A Mini Project Report on

Karatsuba's Algorithm for Fast Multiplication

for the subject

Design and Analysis of Algorithms

By

Himani Kalra (RA2011027010066)
Reeba Mercy Sebastian (RA2011027010072)

To subject in-charge

Mrs R. Radha

INTRODUCTION :

The Kratsuba Algorithm is a fast multiplication algorithm that uses a divide and conquer approach to multiply two numbers. It was discovered by Anatoly Karatsuba in 1960 and published in 1962.

This happens to be the first algorithm to demonstrate that multiplication can be performed at lower complexity than $O(N^2)$ which is by following the classical multiplication technique. Using this algorithm, multiplication of two n-digit numbers is reduced from $O(N^2)$ to $O(N^{\log 3})$ that is $O(N^{1.585})$

Being able to multiply numbers quickly is very important. Computer scientists often consider multiplication to be a constant time $O(1)$ operation, and this is a reasonable simplification for smaller numbers; but for larger numbers, the actual running times need to be factored in, which is $O(n^2)$. The point of the Karatsuba algorithm is to break large numbers down into smaller numbers so that any multiplications that occur happen on smaller numbers. Kratsuba can be used to multiply numbers in all base systems (base 10, base 2, etc.)

As an example, the Karatsuba algorithm requires $3^{10} =$ 59,049 single-digit multiplications to multiply two 1024-digit numbers ($n = 1024 = 2^{10}$), whereas the classical algorithm requires $(2^{10})^2 = 1,048,576$ single-digit multiplications.

The key idea is to reduce the four-sub-problems in multiplication to three sub-problems. Thus, on calculating the three unique sub-problems, the original four sub-problems are solved using addition or subtraction operation. Hence, the speed up.

KARATSUBA ALGORITHM : Uses DIVIDE & CONQUER

Karatsuba is a divide and conquer algorithm that reduces the multiplication of two n-digit numbers to three multiplications of n/2 -digit numbers and, by repeating this reduction, to at most $n^{\log_2 3} \approx n^{1.58}$ single-digit multiplications.

DIVIDE and CONQUER ALGORITHM :

this technique can be divided into the following three parts :

① DIVIDE : this involves dividing the problem into smaller sub-problems.

② CONQUER : Solve sub-problems by calling recursively until solved.

③ COMBINE : Combine the sub-problems to get the final solution of the whole problem.

ALGORITHM:

The Basic principle of Karatsuba's algorithm is Divide-and-conquer, using a formula that allows one to compute the product of two large numbers x and y using three multiplications of smaller number, each with about half as many digits as x or y, plus some additions and digit shifts.

This basic step step is, in fact, a generalization of a similar complex multiplication algorithm, where the imaginary unit i is replaced by a power of the base.

Let x and y be represented as n-digit strings in some base B. For any positive integer m less than n, one can write the two given numbers as,

$$x = x_1 B^m + x_0,$$
$$y = y_1 B^m + y_0,$$

where x_0 and y_0 are less than B^m. The product is then

$$xy = (x_1 B^m + x_0)(y_1 B^m + y_0)$$
$$= x_1 y_1 B^{2m} + (x_1 y_0 + x_0 y_1) B^m + x_0 y_0$$
$$= z_2 B^{2m} + z_1 B^m + z_0,$$

where,

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

These formula require four multiplication. Karatsuba observed that xy can be computed in only three multiplications, at the cost of a few extra additions. With z_0 and z_2 as before one can observe that

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

PSEUDOCODE

```
procedure karatsuba (num1, num2)
    if (num1 < 10) or ( num2 < 10)
        return num1*num2

* calculates the size of the numbers *
  m = max(size_base10 (num1) , size - base 10 (num2))
  m2 = m/2

* split the digit sequences about the middle*
  high1 , low1 = split_at(num1, m2)
  high 2,  low2  = split_at(num2, m2)

* 3 calls made to numbers approximately half the size*
  z0 = karatsuba (low1, low2)
  z1 = karatsuba(( low1 + high1), (low2 + high 2))
  z2  = karatsuba ( high1 , high2)
  return (z2 * 10 ^ (2 * m2)) + ((z1 -z2 -z0 ) * 10 ^ (m2)) +(z0)
```

EXAMPLE:

Perform the following Multiplication using Karatsuba's Multiplication Method:

1234 X 4321

First determine the a value for step 1. This will contain the high bits of x and y since x and y have 4 bits and the left most 2 bits are the high bits.

