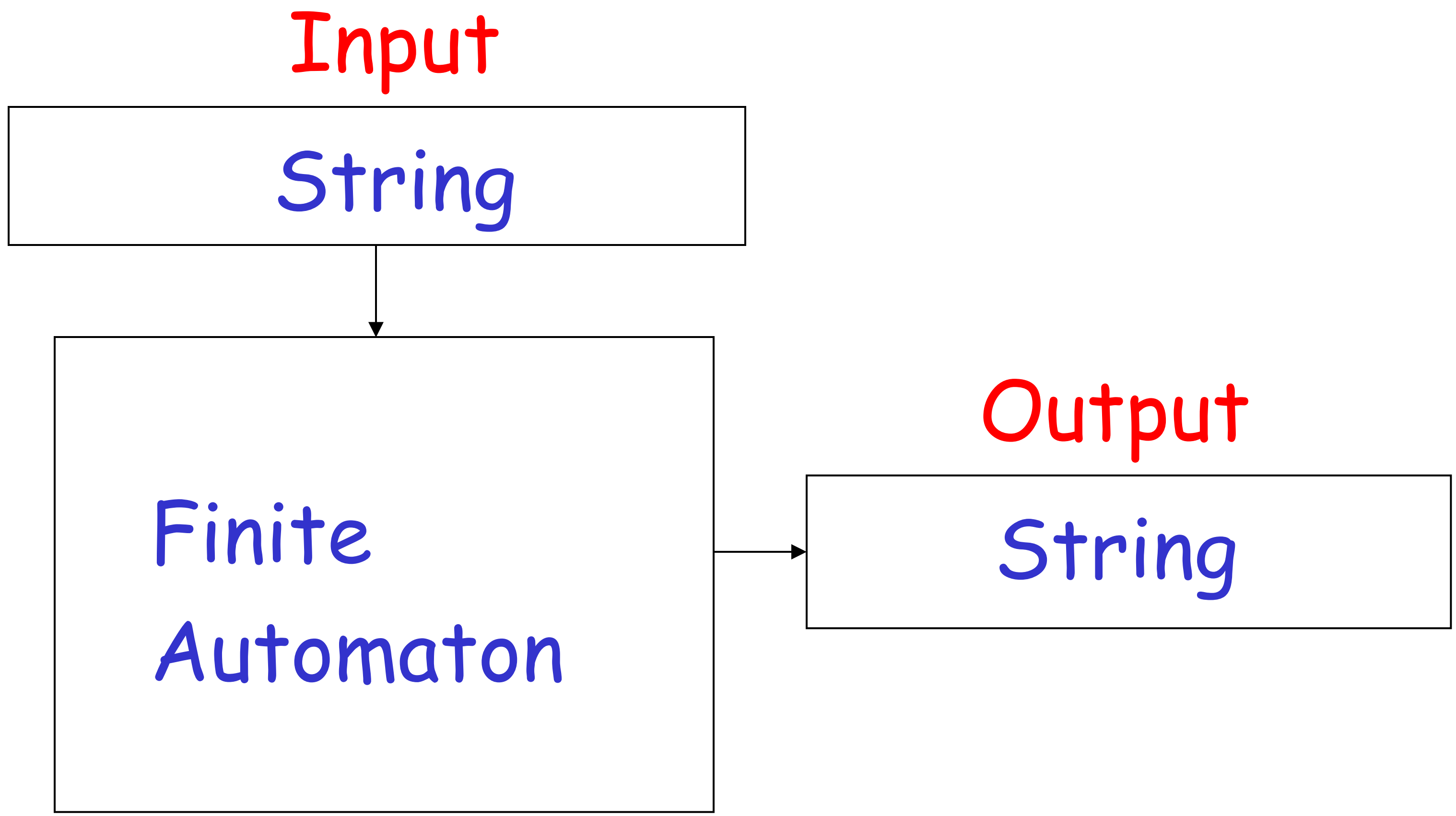
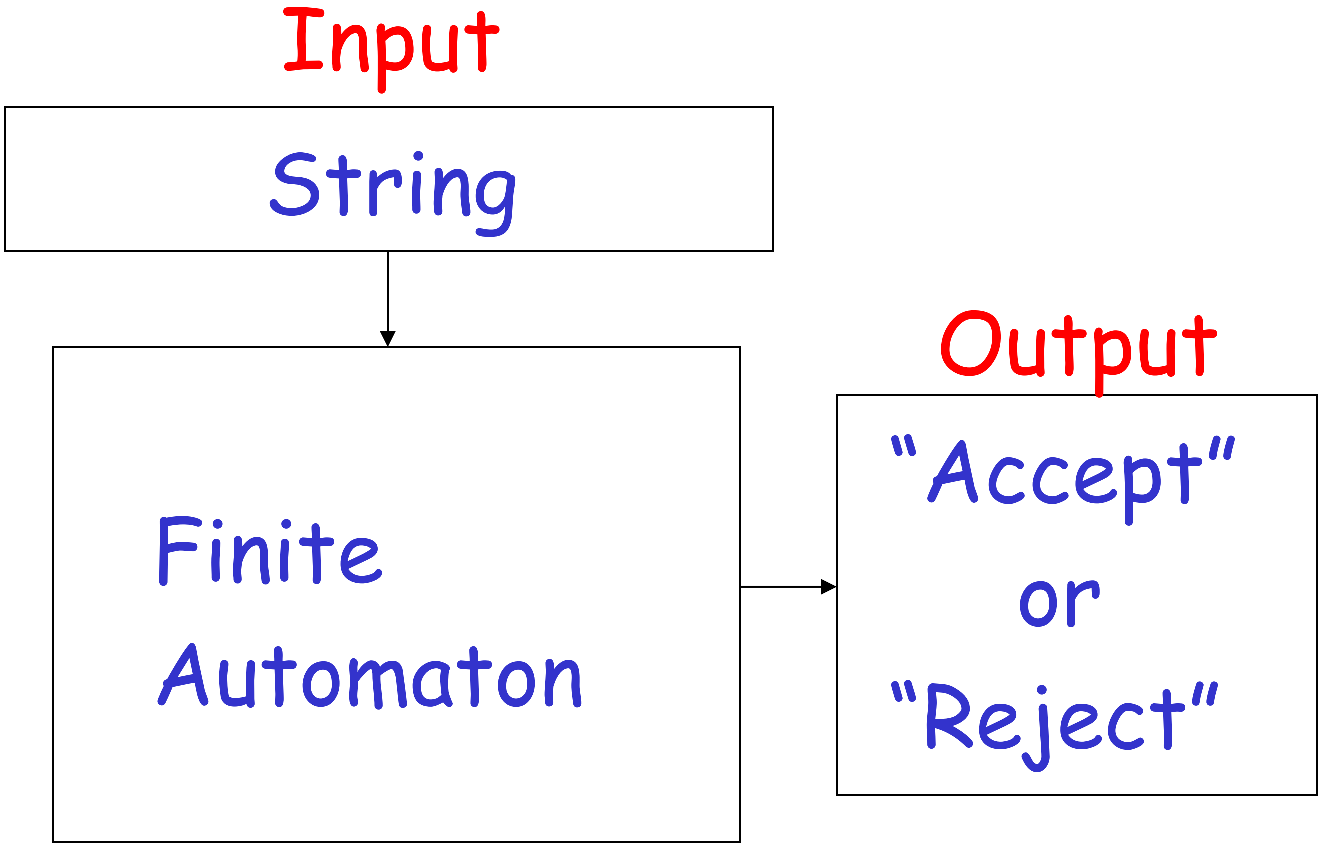
Finite Automata

