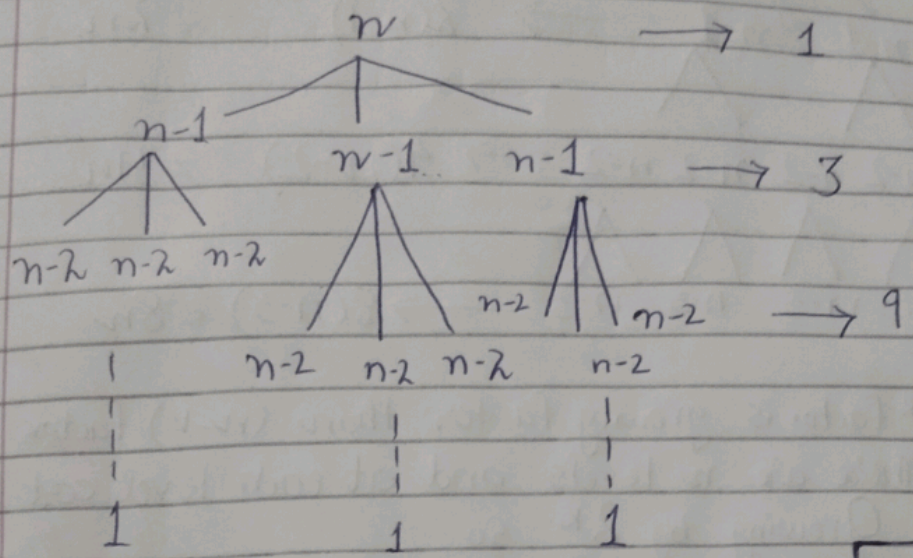
Hira Khalid
232-2594
DS-5C

Assignment no 1

1- $T(n) = 3T(n-1)+1$



Total Cost: $1+3+3^2+3^3+\dots+3^n$

As series is geometric $= \frac{1(3^n-1)}{3-1}$

$\Rightarrow \boxed{O(3^n)}$

As $r>1$

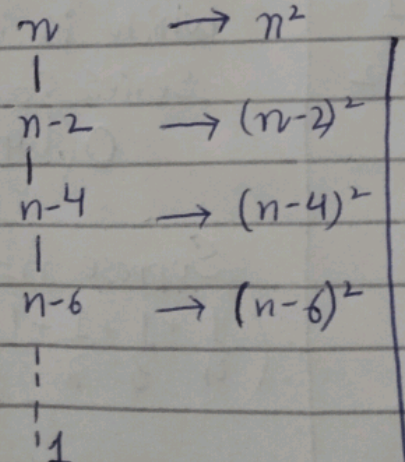
$= \frac{a(r^n-1)}{r-1}$

where

r = common ratio

a = First term

(b) $T(n) = T(n-2)+O(n^2)$



Cost for each term: $(n-2k)^2$

pattern is

$(n-2(0))^2+(n-2(1))^2+(n-2(2))^2+(n-2(3))^2+\dots$

$O(n^3)$

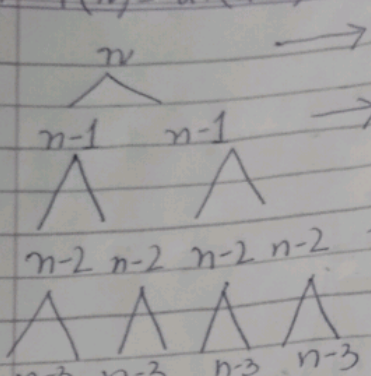
Reference Formula

$1^2+2^2+3^2+4^2+\dots+n$

$n(n+1)(2n+1)/6$

$T(n)=n3$

(c) $T(n) = 2T(n-1) + O(n)$



$= n$

$\approx 2n$

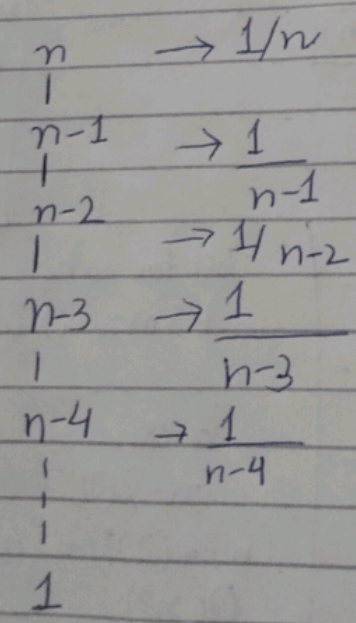
$\approx 4n$

$\approx 8n$

2^k factor is growing faster than $(n-k)$ factor
There are n levels and at each level cost is growing by 2^k so

