Introdução à Teoria dos Grafos

Prof. Alexandre Noma

Semana passada...

- Busca em largura: “**fila**”

- Busca em profundidade: “**pilha**”

Semana passada...

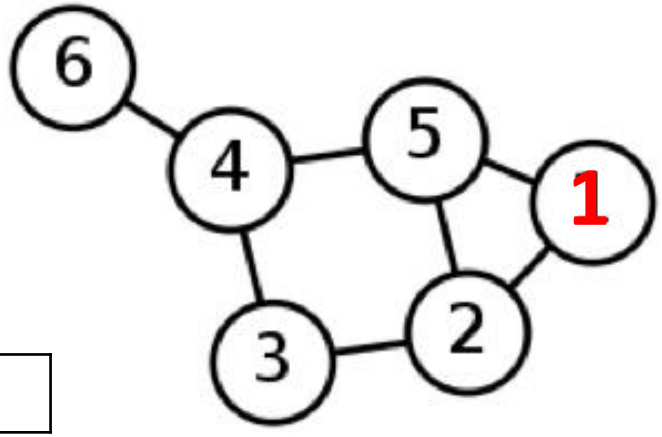
- Busca em **largura**: “fila”

início  fim

- Busca em **profundidade**: “pilha”

 topo

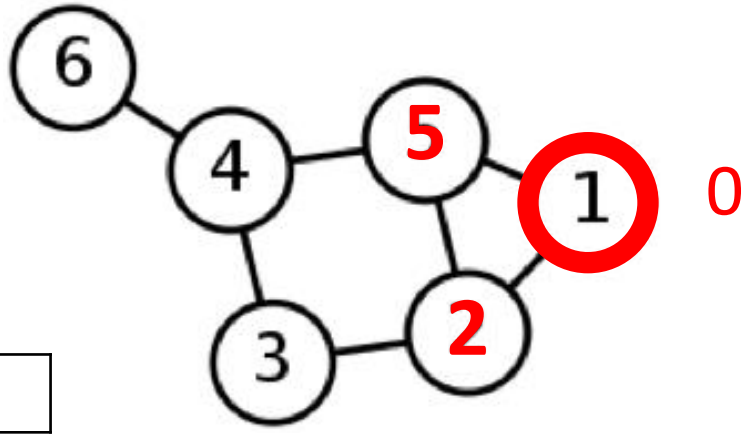
Semana passada: “largura” (fila)



1					

Semana passada: “largura” (fila)

Vértice inicial.

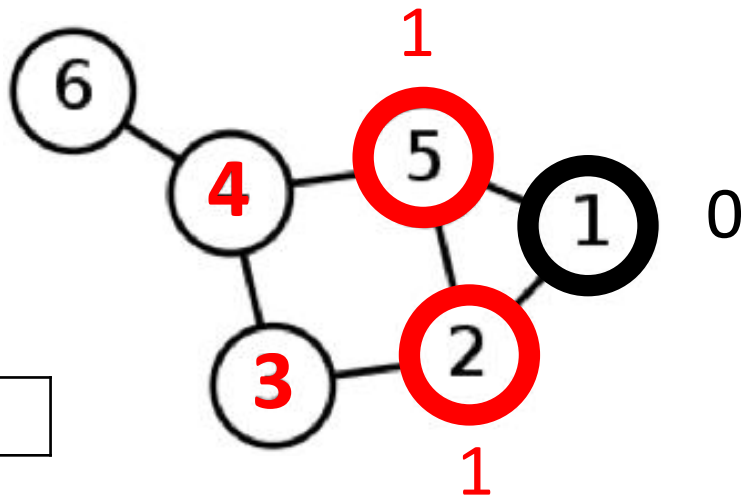


0

~~1~~	2	5			

Semana passada: “largura” (fila)

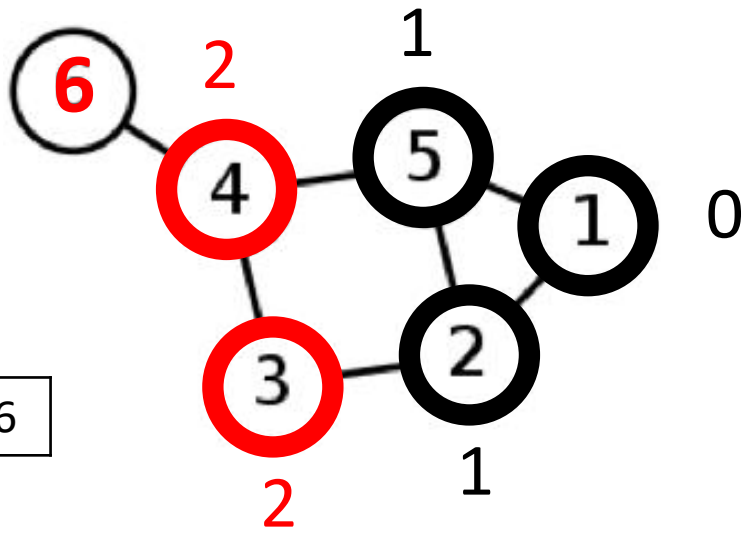
Fogo se propaga para vizinhos.



~~1~~	~~2~~	~~5~~	3	4	

Semana passada: “largura” (fila)

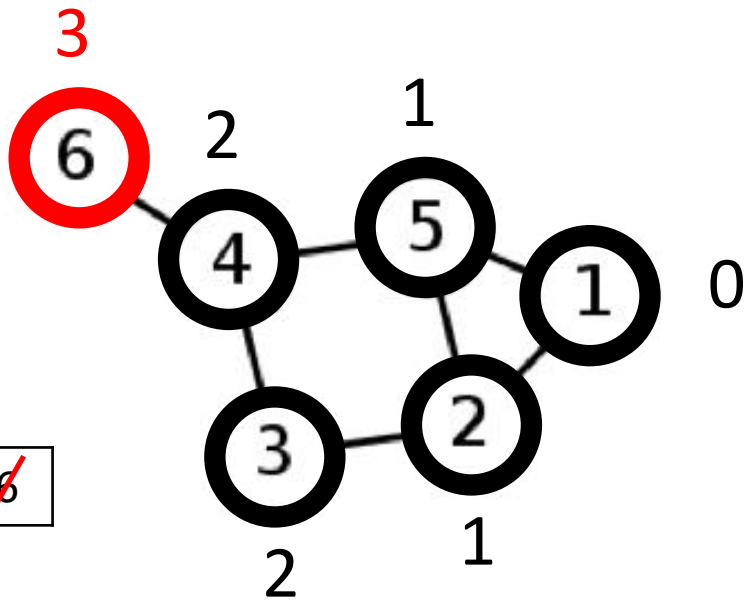
Fogo se propaga para vizinhos.



~~1~~	~~2~~	~~5~~	~~3~~	~~4~~	6

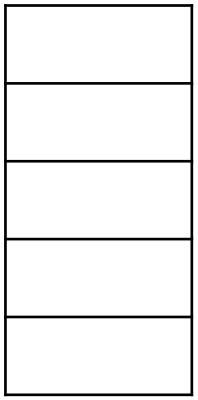
Semana passada: “largura” (fila)

Fogo se propaga para vizinhos.

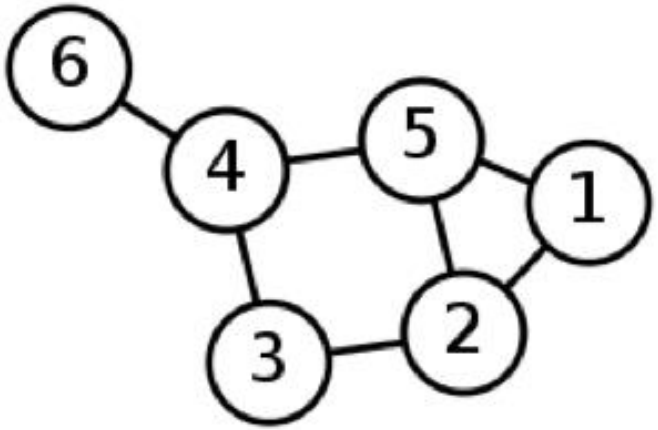


~~1~~	~~2~~	~~5~~	~~3~~	~~4~~	~~6~~

Sem. passada: “profundidade” (pilha)

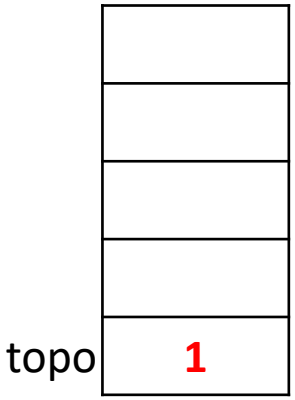


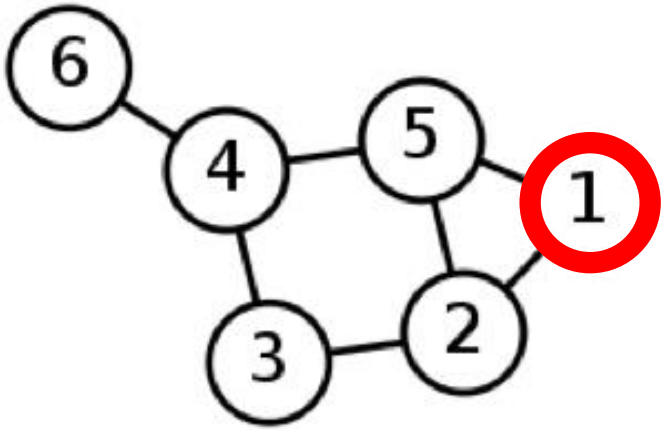
topo = -1 (pilha vazia)



Sem. passada: “profundidade” (pilha)

Vértice inicial.





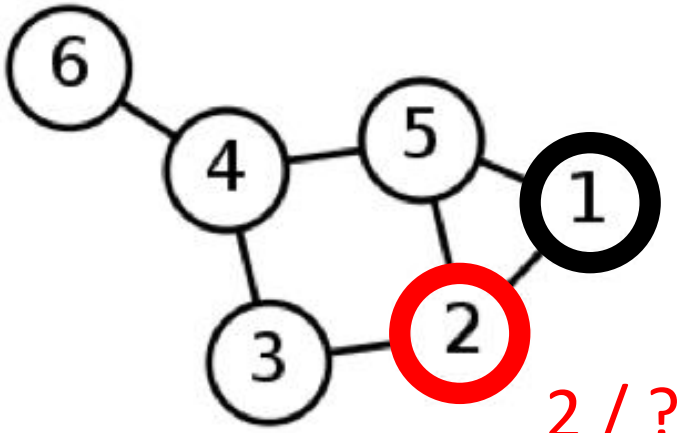
1 / ?

Sem. passada: “profundidade” (pilha)

Vizinho.

topo

2
1



1 / ?

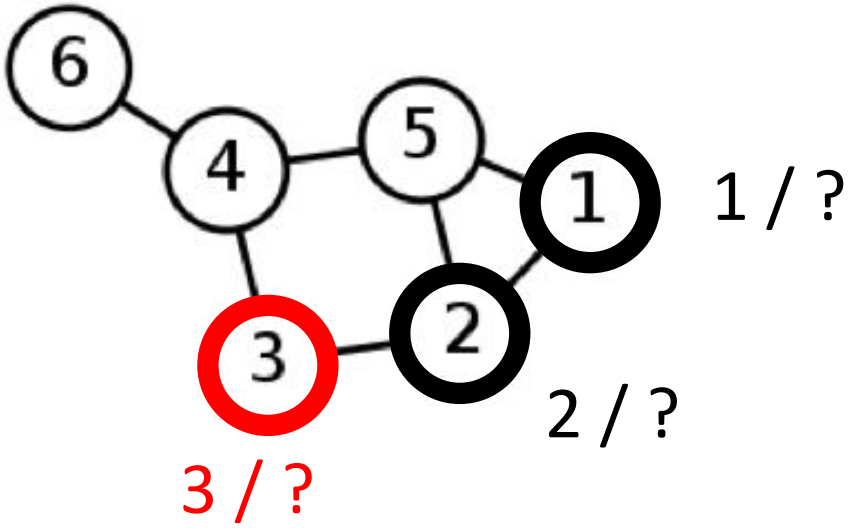
2 / ?

