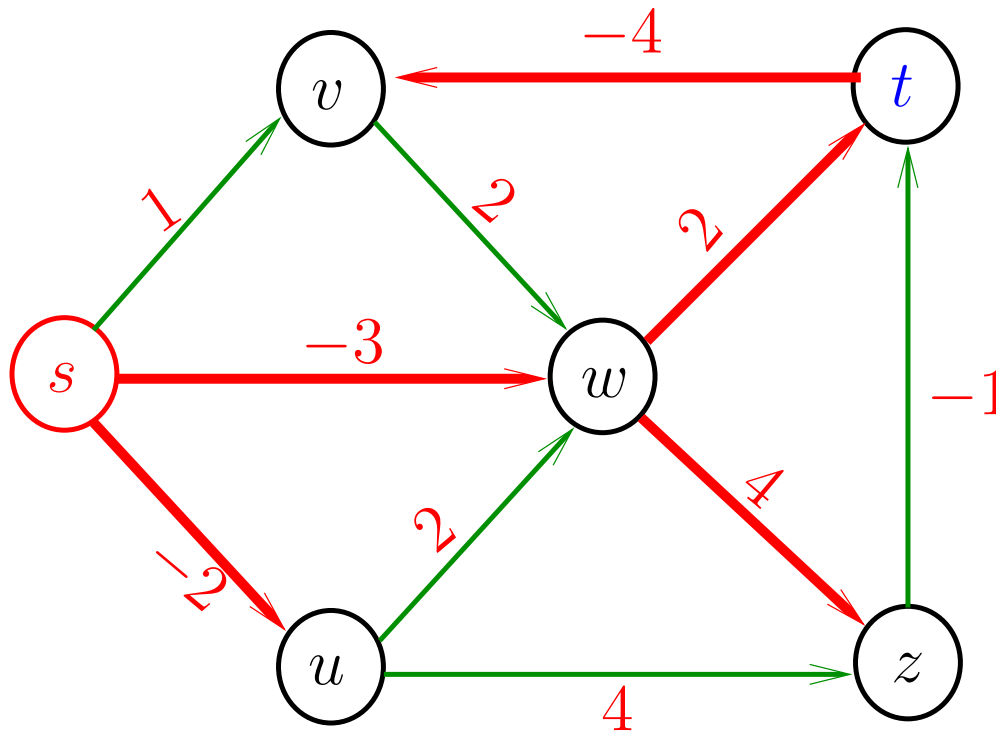


Melhores momentos

AULA PASSADA

Problema

Problema do caminho de custo mínimo: Dada uma rede (N, A, c) com função-custo $c : A \rightarrow \mathbb{Z}$ e um nó s , encontrar, para cada nó t , um caminho de custo mínimo de s a t .



Complexidade computacional

O problema do caminho mínimo é tão difícil quanto o problema do caminho hamiltoniano.

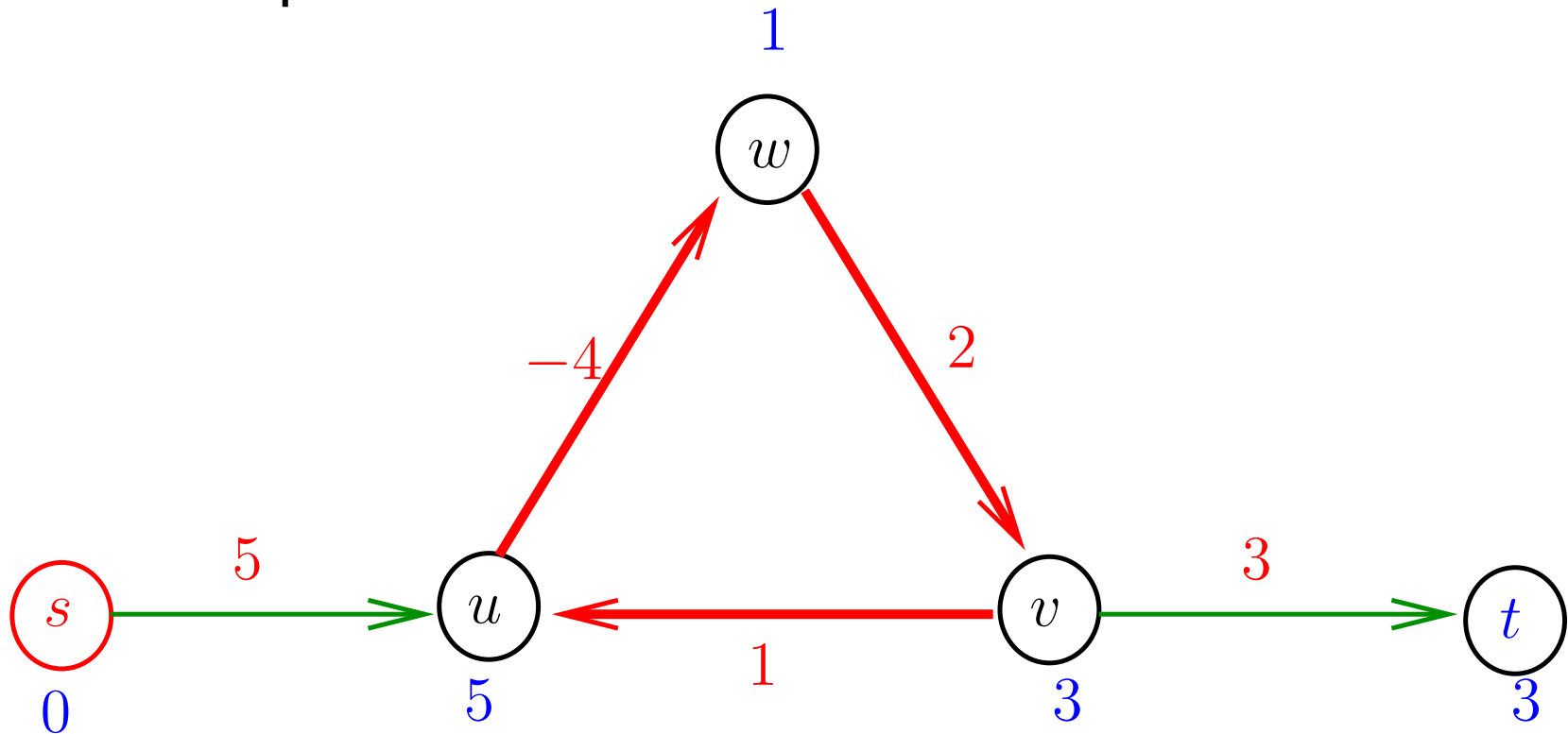
O problema do caminho de custo mínimo é NP-difícil.

Em outras palavras: ninguém conhece um algoritmo eficiente para o problema . . .

Se alguém conhece, não contou para ninguém . . .

Ciclos negativos

Se a rede possui um **ciclo (de custo) negativo** então não existe um c -potencial.



Algoritmo genérico

Recebe uma rede (N, A, c) com função custo $c : A \rightarrow \mathbb{Z}$ **sem ciclos negativos** e um nó s e **devolve** uma função-predecessor π e um c -potencial y com a seguinte propriedade: para todo nó t , a função π determina um caminho P de s a t tal que $c(P) = y(t) - y(s)$.

FORD (N, A, c, s)

1 **para cada** i em N **faça**

2 $y(i) \leftarrow nC + 1 \quad \triangleright C := \max\{|c(ij)| : ij \in A\}$

3 $\pi(i) \leftarrow \text{NIL}$

4 $y(s) \leftarrow 0$

5 **enquanto** $y(j) > y(i) + c(ij)$ **para algum** $ij \in A$ **faça**

6 $y(j) \leftarrow y(i) + c(ij)$

7 $\pi(j) \leftarrow i$

8 **devolva** π e y

Consumo de tempo

O consumo de tempo do algoritmo
CAMINHO-CURTO-GENÉRICO é $O(n^2 m (C + 1))$.

Este consumo de tempo **não** é **polinomial**.

Algoritmo de Ford-Bellman

FORD-BELLMAN (N, A, c, s)

```
1  para cada  $i$  em  $N$  faça
2       $y(i) \leftarrow \infty$ 
3       $\pi(i) \leftarrow \text{NIL}$ 
4   $y(s) \leftarrow 0$ 

5  repita  $n - 1$  vezes
6      para cada arcor  $ij$  em  $A$  faça
7          se  $y(j) > y(i) + c(ij)$ 
8              então  $y(j) \leftarrow y(i) + c(ij)$ 
9                   $\pi(j) \leftarrow i$ 

10 devolva  $\pi$  e  $y$ 
```

Relação invariante chave

Chamemos de **passo** cada execução das linhas 6–9.

Após o k -ésimo passo vale que

se P é um st -passeio com $\leq k$ arcos então

$$y(t) \leq c(P).$$

e o **grafo de predecessores** contém um st -passeio de custo $\leq y(t)$.

Consumo de tempo

O consumo de tempo do algoritmo
FORD-BELLMAN é $O(nm)$.

Este consumo de tempo é (**fortemente**) **polinomial**.

Implementação FIFO de Ford-Bellman

FIFO-FORD-BELLMAN (N , A , c , s)

```
1  para cada  $i$  em  $N$  faça
2       $y(i) \leftarrow \infty$ 
3       $\pi(i) \leftarrow \text{NIL}$ 
4   $y(s) \leftarrow 0$ 
5   $L \leftarrow s$   $\triangleright L$  funciona como uma fila
6  enquanto  $L \neq \langle \rangle$  faça
7      retire o primeiro elemento, digamos  $i$ , de  $L$ 
8      para cada  $ij$  em  $A(i)$  faça
9          se  $y(j) > y(i) + c(ij)$ 
10             então  $y(j) \leftarrow y(i) + c(ij)$ 
11                  $\pi(j) \leftarrow i$ 
12                 se  $j \notin L$ 
13                     então acrescente  $j$  ao final de  $L$ 
14  devolva  $\pi$  e  $y$ 
```

AULA 11

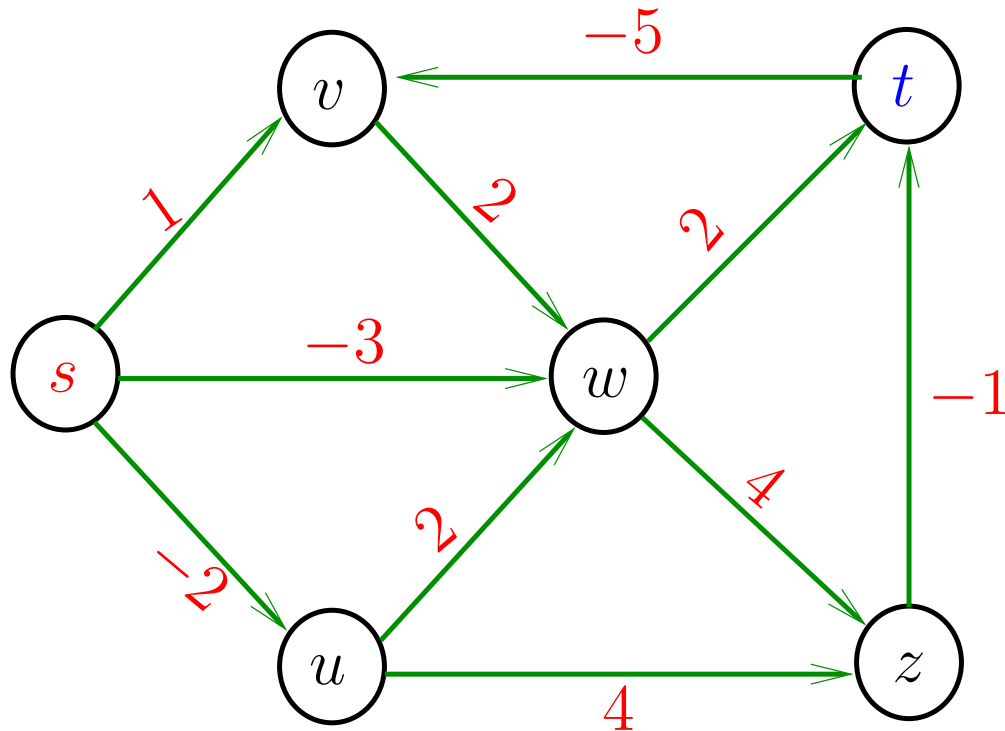
Ciclos negativos

PF 7.1, 7.2, 7.3, 7.4

Problema

Problema do ciclo negativo: Dada uma rede (N, A, c) com função-custo $c : A \rightarrow \mathbb{Z}$ e um nó s , **encontrar**, um ciclo de custo negativo.

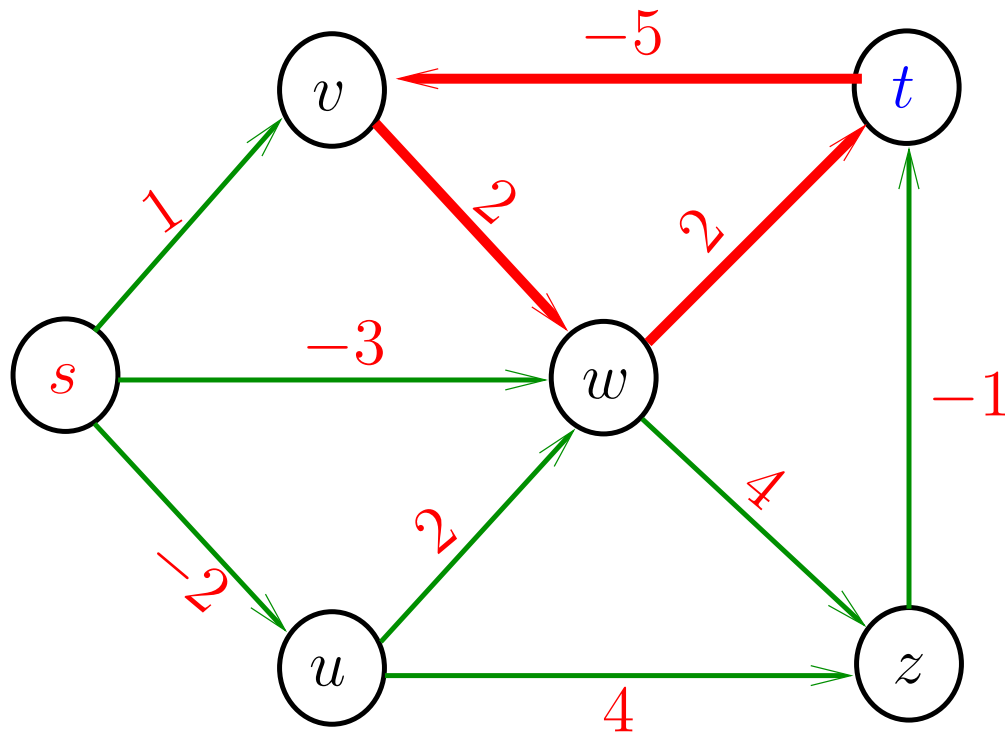
Entra:



Problema

Problema do ciclo negativo: Dada uma rede (N, A, c) com função-custo $c : A \rightarrow \mathbb{Z}$ e um nó s , **encontrar**, um ciclo de custo negativo.

