Languages and Automata

What are the Big Ideas?

Wednesday, September 5, 2007

CS235 Languages and Automata

Department of Computer Science
Wellesley College

Why Take CS235?

1. It's required for CS majors.
2. It teaches some important Big Ideas.
3. It's a lead-in to Programming Languages (CS251) and Compilers (CS301)*, in terms of both theory and practice (scanning, parsing, ML).
4. This semester, we're trying out an approach to teaching the material that puts even more emphasis on hands-on experimentation with the concepts in ML.

* Hasn't been taught for awhile, but could be offered in 2008-09 if there's sufficient interest.

The Big Ideas

1. Models of Computation
2. Undecidability: Not Everything is Computable!
3. Mathematical Foundations for Computer Science
4. Cool Applications

Models of Computation: Regular Languages

E.g.: all binary strings consisting of all 1s or sequences of 10s (includes the empty string, written %)

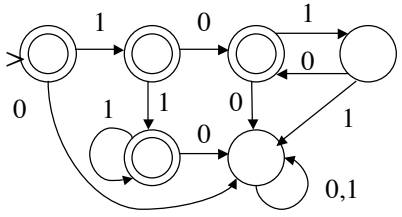
Regular Expression

$1^* \cup (10)^*$

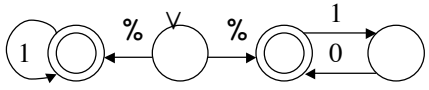
Right-Linear Grammar

S → A
S → B
A → %
A → 1A
B → %
B → 10B

Deterministic Finite Automaton



Nondeterministic Finite Automaton



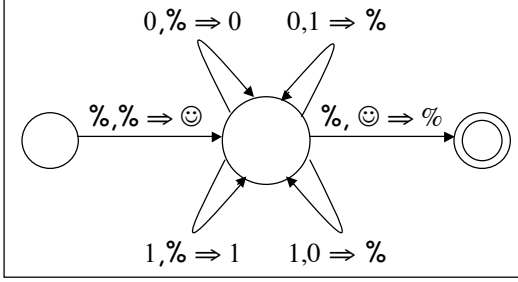
Models of Computation: Context-Free Languages

E.g.: all binary strings with an equal number of 0s and 1s

Context Free Grammar

S → %
S → SS
S → 0S1
S → 1S0

Nondeterministic Pushdown Automaton

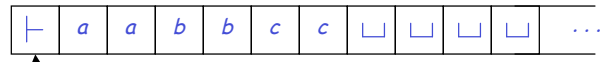


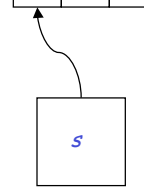
Q: Can we create a deterministic recognizer/parser for a CFG?

A: Yes, but can only do so efficiently for a restricted subclass of CFGs.

Models of Computation: Turing Machines

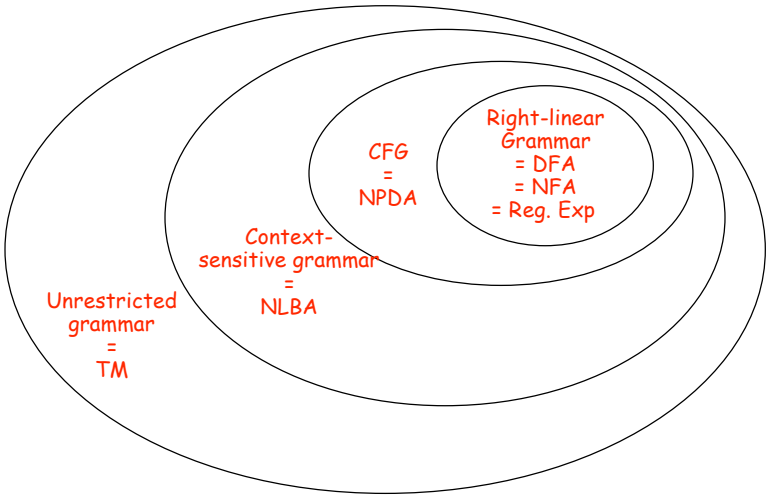
E.g.: all strings of the form $a^n b^n c^n$.





