Métodos de Programação II

LESI / LMCC

21 de Junho 2004

Questão 1 [*Análise Assimptótica*]
Considere dois algoritmos A e B que resolvem o mesmo problema de tamanho N. O algoritmo A resolve o problema em N dias. O algoritmo B resolve o problema em N^2 segundos.

1. (a) Descreva o comportamento assimptótico destes algoritmos utilizando a notação Θ. De acordo com o que aprendeu nas aulas da disciplina, qual o algoritmo preferível em termos assimptóticos?
   (b) Qual o algoritmo que utilizaria, sabendo que o tamanho máximo para o input é N=10000?
   (c) Qual o tamanho necessário para que o algoritmo B demore metade do tempo de A a executar?
   (d) Com base na sua resposta às alíneas anteriores, retire conclusões quanto à utilização da análise assimptótica para a comparação da eficiência de algoritmos.
2. Explique a diferença entre análise assimptótica do pior caso e do caso médio do tempo de execução. Dê exemplos, justificando, de algoritmos que estudou para os quais:
   (a) A diferença entre o caso médio de execução e o pior caso de execução não depende do tamanho do input.
   (b) A diferença entre o caso médio de execução e o pior caso de execução cresce com o tamanho do input.

Nome:_____ **Número:**_____ **Curso:**_____

Métodos de Programação II

LESI / LMCC

21 de Junho 2004

Questão 2 [*Análise de Algoritmos*]
Considere o seguinte problema:

> *Dadas duas listas X e Y, não ordenadas, sem repetições, e com N elementos cada uma, encontrar os M maiores elementos de X que não aparecem em Y.*

1. Considere que X e Y são implementadas como listas ligadas utilizando o tipo `Lista` a seguir definido. Implemente a função `maiores` que resolve este problema.

```
typedef struct lista {
    int num;
    struct lista *next;
} Lista;

Lista * maiores(Lista *X, Lista *Y, int M);
```

2. Efectue uma análise assimptótica do tempo de execução da função que escreveu. No caso de a sua implementação ser menos eficiente do que $T(N) = \Theta(N \log(N))$, explique como poderia atingir este nível de eficiência.

Nome:_____ **Número:**_____ **Curso:**_____

Métodos de Programação II

LESI / LMCC

21 de Junho 2004

Questão 3 Assuma a seguinte definição de tipos para árvores binárias:

```
typedef struct node {
  int elem;
  struct node *esq;
  struct node *dir;
} Node, *Tree;
```

1. Diga o que entende por árvore balanceada AVL, e implemente a função recursiva que testa se uma dada árvore binária satisfaz esses requisitos.

   ```
   int balanced(Tree root);
   ```

   N.B: Deverá escrever também as funções auxiliares de que necessitar.

2. Escreva uma equação de recorrência, analise o tempo de execução da função `balanced`, e exprima esse comportamento utilizando a notação Θ.

3. Considere agora a função *min* que devolve na variável `x` o elemento de menor valor de uma árvore binária de pesquisa, retornando um código de sucesso/insucesso. Escreva um invariante para o ciclo da função e utilize-o para demonstrar a correcção do algoritmo.

   ```
   int min(Tree t, int *x) {
      if (t) {
         while (t->esq)
            t=t->esq;
         *x=t->elem;
         return 0; // resultado encontrado
      }
      else return -1; // erro
   }
   ```

Nome:_____ **Número:**_____ **Curso:**_____

Métodos de Programação II

LESI / LMCC

21 de Junho 2004

4

Questão 4 [*Algoritmos de Grafos*]
Considere o problema da construção de uma *ordenação topológica* para um grafo ordenado acíclico. Considere ainda o grafo

$$G = (\{1,2,3,4,5,6,7,8,9\}, \{(1,2),(2,3),(2,5),(3,6),(4,5),(5,6),(5,8),(6,9),(7,8),(8,9)\})$$

1. Explique o funcionamento do algoritmo que resolve este problema com base numa travessia em profundidade; simule a execução desse algoritmo para o grafo G.

2. Repita a alínea anterior para o algoritmo que resolve o problema com base numa travessia em largura.

   Sugestão: Nesta segunda alínea, utilize um vector auxiliar onde se guardará, para cada nó do grafo, o número dos seus antecessores que ainda não estão incluídos na ordenação topológica (inicializado com o grau de entrada de cada nó). Esta informação permitirá decidir quando inserir um nó na fila de espera.

Nome:_____ **Número:**_____ **Curso:**_____

Métodos de Programação II

LESI / LMCC

21 de Junho 2004

Questão 5 [*Problemas NP-completos*]
Considere o seguinte algoritmo que produz uma coloração de um grafo $G = (V, E)$, com $V = \{1, 2, \ldots n\}$ (as cores são números inteiros positivos).

```
for (i=1 ; i<=n ; i++) {
  c = 1;
  while  (existir um nó j adjacente a i tal que j está pintado com a cor c)
    c++;
  pintar o nó i com a cor c;
}
```

Este algoritmo pinta cada nó i com a menor cor aceitável, isto é, a menor cor ainda não atribuída a nenhum nó adjacente a i.

Diga, justificando, se este algoritmo produz uma solução óptima, i.e., se o número de cores utilizado é ou não sempre o mínimo possível.

Nome:_____ **Número:**_____ **Curso:**_____