

Algoritmo de Bellman-Ford

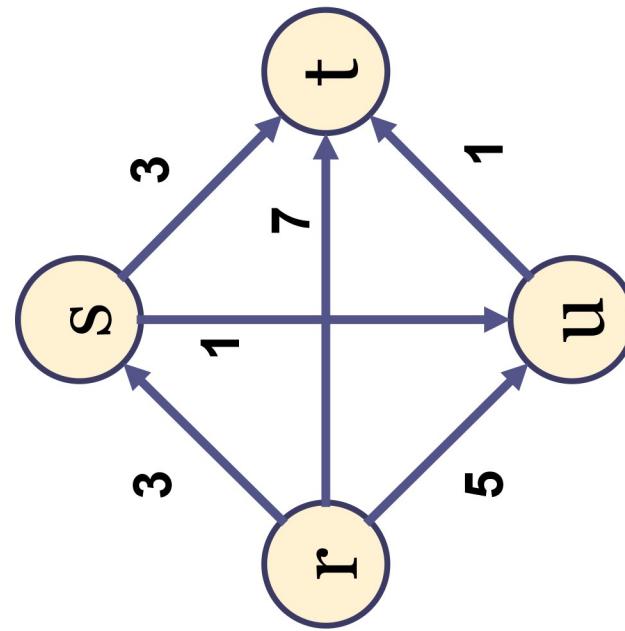


Algoritmo de Bellman-Ford

- Resolve o problema de **caminhos** mais curtos de uma única origem;
- São permitidas arestas com peso negativo (resolve o caso mais geral);
- Retorna verdadeiro se não existe ciclo negativo, e falso em caso contrário;

Algoritmo de Bellman-Ford

- Vamos encontrar os caminhos mais curtos de r para todos os outros vértices do seguinte grafo:



Algoritmo de Bellman-Ford

INICIALIZA($G = (V, A), s$)

para cada $v \in V$

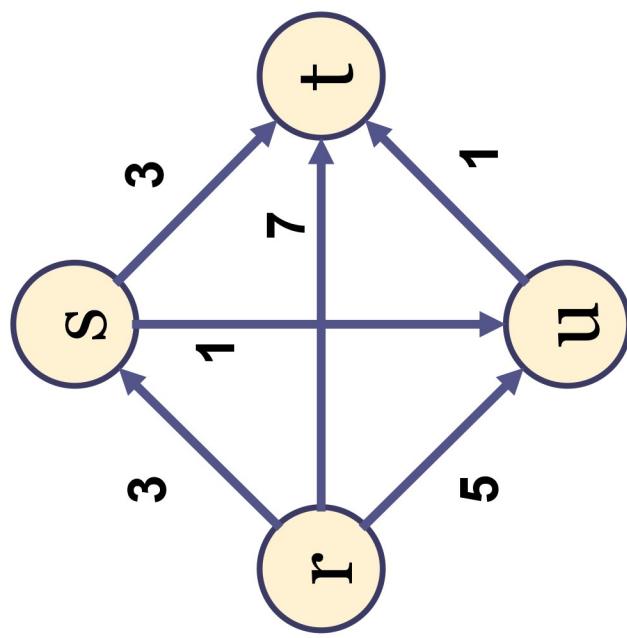
$d[v] = \infty$

$\pi[v] = \text{NULL}$

fim para

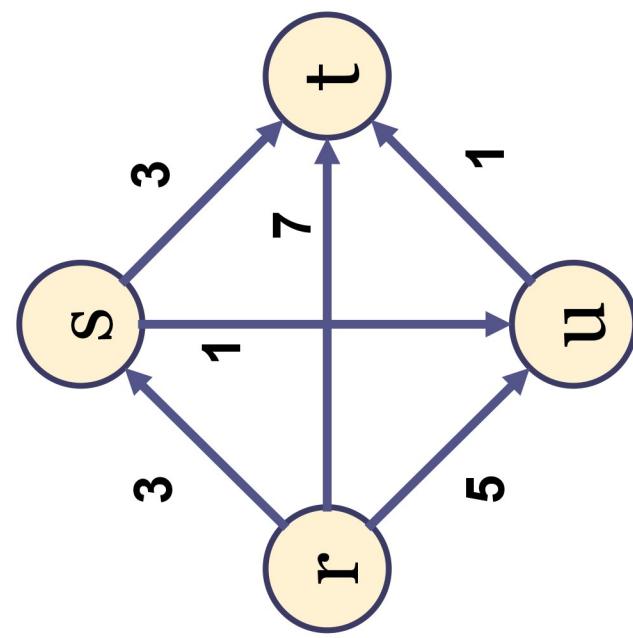
$d[s] = 0$

fim



vértice	r	s	t	u
d	0	∞	∞	∞
π	NULL	NULL	NULL	NULL

Algoritmo de Bellman-Ford



BELLMAN – FORD($G = (V, A)$, w, s)

INICIALIZA(G, s)

para $i \leftarrow 1$ *até* $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA (u, v, w)

fi *m para*

para cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

retorna falso

fi *m se*

fi *m para*

retorna verdadeiro

fi

variável	valor
i	1

vértice	r	s	t	u
d	0	∞	∞	∞
π	NULL	NULL	NULL	NULL

Algoritmo de Bellman-Ford

BELLMAN – FORD($G = (V, A), w, s$)