Sem. passada: “profundidade” (pilha)

Vizinho.

topo

3
2
1

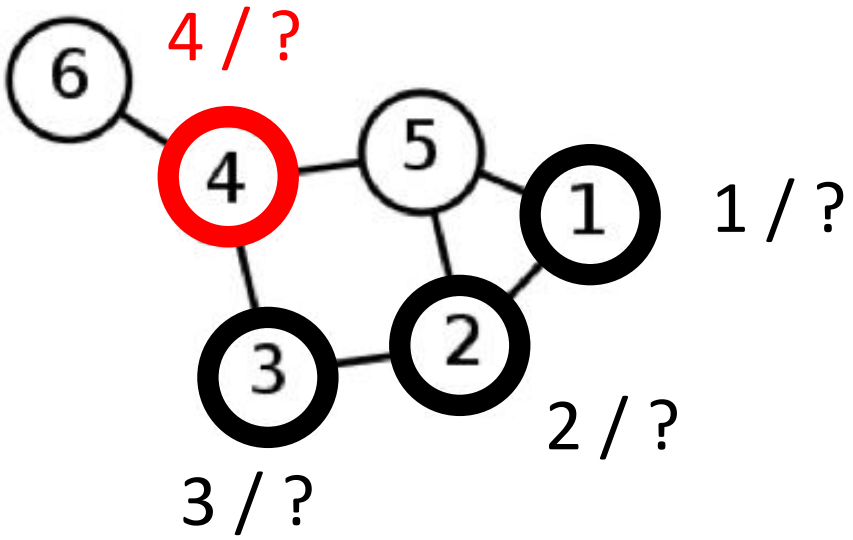


Sem. passada: “profundidade” (pilha)

Vizinho.

topo

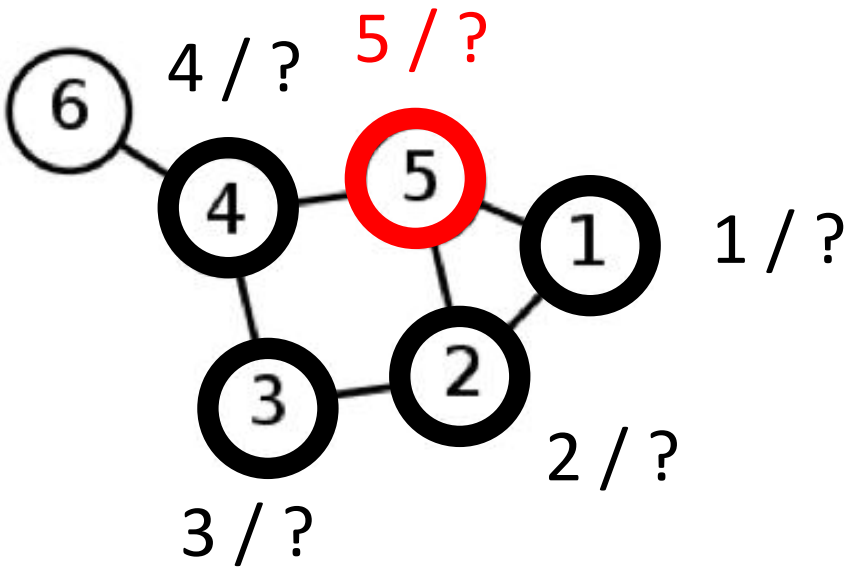
4
3
2
1



Sem. passada: “profundidade” (pilha)

topo	
	5
	4
	3
	2
	1

Vizinho???

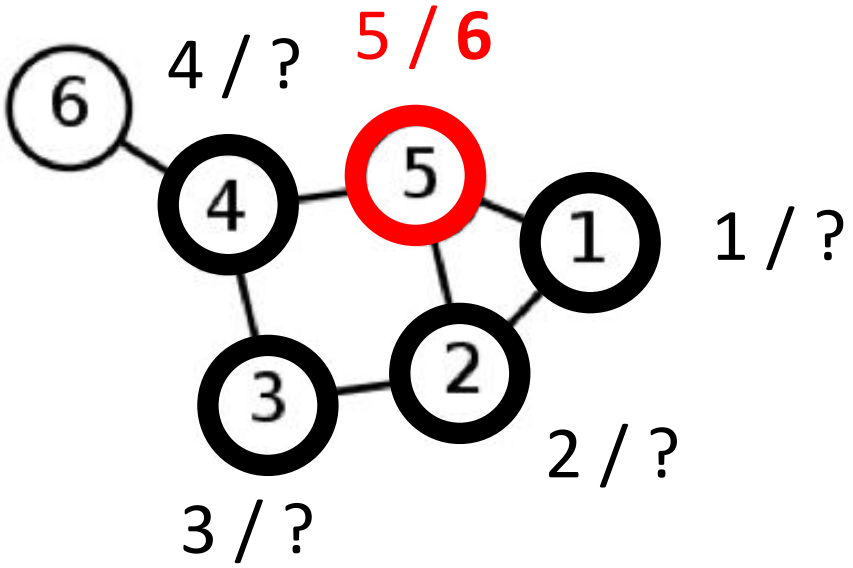


Sem. passada: “profundidade” (pilha)

topo

~~5~~
4
3
2
1

Vizinho???

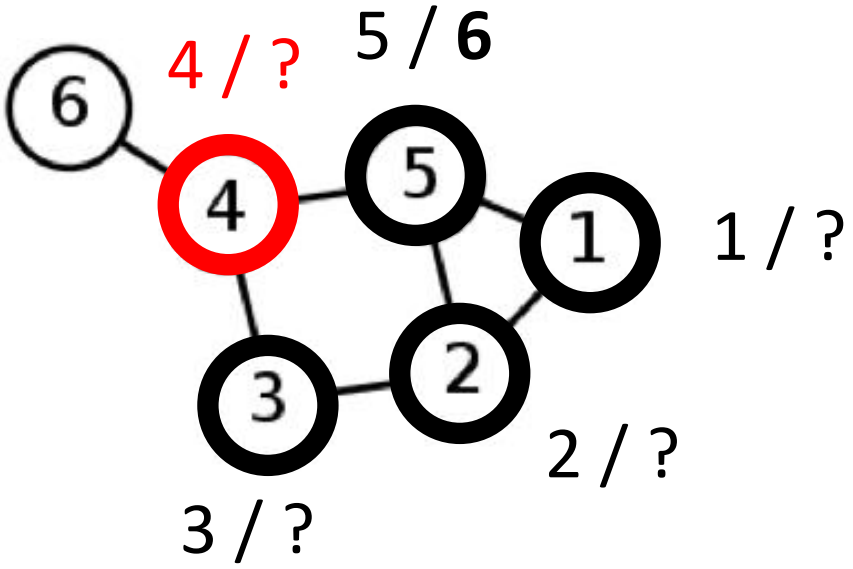


Sem. passada: “profundidade” (pilha)

Vizinho???

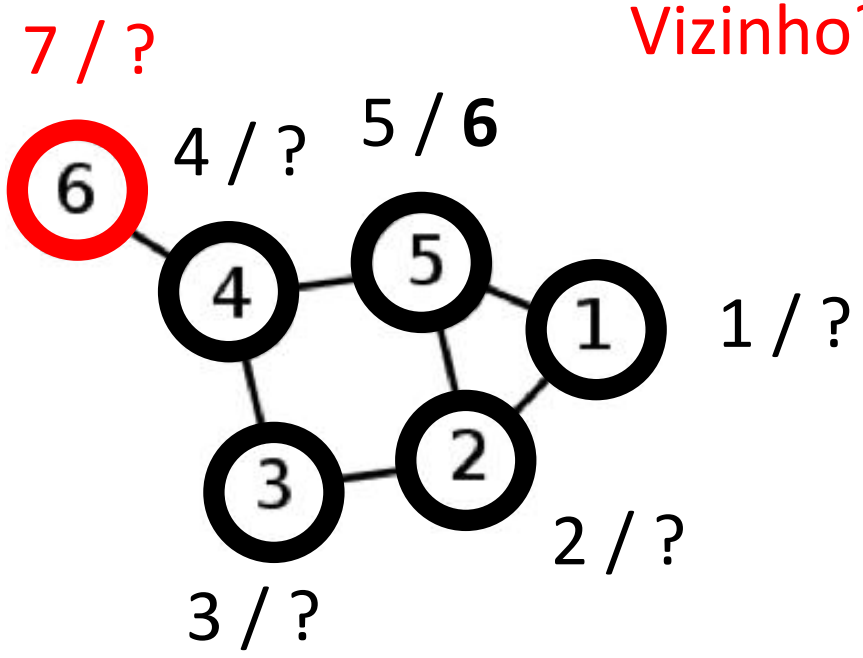
topo

4
3
2
1



Sem. passada: “profundidade” (pilha)

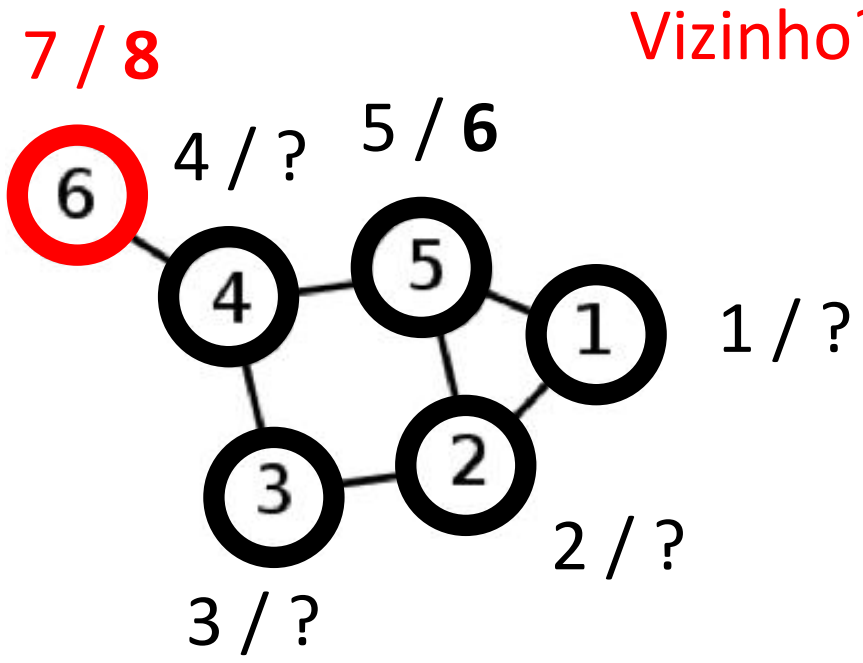
topo	
	6
	4
	3
	2
	1



Sem. passada: “profundidade” (pilha)

