Automata Theory :: Finite Automata

Jörg Endrullis

Vrije Universiteit Amsterdam

Deterministic Finite Automata

Deterministic Finite Automata (DFAs)

A **deterministic finite automaton**, short **DFA**, consists of:

- a finite set Q of **states**
- a finite **input alphabet** Σ
- a **transition function** $\delta : Q \times \Sigma \to Q$
- a **starting state** $q_0 \in Q$
- a set $F \subseteq Q$ of **final states**

Deterministic Finite Automata (DFAs)

A **deterministic finite automaton**, short **DFA**, consists of:

- a finite set Q of **states**
- a finite **input alphabet** Σ
- a **transition function** $\delta : Q \times \Sigma \to Q$
- a **starting state** $q_0 \in Q$
- a set $F \subseteq Q$ of **final states**

Example DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ with $Q = \{q_0, q_1\}$, $\Sigma = \{a, b\}$, $F = \{q_0\}$,

$$\delta(q_0, a) = q_0 \qquad \delta(q_1, a) = q_1$$

$$\delta(q_0, b) = q_1 \qquad \delta(q_1, b) = q_0$$

Deterministic Finite Automata (DFAs)

A **deterministic finite automaton**, short **DFA**, consists of:

- a finite set Q of **states**
- a finite **input alphabet** Σ
- a **transition function** $\delta : Q \times \Sigma \to Q$
- a **starting state** $q_0 \in Q$
- a set $F \subseteq Q$ of **final states**

Example DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ with $Q = \{q_0, q_1\}$, $\Sigma = \{a, b\}$, $F = \{q_0\}$,

$$\delta(q_0, a) = q_0 \qquad \delta(q_1, a) = q_1$$
$$\delta(q_0, b) = q_1 \qquad \delta(q_1, b) = q_0$$

Understanding the transition function $\delta : Q \times \Sigma \to Q$

If the automaton in state q reads the symbol a,
then the resulting state is $\delta(q, a)$.

DFAs Reading Words

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA.

A **configuration** of M is a pair (q, w) with $q \in Q$ and $w \in \Sigma^*$.

So (q, w) means the automaton is in state q and reads word w.

The **step relation** $\vdash$ of M is defined on configurations by

$$(q, aw) \vdash (q', w) \quad \text{if } \delta(q, a) = q'$$

DFAs Reading Words

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA.

A **configuration** of M is a pair (q, w) with $q \in Q$ and $w \in \Sigma^*$.

So (q, w) means the automaton is in state q and reads word w.

The **step relation** $\vdash$ of M is defined on configurations by

$$(q, aw) \vdash (q', w) \quad \text{if } \delta(q, a) = q'$$

Let $M = (Q, \Sigma, \delta, q_0, F)$ with $Q = \{q_0, q_1\}$, $\Sigma = \{a, b\}$, $F = \{q_0\}$,

$$\delta(q_0, a) = q_0 \qquad \delta(q_1, a) = q_1$$
$$\delta(q_0, b) = q_1 \qquad \delta(q_1, b) = q_0$$

Then $(q_0, abba) \vdash$

DFAs Reading Words

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA.

A **configuration** of M is a pair (q, w) with $q \in Q$ and $w \in \Sigma^*$.

So (q, w) means the automaton is in state q and reads word w.

The **step relation** $\vdash$ of M is defined on configurations by

$$(q, aw) \vdash (q', w) \quad \text{if } \delta(q, a) = q'$$

Let $M = (Q, \Sigma, \delta, q_0, F)$ with $Q = \{q_0, q_1\}$, $\Sigma = \{a, b\}$, $F = \{q_0\}$,

$$\delta(q_0, a) = q_0 \qquad \delta(q_1, a) = q_1$$
$$\delta(q_0, b) = q_1 \qquad \delta(q_1, b) = q_0$$

Then $(q_0, abba) \vdash (q_0, bba) \vdash$

DFAs Reading Words

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA.

A **configuration** of M is a pair (q, w) with $q \in Q$ and $w \in \Sigma^*$.

So (q, w) means the automaton is in state q and reads word w.

The **step relation** $\vdash$ of M is defined on configurations by

$$(q, aw) \vdash (q', w) \quad \text{if } \delta(q, a) = q'$$

Let $M = (Q, \Sigma, \delta, q_0, F)$ with $Q = \{q_0, q_1\}$, $\Sigma = \{a, b\}$, $F = \{q_0\}$,

$$\delta(q_0, a) = q_0 \qquad \delta(q_1, a) = q_1$$
$$\delta(q_0, b) = q_1 \qquad \delta(q_1, b) = q_0$$

Then $(q_0, abba) \vdash (q_0, bba) \vdash (q_1, ba) \vdash$

DFAs Reading Words

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA.

A **configuration** of M is a pair (q, w) with $q \in Q$ and $w \in \Sigma^*$.

So (q, w) means the automaton is in state q and reads word w.

The **step relation** $\vdash$ of M is defined on configurations by

$$(q, aw) \vdash (q', w) \quad \text{if } \delta(q, a) = q'$$

Let $M = (Q, \Sigma, \delta, q_0, F)$ with $Q = \{q_0, q_1\}$, $\Sigma = \{a, b\}$, $F = \{q_0\}$,

$$\delta(q_0, a) = q_0 \qquad \delta(q_1, a) = q_1$$
$$\delta(q_0, b) = q_1 \qquad \delta(q_1, b) = q_0$$

Then $(q_0, abba) \vdash (q_0, bba) \vdash (q_1, ba) \vdash (q_0, a) \vdash$

DFAs Reading Words

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA.

A **configuration** of M is a pair (q, w) with $q \in Q$ and $w \in \Sigma^*$.

So (q, w) means the automaton is in state q and reads word w.

The **step relation** $\vdash$ of M is defined on configurations by

$$(q, aw) \vdash (q', w) \quad \text{if } \delta(q, a) = q'$$

Let $M = (Q, \Sigma, \delta, q_0, F)$ with $Q = \{q_0, q_1\}$, $\Sigma = \{a, b\}$, $F = \{q_0\}$,

$$\delta(q_0, a) = q_0 \qquad \delta(q_1, a) = q_1$$
$$\delta(q_0, b) = q_1 \qquad \delta(q_1, b) = q_0$$

Then $(q_0, abba) \vdash (q_0, bba) \vdash (q_1, ba) \vdash (q_0, a) \vdash (q_0, \lambda)$.

DFAs Reading Words

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA.

A **configuration** of M is a pair (q, w) with $q \in Q$ and $w \in \Sigma^*$.

So (q, w) means the automaton is in state q and reads word w.

The **step relation** $\vdash$ of M is defined on configurations by

$$(q, aw) \vdash (q', w) \quad \text{if } \delta(q, a) = q'$$

Let $M = (Q, \Sigma, \delta, q_0, F)$ with $Q = \{q_0, q_1\}$, $\Sigma = \{a, b\}$, $F = \{q_0\}$,

$$\delta(q_0, a) = q_0 \qquad \delta(q_1, a) = q_1$$
$$\delta(q_0, b) = q_1 \qquad \delta(q_1, b) = q_0$$

Then $(q_0, abba) \vdash (q_0, bba) \vdash (q_1, ba) \vdash (q_0, a) \vdash (q_0, \lambda)$.

We define $\vdash^*$ as the **reflexive transitive closure of** $\vdash$.

Continuing the above example, we have $(q_0, abba) \vdash^* (q_0, \lambda)$.

Transition Function in Table Notation

Example DFA

Let $M = (Q, \Sigma, \delta, q_0, F)$ with $Q = \{q_0, q_1\}$, $\Sigma = \{a, b\}$, $F = \{q_0\}$,

$$\delta(q_0, a) = q_0 \qquad \delta(q_1, a) = q_1$$
$$\delta(q_0, b) = q_1 \qquad \delta(q_1, b) = q_0$$

Hint: transition function δ can be written in the form of a **table**:

δ	q_0	q_1
a	q_0	q_1
b	q_1	q_0

DFAs as Transition Graphs

A DFA can be visualised as a **transition graph**, consisting of:

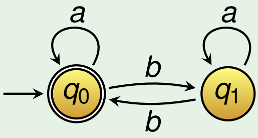
- **states** are the **nodes** of the graph
  - **starting state** indicated by an **extra incoming arrow**
  - **final states** indicated by **double circle**
- **arrows** with labels from Σ: $q \xrightarrow{a} q'$ if $\delta(q, a) = q'$

Let $M = (Q, \Sigma, \delta, q_0, F)$ with $Q = \{q_0, q_1\}$, $\Sigma = \{a, b\}$, $F = \{q_0\}$,

$$\delta(q_0, a) = q_0 \qquad \delta(q_1, a) = q_1$$

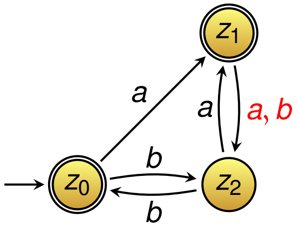
$$\delta(q_0, b) = q_1 \qquad \delta(q_1, b) = q_0$$

is visualised as the transition graph



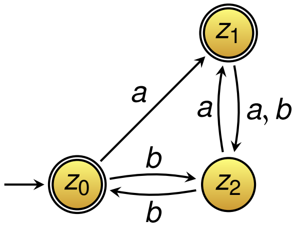
Exercise

An arrow with label a, b is shorthand for two arrows: one with label a and one with label b.