INICIALIZA(G, s)

para $i \leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA(u, v, w)

fim para

fim para

para cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

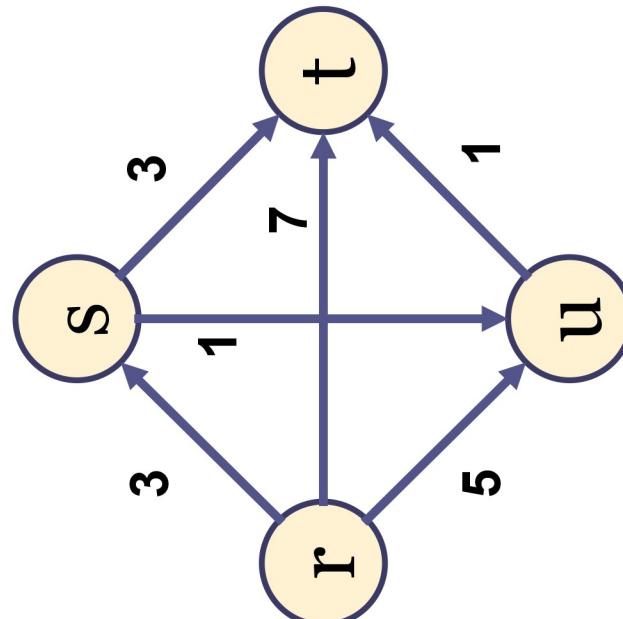
retorna falso

fim se

fim para

retorna verdadeiro

fim



variável	valor
i	1

vértice	r	s	t	u
d	0	∞	∞	∞
π	NULL	NULL	NULL	NULL

Algoritmo de Bellman-Ford

(s, t)

BELLMAN – FORD($G = (V, A), w, s$)

INICIALIZA(G, s)

para $i \leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA(u, v, w)

fim para

fim para

para cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

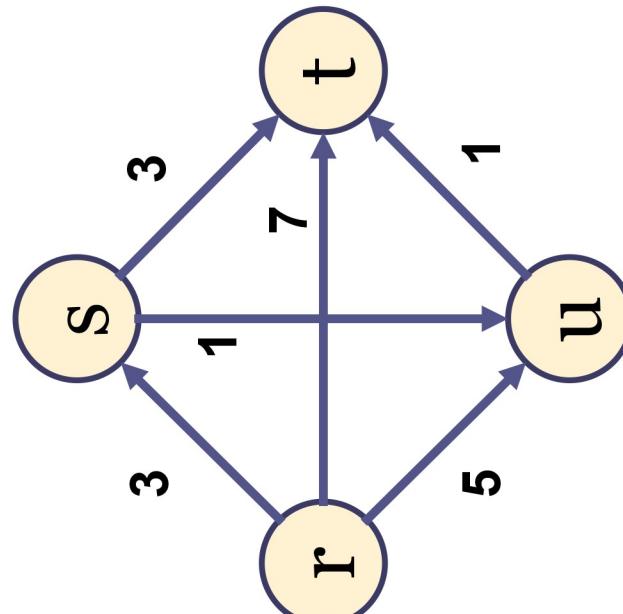
retorna falso

fim se

fim para

retorna verdadeiro

fim



variável	valor
i	1

vértice	r	s	t	u
d	0	∞	∞	∞
π	NULL	NULL	NULL	NULL

Algoritmo de Bellman-Ford

BELLMAN – FORD($G = (V, A), w, s$)

INICIALIZA(G, s)

para $i \leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA(u, v, w)

fim para

fim para

para cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

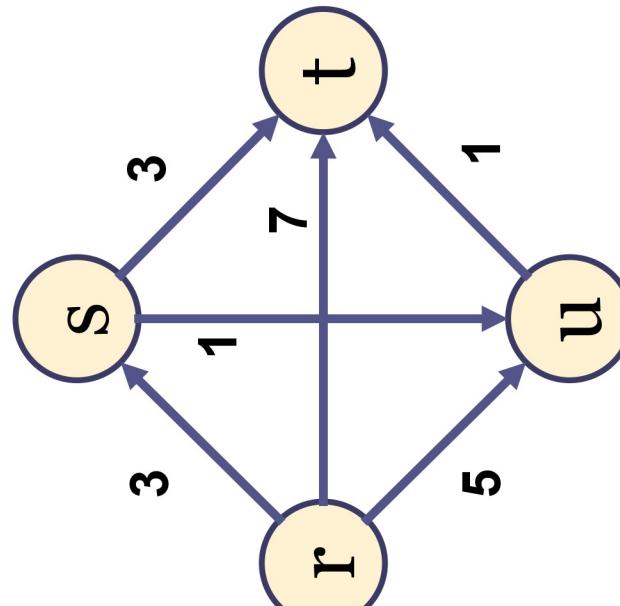
retorna falso

fim se

fim para

retorna verdadeiro

fim



(s, t)

(s, u)

(u, t)

(r, t)

(r, u)

variável	valor
i	1

vértice	r	s	t	u
d	0	∞	∞	∞
π	NULL	NULL	NULL	NULL

Algoritmo de Bellman-Ford

BELLMAN – FORD($G = (V, A), w, s$)

INICIALIZA(G, s)

para $i \leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA(u, v, w)

fim para

fim para

para cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

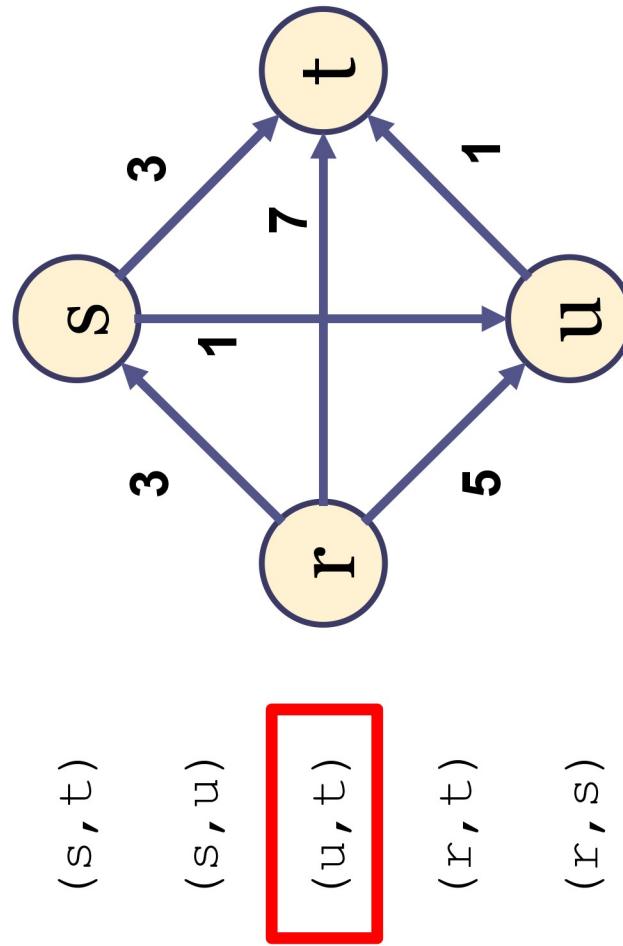
retorna falso

fim se

fim para

retorna verdadeiro

fim



variável	valor
i	1

vértice	r	s	t	u
d	0	∞	∞	∞
π	NULL	NULL	NULL	NULL

Algoritmo de Bellman-Ford

BELLMAN – FORD($G = (V, A), w, s$)

