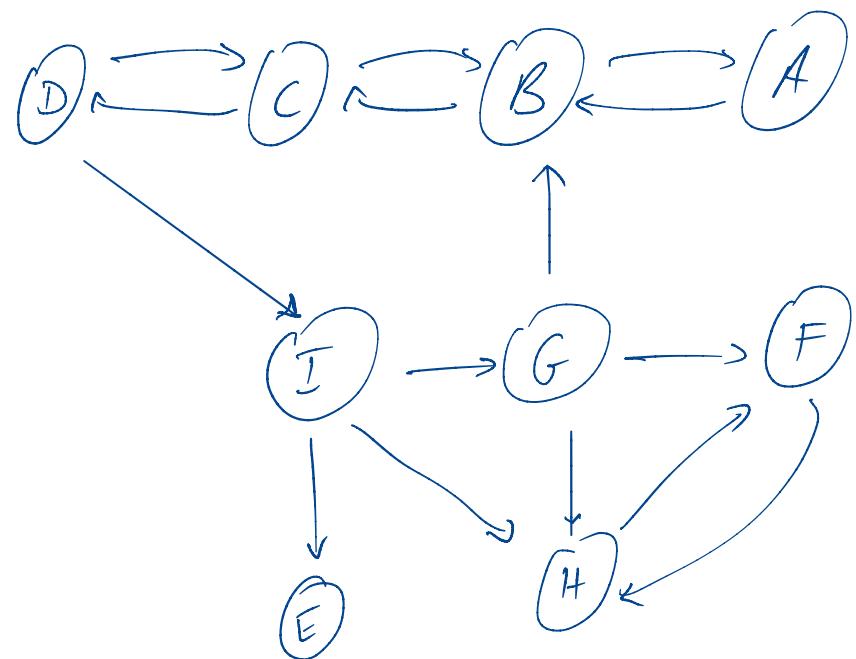


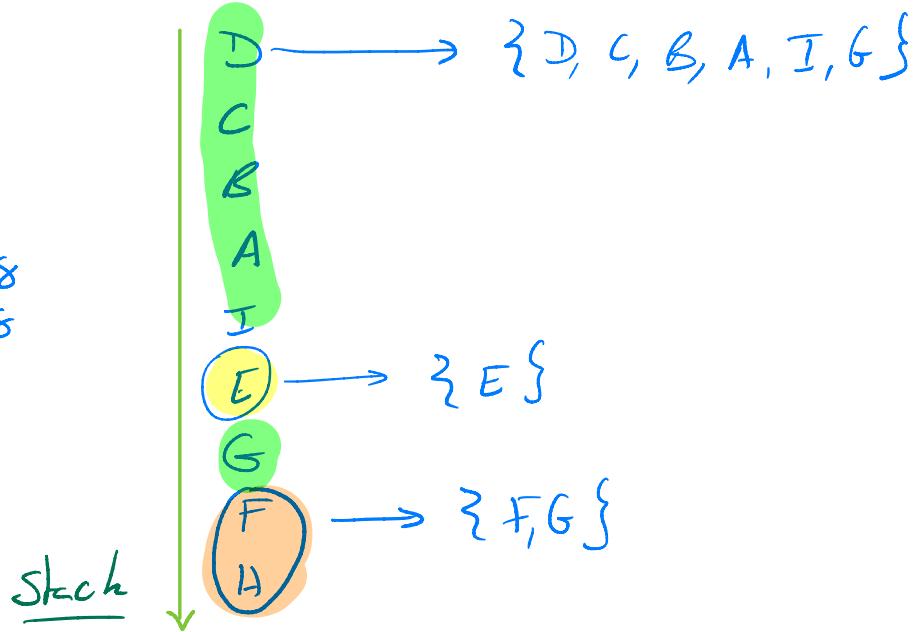
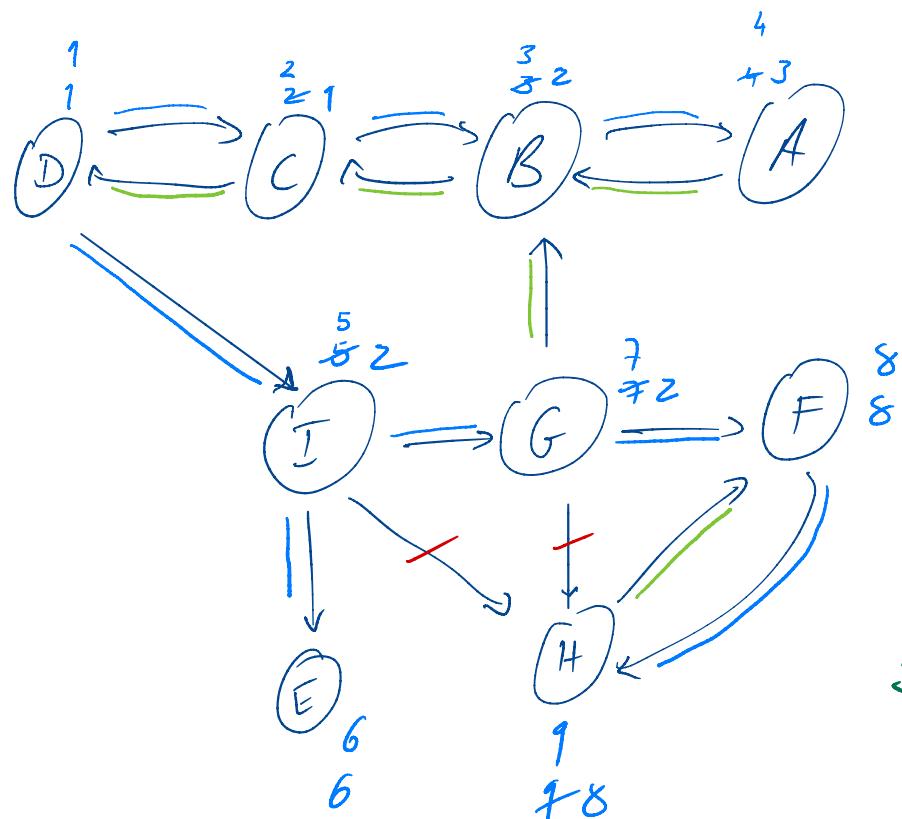
Práctica 3

