Automata and Grammars (BIE-AAG)

2. Deterministic and nondeterm. finite automata

Jan Holub

Department of Theoretical Computer Science
Faculty of Information Technology
Czech Technical University in Prague

Deterministic finite automaton

Definition

Deterministic finite automaton is a quintuple $M = (Q, \Sigma, \delta, q_0, F)$, where

- Q is a finite set of states,
- Σ is a finite input alphabet,
- δ is a mapping from $Q \times \Sigma$ to Q, → grafico
- $q_0 \in Q$ is the initial state,
- $F \subseteq Q$ is the set of final states.

We start in 1 state and go to a definite set of STATE

Configuration of a finite automaton

Definition (Configuration of a finite automaton)
Let $M = (Q, \Sigma, \delta, q_0, F)$ be a finite automaton. Pair $(q, w) \in Q \times \Sigma^*$ is called *configuration of finite automaton* M. Configuration (q_0, w) is called *initial configuration* of finite automaton M, configuration (q, ε), where $q \in F$, is called *accepting configuration* of finite automaton M.

Starts with the complete string (w)
and ends with empty string

Move of DFA

Definition (Move of DFA)
Let $M = (Q, \Sigma, \delta, q_0, F)$ be a deterministic finite automaton. Let $\vdash_M$ is a relation over $Q \times \Sigma^*$ (i.e., subset of $(Q \times \Sigma^*) \times (Q \times \Sigma^*)$) such that $(q, w) \vdash_M (p, w')$ iff $w = aw'$ and $\delta(q, a) = p$ for some $a \in \Sigma$, $w \in \Sigma^*$. An element of relation $\vdash_M$ is called a *move* in automaton M.

$\vdash_M^k$ – the k-th power of relation $\vdash_M$

$(\alpha_0, \beta_0) \vdash_M^k (\alpha_k, \beta_k)$ if
$\exists(\alpha_i, \beta_i), 0 < i < k : (\alpha_i, \beta_i) \vdash_M (\alpha_{i+1}, \beta_{i+1}), 0 \leq i < k$

$\vdash_M^+$ – the transitive closure of relation $\vdash_M$
$\vdash_M^*$ – the transitive and reflexive closure of relation $\vdash_M$

$(q, aw') \vdash_M (p, w')$ means $((q, aw'), (p, w')) \in \vdash_M$

$(\alpha_0, \beta_0) \vdash (\alpha_1, \beta_1) \vdash (\alpha_2, \beta_2) \vdash (\alpha_3, \beta_3)$
$= \quad (\alpha_0, \beta_0) \vdash^3 (\alpha_3, \beta_3)$

Language accepted by DFA

Definition (Language accepted by DFA)
We say that string $w \in \Sigma^*$ *is accepted* by a deterministic finite automaton $M = (Q, \Sigma, \delta, q_0, F)$ if $\exists (q_0, w) \vdash_M^* (q, \varepsilon)$ for some $q \in F$.
$L(M) = \{w : w \in \Sigma^*, \exists q \in F : (q_0, w) \vdash^* (q, \varepsilon)\}$ *is the language accepted by DFA* M.
(String $w \in L(M)$ if there exists a sequence of moves from the initial configuration (q_0, w) into an accepting configuration (q, ε).)

if iT Can Process all the STRings

DFA Configuration

Example

Let $M = (\{q_0, q_1, q_2, q_3\}, \{0, 1\}, \delta, q_0, \{q_0\})$ be a DFA, for which the mapping δ is defined as:

$$\delta(q_0, 0) = q_2, \quad \delta(q_1, 0) = q_3, \quad \delta(q_2, 0) = q_0,$$
$$\delta(q_3, 0) = q_1, \quad \delta(q_0, 1) = q_1, \quad \delta(q_1, 1) = q_0,$$
$$\delta(q_2, 1) = q_3, \quad \delta(q_3, 1) = q_2.$$

The mapping δ can be written in the form of a table:

initial state

FINAL STATE

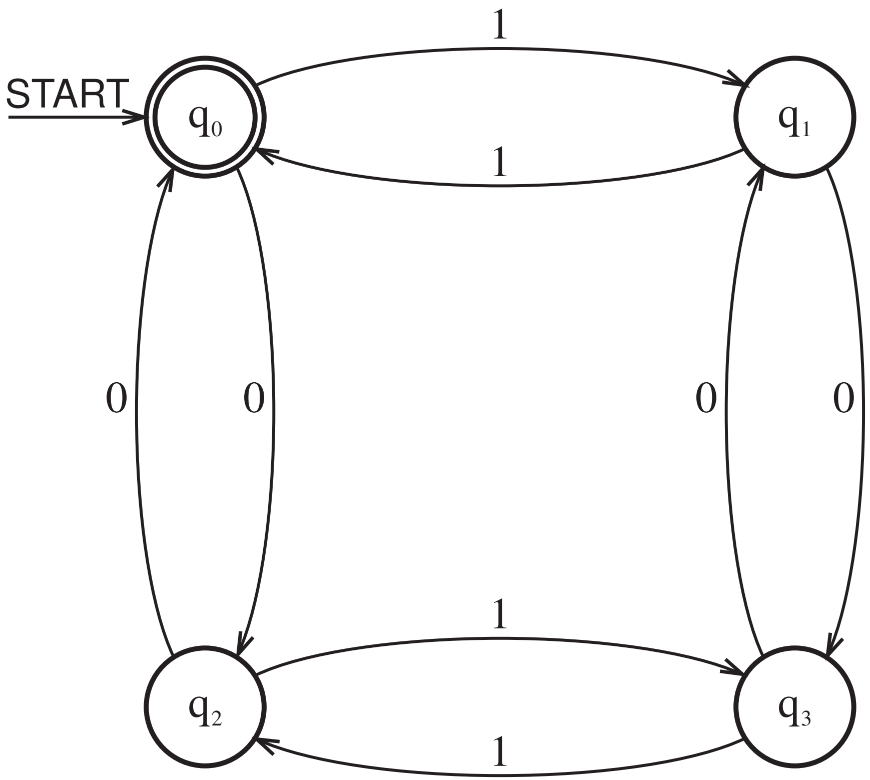
state	input symbol	
δ	0	1
$\rightleftarrows q_0$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

