ECE374 Fall2020

Lab10: Longest Common Subsequence

Name: Zhang Yichi 3180111309

Dec. 15th 2020

1 Introduction

In this lab, we will implement a program to search for the longest common subsequence from two strings using brute force and dynamic programming algorithm.

2 Python Code for the LCS Problem

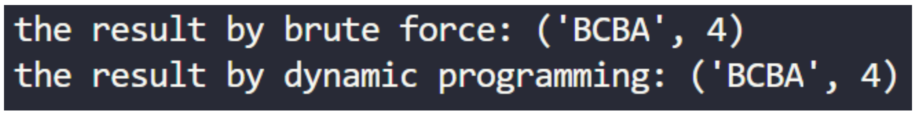
```python
# brute force
# to test if a is b's subsequence
def isSubsequence(a,b):
    i = j = 0
    while i < len(a) and j < len(b):
        if a[i] == b[j]:
            i += 1
        j += 1
    return i==len(a)

def LCS_bf(X,Y):
    subX = []
    m = len(X)
    n = len(Y)
    lcs = None
    maxlen = 0
    for i in range(2**m):
        temp = ""
        for j in range(m):
            if i & (1<<j):
                temp += X[j]
        subX.append(temp)
    for i in subX:
        if len(i) > maxlen and isSubsequence(i,Y):
            maxlen = len(i)
            lcs = i
    return lcs,maxlen

# dynamic programming
```

```
def LCS_dp(X,Y):
    m = len(X)
    n = len(Y)
    if m==0 or n==0:
        return None,0
    c = [[0]*(n+1) for i in range(m+1)]
    b = [[None]*(n+1) for i in range(m+1)]
    for i in range(1,m+1):
        for j in range(1,n+1):
            if X[i-1] == Y[j-1]:
                c[i][j] = 1 + c[i-1][j-1]
                b[i][j] = [i-1,j-1]
            elif c[i][j-1] <= c[i-1][j]:
                c[i][j] = c[i-1][j]
                b[i][j] = [i-1,j]
            else:
                c[i][j] = c[i][j-1]
                b[i][j] = [i,j-1]
    maxlen = c[m][n]
    lcs = ""
    while b[m][n] != None:
        if b[m][n] == [m-1,n-1]:
            lcs = X[m-1] + lcs
        m,n = b[m][n]
    return lcs,maxlen
```

There are two main functions. LCS_bf solves the LCS problem via brute force and LCS_dp solves it by dynamic programming. For LCS_bf, I generate all the subsequences, check if they are the subsequences for both two input strings X, Y and compare them to find the longest one. While for LCS_dp, I first build a 2D table where each entry [i,j] stores the information for the LCS of X[1...i] and Y[1...j] and then complete it from bottom up by two for loops.

3 Test Example for the LCS Problem

```
X = "ABCBDAB"
Y = "BDCABA"
print(f"the result by brute force: {LCS_bf(X,Y)}")
print(f"the result by dynamic programming: {LCS_dp(X,Y)}")
```



```
the result by brute force: ('BCBA', 4)
the result by dynamic programming: ('BCBA', 4)
```

Figure 1: LCS problem test results.

Call the two functions to see the results.

4 Time Complexity

Brute force: $O(n2^m)$
Dynamic programming: $O(mn)$