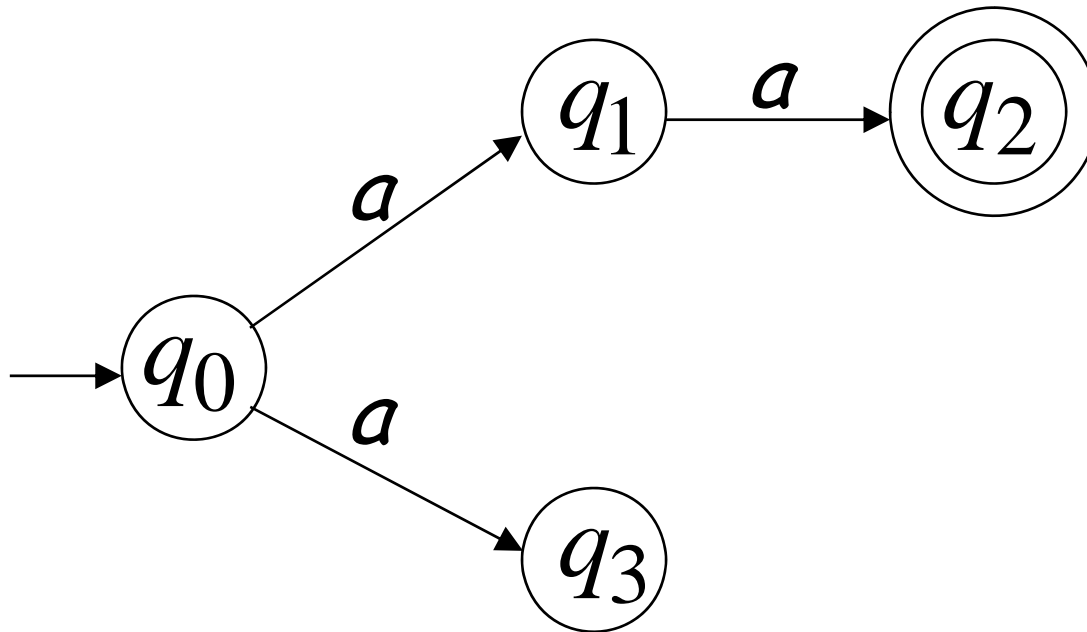


# Non Deterministic Automata

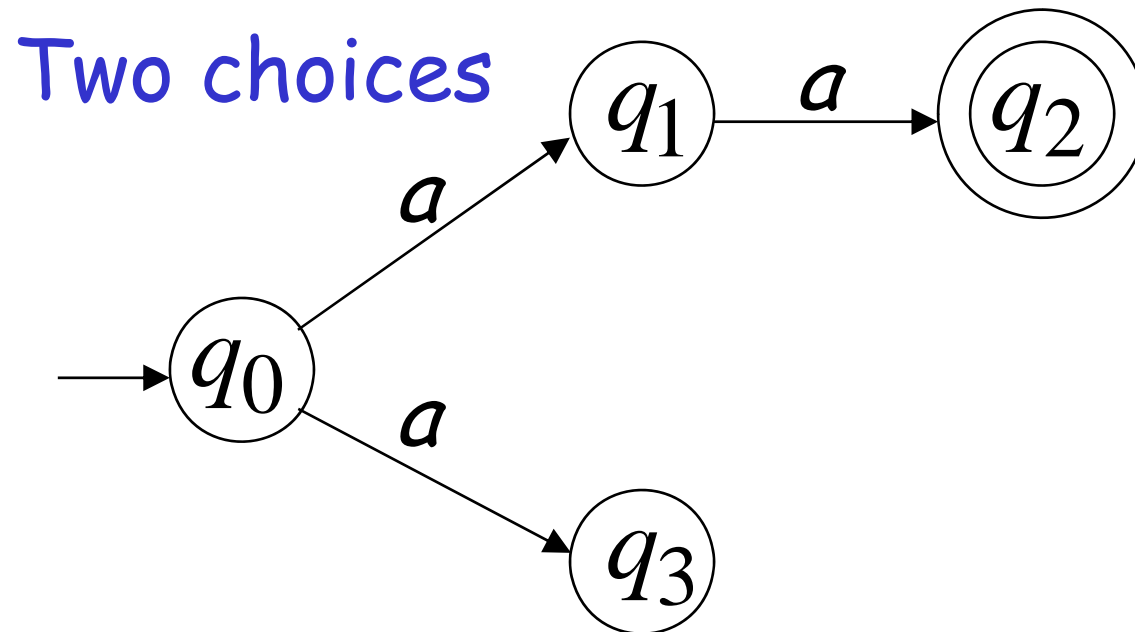
# Nondeterministic Finite Acceptor (NFA)

Alphabet =  $\{a\}$



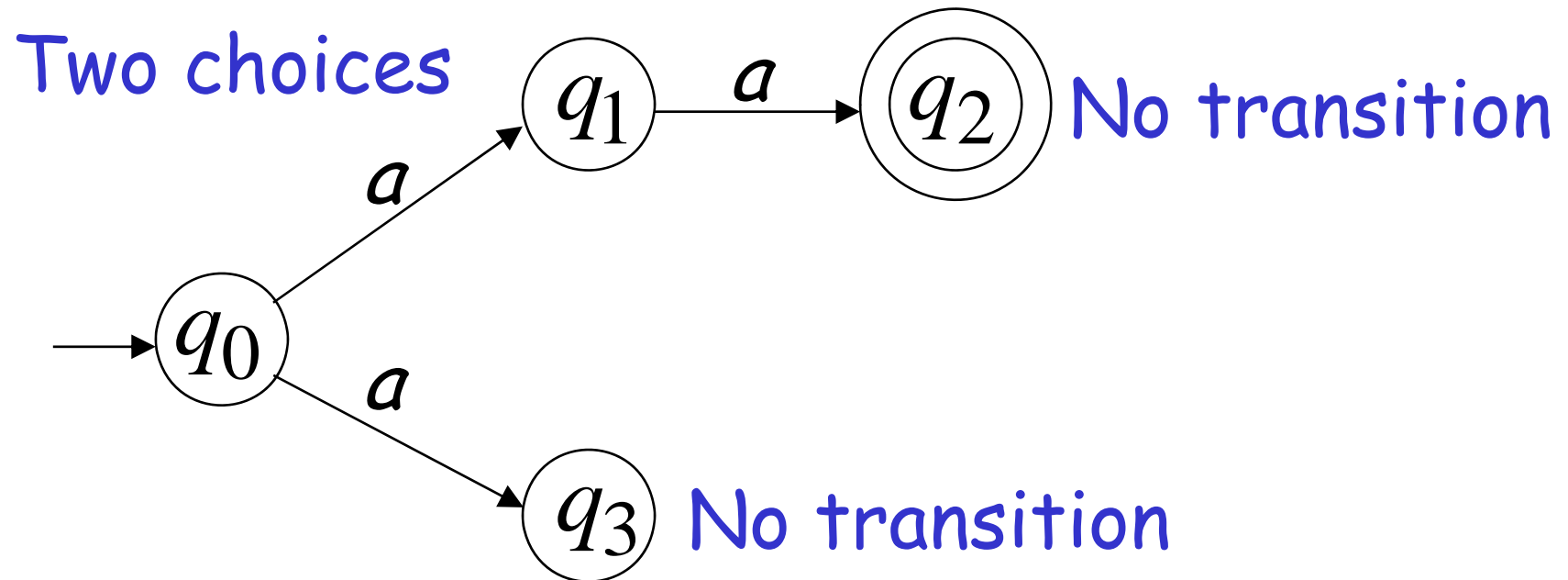
# Nondeterministic Finite Acceptor (NFA)

Alphabet =  $\{a\}$

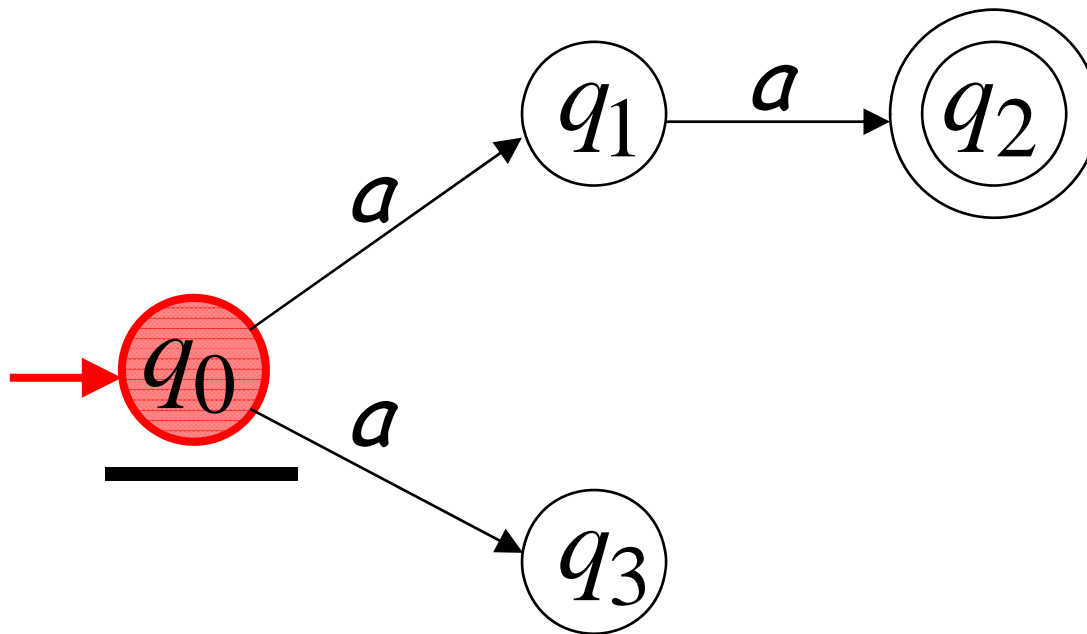
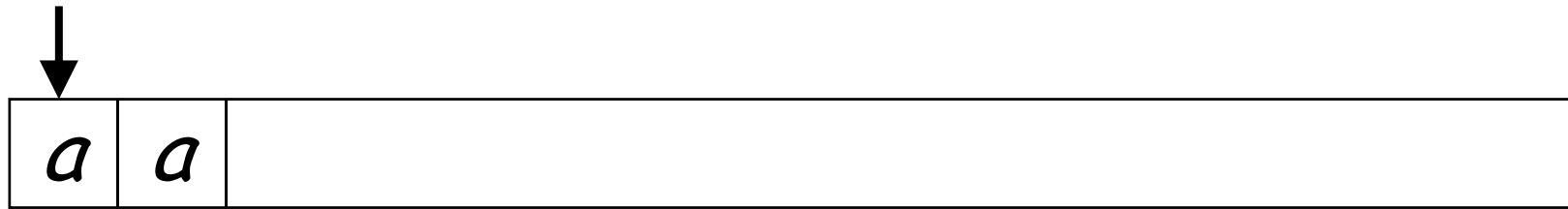


# Nondeterministic Finite Acceptor (NFA)

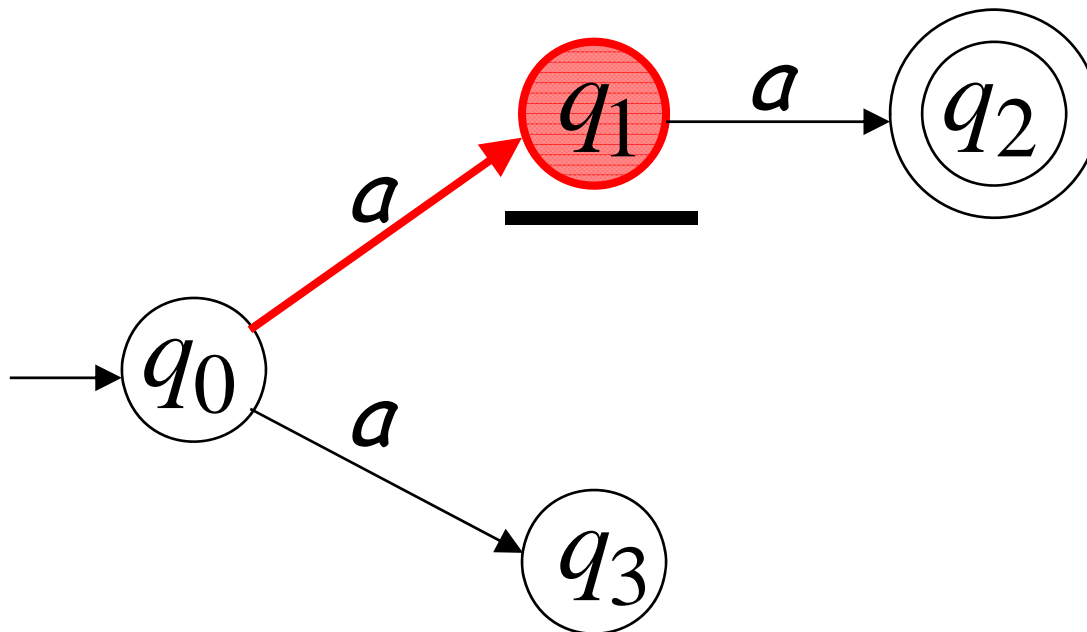
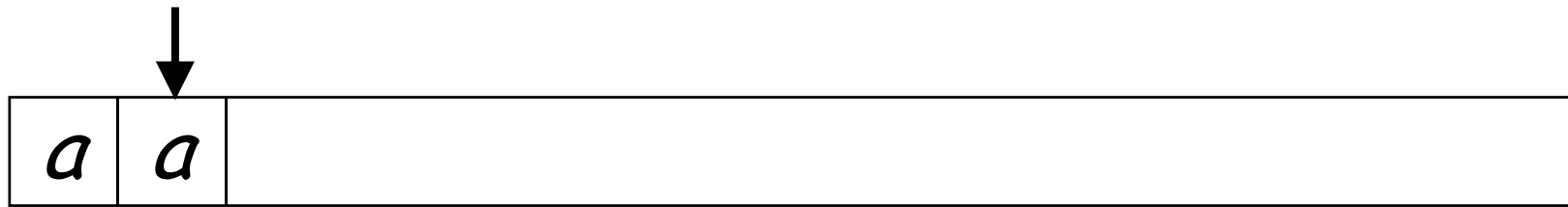
Alphabet =  $\{a\}$



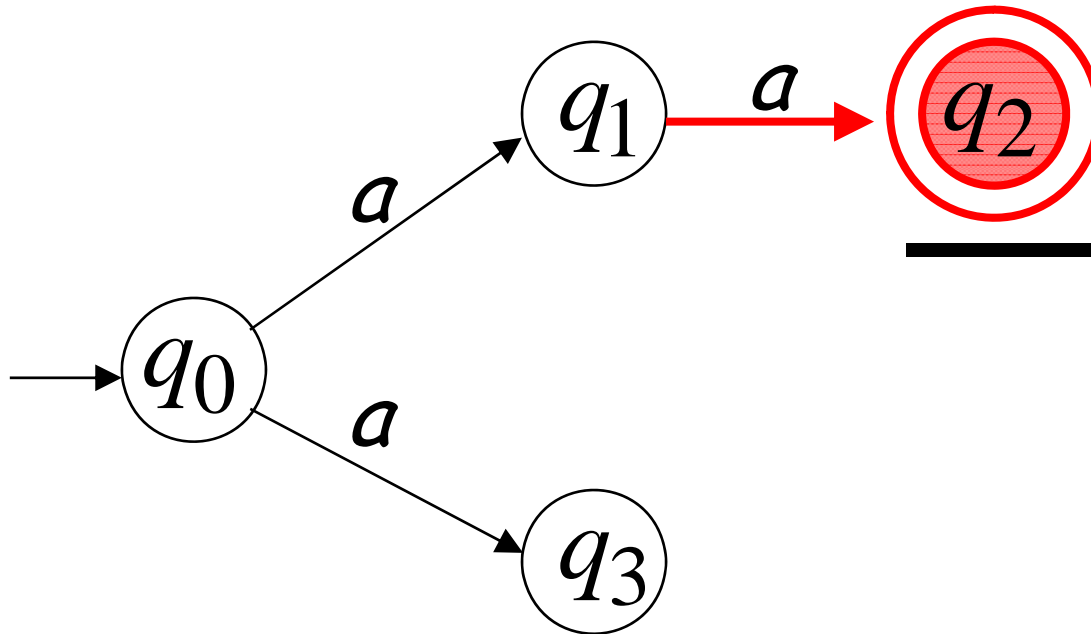
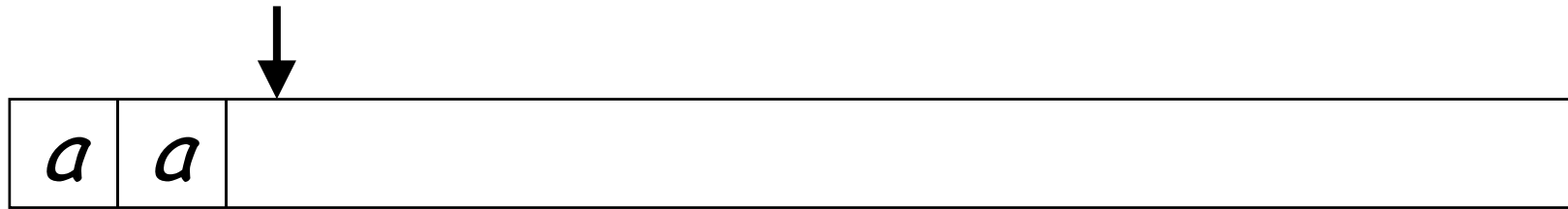
# First Choice



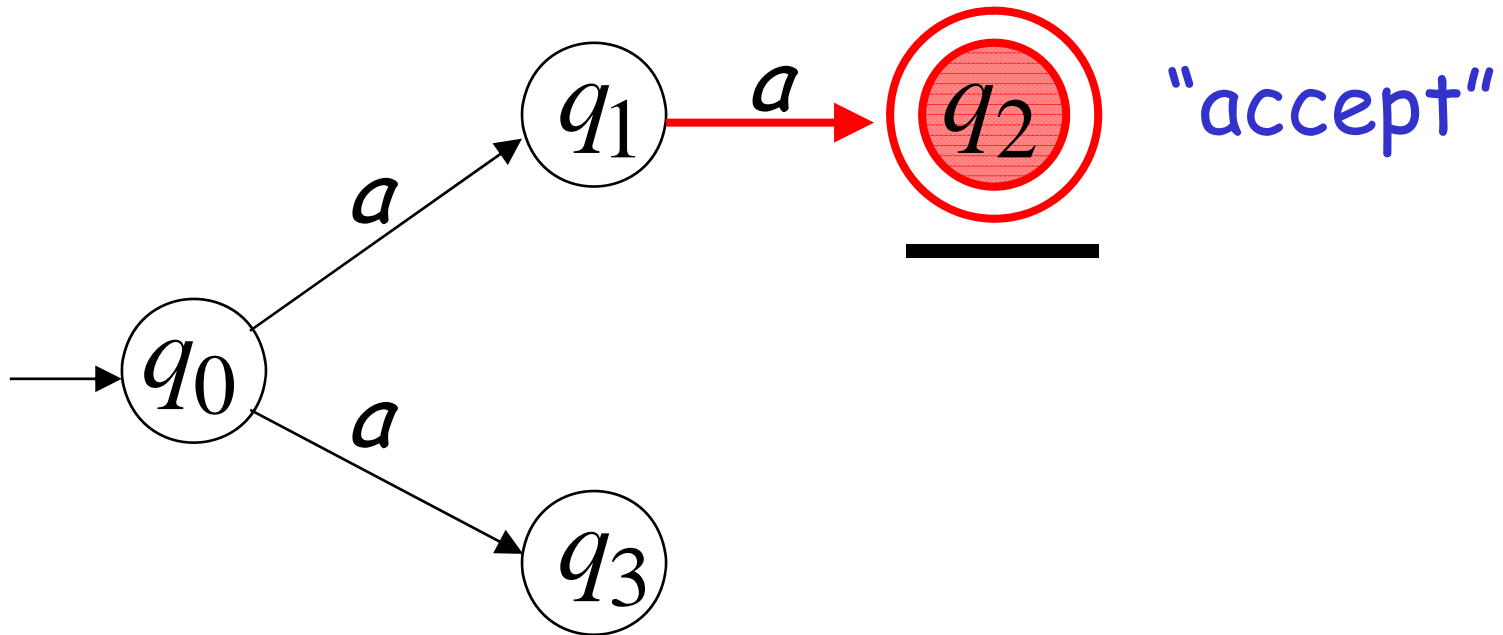
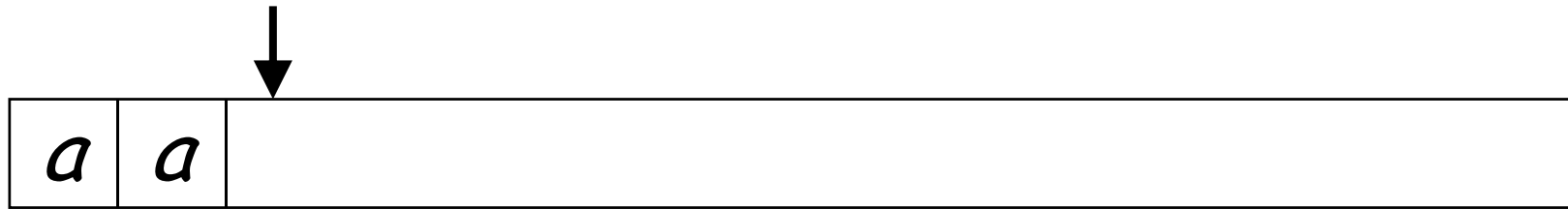
# First Choice



