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

Table of contents

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.

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.

- Σ is an alphabet.

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.

- Σ 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 $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$, where each component is as follows.

- Σ is an alphabet.
- Q is a *finite* (non-empty) set of states.
- $q_0 \in Q$ is the initial state.

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.

- Σ 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.

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.

- Σ 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.

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.

- Σ 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 "*$\mathcal{A}$ is a DFA over alphabet Σ*," or that "*the alphabet of $\mathcal{A}$ is Σ*."

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.
- $F = \{p, q\}$ is the set of *accepting* states.
- The transition function δ is defined as:

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

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.
- $F = \{p, q\}$ is the set of *accepting* states.
- The transition function δ is defined as:

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

This is a valid DFA.

Example 2

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.
- $\underline{F = \emptyset}$, i.e., it does not have any accepting state.
- The transition function δ is defined as:

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

Example 2

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.
- $\underline{F = \emptyset}$, i.e., it does not have any accepting state.
- The transition function δ is defined as:

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

This is also a valid DFA.

Example 3

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

- $\underline{\Sigma = \{a\}}$.
- $Q = \{q, p, r\}$ is the set of states.
- r is the initial state.
- $F = \emptyset$, i.e., it does not have any accepting state.
- The transition function δ is defined as:

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

Example 3

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

- $\underline{\Sigma = \{a\}}$.
- $Q = \{q, p, r\}$ is the set of states.
- r is the initial state.
- $F = \emptyset$, i.e., it does not have any accepting state.
- The transition function δ is defined as:

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

This is also a valid DFA.

Example 4

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

- $\underline{\Sigma = \emptyset}$, 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 δ is not defined since $\Sigma = \emptyset$.

Example 4

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

- $\underline{\Sigma = \emptyset}$, 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 δ is not defined since $\Sigma = \emptyset$.

This is not a valid DFA, since the alphabet Σ must contain at least one symbol.

Example 5

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

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

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

Example 5

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

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

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

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

Example 6

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

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

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

Example 6

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

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

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

This is not a valid DFA, since δ is not defined on $(r, 1)$.

Example 7

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

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

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

Example 7

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

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

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

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

Example 8

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

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

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

Example 8

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

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

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

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 δ is defined as:

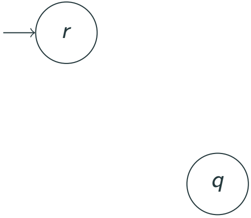
$$\delta(p, a) = p \quad \delta(p, b) = r \quad \delta(q, a) = p \quad \delta(q, b) = p \quad \delta(r, a) = q \quad \delta(r, b) = r$$

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 δ 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:

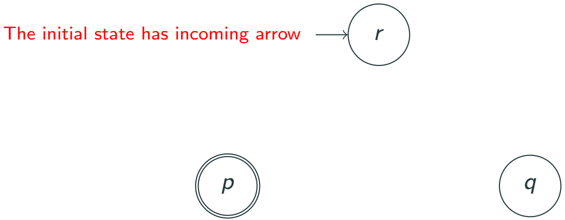


Visualizing DFA

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 δ 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:

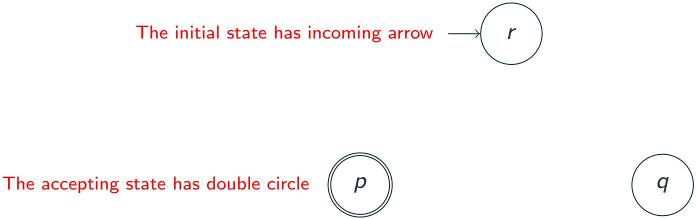


Visualizing DFA

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 δ 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:

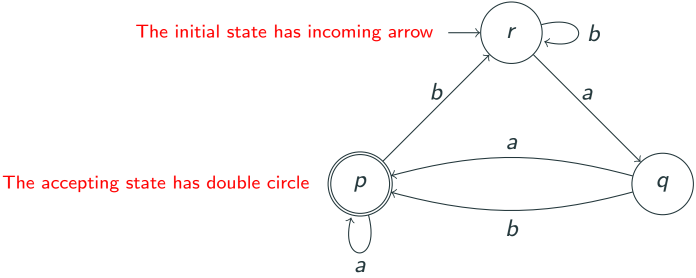


Visualizing DFA

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 δ 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:



Important note!

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

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

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

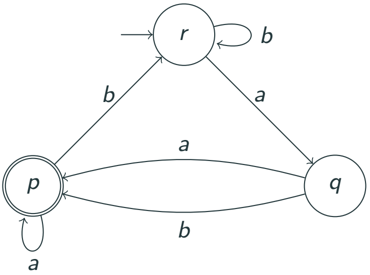
Important note!

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

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

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

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.

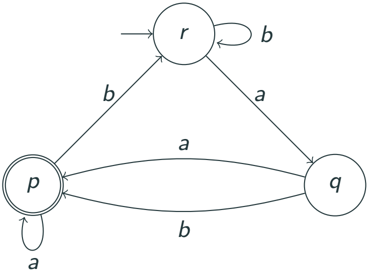
It moves from state to state depending on the input symbol that it reads.

It starts from the initial state.

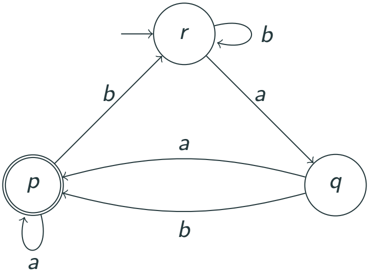
A DFA either accepts/rejects its input.

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

Example

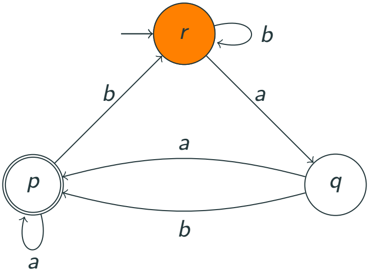


Example



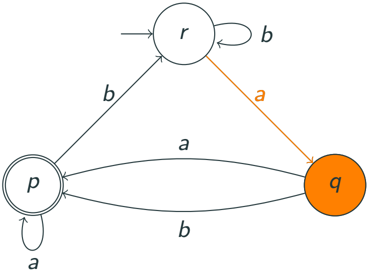
On input string *aba*:

Example



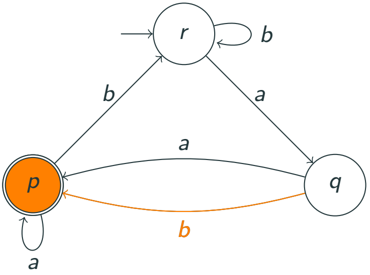
On input string *aba*: r

Example



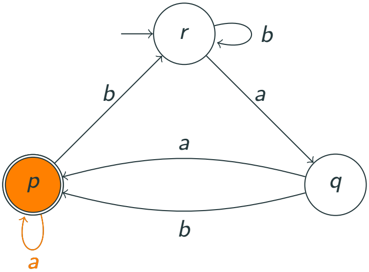
On input string *aba*: $\quad r \quad \underline{a} \quad q$

