INSTITUTO SUPERIOR TÉCNICO

Análise e Síntese de Algoritmos

Ano Lectivo 2017/2018 Exame Época Especial- versão A

RESOLUÇÃO DO EXAME ÉPOCA ESPECIAL

I. (1,25 + 1,25 + 1,25 + 1,25 + 1,25 + 1,25 = 7,5 val.)

I.a)

	A	B	C	D	E	F
d	0	3	1	2	1	2
π	-	D	A	C	A	E

I.b)

	A	B	C	D	E	F
d	1	2	4	6	7	10
f	12	3	5	9	8	11
Numero de Ordenações:			40			

I.c)

	A	B	C	D	E
`key`	0	2	2	1	1

I.d)

$D^{(2)}(3,1) =$	-3	$D^{(2)}(3,2) =$	-2	$D^{(2)}(3,3) =$	-1

I.e)

Expressão:	$4 * T(n/2) + O(n^2)$
Majorante:	$O(n^2 \log n)$

I.f)

	Primeira	*Segunda*	*Terceira*
$c_f(C,t)$	7	6	5

II. (1,0 + 1,5 = 2,5 val.)

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

II.b) $\overline{\langle \text{XXX} \rangle}$

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

III.a) ⟨ XXX ⟩

III.b)

Objectivo	x_1	x_2	x_3
14	0	2/5	22/5

IV. (1,25 + 1,25 = 2,5 val.)

IV.a) ⟨ XXX ⟩

IV.b)

Greedy :	23
Programação Dinâmica :	30

V. (1,25 + 1,25 = 2,5 val.)

V.a)

Expressão	(n+1)n/2

V.b)

	π[3]	π[6]	π[8]	π[12]	π[15]
Valor	2	1	0	3	6

VI. (1,0 + 1,0 = 2,0 val.)

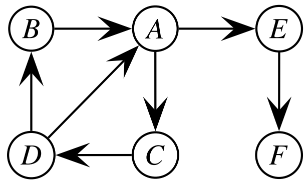
VI.a)

	a)	b)	c)	d)	e)
Resposta	V	V	D	V	F

VI.b) ⟨ XXX ⟩

I. (1,25 + 1,25 + 1,25 + 1,25 + 1,25 + 1,25 = 7,5 val.)

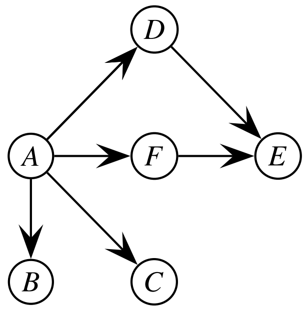
I.a) Considere o grafo dirigido:



Execute uma procura em largura (BFS) com início em A. Indique os valores de d e π para cada um dos vértices.

Nota: As distâncias de uma BFS começam em 0.

I.b) Considere o grafo dirigido acíclico (DAG):

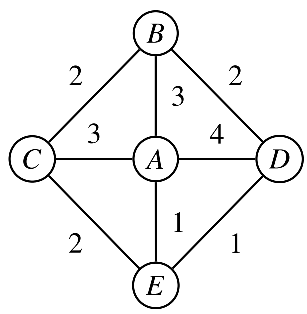


Aplique uma procura em profundidade primeiro (DFS) com início no vértice A e que visita os adjacentes por ordem lexicográfica. Os recomeços da DFS escolhem o vértice não visitado com a menor letra, de acordo com a ordem lexicográfica.

Indique os valores de descoberta (d) e fim (f), para cada um dos vértices. Indique quantas ordenações topológicas existem para este grafo.

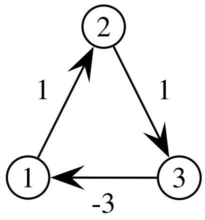
Nota: Os tempos de uma DFS começam em 1.

I.c) Considere o grafo pesado da figura.



Considere a execução do algoritmo de Prim a partir do vértice A. Indique os valores do vector `key` após ter sido processado o vértice D.

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



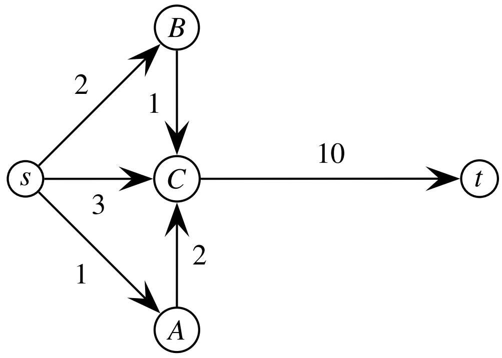
Considere a aplicação do algoritmo de Floyd-Warshall ao grafo. Indique os valoes $D^{(2)}(3,1)$, $D^{(2)}(3,2)$ e $D^{(2)}(3,3)$.

