R.JASHARI@constructor.university Redon Jashari

Problem 5.2

a) Brute-force Time Complexity

Since this includes each bit potentially getting multiplied by each bit of the other integer. This can lead to a nested loop so the time complexity would be $O(n^2)$.

This is due to brute force going through potentially every bit.

b) Divide and Conquer Algorithm

Here we use Karatsuba Algorithm:

We split the integer into two $\frac{n}{2}$-bit halves:

$X = X_L \cdot 2^{\frac{n}{2}} + X_R$

$Y = Y_L \cdot 2^{\frac{n}{2}} + Y_R$

We have the product of $X \times Y$

$(X_L \cdot 2^{\frac{n}{2}} + X_R) \times (Y_L \cdot 2^{\frac{n}{2}} + Y_R)$

$= X_L \cdot Y_L \cdot 2^n + (X_L \cdot Y_R + X_R \cdot Y_L) \cdot 2^{\frac{n}{2}} + X_R \cdot Y_R$

We split X and Y into right and left halves (X_L, X_R, Y_L, Y_R)

We compute:

$P_1 = X_L \times Y_L$

$P_2 = X_R \times Y_R$

$P_3 = (X_L + X_R) \times (Y_L \times Y_R)$

And we get:

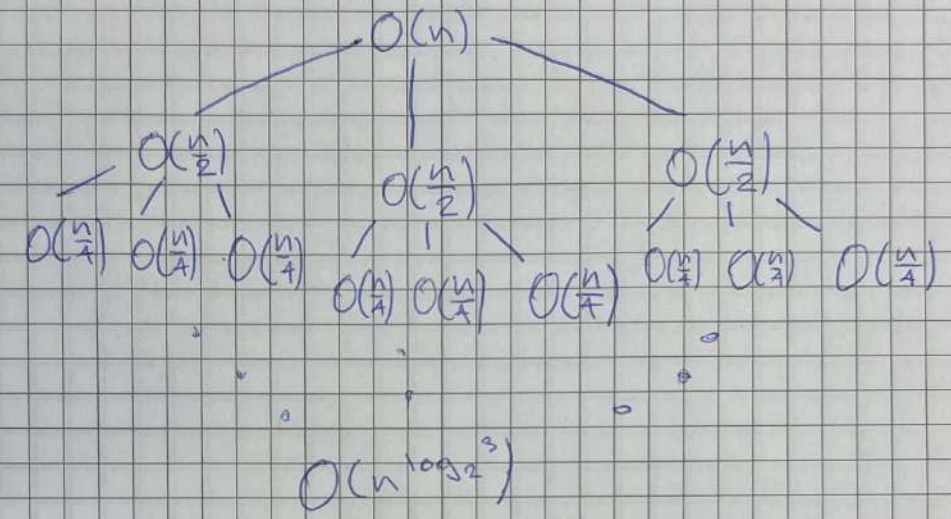
$X \times Y = P_1 \cdot 2^n + (P_3 - P_1 - P_2) \cdot 2^{\frac{n}{2}} + P_2$

c) Recurrence Relation

The recurrence relation for the divide and conquer algorithm is:

$$T(n) = 3T\left(\frac{n}{2}\right) + \Theta(n)$$

d) Recursion Tree



$$T(n) = O(n) + 3 \cdot O(n/2) + 3^2 \cdot O(n/4) + \ldots + 3^{\log_2(n)} \cdot O(1)$$

Dominant Term: $3^{\log_2(n)}$

$$3^{\log_2(n)} = n^{\log_2(3)}$$

$$T(n) = O(n^{1.585})$$

e) Solving Using the Master Theorem

$$T(n) = aT(n/b) + f(n)$$

$a = 3$

$b = 2$

$f(n) = O(n)$

$f(n)$ is $O(n^{\log_b a})$, so:

$$T(n) = O(n^{\log_2(3)}) \approx O(n^{1.585})$$