# First Choice

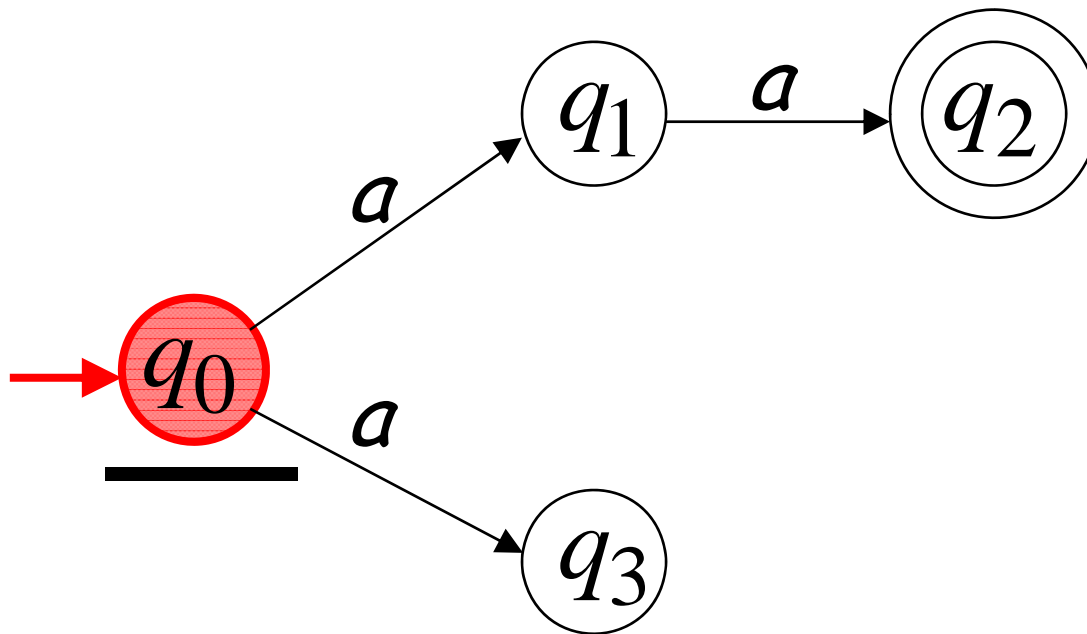
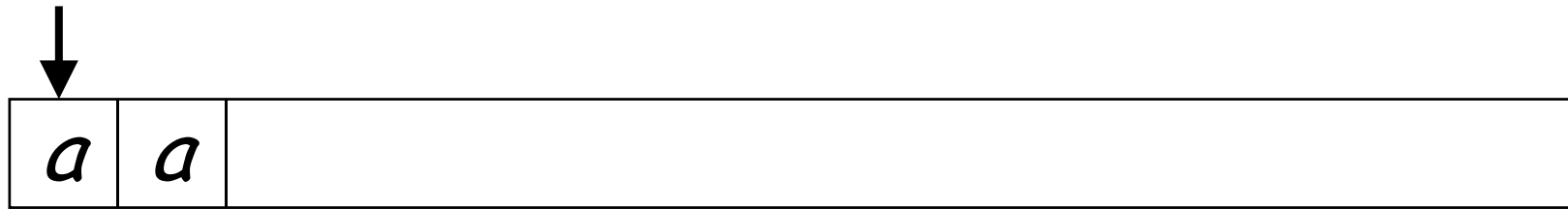


# First Choice

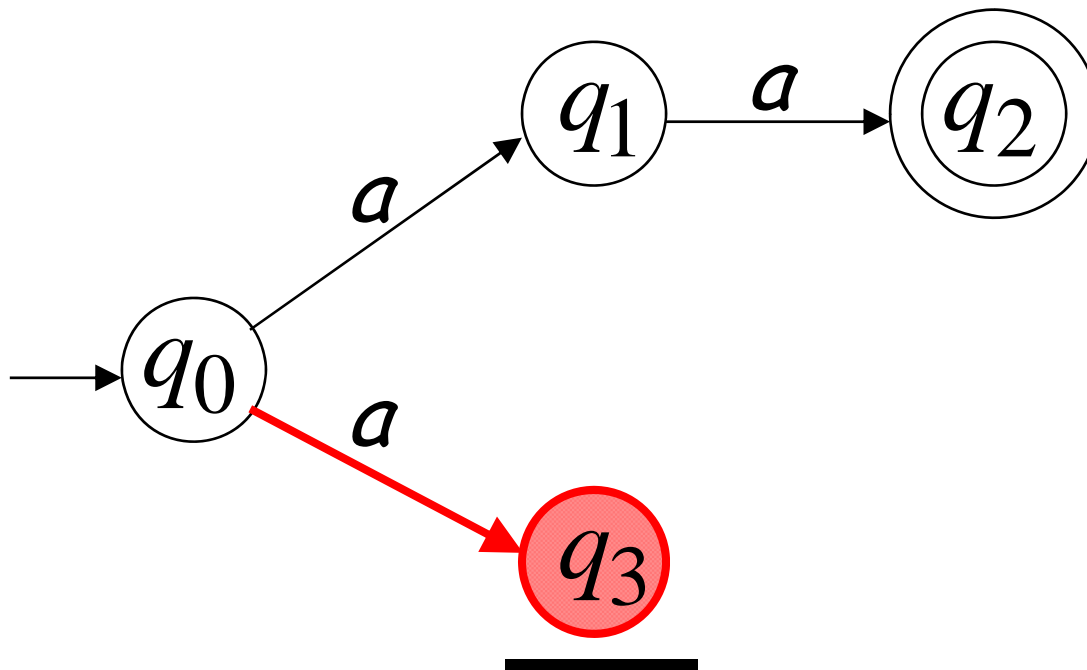
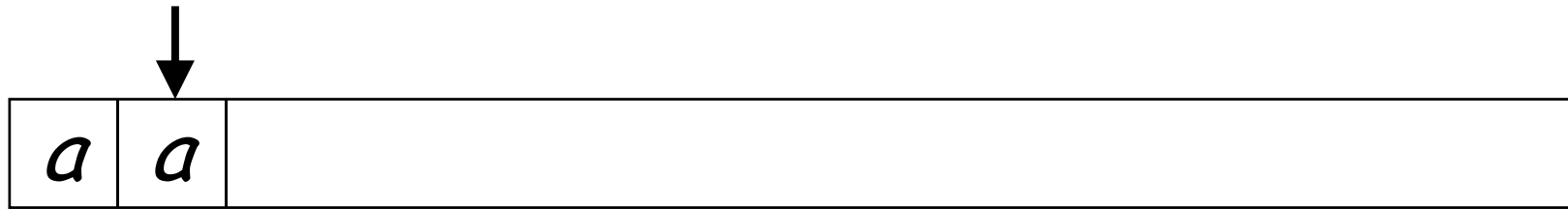




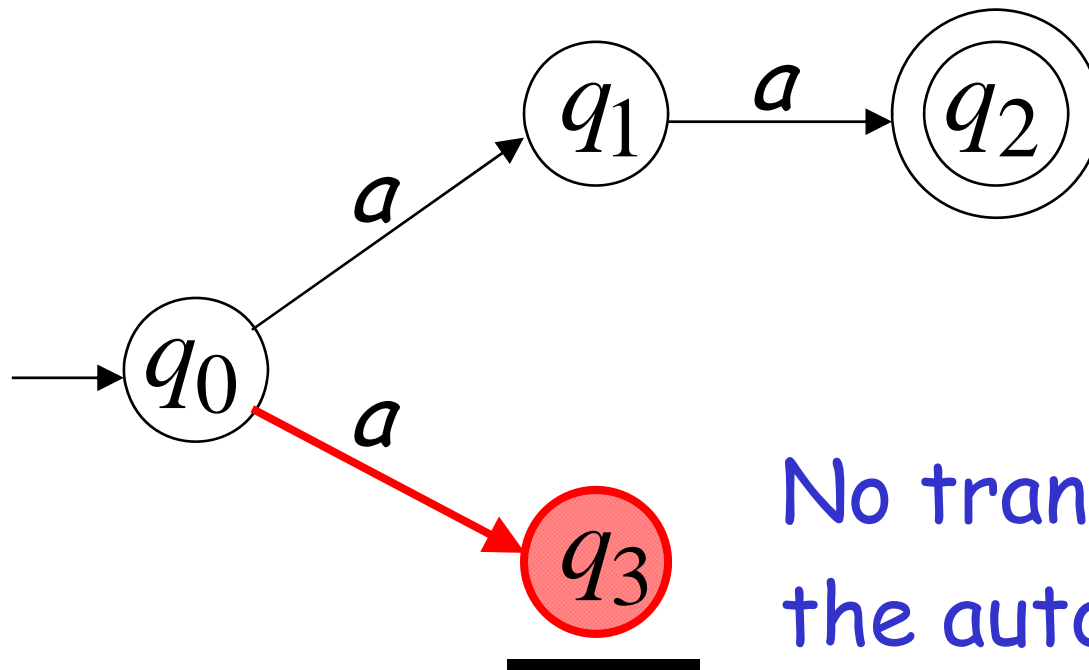
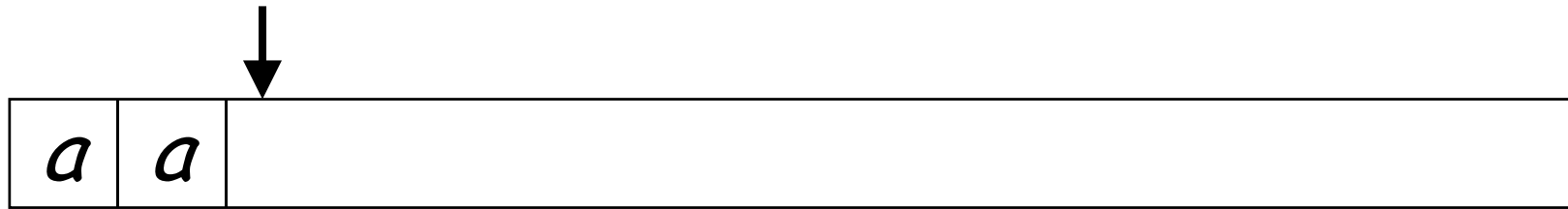
# Second Choice



# Second Choice

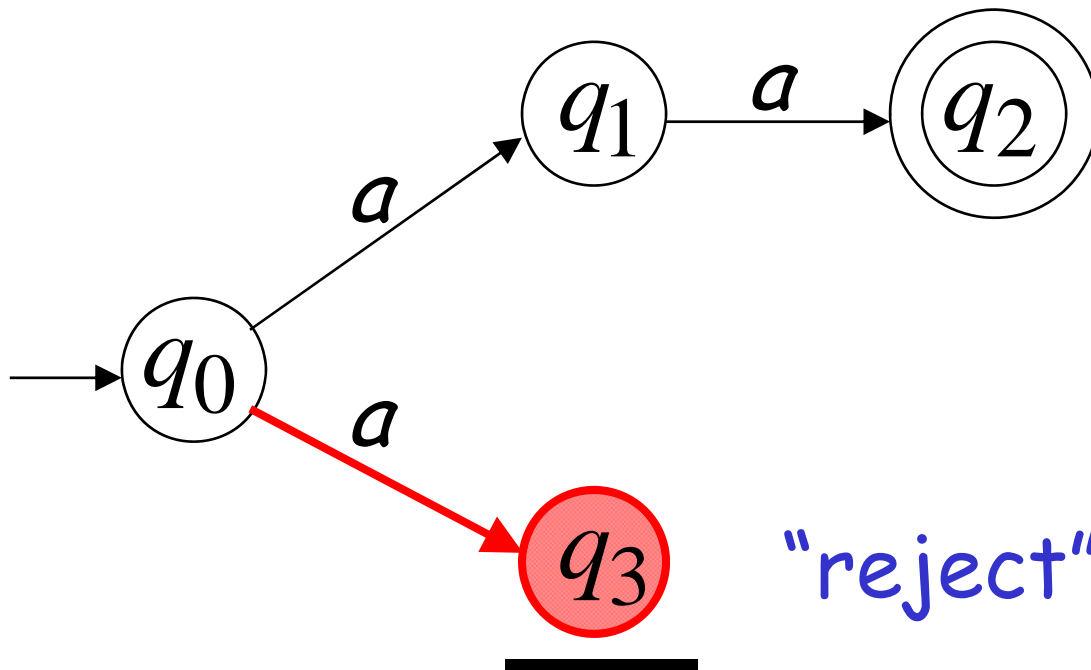
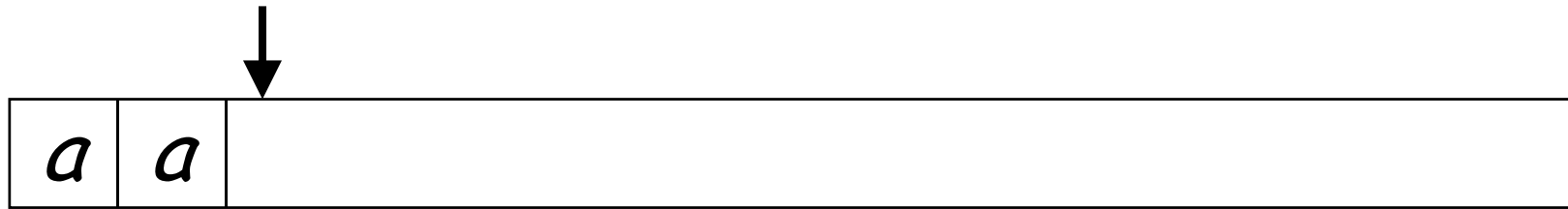


# Second Choice



No transition:  
the automaton hangs

# Second Choice



# Observation

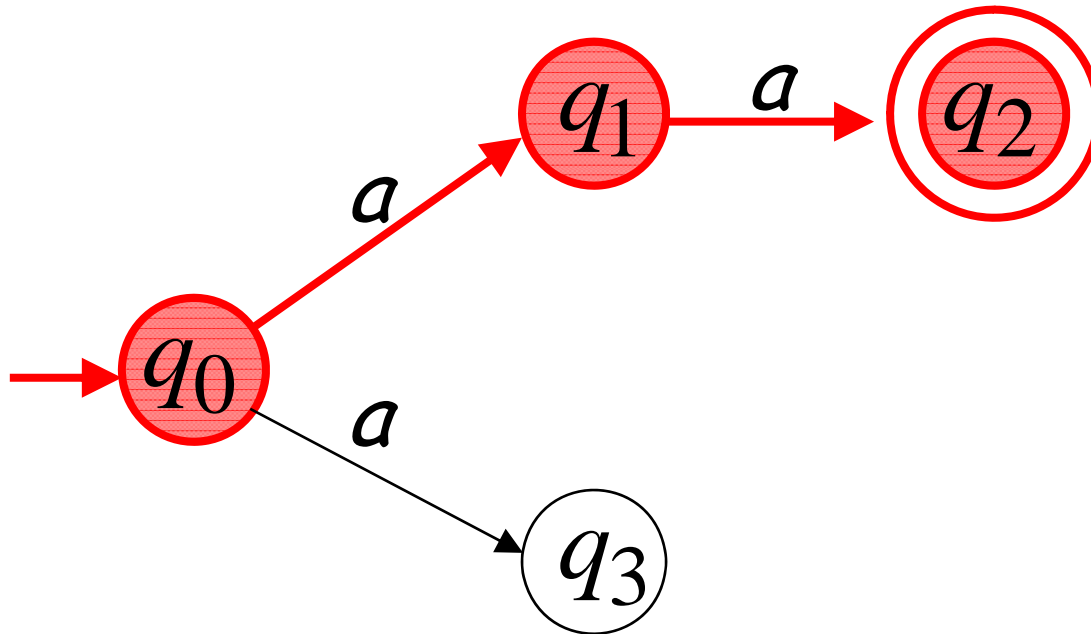
An NFA accepts a string

if

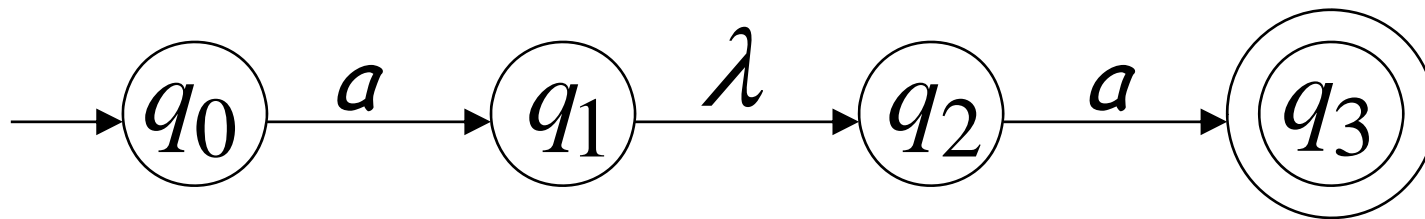
there is a computation of the NFA  
that accepts the string

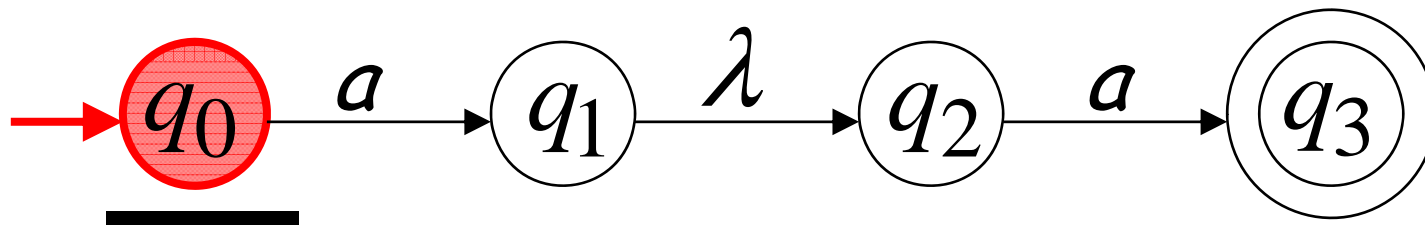
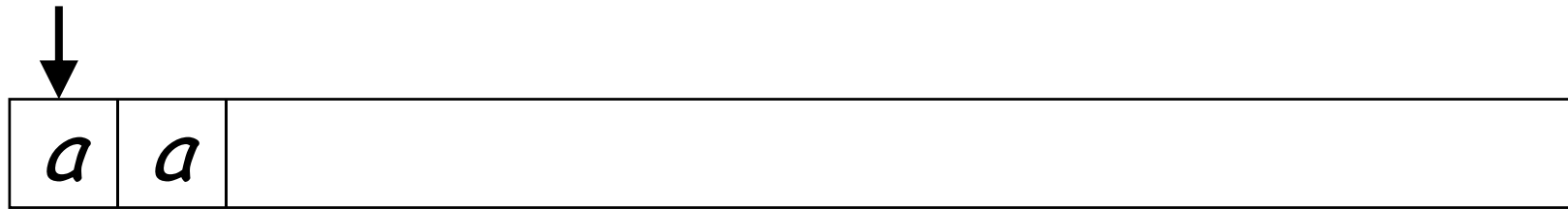
# Example

$aa$  is accepted by the NFA:

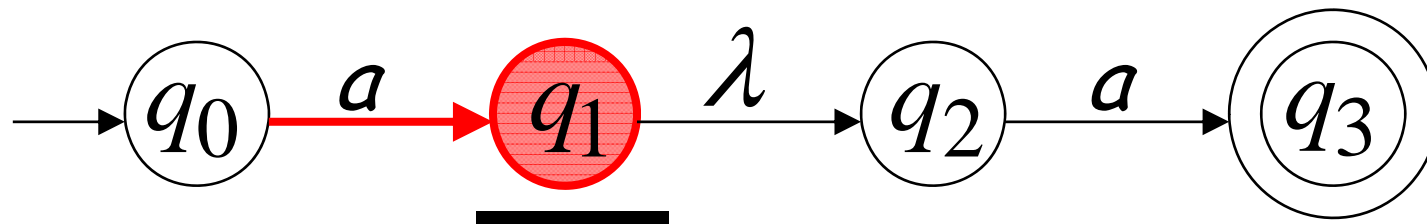
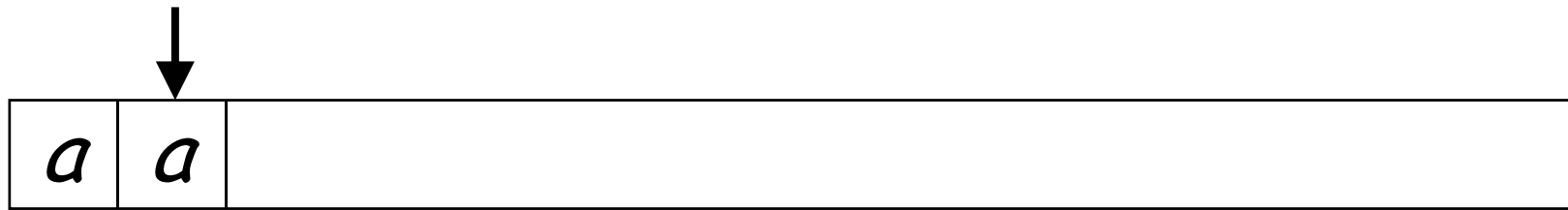


# Lambda Transitions

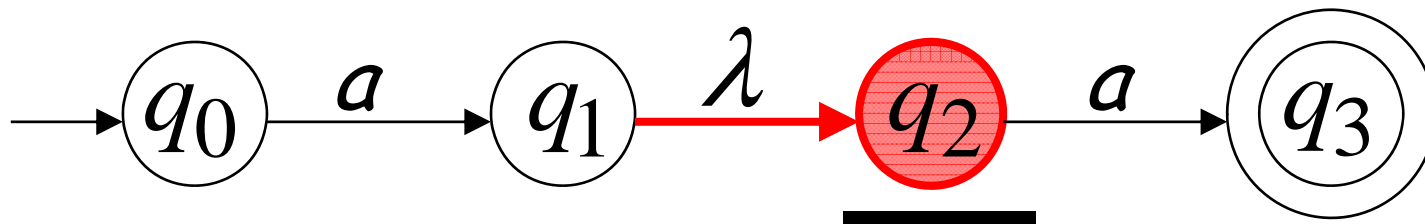
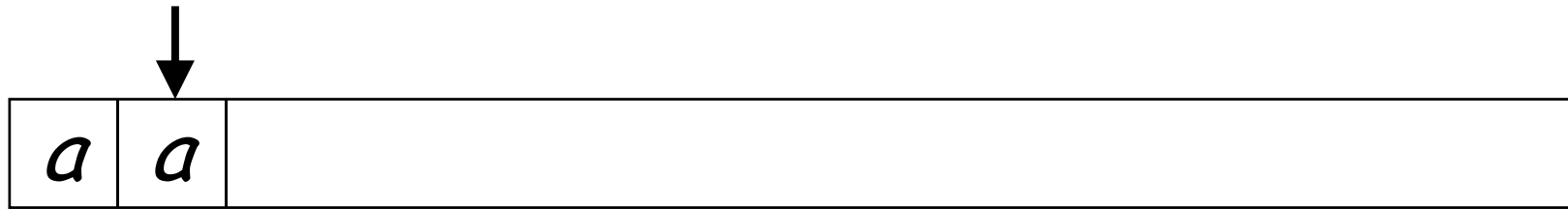