Finite Automaton

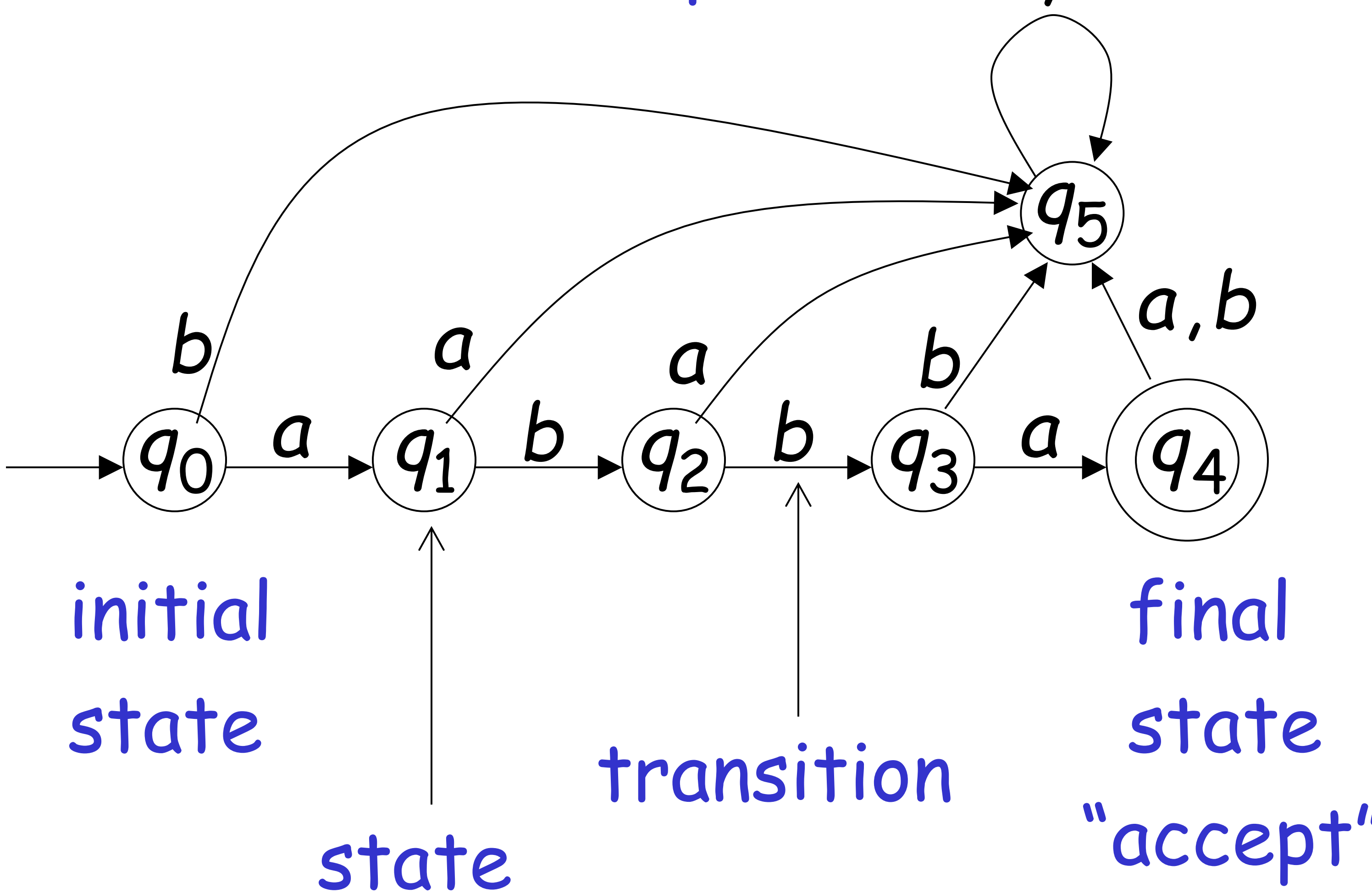


Finite Accepter

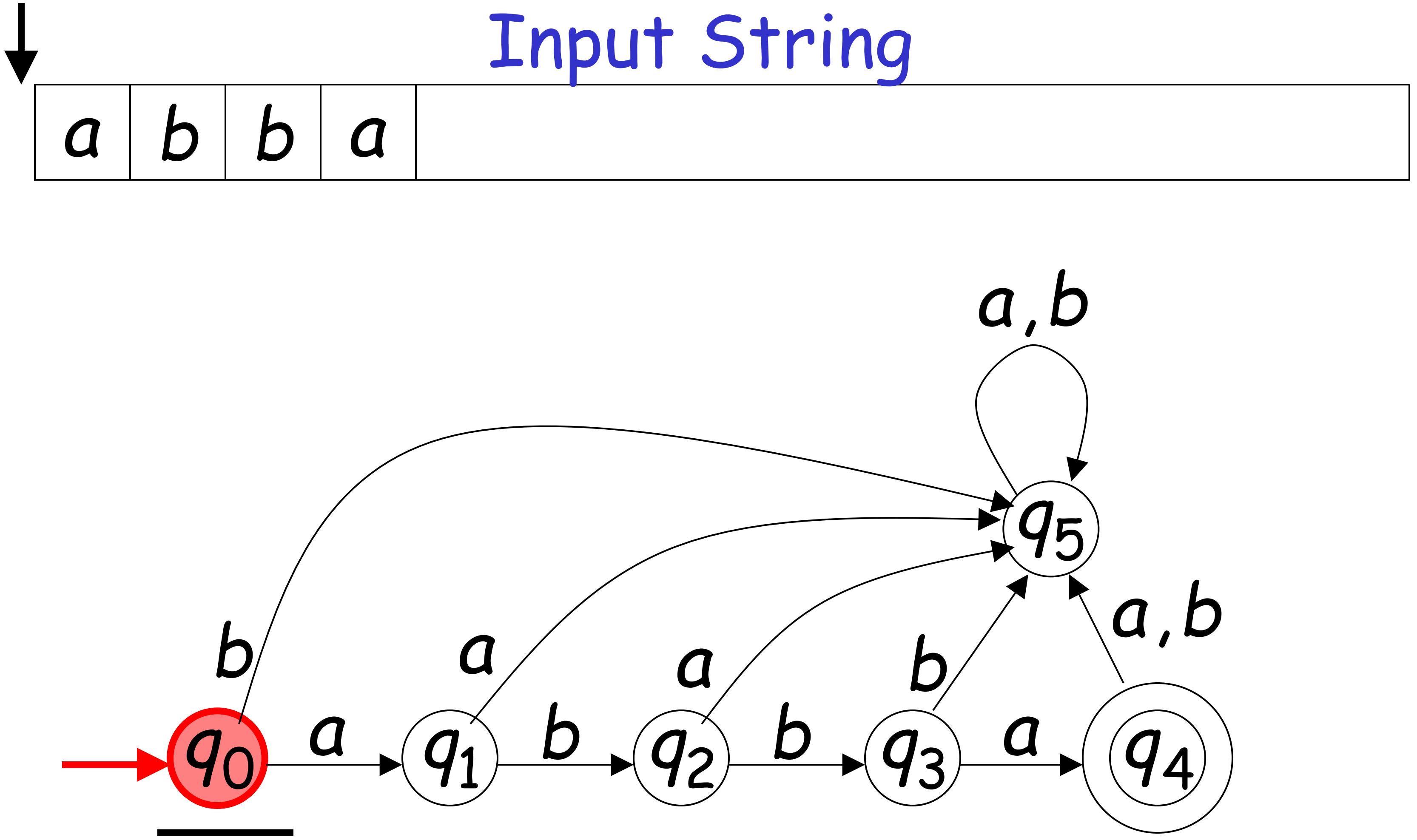


Transition Graph

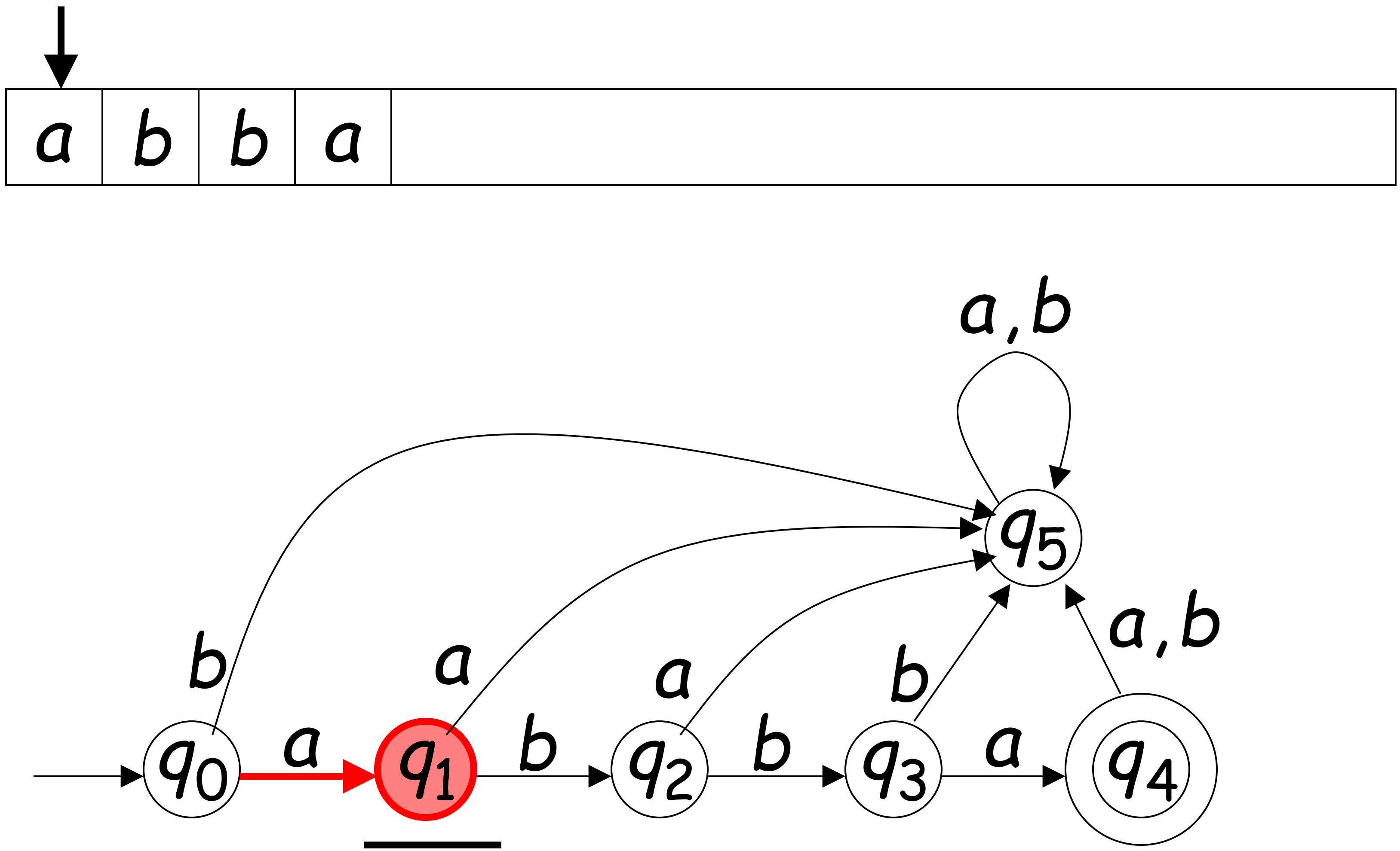
Abba -Finite Accepter

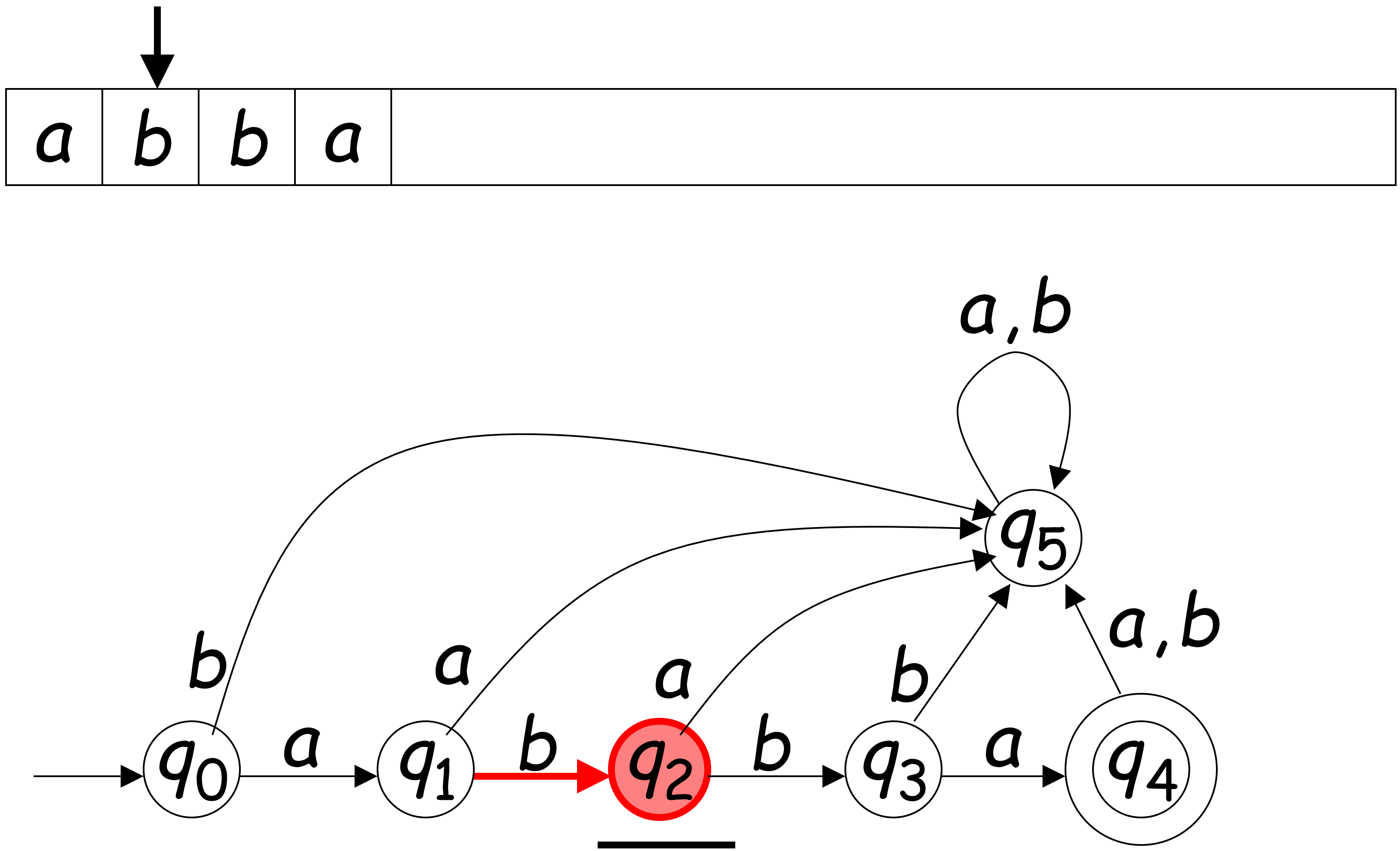


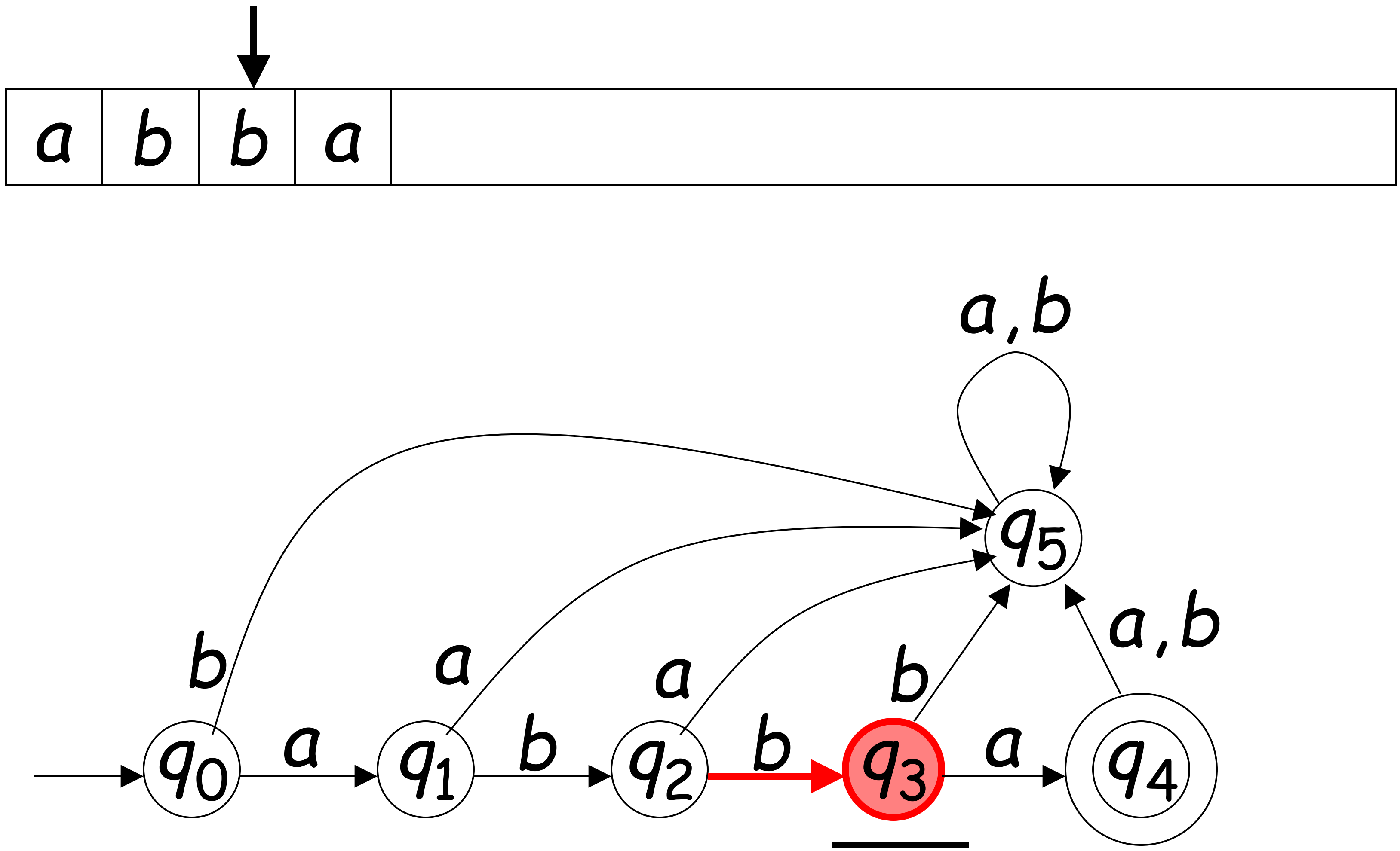
Initial Configuration



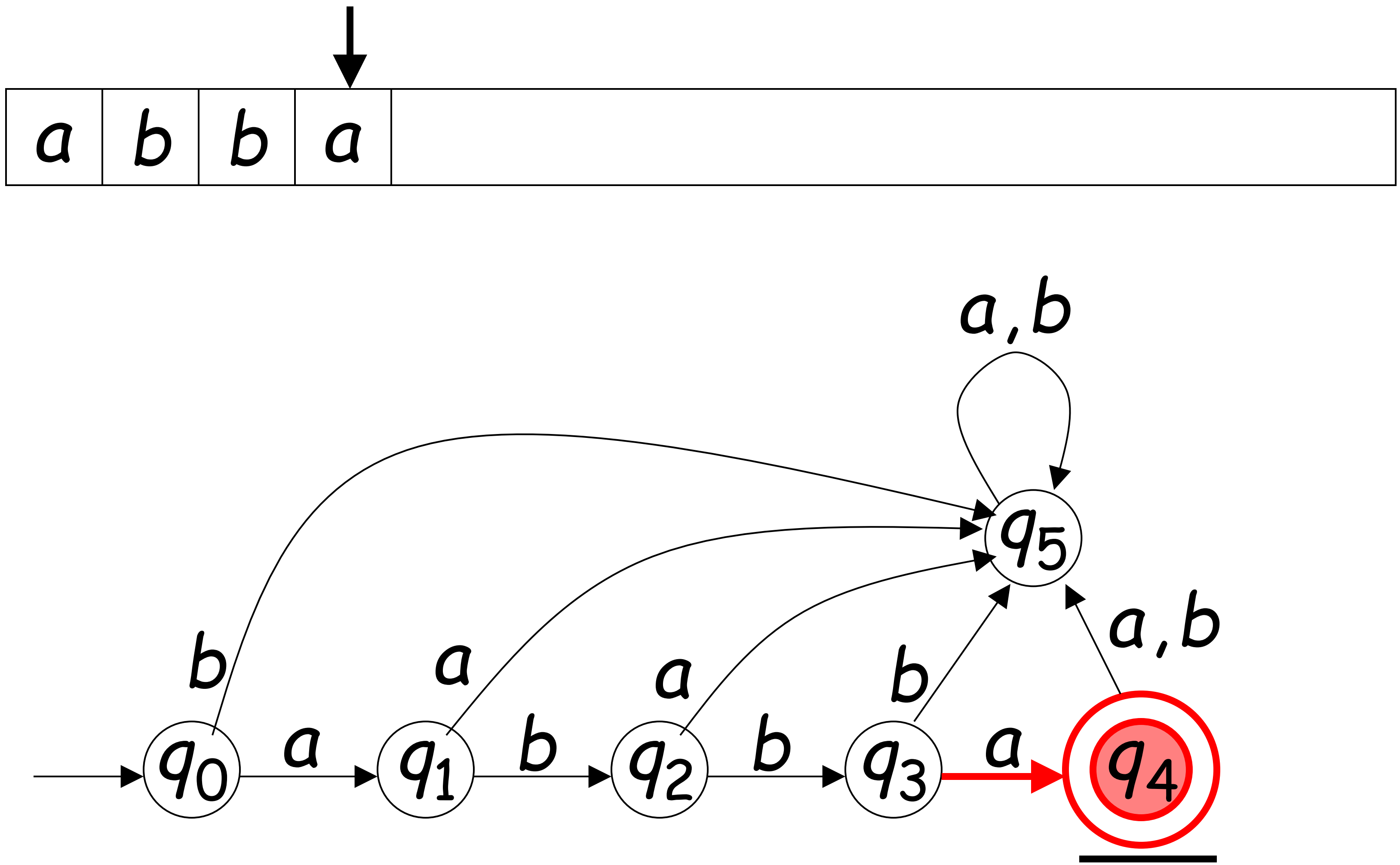
Reading the Input



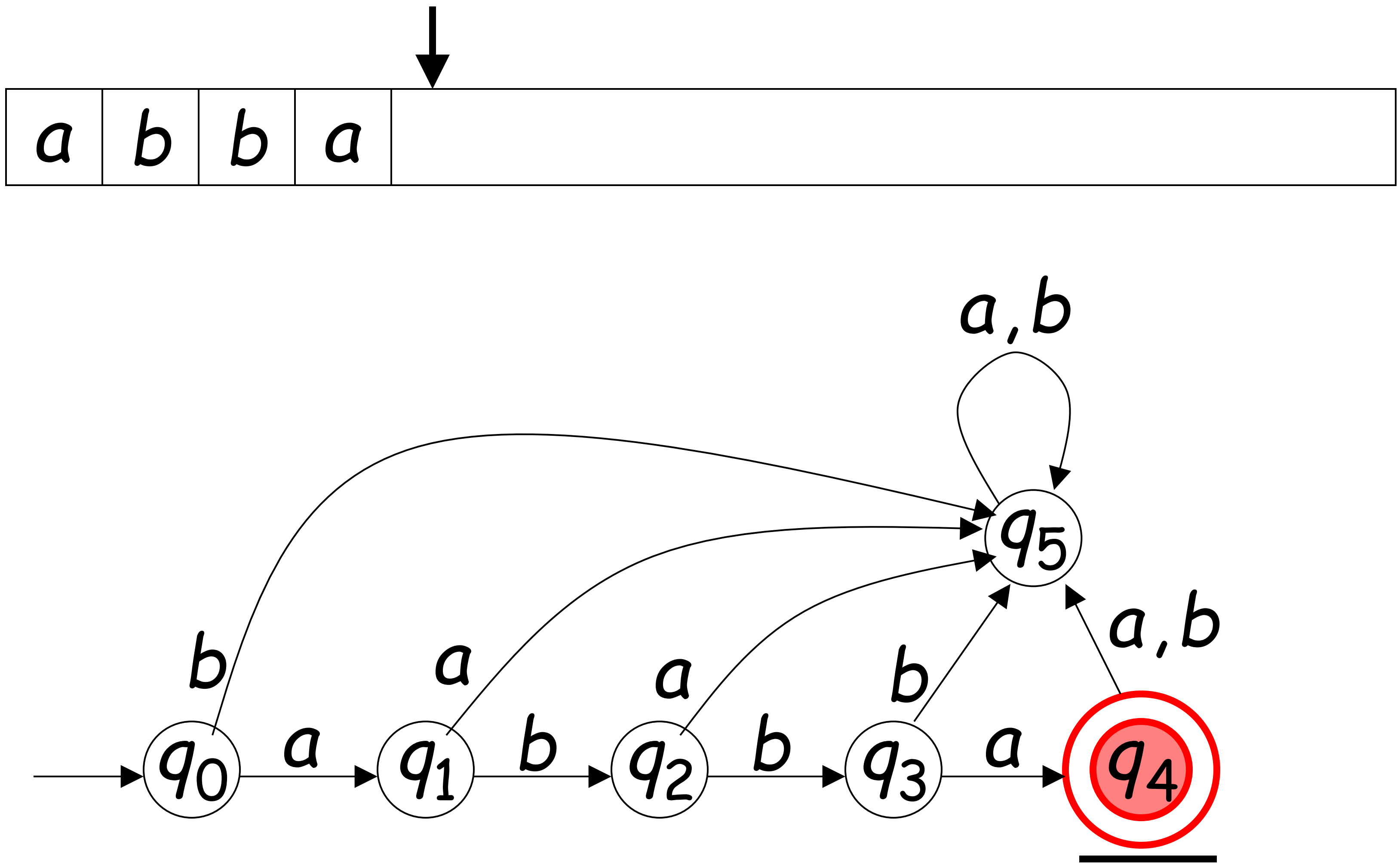

a
b
b
a
a,b
q5
b
a
a
b
a,b
q0
a
q1
b
q2
b
q3
a
q4



a
b
b
a
a,b
q5
b
a
a
b
a,b
q0
a
q1
b
q2
b
q3
a
q4

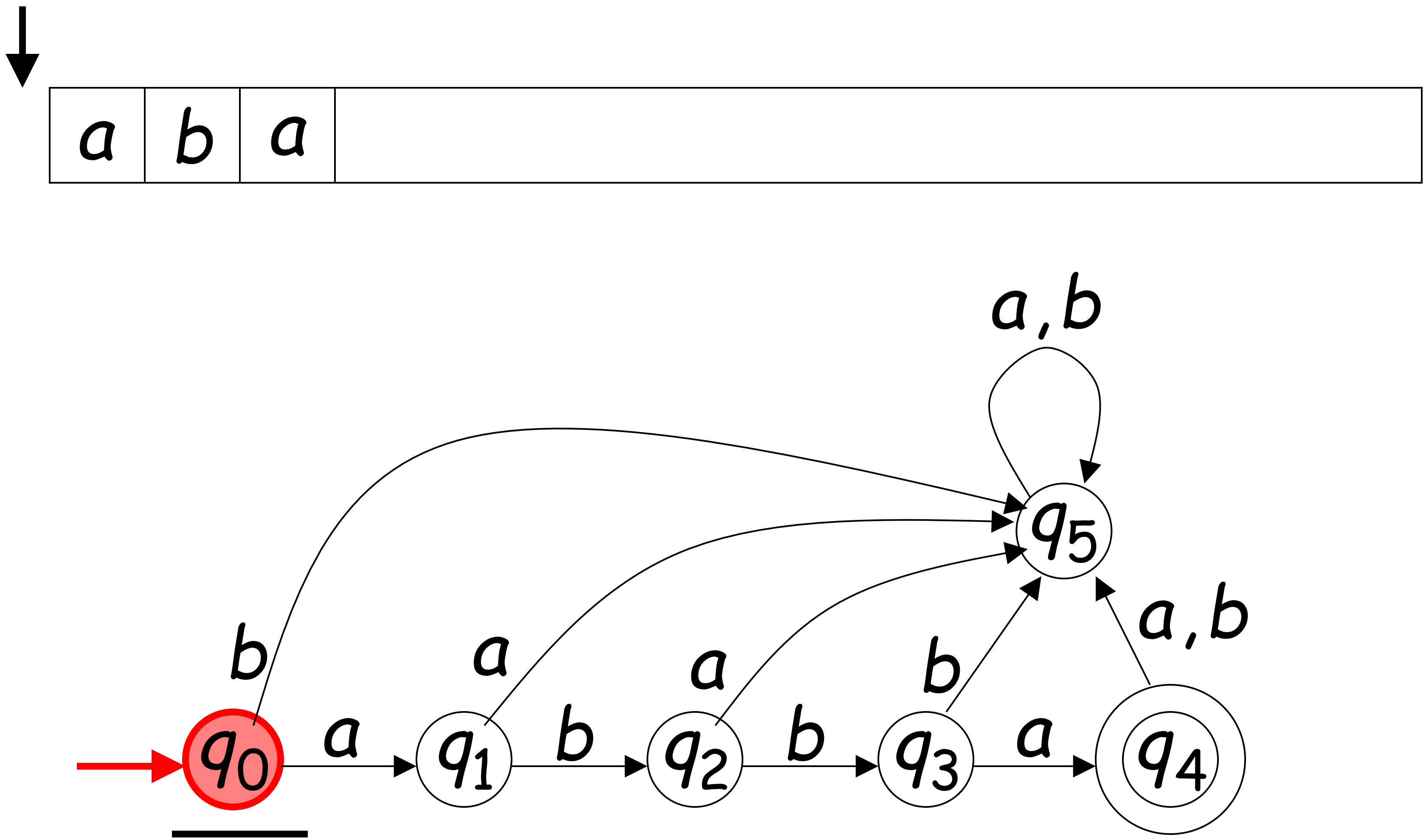

a
b
b
a
a, b
b
a
a
b
a
b
b
a
a, b
q_0
q_1
q_2
q_3
q_4
q_5

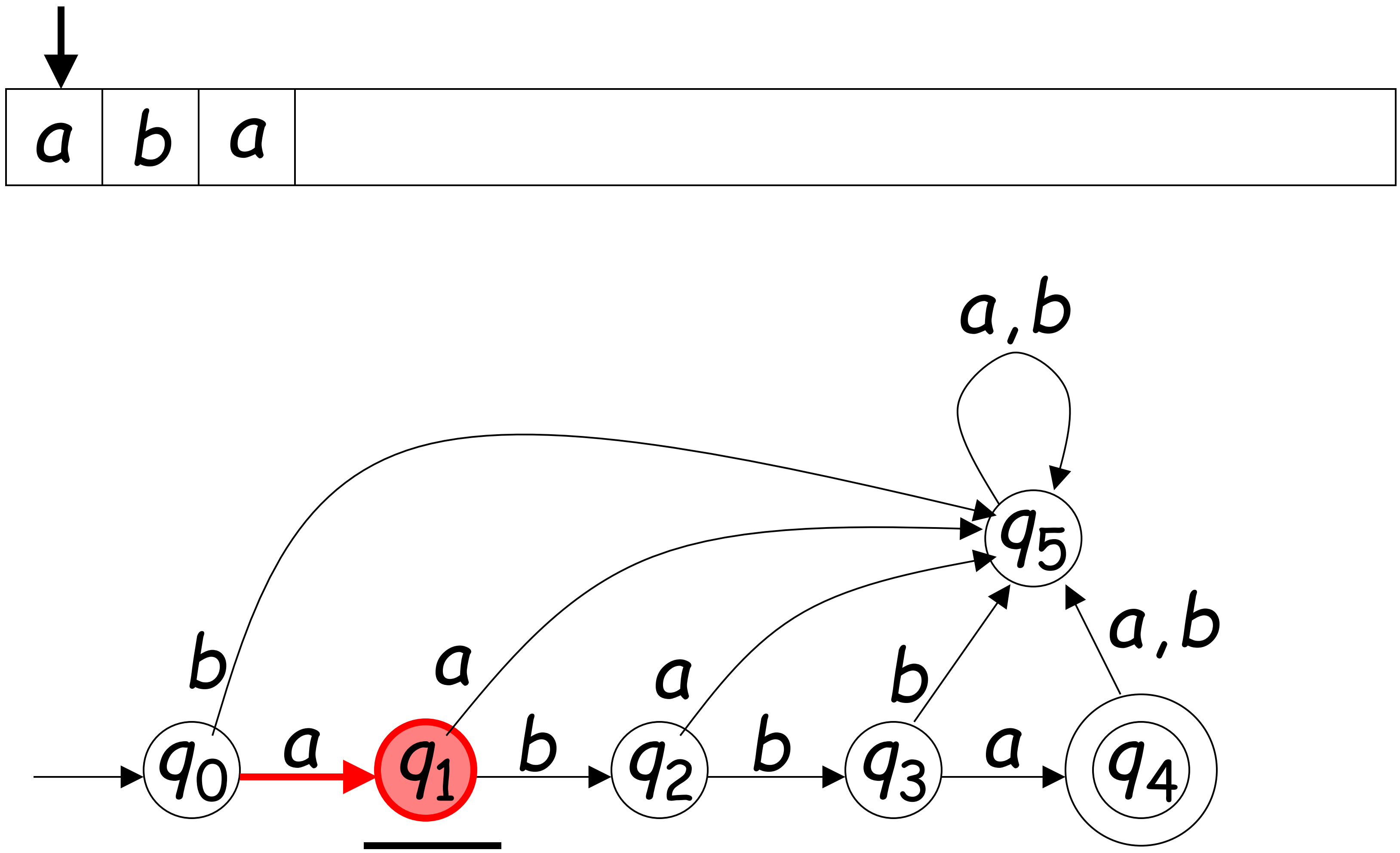
Input finished



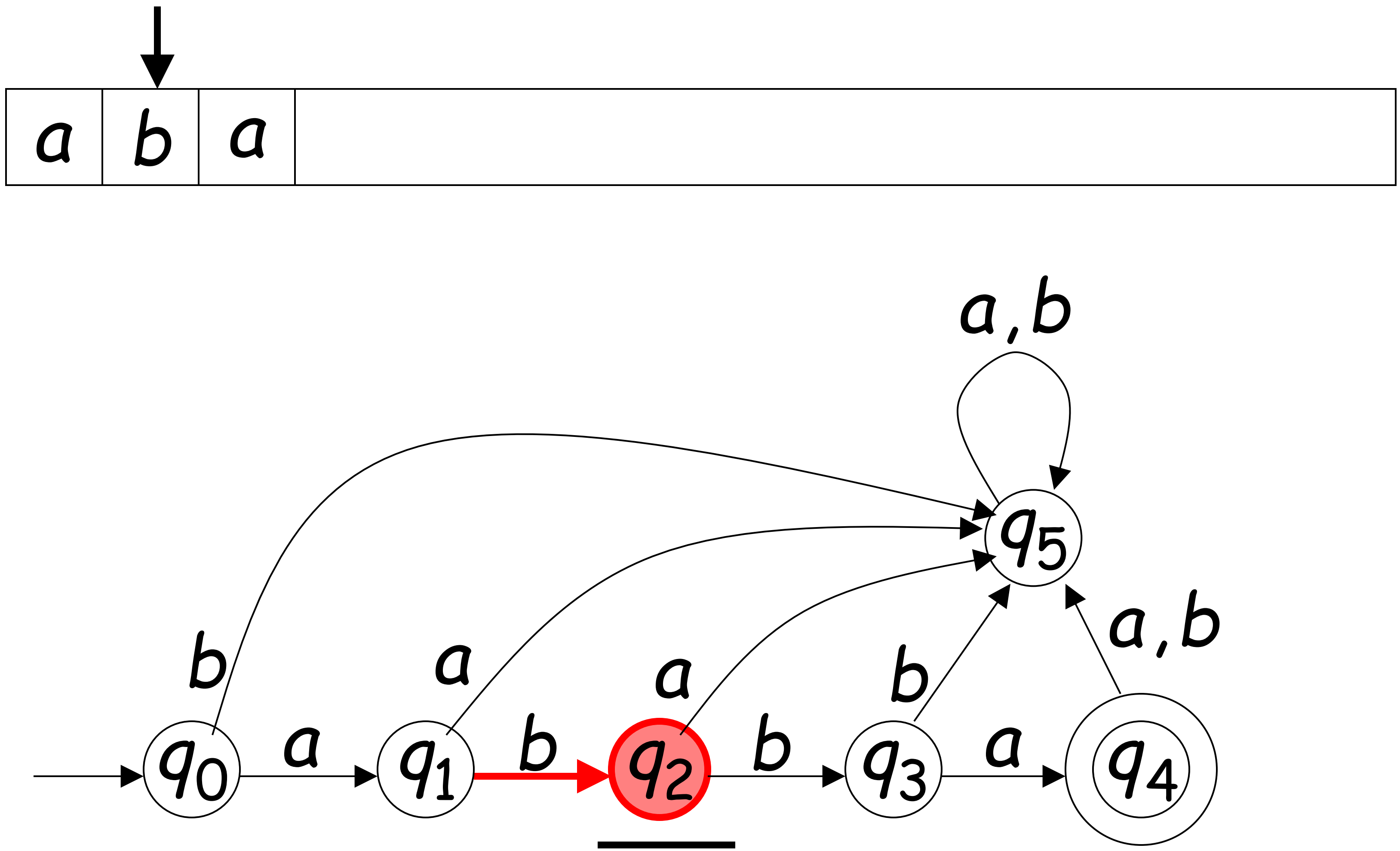
Output: “accept”

