

Grafos

- Representação

- DFS

- Resoluções
Elementares

Grafo - Definições Elementares

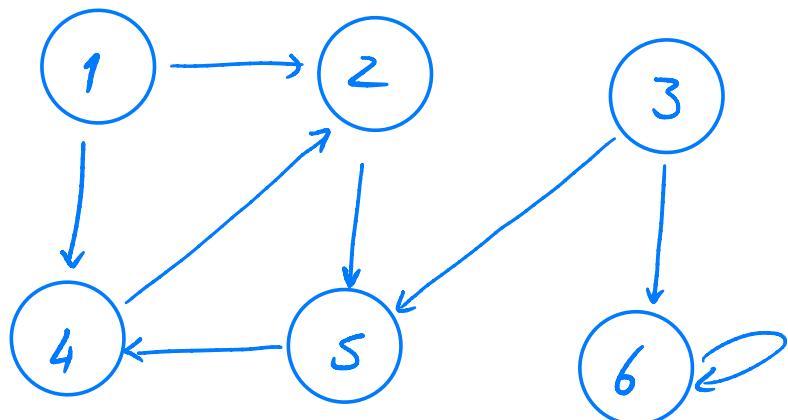
Definição: Um grafo é um par $G = (V, E)$

onde:

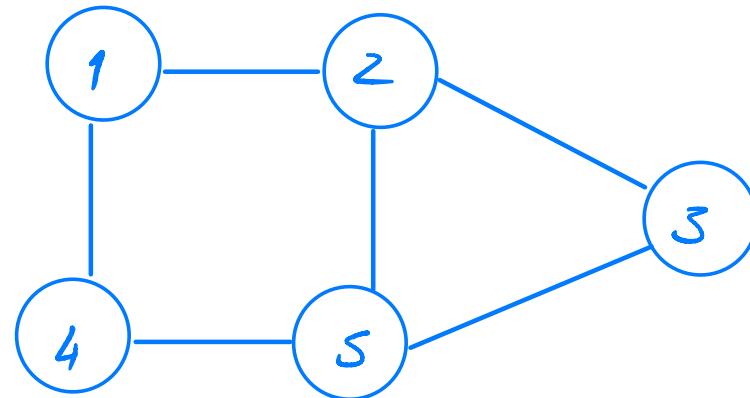
- V - conjunto de vértices
- E - conjunto de arestas ($E \subseteq V \times V$)

• Um grafo diz-se:

- Esparsa: se $|E| \in O(|V|)$
- Denso: se $|E| \in O(|V|^2)$

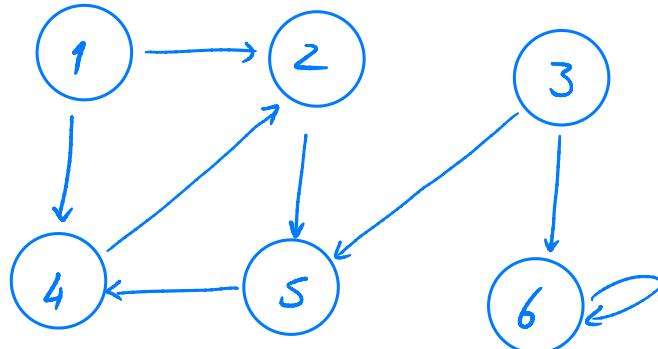


Dirigido: As arestas têm sentido

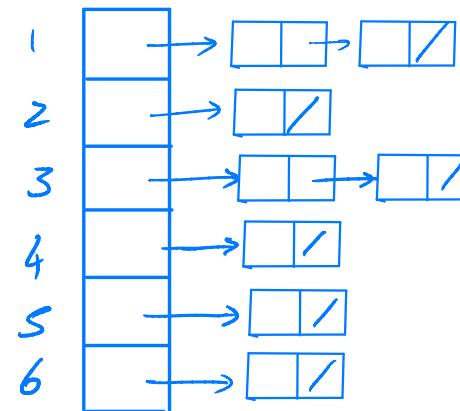


Não dirigido: As arestas não têm sentido

Grafos - Representação



Lista de Adjacências



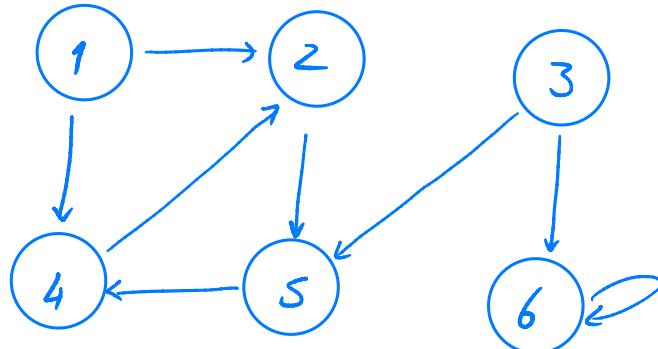
Espaço:
Grafos
Espaços:

Matriz de Adjacências

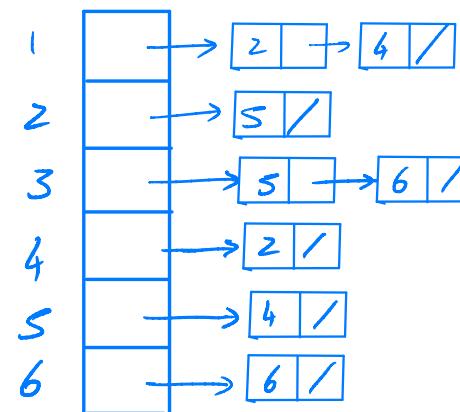
	1	2	3	4	5	6
1						
2						
3						
4						
5						
6						

Espaço:

Grafos - Representação



Lista de Adjacências



Espaço: $O(|V| + |E|)$

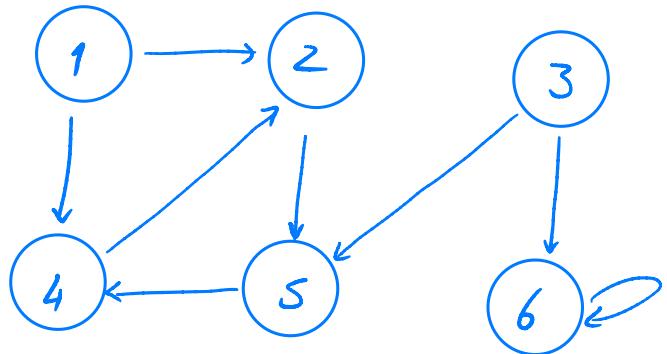
Grafos
Espaços: $O(|V|)$

Matriz de Adjacências

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

Espaço: $O(|V|^2)$

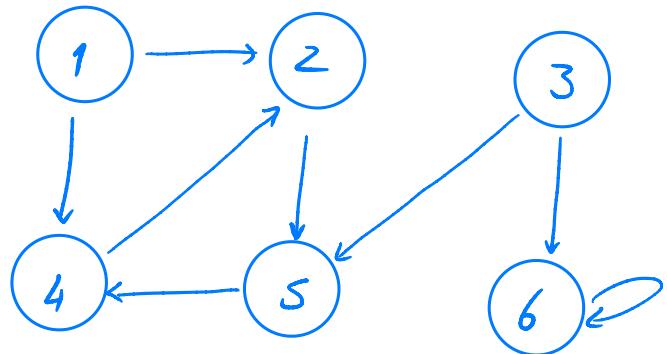
DFS (Depth-First-Search)



