Theorem 2.16

From a right regular grammar G, one can construct an NFA A such that $L(A) = L(G)$.

Proof.

Based on Theorem 2.5 we can first transform the grammar G into an equivalent grammar $G' = (N, T, S, P)$ that is in right normal form, that is, it only has productions of the form

$$A \to aB \text{ and } A \to a, \text{ where } A, B \in N \text{ and } a \in T,$$

and possibly the production $S \to \varepsilon$.

Proof of Theorem 2.16 (cont.)

We take $A := (Q, T, \delta, S', F)$, where
– $Q := N \cup \{X\}$ (X a new symbol),
– $S' := \{S\}$,
– $F := \begin{cases} \{S, X\}, & \text{if } (S \to \varepsilon) \in P, \\ \{X\}, & \text{otherwise}, \end{cases}$
– $\delta(A, a) := \{\, B \mid (A \to aB) \in P \,\} \cup \{\, X \mid (A \to a) \in P \,\}$
for all $A \in N$ and $a \in T$.

Then: $\varepsilon \in L(G)$ iff $(S \to \varepsilon) \in P$
iff $S \in F$
iff $\varepsilon \in L(A)$.

For all $n \geq 1$: $a_1 a_2 \ldots a_n \in L(G)$ iff $a_1 a_2 \ldots a_n \in L(G')$ iff
$\exists A_1, \ldots, A_{n-1} \in N : S \to a_1 A_1 \to \ldots \to a_1 a_2 \ldots a_{n-1} A_{n-1} \to a_1 \ldots a_n$
iff
$\exists A_1, \ldots, A_{n-1} \in N : A_1 \in \delta(S, a_1), A_2 \in \delta(A_1, a_2), \ldots, X \in \delta(A_{n-1}, a_n)$
iff $a_1 a_2 \ldots a_n \in L(A)$. □

Theorem 2.17

The class of languages $\mathcal{L}(\mathrm{DFA})$ is closed under the operation of taking the mirror image.

Proof.

Let $A = (Q, \Sigma, \delta, q_0, F)$ be a DFA.
By A^R we denote the NFA $A^R = (Q, \Sigma, \delta^R, F, \{q_0\})$,
where $\delta^R : Q \times \Sigma \to 2^Q$ is defined as follows:

$$\delta^R(p, a) := \{\, q \mid \delta(q, a) = p \,\} \quad (p \in Q, a \in \Sigma),$$

that is, initial and final states are interchanged,
and each transition is simply reversed.

Then $L(A^R) = (L(A))^R$. □

Corollary 2.18

For any language L, the following statements are equivalent:

(1) *L is a regular language.*

(2) *L is generated by a right regular grammar.*

(3) *L is generated by a left regular grammar.*

(4) *There exists a DFA A such that $L = L(A)$.*

(5) *There exists an NFA B such that $L = L(B)$.*

Proof.

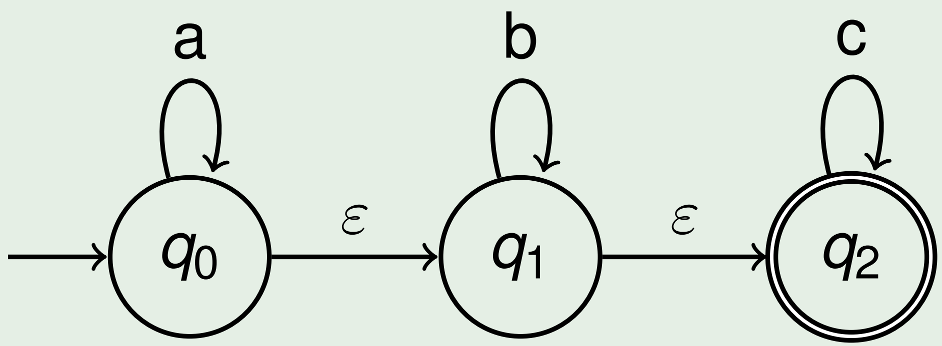
(2) $\rightarrow$ (5): Theorem 2.16.
(5) $\rightarrow$ (4): Theorem 2.15.
(4) $\rightarrow$ (2): Theorem 2.9.
(3) $\rightarrow$ (5): From a left regular grammar for L, we obtain a right regular grammar for L^R by reversing the right-hand sides of all productions.
The above and Theorem 2.17 yield an NFA for $(L^R)^R = L$.
The remaining cases follow analogously. □

Definition 2.19

A nondeterministic finite-state automaton with ε-transitions (ε-NFA) A is given through a 5-tuple $A = (Q, \Sigma, \delta, S, F)$, where Q, Σ, S, and F are defined as for an NFA, while the transition relation δ has the form $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to 2^Q$.

Example:

Let A be the ε-NFA that is given by the following state graph:



Then $L(A) = \{ a^i b^j c^k \mid i, j, k \geq 0 \}$.

Theorem 2.20

From an ε-NFA A, a DFA B can be constructed such that $L(B) = L(A)$.

Proof.

Let $A = (Q, \Sigma, \delta, S, F)$ be an ε-NFA. For $q \in Q$, ε-closure$(q) \subseteq Q$ denotes the set of states of A that can be reached from state q without reading any input symbol, that is,

$$\begin{array}{rcl} \varepsilon\text{-closure}(q) & := & \{\, p \in Q \mid \exists n \geq 0\, \exists\, p_0, p_1, \ldots, p_n \in Q : p_0 = q, \\ & & p_n = p, \text{ and } p_{i+1} \in \delta(p_i, \varepsilon),\ i = 0, 1, \ldots, n-1 \,\}. \end{array}$$

Further, for $P \subseteq Q$, ε-closure$(P) := \bigcup_{q \in P} \varepsilon$-closure$(q)$.

We define the DFA $B := (2^Q, \Sigma, \delta_B, q_0, G)$ through a power set construction:

$$\begin{array}{lrcl} - & q_0 & := & S, \\ - & G & := & \{\, P \subseteq Q \mid \varepsilon\text{-closure}(P) \cap F \neq \emptyset \,\}, \\ - & \delta_B(P, a) & := & \delta(\varepsilon\text{-closure}(P), a) \text{ for all } P \subseteq Q \text{ and } a \in \Sigma. \end{array}$$

Proof of Theorem 2.20 (cont.)

Claim.

$L(B) = L(A)$.

Proof.