I.e) Considere a função recursiva:

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

  i = 0;
  while(i < n)
  {
    j = i;
    while(j < n)
      j++;
    i++;
  }

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

  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) Aplique o algoritmo de Edmonds-Karp à seguinte rede:



Indique o valor da capacidade residual do arco (C,t) após cada iteração do algoritmo, i.e. após o aumento de fluxo usando o caminho de aumento.

II. (1,0 + 1,5 = 2,5 val.)

II.a) Considere uma árvore abrangente de menor custo, sobre um grafo, conexo não dirigido e pesado. Considere a seguinte repesagem: $w'(u,v) = w(u,v) + d$, onde (u,v) é um arco e $d > 0$ é uma constante arbitrária mas fixa. Assuma que todos os arcos são repesados, utilizando os mesmos valores de d.

Argumente que uma árvore abrangente de menor custo do grafo repesado também é abrangente de menor custo no grafo original.

Solução: Esta repesagem preserva a ordenação dos vértices, pelo que o algoritmo de Kruskal processa os arcos exactamente da mesma forma no grafo original e no grafo repesado. Logo uma árvore abrangente de menor custo no grafo repesado também o é no grafo original.

II.b) Neste problema queremos determinar a fiabilidade de uma rede de transportes. Um rede de transportes é composta por diversos locais e transportes que podem ligar de um local de origem para um local de destino. Vamos assumir que cada transporte tem uma certa fiabilidade, que corresponde à probabilidade de que o transporte seja bem sucedido. Assuma que as fiabilidades dos transportes são independentes, ou seja se $p_{u,v}$ for a fiabilidade do local u para o local v e $p_{v,w}$ for a fiabilidade do local v para o local w então $p_{u,v} \times p_{v,w}$ indica a fiabilidade de ir de u para w, passando por v.

1) Modele este problema, utilizando estruturas que conhece.

2) Proponha um algoritmo eficiente que determina o caminho mais fiável entre dois locais.

Indique as complexidades dos algoritmos propostos.

Solução:

1) O problema pode ser modelado como um grafo dirigido e pesado em que os locais são os vertices, e os transportes são os arcos. A cada arco (u,v) associamos o valor $-\log p_{u,v}$, onde $p_{u,v}$ é a fiabilidade do transporte entre os locais.

2) O caminho mais fiável é o caminho mais curto no grafo apresentado, pelo que pode ser ulizado o algoritmo de Dijkstra com complexidade $O((E+V)\log V)$.

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

III.a) Indique o dual do seguinte programa linear:

$$\begin{array}{lrcl} min & -3x_1+2x_2-5x_3 & & \\ s.a. & x_1+x_2-6x_3 & \geq & -3 \\ & 2x_1-x_3 & \geq & 6 \\ & 3x_1+4x_2-3x_3 & \leq & 10 \\ & x_2-4x_3 & \leq & -1 \\ & x_1,x_2,x_3 & \geq & 0 \end{array}$$

Solução:

$$\begin{array}{lrcl} min & 3y_1-6y_2+10y_3-y_4 & & \\ s.a. & -y_1-2y_2+3y_3 & \geq & 3 \\ & -y_1+4y_3+y_4 & \geq & -2 \\ & 6y_1+y_2-3y_3-4y_4 & \geq & 5 \\ & y_1,y_2,y_3,y_4 & \geq & 0 \end{array}$$

III.b) Calcule o valor óptimo da função objectivo e o respectivo valor das variáveis x_1, x_2 e x_3 para o seguinte programa linear:

$$\begin{array}{lrcl} max & x_1+2x_2+3x_3 & & \\ s.a. & x_1-x_2+x_3 & \leq & 4 \\ & 3x_1+3x_2+2x_3 & \leq & 10 \\ & x_1,x_2,x_3 & \geq & 0 \end{array}$$

IV. (1,25 + 1,25 = 2,5 val.)

IV.a) Dado um conjunto S de números inteiros positivos e um valor K, o objectivo é identificar um subconjunto S' de S tal que a soma dos elementos de S' seja o mais próximo possível de K, sem ultrapassar K.

Por exemplo, se $S = \{1,3,5,8,13\}$ e $K = 20$, então a solução seria $S' = \{1,5,13\}$ dado que a soma dos elementos de S' é 19.

Indique um modelo de programação dinâmica para resolver este problema. Analise a complexidade da solução proposta.

Solução:

A abordagem usando programação dinâmica tem complexidade $O(n \times K)$ onde n denota o tamanho do conjunto S.

Considere-se a tabela $\mathtt{v}[\mathtt{i},\mathtt{j}]$, em que $\mathtt{i}$ varia de 0 a n e que itera sobre os elementos de S, e $\mathtt{j}$ varia de 0 a K. O tamanho desta tabela é portanto $O(n \times K)$.

