

CS & IT ENGINEERING

Algorithm

Miscellaneous Topics

Lecture No. - 10

By- Dr. Khaleel Khan
sir



Recap of Previous Lecture



Topic

All Pairs Shortest Paths

Topic

0/1 Knapsack

LCS

Topics to be Covered



Topic

L.C.S



Topic

Matrix Chain Product

$\langle MCP \rangle$ ✓

$S.O.S$ ✓

0/1KNAP ✓

Floyd-Warshall's




Topic : Dynamic Programming: (DP)

Longest Common Subsequence (LCS)

Given Two Strings x & y of length ' n ' & ' m ' charac's, a Subsequence that is common to both ' x ' & ' y ' is known as Common Subsequence;

$x = \langle A B C D \rangle$ $y = \langle B D C \rangle$

$\langle A \rangle \times$ $\langle B \rangle \checkmark = 1$
 $\checkmark \langle B C \rangle \checkmark = 2$
 $\langle B C D \rangle \times$

Given Strings X & Y of length ' n ' & ' m ' char's, 
the Problem of LCS is to determine a Subsequence
of longest length that is Common to both
' X ' & ' Y ;

\Rightarrow (Realistic Algo)

Appl's:

1) Genomics: Genetic Enss: for DNA Strands Sampling;

$$X = \langle \underline{A B C B D A} \rangle, Y = \langle Y X D B C A D K \rangle$$

$$X' = \langle \text{value} \rangle$$

2) version change: in slw Enss.

3) Data gathering Systems: (Search Engines)

4) Plagiarism: (Research) $\langle \text{Ravi ate an apple} \rangle$
 $\left(\text{Apple was eaten by Ravi} \right)^*$

$$1) \quad X = \langle \text{CGATAATTGAGA} \rangle$$

$$\langle \text{CTAAT} \rangle \checkmark$$

$$\langle \text{TTGAC} \rangle \times$$

$$2) \quad X = \langle \text{ABCBDAB} \rangle, \quad Y = \langle \text{BDCABA} \rangle$$

$$\langle \text{BCB} \rangle \checkmark = 3$$

$$\langle \text{AB} \rangle \checkmark = 2$$

$$\langle \text{BCBA} \rangle \checkmark = 4$$

$$\langle \text{BCAB} \rangle \checkmark = 4$$

$$\underline{\underline{\lambda}} = 4$$

D.P based Soln for L.C.S:



Let 'i' & 'j' be indices into the strings 'x' & 'y' as shown

$$X = \langle \overset{-1}{x_0}, \overset{0}{x_1}, \overset{1}{x_2}, \overset{2}{x_3}, \dots, \overset{i}{x_n} \rangle, \quad Y = \langle \overset{-1}{y_0}, \overset{0}{y_1}, \overset{1}{y_2}, \overset{2}{y_3}, \dots, \overset{j}{y_m} \rangle$$

→ Let $L(i, j)$ denote the length of Common Subsequence of the strings x & y as defined above;

$$L(i, j) = \underline{1} + L(i-1, j-1), \quad \text{if } x[i] = y[j]$$

$$= \max \left\{ L(i-1, j), L(i, j-1) \right\}, \quad \text{if } x[i] \neq y[j]$$

$$L[-1, j] = 0, \quad \text{for } i = -1, j = -1, 0, 1, 2, \dots$$

$$L[i, -1] = 0, \quad \text{for } j = -1, i = -1, 0, 1, 2, \dots$$

Case I:

$X = \langle \text{GTTCTAATA} \rangle$
 $Y = \langle \text{CGATAATTGAGA} \rangle$

