CS F364: Design & Analysis of Algorithm

13 Longest Common Subsequence

Dr. Kamlesh Tiwari
Assistant Professor, Department of CSIS,
BITS Pilani, Pilani Campus, Rajasthan-333031 INDIA

Feb 15, 2021 ONLINE (Campus @ BITS-Pilani Jan-May 2021)

http://ktiwari.in/algo

Longest common subsequence

A subsequence of a sequence can be obtained by removing zero or more elements.

- In the longest-common-subsequence problem, we are given two sequences $X =< x_1 x_2 x_3 ... x_m >$ and $Y =< y_1 y_2 y_3 ... y_n >$ and wish to find a maximum-length common subsequence of X and Y.
- Example: DNA sequence $\{A, T, C, G\}$
- Time? exponential
- Prefix uses first few items

Components of Dynamic Programming

1) Optimal substructure, 2) Overlapping subproblems

Let
$X_m =< x_1, x_2, x_3, ..., x_m >$
$Y_n =< y_1, y_2, ..., y_n >$
$Z_k =< z_1, ..., z_k >$ LCS

1. If $(x_m = y_n)$ then $z_k = x_m$ and $Z_{k-1} = LCS(X_{m-1}, Y_{n-1})$
2. If $(x_m \neq y_n)$ then
   If $(z_k \neq x_m)$ then $Z_k = LCS(X_{m-1}, Y_n)$
   If $(z_k \neq y_n)$ then $Z_k = LCS(X_m, Y_{n-1})$

LCS of two sequences contains within LCS of the prefix of the two sequences

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

Solution Sketch

		B	D	C	A	B	A
A							
B							
C							
B							
D							
A							
B							

Longest common subsequence

Algorithm 1: LCS-Length(X, Y)

```
1  m = length[X]
2  n = length[Y]
3  for i = 1 to m do
4      c[i,0] = 0
5  for j = 0 to n do
6      c[0,j] = 0
7  for i = 1 to m do
8      for j = 1 to n do
9          if x_i = y_j then
10             c[i,j] = c[i-1,j-1]+1
11             b[i,j] = ↖
12         else
13             if c[i-1,j]≥c[i,j-1] then
14                 c[i,j] = c[i-1,j]
15                 b[i,j] = ↑
16             else
17                 c[i,j] = c[i,j-1]
18                 b[i,j] = ←
19 return c and b
```

Complexity? $O(n^2)$

Longest common subsequence

Algorithm 2: Print-LCS(b, X, i, j)

```
1  if i=0 or j=0 then
2      return
3  if b[i,j] = ↖ then
4      Print-LCS( b, X, i-1, j-1)
5      print x_i
6  else
7      if b[i,j] = ↑ then
8          Print-LCS( b, X, i-1, j)
9      else
10         Print-LCS( b, X, i, j-1)
```

Example

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

Thank You!

Thank you very much for your attention! (Reference[1])

Queries ?

[1] Book - *Introduction to Algorithm*, By THOMAS H. CORMEN, CHARLES E. LEISERSON, RONALD L. RIVEST, CLIFFORD STEIN