Lecture 3: DFAs and Regular Languages

CSC 320: Foundations of Computer Science

Quinton Yong

quintonyong@uvic.ca

University of Victoria

Is the set of all languages / decision problems countable?

- Firstly, let's consider **how many possible strings** there are over an alphabet $\boldsymbol{\Sigma}$
  - i.e. how large is $\boldsymbol{\Sigma}^*$ for any alphabet $\boldsymbol{\Sigma}$

- Clearly, the alphabet $\boldsymbol{\Sigma}$ is **countable** since it is a **finite set**

- **Claim**: $\boldsymbol{\Sigma}^*$ is countably infinite
- **Proof**: We can enumerate Σ^*. Example, consider $\Sigma = \{0, 1\}$.

$\boldsymbol{\varepsilon}$	$\mathbf{0}$	$\mathbf{1}$	$\mathbf{00}$	$\mathbf{01}$	$\mathbf{10}$	$\mathbf{11}$	$\mathbf{000}$	…
1	2	3	4	5	6	7	8	…

Is the set of all languages / decision problems countable?

- Now, how large is the set of all possible languages over an alphabet $\boldsymbol{\Sigma}$?
- We know that the set of possible strings $\boldsymbol{\Sigma}^*$ is countable (countably infinite)
- Languages are **subsets** of $\boldsymbol{\Sigma}^*$ (select strings which are accepted)
  - Every language is countably infinite or finite
- So, the **set of all languages** over $\boldsymbol{\Sigma}$ is the **set of all subsets of** $\boldsymbol{\Sigma}^*$ (powerset)

$$\boldsymbol{\mathcal{P}}(\boldsymbol{\Sigma}^*)$$

- Thus, the size of the set of all languages over alphabet $\boldsymbol{\Sigma}$ is $|\boldsymbol{\mathcal{P}}(\boldsymbol{\Sigma}^*)|$

Is the set of all languages / decision problems countable?

- We will prove that the **powerset** of any **countably infinite set** is **uncountable**
  - This would imply that $\mathcal{P}(\Sigma^*)$ is uncountable

- Since Σ^* is **countably infinite**, it has a **bijection** with $\mathbb{N}$

- We can just prove that $\mathcal{P}(\mathbb{N})$ is uncountably infinite

Barber Paradox

The barber **shaves everyone** who **doesn't shave themselves**.
Who shaves the barber?

- If the barber **shaves themself**, then the barber **doesn't shave the** barber…
- If the barber **doesn't shave themself**, then the barber **shaves the** barber…

This is a **paradox**.

We will use this idea in the proof.

$\mathcal{P}(\mathbb{N})$ is uncountable

- Assume for a contradiction that $\mathcal{P}(\mathbb{N})$ is countable

- We **list every subset** of $\mathbb{N}$ as $S_1, S_2, S_3, \ldots$ such that every possible subset of $\mathbb{N}$ is equal to a subset S_i for some i

$$S_1 = \{\ldots\}$$
$$S_2 = \{\ldots\}$$
$$S_3 = \{\ldots\}$$
$$\vdots$$

- Consider the subset $D \subseteq \mathbb{N}$ where $D = \{i \in \mathbb{N} \mid i \notin S_i\}$
  - e.g. if S_1 **does not contain** the number **1**, then **1 is in** D
  - e.g. if S_5 **does not contain** the number **5**, then **5 is in** D
  - e.g. if S_8 **contains** the number **8**, then **8 is not in** D

$\mathcal{P}(\mathbb{N})$ is uncountable

- Consider the subset $\boldsymbol{D} \subseteq \mathbb{N}$ where $\boldsymbol{D} = \{\boldsymbol{i} \in \mathbb{N} \mid \boldsymbol{i} \notin \boldsymbol{S_i}\}$
- For example, suppose the list of **every subset of** $\mathbb{N}$ was as follows:

$$\boldsymbol{S_1} = \{\, \mathbf{2}, \mathbf{4}, \mathbf{6}, \dots \}$$
$$\boldsymbol{S_2} = \{\, \mathbf{1}, \mathbf{2}, \mathbf{21}, \dots \}$$
$$\boldsymbol{S_3} = \{\, \mathbf{1}, \mathbf{8}, \mathbf{13}, \dots \}$$
$$\boldsymbol{S_4} = \{\, \mathbf{3}, \mathbf{61}, \mathbf{152}, \dots \}$$
$$\boldsymbol{S_5} = \{\, \mathbf{5}, \mathbf{10}, \mathbf{15}, \dots \}$$
$$\vdots$$

