

# Theory of Computer Science

## B3. Finite Automata

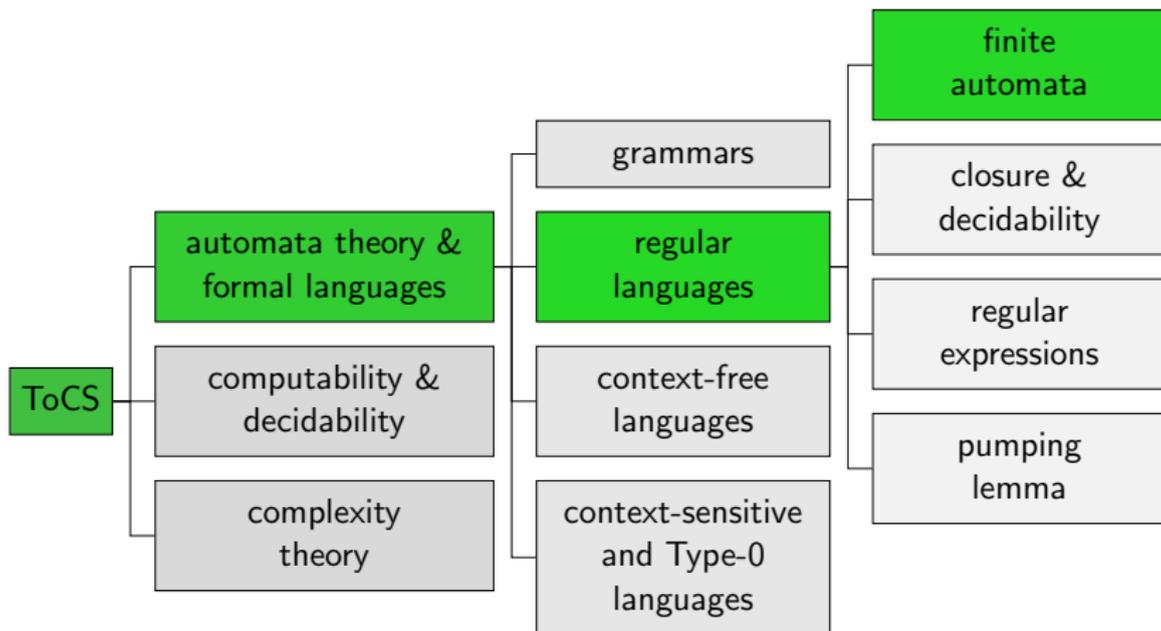
Gabriele Röger

University of Basel

March 4/9, 2026

# Introduction

# Content of the Course

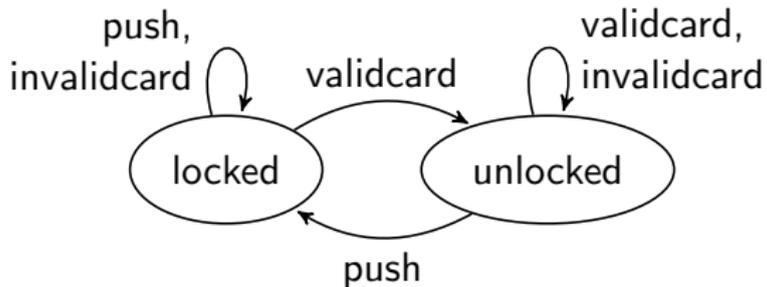


# A Controller for a Turnstile



CC BY-SA 3.0, author: Stolbovsky

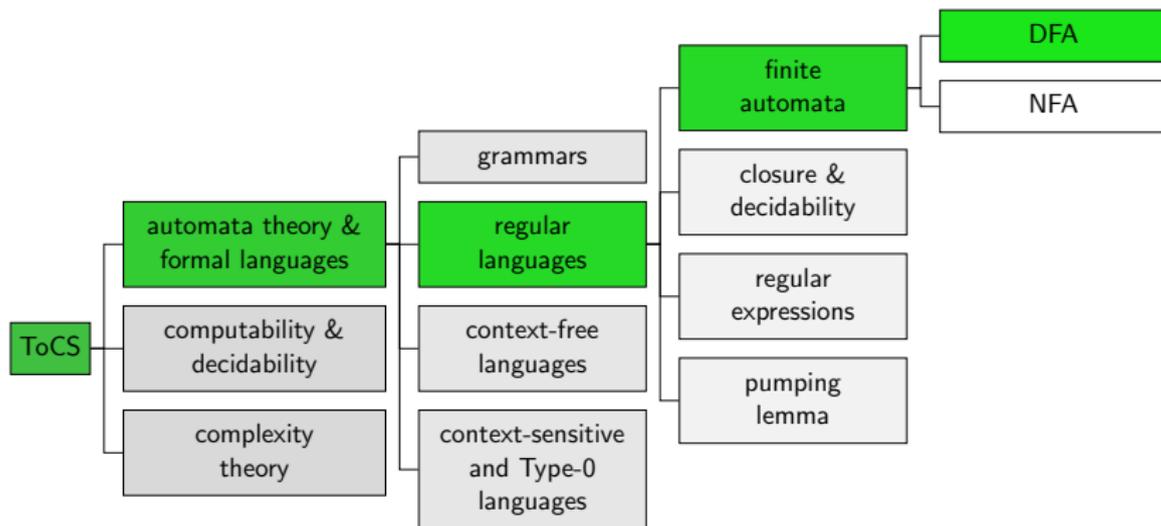
- simple access control
- card reader and push sensor
- card can either be valid or invalid



