BITS, PILANI – K. K. BIRLA GOA CAMPUS

# Design & Analysis of Algorithms

# (CS F364)

**Lecture No. 7**

# Longest Common Subsequence (LCS)

Given two sequences

$$\boldsymbol{X} = \langle x_1, x_2, \ldots, x_m \rangle$$
$$\boldsymbol{Y} = \langle y_1, y_2, \ldots, y_n \rangle$$

find a maximum length common subsequence **(LCS)** of X and Y

**Example**

$X = \langle A, B, C, B, D, A, B \rangle$

**Subsequences of X:**

– A subset of elements in the sequence taken in order

*For example*, $\langle A, B, D \rangle$, $\langle B, C, D, B \rangle$, etc.

# Example

**Example**

$X = \langle A, B, C, B, D, A, B \rangle$

$Y = \langle B, D, C, A, B, A \rangle$

$\langle B, C, B, A \rangle$ is a **longest common subsequence** of X and Y **(length = 4)**

$\langle B, D, A, B \rangle$ is also a **longest common subsequence** of X and Y **(length = 4)**

$\langle B, D, A \rangle$, however is not a LCS of X and Y

# Brute Force Solution

Let **length** of **X be m** & **length** of **Y be n**

**Brute Force**

For every subsequence of X,

check whether it's a subsequence of Y

**Question? How many subsequences are there of X?**

**There are $2^m$ subsequences of X**

**Question?**

**What is time required to check for each subsequence?**

**Each subsequence takes $O(n)$ time to check**

Scan Y for first letter, then scan for second, & so on

**Therefore, Running time: $O(n2^m)$** **Exponential**

# Making the choice

$X = \langle A, B, Z, D \rangle$

$Y = \langle Z, B, D \rangle$

**Choice:** include one element into the common sequence (D) and solve the resulting subproblem

$X = \langle A, B, E, Z \rangle$

$Y = \langle Z, B, E \rangle$

**Choice:** exclude an element from a string and solve the resulting subproblem

# Notations

- **Given a sequence $X = \langle x_1, x_2, \ldots, x_m \rangle$**

**The i-th prefix of X, for $i = 0, 1, 2, \ldots, m$ is**

$$X_i = \langle x_1, x_2, \ldots, x_i \rangle$$

- **$c[i, j]$ = the length of a LCS of the sequences $X_i = \langle x_1, x_2, \ldots, x_i \rangle$ and $Y_j = \langle y_1, y_2, \ldots, y_j \rangle$**

# Recursive Solution

**Case 1: $x_i = y_j$ (Last characters match)**

**Example**

$X_i$ = <D, B, Z, E>

$Y_j$ = <Z, B, E>

$$c[i, j] = c[i - 1, j - 1] + 1$$

- **Append $x_i = y_j$ to the LCS of $X_{i-1}$ and $Y_{j-1}$**
- **Must find a LCS of $X_{i-1}$ and $Y_{j-1}$**

# Recursive Solution

**Case 2:** $x_i \neq y_j$ **(Last characters do not match)**

**In this case** $x_i$ **and** $y_j$ **cannot both be in the LCS**

**Thus either** $x_i$ **is not part of the LCS, or** $y_j$ **is not part of the LCS**

**(and possibly both are not part of the LCS).**

- **In the 1st case LCS of** $X_i$ **and** $Y_j$ **is the LCS of** $X_{i-1}$ **and** $Y_j$
- **In the 2nd case LCS of** $X_i$ **and** $Y_j$ **is the LCS of** $X_i$ **and** $Y_{j-1}$

**Example**

$X_i = \langle A, B, Z, G \rangle$ **&** $Y_j = \langle A, D, Z \rangle$

1. **LCS of** $X_{i-1} = \langle A, B, Z \rangle$ **and** $Y_j = \langle A, D, Z \rangle$
2. **LCS of** $X_i = \langle A, B, Z, G \rangle$ **and** $Y_{j-1} = \langle D, Z \rangle$

