**Quick Assignment: Master Theorem**

**Shubham Swami - A02315672**

**Master Theorem:**

T(n) = aT(n/b) + f(n)

n = size of input

a = number of subproblems in the recursion.

n/b = size of each subproblem.

f(n) = cnk = cost of the work done outside the recursive call.

Case 1: a< bk  => T(n) ~ nk

Case 2: a= bk => T(n) ~ nk logbn

Case 3: a> bk => T(n) ~ n ^ logba

**Time Complexity Analysis**:

**1: Binary Search:**

Here a = 1, b = 2, k = 0 => f(n) = O(1)

T(n) = T(n/2) + O(1)    
bk = 1 => a = bk => Case 2

T(n) = nk logbn => n0 log2n

T(n) = log 2 n

T(n) = Θ(log n)

**2: Merge Sort:**

a = 2, b = 2, k=1 => f(n) = O(n)

T(n) = 2T(n/2) + O(n)

bk = 2 => a = bk

T(n) = nk logbn => n log 2 n  
T(n) = n log 2 n

T(n) = Θ(nlog n)

**3: D&C Polynomial Multiplication:**

From the lecture we simplified

**PQ(x) = P L Q L x0 + (P H Q L + P L Q H ) xn/2 + P H Q H xn**

i.e. 4 subproblems of size n/2 and the D&C polynomial multiply algorithm requires a linear scan through the arrays of size n each recursive call.

a = 4 b = 2, k=1 => f(n) = O(n)

T(n) = 4T(n/2) + O(n) (a = 4, b = 2, k = 1)

bk = 2 => a > bk  => Case 3

T(n)= n ^ logba

T(n) = n^ (log24) and (log24 = 2)

T(n) = n2

T(n) = Θ(n2)

**Summary:**

|  |  |  |
| --- | --- | --- |
| Binary Search | Merge Sort | D&C Polynomial Multiplication |
| a = 1 | a=2 | a=4 |
| b = 2 | b=2 | b=2 |
| k = 0 | k=1 | k=1 |
| T(n) = Θ(log n) | T(n) = Θ(n log n) | T(n) = Θ(n2) |