

CS21 Decidability and Tractability

Lecture 1
January 7, 2008

January 7, 2008

CS21 Lecture 1

1

Outline

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

January 7, 2008

CS21 Lecture 1

2

Administrative Stuff

- **Text:** Introduction to the Theory of Computation – 2nd Edition by Mike Sipser
- Lectures self-contained
- **Weekly homework**
 - collaboration in groups of 2-3 encouraged
 - separate write-ups (clarity counts)
- **Midterm and Final**
 - indistinguishable from homework except cumulative, no collaboration allowed

January 7, 2008

CS21 Lecture 1

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

January 7, 2008

CS21 Lecture 1

4

Motivation/Overview

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

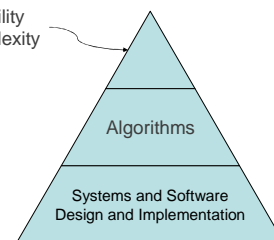
January 7, 2008

CS21 Lecture 1

5

Motivation/Overview

Computability
and Complexity



} Theory

January 7, 2008

CS21 Lecture 1

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

January 7, 2008

CS21 Lecture 1

7

Motivation/Overview

- In a perfect world
 - for each problem we would have an algorithm
 - the algorithm would be the fastest possible
 - how could we be sure we have the fastest one?
 - requires proof that no others are faster

What would CS look like in this world?

January 7, 2008

CS21 Lecture 1

8

Motivation/Overview

- Our world (fortunately) is not so perfect:
 - 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)

January 7, 2008

CS21 Lecture 1

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

January 7, 2008

CS21 Lecture 1

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

January 7, 2008

CS21 Lecture 1

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...)

January 7, 2008

CS21 Lecture 1

12

What is a problem?

- Some examples:
 - given n integers, produce a sorted list
 - given a graph and nodes s and t , find the 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

January 7, 2008

CS21 Lecture 1

13

What is a problem?

- A problem is a function:
 $f: \Sigma^* \rightarrow \Sigma^*$
- Simple. Can we make it simpler?
- Yes. Decision problems:
 $f: \Sigma^* \rightarrow \{\text{accept}, \text{reject}\}$
- Does this still capture our notion of problem, or is it too restrictive?

January 7, 2008

CS21 Lecture 1

14

What is a problem?

- Example: factoring:
 - given an integer m , find its prime factors
 $f_{\text{factor}}: \{0,1\}^* \rightarrow \{0,1\}^*$
- 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).

January 7, 2008

CS21 Lecture 1

15

What is a problem?

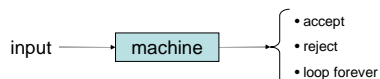
- For most of this course, a problem is a decision problem:
 $f: \Sigma^* \rightarrow \{\text{accept}, \text{reject}\}$
- Equivalent notion: language
 $L \subseteq \Sigma^*$
the set of strings that map to “accept”
- Example: $L =$ set of pairs (m, k) for which m has a prime factor $p < k$

January 7, 2008

CS21 Lecture 1

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

January 7, 2008

CS21 Lecture 1

17

Terminology

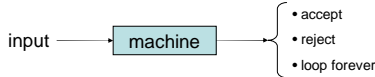
- finite alphabet Σ : a set of symbols
- language $L \subseteq \Sigma^*$: subset of strings over Σ
- a machine takes an input string and either
 - accepts, rejects, or
 - loops forever
- a machine recognizes the set of strings that lead to accept
- a machine decides a language L if it accepts $x \in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

January 7, 2008

CS21 Lecture 1

18

What goes inside the box?



- We want the simplest mathematical formalization of computation possible.
- Strategy:
 - endow box with a feature of computation
 - try to characterize the languages decided
 - identify language we “know” real computers can decide that machine cannot
 - add new feature to overcome limits

January 7, 2008

CS21 Lecture 1

19

Finite Automata

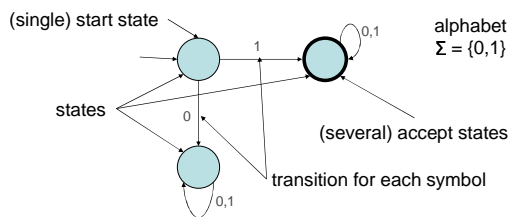
- simple model of computation
- reads input from left to right, one symbol at a time
- maintains state: information about what seen so far (“memory”)
 - finite automaton has finite # of states: cannot remember more things for longer inputs
- 2 ways to describe: by diagram, or formally

January 7, 2008

CS21 Lecture 1

20

FA diagrams



- read input one symbol at a time; follow arrows; accept if end in accept state

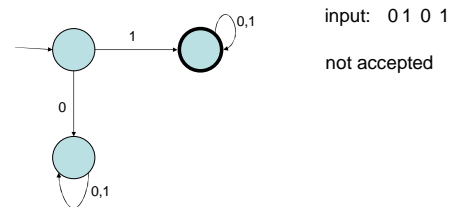
January 7, 2008

CS21 Lecture 1

21

FA operation

- Example of FA operation:



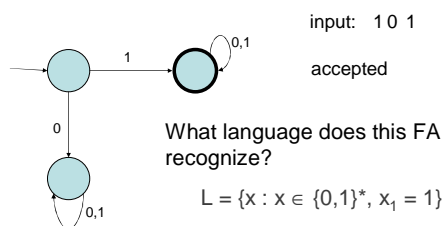
January 7, 2008

CS21 Lecture 1

22

FA operation

- Example of FA operation:



January 7, 2008

CS21 Lecture 1

23

Example FA



- What language does this FA recognize?
 - $L = \{x : x \in \{0,1\}^*, x \text{ has even \# of 1s}\}$
- illustrates fundamental feature/limitation of FA:
 - “tiny” memory
 - in this example only “remembers” 1 bit of info.

January 7, 2008

CS21 Lecture 1

24