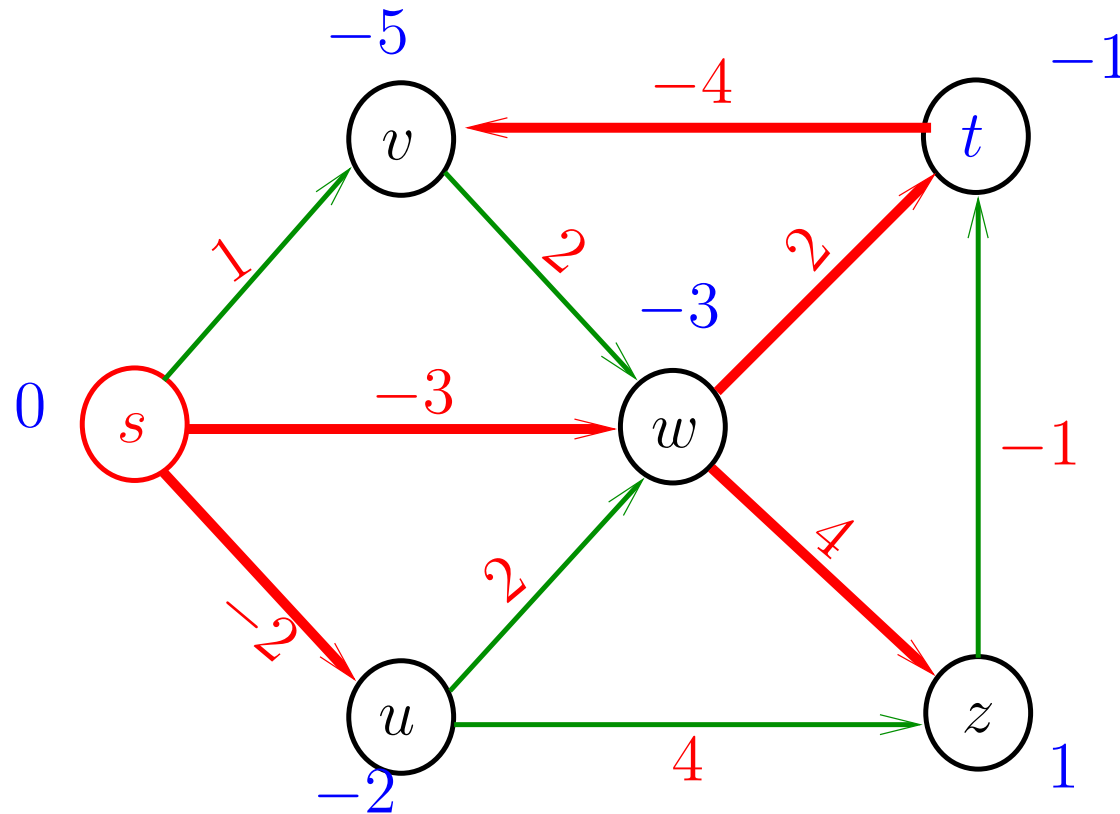
Sai:



Propriedade de c -Potenciais

(Lema da dualidade.) Se P é um passeio de s a t e y é um c -potencial então

$$c(P) \geq y(t) - y(s).$$

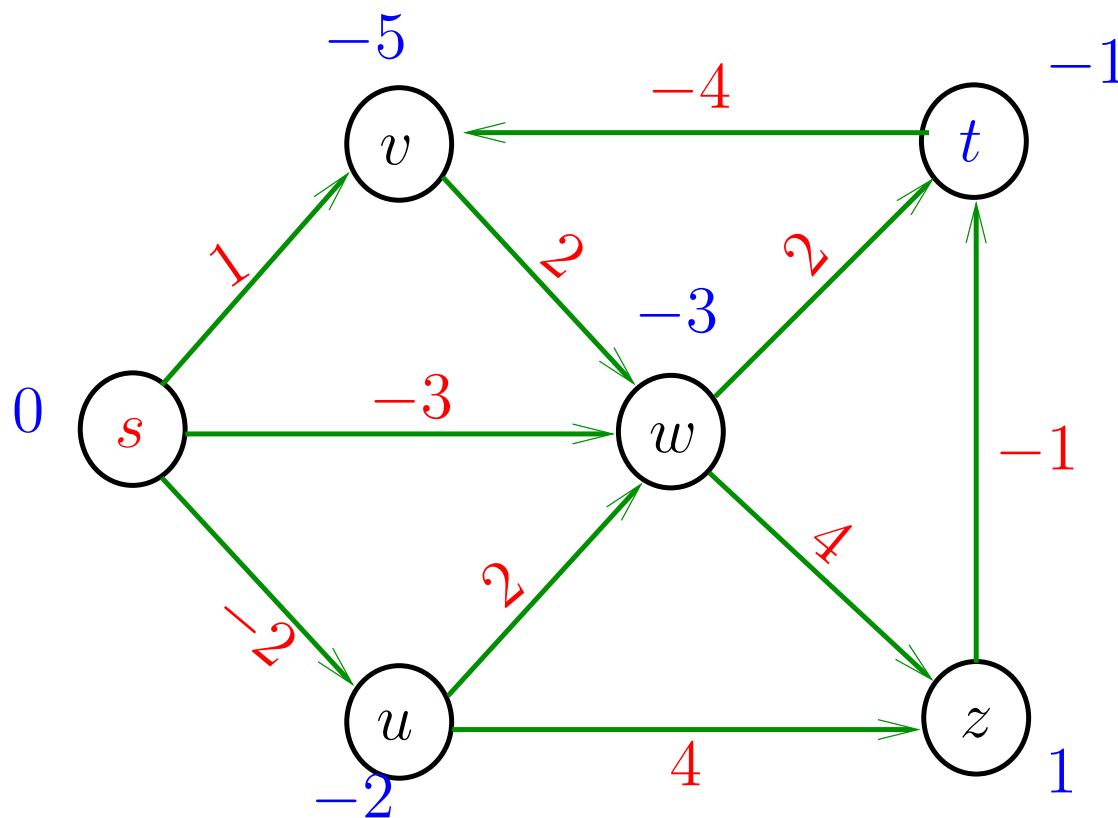


Consequência

Se uma rede (N, A, c) admite um c -potencial então

$$c(O) \geq 0,$$

para todo ciclo O .



Lembra?

Recebe um grafo (N, A) e devolve uma ciclo ou um -1 -potencial.

DAG-GENÉRICO (N, A)

