CS 577 # Study Group – Dynamic Programming Spring 2024

Name: _____ Wisc id: _____

Dynamic Programming Proof Structure

1. **Subproblem definition**
   - Describe the meaning of subproblem in words. Subproblem(s) is often suffix/prefix/substring/combinations. Define all variables you use, and their valid ranges.
2. **Base cases**
   - State solutions for all independent subproblems where Bellman equations break down.
3. **Bellman equation**
   - Relate subproblem solutions recursively
   - Make sure all cases are accounted for
4. **Original problem**
   - Show how to compute solution to the original problem from solutions to subproblem(s).
   - State the order in which you solve the subproblems (e.g., increasing order of $j - i$).
5. **Time and space analysis**
   - Time of original problem = (# subproblems) $\cdot$ (time/problem)
   - Space of original problem = Size of solution matrix + Size of input encoding

Distributing the Treasures of the Iron Bank

The Iron Bank of Braavos, the most powerful financial institution in the known world, is renowned for its immense wealth and influence. After a significant upheaval in the leadership of Braavos, the new rulers decide to distribute some of the Bank's treasures to the two major factions that helped them rise to power: House Lannister and House Tyrell.

Both factions played an equally significant role, and hence, the new leadership wants to ensure that the treasures are divided fairly, reflecting their equal contributions. The treasures include items like Valyrian steel blades, chests filled with golden dragons, and priceless artifacts from distant lands.

You are given a list of n integers, where d_i represents the value of item i. Your task is to help the rulers of Braavos determine if it's possible to split the treasures in such a way that both House Lannister and House Tyrell receive treasures of equivalent value, i.e., both of them receive $\frac{1}{2}\sum_{i=1}^{n} d_i$ value.

The Song of Ice and Fire

Before the Doom, in the heart of Valyria, a grand competition is announced, known as "The Songs of Ice and Fire." Two celebrated bards, one representing Westeros (House Stark's chosen) and the other representing Valyria (House Targaryen's chosen), are to compete in a series of musical duels across the peninsula.

These duels form a best-of-n series, where n is an odd number set by the Dragonlords, influenced by the traditions of Old Valyria. Each duel sees the bards present a song, and the audience votes for their favorite, with each duel being independent of the others.

From whispers in Westeros and Essos, it's known that the Stark's bard has a probability p of winning any given duel based on the resonance of his melodies with the common folk. Conversely, the Targaryen's bard has a probability of $1 - p$, owing to his more opulent and sophisticated tunes.

Construct a 2D array, G, where an entry $G[a, b]$ represents the probability that the Stark's bard has won a duels, while the Targaryen's has won b after $a + b$ duels. Determine the overall probability that the Stark's bard will be crowned the true maestro of The Song of Ice and Fire.

Deciphering the Spider's code

Varys, the Master of Whisperers, has always relied on his intricate network of spies spread across the Seven Kingdoms to gather information. To keep messages discreet, the informants have a unique way of communicating. They encode messages by removing all spaces and converting all the letters to uppercase. For instance, to send the message "Daenerys knows about the plot! Leave the Red Keep." an informant might send "DAENERYSKNOWSABOUTTHEPLOTLEAVETHEREDKEEP".

Petyr Baelish, the Master of Coin, intercepts one such coded message, S, comprising n uppercase letters. Being the clever diplomat that he is, Lord Baelish wants to decipher this message by breaking it into a sequence of valid Common Tongue words. He has access to the grand library of the Red Keep and can quickly determine if any given sequence of letters forms a valid word in the Common Tongue.

Given the string S, help Lord Baelish find an efficient way to break it into a sequence of valid Common Tongue words. You can assume access to a function, $isCTword(s)$, which takes a character string, s, and returns true if and only if s is a recognized word in the Common Tongue. What is the time complexity of your method, given that each call of the function, $isCTword$, executes in $O(1)$ time?