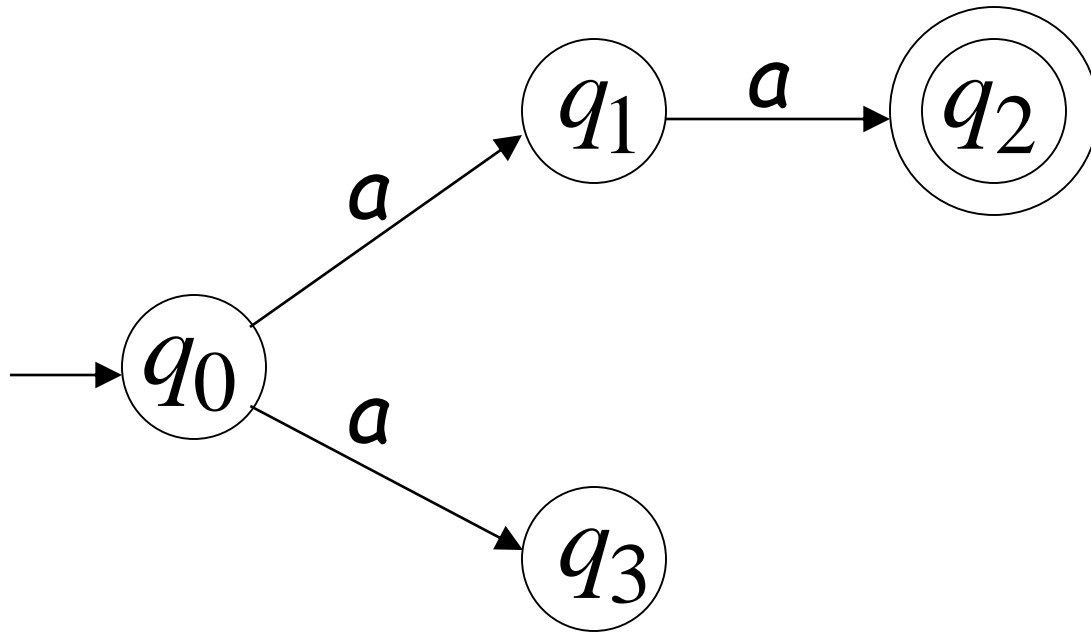


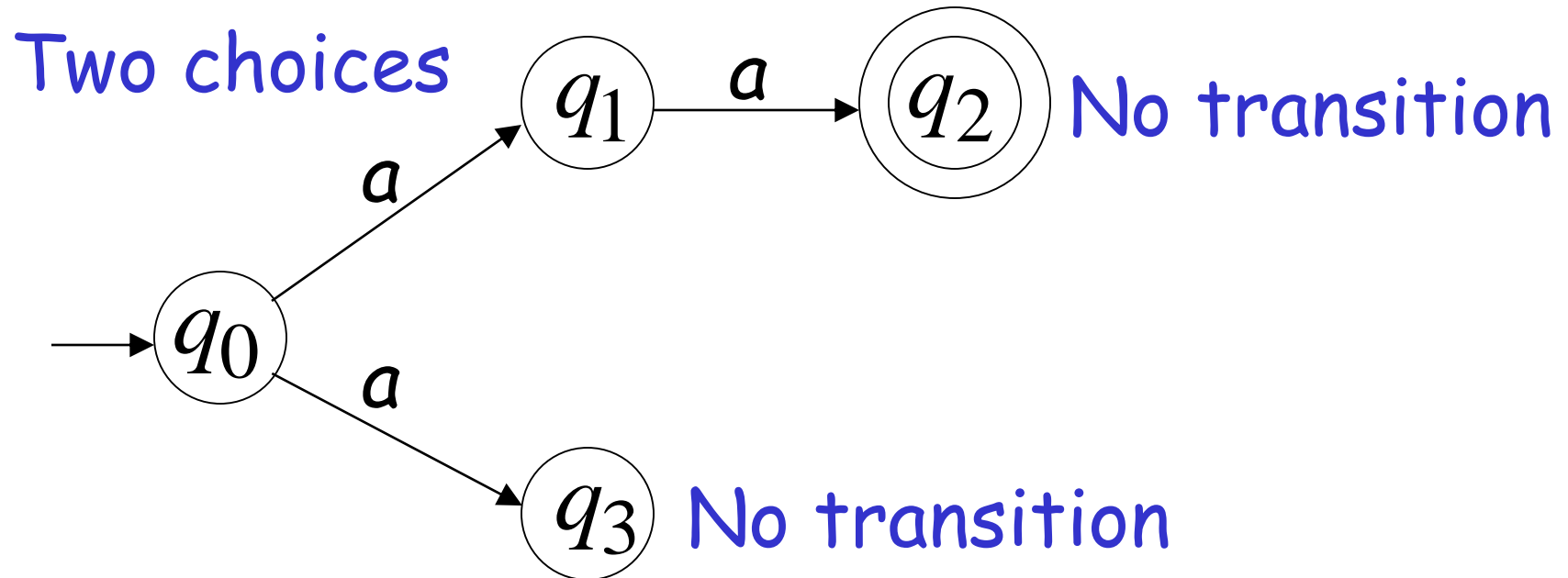
Non-Deterministic Finite Automata

Nondeterministic Finite Automaton (NFA)

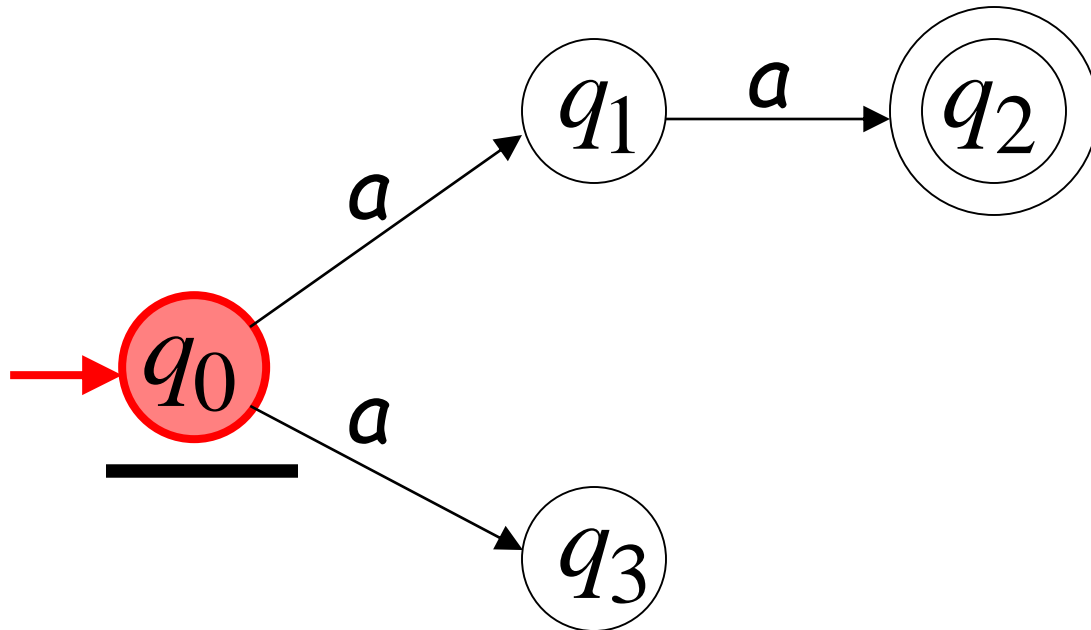
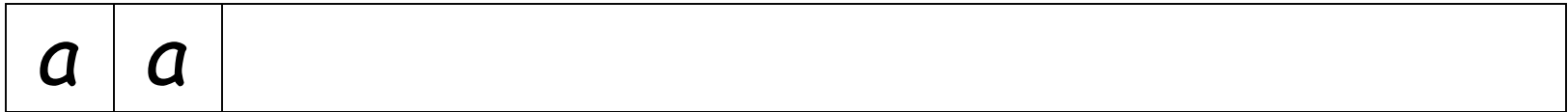
Alphabet = $\{a\}$



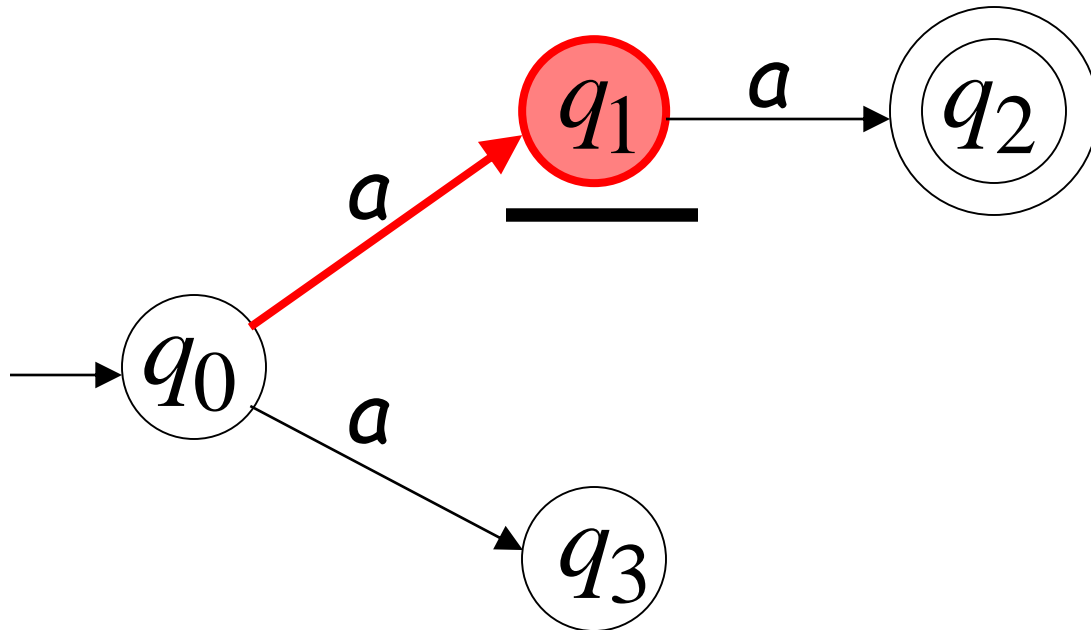
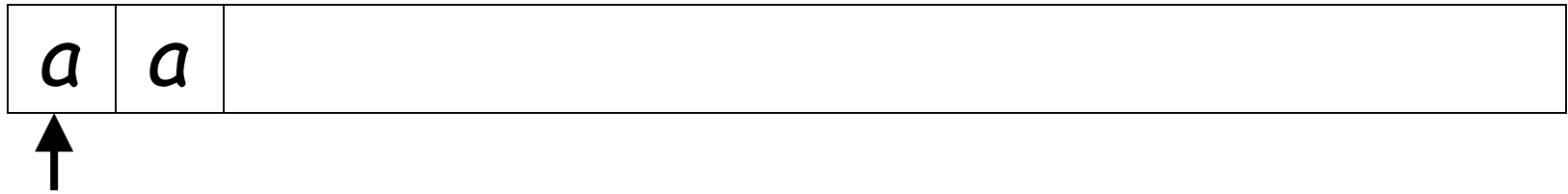
Alphabet = $\{a\}$



First Choice

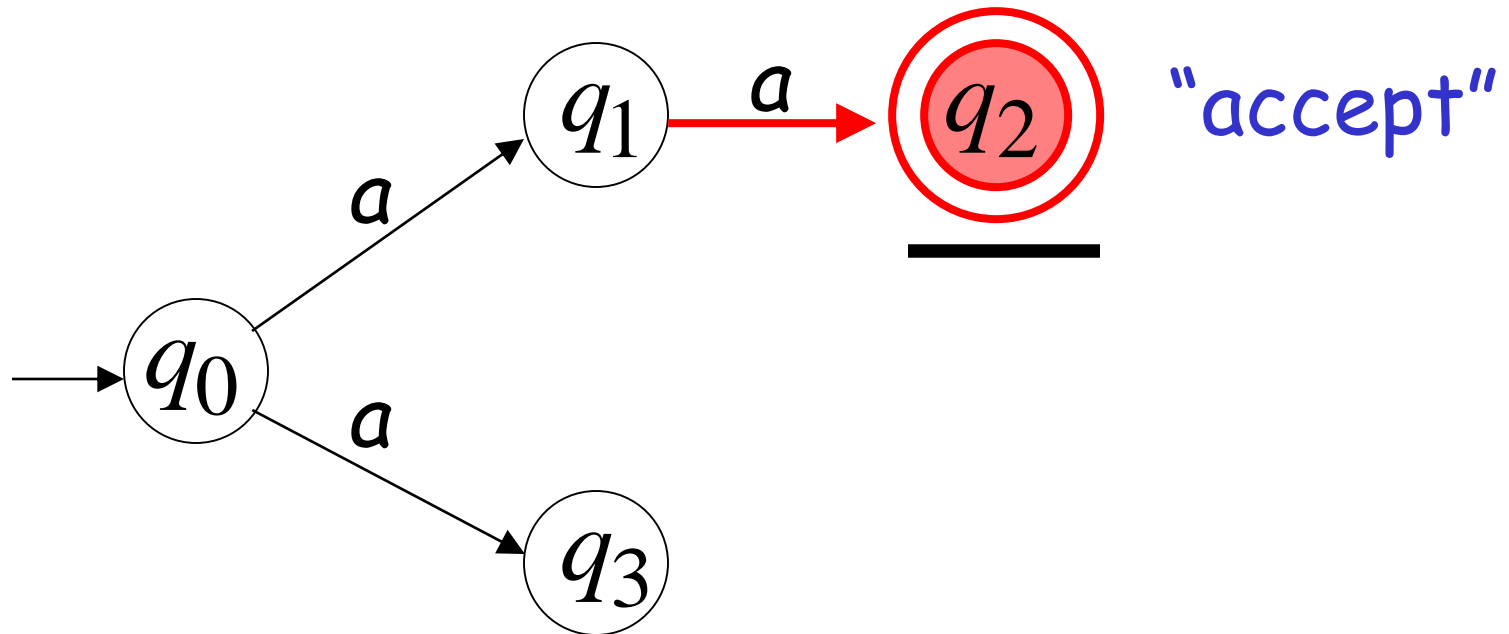
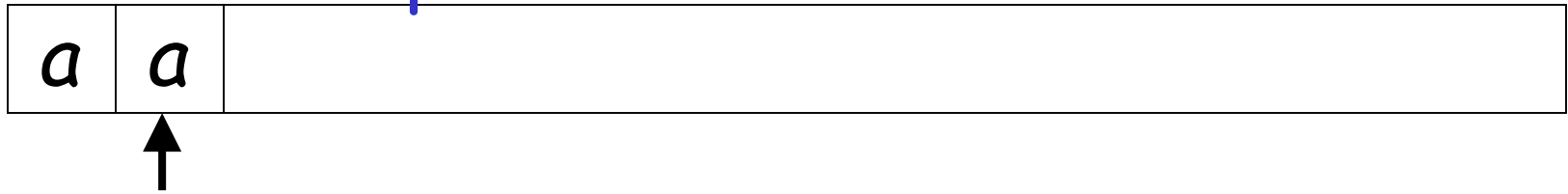


First Choice

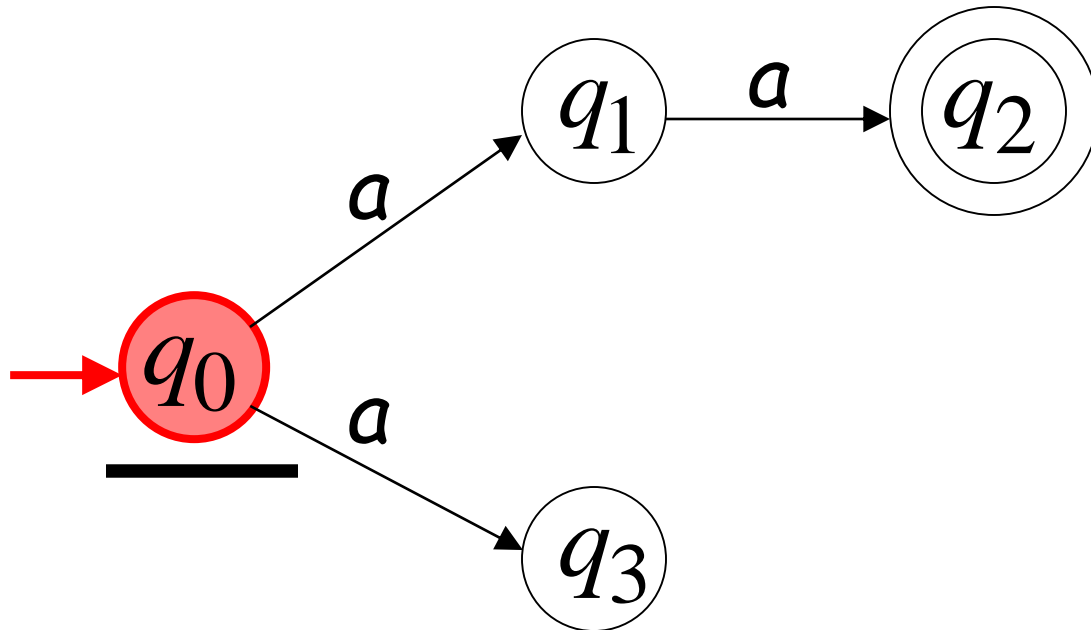


First Choice

All input is consumed

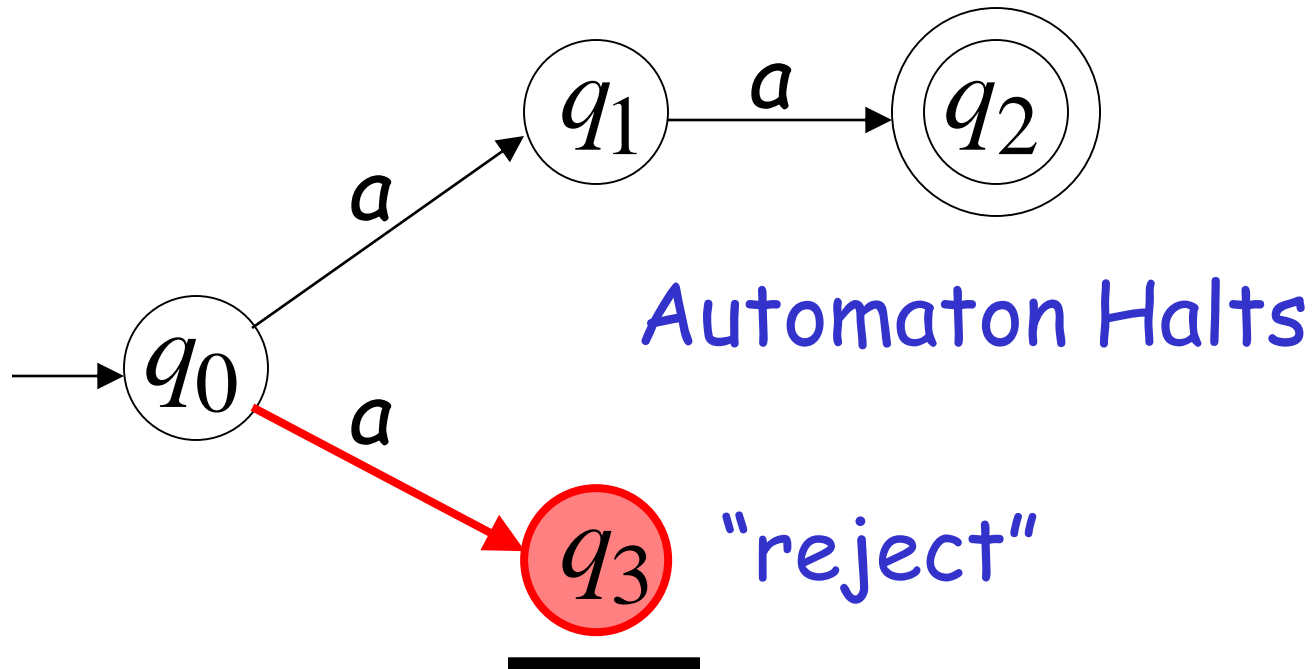
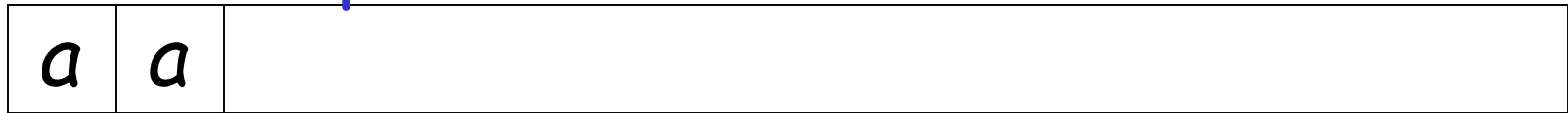


Second Choice



Second Choice

Input cannot be consumed

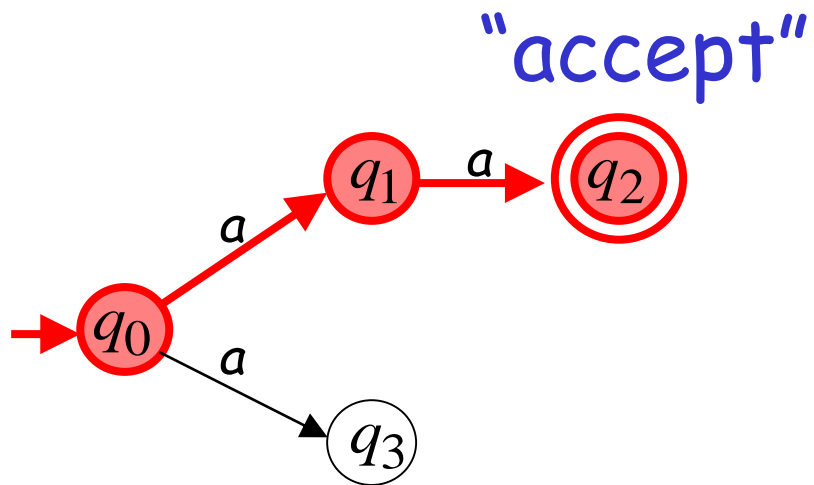


A NFA accepts a string:

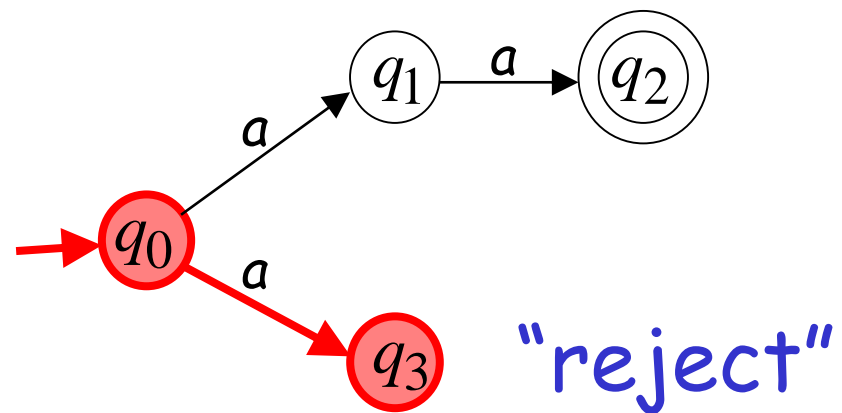
if there is a computation path of the NFA that accepts the string

all the symbols of input string are processed
and
the last is an accepting state

aa is accepted by the NFA:

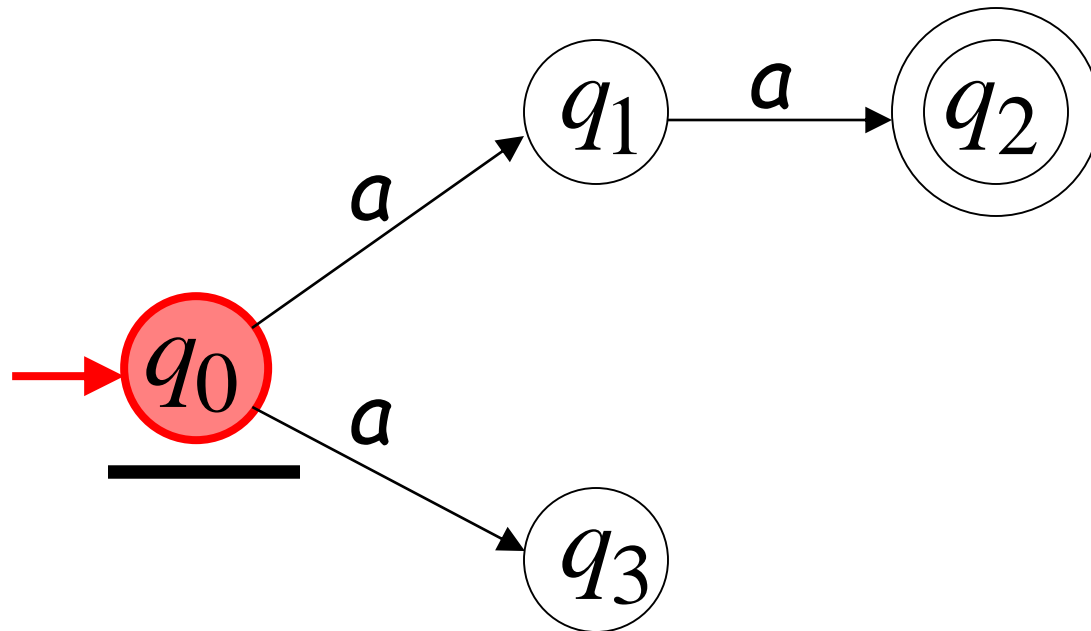
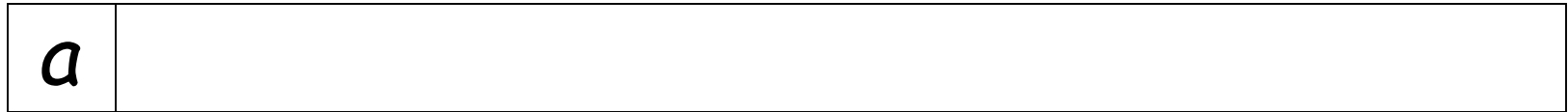


because this
computation
accepts aa

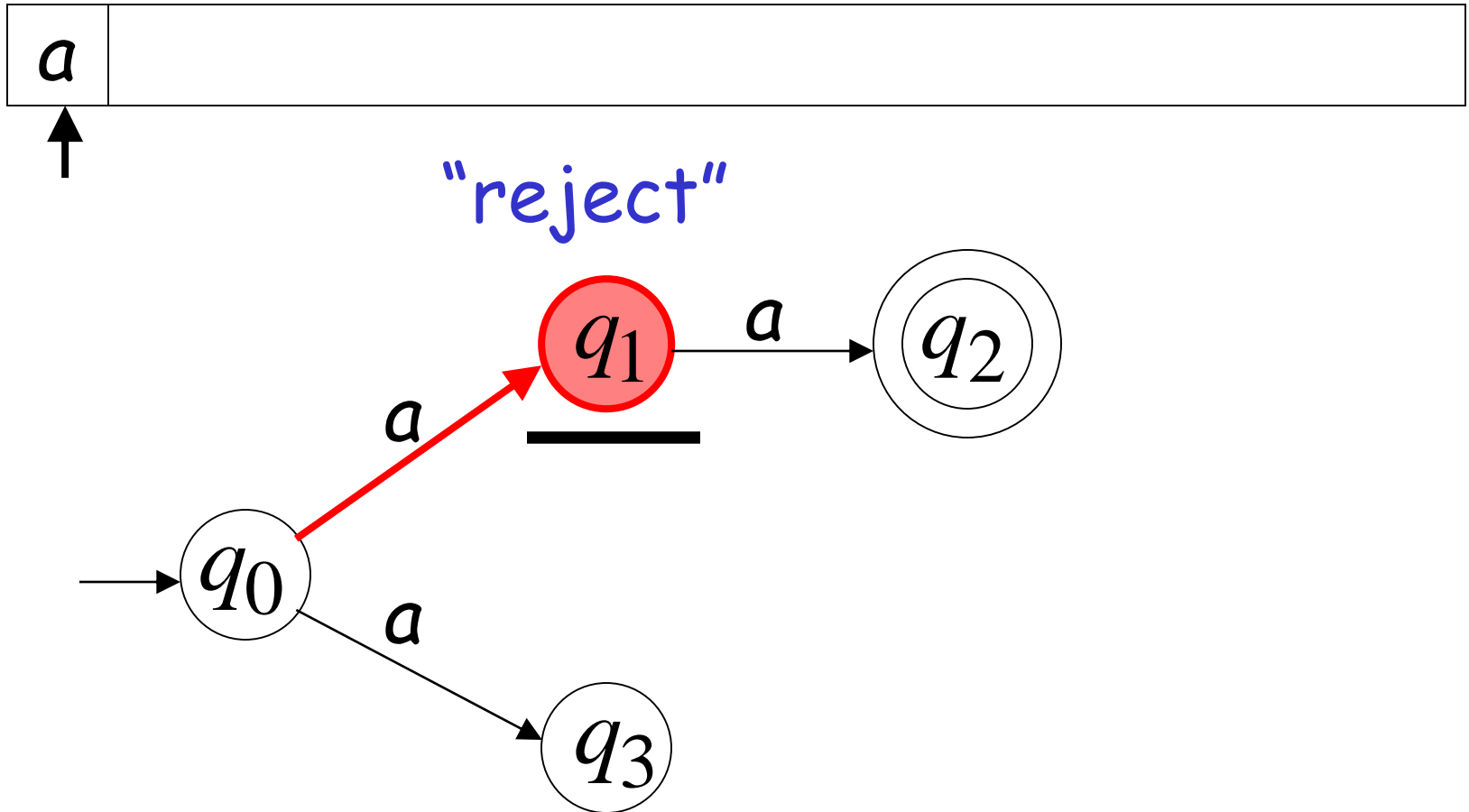


this computation
is ignored

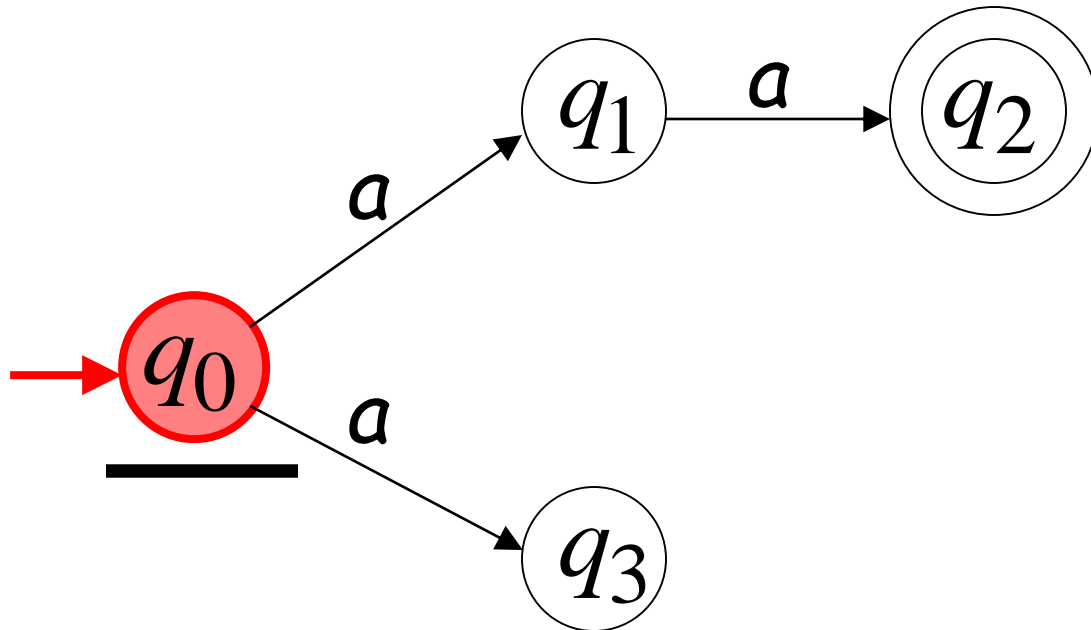
Rejection example



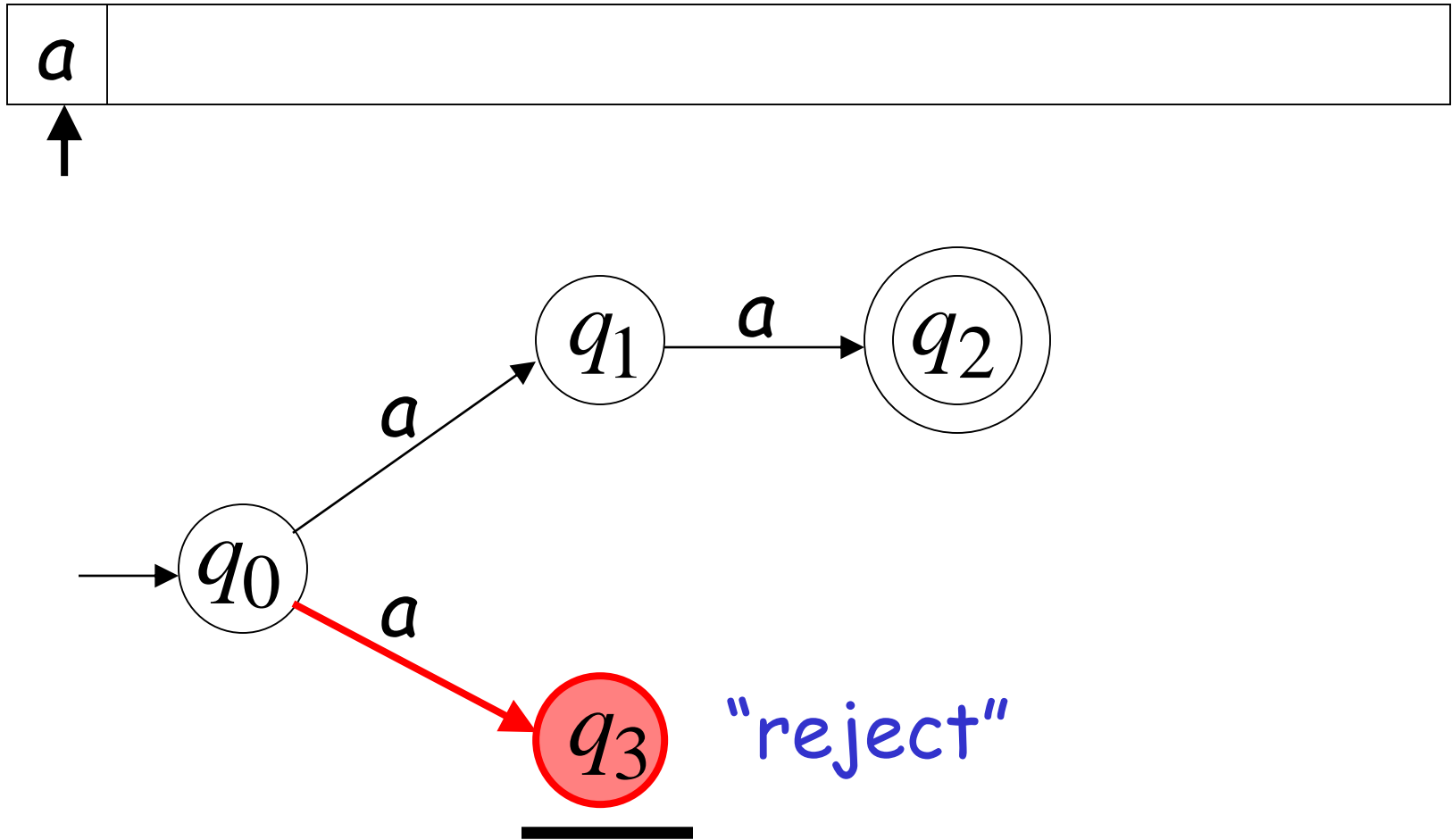
First Choice



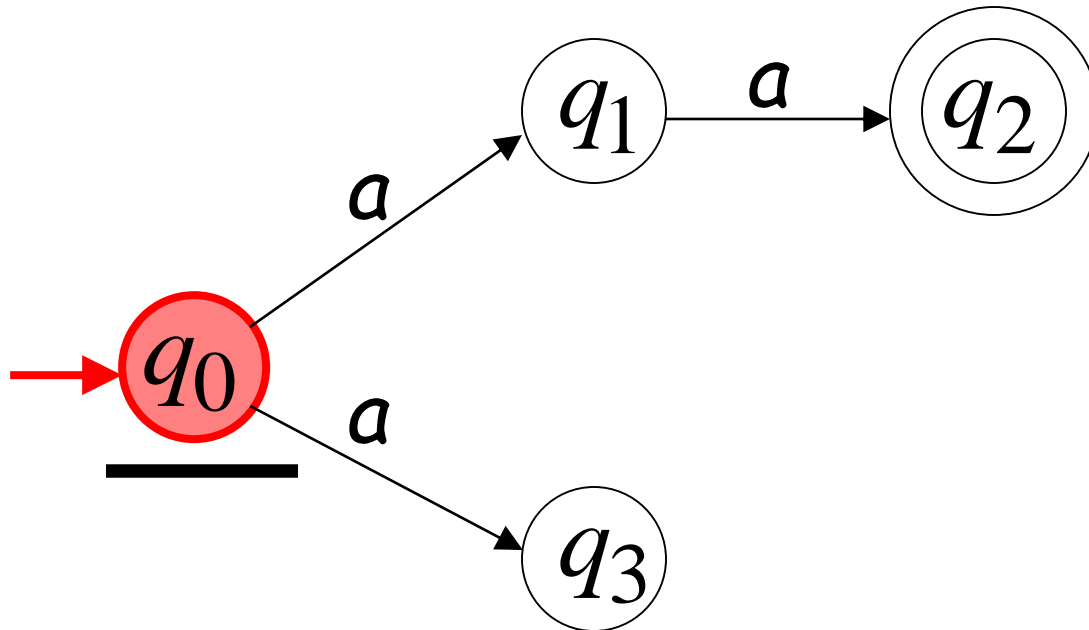
Second Choice



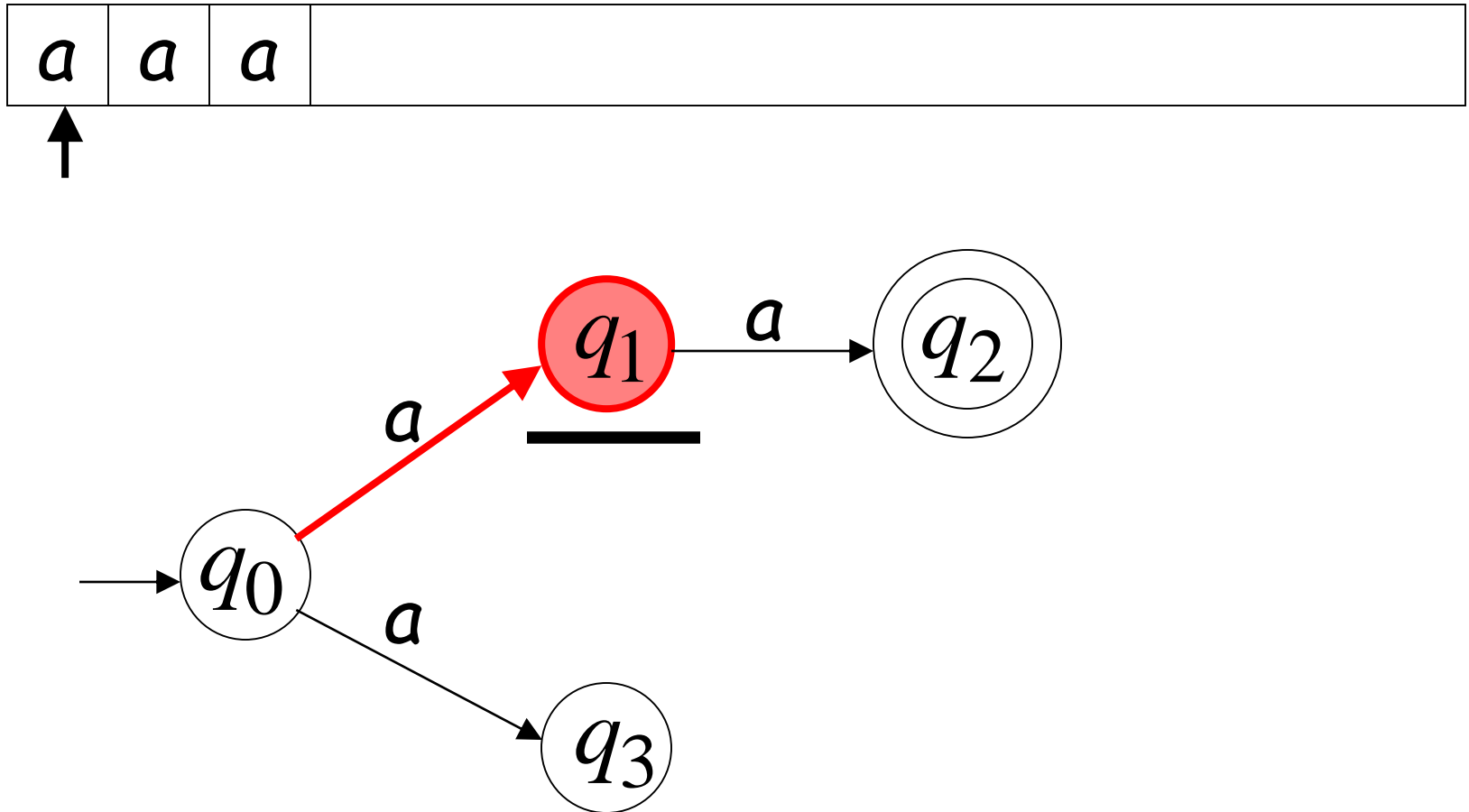
Second Choice



Another Rejection example

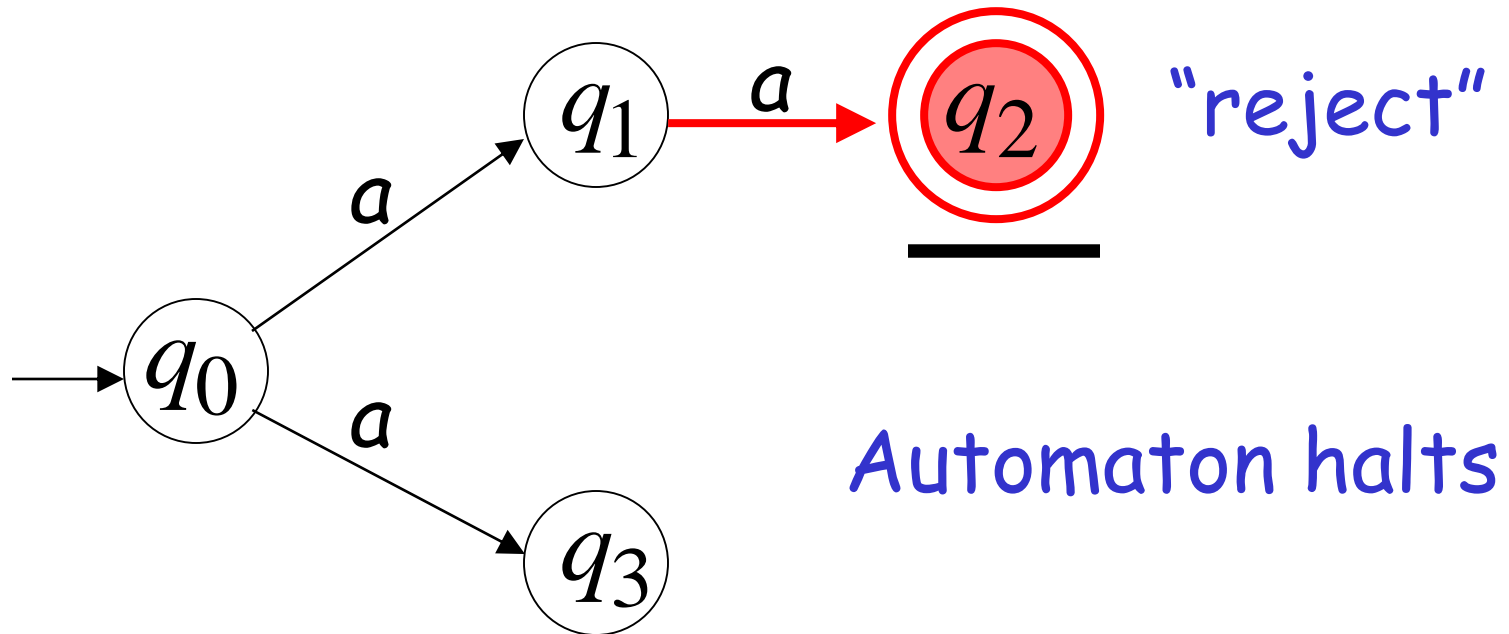


First Choice

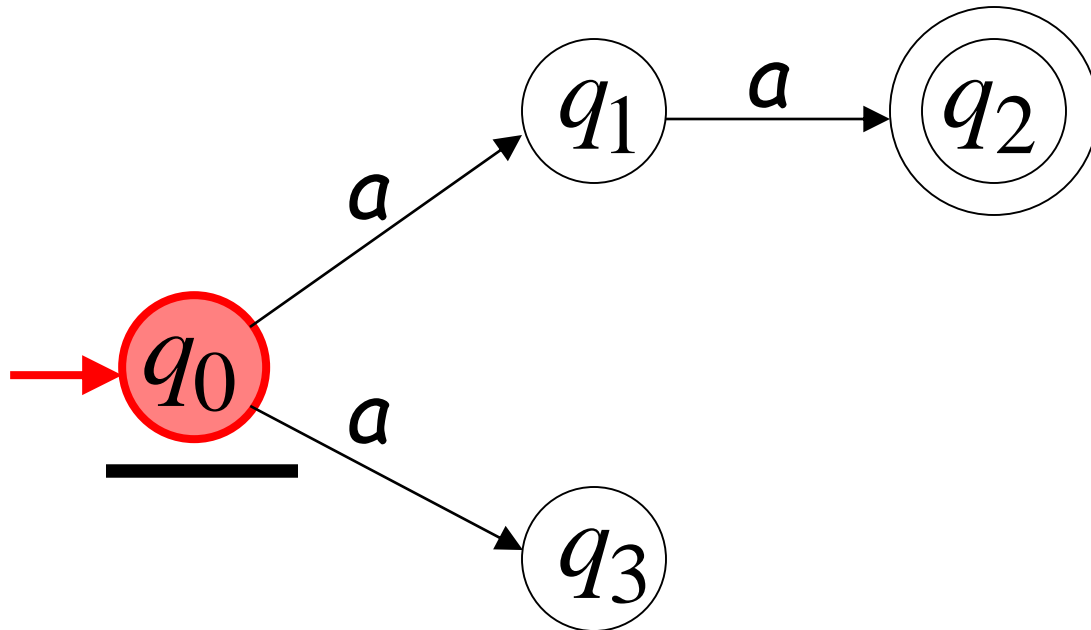


First Choice

Input cannot be consumed

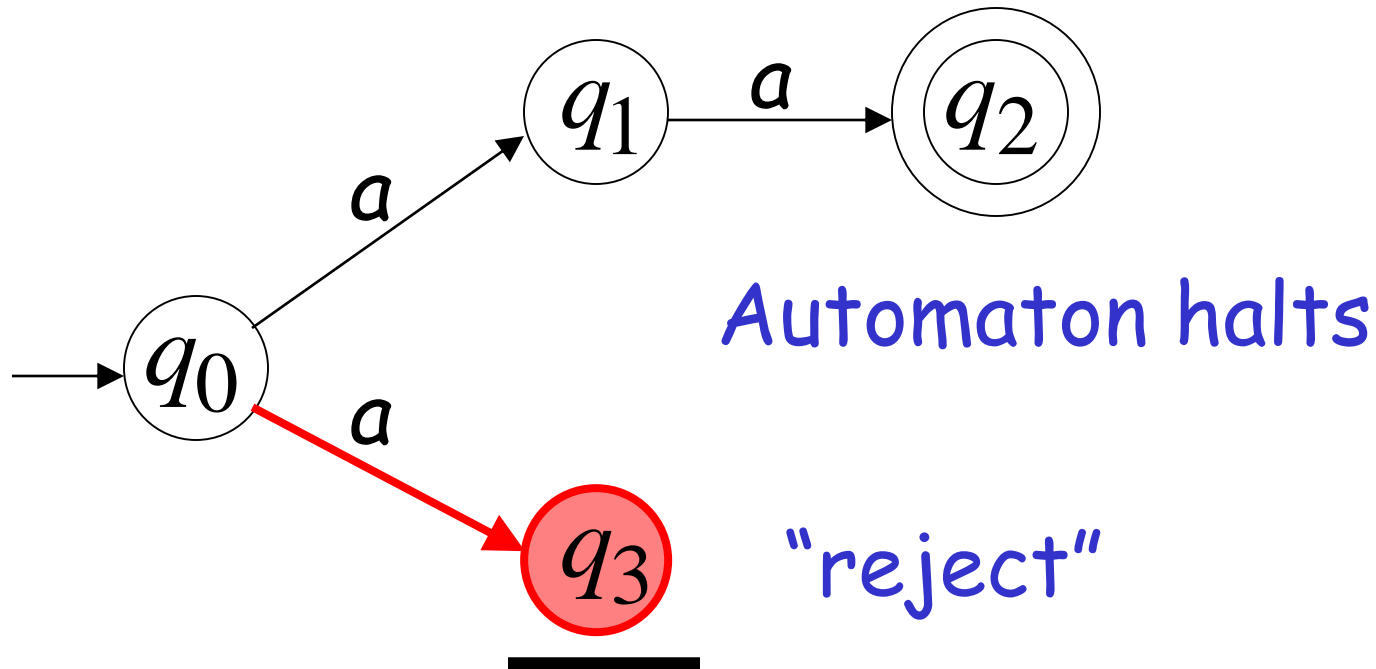


Second Choice



Second Choice

Input cannot be consumed



A NFA rejects a string:

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

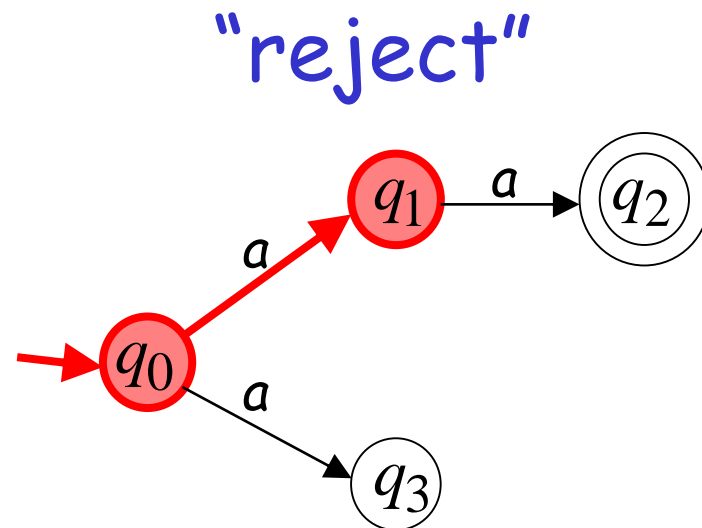
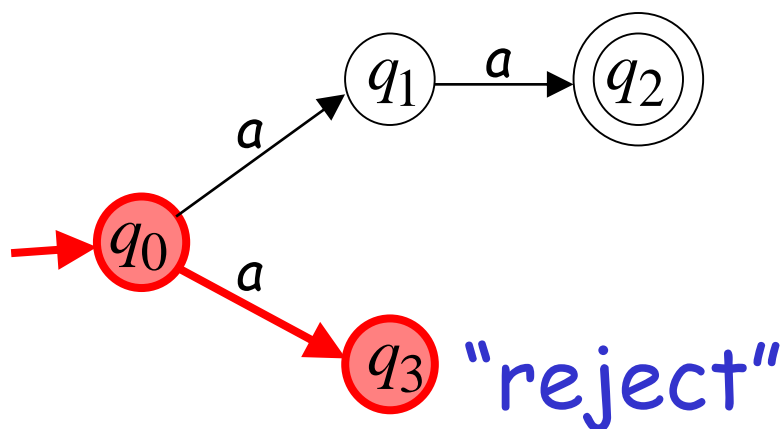
For each possible computation path:

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

OR

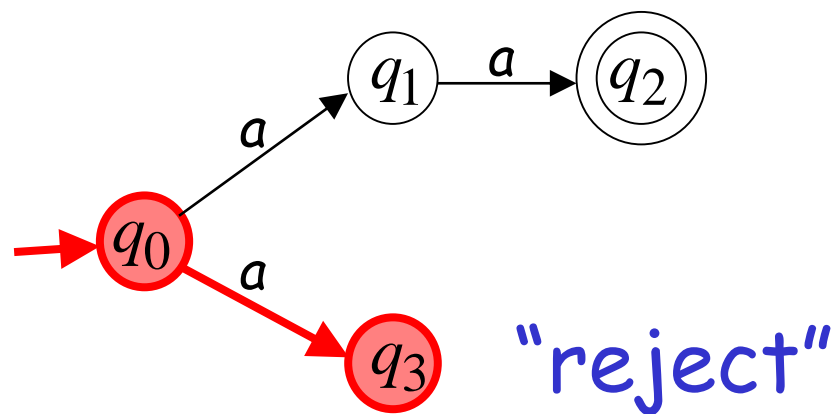
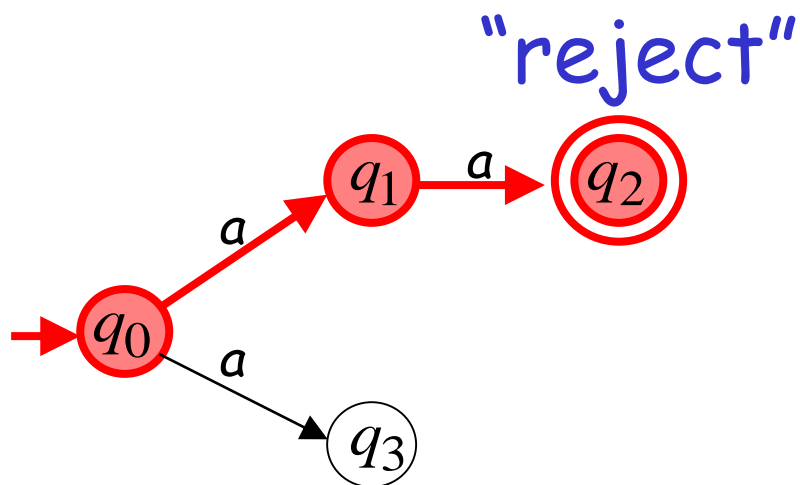
- The input cannot be consumed

a is rejected by the NFA:



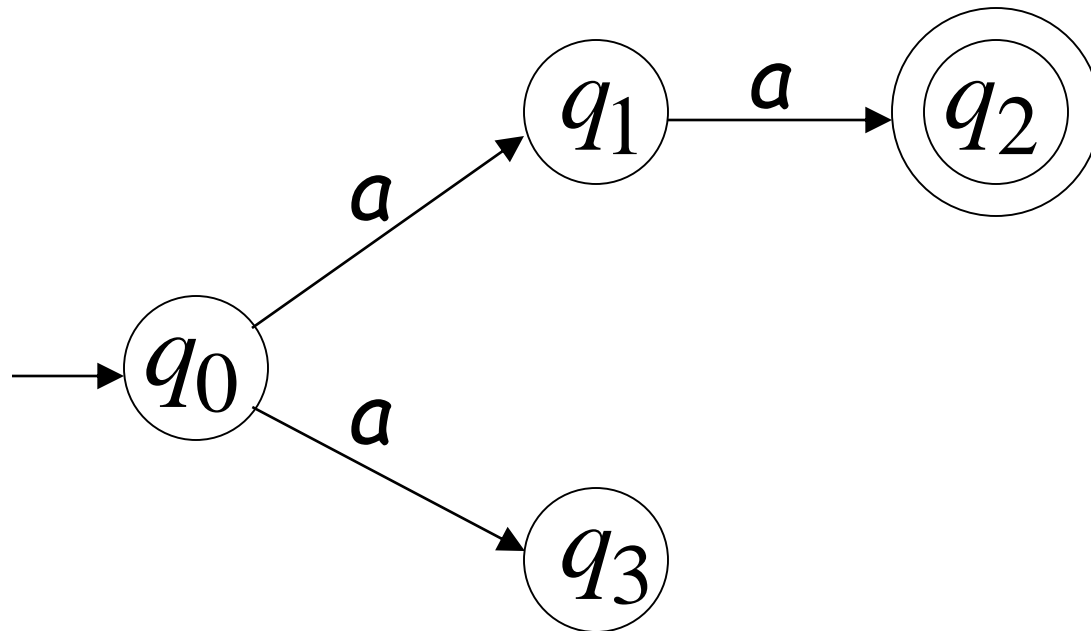
All possible computations lead to rejection

aaa is rejected by the NFA:

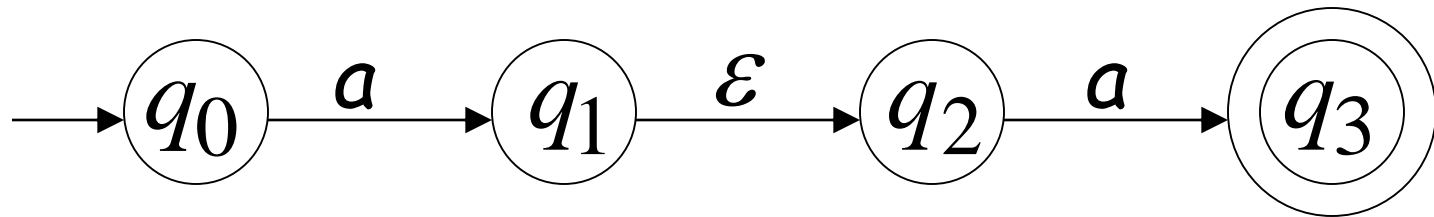


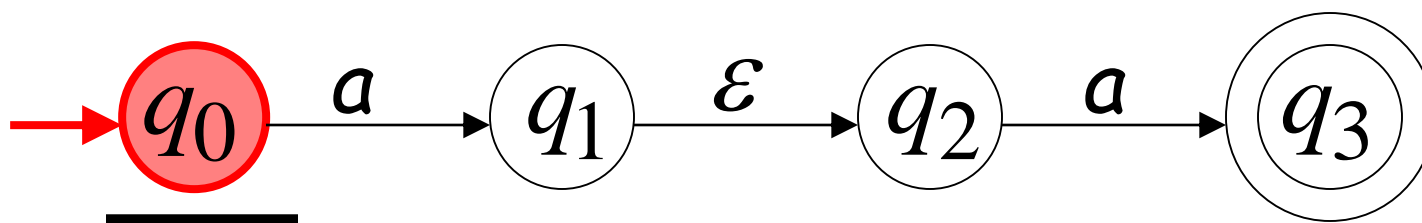
All possible computations lead to rejection

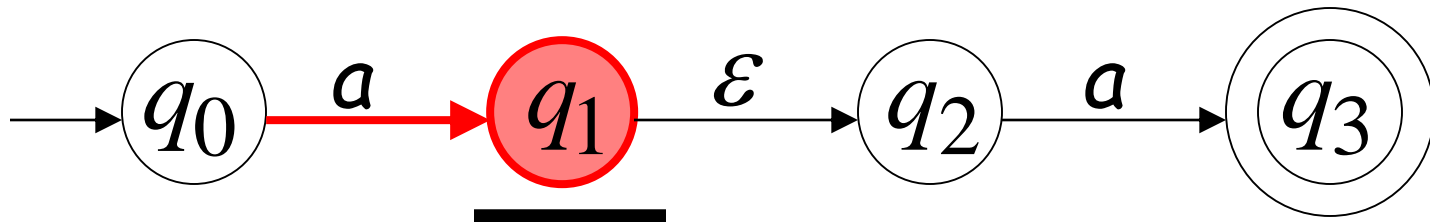
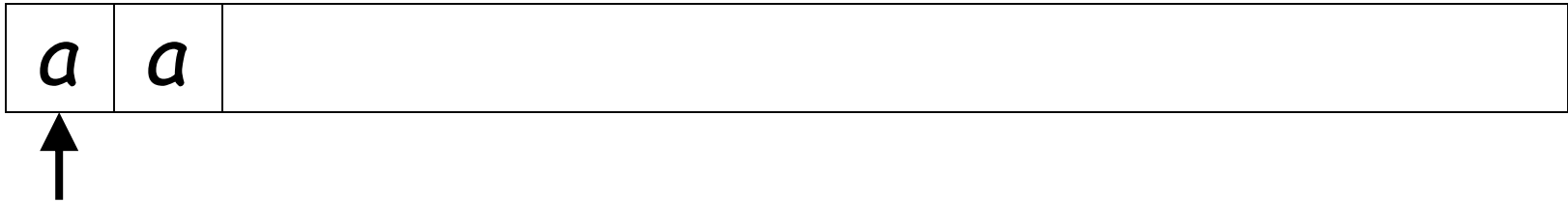
Language accepted: $L = \{aa\}$



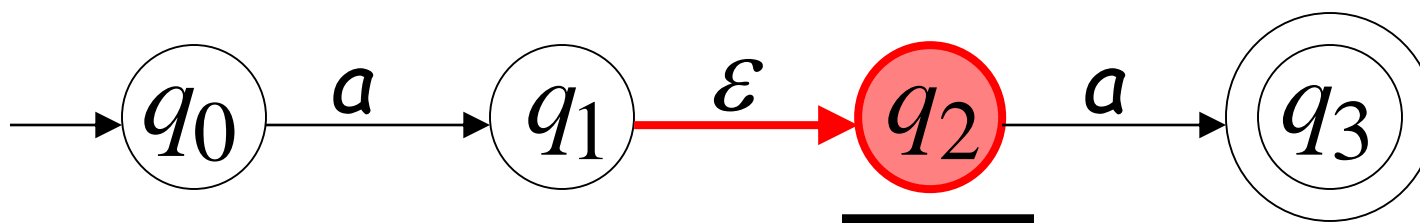
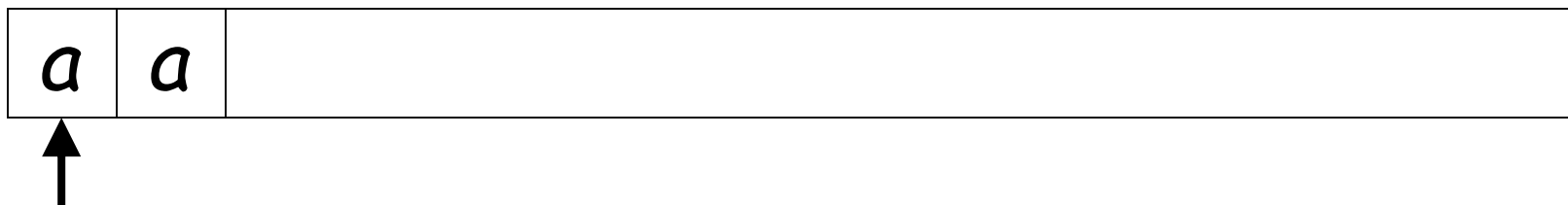
Epsilon Transitions





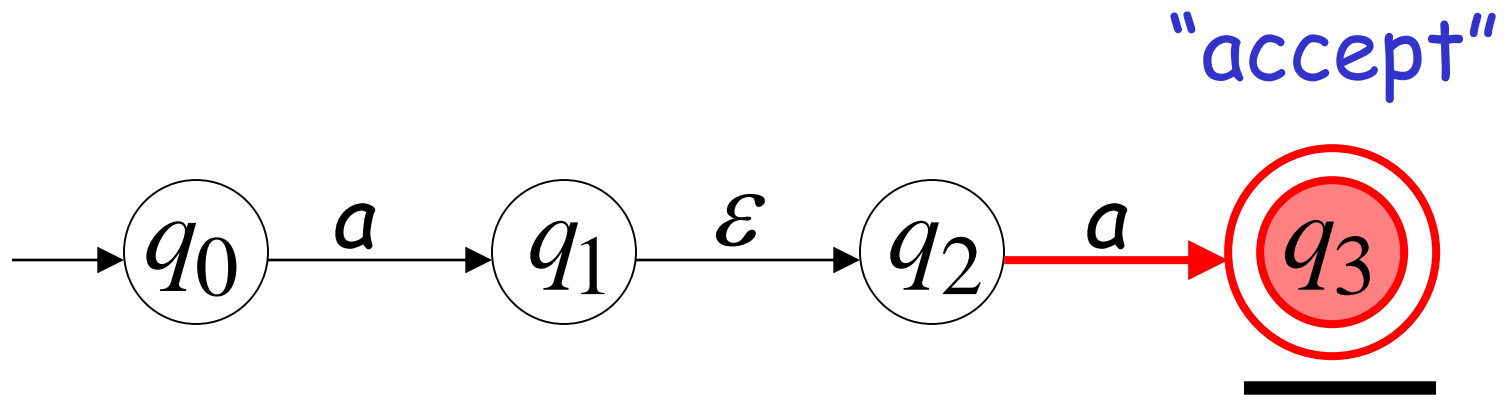


input tape head does not move



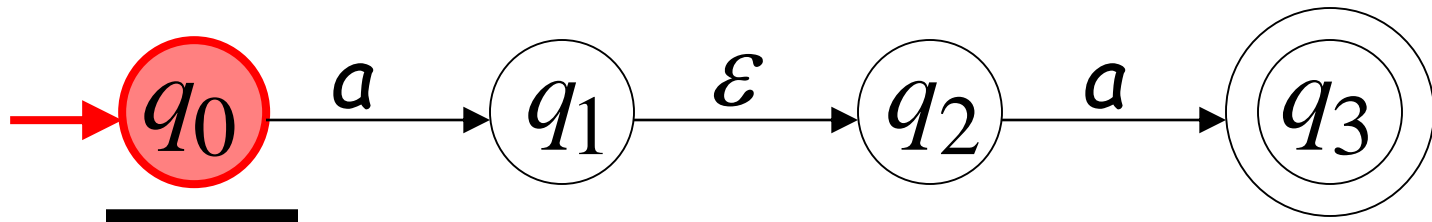
Automaton changes state

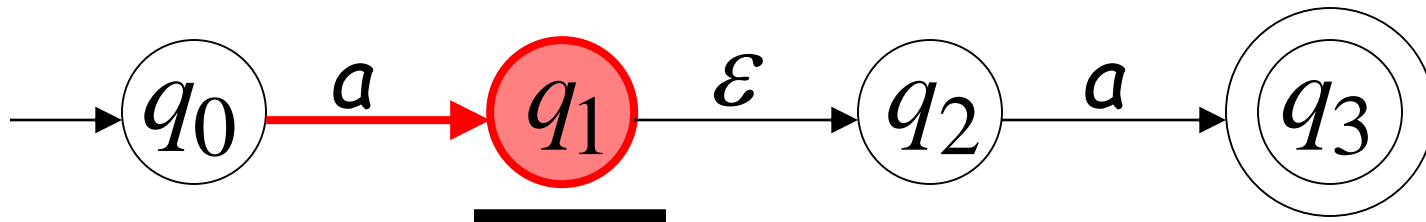
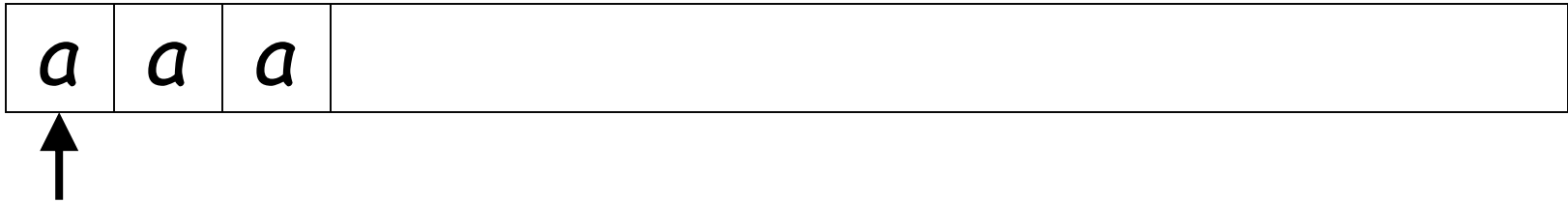
all input is consumed



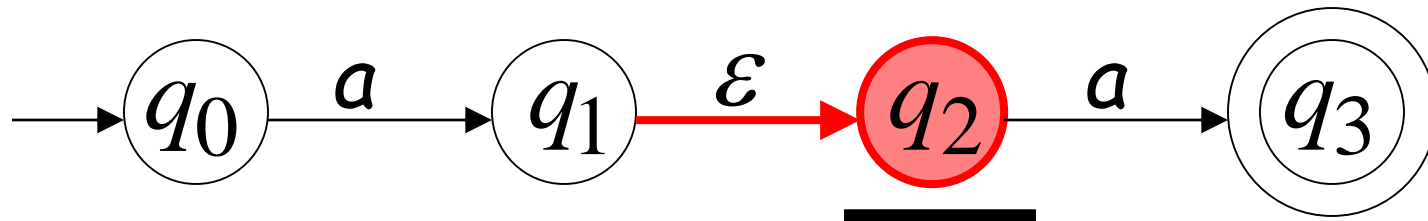
String aa is accepted

Rejection Example





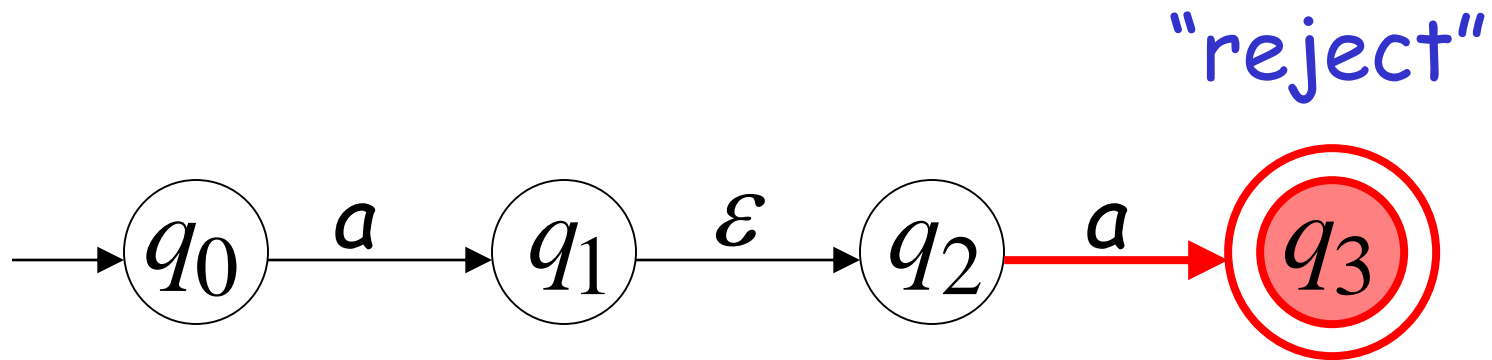
(read head doesn't move)



Input cannot be consumed

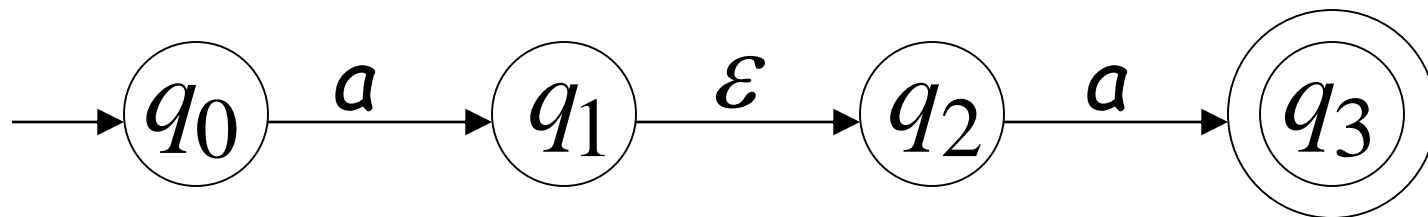


Automaton halts

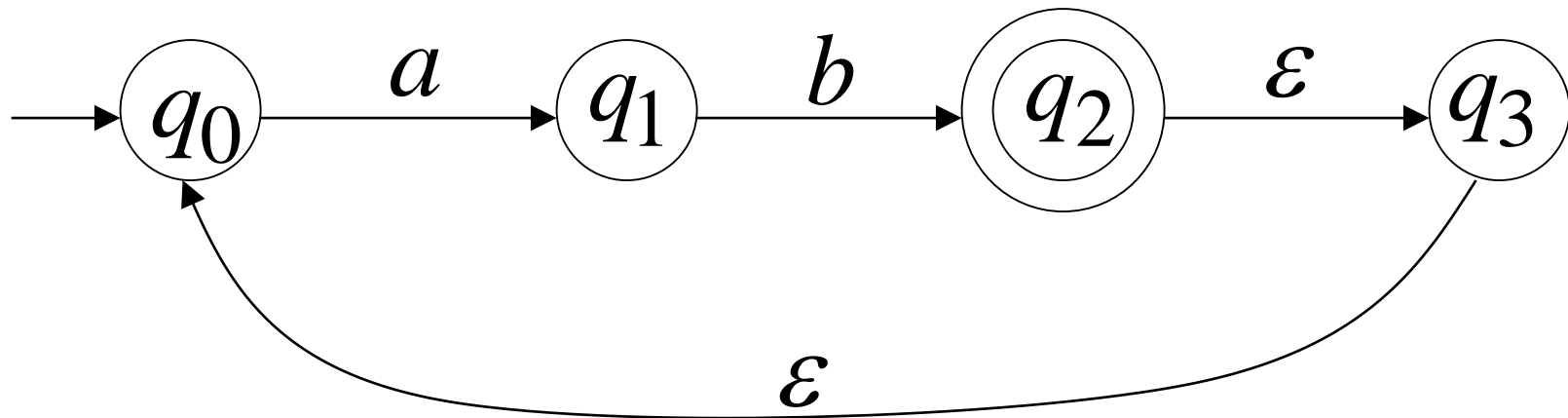


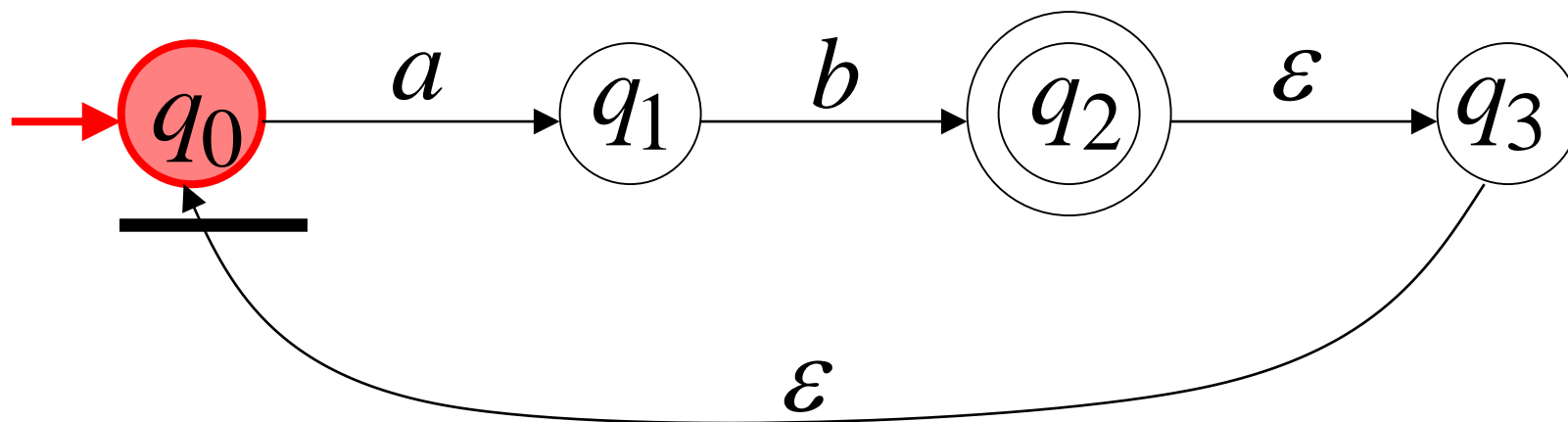
String `aaa` is rejected

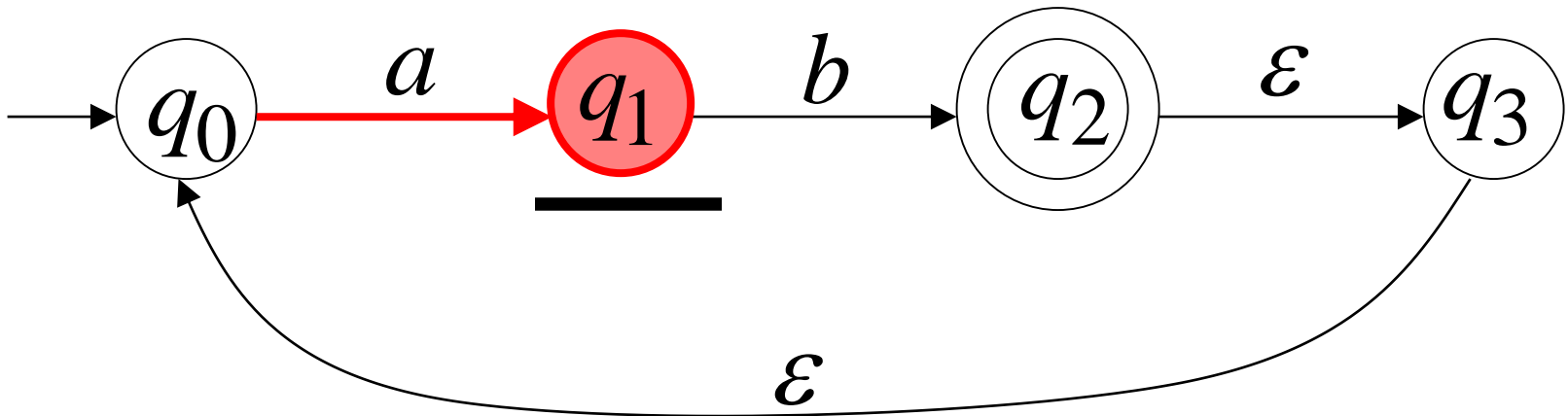
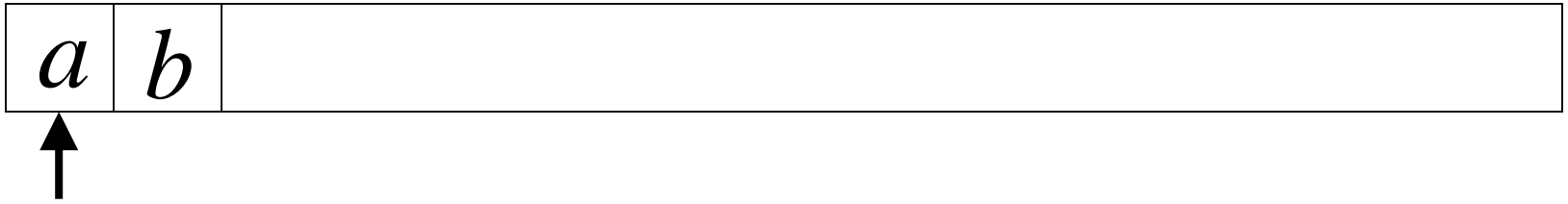
Language accepted: $L = \{aa\}$

