Dynamic Programming

Madiba Hudson-Quansah

Contents

Chapter 1

Introduction

Definition 1.0.1: Dynamic Programming

A programming technique that involves breaking down a problem into smaller subproblems and solving each subproblem only once. The solutions to the subproblems are stored in a table, so that the solutions to the subproblems are not recomputed. This technique is used to solve optimization problems /A general algorithm design technique for solving problems defined by recurrences with overlapping subproblems, i.e.:

$$T(n) = T(n-1) + T(n-2)$$

Where the problem $T(n)$ is made up of the subproblems $T(n-1)$ and $T(n-2)$ and $T(n-1)$ is made up of the subproblems $T(n-2)$ and $T(n-3)$ and so on.

A Dynamic Programming problem is made up of two main components:

- Overlapping Subproblems - Where the problem can be broken down into smaller subproblems that are solved multiple time during the process.
- Optimal Substructure - Where an optimal solution of the original problem can be constructed from the optimal solutions of its subproblems.

There are two main approaches to solving a Dynamic Programming problem:

- Memoization - Top Down Approach
- Tabulation - Bottom Up Approach

Question 1

Given a row of coins of the following values: $7, 5, 2, 10, 6, 3, 4, 8, 1$, pick up coins with a maximum value subject to no adjacent coins can be picked. Produce the optimal solutions in terms of F array with a linear algorithm.

Solution: Given that there are two choices to make when picking coins, skipping the last coin and picking all the others or picking the last coin and skipping the second last coin. The max value of the problem becomes:

$$F(n) = \max\left(C_1 + F(n-2)\ , F(n-1)\right)\ , n > 1$$

Where C is the array of coins.

$$F(i,j) =$$