CS: CS3205 COURSE NAME: DESIGN AND ANALYSIS OF ALGORITHMS

Course Prerequisites: Basic courses on programming, data structures, discrete structures, theory of computing.

Course Objectives:

1. Students will gain understanding of asymptotic notations and will be able to apply suitable mathematical techniques to find asymptotic time and space complexities of algorithms.
2. Students will develop the ability to formulate computational problems in the abstract and mathematically precise manner.
3. Student will gain understanding of different algorithm design paradigms such as divide and conquer, dynamic programming, greedy, backtracking and will apply suitable paradigm for designing algorithms for computational problems
4. Students will develop understanding of notions of NP-hardness and NP-completeness and their relationship with the intractability of decision problems.
5. Students will design randomized, approximation algorithms for some computational problems.
6. Students will be able to incorporate algorithm design principles, data structures and provide efficient solutions for complex computational problems.

Credits: 4....... **Teaching Scheme Theory: 2 Hours/Week**

Tutorial: 1 Hours/Week

Lab: 2 Hours/Week

Course Relevance: This is a foundational course for Computer science and Engineering. This course develops algorithmic thinking capability of students. Designing algorithms using suitable paradigm and analysing the algorithms for computational problems has a high relevance in all domains where computer science plays a crucial role (equally in Industry as well as research). This course is also an essential pre-requisite for advanced domain specific algorithmic courses such as Algorithmic Graph Theory, Algorithmic Number Theory, Computational Geometry, Motion planning and Robotics, etc, to give a few examples.

Once the student gains expertise in Algorithm design and in general gains ability of Algorithmic thinking, it facilitates in systematic study of any other domain (in computer science or otherwise) which demands logical thinking.

This course is also relevant for students who want to pursue research career in theory of computing, computational complexity theory, advanced algorithmic research.

SECTION-I

Topics and Contents:

Unit-I: Basic introduction and time and space complexity analysis:[COs Mapped: 3] [POs Mapped: 2, 3]

Asymptotic notations (Big Oh, small oh, Big Omega, Theta notations). Best case, average case, and worst-case time and space complexity of algorithms. Overview of searching, sorting algorithms. Adversary lower bounds (for the comparison-based sorting algorithms). Using Recurrence relations and Mathematical Induction to get asymptotic bounds on time complexity. Master’s theorem and applications. **[4 Hrs]**

Unit-II Divide and Conquer: [COs Mapped: 1, 2, 3, 6] [POs Mapped: 1, 2, 3, 4, 13]

General strategy, Analyzing Quick sort, Merge sort, Finding a majority element, Order statistics (randomized and deterministic algorithms), Efficient algorithms for Integer arithmetic (Euclid's algorithm, Karatsuba's algorithm for integer multiplication, fast exponentiation). **[4 Hrs]**

Unit-III Dynamic Programming: [COs Mapped: 1, 2, 3, 6] [POs Mapped: 1, 2, 3, 4, 13]

General strategy, simple dynamic programming-based algorithms to compute Fibonacci numbers, binomial coefficients, Matrix Chain multiplication, Coin change problem, 0-1 Knapsack, Traveling Salesperson Problem, All pair shortest path algorithm, Longest increasing subsequence problem, Largest independent set for trees. **[6 Hrs]**

SECTION-II

Topics and Contents:

Unit-IV Greedy and Backtracking strategy: [COs Mapped: 1, 2, 3, 6] [POs Mapped: 1, 2, 3, 4, 13]

Greedy: General strategy, Analysis and correctness proof of minimum spanning tree and shortest path algorithms, fractional knapsack problem, conflict free scheduling.

Backtracking: General strategy, n-queen problem, backtracking strategy for some NP-complete problems (e.g. graph coloring, subset sum problem, SUDOKU) **[4 Hrs]**

Unit-V Introduction to complexity classes and NP-completeness: [COs Mapped: 3, 4] [POs Mapped: 2, 3, 6]

Complexity classes P, NP, coNP, and their interrelation, Notion of polynomial time many one reductions reduction, Notion of NP-hardness and NP-completeness, Cook-Levin theorem and implication to P versus NP question, NP-hardness of halting problem. NP-Complete problems (some selected examples from - Satisfiability problem, Circuit-SAT, 3-CNF SAT, vertex cover problem, independent set problem, clique problem, Hamiltonian-circuit problem, subset sum problem, Integer Linear Programming) **[6 Hrs]**

Unit-VI Introduction to Randomized and Approximation algorithms: [COs Mapped: 3, 5] [POs Mapped: 2, 3, 12]

