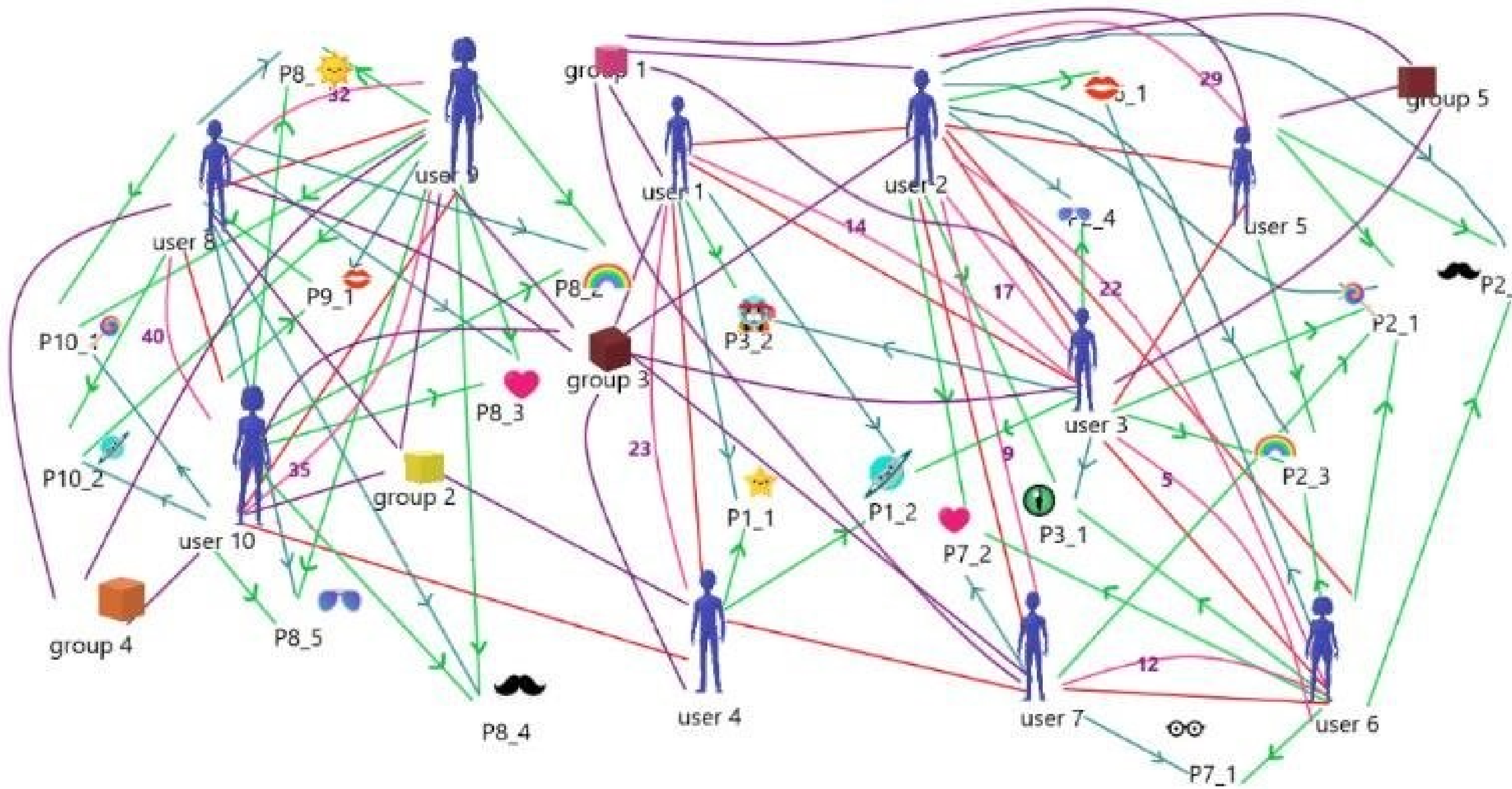
Buscas em grafos

Buscas em grafos

- **Busca em grafos: quando usar?**
  1. Determinar **quais vértices são alcançáveis** através de um vértice inicial.
  2. Verificar se um determinado objeto **está no grafo**.
  3. Identificar **características específicas** de grafos.

- **Adaptações dos algoritmos de busca permitem construir algoritmos que resolvem problemas de:**
  1. Árvore Geradora Mínima (AGM).
  2. Caminho Mínimo (Ex: Dijsktra) e fluxo máximo.
  3. Ordenação Topológica.

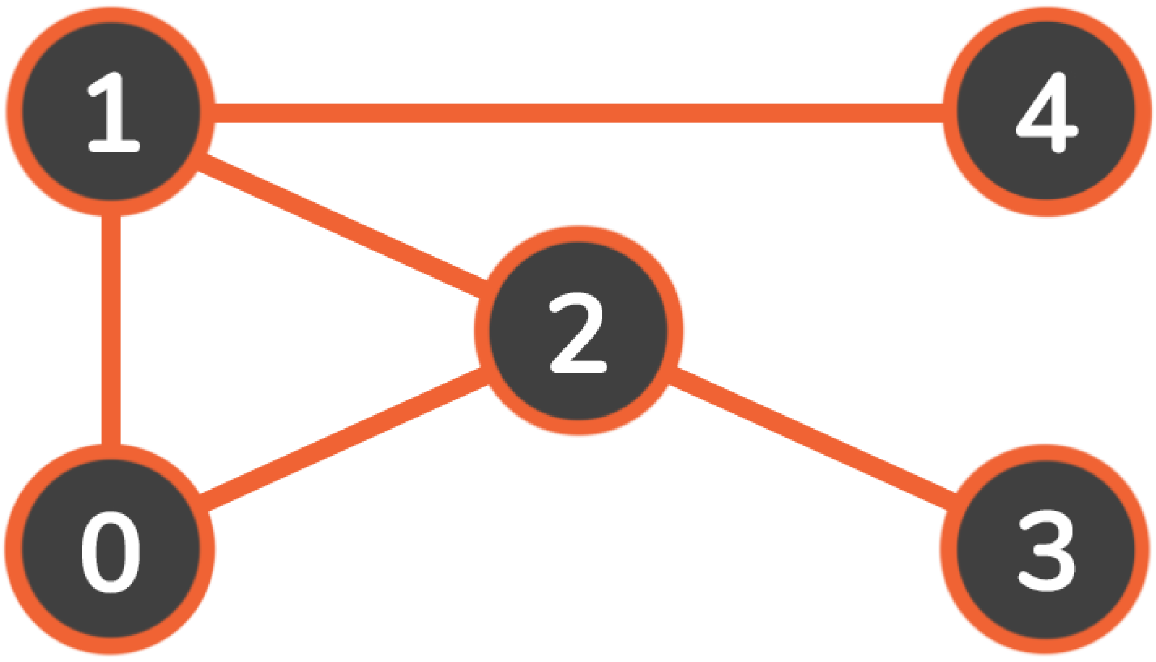
- **Aplicações:**
  - Compiladores.
  - Resolução de problemas complexos (xadrez, por ex.).
  - Função “localizar arquivo” no SO.
  - Análise de redes sociais.



Modelo de grafo do Facebook: Há três tipos de nós: usuários, posts e grupos, além de vários tipos de arestas (amizade – vermelho, reação em posts – verde, etc. Fonte: https://doi.org/10.29322/IJSRP.10.03.2020.p9929

Buscas em grafos

- **Buscas em grafos: algoritmos clássicos**
  - Busca em Profundidade (Depth-First Search – DFS).
  - Buca em Largura (Breadth-First Search – BFS).

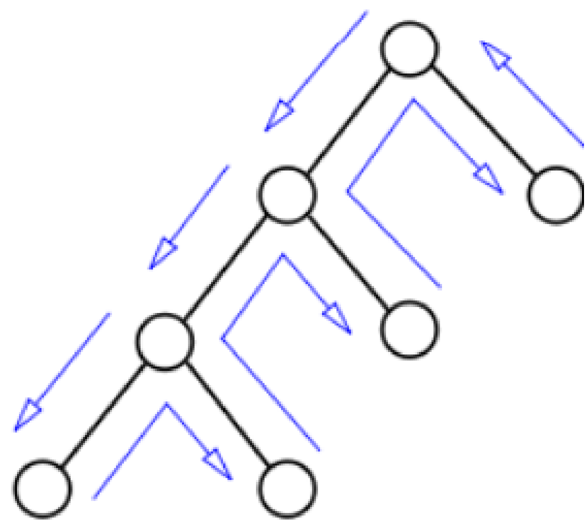


Busca em Profundidade
Depth-First Search – DFS

Buscas em profundidade em grafos

Busca em profundidade (DFS):

- É um algoritmo para caminhar no grafo.
- **Ideia-chave**: buscar, sempre que possível, o mais "**fundo**" no grafo.



- As arestas são exploradas a partir do nó **v** mais recentemente descoberto que ainda não possui nós adjacentes que não foram explorados.

Buscas em profundidade em grafos

Etapas básicas do DFS em grafos:

- Quando todas as arestas adjacentes a **v** tiverem sido exploradas, a busca inicia seu processo de "**andar de volta para trás**" (*backtrack*), de modo a explorar os vértices do qual **v** foi descoberto.
- O processo continua até que sejam descobertos todos os nós que são **alcançáveis a partir do nó original**.
- Algoritmo encerra quanto todos os nós foram descobertos.

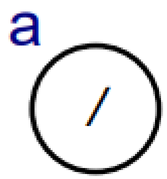
Algoritmo DFS em grafos

Legenda para o algoritmo:

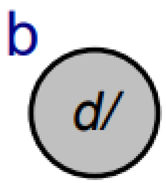
- **Nó Branco**: ainda não visitado.
- **Nó Cinza**: visitado, mas seus nós adjacentes ainda não foram todos visitados.
- **Nó Preto**: visitado, e seus nós adjacentes já foram todos visitados.

Algoritmo DFS em grafos

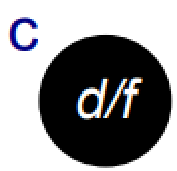
Legenda para controlar a descoberta e finalização



Nó desconhecido



Nó encontrado



Nó encontrado, com fecho transitivo totalmente visitado

- ***d***: marcador do instante em que o nó **c** foi descoberto.
- ***f***: marcador do instante em que o fecho transitivo do nó **c** foi totalmente visitado (considerado então **finalizado**).

Algoritmo DFS em grafos

Input: Grafo $G = G(V, E)$

DFS(G)

Para cada nó $u \in V$:

$cor[u] =$ BRANCO

$tempo = 0$

Para cada nó $u \in V$:

se $cor[u] =$ BRANCO

$DSF\_VISIT(u)$

$DSF\_VISIT(u)$

$cor[u] =$ CINZA

$tempo = tempo + 1$

$d[u] = tempo$

Para cada nó $v \in Adj(u)$

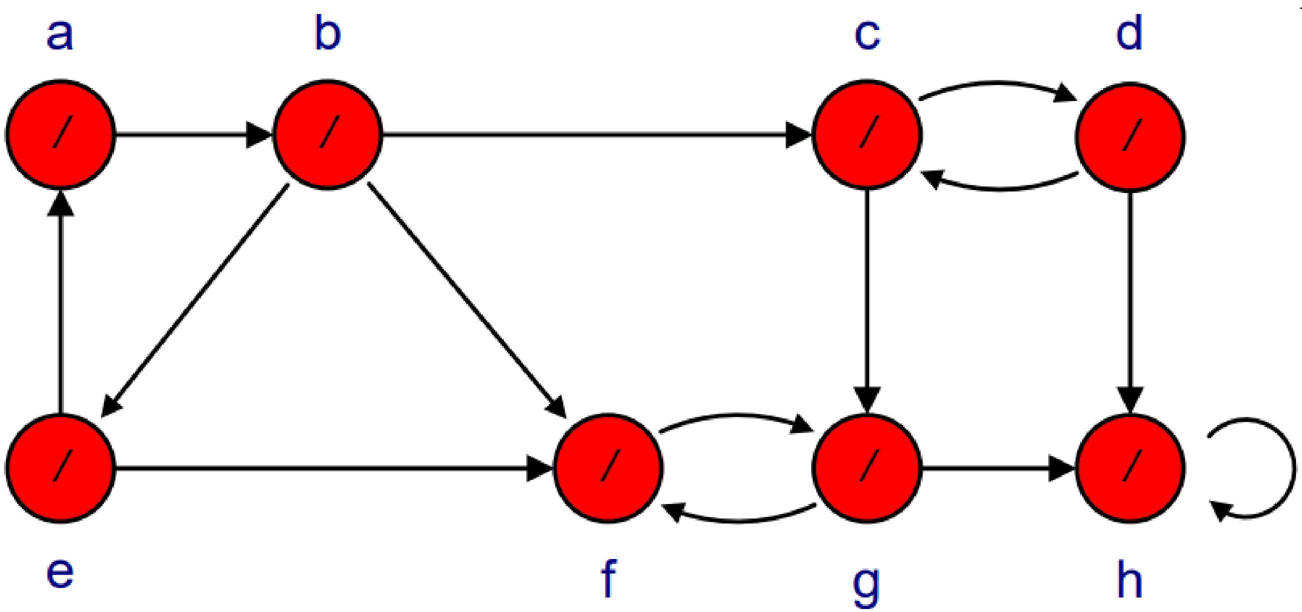
se $cor[v] =$ BRANCO

$DSF\_VISIT(v)$

$cor[u] =$ PRETO

$f[u] = tempo = tempo + 1$

Algoritmo DFS em grafos - exemplo



DFS(G)

Para cada nó $u \in V$:

$cor[u] = \text{BRANCO}$

$tempo = 0$

Para cada nó $u \in V$:

se $cor[u] = \text{BRANCO}$

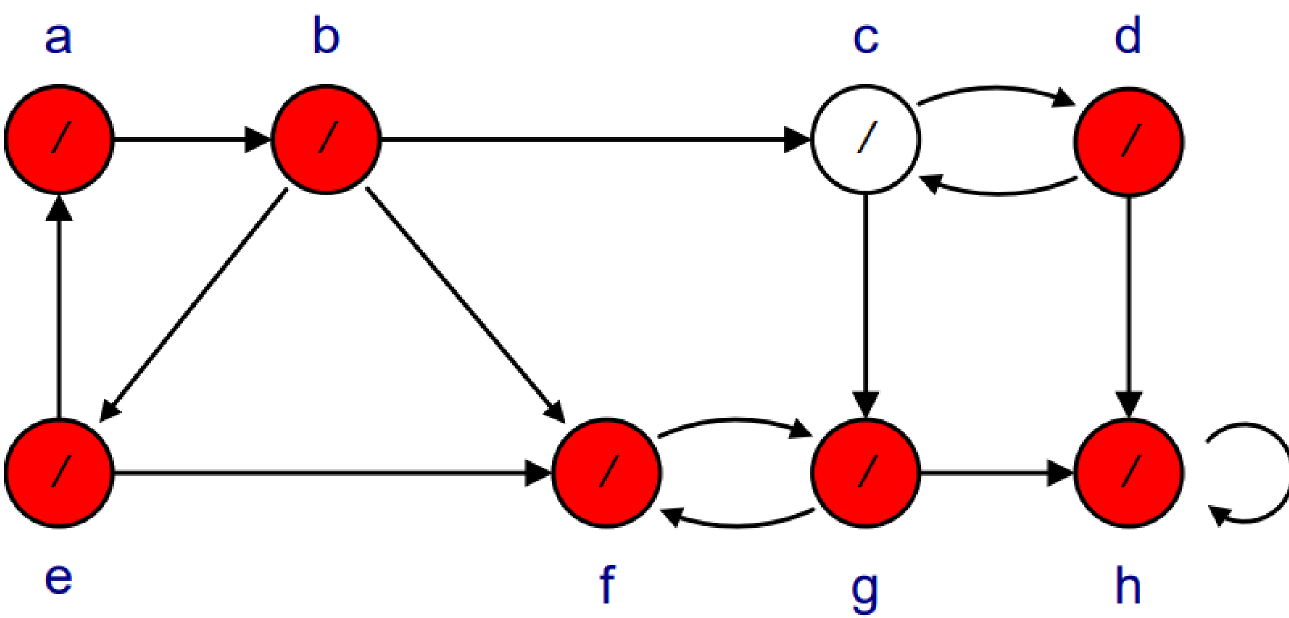
$DSF\_VISIT(u)$

Lista: [c, a, b, d, e, f, g, h]

- Dado um Grafo G, temos uma lista de todos os seus nós

Algoritmo DFS em grafos - exemplo

- Colorindo o nó **c** de **BRANCO**



DFS(G)

Para cada nó $u \in V$:

$cor[u] = \text{BRANCO}$

$tempo = 0$

Para cada $u \in V$

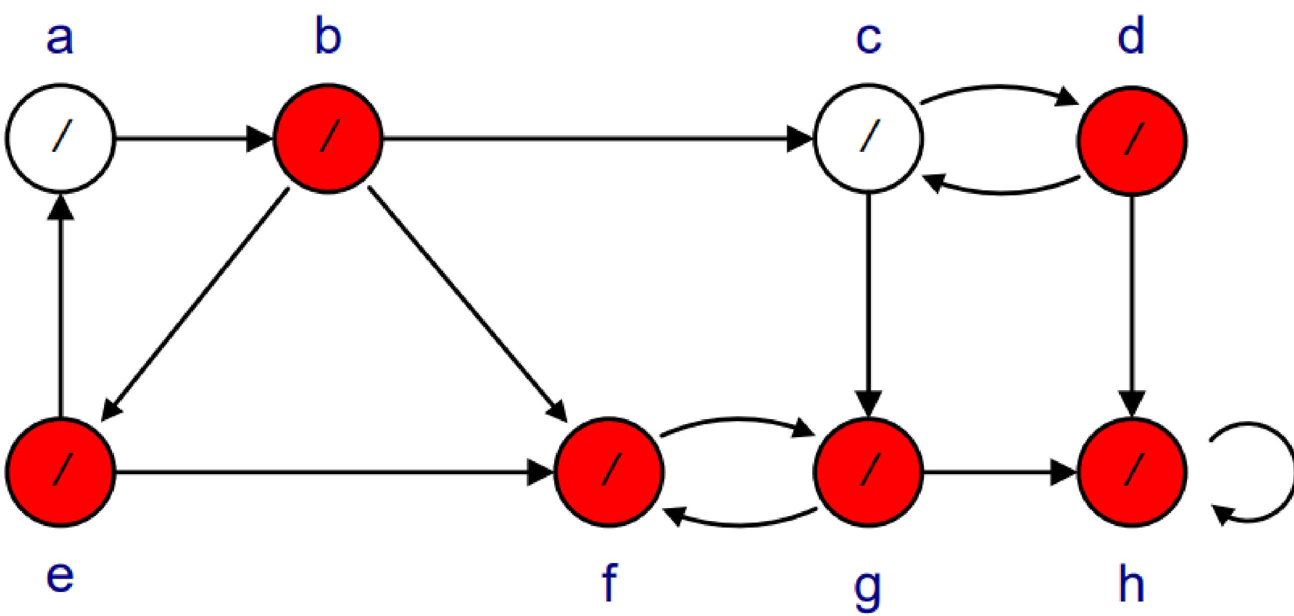
se $cor[u] = \text{BRANCO}$

$DSF\_VISIT(u)$

Lista: [**c**, a, b, d, e, f, g, h]

Algoritmo DFS em grafos - exemplo

- Colorindo o nó **a** de **BRANCO**



DFS(G)

Para cada nó $u \in V$:
$cor[u] = \text{BRANCO}$

$tempo = 0$

Para cada $u \in V$

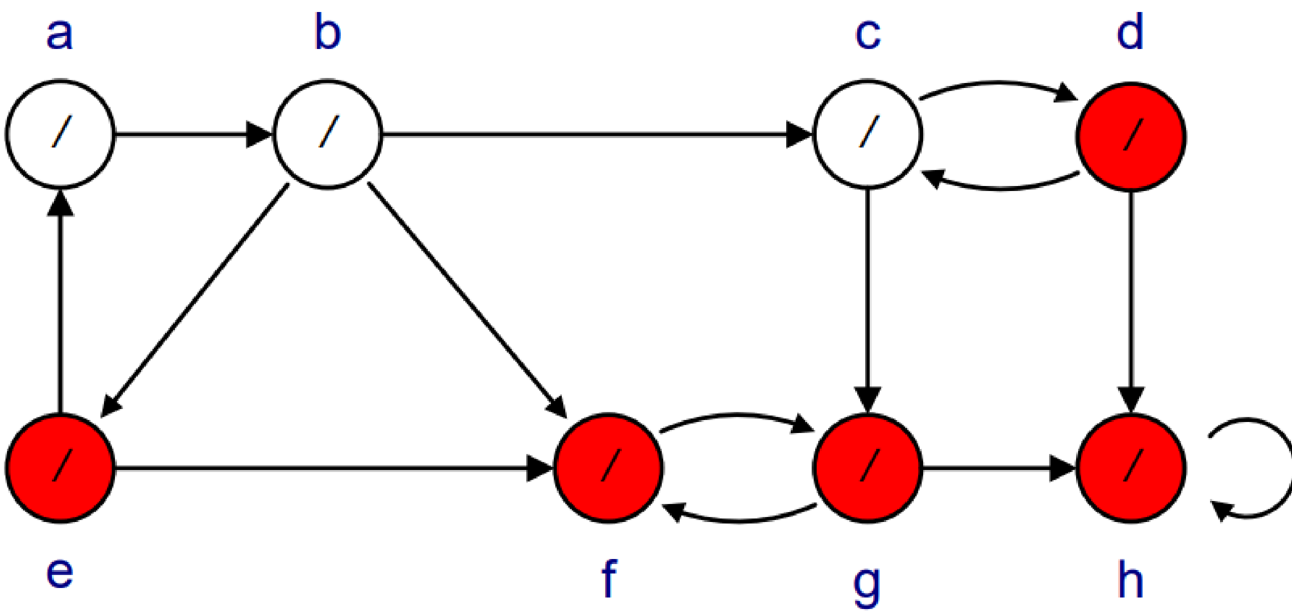
se $cor[u] = \text{BRANCO}$

$DSF\_VISIT(u)$

Lista: [c, **a**, b, d, e, f, g, h]

Algoritmo DFS em grafos - exemplo

- Colorindo o nó **b** de **BRANCO**



DFS(G)

$$\boxed{\begin{array}{l}\text{Para cada nó } u \in V: \\ \quad cor[u] = \text{BRANCO}\end{array}}$$

$tempo = 0$

Para cada $u \in V$

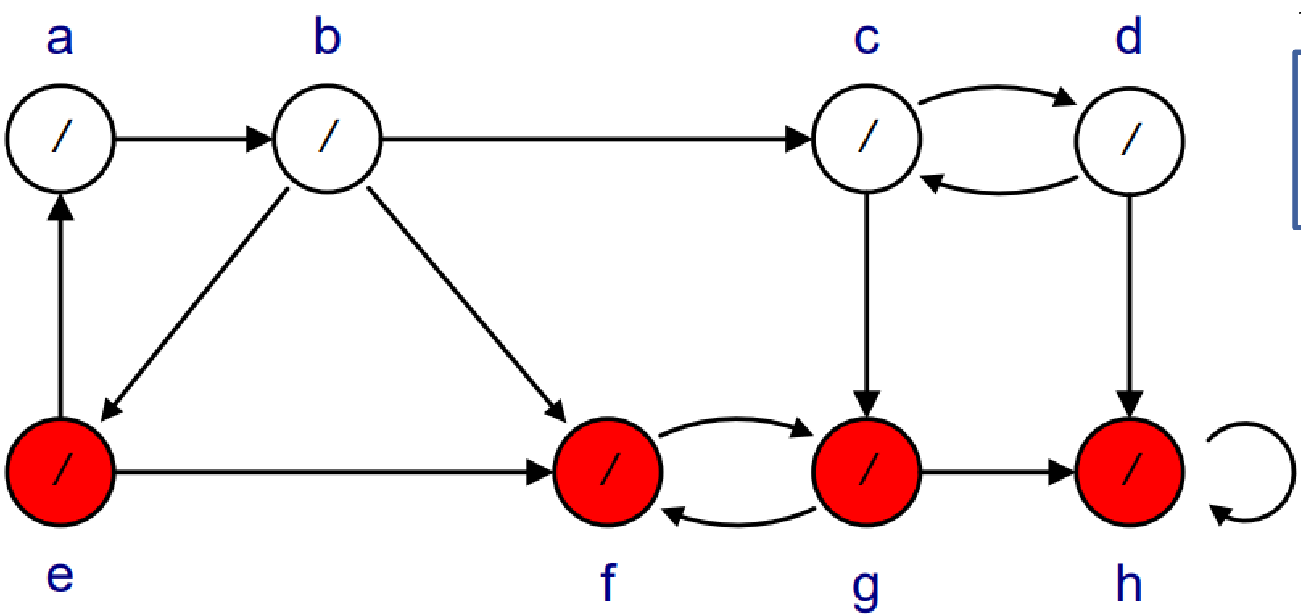
se $cor[u] = \text{BRANCO}$

$DSF\_VISIT(u)$

Lista: [c, a, **b**, d, e, f, g, h]

