Instituto Superior Técnico

Análise e Síntese de Algoritmos

Ano Lectivo 2019/2020 Repescagem 2º Teste

RESOLUÇÃO

I. (2,5 + 2,5 + 2,5 + 2,5 = 10,0 val.)

I.a) Considere o problema de compressão de dados de um ficheiro usando a codificação de Huffman. Indique o código livre de prefixo óptimo para cada carácter num ficheiro com 1.000 caracteres com a seguinte frequência de ocorrências: $f(a) = 29, f(b) = 10, f(c) = 15, f(d) = 10, f(e) = 2, f(f) = 34$.
Quando constrói a árvore, atribua o bit 1 ao nó com menor frequência. Em caso de empate, atribua o bit 1 ao nó que inlcui o carácter que aparece primeiro por ordem alfabética. Analogamente, considere que os nós são mantidos na *min-priority queue* por ordem de frequência, e em caso de empate, considera-se primeiro o nó que inlcui o carácter que aparece primeiro por ordem alfabética.

Indique também o total de bits no ficheiro codificado.

	a	b	c	d	e	f
Codificação						

Total Bits	

I.b) Considere o problema de multiplicar cadeias de matrizes. O objetivo é determinar por que ordem devem ser feitas as multiplicações por forma a minimizar o número total de multiplicações escalares que precisam de ser efetuadas.

Considere uma sequência com 4 matrizes $A_1(4 \times 2)$, $A_2(2 \times 2)$, $A_3(2 \times 1)$, $A_4(1 \times 4)$, com as respetivas dimensões entre parênteses. Resolva este problema preenchendo a matriz `m[i,j]` que guarda o menor número de multiplicações escalares que precisam de ser efectuadas para obter o produto das matrizes de A_i a A_j. Indique os valores de `m[2,4]`, `m[1,4]` e `m[1,3]`. Indique também a colocação de parênteses que obtém o valor indicado em `m[1,4]`. Em caso de empate associe à esquerda.

m[2,4]	m[1,4]	m[1,3]	Parênteses

I.c) Considere o padrão $P = ababaa$ e construa o autómato finito que emparelhe este padrão numa cadeia de caracteres. Indique o estado resultante das seguintes transições:

$\delta(0,b)$	$\delta(1,a)$	$\delta(2,b)$	$\delta(3,a)$	$\delta(4,b)$	$\delta(5,b)$	$\delta(6,a)$	$\delta(6,b)$

Ilustre a sua aplicação no seguinte texto $T = aabaababbaabaab$.

q	0															

I.d) Considere o seguinte programa linear:

$$\begin{array}{rrrrrl}
\min & -5x_1 & + \ x_2 & -2x_3 & & \\
\text{s.a} & -x_1 & -2x_2 & - \ x_3 & \geq & -7 \\
 & x_1 & +3x_2 & + \ x_3 & \leq & 10 \\
 & -x_1 & - \ x_2 & -6x_3 & \geq & -4 \\
 & & & x_1, x_2, x_3 & \geq & 0
\end{array}$$

Indique o valor da função objectivo e o respectivo valor das variáveis básicas e não-básicas após uma única operação de pivot. Em caso de empate utilize a regra de Bland, ou seja, comece por considerar as variáveis de índice mais baixo.

Z	x_1	x_2	x_3	x_4	x_5	x_6

II. (2,5 + 1,5 + 3 + 3 = 10 val.)

II.a) Acredita-se que uma data string de texto $T[1..n]$ corresponde a uma versão corrompida de uma string de texto original à qual foram removidos os espaços entre as palavras; por exemplo "Aquelapraiaextasiadaenua". O Eng. João Caracol foi encarregado de verificar se é possível obter uma sequência de palavras válidas a partir da string de texto T usando um algoritmo baseado em programação dinâmica. Para tal, dispõe de uma função de dicionário **dict** que, dada uma cadeia de caracteres s, verifica se s é uma palavra válida; formalmente:

$$\textbf{dict}(s) = \begin{cases} 1 & \text{se } s \text{ é uma palavra válida} \\ 0 & \text{caso contrário} \end{cases}$$

