

# Lista 5 – Complexidade de Algoritmos Professora: Elvira Padua Lovatte

**Cursos: Ciência da Computação e Engenharia de Computação**

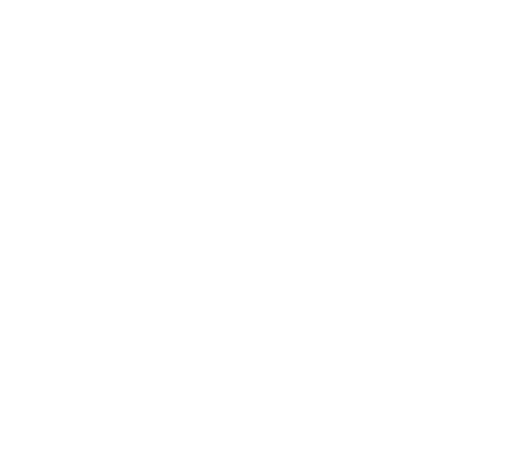
## Nomes: Dimas Curti De Almeida

Assunto: Técnicas de programação

1. Relacione a coluna da esquerda com a da direita:

(**2**) Divide um problema em subproblemas menores a fim de agilizar na resolução final.

O resultado de um subproblema é armazenado em uma estrutura de dados para ser utilizado futuramente e evitar recálculo do mesmo subproblema, diminuindo em muito o custo de algoritmos recursivos.



(1) Algoritmo Guloso

(2) Programação Dinâmica

(3) Divisão e Conquista

(**3**) Divide um problema em subproblemas menores. Os subproblemas são independentes, o risco de haver recálculo dos subproblemas é mínima, não são armazenados os resultados encontrados.

(**1**) Deve-se utilizar essa estratégia somente quando o problema não exigir que a solução seja sempre ótima, ou quando o problema puder ser dividido em um único subproblema, para que não se dependa de escolhas futuras, sendo a solução ótima global neste caso, a solução ótima local.

1. Sejam as matrizes A10x20, B20x130 e C130x50.

Suponha que é necessário determinar o produto P = ABC. Para isso, pode-se fazer:

* 1. (AB)C ou
  2. A(BC)

Nos dois casos o resultado final é P, mas o número de multiplicações executado no caso i é diferente do número de multiplicações realizadas no caso ii. Determine o número de multiplicações executados em cada um dos dois casos.

**i = 91000 e ii = 140000**

1. Sabe-se que para multiplicar n matrizes é preciso escolher a ordem de multiplicação que resultará na menor quantidade de operações aritméticas. Em sala de aula, apresentamos um bom algoritmo dinâmico para resolver esse problema.

Simule (passo a passo) este algoritmo (construa as matrizes M e S) para obtenção da melhor ordem de multiplicação para as seguintes matrizes:

M1 [10,20]; M2 [20, 2]; M3 [2, 1]; M4 [1, 10]

|  |  |  |
| --- | --- | --- |
| **Organização** | **Computação do custo** | **Custo** |
| M 1 × ( ( M 2 × M 3 ) × M 4 ) {\displaystyle M\_{1}\times ((M\_{2}\times M\_{3})\times M\_{4})} M1 x ((M2 x M3) x M4) | 20 x 2 x 1 + 20 x 1 x 10 + 20 x 10 x 20 2 × 30 × 20 + 2 × 20 × 5 + 200 × 2 × 5 {\displaystyle 2\times 30\times 20+2\times 20\times 5+200\times 2\times 5} | 42403.400 {\displaystyle 3.400} |
| ( M 1 × ( M 2 × M 3 ) ) × M 4 {\displaystyle (M\_{1}\times (M\_{2}\times M\_{3}))\times M\_{4}} (M1 x (M2 x M3)) x M4 | 20 x 2 x 1 + 20 x 1 x 20 + 20 x 20 x 102 × 30 × 20 + 200 × 2 × 20 + 200 × 20 × 5 {\displaystyle 2\times 30\times 20+200\times 2\times 20+200\times 20\times 5} | 29.200 {\displaystyle 29.200} 4440 |
| (( M 1 × M 2 ) × ( M 3 × M 4 ) {\displaystyle (M\_{1}\times M\_{2})\times (M\_{3}\times M\_{4})}M1 xMM1 x M2) x (M3 x M4) | 10 x 20 x 2 + 2 x 1 x 10 + 10 x 2 x 10 200 × 2 × 30 + 30 × 20 × 5 + 200 × 30 × 5 {\displaystyle 200\times 2\times 30+30\times 20\times 5+200\times 30\times 5} | 62045.000 {\displaystyle 45.000} |

