CS2202 DESIGN AND ANALYSIS OF ALGORITHMS

Course Objectives:

Upon completion of this course, students will be able to do the following:

- Analyse the asymptotic performance of algorithms.
- Write rigorous correctness proofs for algorithms.
- Demonstrate a familiarity with major algorithms and data structures.
- Apply important algorithmic design paradigms and methods of analysis.
- Synthesize efficient algorithms in common engineering design situations

Course Outcomes:

Students who complete the course will have demonstrated the ability to do the following:

- Argue the correctness of algorithms using inductive proofs and invariants.
- Analysis worst-case running times of algorithms using asymptotic analysis.
- Describe the divide-and-conquer paradigm and explain when an algorithmic design situation calls for it. Recite algorithms that employ this paradigm. Synthesize divide-and conquer algorithms. Derive and solve recurrences describing the performance of divide and-conquer algorithms.
- Describe the dynamic-programming paradigm and explain when an algorithmic design situation calls for it. Recite algorithms that employ this paradigm. Synthesize dynamic programming algorithms, and analysis them.
- Describe the greedy paradigm and explain when an algorithmic design situation calls for it. Recite algorithms that employ this paradigm. Synthesize greedy algorithms, and analysis them.

SYLLABUS

Introduction: What is an Algorithm, Algorithm Specification, Pseudocode Conventions Recursive Algorithm, Performance Analysis, Space Complexity, Time Complexity, Amortized Complexity, Amortized Complexity, Asymptotic Notation, Practical Complexities, Performance Measurement.

Divide and Conquer: General Method, Defective Chessboard, Binary Search, Finding the Maximum and Minimum, Merge Sort, Quick Sort, Performance Measurement, Randomized Sorting Algorithms.

The Greedy Method: The General Method, Knapsack Problem, Job Sequencing with Deadlines, Minimum-cost Spanning Trees, Prim's Algorithm, Kruskal's Algorithms, An Optimal Randomized Algorithm, Optimal Merge Patterns, Single Source Shortest Paths.

Dynamic Programming: All - Pairs Shortest Paths, Multistage graphs, optimal binary search tree, String editing, 0/1 Knapsack, Reliability Design.

Backtracking: The General Method, The 8-Queens Problem, Sum of Subsets, Graph Colouring, Hamiltonian Cycles, Knapsack problem

Branch and Bound: Least cost (LC) Search, The 15-Puzzle, Control Abstraction for LC-Search, Bounding, FIFO Branch-and-Bound, LC Branch and Bound, 0/1 Knapsack Problem, LC Branch-and Bound Solution, FIFO Branch-and-Bound Solution, Traveling Salesperson problem.

Limitations of Algorithm Power: Lower-Bound Arguments, Decision Trees, P, NP and NP – complete problems – Challenges of Numerical Algorithms. Limitations of Algorithms Power: Backtracking – Branch-and Bound– Approximation Algorithms for NP-hard Problems – Algorithms for solving Nonlinear Equations.

Text Books:

1. Fundamentals of computer algorithms E. Horowitz S. Sahni, Sanguthevar Rajasekaran, University Press.
2. Introduction to Algorithms by Thomas H. Corman, Charles E. Leiserson, Ronald R. Rivest & Clifford Stein, Prentice Hall of India, New Delhi, New Delhi.

Reference Books:

1. Data structures and algorithm analysis in C++ / Mark Allen Weiss, Florida International University. — Fourth edition.
2. Introduction to Design & Analysis of Algorithms by Anany Levitin, Pearson Education, New Delhi, 2003
3. Introduction to Algorithms by Thomas H. Corman, Charles E. Leiserson, Ronald R. Rivest & Clifford Stein, Prentice Hall of India, New Delhi, New Delhi.
4. The Design and Analysis of Computer Algorithms, Alfred V. Aho, John E. Hopcroft, Jeffrey D. Ullman