Fernando Marquez

CS 2302 TTR 10:30 - 12:00

Lab 7 Report

*Edit Distance*

The objective of lab 7 was to create a dynamic program using edit distance. Edit distance is an algorithm that can have 2 strings and the task is to find the minimum number of operations to transform string 1 into string 2.

I created a table to store the values in a bottom up manner. I populated the table then checked if the strings were empty. If string 1 was empty then insert all of string 2. If string 2 was empty then remove all the contents of string 2. If the strings have a character in common then ignore the operation and proceed through the rest of the string. If characters are not the same string then find the minimum operations to transform string 1 into string 2. I then wanted to check if strings could be converted in one operation since 1 is the minimum number of operations there can be. So I created the function “are\_strings\_distant” to output if strings were converted by 1 operation or not. After I tested my functions by hardcoding the strings and printing statements within the file.

To test the functions, I gave the strings some team spirit. I tested different lengths and characters of the strings. If edit distance had a minimum operation of 1 the time complexity was O(n). If edit distance had more than one operation the time complexity was O(n^2) due to the strings going into deeper nested iterations in the program.

*Sample runs included:*

String 1 empty

Length of string 1 is less than string 2

Length of string 2 is less than string 2

Characters and lengths are the same

*Sample Runs:*

Strings given: --------------> Miners

Are strings distant: --------> Yes

Edit distance of the strings: 5

Strings given: --------------> Go Team

Are strings distant: --------> No

Edit distance of the strings: 4

Strings given: --------------> Miners Rock

Are strings distant: --------> No

Edit distance of the strings: 4

Strings given: --------------> Miners Miners

Are strings distant: --------> Yes

Edit distance of the strings: 0

In lab 7 I was able to learn how to implement edit distance and find if string 1 could be converted into string 2 with only one edit. Thus, I know how to easily modify the function “ are\_strings\_distant” if we need to find an edit distance of more than 1. I did try to make the program user friendly as much as possible. Ive been trying to learn how to do user inputs which would definitely be a challenge for edit distance.

*Source Code:*

def edit\_distance(val1, val2):

val = [] # where to store the strings

for i in range(val1 + 1):

val.append([0 for i in range(val2 + 1)])

# Create a table for edit distance

for x in range(val1 + 1):

for y in range(val2 + 1):

if x == 0 or y == 0: # Checks if first string or second string is empty

val[x][y] = x

val[x][y] = y

elif s1[x - 1] == s2[y - 1]: # Determines if characters are equal

val[x][y] = val[x - 1][y - 1]

else: # If chars are not equal then find Min resources and take the min value

val[x][y] = 1 + min(val[x][y - 1], val[x - 1][y], val[x - 1][y - 1])

return val[val1][val2] # Returns edit distance of string if any distance is there

def are\_strings\_distant(s1, s2):

# Length of current string

curr1 = len(s1)

curr2 = len(s2)

if (curr1 - curr2) > 1: # Edit distance cannot be more than 1

return "No"

count = 0 # Start count at 0

# Set variables to 0

str1 = 0 # s1

str2 = 0 # s2

while str1 < curr1 and str2 < curr2:

if s1[str1] != s2[str2]:

if count == 1:

return "No" # return false if s1 does not equal to s2

if curr1 > curr2 or curr1 < curr2: # adds 1 to length of string 1 if its greater

str1 += 1 # adds 1 to length of string 2 if its less

str2 += 1

else: # If s1 and s2 are the same strings

str1 += 1

str2 += 1

count += 1 # add 1 to count

else: # Updates both strings if they are the same

str1 += 1

str2 += 1

if str1 < curr1 or str2 < curr2:

count += 1 # increment if strings are either or both less then current

return "Yes" # Return True if strings are distant

# Test if strings are distant and how much edit distance is:

s1, s2 = "Miners", "Miners" # Team Spirit

print("Strings given: -------------->", s1, s2)

print("Are strings distant: -------->", are\_strings\_distant(s1, s2))

print("Edit distance of the strings: ", edit\_distance(len(s1), len(s2)))

“I certify that this project is entirely my own work. I wrote, debugged, and tested the code being presented, performed the experiments, and wrote the report. I also certify that I did not share my code or report or provided inappropriate assistance to any student in the class.”