CS310 : Automata Theory 2019

Lecture 1: Why automata theory?

Instructor: Ashutosh Gupta

IITB, India

Compile date: 2019-01-03

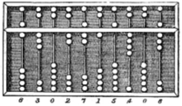
Original computer

(all pictures wikipedia)

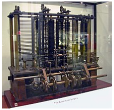
People are sitting around in a room and computing!

There are things that can also compute

Meanwhile, humans have been developing tools that can compute.



abacus



Mechanical calculators

(called analytical engines)

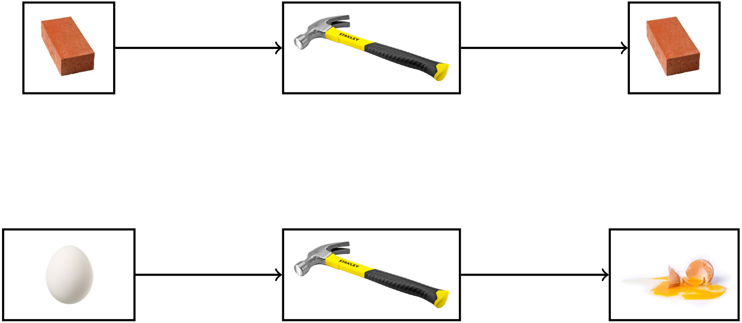
Electomechincal machines

Modern computers

When do we call something a computer?

Is this a computer?

Hammer computes strength of the input objects



Input Processing Output

Computer is the orgnaization in the material

A computer is organized as follows.

Input ⟶ Processing ⟶ Output

Without the organization, the computer is only metal and plastic.

The above characterization is not specific enough to do further analysis.

- When some action or thing becomes an input?
- What happens in processing?
- When some event or thing becomes an output?

For better understanding, let us first see some examples of computing!

Simple program : Hello user!

"I" →

```
int main(int argc, char *argv[])
{
 if( argc > 1 )
  printf("Hello %s\n",argv[1]);
 return 0;
}
```

→ "Hello I!"

One input and one output!

Reactive system : flight control

Control software

Sensors

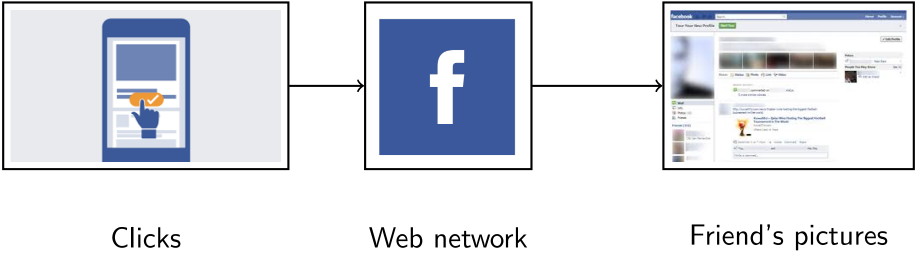
Actuators

Sensors are read in every cycle and actuators are set in every cycle.

Controllers do not terminate.

Commentary: Controllers do terminate when the airplane is off. But, they do not terminate on their own.

Social media : facebook



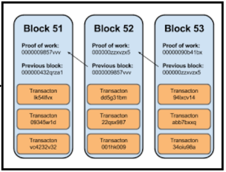
Distributed system involving billions of devices!

Difficult concurrency, privacy, and reliability issues.

Secure computing: Bitcoin

Transactions

Bitcoin



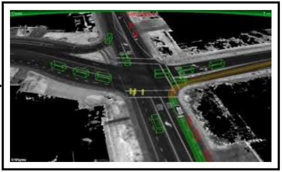
Blockchain ledger

Distributed system involving untrusted devices attempting trusted computing!

Difficult trust, integrity, and reliability issues.

Artificial intelligence: self driving cars

→ Drive software →



Surrounding video

Drive decision

Involves image processing, machine learning, and real-time systems!

Next frontier technology!

What is computing?

A diverse range of things are called computing

We need a model that covers all of them like
physics models the physical world.

The study of the modelling is called automata theory.

Topic 1.1

Automata theory: the course

Teachers

This course will be co-taught by Ashutosh Gupta and S. Akshay

Ashutosh will cover first half and Akshay will cover the second half.

Evaluation

- Quizzes : 30% (4)
- Midterm : 25% (2 hour)
- Final : 40% (3 min)
- Attendance : 5% (random+at the door)

Subject to minor changes!

