

# Divide and Conquer Strategy Overview

**Divide and Conquer** is a strategy for solving computational problems by breaking large problems into smaller subproblems, solving them recursively, and combining results.

This approach belongs to algorithm design paradigms alongside **greedy method**, **dynamic programming**, **backtracking**, and **branch and bound**.

---

## Core Principles of Divide and Conquer


### 1. Problem Decomposition

- Given a large problem **P** of size **n**, **divide** it into **k smaller subproblems** ( $P_1, P_2, \dots, P_k$ ).
- Each subproblem is smaller than the original problem.

### 2. Recursive Solving

- **Solve** each subproblem individually.
- If a subproblem is still large, **apply the same strategy recursively** (divide into sub-subproblems).

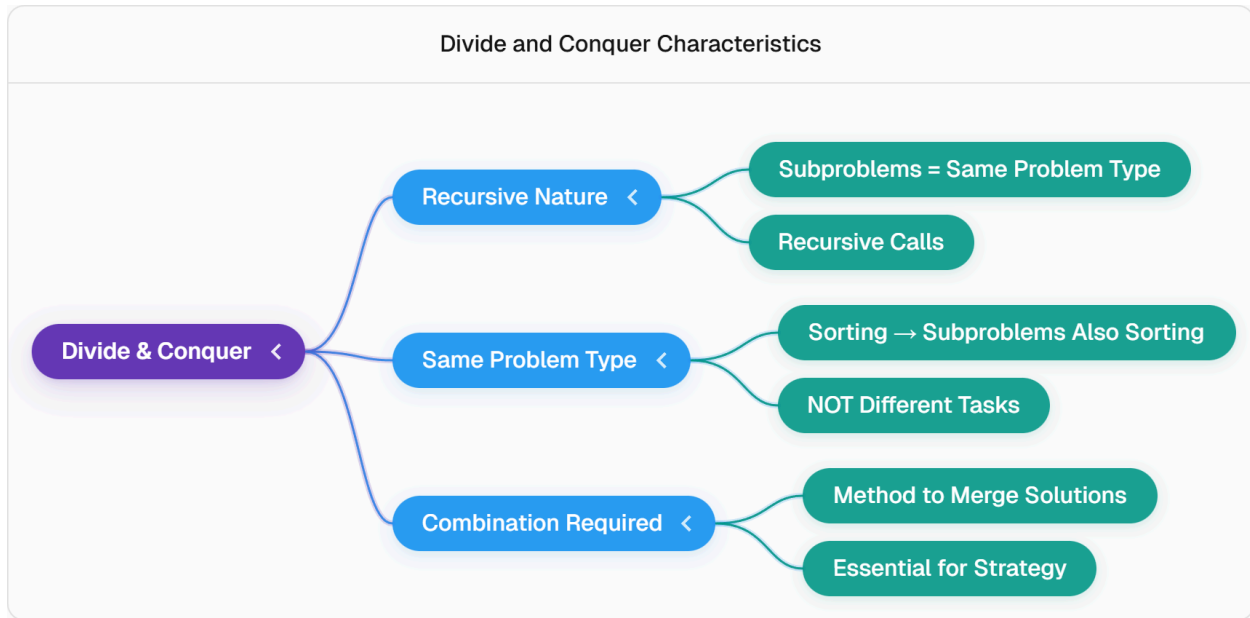
### 3. Solution Combination

- **Combine** solutions of all subproblems to obtain the solution for the original problem **P**.
  -  **Critical:** Must have a method to combine subproblem solutions, otherwise strategy cannot be applied.
- 

## Key Characteristics

**Subproblems must be the SAME TYPE as the original problem.**

Example: If main problem is sorting, subproblems must also be sorting problems (not invitation preparation, poster design, etc.).