- $\boldsymbol{D}$ would be $\{\mathbf{1}, \mathbf{3}, \mathbf{4}, \dots\}$
  - $\mathbf{1}, \mathbf{3}, \mathbf{4} \in \boldsymbol{D}$ since $\mathbf{1} \notin \boldsymbol{S_1}$, $\mathbf{3} \notin \boldsymbol{S_3}$, and $\mathbf{4} \notin \boldsymbol{S_4}$
  - $\mathbf{2}, \mathbf{5} \notin \boldsymbol{D}$ since $\mathbf{2} \in \boldsymbol{S_2}$ and $\mathbf{5} \in \boldsymbol{S_5}$

$\mathcal{P}(\mathbb{N})$ is uncountable

- Consider the subset $\boldsymbol{D} \subseteq \mathbb{N}$ where $\boldsymbol{D} = \{\boldsymbol{i} \in \mathbb{N} | \ \boldsymbol{i} \notin \boldsymbol{S_i}\}$
- Since $\boldsymbol{D} \subseteq \mathbb{N}$, $\boldsymbol{D}$ is **in the list** of subsets and there is an $\boldsymbol{S_j}$ such that $\boldsymbol{S_j} = \boldsymbol{D}$

$$\boldsymbol{S_1} = \{ \dots \}$$
$$\boldsymbol{S_2} = \{ \dots \}$$
$$\vdots$$
$$\boldsymbol{S_j} = \boldsymbol{D} = \{ \dots \}$$

- Is the number $\boldsymbol{j}$ in the set $\boldsymbol{D}$ (which is $\boldsymbol{S_j}$)?
  - If $\boldsymbol{D}$ **contains** $\boldsymbol{j}$, then by definition, $\boldsymbol{j}$ **is not in** $\boldsymbol{D}$
  - If $\boldsymbol{D}$ **does not contains** $\boldsymbol{j}$, then by definition, $\boldsymbol{j}$ **is in** $\boldsymbol{D}$

paradox

- Thus, $\boldsymbol{D}$ **cannot exist** and cannot be listed.

$\mathcal{P}(\mathbb{N})$ is uncountable

- The way we defined $\boldsymbol{D}$ was a **valid subset** of $\mathbb{N}$
- However, we showed that it **cannot exist**, so we **cannot list every subset** of $\mathbb{N}$
- Thus, we have a **contradiction**
- Therefore, $\mathcal{P}(\mathbb{N})$ is **uncountable**

Recap:

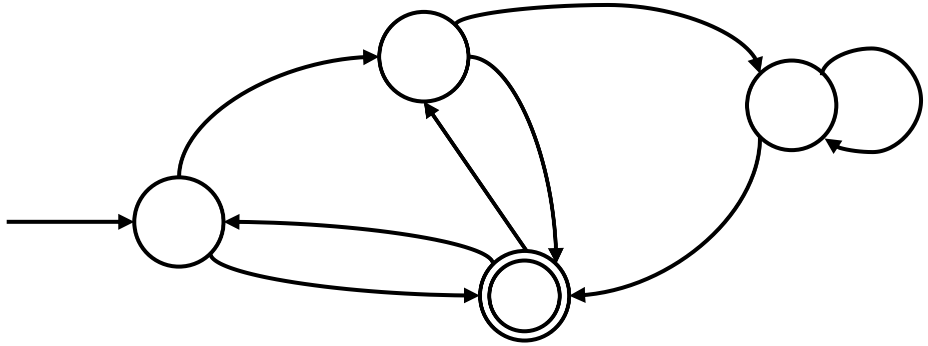
- By showing $\mathcal{P}(\mathbb{N})$ is uncountable, we also have that $\mathcal{P}(\boldsymbol{\Sigma}^*)$ is uncountable
- Therefore, the **set of all languages / decidable problems** is uncountably infinite

- This means there are an **uncountably infinite number** of problems

Finite Automata

- In this course, we want to understand and evaluate **computability** and **computational limits**
- To do that, we need to have **model(s)** of a computer which capture its **computational power**

