Lab Report

Title: Karatsuba Algorithm for Multiplication
Date: September 7th, 2024
Name: William R. Vanderhoff
Student ID: 651461102

1. Introduction to the Lab

Objective: Understand the Karatsuba algorithm and implement it correctly.

The original grade school Multiplication Algorithm is not the most efficient solution with an efficiency of $O(n)$. In 1960, Anatoly Karatsuba found an algorithm which had an efficiency of $O(n^{1.6})$, this opened the door for research into further algorithmic analysis which has produced breakthroughs for multiplication up to the most recent in 2019 where Harvey and van der Hoeven came up with the galactic algorithm with a complexity of $O(n \log n)$. The Multiplication algorithm and divide and conquer method use 4 multiplications, while the Karatsuba only uses 3 and some addition as will be shown in the mathematical analysis. The Karatsuba algorithm can be broken down to 5 basic steps which can be used recursively.

Problem Statement: Understand the problem using to show that when n is sufficiently large enough, the Karatsuba algorithm is faster compared to the Multiplication algorithm.

2. Lab Results

All Goals Achieved: Yes
All Tests Passed: Yes

3. Algorithm Framework

The first step in solving the problem, is to understand the grade-school multiplication algorithm. The grade school multiplication algorithm is as follows:

$$n = x \times y$$

The values of x and y are split into two halves:

$$x = (A \times 10^m + B)$$

$$y = (C \times 10^m + D)$$

Where A, B, C, D are parts of the number and m is the half the length of the original numbers. In the grade-school algorithm, the parts are then multiplied directly:

$$x \times y = (A \times 10^m + B) \times (C \times 10^m + D)$$

This results in the following:

$$x \times y = (A \times C)10^{2m} + (A \times D)10^{m} + (B \times C)10^{m} + (B \times D)$$

Which requires four multiplications, $A \times C, A \times D, B \times C, B \times D$. This is not ideal, as when analyzed the complexity is $O(n^2)$, *(based on the complexity analysis provided from Dr. Xiang Huang in his CSC482 Lecture 2)*, However, the Karatsuba algorithm performs more efficiently for a sufficiently large n.

To understand the Karatsuba algorithm, I found a break down of the steps on YouTube as provided by Stanford Algorithms in (2017) which truly opened up my mind to this approach. The steps to use the algorithm when solving for n where $n = x \times y$ start after the split of the two halves:

$$x = (a \times 10^m + b)$$
$$y = (c \times 10^m + d)$$
$$a \times c \tag{1}$$
$$b \times d \tag{2}$$
$$(a + b) \times (c + d) \tag{3}$$
$$equation3 - equation2 - equation1 \tag{4}$$
$$n = equation1 \times 10^4 + equation2 + equation4 \times 10^2 \tag{5}$$

This sudoMath/English version of the problem truly helped me to understand the overall solution. From these steps, the recursion can be initiated to solve large multiplications. The base case is that when x or y equal a length of one, return the single multiplication of these two numbers. The numbers provided in x and y can be cut in half until the base case is reached. When the base case is reached, the algorithm will kick in and do the work required to add up all the numbers till it completes the recursion. The actual mathematical formula for the Karatsuba algorithm is shown here:

$$n = x \times y$$

Split the values of x and y:

$$x = (a \times 10^m + b)$$
$$y = (c \times 10^m + d)$$

Multiply x and y:

$$x \times y = (a \times 10^m + b) \times (c \times 10^m + d)$$
$$= (a \times c)10^{2m} + (a \times d)10^m + (b \times c)10^m + (b \times d)$$

