ALGORITHM DESIGN AND ANALYSIS	L	T	P	Data Structures
	3	1	0	

Course Objective: To introduce the concept of algorithmic efficiency by analyzing various algorithms such as Searching, Sorting, Divide-and-Conquer algorithms and to know about Dynamic Programming techniques, Greedy Paradigm, Back Tracking, Branch and Bound, and Computational Complexity.

S. NO	Course Outcomes (CO)
CO1	Abilty to analyse and compare the runtime complexities of algorithms using various techniques.

CO2	Abilty to classify and solve problems using Divide and Conquer technique
CO3	Abilty to classify and solve problems using Dynamic Programming Techniques
CO4	Abilty to classify and solve problems using Greedy Paradigm
CO5	Abilty to identify and solve problems using Back-Tracking and Branch and Bound techniques
CO6	Abilty to learn NP-complete and NP-hard problems, and design approximate Solutions.

S. NO	**Contents**	**Contact Hours**
UNIT 1	Introduction: Concept of algorithmic efficiency, run time analysis of algorithms, Asymptotic Notations. Growth of Functions, Master's Theorem, Substitution method and Recursion Tree method.	**6**
UNIT 2	Divide and Conquer: Structure of divide-and-conquer algorithms; examples: binary search, quick sort, Strassen Matrix Multiplication; merge sort, heap sort and Analysis of divide and conquer run time recurrence relations.	**7**
UNIT 3	Dynamic programming: Principles of dynamic programming. Applications: Rod cutting problem, Matrix Chain multiplication, Longest Common subsequence, Travelling salesman Problem, and Floyd-Warshall algorithm for all pair shortest paths.	**8**
UNIT 4	Greedy Method: Overview of the greedy paradigm examples of exact optimization solution: Activity Selection Problem., minimum cost spanning tree, approximate solutions: Knapsack problem, Kruskal’s algorithm and Prim’s algorithm for finding Minimum cost Spanning Trees, Dijkstra’s, and Bellman Ford Algorithm for finding Single source shortest paths, Huffman coding.	**8**
UNIT 5	Back tracking: Overview, 8-queen problem, and Knapsack problem, Traveling Salesman problem. Branch and bound: LC searching Bounding, FIFO branch and bound, LC branch and bound application: 0/1 Knapsack problem.	**7**
UNIT 6	Computational Complexity: Complexity measures, Polynomial Vs non-polynomial time complexity; NP-hard and NP-complete classes, examples: Circuit Satisfiability, Vertex cover, Subset Sum problem, Randomized Algorithms, String Matching, NP-Hard and NP-Completeness, Approximation Algorithms, Sorting Network, Matrix Operations, Polynomials and FFT, Number Theoretic Algorithms.	**6**
	TOTAL	**42**