Joey Troyer

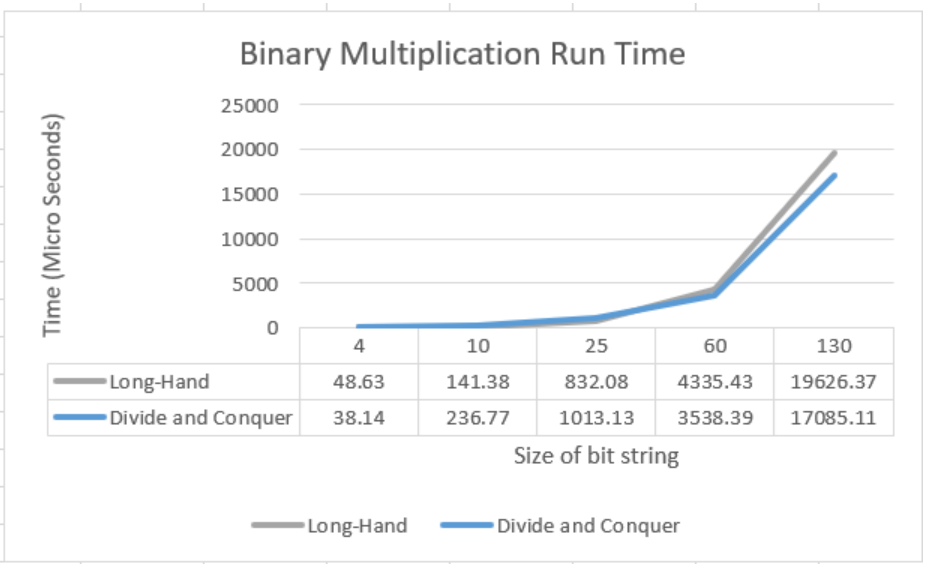
Vishwanathan Roopa

CS 372

26 February 2023

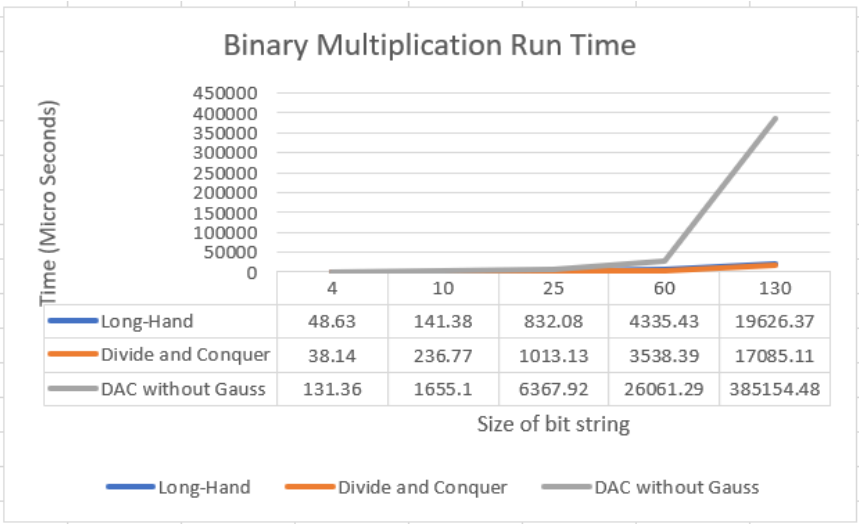
Lab2

For this lab, our primary objective was to comprehend the "divide and conquer" algorithm a widely used technique in computer science. Our task was to design a program that multiplies binary numbers using two distinct approaches. Firstly, we used the traditional long multiplication technique, which is the same if you were to multiply by hand. Secondly, we leveraged the power of the divide-and-conquer algorithm and utilized the Gauss trick to increase the efficiency of the algorithm.



In the following analysis, we compare the run times of the divide and conquer approach with the long-hand division for various bit sizes. For most smaller bit sizes, the long-hand division outperformed the divide and conquer method by a couple hundred microseconds but they were more or less the same. However, as the bit size increased, the divide and conquer method

demonstrated much better performance. For instance, at a bit size of 60, the divide and conquer approach was 1000 microseconds faster. At a bit size of 130, the time difference was even greater, with the divide-and-conquer approach outperforming the long-hand division by over 2500 microseconds. It is reasonable to expect that this time difference will become more pronounced as the bit string size continues to grow.



In this graph, I have run times of all algorithm implementations, including the divide and conquer method, without the gaussian trick. I decided to create a separate graph for this method since removing the gaussian trick resulted in significantly longer run times for larger numbers, which made the comparison of the other two graphs appear insignificant. For instance, when comparing the divide and conquer approach with a size of 130 to the DAC method without the gaussian trick, there was a difference of 215069 microseconds or 0.2 seconds in their run times. This difference can lead to considerably slower run times at a larger scale.

To verify the running time of the divide and conquer with an arity of 4 we can use the master

theorem $T(n) = aT(n/b) + f(n)$. Each level of the recursion tree corresponds to a subproblem of size n/2, and at each level we have to do 4 multiplications.

So we get the formula:

$T(n) = 4T(n/2) + n$

We have $a = 4$, $b = 2$ and $f(n) = n$

$\log_b(a) = \log_2(4) = 2$

Since $n^1 < n^2 \longrightarrow 1 < 2$, this means we are in case 3

So $T(n) = O(n^{\log_2(4)})$ and we get $T(n) = O(n^2)$

So we can say the running time is $O(n^2)$