Departamento de Ciência de Computadores FCUP

Desenho e Análise de Algoritmos (CC211) 2013/14

1º Teste Escrito (13.11.2013) *duração: 2h30* **Cotação: 1+4+3, 4×1.5, 4+2**

N.º Nome

1. Considere o problema de ordenar por **ordem crescente** um vetor com n inteiros, dados nas posições $v[0], v[1], \ldots, v[n-1]$, com $n \geq 2$, e o seguinte algoritmo para o resolver.

```
1. Para k ← 1 até n − 1 fazer
2.     x ← v[k];
3.     j ← k − 1;
4.     Enquanto (j ≥ 0 ∧ x < v[j]) fazer
5.         v[j + 1] ← v[j];
6.         j ← j − 1;
7.     v[j + 1] ← x;
```

a) Para a instância $v = [4, -5, 30, -1, 2], n = 5$, qual é o estado de v, de x e de j após a execução do **bloco 2.-6.**, para $k = 3$?

b) Analise a complexidade temporal do algoritmo. Apresente as conclusões usando as notações Θ, O, e Ω (deve dar uma resposta elucidativa e útil). Justifique, indicando: um modelo de tempos para instruções básicas; expressões que definem o tempo total que o algoritmo demora, no melhor e no pior caso (e descreva casos desses); e provas de que o tempo pertence às classes que referir.

(CONTINUA, v.p.f.)

c) Seja $a_0, a_1, \ldots, a_{n-1}$ o estado inicial de $v[0], v[1], \ldots, v[n-1]$. Para um valor **fixo da variável** k, mas que não poderá particularizar, apresente uma condição sobre o estado das variáveis x, j e v, imediatamente após a execução do bloco de instruções 2.-6. Essa condição deve ser um invariante de ciclo e crucial para a correção do algoritmo. Justifique sucintamente que se trata de um invariante e diga como o usaria para concluir que o algoritmo resolve corretamente o problema de ordenação.

2. O algoritmo apresentado abaixo faz parte de um programa em que se pretende escrever os vértices de um grafo dirigido acíclico G (ou do seu transposto G^T) por ordem topológica. No início do ciclo "Enquanto", a pilha P está vazia, $GrauE[v]$ contém o grau de entrada do vértice v em G, para cada $v \in G.V$, e S é o conjunto dos $v \in G.V$ tais que $GrauE[v] = 0$.

```
Enquanto (S ≠ ∅) fazer
    v ← RetiraUmElemento(S);
    Push(P, v);
    Para cada w ∈ G.Adjs[v] fazer
        GrauE[w] ← GrauE[w] − 1;
        Se GrauE[w] = 0 então S ← S ∪ {w};
Se (opcao = 1) então ResG(P); senão ResTranspG(P);
```

a) Escreva as funções ResG(P) e ResTranspG(P) que imprimem $G.V$ segundo uma ordem topológica de G e de G^T. Justifique que estão corretas e indique a sua complexidade temporal.

(CONTINUA, v.p.f.)

N.º | Nome

(cont. 2a))

b) A função RetiraUmElemento(S) escolhe um elemento de S à sorte, retira-o de S e retorna esse valor. Porque é que é correto escolher v assim?

c) Como é que a instrução $GrauE[w] \leftarrow GrauE[w] - 1$ contribui para a correção do algoritmo?

d) Admita que a complexidade temporal de RetiraUmElemento(S), de $S \leftarrow S \cup \{w\}$, e de Push($P, v$) é $\Theta(1)$. O que quer isso dizer? Justifique que: para DAGs com n vértices e m ramos, a complexidade do algoritmo apresentado é $\Theta(n + m)$, se $G.Adjs[v]$ for uma lista ligada.

(CONTINUA, v.p.f.)

3. Seja $G = (V, E)$ um grafo não dirigido finito, tal que $|V| = n$ e $|E| = m$, e os vértices são identificados por inteiros consecutivos a partir de 1. Pretendemos determinar *o número de vértices da componente conexa de G que tem mais vértices.*

a) Escreva (em pseudocódigo) um algoritmo com complexidade $O(m + n)$ que resolva o problema.

b) Justifique sucintamente a correção do algoritmo que apresentou.

(FIM)