Aula 6

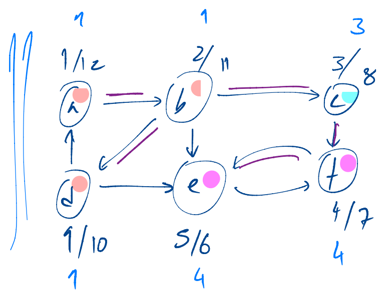
- BFS
  - Correcção
  - Complexidade
  - Exemplo: Identificar grafos bipartidos
- Algoritmo de Dijkstra
  - Correcção
  - Complexidade

Algoritmo de Tarjan

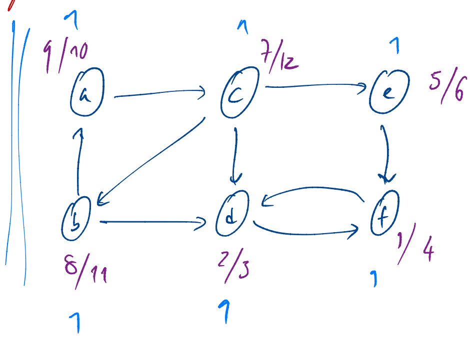
- Ideia: Usar os arcos para trás para identificar

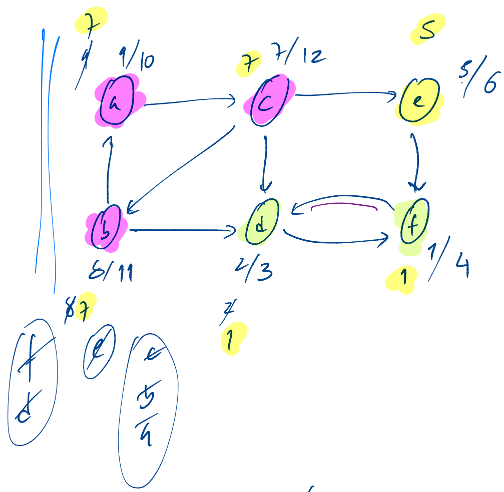
low(v) ⇒ lowest discovery time of a node accessible from v.

$$low(v) = \min \{ u.d \mid v \rightsquigarrow u \}$$

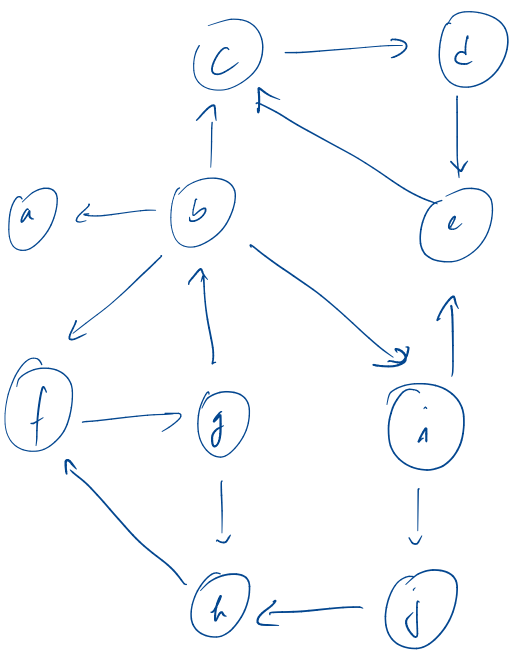


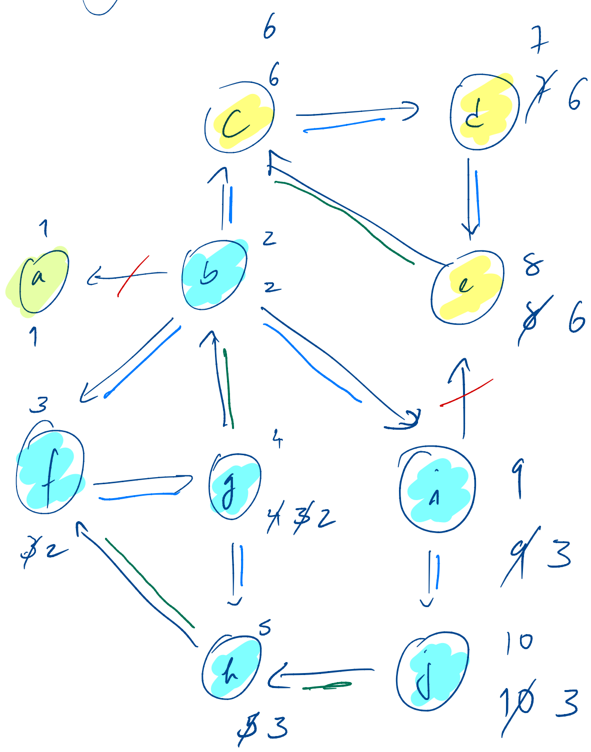
Exemplo 1:

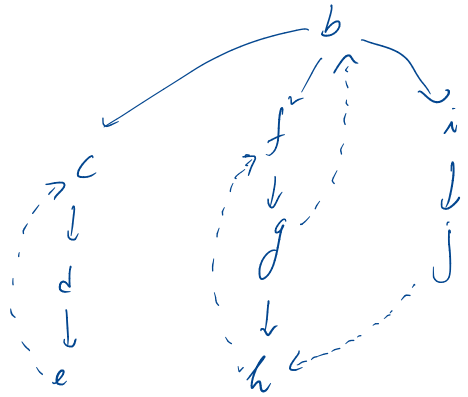




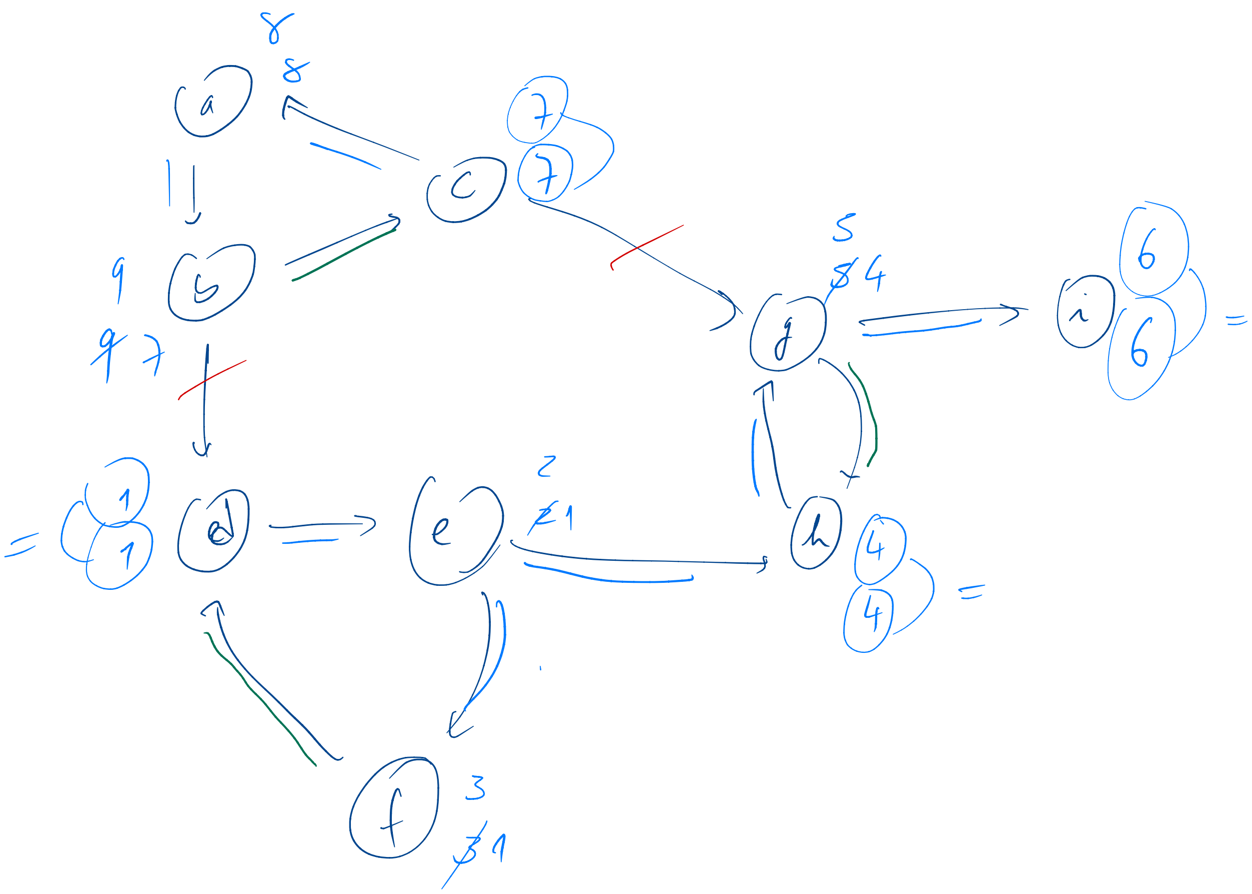
Exemplo 2:



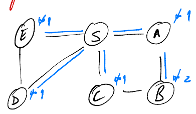


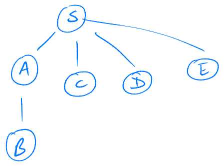


Exemplos:



Exemplo 1 [BFS]





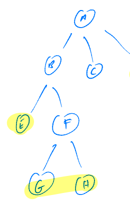
Algoritmo 1 [BFS]

```
BFS(G, s)
  for each v ∈ G.V \ {s}
  |  v.d = ∞
  |  v.color = white ;  v.π = NIL
  s.d = 0 ;  s.color = gray
  s.π = NIL
  Q = EmptyQueue();
  EnQueue(Q, s)
  while ( ! isEmpty(Q) )                      loop1
  |  u = DeQueue(Q);
  |  for each v ∈ G.Adj[u]                    loop2
  |  |  if ( v.d == white )
  |  |  |  v.π = u ;  v.d = u.d + 1 ;  v.color = gray
  |  |  |  Enqueue(Q, v)
  |  u.color = black
```

Complexidade:

* Análise agregada:
  - Cada vértice é colocado na lista fila Q no máximo uma vez. (Depois de um vértice passar a cinzento, ele nunca volta a estar branco.)
  - O corpo do loop 2 é executado $O(|E|)$ vezes.
  - O corpo do loop 1 é executado $O(|V|)$ vezes.
  - A inicialização das estruturas usadas pelo algoritmo faz-se em tempo $O(|V|)$.
  - $O(|V| + |E|)$

Comentário: Suponha que o algoritmo DFS produz a seguinte árvore:



É possível que durante a execução do algoritmo, a fila Q tenha tido o seguinte conteúdo?

- D, E, G, H

Porquê?

Definição 1 [Distância em Grafo]

$\delta_G(u,v)$ = nº de arcos do caminho mais curto entre u e v em G caso tal caminho exista; ∞ no caso contrário.

Lema 1 [Desigualdade Triangular]

$$s, u, v \in G.V \wedge (u,v) \in G.E \Rightarrow \delta_G(s,v) \leq \delta_G(s,u) + 1$$

Prova.

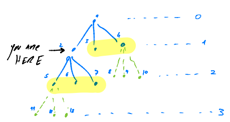
- Consideremos separadamente o caso em que $\delta_G(s,u) = \infty$ e $\delta(s,u) < \infty$
- Se $\delta(s,u) = \infty$, então $\delta(s,u) + 1 = \infty$ e a desigualdade é verificada p/ qualquer valor de $\delta(s,v)$.
- Se $\delta(s,u) < \infty$, então u é atingível a partir de s, $G \vdash s \rightsquigarrow u$. Com $(u,v) \in E$, temos que $G \vdash s \rightsquigarrow v$.
  Em particular, existe um caminho que liga s a v de tamanho $(\delta(s,u) + 1)$, pelo que:
  $$\delta(s,v) \leq \delta(s,u) + 1.$$

Lema 2 [Invariante da BFS]

Seja $\langle v_0, \dots, v_n \rangle$ os vértices na fila Q durante a BFS, então

$$v_0.d \leq v_1.d \leq \dots \leq v_n.d \quad \wedge \quad v_n.d \leq v_0.d + 1$$

→ a BFS explora os vértices fila por fila:



Prova [Esboço]:

Só começamos a colocar nós do nível n+1 na fila quando estamos a explorar nós do nível n, mas só começamos a explorar nós do nível n quando acabamos de explorar os nós do nível n-1. Assim sendo, nunca temos na fila Q ao mesmo tempo nós c/ dois ou mais níveis de distância.

