INSTITUTO SUPERIOR TÉCNICO

Análise e Síntese de Algoritmos

Ano Lectivo 2019/2020 1º Teste

RESOLUÇÃO

I. (2,0 + 2,0 + 2,0 + 2,0 + 2,0 = 10,0 val.)

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

---

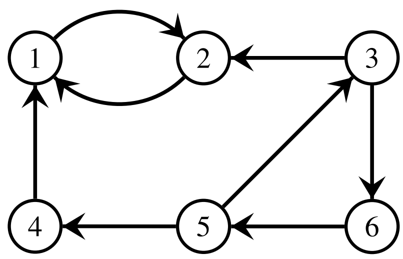
for $i = 0$ *to* 9 **do**
  Make-Set (x_i)
for $i = 0$ *to* 2 **do**
  Union ($x_{2^i}, x_{2^{i+1}}$)
for $i = 0$ *to* 2 **do**
  Union ($x_{3*i}, x_{3*(i+1)}$)
Union (x_5, x_7)
Union (x_1 , x_6)
Union (x_1 , x_5)
return Find-Set(x_4)

Use a estrutura em árvore para representação de conjuntos disjuntos com a aplicação das heuristicas de união por categoria e compressão de caminhos. Para cada elemento x_i ($0 \leq i \leq 9$) indique os valores de categoria ($rank[x_i]$) e o valor do seu pai na árvore que representa os conjuntos ($p[x_i]$).

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.

	x_0	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9
$rank[x_i]$	0	0	1	2	0	0	0	1	0	0
$p[x_i]$	x_3	x_3	x_3	x_3	x_3	x_7	x_3	x_3	x_2	x_3

I.b)



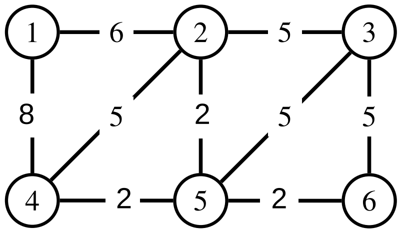
Considere o grafo dirigido e acíclico da figura. Aplique o algoritmo de Tarjan para identificar os componentes fortemente ligados, considerando o vértice 5comoinicial.
Durante a aplicação do algoritmo considere que tanto a escolha dos vértices a visitar, como a pesquisa dos vértices adjacentes são feitas por ordem lexicográfica (ou seja, 1, 2, 3, ...).

Indique os componentes fortemente ligados pela ordem segundo a qual são identificados pelo algoritmo e o valor *low* calculado para cada vértice.

	1	2	3	4	5	6
$low()$	3	3	1	6	1	1

SCCs :	1 , 2	4	3 , 5 , 6

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



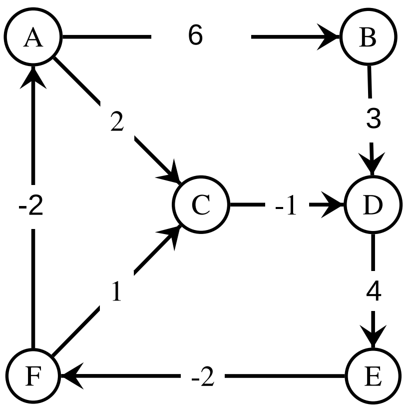
Aplique o algoritmo de Prim ao grafo, considerando o vértice 1 como origem.

Para cada vértice, indique qual o valor da sua chave (k) e o de seu antecessor (π), quando o vértice é removido da fila de prioridade. Em caso de empate, considere os vértices por ordem lexicográfica.
Indique ainda o peso da árvore abrangente de menor custo encontrada.

Ordem vértices	1	2	6	4	3	5
v	1	2	3	4	5	6
$key[v]$	0	6	5	2	2	2
$\pi[v]$	NIL	1	2	5	2	5

Peso MST:	17

I.d) Considere a aplicação do algoritmo de Johnson ao grafo dirigido e pesado da figura.

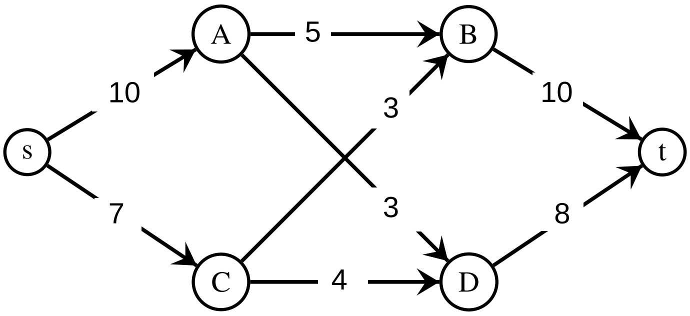


Calcule os valores de $h(u)$ para todos os vértices $u \in V$ do grafo. Calcule também os pesos de todos os arcos após a repesagem.

