

# ALGORITMOS EM GRAFOS

CAMINHAMENTOS

BUSCA EM LARGURA

Prof. João Caram

# Busca sem informação (busca cega)

2

- **Busca sem informação**: usa apenas a informação disponível no estado atual do problema.
  - Geração de *sucedessores* e verificação do objetivo
- Exemplos:
  - Jarras d'água
  - Jogo da velha
  - Xadrez

# Propriedades de grafos

3

- Algumas propriedades de grafos são de simples verificação:
  - Verificação de graus dos vértices
  - Determinação se o grafo é euleriano
  - Determinação se o grafo é completo
- Outras propriedades são relacionadas às arestas e aos caminhos existentes

# Caminhamentos

4

- *Caminhar* em um grafo é mover-se entre seus vértices, verificando propriedades enquanto se caminha

# Caminhamentos

5

- Algoritmos de *busca em grafos* procuram caminhos com objetivos específicos:

*Conectividade*      *Busca de um vértice específico (estado)*

*Caminho mínimo*      *Existência de um caminho*

# Busca em grafos

6

- A busca em grafos tenta encontrar uma sequência de caminhos/ações que leve até a um objetivo
- Uma vez encontrado este objetivo, um programa pode executar tal sequência de ações para atingi-lo

# Busca em grafos

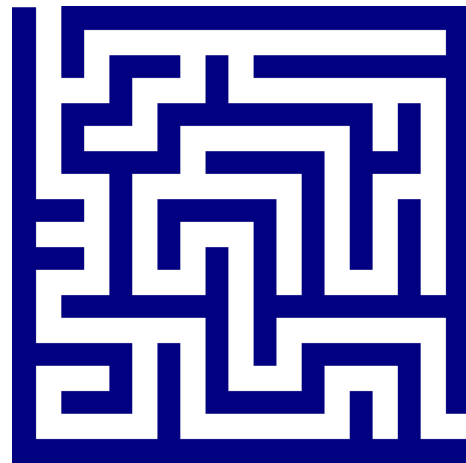
7

- Aplicações
  - Rotas em redes de computadores
  - Caixeiro viajante e variações
  - Jogos digitais
  - Navegação de robôs
  - ...

# Labirinto

8

- Considere o problema de encontrar uma saída em um labirinto
- Modelando cada “sala” como um vértice e suas conexões como arestas, temos um problema de busca em grafos





# Busca em largura

9

- Em inglês, *Breadth First Search* (BFS)
- Consiste em, a partir de um vértice de origem, explorar primeiramente todos os seus vizinhos e, em seguida repetir o procedimento para cada vizinho
- Base para diversos algoritmos importantes que iremos estudar

# Busca em largura

10

- Calcula a distância (caminho mínimo) do vértice de origem até qualquer vértice que possa ser alcançado
- Produz uma árvore que indica todos os vértices que podem ser alcançados
- Funciona em grafos e digrafos

# Busca em largura

11

- Propriedades de um vértice
  - Antecessor ou pai
  - Estado: branco, cinza, preto
  - Distância até o vértice de origem

# Busca em largura

12

- Estados dos vértices
  - Branco: ainda não explorado
  - Cinza: explorado, mas com vizinhos não-explorados
  - Preto: explorado e sem vizinhos não explorados

# Busca em largura

13

- Utiliza uma lista para definir as próximas visitas
- Pode armazenar a árvore de busca e/ou a sequência percorrida até um objetivo

# Algoritmo BFS - inicialização

Para cada vértice  $u$  diferente da origem  $s$  faça

$u.cor = \text{branco};$

$u.distância = \text{max\_value};$

$u.pai = \text{null};$

Fim para

$s.cor = \text{cinza};$

$s.distância = 0;$

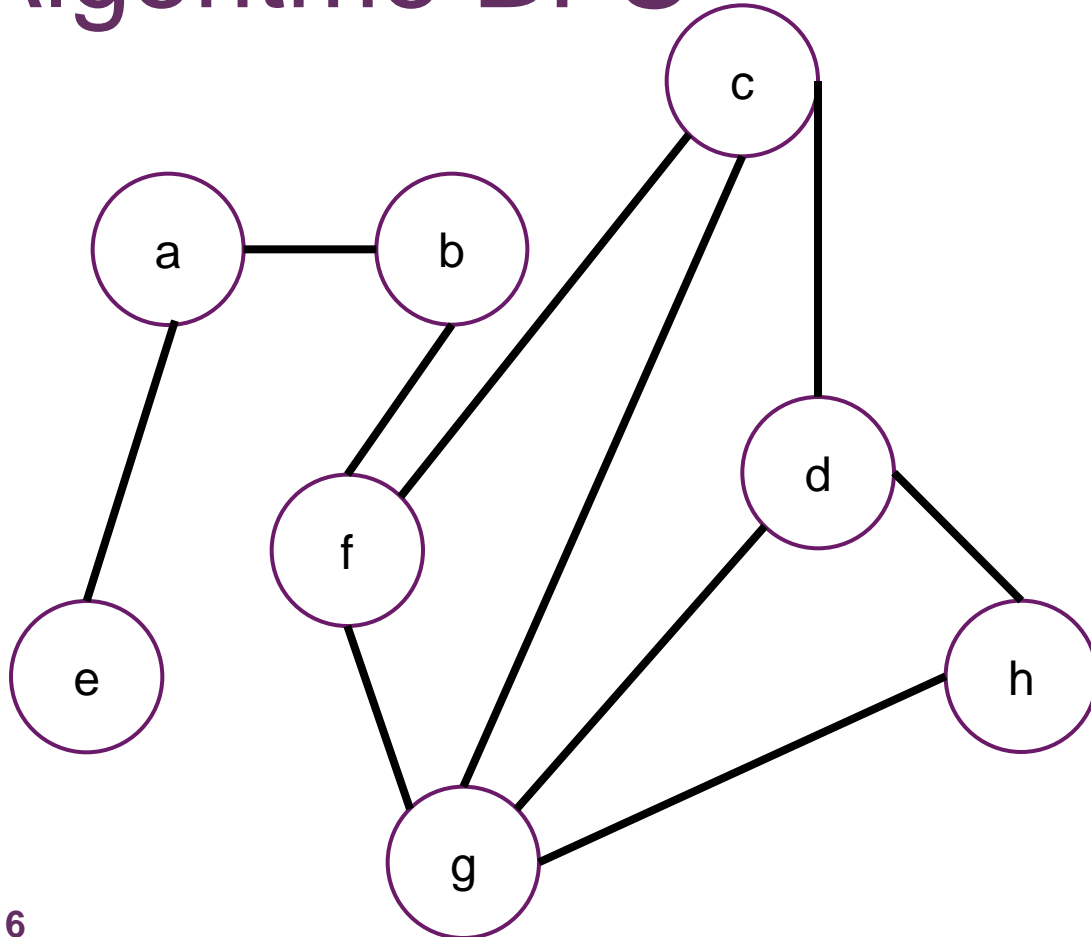
$u.pai = \text{null};$

$Q = \text{nova Fila vazia};$

# Algoritmo BFS – busca principal

```
Q.enqueue(s);
Enquanto (!Q.vazia)
    u = Q.dequeue();
    Para cada vértice v adjacente a u
        se v.cor == branco
            v.cor == cinza;
            v.distância = u.distância+1;
            v.pai = u
            Q.enqueue(v)
    Fim para
    u.cor = preto;
Fim enquanto
```

# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
-	-	-	-	-	-	-	-

Pais

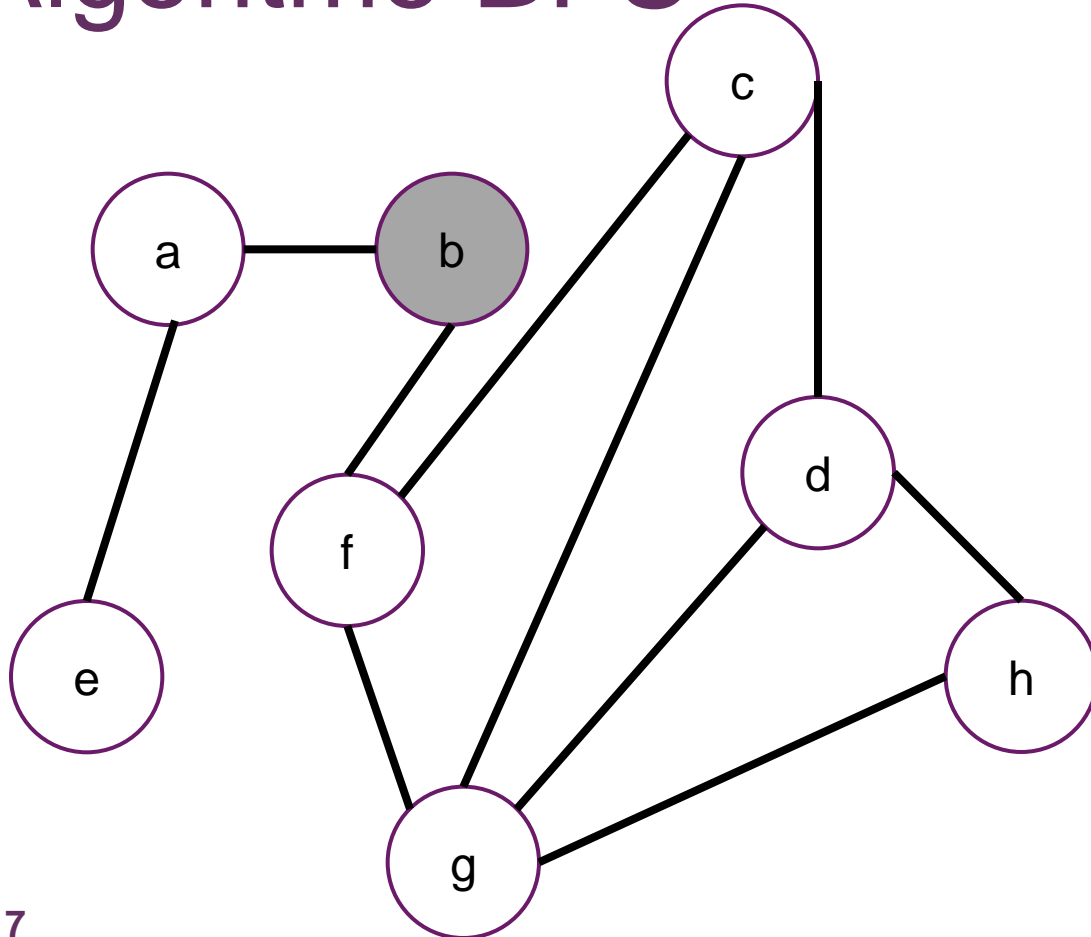
a	b	c	d	e	f	g	h
-	-	-	-	-	-	-	-

Fila Q

--	--	--	--	--	--	--	--



# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
-	0	-	-	-	-	-	-

Pais

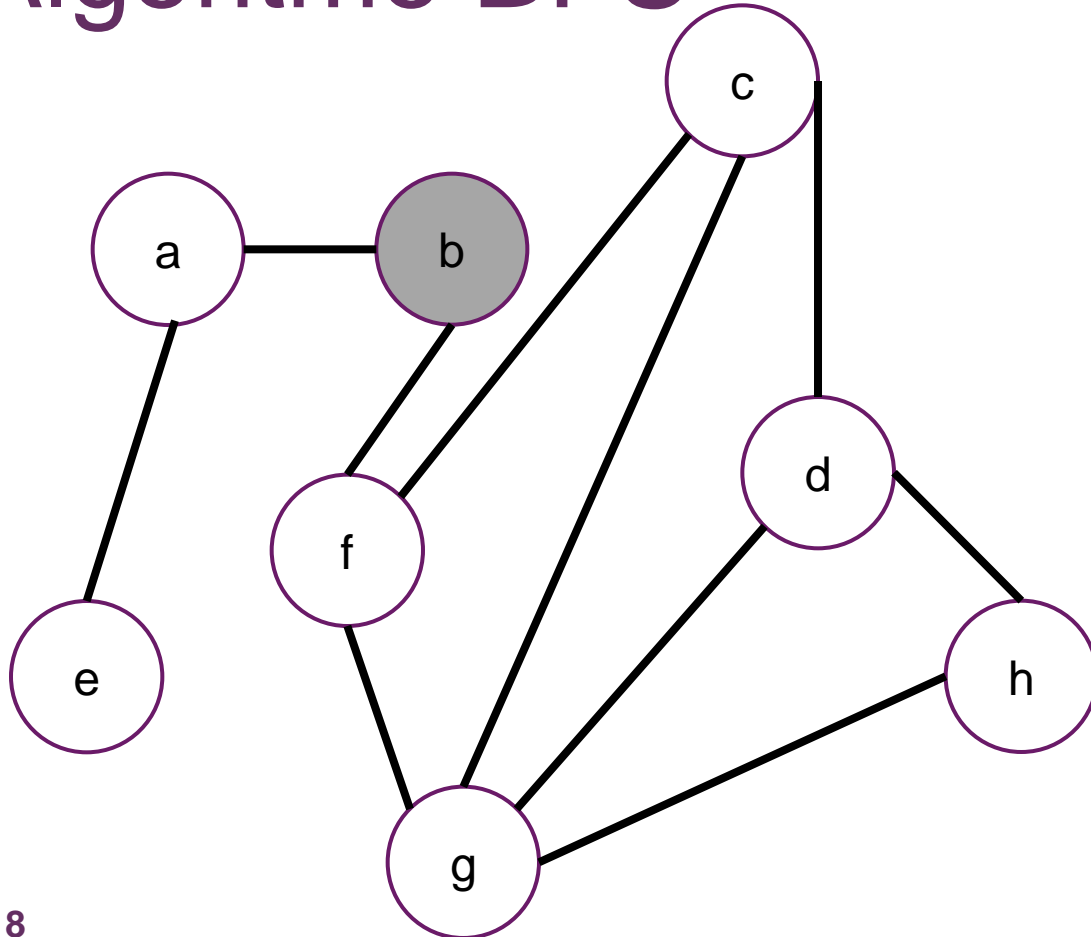
a	b	c	d	e	f	g	h
-	-	-	-	-	-	-	-

Fila Q

b							
---	--	--	--	--	--	--	--

Vértice:

# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
-	0	-	-	-	-	-	-

Pais

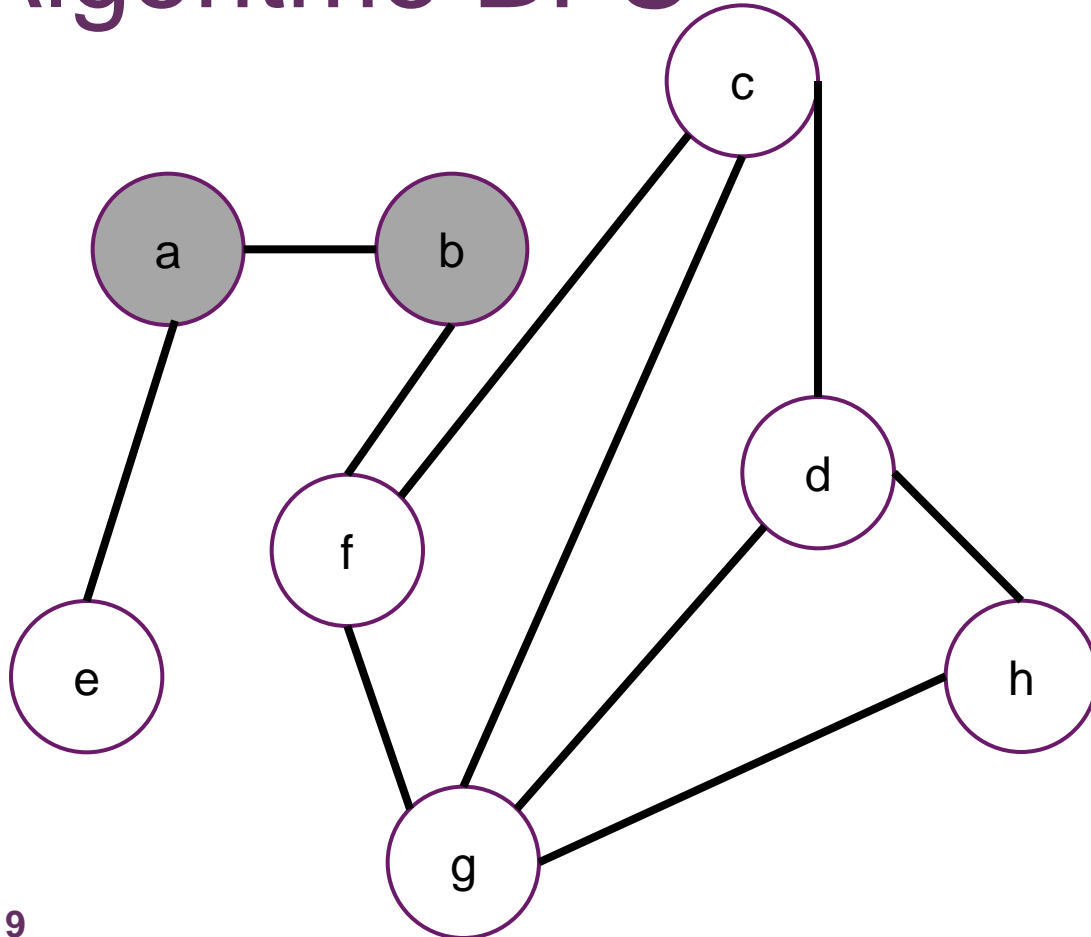
a	b	c	d	e	f	g	h
-	-	-	-	-	-	-	-

Fila Q

--	--	--	--	--	--	--	--

Vértice: b

# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
1	0	-	-	-	-	-	-

Pais

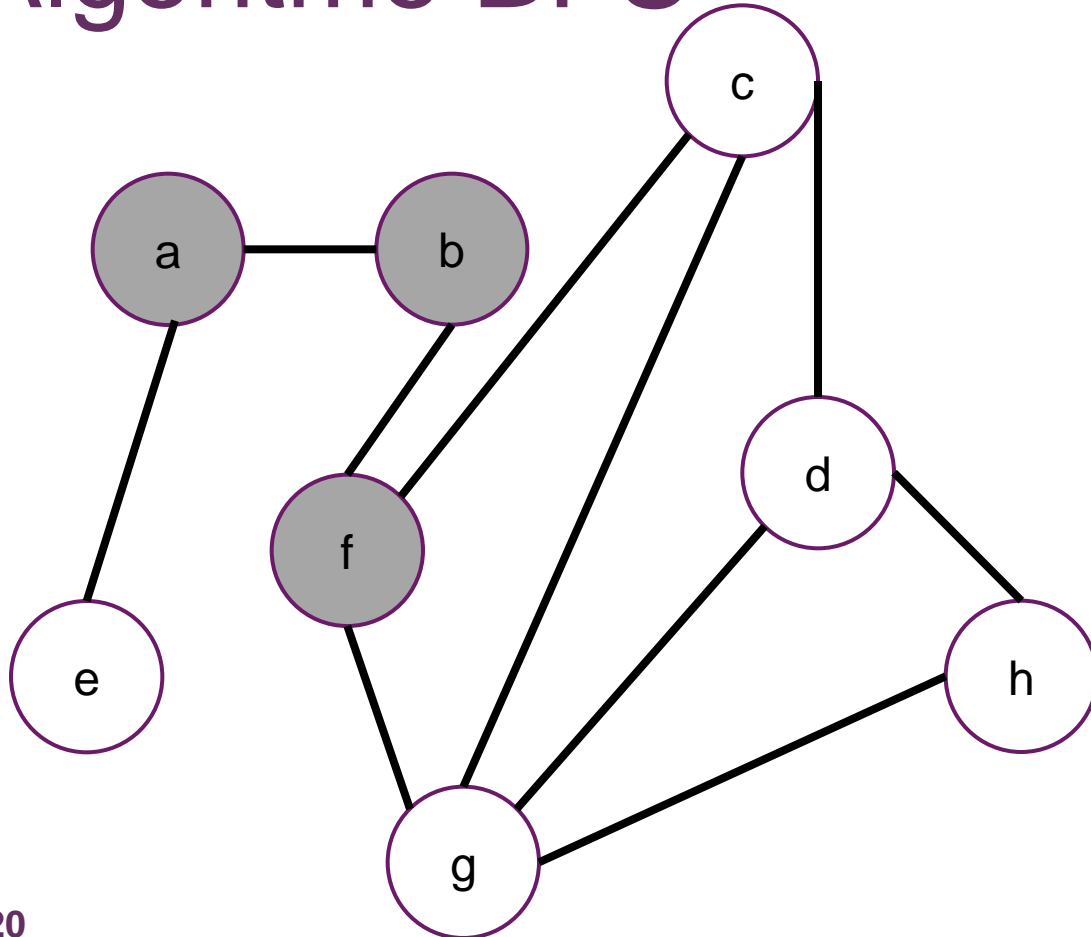
a	b	c	d	e	f	g	h
b	-	-	-	-	-	-	-

Fila Q

a							
---	--	--	--	--	--	--	--

Vértice: b

# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
1	0	-	-	-	1	-	-

Pais

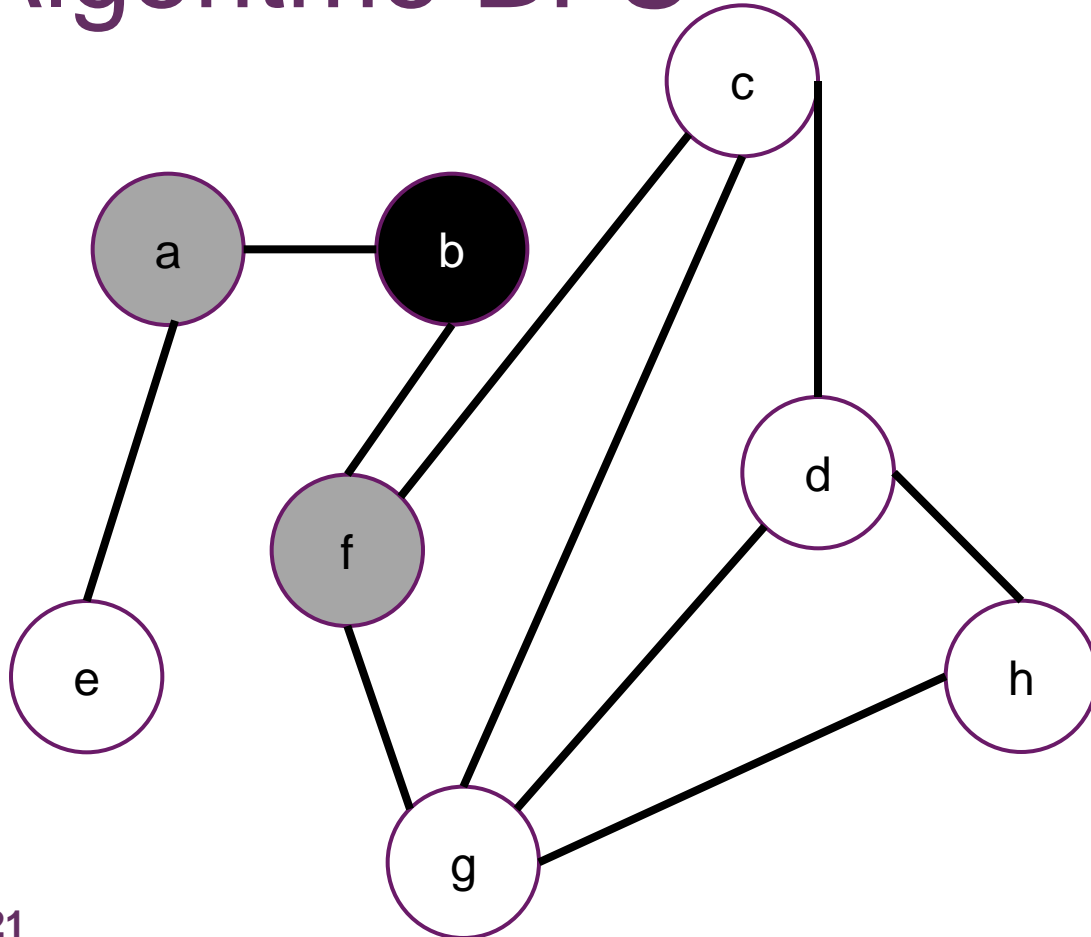
a	b	c	d	e	f	g	h
b	-	-	-	-	b	-	-

Fila Q

a	f						
---	---	--	--	--	--	--	--

Vértice: b

# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
1	0	-	-	-	1	-	-

Pais

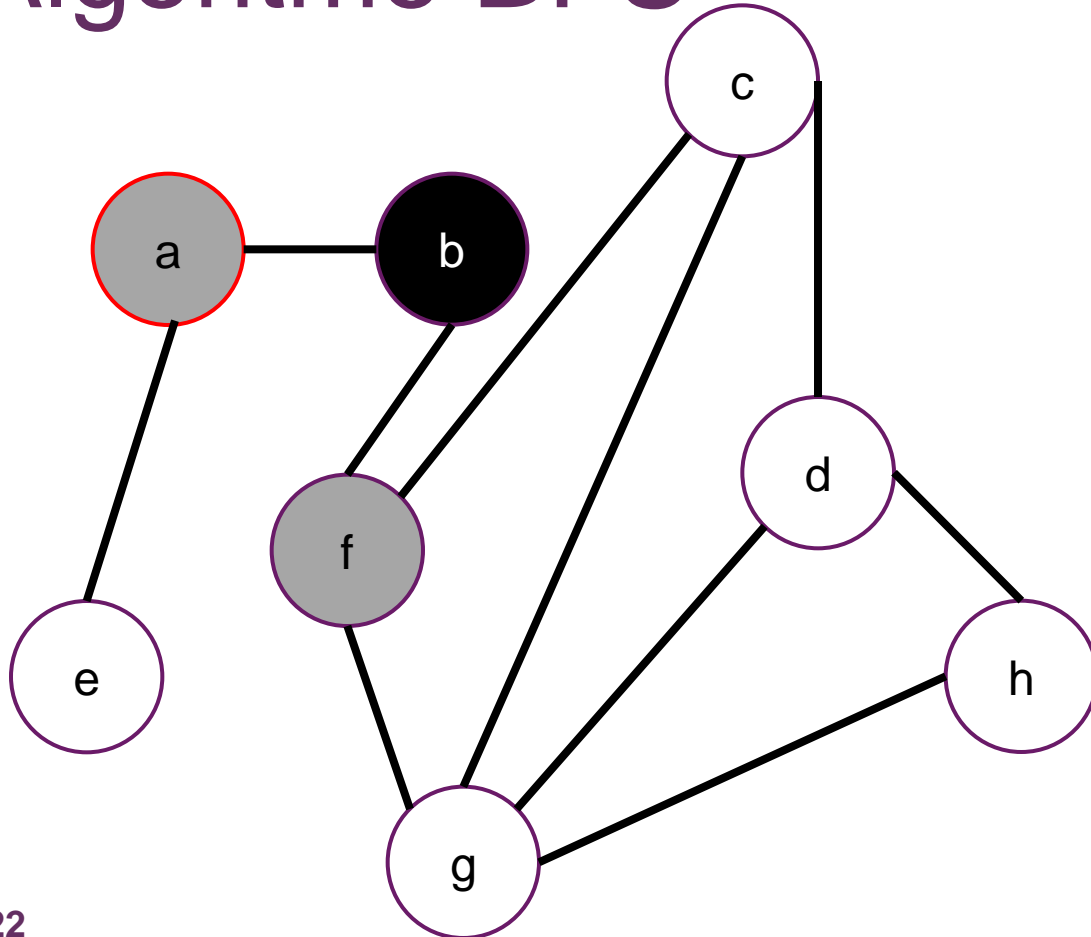
a	b	c	d	e	f	g	h
b	-	-	-	-	b	-	-

Fila Q

a	f						
---	---	--	--	--	--	--	--

Vértice: b

# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
1	0	-	-	-	1	-	-

Pais

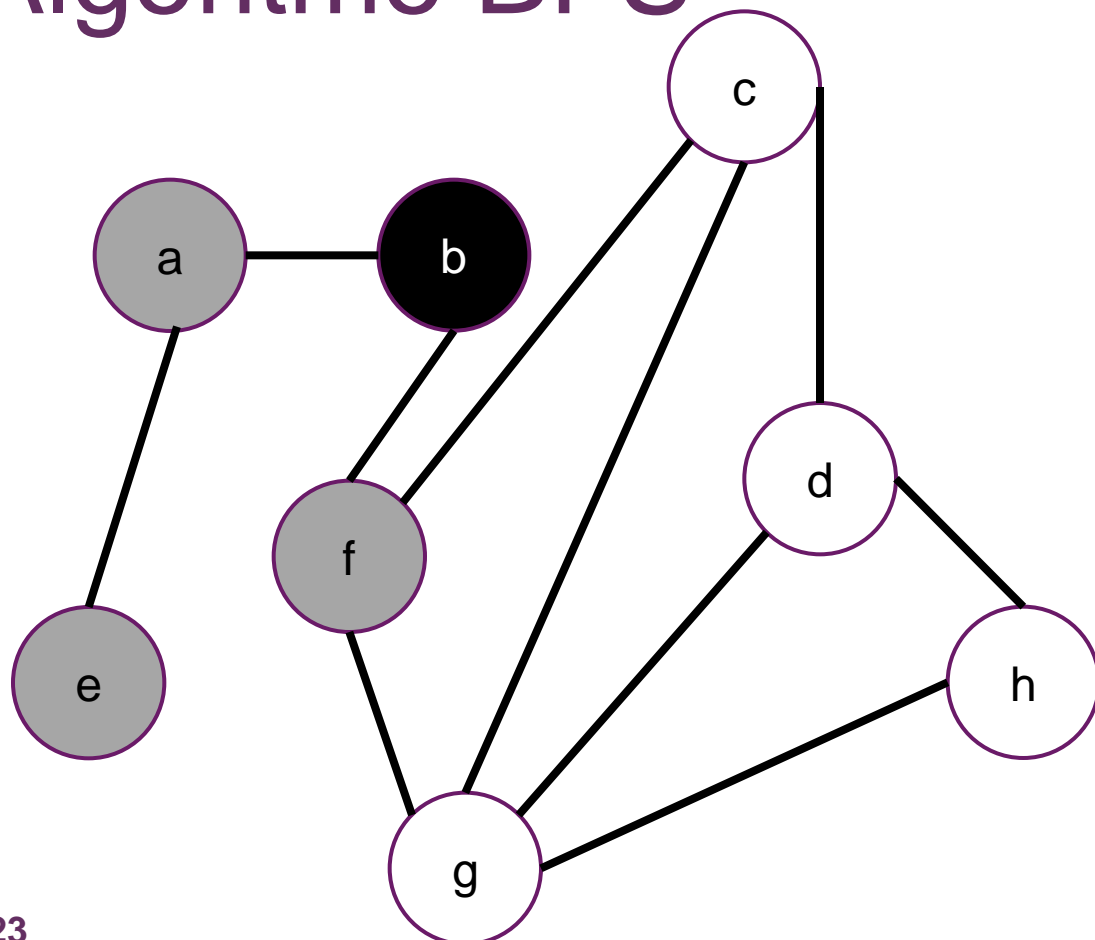
a	b	c	d	e	f	g	h
b	-	-	-	-	b	-	-

Fila Q

f							
---	--	--	--	--	--	--	--

Vértice: a

# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
1	0	-	-	2	1	-	-

Pais

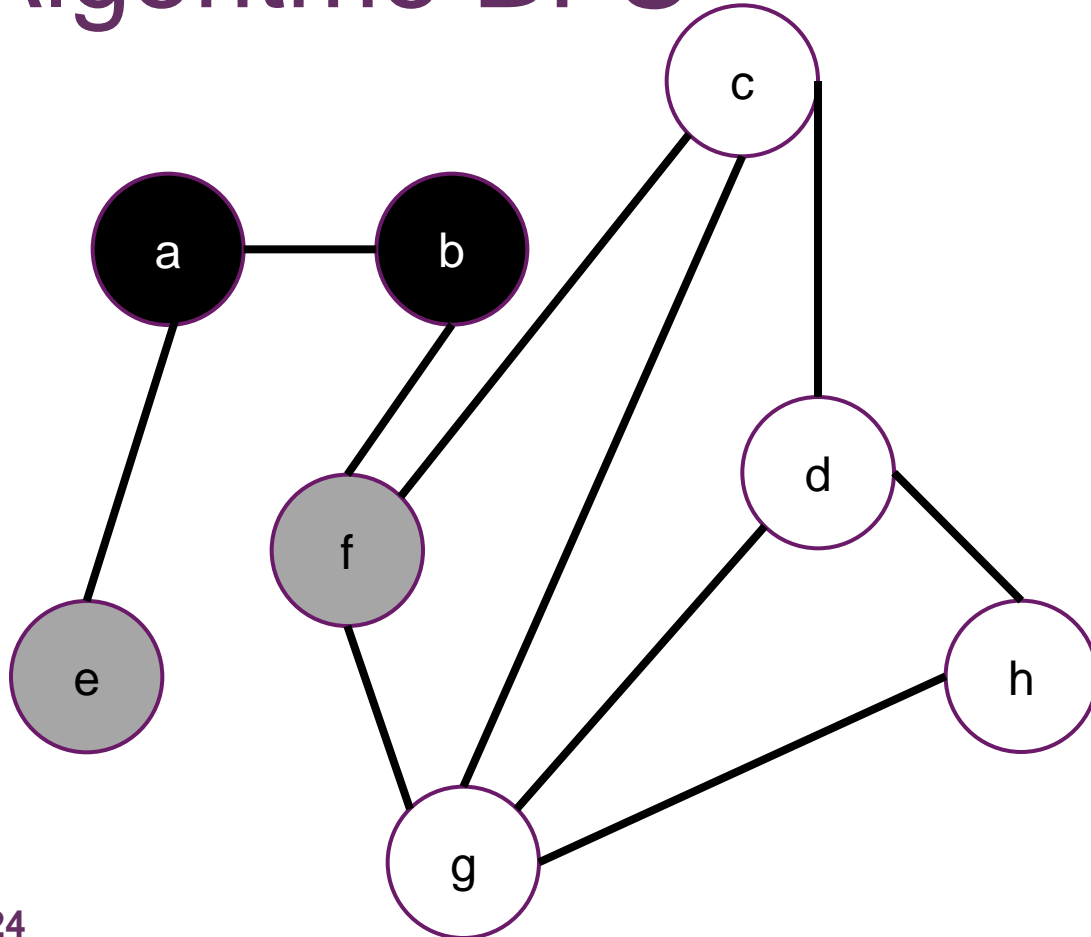
a	b	c	d	e	f	g	h
b	-	-	-	a	b	-	-

Fila Q

f	e						
---	---	--	--	--	--	--	--

Vértice: a

# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
1	0	-	-	2	1	-	-

Pais

a	b	c	d	e	f	g	h
b	-	-	-	a	b	-	-

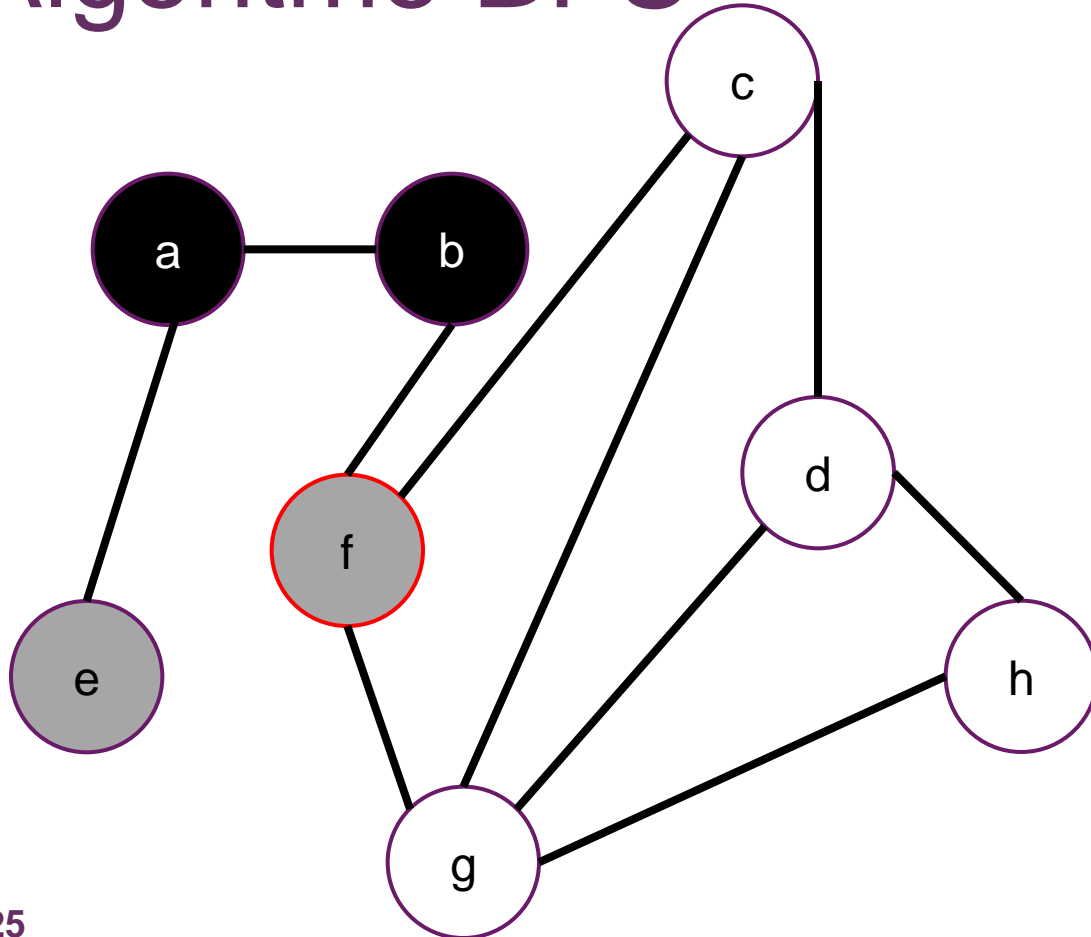
Fila Q

f	e						
---	---	--	--	--	--	--	--

Vértice: a



# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
1	0	-	-	2	1	-	-

Pais

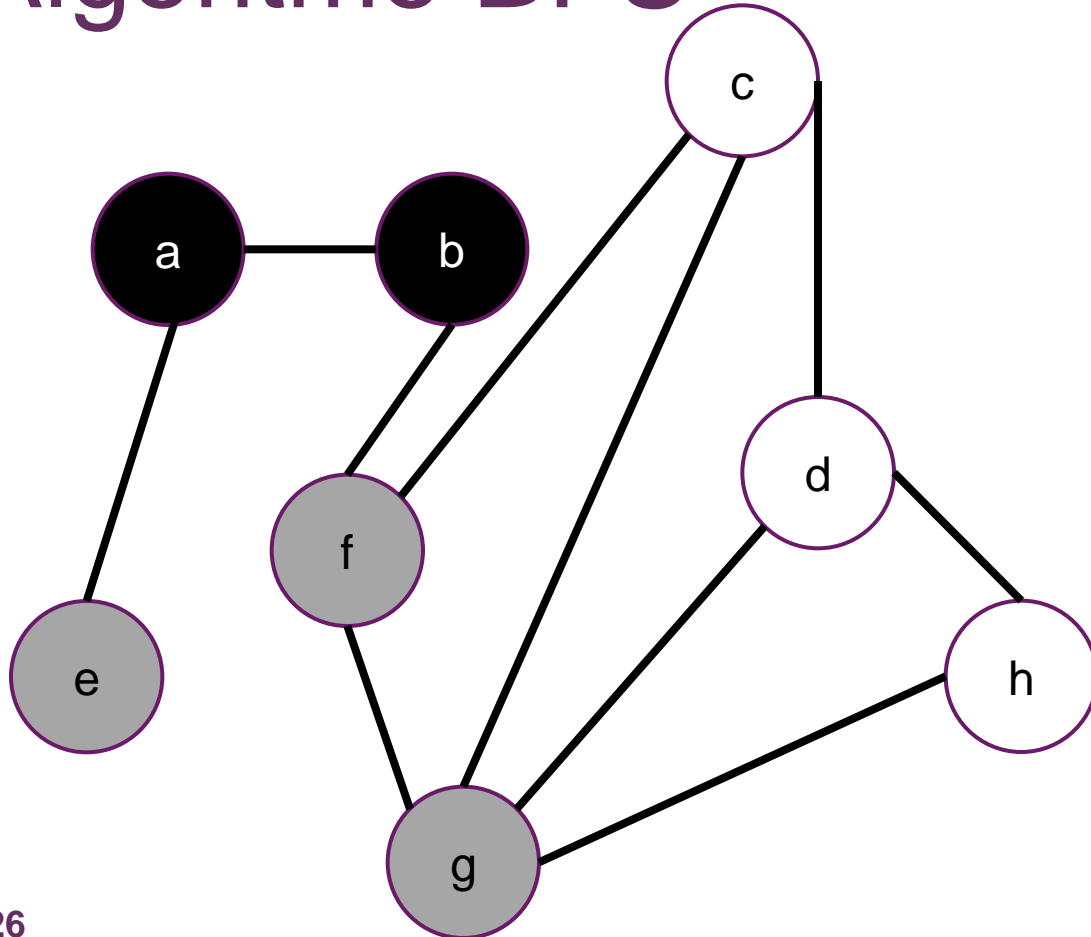
a	b	c	d	e	f	g	h
b	-	-	-	a	b	-	-

Fila Q

e							
---	--	--	--	--	--	--	--

Vértice: f

# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
1	0	-	-	2	1	2	-

Pais

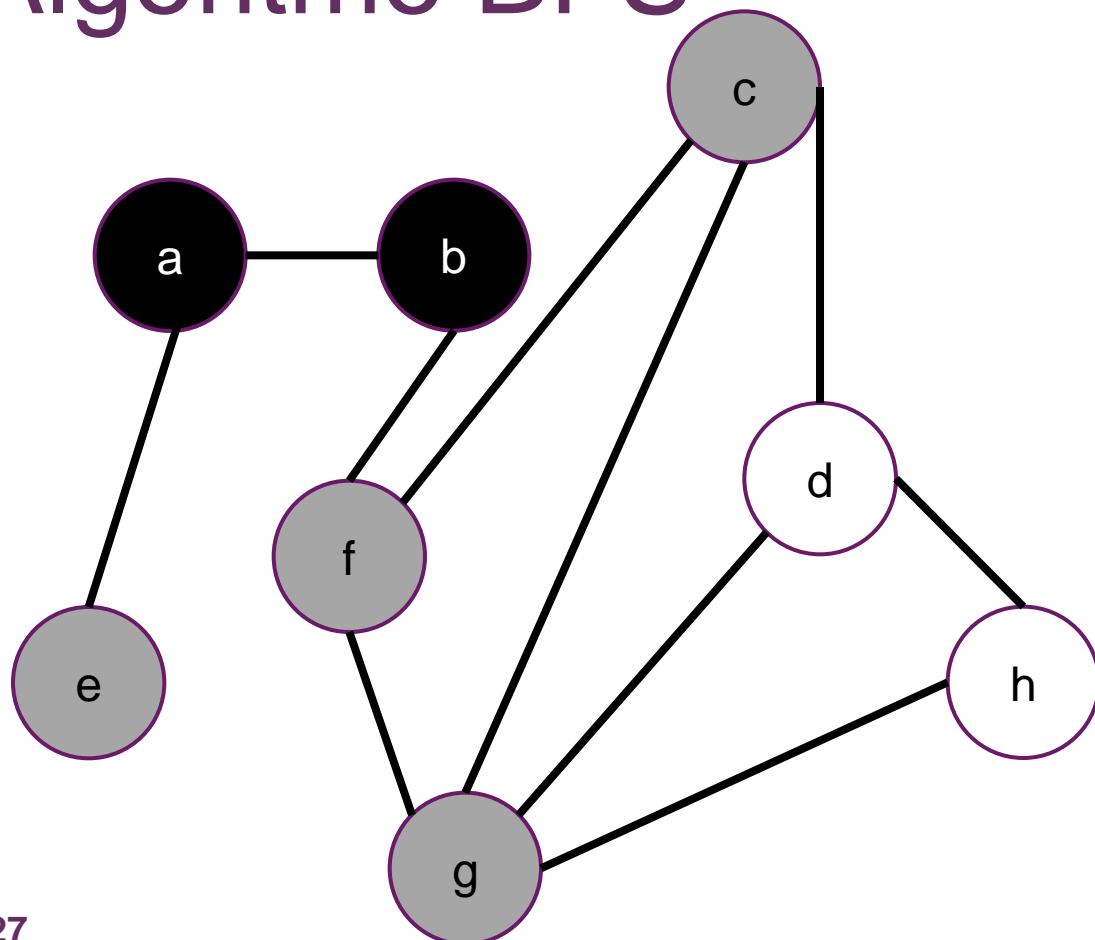
a	b	c	d	e	f	g	h
b	-	-	-	a	b	f	-

Fila Q

e	g						
---	---	--	--	--	--	--	--

Vértice: f

# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
1	0	2	-	2	1	2	-

Pais

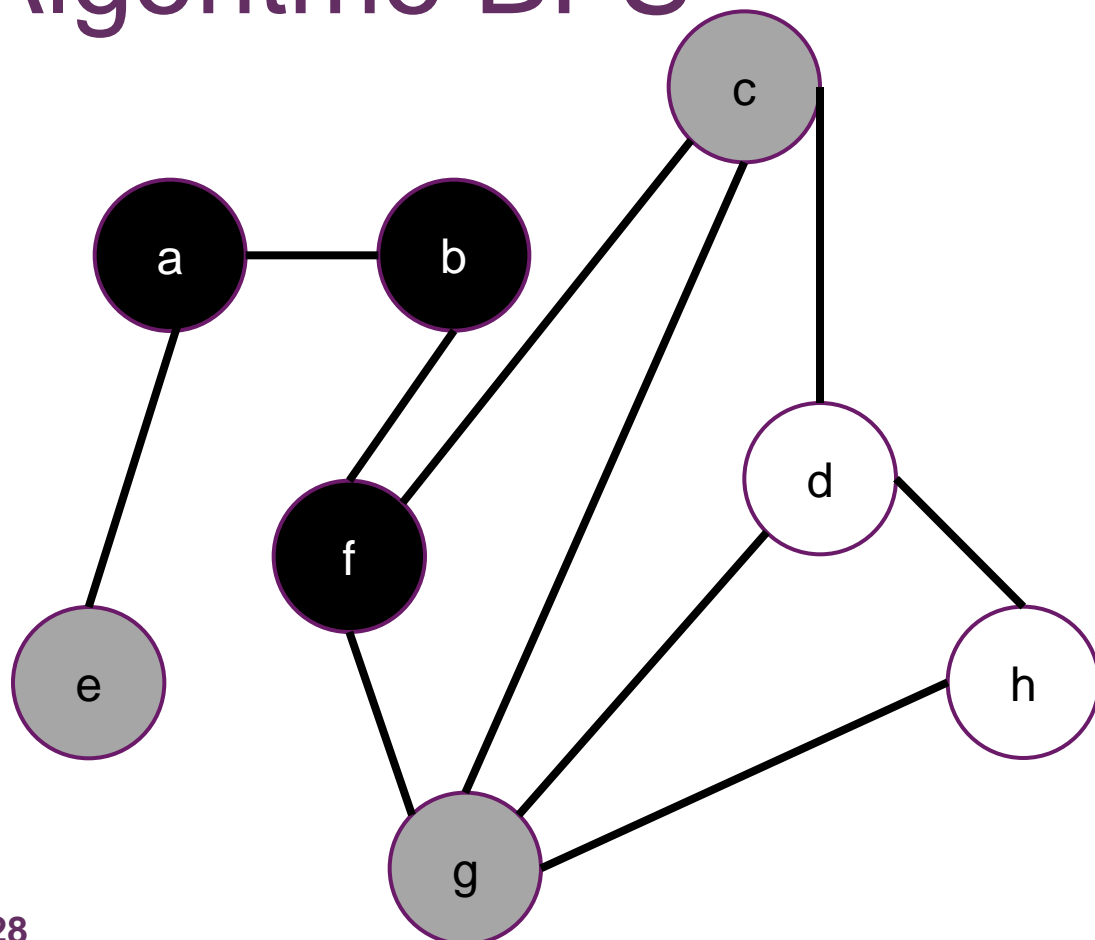
a	b	c	d	e	f	g	h
b	-	f	-	a	b	f	-

Fila Q

e	g	c					
---	---	---	--	--	--	--	--

Vértice: f

# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
1	0	2	-	2	1	2	-

Pais

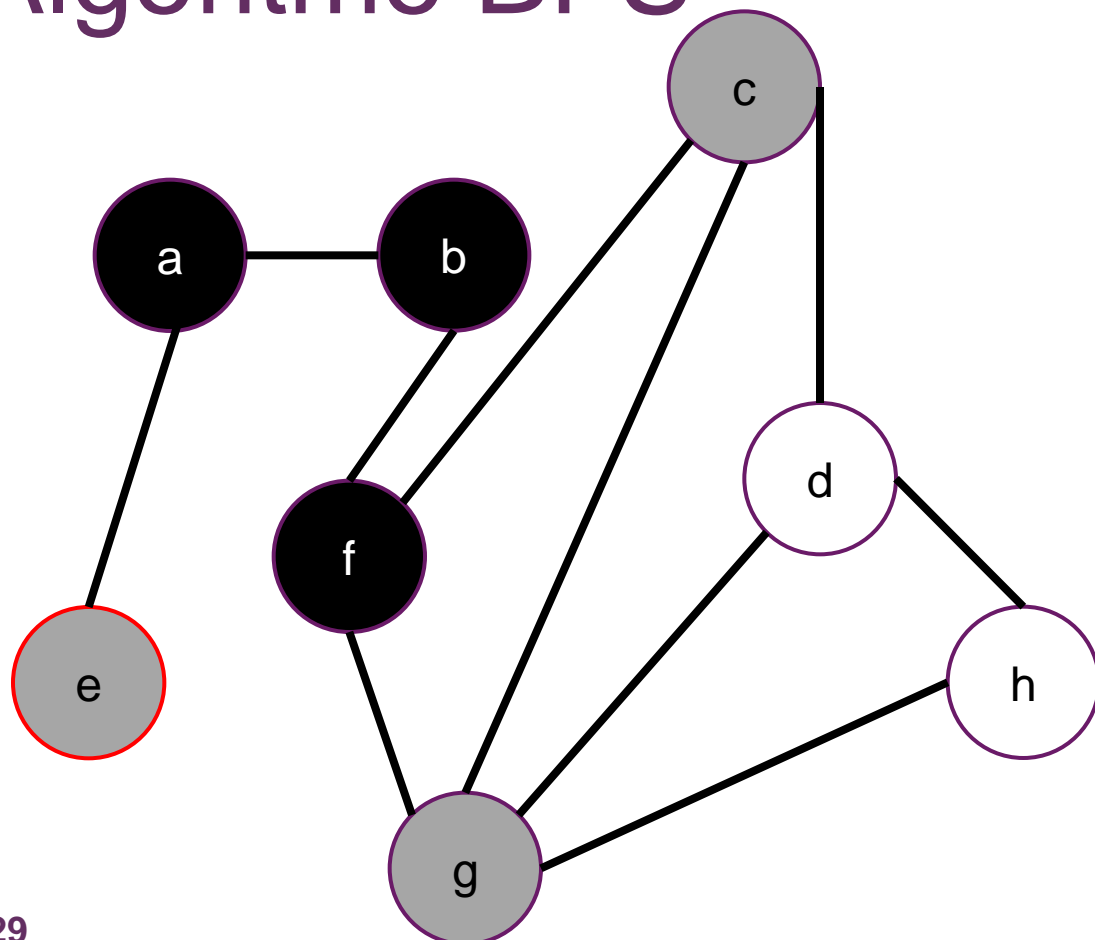
a	b	c	d	e	f	g	h
b	-	f	-	a	b	f	-

Fila Q

e	g	c					
---	---	---	--	--	--	--	--

Vértice: f

# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
1	0	2	-	2	1	2	-

