Lesson 1. Finite state automata
CSIE 3110 – Formal Languages and Automata Theory

Tony Tan
Department of Computer Science and Information Engineering
College of Electrical Engineering and Computer Science
National Taiwan University
Table of contents

1. Deterministic finite state automata

2. Non-deterministic finite state automata

3. Pumping lemma
1. Deterministic finite state automata

2. Non-deterministic finite state automata

3. Pumping lemma
Deterministic finite state automata (DFA)

(Def.) A deterministic finite state automaton (DFA) is a system $A = \langle \Sigma, Q, q_0, F, \delta \rangle$, where each component is as follows.
Deterministic finite state automata (DFA)

(Def.) A *deterministic finite state automaton* (DFA) is a system \( A = \langle \Sigma, Q, q_0, F, \delta \rangle \), where each component is as follows.

- \( \Sigma \) is an alphabet.
- \( Q \) is a finite (non-empty) set of states.
- \( q_0 \in Q \) is the initial state.
- \( F \subseteq Q \) is a set of accepting states.
- \( \delta : Q \times \Sigma \rightarrow Q \) is the transition function.

In this case, we will say that "\( A \) is a DFA over alphabet \( \Sigma \)," or that "the alphabet of \( A \) is \( \Sigma \)."
Deterministic finite state automata (DFA)

(Def.) A deterministic finite state automaton (DFA) is a system $A = \langle \Sigma, Q, q_0, F, \delta \rangle$, where each component is as follows.

- $\Sigma$ is an alphabet.
- $Q$ is a finite (non-empty) set of states.
Deterministic finite state automata (DFA)

(Def.) A deterministic finite state automaton (DFA) is a system \( A = \langle \Sigma, Q, q_0, F, \delta \rangle \), where each component is as follows.

- \( \Sigma \) is an alphabet.
- \( Q \) is a finite (non-empty) set of states.
- \( q_0 \in Q \) is the initial state.
- \( F \subseteq Q \) is a set of accepting states.
- \( \delta : Q \times \Sigma \to Q \) is the transition function.

In this case, we will say that "\( A \) is a DFA over alphabet \( \Sigma \)," or that "the alphabet of \( A \) is \( \Sigma \)."
Deterministic finite state automata (DFA)

(Def.) A deterministic finite state automaton (DFA) is a system \( \mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle \), where each component is as follows.

- \( \Sigma \) is an alphabet.
- \( Q \) is a finite (non-empty) set of states.
- \( q_0 \in Q \) is the initial state.
- \( F \subseteq Q \) is a set of accepting states.
A deterministic finite state automaton (DFA) is a system \( \mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle \), where each component is as follows.

- \( \Sigma \) is an alphabet.
- \( Q \) is a finite (non-empty) set of states.
- \( q_0 \in Q \) is the initial state.
- \( F \subseteq Q \) is a set of accepting states.
- \( \delta : Q \times \Sigma \rightarrow Q \) is the transition function.
Deterministic finite state automata (DFA)

(Def.) A deterministic finite state automaton (DFA) is a system \( \mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle \), where each component is as follows.

- \( \Sigma \) is an alphabet.
- \( Q \) is a finite (non-empty) set of states.
- \( q_0 \in Q \) is the initial state.
- \( F \subseteq Q \) is a set of accepting states.
- \( \delta : Q \times \Sigma \rightarrow Q \) is the transition function.

In this case, we will say that “\( \mathcal{A} \) is a DFA over alphabet \( \Sigma \),” or that “the alphabet of \( \mathcal{A} \) is \( \Sigma \).”
Example 1

Consider the following $A = (\Sigma, Q, q_0, F, \delta)$:

- $\Sigma = \{a, b\}$
- $Q = \{q, p, r\}$ is the set of states.
- $r$ is the initial state.
- $F = \{p, q\}$ is the set of accepting states.
- The transition function $\delta$ is defined as:

\[
\begin{align*}
\delta(p, a) &= p & \delta(p, b) &= r \\
\delta(q, a) &= p & \delta(q, b) &= p \\
\delta(r, a) &= q & \delta(r, b) &= r
\end{align*}
\]
Example 1

Consider the following $A = (\Sigma, Q, q_0, F, \delta)$:

- $\Sigma = \{a, b\}$
- $Q = \{q, p, r\}$ is the set of states.
- $r$ is the initial state.
- $F = \{p, q\}$ is the set of accepting states.
- The transition function $\delta$ is defined as:
  
  $\delta(p, a) = p \quad \delta(p, b) = r$
  
  $\delta(q, a) = p \quad \delta(q, b) = p$
  
  $\delta(r, a) = q \quad \delta(r, b) = r$

This is a valid DFA.
Example 2

Consider the following $A = (\Sigma, Q, q_0, F, \delta)$:

- $\Sigma = \{a, b\}$

- $Q = \{q, p, r\}$ is the set of states.

- $r$ is the initial state.

- $F = \emptyset$, i.e., it does not have any accepting state.

- The transition function $\delta$ is defined as:

  $\delta(p, a) = p$  \quad $\delta(p, b) = r$

  $\delta(q, a) = p$  \quad $\delta(q, b) = p$

  $\delta(r, a) = q$  \quad $\delta(r, b) = r$
Example 2

Consider the following $A = \langle \Sigma, Q, q_0, F, \delta \rangle$:

- $\Sigma = \{a, b\}$

- $Q = \{q, p, r\}$ is the set of states.

- $r$ is the initial state.

- $F = \emptyset$, i.e., it does not have any accepting state.

- The transition function $\delta$ is defined as:

  $\delta(p, a) = p$ \hspace{0.5cm} $\delta(p, b) = r$
  
  $\delta(q, a) = p$ \hspace{0.5cm} $\delta(q, b) = p$
  
  $\delta(r, a) = q$ \hspace{0.5cm} $\delta(r, b) = r$

This is also a valid DFA.
Example 3

Consider the following $A = \langle \Sigma, Q, q_0, F, \delta \rangle$:

- $\Sigma = \{a\}$.

- $Q = \{q, p, r\}$ is the set of states.

- $r$ is the initial state.

- $F = \emptyset$, i.e., it does not have any accepting state.

- The transition function $\delta$ is defined as:
  
  $\delta(p, a) = p$
  $\delta(q, a) = p$
  $\delta(r, a) = q$
Example 3

Consider the following $A = (\Sigma, Q, q_0, F, \delta)$:

- $\Sigma = \{a\}$.

- $Q = \{q, p, r\}$ is the set of states.

- $r$ is the initial state.

- $F = \emptyset$, i.e., it does not have any accepting state.

- The transition function $\delta$ is defined as:

  $\delta(p, a) = p$
  $\delta(q, a) = p$
  $\delta(r, a) = q$

This is also a valid DFA.
Consider the following $A = (\Sigma, Q, q_0, F, \delta)$:

- $\Sigma = \emptyset$, i.e., the alphabet does not contain any symbol.
- $Q = \{q, p, r\}$ is the set of states.
- $r$ is the initial state.
- $F = \{p, q\}$, i.e., it does not have any accepting state.
- The transition function $\delta$ is not defined since $\Sigma = \emptyset$. 

This is not a valid DFA, since the alphabet $\Sigma$ must contain at least one symbol.
Example 4

Consider the following $A = \langle \Sigma, Q, q_0, F, \delta \rangle$:

- $\Sigma = \emptyset$, i.e., the alphabet does not contain any symbol.
- $Q = \{q, p, r\}$ is the set of states.
- $r$ is the initial state.
- $F = \{p, q\}$, i.e., it does not have any accepting state.
- The transition function $\delta$ is not defined since $\Sigma = \emptyset$.

This is not a valid DFA, since the alphabet $\Sigma$ must contain at least one symbol.
Example 5

Consider the following $A = \langle \Sigma, Q, q_0, F, \delta \rangle$:

- $\Sigma = \{0, 1\}$.
- $Q = \{q, p, r\}$ is the set of states.
- $r$ is the initial state.
- $F = \{p, q\}$.
- The transition function $\delta$ is defined as:

  $\delta(p, a) = p$  \hspace{1cm} $\delta(p, b) = r$
  $\delta(q, a) = p$  \hspace{1cm} $\delta(q, b) = p$
  $\delta(r, a) = q$  \hspace{1cm} $\delta(r, b) = r$

This is not a valid DFA, since the transition function $\delta$ is defined on $Q \times \{a, b\}$, but the alphabet should be $\{0, 1\}$.
Consider the following $A = (\Sigma, Q, q_0, F, \delta)$:

- $\Sigma = \{0, 1\}$.
- $Q = \{q, p, r\}$ is the set of states.
- $r$ is the initial state.
- $F = \{p, q\}$.
- The transition function $\delta$ is defined as:

\[
\begin{align*}
\delta(p, a) &= p & \delta(p, b) &= r \\
\delta(q, a) &= p & \delta(q, b) &= p \\
\delta(r, a) &= q & \delta(r, b) &= r
\end{align*}
\]

This is not a valid DFA, since the transition function $\delta$ is defined on $Q \times \{a, b\}$, but the alphabet should be $\{0, 1\}$.
Example 6

Consider the following $A = \langle \Sigma, Q, q_0, F, \delta \rangle$:

- $\Sigma = \{0, 1\}$.
- $Q = \{q, p, r\}$ is the set of states.
- $r$ is the initial state.
- $F = \{p, q\}$.
- The transition function $\delta$ is defined as:
  
  \[
  \begin{align*}
  \delta(p, 0) &= p & \delta(p, 1) &= r \\
  \delta(q, 0) &= p & \delta(q, 1) &= p \\
  \delta(r, 0) &= q
  \end{align*}
  \]

This is not a valid DFA, since $\delta$ is not defined on $(r, 1)$.
Example 6

Consider the following $A = (\Sigma, Q, q_0, F, \delta)$:

- $\Sigma = \{0, 1\}$.
- $Q = \{q, p, r\}$ is the set of states.
- $r$ is the initial state.
- $F = \{p, q\}$.
- The transition function $\delta$ is defined as:
  
  \[
  \begin{align*}
  \delta(p, 0) &= p & & \delta(p, 1) &= r \\
  \delta(q, 0) &= p & & \delta(q, 1) &= p \\
  \delta(r, 0) &= q & &
  \end{align*}
  \]

This is not a valid DFA, since $\delta$ is not defined on $(r, 1)$. 
Example 7

Consider the following $A = (\Sigma, Q, q_0, F, \delta)$:

- $\Sigma = \{0, 1\}$.
- $Q = \{q, p, r\}$ is the set of states.
- There is no initial state.
- $F = \{p, q\}$.
- The transition function $\delta$ is defined as:

$$
\begin{align*}
\delta(p, 0) &= p & \delta(p, 1) &= r \\
\delta(q, 0) &= p & \delta(q, 1) &= p \\
\delta(r, 0) &= q & \delta(r, 1) &= q
\end{align*}
$$

This is not a valid DFA, because DFA must have the initial state.
Example 7

Consider the following $A = (\Sigma, Q, q_0, F, \delta)$:

- $\Sigma = \{0, 1\}$.
- $Q = \{q, p, r\}$ is the set of states.
- There is no initial state.
- $F = \{p, q\}$.
- The transition function $\delta$ is defined as:
  
  $\delta(p, 0) = p \quad \delta(p, 1) = r$
  
  $\delta(q, 0) = p \quad \delta(q, 1) = p$
  
  $\delta(r, 0) = q \quad \delta(r, 1) = q$

This is not a valid DFA, because DFA must have the initial state.
Example 8

Consider the following $A = (\Sigma, Q, q_0, F, \delta)$:

- $\Sigma = \{0, 1\}$. 
- $Q = \{q, p, r\}$ is the set of states.
- $p$ and $r$ are the initial states.
- $F = \{p, q\}$.
- The transition function $\delta$ is defined as:

$$
\begin{align*}
\delta(p, 0) &= p & \delta(p, 1) &= r \\
\delta(q, 0) &= p & \delta(q, 1) &= p \\
\delta(r, 0) &= q & \delta(r, 1) &= q
\end{align*}
$$

This is not a valid DFA, because DFA must have exactly one initial state.
Example 8

Consider the following $A = \langle \Sigma, Q, q_0, F, \delta \rangle$:

- $\Sigma = \{0, 1\}$.
- $Q = \{q, p, r\}$ is the set of states.
- $p$ and $r$ are the initial states.
- $F = \{p, q\}$.
- The transition function $\delta$ is defined as:

$$
\begin{align*}
\delta(p, 0) &= p & \delta(p, 1) &= r \\
\delta(q, 0) &= p & \delta(q, 1) &= p \\
\delta(r, 0) &= q & \delta(r, 1) &= q
\end{align*}
$$

This is not a valid DFA, because DFA must have exactly one initial state.
Visualizing DFA

Consider the following DFA \( \mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle \) over \( \Sigma = \{a, b\} \), where \( Q = \{q, p, r\} \), \( r \) is the initial state, \( F = \{p\} \) and \( \delta \) is defined as:

\[
\delta(p, a) = p \quad \delta(p, b) = r \quad \delta(q, a) = p \quad \delta(q, b) = p \quad \delta(r, a) = q \quad \delta(r, b) = r
\]
Consider the following DFA $A = \langle \Sigma, Q, q_0, F, \delta \rangle$ over $\Sigma = \{a, b\}$, where $Q = \{q, p, r\}$, $r$ is the initial state, $F = \{p\}$ and $\delta$ is defined as:

$$
\begin{align*}
\delta(p, a) &= p & \delta(p, b) &= r \\
\delta(q, a) &= p & \delta(q, b) &= p \\
\delta(r, a) &= q & \delta(r, b) &= r
\end{align*}
$$

We can visualize it as a directed graph:
Visualizing DFA

Consider the following DFA $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$ over $\Sigma = \{a, b\}$, where $Q = \{q, p, r\}$, $r$ is the initial state, $F = \{p\}$ and $\delta$ is defined as:

$\delta(p, a) = p \quad \delta(p, b) = r \quad \delta(q, a) = p \quad \delta(q, b) = p \quad \delta(r, a) = q \quad \delta(r, b) = r$

We can visualize it as a directed graph:

The initial state has incoming arrow $r$

$p \quad q$
Consider the following DFA $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$ over $\Sigma = \{a, b\}$, where $Q = \{q, p, r\}$, $r$ is the initial state, $F = \{p\}$ and $\delta$ is defined as:

$$
\delta(p, a) = p \quad \delta(p, b) = r \quad \delta(q, a) = p \quad \delta(q, b) = p \quad \delta(r, a) = q \quad \delta(r, b) = r
$$

We can visualize it as a directed graph:

- The initial state has incoming arrow
- The accepting state has double circle
Visualizing DFA

Consider the following DFA \( A = \langle \Sigma, Q, q_0, F, \delta \rangle \) over \( \Sigma = \{a, b\} \), where \( Q = \{q, p, r\} \), \( r \) is the initial state, \( F = \{p\} \) and \( \delta \) is defined as:

\[
\delta(p, a) = p \quad \delta(p, b) = r \quad \delta(q, a) = p \quad \delta(q, b) = p \quad \delta(r, a) = q \quad \delta(r, b) = r
\]

We can visualize it as a directed graph:

![Diagram of DFA]

The initial state has incoming arrow

The accepting state has double circle
Important note!

In your solution for homework and exams, don’t write DFA like this:

\[ A = \langle \Sigma, Q, q_0, F, \delta \rangle \] over \( \Sigma = \{a, b\} \), where \( Q = \{q, p, r\} \), \( r \) is the initial state, \( F = \{p\} \) and \( \delta \) is defined as:

\[
\begin{align*}
\delta(p, a) &= p & \delta(p, b) &= r \\
\delta(q, a) &= p & \delta(q, b) &= p \\
\delta(r, a) &= q & \delta(r, b) &= r
\end{align*}
\]
Important note!

In your solution for homework and exams, don’t write DFA like this:

\[ A = \langle \Sigma, Q, q_0, F, \delta \rangle \] over \( \Sigma = \{a, b\} \), where \( Q = \{q, p, r\} \), \( r \) is the initial state, \( F = \{p\} \) and \( \delta \) is defined as:

\[
\begin{align*}
\delta(p, a) &= p & \delta(p, b) &= r \\
\delta(q, a) &= p & \delta(q, b) &= p \\
\delta(r, a) &= q & \delta(r, b) &= r
\end{align*}
\]

But draw the graph representation of DFA like this:
What does DFA do?

A DFA can be viewed as a special kind of computer program/algorithm.
What does DFA do?

A DFA can be viewed as a special kind of computer program/algorithm.

Its input is always a finite string over its alphabet.
What does DFA do?

A DFA can be viewed as a special kind of computer program/algorithm.

Its input is always a finite string over its alphabet.

It moves from state to state depending on the input symbol that it reads.
What does DFA do?

A DFA can be viewed as a special kind of computer program/algorithm.

Its input is always a finite string over its alphabet.

It moves from state to state depending on the input symbol that it reads.

It starts from the initial state.
What does DFA do?

A DFA can be viewed as a special kind of computer program/algorithm.

Its input is always a finite string over its alphabet.

It moves from state to state depending on the input symbol that it reads.

It starts from the initial state.

A DFA either accepts/rejects its input.
What does DFA do?

A DFA can be viewed as a special kind of computer program/algorithm.

Its input is always a finite string over its alphabet.

It moves from state to state depending on the input symbol that it reads.

It starts from the initial state.

A DFA either accepts/rejects its input.

We can view “accept” as returning True and “reject” as returning False.
Example

On input string \textit{aba}:
\[ r \quad (\text{accepted by DFA}) \]

On input string \textit{aab}:
\[ r \quad (\text{not accepted by DFA}) \]

On input string \textit{ε}:
\[ r \quad (\text{not accepted by DFA}) \]
Example

On input string $aba$: 

```
On input string $aab$: 

On input string $\varepsilon$: 
```
On input string $aba$: $r$
Example

On input string $aba$: \[ r \quad a \quad q \]
On input string \(aba\): \(r\ a\ q\ b\ p\)
Example

On input string $aba$: $r \ a \ q \ b \ p \ a \ p$

On input string $aab$: $r \ a \ q \ a \ p \ b \ r$

On input string $\varepsilon$: $r$

(not accepted by DFA)
Example

On input string $aba$: $rpqbpap$ (accepted by DFA)
Example

On input string \(aba\): \(r \ a \ q \ b \ p \ a \ p\) (accepted by DFA)

On input string \(aab\):
Example

On input string \textit{aba}: \quad r \ a \ q \ b \ p \ a \ p \quad \text{(accepted by DFA)}

On input string \textit{aab}: \quad r
Example

On input string $aba$: \[ r \ a \ q \ b \ p \ a \ p \] (accepted by DFA)

On input string $aab$: \[ r \ a \ q \]
Example

On input string $aba$: $r \ a \ q \ b \ p \ a \ p$ (accepted by DFA)

On input string $aab$: $r \ a \ q \ a \ p$
Example

On input string \textit{aba}: \quad r \quad a \quad q \quad b \quad p \quad a \quad p \quad (accepted by DFA)

On input string \textit{aab}: \quad r \quad a \quad q \quad a \quad p \quad b \quad r
On input string $aba$: $r \ a \ q \ b \ p \ a \ p$ (accepted by DFA)

On input string $aab$: $r \ a \ q \ a \ p \ b \ r$ (not accepted by DFA)
On input string $aba$: $r \ a \ q \ b \ p \ a \ p$ (accepted by DFA)

On input string $aab$: $r \ a \ q \ a \ p \ b \ r$ (not accepted by DFA)

On input string $\varepsilon$:
Example

On input string \(aba\): \(r \ a \ q \ b \ p \ a \ p\)  (accepted by DFA)

On input string \(aab\): \(r \ a \ q \ a \ p \ b \ r\)  (not accepted by DFA)

On input string \(\varepsilon\): \(r\)
Example

On input string $aba$: $r \ a \ q \ b \ p \ a \ p$ (accepted by DFA)

On input string $aab$: $r \ a \ q \ a \ p \ b \ r$ (not accepted by DFA)

On input string $\varepsilon$: $r$ (not accepted by DFA)
The formal definition of acceptance/rejection of words by DFA

Let $A = \langle \Sigma, Q, q_0, F, \delta \rangle$.