Exercise

An arrow with label a, b is shorthand for two arrows: one with label a and one with label b.



What is this DFA?

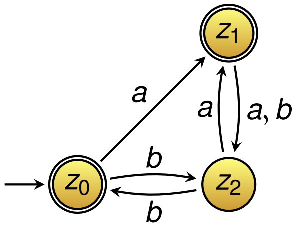
- states $Q =$
- alphabet $\Sigma =$
- transition function $\delta : Q \times \Sigma \to Q$:

δ	z_0	z_1	z_2
a			
b			

- starting state
- final states $F =$

Exercise

An arrow with label a, b is shorthand for two arrows: one with label a and one with label b.



What is this DFA?

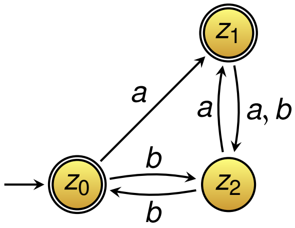
- states $Q = \{ z_0, z_1, z_2 \}$
- alphabet $\Sigma =$
- transition function $\delta : Q \times \Sigma \to Q$:

δ	z_0	z_1	z_2
a			
b			

- starting state
- final states $F =$

Exercise

An arrow with label a, b is shorthand for two arrows: one with label a and one with label b.



What is this DFA?

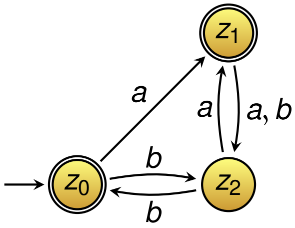
- states $Q = \{ z_0, z_1, z_2 \}$
- alphabet $\Sigma = \{ a, b \}$
- transition function $\delta : Q \times \Sigma \to Q$:

δ	z_0	z_1	z_2
a			
b			

- starting state
- final states $F =$

Exercise

An arrow with label a, b is shorthand for two arrows: one with label a and one with label b.



What is this DFA?

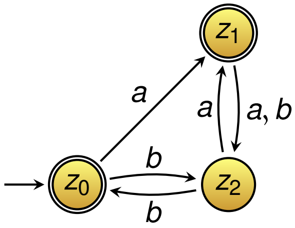
- states $Q = \{ z_0, z_1, z_2 \}$
- alphabet $\Sigma = \{ a, b \}$
- transition function $\delta : Q \times \Sigma \to Q$:

δ	z_0	z_1	z_2
a	z_1		
b			

- starting state
- final states $F =$

Exercise

An arrow with label a, b is shorthand for two arrows: one with label a and one with label b.



What is this DFA?

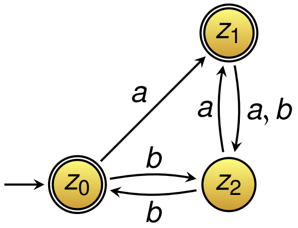
- states $Q = \{ z_0, z_1, z_2 \}$
- alphabet $\Sigma = \{ a, b \}$
- transition function $\delta : Q \times \Sigma \to Q$:

δ	z_0	z_1	z_2
a	z_1	z_2	
b			

- starting state
- final states $F =$

Exercise

An arrow with label a, b is shorthand for two arrows: one with label a and one with label b.



What is this DFA?

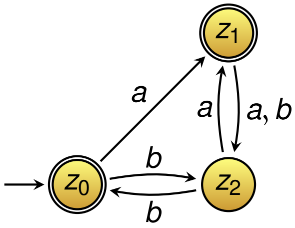
- states $Q = \{ z_0, z_1, z_2 \}$
- alphabet $\Sigma = \{ a, b \}$
- transition function $\delta : Q \times \Sigma \to Q$:

δ	z_0	z_1	z_2
a	z_1	z_2	z_1
b			

- starting state
- final states $F =$

Exercise

An arrow with label a, b is shorthand for two arrows: one with label a and one with label b.



What is this DFA?

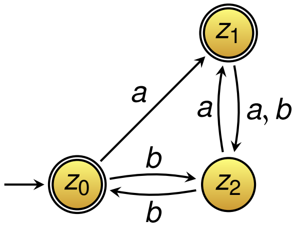
- states $Q = \{ z_0, z_1, z_2 \}$
- alphabet $\Sigma = \{ a, b \}$
- transition function $\delta : Q \times \Sigma \to Q$:

δ	z_0	z_1	z_2
a	z_1	z_2	z_1
b	z_2		

- starting state
- final states $F =$

Exercise

An arrow with label a, b is shorthand for two arrows: one with label a and one with label b.



What is this DFA?

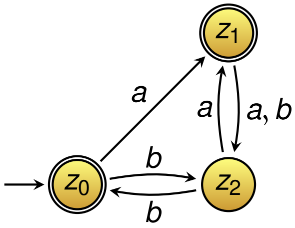
- states $Q = \{ z_0, z_1, z_2 \}$
- alphabet $\Sigma = \{ a, b \}$
- transition function $\delta : Q \times \Sigma \to Q$:

δ	z_0	z_1	z_2
a	z_1	z_2	z_1
b	z_2	z_2	

- starting state
- final states $F =$

Exercise

An arrow with label a, b is shorthand for two arrows: one with label a and one with label b.



What is this DFA?

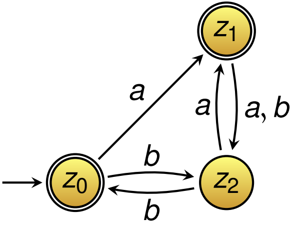
- states $Q = \{ z_0, z_1, z_2 \}$
- alphabet $\Sigma = \{ a, b \}$
- transition function $\delta : Q \times \Sigma \to Q$:

δ	z_0	z_1	z_2
a	z_1	z_2	z_1
b	z_2	z_2	z_0

- starting state
- final states $F =$

Exercise

An arrow with label a, b is shorthand for two arrows: one with label a and one with label b.



What is this DFA?

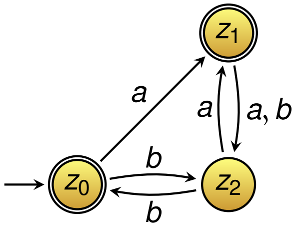
- states $Q = \{ z_0, z_1, z_2 \}$
- alphabet $\Sigma = \{ a, b \}$
- transition function $\delta : Q \times \Sigma \to Q$:

δ	z_0	z_1	z_2
a	z_1	z_2	z_1
b	z_2	z_2	z_0

- starting state z_0
- final states $F =$

Exercise

An arrow with label a, b is shorthand for two arrows: one with label a and one with label b.



What is this DFA?

- states $Q = \{ z_0, z_1, z_2 \}$
- alphabet $\Sigma = \{ a, b \}$
- transition function $\delta : Q \times \Sigma \to Q$:

δ	z_0	z_1	z_2
a	z_1	z_2	z_1
b	z_2	z_2	z_0

- starting state z_0
- final states $F = \{ z_0, z_1 \}$

Paths in DFAs

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA.

For a word $w = a_1 \cdots a_n$, $n \geq 0$, we write

$$q_0 \overset{w}{\longrightarrow\!\!\!\rightarrow} q_n$$

if there are states $q_1, \ldots, q_{n-1}$ such that

$$q_0 \xrightarrow{a_1} q_1 \qquad q_1 \xrightarrow{a_2} q_2 \qquad \ldots \qquad q_{n-1} \xrightarrow{a_n} q_n$$

Paths in DFAs

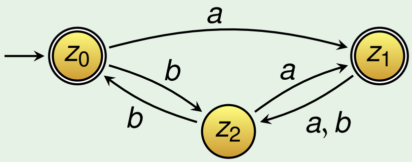
Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA.