1 para cada i em N faça

2 $y(i) \leftarrow 0$

3 $\pi(i) \leftarrow \text{NIL}$

4 enquanto $y(j) > y(i) - 1$ para algum $ij \in A$ faça

5 $y(j) \leftarrow y(i) - 1$

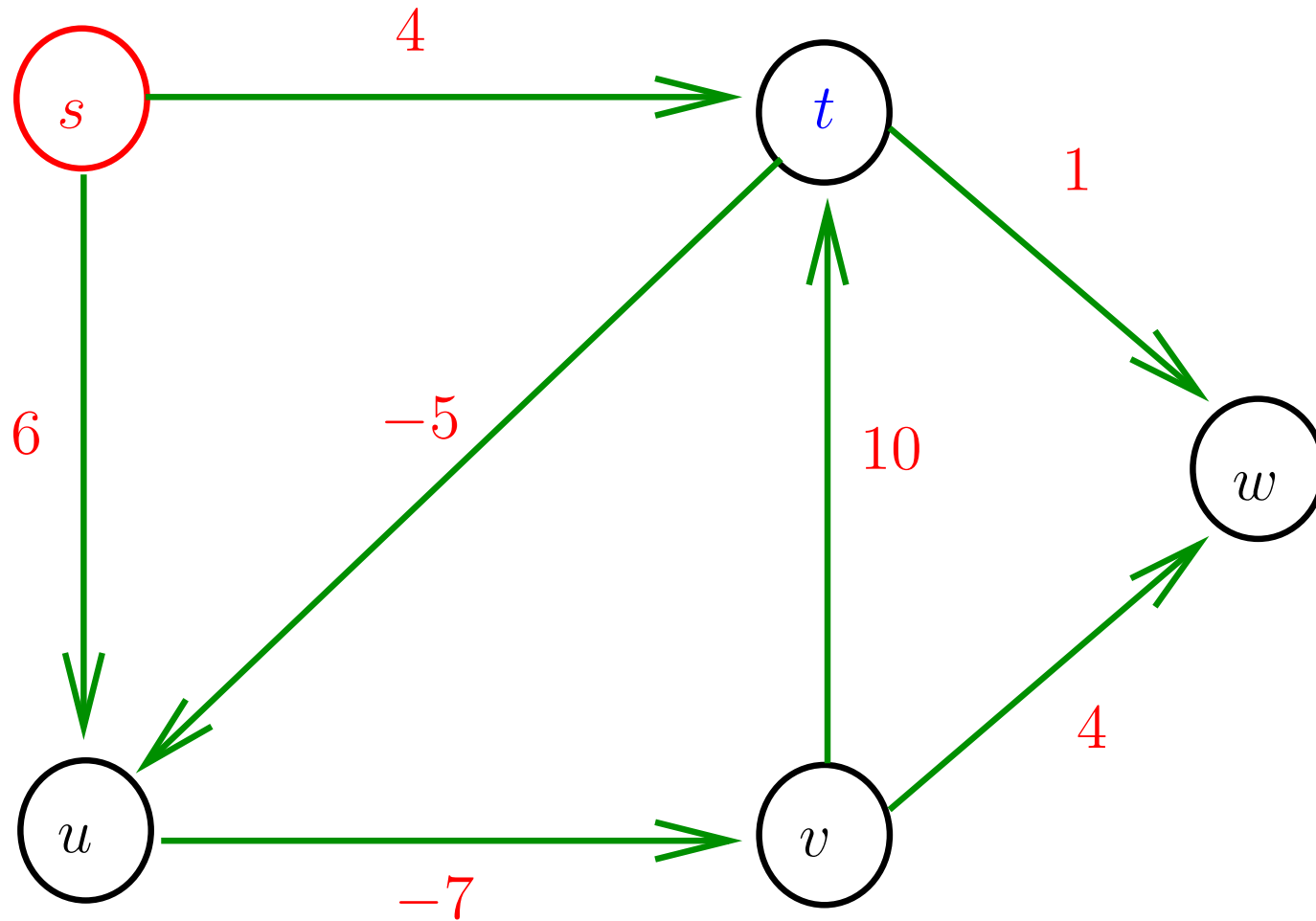
6 $\pi(j) \leftarrow i$

7 se $y(j) \leq -n$

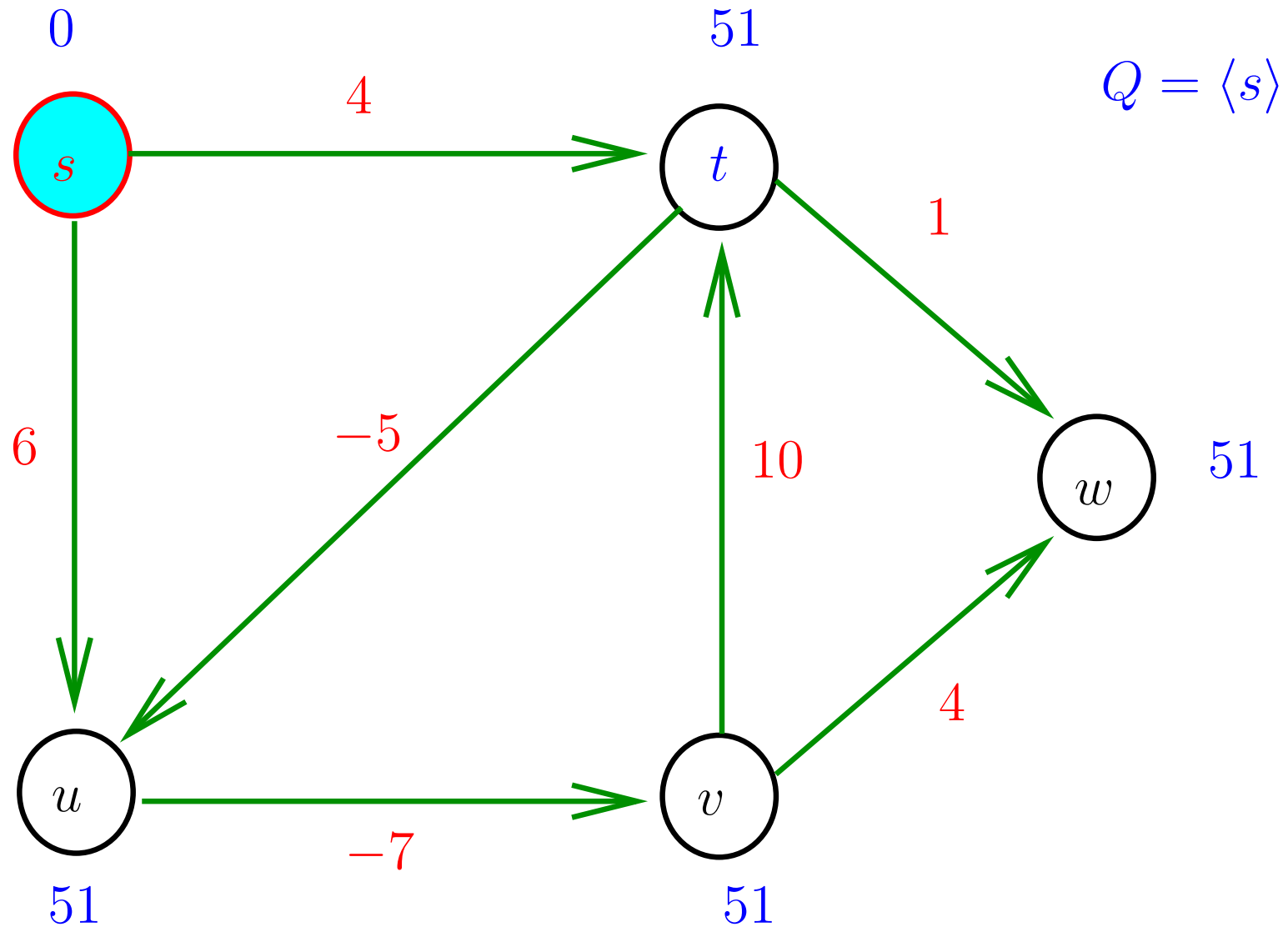
8 então devolva j e pare

9 devolva y

Simulação FIFO-Ford-Bellman

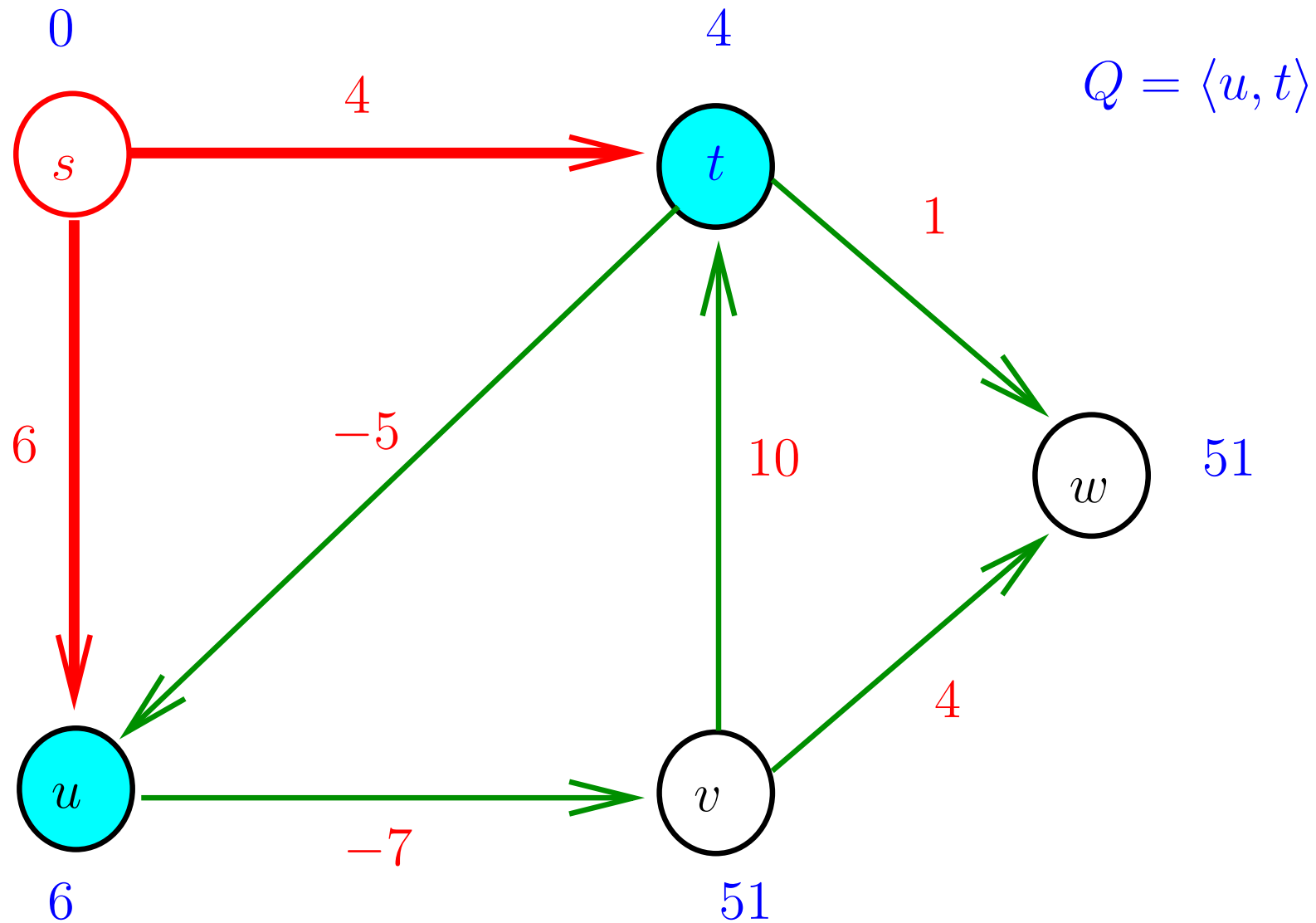


Simulação FIFO-Ford-Bellman



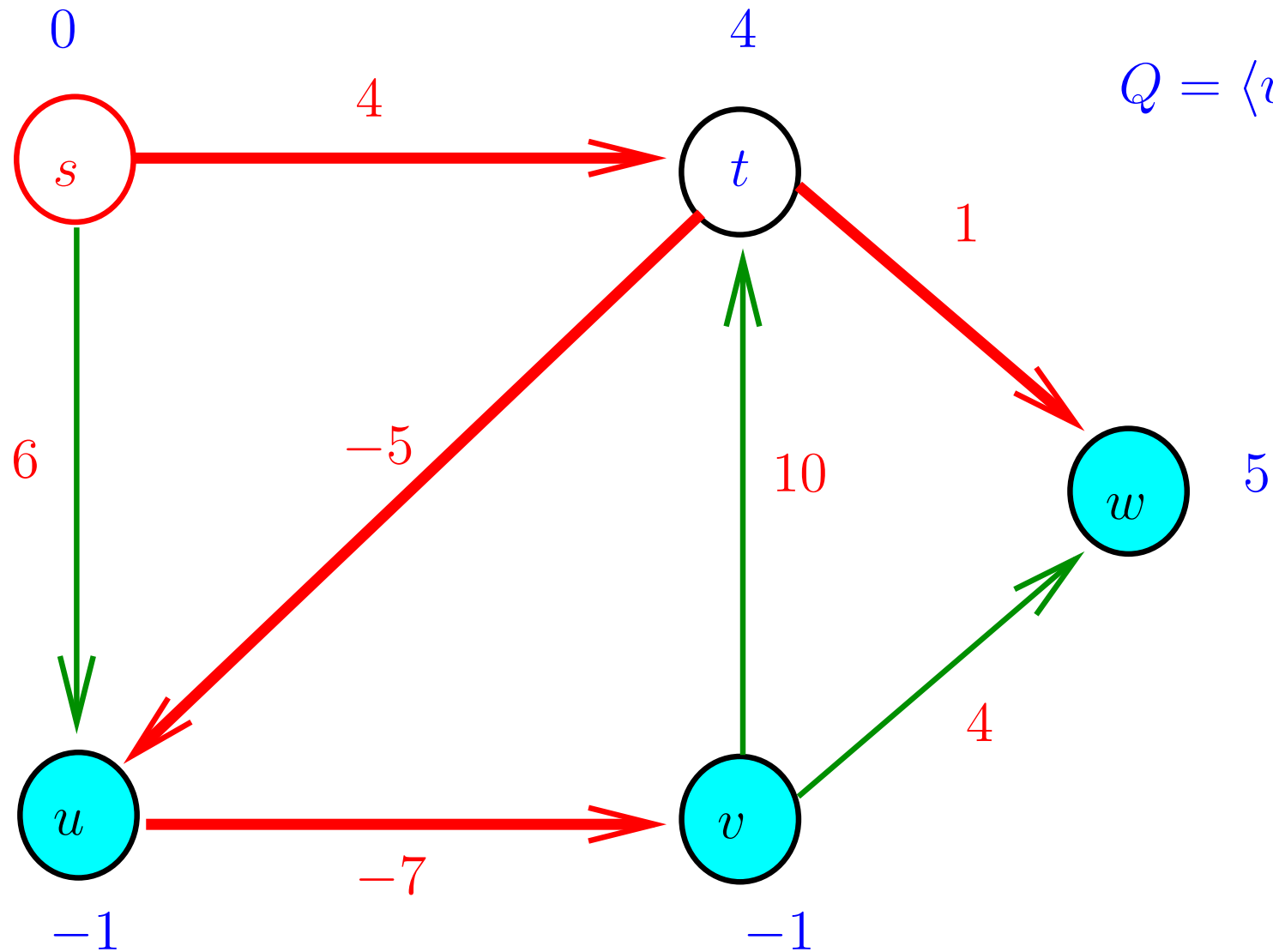
Início passo 0

Simulação FIFO-Ford-Bellman



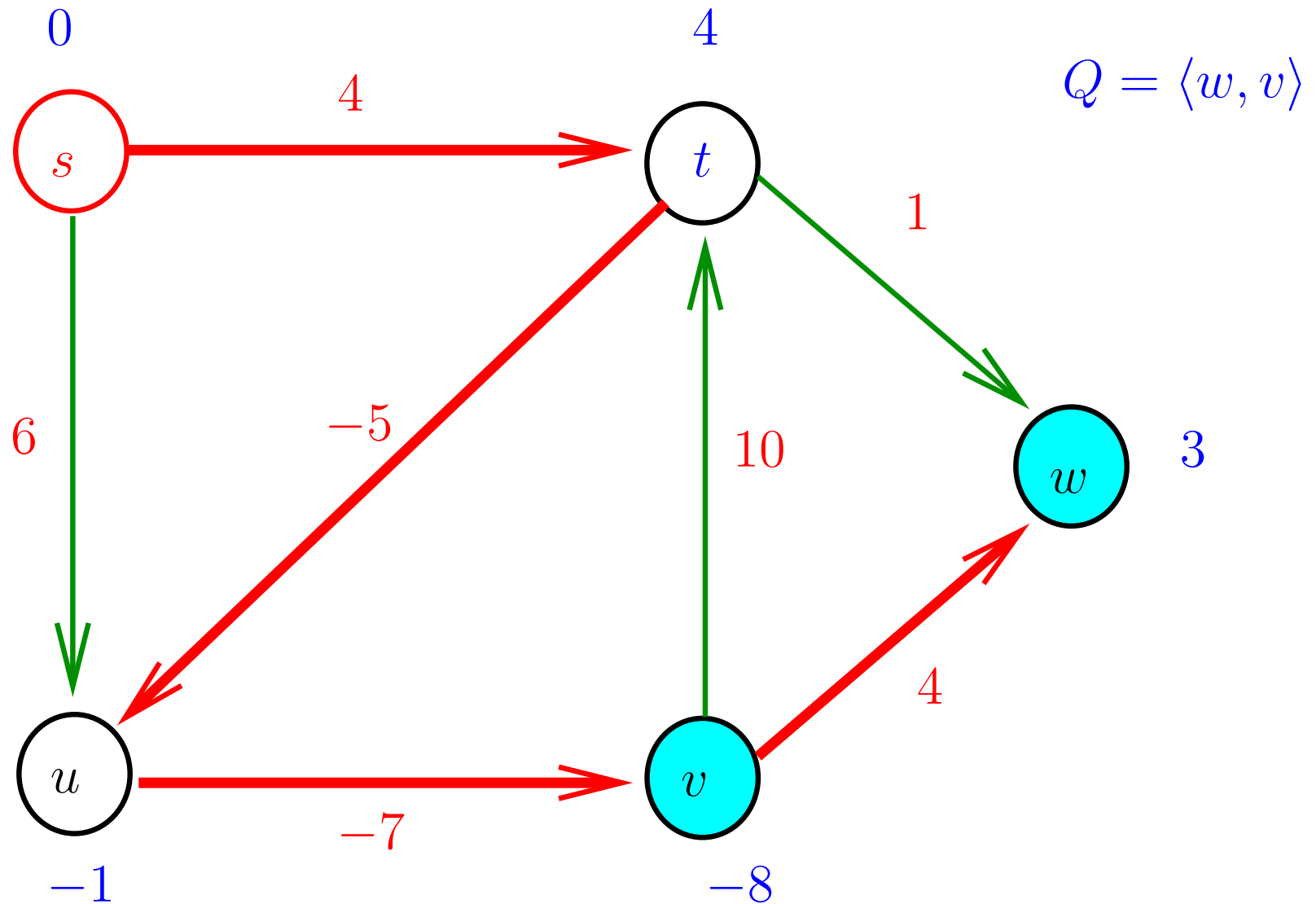
Fim passo 1

Simulação FIFO-Ford-Bellman

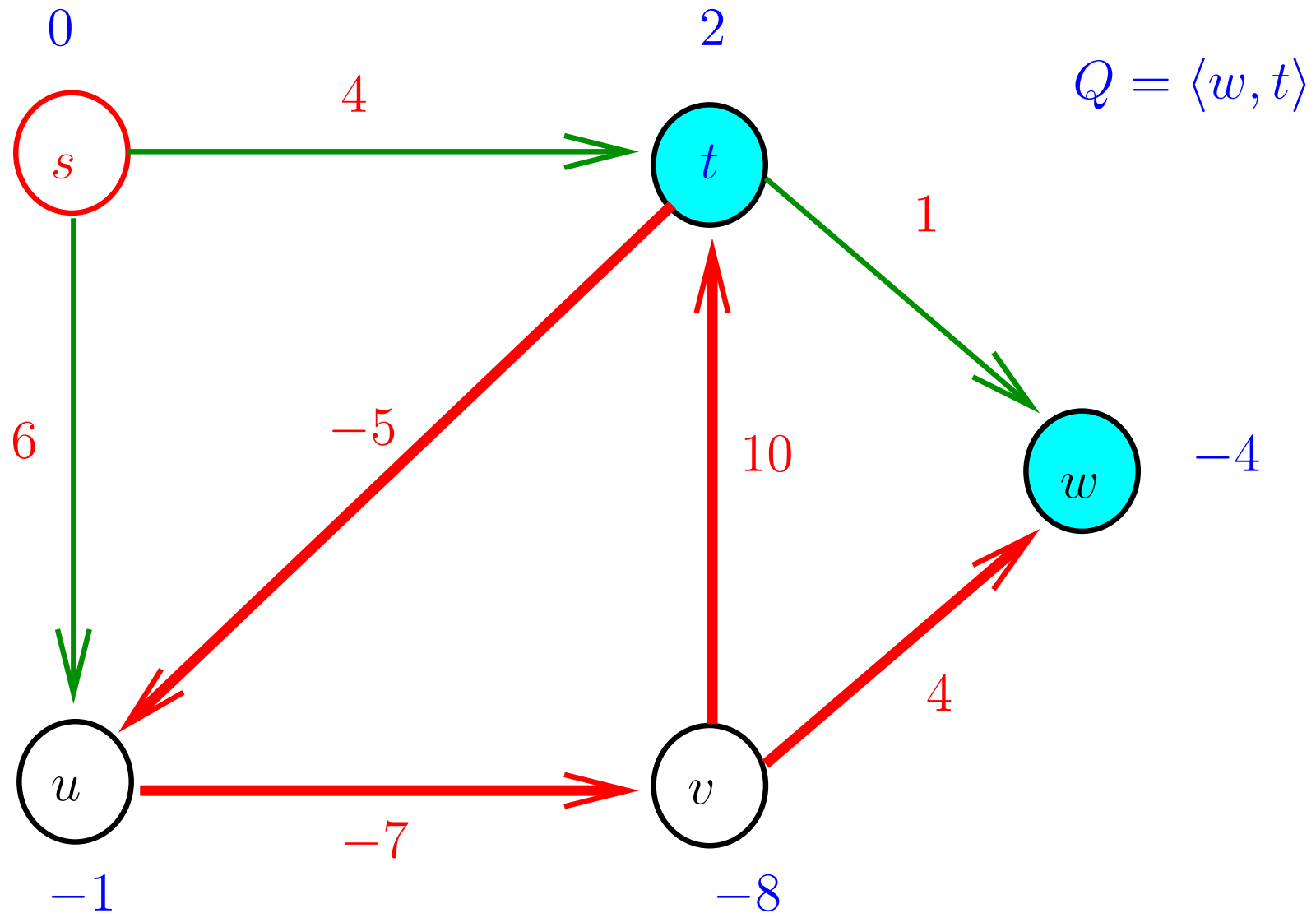


Fim passo 2

Simulação FIFO-Ford-Bellman

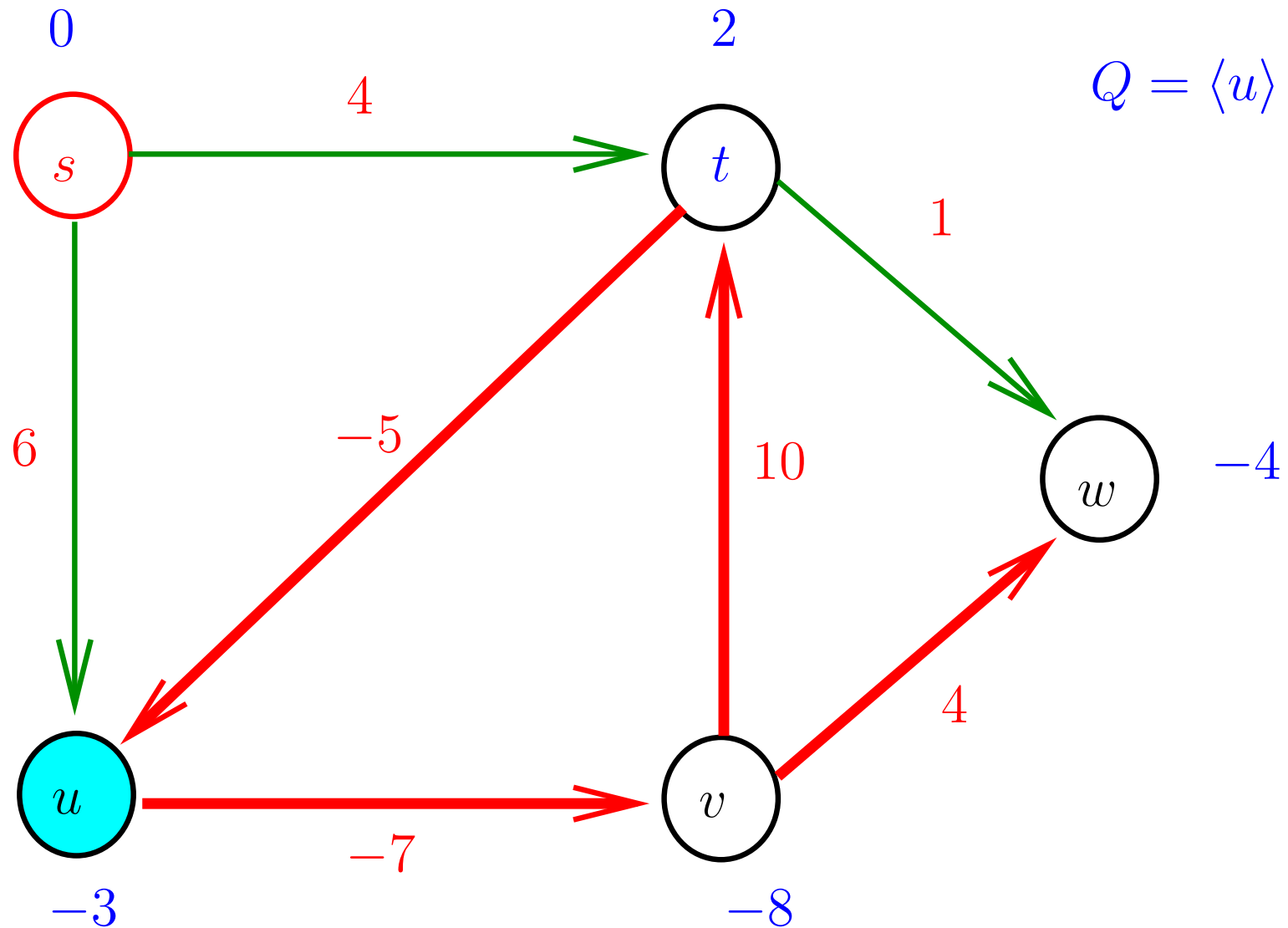


Simulação FIFO-Ford-Bellman



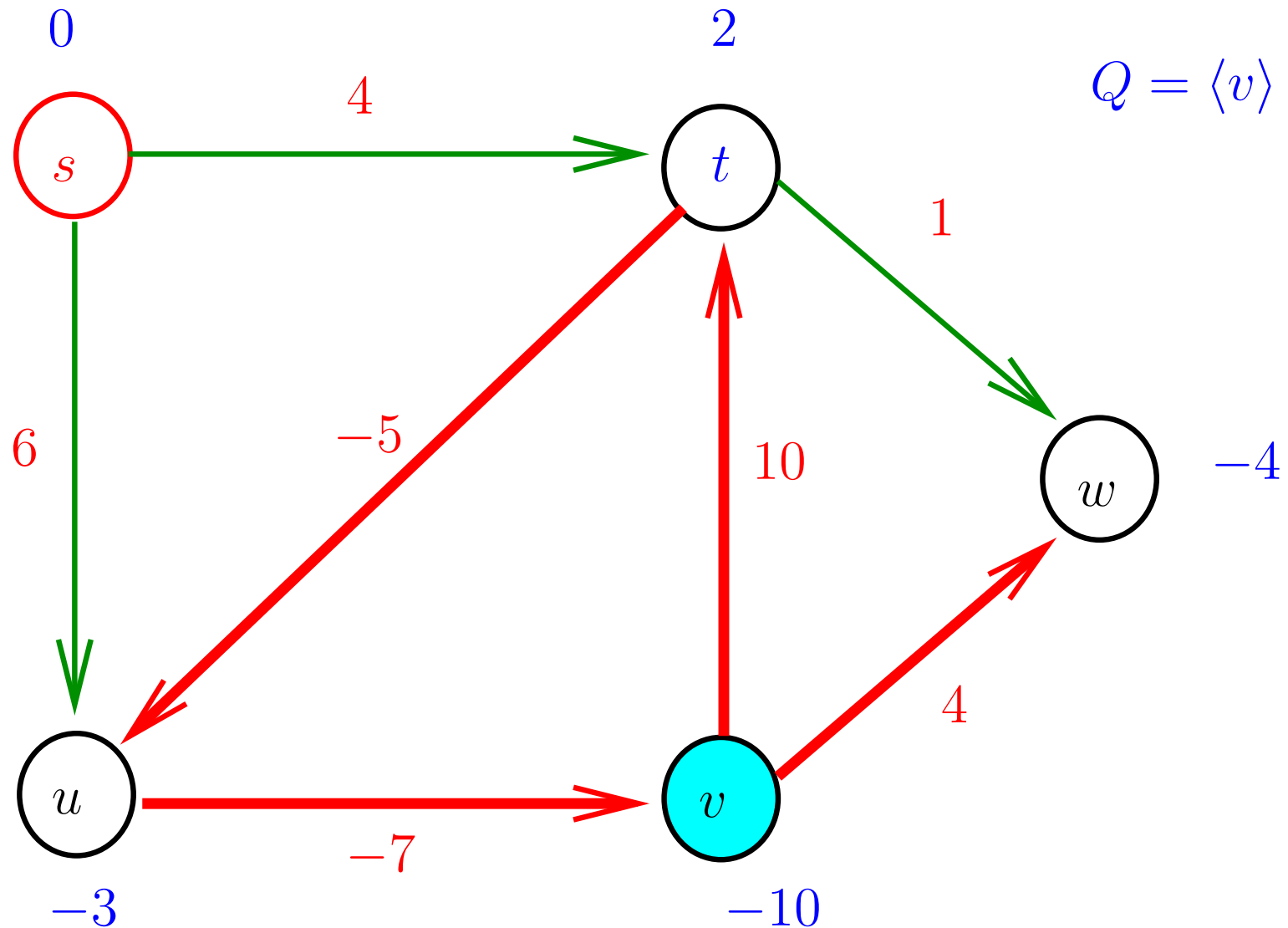
Fim passo 4

Simulação FIFO-Ford-Bellman



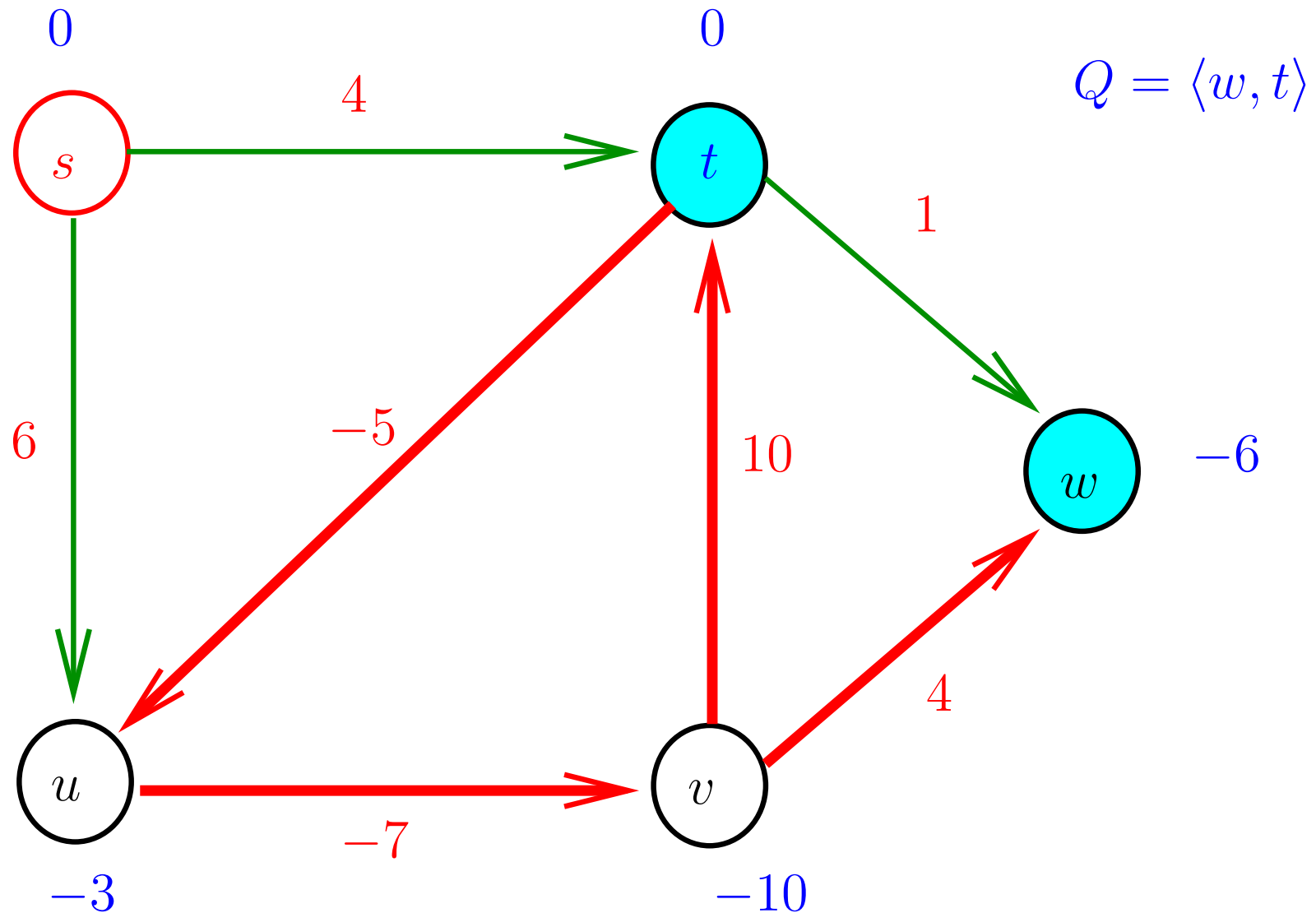
Fim passo 5

Simulação FIFO-Ford-Bellman



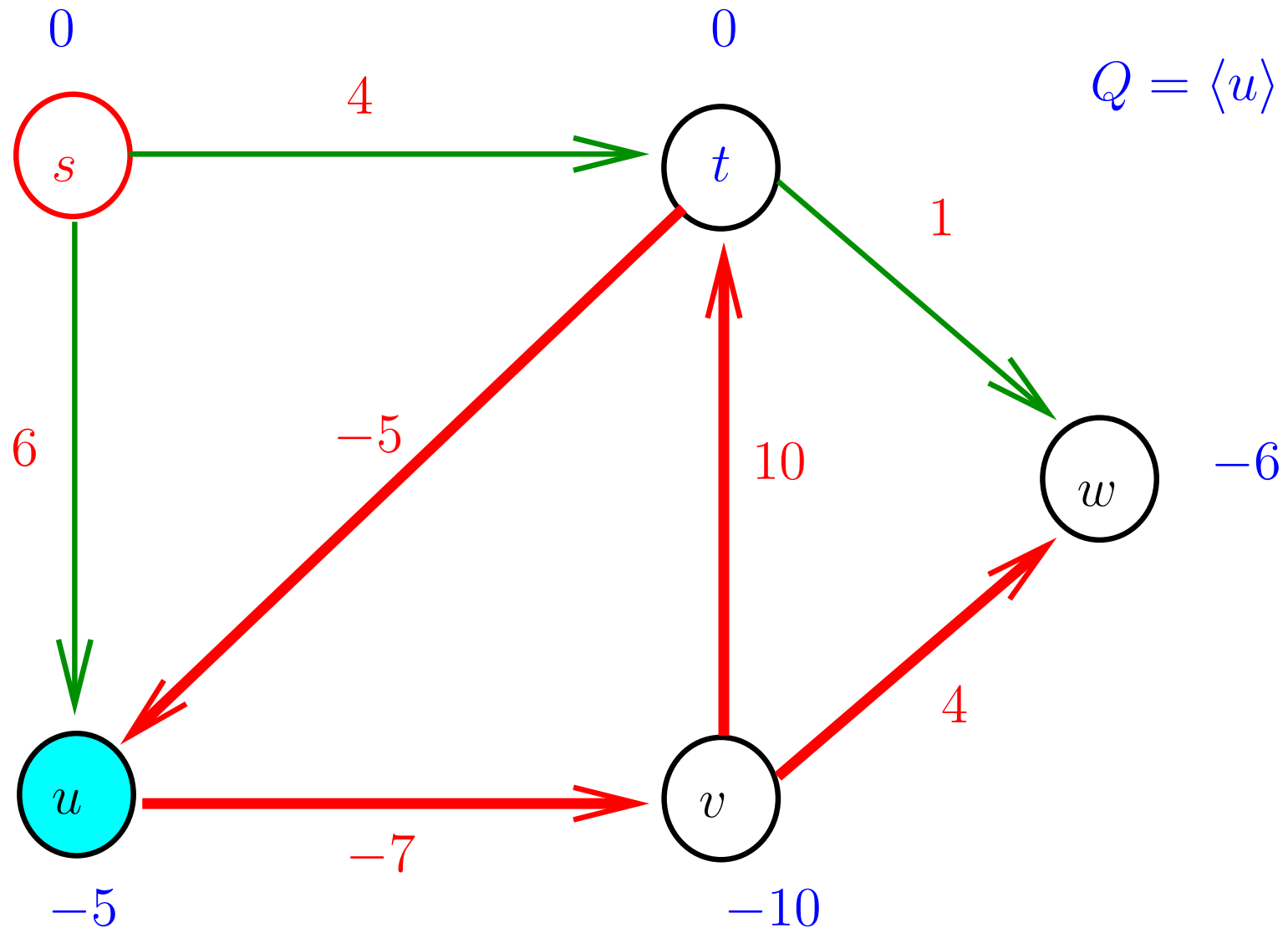
Fim passo 6

Simulação FIFO-Ford-Bellman



Fim passo 7

Simulação FIFO-Ford-Bellman



Fim passo 8

Algoritmo genérico

Recebe uma rede (N, A, c) e devolve um ciclo negativo ou c -potencial.

FORD-CICLO (N, A, c)

```
1   $C \leftarrow \max\{|c(ij)| : ij \in A\}$ 
2  para cada  $i$  em  $N$  faça
3       $y(i) \leftarrow 0$ 
4       $\pi(i) \leftarrow \text{NIL}$ 
5  enquanto  $y(j) > y(i) + c(ij)$  para algum  $ij \in A$  faça
6       $y(j) \leftarrow y(i) + c(ij)$ 