	A	B	C	D	E	F
$h()$	-4	0	-2	-3	0	-2

$\widehat{w}(A,B)$	$\widehat{w}(F,A)$	$\widehat{w}(A,C)$	$\widehat{w}(C,D)$	$\widehat{w}(B,D)$	$\widehat{w}(D,E)$	$\widehat{w}(F,C)$	$\widehat{w}(E,F)$
2	0	0	0	6	1	1	0

I.e) 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 que as listas de vizinhos dos vértices intermédios são as seguintes: $N[A] = < B, D, s >$ $N[B] = < t, C, A >$ $N[C] = < B, D, s >$ $N[D] = < t, A, C >$, e que a lista de vértices inicial é $L = < A, B, C, D >$.



Indique a altura final de cada vértice. Indique ainda um corte mínimo da rede, o valor do fluxo máximo, e a sequência de <u>diferentes</u> configurações de L.

	s	A	B	C	D	t
$h()$	6	7	1	2	1	0
Corte :	$\{s, A, C\}/\{B, D, t\}$			f(S,T) = 15		
L:	$< A, B, C, D >$		$< B, A, C, D >$		$< C, B, A, D >$	$< D, C, B, A >$

II. (2,0 val.) Considere a função recursiva:

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

  for (i = 0; i < n; i++) { // Loop 1
    while (j - i < 2) {
      j++;
    }
  }

  if (n > 0)
    i = 2*f(n/2) + f(n/2) + f(n/2)