Algoritmo DFS em grafos - exemplo

- Colorindo o nó **d** de **BRANCO**



DFS(G)

Para cada nó $u \in V$:

$cor[u] = \text{BRANCO}$

$tempo = 0$

Para cada $u \in V$

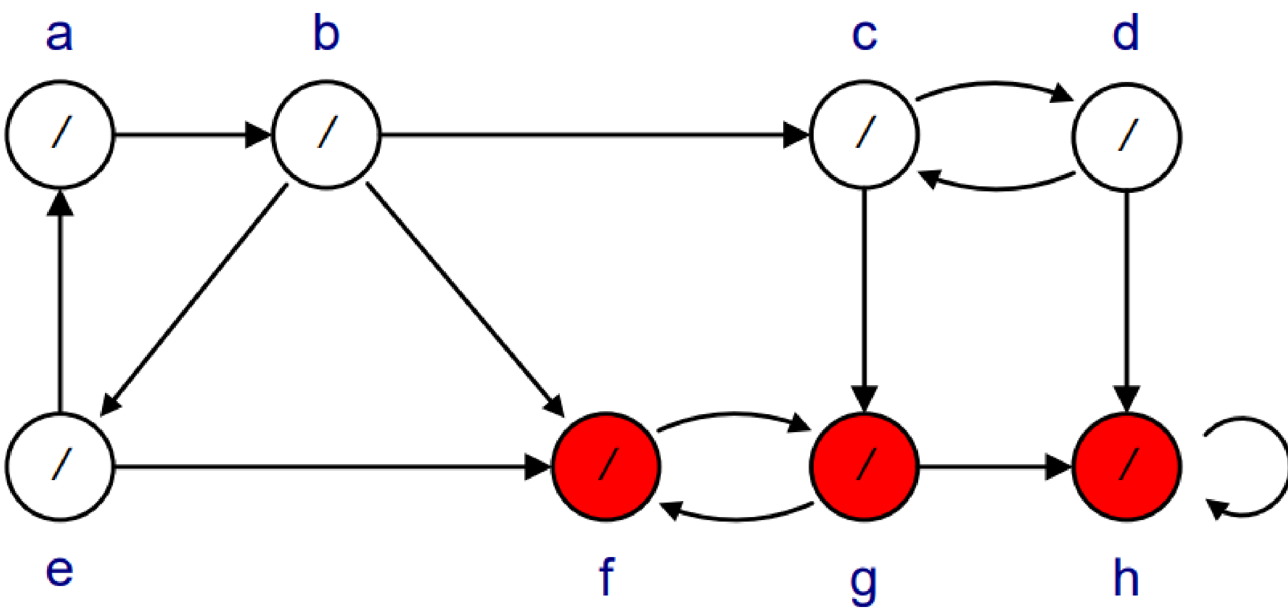
se $cor[u] = \text{BRANCO}$

$DSF\_VISIT(u)$

Lista: [c, a, b, **d**, e, f, g, h]

Algoritmo DFS em grafos - exemplo

- Colorindo o nó **e** de **BRANCO**



DFS(G)

Para cada nó $u \in V$:

$cor[u] = \text{BRANCO}$

$tempo = 0$

Para cada $u \in V$

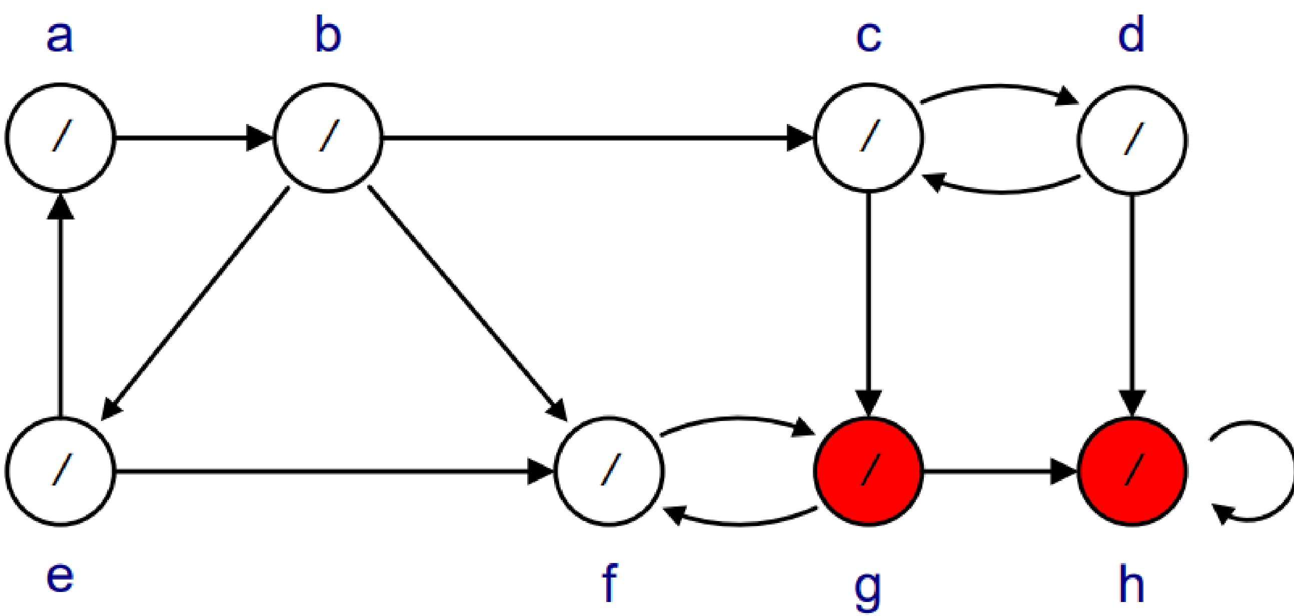
se $cor[u] = \text{BRANCO}$

$DSF\_VISIT(u)$

Lista: [c, a, b, d, **e**, f, g, h]

Algoritmo DFS em grafos - exemplo

- Colorindo o nó **f** de **BRANCO**



DFS(G)

Para cada nó $u \in V$:

$cor[u] = \text{BRANCO}$

$tempo = 0$

Para cada $u \in V$

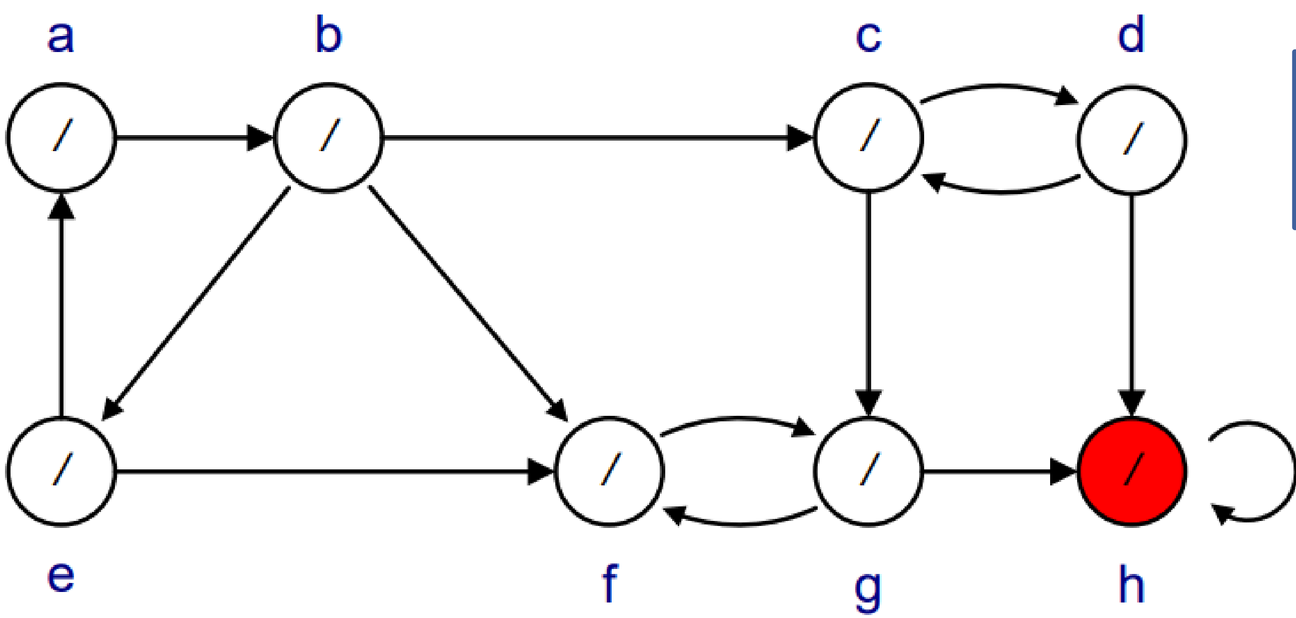
se $cor[u] = \text{BRANCO}$

$DSF\_VISIT(u)$

Lista: [c, a, b, d, e, **f**, g, h]

Algoritmo DFS em grafos - exemplo

- Colorindo o nó **g** de **BRANCO**



DFS(G)

Para cada nó $u \in V$:

$cor[u] = \text{BRANCO}$

$tempo = 0$

Para cada $u \in V$

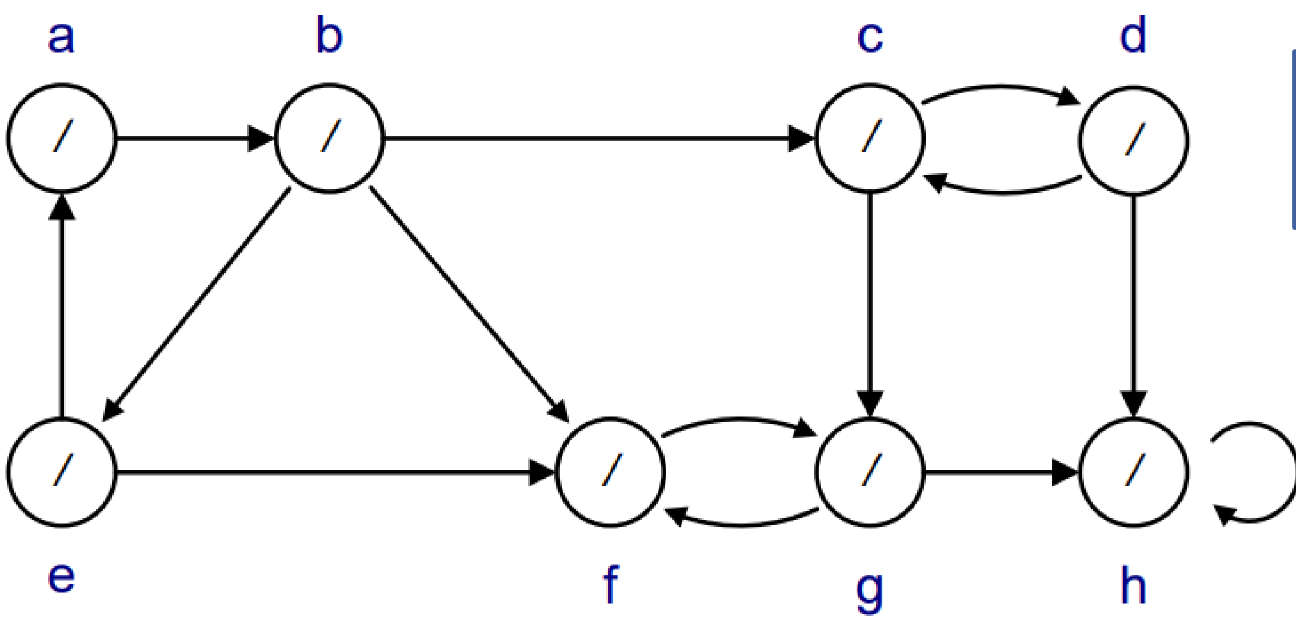
se $cor[u] = \text{BRANCO}$

$DSF\_VISIT(u)$

Lista: [c, a, b, d, e, f, **g**, h]

Algoritmo DFS em grafos - exemplo

- Colorindo o nó **h** de **BRANCO**



DFS(G)

Para cada nó $u \in V$:

$cor[u] = \text{BRANCO}$

$tempo = 0$

Para cada $u \in V$

se $cor[u] = \text{BRANCO}$

$DSF\_VISIT(u)$

Lista: [c, a, b, d, e, f, g, **h**]

Algoritmo DFS em grafos - exemplo

- Iniciando a variável *tempo*

- Iniciando a variável tempo

DFS(G)

  Para cada nó $u \in V$:

    $cor[u] = \text{BRANCO}$

  $\boxed{tempo = 0}$

  Para cada $u \in V$

    se $cor[u] = \text{BRANCO}$

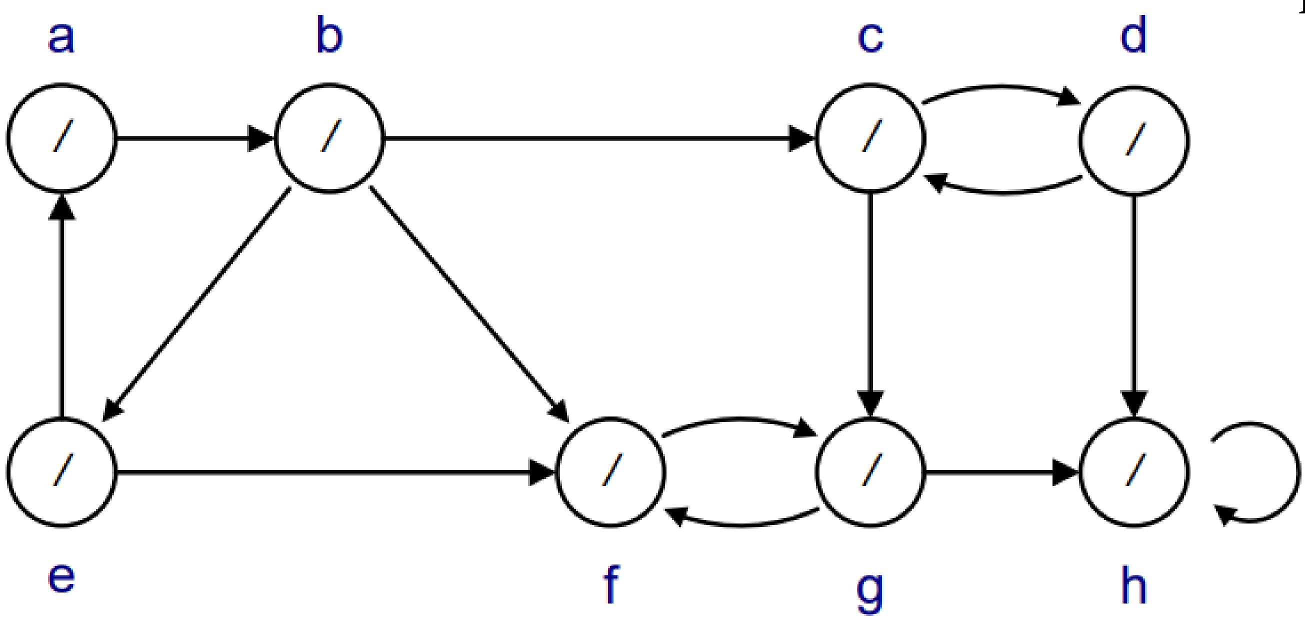
      $DSF\_VISIT(u)$

Lista: [c, a, b, d, e, f, g, h]

$tempo = 0$

Algoritmo DFS em grafos - exemplo

- Para todos os nós do grafo …



DFS(G)

Para cada nó $u \in V$:

$cor[u] = \text{BRANCO}$

$tempo = 0$

Para cada $u \in V$

se $cor[u] = \text{BRANCO}$

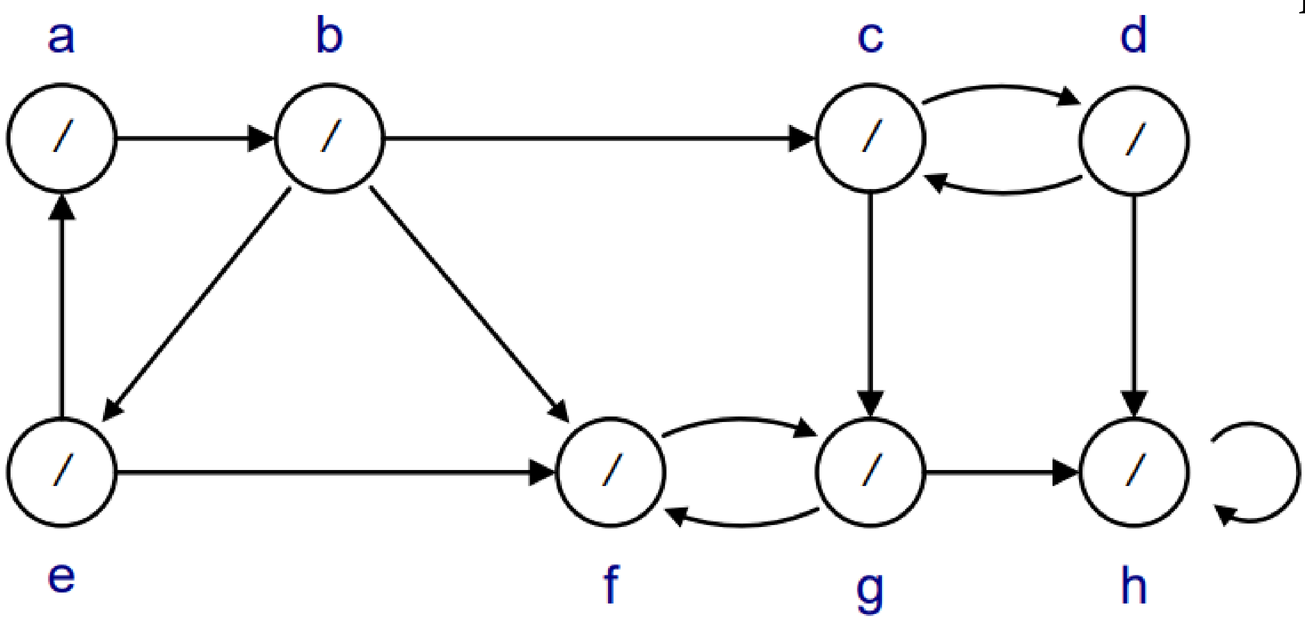
$DSF\_VISIT(u)$

Lista: [**c**, a, b, d, e, f, g, h]

$$tempo = 0$$

Algoritmo DFS em grafos - exemplo

- Cor do nó **c** é BRANCA?



DFS(G)

Para cada nó $u \in V$:

$cor[u] = \text{BRANCO}$

$tempo = 0$

Para cada $u \in V$

se $cor[u] = \text{BRANCO}$

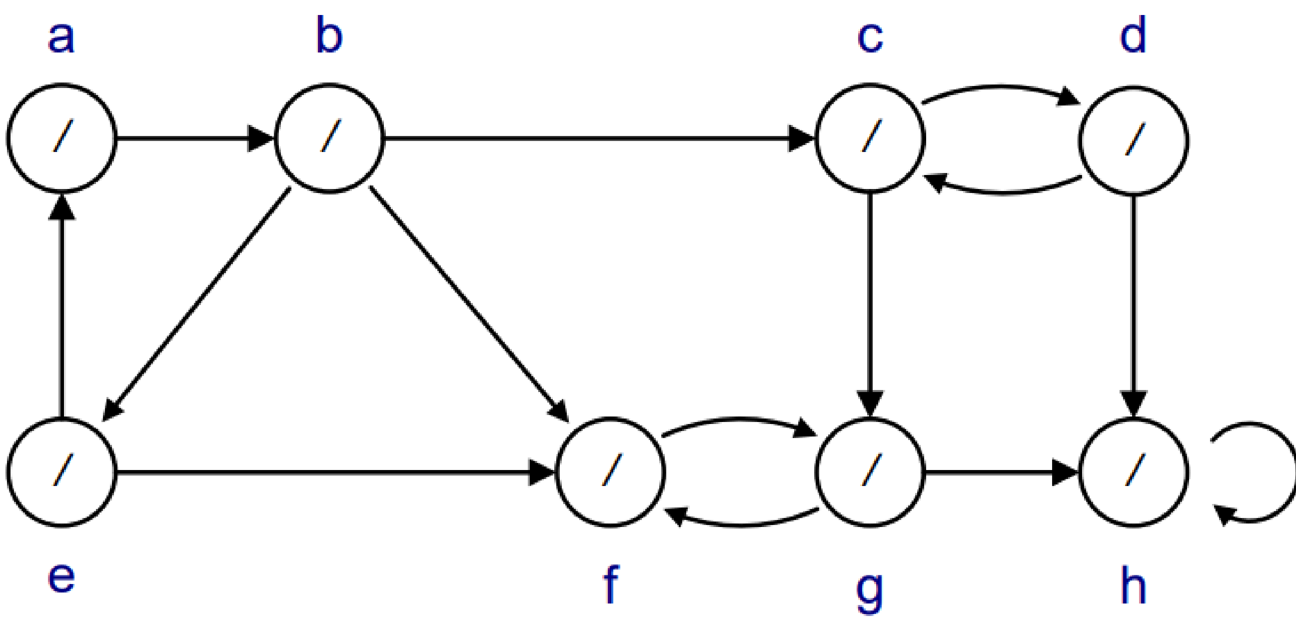
$DSF\_VISIT(u)$

Lista: [**c**, a, b, d, e, f, g, h]

$$tempo = 0$$

Algoritmo DFS em grafos - exemplo

- Chama função $DSF\_VISIT(u = \boldsymbol{c})$
- Empilha a função principal $DSF(G)$ com próximo u=**a**



DFS(G)

Para cada nó $u \in V$:

$cor[u] = \text{BRANCO}$

$tempo = 0$

Para cada $u \in V$

se $cor[u] = \text{BRANCO}$

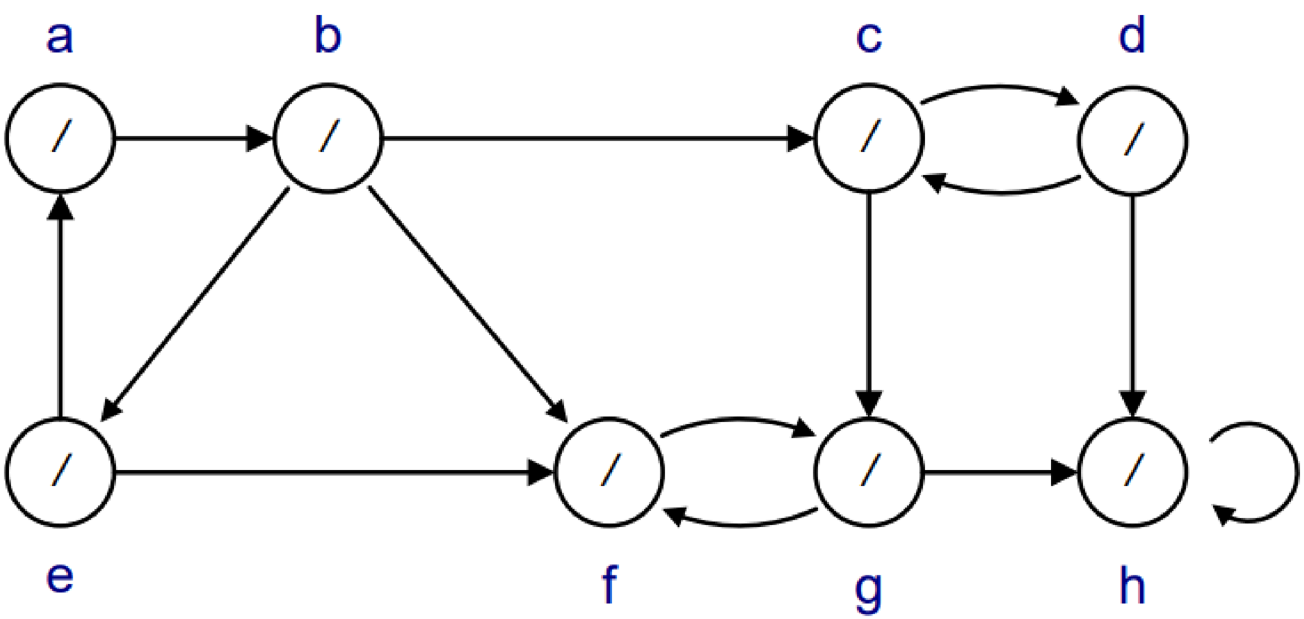
$\boxed{DSF\_VISIT(u)}$

Lista: [**c**, a, b, d, e, f, g, h]

