CS3230 Semester 2 2024/2025
Design and Analysis of Algorithms

Tutorial 07
Dynamic Programming
For Week 08

Document is last modified on: February 19, 2025

1 Lecture Review: Dynamic Programming

The key ideas to solve a problem with Dynamic Programming (DP) are as follows:

- Optimal substructure: Can we express the solution recursively?
  Break the original problem into its subproblems.
- Realizes that there are only a small (maybe polynomial) number of subproblems.
  The naive implementation of the recursive solution encounters many overlapping subproblems.
  The recursive algorithm may take exponential time (solving the same subproblem many times).

So we either:

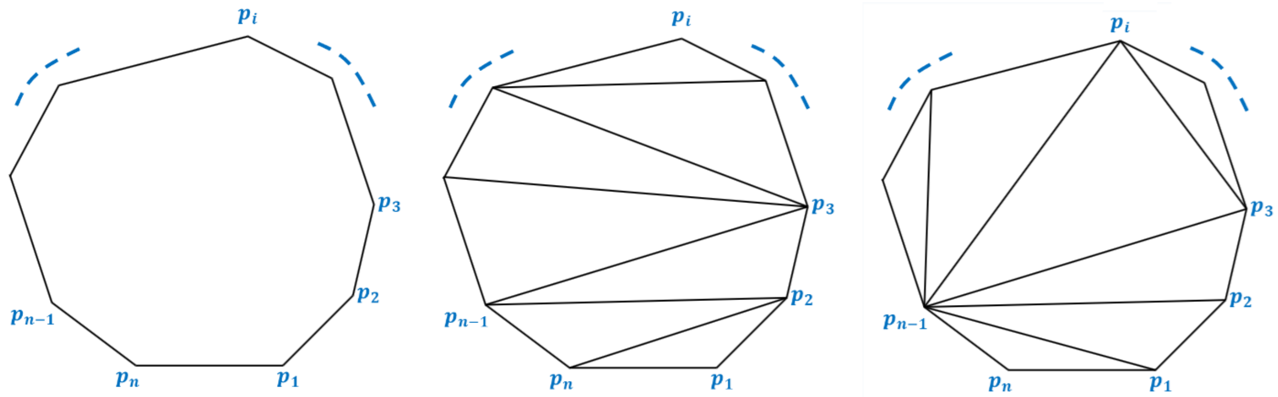
- Top-down: Compute the recursive solution but memoize the solutions of the computed subproblems, so the next computation of the same subproblem can be done in $O(1)$.
- Bottom-up: Compute the recursive solution iteratively in a bottom-up fashion (also called tabulation), starting from the base cases and continue filling the next subproblems that we can compute next, gradually.

Both methods avoids wastage of computation and leads to an efficient implementation.

2 Tutorial 07 Questions

This tutorial is related to the **Convex Polygon Triangulation** problem: Given a convex polygon with n ($n \geq 2$) vertices (labeled with $1, 2, \ldots, n$), divide (or triangulate) the polygon into $n - 2$

triangles. We can triangulate a convex polygon in many ways. The figure below shows 2 ways (middle and right pictures).



A triangle consisting of vertices (x, y, z) will have a weight of $W(x, y, z)$ – for the purpose of this problem, treat W as a black-box $O(1)$ function. Our objective is to minimize the sum of the weights of $n - 2$ triangles in the optimal triangulation!

Q1). Let $TRI(x, y)$ be a function to triangulate a polygon with minimum weight sum, but we only consider the vertices in the range of $(x, x + 1, x + 2, \ldots, y)$. So our problem can be solved by calling $TRI(1, n)$. Your first task is to write a recursive formula of $TRI(x, y)$.

(a) Find the base case of $TRI(x, y)$

(b) Find the recursive case of $TRI(x, y)$

Hint: It calls $TRI(x', y')$ where $x < x'$ or $y' < y$.

Q2). What is the time complexity of this recursive formula $TRI(1, n)$, if implemented verbatim.

(a) $O(n^2)$

(b) $O(n^3)$

(c) $O(3^n)$

Q3). Which one is the correct explanation regarding the findings from (Q2)?

(a) It has 3^n non-overlapping subproblems and each call runs in $\Theta(1)$

(b) It has n^2 non-overlapping subproblems and each call runs in $\Theta(\frac{3^n}{n^2})$

(c) It has n^2 subproblems but there are many overlaps

Q4). Design a Dynamic Programming (DP) solution for **Convex Polygon Triangulation** problem.

(a) Using Top-Down DP

(b) Using Bottom-Up DP