Basic Course Information CS 5300

Course Title: Advanced Algorithm Design and Analysis

Units: 3

C/S Classification #: C-2

Component (select one): Lecture

Instructional Mode (select all appropriate choices): Face-to-Face and web-assisted

Grading Basis (select one): Graded only

Repeat Basis (select one): May be taken only once

Cross listed Course (if offered with another department):

Dual-listed Course (if offered as lower/upper division or undergraduate/graduate):

Major course/Service course/GE course (select all appropriate choices): Major course

Prepared by: Gilbert Young

I Catalog Description

Advanced problem domains, including graph problems, pattern matching, compression, network flow and transforms. Amortized and average case analysis. Lower bounds. Approximation techniques. Probabilistic algorithms.

II Required Coursework and Background

Pre-requisite(s): CS 3310 or consent of instructor

III Expected Outcomes

On successful completion of this course, students will be able to:

1. Master the principles and techniques underlying the design and analysis of efficient computer algorithms.
2. Apply specific algorithm design techniques to design good algorithms for several well-known mathematical problems
3. Analyze the resource requirements of an algorithm
4. Prove the correctness of an algorithm

Outcomes of this course will build student capacity in each of the following areas as defined by programmatic objectives for the computer science major.

P-SLO 1. An ability to frame and model real-world problems that can be addressed computationally, and evaluate multiple computational approaches and select the most appropriate one.

P-SLO 3. An ability to build applications, either individually or in a team, that are robust, reliable, and maintainable.

IV Instructional Materials

Texts may vary with instructor and over time. Examples of possible texts include:

E. Horowitz, S. Sahni and S. Rajasekaran, Computer Algorithms, Computer Science Press, 2008

Computers and Intractability: A Guide to the Theory of NP-Completeness by Garey and Johnson, W.H. Freeman and Company, 1979

Tarjan, Data Structures and Network Algorithms , SIAM, 1983

V Minimum Student Material

Course textbooks

VI Minimum College Facilities

Computer laboratories, Blackboard, classroom with a projection system

VII Course Outline

Algorithm Design: Greedy algorithms, divide-and-conquer, dynamic programming, branch-and-bound, probabilistic methods.  
Algorithm Analysis: Computational models, complexity measures, asymptotic notation, average, worst-case, and amortized analyses, constructing and solving recurrence relations, lower bounds, invariants and correctness proofs.

Classic Problems: Graph problems, matching and network flow, computational geometry, fast Fourier transform, sorting and searching, pattern matching, balanced tree schemes, sorting networks, data encoding, etc.

Advanced Topics: NP-complete and NP-hard problems, polynomial time restrictions, approximation schemes, parallel algorithms.

VIII Instructional Methods

Lecture

Problem-solving

Discussion

Project-based learning

IX Evaluation of Outcomes

A. Student Assessment

i homework assignments

ii term paper

iii exam

B. Meaningful Writing Assignment

Students shall produce written solutions or proofs to problems that are assigned as homework and explain their reasoning. Students are required to write a term paper after conducting a research study on an advanced topic of computer algorithms.

C. A Matrix of Course Student Learning Outcomes vs Methods of Assessment

If the course is being evaluated for accreditation purposes, approved department accreditation assessment tools will additionally be utilized.

|  |  |  |  |
| --- | --- | --- | --- |
| Course Learning Outcomes | Methods of Assessment | | |
| Homework Assignments | Team Paper | Exams |
| Master the principles and techniques underlying the design and analysis of efficient computer algorithms. | x | x | x |
| Apply specific algorithm design techniques to design good algorithms for several well-known mathematical problems | x | x | x |
| Analyze the resource requirements of an algorithm | x | x | x |
| Prove the correctness of an algorithm | x | x | x |