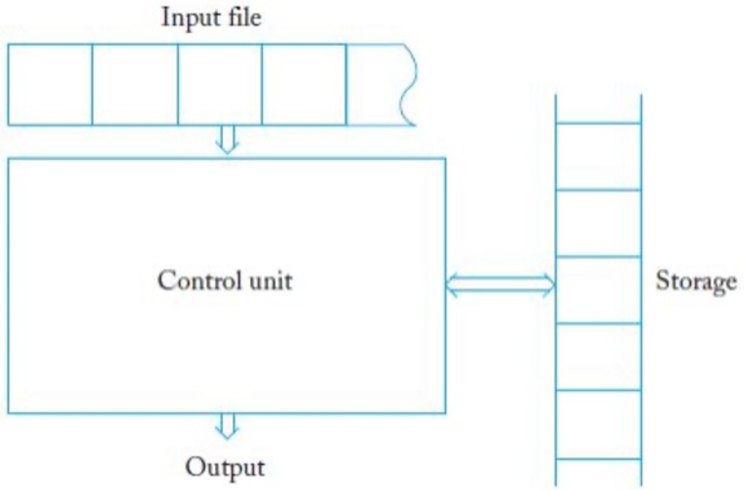
Foundations of Computing

Lecture 1

Arkady Yerukhimovich

January 16, 2024

Modeling Computation



Outline

1. Strings, Languages, and Automata

2. Deterministic Finite Automata (DFA)

Strings

- Alphabet Σ: Set of symbols
  - Ex: $\Sigma = \{a, b\}$, $\Sigma = \{0, 1\}$

Strings

- Alphabet Σ: Set of symbols
  - Ex: $\Sigma = \{a, b\}$, $\Sigma = \{0, 1\}$
- String: finite sequence of symbols from Σ
  - ex: $v = aba$, $w = abaaa$
  - ex: $v = 001$, $w = 11001$
  - λ, ϵ – empty string
  - Length of a string: $|v| = 3$ and $|\lambda| = 0$

Strings

- Alphabet Σ: Set of symbols
  - Ex: $\Sigma = \{a, b\}$, $\Sigma = \{0, 1\}$
- String: finite sequence of symbols from Σ
  - ex: $v = aba$, $w = abaaa$
  - ex: $v = 001$, $w = 11001$
  - λ, ϵ – empty string
  - Length of a string: $|v| = 3$ and $|\lambda| = 0$
- Operations on Strings
  - Concatenation: $vw = abaabaaa$
  - Reverse: $w^R = aaaba$
  - Repeat: $v^2 = abaaba$ and $v^0 = \lambda$

Languages

- Language L: Set of strings
  - We say that any $s \in L$ is in the language

Languages

- Language L: Set of strings
  - We say that any $s \in L$ is in the language
- Examples:
  - $L_1 = \{ab, aa\}$

Languages

- Language L: Set of strings
  - We say that any $s \in L$ is in the language
- Examples:
  - $L_1 = \{ab, aa\}$
  - $L2 = \{a^n b^n : n \geq 0\}$

Languages

- Language L: Set of strings
  - We say that any $s \in L$ is in the language
- Examples:
  - $L_1 = \{ab, aa\}$
  - $L2 = \{a^n b^n : n \geq 0\}$
  - The language of all English sentences

Languages

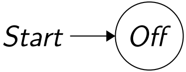
- Language L: Set of strings
  - We say that any $s \in L$ is in the language
- Examples:
  - $L_1 = \{ab, aa\}$
  - $L2 = \{a^n b^n : n \geq 0\}$
  - The language of all English sentences
  - For an alphabet Σ, Σ^* is the set of all strings formed by concatenating zero or more symbols from Σ
    Ex: If $\Sigma = \{0, 1\}$ then $\Sigma^* =$ the set of all binary strings, including λ

Languages

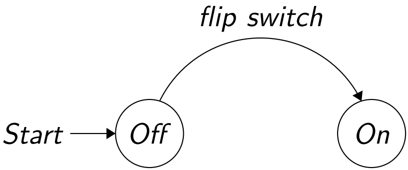
- Language L: Set of strings
  - We say that any $s \in L$ is in the language
- Examples:
  - $L_1 = \{ab, aa\}$
  - $L2 = \{a^n b^n : n \geq 0\}$
  - The language of all English sentences
  - For an alphabet Σ, Σ^* is the set of all strings formed by concatenating zero or more symbols from Σ
    Ex: If $\Sigma = \{0, 1\}$ then $\Sigma^* =$ the set of all binary strings, including λ

We will often be interested in languages recognized by a particular "computer".

