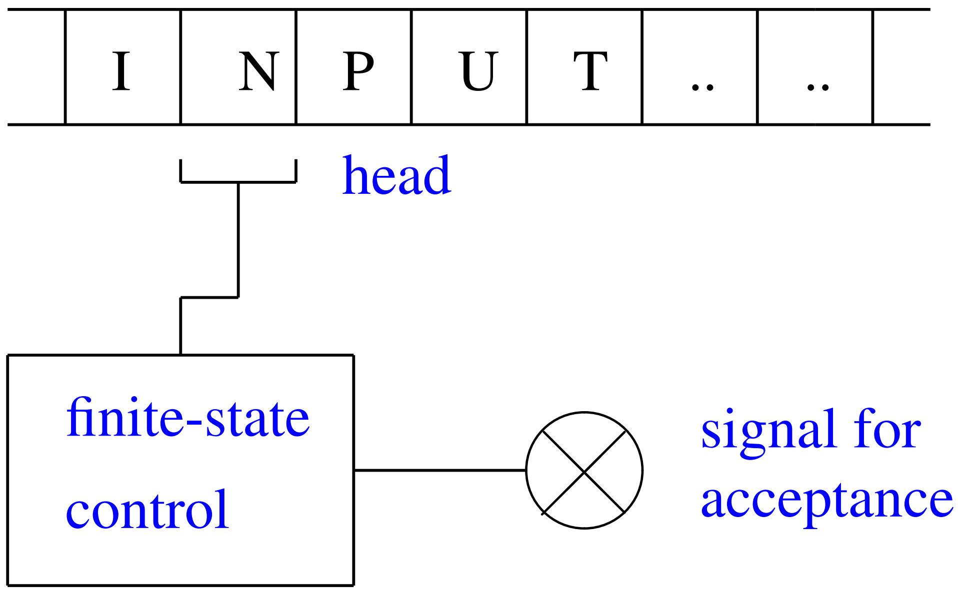
2.3 Deterministic Finite-State Automaton



A finite-state automaton reads an input word letter by letter from left to right and accepts or rejects.

finite-state automaton $\rightsquigarrow$ language of accepted words

Definition 2.6

A deterministic finite-state automaton (DFA) A is given through a 5-tuple $A = (Q, \Sigma, \delta, q_0, F)$, *where*

- Q *is a finite set of (internal) states,*
- Σ *is a finite (input) alphabet,*
- $\delta : Q \times \Sigma \rightsquigarrow Q$ *is a (partial) transition function,*
- $q_0 \in Q$ *is the initial state, and*
- $F \subseteq Q$ *is a set of accepting states.*

If δ *is defined for all* $(q, a) \in Q \times \Sigma$, *then A is a complete* DFA.

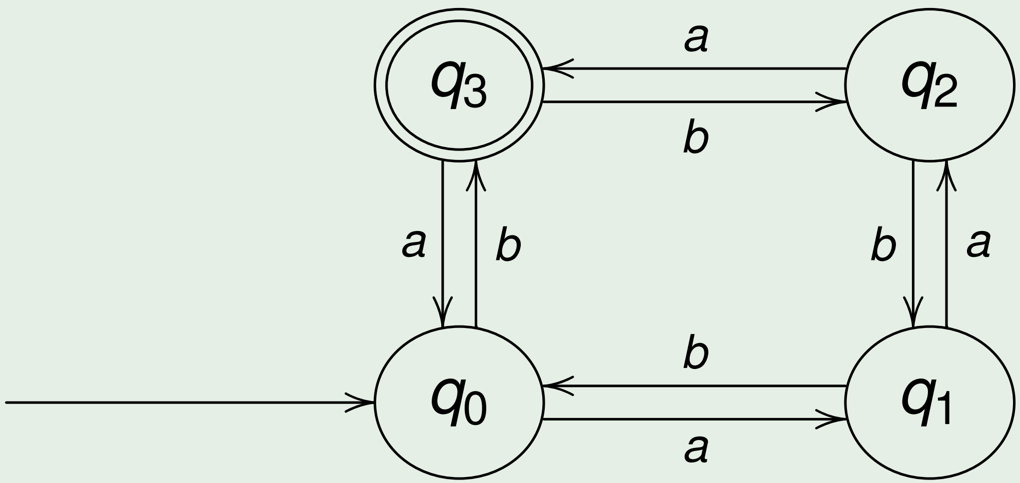
A DFA can be described by a state graph $G = (Kn, Ka)$:

- $Kn := Q$,
- Ka : If $\delta(q_1, a) = q_2$, then there is a directed edge with label a from q_1 to q_2.
- q_0 and $q \in F$ are marked in a special way.

