Estrutura de Dados - 2o. período de 2006

Gabarito da Primeira Avaliação à Distância

1. Para cada item abaixo, responda "certo" ou "errado", justificando (meio ponto cada):

   a. Se a complexidade de caso médio de um algoritmo for $O(f)$, então o número de passos que o algoritmo efetua no pior caso é $\Omega(f)$.

   Resposta: errado. Se a função f limitar superiormente também o pior caso, o número de passos que o algoritmo efetua no pior caso é $O(f)$, e não $\Omega(f)$.

   b. Se a complexidade de melhor caso de um algoritmo for $\Theta(f)$, então o número de passos que o algoritmo efetua, qualquer que seja a entrada, é $\Theta(f)$.

   Resposta: errado. Se a complexidade de melhor caso de um algoritmo for $\Theta(f)$, então o número de passos que o algoritmo efetua, qualquer que seja a entrada, é $\Omega(f)$.

   c. Se a complexidade de pior caso de um algoritmo for $O(f)$, então o número de passos que o algoritmo efetua, qualquer que seja a entrada, é $O(f)$.

   Resposta: certo. Se a complexidade de pior caso de um algoritmo é limitada superiormente por uma função f, então esta função limita superiormente também qualquer entrada para este algoritmo.

   d. A complexidade de pior caso de um algoritmo para um certo problema P é necessariamente maior do que um limite inferior para P.

   Resposta: Se por "maior" entende-se "estritamente maior", isto é, que a complexidade de pior caso é $\Omega(f)$ mas não $\Theta(f)$, onde f é a função de limite inferior, então a afirmativa está **errada**, pois a complexidade de pior caso pode "empatar" com o limite inferior.

   Mas se por "maior" entende-se apenas que a complexidade de pior caso é $\Omega(f)$, então a afirmativa está **certa**, porque a definição de limite inferior diz justamente que a complexidade de pior caso de **qualquer** algoritmo para o problema é $\Omega(f)$.

   e. Se um algoritmo A para um problema P não é ótimo, então existe um algoritmo A' para P cuja complexidade de pior caso é inferior à complexidade de pior caso de A.

   Resposta: certa. Um algoritmo é ótimo quando sua complexidade de pior caso é igual ao limite inferior para o problema. Se o algoritmo A não é ótimo, então é possível construir um algoritmo A' cuja complexidade de pior caso se aproxime mais do limite inferior, sendo, portanto, inferior à complexidade de pior caso de A.

2. (1,0) Considere uma sequência de elementos $f_1, f_2, \ldots, f_n$, definida do seguinte modo: $f_1 = 0$, $f_2 = 1$, $f_3 = 2$, $f_j = f_{j-1} - f_{j-2} + f_{j-3}$ para $j > 3$. Elaborar um algoritmo, não recursivo, para determinar o elemento f_n da sequência, cuja complexidade seja linear em n.

   Resposta:

$f[1] := 0;$

$f[2] := 1;$

$f[3] := 2;$ para $j = 4 \ldots n$ faça $f[j] := f[j-1] - f[j-2] + f[j-3];$

A complexidade do algoritmo acima é $O(n)$.

3. (1,5) Escreva um algoritmo que inverte uma lista simplesmente encadeada com nó cabeça. Exemplo: se a lista contém os elementos $1, 3, 5, 9, 2$, nesta ordem, então a lista resultante contém os elementos $2, 9, 5, 3, 1$. Você deve utilizar uma pilha auxiliar para resolver este problema.

   Resposta:

   $pont := ptLista \uparrow .prox$

   $topo := 0$ % inicialmente a pilha P está vazia

   enquanto $pont \neq \lambda$ faça

       $topo := topo + 1$

       $P[topo] := pont \uparrow .info$ % isere o dado em P

       $pont := pont \uparrow .prox$

   ocupar(L) % cria a lista resultante L

   $L \uparrow .prox := \lambda$

   $ultimo := L$

   enquanto $topo > 0$ faça % remove dados da pilha e os insere em L

       ocupar(pt)

       $pt \uparrow .info := P[topo]$

       $pt \uparrow .prox := \lambda$

       $ultimo \uparrow .prox := pt$

       $ultimo := pt$

       $topo := topo - 1$

4. (2,5) Sejam duas listas L_1 e L_2 simplesmente encadeadas com nó-cabeça. Apresentar um algoritmo que intercale as duas listas de forma que a lista resultante L contenha elementos de L_1 e L_2 alternadamente, da seguinte forma: o primeiro elemento de L é o primeiro elemento de L_1, o segundo elemento de L é o primeiro elemento de L_2, o terceiro elemento de L é o segundo elemento de L_1, e assim por diante. Obs: Os elementos excedentes da lista mais longa entre L_1 e L_2 devem ser colocados diretamente no final da lista L, obedecendo a ordem em que aparecem originalmente.

   Resposta:

$pont1 := L_1 \uparrow .prox$ % ponteiro para a lista L_1

$pont2 := L_2 \uparrow .prox$ % ponteiro para a lista L_2

$L := L_1$ % a lista resultante iniciará em L_1

$i := 1$

enquanto $pont1 \neq \lambda$ e $pont2 \neq \lambda$ faça

se i é ímpar então

$aux := pont1 \uparrow .prox$

$pont1 \uparrow .prox := pont2$

$pont1 := aux$

senão

$aux := pont2 \uparrow .prox$

$pont2 \uparrow .prox := pont1$

$pont2 := aux$

$i := i + 1$

5. (2,5) Determinar a expressão da complexidade média de uma busca não ordenada de n chaves, n par, em que as probabilidades de busca das chaves 1 a $n/2$ são iguais entre si, sendo esse valor igual ao triplo da probabilidade de qualquer chave entre $n/2 + 1$ a n. Supor, ainda, que a probabilidade de a chave se encontrar na lista é igual a 60%.

Resposta:

Seja p a probabilidade de cada chave dentre 1 e $n/2$. Temos então que a probabilidade de cada chave dentre $(n/2 + 1)$ e n é $p/3$. Distribuindo a probabilidade de 60% pelas n chaves, temos que

$$(n/2)p + (n/2)(p/3) = 0,6$$

Concluímos que $p = 0,9/n$. Logo, a expressão da complexidade média neste caso é dada pela expressão:

$$\begin{aligned} C.M. &= \frac{0,9}{n}\sum_{i=1}^{\frac{n}{2}} i \ + \ \frac{0,3}{n}\sum_{i=\frac{n}{2}+1}^{n} i \ + \ 0,4n \\ &= \frac{0,9}{n} \cdot \frac{(n/2)(1+n/2)}{2} + \frac{0,3}{n} \cdot \frac{(n/2)(n/2+1+n)}{2} \\ &= \frac{5n+2,4}{8} \end{aligned}$$