06 – Grafos: Caminhos Mínimos

SCC0503 – Algoritmos e Estruturas de Dados II

Paulo H. R. Gabriel Moacir Ponti Jr.

www.icmc.usp.br/~moacir

Instituto de Ciências Matemáticas e de Computação – USP

2011/1

Conteúdo da Aula

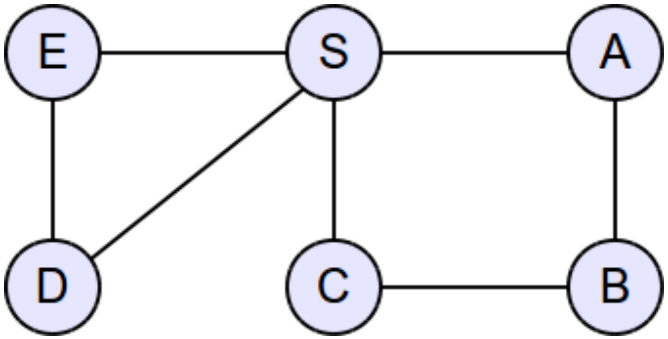
Resumindo e recordando...

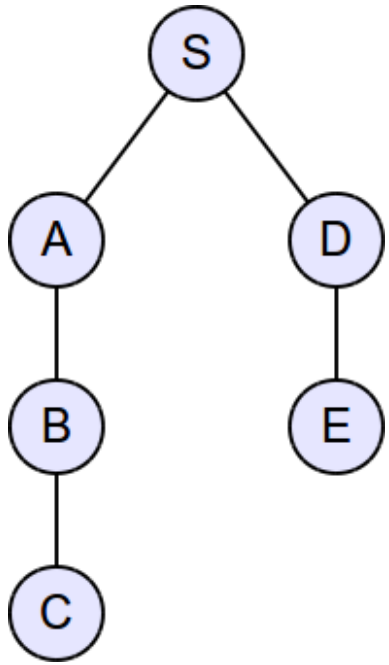
Seja $G = (V, E)$ um (di)grafo

Percurso em G

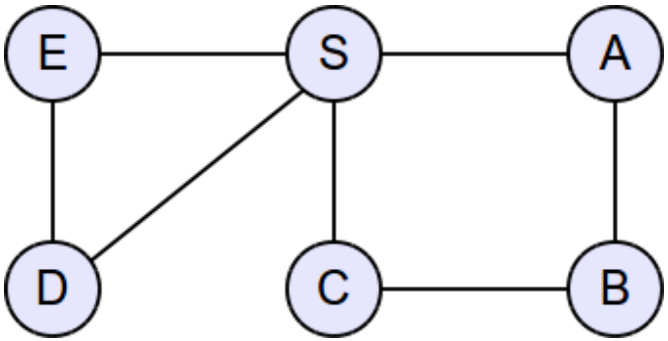
- Busca em profundidade (DFS)
- Busca em largura (BFS)

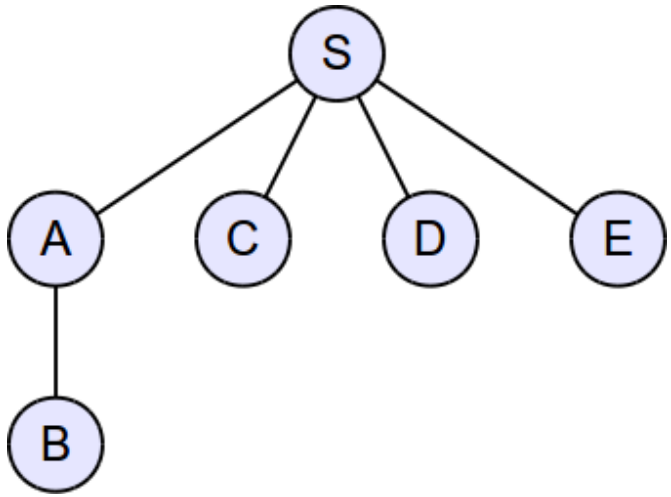
Resumindo e recordando: Um exemplo de percuso (DFS)





Resumindo e recordando: Um exemplo de percurso (BFS)





Resumindo e recordando: BFS

- Algoritmo que encontra o menor caminho a partir de uma determinada fonte até todos os outros vértices

Menor Caminho

Caminho de menor comprimento (menor número de arestas)

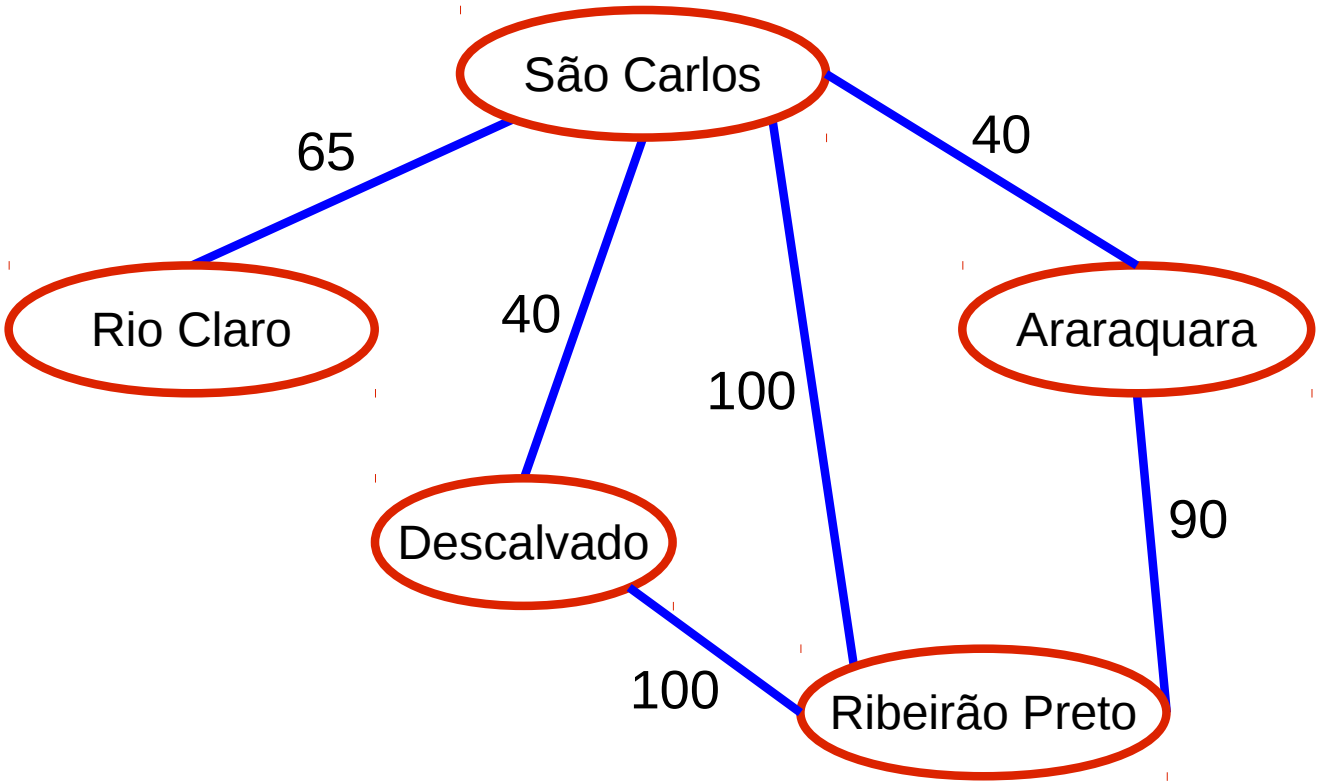
- BFS processa vértices em ordem crescente de distância em relação ao vértice raiz

Tempo de execução

$\mathcal{O}(|V| + |E|)$

Uma aplicação de grafos bem comum

Mapas



Uma aplicação de grafos bem comum

Problema 1

Como saber se duas cidades estão conectadas?

Problema 2

Qual o melhor caminho entre duas cidades?

Em outras palavras. . .

Função Peso

Seja $G = (V, E)$ um (di)grafo ponderado com função peso $w : E \to \Re$. O peso de uma aresta $u \to v$ é denotado por $w(u, v)$
Para $e \in E$, se $e = (u, v)$, denotamos $w(u, v) = w(e)$

Comprimento de um Caminho

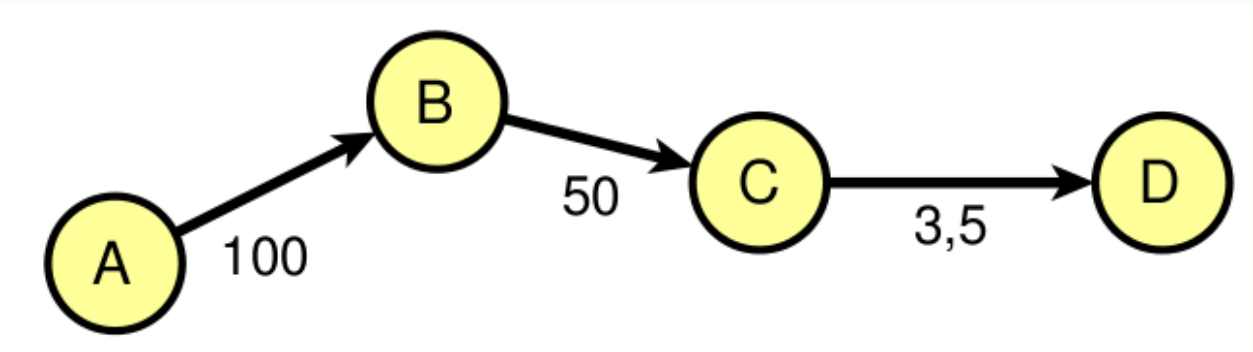
- Seja $p = v_0 \to v_1 \to \ldots \to v_k$ um caminho
- O comprimento é a soma dos pesos das arestas de p
- Matematicamente:

$$L(p) = \sum_{i=1}^{k} w(v_{i-1}, v_i)$$

Em outras palavras. . .

$$L(p) = \sum_{i=1}^{k} w(v_{i-1}, v_i)$$

Exemplo



$$L(p) = 153,5$$

Caminhos mais curto

Mais uma definição

Um caminho mais curto (*shortest path*) de u até v é o caminho de menor comprimento entre u e v

Notação

$\delta(u, v) = \min\{L(p)\}$, sendo p um caminho entre u e v

Observação

$\delta(u, v) = \infty$ se não houver caminho entre u e v

O problema dos caminhos mínimos

Single-source shortest paths

Dado um (di)grafo ponderado, com pesos não-negativos, e um vértice $s \in V$, encontre $\delta(s, v)$, para todo $v \in V$

O problema dos caminhos mínimos – Ideia de solução

Abordagem Gulosa

1. Crie um conjunto S de vértices cujas distâncias a partir de s são conhecidas
2. A cada iteração, acrescente a S o vértice v cuja distância estimada a s é mínima
   - $v \in (V - S)$ (ou seja, não pertence ainda a S)
3. Atualize as distâncias estimadas até v

Algoritmo de Dijkstra (I)

Entrada

- Grafo $G = (V, E)$, direcionado ou não direcionado.
- Vértice $s \in V$
- $w(e)$ para todo $e \in E$

Saída

$d[u]$: distância mínima de s até todo u alcançável

Algoritmo de Dijkstra (II)

Inicialização

1 $d[s] \leftarrow 0$
2 **for** all $v \in V - \{s\}$
3 **do** $d[s] \leftarrow \infty$
4 $S \leftarrow \emptyset$
5 $Q \leftarrow V$

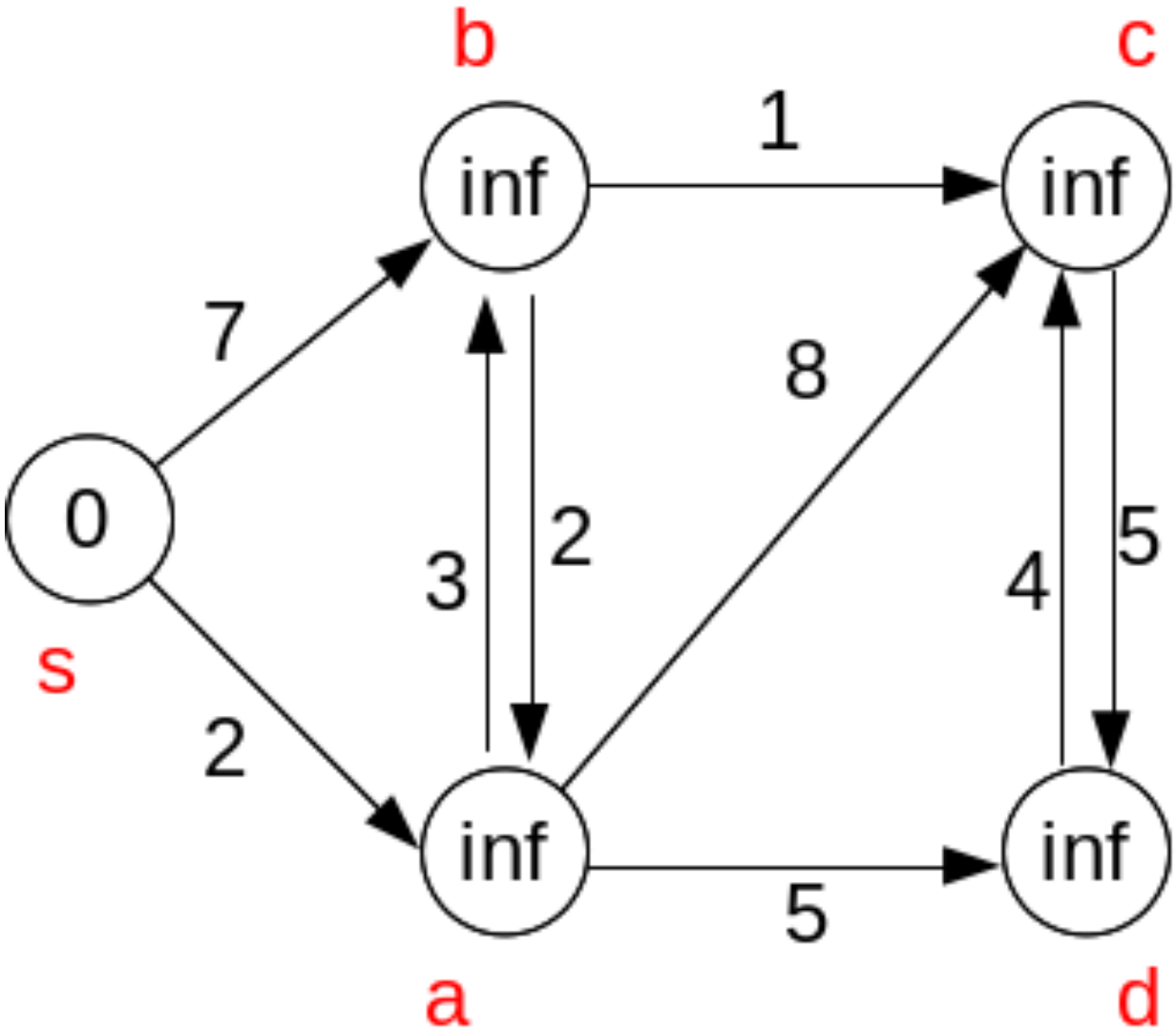
Q é uma fila de prioridades que conterá $V - S$

Algoritmo de Dijkstra (III)

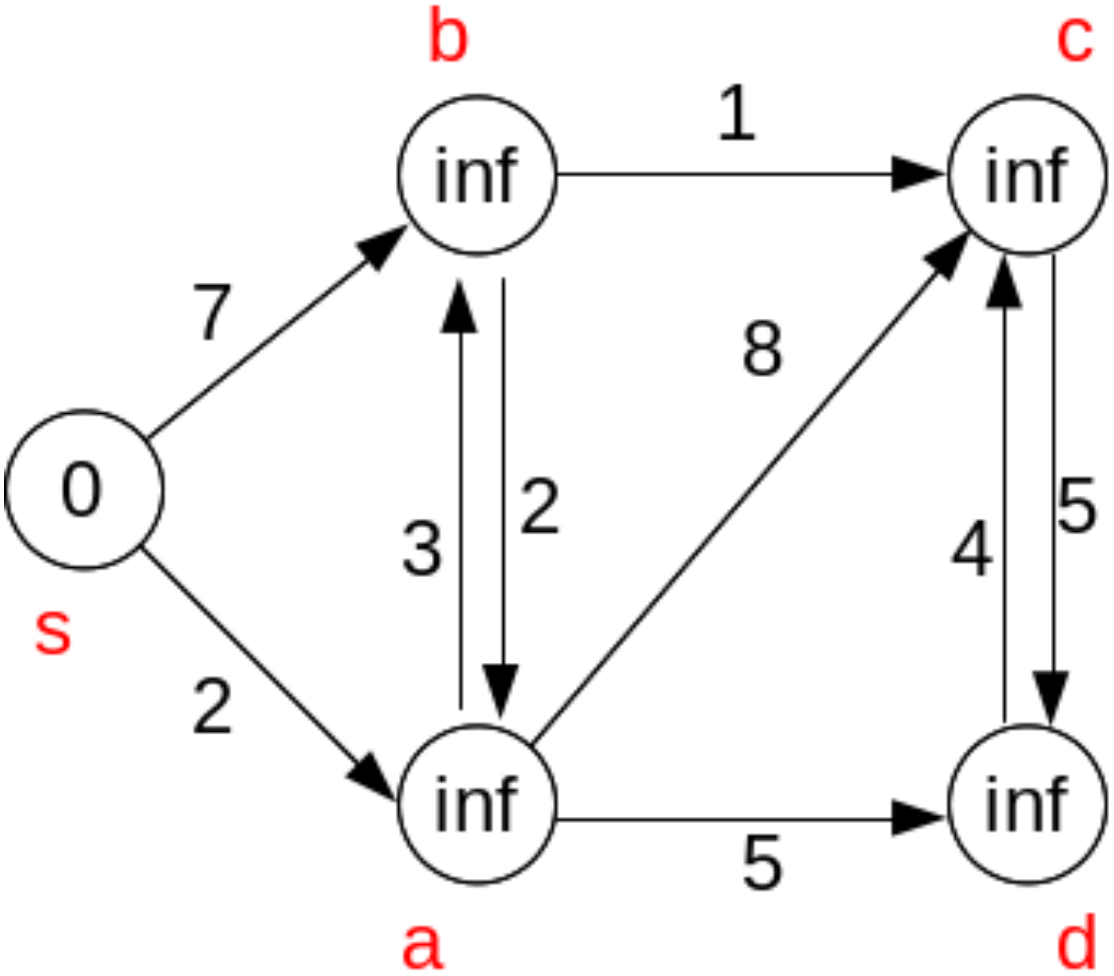
Percurso

```
1  while Q ≠ ∅
2        do u ← Menor_Chave(Q)
3           S ← S ∪ {u}
4           for all v ∈ Adj[u]
5                 do if d[v] > d[u] + w(u, v)
6                       then d[v] ← d[u] + w(u, v)
7                            Decrementa_Chave(Q, v, d[v])
```