R1 16-17 IB-

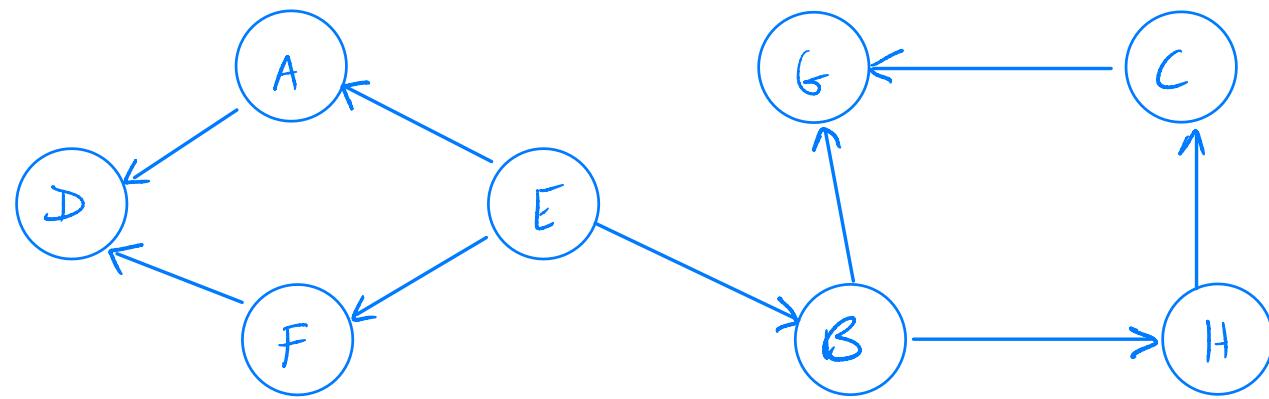
• Come ya se no véchice D



R1 16-17 I. b.