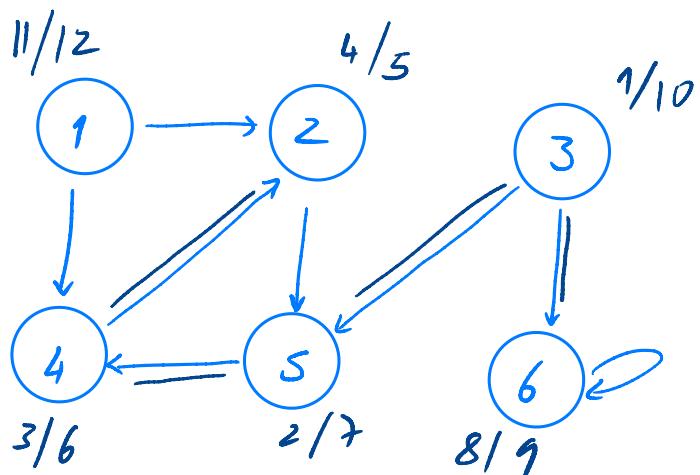
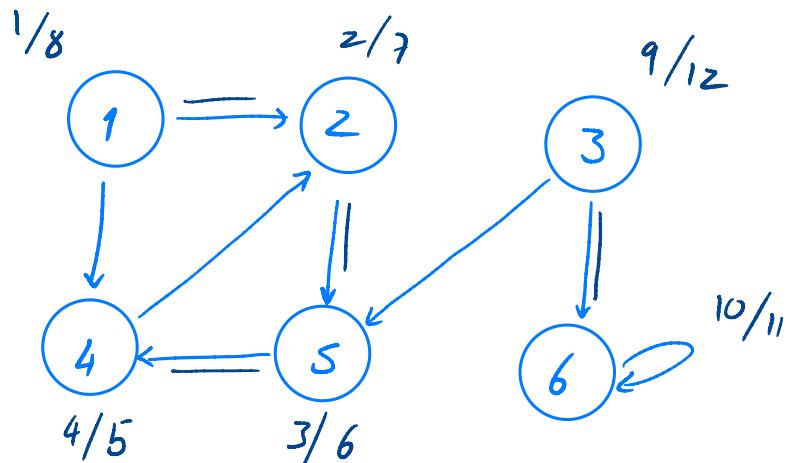
Flor deixa indicação para DFS

$$G_{\pi} = (V, E_{\pi})$$

$$E_{\pi} =$$



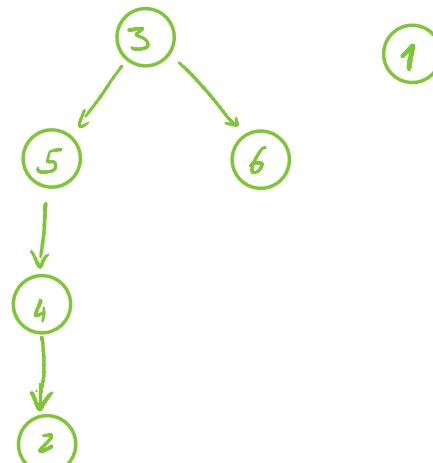
DFS (Depth-First-Search)



Floresca induzida por DFS

$$G_{\bar{\pi}} = (V, E_{\bar{\pi}})$$

$$E_{\bar{\pi}} = \left\{ (\bar{\pi}[u], u) \mid \bar{\pi}(u) \neq NIL \right\}$$



DFS (Depth-First-Search)

DFS(G)

for each $v \in G.V$

time := 0;

for each $v \in G.V$
if

} Set up: cor e parent

} Visitar cada nó

DFS-Visit($G, v, time$)

assent($v.\text{color} == \text{White}$)

for each

:

:

:

return time

} Cor e tempo de descoberta

} Visitar os vizinhos de v

} Cor e tempo de fim

DFS (Depth-First-Search)

DFS(G)

for each $v \in G.V$

$v.\text{color} := \text{White}$; $v.\pi = \text{Nil}$

$\text{time} := 0$;

for each $v \in G.V$

if $v.\text{color} == \text{White}$

$\text{time} := \text{DFS-Visit}(G, v, \text{time})$

Loop 1

DFS-Visit(G, v, time)

assert($v.\text{color} == \text{White}$)

$v.\text{color} := \text{Gray}$; $\text{time} := \text{time} + 1$; $v.d := \text{time}$;

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

if $u.\text{color} == \text{White}$

$u.\pi := v$

$\text{time} := \text{DFS-Visit}(G, u, \text{time})$

Loop 2

$v.\text{color} := \text{Black}$; $\text{time} := \text{time} + 1$; $v.f := \text{time}$;

return time

Complexidade Naif:

Complexidade:

DFS (Depth-First-Search)

DFS(G)

for each $v \in G.V$

$v.\text{color} := \text{White}$; $v.\pi = \text{Nil}$

$\text{time} := 0$;

for each $v \in G.V$

if $v.\text{color} == \text{White}$

$\text{time} := \text{DFS-Visit}(G, v, \text{time})$

Loop 1

DFS-Visit(G, v, time)

assert($v.\text{color} == \text{White}$)

$v.\text{color} := \text{Gray}$; $\text{time} := \text{time} + 1$; $v.d := \text{time}$;

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

if $u.\text{color} == \text{White}$

$u.\pi := v$

$\text{time} := \text{DFS-Visit}(G, u, \text{time})$

Loop 2

$v.\text{color} := \text{Black}$; $\text{time} := \text{time} + 1$; $v.f := \text{time}$;

return time

Complexidade Naif:

$O(|V| \cdot |E|)$

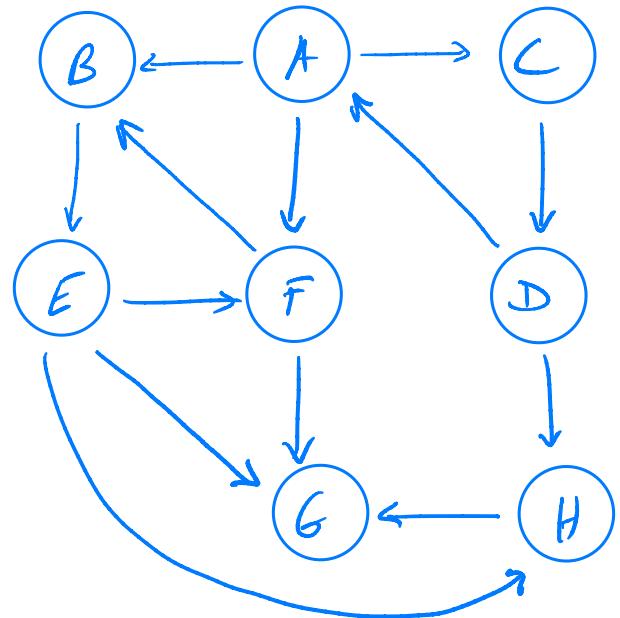
Complexidade:

- DFS-Visit é invocado uma única vez por cada vértice do grafo
↓

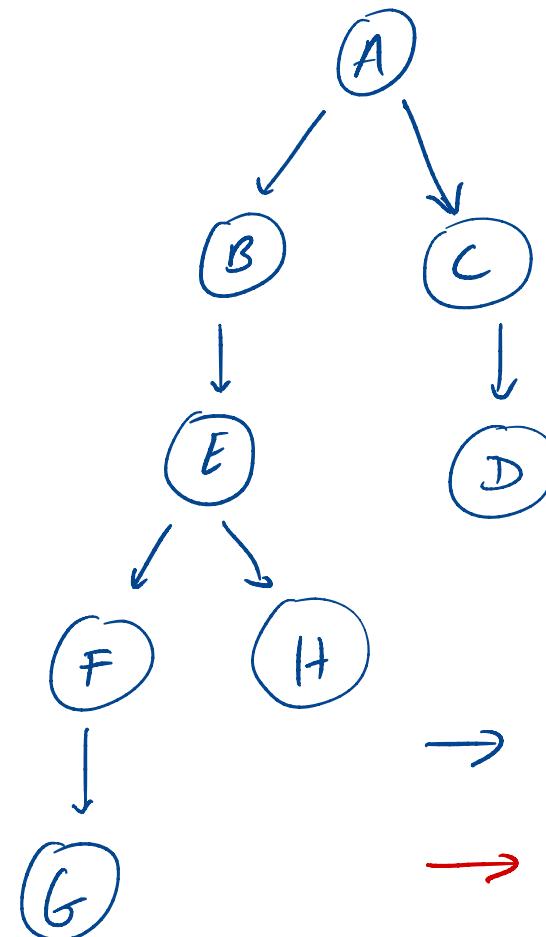
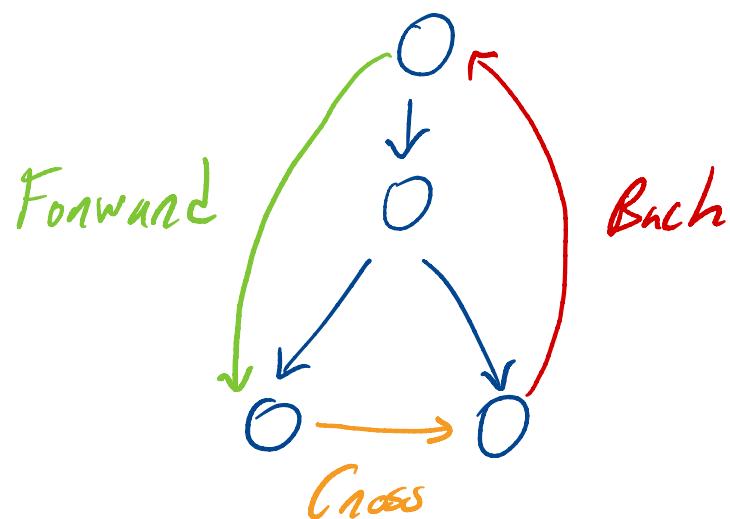
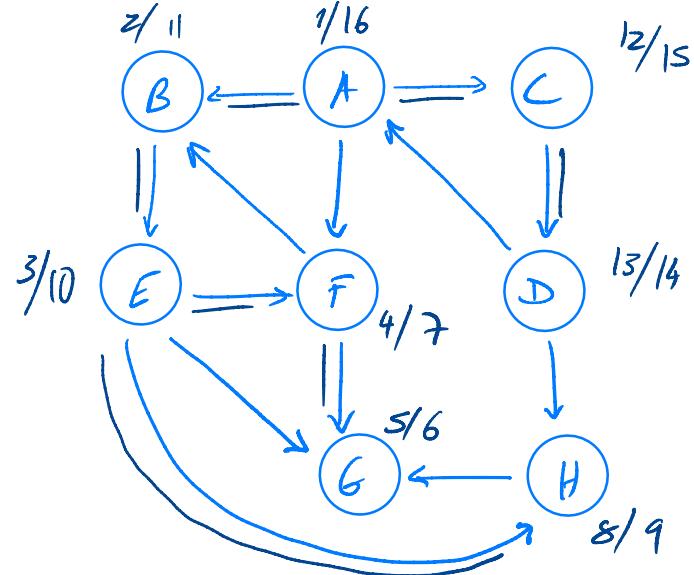
O corpo do loop 2 é executado uma vez por cada arco do grafo