A posição $\mathtt{v}[\mathtt{i},\mathtt{j}]$ representa o valor mais próximo de j que podemos obter usando apenas os primeiros i elementos do conjunto S, sem ultrapassar j. Considere que s_i denota o i-ésimo elemento do conjunto S.

$$\mathtt{v}[\mathtt{i},\mathtt{j}] = \begin{cases} -\infty & \text{, se } \mathtt{j} < 0 \\ 0 & \text{, se } \mathtt{j} \geq 0 \text{ e } \mathtt{i} = 0. \\ \max(\mathtt{v}[\mathtt{i}-1,\mathtt{j}], \mathtt{s}_\mathtt{i} + \mathtt{v}[\mathtt{i}-1,\mathtt{j}-\mathtt{s}_\mathtt{i}]) & \text{, caso contrário} \end{cases}$$

IV.b) Considere uma instância do problema da mochila não fraccionário com 5 objectos. A mochila tem capacidade 14 e os objectos a considerar têm os seguintes valores e pesos:

- $v_1 = 2; w_1 = 2$
- $v_2 = 9; w_2 = 3$
- $v_3 = 12; w_3 = 5$
- $v_4 = 18; w_4 = 8$
- $v_5 = 19; w_5 = 10$

Indique os valores máximos conseguidos por:

- um algoritmo *greedy* com base na ordenação dos objectos por v_i/w_i;
- um algoritmo baseado em programação dinâmica;

V. (1,25 + 1,25 = 2,5 val.)

V.a) Considere o algoritmo de autómatos finitos para o emparelhamento de caracteres. Seja $n \in \mathbb{N}$ e P o padrão $bab^2ab^3a \ldots b^{n-2}ab^{n-1}ab^na$ tal que $a \neq b$, $a,b \in \Sigma$ e $n \geq 2$. Sendo $\delta : Q \times \Sigma \to Q$ a função de transição do autómato, indique a expressão que denota o número de transições para o estado inicial.

V.b) Considere o algoritmo de Knuth-Morris-Pratt. Dado o padrão $P = \mathtt{aaabbaabaaaabba}$, calcule a função de prefixo $\pi[k]$ para o padrão P. Indique os valores de $\pi[3]$, $\pi[6]$, $\pi[8]$, $\pi[12]$ e $\pi[15]$.

VI. (1,0 + 1,0 = 2,0 val.)

VI.a) Para cada uma das afirmações seguintes, indique se é verdadeira (**V**), falsa (**F**) ou se não se sabe (**D**).

a. Se existir um problema X tal que $X \in$ NP-Completo e X é resolúvel em tempo polinomial, então $\mathrm{P} = \mathrm{NP}$
b. Se $X \in$ NP-Difícil, qualquer $Y \in$ NP verifica $Y \leq_p X$
c. $\mathrm{P} =$ co-NP
d. $\mathrm{P} \subseteq (\mathrm{NP} \cap \text{co-NP})$
e. Se para qualquer $Y \in$ NP-Completo temos que $X \leq_p Y$, então $X \in$ NP-Difícil

VI.b) Dada uma matriz A de $m \times n$ valores inteiros e um vector de inteiros b de dimensão m, o problema de Programação Linear Inteira 0-1 (ILP 0-1) consiste em verificar se existe um vector x de dimensão n tal que os elementos de x pertencem a $\{0,1\}$ e $Ax \leq b$.

Dado um grafo não dirigido $G = (V,E)$ e uma constante K, o problema VERTEX-COVER pode ser definido como identificar um subconjunto de vértices U ($U \subseteq V$) tal que o número de vértices de U não é superior a K ($|U| \leq K$) e para todos os arcos $(u,v) \in E$ temos que $u \in U$ ou $v \in U$.

Sabendo que o problema VERTEX-COVER é NP-Completo, prove que o problema ILP 0-1 é NP-Completo usando uma redução a partir de VERTEX-COVER.

(Nota: Prove primeiro que ILP 0-1 $\in$ NP.)

Solução:

Em primeiro lugar provamos que ILP 0-1 $\in$ NP. Consideramos como certificado a atribuição de valores 0 ou 1 aos elementos de x. O algoritmo de verificação valida se essa atribuição satisfaz todas as restrições do problema ($Ax \leq b$). O algoritmo é polinomial ($O(nm)$), pelo que ILP 0-1 $\in$ NP.

De seguida provamos que ILP 0-1 $\in$ NP-Dificil usando uma redução a partir de VERTEX-COVER, ou seja, VERTEX-COVER $\leq_p$ ILP 0-1. A redução é a seguinte:

- Para cada vértice u criamos uma variável x_u na instância ILP 0-1. x_u toma valor 1 se o vértice u está incluido em U e 0 caso contrário.

- Para cada arco (u,v) do grafo, definimos uma restrição $-x_u - x_v \leq -1$ para garantir que pelo menos um dos vértices do arco pertence a U.

- Definimos ainda uma restrição $\sum_{i=1}^{|V|} x_i \leq K$ para garantir que o número de vértices seleccionado não é superior a K.

Esta redução tem complexidade linear. As variáveis x_i com valor 1 na solução da instância ILP 0-1 definem os vértices do grafo que pertencem a U correspondendo à solução da instância de VERTEX-COVER.