Exemplo – Entrada



Exemplo – Passo I

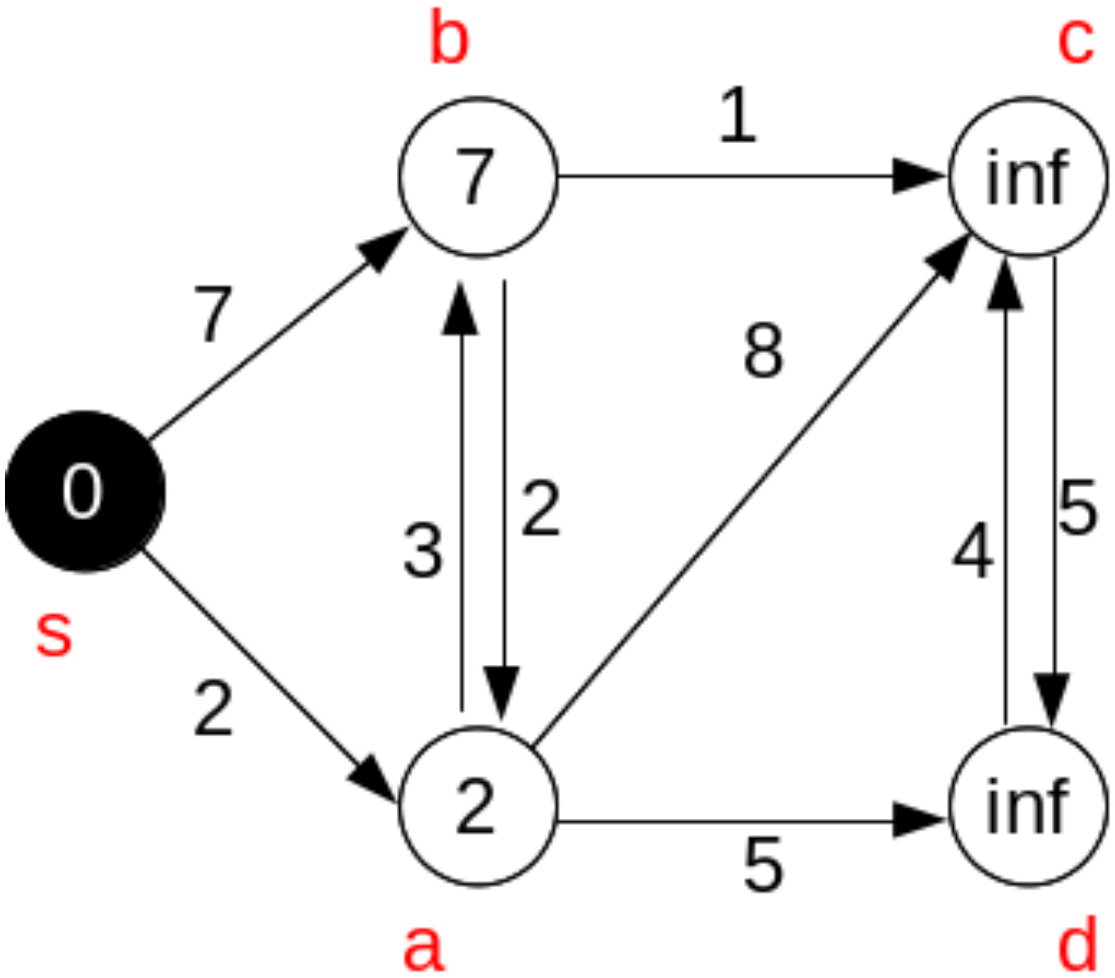


v	s	a	b	c	d
$d[v]$	0	∞	∞	∞	∞
$p[v]$	–	–	–	–	–

v	s	a	b	c	d
$d[v]$	0	∞	∞	∞	∞

Q

Exemplo – Passo II

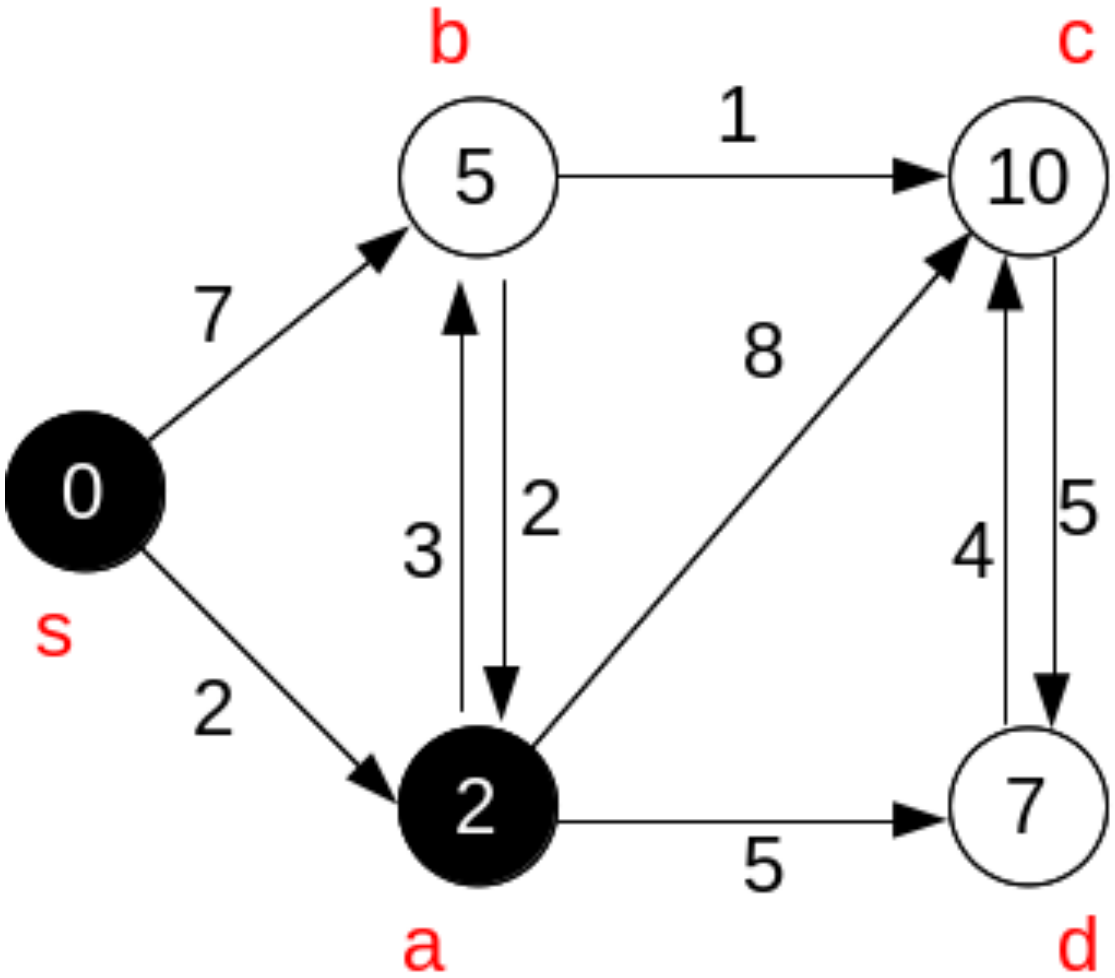


v	s	a	b	c	d
$d[v]$	0	2	7	∞	∞
$p[v]$	–	s	s	–	–

v	a	b	c	d
$d[v]$	2	7	∞	∞

Q

Exemplo – Passo III

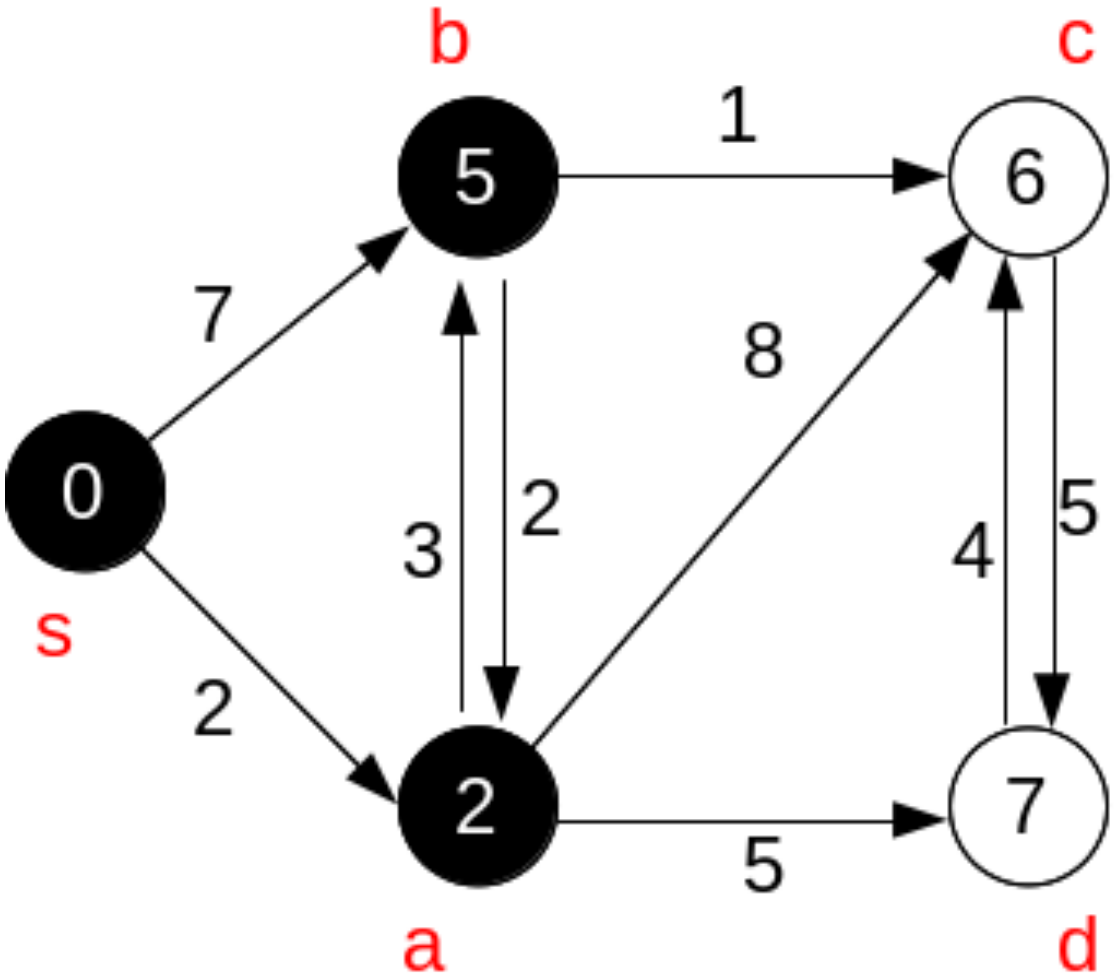


v	s	a	b	c	d
$d[v]$	0	2	5	10	7
$p[v]$	–	s	a	a	a

v	b	c	d
$d[v]$	5	10	7

Q

Exemplo – Passo IV

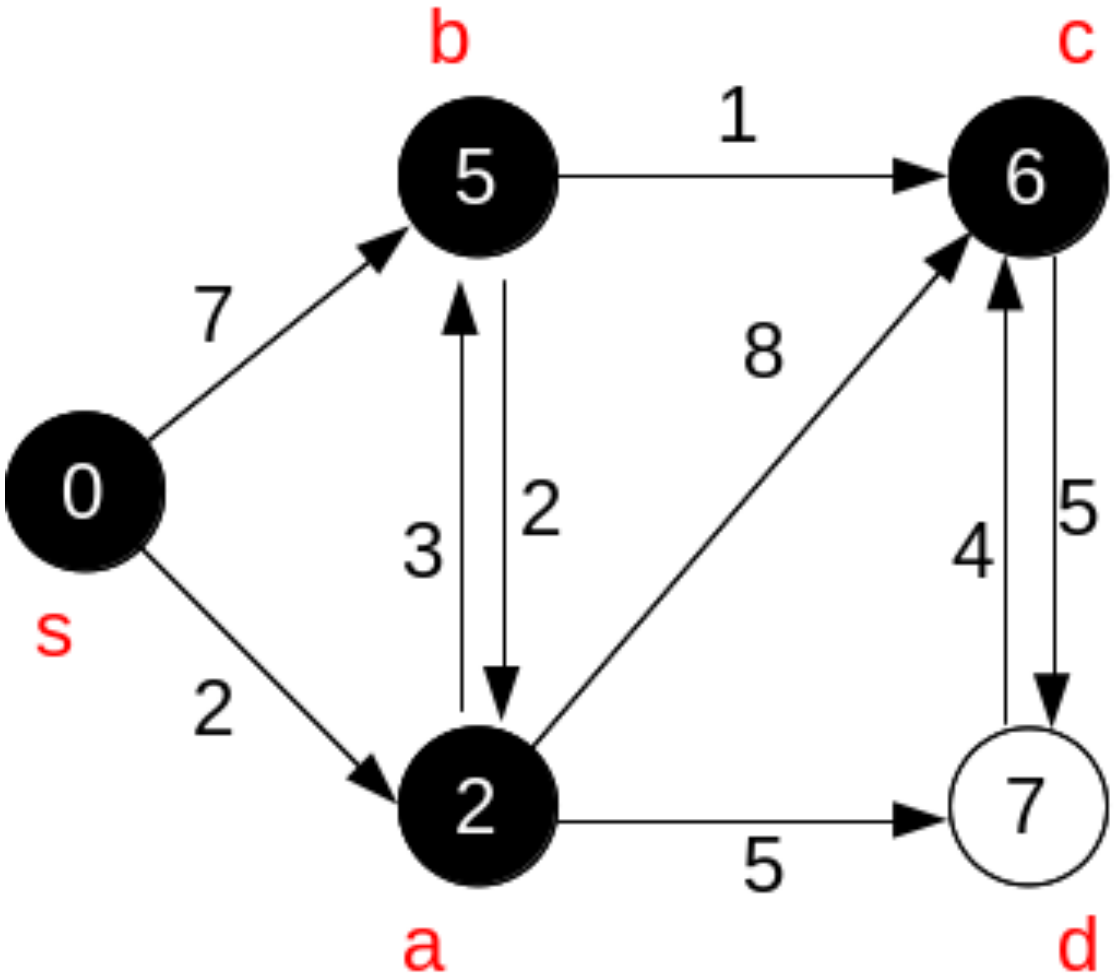


v	s	a	b	c	d
$d[v]$	0	2	5	6	7
$p[v]$	–	s	s	b	a

v	c	d
$d[v]$	6	7

Q

Exemplo – Passo V

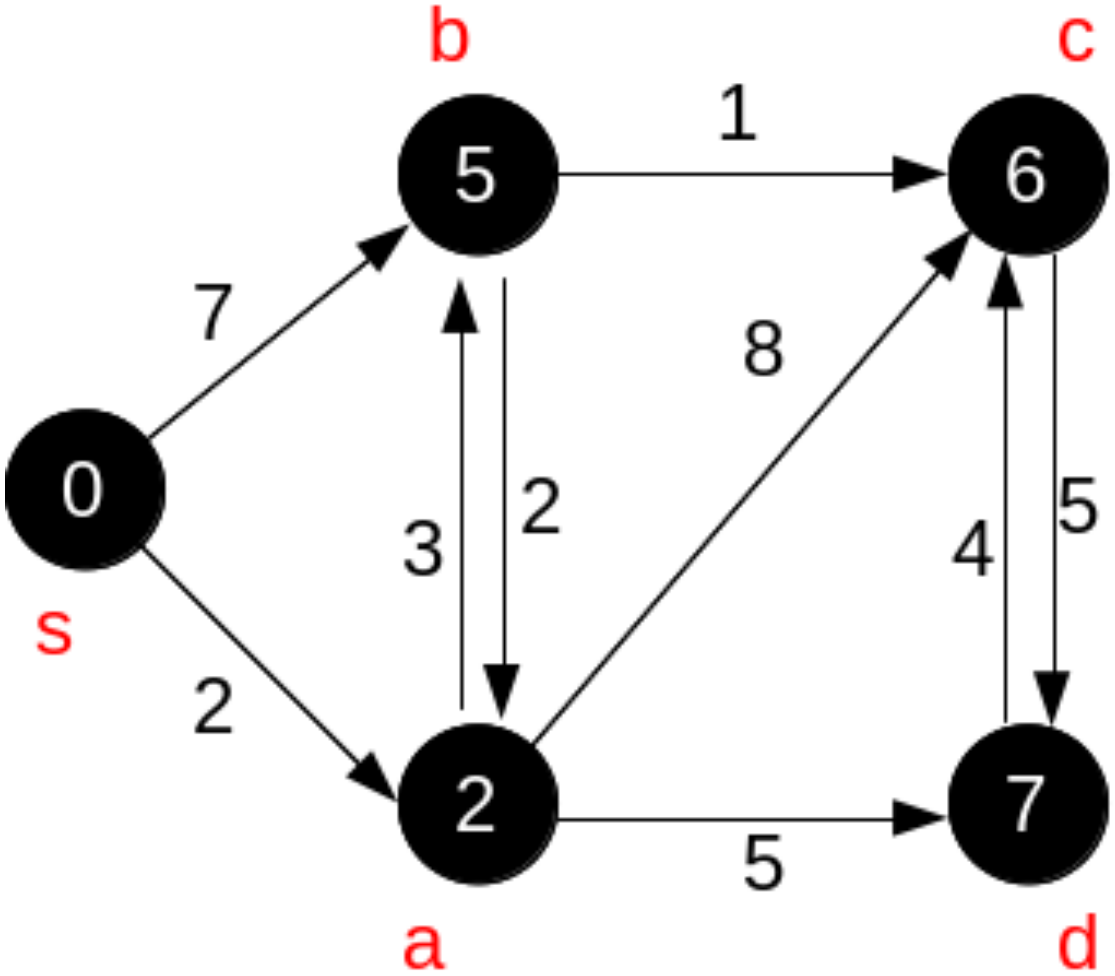


v	s	a	b	c	d
$d[v]$	0	2	5	6	7
$p[v]$	–	s	s	b	a

v	d
$d[v]$	7

Q

