An Introduction to Formal Languages and Automata

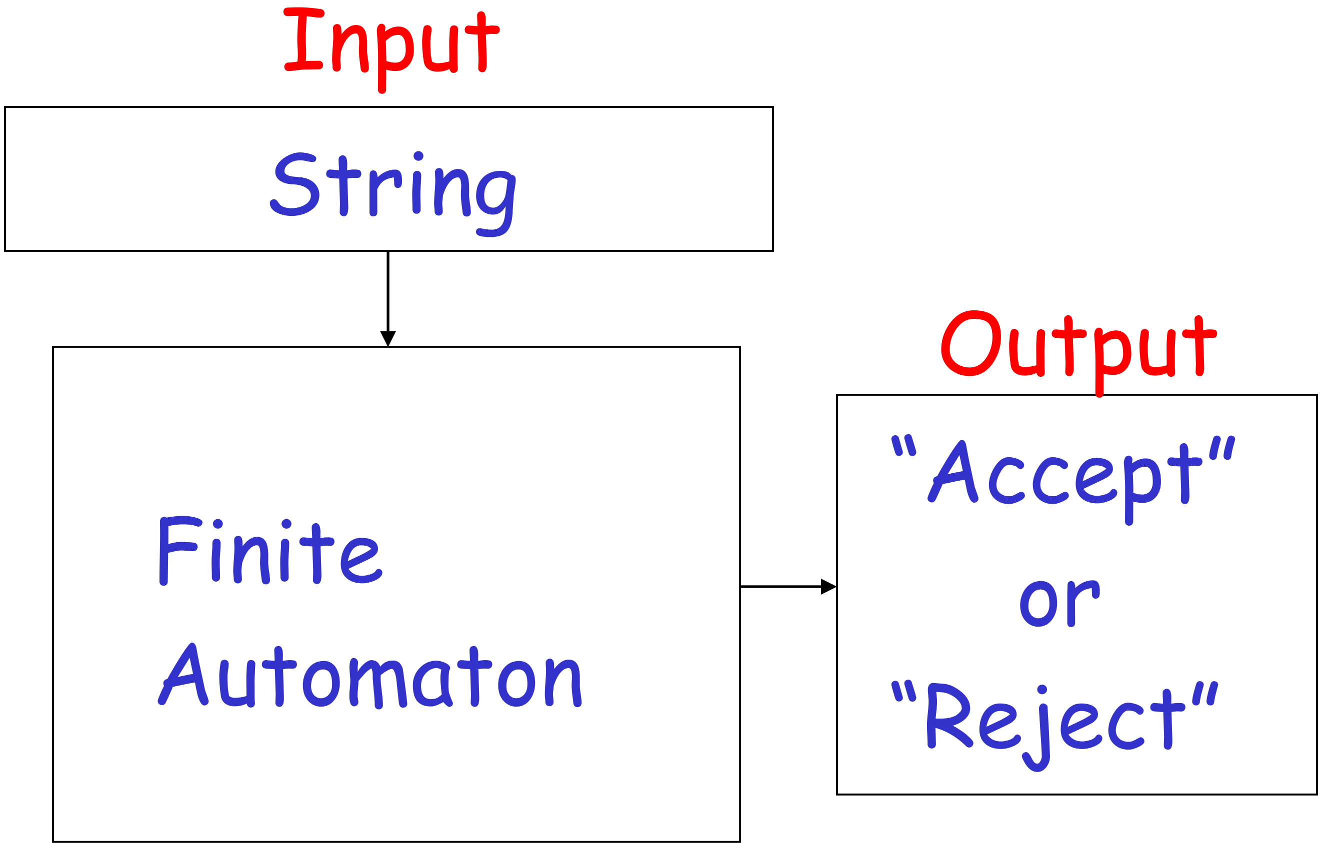
Peter Linz

CHAPTER 2

FINITE AUTOMATA

Module -1	Teaching Hours
INTRODUCTION TO THE THEORY OF COMPUTATION AND FINITE AUTOMATA: Three basic concepts, Some Applications, Deterministic Finite Accepters, Nondeterministic Finite Accepters, Equivalence of Deterministic and Nondeterministic Finite Accepters, Reduction of the Number of States in Finite Automata. **Text Book 1**: Chapter 1:1.2 - 1.3, Chapter 2: 2.1 - 2.4	**08 Hours**

Finite Automaton



Formal Definition

Finite Automaton (FA)

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

Q : set of states

Σ : input alphabet

δ : transition function

q_0 : initial state

F : set of accepting states

Deterministic Finite Automata

Deterministic Finite Automaton (FA)

(QUINTUPLE)

