Exame — MAC5811-1 – Projeto e Análise de Algoritmos

8/8/2006

Instruções:

(*i*) O candidato pode resolver todas as questões.

(*ii*) A banca considerará questões cujos valores somem até 10 pontos de modo que a soma total das notas obtidas seja máxima. Um aluno, para ser aprovado, precisa obter nessas questões pelo menos 7 pontos.

(*iii*) Algoritmos devem ser escritos em pseudocódigo ou de forma informal usando algoritmos bem conhecidos como subrotinas.

(*iv*) Dê justificativas sucintas para suas respostas e explique sempre porque cada algoritmo funciona e estime sua complexidade.

(*v*) A duração da prova é 5 horas.

Questão 1 [1 ponto]
Para cada inteiro positivo i, sejam $f_i(x)$ e $g_i(x)$ funções tais que $f_i(x) = O(g_i(x))$. Defina as funções $F(n)$ e $G(n)$ por $F(n) = \sum_{i=1}^{n} f_i(n)$ e $G(n) = \sum_{i=1}^{n} g_i(n)$. É verdade que $F(n) = O(G(n))$?

Questão 2 [1 ponto]
Considere o seguinte problema.
ENTRADA: inteiros positivos n e k.
SAÍDA: n^k.

1. Qual o tamanho da entrada como função de n e k?
2. Considere o seguinte algoritmo para resolver o problema.
   produto := 1;
   para $i := 1$ até k faça
   produto := produto $\cdot\, n$;

   Assuma que o algoritmo da escola primária é usado para fazer as multiplicações. Para multiplicar um número com x dígitos por um número com y dígitos o algoritmo da escola primária usa $\Theta(x \cdot y)$ passos. Estime da melhor forma possível o tempo de processamento do algoritmo acima.
3. Explique porque o algoritmo não é limitado polinomialmente no tamanho da entrada.

Questão 3 [1 ponto]
Considere a sequência $T(n)$ tal que:

- $T(0) = 0$
- $T(n) = T(\lfloor n - \sqrt{n} \rfloor) + 1$.

Mostre que $T(n)$ é $\Theta(\sqrt{n})$.

Questão 4 [1 ponto]
P=NP se e só se algum problema em P é NP-completo.
Essa afirmativa é verdadeira ou falsa? Justifique.

Questão 5 [1 ponto]
Considere o seguinte algoritmo de busca, que determina se x está ou não no vetor $v[1 \dots n]$. O vetor $b[1 \dots n]$, usado no algoritmo, indica se um índice i já foi ou não examinado na busca e a variável c indica quantos índices diferentes de v já foram examinados. A função RAND(n) devolve um inteiro entre 1 e n, escolhido com probabilidade uniforme.

```
BUSCA_ALEATÓRIA(v, n, x)
para i ← 1 até n faça
    b[i] ← 0
c ← 0
i ← RAND(n)
enquanto c < n e v[i] ≠ x faça
    se b[i] = 0
        então b[i] ← 1
              c ← c + 1
    i ← RAND(n)
se v[i] = x
    então devolva VERDADEIRO
    senão devolva FALSO
```

Qual é o número esperado de iterações do enquanto caso o elemento x apareça exatamente uma vez no vetor?

Questão 6 [2 pontos]
Considere a sequência $T(n)$ tal que:

- $T(1) = 1$
- $T(n) = 2T(\lfloor n/2 \rfloor) + \lfloor n \lg n \rfloor$.

Mostre que $T(n)$ não é $O(n \lg n)$.

Questão 7 [2 pontos]
Para uma árvore de busca binária (ABB), e um item x armazenado nela, seja $N(x)$ o nó da árvore contendo x. Dizemos que uma ABB A é *embutível* em uma ABB B se:

- Todo item armazenado em A também está em B, e
- Para todos os pares x, y de itens, se $N(x)$ é descendente de $N(y)$ em A, então $N(x)$ é descendente de $N(y)$ em B.

Dê um algoritmo que, dadas A e B, decide em tempo linear se A é embutível em B.

Questão 8 [3 pontos]
Uma *ordem parcial* em um conjunto é uma relação binária, denotada usualmente por $\prec$, tal que *(i)* $a \not\prec a$ para cada a do conjunto e *(ii)* se a, b e c são tais que $a \prec b$ e $b \prec c$, então $a \prec c$. Se a, b são tais que nem $a \prec b$ nem $b \prec a$, eles são *incomparáveis*. Um vetor $A[1 \ldots n]$ de elementos dessa ordem *estende* a ordem se, para todo i, j, se $A[i] \prec A[j]$, então $i < j$.

Suponha definido um tipo `Op`, e dada uma função `compara(Op x, Op y)`, que devolve:

`'<'` se $\mathtt{x} \prec \mathtt{y}$

`'>'` se $\mathtt{y} \prec \mathtt{x}$

`'|'` se `x` e `y` são incomparáveis.

Considere o problema: dado um vetor $A[1 \ldots n]$ do tipo `Op`, reordenar esse vetor de modo a estender a ordem.

Mostre que qualquer algoritmo que resolva esse problema usa $\Omega(n^2)$ chamadas de `compara`, no pior caso.

Descreva um algoritmo que faz $O(n^2)$ chamadas de `compara`.

Questão 9 [3 pontos]
Dado um vetor $A[1 \ldots n]$ de strings distintos, definimos, para $i = 1, \ldots, n$

$p(i) =$ o número de índices j tais que $A[j] \leq A[i]$ (na ordem lexicográfica),

$\delta(i) = i - p(i)$.

Dê algoritmos baseados em comparações para, dado A, calcular:

1. $\delta_A = \sum_{i=1}^{n} \delta(i)$

2. $\bar{\delta}_A = \sum_{i=1}^{n} \mid \delta(i) \mid$

Os algoritmos devem ser $o(n^2)$, e um deles deve ser muito mais rápido que o outro.