**(( M 1 × M 2 ) × ( M 3 × M 4 ) {\displaystyle (M\_{1}\times M\_{2})\times (M\_{3}\times M\_{4})}M1 xMM1 x M2) x (M3 x M4) = S [10, 10]**

1. Comparar (construindo as matrizes) duas sequências genéticas das a seguir: ACCGGTCGAGTG

GTCGTTCGGAATGCCGTTGCTCTGTAAA

OBS: usar a Programação Dinâmica: cadeia comum mais longa

**Y = ACCGGTCGAGTG**

**X = GTCGTTCGGAATGCCGTTGCTCTGTAAA**

**c(i, j) como o tamanho da LCS(Xi,Yj):**

**C(𝑖, 𝑗) = 0 𝑠𝑒 𝑖=0 𝑜𝑢 𝑗=0 , 𝐶(𝑖−1, 𝑗−1) +1 se i=j, max {𝑐(𝑖−1, 𝑗), 𝑐(𝑖, 𝑗−1)} 𝑠𝑒 𝑖≠𝑗**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Yj** | A | C | C | G | G | T | C | G | A | G | T | G |
| **Xi** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| G | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| T | 0 | 0 | 0 | 0 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| C | 0 | 0 | 1 | 1 | 1 | 1 | 2 | 3 | 3 | 3 | 3 | 3 | 3 |
| G | 0 | 0 | 1 | 1 | 2 | 2 | 2 | 3 | 4 | 4 | 4 | 4 | 4 |
| T | 0 | 0 | 1 | 1 | 2 | 2 | 3 | 3 | 4 | 4 | 4 | 5 | 5 |
| T | 0 | 0 | 1 | 1 | 2 | 2 | 4 | 4 | 4 | 4 | 4 | 5 | 5 |
| C | 0 | 0 | 1 | 2 | 2 | 2 | 4 | 5 | 5 | 5 | 5 | 5 | 6 |
| G | 0 | 0 | 1 | 2 | 3 | 3 | 4 | 5 | 6 | 6 | 6 | 6 | 6 |
| G | 0 | 0 | 1 | 2 | 3 | 4 | 4 | 5 | 6 | 6 | 7 | 7 | 7 |
| A | 0 | 1 | 1 | 2 | 3 | 4 | 4 | 5 | 6 | 7 | 7 | 7 | 7 |
| A | 0 | 1 | 1 | 2 | 3 | 4 | 4 | 5 | 6 | 7 | 7 | 7 | 7 |
| T | 0 | 1 | 1 | 2 | 3 | 4 | 5 | 5 | 6 | 7 | 7 | 8 | 8 |
| G | 0 | 1 | 1 | 2 | 3 | 4 | 5 | 5 | 6 | 7 | 8 | 8 | 9 |
| C | 0 | 1 | 2 | 2 | 3 | 4 | 5 | 6 | 6 | 7 | 8 | 8 | 9 |
| C | 0 | 1 | 2 | 3 | 3 | 4 | 5 | 6 | 6 | 7 | 8 | 8 | 9 |
| G | 0 | 1 | 2 | 3 | 4 | 4 | 5 | 6 | 7 | 7 | 8 | 8 | 9 |
| T | 0 | 1 | 2 | 3 | 4 | 4 | 5 | 6 | 7 | 7 | 8 | 9 | 9 |
| T | 0 | 1 | 2 | 3 | 4 | 4 | 5 | 6 | 7 | 7 | 8 | 9 | 9 |
| G | 0 | 1 | 2 | 3 | 4 | 5 | 5 | 6 | 7 | 7 | 8 | 9 | 10 |
| C | 0 | 1 | 2 | 3 | 4 | 5 | 5 | 6 | 7 | 7 | 8 | 9 | 10 |
| T | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 6 | 7 | 7 | 8 | 9 | 10 |
| C | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 7 | 7 | 8 | 9 | 10 |
| T | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 7 | 7 | 8 | 9 | 10 |
| G | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 8 | 8 | 9 | 10 |
| T | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 8 | 8 | 9 | 10 |
| A | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 9 | 9 | 10 |
| A | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 9 | 9 | 10 |
| A | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 9 | 9 | 10 |

**R = CCGGTCGGTG**