FRANKLIN'S LECTURES

ALGORITHM ANALYSIS & DESIGN

Module 1 Part 2

CST306

RECURSIVE EQUATIONS/ RECURRENCE RELATIONS

- A recurrence expresses the running time of a recursive algorithm as a function of smaller inputs.
- The structure is typically:

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

- A recurrence is a formula that tells you:
- How much time an algorithm takes based on the time taken by smaller versions of the same problem

- Example
- Binary search splits the input into half each time:

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

- Merge sort divides the array into 2 parts and merges:

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

Recursion fnⁿ time complexity

- → Substitution method.
- → Iteration method.
- → Recursive tree method.
- → Master's theorem

1. Substitution method.

Q_1. $$T(n) = \begin{cases} T(n/2) + c & \text{if } n > 1 \\ 1 & \text{if } n = 1 \end{cases}$$

$\rightarrow$ termination condition

s/n

$T(n) = \boxed{T(n/2)} + c$ —— ①

$T(n/2) = T(n/4) + c$ —— ②

$T(n/4) = T(n/8) + c$ —— ③

⋮ ⋮ ⋮

2 in 1

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

$$= T(n/4) + 2c \quad \text{--- } 4$$

3 in 4

$$T(n) = T(n/8) + c + 2c$$

$$= T(n/8) + 3c$$

$$T(n) = T(n/2) + c \Rightarrow T(n/4) + 2c \Rightarrow T(n/8) + 3c$$

$$\Rightarrow T(n/16) + 4c \Rightarrow \text{— — – . . .}$$

$$\Rightarrow T(n/2^1) + 1c \Rightarrow T(n/2^2) + 2c \Rightarrow T(n/2^3) + 3c \ldots$$

- - - - - — — - - . . — — - -

k terms

$$\Rightarrow \underline{\underline{T(n/2^k)}} + kc$$

if $\underline{\underline{n = 2^k}}$ $\Rightarrow$ $T(n/n) + kc$

$$= T(1) + kc$$

$$\Rightarrow 1 + kc$$

$$2^k = n$$

$$k = \log_2 n$$

$$\Rightarrow 1 + \log_2 n \cdot c$$

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

$$b^x = y$$

$$x = \log_b y$$

2. Qus

$$T(n) = \begin{cases} 1 & \text{if } n = 1 \text{ — base condn.} \\ n * T(n-1) & \text{if } n > 1 \end{cases}$$

s/n.

$$T(n) = n * T(n-1) \quad \text{— (1)}$$

$$T(n-1) = (n-1) * T(n-2) \quad \text{— (2)}$$

$$T(n-2) = (n-2) * T(n-3) \quad \text{— (3)}$$

(2) in (1)

$$T(n) = n * (n-1) * T(n-2) \quad \text{— (4)}$$

(3) in (4)

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

$$= n * (n-1) * (n-2) * (n-3) * T(n-4)$$

$$= n * (n-1) * (n-2) * (n-3) * (n-4) * T(n-5)$$

. . . . $(n-1)$ steps.

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

$$\cdots T(n-(n-1))$$

$$T(\cancel{n}-\cancel{n}+1) \Rightarrow T(1).$$

$$\Rightarrow \underbrace{n * (n-1) * (n-2) * (n-3) * \cdots\cdots 3 * 2 * 1}_{n!}.$$

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

$$= \underline{\underline{O(n!)}}$$

ITERATION METHOD

1. $$T(n) = \begin{cases} 2 & \text{if } n = 0 \\ 2 + T(n-1) & n > 0 \end{cases}$$

$$T(n) = \underbrace{2 + }\, T(n-1)$$

$$2 + [2 + T(n-2)]$$

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

$$= 4 + T(n-2)$$

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

$$T(n-2) = 2 + T(n-1-2)$$
$$\Rightarrow = 2 + T(n-3)$$

$\vdots$

1. $$\cancel{T(n) = \begin{cases} 2 & ; n = 0 \\ 2 + T(n-1) & ; n > 0 \end{cases}}$$

