Welcome to the Stanford Automata Theory Course

Why Study Automata?

What the Course is About

Why Study Automata?

- A survey of Stanford grads 5 years out asked which of their courses did they use in their job.
- Basics like intro-programming took the top spots, of course.
- But among optional courses, CS154 stood remarkably high.
  - 3X the score for AI, for example.

How Could That Be?

- Regular expressions are used in many systems.
  - E.g., UNIX a.*b.
  - E.g., DTD's describe XML tags with a RE format like person (name, addr, child*).
- Finite automata model protocols, electronic circuits.

How? – (2)

- Context-free grammars are used to describe the syntax of essentially every programming language.
  - Not to forget their important role in describing natural languages.
- And DTD's taken as a whole, are really CFG's.

How? – (3)

- When developing solutions to real problems, we often confront the limitations of what software can do.
  - *Undecidable* things – no program whatever can do it.
  - *Intractable* things – there are programs, but no fast programs.
- Automata theory gives you the tools.

Other Good Stuff

- We'll learn how to deal formally with discrete systems.
  - Proofs: You never really prove a program correct, but you need to be thinking of why a tricky technique really works.
- We'll gain experience with abstract models and constructions.
  - Models layered software architectures.

Automata Theory – Gateway Drug

- This theory has attracted people of a mathematical bent to CS, to the betterment of all.
  - Ken Thompson – before UNIX was working on compiling regular expressions.
  - Jim Gray – thesis was automata theory before he got into database systems and made fundamental contributions there.

Course Outline

- Regular Languages and their descriptors:
  - Finite automata, nondeterministic finite automata, regular expressions.
  - Algorithms to decide questions about regular languages, e.g., is it empty?
  - Closure properties of regular languages.

Course Outline – (2)

- Context-free languages and their descriptors:
  - Context-free grammars, pushdown automata.
  - Decision and closure properties.

Course Outline – (3)

- Recursive and recursively enumerable languages.
  - Turing machines, decidability of problems.
  - The limit of what can be computed.
- Intractable problems.
  - Problems that (appear to) require exponential time.
  - NP-completeness and beyond.

Text (Not Required)

- Hopcroft, Motwani, Ullman, *Automata Theory, Languages, and Computation* 3rd Edition.
- Course covers essentially the entire book.