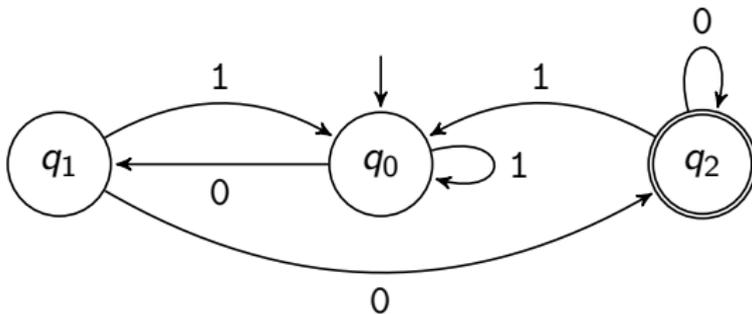
- Finite automata are a good model for computers with very limited memory.  
Where can the turnstile controller store information about what it has seen in the past?
- We will not consider automata that run forever but that process a **finite input sequence** and then classify it as **accepted** or not.

# DFAs

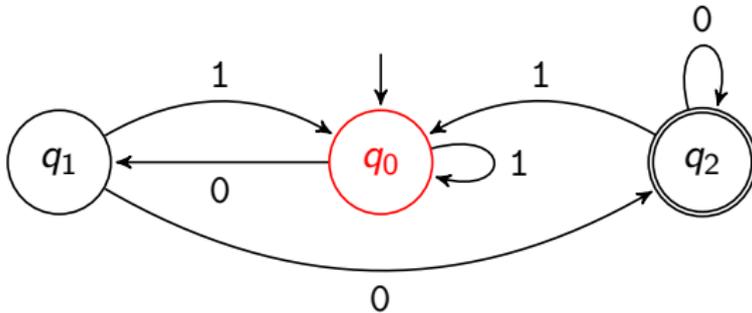
# Content of the Course



# Finite Automaton: Example

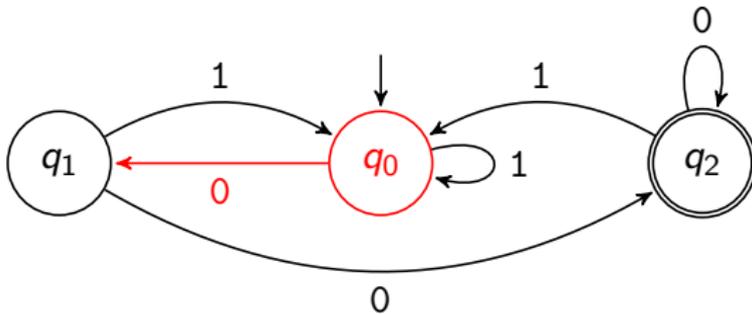


# Finite Automaton: Example



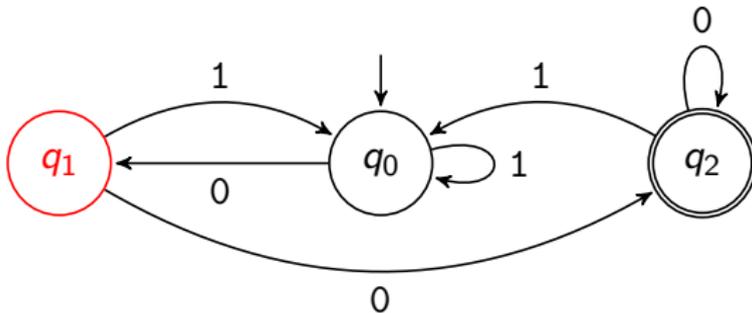
When reading the input 01100 the automaton visits the states  
 $q_0$ ,

# Finite Automaton: Example



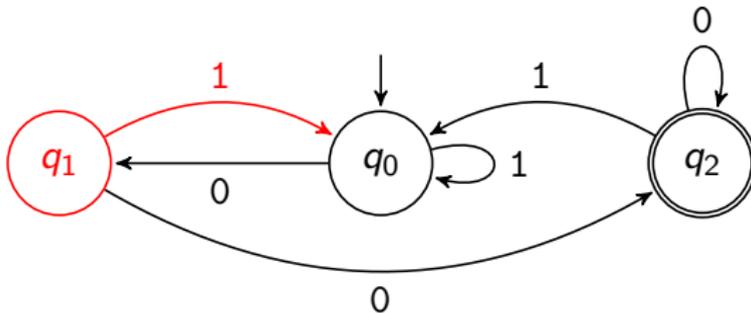
When reading the input **0**1100 the automaton visits the states  $q_0$ ,

# Finite Automaton: Example



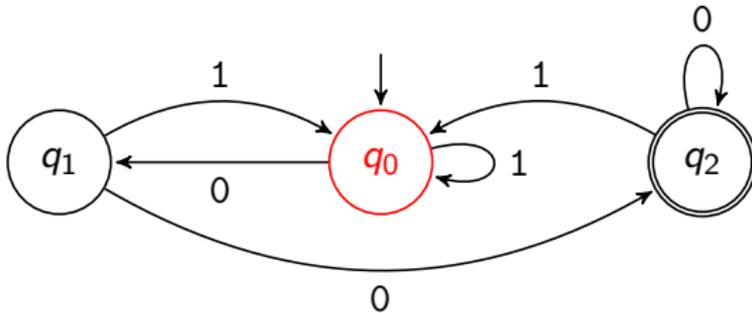
When reading the input 01100 the automaton visits the states  $q_0$ ,  $q_1$ ,

# Finite Automaton: Example



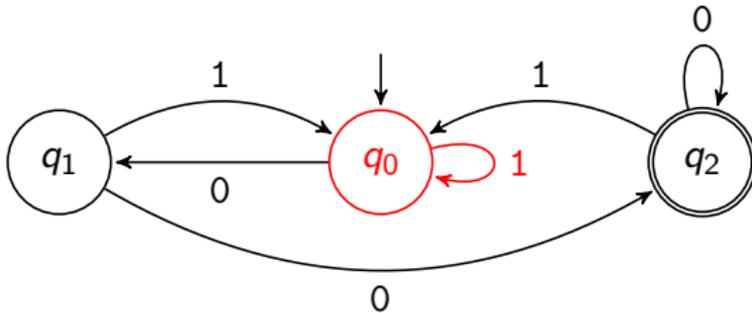
When reading the input 0**1**100 the automaton visits the states  $q_0, q_1,$

# Finite Automaton: Example



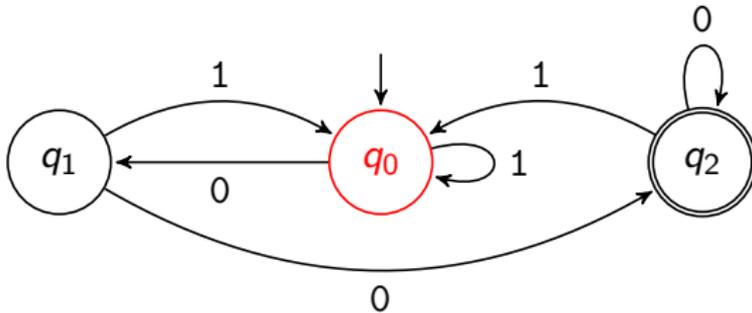
When reading the input 01100 the automaton visits the states  
 $q_0, q_1, q_0,$

# Finite Automaton: Example



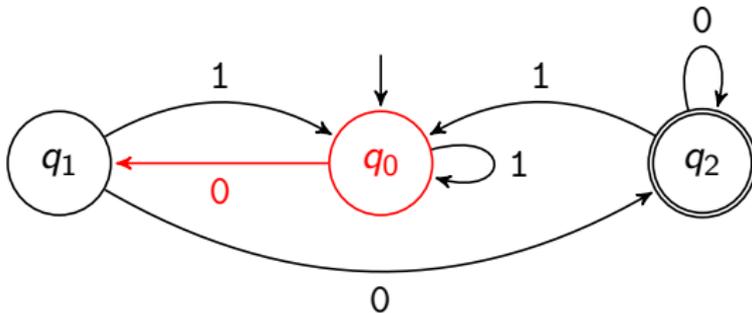
When reading the input 01100 the automaton visits the states  $q_0, q_1, q_0,$

# Finite Automaton: Example



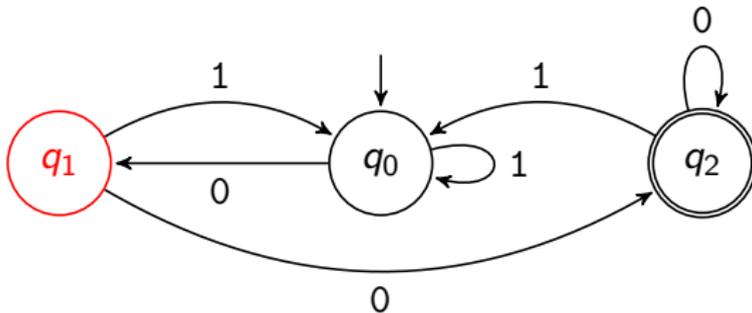
When reading the input 01100 the automaton visits the states  
 $q_0, q_1, q_0, q_0,$

# Finite Automaton: Example



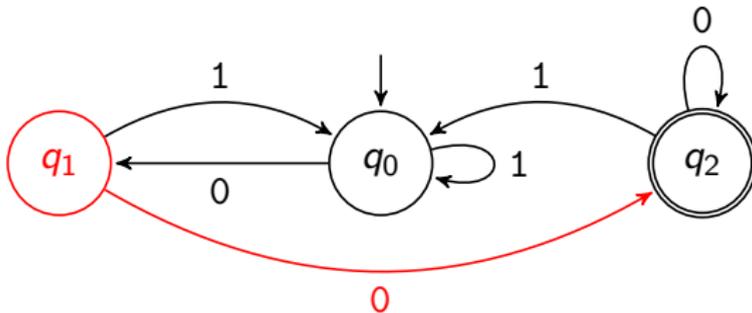
When reading the input 01100 the automaton visits the states  
 $q_0, q_1, q_0, q_0,$