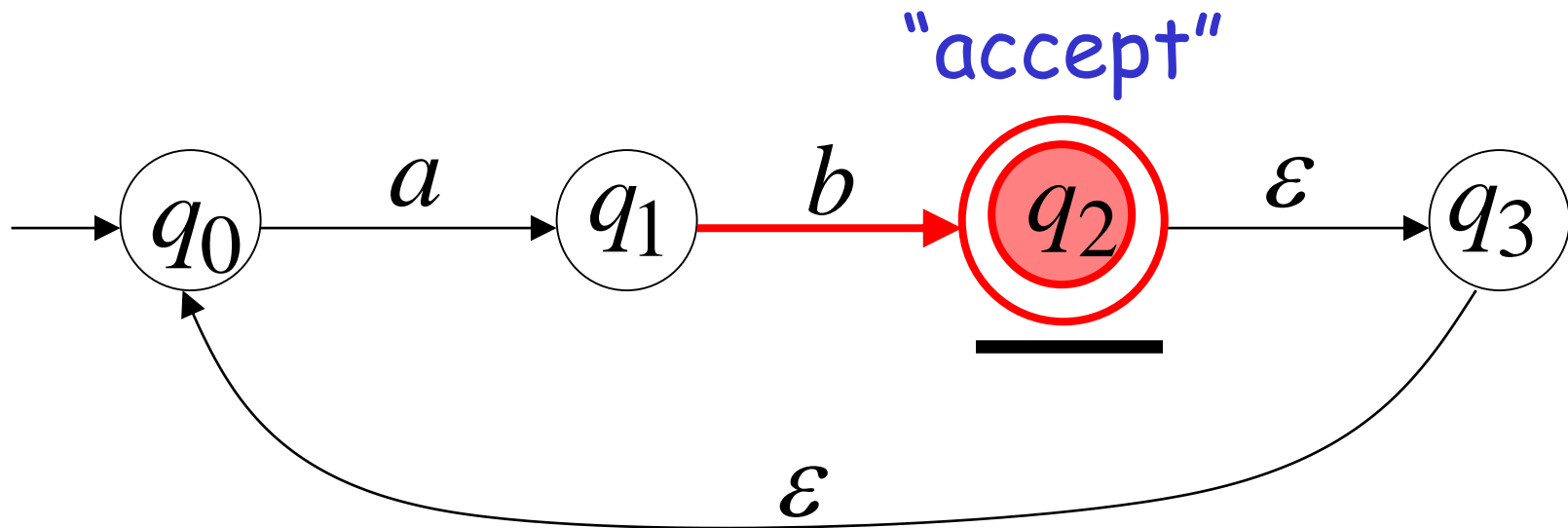


Another NFA Example

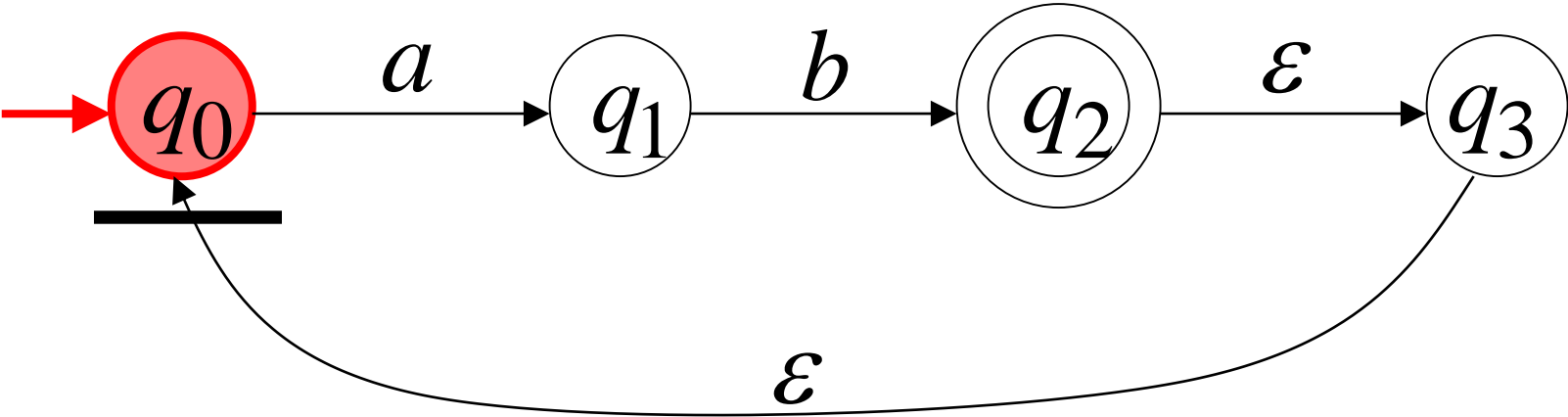


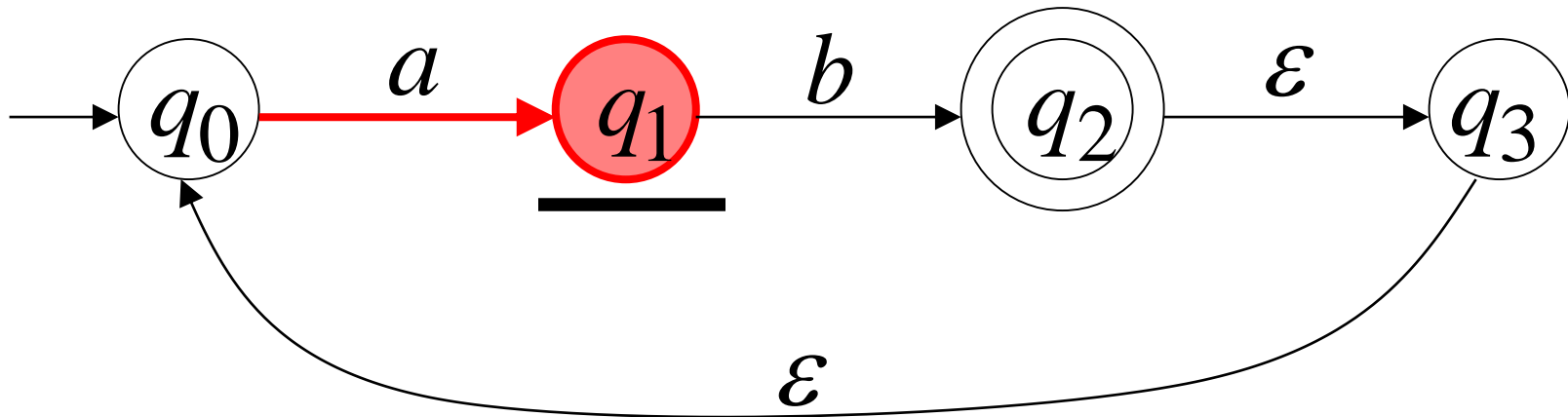


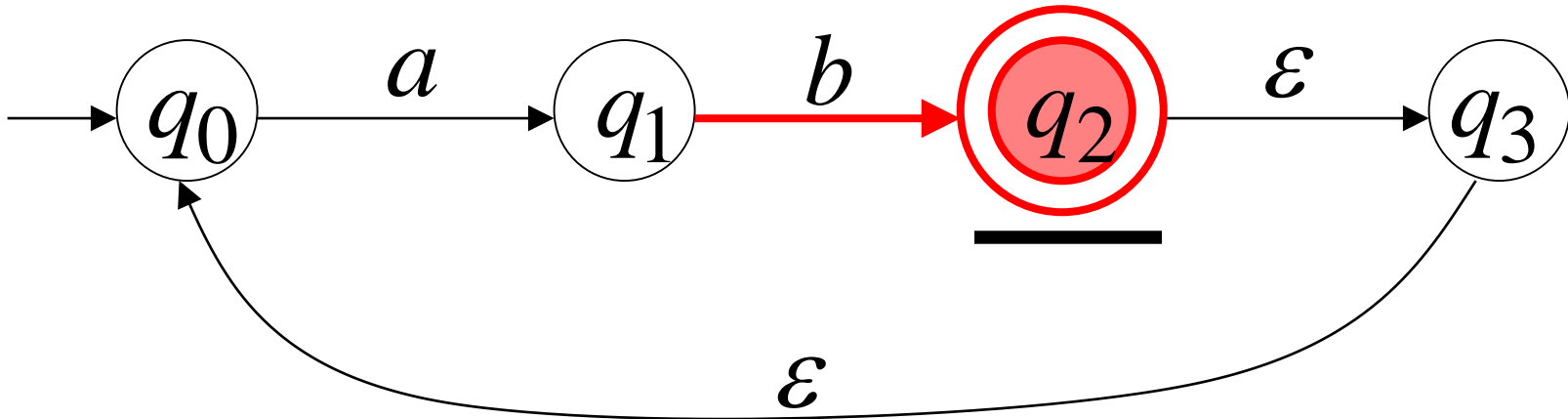


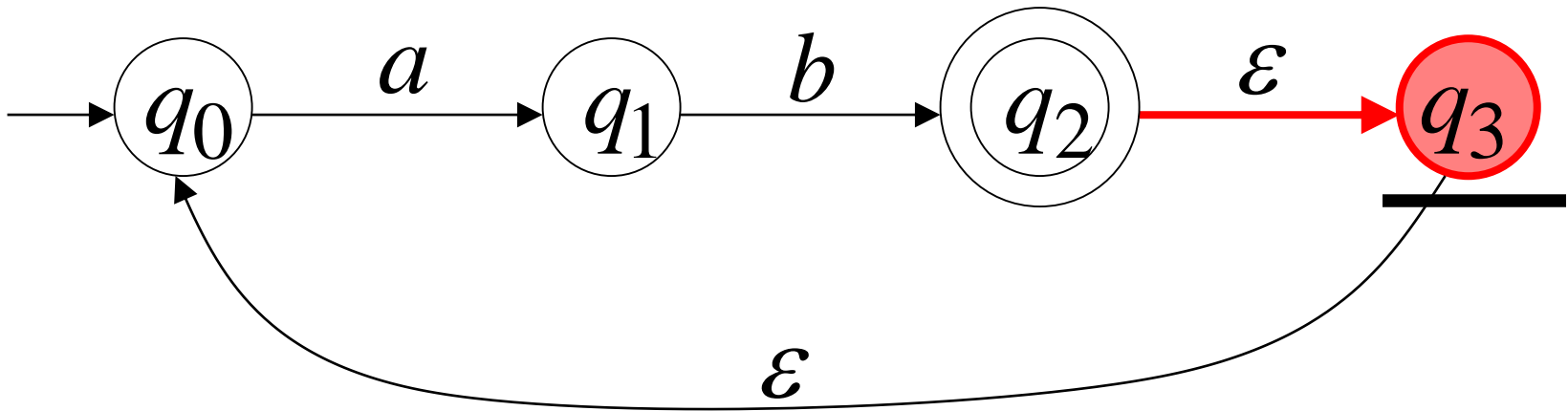
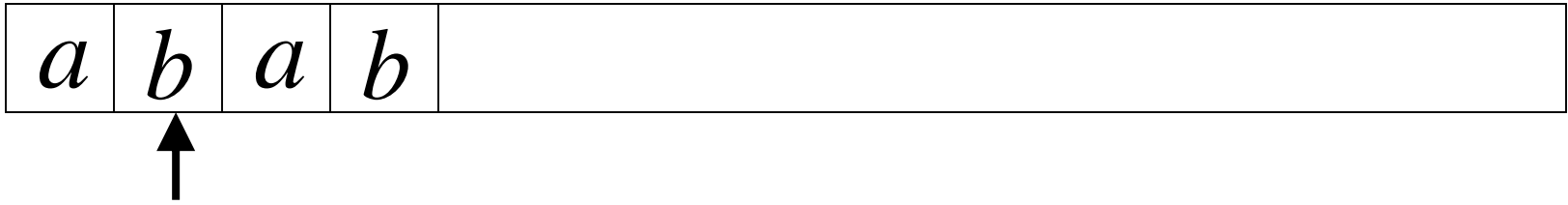


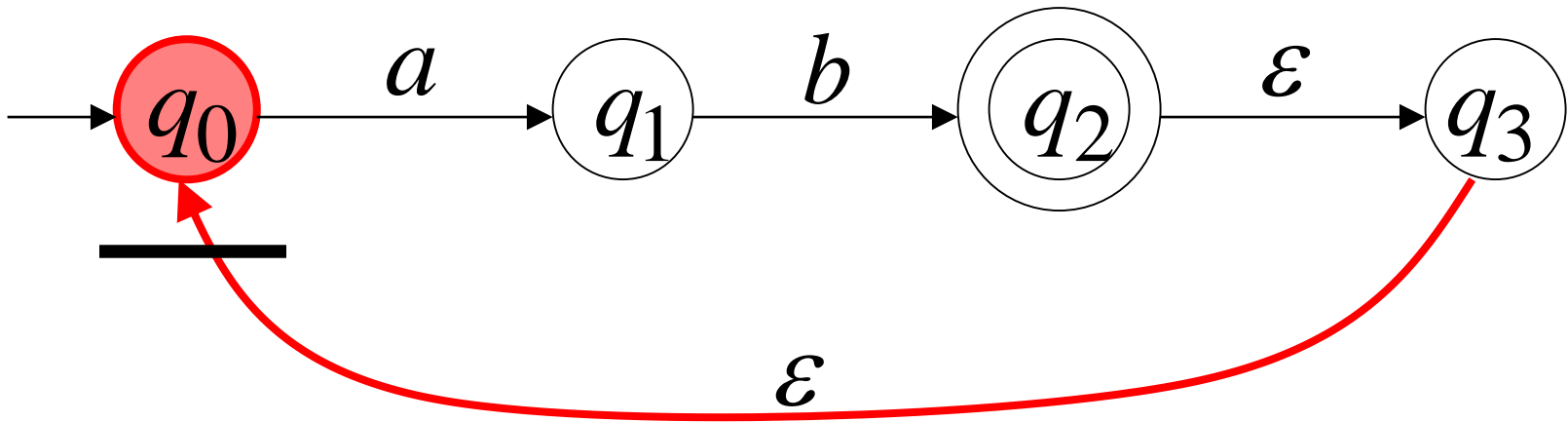
Another String

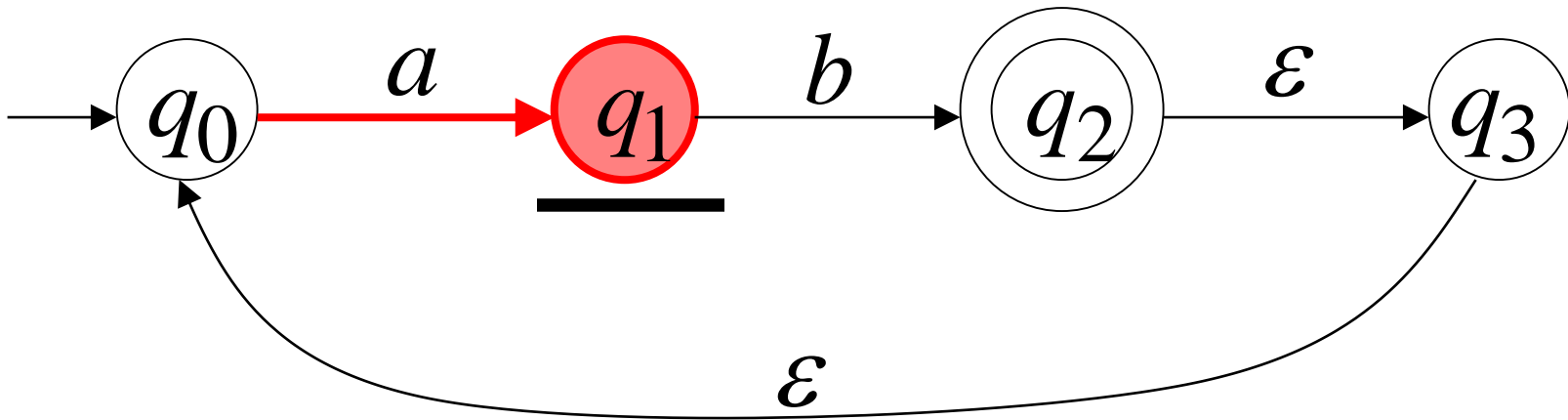
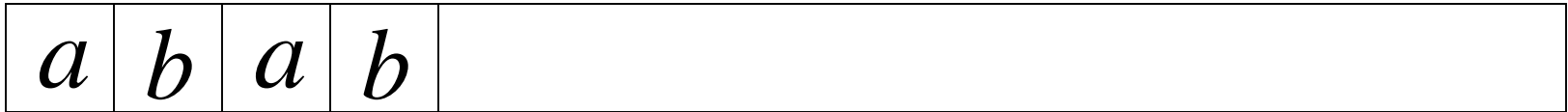


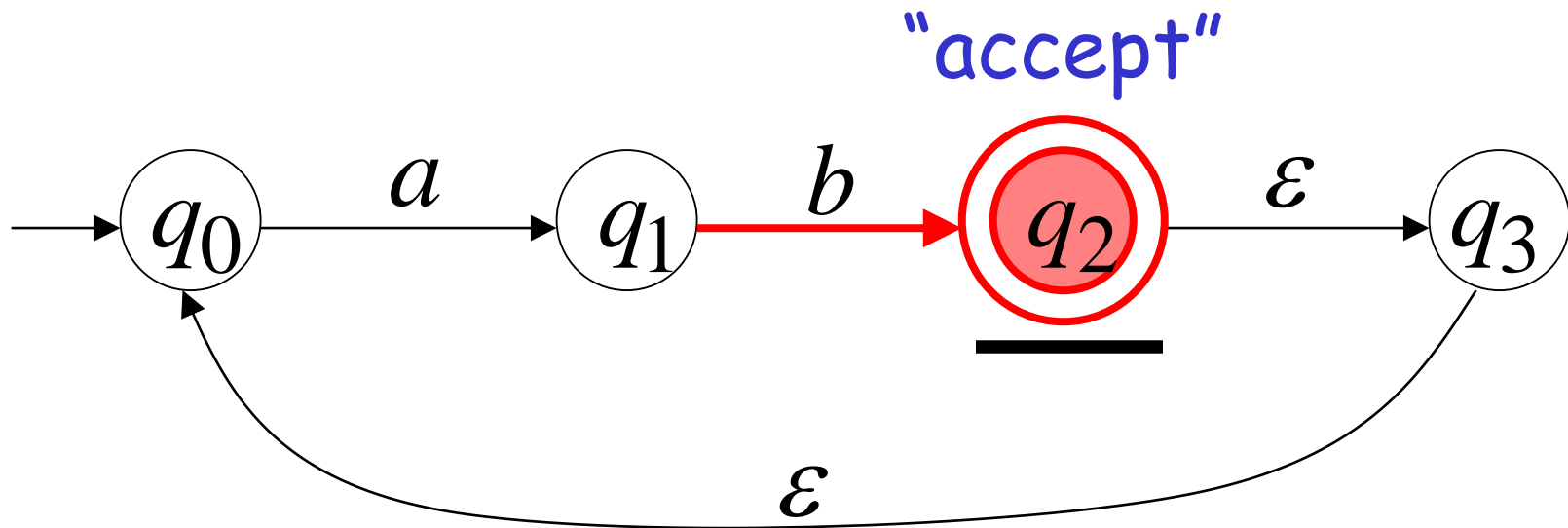






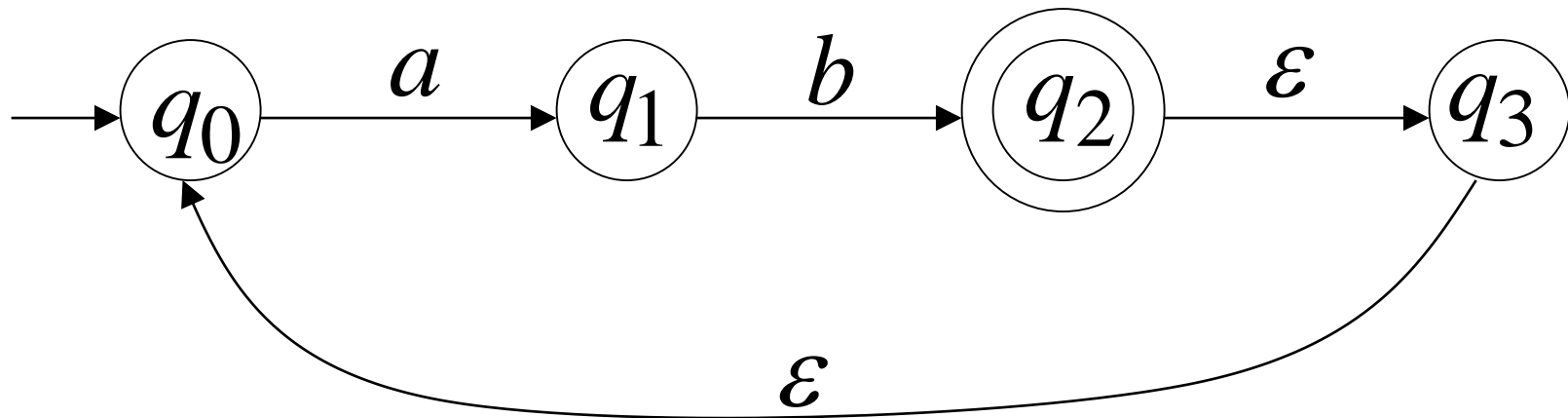




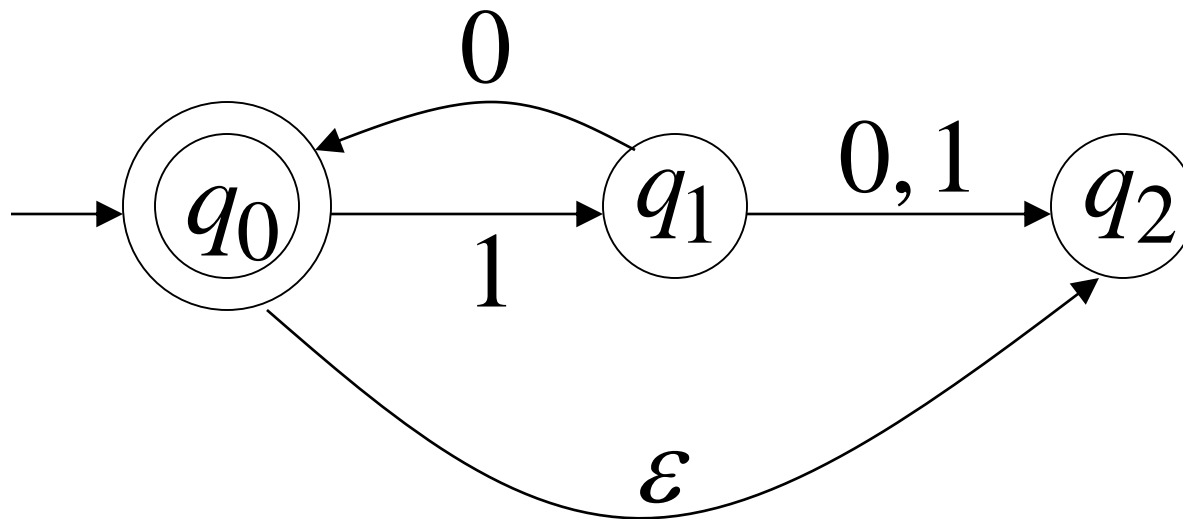


Language accepted

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

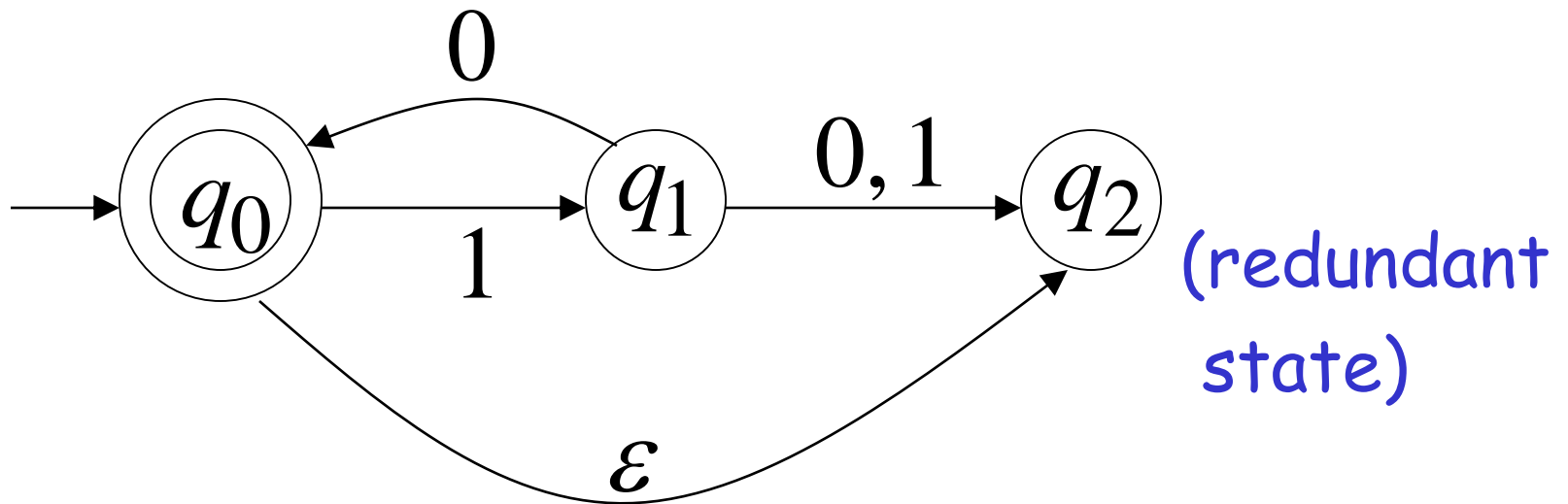


Another NFA Example

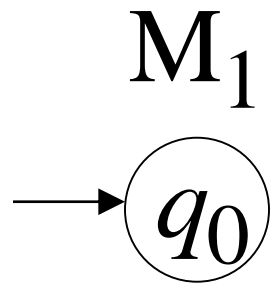


Language accepted

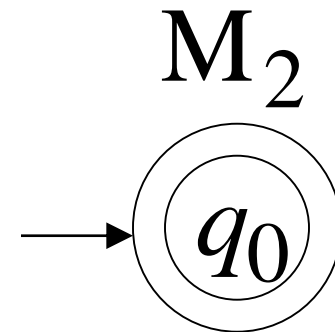
$$L = \{\varepsilon, 10, 1010, 101010, \dots\}$$
$$= \{10\}^*$$



Simple Automata



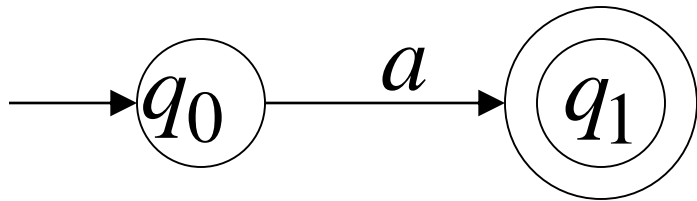
$$L(M_1) = \{\}$$



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

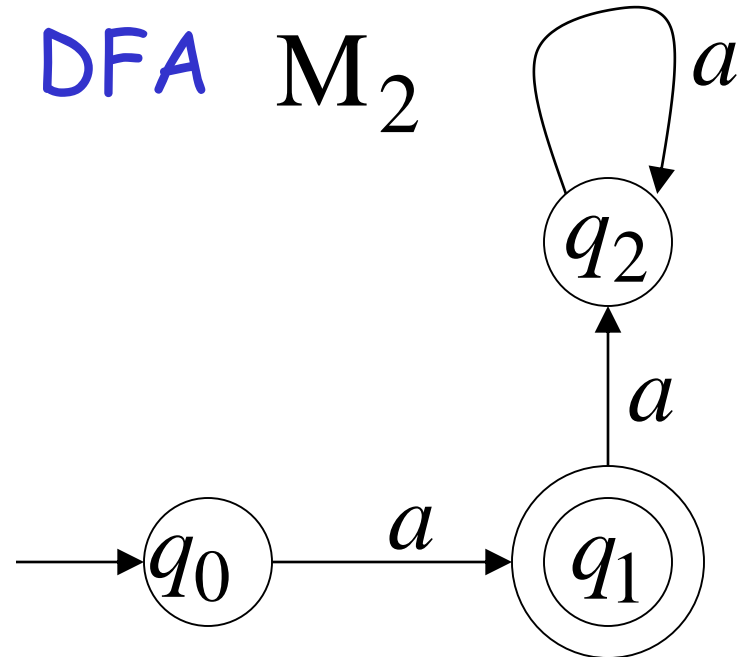
NFAs are interesting because we can express languages easier than DFAs

NFA M_1



$$L(M_1) = \{a\}$$

DFA M_2



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

Formal Definition of NFAs

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

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

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

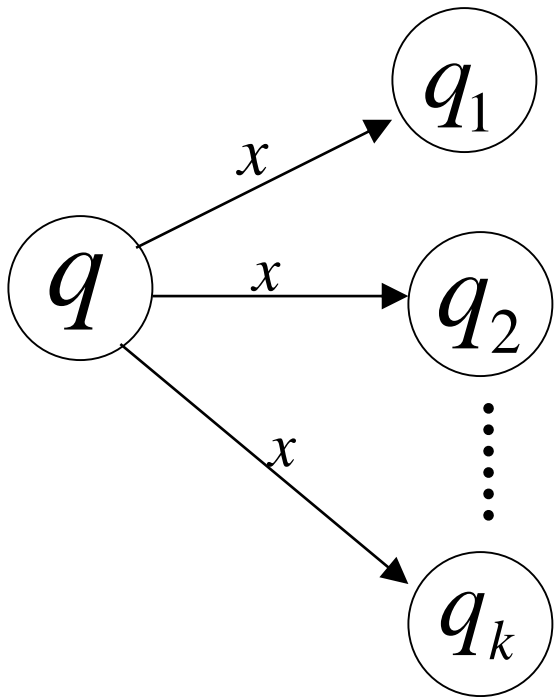
δ : Transition function

q_0 : Initial state

F : Accepting states

Transition Function δ

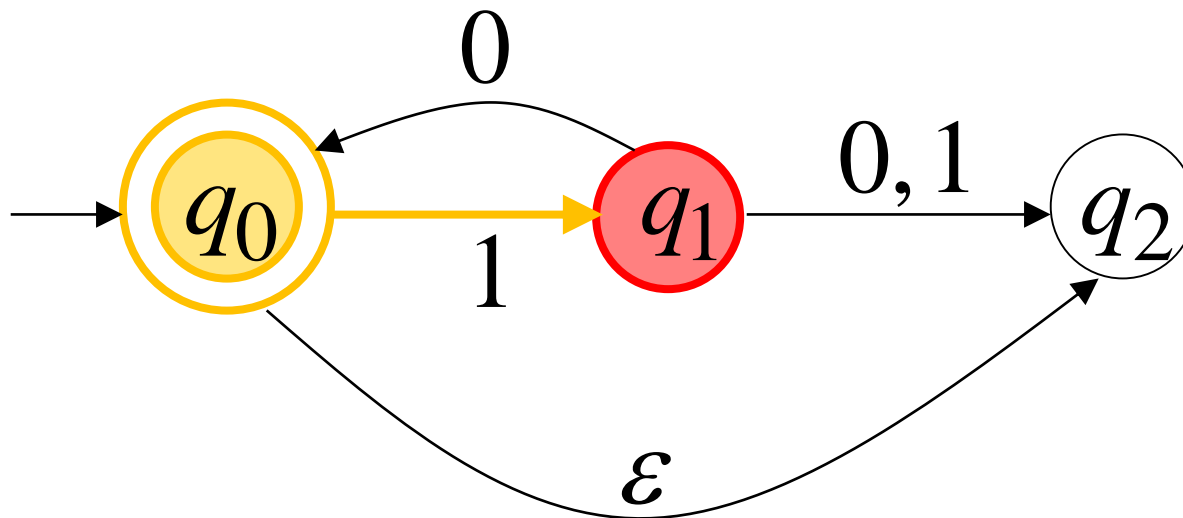
$$\delta(q, x) = \{q_1, q_2, \dots, q_k\}$$



resulting states reached
by following **one** transition
with input symbol x

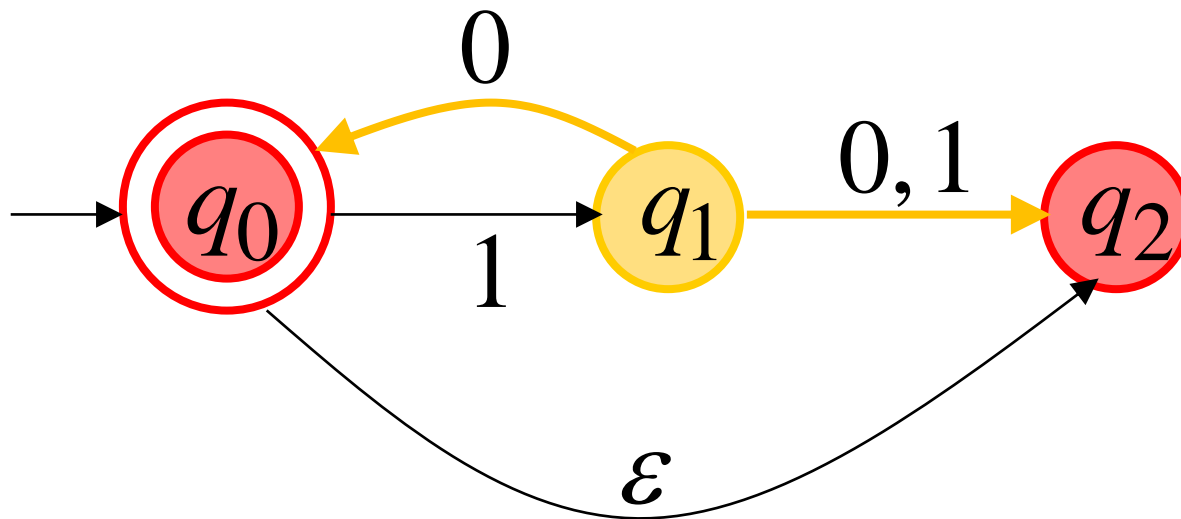
States reachable from q_0 scanning 1

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



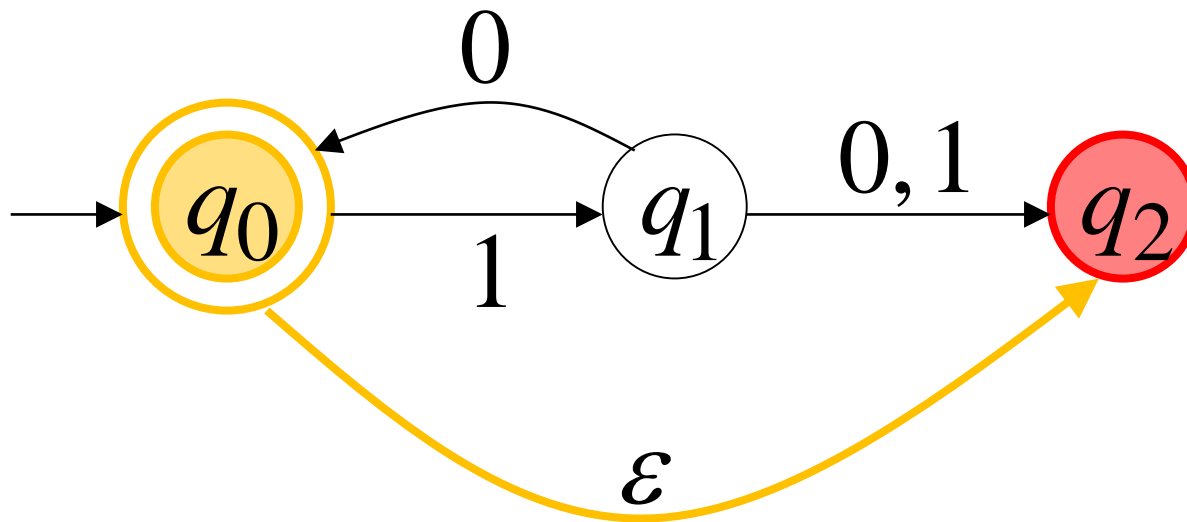
States reachable from q_1 scanning 0

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



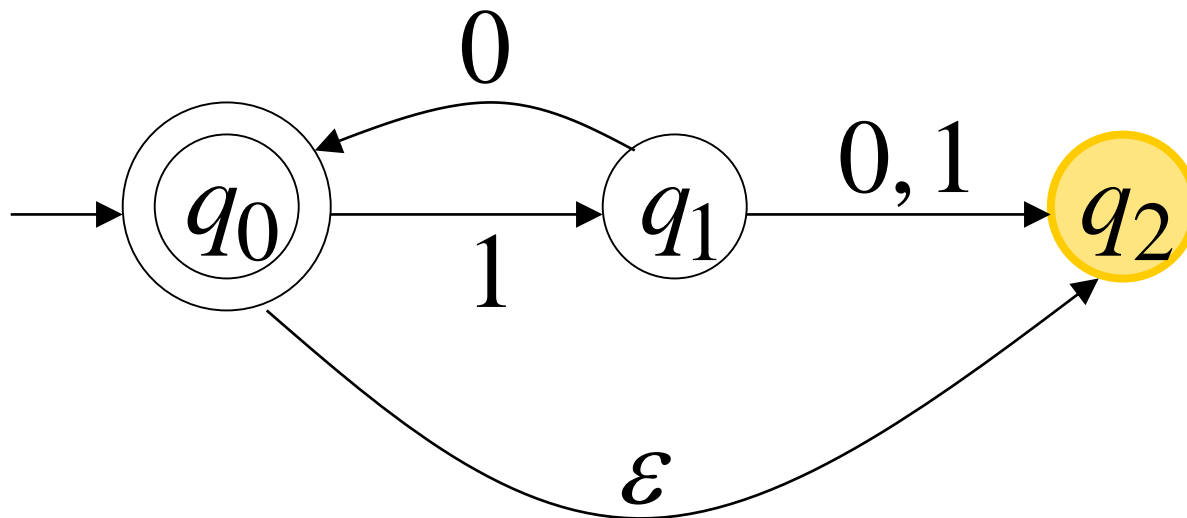
States reachable from q_0 with one transition scanning no input symbol

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



States reachable from q_2 scanning 1

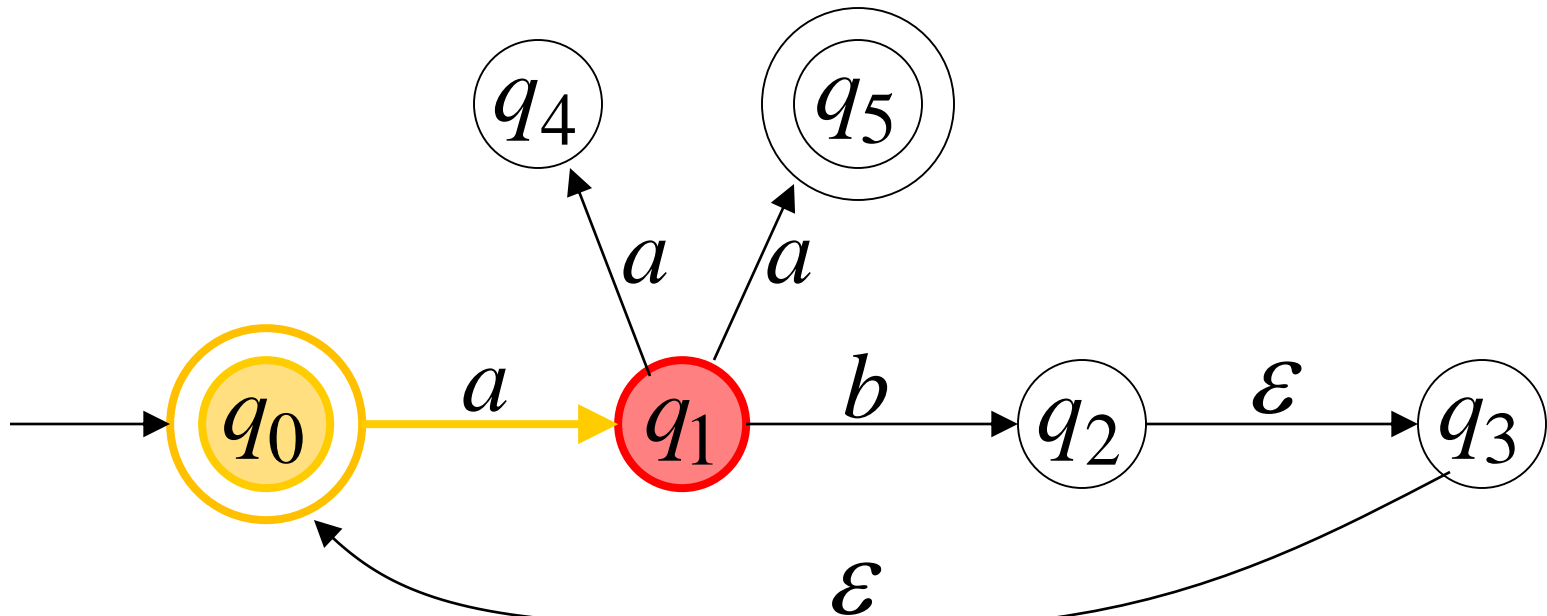
$$\delta(q_2, 1) = \emptyset$$



Extended Transition Function δ^*

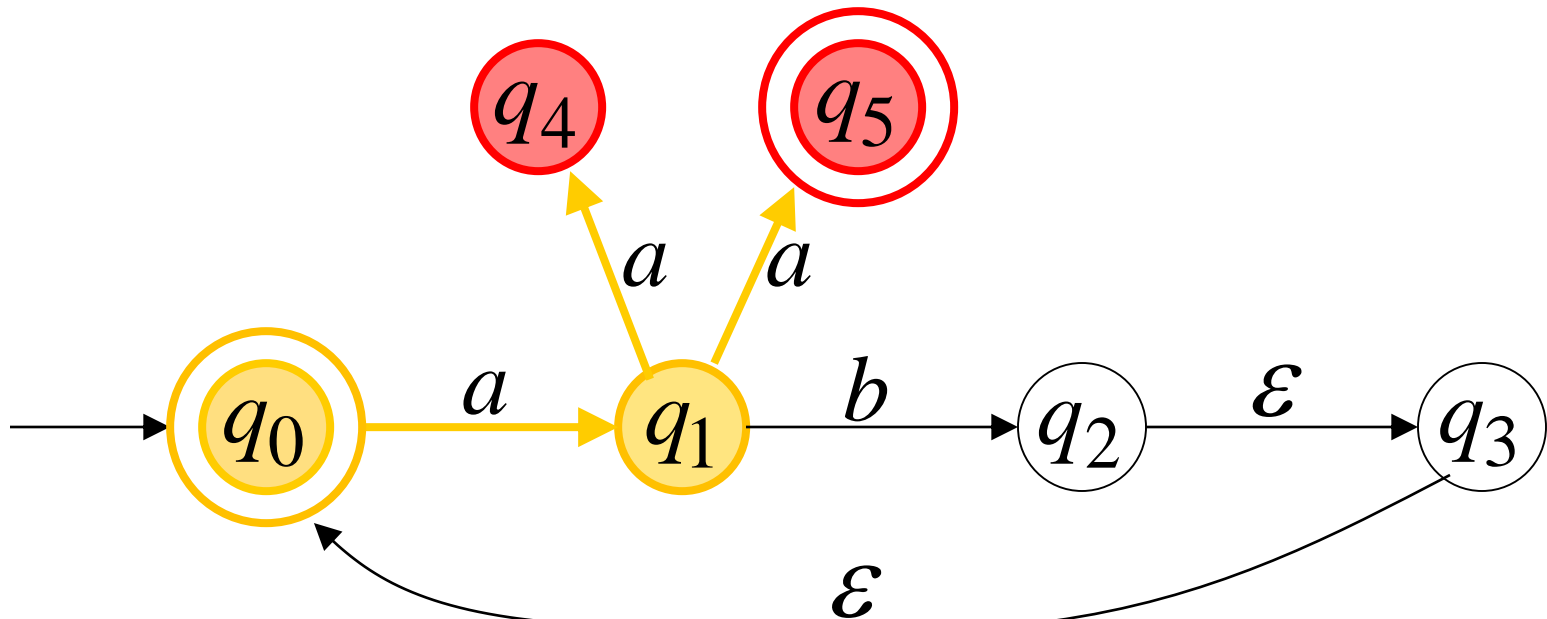
Similar with δ but applied on strings

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



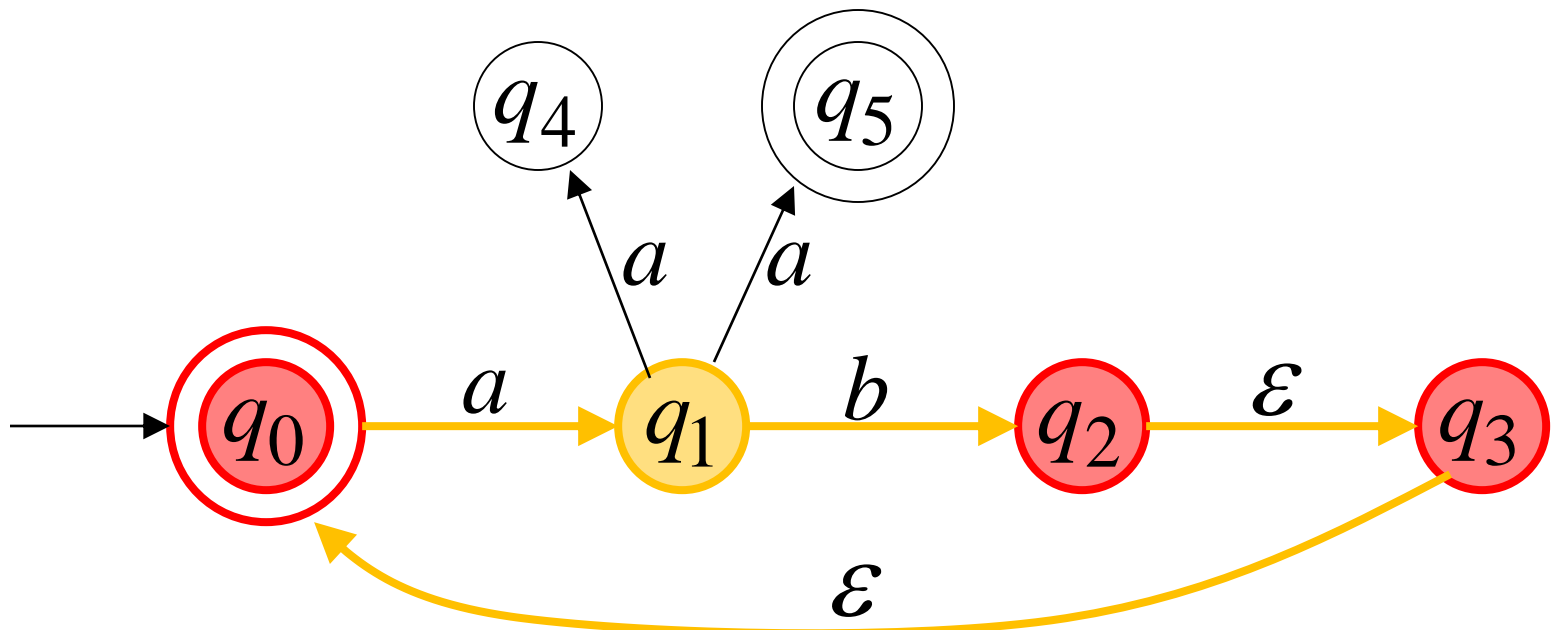
States reachable from q_0 scanning aa

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



States reachable from q_0 scanning ab

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



Special case:

for any state q

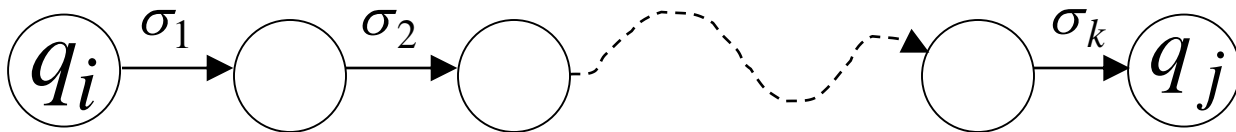
$$q \in \delta^*(q, \varepsilon)$$

In general

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



$$w = \sigma_1 \sigma_2 \Lambda \sigma_k$$



The Language of an NFA M

The language accepted by M is:

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

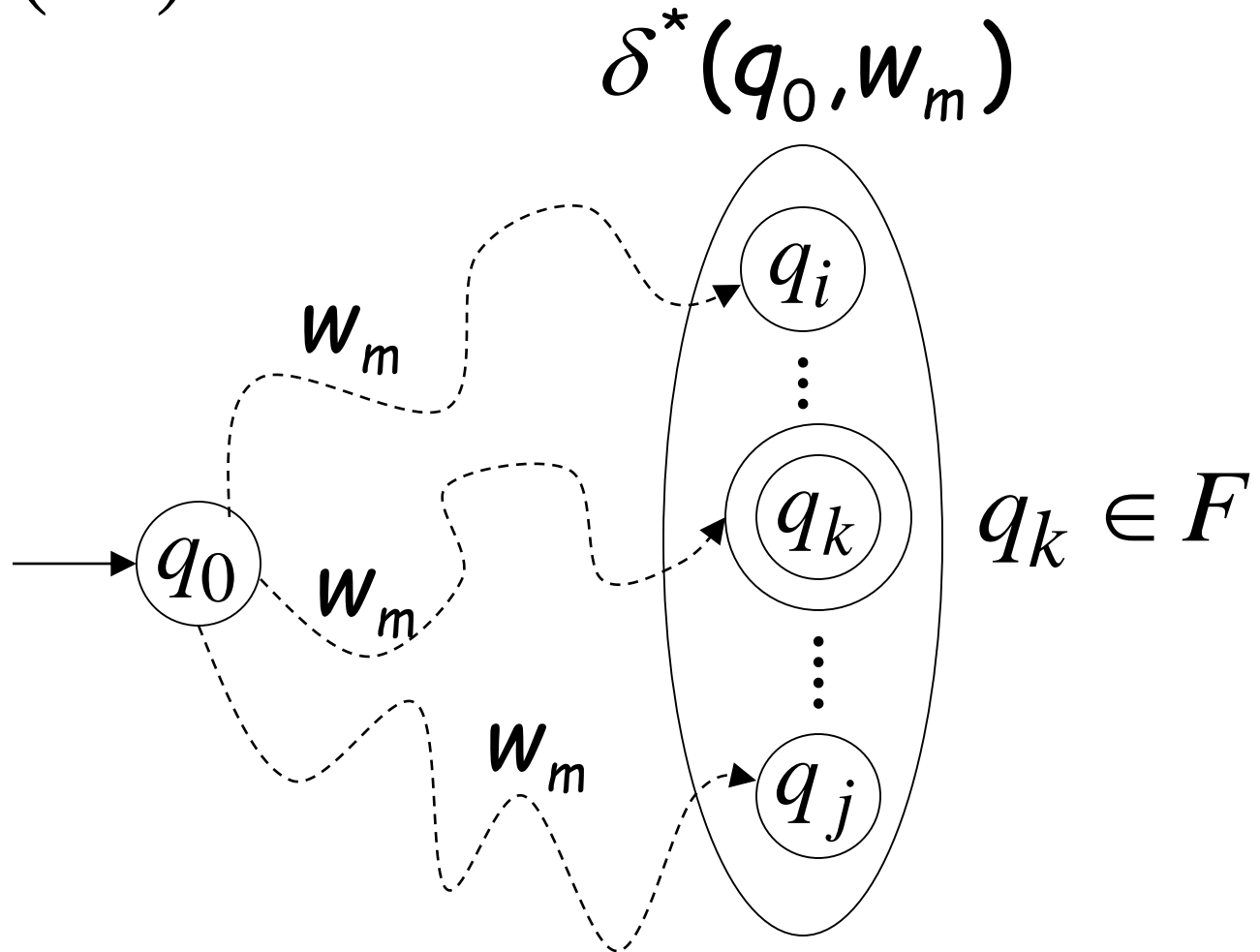
Where for each w_m

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

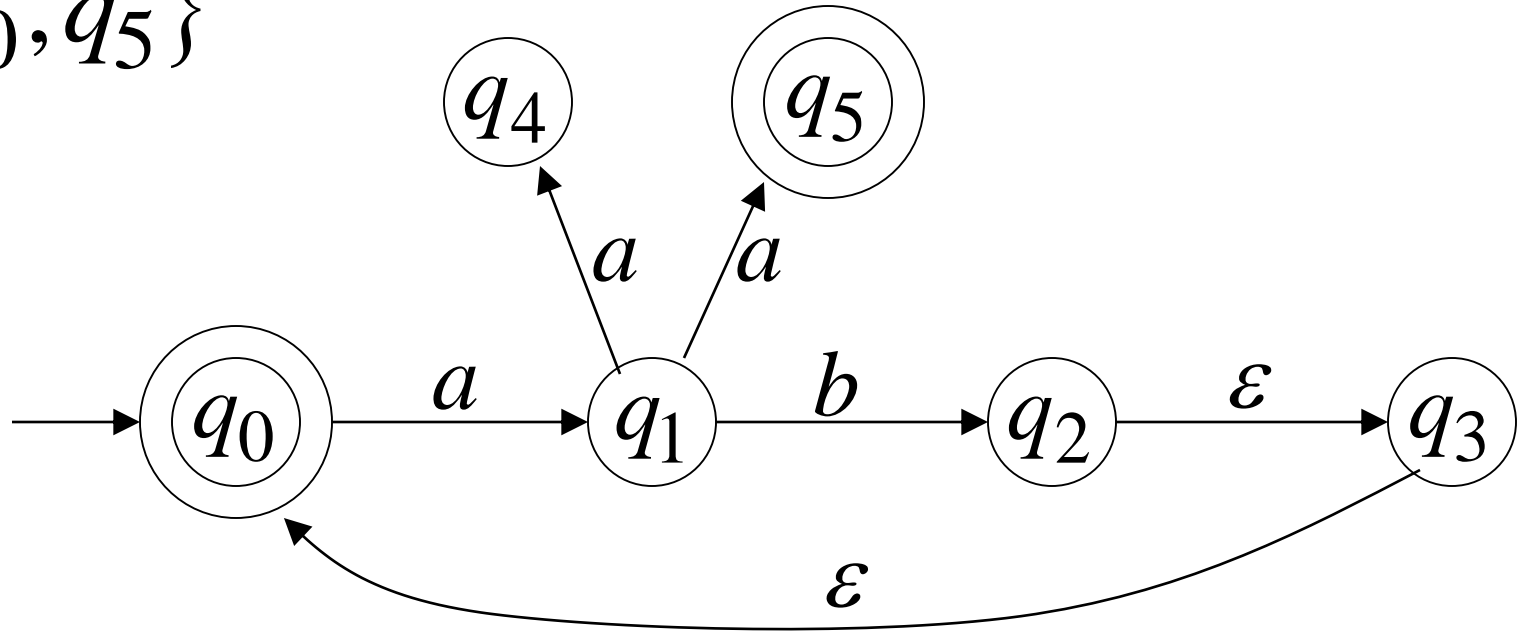
and there is some $q_k \in F$ (accepting state)



