Departamento de Ciência de Computadores FCUP

Algoritmos (CC4010) 2015/16

Teste (02.12.2015) *duração: 3h*

N.º [ ] Nome [ ]

1. Indique se é necessário, possível ou impossível que um algoritmo $O(n \log_2 n)$ seja $\Omega(n^2)$. Justifique.

2. Averigue se $O(5n \log_2 n) \cap \Omega(3n + 7n \log_2 n) \neq \emptyset$. Justifique usando a definição das classes $O(\cdot)$ e $\Omega(\cdot)$.

3. Apresente a ideia principal da prova de que qualquer algoritmo de ordenação que se baseie em comparação tem no pior caso complexidade temporal $\Omega(n \log_2 n)$, sendo n o número de inteiros a ordenar.

4. Considere o problema Coin Change referido nas aulas, com moedas de 1, 2, 5, 10, 20, 50, 100 e 200 cêntimos, em número ilimitado.

a) Qual é a estratégia greedy que resolve o problema?

b) Dê um exemplo em que a estratégia greedy não seria correta, se o número de moedas fosse limitado.

5. Por aplicação de uma estratégia greedy, determine uma solução ótima da instância seguinte do problema "unit task scheduling", sendo a_i, d_i e w_i o identificador da atividade i, o seu deadline e a sua penalização. Apresente o algoritmo e justifique sucintamente a sua correção (baseando-se na matéria dada).

a_i	1	2	3	4	5	6	7	8	9	10	11	12
d_i	4	7	3	4	4	5	2	1	1	3	5	7
w_i	20	10	40	50	70	70	60	10	40	80	10	30

6. Considere a função PARTITION usada por QUICKSORT.

```
PARTITION(A, p, r)
| x = A[r]
| i = p − 1
| for j = p to r − 1
|      if A[j] ≤ x then
|           i = i + 1
|           Exchange A[i] with A[j]
| Exchange A[i + 1] with A[r]
| return i + 1
```

a) Apresente o invariante de ciclo que permite justificar formalmente a correção da função PARTITION.

b) Apresente o algoritmo RANDOMIZED_QUICKSORT(A, p, r) e indique os passos principais da análise da sua complexidade, assumindo que os valores são todos distintos.

7. Apresente um algoritmo estável, baseado em *counting sort*, para ordenar n inteiros com valores num intervalo $[A, B]$ em $O(n)$. Admita que $A < B$ são inteiros dados e $B - A + 1 \in o(n)$.

(Continua, v.p.f.)

8. Recorde a seguinte descrição do algoritmo HEAPSORT, com complexidade temporal $O(n \log n)$.

HEAPSORT(A):

    BUILD_MAX_HEAP(A)
    for $i = A.length$ **downto** 2
        Exchange $A[1]$ with $A[i]$
        $A.heapsize = A.heapsize - 1$
        MAXHEAPIFY($A, 1$)

a) Suponha que na chamada da função $A = [1, 2, 3, 4, 5, 6, 7, 8]$ e $A[1] = 1$ e $A.length = n = 8$. Apresente os passos principais de BUILD_MAX_HEAP(A), representando A por uma árvore.

b) O que faz a função MAXHEAPIFY? Ilustre a sua aplicação na primeira iteração do ciclo **for**.

c) Que invariante é mantido no ciclo **for** para garantir a correção do algoritmo?

d) Apresente um exemplo em que o número de comparações realizadas por MAXHEAPIFY($A, 1$) na primeira iteração do ciclo seria máximo.

e) Apresente sucintamente as ideias principais da prova de que HEAPSORT(A) é $O(n \log n)$, se o número de elementos a ordenar for n.

9. Apresente o algoritmo de Karatsuba para cálculo do produto de dois inteiros positivos, x e y, representados numa base B com n dígitos. Indique a sua complexidade temporal assintótica, começando por apresentar a recorrência que a define (e justificando sucintamente). Assuma que n é uma potência de 2 de expoente natural.

(FIM)

Master theorem:

Let $a \geq 1$ and $b > 1$ be constants, let $f(n)$ be a function, and let $T(n)$ be defined on the nonnegative integers by the recurrence $T(n) = aT(n/b) + f(n)$, where we interpret n/b to mean either $\lfloor n/b \rfloor$ or $\lceil n/b \rceil$. Then $T(n)$ has the following asymptotic bounds:

1. If $f(n) = O(n^{\log_b a - \varepsilon})$ for some constant $\varepsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.
2. If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \log_2 n)$.
3. If $f(n) = \Omega(n^{\log_b a + \varepsilon})$, for some constant $\varepsilon > 0$, and if $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$.

Stirling's approximation:

$$n! = \sqrt{2\pi n}\left(\frac{n}{e}\right)^n (1 + \Theta(1/n)) = \sqrt{2\pi n}\left(\frac{n}{e}\right)^{\alpha_n}, \quad \text{with } 1/(12n+1) < \alpha_n < 1/(12n)$$

Some useful results:

$$\log(\prod_{k=1}^{n} a_k) = \sum_{k=1}^{n} \log a_k \qquad\qquad \sum_{k=0}^{\infty} kx^k = \frac{x}{(1-x)^2}, \ \text{ for } |x| < 1$$

If $(u_k)_k$ is an arithmetic progression (i.e., $u_{k+1} = r + u_k$, for some constant $r \neq 0$), then $\sum_{k=1}^{n} u_k = \frac{(u_1+u_n)n}{2}$.

If $(u_k)_k$ is a geometric progression (i.e., $u_{k+1} = ru_k$, for some constant $r \neq 1$), then $\sum_{k=1}^{n} u_k = \frac{u_{n+1}-u_1}{r-1}$.

If $f \geq 0$ is continuous and a monotonically increasing function, then

$$\int_{m-1}^{n} f(x)dx \leq \sum_{k=m}^{n} f(k) \leq \int_{m}^{n+1} f(x)dx$$