# Finite Automaton: Example



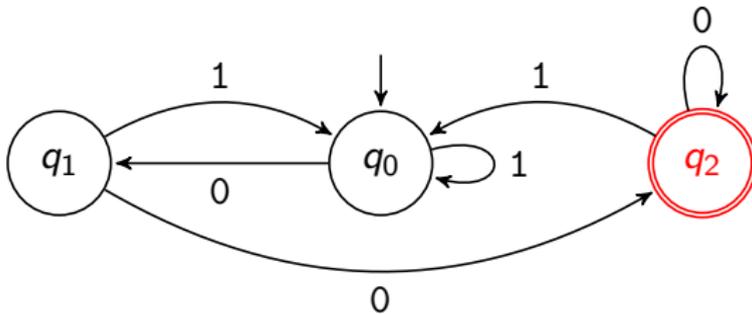
When reading the input 01100 the automaton visits the states  
 $q_0, q_1, q_0, q_0, q_1,$

# Finite Automaton: Example



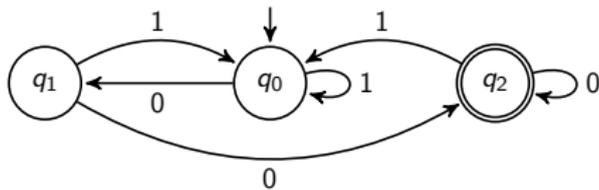
When reading the input 01100 the automaton visits the states  $q_0, q_1, q_0, q_0, q_1,$

# Finite Automaton: Example

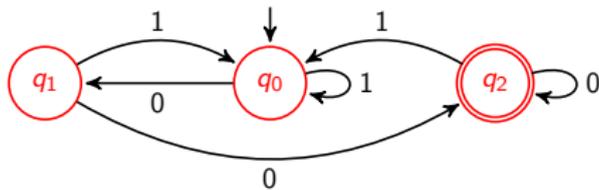


When reading the input 01100 the automaton visits the states  $q_0, q_1, q_0, q_0, q_1, q_2$ .

# Finite Automata: Terminology and Notation

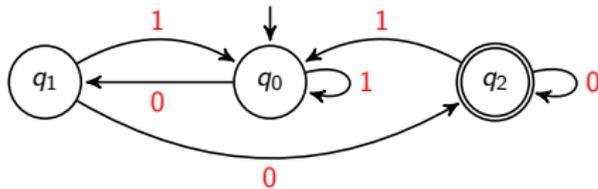


# Finite Automata: Terminology and Notation



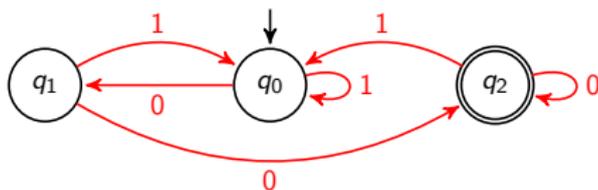
- states  $Q = \{q_0, q_1, q_2\}$

# Finite Automata: Terminology and Notation



- states  $Q = \{q_0, q_1, q_2\}$
- input alphabet  $\Sigma = \{0, 1\}$

# Finite Automata: Terminology and Notation



- states  $Q = \{q_0, q_1, q_2\}$
- input alphabet  $\Sigma = \{0, 1\}$
- transition function  $\delta$

$$\delta(q_0, 0) = q_1$$

$$\delta(q_0, 1) = q_0$$

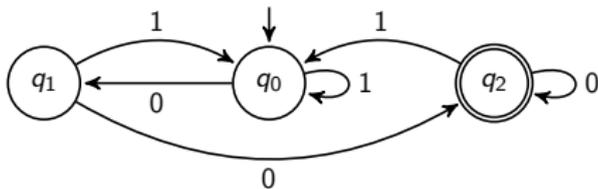
$$\delta(q_1, 0) = q_2$$

$$\delta(q_1, 1) = q_0$$

$$\delta(q_2, 0) = q_2$$

$$\delta(q_2, 1) = q_0$$

# Finite Automata: Terminology and Notation



- states  $Q = \{q_0, q_1, q_2\}$
- input alphabet  $\Sigma = \{0, 1\}$
- transition function  $\delta$

$$\delta(q_0, 0) = q_1$$

$$\delta(q_0, 1) = q_0$$

$$\delta(q_1, 0) = q_2$$

$$\delta(q_1, 1) = q_0$$

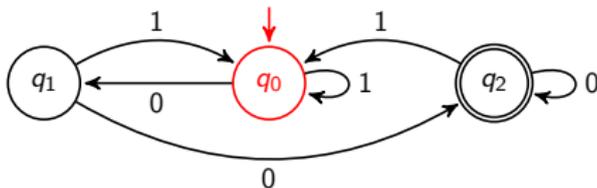
$$\delta(q_2, 0) = q_2$$

$$\delta(q_2, 1) = q_0$$

$\delta$	0	1
$q_0$	$q_1$	$q_0$
$q_1$	$q_2$	$q_0$
$q_2$	$q_2$	$q_0$

table form of  $\delta$

# Finite Automata: Terminology and Notation



- states  $Q = \{q_0, q_1, q_2\}$
- input alphabet  $\Sigma = \{0, 1\}$
- transition function  $\delta$
- start state  $q_0$

$$\delta(q_0, 0) = q_1$$

$$\delta(q_0, 1) = q_0$$

$$\delta(q_1, 0) = q_2$$

$$\delta(q_1, 1) = q_0$$

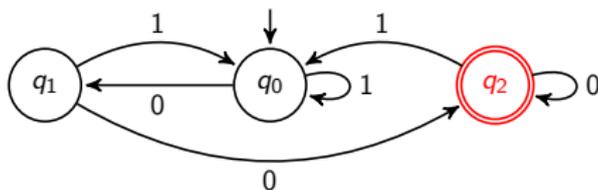
$$\delta(q_2, 0) = q_2$$

$$\delta(q_2, 1) = q_0$$

$\delta$	0	1
$q_0$	$q_1$	$q_0$
$q_1$	$q_2$	$q_0$
$q_2$	$q_2$	$q_0$

table form of  $\delta$

# Finite Automata: Terminology and Notation



- states  $Q = \{q_0, q_1, q_2\}$
- input alphabet  $\Sigma = \{0, 1\}$
- transition function  $\delta$
- start state  $q_0$
- accept states  $\{q_2\}$

$$\delta(q_0, 0) = q_1$$

$$\delta(q_0, 1) = q_0$$

$$\delta(q_1, 0) = q_2$$

$$\delta(q_1, 1) = q_0$$

$$\delta(q_2, 0) = q_2$$

$$\delta(q_2, 1) = q_0$$

$\delta$	0	1
$q_0$	$q_1$	$q_0$
$q_1$	$q_2$	$q_0$
$q_2$	$q_2$	$q_0$

table form of  $\delta$

# Deterministic Finite Automaton: Definition

## Definition (Deterministic Finite Automata)

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

- $Q$  is the finite set of **states**
- $\Sigma$  is the **input alphabet**
- $\delta : Q \times \Sigma \rightarrow Q$  is the **transition function**
- $q_0 \in Q$  is the **start state**
- $F \subseteq Q$  is the set of **accept states** (or **final states**)

## DFA: Accepted Words

Intuitively, a DFA **accepts a word** if its computation terminates in an **accept state**.

## DFA: Accepted Words

Intuitively, a DFA **accepts a word** if its computation terminates in an **accept state**.

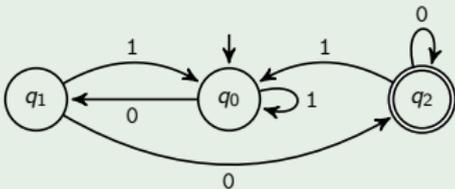
### Definition (Words Accepted by a DFA)

DFA  $M = \langle Q, \Sigma, \delta, q_0, F \rangle$  **accepts the word**  $w = a_1 \dots a_n$  if there is a sequence of states  $q'_0, \dots, q'_n \in Q$  with

- 1  $q'_0 = q_0$ ,
- 2  $\delta(q'_{i-1}, a_i) = q'_i$  for all  $i \in \{1, \dots, n\}$  and
- 3  $q'_n \in F$ .

# Example

## Example



accepts:

00

10010100

01000

does not accept:

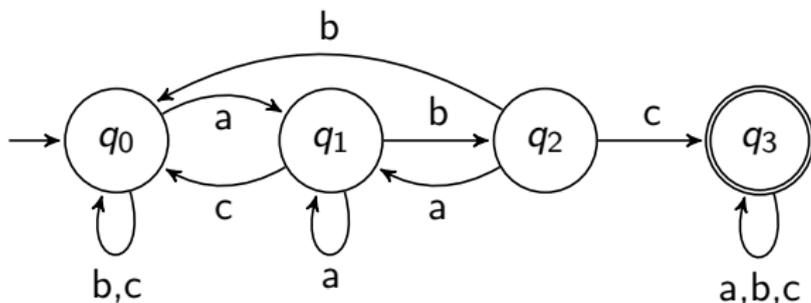
$\epsilon$

1001010

010001

## Exercise (slido)

Consider the following DFA:



Which of the following words does it accept?

- abc
- ababcb
- babbc

# DFA: Recognized Language

## Definition (Language Recognized by a DFA)

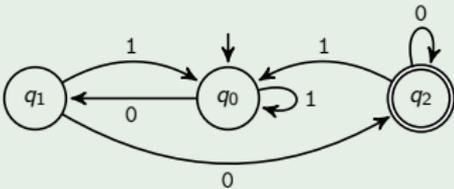
Let  $M$  be a deterministic finite automaton.

