INSTITUTO SUPERIOR TÉCNICO

Análise e Síntese de Algoritmos

Ano Lectivo 2017/2018 Repescagem 1º Teste - versão A

RESOLUÇÃO

I. (2,5 + 2,5 + 2,5 + 2,5 + 2,5 + 2,5 = 15,0 val.)

I.a)

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
$rank[x_i]$	1	0	3	0	1	0	1	0	2	0
$p[x_i]$	x_9	x_9	x_3	x_3	x_3	x_5	x_3	x_3	x_3	x_9

I.b)

	A	B	C	D	E	F	G	H	I	J
d/low	1 / 1	6 / 6	8 / 5	2 / 1	3 / 1	4 / 2	5 / 5	7 / 5	9 / 7	10 / 7

$SSCs$:	$\{B\}$	$\{C,G,H,I,J\}$	$\{A,D,E,F\}$	

I.c)

Ordem arcos	(E,G)	(H,I)	(A,B)	(C,E)	(D,G)	(A,E)	(E,I)	(E,F)
Custo MST	18							
Nº MST	2							

I.d)

	A	B	C	D	E	F	G	H
$h()$	-4	0	-1	-6	-8	-2	0	-10

$\widehat{w}(A,B)$	$\widehat{w}(B,E)$	$\widehat{w}(C,E)$	$\widehat{w}(E,G)$	$\widehat{w}(G,H)$
5	6	16	1	19

I.e)

Expressão	$T(n) = 2 * T(n/2) + O(lg(n))$
Majorante	$O(n)$

I.f)

	A	B	C	D
$h()$	7	8	7	8

Corte :	$\{s,A,B,C,D\}/\{t\}$	$f(S,T) =$	10

II. (2,0 + 3,0 = 5,0 val.)

II.a) $\langle$ XXX $\rangle$ **II.b)** $\langle$ XXX $\rangle$

I. (2,5 + 2,5 + 2,5 + 2,5 + 2,5 + 2,5 = 15,0 val.)

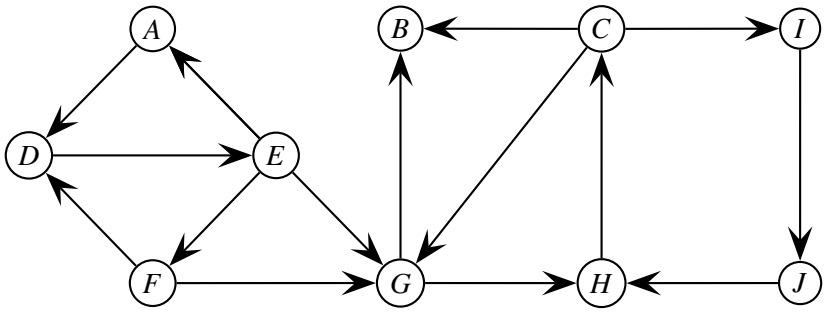
I.a) Considere o seguinte conjunto de operações sobre conjuntos disjuntos:

```
1  for i = 1 to 10 do
2      Make-Set (x_i)
3  for i = 1 to 5 do
4      Union (x_2i, x_2i−1)
5  Union (x_1, x_10)
6  j = Find-Set(x_2)
7  k = Find-Set(x_7)
8  Union (x_7, x_4)
9  Union (j, x_8)
10 Union (x_6, k)
```

Use a estrutura em árvore para representação de conjuntos disjuntos com a aplicação das heurísticas de união por categoria e compressão de caminhos. Para cada elemento x_i, indique os valores de categoria ($rank[x_i]$) e o valor do seu pai na árvore ($p[x_i]$).

Nota: Na operação Make-Set(x), o valor da categoria de x é inicializado a 0. Na operação de Union(x, y), em caso de empate, considere que o representante de y é que fica na raíz.