(Def.) On input word $w = a_1 \cdots a_n$, \textit{the run of $A$ on $w$} is the sequence:

$$p_0 \ a_1 \ p_1 \ a_2 \ p_2 \ \cdots \ a_n \ p_n,$$

where $p_0 = q_0$ and $\delta(p_i, a_{i+1}) = p_{i+1}$, for each $i = 0, \ldots, n - 1$. 

(Def.) The run of $A$ on $w$ starting from state $q$ is defined as the sequence above, but with condition $p_0 = q$.

(Def.) A run is called an \textit{accepting} run, if $p_0 = q_0$ and $q_n \in F$. 

The formal definition of acceptance/rejection of words by DFA

Let $A = \langle \Sigma, Q, q_0, F, \delta \rangle$.

(Def.) On input word $w = a_1 \cdots a_n$, the run of $A$ on $w$ is the sequence:

$$p_0 \ a_1 \ p_1 \ a_2 \ p_2 \ \cdots \ \ a_n \ p_n,$$

where $p_0 = q_0$ and $\delta(p_i, a_{i+1}) = p_{i+1}$, for each $i = 0, \ldots, n - 1$.

(Def.) The run of $A$ on $w$ starting from state $q$ is defined as the sequence above, but with condition $p_0 = q$. 
The formal definition of acceptance/rejection of words by DFA

Let $A = \langle \Sigma, Q, q_0, F, \delta \rangle$.

(Def.) On input word $w = a_1 \cdots a_n$, the run of $A$ on $w$ is the sequence:

$$p_0 \ a_1 \ p_1 \ a_2 \ p_2 \ \cdots \ a_n \ p_n,$$

where $p_0 = q_0$ and $\delta(p_i, a_{i+1}) = p_{i+1}$, for each $i = 0, \ldots, n - 1$.

(Def.) The run of $A$ on $w$ starting from state $q$ is defined as the sequence above, but with condition $p_0 = q$.

(Def.) A run is called an accepting run, if $p_0 = q_0$ and $q_n \in F$. 
The language accepted by DFA

Let \( A = \langle \Sigma, Q, q_0, F, \delta \rangle \).

(Def.) We say that \( A \) accepts \( w \), if there is an accepting run of \( A \) on \( w \).
The language accepted by DFA

Let $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$.

(Def.) We say that $\mathcal{A}$ accepts $w$, if there is an accepting run of $\mathcal{A}$ on $w$.

(Def.) The language of all words accepted by $\mathcal{A}$ is denoted by $L(\mathcal{A})$. 
The language accepted by DFA

Let $A = \langle \Sigma, Q, q_0, F, \delta \rangle$.

(Def.) We say that $A$ accepts $w$, if there is an accepting run of $A$ on $w$.

(Def.) The language of all words accepted by $A$ is denoted by $L(A)$.

(Def.) A language $L$ is called a **regular** language, if there is a DFA $A$ such that $L(A) = L$. 

Some observations on DFA

(Rem. 1.2) Let $A = \langle \Sigma, Q, q_0, F, \delta \rangle$ be a DFA.
Some observations on DFA

(Rem. 1.2) Let $A = \langle \Sigma, Q, q_0, F, \delta \rangle$ be a DFA.

- For every word $w$, there is exactly one run of $A$ on $w$. 
Some observations on DFA

(Rem. 1.2) Let $A = \langle \Sigma, Q, q_0, F, \delta \rangle$ be a DFA.

- For every word $w$, there is exactly one run of $A$ on $w$.
- The empty string $\varepsilon$ is accepted by $A$ if and only if $q_0 \in F$. 
Another example: The language of the binary representations of $0 \mod 3$

A word $w \in \{0, 1\}^*$ can be viewed as a non-negative integer, denoted by $[w]$. 

Another example: The language of the binary representations of $0 \mod 3$

A word $w \in \{0, 1\}^*$ can be viewed as a non-negative integer, denoted by $[w]$.

- $[0] = [000] = 0$. 

We will show that $L_0$ is a regular language.
An example: The language of the binary representations of $0 \mod 3$

A word $w \in \{0, 1\}^*$ can be viewed as a non-negative integer, denoted by $[w]$.

- $[0] = [000] = 0$.
- $[1] = [01] = [00001] = 1$. 

We define $J_{\varepsilon} = 0$.

Define the following language $L_0$:

$L_0 := \{w | J_w \equiv 0 \pmod{3}\}$

We will show that $L_0$ is a regular language.
Another example: The language of the binary representations of 0 mod 3

A word $w \in \{0, 1\}^*$ can be viewed as a non-negative integer, denoted by $[w]$.

- $[0] = [000] = 0$.
- $[1] = [01] = [00001] = 1$.
- $[11001] = [0000011001] = 25$. 

We define $J[\varepsilon] = 0$.

Define the following language $L_0$:

$$L_0 := \{w | J[w] \equiv 0 \pmod{3}\}$$

We will show that $L_0$ is a regular language.
Another example: The language of the binary representations of $0 \mod 3$

A word $w \in \{0, 1\}^*$ can be viewed as a non-negative integer, denoted by $[w]$.

- $[0] = [000] = 0$.
- $[1] = [01] = [00001] = 1$.
- $[11001] = [0000011001] = 25$.
- We define $[\varepsilon] = 0$. 

Define the following language $L_0$:

$L_0 := \{ w | [w] \equiv 0 \pmod{3} \}$

We will show that $L_0$ is a regular language.
Another example: The language of the binary representations of $0 \mod 3$

A word $w \in \{0, 1\}^*$ can be viewed as a non-negative integer, denoted by $[w]$.

- $[0] = [000] = 0$.
- $[1] = [01] = [00001] = 1$.
- $[11001] = [0000011001] = 25$.
- We define $[\varepsilon] = 0$.

Define the following language $L_0$:

$$L_0 := \{ w \mid [w] \equiv 0 \pmod{3} \}$$
Another example: The language of the binary representations of $0 \mod 3$

A word $w \in \{0, 1\}^*$ can be viewed as a non-negative integer, denoted by $[w]$.

- $[0] = [000] = 0$.
- $[1] = [01] = [00001] = 1$.
- $[11001] = [000011001] = 25$.
- We define $[\epsilon] = 0$.

Define the following language $L_0$:

$$L_0 := \{ w \mid [w] \equiv 0 \pmod{3} \}$$

We will show that $L_0$ is a regular language.
Constructing a DFA for $L_0 := \{ w \mid \lfloor w \rfloor \equiv 0 \pmod{3} \}$

For a word $w \in \{0, 1\}^*$ and a symbol $z \in \{0, 1\}$, we have the following identity:
Constructing a DFA for $L_0 := \{ w \mid \left\lfloor w \right\rfloor \equiv 0 \pmod{3} \}$

For a word $w \in \{0, 1\}^*$ and a symbol $z \in \{0, 1\}$, we have the following identity:

$$\left\lfloor wz \right\rfloor = \left\lfloor w \right\rfloor \times 2 + z$$
Constructing a DFA for $L_0 := \{ w \mid \lfloor w \rfloor \equiv 0 \pmod{3} \}$

For a word $w \in \{0, 1\}^*$ and a symbol $z \in \{0, 1\}$, we have the following identity:

$$\lfloor wz \rfloor = \lfloor w \rfloor \times 2 + z$$

$$\lfloor wz \rfloor \equiv \lfloor w \rfloor \times 2 + z \pmod{3}$$
Constructing a DFA for \( L_0 := \{ w \mid [w] \equiv 0 \pmod{3} \} \)

For a word \( w \in \{0, 1\}^* \) and a symbol \( z \in \{0, 1\} \), we have the following identity:

\[
[wz] = [w] \times 2 + z
\]

\[
[wz] \equiv [w] \times 2 + z \pmod{3}
\]

\[
0 \equiv 0 \times 2 + 0 \pmod{3}
\]
Constructing a DFA for $L_0 := \{w \mid \lfloor w \rfloor \equiv 0 \pmod{3}\}$

For a word $w \in \{0, 1\}^*$ and a symbol $z \in \{0, 1\}$, we have the following identity:

\[
\lfloor wz \rfloor = \lfloor w \rfloor \times 2 + z
\]

\[
\lfloor wz \rfloor \equiv \lfloor w \rfloor \times 2 + z \pmod{3}
\]

\[
0 \equiv 0 \times 2 + 0 \pmod{3}
\]

\[
1 \equiv 0 \times 2 + 1 \pmod{3}
\]
Constructing a DFA for $L_0 := \{ w \mid [w] \equiv 0 \pmod{3} \}$

For a word $w \in \{0, 1\}^*$ and a symbol $z \in \{0, 1\}$, we have the following identity:

$$[wz] = [w] \times 2 + z$$

$$[wz] \equiv [w] \times 2 + z \pmod{3}$$

0 $\equiv 0 \times 2 + 0 \pmod{3}$

1 $\equiv 0 \times 2 + 1 \pmod{3}$

2 $\equiv 1 \times 2 + 0 \pmod{3}$
Constructing a DFA for $L_0 := \{w \mid [w] \equiv 0 \pmod{3}\}$

