Time and Space Complexity 2

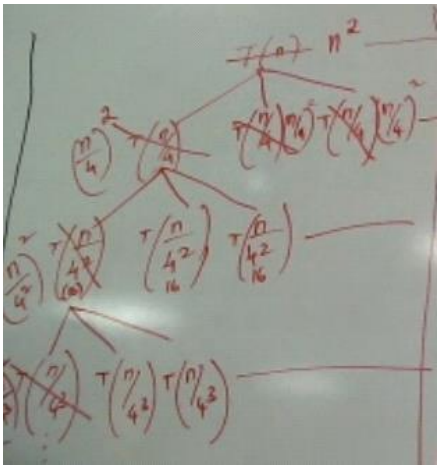
Sunday, February 7, 2021 7:26 AM

$$T(n) = 3 * T\left(\frac{n}{4}\right) + n^2$$

$$T\left(\frac{n}{4}\right) = 3 * T\left(\frac{n}{16}\right) + (\frac{n}{4})^2$$

$$\frac{n}{4} \rightarrow \frac{n}{4^2} \rightarrow \frac{n}{4^3} \rightarrow \frac{n}{4^4}$$

Height	Nodes	Value of level
0	1	1*n^2 = n^2
1	3	3*(n/4)^2
2	9	
3	27	
h	3^(h-1)	3^(h-1)*(n/4^h-1)^2
h-1	3^h	3^(h)*T(1)



$$T(n) = 3^0 * \left(\frac{n}{4^0}\right)^2 + 3^1 * \left(\frac{n}{4^1}\right)^2 + 3^2 * \left(\frac{n}{4^2}\right)^2 + \cdots + 3^{h-1} * \left(\frac{n}{4^{h-1}}\right)^2 + 3^h * T(1)$$

$$T(n) = \sum_{i=0}^{h-1} 3^i * \left(\frac{n}{4^i}\right)^2 + 3^h * T$$

$$T(n) = n^2 \sum_{i=0}^{h-1} \left(\frac{3}{4}\right)^i + 3^h * T(1)$$

Decreasing geometric series equivalent to 1

$T(n) = n^2 + 3^h * T(1)$

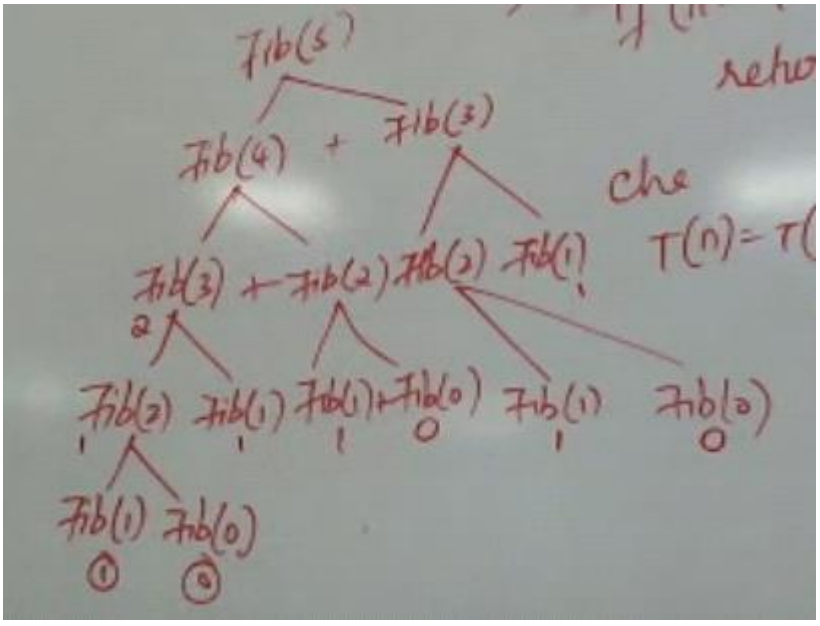
$\frac{n}{4^h} = 1$

$h = \log(4)\, n$

$T(n) = n^2 + 3^{\log(4)n} * 1$
$T(n) = n^2 + n^{\log(4)3}$
$T(n) = n^2 + n^{0.7}$
$T(n) = n^2$

All leaves of a recursive tree will not stop at one the same level

(fibonacci series)
If n==1 or 0
    Return n
Else
    T(n) =T(n-1) + T(n-2)



Complex (unequal) problem since level is not the same

T(n) =T(n-1) + T(n-2) + 1 (1 is for addition)

5 levels: $2^5 \rightarrow (1.61)^2$
Recursion can reuse values if using different stack space

Fibonacci is exponential
Divide and conquer uses recursive problems where subproblems are solved repeatedly (increasing time complexity) and the space taken is also greater

Time complexity
1. Recursive
2. Non-recursive

Space complexity
1. In-place: no additional space
2. Out-place: uses additional space

Quick sort is fastest sorting algorithm because it uses less space

Algorithms with swaps(in) takes greater time than shifting(out)

For large size data, divide data and use insertion, then quicksort

Merge Sort
Divide and conquer
Recursive