Formal Languages
Non-Deterministic 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**

1 Introduction to the Theory of Computation

- 1.2 Three Basic Concepts
  - Languages
  - Grammars
  - Automata
- 1.3 Some Applications*

2 Finite Automata

- 2.1 Deterministic Finite Accepters
  - Deterministic Accepters and Transition Graphs
  - Languages and Dfa's
  - Regular Languages
- 2.2 Nondeterministic Finite Accepters
  - Definition of a Nondeterministic Accepter
  - Why Nondeterminism?
- 2.3 Equivalence of Deterministic and Nondeterministic Finite Accepters
- 2.4 Reduction of the Number of States in Finite Automata*

Definition of a Nondeterministic Accepter

→ Non-determinism means a choice of moves for an automaton

→ Rather than prescribing a unique move in each situation, it allows a set of possible moves.

A **nondeterministic finite accepter** or **nfa** is defined by the <u>quintuple</u>

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

where Q,Σ,q_0,F are defined as for deterministic finite accepters, but

$$\delta : Q \times (\Sigma \cup \{\lambda\}) \to 2^Q.$$

$Q \times \Sigma \to Q$ for DFA

In a nondeterministic accepter, the range of δ is in the powerset 2^Q, so that its value is not a single element of Q but a subset of it.

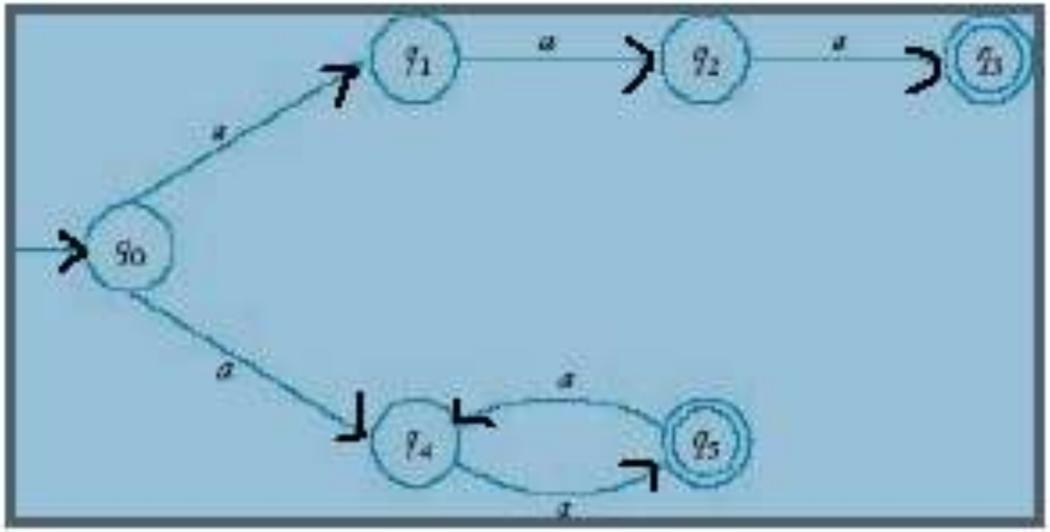
NFA contd.....

$\delta(q_1,a) = \{q_0,q_2\}$

either q0 or q2 could be the next state of the nfa.

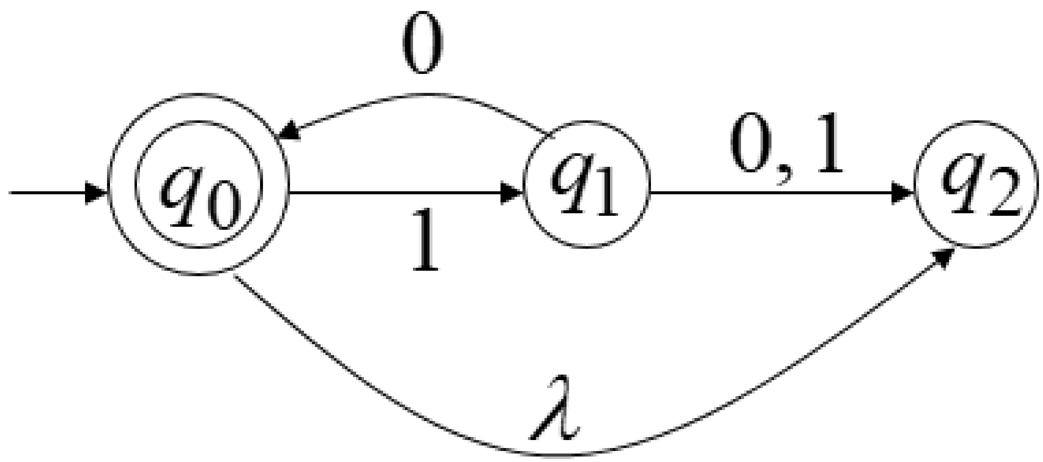
NFA can make a transition without consuming an input symbol also

A string is rejected (that is, not accepted) only if there is no possible sequence of moves by which a final state can be reached



It describes a nondeterministic accepter since there are two transitions labeled a out of q_0.

What is the language accepted by this NFA?



It is nondeterministic because it has a λ-transition……

Here $\delta(q_2, 0) = \varnothing$

The automaton accepts strings λ, 1010, and 101010, but not 110 ,10100, 101…

What about 10???

Note that for 10 there are two alternative walks, one leading to q_0, the other to q_2. Even though q_2 is not a final state, the string is accepted because one walk leads to a final state

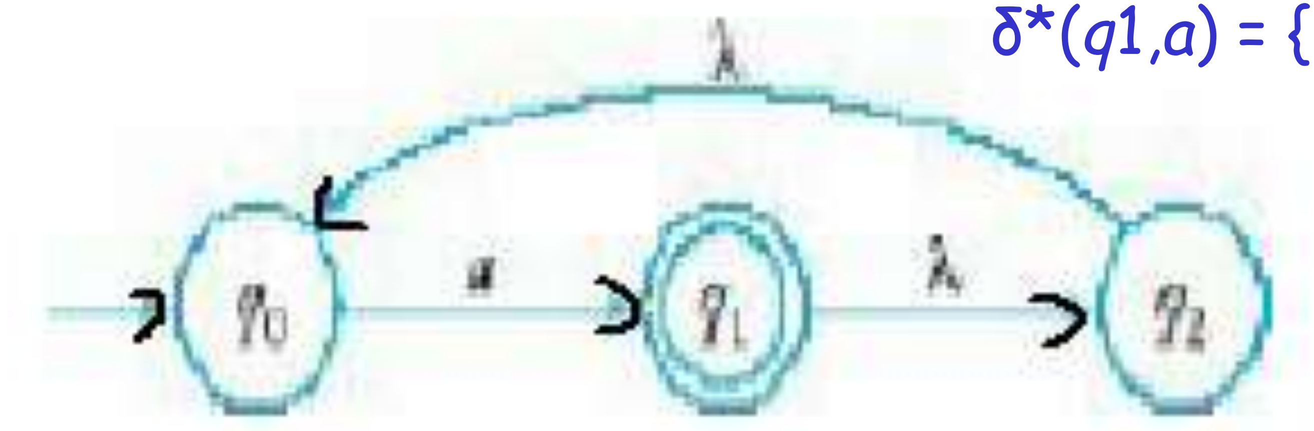
Extended Transition Function....

$$\delta^*(qi, w) = Qj$$

Qj is the set of all possible states the automaton may be in, having started in state *qi* and having read w.

For an nfa, the extended transition function is defined so that δ^* (qi,w) contains *qj* if and only if there is a walk in the transition graph from qi to *qj* labeled w.
This holds for all *qi*, *qj* $\in Q$, and $w \in \Sigma^*$.

Now consider this NFA…



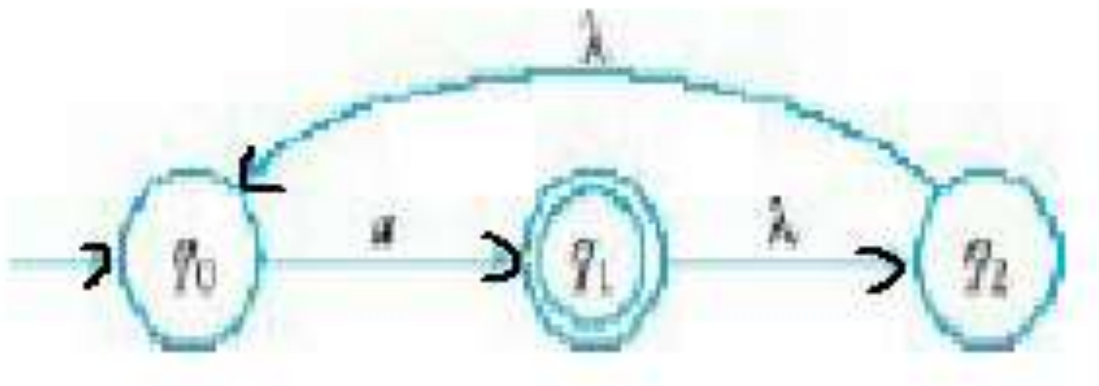
$\delta^*(q1,a) = \{q0,q1,q2\}$.

Suppose we want to find $\delta^*(q1,a)$ and $\delta^*(q2,\lambda)$.

→ There is a walk labeled <u>a</u> involving two λ- transitions from $q1$ to itself.

→By using some of the λ-edges twice, we see that there are also walks involving λ-transitions to $q0$ and $q2$.

→ $\delta^*(q2,\lambda)$also contains $q2$



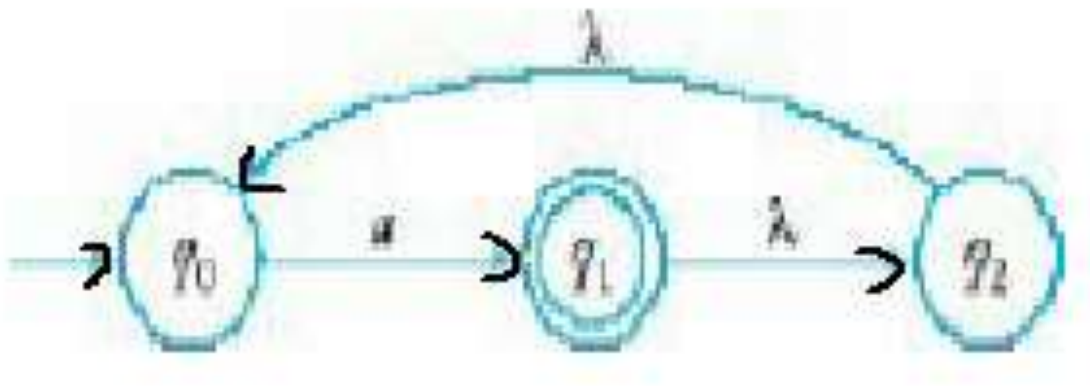
$$\delta^*(q2, \lambda) = \{q0, q2\}$$

$$\delta^*(q2, aa) = \{q1, q2, q0\}$$

The language L accepted by an nfa $M = (Q,\Sigma,\delta, q_0,F)$ is defined as the set of all strings accepted in the above sense. Formally

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

$$\delta^*(q0, aa) = \{q0, q1, q2\}$$



$\delta^*(q0, a) = \{q1, q2, q0\}$

$\delta^*(q0, aa) = \{q1, q2, q0\}$

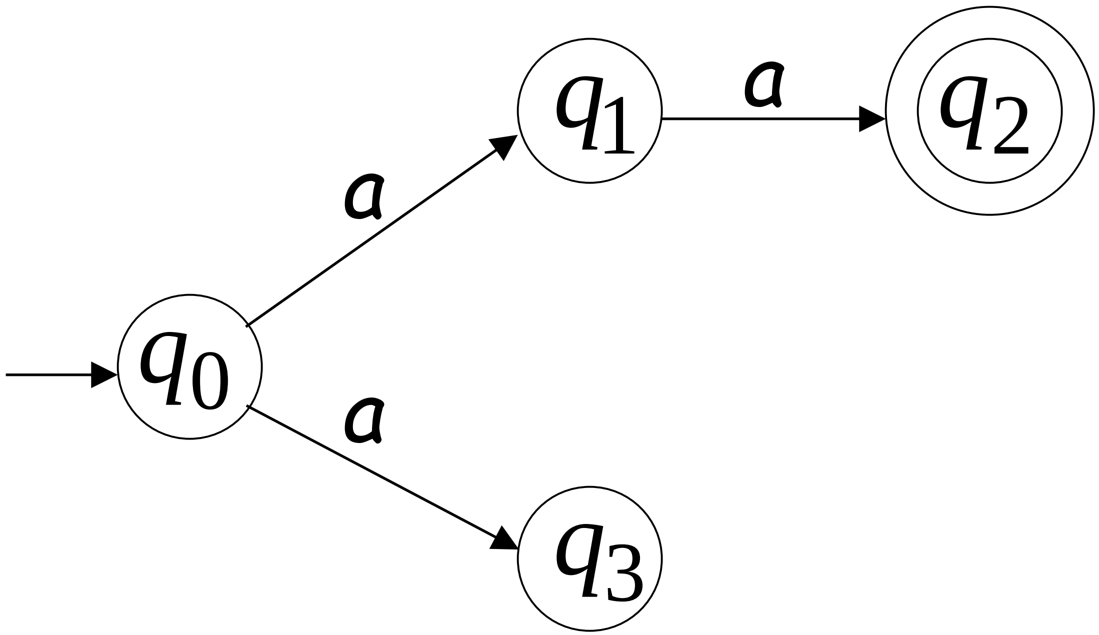
$\delta^*(q2, a) = \{q0, q1, q2\}$

$\delta^*(q1, a) = \{q0, q1, q2\}$

$\delta^*(q1, \lambda) = \{q1, q2, q0\}$

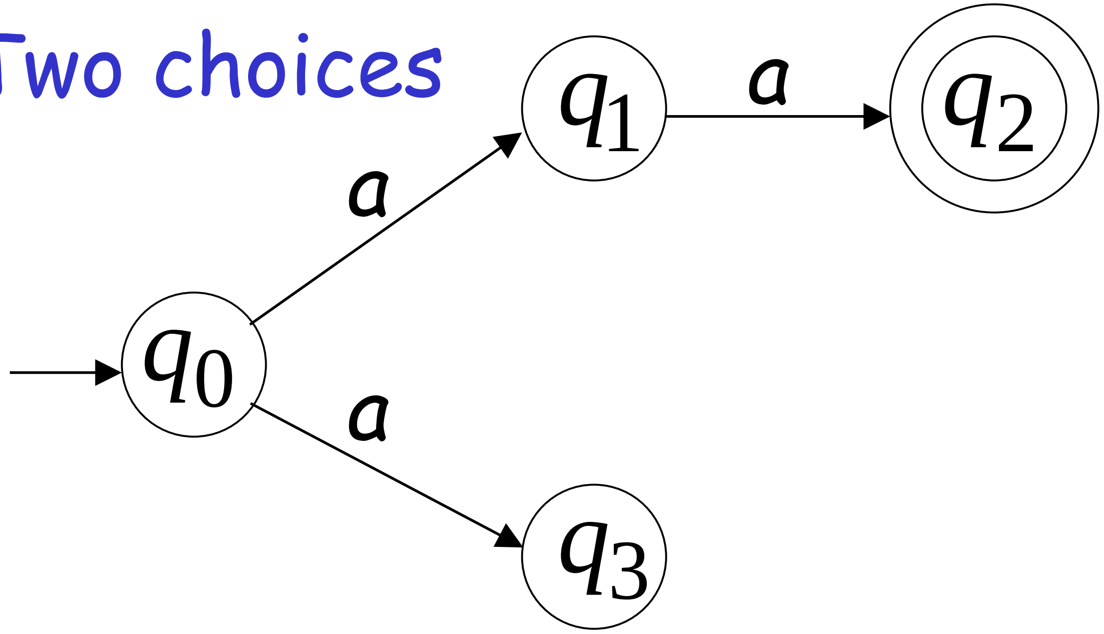
Nondeterministic Finite Automaton (NFA)

Alphabet = $\{a\}$

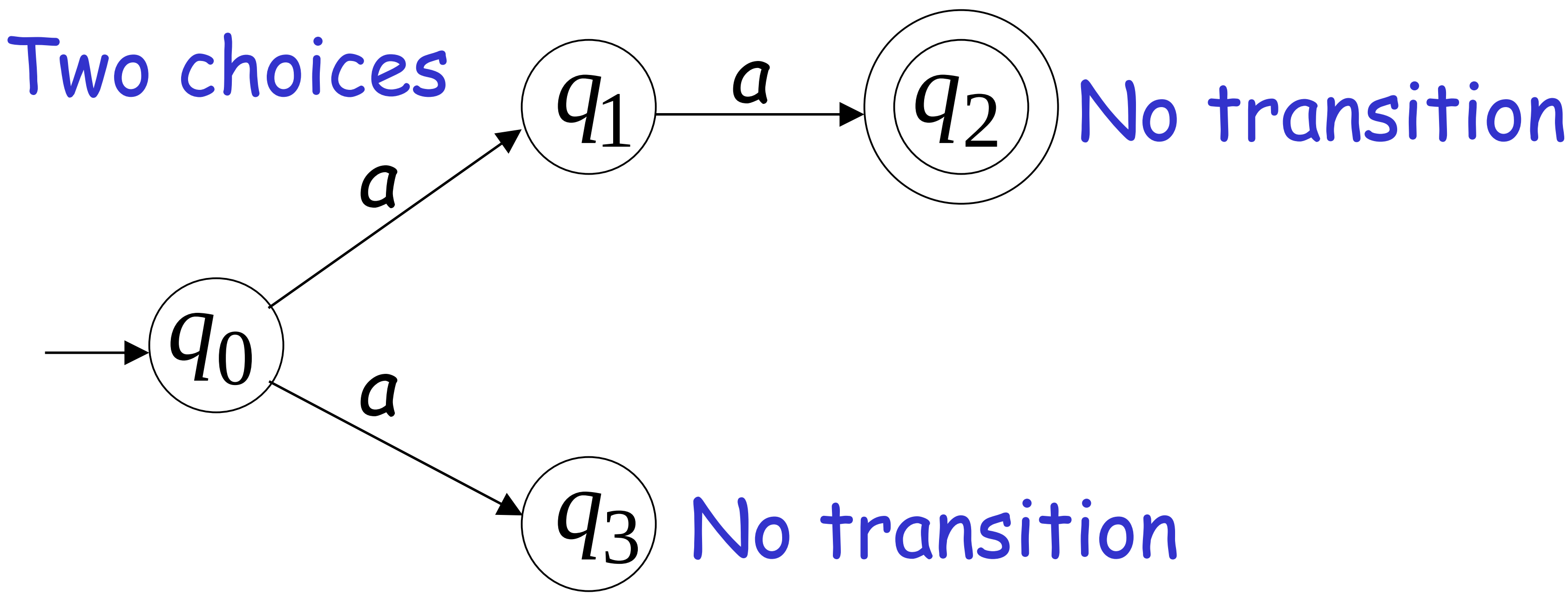


Alphabet = $\{a\}$

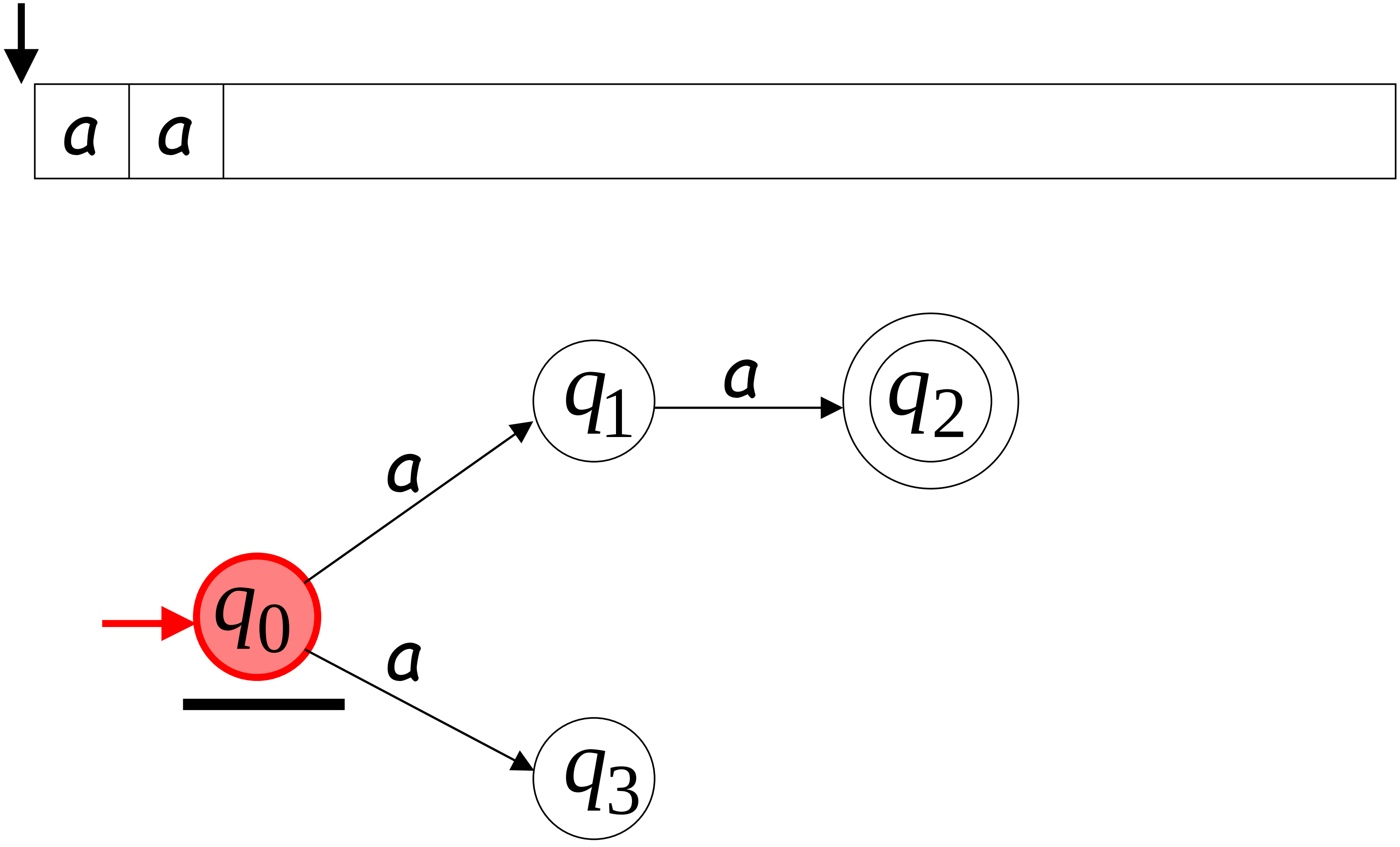
Two choices



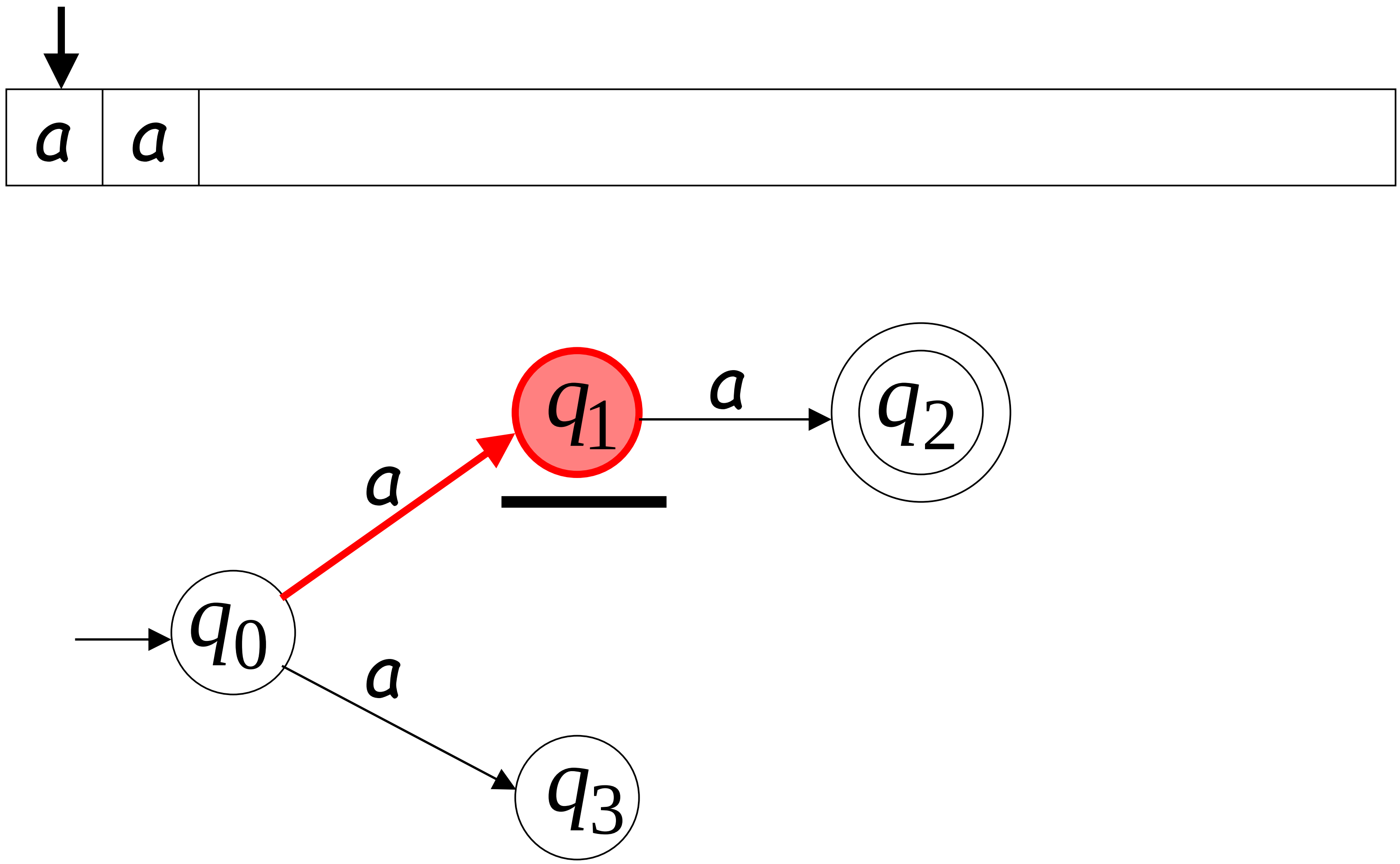
Alphabet = $\{a\}$



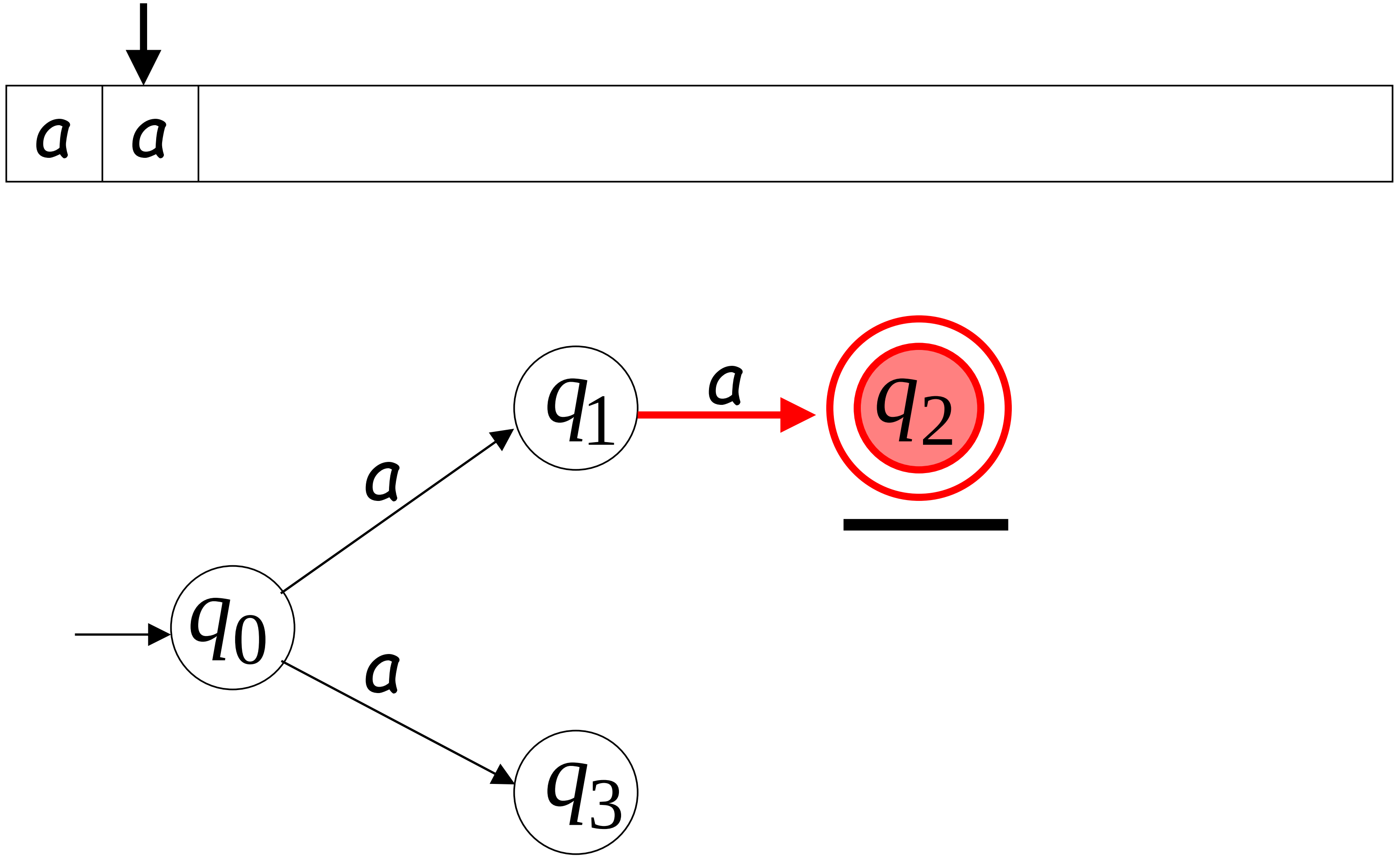
First Choice



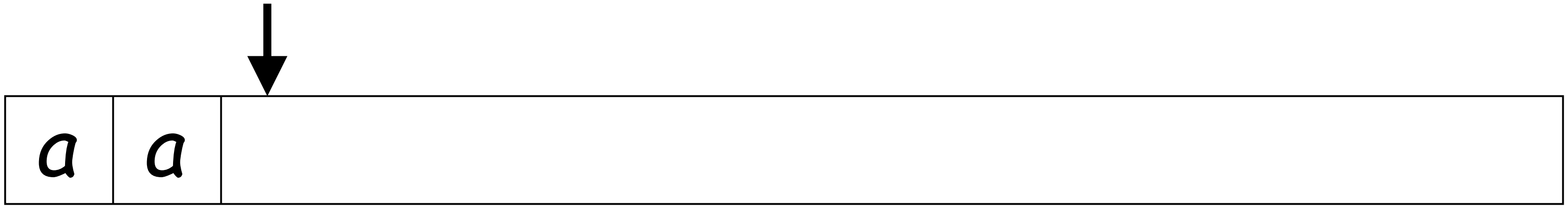
First Choice



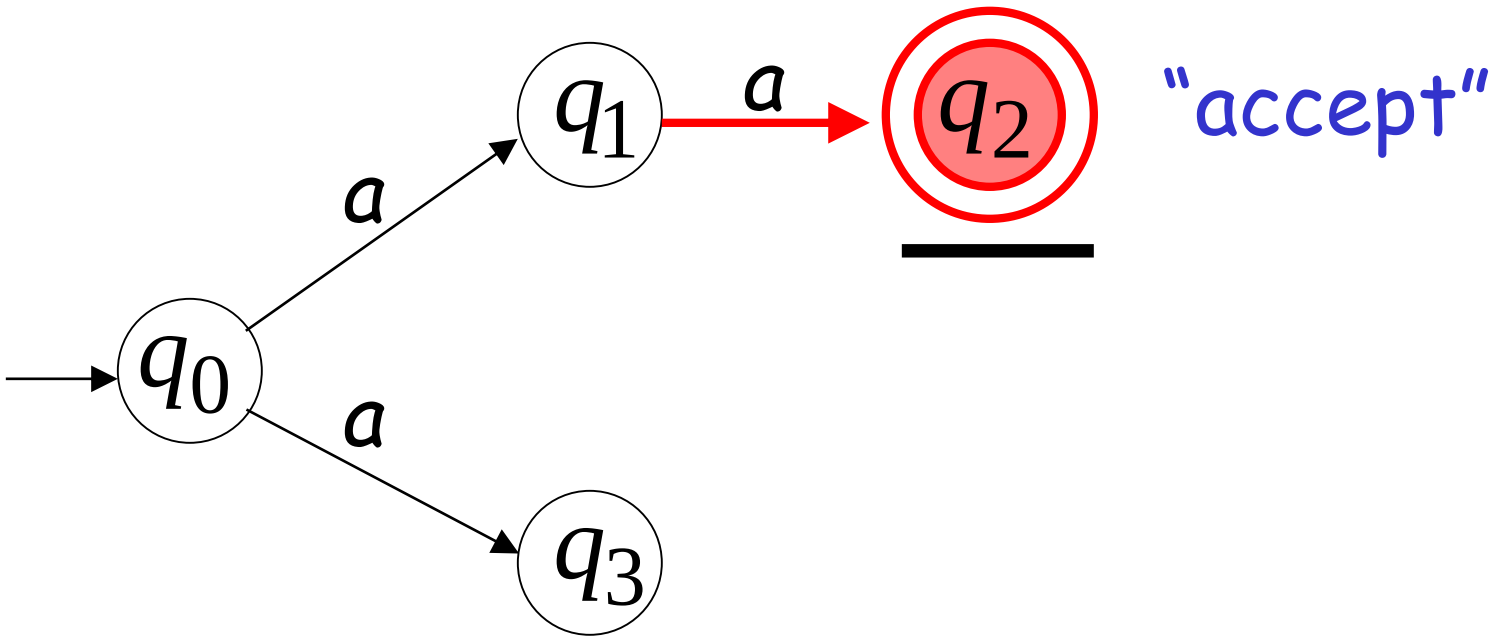
First Choice



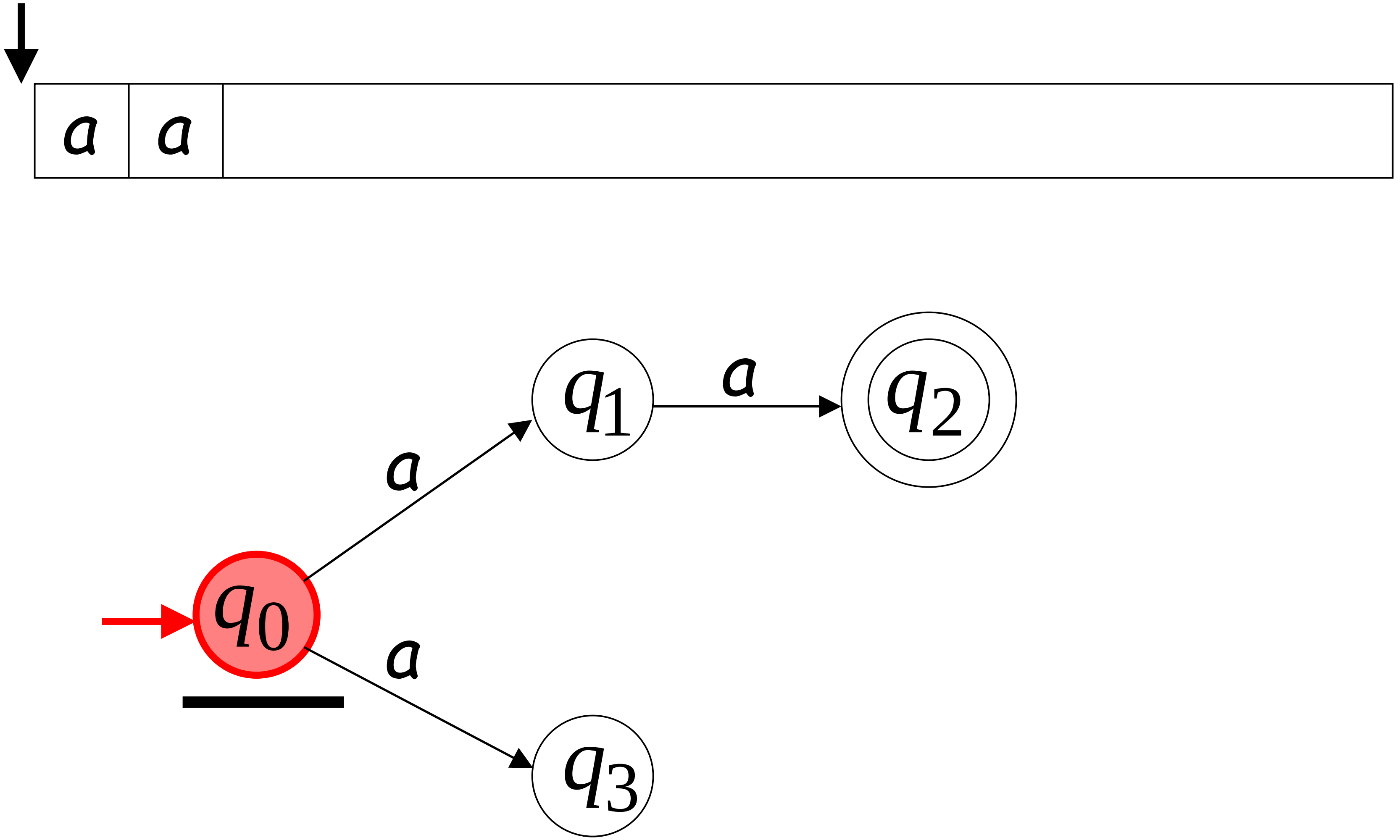
First Choice



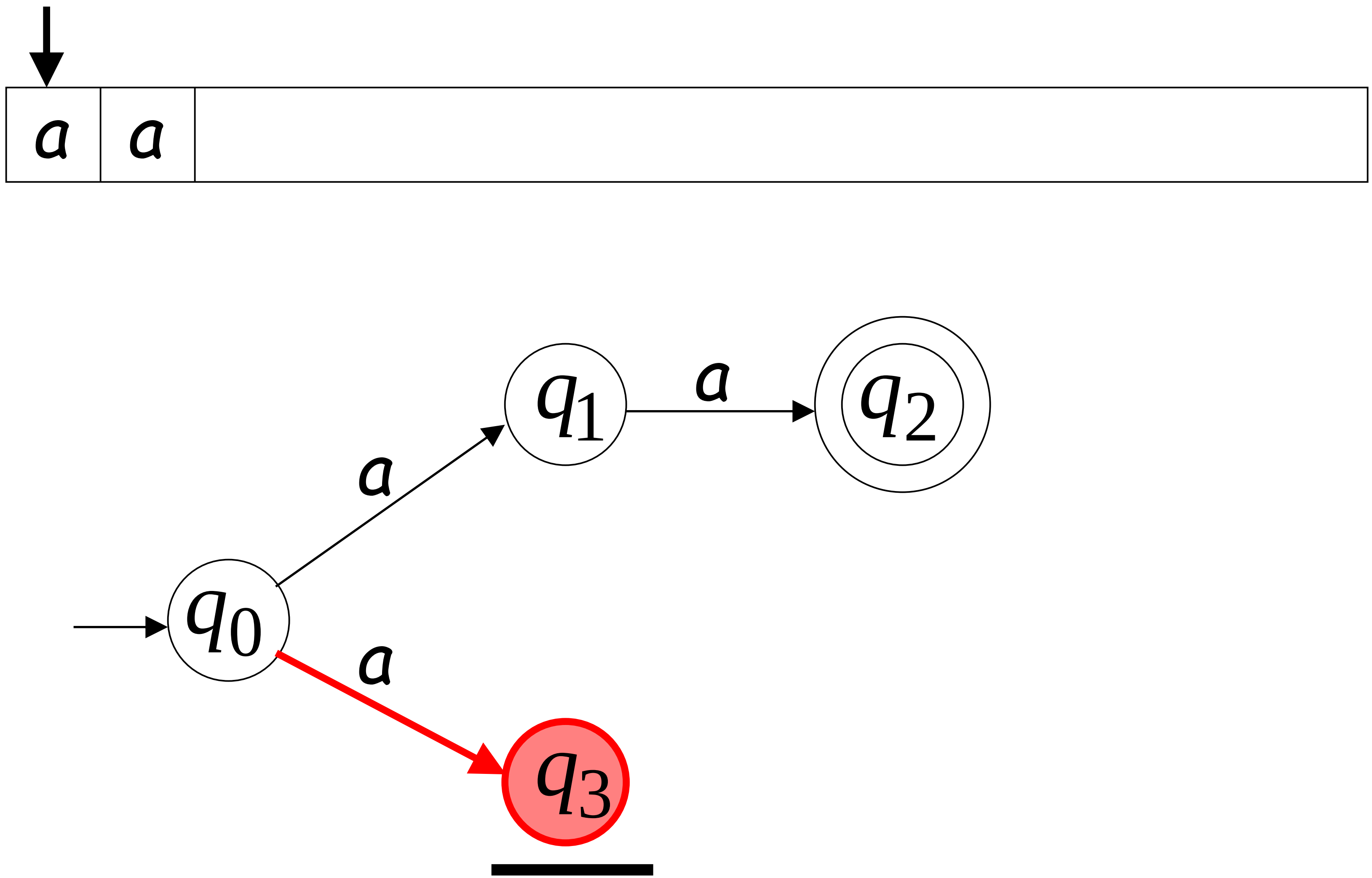
All input is consumed



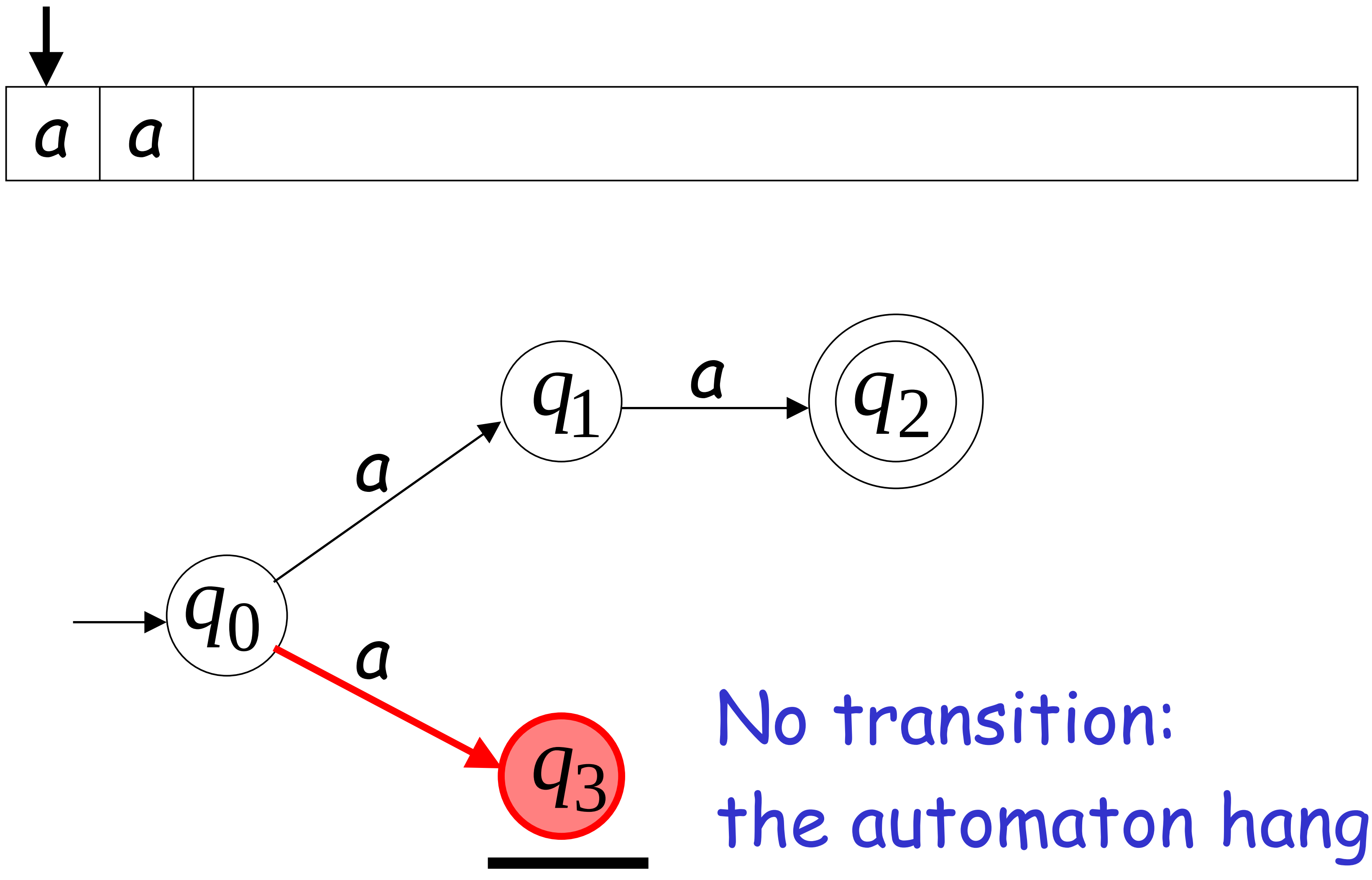
Second Choice



Second Choice

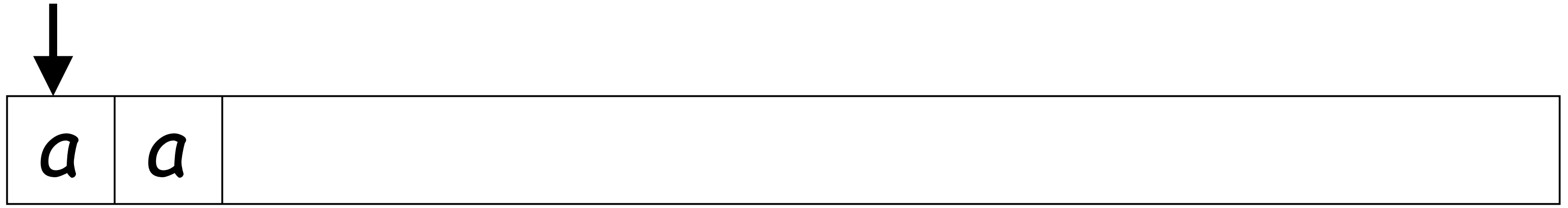


Second Choice

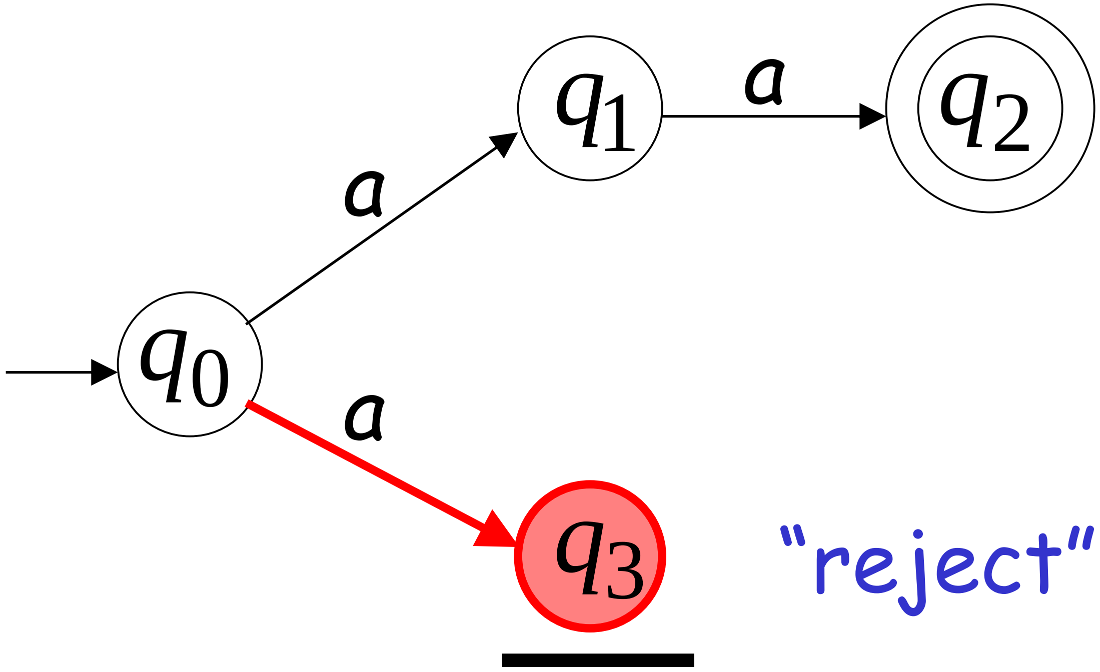


No transition:
the automaton hangs

Second Choice



Input cannot be consumed



"reject"

An NFA accepts a string:

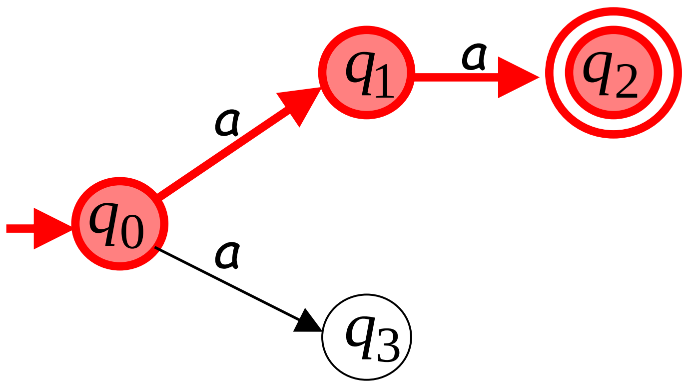
when there is a computation of the NFA
that accepts the string

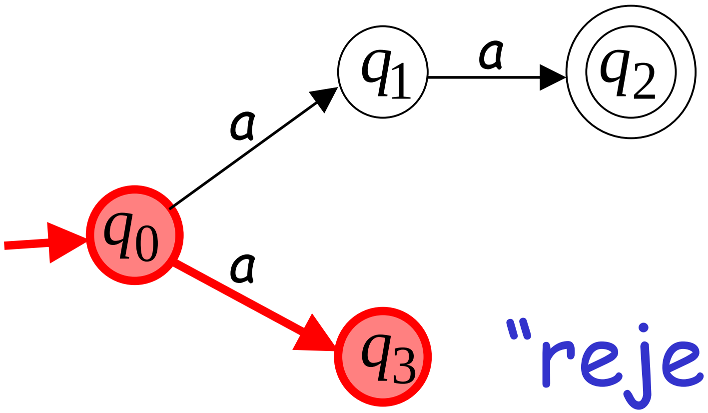
There is a computation: means....
all the input is consumed and the automaton
is in an accepting state

Example

aa is accepted by the NFA:

“accept”

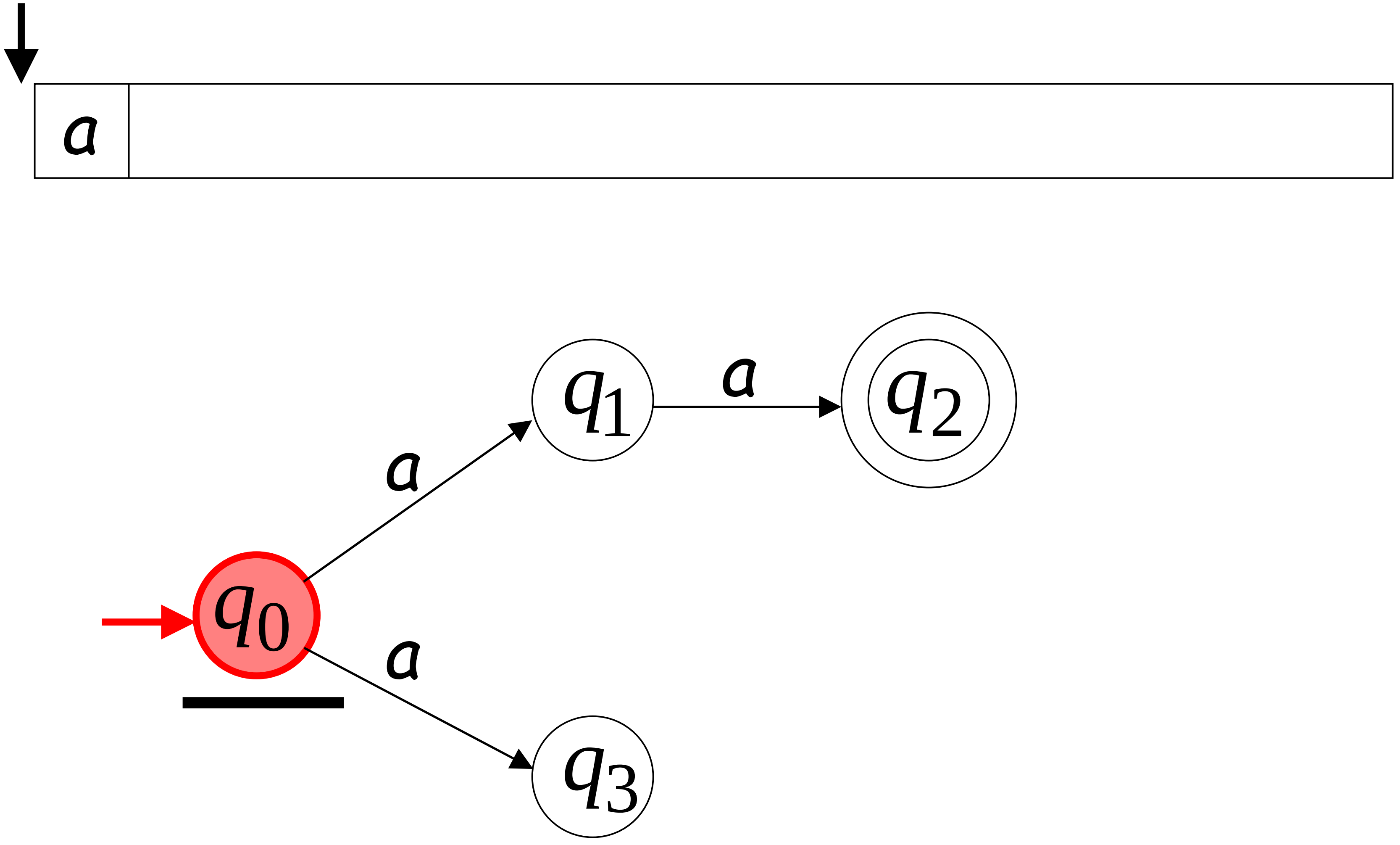




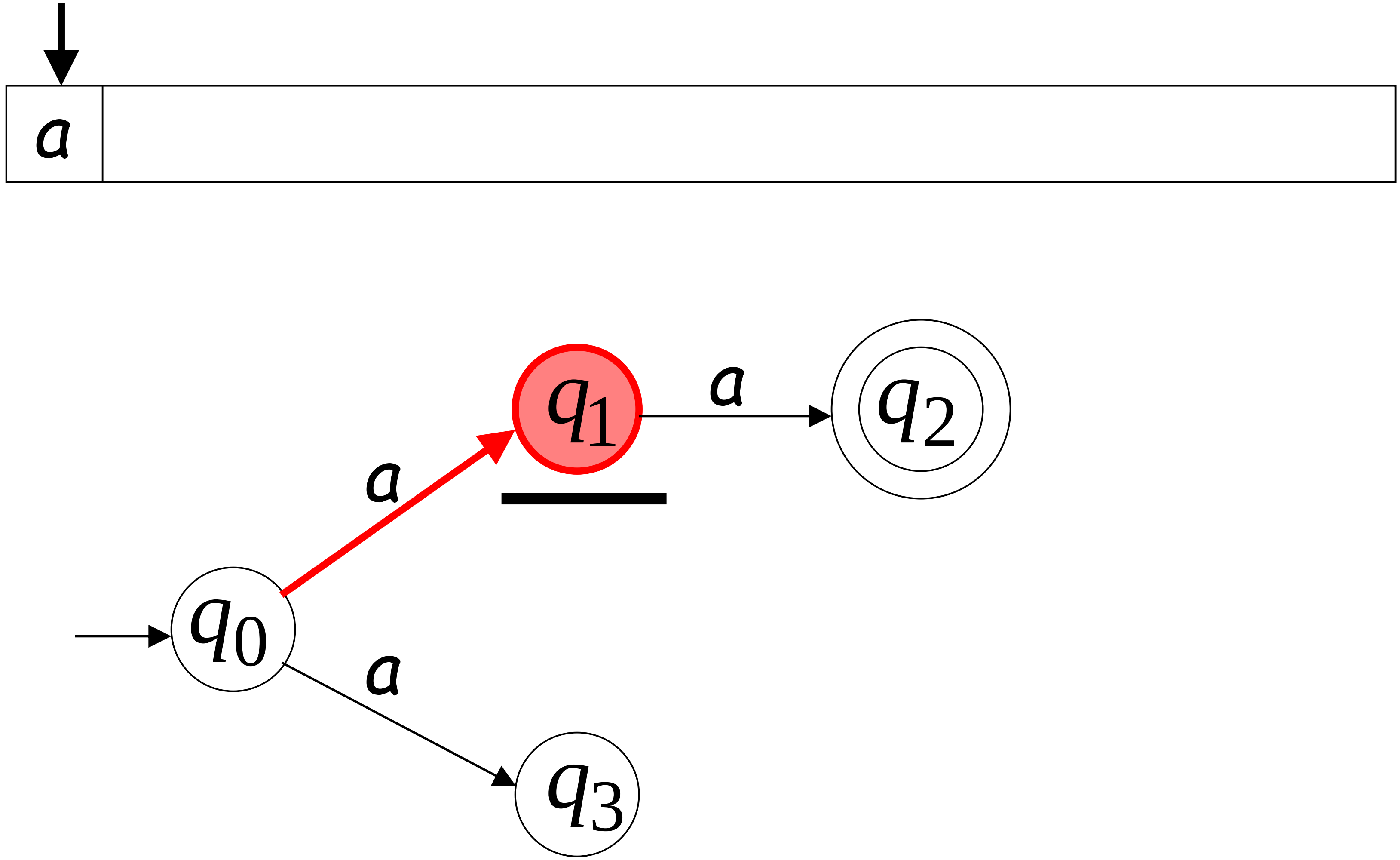
“reject”