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

Q : set of states

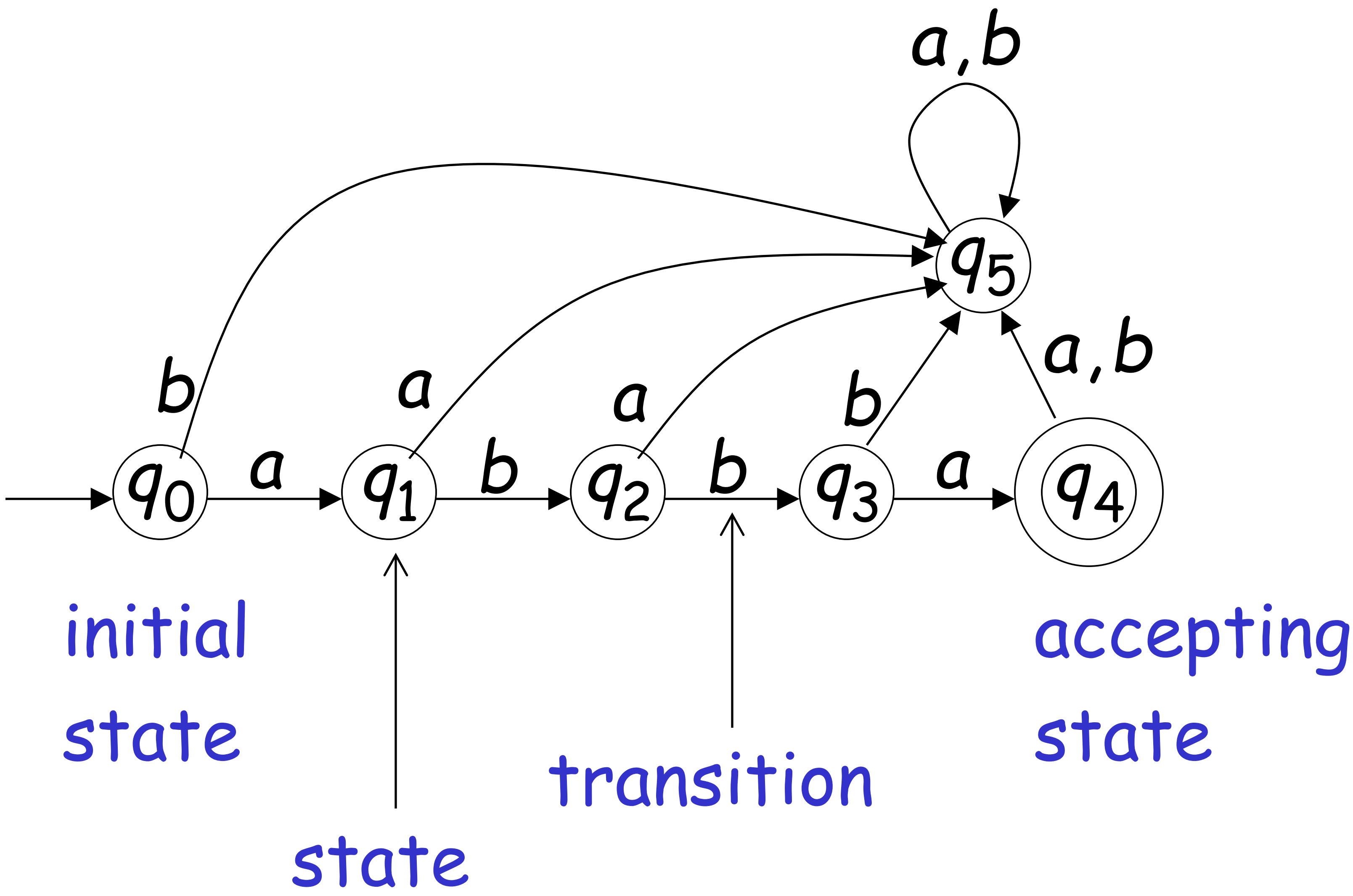
Σ : input alphabet

δ : $Q \times \Sigma \rightarrow Q$ - transition function E.g. $\delta(q0,a)=q1$

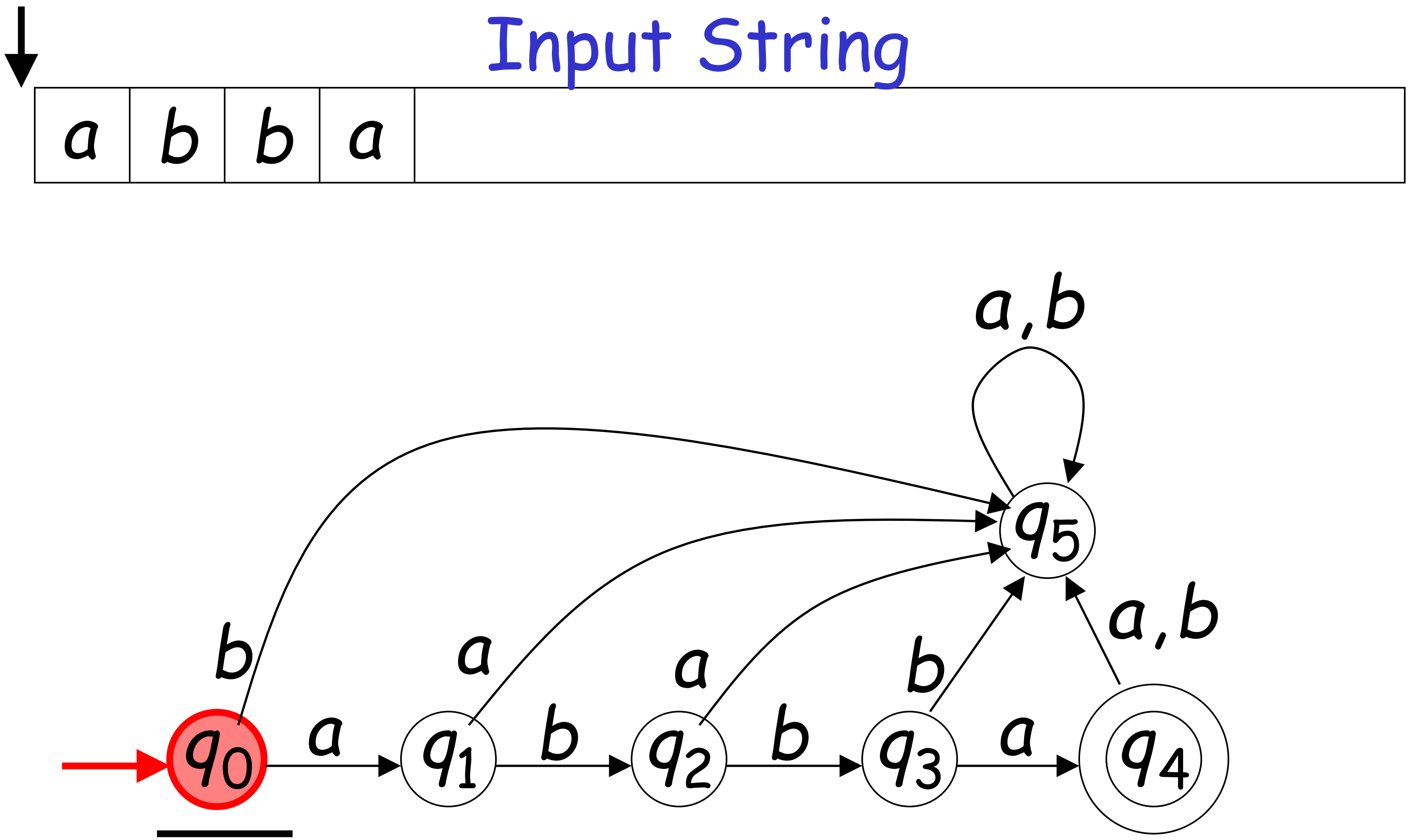
q_0 : initial state

F : set of accepting states

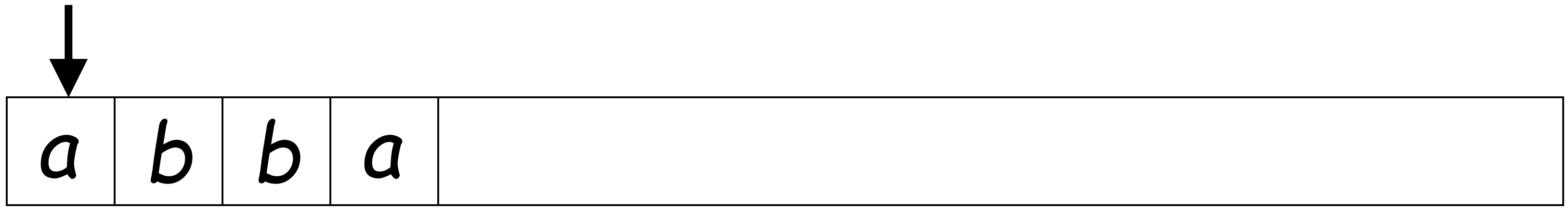
Transition Graph

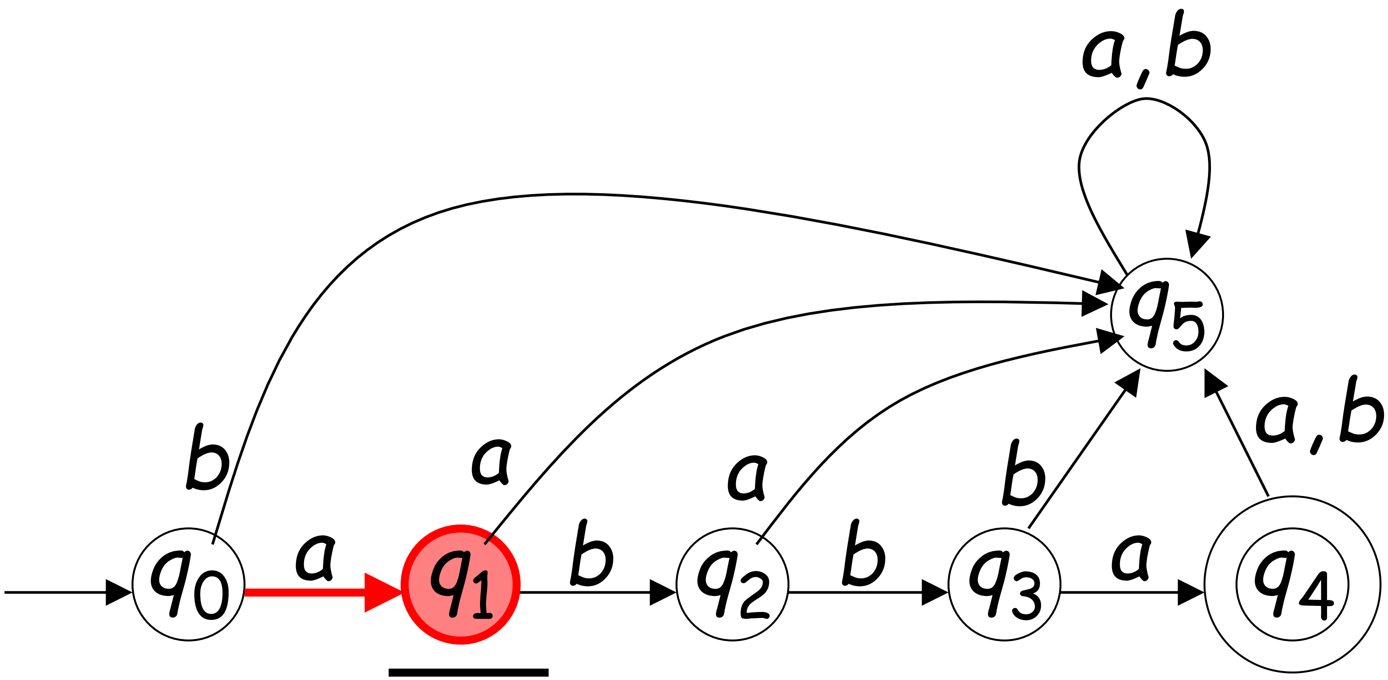


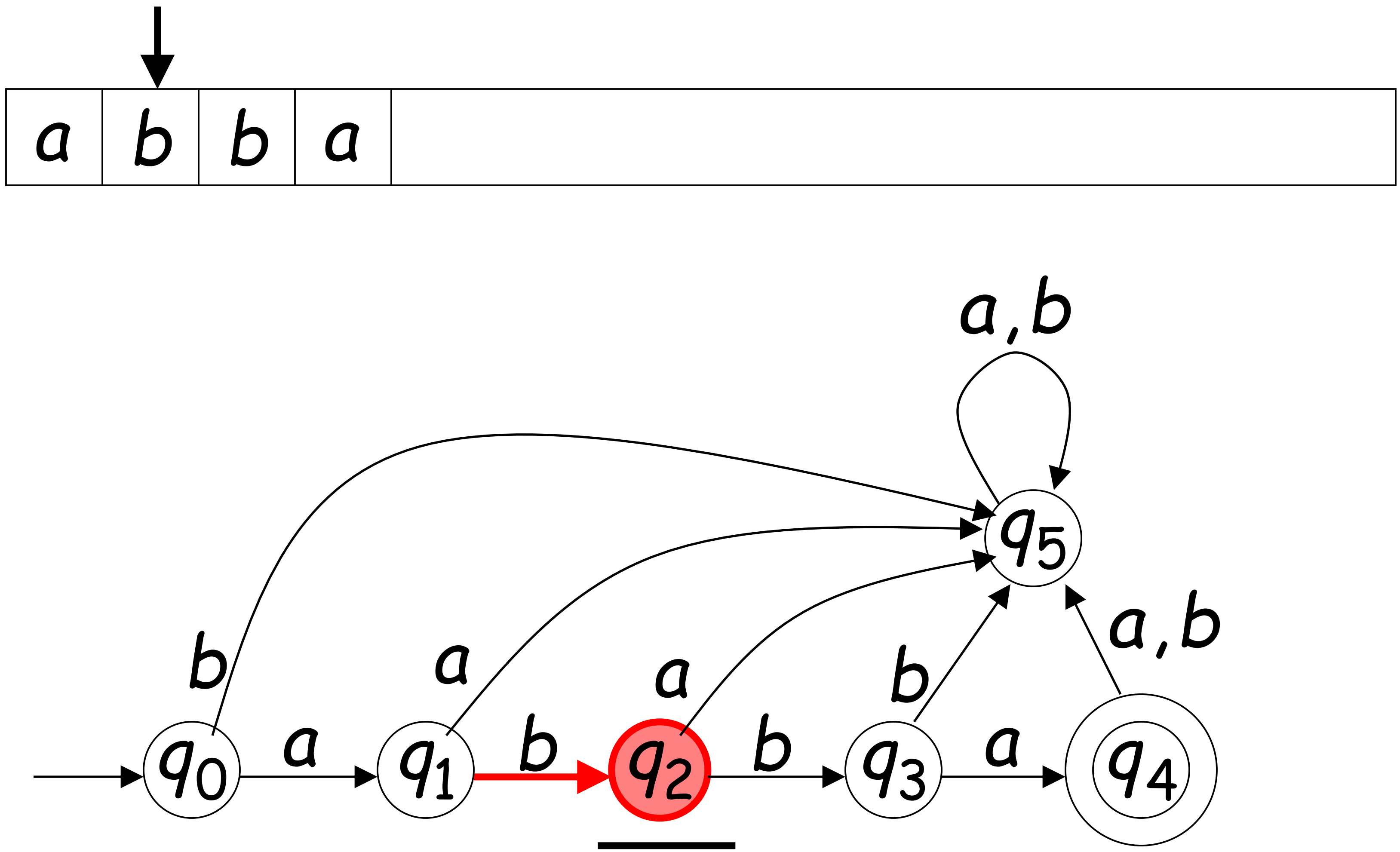
Initial Configuration

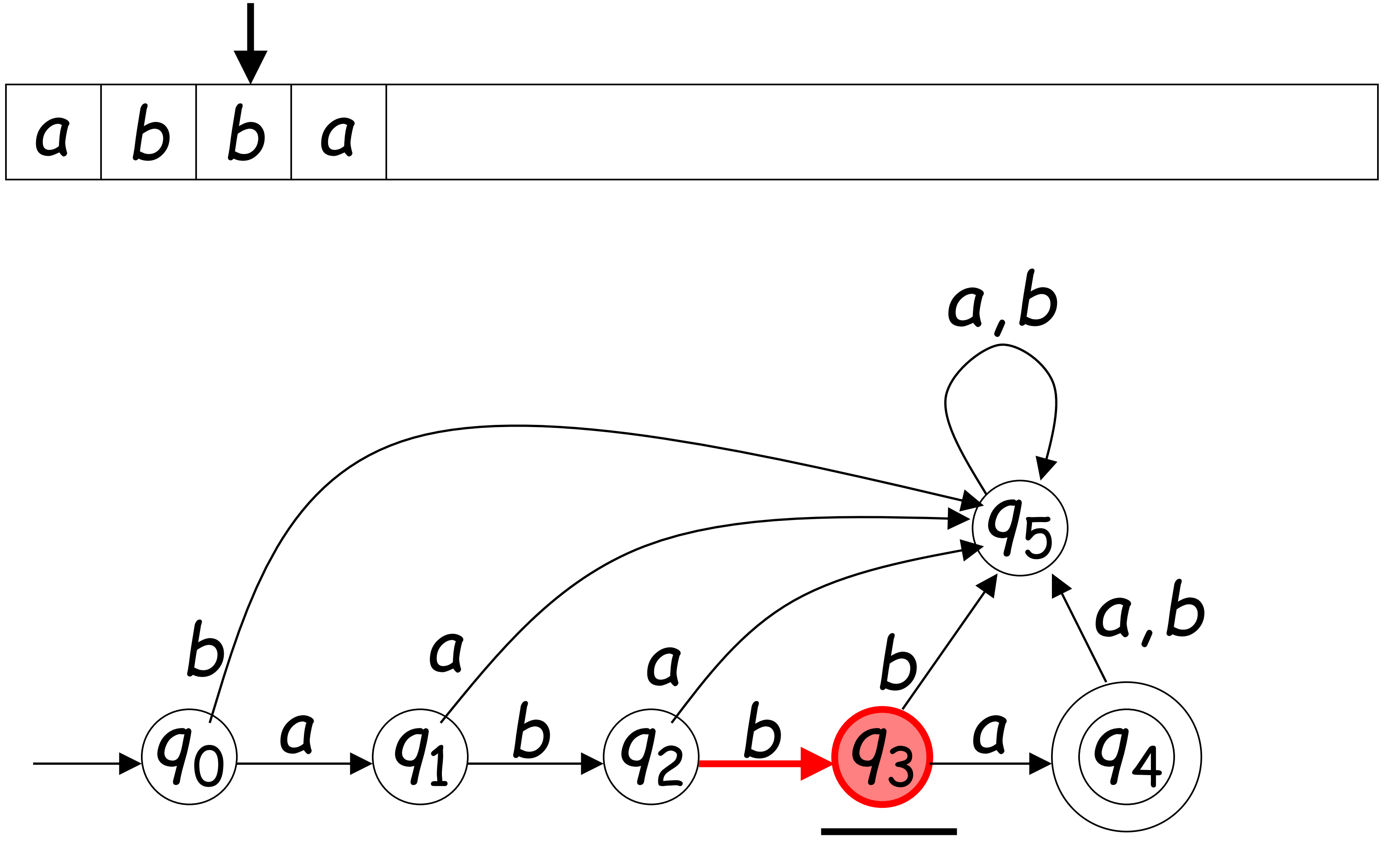


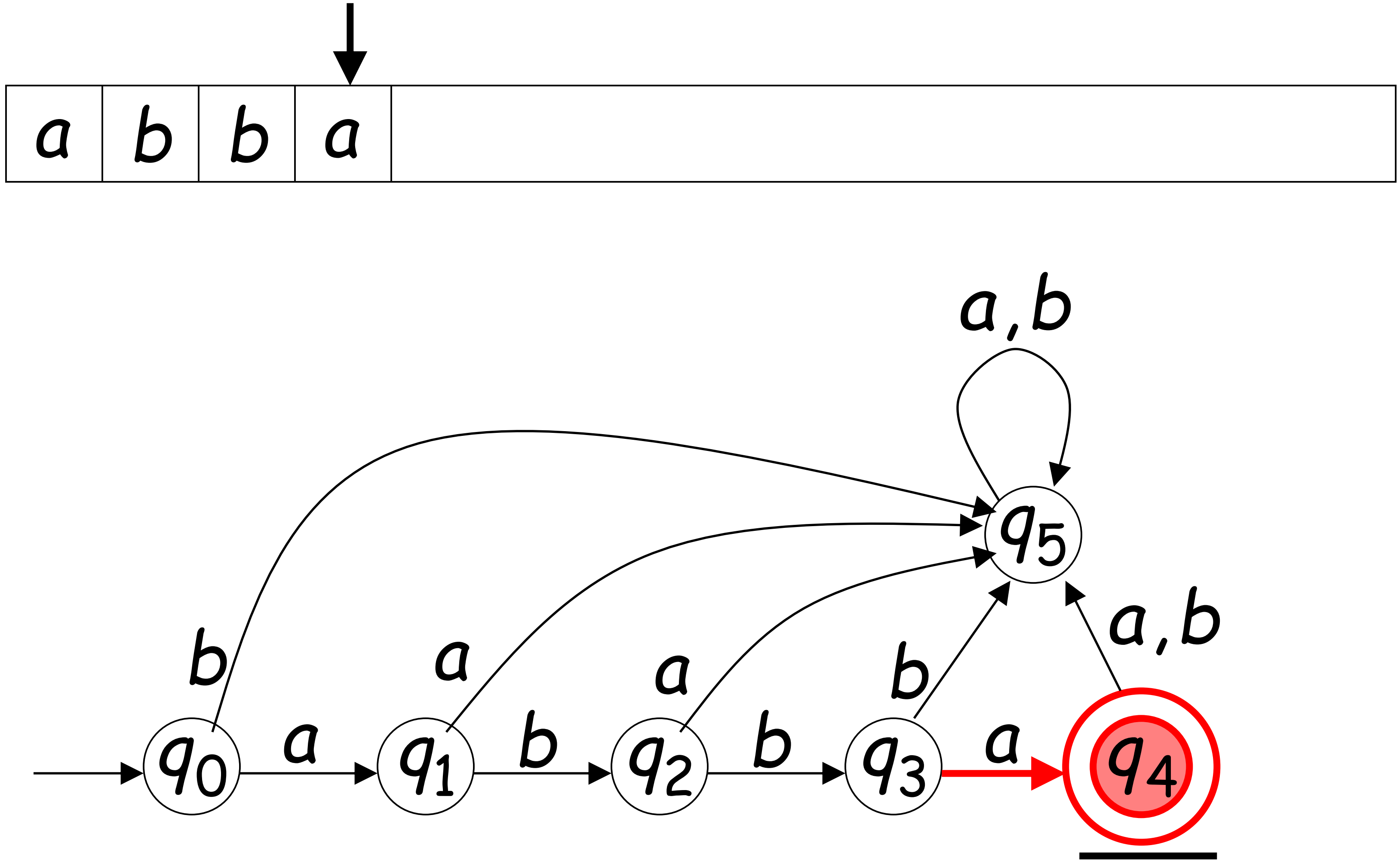
Reading the Input





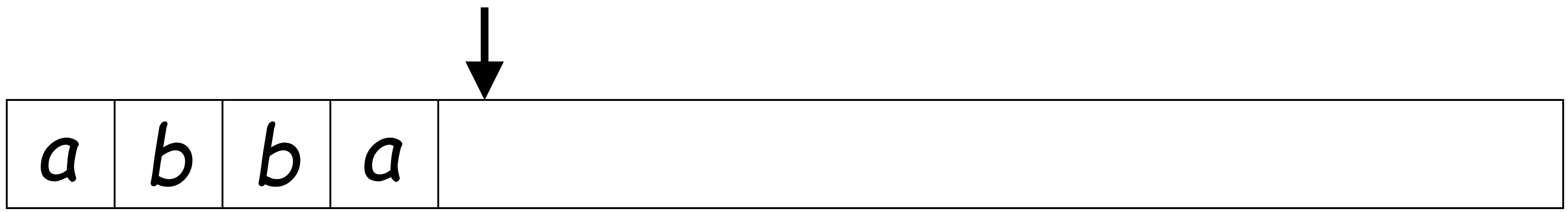

a
b
b
a
a,b
q5
b
a
a
b
a,b
q0
a
q1
b
q2
b
q3
a
q4

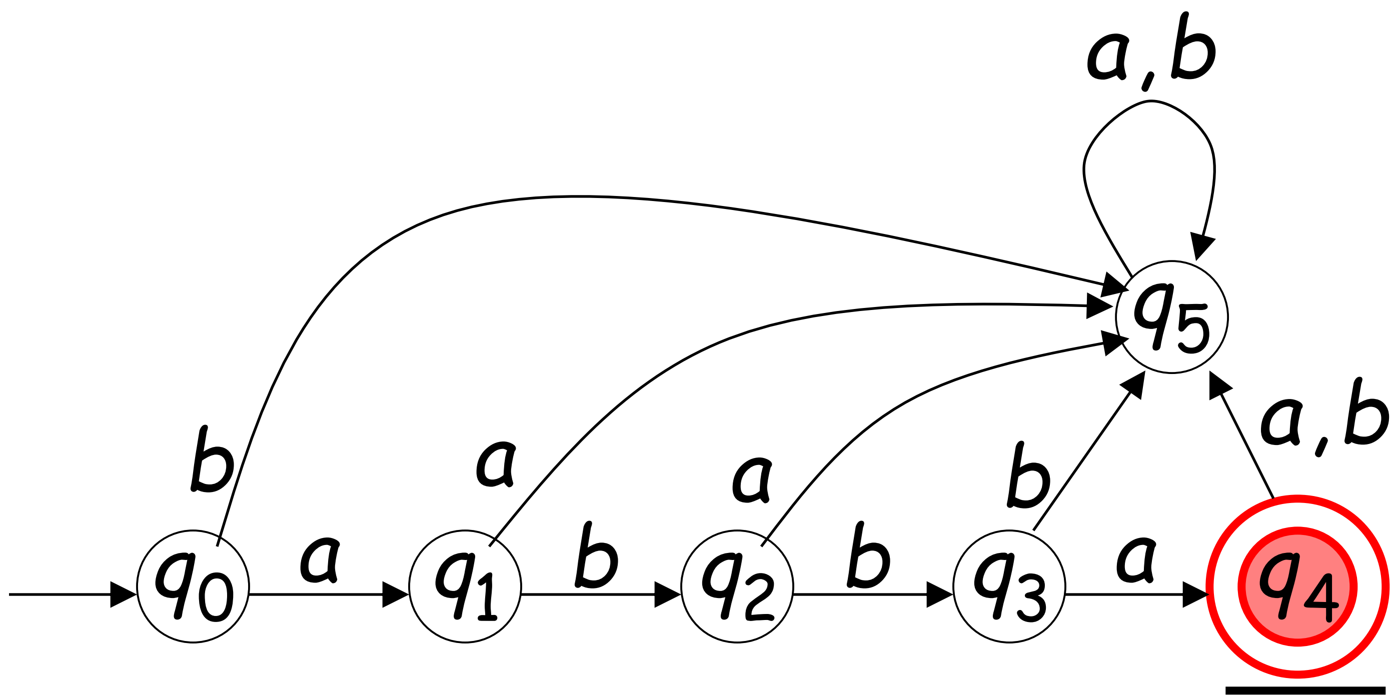

a
b
b
a
a,b
q5
b
a
a
b
a,b
q0
a
q1
b
q2
b
q3
a
q4



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

Input finished

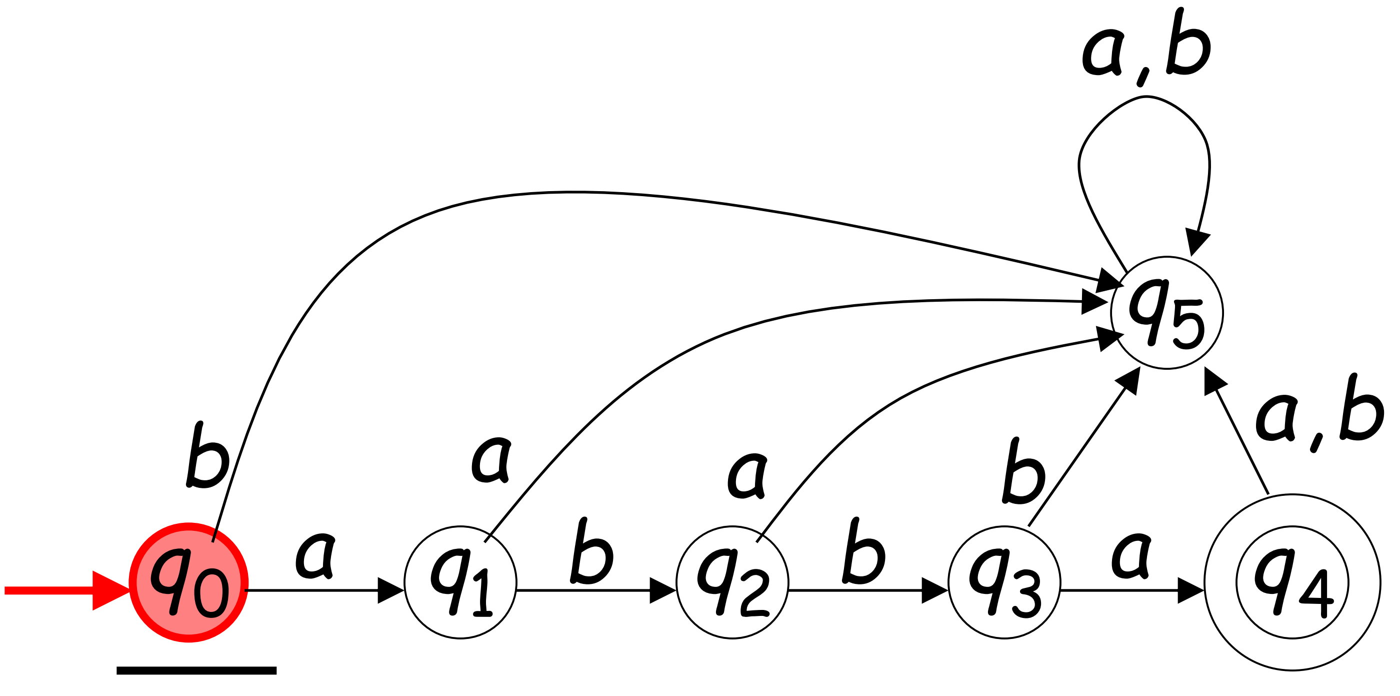


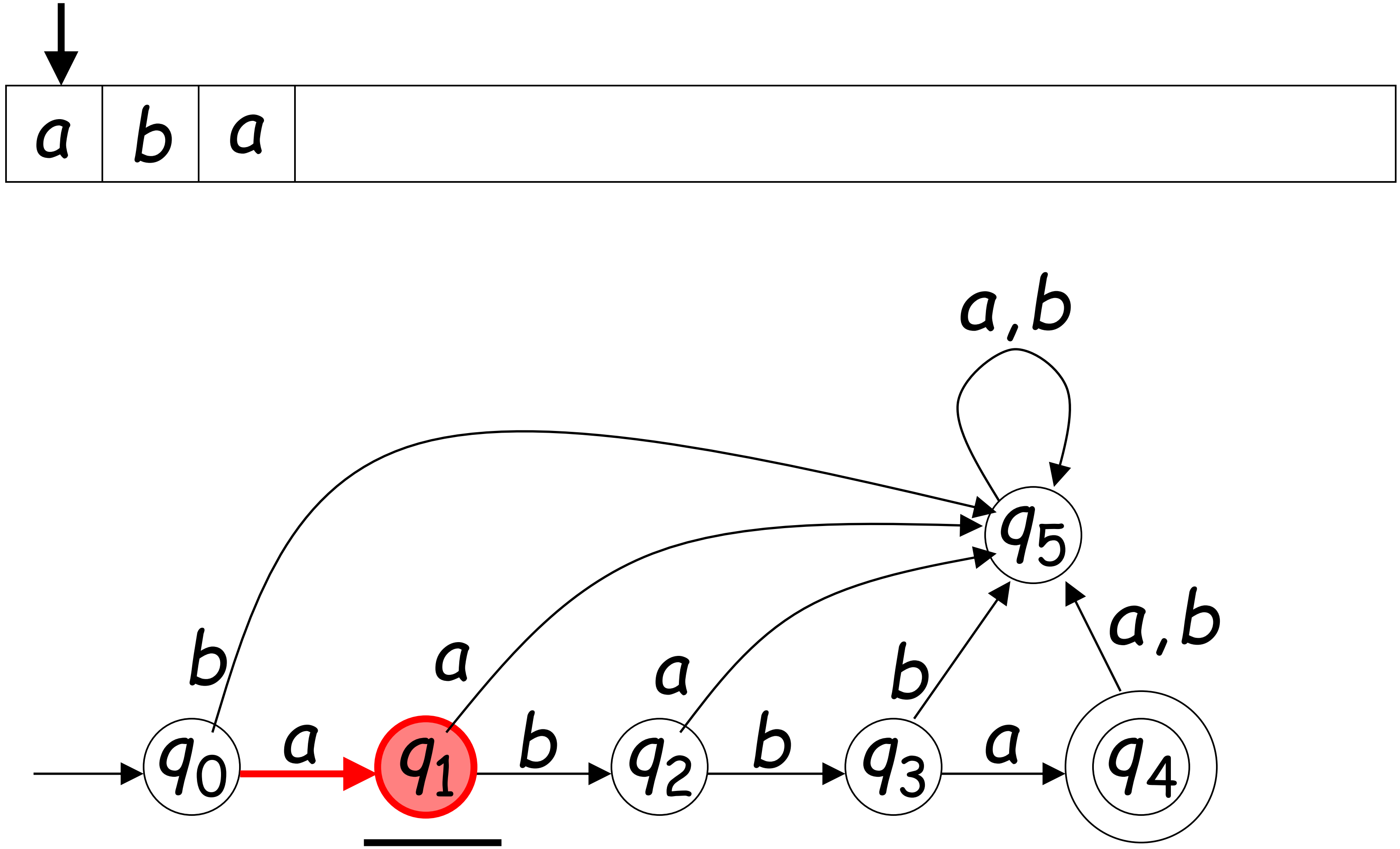


accept

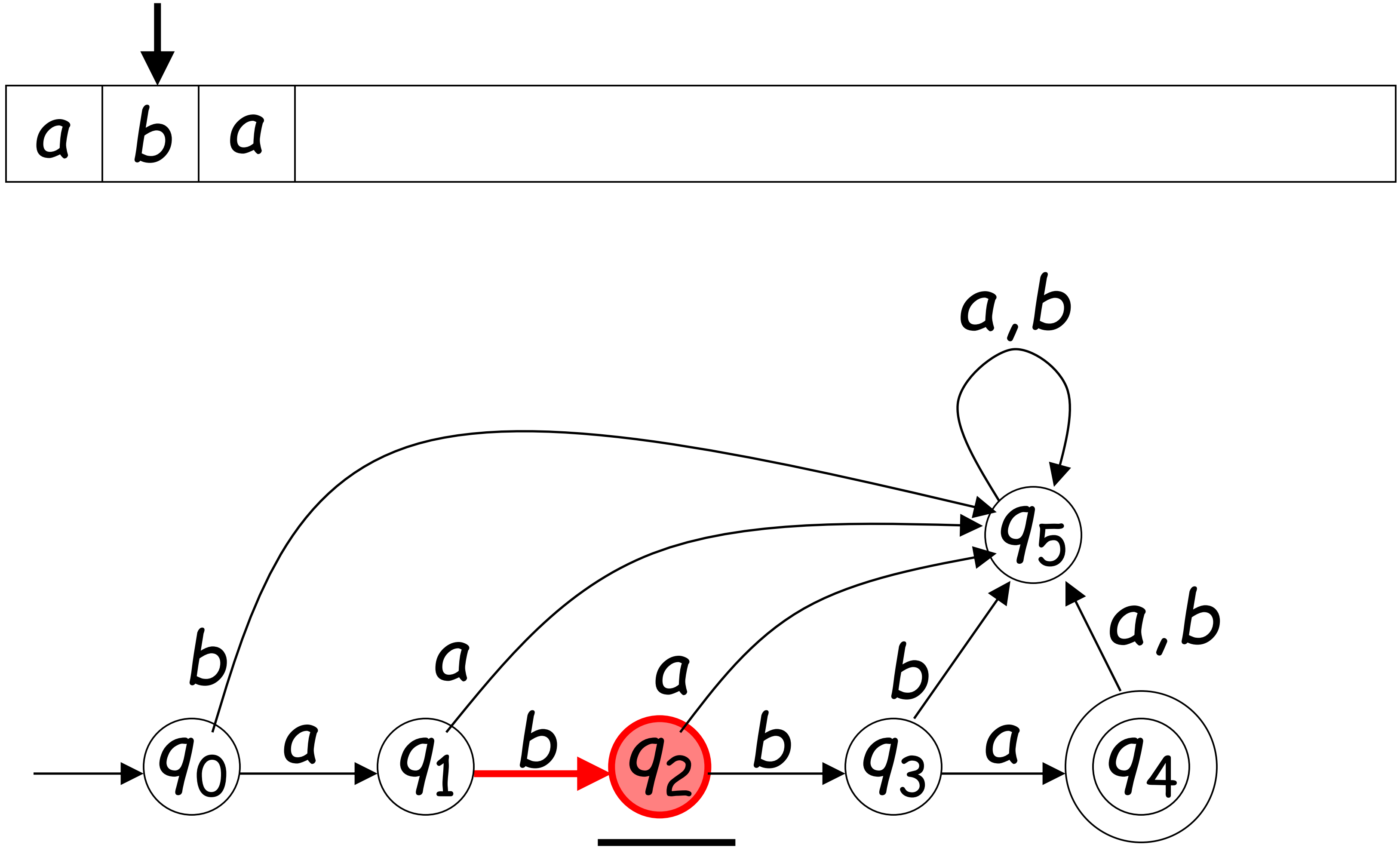
Rejection

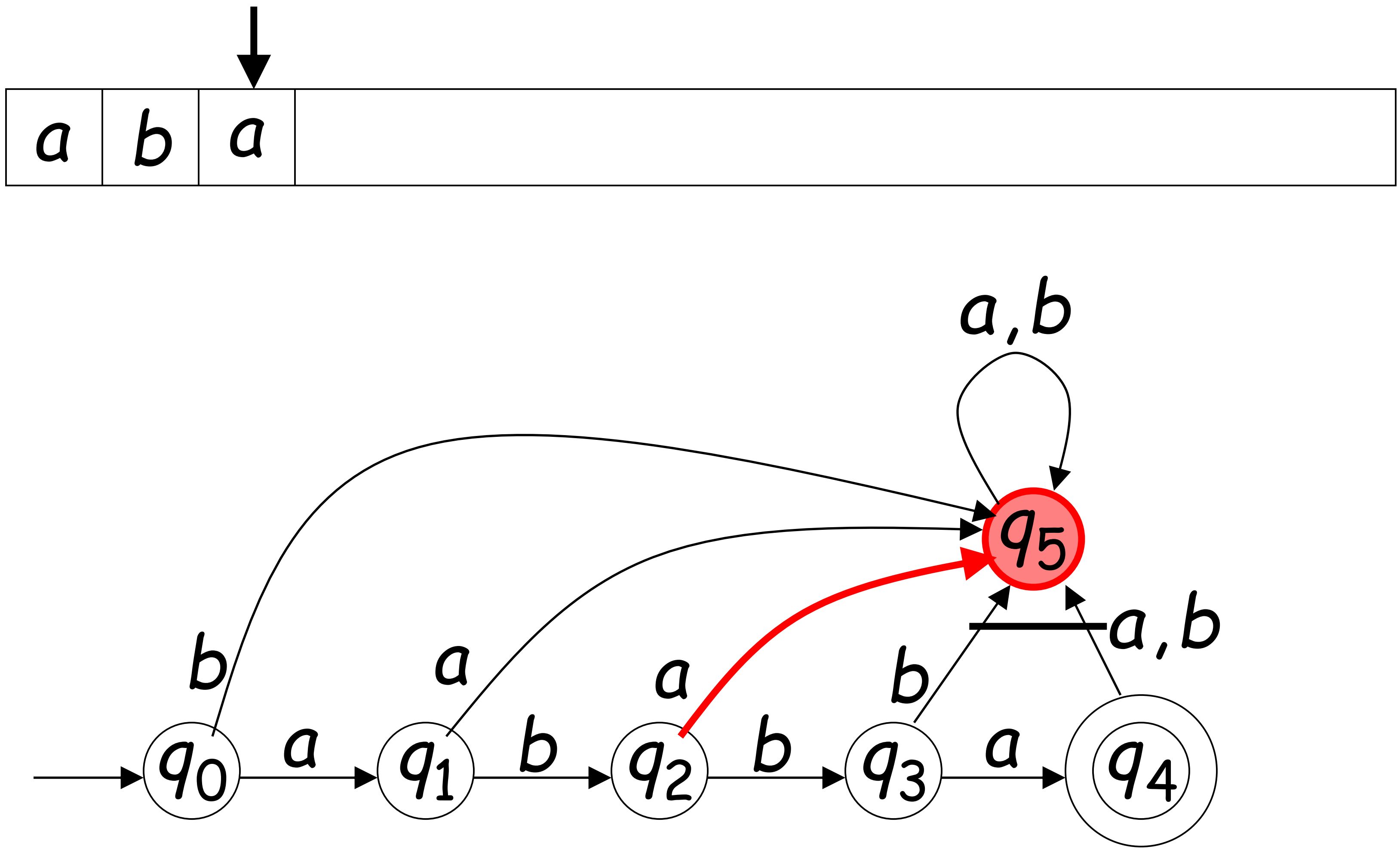
a	b	a	



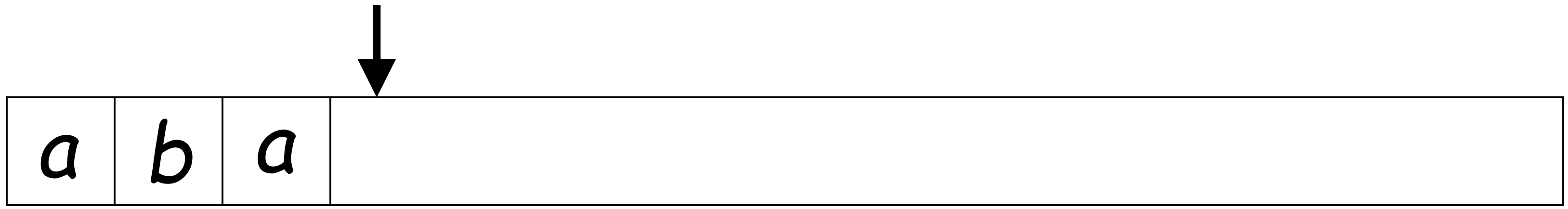


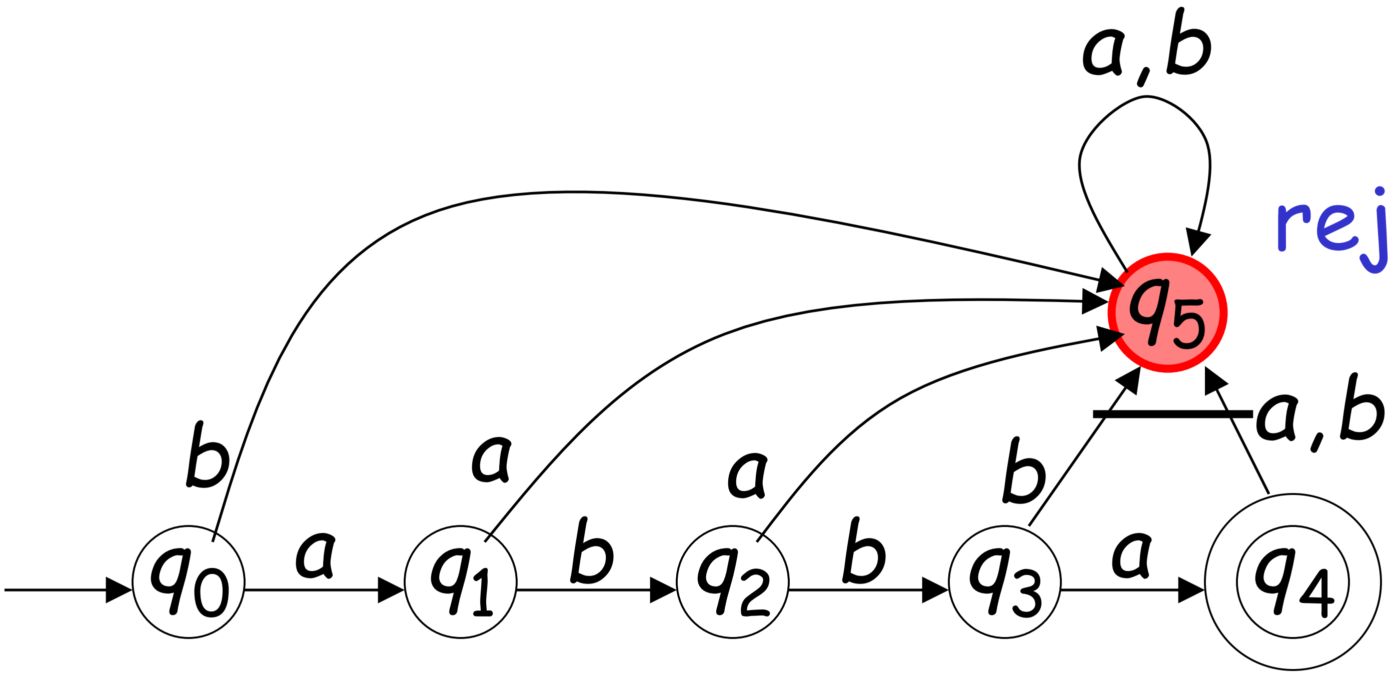
a
b
a
a,b
q5
b
a
a
b
a,b
q0
a
q1
b
q2
b
q3
a
q4


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


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

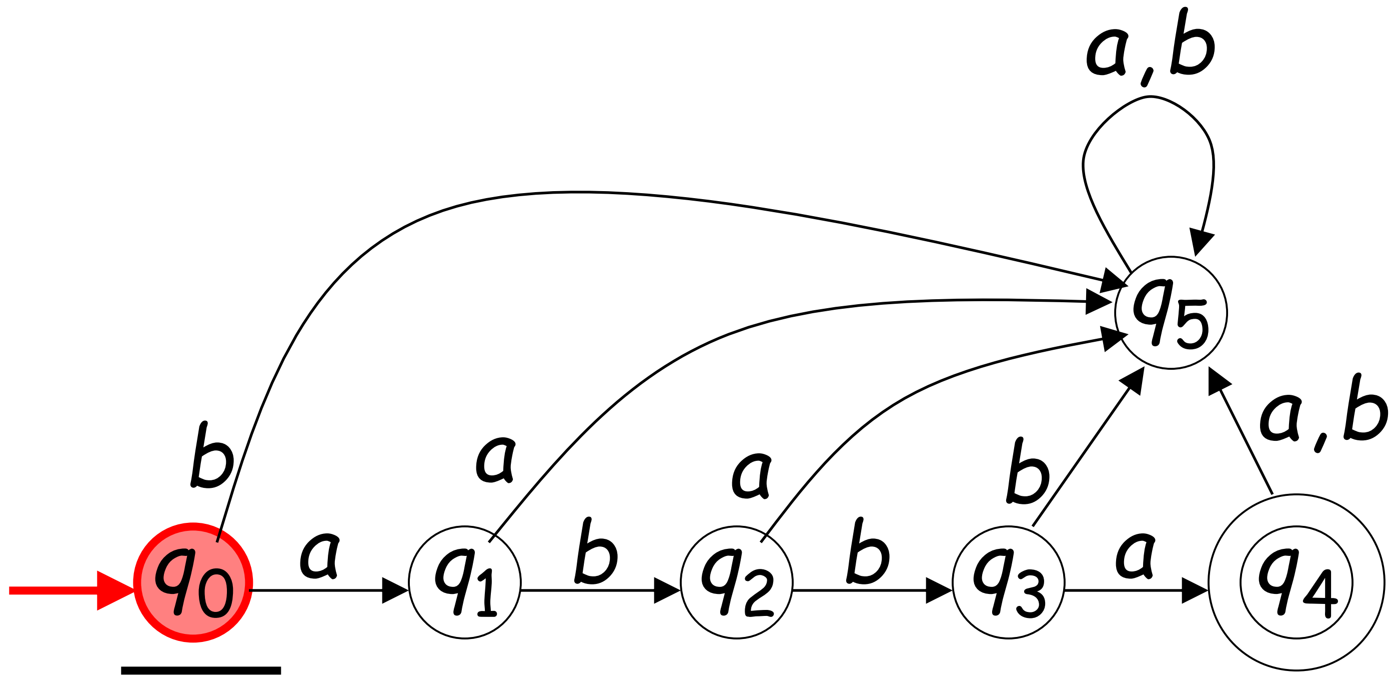
Input finished



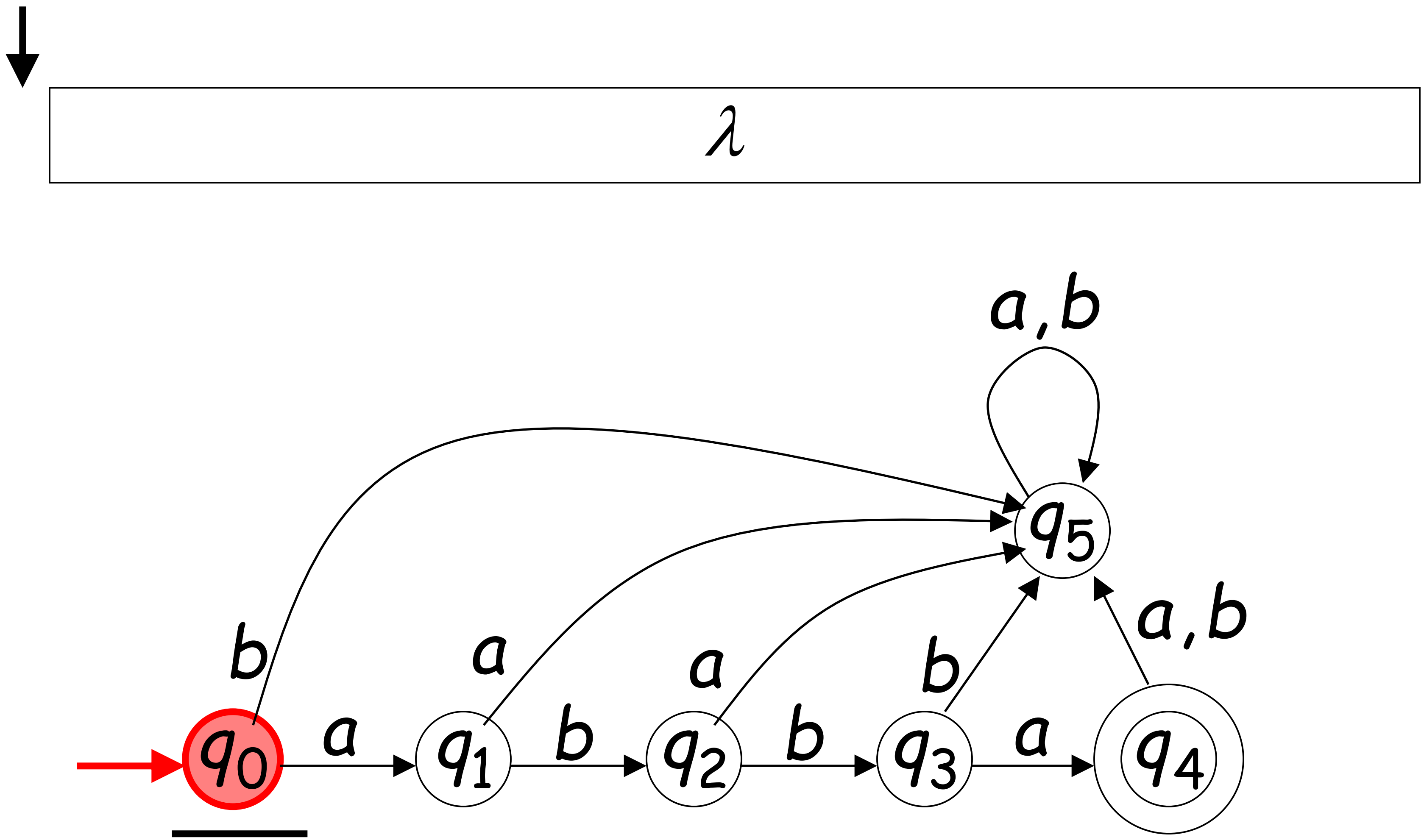


Acceptance or Rejection?

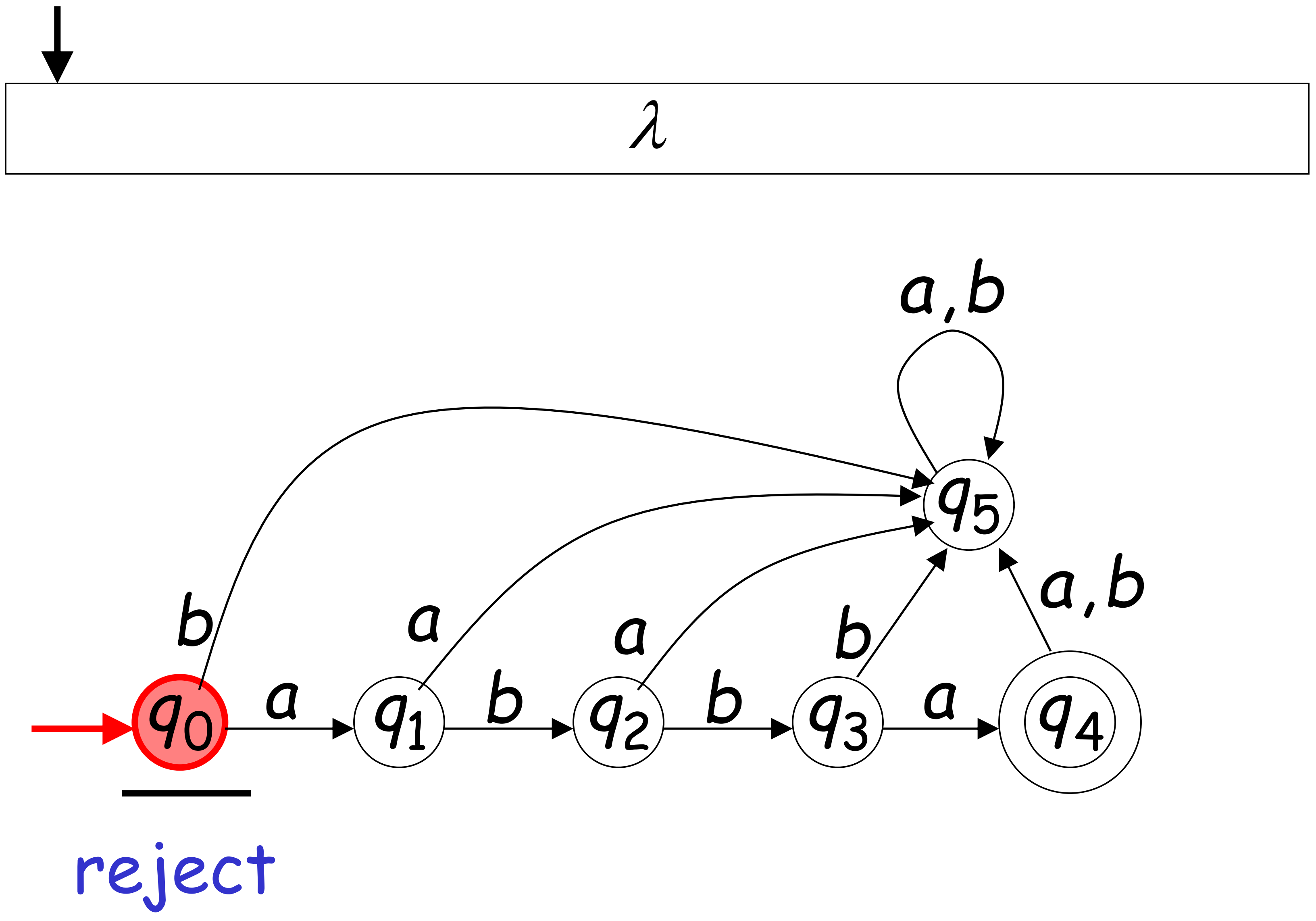
λ



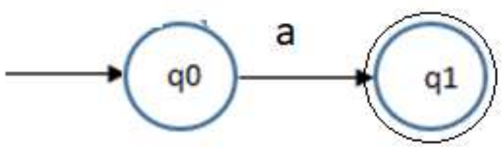
Initial State



Rejection



- To visualize and represent FA , Transition Graph is used.
- Here in graph, vertices represents states and edges represent transition.
- The labels on the vertices are name of the state and label on edges represent the input symbol.
- In the below given graph, qo is the initial state, vertices labelled as q0 and q1.
- An edge from q0 to q1 represents a transition $\delta(q_0, a) = q_1$
- The initial state will be shown by incoming unlabelled arrow not originating from any vertex.
- Final states are shown with double circle.



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

DFA M

DFA

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

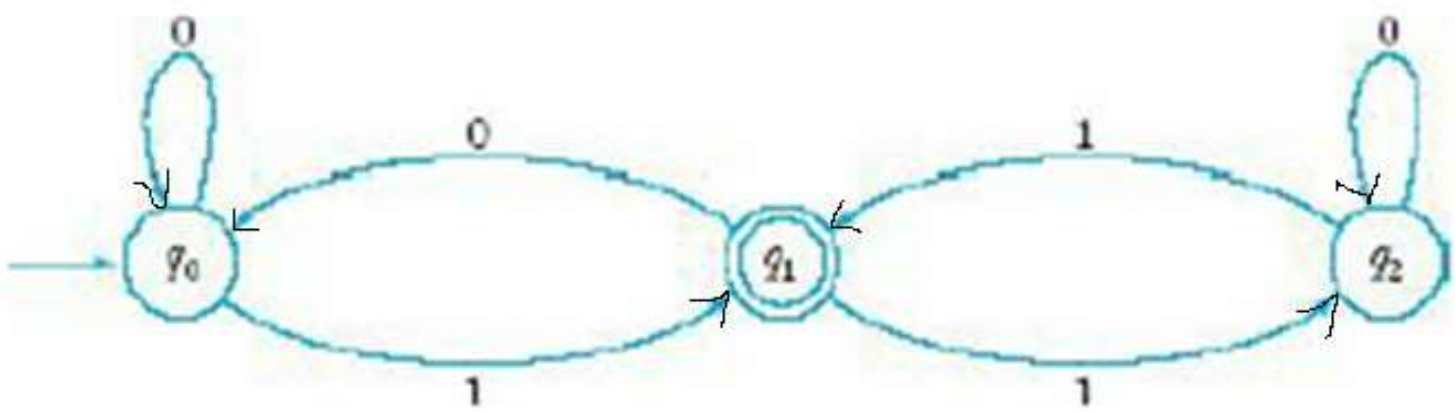
$M = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_1\}),$

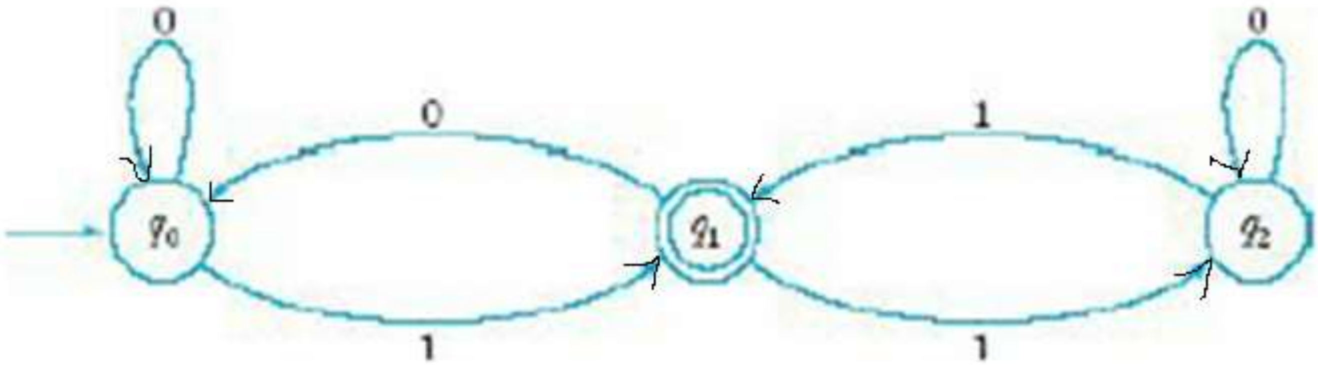
Q ➔ {q0, q1, q2}

Σ → {0,1} δ → transition δ(q0, 1) = q1

q0 → Initial state

F → Accepter state





where δ are given by

$$
\begin{aligned}
\delta(q_0, 0) &= q_0, & \delta(q_0, 1) &= q_1, \\
\delta(q_1, 0) &= q_0, & \delta(q_1, 1) &= q_2, \\
\delta(q_2, 0) &= q_2, & \delta(q_2, 1) &= q_1.
\end{aligned}
$$

The automaton will accept the strings 101, 0111, and11001, but not 100 or 1100

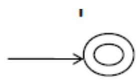
Some Initial DFAs…

1. DFA to accept an empty language L = { Φ }
2. DFA to accept an empty string L = { Λ }
3. DFA to accept exactly one "a"
4. DFA to accept zero or more "a"
5. DFA to accept at least one "a"
6. DFA to accept one "a" or one "b"

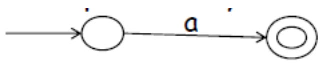
Some Initial DFAs…

1. DFA to accept an empty language L = { Φ }

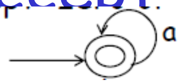
2. DFA to accept an empty string L = { Λ } - If q0 is an accepting state, the automaton accepts the empty string.



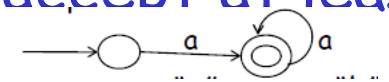
3. DFA to accept exactly one “a”



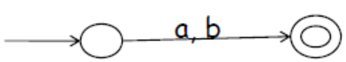
1. DFA to accept zero or more “a”



2. DFA to accept at least one “a”



3. DFA to accept one “a” or one “b”



Extended Transition Function

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

Here the second argument of δ^* is a **string**, rather than a single symbol

For example….

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

$$\delta(q_1, b) = q_2,$$

then where ab is a string

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

We can define δ^* recursively by

$$\delta^* (q, \lambda) = q \qquad (1)$$

$$\delta^* (q, wa) = \delta (\delta^* (q, w), a) \qquad (2)$$

Contd…

$$\delta^*(q_0, ab) = \delta(\delta^*(q_0, a), b). \qquad (3)$$

$$\begin{aligned}\delta^*(q_0, a) &= \delta(\delta^*(q_0, \lambda), a) \\ &= \delta(q_0, a) \\ &= q_1.\end{aligned}$$

So…. Substitute in Eq. (3)

$$\delta^*(q_0, ab) = \delta(q_1, b) = q_2.$$

DFA contd…

- The language accepted by a dfa $M = (Q, \Sigma, \delta, q0, F)$ is the set of all strings on Σ accepted by M. In formal notation,

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

- Non-acceptance means that the DFA stops in a non-final state, so that

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

DFA contd…

- Consider the dfa in Figure

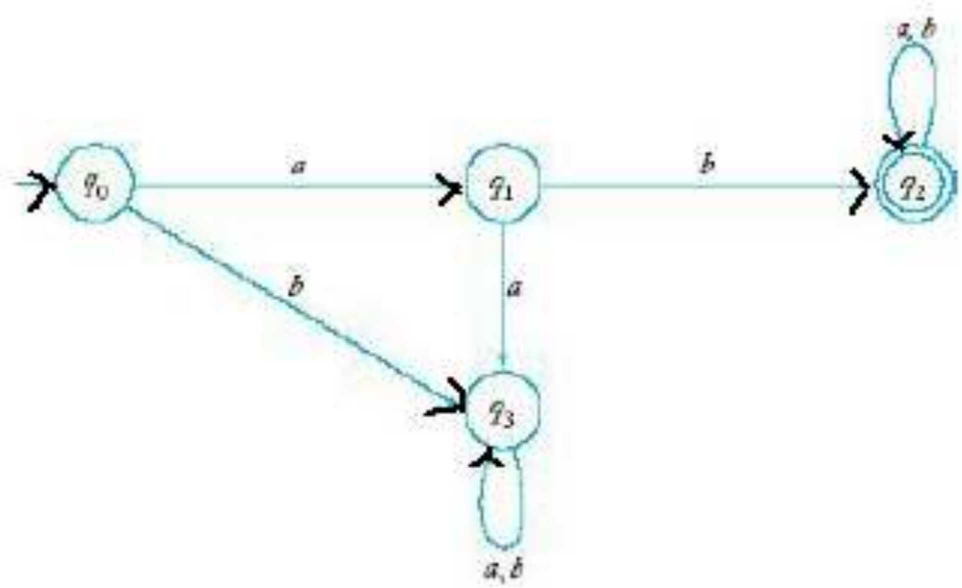
Transition Table

	a	b
q_0	q_0	q_1
q_1	q_2	q_2
q_2	q_2	q_2

- Language for this is $L = \{a^n b: n \geq 0\}$.
- If the string is accepted ..it will go to accepter state that is q1 otherwise it will go to trap state q2 from where it cannot escape.

DFA contd…

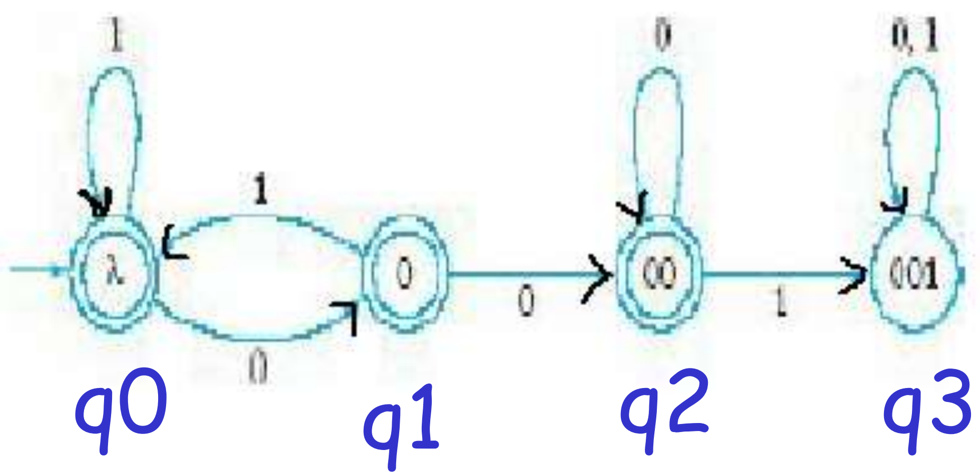
- Find a deterministic finite accepter that recognizes the set of all strings on Σ= {*a*,*b*} starting with the prefix *ab*.



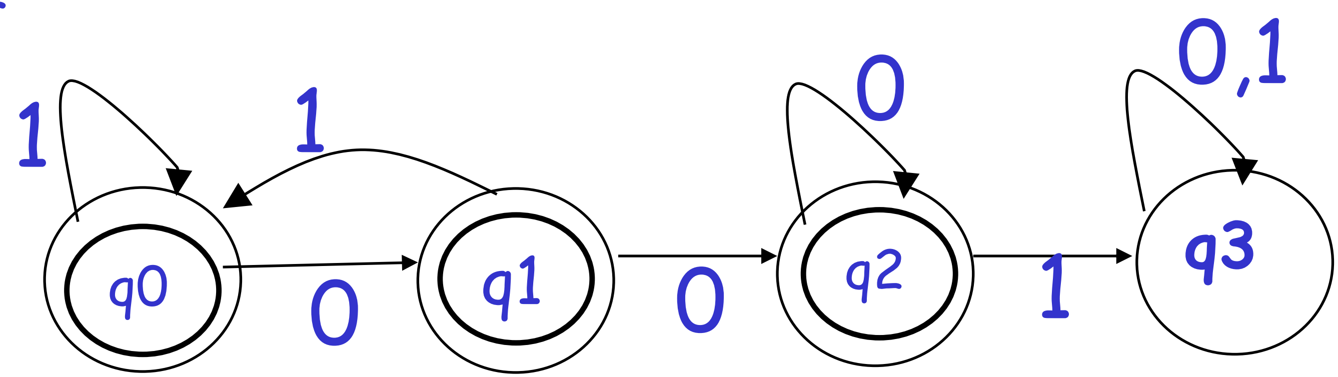
DFA contd…

- Find a dfa that accepts all the strings on {0,1}, except those containing the substring 001.

- Fig 1

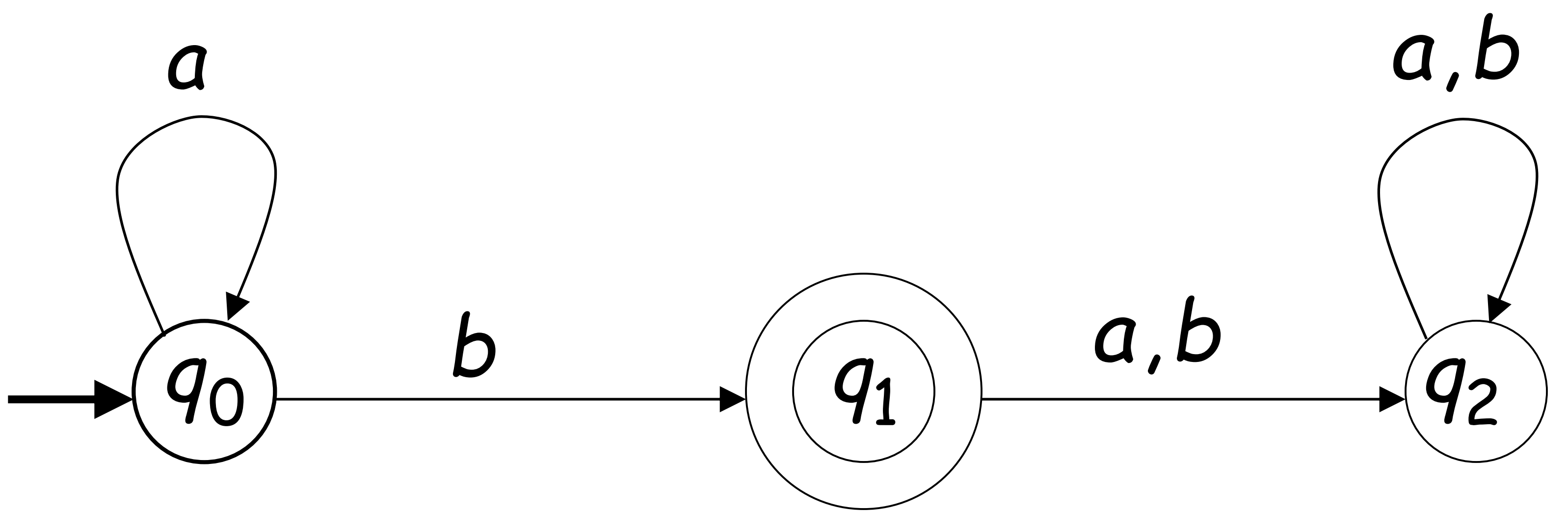


- Fig 2

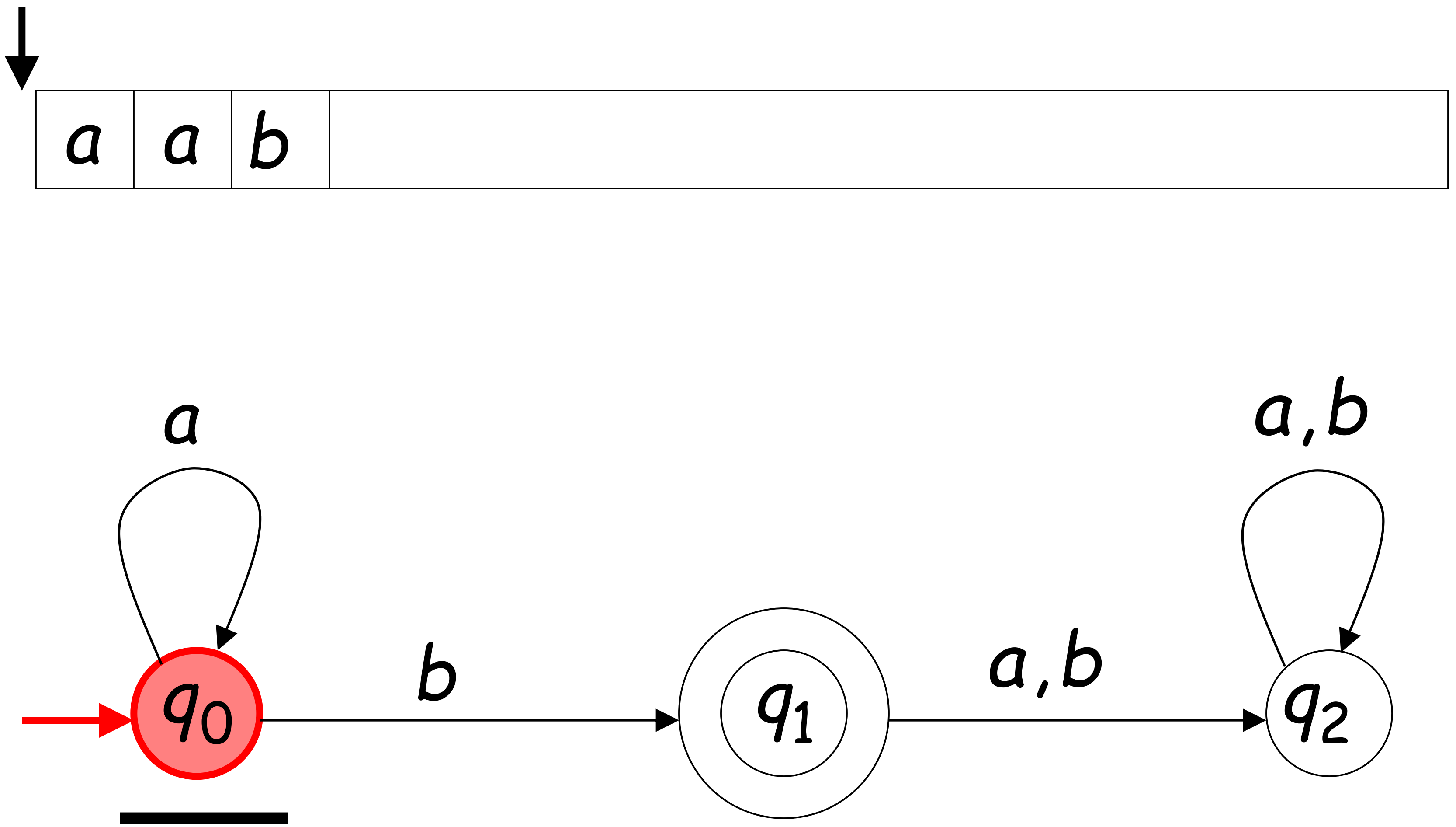


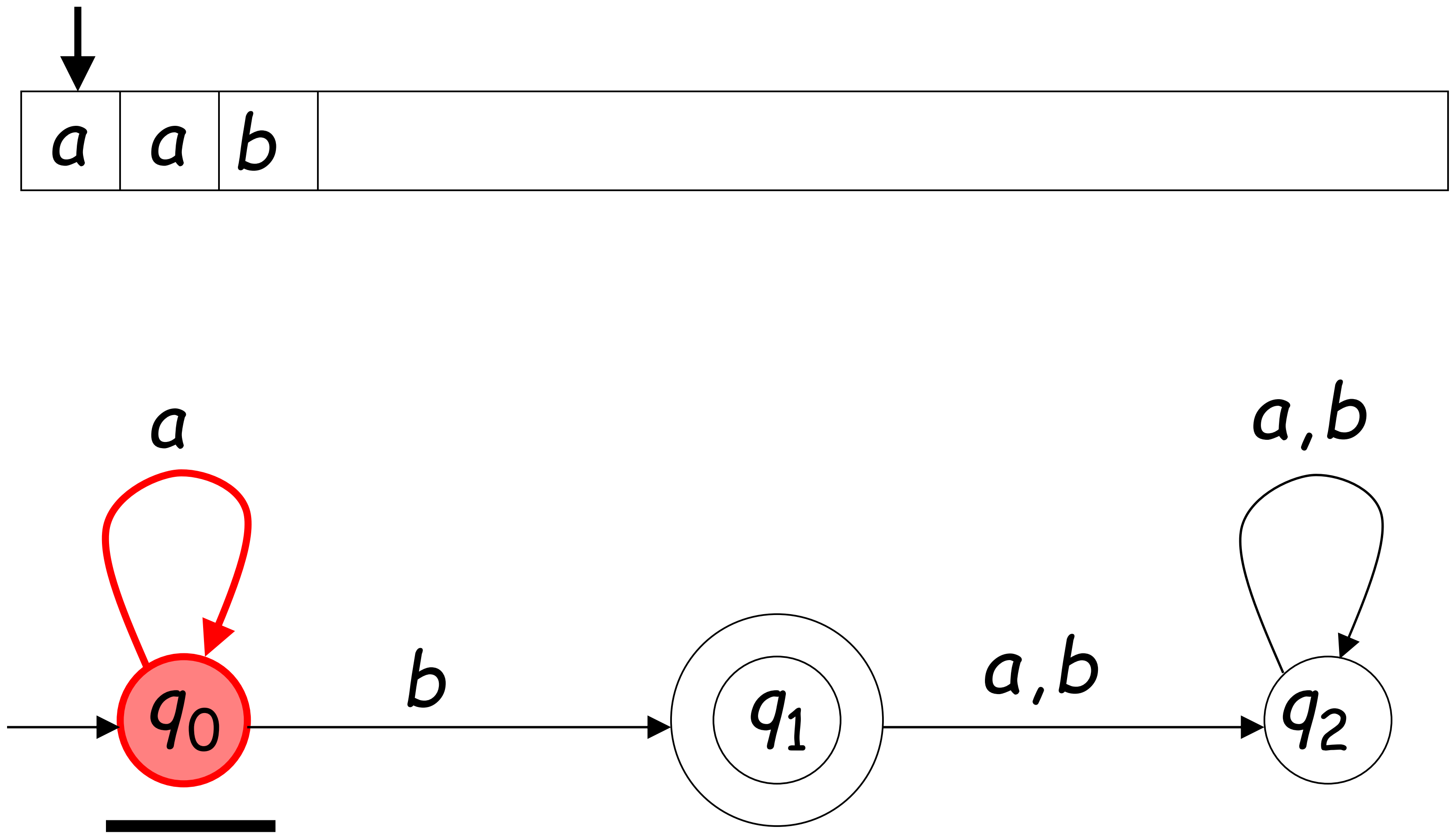
- Fig 1 is same as Fig 2

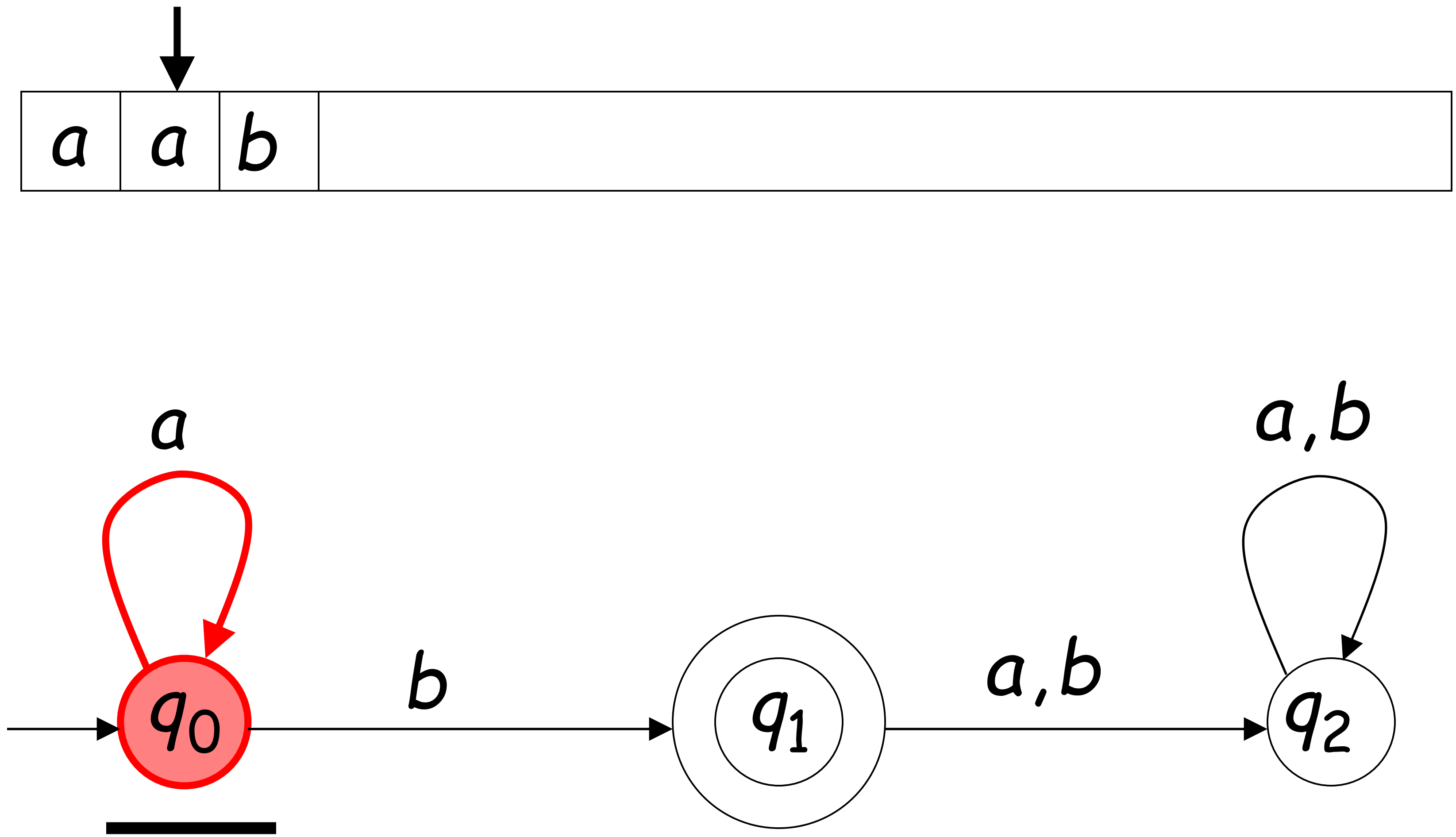
Language?

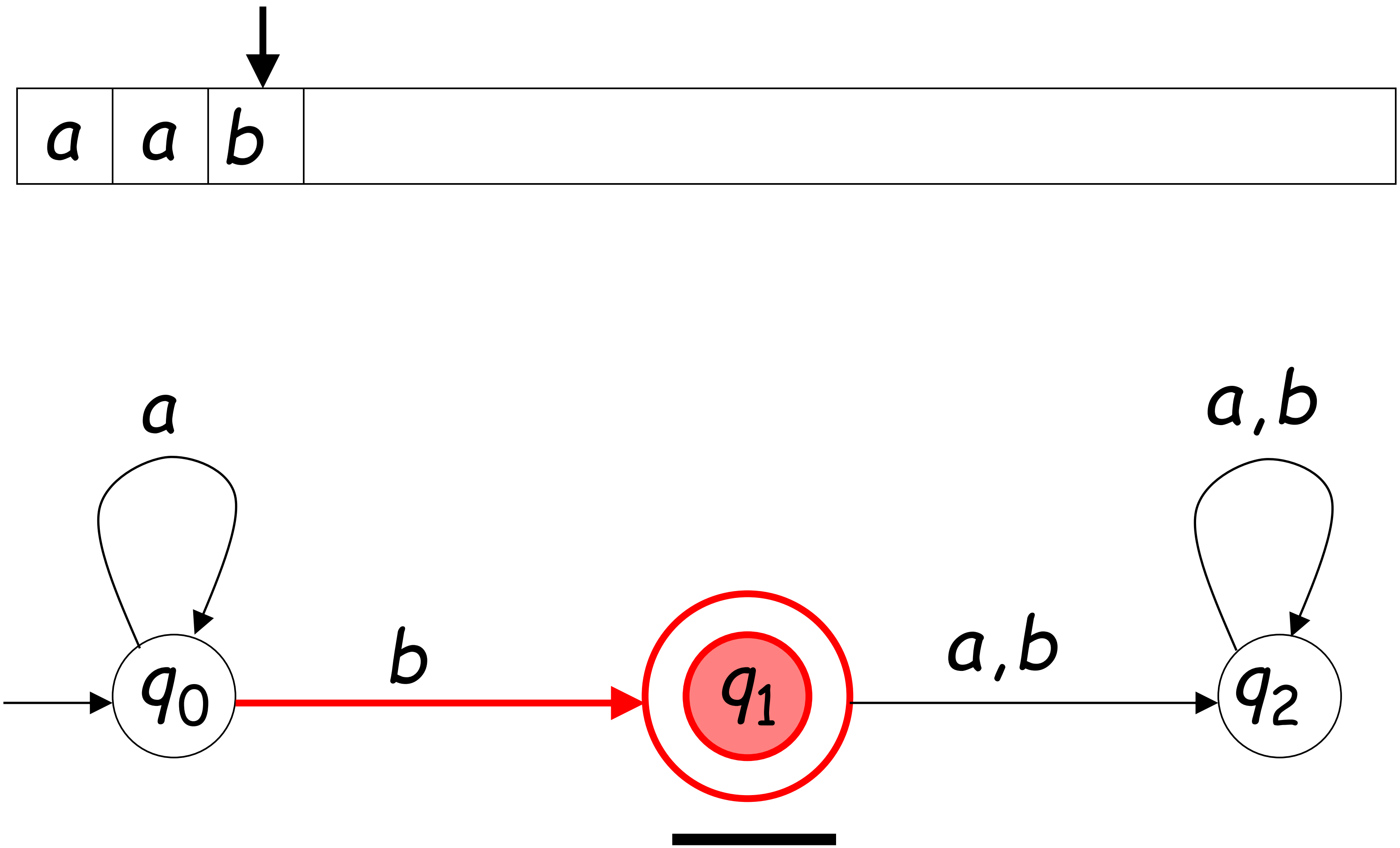


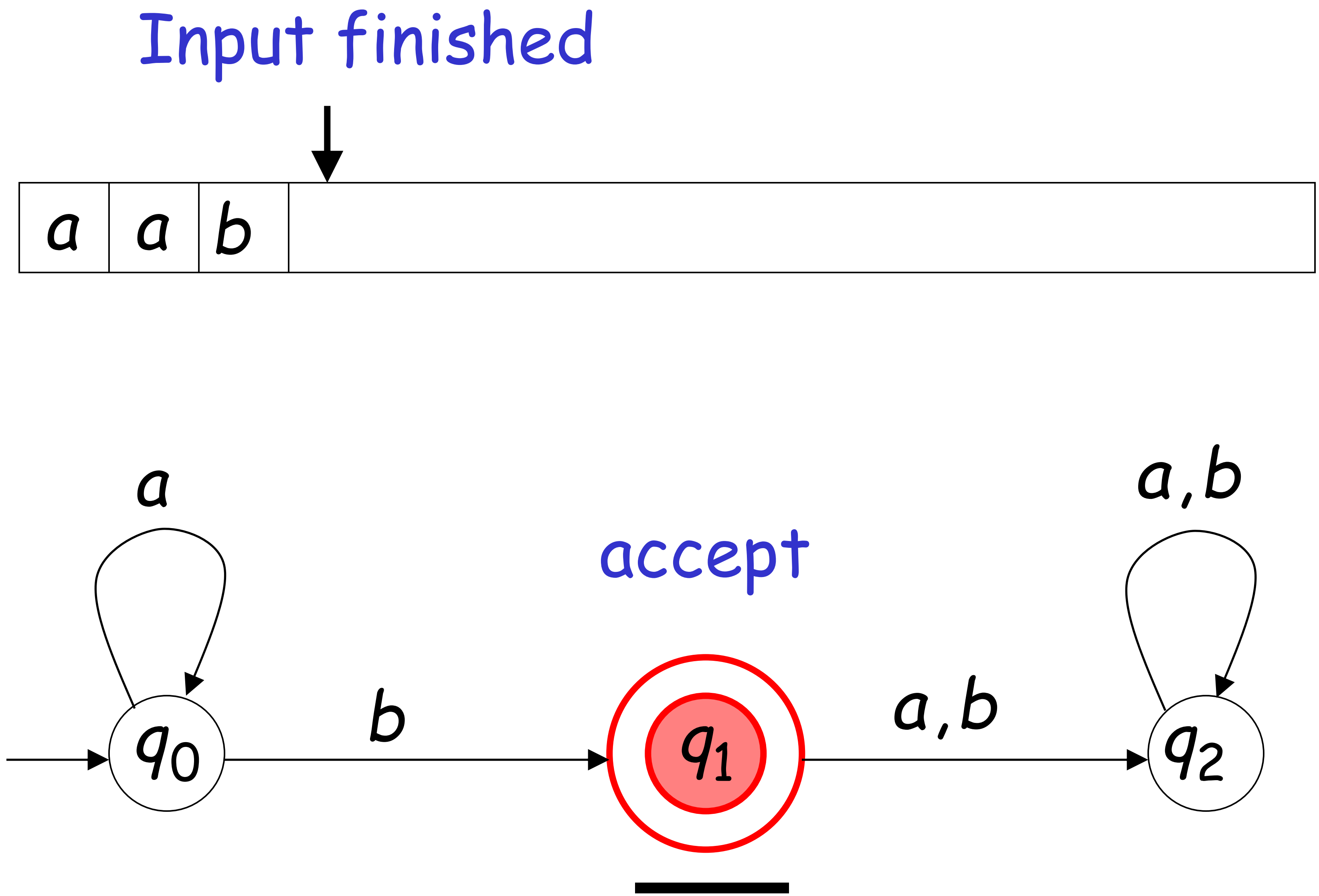
Another Example



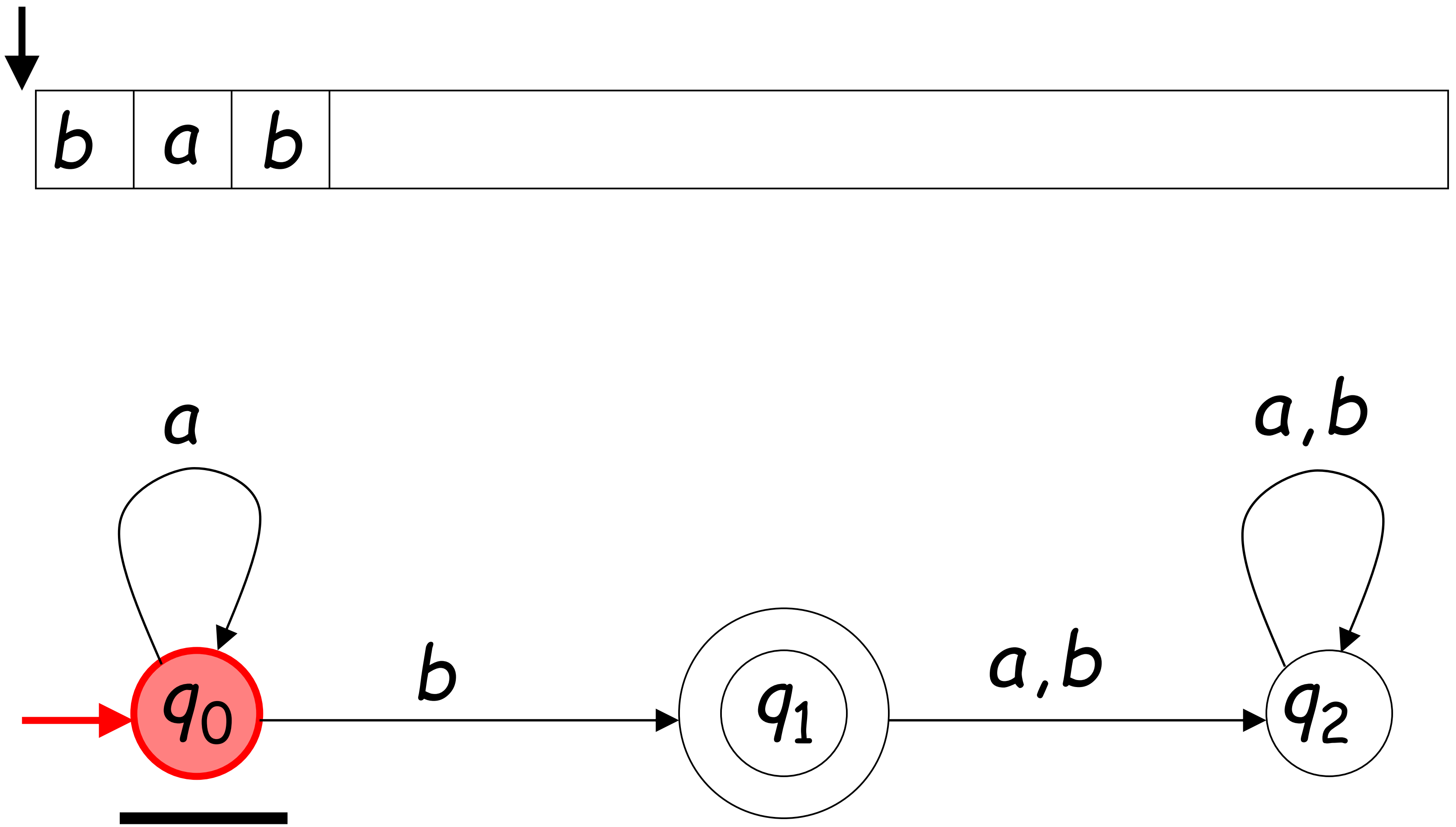

a
a
b
a
a,b
b
a,b
q_0
q_1
q_2

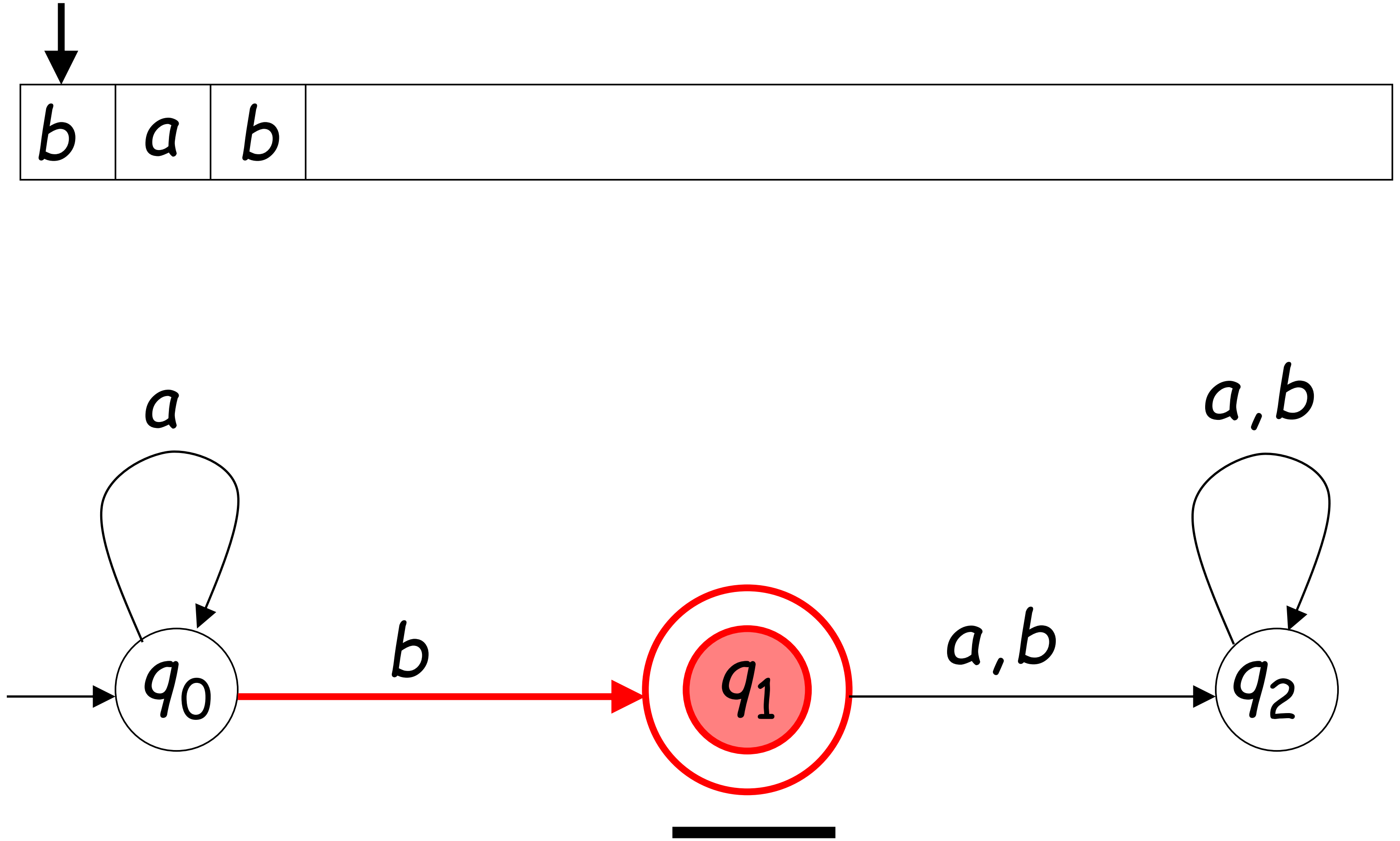

a
a
b
a
b
a,b
a,b
q0
q1
q2


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

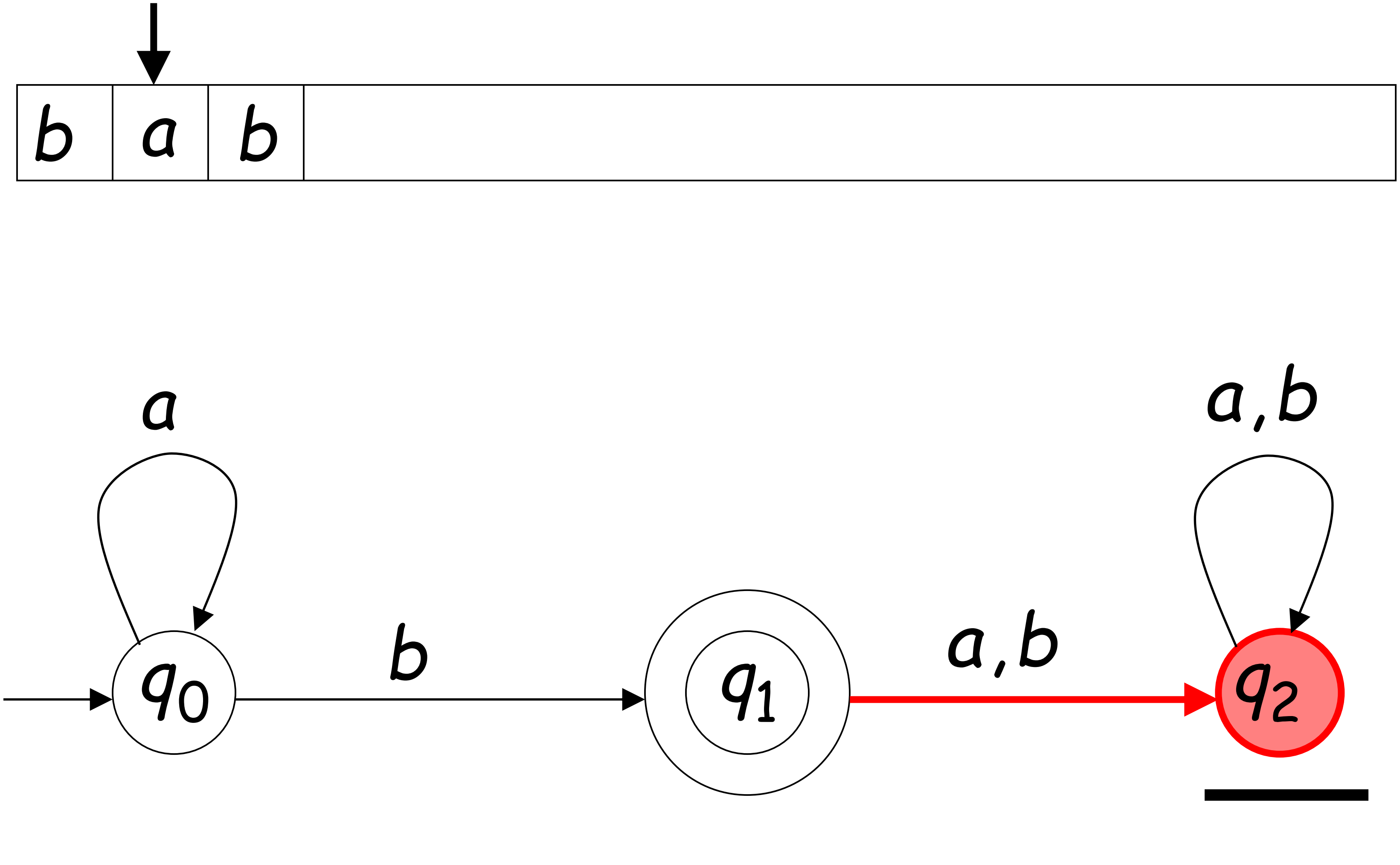

Input finished
a
a
b
a
accept
a,b
b
a,b
q_0
q_1
q_2

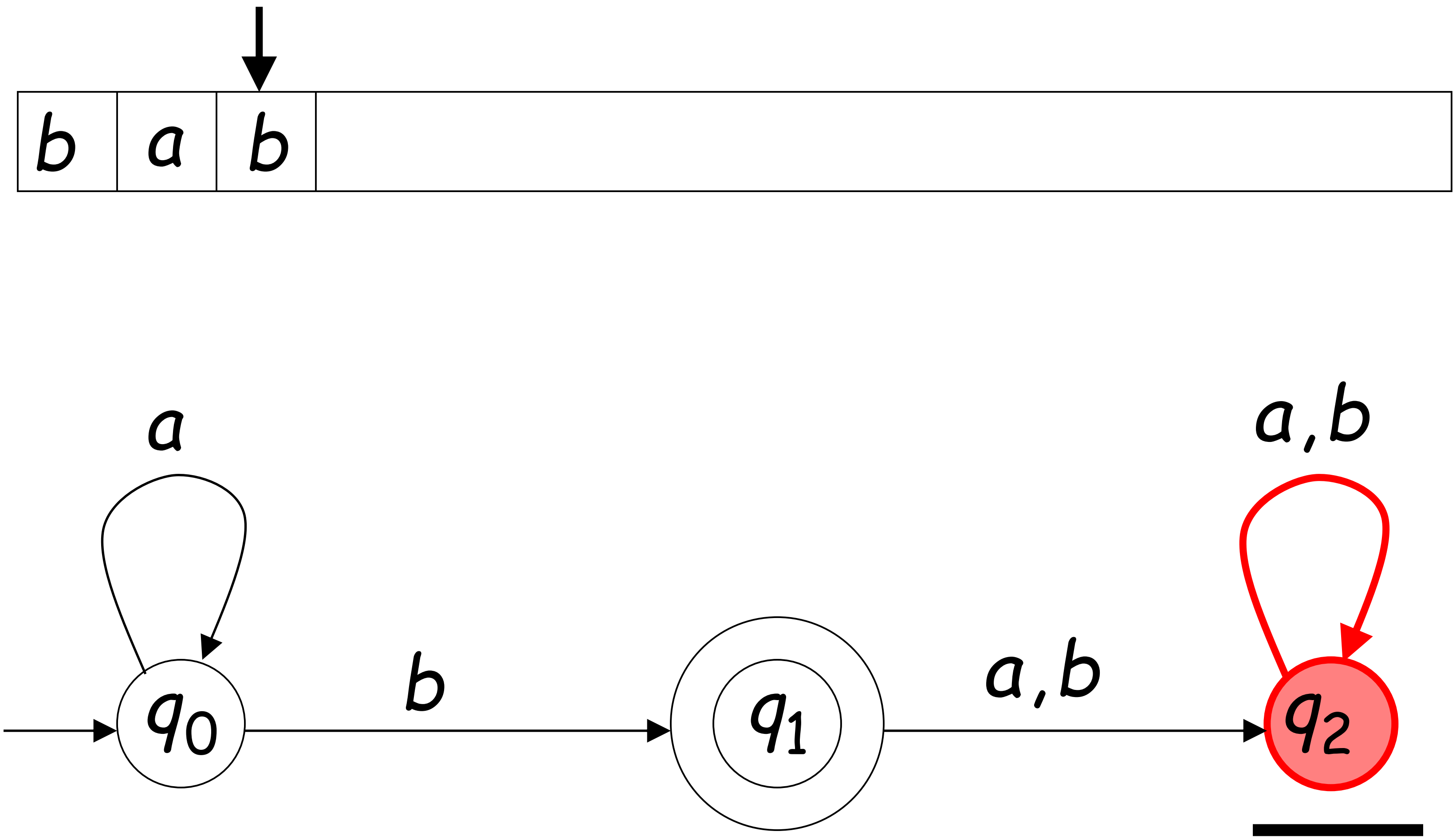
Rejection Example





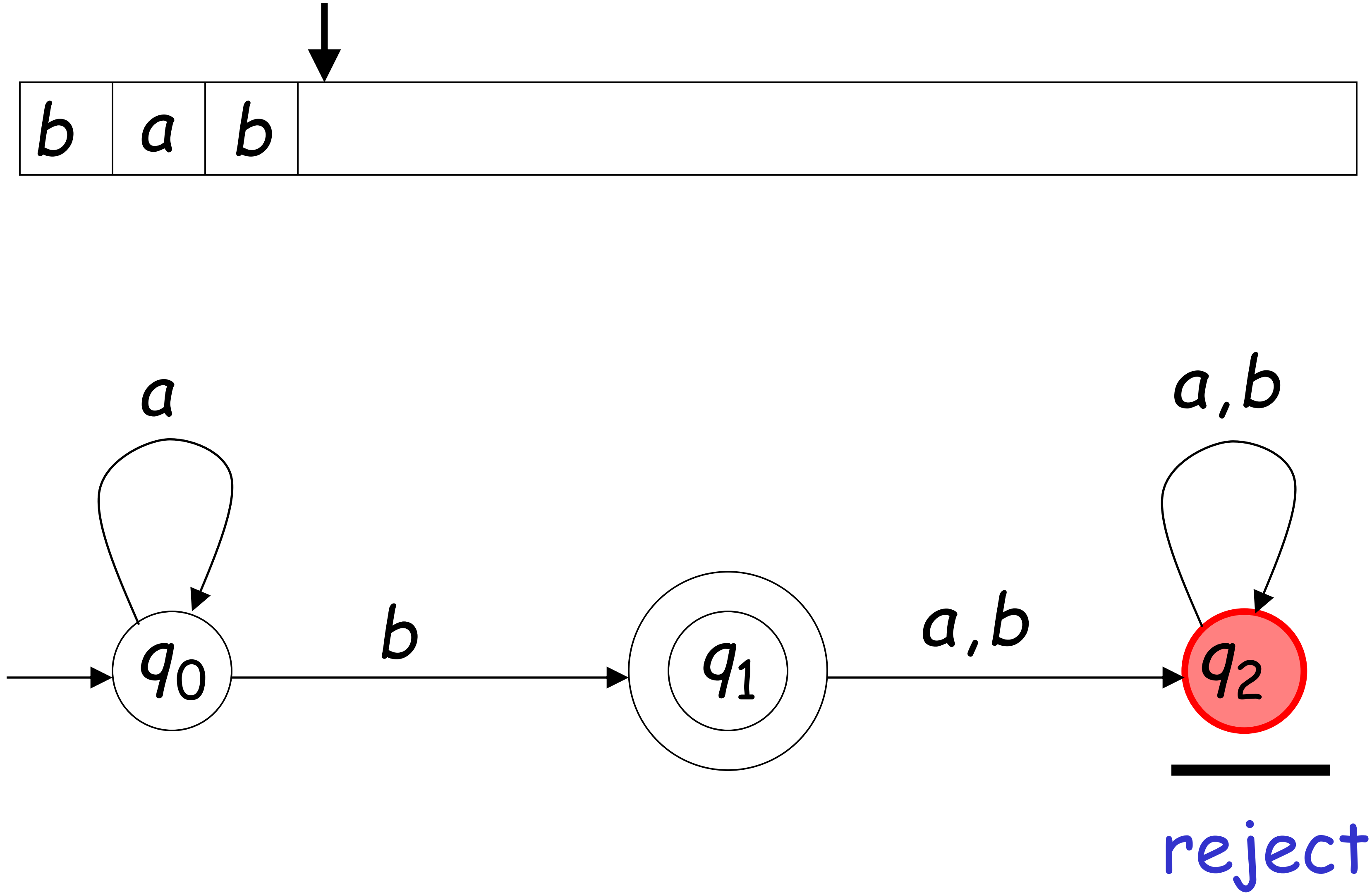
b
a
b
a
a,b
b
a,b
q_0
q_1
q_2


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



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

Input finished



Languages Accepted by FAs

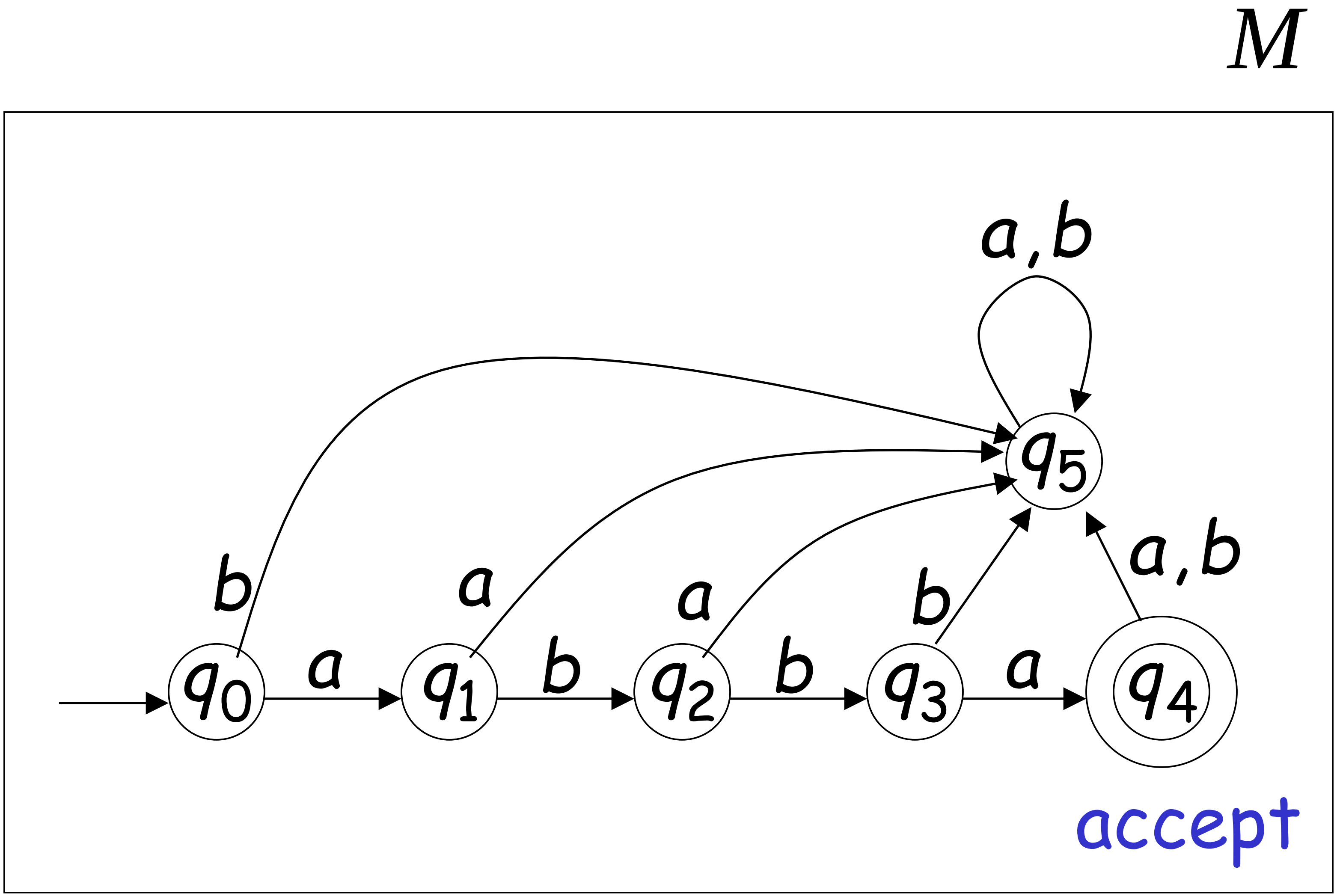
FA

Definition:

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

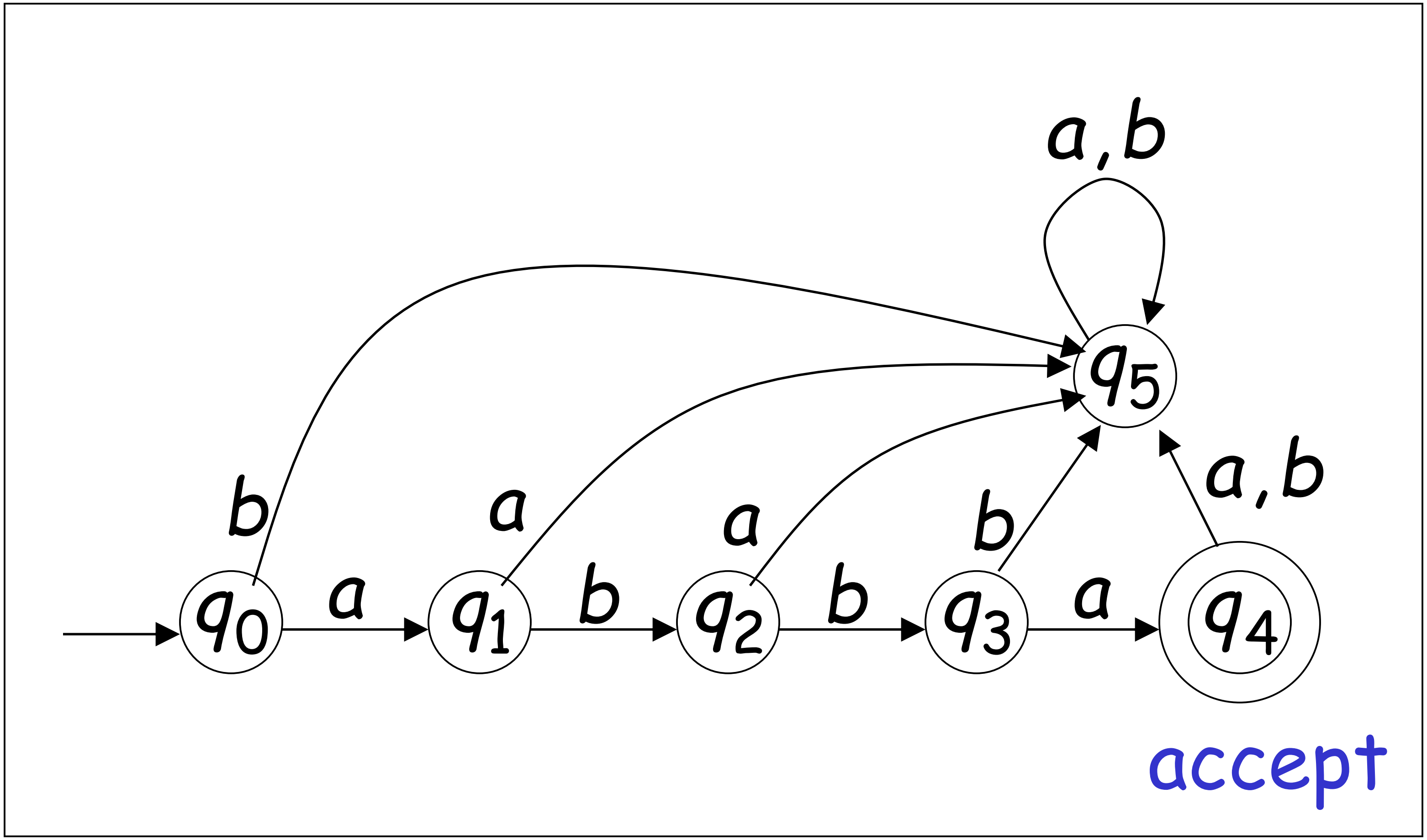
$L(M)$ = { strings that bring M to an accepting state}

Example: L(M) = ?

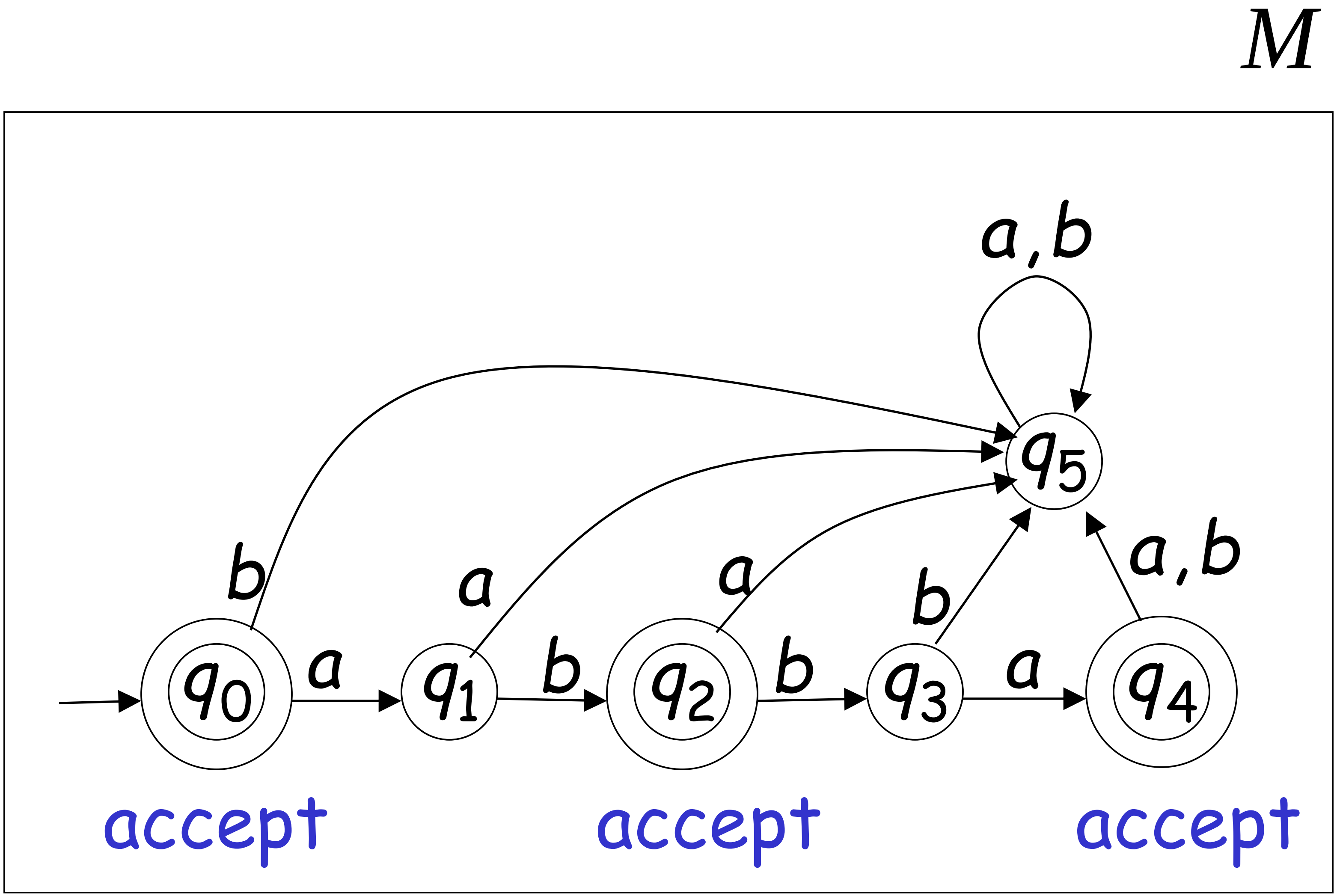


Example

M



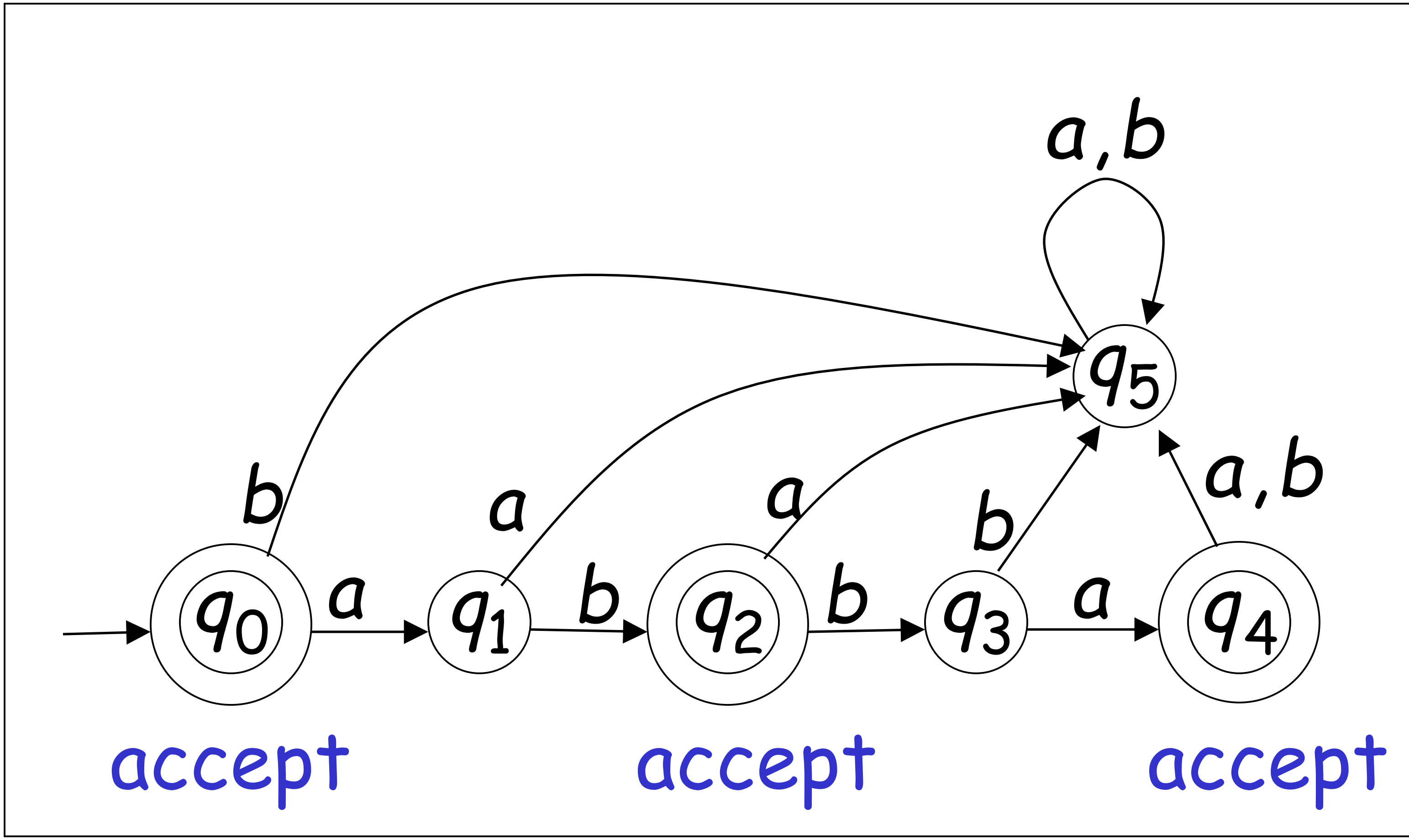
Example: L(M) = ?



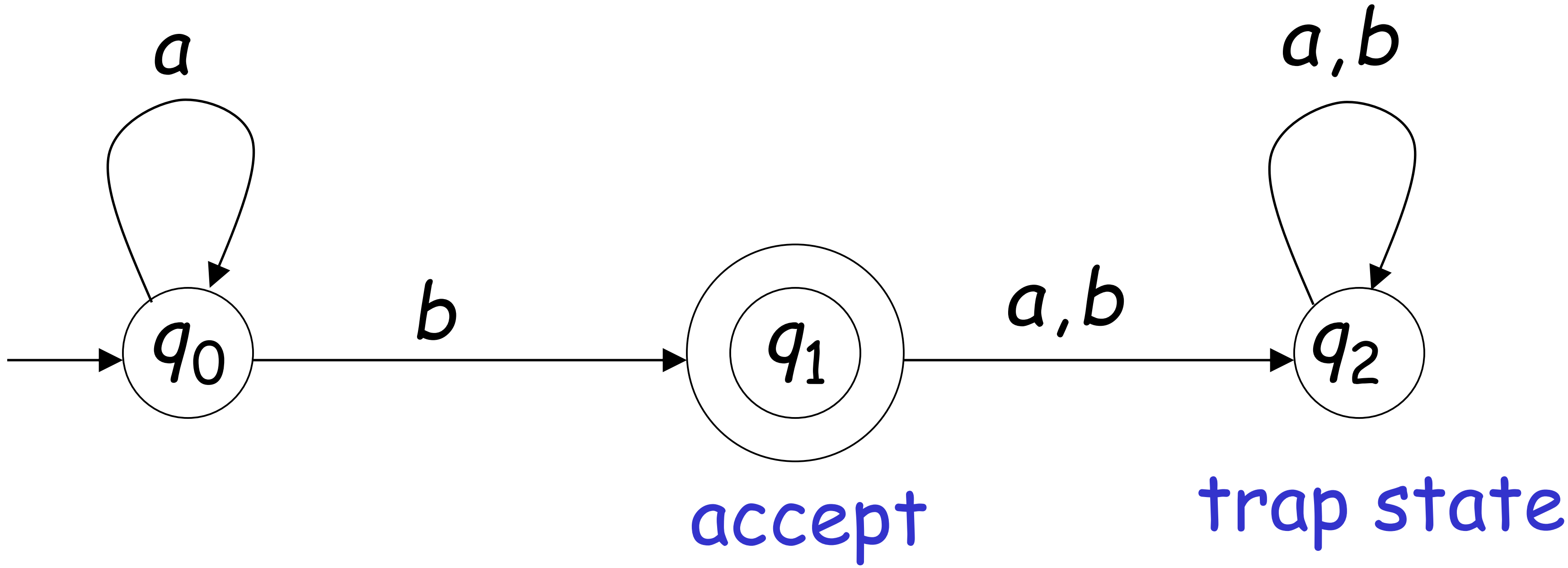
Example

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

M

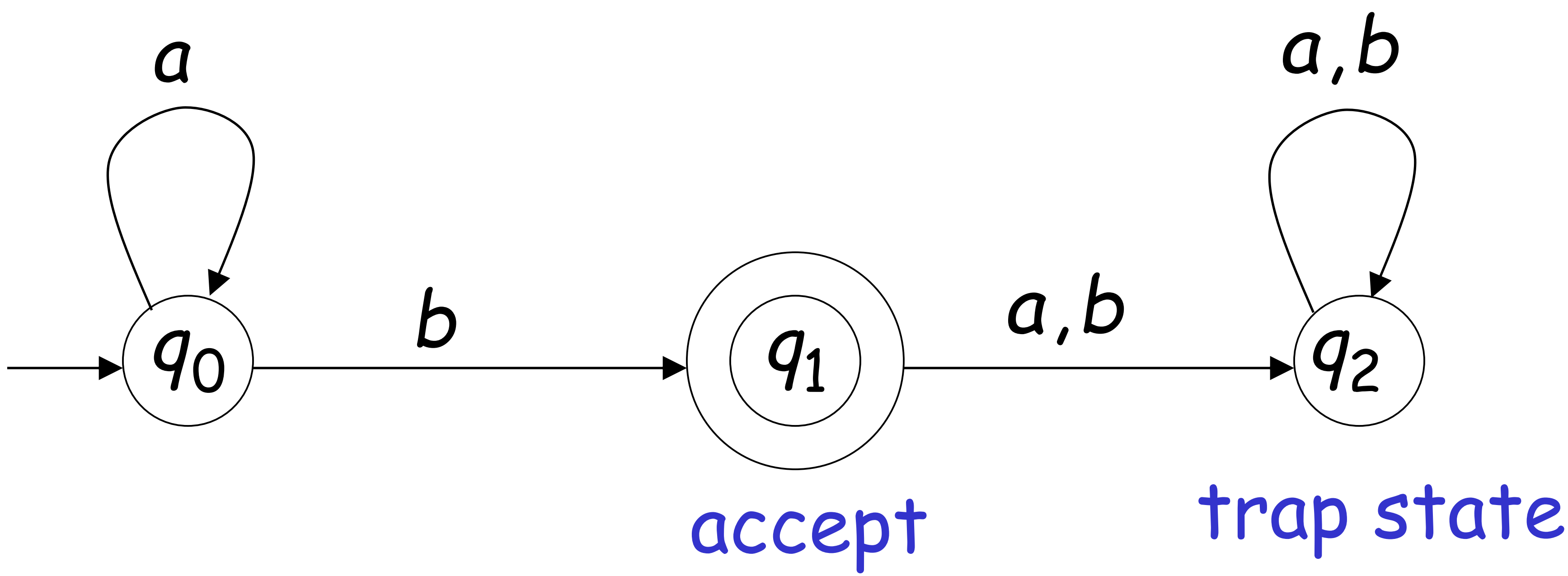


Example: L(M) = ?



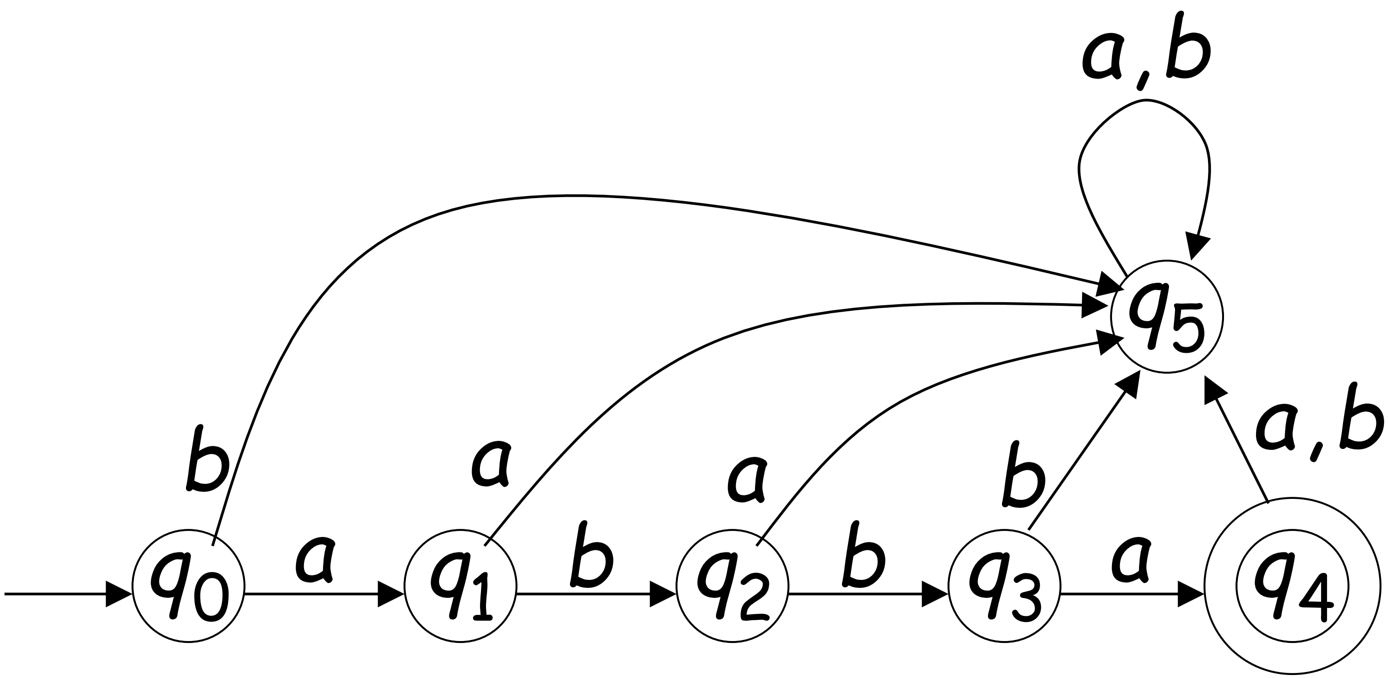
Example

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



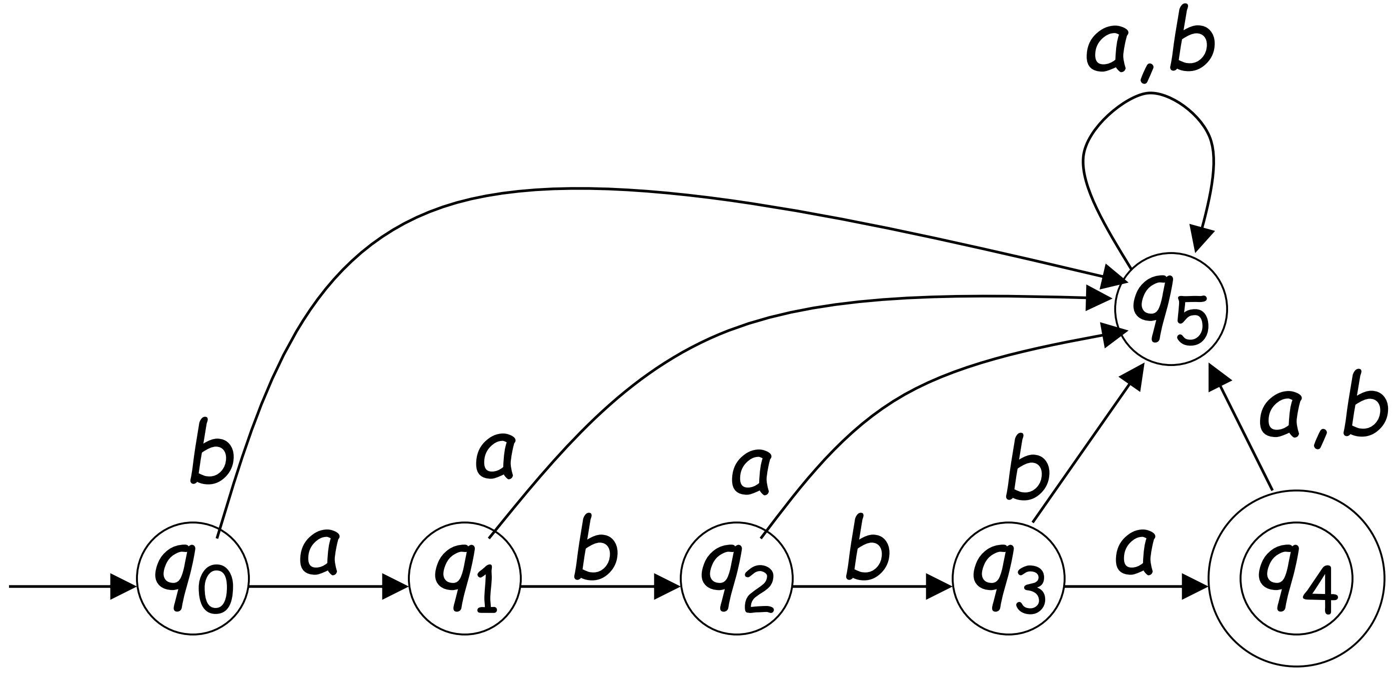
Input Alphabet Σ

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

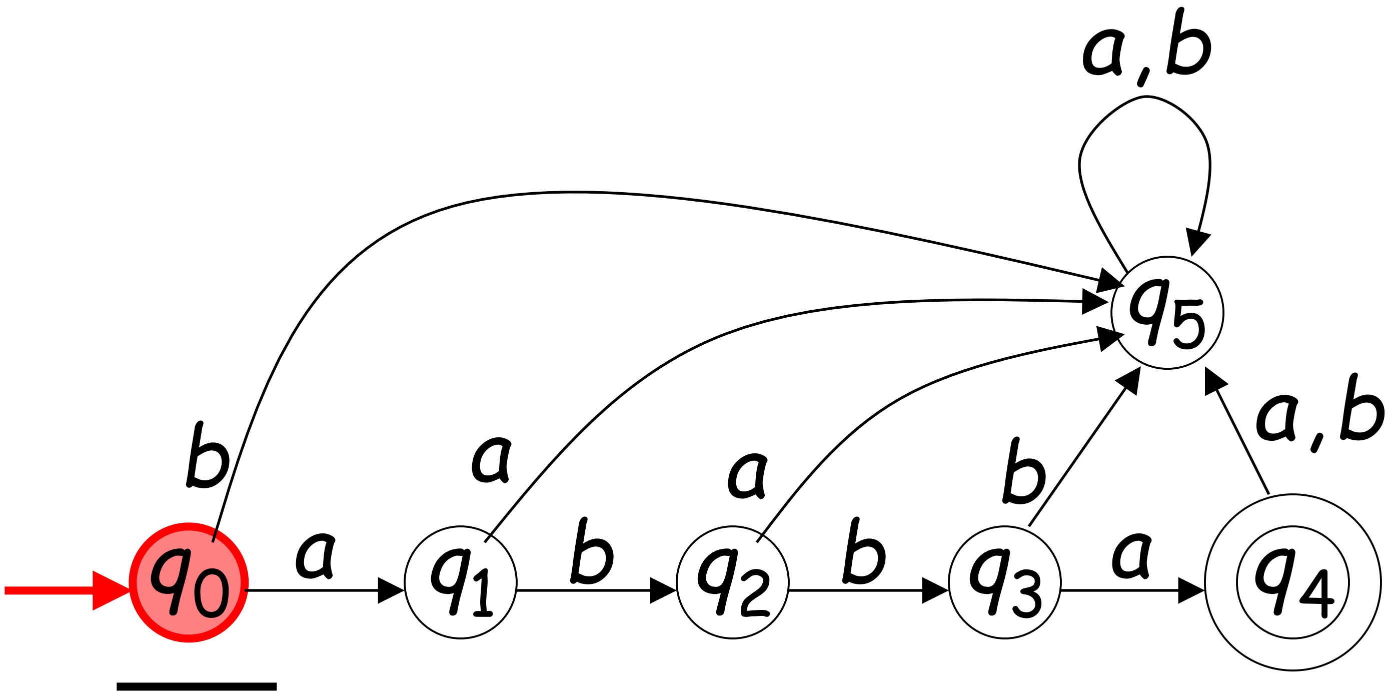


Set of States Q

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

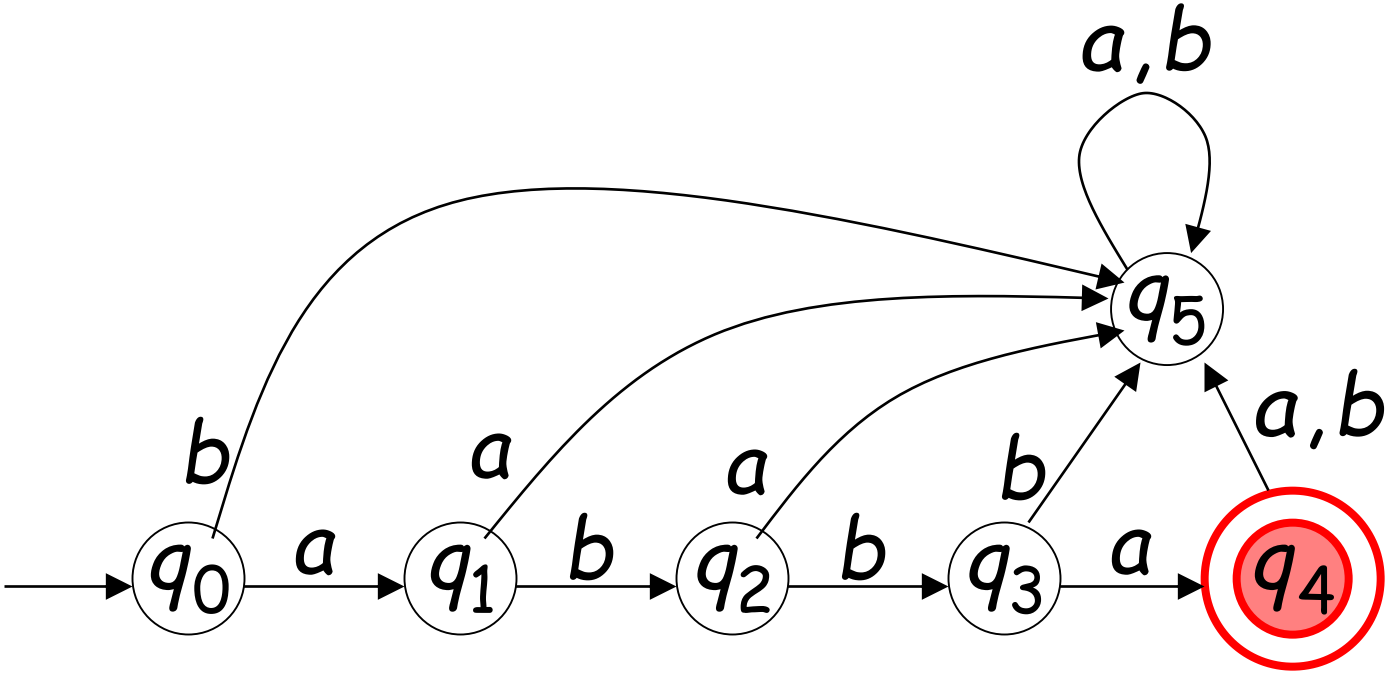


Initial State q_0



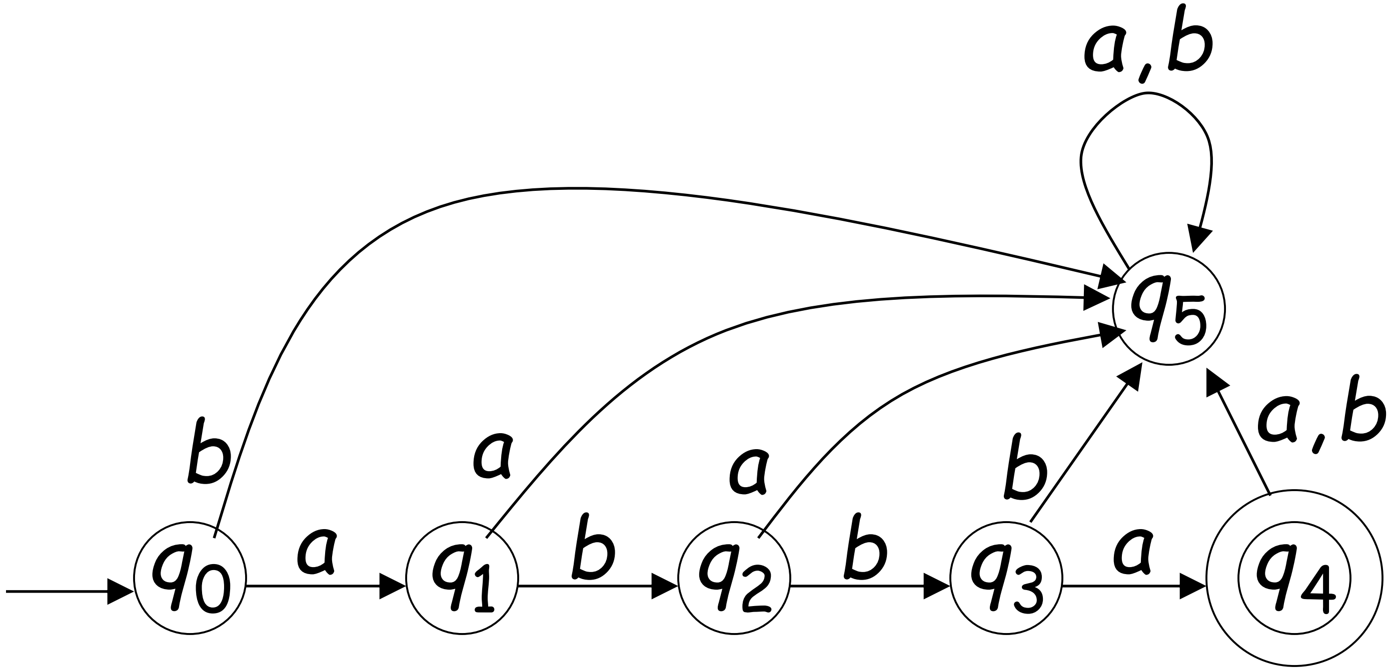
Set of Accepting States F

$$F = \{q_4\}$$

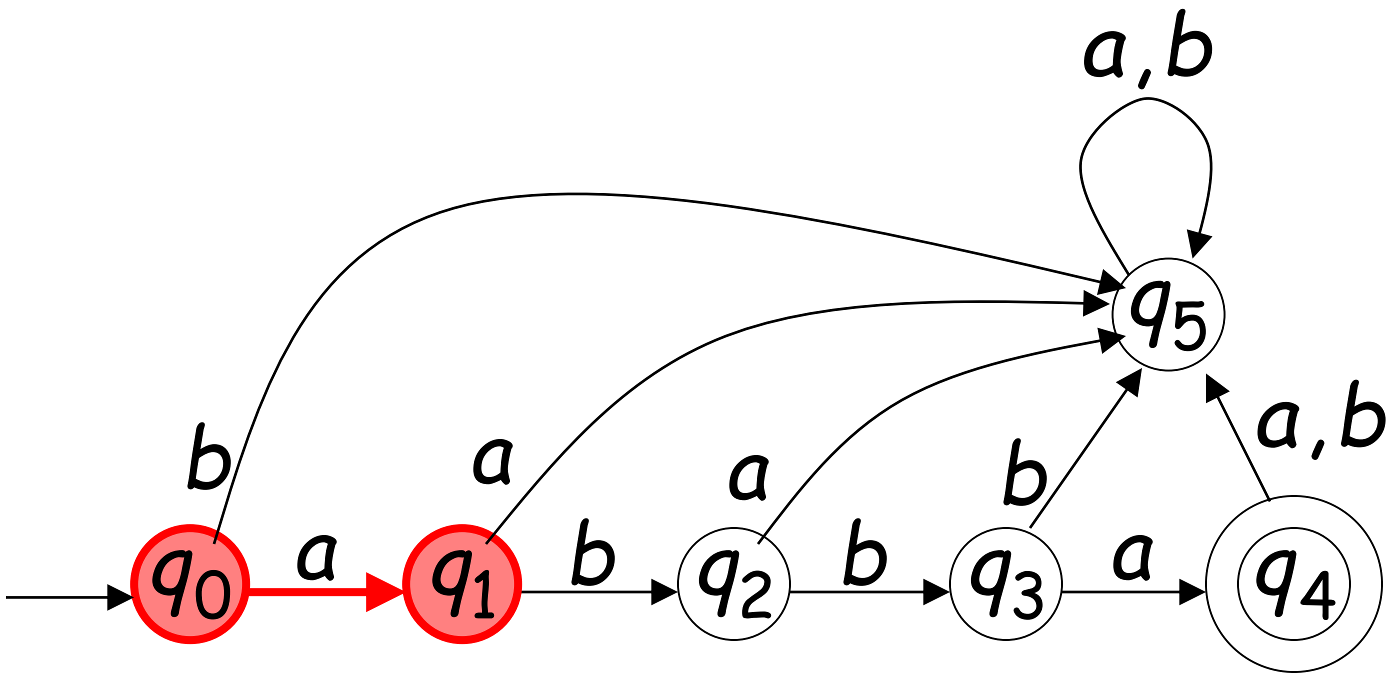


Transition Function δ

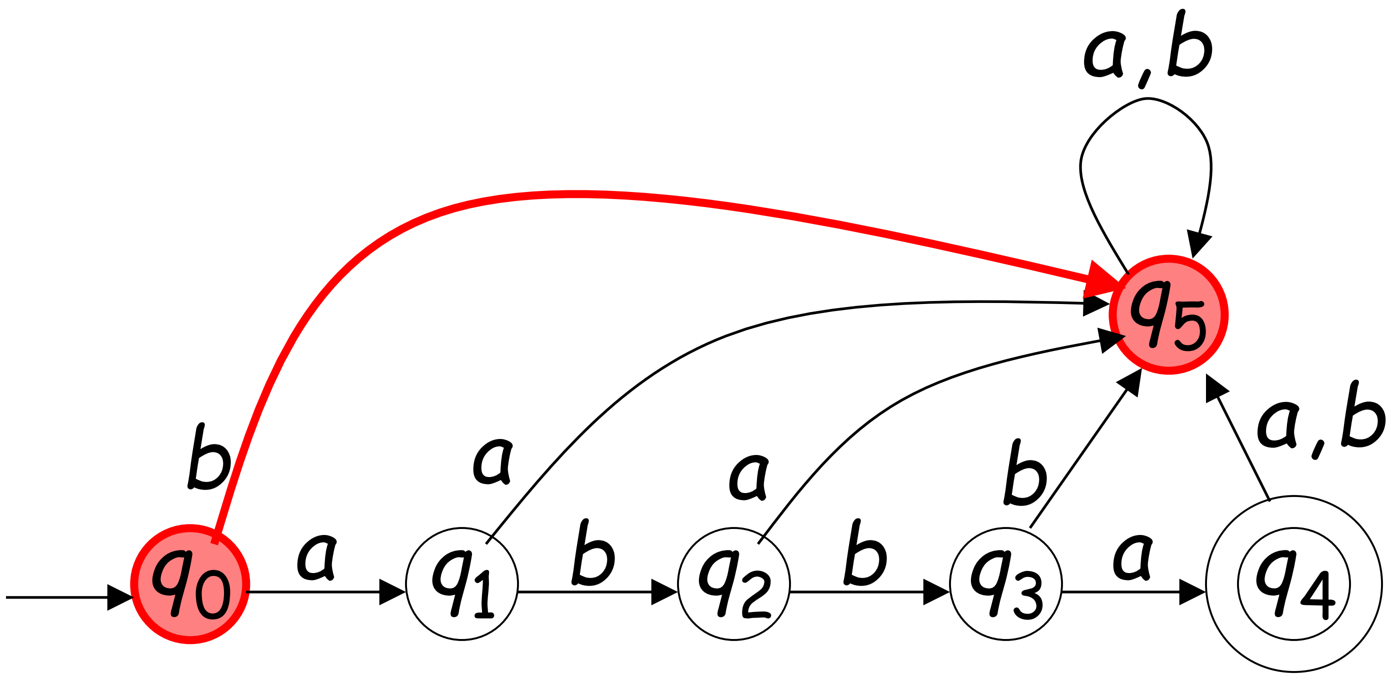
$$\delta : Q \times \Sigma \rightarrow Q$$



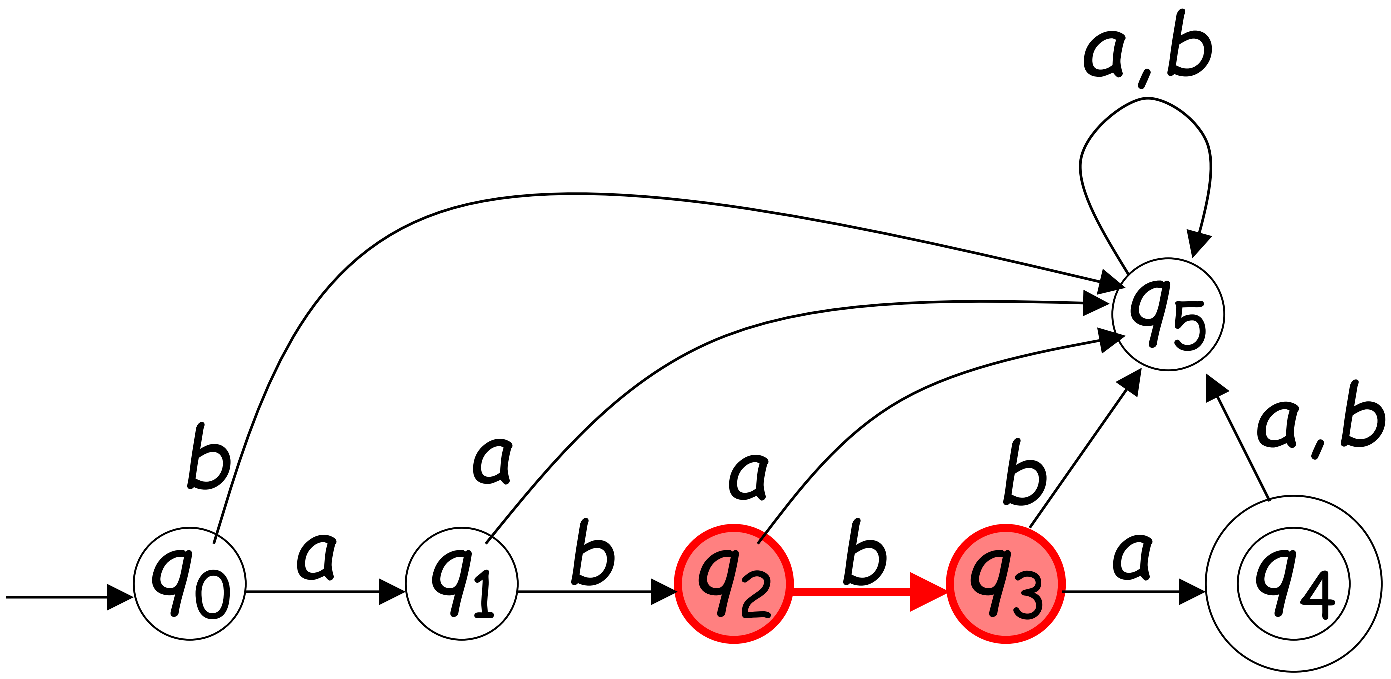
$$\delta(q_0, a) = q_1$$



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

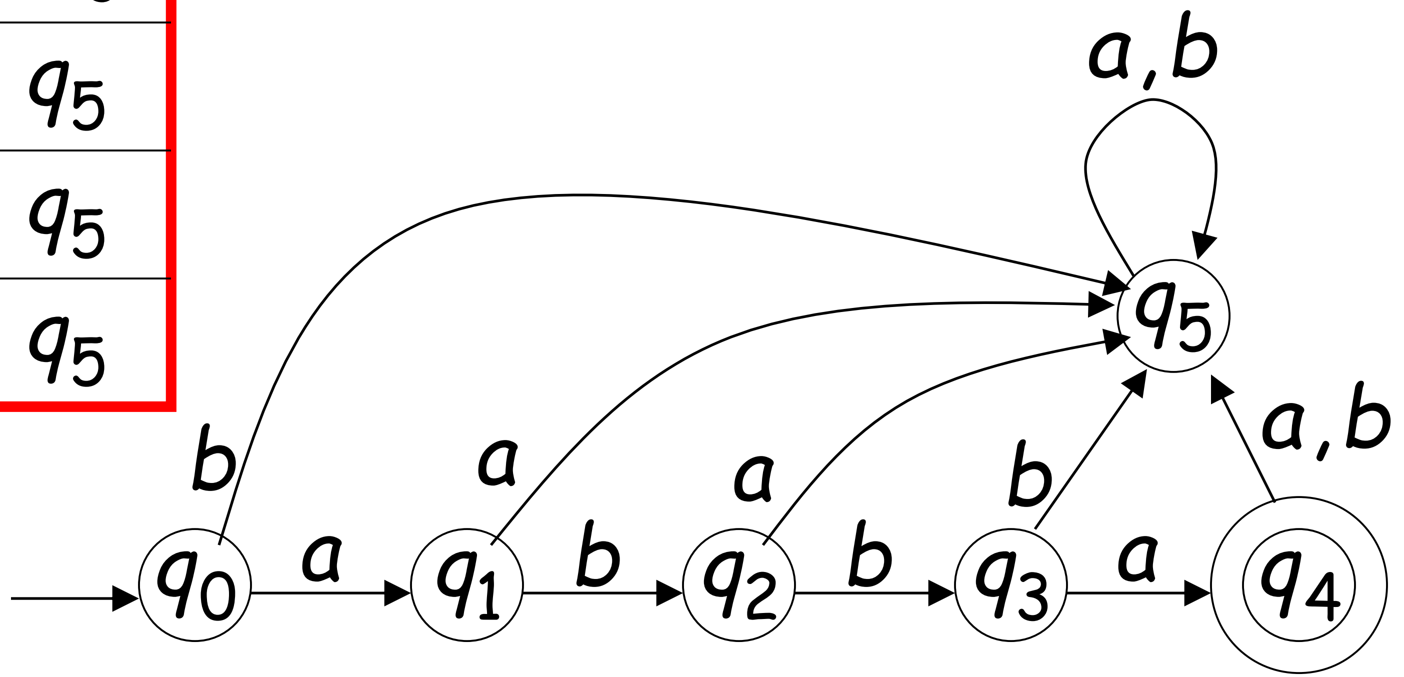


$$\delta(q_2, b) = q_3$$



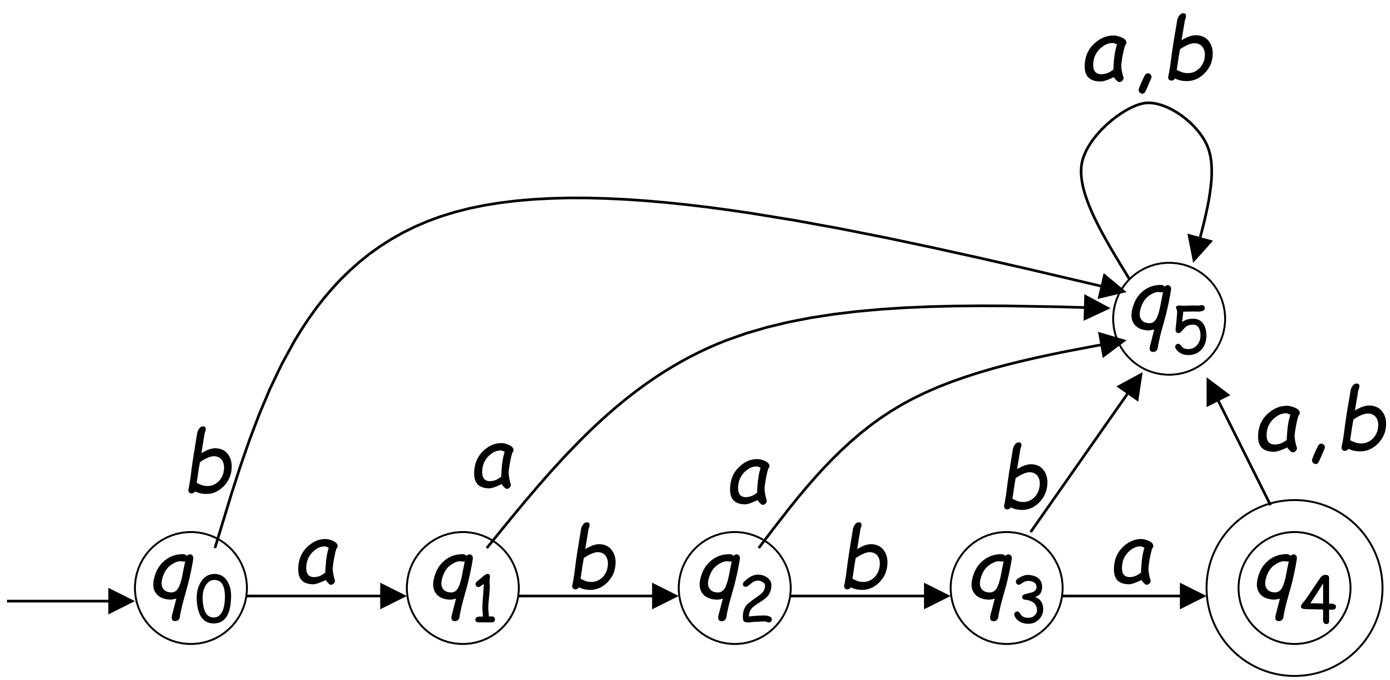
Transition Function δ

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

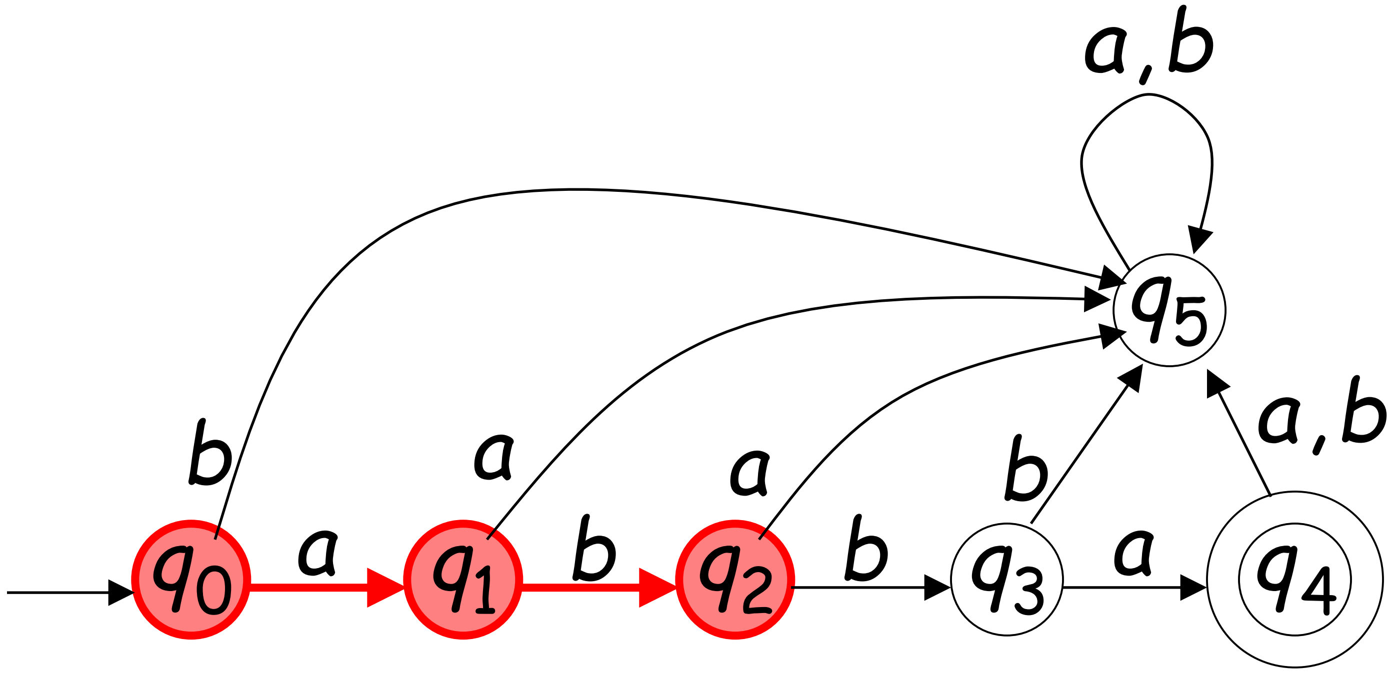


Extended Transition Function $\delta *$

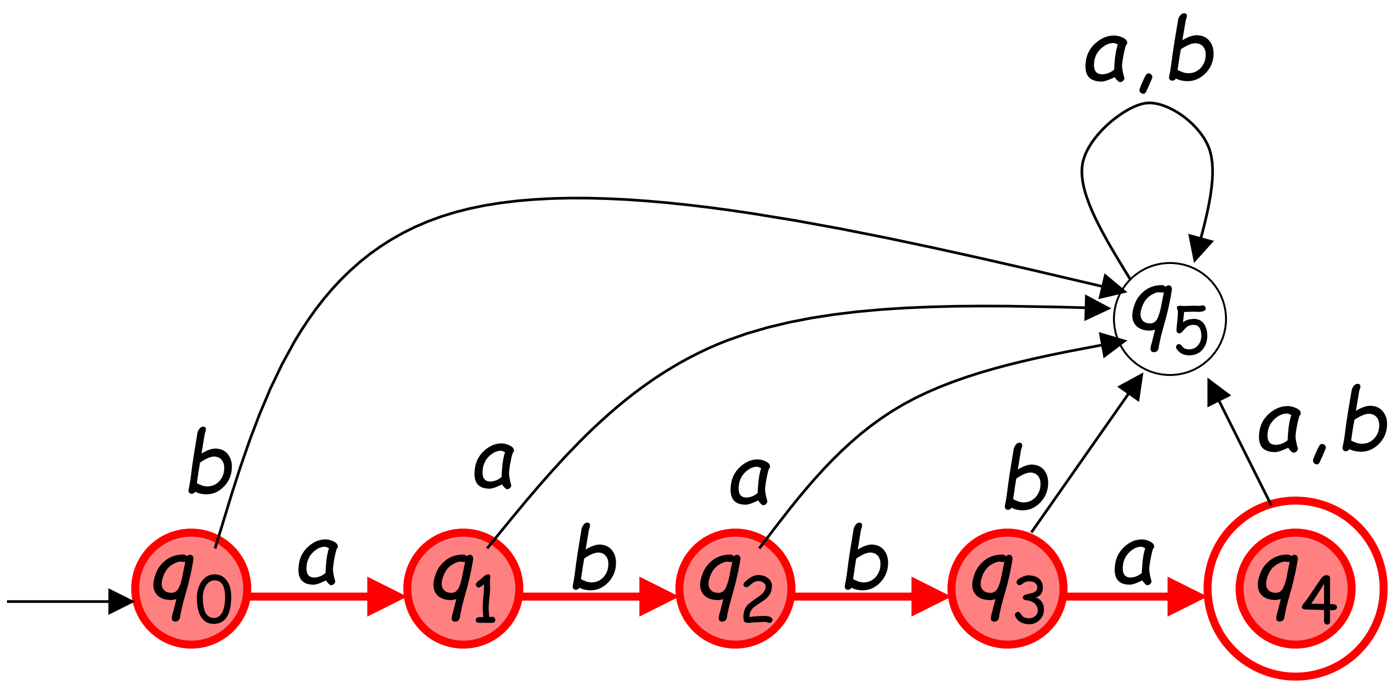
$$\delta^* : Q \times \Sigma^* \to Q$$



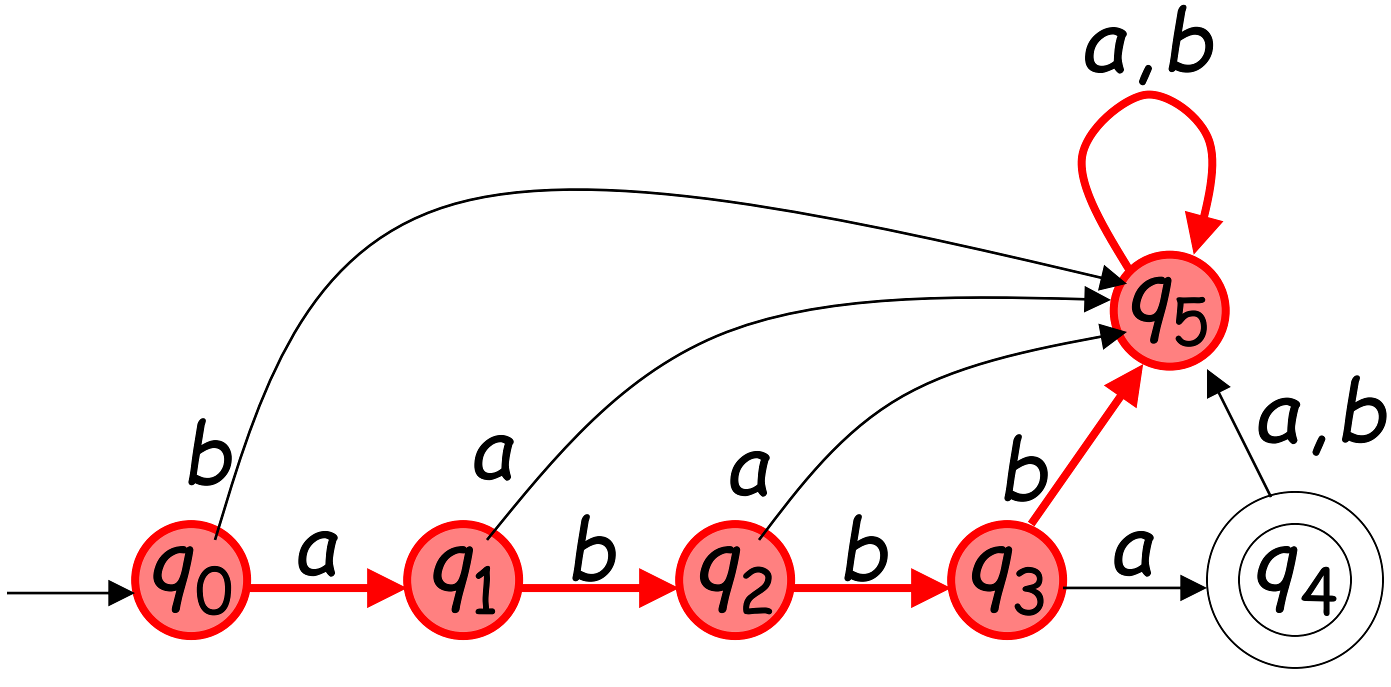
$$\delta^*(q_0, ab) = q_2$$



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

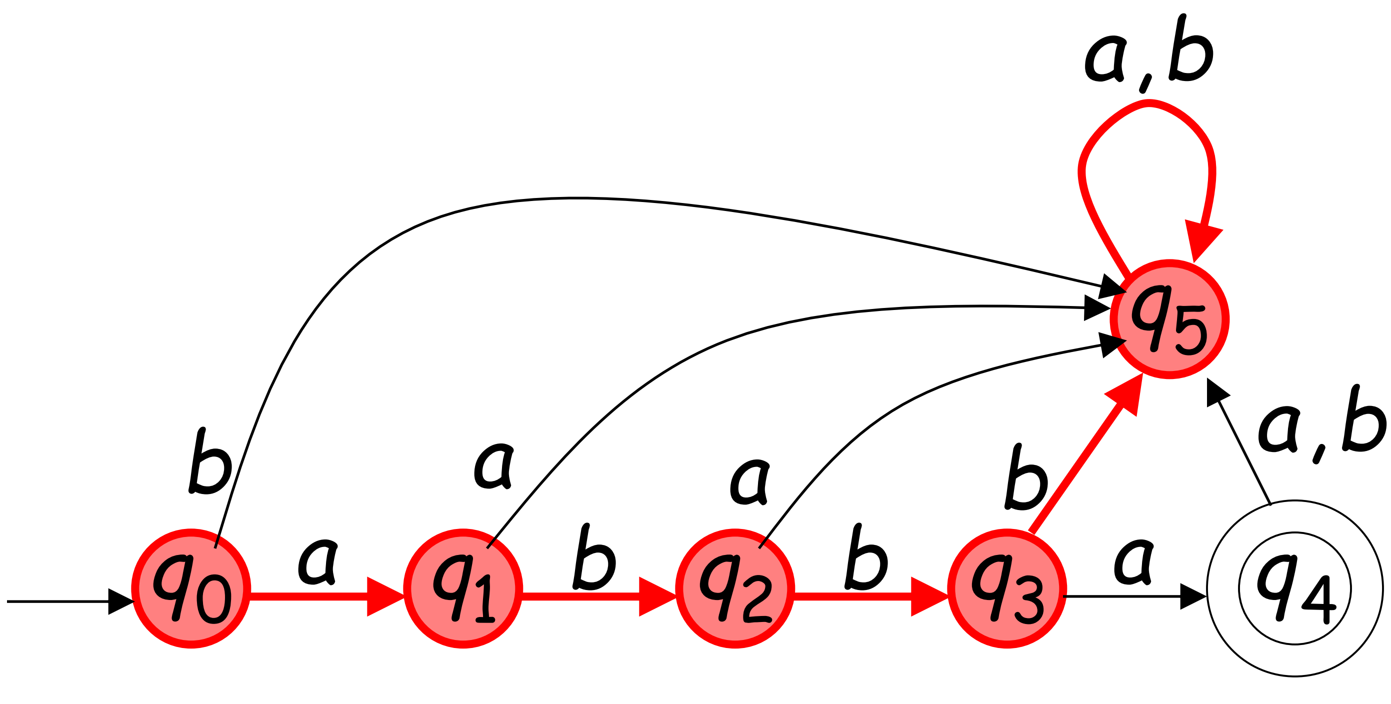


$$\delta^*(q_0, abbbaa) = q_5$$



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

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



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

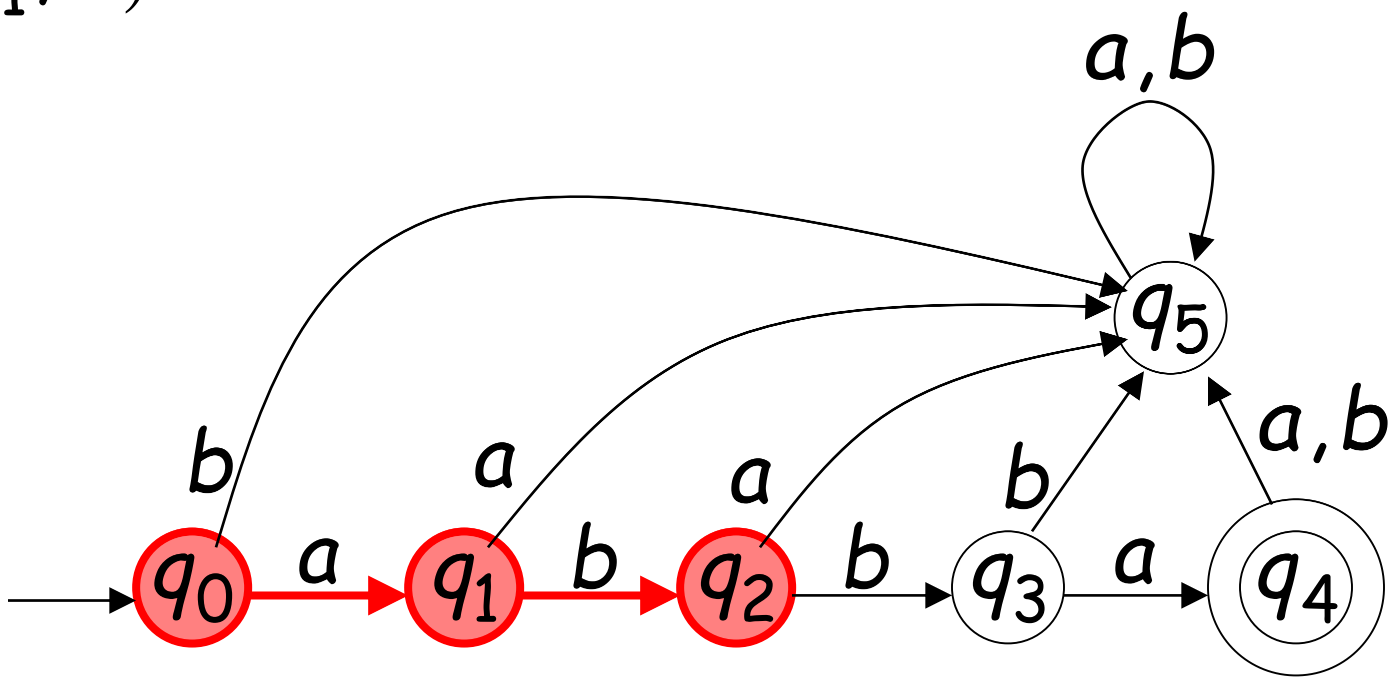
$$\delta(\delta^*(q_0, a), b) =$$

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

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

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

$$q_2$$

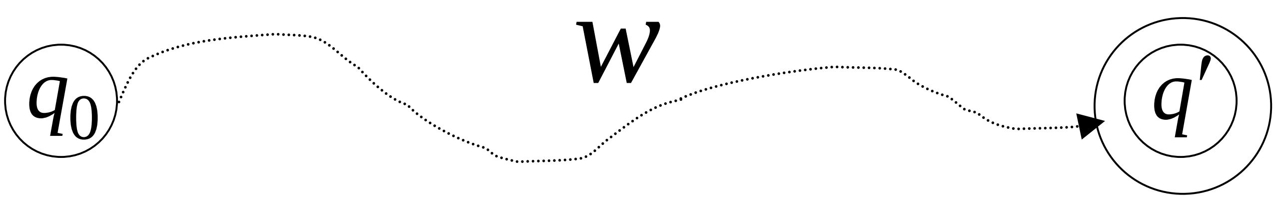


Language Accepted by FAs

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

Language accepted by M :

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



$q' \in F$

Observation

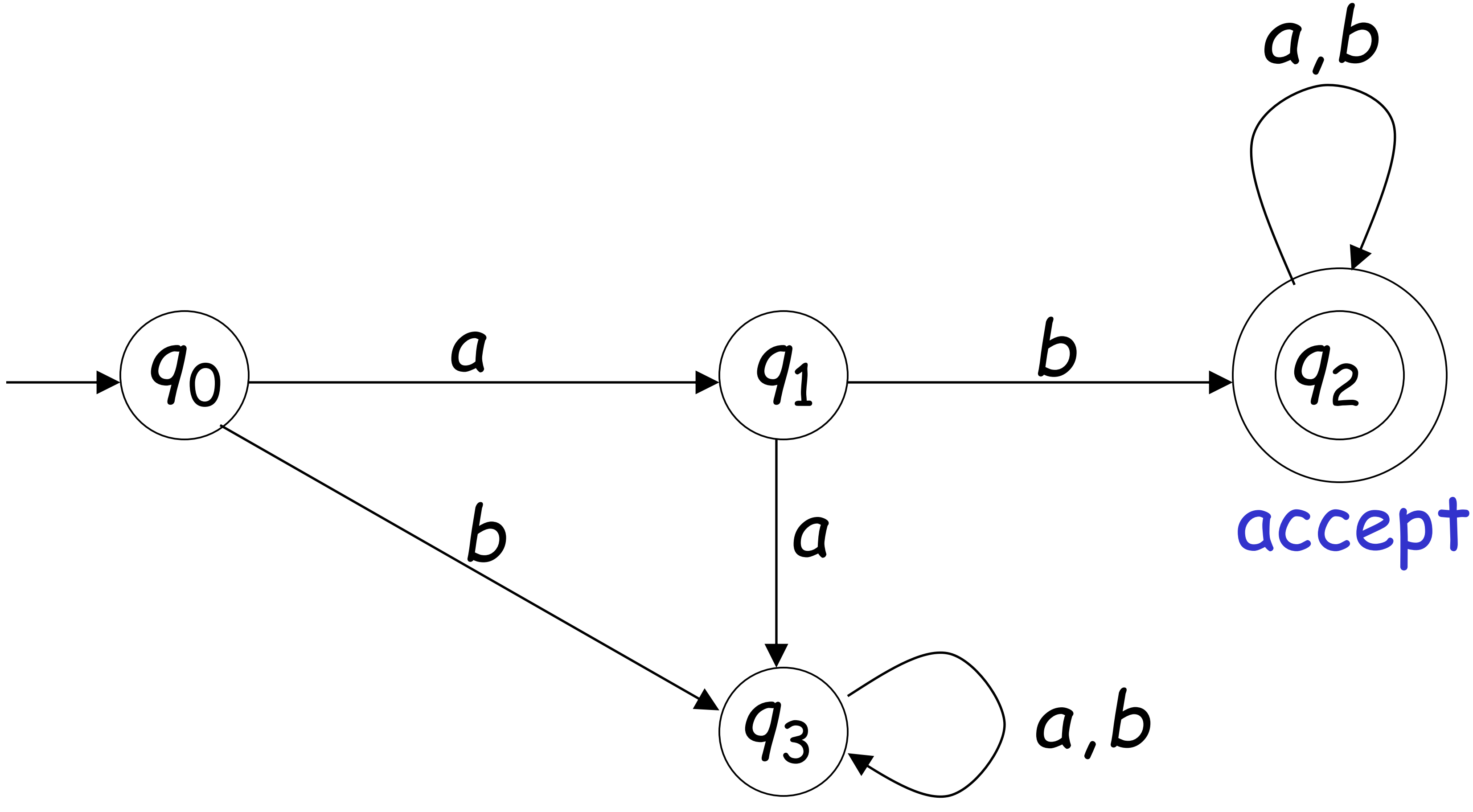
Language rejected by M :

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

$$q_0 \xrightarrow{w} q' \qquad q' \notin F$$

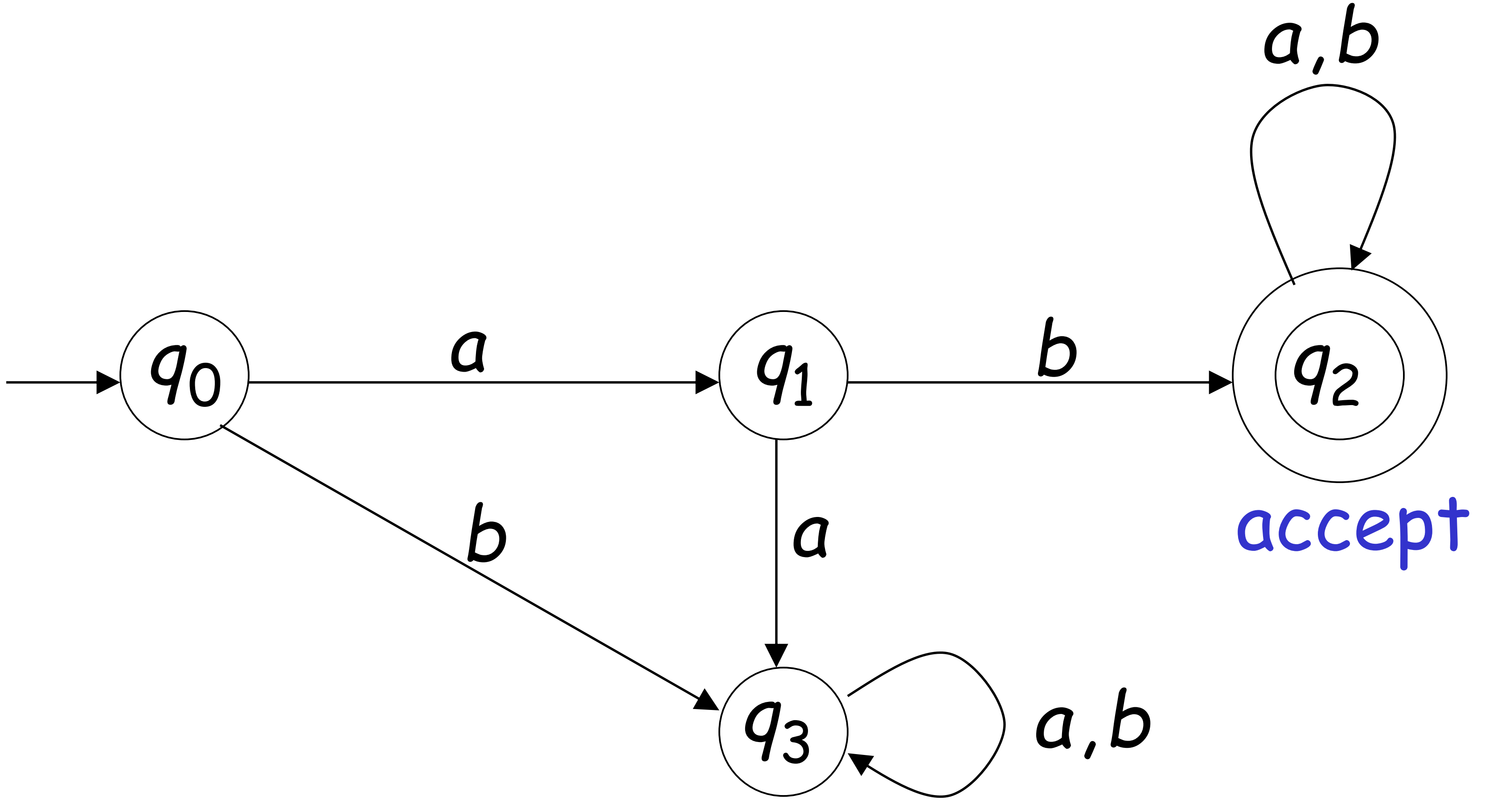
L(M) ?

ab



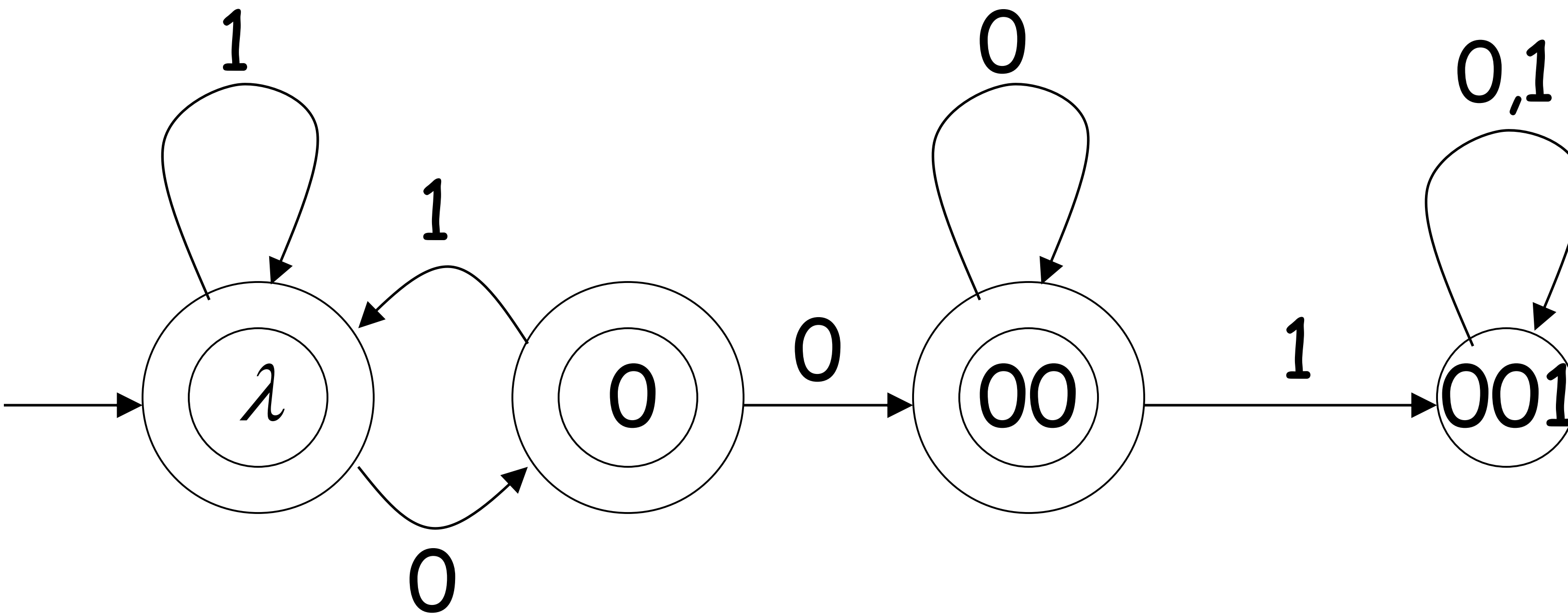
Example

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



Try-Starting with a and ending with b

L(M)?

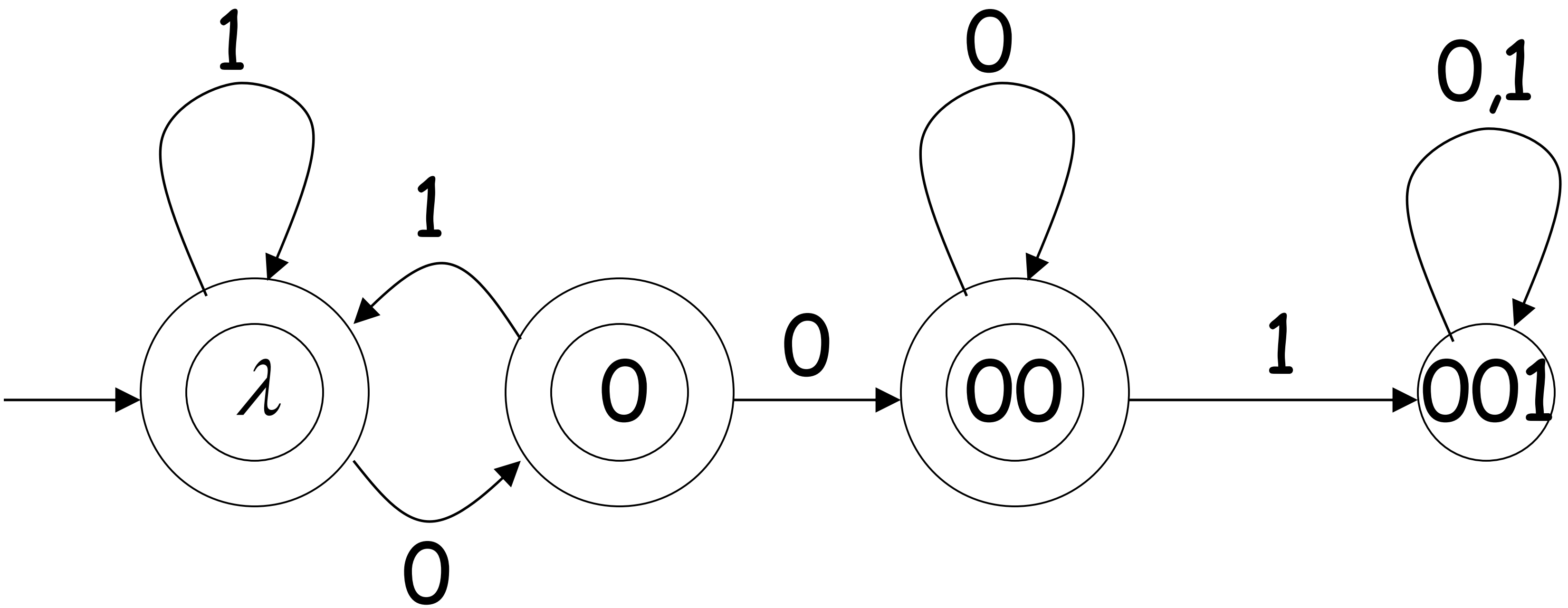


Example

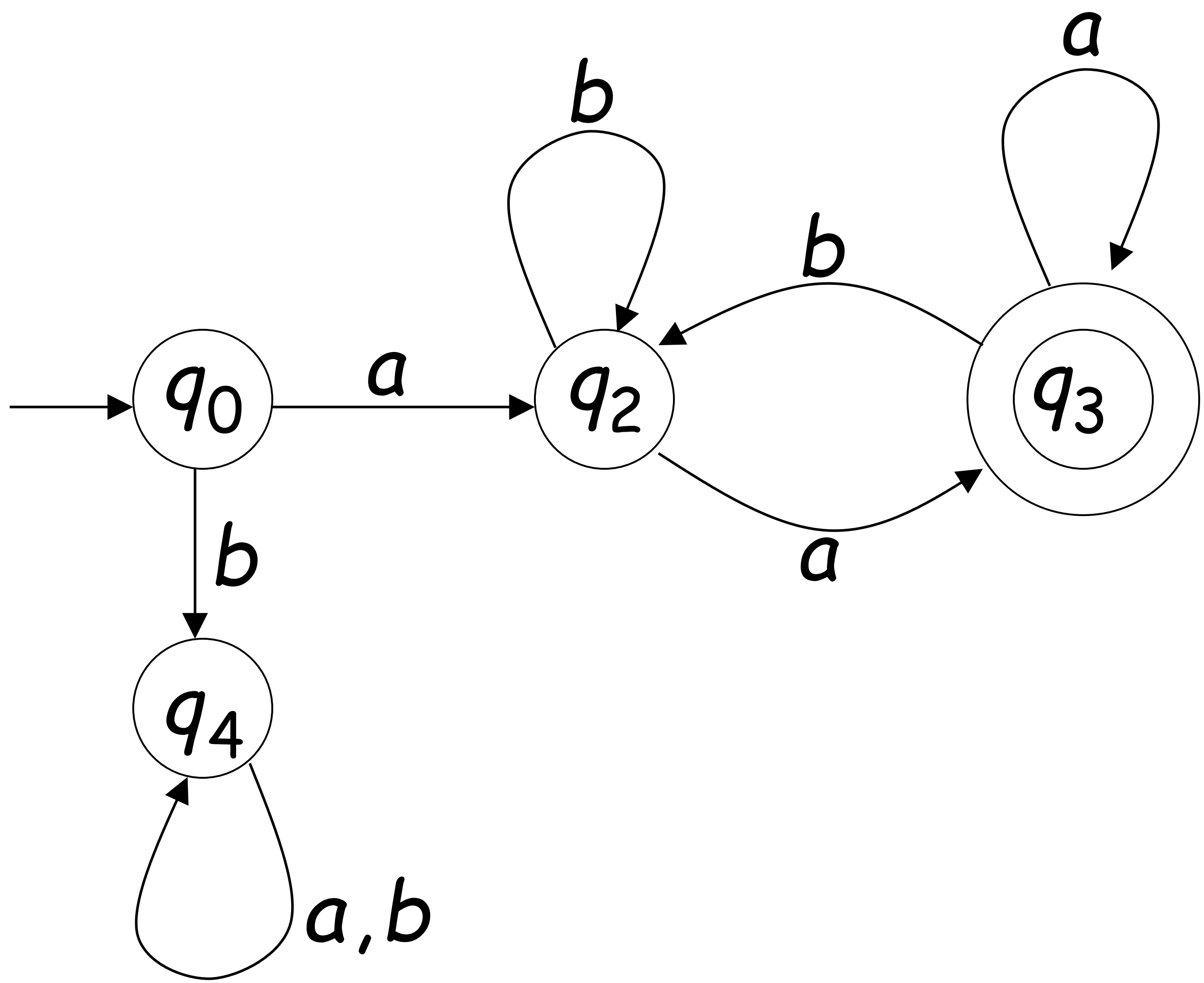
$L(M) = \{$ all strings without substring 001 $\}$

Example

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

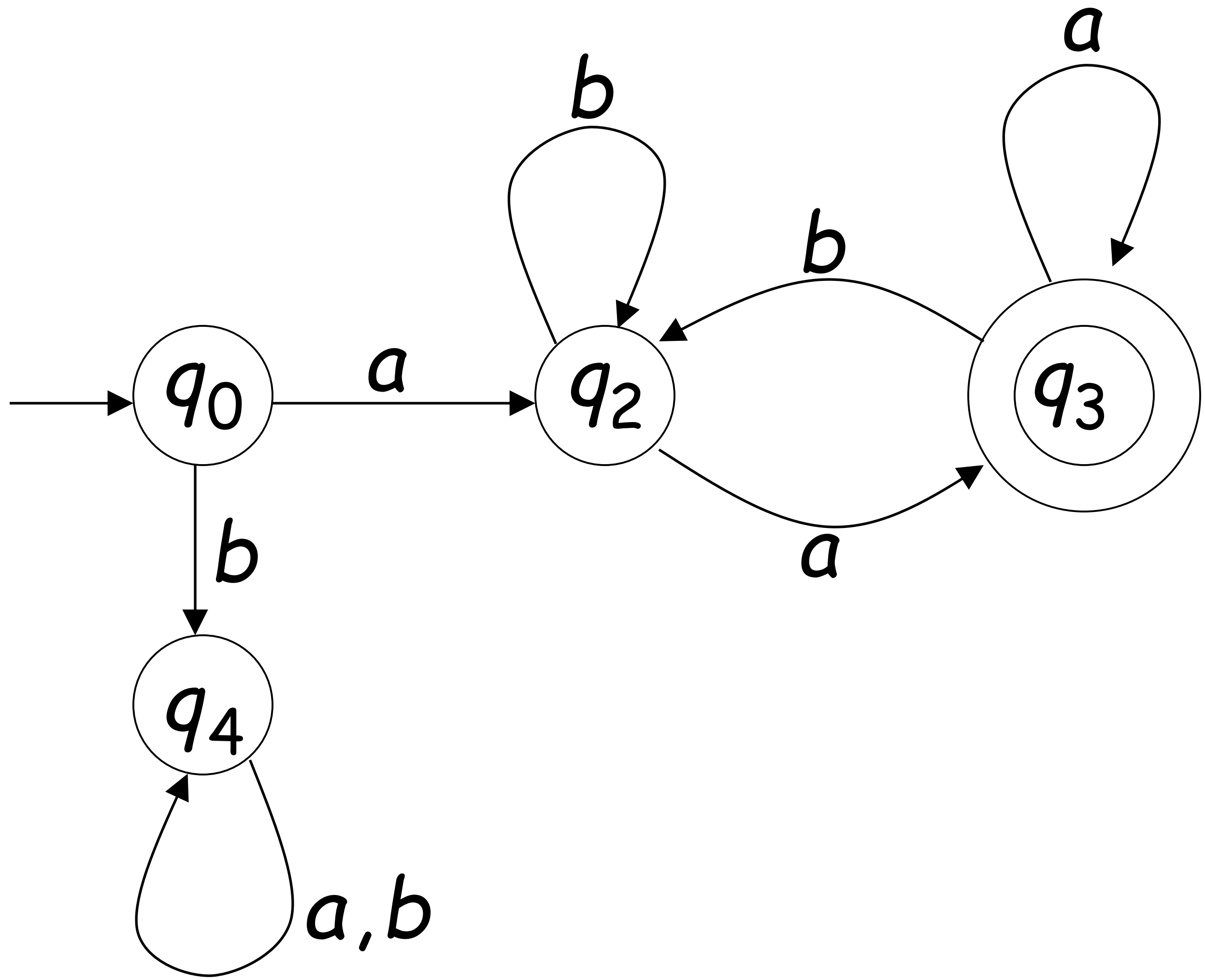


L(M) ?



Example

$$L(M)=\{awa : w \in \{a,b\}^*\}$$



Regular Languages

Definition:

- A language L is called **regular** if and only if there exists some deterministic finite accepter M such that

$$L = L(M)$$

Observation:

All languages accepted by DFAs form the family of regular languages

Examples of regular languages:

$\{abba\}$ $\{\lambda, ab, abba\}$

$\{awa : w \in \{a,b\}^*\}$ $\{a^n b : n \geq 0\}$

{ all strings with prefix ab }

{ all strings without substring 001 }

Can you draw a DFA for this Language...

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

There exist languages which are not Regular:

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

There is no FA that accepts such a language