Universidade Federal de Ouro Preto
Instituto de Ciências Exatas e Aplicadas
Departamento de Engenharia Elétrica

CSI488 – ALGORITMOS ESTRUTURAS DE DADOS I (TURMA 22)

ANALISE DE CPMPLEXIDADE

Jhonatan Figueiredo Almeida

João Monlevade
Agosto de 2022

CSI488 – ALGORITMOS ESTRUTURAS DE DADOS I (TURMA 22)

ANALISE DE COMPLEXIDADE

Discente:	Jhonatan Figueiredo Almeida	Matrícula:	20.1.8164
Professor:	Alexandre Magno de Souza		

João Monlevade
Agosto de 2022

1. **a.**

$$f(n) = 1 + \sum_{i=0}^{n-3} 1 + \sum_{i=0}^{n-3}\left(1 + \sum_{k=i+1}^{n} 4\right) =$$

$$1 + \sum_{i=0}^{n-3} 1 + \sum_{i=0}^{n-3} (1 + (n-(i+1)+1)\cdot 4) =$$

$$1 + \sum_{i=0}^{n-3} 1 + \sum_{i=0}^{n-3} 1 + 4\cdot \sum_{i=0}^{n-3} n - 4\cdot\sum_{i=0}^{n-3} i =$$

$$1 + (n-2) + (n-2) + 4\cdot(n-3+1)\cdot n - 2\cdot(n-3+1)\cdot(n-3) =$$

$$2n^2 + 4n - 15$$

b.

$$f(n) = 1 + \sum_{i=0}^{\frac{n}{2}+1} 1 + \sum_{i=0}^{\frac{n}{2}+1}\left(1 + \sum_{k=0}^{n} 2\right) =$$

$$1 + \sum_{i=0}^{\frac{n}{2}+1} 1 + \sum_{i=0}^{\frac{n}{2}+1} (1 + 2\cdot(n+1)) =$$

$$1 + \sum_{i=0}^{\frac{n}{2}+1} 1 + \sum_{i=0}^{\frac{n}{2}+1} 1 + 2\cdot\sum_{i=0}^{\frac{n}{2}+1} n + \sum_{i=0}^{\frac{n}{2}+1} 2 =$$

$$1 + \left(\frac{n}{2} - 1 + 1\right) + \left(\frac{n}{2} - 1 + 1\right) + 2\cdot n\cdot\left(\frac{n}{2} - 1 + 1\right) + 2\cdot\left(\frac{n}{2} - 1 + 1\right) =$$

$$n^2 + 2n + 1$$

c.

$$f(n) = 2 + \sum_{j=1}^{n} 1 + \sum_{j=1}^{n}\left(1 + \sum_{k=1}^{j} 2\right) =$$

$$2 + \sum_{j=1}^{n} 1 + \sum_{j=1}^{n} (1 + (j-1+1)\cdot 2) =$$

$$2 + \sum_{j=1}^{n} 1 + \sum_{j=1}^{n} (1 + 2j) =$$

$$2 + \sum_{j=1}^{n} 1 + \sum_{j=1}^{n} 1 + 2\cdot\sum_{j=1}^{n} j =$$

$$2 + n + n + 2\cdot\left(\frac{(n-1+1)(n+1)}{2}\right) =$$

$$n^2 + 3n + 2$$

d.

$$f(n) = 1 + \sum_{i=1}^{3}\left(2 + \sum_{j=i}^{n}\left(2 + \sum_{k=j}^{n} 2\right)\right) =$$

$$1 + \sum_{i=1}^{3}\left(2 + \sum_{j=i}^{n}(2 + 2\cdot(n - j + 1))\right) =$$

$$1 + \sum_{i=1}^{3}\left(2 + \sum_{j=i}^{n}(2 + 2n - 2j + 2)\right) =$$

$$1 + \sum_{i=1}^{3}\left(2 + \sum_{j=i}^{n}4 + 2\cdot\sum_{j=i}^{n}n - 2\cdot\sum_{j=i}^{n}j\right) =$$

$$1 + \sum_{i=1}^{3}(2 + 4\cdot(n - i + 1) + 2n\cdot(n - i + 1) - (n - i + 1)(n + i) =$$

$$1 + \sum_{i=1}^{3}(6 + 5n - 5i + n^2 - 2ni + i^2) =$$

$$1 + \sum_{i=1}^{3}6 + 5\cdot\sum_{i=1}^{3}n - 5\cdot\sum_{i=1}^{3}i + \sum_{i=1}^{3}n^2 - 2\cdot\sum_{i=1}^{3}ni + \sum_{i=1}^{3}i^2 =$$

$$1 + 18 + 15n - 5\cdot\frac{(3-1+1)(3+1)}{2} + 3n^2 - 2n\cdot\frac{(3-1+1)(3+1)}{2} + \frac{3\cdot(3+1)(6+1)}{6} =$$

$$19 + 15n - 30 + 3n^2 - 12n + 14 =$$

$$3n^2 + 3n + 3$$

2. A análise de complexidade do número de atribuições dos casos requeridos é apresentada a seguir.

Programa 1: SomaMatriz.c

```c
void somar_matriz(int** matriz1, int** matriz2, int** matriz, int tamanho)
{
    for (int i = 0; i < tamanho; i++)
    {
        for (int j = 0; j < tamanho; j++)
        {
            matriz[i][j] = matriz1[i][j] + matriz2[i][j];
        }
    }
}
```

Programa 2: MatrizTransposta.c

```c
void matriz_transposta(int** matriz1, int** matriz, int tamanho)
{
    for (int i = 0; i < tamanho; i++)
    {
        for (int j = 0; j < tamanho; j++)
        {
            matriz[i][j] = matriz1[j][i];
        }
    }
}
```