Pais

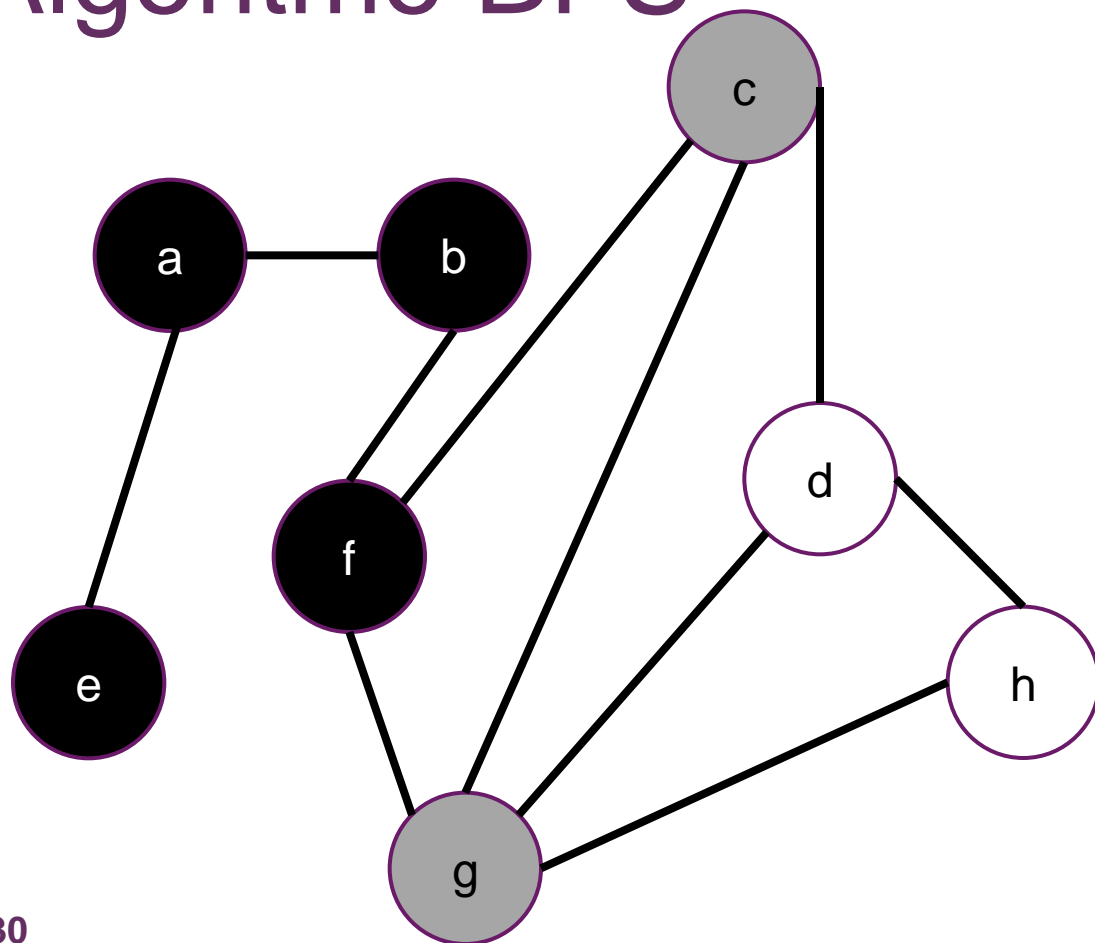
a	b	c	d	e	f	g	h
b	-	f	-	a	b	f	-

Fila Q

g	c						
---	---	--	--	--	--	--	--

Vértice: e

# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
1	0	2	-	2	1	2	-

Pais

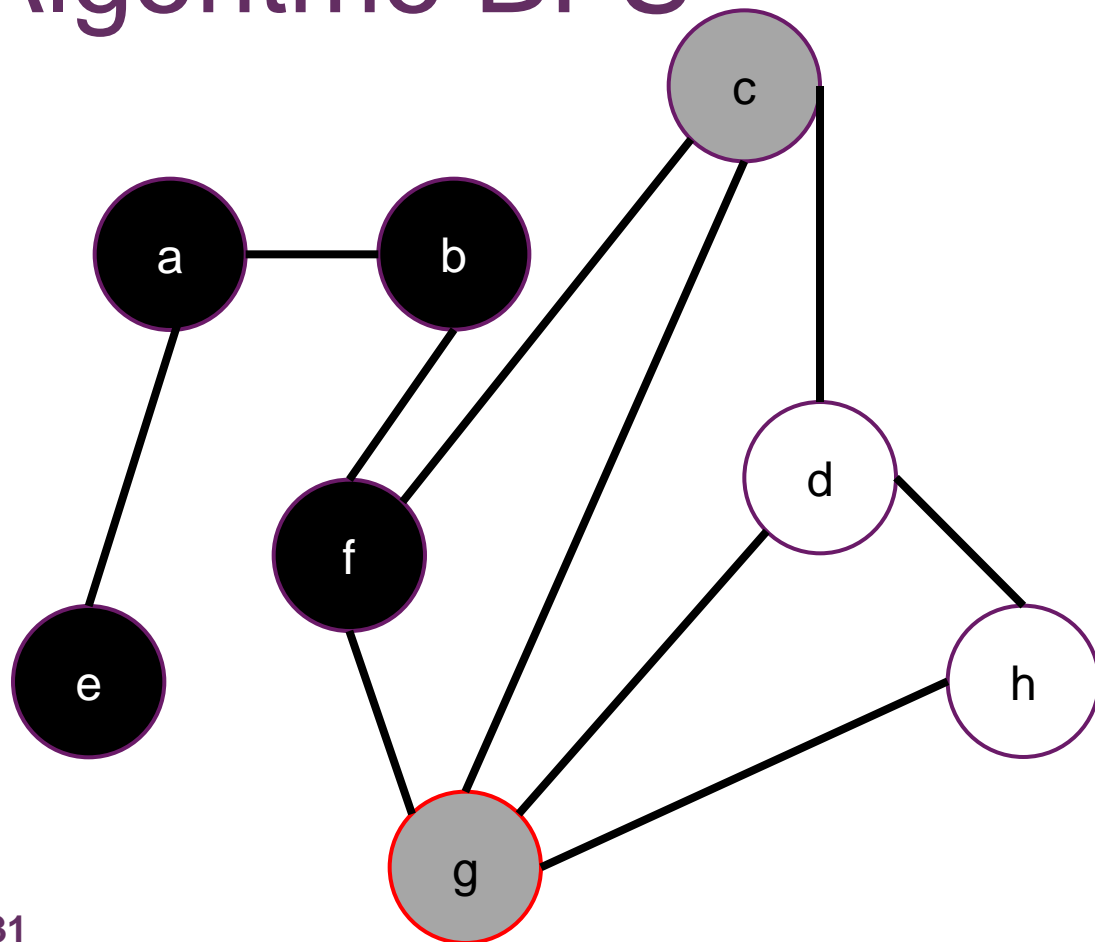
a	b	c	d	e	f	g	h
b	-	f	-	a	b	f	-

Fila Q

g	c						
---	---	--	--	--	--	--	--

Vértice: e

# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
1	0	2	-	2	1	2	-

Pais

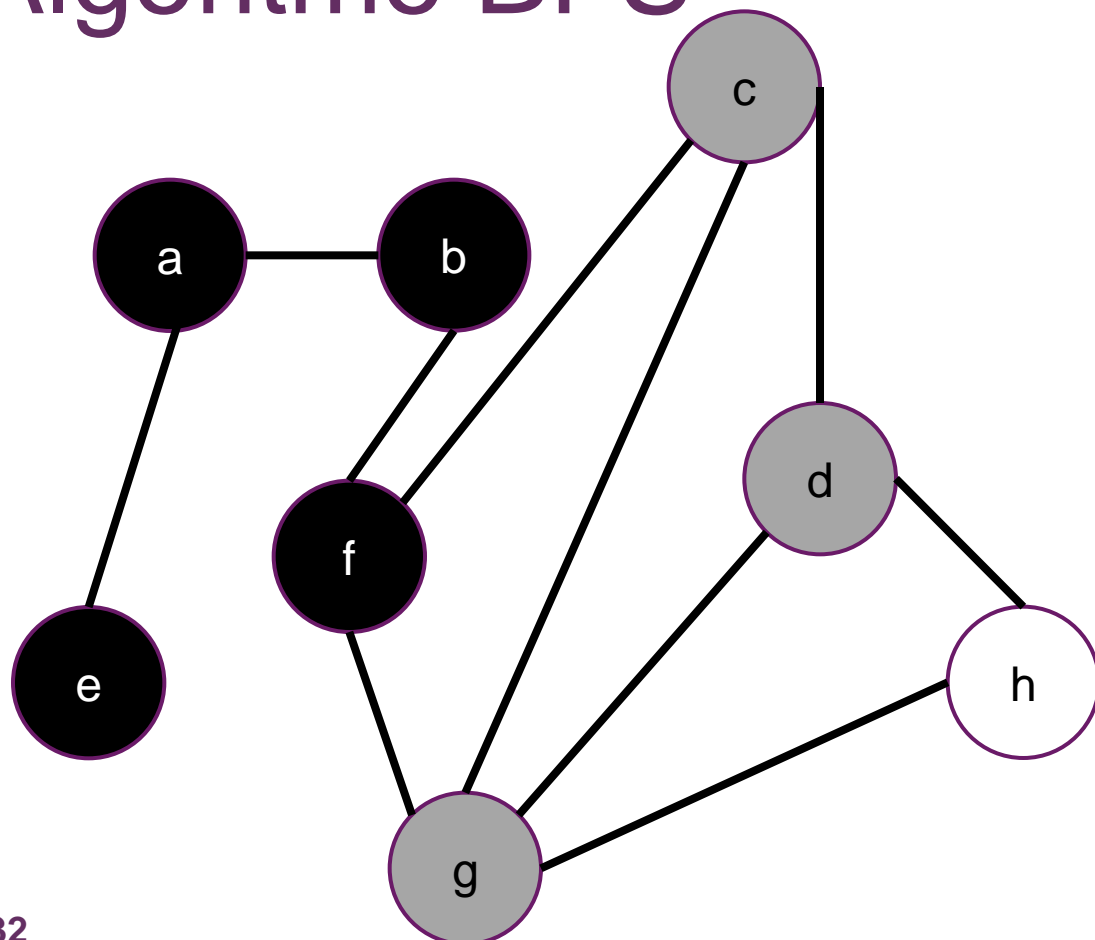
a	b	c	d	e	f	g	h
b	-	f	-	a	b	f	-

Fila Q

c							
---	--	--	--	--	--	--	--

Vértice: g

# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
1	0	2	3	2	1	2	-

Pais

a	b	c	d	e	f	g	h
b	-	f	g	a	b	f	-

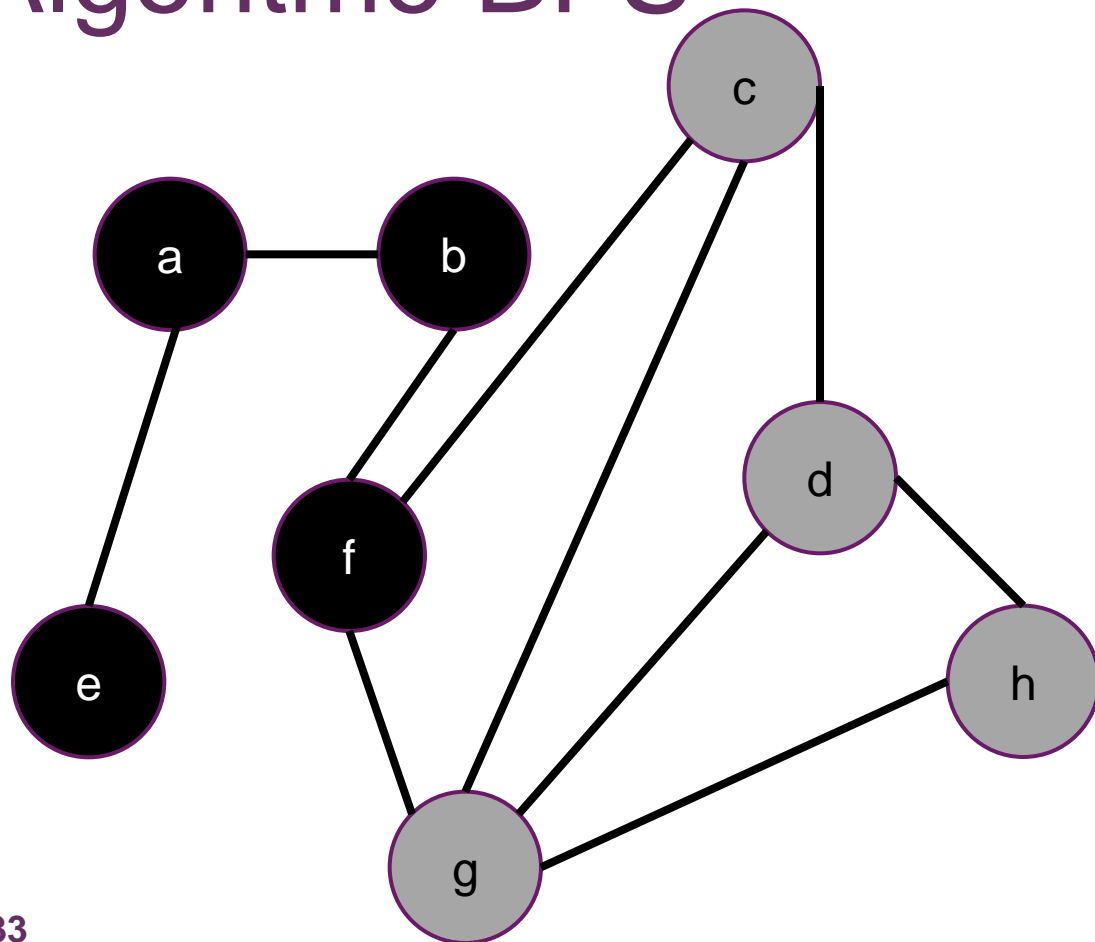
Fila Q

c	d						
---	---	--	--	--	--	--	--

Vértice: g



# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
1	0	2	3	2	1	2	3

Pais

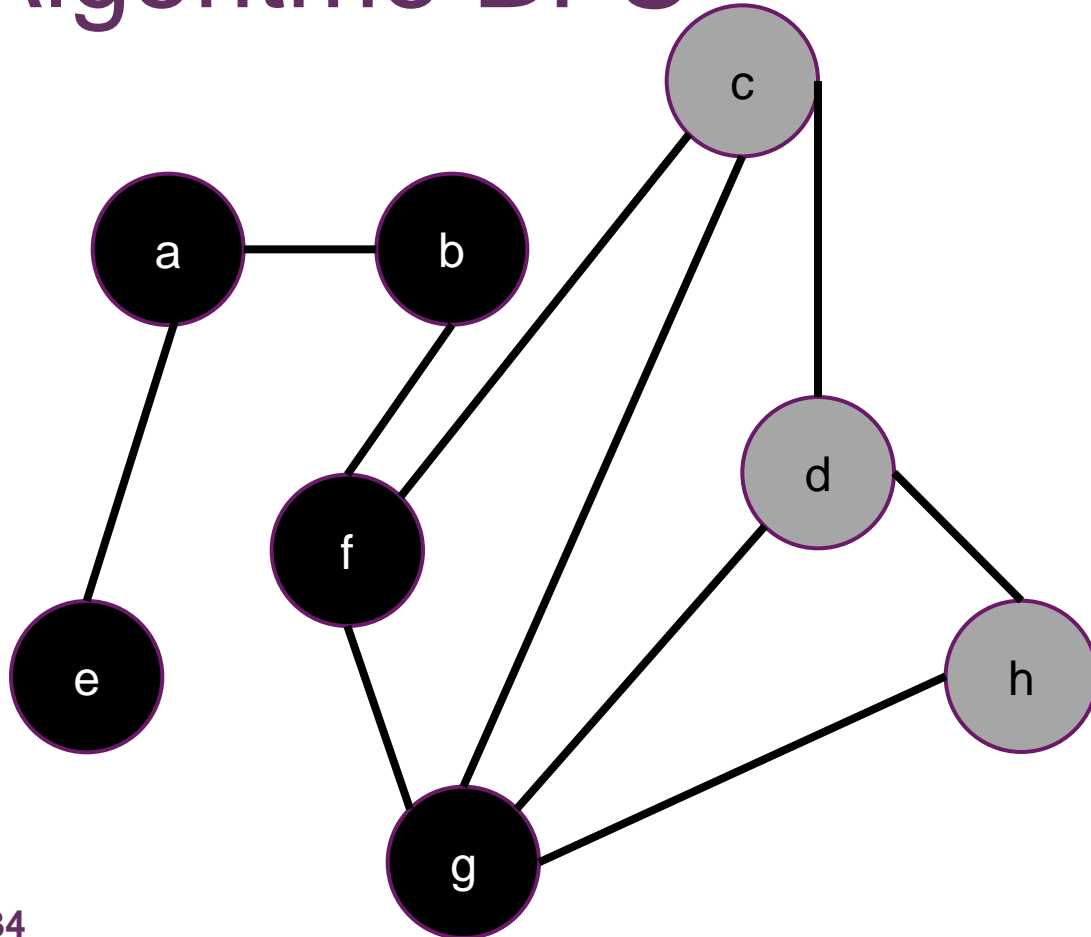
a	b	c	d	e	f	g	h
b	-	f	g	a	b	f	g

Fila Q

c	d	h					
---	---	---	--	--	--	--	--

Vértice: g

# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
1	0	2	3	2	1	2	3

Pais

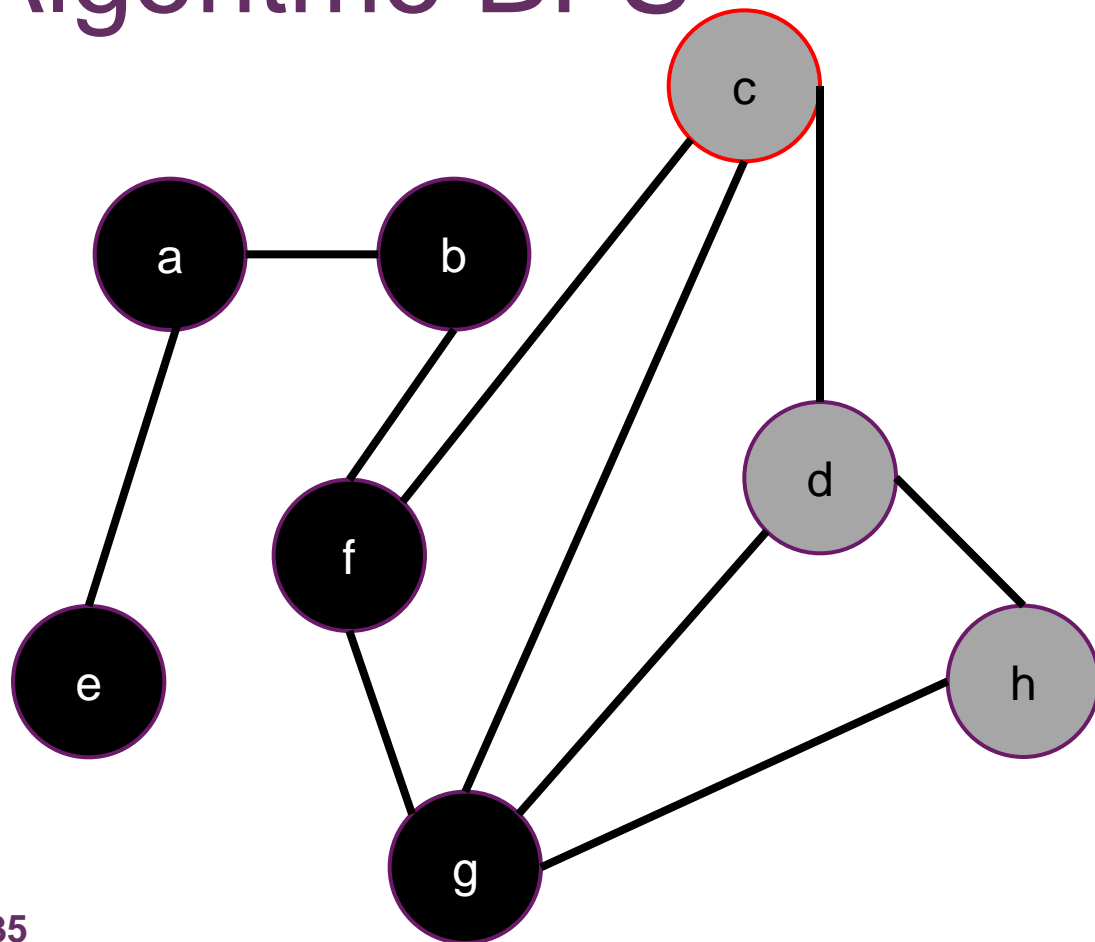
a	b	c	d	e	f	g	h
b	-	f	g	a	b	f	g

Fila Q

c	d	h					
---	---	---	--	--	--	--	--

Vértice:

# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
1	0	2	3	2	1	2	3

Pais

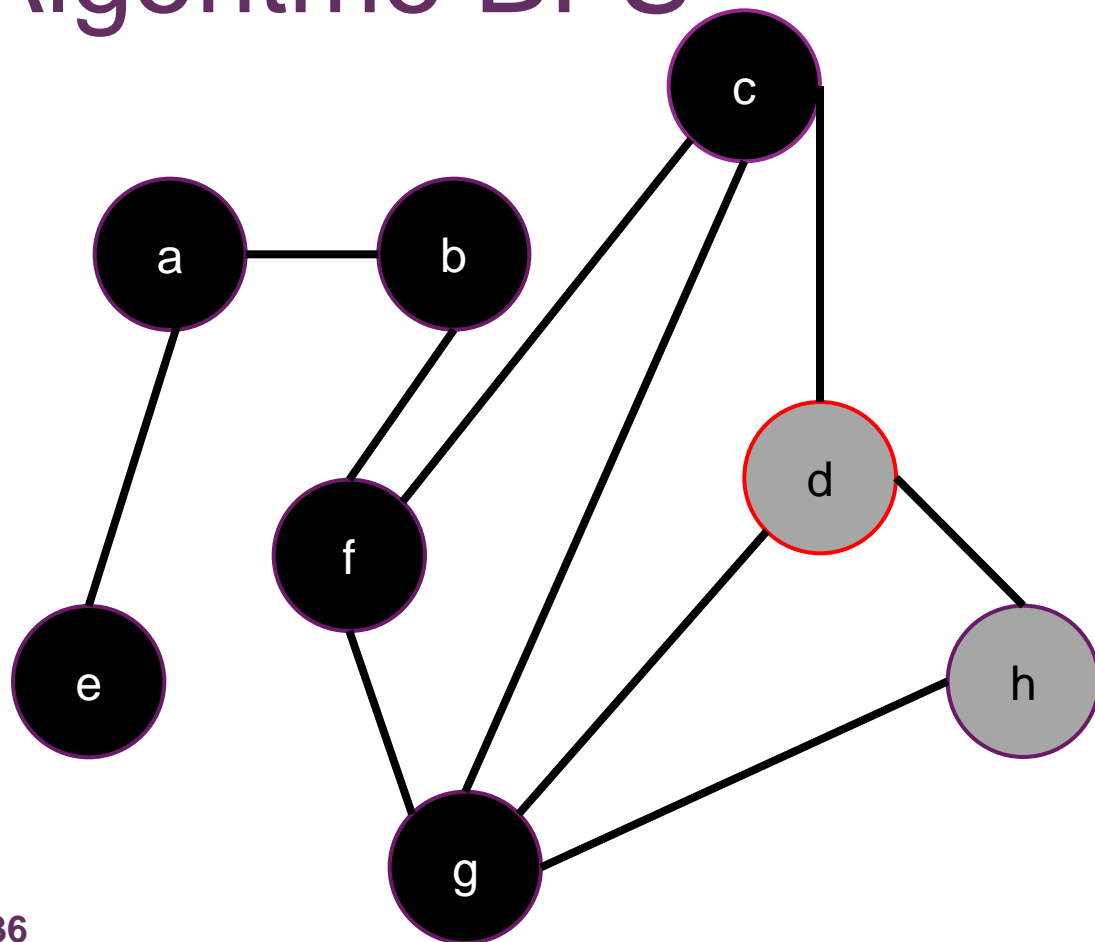
a	b	c	d	e	f	g	h
b	-	f	g	a	b	f	g

Fila Q

d	h						
---	---	--	--	--	--	--	--

Vértice: c

# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
1	0	2	3	2	1	2	3

Pais

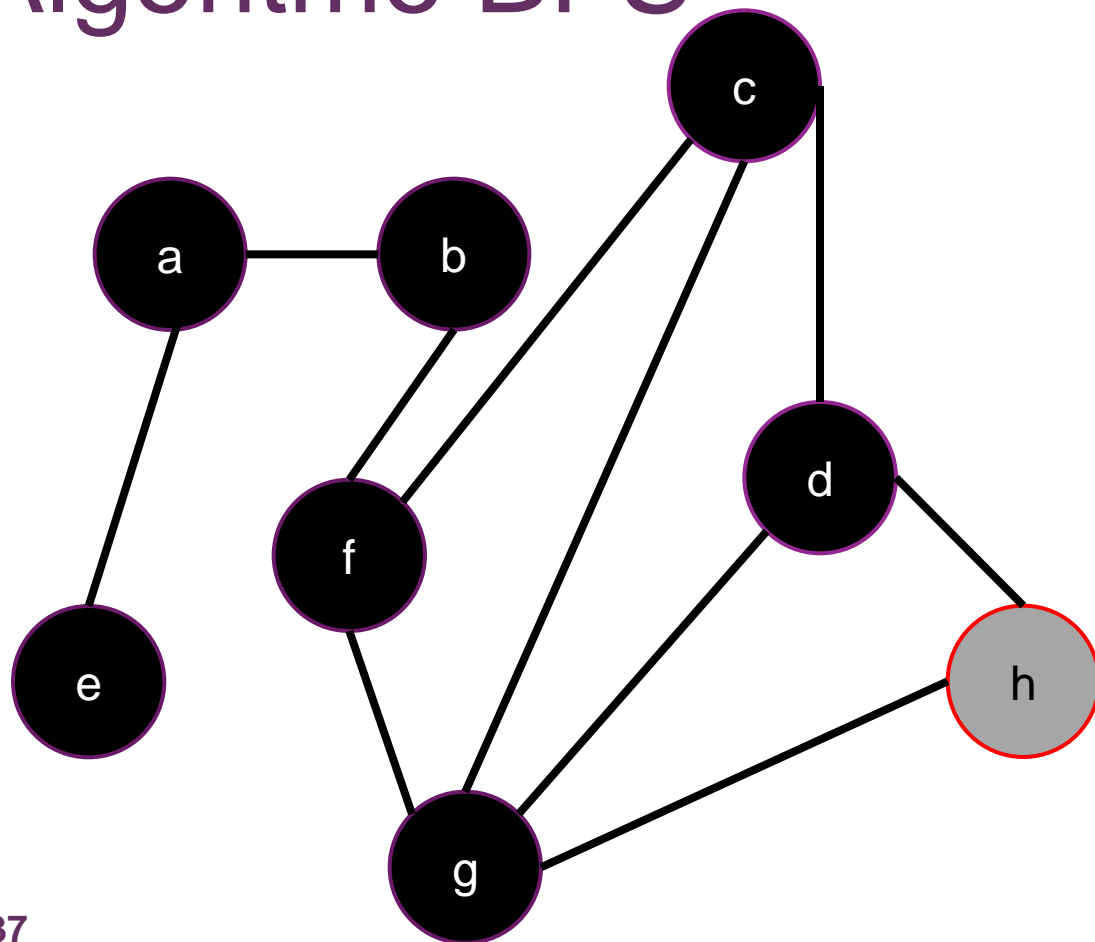
a	b	c	d	e	f	g	h
b	-	f	g	a	b	f	g

Fila Q

h							
---	--	--	--	--	--	--	--

Vértice: d

# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
1	0	2	3	2	1	2	3

Pais

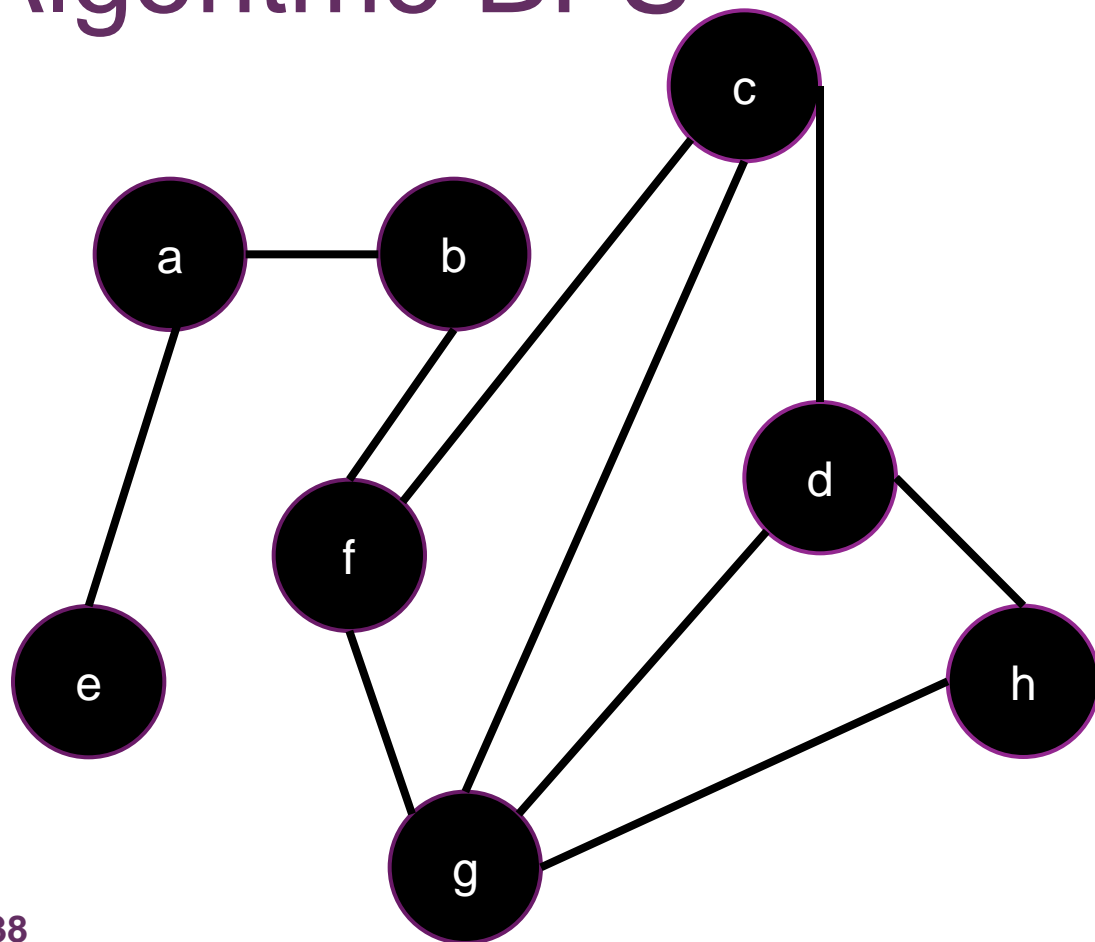
a	b	c	d	e	f	g	h
b	-	f	g	a	b	f	g

Fila Q



Vértice: h

# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
1	0	2	3	2	1	2	3

Pais

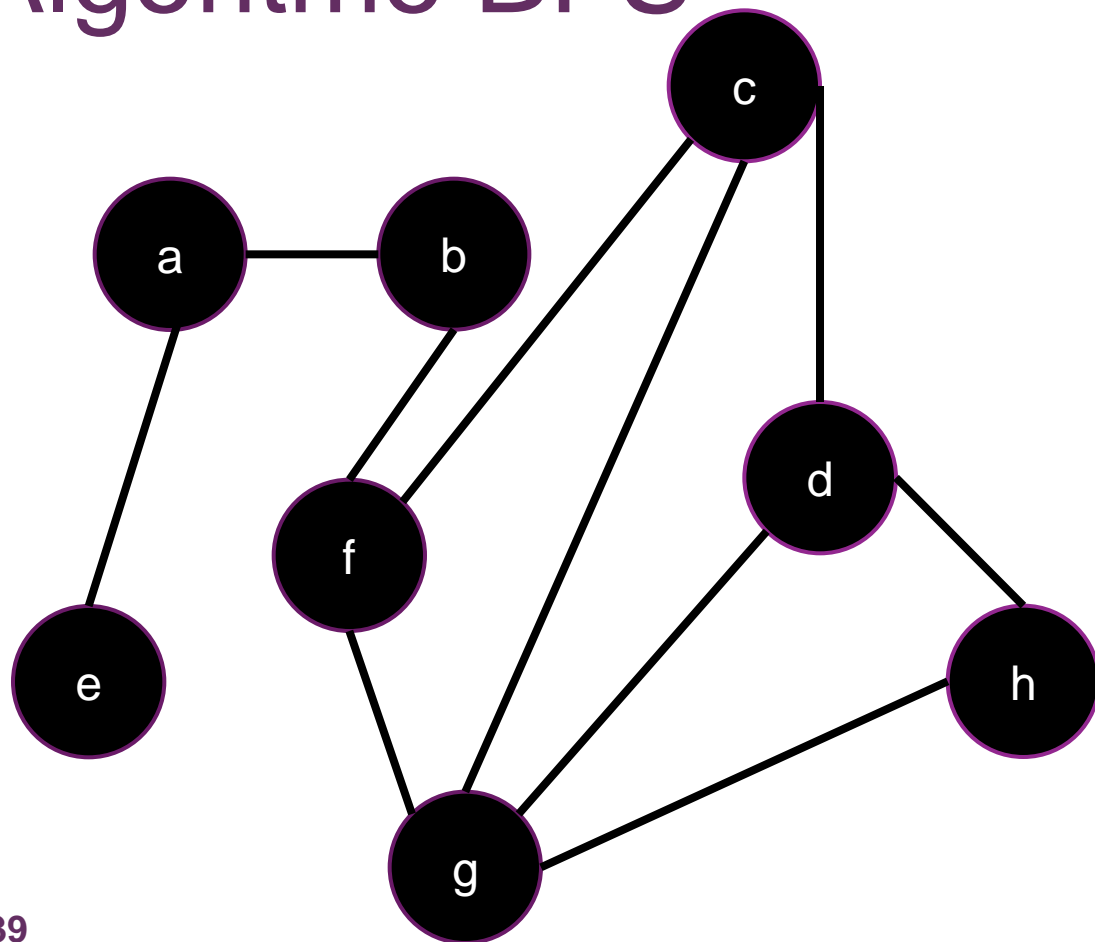
a	b	c	d	e	f	g	h
b	-	f	g	a	b	f	g

Fila Q



**Fila vazia: fim**

# Algoritmo BFS



Distâncias

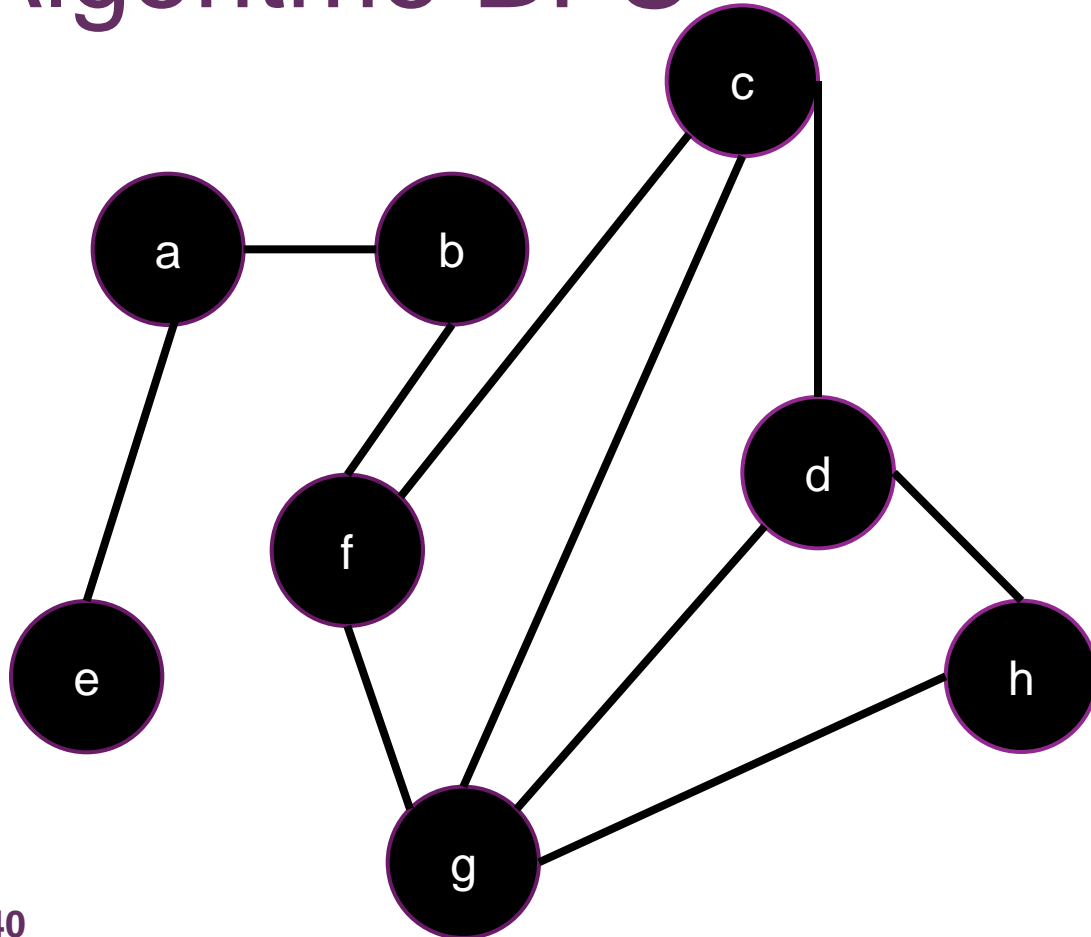
a	b	c	d	e	f	g	h
1	0	2	3	2	1	2	3

Pais

a	b	c	d	e	f	g	h
b	-	f	g	a	b	f	g

Caminho até o vértice h:  
h

# Algoritmo BFS



Distâncias

a	b	c	d	e	f	g	h
1	0	2	3	2	1	2	3

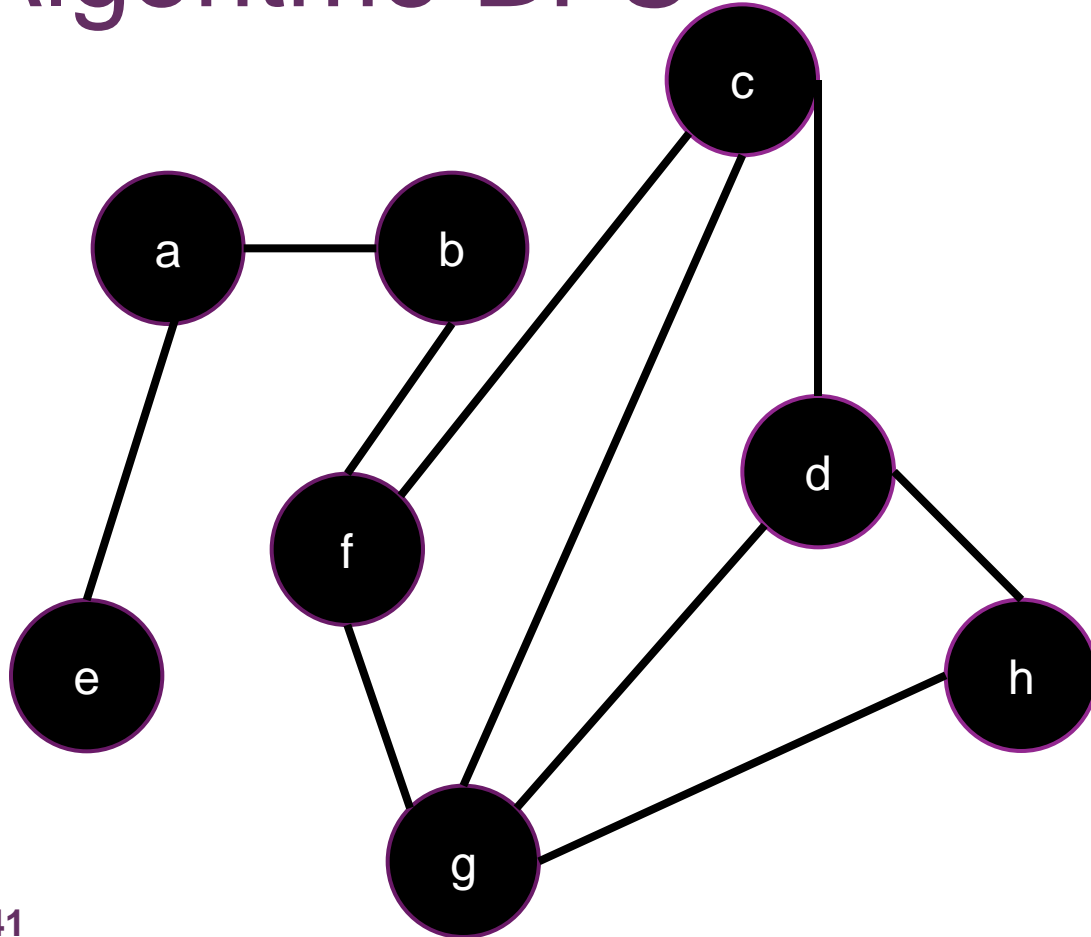
Pais

a	b	c	d	e	f	g	h
b	-	f	g	a	b	f	g

Caminho até o vértice h:  
g-h



# Algoritmo BFS



Distâncias

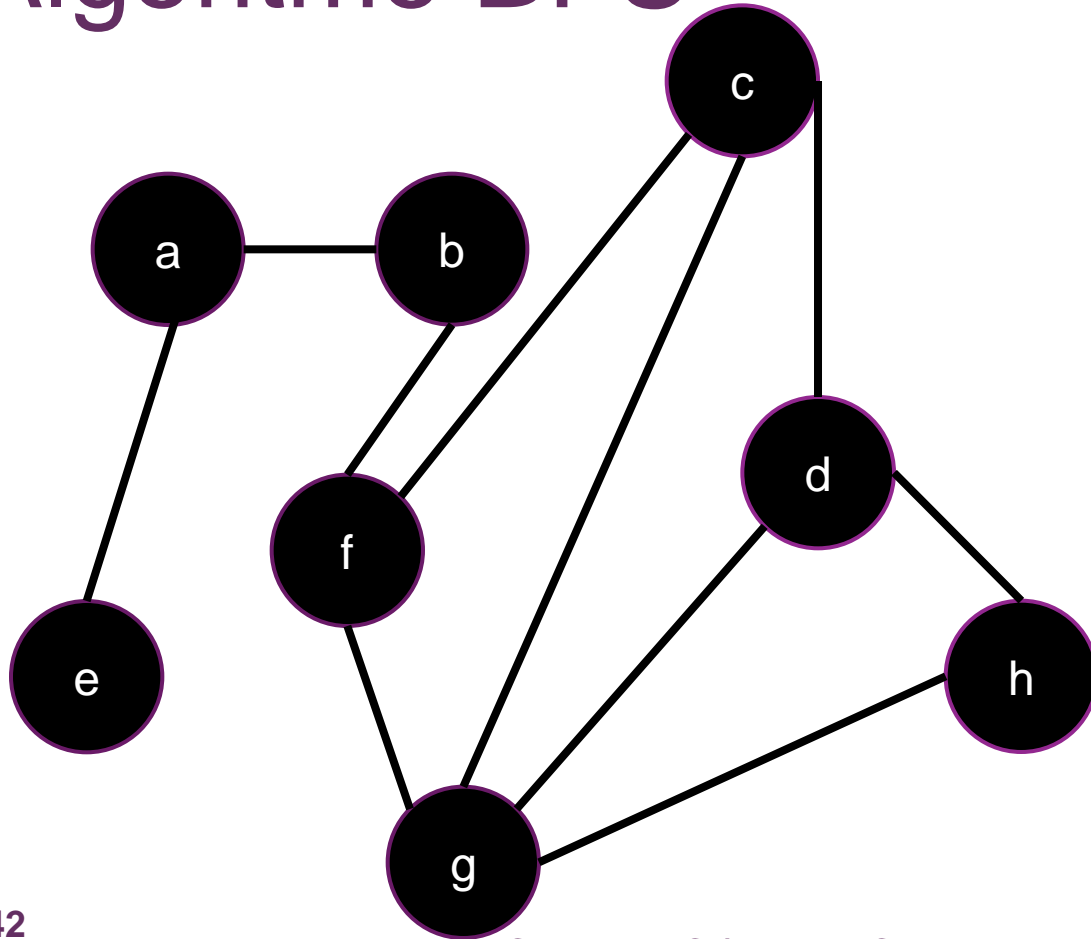
a	b	c	d	e	f	g	h
1	0	2	3	2	1	2	3

Pais

a	b	c	d	e	f	g	h
b	-	f	g	a	b	f	g

Caminho até o vértice h:  
f-g-h

# Algoritmo BFS



Distâncias

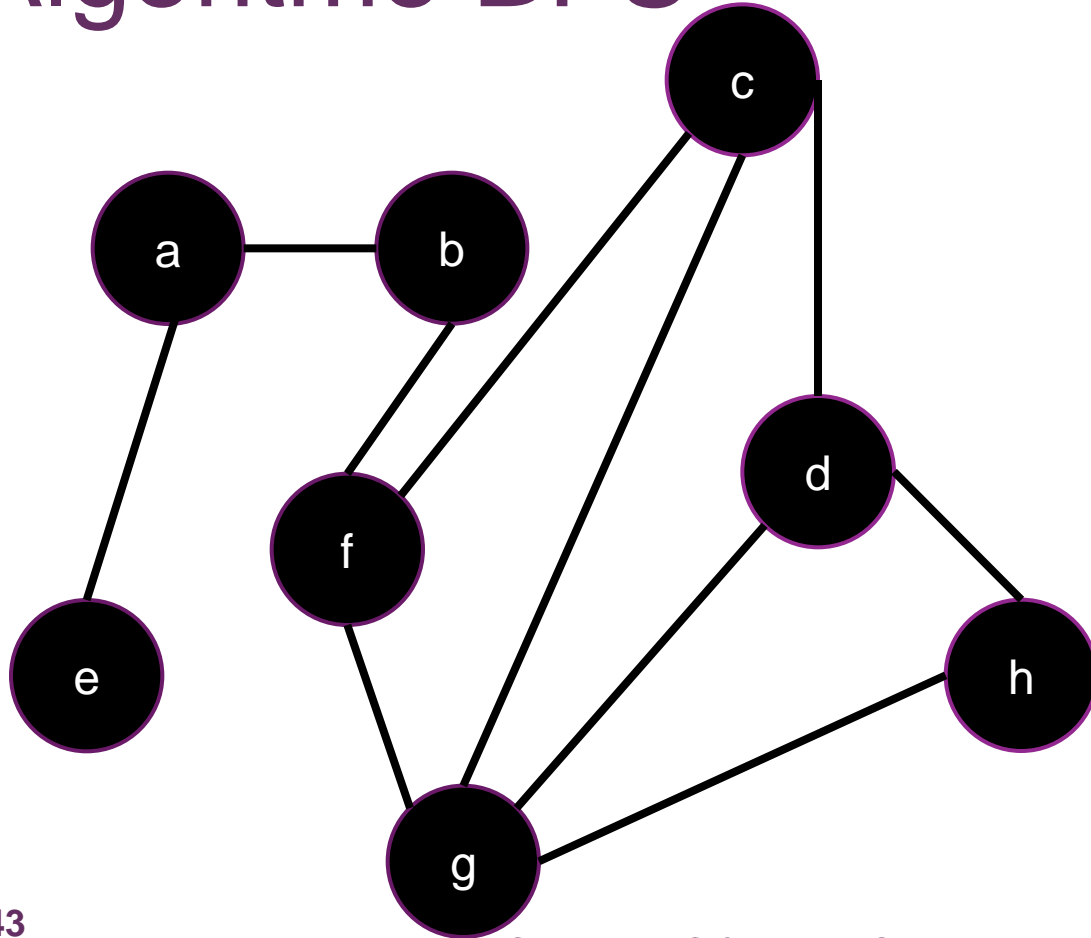