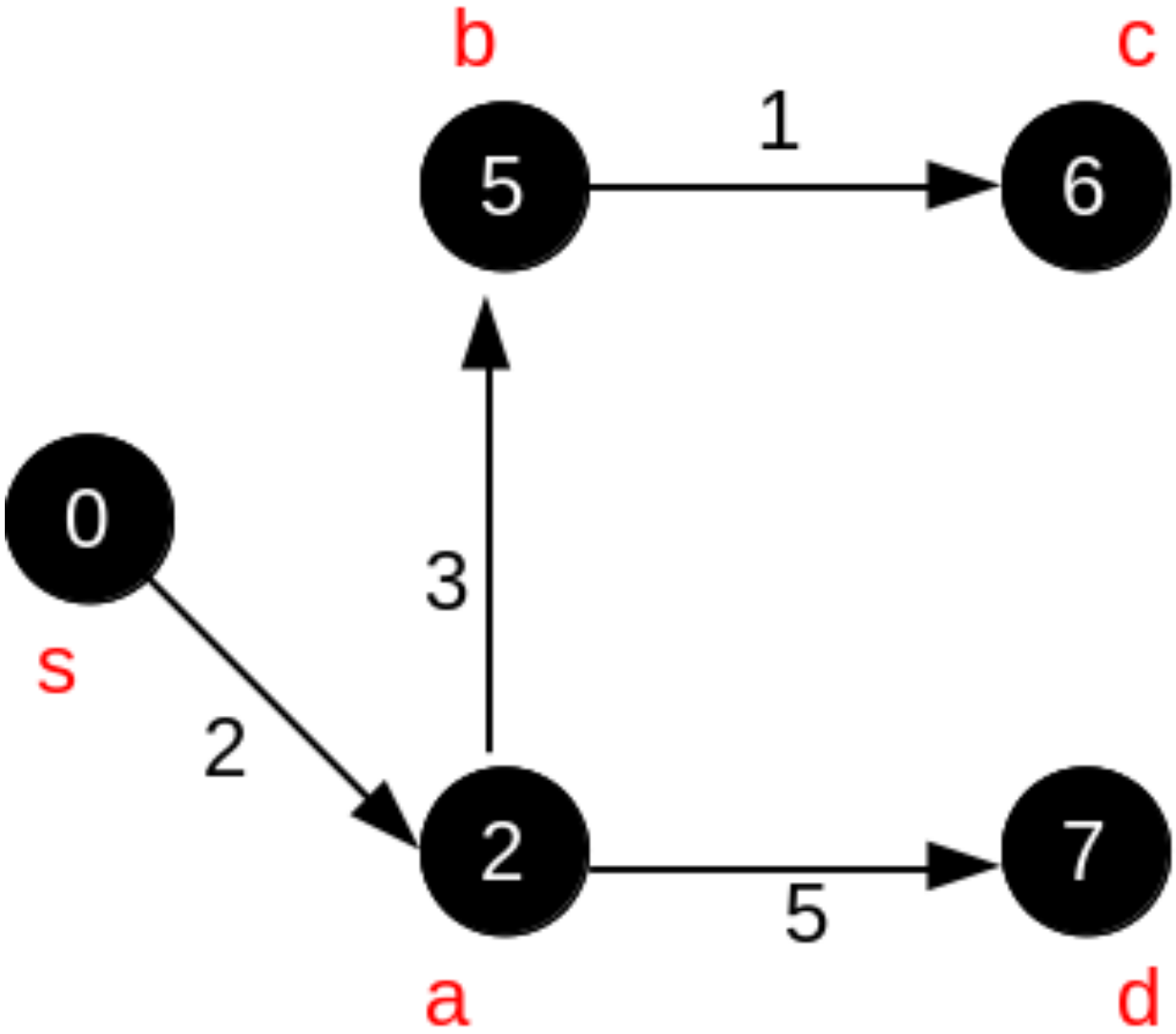
Exemplo – Passo VI



v	s	a	b	c	d
$d[v]$	0	2	5	6	7
$p[v]$	–	s	s	b	a

$$Q = \emptyset$$

Exemplo – Saída



Análise do algoritmo de Dijkstra (I)

Teorema

O algoritmo de Dijkstra sempre devolve $d[v] = \delta(s, v)$ para todo $v \in V$

Análise do algoritmo de Dijkstra (II)

- Inicialização: $\mathcal{O}(|V|)$ vezes
- Durante o percurso:
  - $\mathcal{O}(|V|)$ operações de `Menor_Chave()`
- No laço mais interno:
  - $\mathcal{O}(|V| + |E|)$ operações de `Decrementa_Chave()`

Análise do algoritmo de Dijkstra (III)

Complexidade da Fila

- Depende da implementação
  - Em geral, usa-se uma árvore *heap*
  - Inserção/Atualização: $\mathcal{O}(\log |V|)$

