CSCI 3104 Fall 2022
Instructors: Prof. Grochow and Chandra Kanth Nagesh

Quiz 8 S22

Due Date .... Thursday Nov 10, 2022 8pm MT
Name .... **Tyler Huynh**
Student ID **109603994**
Quiz Code (enter in Canvas to get access to the LaTeX template) .... **IKJHG**

Contents

Instructions

- You may either type your work using this template, or you may handwrite your work and embed it as an image in this template. **If you choose to handwrite your work, the image must be legible, and oriented so that we do not have to rotate our screens to grade your work.** We have included some helpful LaTeX commands for including and rotating images commented out near the end of the LaTeX template.
- You should submit your work through the **class Gradescope page** only. Please submit one PDF file, compiled using this LaTeX template.
- You may not need a full page for your solutions; pagebreaks are there to help Gradescope automatically find where each problem is. Even if you do not attempt every problem, please submit this document with no fewer pages than the blank template (or Gradescope has issues with it).
- You **may not collaborate with other students**. **Copying from any source is an Honor Code violation. Furthermore, all submissions must be in your own words and reflect your understanding of the material.** If there is any confusion about this policy, it is your responsibility to clarify before the due date.
- Posting to **any** service including, but not limited to Chegg, Discord, Reddit, StackExchange, etc., for help on an assignment is a violation of the Honor Code.
- You **must** virtually sign the Honor Code (see Section). Failure to do so will result in your assignment not being graded.

Honor Code (Make Sure to Virtually Sign)

Problem HC.

- My submission is in my own words and reflects my understanding of the material.
- Any collaborations and external sources have been clearly cited in this document.
- I have not posted to external services including, but not limited to Chegg, Reddit, StackExchange, etc.
- I have neither copied nor provided others solutions they can copy.

I agree to the above, Tyler Huynh. □

22 Standard 22: Dynamic Programming: Write down recurrences

Problem 22. Consider a modified version of the string alignment (aka edit distance) problem that we saw in class. The allowed operations are: insert, delete, substitute, and *swap*. **Note that the swap operation is new compared to what we covered in class.** (Also, a no-op, which is free, as in class.)

The **swap operation** replaces any two adjacent letters by swapping them. For example, performing a swap on *there* could result in *htere* (by swapping the first two letters), *tehre* (by swapping the second two letters), *three* (by swapping the *er* in the middle), or *theer* (by swapping the last two letters.

Given two strings x and y, let

$$T[i,j] = \text{ minimum number of operations to transform the prefix } x[1..i] \text{ into the prefix } y[1..j].$$

Your job is to write down a recurrence for this dynamic programming table. Make sure to include your base cases and justify your recurrence.

As a starting point, we have included the recurrence from class for the simpler version of string alignment we did in class; if you choose to start from this recurrence, you will have to edit it appropriately.

Answer. **Recurrence from class. This is not a correct answer to this quiz question, we are just providing it as a starting point.**

$$T[i,j] = \begin{cases} i & j = 0 \\ j & i = 0 \\ T[i-1,j-1] & x_i = y_j \\ 1 + \min\{T[i,j-1], T[j-1,i], T[i-1,j-1]\} & x_i \neq y_j \end{cases}$$

I will first begin by considering the base cases and cases for the problem first, where it follows:

- **Base Cases:**
  When i and j are both equal to 0 then no operations will be done, such that the minimum number of operation to transform the $x[1..i]$ into the prefix $y[1..j]$ would be 0.

  Another base case would be a no-operation where if x_i and y_j are equal to each other than we will perform no-operation on the set of strings.

- **Case 1:**
  In this case when $x_i \neq y_j$, than we have the option to either delete, insert, or substitute the characters in the string, with a cost of 1 + the minimum of these operations.

- **Case 2:**
  In this case when $x_{i-1}x_i = y_i y_{i-1}$, it would be in this instance that you can swap because two adjacent characters in x are the same as two adjacent characters in y, except that the adjacent characters in x are backwards.

  Take for example with the string "there" and the string "htere", where in x the two characters would t and h and the two characters in y would be h and t, from here we can then swap them to allow for the strings to be aligned.

  From this we can see that the recursive structure for the swap operation would be to $T[i-2, j-2]+1$ where $T[i-2, j-2]$ represents the minimum cost of aligning the rest of the string not including the characters that will be swapped because we already know that once we swap these characters, where we know that $x_{i-1}x_i = y_i y_{i-1}$ will be aligned after swapping. When we subtract 2 we are looking at two adjacent characters within the strings. The 1 represents the cost of the operation of swapping.

From the above our updated recurrence will be:

$$T[i,j] = \begin{cases} i & j = 0 \\ j & i = 0 \\ T[i-1, j-1] & x_i = y_j \\ T[i-2, j-2] + 1 & x_{i-1}x_i = y_i y_{i-1} \\ 1 + \min\{T[i, j-1], T[j-1, i], T[i-1, j-1]\} & x_i \neq y_j \end{cases}$$

□