a	b	c	d	e	f	g	h
1	0	2	3	2	1	2	3

Pais

a	b	c	d	e	f	g	h
b	-	f	g	a	b	f	g

Caminho até o vértice h:  
b-f-g-h

# Algoritmo BFS



Distâncias

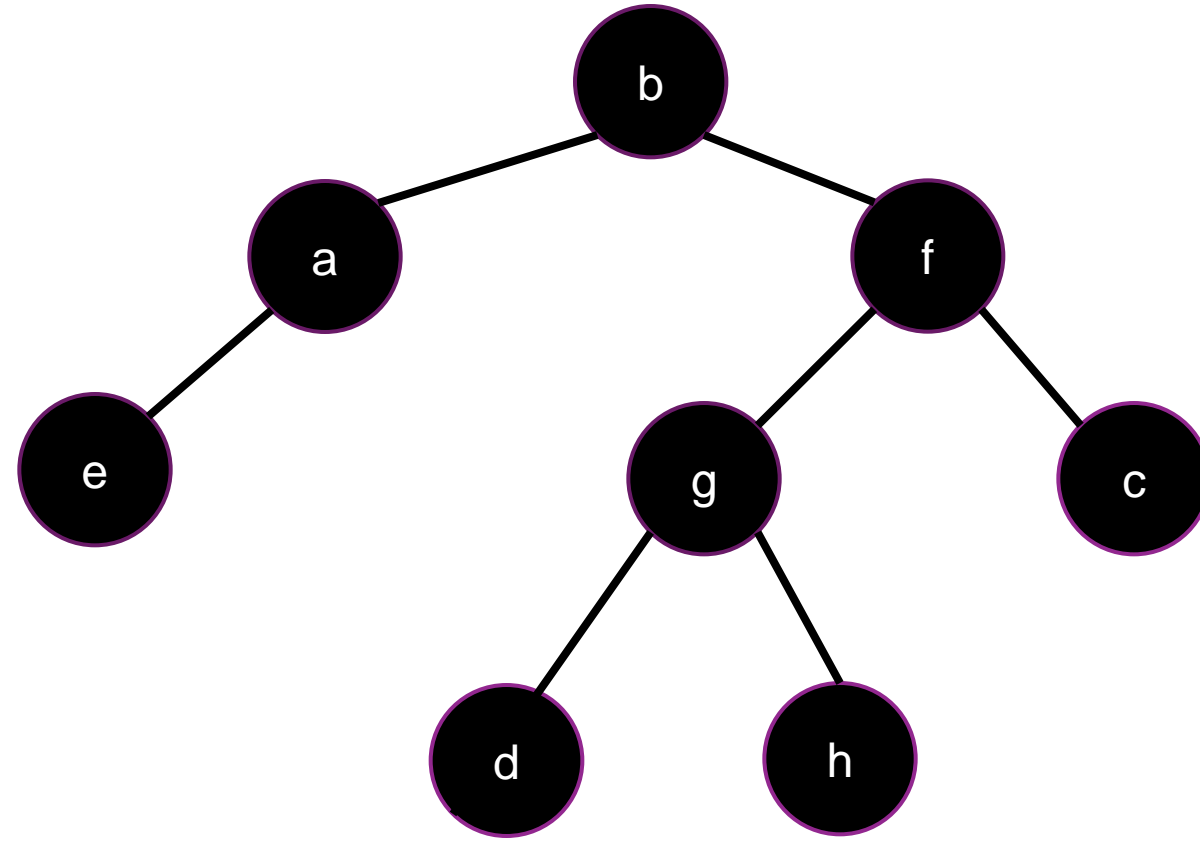
a	b	c	d	e	f	g	h
1	0	2	3	2	1	2	3

Pais

a	b	c	d	e	f	g	h
b	-	f	g	a	b	f	g

Caminho até o vértice h:  
b-f-g-h

# Árvore BFS



OBRIGADO.

Dúvidas?

# Busca em profundidade

46

- Em inglês, *Depth First Search* (DFS)
- A partir de um vértice de origem, busca recursivamente um vértice adjacente, até que não existam mais vértices a visitar
- Pode gerar várias árvores de profundidade (floresta de busca)

# Busca em profundidade

47

- Mantidas as propriedades de estado
- Nova propriedade: *timestamps* (tempo da busca)
  - *Timestamp* de descoberta
  - *Timestamp* de término

# Algoritmo DFS - inicialização

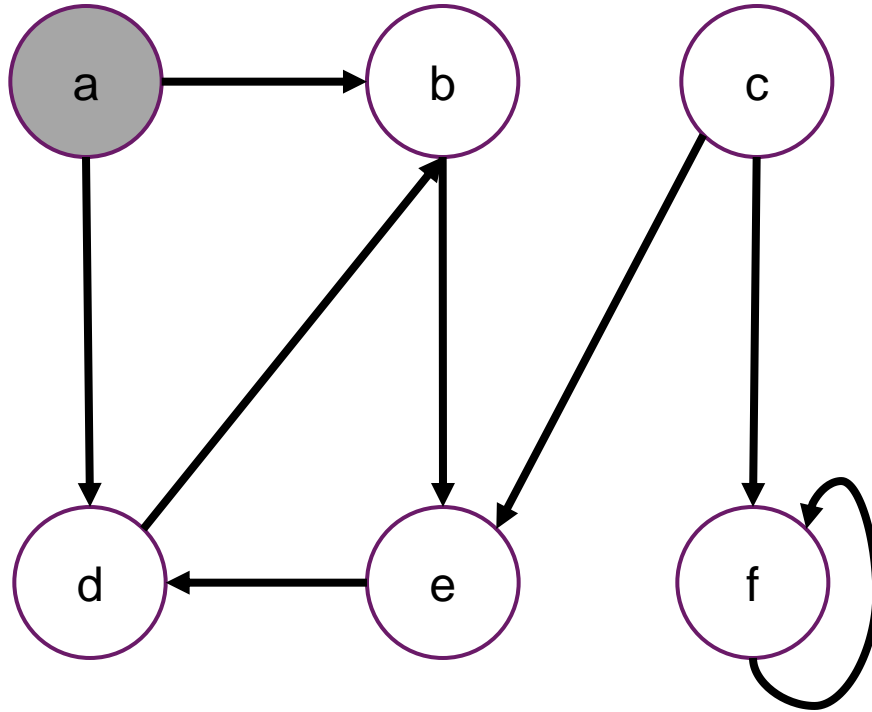
```
Para cada vértice  $u$  faça
     $u.cor = \text{branco};$ 
     $u.pai = \text{null};$ 
Fim para
timestamp = 0
Para cada vértice  $u$  faça
    se  $u.cor == \text{branco}$ 
        Visitar( $u$ );
    Fim se
Fim Para
```



# Algoritmo DFS – principal (visita)

```
timestamp = timestamp + 1;  
u.descoberta = timestamp;  
u.cor = cinza;  
Para cada vértice v vizinho de u faça  
    se v.cor == branco  
        v.pai = u;  
        Visitar(v);  
    Fim se  
Fim Para  
u.cor = preto;  
timestamp = timestamp+1;  
u.término = timestamp;
```

# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	-	-	-	-	-

Finalização

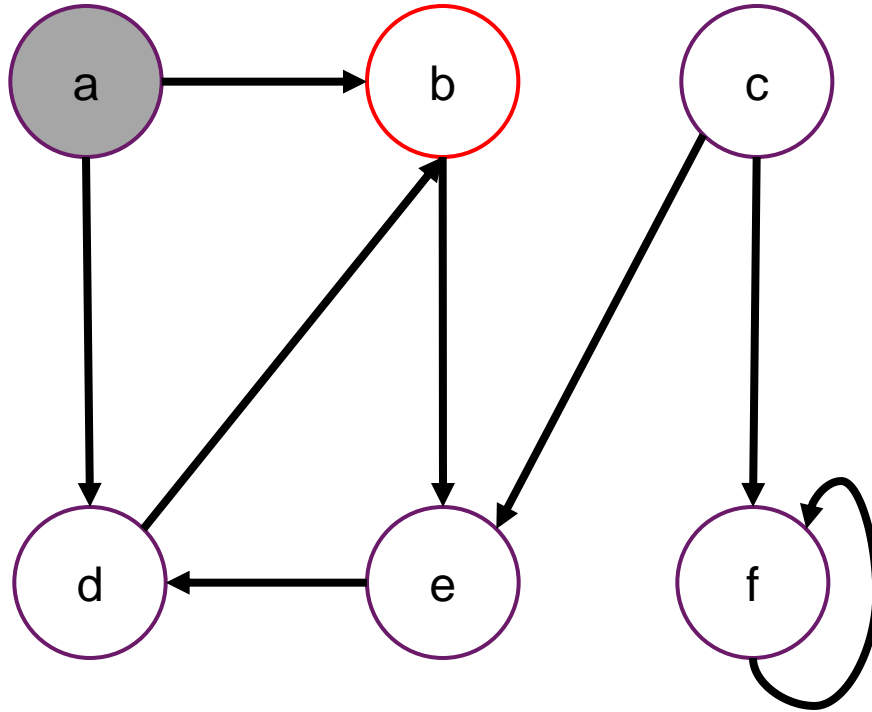
a	b	c	d	e	f
-	-	-	-	-	-

Pais

a	b	c	d	e	f
-	-	-	-	-	-

Timestamp: 1

# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	-	-	-	-	-

Finalização

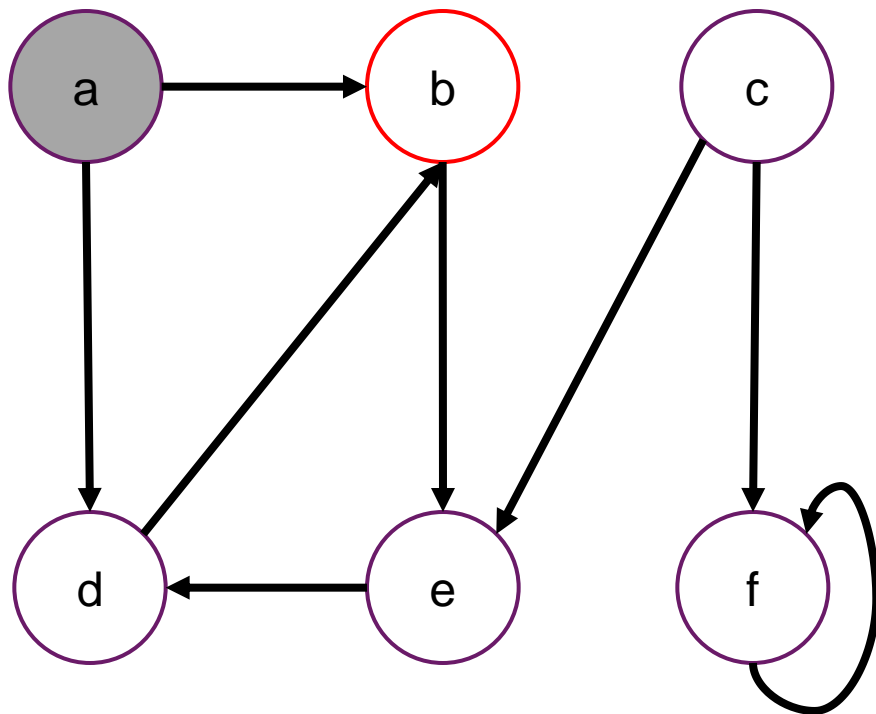
a	b	c	d	e	f
-	-	-	-	-	-

Pais

a	b	c	d	e	f
-	-	-	-	-	-

Timestamp: 1

# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	-	-	-	-	-

Finalização

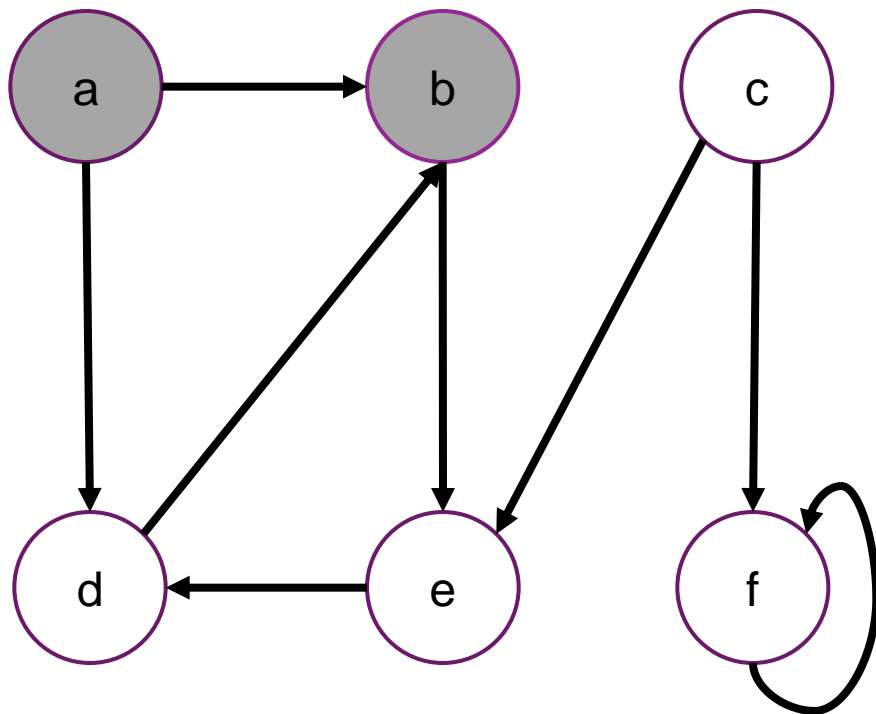
a	b	c	d	e	f
-	-	-	-	-	-

Pais

a	b	c	d	e	f
-	a	-	-	-	-

Timestamp: 1

# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	2	-	-	-	-

Finalização

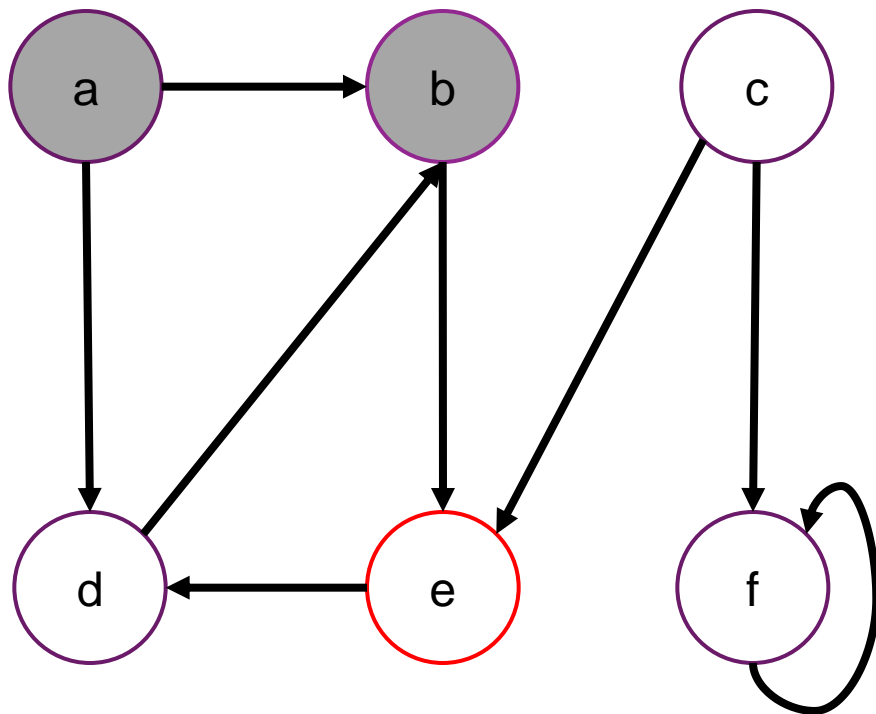
a	b	c	d	e	f
-	-	-	-	-	-

Pais

a	b	c	d	e	f
-	a	-	-	-	-

Timestamp: 2

# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	2	-	-	-	-

Finalização

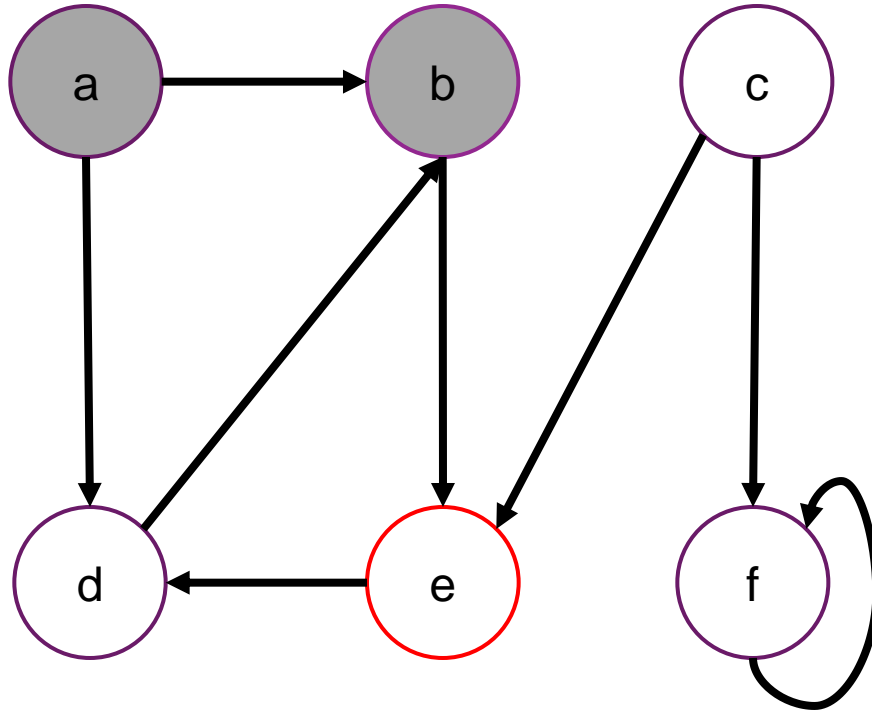
a	b	c	d	e	f
-	-	-	-	-	-

Pais

a	b	c	d	e	f
-	a	-	-	-	-

Timestamp: 2

# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	2	-	-	-	-

Finalização

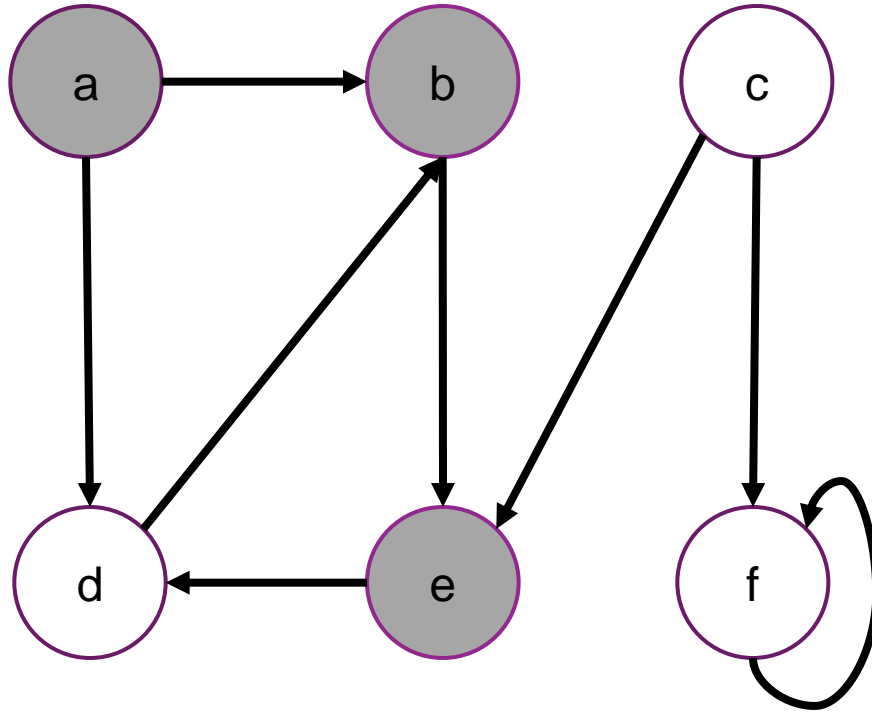
a	b	c	d	e	f
-	-	-	-	-	-

Pais

a	b	c	d	e	f
-	a	-	-	b	-

Timestamp: 2

# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	2	-	-	3	-

Finalização

a	b	c	d	e	f
-	-	-	-	-	-

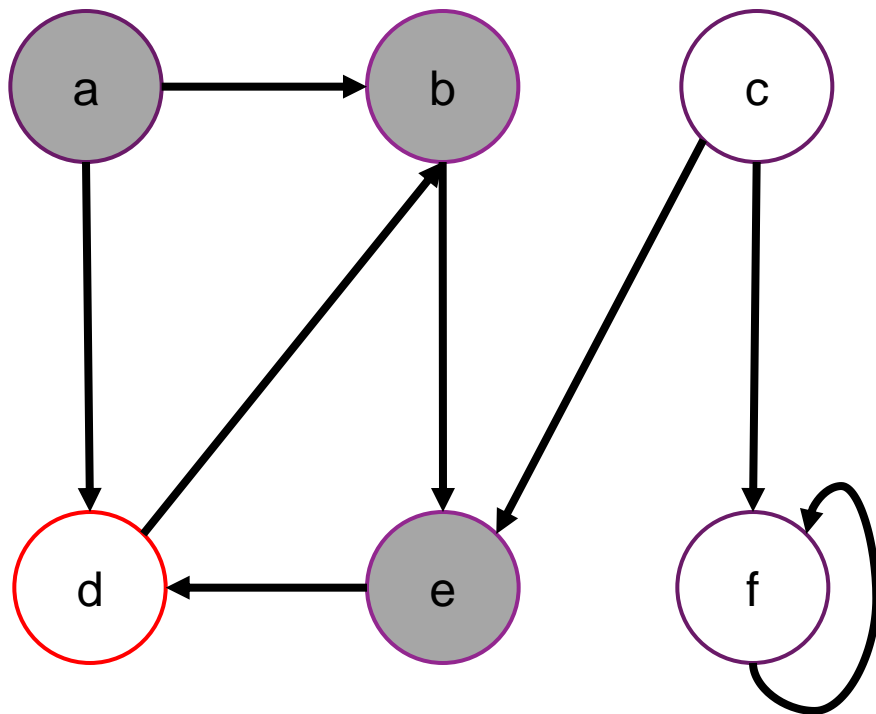
Pais

a	b	c	d	e	f
-	a	-	-	b	-

Timestamp: 3



# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	2	-	-	3	-

Finalização

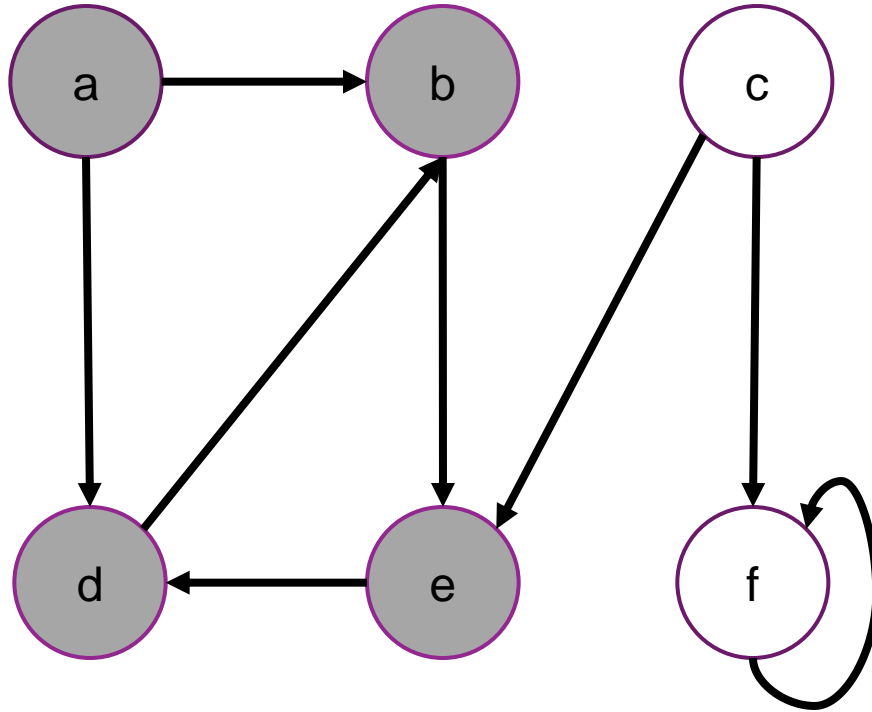
a	b	c	d	e	f
-	-	-	-	-	-

Pais

a	b	c	d	e	f
-	a	-	e	b	-

Timestamp: 3

# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	2	-	4	3	-

Finalização

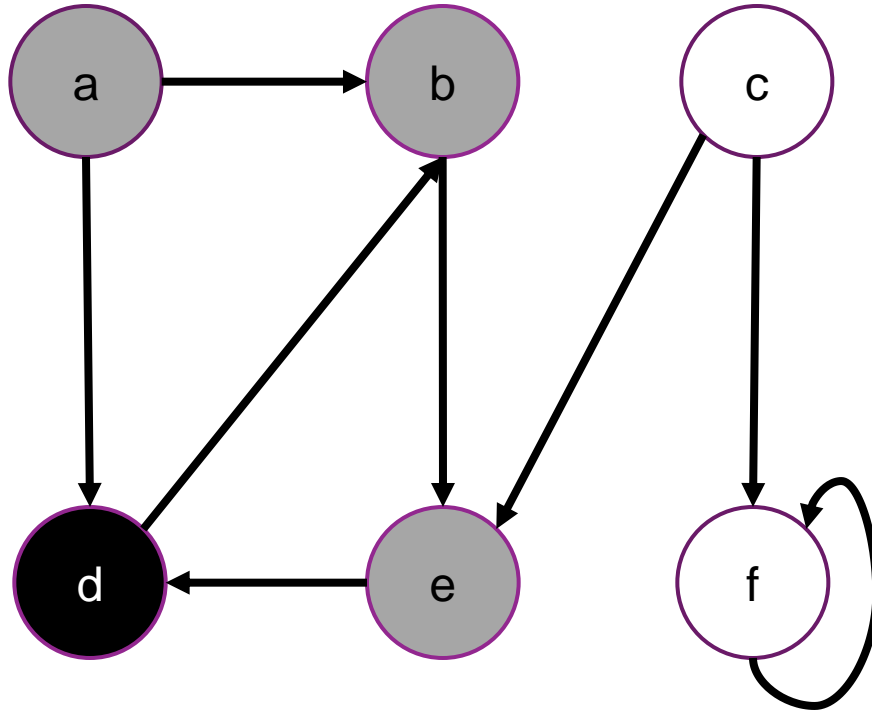
a	b	c	d	e	f
-	-	-	-	-	-

Pais

a	b	c	d	e	f
-	a	-	e	b	-

Timestamp: 4

# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	2	-	4	3	-

Finalização

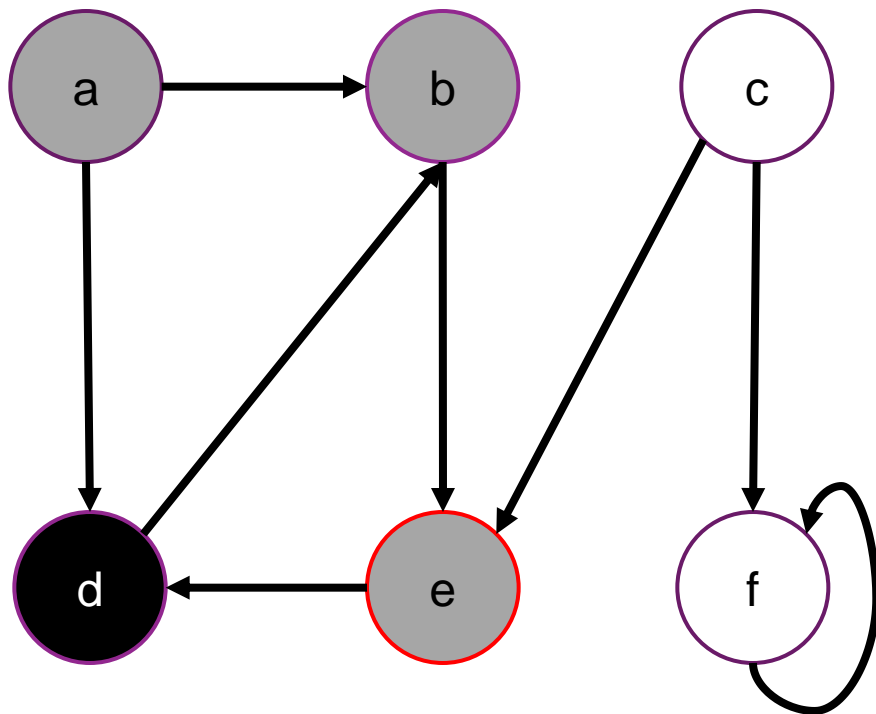
a	b	c	d	e	f
-	-	-	5	-	-

Pais

a	b	c	d	e	f
-	a	-	e	b	-

Timestamp: 5

# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	2	-	4	3	-

Finalização

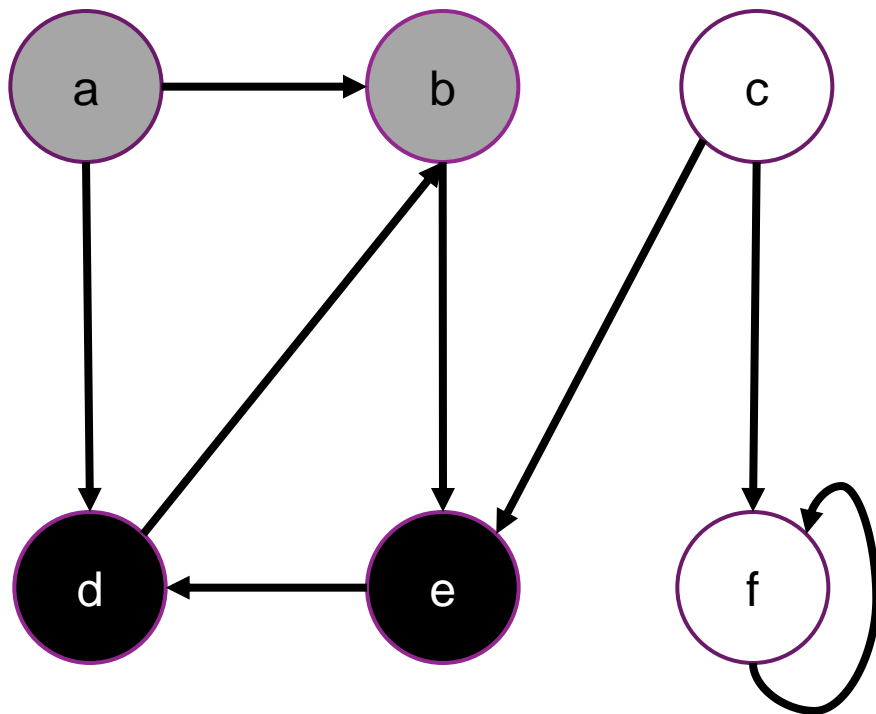
a	b	c	d	e	f
-	-	-	5	-	-

Pais

a	b	c	d	e	f
-	a	-	e	b	-

Timestamp: 5

# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	2	-	4	3	-

Finalização

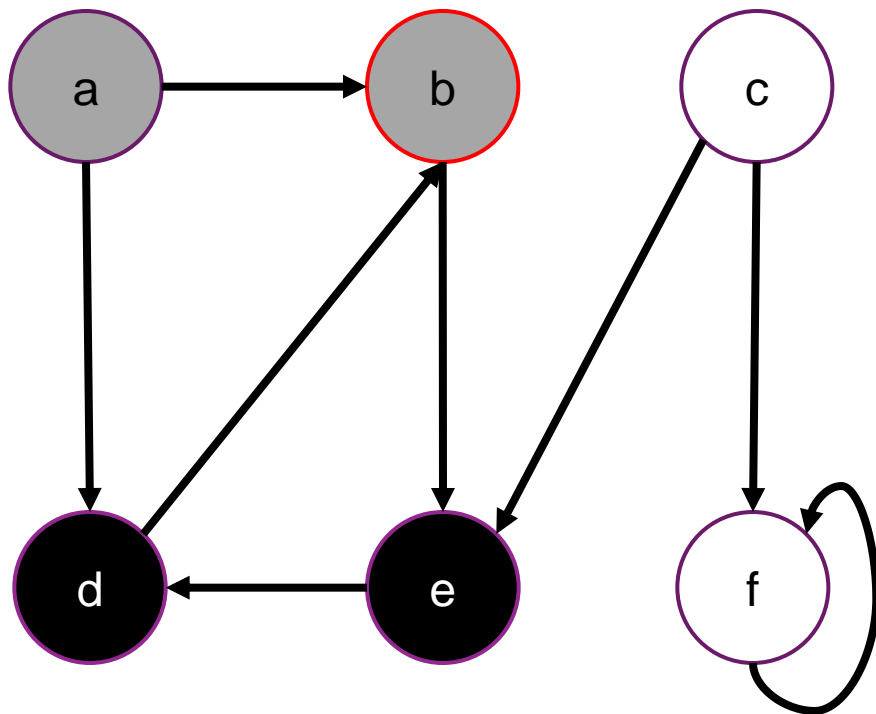
a	b	c	d	e	f
-	-	-	5	6	-

Pais

a	b	c	d	e	f
-	a	-	e	b	-

Timestamp: 6

# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	2	-	4	3	-

Finalização

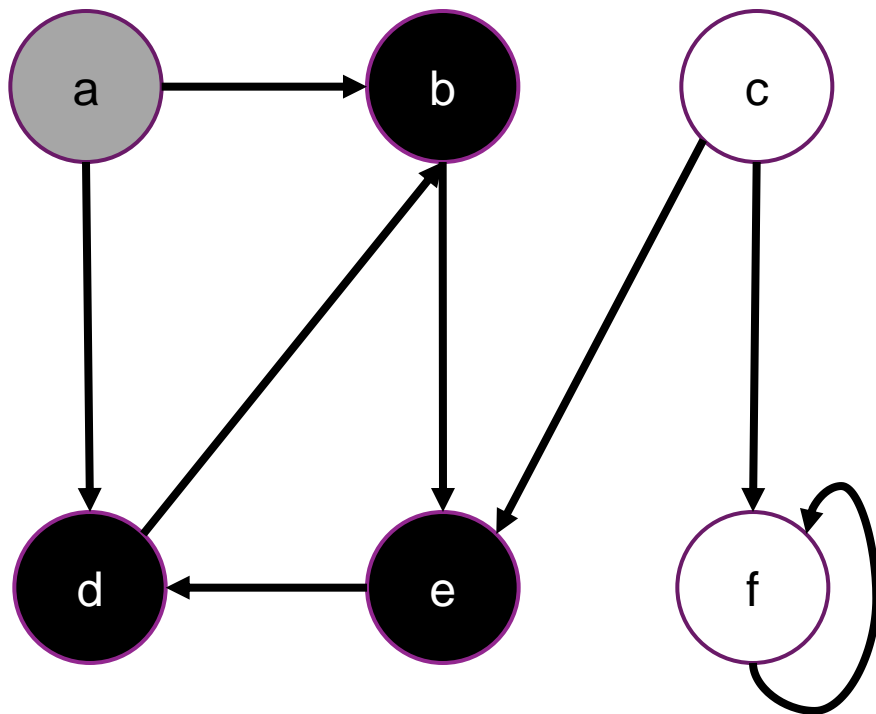
a	b	c	d	e	f
-	-	-	5	6	-

Pais

a	b	c	d	e	f
-	a	-	e	b	-

Timestamp: 6

# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	2	-	4	3	-

Finalização

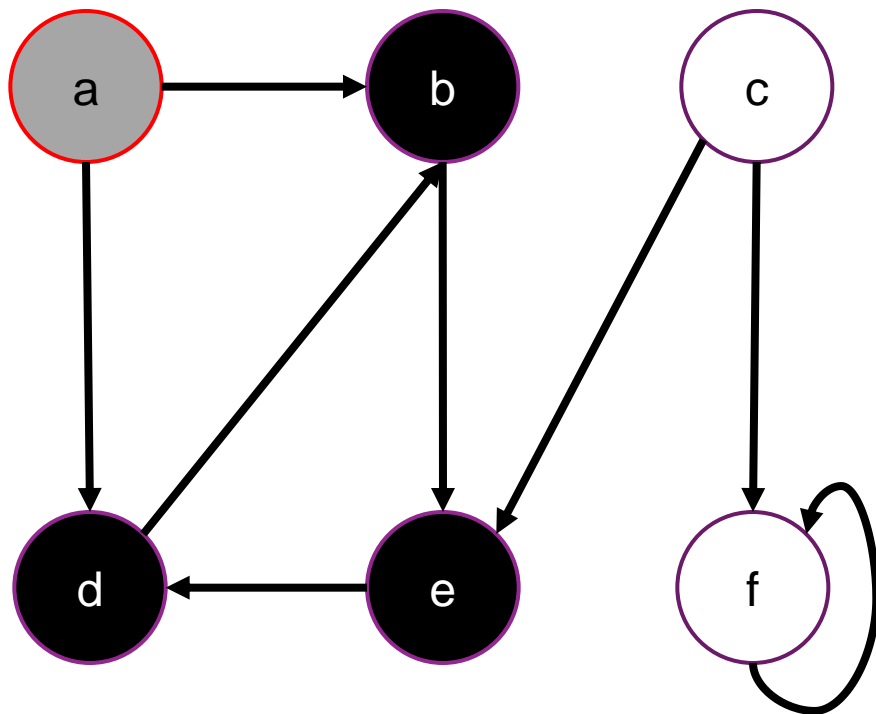
a	b	c	d	e	f
-	7	-	5	6	-

Pais

a	b	c	d	e	f
-	a	-	e	b	-

Timestamp: 7

# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	2	-	4	3	-

Finalização

a	b	c	d	e	f
-	7	-	5	6	-

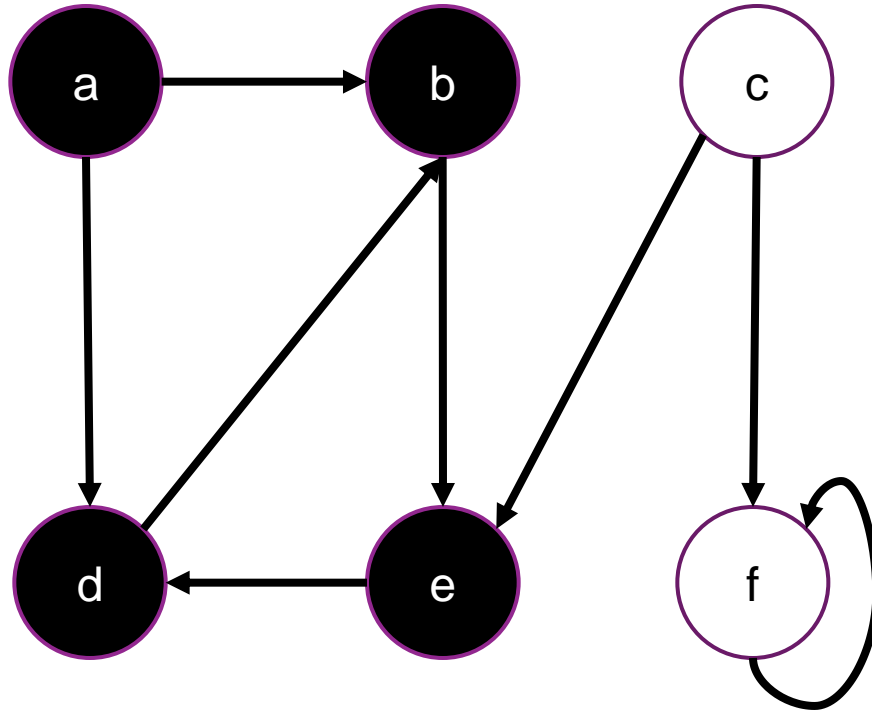
Pais

a	b	c	d	e	f
-	a	-	e	b	-

Timestamp: 7



# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	2	-	4	3	-

Finalização

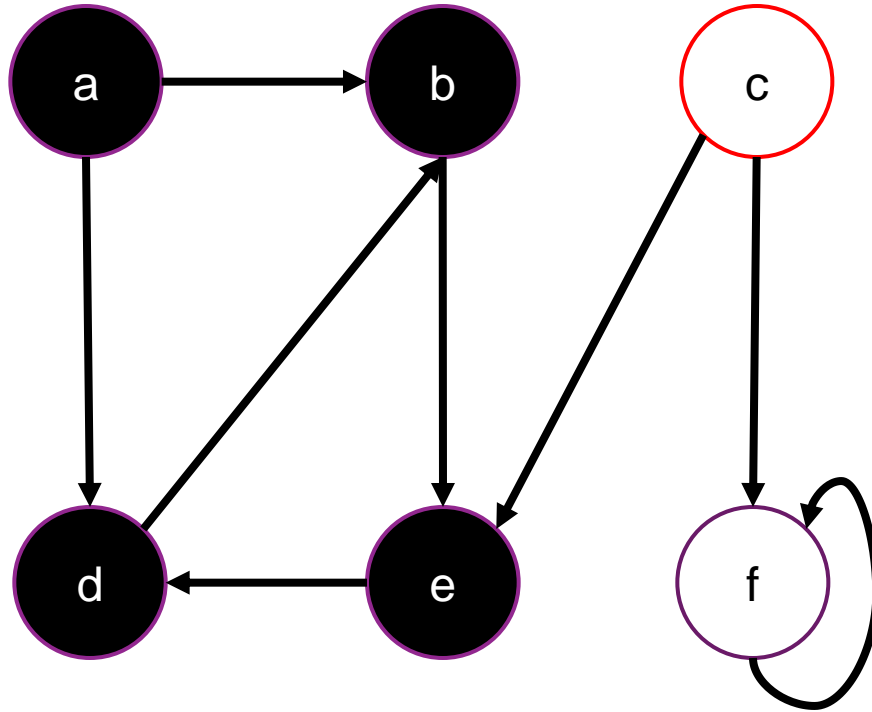
a	b	c	d	e	f
8	7	-	5	6	-

Pais

a	b	c	d	e	f
-	a	-	e	b	-

Timestamp: 8

# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	2	-	4	3	-

Finalização

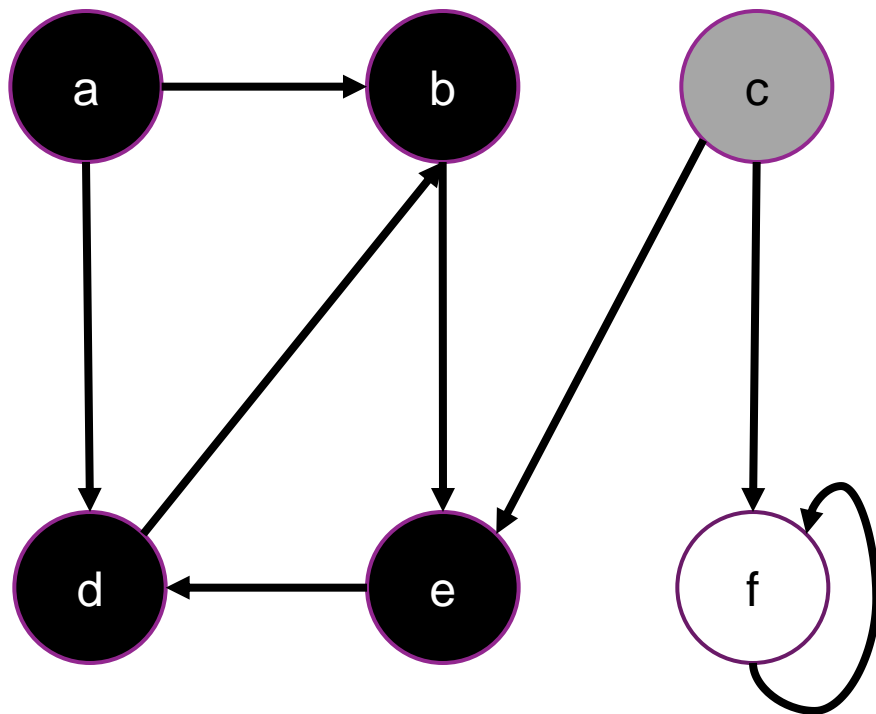
a	b	c	d	e	f
8	7	-	5	6	-

Pais

a	b	c	d	e	f
-	a	-	e	b	-

Timestamp: 8

# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	2	9	4	3	-

Finalização

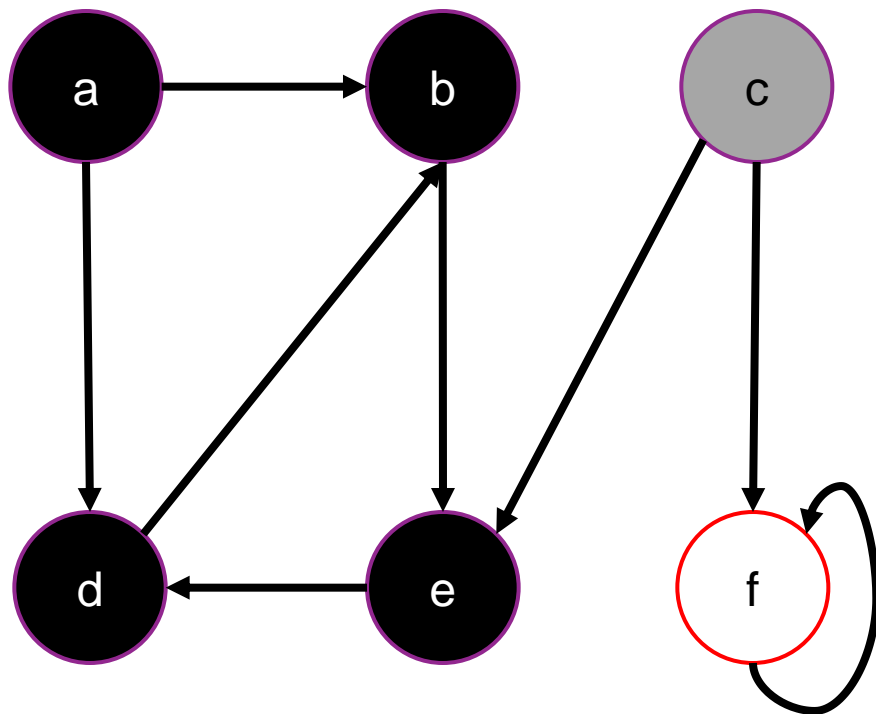
a	b	c	d	e	f
8	7	-	5	6	-

Pais

a	b	c	d	e	f
-	a	-	e	b	-

Timestamp: 9

# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	2	9	4	3	-

Finalização

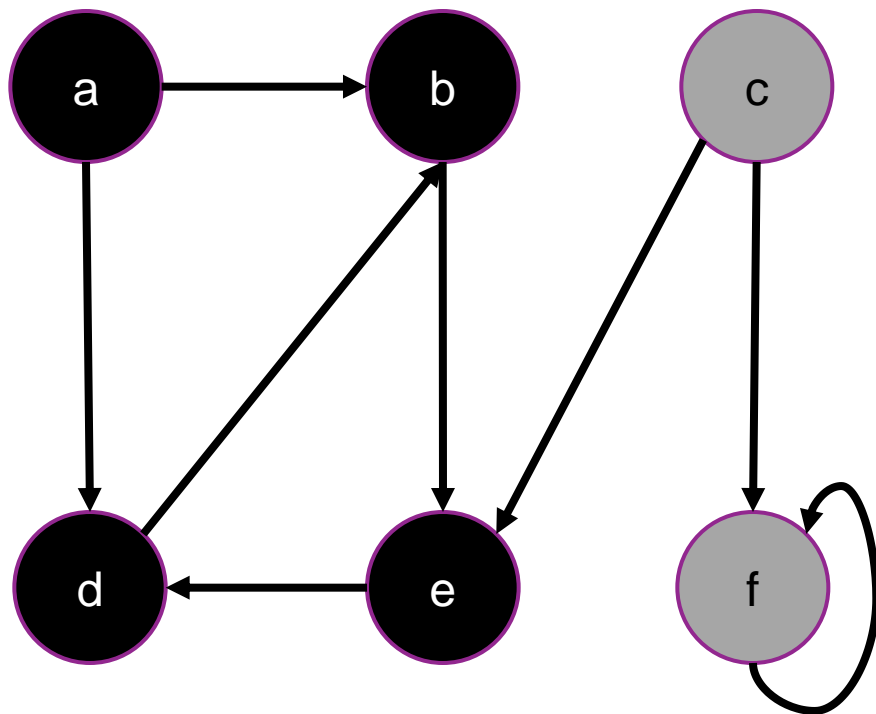
a	b	c	d	e	f
8	7	-	5	6	-

Pais

a	b	c	d	e	f
-	a	-	e	b	c

Timestamp: 9

# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	2	9	4	3	10

Finalização

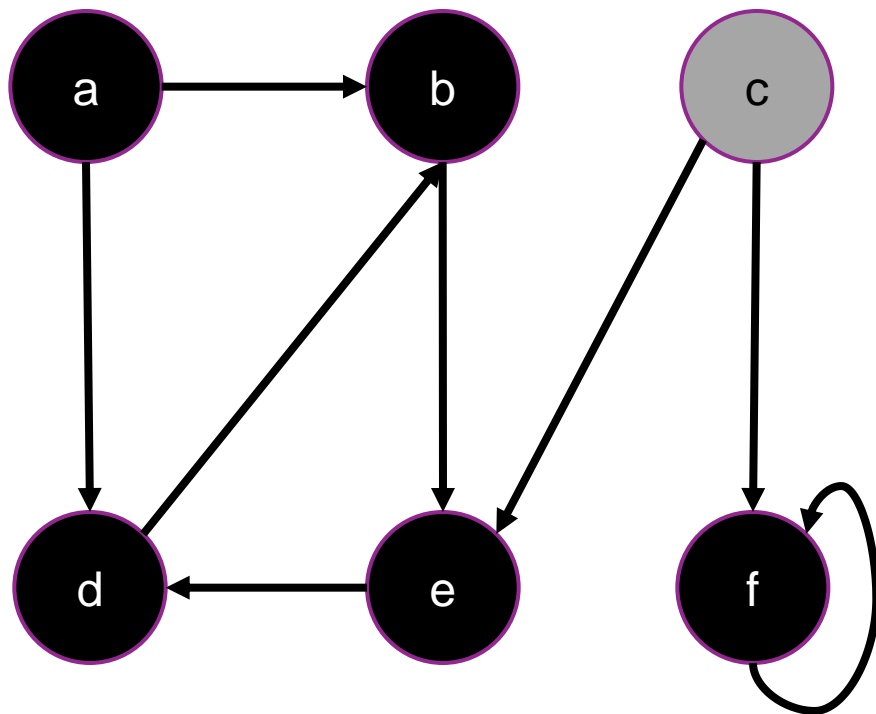
a	b	c	d	e	f
8	7	-	5	6	-

Pais

a	b	c	d	e	f
-	a	-	e	b	c

Timestamp: 10

# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	2	9	4	3	10

Finalização

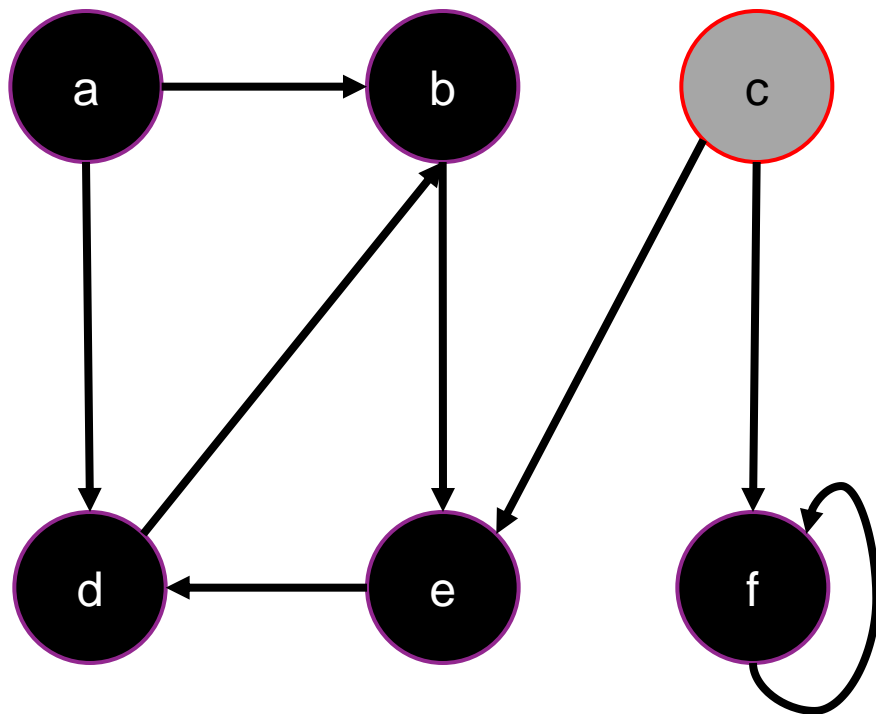
a	b	c	d	e	f
8	7	-	5	6	11

Pais

a	b	c	d	e	f
-	a	-	e	b	c

Timestamp: 11

# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	2	9	4	3	10

Finalização

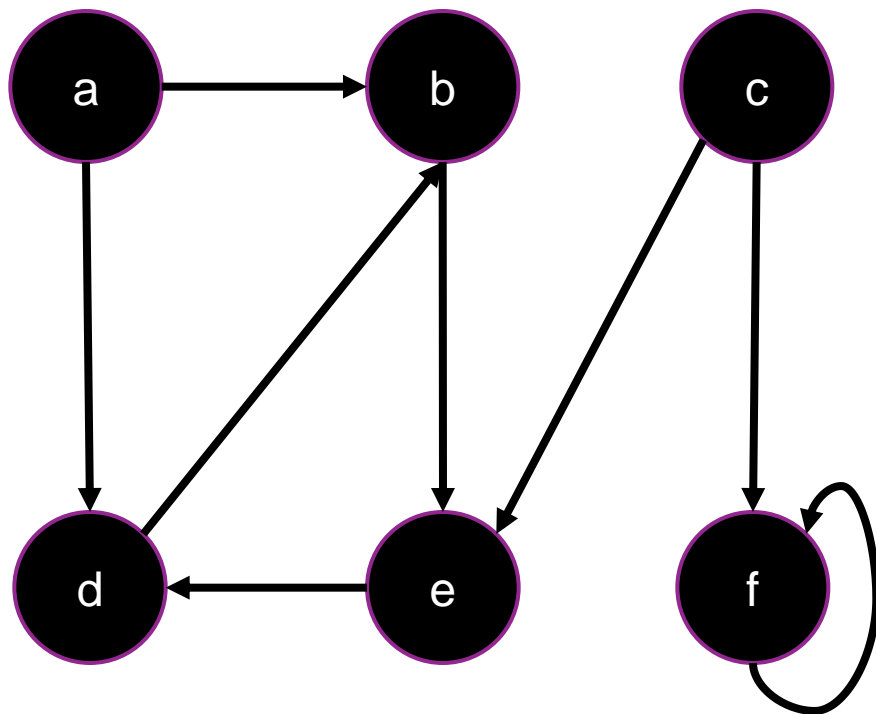
a	b	c	d	e	f
8	7	-	5	6	11

Pais

a	b	c	d	e	f
-	a	-	e	b	c

Timestamp: 11

# Algoritmo DFS



Descoberta

a	b	c	d	e	f
1	2	9	4	3	10

Finalização

a	b	c	d	e	f
8	7	12	5	6	11

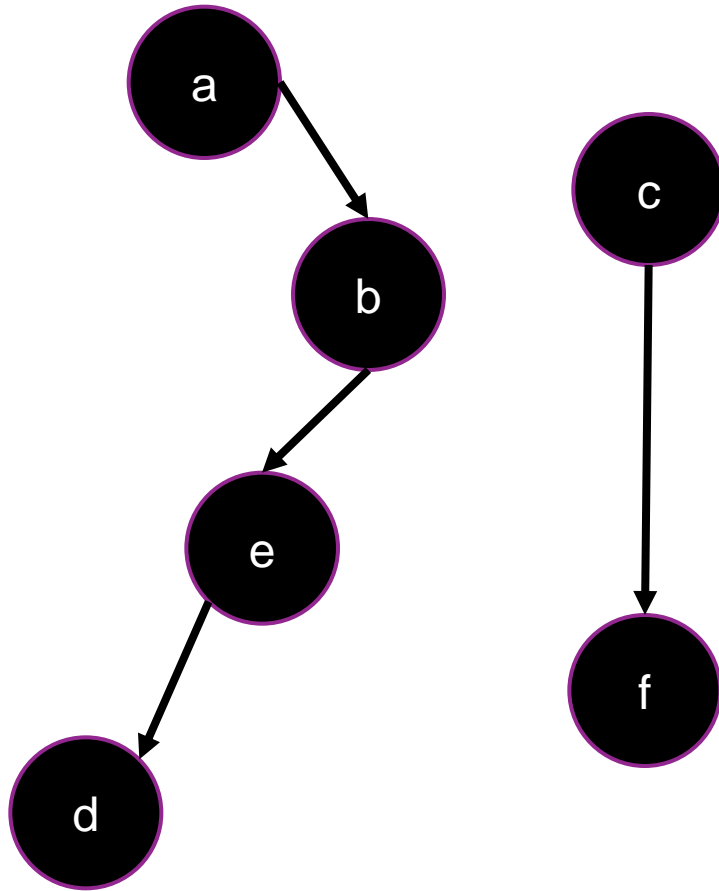
Pais

a	b	c	d	e	f
-	a	-	e	b	c

Timestamp: 12



# Floresta DFS



# Busca em profundidade e PCV

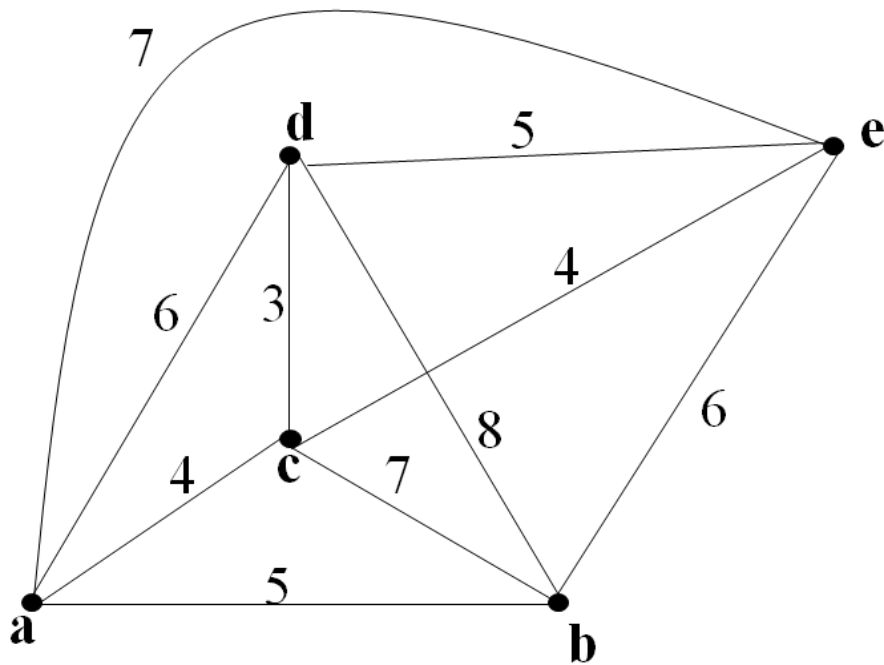
74

- Como adaptar o algoritmo DFS para resolver o problema do Caixeiro Viajante?

# Exercício

75

- Executar, neste grafo:
  - Busca em largura
  - Busca em profundidade
  - PCV
- Todos partindo do vértice *a*



OBRIGADO.

Dúvidas?