

ALGORITHM Multiplication Analyses

By: *Dmitrii Sokolov*

Group: 193-1

Submission Date: *26th April 2020*

Supervisor: *Aleksey Varenikov*

Problem statement

Goal:

To implement 3 different algorithms of multiplication using C++ languages , estimated their work time by construction a graph basing on data that program provides .

Complexity of Algorithms :

1. **Great School Multiplication algorithm** : $O(n^2)$
2. **Karatsuba** : $O(n^{\log(2,3)})$, $T(n) = 3T(n/2) + O(n)$.
3. **Divide and Conquer** : $T(n) = 4T(n/2) + O(n)$, By Master Theorem $O(n^{\log(2,4)})$ or $O(n^2)$.

Implementation of Algorithms

Great School Algorithm

Just implemented the usual way or multiplying , by multiplying each digit of first number on last digit of the second number , then on pre-last and etc.

Divide & Conquer

Did multiplication with following formula : $x1 * x2 = ac * 10^n + ad * 10^{(n/2)} + bc * 10^{(n/2)} + bd$; where a, b are halves of x1 and c, d are halves of x2, and n is the maximum length of multiplied numbers.

Karatsuba

Karatsuba is kind of derivative of D&C , it just a reduce of amount of operators . After reduce we got following formula : $x1x2 = ac * 10^n + (ad+bc) * 10^{(n/2)} + bd$.

Code

I implemented class Number via `std::string` for storing numbers of unlimited length and provide all the necessary semantic for working with them . I did only necessarily overloading , and i want to **pay your attention** , overloading of `*` in Number class is used only for multiplication of two numbers less then 10 (1 digit length) .

Multiplicator class is performed with virtual destructor and virtual function multiply which begin override in derived classes GSM, Karatsuba , Divide&Conquer . Function for generating random number is marked static

In main.cpp I just made function to conduct experiments and write results in csv file .

As we need to call each function of multiplication 3 times ant take average I made following decomposition :

```
typedef std::unique_ptr<Multiplier> M_ptr;

double call_multiply(M_ptr methd, Number& fst, Number& scnd)
{
    clock_t time1;
    double timefinal = 0;
    for(int i = 0; i < 3 ;i++)
    {
        time1 = clock();
        Number c = methd->multiply(fst,scnd);
        time1 = clock() - time1;
        double time_taken1 = ((double)time1) / CLOCKS_PER_SEC;
        timefinal += time_taken1;
    }
    return timefinal / 3;
}
```

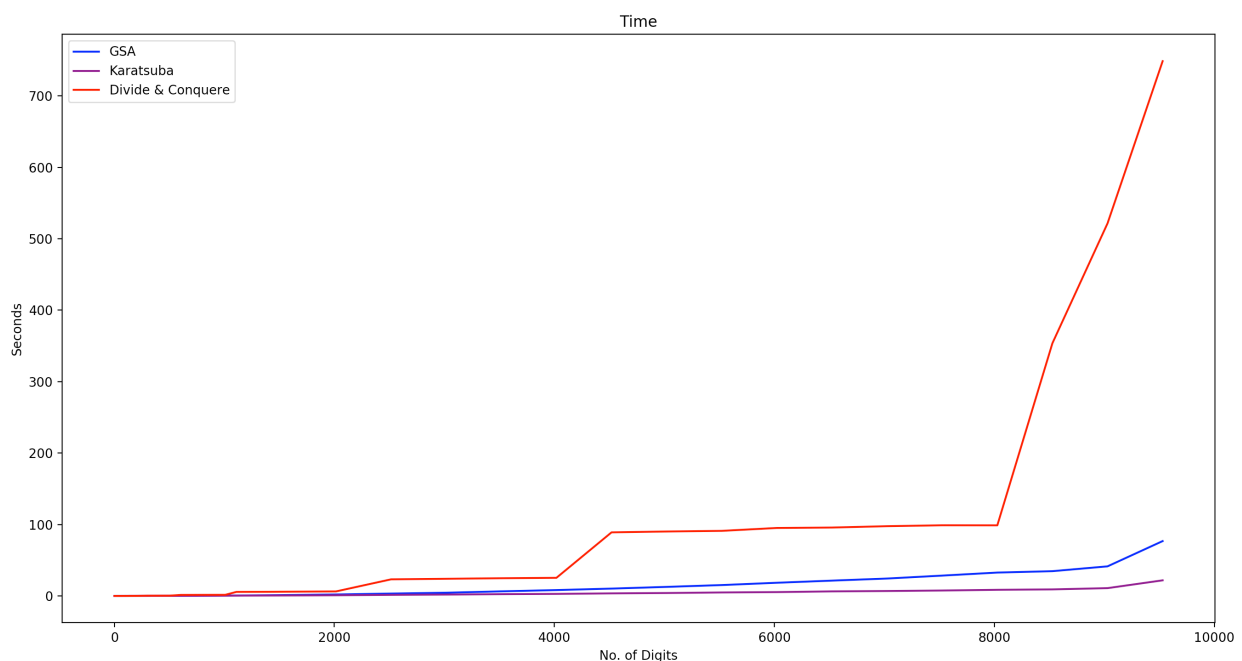
I'm passing a smart pointer to the function I need to call ,as a parameter of `call_multiply` .