Example:

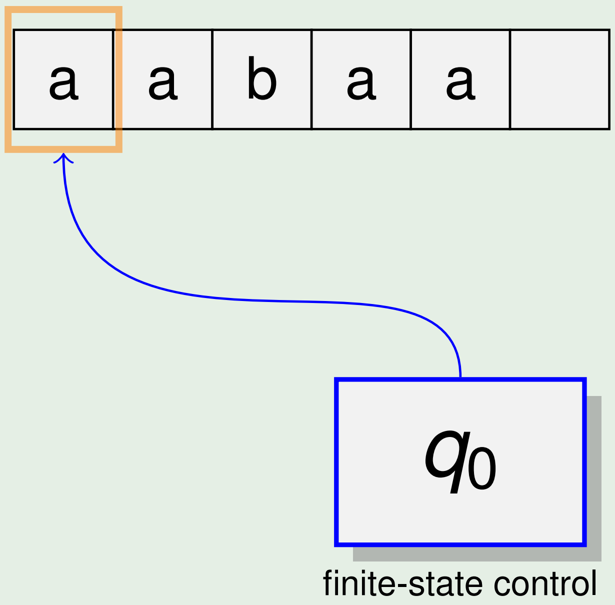
$A = (Q, \Sigma, \delta, q_0, F)$, where
$Q = \{q_0, q_1, q_2, q_3\}$, $\Sigma = \{a, b\}$, $F = \{q_3\}$, and
$\delta(q_0, a) = q_1$, $\delta(q_0, b) = q_3$
$\delta(q_1, a) = q_2$, $\delta(q_1, b) = q_0$
$\delta(q_2, a) = q_3$, $\delta(q_2, b) = q_1$
$\delta(q_3, a) = q_0$, $\delta(q_3, b) = q_2$

State graph:



Example (cont.):

Input: *aabaa*

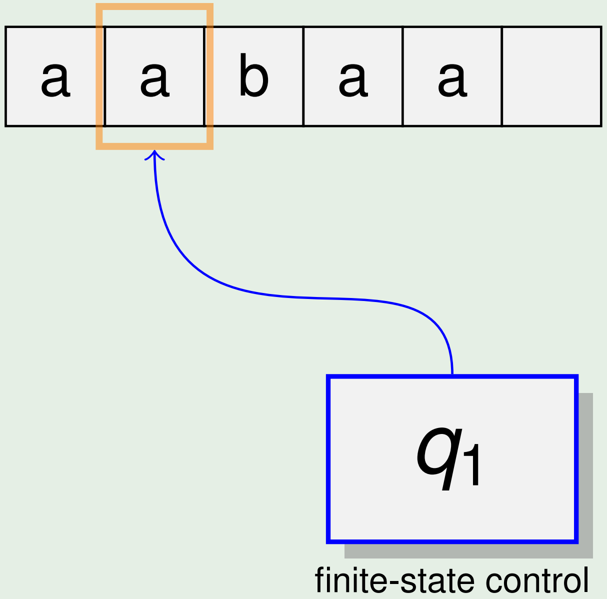


Computation: $q_0 \xrightarrow{a} q_1 \xrightarrow{a} q_2 \xrightarrow{b} q_1 \xrightarrow{a} q_2 \xrightarrow{a} q_3$

Result: *aabaa* is accepted.

Example (cont.):

Input: *aabaa*

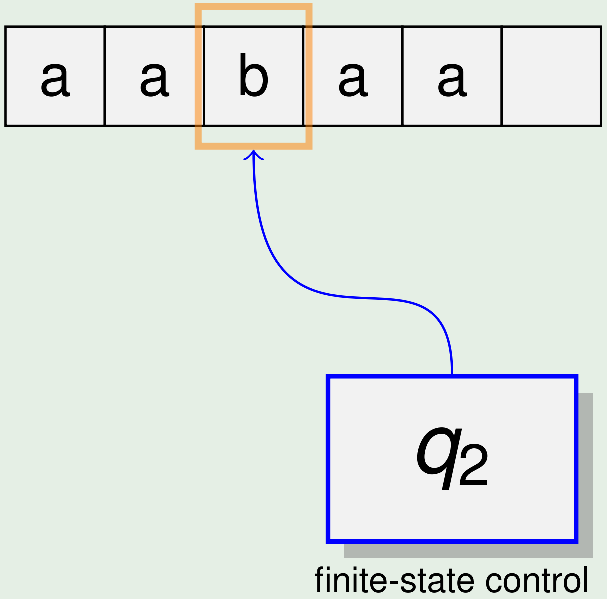


Computation: $q_0 \xrightarrow{a} q_1 \xrightarrow{a} q_2 \xrightarrow{b} q_1 \xrightarrow{a} q_2 \xrightarrow{a} q_3$

Result: *aabaa* is accepted.

Example (cont.):

Input: *aabaa*

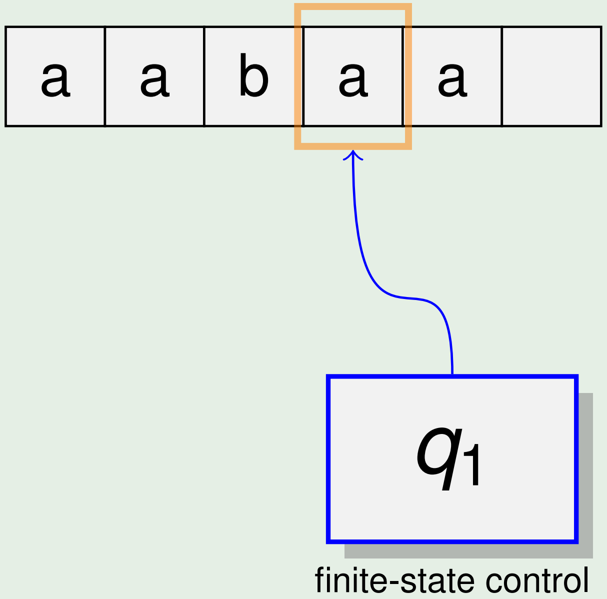


Computation: $q_0 \xrightarrow{a} q_1 \xrightarrow{a} q_2 \xrightarrow{b} q_1 \xrightarrow{a} q_2 \xrightarrow{a} q_3$

Result: *aabaa* is accepted.

Example (cont.):

Input: *aabaa*

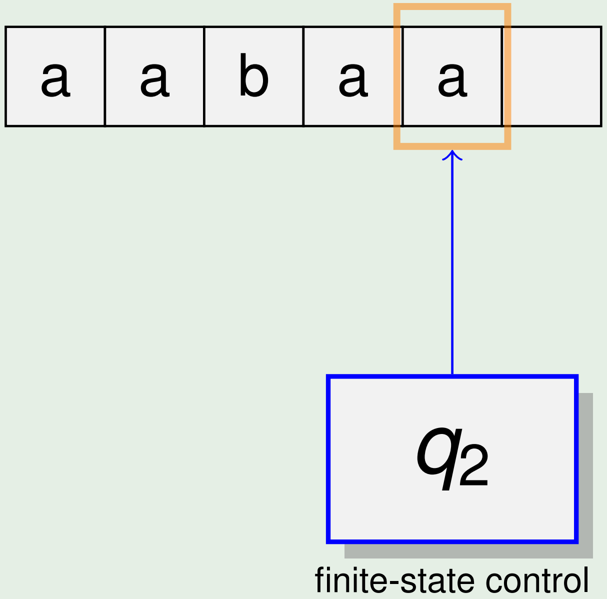


Computation: $q_0 \xrightarrow{a} q_1 \xrightarrow{a} q_2 \xrightarrow{b} q_1 \xrightarrow{a} q_2 \xrightarrow{a} q_3$

Result: *aabaa* is accepted.

Example (cont.):

Input: *aabaa*

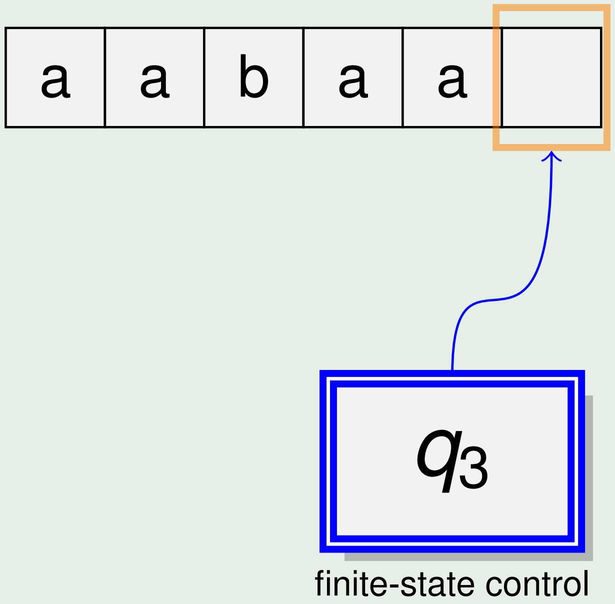


Computation: $q_0 \xrightarrow{a} q_1 \xrightarrow{a} q_2 \xrightarrow{b} q_1 \xrightarrow{a} q_2 \xrightarrow{a} q_3$

Result: *aabaa* is accepted.

Example (cont.):

Input: *aabaa*



Computation: $q_0 \xrightarrow{a} q_1 \xrightarrow{a} q_2 \xrightarrow{b} q_1 \xrightarrow{a} q_2 \xrightarrow{a} q_3$

Result: *aabaa* is accepted.

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA.
The function $\delta : Q \times \Sigma \rightsquigarrow Q$ can be extended to a function $\hat{\delta} : Q \times \Sigma^* \rightsquigarrow Q$, where $q \in Q$, $u \in \Sigma^*$, and $a \in \Sigma$:

- $\hat{\delta}(q, \varepsilon) := q$,
- $\hat{\delta}(q, ua) = \begin{cases} \delta(\hat{\delta}(q, u), a), & \text{if } \hat{\delta}(q, u) \text{ is defined,} \\ \text{undefined,} & \text{otherwise.} \end{cases}$

$L_{A,q_1,q_2} := \{ u \in \Sigma^* \mid \hat{\delta}(q_1, u) = q_2 \}$ is the set of words that take A from state q_1 to state q_2.

If $\hat{\delta}(q_0, w) \in F$, then the word w is accepted by the DFA A.

The set $L(A) := \bigcup_{q \in F} L_{A,q_0,q}$ of all words that are accepted by A is called the language accepted by A.

Notice:

For all $q \in Q$ and $a_1, a_2, \ldots, a_n \in \Sigma$,

$$\begin{aligned}\hat{\delta}(q, a_1 a_2 \ldots a_n) &= \hat{\delta}(\delta(q, a_1), a_2 \ldots a_n) \\ &= \hat{\delta}(\delta(\delta(q, a_1), a_2), a_3 \ldots a_n) \\ &= \delta(\delta(\ldots \delta(\delta(q, a_1), a_2) \ldots, a_{n-1}), a_n)\end{aligned}$$

For all $q \in Q$ and $u, v \in \Sigma^*$, $\hat{\delta}(q, uv) = \hat{\delta}(\hat{\delta}(q, u), v)$.

Example (cont.):

$L(A) = \{ x \in \Sigma^* \mid |x|_a - |x|_b \equiv 3 \bmod 4 \}$.

A word of the form $qu \in Q \cdot \Sigma^*$ is called a configuration of A.

The DFA A induces a computation relation $\vdash_A^*$ on $Q \cdot \Sigma^*$.

It follows that $L(A) = \{ u \in \Sigma^* \mid q_0 u \vdash_A^* q \text{ for some } q \in F \}$.

Lemma 2.7

From a given DFA A, a complete DFA B can be constructed such that $L(B) = L(A)$.

Proof.

Let $A = (Q, \Sigma, \delta, q_0, F)$ be an incomplete DFA.

We define a DFA $B = (Q \cup \{\perp\}, \Sigma, \delta', q_0, F)$ through

$$\delta'(q, a) = \begin{cases} \delta(q, a), & \text{if } \delta(q, a) \text{ is defined,} \\ \perp, & \text{otherwise,} \end{cases}$$

$$\delta'(\perp, a) = \perp \text{ for all } a \in \Sigma.$$

Proof of Lemma 2.7 (cont.).)

Then the following equality holds for all $q \in Q$ and all $u \in \Sigma^*$:

$$\delta'(q,u) = \begin{cases} \delta(q,u), & \text{if } \delta(q,u) \text{ is defined,} \\ \bot, & \text{otherwise.} \end{cases}$$

Hence,

$$L(A) = \{\, u \in \Sigma^* \mid \delta(q_0,u) \in F \,\} = \{\, u \in \Sigma^* \mid \delta'(q_0,u) \in F \,\} = L(B). \qquad \square$$

Let $\mathcal{L}(\text{DFA})$ denote the class of languages that are accepted by DFAs.

Theorem 2.8

$\mathcal{L}(\text{DFA})$ *is closed under the operations of union, intersection, and complement.*

Proof.

Let $L = L(A)$, where $A = (Q, \Sigma, \delta, q_0, F)$ is a complete DFA.

Then $\overline{A} := (Q, \Sigma, \delta, q_0, Q \setminus F)$ is a complete DFA for $\Sigma^* \setminus L(A) = L^C$.
Hence, $\mathcal{L}(\text{DFA})$ is closed under the operation of complement.

Let $A_i = (Q_i, \Sigma, \delta_i, q_0^{(i)}, F_i)$ be a complete DFA for L_i, $i = 1, 2$.
A DFA for the intersection $L_1 \cap L_2$ is obtained by taking
$A = (Q, \Sigma, \delta, q_0, F)$, where
$Q := Q_1 \times Q_2$, $q_0 := (q_0^{(1)}, q_0^{(2)})$, $F := F_1 \times F_2$, and
$\delta((q_1, q_2), a) := (\delta_1(q_1, a), \delta_2(q_2, a))$ for all $q_1 \in Q_1$, $q_2 \in Q_2$, $a \in \Sigma$.

Then $\delta(q_0, w) = (\delta_1(q_0^{(1)}, w), \delta_2(q_2^{(2)}, w)) \in F = F_1 \times F_2$
iff $w \in L_1$ and $w \in L_2$, that is, $L(A) = L_1 \cap L_2$.

A is called the product automaton of A_1 and A_2.

By De Morgan's law, closure under union follows. □

Theorem 2.9

If a language L is accepted by a DFA, then L is right regular.

Proof.

Let $L \subseteq \Sigma^*$ be accepted by a complete DFA $A = (Q, \Sigma, \delta, q_0, F)$.

We define a right regular grammar $G := (V, \Sigma, P, S)$ as follows:

– $V := Q$,

– $S := q_0$,

– $P := \{ q_1 \rightarrow aq_2 \mid \delta(q_1, a) = q_2 \}$
$\quad\quad \cup \{ q_1 \rightarrow a \mid \delta(q_1, a) = q_2 \text{ and } q_2 \in F \}$

If $q_0 \in F$, i.e., $\varepsilon \in L(A)$, then we add the new start symbol $\hat{q}_0$ and the following productions:

$$\{\hat{q}_0 \rightarrow \varepsilon\} \quad \text{and} \quad \{ \hat{q}_0 \rightarrow aq_2 \mid \delta(q_0, a) = q_2 \}$$
$$\text{and} \quad \{ \hat{q}_0 \rightarrow a \mid \delta(q_0, a) \in F\}.$$

Proof of Theorem 2.9 (cont.).

Claim.

$\forall x \in \Sigma^+ : x \in L(A)$ iff $x \in L(G)$.

Proof.

$x = a_1 a_2 \ldots a_n \in L(A)$
iff
$\exists q_1, q_2, \ldots, q_n \in Q : q_n \in F$ and $\delta(q_{i-1}, a_i) = q_i \quad (1 \leq i \leq n)$
iff
$\exists q_1, \ldots, q_n \in V : q_0 \to a_1 q_1 \to \ldots \to a_1 \ldots a_{n-1} q_{n-1} \to a_1 \ldots a_{n-1} a_n$
iff
$x = a_1 a_2 \ldots a_n \in L(G)$. □□

There are many DFA that accept the same language.
Among these is there a smallest DFA?

Let $L \subseteq \Sigma^*$.

Definition 2.10

The Nerode relation $R_L \subseteq \Sigma^* \times \Sigma^*$ for L is defined through

$$x \; R_L \; y \text{ iff } \forall z \in \Sigma^* : (xz \in L \text{ iff } yz \in L).$$

Lemma 2.11

- *R_L is an equivalence relation on Σ^*.*
- *If $x \; R_L \; y$, then $xw \; R_L \; yw$ for all $w \in \Sigma^*$.*
- *R_L partitions Σ^* (and L) into disjoint equivalence classes.*

The index of R_L := number of equivalence classes of R_L.

Theorem 2.12 (Myhill, Nerode)

A language L is accepted by a DFA iff the relation R_L has finite index.

Proof.

"$\Rightarrow$": Let $A = (Q, \Sigma, \delta, q_0, F)$ be a complete DFA for L. We define a binary relation $R_A \subseteq \Sigma^* \times \Sigma^*$: $x \; R_A \; y \quad \text{iff} \quad \hat{\delta}(q_0, x) = \hat{\delta}(q_0, y)$.

Claim. $R_A \subseteq R_L$.

Proof.

Assume that $x \; R_A \; y$, and let $z \in \Sigma^*$.

Then: $xz \in L$ iff $\hat{\delta}(q_0, xz) \in F$ iff $\hat{\delta}(\hat{\delta}(q_0, x), z) \in F$
iff $\hat{\delta}(\hat{\delta}(q_0, y), z) \in F$ iff $\hat{\delta}(q_0, yz) \in F$
iff $yz \in L$.

Hence: $x \; R_L \; y$. □

Thus: $\text{index}(R_L) \leq \text{index}(R_A) \leq |Q| < \infty$.

Proof of Theorem 2.12 (cont.)

"$\Leftarrow$": Assume that R_L has finite index.
Then there are finitely many words $x_1, x_2, \ldots, x_k \in \Sigma^*$ such that
$\Sigma^* = [x_1] \cup [x_2] \cup \ldots \cup [x_k]$.

We define a complete DFA $A_0 := (Q_0, \Sigma, \delta, q_0, F)$:

– $Q_0 := \{ [x_1], [x_2], \ldots, [x_k] \}$,
– $q_0 := [\varepsilon]$,
– $F := \{ [x_i] \mid x_i \in L \}$,
– $\delta([x_i], a) := [x_i a]$ (for all $i = 1, 2, \ldots, k$ and $a \in \Sigma$).

Then $\hat{\delta}([\varepsilon], x) = [x]$ for all $x \in \Sigma^*$, that is,

$$\begin{aligned} x \in L(A_0) \quad & \text{iff} \quad \hat{\delta}(q_0, x) \in F \\ & \text{iff} \quad \hat{\delta}([\varepsilon], x) \in F \\ & \text{iff} \quad [x] \in F \\ & \text{iff} \quad x \in L. \end{aligned}$$

□

A_0 is called the equivalence class automaton for L.

Example 1:

$L = \{ a^n b^n \mid n \geq 1 \}$

Then:

$$\begin{array}{rcl} [ab] & = & L \\ [a^2 b] & = & \{a^2 b, a^3 b^2, a^4 b^3, \ldots\} \\ [a^3 b] & = & \{a^3 b, a^4 b^2, a^5 b^3, \ldots\} \\ & \vdots & \\ [a^k b] & = & \{ a^{k+i-1} b^i \mid i \geq 1 \} \end{array}$$

For all $i \neq j$, $a^i b$ and $a^j b$ are not equivalent w.r.t. R_L, that is, $\mathrm{index}(R_L) = \infty$.

Hence, L is not accepted by any DFA.

Example 2:

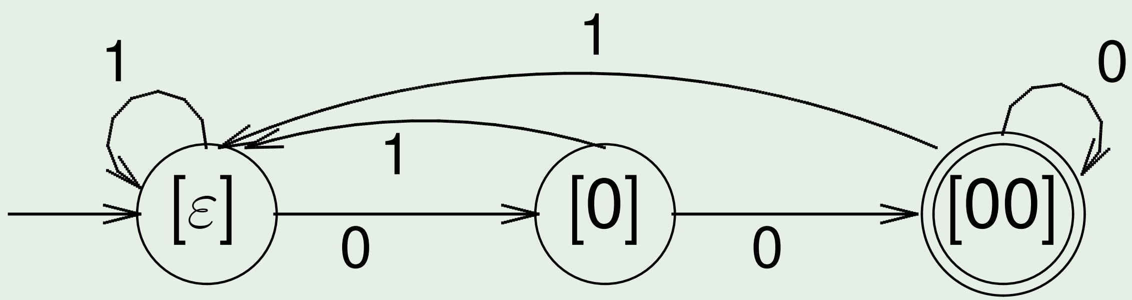
$L = \{ x \in \{0,1\}^* \mid x \text{ has suffix } 00 \}$

Then:

$$\begin{array}{lcl} [\varepsilon] & = & \{ x \mid x \text{ does not have suffix } 0 \} \\ [0] & = & \{ x \mid x \text{ has suffix } 0, \text{ but not suffix } 00 \} \\ [00] & = & \{ x \mid x \text{ has suffix } 00 \} \end{array}$$

$\{0,1\}^* = [\varepsilon] \cup [0] \cup [00]$, and hence, $\text{index}(R_L) = 3$.

Equivalence class automaton A_0 for L:



Assume that L is accepted by a complete DFA A, and let A_0 be the equivalence class automaton for L.

Then: $R_A \subseteq R_L = R_{A_0}$,

and so: $|Q| \geq |Q_0| = \mathrm{index}(R_L)$.

In fact: If $|Q| = |Q_0|$, then $R_A = R_L$, that is, A and A_0 are isomorphic.

Thus: A_0 is the minimal automaton for L.

How to determine whether a given DFA A is minimal?

A is not minimal iff
$\exists q, q' \in Q, q \neq q' : \forall x \in \Sigma^* : \hat{\delta}(q, x) \in F$ iff $\hat{\delta}(q', x) \in F$.

Algorithm "minimal automaton"

Input: a complete DFA $A = (Q, \Sigma, \delta, q_0, F)$.

(0) Delete all states that are not reachable from q_0.

(1) Initialize a table of all pairs $\{q, q'\}$ such that $q \neq q'$.

(2) Mark all pairs $\{q, q'\}$ for which $\{q, q'\} \cap F \neq \emptyset$ and $\{q, q'\} \cap (Q \smallsetminus F) \neq \emptyset$.

(3) For each unmarked pair $\{q, q'\}$ and each $a \in \Sigma$,
check whether $\{\delta(q, a), \delta(q', a)\}$ is marked.
If so, then mark $\{q, q'\}$ as well.

(4) Repeat (3) until no further pairs are marked.

(5) For each unmarked pair $\{q, q'\}$,
merge the states q and q' into a joint state.

Output: The minimal automaton for $L(A)$

Example 1:

A :

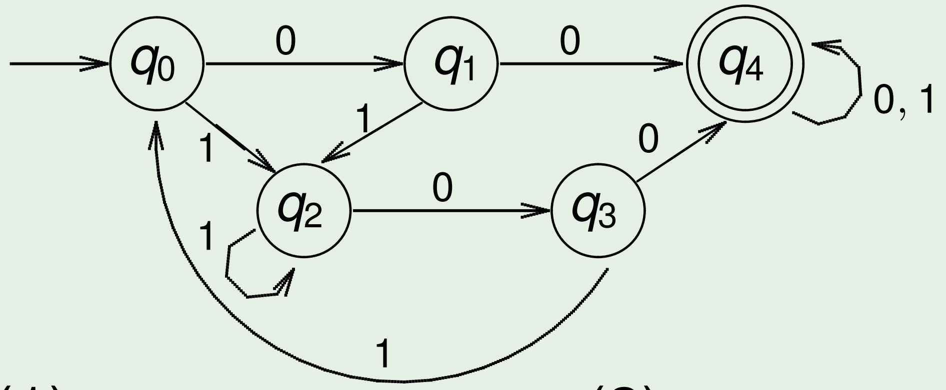


Table: (1)

	q_0	q_1	q_2	q_3
q_1				
q_2				
q_3				
q_4				

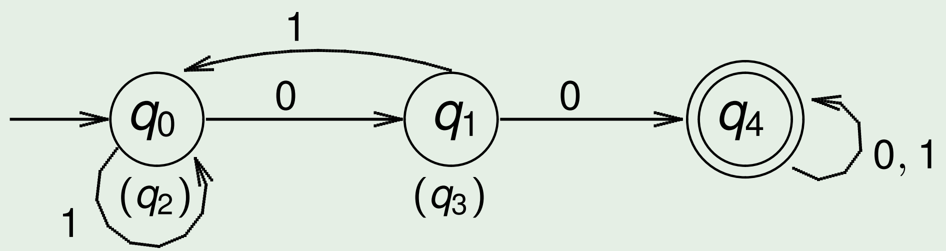
(2)

	q_0	q_1	q_2	q_3
q_1				
q_2				
q_3				
q_4	x	x	x	x

(3)

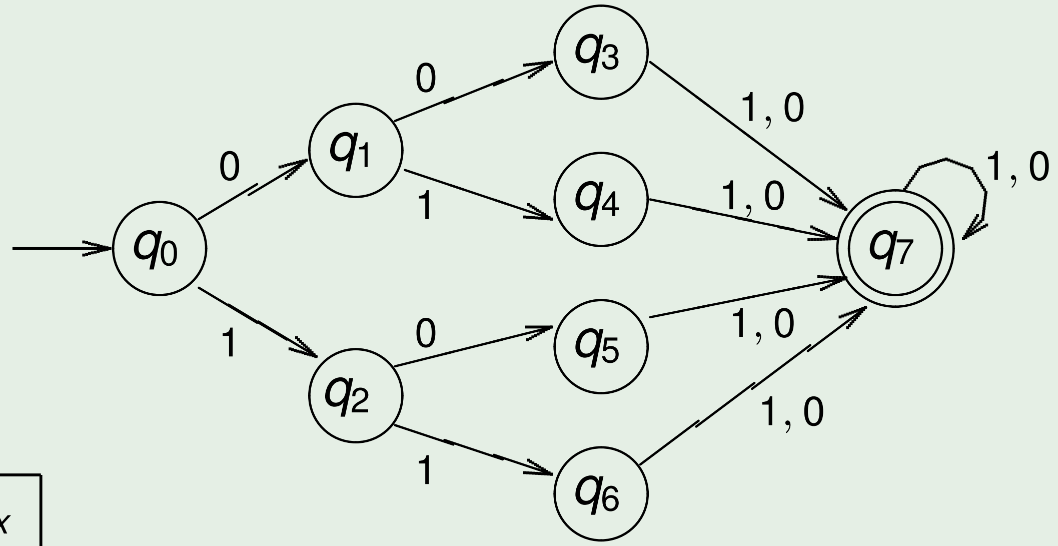
	q_0	q_1	q_2	q_3
q_1	x			
q_2		x		
q_3	x		x	
q_4	x	x	x	x

A_0 :



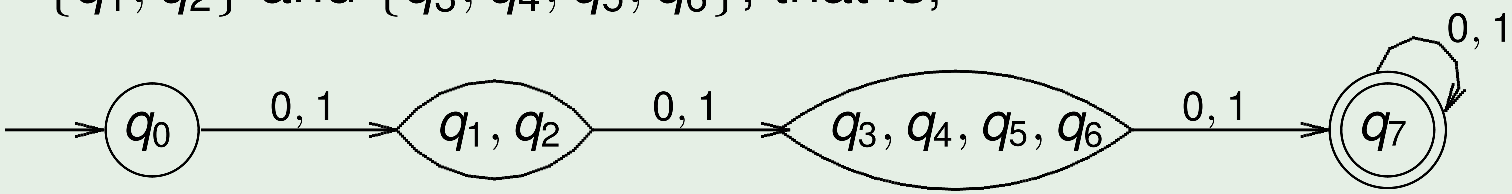
$L(A) = L(A_0) = \{ x \mid x \text{ contains } 00 \text{ as a factor} \}$.

Example 2:



q_1	x						
q_2	x						
q_3	x	x	x				
q_4	x	x	x				
q_5	x	x	x				
q_6	x	x	x				
q_7	x	x	x	x	x	x	x
	q_0	q_1	q_2	q_3	q_4	q_5	q_6

Hence: $\{q_1, q_2\}$ and $\{q_3, q_4, q_5, q_6\}$, that is,



$L(A) = L(A_0) = \{ w \in \{0, 1\}^* \mid |w| \geq 3 \}$

Let $h : \Sigma^* \to \Delta^*$ be a morphism. Then $h^{-1} : \Delta^* \to 2^{\Sigma^*}$ is defined through

$$h^{-1}(v) := \{ u \in \Sigma^* \mid h(u) = v \}.$$

For a language $L \subseteq \Delta^*$, $h^{-1}(L)$ is defined as

$$h^{-1}(L) := \{ u \in \Sigma^* \mid h(u) \in L \}.$$

Theorem 2.13

The language class $\mathcal{L}(\mathrm{DFA})$ is closed under inverse morphisms, that is, if $h : \Sigma^* \to \Delta^*$ is a morphism and $L \subseteq \Delta^*$ is accepted by a DFA, then also $h^{-1}(L)$ is accepted by a DFA.

Proof of Theorem 2.13.

Let $A = (Q, \Delta, \delta, q_0, F)$ be a complete DFA such that $L(A) = L$ and let $h : \Sigma^* \to \Delta^*$ be a morphism.
We construct a DFA $B := (Q, \Sigma, \delta', q_0, F)$ by taking

$$\forall q \in Q \, \forall a \in \Sigma : \delta'(q, a) := \delta(q, h(a)).$$

Claim:

For all $x \in \Sigma^*$, $\delta'(q_0, x) = \delta(q_0, h(x))$.

Proof by induction on $|x|$:

If $x = \varepsilon$, then $\delta'(q_0, x) = q_0 = \delta(q_0, \varepsilon) = \delta(q_0, h(x))$.

Assume that the claim has been proved for some $n \geq 0$,
let x be a word of length n, and let $a \in \Sigma$. Then:

$$\delta'(q_0, xa) = \delta'(\delta'(q_0, x), a) \underset{\text{I.H.}}{=} \delta'(\delta(q_0, h(x)), a) = \delta(\delta(q_0, h(x)), h(a)) = \delta(q_0, h(x)h(a)) = \delta(q_0, h(xa)). \quad \square$$

Thus, for all $x \in \Sigma^*$: $x \in L(B)$ iff $\delta'(q_0, x) \in F$
iff $\delta(q_0, h(x)) \in F$
iff $h(x) \in L$
iff $x \in h^{-1}(L)$,

that is, $L(B) = h^{-1}(L)$. $\square$