At this point, we will reduce the number of multiplications we have to do from four to 3 by introducing an equation which replace $(a \times d)10^m + (b \times c)10^m$ in our equation. The equation is as follows:

$$T = (a + b) \times (c + d) - b \times d - a \times c$$

$$= ac + ad + bc + bd - bd - ac$$

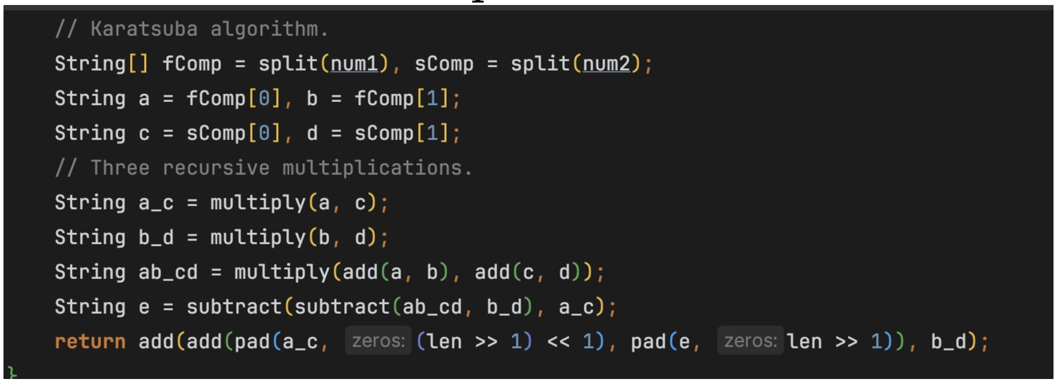
Which can be reduced by reduced to:

$$T = ad + bc$$

This will be inserted into the equation from earlier replacing $(a \times d)10^m + (b \times c)10^m$ so that n can be solved using the Karatsuba algorithm as follows when x is split into a and b, y is split into c and d, and m is the half the length of the original numbers.:

$$n = ac10^{2m} + (ad + bc)10^m + bd$$

Java Implementation



```
    // Karatsuba algorithm.
    String[] fComp = split(num1), sComp = split(num2);
    String a = fComp[0], b = fComp[1];
    String c = sComp[0], d = sComp[1];
    // Three recursive multiplications.
    String a_c = multiply(a, c);
    String b_d = multiply(b, d);
    String ab_cd = multiply(add(a, b), add(c, d));
    String e = subtract(subtract(ab_cd, b_d), a_c);
    return add(add(pad(a_c, zeros: (len >> 1) << 1), pad(e, zeros: len >> 1)), b_d);
}
```

4. Complexity Analysis

Skipped the optional complexity analysis due to time constraints.

- **Time Complexity (Worst Case):** $\mathrm{O}(n^{\log_2 3}) \approx \mathrm{O}(n^{1.585})$

Explanation: The Karatsuba algorithm optimizes the standard grade-school multiplication by reducing the number of recursive multiplications. Instead of performing four multiplications for two large numbers, it only requires three recursive multiplications. This leads to a recurrence relation of $T(n) = 3T\left(\frac{n}{2}\right) + O(n)$ which, when solved using the master theorem, results in a time complexity of $O\left(n^{\log_2 3}\right)$ While the algorithm is more efficient than the traditional $O(n^2)$ approach, it is still not linear due to the recursive nature of the problem.

5. Lessons Learned, Feedback, and Conclusion

- **Challenges Faced:** A challenge that became prevalent for me was understanding the math behind the karatsuba algorithm. Although simple, I have a difficult time understanding an equation without running a number through the equation and seeing how each section works. The video I found on YouTube helped me understand the problem thoroughly by showing implementation of the algorithm before trying to explain it. From there I was able to work out the actual algorithm.
- **Insights Gained:** My 'aha' moment was that the secret for me to understand the mathematics is taking the time to break down each section of the algorithm to understand it fully.

6. References and Use of Tools

ChatGPT/Copilot Usage:

- Used ChatGPT to explain StringBuilder from the JAVA solution on LeetCode.
- Used ChatGPT to try and improve efficiency of the solution, but there were no working available solutions that it designed.

Other References:

[Standford Algorithms]. (2017, January 27). *1 3 Karatsuba Multiplication 13 min*[Video]. YouTube. https://www.youtube.com/watch?v=JCbZayFr9RE

[oluwasayo]. (2015, November 22). *Java Implementation of the Karatsuba Algorithm: $O(n^1.585)$*. LeetCode. https://leetcode.com/problems/multiply-strings/solutions/17626/java-implementation-of-the-karatsuba-algorithm-o-n-1-585/