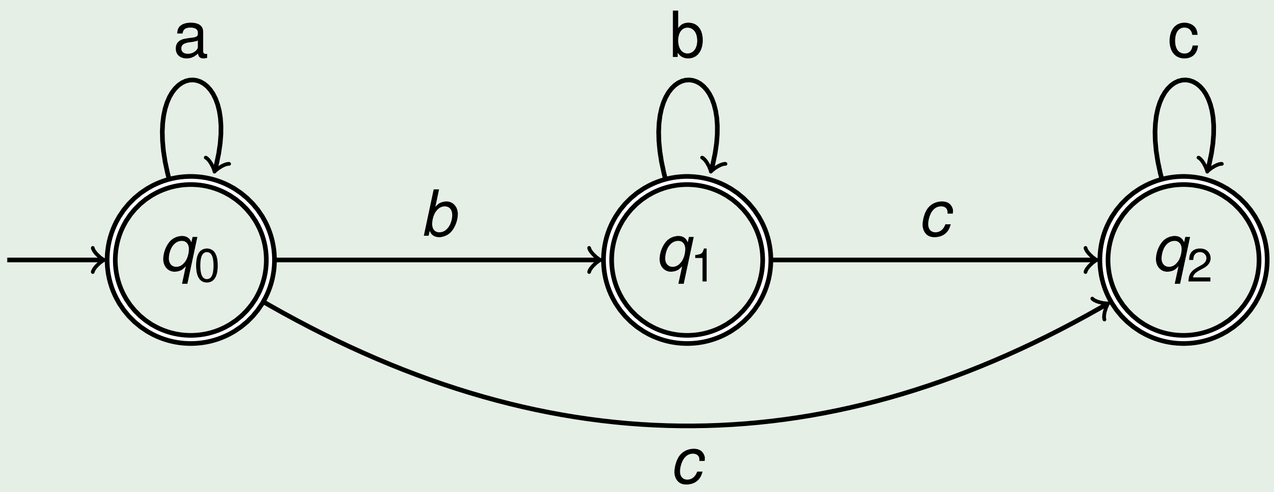
$\varepsilon \in L(A)$ iff ε-closure$(S) \cap F \neq \emptyset$ iff $q_0 \in G$ iff $\varepsilon \in L(B)$.

$$
\begin{array}{ll}
a_1 a_2 \ldots a_n \in L(A) & \text{iff} \quad \exists s \in S \, \exists s_1, q_1, p_1, q_2, p_2, \ldots, q_n, p_n \in Q : \\
\multicolumn{2}{l}{s \to^{\varepsilon^*} s_1 \to^{a_1} q_1 \to^{\varepsilon^*} p_1 \to^{a_2} q_2 \to^{\varepsilon^*} \cdots \to^{\varepsilon^*} p_{n-1} \to^{a_n} q_n \to^{\varepsilon^*} p_n \in F} \\
& \text{iff} \quad \exists Q_1, Q_2, \ldots, Q_n \subseteq Q : \\
& \qquad Q_1 = \delta(\varepsilon\text{-closure}(S), a_1) \wedge Q_n \in G \wedge \\
& \qquad Q_{i+1} = \delta(\varepsilon\text{-closure}(Q_i), a_{i+1}) \; (1 \leq i \leq n-1) \\
& \text{iff} \quad a_1 a_2 \ldots a_n \in L(B).
\end{array}
$$

□

Example (cont.):

The 'lazy' form of this construction yields the following DFA from the given ε-NFA A:



Theorem 2.21

The language class $\mathcal{L}(\text{DFA})$ *is closed under the operations of union, product, and star.*

Proof.

Union: Let $A_i = (Q_i, \Sigma, \delta_i, S_i, F_i)$, $i = 1, 2$, be two NFA s.t. $Q_1 \cap Q_2 = \emptyset$. Then $A := (Q_1 \cup Q_2, \Sigma, \delta, S_1 \cup S_2, F_1 \cup F_2)$, where

$$\delta(q, a) = \begin{cases} \delta_1(q, a), & \text{if } q \in Q_1, \\ \delta_2(q, a), & \text{if } q \in Q_2, \end{cases},$$

accepts the language $L(A) = L(A_1) \cup L(A_2)$.

Proof of Theorem 2.21 (cont.)

Product: Let $A_i = (Q_i, \Sigma, \delta_i, S_i, F_i)$, $i = 1, 2$, be two ε-NFA s.t. $Q_1 \cap Q_2 = \emptyset$.

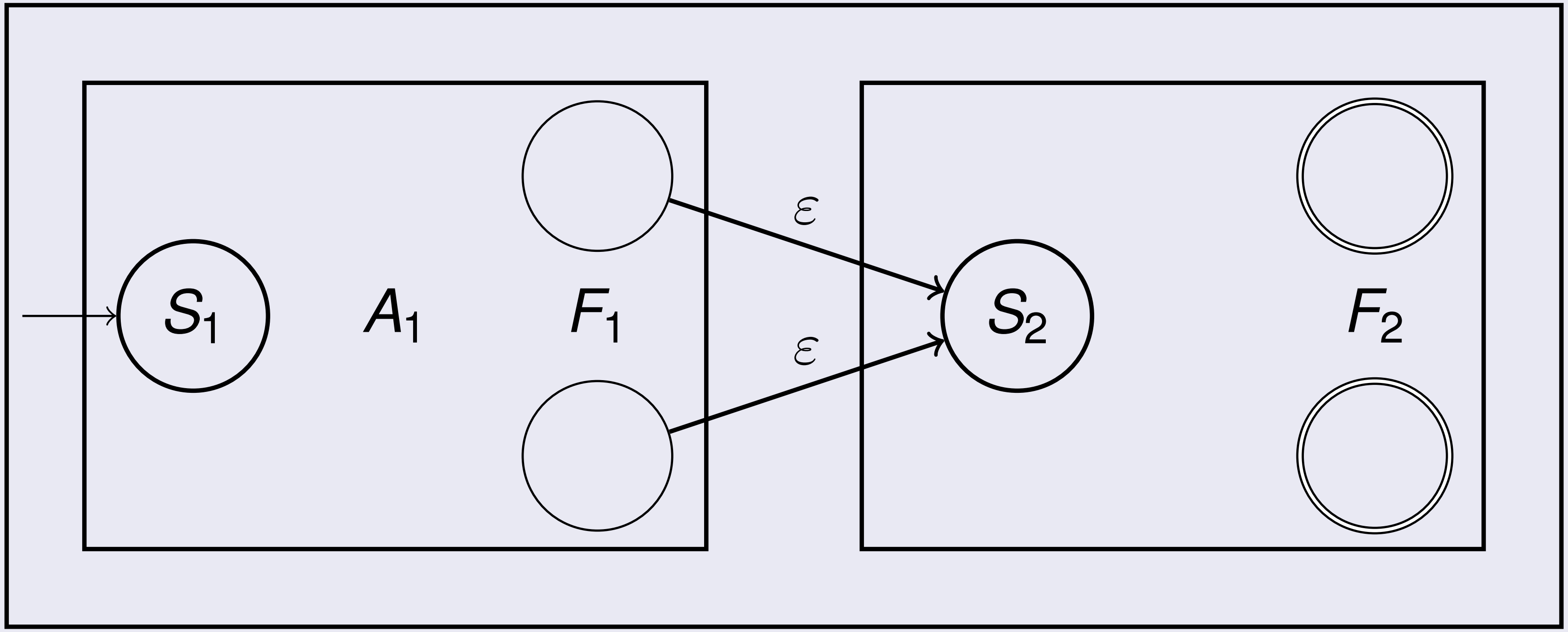
Define $A := (Q_1 \cup Q_2, \Sigma, \delta, S_1, F_2)$, where

$$\begin{array}{rcll} \delta(q, a) & = & \delta_1(q, a) & \text{for all } q \in Q_1 \text{ and } a \in \Sigma, \\ \delta(q, \varepsilon) & = & \begin{cases} \delta_1(q, \varepsilon) & \text{for all } q \in Q_1 \smallsetminus F_1, \\ \delta_1(q, \varepsilon) \cup S_2 & \text{for all } q \in F_1, \end{cases} & \\ \delta(q, a) & = & \delta_2(q, a) & \text{for all } q \in Q_2 \text{ and } a \in \Sigma \cup \{\varepsilon\}. \end{array}$$

Then A is an ε-NFA satisfying $L(A) = L(A_1) \cdot L(A_2)$.

Proof of Theorem 2.21 (cont.)

Graphical representation of A:



Proof of Theorem 2.21 (cont.)

Kleene Star:

Let $A_1 = (Q_1, \Sigma, \delta_1, S_1, F_1)$ be an ε-NFA.

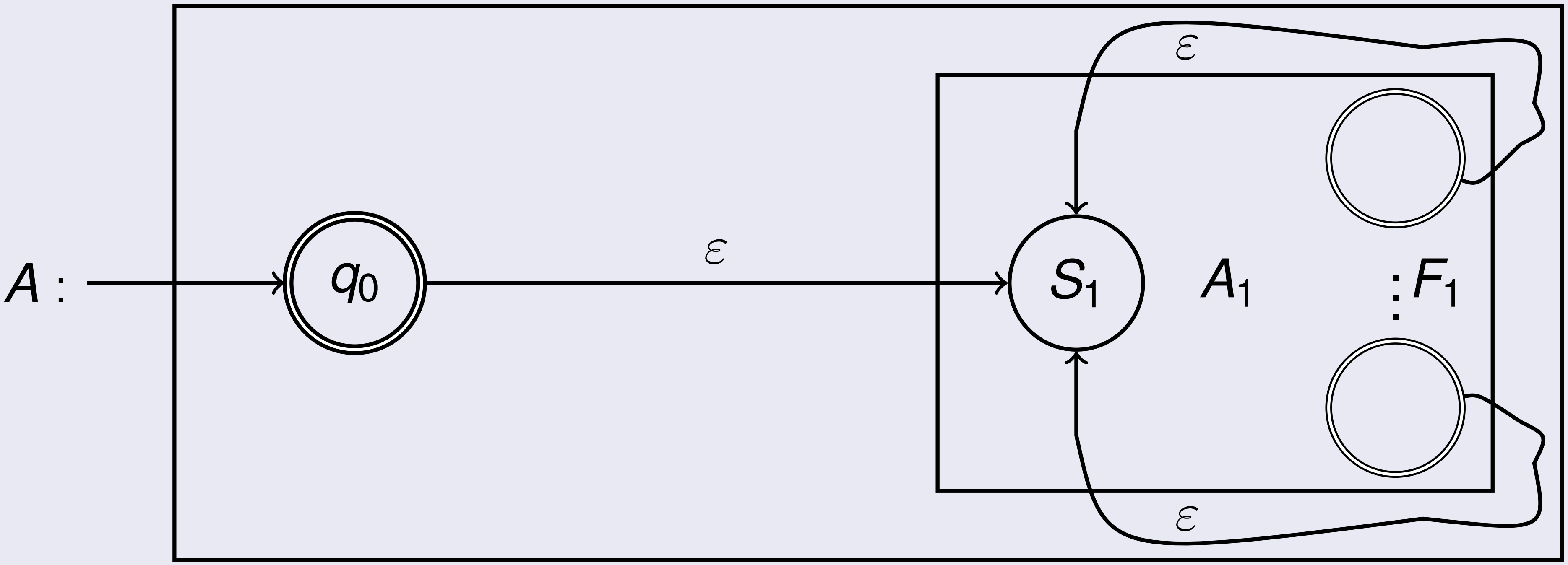
Define $A = (Q_1 \cup \{q_0\}, \Sigma, \delta, \{q_0\}, F_1 \cup \{q_0\})$, where

$$\begin{array}{rcll} \delta(q_0, \varepsilon) & := & S_1, & \\ \delta(q, \varepsilon) & := & \delta_1(q, \varepsilon) \cup S_1 & \text{for all } q \in F_1, \\ \delta(q, \varepsilon) & := & \delta_1(q, \varepsilon) & \text{for all } q \in Q_1 \smallsetminus F_1, \\ \delta(q, a) & := & \delta_1(q, a) & \text{for all } q \in Q_1 \text{ and } a \in \Sigma. \end{array}$$

Then A is an ε-NFA satisfying $L(A) = (L(A_1))^*$.

Proof of Theorem 2.21 (cont.)

Graphical representation of A:



□

2.5 Two-Way Finite-State Automaton

Definition 2.22

A deterministic two-way finite-state automaton (2DFA) *A is defined through a 7-tuple* $A = (Q, \Sigma, \rhd, \lhd, \delta, q_0, F)$*, where*

– Q, Σ, q_0*, and F are defined as for a DFA,*

– $\rhd, \lhd \notin \Sigma$ *are two new letters that serve as border markers for the left and the right end of the tape, and*

$$\delta : Q \times (\Sigma \cup \{\rhd, \lhd\}) \to Q \times \{L, R\}$$

is the transition function satisfying the following restrictions:

$$\forall q \in Q : \delta(q, \rhd) \neq (q', L) \text{ and } \delta(q, \lhd) \neq (q', R).$$

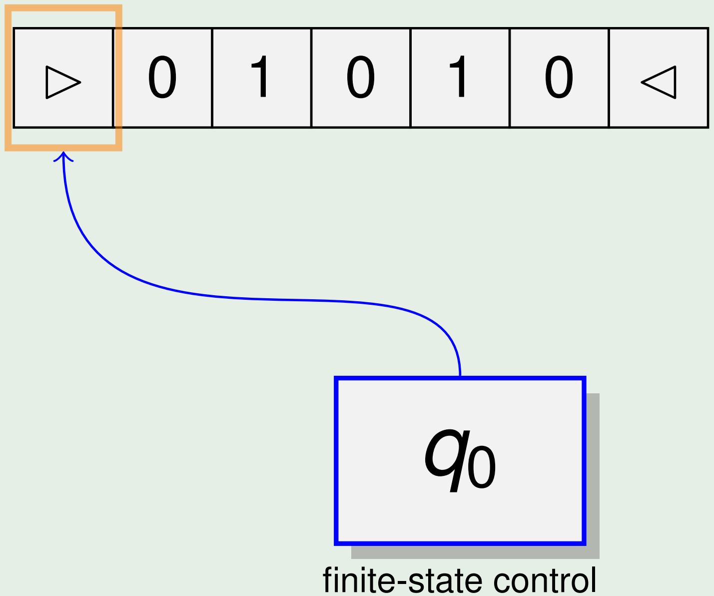
These restrictions ensure that A's head cannot fall off the tape.

Example:

Let $A = (\{q_0, q_1, \ldots, q_k, q_+\}, \{0, 1\}, \triangleright, \triangleleft, \delta, q_0, \{q_+\})$:

δ	$\triangleright$	0	1	$\triangleleft$
q_0	(q_0, R)	(q_0, R)	(q_0, R)	(q_1, L)
q_1	$-$	(q_2, L)	(q_2, L)	$-$
q_2	$-$	(q_3, L)	(q_3, L)	$-$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
q_{k-1}	$-$	(q_k, L)	(q_k, L)	$-$
q_k	$-$	(q_+, R)	$-$	$-$
q_+	$-$	(q_+, R)	(q_+, R)	$-$

Let $k = 3$:

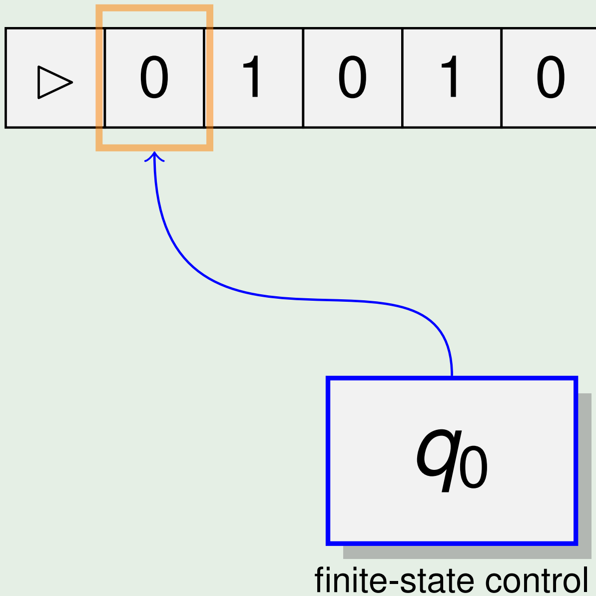


Example:

Let $A = (\{q_0, q_1, \ldots, q_k, q_+\}, \{0, 1\}, \triangleright, \triangleleft, \delta, q_0, \{q_+\})$:

δ	$\triangleright$	0	1	$\triangleleft$
q_0	(q_0, R)	(q_0, R)	(q_0, R)	(q_1, L)
q_1	$-$	(q_2, L)	(q_2, L)	$-$
q_2	$-$	(q_3, L)	(q_3, L)	$-$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
q_{k-1}	$-$	(q_k, L)	(q_k, L)	$-$
q_k	$-$	(q_+, R)	$-$	$-$
q_+	$-$	(q_+, R)	(q_+, R)	$-$

Let $k = 3$:

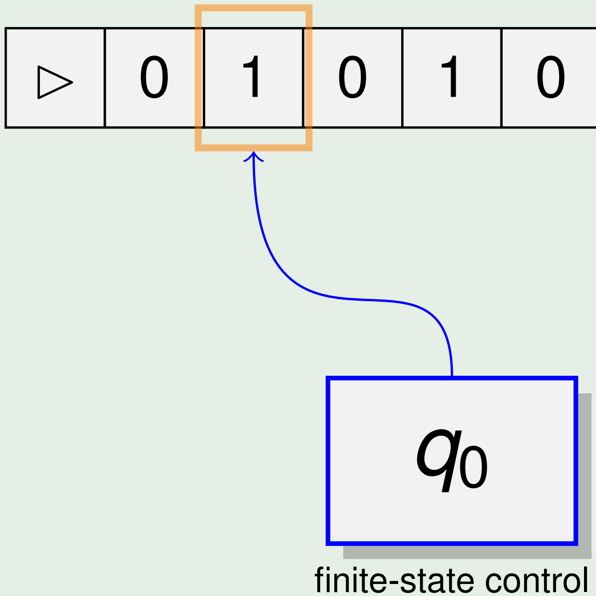


Example:

Let $A = (\{q_0, q_1, \ldots, q_k, q_+\}, \{0, 1\}, \rhd, \lhd, \delta, q_0, \{q_+\})$:

δ	$\rhd$	0	1	$\lhd$
q_0	(q_0, R)	(q_0, R)	(q_0, R)	(q_1, L)
q_1	$-$	(q_2, L)	(q_2, L)	$-$
q_2	$-$	(q_3, L)	(q_3, L)	$-$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
q_{k-1}	$-$	(q_k, L)	(q_k, L)	$-$
q_k	$-$	(q_+, R)	$-$	$-$
q_+	$-$	(q_+, R)	(q_+, R)	$-$

Let $k = 3$:

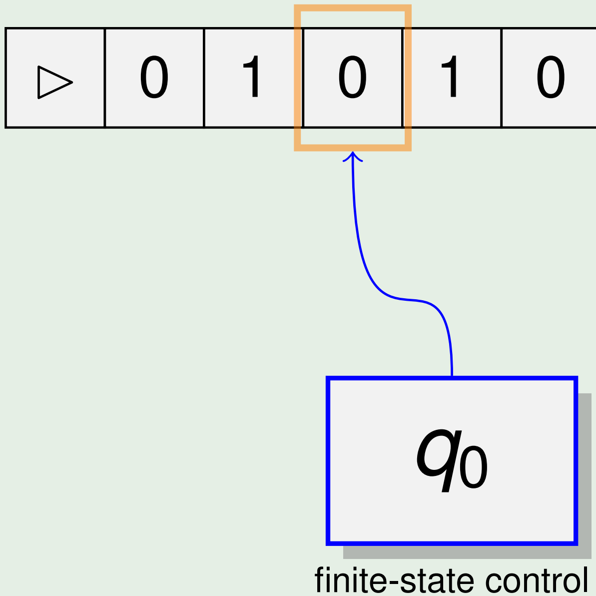


Example:

Let $A = (\{q_0, q_1, \ldots, q_k, q_+\}, \{0, 1\}, \triangleright, \triangleleft, \delta, q_0, \{q_+\})$:

δ	$\triangleright$	0	1	$\triangleleft$
q_0	(q_0, R)	(q_0, R)	(q_0, R)	(q_1, L)
q_1	$-$	(q_2, L)	(q_2, L)	$-$
q_2	$-$	(q_3, L)	(q_3, L)	$-$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
q_{k-1}	$-$	(q_k, L)	(q_k, L)	$-$
q_k	$-$	(q_+, R)	$-$	$-$
q_+	$-$	(q_+, R)	(q_+, R)	$-$

Let $k = 3$:

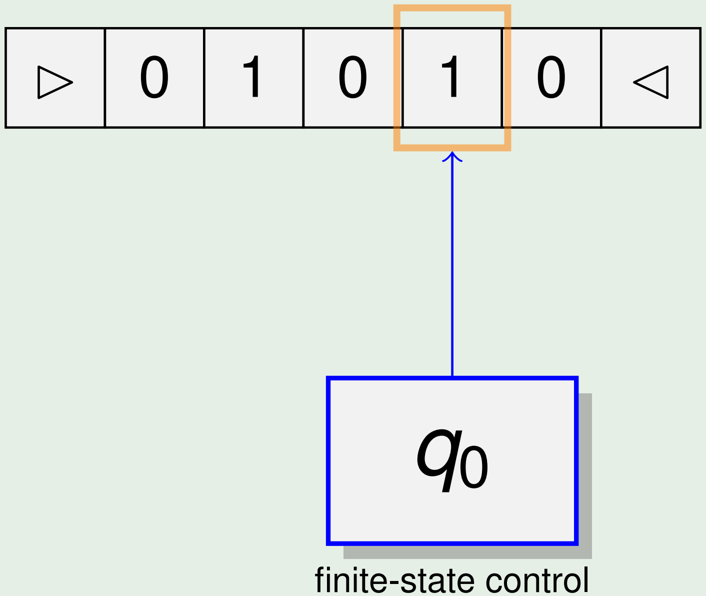


Example:

Let $A = (\{q_0, q_1, \ldots, q_k, q_+\}, \{0, 1\}, \triangleright, \triangleleft, \delta, q_0, \{q_+\})$:

δ	$\triangleright$	0	1	$\triangleleft$
q_0	(q_0, R)	(q_0, R)	(q_0, R)	(q_1, L)
q_1	–	(q_2, L)	(q_2, L)	–
q_2	–	(q_3, L)	(q_3, L)	–
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
q_{k-1}	–	(q_k, L)	(q_k, L)	–
q_k	–	(q_+, R)	–	–
q_+	–	(q_+, R)	(q_+, R)	–