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

DFS - Classificação de Arcos

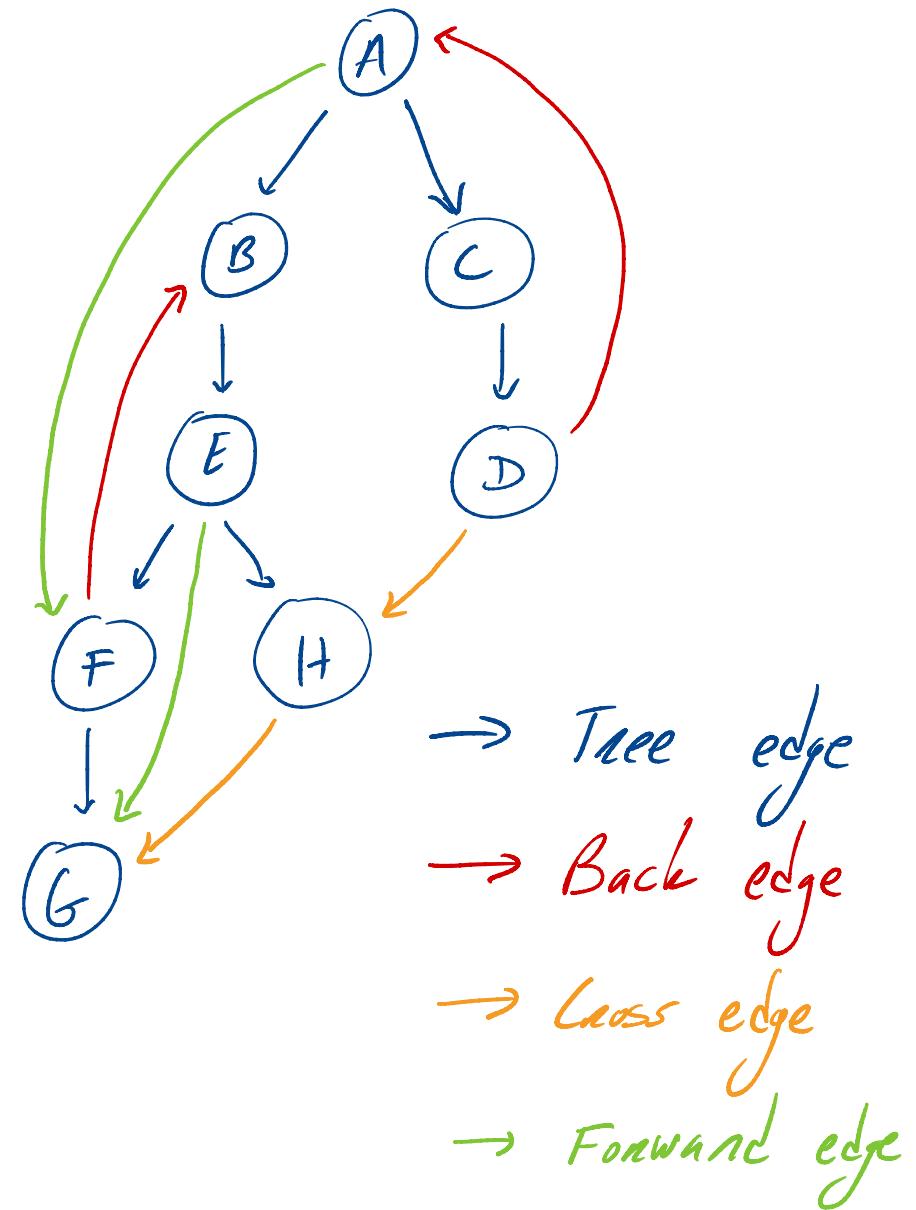
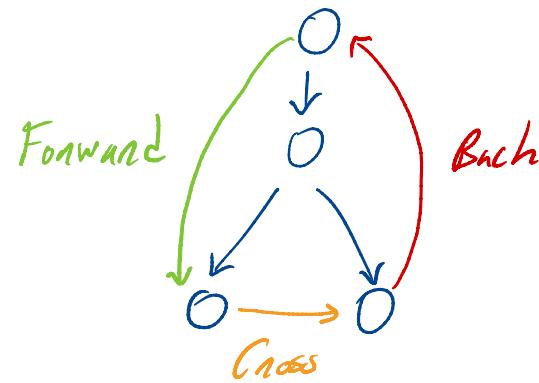
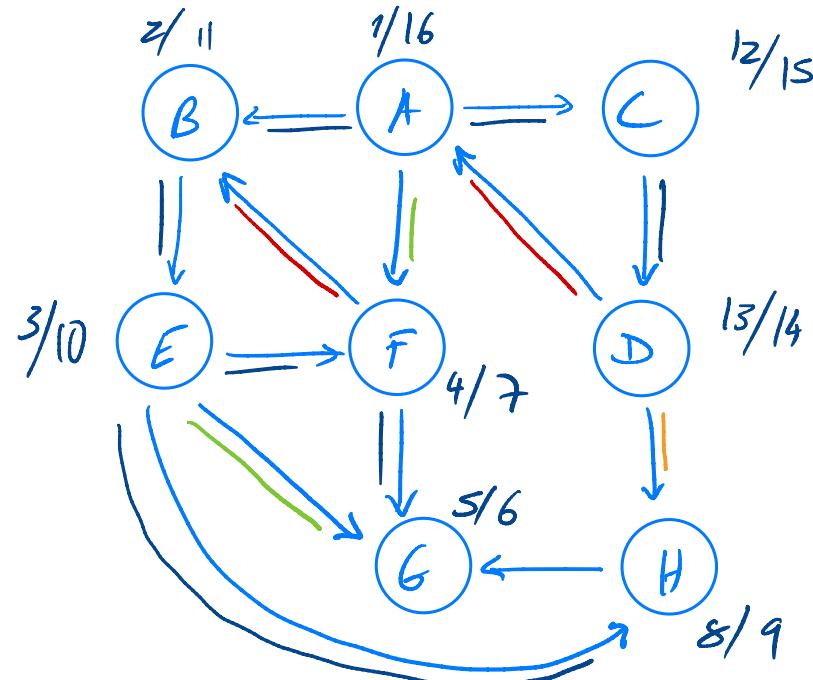


