Projetos e Análise de Algoritmos – 2024/2 – Prova I (25 pontos)
Prof. Marcio Costa Santos

INSTRUÇÕES

(a) As respostas devem estar na prova no espaço designado para tal.
(b) A interpretação das questões faz parte da prova. Explique as suposições que fizer.
(c) Não é permitido o uso de material de consulta.

Nome:

Matrícula:

Questão 1 (5 pontos) (Complexidade e Notação Assintótica) Para cada um dos itens abaixo, apresente um algoritmo que recebe um vetor e seu tamanho como entrada e satisfazem exatamente as complexidades esperadas. Não é preciso explicar seu algoritmo.Cuidado com o uso de operações matemáticas não elementares(que não seja: +,-, x, resto ou /)!

i. Complexidade de pior caso de $\Theta(n^3 + n \log n)$

   Observe que $\Theta(n^3 + n \log n) = \Theta(n^3)$. Então basta fazer um algoritmo com 3 laços aninhados que soma 1 em cada posição do vetor, por exemplo.

ii. Complexidade de melhor caso de $\Theta(n + n^2 + \log n)$

   Observe que $\Theta(n + n^2 + \log n) = \Theta(n^2)$. Então basta fazer um algoritmo com 2 laços aninhados que soma 1 em cada posição do vetor, por exemplo.

Questão 2 (5 pontos) (Complexidade e Notação Assintótica) Indique verdadeiro ou falso para as afirmações abaixo. Não é necessário justificar a sua resposta.

a.(F) Se uma operação em uma estrutura tem complexidade amortizada (ao longo de n chamadas) de $O(\log n)$, sua complexidade de pior caso é $O(\log n)$ também.

> Isso não é necessário que aconteça. No exemplo do contador de bits temos a complexidade de pior caso igual a $O(n)$ e a amortizada de $O(1)$

b.(F) Se uma operação em uma estrutura tem complexidade amortizada (ao longo de n chamadas) de $O(n)$, sua complexidade de melhor caso é $\Omega(n)$ também.

> Observe que uma das operações poderia ser $\Theta(1)$ e ainda assim termos um complexidade amortizada de $O(n)$.

c.(V) Se uma operação em uma estrutura tem complexidade amortizada (ao longo de n chamadas) de $O(1)$, sua complexidade de melhor caso é $\Omega(1)$ também.

> Como a complexidade amortizada é a soma das complexidades de cada operação dividida pela número de operações, pelo menos uma delas tem precisa ter a mesma complexidade da complexidade amortizada.

d.(F) $2n = \omega(2n + 1)$.

> Se $2n = \omega(2n + 1)$, então para todo $c > 0$, temos que $2n \geq c.(2n + 1)$ e logo $2n \geq 2cn + c$ o que implica que $1 - c \geq \frac{c}{2n}$, o que é absurdo pois $1 - c$ pode ser negativo e c e $2n$ são positivos.

e.(F) $n^2 = \Theta(n^{logn})$.

> Se $n^2 = \Theta(n^{logn})$, então existe c tal que $n^2 \geq c.n^{\log n}$. Isso implica que

f.(V) $2^n = \Omega(2^{n+1})$.

> Tome $c = \frac{1}{4}, n_0 = 100$.

g.(F) $\log n = \Omega(\sqrt{n})$.

> Se $\log n = \Omega(\sqrt{n})$, então existem c, n_0 tal que $\log n \geq c\sqrt{n}$ para todo $n \geq n_0$. Note que $\log n \geq c.\sqrt{n}$ e $2^{\log n} \geq 2^{c.\sqrt{n}}$. Então temos que $n \geq 2^{c.\sqrt{n}}$, o que é absurdo.

h.(F) $n^2 = O(2^{\log n})$.

> Depende da base.

i.(V) $\log n = o(n)$.

> Tome $c = 10, n_0 = 100$.

j.(F) $2^n = \Theta(\log n^n)$.

> Se $2^n = \Theta(\log n^n)$, então existe c e n_0 tais que $2^n \leq c.\log n^n$, por propriedade de logaritmo temos $2^n \leq c.n \log n$, note que $n \log n \leq n^2$ pois $\log n \leq n$. Temos então que $2^n \leq c.n \log n \leq c.n^2$, tirando o logaritmo em ambos os lados, $n \leq \log c + 2 \log n$, o que é absurdo.

Questão 3 (5 pontos) (Notação Assintótica) A seguir apresentamos algumas afirmações sobre notação assintótica. Usando as definições formais de notação assintótica, apresente provas formais para os fatos que são verdadeiros e contra-exemplos (com explicação) para os casos que sejam falsos.

a) Considerando duas funções f, g não negativas, temos que se $f(n) = O(g(n))$ então $2^{f(n)} = O(2^{g(n)})$.