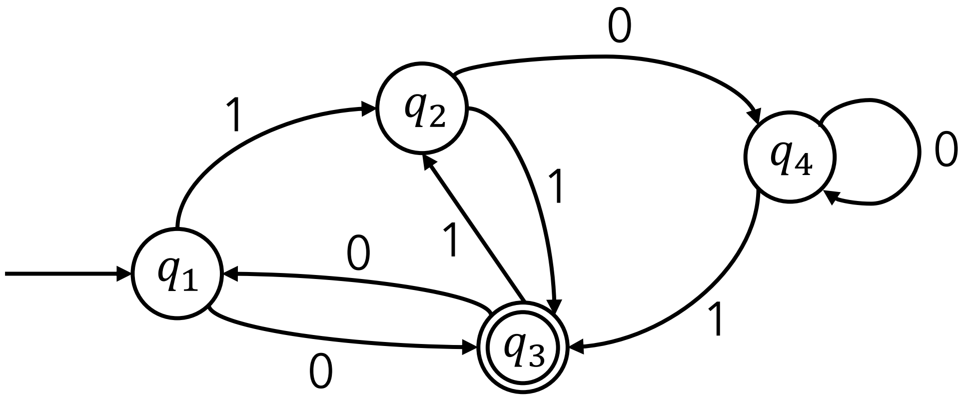
- The simplest model is a **finite state machine** or **finite automaton**
- Very little finite amount of memory independent of problem size



- **Real world examples:** Automatic doors, elevator, household appliances, substring search

State Diagram

State diagrams are used to **describe** finite automata

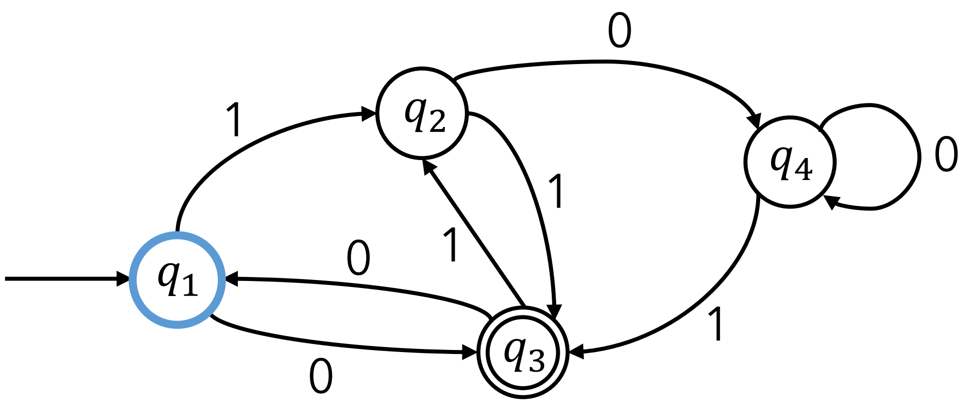


This state diagram contains:

- **States**: $\{q_1, q_2, q_3, q_4\}$
- **Start state**: q_1 (indicated by start arrow)
- **Accept state**: $\{q_3\}$ (indicated by double circle)
- **Transitions**: arrow from state to state (according to received input)
- **Inputs**: labels on transitions (symbols from alphabet, in this case $\Sigma = \{0, 1\}$)

State Diagram

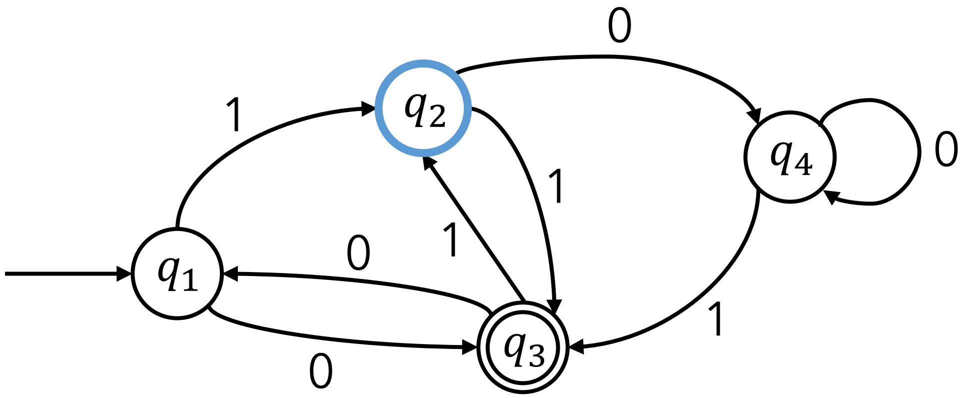
State diagrams are used to **describe** finite automata



What happens when we input string **11011**?