$L(9, 11)$

$X[i] = Y[j]$

$L(i, j) = 1 + L(i-1, j-1)$

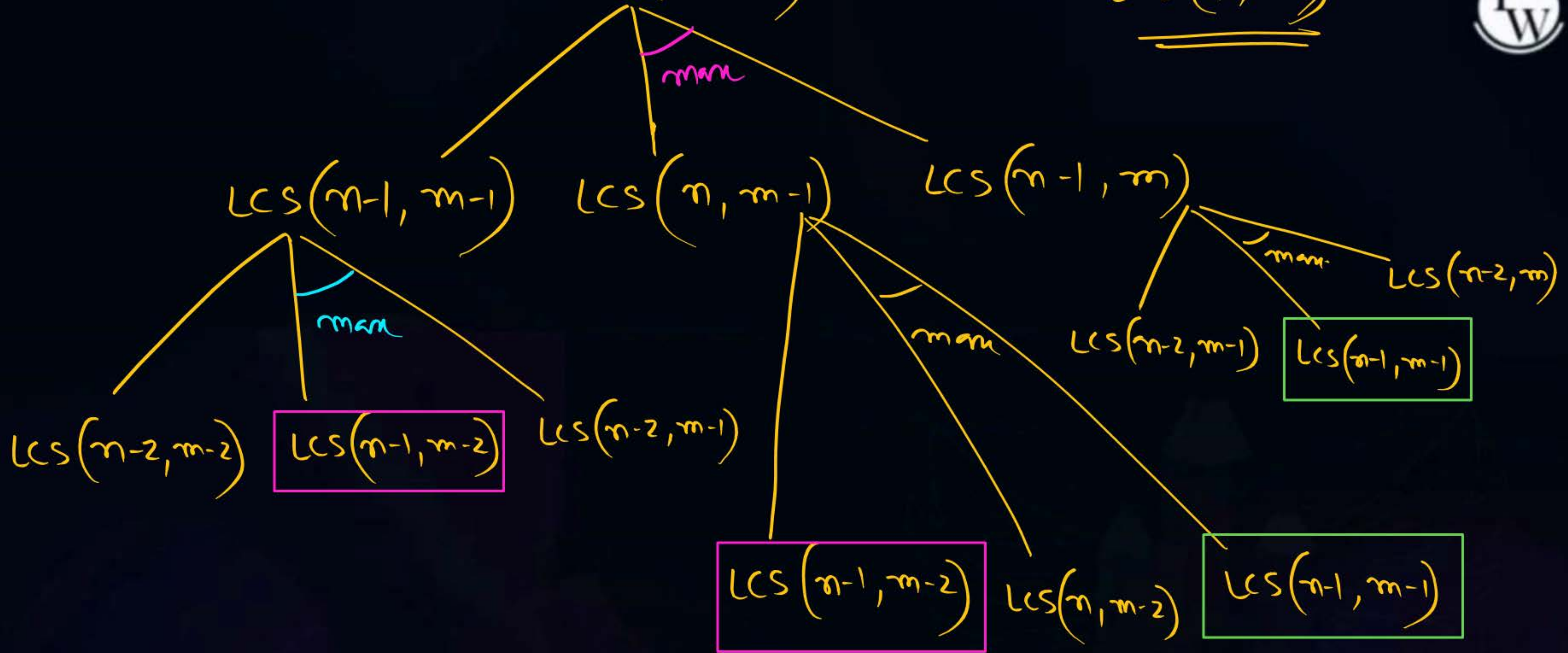
Case II:

$X = \langle \text{GTTCTAATA} \rangle$
 $Y = \langle \text{CGATAATTGAGA} \rangle$

$L(9, 10) =$

$X[i] \neq Y[j]$

$L(i, j) = \max \left\{ L(i-1, j), L(i, j-1) \right\}$

$LCS(n, m)$
 $LCS(4, 5)$


1. Recursion Tree

$$\text{LCS}(\text{"ABBA"} \underline{\text{B}}, \text{"ACBA"} \underline{\text{B}}) = 4$$

$$\text{LCS}(\text{"ABBA"} \downarrow \text{B}, \text{"ACBA"} \downarrow \text{B}) = 3$$

$$\text{LCS}(\text{"ABBA"} \downarrow \text{A}, \text{"ACBA"} \downarrow \text{A}) = 2$$

$$\text{LCS}(\text{"ABBA"} \downarrow \text{B}, \text{"ACBA"} \downarrow \text{B}) = 1$$

$$\text{max} \quad \text{A} \quad \text{1} = \text{LCS}(\underline{\text{"AB"}}, \underline{\text{"A"}})$$

$$\text{A} \quad \text{LCS}(\text{"A"}, \text{"AC"}) = 1$$

$$\text{max} \quad \text{0} = \text{LCS}(\text{"AB"}, \text{" "})$$

$$\text{LCS}(\text{"A"}, \text{"A"}) = 1$$

$$\text{LCS}(\text{"A"}, \text{"A"}) = 1$$

$$\text{0} \quad \text{LCS}(\text{" "}, \text{"AC"}) = 0$$

$$\downarrow \text{+} \quad \text{LCS}(\text{" "}, \text{" "}) = 0$$

$$\downarrow \text{+} \quad \text{LCS}(\text{" "}, \text{" "}) = 0$$

ABAB

AB



$X = \langle A B C B D A B \rangle$; $Y = \langle B D C A B A \rangle$

$-1 \ 0 \ 1 \ 2 \ 3 \ 4 \ 5$

L

		B	D	C	A	B	A
-1		0	0	0	0	0	0
0	A	0	0 \uparrow	0 \uparrow	0 \uparrow	1 \nwarrow_A	1 \nwarrow
1	B	0	1 \nwarrow_B	1 \nwarrow	1 \nwarrow	1 \nwarrow	2 \nwarrow_B
2	C	0	1 \uparrow	1 \uparrow	2 \nwarrow	2 \uparrow	2 \uparrow
3	B	0					
4	D	0					
5	A	0					
6	B	0					4 \nwarrow

x
 $\langle A B C \rangle$
 y
 $\langle B \rangle$
 $\langle B \rangle = 1$

$\langle A B \rangle \langle B D C A B \rangle$

$AB = 2$

$1 + L[i-1, j-1]$

$L[1, 4]$

$L[0, 3] = 1$

(AB)



Topic : Algorithms

Algorithm LCS based on Bottom-up Tabulation method:

Algorithm LCS (x, y)

integer x [0..n] , y[0..m];

{

integer L [0..n , 0..m];

1. for i ← 0 to n - 1

L [i, 0] = 0;

2. for j ← 0 to m - 1

L [0, j] = 0;

3. for i ← 0 to n - 1

for j ← 0 to m - 1

if (x [i] = y [j]) then

L [i, j] = 1 + L [i - 1, j - 1];

else

L [i, j] = max {L[i, j - 1], L [i - 1, j]}

Time : $O(n * m)$

Space: $O(n * m)$

Boundary Condition

6) Matrix-chain Product (MCP)

1) Given Two Square Matrices A & B of order $n \times n$;

The no. of Scalar Multiplications needed to multiply them $(A \times B) = \frac{n^3}{1}$,

$$A \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}_{\substack{2 \times 2 \\ n \times n}} B \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}_{\substack{2 \times 2 \\ n \times n}} = C \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}$$

$$c_{11} = a_{11} \cdot b_{11} + a_{12} \cdot b_{21}$$

$$c_{12} = a_{11} \cdot b_{12} + a_{12} \cdot b_{22}$$

$$c_{21} = a_{21} \cdot b_{11} + a_{22} \cdot b_{21}$$

$$c_{22} = a_{21} \cdot b_{12} + a_{22} \cdot b_{22}$$

$$2 \times 2 \times 2 = 8$$

for $i \leftarrow 1$ to n

for $j \leftarrow 1$ to n

for $k \leftarrow 1$ to n

==

2) Non-Square Matrix Multiplication:



$$A_{2 \times 3} * B_{3 \times 4} = C_{2 \times 4} \Rightarrow \text{Compatible } 3$$

$$\begin{bmatrix} \bullet & \bullet & \bullet \end{bmatrix}_{2 \times 3} * \begin{bmatrix} \bullet \\ \bullet \\ \bullet \end{bmatrix}_{3 \times 4} = \begin{bmatrix} \square \end{bmatrix}_{2 \times 4}$$

$$A_{n \times m} * B_{m \times k} = C_{n \times k}$$

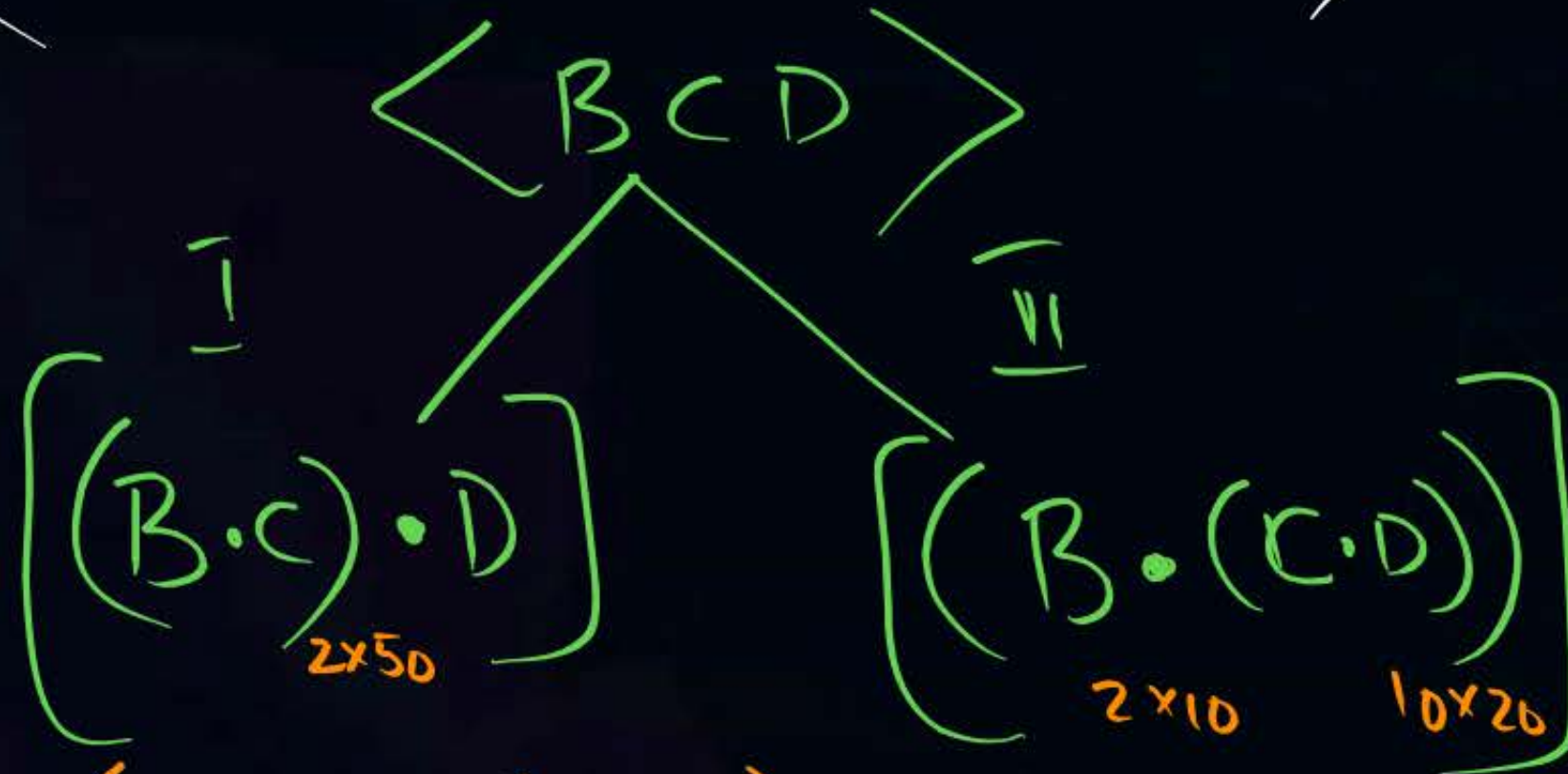
$$\text{Total Scalar Multiplication} = n \times m \times k$$

$$(2 * 4) * 3 = (2 \times 4 \times 3)$$

⇒ Given a chain of Compatible Non-Square Matrices,
it is reqd. to Multiply them together to get
one Final Matrix;



Ex: $\langle B_{2 \times 10} ; C_{10 \times 50} ; D_{50 \times 20} \rangle = \langle BCD \rangle = A_{2 \times 20}$



$$(2 \cdot 10 \cdot 50) + (2 \cdot 50 \cdot 20) \\ 1000 + 2000 = \underline{\underline{3000}}$$

$$2 \cdot 10 \cdot 20 + (10 \cdot 50 \cdot 20) \\ 400 + 10,000 = \underline{\underline{10,400}}$$

(No. of Scalar Multiplications)



Topic : Algorithms

Algorithm Matrix-Chain-Product (p)

```
1  n ← length[p] — 1
2  for i ← 1 to n
3      do m[i, i] ← 0
4  for l ← 2 to n    l is the chain length.
5      for i ← 1 to n - l + 1
6          j ← i + l - 1
7          m[i, j] ← ∞
8          for k ← i to j - 1
9              q ← m[i, k] + m[k + 1, j] + Pi-1PkPj
10             if (q < m[i, j])
11                 then m[i, j] ← q
12                 s[i, j] ← k
13  return m and s
```




Topic : Algorithms

SOS can be implemented using bottom-up DP with Tabulation:

Algorithm SOS (n , M, A)

A [1.....n]

{

integer X[0..n , 0..M];

```
1.  for i ← 0 to n
    for j ← 0 to M
        if (i ≥ 0 and j = 0)
            X [i , j] = T
        else
            if (i = 0 and j > 0)
                X [i, j] = F;
            else
```

if (A [i] > j)

X [i,j] = X [i -1, j]

else

X [i, j] = X [i-1, j] V X [i-1, j -A [i]]

}



Topic : Algorithms

Algorithm Bellman-Ford (G, w, s)

1. Initialize-Single-Source(G, s)
2. for $i \leftarrow 1$ to $|V[G]| - 1$
3. do for each edge $(u, v) \in E[G]$
4. do Relax(u, v, w)
5. for each edge $(u, v) \in E[G]$
6. do if $d[u] > d[v] + w(u, v)$
7. then return FALSE
8. return TRUE



Topic : Algorithms

Initialize-Single-Source (G, s)

```
1  for each vertex  $v \in V[G]$ 
2      do  $d[v] \leftarrow \infty$ 
3       $\pi[v] \leftarrow \text{NIL}$ 
4   $d[s] \leftarrow 0$ 
```

Relax (u, v, w)

```
1  if  $d[v] > d[u] + w(u, v)$ 
2      then  $d[v] \leftarrow d[u] + w(u, v)$ 
3       $\pi[v] \leftarrow u$ 
```


THANK - YOU