Deterministic Finite Automata (DFA)

*The term "**deterministic**" refers to the fact that on each input there is one and only one state to which the automaton can transition from its current state.*

The machine can exist in only one state at any given time.

Deterministic Finite Automata (DFA)

❑ Definition

A DFA is defined by a five-tuple

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

where

Q is a finite set of *states*.

Σ is a finite *input alphabet*.

$\delta : Q \times \Sigma \rightarrow Q$ is the *transition function.*

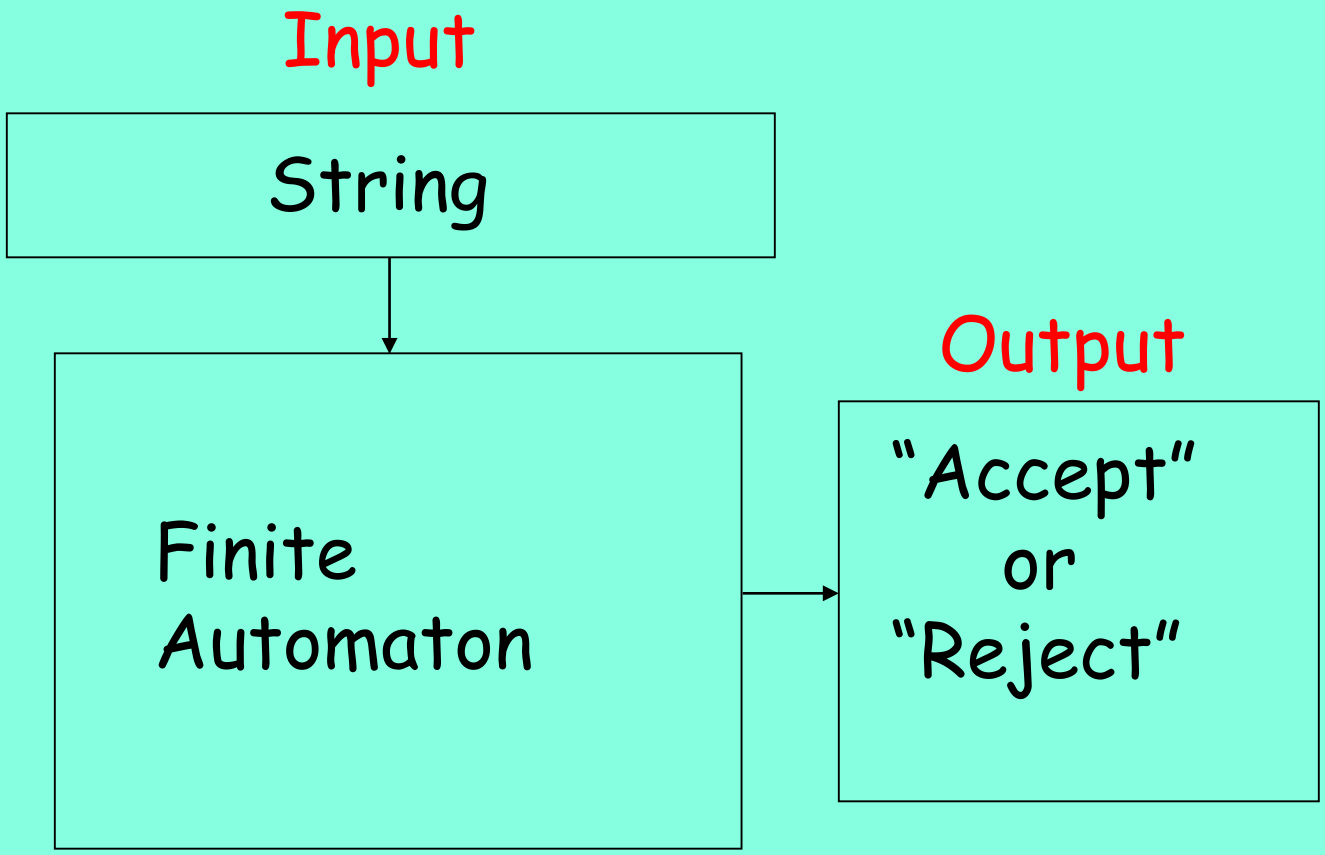
$q_0 \in Q$ is the *start/initial state.*

$F \subseteq Q$ is a set of *final/accepting states*.

What does a DFA do on reading an input string?

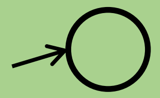
- Input: a word w in Σ^*
- Question: Is w acceptable by the DFA?
- Steps:
  - Start at the "start state" q_0
  - For every input symbol in the sequence w do
    - Compute the next state from the current state, given the current input symbol in w and the transition function
  - If after all symbols in w are consumed, the current state is one of the final states (F) then *accept w;*
  - Otherwise, *reject w.*

Finite Accepter

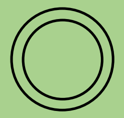


Transition Diagram

- Directed graph consists of set of vertices and edges where vertices represent “states” and edges represent “input/output”
- Circle with an arrow is called initial state