State Diagram

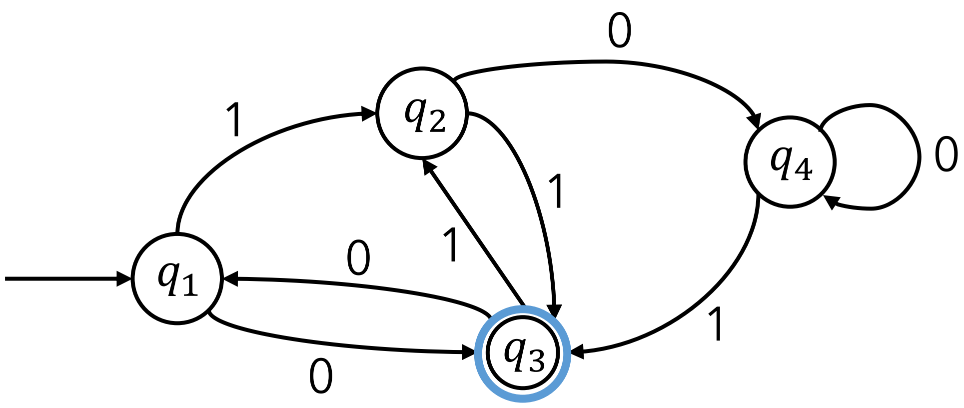
State diagrams are used to **describe** finite automata



What happens when we input string **11011**?

State Diagram

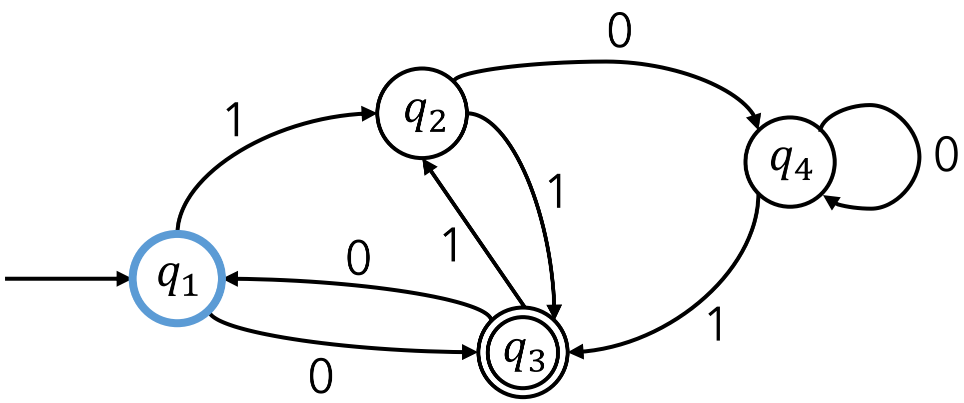
State diagrams are used to **describe** finite automata



What happens when we input string **11011**?

State Diagram

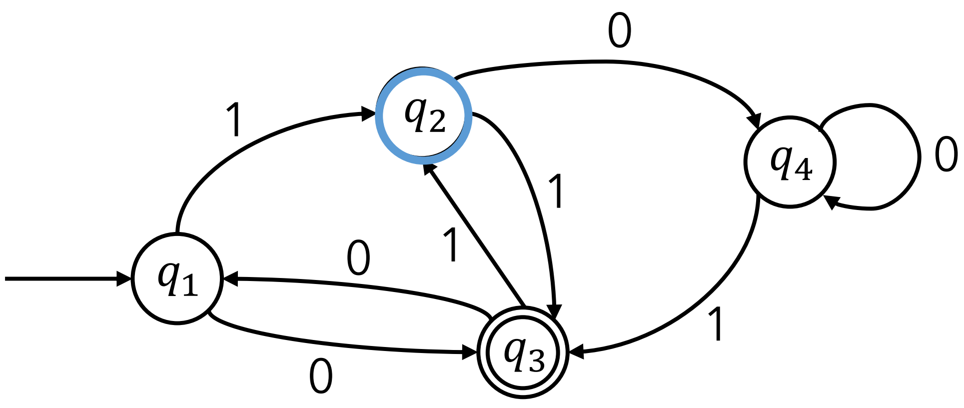
State diagrams are used to **describe** finite automata



What happens when we input string **11011**?

State Diagram

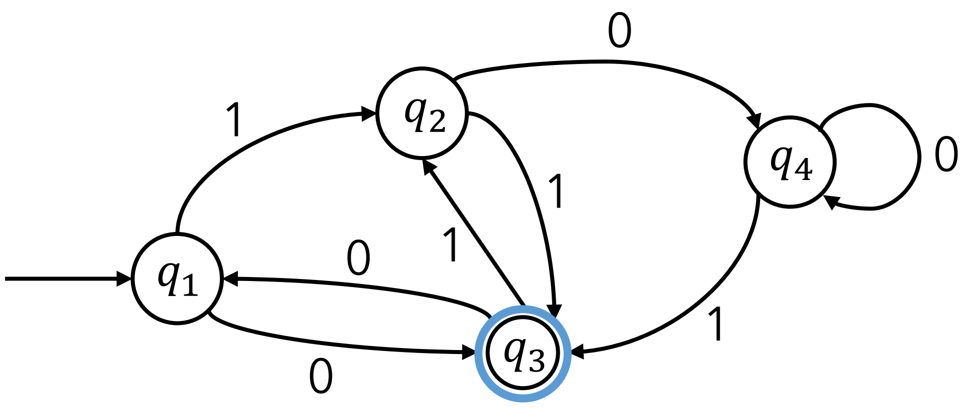
State diagrams are used to **describe** finite automata



What happens when we input string **11011**?

