

CS430 - Introduction to Algorithms

Last Updated - 03/01/02

Course Manager – Dr. Sanjiv Kapoor, Professor

3 credit hours; required for CS, elective for CPE; 150 min. lecture each week

Current Catalog Description - Introduction to the design, behavior, and analysis of computer algorithms. Searching, sorting, and combinatorial algorithms are emphasized. Worst case and average bounds on time and space usage. Prerequisites: (CS 330 and CS 331) or CS401 or CS403.

Textbook

- Cormen, Leiserson and Rivest, *Introduction to Algorithms*, MIT Press/McGraw Hill

References - other textbooks or materials

- A. Aho, J. Hopcroft and J.D. Ullman, *Design and Analysis of Algorithms*, Addison-Wesley.

Course Goals - Students should be able to:

- Use big O, omega, and theta notation to give asymptotic upper, lower, and tight bounds on time and space complexity of algorithms.
- Determine the time complexity of simple algorithms, deduce the recurrence relations that describe the time complexity of recursively defined algorithms, and solve simple recurrence relations.
- Design algorithms using the brute-force, greedy, dynamic programming, divide-and-conquer, branch and bound strategies.
- Design algorithms using at least one other algorithmic strategy from the list of topics for this unit.
- Use and implement the fundamental abstract data types -- specifically including hash tables, binary search trees, and graphs -- necessary to solve algorithmic problems efficiently.
- Solve problems using techniques learned in the design of sequential search, binary search, $O(N \log N)$ sorting algorithms, and fundamental graph algorithms, including depth-first and breadth-first search, single-source and all-pairs shortest paths, and at least one minimum spanning tree algorithm.
- Demonstrate the following abilities: to evaluate algorithms, to select from a range of possible options, to provide justification for that selection, and to implement the algorithm in simple programming contexts.
- Communicate theoretical and experimental analyses of a set of algorithms (i.e. sorting) in a lab report format.

Prerequisites by Topic

- Object-Oriented Programming: functions, pointers, recursion, classes
- Data Structures: abstract data types, lists, stacks, queues, trees
- Discrete Mathematics: sets, functions, counting, proofs

Major Topics Covered in Course

1. Introduction to Algorithm Design, Complexity analysis including elementary tools like O-Notations, Recurrence Relations	4.5 hours
2. Introduction to Backtracking and Branch and Bound	3 hours
3. Introduction to Dynamic Programming	4.5 hours
4. Divide and Conquer and Greedy Methods (using Traveling Salesman Problem, Knapsack Problem and Optimum Triangulation of Convex Polygons)	3 hours
5. Sorting Methods - Quicksort, Mergesort, Heaps and Heapsort, Lower bound on sorting	4.5 hours
6. Searching I - Hash Functions and Hashing, Union Find	3 hours
7. Searching II-- Binary Search Trees, Balanced Binary Search Trees (AVL Trees, 2-3 trees/ Red-Black trees)	6 hours
8. Graph Algorithms I - Depth First Search, Breadth First search, Bi-connectivity, Topological Sort	4.5 hours
9. Graph Algorithms II - Minimum Spanning Trees, Shortest Paths	4.5 hours

10. String Matching	1.5 hours
11. NP-Complete Problems	3 hours
12. Parallel Model of Computing - Example Sorting (*)	1.5 hours
Midterm Exam	1.5 hours
Final Exam	-
(*) Optional Topics	45 hours

Laboratory projects (specify number of weeks on each)

- 1 object-oriented programming project involving algorithmic design/analysis/implementation (individual, 3-4 weeks)

Estimate CSAB Category Content in Credit Hours

	CORE	ADVANCED		CORE	ADVANCED
Data Structures		1	Computer Organization and Architecture	0	
Algorithms		2	Concepts of Programming Languages	0	
Software Design	0				

Oral and Written Communications - Every student is required to submit at least 1 written reports (not including exams, tests, quizzes, or commented programs) of typically 5 pages and to make 0 oral presentations of typically 0 minutes duration. Include only material that is graded for grammar, spelling, style, and so forth, as well as for technical content, completeness, and accuracy.

- Design and analysis for the course algorithms programming project

Social and Ethical Issues - Please list the topics that address the social and ethical implications of computing covered in all course sections. Estimate the class time spent on each topic. In what ways are the students in this course graded on their understanding of these topics (e.g., test questions, essays, oral presentations, and so forth)?

- none

Theoretical Foundations - Please list the types of theoretical material covered, and estimate the time devoted to such coverage in contact (lecture and lab) hours.

- The majority of this course is theoretical material, see Major Topics Covered in Course

Problem Analysis - Please describe the problem analysis experiences common to all course sections.

- 3-4 homework assignments with 10-15 problems each in analyzing, developing and applying algorithms

Solution Design - Please describe the design experiences common to all course sections.

- 1 object-oriented programming project involving algorithmic design/analysis/implementation (individual, 3-4 weeks)

Other Course Information

- Additional Suggested Course Assignments
 - 3-4 homework assignments with 10-15 problems each in analyzing, developing and applying algorithms
 - 1 midterm exam (75 minutes)
 - 1 final exam (120 minutes)
- Planned Course Enhancements
 - Classification as Communications (C) Course (Fall 2002)