$O(n \log 2^n)$

(d) $T(n) = T(n-1) + O(1/n)$



Pattern is

$\left(\frac{1}{n} + \frac{1}{n-1} + \frac{1}{n-2} + \cdots\right)$

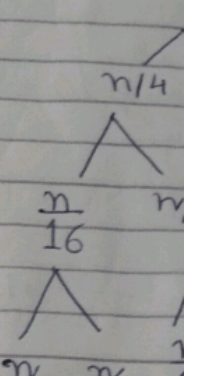
which is harmonic series so

$O(\log n)$

Suppose $n = 4$

$\left(\frac{1}{4} + \frac{1}{3} + \frac{1}{2} + 1\right) \approx 2.08$

(e) $T(n) = 2T($



As series

$2^0 * lo$

+ -

where

if we

2

So

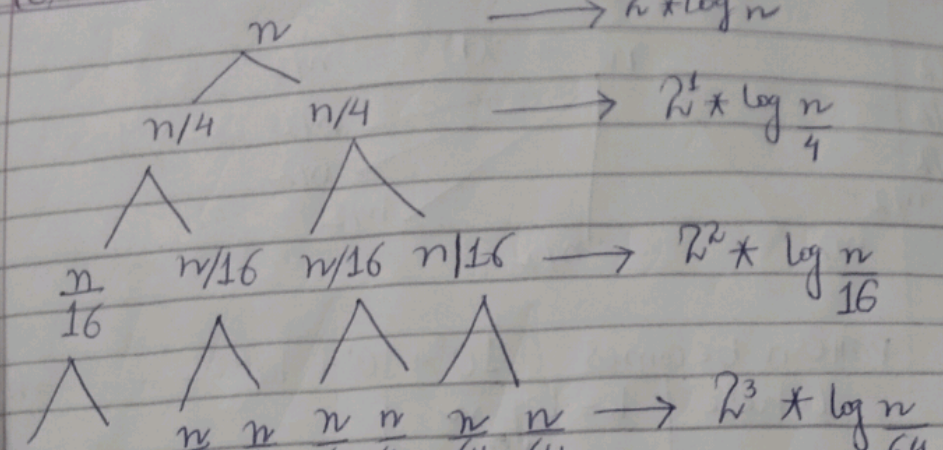
2^0

$\Rightarrow \log$

$1n$

don

(e) $T(n) = 2T(n/4) + O(\log n)$



As series is

$$2^0 * \log n + 2^1 * \log \frac{n}{4} + 2^2 * \log \frac{n}{16} + 2^3 * \log \frac{n}{64}$$

$$+ \text{---} + 2^k * \log \frac{n}{4^k}$$

where $k = \log_4 n$

if we see

$2 * \log \frac{n}{4} \Rightarrow 2 * \log n - \log 4$ where $\log 4$ is constant which will be ignored

So small will be apply for all terms

$2^0 * \log n + 2^1 * \log n + 2^2 * \log n + \text{---} + 2^{\log_4 n} \log n$

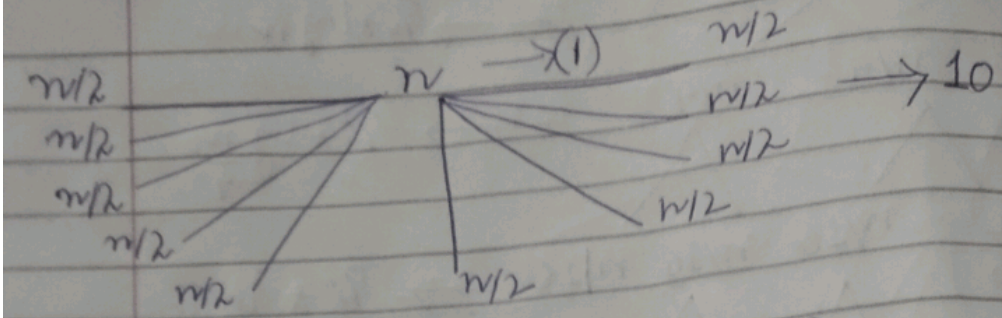
$\Rightarrow \log n (2^0 + 2^1 + 2^2 + 2^3 + \text{---} + 2^{\log_4 n})$