DFS - Classificação de Arcos

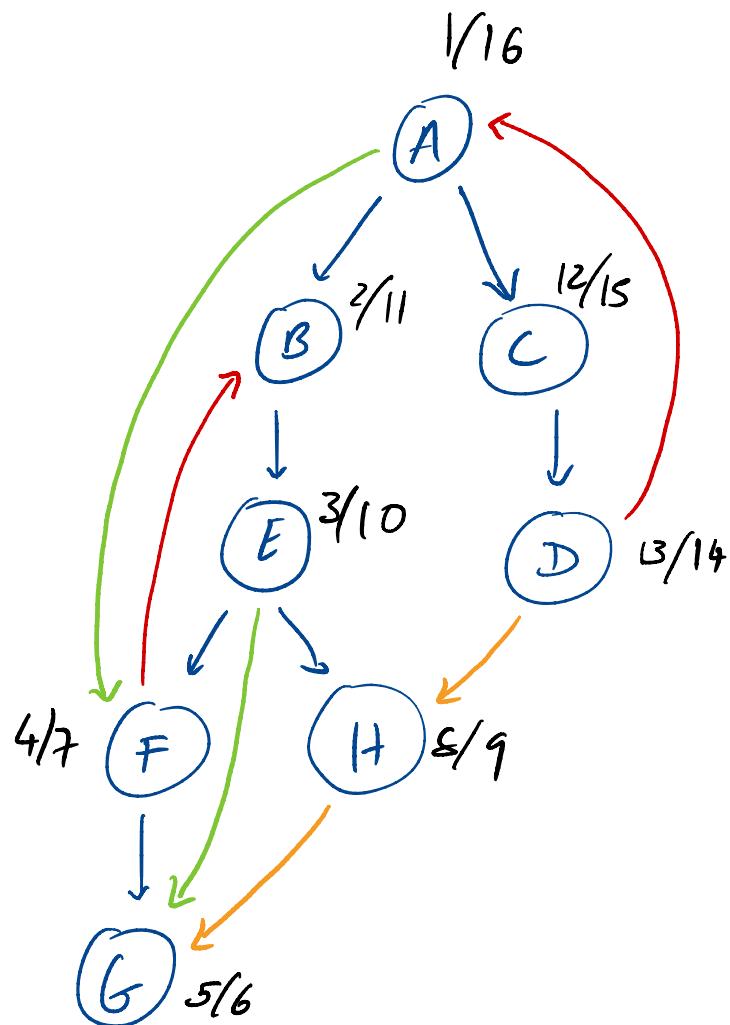


- Tree edge
- Back edge
- Cross edge
- Forward edge

DFS - Classificação de Arcos



DFS - Classificação de Arcos



→ Tree edge: (u, v)

$$[i_u \ i_v \ j_v]_u \Leftrightarrow d_u < d_v < f_v < f_u$$

→ Back edge: (u, v)

$$[i_v \ [i_u \ j_u]_v]_v \Leftrightarrow d_v < d_u < f_u < f_v$$

→ Cross edge: (u, v)

$$[i_v \ j_v \ [i_u \ j_u]_u] \Leftrightarrow d_v < f_v < d_u < f_u$$

→ Forward edge: (u, v)

$$[i_u \ i_v \ j_v \ j_u] \Leftrightarrow d_u < d_v < f_v < f_u$$

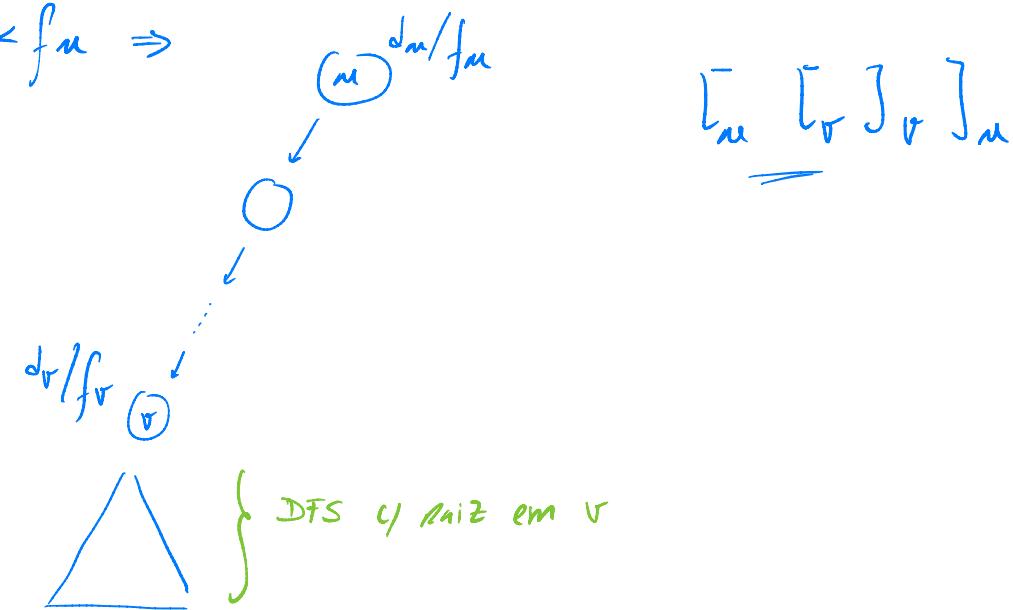
DFS - Propriedades

- Após uma DFS nunca temos: $[l_m \ l_v]_m \ J_v$

* Suponhamos que $d_m < d_v$

① $f_m < f_v \Rightarrow$ Os intervalos não se intersectam
 $[l_m \ l_m]_m \ [l_v \ l_v]_v$