$a_1 = 12 \times 43$

Note, we will have to call the karatsuba algorithm on a_1 since a multiplication is necessary to obtain the value. Before we recurse, through let's find d_1 and e_1.

d contain the lower bits of in this problem since x and y are 4 bits, and the lower bits in this problem are the two right-most bits.

$d_1 = 34 \times 21$

We will also have to recurse on d_1 to obtain the value

$e_1 = (x_H + x_L)(y_H + y_L) - a - d$

$e_1 = (12+34) \times (43+21) - a_1 - d_1$

Now we are stuck and can't simplify e_1 further until we have the value of a_1 and d_1

Solving for a_1

$a_1 = 12 \times 43$

$a_2 = 1 \times 4 = 4$

$d_2 = 2 \times 3 = 6$

$e_2 = (1+2)(4+3) - a_2 - d_2$

$= (1+2)(4+3) - 4 - 6$

$= 11$

Now,

$$xy = ar^n + er^{n/2} + d$$

therefore,

$a_1 = 12 \times 43 = 4 \times 10^2 + 11 \times 10 + 6 = 516$

Solving for d_1:

we have

$d_1 = 34 \times 21$

$a_2 = 3 \times 2 = 6$

$d_2 = 4 \times 1 = 4$

$e_2 = (3+4)(2+1) - a_2 - d_2$

$= 11$

$xy = ar^n + er^{n/2} + d$

$d_1 = 34 \times 21 = 6 \times 10^2 + 11 \times 10 + 4 = 714$

Solving for e_1:

$e_1 = (46 \times 64) - a_1 - d_1$

$a_2 = 4 \times 6 = 24$

$d_2 = 6 \times 4 = 24$

$e_2 = (4+6)(6+4) - a_2 - d_2$

$= 52$

$e_1 = (46 \times 64) - a_1 - d_1$

$= 24 \times 10^2 + 52 \times 10 + 24 - 714 = 1714$

Now we can get the answer to the original problem as follows:-

we have

$a_1 = 516$, $d_1 = 714$, $e_1 = 1714$

plugging into $xy = ar^n + ar^{n/2} + d$

we get

$xy = (516)10^4 + (1714)10^2 + 714 = 5,332,114$. Ans.

Example 2 :

Consider the following multiplication : 47 x 78

$x = 47$

$x = 4 * 10 + 7$

$x1 = 4$

$x2 = 7$

$y = 78$

$y = 7 * 10 + 8$

$y1 = 7$

$y2 = 8$

$a = x1 * y1 = 4 * 7 = 28$

$c = x2 * y2 = 7 * 8 = 56$

$b = (x1 + x2)(y1 + y2) - a - c = 11 * 15 - 28 - 56$

11 * 15 can in turn be multiplied using Karatsuba Algorithm

TIME COMPLEXITY ANALYSIS:

To analyze the complexity of the Karatsuba algorithm, consider the number of multiplication the algorithm perform as a function of n, $M(n)$. Recall that the algorithm multiplies together two n-bit numbers. If $n = 2^k$ for some k, then the algorithm recurses three ~~try~~ times on $\frac{n}{2}$-bit