$$\Rightarrow 4 + T(n-2)$$

$$\Rightarrow 4 + 2 + T(n-3)$$

$$= 6 + T(n-3)$$

$$\vdots$$

$$\Rightarrow 2K + T(n-k)$$

2×1: $2 + T(n-1)$

2×2: $4 + T(n-2)$

2×3: $6 + T(n-3)$

2×4: $8 + T(n-4)$

2×5: $10 + T(n-5)$

$\vdots$

k times

if $n = k$

$$= 2n + T(\cancel{n} - \cancel{k})$$

$$= 2n + T(0)$$

$$T(n) = 2n + \underline{2}$$

$$\underline{\underline{T(n) = O(n)}}$$

2. $T(n) = n + 2T(n/2)$

$$T(n) = n + 2T(\underbrace{n/2})$$

$$T(n/2) = n/2 + 2T(n/4)$$

$$T(n/4) = n/4 + 2T(n/8)$$

$$\vdots$$

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

$$= n + \cancel{2}\cdot\frac{n}{\cancel{2}} + 4T(n/4)$$

$$= n + n + 4T(\underbrace{n/4})$$

$$= n + n + 4\left[n/4 + 2T(n/8) \right]$$

$$= n + n + \cancel{4} \cdot \frac{n}{\cancel{4}} + 8T(n/8)$$

$$= \underbrace{n + n + n}_{} + 8T(n/8)$$

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

⋮

k times.