INICIALIZA(G, s)

para $i \leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA(u, v, w)

fim para

fim para

para cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

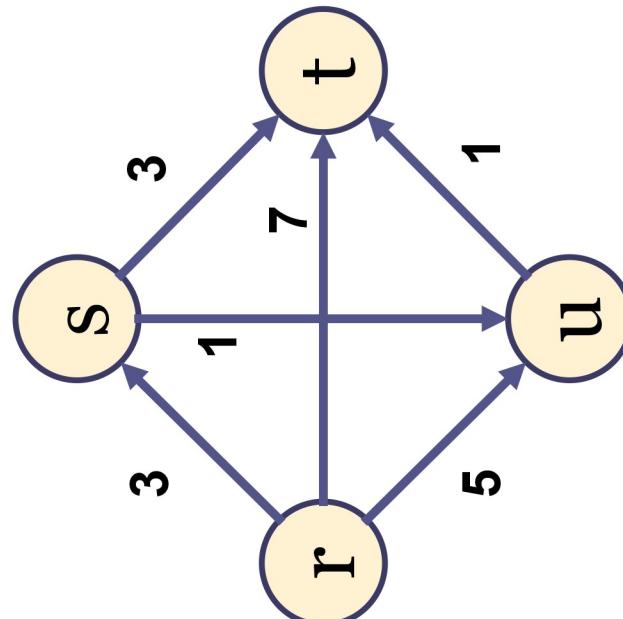
retorna falso

fim se

fim para

retorna verdadeiro

fim



(r, t)

(r, s)

(r, u)

variável	valor
<i>i</i>	1

vértice	r	s	t	u
d	0	∞	7	∞
π	NULL	NULL	r	NULL

Algoritmo de Bellman-Ford

BELLMAN – FORD($G = (V, A), w, s$)

INICIALIZA(G, s)

para $i \leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA(u, v, w)

fim para

fim para
para cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

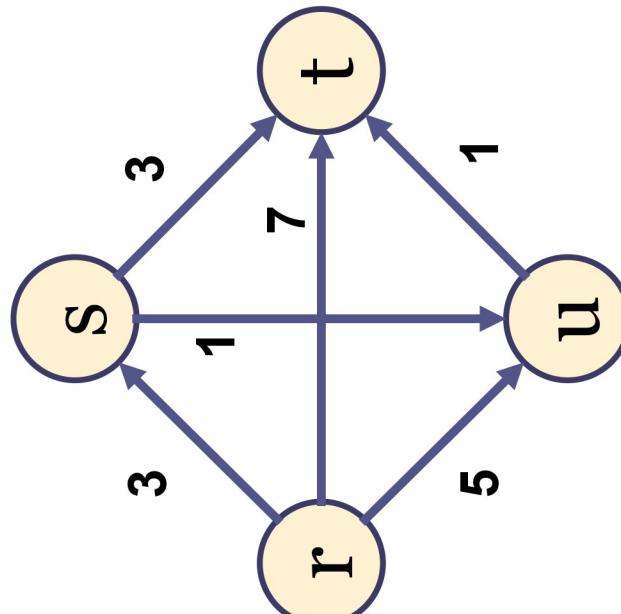
retorna falso

fim se

fim para

retorna verdadeiro

fim



(r, s)

variável	valor
<i>i</i>	1

vértice	r	s	t	u
d	0	3	7	∞
π	NULL	r	r	NULL

Algoritmo de Bellman-Ford

BELLMAN – FORD($G = (V, A), w, s$)

INICIALIZA(G, s)

para $i \leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA(u, v, w)

fim para

fim para

para cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

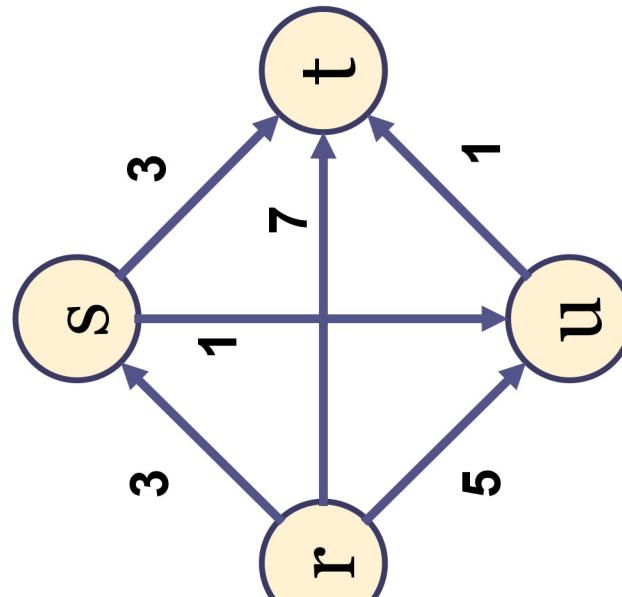
retorna falso

fim se

fim para

retorna verdadeiro

fim



variável	valor
i	1

(r, u)

vértice	r	s	t	u
d	0	3	7	5
π	NULL	r	r	r

Algoritmo de Bellman-Ford

BELLMAN – FORD($G = (V, A), w, s$)

