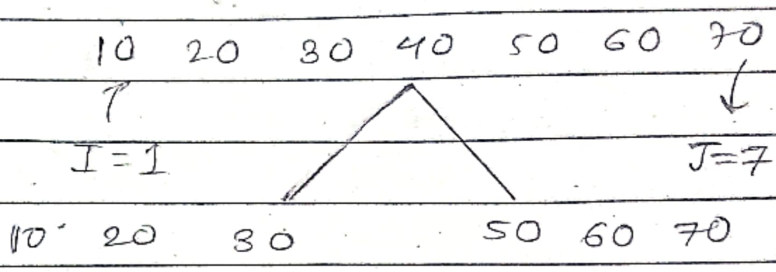
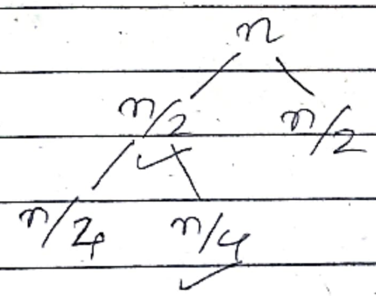
Recurrence relⁿ

BS(a, i, j, x) → 30



~~mf~~

```
mid=(i+j)/2
if(a[mid]==x)
    return mid
else
    if (a[mid]>x)
        BS(a, i, mid-1, x)
    else
        BS(a, mid+1, j, x)
```



$n \rightarrow n/2 \rightarrow (n/2/2) \rightarrow$

$\therefore T(n) = T(n/2) + C$

E-g.
```
int power (int x, int n) {
    if(n==0)
        return 1
    else
        return x* power(x, n-1)
}
```

$T(0)$ = Time to solve problem of size 0
– Base case

$T(n)$ = Time to solve problem of size n
– Recursive case

$T(0) = C_1$

$T(n) = \underbrace{C_2}_{\text{Time for } x} + \underbrace{T(n-1)}_{\text{Time for power}(x, n-1)}$

Substitution method

① $T(n) = \begin{cases} T(n/2) + c & \text{if } n > 1 \\ 1 & \text{if } n = 1 \end{cases}$

$T(n) = T(n/2) + c \quad -①$

$T(n/2) = T(n/4) + c \quad -②$

$T(n/4) = T(n/8) + c \quad -③$

Sub. ② in ①

$T(n) = T(n/4) + c + c$

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

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

$\vdots$

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

To terminate this func^n

if $n = 2^k \quad \therefore k = \log_2 n$

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

$= T(1) + kc$

$\boxed{T(n) = O(\log_2 n)}$

Substitution method solves all recurrence rel^n problems

(2) $T(n) = \begin{cases} 1 & \text{if } n = 1 \\ n * T(n-1) & \text{if } n > 1 \end{cases}$

$T(n) = n * T(n-1)$ — (1)

$T(n-1) = (n-1) * T(n-2)$ — (2)

$T(n-2) = (n-2) * T(n-3)$ — (3)

Sub. (2) in (1)

$T(n-2) = (n-2) * T(n-3)$

$T(n) = n * (n-1) * T(n-2)$

$= n * (n-1) * (n-2) * T(n-3)$

$(n-1)$ steps

$n * (n-1) * (n-2) * (n-3) \cdots T(n-(n-1))$

$= n * (n-1) * (n-2) * (n-3) \cdots * 3 * 2 * 1$

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

$T(n) = O(n^n)$

(2) ~~Ex~~ $T(n) = \begin{cases} 1 & n = 1 \\ 2T\left(\frac{n}{2}\right) + n & \text{otherwise} \end{cases}$

