MC558 2020s1

Teste 2

Tiago de Paula Alves
187679

---

1. Modifique o pseudo-código do algoritmo de busca em profundidade apresentado em aula ou do CLRS (supondo que o grafo de entrada G é orientado) para imprimir cada aresta (u,v) juntamente com seu tipo (aresta da árvore, de avanço, de retorno ou de cruzamento). A complexidade do DFS modificado ainda dever ser $O(V+E)$.

```
DFS(G)
para cada u ∈ V[G] faça
    cor[u] ← BRANCO
    π[u] ← NIL
tempo ← 0
para cada u ∈ V[G] faça
    se cor[u] = BRANCO
        então DFS-VISIT(u)
```

```
DFS-VISIT(u)
cor[u] ← CINZA
tempo ← tempo + 1
d[u] ← tempo
para cada v ∈ Adj[u]
    se cor[v] = BRANCO então
    // próximo vértice a ser visitado, então uv está na floresta BP
        IMPRIME(u, v, "aresta da árvore")

        π[v] ← u
        DFS-VISIT(v)
    senão, se cor[v] = CINZA então
    // v ainda não terminou, então ele é algum ancestral de u
        IMPRIME(u, v, "aresta de retorno")
    senão, se d[u] < d[v] então
    // v já foi visitado, mas antes de terminar u, então ainda é descendente de u
        IMPRIME(u, v, "aresta de avanço")
    senão
        IMPRIME(u, v, "aresta de cruzamento")
cor[u] ← PRETO
tempo ← tempo + 1
f[u] ← tempo
```

---

2. Seja G um grafo orientado acíclico. Suponha que cada aresta $(u,v) \in E[G]$ tem uma cor $cor(u,v)$ que pode ser azul ou vermelha. Um caminho P em G é válido se não possui arestas consecutivas de cor vermelha. [...]

Nesta questão, você deve projetar um algoritmo linear que para cada vértice $u \in V[G]$, devolve o número de caminhos válidos que começam em u.

Definição. Defina azul$[u]$ (respectivamente, verm$[u]$) como o número de caminhos válidos com início em u cuja primeira aresta tem cor azul (respectivamente, vermelha). Note que o caminho trivial válido (u) não contribui para nenhum desses valores.

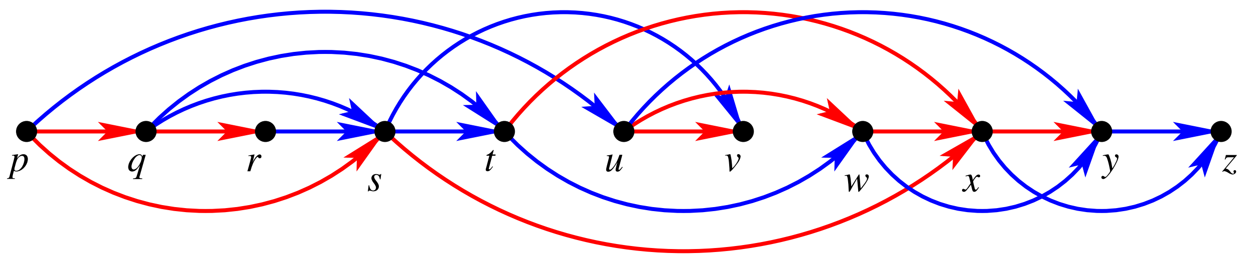

Figura 1

a) Para cada vértice i do grafo acima, indique os valores azul$[i]$ e verm$[i]$ (alguns valores estão preenchidos).

i	p	q	r	s	t	u	v	w	x	y	z
azul$[i]$	7	20	12	9	5	2	0	2	1	1	0
verm$[i]$	31	13	0	2	2	4	0	2	2	0	0

b1) Descreva uma recorrência que relaciona azul$[u]$ em função de azul$[v]$ e verm$[v]$ para $v \in \text{Adj}[u]$.

Se $cor(u,v) = \text{azul}$, então o caminho (u,v) é válido e começa com aresta azul. Além disso, para qualquer caminho válido $(v,\ldots)$ partindo de v, podemos juntar u fazendo o caminho $(u,v,\ldots)$, que continua válido e começa com aresta azul. Como v tem azul$[v]+$ verm$[v]$ caminhos válidos coloridos, podemos encontrar azul$[u]$ por:

$$\text{azul}[u] = \sum_{v \in \text{Adj}[u]} \begin{cases} 1 + \text{azul}[v] + \text{verm}[v] & \text{se } \text{cor}(u,v) = \text{azul} \\ 0 & \text{caso contrário} \end{cases}$$

---

b2) Descreva agora uma recorrência que relaciona verm$[u]$ em função de azul$[v]$ e verm$[v]$ para $v \in \text{Adj}[u]$.

Para $cor(u,v) = \text{vermelho}$, o caminho (u,v) também é válido, já que só tem uma aresta vermelha. No entanto, os únicos caminhos de v que continham válidos depois de prefixar u são os que começam com aresta azul, senão teriam duas arestas vermelhas seguidas. Assim:

$$\text{verm}[u] = \sum_{v \in \text{Adj}[u]} \begin{cases} 1 + \text{azul}[v] & \text{se } \text{cor}(u,v) = \text{vermelho} \\ 0 & \text{caso contrário} \end{cases}$$

---

c) Escreva um pseudo-código de um algoritmo de complexidade $O(V+E)$ que recebe um grafo orientado acíclico G representado por listas de adjacências e um vetor **cor** de cores e devolve um vetor $val[\,]$ indexado por V tal que $val[u]$ é o número de caminhos válidos que começam em u para cada $u \in V[G]$.

```
CAMINHOS-VÁLIDOS(G, cor)
sejam azul[ ] e verm[ ] vetores indexados pelos vértices em V[G]
para u ∈ V[G] faça
      azul[u] ← NIL            // usado para marcar vértices não visitados

seja val[ ] um vetor também indexado por V[G]
para u ∈ V[G] faça
      se azul[u] = NIL então
            CAMINHOS-AZUIS-VERMELHOS(cor, u, azul, verm)

      val[u] = 1 + azul[u] + verm[u]
      // caminho trivial (u), mais os válidos começando com azul ou vermelho
retorna val
```

```
CAMINHOS-AZUIS-VERMELHOS(cor, u, azul, verm)
// função que preenche azul[u] e verm[u] pelas relações do item 2b

azul ← 0
verm ← 0
para v ∈ Adj[u] faça
      se azul[v] = NIL então
            CAMINHOS-AZUIS-VERMELHOS(cor, v, azul, verm)

      se cor[u, v] = AZUL                    // do item 2b
            então azul ← azul + 1 + azul[v] + verm[v]
            senão verm ← verm + 1 + azul[v]

// salva os valores
azul[v] ← azul
verm[v] ← verm
```

---

d) Justifique a complexidade do seu algoritmo do item (c).

O procedimento CAMINHOS-AZUIS-VERMELHOS é chamado no máximo uma vez por vértice $u \in V[G]$. Em cada chamada, o laço das linhas 5 a 11 executa $|\,\mathrm{Adj}[u]\,|$ vezes. Então, os vetores azul[ ] e verm[ ] são preenchidos com tempo total

$$\sum_{u \in V} O\left(|\,\mathrm{Adj}[u]\,|\right) = O\left(\sum_{u \in V} |\,\mathrm{Adj}[u]\,|\right) = O(E)$$

Além disso, CAMINHOS-VÁLIDOS tem os laços das linhas 2 e 6, que executam uma vez por nó, ou seja, em tempo $O(V)$. A complexidade total deve ser, então, $O(V) + O(E) = O(V + E)$.
