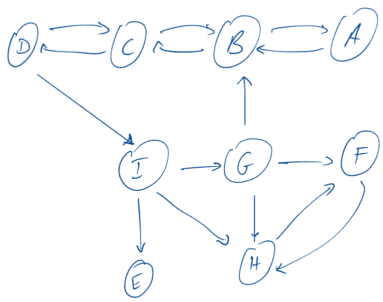
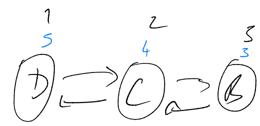
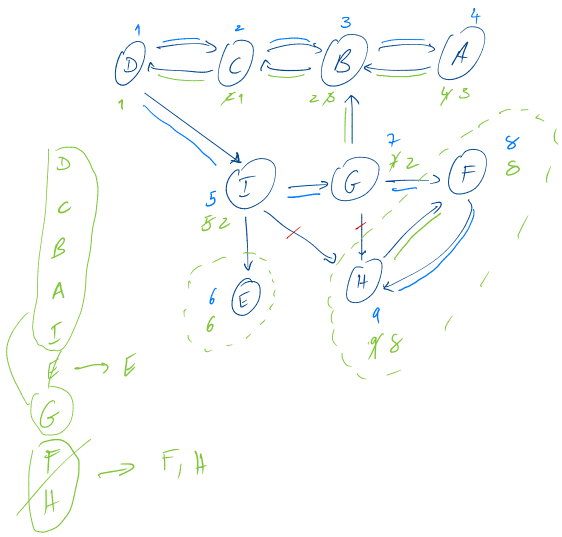
Prática 3

- Componentes Fortemente Ligados
- Ordenação Topológica
- BFS

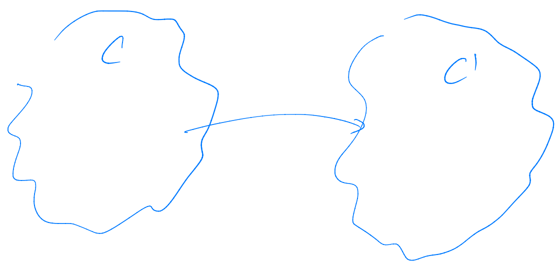
R1 16-17 I.b -







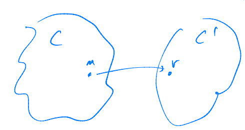
22.5-3



Lema Chave

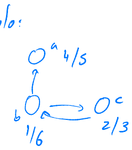
$(u,v) \in E$, $u \in C$ e $v \in C'$

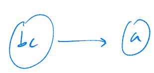
$\Rightarrow f(C) > f(C')$



$\Rightarrow$ O lema chave garante q̄ $f(C) > f(C')$ mas o vértice c/ menor tempo de fim pode estar em C.

Por exemplo:





$f(bc) = 6$

$f(a) = 5$

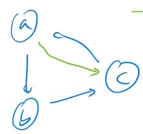
mas c é o vértice com menor tempo de fim!

Portanto, a 2ª DFS vai produzir o resultado errado: um único componente $\{a, b, c\}$

22.5-1

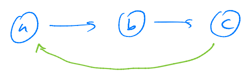
- O nº de SCCs pode:
  a) ficar na mesma
  b) diminuir

a)



→ ⇒ novo arco

b)



- O nº de SCCs não pode aumentar.

Suponhamos q̃ o nº de SCCs aumenta. Seja G o grafo original e G' o grafo estendido com um arco adicional.

Concluímos q̃ existe pelo menos um SCC, seja C, q̃ se divide em vários.

Admitimos, sem perda de generalidade, q̃ C se divide em dois SCCs, C_1 e C_2. Portanto, $C = C_1 \cup C_2$.

Isto significa que existem dois vértices x e y tais que:

$x \in C_1$, $y \in C_2$ e $(G' \vdash x \not\to y \vee G' \vdash y \not\to x)$

Em G', x não é atingível a partir de y.

De onde segue que:

$x \in C$, $y \in C$ e $(G \vdash x \not\to y \vee G \vdash y \not\to x)$

De onde concluímos que C não é um SCC de G, obtendo uma contradição.

22.4-3

- DFS Modificada ⇒ Paramos assim q̃ encontramos o primeiro arco para trás.

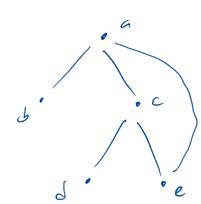
Lema Chave:

Um grafo contém um ciclo sse uma DFS revela um arco para trás.

Algoritmo 1 (DFS')

```
DFS'(G)
  for each v ∈ V[G]
      v.color = white ; v.π = Nil
  time = 0
  for each v ∈ V[G]                      loop1
    if v.color == White
      if(DFS-visit(G,v))
        return true
  return false
```

```
DFS-Visit(G, v)
  if(v.color != White) error "..."
  time = time+1; v.d = time
  v.color = Grey
  for each u ∈ Adj(G,v)                  loop2
    if(u.color == white)
      u.π = v
      time = DFS-Visit(G,u,time)
    else if(v.π ≠ u) return true
  v.color = Black; time = time+1; v.f = time
```



Análise Agregada:

Loop 1 ⇒ É executado no máximo $|V|$ vezes (mesma justificação q̃ a DFS)

Loop 2 ⇒ É executado no máximo $|V|$ vezes:

- A DFS num grafo não dirigido só revela arcos de árvore e arcos p/ trás.
  Nº de arcos de árvore: $|V|-1$.