$T(n) = 2T(n/2) + n \quad -(1)$

$T(n/2) = 2T\left(\frac{n}{4}\right) + \frac{n}{2} \quad -(2)$

$T(n/4) = 2T\left(\frac{n}{8}\right) + \frac{n}{4} \quad -(3)$

$T(n) = 2\left[2T\left(n/4\right) + \frac{n}{2}\right] + n$

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

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

$= 4\left[2T\left(\frac{n}{8}\right) + \frac{n}{4}\right] + 2n$

$= 2^3 T\left(\frac{n}{8}\right) + 3n$

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

$= \vdots$

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

For termination

$$n = 2^k$$
$$k = \log_2 n$$

$$T(n) = 2^k(T(1)) + kn$$

$$T(n) = n\cdot(1) + n\cdot\log n$$

$$T(n) = \cancel{n+1}\; n + n\log n$$
ignore

$$\boxed{T(n) = n\cdot\log(n)}$$

(3) $T(n) = \begin{cases} 1 & n = 1 \\ T(n-1) + \log n & n > 1 \end{cases}$

$$T(n) = T(n-1) + \log n \quad \text{-(1)}$$

$$T(n-1) = T(n-2) + \log(n-1) \quad \text{-(2)}$$

$$T(n-2) = T(n-3) + \log(n-2) \quad \text{-(3)}$$

~~T(n) =~~ Sub. (2) in (1)

$$T(n) = T(n-2) + \log(n-1) + \log(n)$$
$$= T(n-3) + \log(n-2) + \log(n-1) + \log(n)$$
$$\vdots \; k \text{ times}$$
$$= T(n-k) + \log(n-(k-1)) + \log(n-(k-2)) + \dots + \log n$$

$n - k = 0$

$\therefore \cancel{k = n - 1}$

$\therefore k = n$

$\therefore T(0) + \log 1 + \log 2 + \log 3 + \cdots + \log n$

$= 1 + \log(1 \cdot 2 \cdot 3 \cdots n)$

$= 1 + \log(n!)$

$= 1 + \log(n^n)$

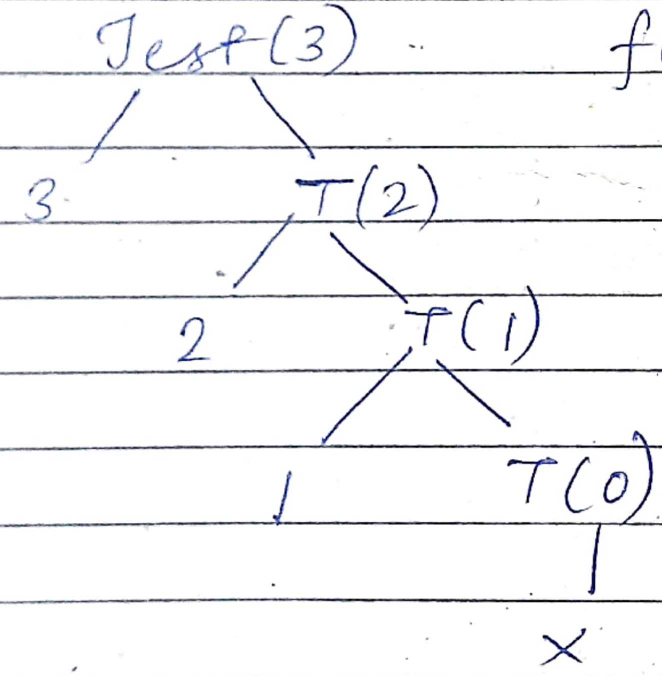
$= 1 + n \log n$

$= O(n \log n)$

E.g.

```
void Test(int n){        → T(n)
    if(n>0){
        pf("%d", n);     — 1
        Test(n-1);       — T(n-1)
    }
}
```

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



$f(n) = n+1$ calls

$O(n)$

$$T(n) = \begin{cases} 1 & n=0 \\ T(n-1)+1 & n>0 \end{cases}$$

Recurrence relⁿ for dec. functions

```
void Test(int n) {                      — T(n)
        if (n>0)                        — 1
        {
            for (i=0 ; i<n; i++)        → n+1
            {
              pf("%d", n);              — n
            }
            Test(n-1);                  — T(n-1)
        }
}
```

