Karatsuba Algorithm - DSA Deep Dive

Sparsh Gupta

March 14, 2024

1 Introduction

The Karatsuba algorithm is a faster multiplication algorithm for integers, particularly helpful in large integer multiplications. The basic strategy behind the algorithm is a divide-and-conquer approach in which the traditional approach of multiplication of two numbers is converted into three multiplications of smaller numbers, each having about half as many digits as the original number.

2 Algorithm

Let us say we want to compute the multiplication of two numbers x and y in base r with n number of digits. For binary representation of numbers, r would be equal to 2 but for all numbers in decimal base, it is 10. For any positive integer ($\mathbb{Z}^+$) m less than n, we can obtain:

$$x = x_1 r^m + x_0$$
$$y = y_1 r^m + y_0$$

where x_0 and x_1 are the halves of x, and y_0 and y_1 are the halves of y, and x_0 and y_0 are less than r^m.

The product of x and y is, therefore,

$$\begin{aligned} xy &= (x_1 r^m + x_0)(y_1 r^m + y_0) \\ &= x_1 y_1 r^{2m} + (x_1 y_0 + x_0 y_1) r^m + x_0 y_0 \end{aligned}$$

Instead of four sub-problems/multiplications, we only have three multiplications using Karatsuba with some extra additions such that:

$$xy = z_2 r^{2m} + z_1 r^m + z_0$$

where

$$z_0 = x_0 y_0$$
$$z_1 = x_1 y_0 + x_0 y_1$$
$$z_2 = x_1 y_1$$

Now, we can assume that $z_3 = (x_1 + x_0)(y1 + y_0)$, and we know that $z_1 = x_1 y_0 + x_0 y_1$ which also gives us

$$z_1 = (x_1 + x_0)(y1 + y_0) - x_1 y_1 - x_0 y_0$$
$$z_1 = z_3 - z_2 - z_0$$

3 Example

We can look at an example for multiplication of large numbers 12345 and 54321. Here, $r = 10$ and we choose $m = 3$ such that we can decompose the numbers using $r^m = 1000$:

$$12345 = 12 \cdot 1000 + 345$$
$$54321 = 54 \cdot 1000 + 321$$

Now, we can compute the partial results z_0, z_1, and z_2.

$$\begin{aligned} z_0 &= 345 \cdot 321 = 110745 \\ z_2 &= 12 \cdot 54 = 648 \\ z_1 &= (12 + 345) \cdot (54 + 321) - z_2 - z_0 \\ &= 357 \cdot 375 - 648 - 110745 \\ &= 133875 - 648 - 110745 \\ &= 22482 \end{aligned}$$

Then, the final result is (we calculate it based on adding the partial sum terms shifted according to the base),

$$\begin{aligned} 12345 \cdot 54321 &= z_2 \cdot (r^m)^2 + z_1 \cdot (r^m)^1 + z_0 \cdot (r^m)^0 \\ &= 648 \cdot (1000)^2 + 22482 \cdot (1000)^1 + 110745 \cdot (1000)^0 \\ &= 648000000 + 22482000 + 110745 \\ &= 670592745 \end{aligned}$$

In this algorithm, to calculate the partial sums which are, in theory, smaller multiplications, we can do that by recursively calling the Karatsuba algorithm on those smaller numbers.

4 Time Complexity Analysis

For the traditional multiplication algorithm, if we are multiplying two n-digit numbers, we multiply each digit in that number with n digits of the other number. This results in n number of sub-products that are then added. Overall, when multiplying these numbers, we require n^2 single digit multiplications and this translates to a time complexity of $\Theta(n^2)$.

Karatsuba's algorithm has been shown to be most efficient when m is equal to $n/2$, rounded up. In this case, $n = 2^k$ where k is an integer and if we assume that we perform recursion until n becomes 1, we will observe that we perform 3^k single-digit multiplications, which is n^c such that $c = \log_2 3$.

Now, we can determine the runtime complexity $T(n)$ such that it denotes the total number of elementary operations required when multiplying two n-digit numbers using the algorithm.

$$T(n) = 3T(\lceil \frac{n}{2} \rceil) + O(n)$$

Using the master theorem, we can determine that

$$T(n) = \Theta(n^{\log_2 3}) \approx \Theta(n^{1.548})$$