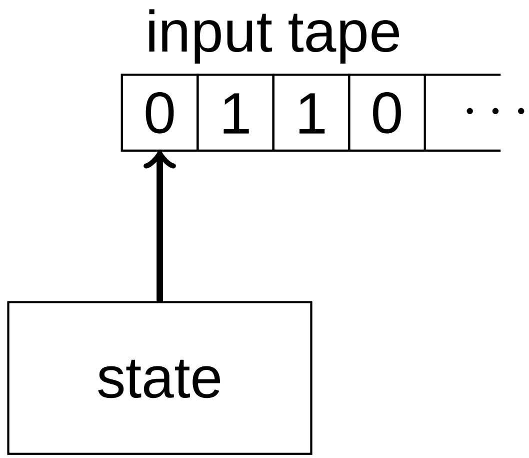
TDDD14/TDDD85
Slides for Lecture 2, 2017

Slides originally for TDDD65 by Gustav Nordh

Minor differences to Kozen:
- Kozen draws states as black dots, not circles.
- Kozen calls the start state s, not q_0.
- Kozen uses the recursive function $\widehat{\delta}$ to define when a DFA accepts a string. Both definitions give the same result.

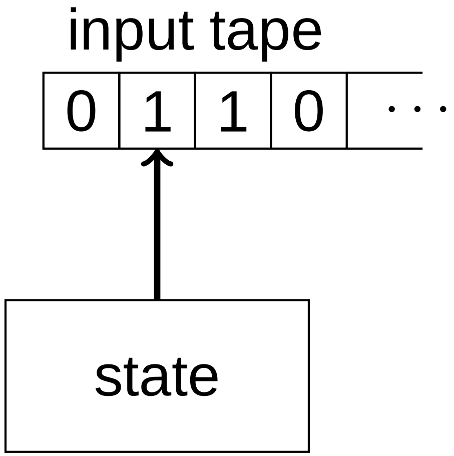
Finite Automata

What can be computed with finite memory?



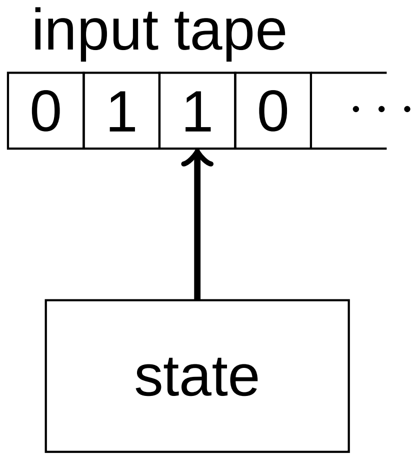
Finite Automata

What can be computed with finite memory?



Finite Automata

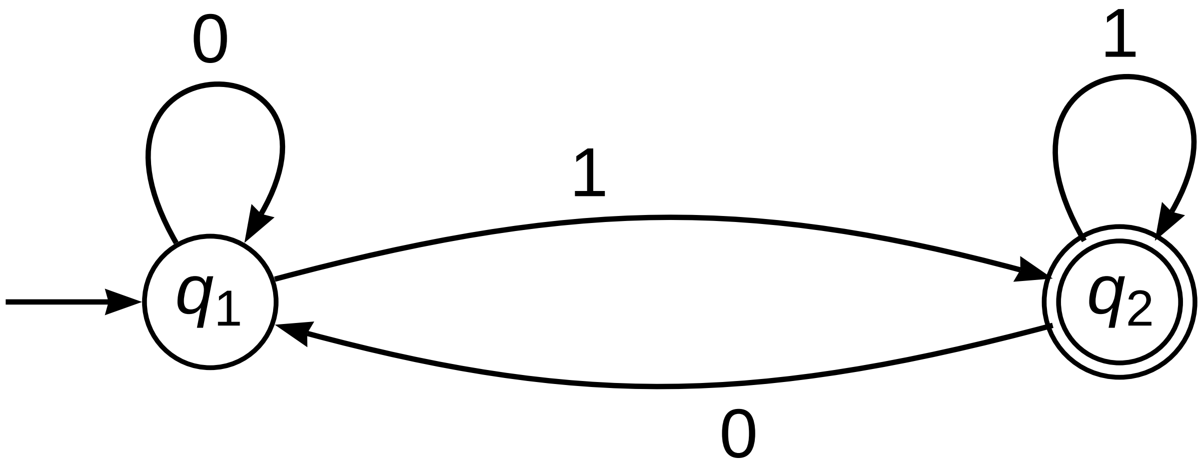
What can be computed with finite memory?



A “Wild” Finite Automaton

- The input is the numbers pressed by the user
- The correct code is 1234
- Actually the door is opened if $\{0, \ldots, 9\}^*1234\{0, \ldots, 9\}^*$ is entered
- The machine needs to remember if the input given so far contains the subsequence 1234

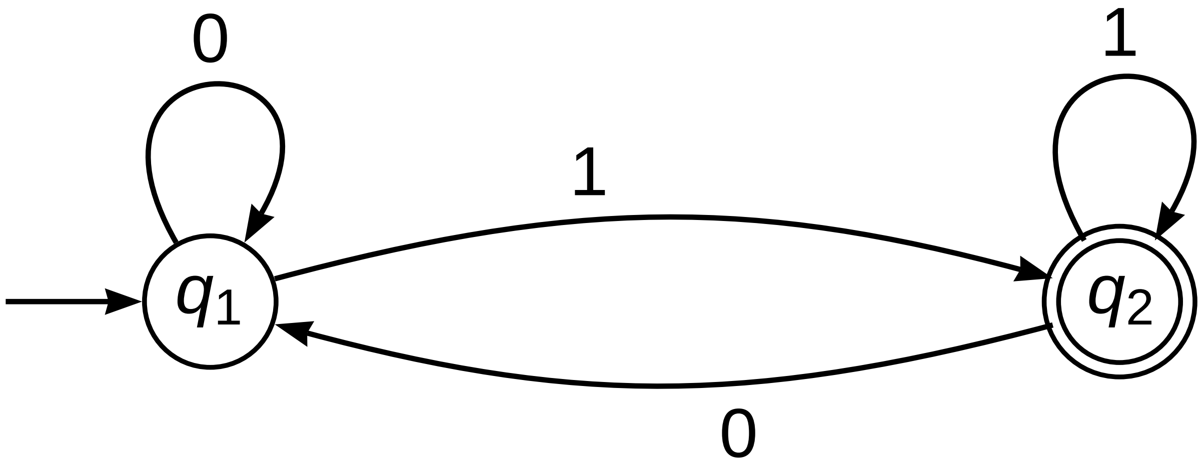
Finite Automata: Informal Definition



The machine M:

- States: q_1 and q_2
- Start state: q_1 (arrow from nowhere)
- Accept state: q_2 (double circle)
- State transitions: arrows