The **language recognized by  $M$**  is defined as

$$\mathcal{L}(M) = \{w \in \Sigma^* \mid w \text{ is accepted by } M\}.$$

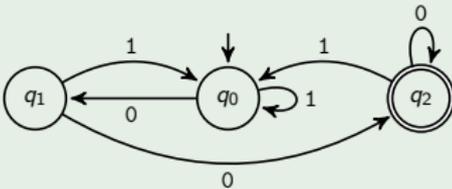
# Example

## Example



# Example

## Example



The DFA recognizes the language  $\{w \in \{0, 1\}^* \mid w \text{ ends with } 00\}$ .

## A Note on Terminology

- In the literature, “accept” and “recognize” are sometimes used synonymously or the other way around.  
DFA recognizes a word or accepts a language.
- We try to stay consistent using the previous definitions (following the text book by Sipser).

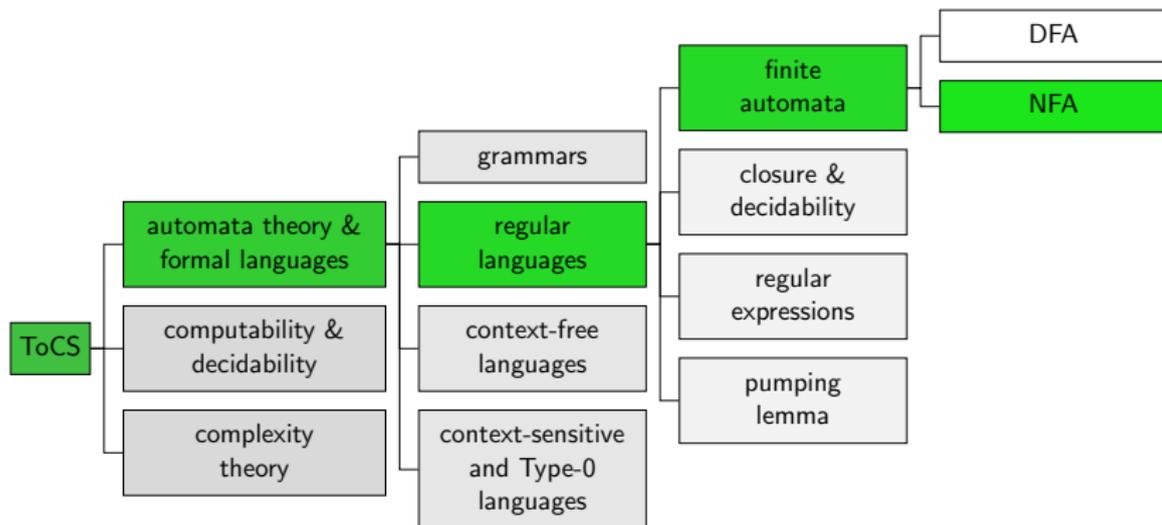
# Questions



Questions?

# NFAs

# Content of the Course



# Nondeterministic Finite Automata

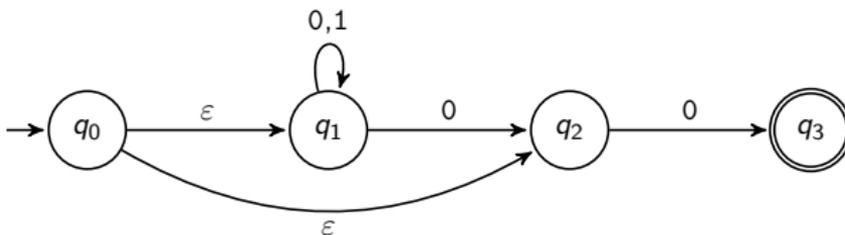
Why are DFAs called **deterministic** automata? What are **nondeterministic** automata, then?



## In what Sense is a DFA Deterministic?

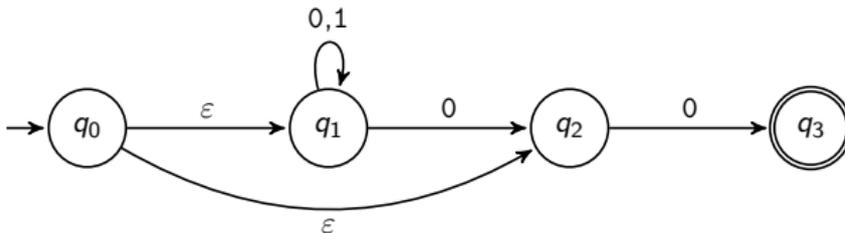
- A DFA has a single fixed state from which the computation starts.
- When a DFA is in a specific state and reads an input symbol, we know what the next state will be.
- For a given input, the entire computation is determined.
- This is a **deterministic** computation.

