CMPSC 465
Data Structures and Algorithms
Required Course in Computer Science and Computer Engineering

Catalog Data: Data Structures and Algorithms (3)
Fundamental concepts of computer science: data structures, analysis of algorithms, recursion, trees, sets, graphs, sorting. Prerequisite: CMPSC 360 or MATH 311W.


Course Objectives: CMPSC 465 provides upper-division students with the principles of efficient algorithm design, and teaches them how to analyze the asymptotic behavior of both recursive and non-recursive algorithms. The first objective is to teach a precise notion of efficiency: time and space complexity. The second is to provide mathematical tools to compare efficiency of various algorithms or programs: asymptotic notation, \(O\) and \(\Omega\), as well as comparing the asymptotic rate of growth of various functions, especially those that can be expected as characterizations of algorithm complexity. These tools contain the basic methods for finding the rates of growth of functions, which include solving recurrence relations. A part of this objective is to convey the dramatic importance of the selection of an appropriate algorithmic method for a given task. The third goal is to provide a variety of tools which are indispensable in the design of efficient algorithms. These tools consist of: (a) building blocks, which include the most important data structures and several classic algorithms, (b) design paradigms, including divide-and-conquer, greedy and dynamic programming; several classic algorithms are presented as applications of these paradigms. It is assumed that students entering CSE 465 know how to analyze the algorithms for correctness, but further mastery of this aspect of algorithm design is also an objective of this course.

Primary Course Outcomes: Upon completion of the course, students should be able to successfully complete:

- For given functions, provide simplified upper and lower bounds using \(O\) and \(\Omega\)-notation.
- Sort functions by their asymptotic growth.
- Simplify functions defined by expressions involving sums.
- For nicely structured non-recursive algorithms (e.g., involving nested loops), write expressions for their running time.
- For recursive algorithms, write recurrence equations for their running time.
- Be able to describe algorithms using pseudo-code.
- Know standard data structures, like linked lists, stacks, queues, priority queues, heaps, search trees, some variant of balanced search trees, hash tables, and data structures to maintain a collection of disjoint sets.
- Be able to select appropriate data structures for given computational tasks.
- Know standard algorithms, like various sorting algorithms, algorithms to do order statistics, algorithms for operating on various data structures like heaps, search trees and hash tables, algorithms for various graph problems like depth-first search, breadth-first search, topological sort, computing shortest paths and minimum spanning trees. Be able to figure out the time and storage complexity of such algorithms.
- Know standard design principles like divide-and-conquer, greedy and dynamic programming.
- Be able to apply these design principles to design efficient algorithms for simple computational tasks.
Relationship to Undergraduate Program Outcomes: CMPSC 465 is providing intensive training in analyzing the space/time complexity of algorithms using discrete mathematics, including the appropriate use of O-notation and recurrence relations, and CMPSC 465 is the main course to systematically study the methods to analyze algorithms and computer code for efficiency.

Required Topics: (One possible order)

- Introduction
- Growth of functions
- Recurrences
- Abstract data types, queues, heaps, priority queues
- Quicksort
- Order statistics, decision trees, radix sort
- Hashing
- Binary search trees
- Balanced binary search trees, B-trees
- Combining data structures; graph representations
- DFS (depth-first search), BFS (breadth-first search), topological sorting
- Dynamic programming
- Data structures for disjoint sets
- Greedy algorithms, spanning trees
- Shortest paths

Class Format: Three lectures per week.

Professional Component: Algorithms are the essence of computing. They are an integral part of almost all sophisticated tasks of computer professionals.

Evaluation:

- 50 – 60 % Exams
- 30 – 50 % Homeworks
- 0 – 10 % Quizzes

Author: Piotr Berman, Martin Fürer, Jonathan Goldstine, Adam Smith, Sofya Raskhodnikova
Last Revised: June 1, 2007