What is computation?

Overview

• What is the Course About?
• Chapter Introduction
• Finite Automata
• Formal Definition
**Expected Outcomes**

- Be able to do symbolic proofs
- Understand that there are limits to computation
- Understand the power of using simpler models of computation
- Understand why heuristics might be needed for some problems
- In practical terms,
  - Just as you might know how computation happens at gate level, operating system level, and programming level, it's good to understand it from a formal standpoint as well
- Different models, computational complexity
- Learn about the theory of computation

**Goals of Course**

- Automata
  - What is a formal model of computing?
  - Are there different models, with different power?
- Computability
  - Are there some problems that are not possible to compute?
  - What is a formal model of computation?
- Complexity
  - How much time or space do problems take?
  - Can we separate different classes of problems?
Homework

- Homework is due Friday at 11:59pm.
- If needed, you can email me with a request for an extension.
- Homework must be emailed to me heemanp@ohsu.edu.

- Attach a single pdf or word document.
- Homework should be typeset.
- Can include pictures of hand-drawn solutions taken with cellphone.
- Use subject line: CS533 HW1.

Logistics

- Grading:
  - 60% homework
  - 20% midterm
  - 20% final

- Course website: csli.ohsu.edu/~heeman/cs533
If you use outside resources, document that as well.

- Nothing in, nothing out; wait at least an hour between
- But, after you have come up with a solution, redo it on your own
- Document how you collaborated; versus changed what the question meant
- Worked on entire question together, versus changed what the question meant
- You can do the homeworks with your colleagues

Critique

- Answer key will be emailed to you when you submit your answers
- You will have until Sunday at 11:59pm to submit a critique
- Explain why you did wrong, and why you made that mistake
- Earn back up to half of the marks that you lost
- Show your work, and understand answer key and subject line: CSS33 HW1 Critique
- See sample homework for how to format this
- Should show that you reviewed and understood answer key and really good explanations might even get more
- Worth up to half the marks that you lost
- Use subject line: CSS33 HW1 Critique

Academic Integrity
Chapter 1: Regular Languages

• What is a computer?
  - Definition gets into all sorts of complicated issues
  - Types of memory
  - How it is addressed
  - Types of operations supported

• Let's focus on an idealized computer

Formal Definition

Finite Automata

Chapter Introduction

What is the Course About?
Automatic Door Problem

What can a machine with extremely limited amount of memory do?

If someone is on the front pad, it should keep the door open until the person is off of the rear pad so that the automatic door does not knock the person over.

If someone steps on the front pad while someone is still on the rear pad, it should not open the door, otherwise it might knock over the person on the rear pad.

Say we want a controller for an automatic door.

Overview

What is the Course About?

Chapter Introduction

Formal Definition

Finite Automata
Many problems can be solved by an FA—e.g., Elevator controller, Thermostat, Dishwasher, ...

So, need to be more formal about this model
- is not capable of
- is capable of

So, let’s understand what this model can solve by an FA. Why use something more complicated?
- If problem solvable by an FA, why use something more complicated?
- e. Elevator controller, Thermostat, Dishwasher ...

Finite Automata

- Transition Table:
- On sensor input, it might traverse to the other state
- Input signal
- What will controller do if started in closed and receives neither, open, neither, neither, both, front, rear, both, neither, rear,

Finite Automata

A Finite Automata
Still need a formal definition

- Need a precise definition for doing proofs about its power.
  - Can there be several transitions from a state for the same character?
  - Must there be a transition for every character from each state?
  - Must there be a reject state? Can there be several?
  - Must there be an accept state? Can there be several?

- But we still need a formal definition.

State diagram gives us a good intuition about what an FA is.
A finite automaton is a 5-tuple \((Q, \Sigma, \delta, q_0, F)\), where

1. \(Q\) is a finite set called the states
2. \(\Sigma\) is a finite set called the alphabet
3. \(\delta: Q \times \Sigma \rightarrow Q\) is the transition function
4. \(q_0 \in Q\) is the start state
5. \(F \subseteq Q\) is the set of accepting states
If $A$ is the set of all strings that machine $M$ accepts, we say that $A$ is the language of machine $M$. We write $L(M) = A$.

We reserve the word accepts to talk about individual strings: $M$ accepts $w$.

If the machine accepts no strings, it still recognizes one language: $\emptyset$.

For our example:

Let $A = \{ w \mid w$ contains at least one 1 and an even number of 0s following the last 1\}$.

$L(M_1) = A$ or $M_1$ recognizes $A$.

If $A$ is the set of all strings that machine $M$ accepts, $A$.

Example

What is the formal definition of $M_1$?
Yet Another Example

Definition of $M_3$:

Language of $M_3$:

What is the relationship between $M_2$ and $M_3$?

Another Example

Definition of $M_2$:

Language of $M_2$:
Formal Definition of Computation

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a finite automaton. Let $w = w_1w_2...w_n$ be a string over $\Sigma$. We say that $M$ accepts $w$ if there is a sequence of states $r_0, ..., r_n$ such that

1. $r_0 = q_0$
2. $r_{i+1} = \delta(r_i, w_{i+1})$ for $i = 0, ..., n-1$
3. $r_n \in F$

Back to language of an FA

- We say that $M$ recognizes language $A$ if $A = \{ w | M$ accepts $w \}$

And Another

- Design a FA that takes a sequence of digits (0-5)
  - Accepts if sum of digits is evenly divisible by 3
  - Design a FA that takes a sequence of digits (0-5)
Designing Finite Automata

- Think of what states you need to represent the important part of what you have seen in the input so far.
- Think of how you can transition between states.
- Examples:
  - Language of all strings with an odd number of 1s
  - Language in which all strings contain the substring 01
  - Language of all strings with an odd number of 1s

Regular Language

Definition:

A language is called a 

Definition: A language is called a **regular language** if some finite automaton recognizes it. A language is called a **regular language** if some finite automaton recognizes it. Note that a language defines what the automaton accepts, so what it answers 'accept' to defines the computation that the automaton does. Note that a language defines what the automaton accepts, so what it answers 'accept' to defines the computation that the automaton does.