② $f_v < f_m \Rightarrow$



DFS - Propriedades

Teorema do Caminho Branco

v é descendente de u na floresta DFS

Se no momento em que u é descoberto

existe um caminho branco a ligar u a v .

$\Rightarrow v$ é descendente de u

\downarrow
 \Rightarrow Existe um caminho branco entre u e v em D_u

\Leftarrow Existe um caminho branco entre u e v

$\Rightarrow v$ é descendente de u na floresta DFS

• Suponhamos q v não é descendente de u

• Admitimos s/ perda de generalidade

q v é o primeiro vértice no caminho
branco q não é descendente de u

DFS - Propriedades

Teorema do Caminho Branco

v é descendente de u na floresta DFS

Se no momento em que u é descoberto

existe um caminho branco a ligar u a v .

$\Rightarrow v$ é descendente de u

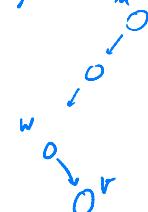
\downarrow
 \Rightarrow Existe um caminho branco entre u e v em Δ_u

$[_u \ [v]_v]_u$

\Leftarrow Existe um caminho branco entre u e v

$\Rightarrow v$ é descendente de u na floresta DFS

• Suponhamos q v não é descendente de u



• Admitimos s/ perda de generalidade q v é o primeiro vértice no caminho branco q não é descendente de u

$[_u \ [v]_v]_u$

em ambos os casos v é
descendente de u

DFS - Propriedades

Teorema do Caminho Branco

v é descendente de u na floresta DFS

Se no momento em que u é descoberto

existe um caminho branco a ligar u a v .

$\Rightarrow v$ é descendente de u

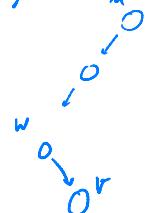
\downarrow
 \Rightarrow Existe um caminho branco entre u e v em Δ_u

$[_u \ [v]_v]_u$

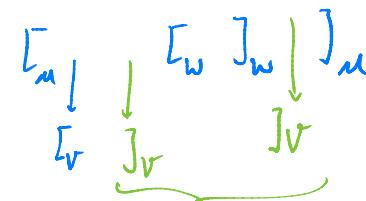
\Leftarrow Existe um caminho branco entre u e v

$\Rightarrow v$ é descendente de u na floresta DFS

• Suponhamos q v não é descendente de u



• Admitimos s/ perda de generalidade q v é o primeiro vértice no caminho branco q não é descendente de u



em ambos os casos v é descendente de u

DFS - Propriedades

- Um grafo tem um caminho circular se e só se o DFS revela um novo para trás.

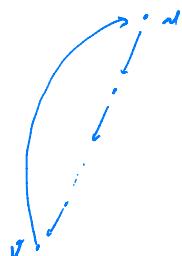
\Leftarrow Back-edge \Rightarrow Caminho circular

\Rightarrow Caminho circular \Rightarrow Back-edge

DFS - Propriedades

- Um grafo tem um caminho circular se e só se o DFS revela um arco para trás.

\Leftarrow Back-edge \Rightarrow Caminho circular



\Rightarrow Caminho circular \Rightarrow Back-edge

$\langle v_1, \dots, v_n \rangle$ caminho circular

$$v_i = v_n$$

\Downarrow re-ordenamos os vértices do caminho circular para começar no vértice com menor tempo de descoberta

$\langle v'_1, \dots, v'_n \rangle$

$$v'_1 = v'_n$$

• Qd v'_1 é descoberto existe um caminho

branco entre v'_1 e v_{n-1} \Rightarrow logo, v_{n-1} é descendente de v'_1 na árvore DFS \Rightarrow (v_{n-1}, v'_n) é arco para trás