# Divisão e conquista - Exercícios

Carlos Eduardo Gonzaga Romaniello de Souza - 19.1.4003

29 de abril de 2022

**1)**Usando o algoritmo que multiplica dois números binários a um custo $\Theta(n^{1.585})$, multiplique 1001 por 0110.

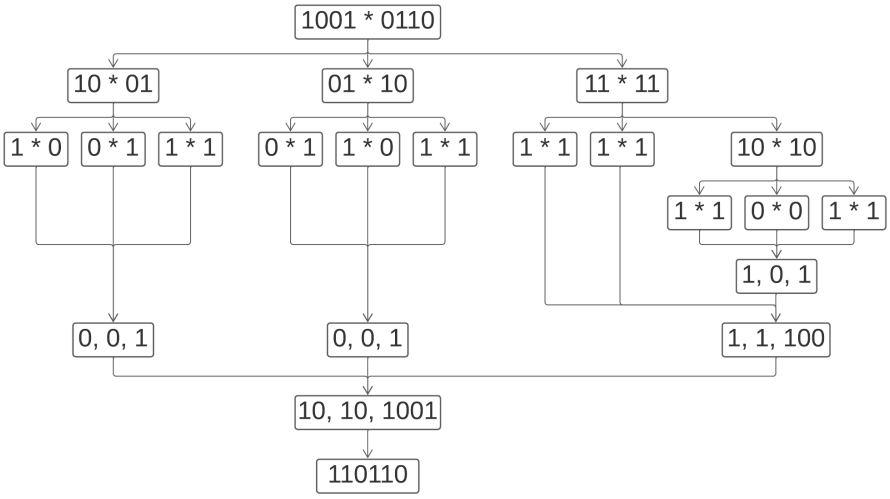


Figure 1: Diagrama

**2)** Suponha que você deseja escolher um dentre os três algoritmos, Qual é o tempo de execução de cada algoritmo em notação O e qual você deveria escolher.

**2.a)**Algoritmo A soluciona problemas por dividí-los em cinco subproblemas com metade do tamanho, recursivamente soluciona cada subproblema e combina as soluções em tempo linear.

- $T(n) = 5 \times T(\frac{n}{2}) + O(n)$
- $a = 5, b = 2, d = 1$
- $1 <= \log_2 5$
- $T(n) = O(n^{\log_2 5})$

**2.b)** Algoritmo B soluciona problemas de tamanho n por, recursivamente, solucionar 2 subproblemas de tamanho n-1 e combinar as soluções em tempo constante.

- $T(1) = O(1)$
- $T(n) = 2 \times T(n-1) + O(1)$
- $T(n) = 2 \times (2 \times T(n-2) + O(1))$
- $T(n) = 2^2 \times T(n-2) + 2 \times O(1) + O(1)$
- $T(n) = 2^k \times T(n-k) + (\sum_{i=0}^{k-1} 2^i \times O(1))$
- Para $T(n-k) = T(1) \rightarrow k = n-1$
- $T(n) = 2^{n-1} \times T(1) + (\sum_{i=0}^{n-2} 2^i \times O(1))$
- $T(n) = 2^{n-1} \times O(1) + (2^{n+1} - 1) \times O(1)$
- $T(n) = O(2^{n-1}) + O(2^{n+1} - 1)$
- $T(n) = O(2^n)$

**2.c)** Algoritmo C soluciona problemas de tamanho n dividindo-os em 9 subproblemas de tamanho n/3, recursivamente soluciona cada subproblema e, então, combina as soluções em tempo quadrático.

- $T(n) = 9 \times T(\frac{n}{3}) + O(n^2)$
- $a = 9, b = 3, d = 2$
- $2 = \log_3 9$
- $T(n) = O(n^2 \times \log n)$

Sendo assim, ao comaprar a complexidade dos 3 algoritmos fica claro que o algoritmo C possui menor complexidade, por isso ele deveria ser escolhido.