$$w_m \in L(M)$$



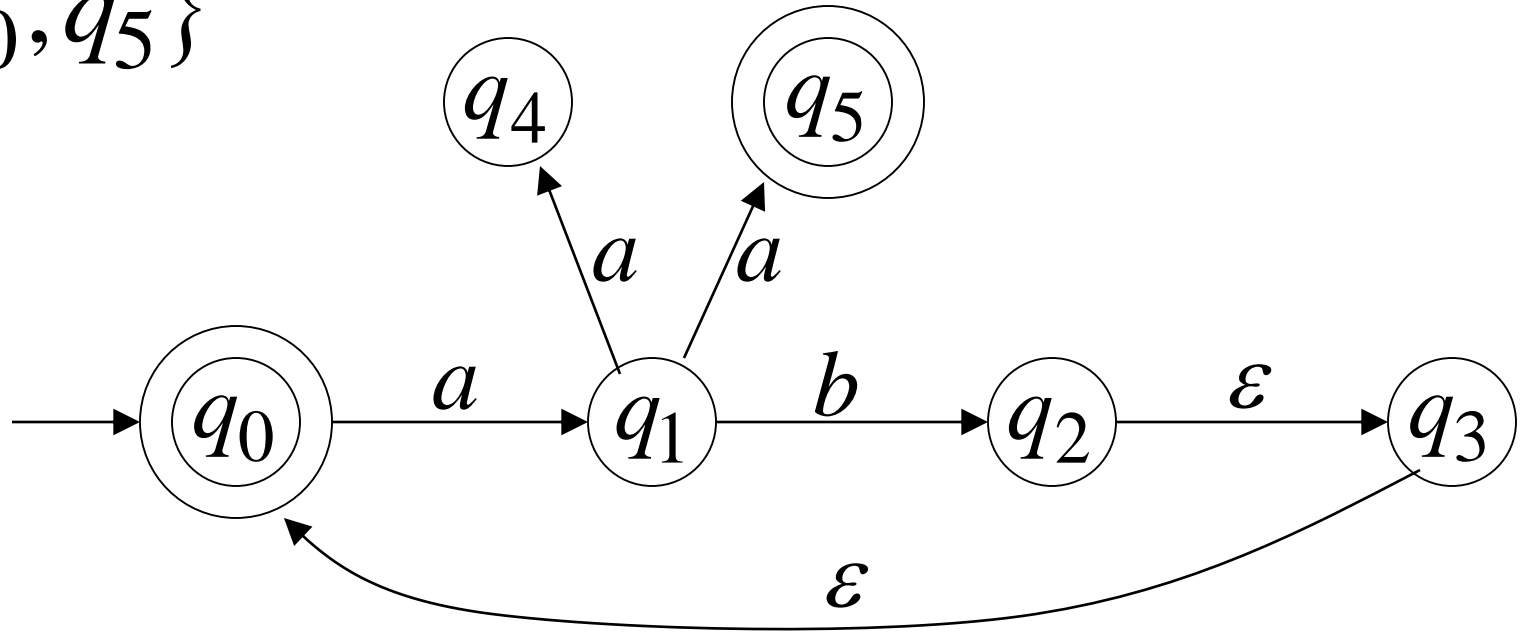
$$F = \{q_0, q_5\}$$



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

$\swarrow \in F$

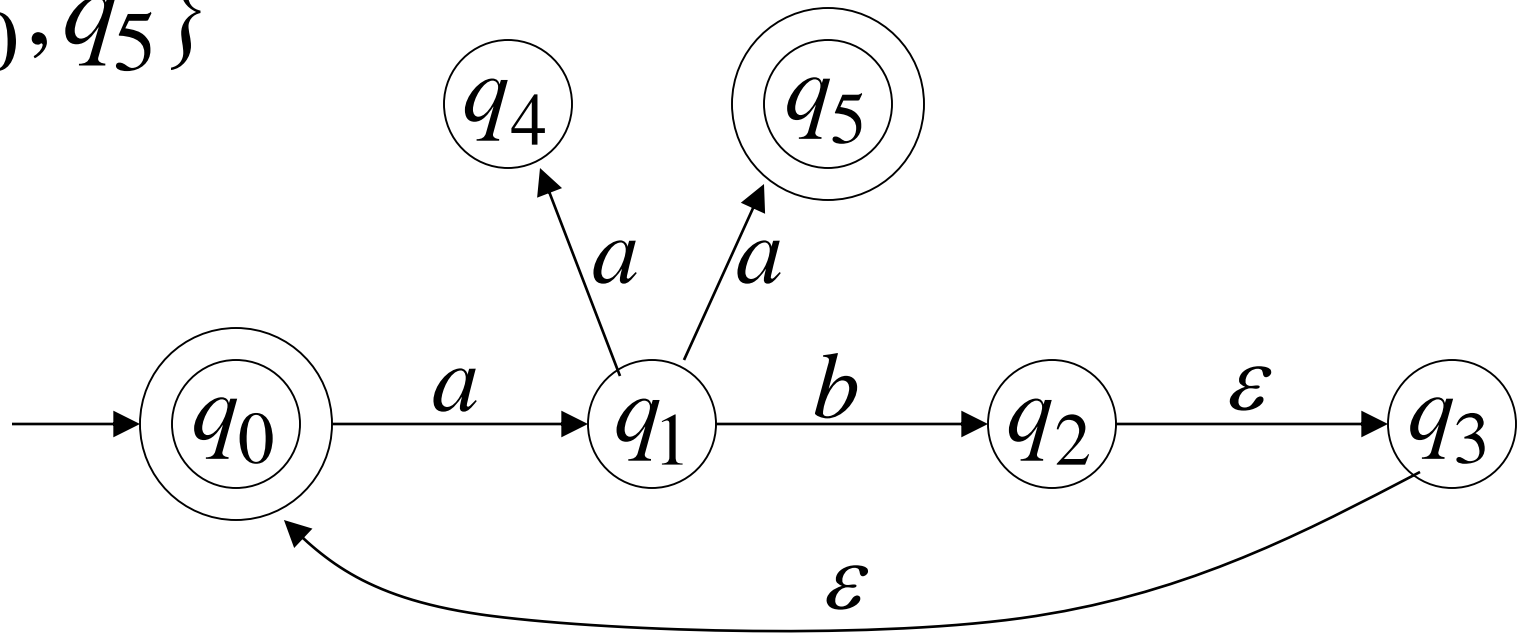
$$F = \{q_0, q_5\}$$



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

\swarrow
 $\in F$

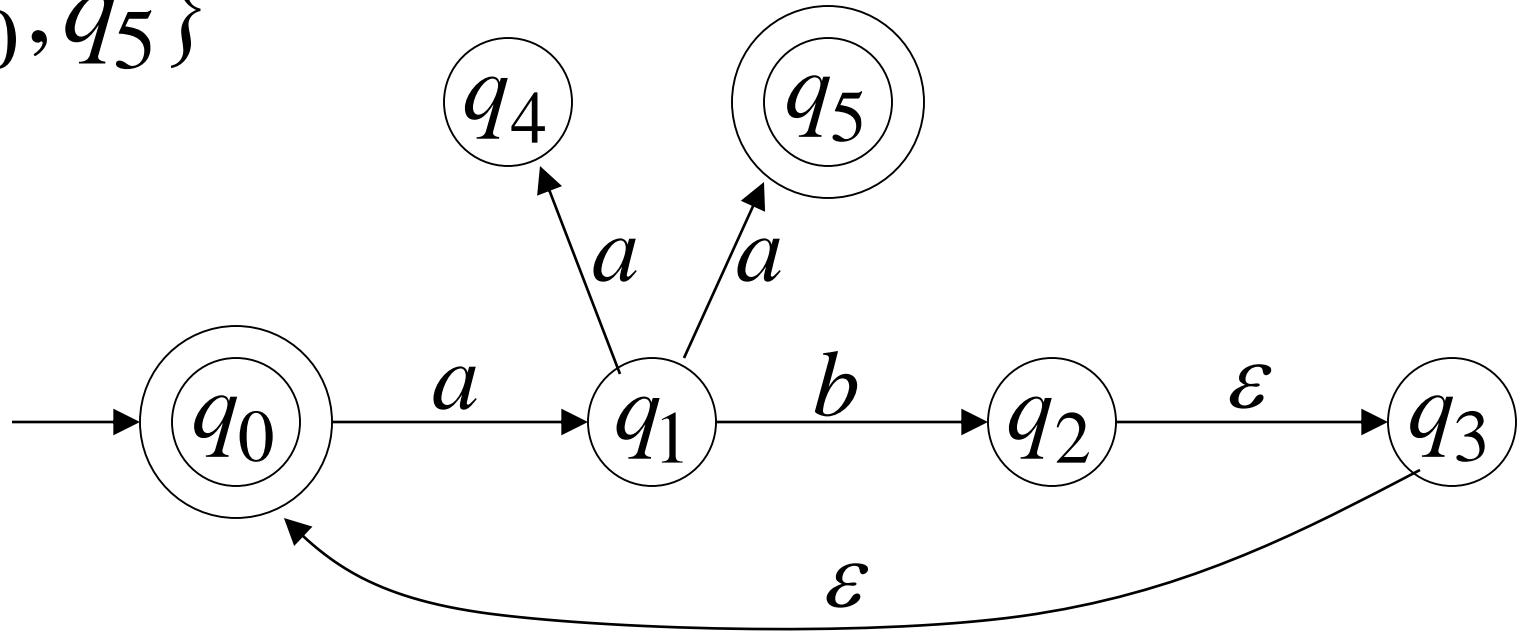
$$F = \{q_0, q_5\}$$



$$\delta^*(q_0, abaa) = \{q_4, \underline{q_5}\} \xrightarrow{\text{yellow arrow}} abaa \in L(M)$$

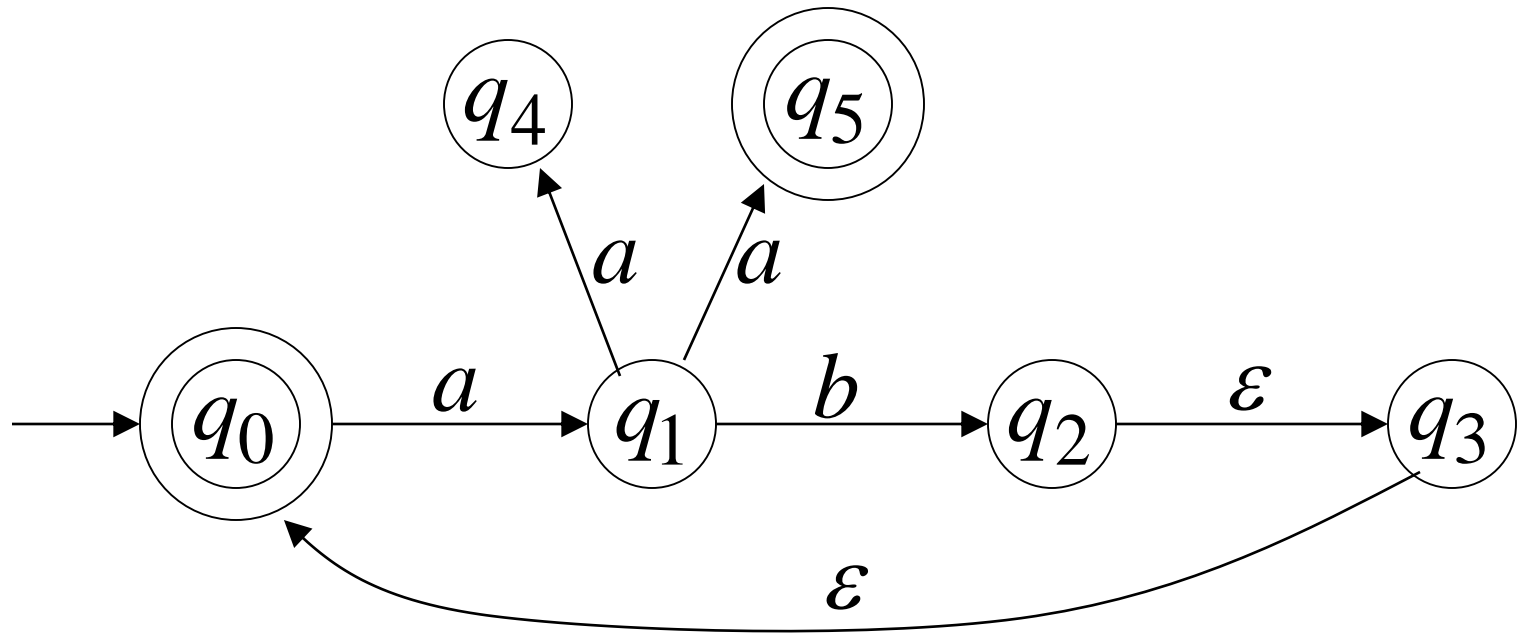
\swarrow
 $\in F$

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



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

$\searrow \notin F$



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

NFAs accept the Regular Languages

Equivalence of Machines

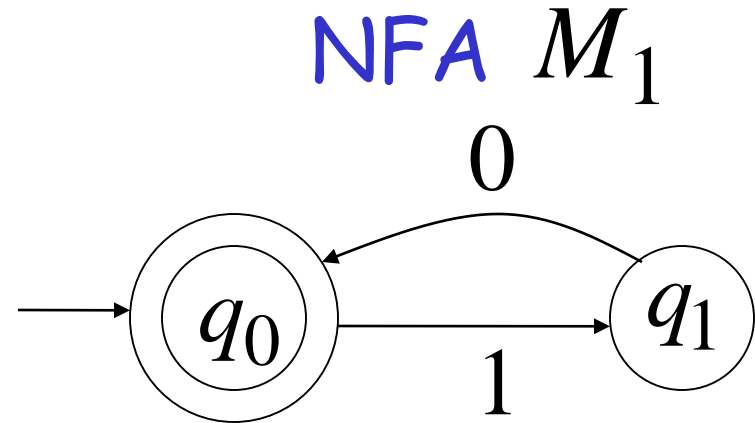
Definition:

Machine M_1 is equivalent to machine M_2

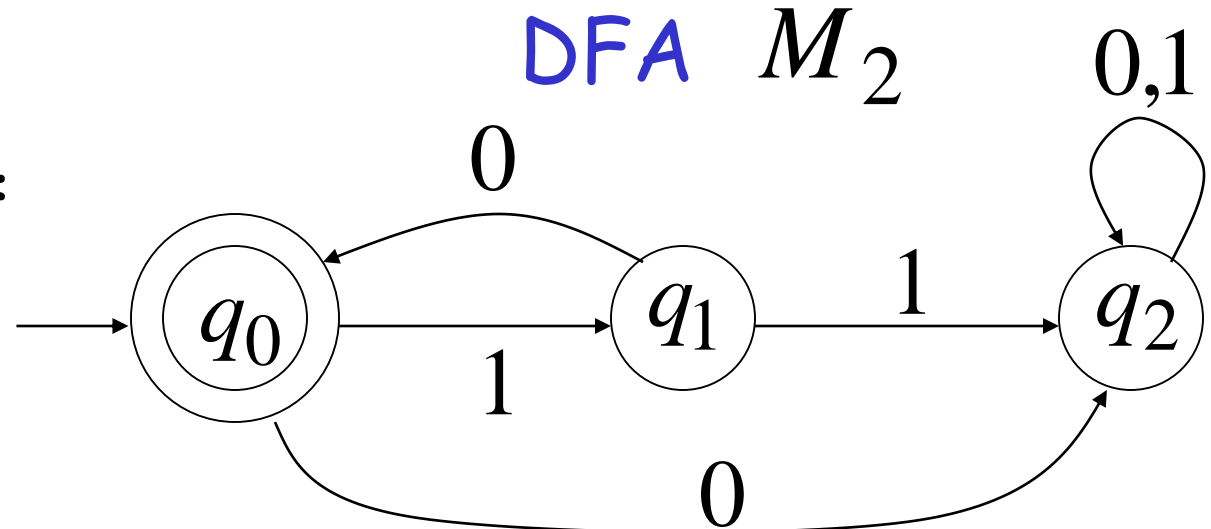
if $L(M_1) = L(M_2)$

Example of equivalent machines

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



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

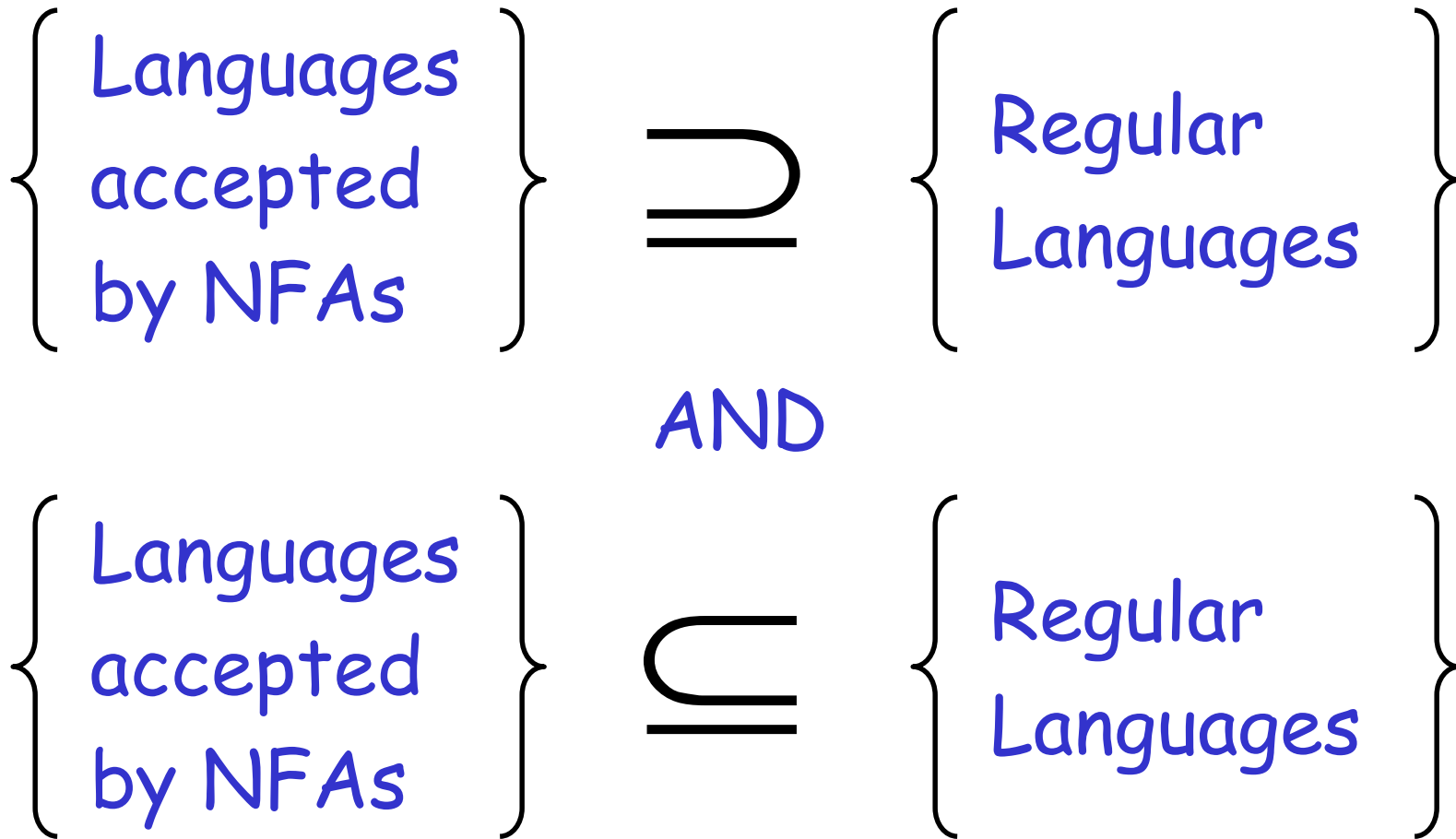


Theorem:

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

NFAs and DFAs have the same computation power, namely, they accept the same set of languages

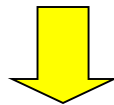
Proof: we need to show



Proof-Step 1

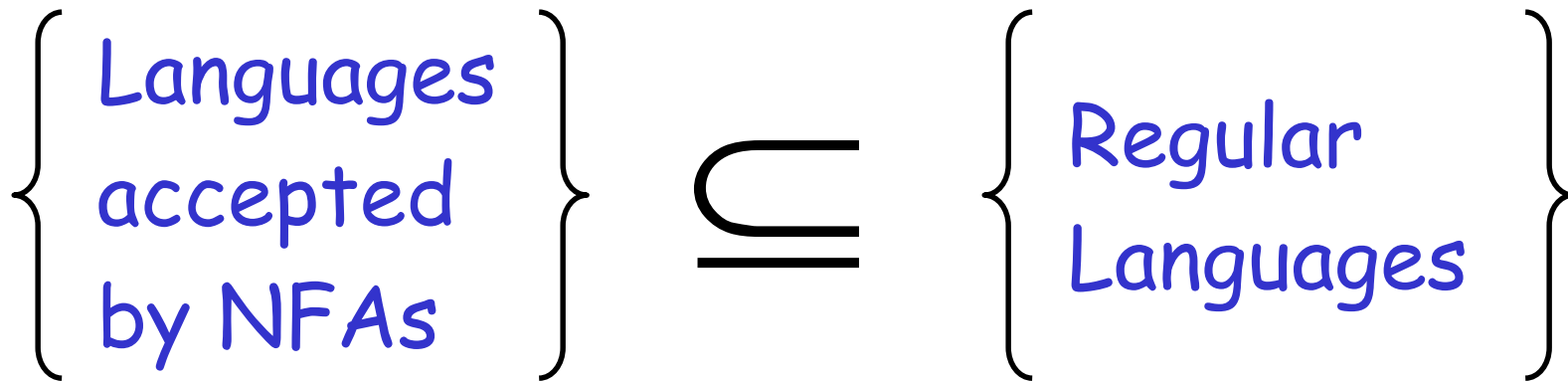
$$\left\{ \begin{array}{l} \text{Languages} \\ \text{accepted} \\ \text{by NFAs} \end{array} \right\} \supseteq \left\{ \begin{array}{l} \text{Regular} \\ \text{Languages} \end{array} \right\}$$

Every DFA is trivially a NFA

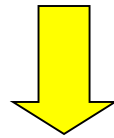


Any language L accepted by a DFA
is also accepted by a NFA

Proof-Step 2



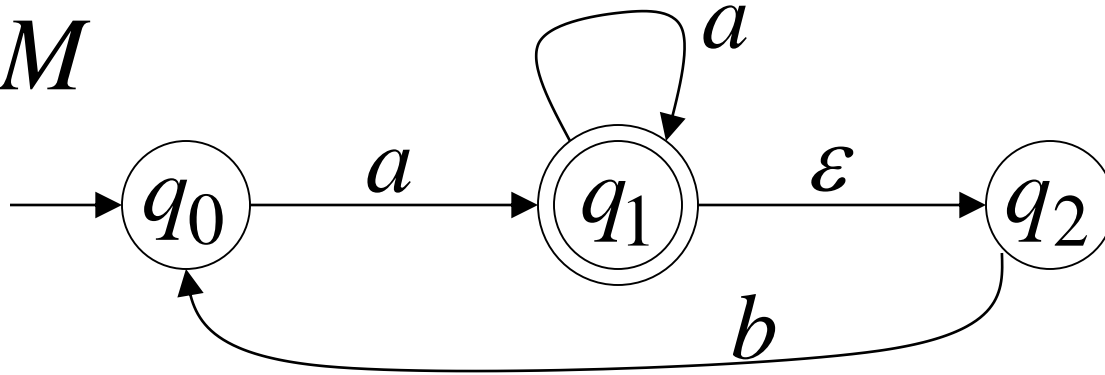
Any NFA can be converted to an equivalent DFA



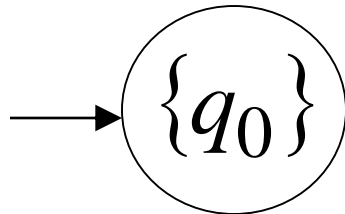
Any language L accepted by a NFA is also accepted by a DFA

Conversion of NFA to DFA

NFA M

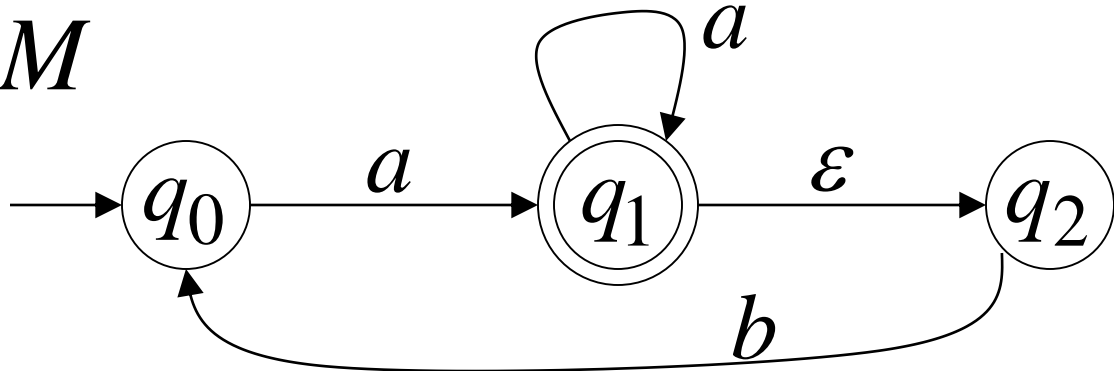


DFA M'