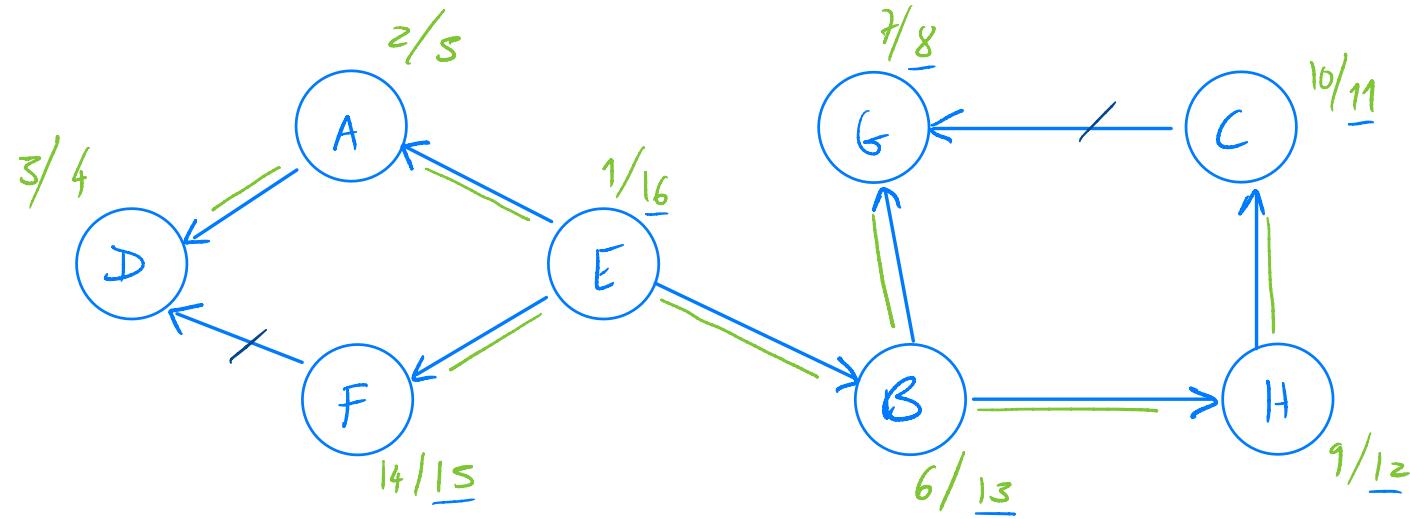


TI - 12-13 - I.b



- Começar no vértice E e visitar os vértices por ordem lexicográfica

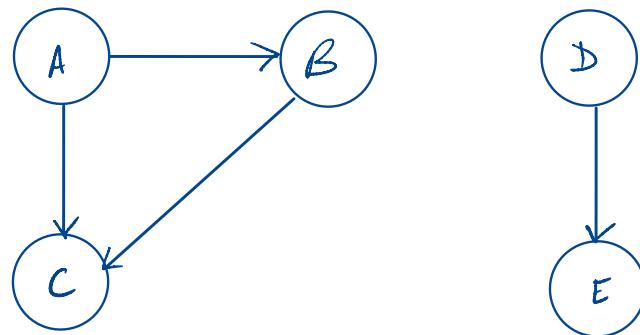
T1 - 12-13 - I.b



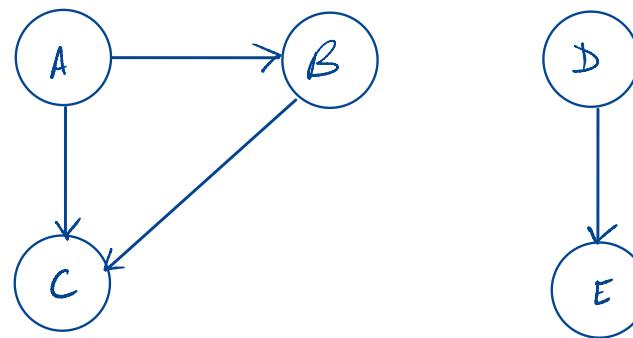
- E, F, B, H, C, G, A, D

T1 06/07 I.2

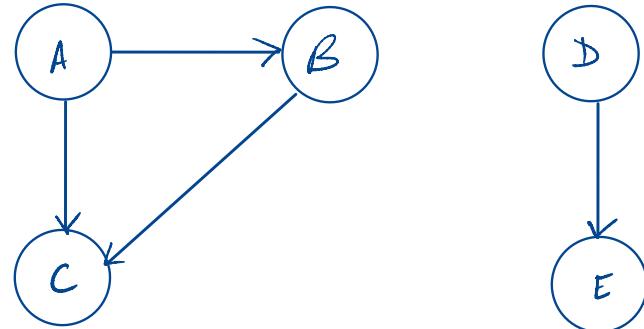
(I)



(II)



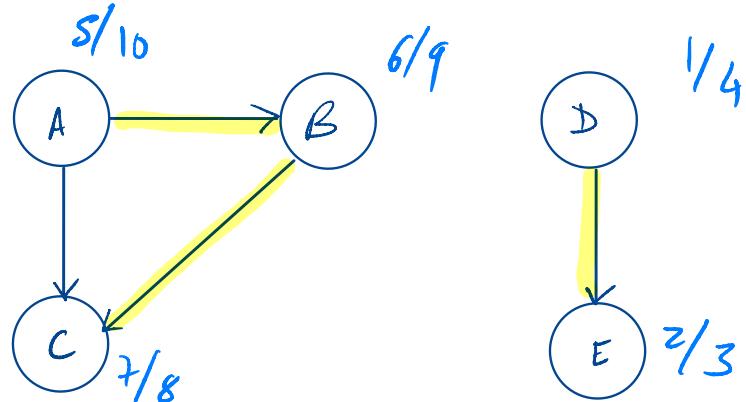
(III)