topo

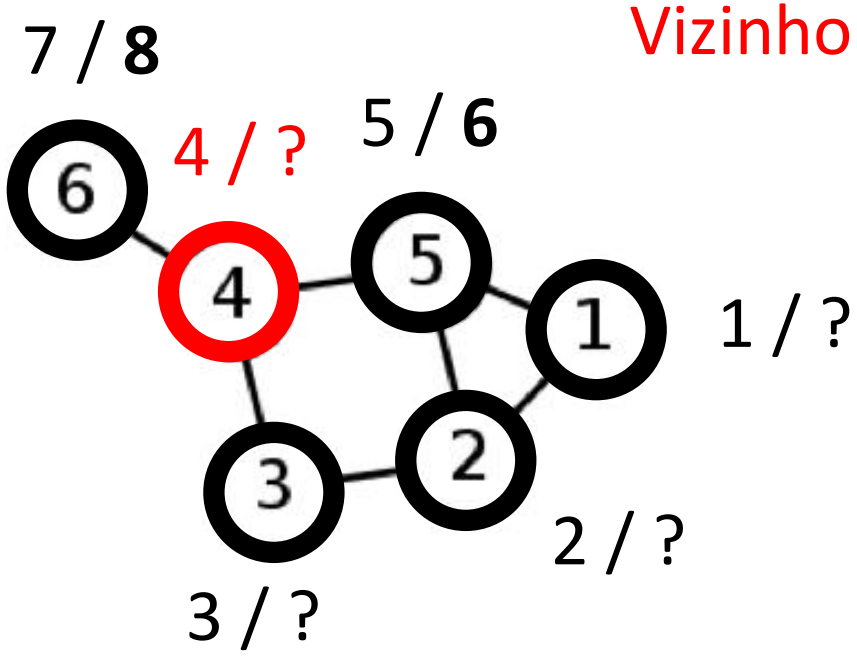
~~6~~
4
3
2
1



Sem. passada: “profundidade” (pilha)

topo

4
3
2
1

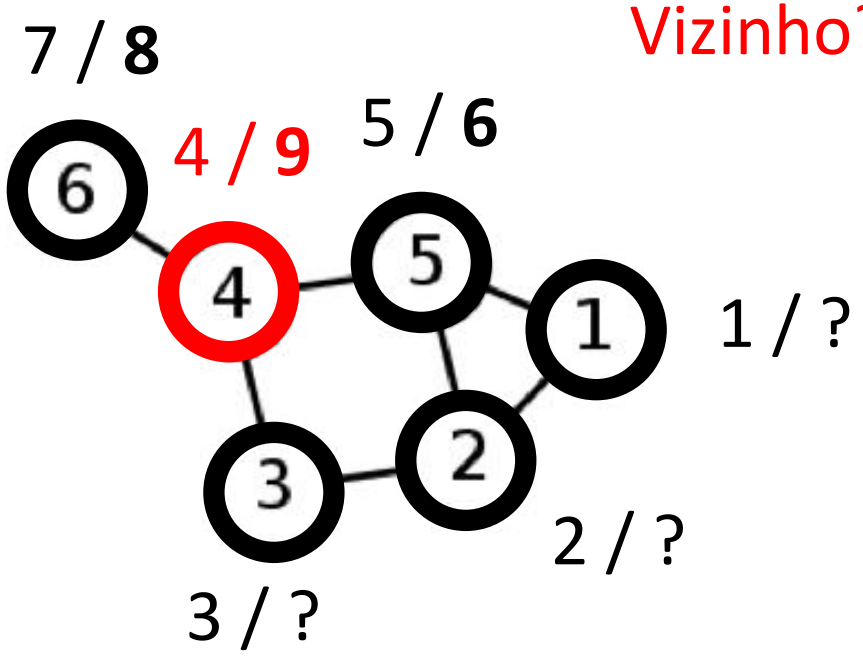


Vizinho???

Sem. passada: “profundidade” (pilha)

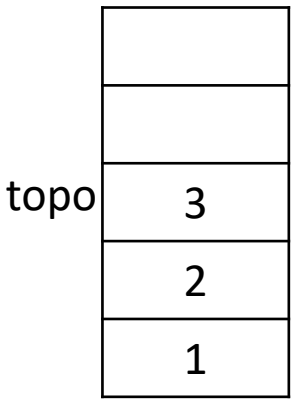
topo

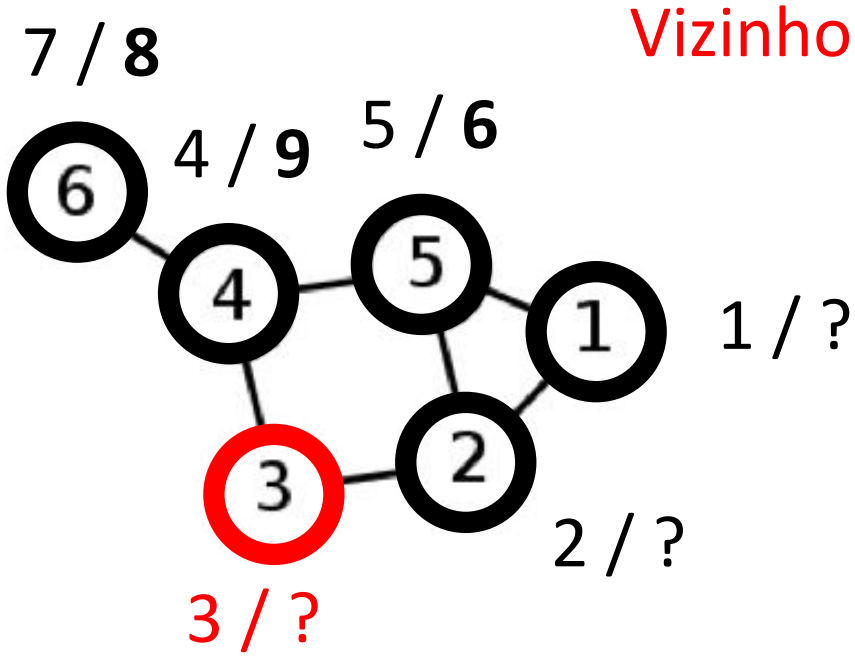
~~4~~
3
2
1



Vizinho???

Sem. passada: “profundidade” (pilha)

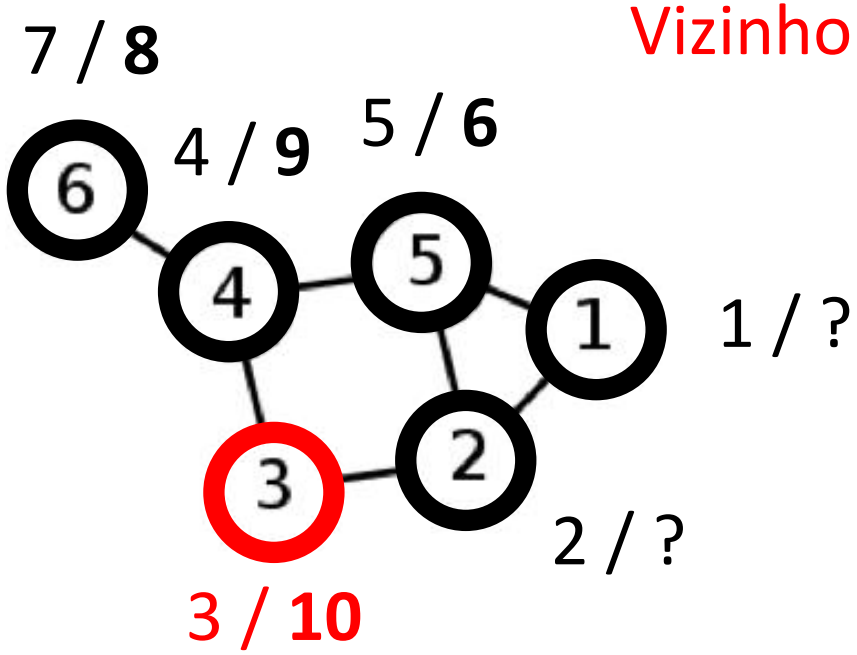




Sem. passada: “profundidade” (pilha)

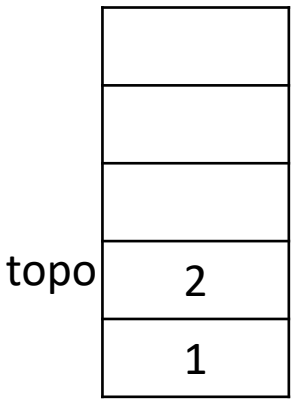
topo

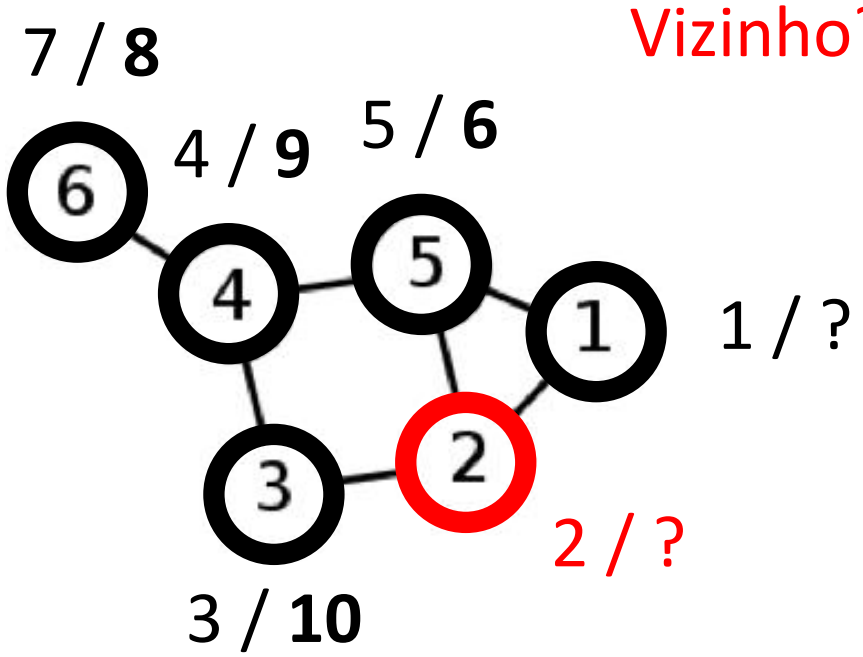
~~3~~
2
1



Vizinho???

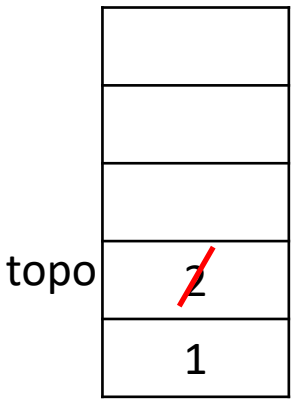
Sem. passada: “profundidade” (pilha)

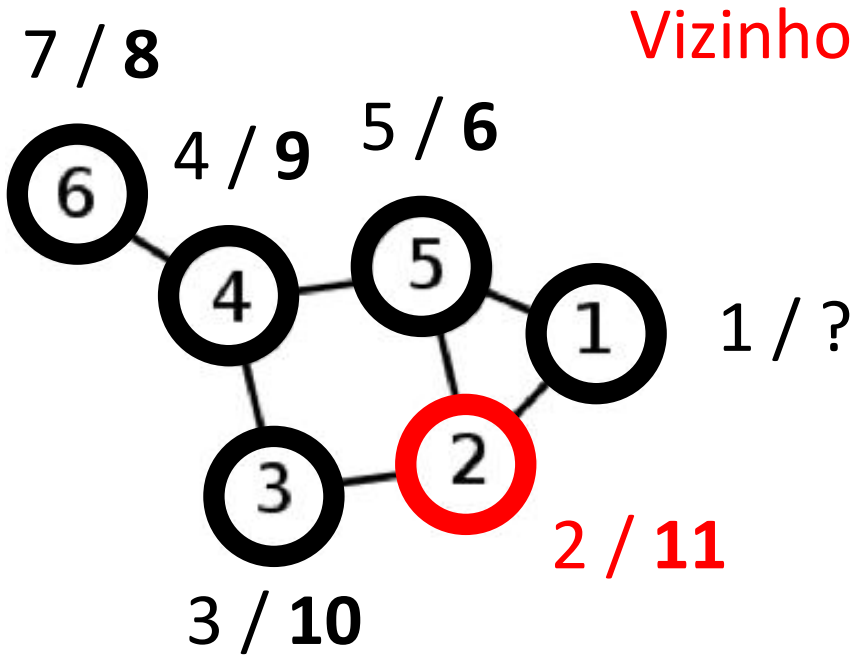




Vizinho???

