20)

$$\text{dist}(s, t) = 3$$



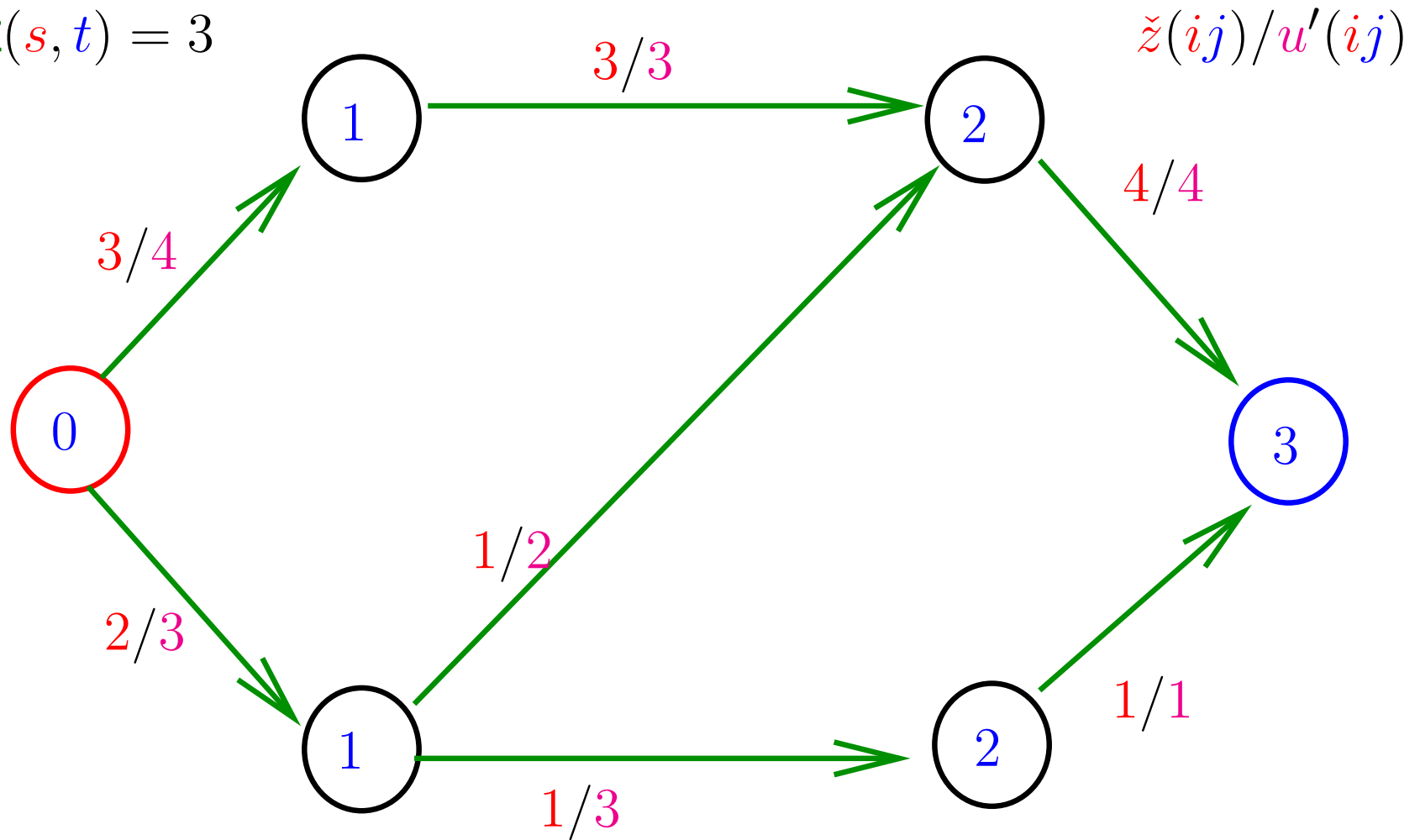
Passo bloqueador (21)

