Assignment-2

Software Systems Lab

Shashank P

IIT Dharwad

https://www.iitdh.ac.in/

August 11, 2021

Dynamic Programming

- Characteristics of Dynamic Programming
  1. *Overlapping Sub-problems*

1

Subproblems are smaller versions of the original problem. Any problem has overlapping sub-problems if finding its solution involves solving the same subproblem multiple times.

  2. *Optimal Substructure*

2

Any problem has optimal substructure property if its overall optimal solution can be constructed from the optimal solutions of its subproblems.

DP Methods

- **Top-down with Memoization**

1

In this approach, we try to solve the bigger problem by recursively finding the solution to smaller sub-problems. Whenever we solve a sub-problem, we cache its result so that we don't end up solving it repeatedly if it's called multiple times. Instead, we can just return the saved result.

DP Methods

- **Top-down with Memoization**

1

In this approach, we try to solve the bigger problem by recursively finding the solution to smaller sub-problems. Whenever we solve a sub-problem, we cache its result so that we don't end up solving it repeatedly if it's called multiple times. Instead, we can just return the saved result.

- **Bottom-up with Tabulation**

2

Tabulation is the opposite of the top-down approach and avoids recursion. In this approach, we solve the problem "bottom-up" (i.e. by solving all the related sub-problems first).

Algorithms

- Divide and conquer
- Greedy Algorithm
- Dynamic Programming

Algorithms

- Divide and conquer
- Greedy Algorithm
- Dynamic Programming

Algorithms

- Divide and conquer
- Greedy Algorithm
- Dynamic Programming

Example:

Quick-Sort: The average case run time of quick sort is $O(n * log\, n)$.**This case happens when we don't exactly get evenly balanced partitions.**

Example:

Merge-Sort: The time complexity of Merge Sort is $O(n * log\, n)$.
Merge Sort is useful for sorting linked lists in $O(n * log\, n)$ time.

Hyperlinks

- Divide and Conquer
- Greedy Algorithm
- Dynamic Programming

List of Data Structures

- Primitive
- Non-Primitive

  - *Linear*
    - **Static**
      1. Array
    - **Dynamic**
      1. Linked List
      2. Stack
      3. Queue
  - *Non-Linear*
    1. Tree
    2. Graph

List of Data Structures

- Primitive
- Non-Primitive

  - *Linear*
    - **Static**
      1. Array
    - **Dynamic**
      1. Linked List
      2. Stack
      3. Queue
  - *Non-Linear*
    1. Tree
    2. Graph

List of Data Structures

- Primitive
- Non-Primitive

  - *Linear*
    - **Static**
      1. Array
    - **Dynamic**
      1. Linked List
      2. Stack
      3. Queue
  - *Non-Linear*
    1. Tree
    2. Graph

List of Data Structures

- Primitive
- Non-Primitive

  - *Linear*
    - **Static**
      1. Array
    - **Dynamic**
      1. Linked List
      2. Stack
      3. Queue
  - *Non-Linear*
    1. Tree
    2. Graph

List of Data Structures

- Primitive
- Non-Primitive

    - *Linear*
        - **Static**
            1. Array
        - **Dynamic**
            1. Linked List
            2. Stack
            3. Queue
    - *Non-Linear*
        1. Tree
        2. Graph

List of Data Structures

- Primitive
- Non-Primitive

  - *Linear*
    - **Static**
      1. Array
    - **Dynamic**
      1. Linked List
      2. Stack
      3. Queue
  - *Non-Linear*
    1. Tree
    2. Graph

Data Structures

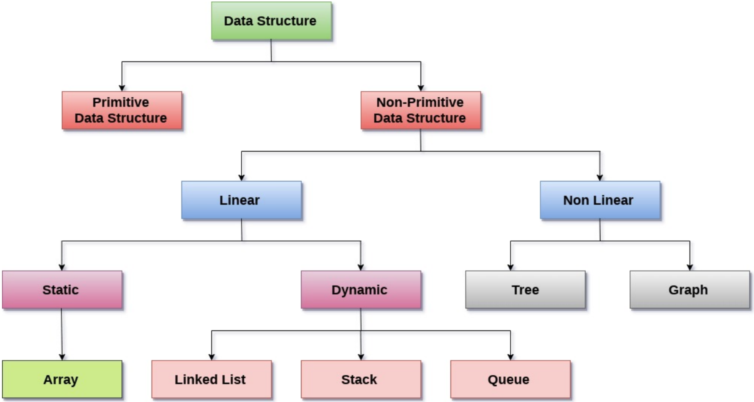

Figure: 1

Algorithm	Best Case	Average Case	Worst Case
Linear Search	$O(1)$	$O(n)$	$O(n)$
Binary Search	$O(1)$	$O(log\,n)$	$O(log\,n)$
Bubble sort	$O(n)$	$O(n^2)$	$O(n^2)$
Selection sort	$O(n^2)$	$O(n^2)$	$O(n^2)$

Table: 1

Theorem (Trigonometric Identity)

$Sin^2\theta + Cos^2\theta = 1$

Theorem

Let a, b, c be lengths of right angled triangle.
By definition

$$sin\theta = b/c\left(\frac{oppositeside}{hypotenuse}\right)$$

$$cos\theta = a/c\left(\frac{adjacentside}{hypotenuse}\right)$$

$sin^2\theta + cos^2\theta = \frac{b^2}{c^2} + \frac{a^2}{c^2} = \frac{a^2+b^2}{c^2}$

From Pythagoras theorem

$c^2 = a^2 + b^2$

$\frac{a^2+b^2}{c^2} = 1 \implies sin^2\theta + cos^2\theta = 1$

Hence Proved.

Multi-line equations

$$
\begin{aligned}
f(x) = x^6 + 7x^3y + 50x^3y^2 + 12x^2y^4 \\
- 19x^5y^4 - 10x^7y^6 + 7y^6 - m^3n^3
\end{aligned}
$$

$$
\begin{aligned}
\rho\Delta x\Delta y\Delta z\Delta\tau\partial_t c_i(t, x, \tau) &= \rho\Delta x\Delta y\Delta z\Delta\tau(p_i - d_i) \\
&\quad - \rho\Delta y, \Delta z\Delta\tau[q_{i,x}(t, x + \Delta x/2, y, z, \tau) \\
&\qquad - q_{i,x}(t, x - \Delta x/2, y, z, \tau)] \\
&\quad - \rho\Delta x, \Delta z\Delta\tau[q_{i,y}(t, x, y + \Delta y/2, y, z, \tau) \\
&\qquad - q_{i,y}(t, x, y - \Delta y/2, z, z, \tau)] \\
&\quad - \rho\Delta x\Delta y\Delta\tau[q_{i,z}(t, x, y, z + \Delta z/2, \tau) \\
&\qquad - q_{i,z}(t, x, y, z - \Delta z/2, \tau)]
\end{aligned}
$$