DFA Configuration

Example (continued)

M: $(q_0, 110101) \vdash (q_1, 10101) \vdash (q_0, 0101) \vdash (q_2, 101) \vdash (q_3, 01) \vdash (q_1, 1) \vdash (q_0, \varepsilon)$

M: $(q_0, 11010) \vdash (q_1, 1010) \vdash (q_0, 010) \vdash (q_2, 10) \vdash (q_3, 0) \vdash (q_1, \varepsilon)$



$L(M) = \{x : x \in \{0,1\}^*$ and the number of zeros and the number of ones in x are even$\}$

DFA Configuration

Example

Given DFA $M = (\{q_0, q_1, q_2\}, \{a, b, c\}, \delta, q_0, \{q_0, q_1, q_2\})$ with δ defined as follows:

	δ_M	a	b	c
$\overrightarrow{\leftarrow}$	q_0	q_0	q_1	q_2
$\leftarrow$	q_1		q_1	q_2
$\leftarrow$	q_2			q_2

M: $(q_0, abc) \vdash (q_0, bc) \vdash (q_1, c) \vdash (q_2, \varepsilon)$
M: $(q_0, abac) \vdash (q_0, bac) \vdash (q_1, ac) \vdash$ *fail* as $\delta(q_1, a) = \emptyset$

Deterministic finite automaton

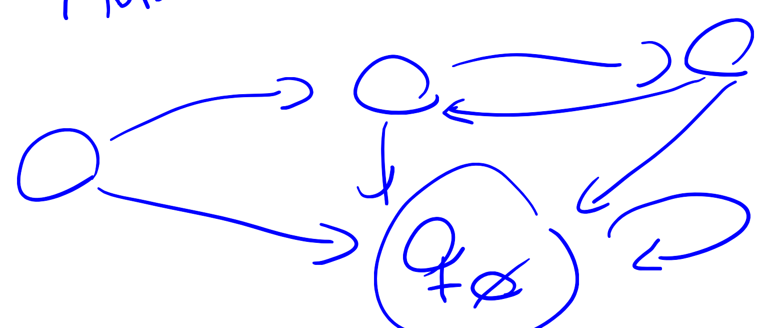
Definition (Total DFA)
DFA $M = (Q, \Sigma, \delta, q_0, F)$ is caled *total* if the mapping δ is defined for all pairs of state $q \in Q$ and symbol $a \in \Sigma$.

Algorithm Completion of DFA to be total.

Input: DFA $M = (Q, \Sigma, \delta, q_0, F)$.
Output: Total DFA $M' = (Q', \Sigma, \delta', q_0, F)$ such that $L(M') = L(M)$.

$Q' \leftarrow Q \cup \{q_\emptyset\}$ $\triangleright$ a new state $q_\emptyset \notin Q$ called "empty"
$\delta'(q, a) \leftarrow \delta(q, a)$, $\forall a \in \Sigma, q \in Q'$, if $\delta(q, a)$ is defined
$\delta'(q, a) \leftarrow q_\emptyset$, $\forall a \in \Sigma, q \in Q'$, if $\delta(q, a)$ is not defined
return M'

Adds An EmPTY State. All Transitions Not defined
go To EmPTy state and Transition from empty state
go To EmPty sTate



Deterministic finite automaton

Example
Given DFA $M = (\{q_0, q_1, q_2\}, \{a, b, c\}, \delta, q_0, \{q_0, q_1, q_2\})$ with δ defined as follows:

	δ_M	a	b	c
$\rightleftarrows$	q_0	q_0	q_1	q_2
$\leftarrow$	q_1		q_1	q_2
$\leftarrow$	q_2			q_2

$\Longrightarrow$

	$\delta_{M'}$	a	b	c
$\rightleftarrows$	q_0	q_0	q_1	q_2
$\leftarrow$	q_1	$q_\emptyset$	q_1	q_2
$\leftarrow$	q_2	$q_\emptyset$	$q_\emptyset$	q_2
	$q_\emptyset$	$q_\emptyset$	$q_\emptyset$	$q_\emptyset$

the resulting total DFA: M'

Nondeterministic finite automaton

Definition

Nondeterministic finite automaton (NFA) M is a quintuple $M = (Q, \Sigma, \delta, q_0, F)$, where

- Q is a finite set of states,
- Σ is a finite input alphabet,
- δ is a mapping from $Q \times \Sigma$ into the set of all subsets Q (denoted by 2^Q),
- $q_0 \in Q$ is the initial state,
- $F \subseteq Q$ is the set of final states.