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

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

$$T(n) = \begin{cases} 1 & n=0 \\ T(n-1)+n & n>0 \end{cases}$$

$T(n)$ — (n)

n $\quad T(n-1)$ — $(n-1)$

$n-1$ $\quad T(n-2)$ — $(n-2)$

$n-2$ $\quad T(n-3)$ ⋯ $(n-3)$

⋮

$T(2)$ $\quad 2$

2 $\quad T(1)$ $\quad 1$

1 $\quad T(0)$ $\quad 0$

$\times$

$$0+1+2+\cdots+n-1+n = \frac{n(n+1)}{2}$$

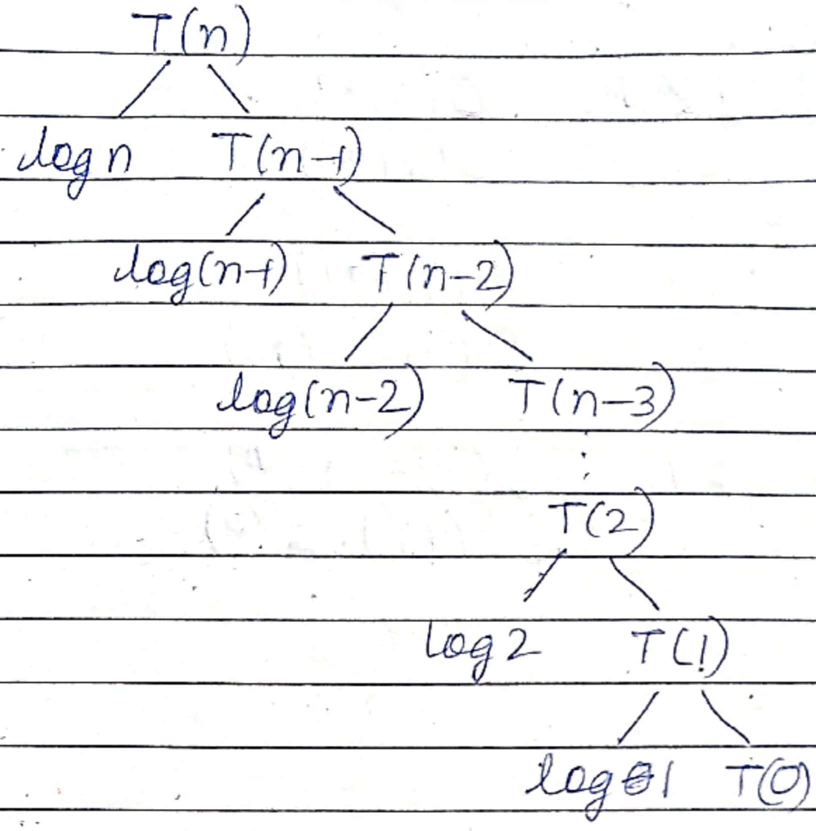
$$T(n) = \frac{n(n+1)}{2}$$

$$T(n) = O(n^2)$$

E.g.

```
void Test(int n){          - T(n)
    if(n>0){
        for(i=1, i<n, i=i*2){
            pf("%d", i);     <- log n  log n
        }
        Test(n-1);           T(n-1)
    }
}
```

$$T(n) = \begin{cases} 1 & n=0 \\ T(n-1)+\log n & n>0 \end{cases}$$



$\log n + \log(n-1) + \cdots + \log 2 + \log 1$

$\log(n \times (n-1) \times \cdots \times 2 \times 1)$

$\log(n!) \rightarrow$ Upper bound $\log n^n$

$\therefore T(n) = O(n \log n)$

Master Theorem for dec. functions:

$$\boxed{T(n) = aT(n-b) + f(n)}$$

$\boxed{a > 0 \quad b > 0}$ & $f(n) = O(n^k)$ where $k \geq 0$

Cases:

1) If $a < 1$ $\quad O(n^k)$

$O(f(n))$

2) if $a = 1$ $\quad O(n^{k+1})$

$O(n * f(n))$

3) if $a > 1$ $\quad O(n^k a^{n/b})$

$O(f(n) \cdot a^{n/b})$

Recurrence relⁿ for dividing funcⁿ

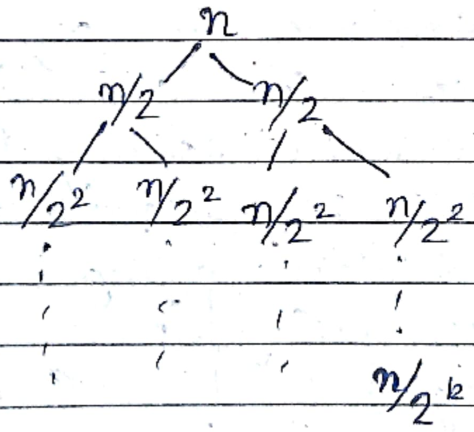
```
void Test (int n){
    if(n>1) {
        for(i=0 ; i<n ; i++){
n  <------> {
            stmt
        }
T(n/2) -- Test (n/2);
T(n/2) -- Test (n/2);

    }

}
```

$$\therefore T(n) = 2T(n/2) + n$$

$$T(n) = \begin{cases} 1 & n=1 \\ 2T(n/2) + n & n>1 \end{cases}$$



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

$n = 2^k \quad k = \log_2 n$

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

Master's thm. for dividing func^n s:

$$\boxed{T(n) = a\,T\left(\frac{n}{b}\right) + f(n)}$$

$a \geq 1 \quad b > 1$

$$\boxed{f(n) = \Theta(n^k \log^p n)}$$

Case 1: if $\log_b a > k$ then $\boxed{\Theta(n^{\log_b a})}$

Case 2: if $\log_b a = k$

- if $p > -1$ $\quad \Theta(n^k \log^{p+1} n)$
- if $p == -1$ $\quad \Theta(n^k \log(\log n))$
- if $p < -1$ $\quad \Theta(n^k)$

Case 3: if $\log_b a < k$

- if $p \geq 0$ $\quad \Theta(n^k \log^p n)$
- if $p < 0$ $\quad \Theta(n^k)$

E.g ① $T(n) = 2T(n/2) + 1$

$a = 2, b = 2$ $\quad$ $\log_2 2 = 1 > k (0)$

$f(n) = \theta(1)$ $\quad$ Case 1:

$= \theta(n^0 \log^0 n)$ $\quad$ $\theta(n^1)$

$k = 0, p = 0$ $\quad$ $\log_b a = \log_2 2 = 1$

② $T(n) = 4T(n/2) + n$

$\log_2 4 = 2 > k(1)$ $\quad$ $p = 0$

$\therefore T(n) = \theta(n^2)$

③ $T(n) = 2T(n/2) + \dfrac{n}{\log n}$

$\log_2 2 = 1$, $k = 1$, $p = -1$

Case 2: $p = -1$

$\theta(n \log(\log n))$

<u>Root functions:</u>

Fn $T(n) = \begin{cases} 1 & n = 2 \\ T(\sqrt{n}) + 1 & n > 2 \end{cases}$

```
void Test(int n){
    if (n>2){
        stmt
        Test(√n);
    }
}
```

$T(n) = T(\sqrt{n}) + 1$

$f(n) = \Theta(n^k \log^p n) = \Theta(1)$

$= \Theta(n^0 \log^0 n) = \Theta(1)$

$k = 0 \quad p = 0$

$a = 1 \quad b = 1$

$\log_1 1 = 1 > k(0)$

We need to change the variable

Let $m = \log_2 n$

$\therefore T(n) = T(2^{m/2}) + 1$

Let new func^n $S(m) = T(2^m)$

$\therefore$ Substituting this into above eq^n

$S(m) = S(m/2) + 1$

$a = 1 \quad b = 2 \quad f(m) = 1$

$\log_b a = \log_2 1 = 0$

$\therefore f(m) = O(m \wedge 0) = O(1)$

$\because \log_b a = 0$

$\therefore$ ~~$f(m) = \Theta(m^1$~~

$f(m) = \Theta(m^0 \log^{0+1} m) = \Theta(\log m)$

$\therefore S(m) = \Theta(\log m)$

$\therefore T(n) = S(m) = \Theta(\log(\log n))$

Master thm.:

Let $T(n)$ be defined on non-negative integers by the recurrence

$$T(n) = a\,T\left(\frac{n}{b}\right) + f(n)$$

where $a \geq 1$ & $b > 1$ be a constant & $f(n)$ be a funcn & n/b can be interpreted as $\left\lfloor \frac{n}{b} \right\rfloor$ or $\left\lceil \frac{n}{b} \right\rceil$.