For a word $w \in \{0, 1\}^*$ and a symbol $z \in \{0, 1\}$, we have the following identity:

$$[wz] = [w] \times 2 + z$$

$$[wz] \equiv [w] \times 2 + z \pmod{3}$$

$$0 \equiv 0 \times 2 + 0 \pmod{3}$$

$$1 \equiv 0 \times 2 + 1 \pmod{3}$$

$$2 \equiv 1 \times 2 + 0 \pmod{3}$$

$$0 \equiv 1 \times 2 + 1 \pmod{3}$$
Constructing a DFA for \( L_0 := \{ w \mid [w] \equiv 0 \pmod{3} \} \)

For a word \( w \in \{0, 1\}^* \) and a symbol \( z \in \{0, 1\} \), we have the following identity:

\[
[wz] = [w] \times 2 + z
\]

\[
[wz] \equiv [w] \times 2 + z \pmod{3}
\]

\[
0 \equiv 0 \times 2 + 0 \pmod{3}
\]

\[
1 \equiv 0 \times 2 + 1 \pmod{3}
\]

\[
2 \equiv 1 \times 2 + 0 \pmod{3}
\]

\[
0 \equiv 1 \times 2 + 1 \pmod{3}
\]

\[
1 \equiv 2 \times 2 + 0 \pmod{3}
\]
Constructing a DFA for $L_0 := \{ w \mid \lfloor w \rfloor \equiv 0 \pmod{3} \}$

For a word $w \in \{0, 1\}^*$ and a symbol $z \in \{0, 1\}$, we have the following identity:

$$\lfloor wz \rfloor = \lfloor w \rfloor \times 2 + z$$

$$\lfloor wz \rfloor \equiv \lfloor w \rfloor \times 2 + z \pmod{3}$$

$$0 \equiv 0 \times 2 + 0 \pmod{3}$$

$$1 \equiv 0 \times 2 + 1 \pmod{3}$$

$$2 \equiv 1 \times 2 + 0 \pmod{3}$$

$$0 \equiv 1 \times 2 + 1 \pmod{3}$$

$$1 \equiv 2 \times 2 + 0 \pmod{3}$$

$$2 \equiv 2 \times 2 + 1 \pmod{3}$$
Constructing a DFA for $L_0 := \{ w \mid \llbracket w \rrbracket \equiv 0 \pmod{3} \}$

For a word $w \in \{0, 1\}^*$ and a symbol $z \in \{0, 1\}$, we have the following identity:

$$\llbracket wz \rrbracket = \llbracket w \rrbracket \times 2 + z$$

$$\llbracket wz \rrbracket \equiv \llbracket w \rrbracket \times 2 + z \pmod{3}$$

- $0 \equiv 0 \times 2 + 0 \pmod{3}$
- $1 \equiv 0 \times 2 + 1 \pmod{3}$
- $2 \equiv 1 \times 2 + 0 \pmod{3}$
- $0 \equiv 1 \times 2 + 1 \pmod{3}$
- $1 \equiv 2 \times 2 + 0 \pmod{3}$
- $2 \equiv 2 \times 2 + 1 \pmod{3}$

$A:$

```
0 ----> 1 ----> 2

0 ----> 1
1 ----> 2
```

\[ 0 \]
\[ 1 \]
\[ 2 \]
Constructing a DFA for $L_0 := \{ w \mid \lceil w \rceil \equiv 0 \pmod{3} \}$

\[ A: \]

```
A: 0 1 2
    1 0 1
0 1 0
1 2 1
2 0 1
```
Constructing a DFA for $L_0 := \{ w \mid [w] \equiv 0 \pmod{3} \}$

For every word $w \in \{0, 1\}^*$:

$A$ accepts $w$ if and only if $[w] \equiv 0 \pmod{3}$. 
Constructing a DFA for $L_0 := \{ w \mid [w] \equiv 0 \pmod{3} \}$

$L_0$ is a language over the alphabet $\{0, 1\}$.

A DFA $A$ accepting $L_0$ is shown in the diagram.

For every word $w \in \{0, 1\}^*$:

$A$ accepts $w$ if and only if $[w] \equiv 0 \pmod{3}$.

So $L(A) = L_0$. 
Theorem 1.3
Regular languages are closed under boolean operations, i.e., complement, intersection and union.
Important property of regular languages

**Theorem 1.3**

Regular languages are closed under boolean operations, i.e., complement, intersection and union.

More formally, Theorem 1.3 can be stated as follows.
Important property of regular languages

**Theorem 1.3**

Regular languages are closed under boolean operations, i.e., complement, intersection and union.

More formally, Theorem 1.3 can be stated as follows.

**Closure under complement:** For every DFA $A$ over alphabet $\Sigma$, there is a DFA $A'$ over the same alphabet $\Sigma$ such that $L(A') = \Sigma^* - L(A)$.
Theorem 1.3

Regular languages are closed under boolean operations, i.e., complement, intersection and union.

More formally, Theorem 1.3 can be stated as follows.

**Closure under complement:** For every DFA $A$ over alphabet $\Sigma$, there is a DFA $A'$ over the same alphabet $\Sigma$ such that $L(A') = \Sigma^* - L(A)$.

**Closure under intersection:** For every two DFA $A_1$ and $A_2$, there is a DFA $A'$ such that $L(A') = L(A_1) \cap L(A_2)$. 
Important property of regular languages

**Theorem 1.3**

Regular languages are closed under boolean operations, i.e., complement, intersection and union.

More formally, Theorem 1.3 can be stated as follows.

**Closure under complement:** For every DFA $A$ over alphabet $\Sigma$, there is a DFA $A'$ over the same alphabet $\Sigma$ such that $L(A') = \Sigma^* - L(A)$.

**Closure under intersection:** For every two DFA $A_1$ and $A_2$, there is a DFA $A'$ such that $L(A') = L(A_1) \cap L(A_2)$.

**Closure under union:** For every two DFA $A_1$ and $A_2$, there is a DFA $A'$ such that $L(A') = L(A_1) \cup L(A_2)$. 
Important property of regular languages

**Theorem 1.3**

Regular languages are closed under boolean operations, i.e., complement, intersection and union.

More formally, Theorem 1.3 can be stated as follows.

**Closure under complement:** For every DFA $A$ over alphabet $\Sigma$, there is a DFA $A'$ over the same alphabet $\Sigma$ such that $L(A') = \Sigma^* - L(A)$.

**Closure under intersection:** For every two DFA $A_1$ and $A_2$, there is a DFA $A'$ such that $L(A') = L(A_1) \cap L(A_2)$.

**Closure under union:** For every two DFA $A_1$ and $A_2$, there is a DFA $A'$ such that $L(A') = L(A_1) \cup L(A_2)$.

See Note 1 for the formal proof of Theorem 1.3.
1. Deterministic finite state automata

2. Non-deterministic finite state automata

3. Pumping lemma
Non-deterministic finite state automata (NFA)

(Def.) A *non-deterministic finite state automaton* (NFA) is a system $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$ where:

- $\Sigma$ is an alphabet.
- $Q$ is a finite set of states.
- $q_0 \in Q$ is the initial state.
- $F \subseteq Q$ is the set of accepting states.
- $\delta \subseteq Q \times \Sigma \times Q$ is the transition relation.

Note: In DFA, $\delta$ is a function $\delta : Q \times \Sigma \rightarrow Q$. In NFA, $\delta$ is any subset of $Q \times \Sigma \times Q$. 

24/45
Non-deterministic finite state automata (NFA)

(Def.) A non-deterministic finite state automaton (NFA) is a system $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$ where:

- $\Sigma$ is an alphabet.
- $Q$ is a finite set of states.
- $q_0 \in Q$ is the initial state.
- $F \subseteq Q$ is the set of accepting states.
- $\delta \subseteq Q \times \Sigma \times Q$ is the transition relation.

Note: In DFA, $\delta$ is a function $\delta: Q \times \Sigma \rightarrow Q$. In NFA, $\delta$ is any subset of $Q \times \Sigma \times Q$. 

Non-deterministic finite state automata (NFA)

(Def.) A *non-deterministic finite state automaton* (NFA) is a system $A = \langle \Sigma, Q, q_0, F, \delta \rangle$ where:

- $\Sigma$ is an alphabet.
- $Q$ is a finite set of states.
- $q_0 \in Q$ is the initial state.
- $F \subseteq Q$ is the set of accepting states.
- $\delta \subseteq Q \times \Sigma \times Q$ is the transition relation.

Note: In DFA, $\delta$ is a function $\delta: Q \times \Sigma \rightarrow Q$. In NFA, $\delta$ is any subset of $Q \times \Sigma \times Q$. 
Non-deterministic finite state automata (NFA)

(Def.) A non-deterministic finite state automaton (NFA) is a system \( A = \langle \Sigma, Q, q_0, F, \delta \rangle \) where:

- \( \Sigma \) is an alphabet.
- \( Q \) is a finite set of states.
- \( q_0 \in Q \) is the initial state.
- \( F \subseteq Q \) is the set of accepting states.
- \( \delta \subseteq Q \times \Sigma \times Q \) is the transition relation.

Note: In DFA, \( \delta \) is a function \( \delta : Q \times \Sigma \rightarrow Q \). In NFA, \( \delta \) is any subset of \( Q \times \Sigma \times Q \).
Non-deterministic finite state automata (NFA)

(Def.) A non-deterministic finite state automaton (NFA) is a system \( A = \langle \Sigma, Q, q_0, F, \delta \rangle \) where:

- \( \Sigma \) is an alphabet.
- \( Q \) is a finite set of states.
- \( q_0 \in Q \) is the initial state.
- \( F \subseteq Q \) is the set of accepting states.
Non-deterministic finite state automata (NFA)

(Def.) A non-deterministic finite state automaton (NFA) is a system \( A = \langle \Sigma, Q, q_0, F, \delta \rangle \) where:

- \( \Sigma \) is an alphabet.
- \( Q \) is a finite set of states.
- \( q_0 \in Q \) is the initial state.
- \( F \subseteq Q \) is the set of accepting states.
- \( \delta \subseteq Q \times \Sigma \times Q \) is the transition relation.
Non-deterministic finite state automata (NFA)

(Def.) A non-deterministic finite state automaton (NFA) is a system $A = \langle \Sigma, Q, q_0, F, \delta \rangle$ where:

- $\Sigma$ is an alphabet.
- $Q$ is a finite set of states.
- $q_0 \in Q$ is the initial state.
- $F \subseteq Q$ is the set of accepting states.
- $\delta \subseteq Q \times \Sigma \times Q$ is the transition relation.