I.b) Considere o grafo dirigido:



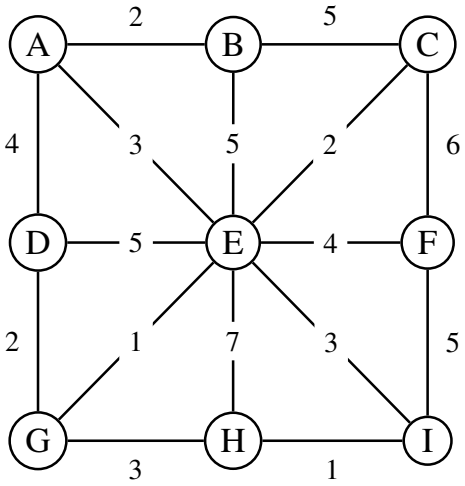
Aplique o algoritmo de Tarjan para encontrar os componentes fortemente ligados do grafo. Os vértices são sempre considerados por ordem lexicográfica (ou seja, A, B, C...). Os adjacentes também são sempre considerados por ordem lexicográfica.

Para cada vértice indique os valores de descoberta d e low após a aplicação do algoritmo.

Indique os componentes fortemente ligados **pela ordem que são descobertos pelo algoritmo**.

Nota: Neste algoritmo os valores de d começam em 1.

I.c) Considere o grafo não dirigido e pesado da figura.

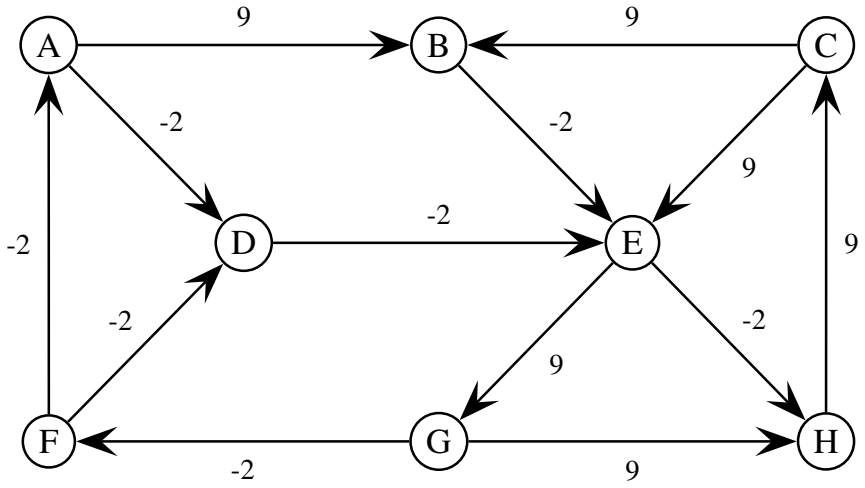


Aplique o algoritmo de Kruskal ao grafo.

Indique a ordem pela qual os arcos são seleccionados para pertencer à árvore abrangente de menor custo (MST). Em caso de empate, considere os vértices por ordem lexicográfica.

Indique o custo de uma MST e quantas MST diferentes existem no grafo.

I.d) Considere o grafo dirigido e pesado da figura.



Considere a aplicação do algoritmo de Johnson ao grafo. Calcule os valores de $h(u)$ para todos os vértices $u \in V$ do grafo.

Indique, os pesos dos seguintes arcos após a repesagem: (A,B), (B,E), (C,E), (E,G), (G,H).

I.e) Considere a função recursiva:

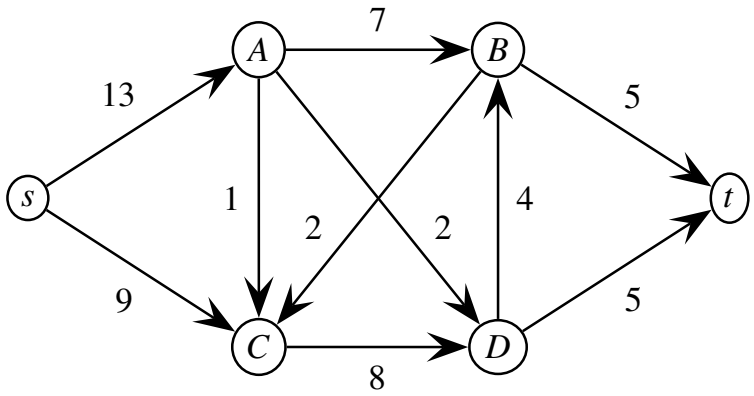
```
int f(int n) {
  int i = n*n, j = 0;

  while(i > 0) {
    i = i / 2;
    j++;
  }

  if(n > 1)
    i = i * f(n/2) + f(n/2);

  while (j > 0) {
    i++;
    j--;
  }
  return i;
}
```

