INSTITUTO SUPERIOR TÉCNICO

Análise e Síntese de Algoritmos

Ano Lectivo 2018/2019 — 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_2	x_3	x_8	x_{11}	x_{12}
$rank[x_i]$	2	0	0	1	0
$p[x_i]$	x_1	x_1	x_1	x_2	x_{11}

I.b)

	A	B	C	D	E	F	G	H	I
$d()$	0	1	2	5	4	3	6	7	8
$low()$	0	0	0	0	0	0	4	4	4

Nº Componentes Fortemente Ligados:	1

I.c)

Ordem vértices	1	2	3	4	5	6
v	A	B	D	E	C	F

Caminho Mais Curto Errado:	A,B,D,F

I.d)

Custo MST	$6X+Y$
Número MST Diferentes	9

I.e)

Expressão	$T(n) = 4 * T(n/2) + O(lg\ n)$
Majorante	$O(n^2)$

I.f)

	A	B	C	D	E
$h()$	1	1	8	2	9

Corte :	$\{s,C,D,E\}/\{A,B,t\}$ **ou** $\{s,C,E\}/\{A,B,D,t\}$	$f(S,T) =$	14

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.)

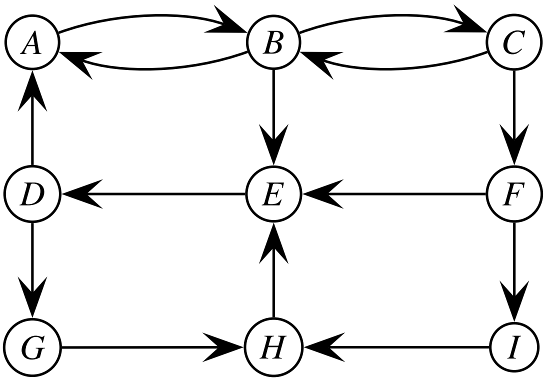
I.a) Considere o seguinte conjunto de operações sobre conjuntos disjuntos:

for i = *1 to 12* **do**
    Make-Set (x_i)
Union (x_6, x_5)
Union (x_4, x_1)
Union (x_3, x_2)
Union (x_6, x_1)
Union (x_8, x_7)
Union (x_6, x_7)
Union (x_9, x_8)
Union (x_3, x_{10})
Union (x_{12}, x_{11})
Union (x_{12}, x_{10})
Union (x_{11}, x_7)
return Find-Set(x_3)

Use a estrutura em árvore para representação de conjuntos disjuntos com a aplicação das heuristicas de união por categoria e compressão de caminhos. Para os elementos $x_2, x_3, x_8, x_{11}, x_{12}$, indique os valores de categoria ($rank[x_i]$) e o valor do seu pai na árvore ($p[x_i]$).

Nota: Na operação $Make\text{-}Set(x)$, 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.

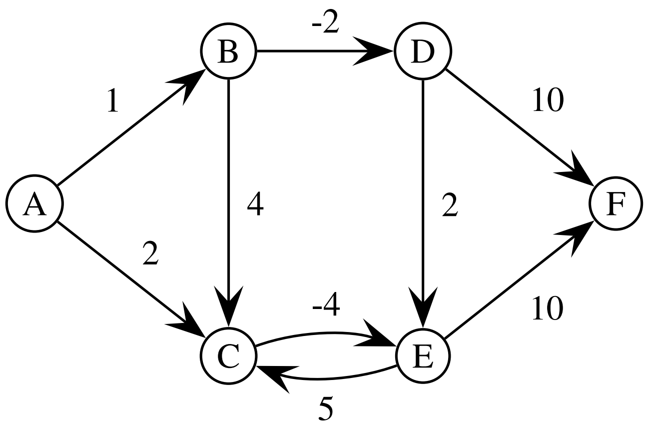
I.b) Considere o seguinte grafo dirigido.



Aplique o algoritmo de Tarjan ao grafo da figura. Inicie a travessia no vértice A. Durante a aplicação do algoritmo considere que os vértices são sempre analisados por ordem lexicográfica (ou seja, A, B, C...).

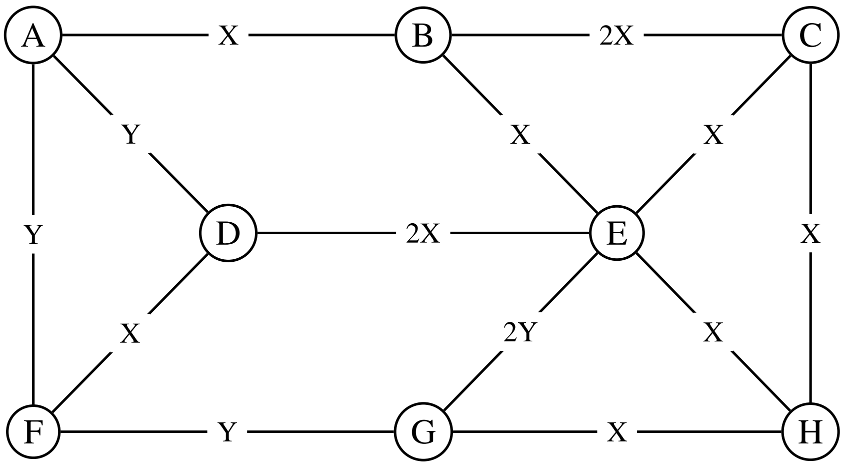
Indique os valores de d e low para todos os vértices do grafo. Considere que os valores de d e low começam em 0. Indique ainda o número de componentes fortemente ligados no grafo.

