Departamento de Ciência de Computadores | FCUP

Algoritmos (CC4010) | 2020/21

Exame (19.04.2021) | *duração: 2h*

N.º ______ Nome ______

1. [2.0] Averigue a veracidade de cada uma das afirmações seguintes e justifique a resposta para **a)** e **d)**, usando **diretamente** a definição das ordens de grandeza.

a) $2^n \in \Theta(3^n)$ ☐ **b)** $\log_2 n \in \Theta(\log_3 n)$ ☐ **c)** $3^n \in \Omega(2^n)$ ☐ **d)** $2^n + 10n \in O(2^n)$ ☐

2. [2.0] Supondo que P $\neq$ NP, classifique os problemas seguintes como membros ou não das classes de problemas P, NP e NP-completos, justificando. Pode usar conhecimentos sobre outros problemas.

a) Dados n e uma sequência $c_1, \ldots, c_n$ de inteiros positivos, e ainda um valor k, decidir se existe $I \subseteq \{1, \ldots, n\}$ tal que $|I| \geq k$ e $c_i \geq c_j$, qualquer que seja $i \in I$ e $j \in \{1, \ldots, n\}$, sendo $|I|$ o número de elementos de I.

b) Dado um grafo não dirigido $G = (V, E)$ e dada uma função $p : V \to \mathbb{Z}^+$ que associa um peso a cada vértice de G e ainda um valor $P \in \mathbb{Z}^+$, decidir se existe um conjunto de vértices $W \subseteq V$ tal que qualquer que seja a aresta $e \in E$, algum dos extremos de e pertence a W, e $\displaystyle\sum_{v \in W} p(v) \leq P$.

Dos problemas 3-9, resolva apenas quatro

3. *"In 1998, the problems of finding the minimum number of point guards, vertex guards, and edge guards required to guard a simple polygon were shown to be APX-hard by Eidenbenz, Widmayer and Stamm."* A menos que P=NP, esse resultado implica que existem instâncias para as quais não se pode determinar uma solução ótima em tempo polinomial.

Admita que os guardas são colocados em vértices e podem ter visão com amplitude até 2π e alcance ilimitado. Sabendo que a triangulação de um polígono simples com n vértices se pode calcular em $O(n \log n)$ (de facto, em $O(n)$) e que se pode encontrar uma coloração mínima da triangulação em $O(n)$, explique porque é que o método de Fisk não pode ser usado para determinar o número mínimo de guardas em tempo polinomial, e a sua localização. Comece por descrever o método de Fisk.

4. Considere a função SELECT assim definida, sendo x é um *array* de n inteiros positivos e distintos $x[1], \ldots, x[n]$, e a e b inteiros tais que $1 \leq a \leq b \leq n$. Queremos que SELECT(x, a, b, k) determine o k-ésimo **maior** elemento de x no segmento definido por a e b. **Admita que a e b satisfazem a condição indicada e $1 \leq k \leq b - a + 1$.**

SELECT(x, a, b, k)
1. **if** $a = b$ **then return** $x[a]$
2. $t =$ MYPARTITION(x, a, b)
3. $p = t - a + 1$
4. **if** $p = k$ **then return** $x[t]$
5. **if** $p < k$ **then**
6.   **return** SELECT$(x, t+1, b, k-p)$
7. **return** SELECT$(x, a, t-1, k)$

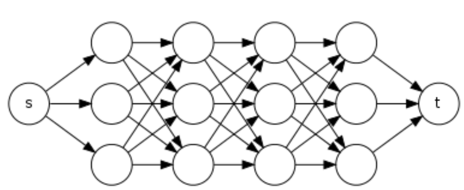
a) Indique o que a função MYPARTITION retorna para que SELECT(x, a, b, k) esteja correta. Indique ainda o estado do *array* x após a execução de MYPARTITION, na linha 3.

b) Assumindo a condição que indicou em **a)**, justifique a correção de SELECT(x, a, b, k), i.e., que retorna o k-ésimo maior elemento no segmento.

c) Para *quickselect* (randomizado) e *median of five medians*, aplicados ao mesmo problema, que passo(s) seria(m) diferente(s)? Explique.

5. Seja $G = (V, E)$ um grafo dirigido acíclico (DAG). Designe por $Adj[v]$ o conjunto dos vértices adjacentes a v, para $v \in V$. Assuma que esse conjunto é representado por uma lista ligada simples. Pretendemos contar o número de caminhos em G de s para t, sendo s e t dois nós dados.

a) Qual é a solução da instância seguinte? Explique as vantagens de se poder usar programação dinâmica.



b) Apresente, em pseudocódigo, uma função CONTA(s, t) para resolver o problema, que use programação dinâmica (com memoização).

c) Assumindo que os cálculos podem ser efetuados em $O(1)$, caraterize a complexidade temporal e espacial da função que apresentou.

6. Considere a determinação do invólucro convexo *(convex hull)* de n pontos $p_1, p_2, p_3, \ldots, p_n$ no plano, pelo algoritmo de Graham (*Graham-scan*), usando varrimento rotacional. Suponha que p_1 é o ponto de menor ordenada não existindo pontos com o mesmo ângulo polar relativamente a um referencial com origem em p_1 nem com a mesma ordenada.

a) Apresente (em pseudocódigo) os restantes passos do algoritmo.

b) Indique a complexidade de cada passo para que tenha complexidade ótima no pior caso. Qual é essa complexidade?

c) Que vantagens/desvantagens pode ter face a *Jarvis march* (gift-wrapping)?

7. Suponha que estavam marcadas várias reuniões para um certo dia, sendo dada a hora de início e a duração de cada reunião. Admita que, por qualquer imprevisto, uma mesma pessoa tem de estar presente nessas reuniões (mas apenas numa única em cada momento). Havendo colisões, algumas reuniões terão de ser canceladas.

a) Apresente uma estratégia *greedy* para selecionar um conjunto de reuniões a realizar de forma a minimizar o número de reuniões a cancelar.

b) Demonstre a correção da estratégia que indicou.

c) Diga como pode ser implementada e indique a complexidade temporal assintótica dessa implementação.

8. Considere o problema da determinação de um ciclo de Hamilton com peso mínimo num grafo $G = (V, E, \omega)$ não dirigido pesado. Admita que os pesos nas arestas satisfazem a desigualdade triangular $\omega(x, y) \leq \omega(x, z) + \omega(z, y)$, quaisquer que sejam $x, y, z \in V$. O grafo pode **não ser completo**.

a) Apresente um algoritmo polinomial com fator de aproximação 2 para o problema, admitindo que G é completo. Apresente a prova de que é polinomial e garante o fator de aproximação que indicou.

b) Sendo G completo, é possível ter um algoritmo polinomial com fator de aproximação inferior a 2?

c) Se G não for completo, o algoritmo que indicou pode ser adaptado? Se sim, indique como. Se não, explique porquê.

9. Recorde a estrutura de *heap* para suporte a filas de prioridade.

a) Apresente o algoritmo para construção de uma *heap* de mínimo a partir de um *array* v de n elementos inteiros em $\Theta(n)$ sem recurso a um *array* auxiliar.

b) Explique porque é que essa construção não pode ser usada para reduzir a ordem de grandeza da complexidade temporal de *heapsort* que é $\Omega(n \log n)$, no pior caso.

c) Na análise da complexidade temporal assintótica de `heapify`, usou-se $T(n) \leq T(2n/3) + c$, com c constante. Explique sucintamente porquê e o que se conclui.

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 $T(n) = \Theta(f(n))$.