Exame Preliminar para o Doutorado

IME–USP, Agosto, 1997

Prova de Análise de Algoritmos

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.

(iii) Mencione os teoremas e propriedades usados para justificar suas afirmações.

Questão 1 [1 ponto]
Suponha que estamos procurando pela chave 363 numa árvore binária de busca cujas chaves são números naturais entre 1 e 1000. Durante a busca, uma certa seqüência de chaves é examinada. Quais das seguintes seqüências são impossíveis? Justifique.

(a) 2, 399, 387, 219, 266, 382, 381, 278, 363.

(b) 935, 278, 347, 621, 299, 392, 358, 363.

Questão 2 [1 ponto]

(1) Defina um *heap*. Explique como um *heap* pode ser representado num vetor.

(2) Rearranje o vetor abaixo de modo que ele seja um *heap*.

8	4	1	2	16	14	9	10	3	7

Questão 3 [1 ponto]
Considere o seguinte problema:

- *dado um conjunto S de números naturais (distintos dois-a-dois) e um número natural x, decidir se S contém dois elementos s_1 e s_2 tais que $s_1 + s_2 = x$.*

Esboce um algoritmo que resolva o problema em tempo $O(n \log n)$ no pior caso, onde $n = |S|$. Justifique.

Sugestão: Ordene o conjunto.

Questão 4 [1 ponto]
Mostre que

$$\sum_{k=1}^{n} k^{99} = \Theta(n^{100}).$$

Questão 5 [2 pontos]
Considere o seguinte algoritmo, cujo argumento n é uma potência de 2. (O algoritmo não faz nada de útil.)

ALGO(n)
{
se $n \leq 1$ *então devolva* 1;
para $i \leftarrow 1$ *até* 8 *faça* $z \leftarrow$ ALGO($n/2$);
para $i \leftarrow 1$ *até* n^3 *faça* $z \leftarrow 0$;
}

(1) Seja $T(n)$ o número de vezes que a atribuição "$z \leftarrow 0$" é executada. Escreva uma recorrência que define $T(n)$.

(2) Mostre diretamente (sem usar o Master Theorem) que $T(n)$ é $\Omega(n^3 \log n)$.

(3) Troque "8" por "7" no algoritmo e mostre diretamente que $T(n)$ é $O(n^3)$.

Questão 6 [2 pontos]
Digamos que um *ponto* é um certo tipo de objeto (a natureza exata desses objetos é irrelevante). Suponha que f é uma função que leva ternos ordenados de pontos em números racionais.

Suponha que $x_1, \ldots, x_n$ é uma seqüência de pontos tal que $f(x_1, x_2, x_i) \geq 0$ para todo $i \geq 3$.

O seguinte algoritmo recebe $x_1, \ldots, x_n$ e devolve uma subseqüência $p_1, \ldots, p_t$:

ALGO($x_1, \ldots, x_n$)
{
$p_1 \leftarrow x_1$;
$p_2 \leftarrow x_2$;
$t \leftarrow 2$;
para $i \leftarrow 3$ *até* n *faça* {
enquanto $f(p_{t-1}, p_t, x_i) < 0$ *faça* $t \leftarrow t - 1$;
$p_{t+1} \leftarrow x_i$;
$t \leftarrow t + 1$;
}
devolva $p_1, \ldots, p_t$;
}

Mostre que o algoritmo consome tempo $O(n)$ no pior caso (supondo que o cálculo de $f(u, v, w)$ consome tempo $O(1)$).

Sugestão: Note que $p_1, \ldots, p_t$ funciona como uma pilha.

Questão 7 [3 pontos]
Suponha que T é uma árvore (isto é, um grafo conexo sem circuitos) com n nós. Para quaisquer dois vértices x e y de T, seja $Q(x, y)$ o seguinte problema:

$Q(x, y)$: *encontrar o caminho em* T *que liga* x *a* y.

(1) Elabore uma estrutura de dados para a árvore T que permita resolver $Q(x,y)$ em tempo $O(d)$, onde d é a distância de x a y em T.

(2) Esboce um algoritmo que resolva $Q(x,y)$ em tempo $O(d)$. Justifique.

(3) Sugira como montar a sua estrutura em tempo $O(n)$. Justifique.

Sugestões: (i) Note que, no item (1), pede-se $O(d)$ e portanto $O(n)$ não basta. (ii) Para construir a estrutura, faça uma busca em largura a partir de um nó arbitrário.

Questão 8 [3 pontos]
Sejam $v_1, \ldots, v_n$, $p_1, \ldots, p_n$ e P números naturais.

Problema (da mochila): Encontrar $x_1, \ldots, x_n$ *em* $\{0,1\}$ *tais que*

$$p_1 x_1 + \cdots + p_n x_n \leq P$$

e $v_1 x_1 + \cdots + v_n x_n$ *é máximo.*

O *algoritmo guloso* consiste no seguinte:

- ajuste a notação de modo que
$$\frac{v_1}{p_1} \geq \frac{v_2}{p_2} \geq \cdots \geq \frac{v_n}{p_n}; \qquad (*)$$
- para $k = 1, \ldots, n$ faça
  se $p_1 x_1 + \cdots + p_{k-1} x_{k-1} + p_k \leq P$
  então $x_k \leftarrow 1$
  senão $x_k \leftarrow 0$

(1) Mostre que o algoritmo guloso, em geral, não resolve o Problema.

(2) Suponha que $p_i \leq P$ para todo i. Mostre que o vetor $x_1, \ldots, x_n$ calculado pelo algoritmo guloso é uma "solução aproximada" do Problema no seguinte sentido:

$$\begin{cases} v_1 x_1 + \cdots + v_n x_n \geq \frac{1}{2} V \\ \qquad\qquad \text{ou} \\ \max\{v_1, \ldots, v_n\} \geq \frac{1}{2} V, \end{cases}$$

onde $V = v_1 x_1^* + \cdots + v_n x_n^*$ e $x_1^*, \ldots, x_n^*$ é qualquer solução (ótima) do Problema.

Sugestão: Como se sabe, a versão relaxada do Problema (isto é, a versão que obtemos ao substituir "$x_i \in \{0,1\}$" por "$0 \leq x_i \leq 1$") tem como solução (ótima) o vetor $x_1, \ldots, x_n$ definido pela seguinte variante do algoritmo guloso acima: supondo novamente que (*) vale, seja k o maior índice tal que

$$p_1 + \cdots + p_k \leq P;$$

então $x_1 = \cdots = x_k = 1$,

$$x_{k+1} = \frac{1}{p_{k+1}} \left\{ P - \sum_{i=1}^{k} p_i \right\},$$

e $x_\ell = 0$ para todo $\ell > k+1$. Use este fato.