$tempo = 0$

Algoritmo DFS em grafos - exemplo

- Chama função $DSF\_VISIT(u = \boldsymbol{c})$



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

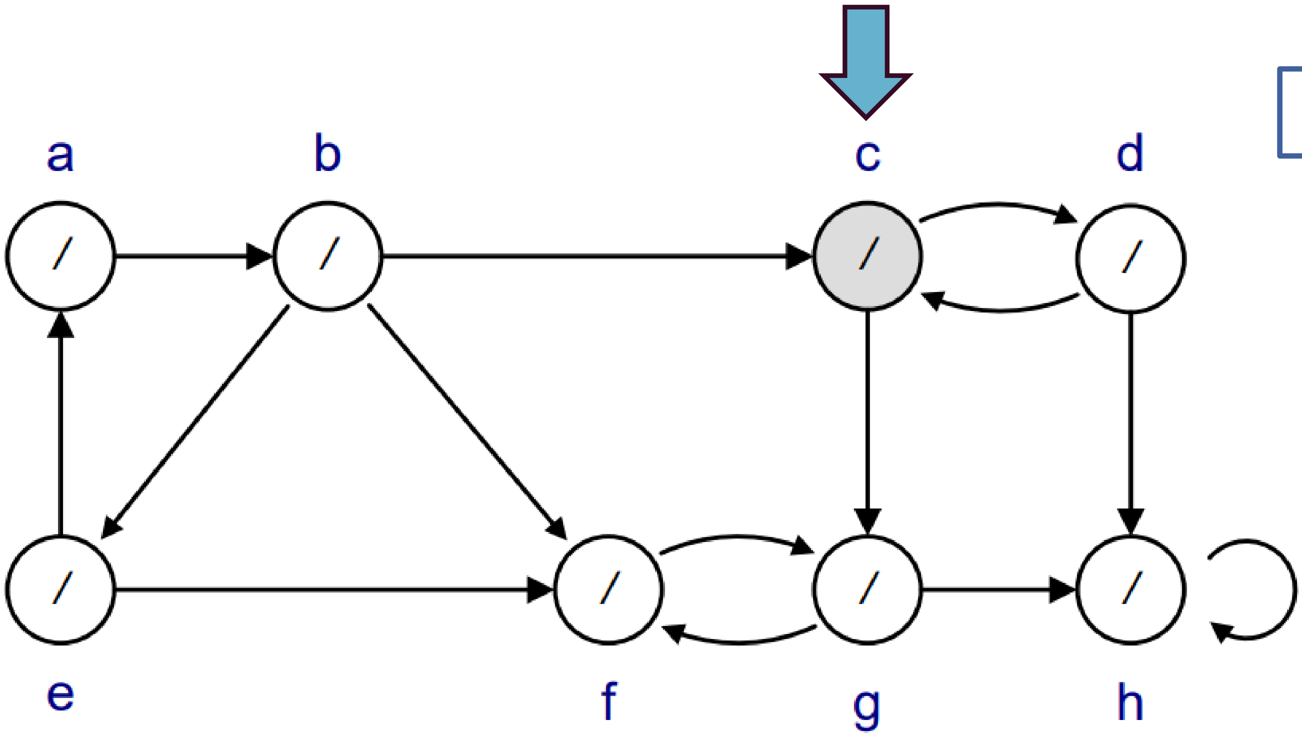
Lista: [c, a, b, d, e, f, g, h]

- Pilha de execução:
  DFS(G) - próximo u=**a** (Para)

$tempo = 0$

Algoritmo DFS em grafos - exemplo

- Cor u=**c** de CINZA



$DSF\_VISIT(u)$

$\boxed{cor[u] = \text{CINZA}}$

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

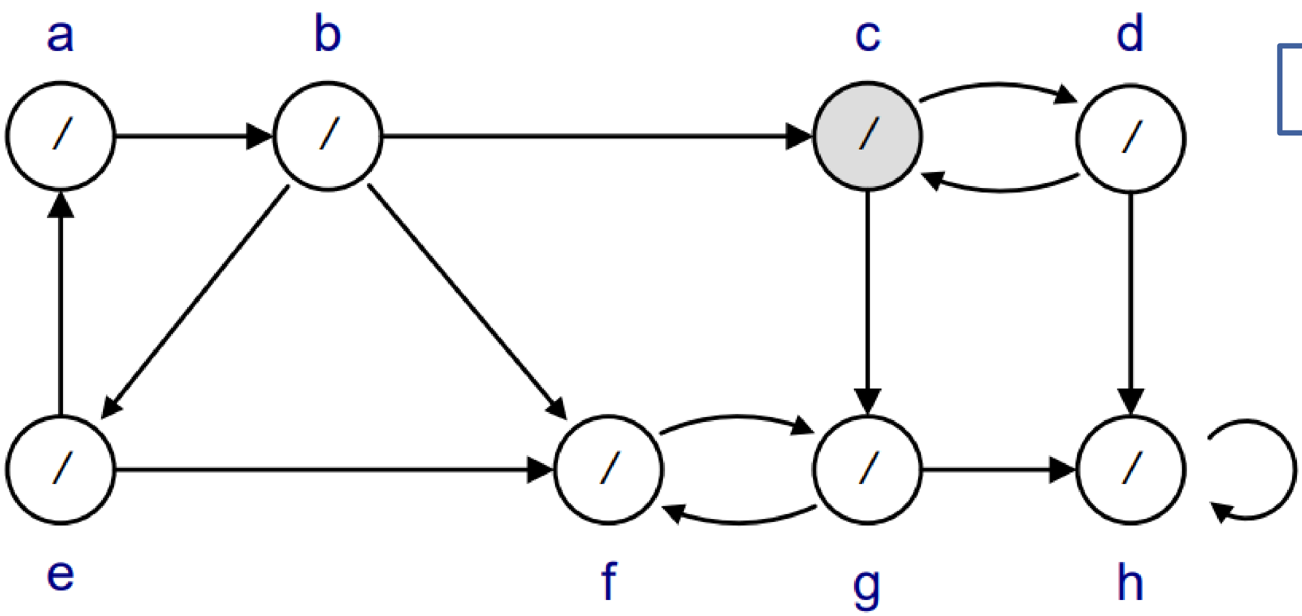
Lista: [c, a, b, d, e, f, g, h]

- Pilha de execução:
  DFS(G) - próximo u=**a** (Para)

$$tempo = 0$$

Algoritmo DFS em grafos - exemplo

- Incrementa o tempo.



$DSF\_VISIT(u)$

$cor[u]$ = CINZA

$\boxed{tempo = tempo + 1}$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v]$ = BRANCO

$DSF\_VISIT(v)$

$cor[u]$ = PRETO

$f[u] = tempo = tempo + 1$

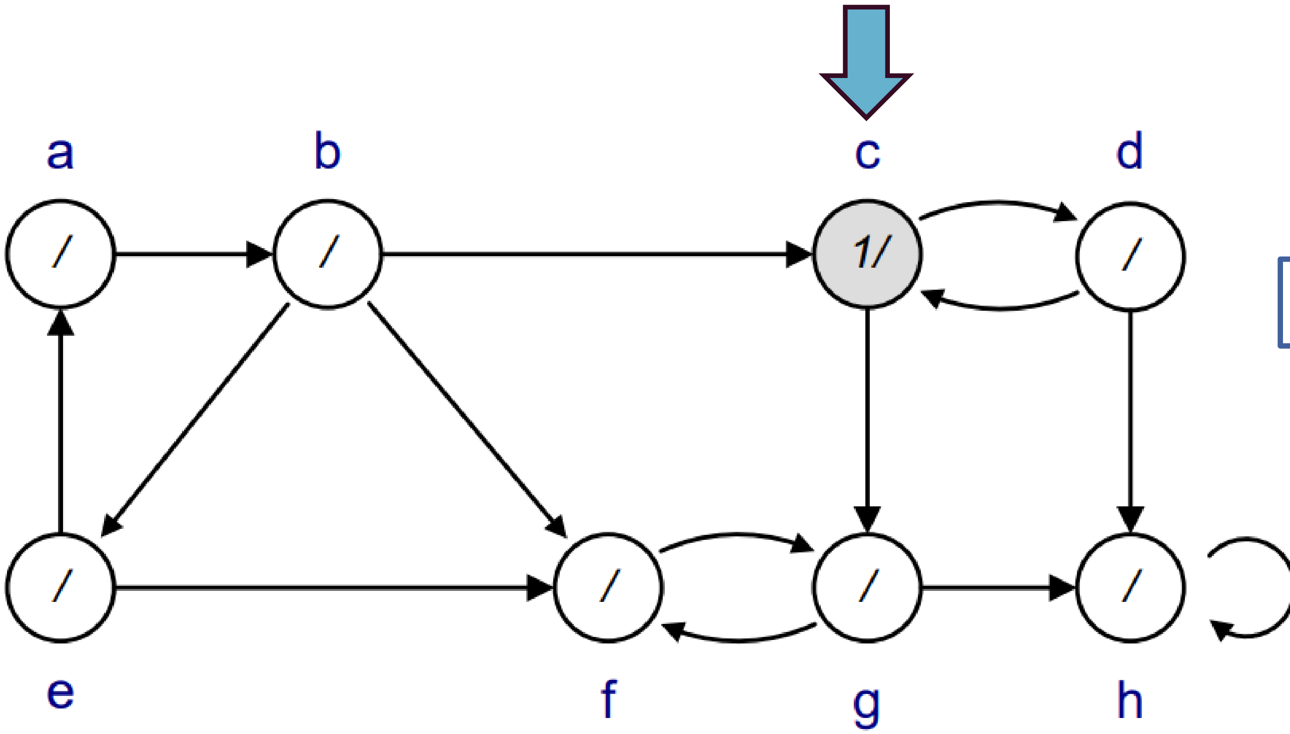
Lista: [c, a, b, d, e, f, g, h]

- Pilha de execução: DFS(G) - próximo u=**a** (Para)

$\Rightarrow$ $tempo = 1$

Algoritmo DFS em grafos - exemplo

- Indica o tempo de descoberta do nó **c**



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$\boxed{d[u] = tempo}$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

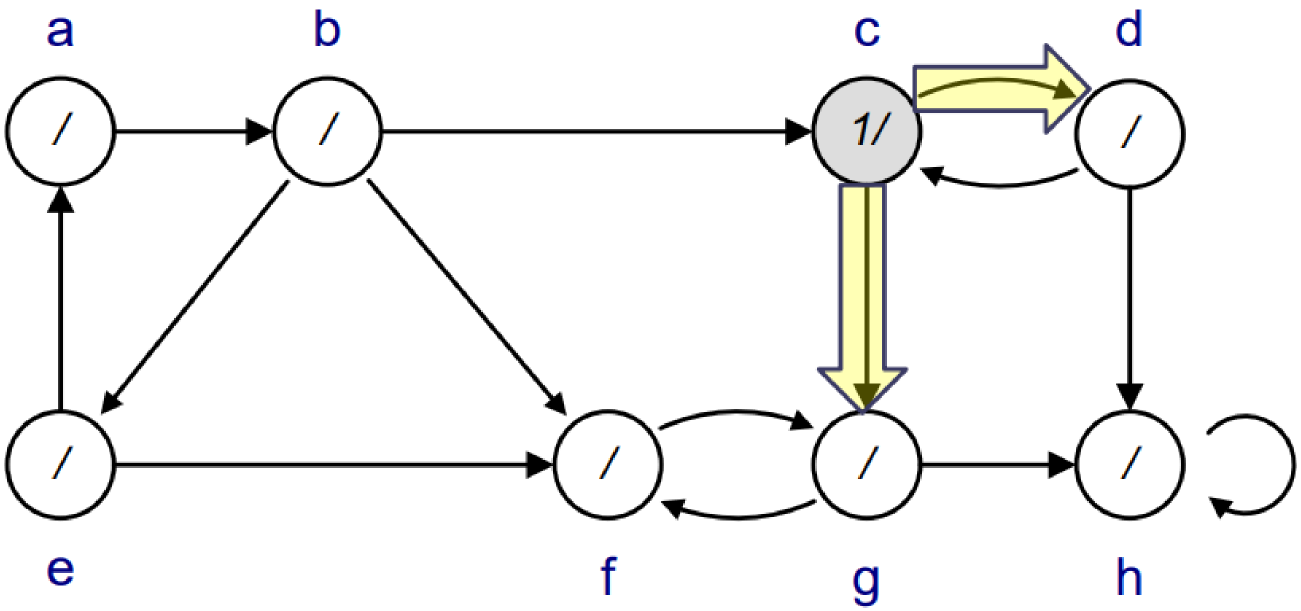
Lista: [c, a, b, d, e, f, g, h]

- Pilha de execução:
  DFS(G) - próximo u=**a** (Para)

$$tempo = 1$$

Algoritmo DFS em grafos - exemplo

- Escolha da ordem em Adj(c) - depende da representação computacional usada)
- Vamos considerar primeiro **g**, depois **d**



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

$\boxed{\text{Para cada } v \in Adj(u)}$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

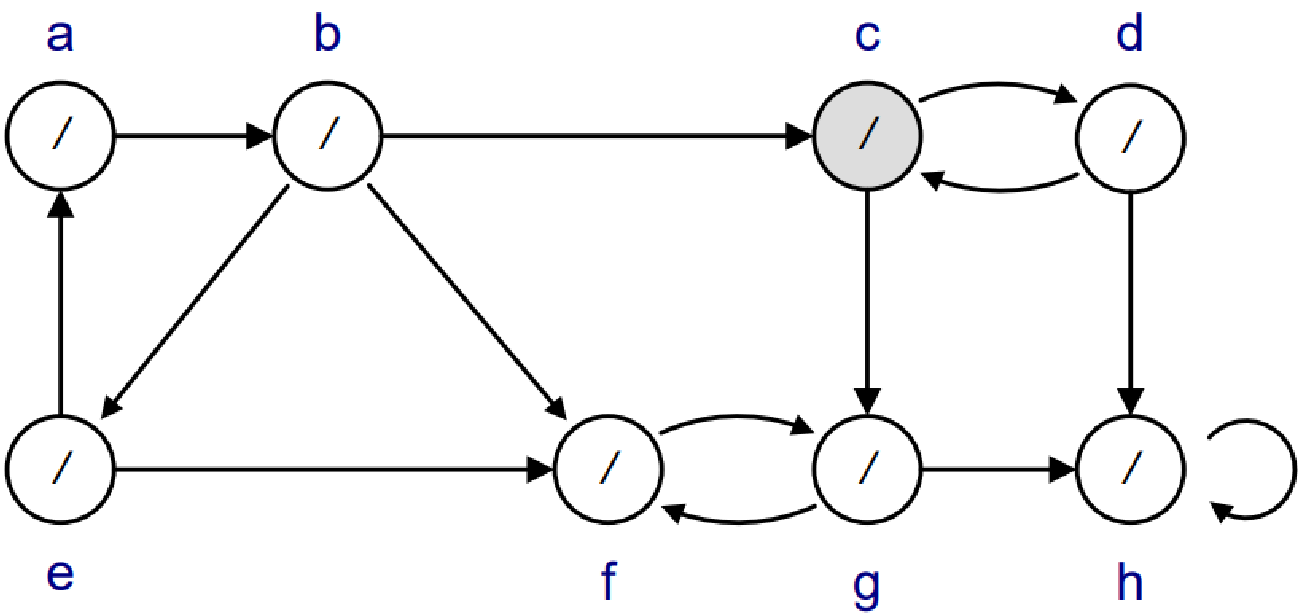
Lista: [c, a, b, d, e, f, g, h]

- Pilha de execução: DFS(G) - próximo u=**a** (Para)

$$tempo = 1$$

Algoritmo DFS em grafos - exemplo

- A cor de **g** é BRANCA?



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

$\boxed{\text{se } cor[v] = \text{BRANCO}}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

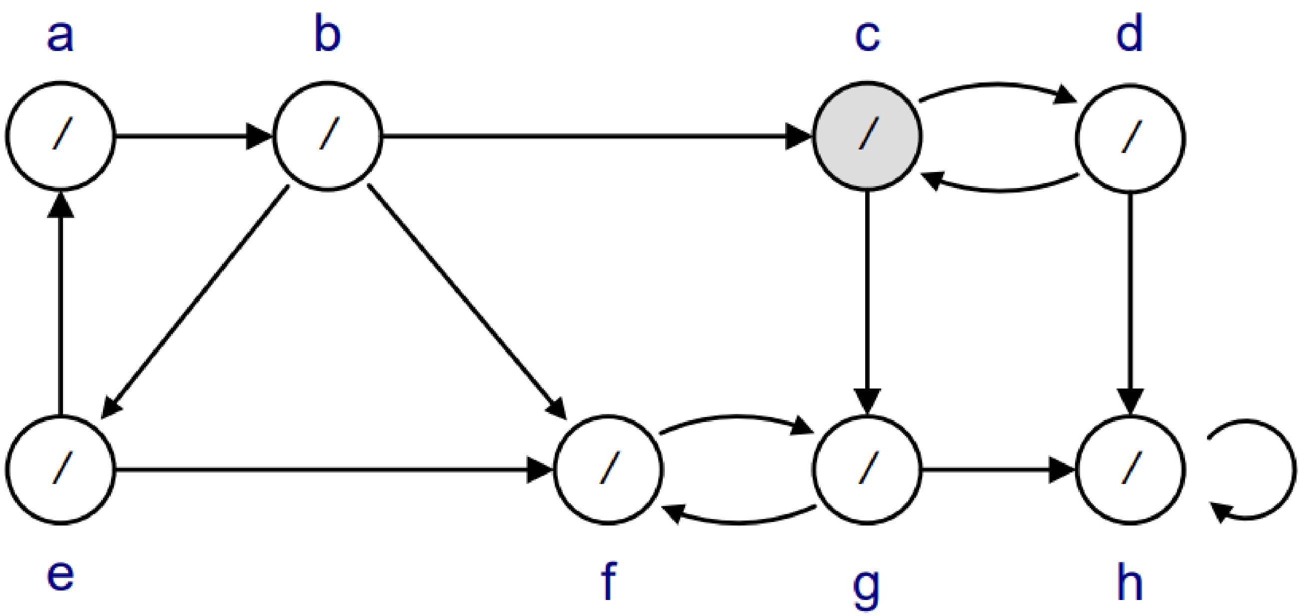
Lista: [c, a, b, d, e, f, g, h]

- Pilha de execução:
  DFS(G) - próximo u=**a** (Para)

$tempo = 1$

Algoritmo DFS em grafos - exemplo

- Chama função $DSF\_VISIT(u = \boldsymbol{g})$
- Vai empilhar a chamada corrente $DSF\_VISIT(\boldsymbol{c})$



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$\boxed{DSF\_VISIT(v)}$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

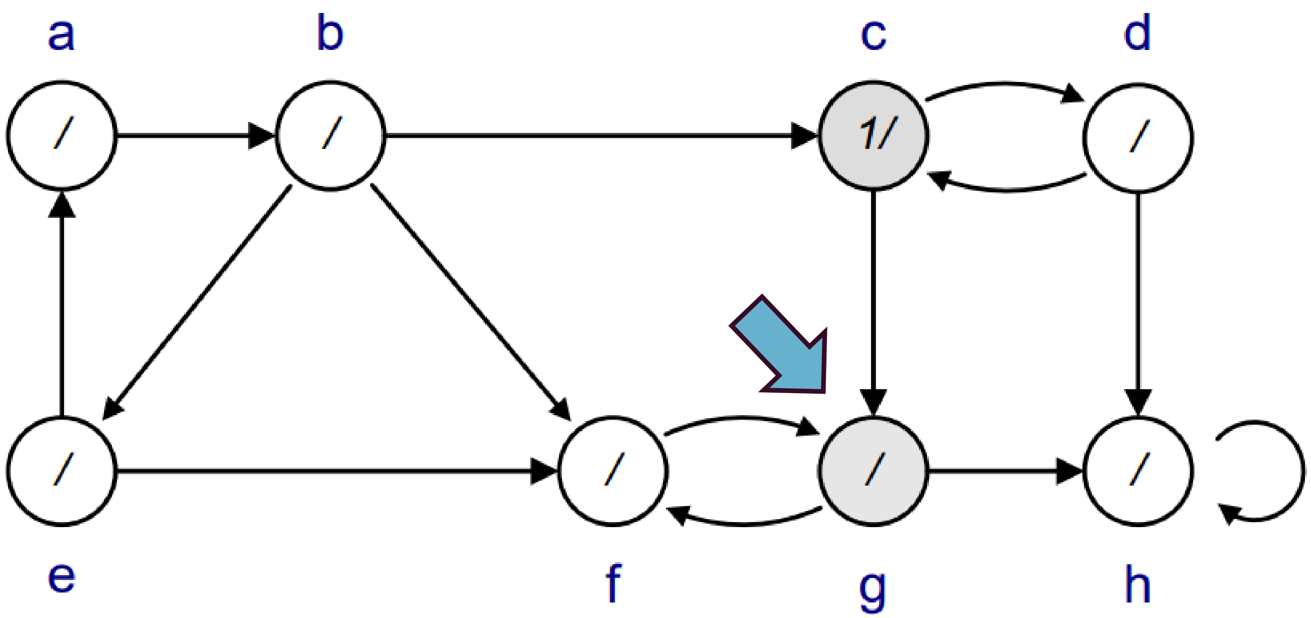
Lista: [c, a, b, d, e, f, g, h]

- Pilha de execução:
DFS(G) - próximo u=**a** (Para)

$$tempo = 1$$

Algoritmo DFS em grafos - exemplo

- Colore **g** de CINZA



$DSF\_VISIT(u)$

$\boxed{cor[u] = \text{CINZA}}$

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

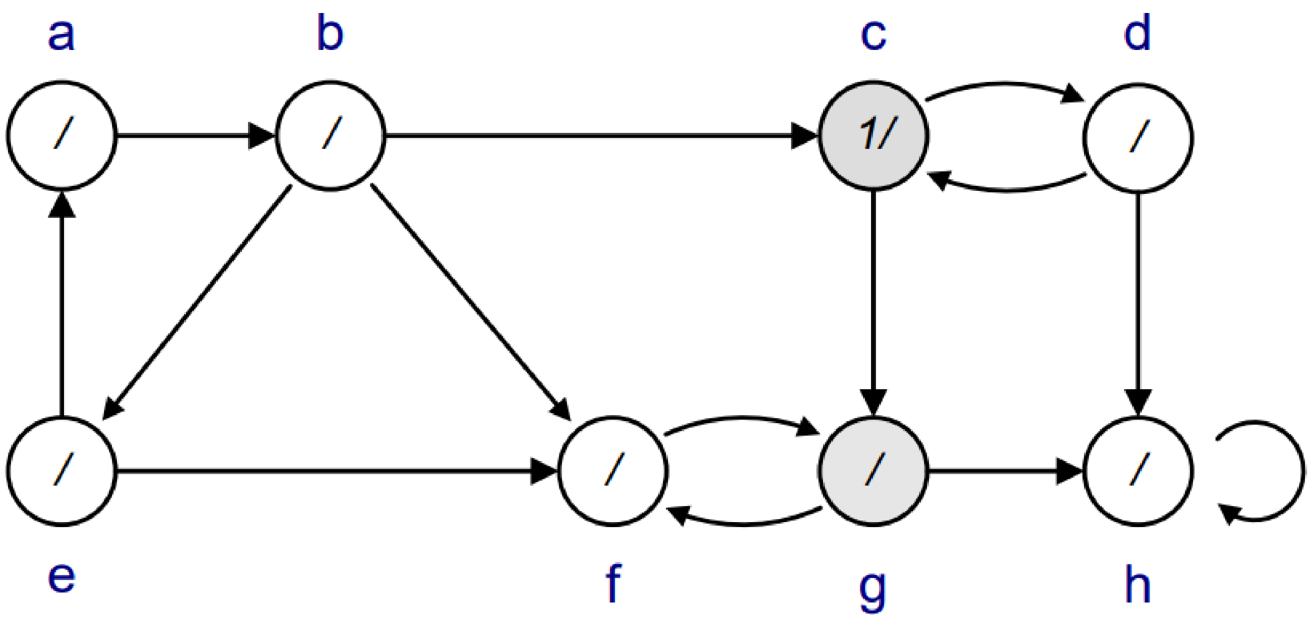
Lista: [c, a, b, d, e, f, g, h]

$tempo = 1$

- Pilha de execução:
DFS_VISIT(**c**)
DFS(G) - próximo u=**a** (Para)

Algoritmo DFS em grafos - exemplo

- Incrementa o tempo



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$\boxed{tempo = tempo + 1}$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

