Homework 2

Casey Bates

19 April 2020

1 Multiplication and Division

(a)

$$O(log^2(N)): \qquad t = k \times n^2$$

It takes 3 nanoseconds to multiply 1000 bit numbers:

$$3 = k \times n^2$$

To find the time to multiply 5000 bit numbers, we must multiply n by a factor of 5:

$$k \times (5 \times n)^2 = 25 \times k \times n^2 \tag{1}$$

$$3 \times 25 = 75 \tag{2}$$

Therefore, it will take 75 nanoseconds to multiply 5000 bit numbers.

(b)

$$O(\sqrt{N}): \qquad t = k \times 2^{\frac{n}{2}}$$

It takes 11 nanoseconds to factor N using trial division:

$$11 = k \times 2^{\frac{n}{2}}$$

We know that N' is 10 bits larger than N:

$$k \times 2^{\frac{10+n}{2}} = k \times 2^5 \times 2^{\frac{n}{2}} = 32 \times k \times 2^{\frac{n}{2}} \tag{1}$$

$$11 \times 32 = 352 \tag{2}$$

Therefore, it will take 352 nanoseconds to factor N' using trial division

2 RSA

We know that the repeated squares algorithm used in RSA is $O(log^3(N))$

$$O(log^3(N)): \qquad t = k \times n^3$$

We must multiply our input by 4 to switch from 1024-bit RSA to 4096-bit RSA:

$$k \times (4 \times n)^3 = 64 \times k \times n^3 \tag{1}$$

$$64 \times k \times n^3 = 64 \times t \tag{2}$$

Therefore, 4096-bit RSA takes 64 times longer than 1024-bit RSA.

3 Summations

(a)
We know that a single addition will have a running time of $O(log_2(N))$
In the equation $(((1+2)+3)...+N)$ there will be $N-1$ additions.
Thus, the running time of this program will be:

$$O(log_2(N) \times (N-1)) = O(Nlog_2(N) - log_2(N))$$

Since $log_2(N)$ is insignificant compared to $Nlog_2(N)$, the running time can be simplified to $O(Nlog_2(N))$

(b)

Running time for addition: $O(log(N))$
Running time for multiplication and division: $O(log^2(N))$
So the total running time of $\frac{N(N-1)}{2}$ is:

$$O(log(N)) + O(log^2(N)) + O(log^2(N)) = O(log(N) + 2log^2(N)) \tag{1}$$

$$O(log(N) + 2log^2(N)) = O(log^2(N)) \tag{2}$$

Therefore, the running time is $O(log^2(N))$

4 6^N

We know that $N \times N$ will take $O(log^2(N))$, so:

$$6 \times 6 = 6^2 : \qquad O(log^2(6)) \tag{1}$$

$$6^2 \times 6 = 6^3 : \qquad O(log^2(6) \times log(6)) = O(log^3(6)) \tag{2}$$

$$\vdots$$

$$6^{N-1} \times 6 = 6^N : \qquad O(log^{N-1}(6) \times log(6)) = O(log^N(6)) \tag{3}$$

$$O(log^N(6)) = O(Nlog(6)) \tag{4}$$

$$O(Nlog(6)) = O(N) \tag{5}$$

Since a program to compute 6^N will have $N-1$ multiplications, the simplified runtime of 6^N will be:

$$(N-1) \times O(N) = O(N^2)$$

Therefore, the running time of 6^N will be $O(N^2)$.

5 X^N

We know that $X \times X$ will take $O(log^2(X))$ so:

$$X \times X = X^2 : \qquad O(log^2(X)) \tag{1}$$

$$X^2 \times X = X^3 : \qquad O(log^2(X) \times log(X)) = O(log^3(X)) \tag{2}$$

$$\vdots$$

$$X^{N-1} \times X = X^N : \qquad O(log^{N-1}(X) \times log(X)) = O(log^N(X)) \tag{3}$$

$$O(log^N(X)) = O(Nlog(X)) \tag{4}$$

Since a program to compute X^N will have $N-1$ multiplications, the simplified running time of X^N will be:

$$(N-1) \times O(Nlog(X)) = O(N^2log(X))$$

Therefore, the running time of X^N will be $O(N^2log(X))$.

6 Fibonacci

Since we know that $F_n = \alpha^n$:

$$\prod_{i=1}^{n-1} F_i = \prod_{i=1}^{n-1} \alpha^i = \alpha^1 \times \alpha^2 \times ... \times \alpha^{n-1} = \alpha^{\frac{n(n-1)}{2}}$$

We also know that $N_1 \times N_2$ will take $O(log(N_1) \times log(N_2))$. Thus:

$$\alpha^{\frac{n(n-1)}{2}} \times \alpha^n : \qquad O(log(\alpha^{\frac{n(n-1)}{2}}) \times log(\alpha^n)) \tag{1}$$

$$O(\frac{n(n-1)}{2} log(\alpha) \times nlog(\alpha)) = O(n^3 log^2(\alpha)) \tag{2}$$

$$O(n^3 log^2(\alpha)) = O(n^3) \tag{3}$$

The equation $(((F_1 \times F_2) \times F_3)... \times F_n)$ will have $n-1$ multiplications, so the running time for the given Fibonacci algorithm is:

$$(n-1) \times O(n^3) = O(n^4)$$

Therefore, the running time of this Fibonacci algorithm will be $O(n^4)$.