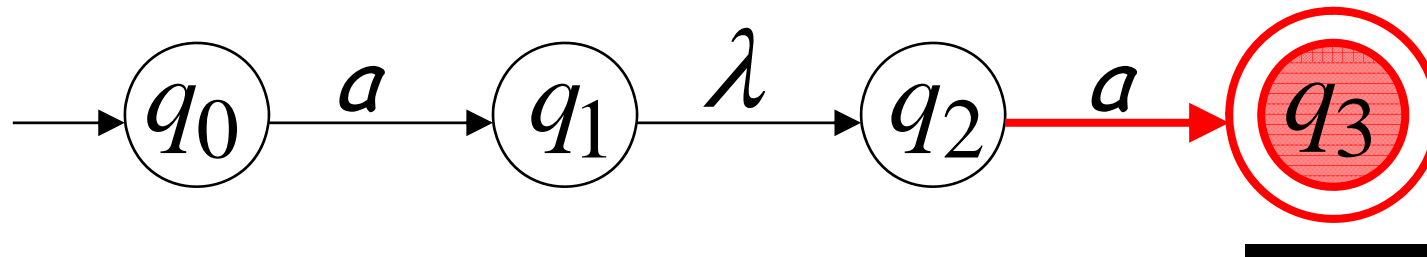
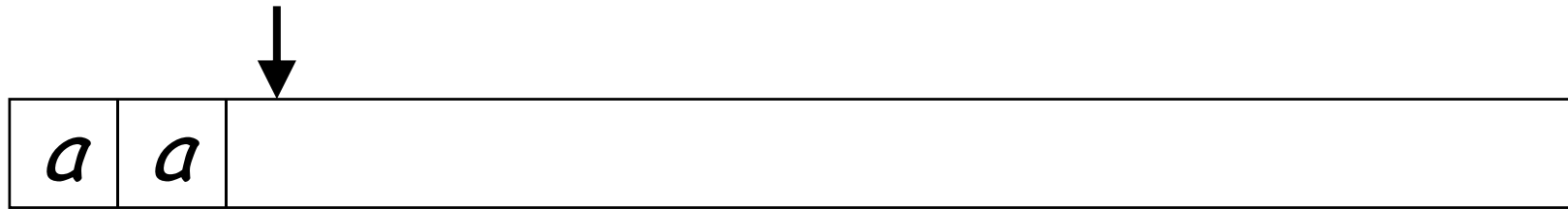


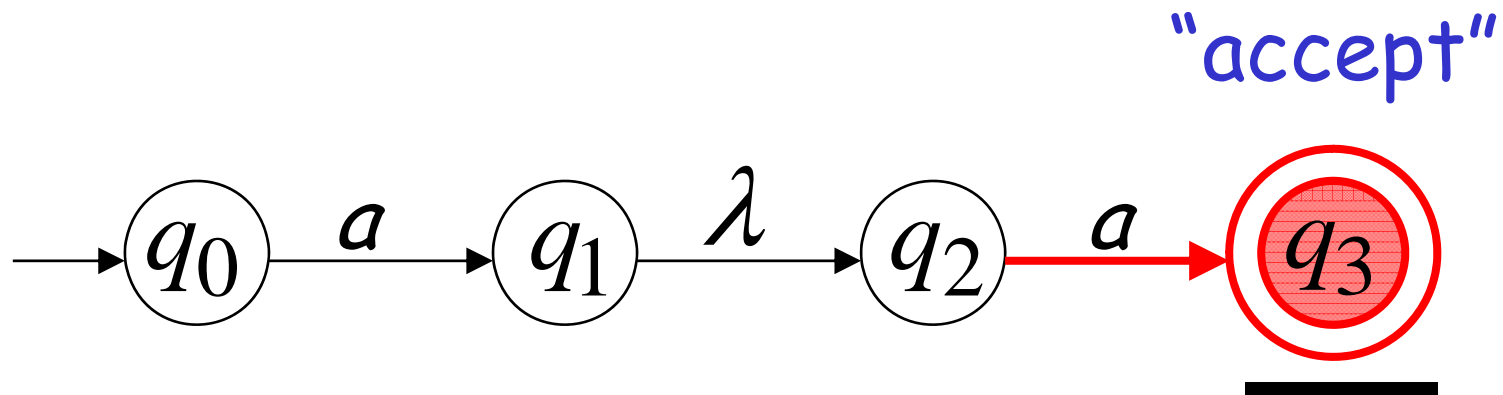
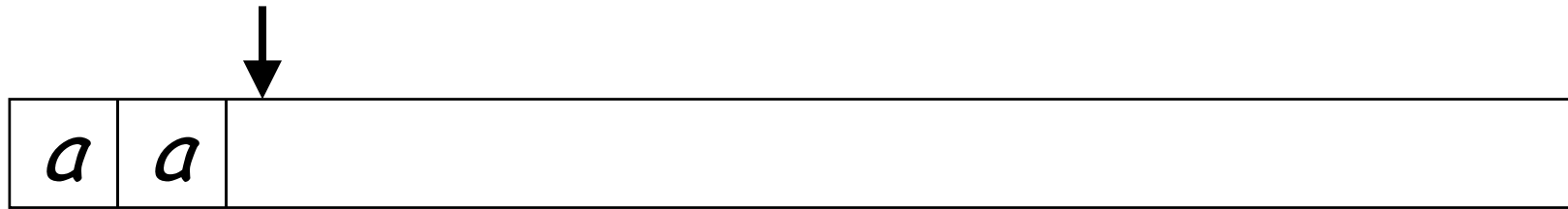




(read head doesn't move)

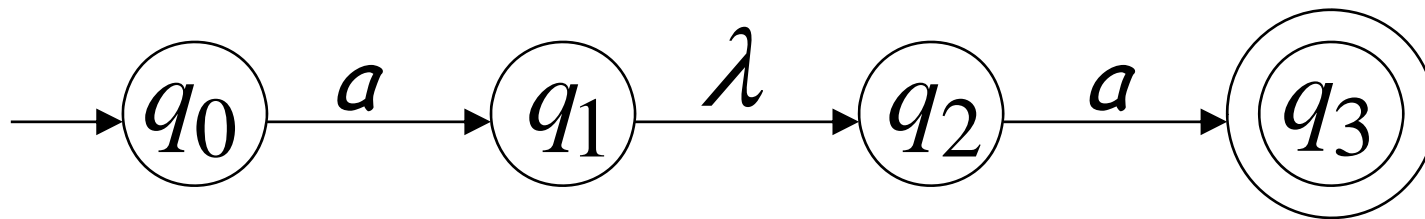




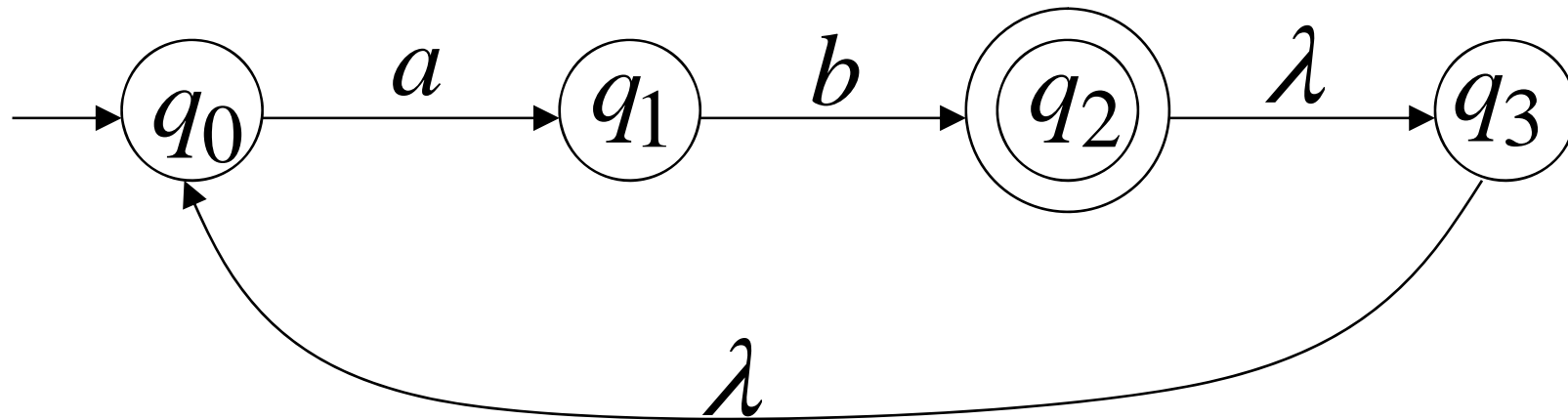


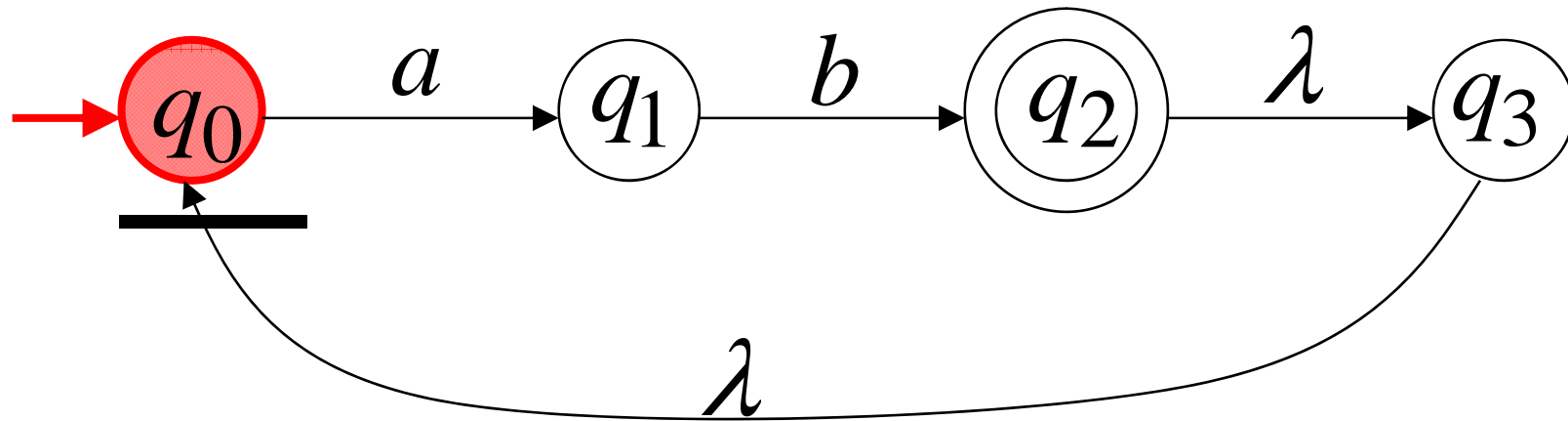
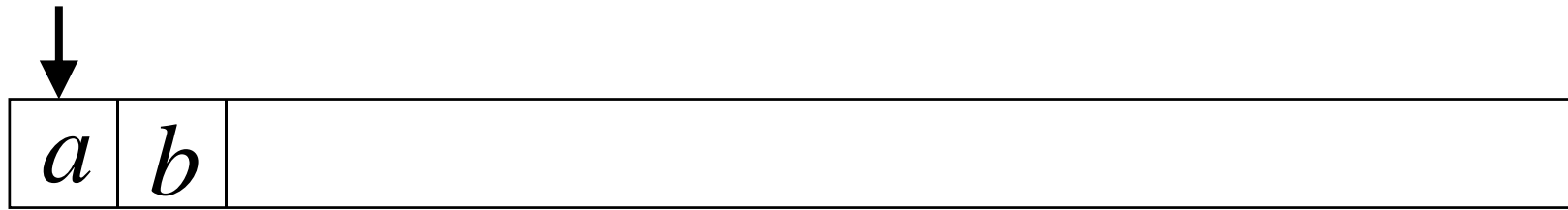
String  $aa$  is accepted

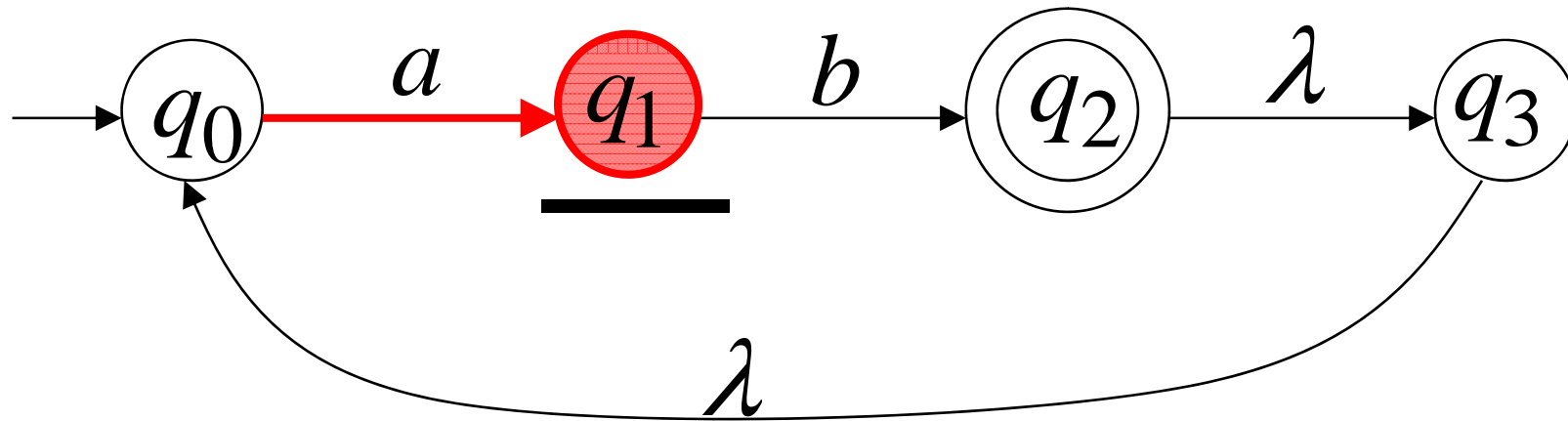
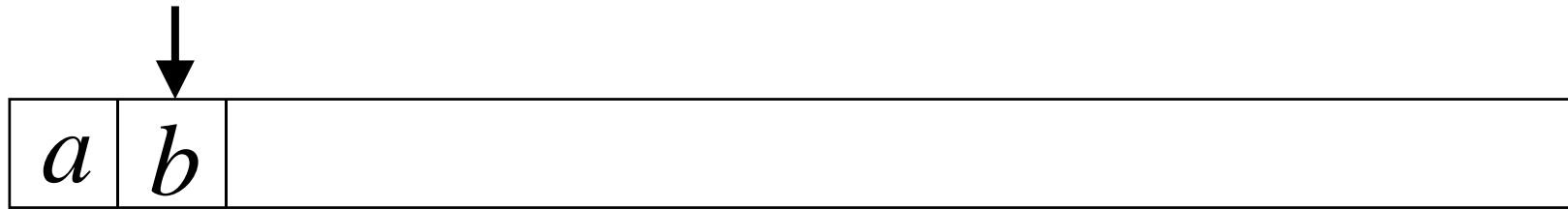
Language accepted:  $L = \{aa\}$



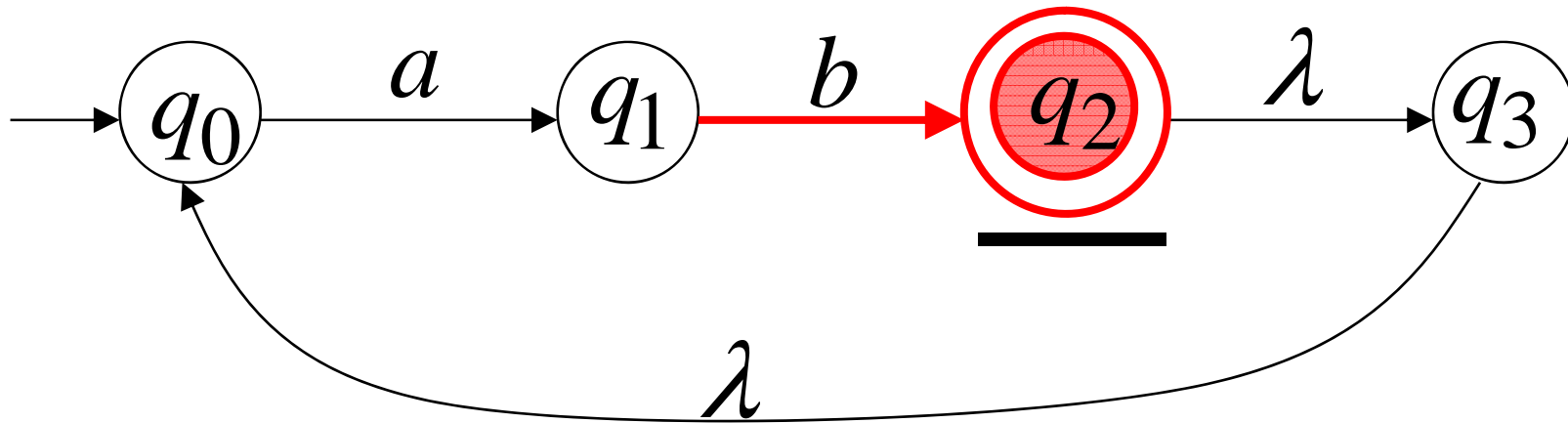
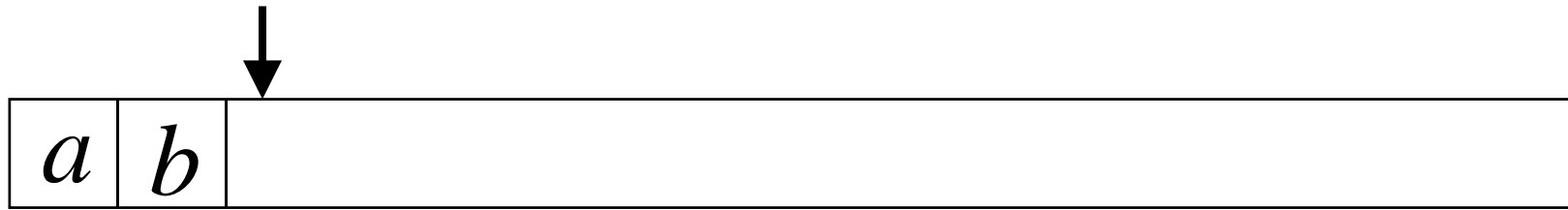
# Another NFA Example

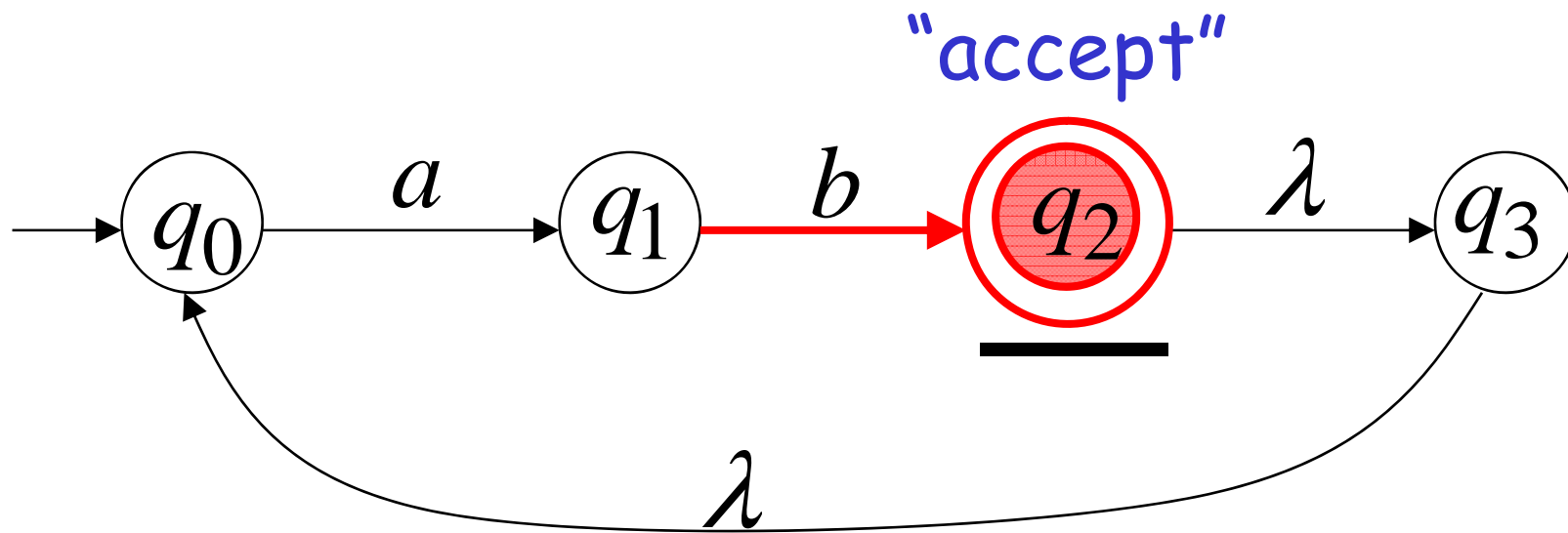
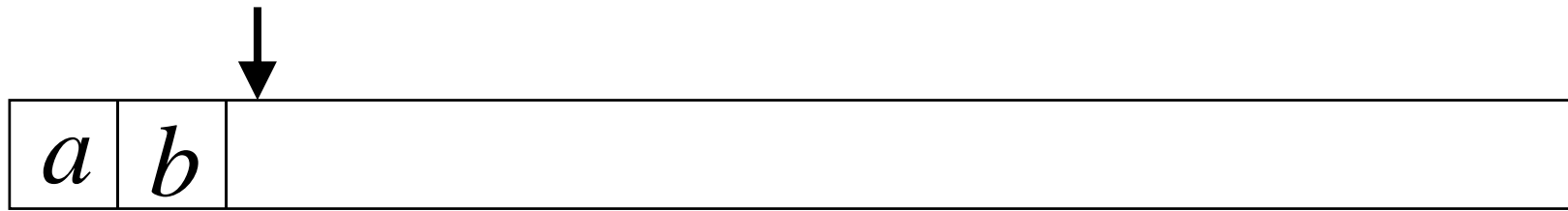




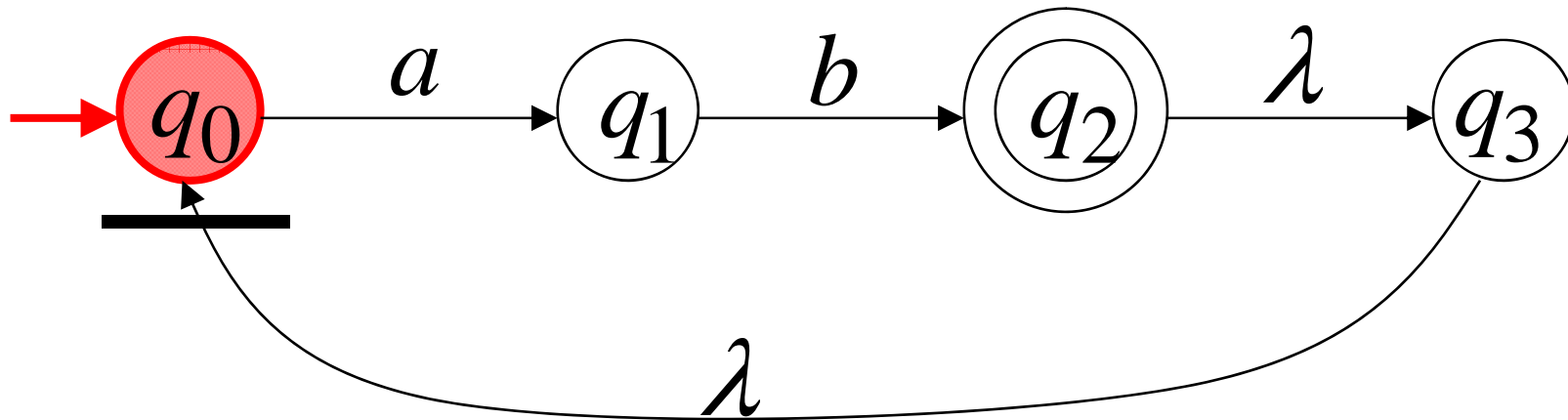
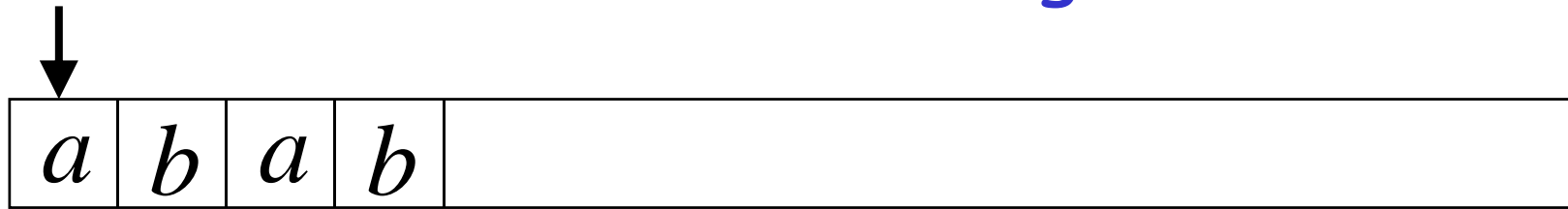


