Algorithms Design and Analysis [ETCS-301]

Dr. A K Yadav
Amity School of Engineering and Technology
(affiliated to GGSIPU, Delhi)
akyadav1@amity.edu
akyadav@akyadav.in
www.akyadav.in
+91 9911375598

August 29, 2019

Longest Common Subsequence

- Given two sequences $X = \langle x_1, x_2, \ldots, x_m \rangle$ of length m and $Y = \langle y_1, y_2, \ldots, y_n \rangle$ of length n
- $Z = \langle z_1, z_2, \ldots, z_k \rangle$ is a common subsequence of X and Y if Z is a subsequence of both X and Y
- Z is a Longest common subsequence of X and Y if X and Y have no common subsequence of length greater then Z
- ith prefix of X is $X_i = \langle x_1, x_2, \ldots, x_i \rangle$ for $i = 0, 1, \ldots, m$

Step 1: Characterizing a longest common subsequence

- Let $X = \langle x_1, x_2, \ldots, x_m \rangle$ and $Y = \langle y_1, y_2, \ldots, y_n \rangle$ be two sequences, and let $Z = \langle z_1, z_2, \ldots, z_k \rangle$ be any LCS of X and Y.
- **Optimal substructure of an LCS**
- If $x_m = y_n$ then $z_k = x_m = y_n$ and Z_{k-1} is an LCS of X_{m-1} and Y_{n-1}
- If $x_m \neq y_n$ then $z_k \neq x_m$ implies that Z is an LCS of X_{m-1} and Y
- If $x_m \neq y_n$ then $z_k \neq y_n$ implies that Z is an LCS of X and Y_{n-1}

Step 2: A recursive solution

- In step 1 there are two cases:
- Case 1: if $x_m = y_n$ then find the LCS of X_{m-1} and Y_{n-1}
- Appending x_m to LCS of X_{m-1} and Y_{n-1} gives LCS of X and Y
- Case 2: if $x_m \neq y_n$ then solve two sub-problems: find LCS of X_{m-1} and Y and find LCS of X and Y_{n-1}
- Longer of these two LCS will be the LCS of X and Y
- Let $c[i, j]$ is the length of an LCS of the sequences X_i and Y_j
- If either of two sequences is empty i.e. $i = 0$ or $j = 0$ then LCS will be of length zero that is $c[i, j] = 0$
- Recursive solution will be

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

Step 3: Computing the length of an LCS I

- Let $X = \langle x_1, x_2, \ldots, x_m \rangle$ and $Y = \langle y_1, y_2, \ldots, y_n \rangle$ be two sequences, and let $Z = \langle z_1, z_2, \ldots, z_k \rangle$ be any LCS of X and Y.
- Let Table $c[i, j]$ to store the length of an LCS of the sequences X_i and Y_j
- $c[0 \ldots m, 0 \ldots n]$
- Let Table $b[i, j]$ to store the direction of the flow of length of LCS to construct an optimal solution
- $b[1 \ldots m, 1 \ldots n]$

Step 3: Computing the length of an LCS II

Bottom-up Approach

LCS-LENGTH(X, Y)

```
m = length(X)
n = length(Y)
Let c[0...m, 0...n] and b[1...m, 1...n] two new tables
for i = 1 to m
    c[i, 0] = 0//Y is empty
for j = 1 to n
    c[0, j] = 0//X is empty
for i = 1 to m
    for j = 1 to n
        if x_i = y_j
            c[i, j] = c[i - 1, j - 1] + 1
```

Step 3: Computing the length of an LCS III

$b[i, j] = \nwarrow$
else if $c[i-1, j] \geq c[i, j-1]$
$c[i, j] = c[i-1, j]$
$b[i, j] = \uparrow$
else
$c[i, j] = c[i, j-1]$
$b[i, j] = \leftarrow$
return c, b

Length of the LCS is **c[m,n]**
Complexity of the algorithm is $\Theta(mn)$

Top-down Approach:

MEMOIZED-LCS(X, Y, m, n)

1. Let $c[0 \ldots m, 0 \ldots n]$
2. for $i = 0$ to m
3. for $j = 0$ to n
4. $c[i,j] = 0$
5. return LCS(X, Y, c, b, m, n)

LCS(X, Y, c, b, i, j)

1. if $c[i,j] > 0$
2. return $c[i,j]$
3. if $x_i = y_j$
4. $c[i,j] = LCS(X, Y, c, b, i-1, j-1] + 1$
5. $b[i,j] = \nwarrow$

```
else if LCS(X, Y, c, b, i − 1, j) ≥ LCS(X, Y, c, b, i, j − 1)
    c[i, j] = LCS(X, Y, c, b, i − 1, j)
    b[i, j] = ↑
else
    c[i, j] = LCS(X, Y, c, b, i, j − 1)
    b[i, j] = ←
return c[i, j]
```

Step 4: Constructing an LCS I

- Step 3 calculates $c[i,j]$ and $b[i,j]$
- $c[i,j]$ gives the length of LCS of X_i and Y_j
- $b[i,j]$ tells from where we calculated $c[i,j]$
- $\nwarrow$ shows that $x_i = y_j$ and part of LCS
- $\uparrow$ shows that $c[i-1,j] \geq c[i,j-1]$
- $\leftarrow$ shows that $c[i,j-1] > c[i-1,j]$
- call PRINT-LCS(b, X, m, n)

PRINT-LCS(b, X, i, j)

1. if $i = 0$ or $j = 0$
2.   return
3. if $b[i,j] = \nwarrow$
4.   PRINT-LCS$(b, X, i-1, j-1)$
5.   print x_i

6. else if $b[i,j] = \uparrow$
7. PRINT-LCS$(b, X, i-1, j)$
8. else
9. PRINT-LCS$(b, X, i, j-1)$

Complexity of the algorithm is $\Theta(m+n)$

Thank you

Please send your feedback or any queries to akyadav1@amity.edu, akyadav@akyadav.in or contact me on +91 9911375598