SILIGURI INSTITUTE OF TECHNOLOGY

COMPUTER SCIENCE & ENGINEERING

COURSE FILE

2020-21

PAPER NAME : Design & Analysis of Algorithm

PAPER CODE : PCC-CS404 & PCC-CS 494

Course Description

Course Title/Code: <u>Design and Analysis of Algorithm/PCC-CS404 & PCC-CS494</u>

Department: - CSE, Semester: - 1st, Year: - 2nd, Group: - A

Name of the Faculty: **Mr. Mithun Roy**

E-mail : mathmithunroy@gmail.com
Mobile No: 7044561269
WhatsApp No: 9434678869

Class Schedule:

Day	Monday (L1)	Friday (L2)
CSE (B)	11:40 AM – 12:30 PM	2:10 PM – 3:50 PM

Laboratory Schedule:

Day	Monday	Tuesday
Group A1	1:20 PM – 4:40 PM	---
Group A2	---	10:00 AM – 1:20 PM

Hours of Meeting Students: Any day (between 4:30 PM to 5:30 PM) (if required)

Course Objective:

i) The aim of this module is to learn how to develop efficient algorithms for simple computational tasks and reasoning about the correctness of them.

ii) Through the complexity measures, different range of behaviors of algorithms and the notion of tractable and intractable problems will be understood.

i) Course Outcomes:

After completion of this course the students are expected to be able to demonstrate following knowledge, skills and attitudes.

The Students will be able to:

ii) **PCC-CS404.1** For a given algorithms analyze worst-case running times of algorithms based on asymptotic analysis and justify the correctness of algorithms.

iii) **PCC-CS404.2** Describe the **greedy** paradigm and explain when an algorithmic design situation calls for it. For a given problem develop the greedy algorithms.

iv) **PCC-CS404.3** Describe the **divide-and-conquer** paradigm and explain when an algorithmic design situation calls for it. Synthesize divide-and-conquer algorithms. Derive and solve recurrence relation.

v) **PCC-CS404.4** Describe the **dynamic-programming** paradigm and explain when an algorithmic design situation calls for it. For a given problems of dynamic-programming and develop the **dynamic-programming** algorithms, and analyze it to determine its computational complexity.

vi) **PCC-CS404.5** develop the **backtracking** algorithms, and analyze it to determine its computational complexity.

vii) **PCC-CS404.6** For a given model engineering problem model it using graph and write the corresponding algorithm to solve the problems.

viii) **PCC-CS404.7** Explain the ways to analyze randomized algorithms (expected running time, probability of error).

ix) **PCC-CS404.8** Explain what an approximation algorithm is. Compute the approximation factor of an approximation algorithm.

a) Once the student has successfully complete this course, he/she must be able to answer the following questions or perform/demonstrate the following:

SN	QUESTION	BT- LEVEL
1.	**What** do you understand by an algorithm?	1
2.	**What** are the analytic issues of an algorithm?	1
3.	**Write** an algorithm to find the **maximum number among three numbers** and also calculate the running time complexity**.**	1
4.	**Write** an algorithm to calculate the **sum of two matrices** and also calculate the running time complexity.	1

5.	**Define Cook's** theorem. Prove that **3-SAT** is NP- Complete.	2
6.	**Find** out the Recurrence relation of recursive **Tower of Hanoi** problem and solve it for the input size n.	3
7.	**Solve** the following recurrence using iteration method. 1. T(n)=2T(n/2) + O(n)	3
8.	**Solve** the following recurrence using master method. 1. T(n)=2T(n/2) + O(n) 2. T(n)=4T(n/2) + O(n) 3. T(n)=T(n/2) + O(n)	3
9.	**Show** that the following equation is correct: $33n^2 + 4n = \Omega(n^2)$	3
10.	**Solve** $T(n) = aT(n/b) + O(n^k)$ where $a > 1$ and $b \geq 1$.	3
11.	**Find** out the running time complexity of the **Quick sort** algorithm in Best, Worst and Average cases.	4
12.	**Find** out the running time complexity of the **N-Queen** problem.	4
13.	**Implement** adjacent matrix and adjacent list of a given graph and also conclude which representation is better**.**	5
14.	**Implement** graph traversal techniques like **BFS** and **DFS**.	5
15.	**Implement Binary Search** with the help of **Divide & Conquer** strategy.	6
16.	**Implement** shortest path using **Dijkstra's** algorithm with the help of dynamic programming strategy**.**	6

Design & Analysis of Algorithm syllabus [in Chapters]
Code: PCC CS 404
Contact: 3L

CHAPTER-1

Complexity Analysis: [5L]

Time and Space Complexity, Different Asymptotic notations – their mathematical significance

CHAPTER-2

Divide and Conquer: [3L]

Basic method with the following case studies with proper analysis.

1) Binary Search.
2) Merge Sort.
3) Quick Sort and their complexity.

CHAPTER-3

Dynamic Programming: [4L]

Basic method with the following case studies with proper analysis.

1) Matrix Chain Multiplication.
2) All pair shortest paths
   a. Floyd-Warshall Algorithm.
3) Single source shortest path.
   a. Dijkstra's Algorithm.
   b. Bellmanford Algorithm.

