Algorithms 2 Project 2

Ahmed Burak Gulhan
150160903

1)
In this project we have to multiply two n digit numbers in an efficient way. We use a recursive function to multiply. In the classical method we have 4 multiplication operations in each step. In the efficient algorithm we have 3.

2)
To multiply a n digit number we use the divide and conquer method to find an efficient algorithm. We divide both the operands in half to get a smaller sub problem. The amount of sub-problems for each step differ according to the algorithm we use. We then recursively divide until we get two 1-digit operands (in binary). Multiplying these 1 digit operands takes constant time (O(1)). Our next step is to combine the results of the sub problems. We do this in two different ways, again depending the algorithm used.

For the classical method: Assume our operands are ab and cd.

We know that the result of ab*cd (in binary) is:
$a*c*2^2 + (b*c + a*d)*2 + d*b$
We can see that there are 4 multiplications needed (a*c, b*c, a*d, d*b). We don't count the 2^2 and 2 multiplications since they can be done by shifting (which is constant time).
We can define this problem as T(n) = 4*T(n/2) + c
where c is a constant.

For the efficient method (karatsuba): our operands are xLxR and yLyR
the result of xLxR yLyR is given as:
$(P1*2^2) + ((P3-P1-P2)*2) + P2$
Where P1 = xL*yL
P2 = xR*yR
P3 = (xL+xR) * (yL+yR)

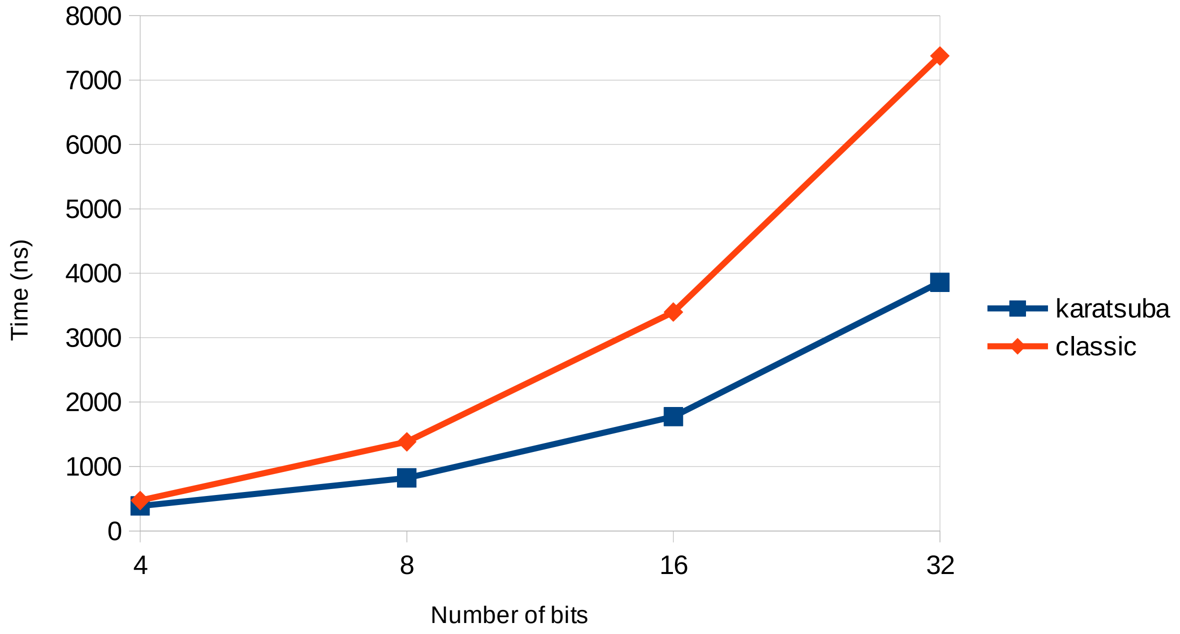
We can see that we only have to de 3 multiplications for each step.
We can define this problem as T(n) = 3*T(n/2) + c
where c is a constant

3)
Efficient algorithm (karatsuba):
T(n) = 3*T(n/2) + c
time complexity (using master method) = $\Theta(n^{lg3}) \sim= \Theta(n^{1.58})$

Classical algorithm
T(n) = 4*T(n/2) + c
time complexity (using master method) = $\Theta(n^2)$

(n is the digit count of the operands)

4)



(karatsuba is the given algorithm)

We can see that the classical algorithm increases at a faster rate compared to the efficient (karatsuba) algorithm. The rate of increase for the classical algorithm is $\Theta(n^2)$. The rate of increase for the efficient (karatsuba) algorithm is $\Theta(n^{1.58})$