The maping leads to a set of states instead of a single state

Move of NFA

Definition (Move of NFA)
Let $M = (Q, \Sigma, \delta, q_0, F)$ be a nondeterministic finite automaton. Let $\vdash_M$ is a relation over $Q \times \Sigma^*$ (i.e., subset of $(Q \times \Sigma^*) \times (Q \times \Sigma^*)$) such that $(q, w) \vdash_M (p, w')$ iff $w = aw'$ and $p \in \delta(q, a)$ for some $a \in \Sigma$, $w \in \Sigma^*$. An element of relation $\vdash_M$ is called a *move* in automaton M.

$\vdash_M^k$ – the k-th power of relation $\vdash_M$
$\vdash_M^+$ – the transitive closure of relation $\vdash_M$
$\vdash_M^*$ – the transitive and reflexive closure of relation $\vdash_M$

SAme as deterministic

Language accepted by NFA

Definition (Language accepted by NFA)
We say that string $w \in \Sigma^*$ *is accepted by nondeterministic finite automaton* $M = (Q, \Sigma, \delta, q_0, F)$ if $\exists (q_0, w) \vdash^* (q, \varepsilon)$ for some $q \in F$.
$L(M) = \{w : w \in \Sigma^*, \exists q \in F, (q_0, w) \vdash^* (q, \varepsilon)\}$ is the *language accepted by NFA M.*

difference with deterministic

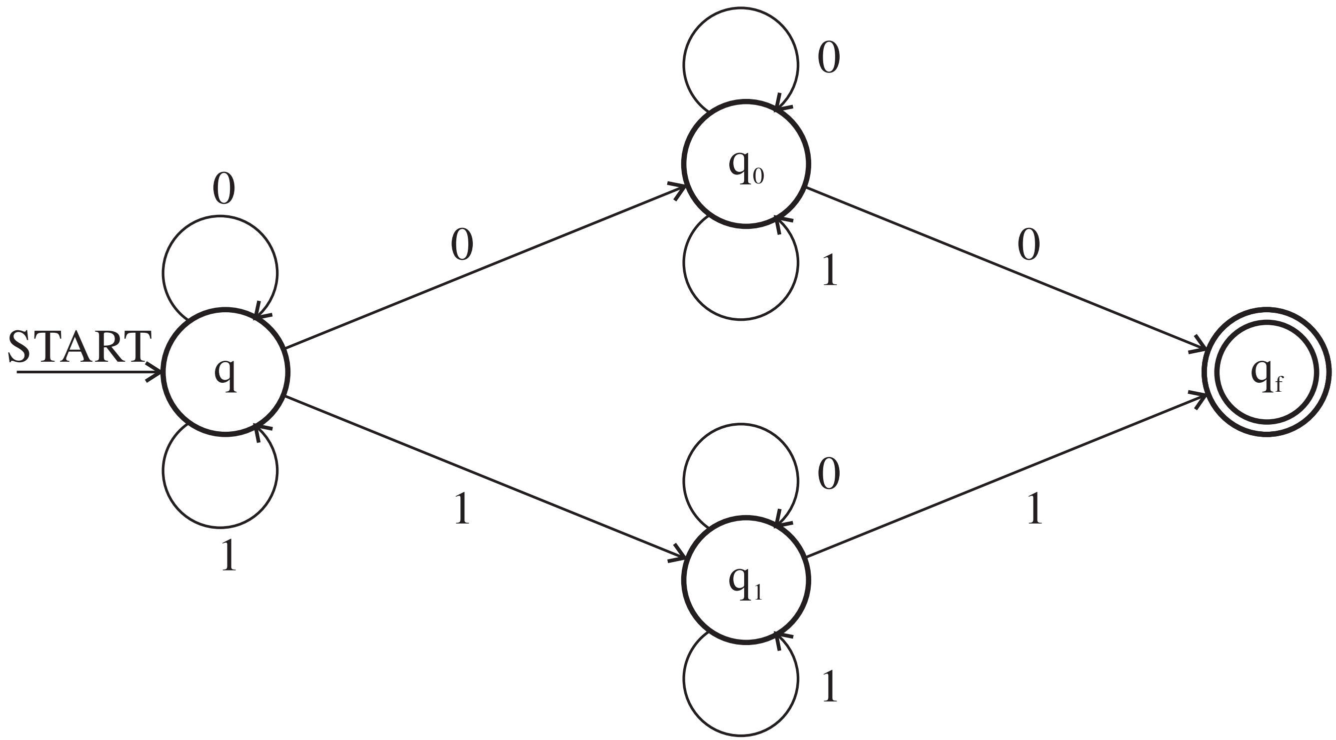
DFA can be transformed to NFA
and its easier to create

NFA configuration

Example

Given NFA $M = (\{q, q_0, q_1, q_f\}, \{0, 1\}, \delta, q, \{q_f\})$, where δ:

$\delta(q, 0) = \{q, q_0\}, \quad \delta(q, 1) = \{q, q_1\}, \quad \delta(q_0, 0) = \{q_0, q_f\},$
$\delta(q_0, 1) = \{q_0\}, \quad \delta(q_1, 0) = \{q_1\}, \quad \delta(q_1, 1) = \{q_1, q_f\}.$

