CS 172: Computability and Complexity

Introduction and Course Overview

Sanjit A. Seshia

EECS, UC Berkeley

Acknowledgments: L.von Ahn, L. Blum, M. Blum, R. Jhala

A Problem

Suppose $A \subseteq \{1, 2, \ldots, 2n\}$ with $|A| = n+1$

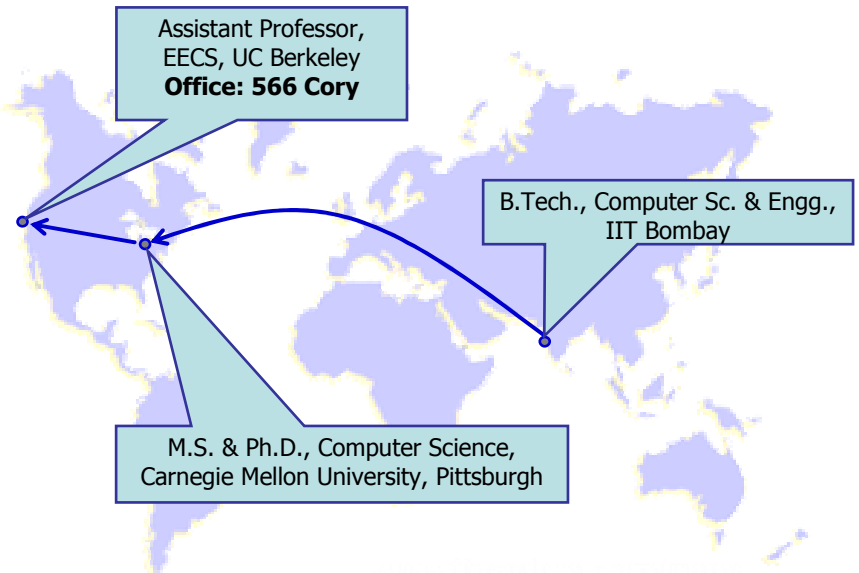
TRUE or FALSE: There are always two numbers in A such that one divides the other

(work out the answer with proof)

What we'll do today

- Introduction to course staff and students
- Motivation
- Course logistics
- Basics of proof and finite automata
- A quick survey at the end of class
- **Pick up**
  - **course information handout**
  - **survey form**

About Me



My Research

+

Theory

Computational Logic, Algorithms

Practice

CAD for VLSI, Computer Security, Embedded Systems, Dependability

Example: Fast **satisfiability (SAT)** solving used to build a better **virus/worm detector**

Class Introductions

- Introduction to Omid Etesami (your GSI)
- and the rest of you…

A 3-Question Course

- **Automata Theory:**
  What are the mathematical models of computation?
- **Computability Theory:**
  What problems can(not) computers solve?
- **Complexity Theory:**
  What makes some problems computationally hard and others easy?

More Succinctly

This course is about
models of computation
and the **limits of computation**

More Succinctly

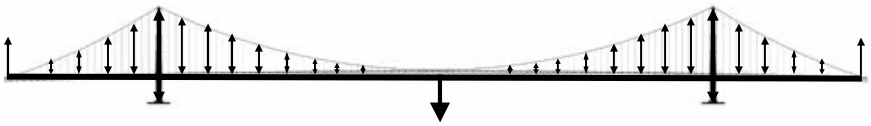
This course is about
models of computation
and the **limits of computation**

WHY SHOULD WE CARE?

Motivation for this Course

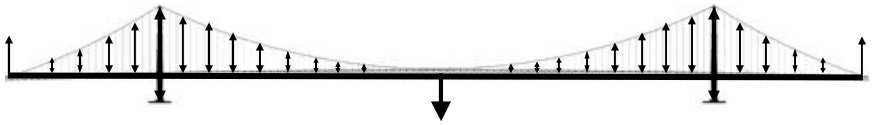
- Relevant for practical applications
- Foundation for theoretical work
- It's fun!

Model: Free Body Diagram



Why build a model?

Model: Free Body Diagram

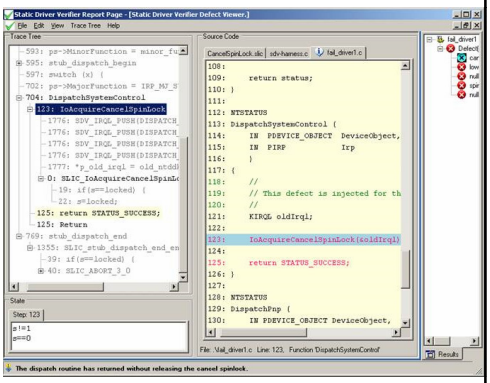


Analyzing a model helps us build reliable bridges, in a cost-effective & timely way

Why not do the same for computers?