Complexidade do algoritmo de Dijkstra

$$\mathcal{O}(|V| + (|E| + |V|) \log(|V|))$$

Limitações no algoritmo de Dijkstra

Ciclos Negativos

- Arestas com peso menor que 0
- Ocorrem quando modelamos determinados problemas do mundo real por meio de (di)grafos
- Impedem a existência de caminhos mínimos

Reformulando Problema

Dado um (di)grafo ponderado e um vértice $s \in V$, encontre $\delta(s, v)$, para todo $v \in V$ OU determine a existência de ciclos negativos

Algoritmo de Bellman-Ford (I)

Inicialização

1 $d[s] \leftarrow 0$
2 **for** all $v \in V - \{s\}$
3 **do** $d[s] \leftarrow \infty$

Algoritmo de Bellman-Ford (II)

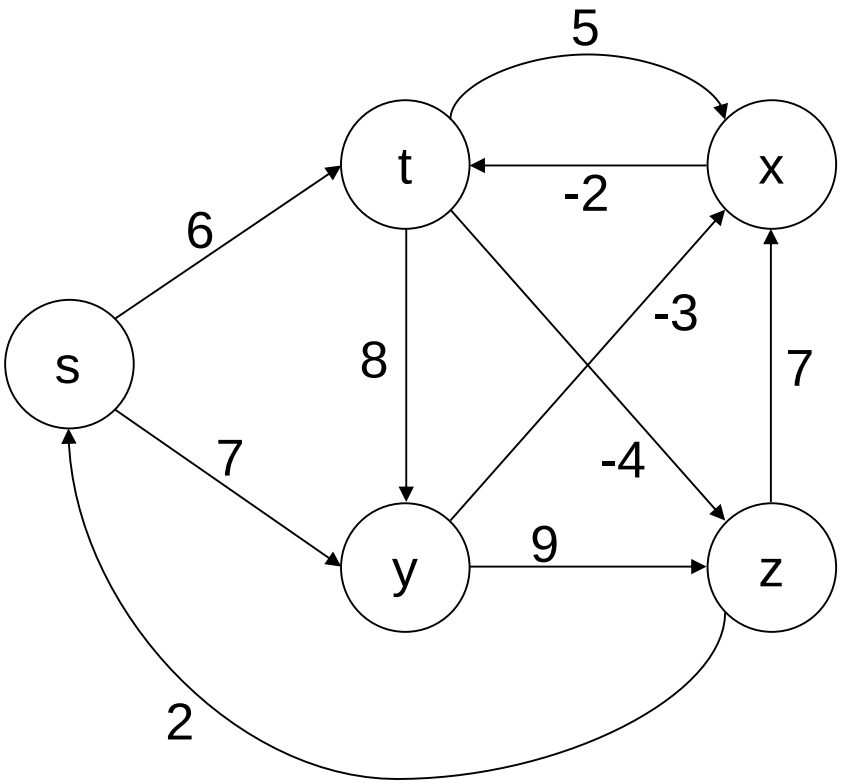
Percurso

```
1  for i ← to |V| − 1
2        do for all (u, v) ∈ E
3              do if d[v] > d[u] + w(u, w)
4                    then d[v] ← d[u] + w(u, v)
5  for all (u, v) ∈ E
6        do if d[v] > d[u] + w(u, v)
7              then reporta existência de ciclo negativo.
```

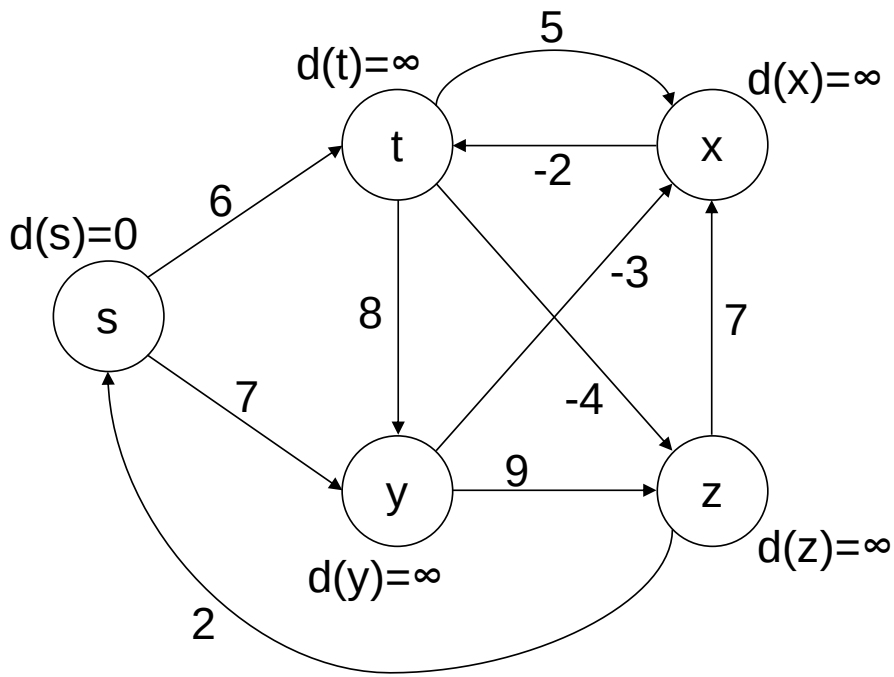
Complexidade

$$\mathcal{O}(|V| \cdot |E|)$$

Exemplo de execução (I)

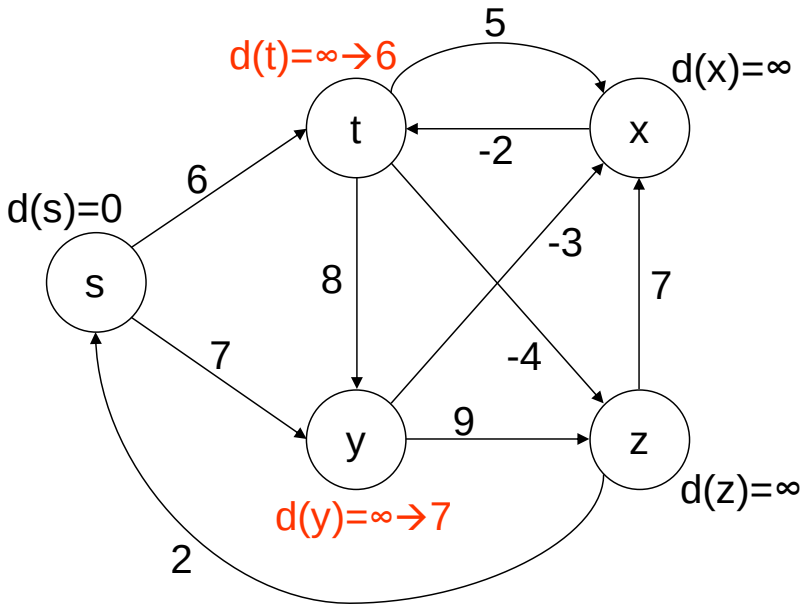


Estimando o tempo...

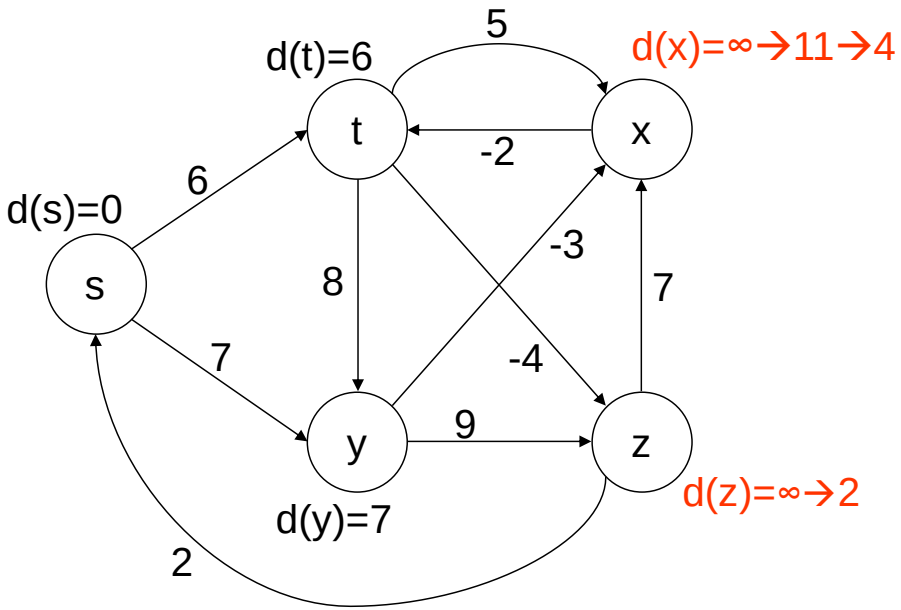


Exemplo de execução (II)

