Approval: OTA in 3rd Convocation Meeting

Course Name: Advanced Data Structures and Algorithms
Course Number: CS-580
Credits: 3-0-1-4
Prerequisites: IC-250, CS-208
Intended for: B.Tech.
Distribution: Compulsory for CS; CS elective for EE and ME
Semester: IV

Preamble:

The proposed new curriculum for CS includes six discipline core courses:

1. Mathematical Foundations of Computer Science

2. Advanced Data Structures and Algorithms

3. Paradigms of Programming

4. Computer Organization

5. Information Systems

6. Communicating Distributed Processes

The proposed course follows the new CS curriculum design approach that strives to cover in the above-mentioned six core courses all the fundamental concepts that a CS undergraduate student must know of.

The course proposal attempts to include the fundamental topics in data structures and algorithms. The topics form the essential core of this course that must be covered comprehensively, with lots of examples, practice exercises, and weekly programming labs. In he proposed contents, there are some topics that are marked "advanced". These topics can be introduced depending on background and interests of the class and if the time permits. These and some other advanced topics should be covered in greater detail in advanced discipline electives.

Objective:

After the students have gone through a course on discrete structures, where they learn the formal and abstract representations of data and its manipulation, a course on data structures and algorithms should teach the students concrete implementations and manipulations of such discrete structures and their use in design and analysis of non-trivial algorithms for a given computational task. On completion of such a course, the students should be able to

- analyze the asymptotic performance of algorithm

- demonstrate their familiarity with major data structures, rules to manipulate those, and their canonical applications

- construct efficient algorithms for some common computer engineering design problems Further, as programming is an integral part of the CS education, in this course the students should implement the data structures and algorithms they learn, compute the corresponding achievable performance (computation

time, memory requirement, etc), and if possible compare the achievable performance among alternative designs and implementations.

Syllabus:

1. **Complexity Analysis: (2 hours)**

Time and Space complexity of algorithms, asymptotic analysis, average and worst case analysis, asymptotic notation, importance of efficient algorithms, program performance measurement, data structures and algorithms.

2. **Stacks and Queues: (4 hours)**

ADT, sequential and linked implementations, representative applications: towers of Hanoi, parenthesis matching, finding path in a maze.

3. **Lists: (6 hours)**

ADT, sequential and linked representations, comparison of insertion, deletion and search operations for sequential and linked lists, list and chain classes, doubly linked lists, circular lists, applications of lists in bin sort, radix sort, sparse tables.

Advanced topic(s): Skip lists

4. **Dictionary: (1 hour)**

ADT, array and tree based implementations.

5. **Hashing: (4 hours)**

Search efficiency in lists and skip lists, hashing as a search structure, hash table, collision resolution, linear open addressing, chains, hash tables in data-compression, LZW algorithm.

Advanced topic(s): Universal hashing

6. **Heaps: (3 hours)**

Heaps as priority queues, heap implementation, insertion and deletion operations, binary heaps, heapsort, heaps in Huffman coding.

Advanced topic(s): Binomial, Fibonacci, and Leftist heaps

7. **Trees: (8 hours)**

ADT, sequential and linked implementations, tree traversal methods and algorithms, Binary trees and their properties, tournament trees, use of winner trees in mergesort, bin packing.

Advanced topic(s): Threaded binary trees - differentiation

8. **Search Trees: (3 hours)**

Binary search trees, search efficiency, insertion and deletion operations, importance of balancing, Tries, 2-3 tree, B-tree.

Advanced topic(s): AVL trees, searching, insertion and deletions in AVL trees

9. **Graphs:** Definition, terminology, directed and undirected graphs, properties, implementation – adjacency matrix and linked adjacency chains, connectivity in graphs, graph traversal – breadth first and depth first, spanning trees.

10. **Basic algorithmic techniques: (7 hours)**

Greedy algorithms, divide & conquer, dynamic programming. Search techniques - backtracking, Sorting algorithms with analysis, integer sorting, selection sort. Graph algorithms: DFS and BFS with applications, MST and shortest paths.

Reference Books:

1. A. Aho and J. Ullman, Foundations of Computer Science, W. H. Freeman, 1992. Available online at: http://infolab.stanford.edu/~ullman/focs.html

2. A. V. Aho, J. D. Ullman, and J. E. Hopcroft, Data Structures and Algorithms, Addison-Wesley, 1983.

3. A. M. Tenenbaum, Y. Langsam, and M. J. Augenstein, Data Structures Using C and C++, Prentice Hall, 2/e, 1995.

4. S. Sahni, Data Structures, Algorithms, and Applications in C++, Silicon Press, 2/e, 2005.

5. E. Horowitz, S. Sahni, and D. Mehta, Fundamentals of Data Structures in C++, Silicon Press, 2/e, 2006.

6. M. A. Weiss, Data Structures and Algorithm Analysis in C++, Prentice Hall, 4/e, 2013.

7. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, Introduction to Algorithms, MIT Press, 3/e, 2009.