For a word $w = a_1 \cdots a_n$, $n \geq 0$, we write

$$q_0 \xrightarrow{w}\!\!\!\rightarrow q_n$$

if there are states $q_1, \ldots, q_{n-1}$ such that

$$q_0 \xrightarrow{a_1} q_1 \qquad q_1 \xrightarrow{a_2} q_2 \qquad \ldots \qquad q_{n-1} \xrightarrow{a_n} q_n$$



$$z_1 \xrightarrow{\lambda}\!\!\!\rightarrow z_1$$

$$z_0 \xrightarrow{ab}\!\!\!\rightarrow z_2$$

$$z_0 \xrightarrow{abba}\!\!\!\rightarrow z_1$$

Paths in DFAs

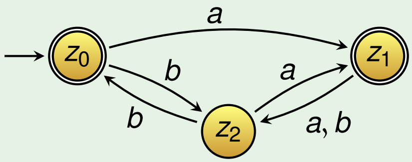
Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA.

For a word $w = a_1 \cdots a_n$, $n \geq 0$, we write

$$q_0 \xrightarrow{w}\!\!\!\!\rightarrow q_n$$

if there are states $q_1, \ldots, q_{n-1}$ such that

$$q_0 \xrightarrow{a_1} q_1 \qquad q_1 \xrightarrow{a_2} q_2 \qquad \ldots \qquad q_{n-1} \xrightarrow{a_n} q_n$$



$z_1 \xrightarrow{\lambda}\!\!\!\!\rightarrow z_1$

$z_0 \xrightarrow{ab}\!\!\!\!\rightarrow z_2$

$z_0 \xrightarrow{abba}\!\!\!\!\rightarrow z_1$

Theorem: $q \xrightarrow{w}\!\!\!\!\rightarrow q' \iff (q, w) \vdash^* (q', \lambda)$.

Regular Languages

A DFA defines (accepts) a language!

Regular Languages

A DFA defines (accepts) a language!

The **language accepted by** DFA $M = (Q, \Sigma, \delta, q_0, F)$ is

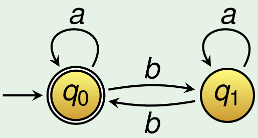
$$\begin{aligned} L(M) &= \{\, w \in \Sigma^* \mid (q_0, w) \vdash^* (q, \lambda) \text{ with } q \in F \,\} \\ &= \{\, w \in \Sigma^* \mid q_0 \xrightarrow{w}\!\!\twoheadrightarrow q \text{ with } q \in F \,\} \end{aligned}$$

Regular Languages

A DFA defines (accepts) a language!

The **language accepted by** DFA $M = (Q, \Sigma, \delta, q_0, F)$ is

$$L(M) = \{ w \in \Sigma^* \mid (q_0, w) \vdash^* (q, \lambda) \text{ with } q \in F \}$$
$$= \{ w \in \Sigma^* \mid q_0 \xrightarrow{w}\!\!\!\twoheadrightarrow q \text{ with } q \in F \}$$



We have

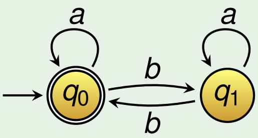
$$(q_0, abba) \vdash (q_0, bba) \vdash (q_1, ba) \vdash (q_0, a) \vdash (q_0, \lambda)$$

Regular Languages

A DFA defines (accepts) a language!

The **language accepted by** DFA $M = (Q, \Sigma, \delta, q_0, F)$ is

$$L(M) = \{ w \in \Sigma^* \mid (q_0, w) \vdash^* (q, \lambda) \text{ with } q \in F \}$$
$$= \{ w \in \Sigma^* \mid q_0 \xrightarrow{w}\!\!\!\!\twoheadrightarrow q \text{ with } q \in F \}$$



We have

$$(q_0, abba) \vdash (q_0, bba) \vdash (q_1, ba) \vdash (q_0, a) \vdash (q_0, \lambda)$$

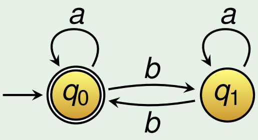
The word *abba* is accepted by M, that is, $abba \in L(M)$.

Regular Languages

A DFA defines (accepts) a language!

The **language accepted by** DFA $M = (Q, \Sigma, \delta, q_0, F)$ is

$$L(M) = \{ w \in \Sigma^* \mid (q_0, w) \vdash^* (q, \lambda) \text{ with } q \in F \}$$
$$= \{ w \in \Sigma^* \mid q_0 \xrightarrow{w}\!\!\twoheadrightarrow q \text{ with } q \in F \}$$



We have

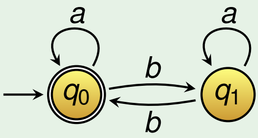
$$(q_0, abba) \vdash (q_0, bba) \vdash (q_1, ba) \vdash (q_0, a) \vdash (q_0, \lambda)$$

The word *abba* is accepted by M, that is, $abba \in L(M)$.

A language L is **regular** if there exists a DFA M with $L(M) = L$.

Exercise (1)

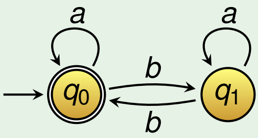
Let M be the following DFA:



Describe the language accepted by M.

Exercise (1)

Let M be the following DFA:



Describe the language accepted by M.

Answer:

$L(M)$ consists of all words over the alphabet $\{ a, b \}$ that contain an even number of b's.

Exercise (2)

Show that the following language is regular:

$$\{\lambda\}$$

Construct a deterministic finite automaton for the language.

Exercise (3)

Show that the following language is regular:

$$\{\, a^n b \mid n \geq 0 \,\}$$

Construct a deterministic finite automaton for the language.

Exercise (4)

Show that the following language is regular:

$$\{ a^{2n+1} \mid n \geq 0 \} \cup \{ b^{2n} \mid n \geq 0 \}$$

Construct a deterministic finite automaton for the language.

DFAs are Deterministic

Recall that δ is a function from $Q \times \Sigma$ to Q.

DFAs are deterministic:

For every state $q \in Q$ and every symbol $a \in \Sigma$, the state q has **precisely one outgoing arrow** with label a.

DFAs are Deterministic

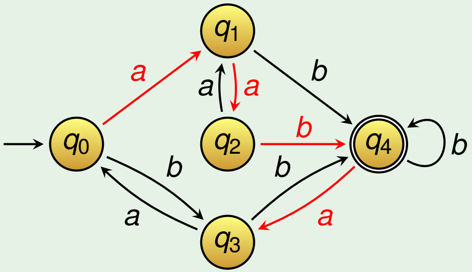
Recall that δ is a function from $Q \times \Sigma$ to Q.

DFAs are deterministic:

For every state $q \in Q$ and every symbol $a \in \Sigma$, the state q has **precisely one outgoing arrow** with label a.

Hence, for every input word, there is precisely one path from the starting state through the transition graph.

The following picture shows the path for *aaba*:



Exercise (5)

Construct deterministic finite automata for the languages:

$$\{\, w \in \{a,b\}^* \mid w \text{ contains the subword } bab \,\}$$

and

$$\{\, w \in \{a,b\}^* \mid w \text{ does } \textbf{not} \text{ contain the subword } bab \,\}$$

Regular Languages: Complement

Theorem

If L is a regular language, then $\overline{L}$ is also regular.

Regular Languages: Complement

Theorem

If L is a regular language, then $\overline{L}$ is also regular.

Proof.

Let L be regular.

Regular Languages: Complement

Theorem

If L is a regular language, then $\overline{L}$ is also regular.

Proof.

Let L be regular.

Then there exists a DFA $M = (Q, \Sigma, \delta, q_0, F)$ with $L(M) = L$.

Regular Languages: Complement

Theorem

If L is a regular language, then $\overline{L}$ is also regular.

Proof.

Let L be regular.

Then there exists a DFA $M = (Q, \Sigma, \delta, q_0, F)$ with $L(M) = L$.

Then $N = (Q, \Sigma, \delta, q_0, Q \setminus F)$ is a DFA with $L(N) = \overline{L}$.

Regular Languages: Complement

Theorem

If L is a regular language, then $\overline{L}$ is also regular.

Proof.

Let L be regular.

Then there exists a DFA $M = (Q, \Sigma, \delta, q_0, F)$ with $L(M) = L$.

Then $N = (Q, \Sigma, \delta, q_0, Q \setminus F)$ is a DFA with $L(N) = \overline{L}$.

Here it is important that for every input word w:

- There is precisely one path starting at q_0 labelled with w.