Let $k = 3$:

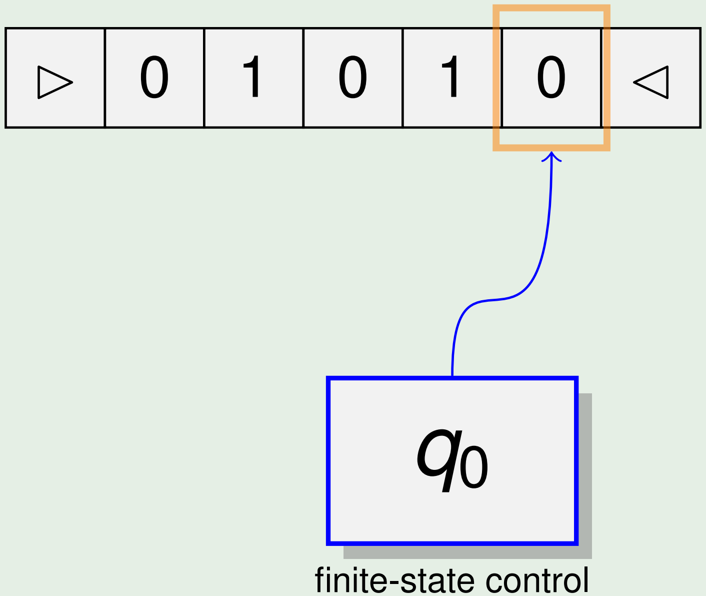


Example:

Let $A = (\{q_0, q_1, \ldots, q_k, q_+\}, \{0,1\}, \triangleright, \triangleleft, \delta, q_0, \{q_+\})$:

δ	$\triangleright$	0	1	$\triangleleft$
q_0	(q_0, R)	(q_0, R)	(q_0, R)	(q_1, L)
q_1	$-$	(q_2, L)	(q_2, L)	$-$
q_2	$-$	(q_3, L)	(q_3, L)	$-$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
q_{k-1}	$-$	(q_k, L)	(q_k, L)	$-$
q_k	$-$	(q_+, R)	$-$	$-$
q_+	$-$	(q_+, R)	(q_+, R)	$-$

Let $k = 3$:

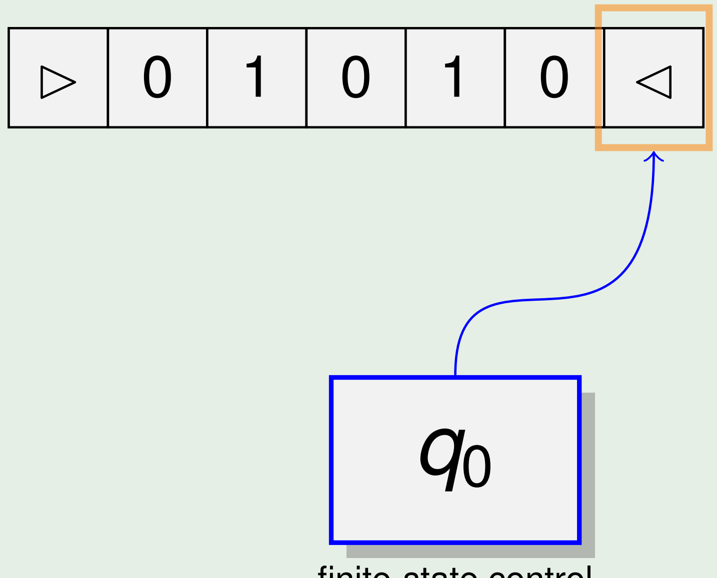


Example:

Let $A = (\{q_0, q_1, \ldots, q_k, q_+\}, \{0,1\}, \triangleright, \triangleleft, \delta, q_0, \{q_+\})$:

δ	$\triangleright$	0	1	$\triangleleft$
q_0	(q_0, R)	(q_0, R)	(q_0, R)	(q_1, L)
q_1	$-$	(q_2, L)	(q_2, L)	$-$
q_2	$-$	(q_3, L)	(q_3, L)	$-$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
q_{k-1}	$-$	(q_k, L)	(q_k, L)	$-$
q_k	$-$	(q_+, R)	$-$	$-$
q_+	$-$	(q_+, R)	(q_+, R)	$-$

Let $k = 3$:

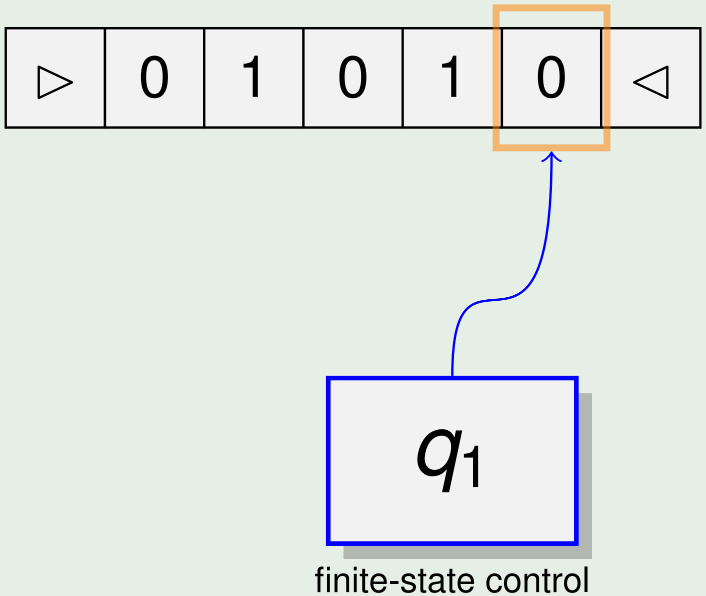


Example:

Let $A = (\{q_0, q_1, \ldots, q_k, q_+\}, \{0,1\}, \triangleright, \triangleleft, \delta, q_0, \{q_+\})$:

δ	$\triangleright$	0	1	$\triangleleft$
q_0	(q_0, R)	(q_0, R)	(q_0, R)	(q_1, L)
q_1	$-$	(q_2, L)	(q_2, L)	$-$
q_2	$-$	(q_3, L)	(q_3, L)	$-$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
q_{k-1}	$-$	(q_k, L)	(q_k, L)	$-$
q_k	$-$	(q_+, R)	$-$	$-$
q_+	$-$	(q_+, R)	(q_+, R)	$-$

Let $k = 3$:

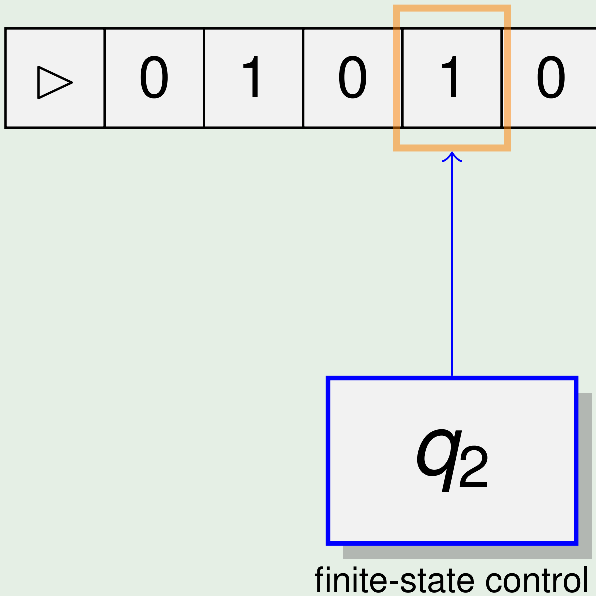


Example:

Let $A = (\{q_0, q_1, \ldots, q_k, q_+\}, \{0, 1\}, \rhd, \lhd, \delta, q_0, \{q_+\})$:

δ	$\rhd$	0	1	$\lhd$
q_0	(q_0, R)	(q_0, R)	(q_0, R)	(q_1, L)
q_1	$-$	(q_2, L)	(q_2, L)	$-$
q_2	$-$	(q_3, L)	(q_3, L)	$-$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
q_{k-1}	$-$	(q_k, L)	(q_k, L)	$-$
q_k	$-$	(q_+, R)	$-$	$-$
q_+	$-$	(q_+, R)	(q_+, R)	$-$

Let $k = 3$:

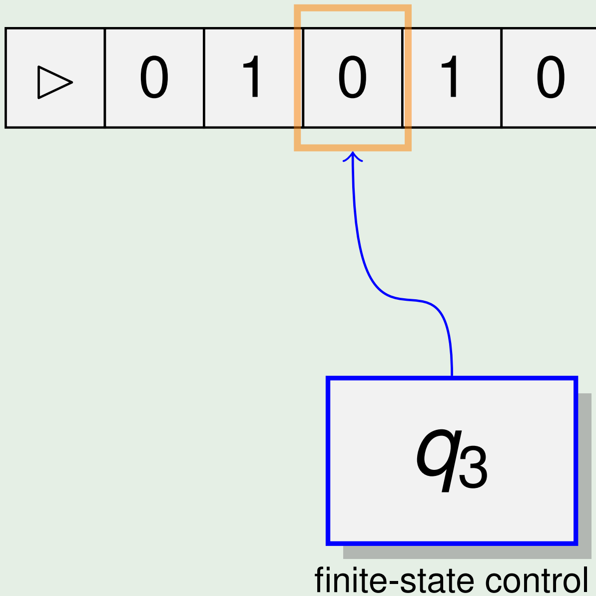


Example:

Let $A = (\{q_0, q_1, \ldots, q_k, q_+\}, \{0, 1\}, \triangleright, \triangleleft, \delta, q_0, \{q_+\})$:

δ	$\triangleright$	0	1	$\triangleleft$
q_0	(q_0, R)	(q_0, R)	(q_0, R)	(q_1, L)
q_1	–	(q_2, L)	(q_2, L)	–
q_2	–	(q_3, L)	(q_3, L)	–
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
q_{k-1}	–	(q_k, L)	(q_k, L)	–
q_k	–	(q_+, R)	–	–
q_+	–	(q_+, R)	(q_+, R)	–

Let $k = 3$:

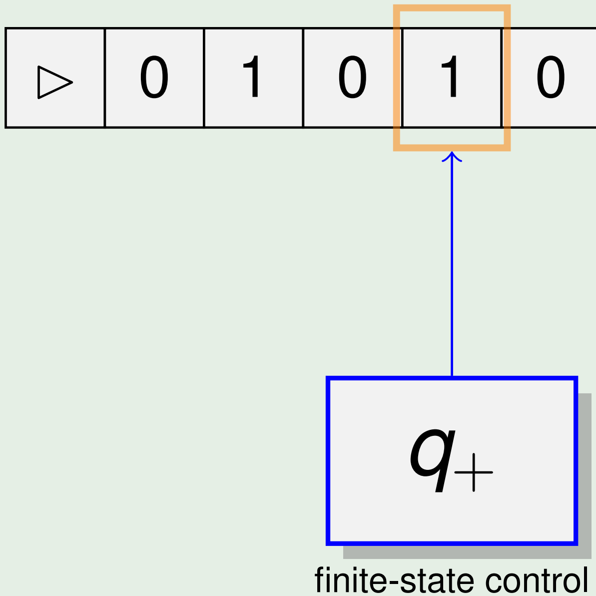


Example:

Let $A = (\{q_0, q_1, \ldots, q_k, q_+\}, \{0, 1\}, \triangleright, \triangleleft, \delta, q_0, \{q_+\})$:

δ	$\triangleright$	0	1	$\triangleleft$
q_0	(q_0, R)	(q_0, R)	(q_0, R)	(q_1, L)
q_1	$-$	(q_2, L)	(q_2, L)	$-$
q_2	$-$	(q_3, L)	(q_3, L)	$-$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
q_{k-1}	$-$	(q_k, L)	(q_k, L)	$-$
q_k	$-$	(q_+, R)	$-$	$-$
q_+	$-$	(q_+, R)	(q_+, R)	$-$

Let $k = 3$:

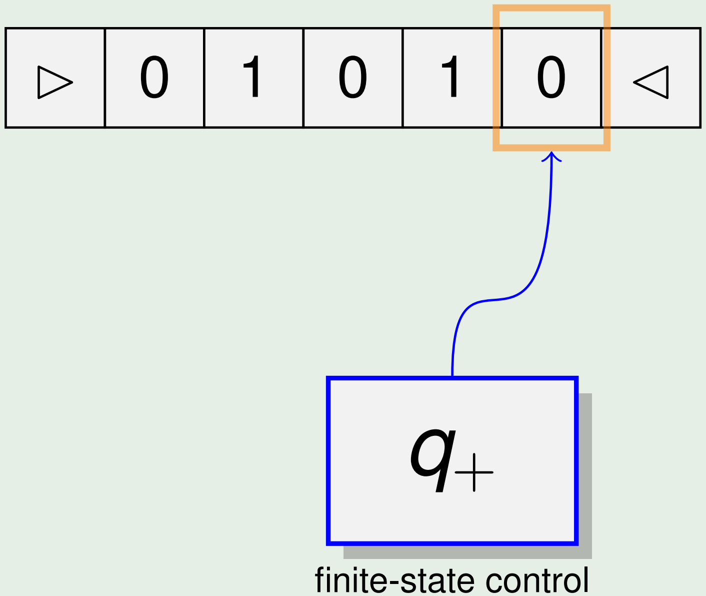


Example:

Let $A = (\{q_0, q_1, \ldots, q_k, q_+\}, \{0, 1\}, \triangleright, \triangleleft, \delta, q_0, \{q_+\})$:

δ	$\triangleright$	0	1	$\triangleleft$
q_0	(q_0, R)	(q_0, R)	(q_0, R)	(q_1, L)
q_1	$-$	(q_2, L)	(q_2, L)	$-$
q_2	$-$	(q_3, L)	(q_3, L)	$-$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
q_{k-1}	$-$	(q_k, L)	(q_k, L)	$-$
q_k	$-$	(q_+, R)	$-$	$-$
q_+	$-$	(q_+, R)	(q_+, R)	$-$

Let $k = 3$:

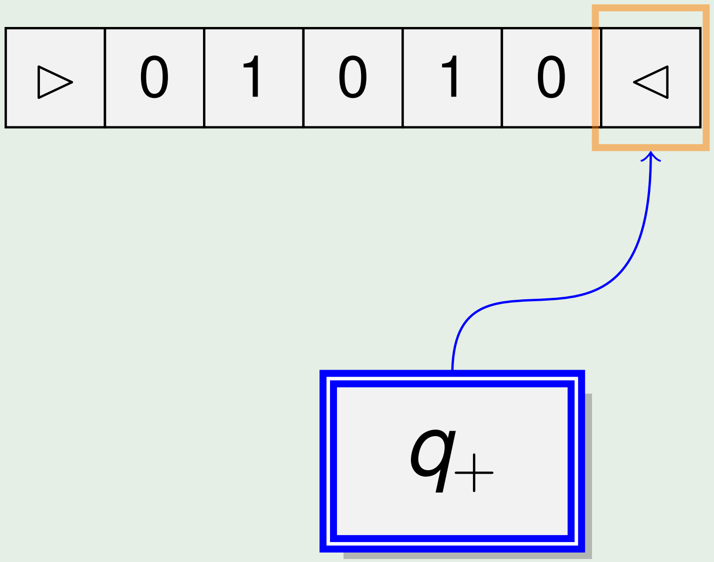


Example:

Let $A = (\{q_0, q_1, \ldots, q_k, q_+\}, \{0, 1\}, \triangleright, \triangleleft, \delta, q_0, \{q_+\})$:

δ	$\triangleright$	0	1	$\triangleleft$
q_0	(q_0, R)	(q_0, R)	(q_0, R)	(q_1, L)
q_1	$-$	(q_2, L)	(q_2, L)	$-$
q_2	$-$	(q_3, L)	(q_3, L)	$-$
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
q_{k-1}	$-$	(q_k, L)	(q_k, L)	$-$
q_k	$-$	(q_+, R)	$-$	$-$
q_+	$-$	(q_+, R)	(q_+, R)	$-$

Let $k = 3$:



Definition 2.22 (cont.)

A configuration of the 2DFA A is described by a word of the form

$$q \rhd w \lhd \text{ or } \rhd w_1 q w_2 \lhd,$$

where $q \in Q$ and $w, w_1, w_2 \in \Sigma^*$.

The transition function δ induces a computation relation $\vdash_A^*$ on the set of configurations, which is the reflexive and transitive closure of the following single-step computation relation $\vdash_A$:

$$\rhd a_1 \ldots a_{i-1} q a_i \ldots a_n \lhd \vdash \begin{cases} \rhd a_1 \ldots a_{i-2} q' a_{i-1} \ldots a_n \lhd, & \text{if } \delta(q, a_i) = (q', L), \\ \rhd a_1 \ldots a_i q' a_{i+1} \ldots a_n \lhd, & \text{if } \delta(q, a_i) = (q', R). \end{cases}$$

The initial configuration for input $w \in \Sigma^*$ is $q_0 \rhd w \lhd$, and a configuration of the form $\rhd w q \lhd$, where $q \in F$, is an accepting configuration.

W.l.o.g. we can assume that $\delta(q, \lhd)$ is undefined for all $q \in F$.

Definition 2.22 (cont.)

The language $L(A)$ accepted by A is defined as follows:

$$L(A) = \{ w \in \Sigma^* \mid q_0 \rhd w \lhd \vdash_A^* \rhd wq \lhd \text{ for some } q \in F \},$$

and $\mathcal{L}$(2DFA) is the class of languages accepted by 2DEAs.

Example (cont.):

Let $k = 3$. On input $w = 01010$, A executes the following computation:

$$\begin{array}{llllll} q_0 \rhd 01010 \lhd & \vdash_A & \rhd q_0 01010 \lhd & \vdash_A^5 & \rhd 01010 q_0 \lhd \\ & \vdash_A & \rhd 0101 q_1 0 \lhd & \vdash_A & \rhd 010 q_2 10 \lhd \\ & \vdash_A & \rhd 01 q_3 010 \lhd & \vdash_A & \rhd 010 q_+ 10 \lhd \\ & \vdash_A^2 & \rhd 01010 q_+ \lhd, & & \end{array}$$

that is, A accepts on input $w = 01010$.

In fact, it is easily seen that $L(A)$ is the language

$$L_k = \{ x = x_1 \ldots x_n \in \{0,1\}^* \mid |x| = n \geq k \text{ and } x_{n-k+1} = 0 \}.$$

Example:

Let $A = (\{q_0, q_1, q_2, q_3\}, \{0, 1\}, \triangleright, \triangleleft, \delta, q_0, \{q_1, q_2, q_3\})$, where δ is given through the following table:

	$\triangleright$	0	1	$\triangleleft$
q_0	(q_1, R)	–	–	–
q_1	–	(q_1, R)	(q_2, R)	–
q_2	–	(q_2, R)	(q_3, L)	–
q_3	–	(q_1, R)	(q_3, L)	–

On input $w = 101001$, A executes the following computation:

$$\begin{array}{lllll} q_0 \triangleright 101001 \triangleleft & \vdash_A & \triangleright q_1 101001 \triangleleft & \vdash_A & \triangleright 1 q_2 01001 \triangleleft \\ & \vdash_A & \triangleright 10 q_2 1001 \triangleleft & \vdash_A & \triangleright 1 q_3 01001 \triangleleft \\ & \vdash_A & \triangleright 10 q_1 1001 \triangleleft & \vdash_A & \triangleright 101 q_2 001 \triangleleft \\ & \vdash_A^2 & \triangleright 10100 q_2 1 \triangleleft & \vdash_A & \triangleright 1010 q_3 01 \triangleleft \\ & \vdash_A & \triangleright 10100 q_1 1 \triangleleft & \vdash_A & \triangleright 101001 q_2 \triangleleft, \end{array}$$

that is, A accepts on input $w = 101001$.

To describe the behaviour of a 2DFA $A = (Q, \Sigma, \triangleright, \triangleleft, \delta, q_0, F)$, we introduce the notion of a crossing sequence.

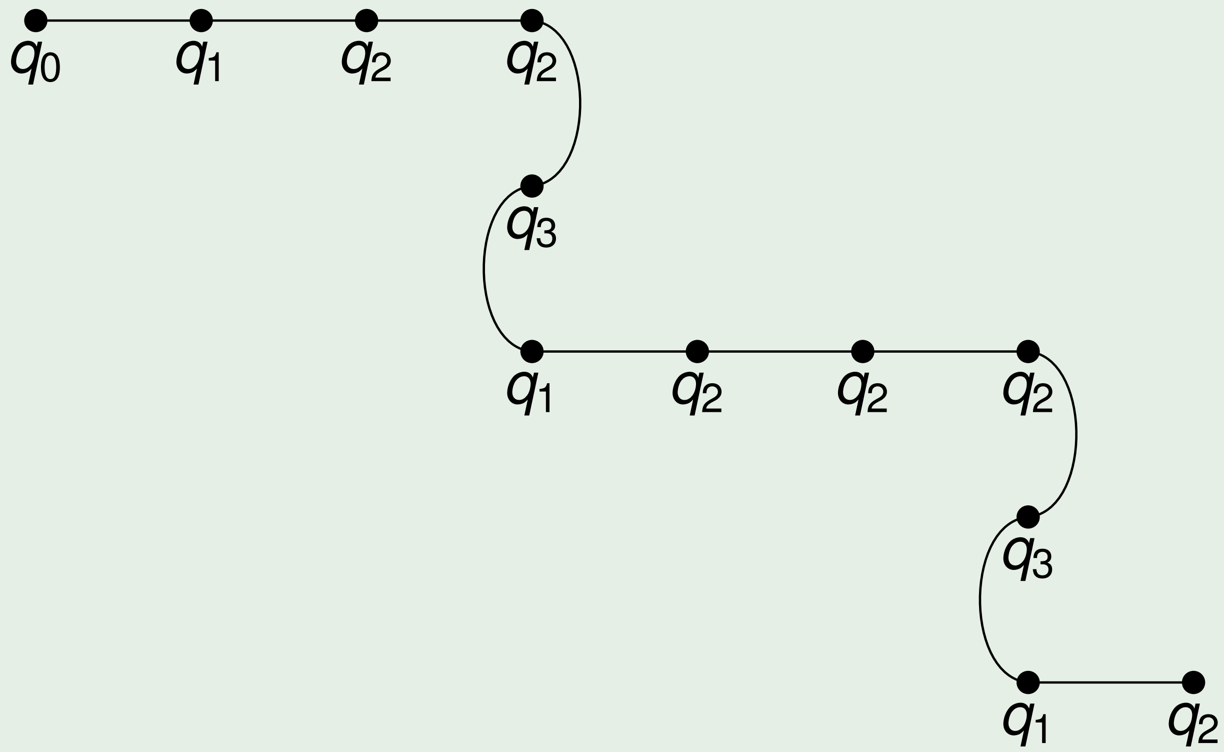
We consider a computation of A on an input $w \in \Sigma^*$, observing the path taken by the head of A. Each time the head crosses the boundary between two tape squares we note down the new state underneath the corresponding boundary. In this way we obtain a sequence of states for each boundary. Such a sequence is called the crossing sequence (CS) of A on input w at the current position.

Example (cont.):

The 2DFA A executes the above computation on input $w = 101001$, which yields the following graphical representation:

Example (cont.):

▷	1	0	1	0	0	1	◁



Then (q_1) is the CS between ▷ and 101001 ◁,
(q_2, q_3, q_1) is the CS between ▷10 and 1001 ◁,
and (q_2, q_3, q_1) is also the CS between ▷10100 and 1 ◁.

Lemma 2.23

Let $(q_{i_1}, \ldots, q_{i_m})$ be a CS of a 2DFA A on input $w \in \Sigma^*$ between $\triangleright w_1$ and $w_2 \triangleleft$. Then the following statements hold:

(a) *In state q_{i_1}, A performs a move-right step from the last letter of $\triangleright w_1$ to the first letter of $w_2 \triangleleft$.*

(b) *In states q_{i_j}, where $j \equiv 1 \bmod 2$, A performs move-right steps, while in states q_{i_j}, where $j \equiv 0 \bmod 2$, it performs move-left steps.*

(c) *If $w \in L(A)$, then each CS of A on input w has odd length.*

(d) *If $w \in L(A)$, then $q_{i_j} \neq q_{i_{j+2r}}$ for all $j \geq 1$ and all $r \geq 1$.*

(c) holds, as A starts on the left delimiter $\triangleright$ and accepts on the right delimiter $\triangleleft$, and (d) holds, as $q_{i_j} = q_{i_{j+2r}}$ would imply that A has reached a cycle within its computation, that is, it will not terminate.

A sequence of states $(q_1, \ldots, q_k)$ is called a valid CS, if k is odd, all states with even index are pairwise distinct, and also all states with odd index are pairwise distinct.

If $s = |Q|$, then a valid CS has length at most $2s - 1$. Hence, there are only finitely many valid CSs for A.

Theorem 2.24

A language is regular iff it is accepted by a 2DFA, that is, $\text{REG} = \mathcal{L}(\text{2DFA})$*.*

Proof.

We construct an NFA B from a given 2DFA A such that $L(A) = L(B)$.

The states of B will correspond to the valid CSs of A.

For this construction we need to be able to check whether neighbouring CSs are consistent with each other and with the actual input symbol.

Proof of Theorem 2.24 (cont.)

Let $(q_1, q_2, \dots, q_k)$ be the left CS of a tape square containing the letter $a \in \Sigma$, and let $(p_1, p_2, \dots, p_\ell)$ be the right CS of that tape square.

We define right-matching and left-matching pairs of CSs:

1. The empty sequence left- and right-matches the empty sequence.
2. If $(q_3, \dots, q_k)$ right-matches $(p_1, \dots, p_\ell)$ and $\delta(q_1, a) = (q_2, L)$, then $(q_1, q_2, q_3, \dots, q_k)$ right-matches $(p_1, \dots, p_\ell)$.
3. If $(q_2, \dots, q_k)$ left-matches $(p_2, \dots, p_\ell)$ and $\delta(q_1, a) = (p_1, R)$, then $(q_1, q_2, \dots, q_k)$ right-matches $(p_1, p_2, \dots, p_\ell)$.
4. If $(q_1, \dots, q_k)$ left-matches $(p_3, \dots, p_\ell)$ and $\delta(p_1, a) = (p_2, R)$, then $(q_1, \dots, q_k)$ left-matches $(p_1, p_2, p_3, \dots, p_\ell)$.
5. If $(q_2, \dots, q_k)$ right-matches $(p_2, \dots, p_\ell)$ and $\delta(p_1, a) = (q_1, L)$, then $(q_1, q_2, \dots, q_k)$ left-matches $(p_1, p_2, \dots, p_\ell)$.

Example (cont.):

For the 2DFA A, we consider a tape square containing the letter 1. The empty sequence left-matches the empty sequence and $\delta(q_1, 1) = (q_2, R)$. Hence, by (3) (q_1) right-matches (q_2).
As $\delta(q_2, 1) = (q_3, L)$, (q_2, q_3, q_1) right-matches (q_2) by (2).

Proof of Theorem 2.24 (cont.)

Let $A = (Q, \Sigma, \triangleright, \triangleleft, \delta, q_0, F)$ and let $B = (Q', \Sigma \cup \{\triangleright, \triangleleft\}, \delta', \{q_0'\}, F')$ be the NFA that is defined as follows:

- Q' consists of all valid CSs of A;
- $q_0' = (q_0)$;
- F' consists of all valid CSs that end in a state from F;
- $\delta'((q_1, \ldots, q_k), a) = \{\, (p_1, \ldots, p_\ell) \mid (p_1, \ldots, p_\ell)$ is a valid CS such that $(q_1, \ldots, q_k)$ right-matches $(p_1, \ldots, p_\ell)$ for the input letter $a \,\}$ for all $a \in \Sigma \cup \{\triangleright, \triangleleft\}$.

Proof of Theorem 2.24 (cont.)

While B reads the word $\triangleright w \triangleleft$ (where $w \in \Sigma^*$) letter by letter from left to right, it guesses valid CSs of A and checks whether the current CS, the new CS, and the current letter are compatible with each other.

It can now be shown that $L(B) = \triangleright \cdot L(A) \cdot \triangleleft$. Hence, the language $\triangleright \cdot L(A) \cdot \triangleleft$ is regular, from which it can be concluded that $L(A)$ is regular. □

The 2DFA can be generalized to the nondeterministic two-way finite-state automaton (2NFA).

Theorem 2.25

A language is regular iff it is accepted by a 2NFA, that is, REG $= \mathcal{L}(\text{2NFA})$.