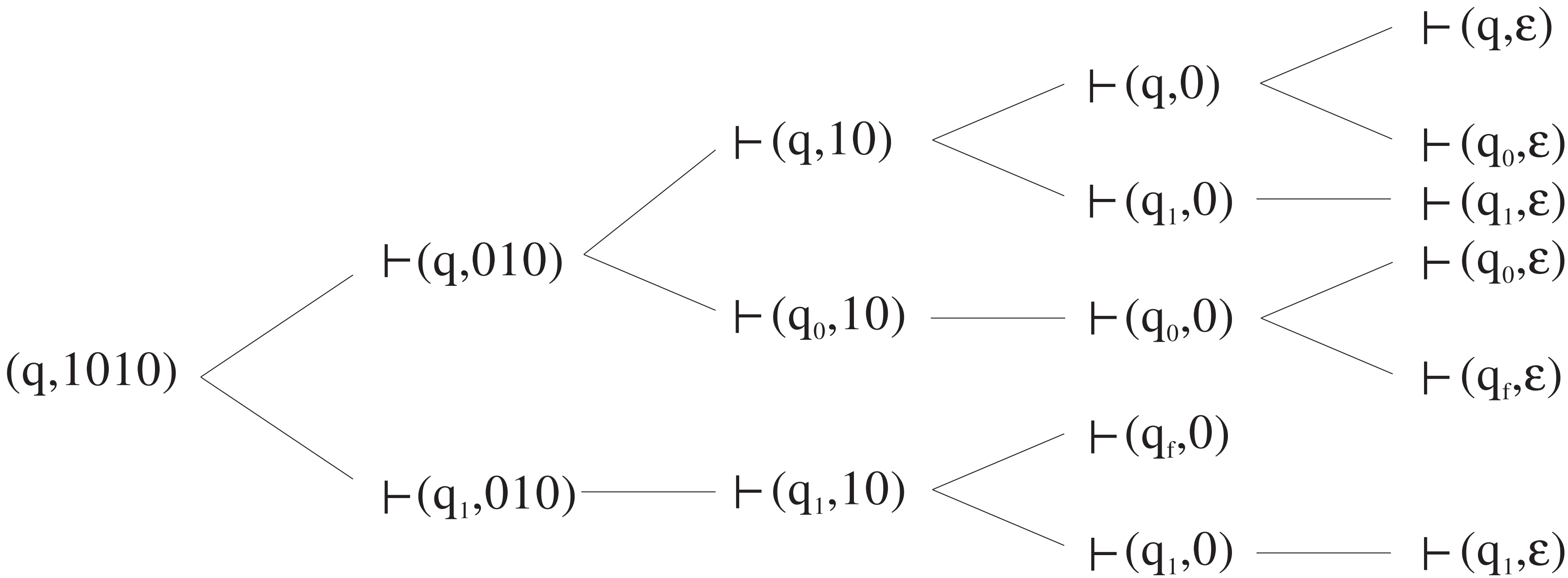


$L(M) = \{w : w \in \{0, 1\}^*$ and w ends with a symbol that has already appeared in w at least once$\}$

NFA configuration

Example (continued)

For a string 1010 the automaton can perform the following sequence of moves:
$(q, 1010) \vdash (q, 010) \vdash (q_0, 10) \vdash (q_0, 0) \vdash (q_f, \varepsilon)$.
Let us display all possible sequences of moves for string 1010.



Accessible state

Definition

Let $M = (Q, \Sigma, \delta, q_0, F)$ be a finite automaton. State $q \in Q$ is called *accessible* if there exists a string $w \in \Sigma^*$ such that there exists a sequence of moves that leads from the initial state q_0 into state q:

$(q_0, w) \vdash^* (q, \varepsilon)$.

State that is not accessible is called *unreachable* state.

Accessible state

Algorithm Identification & removal of unreachable states.

Input: NFA $M = (Q, \Sigma, \delta, q_0, F)$.
Output: NFA M' without unreachable states such that $L(M) = L(M')$.

$Q_0 \leftarrow \{q_0\}$; $i \leftarrow 0$
repeat
  $i \leftarrow i + 1$
  $Q_i \leftarrow \{q : q \in \delta(p, a), p \in Q_{i-1}, a \in \Sigma\} \cup Q_{i-1}$
until $Q_i = Q_{i-1}$
$Q_a \leftarrow Q_i$ ▷ the set of all accessible states Q_a
$M' \leftarrow (Q_a, \Sigma, \delta', q_0, F \cap Q_a)$, where δ': $\delta'(q, x) \leftarrow \delta(q, x)$, $\forall x \in \Sigma, \forall q \in Q_a$
return M'

Accessible state

Example

Given finite automaton $M = (\{p, q, r\}, \{a, b\}, \delta, p, \{r\})$, where δ:

		a	b
$\rightarrow$	p	p, r	r
	q	r	
$\leftarrow$	r	p, r	p

Using the algorithm we find out $Q_0 = \{p\}, Q_1 = \{p, r\}, Q_2 = \{p, r\}$.
State q is unreachable.
$M' = (\{p, r\}, \{a, b\}, \delta', p, \{r\})$, where δ':

		a	b
$\rightarrow$	p	p, r	r
$\leftarrow$	r	p, r	p

Useful/redundant state

Definition
Let $M = (Q, \Sigma, \delta, q_0, F)$ be a finite automaton. State $q \in Q$ is called *useful*, if there exists a string $w \in \Sigma^*$ such that there is a sequence of moves that leads from state q into a final state:

$\exists p \in F : (q, w) \vdash^* (p, \varepsilon)$.

The state which is not useful is called a *redundant* state.

Useful/redundant state

Algorithm Identification of useful states and removal of redundant states.

Input: NFA $M = (Q, \Sigma, \delta, q_0, F)$, $L(M) \neq \emptyset$.
Output: NFA M' without redundant states such that $L(M) = L(M')$.