Note: In DFA, $\delta$ is a function $\delta : Q \times \Sigma \rightarrow Q$.

In NFA, $\delta$ is any subset of $Q \times \Sigma \times Q$. 
Example 1

Consider the following $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$:

- $\Sigma = \{a, b\}$
- $Q = \{q, p, r\}$ is the set of states.
- $r$ is the initial state.
- $F = \emptyset$, i.e., it does not have any accepting state.
- The transition relation $\delta$ is $\{(p, a, q), (r, a, r)\}$. 
Example 1

Consider the following $A = (\Sigma, Q, q_0, F, \delta)$:

- $\Sigma = \{a, b\}$
- $Q = \{q, p, r\}$ is the set of states.
- $r$ is the initial state.
- $F = \emptyset$, i.e., it does not have any accepting state.
- The transition relation $\delta$ is $\{(p, a, q), (r, a, r)\}$.

This is a valid NFA.
Example 1

Consider the following $A = (\Sigma, Q, q_0, F, \delta)$:

- $\Sigma = \{a, b\}$
- $Q = \{q, p, r\}$ is the set of states.
- $r$ is the initial state.
- $F = \emptyset$, i.e., it does not have any accepting state.
- The transition relation $\delta$ is $\{(p, a, q), (r, a, r)\}$.

This is a valid NFA.

The transition relation $\delta \subseteq Q \times \Sigma \times Q$. 
Example 2

Consider the following $A = \langle \Sigma, Q, q_0, F, \delta \rangle$:

- $\Sigma = \{a, b\}$
- $Q = \{q, p, r\}$ is the set of states.
- $r$ is the initial state.
- $F = \emptyset$, i.e., it does not have any accepting state.
- The transition relation $\delta$ is $\{(p, a, q), (r, a, r), (p, a, p)\}$.
Consider the following $A = (\Sigma, Q, q_0, F, \delta)$:

- $\Sigma = \{a, b\}$
- $Q = \{q, p, r\}$ is the set of states.
- $r$ is the initial state.
- $F = \emptyset$, i.e., it does not have any accepting state.
- The transition relation $\delta$ is $\{(p, a, q), (r, a, r), (p, a, p)\}$.

This is a valid NFA.
Example 2

Consider the following $A = \langle \Sigma, Q, q_0, F, \delta \rangle$:

- $\Sigma = \{a, b\}$
- $Q = \{q, p, r\}$ is the set of states.
- $r$ is the initial state.
- $F = \emptyset$, i.e., it does not have any accepting state.
- The transition relation $\delta$ is $\{(p, a, q), (r, a, r), (p, a, p)\}$.

This is a valid NFA.

The transition relation $\delta \subseteq Q \times \Sigma \times Q$. 
Example 3

Consider the following $A = \langle \Sigma, Q, q_0, F, \delta \rangle$:

- $\Sigma = \{a, b\}$
- $Q = \{q, p, r\}$ is the set of states.
- $r$ is the initial state.
- $F = \emptyset$, i.e., it does not have any accepting state.
- The transition relation $\delta$ is $\emptyset$. 
Example 3

Consider the following $\mathcal{A} = (\Sigma, Q, q_0, F, \delta)$:

- $\Sigma = \{a, b\}$
- $Q = \{q, p, r\}$ is the set of states.
- $r$ is the initial state.
- $F = \emptyset$, i.e., it does not have any accepting state.
- The transition relation $\delta$ is $\emptyset$.

This is a valid NFA.
Example 3

Consider the following \( \mathcal{A} = (\Sigma, Q, q_0, F, \delta) \):

- \( \Sigma = \{a, b\} \)
- \( Q = \{q, p, r\} \) is the set of states.
- \( r \) is the initial state.
- \( F = \emptyset \), i.e., it does not have any accepting state.
- The transition relation \( \delta \) is \( \emptyset \).

This is a valid NFA.

The transition relation \( \delta \subseteq Q \times \Sigma \times Q \).
Example 4

Consider \( \mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle \):

- \( \Sigma = \{ a, b \} \)
- \( Q = \{ q, p, r \} \) is the set of states.
- \( r \) is the initial state.
- \( F = \{ p, q \} \) is the set of accepting states.
- The transition relation \( \delta \) is a function defined as:

\[
\begin{align*}
\delta(p, a) &= p & \delta(p, b) &= r \\
\delta(q, a) &= p & \delta(q, b) &= p \\
\delta(r, a) &= q & \delta(r, b) &= r
\end{align*}
\]
Consider $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$:

- $\Sigma = \{a, b\}$
- $Q = \{q, p, r\}$ is the set of states.
- $r$ is the initial state.
- $F = \{p, q\}$ is the set of accepting states.
- The transition relation $\delta$ is a function defined as:
  
  $\delta(p, a) = p$  \hspace{1cm}  $\delta(p, b) = r$
  
  $\delta(q, a) = p$  \hspace{1cm}  $\delta(q, b) = p$
  
  $\delta(r, a) = q$  \hspace{1cm}  $\delta(r, b) = r$

This is also a valid NFA.
Example 4

Consider $A = \langle \Sigma, Q, q_0, F, \delta \rangle$:

- $\Sigma = \{a, b\}$
- $Q = \{q, p, r\}$ is the set of states.
- $r$ is the initial state.
- $F = \{p, q\}$ is the set of accepting states.
- The transition relation $\delta$ is a function defined as:
  \[ \delta(p, a) = p \quad \delta(p, b) = r \]
  \[ \delta(q, a) = p \quad \delta(q, b) = p \]
  \[ \delta(r, a) = q \quad \delta(r, b) = r \]

This is also a valid NFA.

A DFA is a special case of NFA, because function is a special case of relation. (See Note 0.)
Consider an DFA $A = \langle \Sigma, Q, q_0, F, \delta \rangle$ over $\Sigma = \{a, b\}$, where $Q = \{q, p, r\}$, $r$ is the initial state, $F = \{p\}$ and $\delta$ is as follows.

$$\delta = \{(p, a, p), (p, a, q), (p, b, q), (q, b, r), (r, a, q), (r, b, r)\}$$
Consider an DFA $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$ over $\Sigma = \{a, b\}$, where $Q = \{q, p, r\}$, $r$ is the initial state, $F = \{p\}$ and $\delta$ is as follows.

$$\delta = \{(p, a, p), (p, a, q), (p, b, q), (q, b, r), (r, a, q), (r, b, r)\}$$

We can visualize it as a directed graph:
Consider an DFA $A = \langle \Sigma, Q, q_0, F, \delta \rangle$ over $\Sigma = \{a, b\}$, where $Q = \{q, p, r\}$, $r$ is the initial state, $F = \{p\}$ and $\delta$ is as follows.

$$\delta = \{(p, a, p), (p, a, q), (p, b, q), (q, b, r), (r, a, q), (r, b, r)\}$$

We can visualize it as a directed graph:
Consider an DFA $A = \langle \Sigma, Q, q_0, F, \delta \rangle$ over $\Sigma = \{a, b\}$, where $Q = \{q, p, r\}$, $r$ is the initial state, $F = \{p\}$ and $\delta$ is as follows.

$$\delta = \{(p, a, p), (p, a, q), (p, b, q), (q, b, r), (r, a, q), (r, b, r)\}$$

We can visualize it as a directed graph:
Let $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$ be an NFA.

(Def.) On input word $w = a_1 \cdots a_n$, a run of $\mathcal{A}$ on $w$ is the sequence:

$$p_0 a_1 p_1 a_2 p_2 \cdots a_n p_n,$$

where $p_0 = q_0$ and $(p_i, a_{i+1}, p_{i+1}) \in \delta$, for each $i = 0, \ldots, n - 1$.
Acceptance/rejection of words by NFA

Let $A = \langle \Sigma, Q, q_0, F, \delta \rangle$ be an NFA.

(Def.) On input word $w = a_1 \cdots a_n$, a run of $A$ on $w$ is the sequence:

$$p_0, a_1, p_1, a_2, p_2, \cdots, a_n, p_n,$$

where $p_0 = q_0$ and $(p_i, a_{i+1}, p_{i+1}) \in \delta$, for each $i = 0, \ldots, n - 1$.

(Def.) A run of $A$ on $w$ starting from state $q$ is defined as the sequence above, but with condition $p_0 = q$. 
Acceptance/rejection of words by NFA

Let \( \mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle \) be an NFA.

**Def.** On input word \( w = a_1 \cdots a_n \), a run of \( \mathcal{A} \) on \( w \) is the sequence:

\[
p_0 \ a_1 \ p_1 \ a_2 \ p_2 \ \cdots \ a_n \ p_n,
\]

where \( p_0 = q_0 \) and \( (p_i, a_{i+1}, p_{i+1}) \in \delta \), for each \( i = 0, \ldots, n - 1 \).

**Def.** A run of \( \mathcal{A} \) on \( w \) starting from state \( q \) is defined as the sequence above, but with condition \( p_0 = q \).

**Def.** A run is called an accepting run, if \( p_0 = q_0 \) and \( q_n \in F \).
The language accepted by NFA

Let $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$.