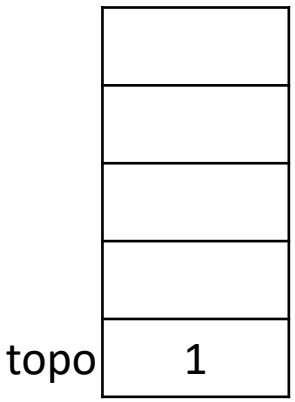
Sem. passada: “profundidade” (pilha)

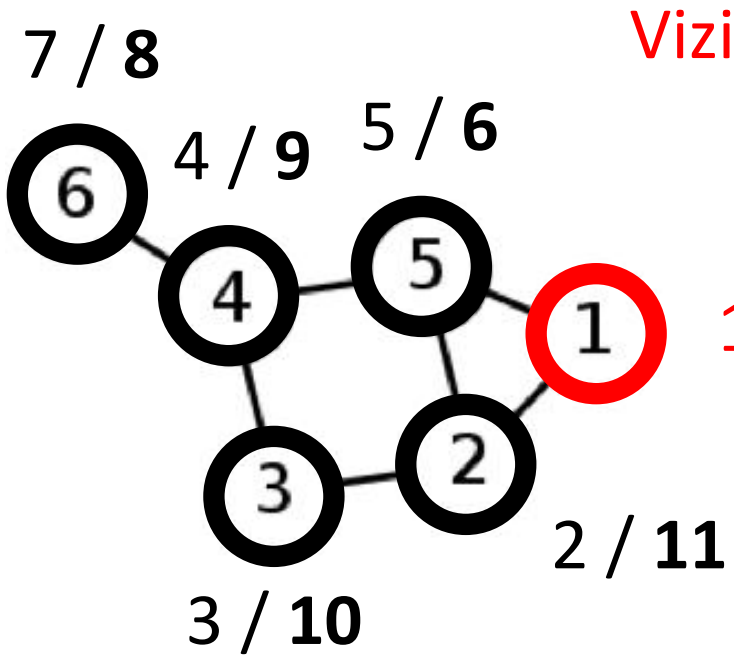




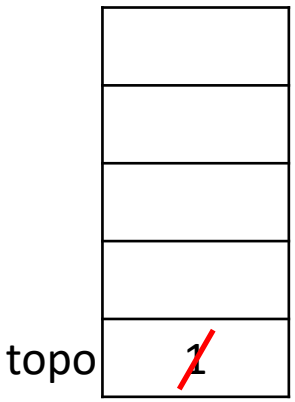
Vizinho???

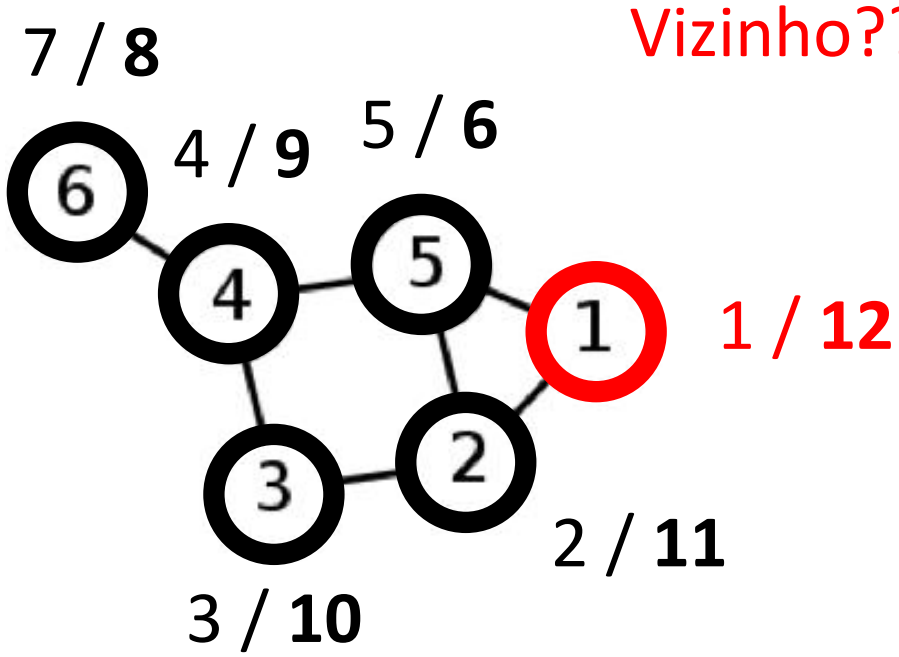
Sem. passada: “profundidade” (pilha)





Sem. passada: “profundidade” (pilha)





Vizinho???

“Aplicações”

- BFS: “matriz de distâncias” (largura)

- DFS: “expressão de parênteses” (profundidade)

- O que é distância?

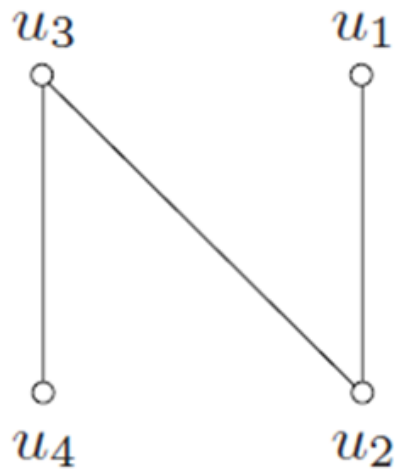
- O que é distância?

- Caminho mínimo?

Caminho

- É uma sequência de vértices conectados por arestas.

$$\langle u_1, u_2, u_3, u_4 \rangle$$

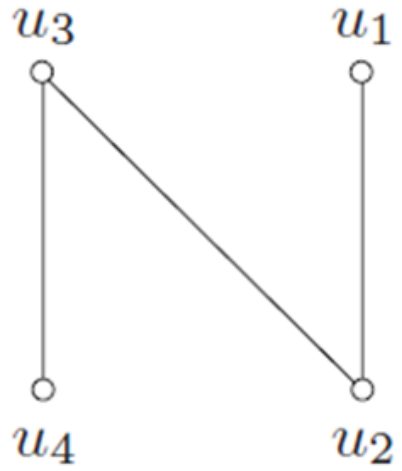


(a)

Caminho

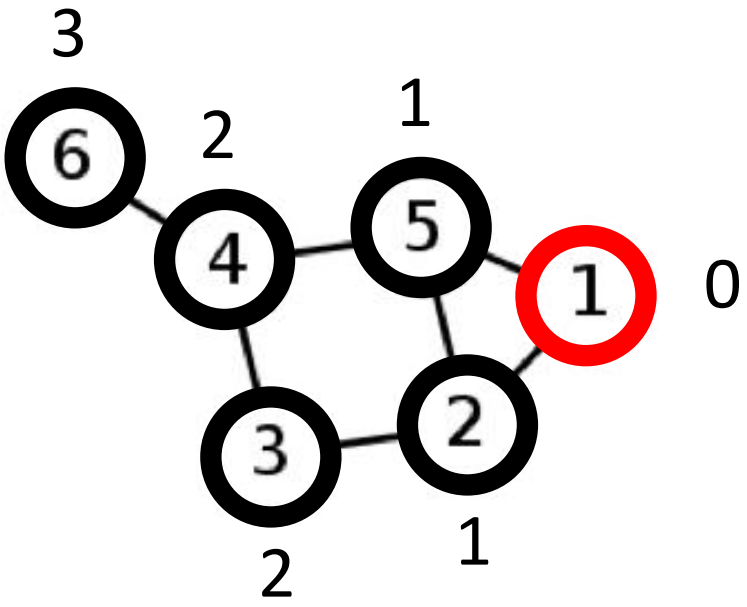
- O comprimento de um caminho é a qtde de arestas.
  - Ex. o comprimento do caminho abaixo vale 3

$$\langle u_1, u_2, u_3, u_4 \rangle$$



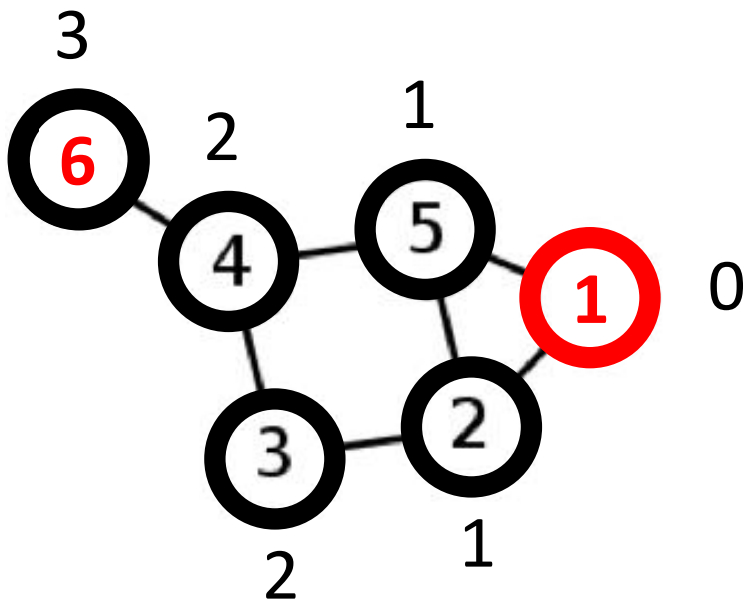
(a)

- Caminho mínimo
  - É um caminho de comprimento mínimo.

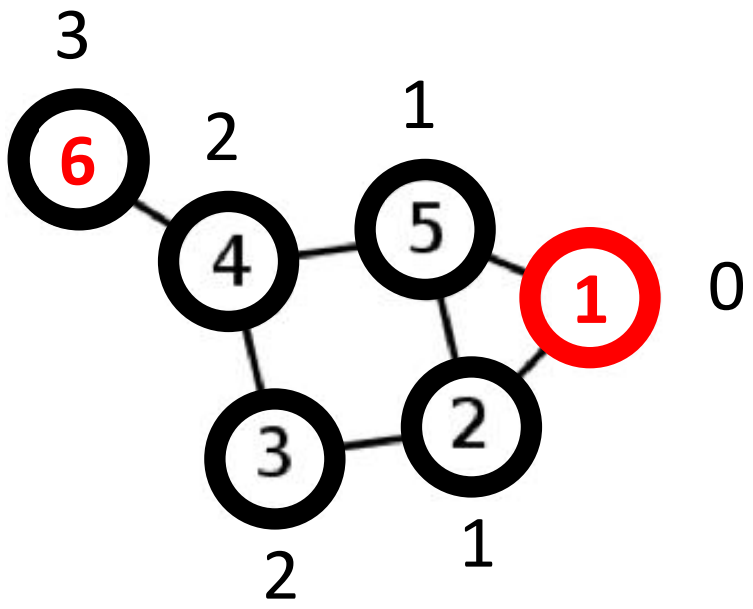


- Caminho mínimo
  - É um caminho de comprimento mínimo.

