National Institute of Technology, Jamshedpur
Department of Computer Science and Engineering

Autumn Semester Aug-Nov 2020

Course Handout

Course Code: CS 1302 (Theory course) and CS 1304 (Laboratory course)

Course Name: ***Design and Analysis of Algorithms***

Instructor: Rajiv R. Suman (CSE dept)

Pre-requisite: ***Data Structure***

Course Objective:

After completion of this course, the student would have a sound concept of:

(1) Notion of algorithms and its properties.
(2) Various strategies for design of algorithms for most classical computing problems of science and engineering.
(3) Expressing the algorithm formally using pseudo code.
(4) Validating an algorithm designed to solve a particular problem. This involves giving formal proof of correctness of the algorithm.
(5) Algorithm analysis, i.e., logically analyzing the pseudo code of an algorithm for its time and space complexities. This also involves comparing pros and cons of various algorithms designed for the same problem.
(6) Implementing algorithms using some programming language and debugging them. Program based on some algorithm is written and executed on sample data sets and tested to determine whether the output produced are correct. If not, algorithm and corresponding program are corrected so as to ensure correct design and implementation.
(7) Profiling or performance measurement of algorithms analysis, i.e., practical analysis of algorithms by implementing them using some programming language, executing a correct program with different data sets, test cases, etc and measuring actual time and space it takes to compute the results.

Scope:

Design and analysis of algorithms finds its application in almost all the areas of Computer Science & Engineering that includes but not limited to ----- operating system design, computer networks design including Internet and their applications, data communication, mobile computing, design, artificial intelligence, text and language processing, data compression, software engineering, computer organization and architecture, image processing, complier design, image processing, computer graphics, database systems design, data mining and warehousing, big data analysis, multimedia systems, VLSI system design, embedded and real time systems, formal testing & verifications, and other applications.

Course Description:

Introduction to algorithms**,** its fundamental properties**, concept of algorithm analysis for asymptotic time and space complexities, recurrence relations, review of important data structures** (array, linked list, stack, queue, trees, binary trees, AVL tree, splay tree, heaps, priority queues, graph, sets and disjoint set union, Union and Find algorithm for sets, etc), red-black tree, B-tree, B+-tree, Trie, 2–3–4 tree, **searching** (sequential, binary and hash searching techniques) **and sorting algorithms** (bubble sort, insertion sort, selection sort, heap sort, shell sort, address calculation sort, radix sort, radix exchange sort, merge sort, quick sort, counting sort, bucket sort, etc), **lower bound for comparison based sorting,** external sorting.

A***lgorithm design strategies***: **divide and conquer** (binary search on ordered list, merge sort, quick sort, finding max and min, fast algorithm to find 2nd largest item in an unordered list, selection of k_{th} smallest value in a list, Integer and matrix multiplications, etc), **greedy method** (Prim-Jarnik's , Kruskal's, and Baruvka's algorithms for minimum spanning tree, Dijkstra's shortest path algorithm, job sequencing with deadlines, activity selection problem, fractional knapsack problem, coin changing problem, Huffman code for data compression, etc), Bellman Ford shortest path algorithm, **dynamic programming** (matrix chain multiplication, longest common subsequence, Floyd Warshall shortest path algorithm, 0/1 knapsack problem, coin changing problem, Tiling problem, optimal binary search tree, optimal polygon triangulation, etc), **backtracking strategy** (n-queens problem, knight's tour on chessboard problem, sum of subset problem, 0/1 knapsack problem, graph coloring problem, Hamiltonian cycle problem, etc), **branch and bound method (**15-puzzle problem, travelling sales person problem, 0/1 knapsack problem, etc), **graph algorithms (**BFS, DFS, and D-**search** in a graph, connected components, articulation point, bridge, topological sort, Maximum flow in flow networks, Ford-Fulkerson method, etc**)**, **string matching algorithms** (naive string matching, Rabin-Karp Algorithm, finite automata based pattern matching, KMP algorithm, Boyer Moore algorithm, **Aho–Corasick** algorithm**,** approximate and partial matching algorithms, etc), **problem classes non-deterministic** algorithms, **P, NP, co-NP, NP-hard and NP-complete problems** (formula satisfiability, circuit satisfiability, 3-CNF satisfiability, clique decision problem, vertex cover problem, graph isomorphism, Hamiltonian cycle problem, bin-packing problem, travelling sales person problem, etc), polynomial-time **reducibility, , Cook's theorem, introduction to approximation algorithms for NP-hard problems**.

Text Book:

1. Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, Clifford Stein, "Introduction to Algorithms", 3rd Edition, PHI, Delhi.

Reference Books:

2. Ellis Horowitz, Sartaj Sahni, Sanguthevar Rajasekaran, "Fundamentals of Computer Algorithms", 2nd Edition, 2010, University Press, Hyderabad, India.
3. Ellis Horowitz and Sartaj Sahni, "Fundamentals of Data Structure", 2nd edition, University Press, Hyderabad, India.
4. Sara Baase and Allen Van Gelder, "Computer Algorithms", 3rd Ed. 2009, Pearson Education, India.
5. Michael T. Goodrich and Roberto Tamassia, "Algorithm Design", Wiley Student Edition, Wiley India, 2002.
6. Alfred V. Aho, John E. Hopcroft and Jeffrey D. Ulman, "The Design and Analysis of Computer Algorithms", Pearson, Delhi, 2011.
7. Robert Sedgewick, Kevin Wayne, "Algorithms", 4th Edition, Indian Edition available??
8. Anany Levitin, "Introduction to the Design and Analysis of Algorithms", 2nd Edition, 2008, Pearson Education, India.
9. Steven S. Skiena, "The Algorithm Design Manual", Springer Germany, 2nd ed. 2008.
10. Gilles Brassard and Paul Bratley, "Fundamentals of Algorithmics", 3rd Edition 2015, Pearson Education, India.
11. Jon Kleinberg, "Algorithm Design", 1/e, 2014, Pearson Education, India.
12. Donald E. Knuth, "The Art of Computer Programming", Vol-I, II, & III (three books), 3rd Ed, PHI, Delhi.
13. Niklaus Wirth, "Algorithm + Data Structure = Programs", Prentice-Hall Series, 1976.