CHAPTER-4

Backtracking: [2L]

Basic method with the following case studies with proper analysis.

1) N queens problem.
2) Graph coloring problem.

CHAPTER-5

Greedy Method: [4L]

Basic method with the following case studies.

1) Knapsack problem.
2) Job sequencing with deadlines.
3) Minimum cost spanning tree
   a. Prim's Algorithm.
   b. Kruskal's Algorithm.

CHAPTER-6

Disjoint set manipulation: [1L]

Set manipulation algorithm like UNION-FIND, union by rank.

CHAPTER-7

Graph traversal algorithm: [4L]

1) Breadth First Search(BFS)
2) Depth First Search(DFS)
3) Classification of edges
4) Topological Sorting

CHAPTER-8

Network Flow: [2L]

Ford Fulkerson algorithm, Max-Flow Min-Cut theorem (Statement and Illustration)

CHAPTER-9

Notion of NP-completeness: [4L]

P class, NP class, NP hard class, NP complete class – their interrelationship, Satisfiability problem, Cook's theorem (Statement only), and Clique decision problem.

CHAPTER-10

Approximation Algorithms: [1L]

Necessity of approximation scheme, performance guarantee, and polynomial time approximation schemes, vertex cover problem, travelling salesman problem.

a) Chapter Layout

Chapter No.	Chapter	Lecture Hours	Laboratory hours
Chapter - 1	Complexity Analysis	5 HRS	4 HRS
Chapter – 2	Divide and Conquer	3 HRS	4 HRS
Chapter – 3	Dynamic Programming	4 HRS	4 HRS
Chapter – 4	Backtracking	2 HRS	2 HRS
Chapter – 5	Greedy Method	4 HRS	2 HRS
Chapter – 6	Disjoint set manipulation	1 HRS	
Chapter – 7	Graph Traversal Algorithm	4 HRS	8 HRS
Chapter – 8	Network Flow	2 HRS	
Chapter – 9	Notion of NP-completeness	4 HRS	
Chapter - 10	Approximation Algorithms	1 HRS	
Total		30 HRS	24 HRS

b) Textbooks:

1. T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein , "Introduction to Algorithms"
2. Aho, J. Hopcroft and J. Ullman "The Design and Analysis of Algorithms" D. E. Knuth "The Art of Computer Programming", Vol. 3
3. Jon Kleiberg and Eva Tardos, "Algorithm Design"

c) Reference Books:

1. K. Mehlhorn , "Data Structures and Algorithms" - Vol. I & Vol. 2.
2. S. Baase "Computer Algorithms"
3. E. Horowitz and Shani "Fundamentals of Computer Algorithms"

d) Evaluation Scheme:

1) THEORY

Evaluation Criteria	Marks
First & Second Internal Exam*	15
Quiz/ Assignments	10
Attendance	5
University Exam	70
Total	100

*Two internal examinations are conducted; based on those two tests, average of them are considered in a scale of 15.

University Grading System:

Grade	Marks
O	90% and above
E	80 – 89.9%
A	70 – 79.9%
B	60 – 69.9%
C	50 – 59.9%
D	40 – 49.9%
F	Below 40%

LABORATORY

Evaluation Criteria	Marks
Internal Exam*	40
University Exam	60
Total	100

* Internal Evaluation will be based on daily lab performance as per the following schedule:

e) Laboratory Evaluation:

Expt. No.	Experiment Name	Schedule	Marks
P1	Experiment on different **Searching** Techniques and also judge the running time complexity. List of Experiments --- **1) Linear Search** **2) Binary Search**	2 HRS	2 + 2
P2	Experiment on some **recursion** problems also judge the running time complexity as well as plot the graph. List of Experiments --- **1) Calculate x^y** **2) N^{th} Fibonacci Number** **3) Tower of Hanoi**	2 HRS	2 + 2 + 2
P3	Experiment on different **Sorting** techniques and also judge the running time complexity. List of Experiments --- **4) Merge Sort** **5) Quick Sort** **6) Max-Min Problem**	4 HRS	2 + 2 + 2
P4	Experiment on **Greedy** algorithm strategy and also judge the running time complexity. **7) Knapsack Problem** **8) Job sequencing with deadlines**	2 HRS	2 + 2
P5	Experiment on **Dynamic Programming** algorithm strategy and also judge the running time complexity. **9) Matrix Chain Multiplication** **10) Floyd's Algorithm / Dijkstra's Algorithm**	4 HRS	3 + 3
P6	Experiment on **Backtracking** algorithm strategy and also judge the running time complexity. List of Experiments --- **11) 8 Queen Problem** **12) Graph Coloring / Hamiltonian Problem**	2 HRS	2+2
P7	Experiment on **Minimum Spanning Tree** and also judge the running time complexity. (Any one) List of Experiments --- **13) Prim's Algorithm** **14) Kruskal's Algorithm**	4 HRS	3 + 3
P8	Experiment on **Graph Traversal Techniques** and also judge the running time complexity. List of Experiments --- **15) BFS** **16) DFS**	4 HRS	2 + 2
P9	Experiment on **String Matching** Algorithm and also judge the running time complexity. **(beyond Syllabus)** **17) KMP**	3 HRS	4