7       $\pi(j) \leftarrow i$ 
8      se  $y(j) \leq -nC$ 
9          então devolva  $j$  e  $\pi$  e pare
10 devolva  $y$ 
```

Invariantes

Na linha 5, antes da verificação da condição
" $y(j) > y(i) + c(ij) \dots$ " valem as seguintes invariantes:

(i1) para cada arco pq no grafo de predecessores tem-se
 $y(q) - y(p) \geq c(pq)$;

(i2) se O é um ciclo no grafo de predecessores, então

$$c(O) < 0.$$

(i3) para cada nó w , se $\pi(w) = \text{NIL}$ então $y(w) = 0$.

Rascunho da demonstração de (i2)

Considere o momento em que o algoritmo está prestes a incluir o arco ji no **grafo de predecessores** formando o ciclo

$$O = P \cdot \langle ji \rangle$$

Assim, $c(ji) < y(i) - y(j)$.

Pela invariante (i1) temos que

$$c(P) \leq y(j) - y(i).$$

Portanto,

$$\begin{aligned} c(O) &= c(P) + c(ji) \\ &\leq y(j) - y(i) + c(ji) \\ &< y(j) - y(i) + y(i) - y(j) = 0. \end{aligned}$$

Correção

Se o algoritmo termina na linha 10 $\Rightarrow$ **beleza!**

Suponha que o algoritmo termina na linha 8 e que

$$\langle j, \pi(j), \pi(\pi(j)), \pi(\pi(\pi(j))), \dots \rangle$$

seja uma seqüência **finita** que termina em w .

Logo, $\pi(w) = \text{NIL}$ e (i3) implica que $y(w) = 0$.

Suponha que P é o correspondente caminho de w a j .

Devido a (i1) temos que

$$y(j) = y(j) - y(w) \geq c(P) > -nC.$$

Logo, a seqüência é **infinita** e o grafo de predecessores possui um ciclo O . Por (i2) temos que $c(O) < 0$.