Example



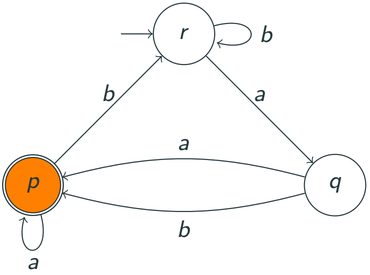
On input string *aba*: $r \quad \underline{a} \quad q \quad \underline{b} \quad p$

Example



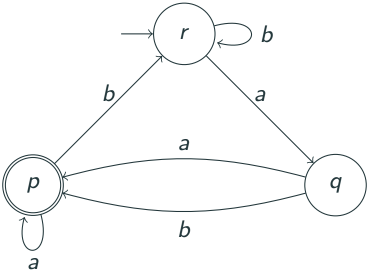
On input string *aba*: $r \quad \underline{a} \quad q \quad \underline{b} \quad p \quad \underline{a} \quad p$

Example



On input string *aba*: $r \;\; \underline{a} \;\; q \;\; \underline{b} \;\; p \;\; \underline{a} \;\; p$ (accepted by DFA)

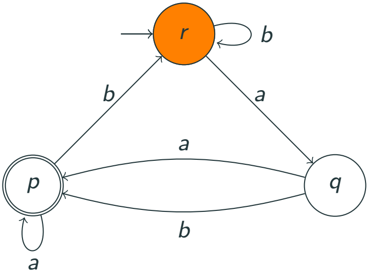
Example



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

On input string *aab*:

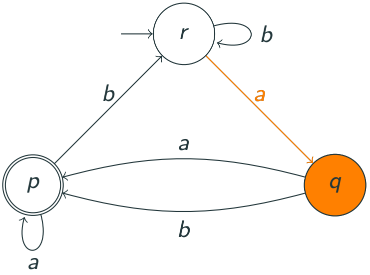
Example



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

On input string *aab*: r

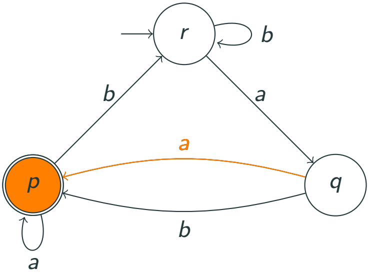
Example



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

On input string *aab*: $r \ \underline{a} \ q$

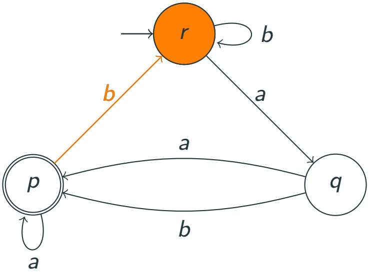
Example



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

On input string *aab*: $r \ \underline{a} \ q \ \underline{a} \ p$

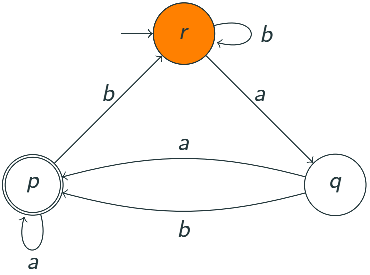
Example



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

On input string *aab*: r $\underline{a}$ q $\underline{a}$ p $\underline{b}$ r

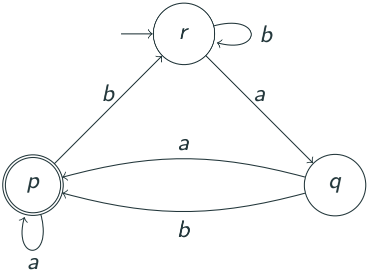
Example



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

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

Example

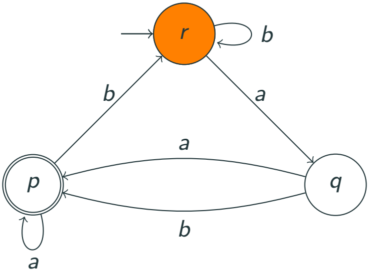


On input string aba: $\quad r \quad \underline{a} \quad q \quad \underline{b} \quad p \quad \underline{a} \quad p$ (accepted by DFA)

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

On input string ε:

Example

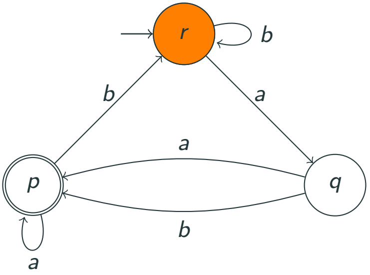


On input string *aba*: $r \quad \underline{a} \quad q \quad \underline{b} \quad p \quad \underline{a} \quad p$ (accepted by DFA)

On input string *aab*: $r \quad \underline{a} \quad q \quad \underline{a} \quad p \quad \underline{b} \quad r$ (not accepted by DFA)

On input string ε: r

Example



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

On input string *aab*: $r \quad \underline{a} \quad q \quad \underline{a} \quad p \quad \underline{b} \quad r$ (not accepted by DFA)

On input string ε: r (not accepted by DFA)

The formal definition of acceptance/rejection of words by DFA

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

(Def.) On input word $w = a_1 \cdots a_n$, *the 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 $\delta(p_i, a_{i+1}) = p_{i+1}$, for each $i = 0, \ldots, n-1$.

The formal definition of acceptance/rejection of words by DFA

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

(Def.) On input word $w = a_1 \cdots a_n$, *the 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 $\delta(p_i, a_{i+1}) = p_{i+1}$, for each $i = 0, \ldots, n-1$.

(Def.) *The run of $\mathcal{A}$ 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 $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$.

(Def.) On input word $w = a_1 \cdots a_n$, *the 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 $\delta(p_i, a_{i+1}) = p_{i+1}$, for each $i = 0, \ldots, n-1$.

(Def.) *The 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 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.

The language accepted by DFA

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

(Def.) We say that $\underline{\mathcal{A} \textit{ accepts } w}$, 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 $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$.

(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 a *regular* language, if there is a DFA $\mathcal{A}$ such that $L(\mathcal{A}) = L$.

Some observations on DFA

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

Some observations on DFA

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

- For every word w, there is exactly *one* run of $\mathcal{A}$ on w.

Some observations on DFA

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

- For every word w, there is exactly *one* run of $\mathcal{A}$ on w.
- The empty string ε is accepted by $\mathcal{A}$ if and only if $q_0 \in F$.

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

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 $0 \bmod 3$

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

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

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

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

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

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

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

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

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

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 $[\![\varepsilon]\!] = 0$.

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

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:

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

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

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:

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

We will show that L_0 is a regular language.