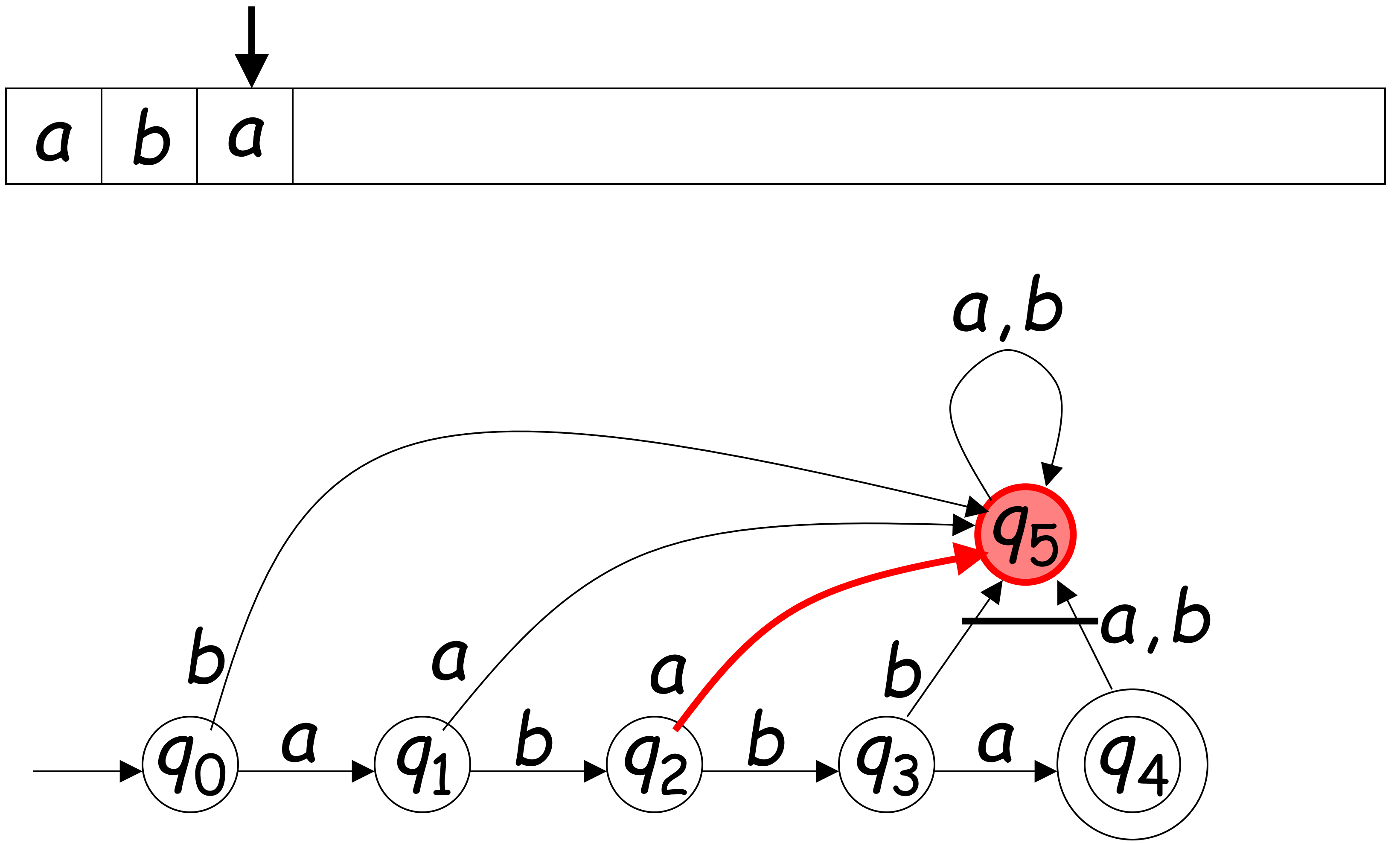
Rejection





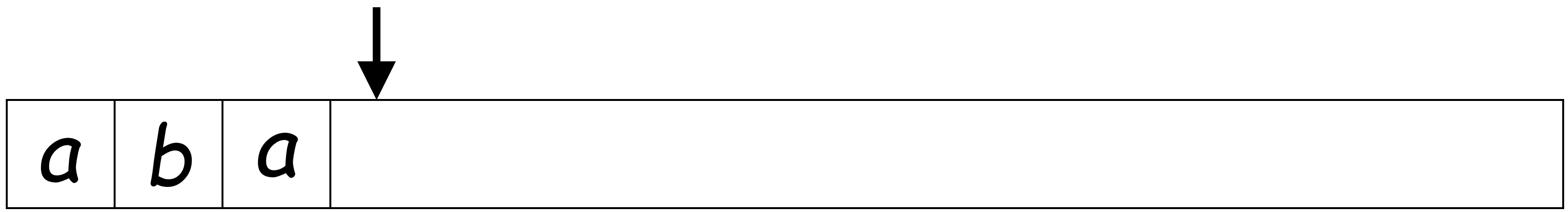
a
b
a
a,b
a,b
b
a
a
b
q_0
a
q_1
b
q_2
b
q_3
a
q_4
q_5

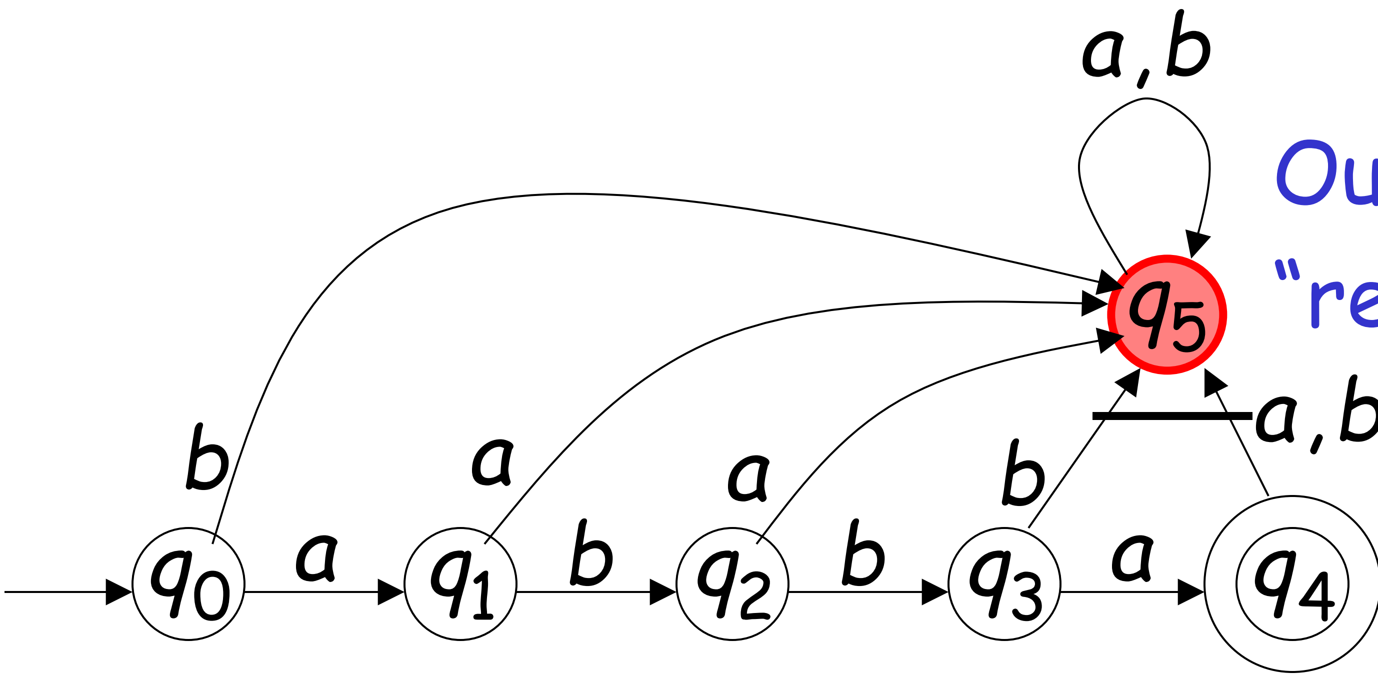

a
b
a
a,b
q5
b
a
a
b
a,b
q0
a
q1
b
q2
b
q3
a
q4



a
b
a
a,b
b
a
a
b
a,b
q0
a
q1
b
q2
b
q3
a
q4
q5

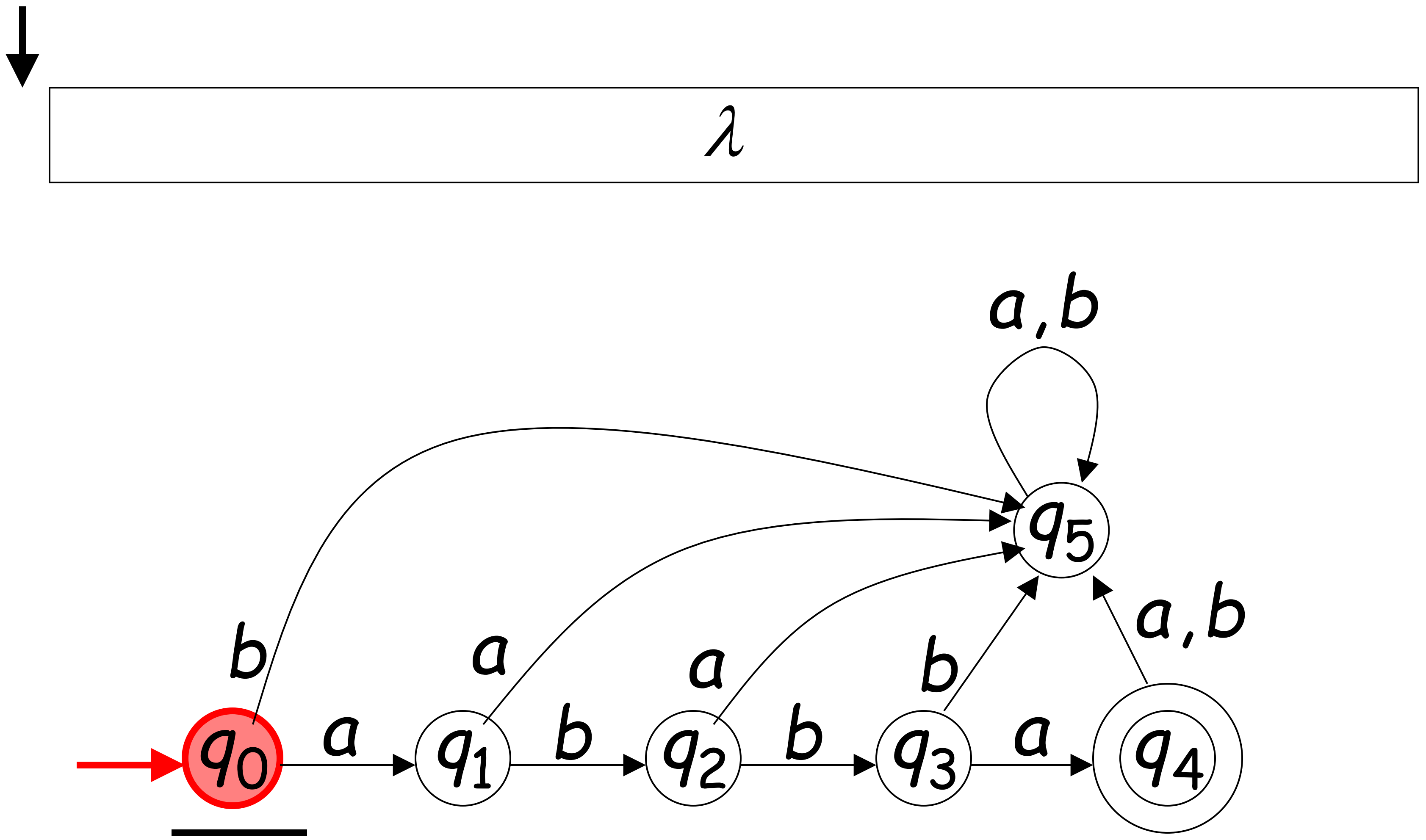
Input finished

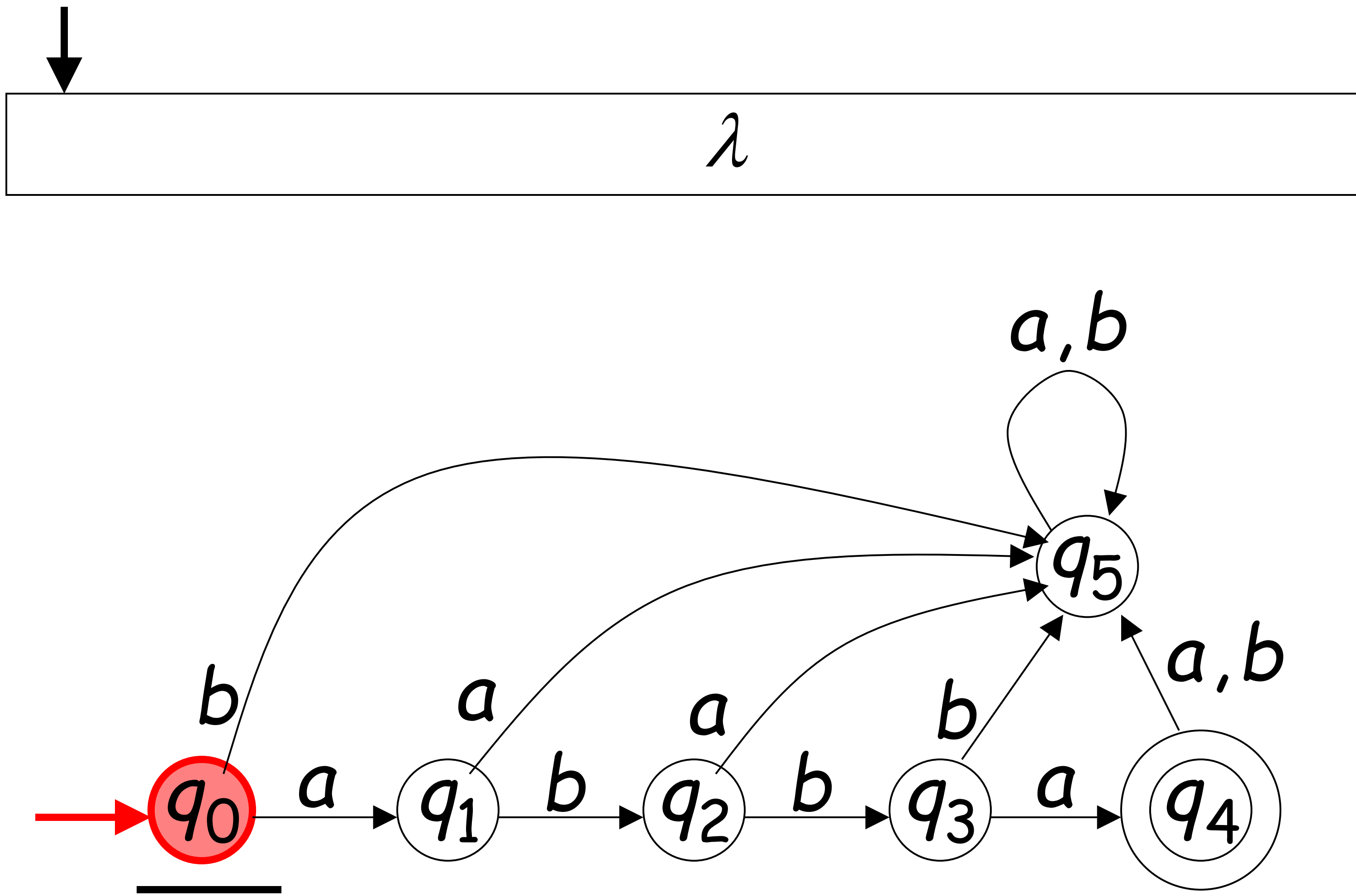




Output:
"reject"