$$= kn + 2^k \underbrace{T(n/2^k)}_{}$$

$$n = 2^k$$

$$k = \log_2 n$$

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

const

$$= \underline{kn} + n \underbrace{T(1)}_{}$$

$$= n \cdot \log_2 n + \text{const.}$$

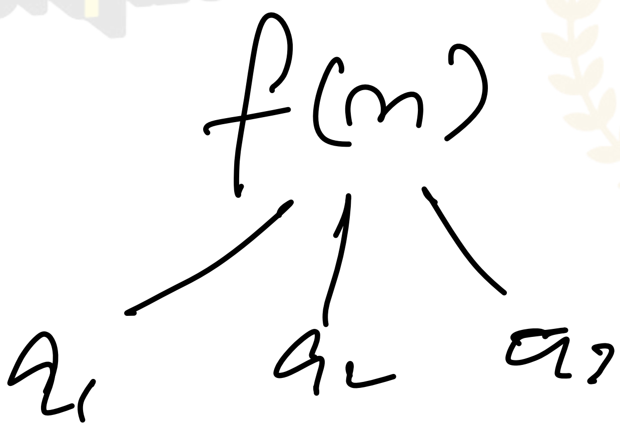
$$\boxed{T(n) = O(n \log n)}$$

RECURSION TREE METHOD

$$T(n) = \underline{2T(\quad) + f(n)}$$

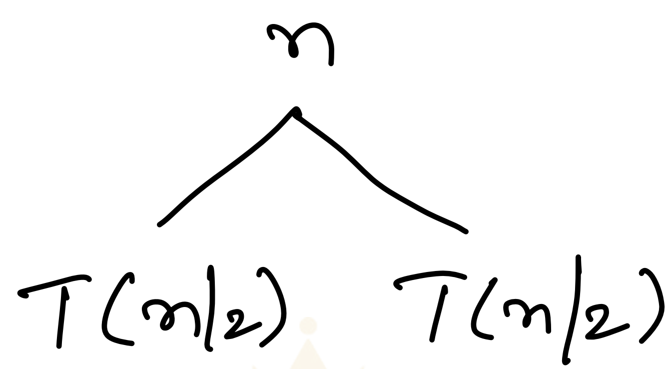
$$= a\,T(n/b) + f(n)$$

$a = 3$



$a = 2$

1. $T(n) = 2T(n/2) + n$



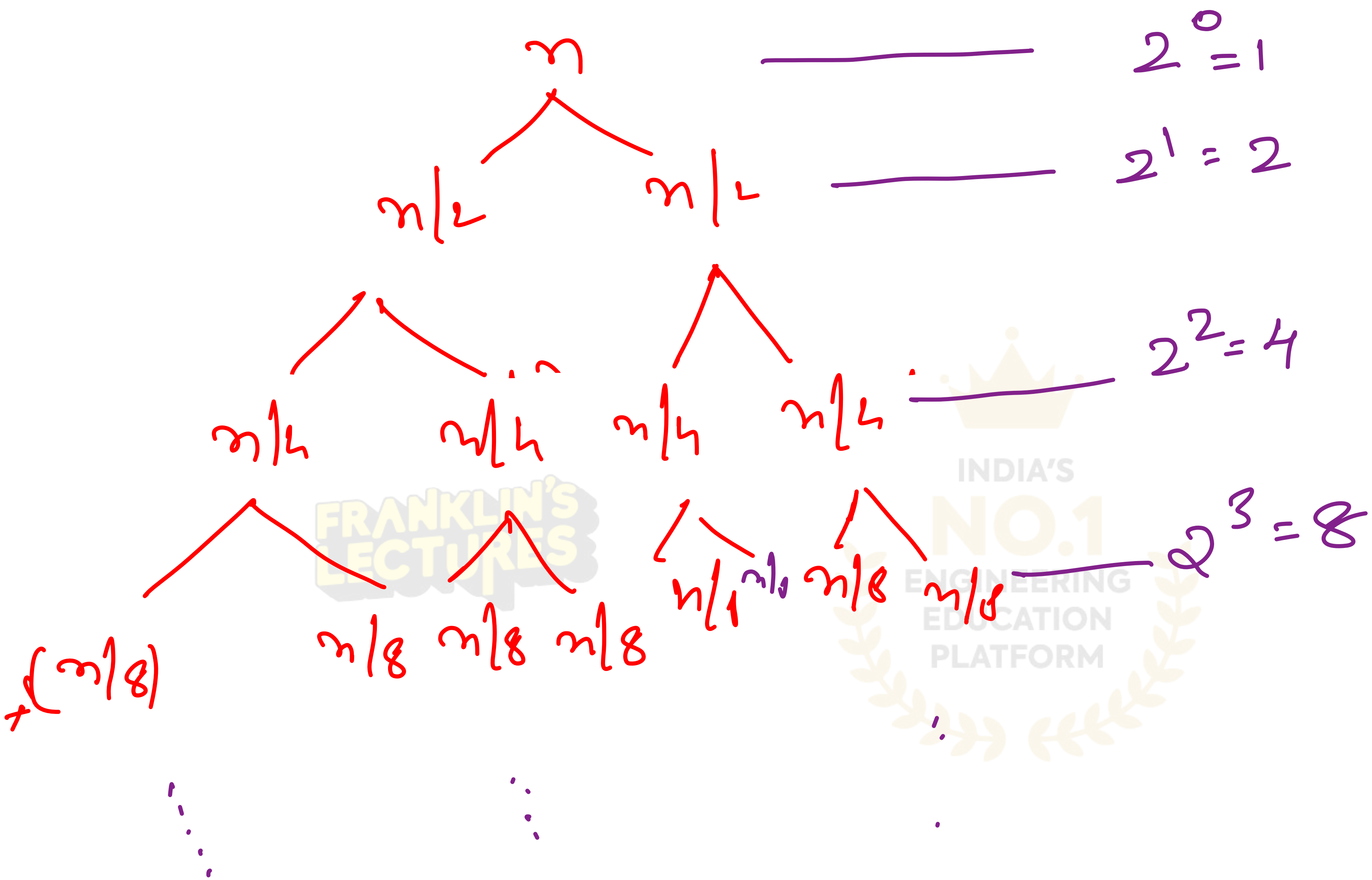
Put $n = n/2$

$T(n/2) = 2 \cdot T(n/4) + n/2$

Put $n = n/4$

$T(n/4) = 2 \cdot T(n/8) + n/4$

$T(n/8) \ldots$



$1 \cdot n = n$

$2 \cdot n/2 = n$

$4 \cdot n/4 = n$

$8 \cdot n/8 = n$

$\underline{\underline{n(k+1)}}$

$T(n/2) \rightarrow T(n/4) \rightarrow T(n/8)$ . . . . —— k^{th} term

$T(n/2^1) \rightarrow T(n/2^2) \rightarrow T(n/2^3)$ . . . . . $T(n/2^k)$

$$T\left(\frac{n}{2^k}\right) = T(1)$$

$n/2^k = 1$

$n = 2^k$

$k = \log_2 n$

total cost

$= n(k+1)$

$= n(\log_2 n + 1)$

$= n \log n + n$

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

② $T(n) = 3T(n/4) + n$

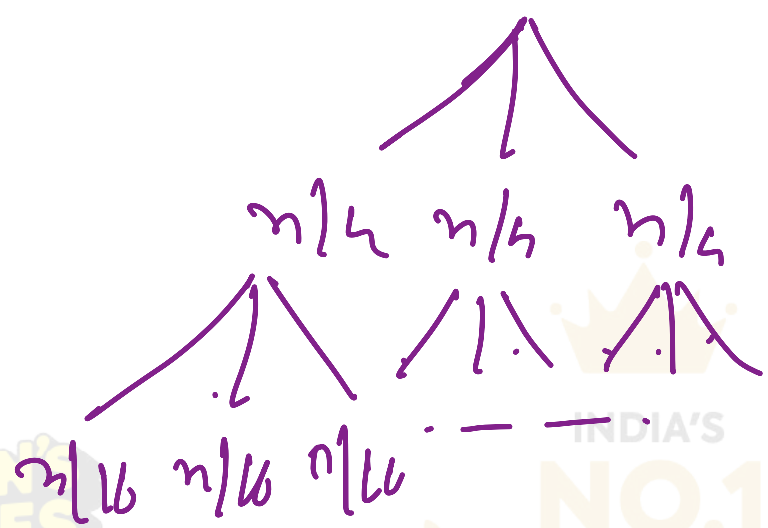
$$n$$

$$T(n/4) \quad T(n/4) \quad T(n/4)$$

$n = n/4$

$T(n/4) = 3T(n/16) + n/4$

$T(n/16) \ldots$



n — n

— $3 \cdot n/4$

— $9 \cdot n/16$

$$\text{total cost} = n + 3 \cdot n/4 + 9 \cdot n/16 + \cdots$$

$$= n\left[\underbrace{1 + 3/4 + (3/4)^2 + (3/4)^3 + \cdots}_{4}\right]$$

$$n\,[\;4\;]$$

$$GP : \frac{a}{1-r} \longrightarrow \quad a = 1, \quad r = 3/4$$

