Design and Analysis of Algorithms

This repository contains the code for the assignments of the course Design and Analysis of Algorithms (19Z402) from PSG College of Technology, Coimbatore.

Syllabus

DIVIDE AND CONQUER :

- Introduction to Algorithm Design techniques
- Divide and Conquer Methodology
- Solving recurrence relations
- Masters Theorem
- Finding Maximum and Minimum Element
- Quick sort
- Merge sort
- Convex Hull

GREEDY METHOD:

- Greedy Strategy
- Knapsack Problem
- Minimum Spanning Trees
- Single Source Shortest Path Method
- Huffman Trees

DYNAMIC PROGRAMMING :

- Principle of Optimality
- Knapsack Problem
- All Pairs Shortest Path
- Optimal Binary Search Tree
- Multistage Graphs

BACKTRACKING:

- State Space Tree
- Knapsack Problem
- The Eight Queens Problem
- Sum of subsets
- Graph Coloring

BRANCH AND BOUND :

- Bounding Functions
- 0/1 Knapsack Problem

- Traveling SalesPerson Problem
- Assignment Problem

Divide and Conquer Algorithm

Divide and Conquer Algorithm breaks a problem into subproblems that are similar to the original problem, recursively solves the subproblems. Each recursion of the algorithm makes the problem smaller until it reaches a base case. The algorithm is split into three parts :

- Divide: This divides each problem into smaller problems allowing us to make the number of elements to be calcuated on smaller.
- Conquer: This allows us to perform the required operation on the elements and solve the subproblems by addressing the base case.
- Combine: This is where we recombine all the parts of the problem to find our final result.

Merge Sort

Merge Sort is based on the divide and conquer approach. It divides the input array into two halves, calls itself for the two halves, and then merges the two sorted halves. The merge() function is used for merging two halves. The merge(arr, l, m, r) is a key process that assumes that arr[l..m] and arr[m+1..r] are sorted and merges the two sorted sub-arrays into one.

Time Analysis:
Merge Sort Function

- Best Case: O(n log n)
- Average Case: O(n log n)
- Worst Case: O(n log n)

Merge Function

- Best Case: O(n)
- Average Case: O(n)
- Worst Case: O(n)

Quick Sort

Quick Sort is based on the divide and conquer approach. It picks an element as pivot and partitions the given array around the picked pivot. There are many different versions of quickSort that pick pivot in different ways. The key process in quickSort is partition(). Target of partitions is, given an array and an element x of array as pivot, put x at its correct position in sorted array and put all smaller elements (smaller than x) before x, and put all greater elements (greater than x) after x. All this should be done in linear time.

Time Analysis:

- Best Case: O(n log n)
- Average Case: O(n log n)

- Worst Case: O(n^2)