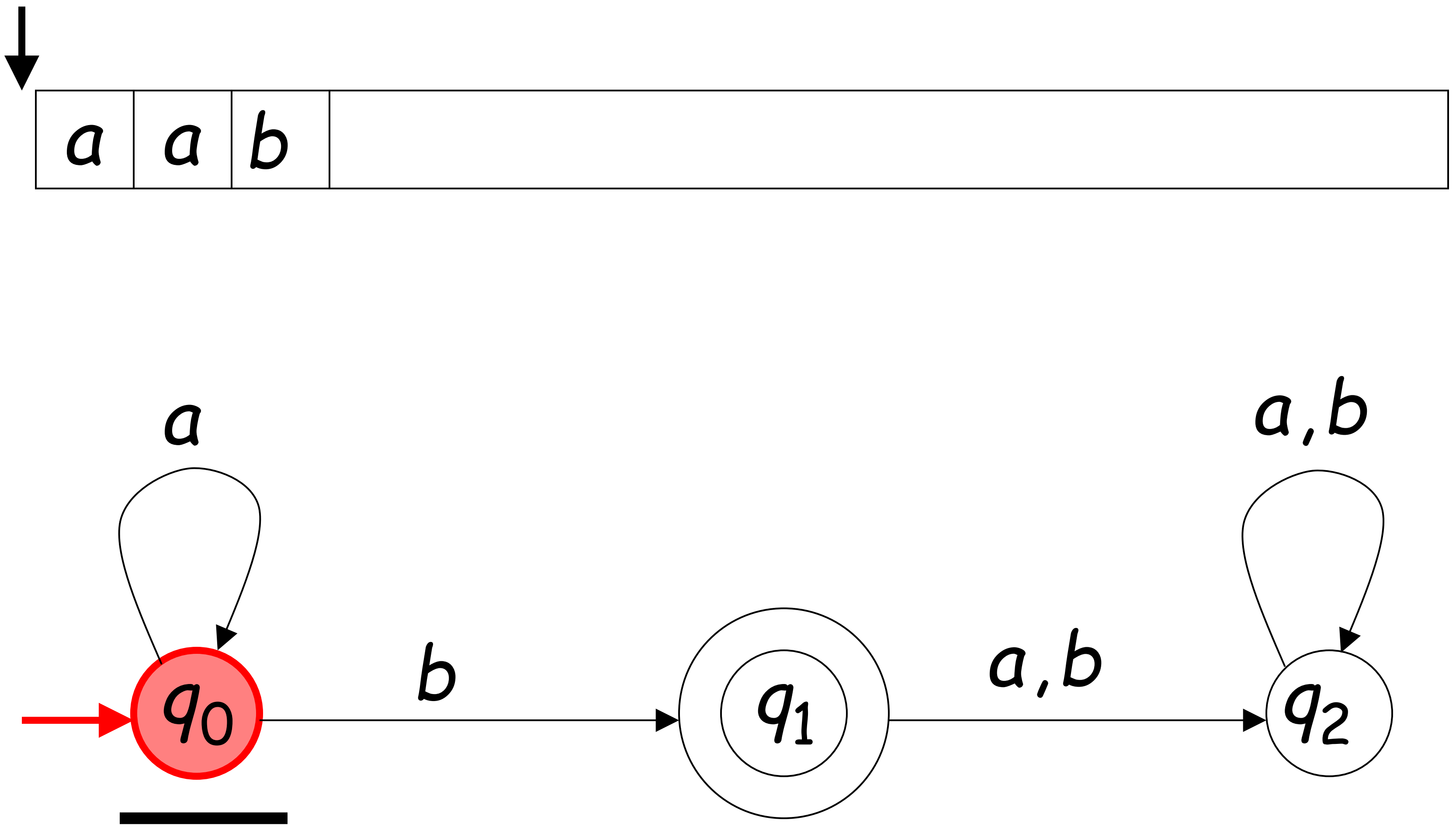
Another Rejection

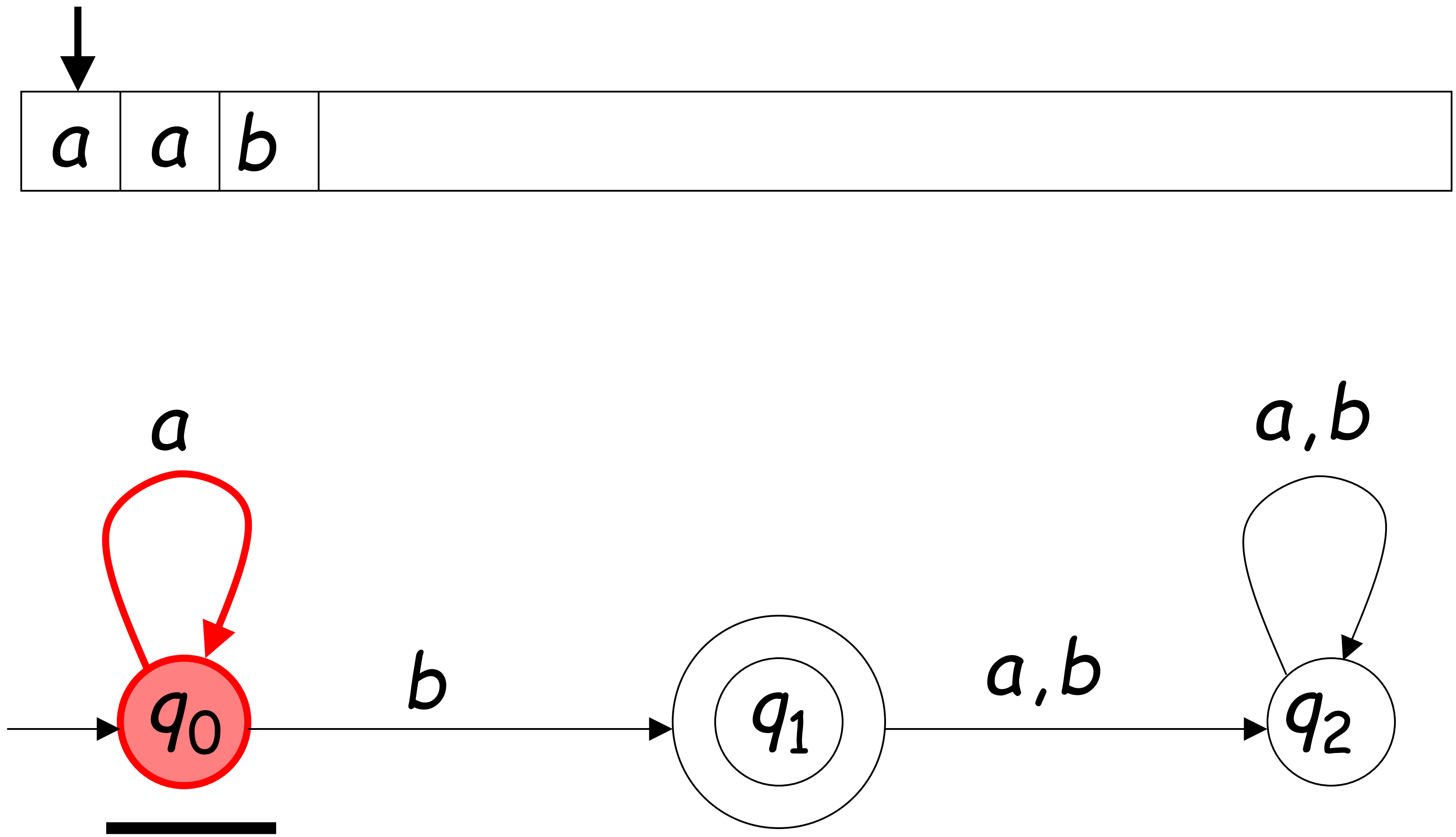




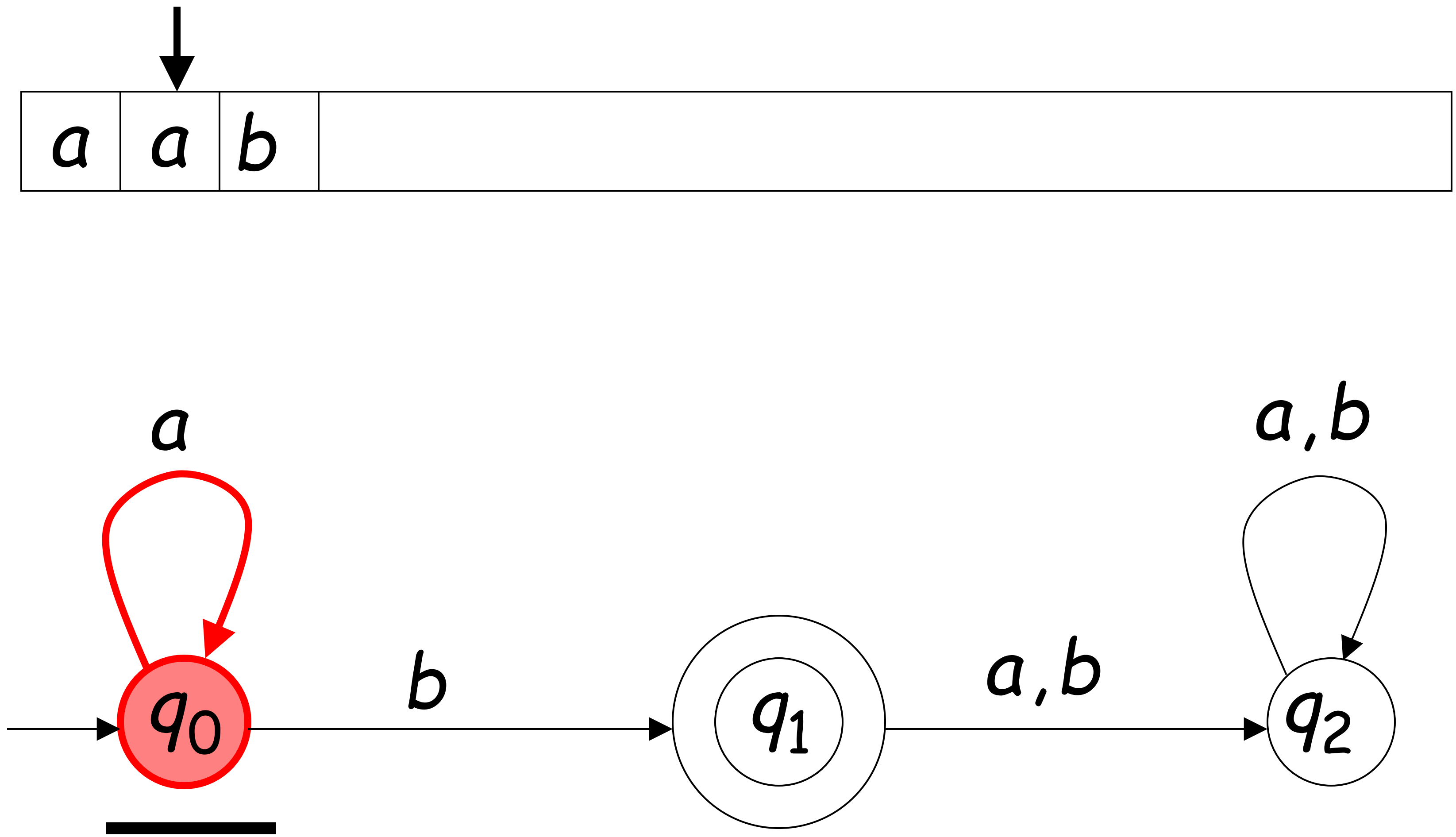
Output:
"reject"