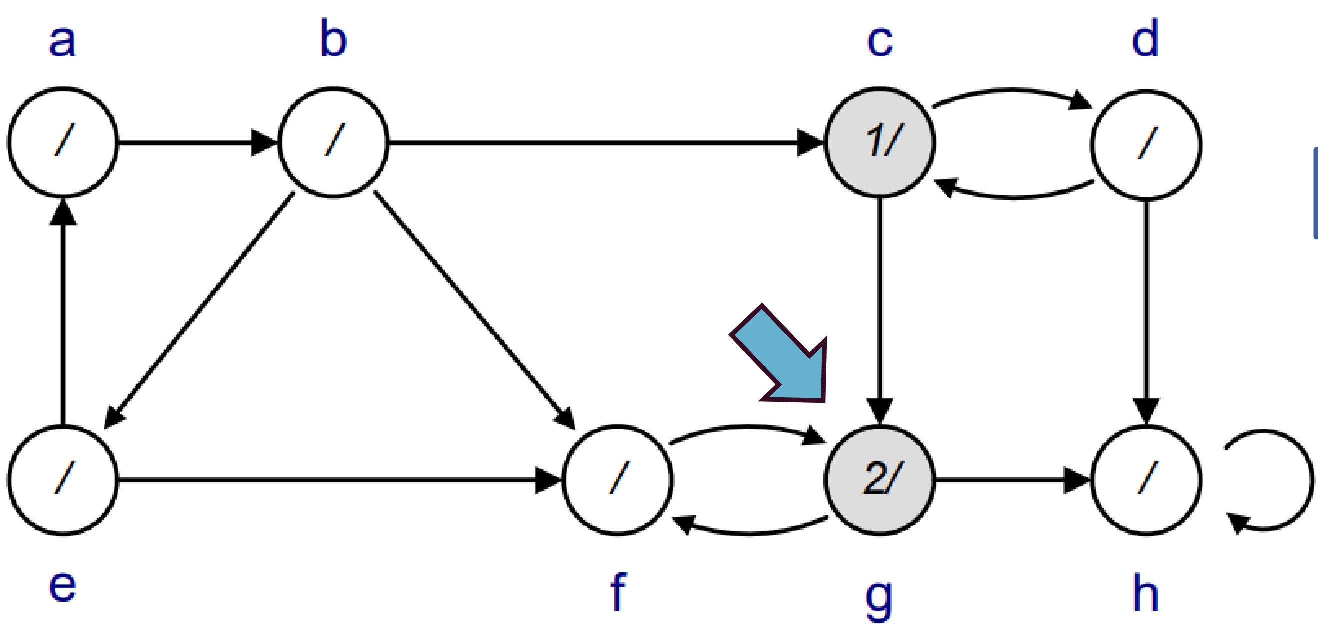
Lista: [c, a, b, d, e, f, g, h]

$tempo = 2$

- Pilha de execução:
  DFS_VISIT(**c**)
  DFS(G) - próximo u=**a** (Para)

Algoritmo DFS em grafos - exemplo

- Indica o tempo de descoberta do nó **g**



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$\boxed{d[u] = tempo}$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

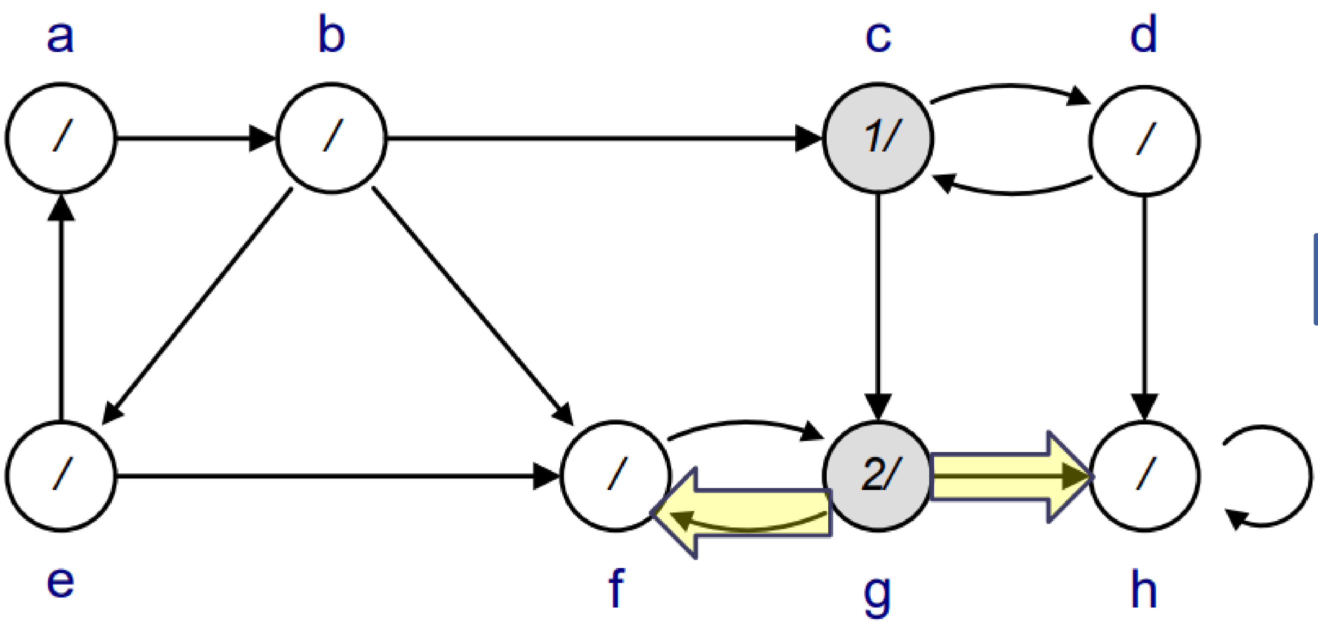
Lista: [c, a, b, d, e, f, g, h]

$tempo = 2$

- Pilha de execução:
  DFS_VISIT(**c**)
  DFS(G) - próximo u=**a** (Para)

Algoritmo DFS em grafos - exemplo

- Para cada nó adjacente à **g**, isto é, Adj(**g**) = {**f**, **h**}



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

$\boxed{\text{Para cada } v \in Adj(u)}$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

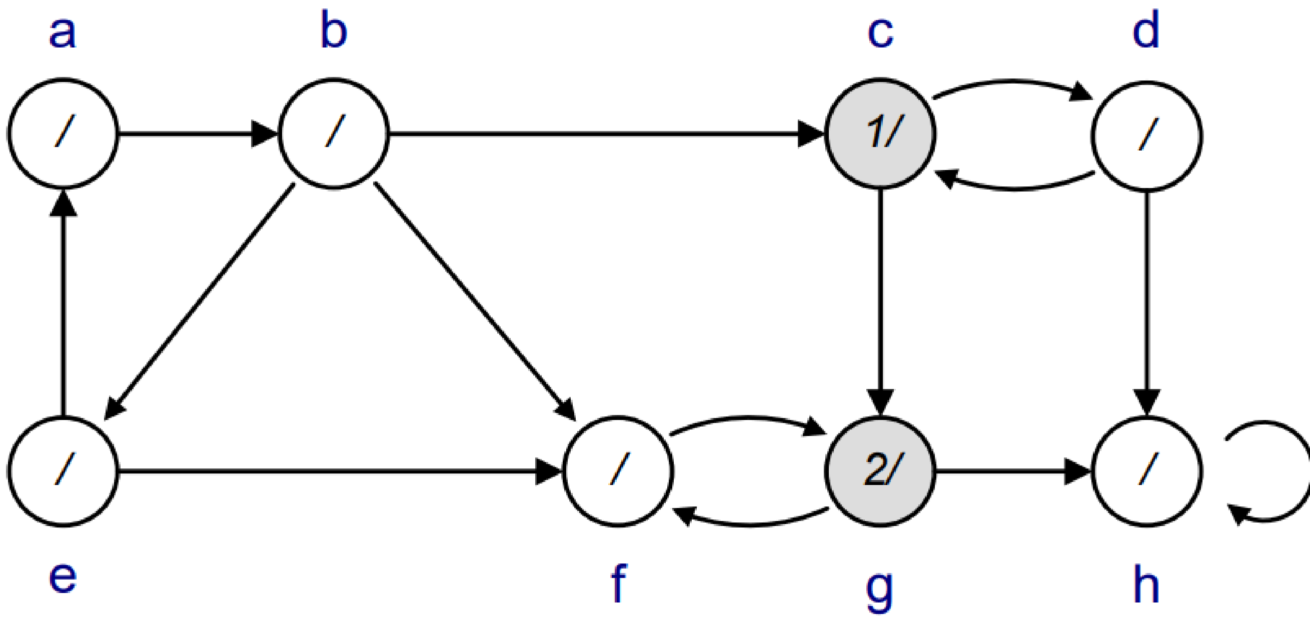
Lista: [c, a, b, d, e, f, g, h]

$tempo = 2$

- Pilha de execução:
  DFS_VISIT(**c**)
  DFS(G) - próximo u=**a** (Para)

Algoritmo DFS em grafos - exemplo

- A cor do nó **f** é BRANCA?
  - Então, invocar DFS_VISIT(**f**)
  - Empilhar DFS_VISIT(**f**) – próximo: v=**h**



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

$\boxed{\text{se } cor[v] = \text{BRANCO}}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

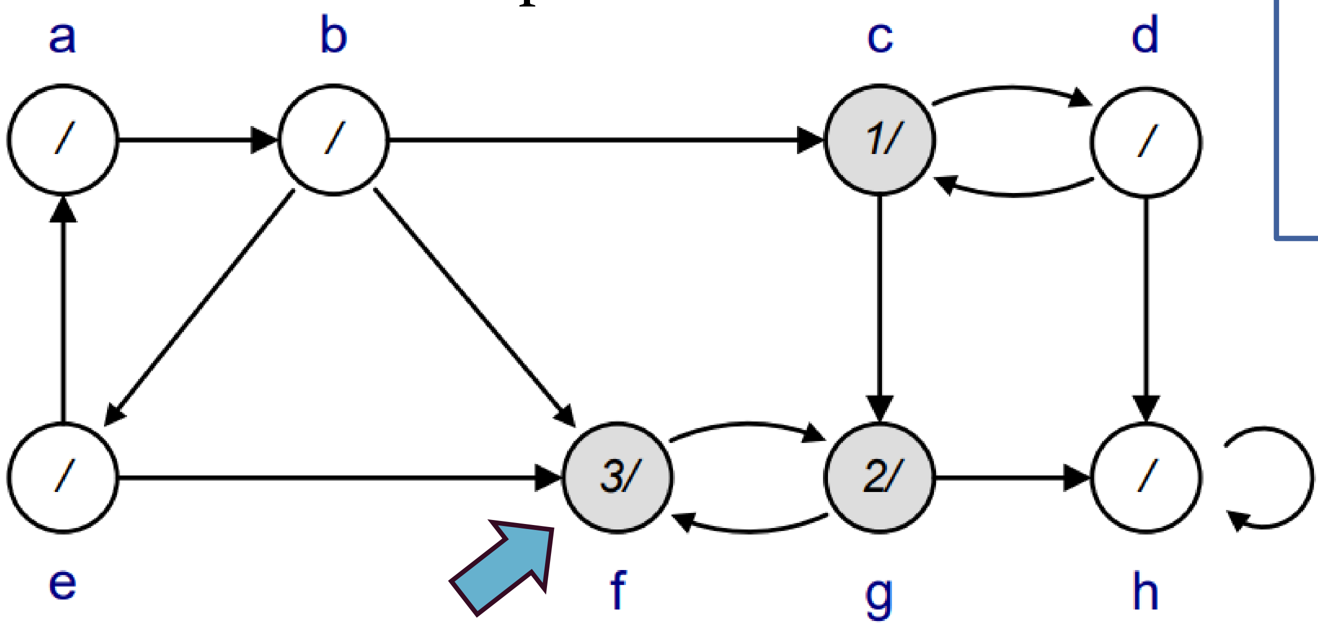
Lista: [c, a, b, d, e, f, g, h]

$tempo = 2$

- Pilha de execução:
  DFS_VISIT(**c**)
  DFS(G) - próximo u=**a** (Para)

Algoritmo DFS em grafos - exemplo

- Marca o nó **f** como CINZA
- Incrementa o tempo
- Indica o tempo de descoberta do nó **f**



$DSF\_VISIT(u)$

$$\boxed{\begin{aligned} & cor[u] = \text{CINZA} \\ & tempo = tempo + 1 \\ & d[u] = tempo \end{aligned}}$$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

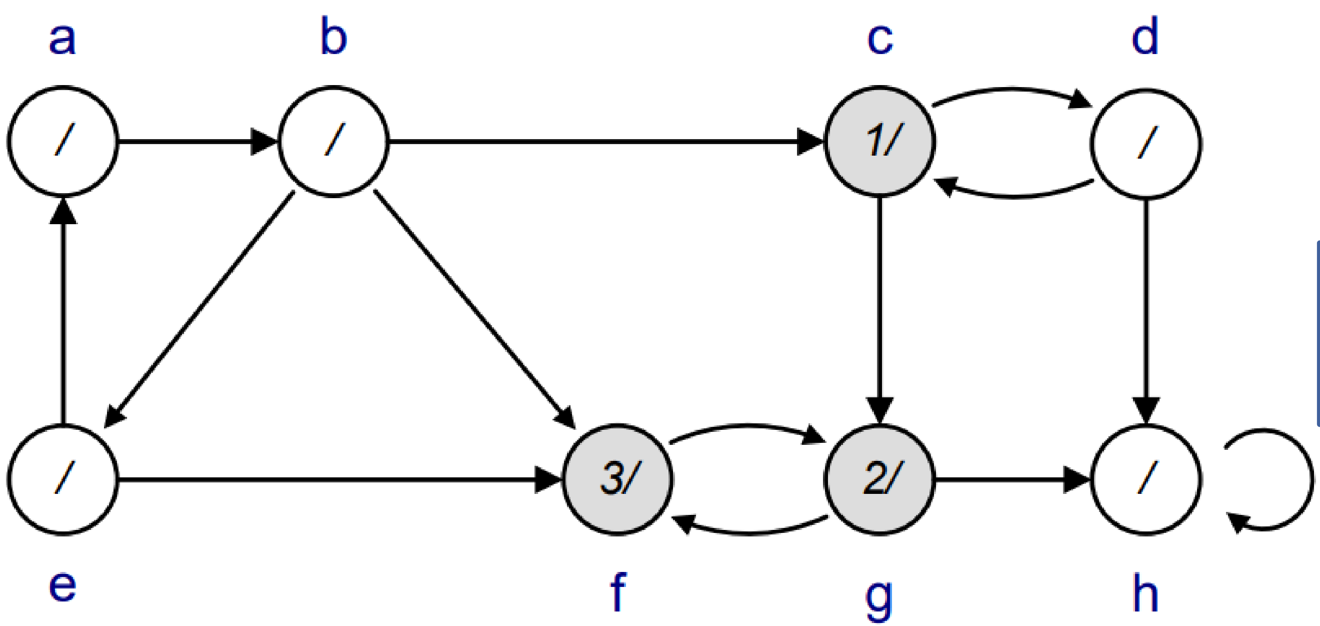
Lista: [c, a, b, d, e, f, g, h]

$tempo = 3$

- Pilha de execução:
  DFS_VISIT(**g**)
  DFS_VISIT(**c**)
  DFS(G) - próximo u=**a** (Para)

Algoritmo DFS em grafos - exemplo

- Nó **f** possui apenas um nó adjacente: {**g**}
- Nó **g** **não** é BRANCO



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

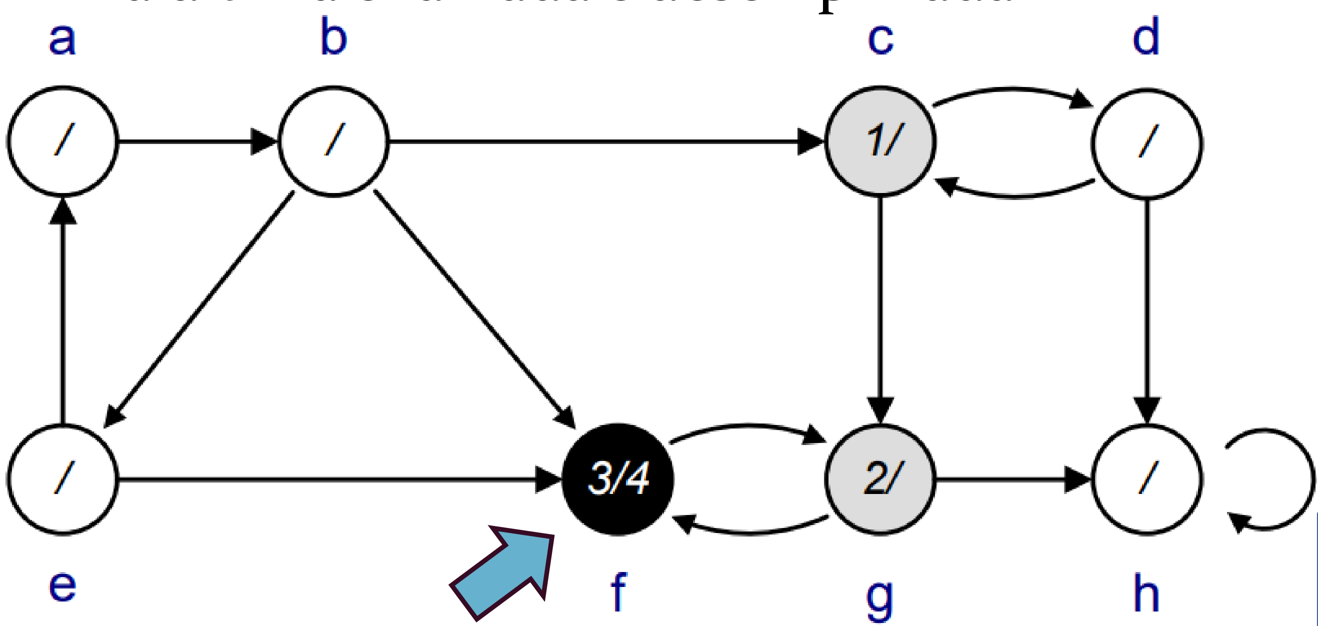
Lista: [c, a, b, d, e, f, g, h]

$tempo = 3$

- Pilha de execução:
  DFS_VISIT(**g**)
  DFS_VISIT(**c**)
  DFS(G) - próximo u=**a** (Para)

Algoritmo DFS em grafos - exemplo

- O laço termina, **f** recebe cor PRETA; Incrementa o tempo; É indicado tempo de finalização de **f**; A função termina.. Agora a última chamada é desempilhada



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

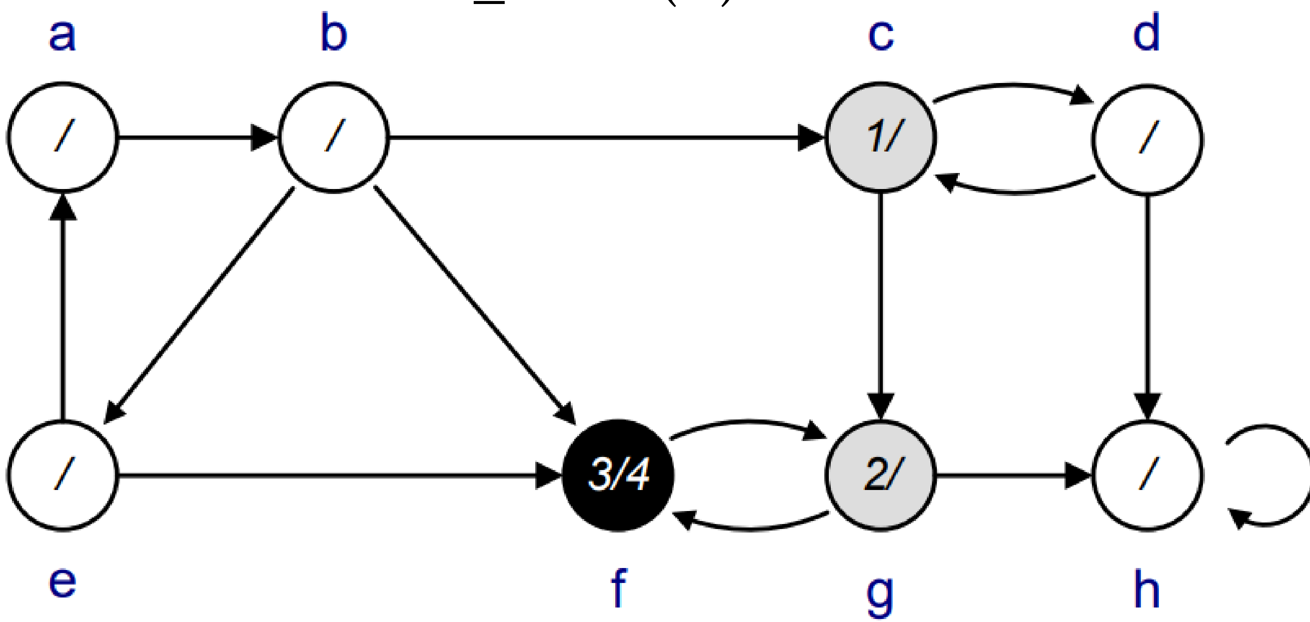
Lista: [c, a, b, d, e, f, g, h]

$tempo = 4$

- Pilha de execução:
  DFS_VISIT(**g**)
  DFS_VISIT(**c**)
  DFS(G) - próximo u=**a** (Para)

Algoritmo DFS em grafos - exemplo

- Desempilhou: DFS_VISIT(**g**)
- Próximo nó adjacente de **g** é v=**h** (e ele é BRANCO!)
- Logo, empilha novamente DFS_VISIT(**g**)
- E chama DFS_VISIT(**h**)



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

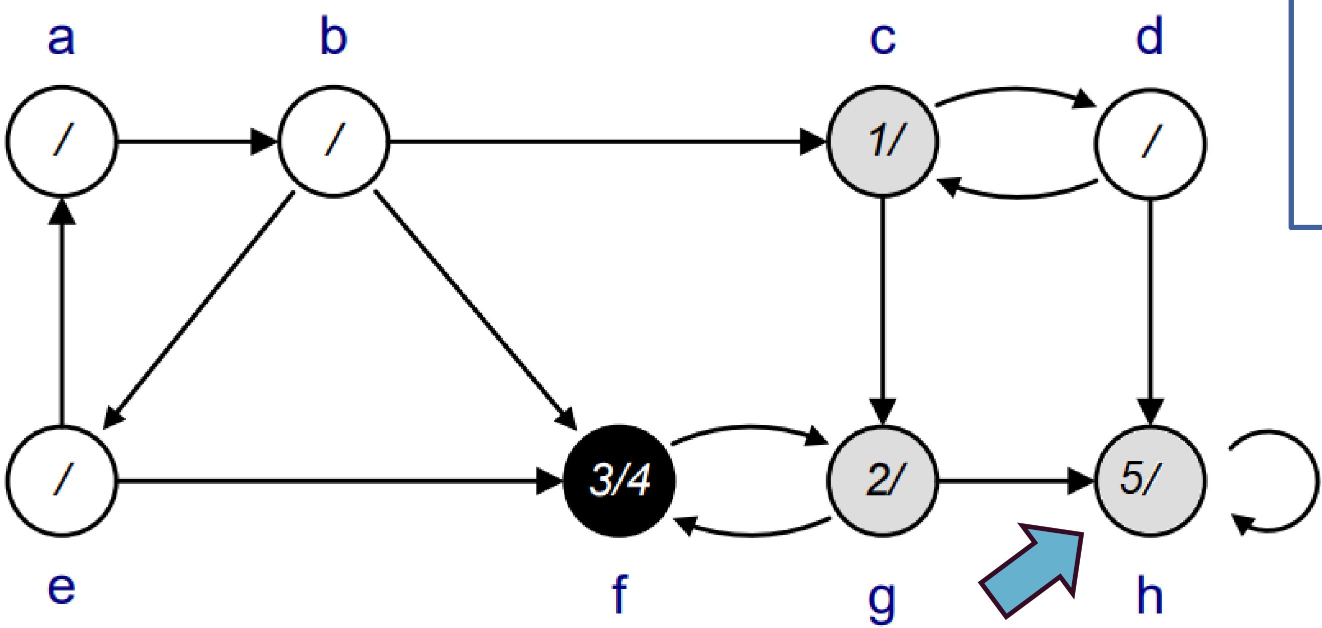
Lista: [c, a, b, d, e, f, g, h]

$tempo = 4$

- Pilha de execução:
  DFS_VISIT(**c**)
  DFS(G) - próximo u=**a** (Para)

