Department of
Computer Science and Engineering

Title: Implemention of Longest Common Subsequence (LCS) Algorithm

Algorithms Lab

CSE 206

Green University of Bangladesh

1 Objective(s)

- To learn about Longest Common Subsequence (LCS) algorithm for determining the length of common subsequences in strings.

2 Problem Analysis

2.1 Longest Common Subsequence (LCS) Algorithm

A subsequence is a group of sequences that appear in the same relative order, whether they are adjacent or not. The longest subsequence that is shared by all the given sequences, is known as the longest common subsequence (LCS), assuming that the elements of the subsequence are not needed to occupy consecutive position within the original sequences.

2.2 How LCS algorithm works

Let the sequences are X=x1,x2,x3,...,xm and Y=y1,y2,y3,...,ym. We will use the following steps to determine the length of the longest common subsequence.

- Create an empty adjacency table with the dimensions n m, where n is the length of the X sequence and m is the length of the Y sequence. The elements in sequence X are represented by the table's rows, and the elements in sequence Y are represented by the table's columns.
- Rows and columns starting at zero must contain only zeros. And the remaining values are filled in based on different cases, by maintaining a counter value.
- The number is increased by one if it comes across a common element in both the X and Y sequences.
- To fill in T[i, j] if the counter does not pass into any common elements in the X and Y sequences, choose the biggest value between T[i-1, j] and T[i, j-1].
- Once the table is filled, backtrack from the last value in the table. Backtracking here is done by tracing the path where the counter incremented first.
- The longest common subsequence which can be determined by observing the path's elements.

2.3 Example of LCS algorithm

- Consider these two strings X=BACDB and Y=BDCB to find the longest common subsequence.
- Considering the steps, we need to calculate two tables 1 and 2.
- Given n = length of X, m = length of Y, X = BDCB, Y = BACDB

2.3.1 Constructing the LCS Tables

1. In the table given in figure 1 below, the zeroeth rows and columns are filled with zeroes. The remaining values are filled by incrementing and choosing the maximum values according to the algorithm.
2. After the values have been filled in, the path is retraced starting at the last value at T[4, 5] in the table.
3. It is possible to figure out the longest common subsequence from the observed path By selecting the values where the counter is first incremented.
4. Throughout this scenerio, the number is increased three times, from B to C to B, so the final count is 3. As a result, BCB is the longest common subsequence of the sequences X and Y. The final table is depicted in figure-2

		0	1	2	3	4
			B	D	C	B
0		0	0	0	0	0
1	B	0	1	1	1	1
2	A	0	1	1	1	1
3	C	0	1	1	2	2
4	D	0	1	2	2	2
5	B	0	1	2	2	3

Figure 1: Constructing the LCS table

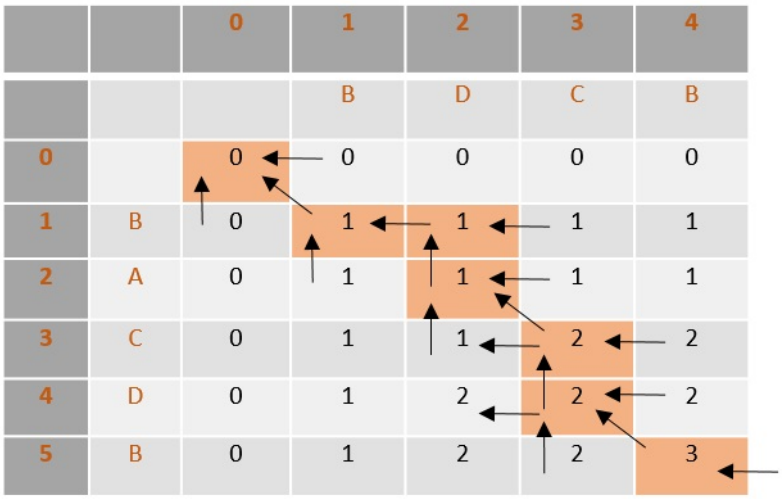

Figure 2: Identifying the longest common subsequence

3 Algorithm

Algorithm 1: LCS Algorithm

```
X and Y be two given sequences
m := length(X)
n := length(Y)
for i = 1 to m do do
    LCS[i, 0] := 0
end
for j = 1 to n do do
    LCS[0, j] := 0
end
for i = 1 to m do do
    for j =1 to n do do
        if X[i] = Y[j] then
            LCS[i][j] = 1 + LCS[i-1, j-1]
        end
        else
            LCS[i][j] = max(LCS[i-1][j], LCS[i][j-1])
        end
    end
end
```

4 Implementation in Java

```java
import java.util.*;
import java.lang.*;
public class LCS {

   public static int lcs(char X[], char Y[], int m ,int n) {
      // The zeroeth rows and columns are filled with zeroes
      if(m == 0 || n == 0) {
         return 0;
      }
      // The remaining values are filled by incrementing and choosing the
         maximum values according to the algorithm.
      if(X[m-1] == Y[n-1]) {
         return 1 + lcs(X,Y,m-1,n-1);
      } else {
         return max(lcs(X, Y, m, n-1), lcs(X,Y, m-1,n));
      }
   }

   // Finding the maximum between lcs(X, Y, m, n-1), lcs(X,Y, m-1,n)
    static int max(int l1, int l2) {
      if(l1<l2) {
         return l2;
      } else {
         return l1;
      }
   }

   public static void main(String[] args) {
      LCS lcs = new LCS();
      String X, Y;
      X = "BDCB"; // Sequence 1
```

```
        Y = "BACDB"; // Sequence 2
        char arr1[] = X.toCharArray();
        char arr2[] = Y.toCharArray();
        // Finding the length of two sequences
        int len1 = arr1.length;
        int len2 = arr2.length;
        // Printing the length of the longest common subsequence
        System.out.print("Length of LCS is: " + lcs(arr1, arr2, len1, len2));
    }
}
```

5 Sample Input/Output

Length of LCS is: 3

6 Lab Task (Please implement yourself and show the output to the instructor)

1. Write a Program in java to print the length of second longest subsequence.
2. Write a Program in java to print the longest common subsequence using LCS algorithm

6.1 Problem analysis

The longest common subsequence problem is a well-known computer science issue that serves as the foundation for data comparison tools like the diff-utility and has a variety of real-world uses.

- For given two strings X and Y, the longest common subsequence (LCS) problem is to find the longest subsequence common to both X and Y.
- Has applications to DNA similarity testing (alphabet is A,C,G,T), bioinformatics etc. It is also widely used by revision control systems, such as SVN and Git to resolve numerous changes made to a collection of files under revision control.
- Example: ABCDEFG and XZACKDFWGH have ACDFG as a longest common subsequence.

We have two nested loops where the outer one iterates m times and the inner one iterates n times. A constant amount of work is done inside each iteration of the inner loop. Thus, the total running time is O(nm). Answer is contained in LCS[m,n] and the subsequence can be recovered from the LCS table.

7 Lab Exercise (Submit as a report)

- Print all the common subsequences according to the descending order of the lengths for two given sequences.
- What will be the LCS for the sequences “ABCDEFGH" & “abcdefgh" ?

8 Discussion & Conclusion

Based on the focused objective(s) to understand about the LCS algorithms, the additional lab exercise made me more confident towards the fulfillment of the objectives(s).

9 Policy

Copying from internet, classmate, seniors, or from any other source is strongly prohibited. 100% marks will be *deducted* if any such copying is detected.