Trabalho Teórioco 04 e 05

Camila Moreira Lopes[1]

[1]Instituto de Ciências Exatas e Informática - Pontífica Universidade Católica de Minas Gerais

camila.lopes.1264894@sga.pucminas.br

1. Qual é a diferença entre as notaçõe O, Ω e Θ

Hoje em dia também existe uma discussão muito recorrente sobre linguagens de programação serem mais eficiente e/ou mais rápidas que outras. Portanto, esse debate gerou uma necessidade de analisar a **Complexidade do Algoritmo**, assim é possível projetar os algoritmos mais eficazes por meio do seu tempo de execução e do espaço (memória) utilizada.

1.1. Notação O

Dentro da Complexidade de Algoritmos, reconhecemos a notação **O** como sendo o limite superior, ou o pior caso, no qual é o maior número de operações usadas para qualquer entrada de tamanho n. Assim, dizemos que, dada uma função g(n), denotamos O(g(n)) o conjunto das funções

$\{ f(n) : \exists$ constantes c e n_0 tais que $0 \leq f(n) \leq cg(n)$ para $n \geq n_0 \}$

Isto é, para valores de n suficienteme grandes, f(n) é igual ou menor que g(n). Assim, um polinômio de grau d é de ordem $O(n^d)$. Como uma constante pode ser considerada como um polinômio de grau 0, então dizemos que um constante é $O(n^0)$, ou seja, O(1).

1.2. Notação Ω

Por conseguinte, temos o melhor caso, isto é, o limite inferior o qual é denotado como: Ω; Nele temos o menor número de operações usadas para qualquer entrada de tamanhp n. Assim, dizemos que, dada uma função g(n), denotamos Ω(g(n)) o conjunto das funções

$\{ f(n) : \exists$ constantes c e n_0 tais que $0 \leq cg(n) \leq f(n)$ para $n \geq n_0 \}$

1.3. Notação Θ

Por último, e não menos importante, temos o caso médio, que supõe conhecida uma certa distribuição da entradas, tendo seus resultados entre o **O** e o Ω.

2. Exercícios da Unidade 01b

2.1. Exercícios Feitos

2.1.1. Exercício 1 - pg. 9

Cálcule o número de subtrações que o código abaixo realiza:

R.: O código faz três subtrações

2.1.2. Exercício 2 - pg. 11

Cálcule o número de adições que o código abaixo realiza:

R.: O código faz três adições no melhor caso e cinco adições no pior caso.

2.1.3. Exercício 3 - pg. 16

Cálcule o número de adições que o código abaixo realiza:

R.: O número máximo de adições acontece quando a primeira condição do if é falsa e a segunda, verdadeira. Se a primeira condição for verdadeira, o Java nem executa a segunda condição.

2.1.4. Exercício 4 - pg. 18

Cálcule o número de subtrações que o código abaixo realiza:

R.: O código realiza 4 subtrações.

2.1.5. Exercício 5 - pg. 22

Cálcule o número de subtrações que o código abaixo realiza:

R.: 2n subtrações.

2.1.6. Exercício 6 - pg. 24

Cálcule o número de subtrações que o código abaixo realiza:

R.: 3 subtrações.

2.1.7. Exercício 7 - pg. 26

Cálcule o número de subtrações que o código abaixo realiza:

R.: (n-3) subtrações.

2.1.8. Exercício 8 - pg. 32

Cálcule o número de subtrações que o código abaixo realiza:

R.: 6 subtrações.

2.1.9. Exercício 9 - pg. 76

Cálcule o número de multiplicações que o código abaixo realiza:

R.: O código faz ln(n) arrendondado para baixo + 1 multiplicações.

2.1.10. Exercício 11 - pg. 86

Encontre o menor valor em um array de inteiros

1°) Qual é a operação relevante? R: Comparação entre elementos do array?

2°) Quantas vezes ela será executada? R: Se tivermos n elementos: T(n) = n - 1

3°) O nosso T(n) = n - 1 é para qual dos três casos? Os três.

2.2. Exercícios Não Feitos

2.2.1. Exercício 1 - pg. 4

(a) $2^0 = 1$
(b) $2^1 = 2$
(c) $2^2 = 4$
(d) $2^3 = 8$
(e) $2^4 = 16$
(f) $2^5 = 32$
(g) $2^6 = 64$
(h) $2^7 = 128$
(i) $2^8 = 256$
(j) $2^9 = 512$
(k) $2^10 = 1024$
(l) $2^11 = 2048$

2.2.2. Exercício 2 - pg. 5

(a) $\lg(2048) = 2^11$
(b) $\lg(1024) = 2^10$
(c) $\lg(512) = 2^9$
(d) $\lg(256) = 2^8$
(e) $\lg(128) = 2^7$
(f) $\lg(64) = 2^6$
(g) $\lg(32) = 2^5$
(h) $\lg(16) = 2^4$
(i) $\lg(8) = 2^3$
(j) $\lg(4) = 2^2$
(k) $\lg(2) = 2^1$
(l) $\lg(1) = 2^0$

2.2.3. Exercício 3 - pg. 6

(a) 4,01 = 5
(b) 4,01 = 4
(c) 4,99 = 5

(d) 4,99 = 4
(e) lg(16) = 4
(f) lg(16) = 4
(g) lg(17) = 4.08746284125034
(h) lg(17) = 4
(i) lg(17) = 5
(j) lg(15) = 3.9068905956085187
(k) lg(15) = 3
(l) lg(15) = 4

2.2.4. Exercício 5 - pg. 29

Cálcule o número de subtrações que o código abaixo realiza:

R.: O código faz 4 subtrações.

2.2.5. Exercício 6 - pg. 30

Cálcule o número de subtrações que o código abaixo realiza:

R.: O código faz 8 subtrações.

2.2.6. Exercício 7 - pg. 31

Cálcule o número de subtrações que o código abaixo realiza:

R.: O código faz 7 subtrações.

2.2.7. Exercício 8 - pg. 69

Cálcule o número de subtrações que o código abaixo realiza:

R.: O código faz n^2 subtrações.

2.2.8. Exercício 9 - pg. 70

Cálcule o número de subtrações que o código abaixo realiza:

R.: O código faz 8 subtrações.

2.2.9. Exercício 10 - pg. 71

Cálcule o número de multiplicações que o código abaixo realiza:

R.: O código faz n*(n-3) multiplicações.

2.2.10. Exercício 11 - pg. 72

Cálcule o número de multiplicações que o código abaixo realiza:

R.: O código faz n*(n-7) multiplicações.

2.2.11. Exercício 12 - pg. 73

Cálcule o número de multiplicações que o código abaixo realiza:

R.: O código faz ln(n) arrendondado para baixo + 1 multiplicações.

2.2.12. Exercício 13 - pg. 74

Cálcule o número de multiplicações que o código abaixo realiza:

R.: O código faz ln(n+4) arrendondado para baixo + 1 multiplicações.

2.2.13. Exercício 14 - pg. 75

Cálcule o número de multiplicações que o código abaixo realiza:

R.: O código faz $(n-7)^2$ arrendondado para baixo + 1 multiplicações.

2.2.14. Exercício 15 - pg. 80

Cálcule o número de multiplicações que o código abaixo realiza:

R.: O código faz ln(n+1) arrendondado para baixo + 1 multiplicações.

2.2.15. Exercício 17 - pg. 82

Cálcule o número de multiplicações que o código abaixo realiza:

R.: O código faz ln(n) arrendondado para baixo + 1 multiplicações.