$$= \frac{1}{1 - 3/4} = \frac{1}{\frac{4-3}{4}} = \frac{1}{1/4} = \underline{\underline{4}}$$

$$T(n) \Rightarrow 4n$$

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

MASTERS METHOD

For recurrences of the form

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

where $a \geq 1$, $b > 1$, $f(n) = O(n^K \log^P n)$

- Case 1: $a > b^k$
  - $T(n) = \Theta(n^{\log_b a})$

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

Case 2: $a = b^K$

- If $p > -1$ → $\Theta(n^{\log_b a} \log^{p+1} n)$
- If $p = -1$ → $\Theta(n^{\log_b a} \log \log n)$
- If $p < -1$ → $\Theta(n^{\log_b a})$

$p > -1 \quad T(n) = \Theta(n^{\log_b a} \cdot \log^{p+1}(n))$

$p = -1 \quad T(n) = \Theta(n^{\log_b a} \log(\log n))$

$p < -1 \quad T(n) = \Theta(n^{\log_b a})$

- Case 3: $a < b^K$
  - If $p \geq 0 \longrightarrow \Theta(n^K \log^K n)$
  - If $p < 0 \longrightarrow O(n^K)$

$$p \geq 0 = \theta(n^k \log^p(n))$$

$$p < 0 = \theta(n^k)$$

① $T(n) = 7T(n/2) + n^2$

$f(n) = n^2 (\log^0 n)$

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

$a = 7$

$b = 2$

$k = 2$

$p = 0$

$a > b^k = 2^2$

$7 > 4$

if $a > b^k$

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

$$\boxed{T(n) = \Theta(n^{\log_2 7})}$$

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

$= 2T(n/2) + n^1 \log^{-1} n$

$a = 2$; $b = 2$; $k = 1$; $p = -1$

$a = b^k$ $\quad p \cancel{>} -1 \quad p \cancel{<} -1 \quad p = -1$

$2 = 2^1$ $\quad \boxed{p = -1}$

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

$$\boxed{T(n) = \theta(n^{\log_2 2} \log(\log n))}$$

THANK YOU