because this computation accepts aa

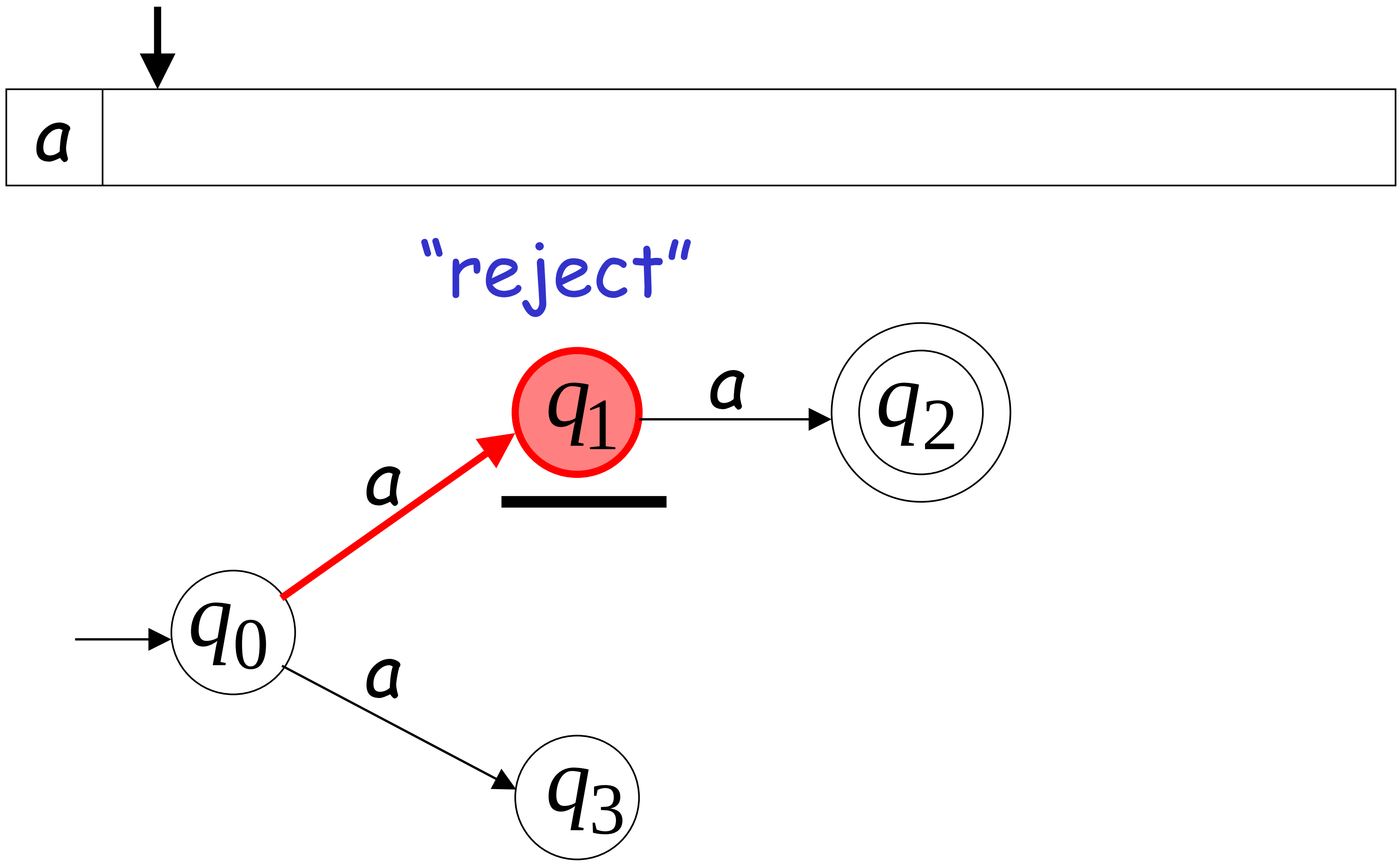
Rejection example



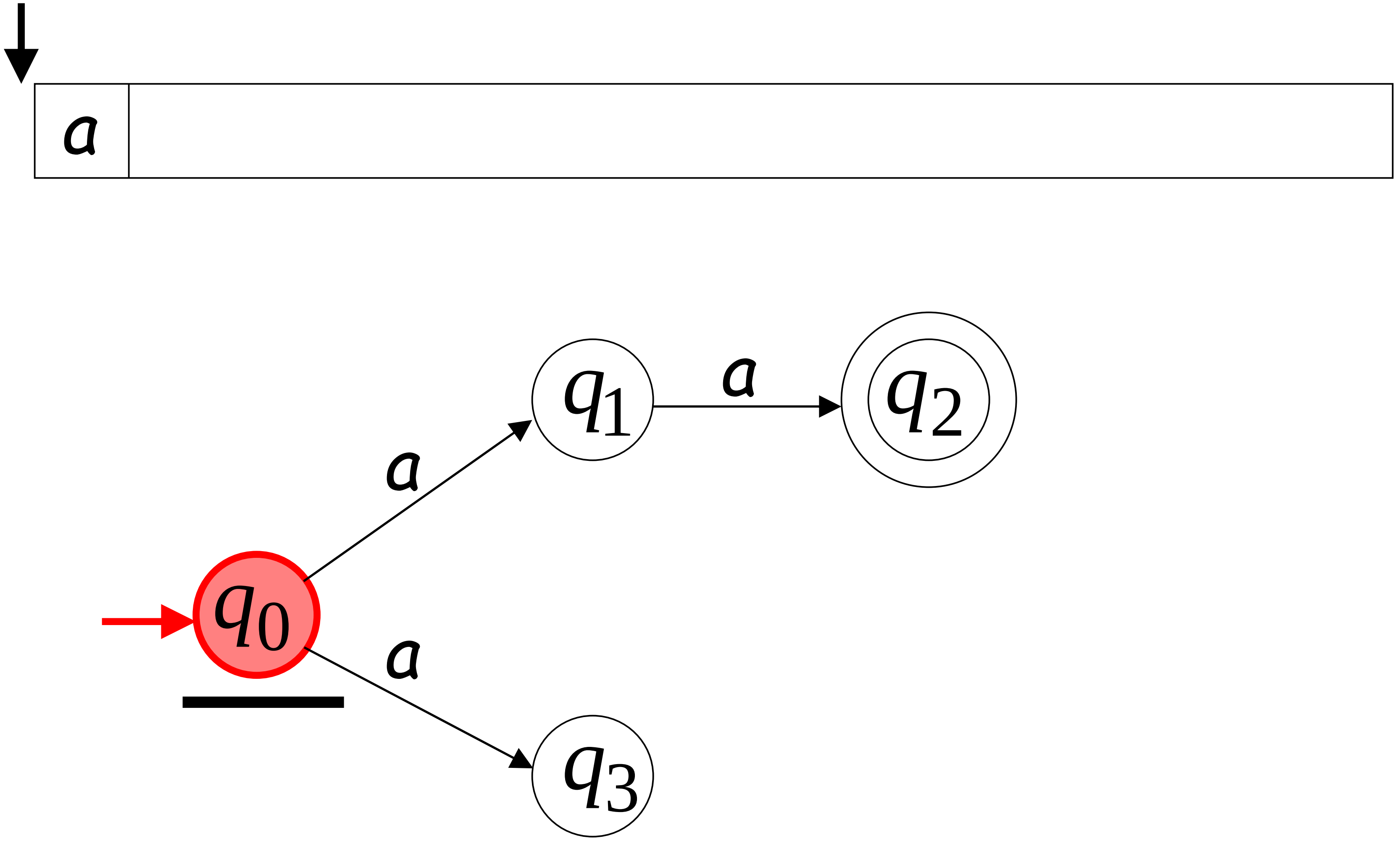
First Choice



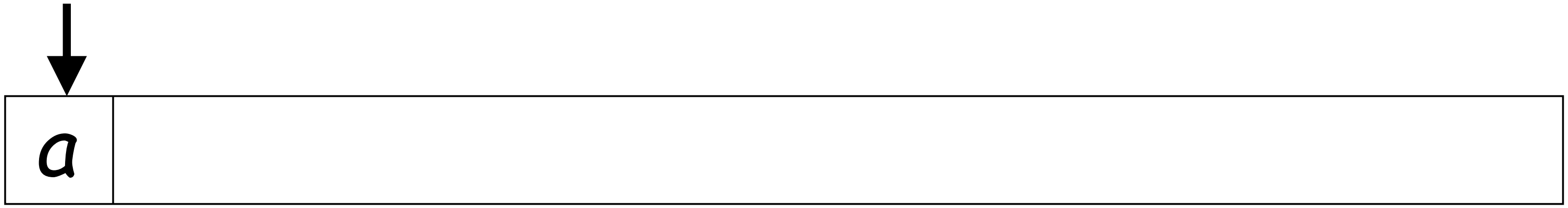
First Choice

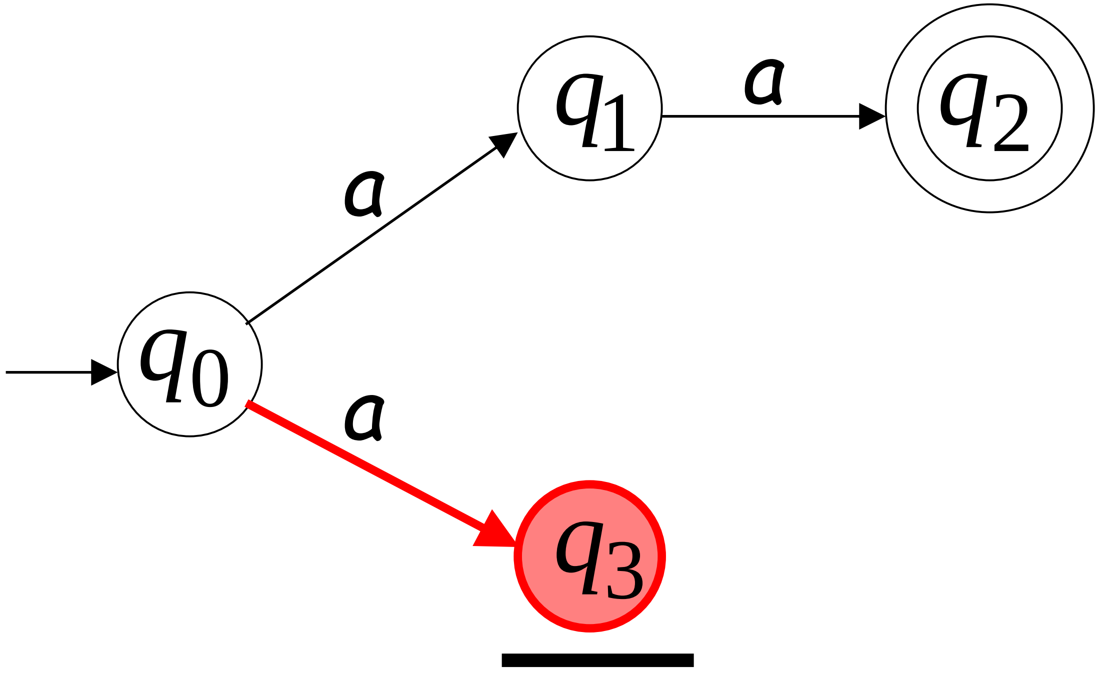


Second Choice

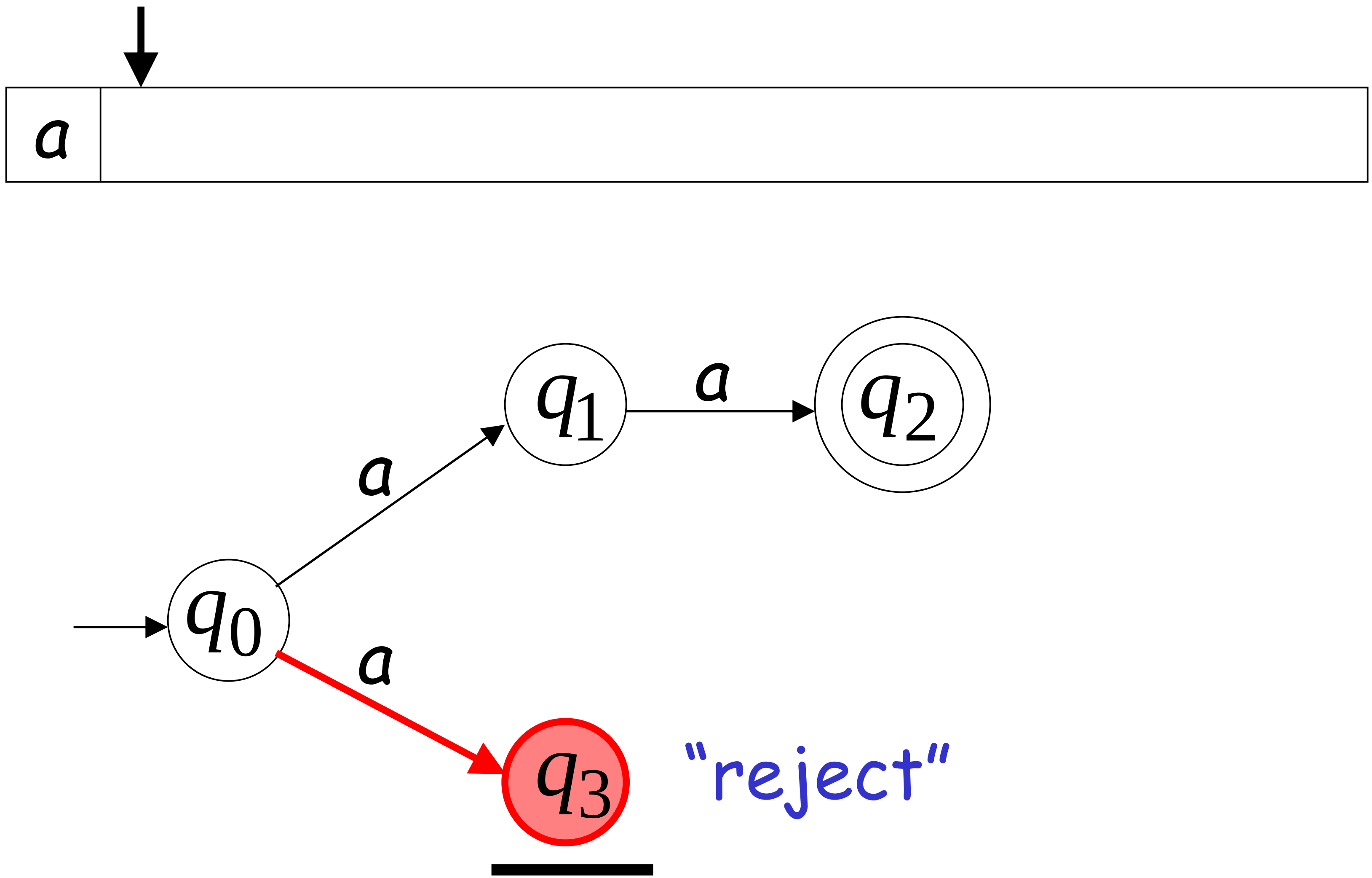


Second Choice





Second Choice



An NFA rejects a string:

when there is no computation of the NFA that accepts the string.

For each computation:

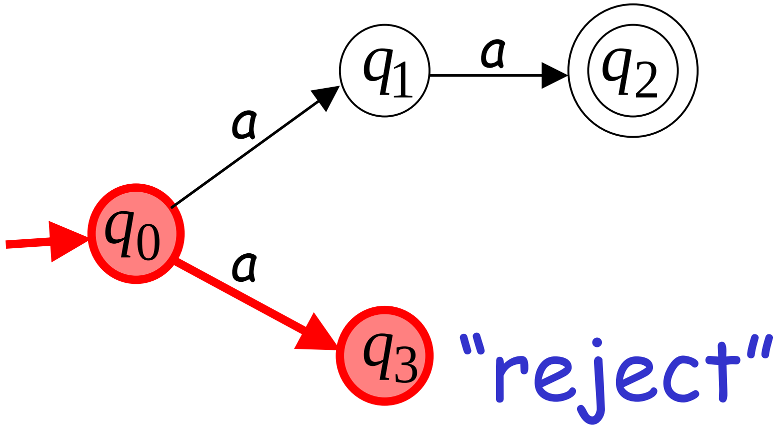
- All the input is consumed and the automaton is in a non final state

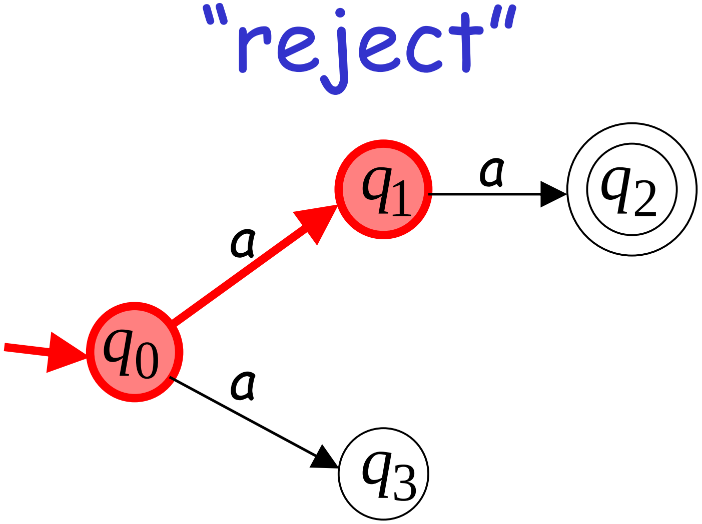
OR

- The input cannot be consumed

Example

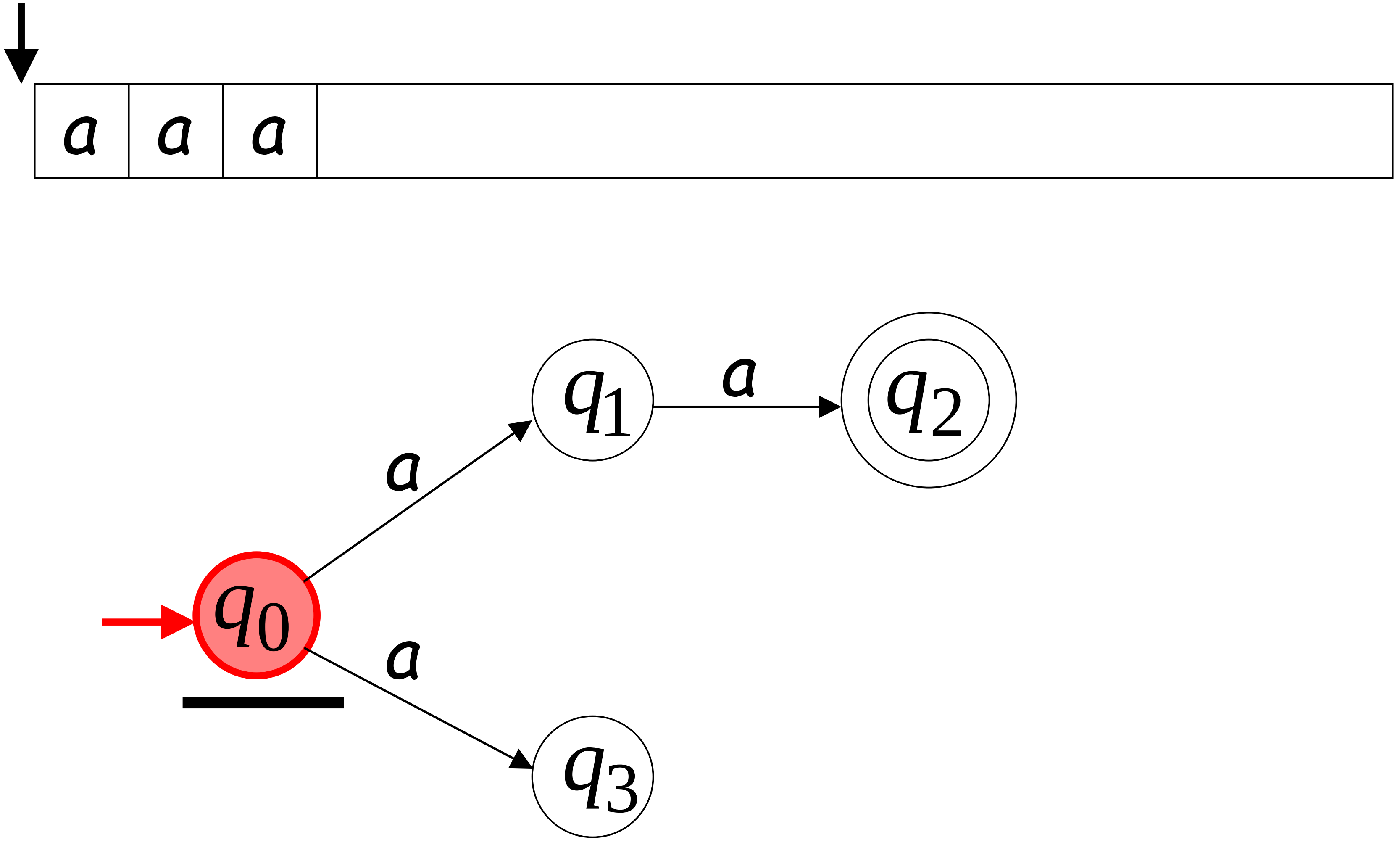
a is rejected by the NFA:



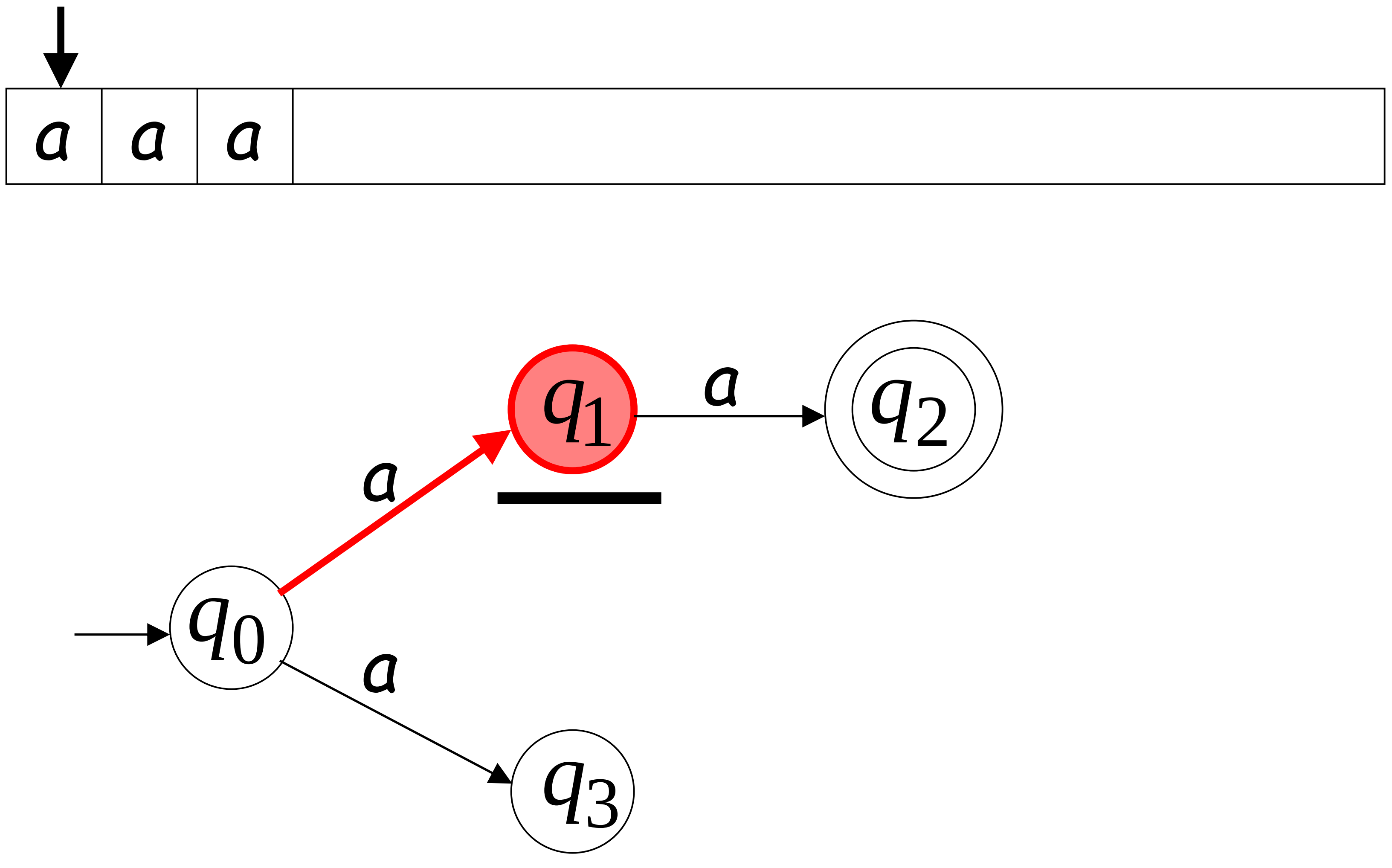


All possible computations lead to rejection

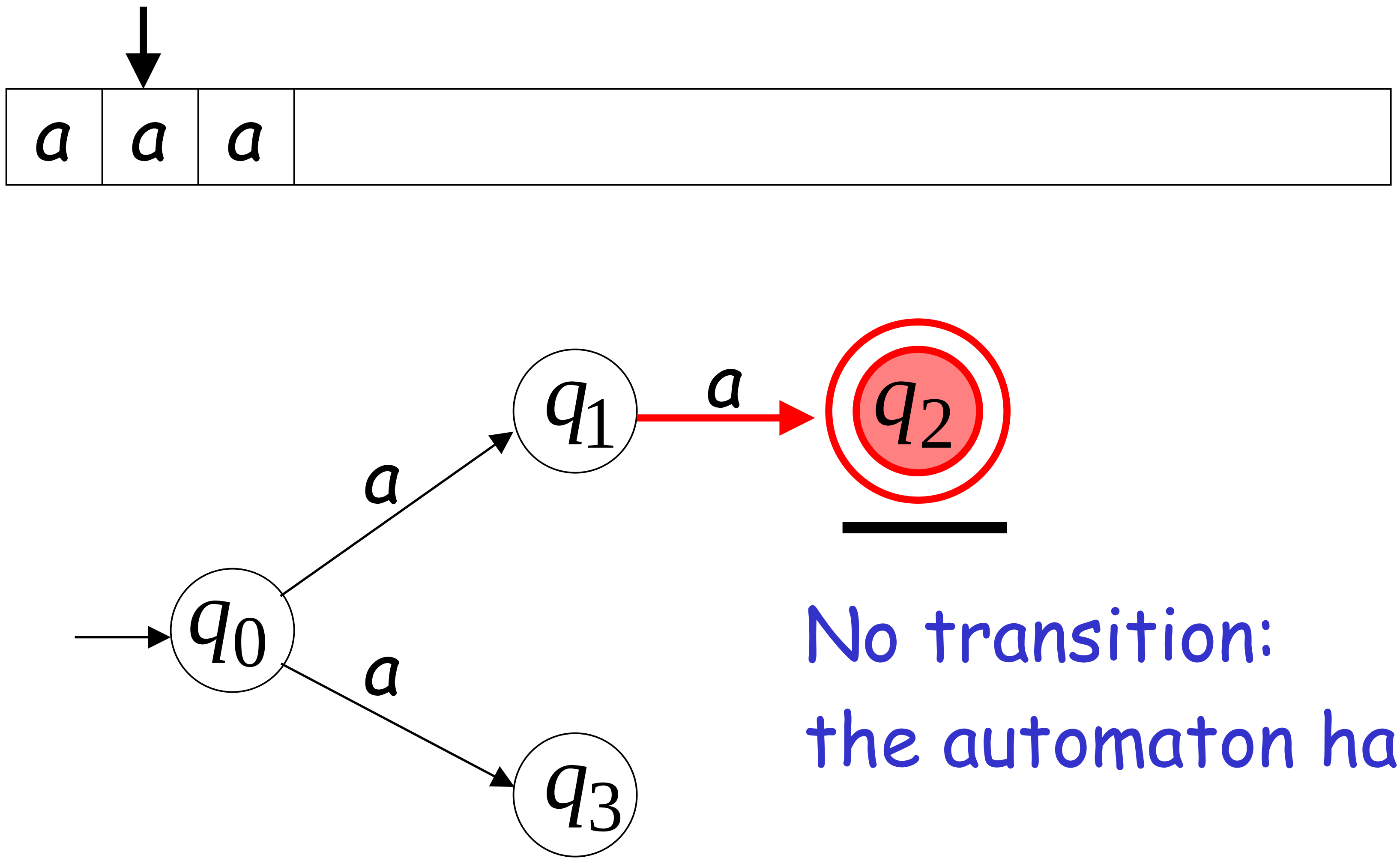
Rejection example



First Choice

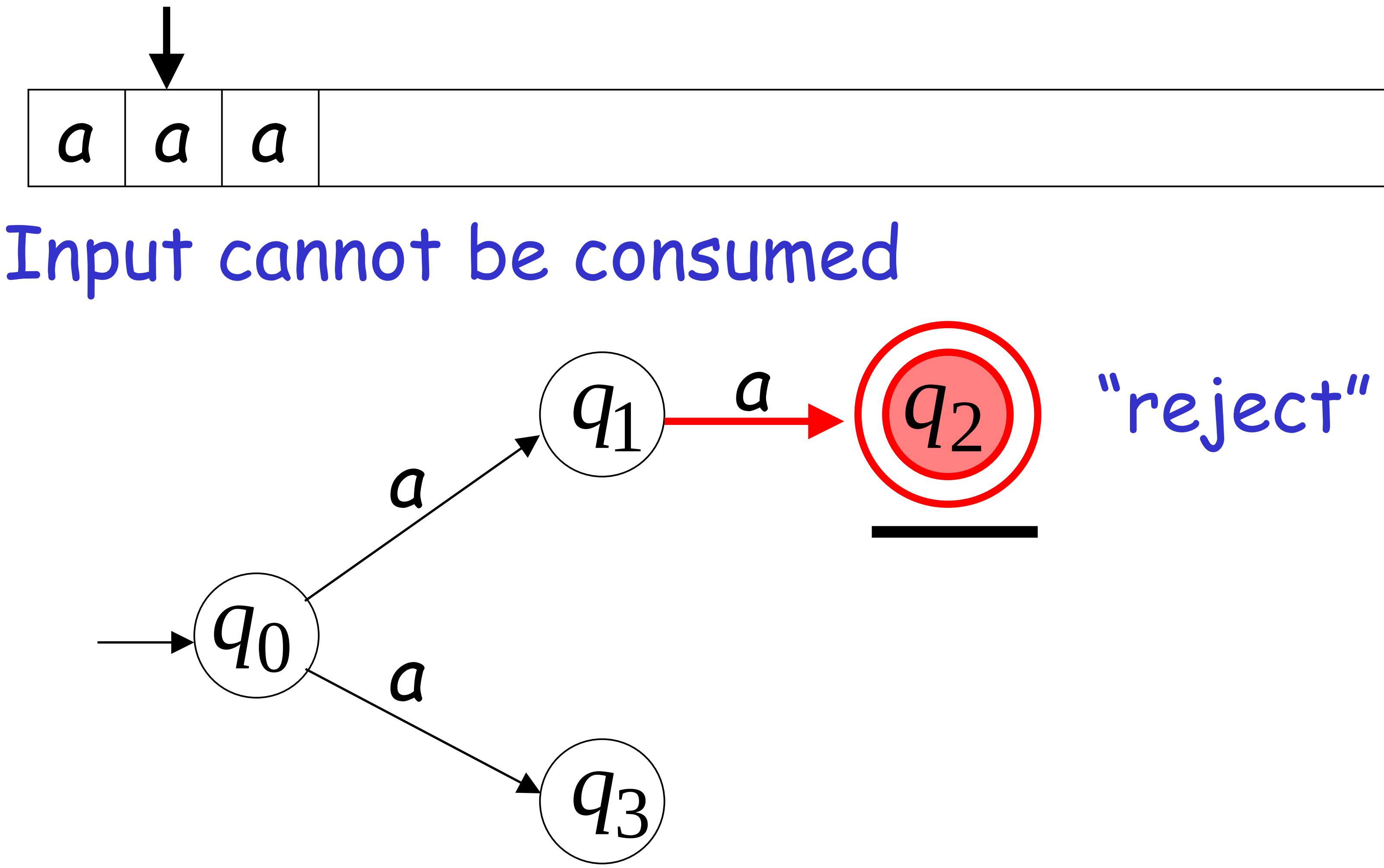


First Choice

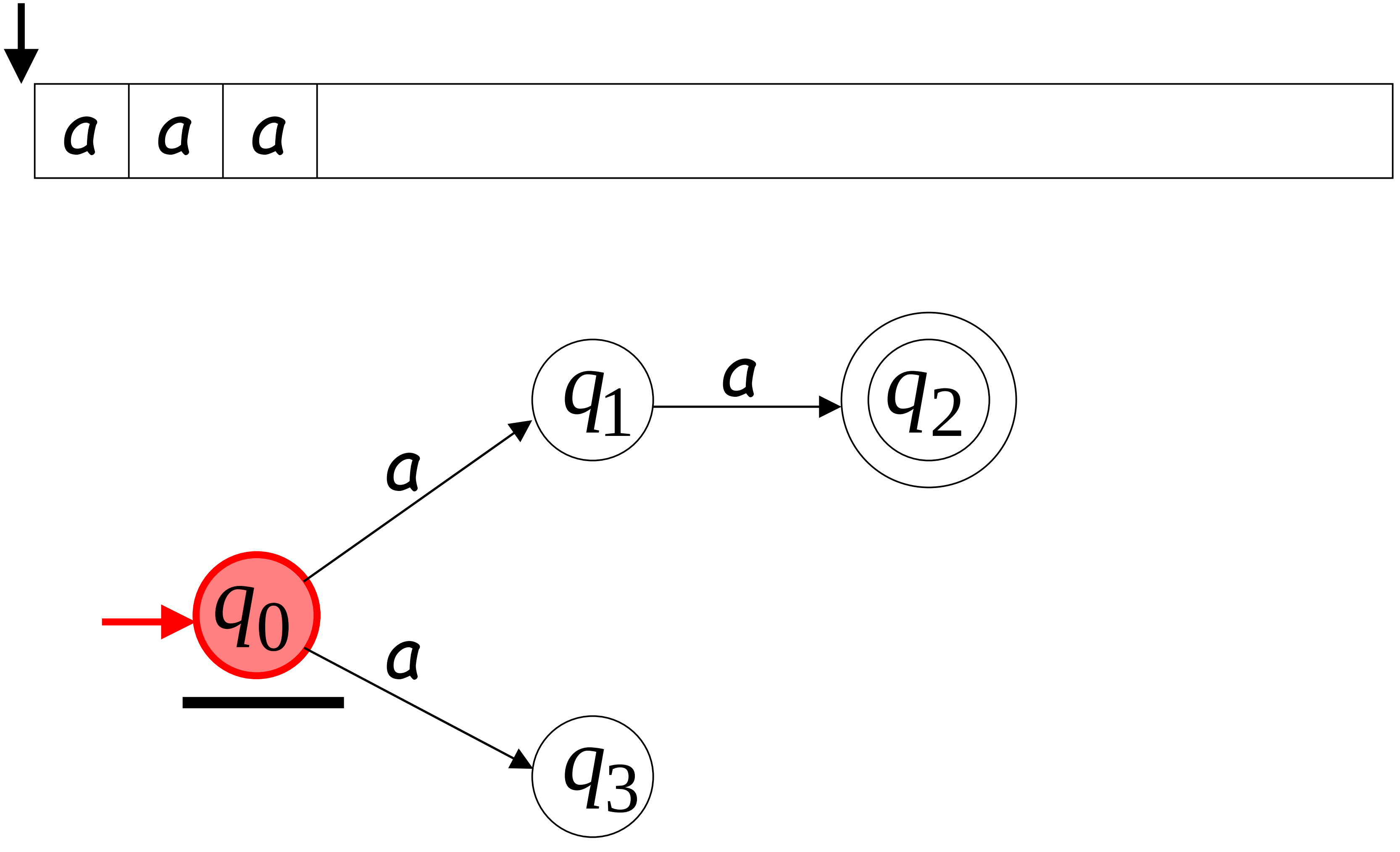


No transition:
the automaton hangs

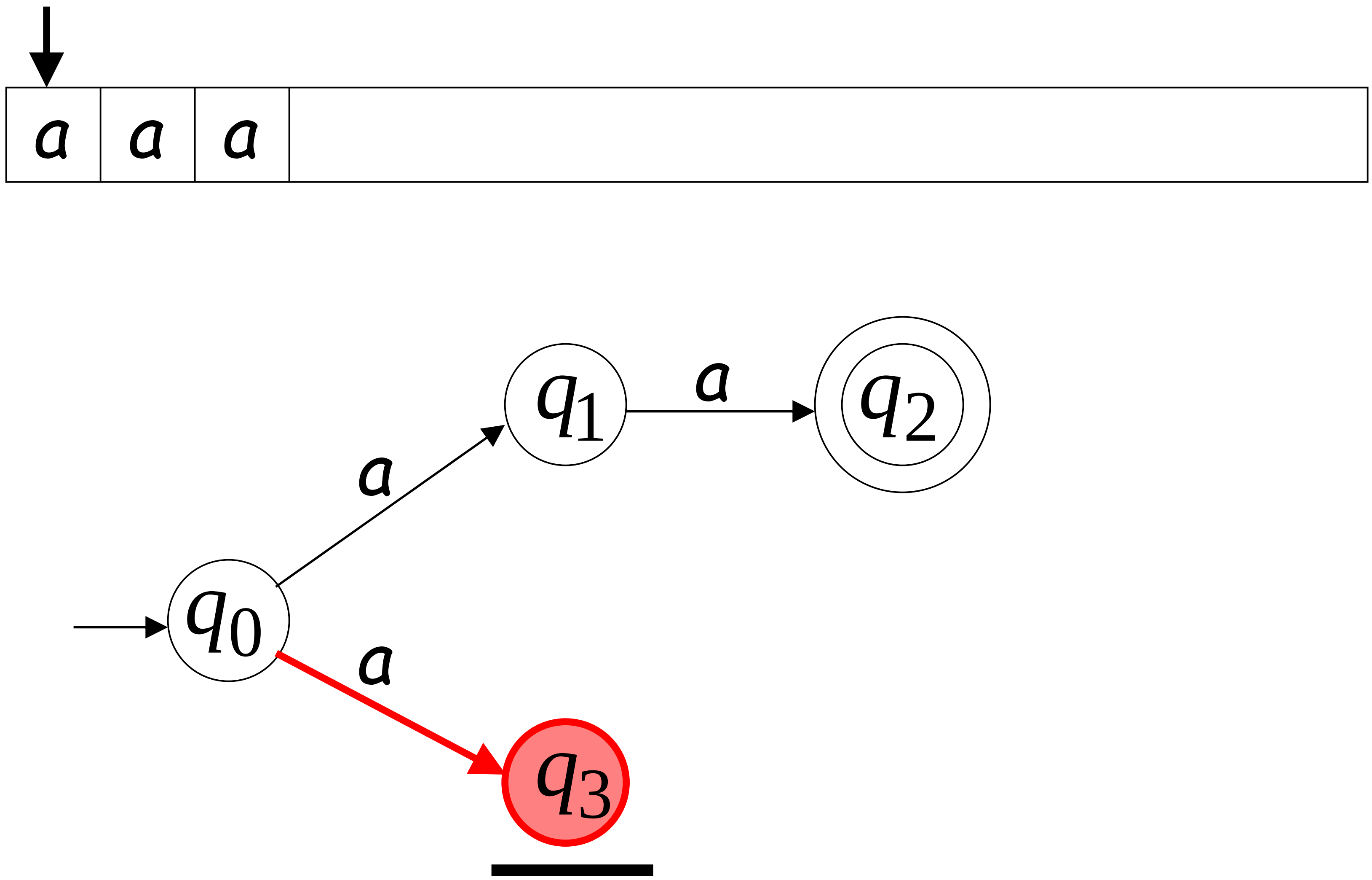
First Choice



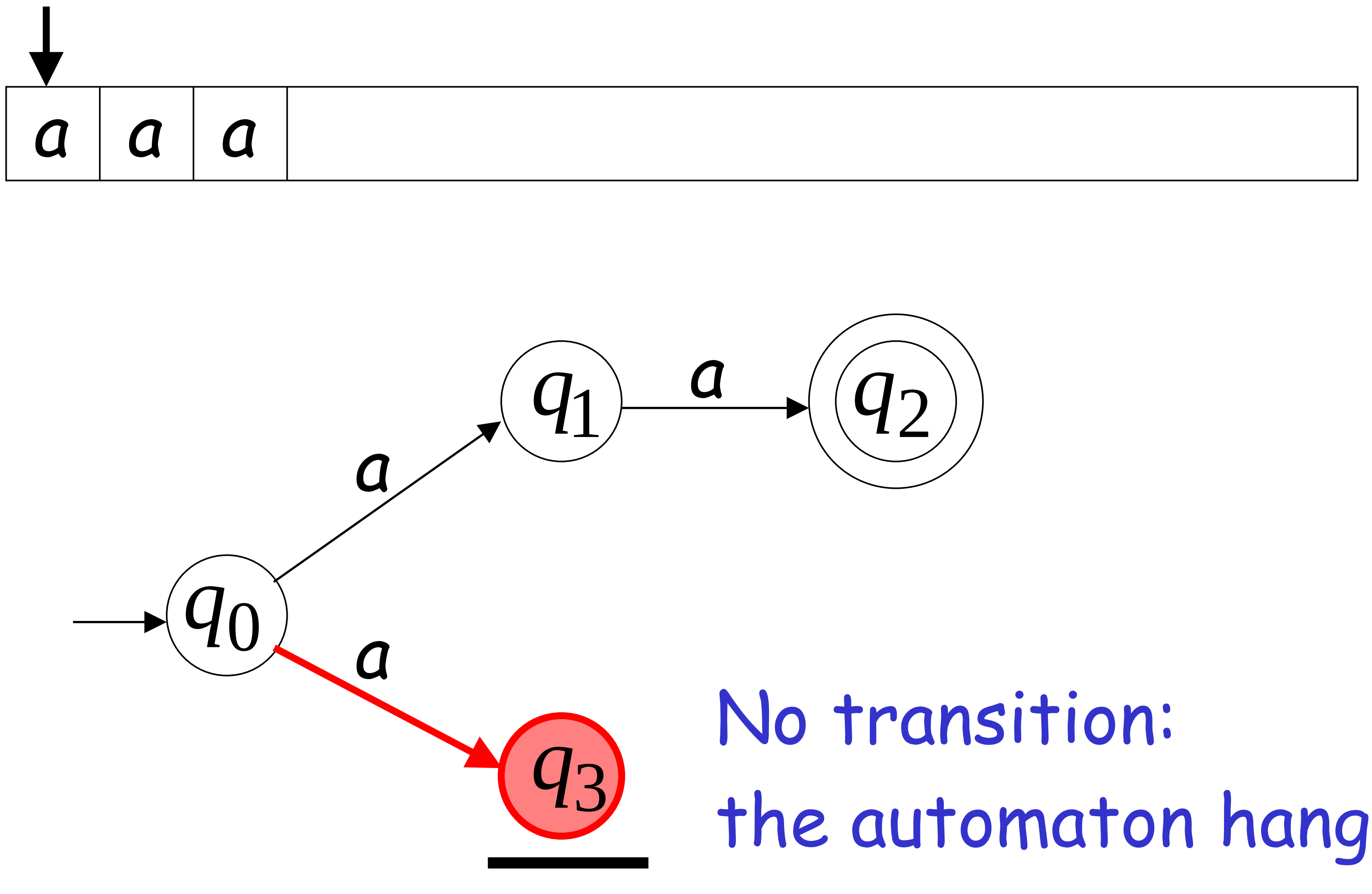
Second Choice



Second Choice



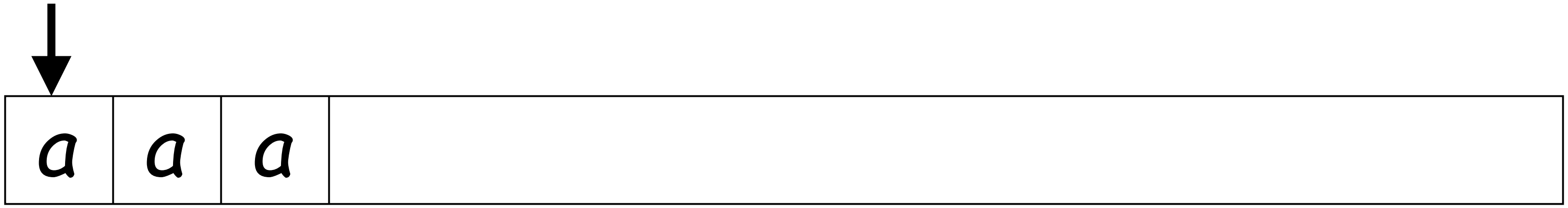
Second Choice



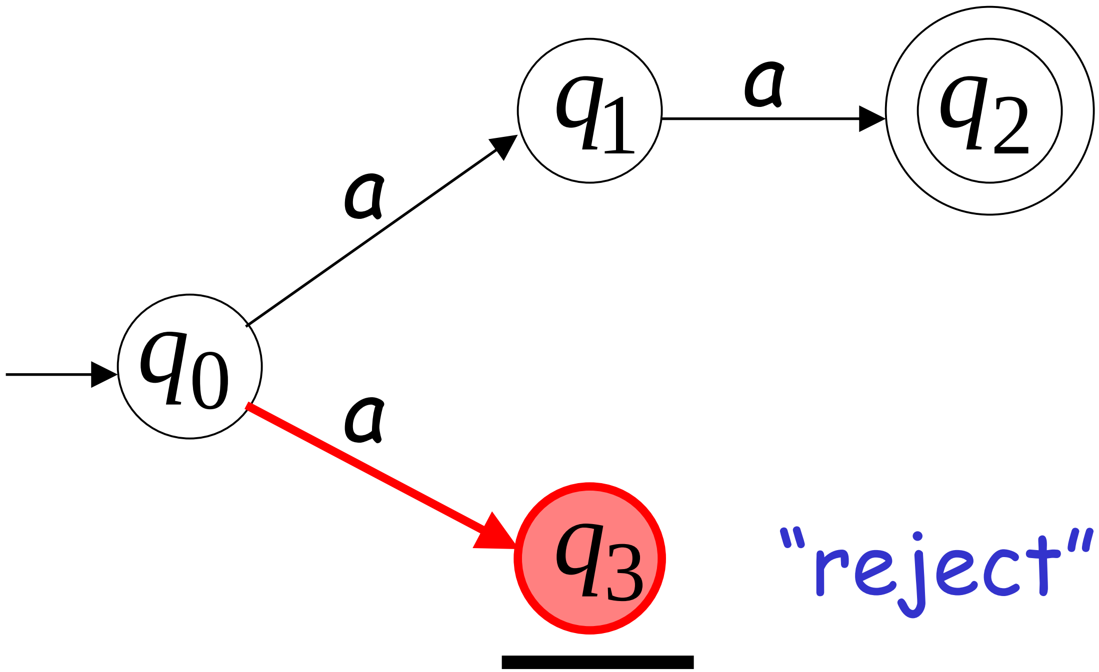
No transition:

the automaton hangs

Second Choice

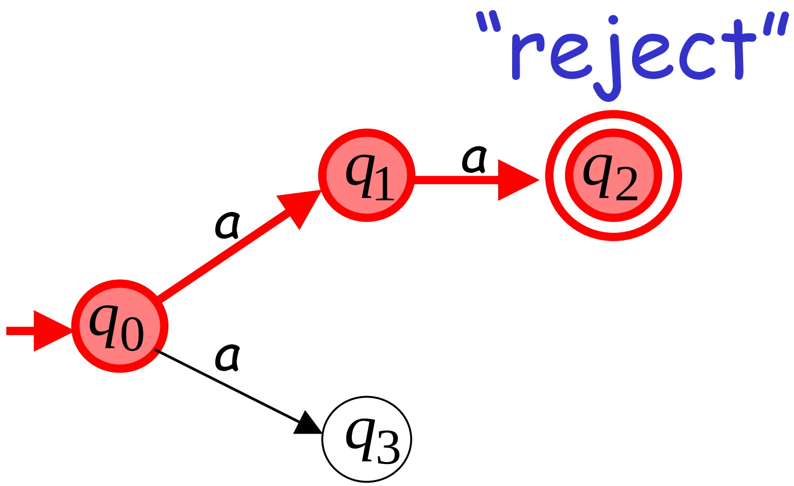


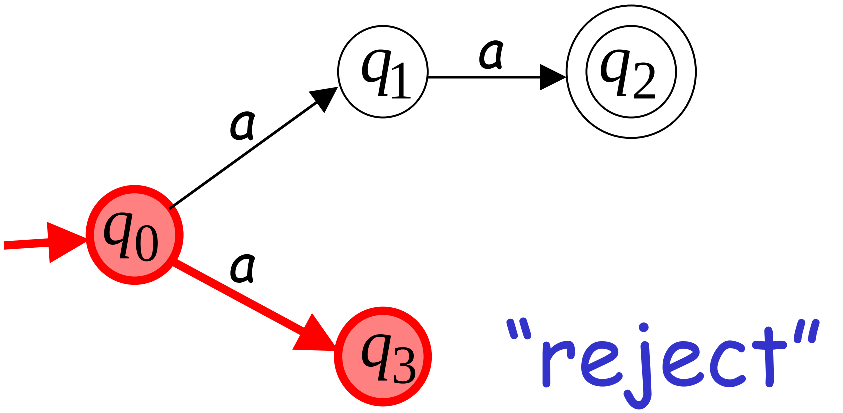
Input cannot be consumed



"reject"

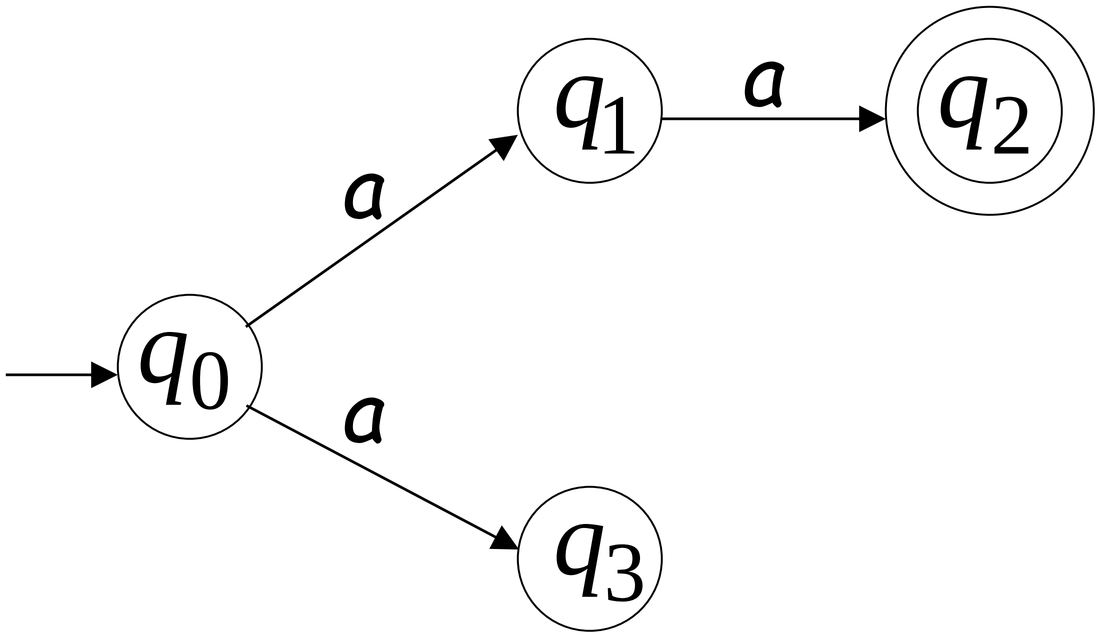
aaa is rejected by the NFA:



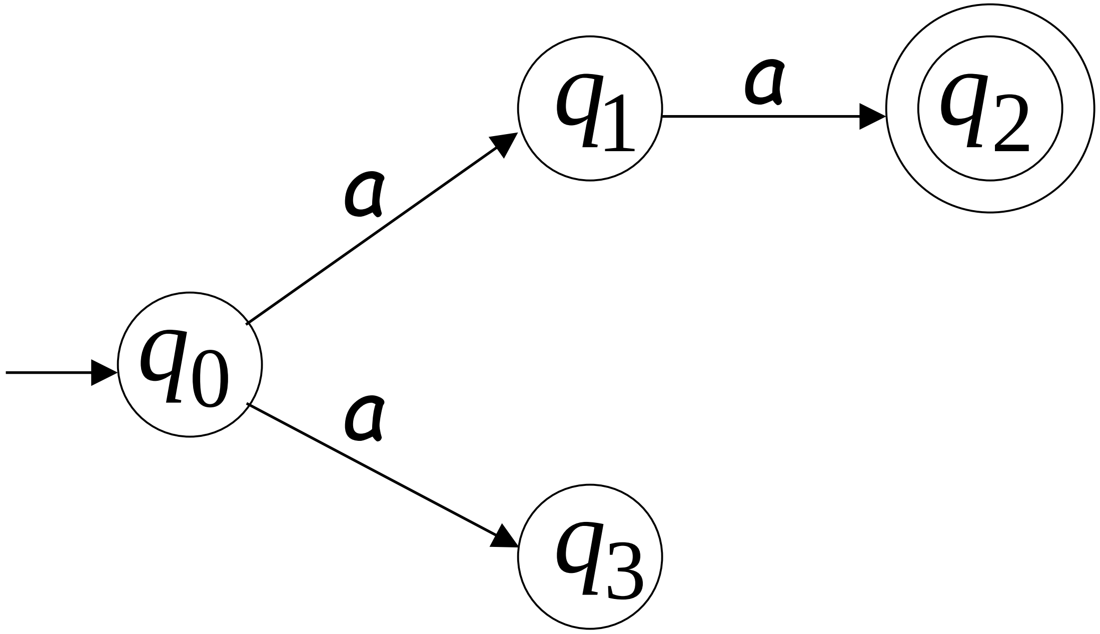


All possible computations lead to rejection

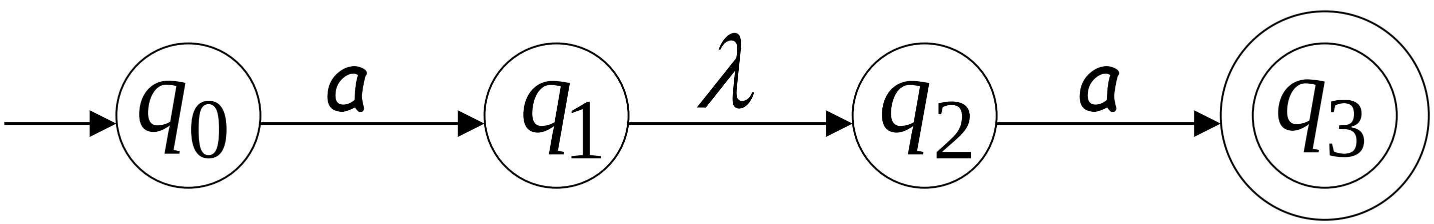
L(M)?

