Dynamic Programming

Those who cannot remember the past are condemned to repeat it

TABLE OF CONTENTS

01

Algorithms

It’s like a recipe

What are algorithms?

An algorithm is a process used to conduct out a computation or solve a problem. In either hardware-based or software-based routines, algorithms function as a detailed sequence of instructions that carry out predetermined operations sequentially.

02

Evaluate an algorithm

Control the laws of time and space

Evaluate an algorithm

- Running time (time complexity)
- Memory space (space complexity)

There are many other metrics but we will focus on these two

Time complexity

- The amount of time it takes an algorithm to run as a function of the input is referred to as its time complexity.
- The big O notation is used to indicate the time complexity of algorithms.

Time complexity

- Constant : o(1)
- Logarithmic : o(log(n))
- Linear : o(n)
- Quadratic : o(n^2)
- Exponentiel : O(a^n)

03

Dynamic programming

Fancy way for saying recursion with cache… kinda

Dynamic programming

An algorithmic technique is used to solve problems more efficiently by dividing them down into simpler subproblems.

Dynamic programming

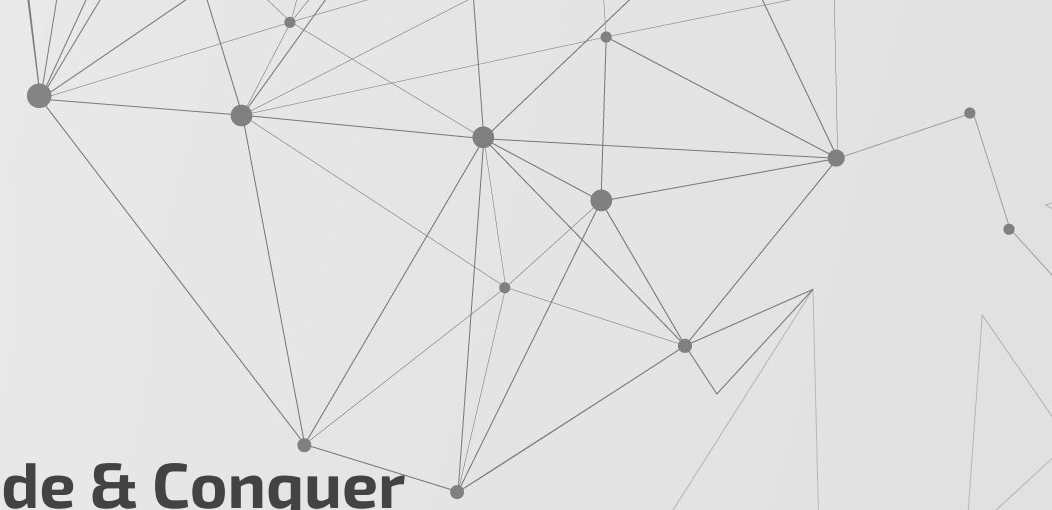
Dynamic programming has two major components:

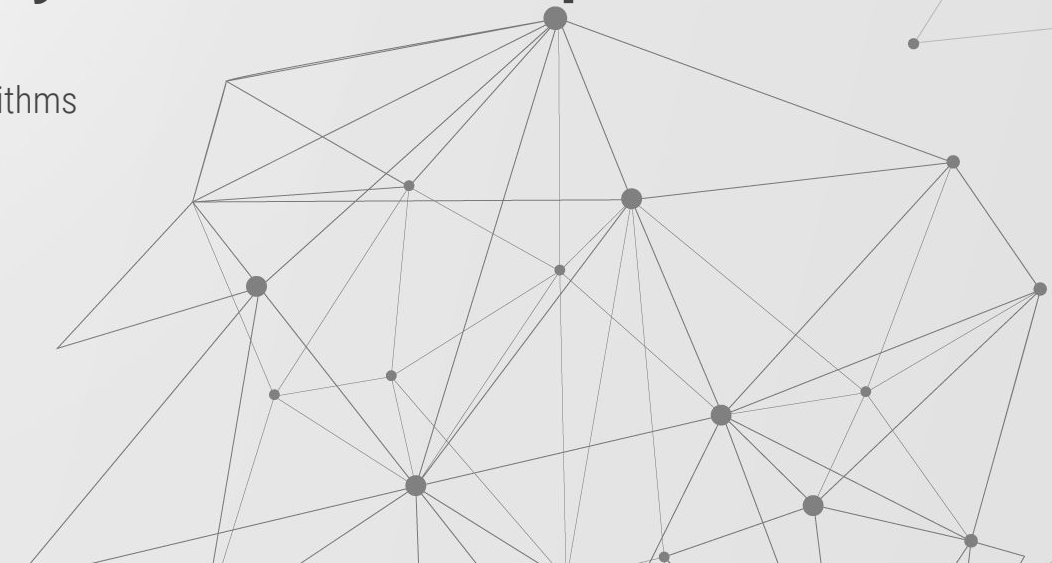
- Overlapping subproblems
- Optimal structure

04

DP vs Greedy vs Divide & Conquer

The battle of leet algorithms





Dynamic programming - Divide and conquer

- They both divide problem into subproblems
- Absence of Overlapping subproblems in the second onde
- DP used the saved results to build the future one, while divide and conquer combine results

Dynamic programming - Greedy algorithms

- Greedy algorithms takes the best local solution whereas DP takes the intermediate result and use it in the following operations

05

DP approaches

Some prefer the upper view, and others enjoy climbing their way up

Dynamic programming - Top-down

Start from the beginning and save the intermediate results, if we come across a subproblem that we have already solved, we will utilize the previously stored solution.

Dynamic programming - bottom-up

We begin at the bottom (case 0), then construct our solution till we accomplish our goal.

06

Practice

Enough boring theory