

Chapter - 4

Divide & Conquer $\begin{cases} \text{Divide} \\ \text{conquer} \\ \text{combine} \end{cases}$

Recurrences:

It is an equation or Inequality that describes a function in term of its value on smaller inputs.

→ may take many form.

$$\text{eg } T(n) = T(n/3) + T(n/3) + \Theta(n)$$

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

Solving recurrences (obtaining Θ or O bounds)

Substitution Method	Recursion-tree Method	Master Method
<ul style="list-style-type: none"> guess a bound MI to prove guess is correct 	<p>converts recurrences into tree whose nodes represent the work incurred at various levels of recursion.</p> <p>(Tech - bounding summation)</p>	$T(n) = aT(n/b) + f(n)$ <p>Sub Problem</p>

Recurrences that are Inequality.

$T(n) \leq 2T(n/2) + \Theta(n) \rightarrow$ Recurrences gives only upperbound on $T(n)$
 i.e. O -notation is req for solution.

$$T(n) \geq 2T(n/2) + \Theta(n) \rightarrow \text{lower bound } \Omega(n)$$

STRASSEN'S ALGORITHM

↳ It is a Recursive algorithm for multiplying $n \times n$ matrices.

If $A = (a_{ij})$ and $B = (b_{ij})$ ($n \times n$ matrices)

Then product $C = A \cdot B$.

$$C_{ij} = \sum_{k=1}^n a_{ik} b_{kj}$$

SQUARE-MATRIX-MULTIPLY - Matrix multiply $O(n^3)$

$$T(n) = \begin{cases} \Theta(1) & , \text{ if } n=1 \\ 7T(n/2) + \Theta(n^2) & , \text{ if } n>1 \end{cases}$$

$S_1 = B_{12} - B_{22}$	$P_1 = A_{11} \cdot S_1$	
$S_2 = A_{11} + A_{12}$	$P_2 = S_2 \cdot B_{22}$	$C_{11} = P_5 + P_4 - P_2$
$S_3 = A_{21} + A_{22}$	$P_3 = S_3 \cdot B_{11}$	$+ P_6$
$S_4 = B_{21} - B_{11}$	$P_4 = A_{22} \cdot S_4$	$C_{12} = P_1 + P_2$
$S_5 = B_{11} + A_{22}$	$P_5 = S_5 \cdot S_6$	$C_{21} = P_3 + P_4$
$S_6 = B_{11} + B_{22}$	$P_6 = S_7 \cdot S_8$	$C_{22} = P_5 + P_1 - P_3$
$S_7 = A_{12} + A_{22}$	$P_7 = S_9 \cdot S_{10}$	$- P_7$
$S_8 = A_{11} + A_{21}$		
$S_8 = B_{21} + B_{22}$		
$S_{10} = B_{11} + B_{12}$		

Substitution Method

$$T(n) = 2T(\lfloor n/2 \rfloor) + n \text{ given } T(n) = O(n \log n)$$

To prove, $T(n) \leq Cn \log n$ for constant $C > 0$

$$T(\lfloor n/2 \rfloor) \leq \lfloor n/2 \rfloor \log(\lfloor n/2 \rfloor)$$

$$T(n) \leq 2(\lfloor n/2 \rfloor \log(\lfloor n/2 \rfloor)) + n$$

$$\leq Cn \log \lfloor n/2 \rfloor + n$$

$$< Cn \log n - Cn \log 2 + n$$

$$= Cn \log n - Cn + n$$

$$\leq Cn \log n.$$

Master's Theorem

$$k \geq 0 \quad a \geq 1 \quad b > 1$$

$$T(n) = aT(n/b) + O(n^k \log^p n)$$

Case 1 Big O $a > b^k$

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

Case 2 Theta Θ $a = b^k$

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

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

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

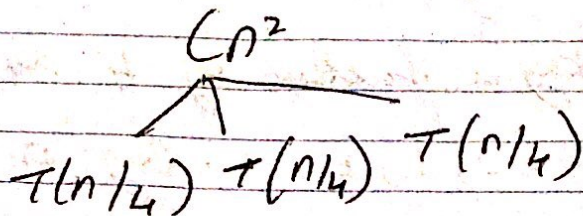
Case 3 Omega Ω $a < b^k$

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

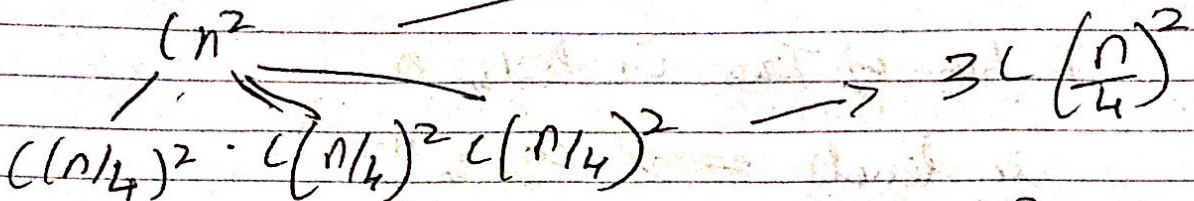
$$p < 0 \quad T(n) = n^k$$

Decision Tree

$$T(n) = 3T(n/4) + Cn^2 \rightarrow \text{Sub Problem size for a node at depth } i = n/4^i$$



$$\rightarrow n/4^i = 1; n = 4^i \quad i = \log_4 n.$$



$$\rightarrow 3C\left(\frac{n}{4}\right)^2$$

$$\rightarrow \left(\frac{3}{16}\right)^2 C\left(\frac{n}{4}\right)^2$$

$$O(n^{\log_4 3})$$

$$T(1) T(1) T(1) T(1) \dots$$

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

$$T(n/4) = 3T(n/16) + C(n/4)^2$$

$$T(n/16) = 3T(n/32) + C(n/16)^2$$

Each level has 3 times more nodes than above level so no of nodes in level i is 3^i .

Subproblem reduces by a factor of 4 as we go down for $i = 0, 1, 2, \dots, \log_4 n - 1$

Depth of Tree is $\log_4 n$
in levels $\Rightarrow \log_4^{n+1}$

Each level has cost $\rightarrow C \left(\frac{n}{4^i} \right)^2$

Multiplying total cost over all depth of the node i

$$i \in 3^i C \left(\frac{n}{4^i} \right)^2 = \left(\frac{3}{16} \right)^i C n^2$$

A depth $\log_4 n \Rightarrow$ has $3^{\log_4 n} = n \log_4^3$ nodes
 $\Rightarrow O(n \log_4^3)$