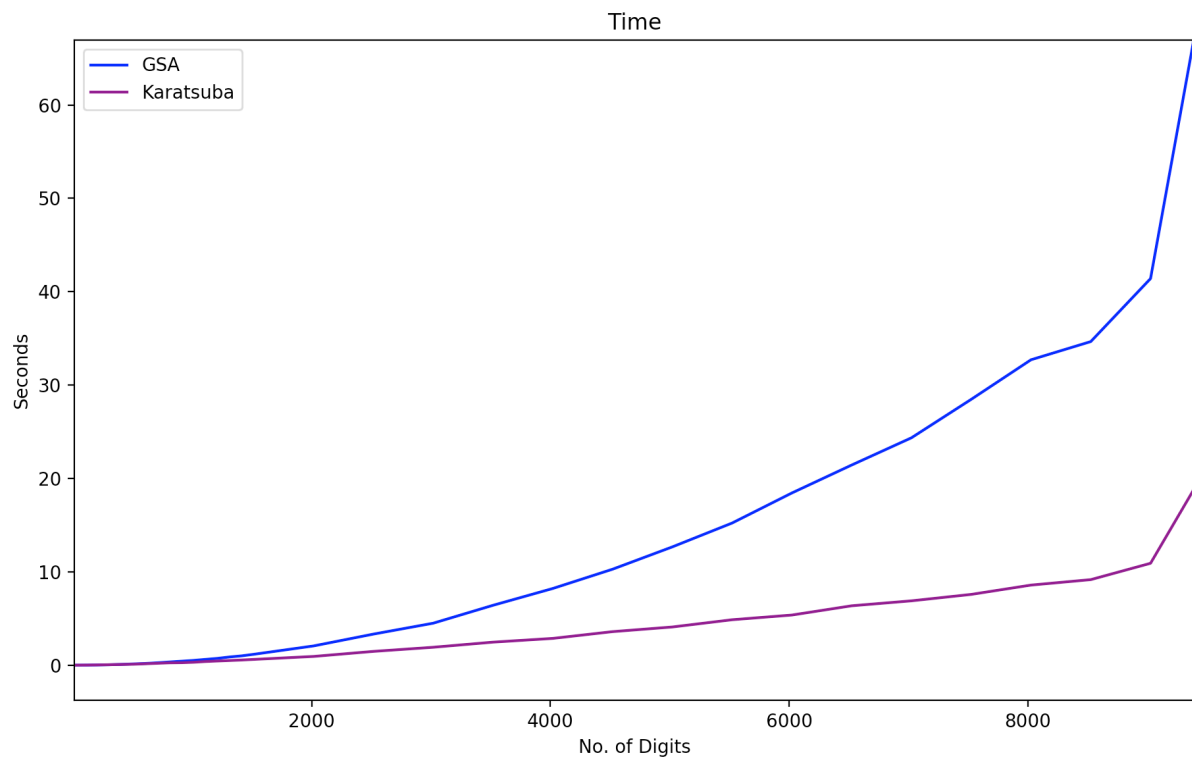
Python part .

Simple code for building graphs with matplotlib, and getting data from csv with panda library . Simplest part of the workshop)

Result

As a result you can observe my graphs which provides with all necessarily information :





Conclusion

As a result , you can observe csv file attached to my project with information about algorithm run time , and visual representation of that data via graph above . I could have improved my project by spending more time on code style and implementing D&C in more efficient way , and in my personal opinion changing base case in Karatsuba and calling GSA in base case would lead to more effective results .

URL to repository :

<https://github.com/SKYMAN44/dsba--ads2020--hw1>