In geometric series, the last term is always dominating so

$2^{\log_4 n} \Rightarrow 2^{\frac{\log_2 n}{\log_2 4}} \Rightarrow 2^{\log_2 \sqrt{n}} \Rightarrow \sqrt{n}$

Time Complexity $= \sqrt{n} * \log n$

(f) $T(n) = 10\,T(\frac{n}{2}) + \theta(1)$



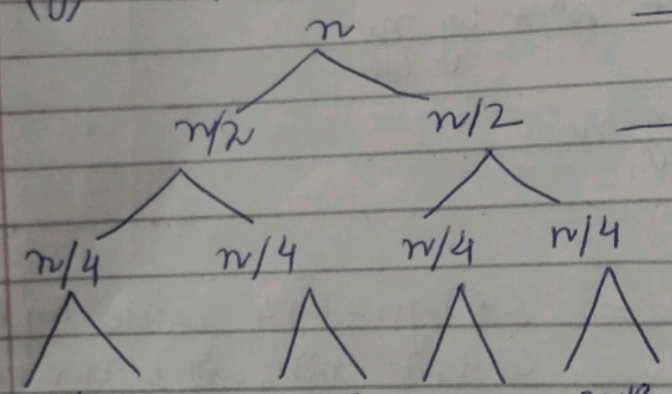
Pattern becomes $10^0 + 10^1 + 10^2 + \dots + 10^k$

where $\boxed{k = \log_2 n}$

In geometric series the last term is always dominating term so

$10^{\log_2 n} \Rightarrow O(n) = n^{\log_2 10}$

(g) $T(n) = 2\,T(n/2) + O(n \log n)$



$\to 2(n \log n)$

$\to 2(\frac{n}{2} \log \frac{n}{2})$

$\to 4(\frac{n}{4} \log \frac{n}{4})$

$\to 8(\frac{n}{8} \log \frac{n}{8})$

pattern becomes:

$n \log n + n \log \frac{n}{2} + n \log \frac{n}{4} + n \log \frac{n}{8} + n \log 1$

$n[\log n + (\log n - 1) + (\log n - 2) + \dots + 1]$

Put $\log n = A$ so

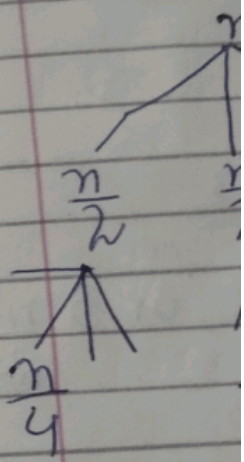
$n[A + (A-1) + (A-2) + \dots + 1]$ which is reverse written terms of arithmetic series

So $= \frac{A(A+1)}{2}$

$= n[\log_2 n$

$O(n) =$

(h) $T(n)$



So at

lev

Patter

(n

((

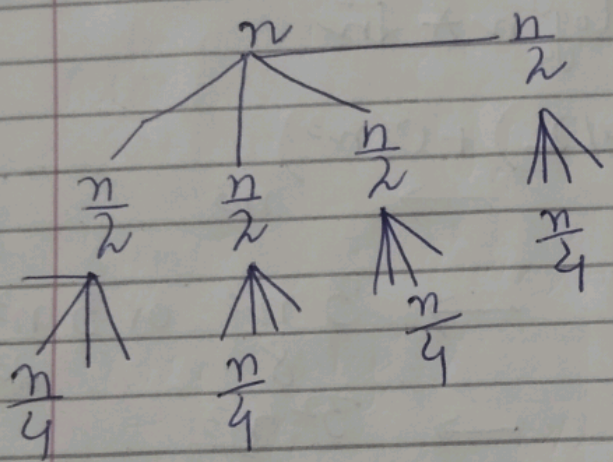
- - -

So $= \frac{A(A+1)}{2}$

$= n\left[\frac{\log_2 n(\log_2 n+1)}{2}\right] \Rightarrow n\left[\frac{(\log_2 n)^2 + \log_2 n}{2}\right]$