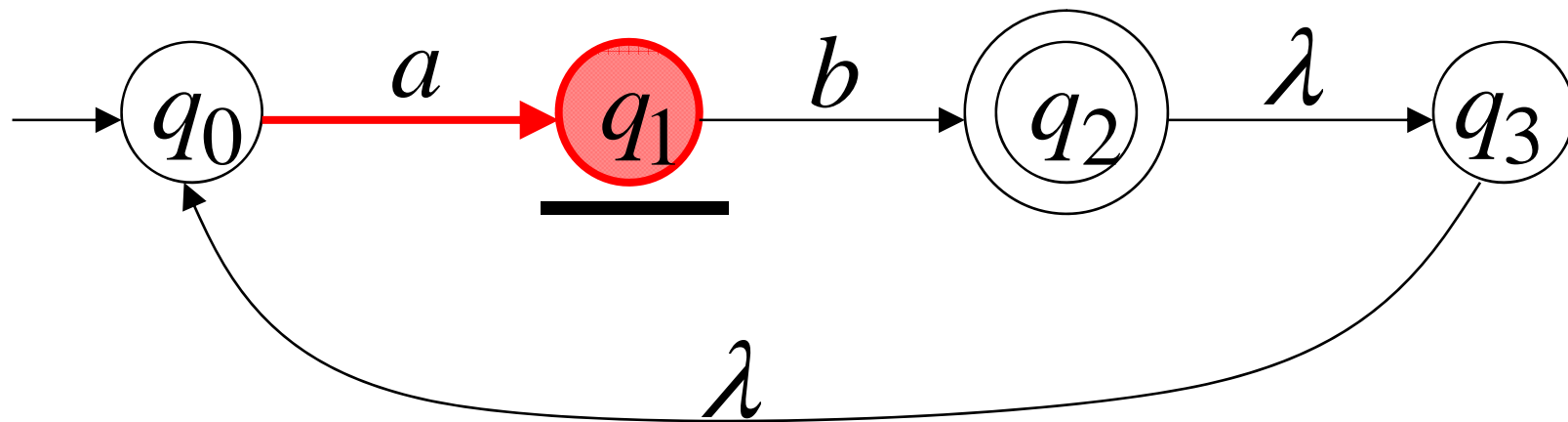
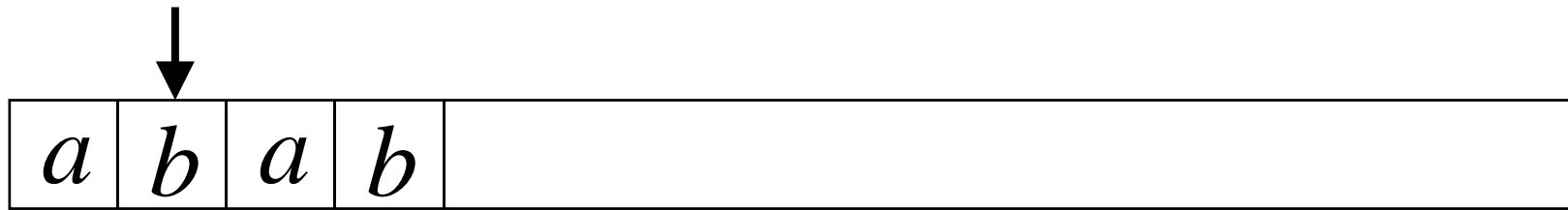


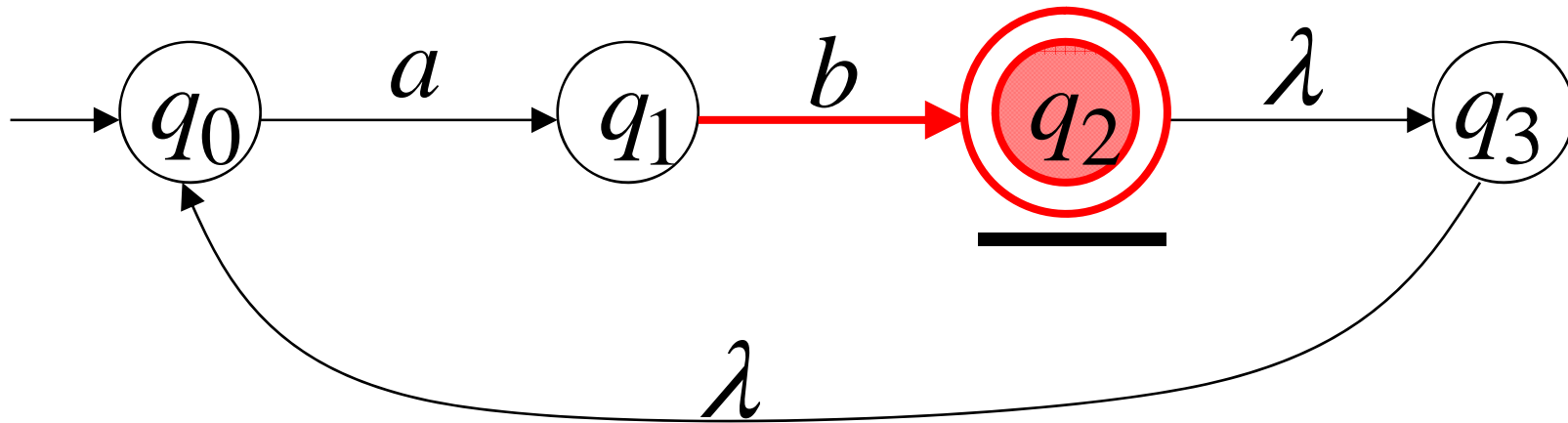
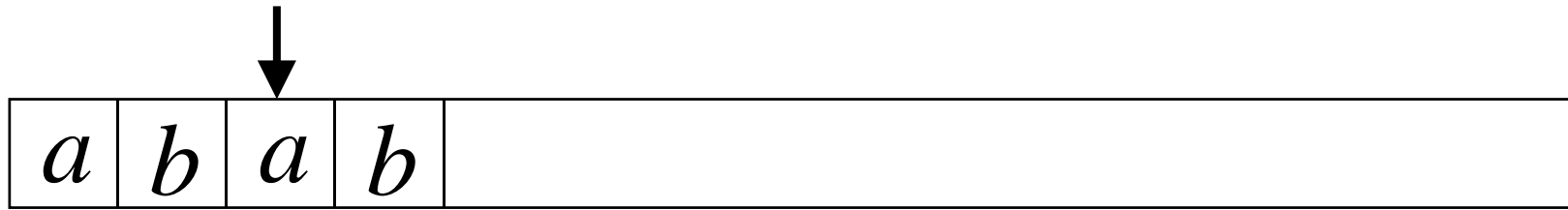


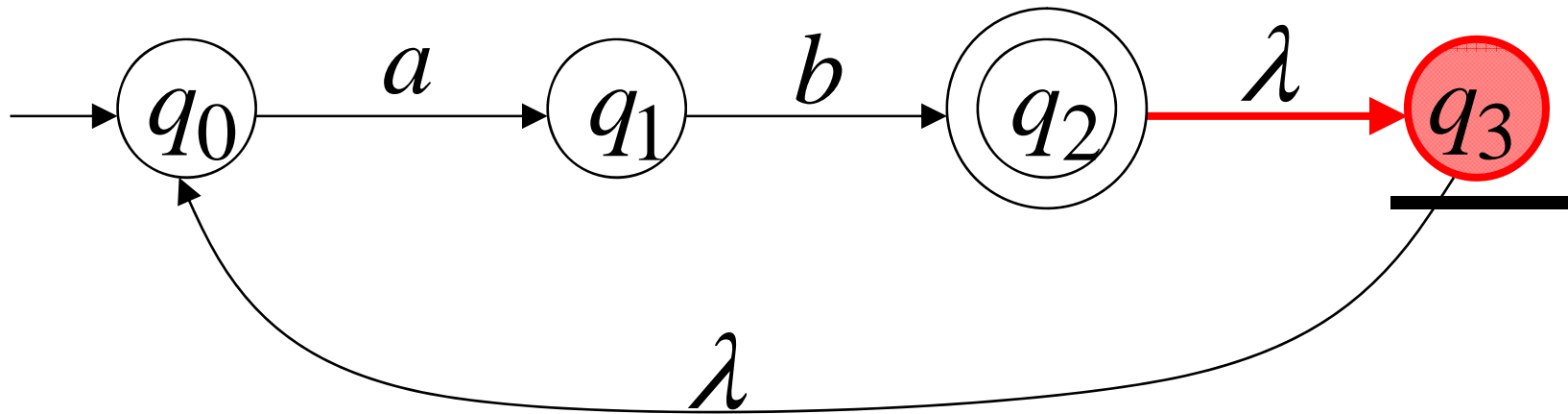
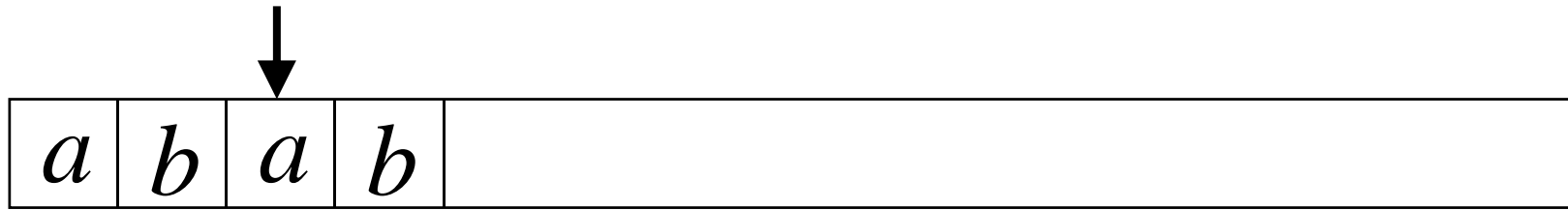


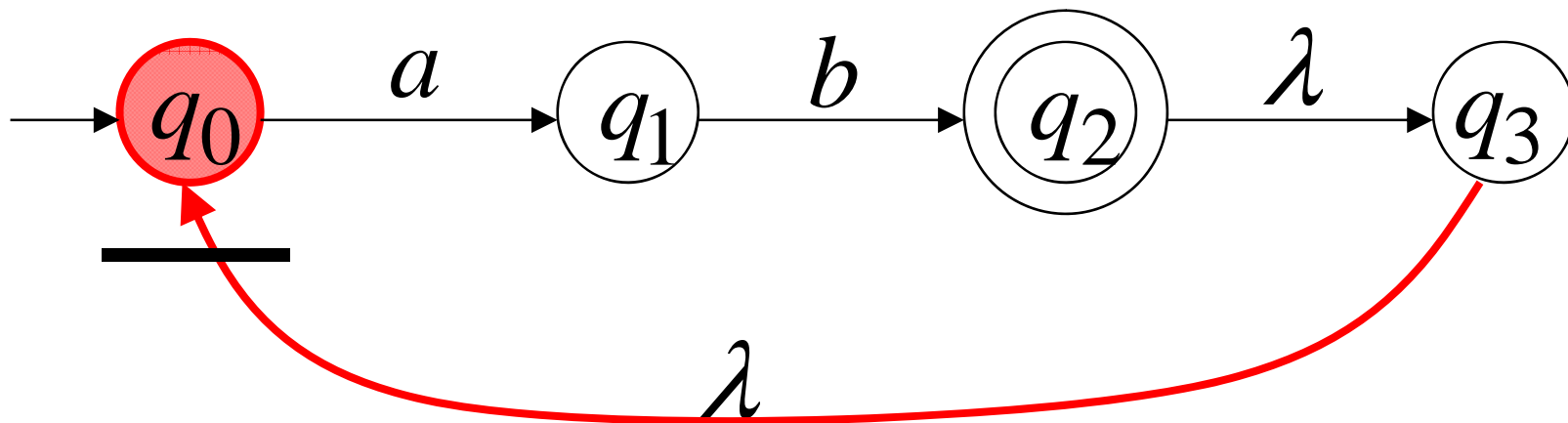
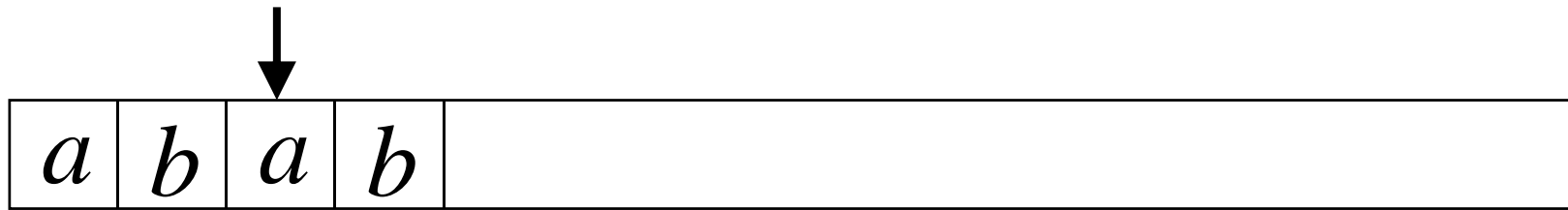
## Another String

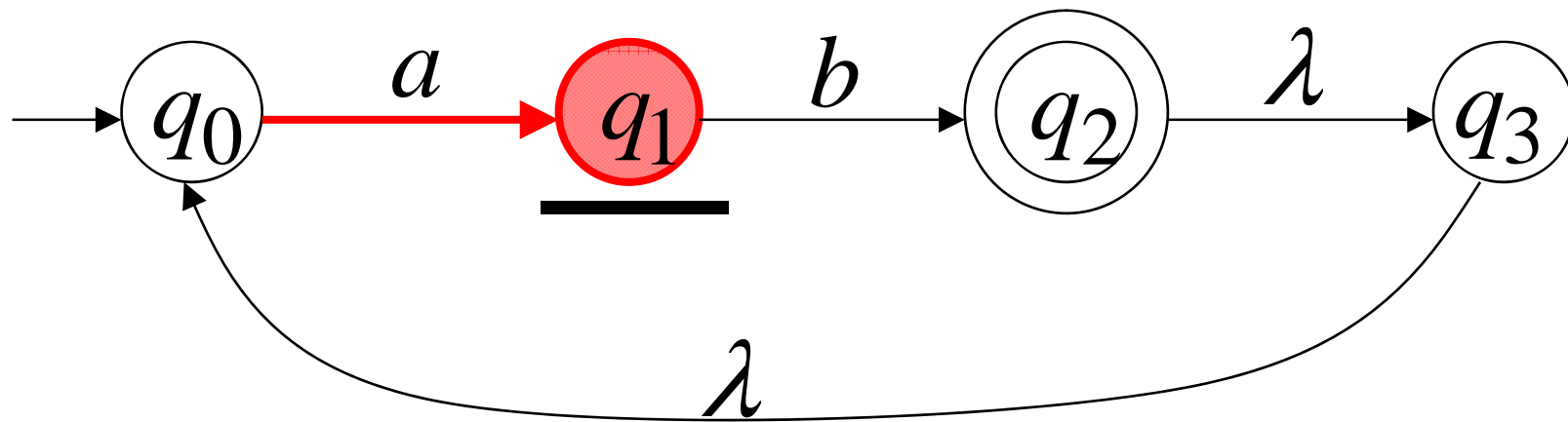
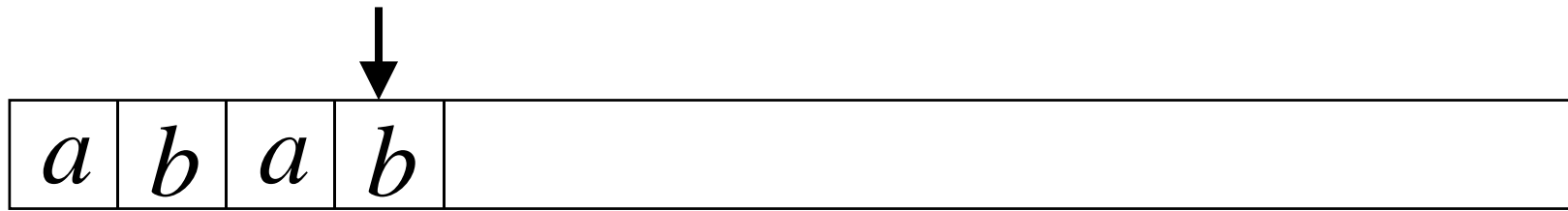




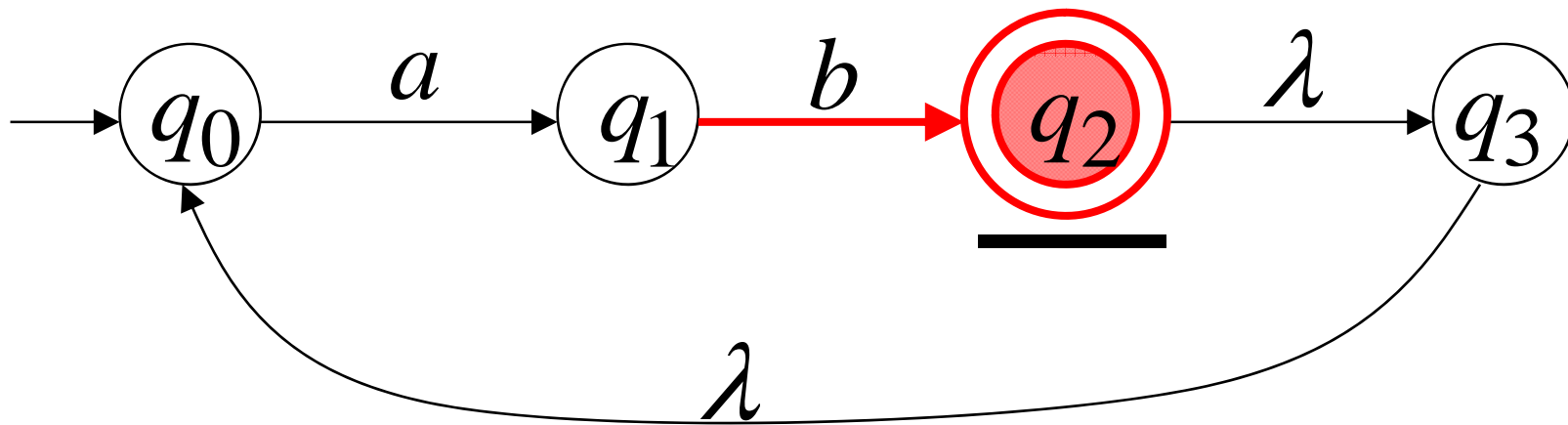
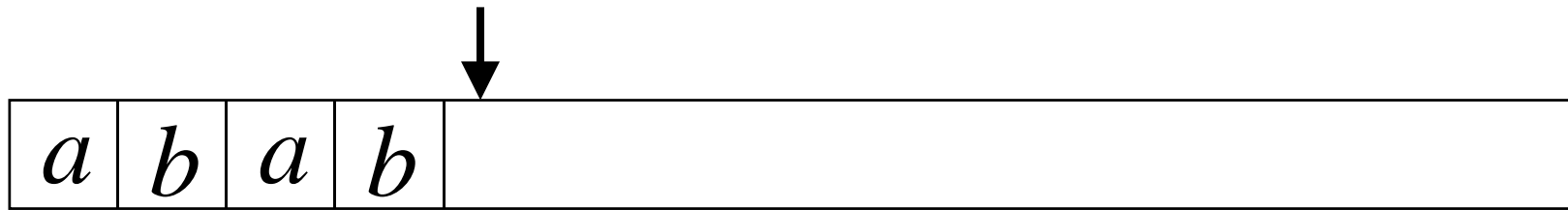


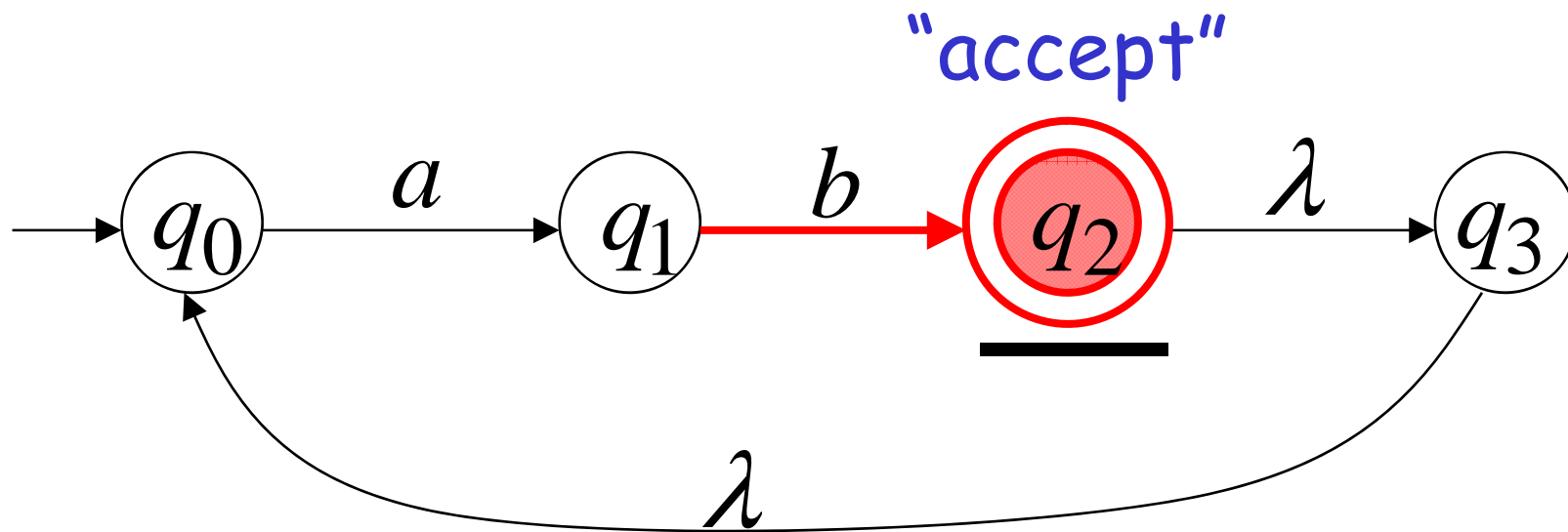
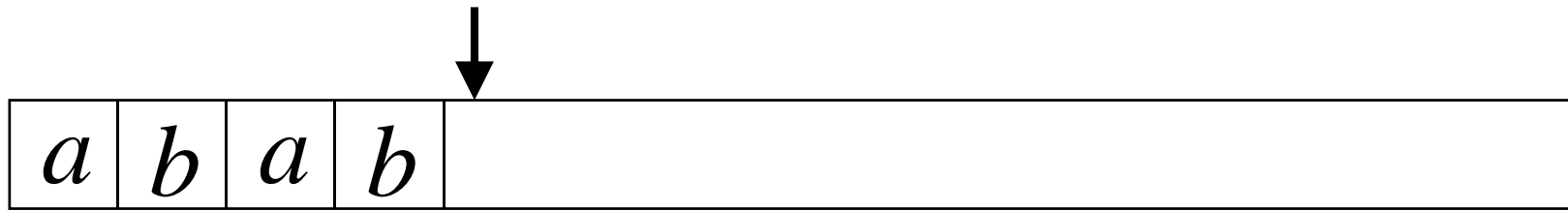