14. D. Samanta, "Classic Data Strucures", 2nd Ed, 2013, PHI Delhi.
15. Rajeev Motwani, Prabhakar Raghavan, "Randomized Algorithms", Cambridge University Press, 1995.
16. Vijay V. Vazirani, "Approximation Algorithms", Springer, Springer-Verlag Berlin Heidelberg, 2003.
17. Ananth Grama, Vipin Kumar, Anshul Gupta, and George Karypis, "An Introduction to Parallel Computing: Design and Analysis of Algorithms", 2/e, Pearson Education, India.

Course Plan

Lecture No.	Topics to be covered	Learning Objective	Refer to Books
1	Introduction, its fundamental properties, role of algorithms in computing	Fundamental concepts	T1, R2
2-3	Concept of asymptotic time and space complexities using O-notation, Ω-notation, θ-notation, o-notation and ω-notation, solving recurrence relations for time and space complexities, the master method.	Concept of asymptotic time and space complexities	T1, R2
4-5	**review of important data structures** (stack, queue, trees, binary trees, heaps, priority queues, sets and disjoint set union, Union and Find algorithm for sets, graphs, etc)	Concept of important data structures required to learn later topics	T1, R2, R4
6-7	**searching** (sequential, binary and hash searching techniques)	Learn different searching strategies	T1
8-10	**sorting algorithms** (bubble sort, insertion sort, selection sort, heap sort, shell sort, address calculation sort, radix sort, radix exchange sort, merge sort, quick sort, etc), **lower bound for comparison based sorting, linear time sorting** (counting sort, bucket sort)	Learn sorting algorithms, their pros, cons, etc, comparison of sorting algorithms,	T1, R2, R13
11-12	**divide and conquer** (binary search on ordered list, merge sort, quick sort, finding max and min, selection of k_{th} smallest value in a list), randomized quick sort, balanced partitioning	Learn to apply divide and conquer strategy to suitable problems	T1, R2
13-14	**greedy method** (Prim-Jarnik's , Kruskal's, and Baruvka's algorithms for minimum spanning tree)	Learn to apply greedy strategy to suitable problems	T1, R2
15-16	**greedy method continued:** Dijkstra's shortest path algorithm, job sequencing with deadlines, fractional knapsack problem, etc),	Learn to apply greedy method for some more problems	T1, R2
17	Bellman Ford shortest path algorithm	Learn a versatile solution algorithm for single source shortest path problem	T1, R4

18-21	**dynamic programming, problems with optimal substructure** (matrix chain multiplication, longest common subsequence, Floyd Warshall shortest path algorithm, 0/1 knapsack problem, coin changing problem, Tiling problem, etc)	Learn to dynamic programming, i.e., bottom up solution strategy to suitable optimization problems	T1, R2-5, R4

Lecture No.	Topics to be covered	Learning Objective	Refer to Chapter, see (Book)
22-24	**backtracking strategy** (n-queens problem, sum of subsets problem, knight's tour on chessboard problem, 0/1 knapsack problem, graph coloring problem, Hamiltonian cycle problem, etc)	Learn techniques to solve some classical computing problems using efficient backtracking strategies	R2-7
25-26	**branch and bound method (**15-puzzle problem, travelling sales person problem, 0/1 knapsack problem, etc)	Learn a variant of backtracking strategy and its applications	R2-8, R9
27-31	**graph algorithms (**BFS, DFS, and D-search in a graph, connected components, articulation point, bridge, topological sort, maximum flow in Flow networks, etc**)**	Learn essential graph based algorithms	T1, R2
32-35	**string matching algorithms** (naive string matching, finite automata based pattern matching, Boyer Moore algorithm, Rabin-Karp Algorithm, KMP algorithm)	Learn string matching and text processing algorithms, brute force based and intelligent techniques	T1,R4
36-39	**problem classes P, NP, co-NP, NP-hard and NP-complete problems** (formula satisfiability, circuit satisfiability, 3-CNF satisfiability, clique decision problem, vertex cover problem, graph isomorphism, Hamiltonian cycle problem, bin-packing problem, travelling sales person problem, etc), **reducibility, non-deterministic algorithms, Cook's theorem.**	Learn the important concept of problem classifications based on their time complexities, learn whether it is feasible to attempt to solve a problem	T1
40-41	**Introduction to approximation algorithms for NP-hard problems**.	Learn to find the approximate solutions of problems that cannot be solved optimally in reasonable time	R2-12
42-45	**Review of concepts discussed, questions, answers, tutorials, etc**	Consolidate the concepts learned in previous classes	

Laboratory Assignments:

1. Writing C programs to implement the algorithms learnt in theory class and execute them with various data sets and analyze the result.
2. A mini project in Java.

Theory Course Evaluation Scheme:

EC No.	Evaluation Component	Duration	Weight	Date & Time	Nature of Component
1.	Mid Sem Exam	2 Hours	30%		Closed Book
2.	End Sem Exam	3 Hrs	50%		Closed Book
3.	Assignments	Two weeks	10%		Take home
4.	Surprise Quizzes	5 min	10%		Closed Book

Laboratory Assignment Evaluation: 70% based on correct implementation and execution of algorithms. Rest 30% based on viva-voce test.

.

Instructor's Email-id: rrsuman.cse@nitjsr.ac.in, Phone: 8434765977.