4.1(c) Average case

The height for average case is $\log(\frac{n}{k})$ so the complexity of this variant merge sort is $f(k) = n\log(\frac{n}{k}) + (\frac{n}{k})k^2$

So is the graph as k increases for the average case, the time increases for the values of n.

worst case

The worst case for insertion sort is $O(n^2)$ but merge sort is $O(n\log n)$. So as observed on the graph as k increased for the worst case, the time increases significantly for the value of n because as k increases the insertion sort sort's more number of elements on the worst case array having a time complexity of n^2. Hence $n^2 + n\log(n)$ the sorting time increases as k increases.

Best case

In best case different value of K requires less time. the time complexity for best case for insertion sort is $\theta(n)$ and when K becomes large the insertion sort is applied so the complexity is $\theta(n)$.

(d) observing (b) and (c) if the array is sorted then it's best if K has maximum value. If the array is random or unsorted the it is best to have K near 1. There is a case until K reaches (1) a value such that the average and worst case of i is decreasing $K = 1$ $K! = n$ generates the n most time

Name - Aayansh Rathi

4.2 (a) $T(n) = 36\,T\left(\frac{n}{6}\right) + 2n$

$a = 36, \quad b = 6, \quad f(n) = 2n$

$= n^{\log_6 a}$

$= n^{\log_6 36}$

$= n^2$

$h(n) = \frac{f(n)}{n^{\log_6 36}} = \frac{2n}{n^2} = \frac{2}{n} = 2n^{-1}$

$n^x < 0$

So case 2: $f(n) = O(n^{2-\varepsilon})$ for $\varepsilon = 1$

$= \Theta(n^2)$

(b) $T(n) = 5\,T\left(\frac{n}{3}\right) + 17n^{1.2}$

$a = 5, \quad b = 3, \quad f(n) = 17n^{1.2}$

$= n^{\log_3 5}$

$= n^{1.46}$

$h(n) = \frac{17n^{1.2}}{n^{1.46}} = 17n^{-0.2}$

So, $\varepsilon = 1$ case 1.

$T(n) = \Theta(n^{1.46})$

Page No.
Date : / /

(c) $T(n) = 12\,T\left(\frac{n}{2}\right) + n^2 \log n$

$a = 12, \; b = 2 \quad f(n) = n^2 \log n$

$n^{\log_2 12} = n^{3.58}$

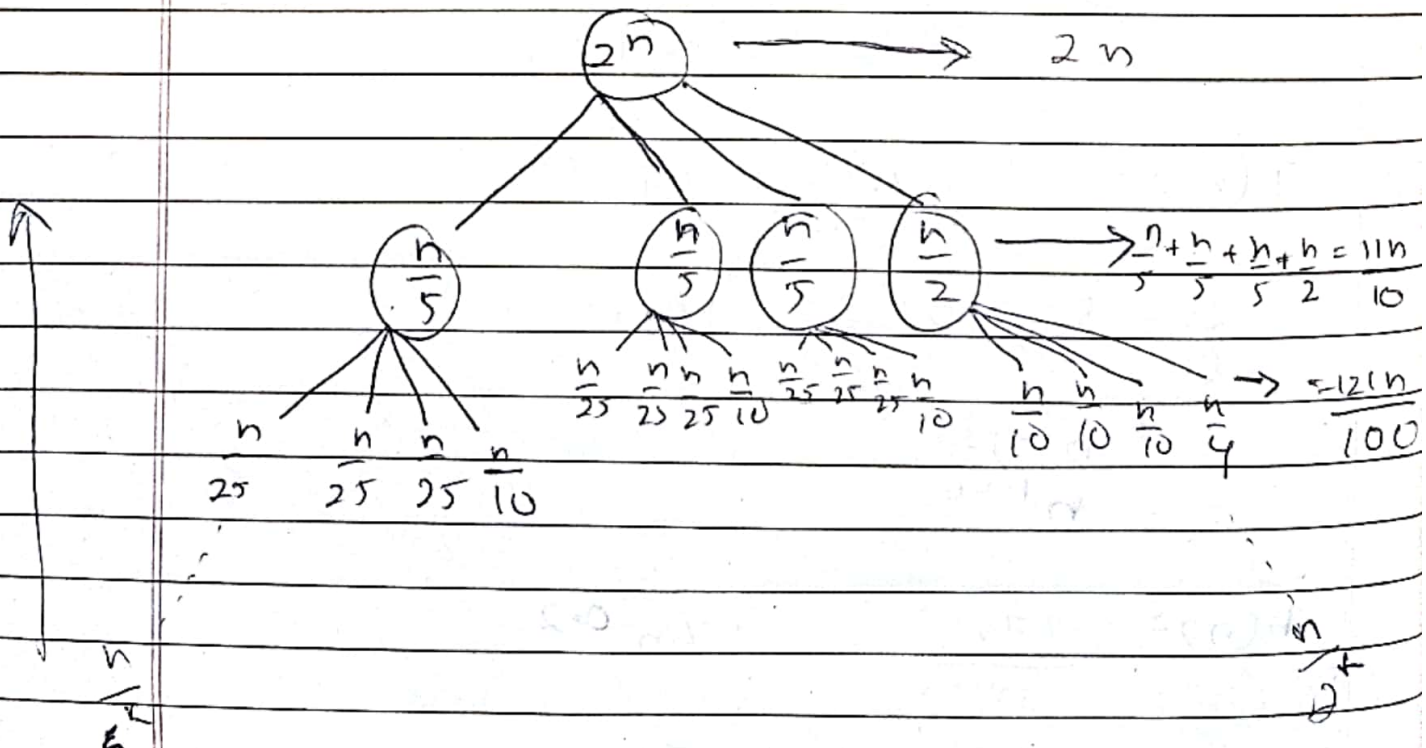
$h(n) = \frac{n^2 \log n}{n^{3.58}} = n^{-1.58} \log n$