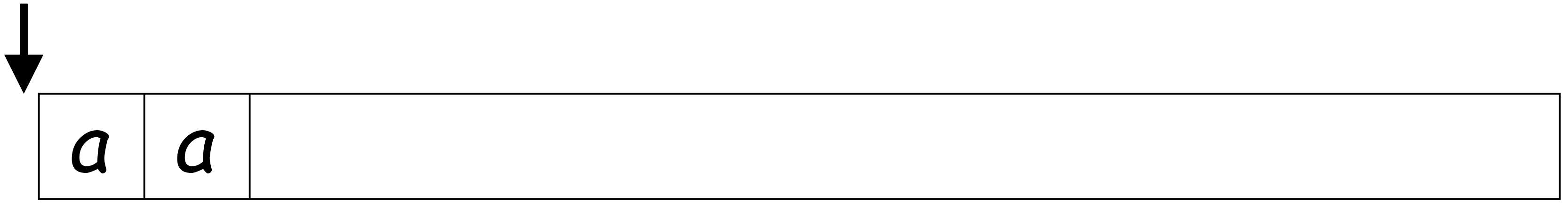


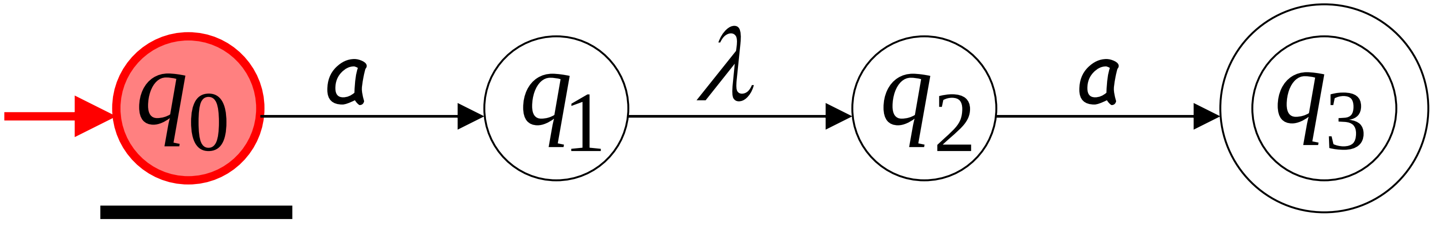
Language accepted: $L = \{aa\}$

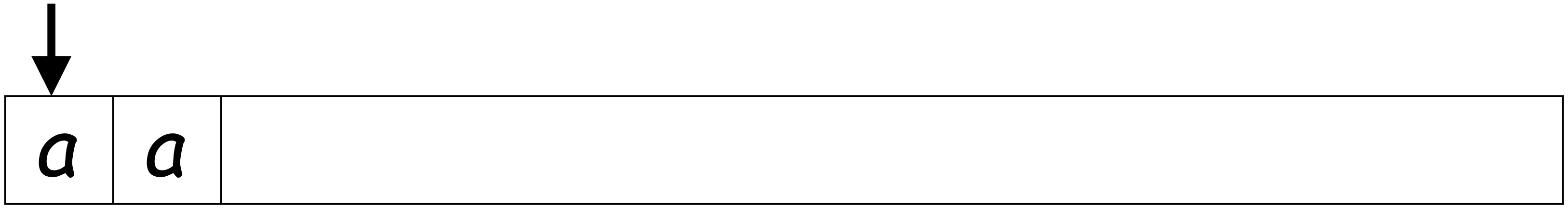


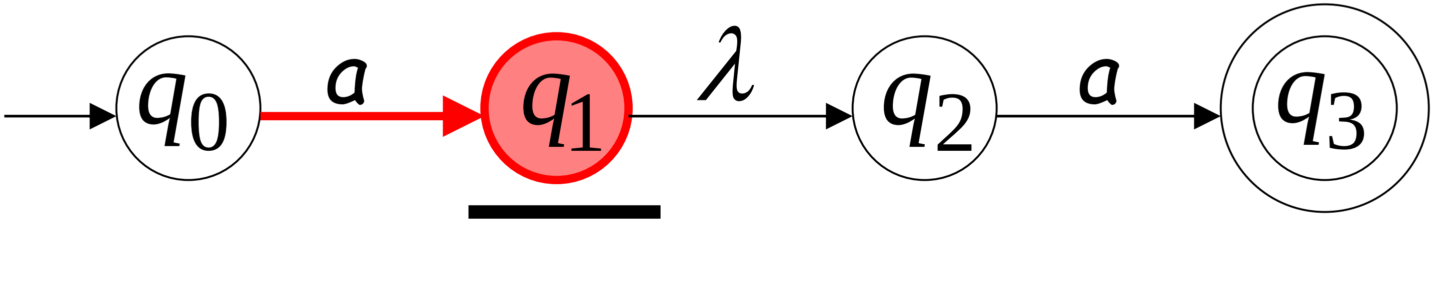
Lambda Transitions



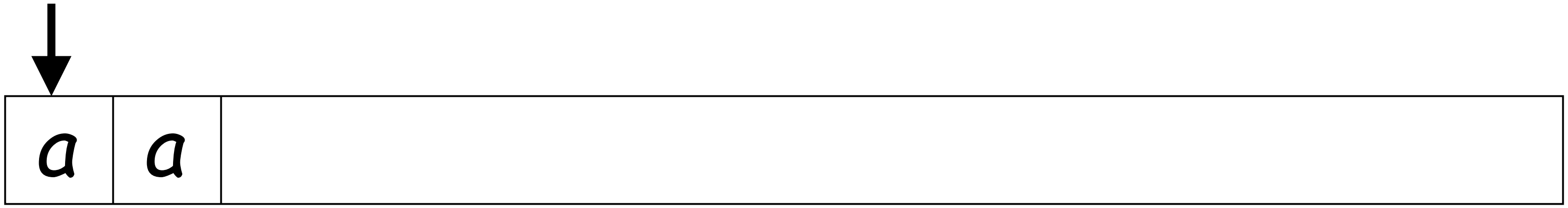

a
a

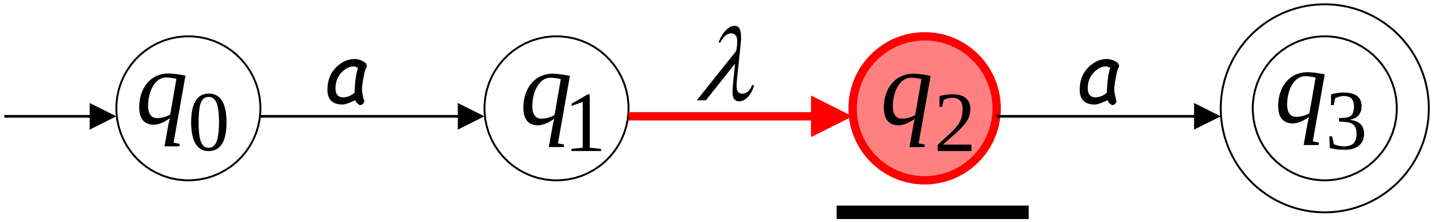

q_0
a
q_1
λ
q_2
a
q_3

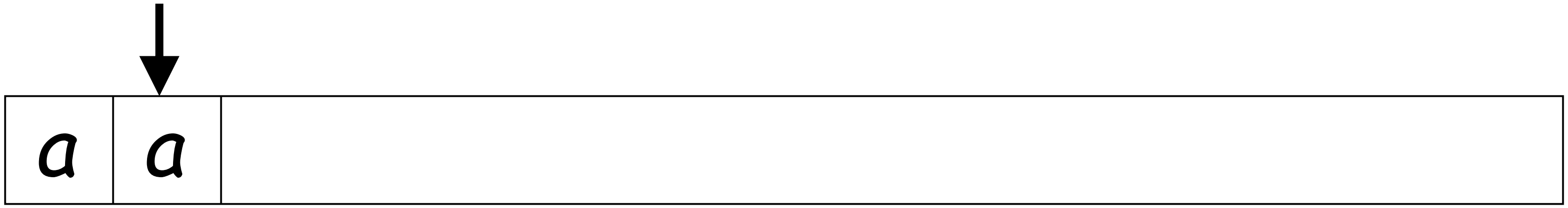

a
a

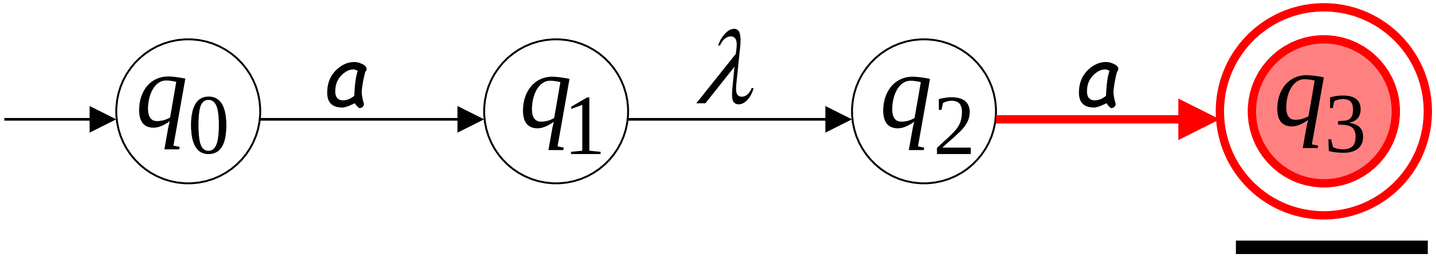

q_0
a
q_1
λ
q_2
a
q_3

(read head does not move)

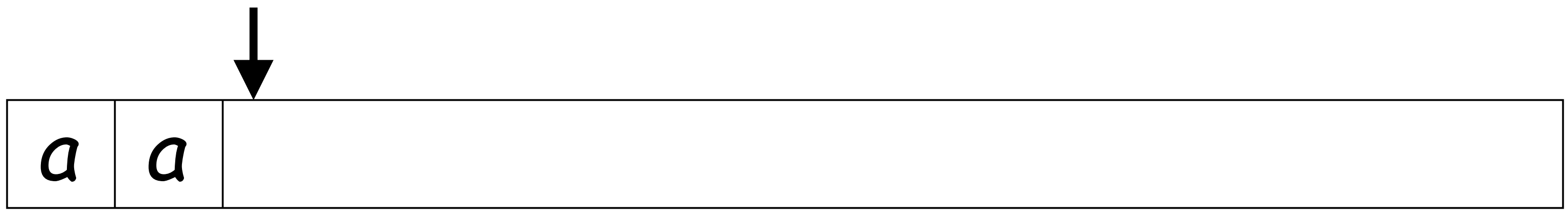


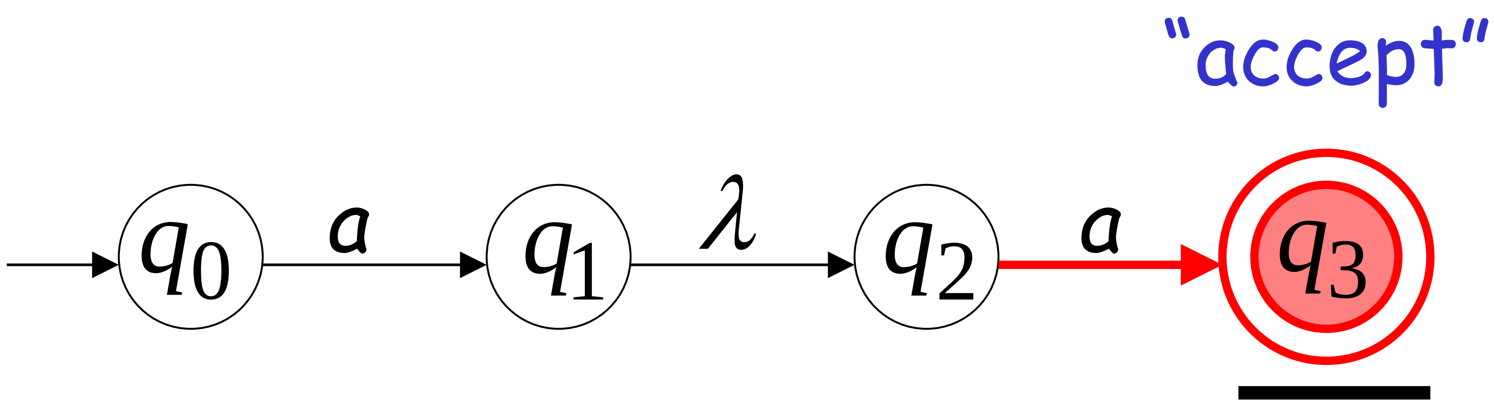



a
a


q_0
a
q_1
λ
q_2
a
q_3

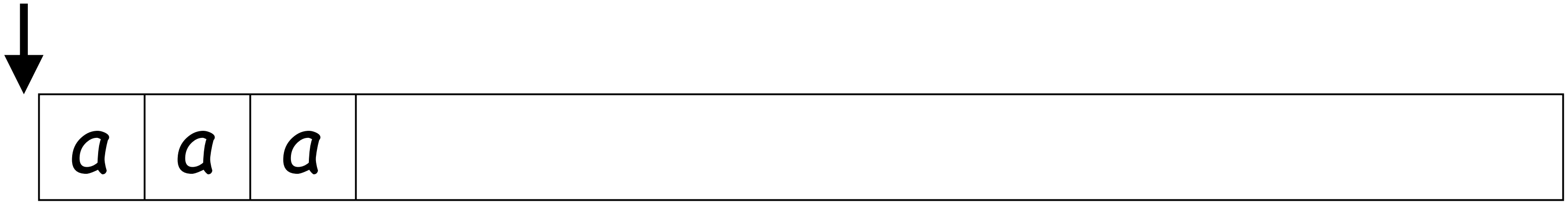
all input is consumed

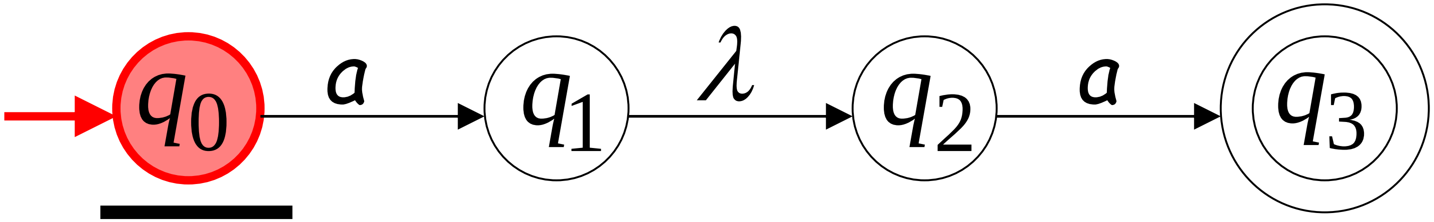


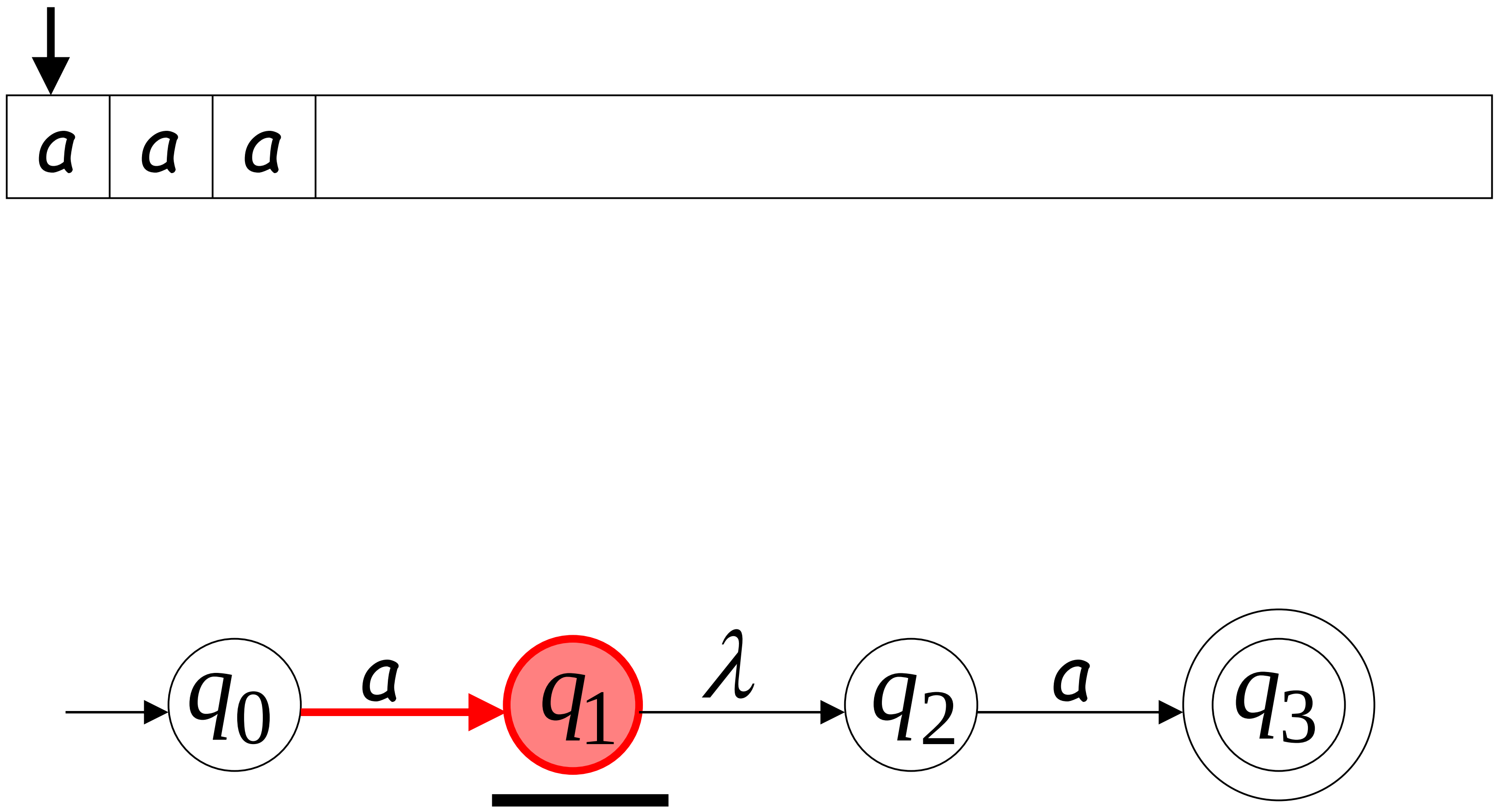


String aa is accepted

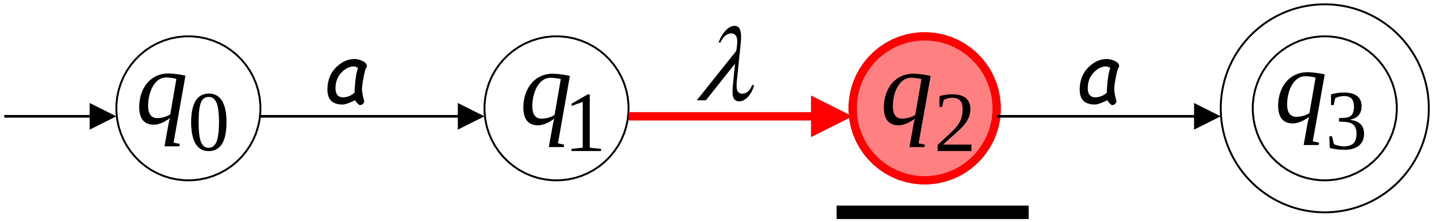
Rejection Example

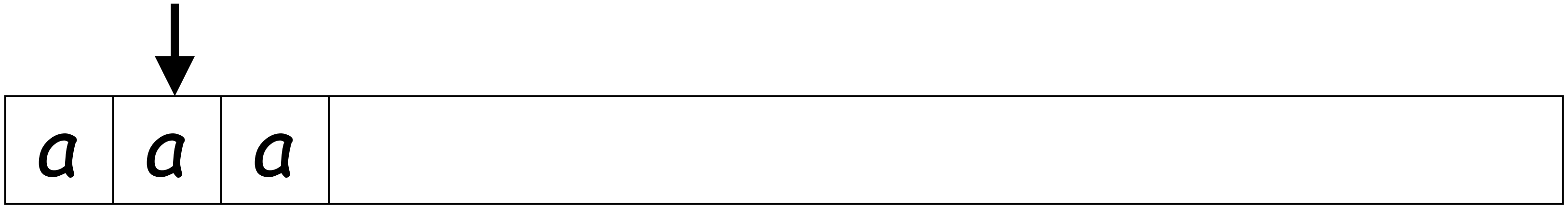


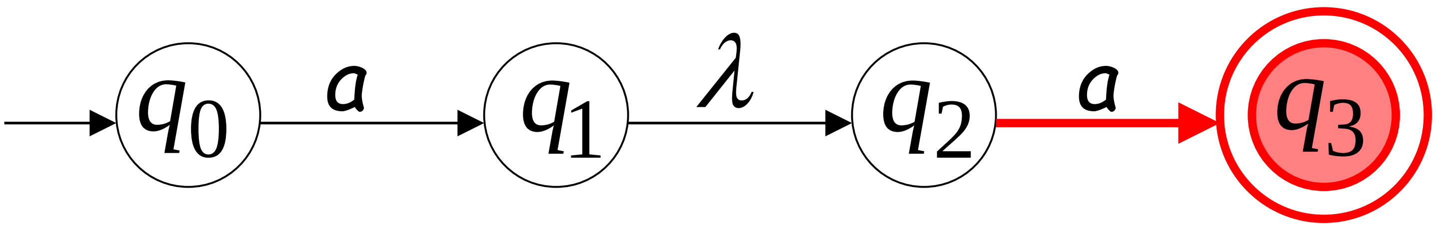



a
a
a
q_0
a
q_1
λ
q_2
a
q_3

(read head doesn't move)

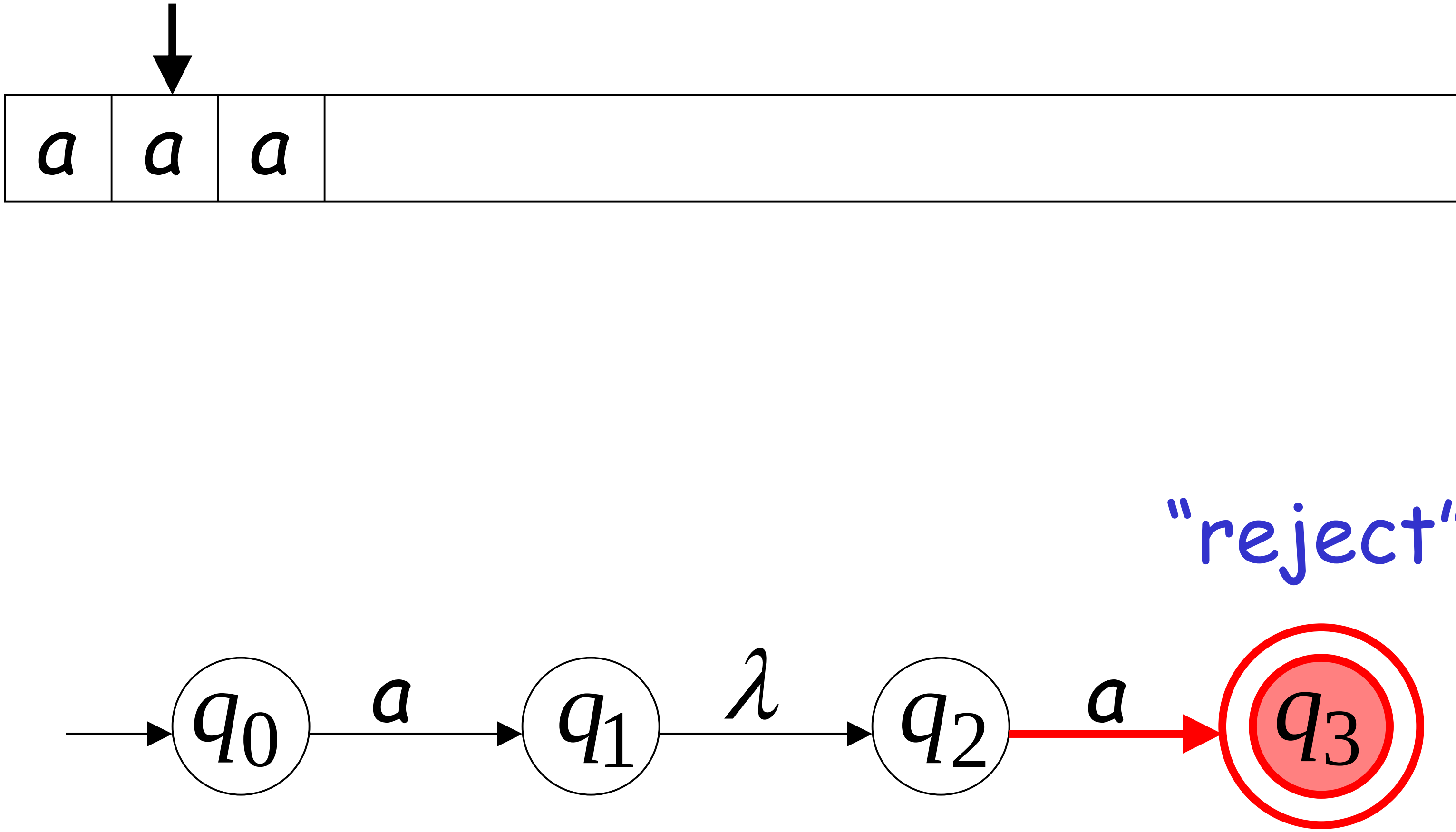






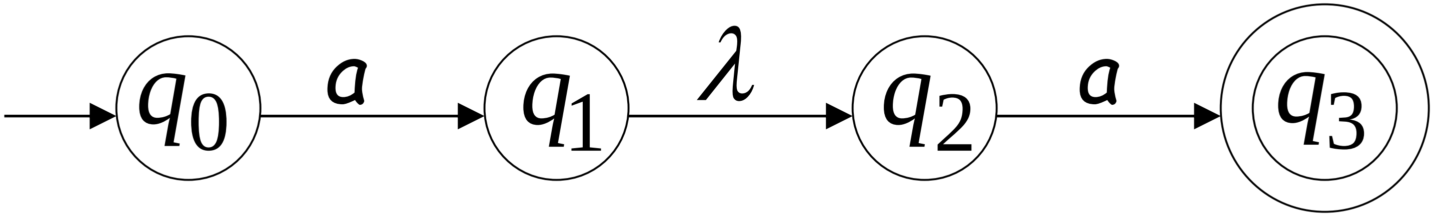
No transition:
the automaton hangs

Input cannot be consumed

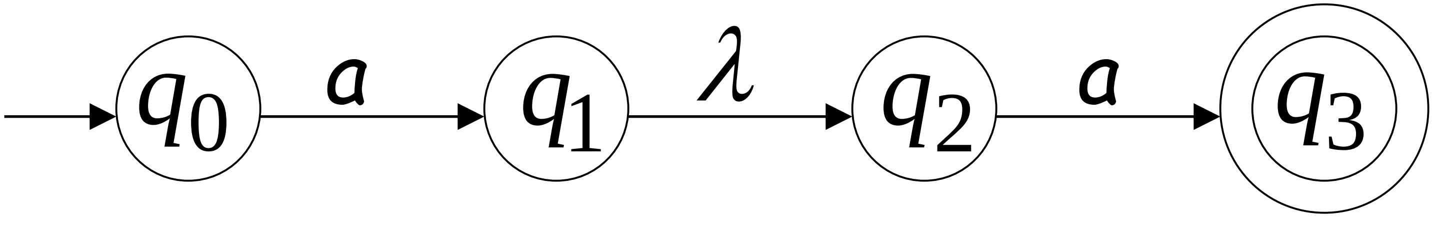


String aaa is rejected

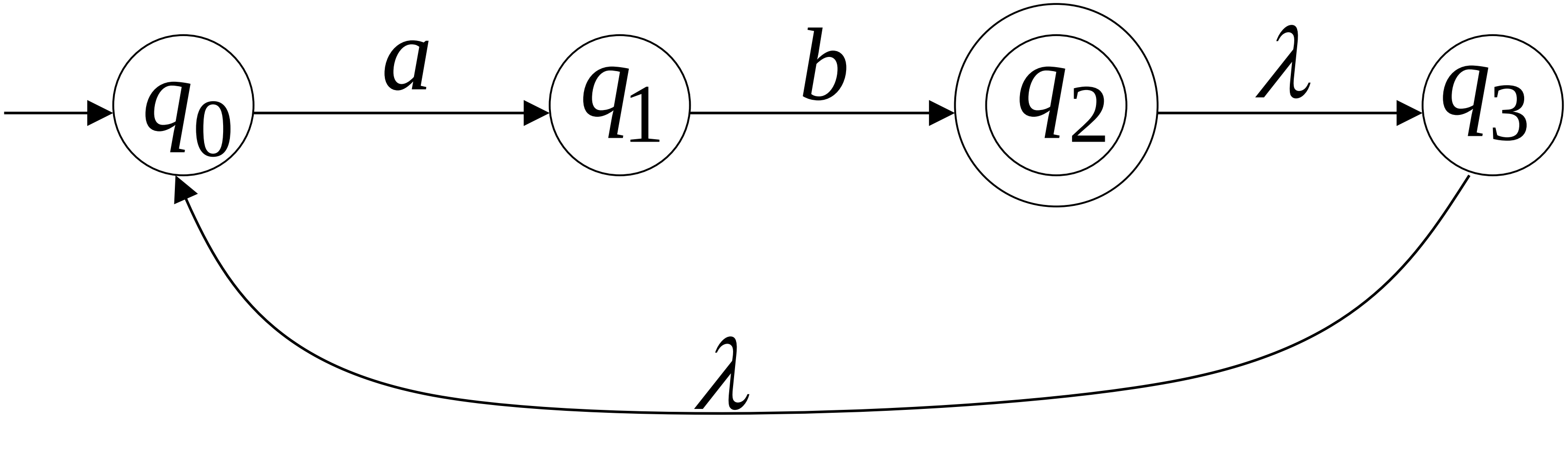
L(M)?

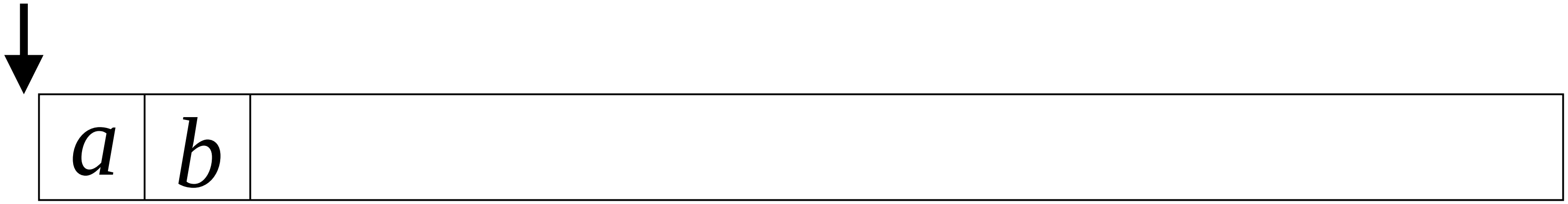


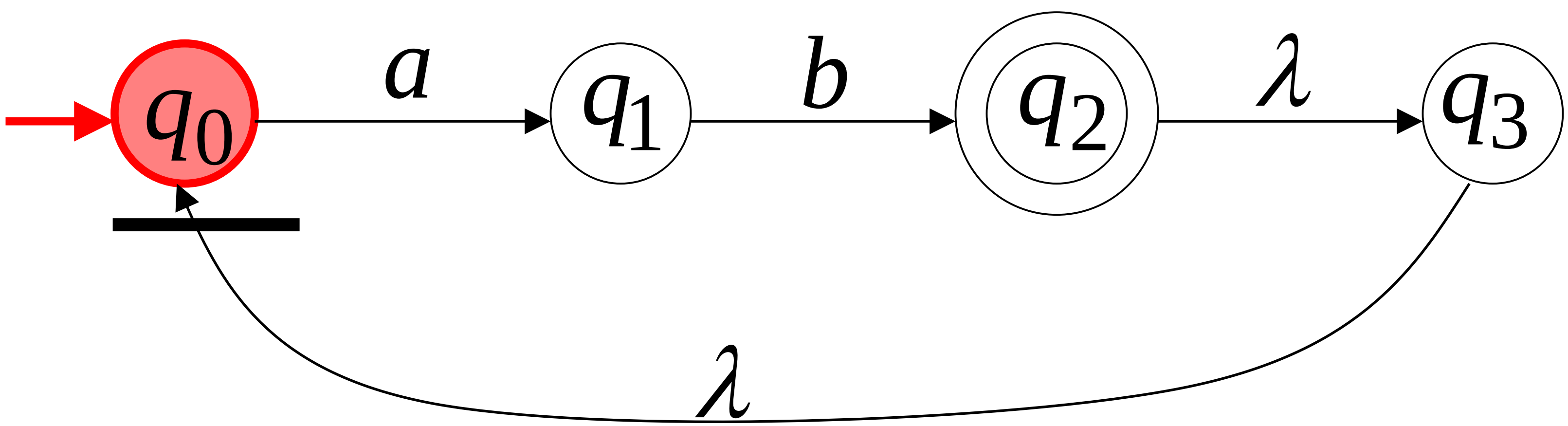
Language accepted: $L = \{aa\}$

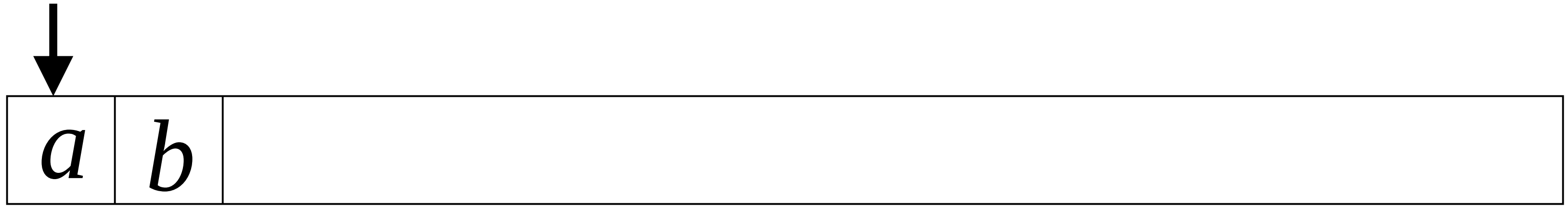


Another NFA Example: L(M)?

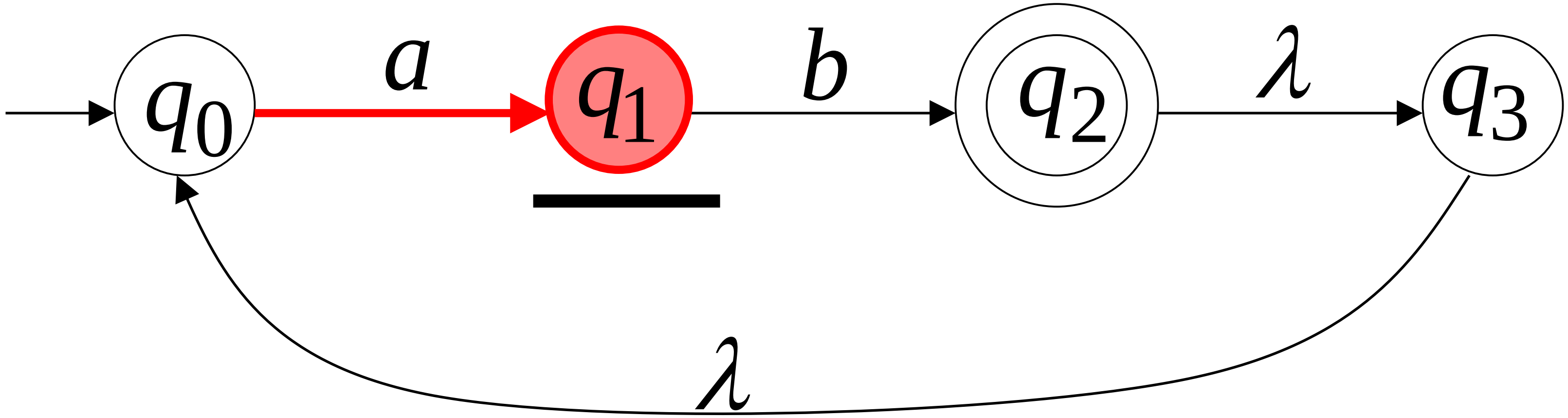



a
b

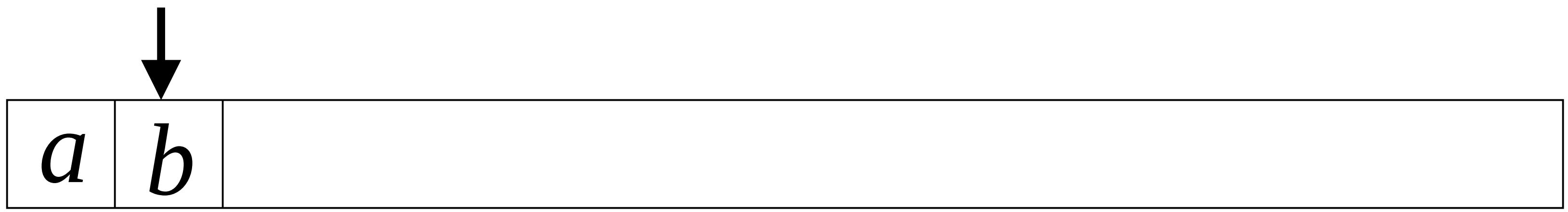

q_0
a
q_1
b
q_2
λ
q_3
λ

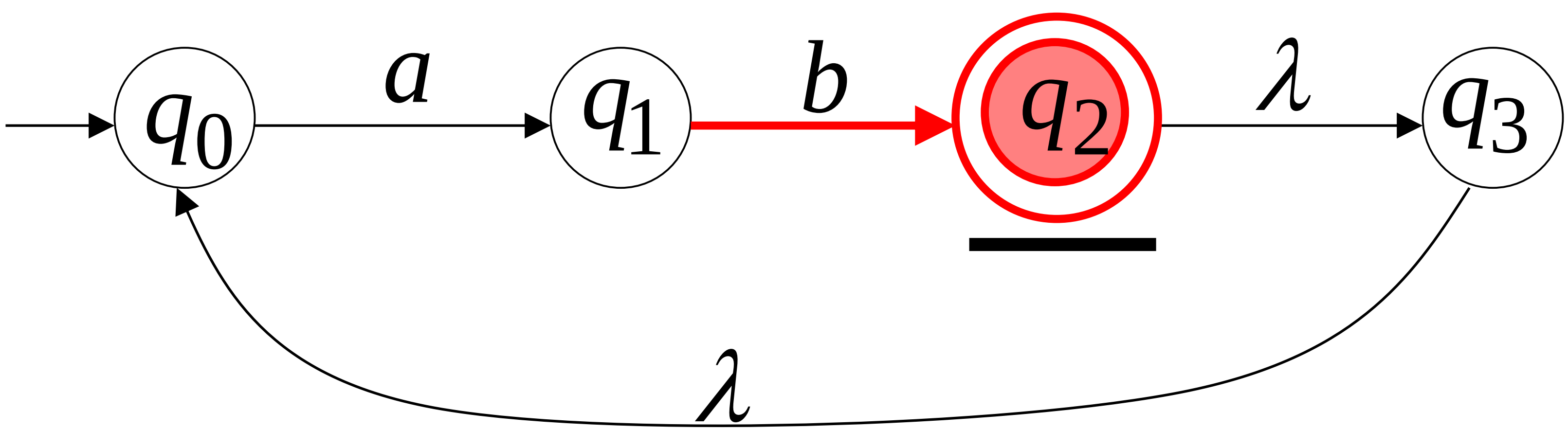


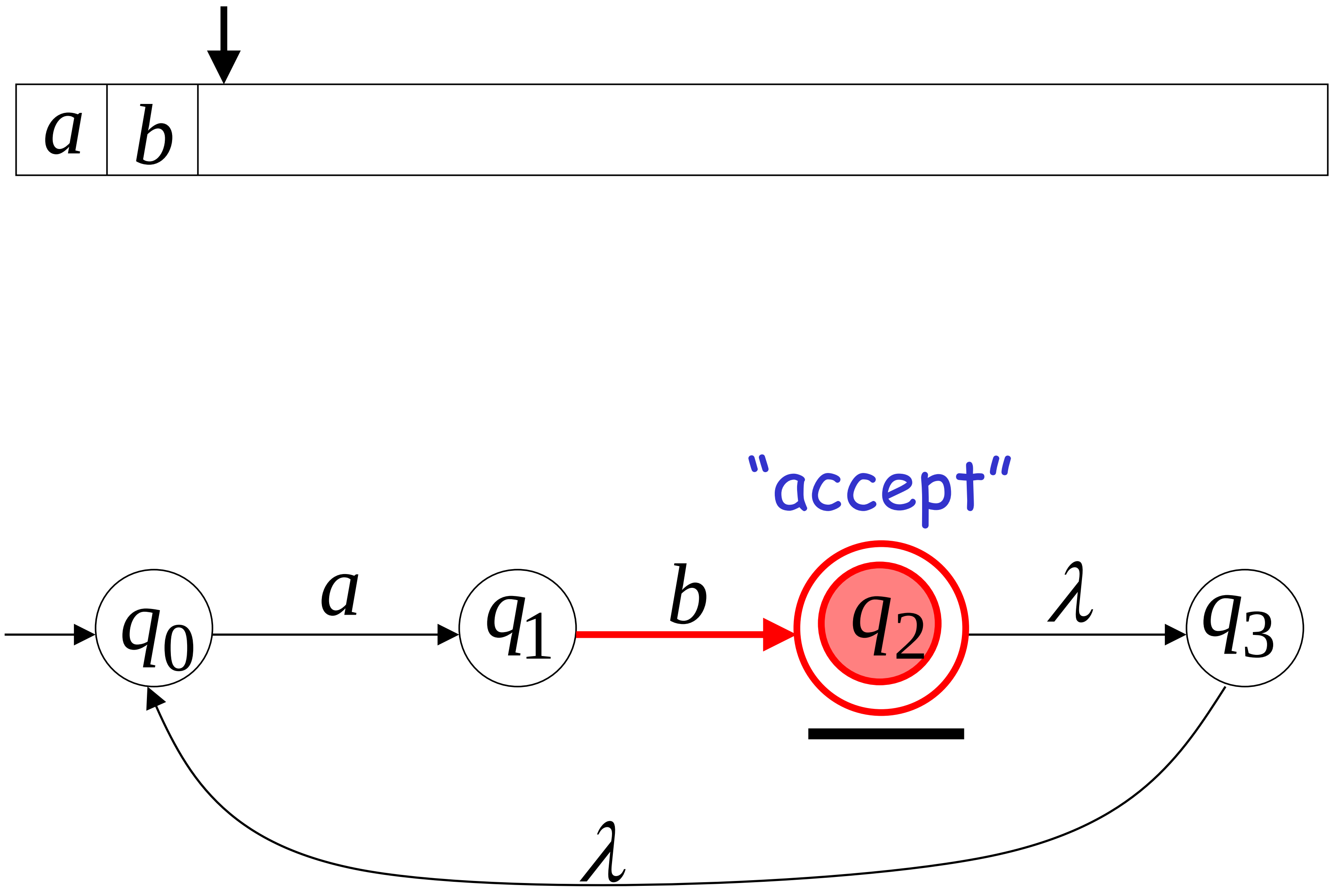
a
b



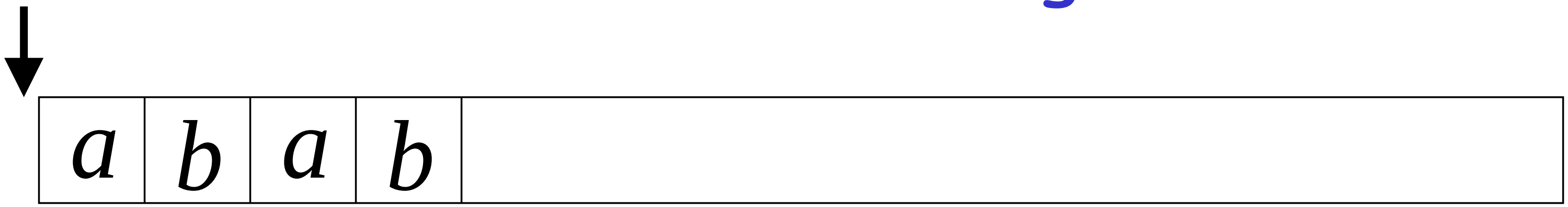
q_0
a
q_1
b
q_2
λ
q_3
λ

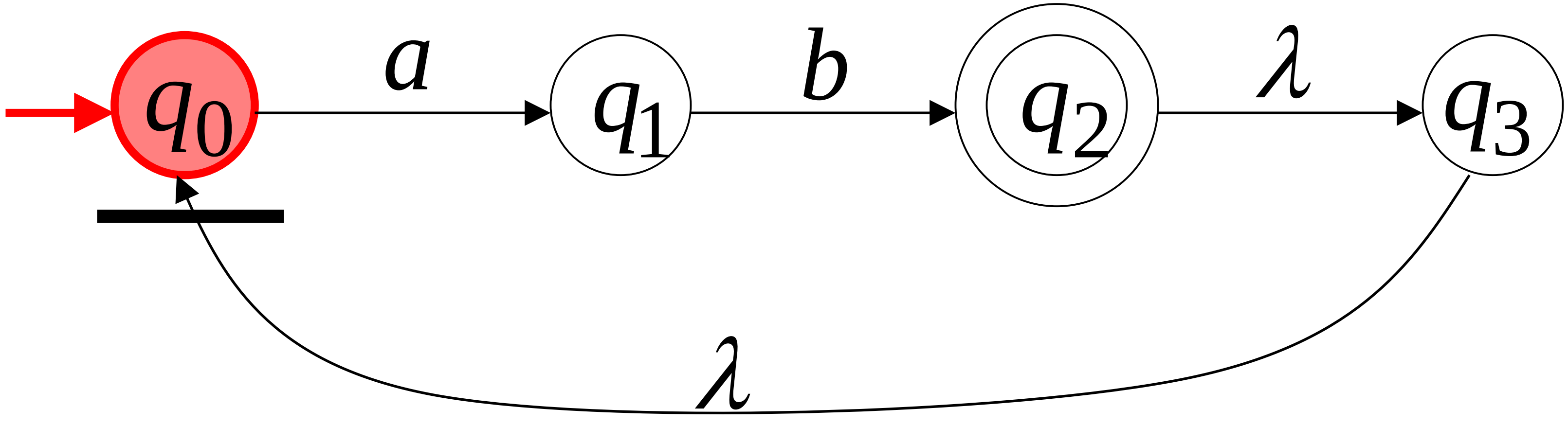

a
b

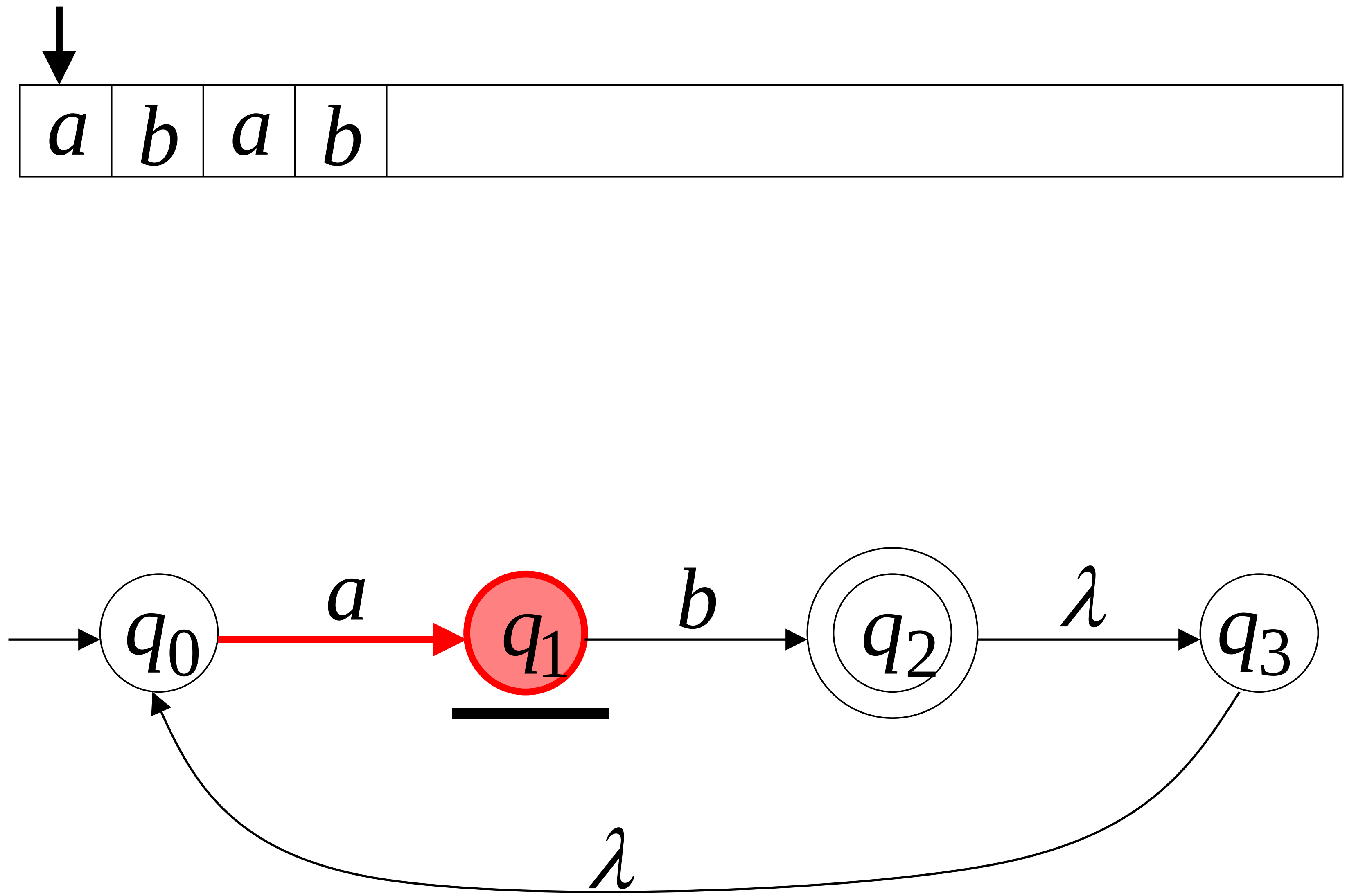

q0
a
q1
b
q2
λ
q3
λ

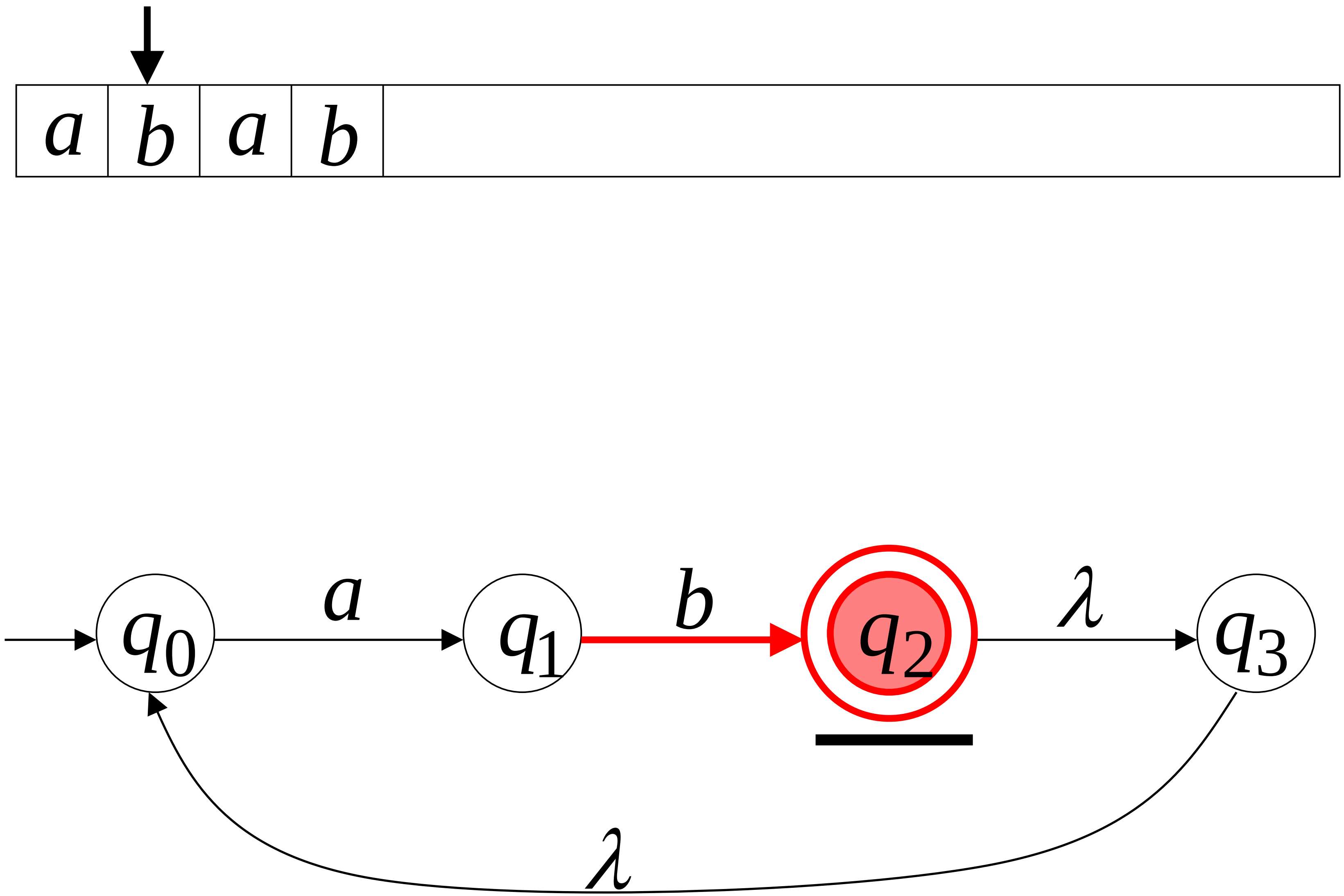

a
b
"accept"
q_0
a
q_1
b
q_2
λ
q_3
λ

Another String

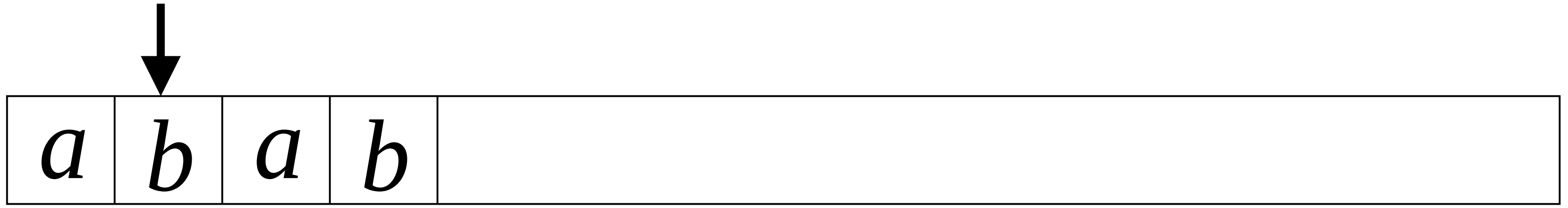


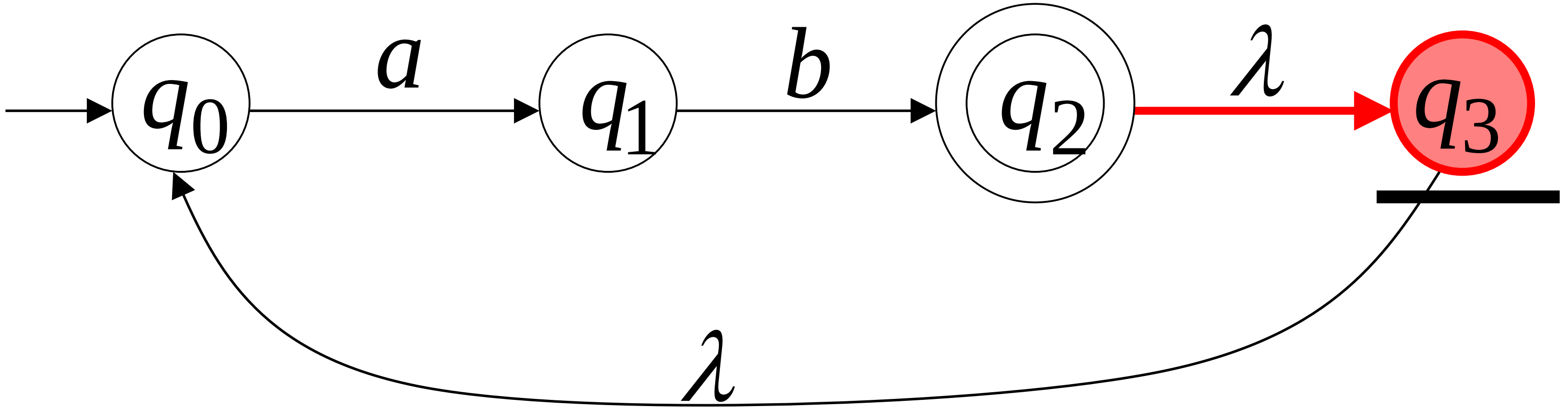


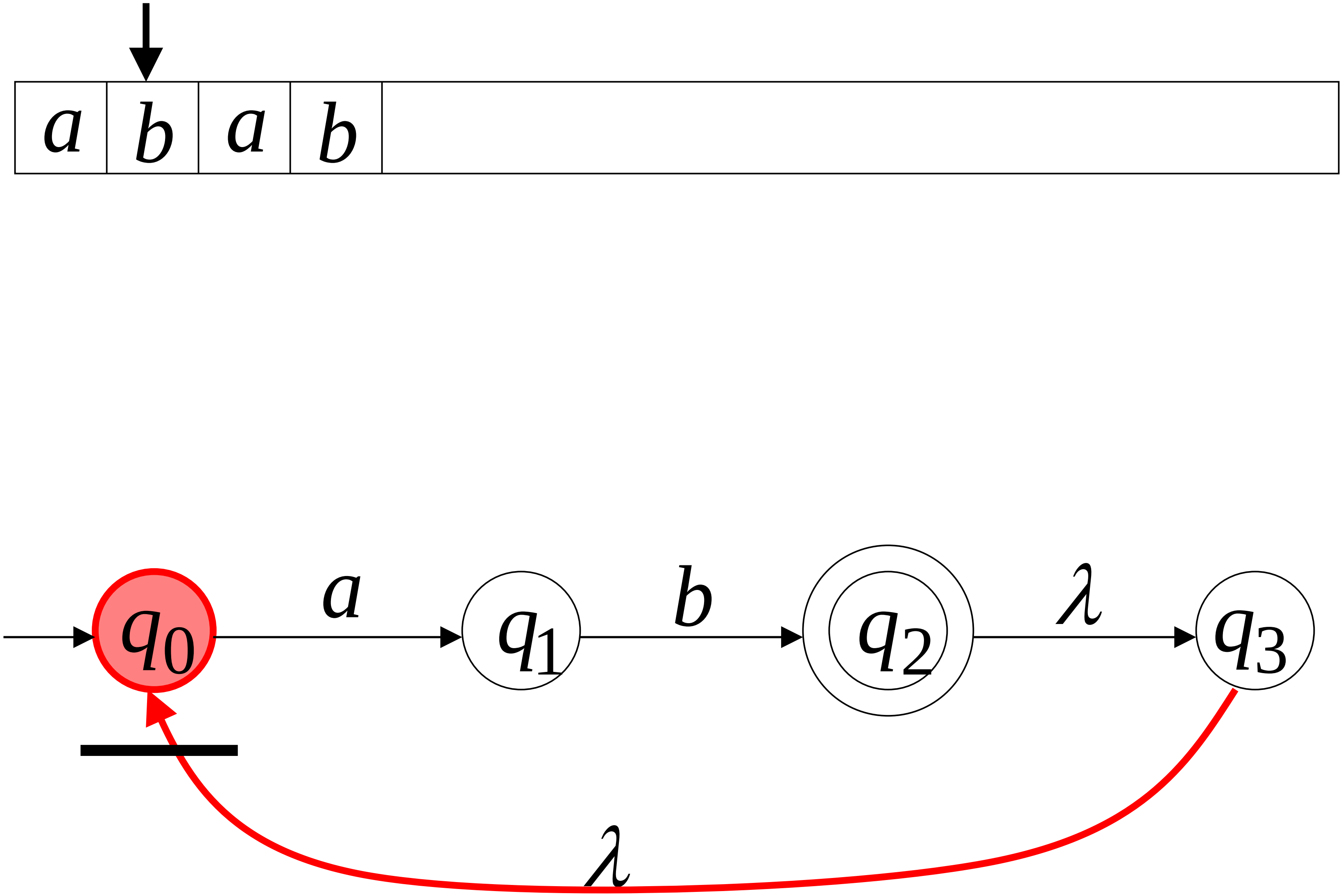

a
b
a
b
a
b
λ
λ
q0
q1
q2
q3



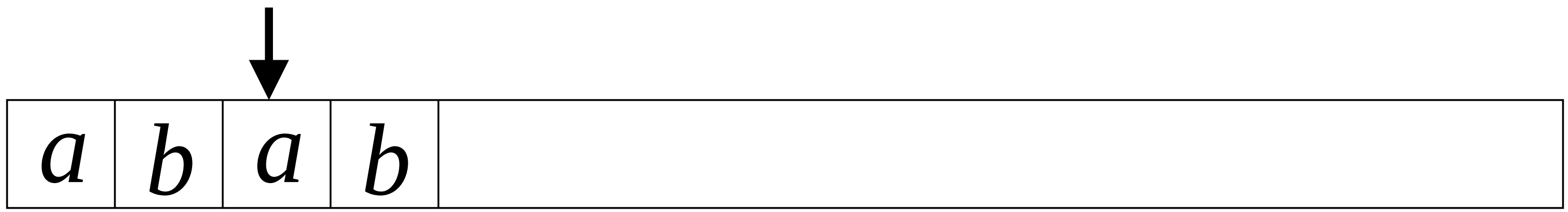
a
b
a
b
a
b
λ
λ
q_0
q_1
q_2
q_3

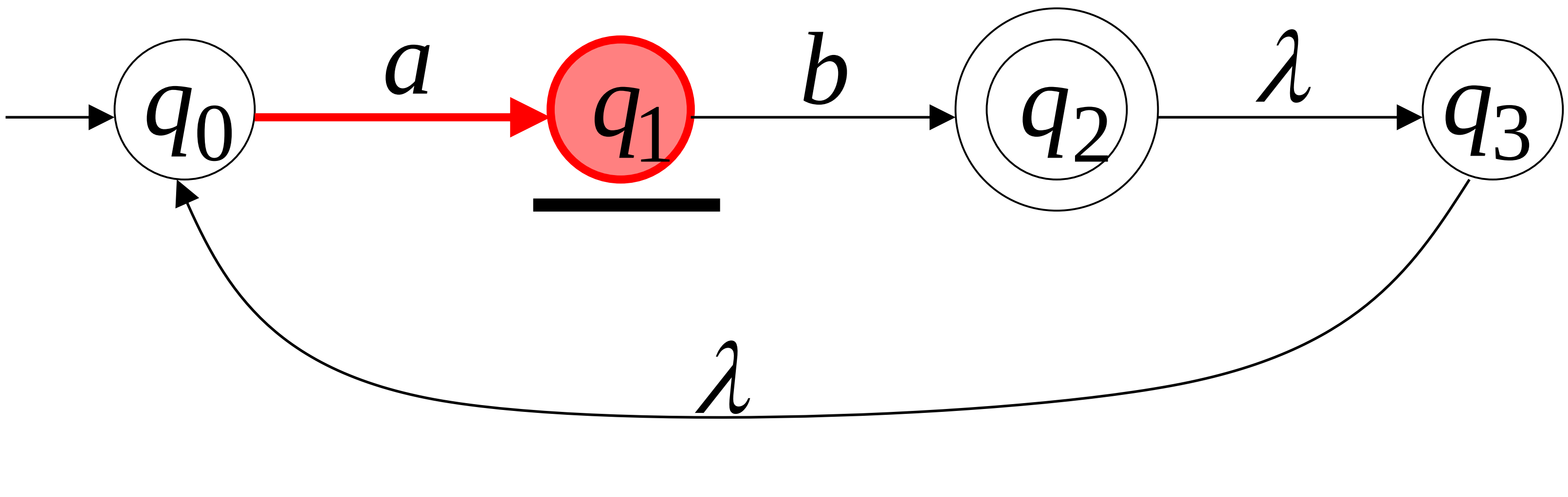

a
b
a
b

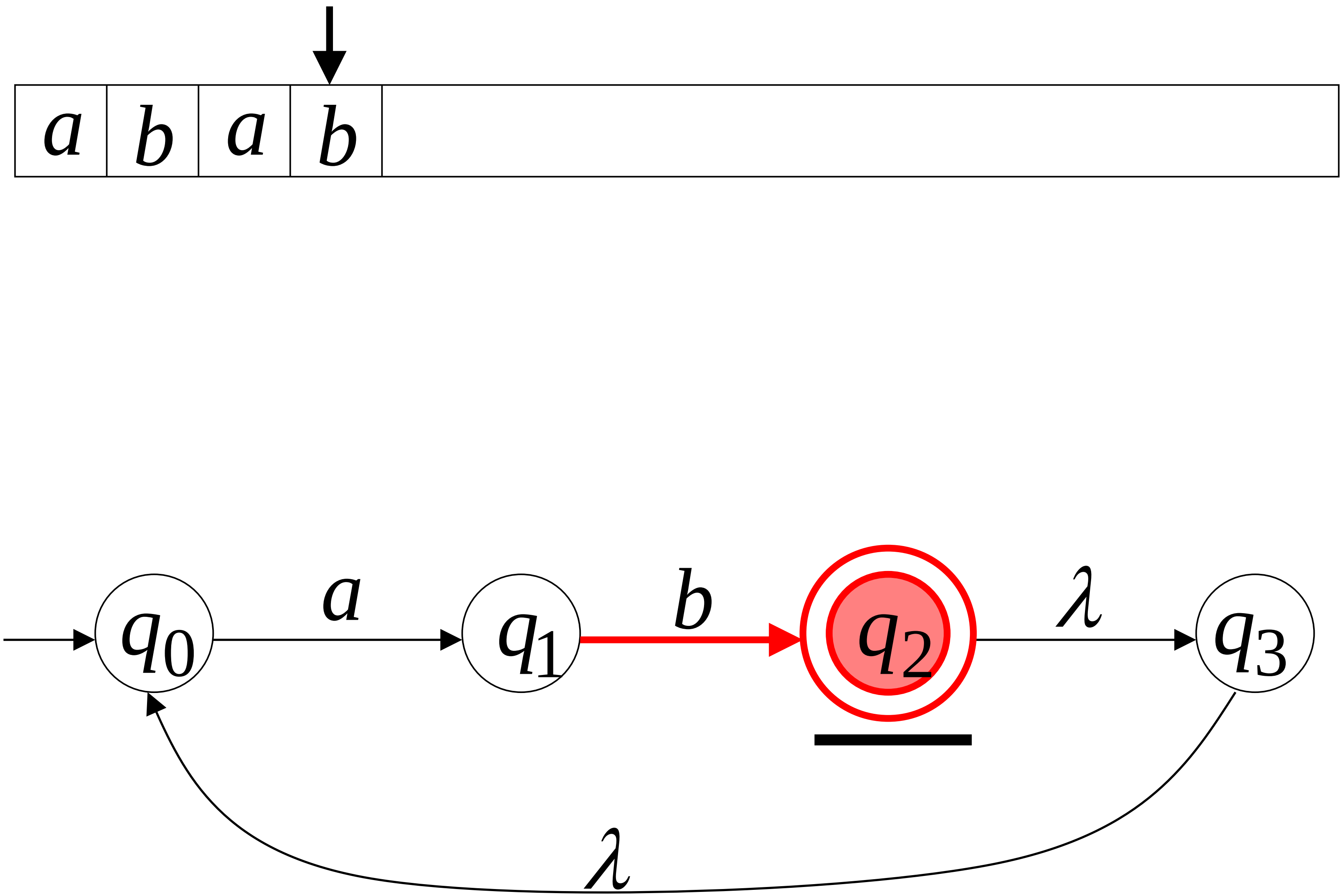

a
b
λ
λ
q_0
q_1
q_2
q_3

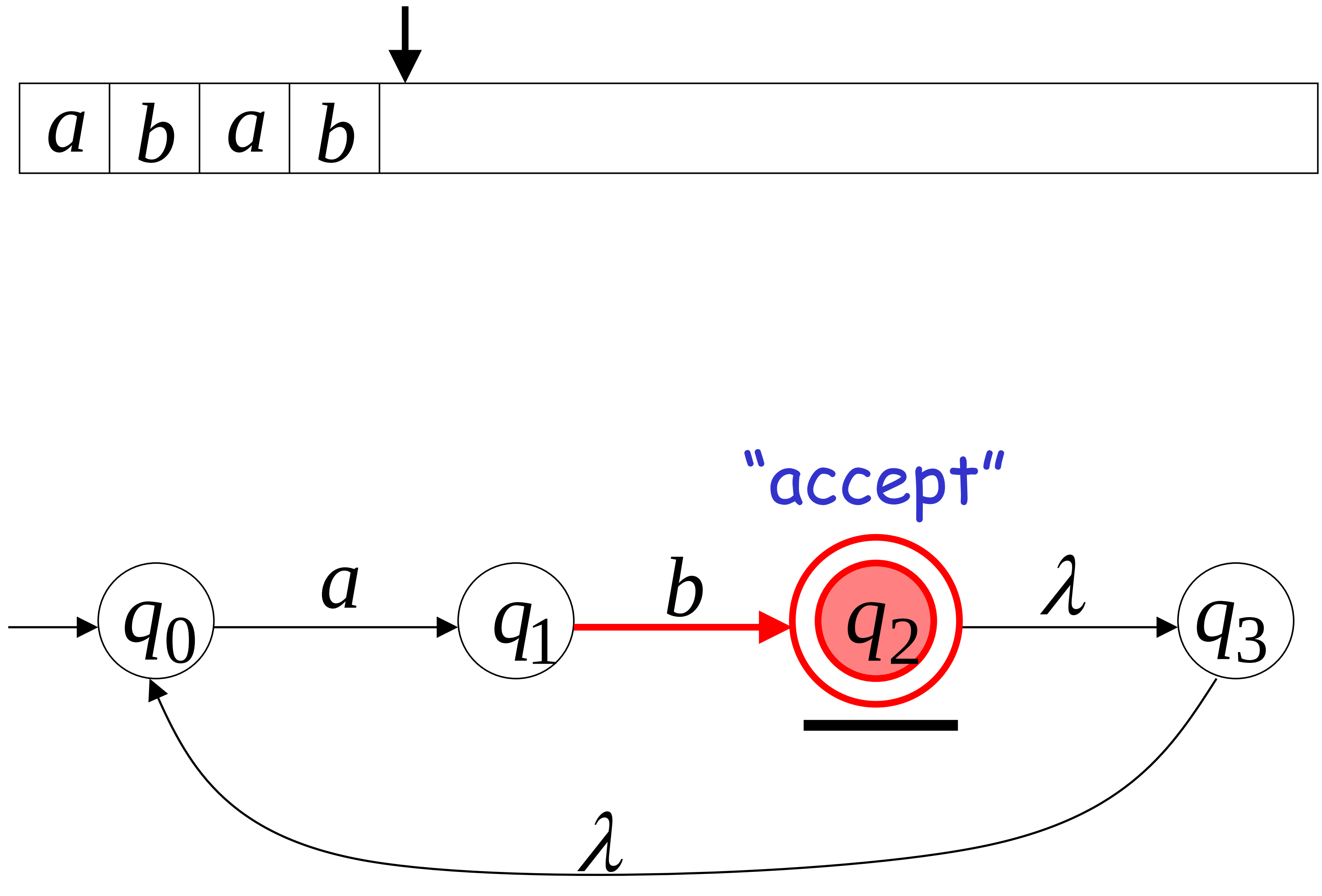


a
b
a
b
q_0
a
q_1
b
q_2
λ
q_3
λ


a
b
a
b


a
b
λ
λ
q_0
q_1
q_2
q_3

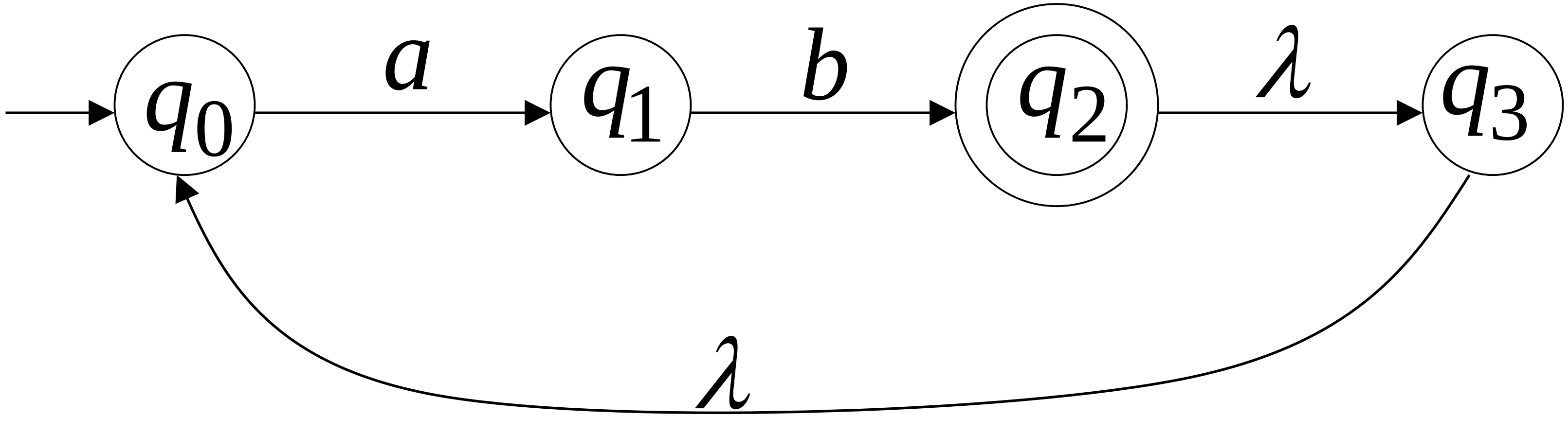

a
b
a
b
a
b
λ
q_0
q_1
q_2
q_3
λ


a
b
a
b
"accept"
a
b
λ
λ
q_0
q_1
q_2
q_3

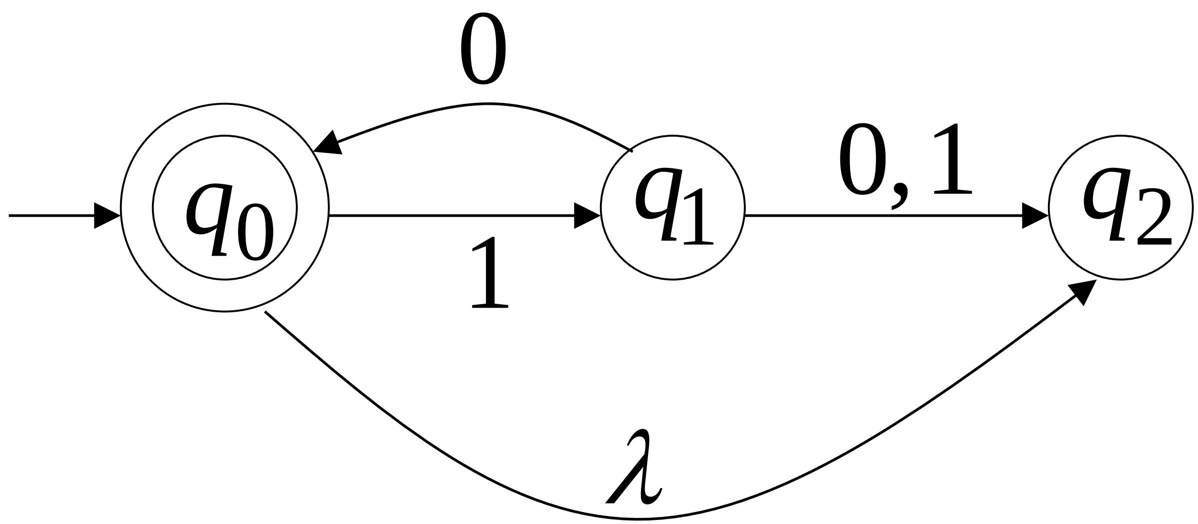
Language accepted

$$L = \{ab,\ abab,\ ababab,\ \ldots\}$$

$$= \{ab\}^{+}$$



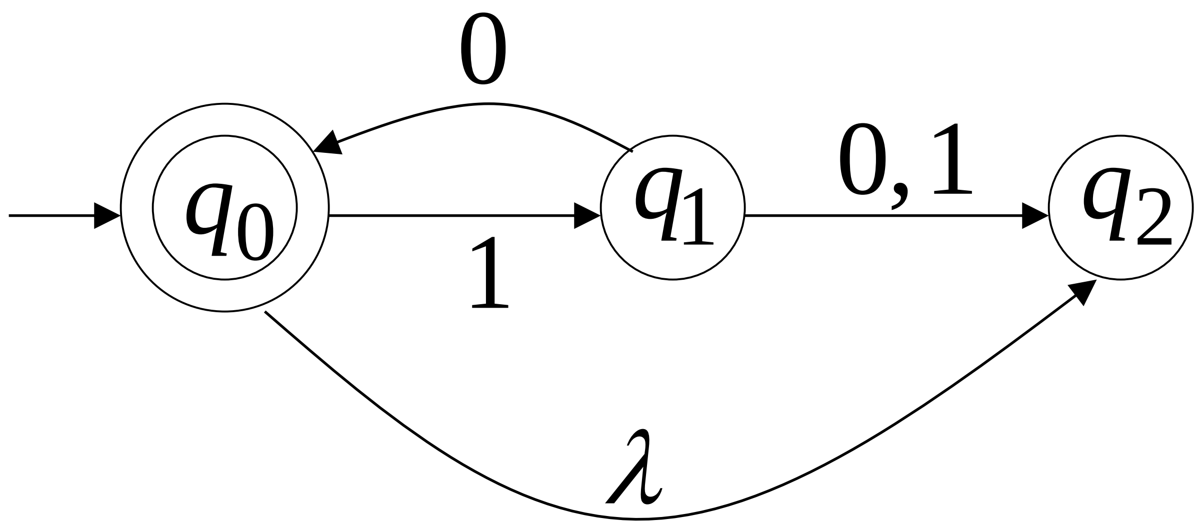
Another NFA Example: L(M)?



Language accepted

$$L(M) = \{\lambda,\ 10,\ 1010,\ 101010,\ \ldots\}$$

$$= \{10\}^*$$



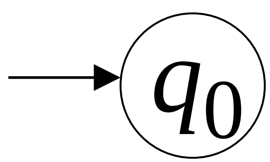
(redundant state)

Remarks:

- The λ symbol never appears on the input tape

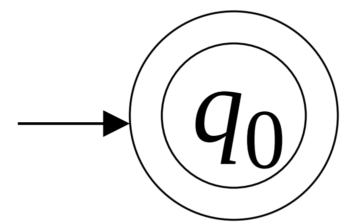
- Simple automata: Languages?

M_1



$L(M_1) = \{\}$

M_2



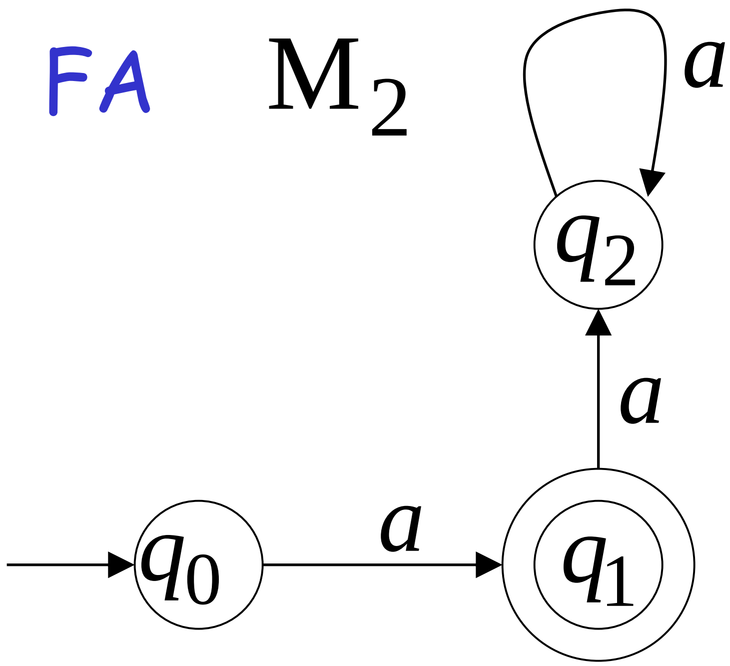
$L(M_2) = \{\lambda\}$

Is there any λ–transition in deterministic automata?

No….only NFA can have Lambda transition

- NFAs are interesting because we can express languages easier than FAs

FA M_2



$$L(M_2) = \{a\}$$

Formal Definition of NFAs

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

$Q:$ Set of states, i.e. $\{q_0, q_1, q_2\}$

$\Sigma:$ Input alphabet, i.e. $\{a, b\}$

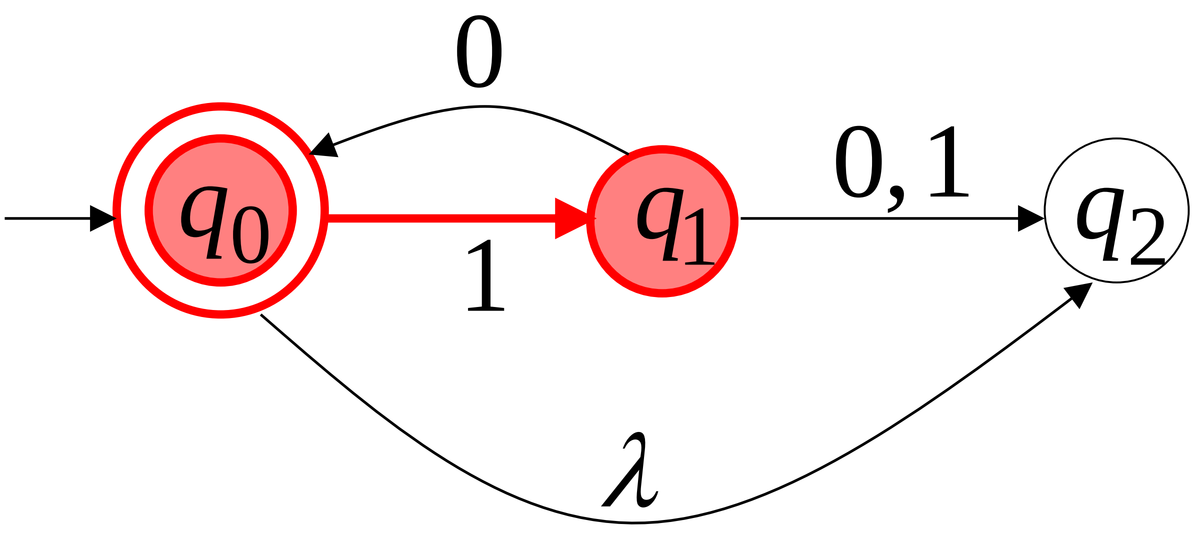
$\delta:$ Transition function

$q_0:$ Initial state

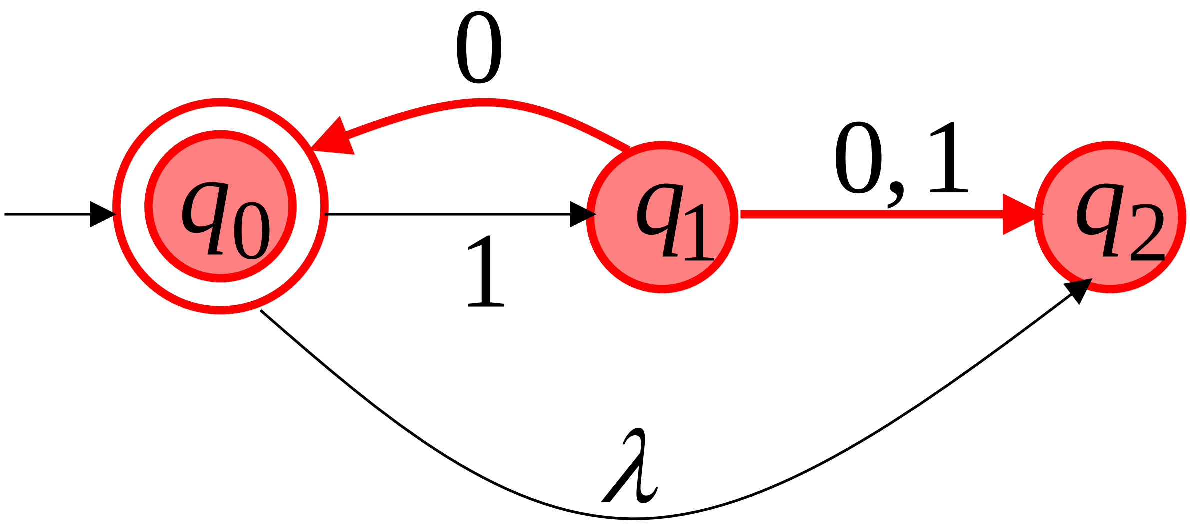
$F:$ Accepting states

Transition Function δ

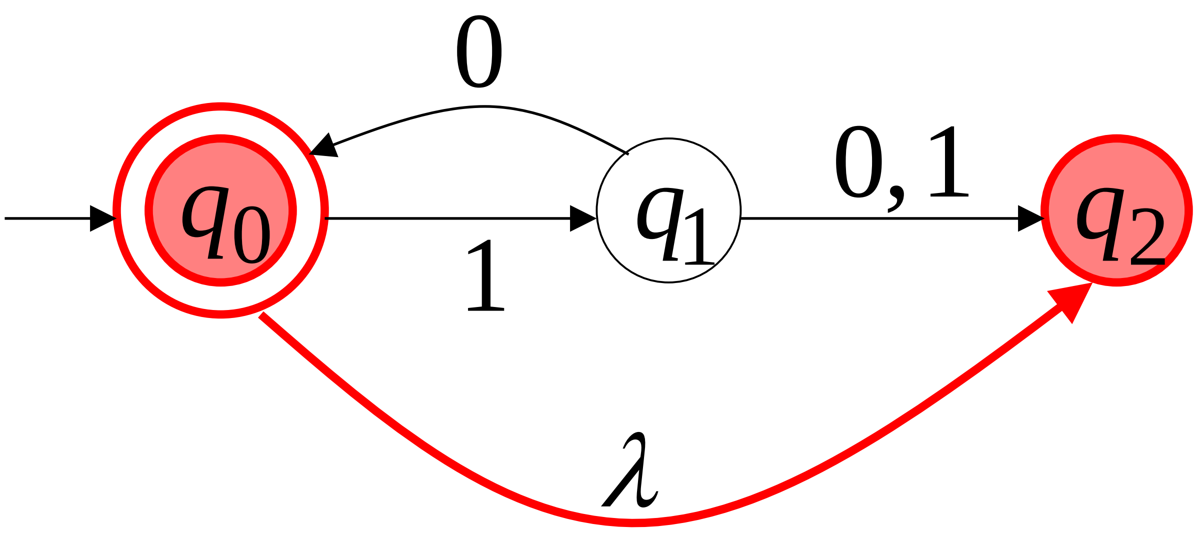
$$\delta(q_0, 1) = \{q_1\}$$



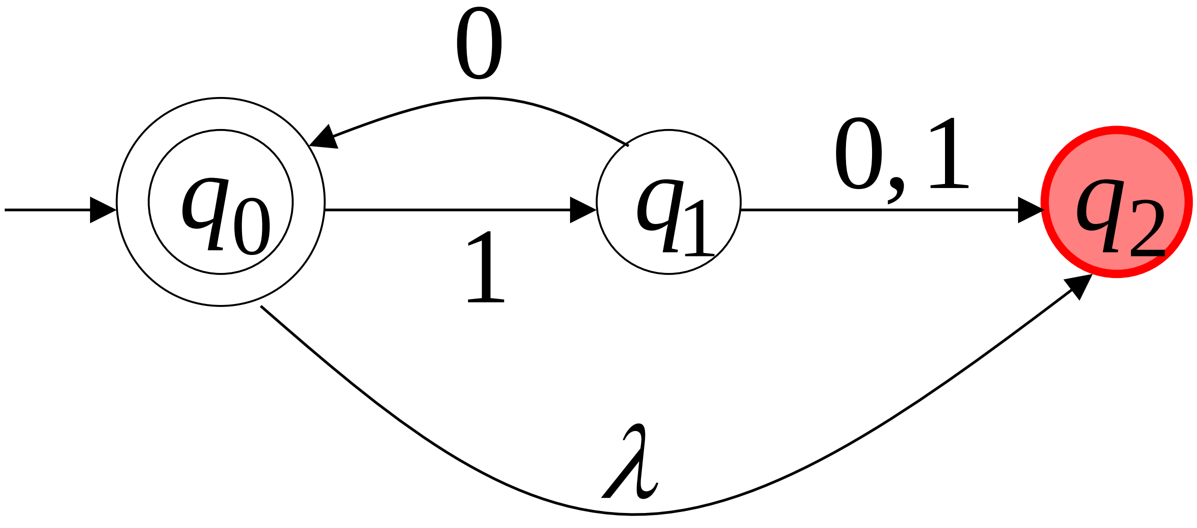
$$\delta(q_1, 0) = \{q_0, q_2\}$$



$$\delta(q_0, \lambda) = \{q_0, q_2\}$$

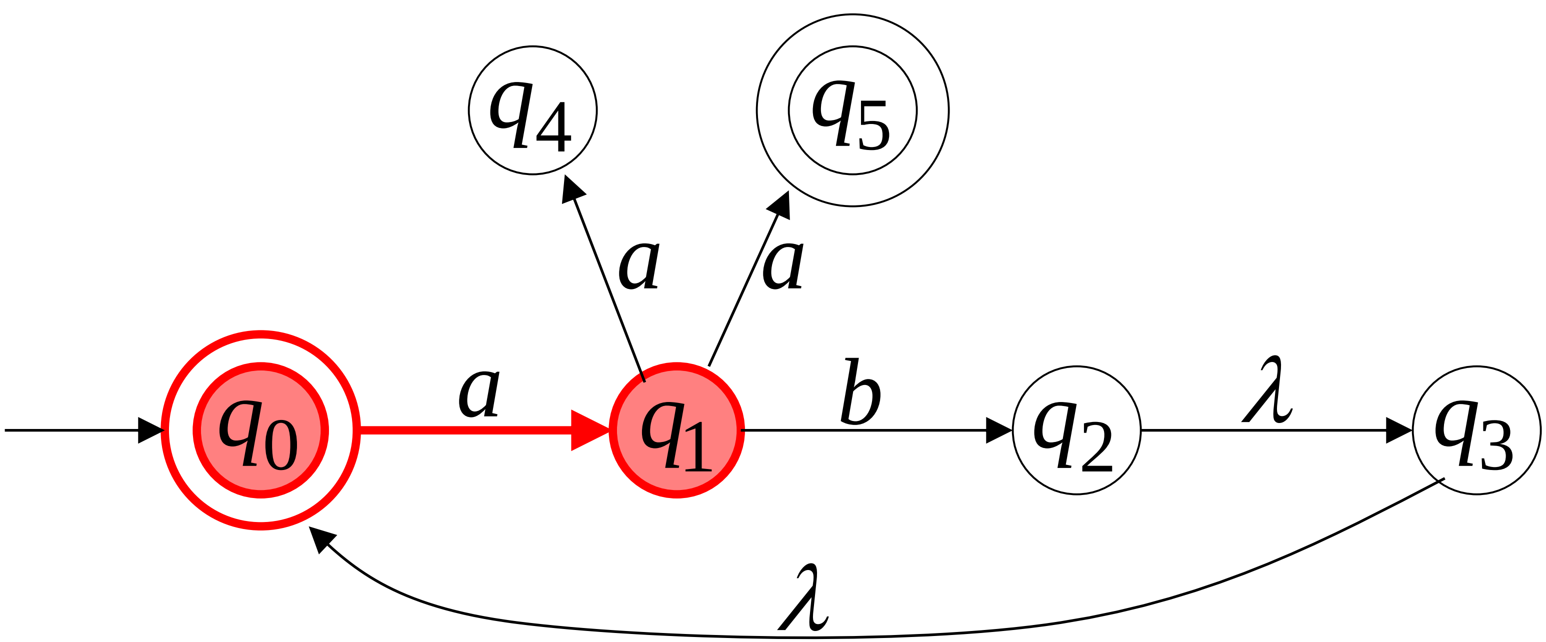


$$\delta(q_2, 1) = \varnothing$$

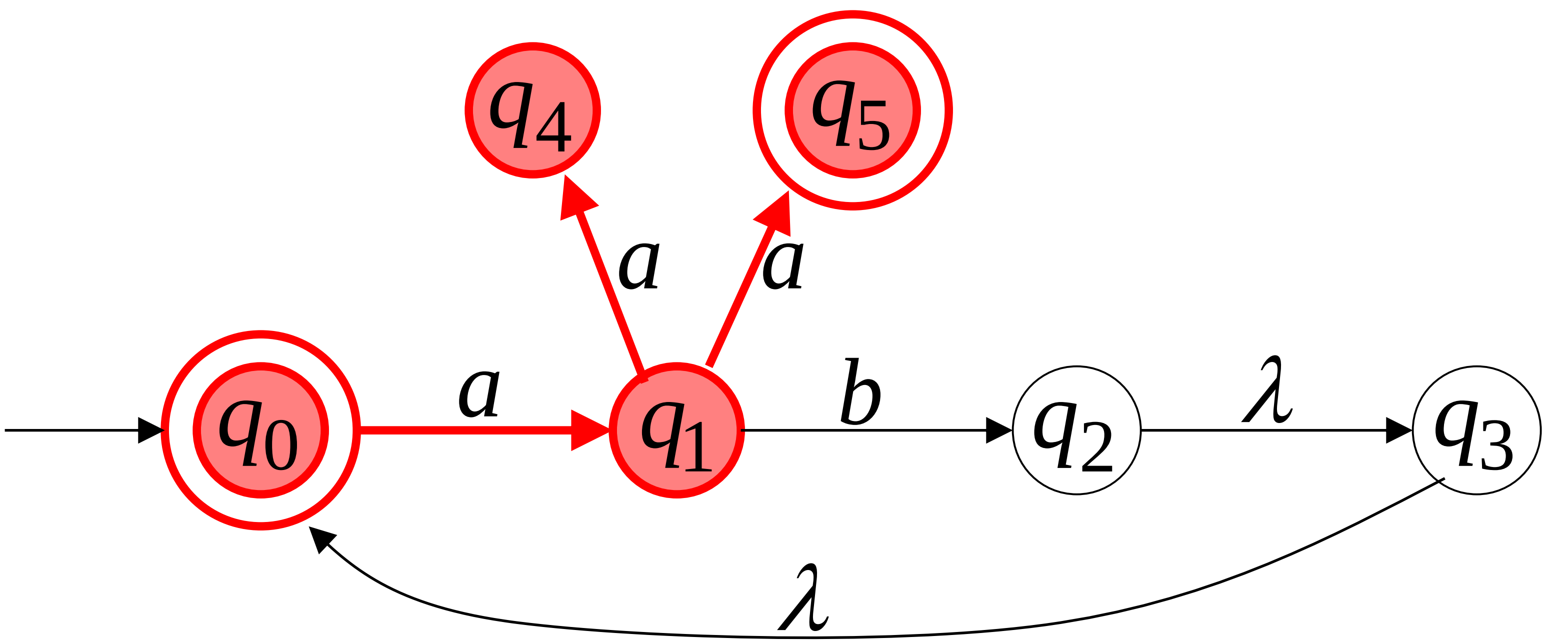


Extended Transition Function δ^*

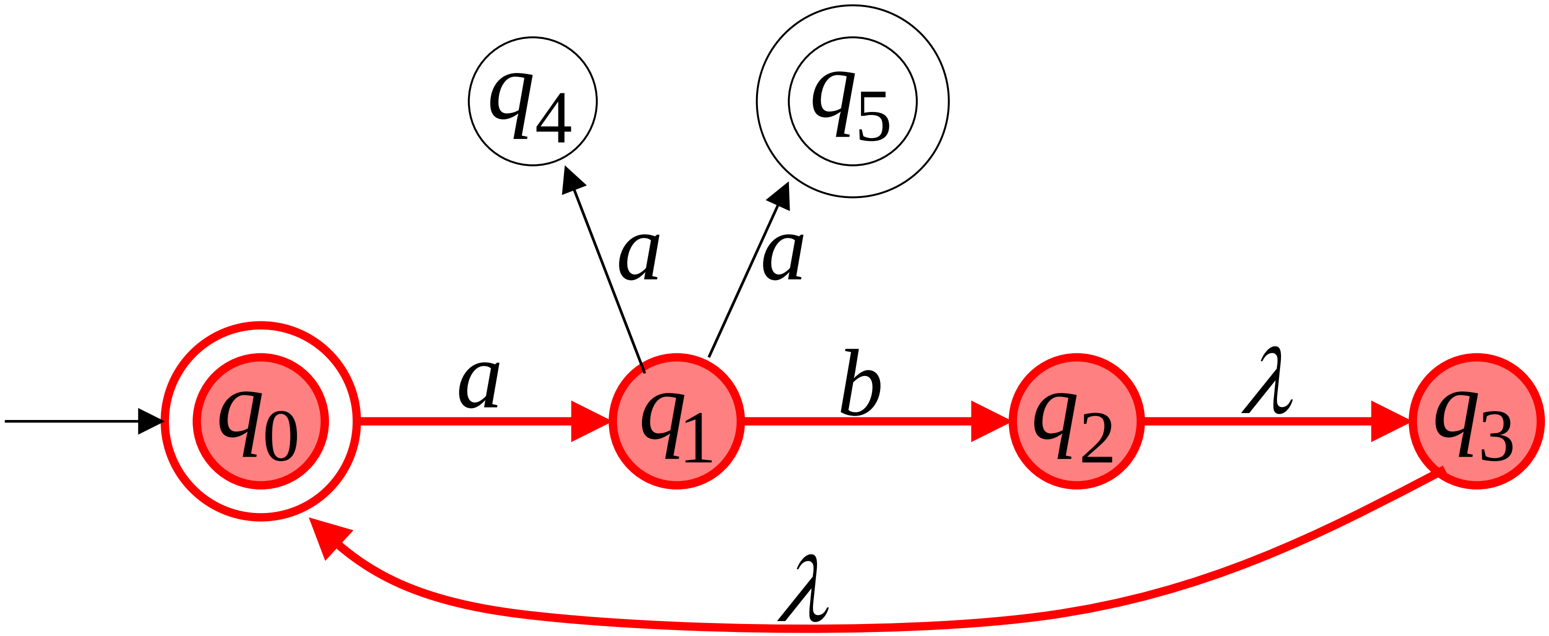
$$\delta^*(q_0, a) = \{q_1\}$$



$$\delta^*(q_0, aa) = \{q_4, q_5\}$$

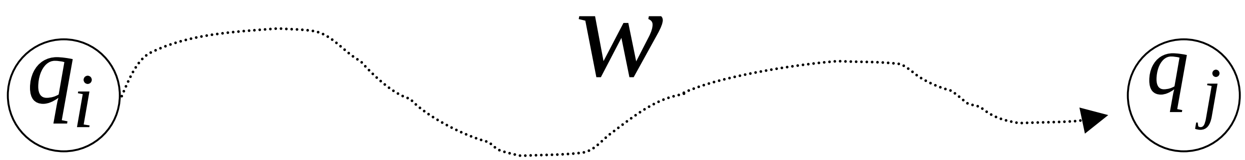


$$\delta^*(q_0, ab) = \{q_2, q_3, q_0\}$$

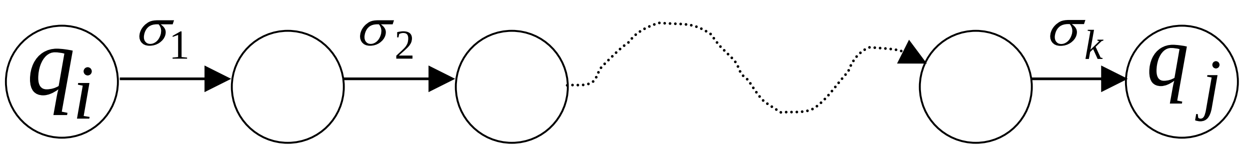


Formally

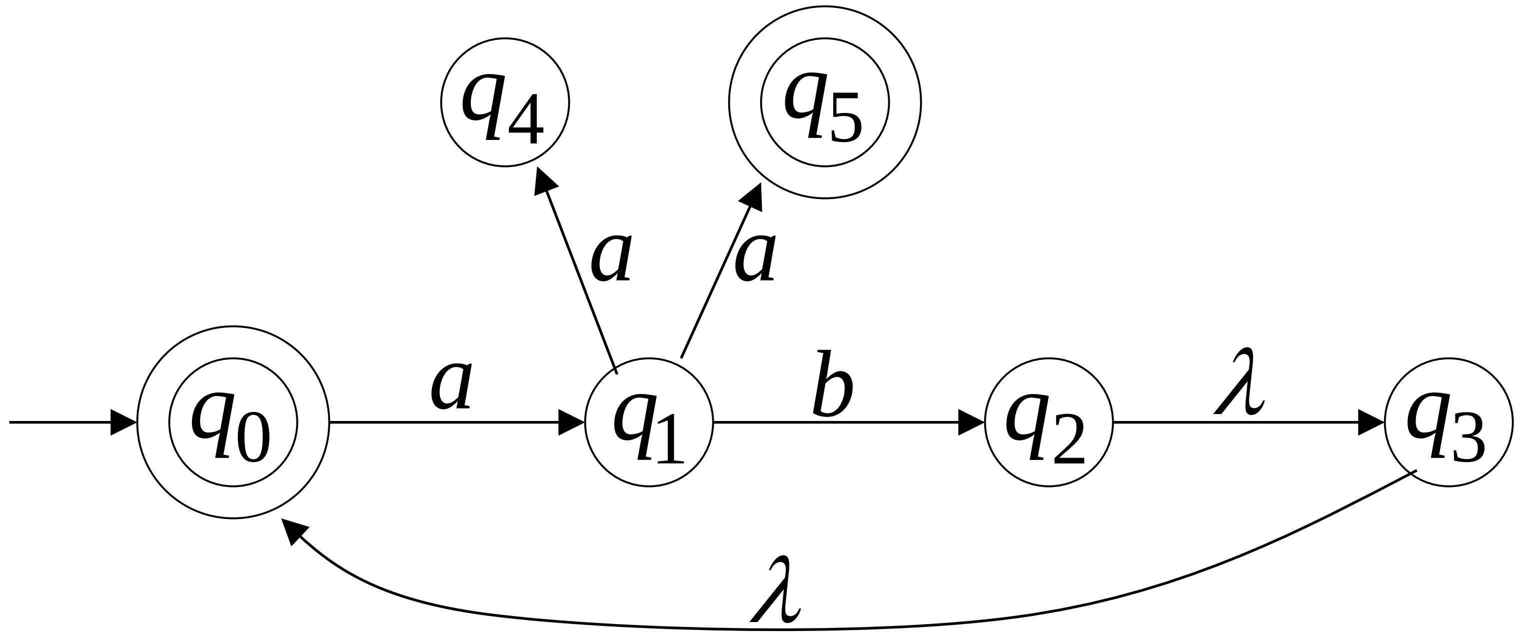
$q_j \in \delta^*(q_i, w)$: there is a walk from q_i to q_j with label w



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

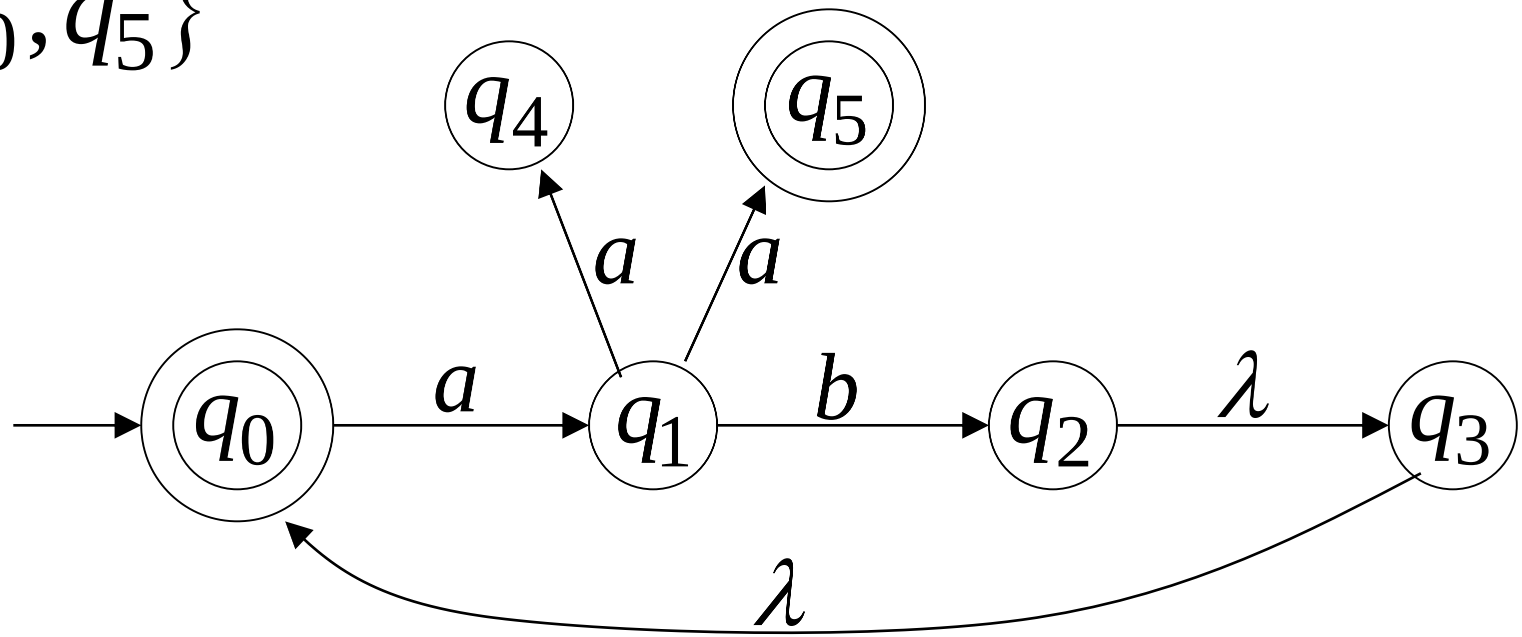


L(M)?



The Language of an NFA M

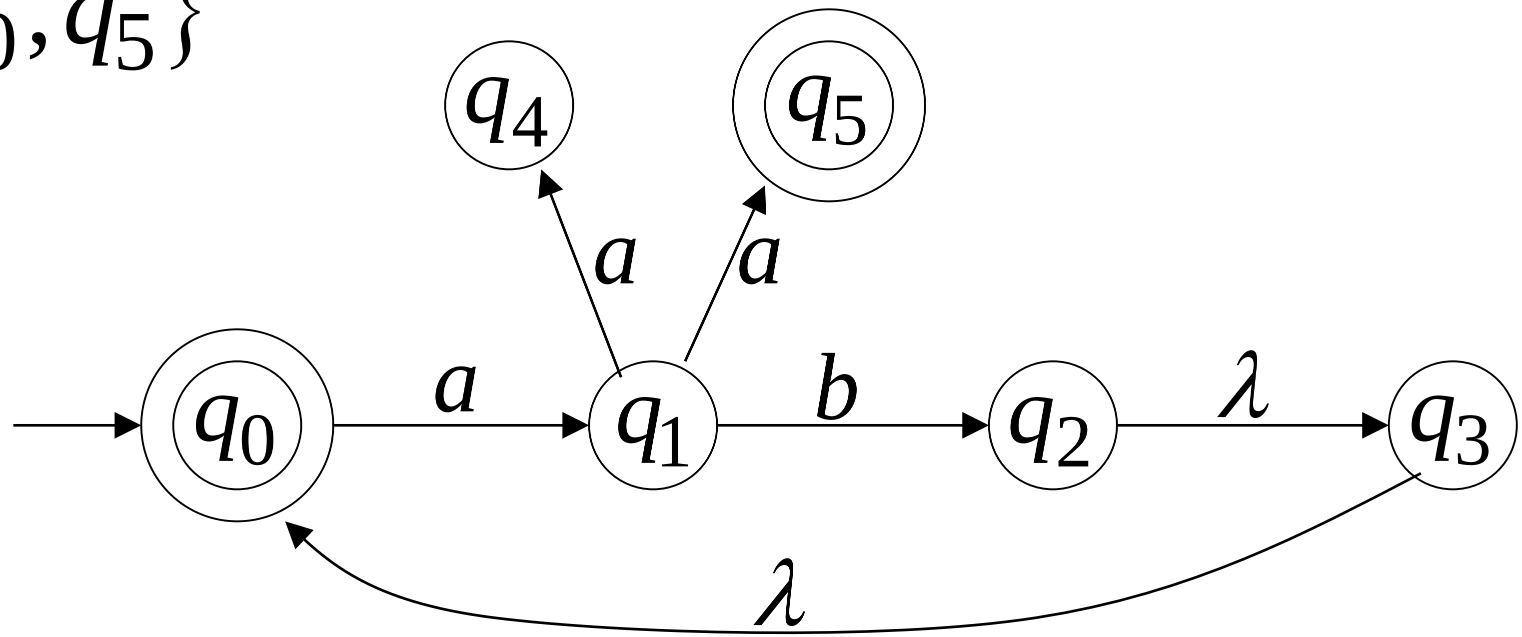
$F = \{q_0, q_5\}$



$\delta^*(q_0, aa) = \{q_4, \underline{q_5}\}$ $\qquad aa \in L(M)$

$\in F$

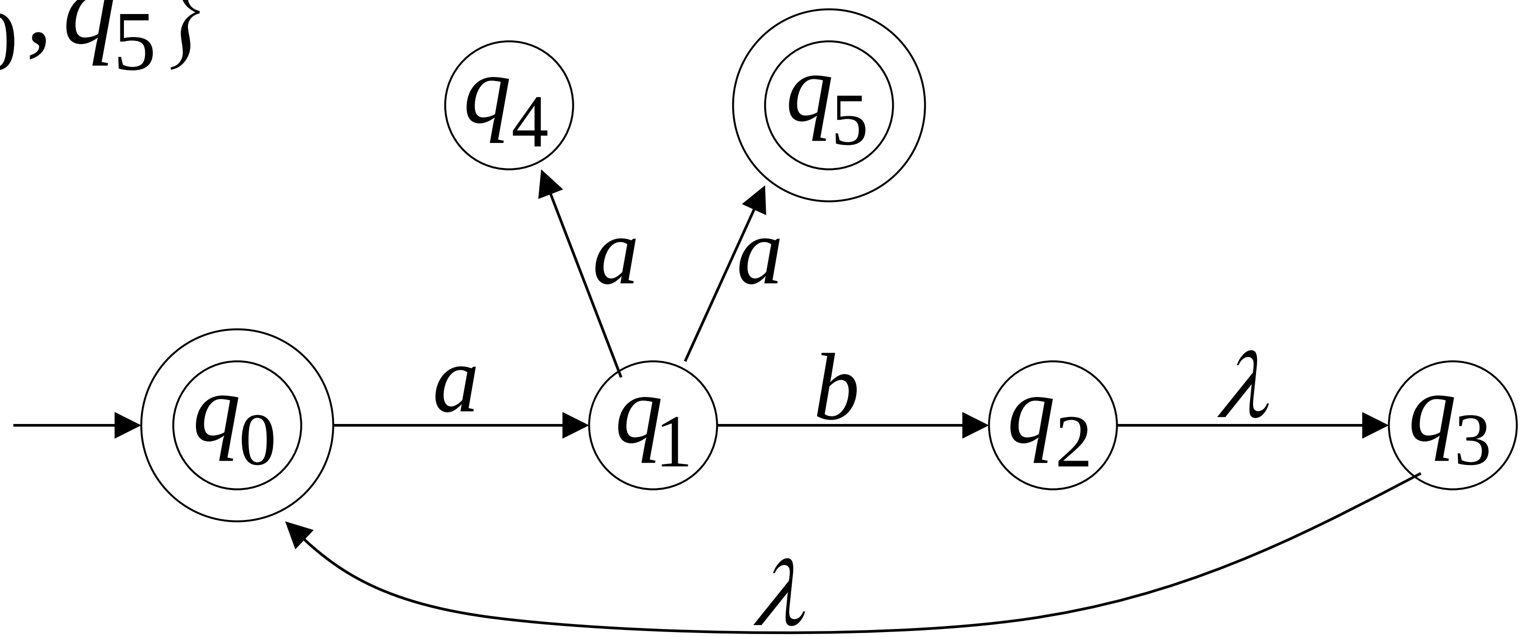
$$F = \{q_0, q_5\}$$



$$\delta^*(q_0, ab) = \{q_2, q_3, \underline{q_0}\} \qquad ab \in L(M)$$

$$\in F$$

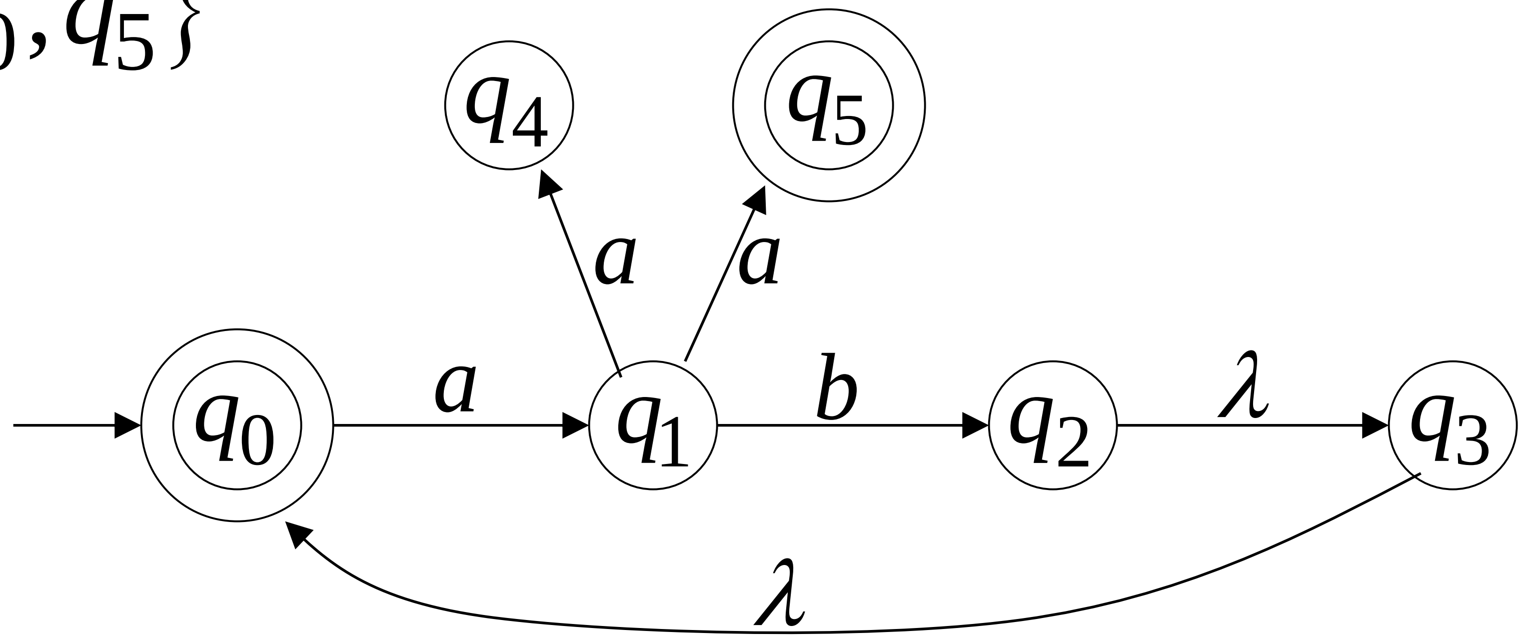
$$F = \{q_0, q_5\}$$



$$\delta^*(q_0, abaa) = \{q_4, \underline{q_5}\} \qquad aaba \in L(M)$$

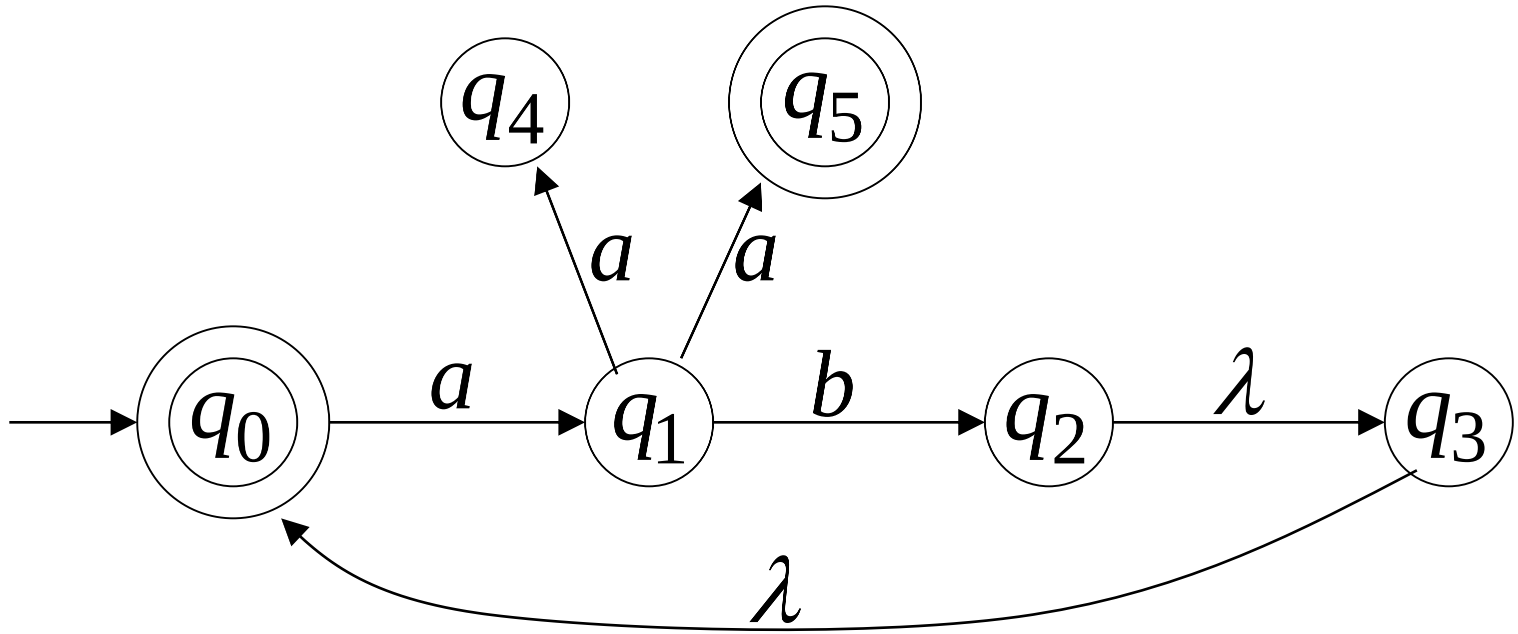
$$\in F$$

$$F = \{q_0, q_5\}$$



$$\delta^*(q_0, aba) = \{q_1\} \qquad aba \notin L(M)$$

$$\notin F$$



$$L(M) = \{\lambda\} \cup \{ab\}^* \{aa\}$$

Formally

The language accepted by NFA M is:

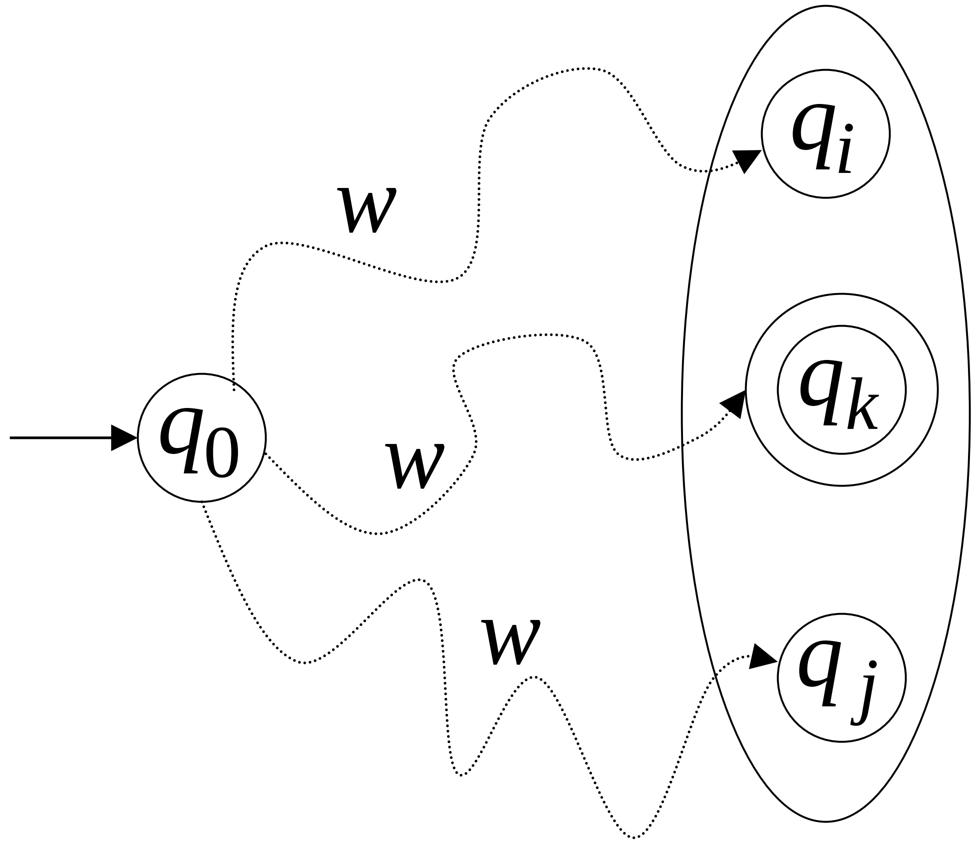
$$L(M) = \{w_1, w_2, w_3, \ldots\}$$

where $\delta^*(q_0, w_m) = \{q_i, q_j, \ldots, q_k, \ldots\}$

and there is some $q_k \in F$ (accepting state)

$w \in L(M)$

$\delta^*(q_0, w)$



$q_k \in F$