The recurrence for this is

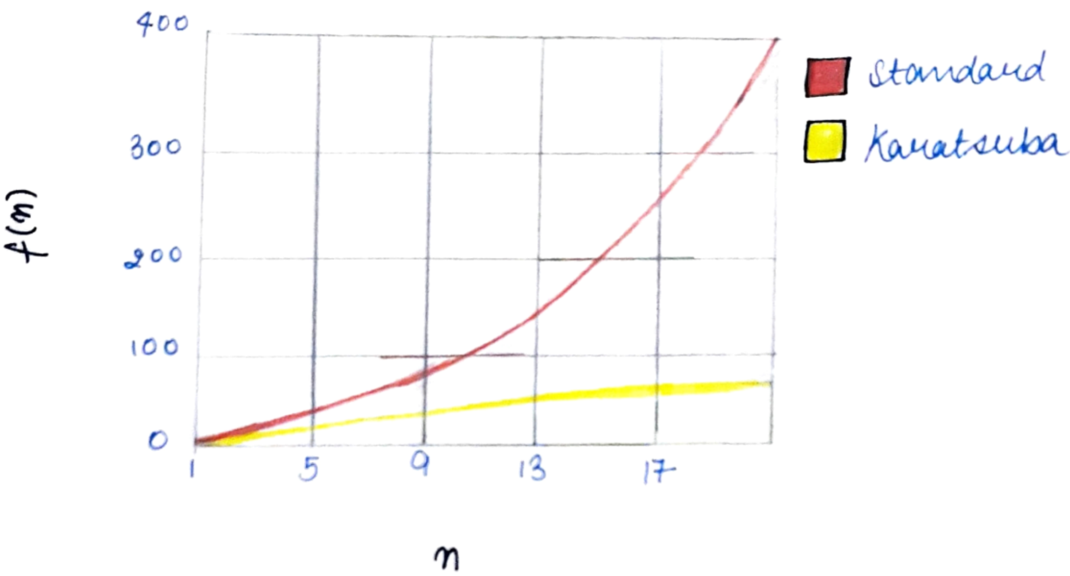
$$M(n) = 3M\left(\frac{n}{2}\right)$$

This takes care of the multiplications required for Karatsuba -- now consider the addition and subtraction. There are $O(n)$ additions and subtractions required for the algorithm. Therefore, the overall recurrence for the Karatsuba algorithm is

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

using Master's theorem on the above recurrence yields that the running time of the Karatsuba algorithm is

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



$\therefore$ $O(n^2)$ grows much faster than $O(n^{\log_2 3})$

APPLICATION :

The Karatsuba algorithm is very efficient in tasks that involve integer multiplication. It can also be useful for polynomial multiplications.

CONCLUSION :

Therefore we have used the Karatsuba's Algorithm for fast multiplication to find the product of two n digit numbers where the no. of digits of both the numbers can be same or different.

By using this algorithm we have explored a new application of divide and conquer Design Technique and this is faster than the classical method of multiplication

```python
from math import ceil, floor
#math.ceil(x) Return the ceiling of x as a float, the smallest integer value greater than or equal to x.
#math.floor(x) Return the floor of x as a float, the largest integer value less than or equal to x.

def karatsuba(x,y):
    #base case
    if x < 10 and y < 10: # in other words, if x and y are single digits
        return x*y

    n = max(len(str(x)), len(str(y)))
    m = ceil(n/2)   #Cast n into a float because n might lie outside the representable range of integers.

    x_H  = floor(x / 10**m)
    x_L = x % (10**m)

    y_H = floor(y / 10**m)
    y_L = y % (10**m)

    #recursive steps
    a = karatsuba(x_H,y_H)
    d = karatsuba(x_L,y_L)
    e = karatsuba(x_H + x_L, y_H + y_L) - a - d

    return int(a*(10**(m*2)) + e*(10**m) + d)

#Driver code
print("\n\n Enter any two Numbers: ")
x = int(input())
y= int (input())
print("The Product of the two numbers is : "+str(karatsuba(x,y)))
print("\n\n")

```

Source Code:

Output:

```
 Enter any Numbers 1:
1234


 Enter any Numbers 2:
4321
The Product of the two numbers is : 5332114
```

```
 Enter any Numbers 1:
47


 Enter any Numbers 2:
78
The Product of the two numbers is : 3666
```

References

- Demaine, E., Indyk, P., & Kellis, M. *Karatsuba's Algorithm*. Retrieved May 29, 2016, from http://courses.csail.mit.edu/6.006/spring11/exams/notes3-karatsuba

- Babai, L. *Divide and Conquer: The Karatsuba–Ofman algorithm*. Retrieved May 30, 2016, from http://people.cs.uchicago.edu/~laci/HANDOUTS/karatsuba.pdf
- https://en.wikipedia.org/wiki/Karatsuba_algorithm
- https://courses.csail.mit.edu/6.006/spring11/exams/notes3-karatsuba
- https://iq.opengenus.org/karatsuba-algorithm/
- https://www.geeksforgeeks.org/karatsuba-algorithm-for-fast-multiplication-using-divide-and-conquer-algorithm/