2

Prova [Formal]:

- Assumimos que o invariante é cumprido e verificamos o que acontece quando se insere um novo vértice v_{n+1} na fila.
- Suponhamos que no início da iteração a fila é:

  $Q = \langle v_0, \dots, v_n \rangle$, $u = v_0$

  $\forall_{0 \le i < n}.\ v_{i+1}.d \geq v_i.d \ \wedge \ v_n.d \leq v_0.d + 1$

  - Depois de retirarmos $u = v_0$ da fila Q, obtemos a fila:

    $Q' = \langle v_1, \dots, v_n \rangle$

    $\forall_{1 \le i < n}.\ v_{i+1}.d \geq v_i.d \ \wedge \ v_n.d \leq v_0.d + 1$

    $\Downarrow$

    $v_n.d \leq v_1.d + 1$

  - Ao colocar o vértice v_{n+1} na fila, obtemos a fila:

    $Q'' = \langle v_1, \dots, v_n, v_{n+1} \rangle$

    Para que o invariante se mantenha temos de provar que $v_{n+1}.d \geq v_n.d$ e $v_{n+1}.d \leq v_1.d + 1$

  - $[v_{n+1}.d \geq v_n.d]$

    1. $v_n.d \leq v_0.d + 1$ · invariante
    2. $v_{n+1}.d = v_0.d + 1$ · BFS
    3. $v_{n+1}.d \geq v_n.d$ · (1) + (2)

  - $[v_{n+1}.d \leq v_1.d + 1]$

    1. $v_{n+1}.d = v_0.d + 1$ · BFS
    2. $v_1.d \geq v_0.d$ · IH
    3. $v_1.d + 1 \geq v_0.d + 1$ · (2)
    4. $v_1.d + 1 \geq v_{n+1}.d$ · (1) + (3) ∎

Lema 3 [Correcção BFS]

Quando BFS(G, s) termina temos que:

$\forall v \in V.\ v.d \geq \delta(s, v)$ (†)

Prova:

- Provemos que antes da execução do loop 1 a proposição (†) é verdade e que sempre que o valor de d de um nó é alterado (†) mantém-se.
- Antes da execução do loop temos que:

  $s.d = 0 \geq \delta(s,s) = 0$

  $\forall v \in V \setminus \{s\}.\ v.d = \infty \geq \delta(s, v)$

- O valor d de um nó v é alterado imediatamente antes do nó ser inserido na fila quando o seu predecessor u na árvore BFS está a ser explorado

  $v.d = u.d + 1$ (BFS)

  $\geq \delta(s, u) + 1$ (invariante)

  $\geq \delta(s, v)$ (Lema 1) ∎

Teorema 1 [Correção BFS]

O algoritmo BFS é correcto.

Prova.

Quando o algoritmo BFS(G,s) termina temos q:

$\forall v \in V.\ v.d = \delta(s, v)$

- Provamos a correcção do algoritmo por contradição. Supunhamos q existe um nó u tal que depois da execução de BFS(G,s), $u.d \neq \delta(s, u)$.
- Assumimos, sem perda de generalidade, que u é o primeiro vértice no caminho mais curto q o liga a s tal cuja distância calculada não coincide c/ a distância mínima dada por S.
- Seja w o predecessor de u no caminho mais curto que o liga a s, temos q $w.d = \delta(s, w)$.
- Quando w é explorado pela DFS, o vértice u pode ser: branco, cinzento, ou preto.

  Analizamos cada um dos 3 casos separadamente.

  [u é branco] $u.d = w.d + 1 = \delta(s,w)+1 = \delta(s,u)$

  contradição

  [u é cinzento] u já se encontra na fila Q para ser explorado.

  $u.d \leq w.d + 1 = \delta(s,w)+1 = \delta(s,u)$ (Lema 8)

  Sabemos pelo Lema 9 que: $u.d \geq \delta(s,u)$.

  Assim concluímos que:

  $\delta(s,u) \leq u.d \leq \delta(s,u)$

  De onde segue que $\delta(s,u) = u.d$. contradição.

  [u é preto] O vértice u foi retirado da fila, de onde segue pelo Lema 8:

  $u.d \leq w.d = \delta(s,w) < \delta(s,u)$

  Combinando com o Lema 9:

  $\delta(s,u) \leq u.d < \delta(s,u)$ contradição ∎

Definição 2 [Árvore BFS]

A árvore BFS resultante de uma procura BFS em $G=(V,E)$ a partir do nó s é dada pelo grafo $G_\pi = (V_\pi, E_\pi)$ onde:

$$V_\pi = \{ v \in V \mid v.\pi \neq NIL \vee v = s \}$$

$$E_\pi = \{ (\pi(v), v) \mid v \in V \wedge v.\pi \neq Nil \}$$

Problema 1 [Grafo Bipartido]

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

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

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

Como determinar se um grafo é bi-partido?

- A ideia é alternar a BFS de forma a associar uma de duas ao vértice cada vez que ele é encontrado.
- O grafo é bi-partido se não forem encontrados arcos que ligam vértices da mesma cor.