$O(n) = O\left[n(\log_2 n)^2\right]$

(h) $T(n) = 4T(n/2) + O(n^2\sqrt{n})$



For 1st level: $n^2\sqrt{n}$
Cost

~~For 2nd level~~

For level 1: $4\left(\frac{n^2}{16}\sqrt{\frac{n}{2}}\right)$

$\frac{n^2}{4}\frac{\sqrt{n}}{\sqrt{2}}$

For level 2: $16\left(\frac{n^2}{16}\sqrt{\frac{n}{4}}\right)$

$\frac{n^2\sqrt{n}}{2}$

For level 3: $64\left(\frac{n^2}{64}\sqrt{\frac{n}{8}}\right)$

$= \frac{n^2\sqrt{n}}{2\sqrt{2}}$

So at each level the cost is

level $i = \frac{n^2\sqrt{n}}{(\sqrt{2})^i}$

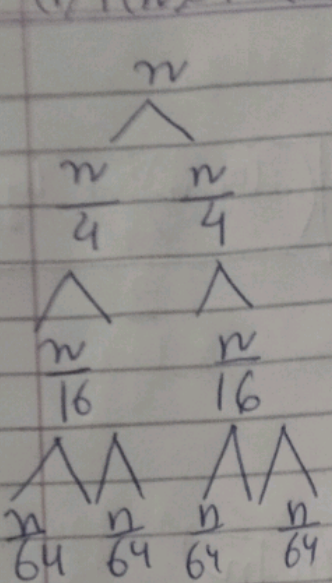
Pattern becomes

$\left(\frac{n^2\sqrt{n}}{(\sqrt{2})^0} + \frac{n^2\sqrt{n}}{(\sqrt{2})^1} + \frac{n^2\sqrt{n}}{(\sqrt{2})^2} + \cdots + \frac{n^2\sqrt{n}}{(\sqrt{2})^{\log_2 n}}\right)$ → (geometric series and we will select last dominating term)

As denominator factor is constant

So $O(n) = O(n^2\sqrt{n})$

(i) $T(n) = 2T(n/4) + O(\sqrt{n})$



$\rightarrow \sqrt{n}$

$\rightarrow 2\sqrt{\frac{n}{4}} = \sqrt{n}$

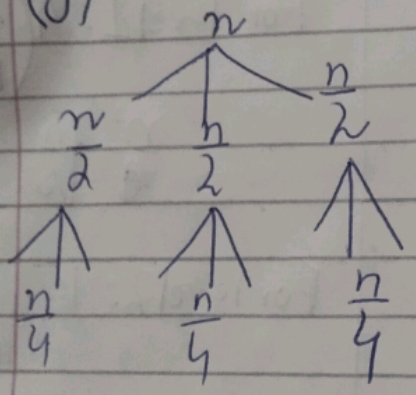
$\rightarrow 4\sqrt{\frac{n}{16}} = \sqrt{n}$

$\rightarrow 8\sqrt{\frac{n}{64}} = \sqrt{n}$

Total cost = Height of tree * each level cost

$= \log_4 n * \sqrt{n}$

(j) $T(n) = 3T(n/2) + O(n^3)$



$\rightarrow n^3$

$\rightarrow 3\frac{n^3}{8}$ or $3\frac{n^3}{2^3}$

$\rightarrow 3^2\frac{n^3}{64}$

So pattern becomes

$$\left(\frac{3}{8}\right)^0 n^3 + \left(\frac{3}{8}\right)^1 n^3 + \left(\frac{3}{8}\right)^2 n^3 + \cdots$$

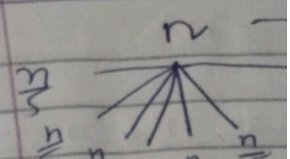
As $\gamma < \frac{1}{2} \Rightarrow \frac{3}{8} = 0.375$ As series is geometric series

