

Lecture 1: Intro and Some Regular Languages

Professor Sampath Kannan

Zach Schutzman

NB: These notes are from CIS511 at Penn. The course followed Michael Sipser's *Introduction to the Theory of Computation* (3ed) text.

Introduction

Why theory?

1. Minimal approach to understanding the idea of **computation**.
2. What makes computation tick?
3. Theory anticipates technology.
4. Models of computation are interesting.

Mathematics!

Should know:

1. Sets
2. Functions
3. Relations
4. Logic
5. Proofs
6. Graphs

Definition 1.1 An **alphabet** is a non-empty, finite set of characters.

Definition 1.2 A **string** s (over an alphabet Σ) is a finite ordered sequence of elements of Σ .

Definition 1.3 The **empty string**, ϵ , is the sequence of no symbols, and is in fact a valid string.

Definition 1.4 Let Σ^* be the set of all strings over Σ .

Definition 1.5 A **language** over Σ is any subset of Σ^* .

The empty set is a language. This is not the same as the language only containing the empty string.

Finite State Machines: A First Model

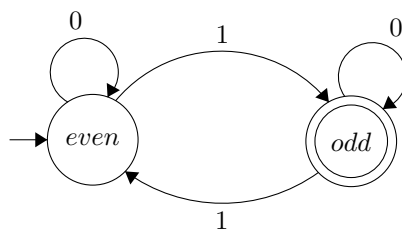
Scalability (asymptotics) is a requirement for any interesting model.

To compute on larger and larger inputs, a computer needs memory. What is the minimum amount of memory you need to do something interesting?

Definition 1.6 A *finite state machine* will be a model with a constant amount of memory.

The states of an FSM correspond to memory (a machine with k states can have 2^k 'bits' of memory).

Example: define the language $\mathcal{L} = \{s \in \Sigma^ \mid s \text{ has an odd number of } 1s\}$.*



Definition 1.7 A *deterministic finite automaton (DFA)*, M , is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$.

Q , the set of states.

Σ , the alphabet

$\delta : Q \times \Sigma \rightarrow Q$, the transition function

q_0 , the start state

F , the accept states

Definition 1.8 M **accepts** a string $s = s_1 s_2 \dots s_k$ if there is a sequence of states in M starting with q_0 and ending in a final state $q_0 q_1 \dots q_k$ such that $\delta(q_i, s_{i+1}) = q_{i+1}$.

Definition 1.9 If $\mathcal{L} = \{s \mid M \text{ accepts } s\}$, then we say M **recognizes** \mathcal{L} .

Definition 1.10 If a language \mathcal{L} is recognized by some DFA, then it is **regular**.

Definition 1.11 A *nondeterministic finite automaton (NFA)*, M , is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$.

Q , the set of states.

Σ , the alphabet

$\delta : Q \times \Sigma \cup \{\epsilon\} \rightarrow \mathcal{P}(Q)$, the transition function

q_0 , the start state

F , the accept states

Now, δ maps the current state and input character or ϵ to some subset of states.

Definition 1.12 A string s is **accepted** by an NFA M if there is some path for s from the start state to a final state.

Definition 1.13 M **recognizes** a language \mathcal{L} consisting of all strings it accepts.

Example: let $\mathcal{L} = \{s \mid \text{the 3rd last character is a 1}\}$.