- 3 ordenações topológicas  
e respectivas DFS

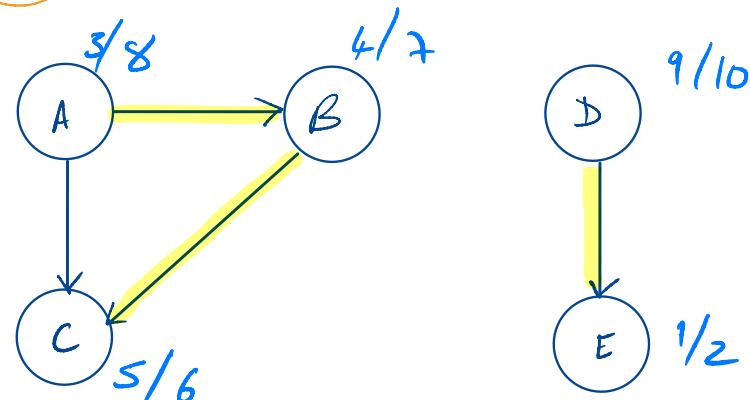
T1 06/07 I.2

I



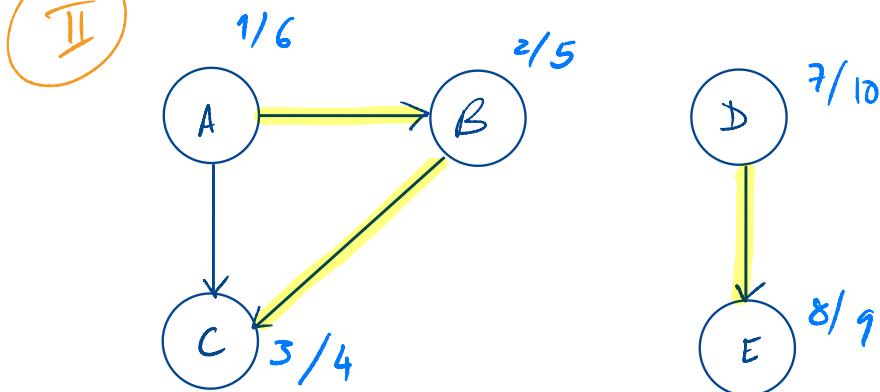
$\langle A, B, C, D, E \rangle$

III



$\langle D, A, B, C, E \rangle$

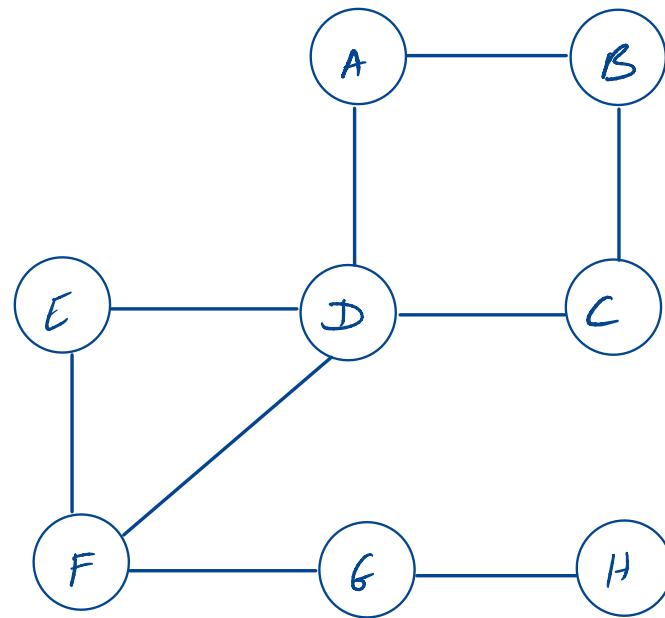
II



$\langle D, E, A, B, C \rangle$

- 3 ordenações topológicas  
e respectivas DFS

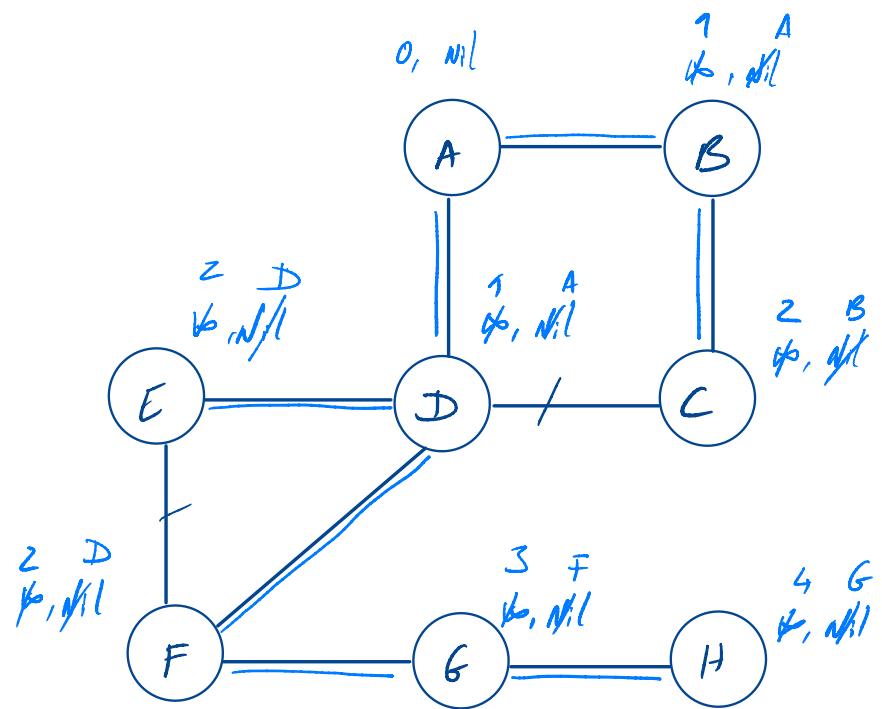
# T1 06/07 I.I



- Começar pelo nó A

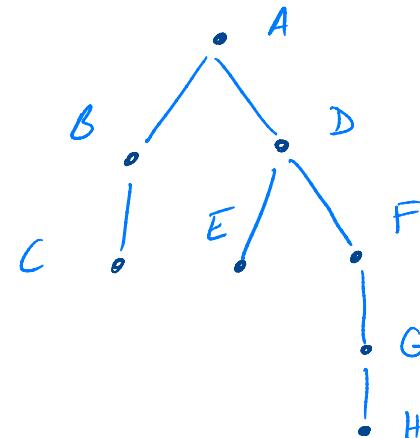
Anotamos cada nó com o par  $(d[\text{nó}], \pi[\text{nó}])$

# T1 06/07 I.I



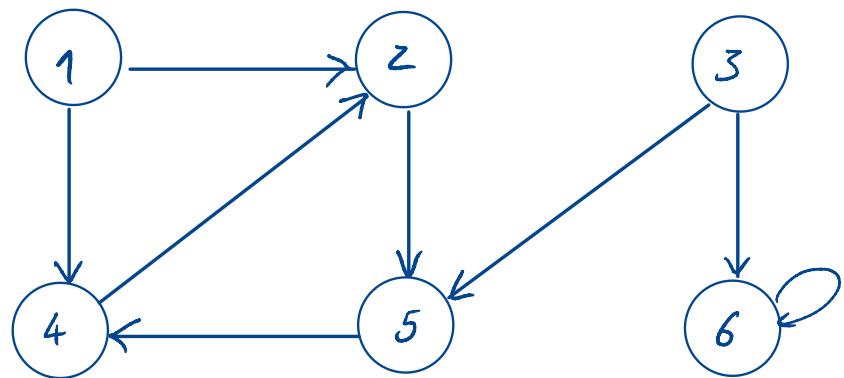
- Queue:

X  $\neq$   $\emptyset$ ,  $\emptyset$ ,  $\emptyset$ ,  $\emptyset$ ,  $\emptyset$ , X



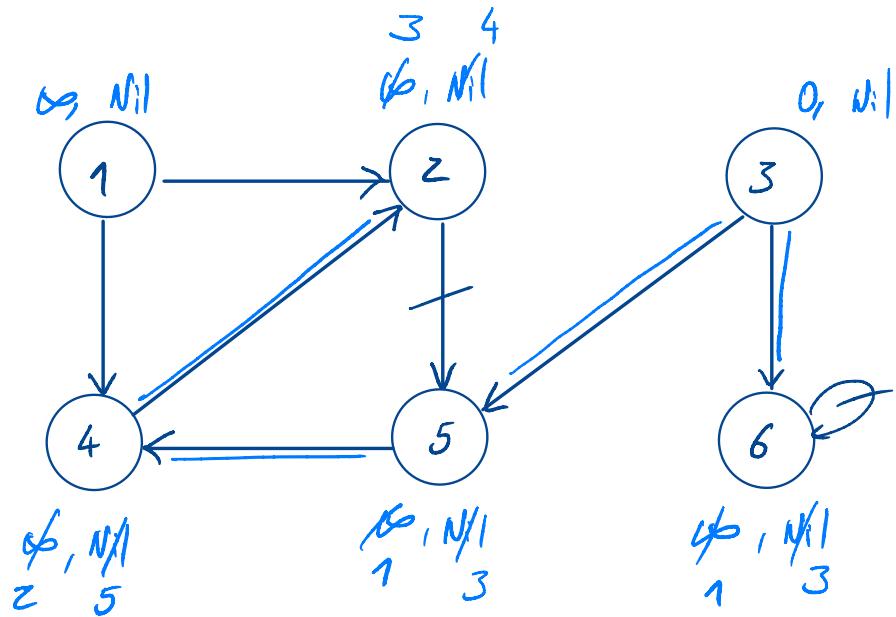
Anotamos cada nó com o par  $(d[m], \pi[m])$

## Ex 22.2-1



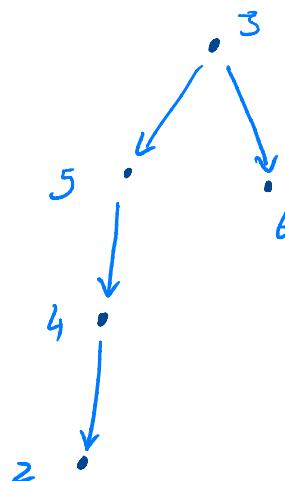
- Começar em 3 e utilizar ordem numérica ...

## Ex 22.2 - 1



- Começar em 3 e utilizar ordem numérica ...

$Q : 3, 5, 6, 4, 2$



R1 08/09 I.3

- 1) BFS permite identificar os caminhos mais curtos a partir do vértice  $s$ .
- 2) Se depois de uma BFS  $v.d > v.u + 1$   
para dois vértices  $u, v \in V$  então  $(u, v) \notin E$ .
- 3) Se o grafo for não dirigido, podem existir anacos  
pela afirmação na aplicação de BFS

R1 08/09 I.3

1) BFS permite identificar os caminhos mais curtos a partir do vértice  $s$ .

Consequência DFS:

$$\forall r \in V. \quad r.d = \delta(s, r) \quad (\text{C})$$

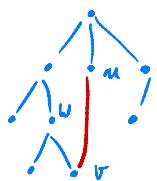
2) Se depois de uma BFS  $r.d > v.d + 1$   
para dois vértices  $u, r \in V$  então  $(u, r) \notin E$ .

$$r.d > v.d + 1$$

$$\Rightarrow \delta(s, r) > \delta(s, u) + 1$$

$$\Rightarrow (u, r) \notin E \quad (\text{desigualdade triangular}) \quad (\text{C})$$

3) Se o grafo for não dirigido, podem existir anacos  
para a frente na aplicação de BFS



$\Rightarrow$  se o anaco  $(u, v)$ , então  $v$  seria filho  
de  $u$  e anão de  $w$  porque os filhos  
de  $u$  são explorados primeiro que os  
filhos de  $w$ .

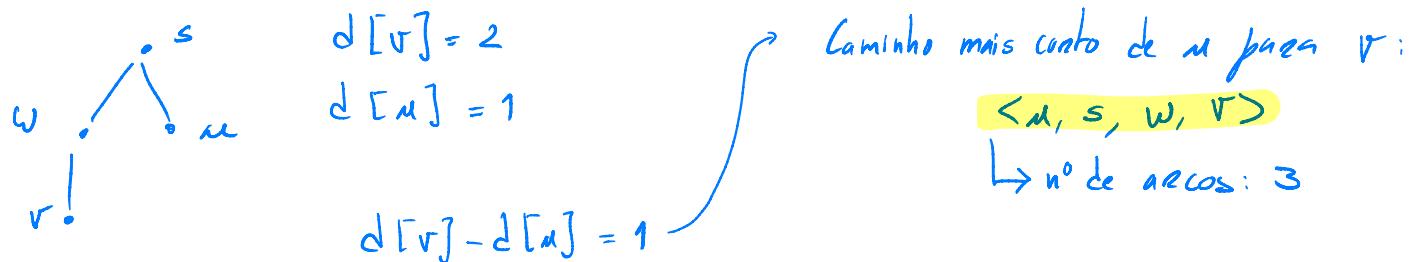
(F)

4) Sejam  $u$ ,  $v$  dois vértices atingíveis a partir de  $s$  tal que  $d[v] > d[u]$   
Então  $d[v] - d[u]$  denota o nº de arcos no caminho mais curto de  
 $u$  para  $v$ .

5) Para cada arco  $(u, v) \in BF$ ,  $d[v] = d[u] + 1$

6) Se o grafo for não-dirigido, na aplicação da BFS  
não existem arcos de cerzamento

- 4) Sejam  $s, u, v$  vértices atingíveis a partir de  $s$  tal que  $d[v] > d[u]$   
 Então  $d[v] - d[u]$  denota o nº de arcos no caminho mais curto de  
 $u$  para  $v$ .



(F)

- 5) Para cada arco  $(u, v) \in BF$ ,  $d[v] = d[u] + 1$

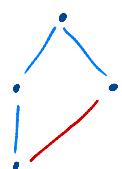
- $(u, v)$  pertence a um caminho mais curto entre  $s$  e  $v$ :

$$\delta(s, v) = \delta(s, u) + 1$$

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

(T)

- 6) Se o grafo for não-dirigido, na aplicação da BFS  
 não existem arcos de corteamento



(F)

22.5.3

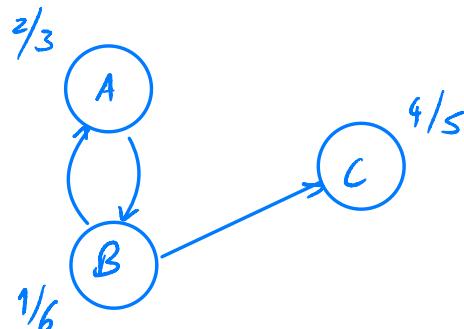
- Algoritmo p/ encontrar SCCs

- DFS(G)

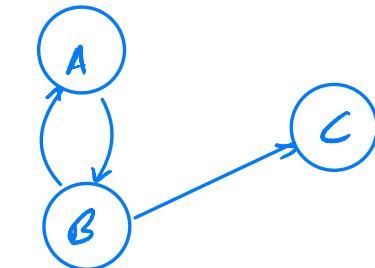
- DFS(G)  $\Rightarrow$  percorrendo os vértices por ordem crescente de tempo de fim

- Observação: O SCC com menor tempo de fim é um SCC sink

- Falha: o vértice com menor tempo de fim não pertence necessariamente ao SCC c/ menor tempo de fim.



- A 2<sup>º</sup> DFS começa pelo vértice A e encontra todo o grafo.

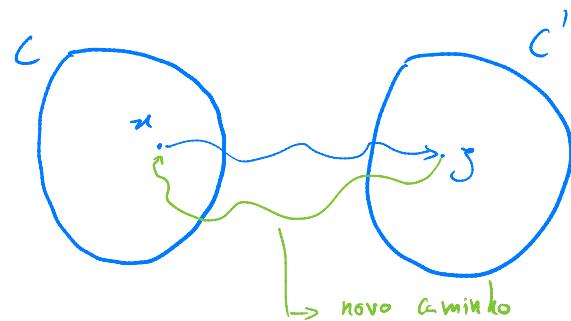


## 22.5.1

- O que acontece ao nº de SCCs de um dada grafo se adicionarmos um caminho entre dois nós

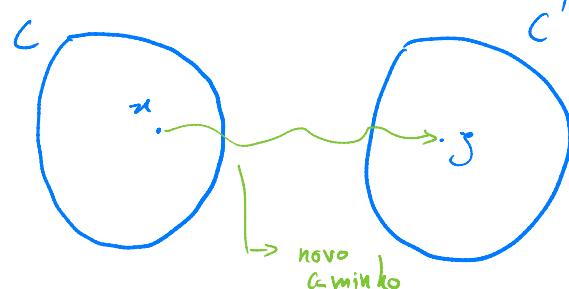
→ Diminui ou mantém-se

- Diminui

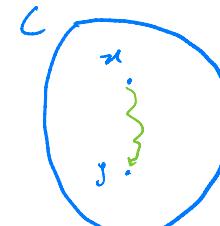


- Mantém-se

(I) Ligamos dois componentes desligados



(II) Nova ligação entre dois vértices dentro do mesmo componente

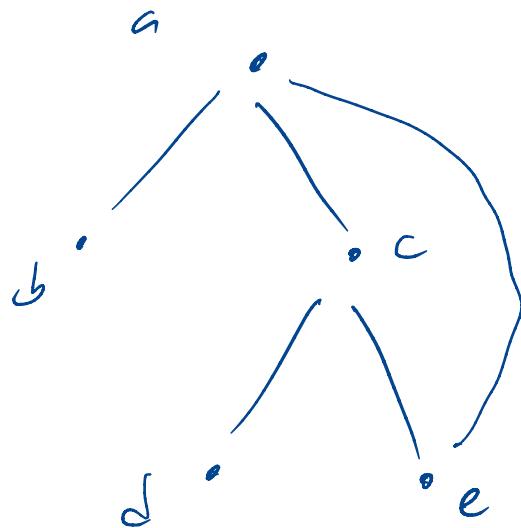


22.4 - 3

Determinar se um grafo não dirigido contém um ciclo simples em tempo  $O(V)$ .

- DFS modificada

Exemplo



22.4 - 3

Determinar se um grafo não dirigido contém um ciclo simples em tempo  $O(V)$ .

- DFS modificada

DFS( $G$ )

for  $v \in G.V$

$v.\text{visited} := \text{false}$ ;  $v.\pi := \text{nil}$

for  $v \in G.V$

  if ( $\neg v.\text{visited}$   $\&$  DFS-visit( $G, v$ ))

  return true

return false

DFS-Visit( $G, v$ )

$v.\text{visited} := \text{true}$

  for each  $u \in G.\text{Adj}[v]$

    if ( $\neg u.\text{visited}$ ) {

$u.\pi := v$ ;

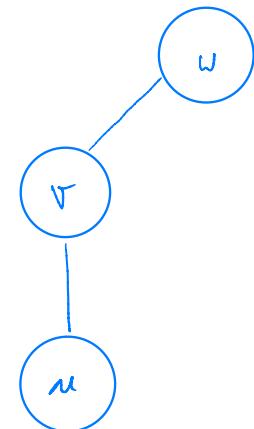
      return DFS-Visit( $G, u$ )

    else if ( $v.\pi \neq u$ ) {

      return true

}

{  
return false



22.4 - 3

Determinar se um grafo não dirigido contém um ciclo simples  
em tempo  $O(V)$ .

- DFS modificada

DFS( $G$ )

for  $v \in G.V$

$v.visited := \text{false}$ ;  $v.PI := \text{nil}$

for  $v \in G.V$

  if ( $\neg v.visited \& \text{DFS-visit}(G, v)$ )

  return true

return false

DFS-Visit( $G, v$ )

$v.visited := \text{true}$

for each  $u \in G.\text{Adj}[v]$

  if ( $\neg u.visited$ ) {

$u.PI := v$ ;

    return DFS-Visit( $G, u$ )

  else if ( $v.PI \neq u$ ) {

    return true

}

return false

22.4 - 3

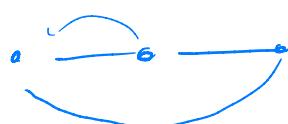
Determinar se um grafo não dirigido contém um ciclo simples em tempo  $O(V)$ .

Loop 1

- É executado no máximo  $|V|$  vezes  
(Vertices visitados não são re-visitados)

Loop 2

- O loop 2 é executado no máximo  $|E|$  vezes  
mas um grafo não dirigido acíclico tem no  
máximo  $|V|-1$  arcos.
- Assim o loop 2 é executado, no pior caso,  $2|V|-1$  vezes.  
O arco  $2|V|-1$  é necessariamente arco para trás  
porque num DFS num grafo não dirigido só havia  
arcos p/ trás e arcos da árvore.



DFS(6)

for  $v \in G.V$

$v.visited := false; v.PI := nil$

Loop 1

for  $v \in G.V$

if ( $v.visited \&& DB-visit(G, v)$ )

return true

return false

DFS-Visit( $G, v$ )

$v.visited := true$

for each  $u \in G.Adj[v]$

if ( $u.visited$ ) {

$u.PI := v;$

return DFS-Visit( $G, u$ )

else if ( $v.PI \neq u$ ) {

return true

Loop 2

}

}

}

return false

22.5 - 5

- Depois de calcular os SCCs calcular o grafo dos SCCs em tempo linear.

- Associam a cada SCC um id único e anotam cada vértice do grafo original com o id do seu SCC. Escreveremos  $\text{al.scc}$  para denotar o id do SCC a que u pertence.

Compute  $\text{ESCC}(G, \text{GSCC})$

$k := |\text{GSCC.V}|$  // number of SCCs

let  $A[1..k]$  be a new array whose elements are initially 0

for each SCC index  $i$  in  $\text{GSCC.V}$

    let  $C$  be the vertices of  $G$  in SCC  $i$

    for each  $m \in C$

        for each  $v \in G.\text{Adj}[m]$

            if  $((m.\text{scc} \neq v.\text{scc}) \text{ or } (A[v.\text{scc}] == 0))$

$\text{GSCC.addEdge}(m.\text{scc}, v.\text{scc})$

$A[v.\text{scc}] := 1$

    for each  $m \in C$

        for each  $v \in G.\text{Adj}[m]$

$A[v.\text{scc}] := 0$

$$\text{GSCC} = (\text{V}_{\text{SCC}}, \text{E}_{\text{SCC}})$$

-  $\text{V}_{\text{SCC}} \Rightarrow$  ids dos SCCs de  $G$

-  $\text{E}_{\text{SCC}}$

Observação: Visitar as adjacências dos vértices de cada SCC, num SCC de cada vez, mantendo um array  $A$  de 0s e 1's de tamanho igual ao n° de SCCs.

$A[j] = 1$  se já foi encontrado um arco a ligar o SCC que a ser visitado ao SCC  $j$ .

Complexidade:

$$\sum_{v \in \text{ESCC}(G)} \left( \sum_{m \in V_i} \left( O(1) + \sum_{v \in \text{Adj}[m]} O(1) \right) \right) = O(|V| + |E|)$$

Ex 22.2 - 7

Um grafo não dirigido diz-se **bipartido** se podemos dividir o conjunto  $V$  de vértices em dois conjuntos  $V_1, V_2$  tais que:

- $V_1 \cup V_2 = V$
- $V_1 \cap V_2 = \emptyset$
- $\text{out}(V_1) \subseteq V_2 \quad \wedge \quad \text{out}(V_2) \subseteq V_1$

$$\begin{aligned}\text{out}(v) &= \{u \mid (v, u) \in G.E\} \\ \text{out}(V) &= \bigcup_{v \in V} \text{out}(v)\end{aligned}$$

Como determinar se um grafo é bi-partido?

- A ideia é alterar a BFS de forma a associar uma de duas cores a cada vértice quando este é encontrado pelo DFS. )  $O(|V| + |E|)$
  - O grafo é bi-partido se não forem encontrados arcos )  $O(|E|)$ )  
- q ligam vértices da mesma cor.
- $\left. \begin{array}{l} O(|V| + |E|) \\ O(|V| + |E|) \end{array} \right\}$