Conclusão

Da propriedade dos c -potenciais (**lema da dualidade**) e da correção do algoritmo **FORD-CICLO** concluimos o seguinte:

(**Teorema da viabilidade**) Se (N, A, c) é uma rede com função custo $c : A \rightarrow \mathbb{Z}$, então vale uma, e apenas uma, das seguintes afirmações:

- ou existe um ciclo negativo
- ou existe um c -potencial.

Consumo de tempo

O número de iterações das linhas 5–9 é $< n^2 C$.

O consumo de tempo do algoritmo FORD-CICLO é
 $O(n^2 m C)$.

Este consumo de tempo não é polinomial.

Implementação FIFO de Ford-Bellman

FIFO-FORD-BELLMAN-CICLO (N, A, c)

```
1  para cada  $i$  em  $N$  faça
2       $y(i) \leftarrow 0$     $\pi(i) \leftarrow \text{NIL}$     $\gamma(i) \leftarrow 0$ 
3   $L \leftarrow N$ 
4  enquanto  $L \neq \langle \rangle$  faça
5      retire o primeiro elemento, digamos  $i$ , de  $L$ 
6      para cada  $ij$  em  $A(i)$  faça
7          se  $y(j) > y(i) + c(ij)$ 
8              então  $y(j) \leftarrow y(i) + c(ij)$ 
9                   $\pi(j) \leftarrow i$ 
10             se  $j \notin L$  então
11                 acrescente  $j$  ao final de  $L$ 
12              $\gamma(i) \leftarrow \gamma(i) + 1$ 
13             se  $\gamma(i) > n - 1$  então devolva  $i$  e  $\pi$  e pare
14  devolva  $y$ 
```

Consumo de tempo

O consumo de tempo do algoritmo
FIFO-FORD-BELLMAN-CICLO é $O(nm)$.

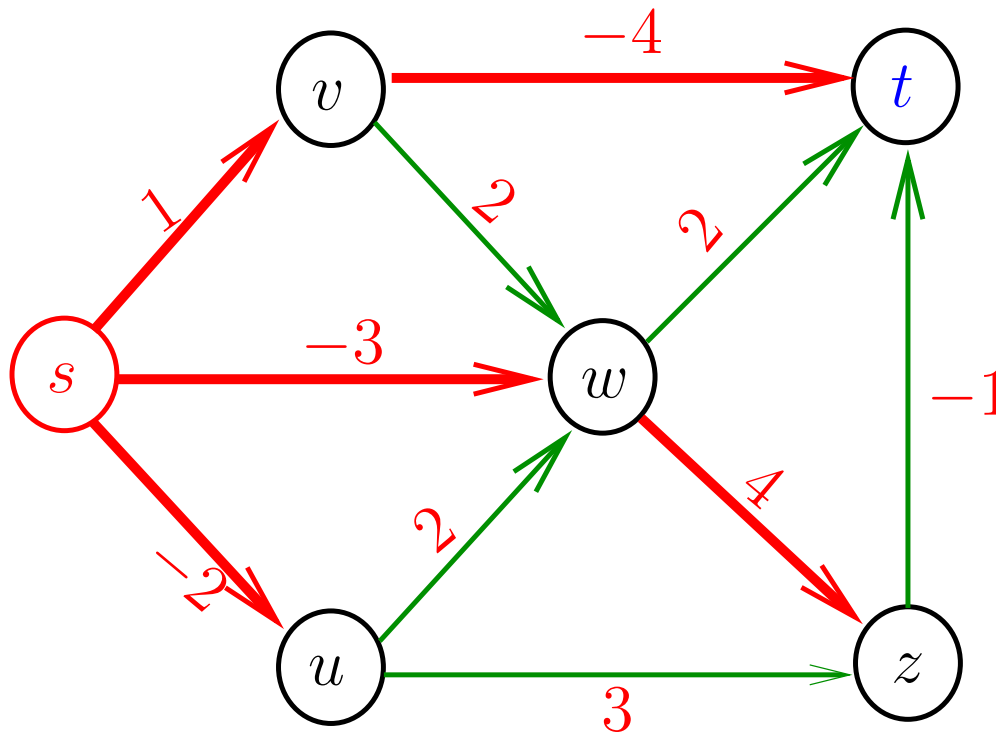
Este consumo de tempo é (fortemente) polinomial.

Caminhos mínimos em redes acíclicas

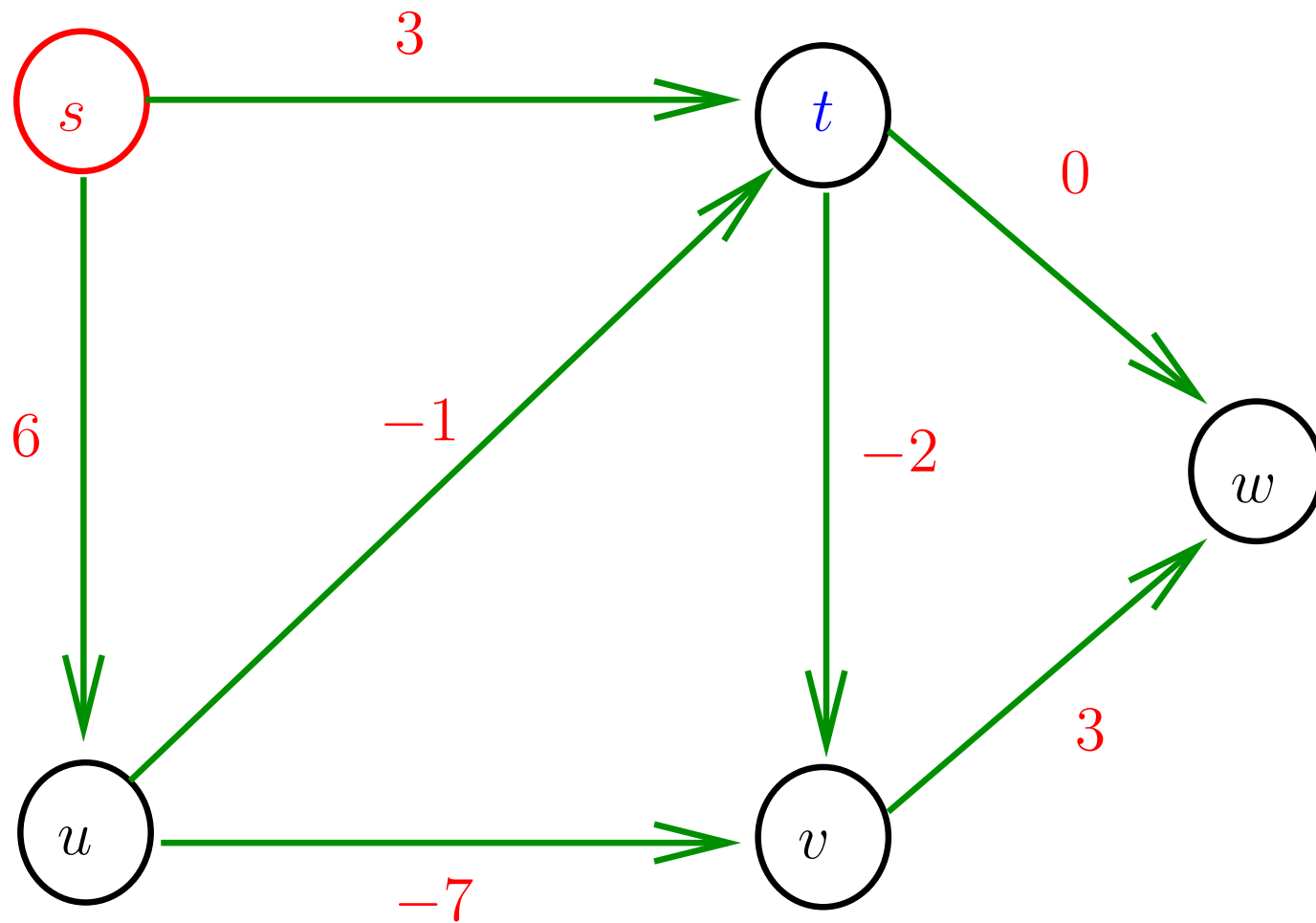
PF 9.1

Problema

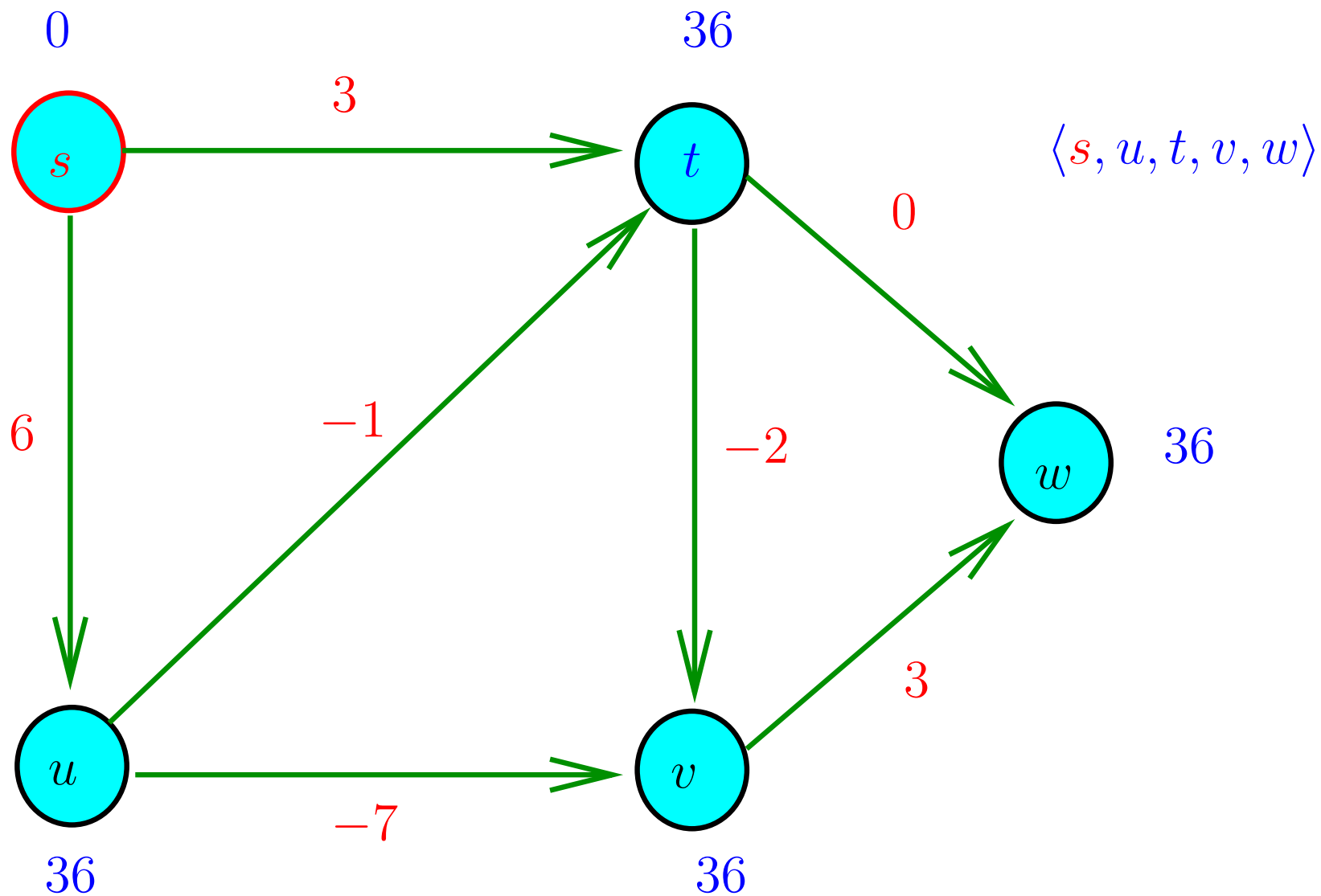
Problema do caminho de custo mínimo em redes acíclicas:
Dada uma rede (N, A, c) acíclica com função-custo $c : A \rightarrow \mathbb{Z}$ e um nó s , encontrar, para cada nó t , um caminho de custo mínimo de s a t .



