ICS 353—Design and Analysis of Algorithms

Term: 071 Section: 3

Time & Place: SMW 11 – 12, Bldg 23-011

Instructor: Ebrahim Malalla
Office: Bldg 22 (124-8)
Phone: 860-3819
E-mail: malalla@kfupm.edu.sa
Course Site: http://www.ccse.kfupm.edu.sa/~malalla/ICS353/index.htm
Office Hours: SMW 12:30 – 2, and whenever you catch me.

Description

The course introduces the student to the classical techniques and paradigms used in the design and analysis of algorithms and data structures. Some of the covered techniques are induction and recursion, divide and conquer, dynamic programming, and greedy approach. Techniques like backtracking, branch and bound, and randomization are also introduced to deal with NP-Complete problems. Students will be able to practice their skills on many well-known algorithms and data structures designed to solve real-life problems.

Current Catalog Description

Algorithms and problem solving; basic algorithmic analysis; advanced algorithmic analysis; advanced data structures; algorithmic strategies & analysis of fundamental computing algorithms; basic computability; the complexity classes P and NP.

Prerequisites

ICS 202 and ICS 253 are the only official prerequisite for this course. In particular, some programming skills and a good background in discrete mathematics, data structures, and probability will be very helpful.

Course Objectives

1. To know the importance of studying the complexity of a given algorithm.
2. To study various algorithmic design techniques.
3. To utilize data structures and/or algorithmic design techniques in solving new problems.
4. To know and understand basic computability concepts and the complexity classes P, NP, and NP-Complete.
5. To study some techniques for solving hard problems.

Course Learning Outcomes

After completion of this course, the student shall be able to:

1. Describe, apply and analyze the complexity of certain divide and conquer, greedy, and dynamic programming algorithms.
2. Identify and analyze criteria and specifications appropriate to new problems, and choose the appropriate algorithmic design technique for their solution.
3. Describe the classes P, NP, and NP-Complete and be able to prove that a certain problem is NP-Complete.
4. Explain and apply backtracking and branch and bound techniques to deal with some hard problems.

Textbook

The official textbook is

M. H. Alsuwaiyel, *Algorithms: Design Techniques and Analysis*, Lecture Notes Series on Computing, vol. 7, 1999

Students are also encouraged to refer to other books on design and analysis of algorithms available in the library. Some of the highly recommended books are:

1. D. Knuth, *The Art of Computer Programming: Sorting and Searching*, Vol. 3, 3rd Ed., Addison-Wesley, 1998.
2. T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, *Introduction to Algorithms*, 2nd Ed., McGraw Hill, 2001.
3. R. Sedgewick and P. Flajolet, *An Introduction to the Analysis of Algorithms*, Addison-Wesley, 1996.

Evaluation

Assignments & Quizzes	15%
Major Exam I Sat. Oct. 27th at 7-9 pm.	25%
Major Exam II Wed. Jun. 2nd at 5-7 pm.	25%
Comprehensive Final Exam	35%

Contents

The following schedule is tentative and subjected to changes. Any change will be announced in the class and course website/ WebCT.

Weeks	Topics	Chapters
1-3	**Overview of algorithm design and analysis.** Sequential and binary search. Selection and Insertion Sorts. Bottom-up Mergesort. Time and space complexities. Worst and average-cases analysis. Asymptotic notations.	1 – {1.11, 1.13}
4-5	**Heaps**. Operations on heaps. Heapsort. (*Disjoint Set Data Structure. Union-Find algorithms*).	4
6-7	**Techniques based on Induction and Recursion.** Converting iteration to recursion. Radix Sort. Polynomial evaluation. Integer exponentiation. Finding the majority element.	5 – {5.6}
8-9	**Divide and Conquer.** Top-down Mergesort. Deterministic and randomized Quicksort algorithms. Finding Median and k-th smallest element. Selection algorithm.	6 – {6.7, 6.8, 6.9}
10-11	**Greedy Approach.** Dijkstra's single source shortest path algorithm. Prim's and Kruskal's minimum spanning tree. Huffman's compression algorithm.	8
11-12	**Dynamic Programming.** The longest common subsequence problem The Knapsack problem. Floyd's all-pairs shortest path algorithm.	7 – {7.3}
14	**Graph traversal.** DFS and BFS algorithms and their applications.	9 – {9.3.3}
15	**Coping with NP-Completeness**. P, NP and NP-complete classes. Backtracking, randomization, and approximation. Examples: 3-colorability, 8-queens problem, and random sampling.	10 – {10.5., 10.6, 10.7} +13.3+14.7

REMINDERS

1. It is strongly recommended that class notes be taken on a regular basis.
2. The course website/WebCT is an important source of information. It will be updated regularly to contain up-to-date announcements, handouts, assignments and their solutions.
3. By the university rules, 9 absences yield a DN grade.
4. No assignments would be accepted without penalty after the due date.