	$\vdash$	a	b	c	$\sqcup$	$\dashv$
s	$(s, \vdash, R)$	(s, a, R)	(q_1, b, R)	(q_2, c, R)	$(q_3, \dashv, L)$	–
q_1	–	$(r, -, -)$	(q_1, b, R)	(q_2, c, R)	$(q_3, \dashv, L)$	–
q_2	–	$(r, -, -)$	$(r, -, -)$	(q_2, c, R)	$(q_3, \dashv, L)$	–
q_3	$(t, -, -)$	$(r, -, -)$	$(r, -, -)$	$(q_4, \sqcup, L)$	$(q_3, \sqcup, L)$	–
q_4	$(r, -, -)$	$(r, -, -)$	$(q_5, \sqcup, L)$	(q_4, c, L)	$(q_4, \sqcup, L)$	–
q_5	$(r, -, -)$	$(q_6, \sqcup, L)$	(q_5, b, L)	–	$(q_5, \sqcup, L)$	–
q_6	$(q_7, \vdash, R)$	(q_6, a, L)	–	–	$(q_6, \sqcup, L)$	–
q_7	–	$(q_8, \sqcup, R)$	$(r, -, -)$	$(r, -, -)$	$(q_7, \sqcup, R)$	$(t, -, -)$
q_8	–	(q_8, a, R)	$(q_9, \sqcup, R)$	$(r, -, -)$	$(q_8, \sqcup, R)$	$(r, -, -)$
q_9	–	–	(q_9, b, R)	$(q_{10}, \sqcup, R)$	$(q_9, \sqcup, R)$	$(r, -, -)$
q_{10}	–	–	–	(q_{10}, c, R)	$(q_{10}, \sqcup, R)$	$(q_3, \dashv, L)$

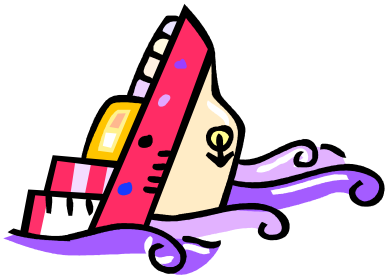
Models of Computation: The Chomsky Hierarchy



Undecidability: Not Everything is Computable!

The Halting Problem: It is impossible to write a program that always determines whether or not other programs halt.

Rice's Theorem: Any nontrivial question we might ask about programs in general cannot be answered by other programs.



Mathematical Foundations of Computer Science

Explain computation by using standard mathematical structures

- Sets
- Strings
- Tuples
- Relations
- Functions
- Trees
- Graphs

and proof techniques, including

- Proof by construction
- Proof by contradiction (including diagonalization)
- Proof by algebra
- Proof by picture
- Proof by induction

Cool Applications of the Material in this Course

- Using Unix tools like grep, sed, and awk.
- Writing virus detection scanners.
- Designing hardware (CS240).
- Programming language implementation: creating scanners and parsers using tools like Lex and Yacc.
- ML programming (appetizer for CS251, CS301).
- Knowing which problems *not* to solve.

Administrivia

- Syllabus, problem sets, office hours, etc. on the CS235 home page:

  http://cs.wellesley.edu/~cs235

- Weekly problem sets, typically due in class on Thursdays. I aim to grade them in a timely fashion. The next problem set will not be due until at least one day after the previous one has been returned!
- There are two exams:
  1. An in-class midterm on Thu. Oct. 18.
  2. A self-scheduled open-book&notes final exam.
- Some assignments will involve simple programming in ML, mostly as a "calculator". We'll use SMLNJ (similar to OCaml, used in CS251 and CS301) on Linux.
- There are no teaching assistants/tutors for CS235 this semester (but maybe you'll be one next year!) I will extend/adapt my office hours as necessary.

Textbook and Software

Alley Stoughton's Forlan project (Kansas State University)

http://people.cis.ksu.edu/~stough/forlan/

- Draft textbook
- Textbook slides
- Software = ML modules, Java programs
- Alley's KSU course: CIS 570

Other resources:

- Michael Sipser, Introduction to the Theory of Computation
- Dexter Kozen, Automata and Computability
- L.C. Paulson, ML for the Working Programmer
- Andrew Appel, Modern Compiler Implementation in ML

Collaboration Policies

- You *may* **talk** with anyone about any nonexam problem, but must write up any proofs/programs on your own.
- You *may not* look at anyone else's written proofs/programs.
- You *may not* look at any solutions from previous semesters of CS235.
- You are encouraged to explore materials for Alley's CIS 570, including problems and solutions from 2006 and 2007, but *not* any earlier semesters.
- You *may* find other information in textbooks and on the Internet, but must cite any source you use for solutions.