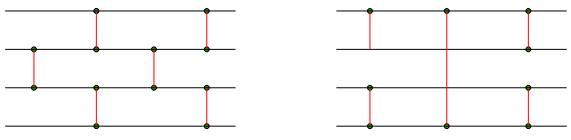
Tópicos Avançados em Algoritmos - Folha de exercícios com resolução sumária

1. Verdadeiro ou falso?
   Notas. Elemento de ordem i de um vector $v[1 \cdots n]$: aquele que ficaria na posição i se o vector estivesse ordenado. Mediana: elemento de ordem $\lfloor i/2 \rfloor$. Os tempos considerados são tempos no pior caso; $n = |x|$.
   (a) No algoritmo aleatorizado do quick-sort, quaisquer que sejam os dados x (vector a ordenar com n valores), o tempo médio é de ordem $O(n \log n)$. [v] (b) Qualquer linguagem na classe BPP é recursiva. [v] (c) O menor elemento de um vector já ordenado pode ser determinado em tempo $O(1)$ [v] (d) O menor elemento de um vector (não nece. ordenado) pode ser determinado em tempo $O(1)$ [f] (e) Para qualquer i, o elemento de ordem i de um vector (não nece. ordenado) pode ser determinado em tempo $O(\log n)$ [f] (f) A mediana de um vector (não nece. ordenado) não pode ser determinada em tempo $O(n)$ [f]
2. Suponha que uma linguagem L pertence à classe RP, existindo um algoritmo aleatorizado polinomial A tal que $x \in L \Rightarrow [\text{prob}\ \{A(x) = 1\} \geq 1/2]$ e $x \notin L \Rightarrow [\text{prob}\ A(x) = 1 = 0]$. Mostre como transformar este algoritmo num algoritmo B que é análogo a A excepto no facto de a probabilidade de errar ser $\leq 10^{-6}$. Justifique.
   →Consideremos o seguinte "algoritmo" (com probabilidade 0 pode não terminar)

```
def B(x):
  for i=1 to 20  // 20 vezes
    r = A(x)
    if r==1
      return TRUE
  return FALSE
```

   Se $x \in L$ e o resultado é FALSE, o algoritmo A errou 20 vezes; a probabilidade de isso ancontecer (sucessão de 20 "experiências" independentes) é $\leq 2^{-20} \approx 9.53 \times 10^{-7} < 10^{-6}$. É esta a única situação em que o algoritmo B pode errar.
   Nota. O valor 20 foi calculado (mentalmente) pela condição $2^{-m} \leq 10^{-6}$, o que corresponde a $m \geq 20$.
3. Na demonstração (por indução) de que o número de comparações $t(n)$ efectuadas pelo algoritmo clássico "quicksort"...
   →Ver apontamentos
4. Pretende-se ordenar um vector com n inteiros, eventualmente repetidos, compreendidos entre 1 e u, sendo $u \leq 1000$. Descreva numa linguagem de programação informal (mas de forma exacta) um algoritmo para esse fim com eficiência $O(n + u)$.
   →Ver o "algoritmo da contagem" nos apontamentos
5. Mostre que a ordenação efectuada em cada etapa (excepto na última) do "radix sort" tem que ser estável.
   →Seja 1 a ordem menos significativa (mais à direita). O algoritmo baseia-se no facto de que, quando se está a ordenar segundo os caracteres (ou dígitos) de ordem j, os valores se encontrarem já ordenados lexicograficamente segundo as ordens $j-1$, $j-2$,..., 1 (esta condição é vacuosamente verificada na primeira etapa). Assim, há que atender
   a) à ordenação prévia lexicográfica segundo as ordens menores que j
   b) à ordenação segundo a ordem j

   É óbvio que a ordenação em b) não pode "destruir" as ordenações anteriores (correctas e estáveis) de a), isto é, tem que ser estável.
6. Nem sempre é fácil mostrar que uma rede de comparadores é uma rede de ordenação. Por exemplo, para a rede **A** em baixo, tal prova não é trivial.
   (a) Indique um resultado que permite poupar bastante tempo na demonstração referida.
   →Enuncie o princípio 0/1.
   (b) Suponha que as entradas na rede **A** são, de cima para baixo, 4, 3, 2, e 1. Indique os valores nos diversos pontos da rede.
   →Sucessivamente em linhas verticais, da esquerda para a direita: 4321, 4231, 2413, 2143, 1234.
   (c) Mostre que a rede **B** não é de ordenação.
   →Vimos num exercício que toda a rede de ordenação com n linhas tem que ter comparadores $1 \leftrightarrow 2$, $2 \leftrightarrow 3$,..., $(n-1) \leftrightarrow n$; mas a rede B não tem comparador $2 \leftrightarrow 3$.
   (d) Determine a profundidade da rede **A**. →4.
   (e) Determine a profundidade da rede **B**. →3.