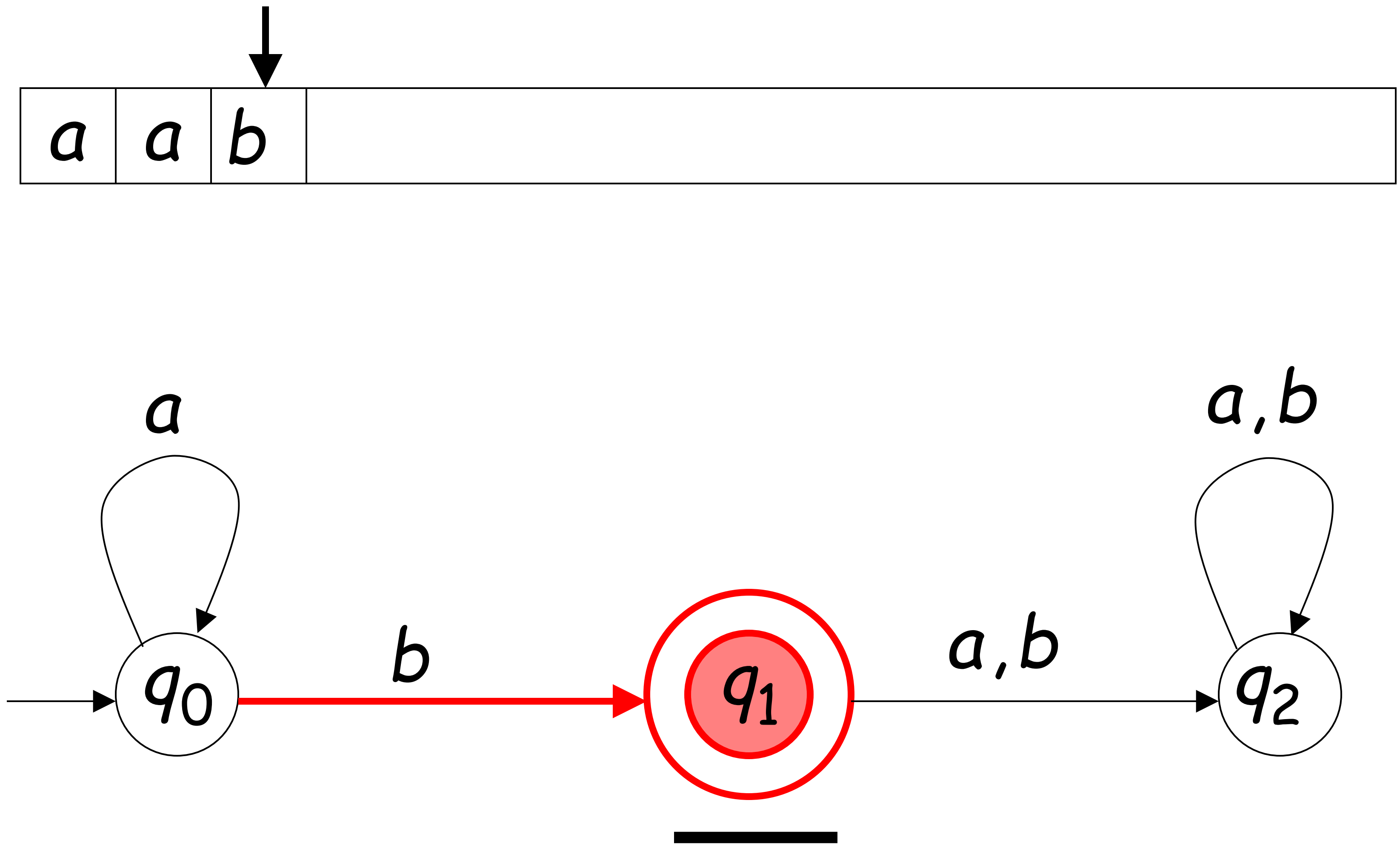
Another Example



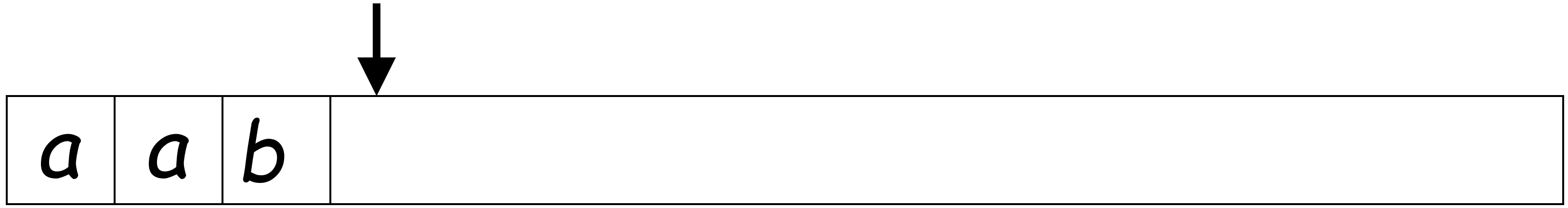


a
a
b
a
a,b
b
a,b
q0
q1
q2

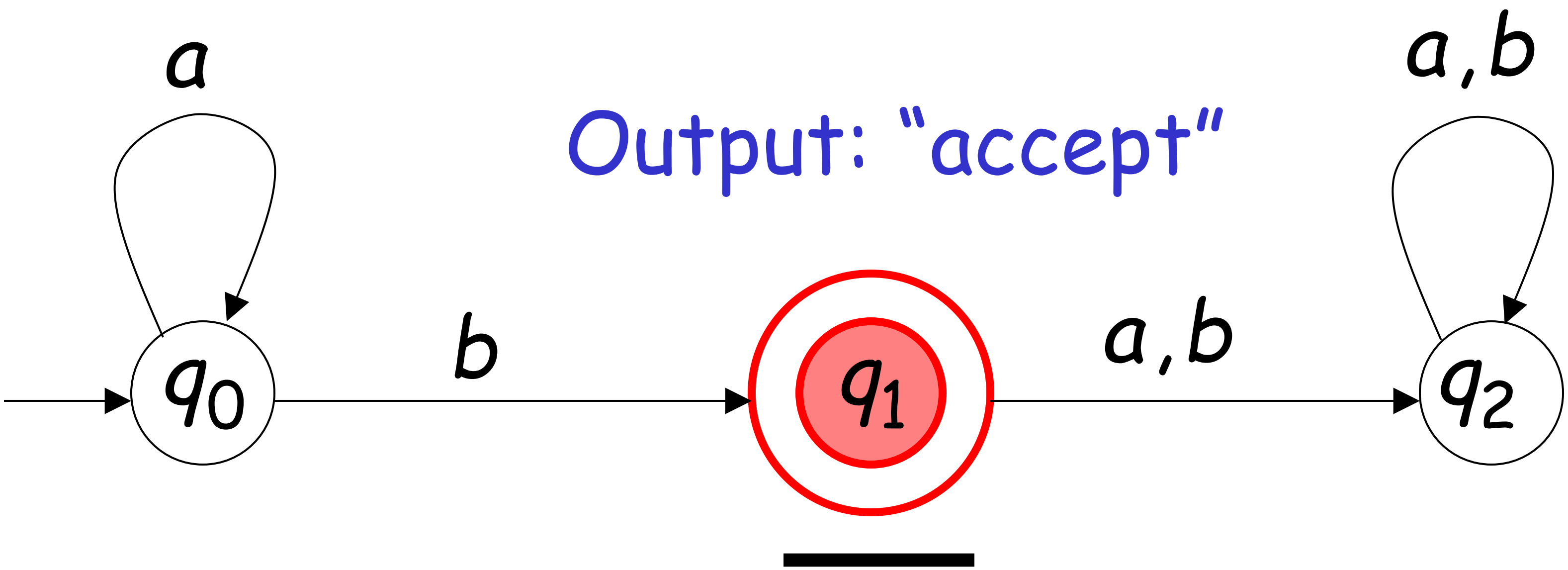

a
a
b
a
b
a,b
a,b
q_0
q_1
q_2


a
a
b
a
a,b
b
a,b
q_0
q_1
q_2

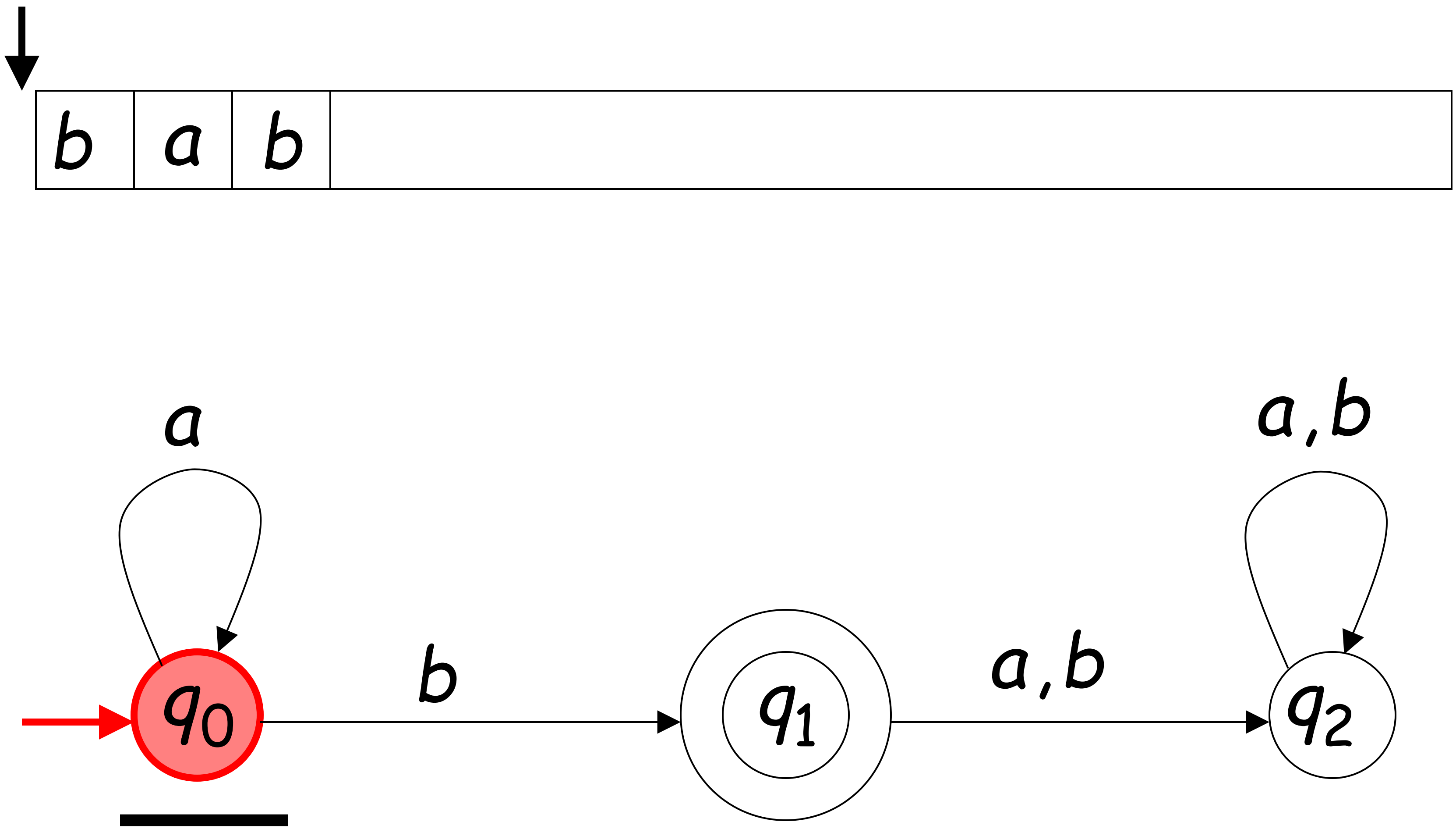
Input finished

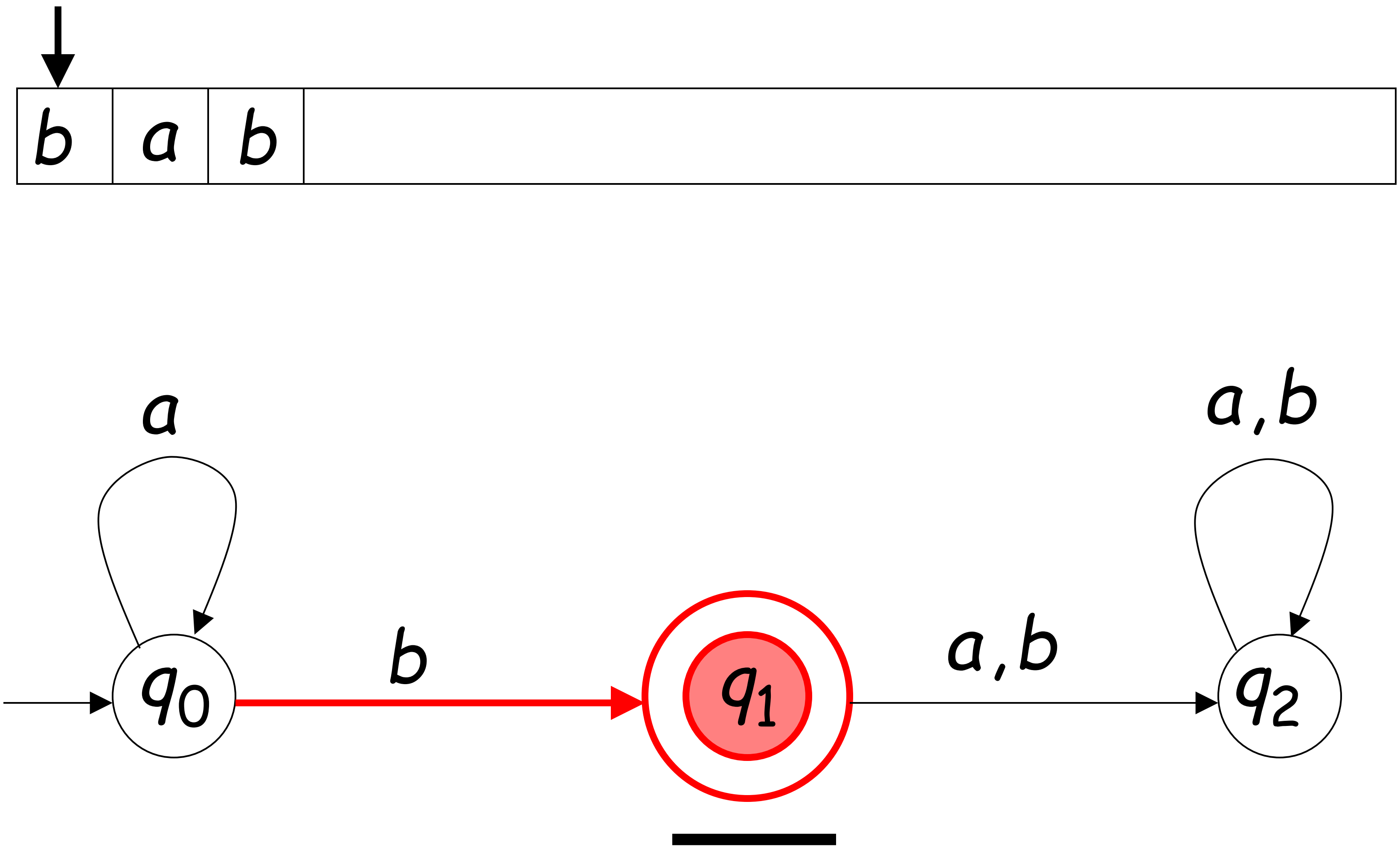


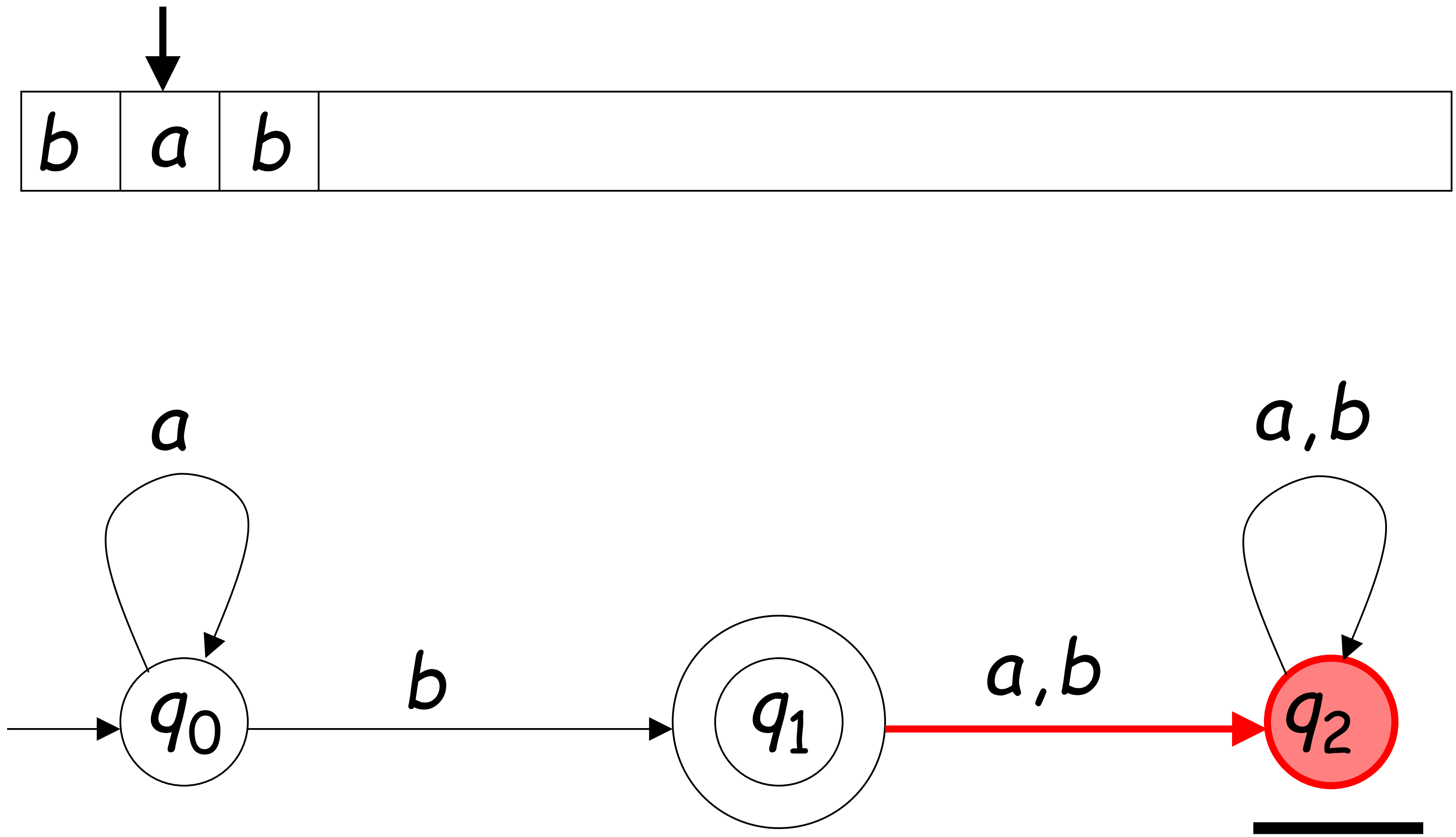
Output: "accept"

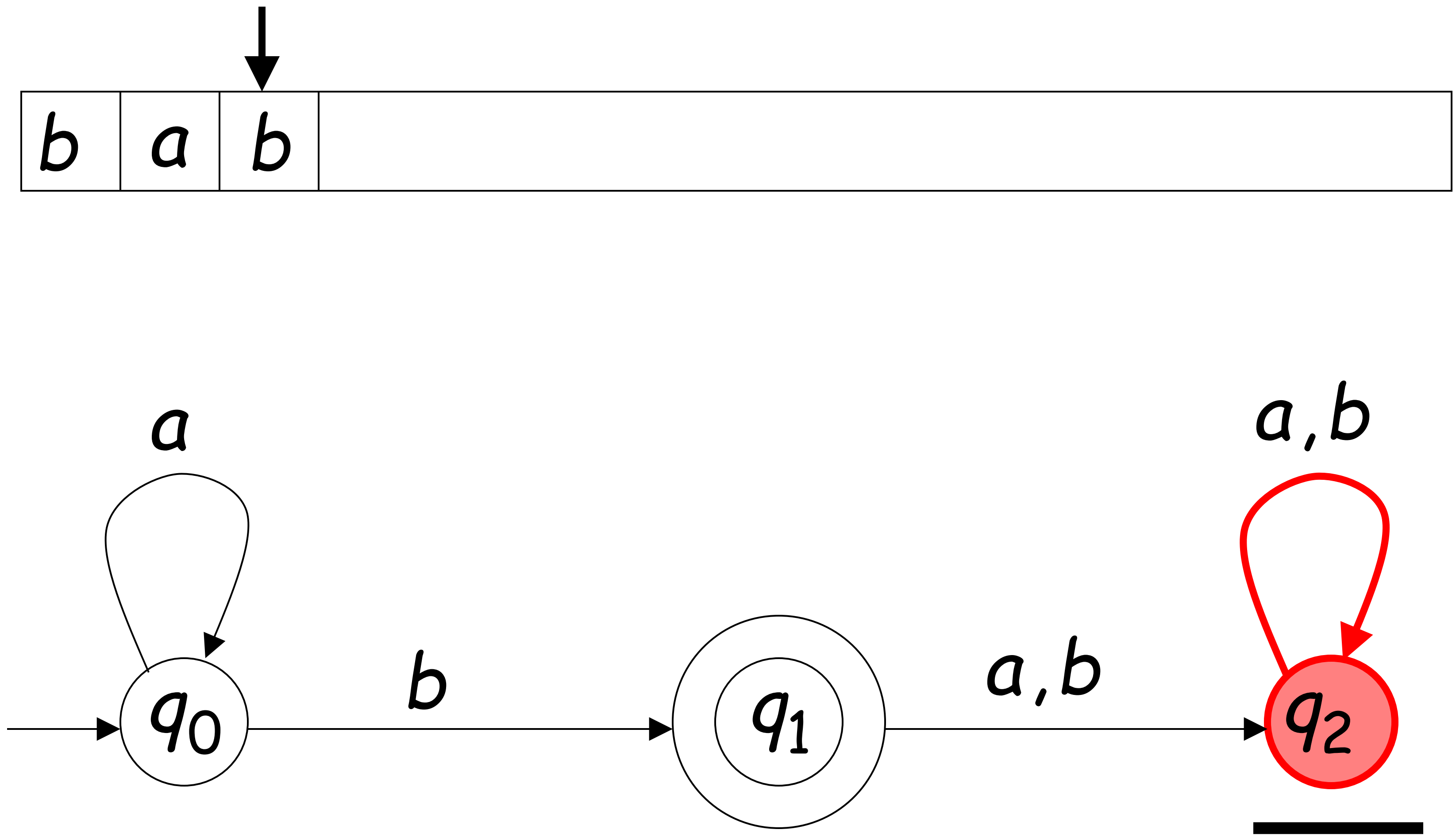


Rejection

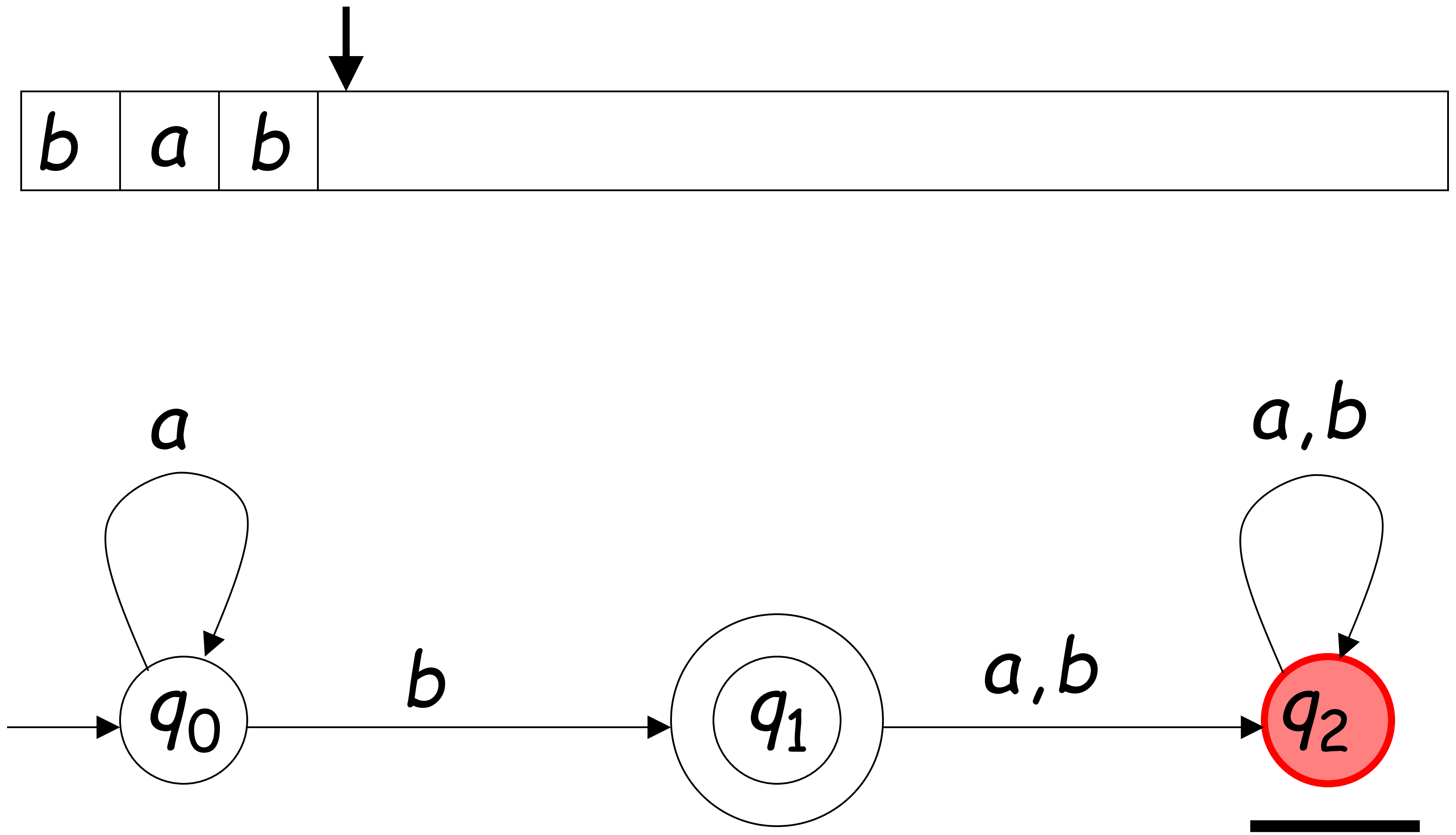



b
a
b
a
a,b
b
a,b
q0
q1
q2


b
a
b
a
b
q0
q1
a,b
a,b
q2


b
a
b
a
b
a,b
a,b
q_0
q_1
q_2

Input finished



Output: “reject”

Formalities

Deterministic Finite Accepter (DFA)

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

Q : set of states

Σ : input alphabet

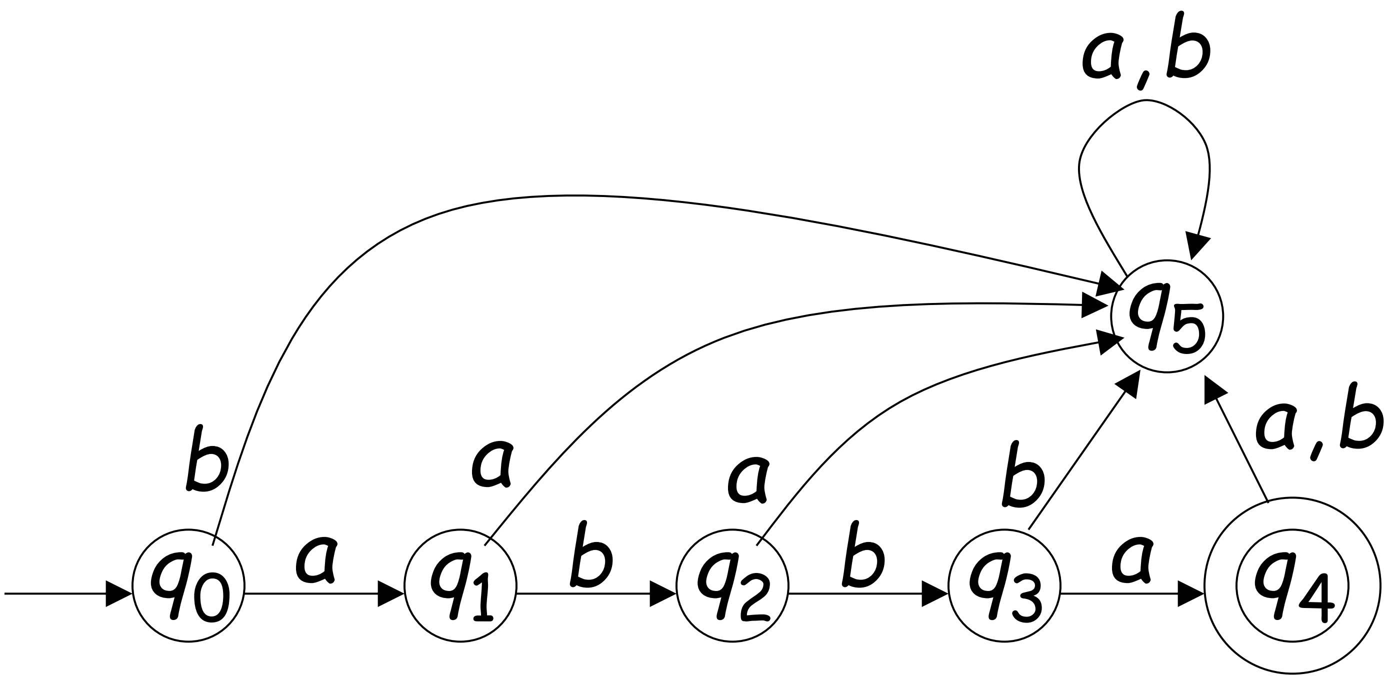
δ : transition function

q_0 : initial state

F : set of final states

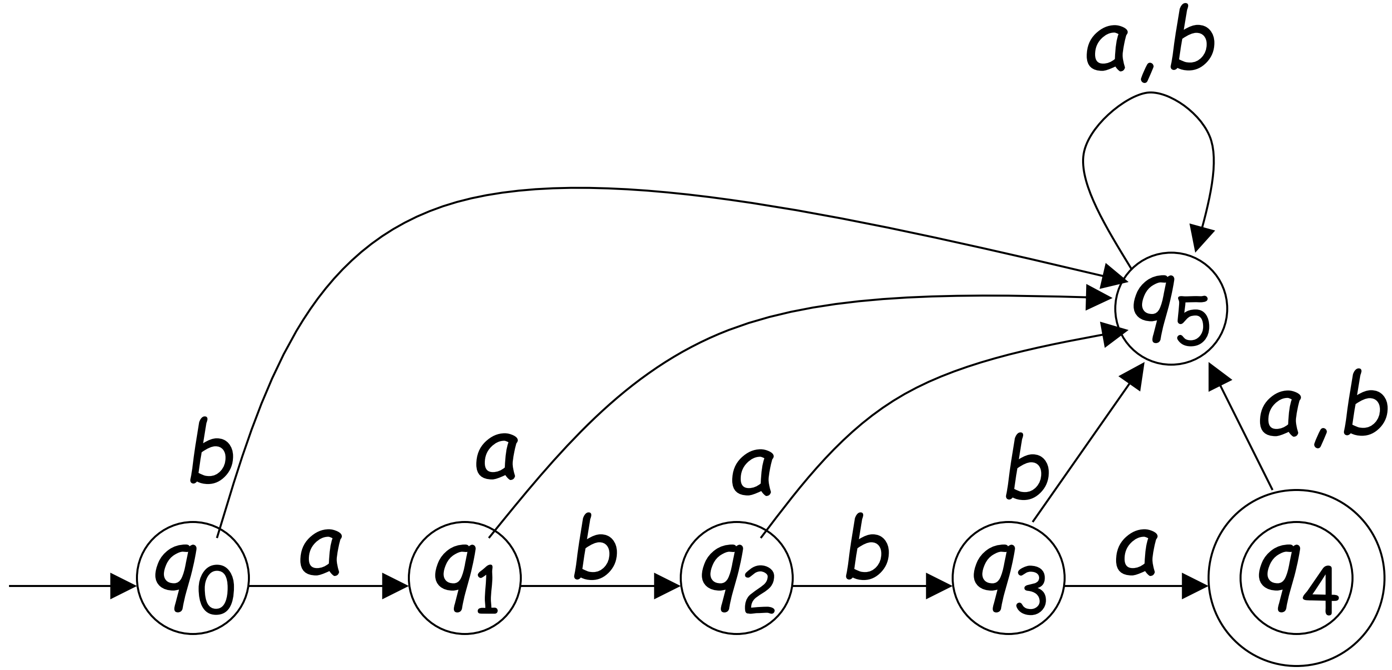
Input Alphabet Σ

$$\Sigma = \{a, b\}$$

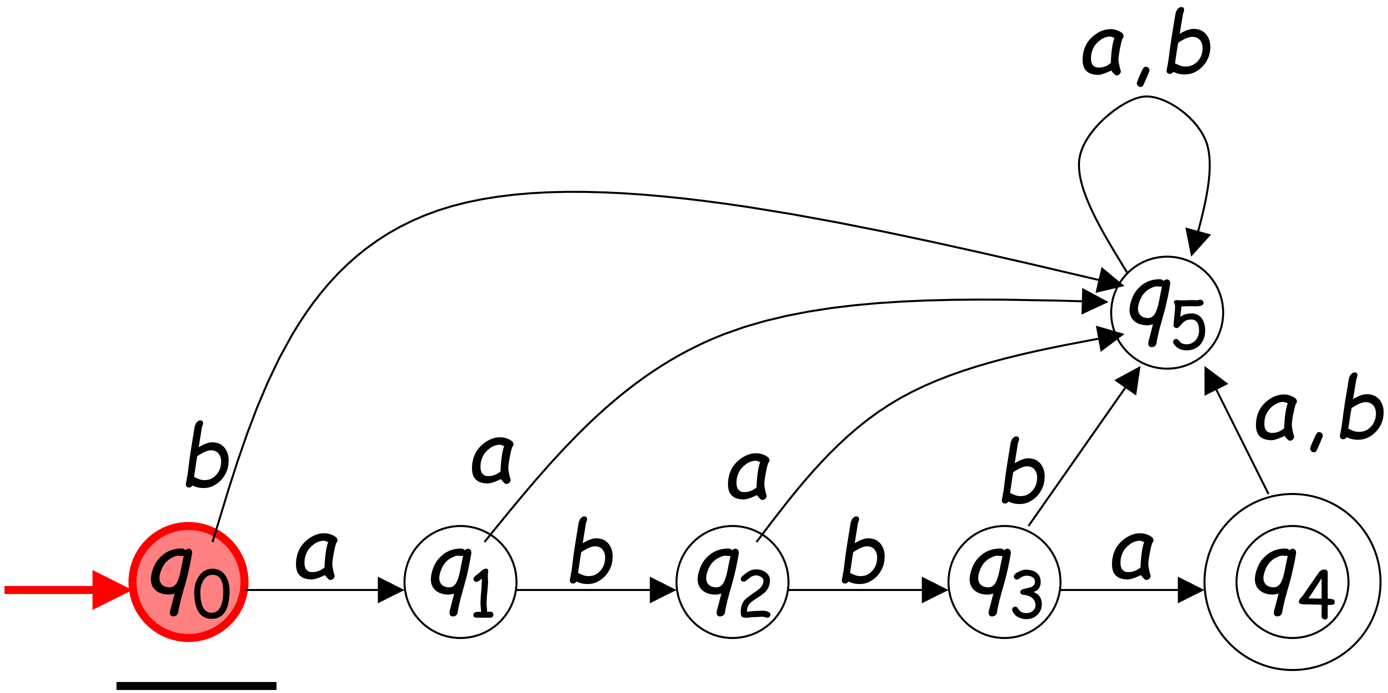


Set of States Q

$$Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$$

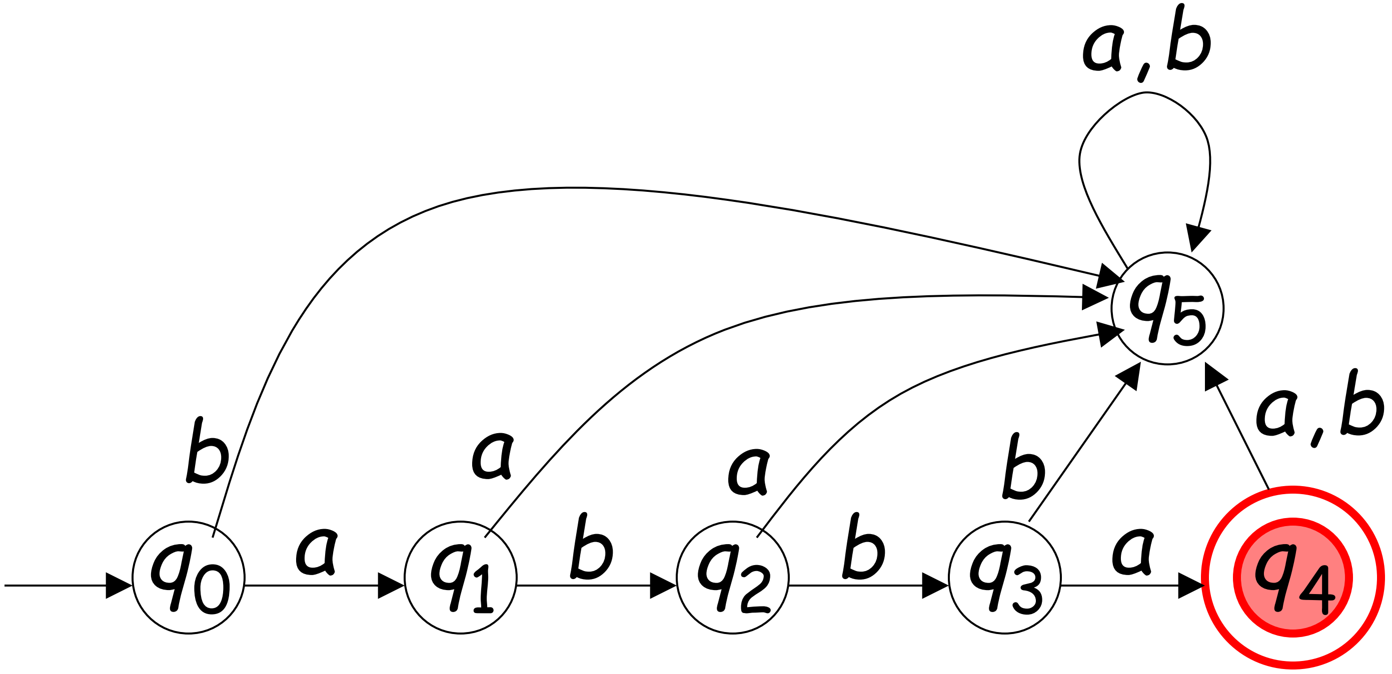


Initial State q_0



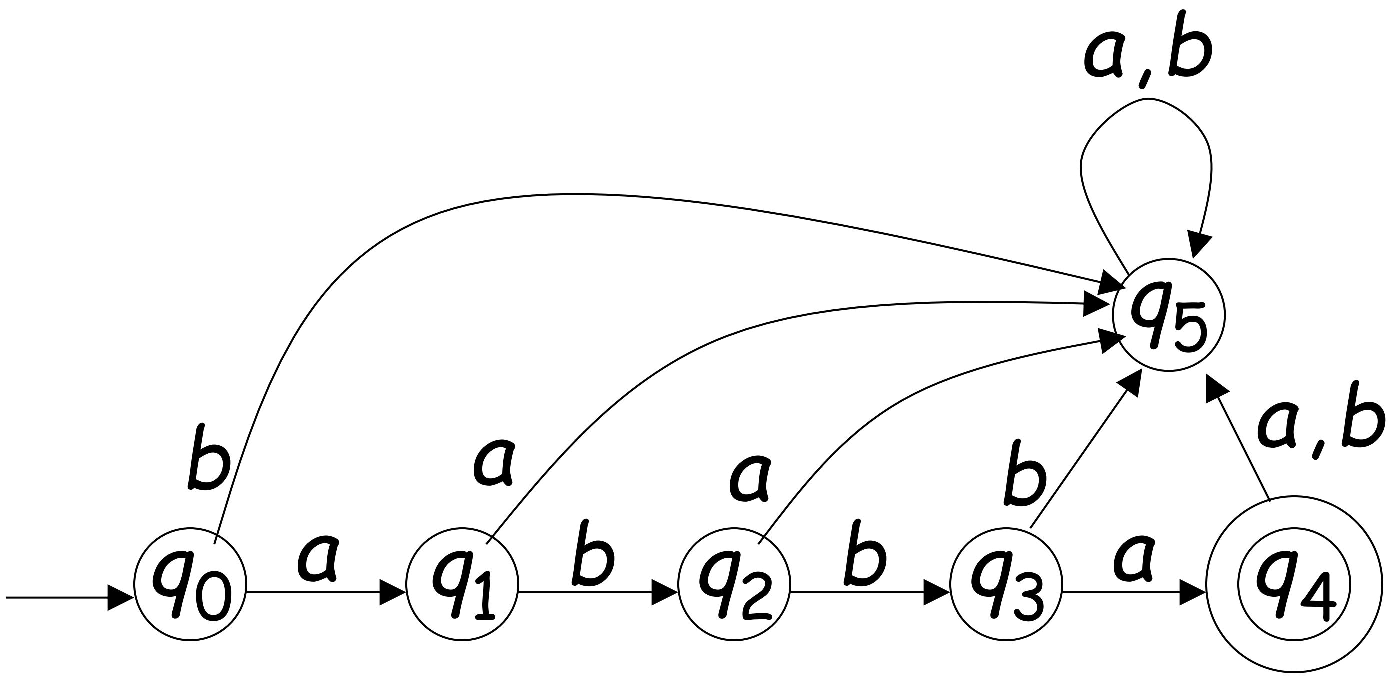
Set of Final States F

$$F = \{q_4\}$$

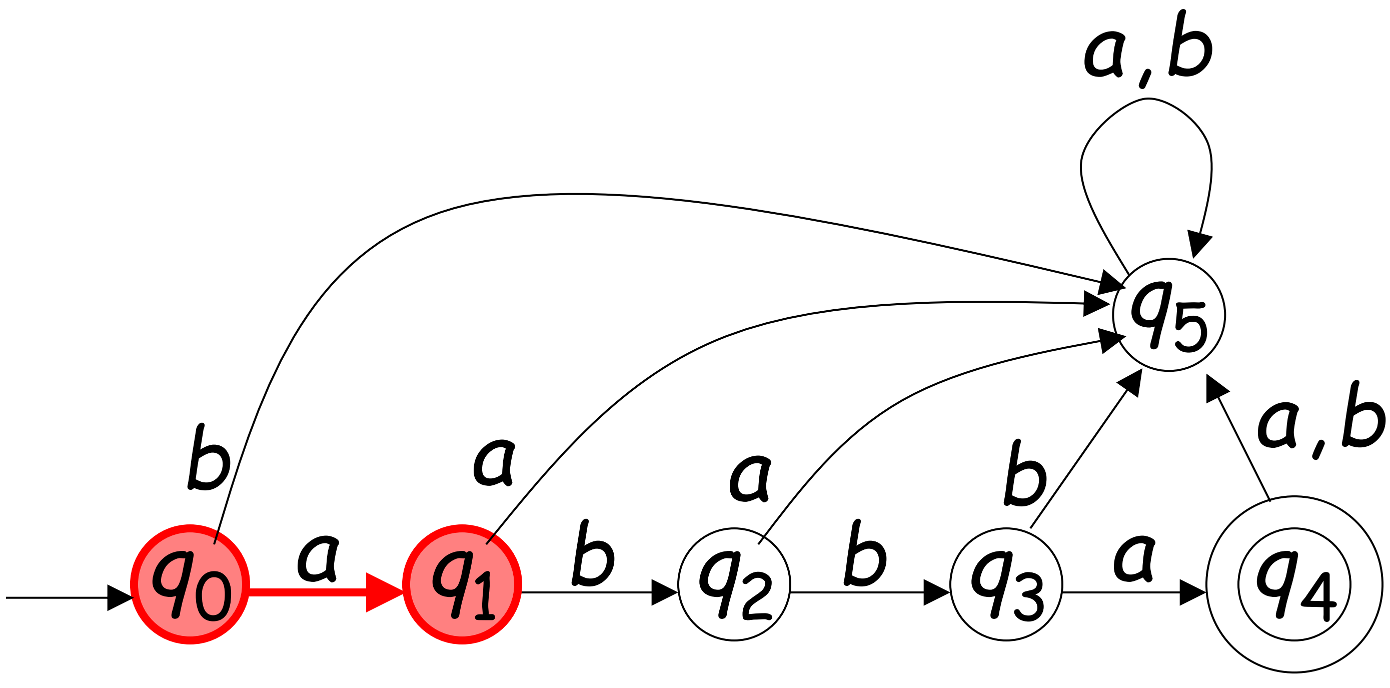


Transition Function δ

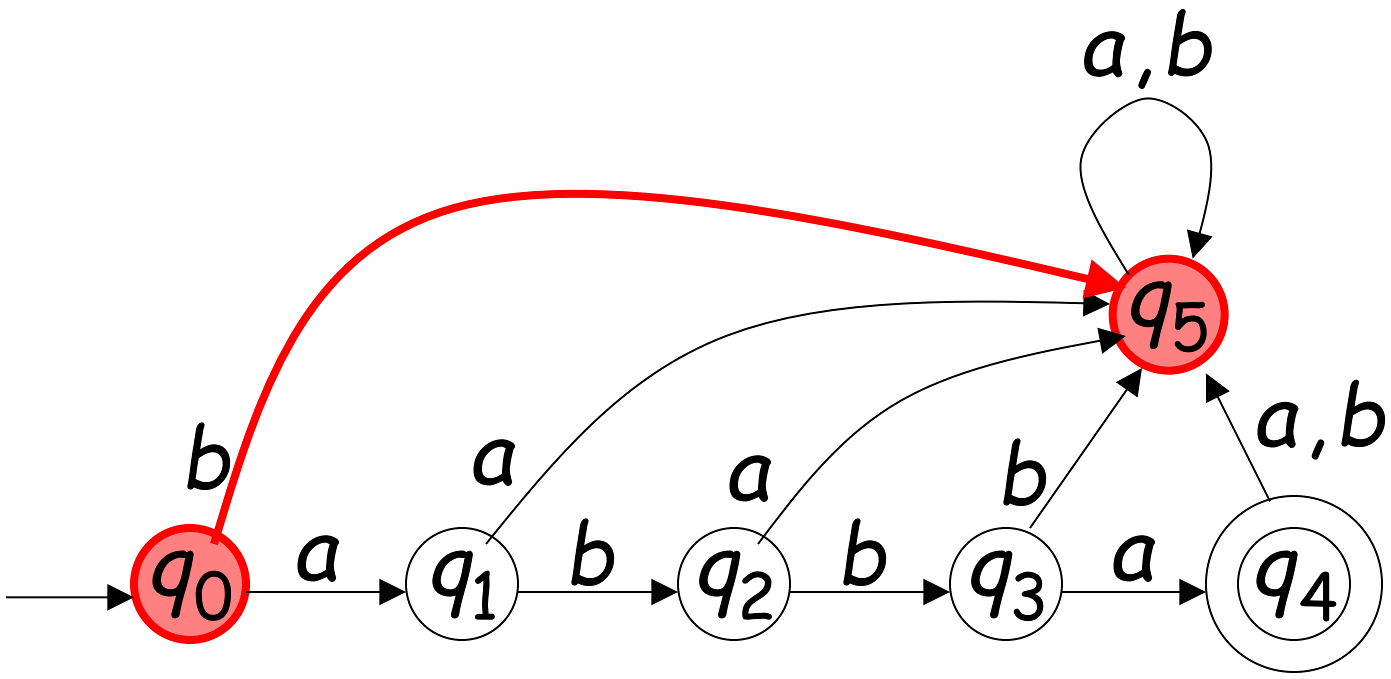
$$\delta : Q \times \Sigma \rightarrow Q$$