Introduction to randomness in computation, Las-Vegas and Monte-Carlo algorithms, Abundance of witnesses/solutions and application of randomization, solving SAT for formulas with "many" satisfying assignments, randomized quick sort, Karger's Min-cut algorithm, coupon collector problem,

Introduction to Approximation algorithms for NP-optimization problems, Approximation algorithm for Vertex Cover, metric Traveling-Sales-Person Problem (metric-TSP), Hardness of approximation for TSP. **[4 Hrs]**

Tutorials:

List of Tutorials (Any Thirteen)

Unit I: [COs Mapped: 3] [POs Mapped: 2, 3]

1. Problem solving based on asymptotic notations, solution of recurrences
2. Proving correctness of algorithms: some techniques

Unit II, III, IV: [COs Mapped: 1, 2, 3, 6] [POs Mapped: 1, 2, 3, 4, 13]

3. Problem solving based on Divide and Conquer strategy (Binary search interesting applications, counting inversions)
4. Advanced problem solving based on Divide and Conquer strategy (Discrete Ham-Sandwich theorem, efficient algorithm for Josephus problem)
5. Problem solving based on Dynamic Programming strategy (Largest sum contiguous block and generalizations, Optimal binary search tree (OBST) construction)
6. Advanced problem solving based on Dynamic Programming strategy (Winning strategy for two player games, Variants of shortest path algorithms)

7. Problem solving based on Greedy strategy with emphasis on proof of correctness
8. Problem solving based on Backtracking strategy

Unit V: [COs Mapped: 3, 4] [POs Mapped: 2, 3, 6]

9. reducing NP problems to Integer Linear Programming**.**
10. Problem solving based on complexity classes, NP-completeness.

Unit VI: [COs Mapped: 3, 5] [POs Mapped: 2, 3, 12]

11. Problem solving based on Randomized Algorithms
12. Problem solving based on Approximation Algorithms

Practical's:

List of Practical's (Any Six)

Unit II, III, IV: [COs Mapped: 1, 2, 3, 6] [POs Mapped: 1, 2, 3, 4, 13]

1. Assignment based on some simple coding problems on numbers, graphs, matrices
2. Assignment based on Divide and Conquer strategy (e.g. majority element search, finding kth rank element in an array)
3. Assignment based on Divide and Conquer strategy (e.g. efficient algorithm for Josephus problem using recurrence relations, fast modular exponentiation)
4. Assignment based on Dynamic Programming strategy (e.g. Matrix chain multiplication, Longest increasing subsequence)
5. Assignment based on Dynamic Programming strategy (e,g, All pair shortest path, Traveling Sales Person problem)
6. Assignment based on Greedy strategy (e.g. Huffman encoding)
7. Assignment based on Backtracking (e.g. graph coloring, n-queen problem)

Unit VI: [COs Mapped: 3, 5] [POs Mapped: 2, 3, 12]

8. Assignment based on analysis of quick sort (deterministic and randomized variant)
9. Assignment based on Las-Vegas and Monte-Carlo algorithm for majority element search
10. Assignment based on factor-2 approximation algorithm for metric-TSP

Course Projects:

List of Course Project Topics

1. Applications of A* algorithm in gaming
2. Pac-Man game
3. File compression techniques
4. Solution of Maze (comparing the backtracking based solution and Dijkstra’s algorithm)
5. Different exact and approximation algorithms for Travelling-Sales-Person Problem
6. Creation of Maze using backtracking
7. Knight tour algorithms
8. Network flow optimization and maximum matching
9. AI for different games such as minesweeper, shooting games, Hex, connect-4, sokoban, etc
10. SUDOKU solver
11. Graph theoretic algorithms
12. Computational Geometry Algorithms
13. AKS primality testing
14. Algorithms for factoring large integers
15. Randomized algorithms for primality testing (Miller-Rabin, Solovay-Strassen)
16. Slider puzzle game

Seminars:

List of Course Seminar Topics

1. Divide and Conquer Vs Dynamic Programming
2. Greedy strategy
3. NP-hardness
4. Backtracking strategy
5. Dynamic Programming Vs Greedy
6. Computational Complexity
7. Philosophical relevance of P Vs NP question
8. Complexity classes
9. Space complexity
10. Compression Techniques
11. Real world applications of Graph theoretic algorithms
12. Approximation algorithms
13. Hardness of approximation
14. Pseudorandom number generators

Group Discussion:

List of Group Discussion Topics

1. Greedy Algorithms
2. Dynamic Programming strategy
3. Dynamic Programming Vs Greedy
4. NP-completeness
5. P Vs NP question
6. Algorithm design paradigms
7. Different Searching techniques
8. Backtracking strategy
9. Relevance of Cook-Levin theorem
10. Randomness in computation
11. Approximation Algorithms
12. Application of Recursion

List of Home Assignments:

List of Design Based Home Assignments

1. Problem solving based on Divide and Conquer strategy
2. Problem solving based on Dynamic Programming strategy
3. Problem solving based on Greedy strategy
4. Problem solving based on Backtracking strategy
5. Problems on Randomized Algorithms
6. Problems on Approximation Algorithms
7. Problems on NP completeness

List of Case Study Based Home Assignments

1. AKS primality test
2. Quadratic sieve factoring algorithm
3. Huffman Encoding, LZW encoding
4. Network flow optimization algorithms
5. Approximation algorithms for TSP
6. Cook-Levin theorem and its relationship with intractability of computational problems
7. Sorting techniques

List of Blog Based Home Assignment

1. Approximation Algorithms
2. Randomized Algorithms
3. Computational Geometry Algorithms
4. Number Theoretic Algorithms
5. Graph Theoretic Algorithms
6. P Vs NP Problem
7. Complexity classes
8. Greedy Algorithms
9. Divide and Conquer Vs Dynamic Programming

List of Survey Based Home Assignments

1. Primality Testing Algorithms
2. Integer Factoring Algorithms
3. NP-complete problems
4. Compression Techniques
5. Shortest Path Algorithms
6. Algorithms for finding Minimum Weight Spanning Tree

Suggest an assessment Scheme:

Suggest an Assessment scheme that is best suited for the course. Ensure 360 degree assessment and check if it covers all aspects of Bloom's Taxonomy.

Text Books: ***(As per IEEE format)***

4. *Cormen, Leiserson, Rivest and Stein "Introduction to Algorithms" ,PHI 3nd edition, 2009. ISBN 81-203-2141-*

5. *Jon Kleinberg, Eva Tardos "Algorithm Design", Pearson, 1st edition, 2005. ISBN 978-81-317-0310-6*
6. *Dasgupta, Papadimitriu, Vazirani "Algorithms" McGraw-Hill Education; 1 edition (September 13, 2006), ISBN-10: 9780073523408, ISBN-13: 978-0073523408*

Reference Books: ***(As per IEEE format)***

5. *Motwani, Raghavan "Randomized Algorithms", Cambridge University Press; 1 edition (August 25, 1995), ISBN-10: 0521474655, ISBN-13: 978-0521474658*
6. *Vazirani, "Approximation Algorithms", Springer (December 8, 2010), ISBN-10: 3642084699, ISBN-13: 978-3642084690Gerd Keiser, MC Graw Hill International edition, optical fiber communication , third edition*

MOOCs Links and additional reading material:

1. *www.nptelvideos.in*

Course Outcomes:

Course Outcomes:

On the completion of course, student will able to

1. To formulate computational problems in abstract and mathematically precise manner
2. To design efficient algorithms for computational problems using appropriate algorithmic paradigm
3. To analyze asymptotic complexity of the algorithm for a complex computational problem using suitable mathematical techniques.
4. To establish NP-completeness of some decision problems, grasp the significance of the notion of NP-completeness and its relationship with intractability of the decision problems.
5. To understand significance of randomness, approximability in computation and design randomized algorithms for simple computational problems and design efficient approximation algorithms for standard NP-optimization problems.
6. To incorporate appropriate data structures, algorithmic paradigms to craft innovative scientific solutions for complex computing problems.

CO-PO Map:

CO1	CO2	CO3	CO4	CO5	CO6
PO1	PO3,PO4	PO2,PO3	PO6	PO12	PSO1
3	2,3	2,3	2	2	3

CO attainment levels:

CO No.	1	2	3	4	5	6
Attainment level	1	3	2	3	4	5

Future Course Mapping:

Mention other courses that can be taken after completion of this course

Advanced Algorithms, Computational Complexity, Computational Geometry, Algorithmic Number Theory, Algorithmic Graph Theory

Job Mapping:

What are the Job opportunities that one can get after learning this course

Algorithm design lie at heart of any Computer Science/Engineering application. Once the student gains expertise in Algorithm design and in general gains ability of Algorithmic thinking, it facilitates in systematic studying any other domain (in computer science or otherwise) which demands logical thinking. Algorithm design is an essential component of any job based on programming. All Industries in computer Engineering always look for a strong knowledge in Algorithm design and Data structures. If student wants to pursue higher education/ research in Computer Science, this course is must.