Primeira rodada

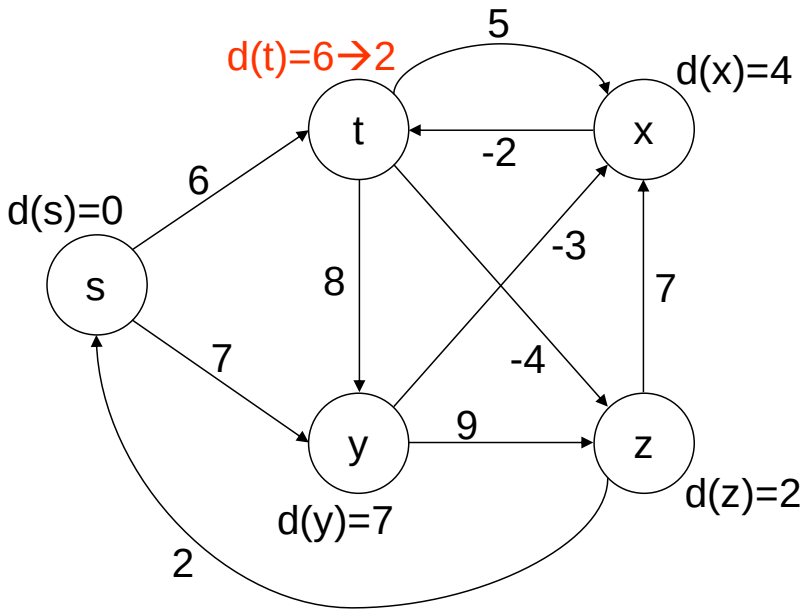


Segunda rodada

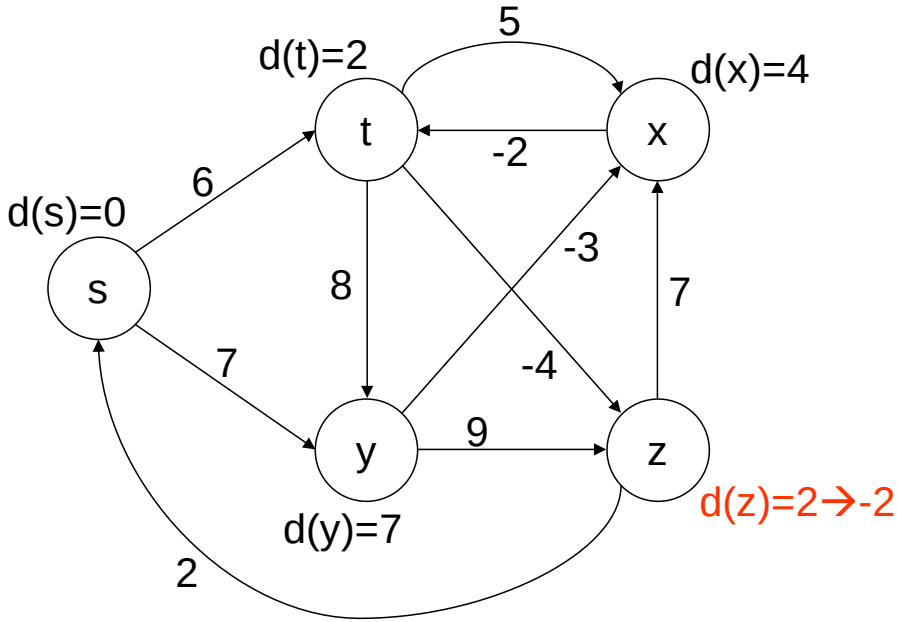


Exemplo de execução (III)

Terceira rodada



Quarta rodada



Outro problema envolvendo caminhos

- Até agora, vimos o problema envolvendo uma única fonte
- Há casos em que temos a necessidade de saber menores caminhos entre todos os vértices
- Problema de *All-Pairs Shortest Paths*

Possível Solução

Utilizar o algoritmo de Dijkstra considerando cada vértice como origem alternadamente

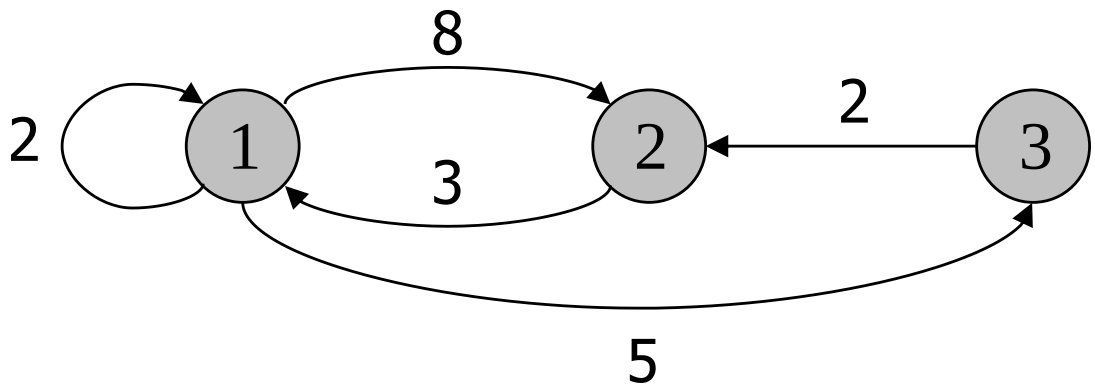
Outra possibilidade

Algoritmo de Floyd-Warshall

Utiliza uma matriz $|V| \times |V|$ para calcular e armazenar os tamanhos dos caminhos mais curtos

Algoritmo de Floyd-Warshall – Exemplo

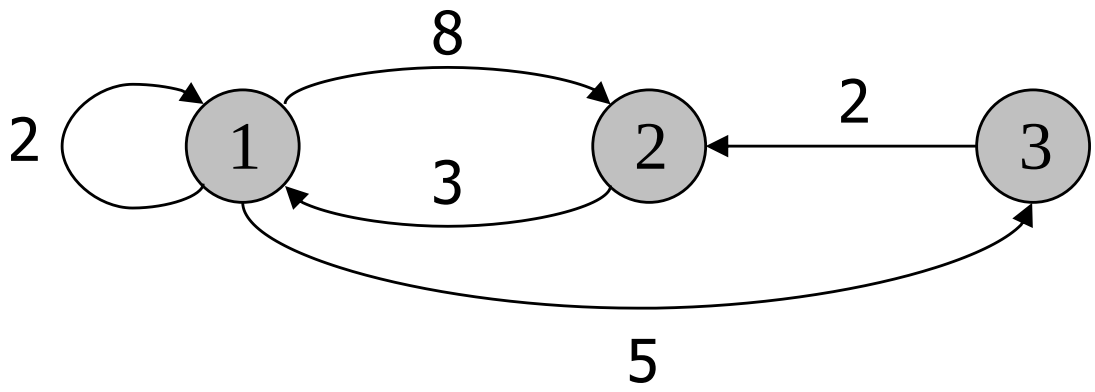
- Inicialização: custos entre vértices adjacentes são inseridos na matriz
- Ignorar *loops*



	1	2	3
1	0	8	5
2	3	0	∞
3	∞	2	0

Algoritmo de Floyd-Warshall – Exemplo

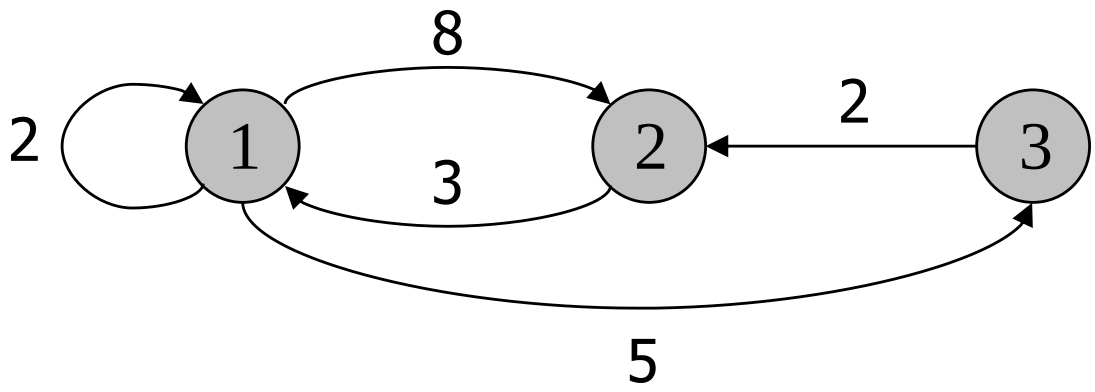
- A matriz é percorrida $|V|$ vezes
- A cada iteração k, verifica-se se um caminho entre dois vértices (v, w) que passa também pelo vértice k é mais curto que o caminho mais curto conhecido



$$d[v, w] = \min\{d[v, w], d[v, k]+d[k, w]\}$$

	1	2	3
1	0	8	5
2	3	0	∞
3	∞	2	0

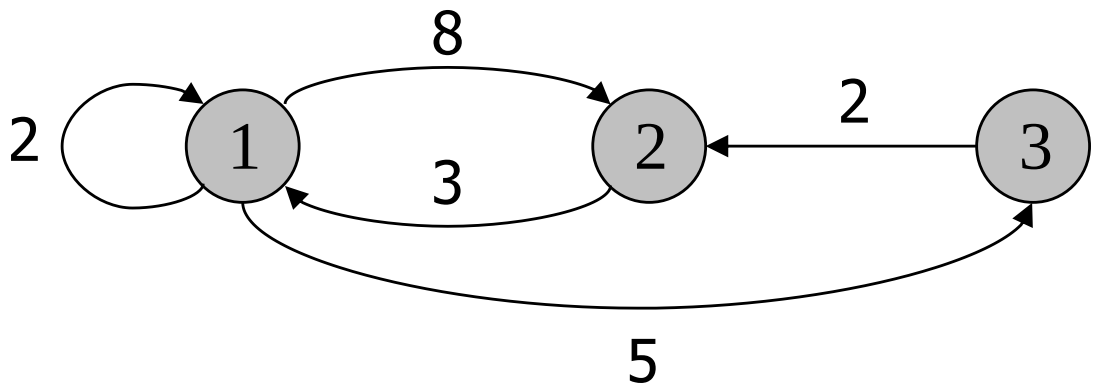
Algoritmo de Floyd-Warshall – Exemplo



$d[1,1] = \min\{d[1,1], d[1,1]+d[1,1]\}$

	1	2	3
1	0	8	5
2	3	0	∞
3	∞	2	0

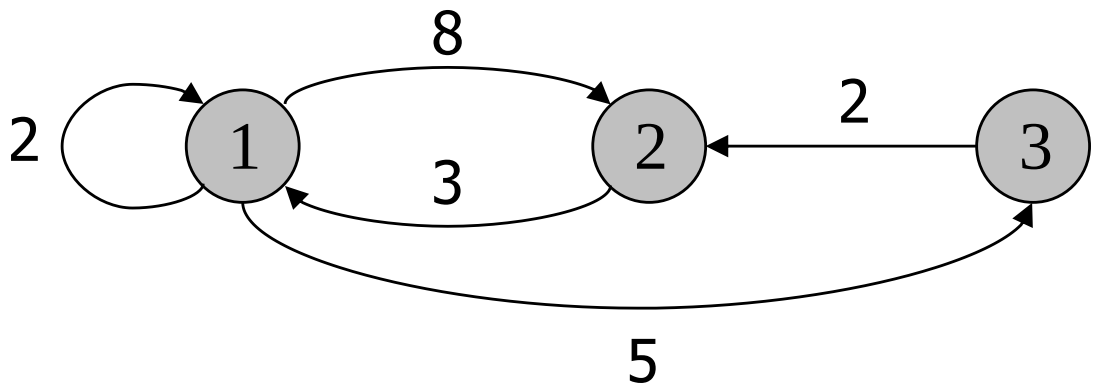
Algoritmo de Floyd-Warshall – Exemplo



$d[1,2] = \min\{d[1,2], d[1,1]+d[1,2]\}$

	1	2	3
1	0	8	5
2	3	0	∞
3	∞	2	0

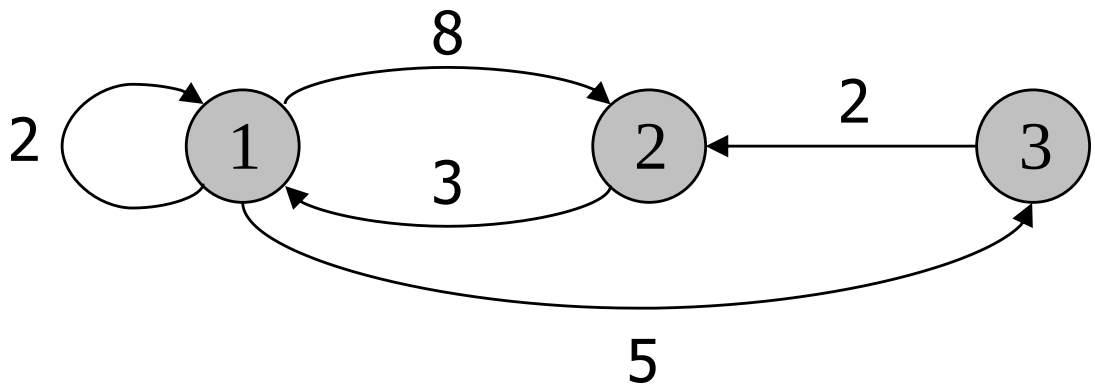
Algoritmo de Floyd-Warshall – Exemplo



$d[1,3] = \min\{d[1,3], d[1,1]+d[1,3]\}$

	1	2	3
1	0	8	5
2	3	0	∞
3	∞	2	0

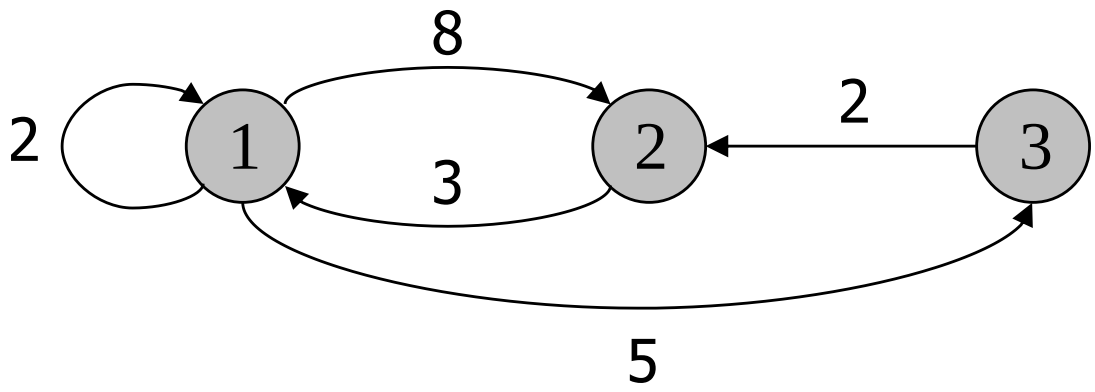
Algoritmo de Floyd-Warshall – Exemplo



$d[2,1] = \min\{d[2,1], d[2,1]+d[1,1]\}$

	1	2	3
1	0	8	5
2	3	0	∞
3	∞	2	0

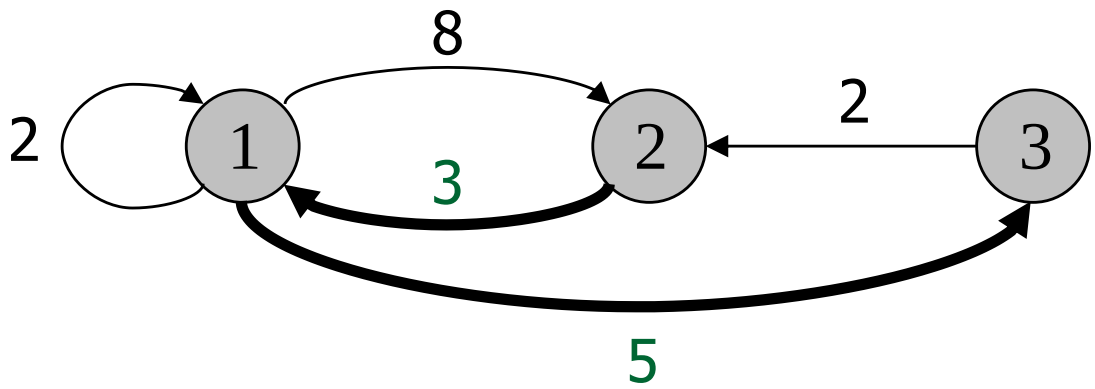
Algoritmo de Floyd-Warshall – Exemplo



$d[2,2] = \min\{d[2,2], d[2,1]+d[1,2]\}$

	1	2	3
1	0	8	5
2	3	0	∞
3	∞	2	0

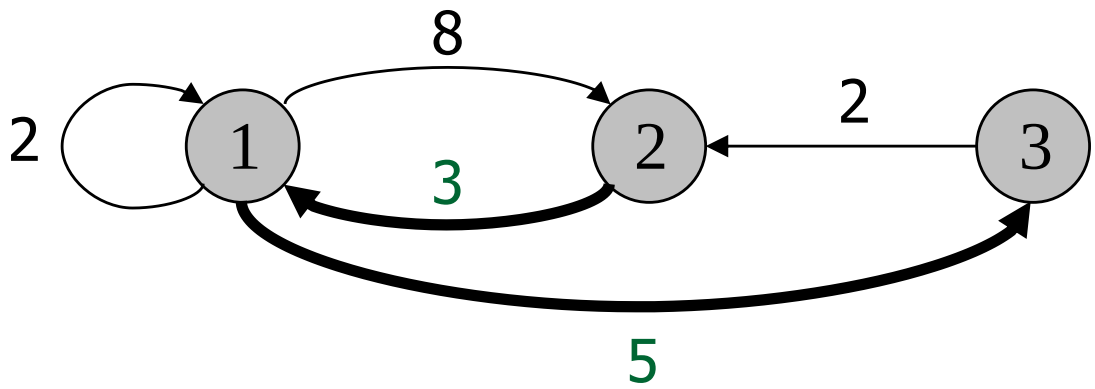
Algoritmo de Floyd-Warshall – Exemplo



$d[2,3] = \min\{d[2,3], d[2,1]+d[1,3]\}$

	1	2	3
1	0	8	5
2	3	0	∞
3	∞	2	0

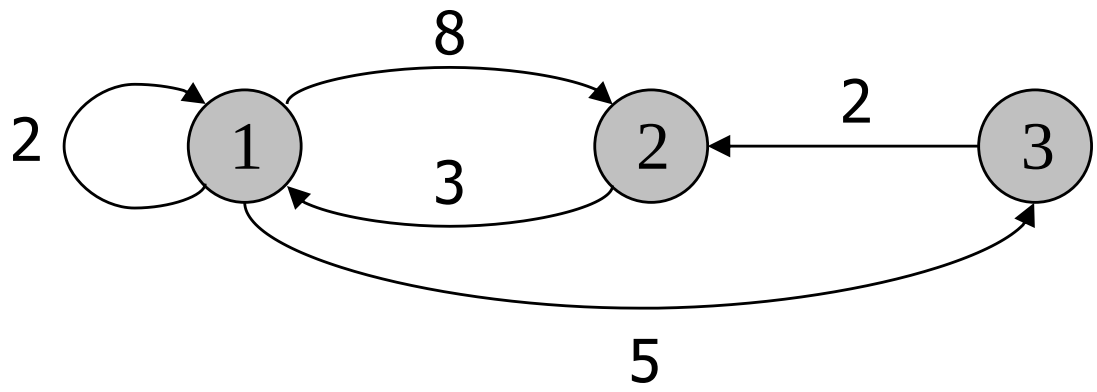
Algoritmo de Floyd-Warshall – Exemplo



$d[2,3] = \min\{d[2,3], d[2,1]+d[1,3]\}$

	1	2	3
1	0	8	5
2	3	0	8
3	∞	2	0

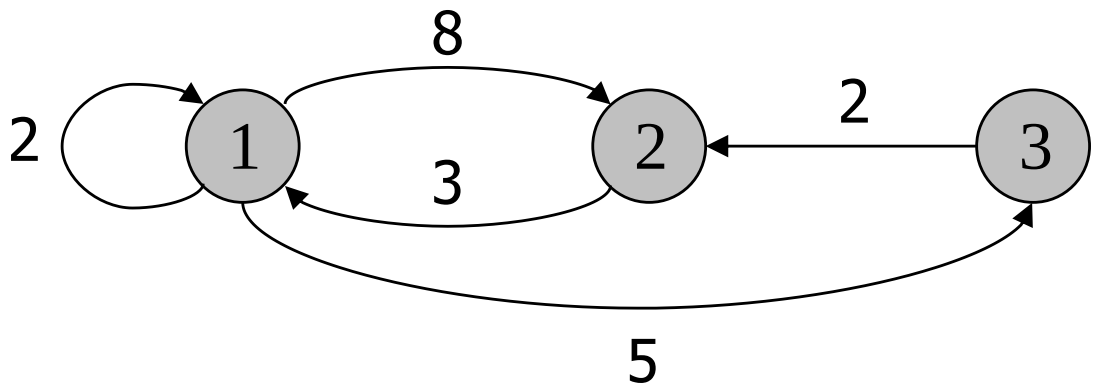
Algoritmo de Floyd-Warshall – Exemplo



$d[3,1] = \min\{d[3,1], d[3,1]+d[1,3]\}$

	1	2	3
1	0	8	5
2	3	0	8
3	∞	2	0

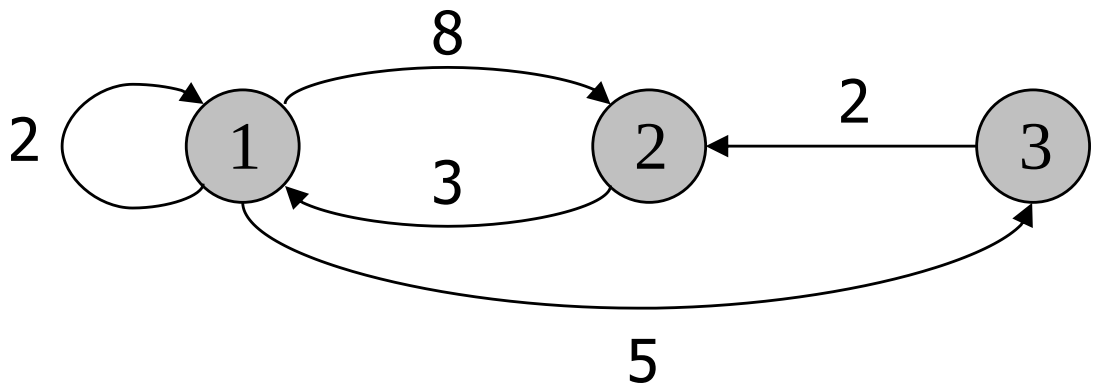
Algoritmo de Floyd-Warshall – Exemplo



$d[3,2] = \min\{d[3,2], d[3,1]+d[1,2]\}$

	1	2	3
1	0	8	5
2	3	0	8
3	∞	2	0

Algoritmo de Floyd-Warshall – Exemplo

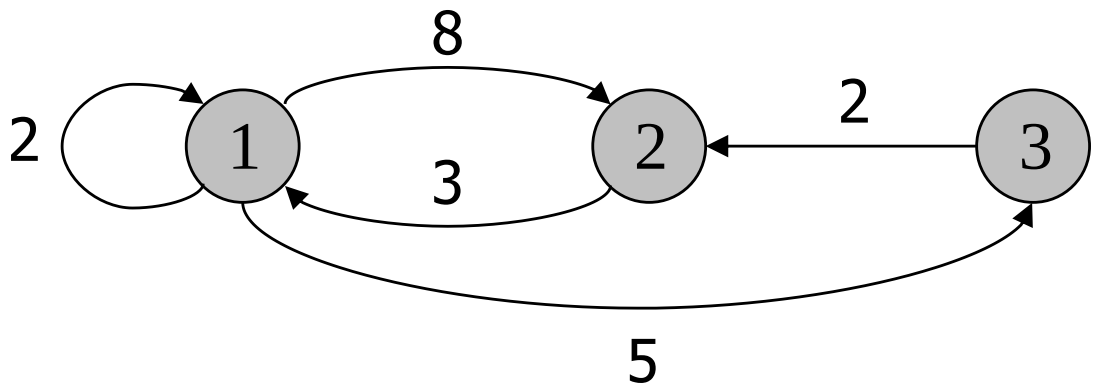


$d[3,3] = \min\{d[3,3], d[3,1]+d[1,3]\}$

	1	2	3
1	0	8	5
2	3	0	8
3	∞	2	0

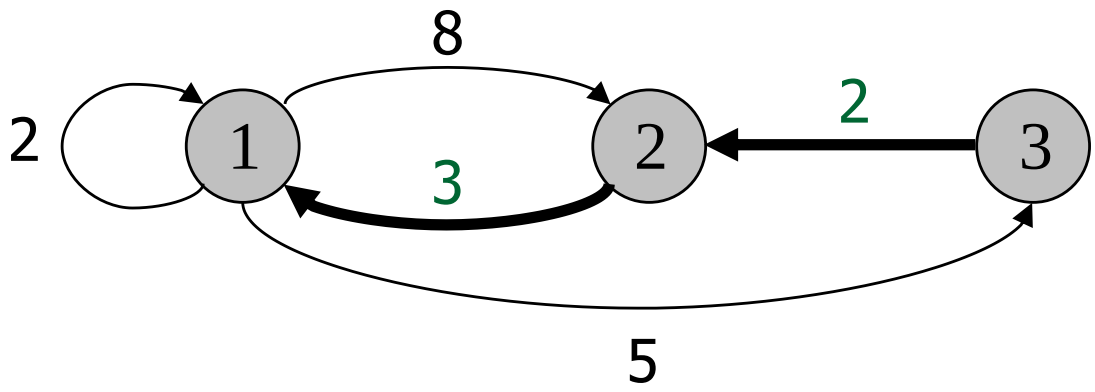
Algoritmo de Floyd-Warshall – Exemplo

- Ao final da primeira iteração, sabemos os caminhos mais curtos entre v e w que passam por 1
- Repete-se o processo considerando os demais vértices



	1	2	3
1	0	8	5
2	3	0	8
3	∞	2	0

Algoritmo de Floyd-Warshall – Exemplo

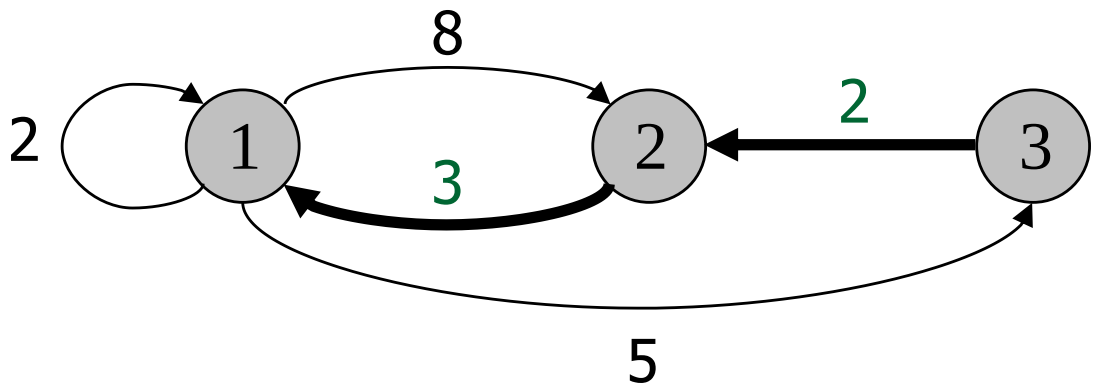


...

$d[3,1] = \min\{d[3,1], d[3,2]+d[2,1]\}$

	1	2	3
1	0	8	5
2	3	0	8
3	∞	2	0

Algoritmo de Floyd-Warshall – Exemplo

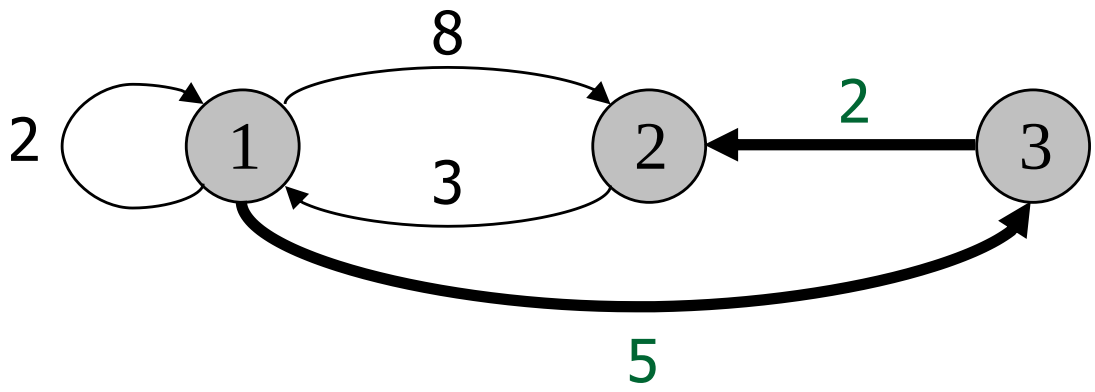


...

$d[3,1] = \min\{d[3,1], d[3,2]+d[2,1]\}$

	1	2	3
1	0	8	5
2	3	0	8
3	5	2	0

Algoritmo de Floyd-Warshall – Exemplo

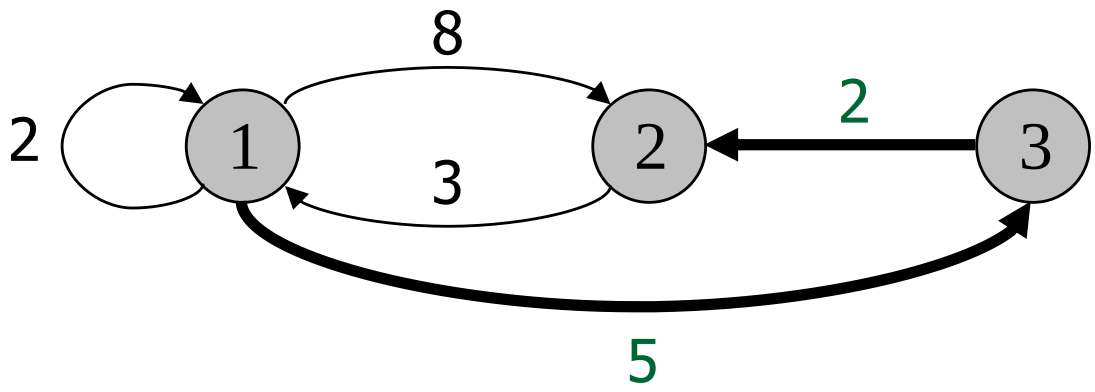


...

$d[1,2] = \min\{d[1,2], d[1,3]+d[3,2]\}$

	1	2	3
1	0	8	5
2	3	0	8
3	5	2	0

Algoritmo de Floyd-Warshall – Exemplo



...

$d[1,2] = \min\{d[1,2], d[1,3]+d[3,2]\}$

	1	2	3
1	0	7	5
2	3	0	8
3	5	2	0

Algoritmo de Floyd-Warshall – Pseudocódigo

Inicialização

```
1  for i ← 1 to |V|
2        do for j ← 1 to |V|
3              do d[i,j] ← w(i,j)
4                 p[i,j] ← null
```

Algoritmo de Floyd-Warshall – Pseudocódigo

Programação Dinâmica

```
1  for k ← 1 to |V|
2        do for i ← 1 to |V|
3              do for j ← 1 to |V|
4                    do if (d[i,k] + d[k,j] < d[i,j])
5                          do d[i,j] ← d[i,k] + d[k,j]
6                             p[i,j] ← k
```

Complexidade

$$\mathcal{O}(|V|^3)$$

Sumário

Dois problemas de caminhos mínimos

Single-source shortest paths

- Dijkstra
  - Arestas não-negativas
  - $\mathcal{O}(|V| + (|E| + |V|) \log(|V|))$
- Bellman-Ford
  - $\mathcal{O}(|V| \cdot |E|)$

All-Pairs Shortest Paths Problem

- Floyd-Warshall
  - $\mathcal{O}(|V|^3)$

Leitura Recomendada I


CORMEN, T. H. et al.
Introduction to algorithms,The MIT Press, 2.ed., 2001.
Cap. 24.


SEDGEWICK, R.
Algorithms in C: part 5, 3.ed., Addison-Wesley, 2002.
Cap. 21 - Sec. 21.1, 21.2, 21.3, 21.7


ZIVIANI, N.
Projeto de Algoritmos, 3.ed. Cengage, 2004.
Cap. 7 - Sec. 7.8


DASGUPTA, S. et al.
Algoritmos, McGraw-Hill, 2008.
Cap. 4

Agradecimentos

Ao prof. Thiago Pardo, que forneceu parte das figuras utilizadas nesta aula.