Website

For further information

`https://www.cse.iitb.ac.in/~akg/courses/2019-cs310/`

All the problem sheets and slides will be posted at the website.

Please carefully read the course rules at the website

TAs and tutorial sessions

We will do three tutorial sessions before midterm.

TAs will contact you via email for the planning for tutorials.

Topic 1.2

Automata theory

Automata theory

Let us start modelling parts of computers

- inputs
- processing
- outputs

Now onward we will refer to the computing devices as automata.

Commentary: If the word is not familiar to you, automata is plural and automaton is singular.

What is Input?

- Not each interaction with automata is an input
- We need to specify the interactions that are inputs
- Furthermore, we need to say that the inputs are from
  - a domain, i.e., the set of possibilities and
  - a means of collecting the possibilities, i.e, tuples, sets, or sequences.

Example 1.1

Consider a human. Let there be only two possible inputs eat or run.

So a typical day of the human will be like

eat run run eat eat .....

Unbounded inputs : word

- The domain is called alphabet usually denoted by Σ
  - e. g., $\Sigma = \{eat, run\}$
- The elements of the alphabet are called letters.
- Inputs are sequences of letters, usually called words (or string)
  - e. g., word *eat run run eat eat* is element of Σ^*

What is this superscript *?

Notation alert : Σ^n

Σ^n is the set of strings of length n.
Formally,

- $\Sigma^1 \triangleq \Sigma$
- $\Sigma^2 \triangleq \Sigma \times \Sigma$
- $\vdots$
- $\Sigma^n \triangleq \underbrace{\Sigma \times \cdots \times \Sigma}_{n \text{ times}}$

Example 1.2

Let $\Sigma = \{a, b, c\}$.

When there is no ambiguity we drop spaces between letters in a word

- $a \in \Sigma^1$
- $ac \in \Sigma^2$
- $cac \in \Sigma^3$
- $abccba \in \Sigma^6$

Exercise 1.1

What is the natural definition of Σ^0*?*

Notation alert : empty string ϵ

Σ^0 is set of strings of length 0.

There is only one single empty string, denoted ϵ.

Therefore,

$$\Sigma^0 \triangleq \{\epsilon\}.$$

Notation alert : Σ^* and Σ^+

Σ^* consists of all strings of finite lengths.

$$\Sigma^* \triangleq \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \dots \qquad \Sigma^+ \triangleq \Sigma^* - \Sigma^0$$

Σ^+ does not have the empty string.

Example 1.3

Let $\Sigma = \{a, b, c\}$.

$$\{ab, cac, a, \epsilon, abcabc\} \subseteq \Sigma^*$$

$$\{ab, cac, a, abcabc\} \subseteq \Sigma^+$$

Exercise 1.2

a. What is $\Sigma^* - \Sigma^+$*?*

b. What is $\Sigma^+ - \Sigma^*$*?*

Modelling processing : states

In our modelling, we suppose automata

- process one letter in a word at a time and
- maintain some information about the input seen so far.

We refer to *the maintained information* as state.

We typically denote the set of all possible states of an automaton as Q.

Example 1.4

States of a human $Q = \{thin, fit, fat\}$

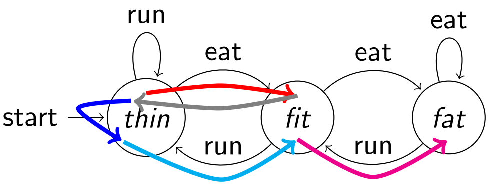
Modelling processing : transitions

As automaton reads a word, it moves from one state to the another state according to the next letter in the word.The movements are called transitions.

Naturally, we need to identify the starting/initial state.

Example 1.5

Let $\Sigma = \{eat, run\}$ *and* $Q = \{thin, fit, fat\}$

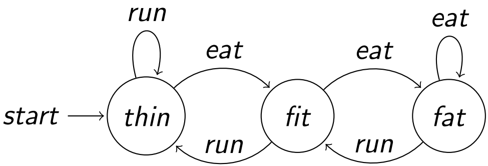


Consider word $w =$ *eat run run eat eat*

Commentary: The graph with states is also called “transition system” or “state machines”.

Where is the automaton?

Exercise 1.3



Where is the above automaton after reading the following word?

