CSE408:DESIGN AND ANALYSIS OF ALGORITHMS

L:3 T:0 P:2 Credits:4

Course Outcomes: Through this course students should be able to

CO1 :: Analyze the efficiency of algorithms using complexity analysis, recurrence solving techniques, and Divide-and-Conquer strategies.

CO2 :: Apply string matching algorithms, Trie data structures, and computational geometry techniques to solve pattern matching and geometric problems.

CO3 :: Apply Dynamic Programming techniques to solve optimization and combinatorial problems.

CO4 :: Apply Graph algorithms, Network Optimization, and Greedy techniques to solve graph-based optimization problems.

CO5 :: Apply Transform-and-Conquer and advanced algorithmic techniques to solve searching, sorting, and sequence-processing problems efficiently.

CO6 :: Evaluate the suitability of Backtracking, Branch and Bound, Approximation Algorithms, and Complexity Classes for solving computationally hard problems.

Unit I

Analysis of Algorithms and Complexity Measures : Time and Space Complexity, Complexity Analysis of Insertion Sort and Merge Sort, Analysis of Iterative Algorithms, Analysis of Recursive Algorithms: Substitution Method, Recursion Tree Method and Master Method

Divide-and-Conquer Technique : Strassen's Matrix Multiplication, Order Statistics: Quick select, k-th Smallest and k-th Largest Element

Unit II

String Matching Algorithms and Computational Geometry : Sequential Search and Brute-Force String Matching, Naive Pattern Matching, Rabin-Karp Algorithm, Knuth-Morris-Pratt Algorithm, Data Structures for String Processing: Trie (Prefix Tree), Computational Geometry: Closest-Pair Problem, Convex Hull

Unit III

Dynamic Programming : Introduction to Dynamic Programming, Computing a Binomial Coefficient, Memory Functions, Knapsack Problem, Matrix-Chain Multiplication, Longest Common Subsequence, Optimal Binary Search Trees

Unit IV

Graph Algorithms and Network Optimization : Introduction to Graphs, Breadth-First Search (BFS), Depth-First Search (DFS), Topological Sort

Greedy Technique : Introduction to Greedy approach, Knapsack Problem, Minimum Spanning Trees, Prim's Algorithm, Kruskal's Algorithm, Single-Source Shortest Paths, All-Pairs Shortest Paths

Unit V

Transform-and-Conquer Techniques : Balanced Search Trees, Counting Sort, Radix Sort, Bucket Sort

Advanced Algorithmic Techniques : Monotonic Stack, Monotonic Queue, Two pointers Technique

Unit VI

Backtracking, Approximation and Complexity Classes : n-Queens Problem, Hamiltonian Circuit, Subset-Sum Problem, Branch and Bound: Assignment Problem, Knapsack Problem, Traveling Salesman Problem, Approximation Algorithms: Vertex-Cover, Set-Covering, Bin Packing Problems, Complexity Classes: Concepts of P, NP, NP-Hard, NP-Complete Problems

List of Practicals / Experiments:

List Of Practical

- 1. Implement Insertion Sort algorithm and analyze its time complexity.
- 2. Implement Merge Sort algorithm and analyze its time complexity.
- 3. Implement Strassen's Matrix Multiplication algorithm.
- 4. Implement the Quick Select algorithm to find the k-th smallest and k-th largest element.

- 5. Implement the Naïve String Matching algorithm.
- 6. Implement the Rabin-Karp algorithm for string matching using hashing.
- 7. Implement the Knuth-Morris-Pratt (KMP) algorithm for efficient string matching.
- 8. Implement a Trie (Prefix Tree) for insertion and searching of strings.
- 9. Implement the Convex Hull algorithm using Graham Scan
- 10. Implement the Longest Common Subsequence (LCS) algorithm using Dynamic Programming.
- 11. Implement Matrix Chain Multiplication using Dynamic Programming.
- 12. Implement the 0/1 Knapsack problem using Dynamic Programming.
- 13. Implement Breadth-First Search (BFS) and Depth-First Search (DFS) graph traversal algorithms
- 14. Implement Prim's algorithm to find the Minimum Spanning Tree of a weighted graph.
- 15. Implement Kruskal's algorithm to find the Minimum Spanning Tree of a weighted graph.
- 16. Implement Dijkstra's algorithm to find the Single-Source Shortest Path in a weighted graph
- 17. Implement the Largest Rectangle in Histogram problem using the Monotonic Stack technique.
- 18. Implement the Longest Palindromic Substring problem using the Two-Pointer technique.
- 19. Implement the N-Queens problem using the Backtracking technique.
- 20. Implement the Traveling Salesman Problem using the Branch and Bound technique.

Text Books:

1. INTRODUCTION TO THE DESIGN AND ANALYSIS OF ALGORITHM by ANANY LEVITIN, PEARSON

References:

1. INTRODUCTION TO ALGORITHMS by C.E. LEISERSON, R.L. RIVEST AND C. STEIN, THOMAS TELFORD LTD.