INICIALIZA(G, s)

para $i \leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA(u, v, w)

fim para

fim para

para cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

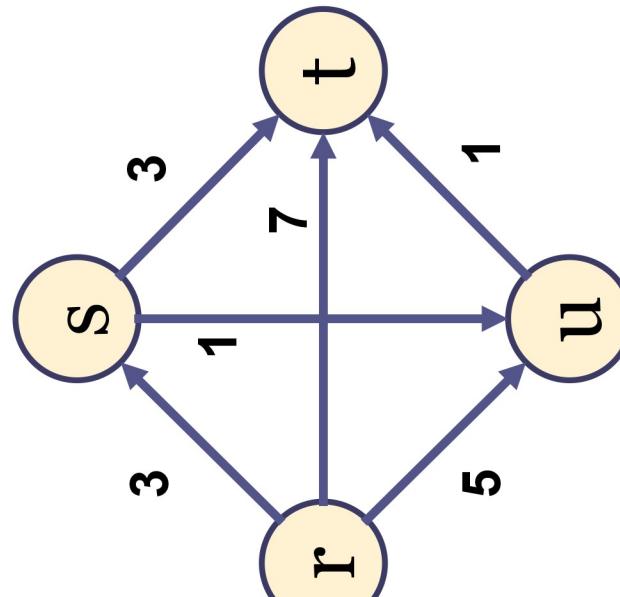
retorna falso

fim se

fim para

retorna verdadeiro

fim



variável	valor
i	2

vértice	r	s	t	u
d	0	3	7	5
π	NULL	r	r	r

Algoritmo de Bellman-Ford

(s, t)

BELLMAN - FORD($G = (V, A), w, s$)

INICIALIZA(G, s)

para $i \leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA(u, v, w)

fim para

fim para

para cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

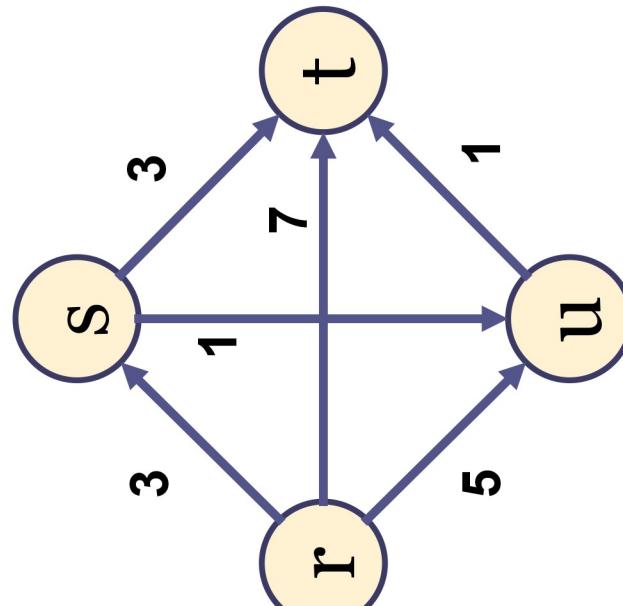
retorna falso

fim se

fim para

retorna verdadeiro

fim



variável	valor
i	2

vértice	r	s	t	u
d	0	3	6	5
π	NULL	r	s	r

Algoritmo de Bellman-Ford

BELLMAN – FORD($G = (V, A), w, s$)

INICIALIZA(G, s)

para $i \leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA(u, v, w)

fim para

fim para

para cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

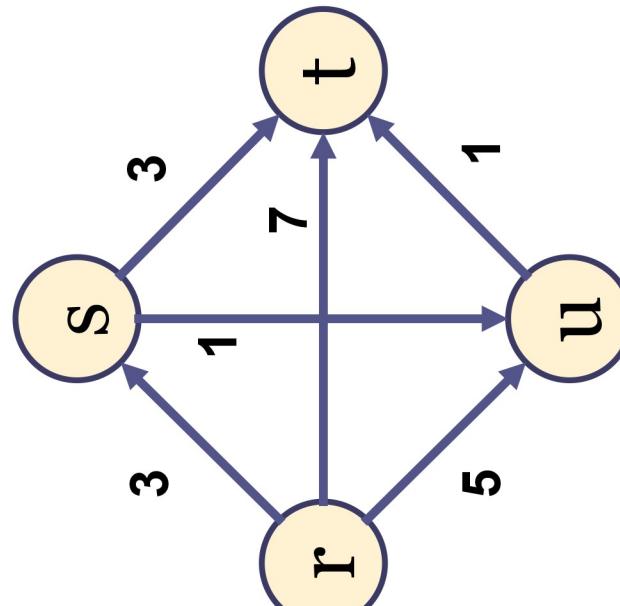
retorna falso

fim se

fim para

retorna verdadeiro

fim



(s, t)

$\boxed{(s, u)}$

(u, t)

(r, t)

(r, s)

(r, u)

variável	valor
i	2

vértice	r	s	t	u
d	0	3	6	4
π	NULL	r	s	s

Algoritmo de Bellman-Ford

BELLMAN – FORD($G = (V, A), w, s$)

INICIALIZA(G, s)

para $i \leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA(u, v, w)