Finite Automata: Informal Definition

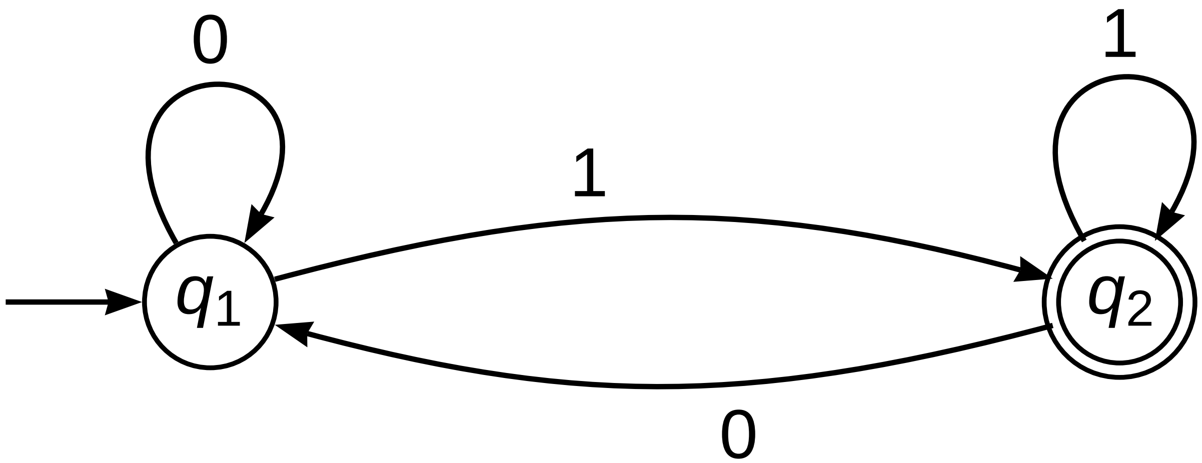


The machine M:

On input string $s = s_1 s_2 \cdots s_n$, M operates as follows:

- Begins in start state q_1 and reads the string s from left to right
- When reading symbol s_i it follows the transition labeled s_i from the current state
- After reading s_n, the last symbol in the string, it
  - accepts s if it is in an accept state
  - rejects s if it is not in an accept state

Finite Automata: Informal Definition

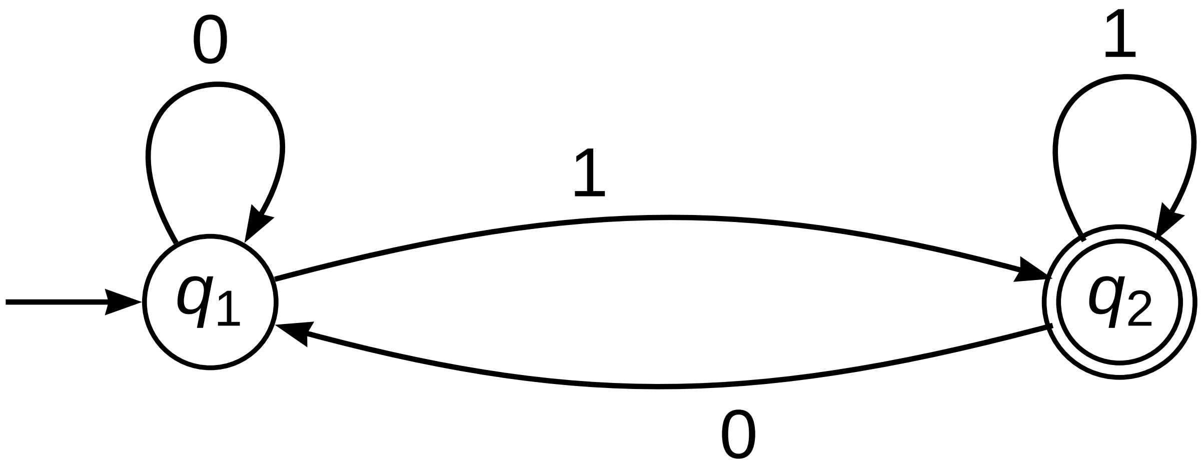


The machine M:

On input:

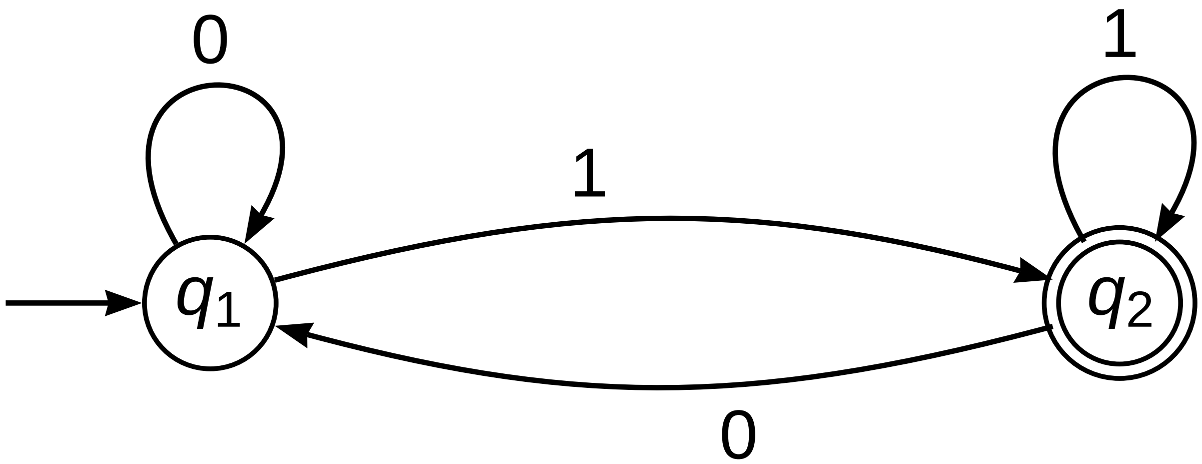
011

Finite Automata: Informal Definition



The machine M:
On input:
011 ACCEPT

Finite Automata: Informal Definition



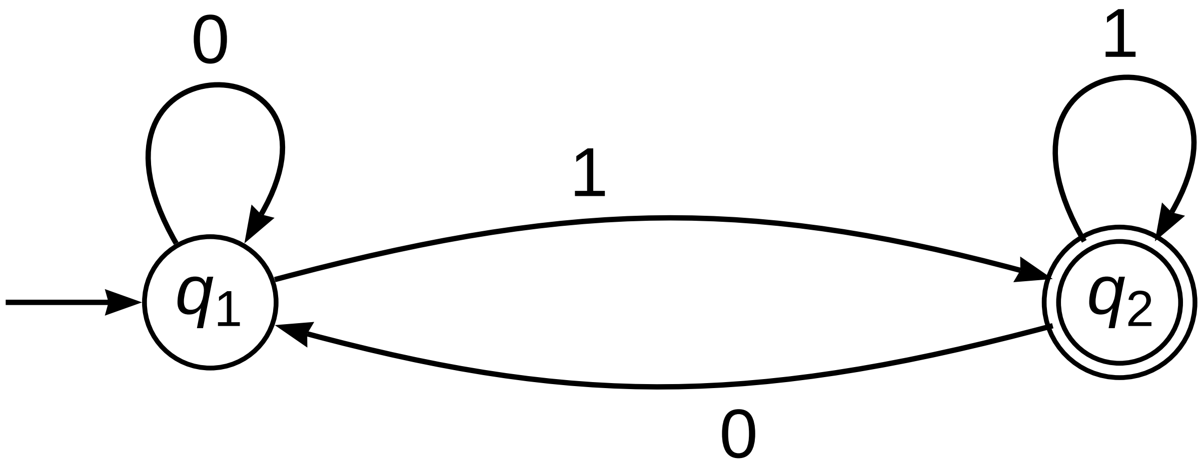
The machine M:

On input:

011 ACCEPT

10

Finite Automata: Informal Definition



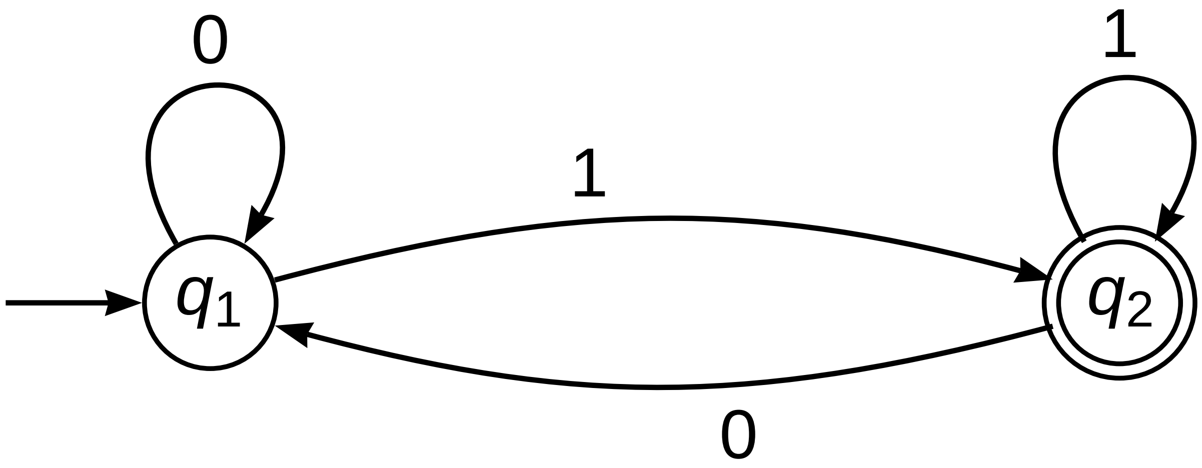
The machine M:

On input:

011 ACCEPT

10 REJECT

Finite Automata: Informal Definition



The machine *M*:

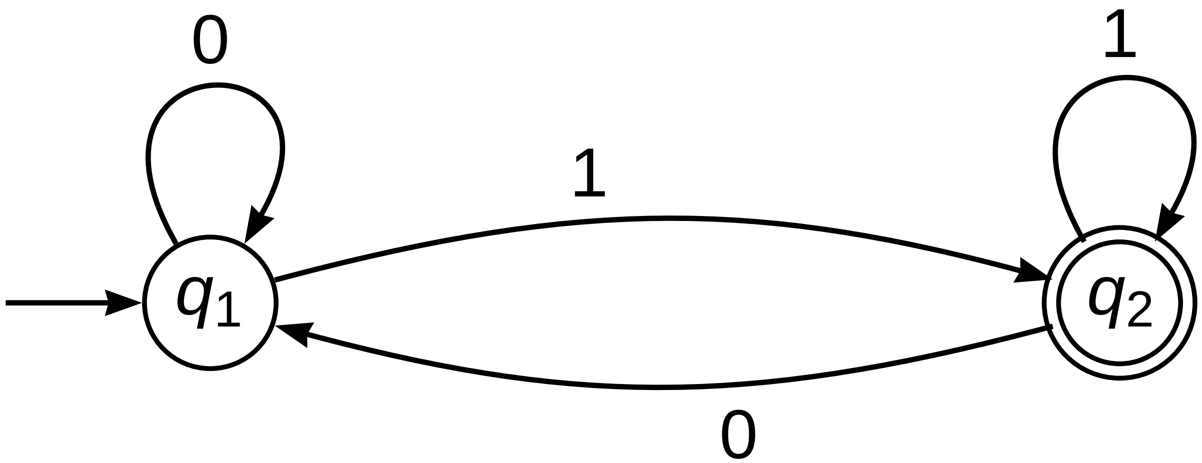
On input:

011 ACCEPT

10 REJECT

110

Finite Automata: Informal Definition



The machine M:

On input:

011 ACCEPT

10 REJECT

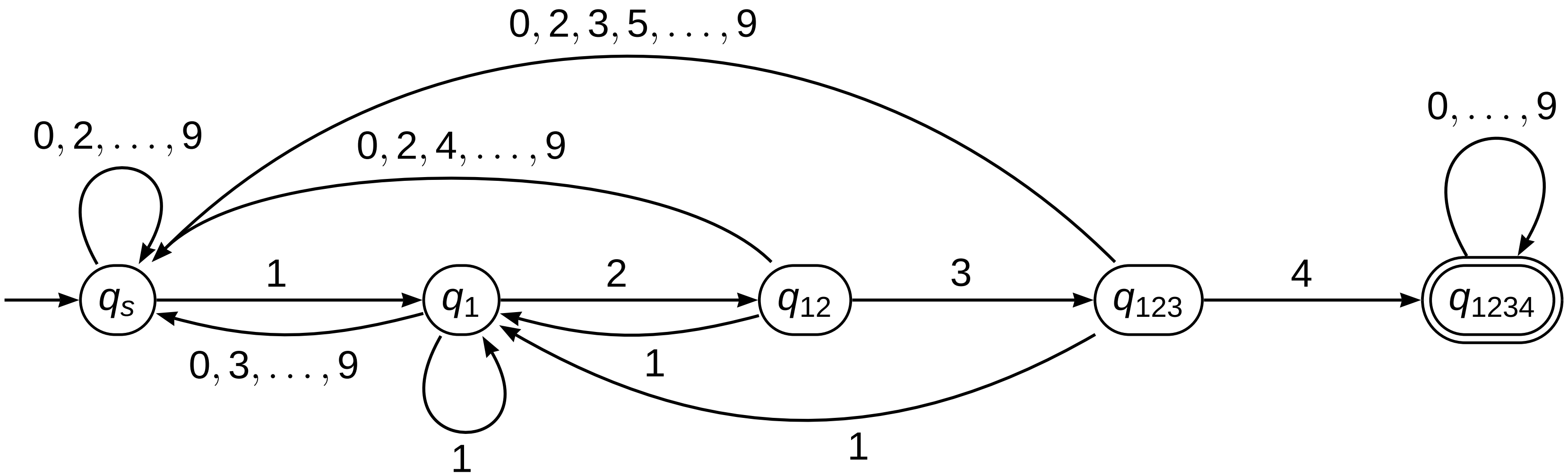
110 REJECT

A “Wild” Finite Automaton

- The input is the numbers pressed by the user
- The correct code is 1234
- Actually the door is opened if $\{0, \ldots, 9\}^* 1234 \{0, \ldots, 9\}^*$ is entered
- The machine needs to remember if the input given so far contains the subsequence 1234

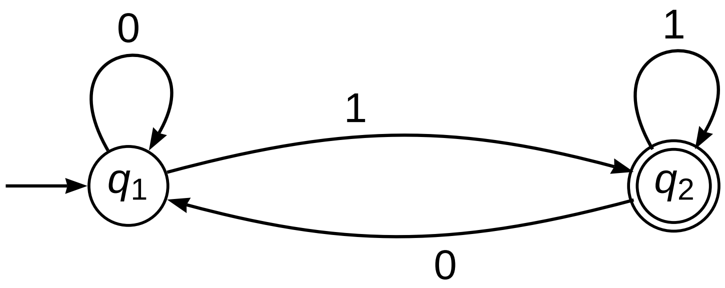
A “Wild” Finite Automaton

- The input is the numbers pressed by the user
- The correct code is 1234
- Actually the door is opened if $\{0, \ldots, 9\}^* 1234 \{0, \ldots, 9\}^*$ is entered
- The machine needs to remember if the input given so far contains the subsequence 1234



Representation of finite automata

- State diagram



- Transition table

	0	1
$\rightarrow q_1$	q_1	q_2
$F\ q_2$	q_1	q_2

Definition of DFAs

Definition

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

- Q is a finite set called the states
- Σ is a finite set called the 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

The language recognized by a DFA

Definition

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA and $s = s_1 s_2 \cdots s_n$ a string over Σ. M accepts s if there is a sequence of states $r_0, r_1, \ldots r_n$ from Q such that

- $r_0 = q_0$,
- $\delta(r_i, s_{i+1}) = r_{i+1}$ ($i = 0, \ldots, n-1$), and
- $r_n \in F$

The language recognized by a DFA

Definition

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA and $s = s_1 s_2 \cdots s_n$ a string over Σ. M accepts s if there is a sequence of states $r_0, r_1, \ldots r_n$ from Q such that

- $r_0 = q_0$,
- $\delta(r_i, s_{i+1}) = r_{i+1}$ ($i = 0, \ldots, n-1$), and
- $r_n \in F$

Definition

- M recognizes language L if $L = \{s \mid M \textit{ accepts } s\}$
- $L(M)$ denotes the language recognized by M

Definition

A language is a **regular language** if some DFA recognizes it