Estrutura de Dados - 2o. período de 2006

Gabarito da Segunda Avaliação à Distância

1. (2,0) Descreva um algoritmo que imprima os nós de uma árvore binária na ordem de suas alturas. Isto é, primeiro são impressos os nós de altura 1, depois os de altura 2, e assim por diante. Veja que há várias soluções para este problema.

   Resposta:

   Seja h a altura da árvore dada. O algoritmo a seguir percorre a árvore $h-1$ vezes, imprimindo e removendo suas folhas (enquanto a raiz não for uma folha). No procedimento busca-folhas, é utilizada uma variável chamada *lado*, configurada com 0 quando o nó é filho esquerdo de seu pai, ou com 1, quando é filho direito.

   enquanto $(ptraiz \uparrow .esq \neq \lambda)$ ou $(ptraiz \uparrow .dir \neq \lambda)$ faça
   se $(ptraiz \uparrow .esq \neq \lambda)$ então
   busca-folhas$(ptraiz \uparrow .esq, ptraiz, 0)$
   se $(ptraiz \uparrow .dir \neq \lambda)$ então
   busca-folhas$(ptraiz \uparrow .dir, ptraiz, 1)$
   $imprimir(ptraiz \uparrow .info)$

   procedimento busca-folhas$(no, pai, lado)$
   se $(no \uparrow .esq = \lambda)$ e $(no \uparrow .dir = \lambda)$ então
   se $(lado = 0)$ então $pai \uparrow .esq = \lambda$
   senão $pai \uparrow .dir = \lambda$
   $imprimir(no)$
   $desocupar(no)$
   senão
   se $(no \uparrow .esq \neq \lambda)$ então busca-folhas$(no \uparrow .esq, no, 0)$
   se $(no \uparrow .dir \neq \lambda)$ então busca-folhas$(no \uparrow .dir, no, 1)$

2. (1,0) Prove ou dê contra-exemplo: Uma árvore binária pode ser construída, de forma única, a partir das seguintes informações: (i) percurso em nível e (ii) número de nós em cada subárvore.

   Reposta:

   A afirmação é falsa. Considere duas árvores binárias T_1 e T_2, onde cada uma delas contém apenas dois nós A e B de forma que:
   - em T_1, B é filho esquerdo de A;
   - em T_2, B é filho direito de A.

Para ambas as árvore acima, o percurso em nível é AB, A possui 2 nós em sua subárvore e B possui um nó. No entanto, elas são distintas.

3. (2,0) Uma árvore ternária é definida como uma árvore em que cada nó tem no máximo três filhos. Escreva um algoritmo que calcule as alturas de todos os nós de uma árvore ternária.

   Resposta:

   Sejam *esq*, *cen* e *dir* os ponteiros para os filhos esquerdo, central e direito, respectivamente, de um dado nó.

   procedimento altura(pt)

   se ($pt \uparrow .esq \neq \lambda$) então $h_1 := altura(pt \uparrow .esq)$

   senão $h_1 := 0$

   se ($pt \uparrow .cen \neq \lambda$) então $h_2 := altura(pt \uparrow .cen)$

   senão $h_2 := 0$

   se ($pt \uparrow .dir \neq \lambda$) então $h_3 := altura(pt \uparrow .dir)$

   senão $h_3 := 0$

   retornar $1 + max\{h_1, h_2, h_3\}$

   Chamada externa: $altura(ptraiz)$

4. (2,0) Determinar a árvore binária de custo mínimo relativa às seguintes freqüências: $f_1 = 1$, $f_2 = 0$, $f_3 = 2$, $f'_0 = 0$, $f'_1 = 3$, $f'_2 = 1$, $f'_3 = 2$.

   Resposta:

   As matrizes do algoritmo de cálculo da árvore ótima são:

   Matriz dos custos $c[i, j]$:

0	4	9	18
-	0	4	12
-	-	0	5
-	-	-	0

   Matriz dos valores $F[i, j]$:

0	4	5	9
-	3	4	8
-	-	1	5
-	-	-	2

   Matriz dos valores minimizantes k:

-	1	1 (2)	2 (3)
-	-	2	3
-	-	-	3
-	-	-	-

Da última matriz acima, obtemos três possíveis árvores ótimas:
- raiz s_2, filho esquerdo s_1 e direito s_3;
- raiz s_3, sendo s_1 filho esquerdo de s_3, e s_2 filho direito de s_1;
- raiz s_3, sendo s_2 filho esquerdo de s_3, e s_1 filho esquerdo de s_2.

5. (1,0) Prove ou dê contra-exemplo: Dado n inteiro positivo, sempre existe uma árvore B com exatamente n nós.

   Resposta:

   A afirmação é falsa. Não existe árvore B com 2 nós, por exemplo, já que a raiz de uma árvore B é folha ou tem pelo menos 2 filhos.

6. (2,0) Descreva como são as árvores rubro-negras com número máximo de nós rubros.

   Resposta:

   Os caminhos raiz-folha mais longos têm os nós internos alternadamente rubros e negros, já que o pai de um nó rubro deve ser negro. Sendo h o total de nós internos em um caminho raiz-folha mais longo, este caminho terá $\lfloor n/2 \rfloor$ nós negros (já que a raiz pode ser rubra). Logo, os demais caminhos terão nós rubros (além da raiz) somente se seu total de nós for maior que $\lfloor n/2 \rfloor$. Cada caminho com total de nós $k > \lfloor n/2 \rfloor$ nós terá portanto $k - \lfloor n/2 \rfloor$ nós rubros.