$Q_0 \leftarrow F$; $i \leftarrow 0$
repeat
    $i \leftarrow i + 1$
    $Q_i \leftarrow \{q : \exists p \in Q_{i-1}, p \in \delta(q, a), a \in \Sigma\} \cup Q_{i-1}$
until $Q_i = Q_{i-1}$
$Q_u \leftarrow Q_i$ ▷ the set of all useful states Q_u
$M' \leftarrow (Q_u, \Sigma, \delta', q_0, F)$, where δ': $\delta'(q, x) \leftarrow \delta(q, x) \cap Q_u$, $\forall x \in \Sigma, \forall q \in Q_u$
return M'

Useful/redundant state

Example

Given finite automaton $M = (\{p, q, r\}, \{a, b\}, \delta, p, \{r\})$, where δ:

		a	b
$\rightarrow$	p	q, r	p, r
	q	q	q
$\leftarrow$	r	p	r

Using the algorithm we find out $Q_0 = \{r\}, Q_1 = \{p, r\}, Q_2 = \{p, r\}$.
Therefore $Q_u = \{p, r\}$ and we see that q is redundant.
$M' = (\{p, r\}, \{a, b\}, \delta', p, \{r\})$, where δ':

		a	b
$\rightarrow$	p	r	p, r
$\leftarrow$	r	p	r

Finite automata with ε-transitions

Definition

Nondeterministic finite automaton with ε-transitions is a quintuple $M = (Q, \Sigma, \delta, q_0, F)$, where Q, Σ, q_0, F are the same as in the definition of NFA. Mapping δ is defined as follows:
δ is a mapping from $Q \times (\Sigma \cup \{\varepsilon\})$ into the set of all subsets of Q.

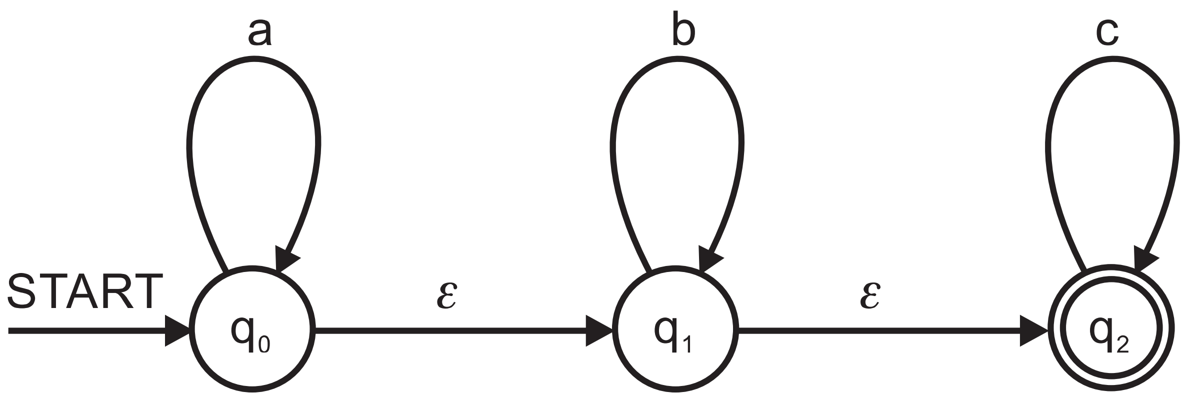
Finite automata with ε-transitions

Example

$M = (\{q_0, q_1, q_2\}, \{a, b, c\}, \delta, q_0, \{q_2\})$, where δ:

$$\begin{aligned} \delta(q_0, a) &= \{q_0\}, & \delta(q_0, \varepsilon) &= \{q_1\}, \\ \delta(q_1, b) &= \{q_1\}, & \delta(q_1, \varepsilon) &= \{q_2\}, \\ \delta(q_2, c) &= \{q_2\}. \end{aligned}$$

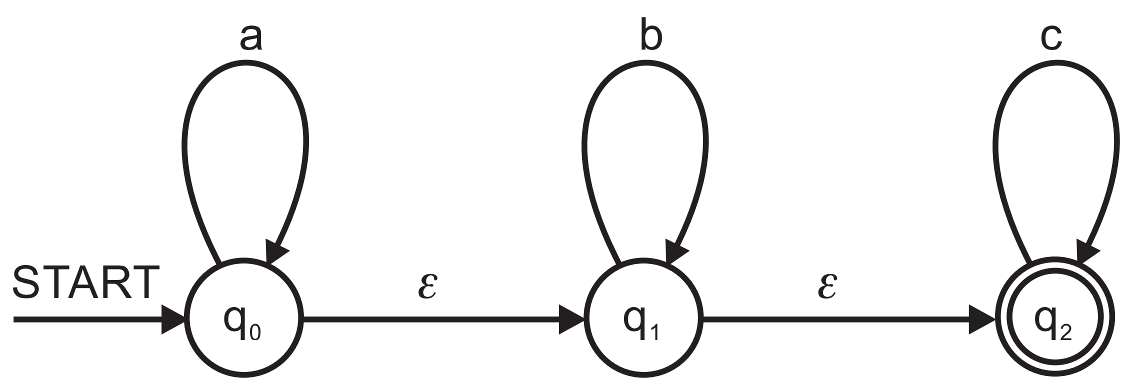
In other cases $\delta(q, x) = \emptyset$.



Finite automata with ε-transitions

Definition
Let $M = (Q, \Sigma, \delta, q_0, F)$ be an NFA with ε-transitions. An element of relation $\vdash_M \subseteq (Q \times \Sigma^*) \times (Q \times \Sigma^*)$ is called *move* in automaton M. If $p \in \delta(q, a),\ a \in \Sigma \cup \{\varepsilon\}$, then $(q, aw) \vdash_M (p, w)$ for every $w \in \Sigma^*$.

Example



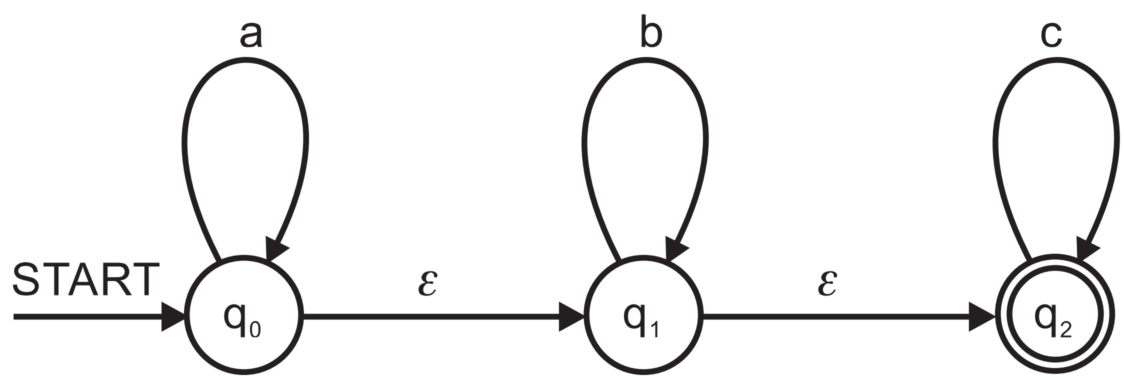
Finite automaton will perform the following sequence of moves for input string abc:
$(q_0, abc) \vdash (q_0, bc) \vdash (q_1, bc) \vdash (q_1, c) \vdash (q_2, c) \vdash (q_2, \varepsilon)$.

ε-*Closure*

Definition

Function ε-$Closure$: $Q \mapsto 2^Q$ for a finite automaton $M = (Q, \Sigma, \delta, q_0, F)$ is defined as follows:
$\varepsilon\text{-}Closure(q) = \{p : (q, \varepsilon) \vdash^* (p, \varepsilon), p \in Q\}$.

Example



$$\begin{aligned} \varepsilon\text{-}Closure(q_0) &= \{q_0, q_1, q_2\}, \\ \varepsilon\text{-}Closure(q_1) &= \{q_1, q_2\}, \\ \varepsilon\text{-}Closure(q_2) &= \{q_2\}. \end{aligned}$$

Removal of ε-transitions

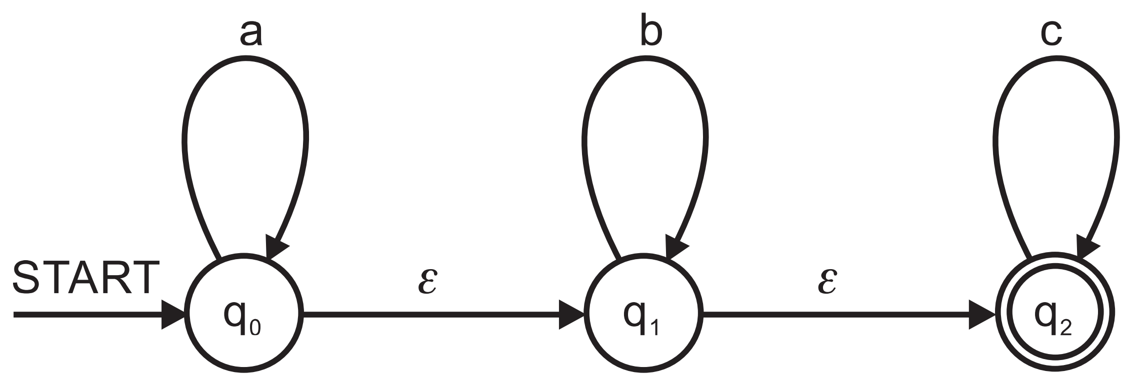
Algorithm Convertion of NFA with ε-transitions into NFA without ε-transitions.

Input: NFA $M = (Q, \Sigma, \delta, q_0, F)$ with ε-transitions.
Output: NFA M' without ε-transitions such that $L(M) = L(M')$.

1: $M' \leftarrow (Q, \Sigma, \delta', q_0, F')$
2: $\delta'(q, a) \leftarrow \bigcup_{p \in \varepsilon\text{-}Closure(q)} \delta(p, a)$, $\forall a \in \Sigma$
3: $F' \leftarrow \{q : \varepsilon\text{-}Closure(q) \cap F \neq \emptyset, q \in Q\}$
4: **return** M'

Removal of ε-transitions

Example



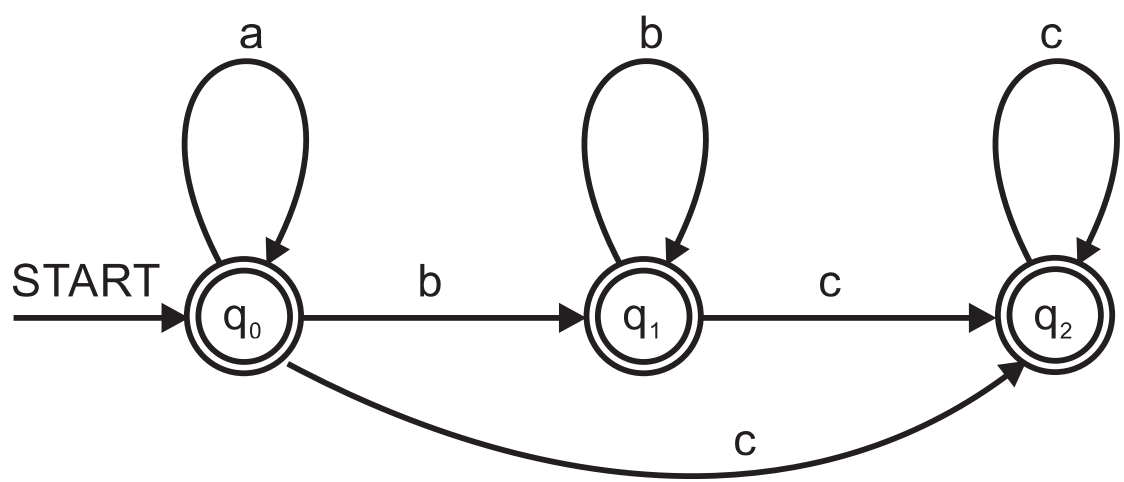
$M' = (Q, \Sigma, \delta', q_0, F')$, where

$$
\begin{aligned}
\delta'(q_0, a) &= \delta(q_0, a) \cup \delta(q_1, a) \cup \delta(q_2, a) = \{q_0\} \cup \emptyset \cup \emptyset = \{q_0\},\\
\delta'(q_0, b) &= \delta(q_0, b) \cup \delta(q_1, b) \cup \delta(q_2, b) = \emptyset \cup \{q_1\} \cup \emptyset = \{q_1\},\\
\delta'(q_0, c) &= \delta(q_0, c) \cup \delta(q_1, c) \cup \delta(q_2, c) = \emptyset \cup \emptyset \cup \{q_2\} = \{q_2\},\\
\delta'(q_1, a) &= \delta(q_1, a) \cup \delta(q_2, a) = \emptyset \cup \emptyset = \emptyset,\\
\delta'(q_1, b) &= \delta(q_1, b) \cup \delta(q_2, b) = \{q_1\} \cup \emptyset = \{q_1\},\\
\delta'(q_1, c) &= \delta(q_1, c) \cup \delta(q_2, c) = \emptyset \cup \{q_2\} = \{q_2\},\\
\delta'(q_2, a) &= \emptyset,\ \delta'(q_2, b) = \emptyset,\ \delta'(q_2, c) = \{q_2\}.
\end{aligned}
$$

Removal of ε-transitions

Example (continued)

$F' = \{q_0, q_1, q_2\}$, because
$\varepsilon\text{-}Closure(q_0) \cap F = \{q_2\}$,
$\varepsilon\text{-}Closure(q_1) \cap F = \{q_2\}$,
$\varepsilon\text{-}Closure(q_2) \cap F = \{q_2\}$.



The resulting automaton is deterministic if we identify one-element set with the element (i.e., $q_i = \{q_i\}$).

FA with multiple initial states

Definition

Nondeterministic finite automaton with set I of initial states is a quintuple $M = (Q, \Sigma, \delta, I, F)$, where

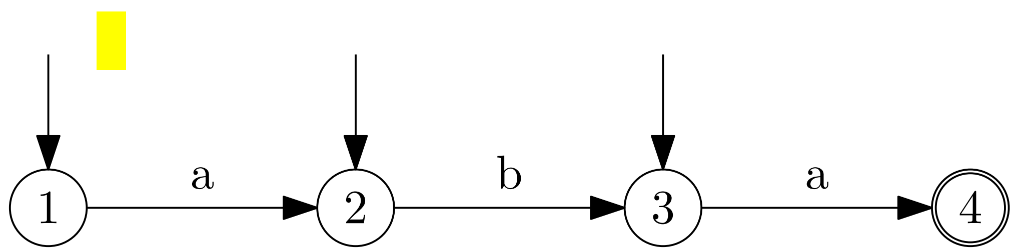
- Q, Σ, δ, F are the same as in the NFA definition,
- I is a nonempty subset of the set of states, $I \subseteq Q$.

The sequence of moves of this finite automaton can start in any state $q \in I$.

FA with multiple initial states

Example

Given $M = (\{q_1, q_2, q_3, q_4\}, \{a, b\}, \delta, \{q_1, q_2, q_3\}, \{q_4\})$, where δ:
$\delta(q_1, a) = \{q_2\}$, $\delta(q_3, a) = \{q_4\}$, $\delta(q_2, b) = \{q_3\}$.
In the other cases $\delta(q, x) = \emptyset$, where $x \in \{a, b\}$.

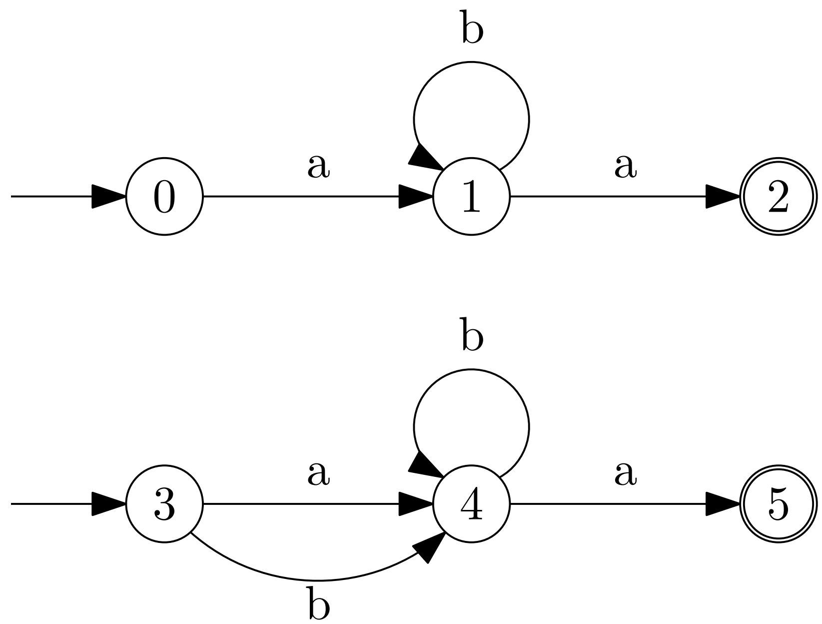


$$M\colon \begin{array}{r} (q_1, aba) \vdash (q_2, ba) \vdash (q_3, a) \vdash (q_4, \varepsilon), \\ (q_2, ba) \vdash (q_3, a) \vdash (q_4, \varepsilon), \\ (q_3, a) \vdash (q_4, \varepsilon). \end{array}$$

FA with multiple initial states

Example

$M = (\{q_0, q_1, q_2, q_3, q_4, q_5\}, \{a, b\}, \delta, \{q_0, q_3\}, \{q_2, q_5\})$, where δ:



M: $(q_0, aba) \vdash (q_1, ba) \vdash (q_1, a) \vdash (q_2, \varepsilon)$,
$(q_3, bba) \vdash (q_4, ba) \vdash (q_4, a) \vdash (q_5, \varepsilon)$.

Transform. to NFA with one initial state

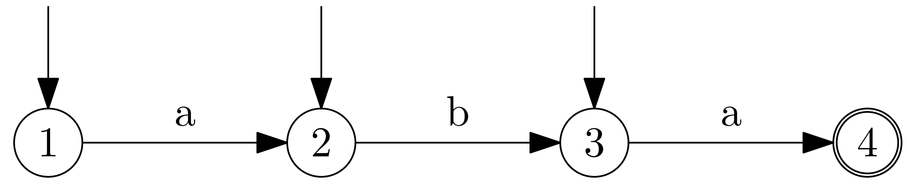
Algorithm Transformation of NFA with multiple intial states to NFA with one initial state

Input: NFA $M = (Q, \Sigma, \delta, I, F),\ |I| > 1.$
Output: NFA M') such that $L(M) = L(M')$.

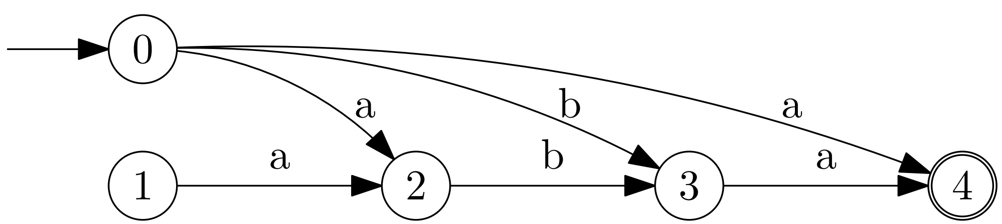
$M' \leftarrow (Q', \Sigma, \delta', q_0, F')$
$Q' \leftarrow Q \cup \{q_0\}, q_0 \notin Q$
$\delta'(q_0, a) \leftarrow \bigcup_{q \in I} \delta(q, a), \forall a \in \Sigma$
$\delta'(q, a) \leftarrow \delta(q, a), \forall a \in \Sigma, \forall q \in Q$
$F' \leftarrow F$ if $F \cap I = \emptyset$
$F' \leftarrow F \cup \{q_0\}$, if $F \cap I \neq \emptyset$
return M'

Transform. to NFA with one initial state

Example

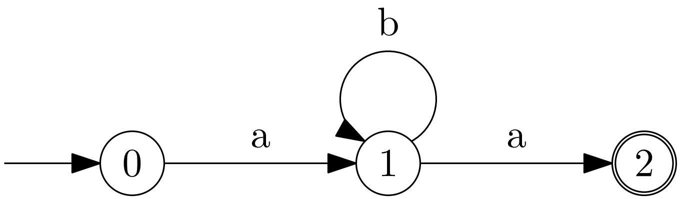


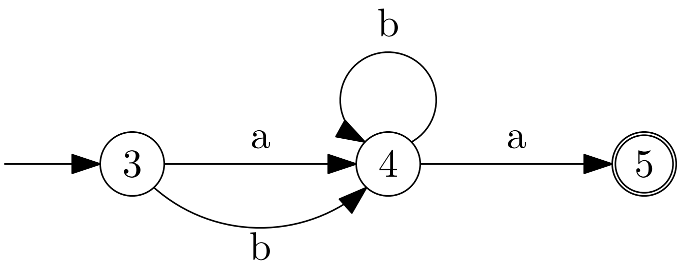
We construct an equivalent NFA with only one initial state:
$M' = (Q', \Sigma, \delta', q_0, F')$, $Q' = \{q_0, q_1, q_2, q_3, q_4\}$,
$\delta'(q_0, a) = \{q_2, q_4\}$, $\delta'(q_0, b) = \{q_3\}$,
$F' = F = \{q_4\}$



Transform. to NFA with one initial state

Example





$M' = (Q', \Sigma, \delta', q_6, F)$, $Q' = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6\}$
$\delta'(q_6, a) = \{q_1, q_4\}$, $\delta'(q_6, b) = \{q_4\}$
$F' = F = \{q_2, q_5\}$