State Diagram

State diagrams are used to **describe** finite automata

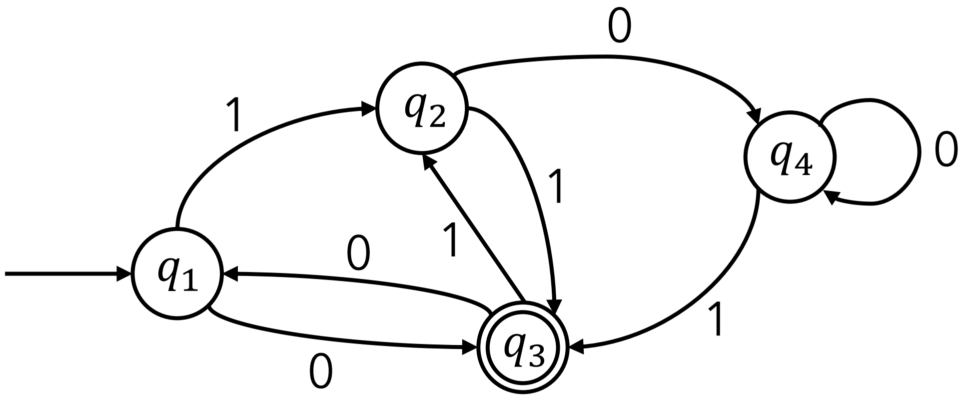


What happens when we input string **11011**?

- Once we **read each symbol** of the input, we land on an **accept state**
- This finite state machine **accepts** the string

State Diagram

State diagrams are used to **describe** finite automata



What other strings are accepted by this finite automaton?

- Is the **empty string $\boldsymbol{\varepsilon}$** accepted?
- Is the string **00**?
- Is the string **000**?

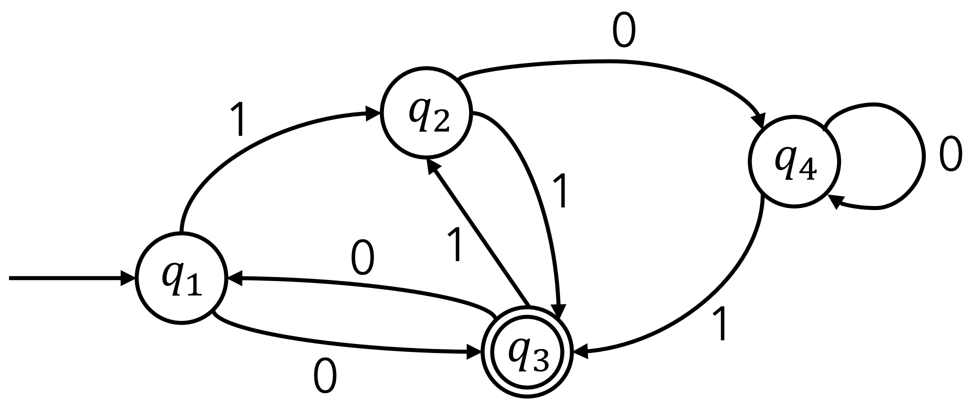
What **language is recognized** by this finite automaton (i.e. what are all the strings which are accepted)?

Formal Definition: Deterministic Finite Automaton

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

- $\boldsymbol{Q}$ is a **finite set** called the **states**
- $\boldsymbol{\Sigma}$ is a **finite set** called the **alphabet**
- $\boldsymbol{\delta}$: $\boldsymbol{Q} \times \boldsymbol{\Sigma} \to \boldsymbol{Q}$ is the **transition function** (i.e. $\boldsymbol{\delta}(\mathbf{curr\_state}, \mathbf{symbol}) = \mathbf{next\_state}$)
- $\boldsymbol{q_0} \in \boldsymbol{Q}$ is the **start state**
- $\boldsymbol{F} \subseteq \boldsymbol{Q}$ is the **set of accept states**