So $\frac{1}{1-\gamma} = \frac{1}{1-0.375} = 1.6$

$1.6\, n^3$ Constant will be ignored

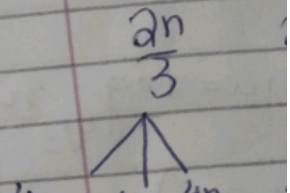
$O(n^3)$

(k) $T(n) = T$



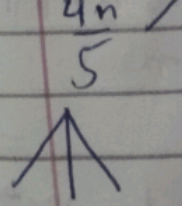
$T^0 + T^1 +$

(l) $T(n) =$

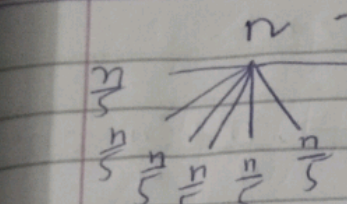


As seri

(m)



(k) $T(n) = 7T(n/5) + O(1)$



$n \longrightarrow 7^0$

$\longrightarrow 7^1$

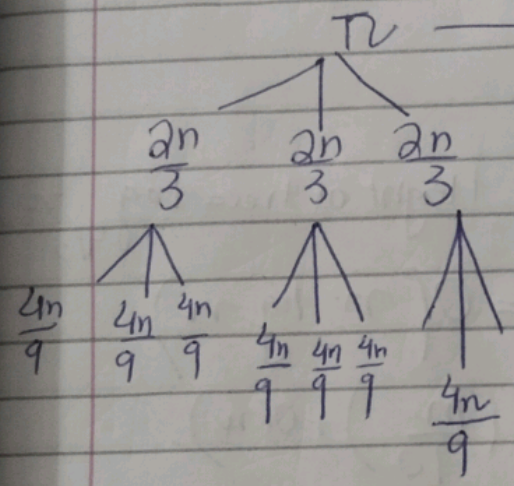
$7^0 + 7^1 + 7^2 + \cdots + 7^K$

where $K = \log_5 n$

So $7^{\log_5 n} \approx n^{\log_5 7}$

$$O(n) = n^{\log_5 7}$$

(l) $T(n) = 3T\left(\frac{2n}{3}\right) + O(1)$



$n \longrightarrow 1$

$\longrightarrow 3$

$\longrightarrow 9$

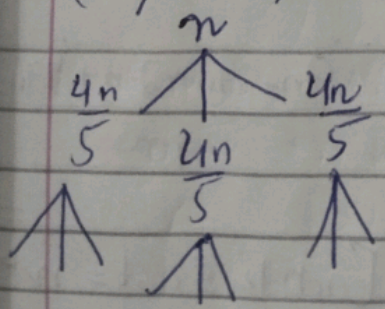
$3^0 + 3^1 + 3^2 + 3^3 + \cdots + 3^K$

where $K = \log_{3/2} n$

As series is geometric so

$$O(n) = O\left(n^{\log_{3/2} 3}\right)$$

(m) $T(n) = 3T\left(\frac{4n}{5}\right) + O(1)$



$\longrightarrow 1$

$\longrightarrow 3$

$\longrightarrow 9$

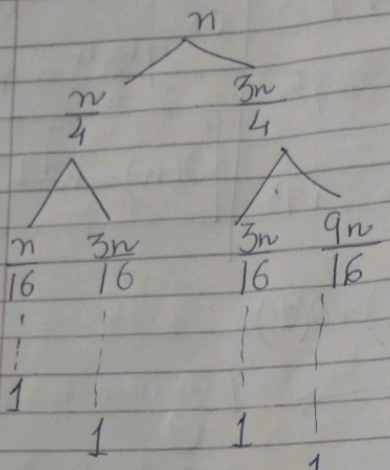
Series become (which is geometric)