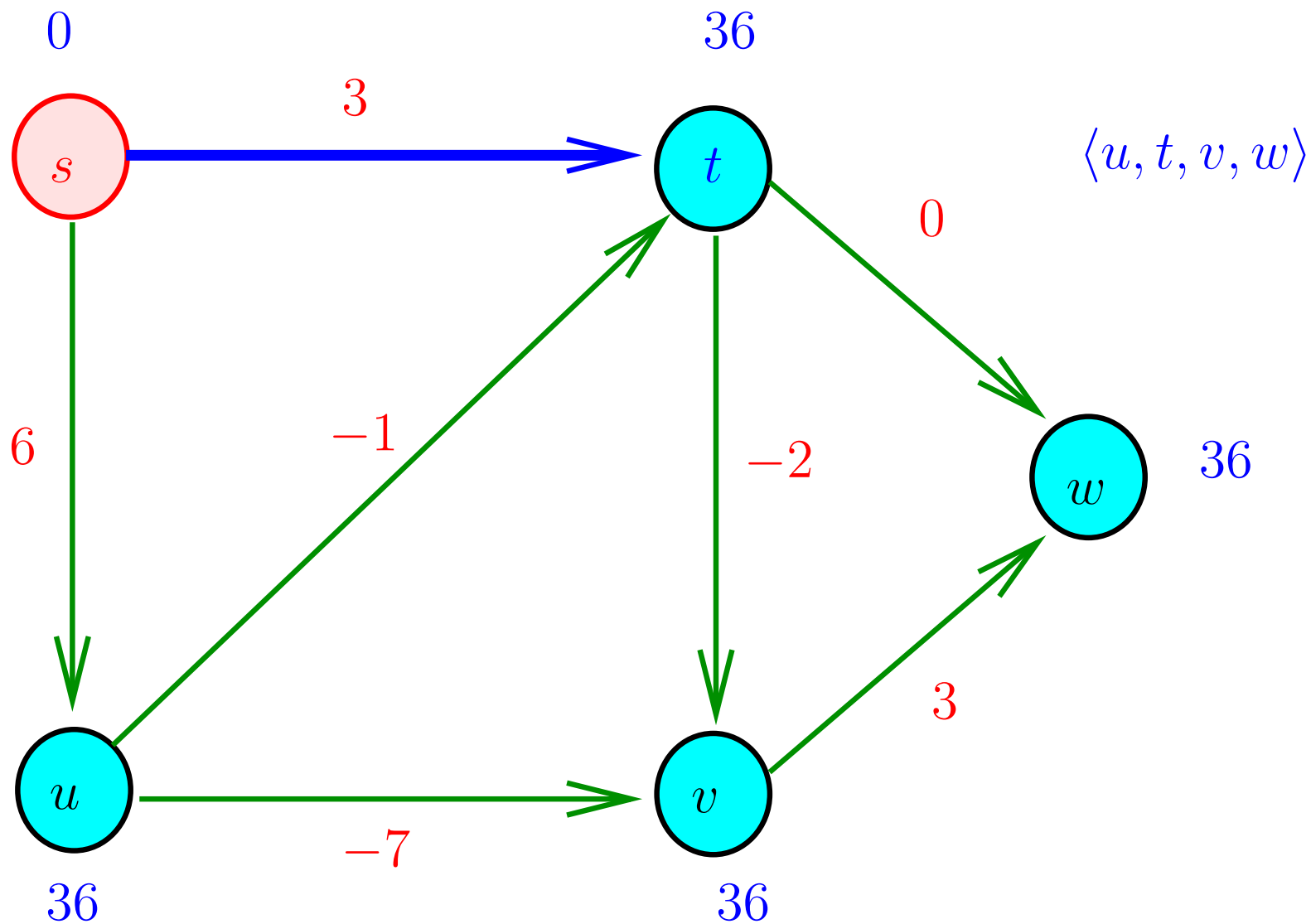
Simulação



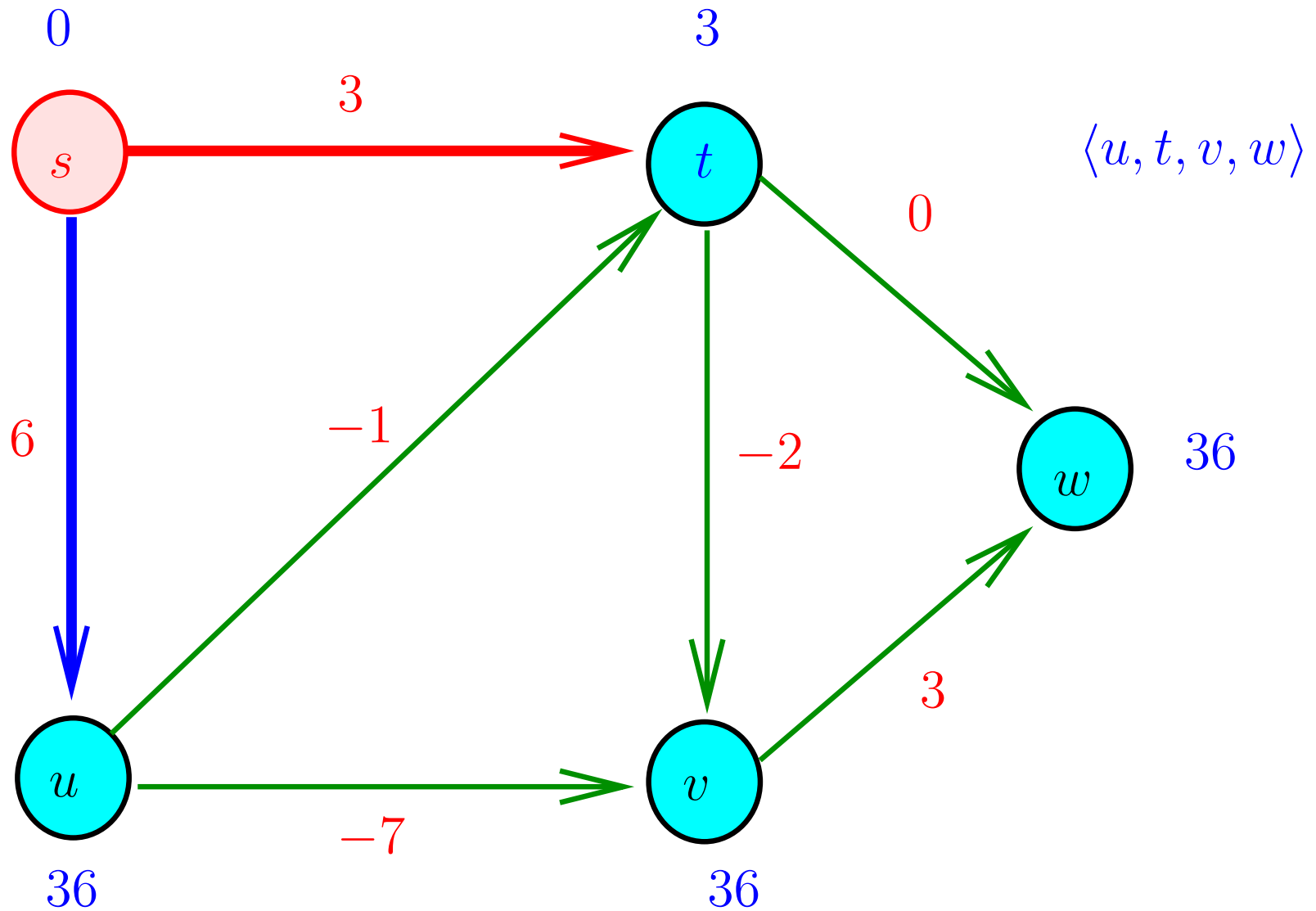
Simulação



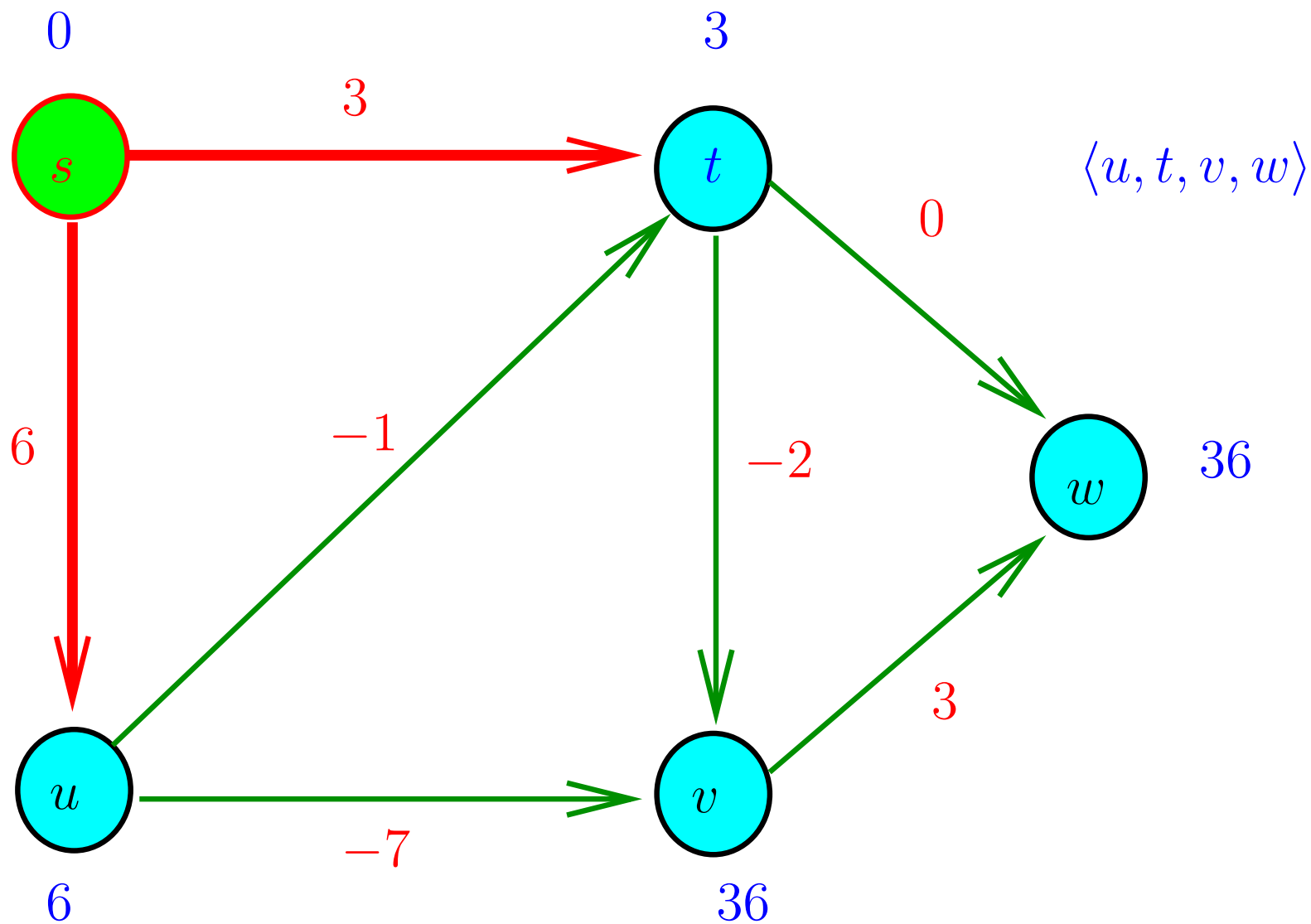
Simulação



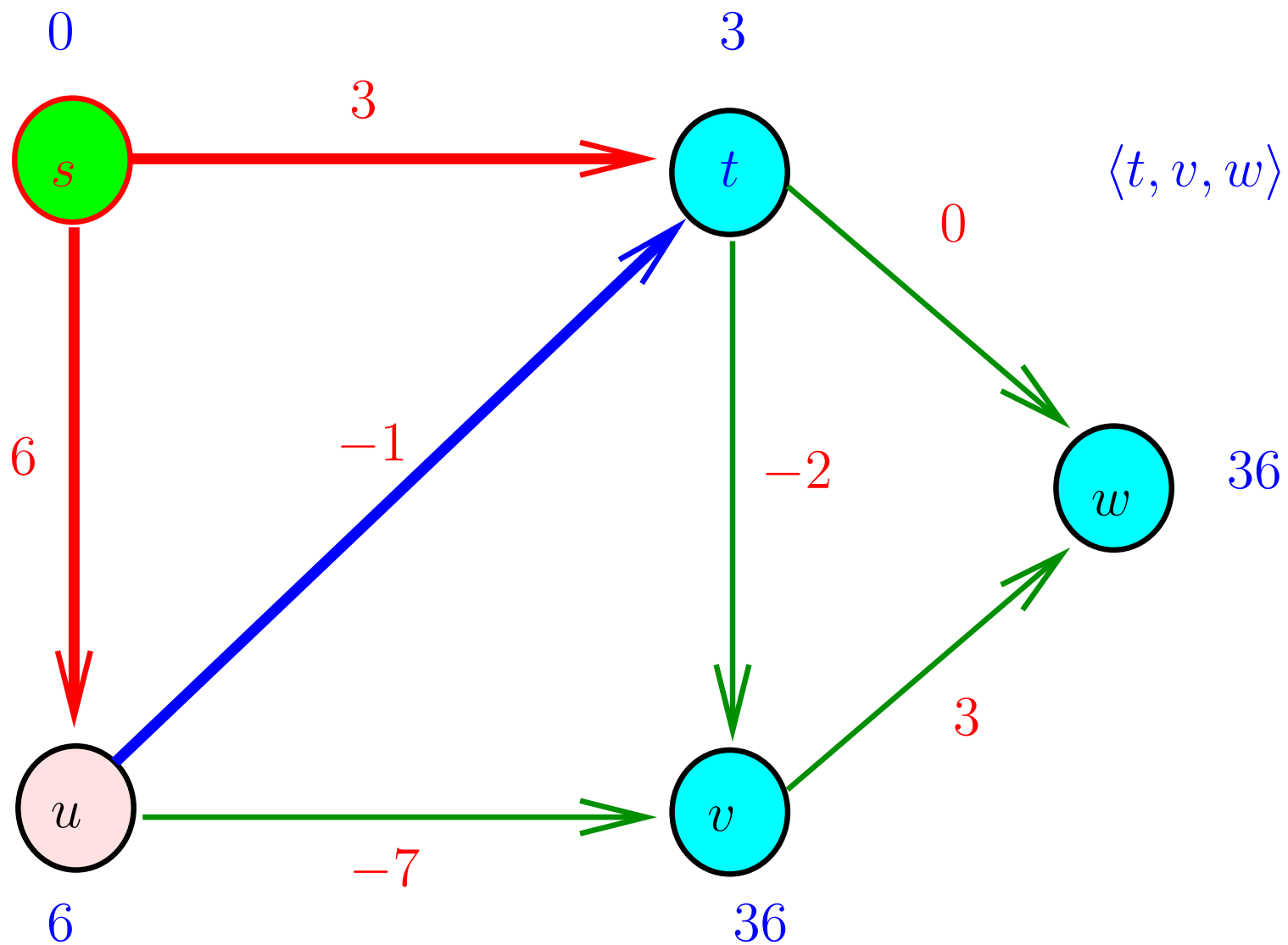
Simulação



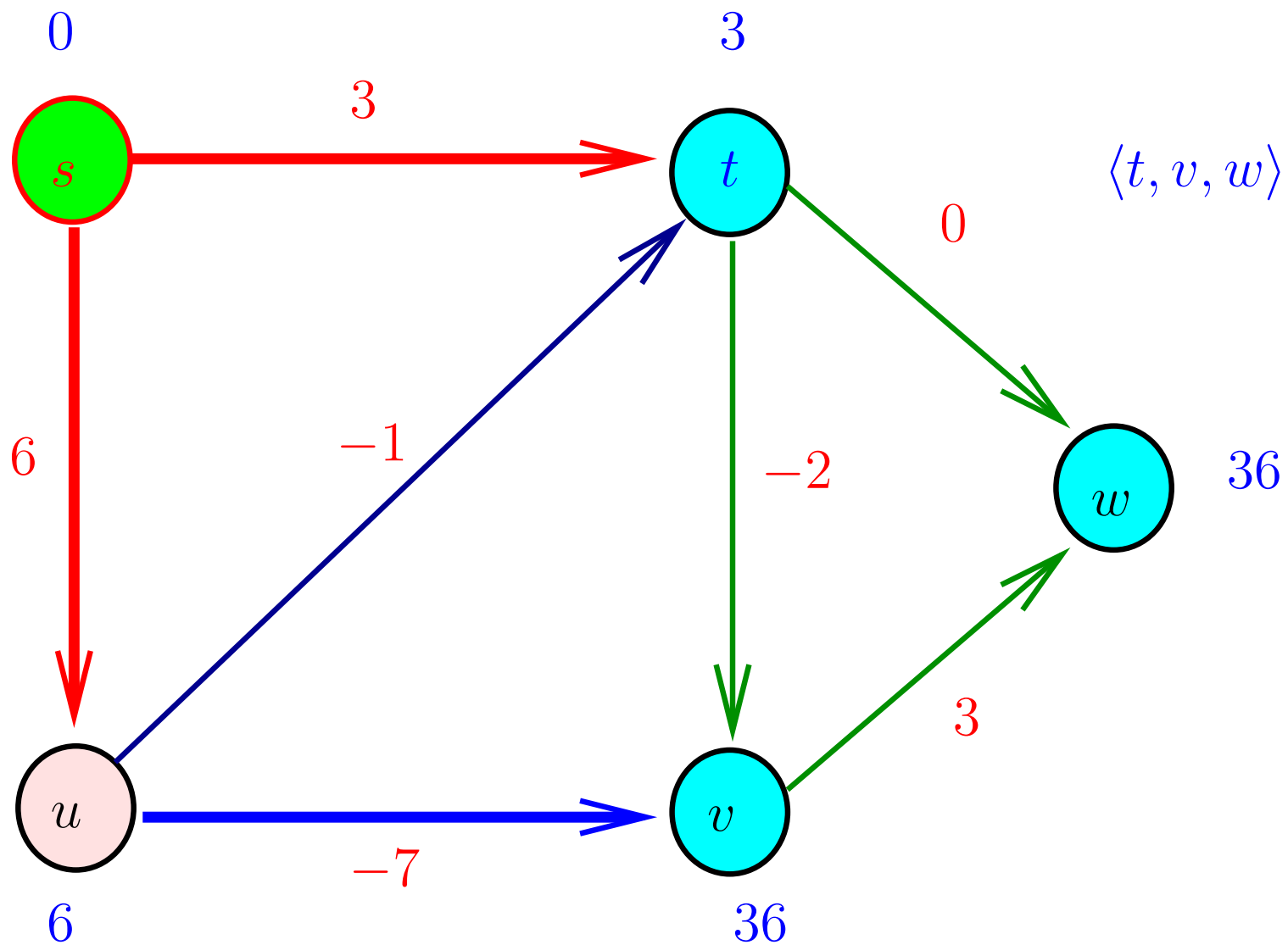
Simulação



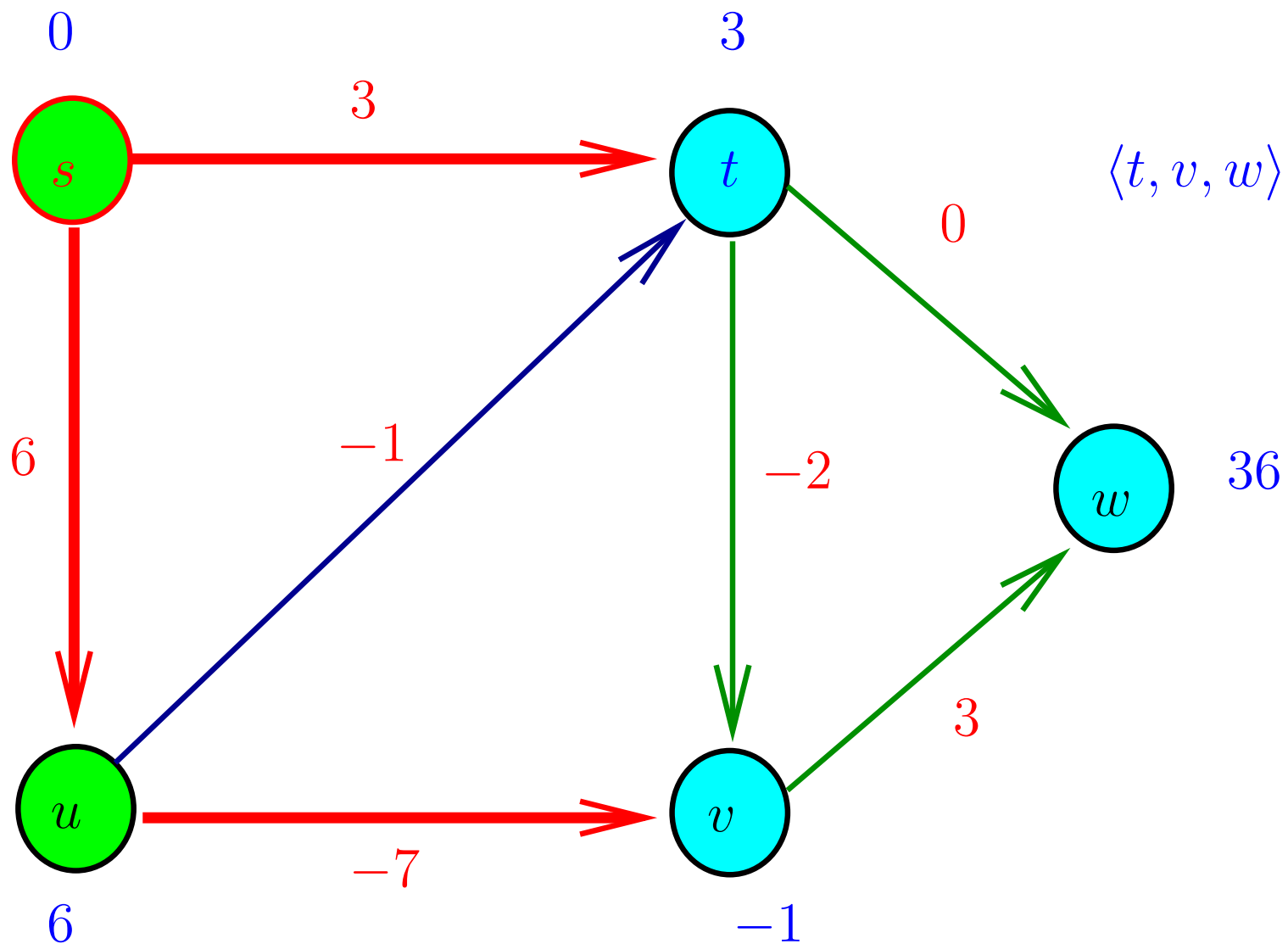
Simulação



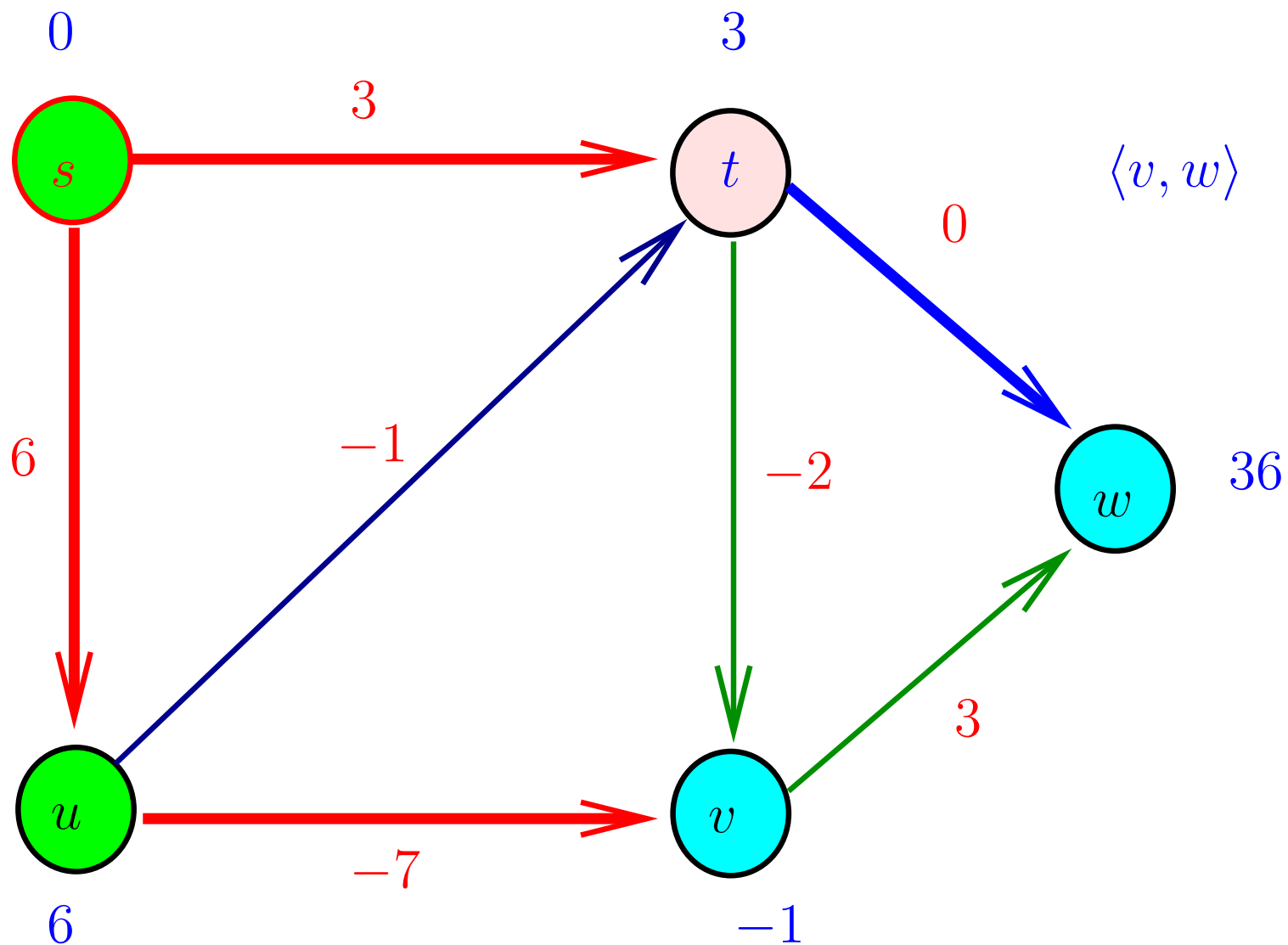
Simulação



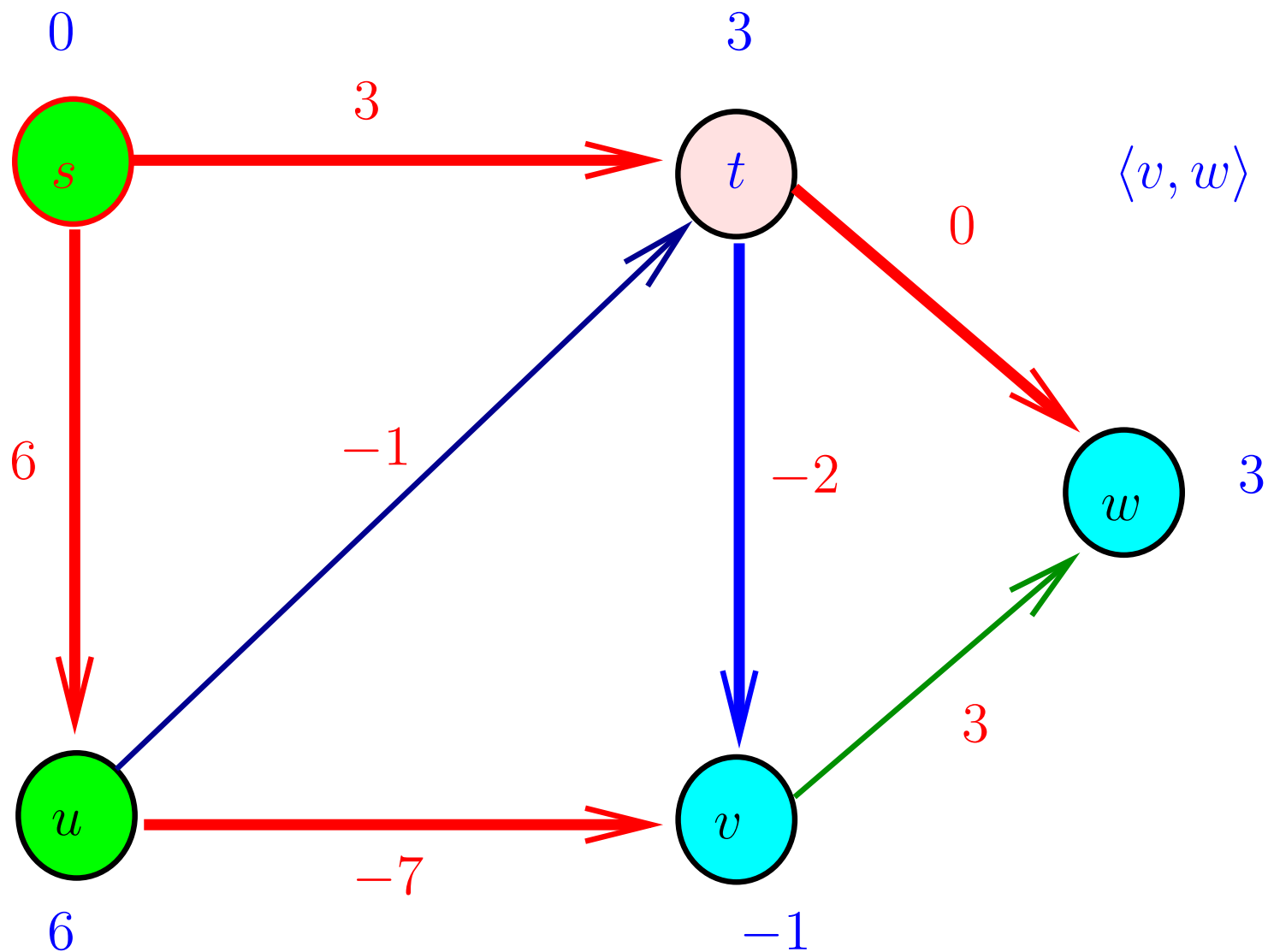
Simulação



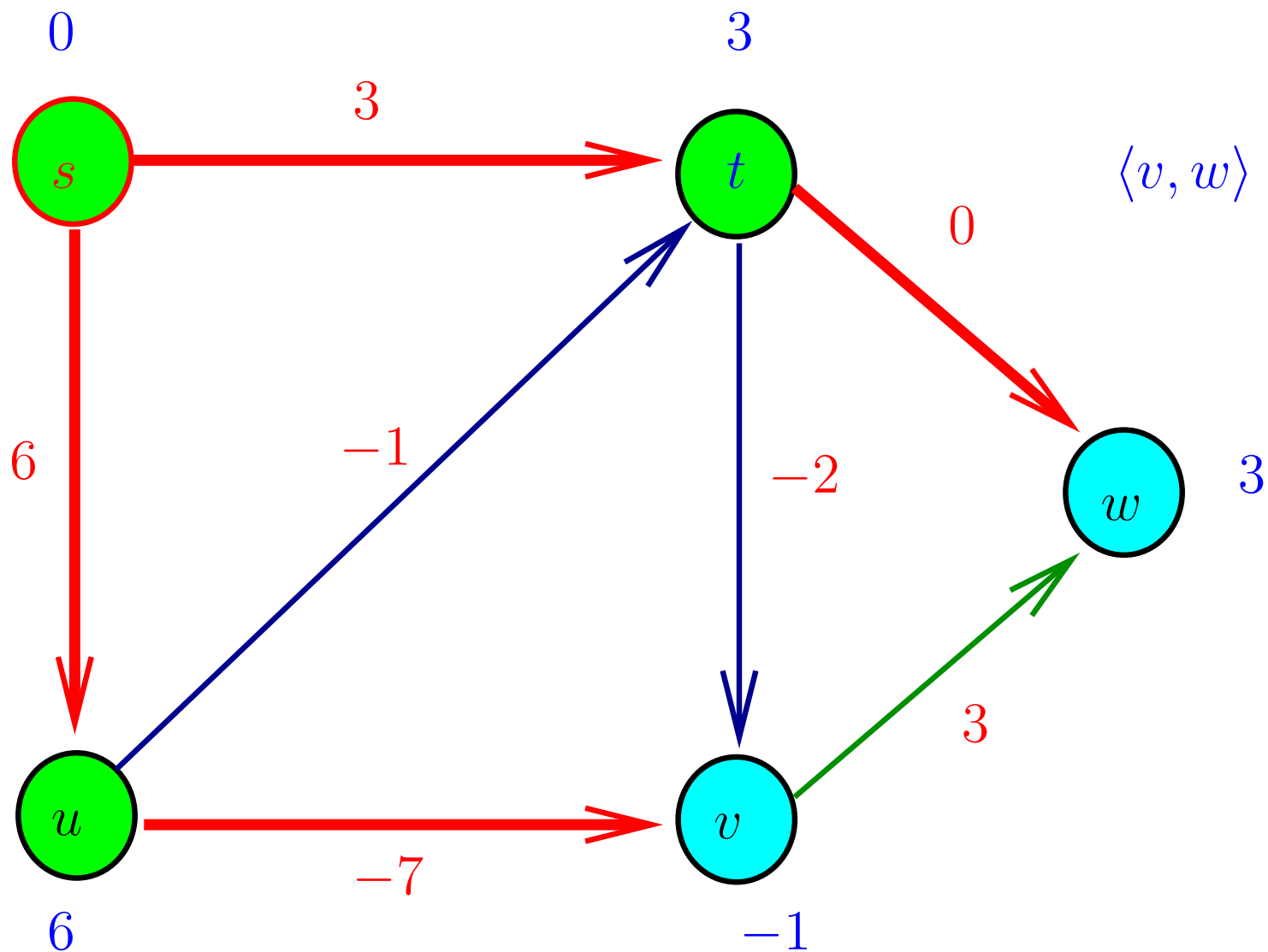
Simulação



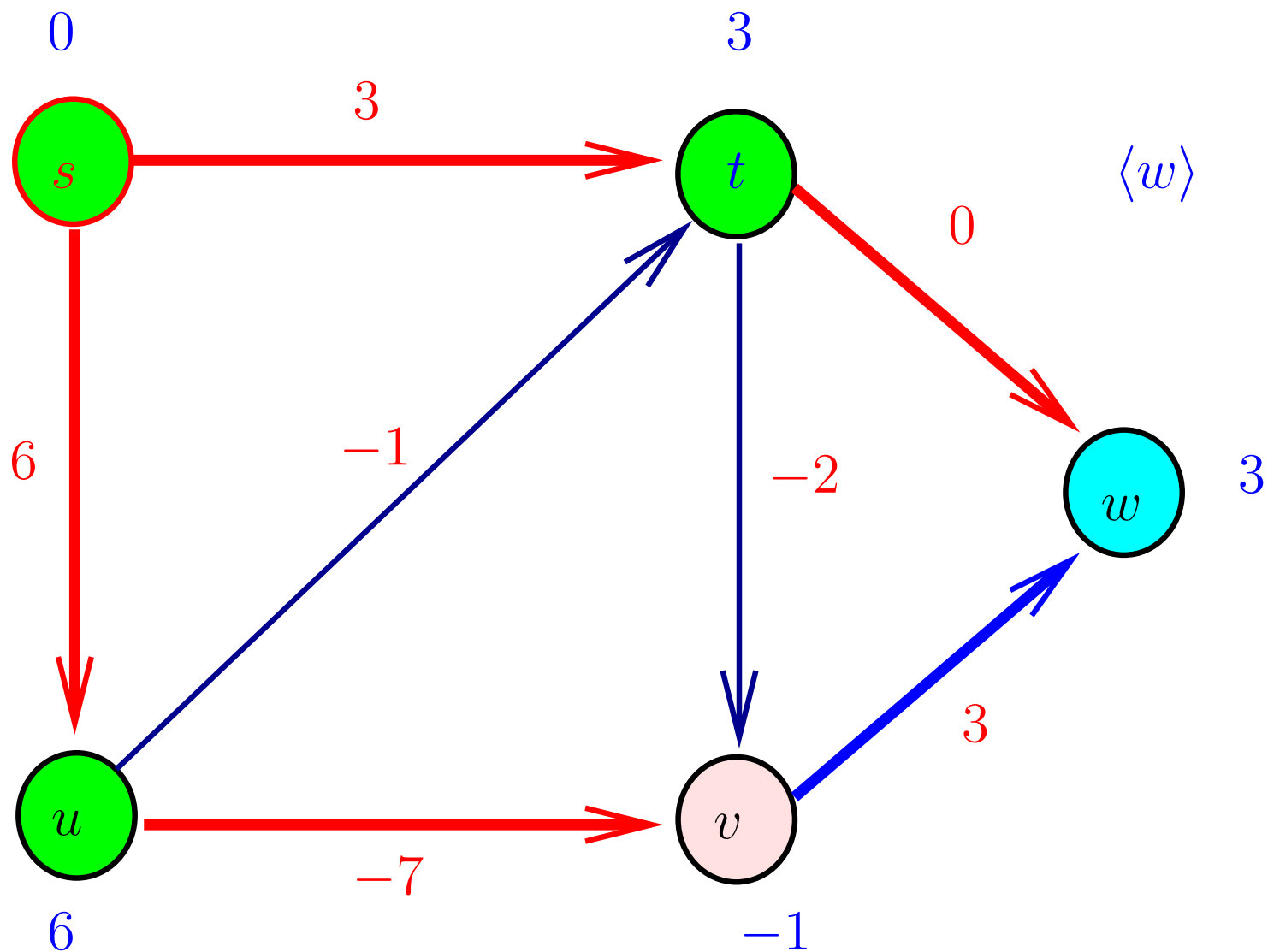
Simulação



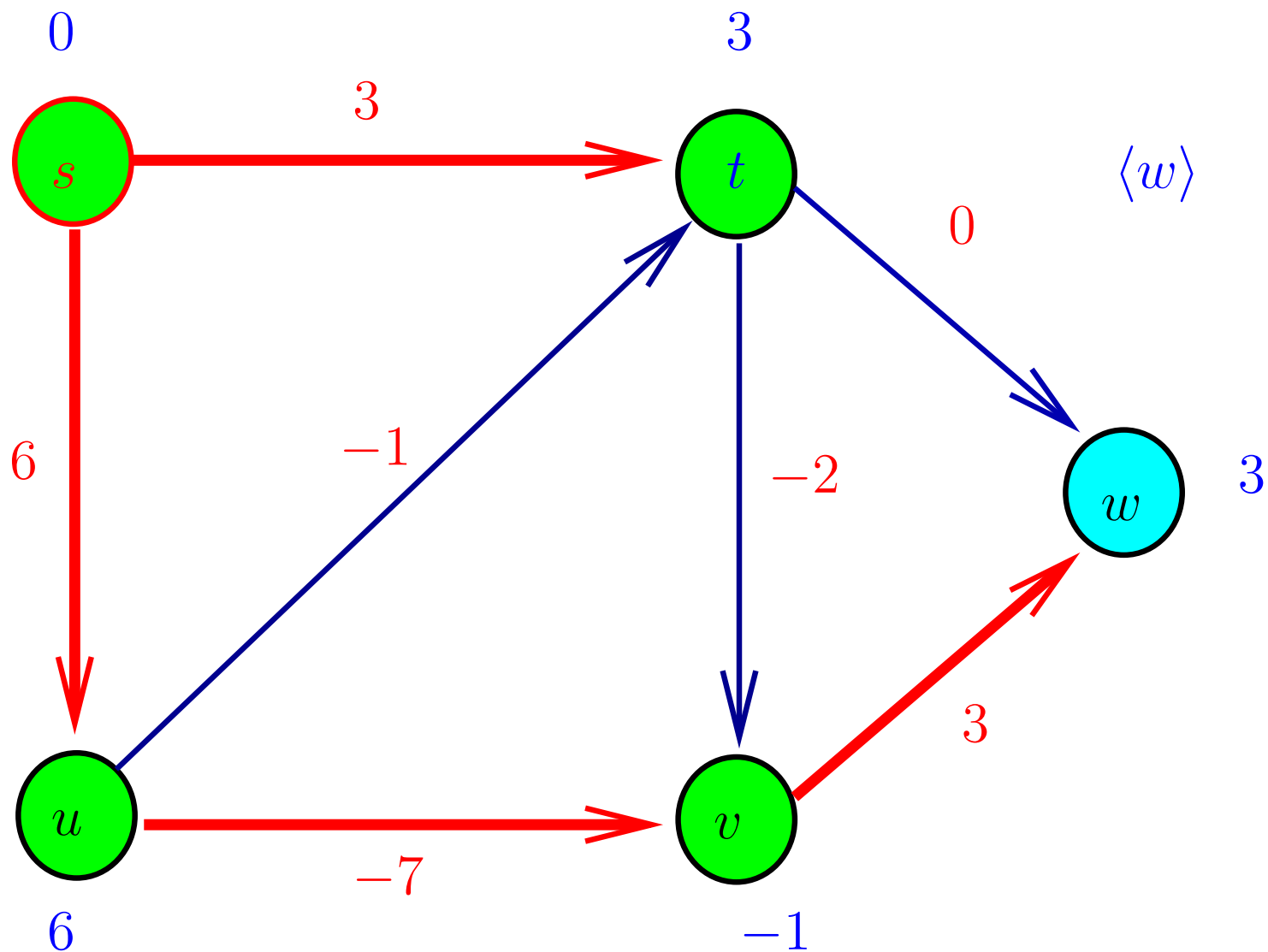
Simulação



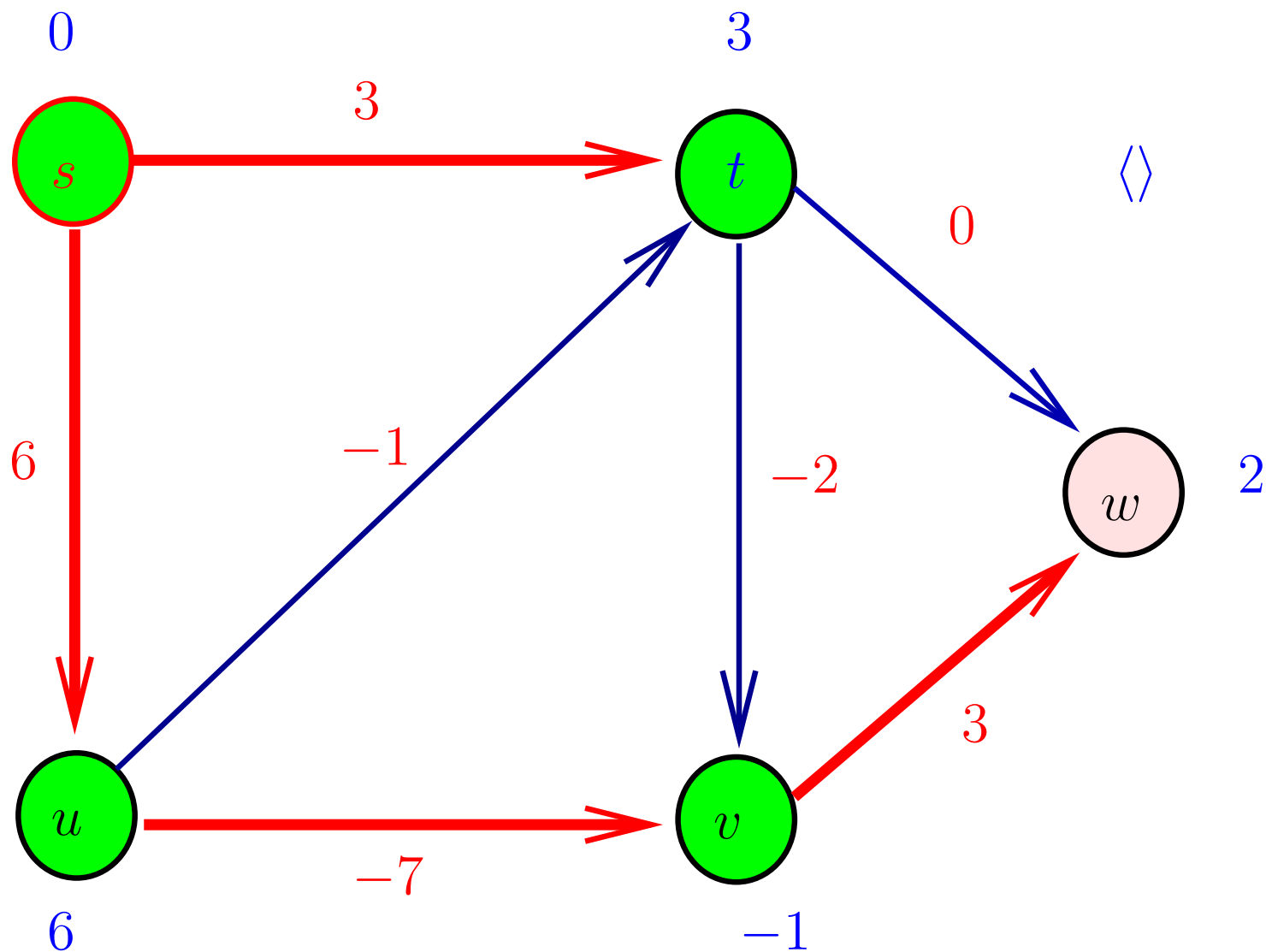
Simulação



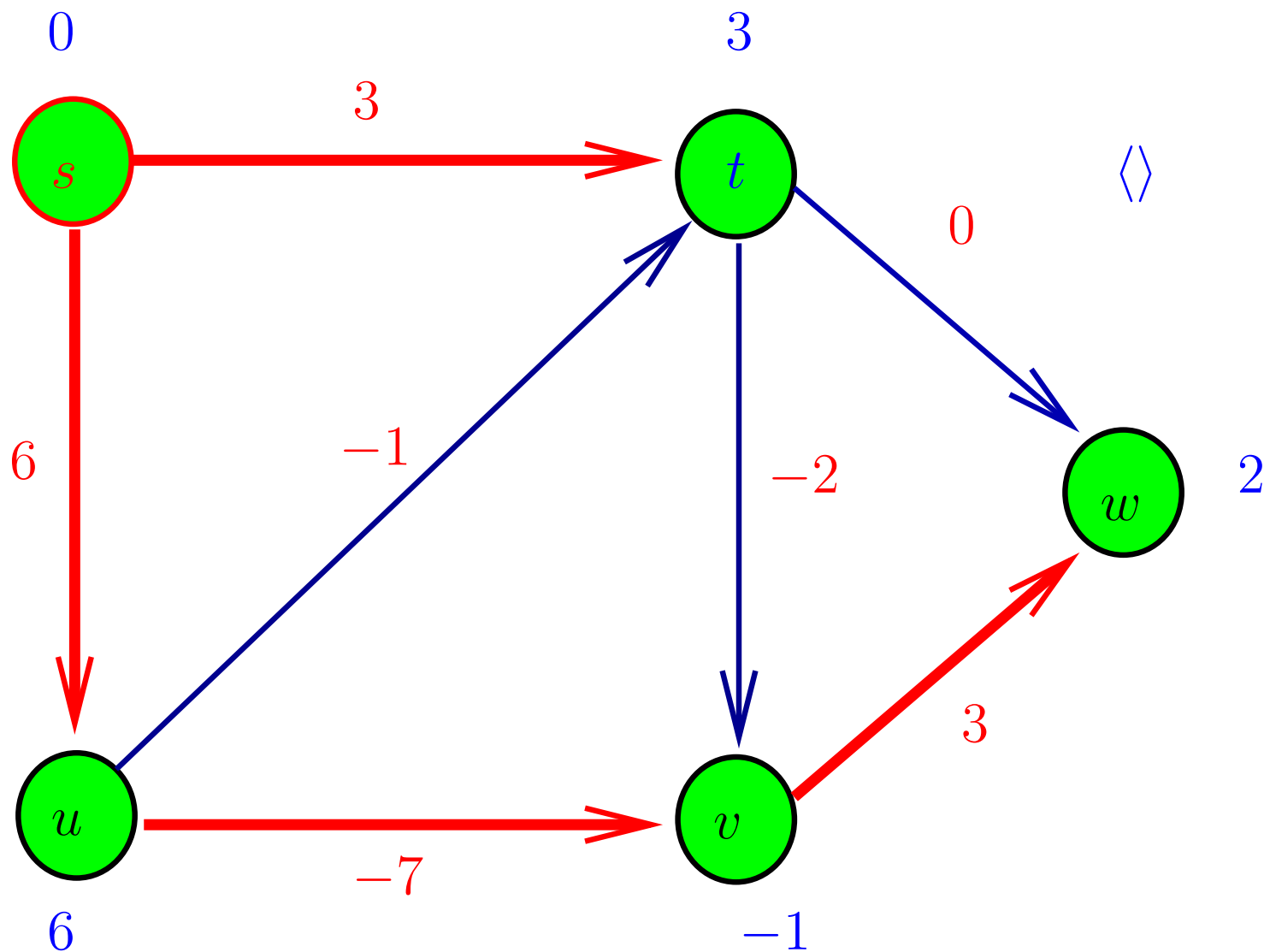
Simulação



Simulação



Simulação



Algoritmo genérico

Recebe uma rede acíclica (N, A, c) , uma ordem topológica dos nós e um índice o e devolve um c -potencia y e, para cada t um caminho P de v_o a t tal que $c(P) = y(t) - y(s)$.

MIN-COST-PATH-IN-DAG $(N, A, c, \langle v_1, \dots, v_n \rangle, o)$

```
1  para cada  $i$  em  $N$  faça
2       $y(i) \leftarrow \infty$ 
3       $\pi(i) \leftarrow \text{NIL}$ 
4   $y(v_o) \leftarrow 0$ 
5  para  $h \leftarrow o$  até  $n$  faça
6      para cada arco  $ij$  em  $A(v_h)$  faça
7          se  $y(j) > y(i) + c(ij)$ 
8              então  $y(j) \leftarrow y(i) + c(ij)$ 
9                   $\pi(j) \leftarrow i$ 
10 devolva  $\pi$  e  $y$ 
```

Consumo de tempo

O consumo de tempo do algoritmo
MIN-COST-PATH-IN-DAG é $O(n + m)$.