Keywords

network topologies, OSI model, TCP/IP model, client server model.

Design and Analysis of Algorithms (BHCS08) Discipline Specific Core Course - (DSC) Credit: 06

Course Objective

This course is designed to introduce the students to design and analyse algorithms in terms of efficiency and correctness. The course focuses on highlighting difference between various problem solving techniques for efficient algorithm design.

Course Learning Outcomes:

On successful completion of this course, the student will be able to:

1. Given an algorithm, identify the problem it solves.
2. Write algorithms choosing the best one or a combination of two or more of the algorithm design techniques: Iterative, divide-n-conquer, Greedy, Dynamic Programming using appropriate data structures.
3. Write proofs for correctness of algorithms.
4. Re-write a given algorithm replacing the (algorithm design) technique used with a more appropriate/efficient (algorithm design) technique.

Detailed Syllabus

Unit 1

Algorithm Design Techniques: Iterative technique: Applications to Sorting and Searching (review), their correctness and analysis. Divide and Conquer: Application to Sorting and Searching (review of binary search), merge sort, quick sort, their correctness and analysis.

Dynamic Programming: Application to various problems (for reference; Weighted Interval Scheduling, Sequence Alignment, Knapsack), their correctness and analysis. Greedy Algorithms: Application to various problems, their correctness and analysis.

Unit 2

More on Sorting and Searching: Heapsort, Lower Bounds using decision trees, sorting in Linear Time - Bucket Sort, Radix Sort and Count Sort, Medians & Order Statistics, complexity analysis and their correctness.

Unit 3

Advanced Analysis Technique: Amortized analysis

Unit 4

Graphs: Graph Algorithms - Breadth First Search, Depth First Search and its Applications.

Practical

1. a)Implement Insertion Sort (The program should report the number of comparisons)
   b)Implement Merge Sort(The program should report the number of comparisons)

2. Implement Heap Sort (The program should report the number of comparisons)

3. Implement Randomized Quick sort (The program should report the number of comparisons)

4. Implement Radix Sort

5. Create a Red-Black Tree and perform following operations on it: i. Insert a node ii. Delete a node iii. Search for a number & also report the color of the node containing this number.

6. Write a program to determine the LCS of two given sequences

7. Implement Breadth-First Search in a graph

8. Implement Depth-First Search in a graph

9. Write a program to determine the minimum spanning tree of a graph

For the algorithms at S.No 1 to 3 test run the algorithm on 100 different inputs of sizes varying from 30 to 1000. Count the number of comparisons and draw the graph. Compare it with a graph of nlogn.

References

1. Kleinberg, J., & Tardos, E. (2013). *Algorithm Design*. 1st edition. Pearson Education India.

Additional Resources

1. Cormen, T.H., Leiserson,C.E. Rivest, R.L., & Stein, C.(2015). *Introduction to Algorithms.* 3rd edition. PHI.

2. Sarabasse & Gleder A. V. (1999). *Computer Algorithm – Introduction to Design and Analysis*. 3rd edition. Pearson Education

Course Teaching Learning Process

- Use of ICT tools in conjunction with traditional class room teaching methods
- Interactive sessions
- Class discussions

Tentative weekly teaching plan is as follows:

Week	Content
1	Iterative technique: Applications to Sorting and Searching (review), their correctness and analysis
2	Divide and Conquer: Application to Sorting and Searching (review of binary search), merge sort, their correctness and analysis.
3	Divide and Conquer: quick sort, its correctness and analysis.
4	Heapsort, its correctness and analysis
5	Lower Bounds using decision trees, sorting in Linear Time - Bucket Sort, Radix Sort and Count Sort, their analysis
6	Medians & Order Statistics with analysis
7-9	Graph Algorithms: Graph Representation, Breadth First Search, Depth First Search, Applications
10-11	Greedy Algorithms: Application to various problems, their correctness and analysis
12-14	Dynamic Programming: Application to various problems, their correctness and analysis
15	Amortized analysis

Assessment Methods

Written tests, assignments, quizzes, presentations as announced by the instructor in the class.

Keywords

Brute Force Algorithm, divide and conquer, greedy, dynamic programming approaches, inplace algorithm, best / average / worst case running time of algorithms.