Homework 9: Dynamic Programming

Due: Friday, April 10 at 11:59 pm on Canvas

Concepts: dynamic programming, principle of optimality, bellman equations

1. (9 points) In this question, you will develop (but not implement) a dynamic programming approach to solve the **minimum edit distance problem**. In the minimum edit distance problem, you are given two strings $s1$ and $s2$ and the goal is to find the minimum number of edits needed to transform $s1$ into $s2$, where a single edit consists of either (a) an insertion of a single character, (b) a deletion of a single character, or (c) a substitution of a single character.

   For example, if $s1$ = 'cake' and $s2$ = 'cat' then the minimum edit distance would be two edits: deleting 'e' and substituting 't' for 'k' (or equivalently deleting 'k' and substituting 't' for 'e').

   (a) (7 points) Give a set of dynamic programming equations to find the minimum edit distance between two strings. Be sure to state what your value function calculates. Then, in 3-5 sentences, argue correctness of your DP solution using the principle of optimality.

   (b) (2 points) What is the runtime of calculating your DP solution?

2. (15 points) In this question, you will see and implement an example of dynamic programming for a problem that does *not* involve optimization. In the **wildcard matching problem**, you are given a pattern string $s1$ and a wildcard string $s2$. While $s1$ is a fixed string of $a-z$ characters, $s2$ may contain one or more wildcard characters $*$ which represent any possible substring (including the empty string). The goal of this problem is to find whether there exists a substitution into the wildcard characters such that the end result yields $s1$.

   For example, if $s1$ = 'lemondrop' and $s2$ = 'l*dr*p*' the answer would be True since we can substitute in 'emon', 'o', and '' for each of the wildcard characters, respectively. However, if $s2$ = 'lem*m*dr*p' the answer would be False because there is no possible wildcard match that would generate $s1$.

   (a) (7 points) Give a set of dynamic programming equations to find whether or not there is a wildcard match between two strings $s1$ and $s2$. Be sure to state what your value function represents. Then, in 3-5 sentences, argue correctness of your DP solution using the principle of optimality. **Hint:** Your value function should evaluate to True or False.

   (b) (2 points) What is the runtime of calculating your DP solution?

   (c) (6 points) Implement and test your dynamic program. For your test function, construct 5-7 test cases that you think contain different possible structures.

   This assignment requires a **course assistant check-off** so be prepared to explain your code structure and to talk through how you constructed your tests.