Programa 3: MultiplicaMatriz.c

```c
void multiplicar_matriz(int** matriz1, int** matriz2, int** matriz, int tamanho)
{
    int soma;
    for (int i = 0; i < tamanho; i++)
    {
        for (int j = 0; j < tamanho; j++)
        {
            soma = 0;
            for (int k = 0; k < tamanho; k++)
            {
                soma = soma + (matriz1[i][k]) * (matriz2[k][j]);
            }
            matriz[i][j] = soma;
        }
    }
}
```

Ao realizar a observação dos códigos, é possível perceber que a função de complexidade do número atribuições é a mesma para a soma e para a transposição de matrizes, dessa forma podemos escrever que:

$$f(n) = 1 + \sum_{i=0}^{n-1} 1 + \sum_{i=0}^{n-1} \left(1 + \sum_{j=0}^{n-1} 2 \right) =$$

$$1 + n + \sum_{i=0}^{n-1} (1 + 2n) =$$

$$1 + \sum_{i=0}^{n-1} 1 + \sum_{i=0}^{n-1} 2n =$$

$$1 + n + n + 2n^2 =$$

$$2n^2 + 2n + 1$$

Para a multiplicação de matrizes, temos:

$$f(n) = 1 + \sum_{i=0}^{n-1} 1 + \sum_{i=0}^{n-1} \left(1 + \sum_{j=0}^{n-1} \left(4 + \sum_{k=0}^{n-1} 2 \right) \right) =$$

$$1 + n + \sum_{i=0}^{n-1} \left(1 + \sum_{j=0}^{n-1} (4 + 2n) \right) =$$

$$1 + n + \sum_{i=0}^{n-1} \left(1 + \sum_{j=0}^{n-1} 4 + \sum_{j=0}^{n-1} 2n \right) =$$

$$1 + n + \sum_{i=0}^{n-1} (1 + 4n + 2n^2) =$$

$$1 + n + \sum_{i=0}^{n-1} 1 + \sum_{i=0}^{n-1} 4n + \sum_{i=0}^{n-1} 2n^2 =$$

$$1 + n + n + 4n^2 + 2n^3 =$$

$$2n^3 + 4n^2 + 2n + 1$$

4.

a) $f(n) = 3n^3 + 2n^2 + n$

$O g(n^3)$

Análise $\Rightarrow 3n^3 > 2n^2 \quad n > 1 \qquad n_0 = 2$

$3n^3 + 2n^2 + n \leq c n^3$

$\frac{3n^3}{n^3} + \frac{2n^2}{n^3} + \frac{n}{n^3} \leq c$

$3 + \frac{2}{n} + \frac{1}{n^2} \leq c$

$3 + 1 + \frac{1}{4} \leq c$

$\frac{4}{1} + \frac{1}{4} \leq c$

$\frac{17}{4} \leq c$ logo $n_0 = 2$, $c = \frac{17}{4}$

$\frac{17}{4} \leq c$ logo $n_0 = 2$, $c = \frac{17}{4}$

b) $f(n) = 2n^2 + 4n - 15$

$O(g(n^2))$

Análise $2n^2 > 4n \qquad n_0 = 4, c$

$2n^2 + 4n - 15 \leq c n^2$

$\frac{2n^2}{n^2} + \frac{4n}{n^2} - \frac{15}{n^2}$

$2 + \frac{4}{n} - \frac{15}{n^2} \leq c$

$2 + \frac{4}{4} - \frac{15}{16} \leq c$

$2 - \frac{15}{16} \leq c$

$\frac{32 - 15}{16} \leq c$

$\frac{17}{16} \leq c$

5.

d. Se $f(n) = O(g(n))$, então $g(n) = O(f(n))$
Não é verdade para todos os casos.

$$f(n) = O(g(n)) \longrightarrow f(n) \leq c_1 \cdot g(n)$$
$$g(n) = O(f(n)) \longrightarrow g(n) \leq c_2 \cdot f(n)$$

Com isso, deveríamos ter que:

$$f(n) \leq c_1 \cdot g(n) \leq c_2 \cdot f(n)$$

Desta forma, para que a inequação seja verdadeira, $f(n)$ deve ser igual a $g(n)$.

f. $2^{2n} = O(2^n)$
Não é verdade.

h. $f(n) = O(u(n))$ e $g(n) = O(v(n))$ então $f(n) - g(n) = O(u(n) - v(n))$
Não é verdade. Não são realizadas as operações de subtração e divisão com a notação O-grande.

6 .

a. Todas as afirmativas são falsas. Na verdade, n_0 deve ser maior ou igual a 0, e não maior ou igual a 1, permitindo com que n possa ser maior ou igual a 0.

7.

b. $f(n) = \Theta(h(n))$ e $i(n) = \Omega(h(n))$.

F(n)	G(n)	O	o	Ω	ω	Θ
Log log n	N^ c	Sim	Sim	Não	Não	não
2 ^n	N^ n	Sim	Sim	Não	Não	Não
N^ 2	n log n	Não	Não	Sim	Sim	Não
N^log m	M^log n	sim	Não	sim	Não	Sim
Log n !	Log n^n	Sim	Não	Sim	Não	Sim

9. B O(n)

10.E) Nenhum das anteriores

11. E) N^2 log n

13. $\begin{cases} 3T(n/4) + Cn^2 & \text{para} > 0 \\ 1, & \text{para } n = 0 \end{cases}$

Teorema mestre

$a = 3$

$b = 4$

$f(n) = Cn^2$

Logo se encaixa no 3º caso

$T(n) = \Theta(Cn^2)$