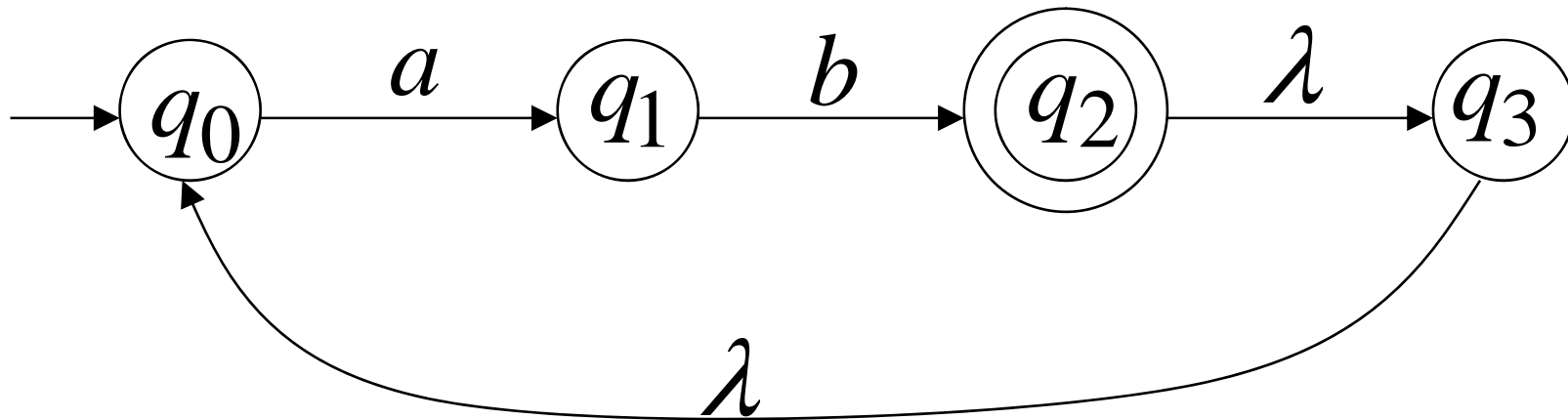




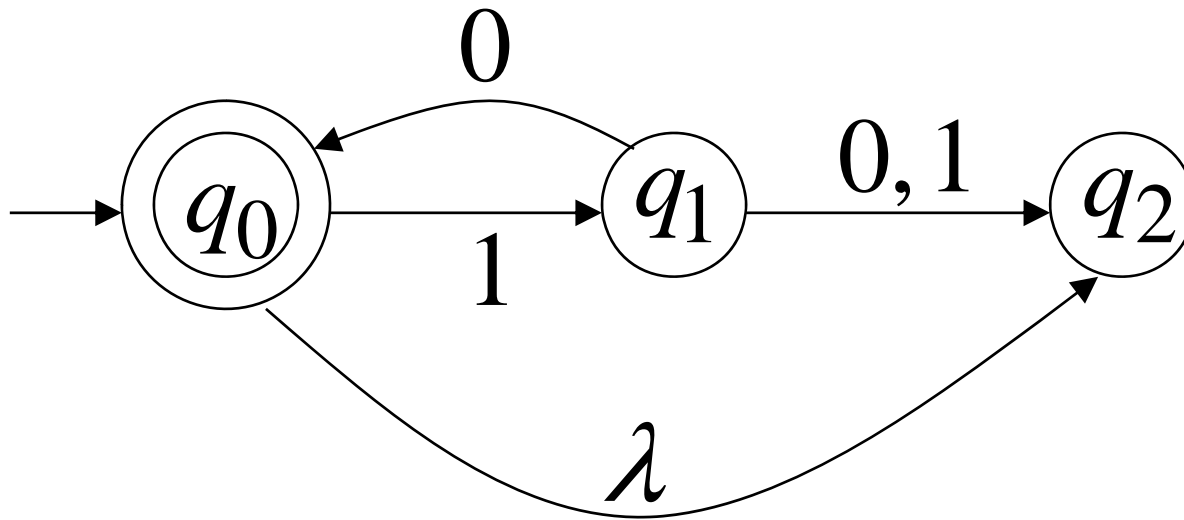


Language accepted

$$L = \{ab, abab, ababab, \dots\}$$
$$= \{ab\}^+$$

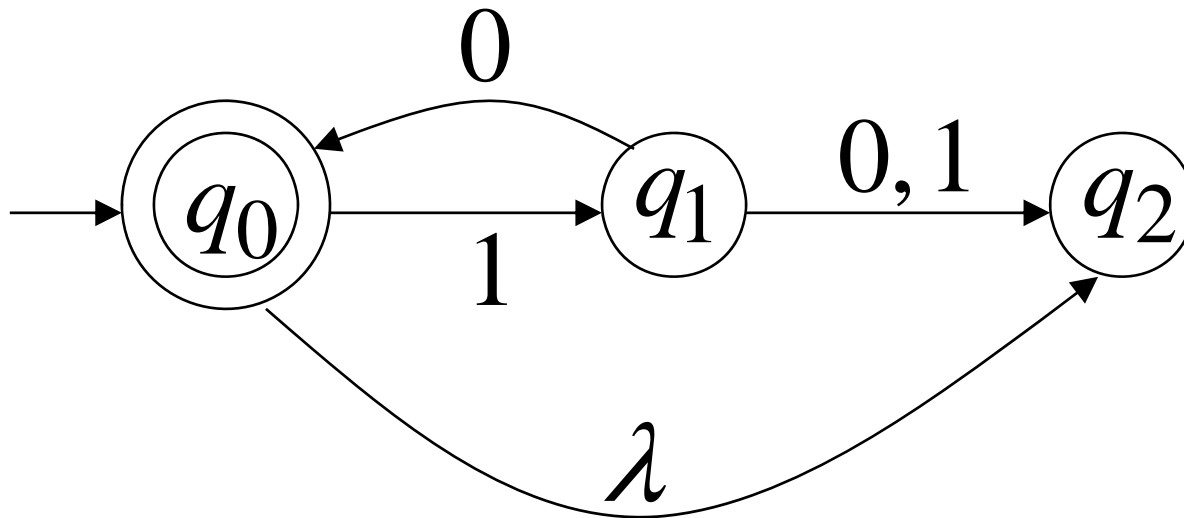


# Another NFA Example



## Language accepted

$$L = \{\lambda, 10, 1010, 101010, \dots\}$$
$$= \{10\}^*$$



# 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

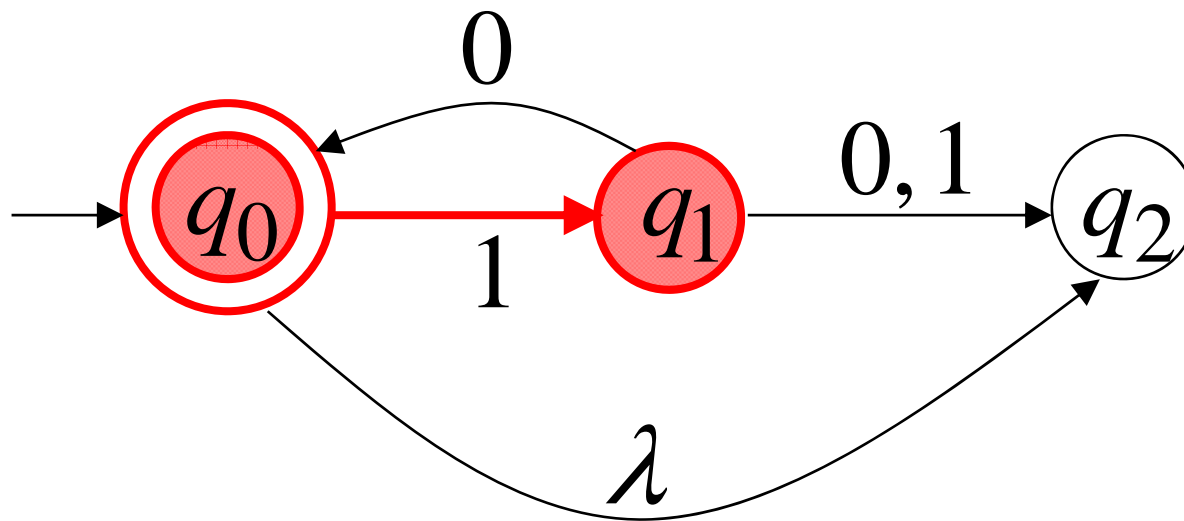
$$\delta : Q \times (\Sigma \cup \{\lambda\}) \rightarrow 2^Q$$

$q_0$ : Initial state

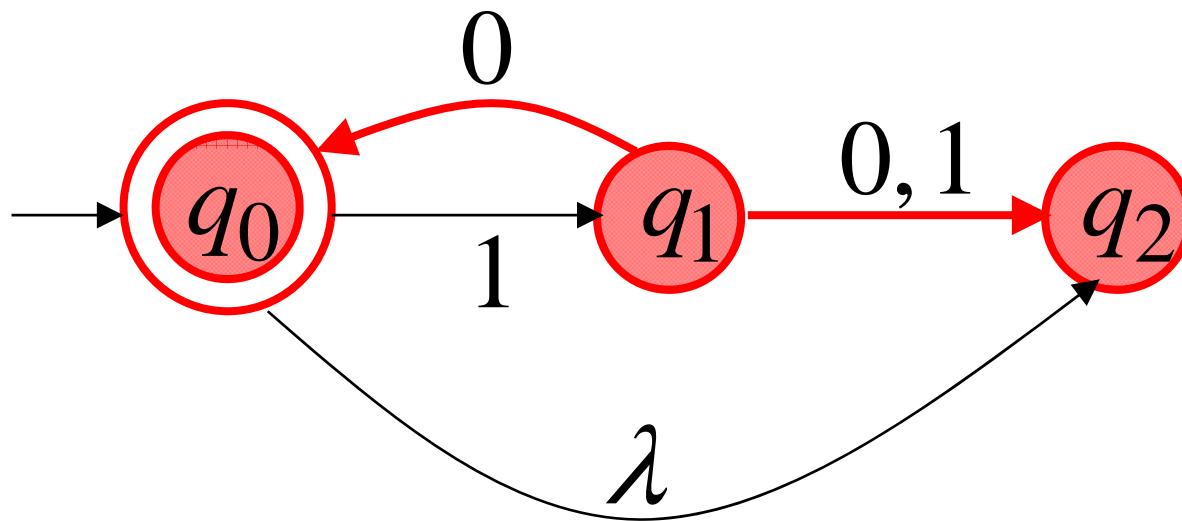
$F$ : Final states

# Transition Function $\delta$

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

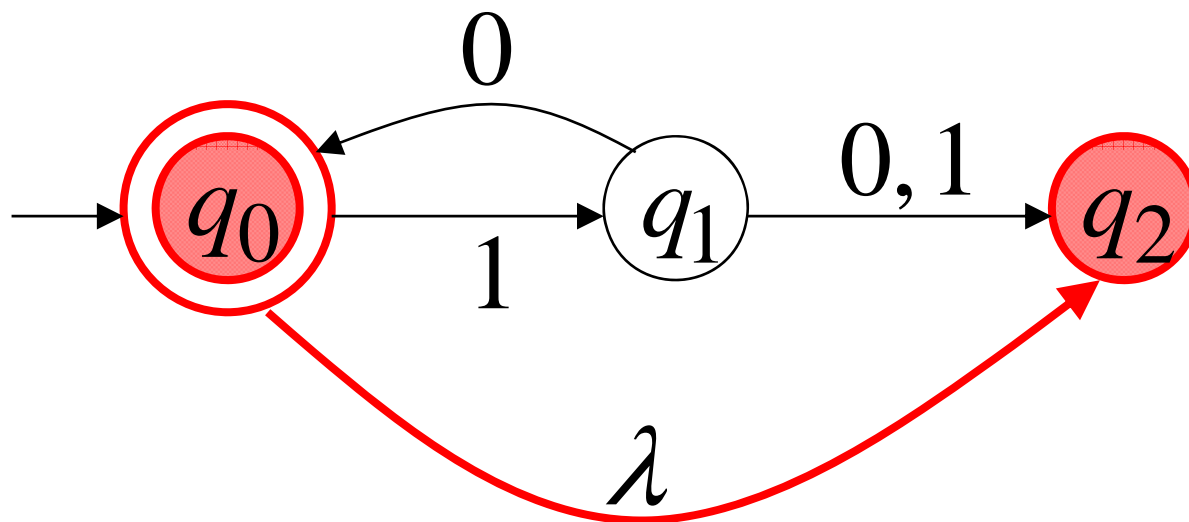


$$\delta(q_1, 0) = \{q_0, q_2\}$$

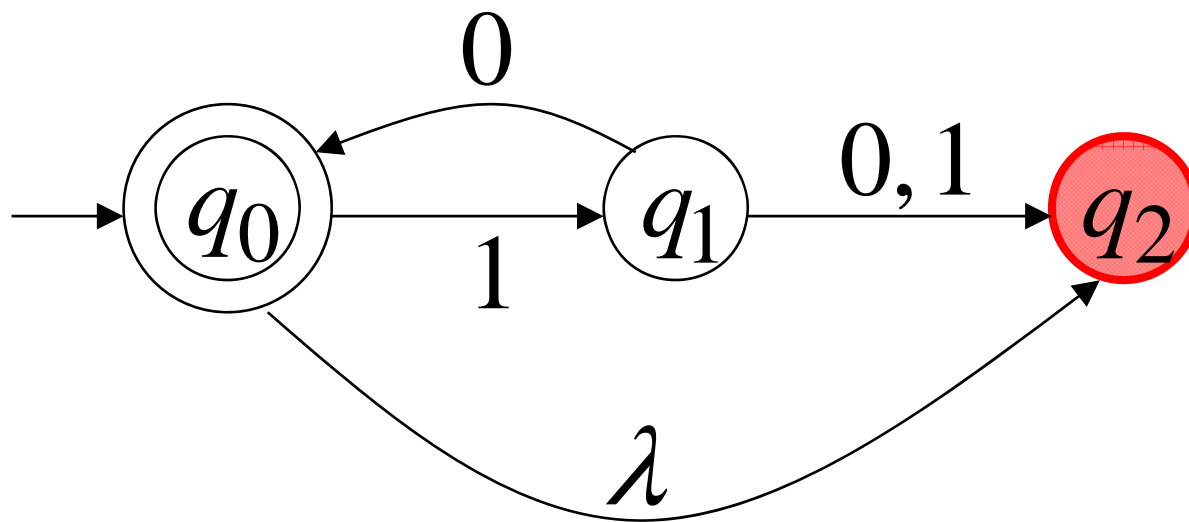




$$\delta(q_0, \lambda) = \{q_0, q_2\}$$

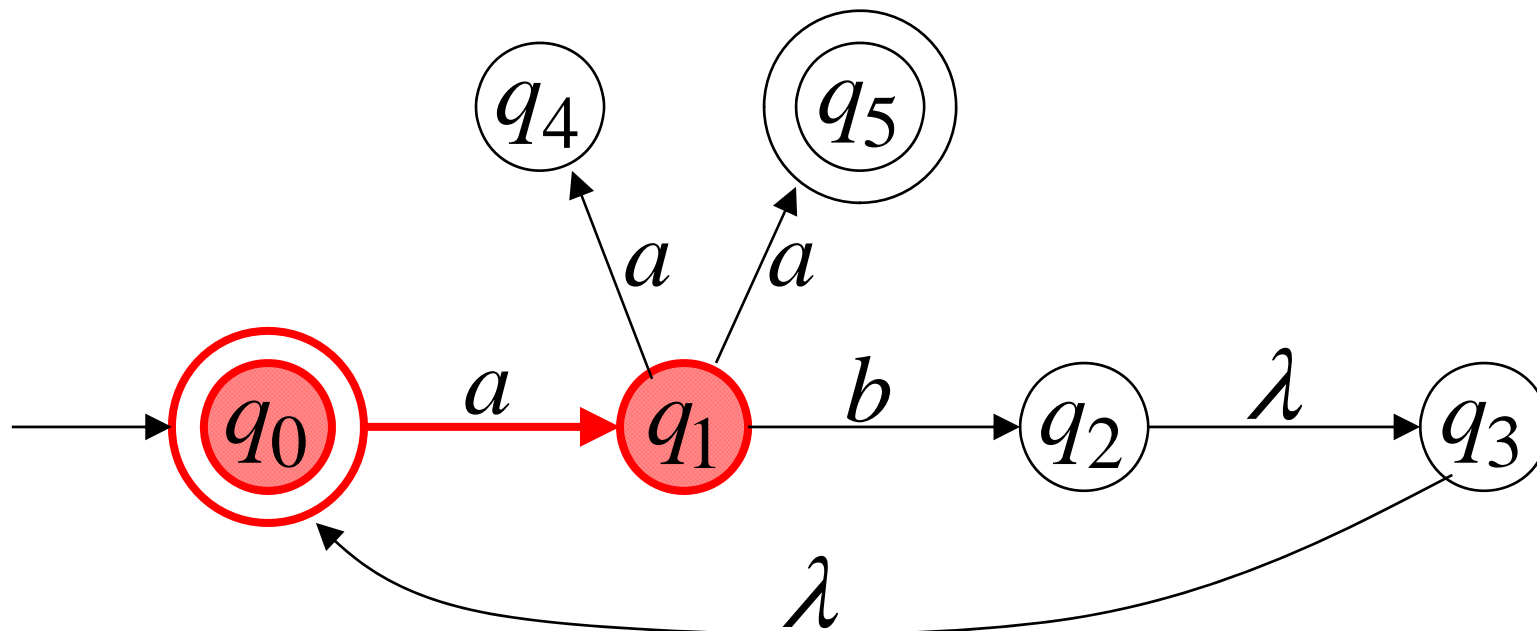


$$\delta(q_2, 1) = \emptyset$$

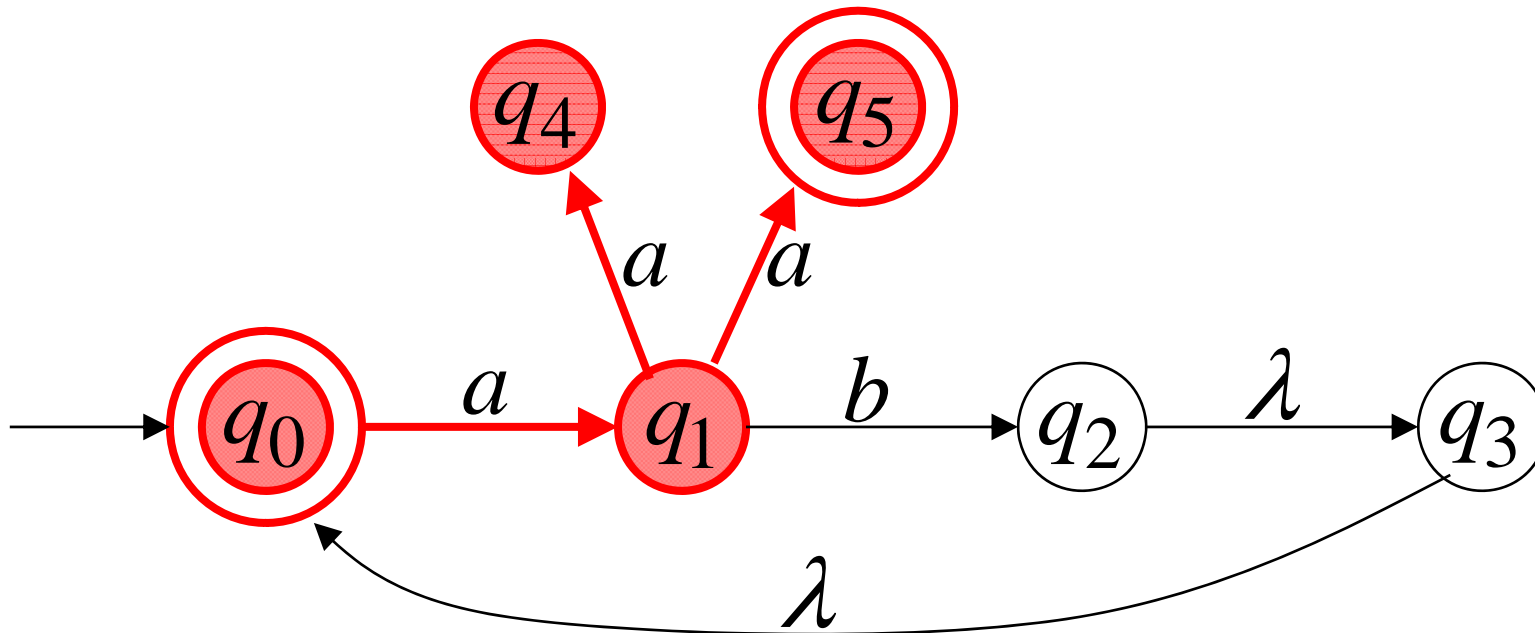


# Extended Transition Function $\delta^*$

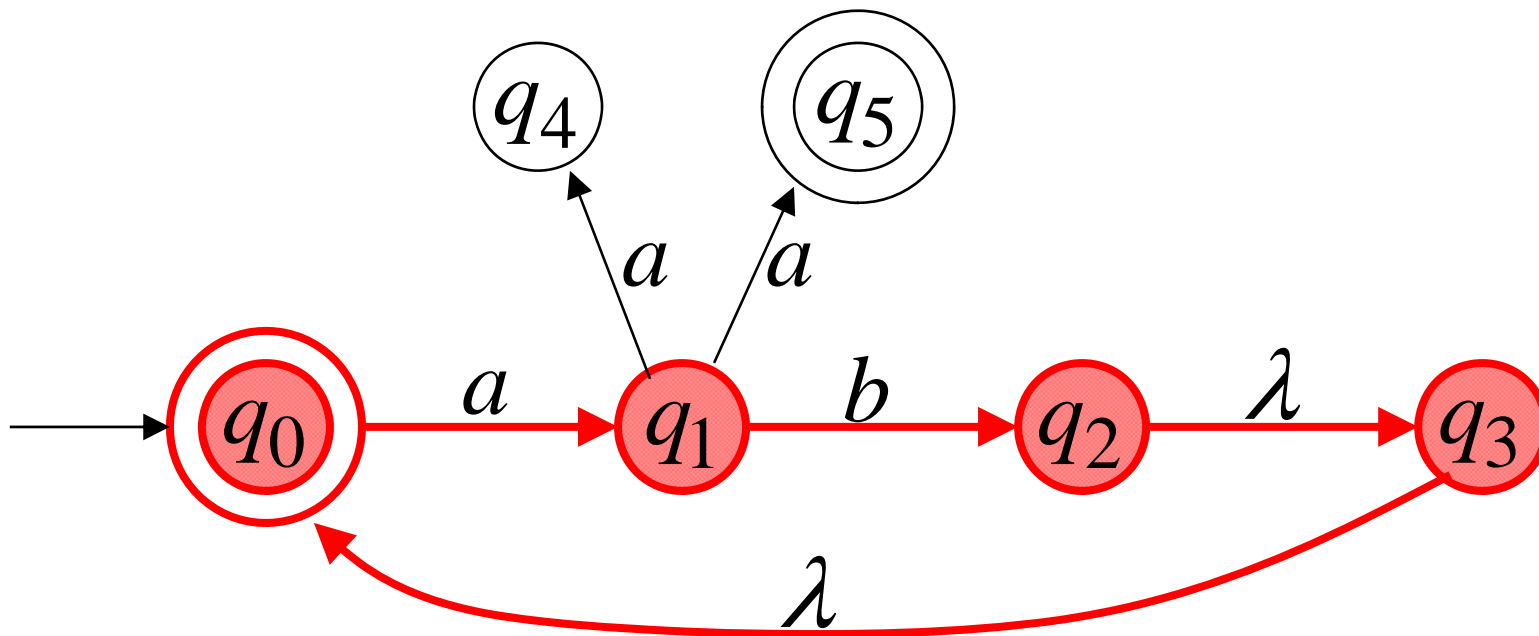
$$\delta^*(q_0, a) = \{q_1\}$$



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



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



Formally

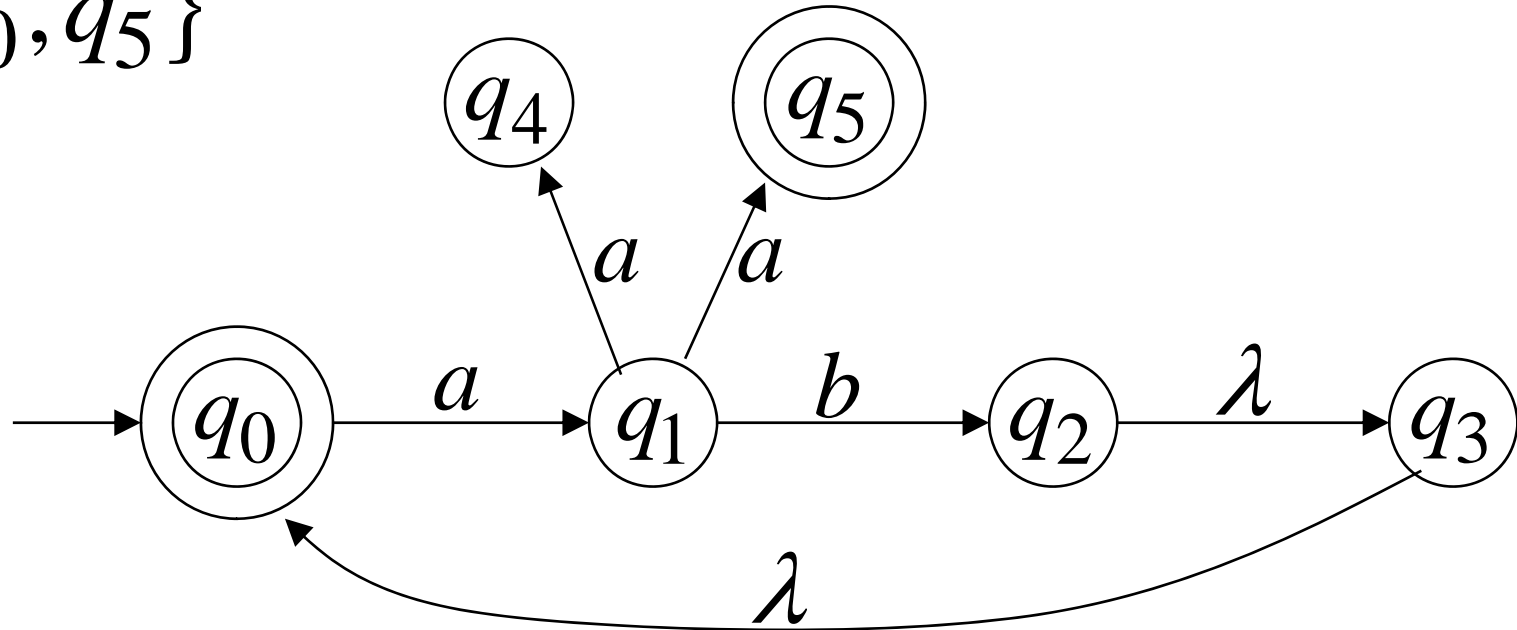
It holds  $q_j \in \delta^*(q_i, w)$

if and only if

there is a walk from  $q_i$  to  $q_j$   
with label  $w$

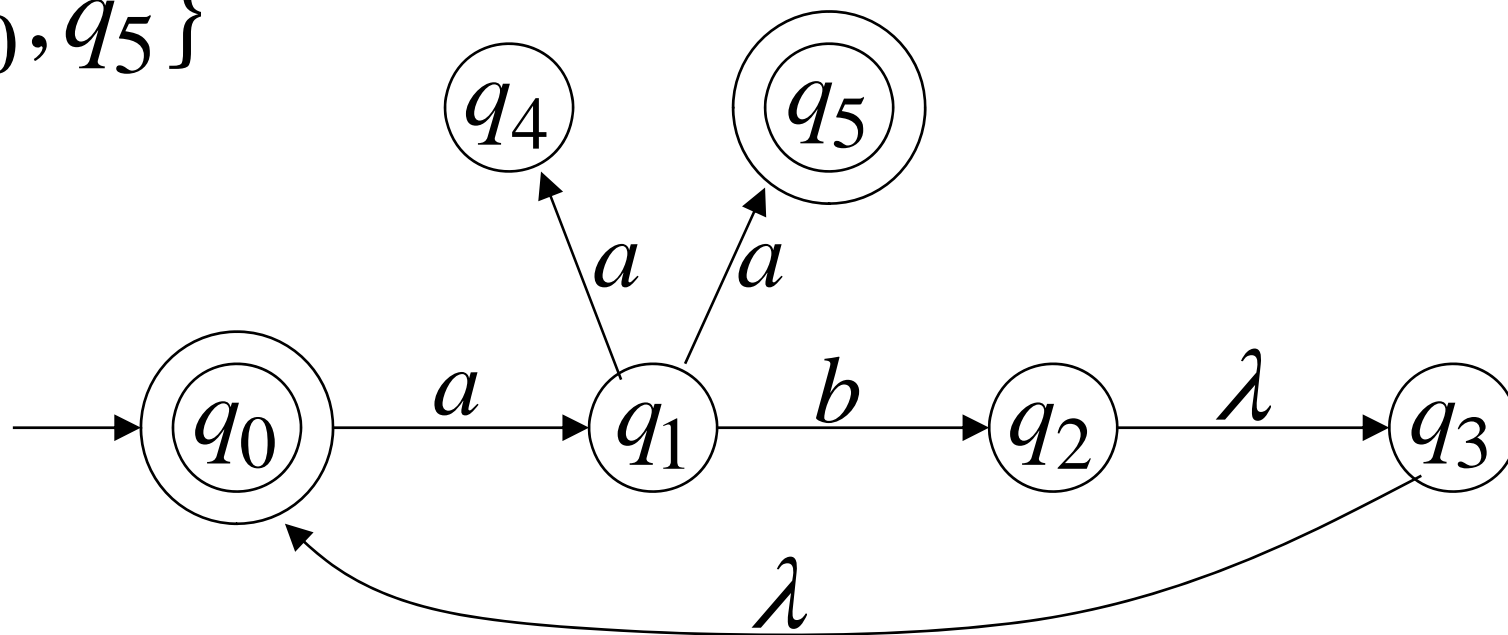
# The Language of an NFA $M$

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



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

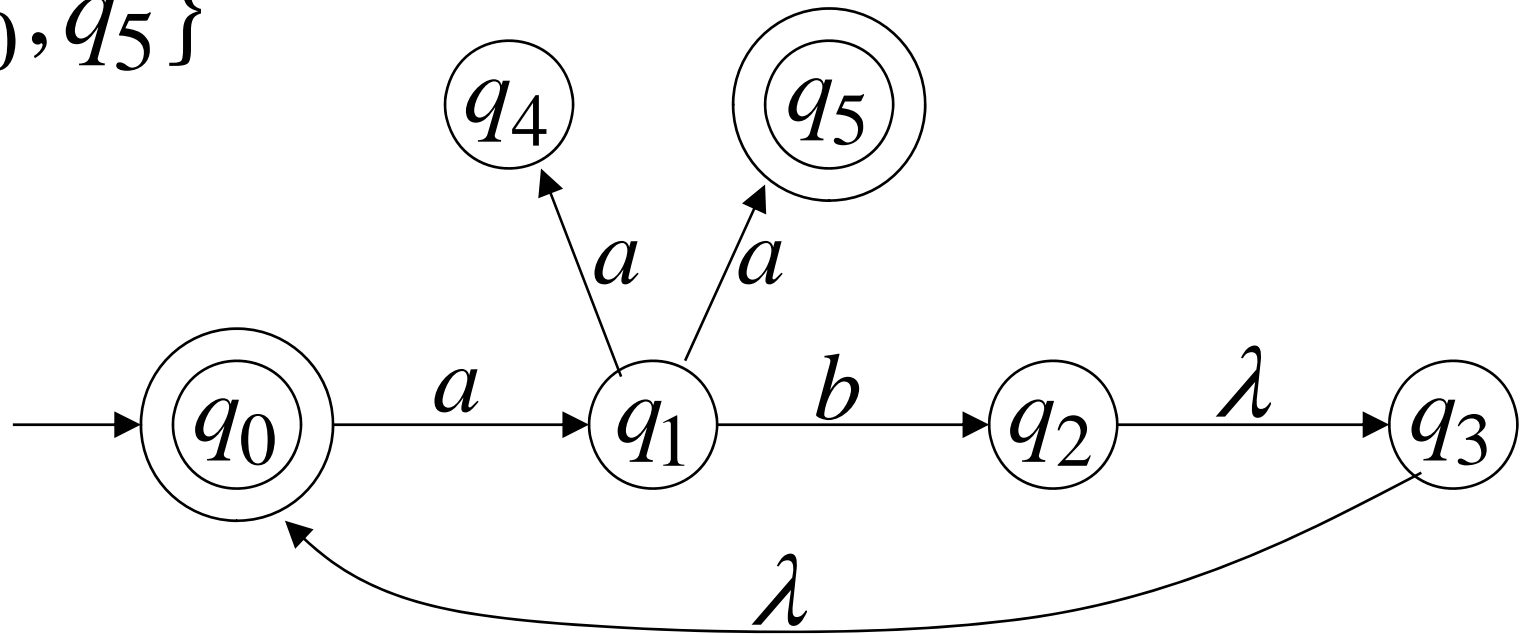
$$F = \{q_0, q_5\}$$



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

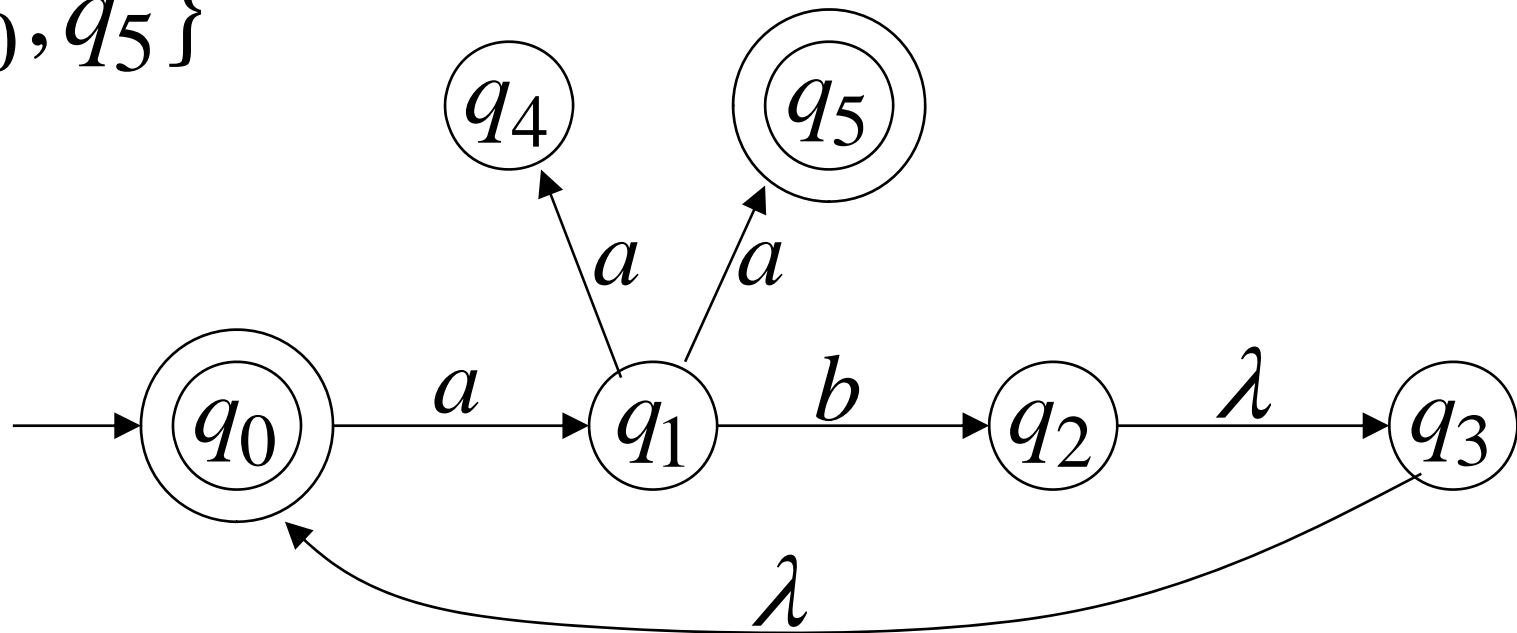


$$F = \{q_0, q_5\}$$



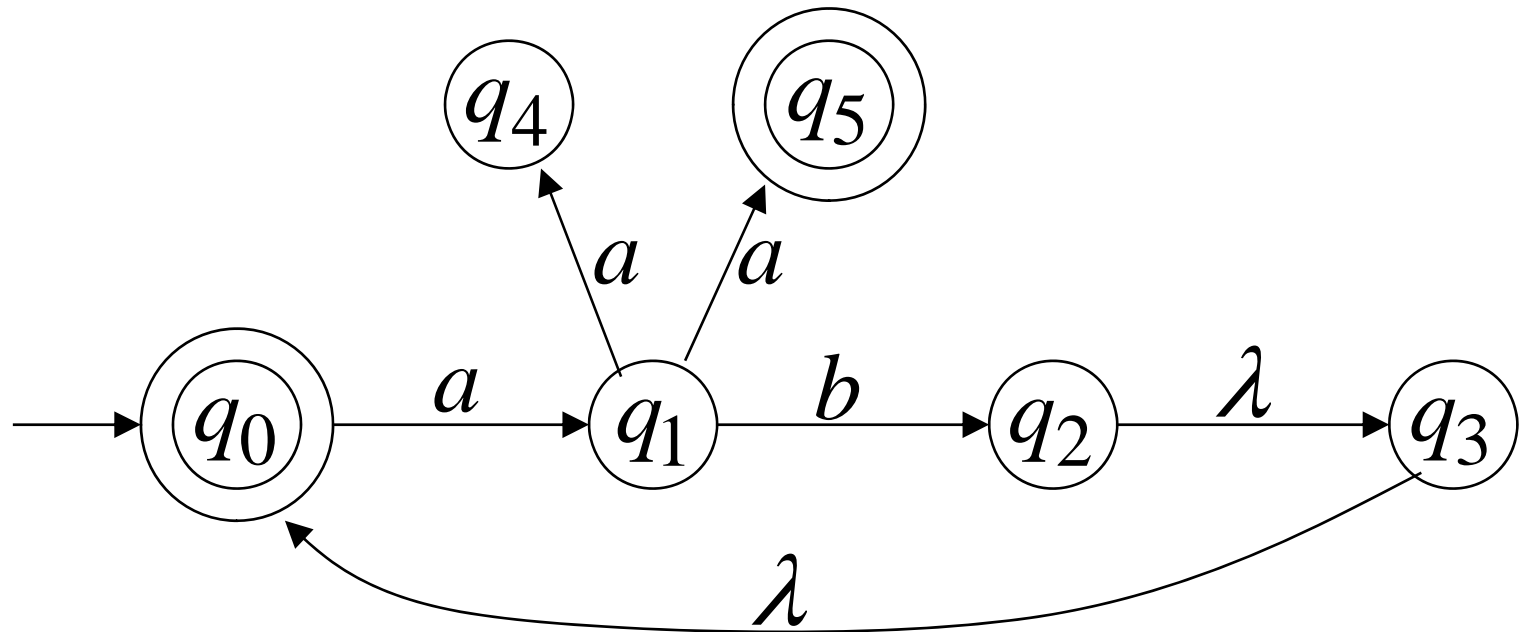
$$\delta^*(q_0, abaa) = \{q_4, \underline{q_5}\} \quad aaba \in L(M)$$

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



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

$$aba \notin L(M)$$



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

## Formally

The language accepted by NFA  $M$  is:

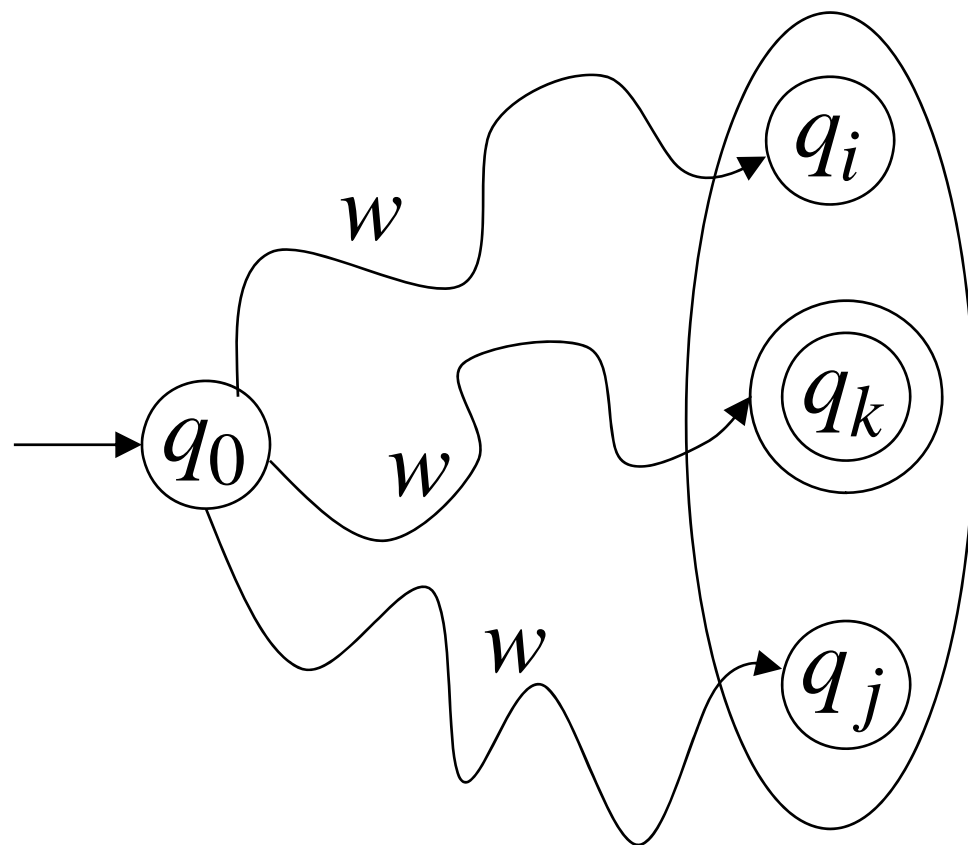
$$L(M) = \{w_1, w_2, w_3, \dots\}$$

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

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

$$w \in L(M)$$

$$\delta^*(q_0, w)$$



$$q_k \in F$$

# Equivalence of NFAs and DFAs

# Equivalence of Machines

For DFAs or NFAs:

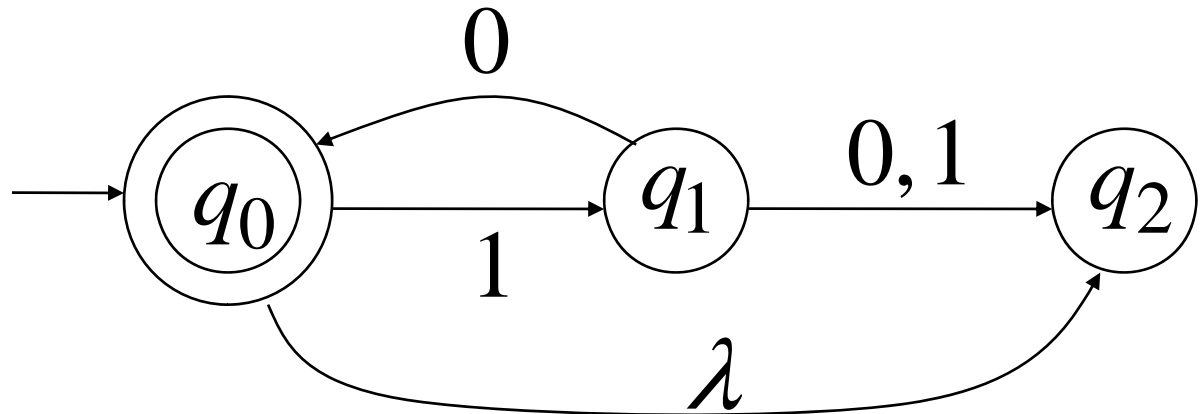
Machine  $M_1$  is equivalent to machine  $M_2$

if  $L(M_1) = L(M_2)$

# Example

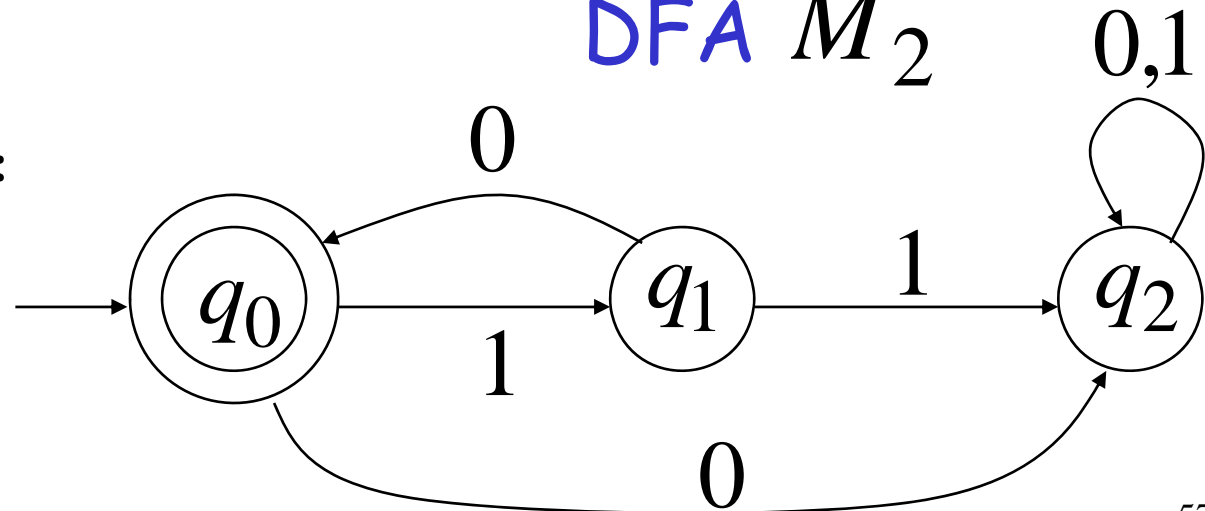
NFA  $M_1$

$$L(M_1) = \{10\}^*$$



DFA  $M_2$

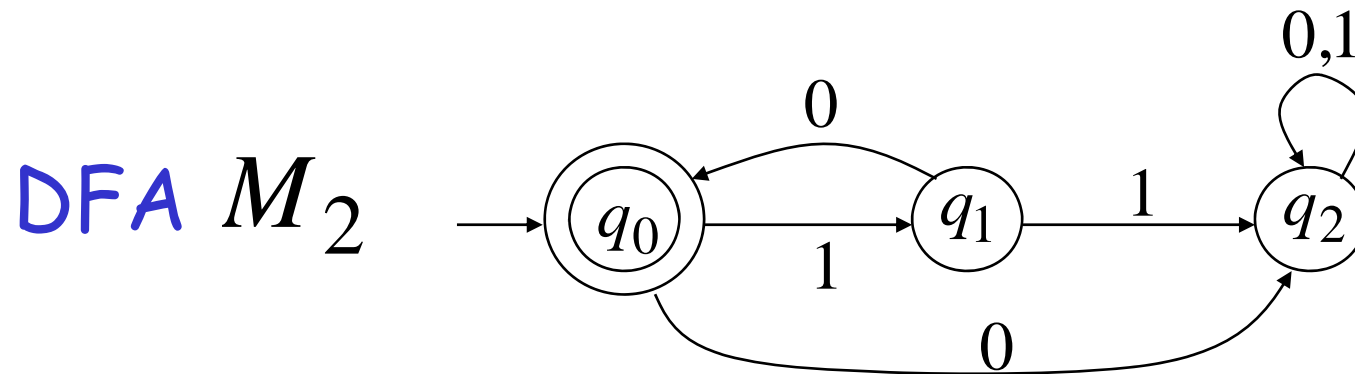
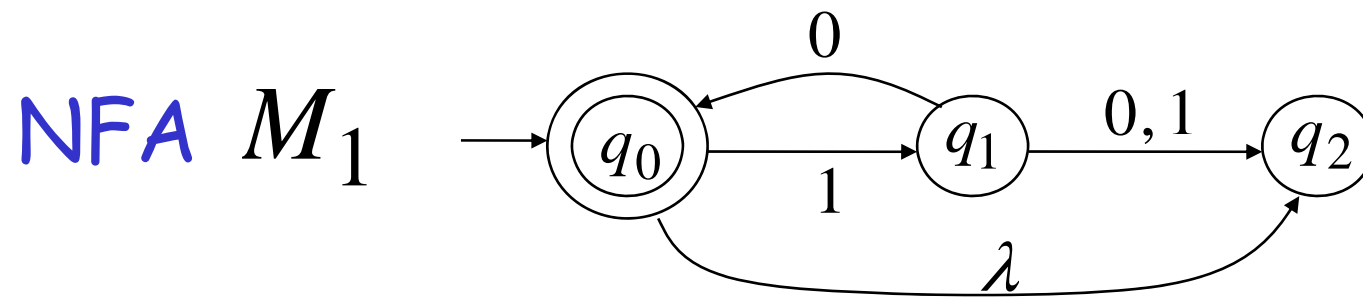
$$L(M_2) = \{10\}^*$$





Since  $L(M_1) = L(M_2) = \{10\}^*$

machines  $M_1$  and  $M_2$  are equivalent



# Equivalence of NFAs and DFAs

Question: NFAs = DFAs ?



Same power?

Accept the same languages?

# Equivalence of NFAs and DFAs

Question: NFAs = DFAs ? YES!

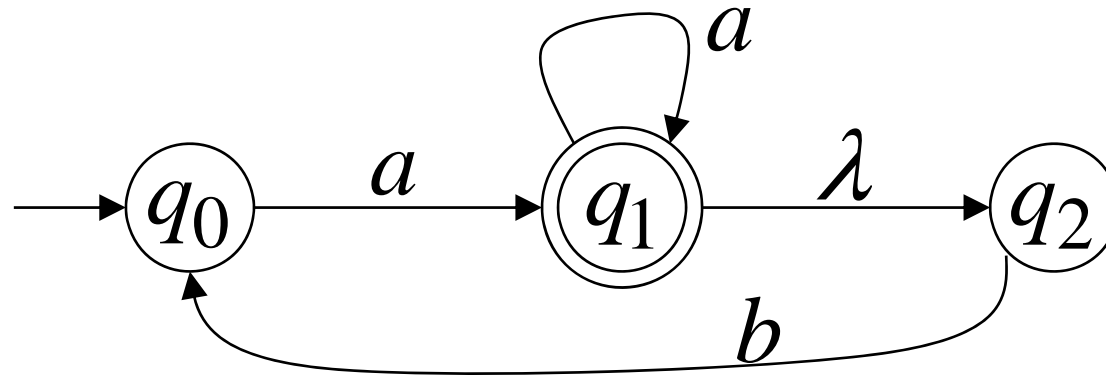


Same power?

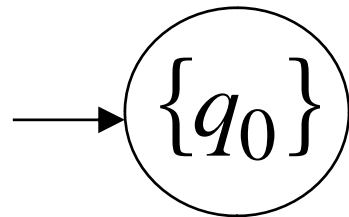
Accept the same languages?