1. Seja $B(i)$ o valor booleano que indica se a cadeia de caracteres $T[1..i]$ forma uma sequência de palavras válidas. Defina $B(i)$ recursivamente completando os campos em baixo.

$$B(i) = \begin{cases} \boxed{} & \text{se } i \geq 1 \\ \boxed{} & \text{se } i = 0 \end{cases}$$

2. Complete o template de código em baixo que calcula a quantidade $B(n)$ e indique a respectiva complexidade assimptótica.

```
FindWords(t[1..n])
  let B[0..n] be a new array of size n
  B[0] = true
  for i = 1 to n do
    [          ]
  endfor
  return B[n]
```

Solução:

1. $B(i) = \begin{cases} \bigvee\{B(j) \wedge \textbf{dict}(t[(j+1)..i]) \mid j < i\} & \text{se } i > 1 \\ \textbf{true} & \text{se } i = 0 \end{cases}$

2. Em baixo o código completo:

```
FindWords(t[1..n])
  let B[0..n] be a new array of size n+1
  B[0] = true
  for i = 1 to n do
    B[i] = false
    j = i − 1
    while j ≥ 0 ∧ (not B[i]) do
      B[i] = B[j] ∧ dict(t[(j + 1)..i])
      j = j−1
    endwhile
  endfor
  return B[n]
```

Complexidade: $O(n^2)$

II.b) Considere o algoritmo de *Knuth-Morris-Pratt (KMP)* estudado nas aulas. Seja $\pi_1 = [0, 1, 0, 1, 2, 3, 4, 0]$ a função de prefixo de uma dada string s sobre um alfabeto $\Sigma = \{a, b\}$.

1. Liste todas as strings que correspondem à função de prefixo π_1.

2. Suponha que, em vez do alfabeto $\Sigma = \{a, b\}$, consideramos o alfabeto $\Sigma' = \{a, b, c\}$. Indique o número de strings sobre o alfabeto Σ' com função de prefixo π_1.

Solução:

1. Strings sobre o alfabeto Σ com função de prefixo π_1:

   - $s_1 = aabaabab$
   - $s_2 = bbabbaba$

2. $3 * 2 * 2 = 12$

II.c) Considere o seguinte programa linear:

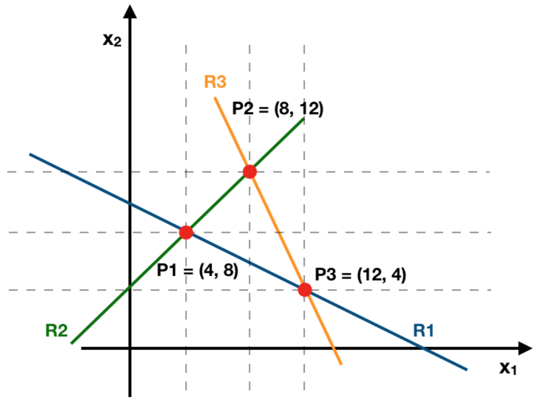
$$
\begin{array}{rrrcl}
\max & x_1 & +2x_2 & & \\
\text{s.a} & \frac{1}{2}x_1 & +x_2 & \geq & 10 \\
 & x_1 & -x_2 & \geq & -4 \\
 & 2x_1 & +x_2 & \leq & 28 \\
 & & x_1, x_2 & \geq & 0
\end{array}
$$

1. Desenhe o conjunto exequível e resolva geometricamente o programa linear. A resposta deve incluir: o valor máximo, as coordenadas onde esse valor é atingido e as equações das rectas que delimitam a região exequível.