A afirmação é falsa! Considere $f(n) = \log_2 n$ e $g(n) = \log_4 n$. Nós temos que $\log_2 n = O(\log_4 n)$, mas $2^{\log_2 n} = n$ não é $O(2^{\log_4 n}) = O(2^{\frac{1}{2}\log_2 n}) = O(2^{\log_2 n^{\frac{1}{2}}}) = O(n^{\frac{1}{2}}) = O(\sqrt{n})$

b) Considerando duas funções f, g não negativas, temos que se $f(n) = O(g(n))$ então $g(n) = \Omega(f(n))$.

A afirmação é verdadeira. Vamos apresentar uma demostração direta para a mesma. Como $f(n) = O(g(n))$ temos que existem $c > 0, n_0$ tal que $f(n) \leq cg(n)$ para todo $n \geq n_0$. Isso significa que $\frac{1}{c}f(n) \leq g(n)$, como $c > 0$ temos que $\frac{1}{c} > 0$ logo $g(n) = \Omega(f(n))$

Questão 4 (5 pontos) (Funções Recursivas e Recorrências) Indique uma equivalência assintótica (Θ) para cada uma das funções recursivas apresentadas abaixo. Para ambas, considere que os casos bases das recursões são todos (e quantos necessários forem) iguais a 1. Apresente cálculos ou justifique sua reposta.

i. $F(n) = F(n-1) + \frac{n}{2}$

$$\begin{aligned} F(n) &= F(n-1) + \frac{n}{2} \\ &= F(n-2) + \frac{n-1}{2} + \frac{n}{2} \\ &= F(n-3) + \frac{n-1}{2} + \frac{n-1}{2} + \frac{n}{2} \\ &= \dots \\ &= F(n-k) + \sum_{i=0}^{k-1} \frac{n-i}{2} \\ &= \dots \\ &= F(0) + \sum_{i=1}^{n} \frac{i}{2} \end{aligned}$$

Então temos que $F(n) = F(0) + \sum_{i=1}^{n} \frac{i}{2} = 1 + \frac{1}{2}.\frac{n(n+1)}{2} = \Theta(n^2)$

ii. $F(n) = F(n-1) + \log n$

$$\begin{aligned} F(n) &= F(n-1) + \log n \\ &= F(n-2) + \log(n-1) + \log n \\ &= F(n-3) + \log(n-2) + \log(n-1) + \log n \\ &= \dots \\ &= F(n-k) + \sum_{i=0}^{k-1} \log(n-i) \\ &= \dots \\ &= F(0) + \sum_{i=1}^{n} \log i \end{aligned}$$

$F(n) = F(0) + \sum_{i=1}^{n} \log i = 1 + \log(\prod_{i=1}^{n} i) = 1 + \log n! = \Theta(\log n!) = \Theta(\log n^n) = \Theta(n \log n)$

Questão 5 (5 pontos) (Teorema Mestre) Considere a seguinte recorrência

$$T(n) = 4T(\frac{n}{2}) + X$$

que possui como caso base (o valor inicial) $O(1)$. O teorema mestre permite que determinemos uma equivalência assintótica para recorrências desse tipo de forma quase automática. Então, usando o teorema mestre, indique para cada um dos itens abaixo uma função para assumir o lugar de X que torna:

i. $T(n) = \Theta(n^2)$.

$f(n) = 1$

ii. $T(n) = \Theta(n^2 \log n)$.

$f(n) = n^2$

iii. $T(n) = \Theta(n^3 \log n)$.

$f(n) = n^3 \log n$

iv. a complexidade de $T(n)$ impossível de ser determinada pela aplicação direta do teorema mestre.

$f(n) = n^2 \log n$

RASCUNHO:

Informações Úteis!

1. Teorema Mestre:Sejam $a \geq 1$ e $b > 1$ constantes, $f(n)$ uma função e

$$T(n) = aT(\frac{n}{b}) + f(n),$$

   Então, para algum $\epsilon > 0$

   Se $f(n) = O(n^{\log_b a - \epsilon})$ $T(n) = \Theta(n^{\log_b a})$
   Se $f(n) = \Theta(n^{\log_b a})$ $T(n) = \Theta(n^{\log_b a} \log n)$
   Se $f(n) = \Omega(n^{\log_b a + \epsilon})$ e $af(\frac{n}{b}) \leq cf(n)$ então $T(n) = \Theta(f(n))$

   .

2. $n! = \Theta(n^n)$.