State Diagram



DFA Definition

$(\{q_1, q_2, q_3, q_4\}, \{0, 1\}, \delta, q_1, \{q_3\})$with
$\delta: Q \times \Sigma \to Q$ defined by

input

$\boldsymbol{\delta}$	$\mathbf{0}$	$\boxed{\mathbf{1}}$
$\boxed{q_1}$	q_3	$\boxed{q_2}$
q_2	q_4	q_3
q_3	q_1	q_2
q_4	q_4	q_3

current

next

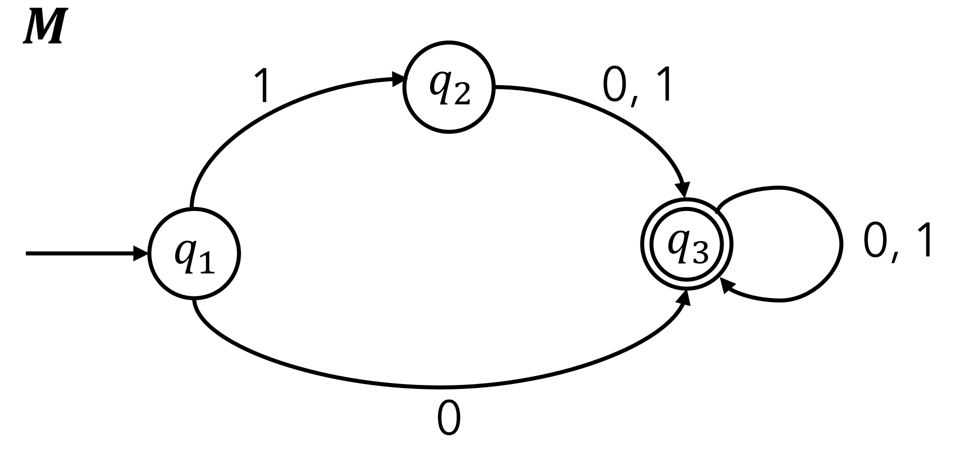
Language of a DFA

Let $\boldsymbol{M} = (\boldsymbol{Q}, \boldsymbol{\Sigma}, \boldsymbol{\delta}, \boldsymbol{q_0}, \boldsymbol{F})$ be a DFA and $\boldsymbol{A}$ be the **set of all strings that M accepts**

- $\boldsymbol{A}$ is called the **language** of machine $\boldsymbol{M}$
- $\boldsymbol{L}(\boldsymbol{M}) = \boldsymbol{A}$ (i.e. $L(M)$ denotes **the language of M**)
- $\boldsymbol{M}$ **recognizes** the language A

Note: a machine that **accepts no string** recognizes the **empty language** $\emptyset$

Example: Language $L(M)$ of a DFA M



$(\{q_1, q_2, q_3\}, \{0, 1\}, \delta, q_1, \{q_3\})$with
$\delta: Q \times \Sigma \rightarrow Q$ defined by

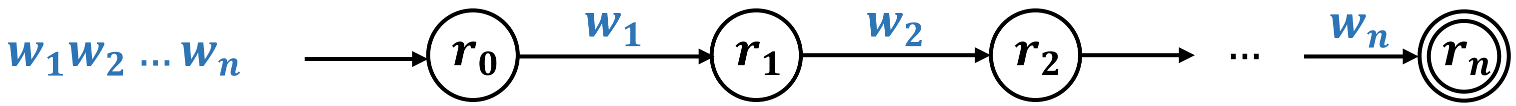
$\boldsymbol{\delta}$	$\mathbf{0}$	$\mathbf{1}$
q_1	q_3	q_2
q_2	q_3	q_3
q_3	q_1	q_2

What is the language of DFA $\boldsymbol{M}$?

$\boldsymbol{L}(\boldsymbol{M}) = \{w \in \Sigma^* \mid |\boldsymbol{w}| \geq 1, \text{if } \boldsymbol{w} \text{ starts with } 1 \text{ then } |\boldsymbol{w}| \geq 2\}$

DFA: Formal Definition of Computation

- Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA and let $w = w_1 w_2 \dots w_n$ be a string over Σ
- Then M **accepts** w if there is a sequence of states $r_0, r_1, r_2, \dots, r_n$ in Q such that
  1. $r_0 = q_0$
  2. $\delta(r_i, w_{i+1}) = r_{i+1}$
  3. $r_n \in F$