Constructing a DFA for $L_0 := \{w \mid [\![w]\!] \equiv 0 \pmod 3\}$

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

Constructing a DFA for $L_0 := \{w \mid [\![w]\!] \equiv 0 \pmod 3\}$

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

$$[\![wz]\!] \quad = \quad [\![w]\!] \times 2 \; + \; z$$

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:

$$\begin{aligned} \llbracket wz \rrbracket &= \llbracket w \rrbracket \times 2 + z \\ \llbracket wz \rrbracket &\equiv \llbracket w \rrbracket \times 2 + z \pmod 3 \end{aligned}$$

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:

$$
\begin{array}{rcl}
[\![wz]\!] & = & [\![w]\!] \times 2 \; + \; z \\
[\![wz]\!] & \equiv & [\![w]\!] \times 2 \; + \; z \pmod 3 \\
\color{red}{0} & \equiv & \color{red}{0} \times 2 \; + \; \color{red}{0} \pmod 3
\end{array}
$$

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:

$$
\begin{aligned}
[\![wz]\!] &= [\![w]\!] \times 2 + z \\
[\![wz]\!] &\equiv [\![w]\!] \times 2 + z \pmod 3 \\
0 &\equiv 0 \times 2 + 0 \pmod 3 \\
1 &\equiv 0 \times 2 + 1 \pmod 3
\end{aligned}
$$

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:

$$
\begin{array}{rcllll}
[\![wz]\!] & = & [\![w]\!] \times 2 & + & z & \\
[\![wz]\!] & \equiv & [\![w]\!] \times 2 & + & z & \pmod 3 \\
0 & \equiv & 0 \times 2 & + & 0 & \pmod 3 \\
1 & \equiv & 0 \times 2 & + & 1 & \pmod 3 \\
2 & \equiv & 1 \times 2 & + & 0 & \pmod 3
\end{array}
$$

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:

$$
\begin{array}{ccll}
[\![wz]\!] & = & [\![w]\!] \times 2 + z & \\
[\![wz]\!] & \equiv & [\![w]\!] \times 2 + z & \pmod 3 \\
0 & \equiv & 0 \times 2 + 0 & \pmod 3 \\
1 & \equiv & 0 \times 2 + 1 & \pmod 3 \\
2 & \equiv & 1 \times 2 + 0 & \pmod 3 \\
0 & \equiv & 1 \times 2 + 1 & \pmod 3
\end{array}
$$

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:

$$
\begin{array}{ccccccl}
[\![wz]\!] & = & [\![w]\!] & \times 2 & + & z & \\
[\![wz]\!] & \equiv & [\![w]\!] & \times 2 & + & z & \pmod 3 \\
0 & \equiv & 0 & \times 2 & + & 0 & \pmod 3 \\
1 & \equiv & 0 & \times 2 & + & 1 & \pmod 3 \\
2 & \equiv & 1 & \times 2 & + & 0 & \pmod 3 \\
0 & \equiv & 1 & \times 2 & + & 1 & \pmod 3 \\
1 & \equiv & 2 & \times 2 & + & 0 & \pmod 3
\end{array}
$$

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:

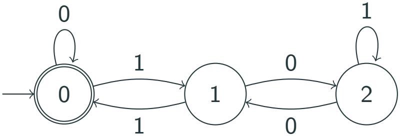
$$
\begin{array}{ccll}
[\![wz]\!] & = & [\![w]\!] \times 2 + z & \\
[\![wz]\!] & \equiv & [\![w]\!] \times 2 + z & \pmod 3 \\
0 & \equiv & 0 \times 2 + 0 & \pmod 3 \\
1 & \equiv & 0 \times 2 + 1 & \pmod 3 \\
2 & \equiv & 1 \times 2 + 0 & \pmod 3 \\
0 & \equiv & 1 \times 2 + 1 & \pmod 3 \\
1 & \equiv & 2 \times 2 + 0 & \pmod 3 \\
2 & \equiv & 2 \times 2 + 1 & \pmod 3
\end{array}
$$

Constructing a DFA for $L_0 := \{w \mid [\![w]\!] \equiv 0 \pmod 3\}$

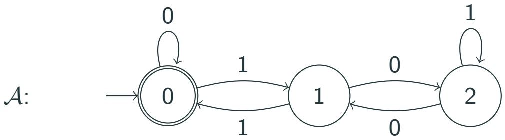
For a word $w \in \{0,1\}^*$ and a symbol $z \in \{0,1\}$, we have the following identity:

$$
\begin{array}{ccl}
[\![wz]\!] & = & [\![w]\!] \times 2 + z \\
[\![wz]\!] & \equiv & [\![w]\!] \times 2 + z \pmod 3 \\
0 & \equiv & 0 \times 2 + 0 \pmod 3 \\
1 & \equiv & 0 \times 2 + 1 \pmod 3 \\
2 & \equiv & 1 \times 2 + 0 \pmod 3 \\
0 & \equiv & 1 \times 2 + 1 \pmod 3 \\
1 & \equiv & 2 \times 2 + 0 \pmod 3 \\
2 & \equiv & 2 \times 2 + 1 \pmod 3
\end{array}
$$

$\mathcal{A}$:

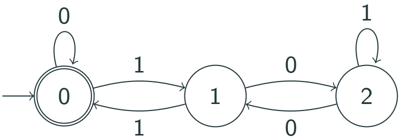


Constructing a DFA for $L_0 := \{w \mid [\![w]\!] \equiv 0 \pmod 3\}$



Constructing a DFA for $L_0 := \{w \mid [\![w]\!] \equiv 0 \pmod 3\}$

$\mathcal{A}$:

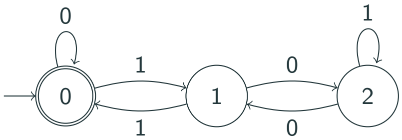


For every word $w \in \{0,1\}^*$:

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

Constructing a DFA for $L_0 := \{w \mid [\![w]\!] \equiv 0 \pmod 3\}$

$\mathcal{A}$:



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

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

So $L(\mathcal{A}) = L_0$.

Important property of regular languages

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 $\mathcal{A}$ over alphabet Σ, there is a DFA $\mathcal{A}'$ over the same alphabet Σ such that $L(\mathcal{A}') = \Sigma^* - L(\mathcal{A})$.

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 $\mathcal{A}$ over alphabet Σ, there is a DFA $\mathcal{A}'$ over the same alphabet Σ such that $L(\mathcal{A}') = \Sigma^* - L(\mathcal{A})$.

Closure under intersection: For every two DFA $\mathcal{A}_1$ and $\mathcal{A}_2$, there is a DFA $\mathcal{A}'$ such that $L(\mathcal{A}') = L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$.

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 $\mathcal{A}$ over alphabet Σ, there is a DFA $\mathcal{A}'$ over the same alphabet Σ such that $L(\mathcal{A}') = \Sigma^* - L(\mathcal{A})$.

Closure under intersection: For every two DFA $\mathcal{A}_1$ and $\mathcal{A}_2$, there is a DFA $\mathcal{A}'$ such that $L(\mathcal{A}') = L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$.

Closure under union: For every two DFA $\mathcal{A}_1$ and $\mathcal{A}_2$, there is a DFA $\mathcal{A}'$ such that $L(\mathcal{A}') = L(\mathcal{A}_1) \cup L(\mathcal{A}_2)$.

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 $\mathcal{A}$ over alphabet Σ, there is a DFA $\mathcal{A}'$ over the same alphabet Σ such that $L(\mathcal{A}') = \Sigma^* - L(\mathcal{A})$.

Closure under intersection: For every two DFA $\mathcal{A}_1$ and $\mathcal{A}_2$, there is a DFA $\mathcal{A}'$ such that $L(\mathcal{A}') = L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$.

Closure under union: For every two DFA $\mathcal{A}_1$ and $\mathcal{A}_2$, there is a DFA $\mathcal{A}'$ such that $L(\mathcal{A}') = L(\mathcal{A}_1) \cup L(\mathcal{A}_2)$.