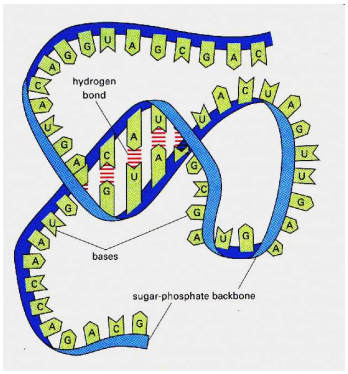
Programs and Pushdown Automata

- Sequential (single-threaded) programs can be mathematically modeled as "*pushdown automata*"
  - E.g., Microsoft's Static Driver Verifier searches for bugs in device driver code after creating a pushdown automaton model of the program



Programs and Pushdown Automata

- Variants of pushdown automata are even used in modeling structure of RNA!
  - This course is not just about computers, it's about *computation*!



Analyzing correctness of computers is hard!

Limits of computation

Windows

```
An exception  06 has occured at 0028:C11B3ADC in VxD DiskTSD(03) +
00001660.  This was called from 0028:C11B40C8 in VxD voltrack(04) +
00000000.  It may be possible to continue normally.

*  Press any key to attempt to continue.
*  Press CTRL+ALT+RESET to restart your computer.  You will
   lose any unsaved information in all applications.

                Press any key to continue
```

Theoretically Useful

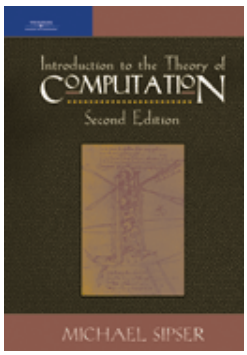
- Mathematical models of computation give us systematic ways to think about computers
  - Independent of how they are implemented
  - So the results long outlast the specific machines (anybody remember VAX machines? 486-based PCs running Windows 95?)

Course Logistics

- The course webpage is the definitive source of information

  www.eecs.berkeley.edu/~sseshia/172/

  We'll also use bSpace
- 1 required textbook
- 3 other references
- All 4 under reserve in Engineering library



Grading & Policies

- 10 homeworks: 30% weightage
- 2 mid-terms: 20% each
- 1 final: 30%

- Homeworks assigned approx. weekly, due in class/drop-box
  - First homework out next Monday
  - No late homeworks
  - Must list collaborators and references
- Important! See webpage for course policies

Back to the Problem

Suppose $A \subseteq \{1, 2, \ldots, 2n\}$ with $|A| = n+1$

TRUE or FALSE: There are always two numbers in A such that one divides the other

TRUE

Got Proof?

- Hint #1: the Pigeonhole Principle

  If you put 4 pigeons in 3 holes, at least one hole will contain > 1 pigeon

- Hint #2:

  Every integer a can be written as $a = 2^k m$, where m is an odd number

Suppose $A \subseteq \{1, 2, \ldots, 2n\}$ with $|A| = n+1$

Write every number in A as $a = 2^k m$, where m is an odd number between 1 and 2n-1

How many odd numbers in {1, …, 2n-1}? n

Since $|A| = n+1$, there must be two numbers in A with the same odd part (pigeonhole)

Say a_1 and a_2 have the same odd part m. Then $a_1 = 2^i m$ and $a_2 = 2^j m$, so one must divide the other

This class will emphasize PROOFS

A good proof should be:

Easy to understand

Correct

3-Level Proofs

- Recommendation: Organize your proofs into 3 levels:
  1. A one-phrase HINT
     - E.g., “proof by contradiction”, “the pigeonhole principle”, etc.
  2. A brief one-paragraph PROOF SKETCH
  3. The complete proof

This will help you with the proof and also to get partial credit for your approach, even if you get some details wrong

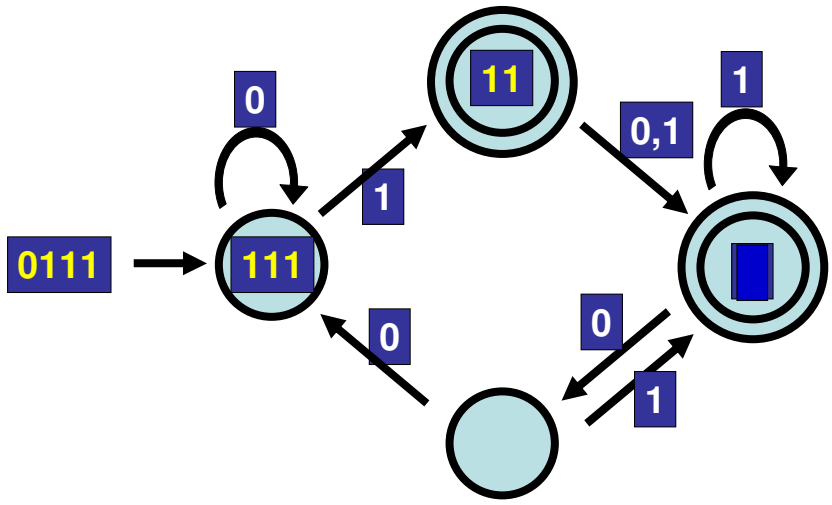
Double Standards?