Indique a expressão (recursiva) que descreve o tempo de execução da função em termos do número n, e de seguida, utilizando os métodos que conhece, determine o menor majorante assimptótico.

I.f) Considere a rede de fluxo da figura onde s e t são respectivamente os vértices fonte e destino na rede.

Aplique o algoritmo Relabel-To-Front na rede de fluxo. Considere que a lista de vértices é inicializada por ordem alfabética e que os vizinhos de cada vértice também estão ordenados alfabeticamente. Assim, as listas de vizinhos dos vértices intermédios são as seguintes:

$N[A] = < B, C, D, s >$ $N[B] = < A, C, D, t >$ $N[C] = < A, B, D, s >$ $N[D] = < A, B, C, t >$



Indique a altura final de cada vértice. Indique ainda o corte mínimo da rede e o valor do fluxo máximo.

II. (2,0 + 3,0 = 5,0 val.)

II.a) Considere uma rede de fluxo $G = (V, E)$. Suponha que foi calculada uma função de fluxo $f : E \to N$ que define o fluxo máximo na rede de fluxo G.

Considere que é adicionado um novo arco (u, v) à rede de fluxo com capacidade k. Ou seja, temos uma nova rede $G' = (V, E \cup \{(u, v)\})$ onde $u, v \in V$.

Proponha um algoritmo que calcule o fluxo máximo da rede G', actualizando a função de fluxo f calculada anteriormente para G.

Indique a complexidade do algoritmo proposto.

Solução:

Ao adicionar um novo arco (u, v), o valor do fluxo poderá aumentar se (u, v) aumentar a capacidade do corte mínimo. Se o arco não atravessar o corte mínimo de G, então o valor do fluxo mantém-se porque não existe nenhum caminho de s para t na rede residual.

Supondo que (u, v) atravessa o corte mínimo de G, então todos os novos caminhos de aumento de s para t precisam passar por (u, v). Caso contrário, o fluxo máximo ainda não teria sido calculado para G. Desta forma, o aumento do valor do fluxo é limitado pela capcidade do arco (u, v). Se usarmos o algoritmo de Ford-Fulkerson, então a actualização da rede pode ser feita em $O(E * c(u, v))$.

II.b) Considere um grafo $G = (V, E)$ dirigido e sem pesos. Um vértice $u \in V$ é denominado como vértice raíz se para todos os outros vértices $v \in V$ existe um caminho de u para v. Ou seja, todos os vértices do grafo são atingíveis a partir de um vértice raíz.

Defina um algoritmo eficiente que permite identificar **todos** os vértices raíz de um grafo G. Note que o grafo G pode não ter um vértice raíz, pelo que o seu algoritmo deve identificar essa situação.

Indique a complexidade da solução proposta.

Solução:

A solução deste problema é semelhante a identificar se o grafo G é semi-ligado.

Suponha que aplica o algoritmo de Tarjan para identificar os componentes fortemente ligados (SCC) do grafo G.

Seja u o último vértice a ser fechado pelo algoritmo. Se existir um vértice raíz em G, então u é vértice raíz porque pertence ao último SCC identificado e este SCC não tem arcos de entrada. Caso contrário, não seria o último SCC a ser identificado.

Para testar se u é vértice raíz, basta verificar se todos os outros vértices do grafo são atingíveis a partir de u. Isto pode ser feito usando uma DFS. Se u não for vértice raíz, então o grafo não tem vértice raíz. Caso contrário, todos os vértices do mesmo SCC de u são vértices raíz.

A complexidade deste algoritmo é $O(V + E)$ devido à aplicação do algoritmo de Tarjan e DFS.