$3^0 + 3^1 + 3^2 + 3^3 + \cdots + 3^K$

where $K = \log_{\frac{5}{4}} n$

So $O(n) = O\left(n^{\log_{\frac{5}{4}} 3}\right)$

(n) $T(n) = T(n/4) + T(3n/4) + O(n^2)$



$\rightarrow n^2$

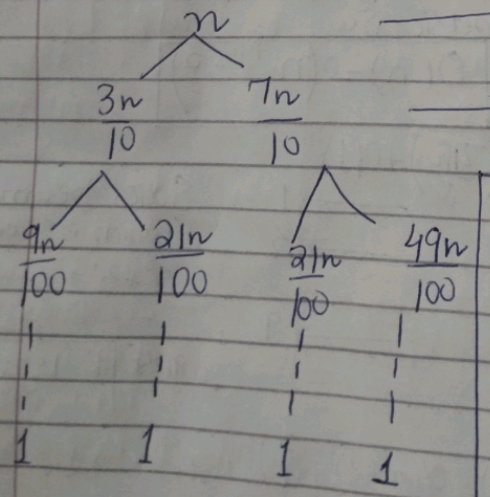
$\rightarrow \frac{n^2}{4} + \frac{3n^2}{4} = n^2$

$\rightarrow \frac{n^2 + 3n^2 + 3n^2 + 9n^2}{16} = n^2$

Each level cost = n^2 Height of tree = $\log_{4/3} n$

So total cost or $O(n) = O(n^2 \log n)$

(o) $T(n) = T\left(\frac{3n}{10}\right) + T\left(\frac{7n}{10}\right) + O(n)$



$\rightarrow n$

$\rightarrow \frac{3n+7n}{10} = n$

$\rightarrow \frac{9n+21n+21n+49n}{100} = n$

Each level cost = n

Height of tree = $\log_{10/7} n$

So $O(n) = O(n \log n)$

Q-2

Derive Recurrence Equation

Part a Initially left=1 Right=n

1st Recursive call

Split Data (arr, left, right-1)

Size = right-1 -left +1

= n-1 -1+1

Size = n-1

2nd Recursive call

Split Data(arr, left+1, right)

Size = right - left -1 + 1

= n-1-1+1

Size = n-1

In viewData function:

→ Outer loop

while (i<right)

where i=left= 1

right = n

So O(n)

→ inner loop

while (j<right)

where j=j*2 always

So $O(\log_2 n)$

Overall $O(n \log_2 n)$

Recurrence Eq is

$T(n) = 2T(n-1) + O(n \log_2 n)$

Part b

In this temp function there are 16 recursive calls and the size in them, is decreased by n/2

Also, Solve (A[], B[], N) In this function

→ outer loop
for (i=1 to N)

→ inner loop
for (j=1 to i)

So n times loop will run

It will also run for n times

Overall this function will take $O(n^2)$

Recurrence equation will be
$T(n) = 16T(n/2) + O(n^2)$

Part C

→ First recursive call mystery (2n/3)

→ 2nd recursive call mystery (n/5)

for (i = 1 to n)
$O(n)$

Recurrence eq will be
$T(n) = 2T(n/3) + T(n/5) +$

Part e

$k = \frac{2*(right - left + 1)}{7} = \frac{2*(n-1+1)}{7} =$

→ 1st recursive call size = $k = \frac{2n}{7}$

→ 2nd recursive call remaining part $n - \frac{2n}{7} = \frac{5n}{7}$

In update f
left=1
loops
for (i=left,
for (j=lef
Series/ patte
1, 2, 4,
which is
Recurren
T(n) =
Part d
→ 1st
→ 2
→

In update function

left=1 right=n

loops

for(i=left; i<=right; i*=2)

for(j=left; j<=i; j++)

Series/pattern becomes

1, 2, 4, 8, ---n

which is geometric series

As $r > \frac{1}{2}$ so

$a(r^n-1)/r-1$

where $a=1$ $r=2$ so

$\frac{1(2^{\log_2 n}-1)}{2-1}$

Recurrence eq is

$T(n) = T(2n/3) + T(5n/3) + O(n)$

$\Rightarrow 2^{\log_2 n}$

$\Rightarrow n^{\log_2 2}$

$\Rightarrow O(n)$

Part d

$$third = \frac{(right - left + 1)}{3}$$

$$third = \frac{n}{3}$$

→ 1st recursive call Stooge-Sort(arr, left, right-third)

left = 1 right - third = $n - \frac{n}{3} = \frac{2n}{3}$ (1st 2/3rd part sorting)

→ 2nd recursive call Stooge-Sort(arr, left+third, right)

left + third = $\frac{n}{3}$, right = n

last 2/3rd portion of array will be sort here

→ 3rd recursive call

Again first $\frac{2}{3}^{rd}$ portion will be sort here