COMP3121: Assignment 2 – Q1

Gerald Huang
$z5209342$
Updated: June 29, 2020

Let M and n be positive integers. We will calculate M^n in $O(\log n)$ many multiplications by applying the **exponentiation by squaring** method. This is a divide-and-conquer type recursive algorithm that breaks the problem from n bits of information to $n/2$ bits of information with each recursive call.

To fully understand this algorithm, we need to understand the underlying mathematical concept that holds this algorithm together. For any positive integers x, y, we have the equality

$$x^y = \left(x^{2\times y/2}\right) = \left(x^2\right)^{y/2}.$$

So we will efficiently square the base and halve the exponent all at once!

Now consider different cases for n. Suppose that n is even. Then break up the exponent into $x^n = \left(x^2\right)^{n/2}$. By calling the function recursively, there are going to be $\Theta(\log n)$ many multiplications since each exponentiation is directly proportional to the number of logarithmic operations on x^n.

Now suppose that n is odd. Then consider breaking up the exponent into $x^n = x\left(x^2\right)^{(n-1)/2}$. We note that there are $\log_2 n - 1$ many multiplications. Hence the algorithm (recursively) will run with $O(\log n)$ many multiplications.

In either case, we deduce that there are at most $O(\log n)$ many multiplications regardless of our choice of M and n.