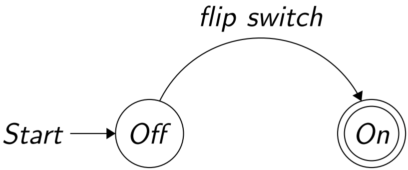
A Simple Example: A Light Switch



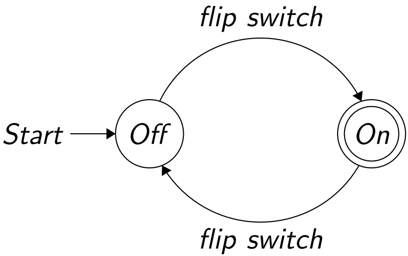
A Simple Example: A Light Switch



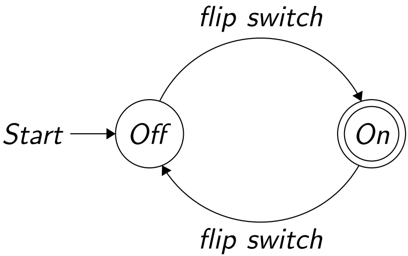
A Simple Example: A Light Switch



A Simple Example: A Light Switch



A Simple Example: A Light Switch



Viewing this as a language

L_{light} = {set of all flip sequences resulting in the light being on}
L_{light} = {1 flip, 3 flips, 5 flips, ...}

Finite Automata

- An automaton is an abstract model of a computing device

Finite Automata

- An automaton is an abstract model of a computing device
- An automaton consists of:
  - An input mechanism

Finite Automata

- An automaton is an abstract model of a computing device
- An automaton consists of:
  - An input mechanism
  - A control unit

Finite Automata

- An automaton is an abstract model of a computing device
- An automaton consists of:
  - An input mechanism
  - A control unit
  - Possibly, a storage mechanism

Finite Automata

- An automaton is an abstract model of a computing device
- An automaton consists of:
  - An input mechanism
  - A control unit
  - Possibly, a storage mechanism
  - Possibly, an output mechanism

Finite Automata

- An automaton is an abstract model of a computing device
- An automaton consists of:
  - An input mechanism
  - A control unit
  - Possibly, a storage mechanism
  - Possibly, an output mechanism
- Control unit transitions between internal states, as determined by a next-state or transition function
- There are a finite number of states – size of the automaton

Finite Automata

- An automaton is an abstract model of a computing device
- An automaton consists of:
  - An input mechanism
  - A control unit
  - Possibly, a storage mechanism
  - Possibly, an output mechanism
- Control unit transitions between internal states, as determined by a next-state or transition function
- There are a finite number of states – size of the automaton

A note of input size

An automaton must be able to accept input of arbitrary length. The length of the input may be much larger than the number of states.

Automata we will study

- Finite Automata (Deterministic and Non-deterministic)
    - These model Finite State Machines with no memory

Automata we will study

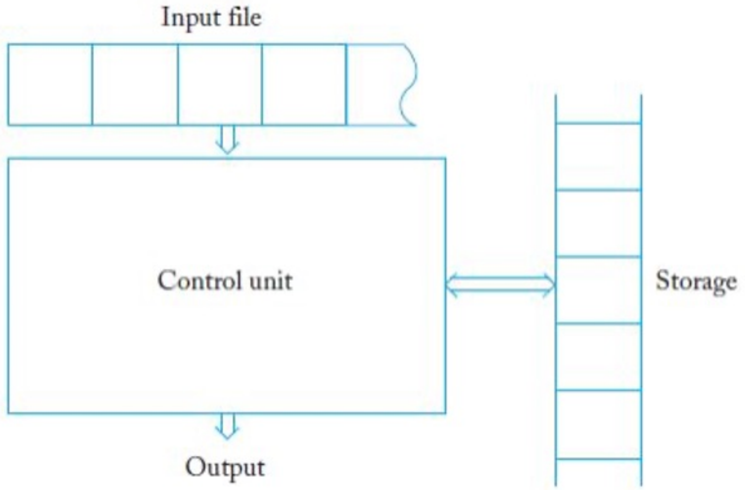
- Finite Automata (Deterministic and Non-deterministic)
  - These model Finite State Machines with no memory
- Pushdown automata
  - Add the simplest form of memory to a Finite State Machine

