Integer Multiplication using Divide and Conquer - A study

Surya Raghav B

June 2024

1 The Problem

We are two numbers x and y in binary representation and we have to produce the product. The elementary partial product method takes $O(n^2)$, where we multiply each digit of y separately by x, and add all the partial products up to get the result.

2 Designing the Algorithm

We can write the numbers as $x_1.2^{n/2} + x_0$, where x_1 corresponds to the higher $n/2$ bits, and x_0 corresponds to the lower $n/2$ bits. Thus we have

$$\begin{aligned} xy &= (x_1.2^{n/2} + x_0)(y_1.2^{n/2} + y_0) \\ &= x_1y_1.2^n + (x_1y_0 + x_0y_1).2^{n/2} + x_0y_0 \end{aligned} \tag{1}$$

This reduces single n-bit instance to four n/2-bit instances. Combining requires constant number of additions of $O(n)$-bit numbers. Thus the running time is

$$T(n) \leq 4T(n/2) + cn$$

But this implies again $T(n) \leq O(n^2)$, which is nothing better than the elementary solution. So we look at a way to reduce the number recursive calls to three.

Consider $(x_1 + x_0)(y_1 + y_0) = x_1y_1 + x_1y_0 + x_0y_1 + x_0y_0$, and compare it Equation 1, the outer terms can determined recursively, and the middle term can be determined by subtracting the outer terms from this product.

Algorithm 1 Recursive-Multiply(x,y):

Write $x = x_1.2^{n/2} + x_0$, and $y = y_1.2^{n/2} + y_0$
Compute $x_1 + x_0$ and $y_1 + y + 0$
$p =$ Recursive-Multiply($x_1 + x_0$, $y_1 + y_0$)
$x_1y_1 =$ Recursive-Multiply(x_1, y_1)
$x_0y_0 =$ Recursive-Multiply(x_0, y_0)
Return $x_1y_1.2^n + (x_1y_0 + x_0y_1).2^{n/2} + x_0y_0$

3 Analysing the Algorithm

Given two n-bit numbers it performs constant number of additions on $O(n)$-bit numbers in addition to the three recursive calls on the n/2-bit instances(The terms x_1+x_0 and y_1+y_0 may have n/2+1 bits, but does not affect the asymptotic results). So now our recurrence becomes

$$T(n) \leq 3T(n/2) + cn$$

for a constant c. The solution to this recurrence is

$$T(n) \leq O(n^{\log_2^3}) = O(n^{1.59})$$