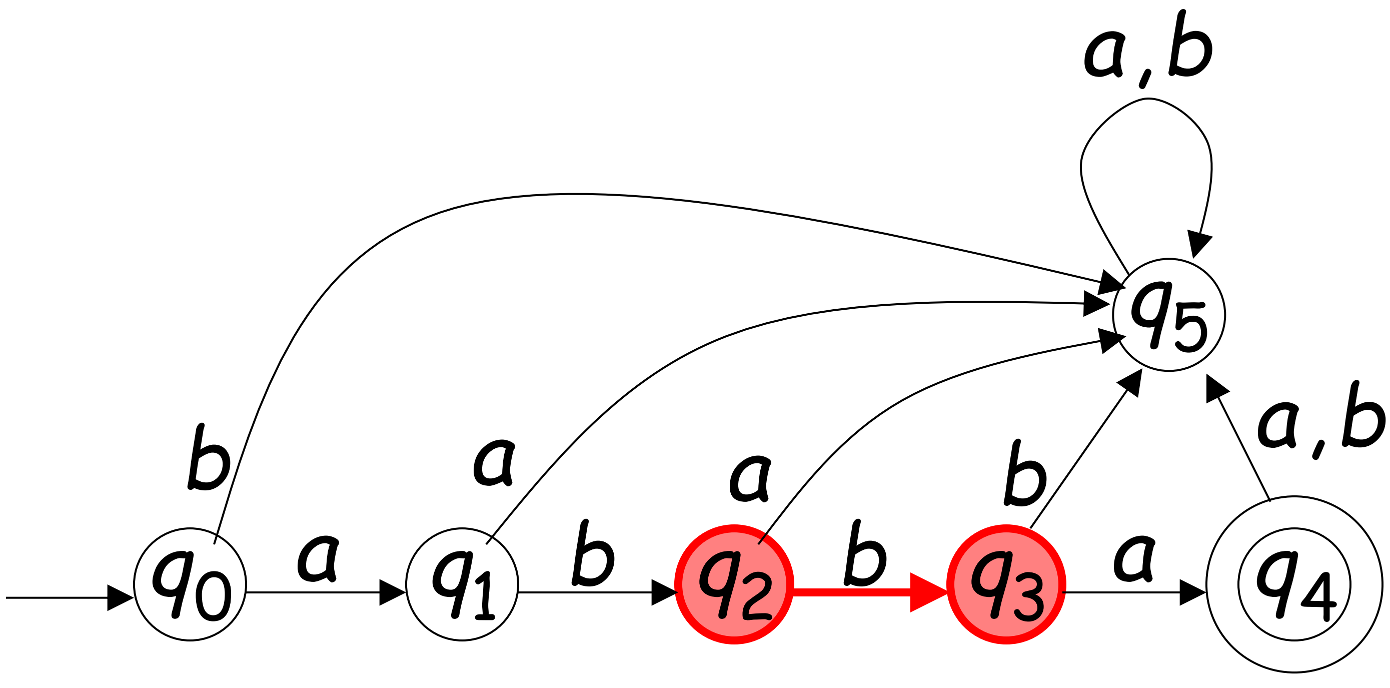
$$\delta(q_0, a) = q_1$$



$$\delta(q_0, b) = q_5$$

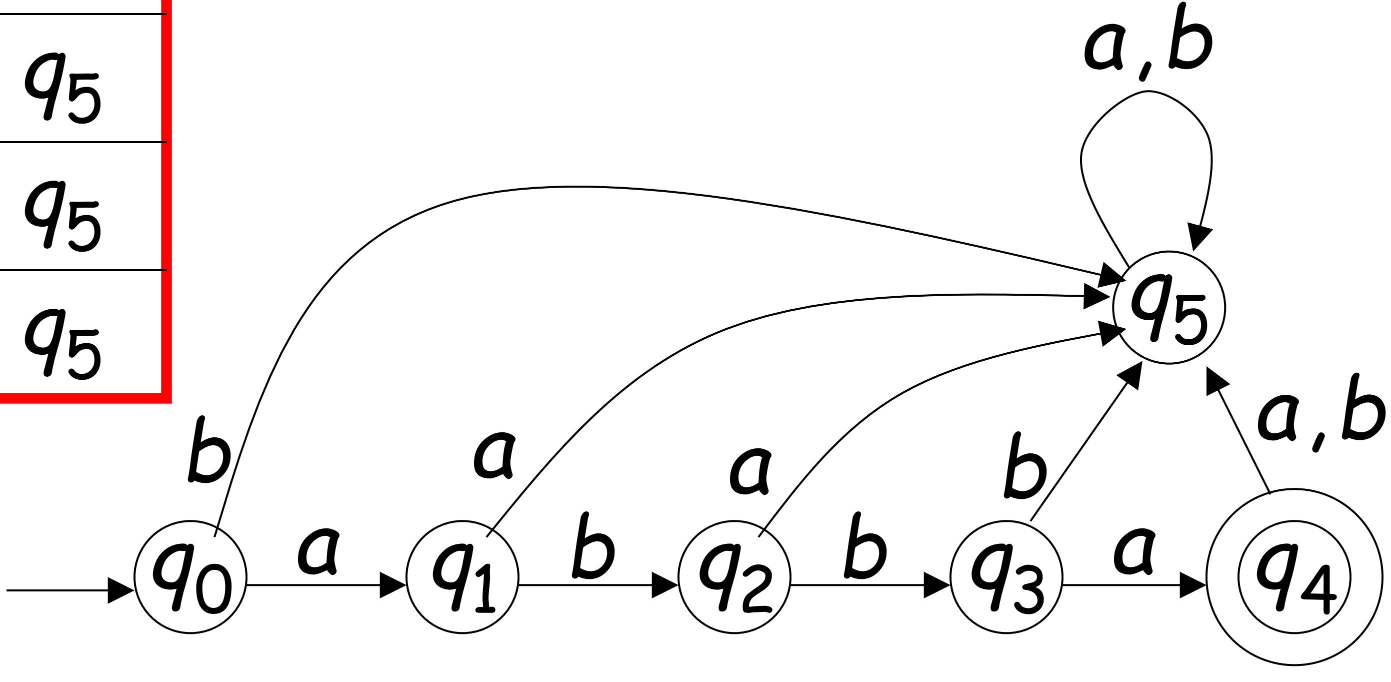


$$\delta(q_2, b) = q_3$$



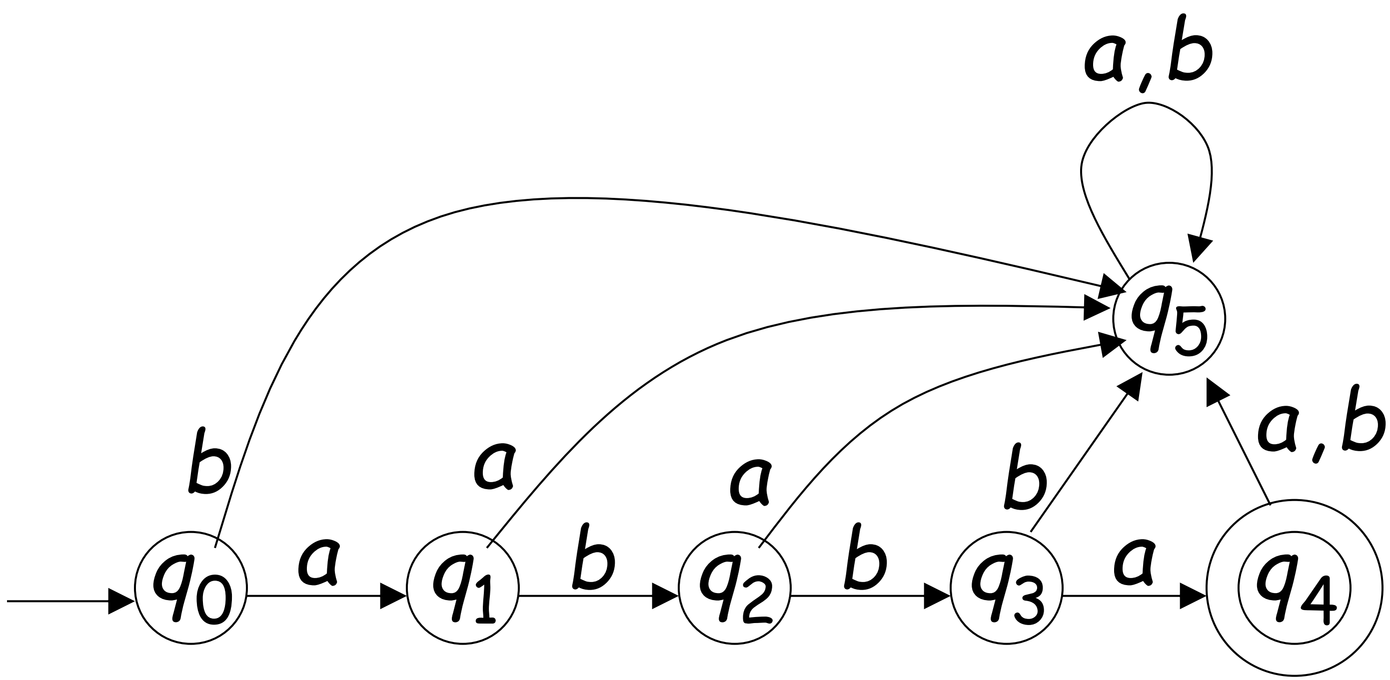
Transition Function δ

δ	a	b
q_0	q_1	q_5
q_1	q_5	q_2
q_2	q_5	q_3
q_3	q_4	q_5
q_4	q_5	q_5
q_5	q_5	q_5

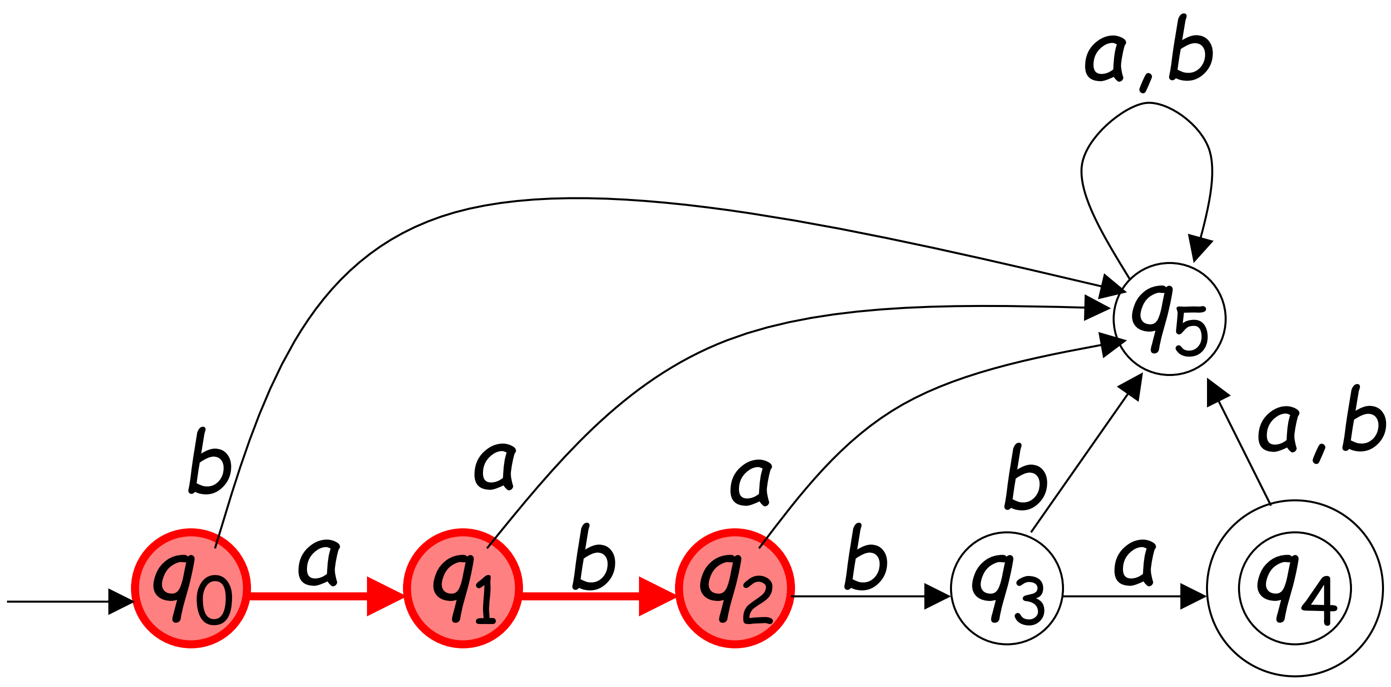


Extended Transition Function δ^*

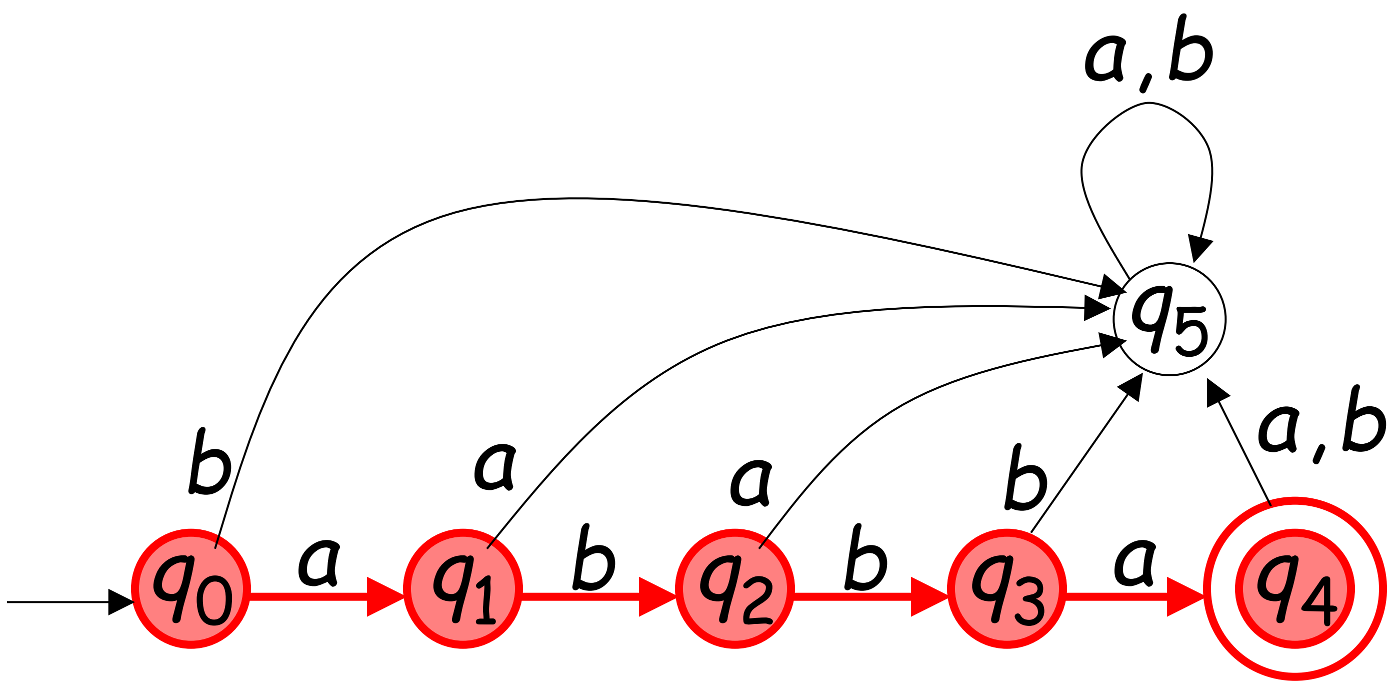
$$\delta^* : Q \times \Sigma^* \to Q$$



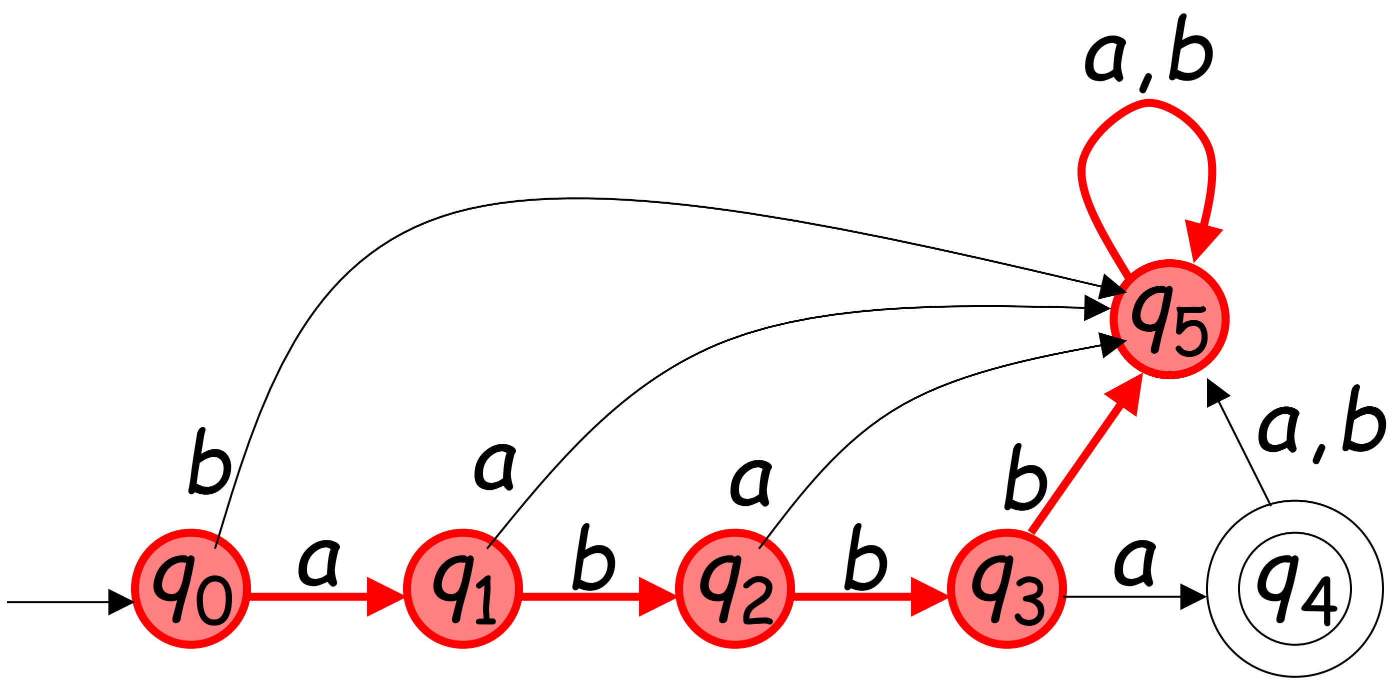
$$\delta^*(q_0, ab) = q_2$$



$$\delta^*(q_0, abba) = q_4$$

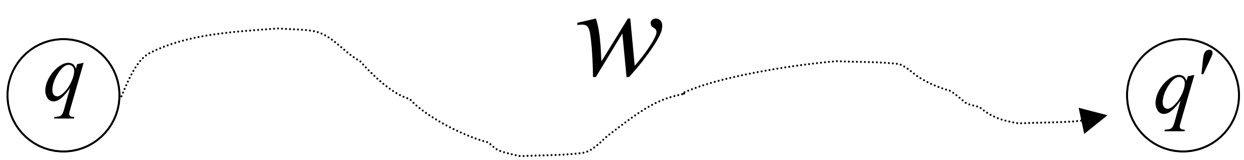


$$\delta^*(q_0, abbbaa) = q_5$$

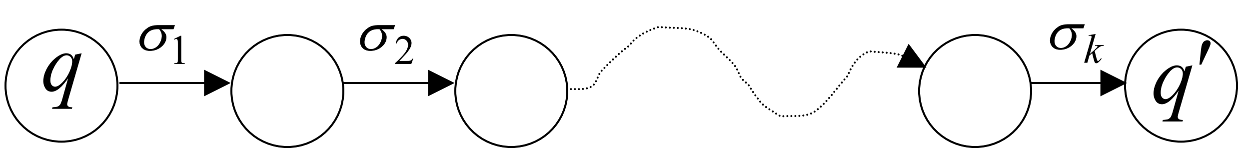


Observation: There is a walk from q to q' with label w

$$\delta^*(q, w) = q'$$

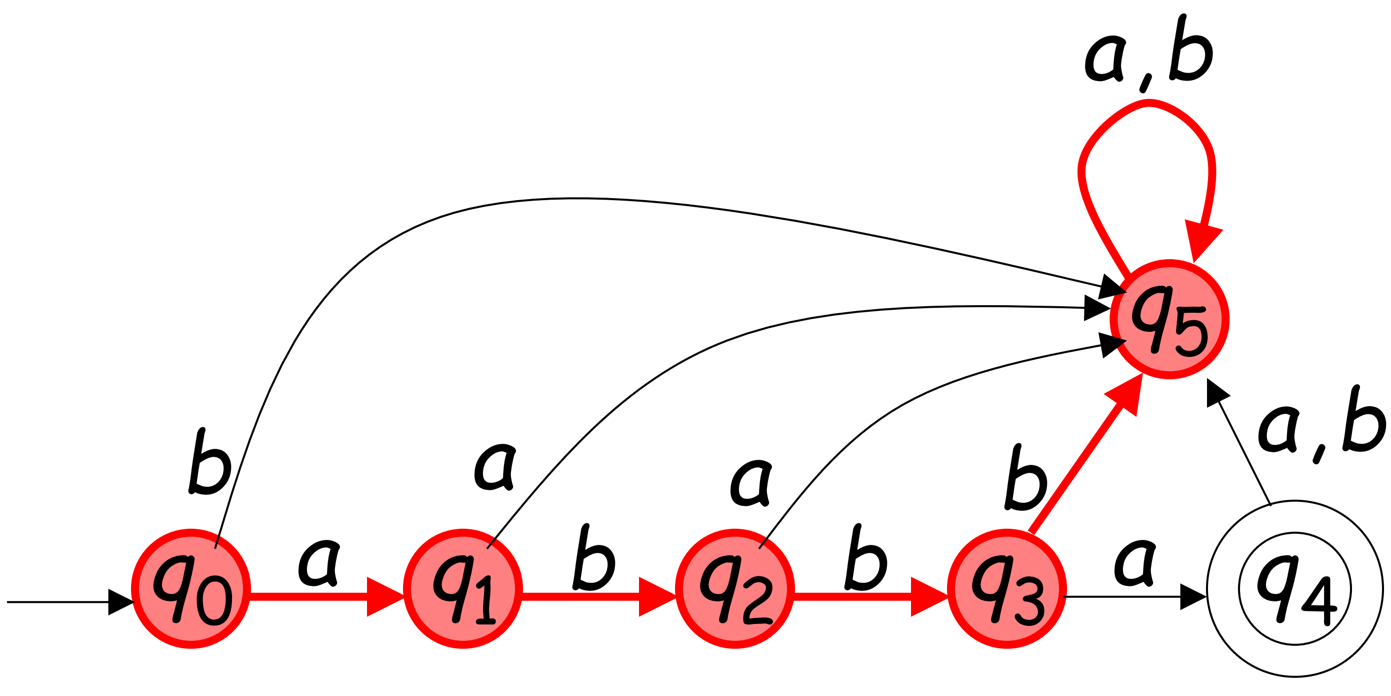


$$w = \sigma_1 \sigma_2 \cdots \sigma_k$$



Example: There is a walk from q_0 to q_5 with label $abbbaa$

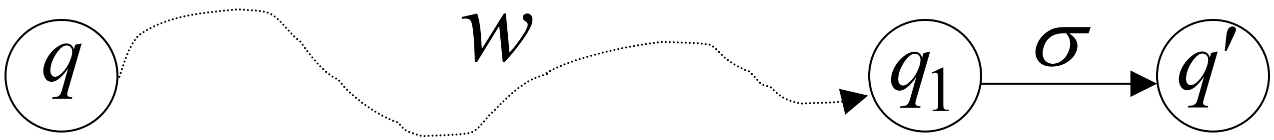
$$\delta^*(q_0, abbbaa) = q_5$$



Recursive Definition

$$\delta^*(q,\lambda)=q$$

$$\delta^*(q,w\sigma)=\delta(\delta^*(q,w),\sigma)$$



$$\left.\begin{array}{l}\delta^*(q,w\sigma)=q'\\[2em] \delta(q_1,\sigma)=q'\end{array}\right\} \Rightarrow \delta^*(q,w\sigma)=\delta(q_1,\sigma)$$

$$\left.\begin{array}{l}\delta^*(q,w\sigma)=\delta(q_1,\sigma)\\[2em] \delta^*(q,w)=q_1\end{array}\right\} \Rightarrow \delta^*(q,w\sigma)=\delta(\delta^*(q,w),\sigma)$$

$$\delta^*(q_0, ab) =$$

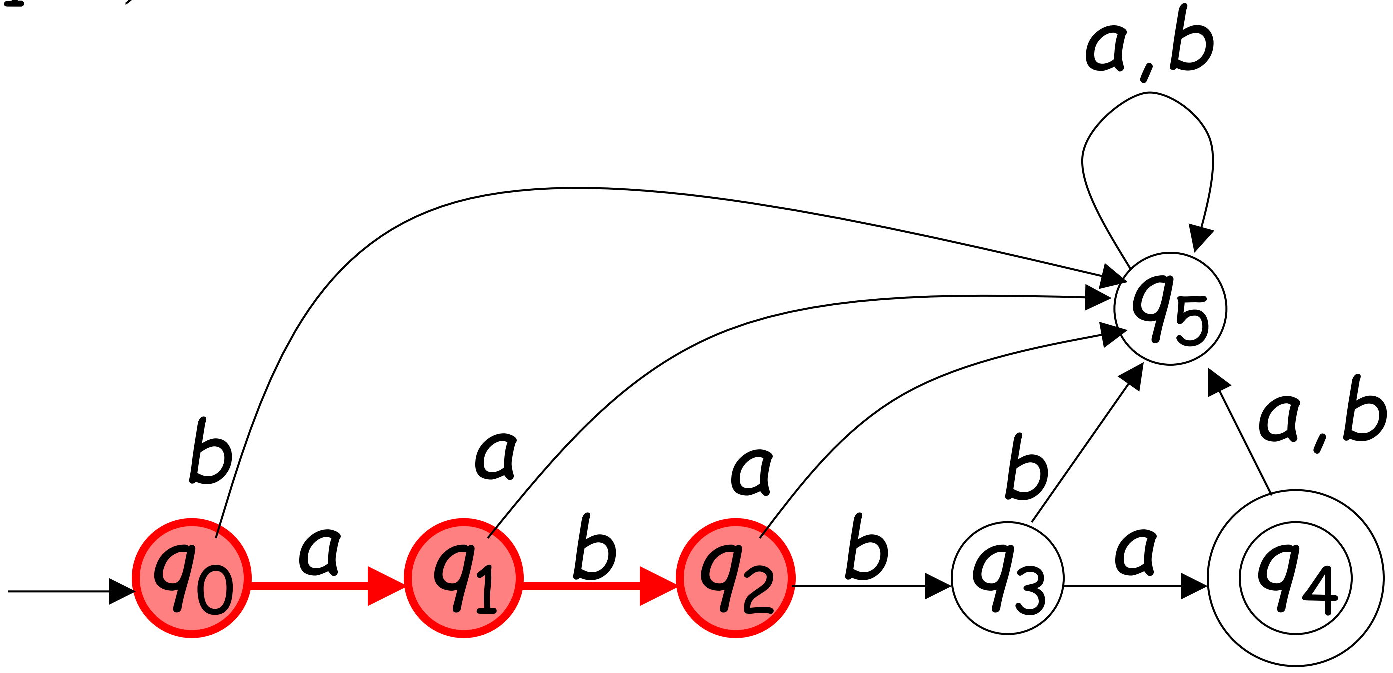
$$\delta(\delta^*(q_0, a), b) =$$

$$\delta(\delta(\delta^*(q_0, \lambda), a), b) =$$

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

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

$$q_2$$



Languages Accepted by DFAs

Take DFA M

Definition:

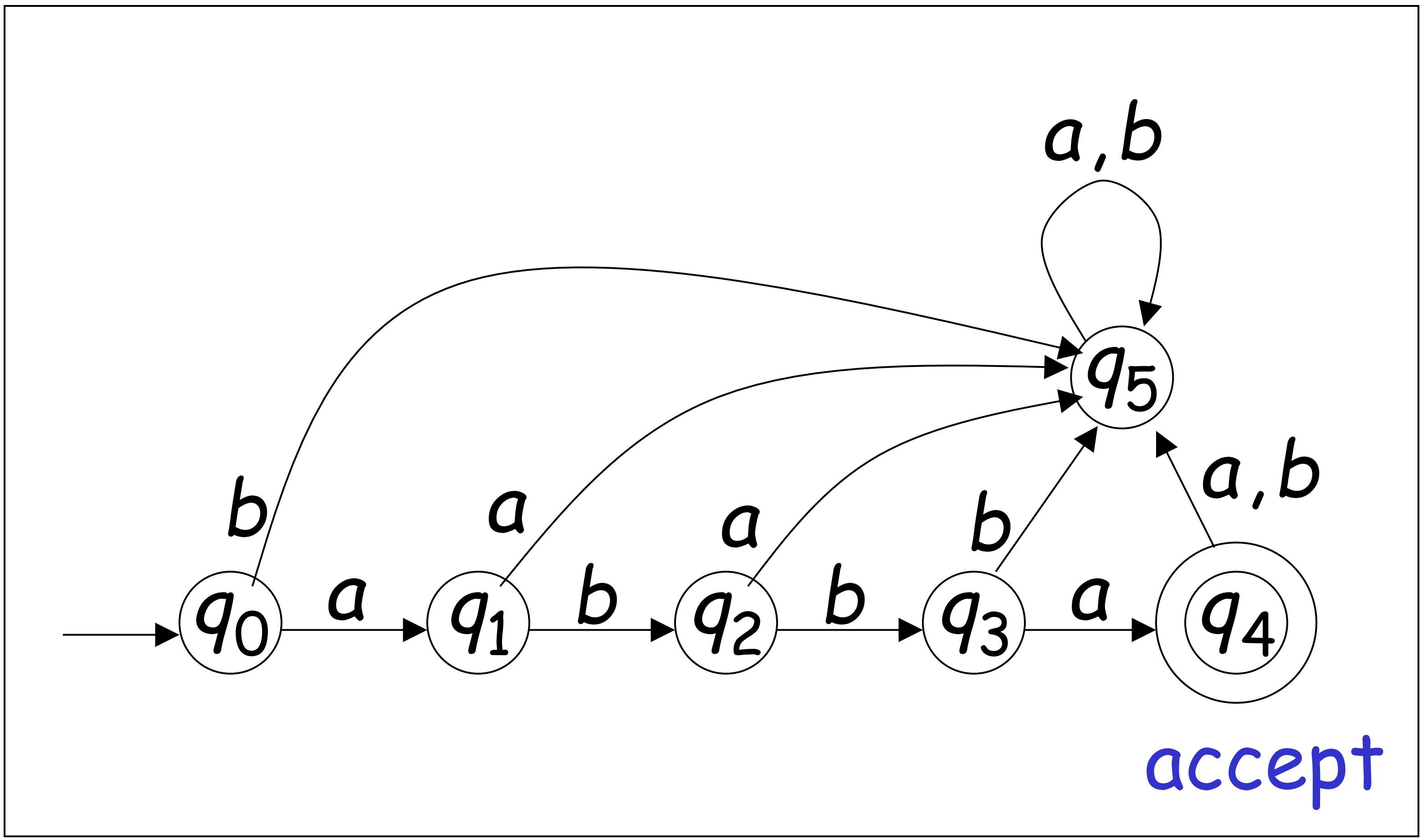
The language $L(M)$ contains all input strings accepted by M