- During lectures, my proofs might only have the first two levels
  - There's a lot to cover in each class
  - I have to keep you awake
  - I KNOW that my proofs are correct ☺

More on Proofs & Background

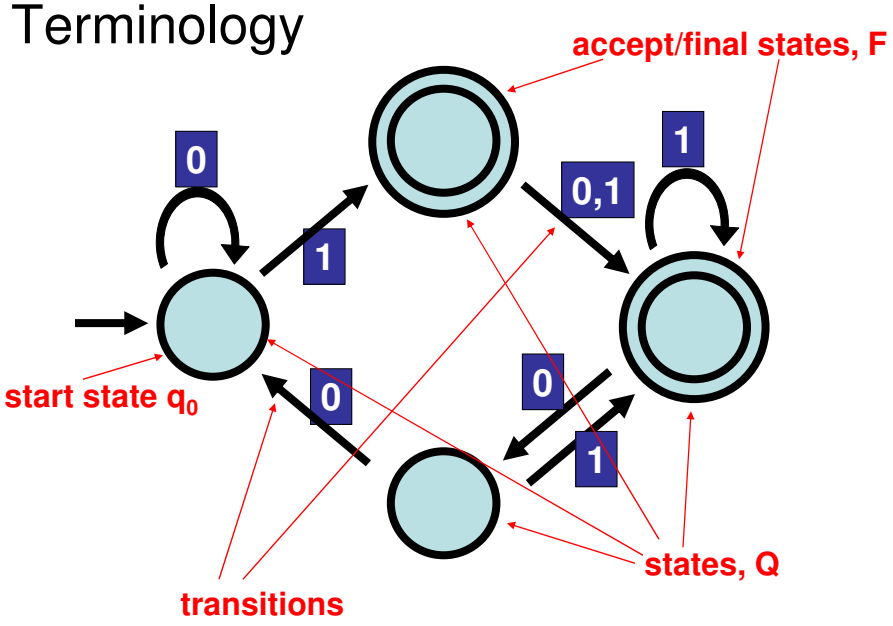
- Read Sipser, Chapter 0

Introduction to Finite Automata (Finite State Machines)



The machine accepts a string if the process ends in a double circle

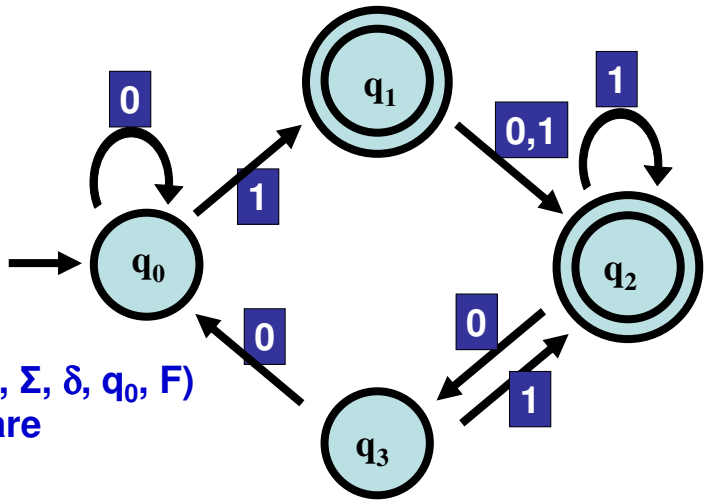
Terminology



A finite automaton is a 5-tuple $M = (Q, \Sigma, \delta, q_0, F)$

- Q is the set of states
- Σ is the alphabet
- $\delta : Q \times \Sigma \rightarrow Q$ is the transition function
- $q_0 \in Q$ is the start state
- $F \subseteq Q$ is the set of accept states

L(M) = the language of machine M
= set of all strings machine M accepts



M = (Q, Σ, δ, q_0, F)
What are
1. Q
2. F
3. Σ
4. δ

Applications of Finite Automata

- String matching
- Modeling sequential digital circuits
- Embedded controllers in your car
- …

Next Steps

- Read Sipser 1.1 in preparation for next lecture