$\text{dist}(s, t) = 3$



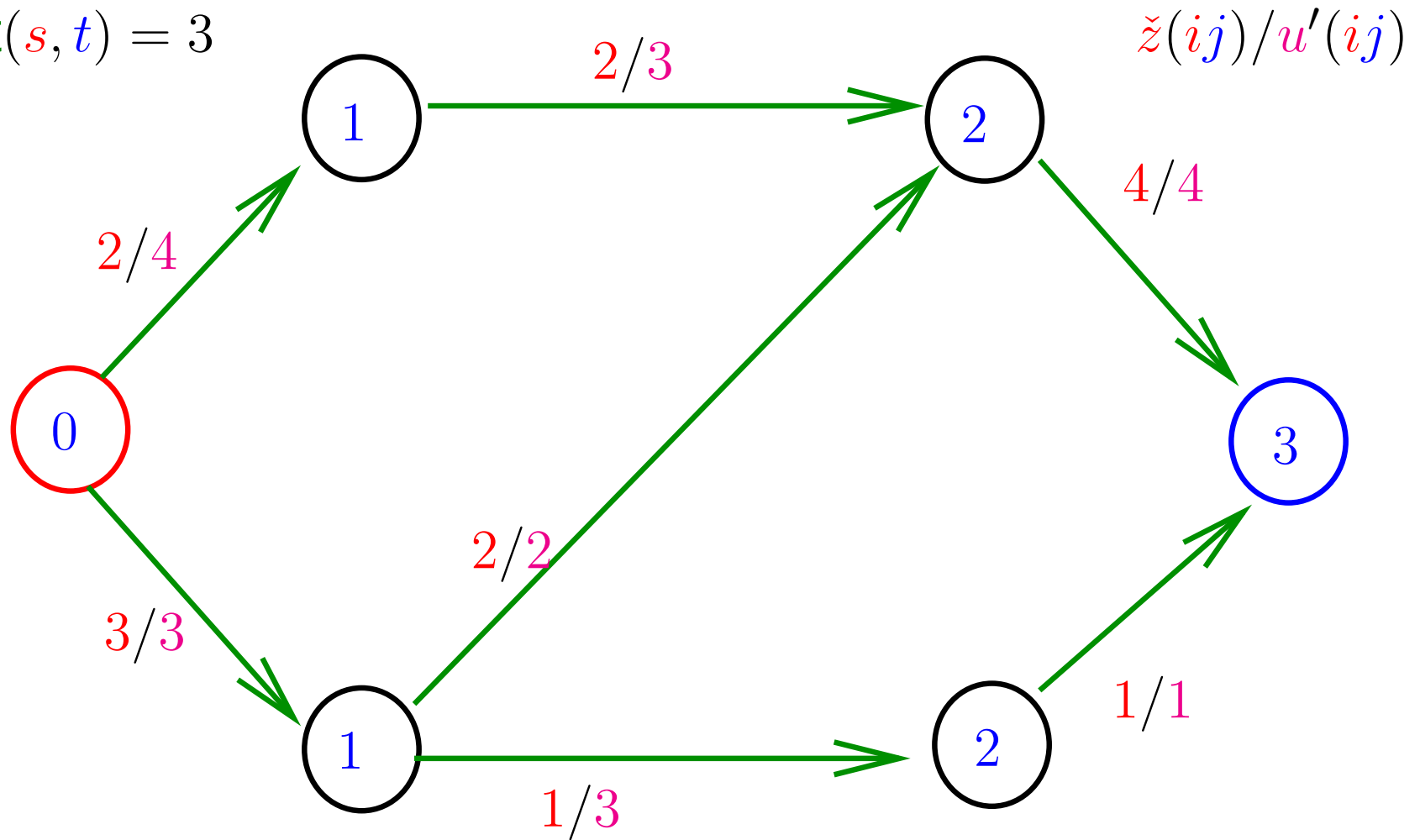
Fluxo bloqueador

$\text{dist}(s, t) = 3$



Veja outro fluxo bloqueador

$\text{dist}(s, t) = 3$



Passo bloqueador

PASSO-BLOQUEADOR ($N, A_{\check{x}}, s, t, \check{x}, y$)

- 1 $B \leftarrow \emptyset$ $\triangleright$ armazena os nós bloqueados
- 2 $P \leftarrow \langle s \rangle$
- 3 $i \leftarrow s$
- 4 **enquanto** $s \notin B$ **faça**
- 5 **se** $i = t$ **então**
- 6 $\langle \check{x}, A_{\check{x}} \rangle \leftarrow \text{INCREMENTE-FLUXO}(\check{x}, P, A_{\check{x}}, u)$
- 7 $P \leftarrow \langle s \rangle$
- 8 $i \leftarrow s$
- 9 **se** algum ij em $A_{\check{x}}$ é justo e $j \notin B$
- 10 **então** $\text{AVANCE}(ij)$
- 11 **senão** $\text{RETROCEDA}(i)$
- 12 **devolva** $\check{x}$

Invariantes

No início de cada iteração das linhas 4–11 vale que:

- (i0) $\tilde{x}$ é um st -pseudofluxo que respeita u ;
- (i2) para cada i em B não existe caminho de i a t em $(N, A_{\tilde{x}})$ contendo apenas arcos justos (caminho de i a t no “grafo em camadas”);
- (i3) P é um caminho de s a i em $(N, A_{\tilde{x}})$;
- (i4) todos os arcos de P são justos;
- (i5) todos os nós de P estão em $N - B$, exceto talvez se $P = \langle s \rangle$;

Avance e retroceda

AVANCE (i, j)

- 1 acrescente i, j ao final de P
- 2 $i \leftarrow j$

RETROCEDA (i)

- 1 $B \leftarrow B \cup \{i\}$ $\triangleright$ nó i está “bloqueado”
- 2 **se** $i \neq s$
- 3 **então** remova o último nó de P
- 4 $i \leftarrow$ último nó de P

Consumo de tempo: $O(1)$.

Algoritmo de incremento

Recebe um *st*-pseudofluxo $\check{x}$ e um caminho de incremento P e **devolve** o pseudofluxo $\check{x}$ após enviar “ δ unidades de fluxo através de P ” e os arcos que ainda **não estão saturados**.

INCREMENTE-FLUXO ($\check{x}, P, A_{\check{x}}, u$)

- 1 $\delta \leftarrow \min\{u(ij) - \check{x}(ij) : ij \text{ é arco de } P\}$
- 2 **para cada** arco ij em P **faça**
- 3 $\check{x}(ij) \leftarrow \check{x}(ij) + \delta$
- 4 $\check{x}(ji) \leftarrow \check{x}(ji) - \delta$
- 5 **se** $\check{x}(ij) = u(ij)$
- 6 **então** $A_{\check{x}} \leftarrow A_{\check{x}} - \{ij\}$
- 7 **devolva** $\check{x}$ e $A_{\check{x}}$

Consumo de tempo: $O(|P|) = O(n)$

Consumo de tempo do passo bloqueador

Considere uma execução de PASSO-BLOQUEADOR.

Fato 1. O número de execuções de INCREMENTE-FLUXO é $\leq m$.

Demonstração (rascunho): Em cada execução pelo menos um arco é saturado e removido de $A_{\tilde{x}}$.

Fato 2. O número de execuções de RETROCEDA é $\leq n$.

Demonstração (rascunho): Em cada execução um nó é acrescentado a B e jamais é removido.

Consumo de tempo do passo bloqueador

Fato 3. O número de execuções de **AVANCE** é $\leq nm$.

Demonstração (rascunho): Hmmmm, aqui precisamos pensar... *current-arc data structure*.

Antes da execução de **INCREMENTE-FLUXO** o número de **AVANCE**(ij) em que o arco ij **será** um arco do caminho de incremento resultante é $\leq n - 1$. Logo, devido ao fato 1, o número total de execuções desses **AVANCE** é $\leq (n - 1)m$;

O número total de **AVANCE**(ij) em que o arco ij **não será** um arco do próximo caminho de incremento é $\leq$ número de execuções de **RETROCEDA**. Logo o número de execuções desses **AVANCE** é $\leq n$;

Portanto, o número de execuções de **AVANCE** é

$$\leq (n - 1)m + n = nm$$

Consumo de tempo do passo bloqueador

Algoritmo	número máximo de execuções	consumo total de tempo
AVANCE	$\leq nm$	$O(nm)$
RETROCEDA	$\leq n$	$O(n)$
INCREMENTE-FLUXO	$\leq m$	$O(nm)$

O consumo de tempo do algoritmo
PASSO-BLOQUEADOR é $O(nm)$.

Consumo de tempo do Dinits

Algoritmo	número máximo de execuções	consumo total de tempo
POTENCIAL-ÓTIMO	$\leq n - 1$	$O(n(n + m))$
PASSO-BLOQUEADOR	$\leq n$	$O(n^2 m)$

O consumo de tempo do algoritmo **DINITS** é $O(n^2 m)$.

Resumo

O número de fases do algoritmo **DINITS** é $\leq n - 1$.

O consumo de tempo do algoritmo
PASSO-BLOQUEADOR é $O(nm)$.

O consumo de tempo do algoritmo **DINITS** é
 $O(n^2m)$.

Observações

	consumo de tempo passo bloqueador	consumo de tempo
DINITS	$O(nm)$	$O(n^2m)$
Karzanov	$O(n^2)$	$O(n^3)$
Sleator-Tarjan	$m \log n$	$O(nm \log n)$

Karzanov utilizou “pré-fluxos”.

Sleator-Tarjan utilizaram “*dynamic-tree data structure*”.

Problema: É possível encontrar um fluxo bloqueador em tempo $O(n + m)$?