(Def.) We say that $\mathcal{A}$ accepts $w$, if there is an accepting run of $\mathcal{A}$ on $w$. 
The language accepted by NFA

Let $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$.

(Def.) We say that $\mathcal{A}$ accepts $w$, if there is an accepting run of $\mathcal{A}$ on $w$.

(Def.) The language of all words accepted by $\mathcal{A}$ is denoted by $L(\mathcal{A})$. 
The language accepted by NFA

Let \( \mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle \).

(Def.) We say that \( \mathcal{A} \) accepts \( w \), if there is an accepting run of \( \mathcal{A} \) on \( w \).

(Def.) The language of all words accepted by \( \mathcal{A} \) is denoted by \( L(\mathcal{A}) \).

(Def.) A language \( L \) is called an NFA language, if there is a NFA \( \mathcal{A} \) such that \( L(\mathcal{A}) = L \).
Example 5

On input string 10110, there are many possible runs:

- $p_1 p_0 p_1 p_1$ (not an accepting run).
- $p_1 p_0 p_1 q_1 r_0 r$. (an accepting run).
- ...

There is an accepting run so $A$ accepts 10110.
On input string 10110, there are many possible runs:

- $p_1 p_0 p_1 p_1$. (not an accepting run).
- $p_1 p_0 p_1 q_1 r_0 r$. (an accepting run).
- (there are many other runs)

There is an accepting run so $A$ accepts 10110.
Example 5

On input string 10110, there are many possible runs:

- \(p \ 1 \ p \ 0 \ p \ 1 \ p \ 1 \ p \ 0 \ p\). (not an accepting run).

There is an accepting run so \(A\) accepts 10110.
Example 5

On input string 10110, there are many possible runs:

- $p \ 1 \ p \ 0 \ p \ 1 \ p \ 1 \ p \ 0 \ p$. (not an accepting run).
- $p \ 1 \ p \ 0 \ p \ 1 \ p \ 1 \ q$. (stuck in $q$, not an accepting run).
- $p \ 1 \ p \ 0 \ p \ 1 \ p \ 1 \ q$. (an accepting run).
Example 5

On input string 10110, there are many possible runs:

- $p \, 1 \, p \, 0 \, p \, 1 \, p \, 1 \, p \, 0 \, p$. (not an accepting run).
- $p \, 1 \, p \, 0 \, p \, 1 \, p \, 1 \, q$. (stuck in $q$, not an accepting run).
- $p \, 1 \, p \, 0 \, p \, 1 \, q \, 1 \, r \, 0 \, r$. (an accepting run).
On input string 10110, there are many possible runs:

- $p\ 1\ p\ 0\ p\ 1\ p\ 1\ p\ 0\ p$. (not an accepting run).
- $p\ 1\ p\ 0\ p\ 1\ p\ 1\ q$. (stuck in $q$, not an accepting run).
- $p\ 1\ p\ 0\ p\ 1\ q\ 1\ r\ 0\ r$. (an accepting run).
- ... (there are many other runs)
Example 5

On input string 10110, there are many possible runs:

- \( p \ 1 \ p \ 0 \ p \ 1 \ p \ 1 \ p \ 1 \ p \ 0 \ p \). (not an accepting run).
- \( p \ 1 \ p \ 0 \ p \ 1 \ p \ 1 \ q \). (stuck in \( q \), not an accepting run).
- \( p \ 1 \ p \ 0 \ p \ 1 \ q \ 1 \ r \ 0 \ r \). (an accepting run).
- \( \ldots \) (there are many other runs)

There is an accepting run so \( A \) accepts 10110.
Example 5

On Input word: 10110
Example 5

On Input word: 10110

$p$
Example 5

On Input word: 10110
Example 5

On Input word: 10110

Diagram of the input and state transitions.
On Input word: 10110
Example 5

On Input word: 10110
Example 5

On Input word: 10110

On Input word: 10110
Example 5

On Input word: 10110
Example 5

On Input word: 10110

```
0, 1  
\rightarrow p  
\downarrow 1  
\rightarrow q  
\downarrow 1  
\rightarrow r  
0, 1
```

```
\begin{array}{c}
p \\
\downarrow 1 \\
p, q \\
\downarrow 0 \\
p \\
\downarrow 1 \\
p, q, r \\
\downarrow 0 \\
p, q, r \\
\end{array}
```
Example 5

On Input word: 10110
Example 5

On Input word: 10110
Example 5

On Input word: 10110
Example 5

On Input word: 10110
Example 5

On Input word: 10110 (accepted)
(Remark 1.4) NFA languages are closed under intersection and union.
Closure under union and intersection

(Remark 1.4) NFA languages are closed under intersection and union.

More formally, it can be stated as follows.

- For every two NFA $A_1$ and $A_2$, there is an NFA $A'$ such that $L(A') = L(A_1) \cap L(A_2)$.

- For every two NFA $A_1$ and $A_2$, there is an NFA $A'$ such that $L(A') = L(A_1) \cup L(A_2)$.
Closure under union and intersection

(Remark 1.4) NFA languages are closed under intersection and union.

More formally, it can be stated as follows.

- For every two NFA $A_1$ and $A_2$, there is an NFA $A'$ such that $L(A') = L(A_1) \cap L(A_2)$.

- For every two NFA $A_1$ and $A_2$, there is an NFA $A'$ such that $L(A') = L(A_1) \cup L(A_2)$.

The proof is the same as the one for DFA.
Theorem 1.5
For every NFA $\mathcal{A}$, there is a DFA $\mathcal{A}'$ such that $L(\mathcal{A}) = L(\mathcal{A}')$.
NFA can be converted to DFA

Theorem 1.5
For every NFA \( A \), there is a DFA \( A' \) such that \( L(A) = L(A') \).

*Proof* Let \( A = \langle \Sigma, Q, q_0, F, \delta \rangle \) be an NFA.
NFA can be converted to DFA

**Theorem 1.5**
For every NFA $A$, there is a DFA $A'$ such that $L(A) = L(A')$.

**(Proof)** Let $A = \langle \Sigma, Q, q_0, F, \delta \rangle$ be an NFA.

Consider the following DFA $A' = \langle \Sigma, Q', q'_0, F', \delta' \rangle$. 
Theorem 1.5
For every NFA $A$, there is a DFA $A'$ such that $L(A) = L(A')$. 

(Proof) Let $A = \langle \Sigma, Q, q_0, F, \delta \rangle$ be an NFA.

Consider the following DFA $A' = \langle \Sigma, Q', q'_0, F', \delta' \rangle$.

- $Q' = 2^Q$, i.e., the set of all subsets of $Q$, including $\emptyset$ and $Q$. 

Theorem 1.5

For every NFA $A$, there is a DFA $A'$ such that $L(A) = L(A')$.

(Proof) Let $A = \langle \Sigma, Q, q_0, F, \delta \rangle$ be an NFA.

Consider the following DFA $A' = \langle \Sigma, Q', q'_0, F', \delta' \rangle$.

- $Q' = 2^Q$, i.e., the set of all subsets of $Q$, including $\emptyset$ and $Q$.
- The initial state is $\{q_0\}$.
Theorem 1.5
For every NFA \( A \), there is a DFA \( A' \) such that \( L(A) = L(A') \).

(Proof) Let \( A = \langle \Sigma, Q, q_0, F, \delta \rangle \) be an NFA.

Consider the following DFA \( A' = \langle \Sigma, Q', q_0', F', \delta' \rangle \).

- \( Q' = 2^Q \), i.e., the set of all subsets of \( Q \), including \( \emptyset \) and \( Q \).
- The initial state is \( \{q_0\} \).
- \( F' \) consists of the subset \( S \subseteq Q \) where \( S \cap F \neq \emptyset \).
Theorem 1.5

For every NFA $A$, there is a DFA $A'$ such that $L(A) = L(A')$.

(Proof) Let $A = \langle \Sigma, Q, q_0, F, \delta \rangle$ be an NFA.

Consider the following DFA $A' = \langle \Sigma, Q', q'_0, F', \delta' \rangle$.

- $Q' = 2^Q$, i.e., the set of all subsets of $Q$, including $\emptyset$ and $Q$.
- The initial state is $\{q_0\}$.
- $F'$ consists of the subset $S \subseteq Q$ where $S \cap F \neq \emptyset$.
- The transition function $\delta : 2^Q \times \Sigma \rightarrow 2^Q$ is defined as follows.

$$
\delta'(S, a) = \{p \mid \text{there is } q \in S \text{ such that } (q, a, p) \in \delta\}
$$

It can be shown that $L(A') = L(A)$. See Note 1 for more details.
Theorem 1.5
For every NFA $A$, there is a DFA $A'$ such that $L(A) = L(A')$.

(Proof) Let $A = \langle \Sigma, Q, q_0, F, \delta \rangle$ be an NFA.

Consider the following DFA $A' = \langle \Sigma, Q', q'_0, F', \delta' \rangle$.

- $Q' = 2^Q$, i.e., the set of all subsets of $Q$, including $\emptyset$ and $Q$.
- The initial state is $\{q_0\}$.
- $F'$ consists of the subset $S \subseteq Q$ where $S \cap F \neq \emptyset$.
- The transition function $\delta : 2^Q \times \Sigma \rightarrow 2^Q$ is defined as follows.

$$\delta'(S, a) = \{ p \mid \text{there is } q \in S \text{ such that } (q, a, p) \in \delta \}$$

It can be shown that $L(A') = L(A)$. See Note 1 for more details.
The intuitive idea

On input 10110:

On input $w$, the set of states it can get to is a subset of \{p, q, r\}

The DFA is:

\[
\begin{array}{c}
p \\
\downarrow 1 \rightarrow q \\
\downarrow 1 \rightarrow r \\
\end{array}
\]

0, 1

∅
The intuitive idea

On input 10110:

On input \textit{w}, the set of states it can get to is a subset of \{p, q, r\}

The DFA is:
The intuitive idea

On input 10110:

On input $w$, the set of states it can get to is a subset of \{p, q, r\}
The intuitive idea

The DFA is:
The intuitive idea

On input 10110:

$p, q, r$

The DFA is:

$p, q, r$

∅
The intuitive idea

On input 10110:

The DFA is:
The intuitive idea

On input 10110:

The DFA is:
The intuitive idea

On input 10110:

The DFA is:
The intuitive idea

On input 10110:
- $p_0, q_1, r_1, q_0, r_1$

The DFA is:

- $\emptyset$
- $q$
- $r$
- $q, r$
The intuitive idea

On input 10110:

The DFA is:
The intuitive idea

On input 10110:

The DFA is:
The intuitive idea

On input 10110:

The DFA is:
The intuitive idea

On input 10110:

The DFA is:
The intuitive idea

On input 10110:

The DFA is:
The intuitive idea

On input 10110:

The DFA is:
The intuitive idea

On input 10110:

On input \(w\), the set of states it can get to is a subset of \(\{p, q, r\}\)

The DFA is:
The intuitive idea

On input 10110:

The DFA is:
The intuitive idea

On input 10110:
- p
- q
- r

The set of states it can get to is a subset of {p, q, r}

The DFA is:

The states and transitions are as follows:
- p
- q
- r
- ∅
The intuitive idea

On input 10110:

The DFA is:

The set of states it can get to is a subset of \{p, q, r\}.
Theorem 1.5

For every NFA \( A \), there is a DFA \( A' \) such that \( L(A) = L(A') \).
Theorem 1.5

For every NFA $A$, there is a DFA $A'$ such that $L(A) = L(A')$.

From this theorem, we can say that a language is regular if and only if it is accepted by an NFA.
NFA and DFA

**Theorem 1.5**
For every NFA $\mathcal{A}$, there is a DFA $\mathcal{A}'$ such that $L(\mathcal{A}) = L(\mathcal{A}')$.

From this theorem, we can say that a language is regular if and only if it is accepted by an NFA.

**Corollary 1.6**
NFA languages are closed under complement.
NFA and DFA

**Theorem 1.5**
*For every NFA \( A \), there is a DFA \( A' \) such that \( L(A) = L(A') \).*

From this theorem, we can say that a language is regular if and only if it is accepted by an NFA.

**Corollary 1.6**
*NFA languages are closed under complement.*

More precisely, we can say that for every NFA \( A \) over alphabet \( \Sigma \), there is a DFA \( A' \) over the same alphabet \( \Sigma \) such that \( L(A') = \Sigma^* - L(A) \).
Concatenation and Kleene star

(Def.) For two words $u$ and $v$, $u \cdot v$ denotes the word obtained by concatenating $v$ at the end of $u$.

($u \cdot v$ reads: $u$ concatenates with $v$.)
Concatenation and Kleene star

(Def.) For two words $u$ and $v$, $u \cdot v$ denotes the word obtained by concatenating $v$ at the end of $u$.

($u \cdot v$ reads: $u$ concatenates with $v$.)

For languages $L_1$, $L_2$ and $L$:

\[
L_1 \cdot L_2 := \{ uv \mid u \in L_1 \text{ and } v \in L_2 \} \quad \text{(Concatenation)}
\]

\[
L^n := \{ u_1 \cdots u_n \mid \text{each } u_i \in L \}
\]

\[
L^* := \bigcup_{n \geq 0} L^n \quad \text{(Kleene star)}
\]
Concatenation and Kleene star

(Def.) For two words \( u \) and \( v \), \( u \cdot v \) denotes the word obtained by concatenating \( v \) at the end of \( u \).

(\( u \cdot v \) reads: \( u \) concatenates with \( v \).)

For languages \( L_1, L_2 \) and \( L \):

\[
L_1 \cdot L_2 := \{ uv \mid u \in L_1 \text{ and } v \in L_2 \}
\]

(Concatenation)

\[
L^n := \{ u_1 \cdots u_n \mid \text{each } u_i \in L \}
\]

\[
L^* := \bigcup_{n \geq 0} L^n
\]

(Kleene star)

As before, we usually write \( L_1 L_2 \) to denote \( L_1 \cdot L_2 \).
(Def.) For two words $u$ and $v$, $u \cdot v$ denotes the word obtained by \textit{concatenating} $v$ at the end of $u$.

($u \cdot v$ reads: $u$ concatenates with $v$.)

For languages $L_1$, $L_2$ and $L$:

$$L_1 \cdot L_2 := \{ uv \mid u \in L_1 \text{ and } v \in L_2 \}$$

(Concatenation)

$$L^n := \{ u_1 \cdots u_n \mid \text{each } u_i \in L \}$$

(Kleene star)

$$L^* := \bigcup_{n \geq 0} L^n$$

As before, we usually write $L_1 L_2$ to denote $L_1 \cdot L_2$.

$L_1 L_2$ reads as $L_1$ concatenates with $L_2$. 

By default, for any set $X \subseteq \Sigma^*$, $X^0 = \{ \epsilon \}$.

Thus, $\emptyset^* = \{ \epsilon \}$. 

38/45
(Def.) For two words $u$ and $v$, $u \cdot v$ denotes the word obtained by \textit{concatenating} $v$ at the end of $u$.

($u \cdot v$ reads: $u$ concatenates with $v$.)

For languages $L_1, L_2$ and $L$:

$$L_1 \cdot L_2 := \{uv \mid u \in L_1 \text{ and } v \in L_2\} \quad \text{(Concatenation)}$$

$$L^n := \{u_1 \cdots u_n \mid \text{each } u_i \in L\}$$

$$L^* := \bigcup_{n \geq 0} L^n \quad \text{(Kleene star)}$$

As before, we usually write $L_1L_2$ to denote $L_1 \cdot L_2$.

$L_1L_2$ reads as $L_1$ concatenates with $L_2$.

By default, for any set $X \subseteq \Sigma^*$, $X^0 = \{\epsilon\}$.

Thus, $\emptyset^* = \{\epsilon\}$. 

Closure under concatenation and Kleene star

**Theorem 1.8**

Regular languages (NFA languages) are closed under concatenation and Kleene star.
Closure under concatenation and Kleene star

**Theorem 1.8**

Regular languages (NFA languages) are closed under concatenation and Kleene star.

More formally, it can be stated as follows.

- If $L_1$ and $L_2$ are regular languages, so is $L_1L_2$.
- If $L$ is a regular language, so is $L^*$.
Closure under concatenation and Kleene star

**Theorem 1.8**

Regular languages (NFA languages) are closed under concatenation and Kleene star.

More formally, it can be stated as follows.

- If $L_1$ and $L_2$ are regular languages, so is $L_1L_2$.
- If $L$ is a regular language, so is $L^*$.

The proof can be found in Note 1.
Table of contents

1. Deterministic finite state automata

2. Non-deterministic finite state automata

3. Pumping lemma
Pumping lemma – A tool for showing non-regularity of a language

(Def.) For a word $w$ and an integer $n \geq 0$, $w^n$ is a word where $w$ is repeated $n$ number of times, i.e.,

$w \cdots w$

\[ n \text{ times} \]

By default, we define $w^0 = \varepsilon$. 


Pumping lemma – A tool for showing non-regularity of a language

(Def.) For a word \( w \) and an integer \( n \geq 0 \), \( w^n \) is a word where \( w \) is repeated \( n \) number of times, i.e.,

\[
\underbrace{w \cdots w}_n \text{ times}
\]

By default, we define \( w^0 = \varepsilon \).

**Lemma 1.9 (pumping lemma)**

Let \( A = \langle \Sigma, Q, q_0, F, \delta \rangle \) be an NFA. Let \( x \in L(A) \) be a word such that \( |x| \geq |Q| \). Then, the word \( x \) can be divided into three parts \( u, v, w \), i.e., \( x = uvw \), such that \( |v| \geq 1 \) and for every integer \( k \geq 0 \), \( uv^kw \in L(A) \).
Proof of pumping lemma

Let \( x = a_1 \cdots a_n \) and \( x \in L(A) \), where \( n \geq |Q| \).
Proof of pumping lemma

Let $x = a_1 \cdots a_n$ and $x \in L(A)$, where $n \geq |Q|$.

Let the following be its accepting run:

$$p_0 a_1 p_1 a_2 p_2 \cdots a_n p_n$$
Proof of pumping lemma

Let \( x = a_1 \cdots a_n \) and \( x \in L(A) \), where \( n \geq |Q| \).

Let the following be its accepting run:

\[
p_0 a_1 p_1 a_2 p_2 \cdots a_n p_n
\]

Since \( n \geq |Q| \), there are \( 0 \leq i < j \leq n \) such that \( p_i = p_j \).
Proof of pumping lemma

Let $x = a_1 \cdots a_n$ and $x \in L(A)$, where $n \geq |Q|$.

Let the following be its accepting run:

$$p_0 \ a_1 \ p_1 \ a_2 \ p_2 \ \cdots \ a_n \ p_n$$

Since $n \geq |Q|$, there are $0 \leq i < j \leq n$ such that $p_i = p_j$.

Let $u = a_1 \cdots a_i$, $v = a_{i+1} \cdots a_j$ and $w = a_{j+1} \cdots a_n$. 
Proof of pumping lemma

Let $x = a_1 \cdots a_n$ and $x \in L(A)$, where $n \geq |Q|$.

Let the following be its accepting run:

$$p_0 a_1 p_1 a_2 p_2 \cdots a_n p_n$$

Since $n \geq |Q|$, there are $0 \leq i < j \leq n$ such that $p_i = p_j$.

Let $u = a_1 \cdots a_i$, $v = a_{i+1} \cdots a_j$ and $w = a_{j+1} \cdots a_n$.

Then, for every integer $k \geq 0$, the following is an accepting run of $A$ on $uv^k w$:

$$p_0 a_1 p_1 a_2 p_2 \cdots a_i p_i a_{i+1} p_{i+1} \cdots a_j p_j a_{j+1} p_{j+1} \cdots a_n p_n$$

repeat $k$ times
Variations of pumping lemma

Lemma 1.11 (more refined pumping lemma)
Let \( A = \langle \Sigma, Q, q_0, F, \delta \rangle \) be an NFA. Let \( x \in L(A) \) be a word and \( x = szt \), where \( |z| \geq |Q| \). Then, the word \( z \) can be divided into three parts \( u, v, w \) such that \( |v| \geq 1 \) and for every positive integer \( k \geq 0 \), \( suv^k w t \in L(A) \).
Variations of pumping lemma

**Lemma 1.11 (more refined pumping lemma)**

Let $A = \langle \Sigma, Q, q_0, F, \delta \rangle$ be an NFA. Let $x \in L(A)$ be a word and $x = szt$, where $|z| \geq |Q|$. Then, the word $z$ can be divided into three parts $u, v, w$ such that $|v| \geq 1$ and for every positive integer $k \geq 0$, $sv^kw \in L(A)$.

Pumping lemma can also be stated more elegantly as follows.

**Lemma 1.10 (pumping lemma)**

For every regular language $L$, there is an integer $n \geq 1$ such that for every word $x \in L$ with length $|x| \geq n$, there are $u, v, w$ where $x = uvw$ and $|v| \geq 1$ and for every integer $k \geq 0$, $uv^kw \in L$. 
Using pumping lemma to prove non-regularity

We would like to show that $L_1 = \{a^k b^k \mid k \geq 0\}$ is not regular.
Using pumping lemma to prove non-regularity

We would like to show that $L_1 = \{a^k b^k \mid k \geq 0\}$ is not regular.

In other words, there is no NFA that accepts $L_1$. 
We would like to show that $L_1 = \{a^k b^k \mid k \geq 0\}$ is not regular.

In other words, there is no NFA that accepts $L_1$.

Suppose there is an NFA $\mathcal{A}$ that accepts $L_1$ where $Q$ is the set of states.
Using pumping lemma to prove non-regularity

We would like to show that $L_1 = \{a^k b^k \mid k \geq 0\}$ is not regular.

In other words, there is no NFA that accepts $L_1$.

Suppose there is an NFA $A$ that accepts $L_1$ where $Q$ is the set of states.
Consider the following word: $a^k b^k$ where $k \geq |Q|$.
Using pumping lemma to prove non-regularity

We would like to show that $L_1 = \{a^k b^k \mid k \geq 0\}$ is not regular.

In other words, there is no NFA that accepts $L_1$.

Suppose there is an NFA $A$ that accepts $L_1$ where $Q$ is the set of states.

Consider the following word: $a^k b^k$ where $k \geq |Q|$.

By (more refined) pumping lemma, we can divide $a^k$ into three parts $u, v, w$ such that:

$$u \cdot v^\ell \cdot w \cdot b^k \in L(A) \text{ for every } \ell \geq 0$$

all are $a$’s here
Using pumping lemma to prove non-regularity

We would like to show that \( L_1 = \{a^k b^k \mid k \geq 0\} \) is not regular.

In other words, there is no NFA that accepts \( L_1 \).

Suppose there is an NFA \( \mathcal{A} \) that accepts \( L_1 \) where \( Q \) is the set of states.

Consider the following word: \( a^k b^k \) where \( k \geq |Q| \).

By (more refined) pumping lemma, we can divide \( a^k \) into three parts \( u, v, w \) such that:

\[
\underbrace{u \ v^\ell \ w}_{\text{all are } a's \ here} \ b^k \quad \in \ L(\mathcal{A}) \quad \text{for every } \ell \geq 0
\]

This means that the number of \( a \)'s becomes different from the number of \( b \)'s, which contradicts the assumption that \( \mathcal{A} \) accepts \( L_1 \).
Using pumping lemma to prove non-regularity

We would like to show that \( L_1 = \{a^k b^k \mid k \geq 0\} \) is not regular.

In other words, there is no NFA that accepts \( L_1 \).

Suppose there is an NFA \( \mathcal{A} \) that accepts \( L_1 \) where \( Q \) is the set of states.

Consider the following word: \( a^k b^k \) where \( k \geq |Q| \).

By (more refined) pumping lemma, we can divide \( a^k \) into three parts \( u, v, w \) such that:

\[
\underbrace{u \ v^\ell \ w}_{\text{all are } a's \ here} \ b^k \in L(\mathcal{A}) \quad \text{for every } \ell \geq 0
\]

This means that the number of \( a's \) becomes different from the number of \( b's \), which contradicts the assumption that \( \mathcal{A} \) accepts \( L_1 \).

Therefore, there is no NFA that accepts \( L_1 \) and \( L_1 \) is not regular.
Using pumping lemma to prove non-regularity

We would like to show that $L_2 = \{w \mid |w| \text{ is a prime number}\}$ is not regular, i.e., there is no NFA that accepts $L_2$. 
Using pumping lemma to prove non-regularity

We would like to show that \( L_2 = \{ w \mid |w| \text{ is a prime number} \} \) is not regular, i.e., there is no NFA that accepts \( L_2 \).

Suppose there is an NFA \( \mathcal{A} \) that accepts \( L_2 \) where \( Q \) is the set of states.
Using pumping lemma to prove non-regularity

We would like to show that $L_2 = \{w \mid |w| \text{ is a prime number}\}$ is not regular, i.e., there is no NFA that accepts $L_2$.

Suppose there is an NFA $A$ that accepts $L_2$ where $Q$ is the set of states. Consider the following word: $a^k$ where $k \geq |Q|$.
Using pumping lemma to prove non-regularity

We would like to show that $L_2 = \{ w \mid |w| \text{ is a prime number} \}$ is not regular, i.e., there is no NFA that accepts $L_2$.

Suppose there is an NFA $A$ that accepts $L_2$ where $Q$ is the set of states.
Consider the following word: $a^k$ where $k \geq |Q|$.
By pumping lemma, we can divide $a^k$ into three parts $u, v, w$ such that:

$$u^\ell v^\ell w \in L(A) \quad \text{for every } \ell \geq 0$$
Using pumping lemma to prove non-regularity

We would like to show that \( L_2 = \{ w \mid |w| \text{ is a prime number} \} \) is not regular, i.e., there is no NFA that accepts \( L_2 \).

Suppose there is an NFA \( \mathcal{A} \) that accepts \( L_2 \) where \( Q \) is the set of states. Consider the following word: \( a^k \) where \( k \geq |Q| \).

By pumping lemma, we can divide \( a^k \) into three parts \( u, v, w \) such that:

\[
\begin{align*}
u^\ell w \in L(\mathcal{A}) & \quad \text{for every } \ell \geq 0 \\
|u^\ell w| &= |u| + \ell|v| + |w|.
\end{align*}
\]

So this contradicts the assumption that \( \mathcal{A} \) accepts \( L_2 \).

Therefore, there is no NFA that accepts \( L_1 \), i.e., \( L_1 \) is not regular.
Using pumping lemma to prove non-regularity

We would like to show that $L_2 = \{ w \mid |w| \text{ is a prime number} \}$ is not regular, i.e., there is no NFA that accepts $L_2$.

Suppose there is an NFA $A$ that accepts $L_2$ where $Q$ is the set of states. Consider the following word: $a^k$ where $k \geq |Q|$.

By pumping lemma, we can divide $a^k$ into three parts $u, v, w$ such that:

$$u \cdot v^\ell \cdot w \in L(A) \quad \text{for every } \ell \geq 0$$

The length $|u \cdot v^\ell \cdot w| = |u| + \ell|v| + |w|$.

If we put $\ell = |u| + |w|$, we have:

$$|u \cdot v^\ell \cdot w| = (|u| + |w|)(|v| + 1) \quad \text{which is not prime}$$
Using pumping lemma to prove non-regularity

We would like to show that \( L_2 = \{ w \mid |w| \text{ is a prime number} \} \) is not regular, i.e., there is no NFA that accepts \( L_2 \).

Suppose there is an NFA \( \mathcal{A} \) that accepts \( L_2 \) where \( Q \) is the set of states.

Consider the following word: \( a^k \) where \( k \geq |Q| \).

By pumping lemma, we can divide \( a^k \) into three parts \( u, v, w \) such that:

\[
\text{for every } \ell \geq 0 \quad u \ v^\ell \ w \in L(\mathcal{A})
\]

The length \( |u \ v^\ell \ w| = |u| + \ell |v| + |w| \).

If we put \( \ell = |u| + |w| \), we have:

\[
|u \ v^\ell \ w| = (|u| + |w|)(|v| + 1) \quad \text{which is not prime}
\]

So this contradicts the assumption that \( \mathcal{A} \) accepts \( L_2 \).
Using pumping lemma to prove non-regularity

We would like to show that \( L_2 = \{ w \mid |w| \text{ is a prime number} \} \) is not regular, i.e., there is no NFA that accepts \( L_2 \).

Suppose there is an NFA \( \mathcal{A} \) that accepts \( L_2 \) where \( Q \) is the set of states. Consider the following word: \( a^k \) where \( k \geq |Q| \).

By pumping lemma, we can divide \( a^k \) into three parts \( u, v, w \) such that:

\[
  u \ v^\ell \ w \in L(\mathcal{A}) \quad \text{for every } \ell \geq 0
\]

The length \( |u \ v^\ell \ w| = |u| + \ell |v| + |w| \).

If we put \( \ell = |u| + |w| \), we have:

\[
  |u \ v^\ell \ w| = (|u| + |w|)(|v| + 1) \quad \text{which is not prime}
\]

So this contradicts the assumption that \( \mathcal{A} \) accepts \( L_2 \).

Therefore, there is no NFA that accepts \( L_1 \), i.e., \( L_1 \) is not regular.
End of Lesson 1