- Ex. qual seria um caminho mínimo de 1 até 6?

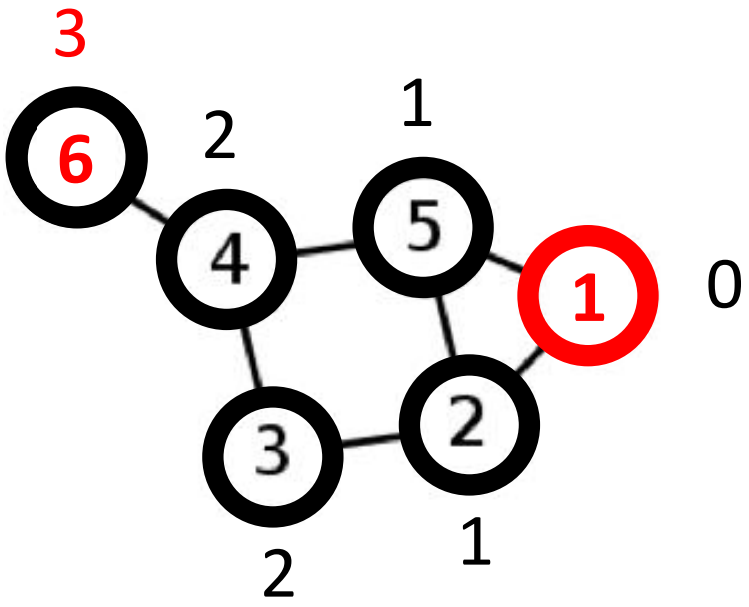


- Qual é a distância entre dois vértices, A e B?
  - É o comprimento de um caminho mínimo entre A e B.



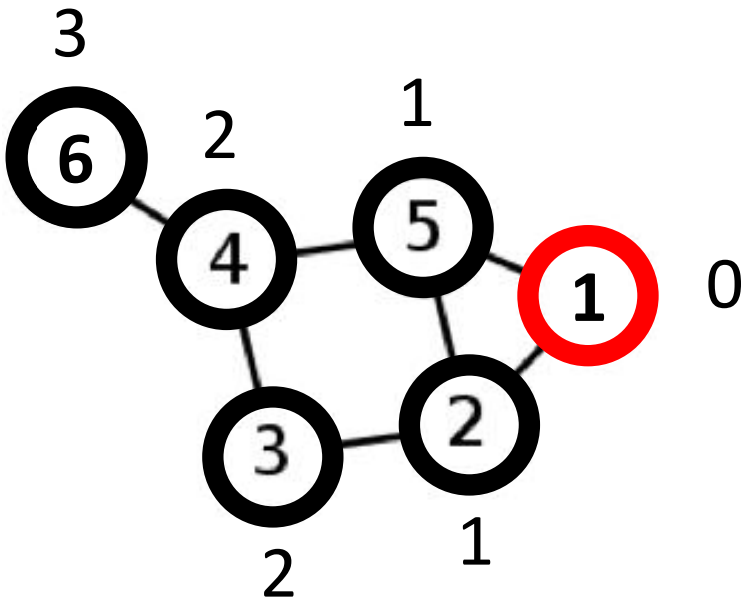
- Qual é a distância entre dois vértices, A e B?
  - É o comprimento de um caminho mínimo entre A e B.

- Ex. Qual é a distância entre 1 e 6?



Exercício

- Use o algoritmo BFS (largura) para calcular a matriz de distâncias.



	1	2	3	4	5	6
1	**0**	**1**	**2**	**2**	**1**	**3**
2						
3						
4						
5						
6						

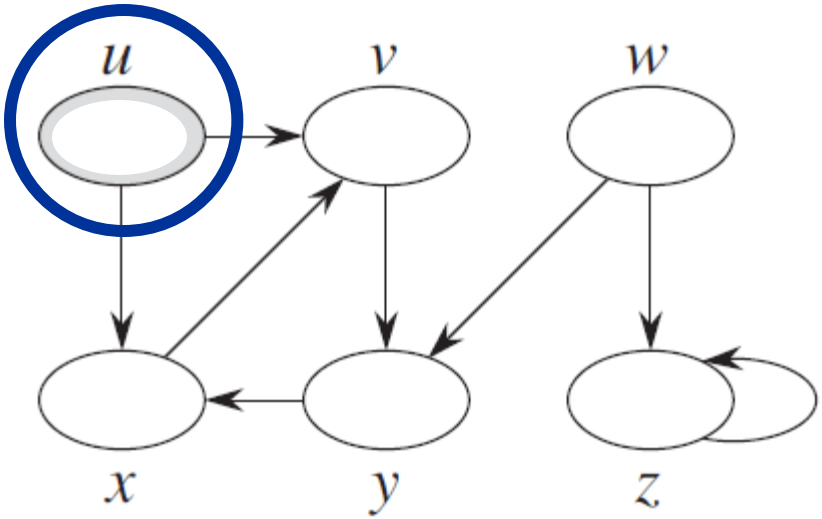
“Aplicações”

- BFS: “matriz de distâncias” (largura)

- DFS: “expressão de parênteses” (profundidade)

DFS (profundidade)

- Ideia: "descoberta" e "finalização"

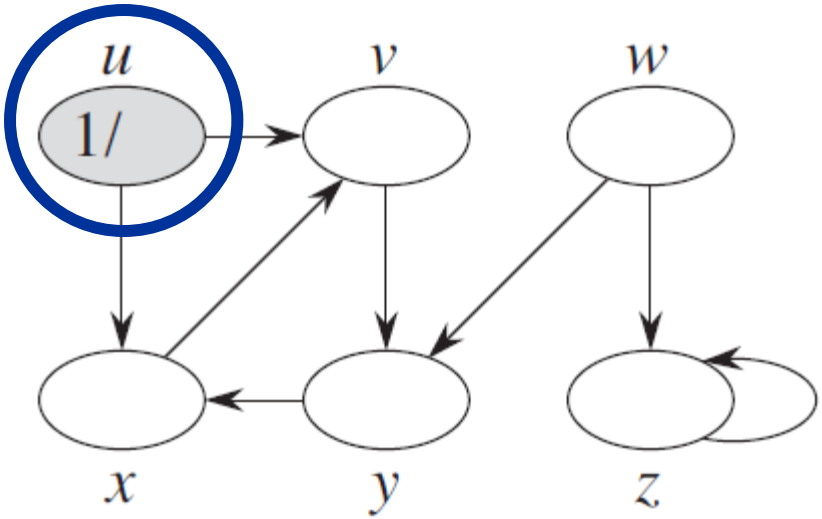


(Pilha)

S =

DFS (profundidade)

- Ideia: "descoberta" e "finalização"

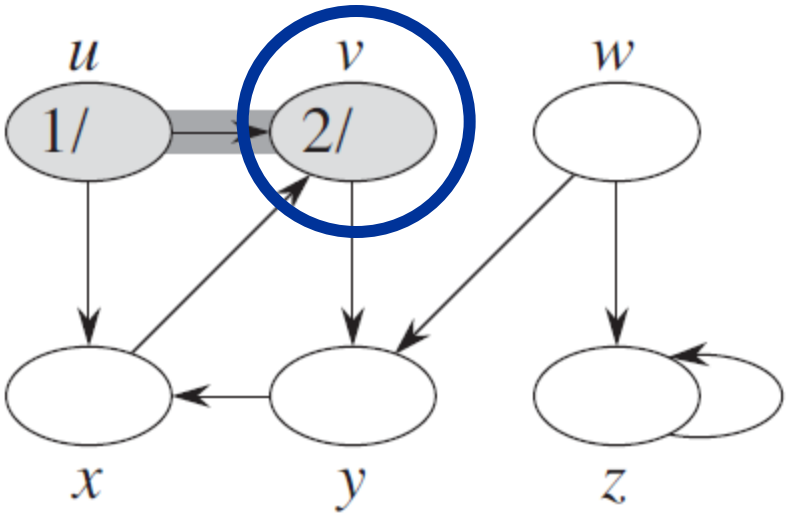


(Pilha)

S = u

DFS (profundidade)

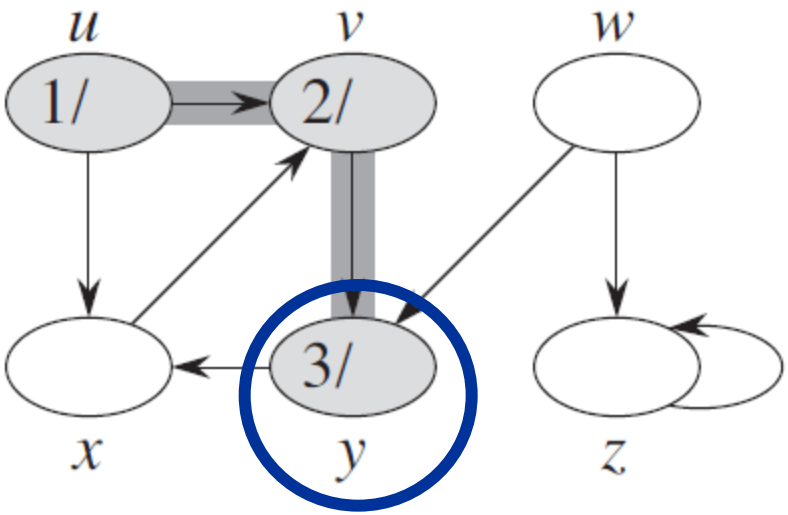
- Ideia: "descoberta" e "finalização"



S = u, v

DFS (profundidade)

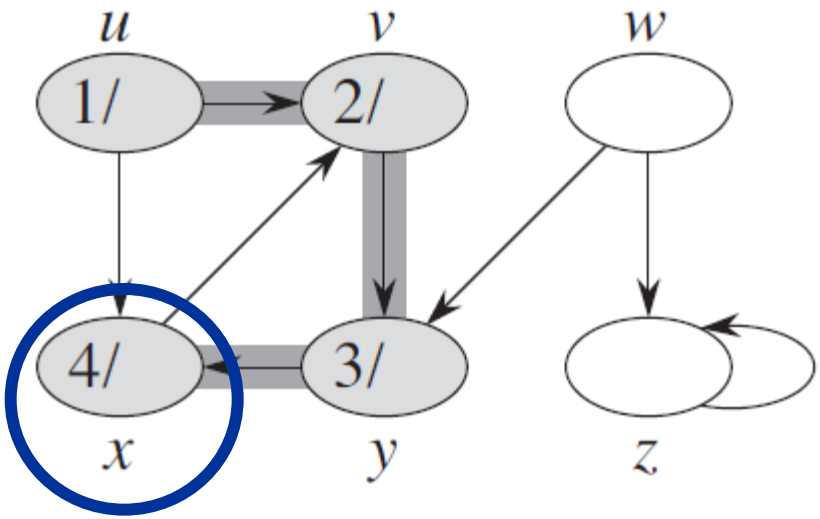
- Ideia: "descoberta" e "finalização"



S = u, v, y

DFS (profundidade)

- Ideia: "descoberta" e "finalização"