Automata we will study

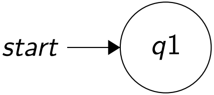
- Finite Automata (Deterministic and Non-deterministic)
  - These model Finite State Machines with no memory
- Pushdown automata
  - Add the simplest form of memory to a Finite State Machine
- Turing Machines
  - Add unrestricted memory to a Finite State Machine
  - Believed to be as powerful as any other model of computation
  - This will be the main model of computation used in computability and complexity theory

Outline

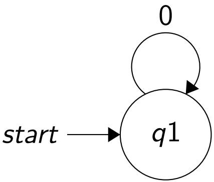
Modeling Computation



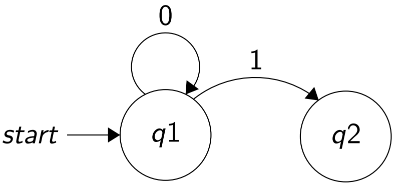
Finite Automata by Picture



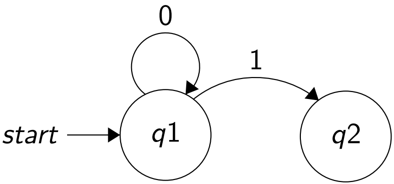
Finite Automata by Picture



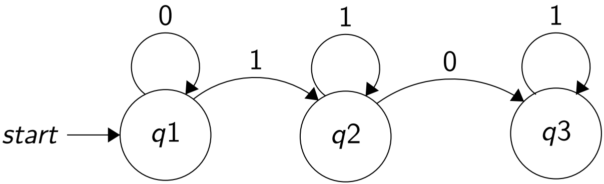
Finite Automata by Picture



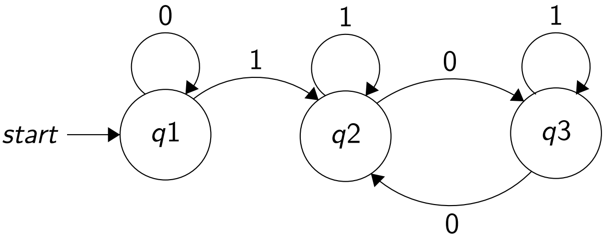
Finite Automata by Picture



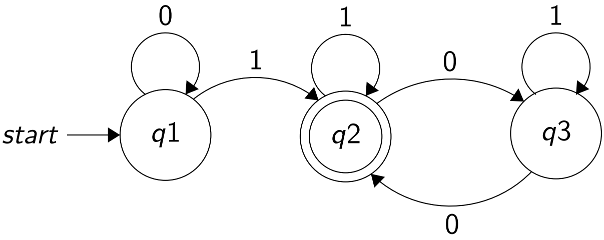
Finite Automata by Picture



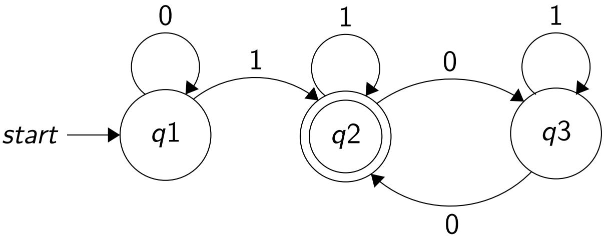
Finite Automata by Picture



Finite Automata by Picture



Finite Automata by Picture



Computation on string $x = 1101$

1. Start in state $q1$
2. read 1, follow transition to $q2$
3. read 1, follow transition to $q2$
4. read 0, follow transition to $q3$
5. read 1, follow transition to $q2$
6. "accept" (output 1) because $q2$ is an accept state

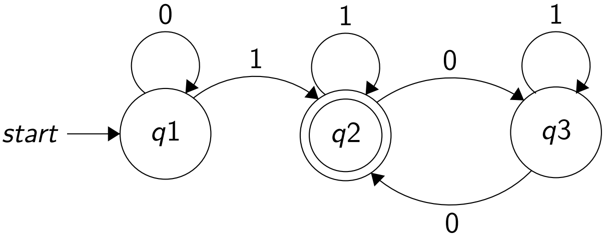
Finite Automaton – Formal Definition

Finite Automaton

A finite automaton is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where:

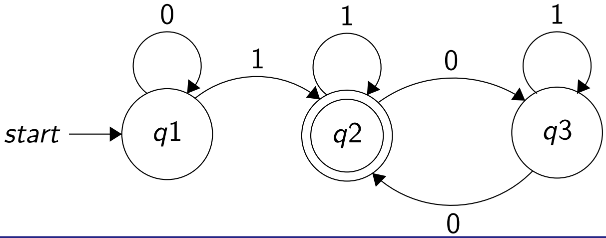
- Q is a finite set of states
- Σ is a finite input alphabet
- $\delta : Q \times \Sigma \to Q$ is the transition function
- $q_0 \in Q$ is the start state
- $F \subseteq Q$ is the set of accept states

Example Automaton



Defining this formally: $M = (Q, \Sigma, \delta, q1, F)$

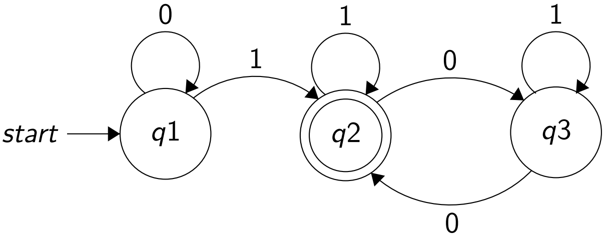
Example Automaton



Defining this formally: $M = (Q, \Sigma, \delta, q1, F)$

- $Q = \{q1, q2, q3\}$

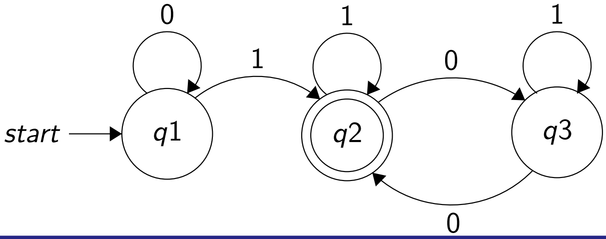
Example Automaton



Defining this formally: $M = (Q, \Sigma, \delta, q1, F)$

- $Q = \{q1, q2, q3\}$
- $\Sigma = \{0, 1\}$

Example Automaton

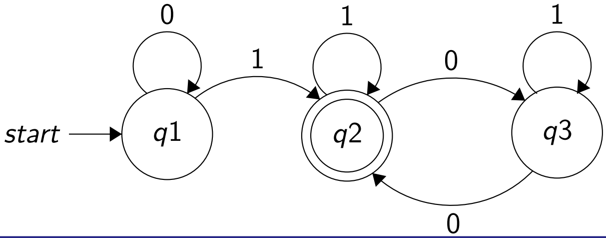


Defining this formally: $M = (Q, \Sigma, \delta, q1, F)$

- $Q = \{q1, q2, q3\}$
- $\Sigma = \{0, 1\}$
- $\delta =$

	0	1
q1	q1	q2

Example Automaton

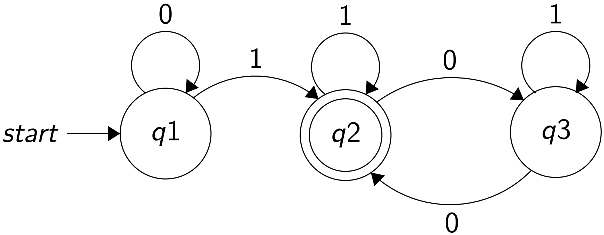


Defining this formally: $M = (Q, \Sigma, \delta, q1, F)$

- $Q = \{q1, q2, q3\}$
- $\Sigma = \{0, 1\}$
- $\delta =$

	0	1
q1	q1	q2
q2	q3	q2
q3	q2	q2

Example Automaton



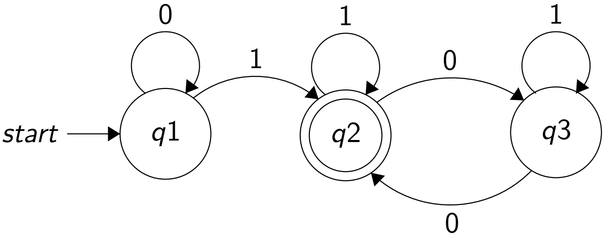
Defining this formally: $M = (Q, \Sigma, \delta, q1, F)$

- $Q = \{q1, q2, q3\}$
- $\Sigma = \{0, 1\}$
- $\delta =$

	0	1
q1	q1	q2
q2	q3	q2
q3	q2	q2

- $q1$ is the start state

Example Automaton



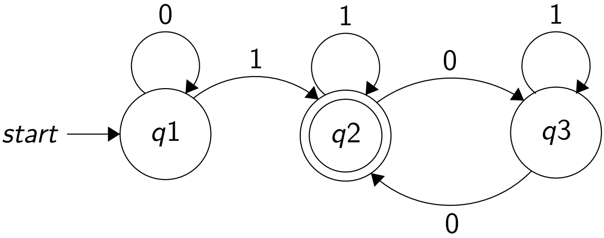
Defining this formally: $M = (Q, \Sigma, \delta, q1, F)$

- $Q = \{q1, q2, q3\}$
- $\Sigma = \{0, 1\}$
- $\delta =$

	0	1
q1	q1	q2
q2	q3	q2
q3	q2	q2

- $q1$ is the start state
- $F = \{q2\}$

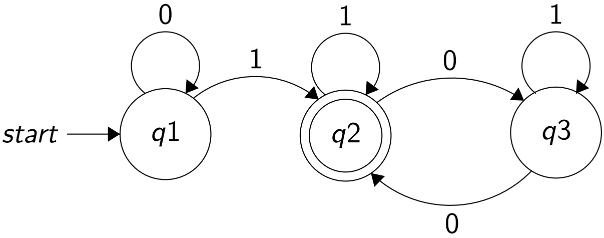
Language accepted by M



Accepting a string

- M accepts a string x (over Σ) if $M(x)$ stops in an accept state
- We already saw that this M accepts 1101
- What other strings does M accept?

Language accepted by M



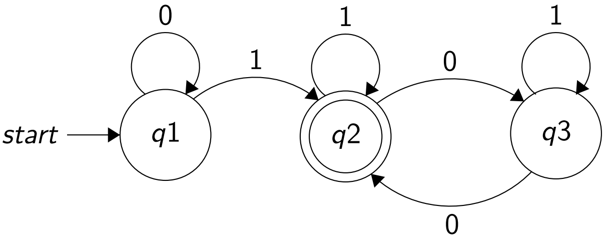
Accepting a string

- M accepts a string x (over Σ) if $M(x)$ stops in an accept state
- We already saw that this M accepts 1101
- What other strings does M accept?

Accepting a language

- M accepts a language L if it accepts ALL strings in L and NO strings not in L
- Every M accepts exactly one language $L(M)$

What language does M accept?



$L(M)$:

- String must contain at least one 1
- After the first string of 1's, there must be an even number of 0's or no 0's

Why study this?

- Finite Automata are one of the most basic models of computation
- Turns out they capture some very useful functionalities
  - We will see next week, that finite automata correspond to regular expressions

Next...

- Labs this week:
  - Review of proof techniques
  - Review languages/strings/graphs
  - In-class exercises

Next...

- Labs this week:
  - Review of proof techniques
  - Review languages/strings/graphs
  - In-class exercises
- Thursday lecture:
  - More about finite automata and their properties

Next...

- Labs this week:
  - Review of proof techniques
  - Review languages/strings/graphs
  - In-class exercises
- Thursday lecture:
  - More about finite automata and their properties
- Your to do list:
  - Sign up for Gradescope
  - Sign up for Piazza
  - (optional) Download and install JFLAP (check tutorial on course webpage)