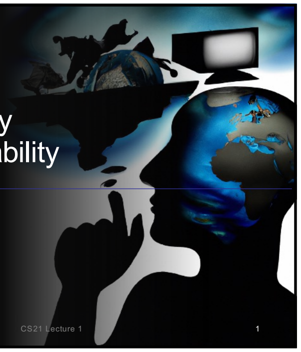
CS21 Decidability and Tractability

Lecture 1
January 3, 2024



1

Outline

- administrative stuff
- motivation and overview of the course
- problems and languages
- Finite Automata

2

Administrative Stuff

- Text: Introduction to the Theory of Computation – 3rd Edition by Mike Sipser
- Lectures self-contained
- Weekly homework
  - collaboration in small groups encouraged
  - separate write-ups (clarity counts)
- Midterm and final
  - indistinguishable from homework except cumulative, no collaboration allowed

3

Administrative Stuff

- No programming in this course
- Things I assume you are familiar with:
  - programming and basic algorithms
  - asymptotic notation “big-oh”
  - sets, graphs
  - proofs, especially induction proofs

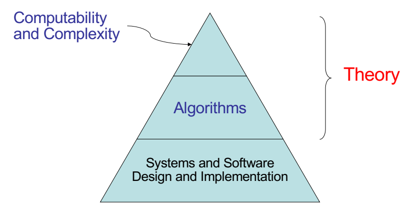
4

Motivation/Overview

- This course: introduction to
  Theory of Computation
  - what does it mean?
  - why do we care?
  - what will this course cover?

5

Motivation/Overview



6

Motivation/Overview

- At the heart of programs lie algorithms
- To study algorithms we must be able to speak ***mathematically*** about:
  - computational problems
  - computers
  - algorithms

7

Motivation/Overview

- You might imagine that in principle
  - for each problem we would have an algorithm
  - the algorithm would be the fastest possible
    (requires proof that no others are faster)

8

Motivation/Overview

- Our world (fortunately) is more interesting:
  - not all problems have algorithms (we will prove this)
  - for many problems we know embarrassingly little about what the fastest algorithm is
    - multiplying two integers
    - factoring an integer into primes
    - determining shortest tour of given n cities
  - for certain problems, fast algorithms would change the world (we will see this)

9

Motivation/Overview

Part One:

computational problems, models of computation, characterizations of the problems they solve, and limits on their power

- Finite Automata and Regular Languages
- Pushdown Automata and Context Free Grammars

10

Motivation/Overview

Part Two:

Turing Machines, and limits on their power (undecidability), reductions between problems

Part Three:

complexity classes P and NP, NP-completeness, limits of efficient computation

11

Main Points of Course

(un)-decidability

Some problems have no algorithms!

(in)-tractability

Many problems that we'd like to solve have no efficient algorithms!

(no one knows how to prove this yet…)

12

What is a problem?

- Some examples:
  - given n integers, produce a sorted list
  - given a graph and nodes s and t, find the (first) shortest path from s to t
  - given an integer, find its prime factors
- problem associates each input to an output
- input and output are strings over a finite *alphabet* Σ

13

What is a problem?

- A problem is a function:

$$f:\Sigma^* \rightarrow \Sigma^*$$

- Simple. Can we make it simpler?
- Yes. Decision problems:

$$f:\Sigma^* \rightarrow \{accept, reject\}$$

- Does this still capture our notion of problem, or is it too restrictive?

14

What is a problem?

- Example: factoring:
  - given an integer m, find its prime factors

$$f_{factor}: \{0,1\}^* \rightarrow \{0,1\}^*$$

- Decision version:
  - given 2 integers m,k, accept iff m has a prime factor p < k
- Can use one to solve the other and vice versa. True in general (homework).

15

What is a problem?

- For most of this course, a problem is a decision problem:

$$f:\Sigma^* \rightarrow \{accept, reject\}$$

- Equivalent notion: language

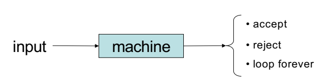
$$L \subseteq \Sigma^*$$

the set of strings that map to "accept"

- Example: L = set of pairs (m,k) for which m has a prime factor p < k

16

What is computation?



- the set of strings that lead to "accept" is the language recognized by this machine
- if every other string leads to "reject", then this language is decided by the machine

17

Terminology

- finite alphabet Σ : a set of symbols
- language $L \subseteq \Sigma^*$: subset of strings over Σ
- a machine takes an input string and either
  - accepts, rejects, or
  - loops forever
- a machine recognizes the set of strings that lead to accept
- a machine decides a language L if it accepts $x \in L$ and rejects $x \notin L$

18