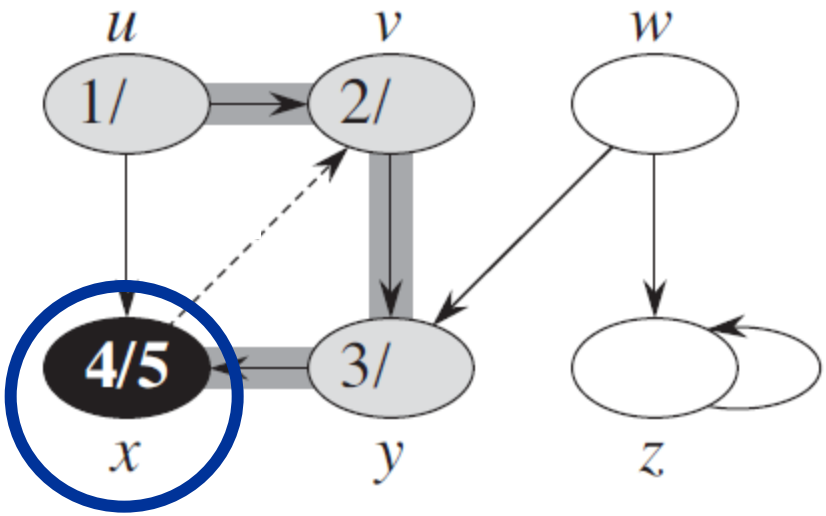


Topo

S = u, v, y, x

DFS (profundidade)

- Ideia: "descoberta" e "finalização"

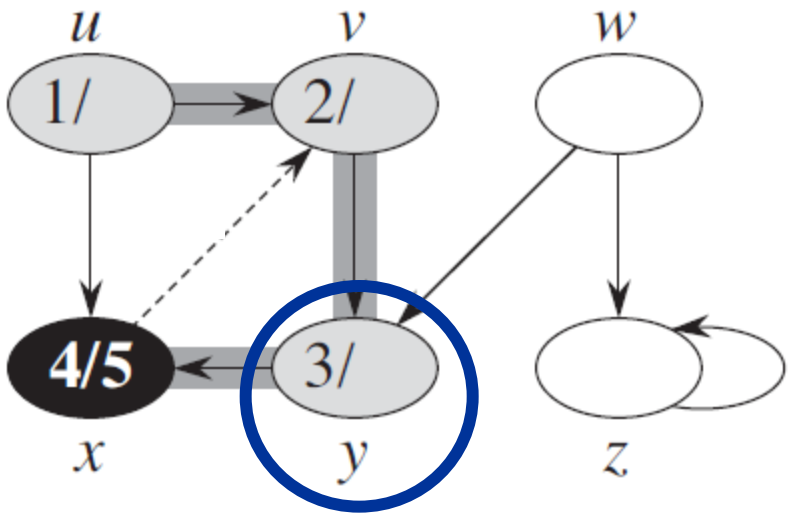


Topo

S = u, v, y, ~~x~~

DFS (profundidade)

- Ideia: "descoberta" e "finalização"

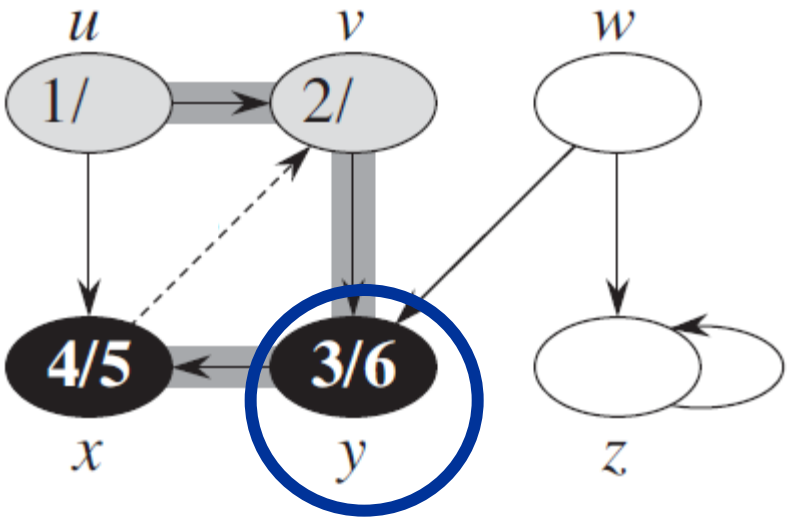


Topo

S = u, v, y

DFS (profundidade)

- Ideia: "descoberta" e "finalização"

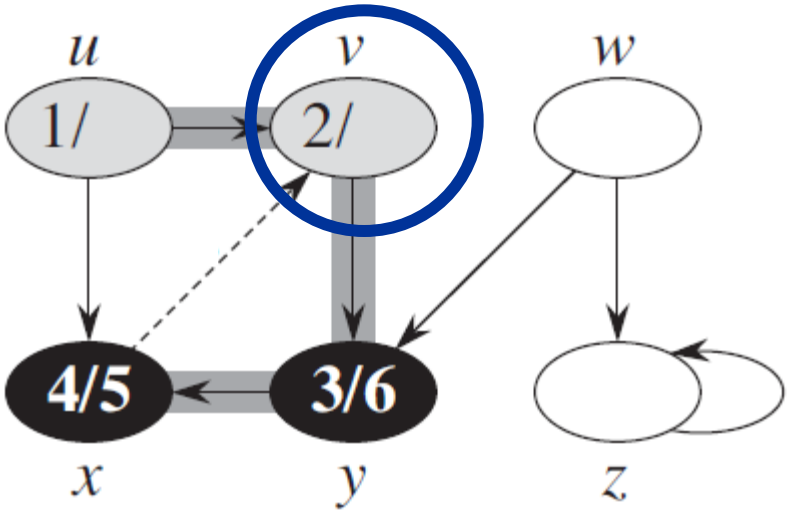


Topo

S = u, v, ~~y~~

DFS (profundidade)

- Ideia: "descoberta" e "finalização"

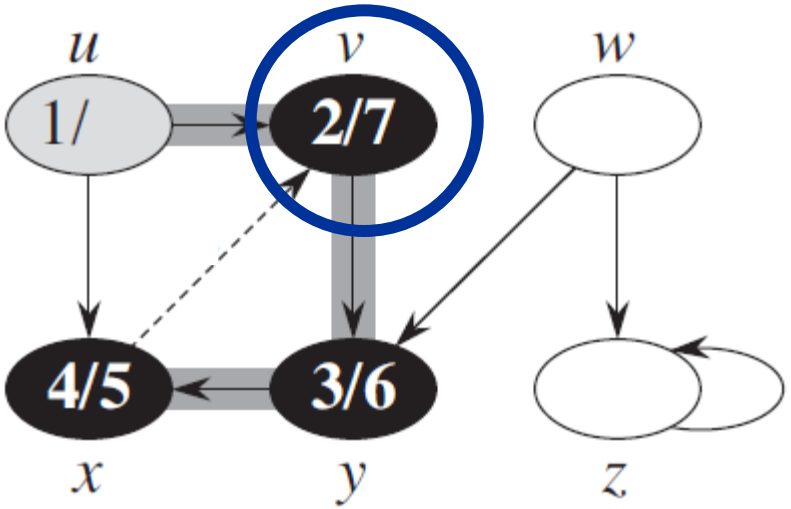


Topo

S = u, v

DFS (profundidade)

- Ideia: "descoberta" e "finalização"

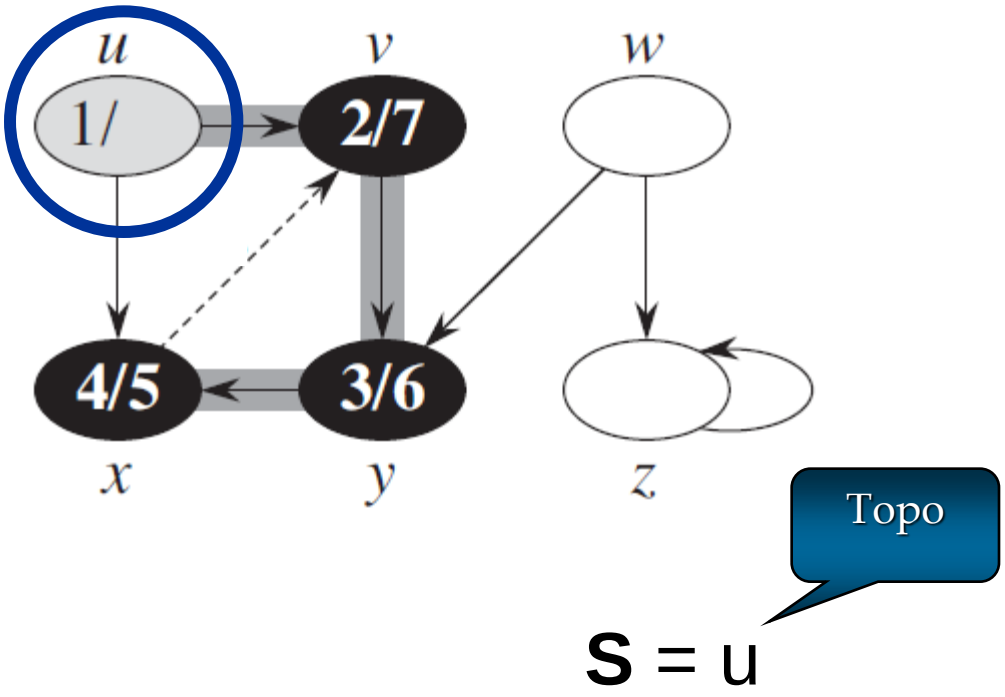


Topo

S = u, ~~v~~

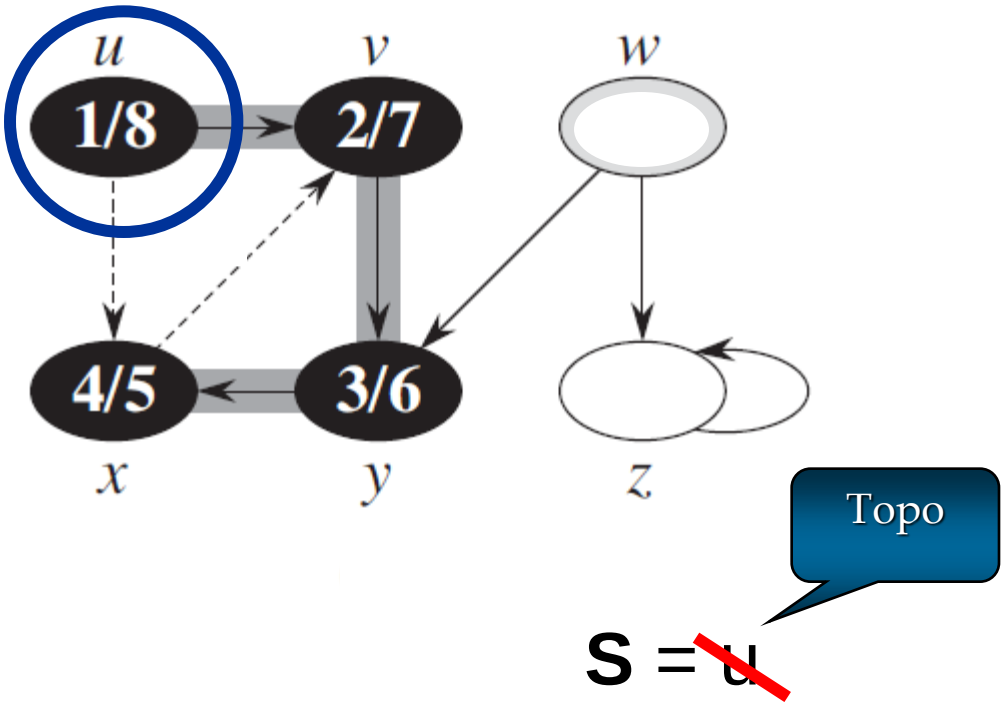
DFS (profundidade)

- Ideia: "descoberta" e "finalização"