**Therefore,** $c[i, j] = \max \{ c[i-1, j], c[i, j-1] \}$

# Recursive Solution

$$c[i, j] = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ c[i-1, j-1] + 1 & \text{if } x_i = y_j \\ \max(c[i, j-1], c[i-1, j]) & \text{if } x_i \neq y_j \end{cases}$$

# Optimal Substructure

**Optimal Substructure Property**

Easy to prove that in both the cases

**Optimal solution to a problem includes optimal solutions to subproblems**

**Cut-Paste Argument**

# Overlapping Subproblems

**To find a LCS of X and Y**

**We may need to find the LCS between**

**X and $Y_{n-1}$ and that of $X_{m-1}$ and Y**

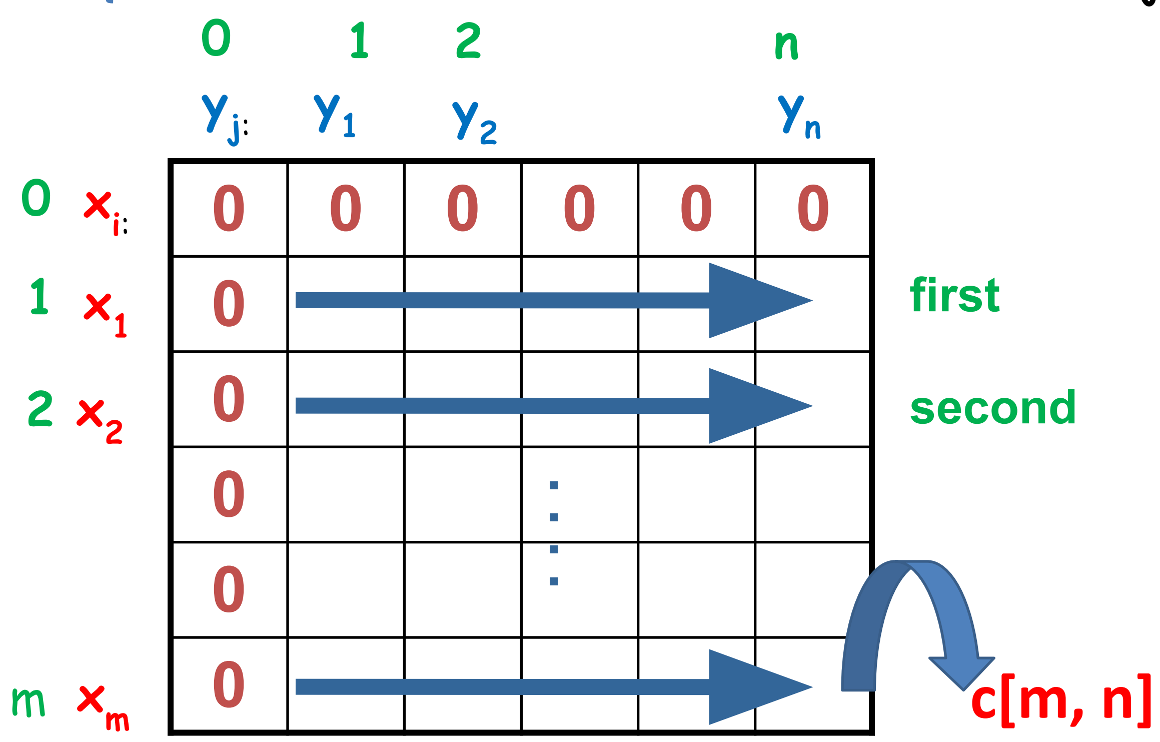
**Both the above subproblems has the subproblem of finding the following:**

**LCS of $X_{m-1}$ and $Y_{n-1}$**

**Subproblems share subsubproblems**

# Computing the Length of the LCS

$$c[i, j] = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ c[i-1, j-1] + 1 & \text{if } x_i = y_j \\ \max(c[i, j-1], c[i-1, j]) & \text{if } x_i \neq y_j \end{cases}$$