A **B**

7. Quais das seguintes sequências são bitónicas: ε, 000, 010, 000111000, 0101? →Todas excepto a última.

8. Como sabe, a rede "ordenador bitónico" BS(n) que foi estudada é constituída, para n da forma $n = 2^p$ com $p \geq 1$, por (i) uma rede HC(n) seguida de (ii) duas redes BS($n/2$). Sabendo que o

   – o número de componentes de uma rede HC(n) é $n/2$

   – em qualquer saída a profundidade de uma rede HC(n) é 1

   determine

   (a) a profundidade $d(n)$ de BS(n)
   →Vamos mostrar que a profundidade $d_b(n)$ em todas as linhas de saída de BS(n) é $\log n$. Usando o facto (fácil de verificar) que a profundidade $d_b(n)$ em todas as linhas de HC(n) é 1, temos

   $$\begin{cases} d_b(1) = 0 \\ d_b(n) = 1 + d_b(n/2) \quad \text{para } n \geq 2 \end{cases}$$

   A solução desta recorrência é $d_b(n) = \log(n)$.

   (b) a número de componentes $c(n)$ de BS(n)
   →Vamos mostrar que o número de componentes $c_b(n)$ de BS(n) é $\frac{n}{2}\log n$. Usando o facto (fácil de verificar) que o número de componentes de HC(n) é $n/2$, temos

   $$\begin{cases} c_b(1) = 0 \\ c_b(n) = n/2 + c_b(n/2) \quad \text{para } n \geq 2 \end{cases}$$

   A solução desta recorrência é $c_b(n) = \frac{n}{2}\log(n)$.

9. Suponha que para um determinado n fixo temos uma rede de ordenação com c comparadores. Mostre que existe um um programa sequencial, sem ciclos nem chamadas a funções, que ordena qualquer sequência de n valores efectuando exactamente c comparações. Todas as instruções do programa devem ser da forma

   $$\texttt{if x[i]} < \texttt{x[j]} : \ \texttt{x[i]} \leftrightarrow \texttt{x[j]}$$

   Exemplifique para a rede **A**.
   →Vamos caracterizar um algoritmo que constrói tal programa a partir de uma rede dada. O programa tem uma variável `x[i]` por cada linha.

   A) Determina a profundidade em cada saída de um ordenador da rede. O método é: sempre que um ordenador tem definidas as profundidades d_1 e d_2 das suas entradas, determina-se a profundidade da saída, $\max\{d_1, d_2\}$.

   B) Para $k = 1, 2, :$
      Para cada um dos comparadores com profundidade à saída k
      Sejam i e j as linhas de entrada do ordenador
      produz-se a instrução: `if x[i]<x[j]: x[i] <--> x[j]`

   Exemplifique para a rede **A**!

10. "Para $n \geq 4$ toda a rede de ordenação com n linhas tem a seguinte propriedade: qualquer linha é extremidade de pelo menos 2 ordenadores".
    Demonstre a afirmação anterior ou, se ela for falsa, apresente um contra-exemplo.
    →É falsa, um contra-exemplo é a rede estudada baseada no método da inserção (ou no "bubble sort") em que a última linha liga apenas a um comparador.

11. "Se as entradas de um comparador são x_1 e x_2 e as saídas y_1 e y_2 e se f for uma função que satisfaz $z \leq z' \Rightarrow f(z) \leq f(z')$, então, se as entradas forem $f(x_1)$ e $f(x_2)$, as saídas são $f(y_1)$ e $f(y_2)$".
    Demonstre a afirmação anterior ou, se ela for falsa, apresente um contra-exemplo.
    →É verdadeira. A relação entre as entradas e as saídas é

    $$\begin{cases} y_1 = \min(x_1, x_2) \\ y_2 = \max(x_1, x_2) \end{cases}$$

    Assim, se as entradas forem $f(x_1)$ e $f(x_2)$, as saídas são

    $$\begin{cases} w = \min(f(x_1), f(x_2)) = f(\min(x_1, x_2)) \\ z = \max(f(x_1), f(x_2)) = f(\max(x_1, x_2)) \end{cases}$$

    Pois $x_1 \leq x_2 \Rightarrow f(x_1) \leq f(x_2)$ (monotonia de f) o que implica

    $$\min(f(x_1), f(x_2)) = f(x_1) = \min(x_1, x_2)$$

    e analogamente nos ouros casos.