DAA432C
Assignment 1

Multiplication of two integers of arbitrary digits by only using multiplication routine for single digit numbers

Prepared By

Sheldon Tauro
Aswin VB
Avinash Yadav
Nistala Venkata Sharma

Indian Institute Of Information Technology
Allahabad

2018

Multiplication of two integers of arbitrary digits by only using multiplication routine for single digit numbers

Sheldon Tauro*, Aswin VB†, Avinash Yadav‡ and N.V.K. Sharma§
Indian Institute of Information Technology,Allahabad
Allahabad-211012
Email: *iit2016137@iiita.ac.in, †iit2016106@iiita.ac.in, ‡itm2016004@iiita.ac.in, §ism2016005@iiita.ac.in

I. Introduction and Literature Survey

If a positional *numeral system* is used, a natural way of multiplying numbers is taught in schools as long multiplication, sometimes called grade-school multiplication, sometimes called Standard Algorithm: multiply the multiplicand by each digit of the multiplier and then add up all the properly shifted results. It requires memorization of the *multiplication table* for single digits.
This is the usual algorithm for multiplying larger numbers by hand in base 10. Computers initially used a very similar shift and add algorithm in base 2, but modern processors have optimized circuitry for fast multiplications using more efficient algorithms, at the price of a more complex hardware realization. A person doing long multiplication on paper will write down all the products and then add them together; an abacus-user will sum the products as soon as each one is computed.

```
     23958233
×        5830
———————————————
     00000000 ( =      23,958,233 ×     0)
    71874699  ( =      23,958,233 ×    30)
   191665864  ( =      23,958,233 ×   800)
+ 119791165   ( =      23,958,233 × 5,000)
———————————————
  139676498390 ( = 139,676,498,390        )
```

Fig. 1.

A *numeral system*(or system of numeration) is a writing system for expressing numbers; that is, a mathematical notation for representing numbers of a given set, using digits or other symbols in a consistent manner. It can be seen as the context that allows the symbols "11" to be interpreted as the binary symbol for three, the decimal symbol for eleven, or a symbol for other numbers in different bases.
In mathematics, a *multiplication table* (sometimes, less formally, a times table) is a mathematical table used to define a multiplication operation for an algebraic system.
The decimal multiplication table was traditionally taught as an essential part of elementary arithmetic around the world, as it lays the foundation for arithmetic operations with base-ten numbers.

II. Algorithm Design

Data: a[] is multiplicand and b[] is multiplier
Result: ans[] having the result after multiplication

$carry \leftarrow 0$
$index \leftarrow 0$
if (n1<n2) **then**
  for $i \leftarrow (n1-1)$ to 0 **do**
    $index \leftarrow (n1-i-1)$
    for $j \leftarrow (n2-1)$ to 0 **do**
      $var \leftarrow (a[i]*b[i]) + carry + ans[index]$
      $carry \leftarrow var/10$
      $ans[index] \leftarrow var\%10$
      $index \leftarrow index + 1$
    end for
    $ans[index] \leftarrow ans[index] + carry$
    $carry \leftarrow 0$
  end for
else
  for $i \leftarrow (n2-1)$ to 0 **do**
    $index \leftarrow (n2-i-1)$
    for $j \leftarrow (n2-1)$ to 0 **do**
      $var \leftarrow (a[i]*b[i]) + carry + ans[index]$
      $carry \leftarrow var/10$
      $ans[index] \leftarrow var\%10$
      $index \leftarrow index + 1$
    end for
    $ans[index] \leftarrow ans[index] + carry$
    $carry \leftarrow 0$
  end for
end if

$Explaination$

We have a variable $carry$ which stores the carry produced in each single integer multiplication and an $index$ variable which denotes the index to which the answer of single digit multiplication has to be stored.

In the algorithm we have an if else statement depending on the number of digits in the multiplicand and multiplier.inside both if and else statement we have two for loops which denotes the multiplication of each digit of the multiplicand with the digits of multiplier. When the control flow enters the outer loop it calculates the index in the ans array into which the calculation has to be stored. In the inner for loop the single digit multiplication is performed and the result is added with the carry(if there is any carry that has been produced in the previous iteration else it will be zero). New carry is calculated and stored in the $carry$ variable and the lsb of the result is stored in the ans array pointed by index variable and the $index$ variable is incremented by 1. Once the inner loop is completed the $carry$ variable is reinitialized to 0 again.

The final answer will be present in the ans array

III. Analysis

n1 is the number of digits of multiplicand AND n2 is the number of digits of multiplier. Assuming n1 is smaller than n2 i.e. number of digits in multiplicand is smaller than number of digits in multiplier.Our time analysis is as described below.

- $time_{multiplicationpart} \propto min(n1, n2) * (13 + 21 * max(n1, n2))$
  $time_{multiplicationpart} \propto 21(n1 * n2) + 13 * min(n1, n2)$
- $time_{restpart} \propto 4$

$t_{total} \propto 21(n1 * n2) + 13 * n1 + 4$

1) Time Analysis:

$$time_{best} <= t_{average} <= t_{worst}$$

$$f(n) = 21(n1 * n2) + 13 * min(n1, n2) + 4$$

$$\Omega(f(n)) = O(f(n))$$

hence,

$$time_{best} = t_{average} = t_{worst}$$

- The above equation of time analysis shows that there is no difference in computing worst case or best case.This is because in the code itself there is no conditional statement which drives every case to take the same time.

IV. Experimental study

$$TimeComplexity$$

As seen in the previous section,

$O(f(n)) \propto$ O(n1*n2)

since

$$time_{best} = t_{average} = t_{worst} = t$$

Then Time Complexity is $O(n1 * n2)$, where $n1$ and $n2$ are length of two number that need to be multiplied.

Test Cases				
n1	523	348	644	884
n2	48	727	963	14
t	527812	5317444	13031988	260082

- When we plot for $n1 * n2$ vs t It can be noted that, the graph is linear as in fig(2)
- When we plot for $(n1 + n2)/2$ vs t It can be noted that, the graph is a parabola as in fig(3).

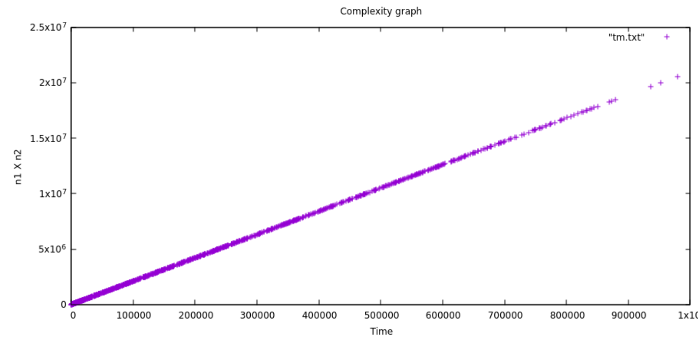

Fig. 2. Complexity Graph between n1*n2 vs time.

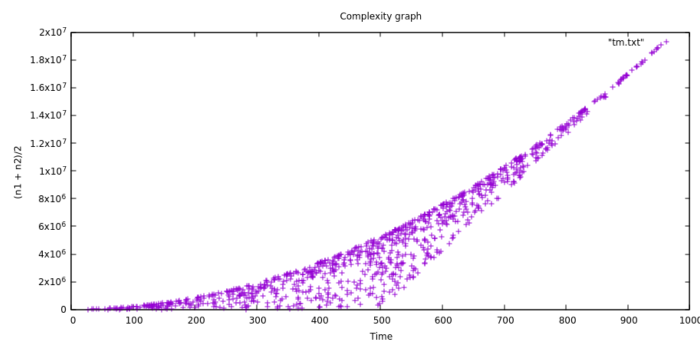

Fig. 3. Complexity graph between (n1+n2)/2 and time.

V. Discussions

We start from last digit of second number multiply it with first number. Then we multiply second digit of second number with first number, and so on. We add all these multiplications. While adding, we put i-th multiplication shifted.

The approach used in below solution is to keep only one array for result. We traverse all digits first and second numbers in a loop and add the result at appropriate position.

VI. Conclusion

Multiplication of two integers of arbitrary digits by only using multiplication routine for single digit numbers.In this problem we used the long multiplication algorithm in which the multiplier and the multiplicand is taken in two arrays and each digit present in the multiplicand array is multiplied one by one with the digits of multiplier array and the result is stored in an answer array. On plotting the graph on the average of the number of digits in the multiplier and multiplicand and the time it is observed that the graph is a parabola and on plotting the graph on product of the number of digits of multiplier and multiplicand and the time it is observed that the graph is a straight line.

REFERENCES

[1] https://en.wikipedia.org/wiki/Multiplicationalgorithm