Regular Languages: Complement

Theorem

If L is a regular language, then $\overline{L}$ is also regular.

Proof.

Let L be regular.

Then there exists a DFA $M = (Q, \Sigma, \delta, q_0, F)$ with $L(M) = L$.

Then $N = (Q, \Sigma, \delta, q_0, Q \setminus F)$ is a DFA with $L(N) = \overline{L}$.

Here it is important that for every input word w:

- There is precisely one path starting at q_0 labelled with w.
- There is precisely one state q with $q_0 \xrightarrow{w}\!\!\twoheadrightarrow q$.

Regular Languages: Complement

Theorem

If L is a regular language, then $\overline{L}$ is also regular.

Proof.

Let L be regular.

Then there exists a DFA $M = (Q, \Sigma, \delta, q_0, F)$ with $L(M) = L$.

Then $N = (Q, \Sigma, \delta, q_0, Q \setminus F)$ is a DFA with $L(N) = \overline{L}$.

Here it is important that for every input word w:

- There is precisely one path starting at q_0 labelled with w.
- There is precisely one state q with $q_0 \xrightarrow{w}\!\!\twoheadrightarrow q$. Thus

  $$w \in L \iff w \in L(M) \iff q \in F$$

□

Regular Languages: Complement

Theorem

If L is a regular language, then $\overline{L}$ is also regular.

Proof.

Let L be regular.

Then there exists a DFA $M = (Q, \Sigma, \delta, q_0, F)$ with $L(M) = L$.

Then $N = (Q, \Sigma, \delta, q_0, Q \setminus F)$ is a DFA with $L(N) = \overline{L}$.

Here it is important that for every input word w:

- There is precisely one path starting at q_0 labelled with w.
- There is precisely one state q with $q_0 \xrightarrow{w}\!\!\!\!\twoheadrightarrow q$. Thus

$$w \in L \iff w \in L(M) \iff q \in F$$
$$w \in \overline{L} \iff w \in \overline{L(M)} \iff q \in (Q \setminus F) \iff w \in L(N)$$

□

Regular Languages: Union

Theorem

If L_1 and L_2 are regular, then $L_1 \cup L_2$ is regular.

Regular Languages: Union

Theorem

If L_1 and L_2 are regular, then $L_1 \cup L_2$ is regular.

Construction (Product)

There exists a DFAs

$$M_1 = (Q_1, \Sigma, \delta_1, q_{0,1}, F_1) \qquad M_2 = (Q_2, \Sigma, \delta_2, q_{0,2}, F_2)$$

such that $L(M_1) = L_1$ and $L(M_2) = L_2$.

Regular Languages: Union

Theorem

If L_1 and L_2 are regular, then $L_1 \cup L_2$ is regular.

Construction (Product)

There exists a DFAs

$$M_1 = (Q_1, \Sigma, \delta_1, q_{0,1}, F_1) \qquad M_2 = (Q_2, \Sigma, \delta_2, q_{0,2}, F_2)$$

such that $L(M_1) = L_1$ and $L(M_2) = L_2$.

Idea: We run M_1 and M_2 in parallel.

Regular Languages: Union

Theorem

If L_1 and L_2 are regular, then $L_1 \cup L_2$ is regular.

Construction (Product)

There exists a DFAs

$$M_1 = (Q_1, \Sigma, \delta_1, q_{0,1}, F_1) \qquad M_2 = (Q_2, \Sigma, \delta_2, q_{0,2}, F_2)$$

such that $L(M_1) = L_1$ and $L(M_2) = L_2$.

Idea: We run M_1 and M_2 in parallel.

We define a DFA $N = (Q, \Sigma, \delta, q_0, F)$ where

Regular Languages: Union

Theorem

If L_1 and L_2 are regular, then $L_1 \cup L_2$ is regular.

Construction (Product)

There exists a DFAs

$$M_1 = (Q_1, \Sigma, \delta_1, q_{0,1}, F_1) \qquad M_2 = (Q_2, \Sigma, \delta_2, q_{0,2}, F_2)$$

such that $L(M_1) = L_1$ and $L(M_2) = L_2$.

Idea: We run M_1 and M_2 in parallel.

We define a DFA $N = (Q, \Sigma, \delta, q_0, F)$ where

- $Q =$

Regular Languages: Union

Theorem

If L_1 and L_2 are regular, then $L_1 \cup L_2$ is regular.

Construction (Product)

There exists a DFAs

$$M_1 = (Q_1, \Sigma, \delta_1, q_{0,1}, F_1) \qquad M_2 = (Q_2, \Sigma, \delta_2, q_{0,2}, F_2)$$

such that $L(M_1) = L_1$ and $L(M_2) = L_2$.

Idea: We run M_1 and M_2 in parallel.

We define a DFA $N = (Q, \Sigma, \delta, q_0, F)$ where

- $Q = Q_1 \times Q_2 = \{ (q_1, q_2) \mid q_1 \in Q_1,\ q_2 \in Q_2 \}$

Regular Languages: Union

Theorem

If L_1 and L_2 are regular, then $L_1 \cup L_2$ is regular.

Construction (Product)

There exists a DFAs

$$M_1 = (Q_1, \Sigma, \delta_1, q_{0,1}, F_1) \qquad M_2 = (Q_2, \Sigma, \delta_2, q_{0,2}, F_2)$$

such that $L(M_1) = L_1$ and $L(M_2) = L_2$.

Idea: We run M_1 and M_2 in parallel.

We define a DFA $N = (Q, \Sigma, \delta, q_0, F)$ where

- $Q = Q_1 \times Q_2 = \{ (q_1, q_2) \mid q_1 \in Q_1,\ q_2 \in Q_2 \}$
- $\delta((q_1, q_2), a) =$

Regular Languages: Union

Theorem

If L_1 and L_2 are regular, then $L_1 \cup L_2$ is regular.

Construction (Product)

There exists a DFAs

$$M_1 = (Q_1, \Sigma, \delta_1, q_{0,1}, F_1) \qquad M_2 = (Q_2, \Sigma, \delta_2, q_{0,2}, F_2)$$

such that $L(M_1) = L_1$ and $L(M_2) = L_2$.

Idea: We run M_1 and M_2 in parallel.

We define a DFA $N = (Q, \Sigma, \delta, q_0, F)$ where

- $Q = Q_1 \times Q_2 = \{ (q_1, q_2) \mid q_1 \in Q_1,\ q_2 \in Q_2 \}$
- $\delta((q_1, q_2), a) = (\delta_1(q_1, a),\ \delta_2(q_2, a))$

Regular Languages: Union

Theorem

If L_1 and L_2 are regular, then $L_1 \cup L_2$ is regular.

Construction (Product)

There exists a DFAs

$$M_1 = (Q_1, \Sigma, \delta_1, q_{0,1}, F_1) \qquad M_2 = (Q_2, \Sigma, \delta_2, q_{0,2}, F_2)$$

such that $L(M_1) = L_1$ and $L(M_2) = L_2$.

Idea: We run M_1 and M_2 in parallel.

We define a DFA $N = (Q, \Sigma, \delta, q_0, F)$ where

- $Q = Q_1 \times Q_2 = \{ (q_1, q_2) \mid q_1 \in Q_1,\ q_2 \in Q_2 \}$
- $\delta((q_1, q_2), a) = (\delta_1(q_1, a),\ \delta_2(q_2, a))$
- $q_0 =$

Regular Languages: Union

Theorem

If L_1 and L_2 are regular, then $L_1 \cup L_2$ is regular.

Construction (Product)

There exists a DFAs

$$M_1 = (Q_1, \Sigma, \delta_1, q_{0,1}, F_1) \qquad M_2 = (Q_2, \Sigma, \delta_2, q_{0,2}, F_2)$$

such that $L(M_1) = L_1$ and $L(M_2) = L_2$.

Idea: We run M_1 and M_2 in parallel.

We define a DFA $N = (Q, \Sigma, \delta, q_0, F)$ where

- $Q = Q_1 \times Q_2 = \{ (q_1, q_2) \mid q_1 \in Q_1, \ q_2 \in Q_2 \}$
- $\delta((q_1, q_2), a) = (\delta_1(q_1, a), \ \delta_2(q_2, a))$
- $q_0 = (q_{0,1}, \ q_{0,2})$

Regular Languages: Union

Theorem

If L_1 and L_2 are regular, then $L_1 \cup L_2$ is regular.

Construction (Product)

There exists a DFAs

$$M_1 = (Q_1, \Sigma, \delta_1, q_{0,1}, F_1) \qquad M_2 = (Q_2, \Sigma, \delta_2, q_{0,2}, F_2)$$

such that $L(M_1) = L_1$ and $L(M_2) = L_2$.

Idea: We run M_1 and M_2 in parallel.

We define a DFA $N = (Q, \Sigma, \delta, q_0, F)$ where

- $Q = Q_1 \times Q_2 = \{\, (q_1, q_2) \mid q_1 \in Q_1,\ q_2 \in Q_2 \,\}$
- $\delta((q_1, q_2), a) = (\delta_1(q_1, a),\ \delta_2(q_2, a))$
- $q_0 = (q_{0,1},\ q_{0,2})$
- $F =$

Regular Languages: Union

Theorem

If L_1 and L_2 are regular, then $L_1 \cup L_2$ is regular.

Construction (Product)

There exists a DFAs

$$M_1 = (Q_1, \Sigma, \delta_1, q_{0,1}, F_1) \qquad M_2 = (Q_2, \Sigma, \delta_2, q_{0,2}, F_2)$$

such that $L(M_1) = L_1$ and $L(M_2) = L_2$.

Idea: We run M_1 and M_2 in parallel.

We define a DFA $N = (Q, \Sigma, \delta, q_0, F)$ where

- $Q = Q_1 \times Q_2 = \{ (q_1, q_2) \mid q_1 \in Q_1,\ q_2 \in Q_2 \}$
- $\delta((q_1, q_2), a) = (\delta_1(q_1, a),\ \delta_2(q_2, a))$
- $q_0 = (q_{0,1},\ q_{0,2})$
- $F = \{ (q_1, q_2) \in Q \mid q_1 \in F_1 \text{ or } q_2 \in F_2 \}$

Regular Languages: Union

Theorem

If L_1 and L_2 are regular, then $L_1 \cup L_2$ is regular.

Construction (Product)

There exists a DFAs

$$M_1 = (Q_1, \Sigma, \delta_1, q_{0,1}, F_1) \qquad M_2 = (Q_2, \Sigma, \delta_2, q_{0,2}, F_2)$$

such that $L(M_1) = L_1$ and $L(M_2) = L_2$.

Idea: We run M_1 and M_2 in parallel.

We define a DFA $N = (Q, \Sigma, \delta, q_0, F)$ where

- $Q = Q_1 \times Q_2 = \{ (q_1, q_2) \mid q_1 \in Q_1,\ q_2 \in Q_2 \}$
- $\delta((q_1, q_2), a) = (\delta_1(q_1, a),\ \delta_2(q_2, a))$
- $q_0 = (q_{0,1},\ q_{0,2})$
- $F = \{ (q_1, q_2) \in Q \mid q_1 \in F_1 \text{ or } q_2 \in F_2 \}$

Then it follows that $L(N) = L(M_1) \cup L(M_2) = L_1 \cup L_2$.

Regular Languages: Intersection, Difference

Question

Change the product construction to show that

- $L_1 \cap L_2$ is regular, and
- $L_1 \setminus L_2$ is regular ?

Regular Languages: Intersection, Difference

Question

Change the product construction to show that

- $L_1 \cap L_2$ is regular, and
- $L_1 \setminus L_2$ is regular ?

Answer: it suffices to change the definition of the final states

- for $L_1 \cup L_2$: $F = \{ (q_1, q_2) \in Q \mid q_1 \in F_1$ or $q_2 \in F_2 \}$

Regular Languages: Intersection, Difference

Question

Change the product construction to show that

- $L_1 \cap L_2$ is regular, and
- $L_1 \setminus L_2$ is regular ?

Answer: it suffices to change the definition of the final states

- for $L_1 \cup L_2$: $F = \{ (q_1, q_2) \in Q \mid q_1 \in F_1$ or $q_2 \in F_2 \}$
- for $L_1 \cap L_2$: $F =$

Regular Languages: Intersection, Difference

Question

Change the product construction to show that

- $L_1 \cap L_2$ is regular, and
- $L_1 \setminus L_2$ is regular ?

Answer: it suffices to change the definition of the final states

- for $L_1 \cup L_2$: $F = \{ (q_1, q_2) \in Q \mid q_1 \in F_1$ or $q_2 \in F_2 \}$
- for $L_1 \cap L_2$: $F = \{ (q_1, q_2) \in Q \mid q_1 \in F_1$ and $q_2 \in F_2 \}$

Regular Languages: Intersection, Difference

Question

Change the product construction to show that

- $L_1 \cap L_2$ is regular, and
- $L_1 \setminus L_2$ is regular ?

Answer: it suffices to change the definition of the final states

- for $L_1 \cup L_2$: $F = \{ (q_1, q_2) \in Q \mid q_1 \in F_1 \text{ or } q_2 \in F_2 \}$
- for $L_1 \cap L_2$: $F = \{ (q_1, q_2) \in Q \mid q_1 \in F_1 \text{ and } q_2 \in F_2 \}$
- for $L_1 \setminus L_2$: $F =$

Regular Languages: Intersection, Difference

Question

Change the product construction to show that

- $L_1 \cap L_2$ is regular, and
- $L_1 \setminus L_2$ is regular ?

Answer: it suffices to change the definition of the final states

- for $L_1 \cup L_2$: $F = \{ (q_1, q_2) \in Q \mid q_1 \in F_1$ or $q_2 \in F_2 \}$
- for $L_1 \cap L_2$: $F = \{ (q_1, q_2) \in Q \mid q_1 \in F_1$ and $q_2 \in F_2 \}$
- for $L_1 \setminus L_2$: $F = \{ (q_1, q_2) \in Q \mid q_1 \in F_1$ and $q_2 \notin F_2 \}$

Regular Languages: Intersection, Difference

Question

Change the product construction to show that

- $L_1 \cap L_2$ is regular, and
- $L_1 \setminus L_2$ is regular ?

Answer: it suffices to change the definition of the final states

- for $L_1 \cup L_2$: $F = \{ (q_1, q_2) \in Q \mid q_1 \in F_1 \text{ or } q_2 \in F_2 \}$
- for $L_1 \cap L_2$: $F = \{ (q_1, q_2) \in Q \mid q_1 \in F_1 \text{ and } q_2 \in F_2 \}$
- for $L_1 \setminus L_2$: $F = \{ (q_1, q_2) \in Q \mid q_1 \in F_1 \text{ and } q_2 \notin F_2 \}$

Theorem

If L_1 and L_2 are regular, then $L_1 \cap L_2$ is regular.

Regular Languages: Intersection, Difference

Question

Change the product construction to show that

- $L_1 \cap L_2$ is regular, and
- $L_1 \setminus L_2$ is regular ?

Answer: it suffices to change the definition of the final states

- for $L_1 \cup L_2$: $F = \{ (q_1, q_2) \in Q \mid q_1 \in F_1 \text{ or } q_2 \in F_2 \}$
- for $L_1 \cap L_2$: $F = \{ (q_1, q_2) \in Q \mid q_1 \in F_1 \text{ and } q_2 \in F_2 \}$
- for $L_1 \setminus L_2$: $F = \{ (q_1, q_2) \in Q \mid q_1 \in F_1 \text{ and } q_2 \notin F_2 \}$

Theorem

If L_1 and L_2 are regular, then $L_1 \cap L_2$ is regular.

Theorem

If L_1 and L_2 are regular, then $L_1 \setminus L_2$ is regular.

Exercise

Question

Is the following language regular?

$$\{\, a^n b^n \mid n \geq 0 \,\}$$

Exercise

Question

Is the following language regular?

$$\{\, a^n b^n \mid n \geq 0 \,\}$$

This language is not regular!

Exercise

Question

Is the following language regular?

$$\{ a^n b^n \mid n \geq 0 \}$$

This language is not regular!

Intuition: a DFA has only a finite memory (the states).

We will later prove this using the **pumping lemma**.

Finite Languages are Regular

Theorem

Every **finite** language L regular.

Finite Languages are Regular

Theorem

Every **finite** language L regular.

Construction

Let N be the length of the longest word in L.

Finite Languages are Regular

Theorem

Every **finite** language L regular.

Construction

Let N be the length of the longest word in L.

Define the DFA $M = (Q, \Sigma, \delta, q_\lambda, F)$ by

Finite Languages are Regular

Theorem

Every **finite** language L regular.

Construction

Let N be the length of the longest word in L.

Define the DFA $M = (Q, \Sigma, \delta, q_\lambda, F)$ by

- $Q = \{ q_w \mid w \in \Sigma^*, \; |w| \leq N \} \cup \{ q_\perp \}$

Finite Languages are Regular

Theorem

Every **finite** language L regular.

Construction

Let N be the length of the longest word in L.

Define the DFA $M = (Q, \Sigma, \delta, q_\lambda, F)$ by

- $Q = \{ q_w \mid w \in \Sigma^*, \ |w| \leq N \} \cup \{ q_\perp \}$
- $F = \{ q_w \mid w \in L \}$

Finite Languages are Regular

Theorem

Every **finite** language L regular.

Construction

Let N be the length of the longest word in L.

Define the DFA $M = (Q, \Sigma, \delta, q_\lambda, F)$ by

- $Q = \{\, q_w \mid w \in \Sigma^*,\ |w| \leq N \,\} \cup \{\, q_\perp \,\}$
- $F = \{\, q_w \mid w \in L \}$
- the transition function δ is defined by

$$\delta(q_w, a) = \begin{cases} q_{wa} & \text{if } |wa| \leq N, \\ q_\perp & \text{if } |wa| > N \end{cases} \qquad \delta(q_\perp, a) = q_\perp$$

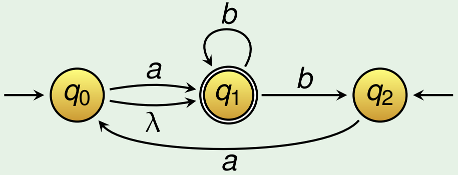
  for every $w \in \Sigma^*$ with $|w| \leq N$ and $a \in \Sigma$

Nondeterministic Finite Automata

Nondeterministic Finite Automata

NFAs are defined like DFAs, except that NFAs allow for:

- **Multiple starting states.**
- **Any number of outgoing arrows** with the same label.
- **Empty steps**: arrows labelled λ (do not consume input).



Note that:

- both q_0 and q_2 are starting states
- the state q_1 has two outgoing arrows with label b
- there is an empty step from q_0 to q_1

Nondeterministic Finite Automata

A **nondeterministic finite automaton**, short **NFA**, consists of:

- a finite set Q of states
- a finite input alphabet Σ
- a transition function $\delta : Q \times (\Sigma \cup \{\lambda\}) \to 2^Q$
- a set $S \subseteq Q$ of starting states
- a set $F \subseteq Q$ of final states

Here 2^Q is the set of all subsets of Q: $2^Q = \{ X \mid X \subseteq Q \}$.

The NFA on the preceding slide is $M = (Q, \Sigma, \delta, S, F)$ where

$$Q = \{ q_0, q_1, q_2 \}$$
$$\Sigma = \{ a, b \}$$
$$S = \{ q_0, q_2 \}$$
$$F = \{ q_1 \}$$

δ	q_0	q_1	q_2
a	$\{ q_1 \}$	$\varnothing$	$\{ q_0 \}$
b	$\varnothing$	$\{ q_1, q_2 \}$	$\varnothing$
λ	$\{ q_1 \}$	$\varnothing$	$\varnothing$

NFAs Reading Words

Let $M = (Q, \Sigma, \delta, S, F)$ be a NFA.

The **step relation** $\vdash$ of M is defined on configurations by

$$(q, \alpha w) \vdash (q', w) \quad \text{if } q' \in \delta(q, \alpha) \text{ with } \alpha \in \Sigma \cup \{\lambda\}$$

NFAs Reading Words

Let $M = (Q, \Sigma, \delta, S, F)$ be a NFA.

The **step relation** $\vdash$ of M is defined on configurations by

$$(q, \alpha w) \vdash (q', w) \quad \text{if } q' \in \delta(q, \alpha) \text{ with } \alpha \in \Sigma \cup \{\lambda\}$$

Note that if $\alpha = \lambda$, then

- the state changes (q to q'), but
- the input word stays the same ($\lambda w = w$).

NFAs Reading Words

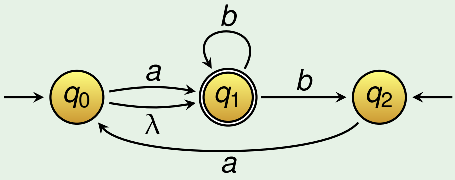
Let $M = (Q, \Sigma, \delta, S, F)$ be a NFA.

The **step relation** $\vdash$ of M is defined on configurations by

$$(q, \alpha w) \vdash (q', w) \quad \text{if } q' \in \delta(q, \alpha) \text{ with } \alpha \in \Sigma \cup \{\lambda\}$$

Note that if $\alpha = \lambda$, then

- the state changes (q to q'), but
- the input word stays the same ($\lambda w = w$).



$$(q_0, abbab) \vdash (q_1, bbab) \vdash (q_1, bab) \vdash (q_2, ab)$$
$$\vdash (q_0, b) \vdash (q_1, b) \vdash (q_1, \lambda)$$

Paths in NFAs

Let $M = (Q, \Sigma, \delta, S, F)$ be a NFA.

For a word w, we write

$$q \overset{w}{\twoheadrightarrow} q'$$

if $w = \alpha_1 \cdots \alpha_n$ for some $\alpha_1, \ldots, \alpha_n \in (\Sigma \cup \{\lambda\})$ and there are states $q_1, \ldots, q_{n-1}$ such that

$$q \overset{\alpha_1}{\to} q_1 \qquad q_1 \overset{\alpha_2}{\to} q_2 \qquad q_2 \overset{\alpha_3}{\to} q_3 \qquad \ldots \qquad q_{n-1} \overset{\alpha_n}{\to} q'$$

Paths in NFAs

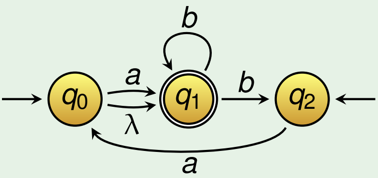
Let $M = (Q, \Sigma, \delta, S, F)$ be a NFA.

For a word w, we write

$$q \xrightarrow{w}\!\!\!\twoheadrightarrow q'$$

if $w = \alpha_1 \cdots \alpha_n$ for some $\alpha_1, \ldots, \alpha_n \in (\Sigma \cup \{\lambda\})$ and there are states $q_1, \ldots, q_{n-1}$ such that

$$q \xrightarrow{\alpha_1} q_1 \qquad q_1 \xrightarrow{\alpha_2} q_2 \qquad q_2 \xrightarrow{\alpha_3} q_3 \qquad \ldots \qquad q_{n-1} \xrightarrow{\alpha_n} q'$$



$$q_0 \overset{\lambda}{\twoheadrightarrow} q_0 \qquad q_1 \overset{ba}{\twoheadrightarrow} q_0$$

$$q_0 \overset{\lambda}{\twoheadrightarrow} q_1 \qquad q_1 \overset{ba}{\twoheadrightarrow} q_1$$

$$q_1 \overset{b}{\twoheadrightarrow} q_1 \qquad q_0 \overset{abbab}{\twoheadrightarrow} q_1$$

$$q_1 \overset{b}{\twoheadrightarrow} q_2 \qquad q_0 \overset{abbab}{\twoheadrightarrow} q_2$$

Paths in NFAs

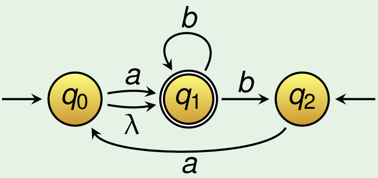
Let $M = (Q, \Sigma, \delta, S, F)$ be a NFA.

For a word w, we write

$$q \xrightarrow{w}\!\!\!\!\rightarrow q'$$

if $w = \alpha_1 \cdots \alpha_n$ for some $\alpha_1, \ldots, \alpha_n \in (\Sigma \cup \{\lambda\})$ and there are states $q_1, \ldots, q_{n-1}$ such that

$$q \stackrel{\alpha_1}{\rightarrow} q_1 \qquad q_1 \stackrel{\alpha_2}{\rightarrow} q_2 \qquad q_2 \stackrel{\alpha_3}{\rightarrow} q_3 \qquad \ldots \qquad q_{n-1} \stackrel{\alpha_n}{\rightarrow} q'$$



$$q_0 \xrightarrow{\lambda}\!\!\!\!\rightarrow q_0 \qquad q_1 \xrightarrow{ba}\!\!\!\!\rightarrow q_0$$

$$q_0 \xrightarrow{\lambda}\!\!\!\!\rightarrow q_1 \qquad q_1 \xrightarrow{ba}\!\!\!\!\rightarrow q_1$$

$$q_1 \xrightarrow{b}\!\!\!\!\rightarrow q_1 \qquad q_0 \xrightarrow{abbab}\!\!\!\!\rightarrow q_1$$

$$q_1 \xrightarrow{b}\!\!\!\!\rightarrow q_2 \qquad q_0 \xrightarrow{abbab}\!\!\!\!\rightarrow q_2$$

Theorem: $q \xrightarrow{w}\!\!\!\!\rightarrow q' \iff (q, w) \vdash^* (q', \lambda)$.

NFAs Accepting Languages

The **language accepted by** NFA $M = (Q, \Sigma, \delta, S, F)$ is

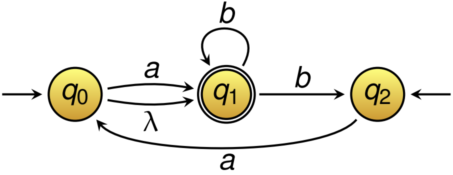
$$L(M) = \{\, w \in \Sigma^* \mid (q_0, w) \vdash^* (q, \lambda) \text{ with } q_0 \in S,\ q \in F \,\}$$

$$= \{\, w \in \Sigma^* \mid q_0 \xrightarrow{w}\!\!\!\!\rightarrow q \text{ with } q_0 \in S,\ q \in F \,\}$$

NFAs Accepting Languages

The **language accepted by** NFA $M = (Q, \Sigma, \delta, S, F)$ is

$$L(M) = \{ w \in \Sigma^* \mid (q_0, w) \vdash^* (q, \lambda) \text{ with } q_0 \in S,\ q \in F \}$$
$$= \{ w \in \Sigma^* \mid q_0 \xrightarrow{w}\!\!\!\!\rightarrow q \text{ with } q_0 \in S,\ q \in F \}$$

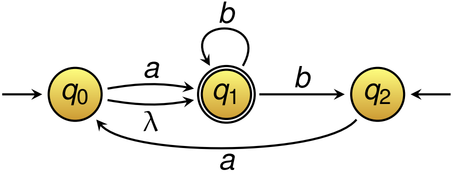


Paths are not unique!

NFAs Accepting Languages

The **language accepted by** NFA $M = (Q, \Sigma, \delta, S, F)$ is

$$L(M) = \{\, w \in \Sigma^* \mid (q_0, w) \vdash^* (q, \lambda) \text{ with } q_0 \in S,\ q \in F \,\}$$
$$= \{\, w \in \Sigma^* \mid q_0 \xrightarrow{w}\!\!\!\!\rightarrow q \text{ with } q_0 \in S,\ q \in F \,\}$$

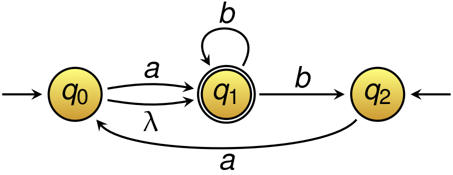


Paths are not unique! Paths for input word *ab*:

NFAs Accepting Languages

The **language accepted by** NFA $M = (Q, \Sigma, \delta, S, F)$ is

$$L(M) = \{\, w \in \Sigma^* \mid (q_0, w) \vdash^* (q, \lambda) \text{ with } q_0 \in S,\ q \in F \,\}$$
$$= \{\, w \in \Sigma^* \mid q_0 \xrightarrow{w}\!\!\!\!\rightarrow q \text{ with } q_0 \in S,\ q \in F \,\}$$



Paths are not unique! Paths for input word *ab*:

$(q_0, ab) \vdash (q_1, b) \vdash (q_1, \lambda)$

$(q_0, ab) \vdash (q_1, b) \vdash (q_2, \lambda)$

$(q_2, ab) \vdash (q_0, b) \vdash (q_1, b) \vdash (q_1, \lambda)$

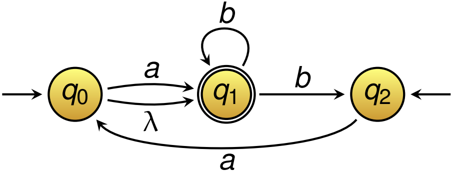
$(q_2, ab) \vdash (q_0, b) \vdash (q_1, b) \vdash (q_2, \lambda)$

NFAs Accepting Languages

The **language accepted by** NFA $M = (Q, \Sigma, \delta, S, F)$ is

$$L(M) = \{\, w \in \Sigma^* \mid (q_0, w) \vdash^* (q, \lambda) \text{ with } q_0 \in S,\ q \in F \,\}$$

$$= \{\, w \in \Sigma^* \mid q_0 \xrightarrow{w}\!\!\!\rightarrow q \text{ with } q_0 \in S,\ q \in F \,\}$$



Paths are not unique! Paths for input word *ab*:

$(q_0, ab) \vdash (q_1, b) \vdash (q_1, \lambda)$ (ends in accepting state)

$(q_0, ab) \vdash (q_1, b) \vdash (q_2, \lambda)$

$(q_2, ab) \vdash (q_0, b) \vdash (q_1, b) \vdash (q_1, \lambda)$ (ends in accepting state)

$(q_2, ab) \vdash (q_0, b) \vdash (q_1, b) \vdash (q_2, \lambda)$

One accepting path suffices! So *ab* is accepted.

NFAs with a Single Starting State

For every NFA M there is an NFA N such that $L(M) = L(N)$ and N has a **single starting state**.

NFAs with a Single Starting State

For every NFA M there is an NFA N such that $L(M) = L(N)$ and N has a **single starting state**.

Construction

Let $N = (Q, \Sigma, \delta, S, F)$ be an NFA.

NFAs with a Single Starting State

For every NFA M there is an NFA N such that $L(M) = L(N)$ and N has a **single starting state**.

Construction

Let $N = (Q, \Sigma, \delta, S, F)$ be an NFA.

Define M the be obtained from N as follows

NFAs with a Single Starting State

For every NFA M there is an NFA N such that $L(M) = L(N)$ and N has a **single starting state**.

Construction

Let $N = (Q, \Sigma, \delta, S, F)$ be an NFA.

Define M the be obtained from N as follows

- add a fresh state q_0,

NFAs with a Single Starting State

For every NFA M there is an NFA N such that $L(M) = L(N)$ and N has a **single starting state**.

Construction

Let $N = (Q, \Sigma, \delta, S, F)$ be an NFA.

Define M the be obtained from N as follows

- add a fresh state q_0,
- add transitions $q_0 \xrightarrow{\lambda} q$ for every $q \in S$, and

NFAs with a Single Starting State

For every NFA M there is an NFA N such that $L(M) = L(N)$ and N has a **single starting state**.

Construction

Let $N = (Q, \Sigma, \delta, S, F)$ be an NFA.

Define M the be obtained from N as follows

- add a fresh state q_0,
- add transitions $q_0 \xrightarrow{\lambda} q$ for every $q \in S$, and
- make q_0 the only starting state of M.

NFAs with a Single Starting State

For every NFA M there is an NFA N such that $L(M) = L(N)$ and N has a **single starting state**.

Construction

Let $N = (Q, \Sigma, \delta, S, F)$ be an NFA.

Define M the be obtained from N as follows

- add a fresh state q_0,
- add transitions $q_0 \xrightarrow{\lambda} q$ for every $q \in S$, and
- make q_0 the only starting state of M.

Then M has a single starting state and $L(N) = L(M)$.

NFAs with a Single Starting State

For every NFA M there is an NFA N such that $L(M) = L(N)$ and N has a **single starting state**.

Construction

Let $N = (Q, \Sigma, \delta, S, F)$ be an NFA.

Define M the be obtained from N as follows

- add a fresh state q_0,
- add transitions $q_0 \xrightarrow{\lambda} q$ for every $q \in S$, and
- make q_0 the only starting state of M.

Then M has a single starting state and $L(N) = L(M)$.

Convention

We denote NFAs $(Q, \Sigma, \delta, S, F)$ with a single starting state $S = \{ q_0 \}$ by $(Q, \Sigma, \delta, q_0, F)$.

DFAs and NFAs are Equally Expressive

Theorem

A language L is accepted by a NFA $\iff$ L is regular.

DFAs and NFAs are Equally Expressive

Theorem

A language L is accepted by a NFA $\iff$ L is regular.

Construction (Powerset)

Let $M = (Q, \Sigma, \delta, S, F)$ be a NFA.

DFAs and NFAs are Equally Expressive

Theorem

A language L is accepted by a NFA $\iff$ L is regular.

Construction (Powerset)

Let $M = (Q, \Sigma, \delta, S, F)$ be a NFA.