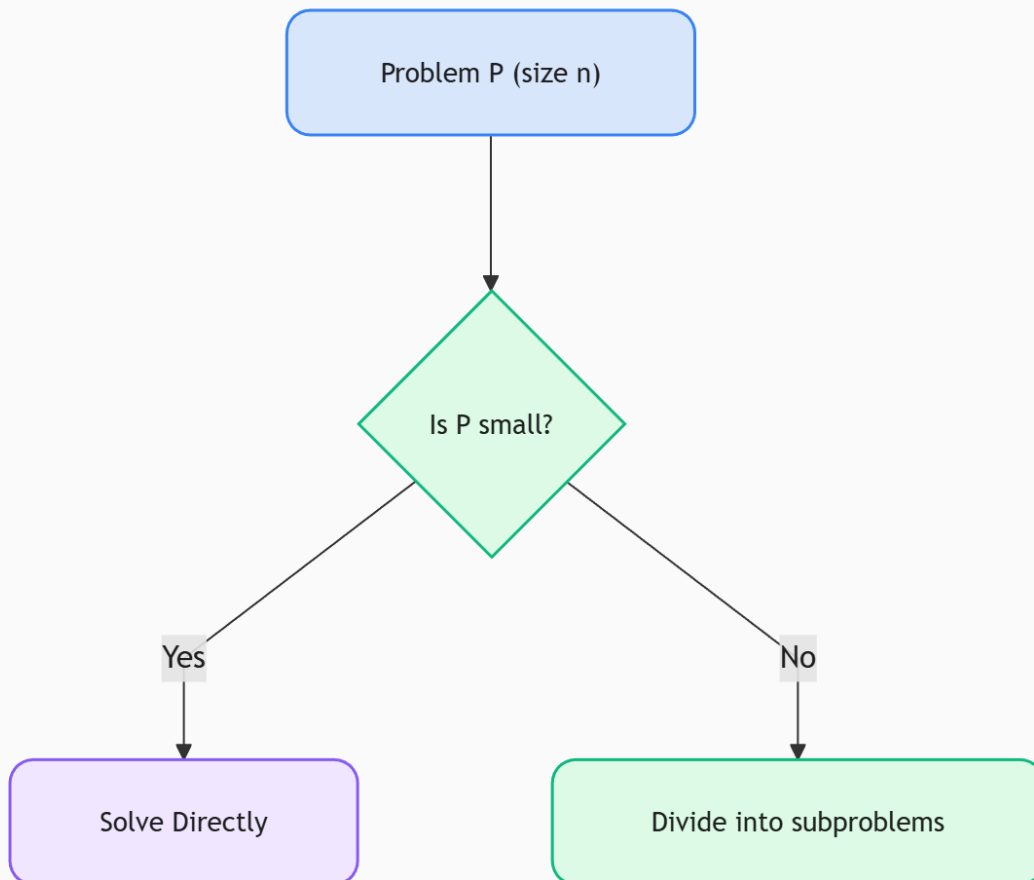


## General Algorithm Structure

### Step-by-Step Process

1. **Base Case:** If problem **P** is **small**, solve **directly**.
2. **Divide Step:** Break **P** into **k subproblems**  $P_1, P_2, \dots, P_k$ .
3. **Conquer Step:** **Recursively** apply Divide and Conquer to each subproblem.
4. **Combine Step:** Merge subproblem solutions to get **Solution(P)**.

## Divide and Conquer General Method



## Guidelines for Applicability

Apply Divide and Conquer when:

✓ YES	✗ NO
Subproblems are <b>same type</b> as original	Subproblems are <b>different tasks</b>
<b>Combination method exists</b>	No way to <b>merge solutions</b>
Problem exhibits <b>recursive structure</b>	Problem doesn't naturally divide

# Problems Covered in This Topic

## Classic Divide and Conquer Algorithms:

- Binary Search
  - Finding Maximum and Minimum
  - Merge Sort
  - Quick Sort
  - Strassen's Matrix Multiplication
- 

## Analysis Requirements

Since Divide and Conquer algorithms are **recursive**, you must master:

1. **Writing recursive functions**
2. **Analyzing recursive algorithms**
3. **Solving recurrence relations** (time complexity)

**Next Topic:** Recurrence Relations (multiple videos coming).