De modo q̃ o $|V|$-ésimo arco a ser percorrido terá de ser necessariamente para trás.

22.5 - 5

- Associar a cada SCC um id único e marcar cada vértice do grafo original com o id do seu SCC (u.scc denota o id do scc q contém u). $O(|V|)$
- $G_{SCC} = (V_{SCC}, E_{SCC})$ $V_{SCC} \Rightarrow$ Conjunto dos ids dos SCCs.
- Calcular E_{SCC}

number of SCCs

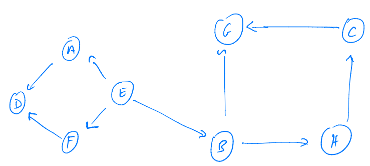
```
let A[1..k] be a new array whose elements are initialized to 0
for each SCC index i
    let V_i be the set of vertices of the original graph in the SCC with index i
    for each u ∈ V_i
        for each v ∈ G.Adj[u]
            if ((u.scc ≠ v.scc) && (A[v.scc] == 0))
                then A[v.scc] = 1; Add (u.scc, v.scc) to E_scc
    for each u ∈ V_i
        for each v ∈ G.Adj[u]
            A[v.scc] = 0
```

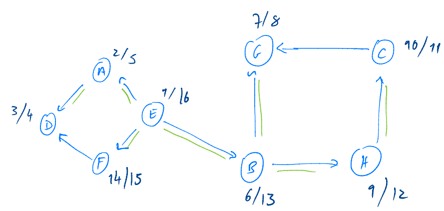
Observação: Visitar as adjacências dos vértices de cada SCC, um SCC de cada vez, mantendo um array A de 0's e 1's de tamanho igual ao nº de SCCs.
$A[j] = 1$ sse já foi encontrado um arco a ligar o SCC q está a ser visitado ao SCC j.

Complexidade:

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

TI - 12-13 - I.b

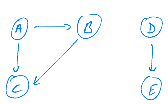




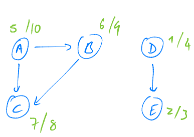
Ordenação Topológica

< E, F, B, H, C, G, A, D >

TI 06/07 I.2 -

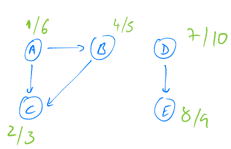


1)



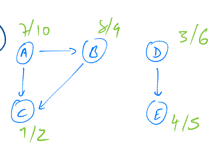
< A, B, C, D, E >

2)



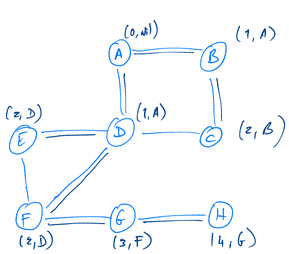
< D, E, A, B, C >

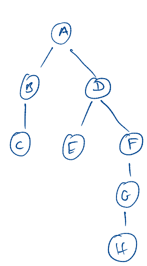
3)



< A, B, D, E, C >

TI 06/07 I.1 -

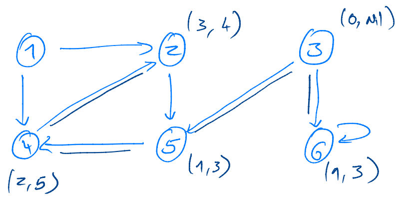


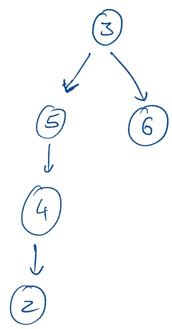


Anotamos cada nó com o par (d[u], π[u])

Ex 22.2-1

$u \Rightarrow (d[u], \pi[u])$





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:

- $V_1 \cup V_2 = V$
- $V_1 \cap V_2 = \emptyset$
- $out(V_1) \subseteq V_2 \ \wedge \ out(V_1) \subseteq V_2$

$out(v) = \{ u \mid (v,u) \in G.E \}$

$out(V) = \bigcup_{v \in V} out(v)$

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 pela DFS. $O(|V|+|E|)$
- O grafo é bi-partido se não forem encontrados arcos q̄ ligam vértices de mesma cor. $O(|E|)$

$O(|V|+|E|)$

R1 08/09 I.3

1. V (Correcção de BFS. $\forall v \in V.\ v.d = \delta(s,v)$)
2. V

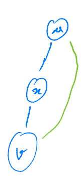
$v.d > u.d+1 \Rightarrow (u,v) \notin E$

$\Leftrightarrow (u,v) \in E \Rightarrow v.d \leq u.d+1 \quad (a \Rightarrow b \Leftrightarrow \neg b \Rightarrow \neg a)$

$\Leftrightarrow \underbrace{(u,v) \in E \Rightarrow \delta(s,v) \leq \delta(s,u)+1}_{\text{Desigualdade Triangular V}}$ (Correcção de BFS)

3. F

Suponhamos q̄ (u,v) é um forward edge. Então existe pelo menos um vértice x tal que:

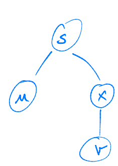


Neste caso, temos que:

$v.d > u.d+1 \quad \therefore$

4. F

- Considere o grafo:



$v.d - u.d = 1$

distância entre u e $v = \infty$

5 - V

6 - F