# Computing the table

$$c[i, j] = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ c[i-1, j-1] + 1 & \text{if } x_i = y_j \\ \max(c[i, j-1], c[i-1, j]) & \text{if } x_i \neq y_j \end{cases}$$

**Along with c[i, j] we also compute and record b[i, j] which tells us what choice was made to obtain the optimal value**

If $x_i = y_j$

$b[i, j] = $ "↖"

Else, if $c[i-1, j] \geq c[i, j-1]$

$b[i, j] = $ " ↑ "

Else

$b[i, j] = $ " ← "

# Pseudo Code for LCS

```
for i ← 1 to m
    do c[i, 0] ← 0
for j ← 0 to n
    do c[0, j] ← 0
for i ← 1 to m
    do for j ← 1 to n
        do if x_i = y_j
            then c[i, j] ← c[i - 1, j - 1] + 1
                b[i, j ] ← "↖"
            else if c[i - 1, j] ≥ c[i, j - 1]
                then c[i, j] ← c[i - 1, j]
                    b[i, j] ← "↑"
                else c[i, j] ← c[i, j - 1]
                    b[i, j] ← "←"
return c and b
```

**Running time: $O(mn)$**

# Example

$X = \langle A, B, C, B, D, A, B \rangle$

$Y = \langle B, D, C, A, B, A \rangle$

$$c[i, j] = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ c[i-1, j-1] + 1 & \text{if } x_i = y_j \\ \max(c[i, j-1], c[i-1, j]) & \text{if } x_i \neq y_j \end{cases}$$

If $x_i = y_j$

$b[i, j] =$ " ↖ "

Else if $c[i-1, j] \geq c[i, j-1]$

$b[i, j] =$ " ↑ "

else

$b[i, j] =$ " ← "

| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|---|
| | $y_j$ | | B | D | C | A | B | A |
| 0 | $x_i$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | A | 0 | 0 ↑ | 0 ↑ | 0 ↑ | ↖ 1 | ← 1 | ↖ 1 |
| 2 | B | 0 | ↖ 1 | ← 1 | ← 1 | 1 ↑ | ↖ 2 | ← 2 |
| 3 | C | 0 | ↑ 1 | ↑ 1 | ↖ 2 | ← 2 | ↑ 2 | ↑ 2 |
| 4 | B | 0 | ↖ 1 | ↑ 1 | ↑ 2 | ↑ 2 | ↖ 3 | ← 3 |
| 5 | D | 0 | ↑ 1 | ↖ 2 | ↑ 2 | ↑ 2 | ↑ 3 | ↑ 3 |
| 6 | A | 0 | ↑ 1 | ↑ 2 | ↑ 2 | ↖ 3 | ↑ 3 | ↖ 4 |
| 7 | B | 0 | ↖ 1 | ↑ 2 | ↑ 2 | ↑ 3 | ↖ 4 | ↑ 4 |

# Constructing a LCS

**Start at b[m, n] and follow the arrows**

**When we encounter a "↖" in b[i, j]**
**⇒ $x_i = y_j$ is an element of the LCS**

**LCS is BCBA**

| | | $y_j$ | B | D | C | A | B | A |
|---|---|---|---|---|---|---|---|---|
| 0 | $x_i$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | A | 0 | 0↑ | 0↑ | 0↑ | ↖1 | ←1 | ↖1 |
| 2 | B | 0 | ↖1 | ←1 | ←1 | 1↑ | ↖2 | ←2 |
| 3 | C | 0 | ↑1 | ↑1 | ↖2 | ←2 | ↑2 | ↑2 |
| 4 | B | 0 | ↖1 | ↑1 | ↑2 | ↑2 | ↖3 | ←3 |
| 5 | D | 0 | ↑1 | ↖2 | ↑2 | ↑2 | ↑3 | ↑3 |
| 6 | A | 0 | ↑1 | ↑2 | ↑2 | ↖3 | ↑3 | ↖4 |
| 7 | B | 0 | ↖1 | ↑2 | ↑2 | ↑3 | ↖4 | ↑4 |