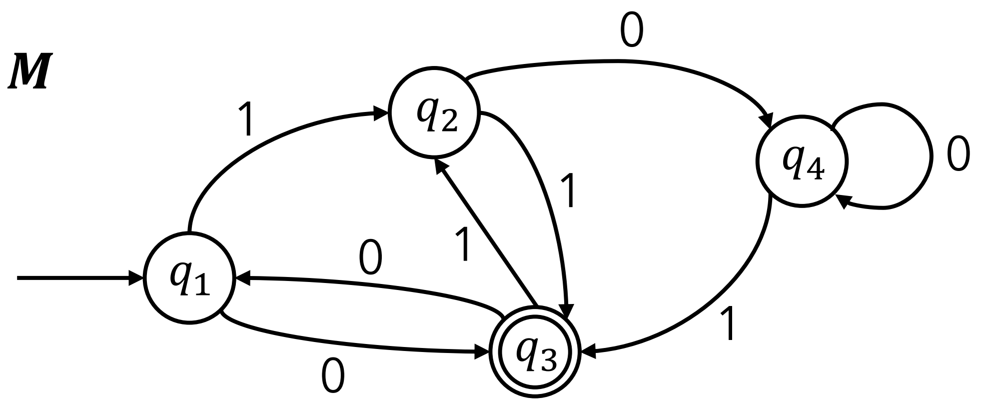


- M recognizes language L if $L = L(M) = \{ w \in \Sigma^* \mid M \text{ accepts } w \}$

Computation of DFA – Sequence of States

- What is the sequence of states when the following DFA $\boldsymbol{M}$ computes

$$\boldsymbol{w = 001101}$$

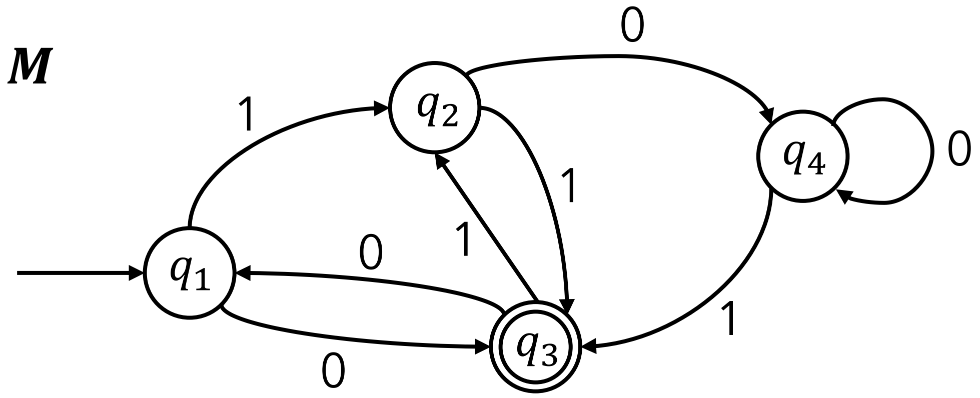


- $q_1\,,\ q_3\,,\ q_1\,,\ q_2\,,\ q_3\,,\ q_1\,,\ q_2$
- Since q_2 is not a final state, $\boldsymbol{w}$ is **not accepted**
- $\boldsymbol{w \notin L(M)}$

Computation of DFA – Sequence of States

- What is the sequence of states when the following DFA $\boldsymbol{M}$ computes

$$\boldsymbol{w = 0011011}$$



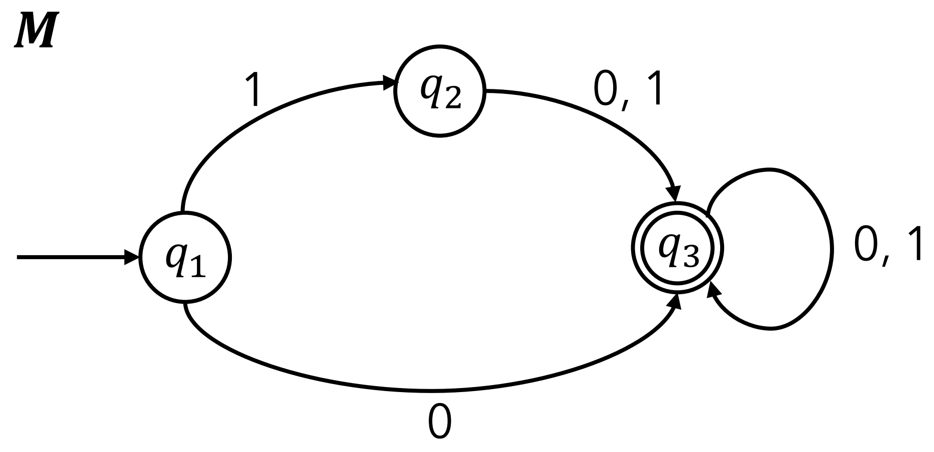
- q_1 , q_3 , q_1 , q_2 , q_3 , q_1 , q_2 , q_3
- Since q_3 is a final state, $\boldsymbol{w}$ is **accepted**
- $\boldsymbol{w} \in \boldsymbol{L(M)}$

Regular Languages

- A language L is called a **regular language** if there **exists a deterministic finite automaton that recognizes** L

- The **set of all languages** recognized by **the set of all possible DFAs** is called the **regular languages**

Regular Languages

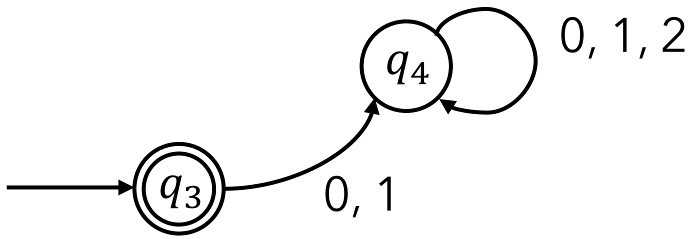


- In this DFA $\boldsymbol{M}$ which we saw previously, the language of $\boldsymbol{M}$ is
$$\boldsymbol{L}(\boldsymbol{M}) = \{w \in \Sigma^* \mid |\boldsymbol{w}| \geq 1, \text{if } \boldsymbol{w} \text{ starts with } 1 \text{ then } |\boldsymbol{w}| \geq 2\}$$

- Since there exists a DFA which recognizes $\boldsymbol{L}(\boldsymbol{M})$, $\boldsymbol{L}(\boldsymbol{M})$ is a **regular language**

DFA Example 1

Is this the state diagram for a valid DFA?



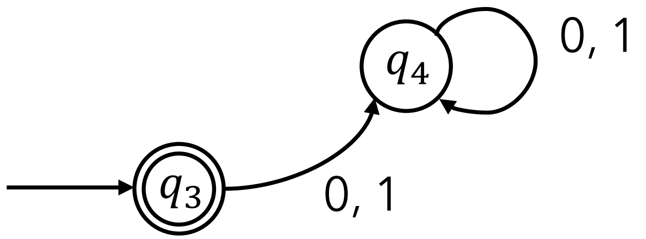
Try to write the **formal DFA definition** (i.e. the 5-tuple): $\boldsymbol{M = (Q, \Sigma, \delta, q_0, F)}$ where

- $Q = \{q_3, q_4\}$
- $\Sigma = \{0, 1, 2\}$
- $q_0 = q_3$
- $F = \{q_3\}$

- However, in a DFA, the transition function $\boldsymbol{\delta: Q \times \Sigma \to Q}$ must have an **outgoing transition** for **each alphabet symbol**
- This state diagram does not correspond to a valid DFA since $\boldsymbol{q_3}$ does not have an **outgoing transition for 2**

DFA Example 2

Is this the state diagram for a valid DFA?



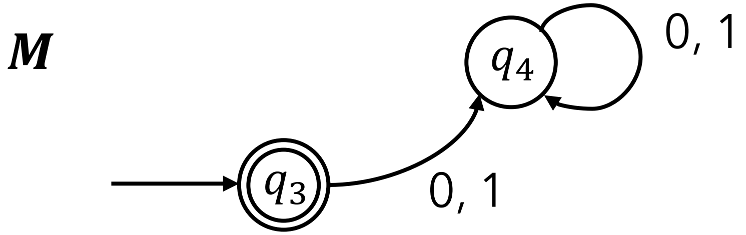
Yes. The DFA is $\boldsymbol{M} = (\boldsymbol{Q}, \boldsymbol{\Sigma}, \boldsymbol{\delta}, \boldsymbol{q_0}, \boldsymbol{F})$ where

- $Q = \{q_3, q_4\}$
- $\Sigma = \{0, 1\}$
- $q_0 = q_3$
- $F = \{q_3\}$
- δ is as described by the state diagram

Note: When asked to provide a DFA, **you must give the 5-tuple** (the state diagram by itself is not a formal DFA). For the **transition function**, unless specified, you can provide the table or say "described by the state diagram".

DFA Example 2

What is the **language** of the DFA corresponding to this state diagram?



- $\boldsymbol{L(M) = \{\varepsilon\}}$

- Note that this is **not the same** as writing $\boldsymbol{L(M)} = \emptyset$
- This DFA accepts the empty string
- $\boldsymbol{L(M)}$ would be $\emptyset$ if the DFA doesn't accept any string