$f(n)$ should be of the form n^k, K is constant

Then $T(n)$ can be bound asymptotically as follows:

① If $f(n) = O(n^{\log_b a - \varepsilon})$ for some constant $\varepsilon > 0$, $f(n)$ is polynomially slower than $n^{\log_b a}$

Then

$$\boxed{T(n) = \Theta(n^{\log_b a})}$$

② If $f(n) = \Theta(n^{\log_b a})$, then $f(n)$ & $n^{\log_b a}$ grow at same rate

$$\boxed{T(n) = \Theta(n^{\log_b a} \cdot \log n)}$$

(3) If $f(n) = \Omega(n^{\log_b a + \varepsilon})$ for some constant $\varepsilon > 0$, & if $af(n/b) \leq cf(n)$ for some constant $c < 1$ & all sufficiently large n, then ~~$T(n) = \theta$~~ $f(n)$ is polynomially faster $n^{\log_b a}$

$$\boxed{T(n) = \theta(f(n))}$$

eg (1) $T(n) = 2T(n/4) + \sqrt{n}$

Comparing.

$a = 2 \quad b = 4 \quad f(n) = \sqrt{n}$

$n^{\log_b a} = n^{\log_4 2} = n^{1/2} = \sqrt{n}$

& $f(n) = \sqrt{n} = n^{\log_b a}$

Hence, case (2)

$T(n) = \theta(n^{\log_b a} \cdot \log n)$

$T(n) = \theta(\sqrt{n} \log n)$

(2) $T(n) = 2T(\sqrt{n}) + \log n$

Let $m = \log_2 n$

$n = 2^m$

Eqn becomes

$T(2^m) = 2T(2^{m/2}) + \log 2^m$

$T(2^m) = 2T(2^{m/2}) + m \log_2 2$

$T(2^m) = 2T(2^{m/2}) + m \quad$ —(1)

let $S(m) = T(2^m)$
Eq^n (1) becomes

$$S(m) = 2S(m/2) + m$$

comparing with $T(n) = aT(n/b) + f(n)$

$a = 2$ $\quad$ $b = 2$ $\quad$ ~~$f(m) = m$ $f(n) = m$~~

$f(n) = m$

$$m^{\log_b a} = m^{\log_2 2} = m$$

$f(n) = m = m^{\log_b a}$
$\therefore$ Case 2

$$S(m) = \Theta(m^{\log_b a} . \log m)$$
$$= \Theta(m \log m)$$

$$T(n) = T(2^m) = S(m) = \Theta(\log n . \log(\log n))$$

Recursion Tree Method:

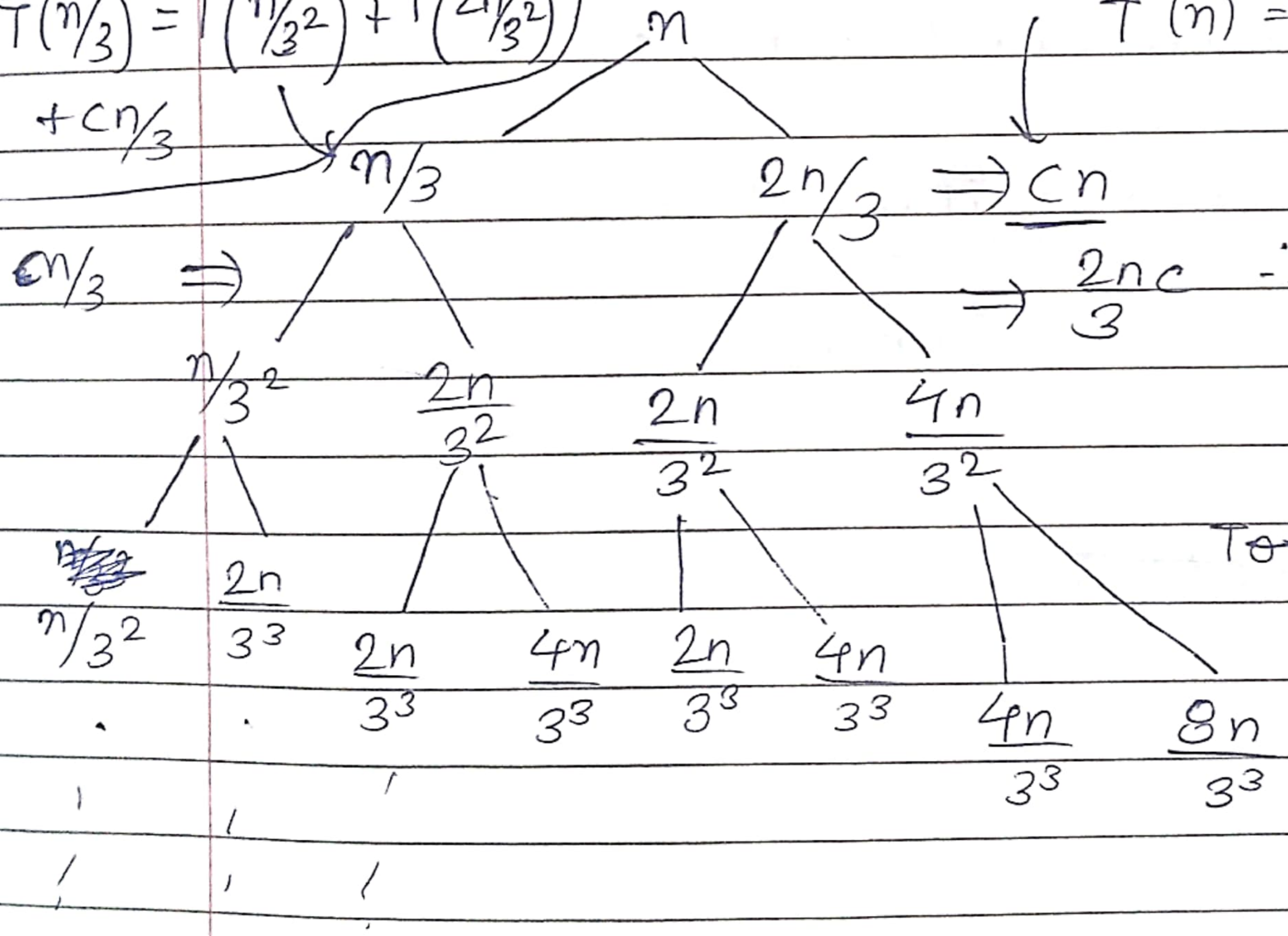
(1) $T(n) = T(n/3) + T(2n/3) + cn$

$T(n/3) = T(n/3^2) + T(2n/3^2) + cn/3$

cost in cn :;

$T(n) = T(n/3) + T(2n/3) + cn$

cost



∴ cost =

$\frac{cn}{3} + \frac{c2n}{3} = \cancel{\frac{n}{3}}\ cn$

~~Total cos~~

$\text{cost} = \left(\frac{n}{3^2} + \frac{2n}{3^2} + \frac{2n}{3^2} + \frac{4n}{3^2}\right)c$

$= \frac{9n}{\cancel{3^2}} = nc$

PAGE NO.:

Consider the max. height for the height of tree

~~For left node the~~

The leaf node of left subtree would be $\cancel{\frac{n}{3^d}}$ $\frac{n}{3^d}$

& that of right subtree would be

$$\cancel{\left(\frac{2}{3}\right)^d n = \frac{n}{(3/2)^d}} \quad \left(\frac{2}{3}\right)^d n = \frac{n}{(3/2)^d}$$

$$\therefore \frac{n}{(3/2)^d} = 1$$

$$n = \left(\frac{3}{2}\right)^d$$

$$\boxed{d = \log_{3/2} n}$$

Cost for each level was cn

$\therefore$ Time complexity $= \Theta(cn \log_{3/2} n)$

$= \Theta(n \log_{3/2} n)$