CS 325 Recitation Session 2

Chirag Kaudan

April 2025

1 Introduction

- Survey of how many people want new recitation material vs how many want test prep/HW
- Are people confused about some conceptual stuff?
- You can decide on whether or not brief remarks on last time's recitation are needed: more emphasis on bottom-up iterative as capturing the spirit of DP
- More dynamic programming practice

2 Review from Recitation 1

Principles of DP, ordering subproblems is important

2.1 Ordering Subproblems and D.P DAGs

- Importance of solving subproblems in particular order; associated DAG with every DP problem
- Subproblems correspond to conceptual vertices, dependencies between subproblems correspond to conceptual (directed) edges
- Facts about DAGs (they might not be familiar with basic properties of DAGs)
- Talk about longest paths in DAGs, relationship to LIS

2.2 Subproblem Tips

There is real subtlety in designing dynamic programming algorithms. Sometimes it helps conceptually to think about a recursive formula and then find the set of subproblems that are required to unravel that recurrence, but other times it helps more to first think about a natural set of subproblems and then find a recurrence between them. Fortunately, many of the dynamic programming problems you'll see in this course do not get so much more trickier than this, so this worked through example and informal guidelines should serve you well!

3 String Decoding

Basic Setup: You are a spy working for an intelligence agency. Your agency uses an encoding scheme to send secret messages so that rival agencies cannot intercept and understand your communications easily. The secret messages are strings of digits $(0-9)$ that represent a word in English. As a diligent spy, you have already memorized the encoding scheme used to translate digits to English letters for the day. You want to figure out how many possible ways there are to decode a given message.

Problem Statement: You are given a secret message as a string $s_1 s_2 \dots s_n$ of length n where each $s_i \in \{0, 1, 2, \dots, 9\}$. You also have the encoding scheme used to translate digits to English letters. It is the following map called f:

$$1 \to \text{'A'}, 2 \to \text{'B'}, \dots, 26 \to \text{'Z'}$$

Compute how many possible ways there are to decode this message.

3.1 Example of Setup/Problem

Example string: $1232 \to ABCB$ or LCB or AWB so the answer is 3
DP table for this (later): $[1, 1, 2, 3, 3]$
Example string: $11264 \to AABFD$ or $KBFD$ or $AGFD$ or KZD or $AAZD$ so the answer is 5
DP table for this (later): $[1, 1, 2, 3, 5, 5]$

3.2 Algorithm

Input : A string $s = s_1 s_2 \dots s_n$ of length n with $s_i \in \{0, 1, \dots 9\}$ for all integers $1 \leq i \leq n$ and the map f

```
if n ≤ 0 then
    return 0
Let dp be an array of length n + 1
Initialize dp to contain 0 in all entries
Set dp[0] and dp[1] to 1
for i from 1 to n do
    if s[i] ≠ 0 then
        dp[i] = dp[i − 1]
    if 10 ≤ s[i − 1] concatenated with s[i] ≤ 26 then
        dp[i] = dp[i] + dp[i − 2]
return dp[n]
```

We are assuming the concatenation of two symbols s_x and s_y is equivalent (numerically) to the two digit decimal number $s_x s_y$, with $s_x s_y = s_y$ if $s_x = 0$. Additionally, let s_x be the empty string ϵ if $x \leq 0$.

- Discuss similarities with stair climbing/Fibonacci problem, worst case
- Would this be an example of when top down memoization is more useful?