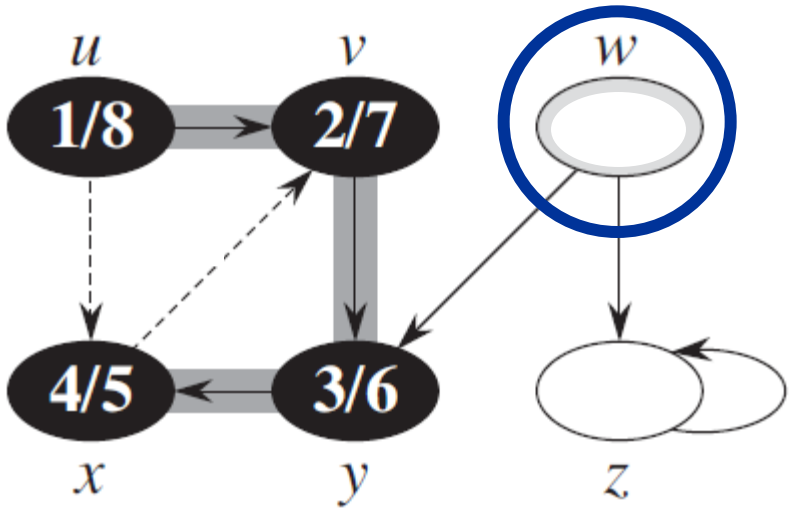
DFS (profundidade)

- Ideia: "descoberta" e "finalização"



DFS (profundidade)

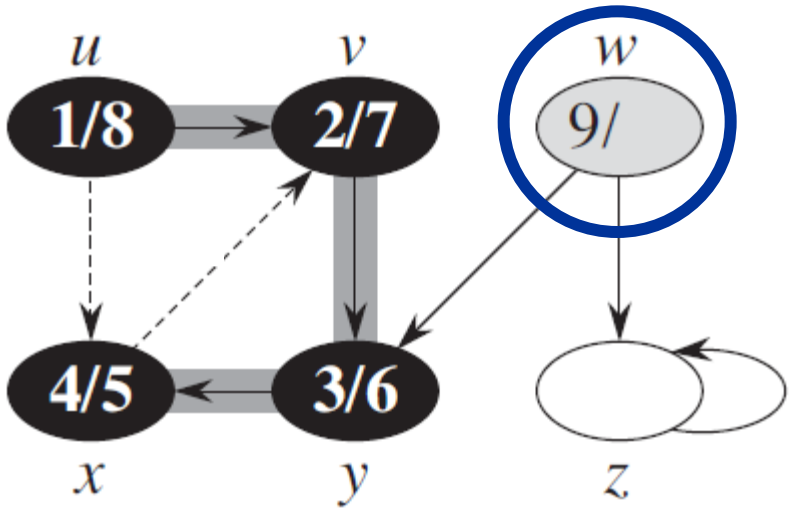
- Ideia: "descoberta" e "finalização"



S =

DFS (profundidade)

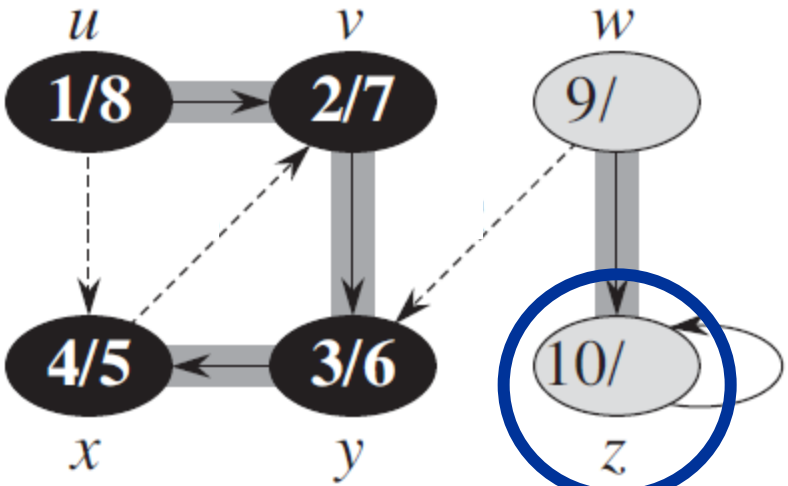
- Ideia: "descoberta" e "finalização"



s = w

DFS (profundidade)

- Ideia: "descoberta" e "finalização"

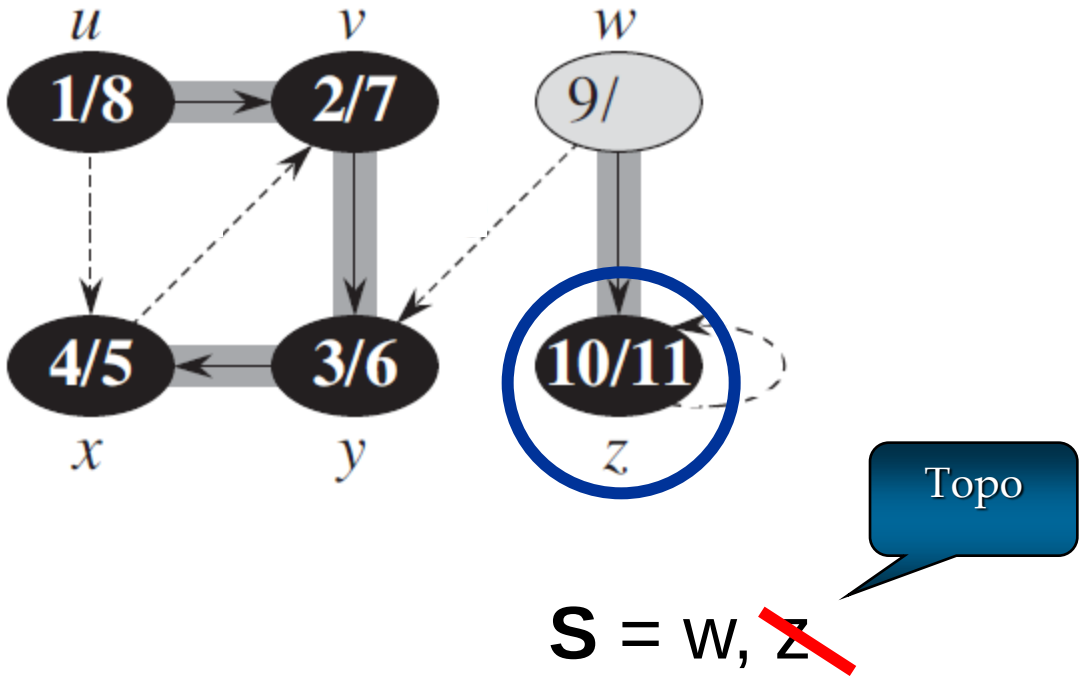


Topo

S = w, z

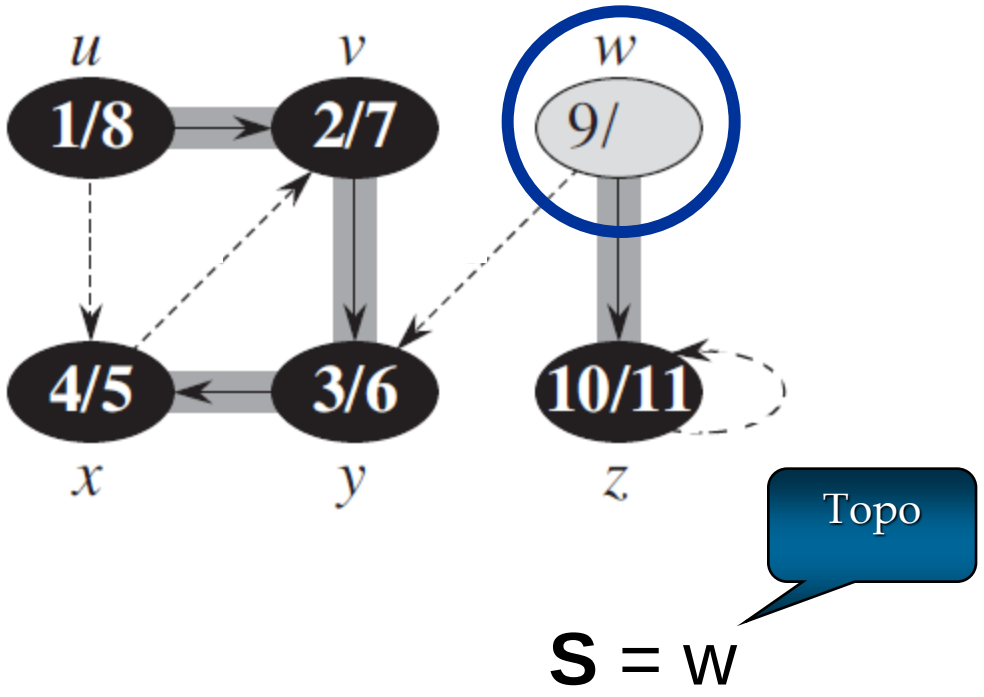
DFS (profundidade)

- Ideia: "descoberta" e "finalização"



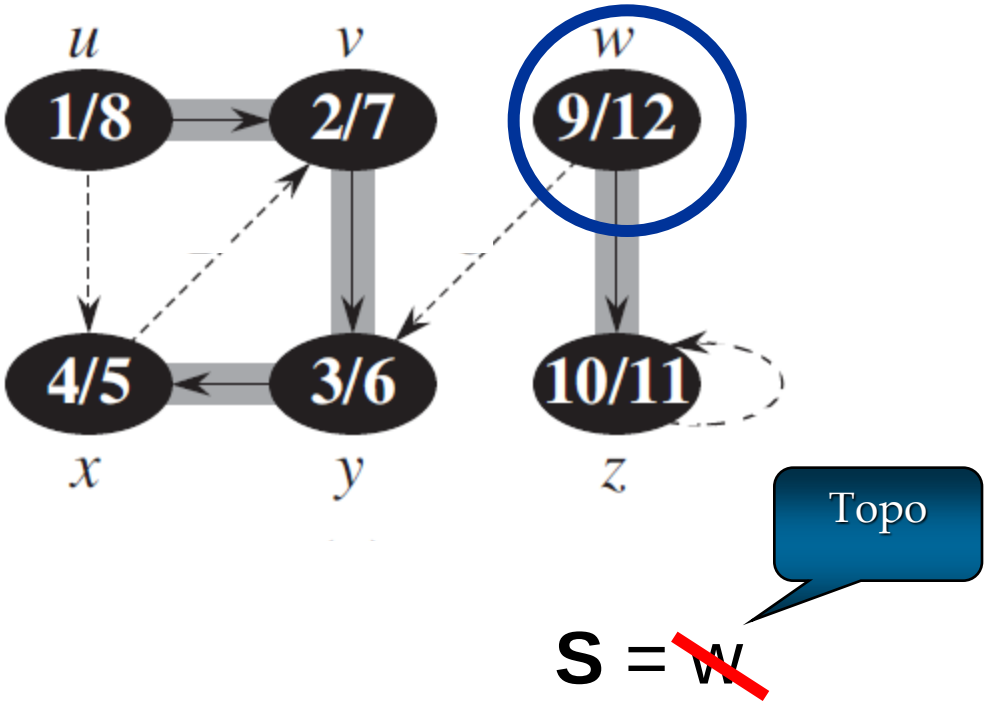
DFS (profundidade)

- Ideia: "descoberta" e "finalização"