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

  return j;
}
```

II.a) Determine um upper bound medido em função do parâmetro n para o número de iterações dos loops **1** e **2** da função f.

II.b) Determine o menor majorante assimptótico da função f em termos do número n utilizando os métodos que conhece.

Expressão	$T(n) = 3 * T(n/2) + O(n)$
Majorante	$O(n^{(lg_2 3)}$

III. (1,5 + 1,5 = 3,0 val.)

O Presidente da Câmara de Caracolândia declarou na última conferência de imprensa que os habitantes de Caracolândia conseguem deslocar-se entre quaisquer dois cruzamentos da cidade sem sair da cidade. A oposição não está completamente convencida.

III.a) Proponha um algoritmo eficiente para verificar a veracidade da afirmação do Presidente da Câmara, admitindo que há estradas de sentido único em Caracolândia. Indique a complexidade do algoritmo.

Solução: Determinar o grafo $G = (V, E)$ onde os vértices V correspondem aos cruzamentos de Caracolândia e, dados dois vértices $u, v \in V$, $(u, v) \in E$ se e apenas se existe uma estrada a ligar os vértices u e v. Usar o algoritmo de Tarjan ou o algoritmo de Kosaraju's para identificar os componentes fortemente ligados de G. A afirmação é verdadeira se o número de componentes fortemente ligados for 1.

A complexidade do algoritmo proposto é dominada pela complexidade do algoritmo para identificar os SCCs de G, $O(V + E)$.

III.b) Verificou-se que afinal a afirmação original do Presidente da Câmara não é verdadeira: existem pares de cruzamentos que não cumprem o critério definido. O Presidente da Câmara restringiu então a asserção original; afirma agora que: quem chegar a um cruzamento a partir da câmara municipal, consegue regressar à câmara municipal sem sair da cidade. Proponha um algoritmo eficiente para verificar a veracidade da nova asserção do Presidente da Câmara. Indique a complexidade do algoritmo proposto.

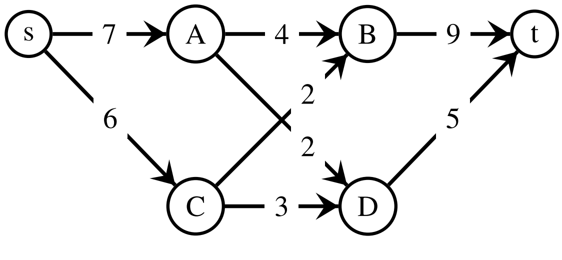
Solução:

- Determinar o grafo $G = (V, E)$ que representa Caracolândia;
- Determinar o conjunto de vértices atingíveis a partir do cruzamento da câmara municipal, chamando DFS_Visit no vértice da câmara municipal (complexidade $O(V + E)$;
- Determinar os componentes fortemente ligados de G usando o algoritmo de Tarjan ou o algoritmo de Kosaraju's (complexidade: $O(V + E)$);
- Verificar se os vértices atingíveis a partir do cruzamento da câmara municipal formam um SCC (complexidade: $O(V)$.

Complexidade do algoritmo: $O(V + E)$.

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

Numa rede de fluxo, um arco é considerado um *limitante* se, aumentando o valor da sua capacidade, o valor do fluxo máximo é aumentado. Considere a seguinte versão da rede de fluxo do exercício **I.e**.

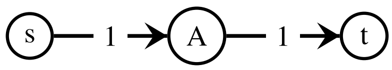


IV.a) Liste os arcos *limitantes* da rede de fluxo apresentada em cima.

Solução: Arcos *limitantes*: (A,B), (C,B).

IV.b) Nem todas as redes de fluxo têm arcos *limitantes*. Dê um exemplo com no máximo quatro vértices.

Solução:

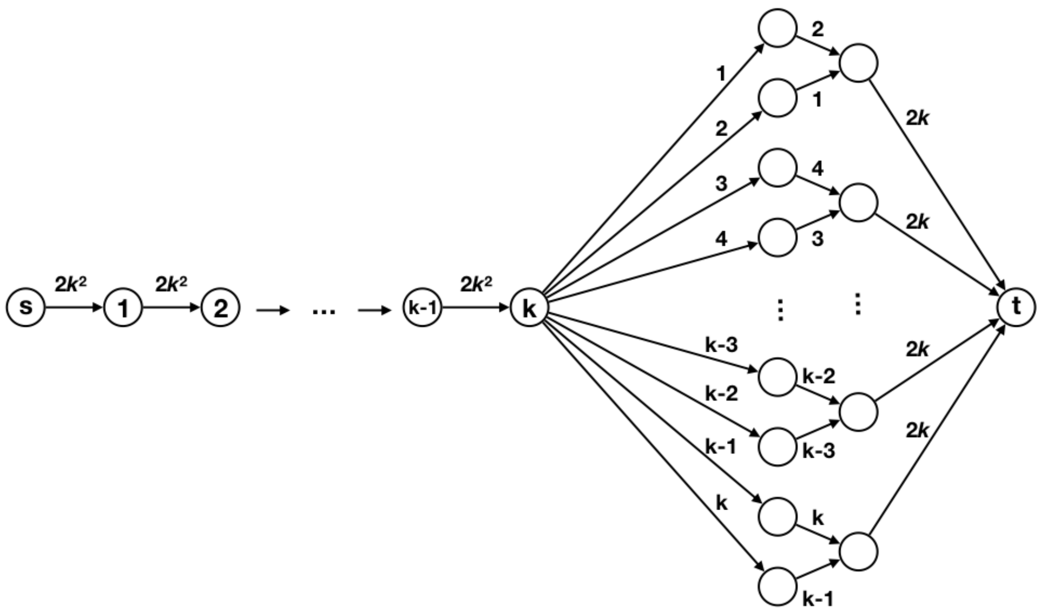


IV.c) Dado um fluxo máximo f^* numa rede de fluxo G e a respectiva rede residual G_{f^*}, proponha um algoritmo eficiente para determinar os arcos *limitantes* de G e identifique a complexidade do mesmo.

Solução: Começamos por notar que a rede residual G_{f^*} induz um corte s-t dos vértices de G; seja (S,T) esse corte. O algoritmo procede como descrito em baixo.

- Calcular o grafo transposto da rede residual G_{f^*}, $G^T_{f^*}$; complexidade: $O(V+E)$.
- Determinar o conjunto T' de vértices de T a partir dos quais é possível atingir o vértice alvo, t, executando uma DFS começando em t no grafo $G^T_{f^*}$; complexidade: $O(V+E)$.
- Determinar o conjunto S' de vértices de S atingíveis a partir do vértice origem, s, executando uma DFS começando em s em G_{f^*}; complexidade: $O(V+E)$.
- Percorrer todos os arcos de G, verificando se cada um deles é, ou não, limitante; um arco $(u,v) \in E$ é limitante se $u \in S'$ e $v \in T'$. Complexidade: $O(E)$.

V. (1,0 + 1,0 = 2,0 val.) Considere a rede de fluxo da figura em baixo cuja topologia (número de nós e de arestas e respectivas capacidades) depende de um dado parâmetro k.



V.a) Calcule o fluxo máximo do grafo da figura em função de k. (*Pista:* o fluxo deve ser calculado por inspecção da figura e não aplicando um algoritmo de fluxo). Nota: $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$

Solução: $f^* = k^2/2$.

V.b) Considere a aplicação de um algoritmo para o cálculo do fluxo máximo ao grafo acima; determine o menor majorante assimptótico medido em função do parâmetro k para:

a. um algoritmo baseado em caminhos de aumento (*Pista:* estabeleça um upper bound para o número de caminhos de aumento);

b. um algoritmo baseado em pré-fluxos (*Pista:* estabeleça um upper bound para o número de *pushes*).

Solução:

- $O(k^2)$
- $O(k^2)$