# Nondeterministic Finite Automata: Example



differences to DFAs:

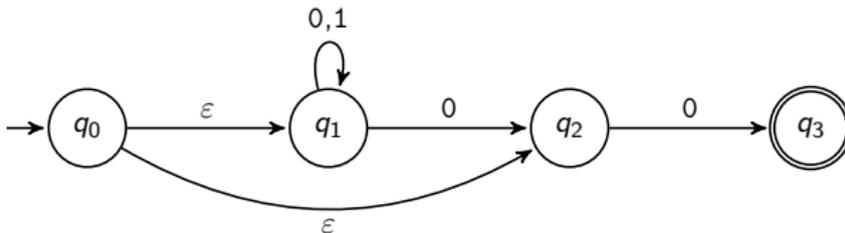
# Nondeterministic Finite Automata: Example



differences to DFAs:

- transition function  $\delta$  can lead to **zero** or **more** successor states for the **same**  $a \in \Sigma$

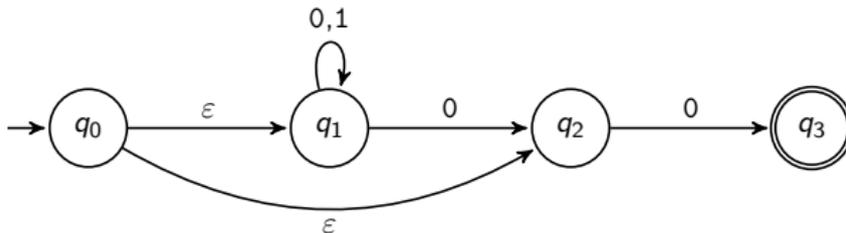
# Nondeterministic Finite Automata: Example



differences to DFAs:

- transition function  $\delta$  can lead to **zero** or **more** successor states for the **same**  $a \in \Sigma$
- **$\varepsilon$ -transitions** can be taken without “consuming” a symbol from the input

# Nondeterministic Finite Automata: Example



differences to DFAs:

- transition function  $\delta$  can lead to **zero** or **more** successor states for the **same**  $a \in \Sigma$
- **$\varepsilon$ -transitions** can be taken without “consuming” a symbol from the input
- the automaton accepts a word if there is **at least one** accepting sequence of states

# Nondeterministic Finite Automaton: Definition

## Definition (Nondeterministic Finite Automata)

A **nondeterministic finite automaton (NFA)** is a 5-tuple  $M = \langle Q, \Sigma, \delta, q_0, F \rangle$  where

- $Q$  is the finite set of **states**
- $\Sigma$  is the **input alphabet**
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow \mathcal{P}(Q)$  is the transition function (mapping to the **power set** of  $Q$ )
- $q_0 \in Q$  is the **start state**
- $F \subseteq Q$  is the set of **accept states**

# Nondeterministic Finite Automaton: Definition

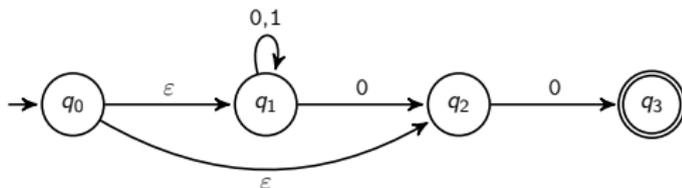
## Definition (Nondeterministic Finite Automata)

A **nondeterministic finite automaton (NFA)** is a 5-tuple  $M = \langle Q, \Sigma, \delta, q_0, F \rangle$  where

- $Q$  is the finite set of **states**
- $\Sigma$  is the **input alphabet**
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow \mathcal{P}(Q)$  is the transition function (mapping to the **power set** of  $Q$ )
- $q_0 \in Q$  is the **start state**
- $F \subseteq Q$  is the set of **accept states**

DFAs are (essentially) a special case of NFAs.

# Accepting Computation: Example



$w = 0100$

↪ computation tree on blackboard

## $\varepsilon$ -closure of a State

For a state  $q \in Q$ , we write  $E(q)$  to denote the set of states that are reachable from  $q$  via  $\varepsilon$ -transitions in  $\delta$ .

## $\varepsilon$ -closure of a State

For a state  $q \in Q$ , we write  $E(q)$  to denote the set of states that are reachable from  $q$  via  $\varepsilon$ -transitions in  $\delta$ .

### Definition ( $\varepsilon$ -closure)

For NFA  $M = \langle Q, \Sigma, \delta, q_0, F \rangle$  and state  $q \in Q$ , state  $p$  is in the  $\varepsilon$ -closure  $E(q)$  of  $q$  iff there is a sequence of states  $q'_0, \dots, q'_n$  with

- 1  $q'_0 = q$ ,
- 2  $q'_i \in \delta(q'_{i-1}, \varepsilon)$  for all  $i \in \{1, \dots, n\}$  and
- 3  $q'_n = p$ .

## $\varepsilon$ -closure of a State

For a state  $q \in Q$ , we write  $E(q)$  to denote the set of states that are reachable from  $q$  via  $\varepsilon$ -transitions in  $\delta$ .

### Definition ( $\varepsilon$ -closure)

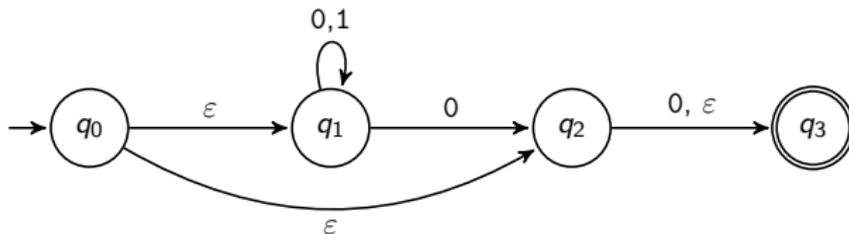
For NFA  $M = \langle Q, \Sigma, \delta, q_0, F \rangle$  and state  $q \in Q$ , state  $p$  is in the  $\varepsilon$ -closure  $E(q)$  of  $q$  iff there is a sequence of states  $q'_0, \dots, q'_n$  with

- 1  $q'_0 = q$ ,
- 2  $q'_i \in \delta(q'_{i-1}, \varepsilon)$  for all  $i \in \{1, \dots, n\}$  and
- 3  $q'_n = p$ .

