Longest common subsequence

i

Xi = i-1D….

**A B C**

j

Yj = j-1 D……

**A C B**

We assume that we have found solutions to smaller sub problems you know the longest common subsequence till the subproblem i-1 length. I know the solution to Xi till i-1 and Yj till j-1. Here its AB or AC. Cn have 2 solutions

If you are at index i in Xi and index j in Yj, we can look into the to add the character pointed to by I and j if both matches.

Algorithm is

LCS[i][j] = LCS[i-1][j-1] + 1 if X[i] = Y[j] // length

Xi = i-1D….

**A B C**

j

Yj = j-1 A……

**B A D**

Here length of common subsequence till i-1 ,j-1 is 1

LCS[i-1][j-1] = 1

= LCS[i-1][j-1] if X[i] # Y[j] is wrong

It should be

Max[LCS[i-1][j], LCS[i][j-1]) if X[i] # Y[j]

LCS[i][j-1] = 2 // ABCD and BAD

LCS[i-1][j] = 1 // ABC BADA

Max[LCS[i-1][j],LCS[i][j-1]) if X[i] # Y[j]

We use this logic to solve solutions to sub problems, storing in table which leads to solution to bigger problem.

Base case 0 if I = 0 and j = 0

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Y[j]  X[i] | None | B | D | C | A | B |
| None | 0 | 0 | 0 | 0 | 0 | 0 |
| A | 0 | 0 | 0 | 0 | 1 | 1 |
| B | 0 | 1 | 1 | 1 | 1 | 2 |
| C | 0 | 0 | 1 | 2 | 2 | 2 |
| B | 0 | 0 | 1 | 2 | 2 | 3 |
| D | 0 | 0 | 1 | 2 | 2 | **3** |

t[i][j] = LCS(Xi,Yj)

Xi is string through I and Yj is the string through j

X is empty first row is empty i.e. 0. If Y is empty first column s empty i.e. 0.

Now use the logic above if 2 characters are equal and not equal.

Length of longest LCS is 3

Diagonal ones will tell you if character is taken or not . or corresponding characters are equal.

Algorithm

LCS(X,Y)

M = X.length n = Y.length

Create a 2D array t with m+1 rows and n+1 columns

Fill the first row and first column with 0

For I = 1 to m

For j = 1 to n

If (X[i] = Y[j])

t[i][j] = t[i-1][j-1] + 1

else

t[i][j] = Max(t[i-1][j],t[i][j-1])

if ( t[i-1][j] > t[i][j-1)

t[i][j] = t[i-1][j] // previous row

else

t[i][j] = t[i][j-1] // previous column

Construct LCS

I = m; j = n

While (I >= 1 and j >= 1)

If (x[i] = y[j])

Add x[i] to LCS

i--; j— //go diagonal

else

if t[i-1][j] >= t[i][j-1]

i— // previous row and move, donot take any characters

else

j-- // previous column