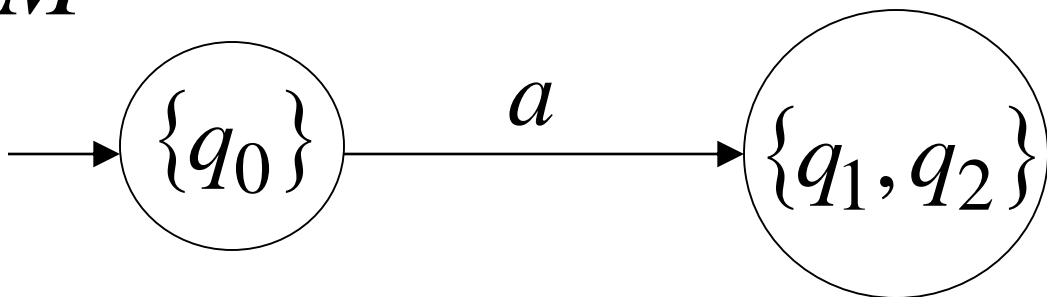


$$\delta^*(q_0, a) = \{q_1, q_2\}$$

NFA M

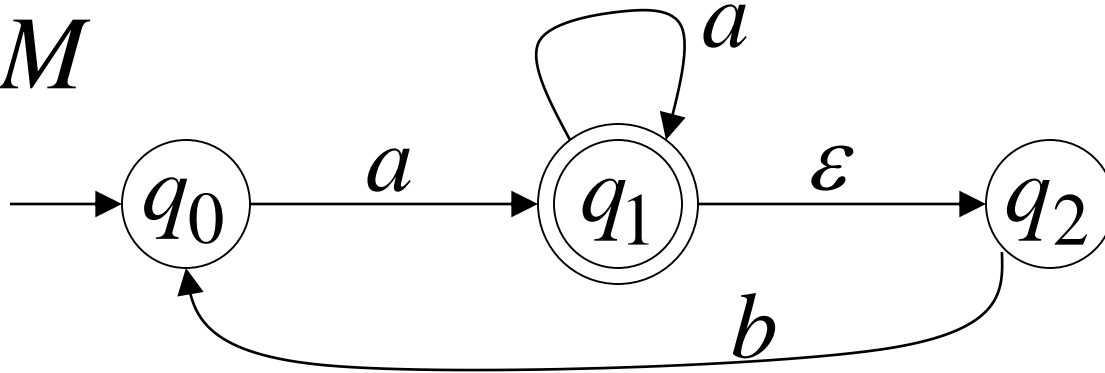


DFA M'

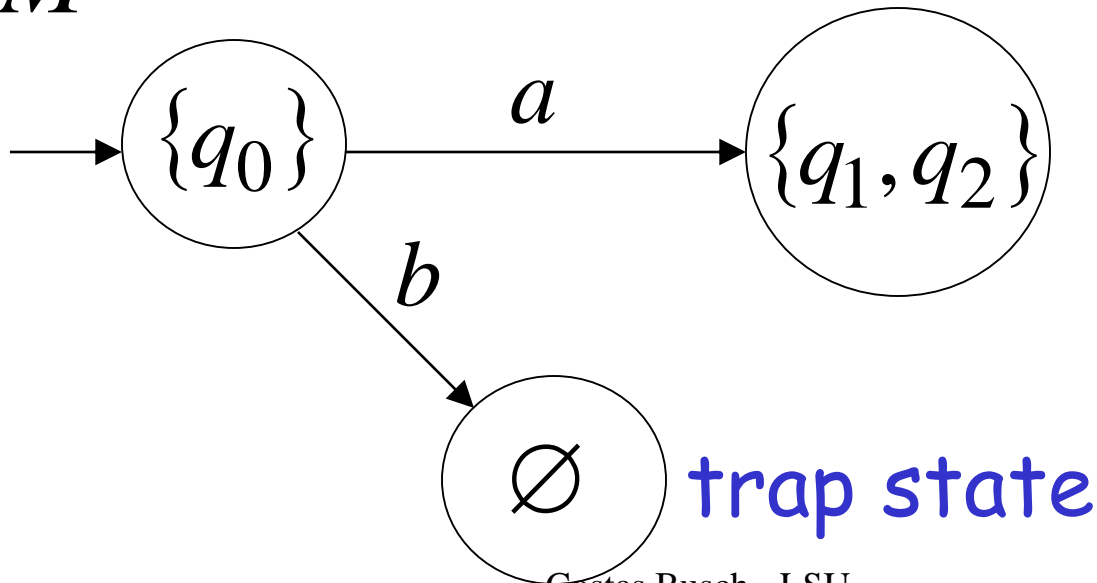


$$\delta^*(q_0, b) = \emptyset \quad \text{empty set}$$

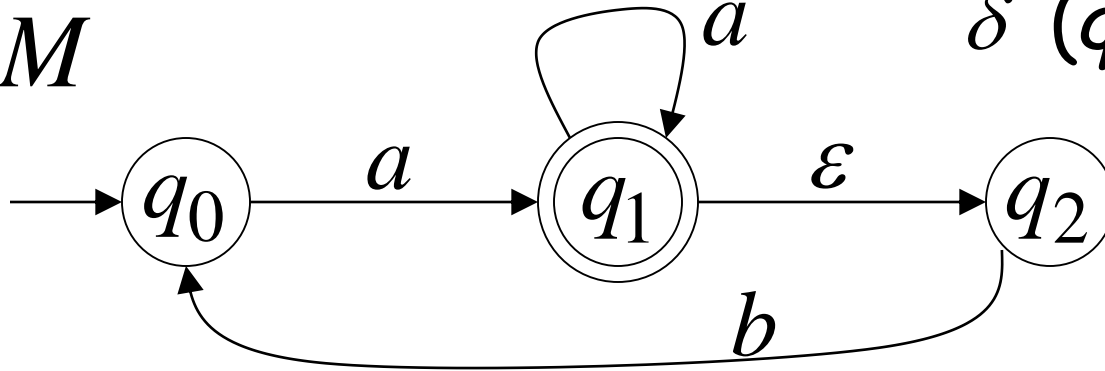
NFA M



DFA M'



NFA M



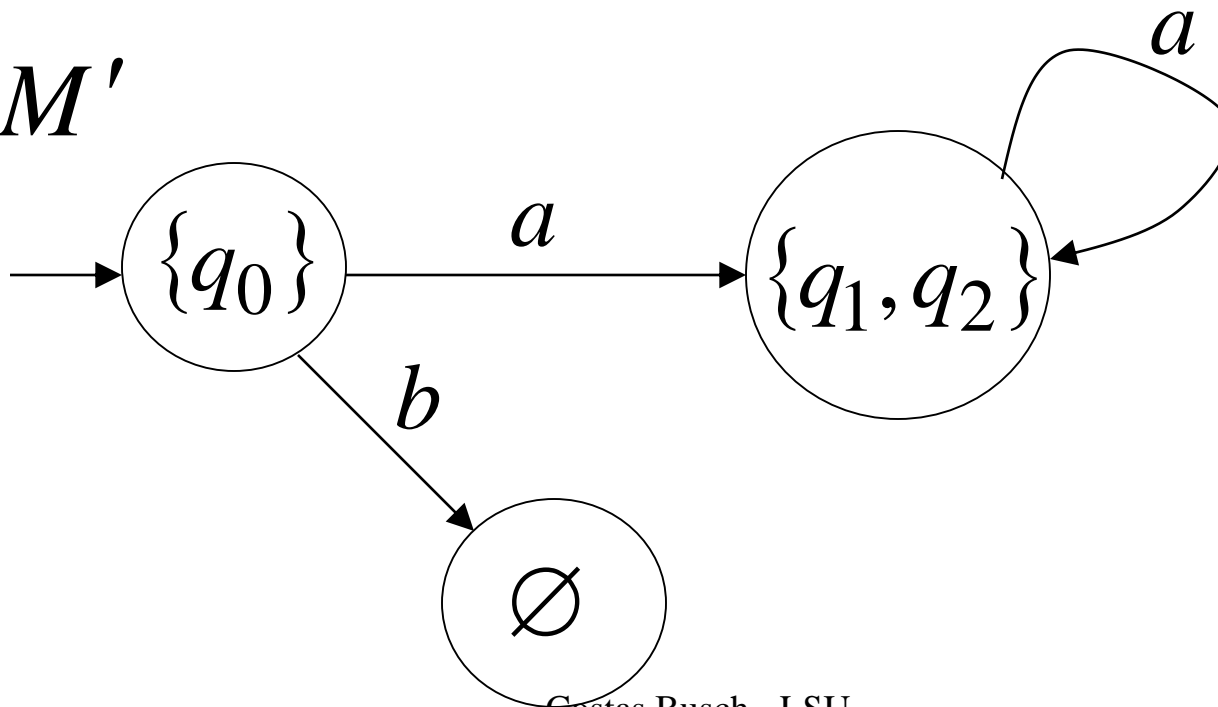
$$\delta^*(q_1, a) = \{q_1, q_2\}$$

$$\delta^*(q_2, a) = \emptyset$$

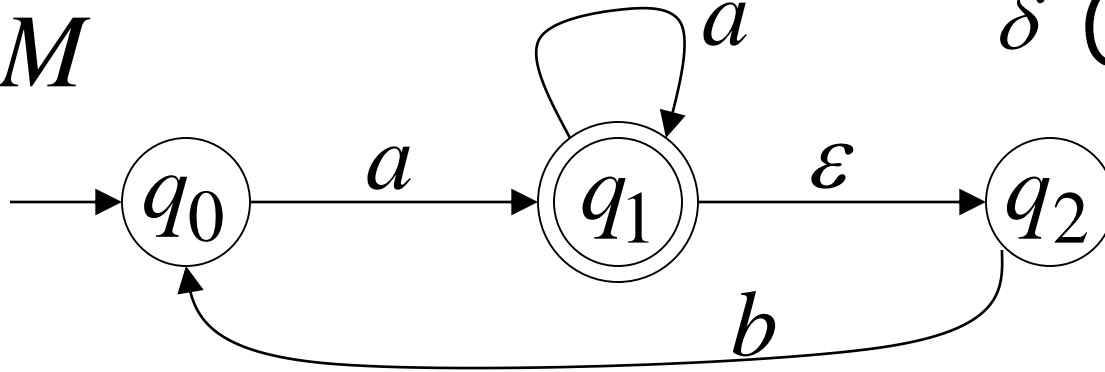
union

$$\{q_1, q_2\}$$

DFA M'



NFA M



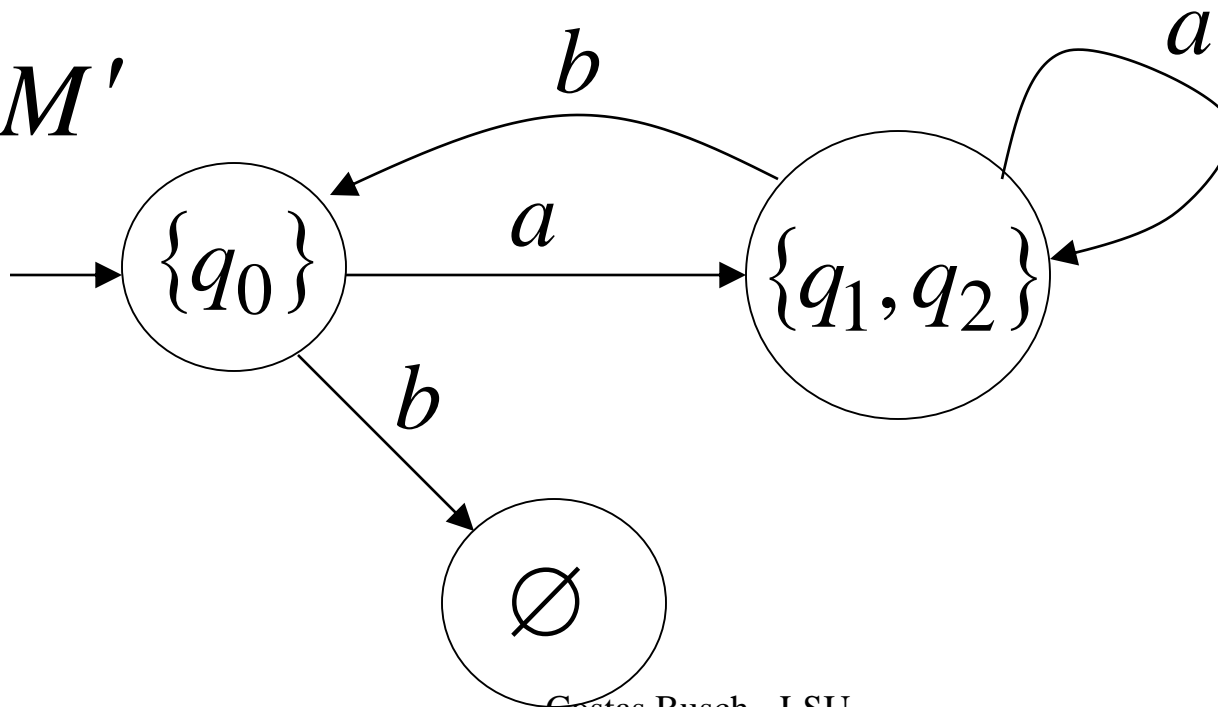
$$\delta^*(q_1, b) = \{q_0\}$$

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

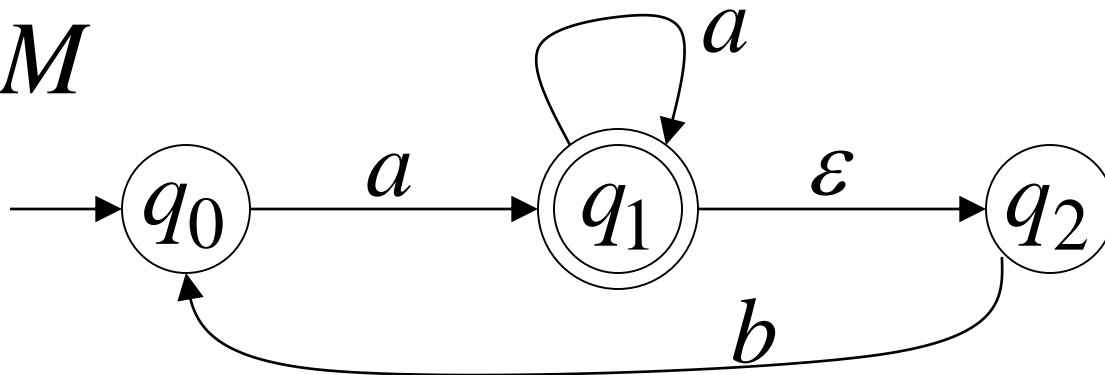
union

$$\{q_0\}$$

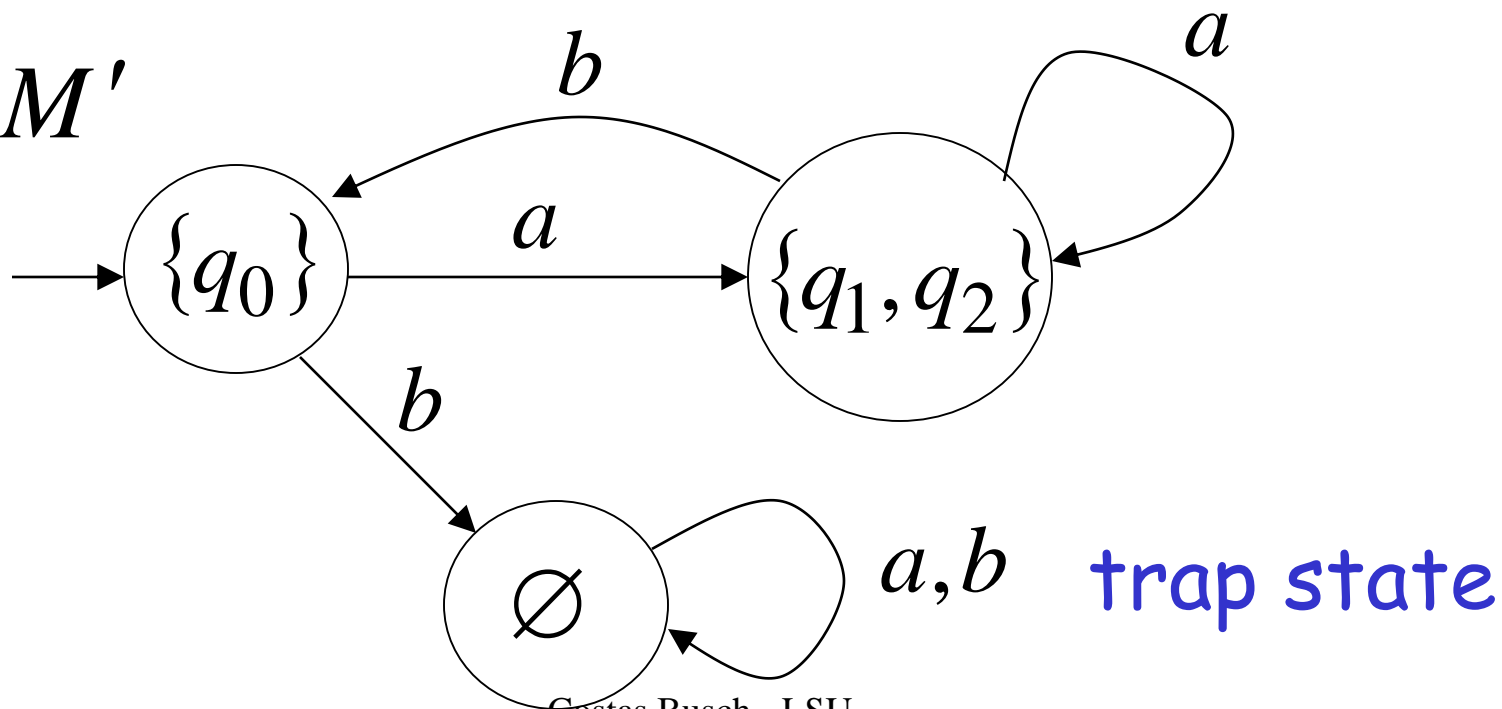
DFA M'



NFA M

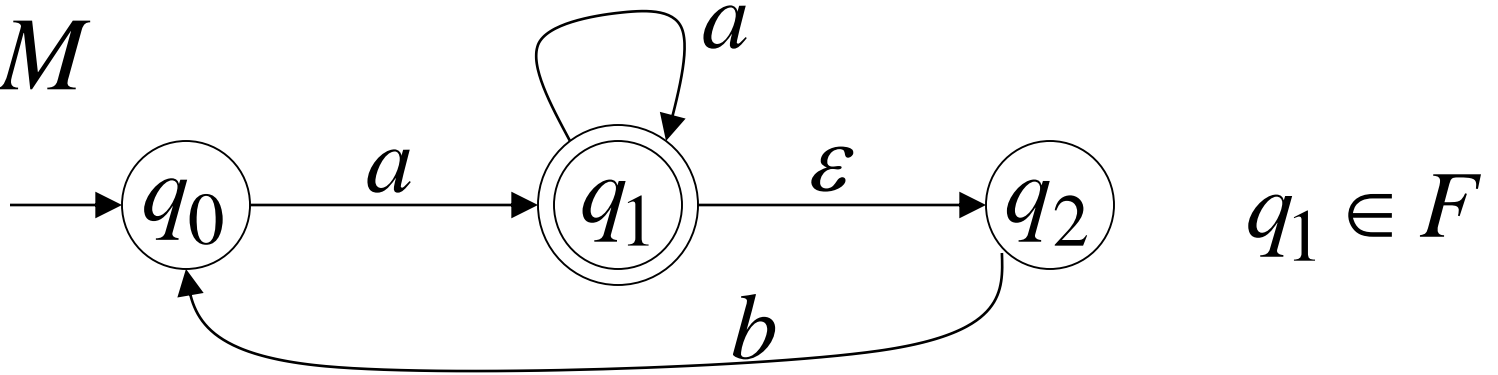


DFA M'

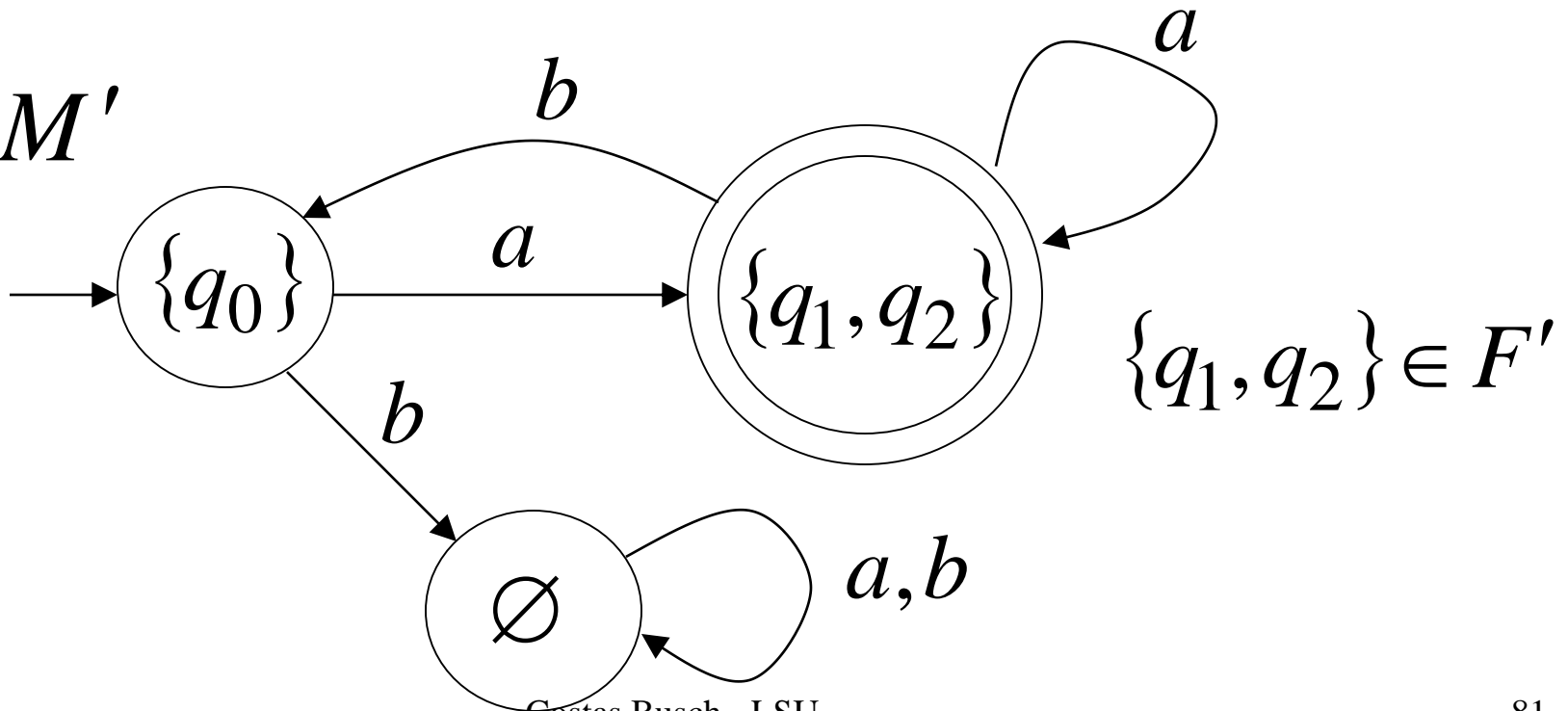


END OF CONSTRUCTION

NFA M



DFA M'



General Conversion Procedure

Input: an NFA M

Output: an equivalent DFA M'
with $L(M) = L(M')$

The NFA has states q_0, q_1, q_2, \dots

The DFA has states from the power set

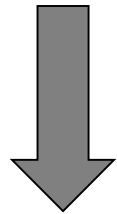
$\emptyset, \{q_0\}, \{q_1\}, \{q_0, q_1\}, \{q_1, q_2, q_3\}, \dots$

Conversion Procedure Steps

step

1. Initial state of NFA: q_0

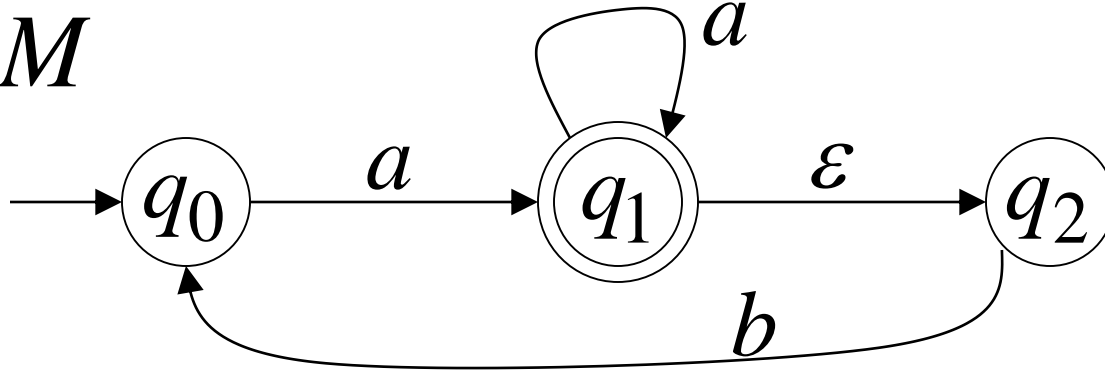
$$\delta^*(q_0, \varepsilon) = \{q_0, K\}$$



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

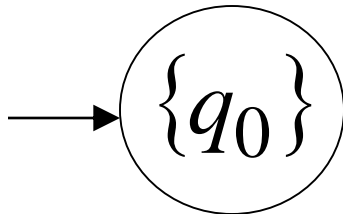
Example

NFA M



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

DFA M'



step

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

compute in the NFA

$$\left. \begin{array}{l} \delta^*(q_i, a) \\ \cup \delta^*(q_j, a) \\ \dots \\ \cup \delta^*(q_m, a) \end{array} \right\} = \text{Union } \{q'_k, q'_l, \dots, q'_n\}$$

