CS 101 Fall 2021 - Quiz 10-A

12/20/2021 - 25 Minutes **Name:** **ID number:**

NOTE: Please write down the subproblem, recurrence equation and the time complexity. Breifly explain why.

Problem 1 Longest Common Subsequence (5 pts)

Given two strings s1 and s2, find out the length of their longest common subsequence using dynamic programming. A subsequence of a string is a new string generated from the original string with some characters (can be none) deleted without changing the relative order of the remaining characters. For example, "ace" is a subsequence of "abcde". A common subsequence of two strings is a subsequence that both strings have.

Write down the subproblem and recurrence equation.

Subprolem: dp[i][j] as the length of the longest common subsequence considering s1[1..i] and s2[1..j]

Bellman Equationi: $dp[i][j] = \begin{cases} dp[i-1][j-1]+1, & s1[i] = s2[j] \\ \max(dp[i-1][j], dp[i][j-1]), & s1[i] \neq s2[j] \end{cases}$

base case: dp[0][0...n],dp[0...m][0]

Problem 2 "01"-Problem (5 pts)

In the computer world, use restricted resources to generate maximum benefit is what we always want to pursue. Assume you are given a set of binary strings $strs$ of size l, and two integers m and n.

You need to find out the maximum number of strings in $strs$ that you can form using m 0's and n 1's (you don't need to use all of them and every 0 and 1 can only be used at most once).

For example, strs = {"10", "0", "1"}, m = 1, n = 1, the maximum number of strings in strs that you can form is 2.

Describe your dynamic programming algorithm and the time complexity.

Define subprolems as dp[i][j][k] as the maximum number of strings that can be formed considering the first i strings in $strs$ and using j 0's and k 1's.

If we count the number of 0 and 1 in i'th string, denoting them as zeros and ones, then if j $\leq$ zeros or k $\leq$ ones, we can't add i'th string to the final set, otherwise we take max on the two cases.

So we have the following recurrence equation:

$$dp[i][j][k] = \begin{cases} dp[i-1][j][k], & j < \text{ zeros } \mid k < \text{ ones} \\ \max(dp[i-1][j][k], dp[i-1][j - \text{ zeros }][k - \text{ ones }] + 1), & j \geq \text{ zeros } \& k \geq \text{ ones} \end{cases}$$

The time complexity is $O(lmn)$ since this is a 3-D bellman equation.