CS 205: ALGORITHMS (3-0-2: 4)

Introduction and basic concepts: Complexity measures, worst-case and average-case complexity functions, problem complexity, quick review of basic data structures and algorithm design principles.

Sorting and selection: Finding maximum and minimum, k largest elements in order; Sorting by selection, tournament and heap sort methods, lower bound for sorting, other sorting algorithms - radix sort, quick sort, merge sort; Selection of k-th largest element.

Searching and set manipulation: Searching in static table – binary search, path lengths in binary trees and applications, optimality of binary search in worst cast and average-case, binary search trees, construction of optimal weighted binary search trees; Searching in dynamic table – randomly grown binary search trees, AVL tree.

Hashing: Basic ingredients, analysis of hashing with chaining and with open addressing.

Union-Find problem: Tree representation of a set, weighted union and path compression-analysis and applications.

Graph problems: Graph searching – BFS, DFS, shortest first search, topological sort; connected and biconnected components; minimum spanning trees – Kruskal's and Prim's algorithms – Johnson's implementation of Prim's algorithm using priority queue data structures.

Algebraic problems: Evaluation of polynomials with or without preprocessing. Winograd's and Strassen's matrix multiplication algorithms and applications to related problems, FFT, simple lower bound results.

String processing: String searching and Pattern matching, Knuth-Morris-Pratt algorithm and its analysis.

Suggested Laboratory Assignments:

1. Implement Quickselect algorithm to find the k-th largest element.
2. Implement Quicksort algorithm.
3. Write program to construct an optimal binary search tree (OBST) from the keys.
4. Write the functions insert (key, value), delete (key) and find (key) to maintain a hash table of integer values.
5. Implement a disjoint sets forest data structure to represent disjoint sets as trees. Implement the functions MAKE (x), UNION (x, y) and FIND (x) on the data structure.
6. Given a graph in adjacency matrix form, implement a routine to print the topological sort of the given graph if it exists.
7. Write a routine to construct the minimal spanning tree of a given connected graph using Prim's algorithm.
8. Write a routine to multiply two polynomials given in coefficient form in time $\Theta(n.\log(n))$, where n is the degree bound of the two polynomials.
9. Implement Strassen's algorithm for matrix multiplication.
10. Given a text string T and a pattern string P, write a routine implementing algorithm KMP to print if pattern P matches in text string T along with the index where it matches.

Text Book:

1. A. Aho, J. Hopcroft and J. Ullman; The Design and Analysis of Computer Algorithms, Addison-Wesley.

References:

1. S. Baase; Computer Algorithms: Introduction to Design and Analysis, Addison-Wesley.
2. T. H. Cormen, C.E. Leiserson and R.L.Rivest: Introduction to Algorithms, PHI.
3. E. Horowitz and S. Sahni: Fundamental of Computer Algorithms, Galgotia Publ.
4. K. Mehlhorn: Data Structures and Algorithms, Vol. 1 and Vol. 2, Springer-Verlag.