$q \in E(q)$  for every state  $q$

# Exercise (slido)

Consider the following NFA:



Which states are in the  $\varepsilon$ -closure  $E(q_0)$ ?

- $q_0$
- $q_1$
- $q_2$
- $q_3$



# NFA: Accepted Words

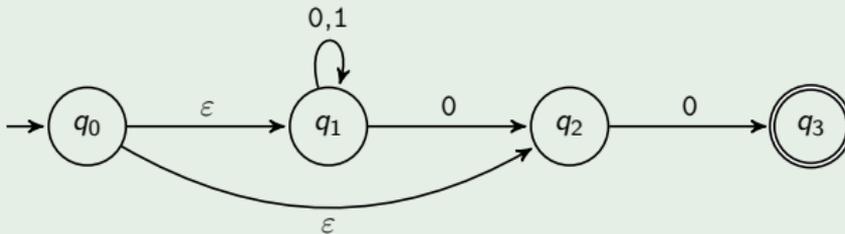
## Definition (Words Accepted by an NFA)

NFA  $M = \langle Q, \Sigma, \delta, q_0, F \rangle$  **accepts the word**  $w = a_1 \dots a_n$  if there is a sequence of states  $q'_0, \dots, q'_n \in Q$  with

- 1  $q'_0 \in E(q_0)$ ,
- 2  $q'_i \in \bigcup_{q \in \delta(q'_{i-1}, a_i)} E(q)$  for all  $i \in \{1, \dots, n\}$  and
- 3  $q'_n \in F$ .

# Example: Accepted Words

## Example



accepts:

0

10010100

01000

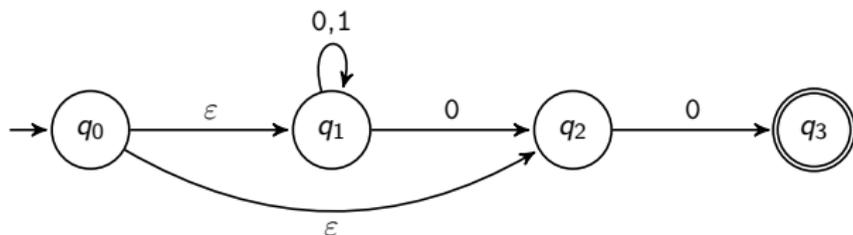
does not accept:

ε

1001010

010001

# Exercise (slido)



Does this NFA accept input 01010?

# NFA: Recognized Language

## Definition (Language Recognized by an NFA)

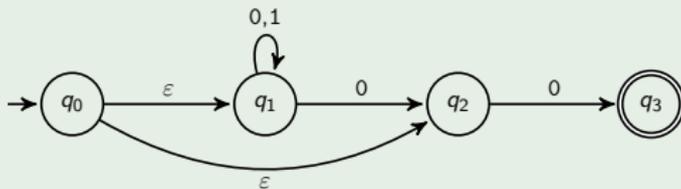
Let  $M$  be an NFA with input alphabet  $\Sigma$ .

The **language recognized by  $M$**  is defined as

$$\mathcal{L}(M) = \{w \in \Sigma^* \mid w \text{ is accepted by } M\}.$$

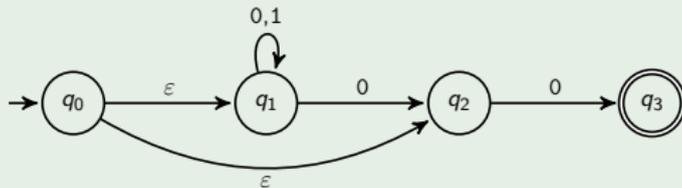
# Example: Recognized Language

## Example



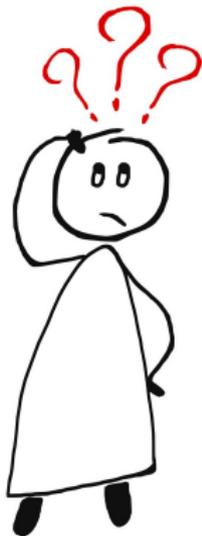
# Example: Recognized Language

## Example



The NFA recognizes the language  
 $\{w \in \{0, 1\}^* \mid w = 0 \text{ or } w \text{ ends with } 00\}$ .

# Questions



Questions?

# Summary

# Summary

- **DFAs** are automata where **every state transition is uniquely determined**.
- **NFAs** can have zero, one or more transitions for a given state and input symbol.
- **NFAs** can have  $\epsilon$ -transitions that can be taken without reading a symbol from the input.
- **NFAs** accept a word if there is **at least one accepting sequence of states**.