Universidade Federal do Ceará – Campus Quixadá

QXD0041 – Projeto e Análise de Algoritmos – 1º Semestre de 2024

Lista de Exercícios

Crescimento de funções

Notação assintótica

Questão 1. Analise a complexidade assintótica do algoritmo apresentado abaixo no melhor e pior caso.

```
Algoritmo1 (S, x):
Para i <- 1, 2, ..., |S| faça
   Para j <- i + 1, i + 2, ..., |S| faça
      Se S[i] + S[j] = x então
         Retorne V
Retorne F
```

Questão 2. Escreva o pseudocódigo de um algoritmo que verifica se um número n é primo ou não. No pior caso o algoritmo descrito deve executar na ordem de $\sqrt{n}$ instruções. Para que isso seja possível, considere o teorema abaixo.

Teorema 1 *Se um número n não é primo então n possui pelo menos um divisor no intervalo* $[2, \sqrt{n}]$.

Questão 3. Forneça a complexidade de tempo do algoritmo apresentado abaixo.

```
Algoritmo2 (N):
i <- 1
soma <- 0
Enquanto i < N faça
   Para j <- 1, ..., i faça
      soma <- soma + 1
   i <- 2i
Retorne soma
```

Questão 4. Mostre, via definição, que:

1. $\frac{n(n-10)}{2} = O(n^2)$
2. $\sum_{i=1}^{n} i = \Omega(n^2)$
3. $2^{n-1} = \Omega(2^n)$
4. $\binom{n}{2} = \Theta(n^2)$

Questão 5. Para cada um dos pares de funções abaixo existem três possibilidades:
(i) $f(n) = O(g(n))$, (ii) $f(n) = \Omega(g(n))$ ou (iii) $f(n) = \Theta(g(n))$
Determine qual das opções se aplica e explique brevemente porquê.

1. $f(n) = \log_2 n^2$, $g(n) = \log_2 n + 5$
2. $f(n) = \log_2 n$, $g(n) = \log_3 n$
3. $f(n) = \sqrt{n}$, $g(n) = \log_2 n$
4. $f(n) = 100n$, $g(n) = n^2$
5. $f(n) = 500n + 100n^{0.5} + 50n^2$, $g(n) = n^3$
6. $f(n) = n^2$, $g(n) = n\log_2 n$
7. $f(n) = 2^n$, $g(n) = 10n^2$
8. $f(n) = 2^n$, $g(n) = 3^n$

Questão 6. Prove o seguinte teorema: "Para quaisquer duas funções $f(n)$ e $g(n)$, temos que $f(n) = \Theta(g(n))$ se, e somente se, $f(n) = O(g(n))$ e $f(n) = \Omega(g(n))$"

Questão 7. Sejam $f(n)$ e $g(n)$ duas funções assintóticamente não negativas. Usando a definição básica da notação Θ, prove que $\max\{f(n), g(n)\} = \Theta(f(n) + g(n))$.

Questão 8. É verdade que $2^{n+1} = O(2^n)$? É verdade que $2^{2n} = O(2^n)$?

Questão 9. Prove que o tempo de execução de um algoritmo é $\Theta(g(n))$ se e somente se seu tempo de execução no pior caso é $O(g(n))$ e seu tempo de execução no melhor caso é $\Omega(g(n))$.

Questão 10. (Adaptado de Horowitz et al.) Um quadrado mágico é uma matriz $n \times n$ de inteiros de 1 até n^2 tais que a soma de cada linha, coluna, ou diagonal é a mesma. H. Coxter deu a seguinte regra para para criar um quadrado mágico quando n é ímpar.

> "Comece com 1 no centro da última linha; então vá para baixo e para a direita, atribuindo números em ordem crescente nos quadrados vazios; se você sair do quadrado, imagine que o mesmo quadrado está ladrilhando o plano e continue; se o próximo quadrado já estiver ocupado, suba uma linha (ao invés de se mover para baixo e para a direita) e continue."

1. Escreva um algoritmo no formato de pseudocódigo que implemente a regra de Coxter acima. Lembre-se de dar um nome com os parâmetros de entrada.
2. Analise a complexidade desse algoritmo. Justifique a resposta.

Questão 11. Encontre um contraexemplo para a seguinte afirmação: $f(n) = O(s(n))$ e $g(n) = O(r(n))$ implicam que $f(n) - g(n) = O(s(n) - r(n))$.

Questão 12. Encontre duas funções $f(n)$ e $g(n)$, ambas monotonicamente crescentes, tais que $f(n) \neq O(g(n))$ e $g(n) \neq O(f(n))$.