# NFA to DFA

NFA

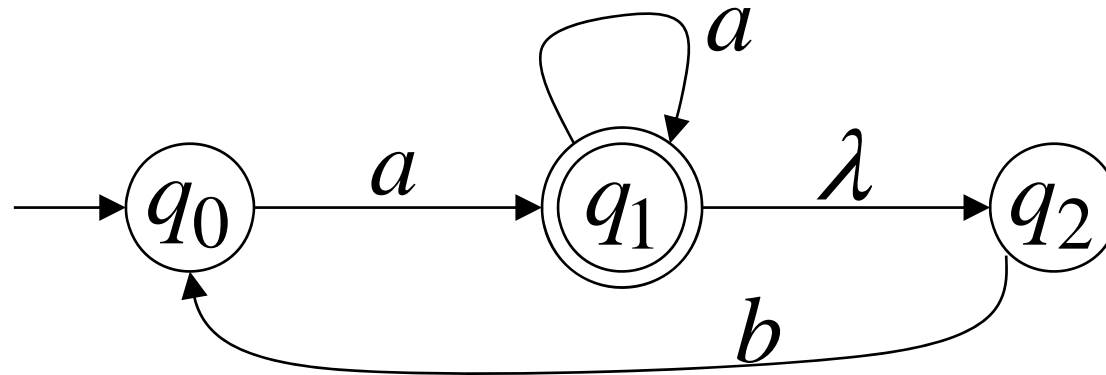


DFA

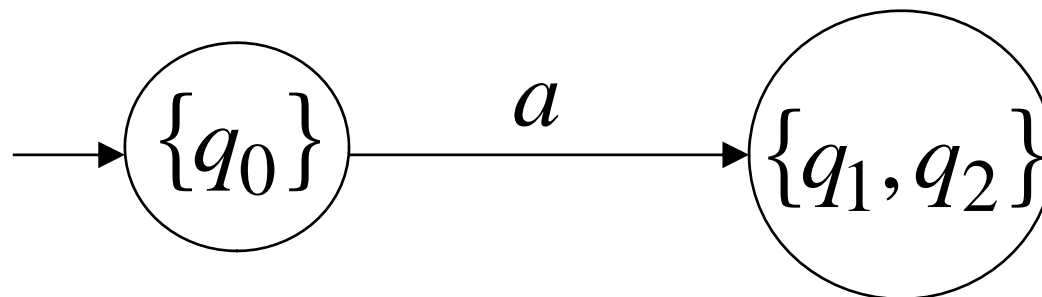


# NFA to DFA

NFA

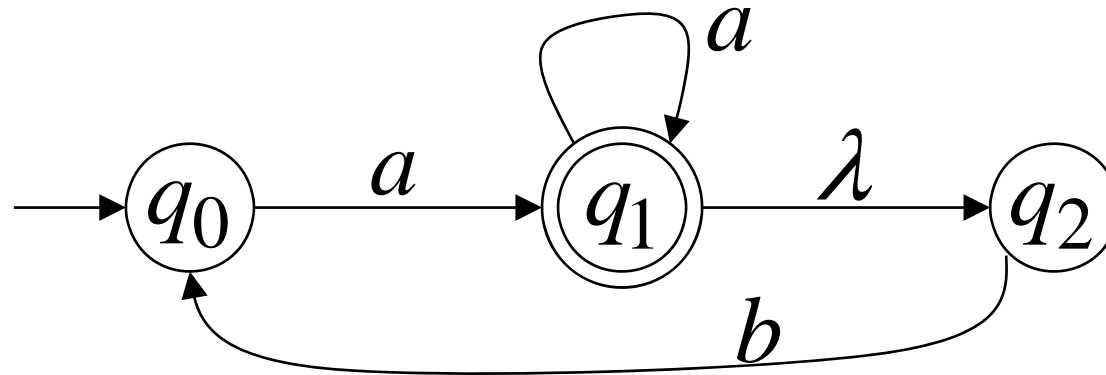


DFA

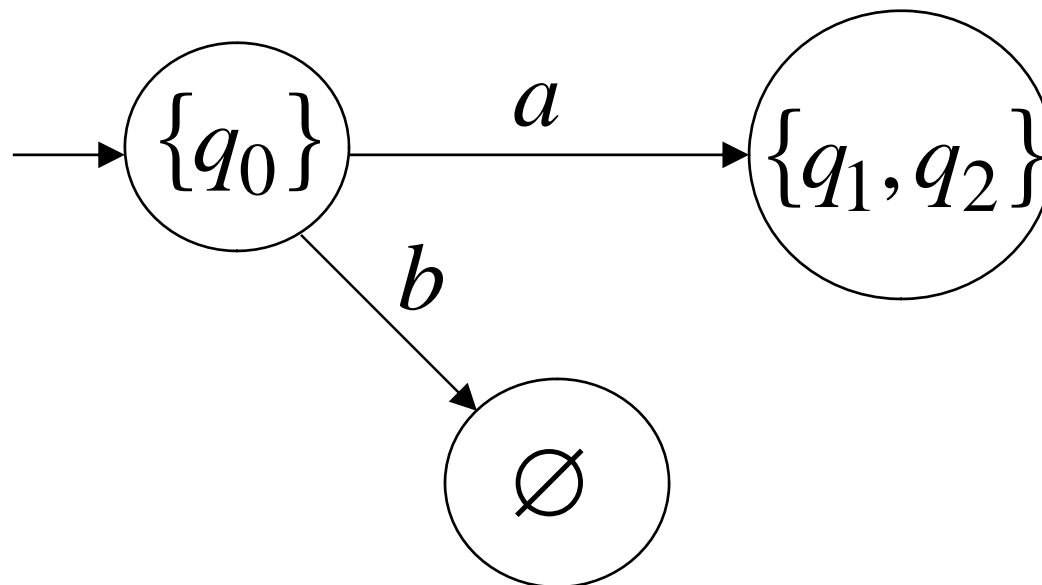


# NFA to DFA

NFA

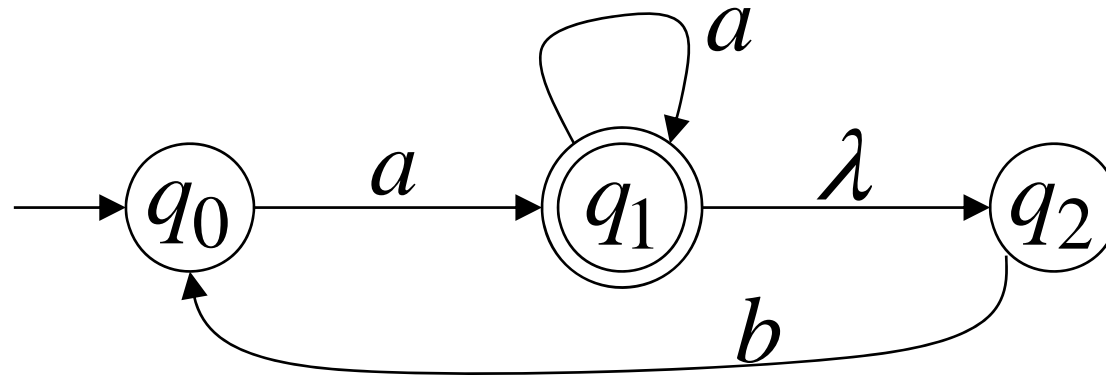


DFA

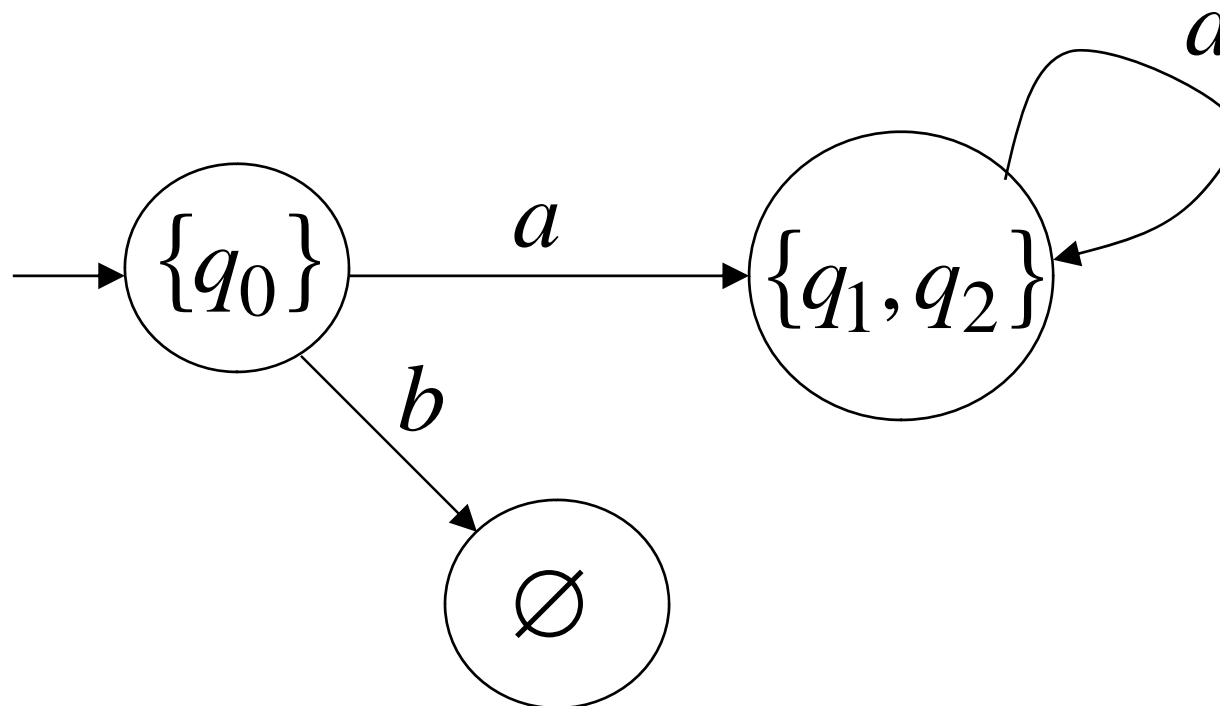


# NFA to DFA

NFA

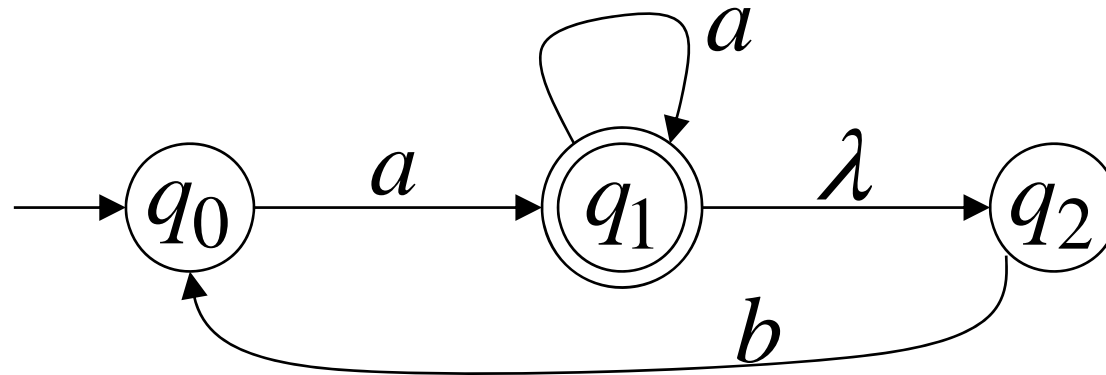


DFA

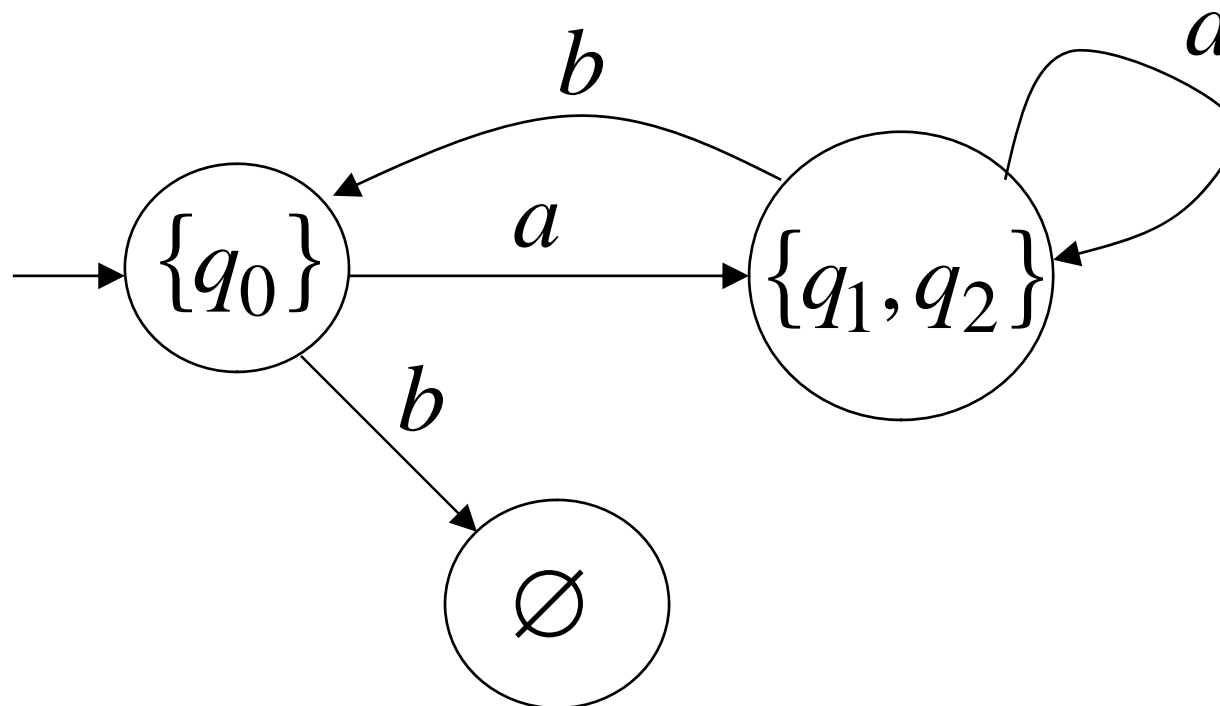


# NFA to DFA

NFA



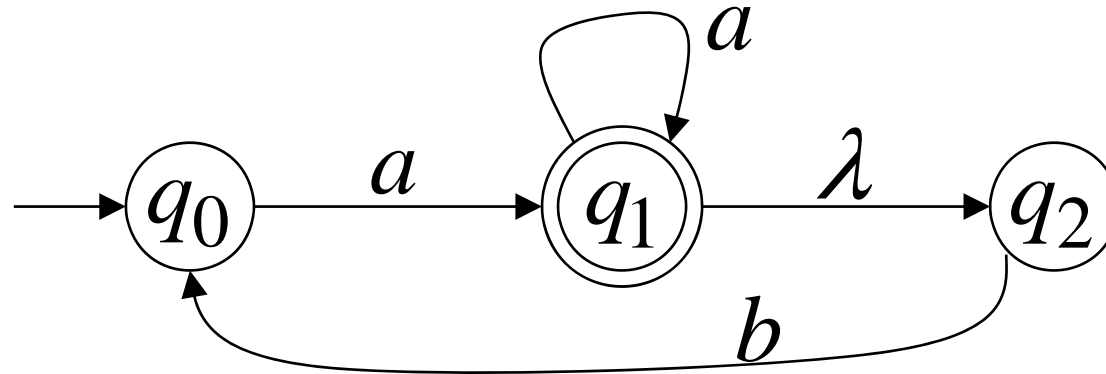
DFA



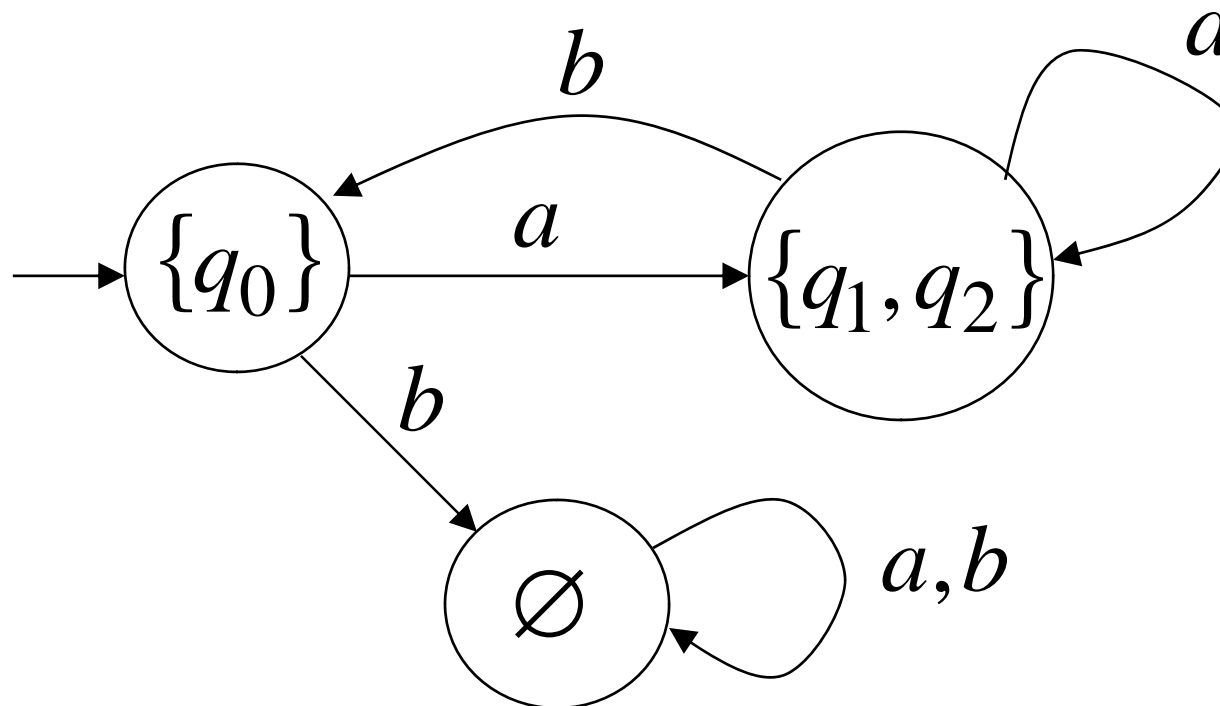


# NFA to DFA

NFA

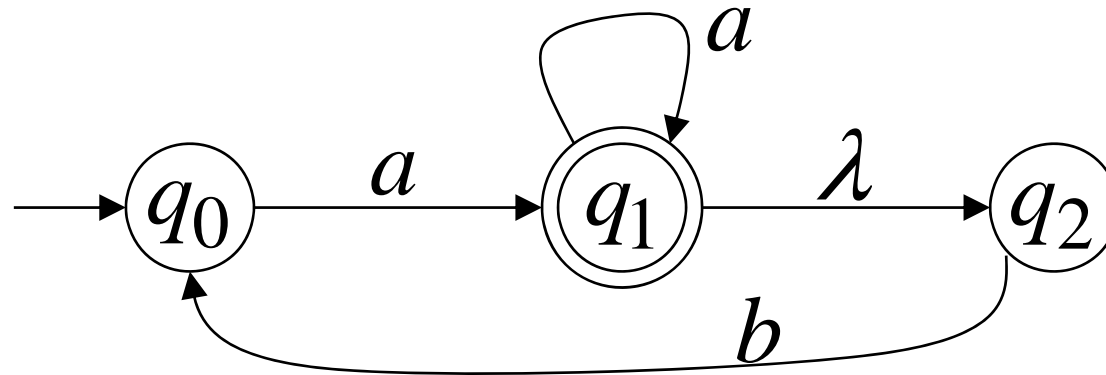


DFA

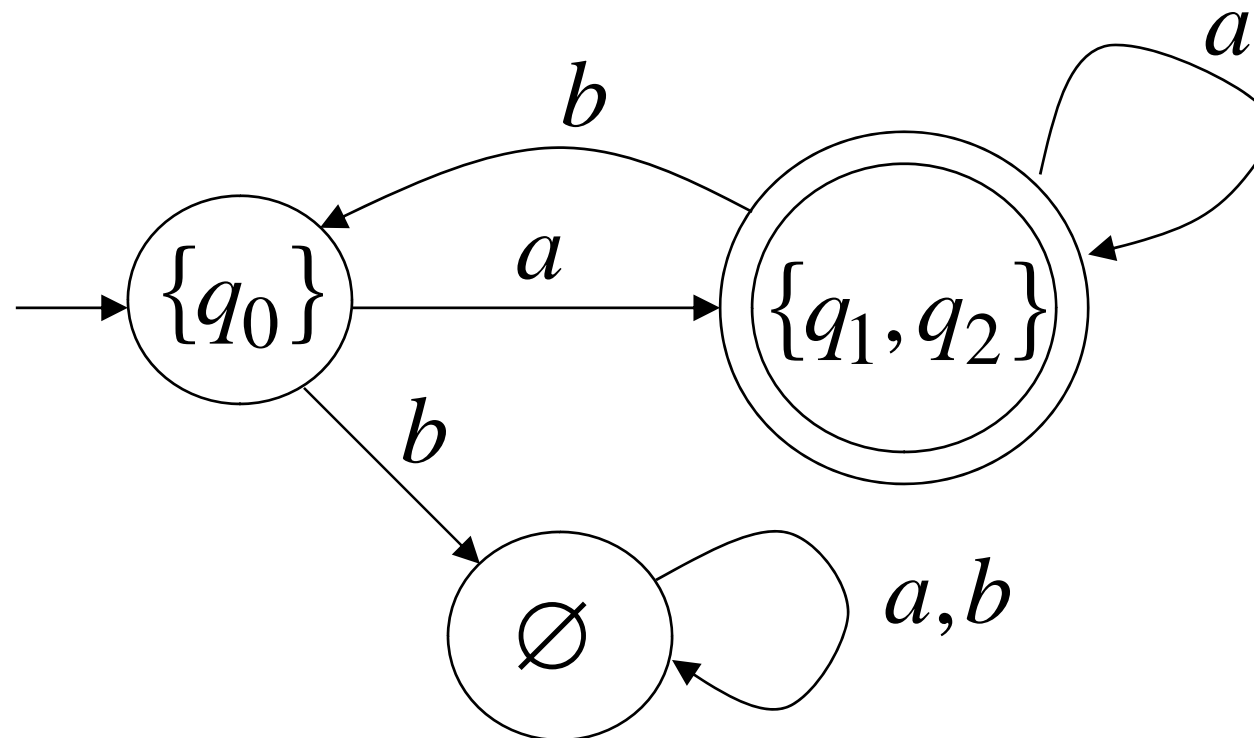


# NFA to DFA

NFA



DFA



# NFA to DFA: Remarks

We are given an NFA  $M$

We want to convert it  
to an equivalent DFA  $M'$

With  $L(M) = L(M')$

If the NFA has states

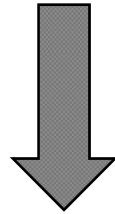
$$q_0, q_1, q_2, \dots$$

the DFA has states in the powerset

$$\emptyset, \{q_0\}, \{q_1\}, \{q_1, q_2\}, \{q_3, q_4, q_7\}, \dots$$

# Procedure NFA to DFA

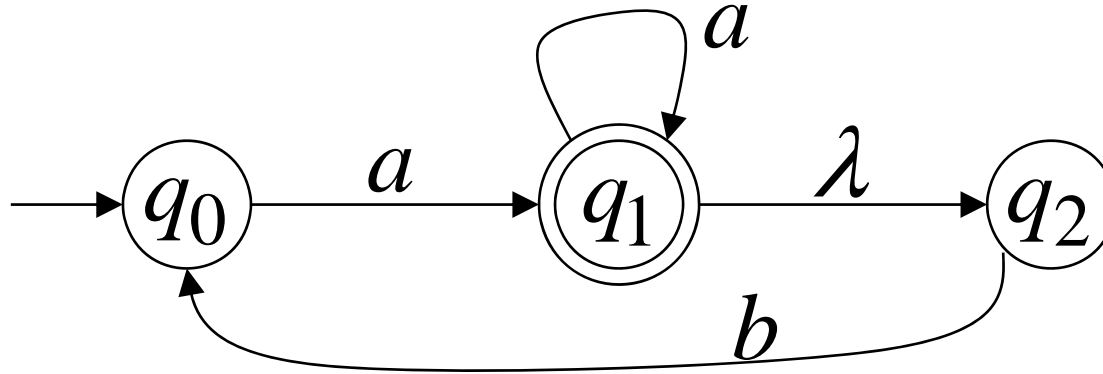
1. Initial state of NFA:  $q_0$



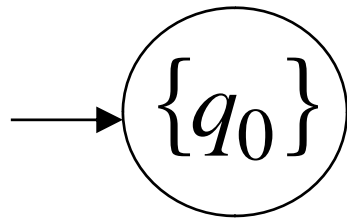
Initial state of DFA:  $\{q_0\}$

# Example

NFA



DFA



## Procedure NFA to DFA

2. For every DFA's state  $\{q_i, q_j, \dots, q_m\}$

Compute in the NFA

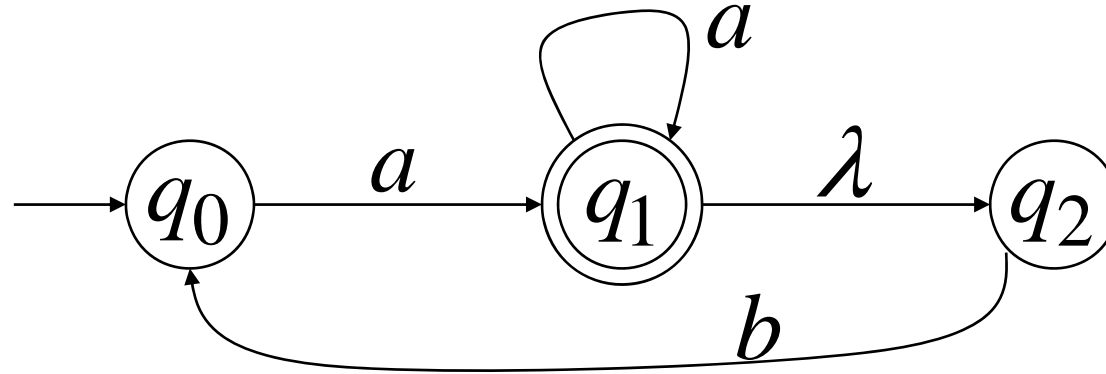
$$\left. \begin{array}{l} \delta^*(q_i, a), \\ \delta^*(q_j, a), \\ \dots \end{array} \right\} = \{q'_i, q'_j, \dots, q'_m\}$$

Add transition

$$\delta(\{q_i, q_j, \dots, q_m\}, a) = \{q'_i, q'_j, \dots, q'_m\}$$

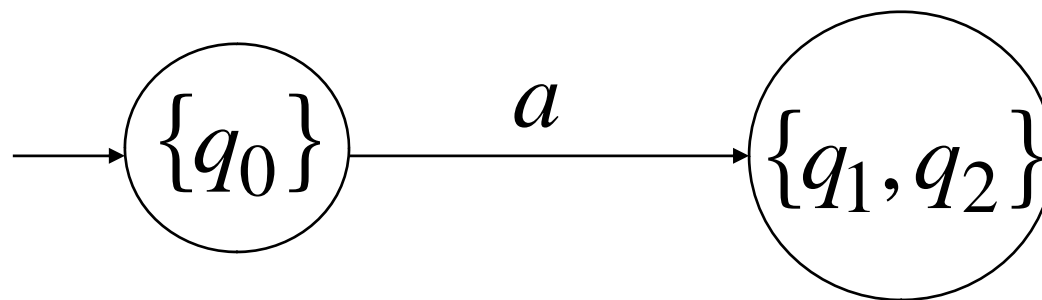
# Example

NFA



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

DFA



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

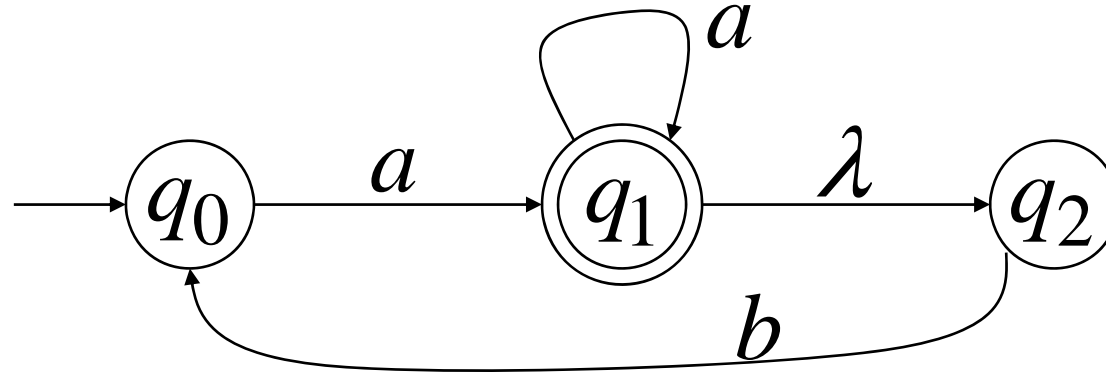


# Procedure NFA to DFA

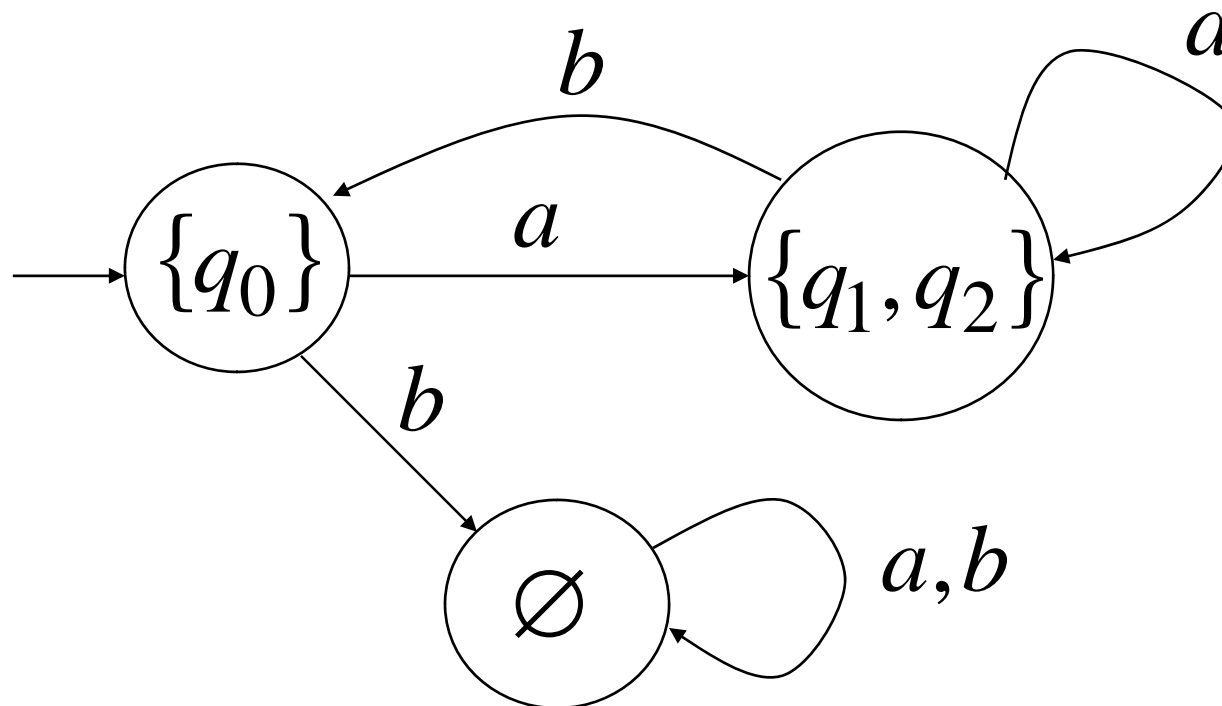
Repeat Step 2 for all letters in alphabet,  
until  
no more transitions can be added.

# Example

NFA



DFA



## Procedure NFA to DFA

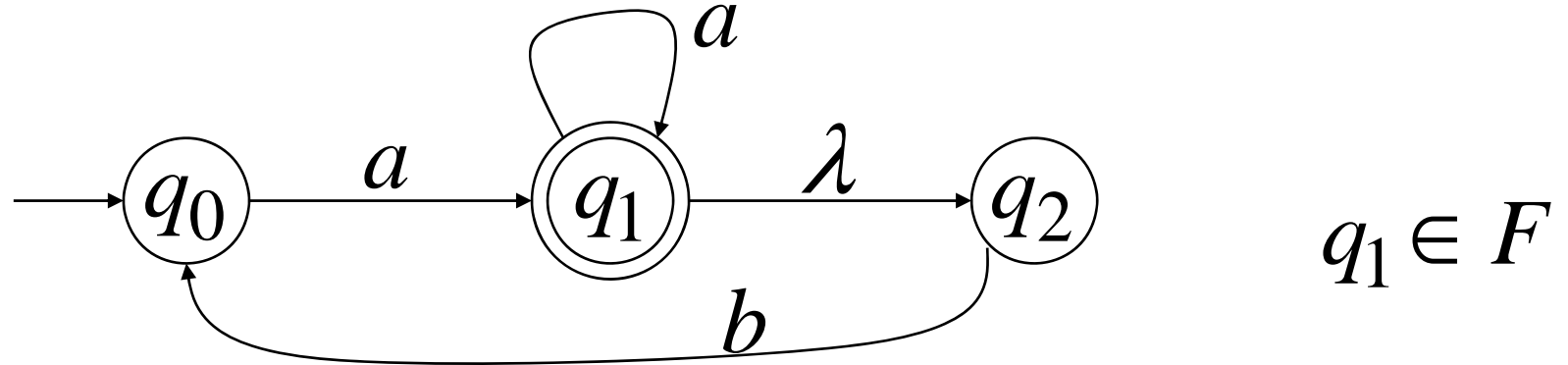
3. For any DFA state  $\{q_i, q_j, \dots, q_m\}$

If some  $q_j$  is a final state in the NFA

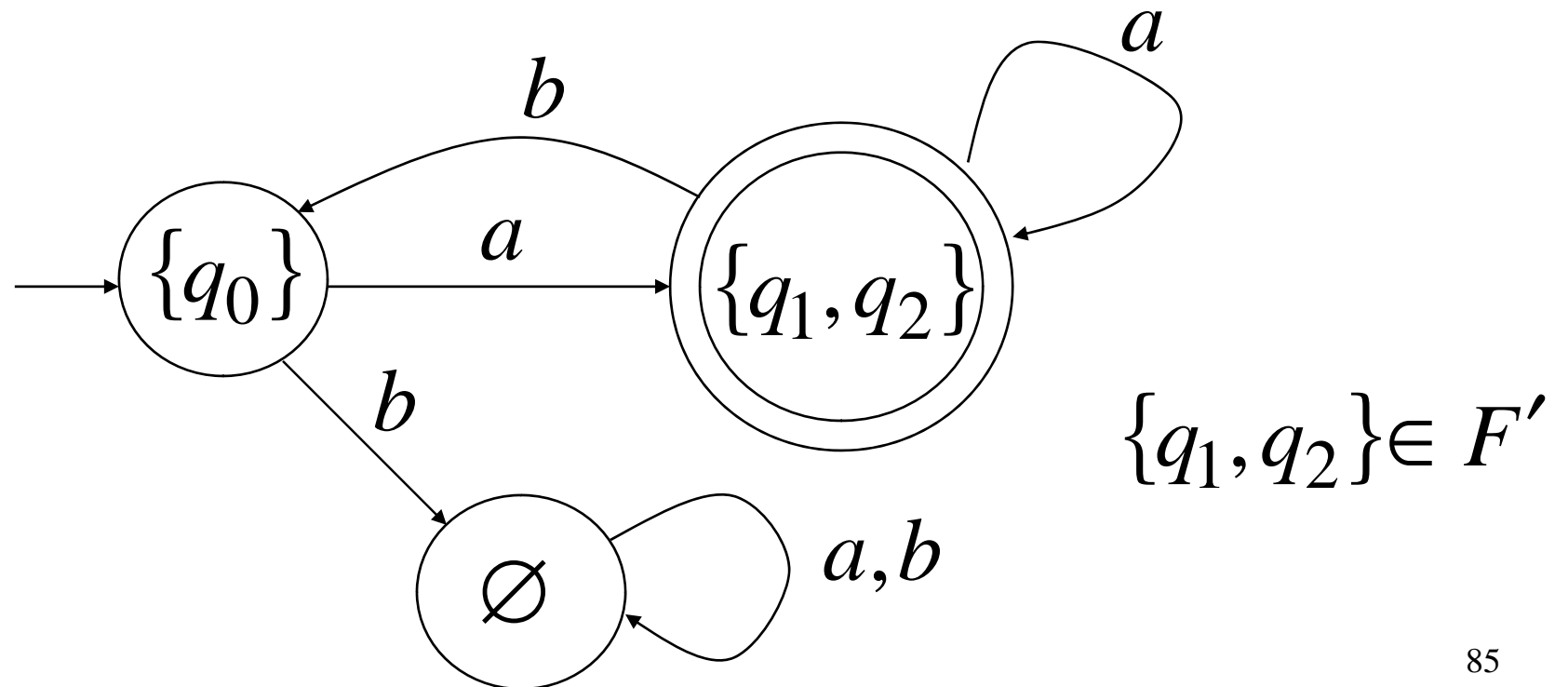
Then,  $\{q_i, q_j, \dots, q_m\}$   
is a final state in the DFA

# Example

NFA



DFA



# Theorem

Take NFA  $M$

Apply procedure to obtain DFA  $M'$

Then  $M$  and  $M'$  are equivalent :

$$L(M) = L(M')$$

# Finally

We have proven

$$\left\{ \begin{array}{l} \text{Languages} \\ \text{accepted} \\ \text{by NFAs} \end{array} \right\} = \left\{ \begin{array}{l} \text{Languages} \\ \text{accepted} \\ \text{by DFAs} \end{array} \right\}$$

We have proven

$$\left\{ \begin{array}{l} \text{Languages} \\ \text{accepted} \\ \text{by NFAs} \end{array} \right\} = \left\{ \begin{array}{l} \text{Languages} \\ \text{accepted} \\ \text{by DFAs} \end{array} \right\}$$

Regular Languages

We have proven

$$\left\{ \begin{array}{l} \text{Languages} \\ \text{accepted} \\ \text{by NFAs} \end{array} \right\} = \left\{ \begin{array}{l} \text{Languages} \\ \text{accepted} \\ \text{by DFAs} \end{array} \right\}$$

Regular Languages

Regular Languages