See Note 1 for the formal proof of Theorem 1.3.

Table of contents

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:

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:

- Σ is an alphabet.

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:

- Σ is an alphabet.
- Q is a finite set of *states*.

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:

- Σ is an alphabet.
- Q is a finite set of *states*.
- $q_0 \in Q$ is the *initial* state.

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:

- Σ 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 $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$ where:

- Σ 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 <u>*transition relation*</u>.

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:

- Σ 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, δ is a function $\delta : Q \times \Sigma \to Q$.

In NFA, δ 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.
- $\underline{F = \emptyset}$, i.e., it does not have any accepting state.
- The transition relation δ is $\{(p, a, q), (r, a, r)\}$.

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.
- $\underline{F = \emptyset}$, i.e., it does not have any accepting state.
- The transition relation δ is $\{(p, a, q), (r, a, r)\}$.

This is a valid NFA.

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.
- $\underline{F = \emptyset}$, i.e., it does not have any accepting state.
- The transition relation δ 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 $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$:

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

Example 2

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.
- $\underline{F = \emptyset}$, i.e., it does not have any accepting state.
- The transition relation δ is $\{(p, a, q), (r, a, r), (p, a, p)\}$.

This is a valid NFA.

Example 2

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.
- $\underline{F = \emptyset}$, i.e., it does not have any accepting state.
- The transition relation δ 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 $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$:

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

Example 3

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.
- $\underline{F = \emptyset}$, i.e., it does not have any accepting state.
- The transition relation δ is $\emptyset$.

This is a valid NFA.

Example 3

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.
- $\underline{F = \emptyset}$, i.e., it does not have any accepting state.
- The transition relation δ 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 δ is a function defined as:

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

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 δ is a function defined as:

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

This is also a valid NFA.

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 δ is a function defined as:

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

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.)

Visualizing NFA

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 δ is as follows.

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

Visualizing NFA

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 δ is as follows.

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

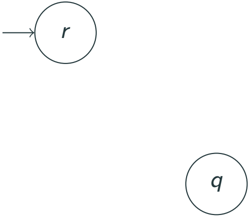
We can visualize it as a directed graph:

Visualizing NFA

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 δ 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:

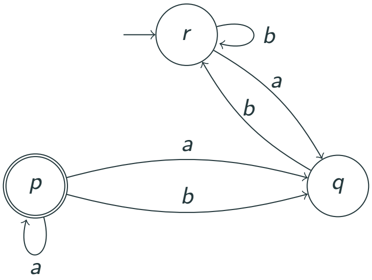


Visualizing NFA

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 δ 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:



Acceptance/rejection of words by NFA

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

(Def.) On input word $w = a_1 \cdots a_n$, <u>a</u> *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 $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$ 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 $p_0 = q$.

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 $\underline{\mathcal{A} \textit{ accepts } w}$, if <u>there is</u> 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 $\underline{\mathcal{A} \textit{ accepts } w}$, if <u>there is</u> 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

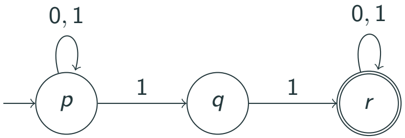
Let $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$.

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

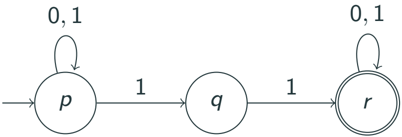
(Def.) The language of all words accepted by $\mathcal{A}$ is denoted by $L(\mathcal{A})$.

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

Example 5

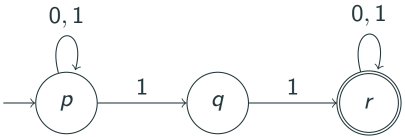


Example 5



On input string 10110, there are many possible runs:

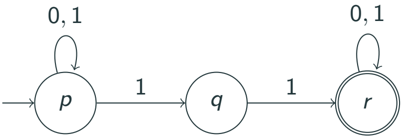
Example 5



On input string 10110, there are many possible runs:

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

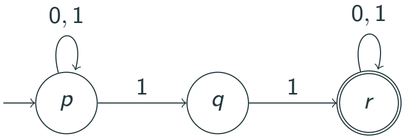
Example 5



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).

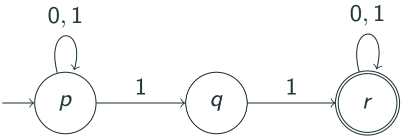
Example 5



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).

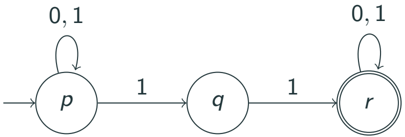
Example 5



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).
- ... (there are many other runs)

Example 5

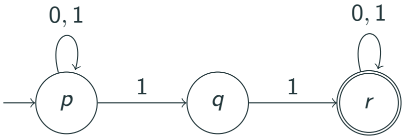


On input string 10110, there are many possible runs:

- $p\ 1\ p\ 0\ p\ 1\ p\ 1\ p\ 0\ p$. (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)

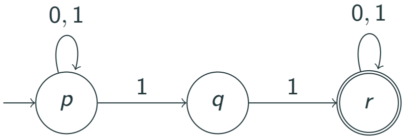
There is an accepting run so $\mathcal{A}$ accepts 10110.