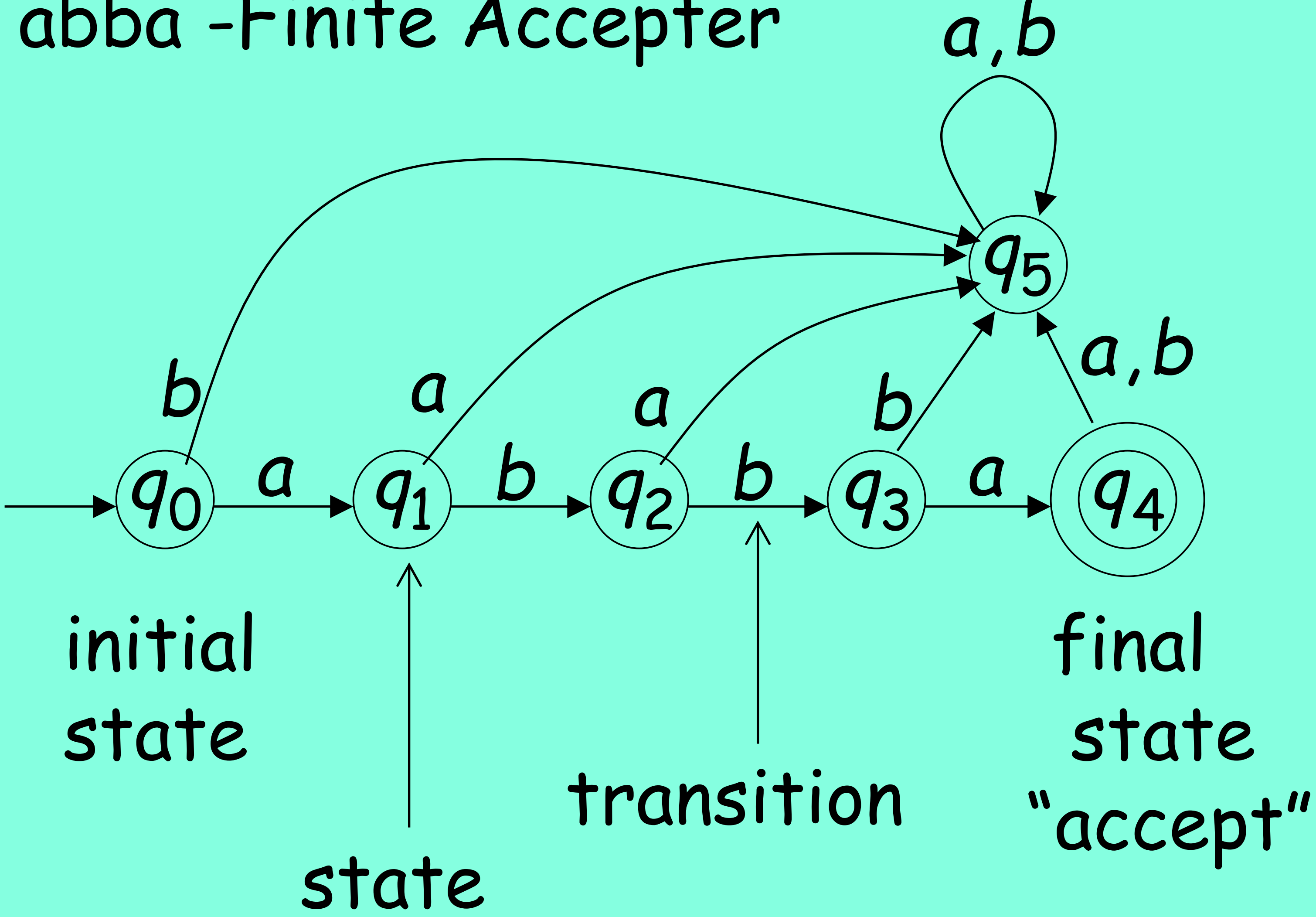


- Two concentric circle represents the final state



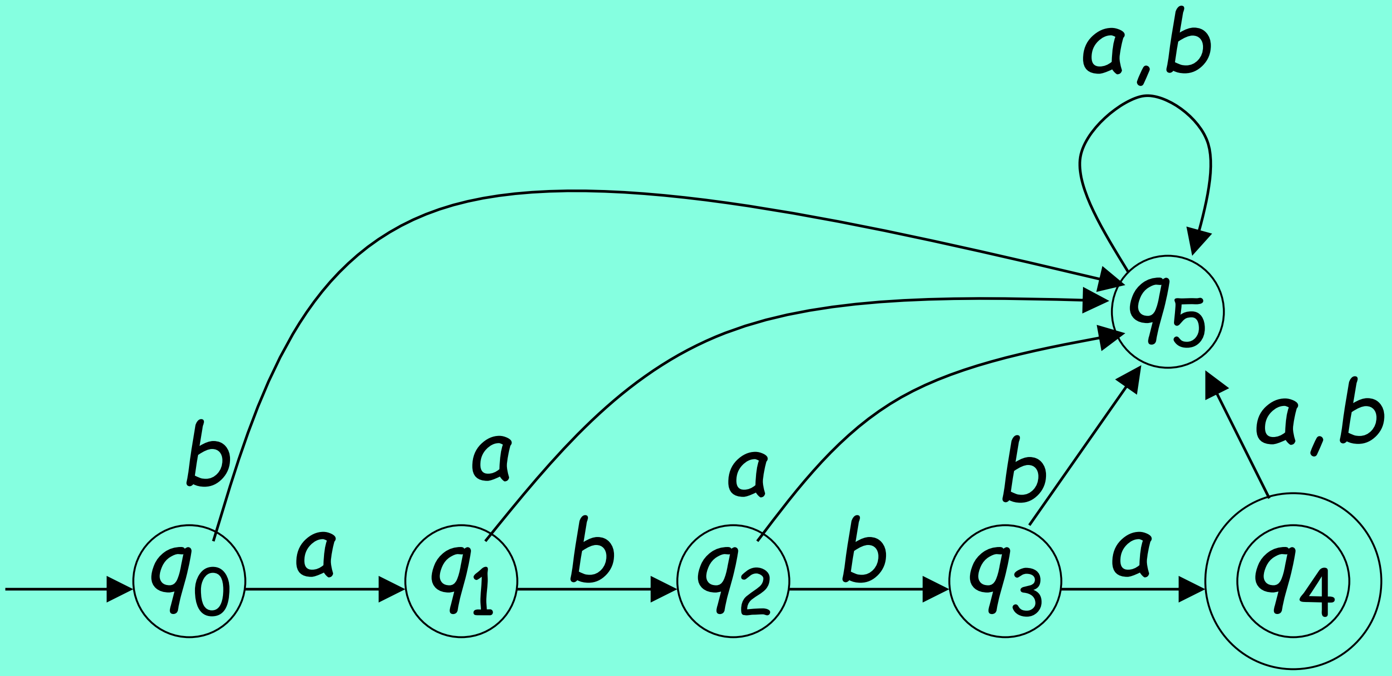
Transition Diagram

abba -Finite Accepter



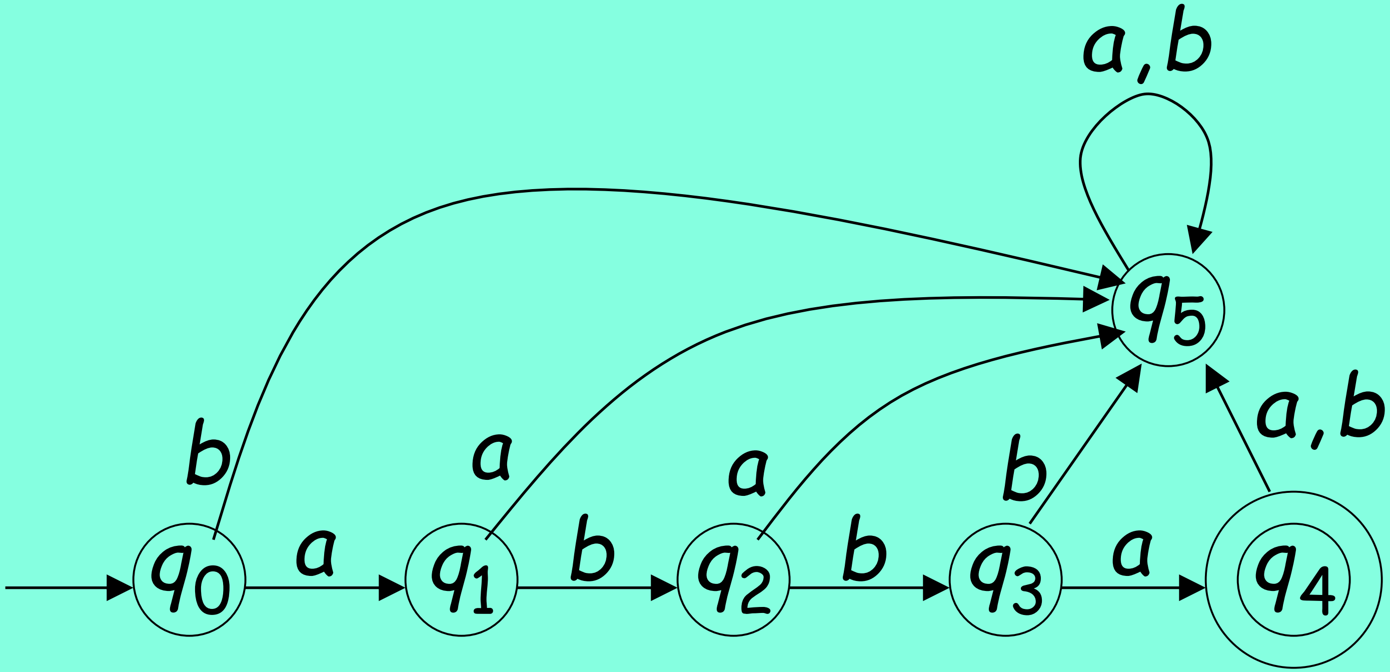
Input Alphabet Σ

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

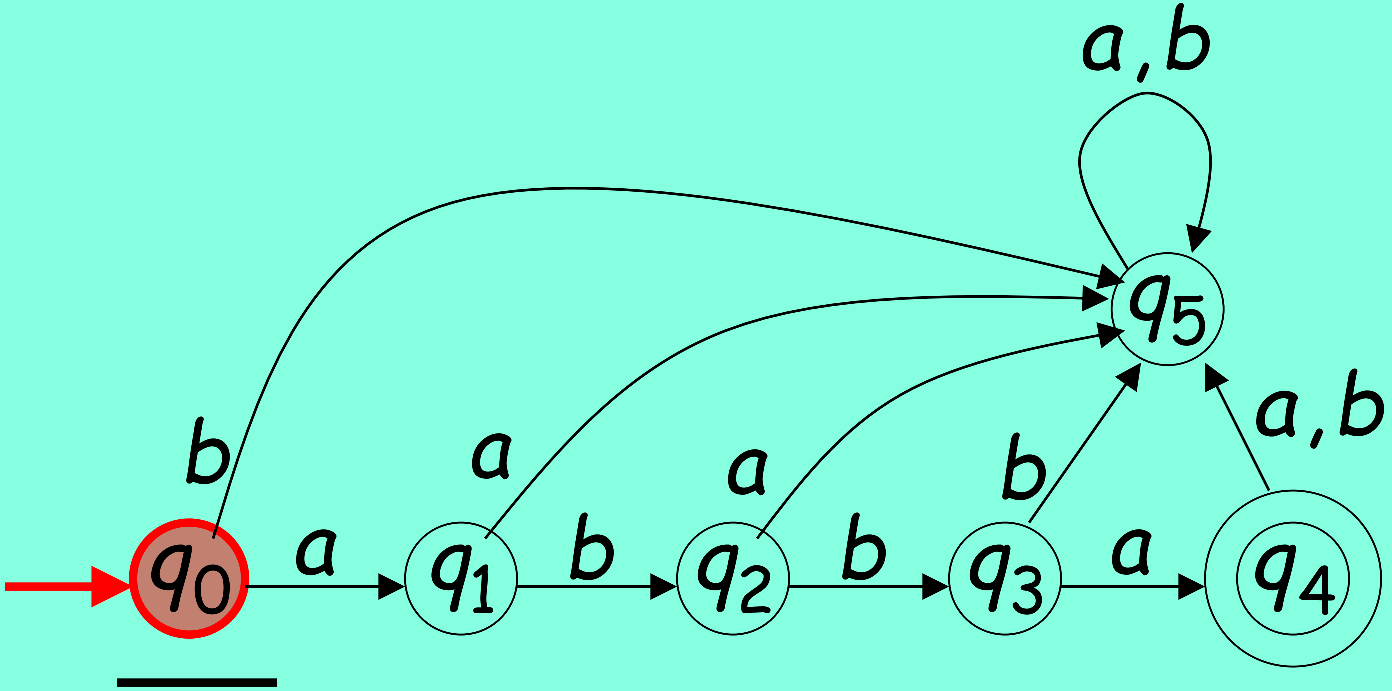


Set of States Q

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

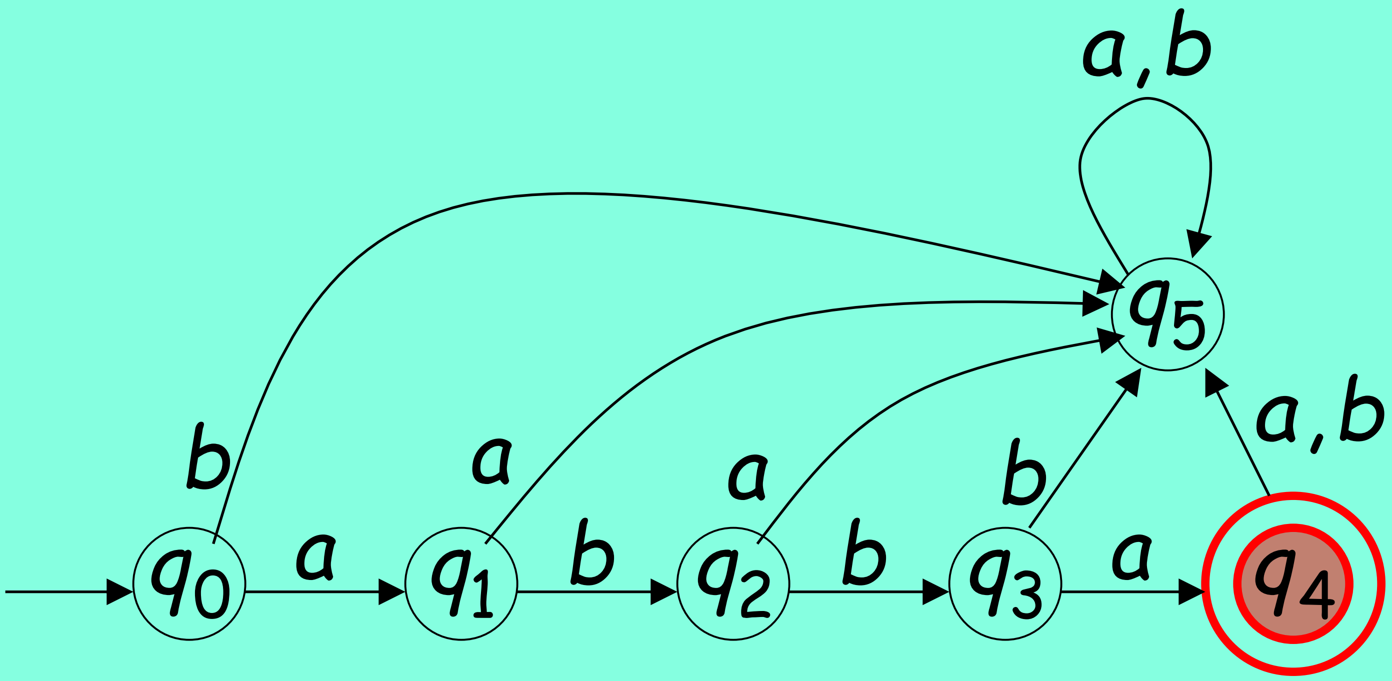


Initial State q_0



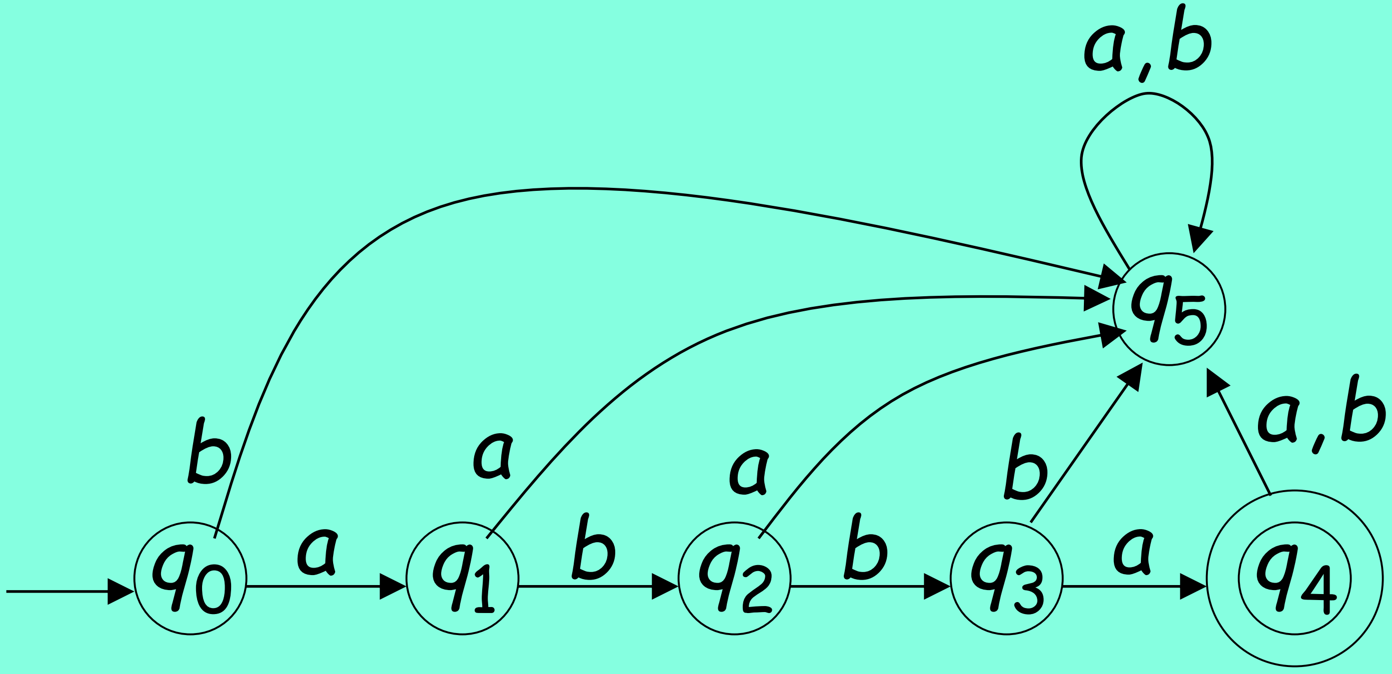
Set of Final States F

$$F = \{q_4\}$$

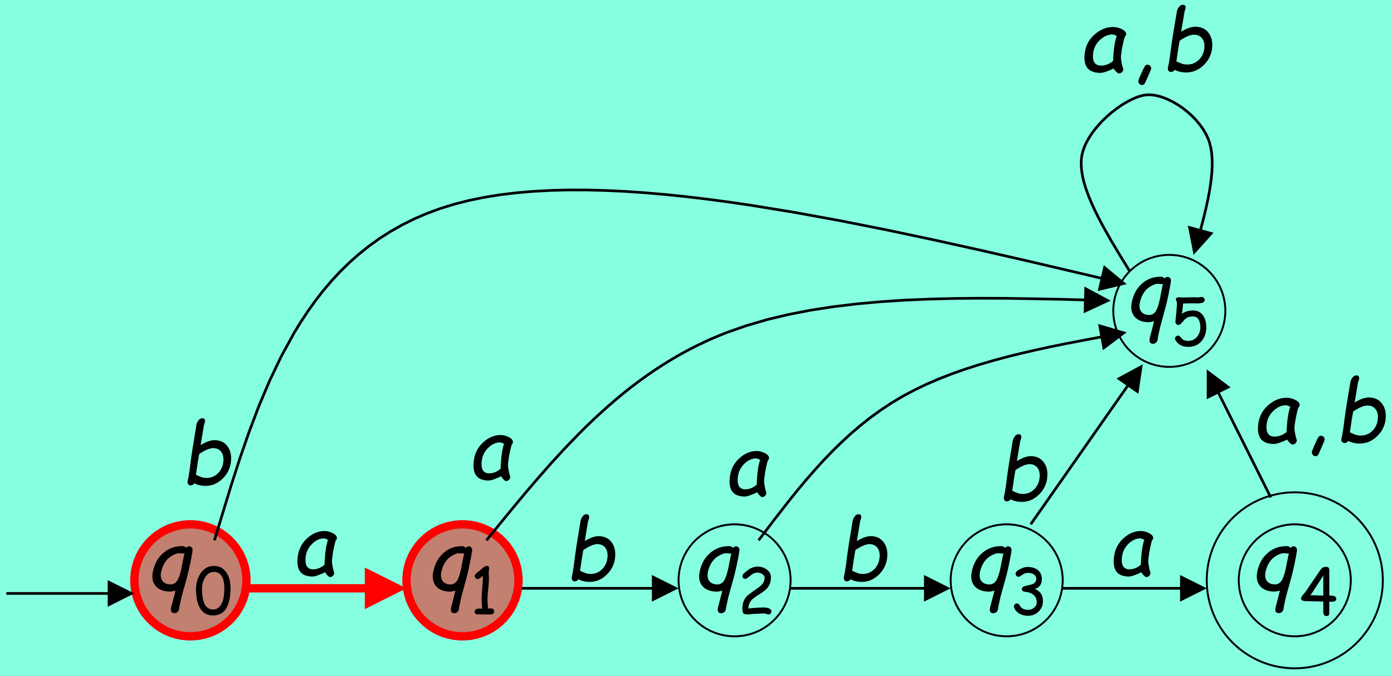


Transition Function δ

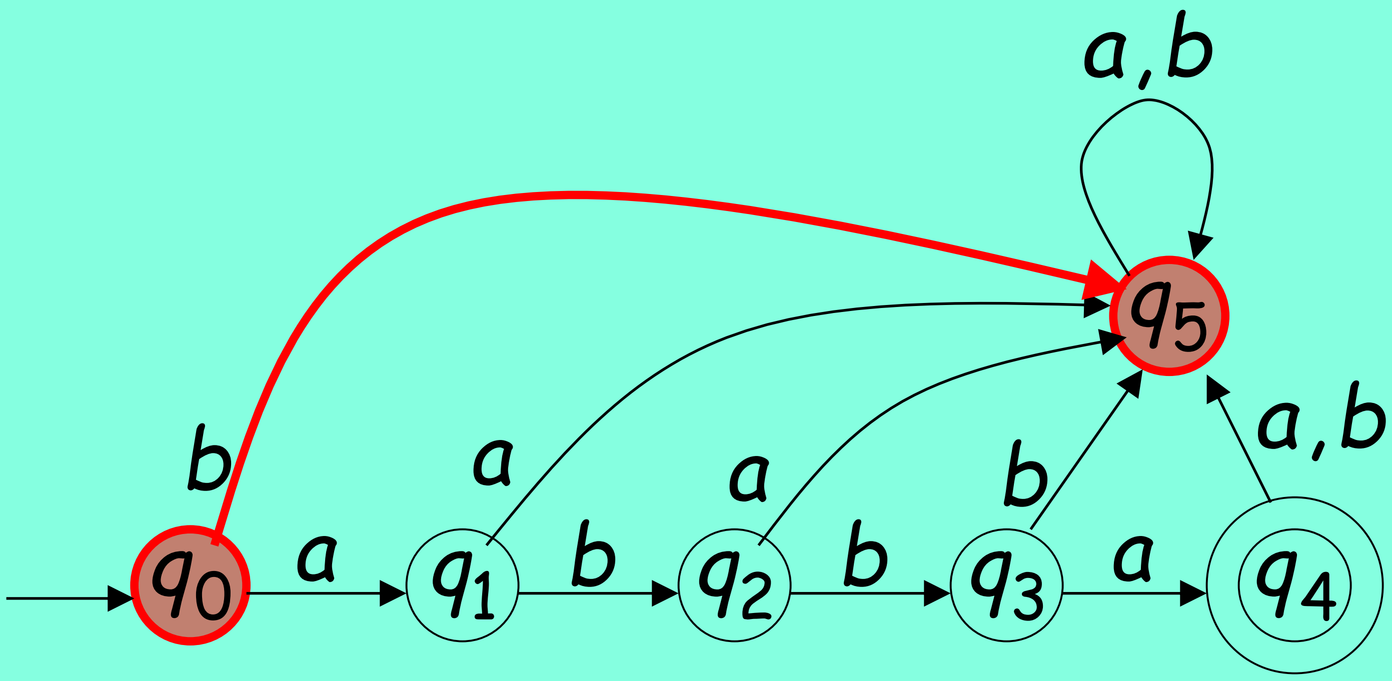
$$\delta : Q \times \Sigma \rightarrow Q$$



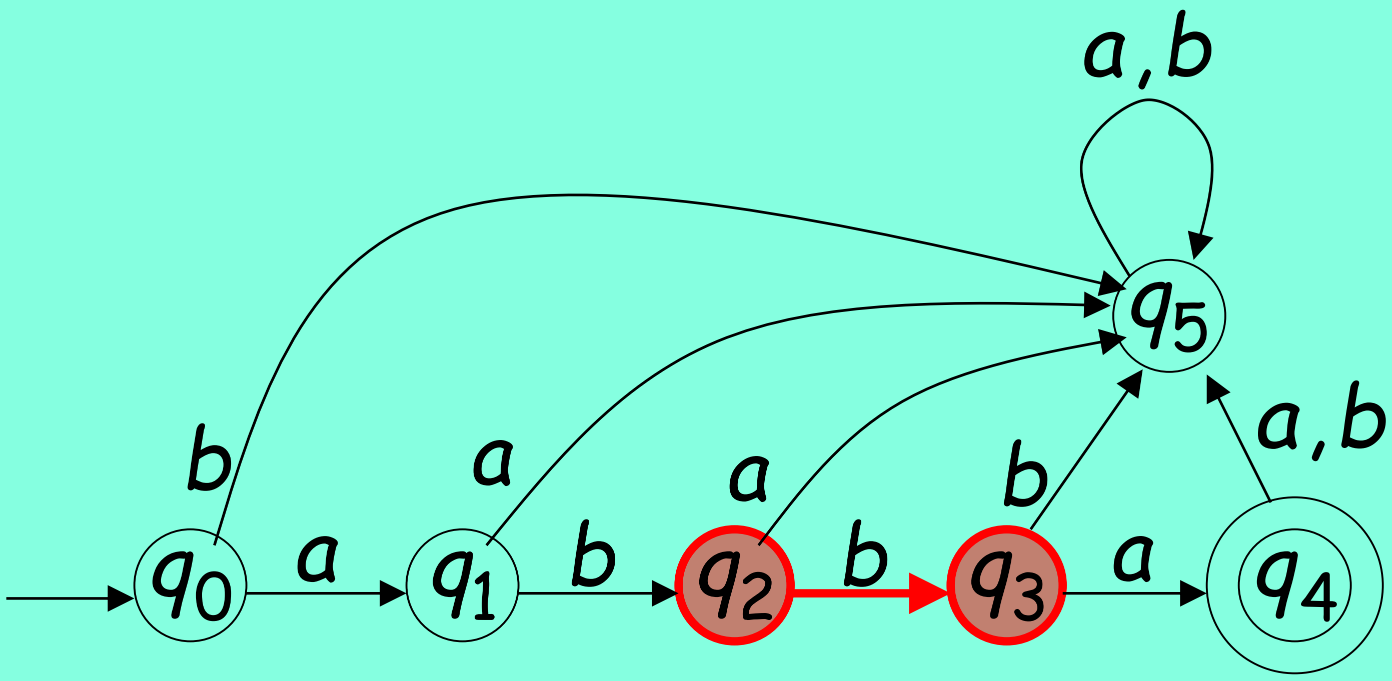
$$\delta(q_0, a) = q_1$$



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

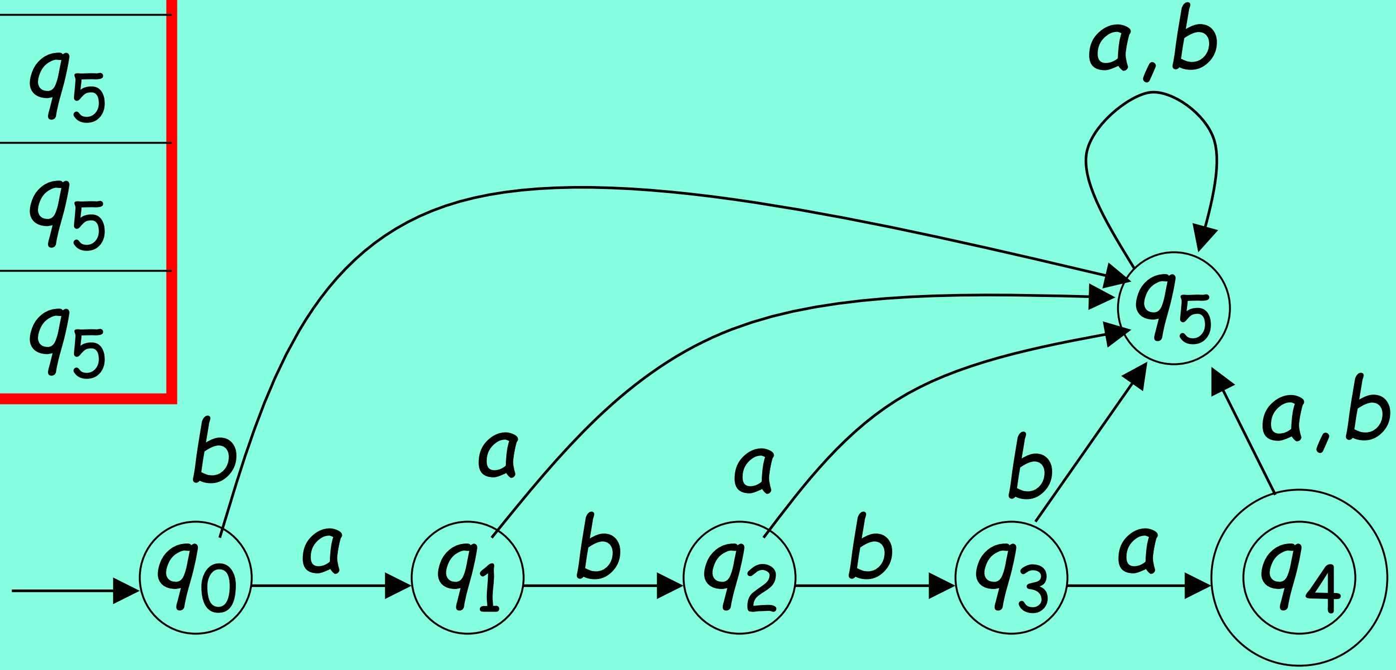


$$\delta(q_2, b) = q_3$$

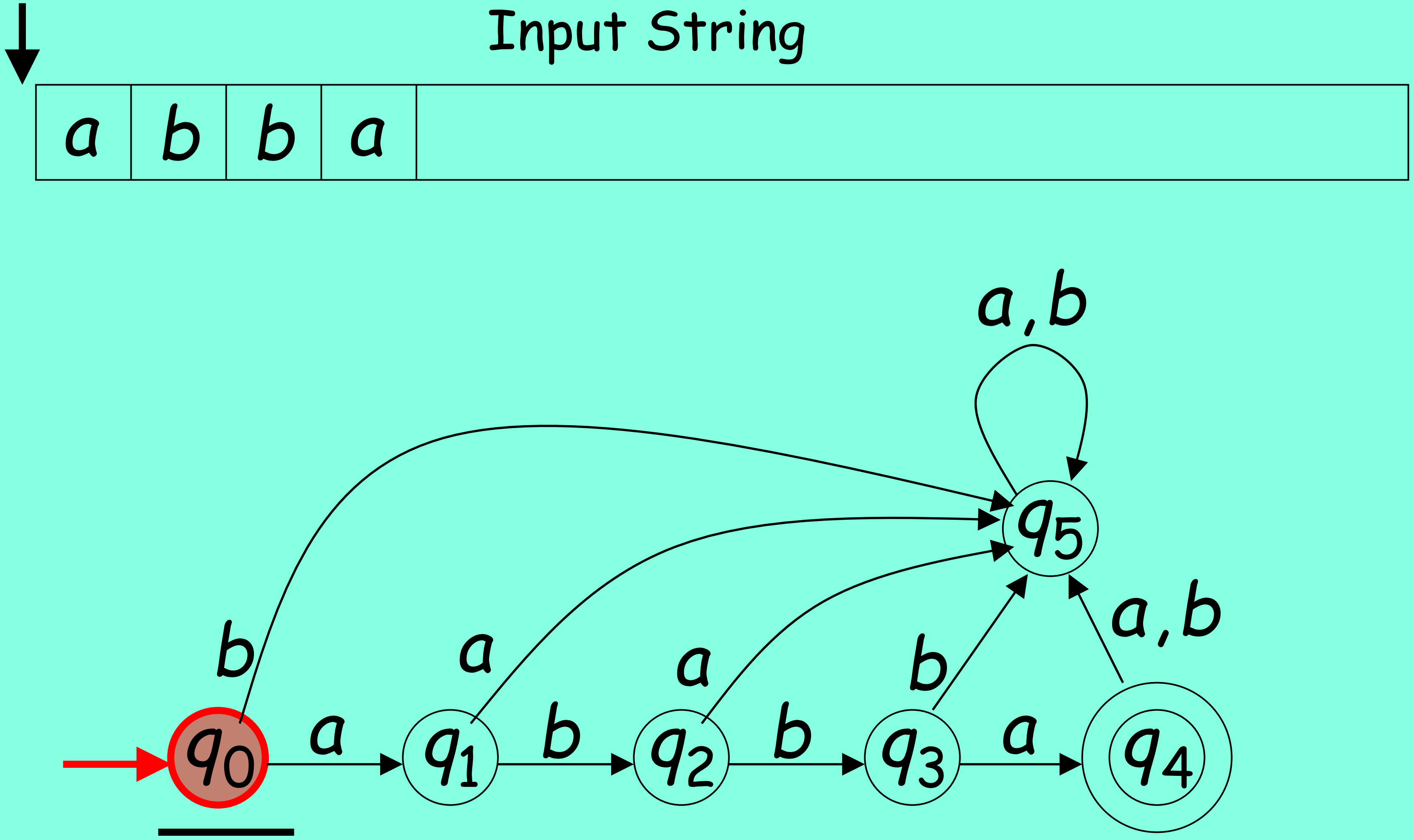


Transition Table

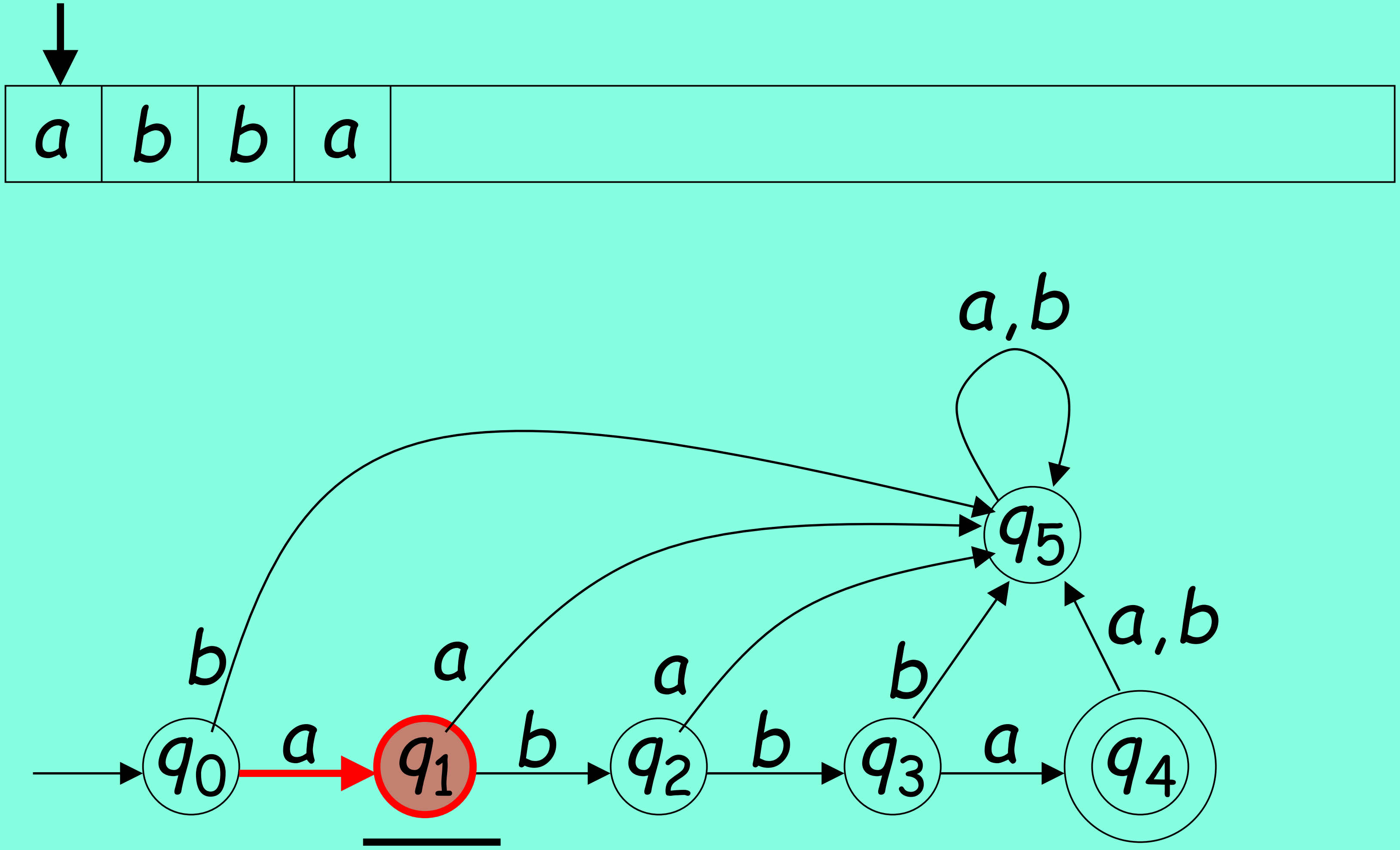
δ	a	b
q_0	q_1	q_5
q_1	q_5	q_2
q_2	q_5	q_3
q_3	q_4	q_5
q_4	q_5	q_5
q_5	q_5	q_5

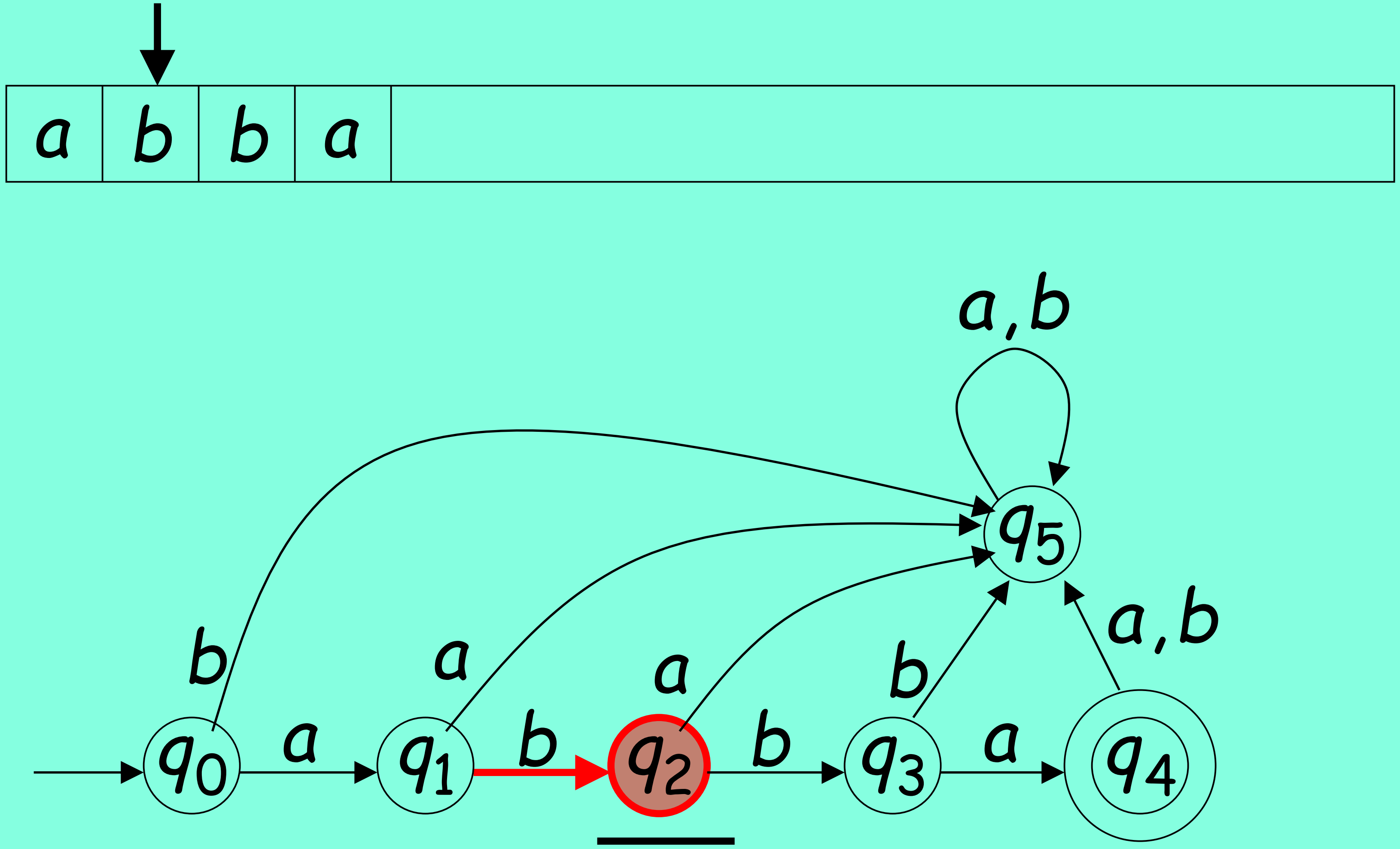


Initial Configuration

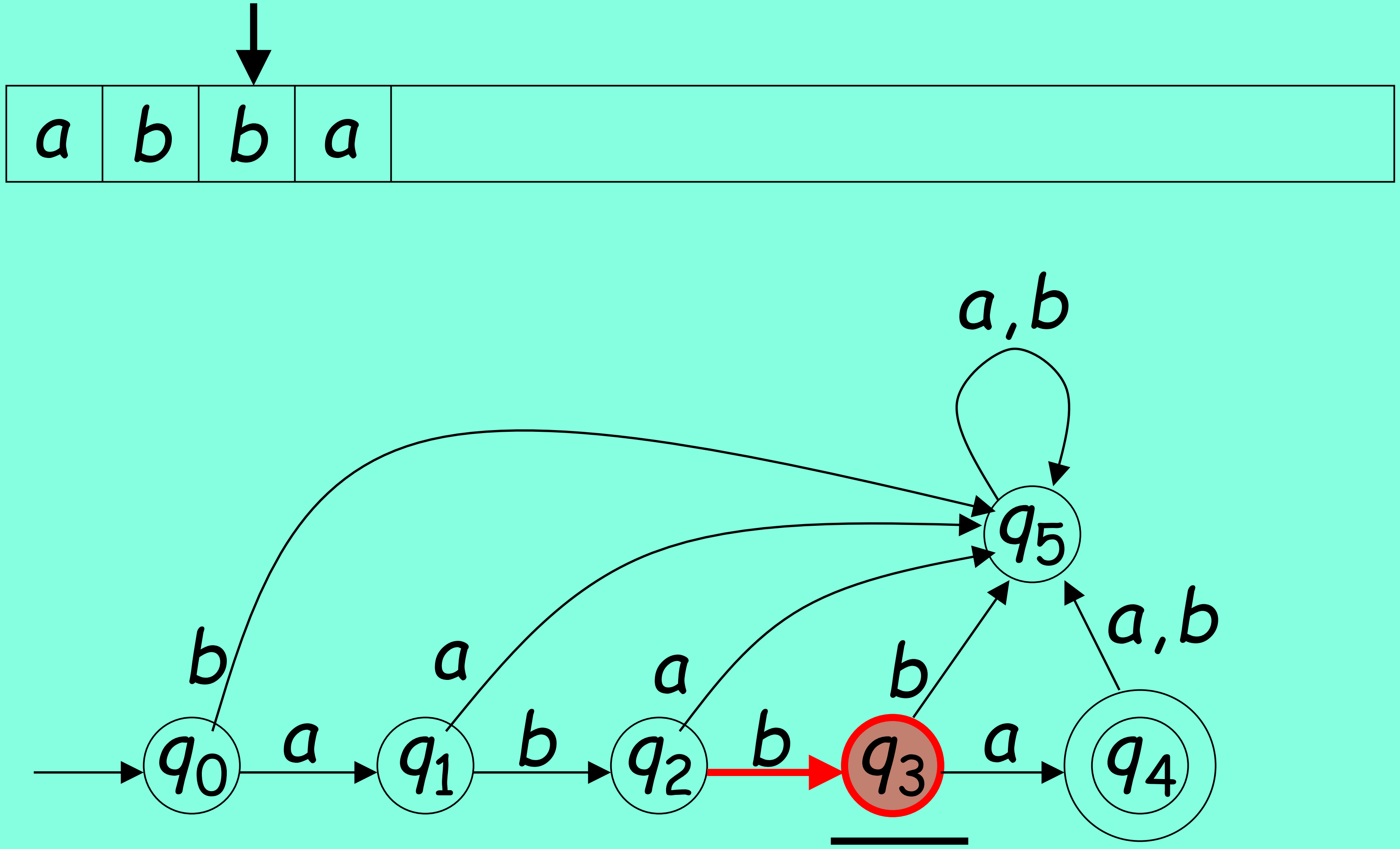


Reading the Input

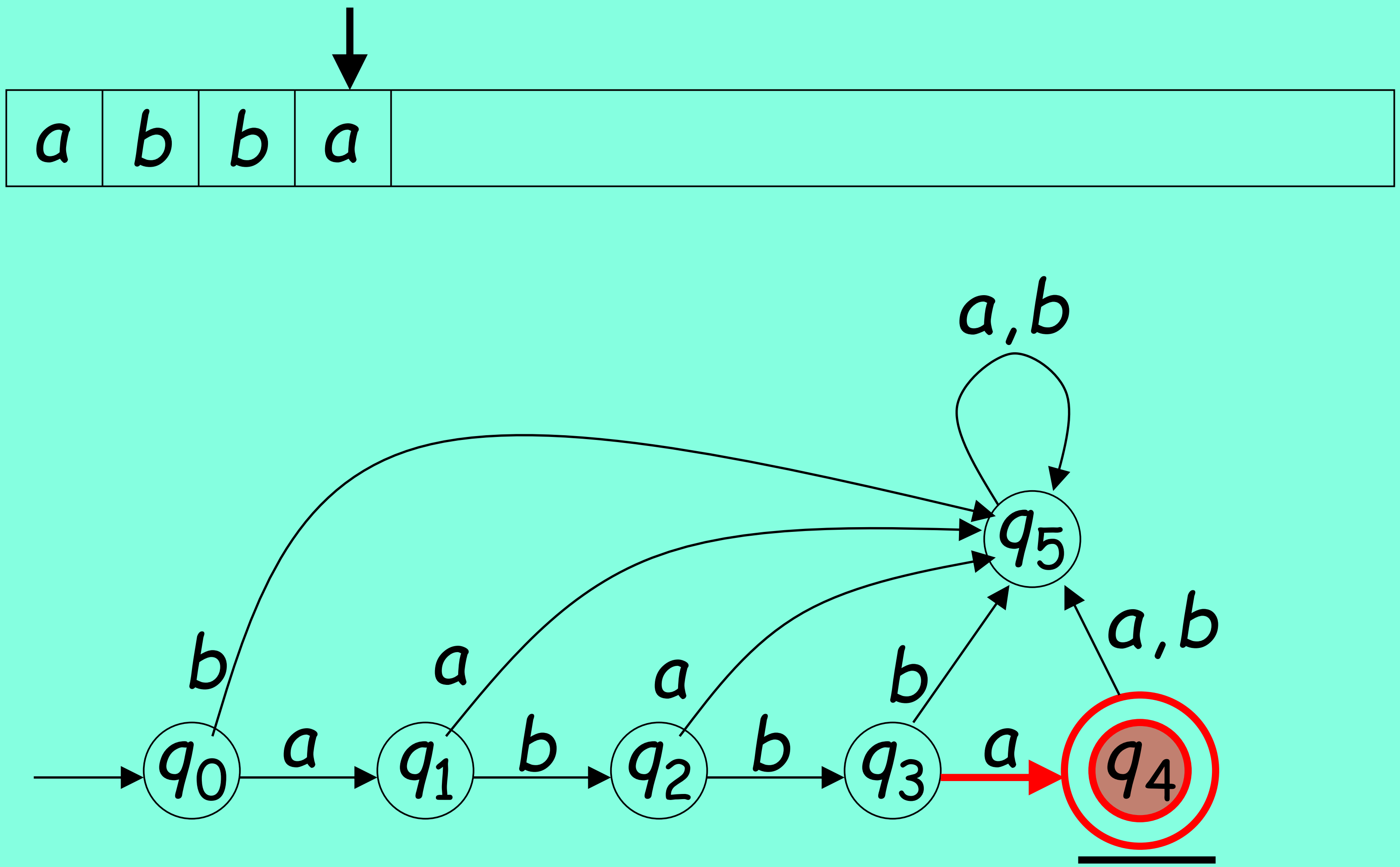




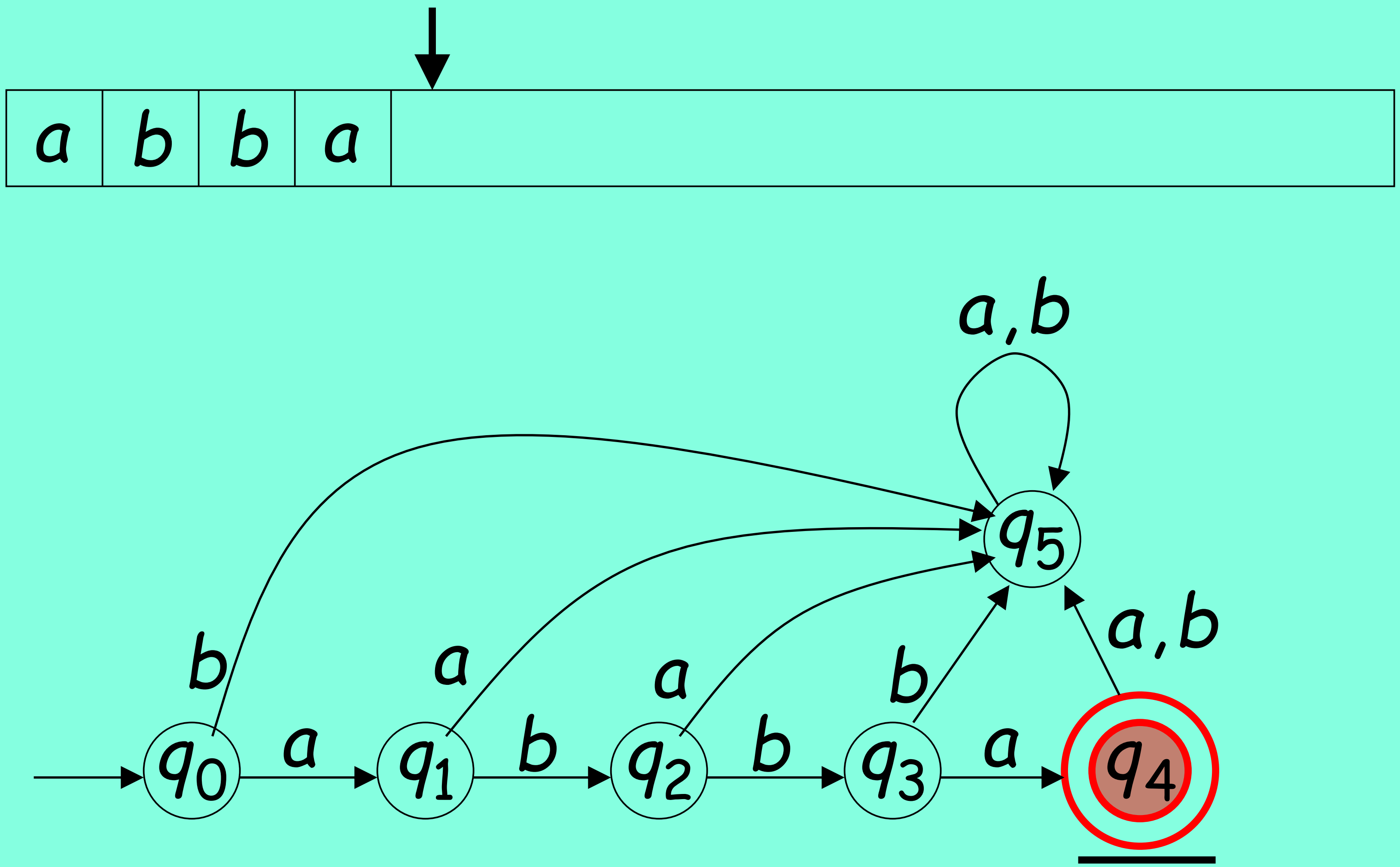
a b b a
a,b
b
a
a
b
a,b
a
b
b
a
q_0
q_1
q_2
q_3
q_4
q_5



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

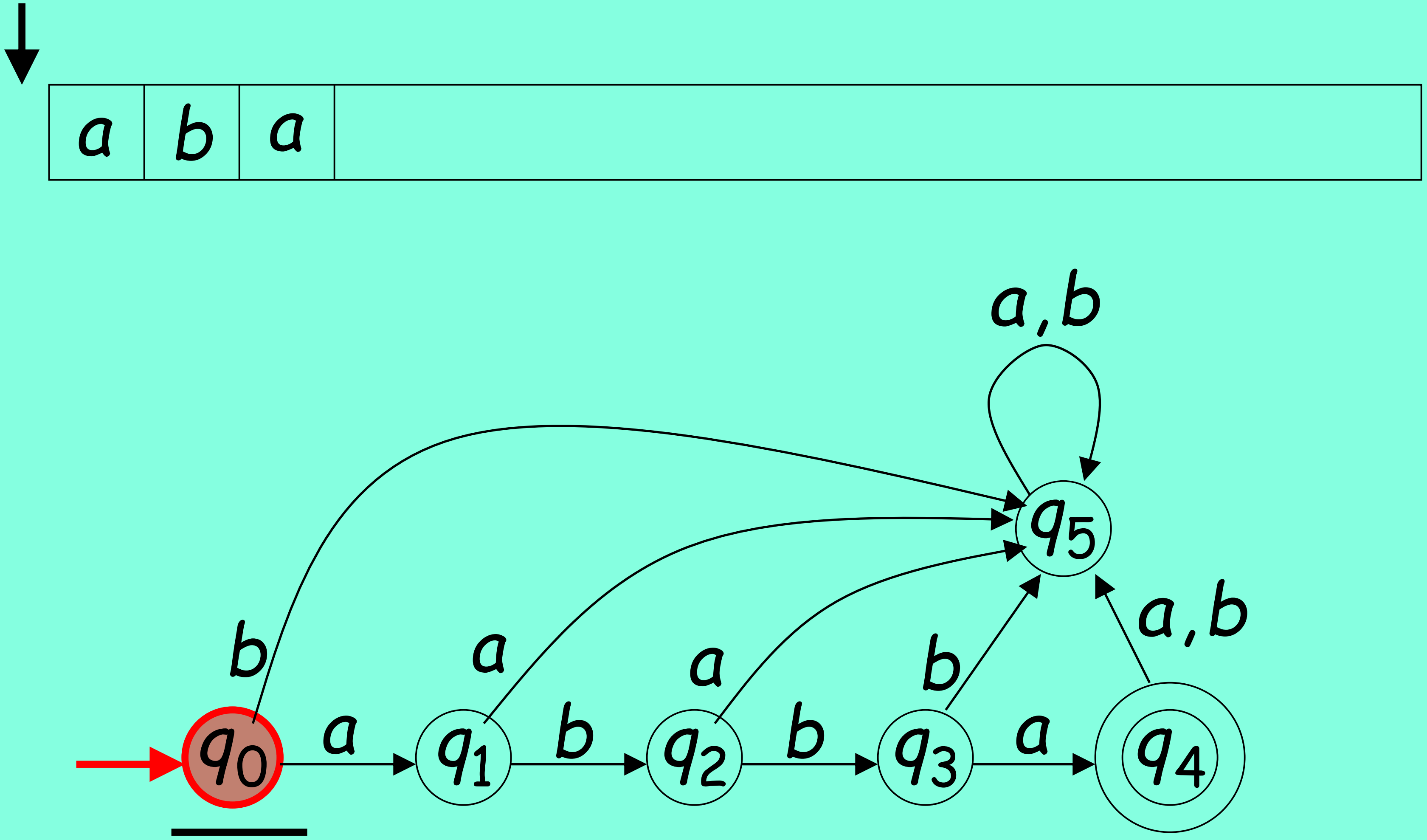

a b b a
a,b
b
a
a
b
a
a,b
a
b
b
a
q_0
q_1
q_2
q_3
q_4
q_5

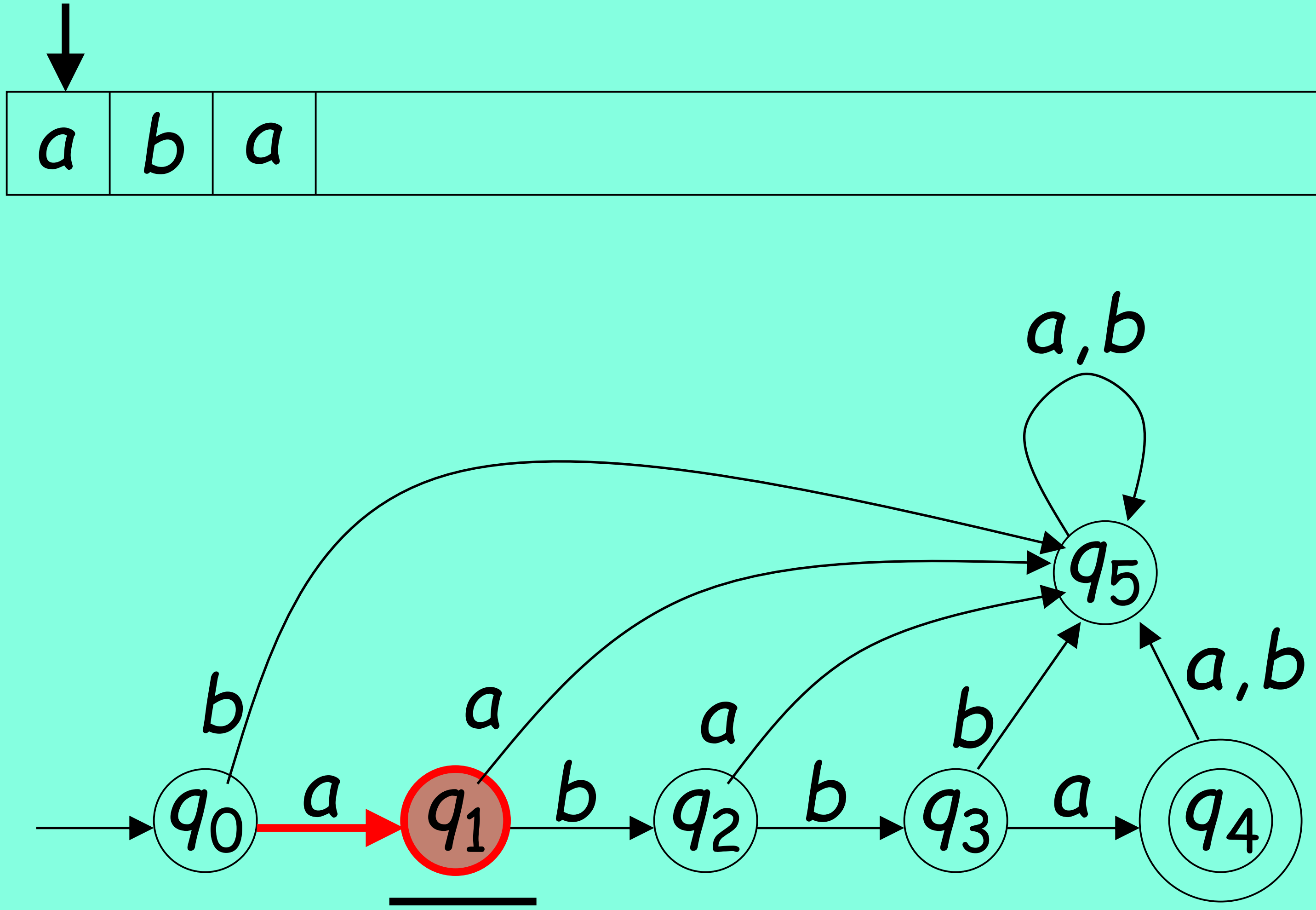
Input finished



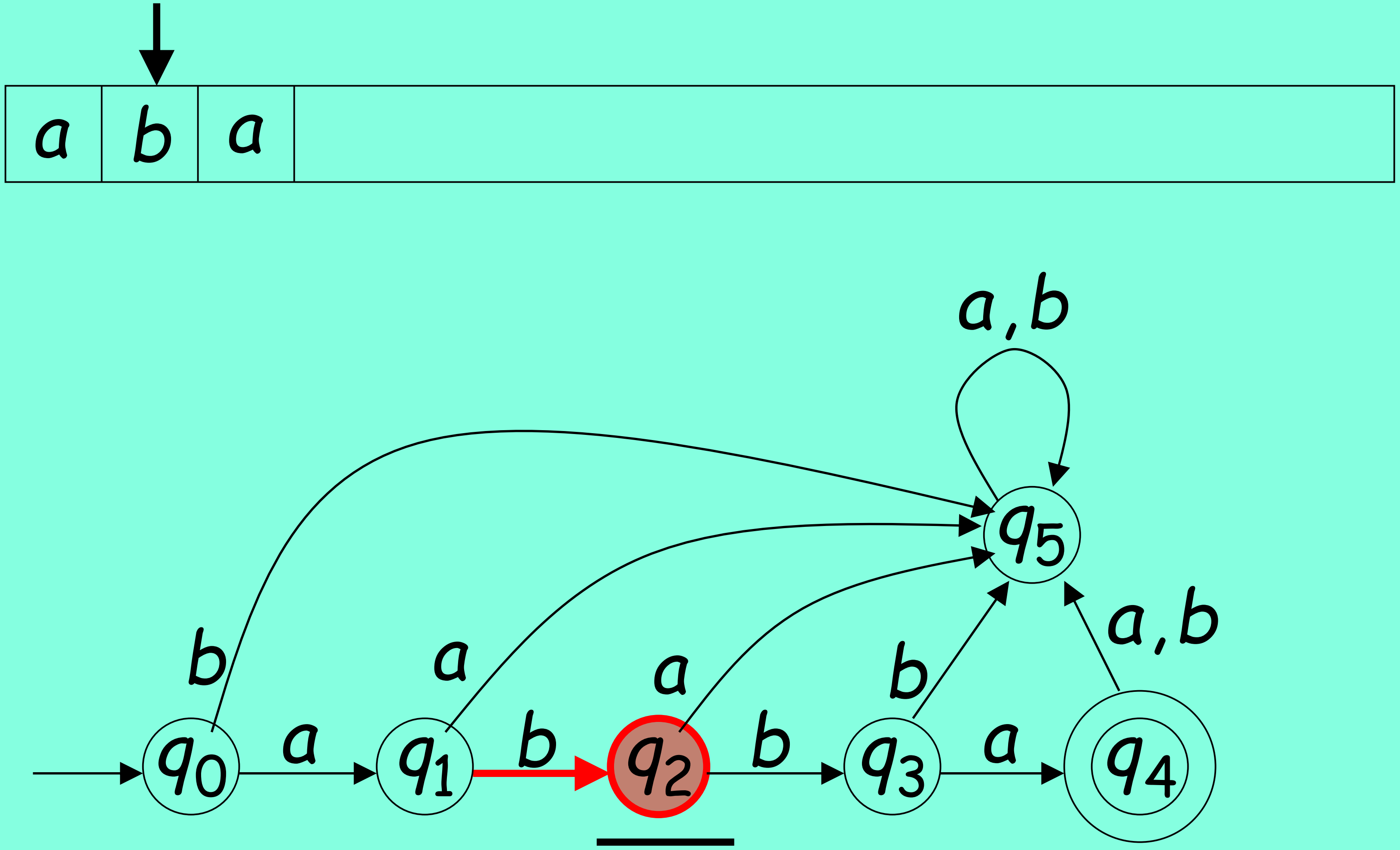
Output: “accept”

Rejection

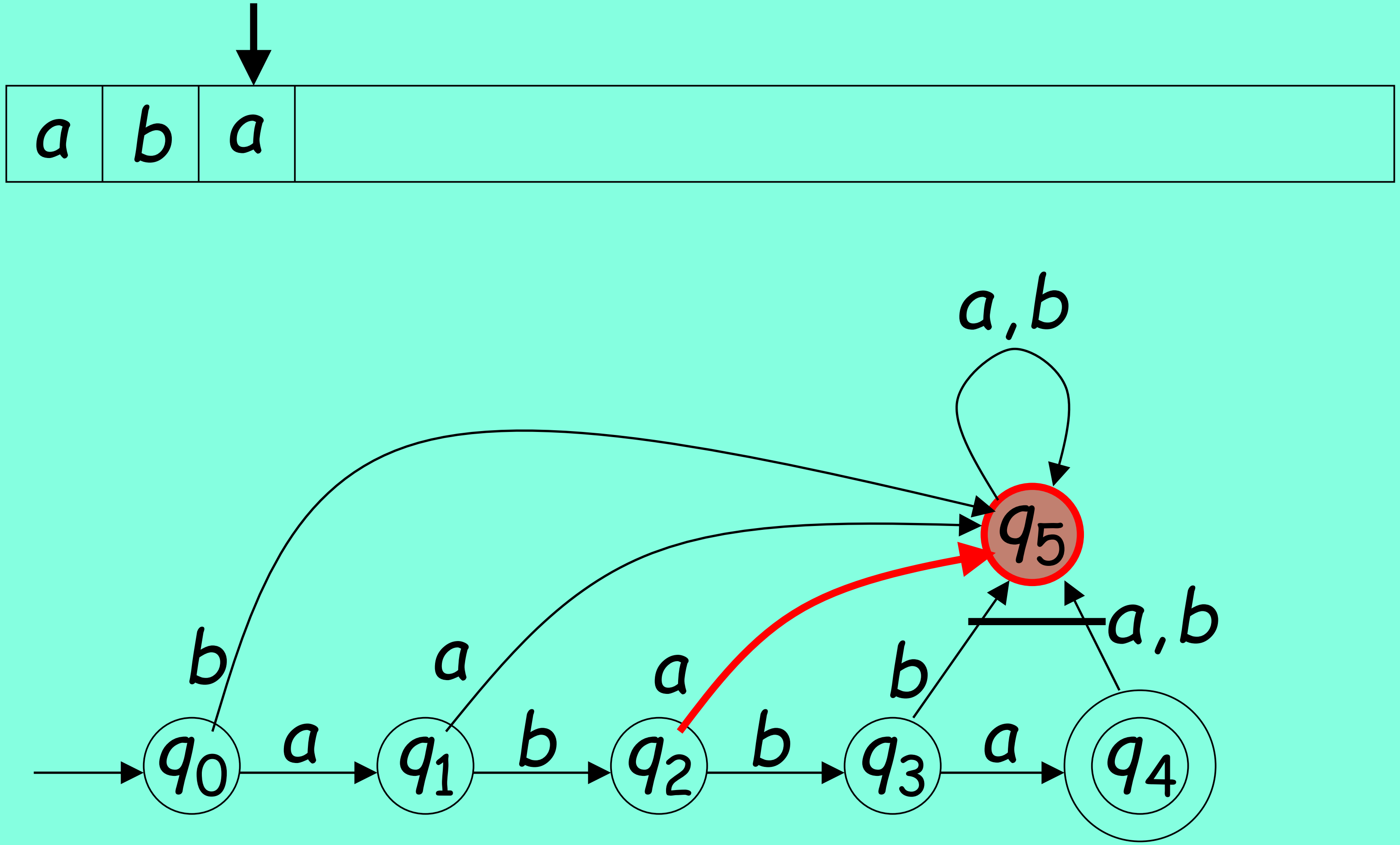




a
b
a
a,b
b
a
a
b
a,b
a
b
b
a
q_0
q_1
q_2
q_3
q_4
q_5

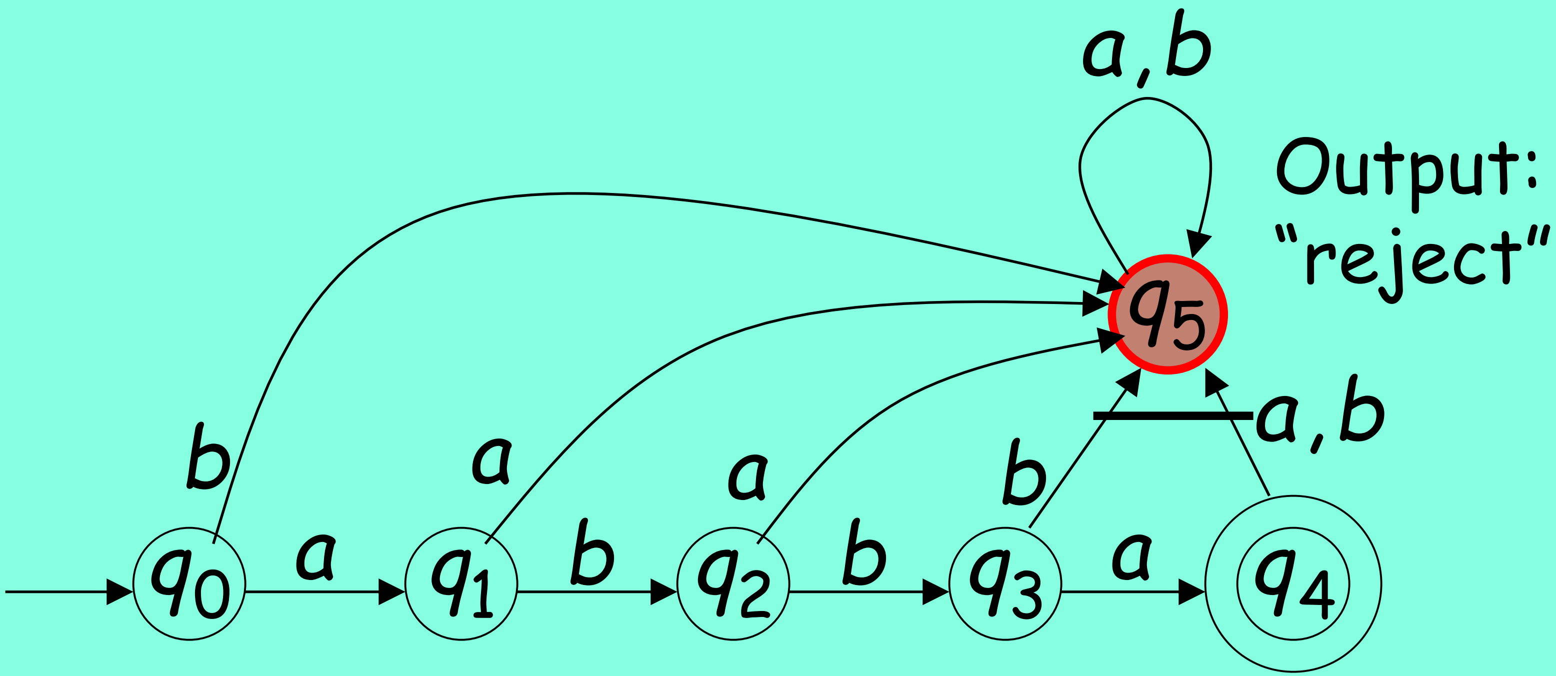


a
b
a
a,b
b
a
a
b
a,b
q_0
a
q_1
b
q_2
b
q_3
a
q_4
q_5

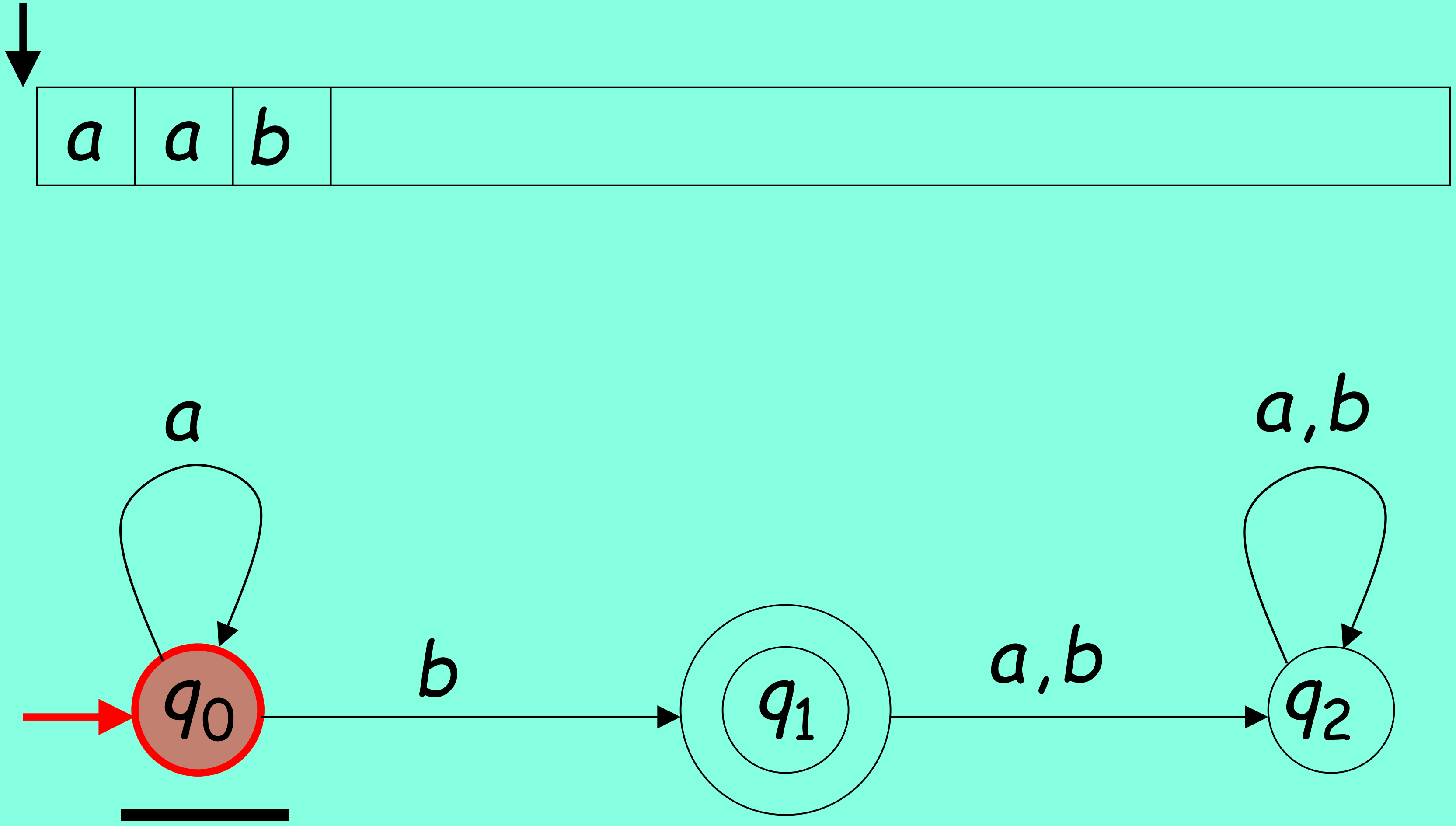

a
b
a
a,b
b
a
a
b
a,b
a
b
b
a
q_0
q_1
q_2
q_3
q_4
q_5

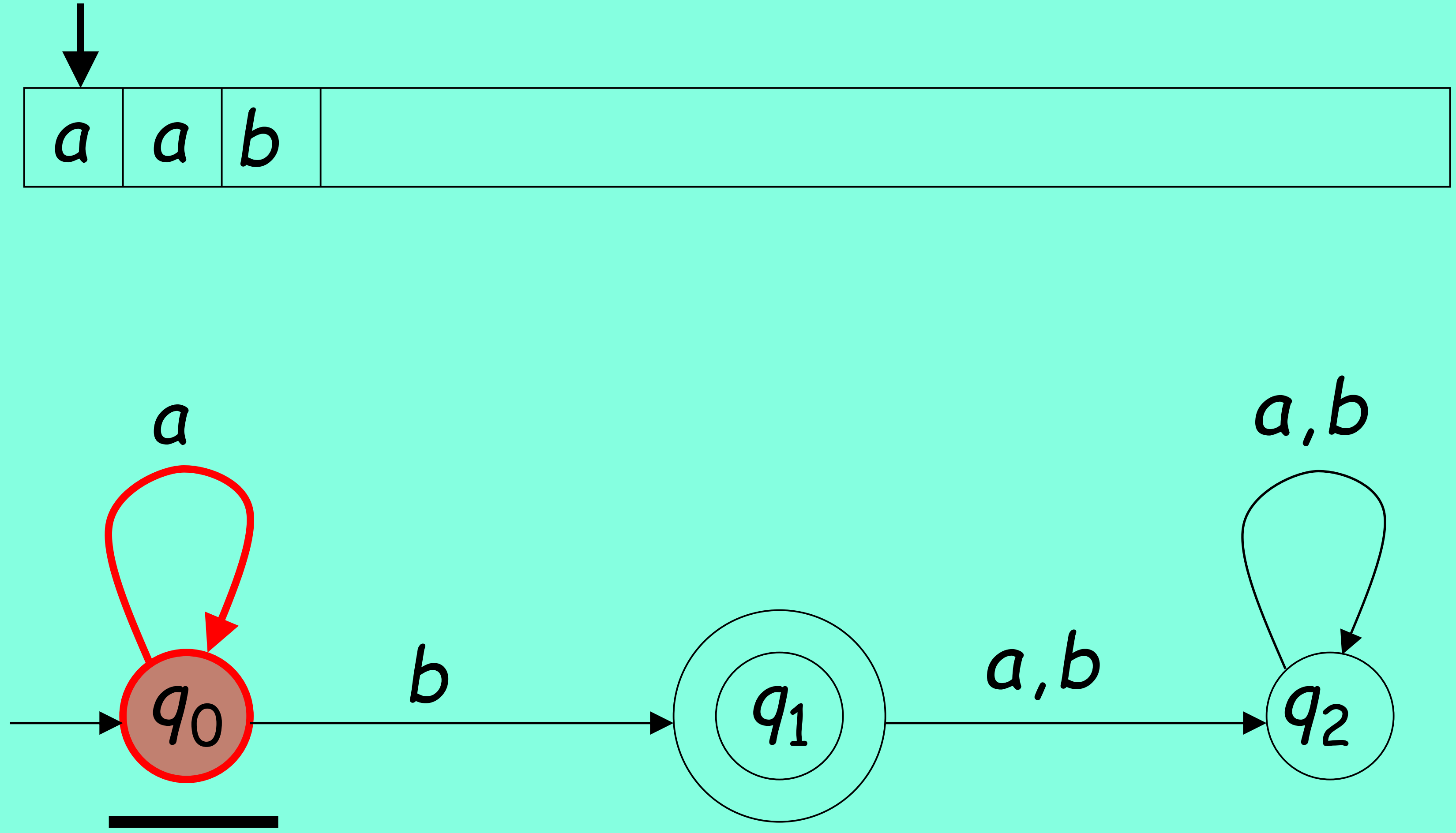
Input finished

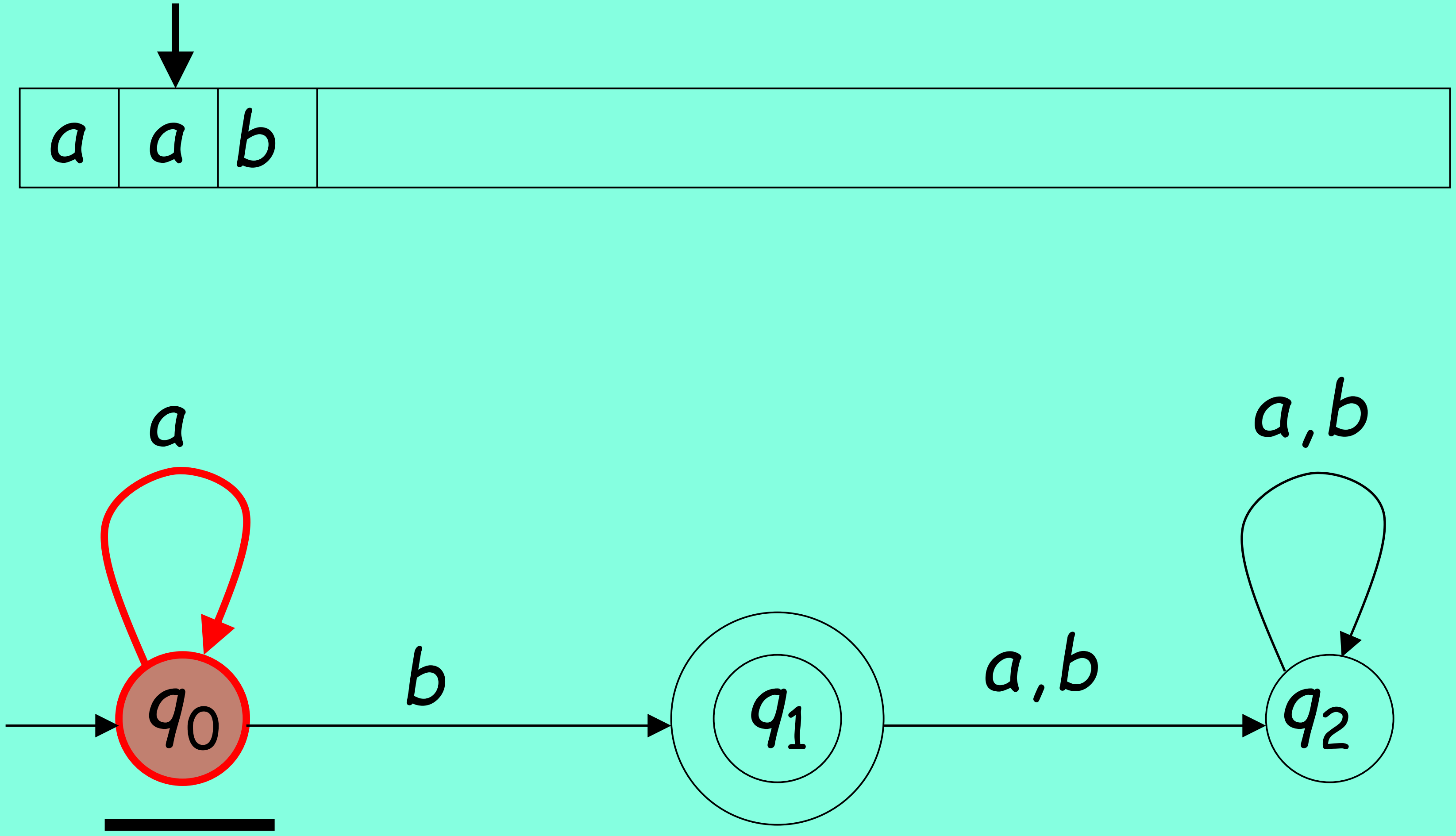
a	b	a	



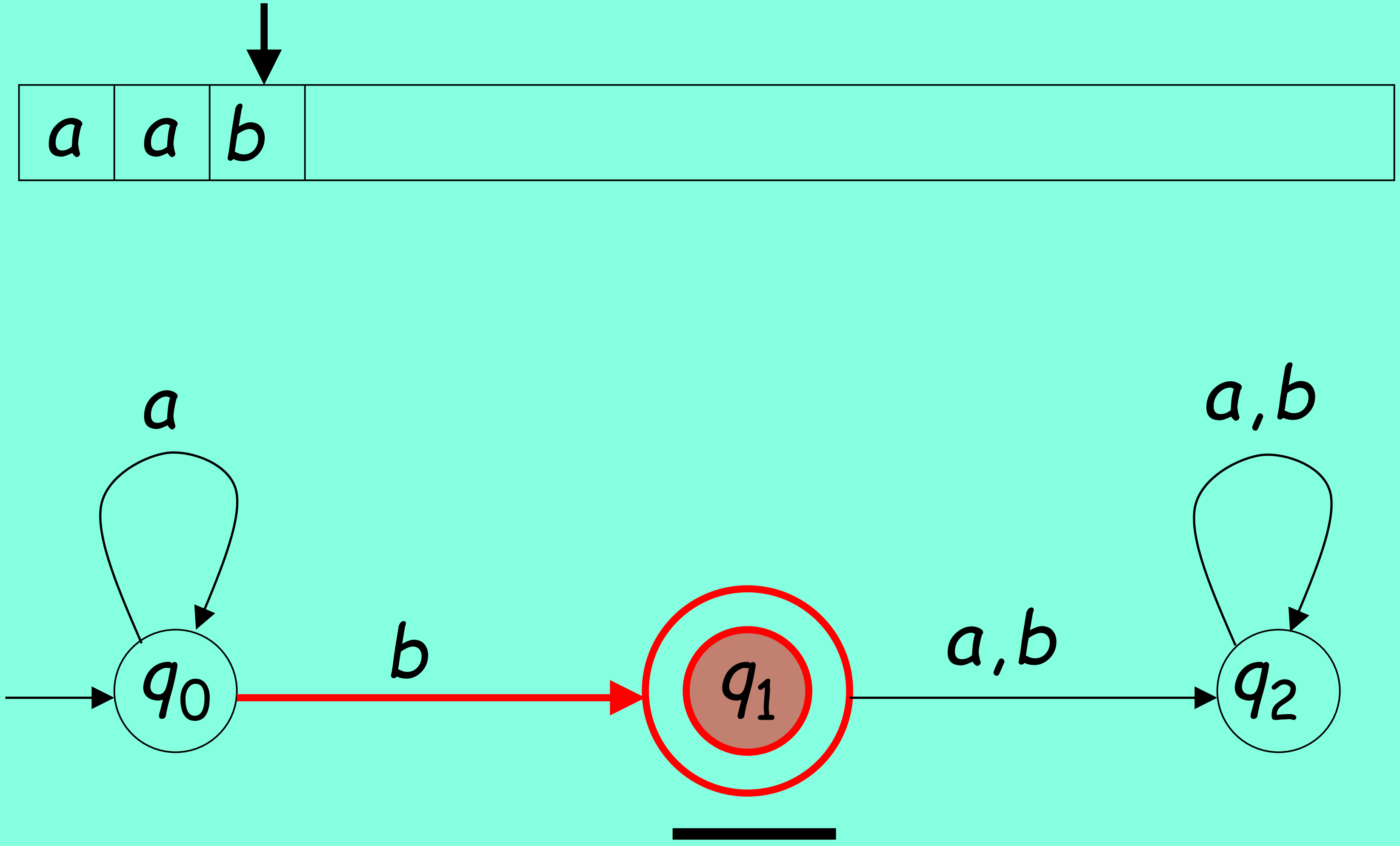
Another Example



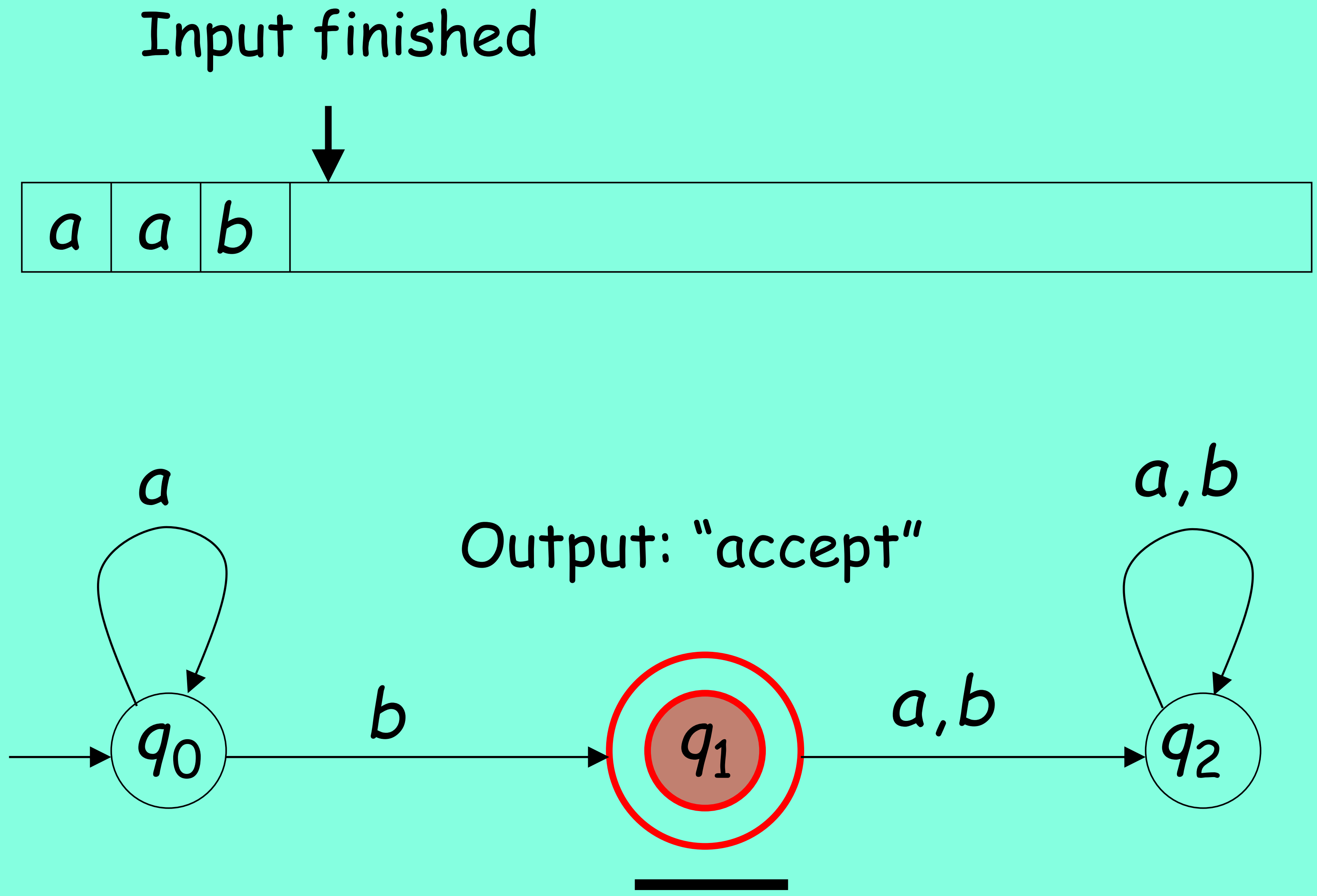

a a b
a
a,b
b
a,b
q_0
q_1
q_2



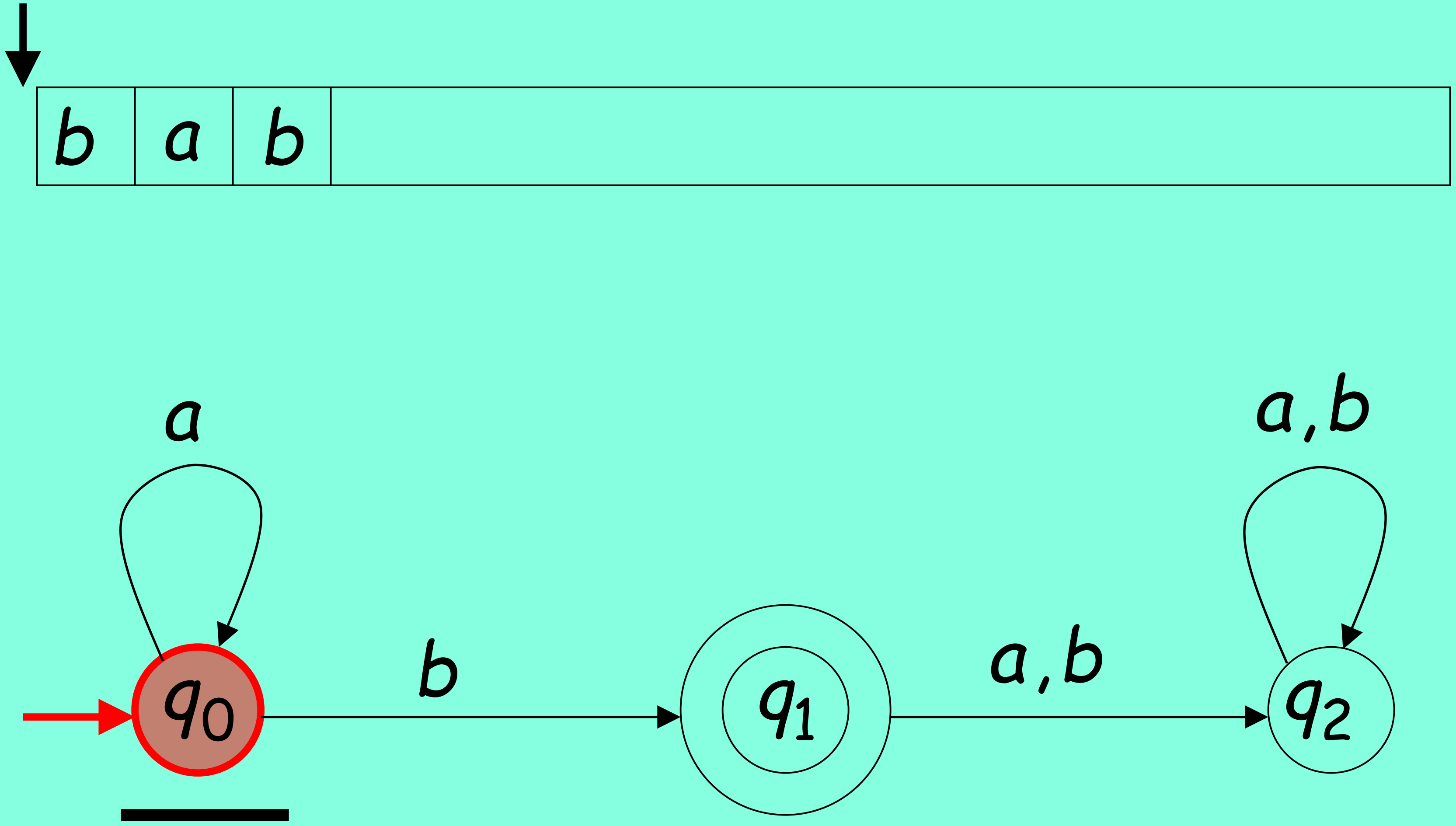
a
a
b
a
b
a,b
a,b
q0
q1
q2

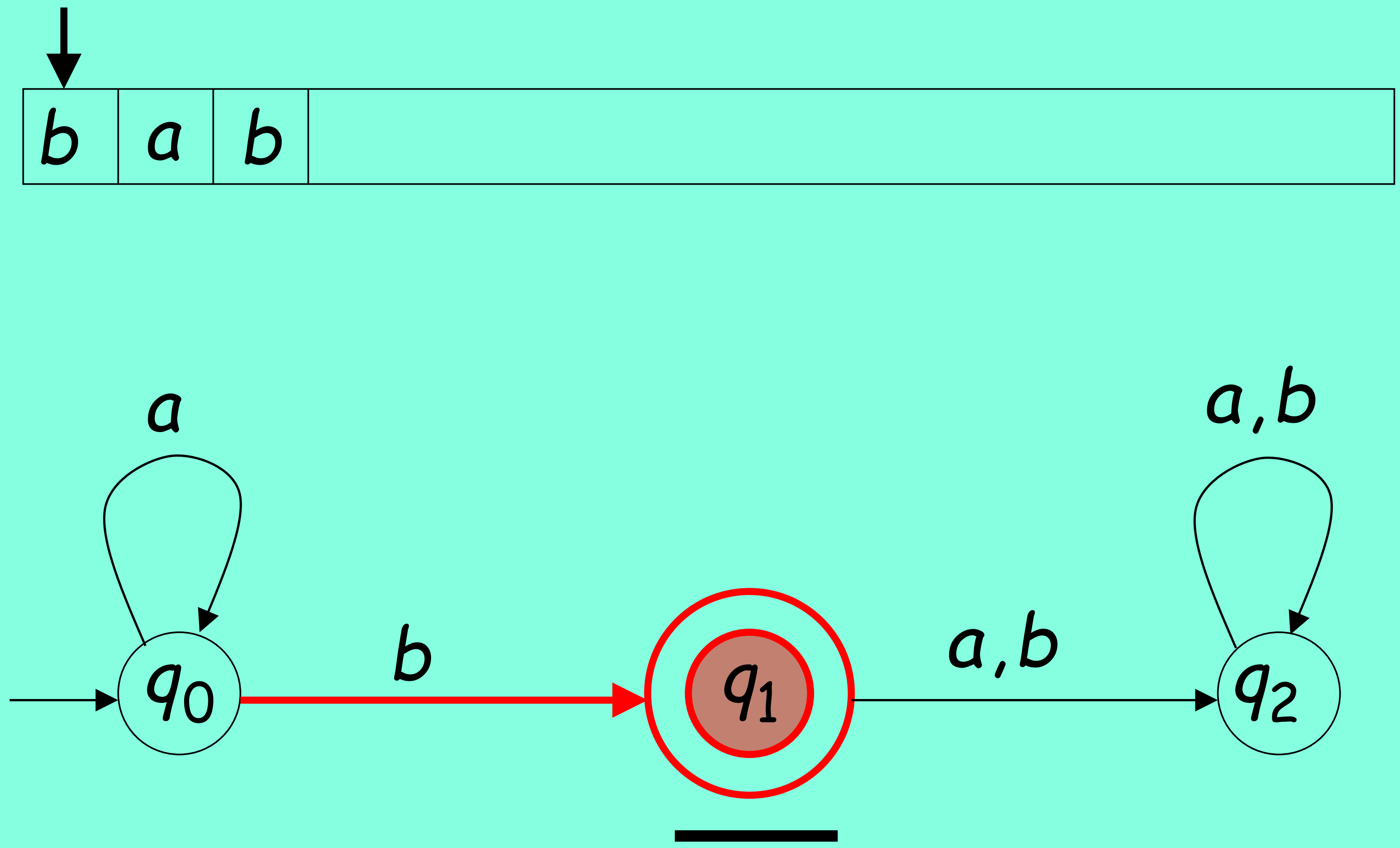


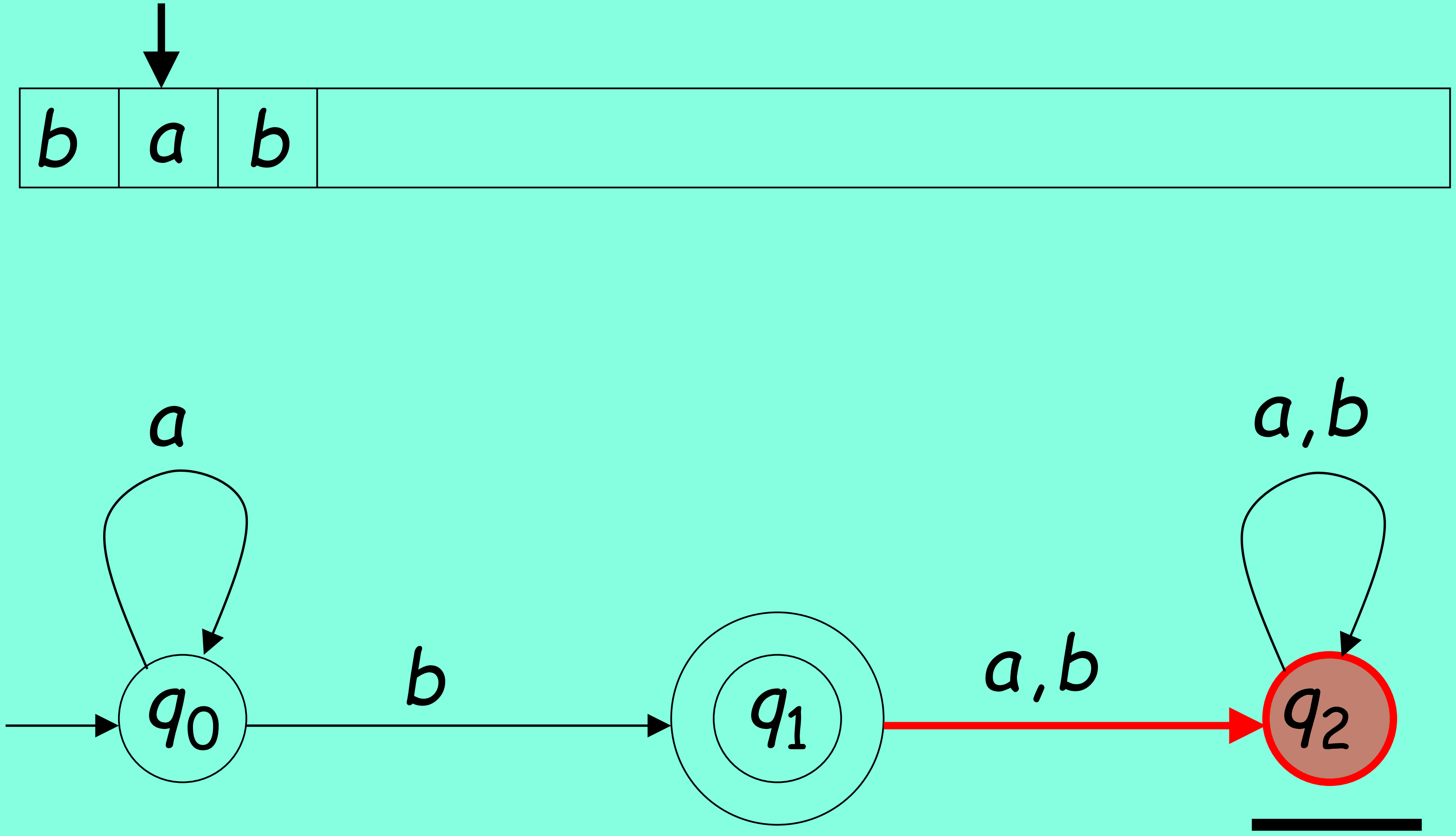
a
a
b
a
b
a,b
a,b
q_0
q_1
q_2

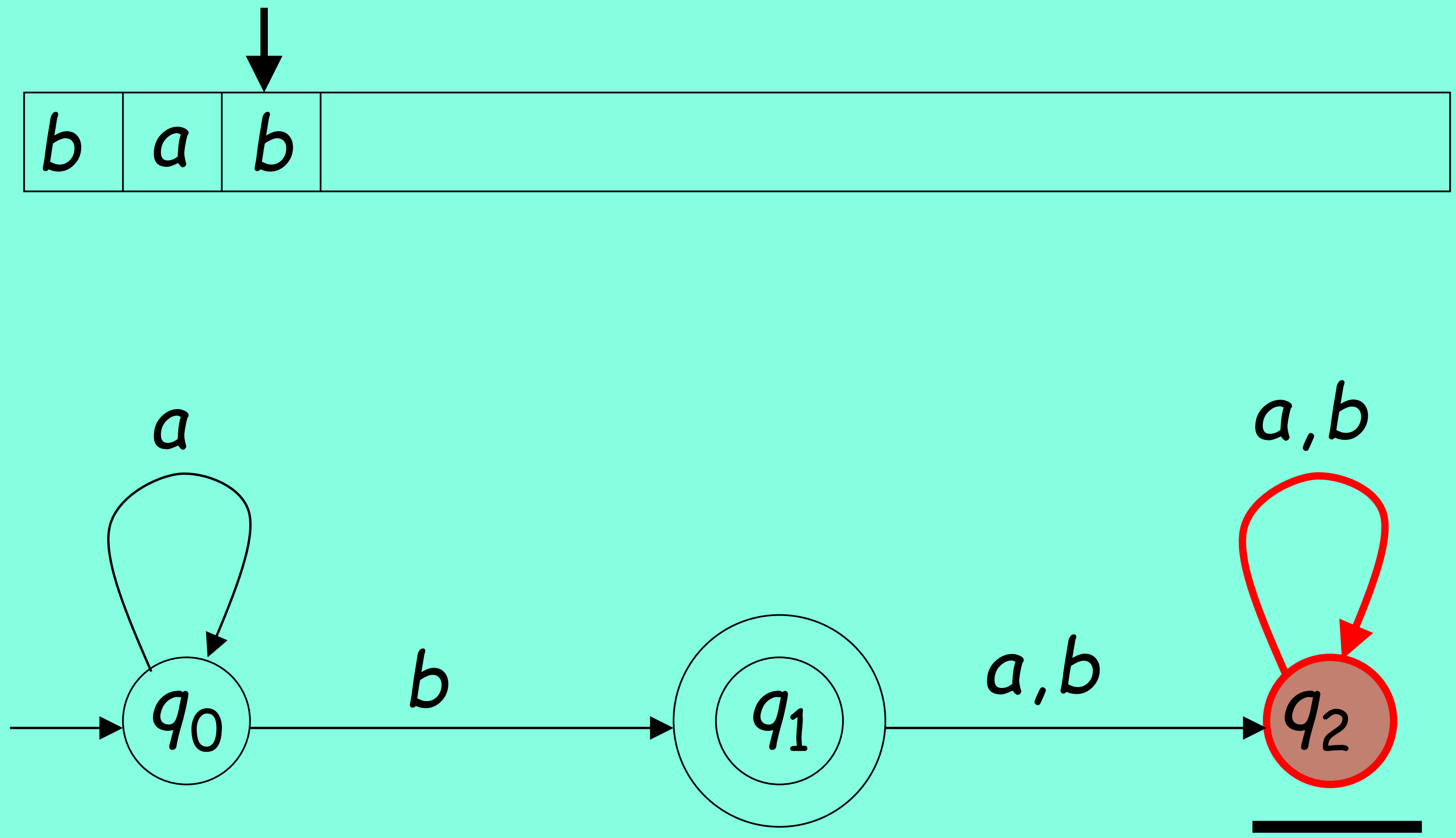

Input finished
a
a
b
a
Output: "accept"
a,b
b
a,b
q_0
q_1
q_2

Rejection



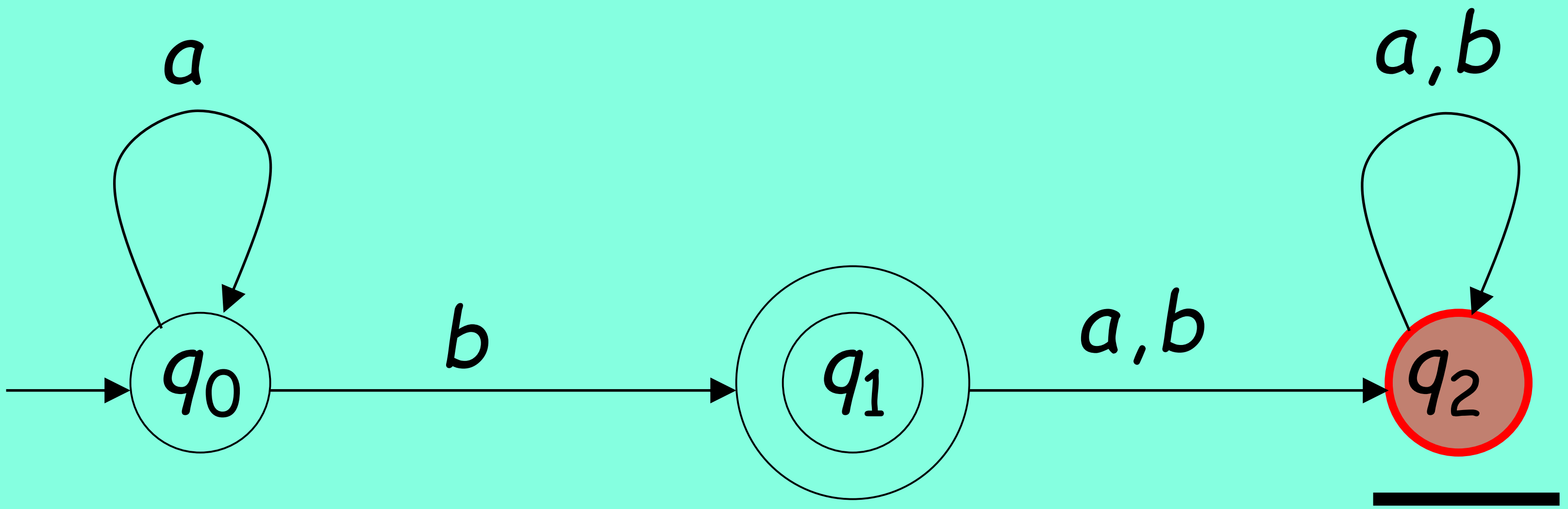

b
a
b
a
a, b
b
q_0
q_1
a, b
q_2


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


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

Input finished

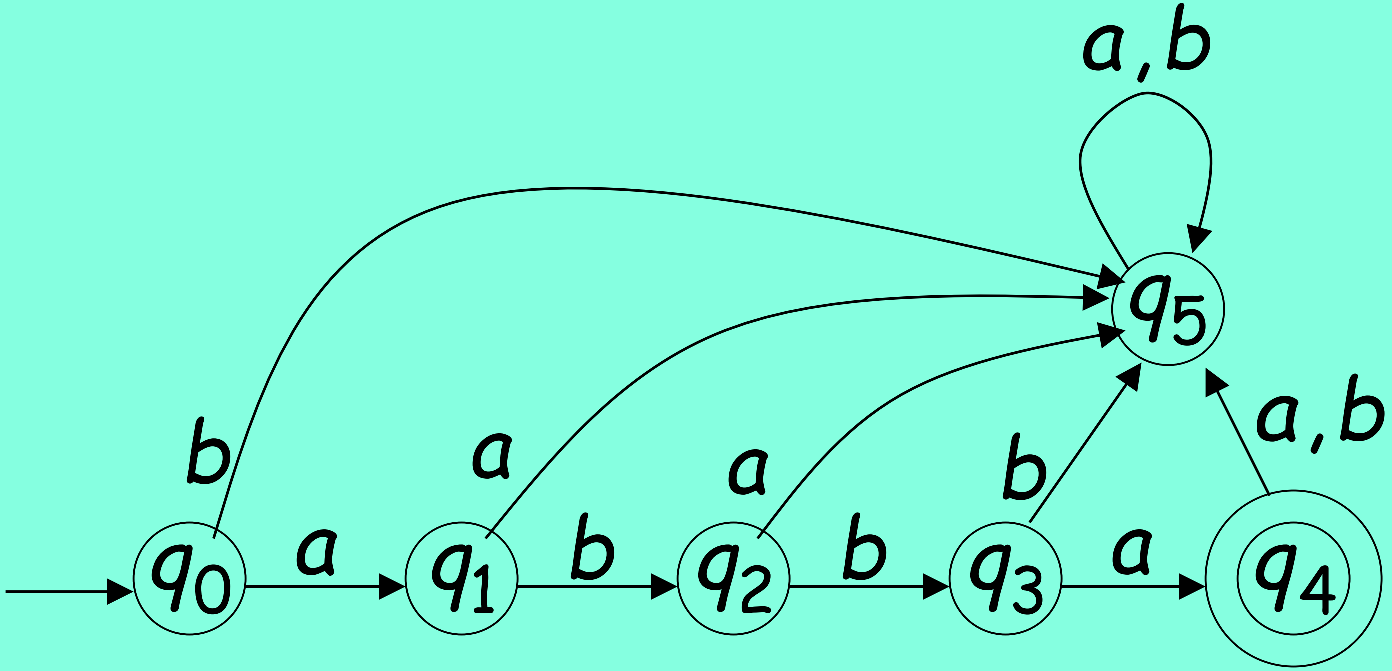
b	a	b	



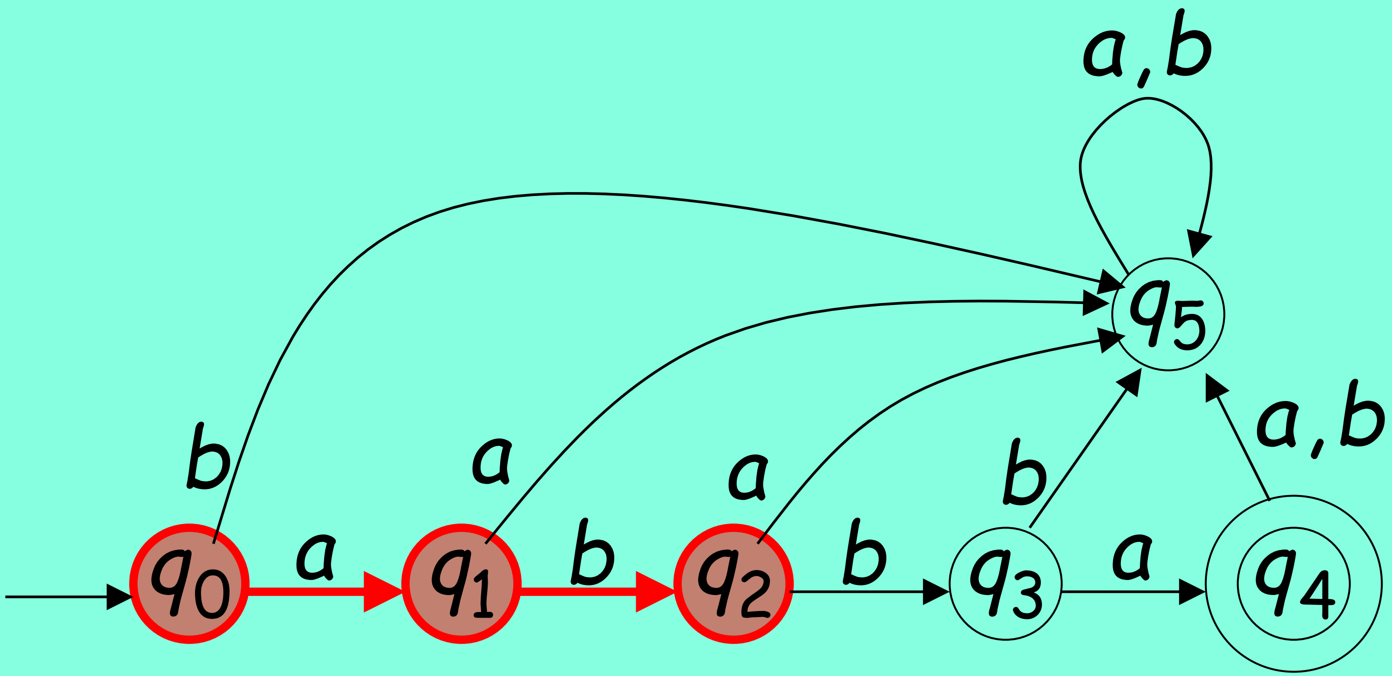
Output: "reject"

Extended Transition Function $\delta *$

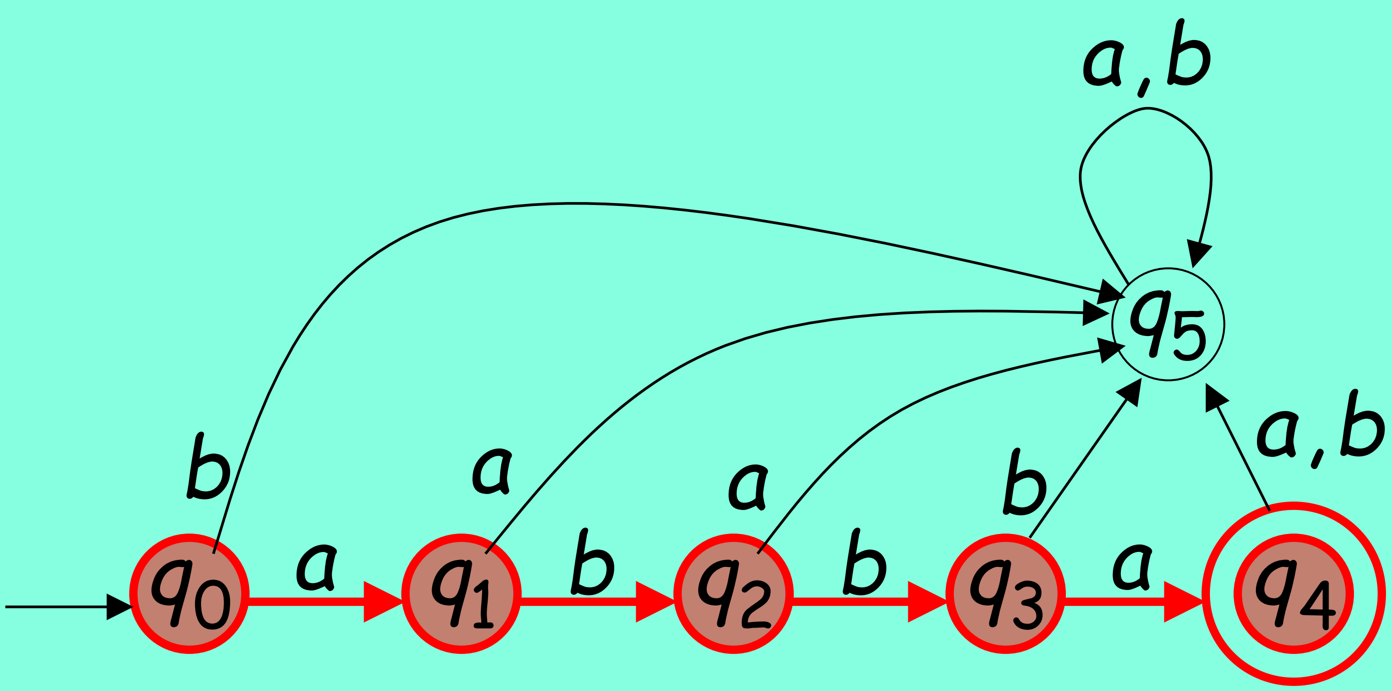
$$\delta * : Q \times \Sigma * \rightarrow Q$$



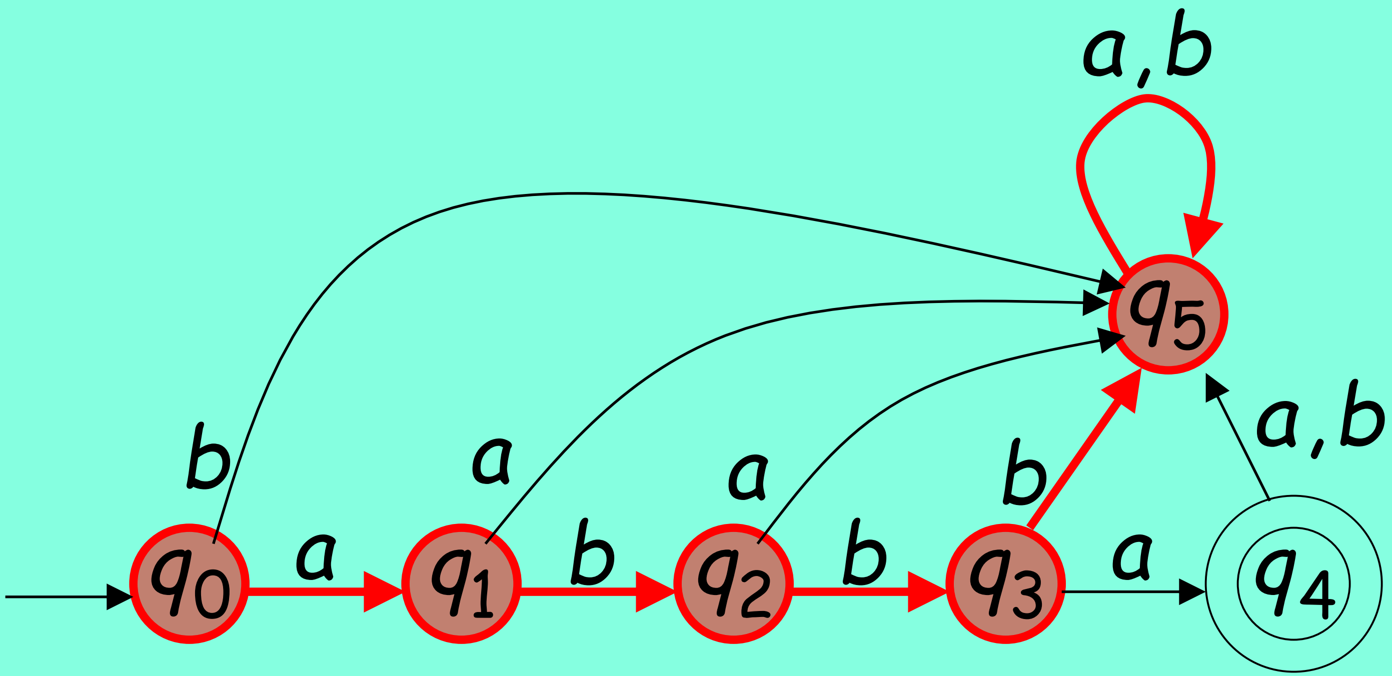
$$\delta^*(q_0, ab) = q_2$$



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



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

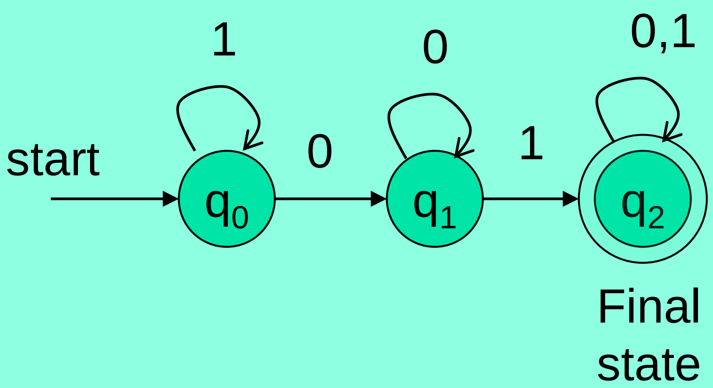


Example

- Build a DFA for the following language:
  - $L = \{w \mid w$ is a binary string that contains 01 as a substring$\}$
- Steps for building a DFA to recognize L:
  - $\Sigma = \{0,1\}$
  - Decide on the states: Q
  - Designate start state and final state(s)
  - δ: Decide on the transitions

DFA for strings containing 01

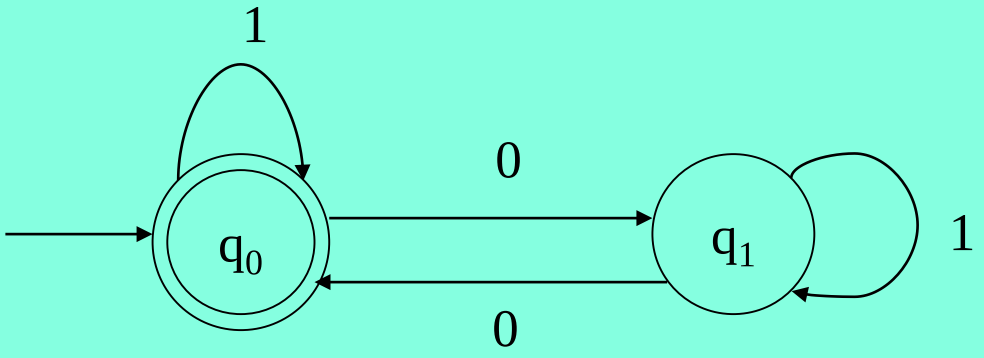
- What makes this DFA deterministic?



- $Q = \{q_0,q_1,q_2\}$
- $\sum = \{0,1\}$
- start state = q_0
- $F = \{q_2\}$
- Transition table

δ	**0**	**1**
→ **q_0**	q_1	q_0
q_1	q_1	q_2
***q_2**	q_2	q_2

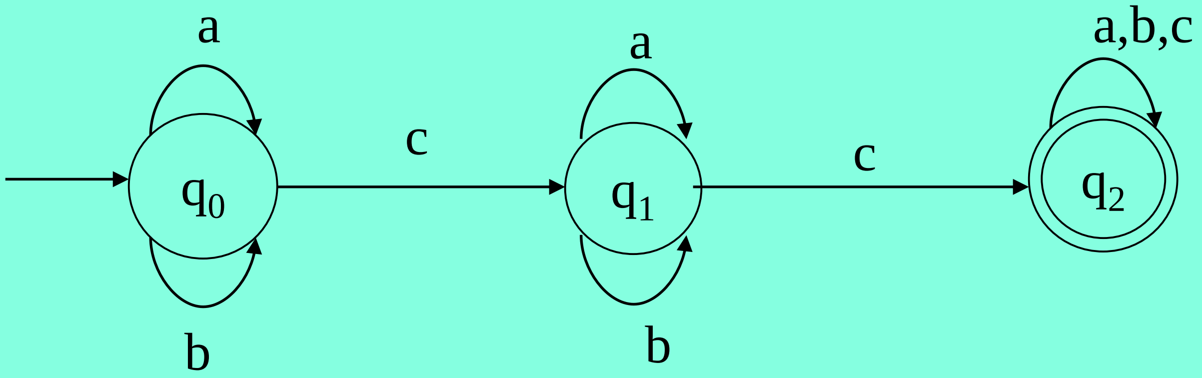
Example



	1		0		0		1		1	
q_0		q_0		q_1		q_0		q_0		q_0

The above DFA accepts those strings that contain an even number of 0’s

Example



	a		c		c		c		b		
q_0		q_0		q_1		q_2		q_2		q_2	<u>accepted</u>

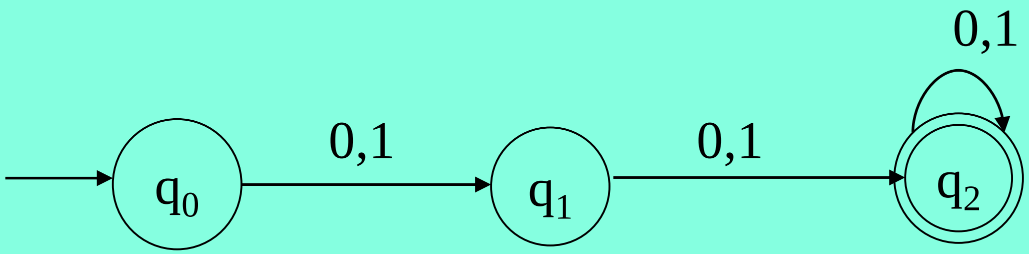
	a		a		c		
q_0		q_0		q_0		q_1	<u>rejected</u>

Accepts those strings that contain <u>at least</u> two c's

Example

Give a DFA M such that:

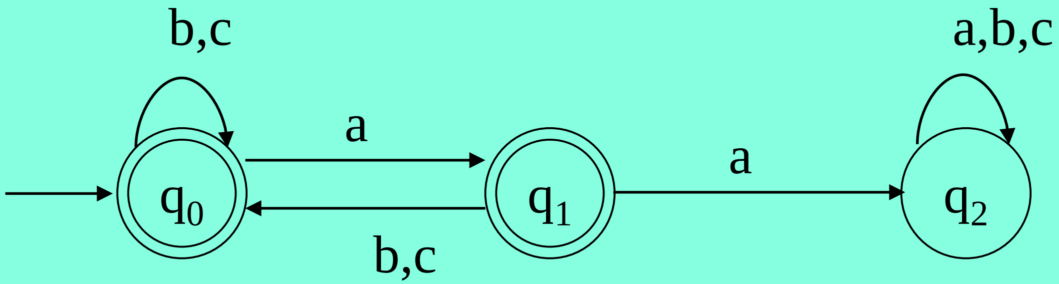
L(M) = {x | x is a string of 0’s and 1’s and |x| >= 2}



Example

Give a DFA M such that:

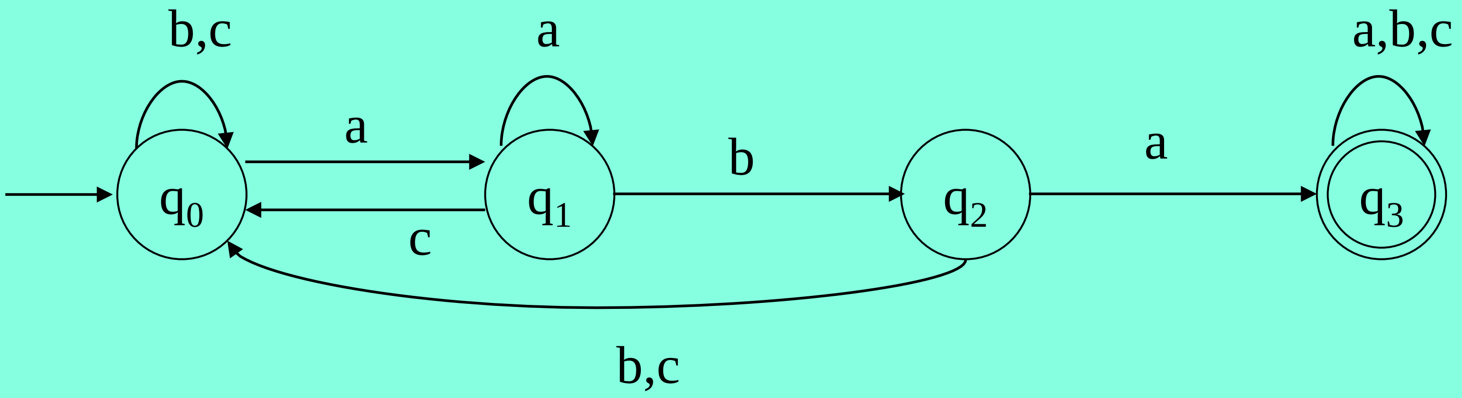
L(M) = {x | x is a string of (zero or more) a's, b's and c's such that x does not contain the substring *aa*}



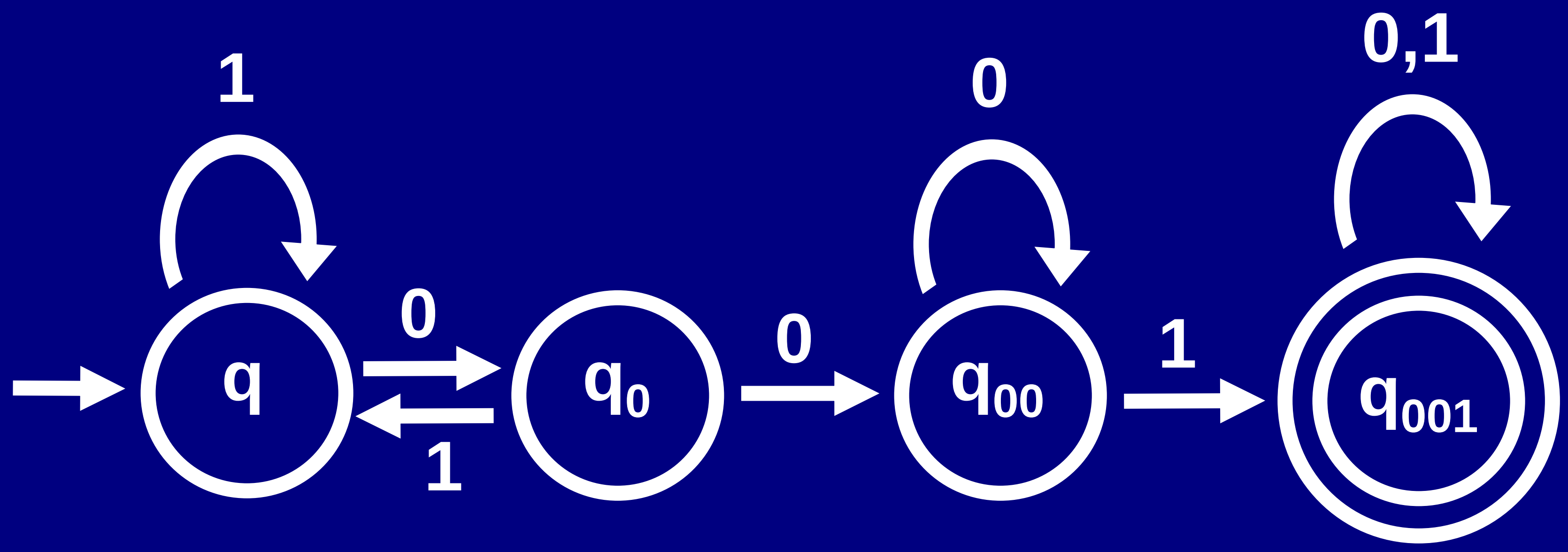
Example

Give a DFA M such that:

L(M) = {x | x is a string of a's, b's and c's such that x contains the substring *aba*}



Build an automaton that accepts all and only those strings that contain 001

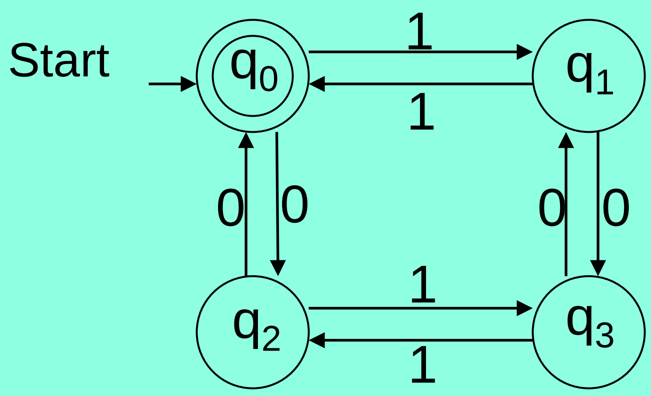


Example

- Build a DFA for the following language:
  L = { w | w is a binary string that has even number of 1s and even number of 0s}

States	Inputs	
	0	1
q_0	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

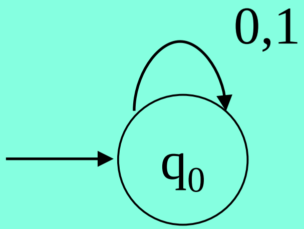
OR



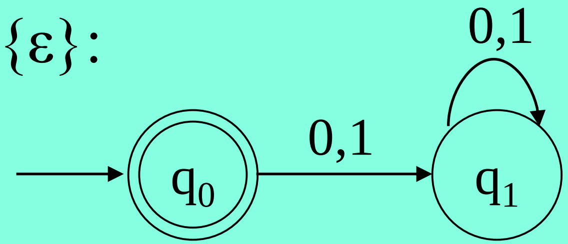
Example

Let $\Sigma = \{0,1\}$. Give DFAs for $\{\}$, $\{\varepsilon\}$, Σ^*, and Σ^+.

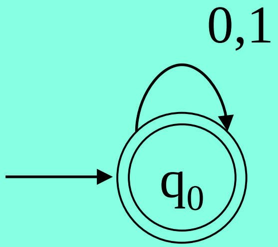
For $\{\}$:



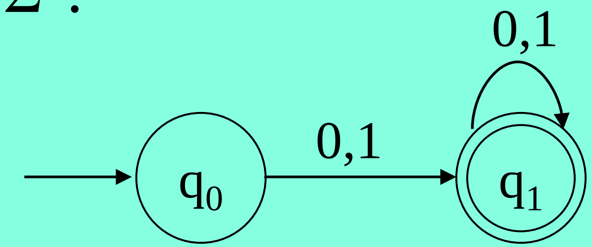
For $\{\varepsilon\}$:



For Σ^*:



For Σ^+:



Language of a DFA

- The "language" of a DFA is the set of all strings that the DFA accepts.
- **Definition**

  The language accepted by a DFA $M = (Q, \Sigma, \delta, q_0, F)$ is the set of all strings on Σ accepted by M.
- Language accepted by M:

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

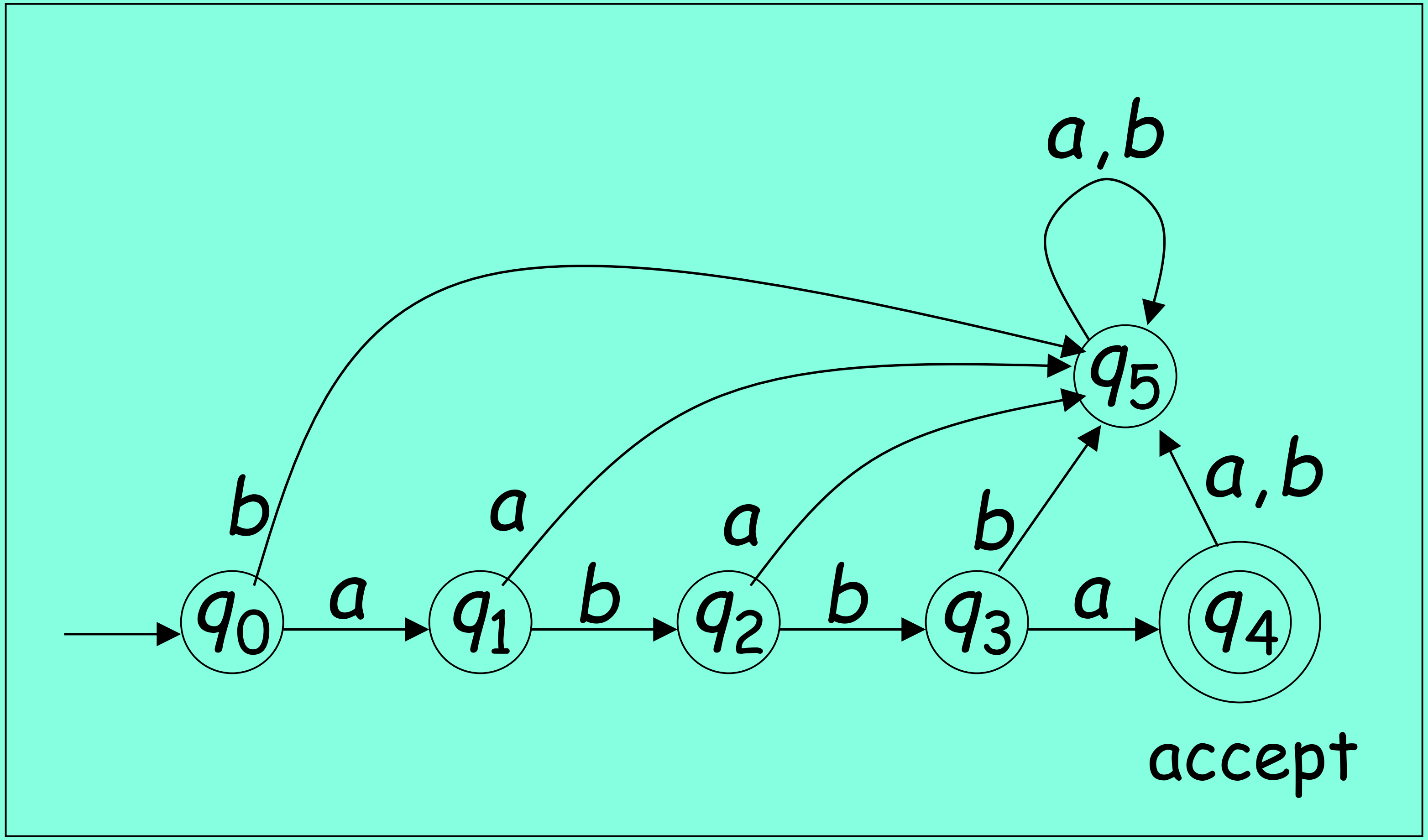
- Language rejected by M:

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

Example

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

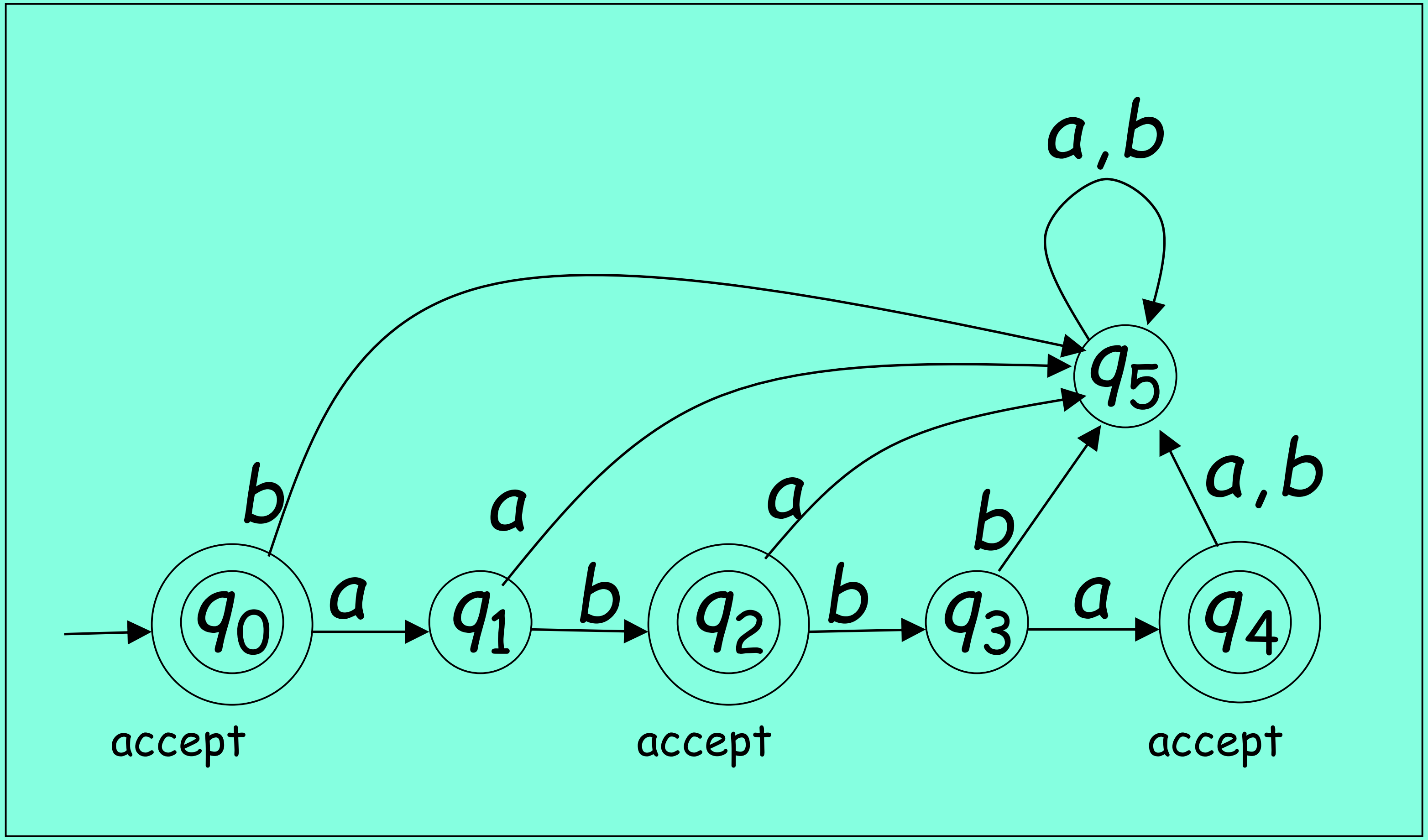
M



Another Example

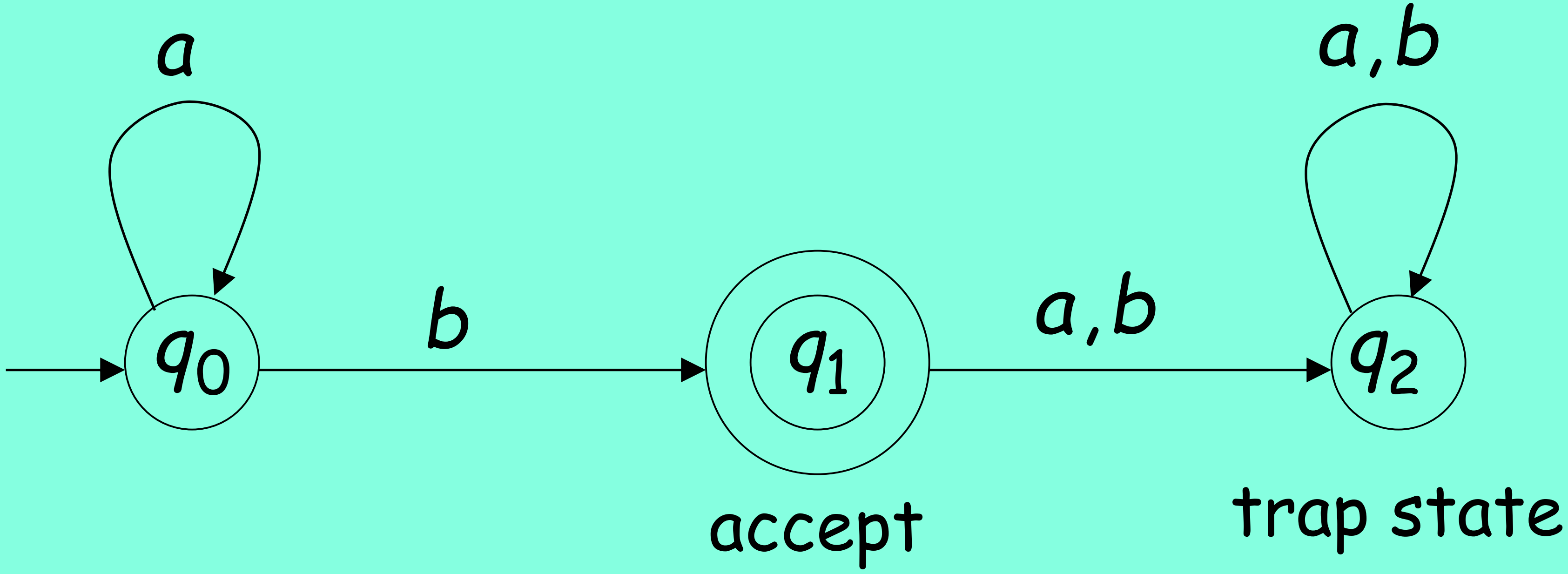
$$L(M) = \{\lambda, ab, abba\}$$

M

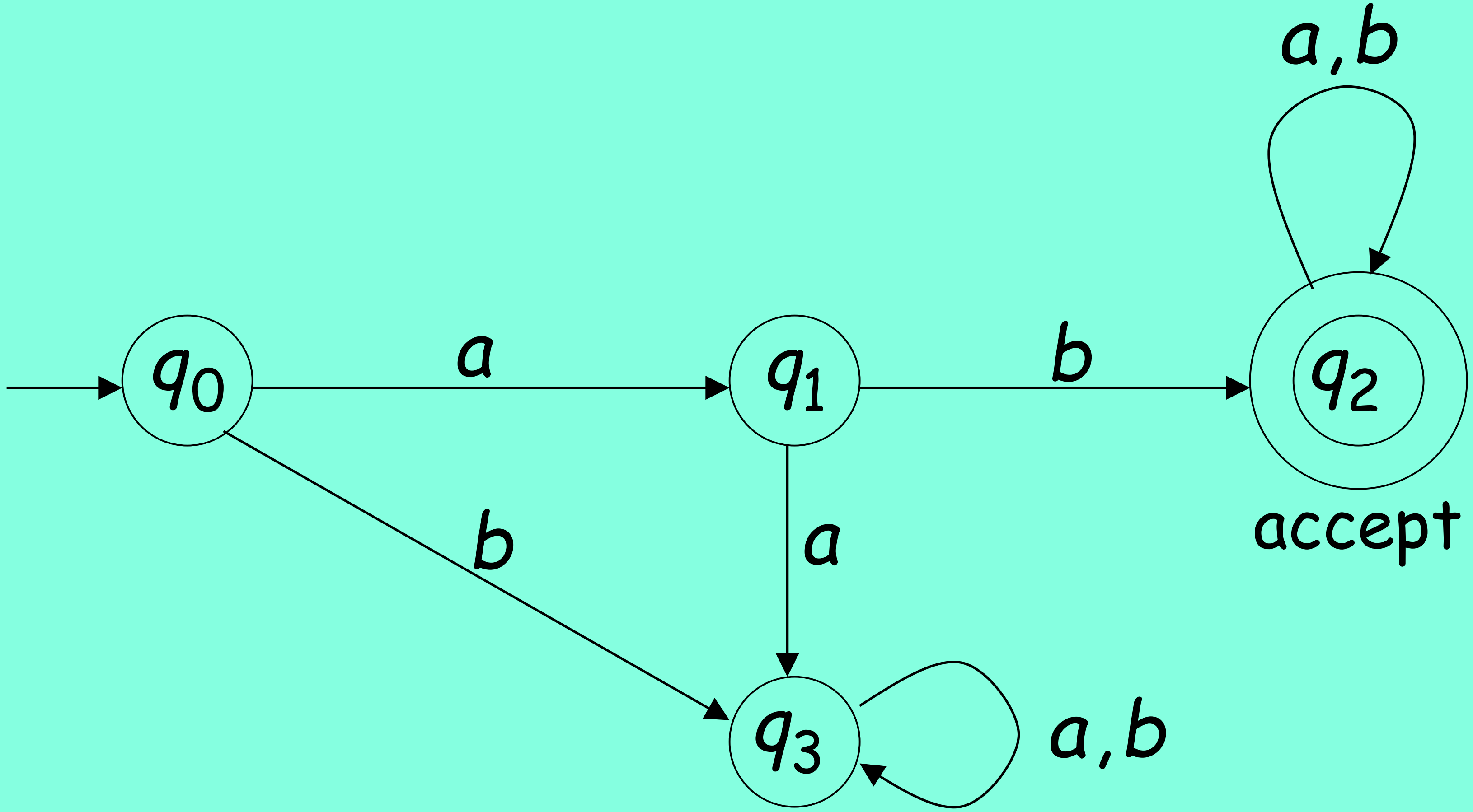


More Examples

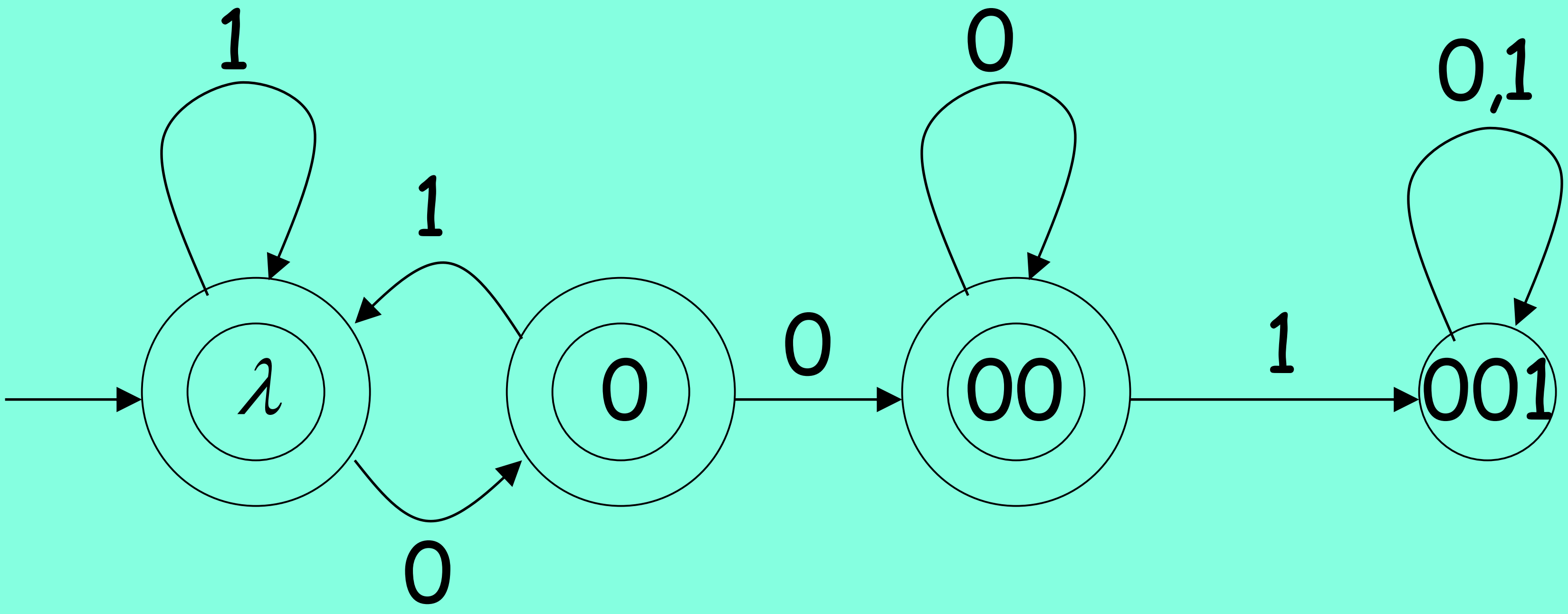
$$L(M) = \{a^n b : n \geq 0\}$$



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



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



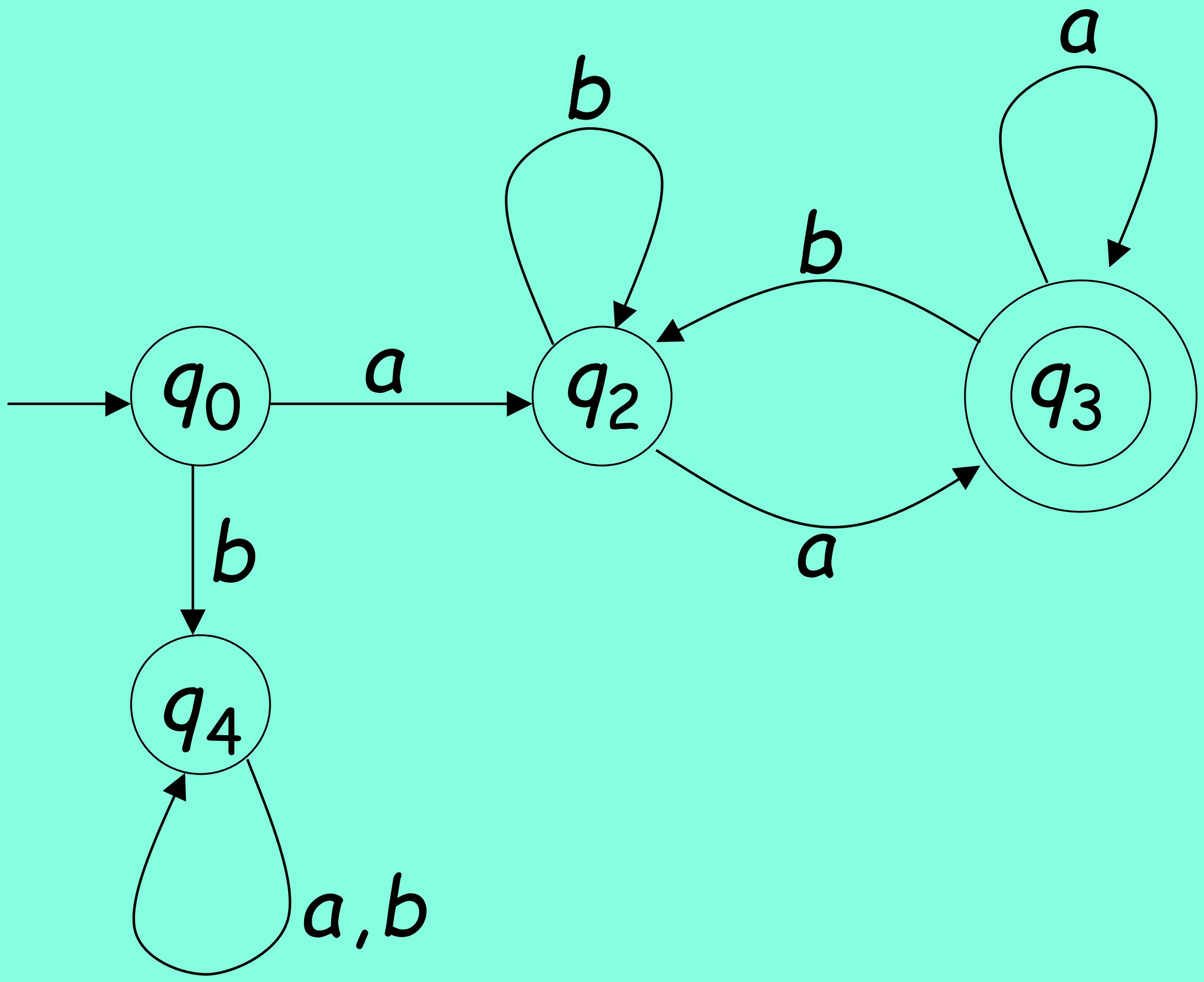
Regular Language

A language L is called **regular** if and only if there exists some DFA M such that

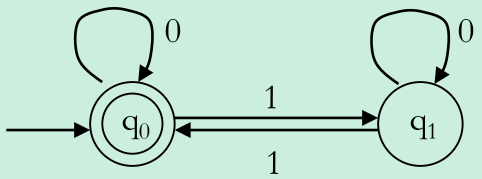
$$L=L(M)$$

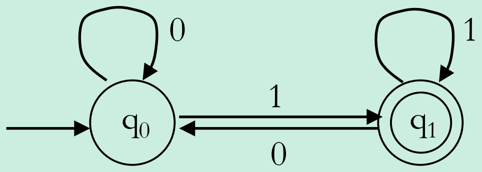
Example

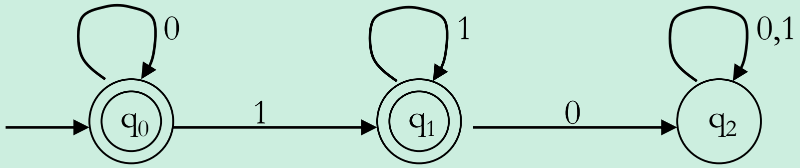
The language $L = \{awa : w \in \{a,b\}^*\}$ is regular:



Examples







What are the languages of these DFAs?