So $\varepsilon =$ [redacted] 1

Case 1:

$T(n) = \Theta(n^{3.58})$

(d) $T(n) = 3T\left(\frac{n}{5}\right) + T\left(\frac{n}{2}\right) + 2n$



let $\frac{n}{5^k} = 1$

$n = 5^k$

$\log n = k \log 5$

$\boxed{k = \log_5 n}$ → for left hand height

let $\frac{n}{2^k} = 1$

$\boxed{k = \log_2 n}$ → for right hand height

$n * k$

$n * \log_2 n$

$\boxed{T(n) = n \log_2 n}$ → time complexity

(d) As $T(n)$ is an increasing and tends to ∞, $T(n)$ grows faster than $T\left(\frac{n}{5}\right)$. So $T\left(\frac{n}{2}\right)$ is much more significant than $T\left(\frac{n}{5}\right)$

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

$a = 4,\ b = 2,\ f(n) = 2^n$

$$= n^{\log_2 4}$$
$$= n^2$$

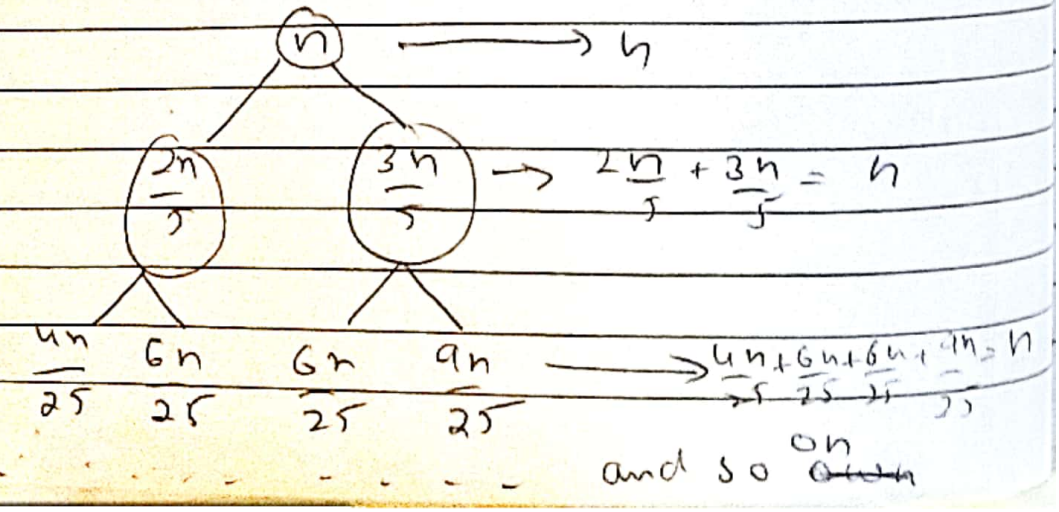
$$h(n) = \frac{f(n)}{n^{\log_2 4}} = \frac{2^n}{n^2}$$

as $2^n > n^2$

This is case 3 as $f(n)$ is polynomially larger than 2^n

$$T(n) = \Theta(f(n)) = \Theta(2^n).$$

(e) $$T(n) = T\left(\frac{2n}{5}\right) + T\left(\frac{3n}{5}\right) + \Theta(n)$$



So left most path height will be

$$\frac{n}{\left(\frac{5}{2}\right)^k}$$

Right most path height will be

$$\frac{n}{\left(\frac{5}{3}\right)^k}$$

left path	right path
let $\frac{n}{\left(\frac{5}{2}\right)^k} = 1$	let $\frac{n}{\left(\frac{5}{3}\right)^k} = 1$
$n = \left(\frac{5}{2}\right)^k$	$\log n = k \log\left(\frac{5}{3}\right)$
$\log n = k \log\left(\frac{5}{2}\right)$	$k = \log_{\left(\frac{5}{3}\right)} n$
$k = \log_{\left(\frac{5}{2}\right)} n$	

So time complexity will be

$n \propto$ left height
$n \propto \log_{\left(\frac{5}{2}\right)} n$

$n \propto$ right path
$n \propto \log_{\left(\frac{5}{3}\right)} n$
$= n \log_{\left(\frac{5}{3}\right)} n$

$n \propto$ left path
$n \propto \log_{\left(\frac{5}{2}\right)} n$

The two paths differ by a constant factor.

So

$$T(n) = \Theta(n \log n)$$