- *eat eat eat eat*
- *run run run run*
- *eat run eat run eat*
- *eat* (*eat run*)*

Commentary: * on a string is a new notation. We shall study in detail.

Modelling output : single bit

Theorists say, "every computing problem can be reduced to yes/no question!"

- Is fifty divided by five equals to ten?
- Is that object an apple?
- .....

Later half of the course will make it clear why?

For simpler understanding, we may assume that we are only considering automata that either accept or reject the input.

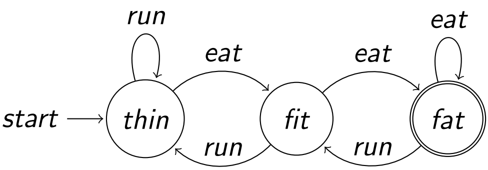
Modelling output: accepting states

In the automaton, we designate some especial states to be accepting states.

If an input word moves the automaton to an accepting state it is accepted. Otherwise, rejected.

Example 1.6

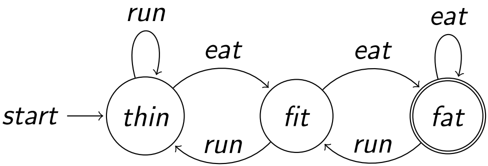
In the following automaton fat is the accepting state. Drawn with double circle.



Word eat eat is accepted by the automaton and word eat run is rejected.

Is the input accepted?

Exercise 1.4



Which of the following words accepted by the above automaton?

- *eat eat eat eat*
- *run run run run*
- *eat run eat run eat*
- *eat* $($*eat run*$)^*$ *eat*

Automaton

Looks like a very simple model!

Can it model every kind of computing?

Let us see a means for defining kinds of computing.

Automaton defines a set of words

Recall, automaton accepts or rejects words.

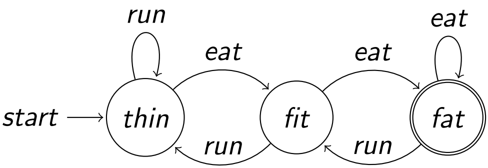
We can collect all the words that are accepted by an automaton.

The sets form one of the fundamental ideas for modelling computing.

Languages

Example: accepted words

Example 1.7



The following is the set of words that are accepted by the above automaton.

$$\{eat\ eat, run\ eat\ eat, run\ eat\ eat\ eat, ...\}$$

Exercise 1.5

What is the size of the above set?

Topic 1.3

Languages

Languages

Definition 1.1

Let Σ be a finite alphabet. A language over Σ is a subset of Σ^*.

A language is acceptable orderings of symbols, which corresponds to the idea of natural or programming languages.

Example 1.8

Let $\Sigma = \{0, 1\}$.

- *Words that end with 1.*

$$\{1, 01, 11, 0101, ...\}$$

- *Words consisting of n 0's followed by n 1's for some $n \geq 0$.*

$$L_{eq} = \{\epsilon, 01, 0011, ...\}$$

Example languages

Example 1.9

Let $\Sigma = \{0, 1\}$.

- *The set of binary strings whose value is a prime number*

$$L_{prime} = \{10, 11, 101, 111, ...\}$$

Exercise 1.6

- *Give a language that has finite elements?*
- *Give a language that is sub-language of all languages?*
- *Give a language that is supper-language of all languages?*

Defining languages using set notation

We will mostly use set notation to communicate the languages under consideration.

Example 1.10

$$L_{eq} = \{0^n1^n | n \geq 0\}$$

$$L_{prime} = \{w | w \text{ is binary encoding of a prime number}\}$$

Exercise 1.7

Write the following languages in set notation.

- $0s$ *followed by more* $1s$.
- *repeating* $01s$
- $1s$ *followed by three* $0s$

Languages as problems

Any computing "problem" can be viewed as a language membership question.

Example 1.11

Consider problem "Is n prime for some given n?".

Let $binary(n)$ *be the binary encoding of n, e. g.,* $binary(12) = 1100$.

Language theoretic formulation of the problem.

$$binary(n) \in L_{prime}?$$

Exercise 1.8

Can there be an automaton that only accepts words from L_{prime}*?*

Key observation

Languages view of problems provide

a clean and handy

way of studying complexity of problems.

Exercise 1.9

Can there be a small automaton that only accepts words from L_{prime}?

Commentary: If you do not follow the above point, please do not worry for now! As we will goes through the course, it will become more and more clear.

End of Lecture 1