$L(M)$ = { strings that drive M to a final state}

Example

$$L(M) = \{abba\}$$

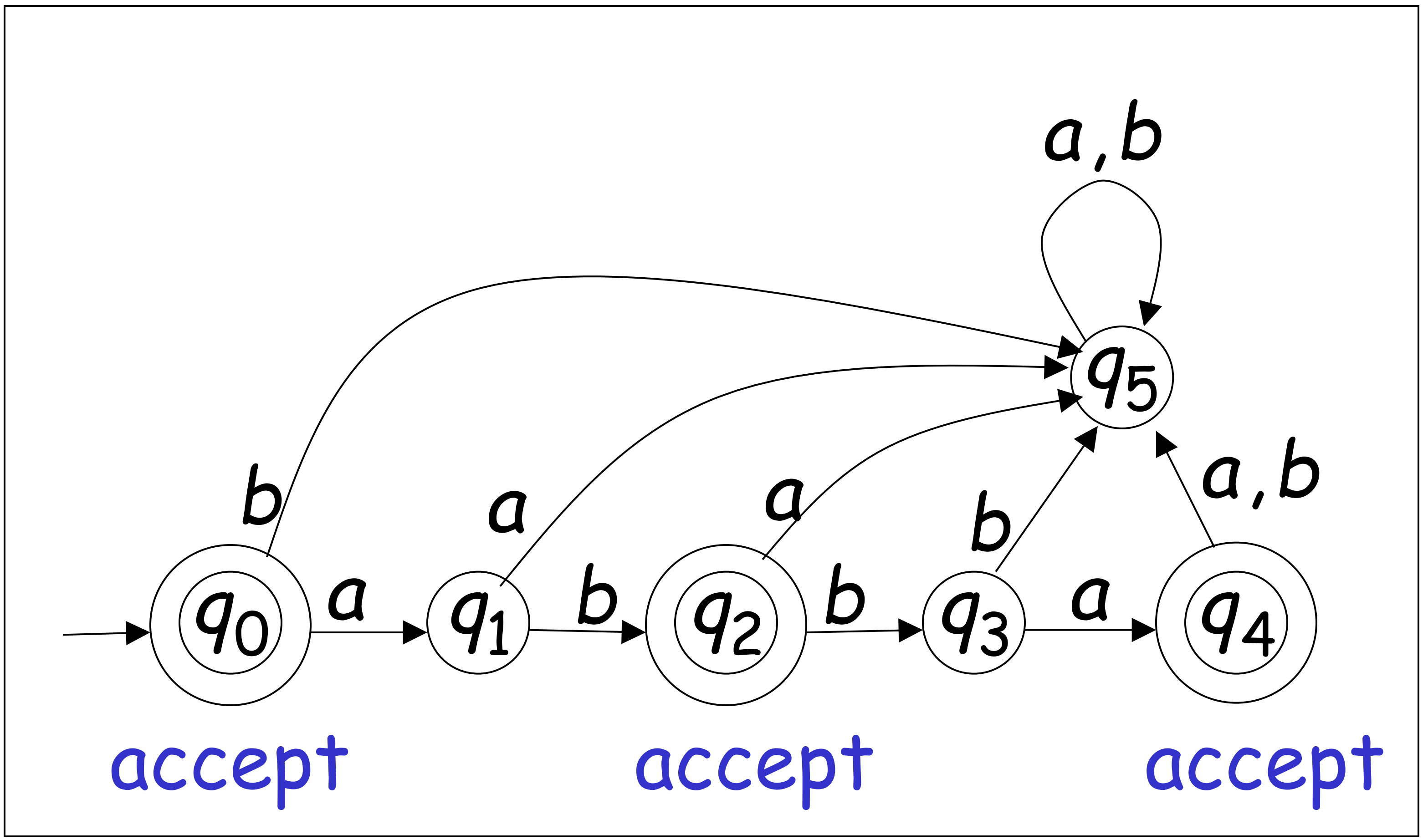
M



Another Example

$$L(M) = \{\lambda, ab, abba\}$$

M

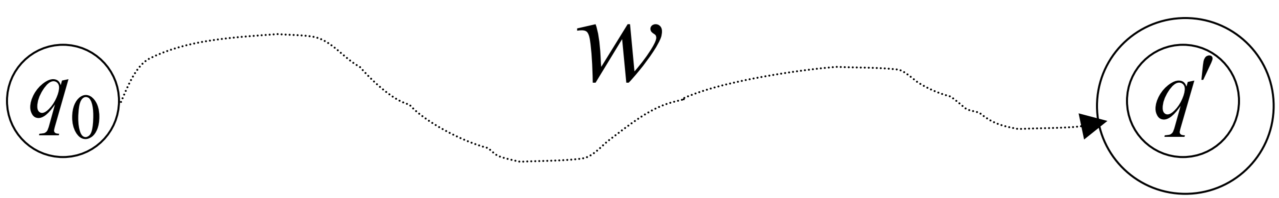


Formally

For a DFA $M = (Q, \Sigma, \delta, q_0, F)$

Language accepted by M :

$$L(M) = \{w \in \Sigma^* : \delta^*(q_0, w) \in F\}$$

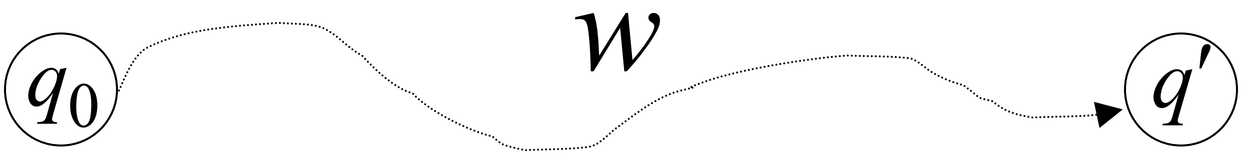


$q' \in F$

Observation

Language rejected by M :

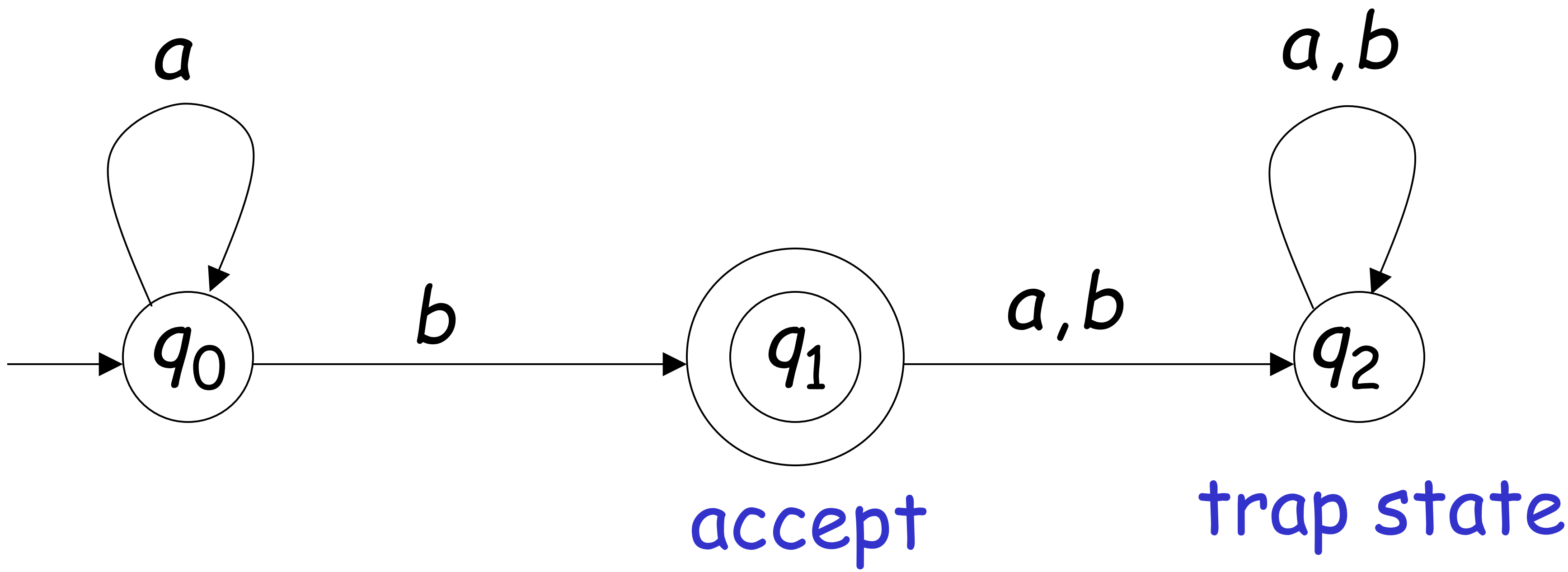
$$\overline{L(M)} = \{ w \in \Sigma^* : \delta^*(q_0, w) \notin F \}$$



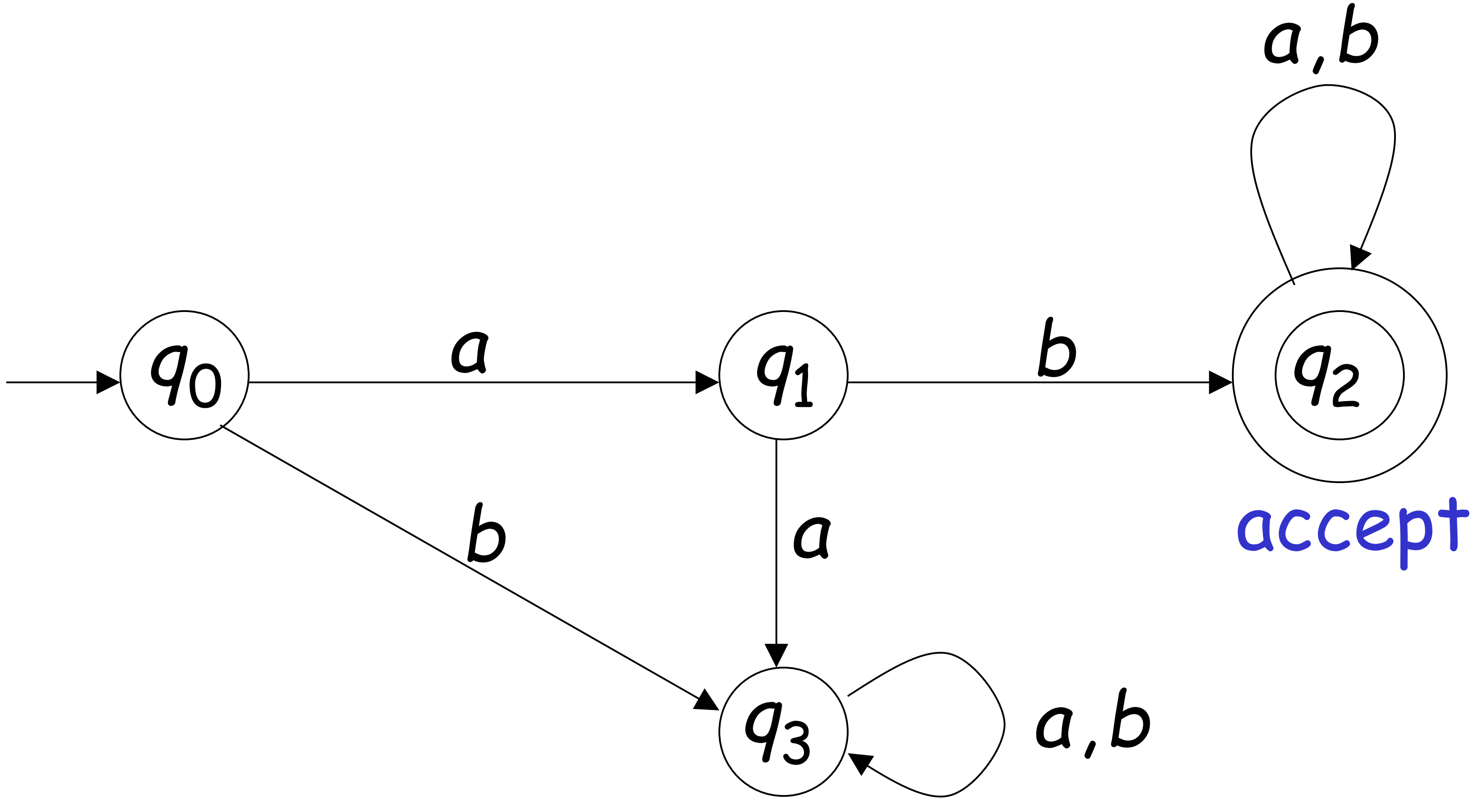
$q' \notin F$

More Examples

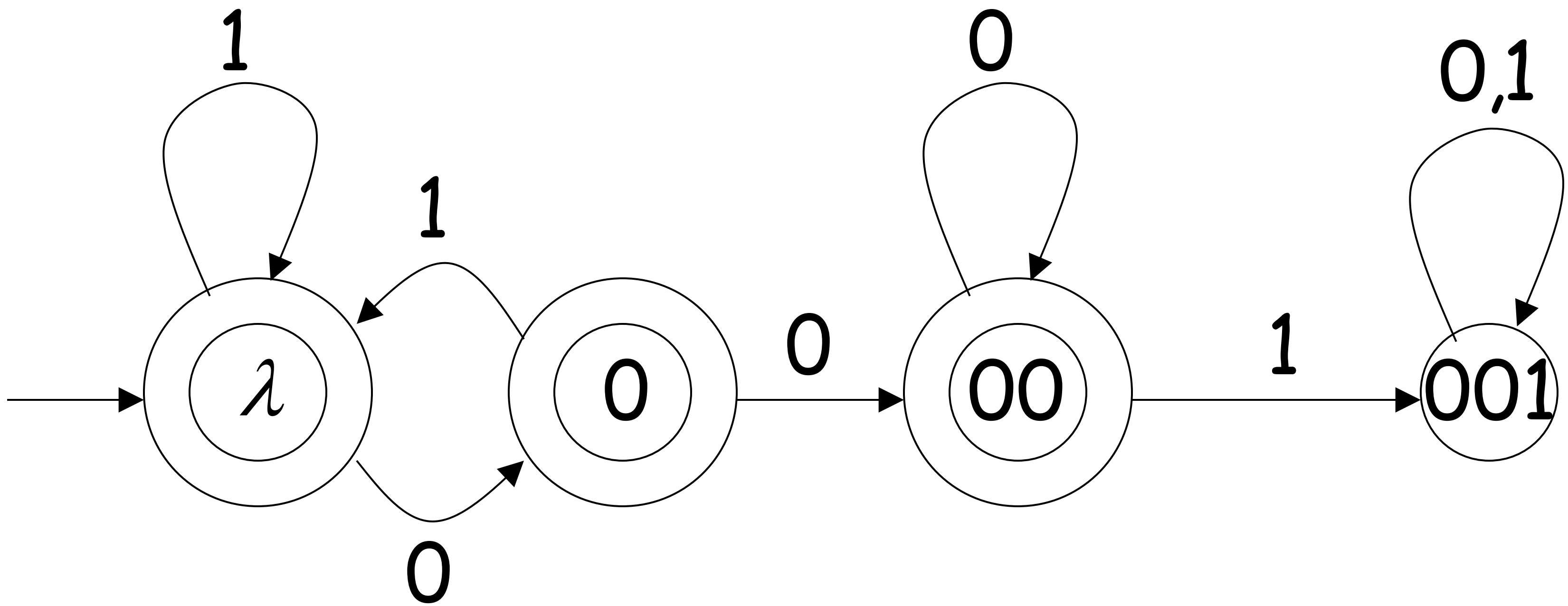
$$L(M) = \{a^n b : n \geq 0\}$$



$L(M) =$ { all strings with prefix ab }



$L(M)$ = { all strings without substring 001 }



Regular Languages

A language L is regular if there is a DFA M such that $L = L(M)$

All regular languages form a language family

Examples of regular languages:

$\{abba\}$ $\quad \{\lambda, ab, abba\}$ $\quad \{a^n b : n \geq 0\}$

{ all strings with prefix ab }

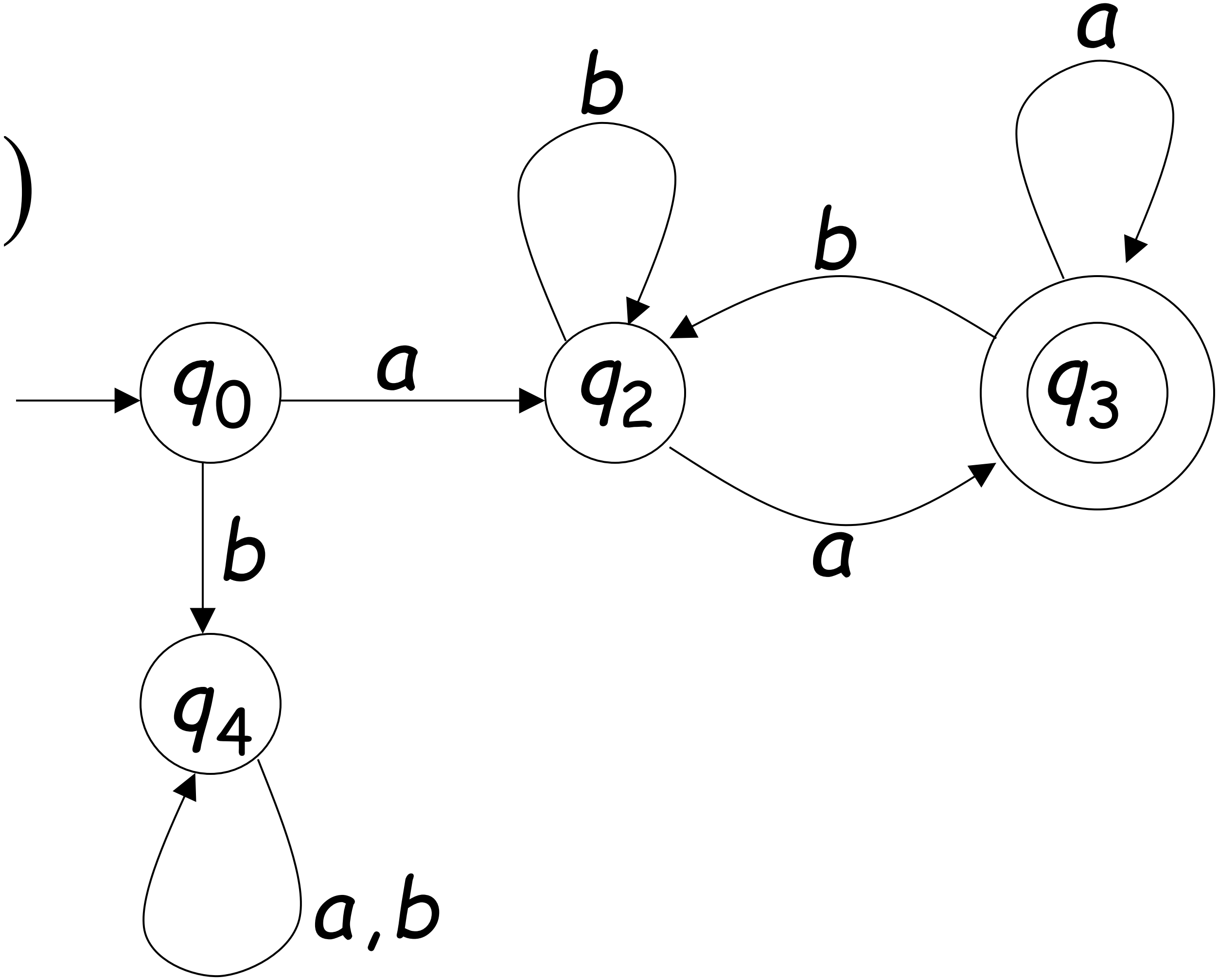
{ all strings without substring 001 }

There exist automata that accept these Languages (see previous slides).

Another Example

The language $L = \{awa : w \in \{a,b\}^*\}$ is regular:

$L = L(M)$



There exist languages which are <u>not</u> Regular:

Example: $L = \{a^n b^n : n \geq 0\}$

There is no DFA that accepts such a language

(we will prove this later in the class)