fim para

par cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

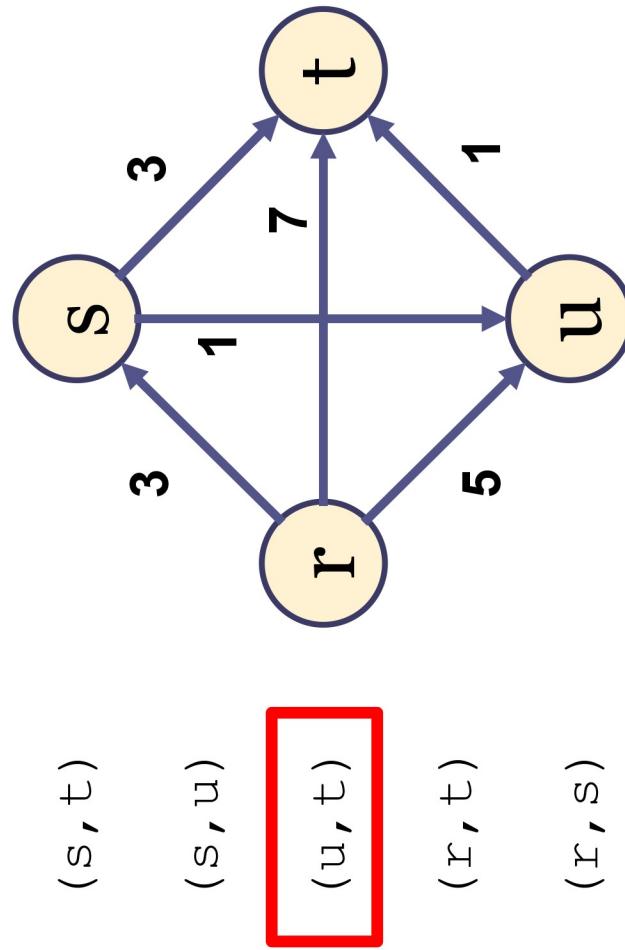
retorna falso

fim se

fim para

retorna verdadeiro

fim



variável	valor
i	2

vértice	r	s	t	u
d	0	3	5	4
π	NULL	r	u	s

Algoritmo de Bellman-Ford

BELLMAN – FORD($G = (V, A), w, s$)

INICIALIZA(G, s)

para $i \leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA(u, v, w)

fim para

fim para

para cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

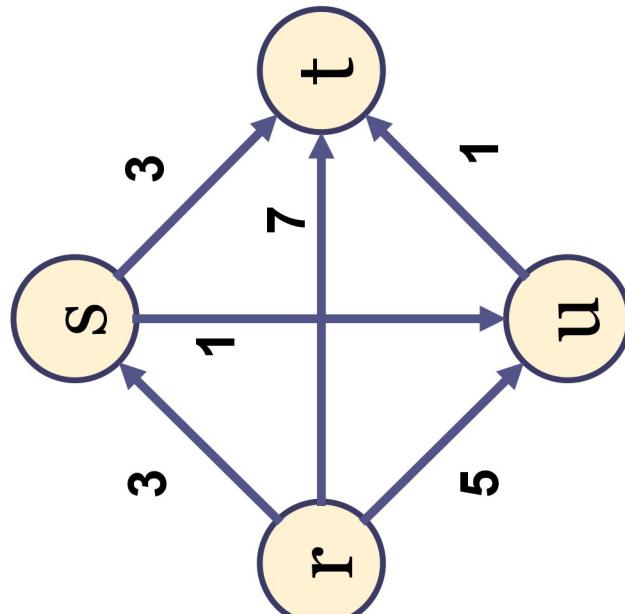
retorna falso

fim se

fim para

retorna verdadeiro

fim



(S, t)

(S, u)

(u, t)

(r, t)

(r, s)

(r, u)

variável	valor
i	2
r	0
π	NULL

Algoritmo de Bellman-Ford

BELLMAN – FORD($G = (V, A), w, s$)

INICIALIZA(G, s)

para $i \leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA(u, v, w)

fim para

fim para

para cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

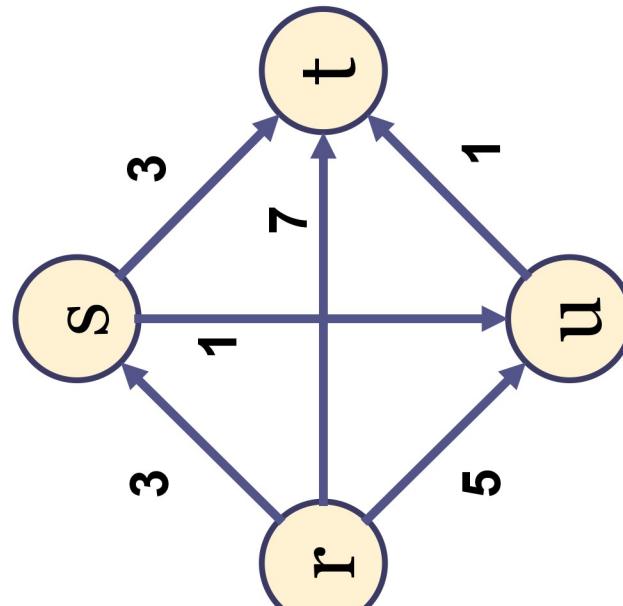
retorna falso

fim se

fim para

retorna verdadeiro

fim



(r, s)

(s, t)

(s, u)

(u, t)

(r, t)

variável	valor
<i>i</i>	2
vértice	r s t u
d	0 3 5 4
π	NULL r u s

Algoritmo de Bellman-Ford

BELLMAN – FORD($G = (V, A), w, s$)

INICIALIZA(G, s)

para $i \leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA(u, v, w)

fim para

fim para

para cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

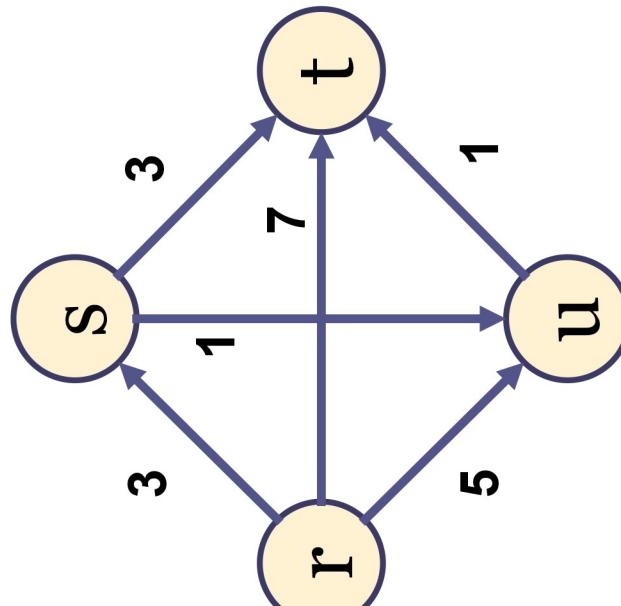
retorna falso

fim se

fim para

retorna verdadeiro

fim



(s, t)

(s, u)

(u, t)

(r, t)

(r, s)

(r, u)

variável	valor
i	2

vértice	r	s	t	u
d	0	3	5	4
π	NULL	r	u	s

Algoritmo de Bellman-Ford

BELLMAN – FORD($G = (V, A), w, s$)

INICIALIZA(G, s)

para i $\leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA(u, v, w)

fim para

fim para

para cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

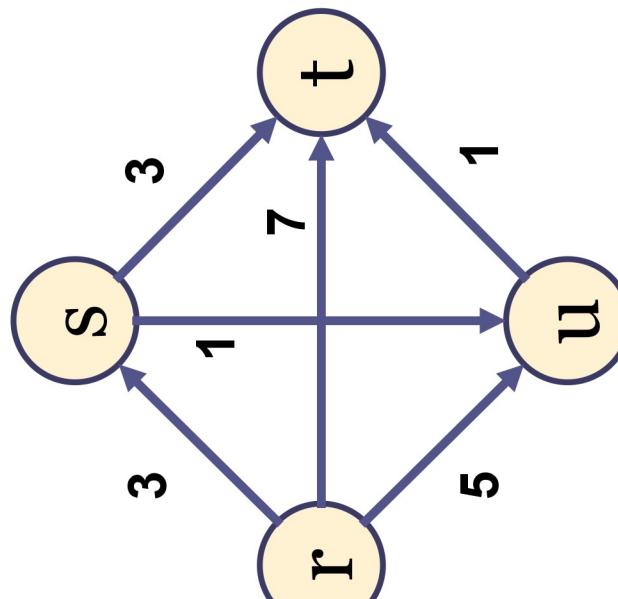
retorna falso

fim se

fim para

retorna verdadeiro

fim



(s, t)

(s, u)

(u, t)

(r, t)

(r, s)

(r, u)

variável	valor
i	3

vértice	r	s	t	u
d	0	3	5	4
π	NULL	r	u	s

Algoritmo de Bellman-Ford

(s, t)

BELLMAN - FORD($G = (V, A), w, s$)

INICIALIZA(G, s)

para $i \leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA(u, v, w)

fim para

fim para

para cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

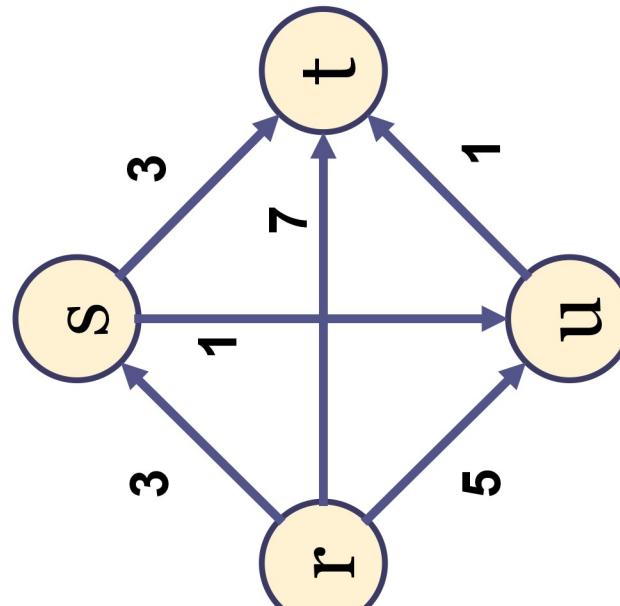
retorna falso

fim se

fim para

retorna verdadeiro

fim



variável	valor
i	3

vértice	r	s	t	u
d	0	3	5	4
π	NULL	r	u	s

Algoritmo de Bellman-Ford

$BELLMAN - FORD(G = (V, A), w, s)$

$INICIALIZA(G, s)$

para $i \leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

$\uparrow \uparrow$ $RELAXA(u, v, w)$

fin para

fin para

para cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

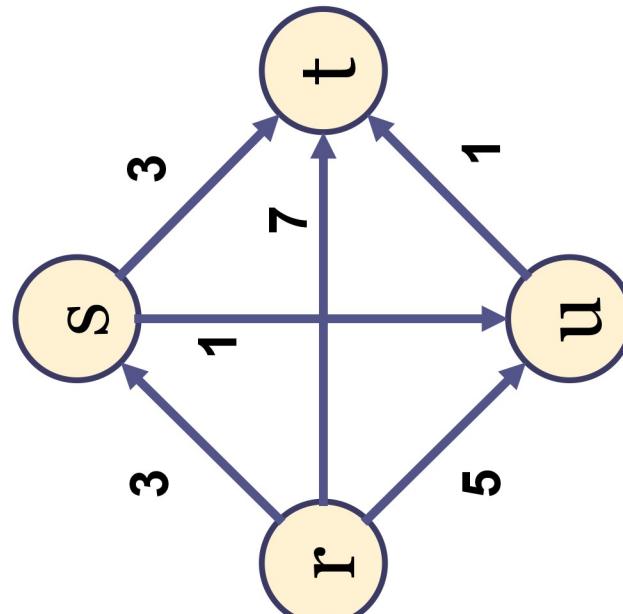
retorna falso

fin se

fin para

retorna verdadeiro

fin



(s, t)

(s, u)

(u, t)

(r, t)

(r, s)

(r, u)

variável	valor
i	3

vértice	r	s	t	u
d	0	3	5	4
π	NULL	r	u	s

Algoritmo de Bellman-Ford

BELLMAN – FORD($G = (V, A), w, s$)

INICIALIZA(G, s)

para $i \leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA(u, v, w)

fim para

fim para

para cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

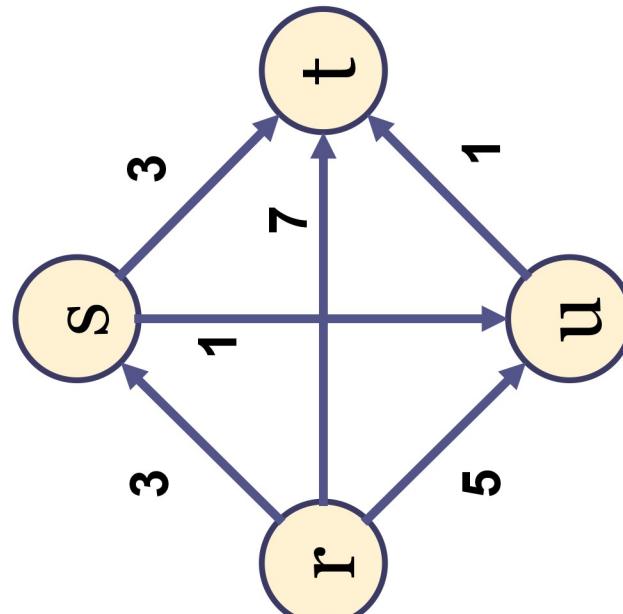
retorna falso

fim se

fim para

retorna verdadeiro

fim



(u, t)

(r, t)

(r, s)

(r, u)

vertice	r	s	t	u
d	0	3	5	4
π	NULL	r	u	s

Algoritmo de Bellman-Ford

BELLMAN – FORD($G = (V, A), w, s$)

INICIALIZA(G, s)

para $i \leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA(u, v, w)

fim para

fim para

para cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

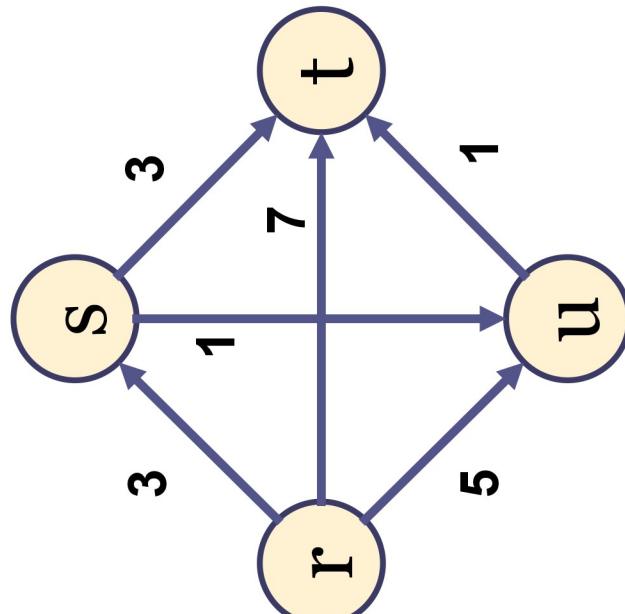
retorna falso

fim se

fim para

retorna verdadeiro

fim



(S, t)

(S, u)

(u, t)

(r, t)

(r, s)

(r, u)

variável	valor
i	3

vértice	r	s	t	u
d	0	3	5	4
π	NULL	r	u	s

Algoritmo de Bellman-Ford

BELLMAN – FORD($G = (V, A), w, s$)

INICIALIZA(G, s)

para $i \leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA(u, v, w)

fim para

par cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

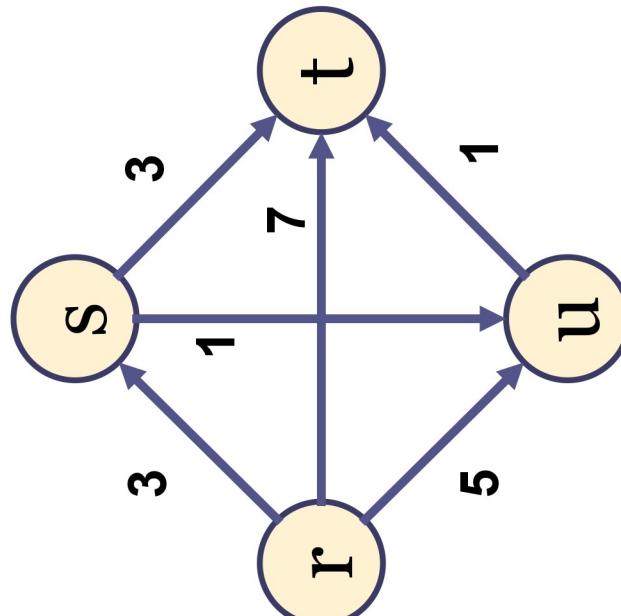
retorna falso

fim se

fim para

retorna verdadeiro

fim



(r, s)

variável	valor
<i>i</i>	3
vértice	r s t u
d	0 3 5 4
π	NULL r u s

Algoritmo de Bellman-Ford

BELLMAN – FORD($G = (V, A), w, s$)

INICIALIZA(G, s)

para $i \leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA(u, v, w)

fim para

fim para

para cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

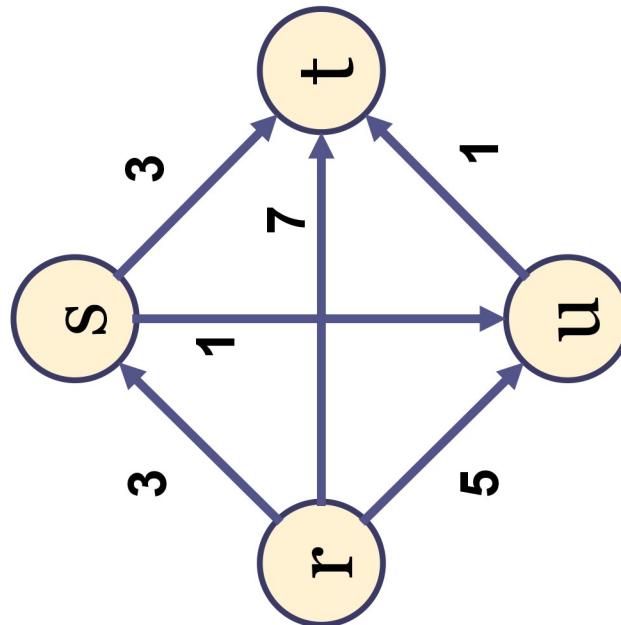
retorna falso

fim se

fim para

retorna verdadeiro

fim



variável	valor
i	3

(r, u)

vértice	r	s	t	u
d	0	3	5	4
π	NULL	r	u	s

Algoritmo de Bellman-Ford

BELLMAN – FORD($G = (V, A), w, s$)

INICIALIZA(G, s)

para $i \leftarrow 1$ até $|V| - 1$

para cada aresta $(u, v) \in A$

RELAXA(u, v, w)

fim para

fim para

para cada aresta $(u, v) \in A$

se $d[v] > d[u] + w(u, v)$

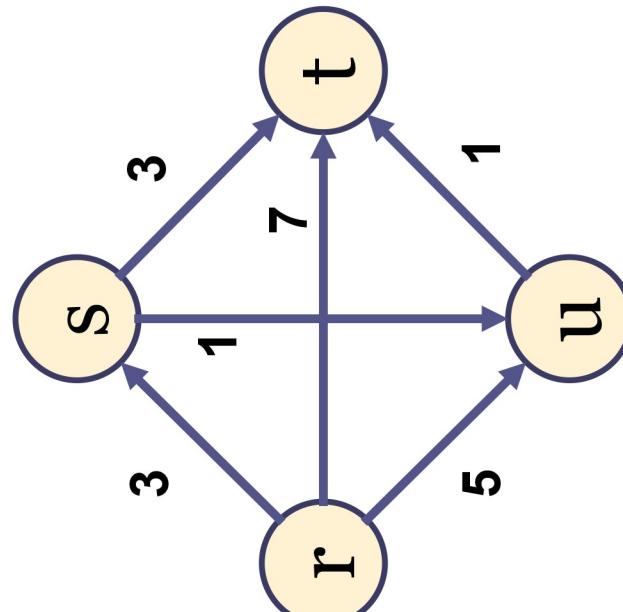
retorna falso

fim se

fim para

retorna verdadeiro

fim

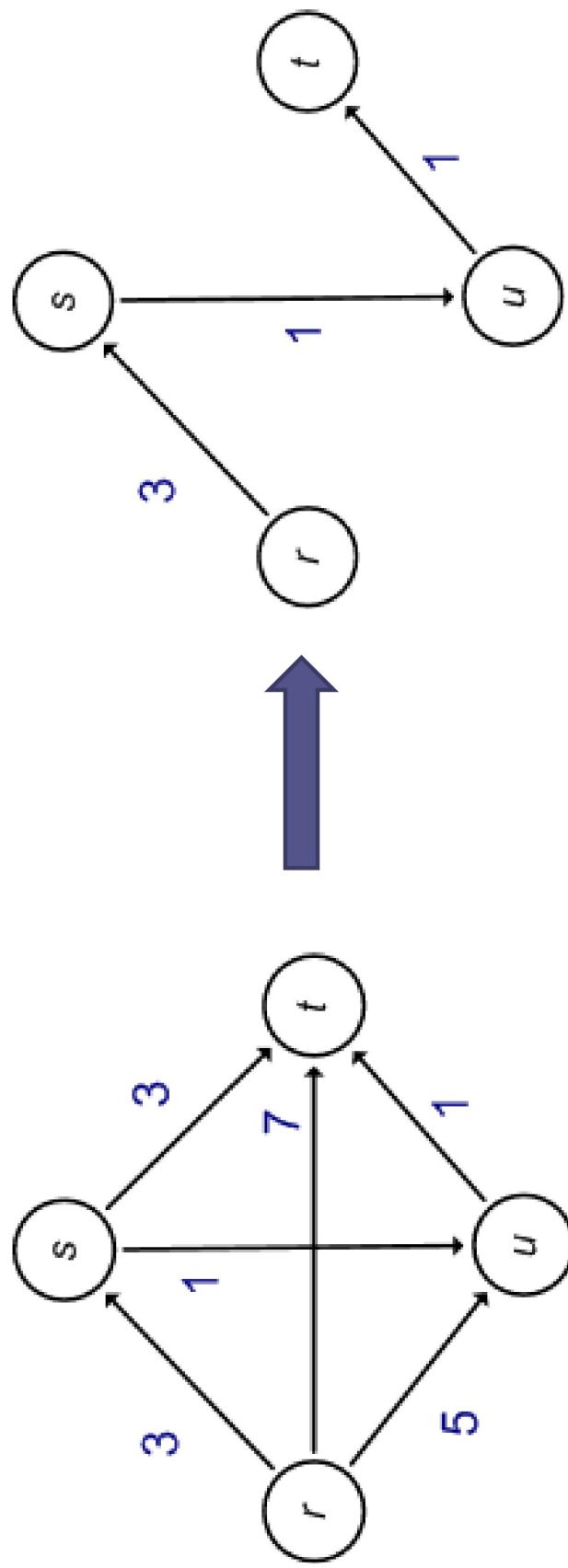


variável	valor
i	3

vértice	r	s	t	u
d	0	3	5	4
π	NULL	r	u	s

Algoritmo de Bellman-Ford

- Solução do exemplo...



vértice	r	s	t	u
d	0	3	5	4
π	NULL	r	u	s

Bellman-Ford

- **Aplicação:**

- Uma variação distribuída deste algoritmo é implementada na área de redes de computadores;
- O algoritmo de roteamento chamado **vetor de distâncias**;
- Seu objetivo é fornecer informação para cada *host* que deseja enviar pacotes (criar tabelas de encaminhamento);
- Por qual saída ele deve enviar um pacote de modo que o mesmo chegue rapidamente ao destinatário;

Bellman-Ford

- Aplicação:
 - Na área de redes de computadores devemos **tomar cuidado com os “caminhos mais curtos”**, ou os “caminhos de maior vazão”;
 - Podemos **congestionar** a toda a rede se, na execução do Algoritmo do Vettor de Distâncias, um **host se tornar o “gargalo” da rede.**
 - Geralmente este problema é “resolvido” no protocolo TCP;