Theory of Computation

CSC 544 – Spring 2015

Webpage: http://homepage.cs.uri.edu/faculty/hamel/courses/2015/spring2015/csc544

Introduction

Computation and algorithms seem to occur naturally in our daily lives. Consider counting change or following a recipe to make your favorite dish. The art of programming takes this to the limit by formally encoding computations and algorithms for a machine to follow. What are the limits of this algorithmic approach? Are there mathematical objects that cannot be computed by a machine via an algorithm? If we can express a computation via an algorithm how long would it take to compute? How much space would the computation consume?

In this course we will investigate many of these interesting questions. As our main tool we will use an idealized general purpose computer invented by Alan Turing: the Turing Machine. This idealized machine allows us to study the limits of computability and the complexity of computations without having to worry too much about actual hardware. We will also touch on an alternative formulation of computation based on recursive function applications. This approach was proposed by Alonso Church in the 1940's and is now known as the lambda-calculus. Church and Turing reached the same conclusions about computability using their models and we often refer to this central result as the "Church-Turing Thesis".

The goal of the course is that you are exposed to the terminology of the theory of computing and are familiar with some of the major results in computability and complexity theory. This course will allow you to understand current research in algorithmics, especially in the area of bioinformatics where computability and complexity are major concerns. The hope is that studying algorithms, computation, and complexity at this abstract level will provide you with some insights into your programming tasks and will provide perhaps a different vantage point of computing itself.

Required Text

Introduction to the Theory of Computation, Michael Sipser (any edition)

Instructor

Dr. Lutz Hamel
email: hamel@cs.uri.edu
office: Tyler Hall, Rm 251

Prerequisites:

CSC440, CSC445 or equivalent.

Grading:

Homework	50%
Midterm	25%
Final	25%

Policies:

- Check the website (often)! I will try to keep the website as up-to-date as possible.
- Class attendance, promptness, participation, and adequate preparation for each class are expected. If you are absent, it is your responsibility to find out what you missed (e.g. handouts, announcements, assignments, new material, etc.)
- **Late assignments:** Late assignments will **not** be accepted.
- Make-up quizzes and exams will **not** be given without a valid excuse, such as illness. If you are unable to attend a scheduled examination due to valid reasons, please inform myself, or the department office in Tyler Hall, prior to the exam time. Under such circumstances, you are not to discuss the exam with any other class member until after a make-up exam has been completed.
- All work is to be the result of your own individual efforts unless explicitly stated otherwise. Plagiarism, unauthorized cooperation or any form of cheating will be brought to the attention of the Dean for disciplinary action. See the appropriate sections (8.27) of the University Manual.
- Software piracy will be dealt with exactly like stealing of university or departmental property. Any abuse of computer or software equipment will be subject to disciplinary action.

Tentative Course Outline:

Computability Theory: Turing machines, Church-Turing thesis, decidability, halting problem, reducibility, recursion, lambda-calculus, primitive recursive functions, an algorithmic definition of information, Turing-completeness.

Complexity Theory: Time and space measures, hierarchy theorems, complexity classes P, NP, complete problems, P versus NP conjecture, provably hard problems.