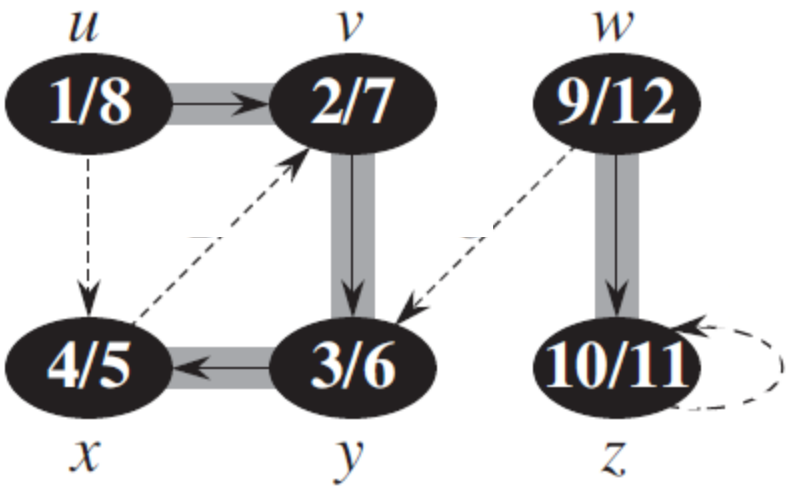
DFS (profundidade)

- Ideia: "descoberta" e "finalização"



DFS (profundidade)

- Expressão com parênteses:

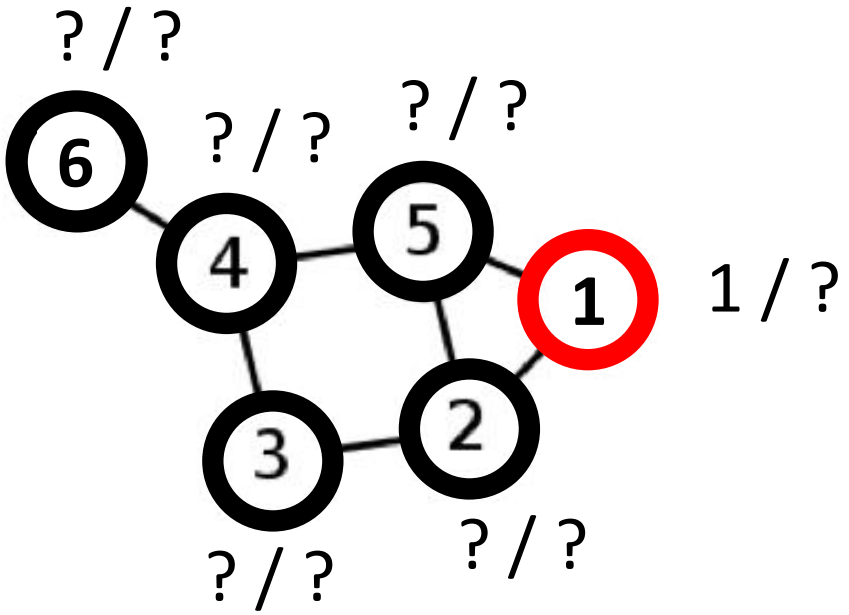


1 2 3 4 5 6 7 8 9 10 11 12

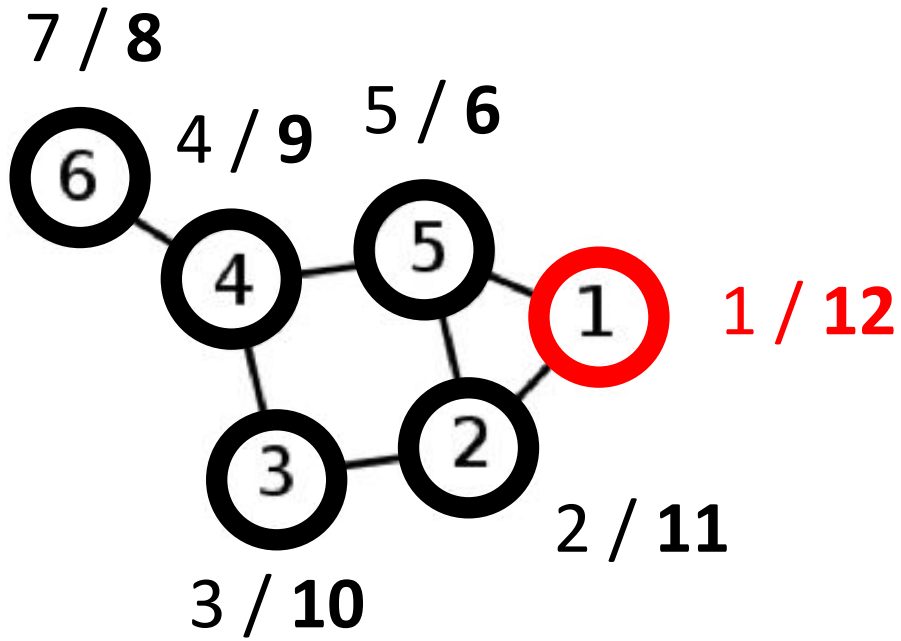
(u (v (y (x x) y) v) u) (w (z z) w)

Exercício

- Use o algoritmo DFS (profundidade) para calcular uma expressão de parênteses., escolhendo sempre o “menor vértice”.

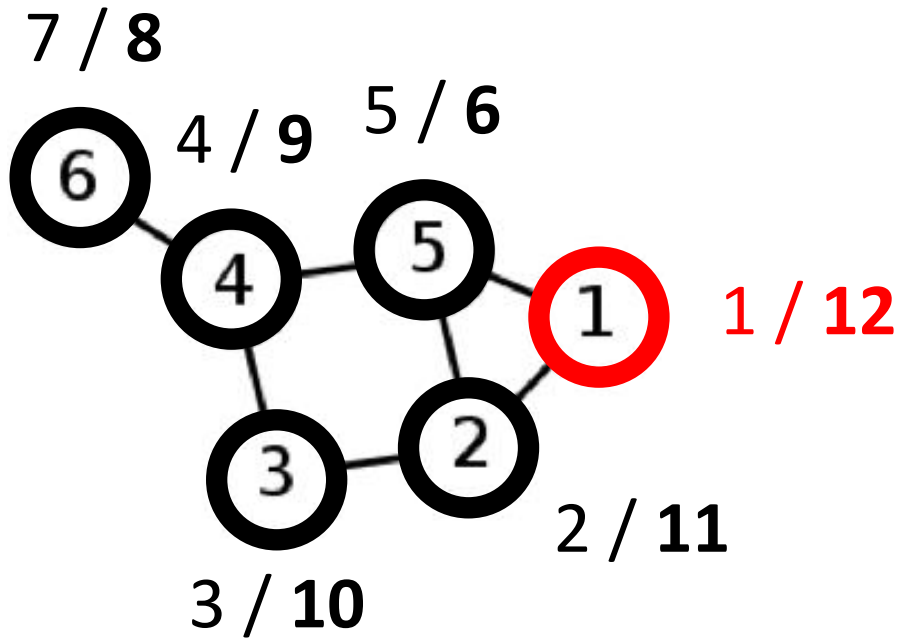


- Use o algoritmo DFS (profundidade) para calcular uma expressão de parênteses., escolhendo sempre o “menor vértice”.



1	2	3	4	5	6	7	8	9	10	11	12
?	?	?	?	?	?	?	?	?	?	?	?

- Use o algoritmo DFS (profundidade) para calcular uma expressão de parênteses., escolhendo sempre o “menor vértice”.

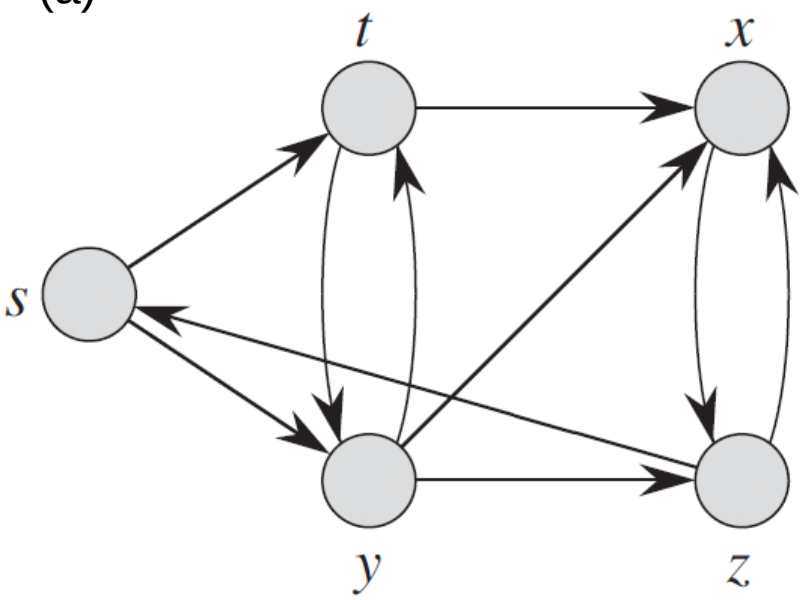


1	2	3	4	5	6	7	8	9	10	11	12
(1	(2	(3	(4	(5	5)	(6	6)	4)	3)	2)	1)

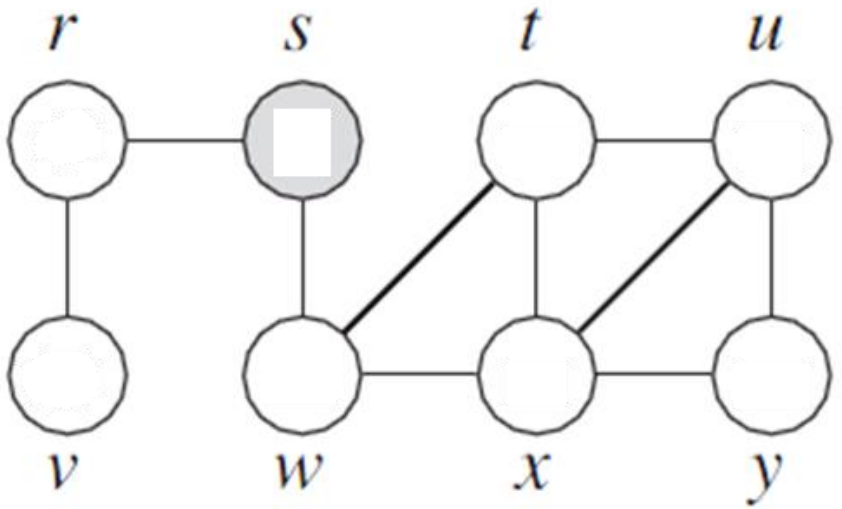
Exercícios

- Para cada grafo, calcule:
  - Matriz de distâncias
  - Expressão de parênteses

(a)



(b)



Matriz de distâncias

- Consumo de tempo?

```
                                                    Consumo
MatrizDeDistancias(G)                               de tempo:
 1 matrizDist = []                                    ???
 1 para cada vértice u_i em G.V faça                  ???
 2     matrizDist.append(bfs(G, u_i))                 ???
 3 return matrizDist                                  ???

Total:                                        T(n,m) = ???
```

Matriz de distâncias

- Consumo de tempo?

```
                                                    Consumo
MatrizDeDistancias(G)                               de tempo:
 1 matrizDist = []                                    O(n)
 1 para cada vértice u_i em G.V faça                  O(n)
 2     matrizDist.append(bfs(G, u_i))                 O(n)*O(n+m)
 3 return matrizDist                                  O(1)

Total:                                  T(n,m) = O(n*n + n*m)
```

Exercício Programa

- 04-bfs-matrizDeDistancias.py

Expressão de parênteses

- Consumo de tempo?
  - **Descoberta**: abre parênteses
  - **Finalização**: fecha parênteses

Expressão de parênteses

- Consumo de tempo? $T(n, m) = O(n + m)$
  - **Descoberta**: abre parênteses
  - **Finalização**: fecha parênteses

Exercício Programa

- 05-dfs-parenteses.py