ALGORITHM ANALYSIS

Divide & Conquer

A fundamental paradigm for solving complex problems by breaking them down.

SUBMITTED BY

Syed Muzammil

Roll No: 2316

EVALUATOR

Prof Khalid Asad

Faculty of CS

SUBJECT

Analysis of Algorithm

5th Semester

Learning Objectives

By the end of this presentation, you will be able to:

Understand the Concept
Grasp the 3-step process of Divide, Conquer, and Combine

Identify Algorithms
Recognize when to apply Divide & Conquer approach

Analyze Examples
Understand Merge Sort, Quick Sort & Binary Search

Real-world Applications
See how D&C solves practical problems

THE GOAL

"Divide complex problems into simpler ones, solve them, and combine the solutions."

The Core Concept

1. Divide

Break the original problem into a set of sub-problems that are smaller instances of the same problem.

2. Conquer

Solve the sub-problems recursively. If they are small enough (Base Case), solve them directly.

3. Combine

Combine the solutions of the sub-problems to create a solution for the original problem.

Why Use Divide & Conquer?

This approach offers several advantages that make it powerful for algorithm design

Efficiency

Reduces time complexity from $O(n^2)$ to $O(n \log n)$ in many cases

Parallelism

Subproblems can be solved independently on different processors

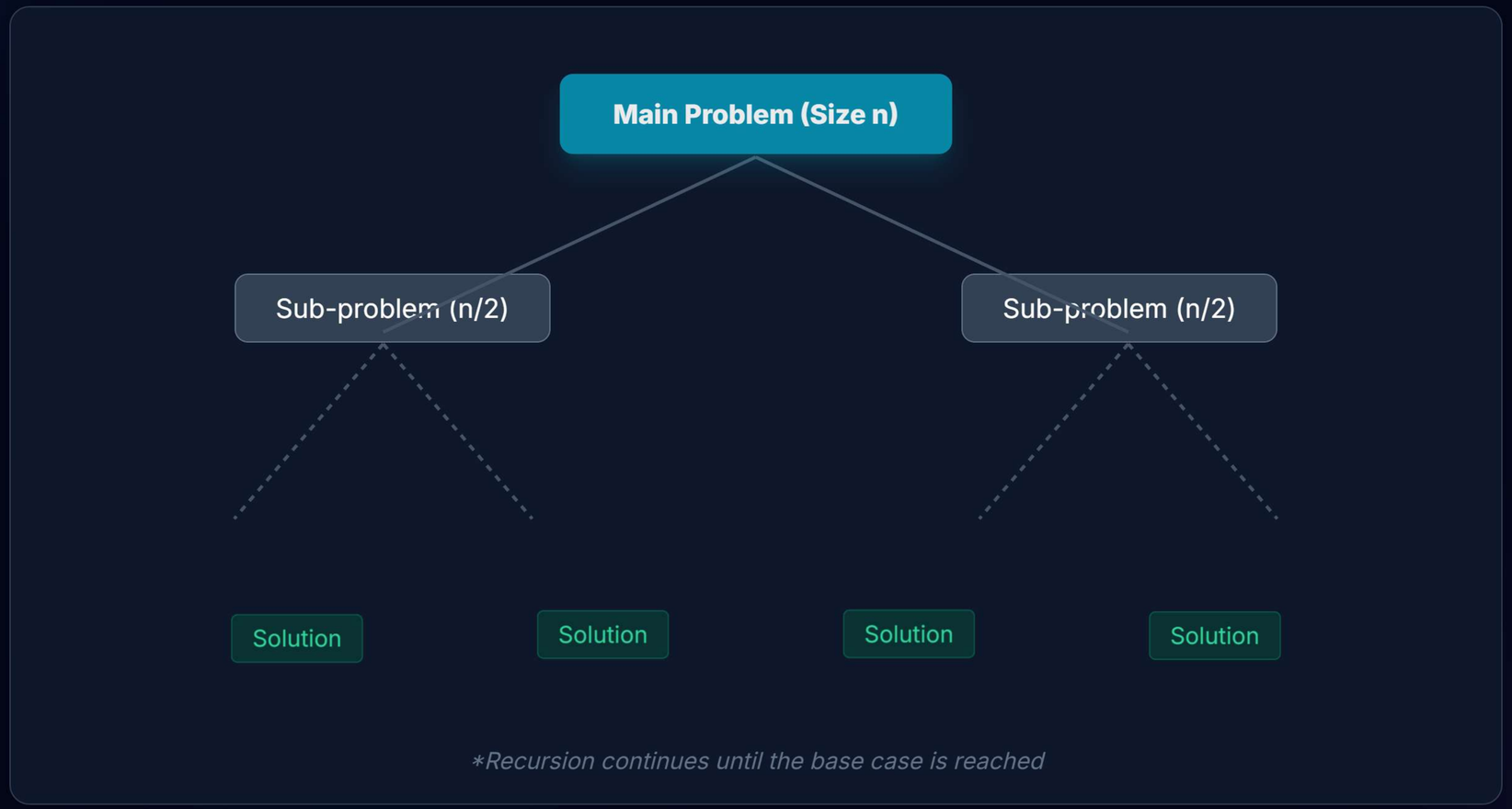
Simplicity

Breaks complex problems into manageable, understandable pieces

When to Use?

When the problem can be divided into independent sub-problems, and the solutions can be combined efficiently. Perfect for sorting, searching, and optimization problems.

Visualizing the Flow



*Recursion continues until the base case is reached

Merge Sort Example

Divide into halves

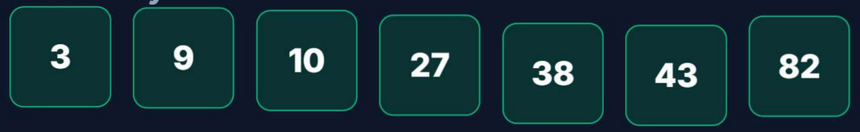
Original Array

Divide
Split array into halves

Sort
Recursively sort halves

Merge
Merge sorted halves

Sorted Array



How Merge Sort Works

- Divide array into two equal halves
- Recursively sort each half
- Merge sorted halves into final array

Complexity Analysis

Time Complexity

O(n log n) — Efficient

Space Complexity

O(n) — Moderate

⚡ Quick Sort Example

How It Works

1	2	3
Pick Pivot	**Partition**	**Recurse**
Choose a pivot element from array	Move smaller left, larger right	Sort left & right recursively

Original Array

Pivot Selected: 6

After Partition

● ≤ Pivot ● Pivot ● > Pivot

Binary Search Example

Searching for:

23

Sorted Array (Prerequisite for Binary Search)

1	5	7	12	15	18	23	23	30	35

Step 2: Check middle element at index 8

Current Position
Index: 8

Comparison
Target == Array[8]

Search Process

Found!

Target found at index 8

Binary Search Complexity

Best Case:	**Ω(1)**
Worst Case:	**O(log n)**

Common Algorithms Comparison

Merge Sort

Stable Sort

Divides array into halves, sorts recursively, then merges

Time:	O(n log n)
Space:	O(n)
Stable:	Yes

Quick Sort

In-place Sort

Picks pivot, partitions array, sorts recursively

Best Time:	Ω(n log n)
Worst Time:	O(n²)
In-place:	Yes

Binary Search

Decrease & Conquer

Searches sorted array by halving search space

Best Case:	Ω(1)
Worst Case:	O(log n)
Requires:	Sorted Input

Choosing the Right Algorithm

Merge Sort for stable sorting, **Quick Sort** for in-place efficient sorting, and **Binary Search** for fast searching in sorted data.

Real World Applications

MapReduce

Google's framework for processing massive datasets across clusters.

Strassen's Algorithm

Fast matrix multiplication for graphics and scientific computing.

FFT

Fast Fourier Transform processes signals in O(n log n) time.

Database Sorting

External Merge Sort handles data larger than RAM.

Game AI

Minimax algorithm for decision making in chess and other games.

Closest Pair

Finding nearest points in GPS navigation and mapping.

Version Control

Git merge uses 3-way merge algorithm (D&C based).

Image Processing

JPEG compression uses Discrete Cosine Transform.

In Summary

Divide and Conquer is not just an algorithm; it's a philosophy. It enables parallel processing, efficient cache usage, and solves some of the hardest problems in Computer Science with elegant recursion.

FINAL THOUGHT

"To understand the complex, divide it into the simple."

Presentation for Analysis of Algorithm
Roll No: 2316 | 2026

Thank You!

We hope this presentation helped you understand the Divide and Conquer paradigm.

Questions?

Feel free to ask any questions about Divide and Conquer algorithms!

Created with ❤️ By @MrSyedMuzammil

Syed Muzammil | Roll No: 2316 | Evaluator : Prof Khalid Asad