

# Algoritmos e Complexidade

Exame de Recurso, MIEI, LCC 2º ano

2 de Fevereiro de 2019 – Duração: 2:30 h

1. Considere a definição de uma função que calcula o quadrado de um número natural (inteiro não negativo).  
De forma a provar a **correção parcial** da função apresentada

- (a) Complete a anotação do código apresentado, determinando um invariante  $I$  apropriado, e apresente as condições de verificação correspondentes.
- (b) Analise a complexidade desta função no pior caso (em que todos os  $N$  bits do argumento se encontram a 1) em função do número de bits necessários para representar o argumento.
- (c) Analise a complexidade média desta função assumindo que o argumento é um número aleatório e como tal, a probabilidade de cada um dos seus bits ser 1 é 0.5. Faça a análise contando apenas o número de adições.

```
int square (int x) {  
    // x==x0 && x>=0  
    int a, b;  
    a = x; b = a;  
    x = 0;  
    // I  
    while (a!=0) {  
        // I  
        if (a%2 != 0)  
            { x = x + b; }  
        a=a/2; b=b*2;  
    }  
    // x == x0^2  
    return x;  
}
```

2. Considere as definições ao lado para representar árvores AVL de inteiros e de uma função que substitui em cada nodo o factor de balanço pela altura da árvore que aí se inicia.

- (a) Admitindo que a função **altura** executa em tempo logaritmico no tamanho da árvore argumento, determine a complexidade da função apresentada. Admita que as árvores em questão são balanceadas.
- (b) Apresente uma definição alternativa da função **converte** que seja linear no número de elementos da árvore argumento.

```
typedef struct avlNode {  
    int valor;  
    int bal; //1:Esq, -1:Dir  
    struct avlNode *esq, *dir;  
} *AVL;  
  
void converte (AVL a) {  
    if (a!=NULL) {  
        a->bal = altura (a);  
        converte (a->esq);  
        converte (a->dir);  
    }  
}
```

3. Relembre o problema de determinar o fecho transitivo de um grafo, i.e., de dado um grafo não pesado  $G$  com  $N$  vértices, preencher uma matriz  $T[N][N]$  de tal forma que  $T[i][j] == 1$  sse existe em  $G$  um caminho com origem  $i$  e destino  $j$ .

- (a) Defina uma função **int travessia (Grafo g, int o, int v[])** que preenche o array  $v$  de forma a que  $v[i] == 1$  sse o vértice  $i$  é alcançável a partir de  $o$  (existe caminho desde  $o$  até  $i$ ).
- (b) Use a função definida na alínea anterior para definir a função **void FechoT(Grafo g, int T[N][N])** que preenche a matriz  $T$  com o fecho transitivo do grafo  $g$ .  
Admitindo que a função **travessia** tem um custo assintótico de  $\mathcal{O}(V + E)$  qual o custo assintótico da função **FechoT** que definiu?

4. Considere uma estrutura de dados sobre a qual é executada uma sequência de operações. É sabido que o tempo de execução da  $n$ -ésima operação da sequência é dado por  $T_{op}(n) = n$  quando  $n$  é uma potência de 2 (i.e.,  $n = 2^k$  para algum  $k \in \mathbb{N}$ ), e  $T_{op}(n) = 1$  para todos os outros valores de  $n$ .

Efectue a análise amortizada do tempo de execução de uma operação arbitrária desta sequência, utilizando para isso **duas técnicas** à sua escolha.