CS 360: Automata and Formal Languages

Course Syllabus, Fall 2013

Contents

1 Course description

2 Course Catalog description

Basic theoretical concepts are introduced, including finite state automata, regular expressions, context-free grammars, pushdown automata, Turing machines, recursively enumerable languages, the halting problem, and Church-Turing thesis. (PR: CS 300)

3 Instructor information and office hours

- Dr. V.N. Gudivada, Gullickson Hall Room 207, Phone: 304-696-5452. Please use Blackboard email for all communication related to this course.
- Course meets on TuTh 2:00 PM – 3:15 PM in GH 211.
- Office hours:
  - Monday: 12:00 Noon - 2.00 PM
  - Tuesday: 10:00 AM - 11:00 AM and 1:00 PM - 2:.00 PM
  - Thursday: 10:00 AM - 11:00 AM and 1.00 PM - 2.00 PM
  - Other times by appointment

4 Prerequisites

- CS 300 (Programming Languages)

5 Course topics

- Languages and strings
- Computation
- Finite state automata and regular languages
- Pushdown automata and context-free languages
- Turing machines and undecidability
- Computational complexity

6 BSCS degree program goals

a. an ability to apply knowledge of computing and mathematics appropriate to the discipline, including the ability to analyze and evaluate performance tradeoffs of algorithms, data structures, and hardware solutions;

b. an ability to analyze a problem, and identify and define the computing requirements appropriate to its solution;

c. an ability to design, implement, and evaluate a computer-based system, process, component, or program, including software systems of varying complexity, to meet desired needs;

d. an ability to function effectively on teams to accomplish a common goal;

e. an understanding of professional, ethical, legal, security, and social issues and responsibilities;

f. an ability to communicate effectively, both written and oral, with a range of audiences;

g. an ability to analyze the local and global impact of computing on individuals, organizations, and society;

h. a recognition of the need for and an ability to engage in continuing professional development;

i. an ability to use current techniques, skills, and tools necessary for computing practice, including the ability of expressing algorithms in at least two of the most important computer languages currently in use in academia and industry.

7 Course goals and relationship to BSCS program goals

After successful completion of this course, students should be able to:

① Explain the fundamental principles and theory that underlie modern computing (contributes to degree program goal a).

② Use the theory in solving various computing problems (contributes to degree program goal i).

8 Course timeline

- Week 1
  - Languages and strings, language hierarchy, computation
- Weeks 2 – 4
  - Finite state machines and regular languages
- Weeks 5 – 8
  - Pushdown automata and context-free languages
  - Midterm exam
- Weeks 9 - 11
  - Turing machines and undecidability
- Weeks 12 - 14
  - Computational complexity
  - Final exam (12 December 2013, 12.15 PM - 2.45 PM)

9 Instructional materials

Required textbook:

[1] Elaine Rich. *Automata, Computability, and Complexity*. Prentice Hall, 2008.

Lecture materials for other languages will be drawn from various resources listed in the Bibliography (section 14). Lecture slides and handouts will be made available on muOnline. You should also consult Web resources listed in section 9.

Web resources

- http://csunplugged.org/finite-state-automata
- TIOBE Programming Community Index. `http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html`

10 Course assessment

The course assessment components include: homework (30%), project (20%), midterm exam (20%), and final exam (30%). Course grade is awarded based on the following scheme:

Score	Letter Grade
>= 90	A
>= 80 & < 90	B
>= 70 & < 80	C
>= 60 & < 70	D
< 60	F

11 Classroom etiquette

- Students are expected to show up for class on time and participate in the class constructively.
- Attendance will be taken at the beginning of the class. However, attendance has no bearing on students' course grade. Students are not penalized for not attending classes. However, they are responsible for turning in assignments and projects on time and taking exams as scheduled.
- During the class and exams, students should turn off all types of electronic gadgets including mobile/smart phones, iPhones, iPods, blackberries, laptops. These devices must remain out of sight for the entire duration of the class. Students who violate this policy will be asked to leave the classroom.
- No internet browsing is allowed in the class.

12 muOnline

It is important that students visit muOnline regularly for up-to-date information about the course. muOnline hosts all the course materials including assignments, handouts, lecture notes, and reading materials.

13 Policy for students with disabilities

Marshall University is committed to equal opportunity in education for all students, including those with physical, learning and psychological disabilities. University policy states that it is the responsibility of students with disabilities to contact the Office of Disabled Student Services (DSS) in Prichard Hall 117, phone 304-696-2271, to provide documentation of their disability. Following this, the DSS Coordinator will send a letter to each of the student's instructors outlining the academic accommodation he/she will need to ensure equality in classroom experiences, outside assignment, testing and grading. The instructor and student will meet to discuss how the accommodation(s) requested will be provided. For more information, please visit http://www.marshall.edu/disabled or contact Disabled Student Services Office at Prichard Hall 117, phone 304-696-2271.

14 Bibliography

[1] TIOBE Software. *TIOBE Programming Community Index*. `http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html`. 2013.

[2] Michael Sipser. *Introduction to the Theory of Computation*. Third. Cengage Learning, 2012.

[3] Peter Linz. *An Introduction to Formal Languages and Automata*. Fifth. Jones & Bartlett Learning, 2011.

[4] Michael L. Scott. *Programming Language Pragmatics*. Third. Morgan Kaufmann, 2009.

[5] Daniel P. Friedman and Mitchell Wand. *Essentials of Programming Languages*. Third. The MIT Press, 2008.

[6] Elaine Rich. *Automata, Computability, and Complexity*. Prentice Hall, 2008.

[7] M. Ben-Ari. *Understanding Programming Languages*. `http://www.freetechbooks.com/understanding-programming-languages-t657.html`, 2006.

[8] Susan H. Rodger and Thomas W. Finley. *JFLAP: An Interactive Formal Languages and Automata Package*. Jones & Bartlett Learning, 2006.

[9] Peter Van Roy and Seif Haridi. *Concepts, Techniques, And Models of Computer Programming*. The MIT Press, 2004.

[10] Harold Abelson and Gerald Jay Sussman. *Structure And Interpretation Of Computer Programs*. Second. The MIT Press, 1996.