Idea: state of DFA = set of all states the NFA can be in

DFAs and NFAs are Equally Expressive

Theorem

A language L is accepted by a NFA $\iff$ L is regular.

Construction (Powerset)

Let $M = (Q, \Sigma, \delta, S, F)$ be a NFA.

Idea: state of DFA = set of all states the NFA can be in

We construct a DFA $N = (Q', \Sigma, \delta', q_0', F')$ where

$$
\begin{aligned}
Q' &= \\
\delta'(X, a) &= \\
q_0' &= \\
F' &=
\end{aligned}
$$

DFAs and NFAs are Equally Expressive

Theorem

A language L is accepted by a NFA $\iff$ L is regular.

Construction (Powerset)

Let $M = (Q, \Sigma, \delta, S, F)$ be a NFA.

Idea: state of DFA = set of all states the NFA can be in

We construct a DFA $N = (Q', \Sigma, \delta', q_0', F')$ where

$$
\begin{aligned}
Q' &= 2^Q = \{ X \mid X \subseteq Q \} \\
\delta'(X, a) &= \\
q_0' &= \\
F' &=
\end{aligned}
$$

DFAs and NFAs are Equally Expressive

Theorem

A language L is accepted by a NFA $\iff$ L is regular.

Construction (Powerset)

Let $M = (Q, \Sigma, \delta, S, F)$ be a NFA.

Idea: state of DFA = set of all states the NFA can be in

We construct a DFA $N = (Q', \Sigma, \delta', q_0', F')$ where

$$
\begin{aligned}
Q' &= 2^Q = \{ X \mid X \subseteq Q \} \\
\delta'(X, a) &= \{ q' \in Q \mid q \xrightarrow{a}\!\!\twoheadrightarrow q' \text{ for some } q \in X \} \\
q_0' &= \\
F' &=
\end{aligned}
$$

DFAs and NFAs are Equally Expressive

Theorem

A language L is accepted by a NFA $\iff$ L is regular.

Construction (Powerset)

Let $M = (Q, \Sigma, \delta, S, F)$ be a NFA.

Idea: state of DFA = set of all states the NFA can be in

We construct a DFA $N = (Q', \Sigma, \delta', q_0', F')$ where

$$
\begin{aligned}
Q' &= 2^Q = \{ X \mid X \subseteq Q \} \\
\delta'(X, a) &= \{ q' \in Q \mid q \xrightarrow{a}\!\!\twoheadrightarrow q' \text{ for some } q \in X \} \\
q_0' &= \{ q' \in Q \mid q \xrightarrow{\lambda}\!\!\twoheadrightarrow q' \text{ for some } q \in S \} \\
F' &=
\end{aligned}
$$

DFAs and NFAs are Equally Expressive

Theorem

A language L is accepted by a NFA $\iff$ L is regular.

Construction (Powerset)

Let $M = (Q, \Sigma, \delta, S, F)$ be a NFA.

Idea: state of DFA = set of all states the NFA can be in

We construct a DFA $N = (Q', \Sigma, \delta', q_0', F')$ where

$$
\begin{aligned}
Q' &= 2^Q = \{\, X \mid X \subseteq Q \,\} \\
\delta'(X, a) &= \{\, q' \in Q \mid q \xrightarrow{a}\!\!\!\!\twoheadrightarrow q' \text{ for some } q \in X \,\} \\
q_0' &= \{\, q' \in Q \mid q \xrightarrow{\lambda}\!\!\!\!\twoheadrightarrow q' \text{ for some } q \in S \,\} \\
F' &= \{\, X \subseteq Q \mid X \cap F \neq \varnothing \,\}
\end{aligned}
$$

DFAs and NFAs are Equally Expressive

Theorem

A language L is accepted by a NFA $\iff$ L is regular.

Construction (Powerset)

Let $M = (Q, \Sigma, \delta, S, F)$ be a NFA.

Idea: state of DFA = set of all states the NFA can be in

We construct a DFA $N = (Q', \Sigma, \delta', q_0', F')$ where

$$\begin{aligned}
Q' &= 2^Q = \{\, X \mid X \subseteq Q \,\} \\
\delta'(X, a) &= \{\, q' \in Q \mid q \xrightarrow{a}\!\!\!\twoheadrightarrow q' \text{ for some } q \in X \,\} \\
q_0' &= \{\, q' \in Q \mid q \xrightarrow{\lambda}\!\!\!\twoheadrightarrow q' \text{ for some } q \in S \,\} \\
F' &= \{\, X \subseteq Q \mid X \cap F \neq \varnothing \,\}
\end{aligned}$$

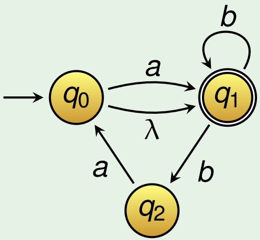
For every $w \in \Sigma^*$ and $X \subseteq Q$ it holds that

$$X \xrightarrow{w}\!\!\!\twoheadrightarrow X' \text{ in } N \quad \iff \quad X' = \{\, q' \mid q \in X,\ q \xrightarrow{w}\!\!\!\twoheadrightarrow q' \text{ in } M \,\}$$

From this property it follows that $L(N) = L(M)$.

Exercise

Given is the following NFA:



Construct a DFA that accepts the same language.

Regular Languages: Reversal

Theorem

If L is regular, then its reverse L^R is regular.

Regular Languages: Reversal

Theorem

If L is regular, then its reverse L^R is regular.

Construction

Let L be a regular language.

Regular Languages: Reversal

Theorem

If L is regular, then its reverse L^R is regular.

Construction

Let L be a regular language.

Then there is an NFA $M = (Q, \Sigma, \delta, S, F)$ with $L(M) = L$.

Regular Languages: Reversal

Theorem

If L is regular, then its reverse L^R is regular.

Construction

Let L be a regular language.

Then there is an NFA $M = (Q, \Sigma, \delta, S, F)$ with $L(M) = L$.

Let N be the NFA obtained from M by

Regular Languages: Reversal

Theorem

If L is regular, then its reverse L^R is regular.

Construction

Let L be a regular language.

Then there is an NFA $M = (Q, \Sigma, \delta, S, F)$ with $L(M) = L$.

Let N be the NFA obtained from M by

- reversing all arrows (transitions),

Regular Languages: Reversal

Theorem

If L is regular, then its reverse L^R is regular.

Construction

Let L be a regular language.

Then there is an NFA $M = (Q, \Sigma, \delta, S, F)$ with $L(M) = L$.

Let N be the NFA obtained from M by

- reversing all arrows (transitions),
- exchanging starting states S and final states F.

Regular Languages: Reversal

Theorem

If L is regular, then its reverse L^R is regular.

Construction

Let L be a regular language.

Then there is an NFA $M = (Q, \Sigma, \delta, S, F)$ with $L(M) = L$.

Let N be the NFA obtained from M by

- reversing all arrows (transitions),
- exchanging starting states S and final states F.

Then we have

$$q \xrightarrow{w}\!\!\!\!\rightarrow q' \text{ in } M \quad \iff \quad q' \xrightarrow{w^R}\!\!\!\!\rightarrow q \text{ in } N$$

Regular Languages: Reversal

Theorem

If L is regular, then its reverse L^R is regular.

Construction

Let L be a regular language.

Then there is an NFA $M = (Q, \Sigma, \delta, S, F)$ with $L(M) = L$.

Let N be the NFA obtained from M by

- reversing all arrows (transitions),
- exchanging starting states S and final states F.

Then we have

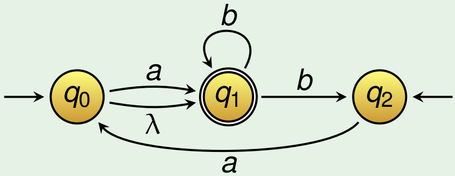
$$q \xrightarrow{w}\!\!\!\!\rightarrow q' \text{ in } M \quad \Longleftrightarrow \quad q' \xrightarrow{w^R}\!\!\!\!\rightarrow q \text{ in } N$$

Since starting and final states are swapped, it follows that

$$w \in L(M) \iff w^R \in L(N)$$

Exercise

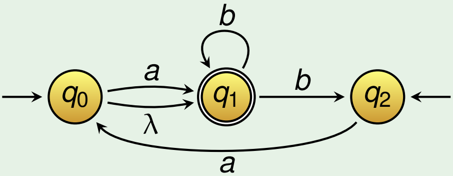
Given is the following NFA:



Construct an NFA that accepts the reverse language:

Exercise

Given is the following NFA:



Construct an NFA that accepts the reverse language: