Tópicos Avançados em Algoritmos 2008/2009 - época normal

Enunciado com frente e verso / Cotação indicada em percentagem / Duração da prova: 2h15m

1. [16%] **Diga se cada uma das seguintes afirmações é verdadeira ou falsa (não é necessário justificar)**
   - (a) **É possível multiplicar 2 matrizes quadradas de dimensão $n \times n$ em tempo $O(n^{2.9})$.**
   - (b) **Para qualquer função $f : \mathbb{N} \to \mathbb{R}^+$ verifica-se $\Theta(f(n)) \subseteq O(f(n))$.**
   - (c) **A mediana de um vector (não necessariamente ordenado) pode ser determinada em tempo $O(n)$ no caso mais desfavorável.**
   - (d) **Qualquer problema de decisão pertencente à classe RP é decidível.**
   - (e) **Existe uma constante inteira c, tal que qualquer algoritmo de ordenação (de um vector com n elementos) baseado em comparações efectua, para n suficientemente grande, pelo menos $cn \log n$ comparações no caso mais favorável.**
   - (f) **O comprimento máximo de uma sequência (não necessariamente consecutiva) comum às "strings" s e t pode ser determinado (no pior caso) em tempo polinomial.**

2. **Procura de x num vector ordenado**
   Considere o problema de procurar um valor x num vector `v[1..n]` com n elementos distintos, ordenado por ordem crescente. Algoritmo da pesquisa binária:

```
// Chamada inicial da forma pb(x,v[1..n])
function pb(x,v[a..b]):
     if a>b: return -1
     m = (a+b)/2              //  divisão inteira
     compare x com v[m]:      (*)
     if x==v[m]: return m
     if x<v[m]:  return pb(x,v[a..m-1])
     return pb(x,v[m+1..b])
```

   Suponha que `n` é da forma `n`$= 2^p - 1$ com p inteiro positivo.
   - (a) [05%] **Escreva uma recorrência que define o número máximo de comparações entre `x` e elementos de `v[]` (linha (*)).**
   - (b) [05%] **Mostre que a solução da recorrência que definiu na alínea anterior é $\log(n+1)$.**
   - (c) [05%] **Usando a Teoria da Informação (princípio da informação necessária) mostre que qualquer algoritmo baseado em comparações que determine o índice em que se encontra x no vector `v` (ou `-1` se x não corre em `v`) efectua, no pior caso, $\lceil \log_3(n+1) \rceil$ comparações. Que pode afirmar relativamente à eficiência do algoritmo de pesquisa binária (2 primeiras alíneas deste problema)?**

3. **Ordenação estável num universo pequeno**
   Considere o seguinte algoritmo de ordenação do vector `v[]`. Supõe-se que os valores que estão no vector estão compreendidos entre 1 e `u`.

```
 Algoritmo de ordenação por contagem
 Vector a ordenar: v[1..n]
 O resultado fica no vector w[1..n]

1   for i=1 to u:  c[i] = 0
2   for i=1 to n:  c[v[i]] = c[v[i]]+1
3   for i=2 to u:  c[i] = c[i]+c[i-1]
4   for i=n downto 1:
5     w[c[v[i]]] = v[i]
6     c[v[i]] = c[v[i]] -1
```

   - (a) [04%] **Descreva por palavras o conteúdo do vector `c` (isto é, para cada `i`, o que contém `c[i]`?) imediatamente <u>antes da primeira</u> execução da linha 4.**
   - (b) [04%] **Descreva por palavras o conteúdo do vector `c` imediatamente <u>antes de cada</u> execução da linha 4.**
   - (c) [04%] **Descreva por palavras o conteúdo do vector `c` <u>no final</u> da execução do algoritmo.**
   - (d) [10%] **Mostre que a ordenação efectuada por este algoritmo é estável.**

4. **Recorrências e teoremas...**
   [10%] **Apresente um exemplo de uma equação geral de uma recorrência em que seja aplicável tanto o Teorema 1 como o Teorema 2 ("Recorrências com solução $O(n)$"), ver enunciados no fim desta prova; mostre que há concordância entre essas 2 aplicações.**

Continua do outro lado

5. **Custo amortizado do contador binário**

(a) [05%] **Defina (majorante do) custo amortizado de uma sequência de operações.**

(b) [09%] **Considere um contador binário, inicialmente com 0; são efectuados n operações de `INCR` (incremento de 1) e pretende-se mostrar que um custo amortizado dessa sequência de operações é <u>2</u>. O modelo de custos é o seguinte: cada bit que muda de valor contribui com 1 para o custo. Por exemplo, o custo de `INCR(0101011)`→0101100 é 3.**
→Para resolver o problema use um método directo em que se contabiliza para cada $i \geq 0$ o número de vezes que o bit de ordem i é modificado.

6. **Comprimento da mais longa sub-sequência comum**
Considere o problema da determinação do comprimento da mais longa sub-sequência (não necessariamente consecutiva) comum a 2 strings s e t.

(a) [08%] **"Se[1] $s, t \in \{a, b\}^\star$, $s_a > s_b$ e $t_a > t_b$, então a maior sub-sequência comum a s e a t tem comprimento $\min\{s_a, t_a\}$". A afirmação anterior é verdadeira ou falsa? <u>Justifique.</u>**

(b) [15%] **Sendo $i \geq 2$ e $j \geq 2$, a seguinte recorrência define $c(i, j)$, o comprimento da mais longa sub-sequência comum a `s[0..i]` e `t[0..j]`.**

$$c(i,j) = \begin{cases} 1 + c(i-1, j-1) & \text{se } x_i = y_j \quad \textbf{(1)} \\ \max(c(i-1,j), c(i, j-1)) & \text{se } x_i \neq y_j \quad \textbf{(2)} \end{cases}$$

Mostre através de um exemplo que, mesmo que seja $x_i = y_j$, a escolha da segunda opção (2) <u>pode</u> fornecer também o comprimento da mais longa sub-sequência comum (e corresponder a uma sub-sequência máxima comum).

Fim da prova

<u>Teorema 1</u>
A solução de uma recorrência com a equação geral da forma $t(n) = at(n/b) + cn^k$ onde a e b são inteiros com $a \geq 1$ e $b \geq 2$, c e k são reais positivos, tem a seguinte ordem de grandeza

$$\begin{cases} t(n) \in \Theta(n^{\log_b a}) & \text{se } a > b^k \\ t(n) \in \Theta(n^k \log n) & \text{se } a = b^k \\ t(n) \in \Theta(n^k) & \text{se } a < b^k \end{cases}$$

<u>Teorema 2</u> (Recorrências com solução $O(n)$)
Seja a equação geral de uma recorrência

$$f(n) = f(k_1 n) + f(k_2 n) + \ldots + f(k_p n) + cn$$

onde $c \geq 0$ e k_1, k_2,..., k_p são constantes positivas com $k_1 + k_2 + \ldots + k_p < 1$. Então $f(n)$ é de ordem $O(n)$. Inversamente, se $k_1 + k_2 + \ldots + k_p > 1$, então $f(n)$ não é de ordem $O(n)$.

[1] s_a indica o número de a's na sequência s; por exemplo, se $s = ababaa$, é $s_a = 4$.