**Basic Course Information:** CS 5310

Course Title: Computability & Complexity Theory

Units: 3 units

CS number: C-2

Component: Lecture

Instructional Mode: Face-to-Face and web-assisted

Grading Basis: Graded only

Repeated Basis: May be taken only once

Cross listed Course: N/A

Dual-listed Course: N/A

Major course/Service course/GE course: Major course

Date Prepared: March 31, 2015

Prepared by: Daisy Sang

# I. Catalog Description

Formalizing problems and algorithms. Characterizations and properties of complexity classes, undecidability. Cook-Levin Theorem, NP-complete problems, proof of NP-completeness.

**II. Required Coursework and Background**

Pre-requisite(s): CS 3110 and CS 3310, or consent of instructor.

# III. Expected Outcomes

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

* Design Turing machines for recursively enumerable languages
* Learn Church-Turing Thesis
* Gain Cantor’s diagonalization argument
* Master the use of reductions to prove certain problems undecidable
* Master the use of polynomial time reductions to prove certain problems NP-complete
* Recognize various complexity classes

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.

# IV. Instructional Materials

Required text:

Michael Sipser, Introduction to the Theory of Computation, 3rd Edition, Cengage Learning, 2012. ISBN-13: 978-1133187790

References:

M. Garey and D. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman, 1979. ISBN-13: 978-0716710455

Elaine Rich, Automata, Computability and Complexity: Theory and Applications, Prentice Hall, 2007. ISBN-13: 978-0132288064

# V. Minimum Student Material

Textbook and class handouts

# VI. Minimum College Facilities

A classroom with a projection system and a computer lab

# VII. Course Outline

1. Computability Theory
   1. Turing machines
   2. Decidability and recursive enumerability
   3. The Church-Turing Thesis
   4. Equivalence with other models of computation
   5. The definition of algorithms
   6. Diagonalization and the Halting problem
   7. Mapping reducibility
2. Complexity Theory
   1. Structural complexity vs Computational complexity
   2. The Chomsky hierarchy (Regular, CFL, CSL, Recursively enumerable)
   3. The Time hierarchy (P, NP, NP-complete, NP-hard, and examples)
   4. The Space hierarchy (Deterministic space, nondeterministic space, logarithmic space, polynomial space)
   5. Intractability
   6. The Cook-Levin Theorem
   7. Proof of NP-completeness by reduction

# VIII. Instructional Methods

Lecture

Problem-solving/Discussion

In-class exercises

Small group activities

Assigned readings

# IX. Evaluation of Outcomes

A. Student Assessment

1. Homework assignments

2. Midterm exam

3. Final exam

B. Meaningful Writing Assignment

Students shall produce written solutions or proofs to problems that are assigned as homework and explain their reasoning.

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 | Exams |
| Design Turing machines for recursively enumerable languages | x | x |
| Learn Church-Turing Thesis | x | x |
| Gain Cantor’s diagonalization argument | x | x |
| Master the use of reductions to prove certain problems undecidable | x | x |
| Master the use of polynomial time reductions to prove certain problems NP-complete | x | x |
| Recognize various complexity classes | x | x |