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



Aplique o algoritmo de Dijkstra ao grafo, considerando o vértice A como origem. Indique a ordem pela qual os vértices são removidos da fila de prioridade durante a execução do algoritmo. Em caso de empate, considere os vértices por ordem lexicográfica.
Como existem arcos com pesos negativos, é possível que o algoritmo não produza resultados correctos. Caso haja algum caminho calculado que não seja um caminho mais curto, indique o caminho incorrectamente calculado desde a origem ao destino. Caso contrário, indique "Nenhum" na sua resposta.

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



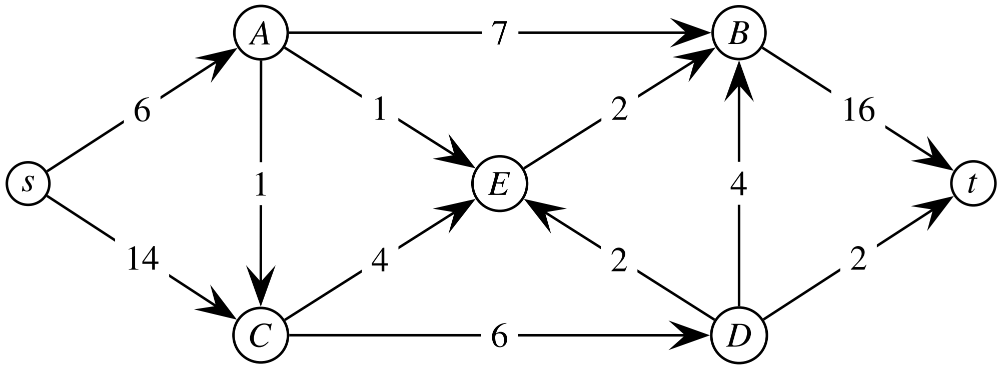
Sabendo que $2X > Y > X > 0$, calcule o peso de uma árvore abrangente de menor custo (MST) do grafo. Indique o número de árvore abrangentes de menor custo diferentes que podemos formar no grafo.

I.e) Considere a função recursiva:

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

  if (n <= 1) return 1;

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

  for (int k = 0; k < 4; k++)
    j += f(n/2);

  while (i > 0) {
    j = j + 2;
    i--;
  }
  return j;
}
```

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

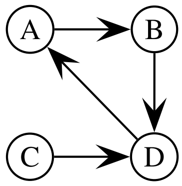


Indique a altura final de cada vértice. Indique ainda o corte mínimo da rede e o valor do fluxo máximo.

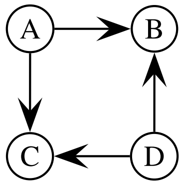
II. (2,0 + 3,0 = 5,0 val.)

II.a) Considere um grafo $G = (V, E)$ dirigido. Um grafo diz-se semi-ligado se entre quaisquer dois vértices $u, v \in V$ existe um caminho de u para v **ou** existe um caminho de v para u.

Por exemplo, o grafo da figura abaixo é semi-ligado.



Por outro lado, o grafo seguinte não é semi-ligado porque entre os vértices A e D não existe nenhum caminho em qualquer dos sentidos.



Proponha um algoritmo eficiente que recebe um grafo e determina se o grafo é semi-ligado. Indique a complexidade do algoritmo proposto.

Solução:

II.b) Considere um conjunto D de disciplinas com dimensão m. Cada disciplina $d_i \in D$ tem um número limitado de inscrições. Seja $\lim(d_i)$ o número máximo de alunos que se pode inscrever à disciplina d_i ($1 \leq i \leq m$).

Seja A um conjunto de alunos com dimensão n. Cada aluno tem que se inscrever a 5 disciplinas. No entanto, como há limites nas inscrições, os alunos podem candidatar-se a um número superior a 5 disciplinas. Seja $cand(a_j)$ o conjunto de disciplinas a que o aluno a_j ($1 \leq j \leq n$) se candidata a frequentar. Pode assumir que o número de disciplinas a que cada aluno se candidata é pelo menos 5, ou seja, $|cand(a_j)| \geq 5$.

Proponha um algoritmo eficiente que dado um conjunto D de disciplinas (onde cada disciplina tem um número limitado de inscrições) e um conjunto de alunos A (onde cada aluno se candidata a pelo menos 5 disciplinas), determina uma possível atribuição dos alunos às disciplinas de tal forma que cada aluno está inscrito precisamente a 5 disciplinas. Note que cada aluno a_j apenas pode ser inscrito nas disciplinas a que se cadidata ($cand(a_j)$). Caso não seja possível inscrever cada aluno a 5 disciplinas, o seu algoritmo deve retornar "Impossível".

Indique a complexidade do algoritmo proposto.

Solução: