Estrutura de Dados - 1o. período de 2007

Gabarito da Primeira Avaliação à Distância

1. Escrever as seguintes funções em notação O:
   $n^2 - n^3$; $10n + 2\log^2 n$; $8^n - 16n!$; $(n+1)^n - n^{n+1}$; 2.5.

   Resposta: $n^2 - n^3 = O(n^2)$; $10n + 2\log^2 n = O(n)$; $8^n - 16n! = O(8^n)$; $(n+1)^n - n^{n+1} = O(n^n)$; $2.5 = O(1)$.

2. Para cada item abaixo, responda "certo", "errado" ou "nada se pode concluir". Justifique.

   a. Se um limite inferior para um problema P é n^2, então existe um algoritmo ótimo para P de complexidade de pior caso $\Theta(n^2)$.

   Resposta: Nada se pode concluir. O fato de um limite inferior conhecido para P ser n^2 não significa que não exista limite inferior maior para P (por exemplo, pode ainda não ter sido provado um limite inferior $n^{2.3}$ para P). Logo, se o limite inferior for na verdade maior, então não é possível construir um algoritmo com complexidade de pior caso $\Theta(n^2)$.

   b. Se um limite inferior para um problema P é n^2, então nenhum algoritmo ótimo para P pode ter complexidade de pior caso $O(n \log n)$.

   Resposta: Certo. A complexidade de pior caso de um algoritmo ótimo para P é $\Omega(n^2)$, logo não pode ser $O(n \log n)$.

   c. Se um limite inferior para um problema P é n^2, então todo algoritmo ótimo para P tem complexidade de pior caso $\Omega(n^2)$.

   Resposta: Certo. Pela definição de limite inferior, a complexidade de pior caso de qualquer algoritmo que resolva P é $\Omega(n^2)$.

3. No instante $t = 0$, uma cultura de bactérias contém 8×10^6 indivíduos. No instante $t = i$ (sendo i um inteiro positivo que expressa o número de horas), o número de indivíduos na cultura é o dobro do número de indivíduos no instante anterior menos 7×10^3. Escreva dois algoritmos, um recursivo e outro não-recursivo, que calculem o número de indivíduos presentes na cultura no instante i. Calcule as complexidades dos algoritmos.

   Resposta:

   Algoritmo recursivo:

   função $ind(j)$
   se $j = 0$ então
   $ind(j) := 8 \times 10^6$
   senão
   $ind(j) := 2\, ind(j-1) - 7 \times 10^3$

   Chamada externa: $ind(i)$

Complexidade: Seja $T(j)$ o número de passos do algoritmo no instante $t = j$. Temos:

$T(0) = 1$

$T(j) = T(j-1) + 1$

Resolvendo esta recorrência, verificamos que a complexidade deste algoritmo é $\Theta(i)$.

Algoritmo não recursivo:

$ind[0] := 8 \times 10^6$
para $j = 1, \cdots, i$ faça
$ind[j] := 2 \times ind[j-1] - 7 \times 10^3$

Complexidade: Este algoritmo possui apenas um loop que realiza exatamente i iterações, e em cada uma delas apenas um comando é executado. Logo, sua complexidade é $\Theta(i)$.

4. Determinar a expressão da complexidade média de uma busca não ordenada de n chaves, em que a probabilidade de busca da chave i é o triplo da probabilidade de busca da chave $i-1$, para $i = 2, \cdots, n$. Supor, ainda, que a probabilidade de a chave procurada se encontrar na lista é igual a 50%.

   Resposta:

   Seja p a probabilidade de busca da chave 1. Temos então que $p+3p+9p+\cdots+3^{n-1}p = 0,5$. Logo,

$$\begin{aligned} p\sum_{i=1}^{n} 3^{i-1} &= \frac{1}{2} \\ p\left(\frac{3^n - 1}{2}\right) &= \frac{1}{2} \\ p &= \frac{1}{3^n - 1} \end{aligned}$$

   A expressão da complexidade média neste caso é dada por:

$$\begin{aligned} C.M. &= \sum_{i=1}^{n}(t_i \,.\, p_i) + 0,5n \\ &= \sum_{i=1}^{n}\left(i \,.\, \frac{3^{i-1}}{3^n - 1}\right) + 0,5n \\ &= \frac{1}{3^n - 1}\sum_{i=1}^{n}(i \,.\, 3^{i-1}) + 0,5n \end{aligned}$$

5. Escrever algoritmos de busca, inserção e remoção em LISTAS CIRCULARES ENCADEADAS NÃO ORDENADAS.

   Resposta:

Busca:

procedimento $busca(x, ant, pont)$
    $ant := ptlista$
    $pont := ptlista \uparrow .prox$
    enquanto $(pont \neq ptlista)$ e $(pont \uparrow .chave \neq x)$ faça
        $ant := pont$
        $pont := pont \uparrow .prox$
    se $pont \neq ptlista$ então
        "chave localizada"
    senão *"chave não localizada"*

Inserção:

$busca(x, ant, pont)$
se $pont \neq ptlista$ então
    "elemento já existe na lista"
senão
    $ocupar(pt)$
    $pt \uparrow .info :=$ novo_valor
    $pt \uparrow .chave := x$
    $pt \uparrow .prox := ant \uparrow .prox$
    $ant \uparrow .prox := pt$

Remoção:

$busca(x, ant, pont)$
se $pont = ptlista$ então
    "elemento não existe na lista"
senão
    $ant \uparrow .prox := pont \uparrow .prox$
    $desocupar(pont)$

6. Sejam L_1 e L_2 duas listas ordenadas, simplesmente encadeadas com nó-cabeça. Apresentar um algoritmo que construa uma lista ordenada contendo os elementos que pertencem exclusivamente a L_2. (Isto é, aqueles elementos que pertencem a L_2 mas não a L_1.)

   Resposta:

   Algoritmo:

   $ocupar(ptlista3)$ % ponteiro para a nova lista L_3
   $ptlista3 \uparrow .prox := \lambda$
   $pont1 := ptlista1 \uparrow .prox$ % ponteiro para a lista L_1
   $pont2 := ptlista2 \uparrow .prox$ % ponteiro para a lista L_2
   $ptaux := ptlista3$

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

se $pont1 \uparrow .info = pont2 \uparrow .info$ então

$pont1 := pont1 \uparrow .prox$

$pont2 := pont2 \uparrow .prox$

senão

se $pont1 \uparrow .info < pont2 \uparrow .info$ então

$pont1 := pont1 \uparrow .prox$

senão % o elemento pertence exclusivamente a L_2

$incluir\_no(pont2, ptaux)$

enquanto $pont2 \neq \lambda$ faça

$incluir\_no(pont2, ptaux)$

procedimento $incluir\_no(pont2, ptaux)$

$ocupar(pt)$

$pt \uparrow .info := pont2 \uparrow .info$

$pt \uparrow .prox := \lambda$

$ptaux \uparrow .prox := pt$

$ptaux := pt$

$pont2 := pont2 \uparrow .prox$

7. Escreva uma versão NÃO RECURSIVA do Algoritmo das Torres de Hanói que se encontra no livro-texto. Sugestão: utilize pilhas!

   CANCELADA

8. Um lava-rápido tem capacidade máxima de atendimento de 5 carros (um que está sendo lavado, e quatro em espera). Cada lavagem leva 3 minutos. Ao chegar um novo cliente, o sistema ou o atende imediatamente (caso esteja completamente livre), ou o coloca em espera, ou o rejeita (caso já existam 5 carros sendo atendidos). Escreva um algoritmo que leia um inteiro $n \geq 1$ e um vetor binário (por exemplo, 001011100111000111), onde o i-ésimo bit da esquerda para a direita vale "1" se um novo cliente chega no i-ésimo minuto, e "0" se nenhum novo cliente chega no i-ésimo minuto. A seguir, o algoritmo deve calcular quantos carros foram lavados pelo sistema até o n-ésimo minuto. (Suponha que o vetor tem pelo menos n bits.) Use uma fila para representar o sistema a cada minuto que passa.

   Resposta:

   O algoritmo proposto recebe um inteiro n e um vetor binário S com pelo menos n bits, e utiliza uma fila circular F com 5 posições, que representa o sistema a cada minuto.

F está vazia quando os ponteiros ini e fim, que apontam para o início e o final de F, respectivamente, valem 0. Se $ini \neq 0$, então há algum carro sendo lavado. Ao final, a variável *ncarros* armazena o total de carros que foram lavados até o n-ésimo minuto.

```
procedimento lava-carros(n, S)
    ini := 0
    fim := 0
    ncarros := 0
    inicio := 0                    % armazena o minuto em que um carro começou a ser lavado
    para i = 1 ··· n faça
        se S[i] = 1 então
            se ini = 0 então                % não há nenhum carro sendo lavado
                ini := 1
                fim := 1
                F[fim] := 1
                inicio := i                 % a lavagem do carro é iniciada
            senão
                se ini ≠ (fim mod 5) + 1 então          % F não está cheia
                    fim := (fim mod 5) + 1
                    F[fim] := 1
                senão "carro rejeitado"                 % F está cheia
        se ini ≠ 0 então                    % algum carro está sendo lavado
            se i = inicio + 2 então         % sua lavagem termina no fim do minuto i
                F[ini] := 0                 % termina a lavagem do carro atual
                ncarros := ncarros + 1
                se ini ≠ fim então          % há carros na fila
                    inicio := i+1           % a lavagem do próximo carro se inicia no minuto seguinte
                    ini := (ini mod 5) + 1  % ini aponta para o carro que será lavado
                senão                       % não há carros na fila
                    ini := 0
                    fim := 0
    imprimir ("Total de carros lavados" + ncarros )
```