Example 5



On Input word: 10110

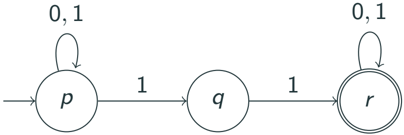
Example 5



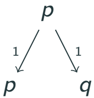
On Input word: 10110

p

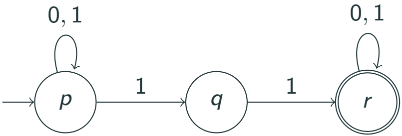
Example 5



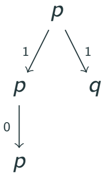
On Input word: 10110



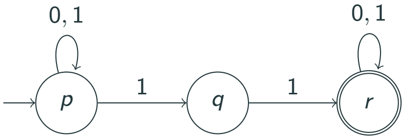
Example 5



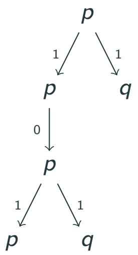
On Input word: 10110



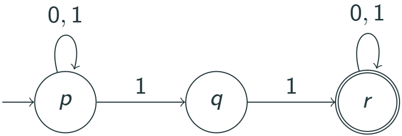
Example 5



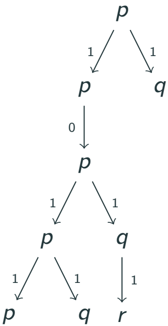
On Input word: 10110



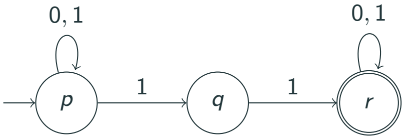
Example 5



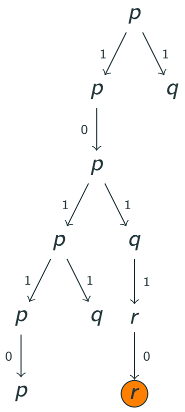
On Input word: 10110



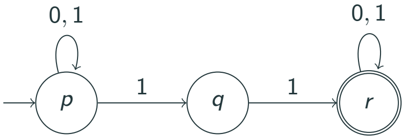
Example 5



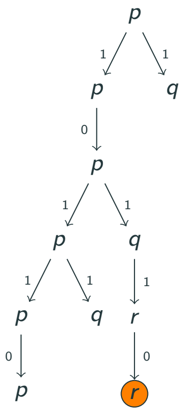
On Input word: 10110



Example 5

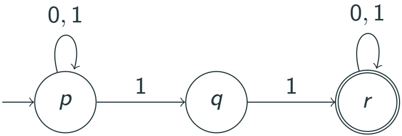


On Input word: 10110

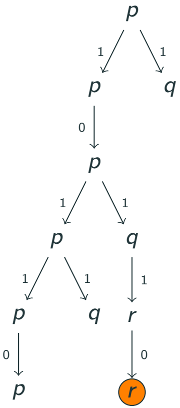


p

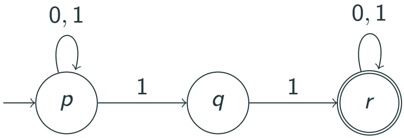
Example 5



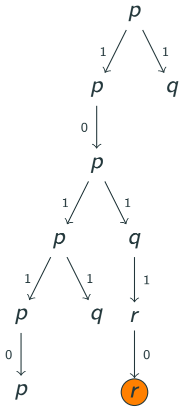
On Input word: 10110

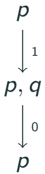


Example 5

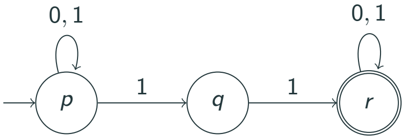


On Input word: 10110

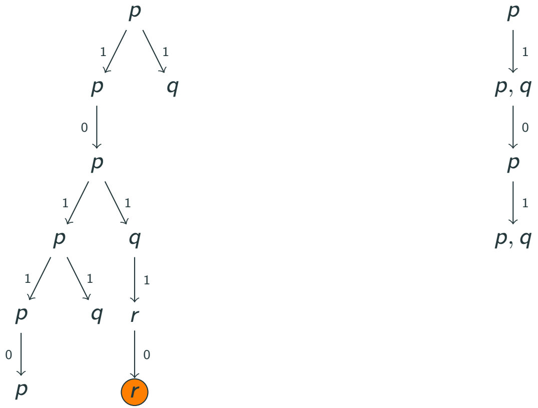




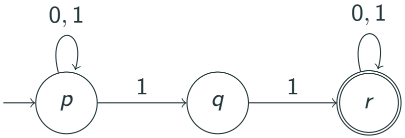
Example 5



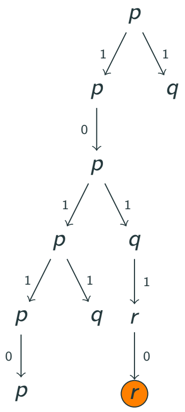
On Input word: 10110

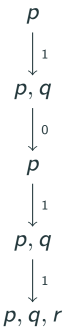


Example 5

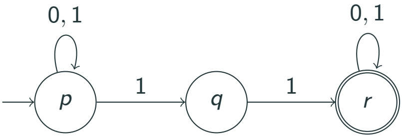


On Input word: 10110

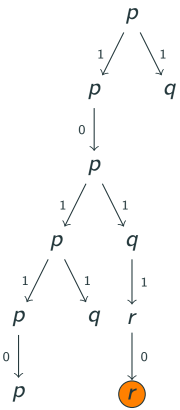


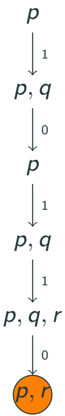


Example 5

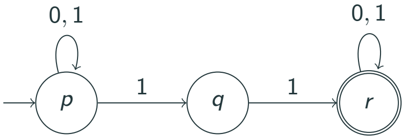


On Input word: 10110

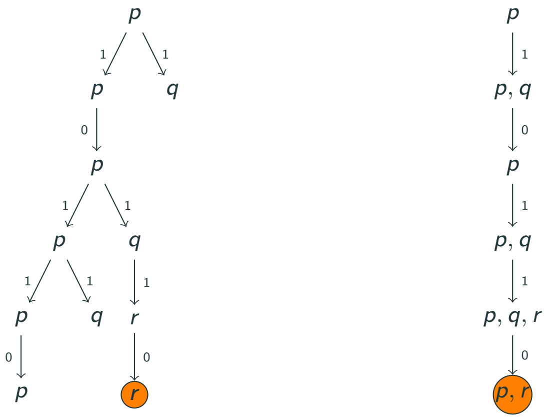




Example 5



On Input word: 10110 (accepted)



Closure under union and intersection

(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 $\mathcal{A}_1$ and $\mathcal{A}_2$, there is an NFA $\mathcal{A}'$ such that $L(\mathcal{A}') = L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$.
- For every two NFA $\mathcal{A}_1$ and $\mathcal{A}_2$, there is an NFA $\mathcal{A}'$ such that $L(\mathcal{A}') = L(\mathcal{A}_1) \cup L(\mathcal{A}_2)$.

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 $\mathcal{A}_1$ and $\mathcal{A}_2$, there is an NFA $\mathcal{A}'$ such that $L(\mathcal{A}') = L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$.
- For every two NFA $\mathcal{A}_1$ and $\mathcal{A}_2$, there is an NFA $\mathcal{A}'$ such that $L(\mathcal{A}') = L(\mathcal{A}_1) \cup L(\mathcal{A}_2)$.

The proof is the same as the one for DFA.

NFA can be converted to 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 $\mathcal{A}$, there is a DFA $\mathcal{A}'$ such that $L(\mathcal{A}) = L(\mathcal{A}')$.

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

NFA can be converted to DFA

Theorem 1.5

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

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

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

NFA can be converted to DFA

Theorem 1.5

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

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

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

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

NFA can be converted to DFA

Theorem 1.5

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

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

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

- $Q' = 2^Q$, i.e., the set of all subsets of Q, including $\emptyset$ and Q.
- The initial state is $\{q_0\}$.

NFA can be converted to DFA

Theorem 1.5

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

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

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

- $Q' = 2^Q$, 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 $S \cap F \neq \emptyset$.

NFA can be converted to DFA

Theorem 1.5

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

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

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

- $Q' = 2^Q$, 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 $\delta : 2^Q \times \Sigma \to 2^Q$ is defined as follows.

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

NFA can be converted to DFA

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

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

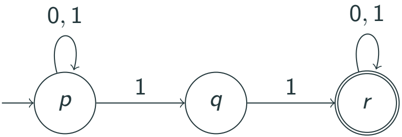
Consider the following DFA $\mathcal{A}' = \langle \Sigma, Q', q_0', F', \delta' \rangle$.

- $Q' = 2^Q$, 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 $\delta : 2^Q \times \Sigma \to 2^Q$ is defined as follows.

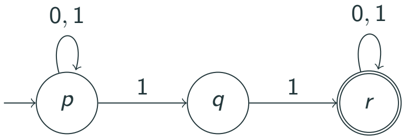
$$\delta'(S, a) \quad = \quad \{p \mid \text{ there is } q \in S \text{ such that } (q, a, p) \in \delta\}$$

It can be shown that $L(\mathcal{A}') = L(\mathcal{A})$. See Note 1 for more details.

The intuitive idea



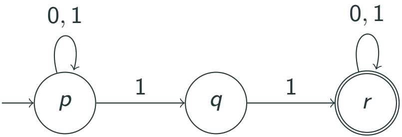
The intuitive idea



On input 10110:

$$\begin{tikzcd}[row sep=large]
p \arrow[d, "1"] \\
p, q \arrow[d, "0"] \\
p \arrow[d, "1"] \\
p, q \arrow[d, "1"] \\
p, q, r \arrow[d, "0"] \\
p, r
\end{tikzcd}$$

The intuitive idea

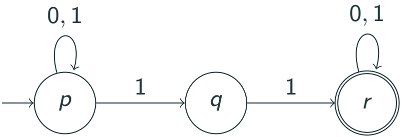


On input 10110:

$$p \xrightarrow{1} p,q \xrightarrow{0} p \xrightarrow{1} p,q \xrightarrow{1} p,q,r \xrightarrow{0} p,r$$

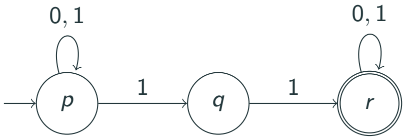
On input w, the set of states it can get to is a subset of $\{p, q, r\}$

The intuitive idea

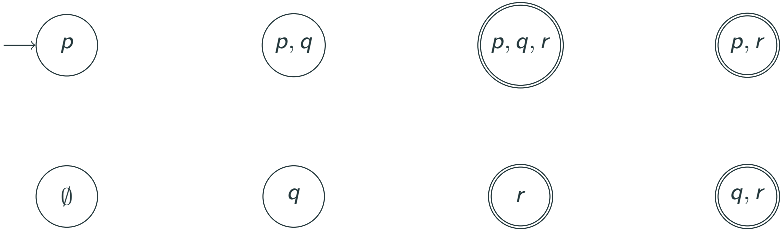


The DFA is:

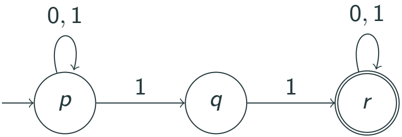
The intuitive idea



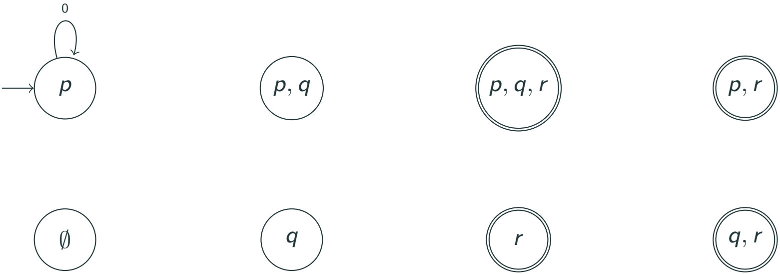
The DFA is:



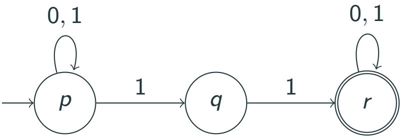
The intuitive idea



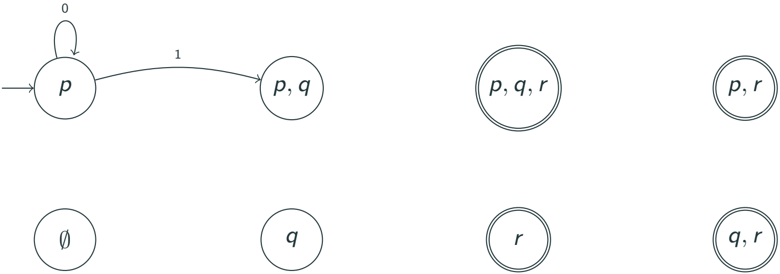
The DFA is:



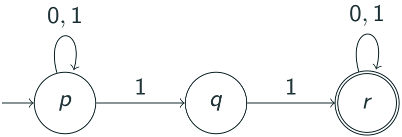
The intuitive idea



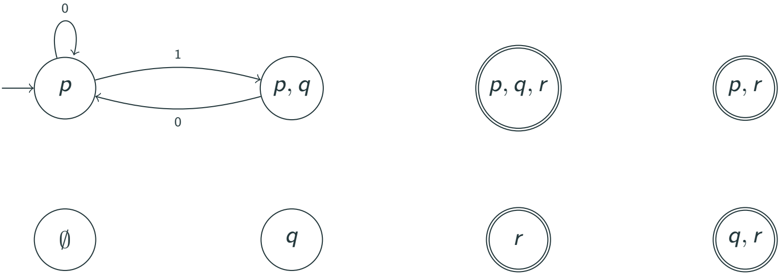
The DFA is:



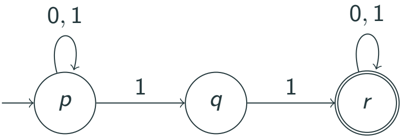
The intuitive idea



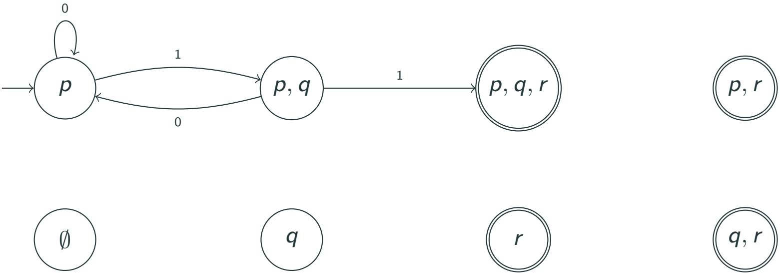
The DFA is:



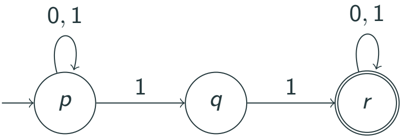
The intuitive idea



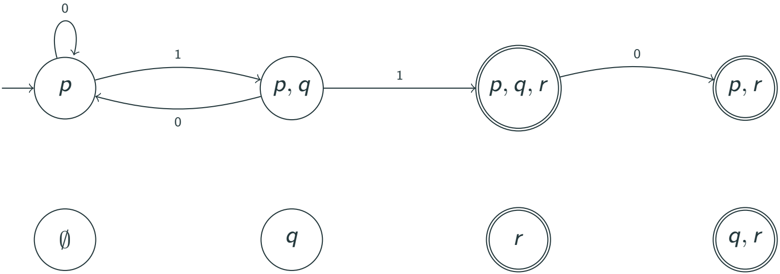
The DFA is:



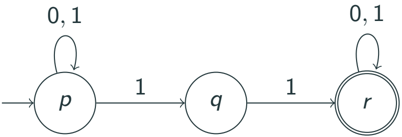
The intuitive idea



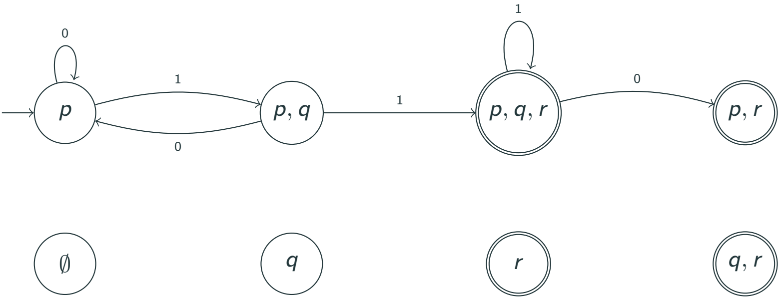
The DFA is:



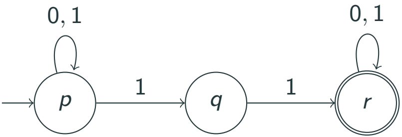
The intuitive idea



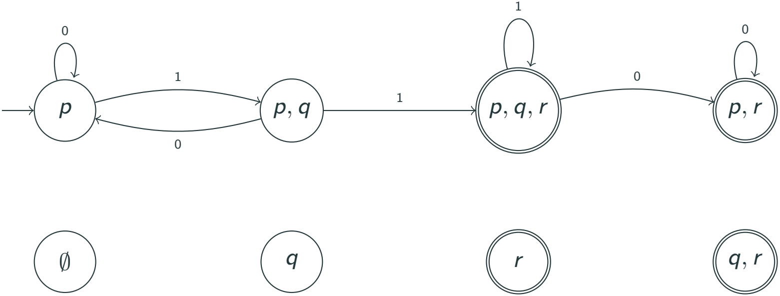
The DFA is:



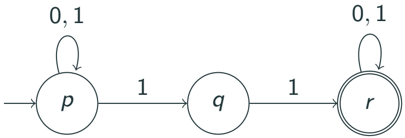
The intuitive idea



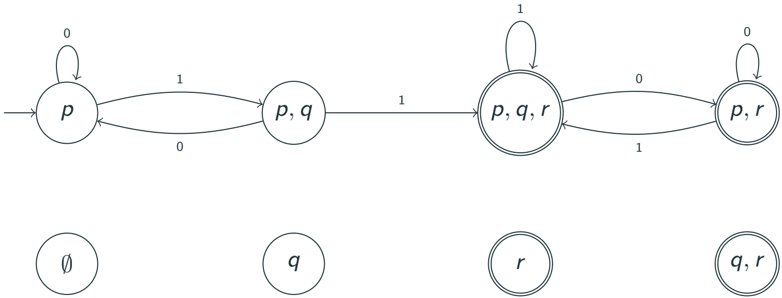
The DFA is:



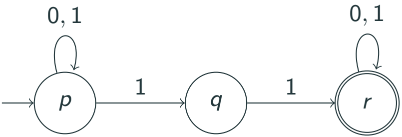
The intuitive idea



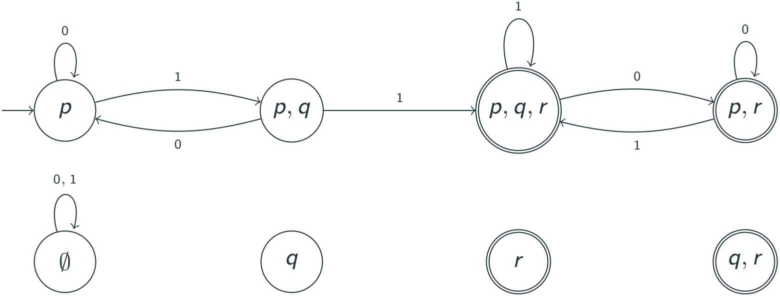
The DFA is:



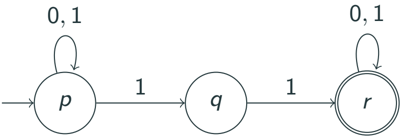
The intuitive idea



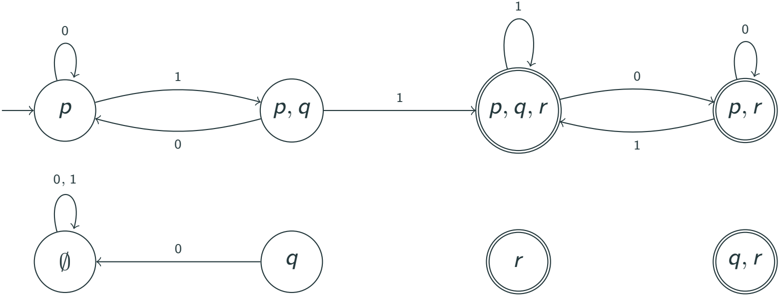
The DFA is:



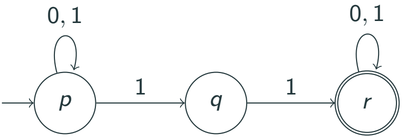
The intuitive idea



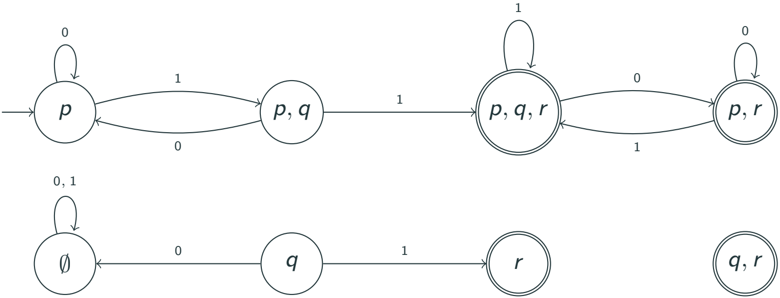
The DFA is:



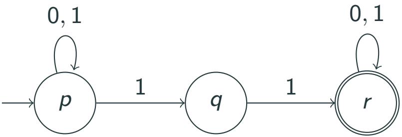
The intuitive idea



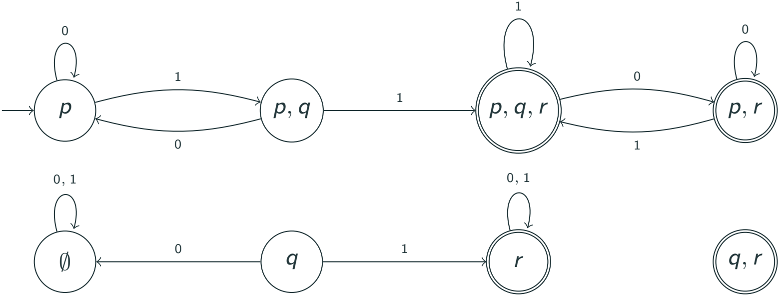
The DFA is:



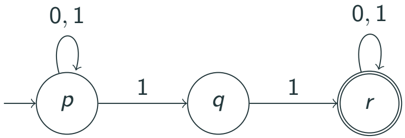
The intuitive idea



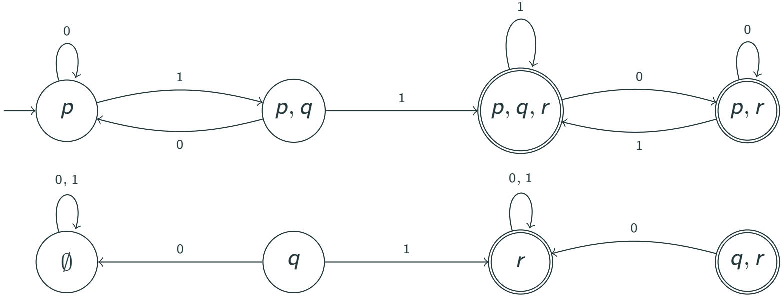
The DFA is:



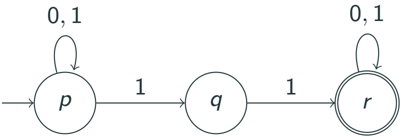
The intuitive idea



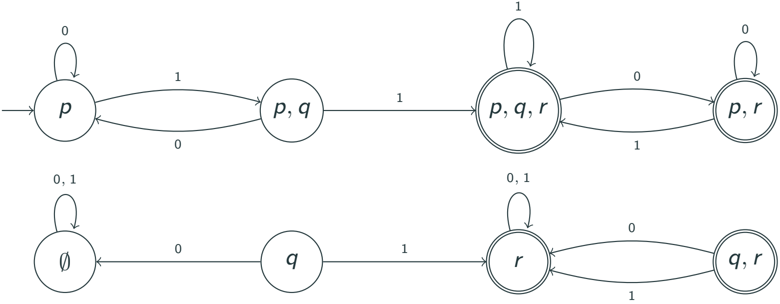
The DFA is:



The intuitive idea



The DFA is:



NFA and DFA

Theorem 1.5

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

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.

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 $\mathcal{A}$, 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.

More precisely, we can say that for every NFA $\mathcal{A}$ over alphabet Σ, there is a DFA $\mathcal{A}'$ over the same alphabet Σ such that $L(\mathcal{A}') = \Sigma^* - L(\mathcal{A})$.

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:

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

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

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

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:

$$
\begin{aligned}
L_1 \cdot L_2 &:= \{uv \mid u \in L_1 \text{ and } v \in L_2\} && \text{(Concatenation)} \\
L^n &:= \{u_1 \cdots u_n \mid \text{each } u_i \in L\} \\
L^* &:= \bigcup_{n \geqslant 0} L^n && \text{(Kleene star)}
\end{aligned}
$$

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

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:

$$
\begin{aligned}
L_1 \cdot L_2 &:= \{uv \mid u \in L_1 \text{ and } v \in L_2\} && \text{(Concatenation)} \\
L^n &:= \{u_1 \cdots u_n \mid \text{each } u_i \in L\} \\
L^* &:= \bigcup_{n \geqslant 0} L^n && \text{(Kleene star)}
\end{aligned}
$$

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.

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:

$$\begin{aligned} L_1 \cdot L_2 &:= \{uv \mid u \in L_1 \text{ and } v \in L_2\} && \text{(Concatenation)} \\ L^n &:= \{u_1 \cdots u_n \mid \text{each } u_i \in L\} \\ L^* &:= \bigcup_{n \geqslant 0} L^n && \text{(Kleene star)} \end{aligned}$$

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

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 $L_1 L_2$.
- 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 $L_1 L_2$.
- If L is a regular language, so is L^*.

The proof can be found in Note 1.

Table of contents

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

(Def.) For a word w and an integer $n \geqslant 0$, 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 $w^0 = \varepsilon$.

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

(Def.) For a word w and an integer $n \geqslant 0$, 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 $\mathcal{A} = \langle \Sigma, Q, q_0, F, \delta \rangle$ be an NFA. Let $x \in L(\mathcal{A})$ be a word such that $|x| \geqslant |Q|$. Then, the word x can be divided into three parts u, v, w, i.e., $x = uvw$, such that $|v| \geqslant 1$ and for every integer $k \geqslant 0$, $uv^k w \in L(\mathcal{A})$.

Proof of pumping lemma

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

Proof of pumping lemma

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

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 $x \in L(\mathcal{A})$, where $n \geqslant |Q|$.

Let the following be its accepting run:

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

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

Proof of pumping lemma

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

Let the following be its accepting run:

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

Since $n \geqslant |Q|$, there are $0 \leqslant i < j \leqslant n$ 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 $x \in L(\mathcal{A})$, where $n \geqslant |Q|$.

Let the following be its accepting run:

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

Since $n \geqslant |Q|$, there are $0 \leqslant i < j \leqslant n$ 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 $k \geqslant 0$, the following is an accepting run of $\mathcal{A}$ on $uv^k w$:

$$p_0 \ a_1 \ p_1 \ a_2 \ p_2 \ \cdots \ a_i \ p_i \ \underbrace{a_{i+1} \ p_{i+1} \ \cdots \ a_j \ p_j}_{\text{repeat } k \text{ times}} \ a_{j+1} \ p_{j+1} \ \cdots \ a_n \ p_n$$

Variations of pumping lemma

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

Variations of pumping lemma

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

Pumping lemma can also be stated more elegantly as follows.

Lemma 1.10 (pumping lemma)
For every regular language L, there is an integer $n \geqslant 1$ such that for every word $x \in L$ with length $|x| \geqslant n$, there are u, v, w where $x = uvw$ and $|v| \geqslant 1$ and for every integer $k \geqslant 0$, $uv^k w \in L$.

Using pumping lemma to prove non-regularity

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

Using pumping lemma to prove non-regularity

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

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

Using pumping lemma to prove non-regularity

We would like to show that $L_1 = \{a^k b^k \mid k \geqslant 0\}$ 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 $L_1 = \{a^k b^k \mid k \geqslant 0\}$ 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 $k \geqslant |Q|$.

Using pumping lemma to prove non-regularity

We would like to show that $L_1 = \{a^k b^k \mid k \geqslant 0\}$ 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 $k \geqslant |Q|$.

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

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

Using pumping lemma to prove non-regularity

We would like to show that $L_1 = \{a^k b^k \mid k \geqslant 0\}$ 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 $k \geqslant |Q|$.

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

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

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 $L_1 = \{a^k b^k \mid k \geqslant 0\}$ 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 $k \geqslant |Q|$.

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

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

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 $\mathcal{A}$ that accepts L_2 where Q is the set of states.

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

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 $\mathcal{A}$ that accepts L_2 where Q is the set of states.

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

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

$$u \; v^{\ell} \; w \qquad \in L(\mathcal{A}) \qquad\qquad \text{for every } \ell \geqslant 0$$

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 $k \geqslant |Q|$.

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

$$u\ v^{\ell}\ w \qquad \in L(\mathcal{A}) \qquad\qquad \text{for every } \ell \geqslant 0$$

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

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 $k \geqslant |Q|$.

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

$$u\ v^\ell\ w \qquad \in L(\mathcal{A}) \qquad\qquad \text{for every } \ell \geqslant 0$$

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

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

$$|u\ v^\ell\ w| \ = \ (|u| + |w|)(|v| + 1) \qquad \text{which is not prime}$$

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 $k \geqslant |Q|$.

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

$$u\ v^\ell\ w \quad \in L(\mathcal{A}) \qquad \text{for every } \ell \geqslant 0$$

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

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

$$|u\ v^\ell\ w| \ = \ (|u| + |w|)(|v| + 1) \qquad \text{which is not prime}$$

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 $k \geqslant |Q|$.

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

$$u\ v^\ell\ w \qquad \in L(\mathcal{A}) \qquad\qquad \text{for every } \ell \geqslant 0$$

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

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

$$|u\ v^\ell\ w| \ = \ (|u| + |w|)(|v| + 1) \qquad \text{which is not prime}$$

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