CSE 322
Intro to Formal Models in CS
Course Organization

W. L. Ruzzo 26 Sep 07

Instructor:
Larry Ruzzo
554 CSE, 543-6298
ruzzo@cs.washington.edu

TAs:
Zizhen Yao
yzizhen@cs.washington.edu

Class Time & Place: MWF, 1:30–2:20, EEB 037

Prerequisites: CSci 321

Text: Sipser, *Intro. to the Theory of Computation*, PWS Publishing, 1997.

Grading: There will be written homework assignments (about weekly), a midterm, and a final. Relative weights *approximately* 50%, 20%, 30%, give or take 10%. Some of the homework may include smallish programming projects.

Course Web: Check the web periodically for announcements, homework clarifications, etc.:
`http://www.cs.washington.edu/education/courses/322/`

Course Email List: See course web for subscription information. Email is also logged to the course web.

Reading Assignments: Keep up. I plan march through the book pretty much in sequence. We'll cover most of Ch. 1 & 2, and perhaps dip into 3 & 4. Read all of Ch. 0 (321 review), and start Ch. 1 by next class.

Catalog Description: Finite automata and regular expressions; context free grammars and pushdown automata; nondeterminism; Turing machines and the halting problem. Emphasis on understanding models and their applications and on rigorous use of basic techniques of analysis. Induction proofs, simulation, diagonalization and reduction arguments. CSci majors only.

Goals: The main goal of the course is to give students an ability to develop and rigorously reason about abstract formal models of computational devices, and an appreciation for the powers and limitations of such formalisms. An important secondary goal is to teach a body of facts about and techniques for studying "classical" models, such as finite automata and context-free grammars, having important applications in a variety of other areas of computer science, e.g. compilers and program specification.

Outline: Core material outlined below usually constitutes most of the course work. Some selection of optional material marked below or other topics fills the rest.

1. Alphabets, strings, languages; operations on them.
2. Ways of formally defining models; states, transitions, acceptance, etc.; nondeterminism.
3. Finite Automata and Regular Expressions (4-5 weeks).
   (a) Deterministic and non-deterministic FA.
   (b) ϵ-moves.
   (c) Regular expressions.
   (d) Right-, and left-linear grammars
   (e) Equivalence of all of these.
   (f) Pumping lemma.
   (g) Closure under $\cup, \cap, \cdot, ^*, \neg$.
   (h) Optional: two-way FA, transducers, other closure results, state minimization.
4. Context-Free Grammars and Pushdown Automata (4-5 weeks)
   (a) Grammars, derivations, derivation trees, ambiguity.
   (b) PDA's and DPDA's.
   (c) Equivalence of CFG's and PDA's.
   (d) Pumping Lemma.
   (e) Closure under $\cup, \cdot, ^*$; *non*-closure under $\cap$.
   (f) Introduction to parsing.
   (g) Optional: Grammar manipulations: useless rules, Chomsky & Greibach forms; Ogden's lemma; linear CFL's; variations on acceptance in PDA's; Cocke-Kasami-Younger Algorithm; closure under $\cap$-with-regular-set; other closure results.
5. Optional: Turing Machines and Decidability (1–2 weeks; this material is covered in more depth in 431)
   (a) Definitions.
   (b) Church-Turing Thesis.
   (c) Halting Problem.
   (d) Optional: variations on TM's, an undecidable grammar problem, e.g. CFG intersection $= \emptyset$, Post's correspondence problem.
6. Optional: general phrase-structure and context-sensitive grammars, Chomsky hierarchy.