2. Formule o programa linear auxiliar e indique três soluções diferentes para o mesmo.

Solução:

1. Representamos a região exequível no diagrama em baixo.



O Teorema Fundamental da Programação Linear estabelece que o valor óptimo da função objectivo, a existir, ocorre num vértice da região exequível. Assim sendo, concluímos que o valor óptimo é 32 e ocorre no ponto $P_2 = (8, 12)$. Equações das rectas que delimitam o conjunto exequível:

- **R1:** $x_2 = -0.5x_1 + 10$
- **R2:** $x_2 = x_1 + 4$
- **R3:** $x_2 = -2x_1 + 28$

2. O programa linear auxiliar é definido em baixo:

$$
\begin{array}{rrrrcl}
\max & & & -x_0 & & \\
\text{s.a} & -\frac{1}{2}x_1 & -x_2 & +x_0 & \leq & -10 \\
 & -x_1 & +x_2 & +x_0 & \leq & 4 \\
 & 2x_1 & +x_2 & +x_0 & \leq & 28 \\
 & & & x_1, x_2, x_0 & \geq & 0
\end{array}
$$

Qualquer ponto da região exequível é uma solução do programa auxiliar. Assim podemos considerar, por exemplo, os pontos P_1, P_2 e P_3 definidos na figura (estendidos com a coordenada $x_0 = 0$.

- $P_1' = (4, 8, 0)$
- $P_2' = (8, 12, 0)$
- $P_3' = (12, 4, 0)$

II.d) Seja $C = \{c_1, ..., c_n\}$ uma colecção de cromos e $\mathcal{A} = \{a_1, ..., a_m\}$ um grupo de amigos que coleccionam cromos. Cada membro do grupo detém um subconjunto de C; seja C_i o conjunto de cromos detido por a_i e $\mathcal{C} = \{C_i \mid 1 \leq i \leq m\}$ o conjunto dos conjuntos de cromos de todos os membros do grupo. Os membros do grupo pretendem determinar o mais pequeno conjunto de cromos que contém pelo menos um cromo detido por cada membro do grupo. Formalmente, este problema pode ser modelado através do seguinte problema de decisão:

$$\textbf{SharedStickers} = \{\langle C, \mathcal{C}, k\rangle \mid \exists X \subseteq C. |X| = k \wedge \forall_{1 \leq i \leq m}. C_i \cap X \neq \emptyset\}$$

1. Mostre que o problema **SharedStickers** está em **NP**.
2. Mostre que o problema **SharedStickers** é **NP**-difícil por redução a partir do problema da *Cobertura de Vértices* que foi estudado nas aulas e que recordamos em baixo.

Problema da Cobertura de Vértices: Seja $G = (V, E)$ um grafo não dirigido; dizemos que $V' \subseteq V$ é uma cobertura de vértices se e só se: $\forall (u, v) \in E. u \in V' \vee v \in V'$. O problema da cobertura de vértices, **VCover**, define-se formalmente da seguinte maneira:

$$\textbf{VCover} = \{\langle G, k\rangle \mid G \text{ contém uma cobertura de vértices de tamanho } k\}$$

Solução:

1. O algoritmo de verificação recebe como input uma possível instância $\langle C, \mathcal{C}, k\rangle$ e um conjunto de cromos X (o certificado). O algoritmo tem de verificar que $|X| = k$ e que $X \cap C_i \neq \emptyset$, para todo o C_i em $\mathcal{C}$. Observamos os certificados têm tamanho $O(n)$ e que a verificação se faz em tempo $O(m.n^2)$, a complexidade de se calcular a intersecção de cada um dos conjuntos em $\mathcal{C}$ com X utilizando um algoritmo naif de complexidade quadrática.

2. Dada uma possível instância $\langle G, k\rangle$ do problema **VCover**, começamos por construir uma colecção de cromos C e um conjunto de conjuntos $\mathcal{C}$. Intuitivamente, os vértices do grafo G correspondem aos cromos da colecção e cada arco corresponde a um conjunto de dois vértices. Formalmente:

   $$\langle G, k\rangle \in \textbf{VCover} \Leftrightarrow \langle C, \mathcal{C}, k'\rangle \in \textbf{SharedStickers}$$

   onde:

   - $C = V$;
   - $\mathcal{C} = \{\{u, v\} | (u, v) \in E\}$;
   - $k' = k$.

   Complexidade da redução: $O(V + E)$.