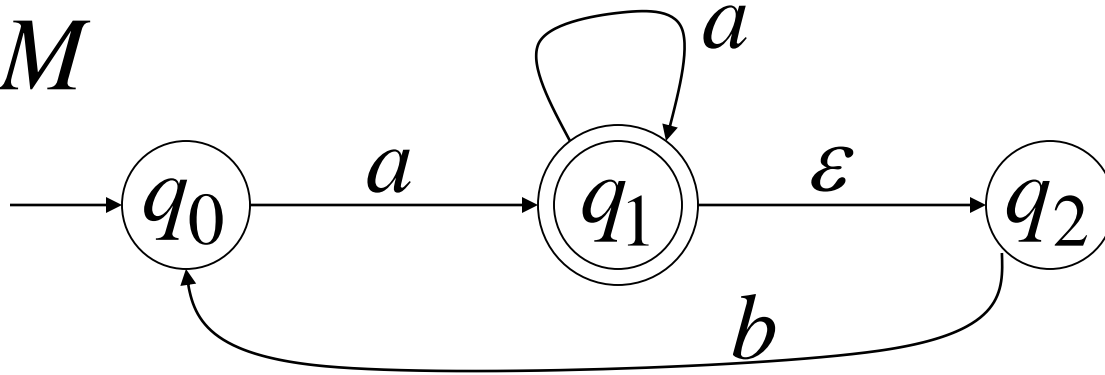
add transition to DFA

$$\delta(\{q_i, q_j, \dots, q_m\}, a) = \{q'_k, q'_l, \dots, q'_n\}$$

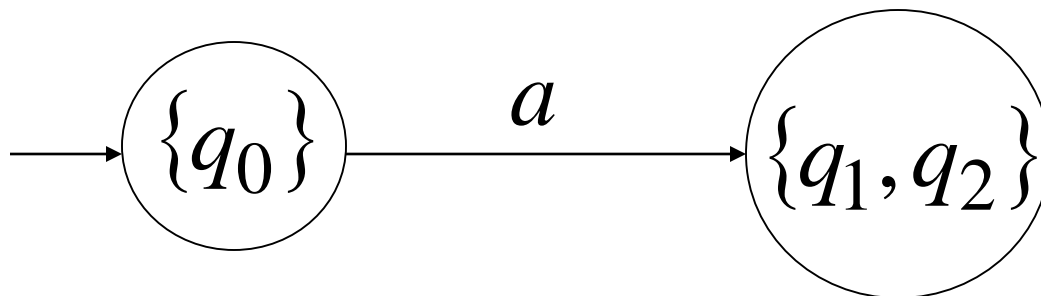
Example

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

NFA M



DFA M'



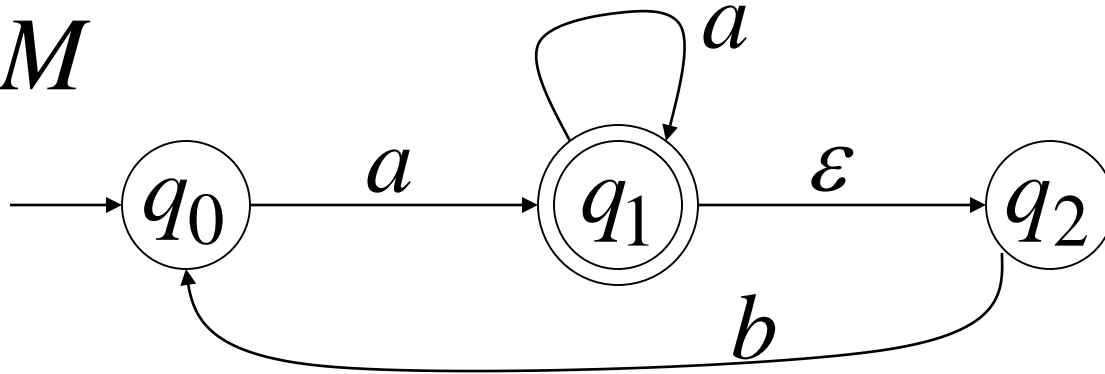
$$\delta(\{q_0\}, a) = \{q_1, q_2\}$$

step

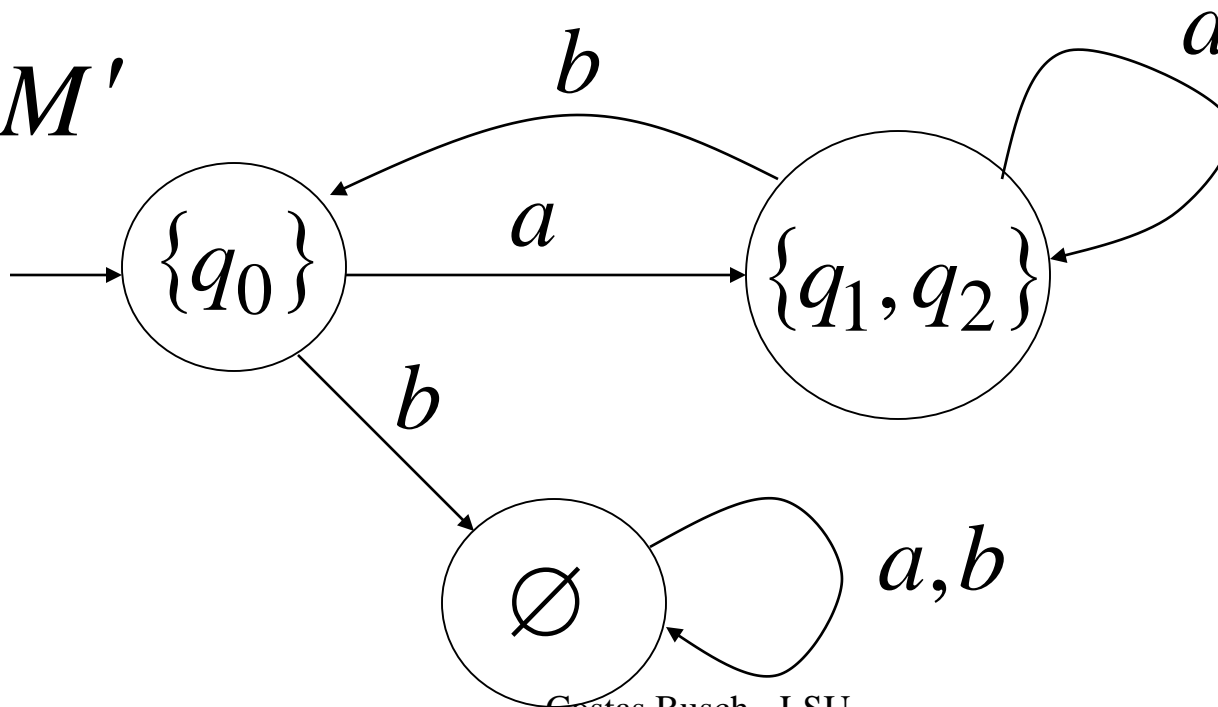
3. Repeat Step 2 for every state in DFA and symbols in alphabet until no more states can be added in the DFA

Example

NFA M



DFA M'



step

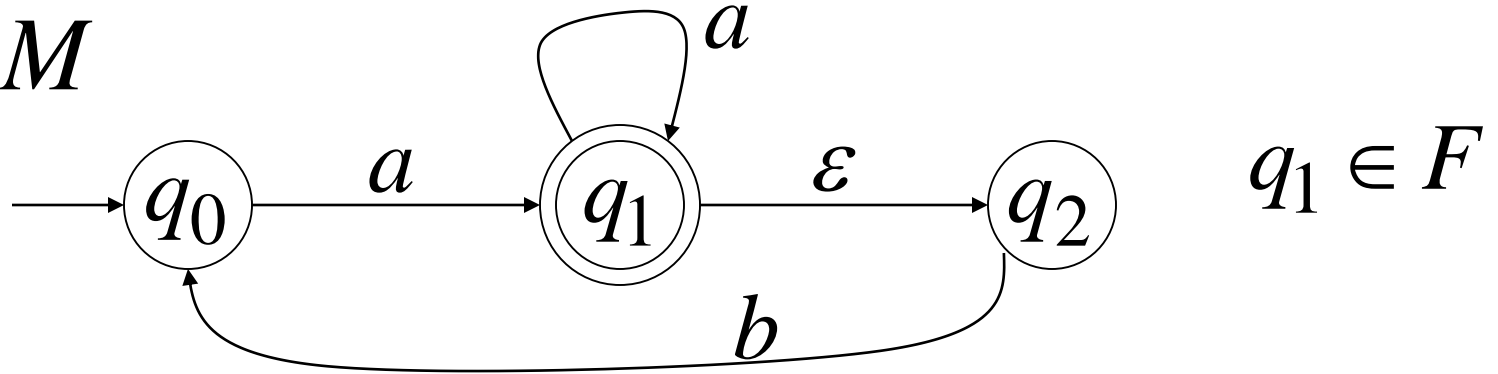
4. For any DFA state $\{q_i, q_j, \dots, q_m\}$

if some q_j is accepting state in NFA

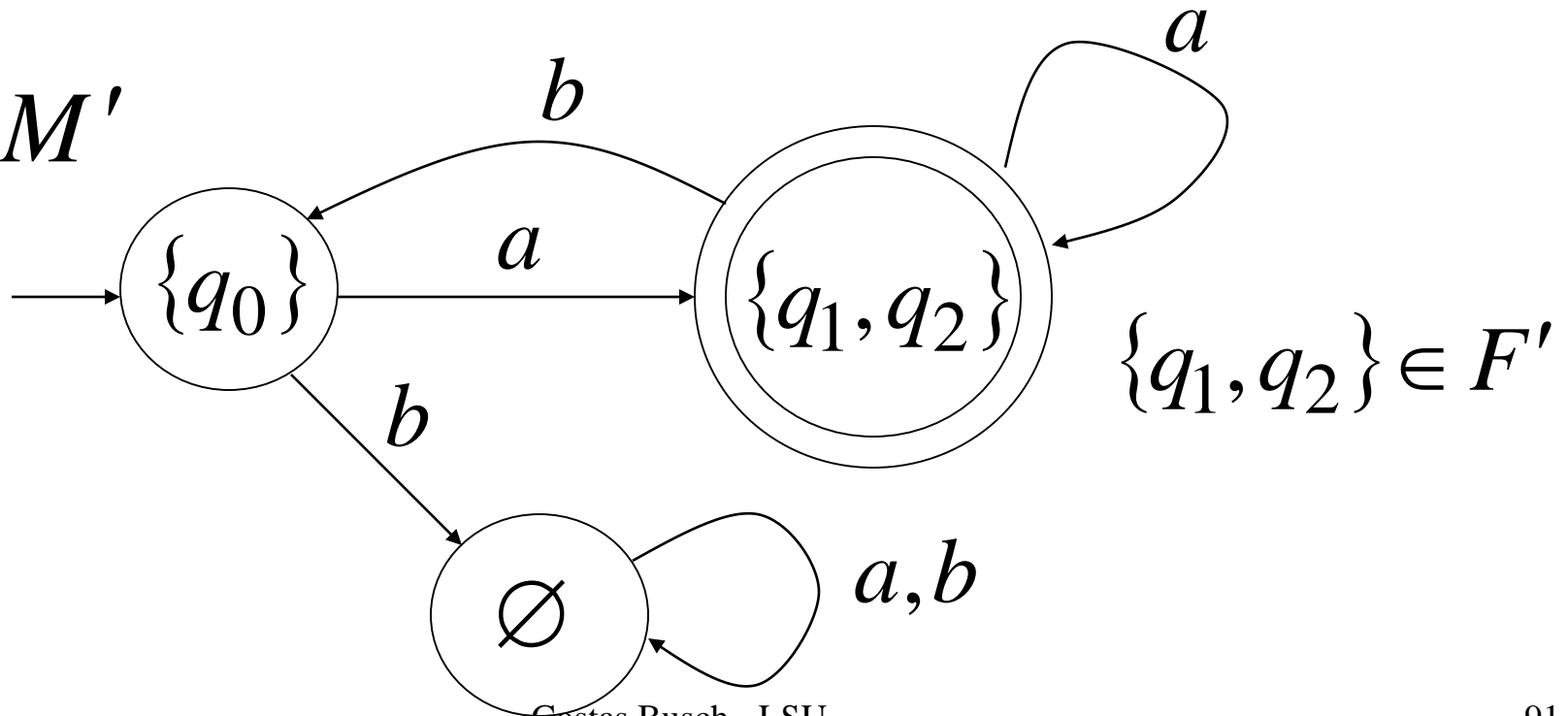
Then, $\{q_i, q_j, \dots, q_m\}$
is accepting state in DFA

Example

NFA M



DFA M'



Lemma:

If we convert NFA M to DFA M'
then the two automata are equivalent:

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

Proof:

We need to show: $L(M) \subseteq L(M')$

AND

$$L(M) \supseteq L(M')$$

First we show: $L(M) \subseteq L(M')$

We only need to prove:

$$w \in L(M) \quad \longrightarrow \quad w \in L(M')$$

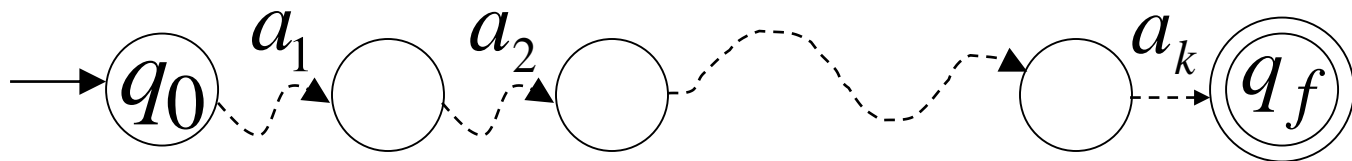
NFA

Consider $w \in L(M)$

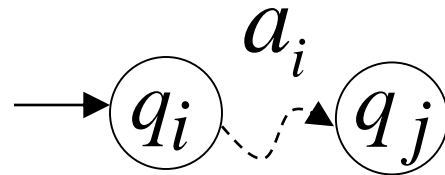


symbols

$$w = a_1 a_2 \Lambda a_k$$

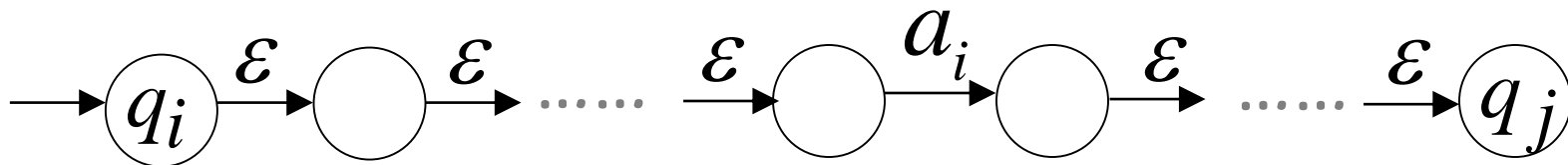


symbol



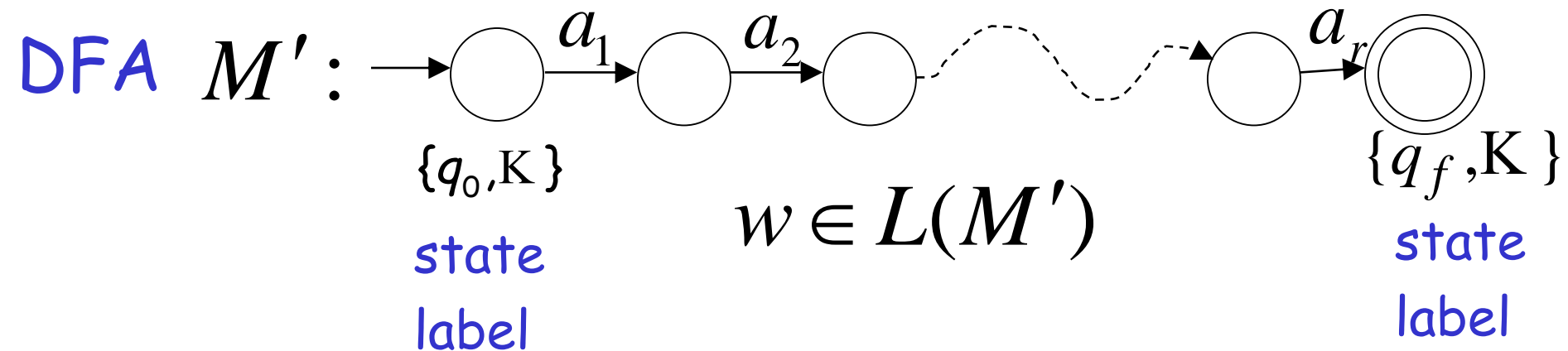
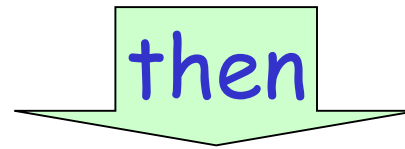
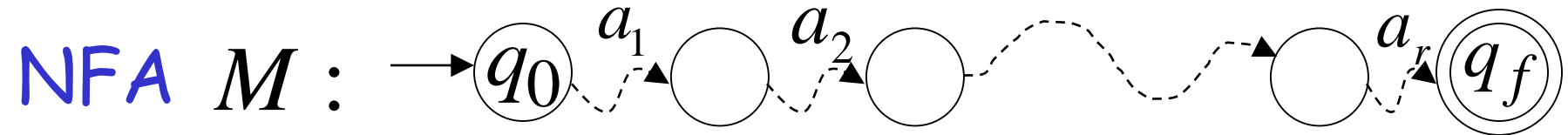
denotes a possible sub-path like

symbol



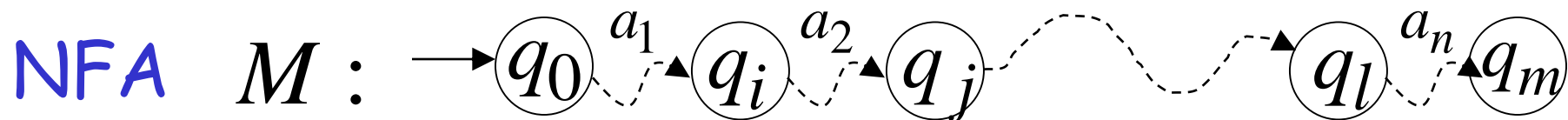
We will show that if $w \in L(M)$

$$w = a_1 a_2 \Lambda a_r$$

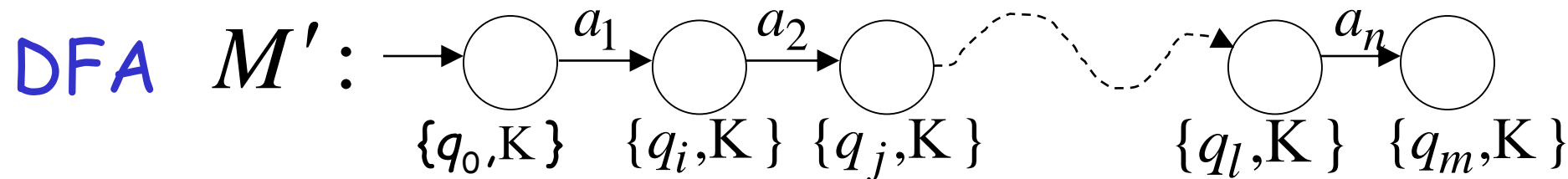


More generally, we will show that if in M :

(arbitrary prefix) $v = a_1 a_2 \Lambda a_n \quad n \leq r$

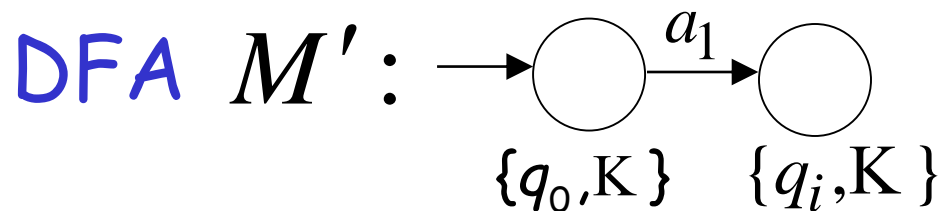
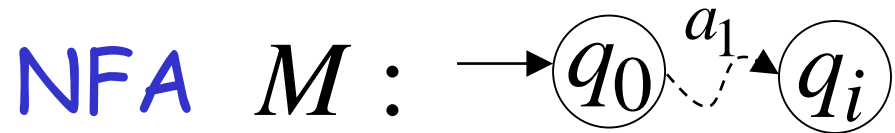


then



Proof by induction on $|v|$

Induction Basis: $|v| = 1$ $v = a_1$

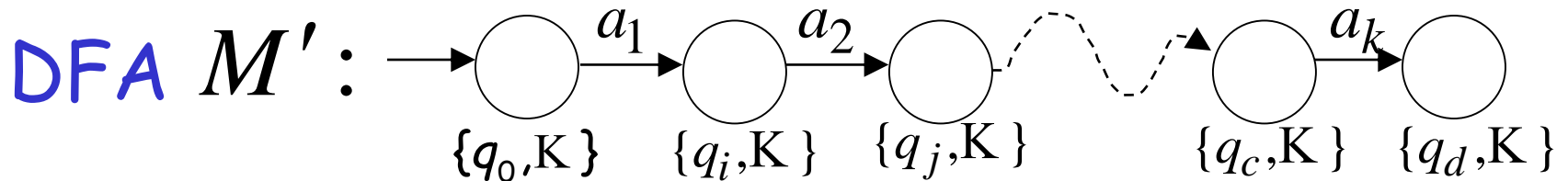
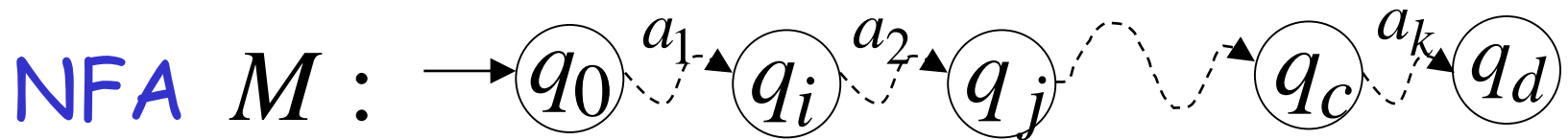


is true by construction of M'

Induction hypothesis: $1 \leq |v| \leq k$

$$v = a_1 a_2 \Lambda a_k$$

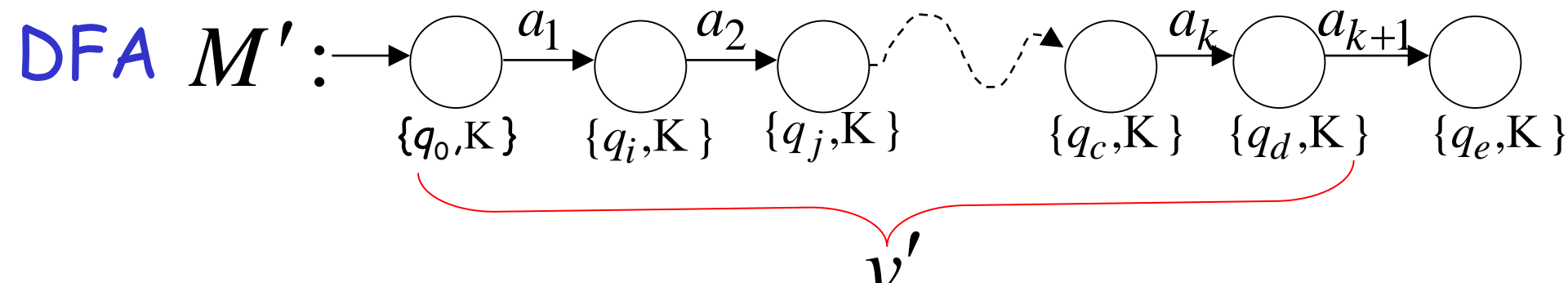
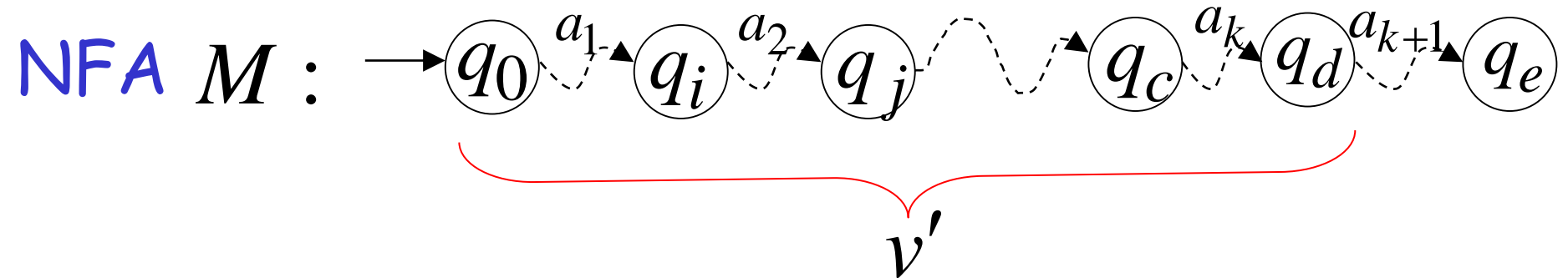
Suppose that the following hold



Induction Step: $|v| = k + 1$

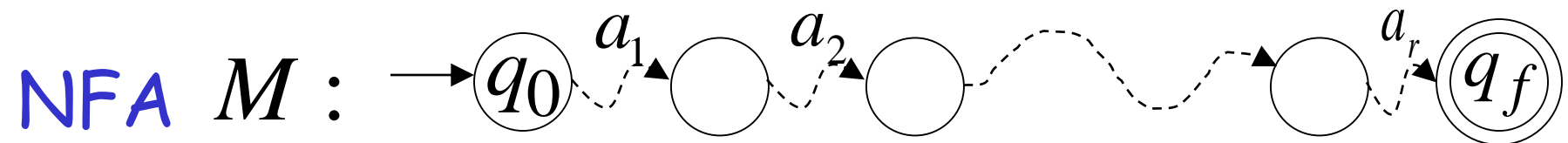
$$v = \underbrace{a_1 a_2 \dots a_k}_{v'} a_{k+1} = v' a_{k+1}$$

Then this is true by construction of M'

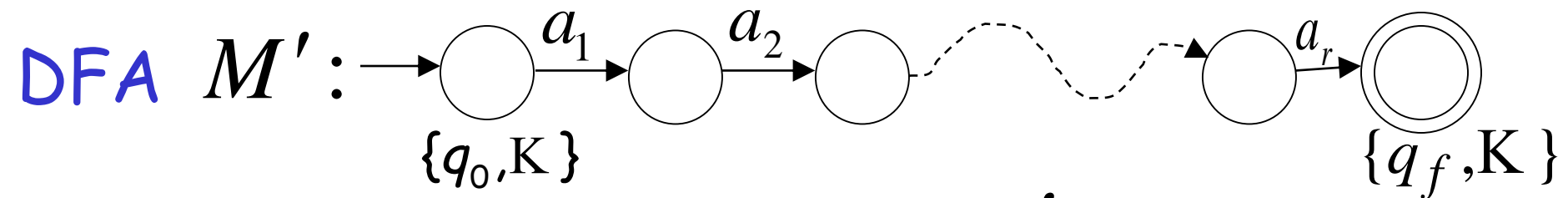


Therefore if $w \in L(M)$

$$w = a_1 a_2 \Lambda a_r$$



then



$$w \in L(M')$$

We have shown: $L(M) \subseteq L(M')$

With a similar proof

we can show: $L(M) \supseteq L(M')$

Therefore: $L(M) = L(M')$

END OF PROOF