Algoritmo DFS em grafos - exemplo

- Colore o nó **h** de CINZA;
- Incrementa o tempo;
- Indica o tempo que o nó **h** foi localizado



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

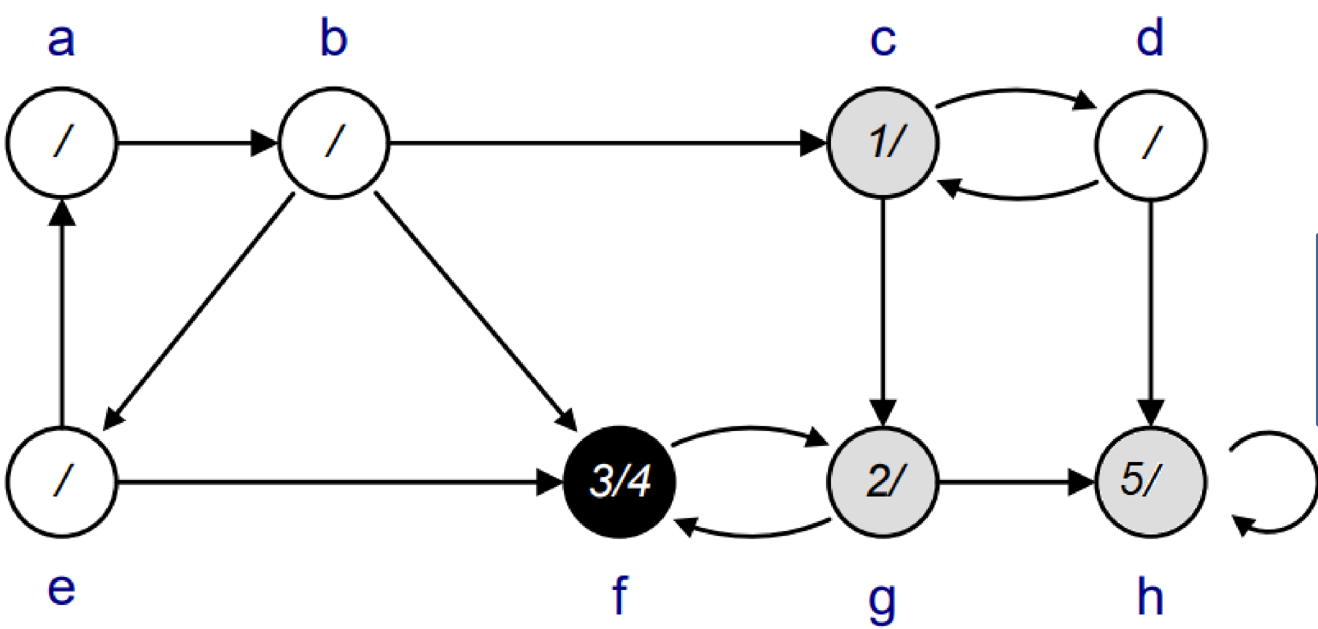
Lista: [c, a, b, d, e, f, g, h]

$tempo = 5$

- Pilha de execução:
  DFS_VISIT(**g**)
  DFS_VISIT(**c**)
  DFS(G) - próximo u=**a** (Para)

Algoritmo DFS em grafos - exemplo

- Para cada nó adjacente de **h**: {**h**}
- Mas o nó **h** é CINZA
- Então a busca sobre **h** será finalizada



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

Lista: [c, a, b, d, e, f, g, h]

$tempo = 5$

- Pilha de execução:
  DFS_VISIT(**g**)
  DFS_VISIT(**c**)
  DFS(G) - próximo u=**a** (Para)

Algoritmo DFS em grafos - exemplo

- Marca **h** de PRETO
- Incrementa o tempo
- Indica o tempo de finalização de h
- Desempilha

$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

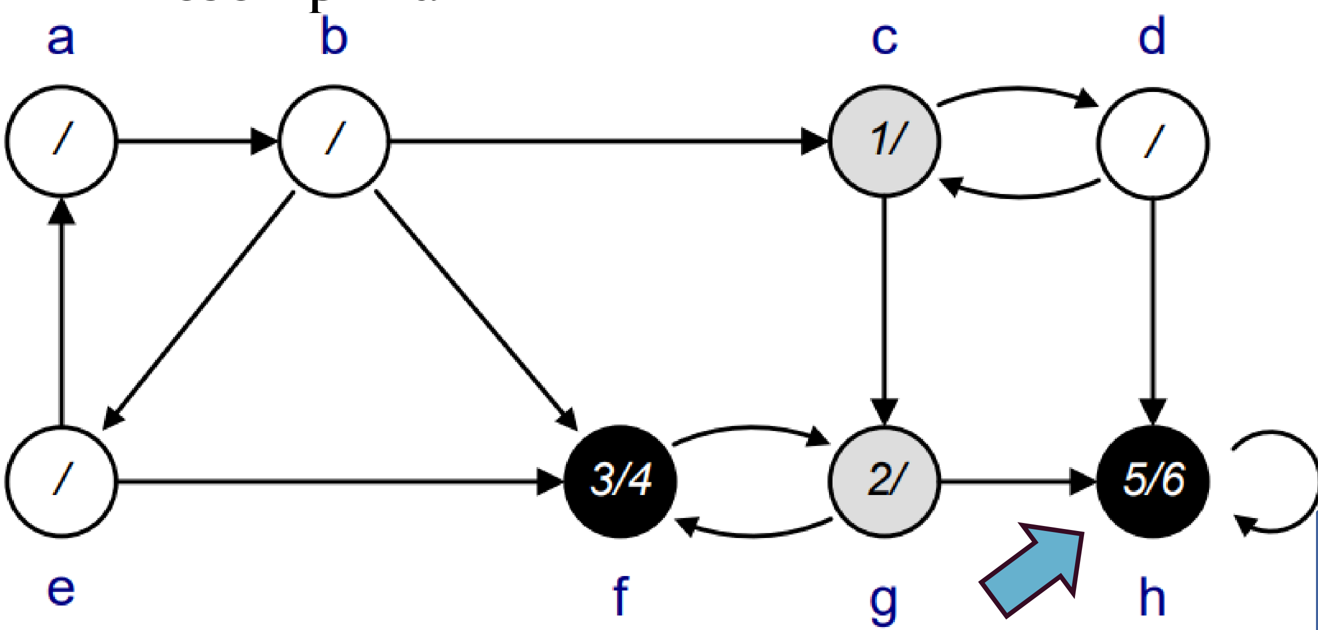
Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$



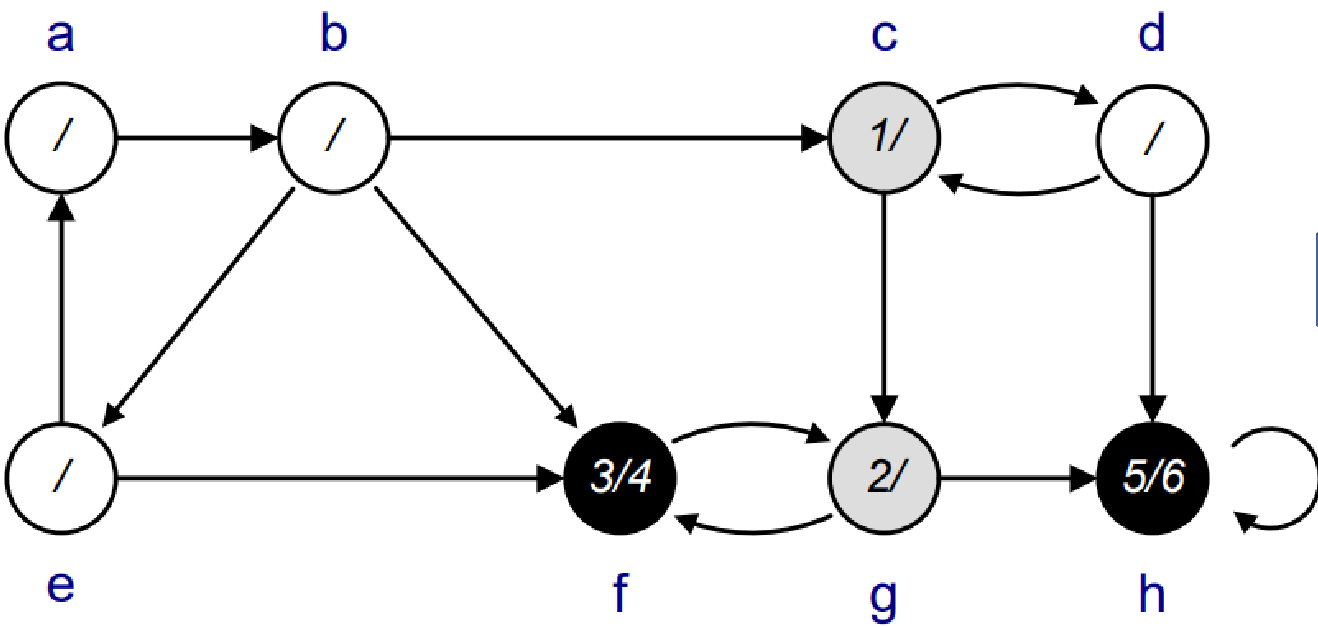
Lista: [c, a, b, d, e, f, g, h]

$tempo = 6$

- Pilha de execução:
  DFS_VISIT(**g**)
  DFS_VISIT(**c**)
  DFS(G) - próximo u=**a** (Para)

Algoritmo DFS em grafos - exemplo

- Desempilhou: DFS_VISIT(**g**)
- O nó **g** não possui mais nós adjacentes não visitados
- Assim, a busca em **g** termina



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

$\boxed{\text{Para cada } v \in Adj(u)}$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

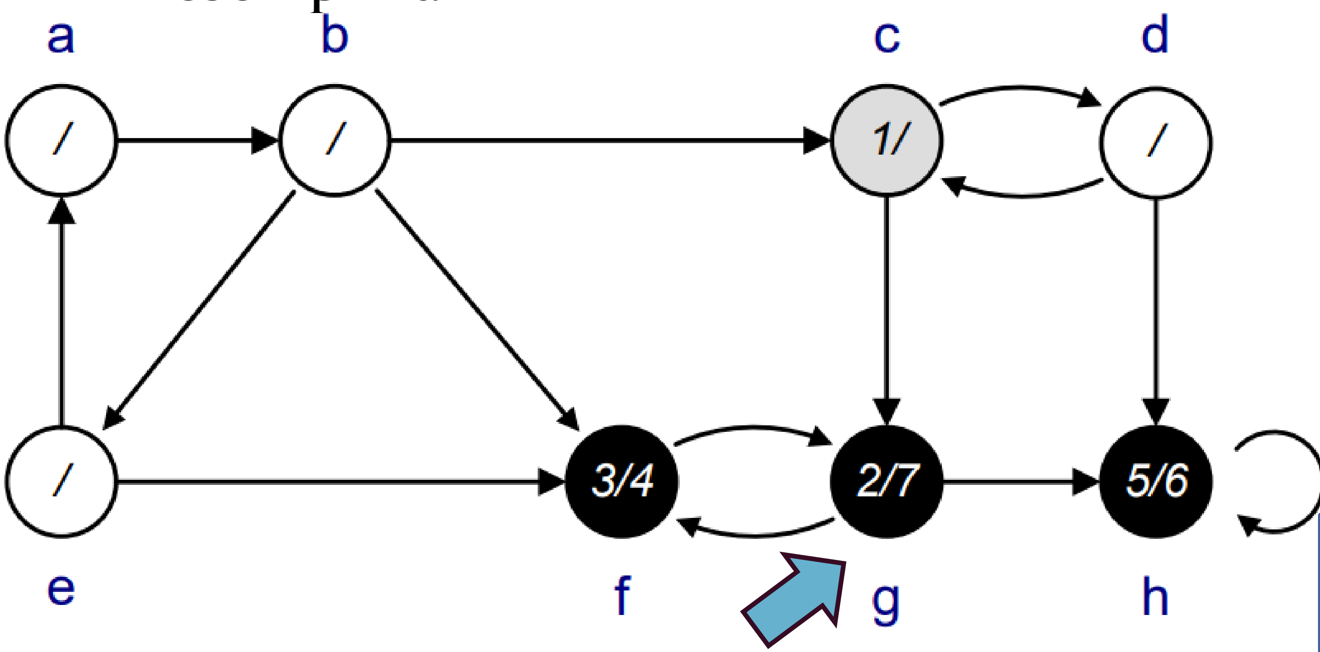
Lista: [c, a, b, d, e, f, g, h]

$tempo = 6$

- Pilha de execução:
  DFS_VISIT(**c**)
  DFS(G) - próximo u=**a** (Para)

Algoritmo DFS em grafos - exemplo

- Marca g de PRETO
- Incrementa o tempo
- Indica o tempo de finalização de g
- Desempilha



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

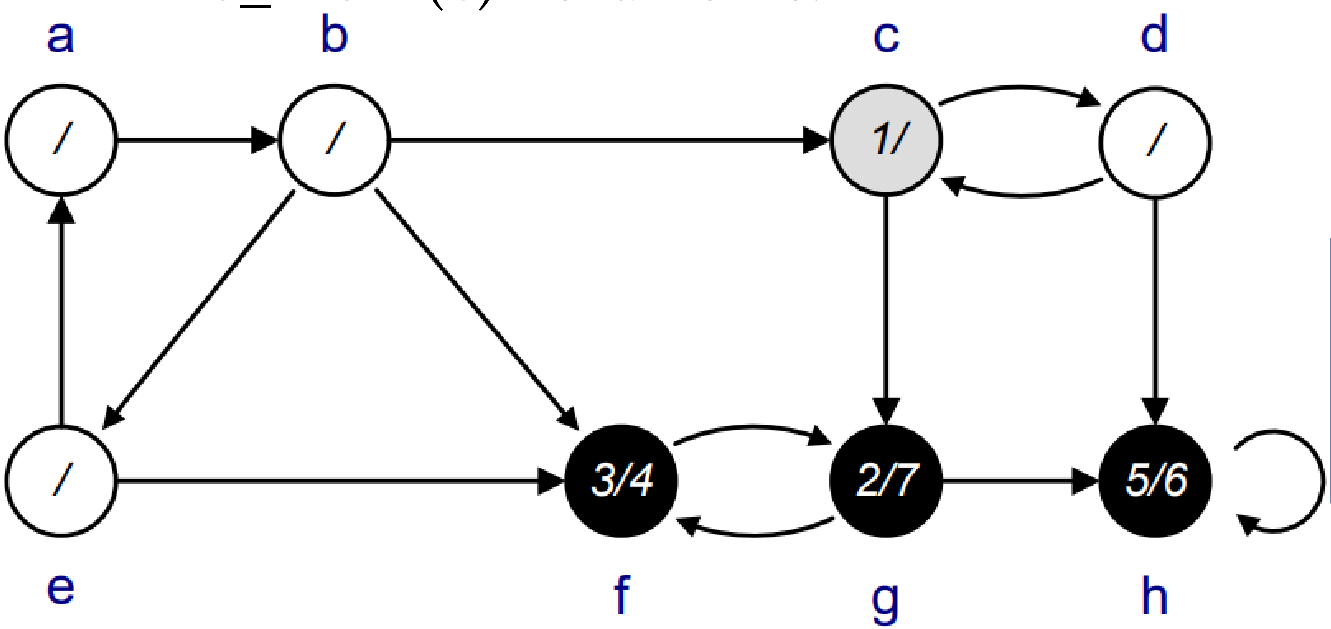
Lista: [c, a, b, d, e, f, g, h]

$tempo = 7$

- Pilha de execução:
  DFS_VISIT(c)
  DFS(G) - próximo u=a (Para)

Algoritmo DFS em grafos - exemplo

- Desempilhou DFS_VISIT(**c**)
- Próximo nó adjacente ao nó **c:** é o nó **d**, que é BRANCO
- Assim, invoca DFS_VISIT(**d**) e empilha DFS_VISIT(**c**) novamente.



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

$\boxed{\begin{array}{l} \text{Para cada } v \in Adj(u) \\ \quad \text{se } cor[v] = \text{BRANCO} \\ \qquad DSF\_VISIT(v) \end{array}}$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

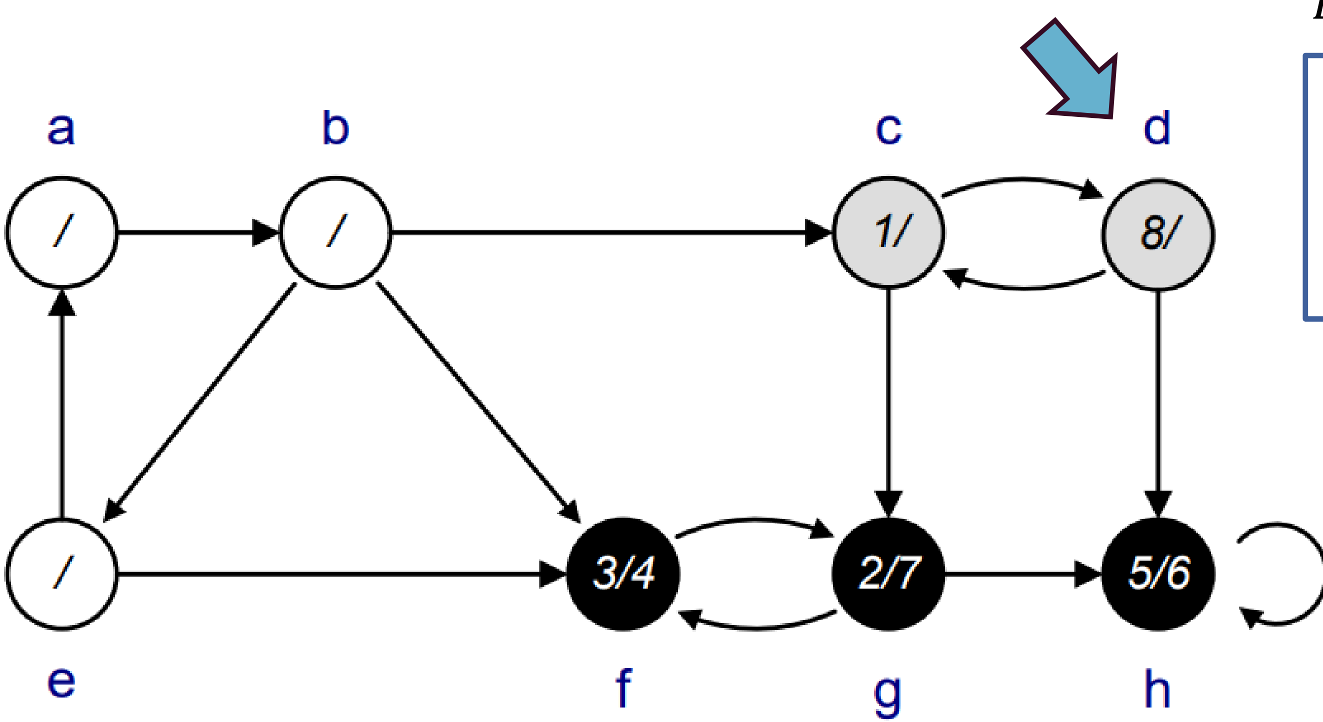
Lista: [c, a, b, d, e, f, g, h]

- Pilha de execução:
  DFS(G) - próximo u=**a** (Para)

$$tempo = 7$$

Algoritmo DFS em grafos - exemplo

- Colore o nó **d** de CINZA, incrementa o tempo, e indica o tempo que o nó **d** foi localizado ...



$DSF\_VISIT(u)$

$\boxed{\begin{array}{l} cor[u] = \text{CINZA} \\ tempo = tempo + 1 \\ d[u] = tempo \end{array}}$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

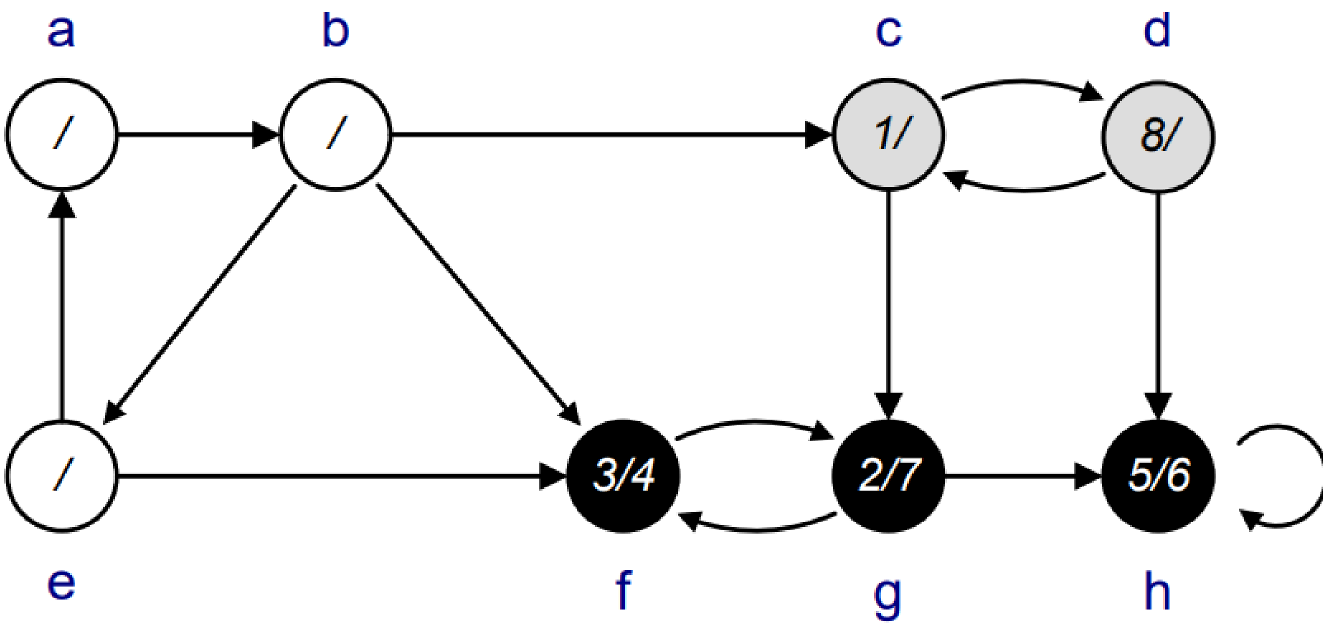
Lista: [c, a, b, d, e, f, g, h]

$tempo = 8$

- Pilha de execução:
  DFS_VISIT(**c**)
  DFS(G) - próximo u=**a** (Para)

Algoritmo DFS em grafos - exemplo

- O nó **d** possui apenas um nó adjacente, que não é BRANCO, assim a busca sobre **d** será finaliza



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

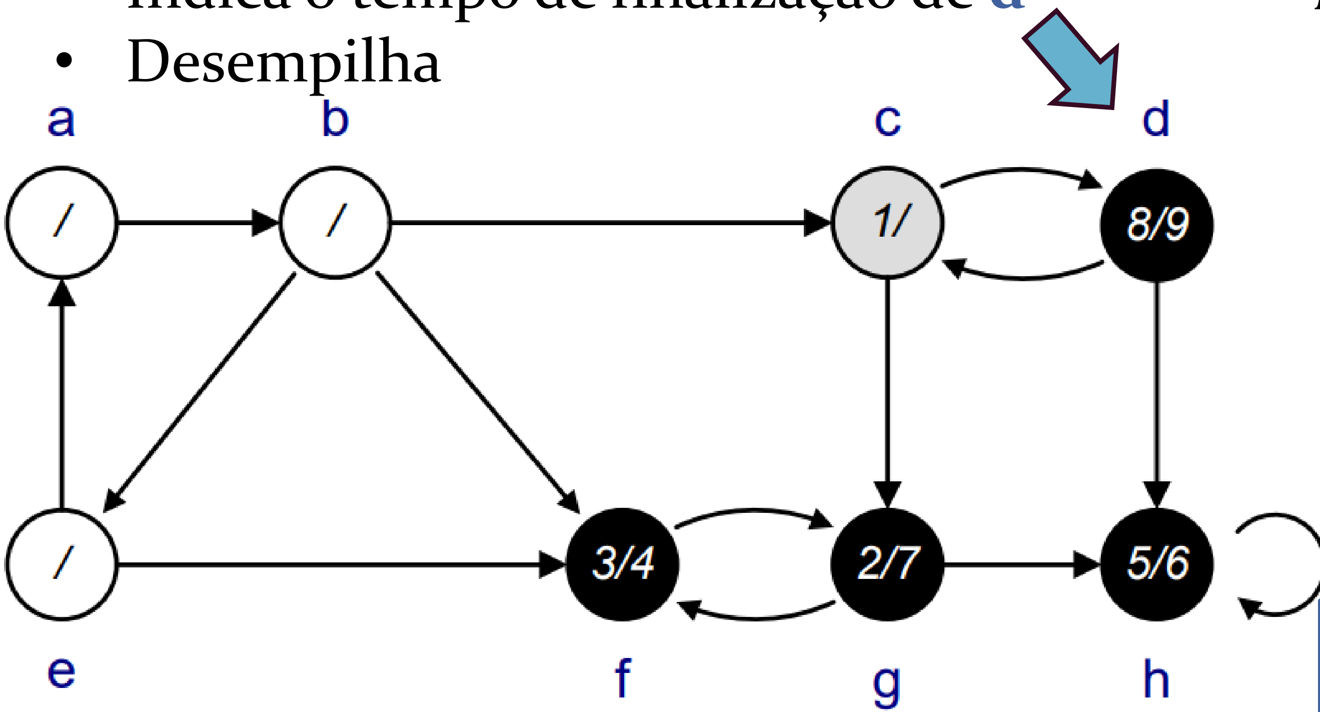
Lista: [c, a, b, d, e, f, g, h]

$tempo = 8$

- Pilha de execução:
  DFS_VISIT(**c**)
  DFS(G) - próximo u=**a** (Para)

Algoritmo DFS em grafos - exemplo

- Marca **d** de PRETO
- Incrementa o tempo
- Indica o tempo de finalização de **d**
- Desempilha



$DSF\_VISIT(u)$

$cor[u] =$ CINZA

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v] =$ BRANCO

$DSF\_VISIT(v)$

$cor[u] =$ PRETO

$f[u] = tempo = tempo + 1$

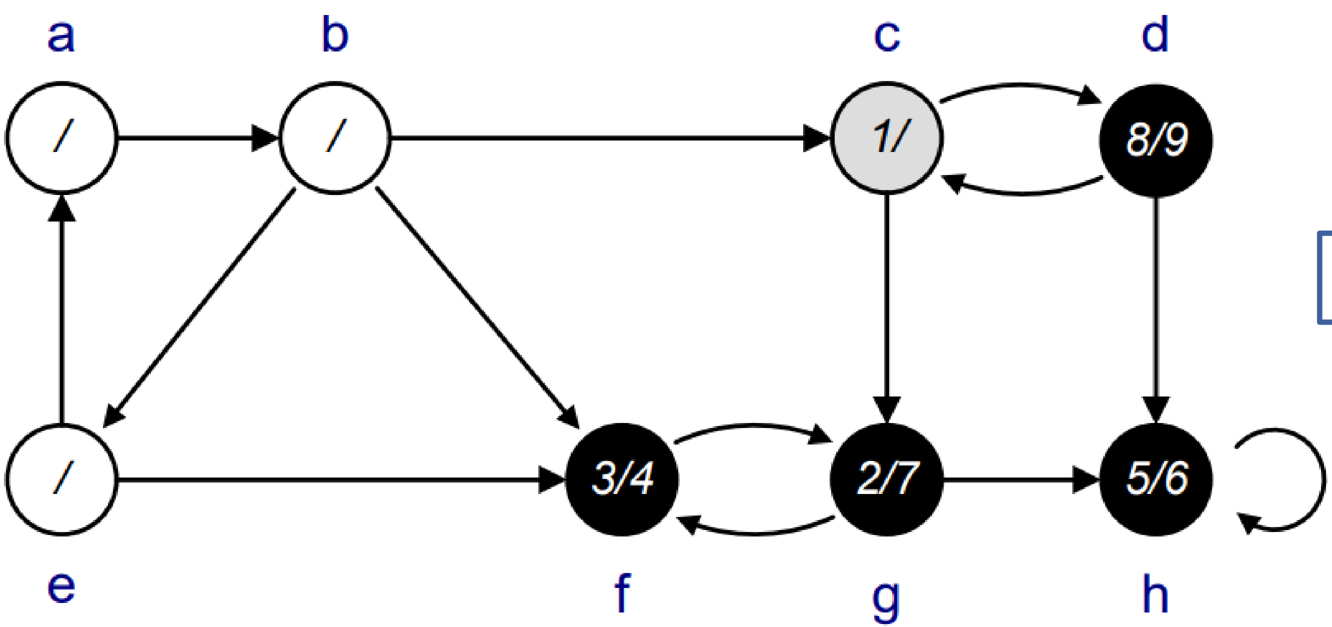
Lista: [c, a, b, d, e, f, g, h]

$tempo = 9$

- Pilha de execução:
  DFS_VISIT(**c**)
  DFS(G) - próximo u=**a** (Para)

Algoritmo DFS em grafos - exemplo

- Desempilhou DFS_VISIT(**c**)
- Não possui mais nós adjacentes, então a busca sobre **c** será finalizada



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

$\boxed{\text{Para cada } v \in Adj(u)}$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

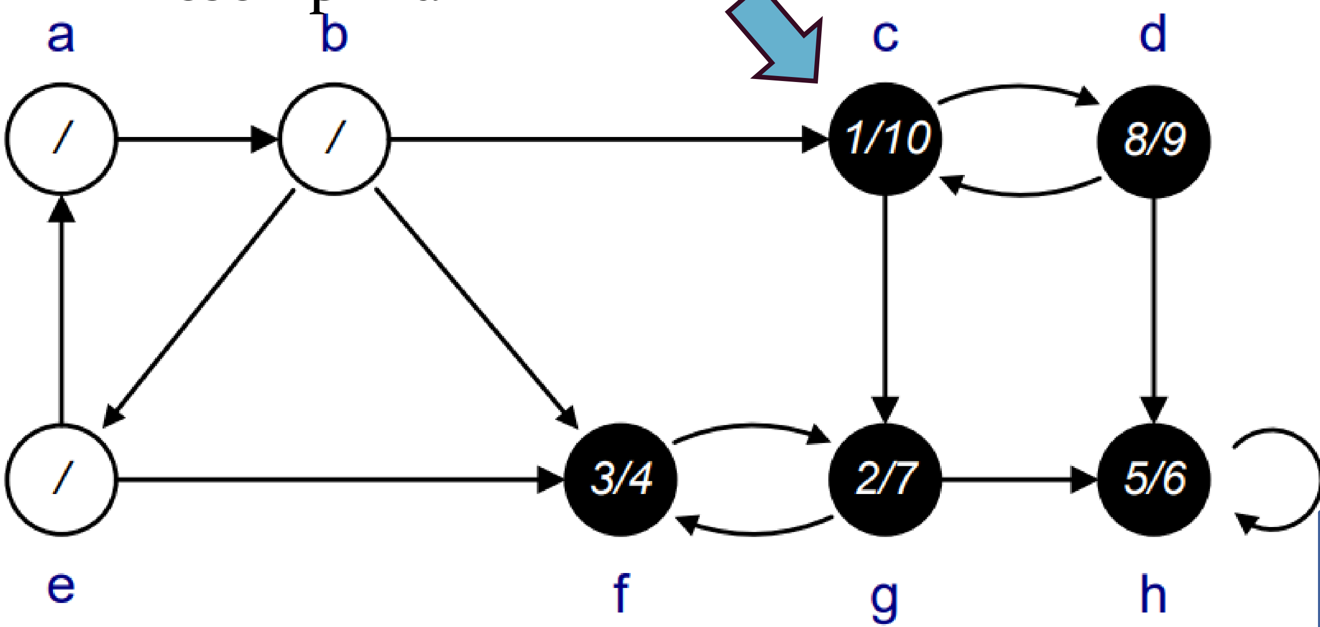
Lista: [c, a, b, d, e, f, g, h]

- Pilha de execução:
  DFS(G) - próximo u=**a** (Para)

$$tempo = 9$$

Algoritmo DFS em grafos - exemplo

- Marca **c** de PRETO
- Incrementa o tempo
- Indica o tempo de finalização de **c**
- Desempilha



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

Lista: [c, a, b, d, e, f, g, h]

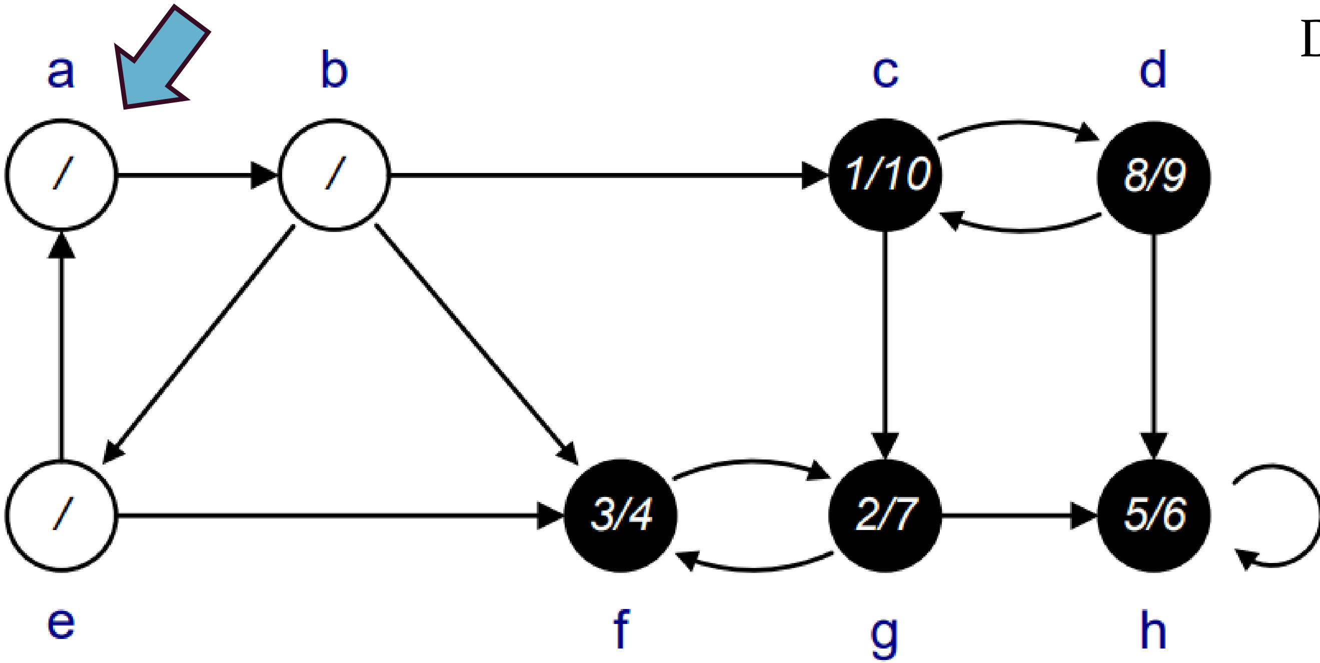
- Pilha de execução:
DFS(G) - próximo u=**a** (Para)

$tempo = 10$


1 HORA EM PYTHON
terminalroot.com
SÃO 7 HORAS EM JAVA

Algoritmo DFS em grafos - exemplo

- Desempilhou DFS_(G) – próximo: u=**a**
- O nó **a** é BRANCO, então chama DFS_VISIT(**a**)
- Empilha DFS(G) – próximo: u=**b**



DFS(G)

Para cada nó $u \in V$:

$cor[u] = \text{BRANCO}$

$tempo = 0$

Para cada $u \in V$

se $cor[u] = \text{BRANCO}$

$DSF\_VISIT(u)$

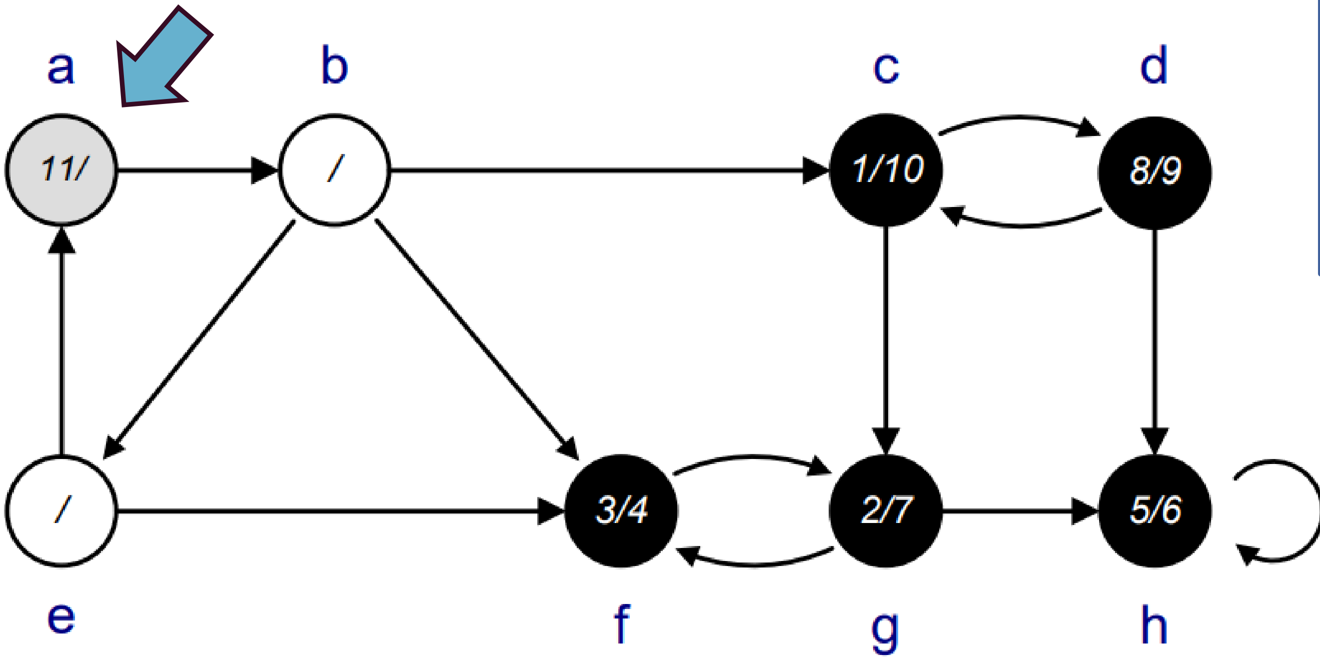
Lista: [c, **a**, b, d, e, f, g, h]

- Pilha de execução: **VAZIA!!!**

$$tempo = 10$$

Algoritmo DFS em grafos - exemplo

- Marca **a** de CINZA
- Incrementa o tempo
- Atribui o tempo de descoberta de **a**



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

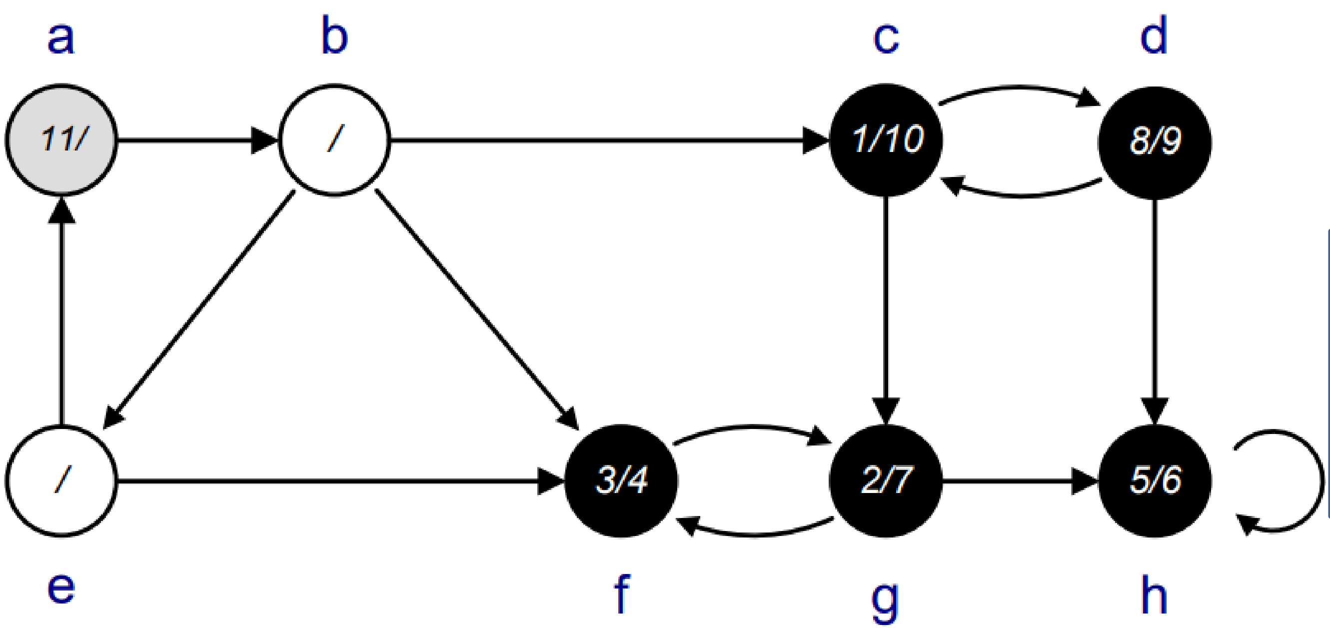
Lista: [c, a, b, d, e, f, g, h]

- Pilha de execução: DFS(G) - próximo: u=**b**

$tempo = 11$

Algoritmo DFS em grafos - exemplo

- Para todos os nós adjacentes do nó **a** = {**b**}
- Se a cor de **b** for BRANCA, então DFS_VISIT(b)
- Empilha DFS_VISIT(**a**)



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

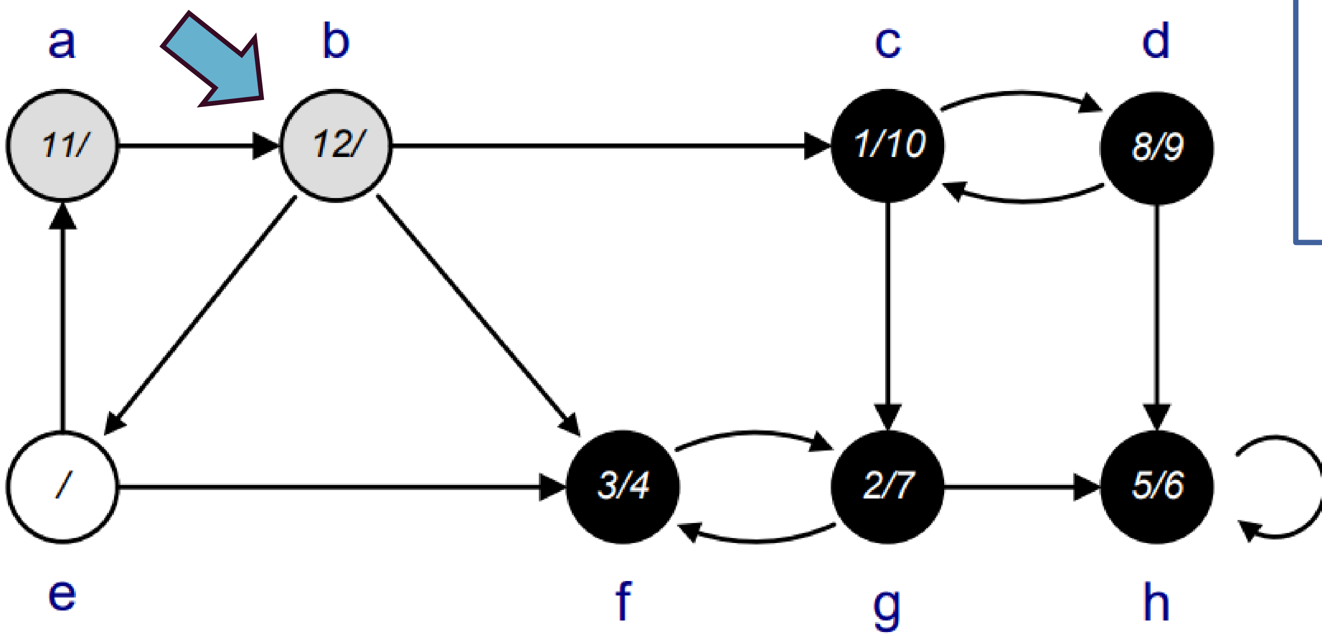
Lista: [c, a, b, d, e, f, g, h]

- Pilha de execução: DFS(G) - próximo: u=**b**

$tempo = 11$

Algoritmo DFS em grafos - exemplo

- Marca **b** de CINZA
- Incrementa o tempo
- Atribui o tempo de descoberta de **b**



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

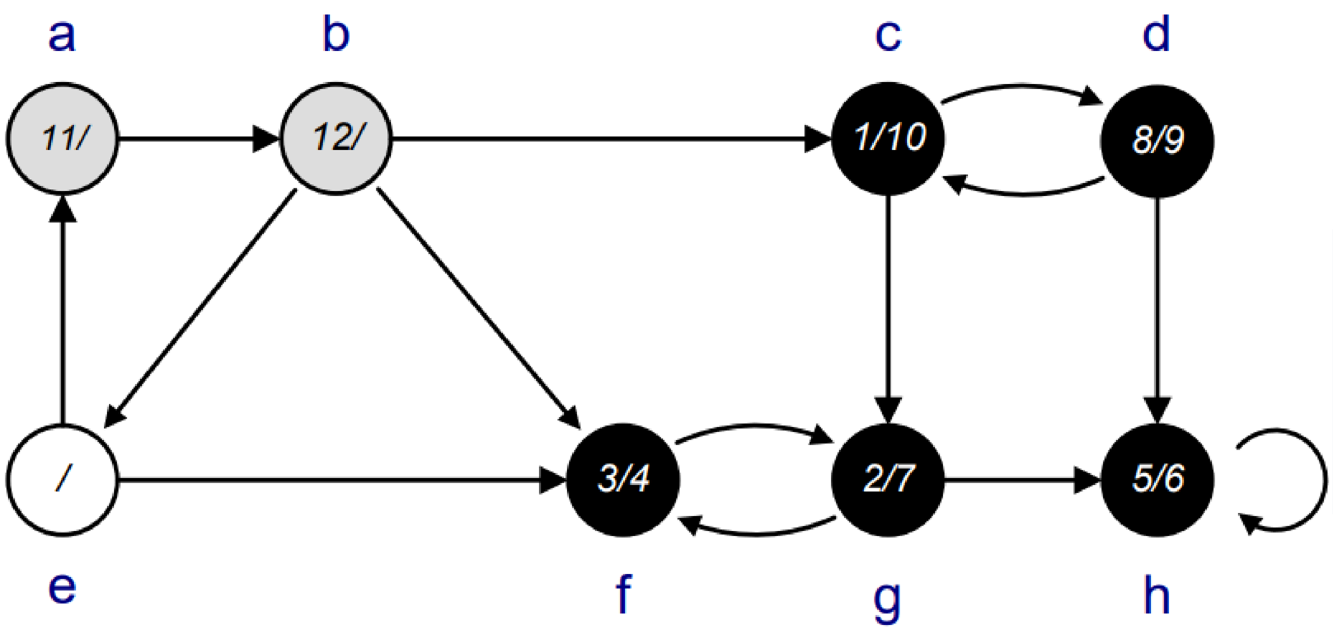
Lista: [c, a, b, d, e, f, g, h]

$tempo = 12$

- Pilha de execução:
  DFS_VISIT(**a**)
  DFS(G) - próximo: u=**b**

Algoritmo DFS em grafos - exemplo

- Para todos os nós adjacentes do nó **b** = {**c**, **e**, **f**}
- A cor de **c** e de **f** é PRETA, então pula.
- Como cor de **e** é BRANCA ➔ DFS_VISIT(**e**)



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

$\boxed{\begin{array}{l} \text{Para cada } v \in Adj(u) \\ \quad \text{se } cor[v] = \text{BRANCO} \\ \qquad DSF\_VISIT(v) \end{array}}$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

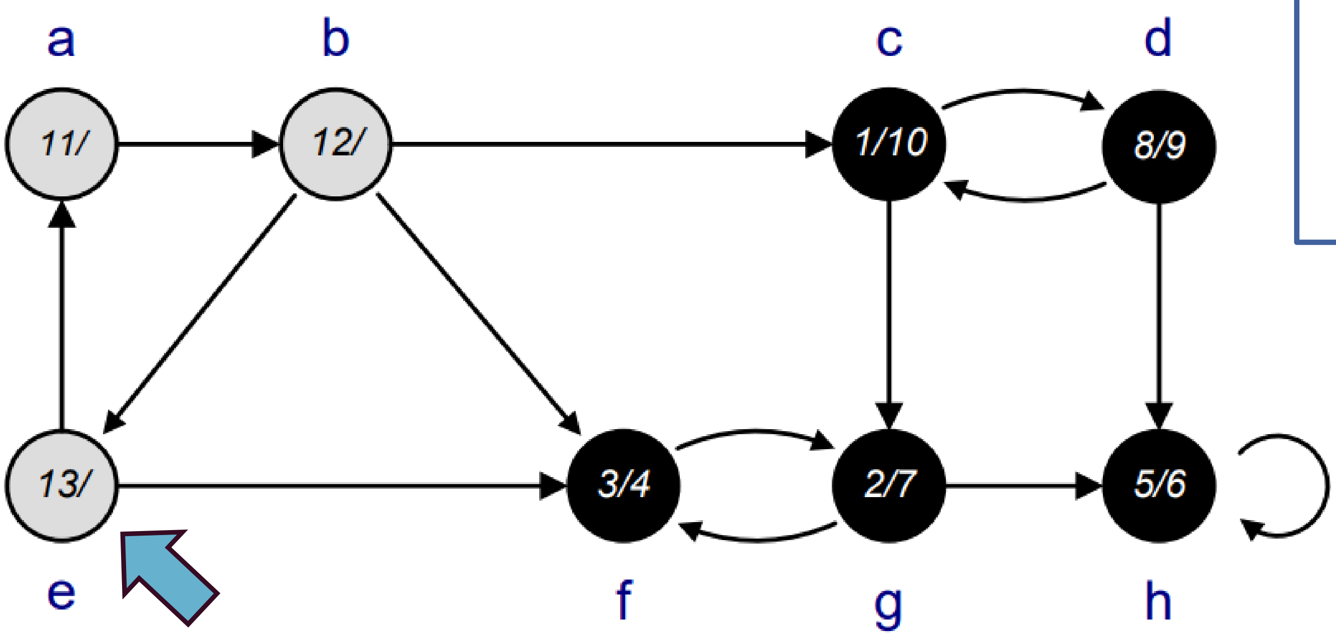
Lista: [c, a, b, d, e, f, g, h]

$tempo = 12$

- Pilha de execução:
  DFS_VISIT(**a**)
  DFS(G) - próximo: u=**b**

Algoritmo DFS em grafos - exemplo

- Marca **e** de CINZA
- Incrementa o tempo
- Atribui o tempo de descoberta de **e**



$DSF\_VISIT(u)$

$$\boxed{\begin{array}{l} cor[u] = \text{CINZA} \\ tempo = tempo + 1 \\ d[u] = tempo \end{array}}$$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

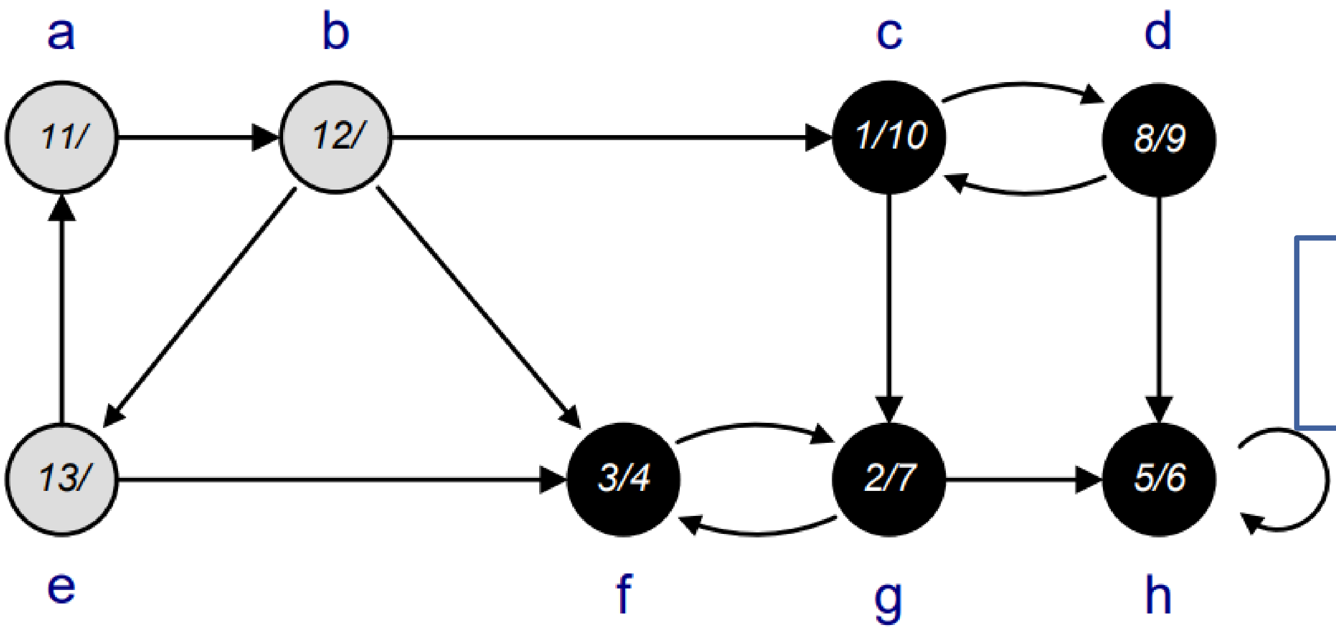
Lista: [c, a, b, d, e, f, g, h]

$tempo = 13$

- Pilha de execução:
  DSF_VISIT(**b**)
  DFS_VISIT(**a**)
  DFS(G) - próximo: u=**b**

Algoritmo DFS em grafos - exemplo

- Avalia os nós adjacente ao nó **e** = {**a, f**}
- Os nós **a** e **f** não são BRANCO(S), então finaliza a busca sobre o nó **e**.



$DSF\_VISIT(u)$

$cor[u]$ = CINZA

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v]$ = BRANCO

$DSF\_VISIT(v)$

$cor[u]$ = PRETO

$f[u] = tempo = tempo + 1$

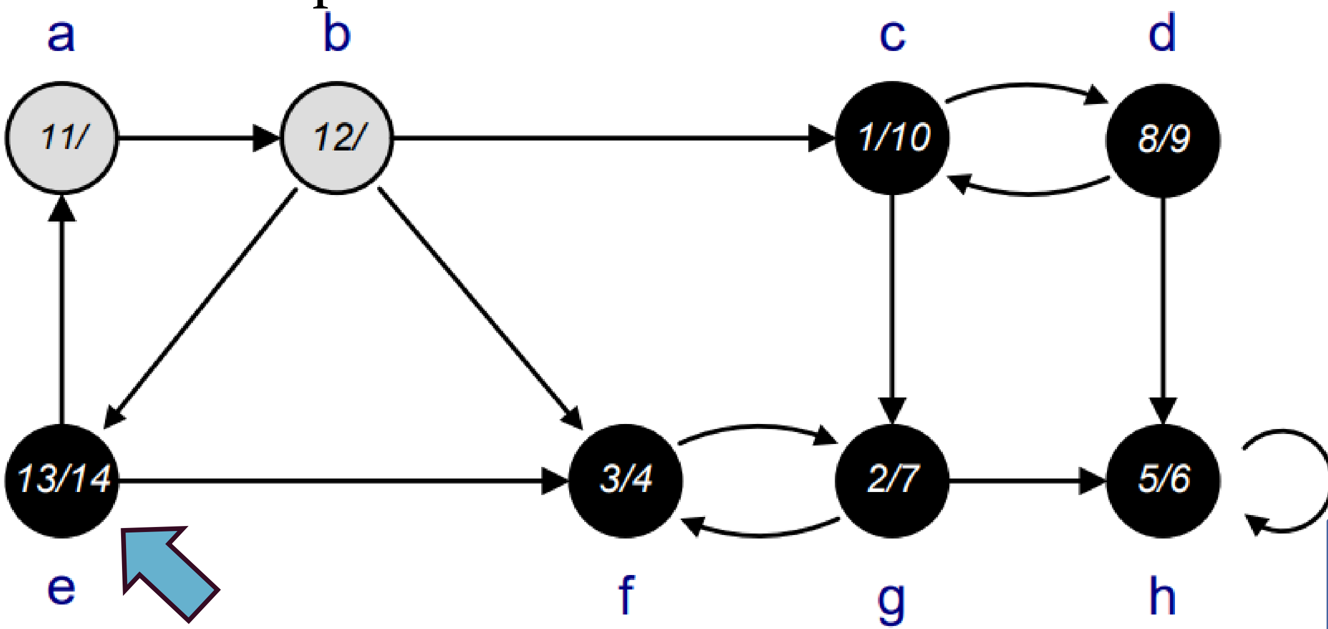
Lista: [c, a, b, d, e, f, g, h]

$tempo = 13$

- Pilha de execução:
  DSF_VISIT(**b**)
  DFS_VISIT(**a**)
  DFS(G) - próximo: u=**b**

Algoritmo DFS em grafos - exemplo

- Marca **e** de PRETO
- Incrementa o tempo
- Atribui o tempo de finalização de **e**
- Desempilha



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

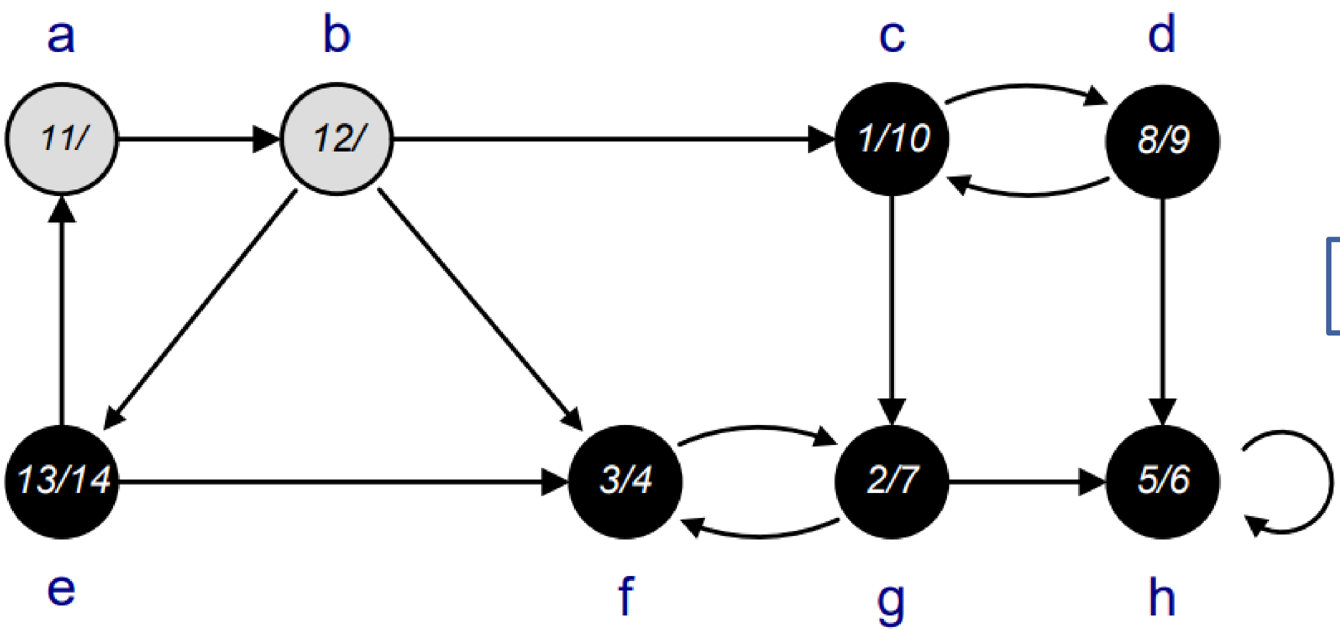
Lista: [c, a, b, d, e, f, g, h]

$tempo = 14$

- Pilha de execução:
  DSF_VISIT(**b**)
  DFS_VISIT(**a**)
  DFS(G) - próximo: u=**b**

Algoritmo DFS em grafos - exemplo

- Não possui mais nós adjacentes
- Assim, finaliza a busca em **b**



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

$\boxed{\text{Para cada } v \in Adj(u)}$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

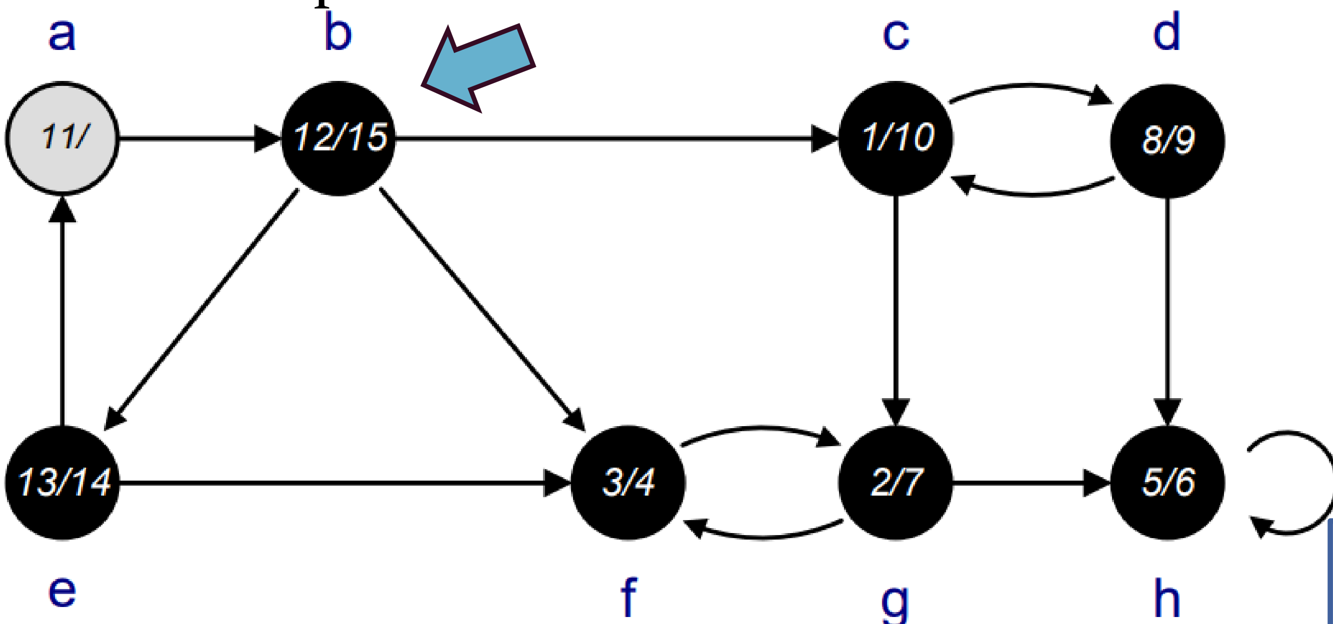
Lista: [c, a, b, d, e, f, g, h]

$\Rightarrow tempo = 14$

- Pilha de execução:
  DFS_VISIT(**a**)
  DFS(G) - próximo: u=**b**

Algoritmo DFS em grafos - exemplo

- Marca **b** de PRETO
- Incrementa o tempo
- Atribui o tempo de finalização de **b**
- Desempilha



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

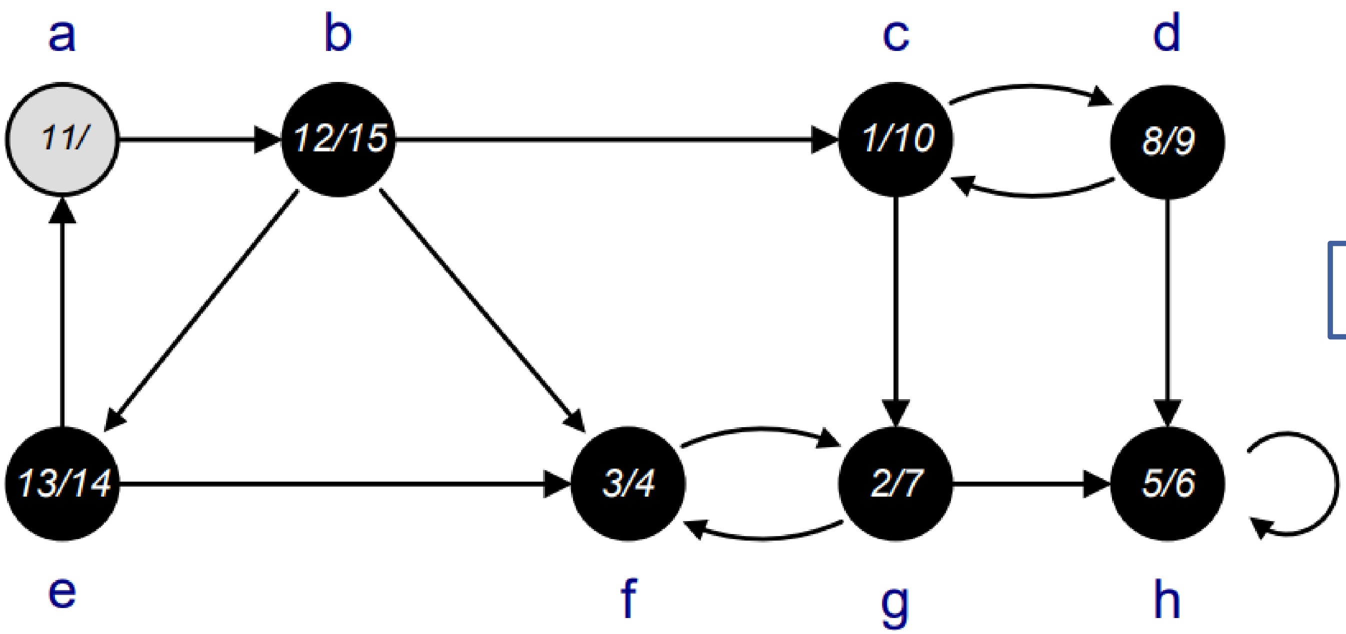
Lista: [c, a, b, d, e, f, g, h]

$tempo = 15$

- Pilha de execução:
  DFS_VISIT(**a**)
  DFS(G) - próximo: u=**b**

Algoritmo DFS em grafos - exemplo

- Desempilhou DFS_VISIT(**a**)
- Mas **a** não possui mais nós adjacentes BRANCOS
- Assim, finaliza a busca sobre **a**



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

$\boxed{\text{Para cada } v \in Adj(u)}$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$cor[u] = \text{PRETO}$

$f[u] = tempo = tempo + 1$

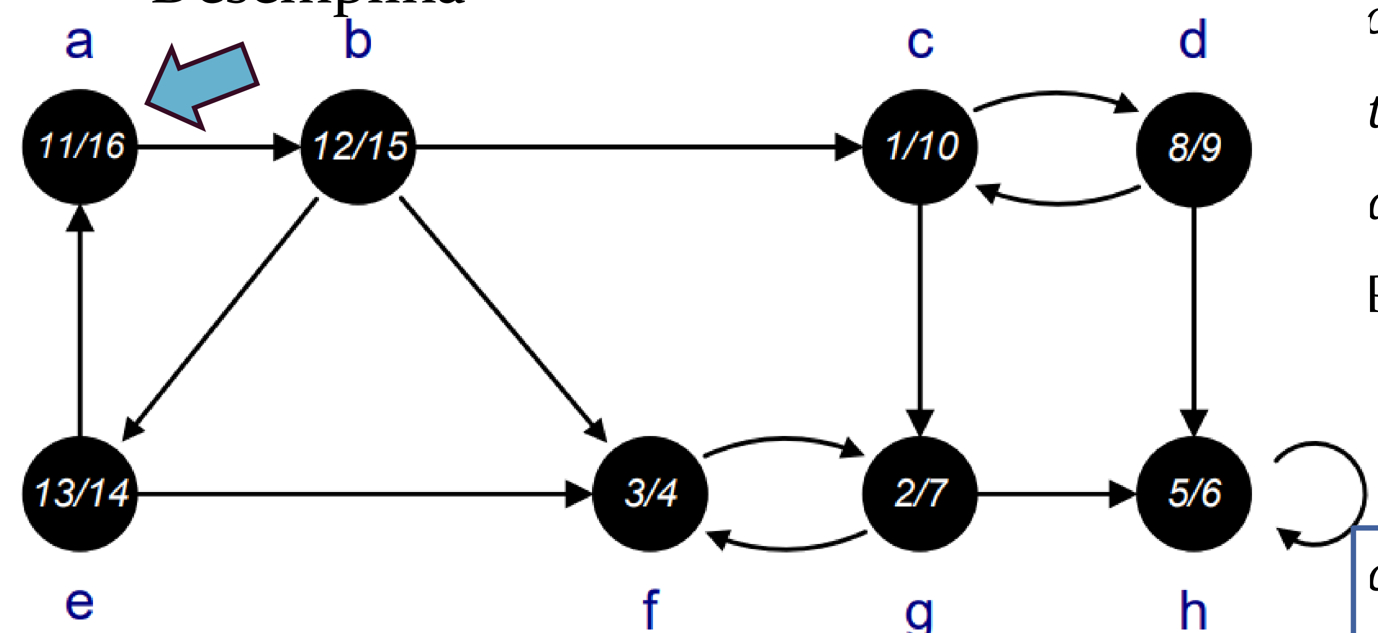
Lista: [c, a, b, d, e, f, g, h]

- Pilha de execução: DFS(G) - próximo: u=**b**

$\Rightarrow$ $tempo = 15$

Algoritmo DFS em grafos - exemplo

- Marca **a** de PRETO
- Incrementa o tempo
- Atribui o tempo de finalização de **a**
- Desempilha



$DSF\_VISIT(u)$

$cor[u] = \text{CINZA}$

$tempo = tempo + 1$

$d[u] = tempo$

Para cada $v \in Adj(u)$

se $cor[v] = \text{BRANCO}$

$DSF\_VISIT(v)$

$\boxed{cor[u] = \text{PRETO} \quad f[u] = tempo = tempo + 1}$

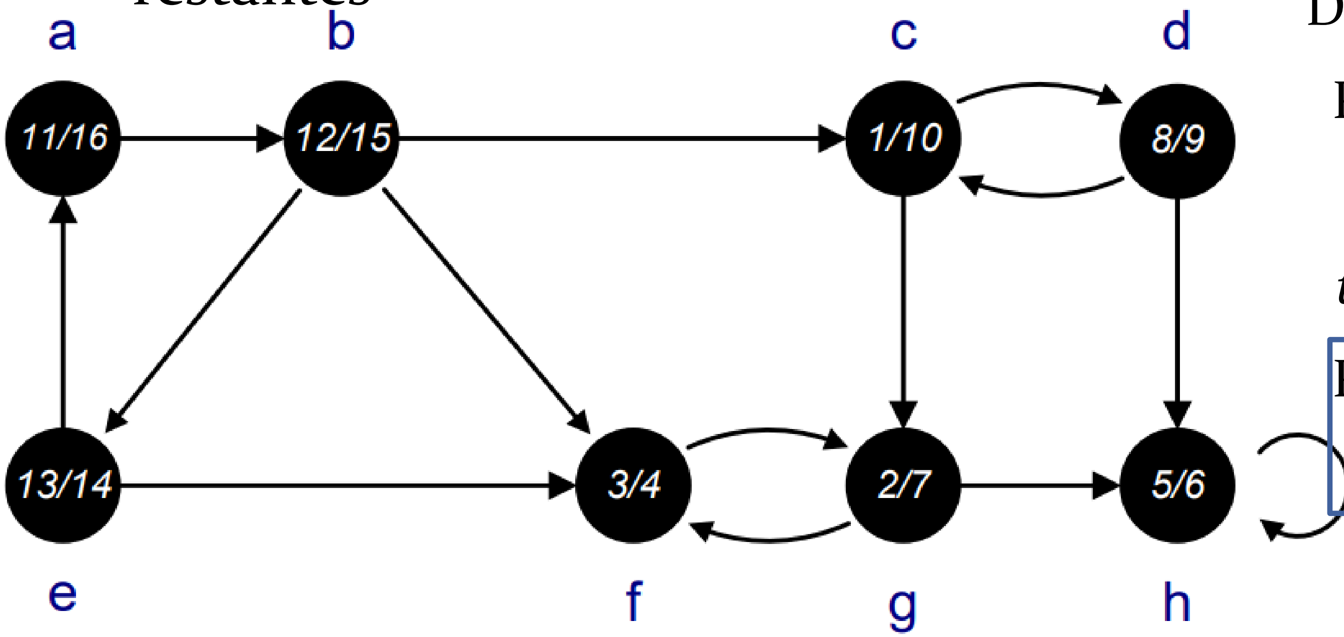
Lista: [c, a, b, d, e, f, g, h]

- Pilha de execução: DFS(G) - próximo: u=**b**

$tempo = 16$

Algoritmo DFS em grafos - exemplo

- Desempilhou DFS(G) – próximo: u=**b**
- Mas todos os nós não são mais BRANCO(S)
- Assim, a DFS(G) termina, verificando a cor de todos os nós restantes



DFS(G)

Para cada nó $u \in V$:

$cor[u] = \text{BRANCO}$

$tempo = 0$

Para cada $u \in V$

se $cor[u] = \text{BRANCO}$

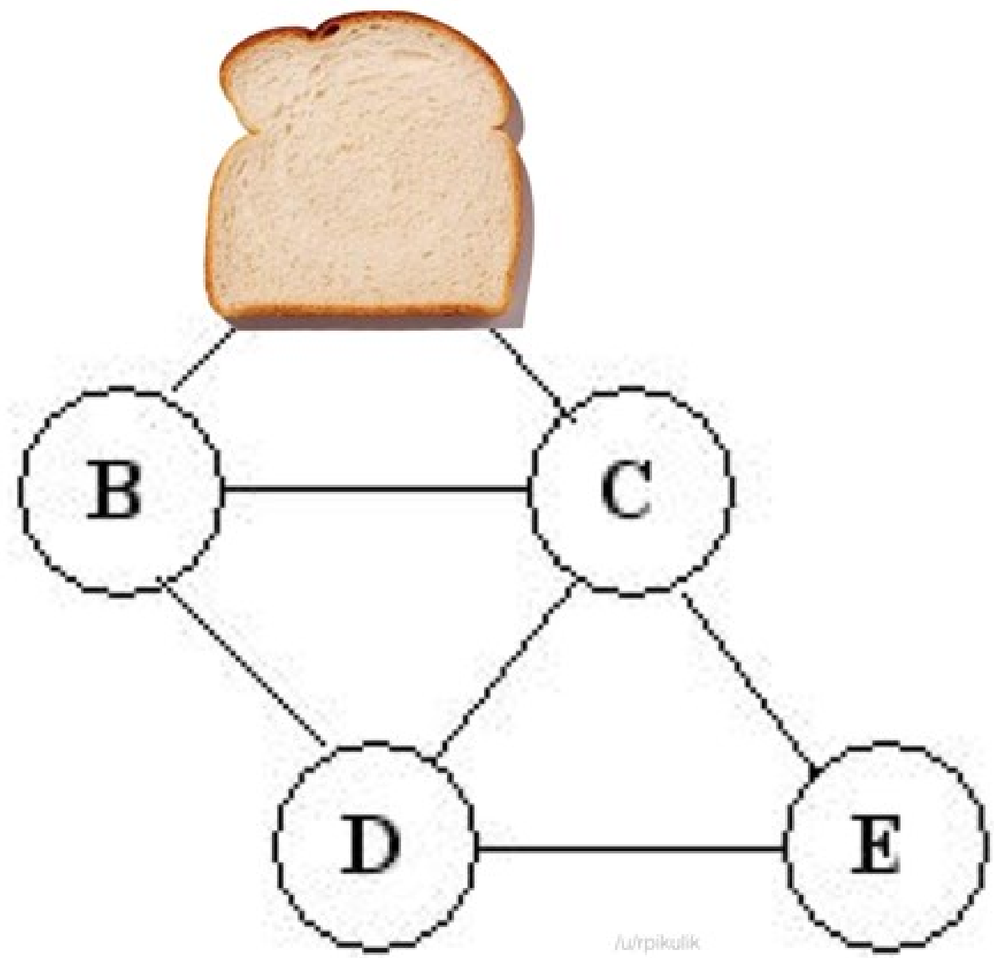
$DSF\_VISIT(u)$

Lista: [c, a, **b**, d, e, f, g, h]

- Pilha de execução: **VAZIA!!!**

$tempo = 16$

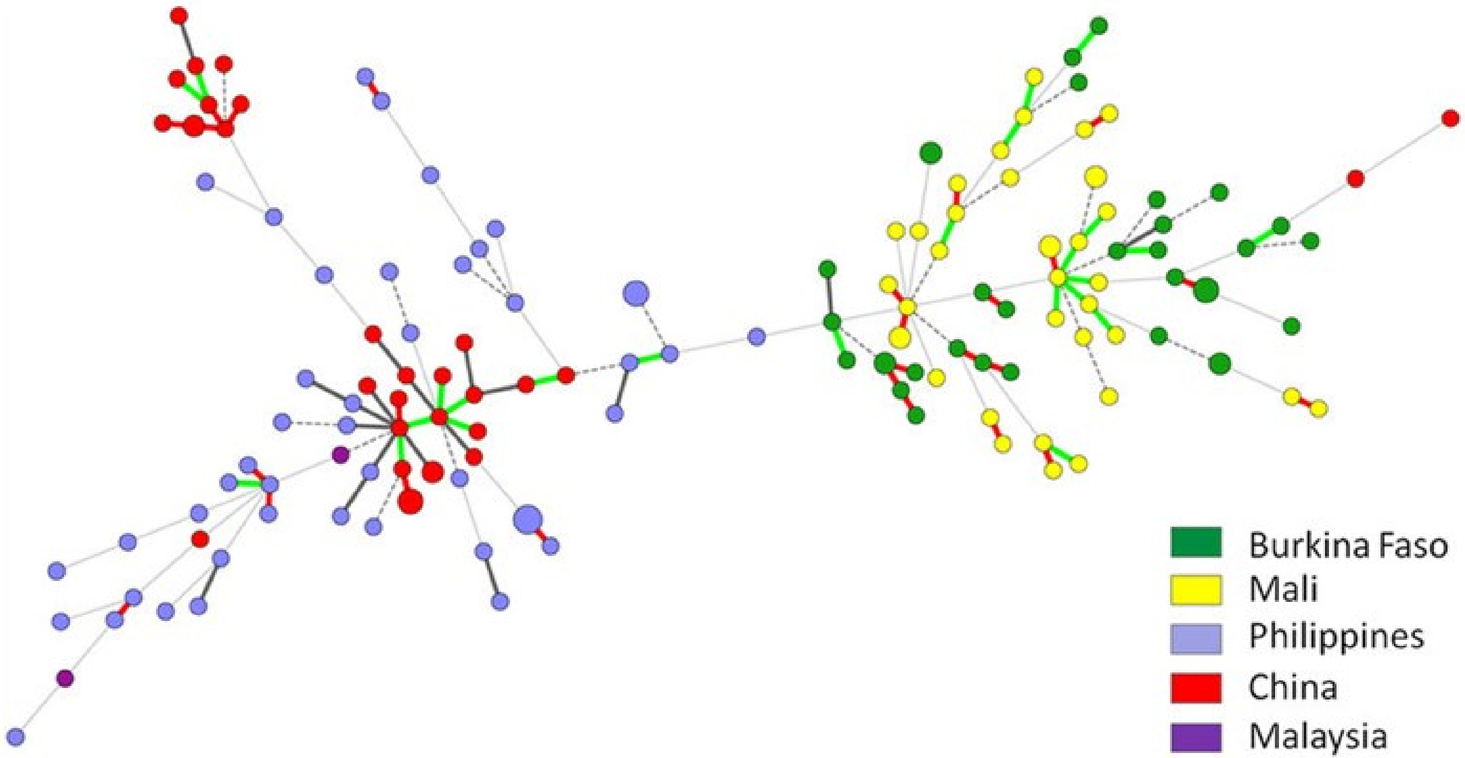
Busca em largura (Breadth-First Search)



We are done.

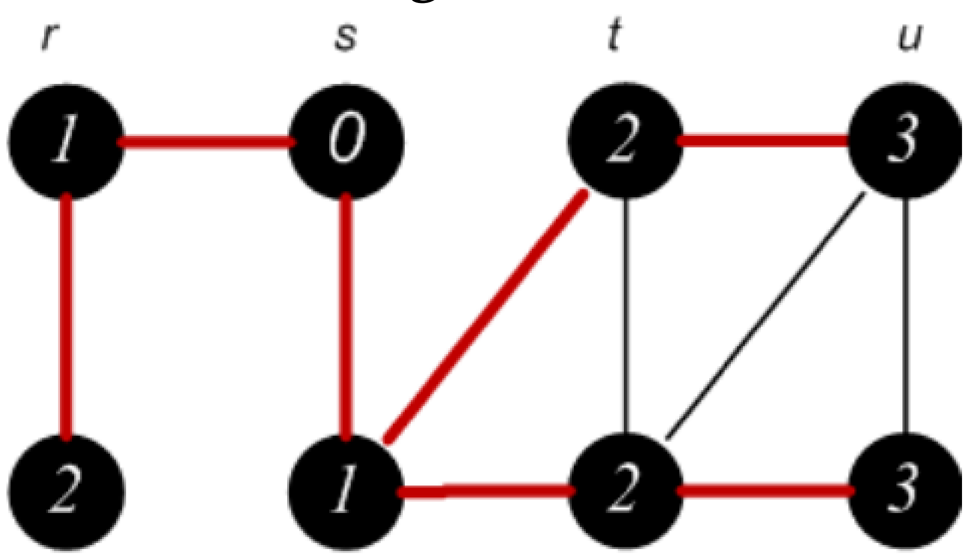
Busca em largura

- É a **base** de vários algoritmos:
  - Caminho Mínimo (Dijkstra).
  - Árvore Geradora Mínima (AGM – Prim).
    - Usado para interligar localidades a um custo mínima.



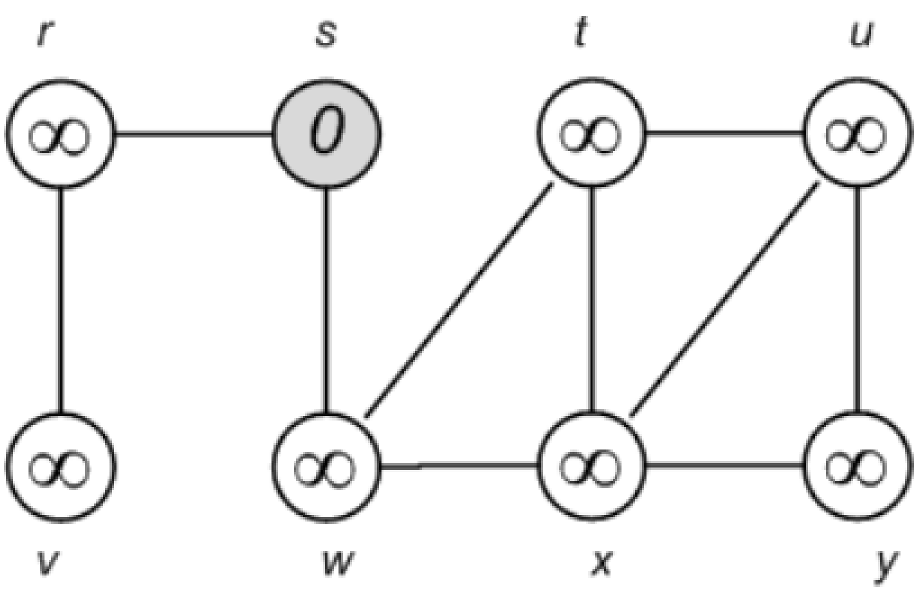
Busca em largura

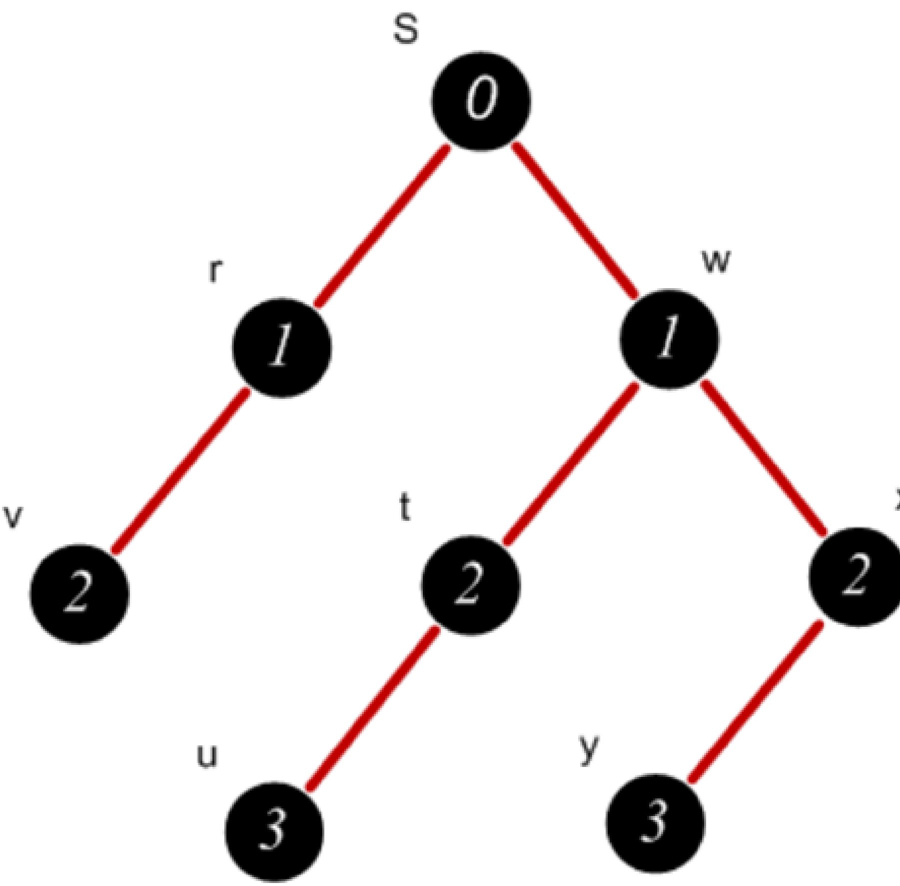
- **Como funciona?**
- O algoritmo calcula a "distância" (**menor número de arestas**) desde o nó raiz **s** até todos os nós acessíveis.
  - Não considera a distância como a soma dos pesos não-unitários das arestas.
  - Considera o número de saltos necessários mínimos para alcançar outro vértice do grafo.



Busca em largura

- Ele também produz uma "**Árvore Primeiro na Extensão**", com raiz no vértice de partida, e que contém todos os demais vértices acessíveis.

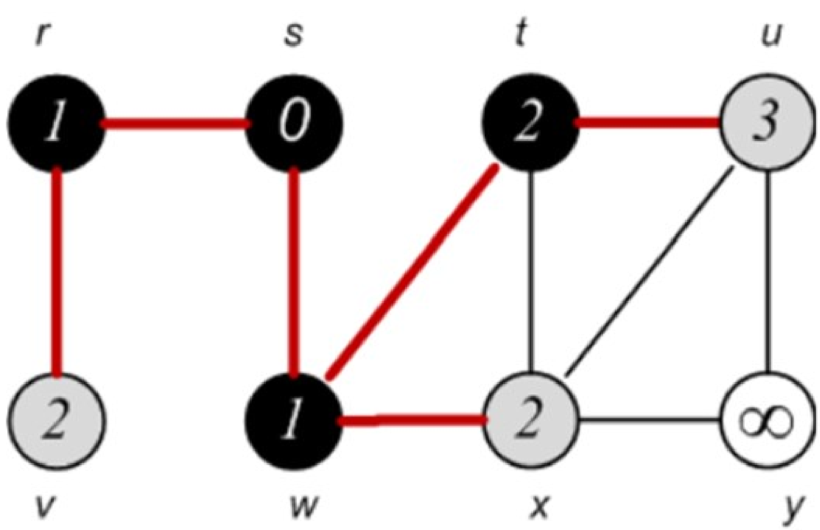




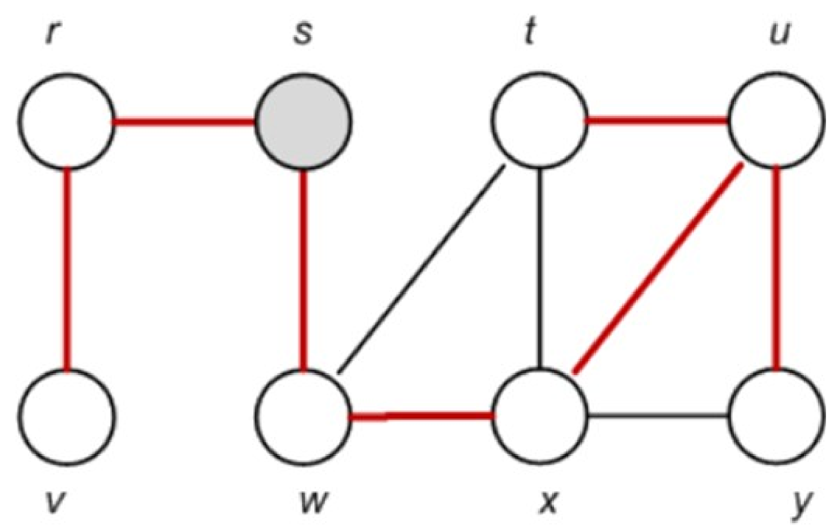
Busca em largura

- O caminho de **s** a qualquer nó **v** na **Árvore Primeiro na Extensão** corresponde a um "**caminho mais curto**" (que contém o **número mínimo de arestas**) entre esses nós.
  - Isso é possível, pois a busca é guiada "de nível em nível".
- Mas **não é possível** de se obter na busca em profundidade.

d(s,t)=2
na BFS



d(s,t)=4
na DFS



Busca em largura

- A busca em largura recebe esse nome, pois expande a fronteira entre nós descobertos e não-descobertos **uniformemente**.
- **Em resumo**: o BSF descobre todos os nós à distância **k** a partir de **s**, antes de descobrir quaisquer nós à distância **k+1.**
- Analogia com o movimento da água!!!

Algoritmo BFS em grafos

O controle do descobrimento dos nós na busca em largura (BFS) é feito de forma similar ao controle do DFS:

- **Nó Branco**: Não conhecido/não visitado.
- **Nó Cinza**: Nó conhecido/não visitado; Seus nós adjacentes não foram inseridos na fila de acesso.
- **Nó Preto**: Nó conhecido/nó visitado; Todos os seus nós adjacentes foram inseridos na fila (não necessariamente visitados, como na DFS).

Algoritmo BFS em grafos

- Um nó é **descoberto** na primeira vez em que é encontrado.
- Neste momento, ele se torna **NÃO BRANCO**.
- Assim como no DFS, nós de cor **CINZA** e **PRETA** diferenciam os nós **já localizados em duas classes**.
- Nós de cor **CINZA** podem ter alguns nós adjacentes **BRANCOS**. Eles representam a **fronteira** entre os nós descobertos e não-descobertos;

Algoritmo BFS em grafos

- A BFS constrói uma **Árvore Primeiro na Extensão** contendo inicialmente apenas sua raiz.

- Sempre que um nó **v** é descoberto no curso da varredura da lista de adjacência de um nó **u** já descoberto, o nó **v** e a aresta (**u**,**v**) são adicionados à Árvore.

- Neste caso, dizemos que **u** é predecessor (ou pai) de **v** na Árvore.

Algoritmo BFS em grafos

- Como um nó é **descoberto no máximo uma vez**, este possui apenas **um antecessor (pai)**.
  - A relação de pai depende da organização dos dados e representação do grafo (da relação de adjacência).

- **Conceito de Ancestral**:
  - Se **u** está no caminho na Árvore a partir da raiz **s** até o nó **v**, então **u** é ancestral de **v**, e **v** é um descendente de **u**.

Algoritmo BFS em grafos

- Estruturas auxiliares do algoritmo BFS:

  - cor[u]: indicativo de atingibilidade
  - p[u]: indica o nó antecessor de u (pai).
  - d[u]: distância desde a origem d(s,u) – em quantidade de arestas.
  - F: fila de acessos.

Algoritmo DFS em grafos

Input: Grafo $G = G(V, E)$ e s raiz

BFS(G, s)

  Para cada nó $u \in V \setminus \{s\}$:

    $cor[u] = \text{BRANCO}$

    $d[u] = \infty$

    $p[u] = NULL$

  $cor[s] = \text{CINZA}$

  $d[s] = 0$

  $p[s] = NULL$

  $F = DefineFila()$

  $Enfileira(F, s)$

Enquanto !Vazia(F)

  $u = Desenfileira(F)$

  Para cada $v \in Adj[u]$

    se $cor[v] = \text{BRANCO}$

      $cor[v] = \text{CINZA}$

      $d[v] = d[u] + 1$

      $p[v] = u$

      $Enfileira(F, v)$

  $cor[u] = \text{PRETO}$

Buscas em grafos

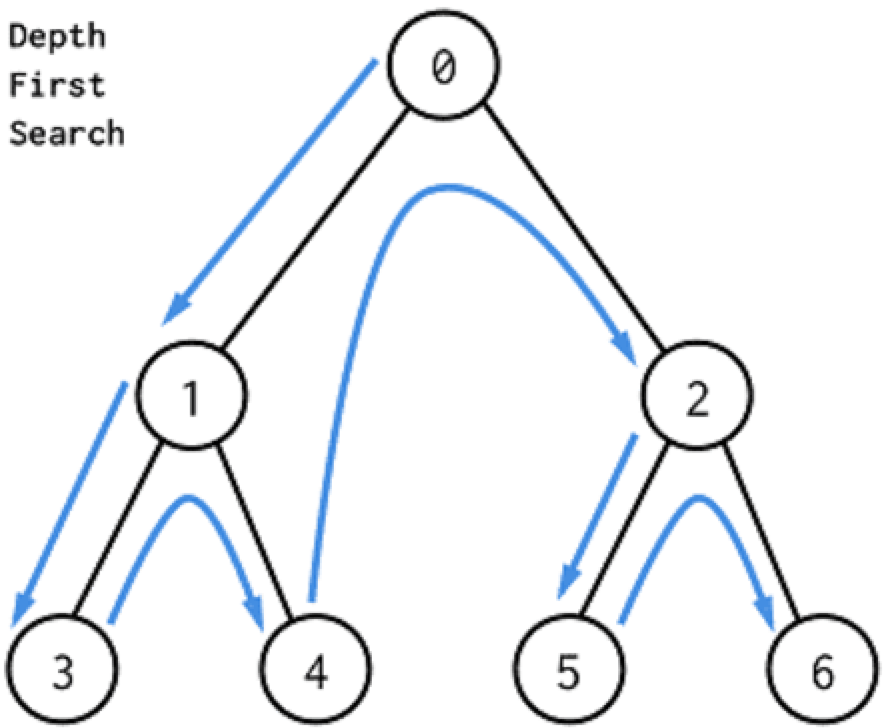
- Quanto tempo leva?

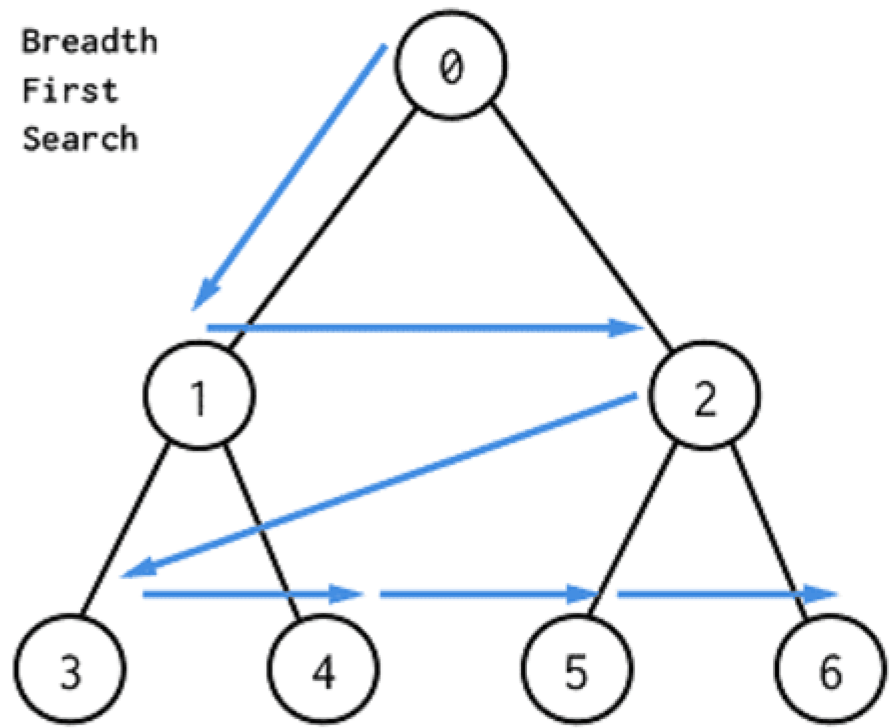
- A busca percorre todos os nós.
  - E empilha/enfileira seus nós adjacentes não visitados.

- Se usarmos Matriz de Adjacência: O(n^2).
- Se usarmos Listas de Adjacência:
  - Cada aresta é analisada apenas duas vezes.
  - Gatamos tempo O(max{n,m}) = O(n+m).
    - Linera no tamanho do grafo.

Algoritmos de busca em grafos

- Lembrar sempre da analogia em árvores binárias





Exercícios – não é para entregar

- Implementar em C ambos os algoritmos.

- Rodar "na mão", a partir de algum exemplo prático, o algoritmo BSF. Imprima os vetores **p** e **d** no final.