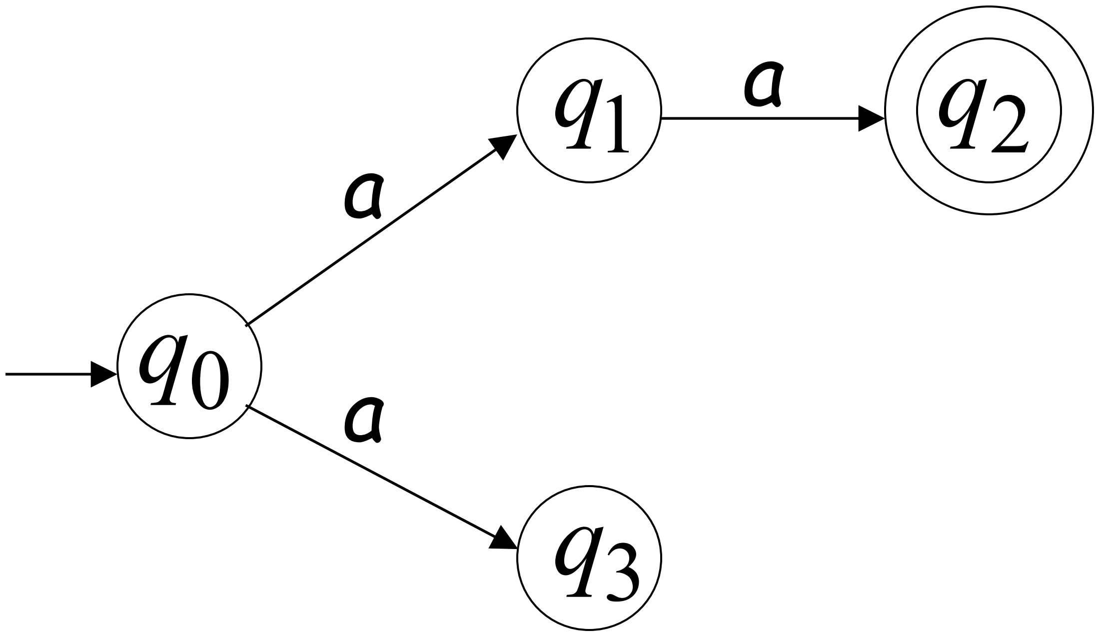
Nondeterministic Finite Automata

Nondeterministic Finite Automata (NFA)

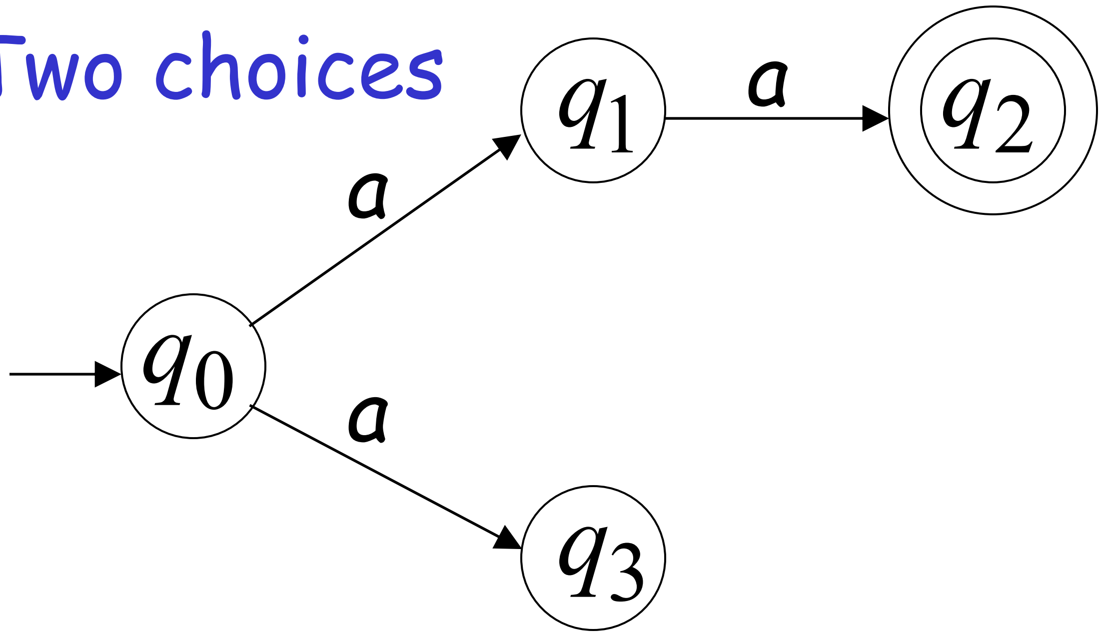
Alphabet = $\{a\}$



Nondeterministic Finite Automata (NFA)

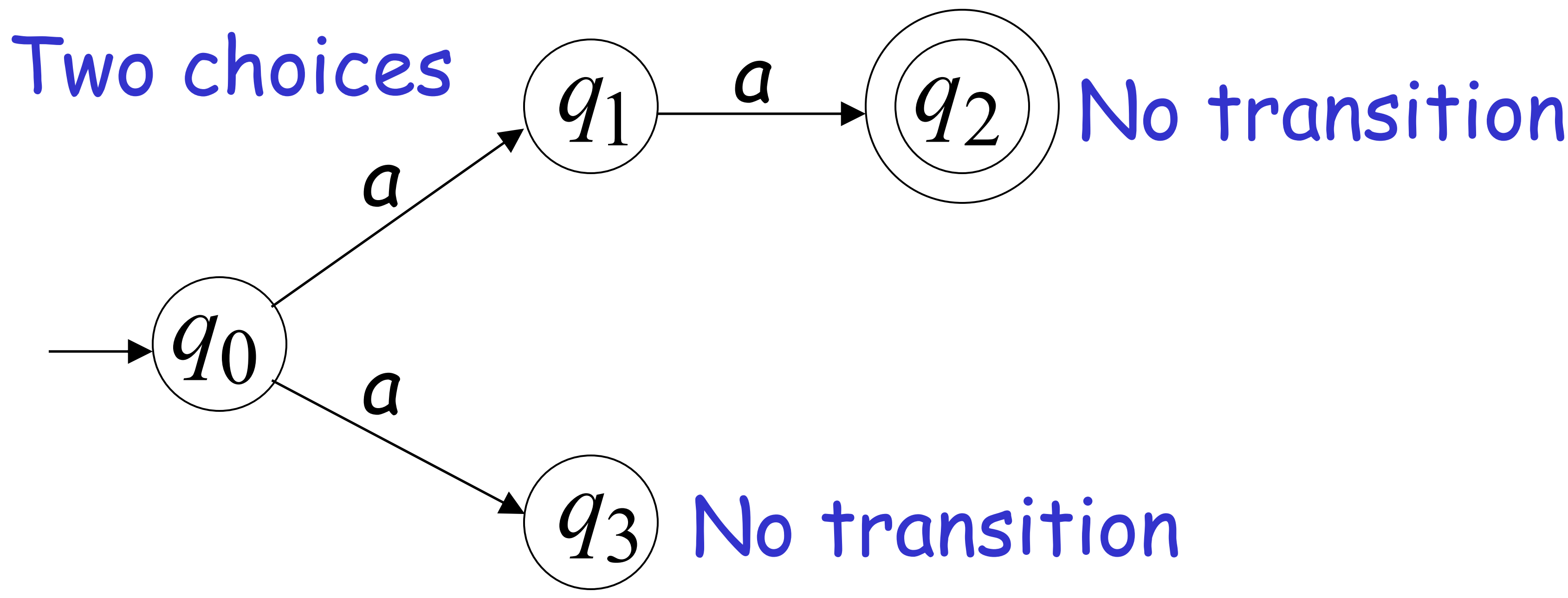
Alphabet = $\{a\}$

Two choices

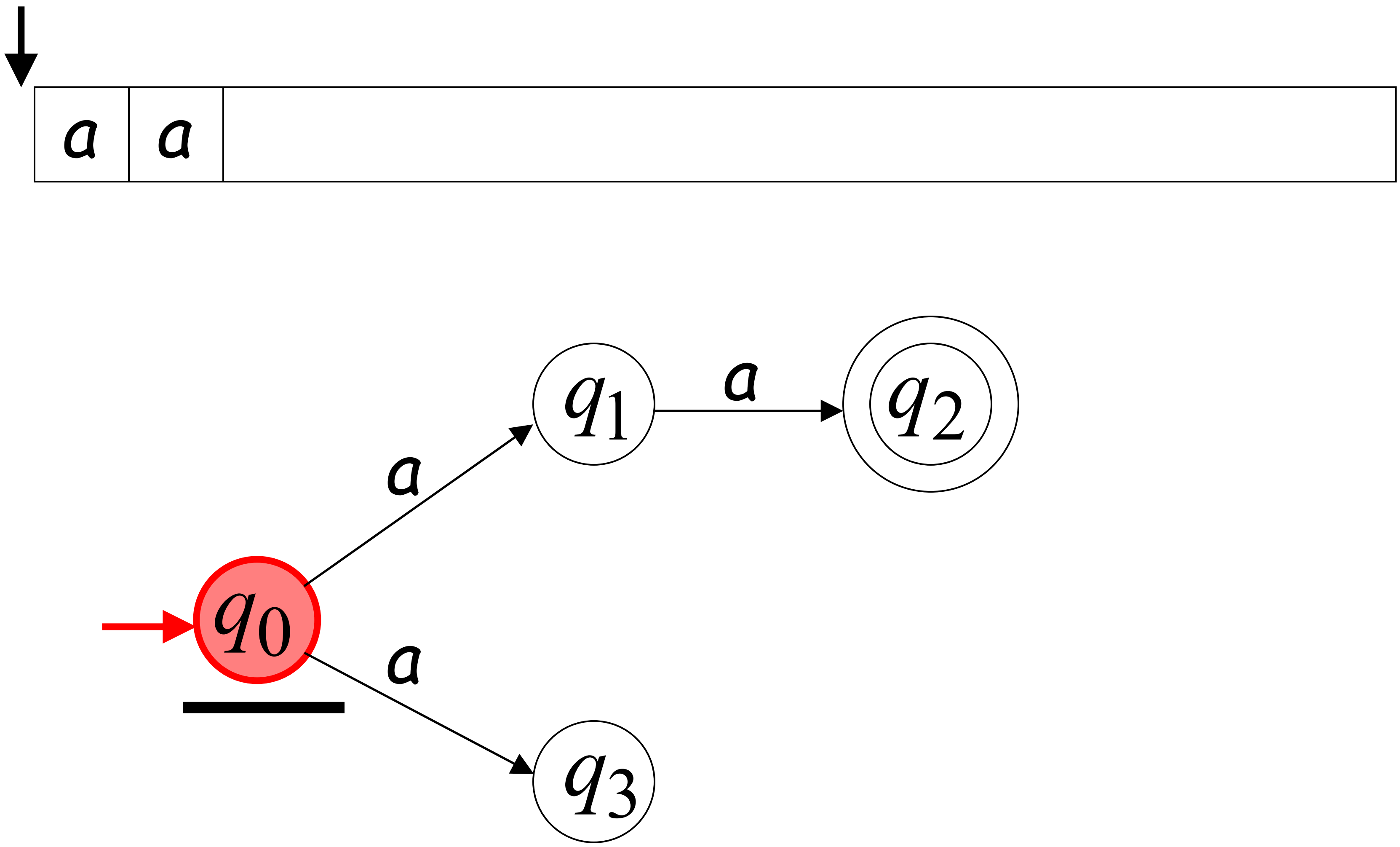


Nondeterministic Finite Automata (NFA)

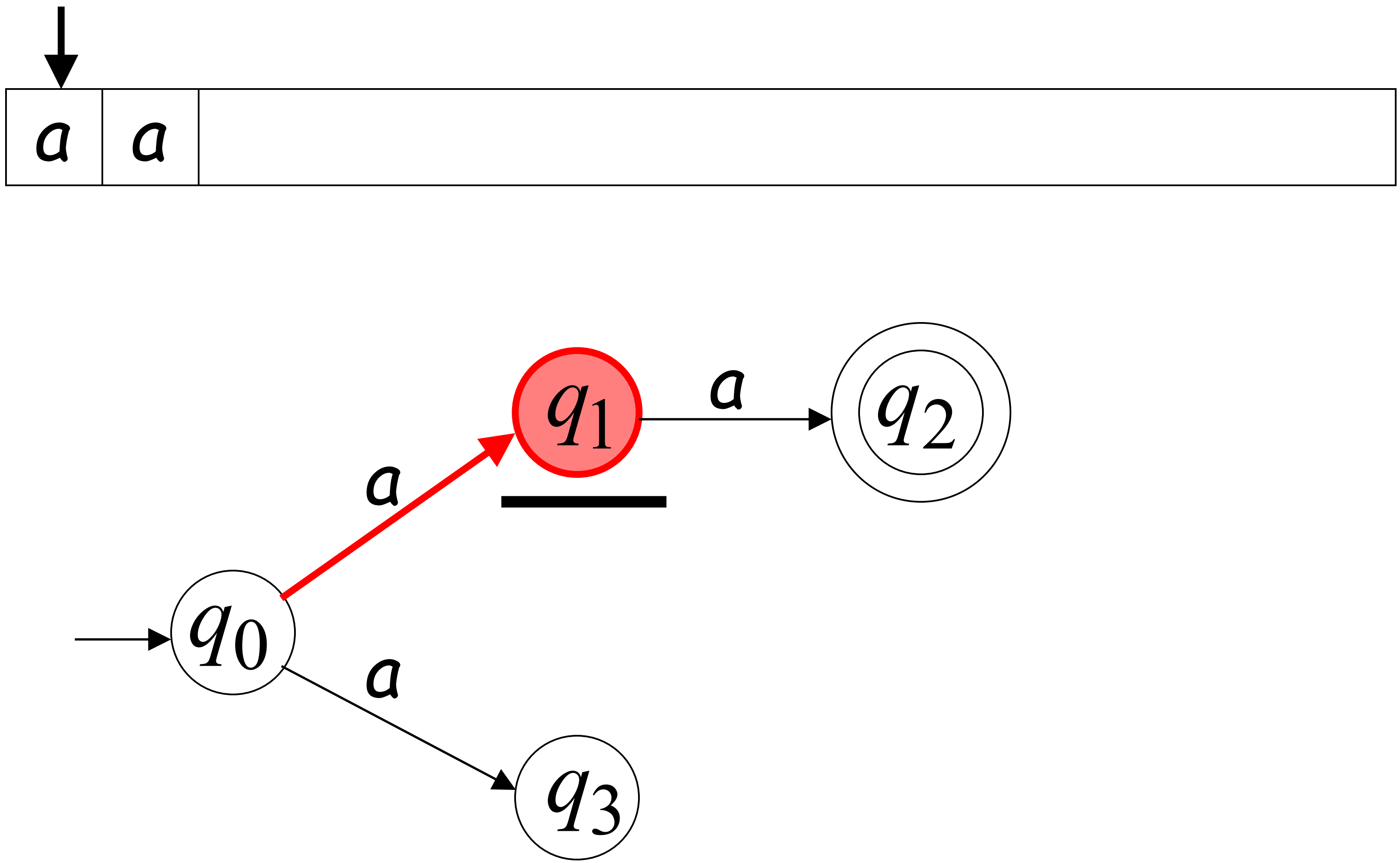
Alphabet = $\{a\}$



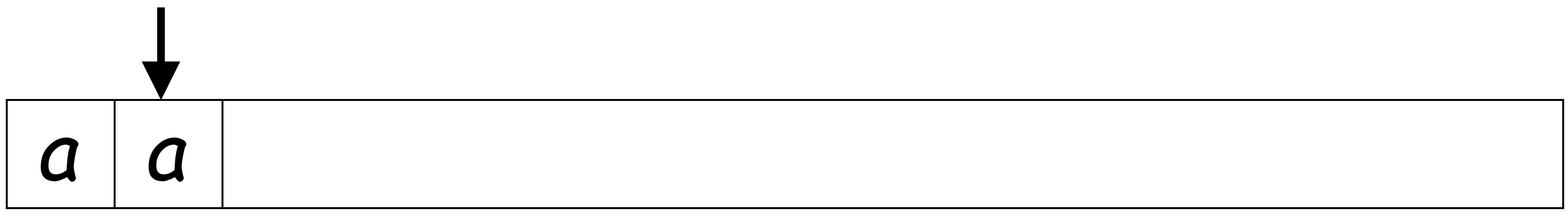
First Choice

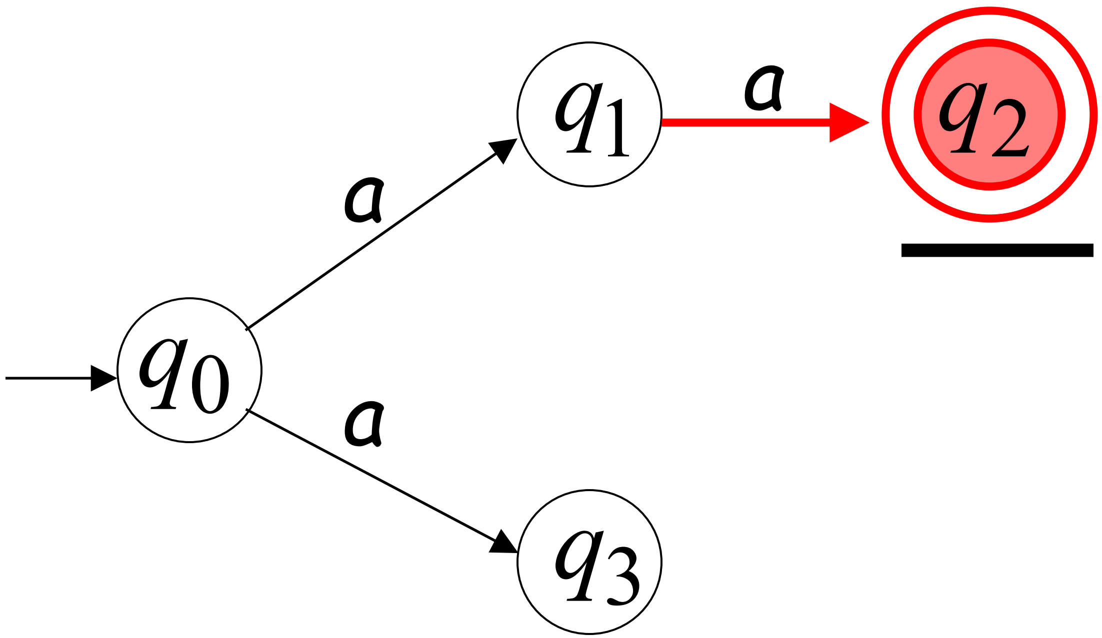


First Choice

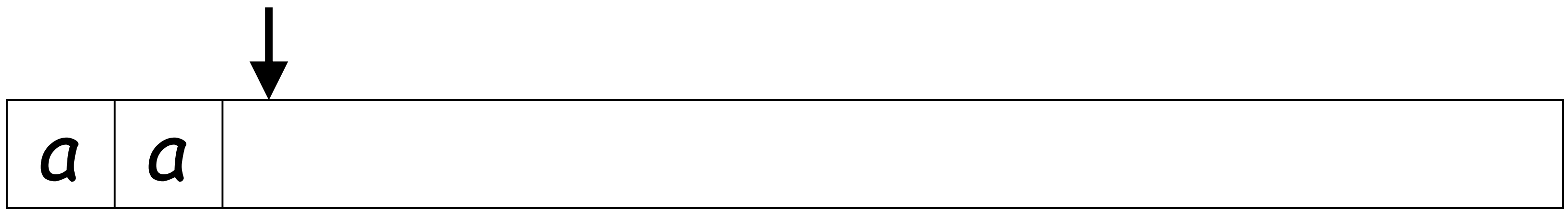


First Choice

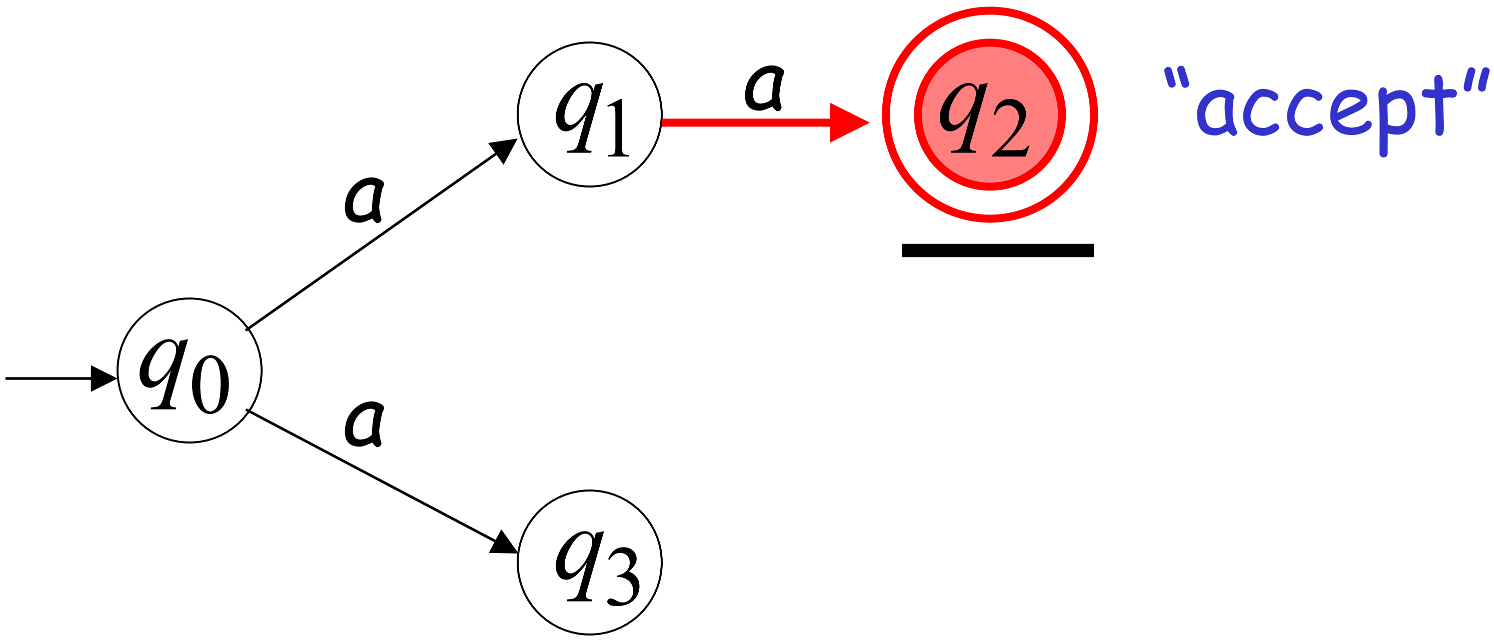




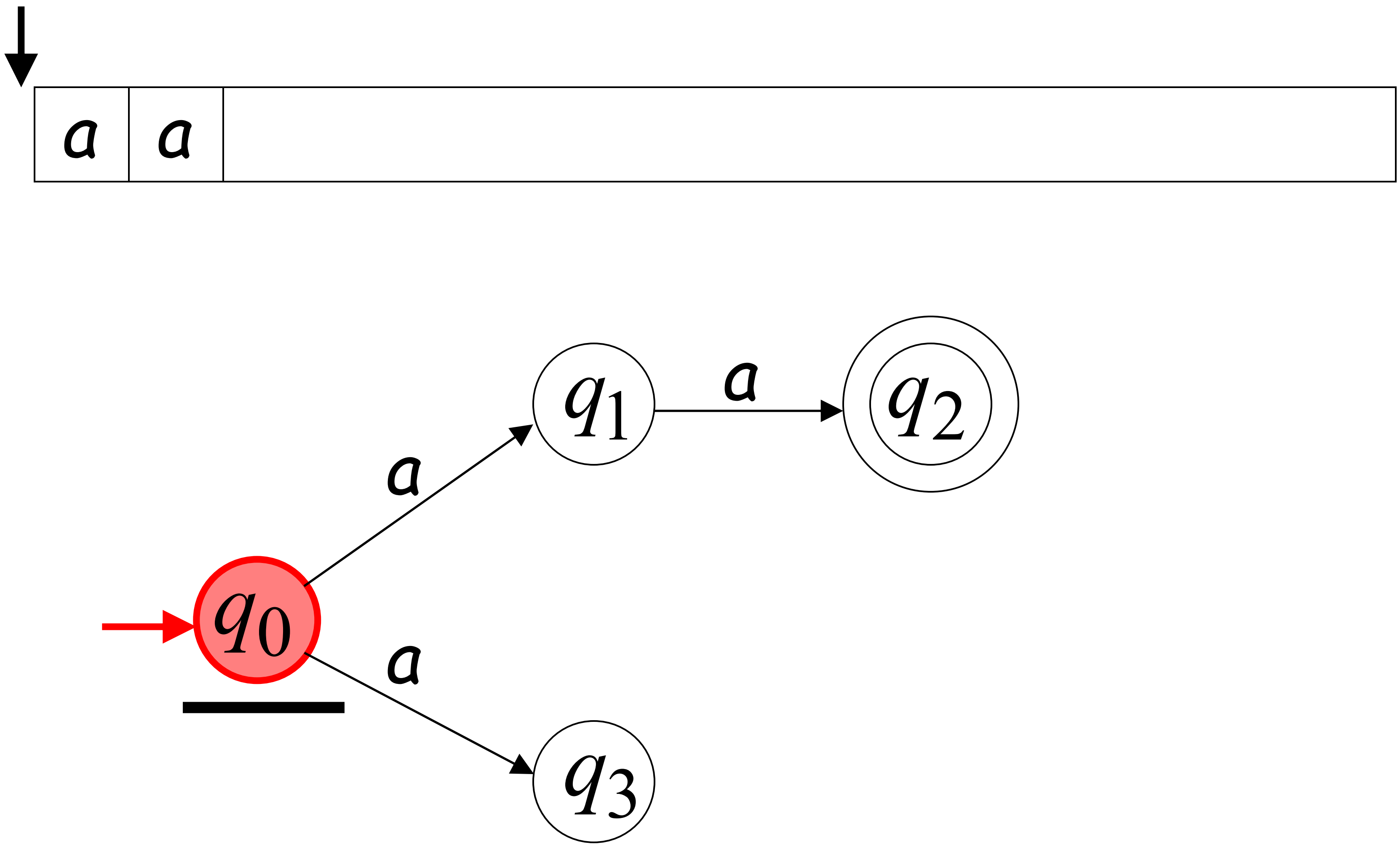
First Choice



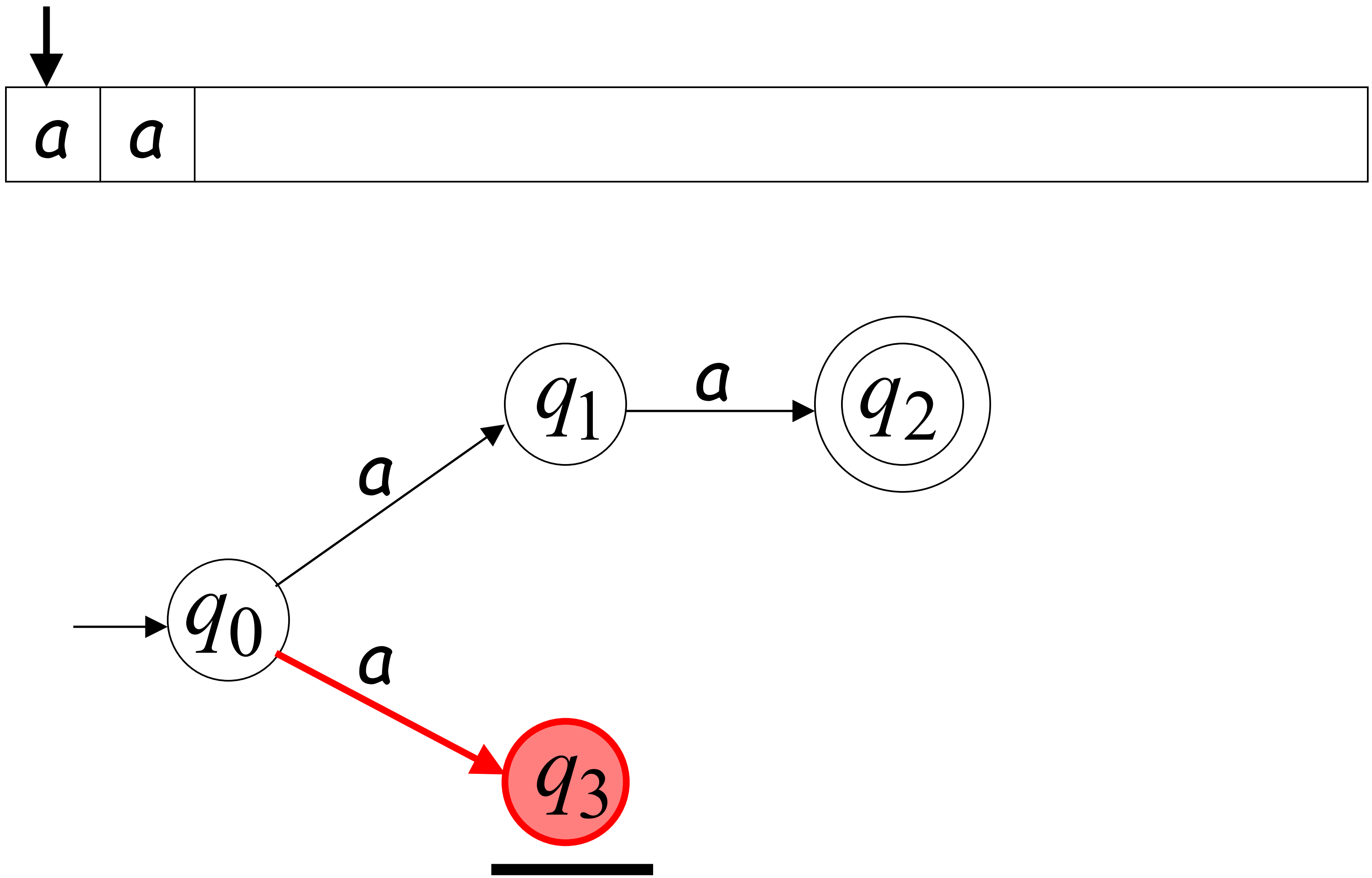
All input is consumed



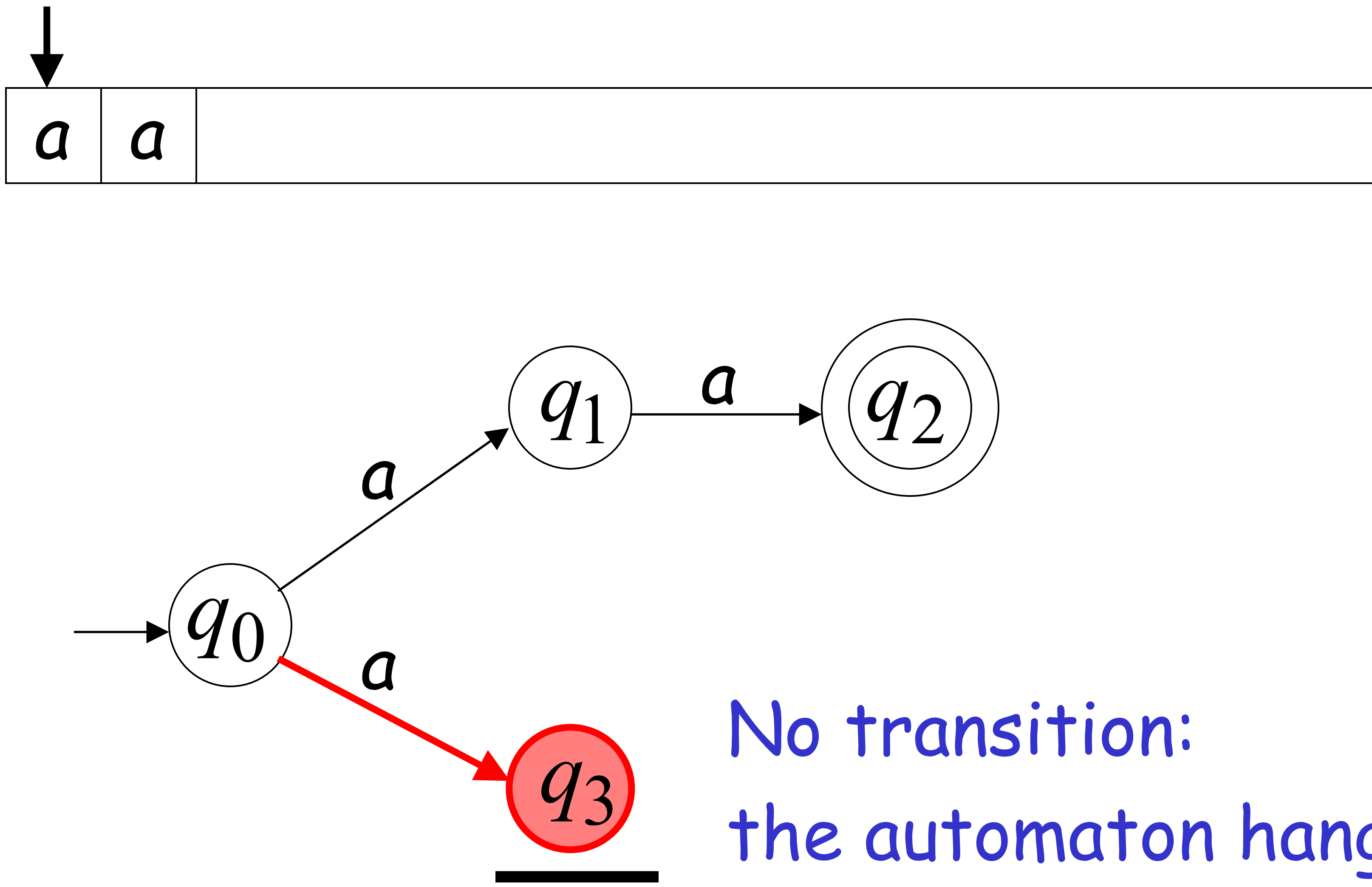
Second Choice



Second Choice

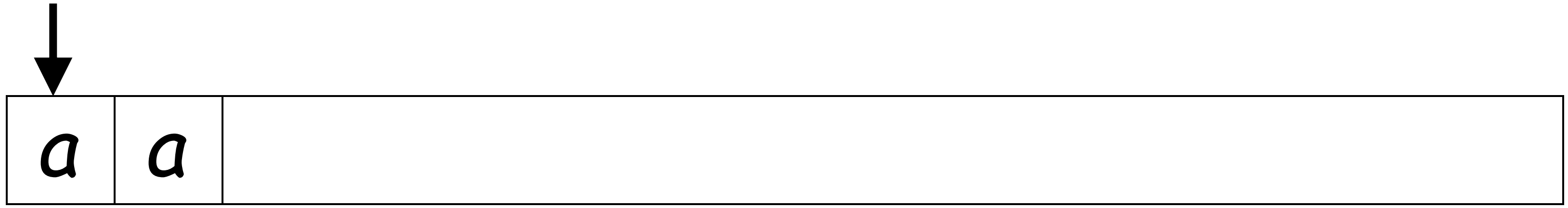


Second Choice

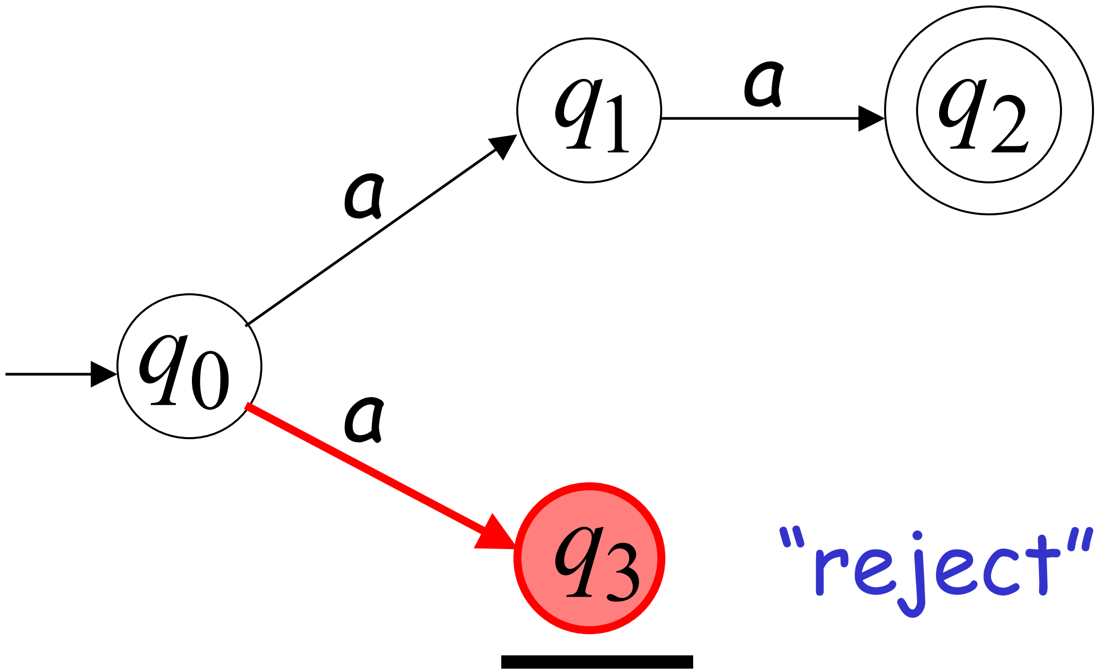


No transition:
the automaton hangs

Second Choice



Input cannot be consumed



"reject"

An NFA accepts a string:

จะยอมรับ string ถ้าเงื่อนไขเป็นจริง

If there is a computation such that:

ทุก Input ต้องได้ใช้

① All the input is consumed

AND

จะต้องหยุดที่ final state

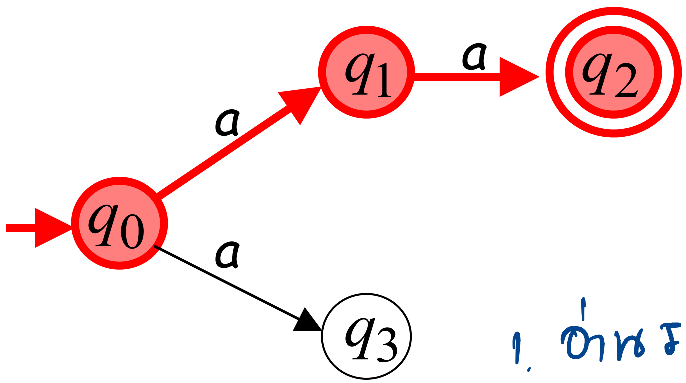
② The automata is in a final state

อย่างน้อย 1 ทางไปที่ final state

Example

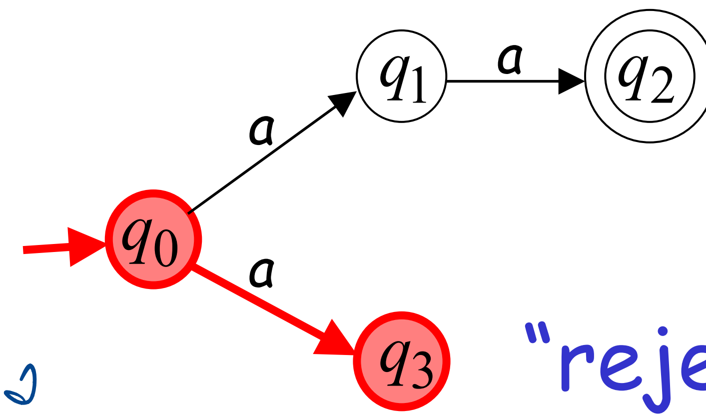
aa is accepted by the NFA:

"accept"



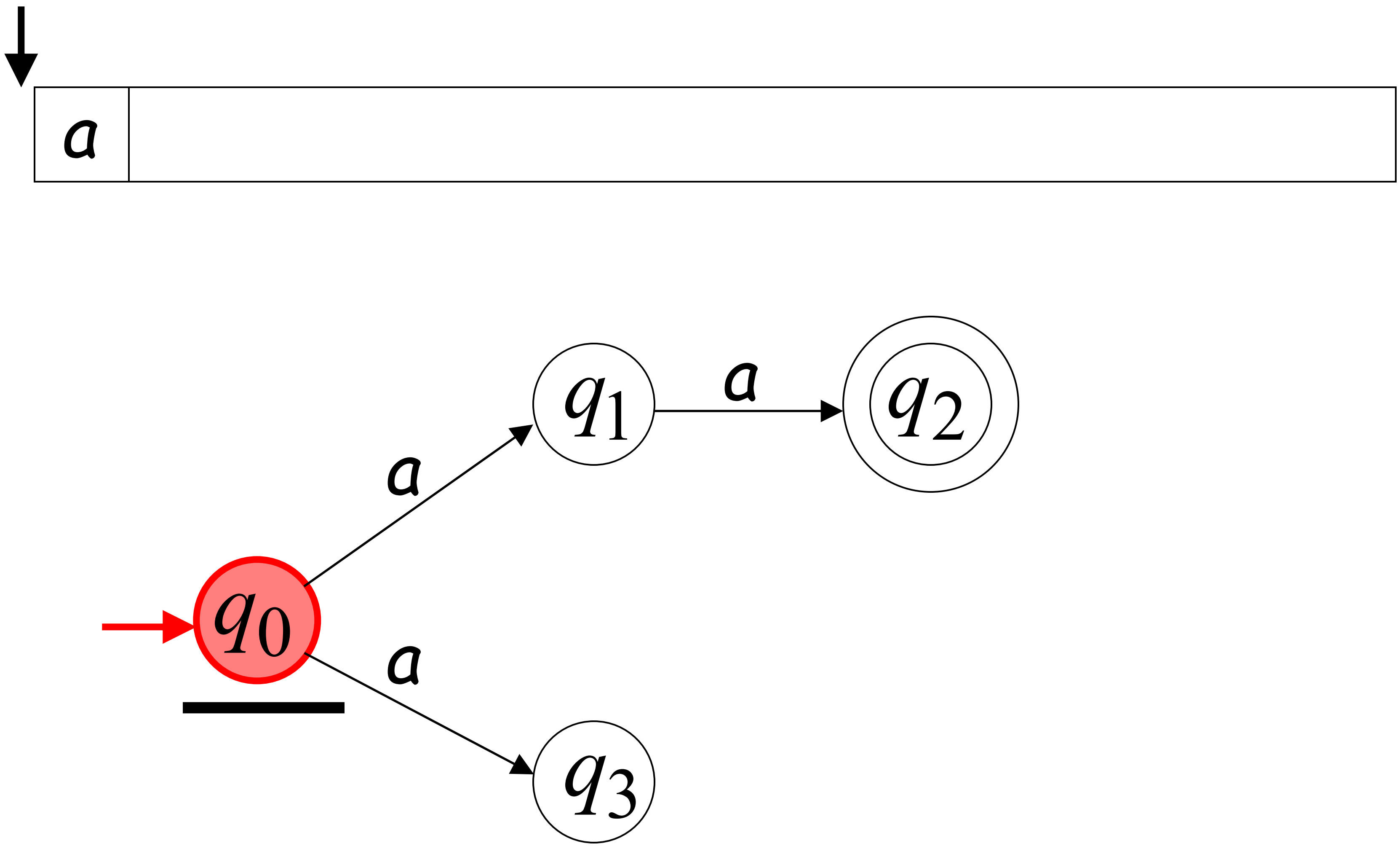
because this computation accepts aa

1. อ่าน input หมด
2. อย่างน้อย 1 ทางไปที่ final state

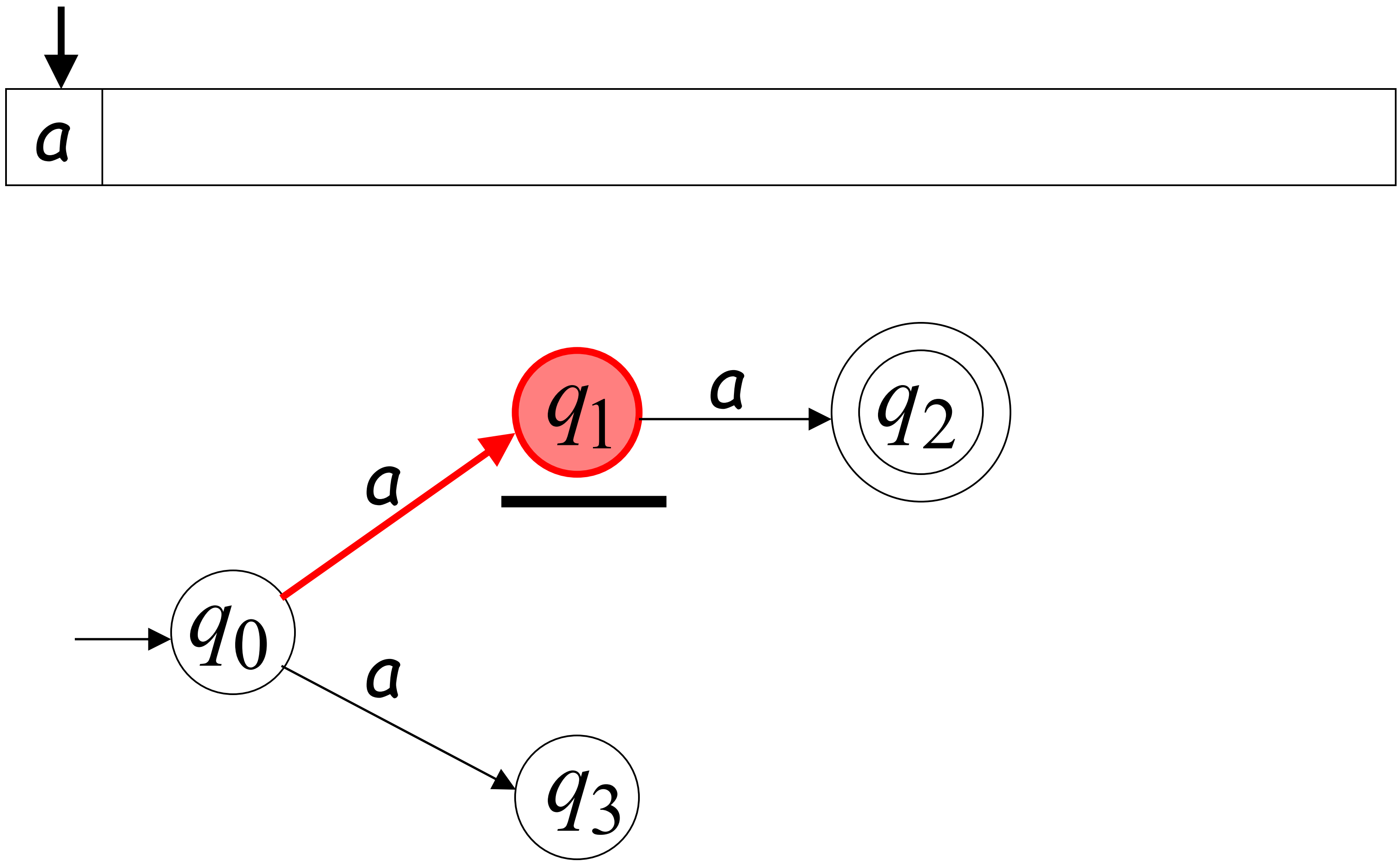


"reject"

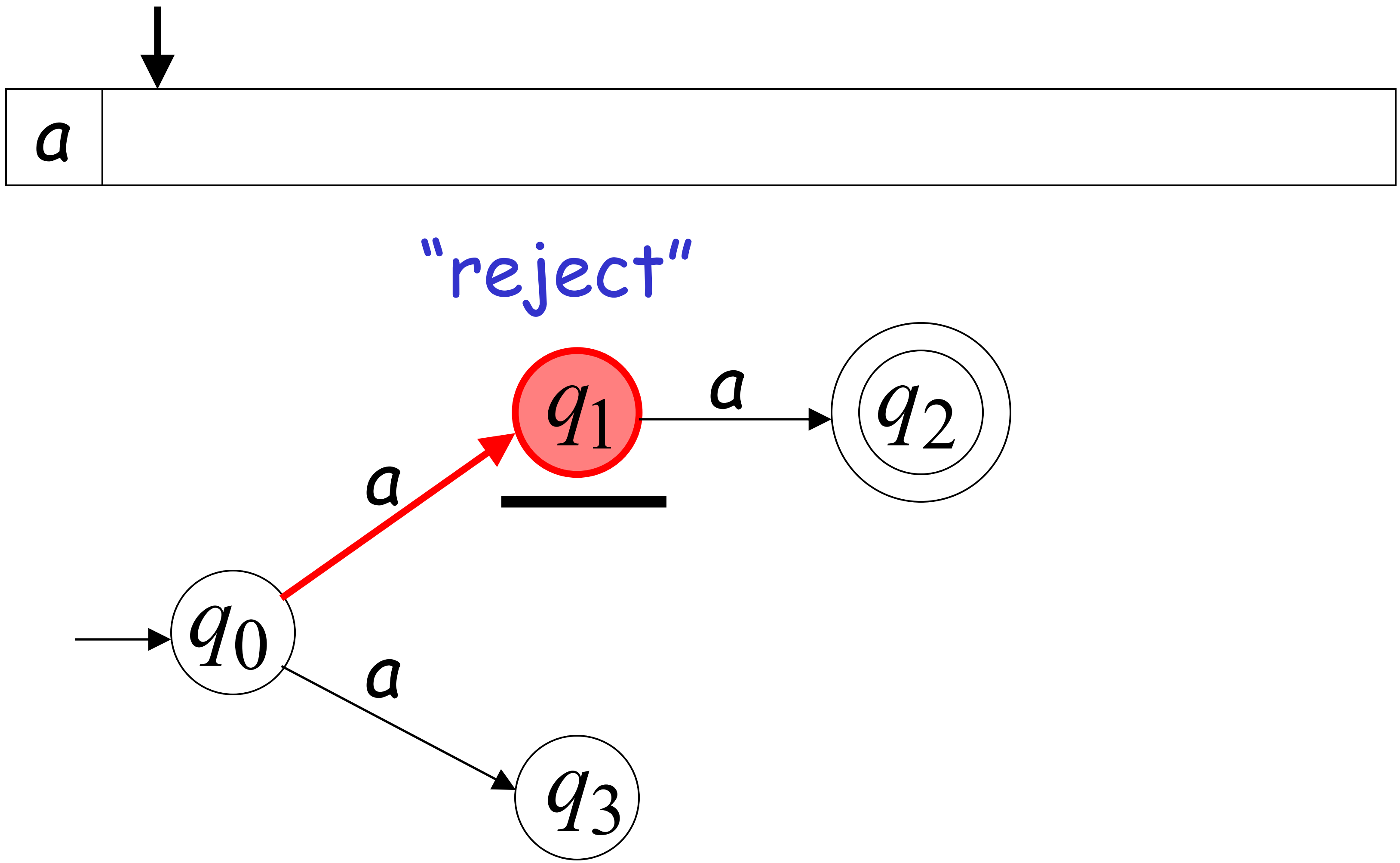
Rejection example



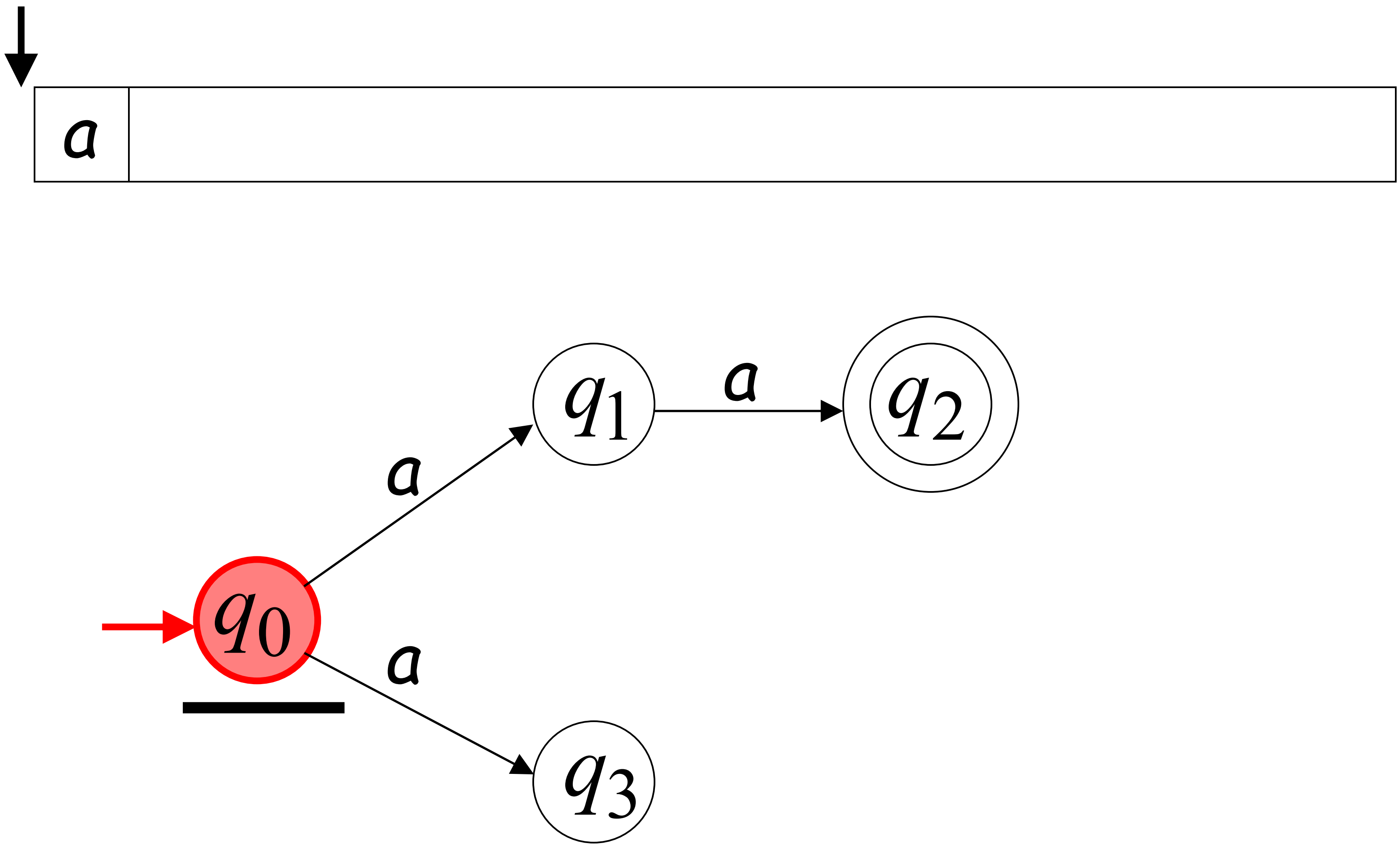
First Choice



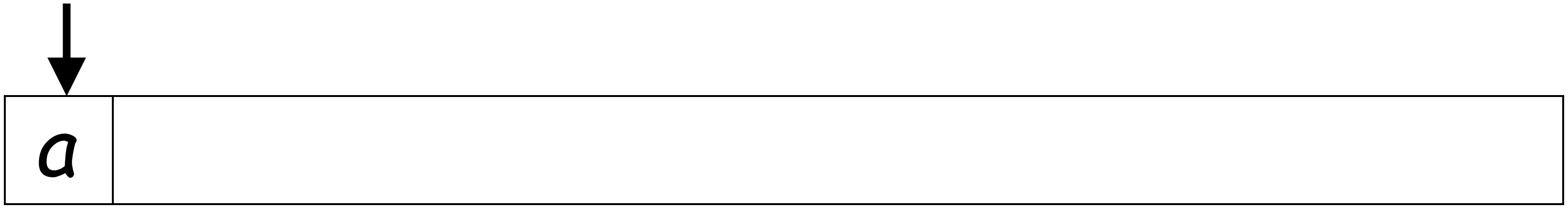
First Choice

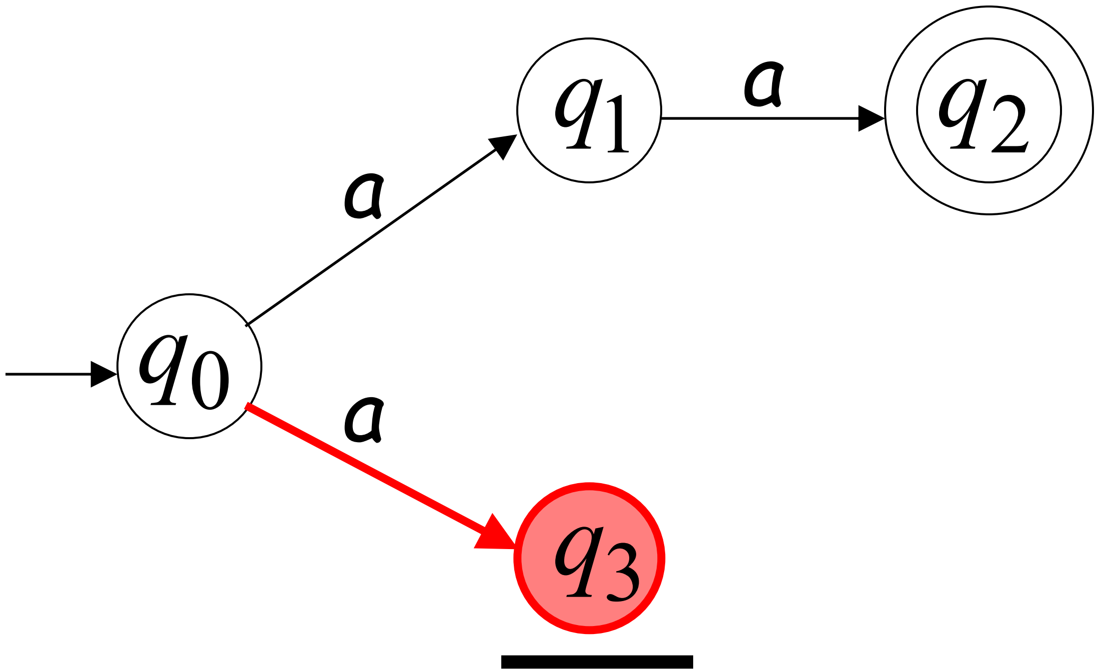


Second Choice

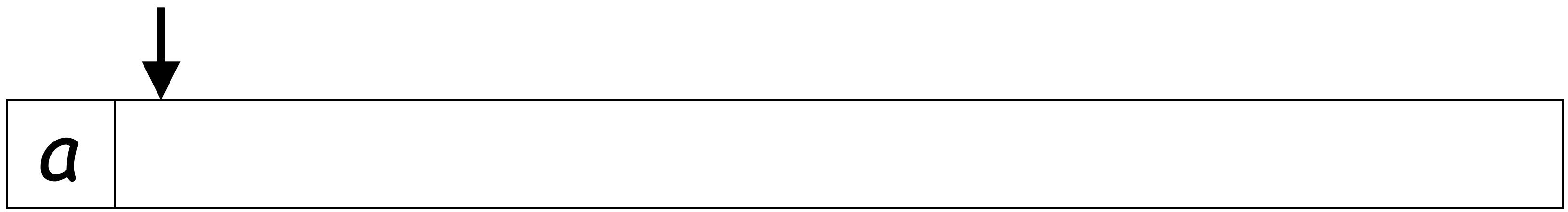


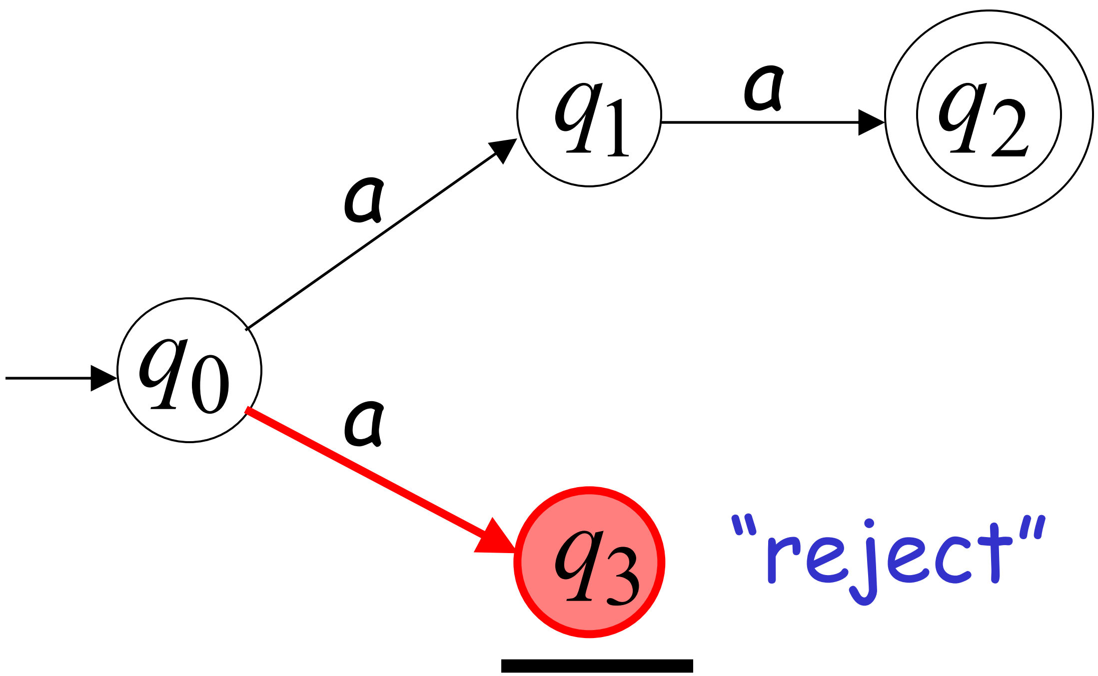
Second Choice



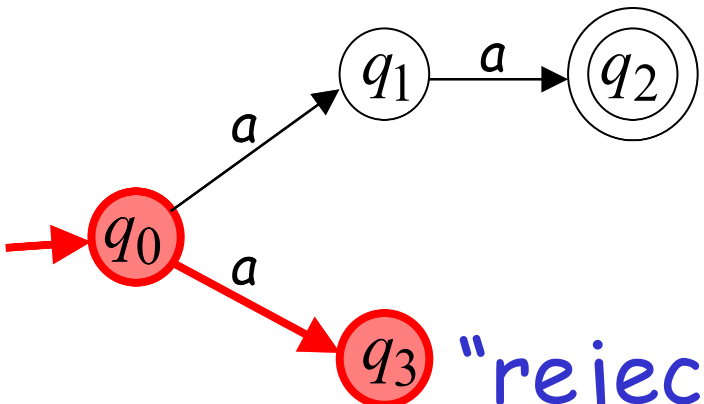


Second Choice



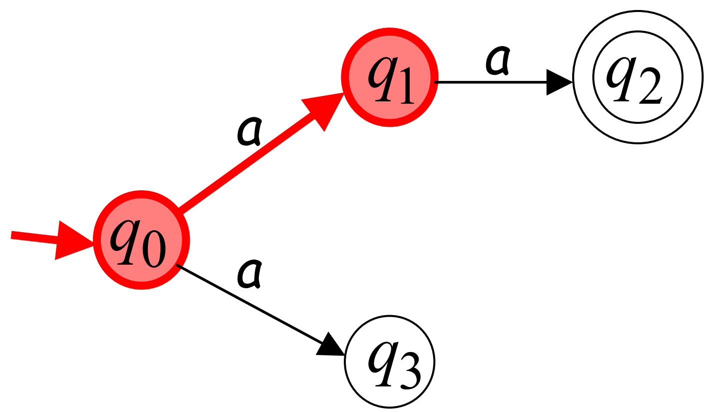


a is rejected by the NFA:



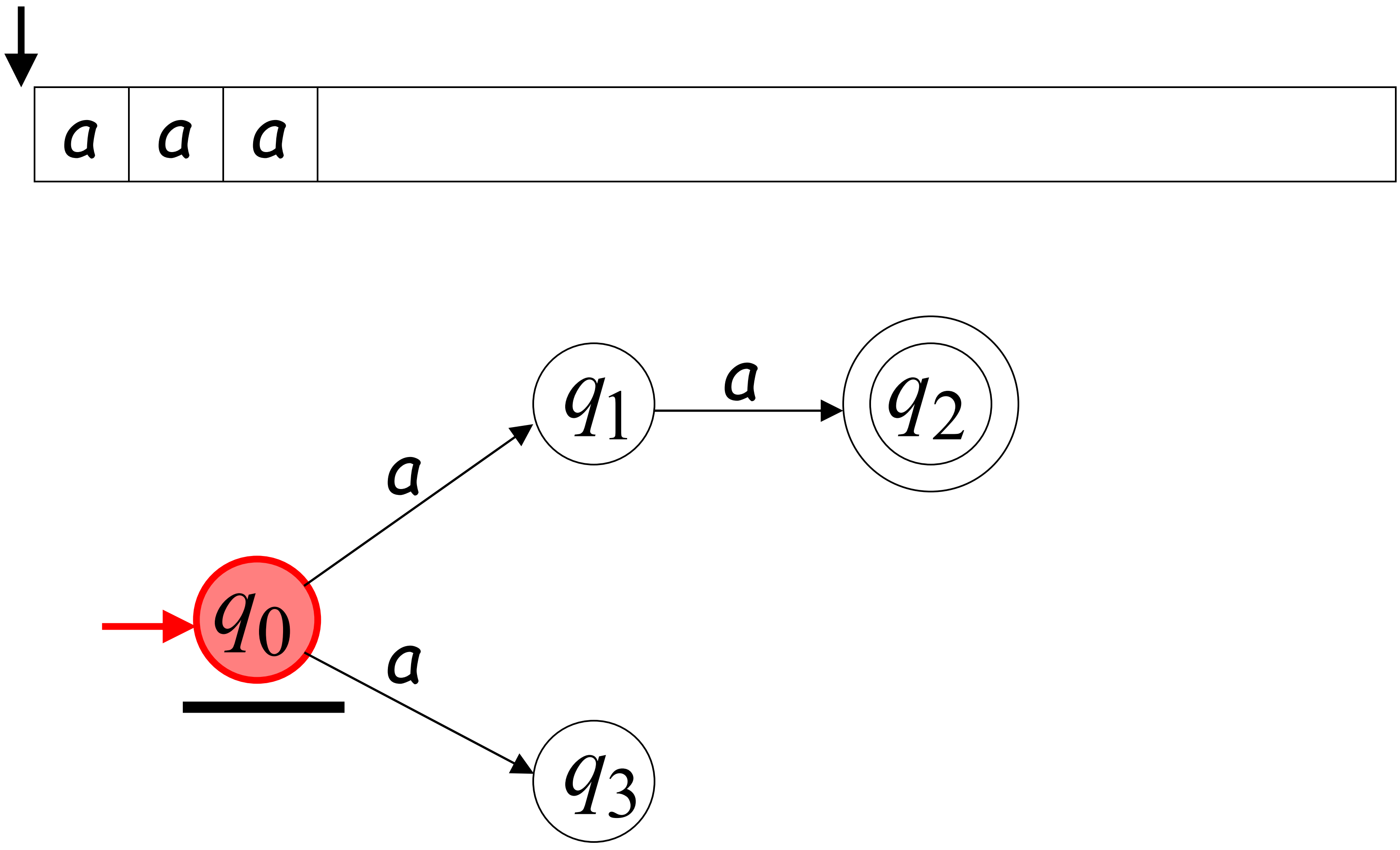
"reject"

"reject"

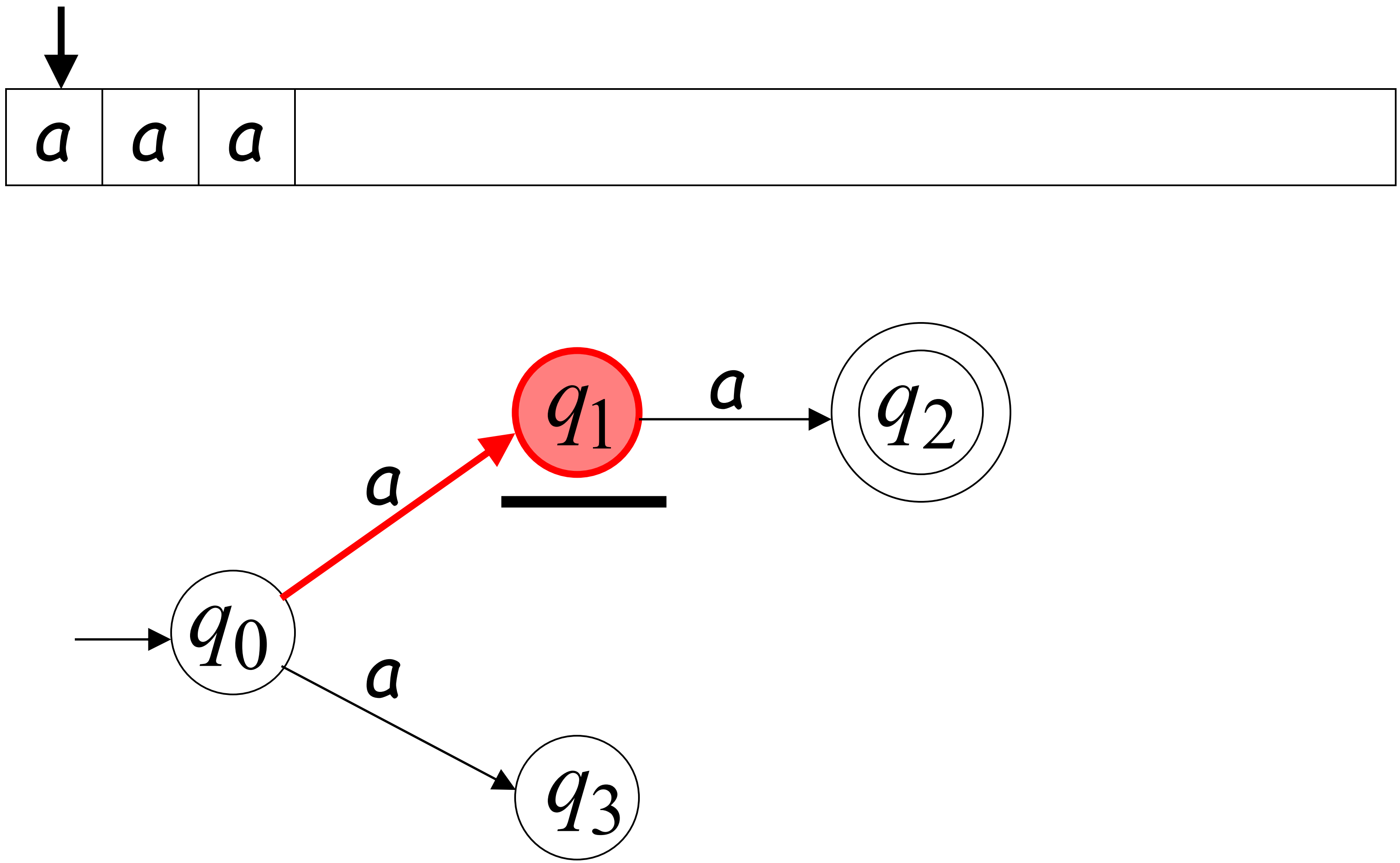


All possible computations lead to rejection

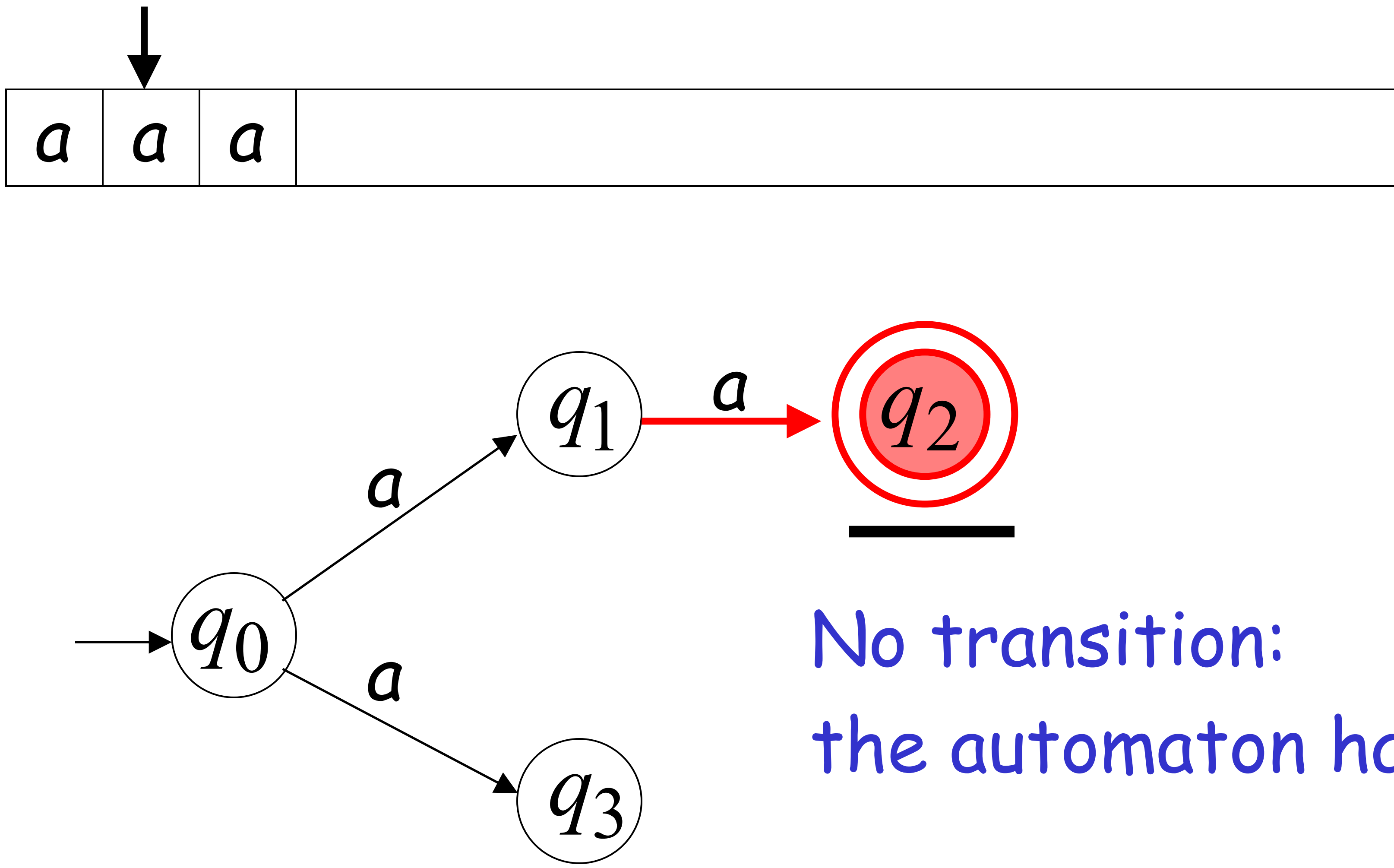
Rejection example



First Choice

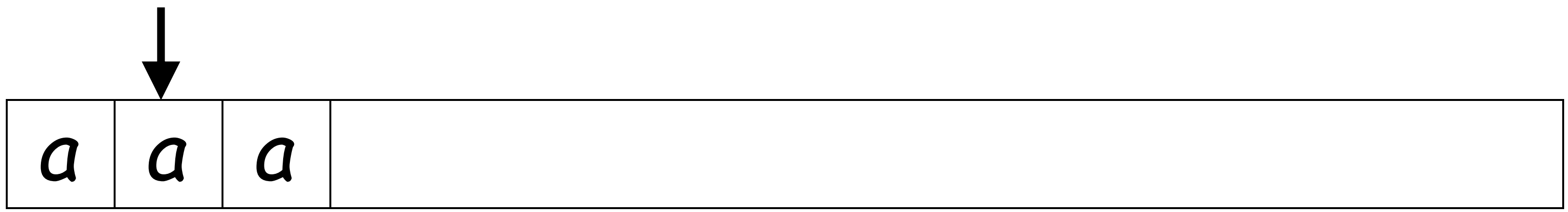


First Choice



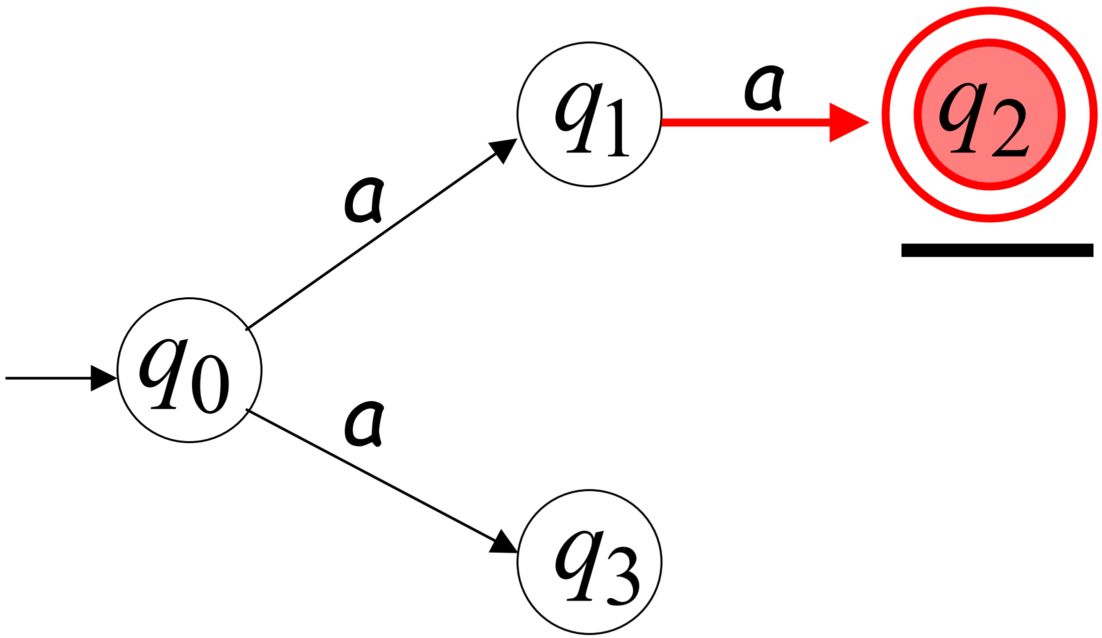
No transition:
the automaton hangs

First Choice



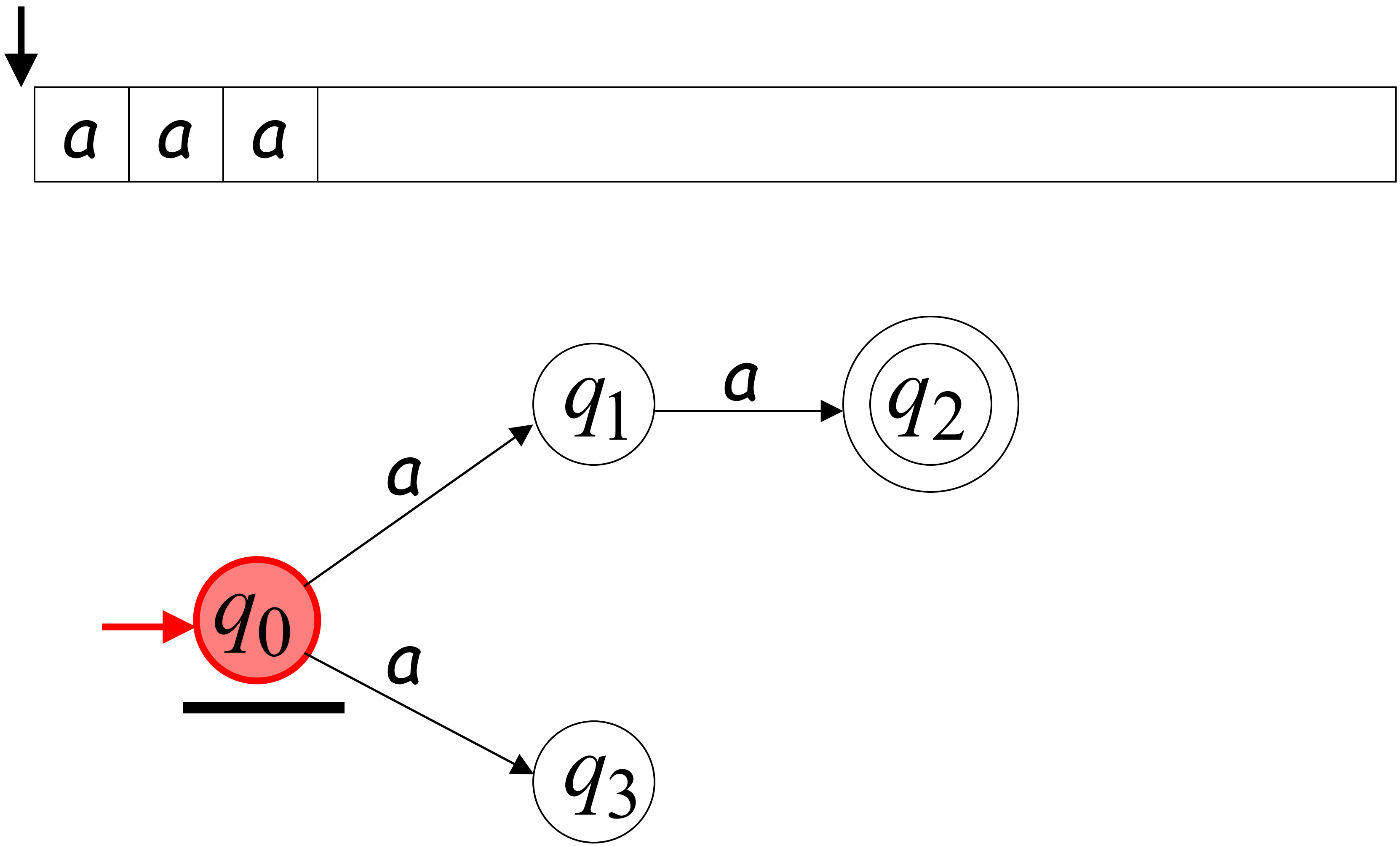
Input cannot be consumed

เพราะ: อ่าน Input ไม่ครบ

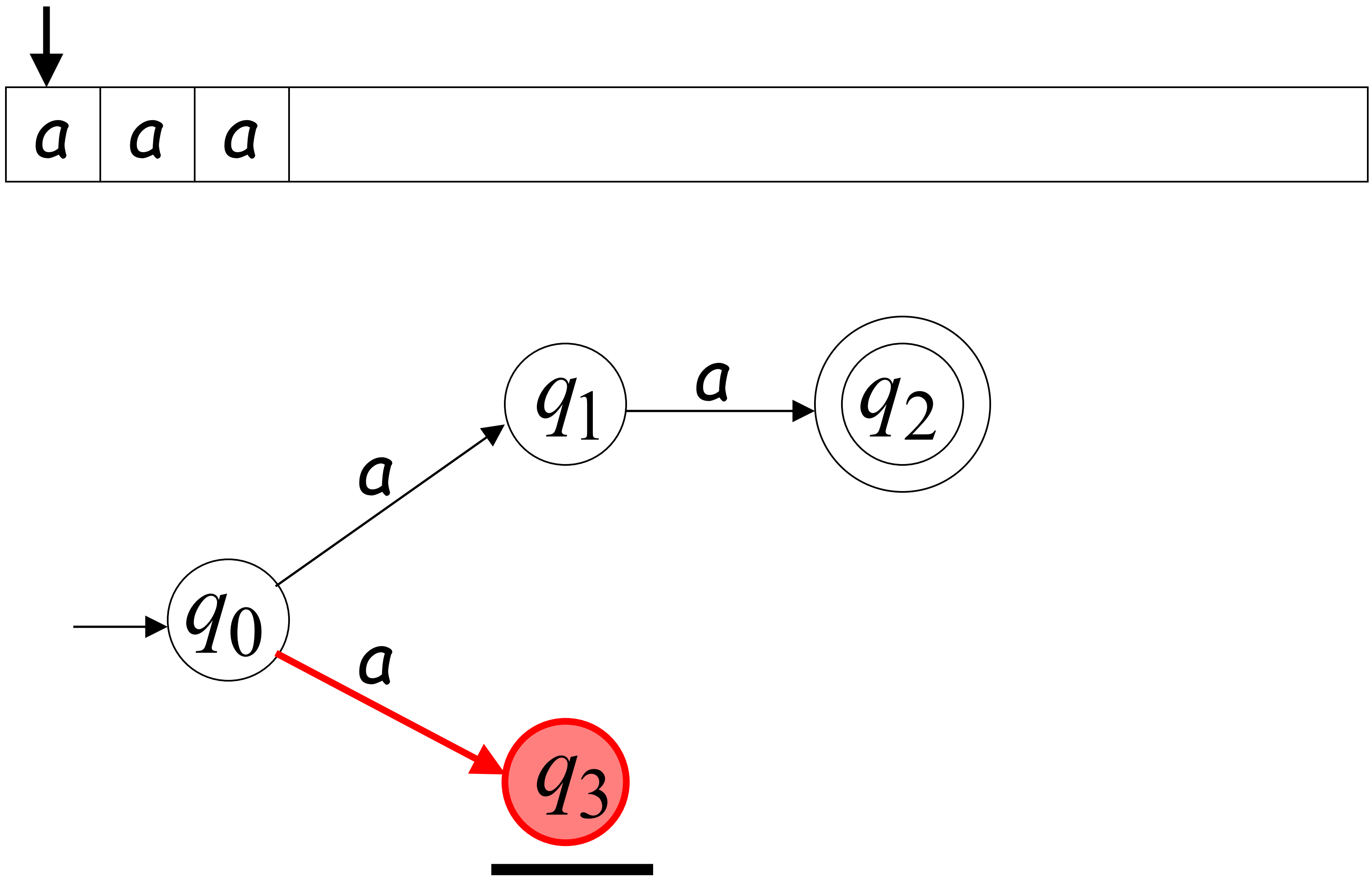


"reject"

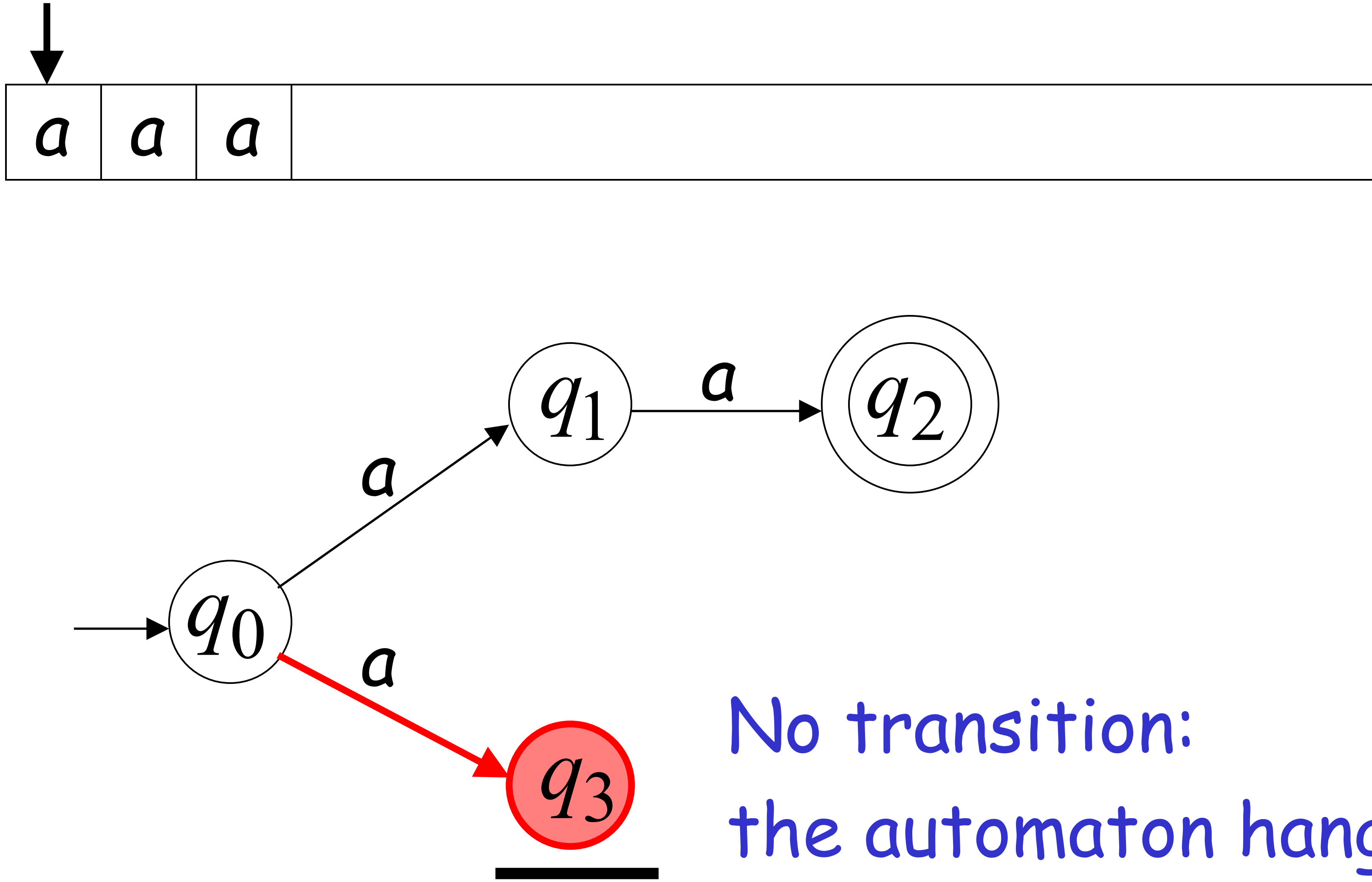
Second Choice



Second Choice

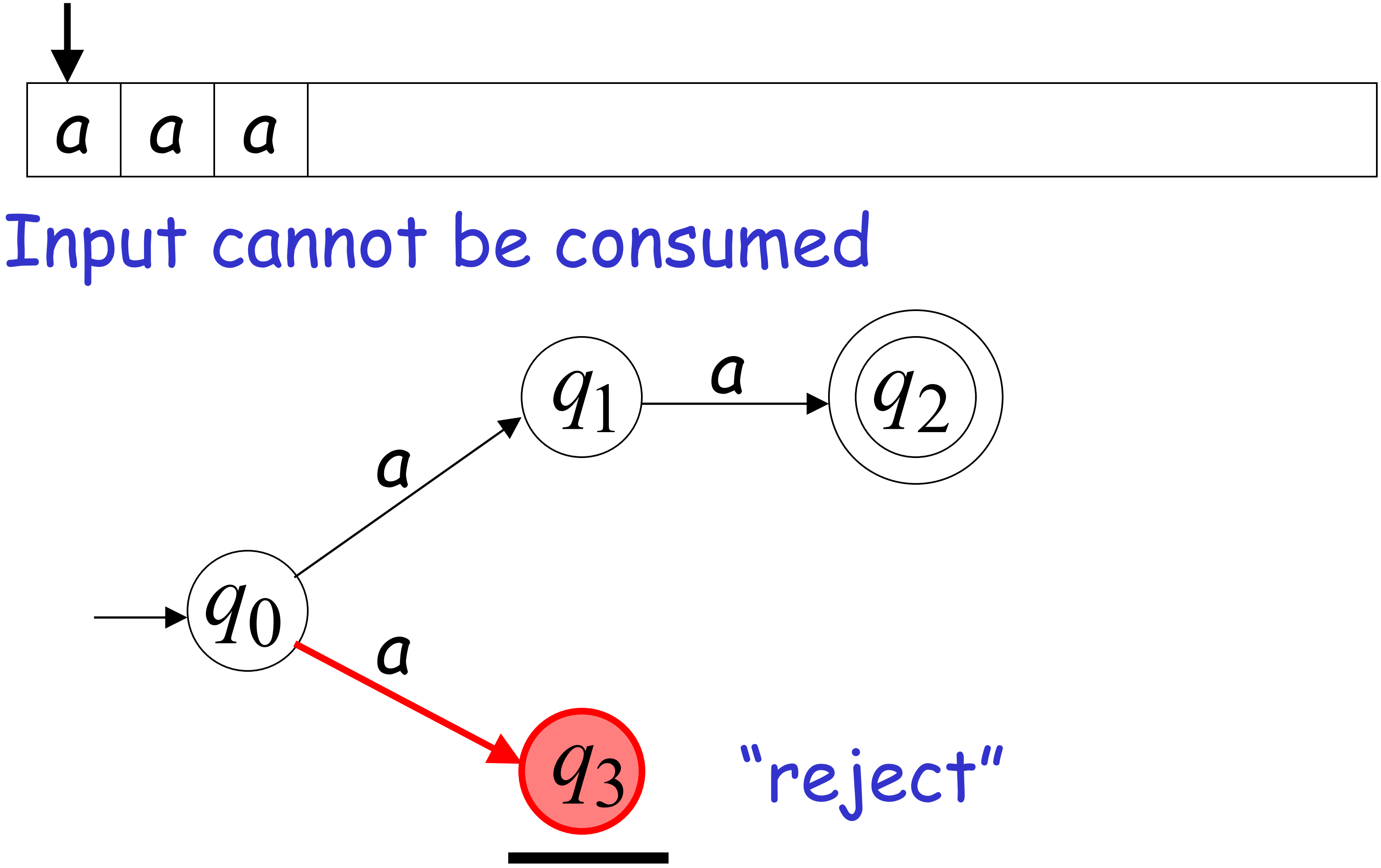


Second Choice



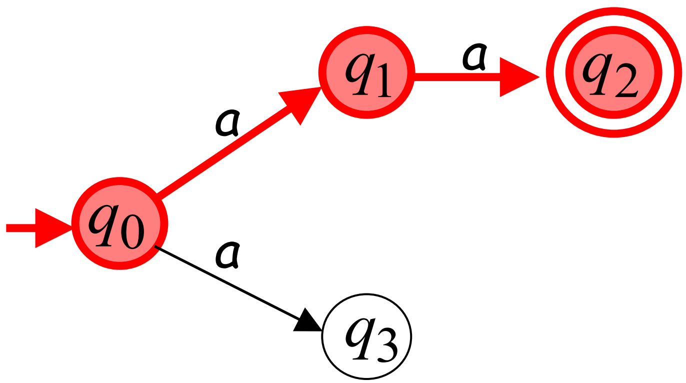
No transition:
the automaton hangs

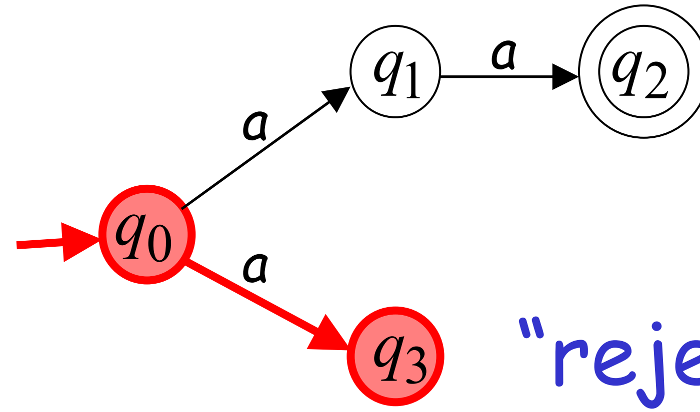
Second Choice



aaa is rejected by the NFA:

"reject"

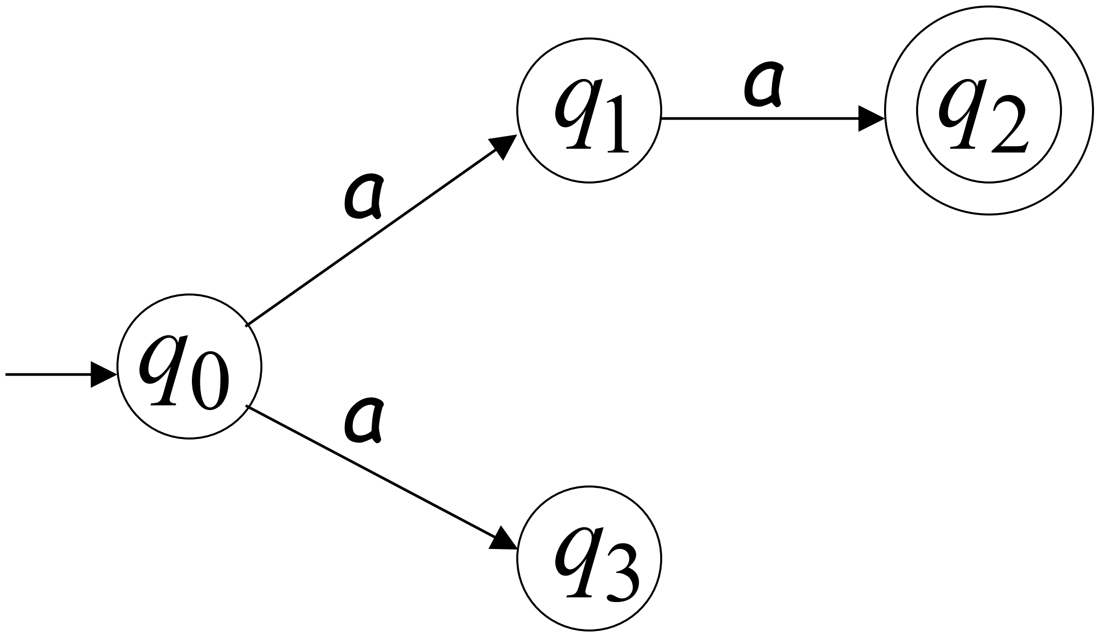




"reject"

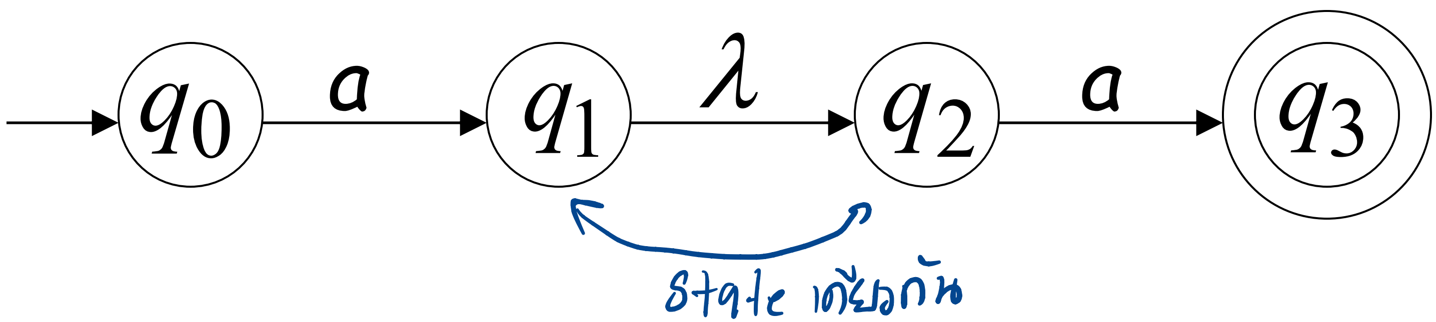
All possible computations lead to rejection

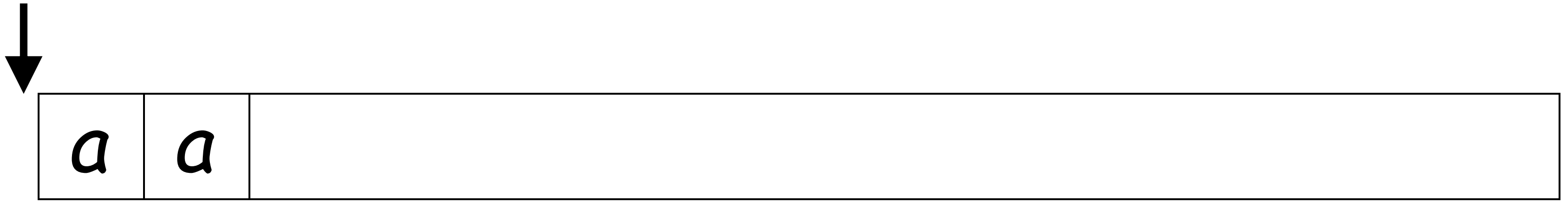
Language accepted: $L = \{aa\}$



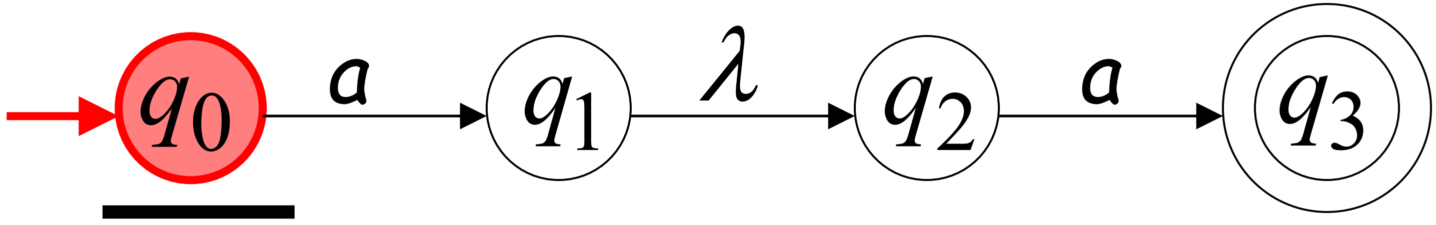
Lambda Transitions

empty string

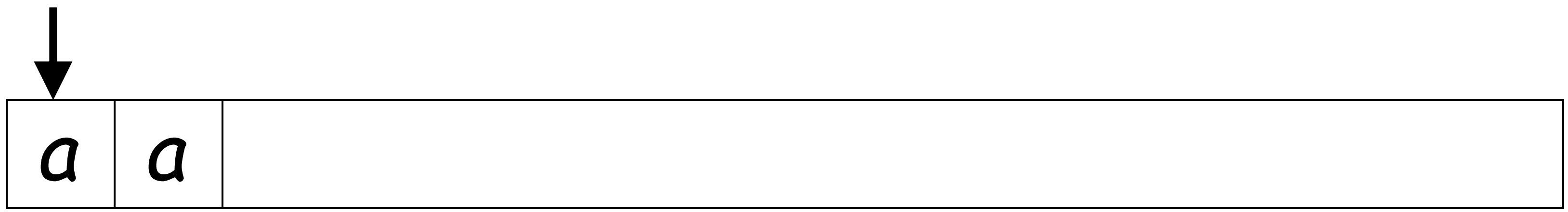


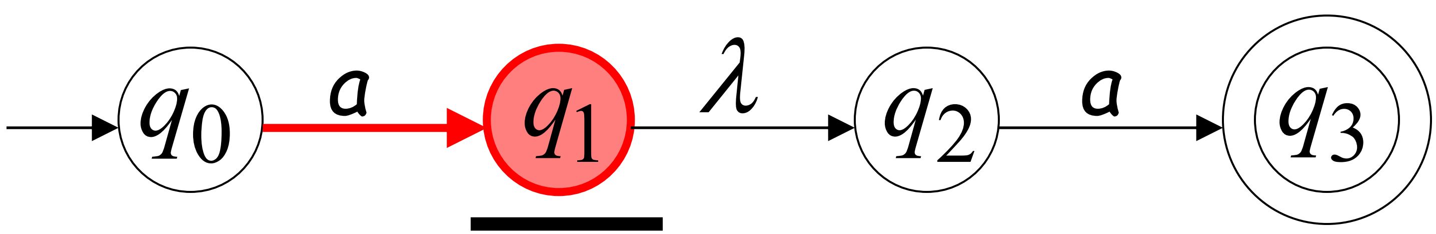


a
a

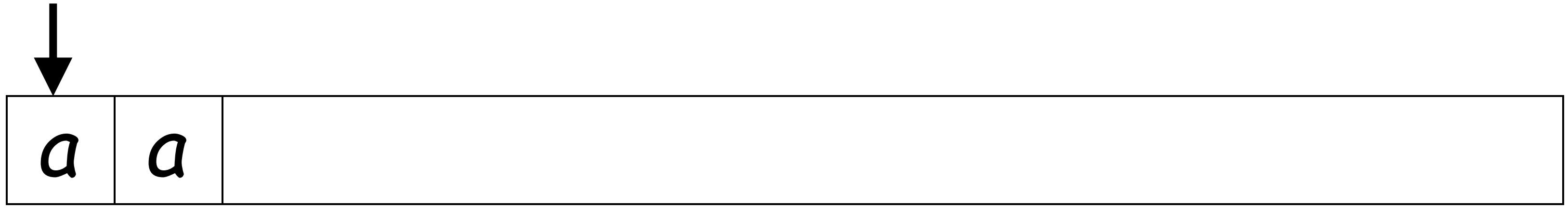


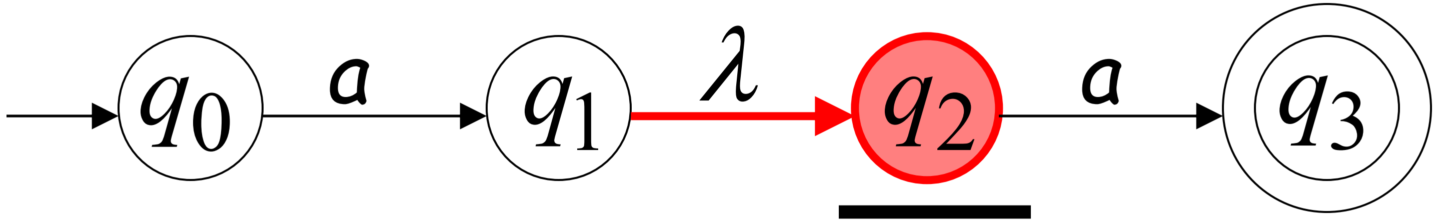
q_0
a
q_1
λ
q_2
a
q_3

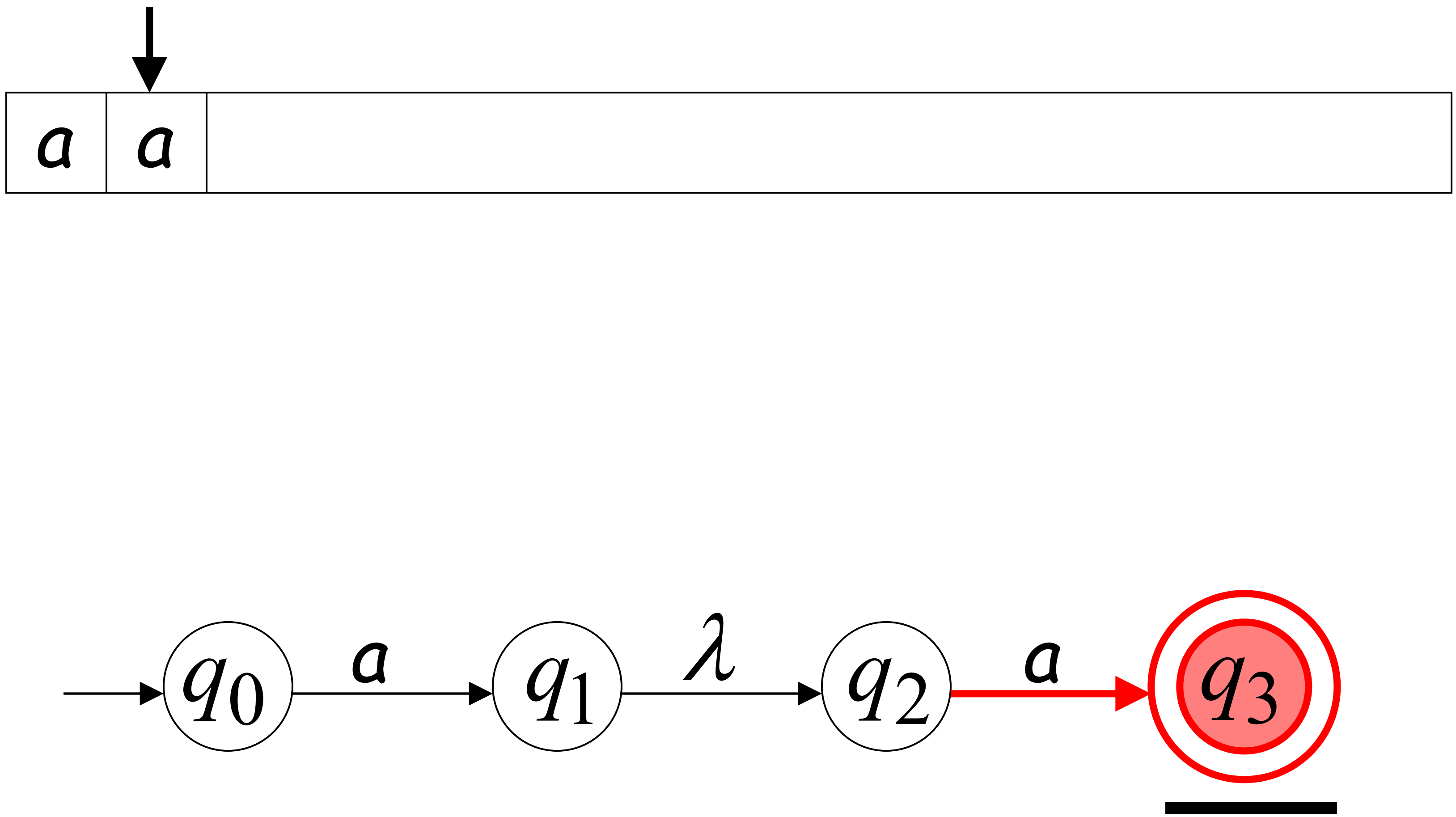

a
a


q_0
a
q_1
λ
q_2
a
q_3

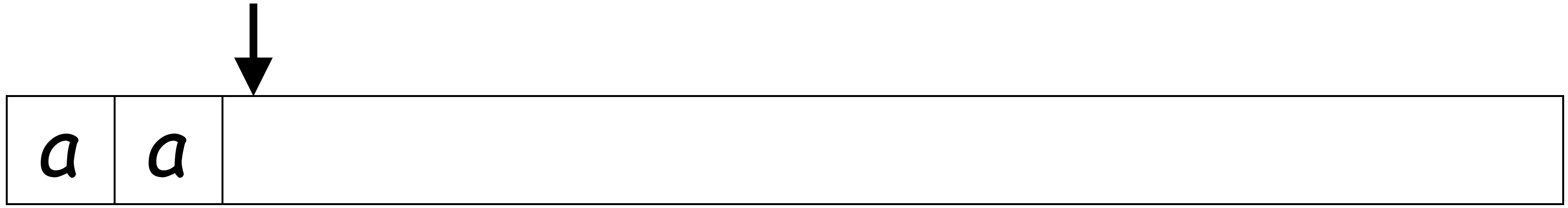
(read head does not move)



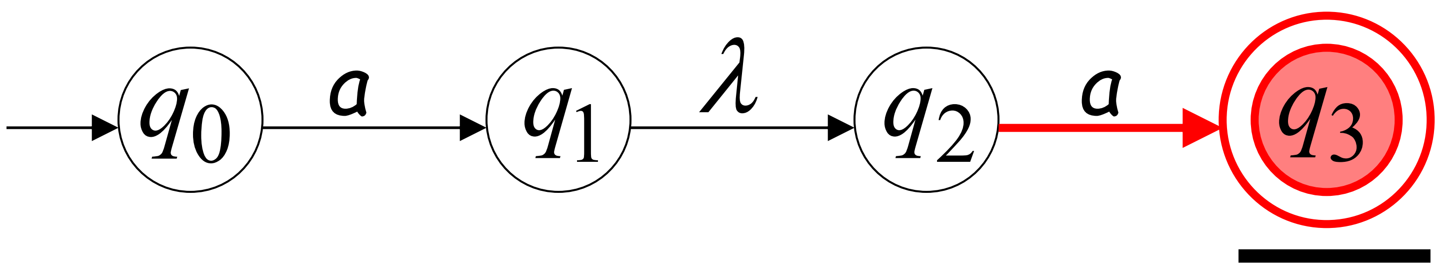



a
a
a
λ
a
q_0
q_1
q_2
q_3

all input is consumed

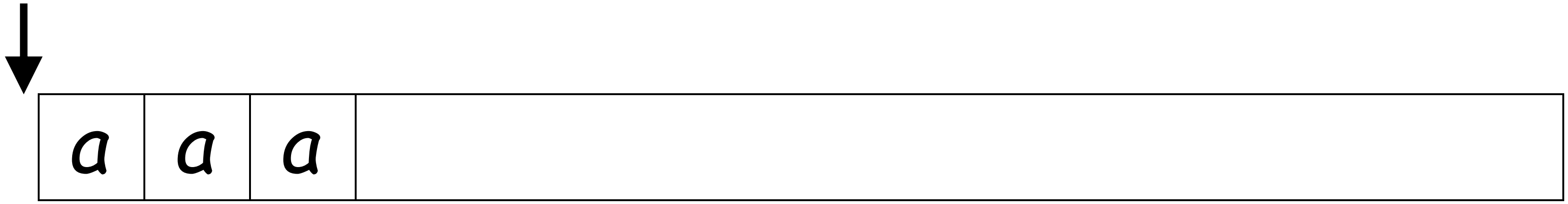


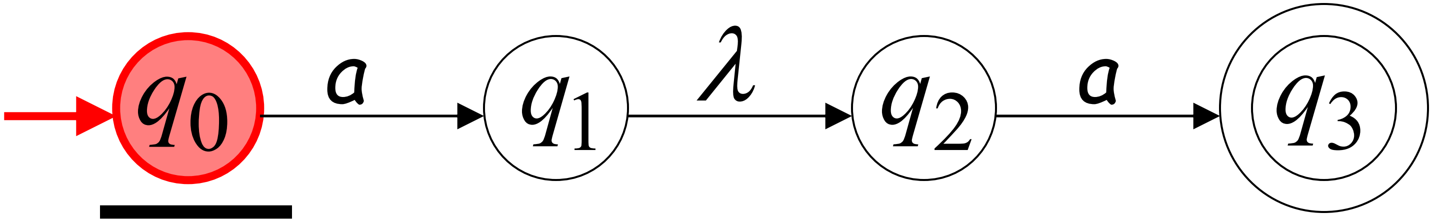
“accept”

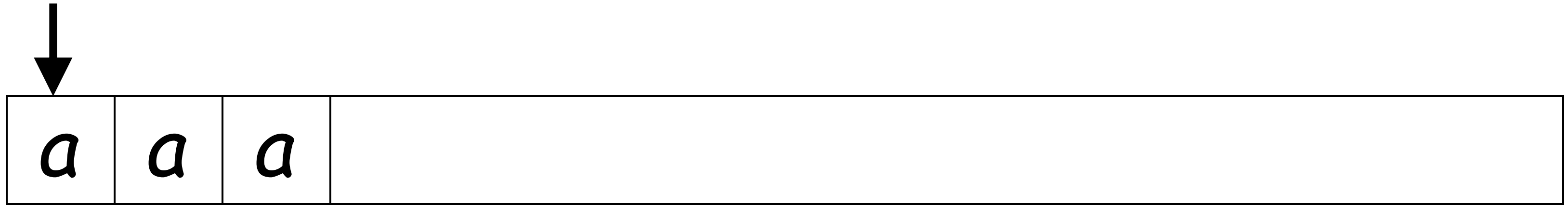


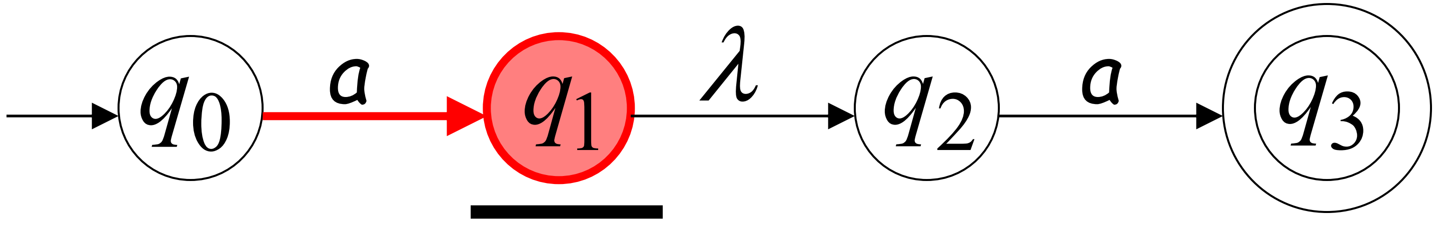
String aa is accepted

Rejection Example

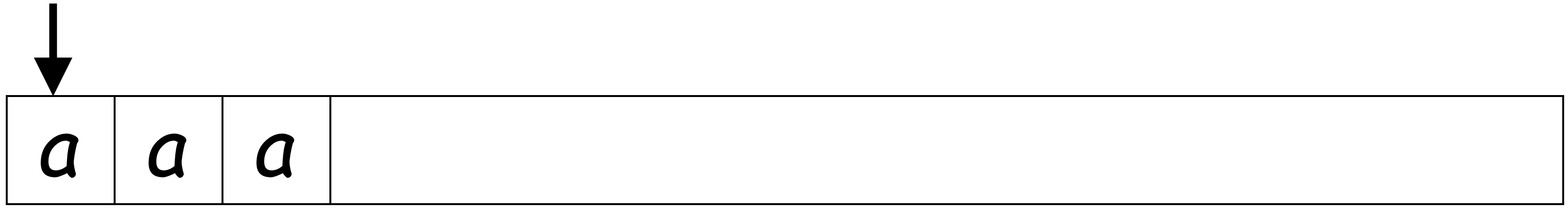


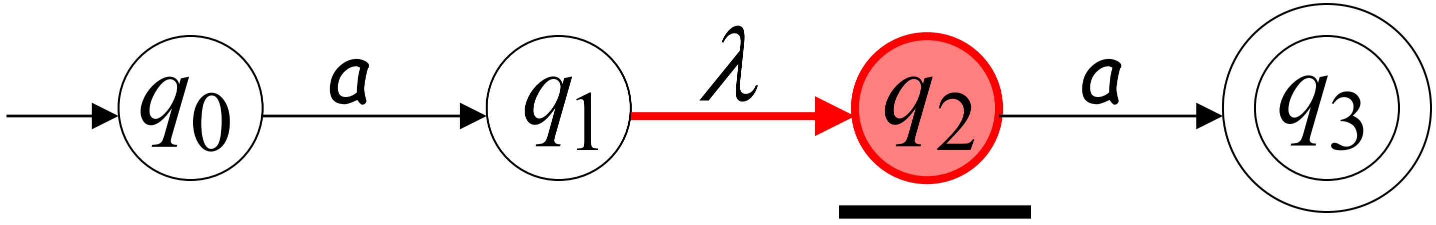


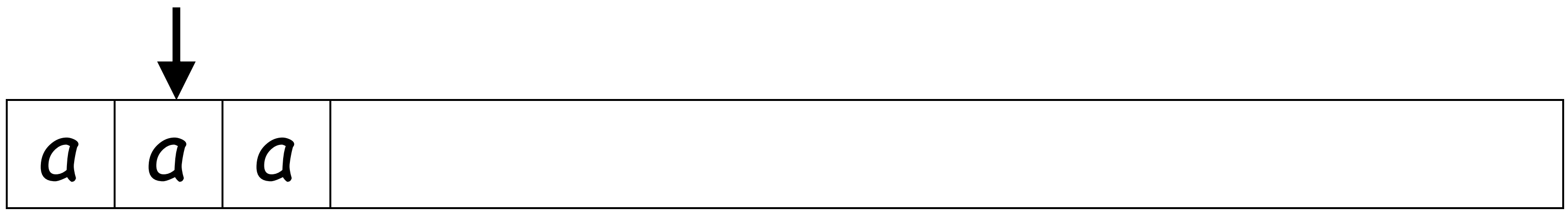

a
a
a

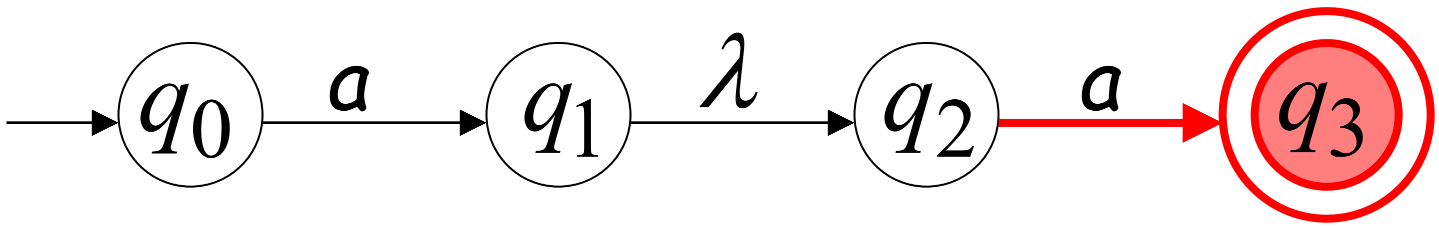

q_0
a
q_1
λ
q_2
a
q_3

(read head doesn't move)





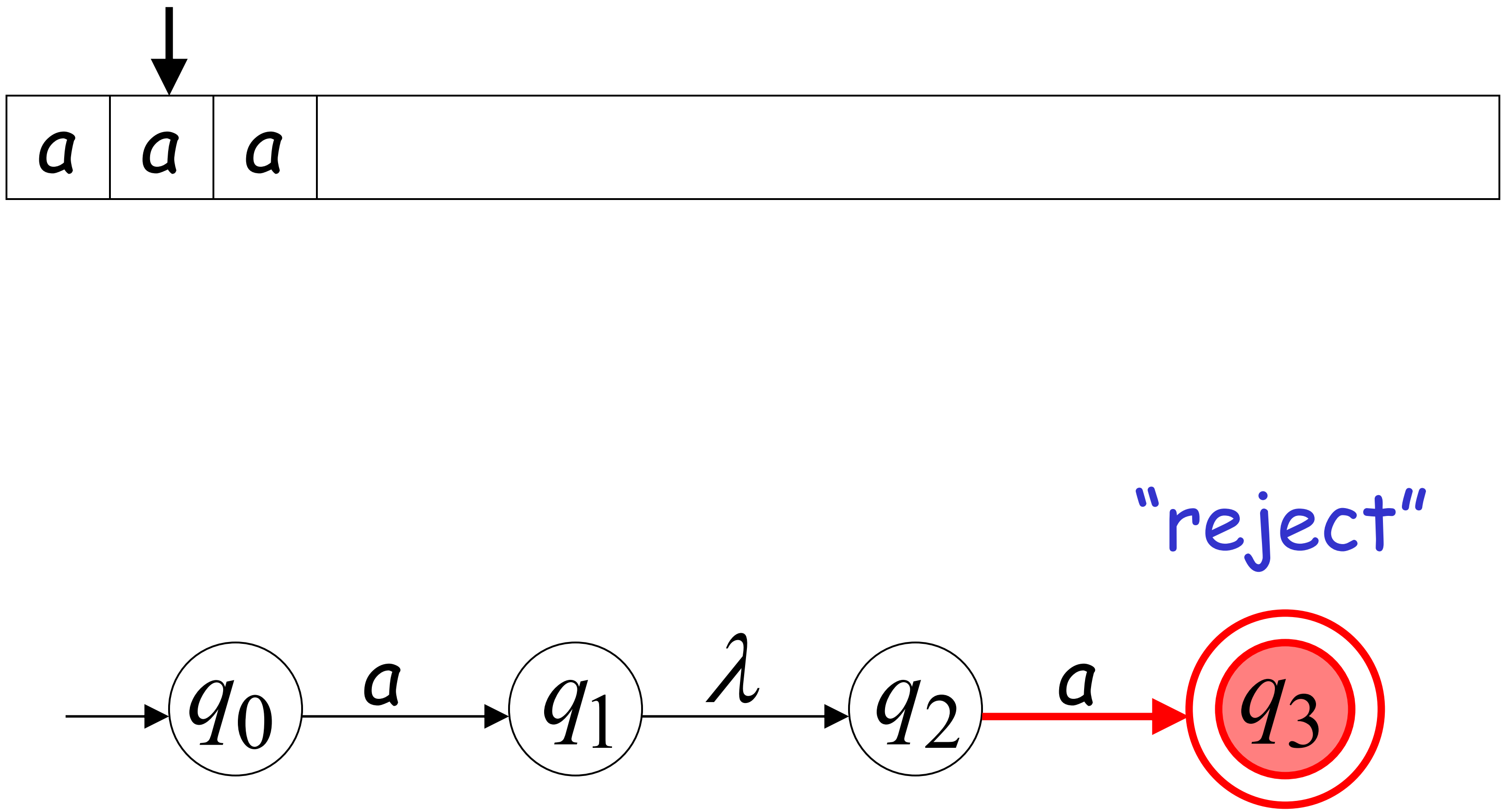




No transition:

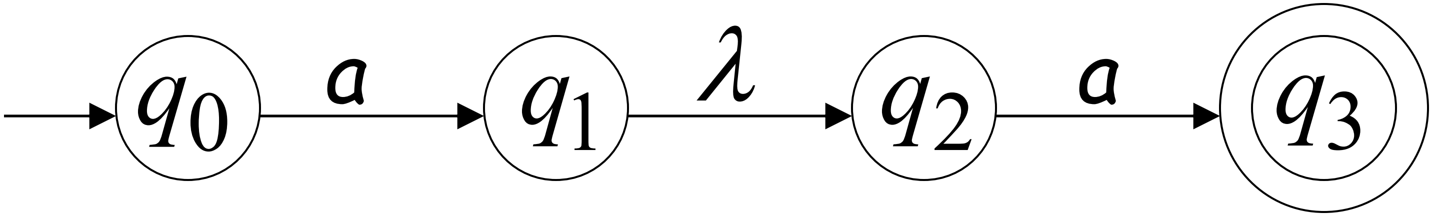
the automaton hangs

Input cannot be consumed

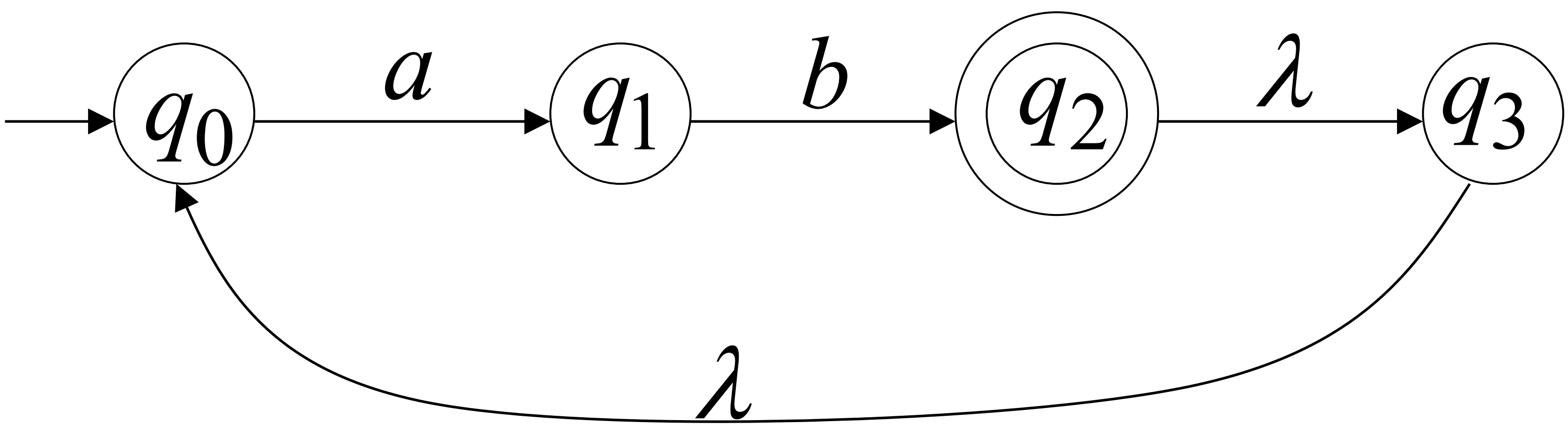


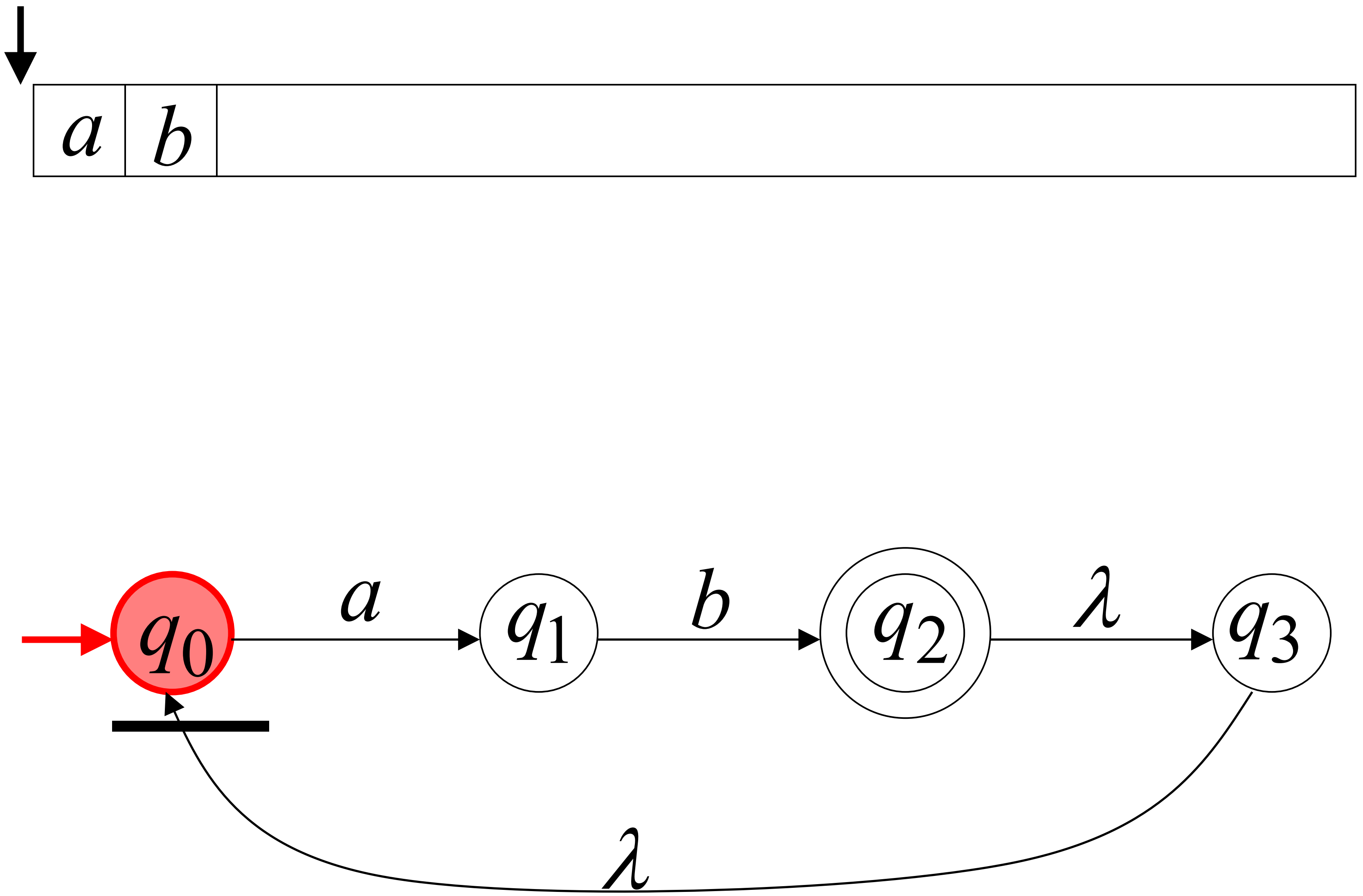
String **aaa** is rejected

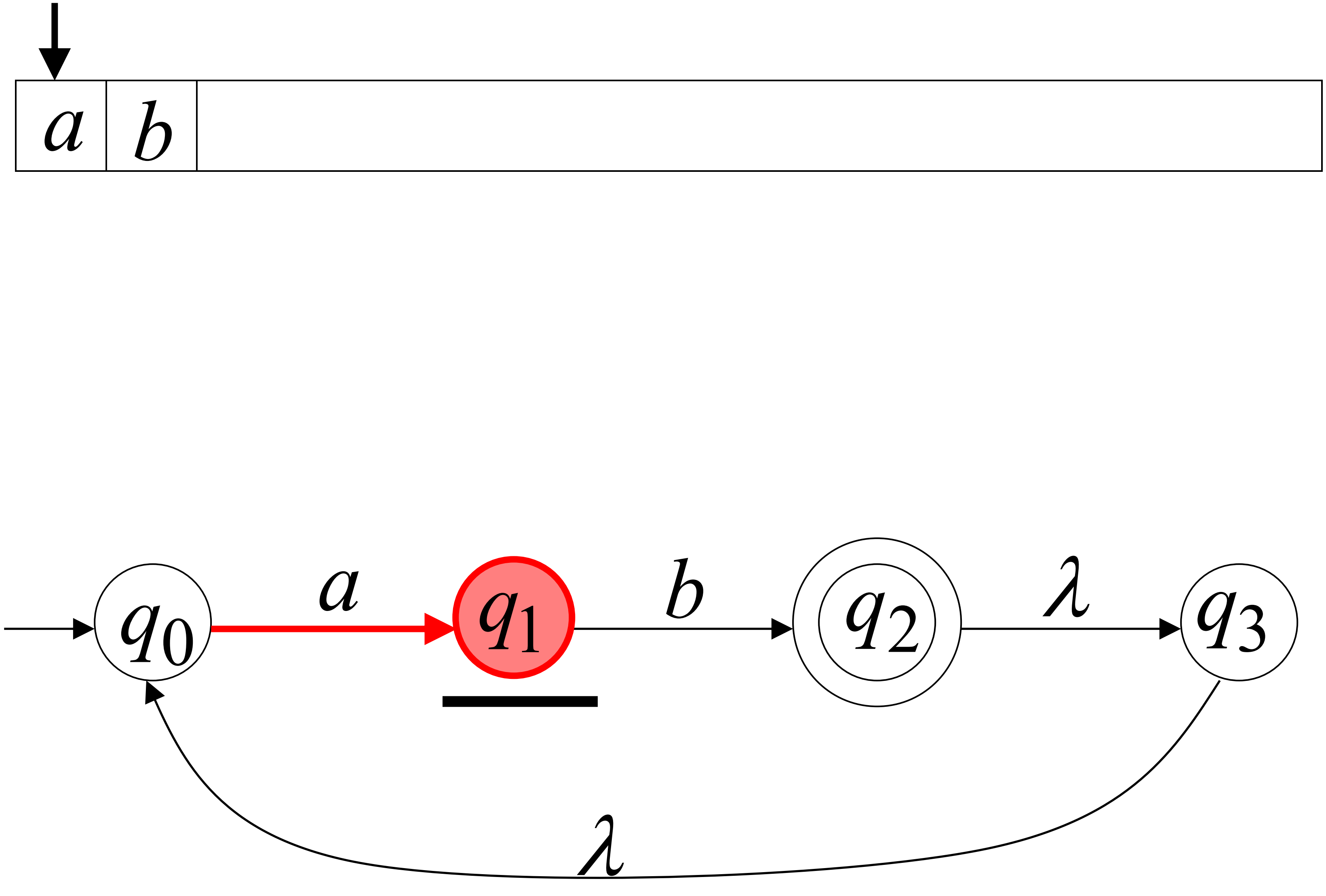
Language accepted: $L = \{aa\}$

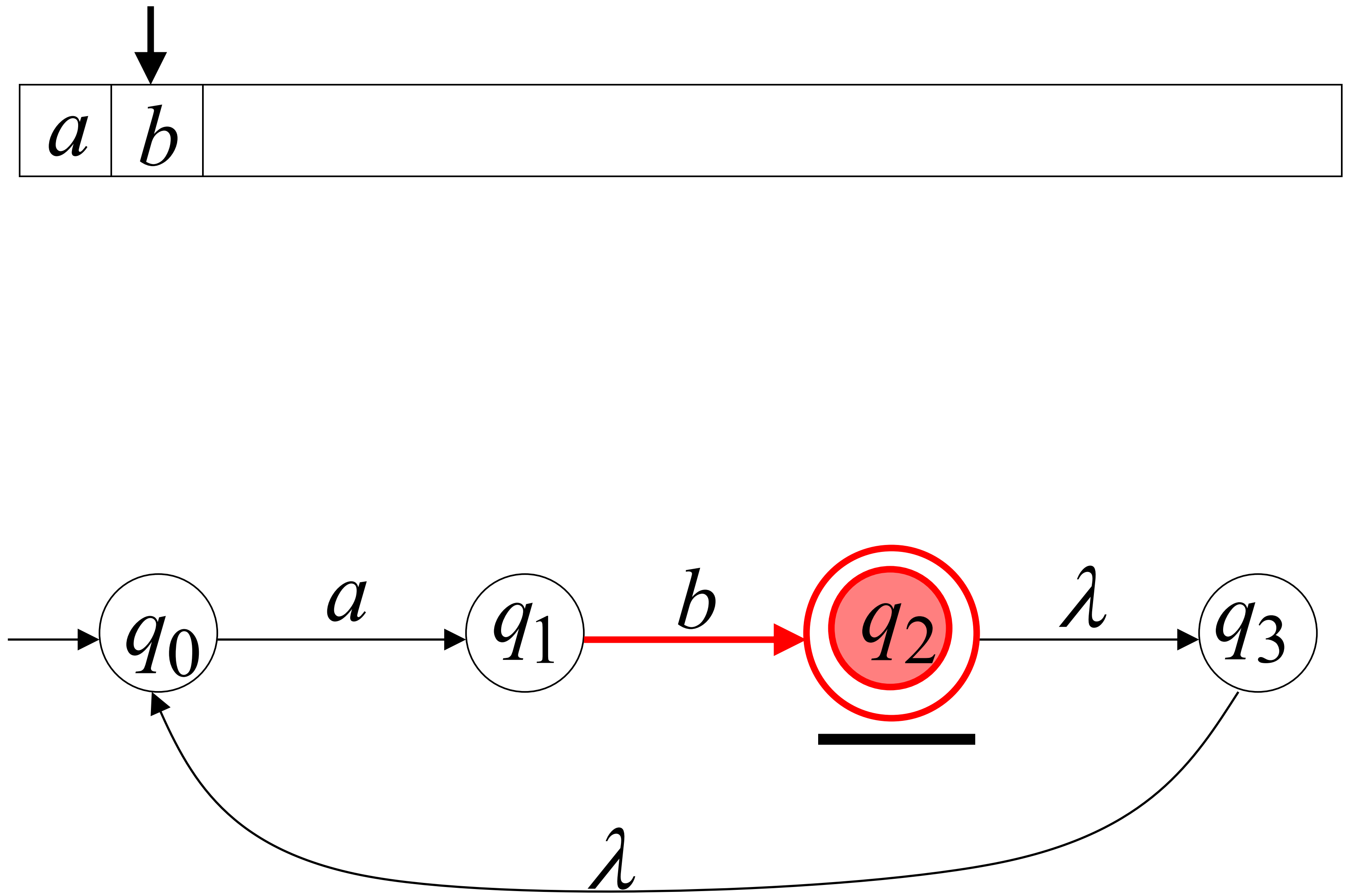


Another NFA Example

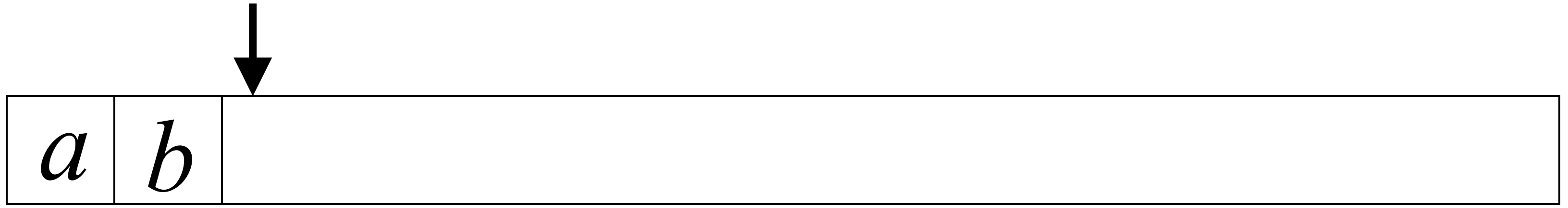



a
b
a
b
λ
λ
q_0
q_1
q_2
q_3

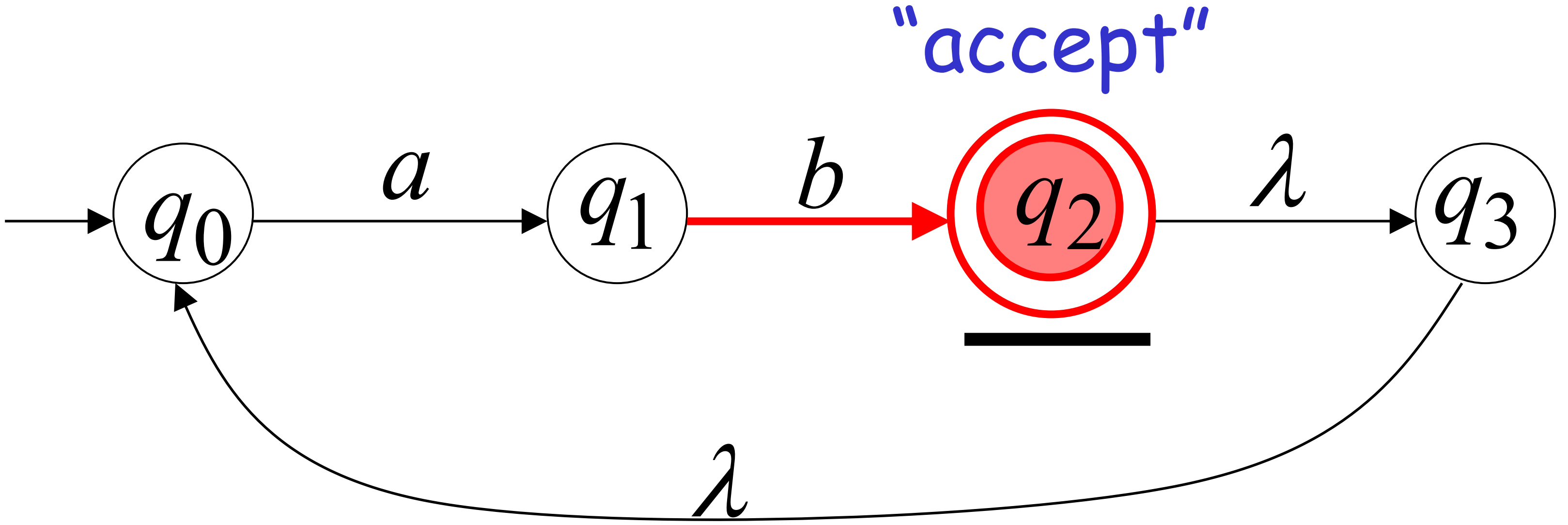

a
b
a
b
λ
λ
q_0
q_1
q_2
q_3



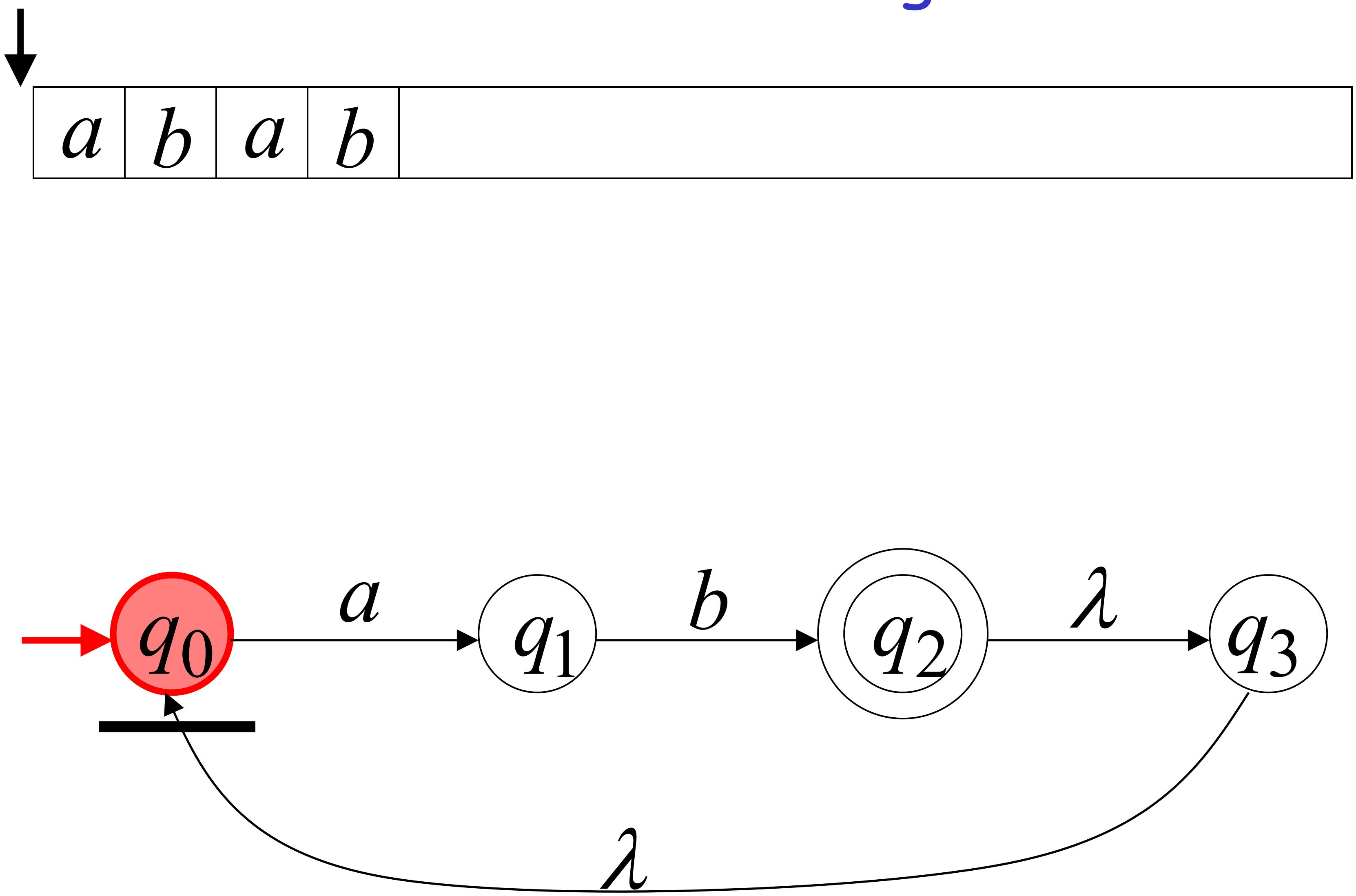
a
b
a
b
λ
λ
q_0
q_1
q_2
q_3

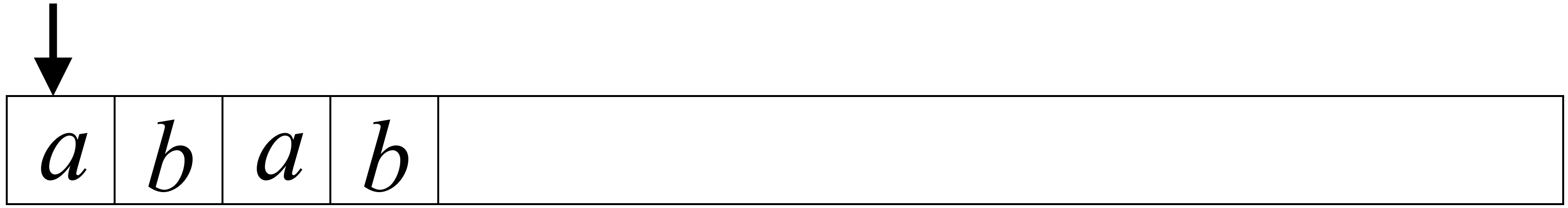


a
b
"accept"
a
b
λ
λ

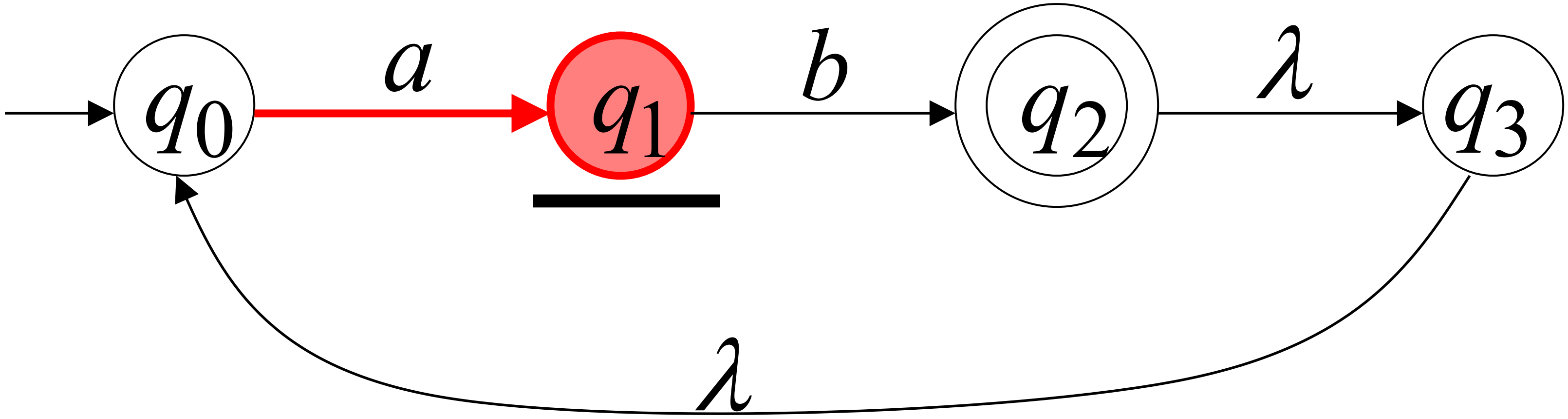


Another String

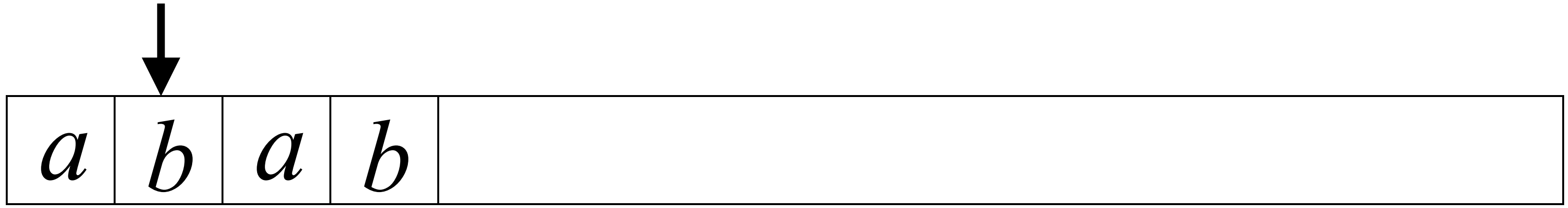


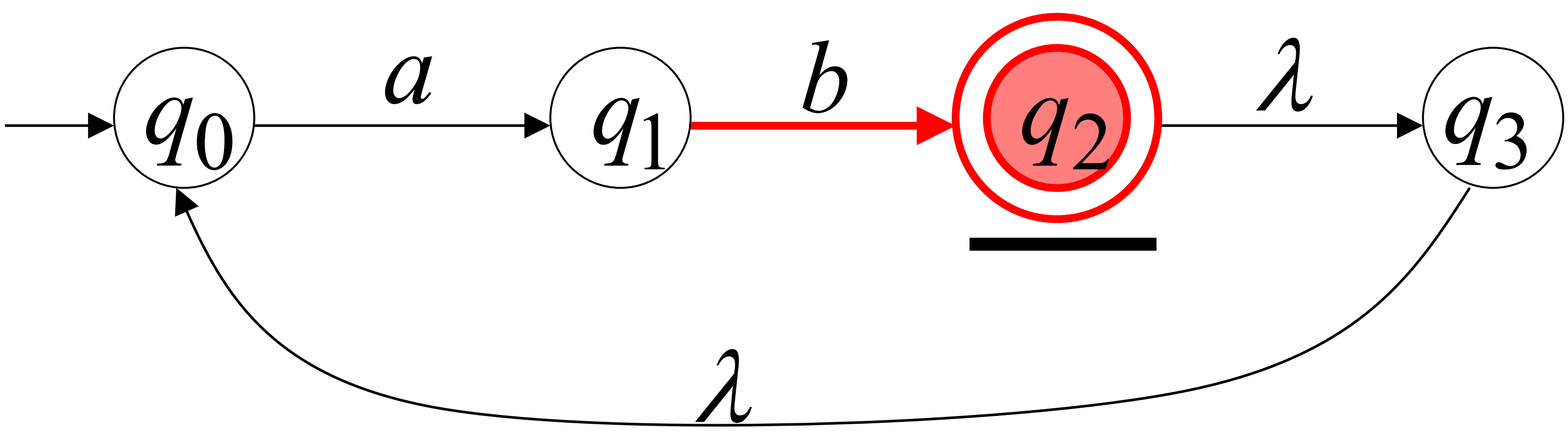


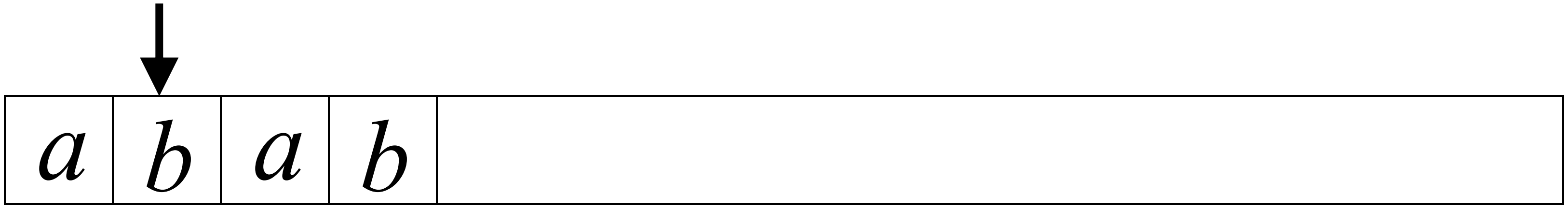
a
b
a
b

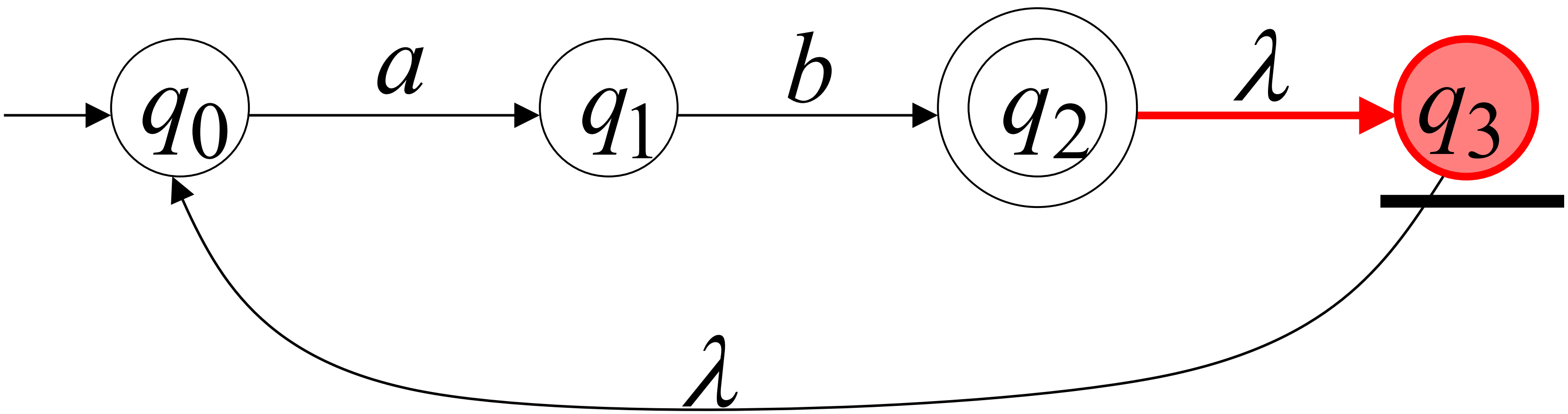


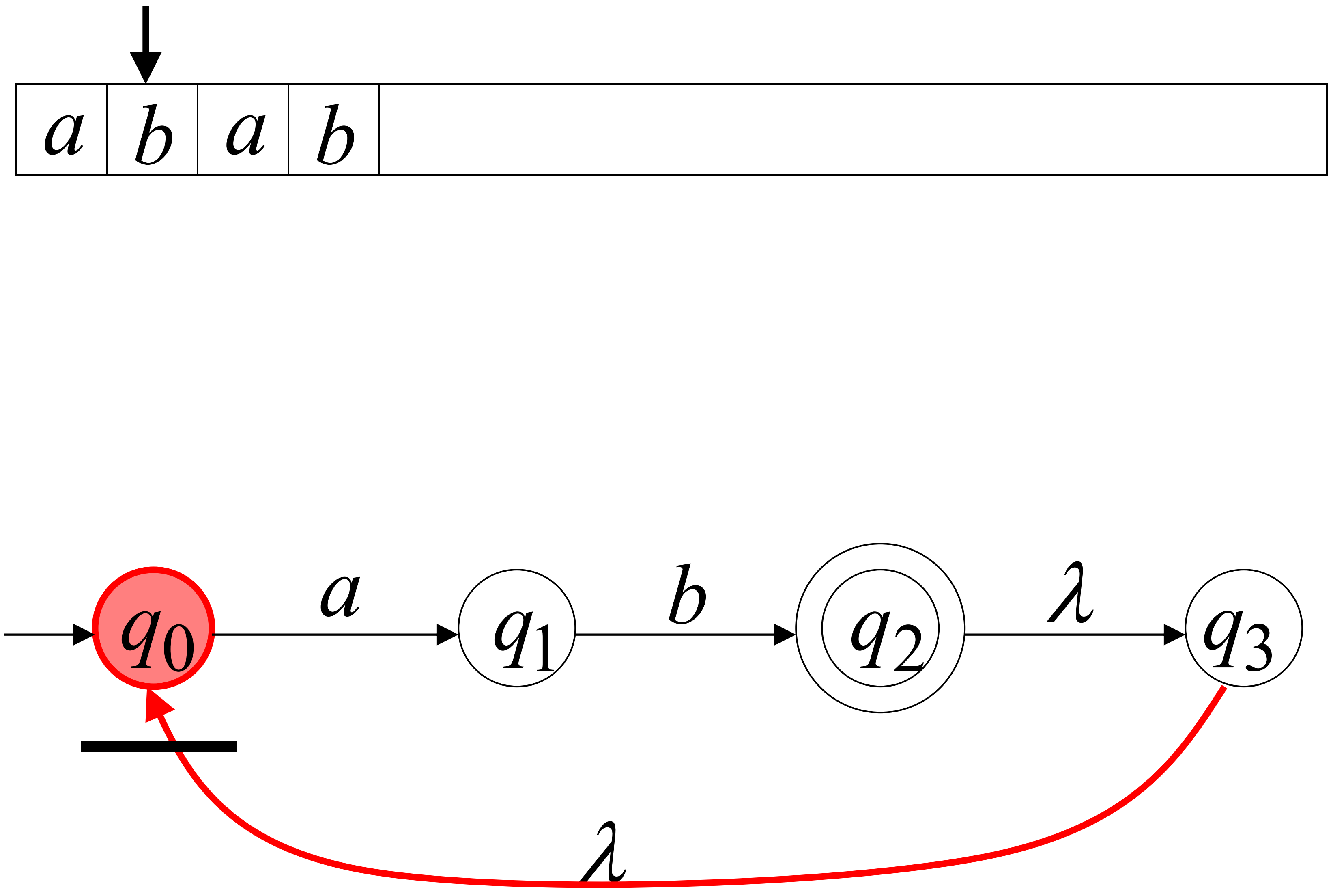
q_0
a
q_1
b
q_2
λ
q_3
λ

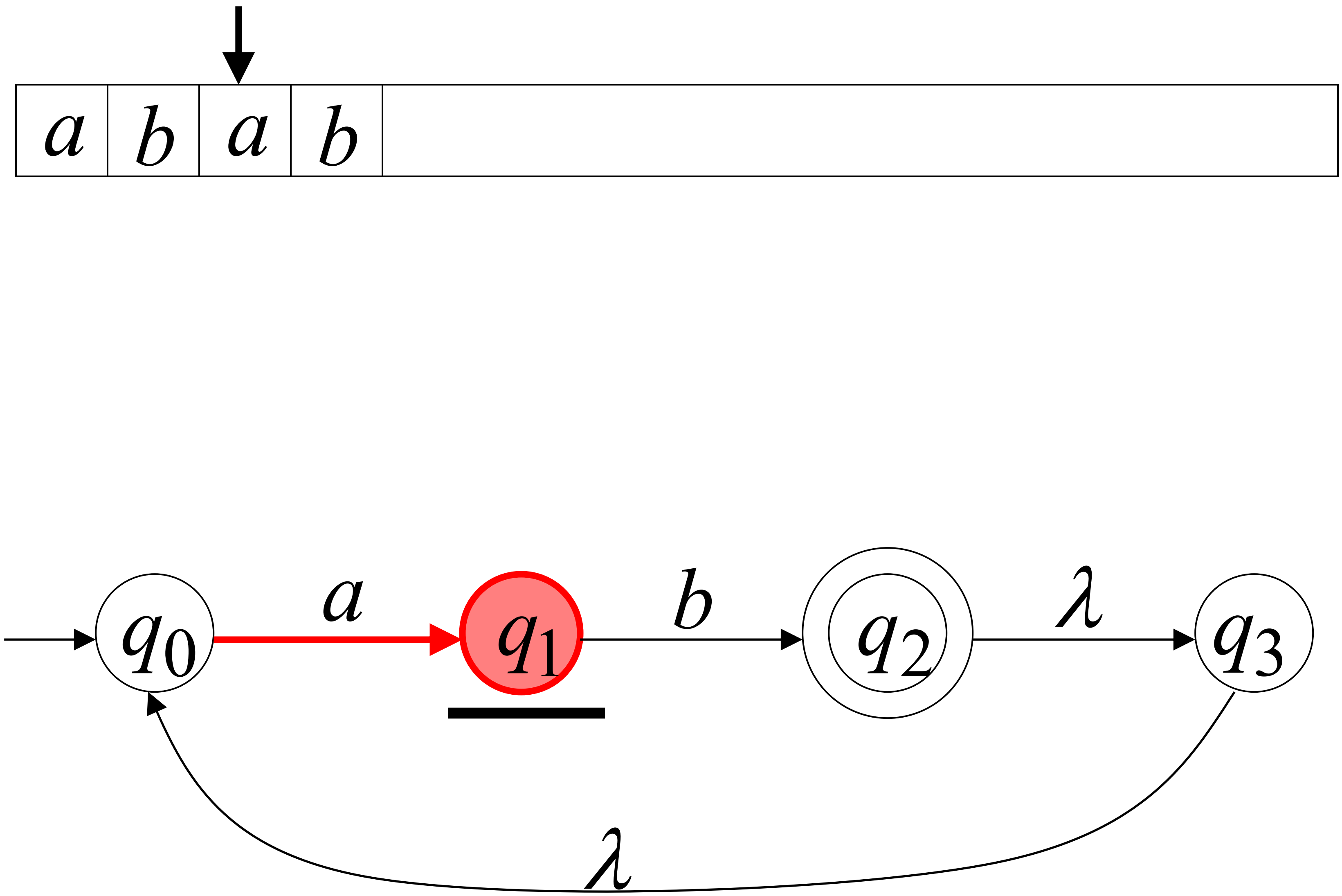

a
b
a
b


q_0
a
q_1
b
q_2
λ
q_3
λ

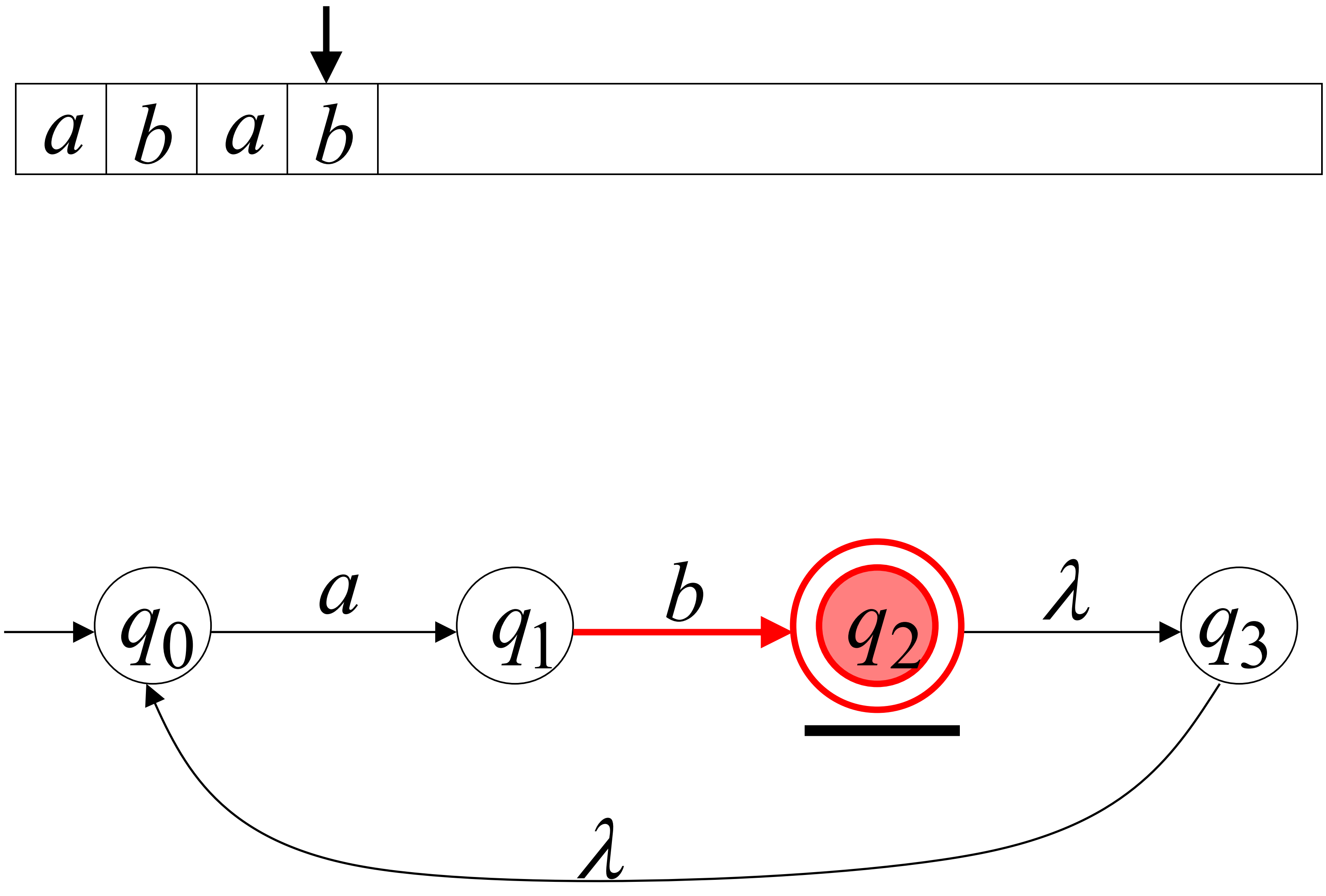

a
b
a
b

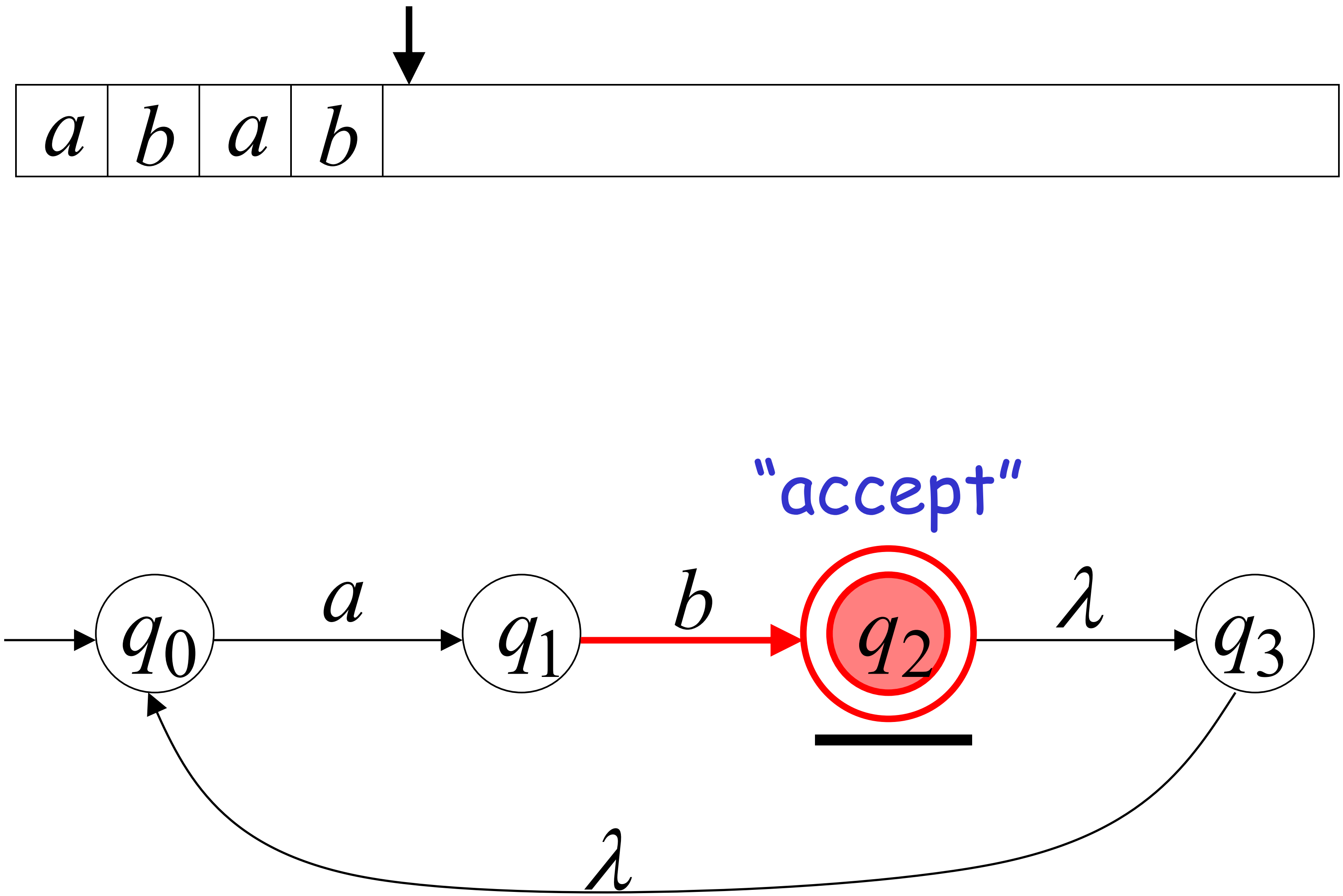

q_0
a
q_1
b
q_2
λ
q_3
λ


a b a b
a
b
λ
λ
q_0
q_1
q_2
q_3



a
b
a
b
q0
a
q1
b
q2
λ
q3
λ


a
b
a
b
q0
a
q1
b
q2
λ
q3
λ

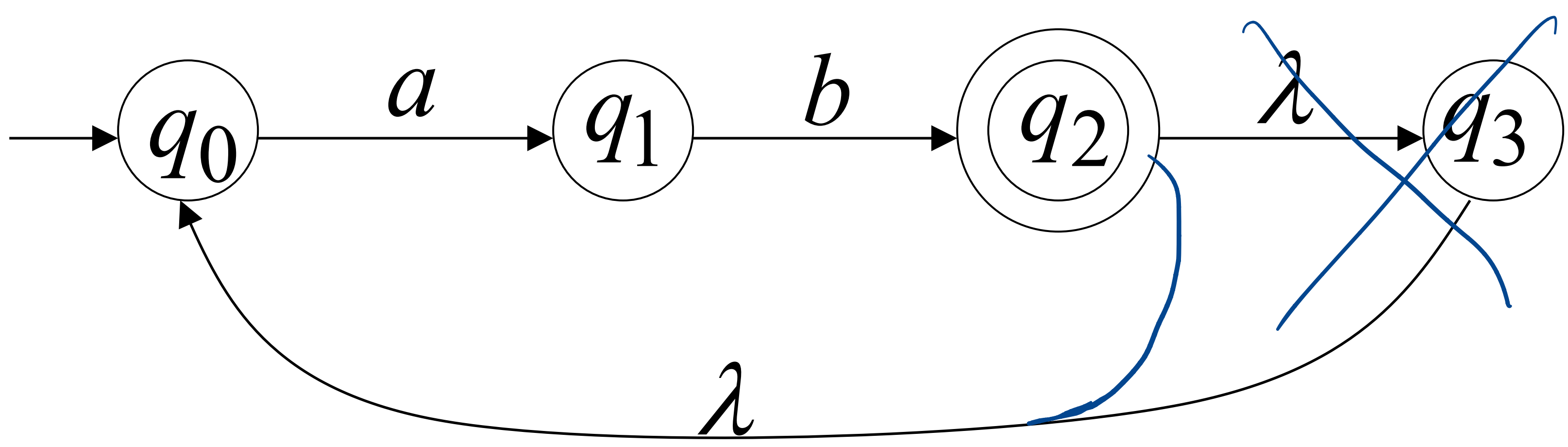


a
b
a
b
"accept"
q_0
q_1
q_2
q_3
a
b
λ
λ

Language accepted

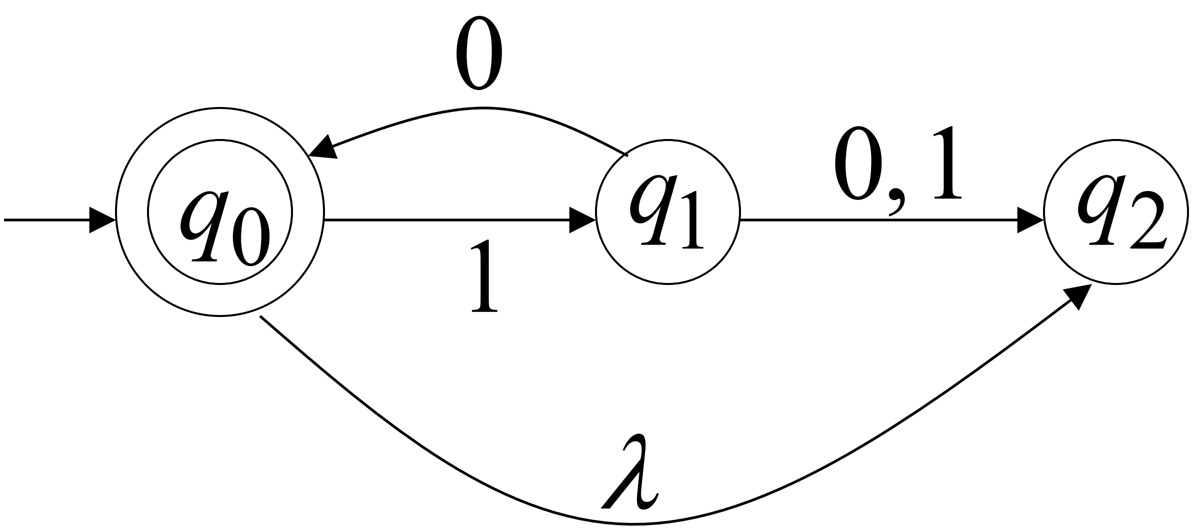
$$L = \{ab,\ abab,\ ababab,\ \ldots\}$$

$$= \{ab\}^{+} \quad \notin \lambda$$



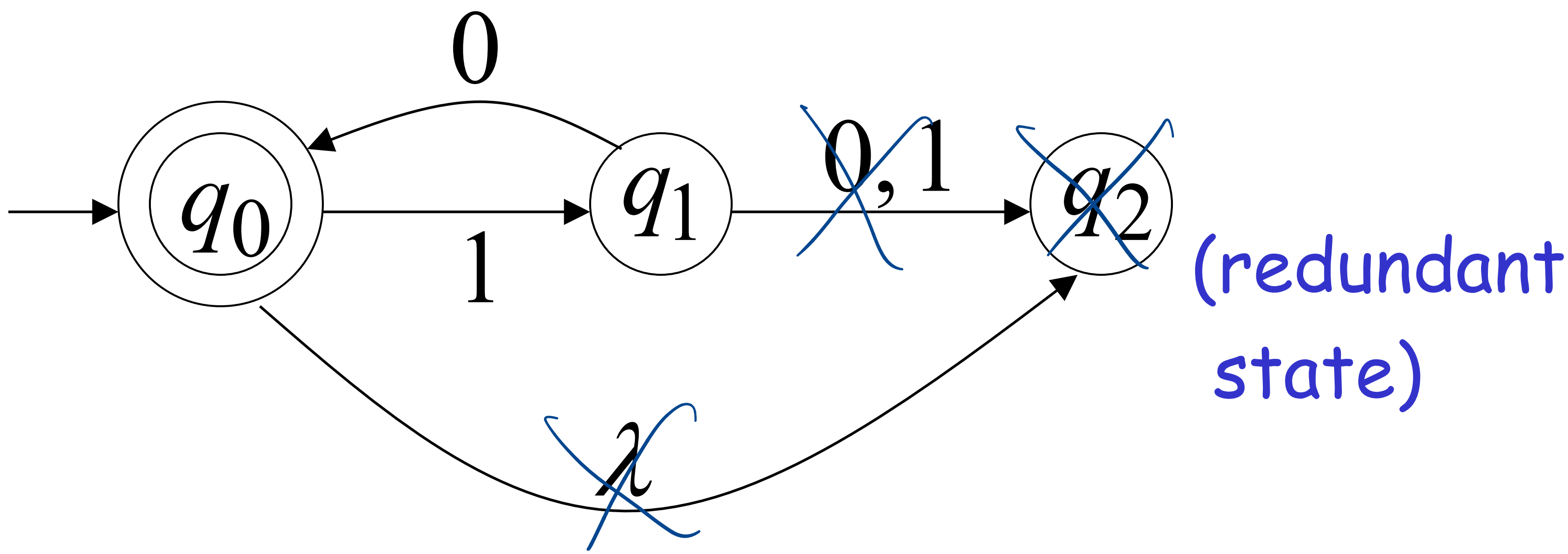
Another NFA Example

$\{\lambda, 10, 1010, 101010, \ldots\}$



Language accepted

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



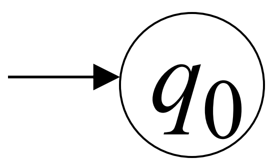
Remarks:

- The λ symbol never appears on the input tape

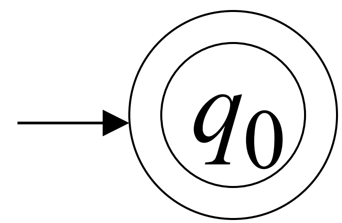
- Simple automata:

รับแค่ string เดียว คือ λ

M_1



M_2



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

ไม่มี string แม้แต่ empty string

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

ภาษาที่มี string นี้

(ตามทัษ = string ว่าง)

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

อธิบายตัวภาษาง่ายกว่า DFAs

แต่รองรับภาษาได้เหมือนกัน

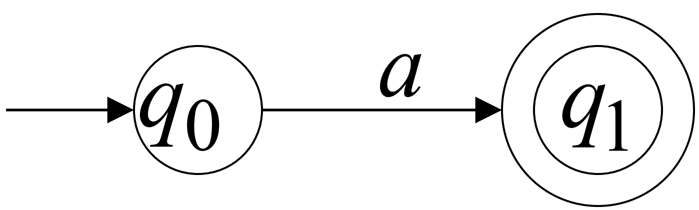
เพราะความสามารถ ทำได้เหมือนกัน

มี trap state ผิดเอง

(trap st.)

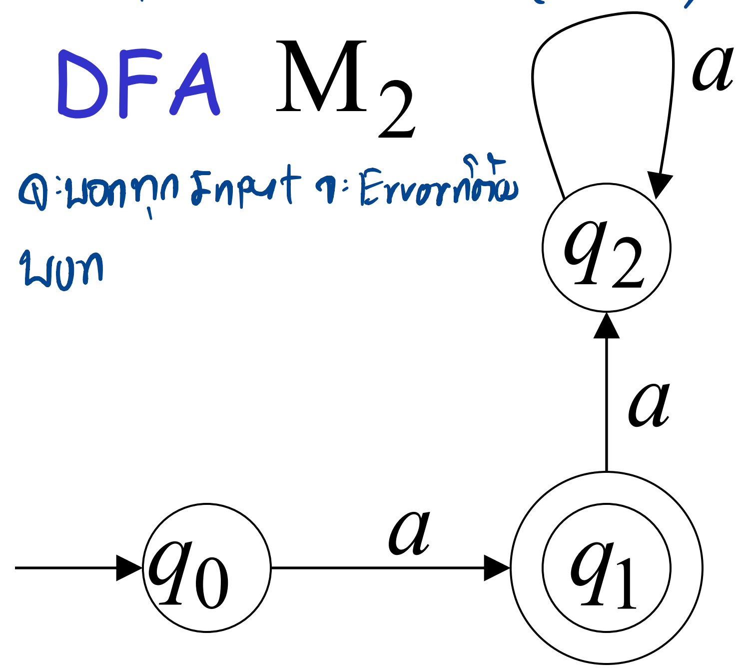
NFA M_1

ถ้า Error จะละไว้ในฐานที่เข้าใจ



DFA M_2

จะบอกทุก Input จะ Error ก็ต้องบอก



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

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

Formal Definition of NFAs

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

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

set ของ

Σ : Input aplhabet, i.e. $\{a, b\}$

δ : Transition function ต่างกับ DFAs

q_0 : Initial state

F : Final states

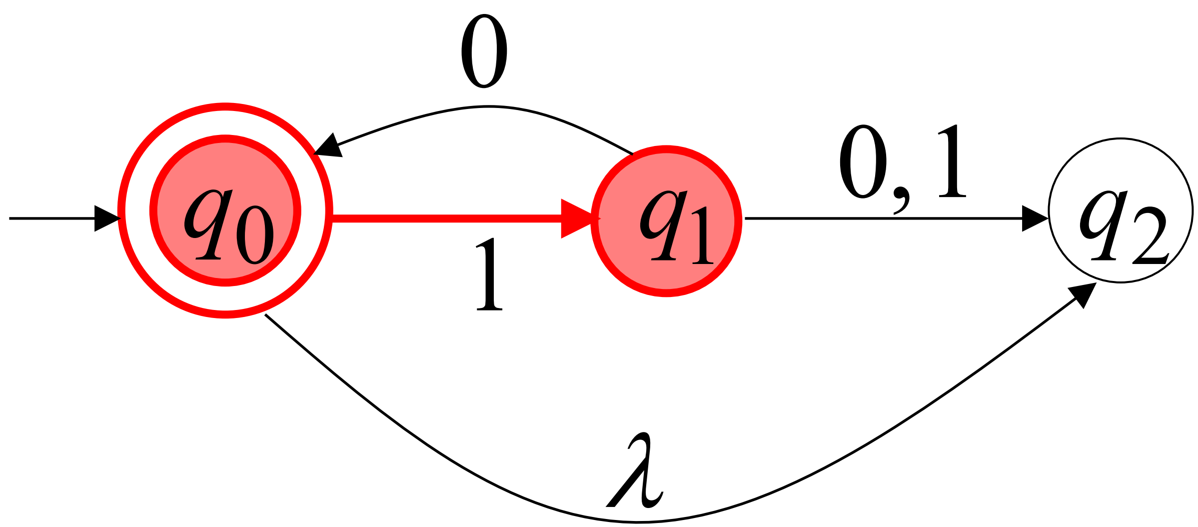
Transition Function δ

DFA $\delta(q_0, 1) = q_1$

input

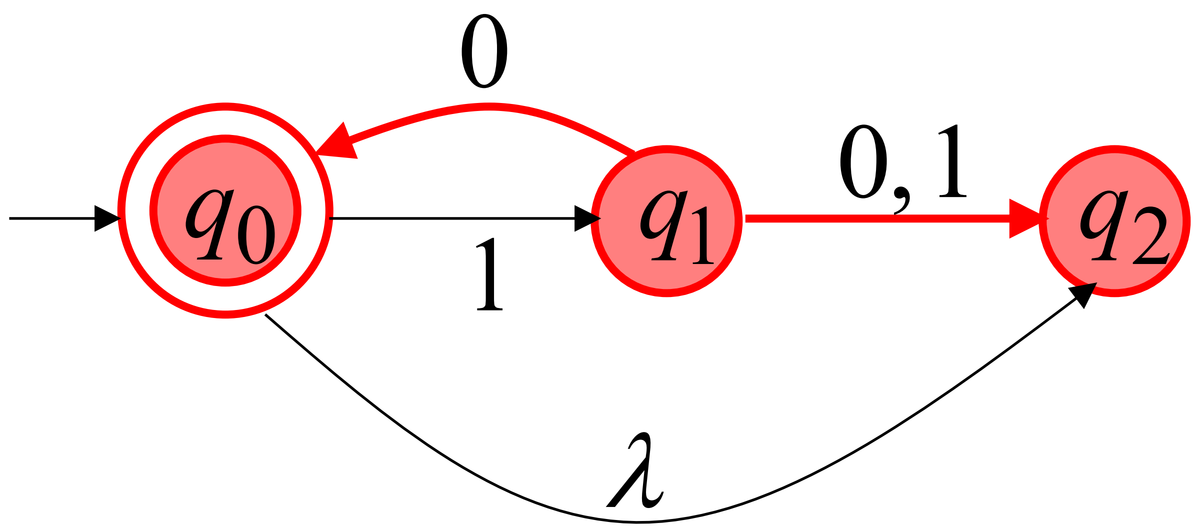
มี set

NFA $\delta(q_0, 1) = \{q_1\}$ เป็น set ของ state



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

ไปได้มากกว่า 1 state

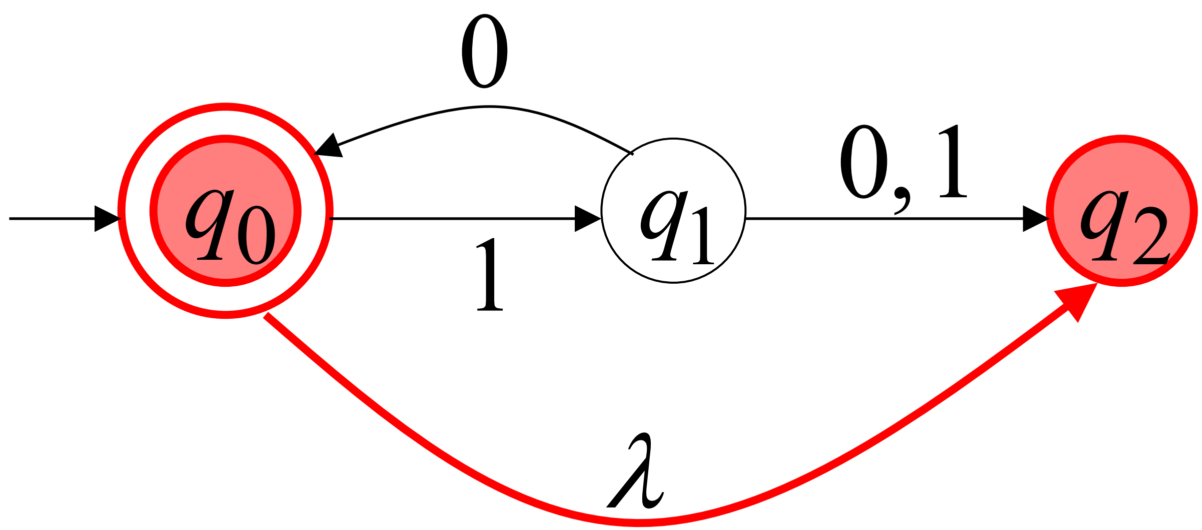


DFA ไม่อนุญาตให้มีค่า input เป็น λ

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

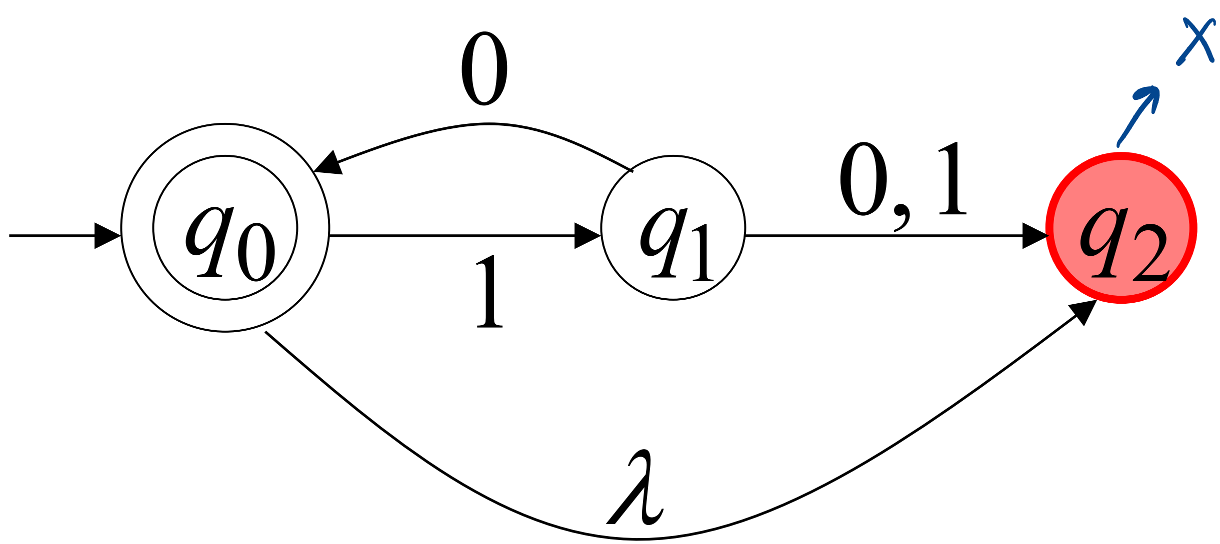
NFA แกจะเลือกอ่านก็ได้ (ไปอยู่ q_2)

ไม่อ่าน (ไปอยู่ q_0)



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

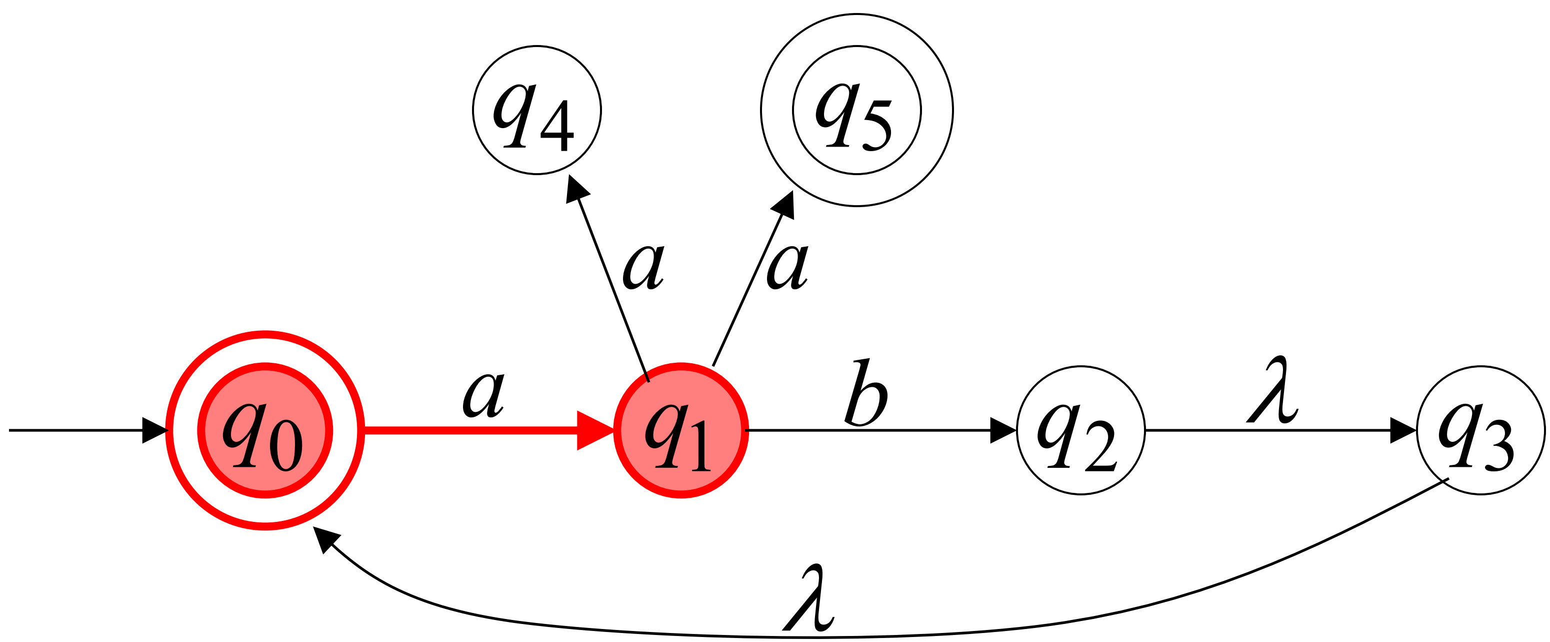
ละไว้ได้พวก Error ต่างๆ



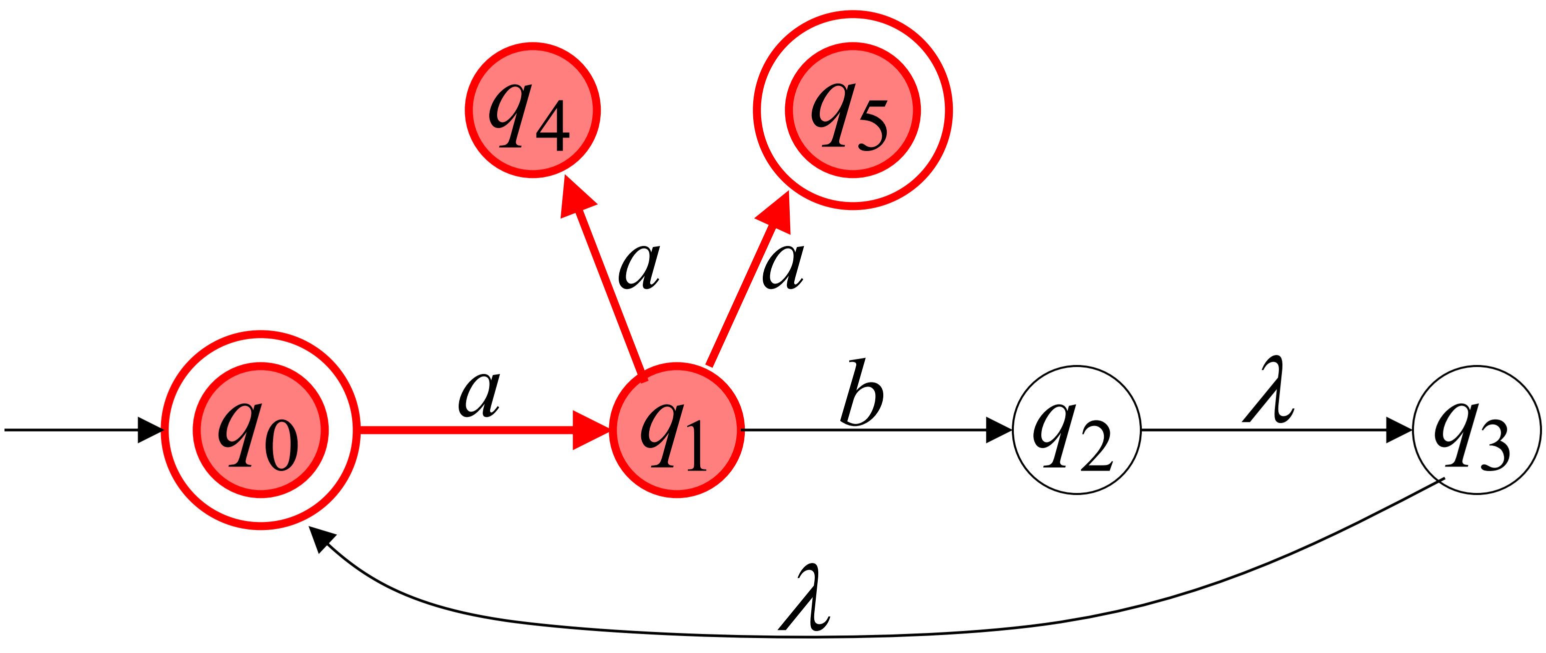
Extended Transition Function δ^*

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

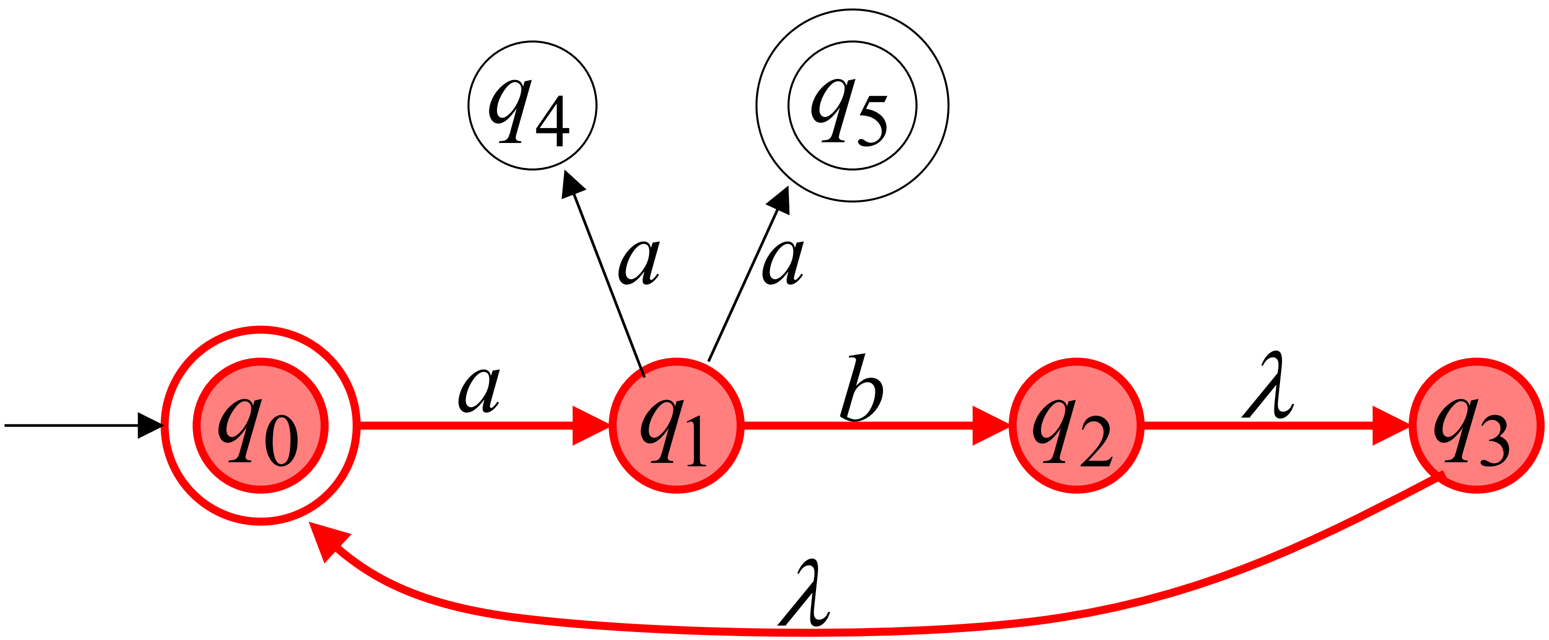
ในนี้มองว่าเป็น string (จริงๆเป็น alphabet)



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



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

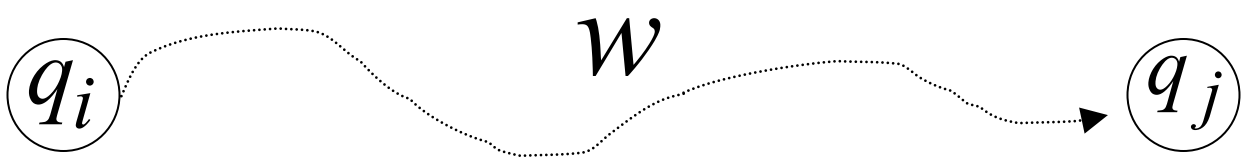


Input

Formally

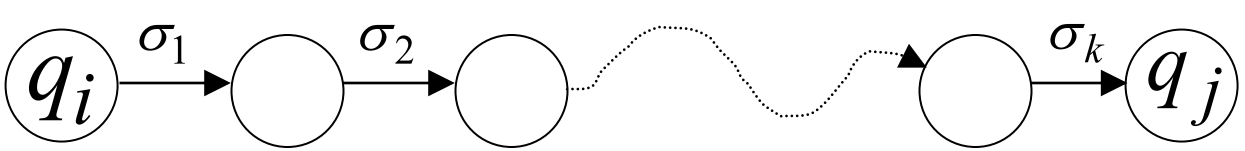
$q_j \in \delta^*(q_i, w)$: there is a walk from q_i to q_j with label w

$\delta^*(q_i, w) = q_j$



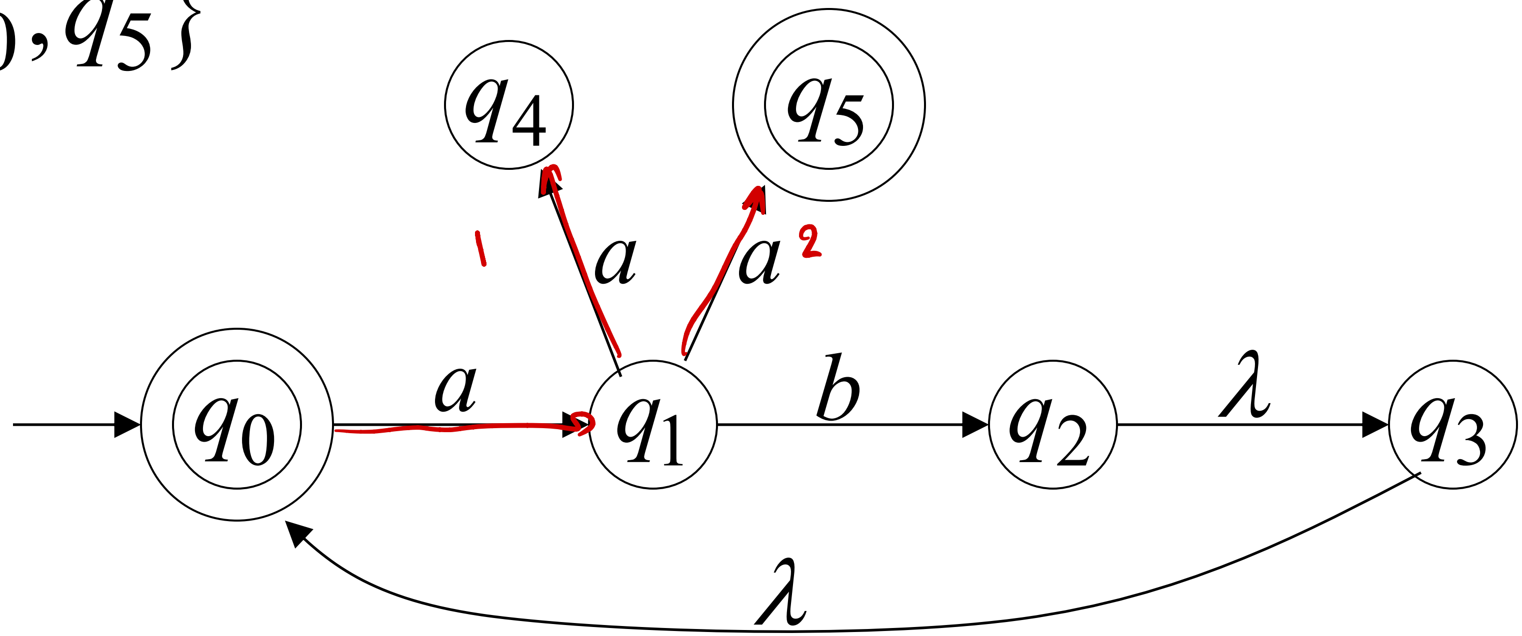
alphabet หลายๆตัวต่อกัน

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



The Language of an NFA M

$F = \{q_0, q_5\}$

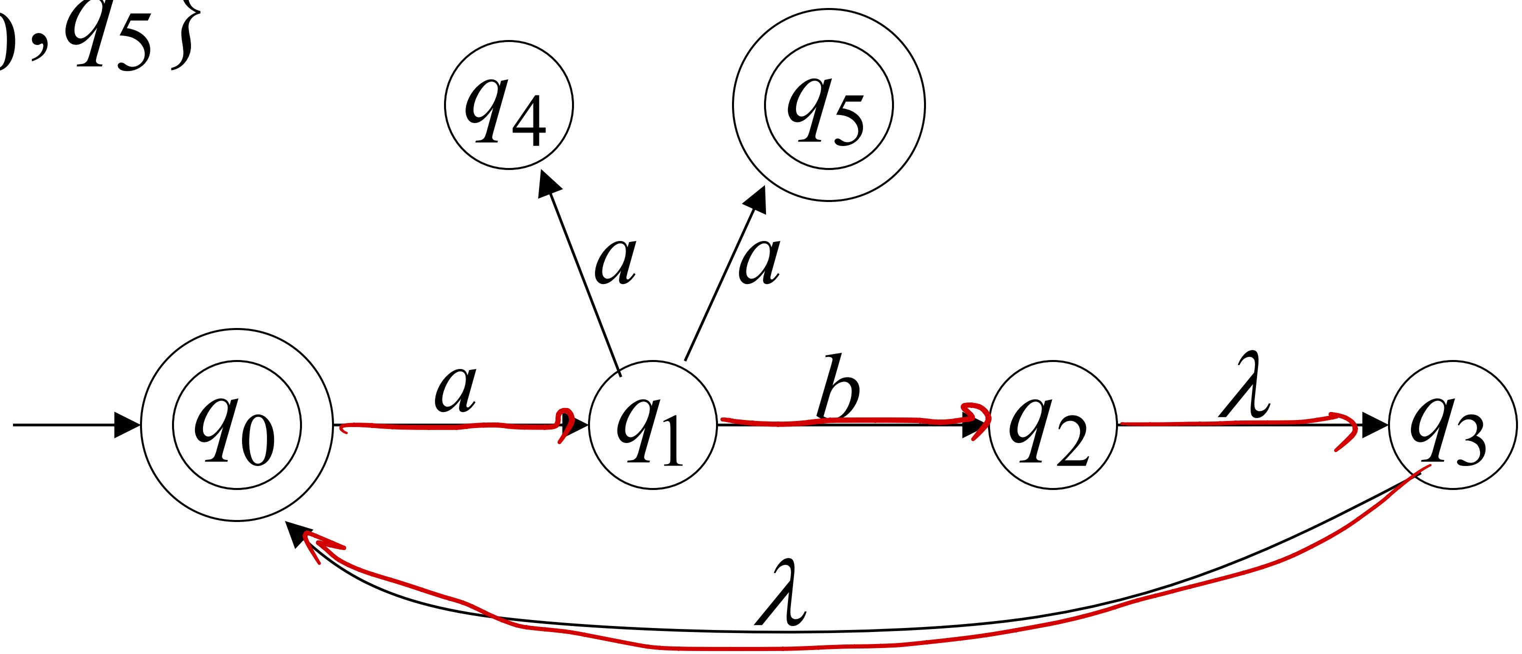


$$\delta^*(q_0, aa) = \{q_4, \underline{q_5}\}$$

$\searrow \in F$

$aa \in L(M)$

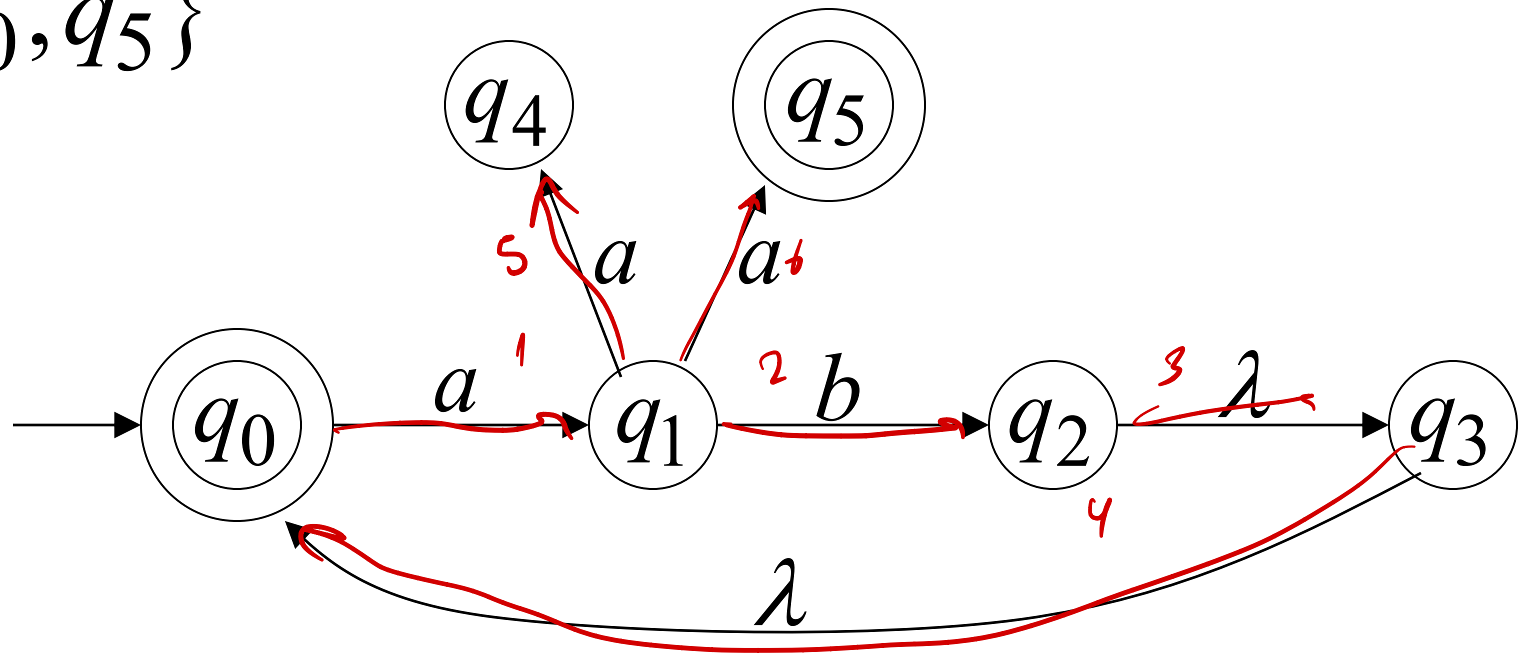
$$F = \{q_0, q_5\}$$



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

$$\searrow \in F$$

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



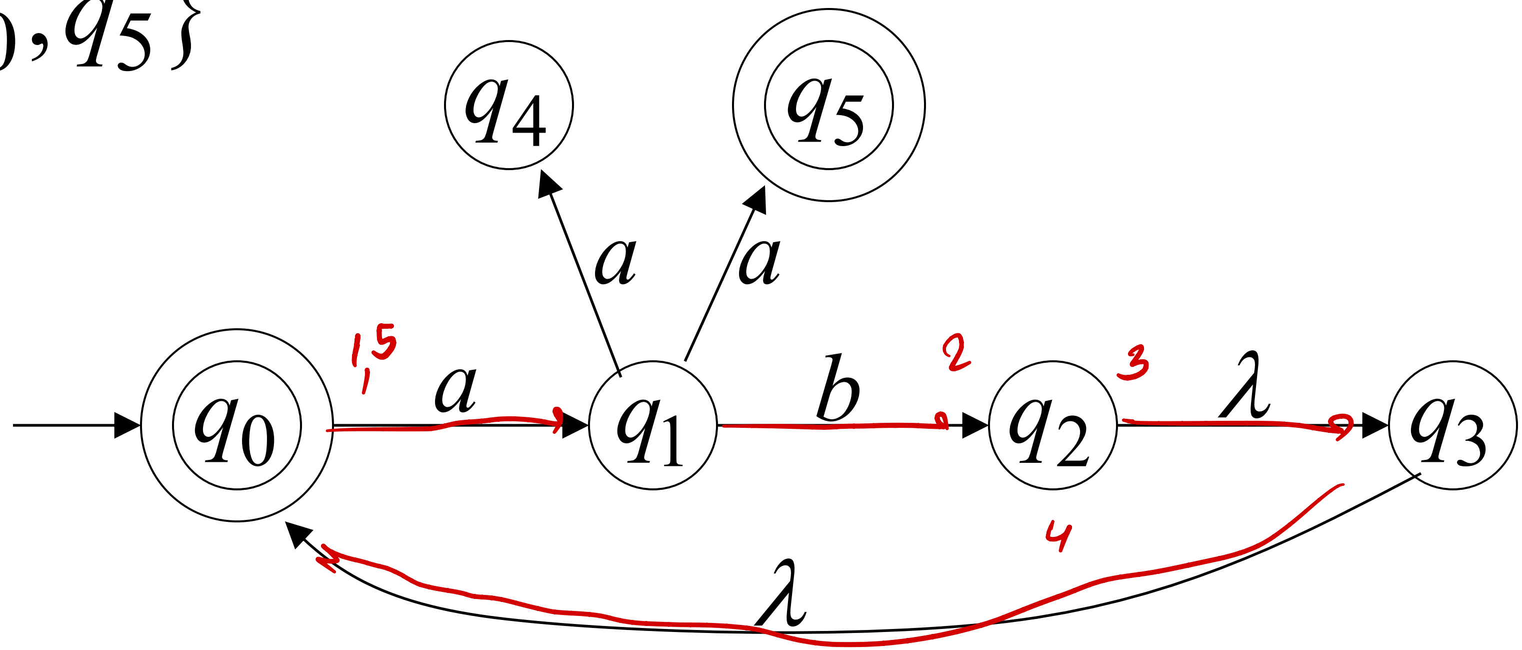
$$\delta^*(q_0, abaa) = \{q_4, \underline{q_5}\}$$

$\searrow \in F$

abaa

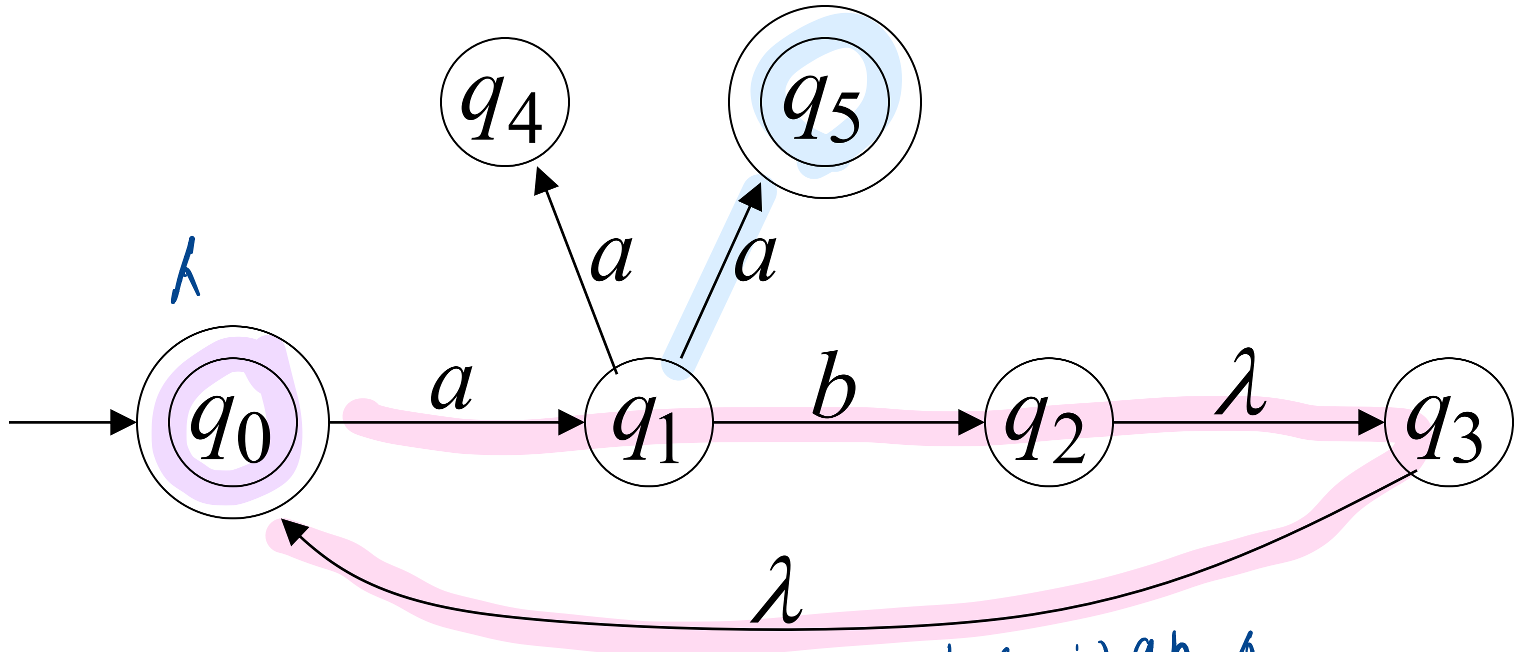
$\cancel{aaba} \in L(M)$

$F = \{q_0, q_5\}$



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

$aba \notin L(M)$



concate (การต่อ) ab . λ

ต่อ

ab . aa จะหยุดที่ q5

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

$$= \{ab\}^*.\{\lambda, aa\}$$

$\{\lambda\} \cup \{\lambda\}.\{\lambda\} = \{\lambda\}$

2 ตัดออกได้

Formally

The language accepted by NFA M is:

Set ของstring

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

เริ่มที่

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

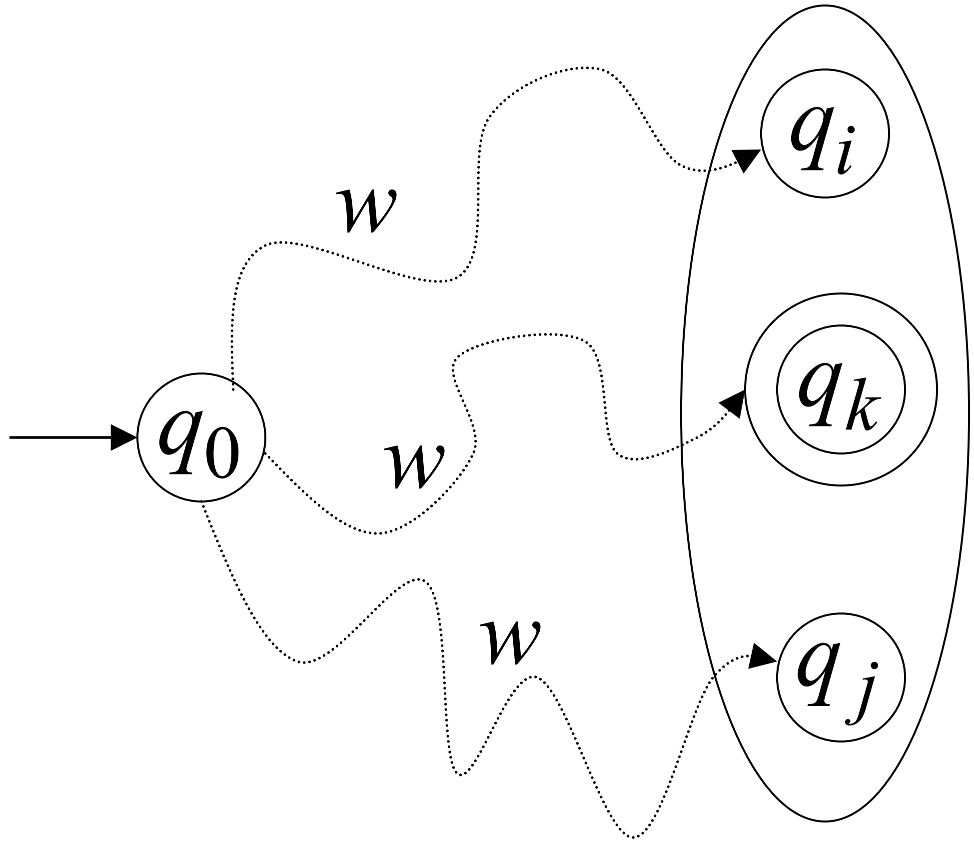
(ป้อนทีละ string)

ต้องมีอย่างน้อย 1state ที่เป็น final state

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

$w \in L(M)$

$\delta^*(q_0, w)$



$q_k \in F$

$\{q_i, q_k, q_j\}$

final state

แสดงว่า $w \in L(M)$

NFAs accept the Regular Languages

Equivalence of Machines

Definition for Automata:

เท่าเทียมกันทุกประการ

Machine M_1 is equivalent to machine M_2

1.ถ้า

if $\boxed{L(M_1) = L(M_2)}$

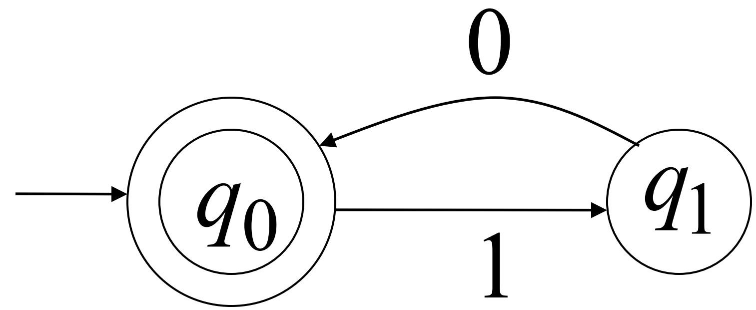
2 สรุปว่า

Example of equivalent machines

กัน เพราะว่า รองรับ
ภาษาเดียวกัน

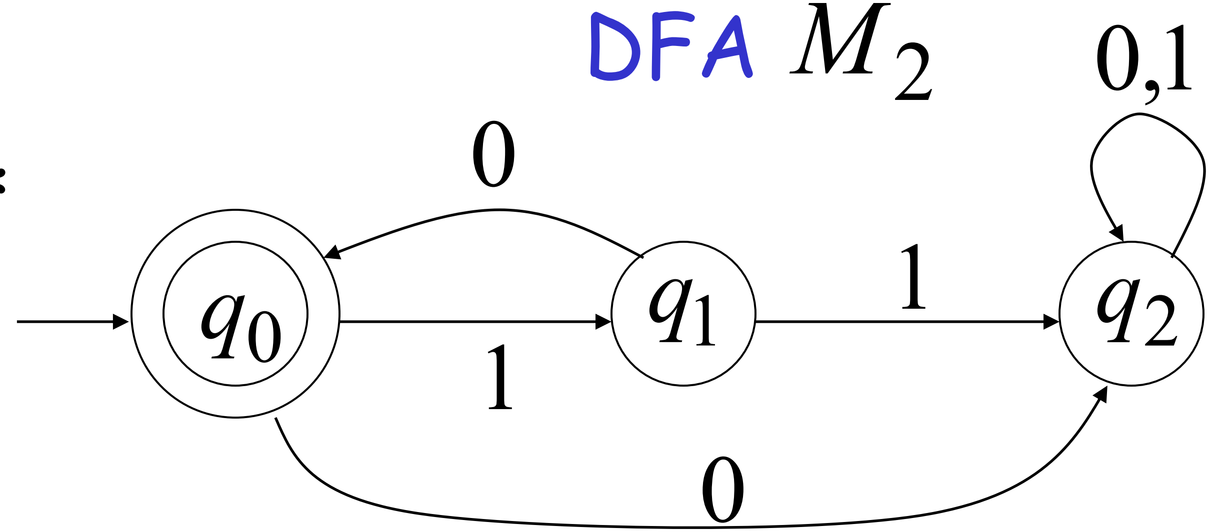
NFA M_1

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



DFA M_2

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



We will prove:

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

Languages
accepted
by DFAs

NFAs and DFAs have the same computation power

$Q \times \Sigma \longrightarrow Q$ DFA : $\delta(q_i, a) = q_j$

Step 1

$Q \times \Sigma \rightarrow 2^Q$ NFA : $\delta(q_i, a) = \{q_i, q_j\}$

เป็น subset ของ

powerset ของ Q

หมายความ $\boxed{Q \subset 2^Q}$

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

Proof: Every DFA is trivially an NFA

⇓

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

Step 2

เป็น subset ของ

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

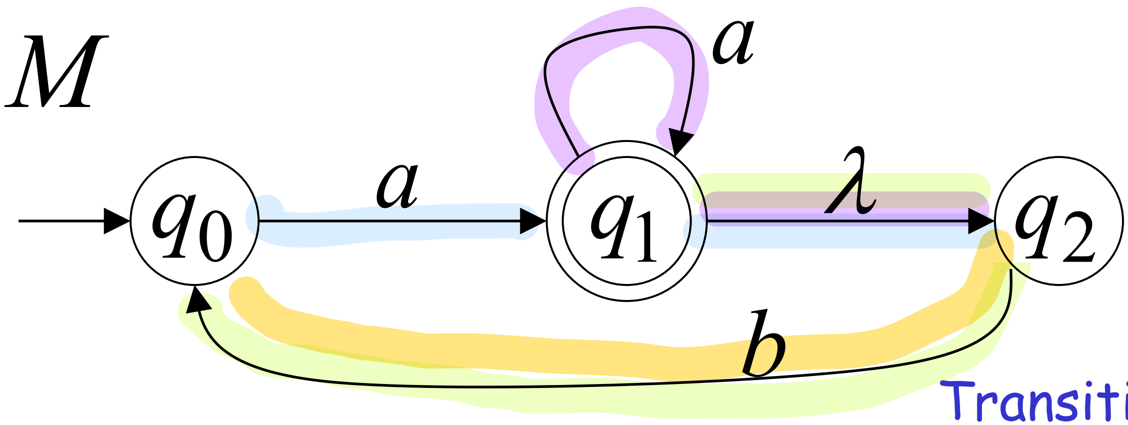
Proof: Any NFA can be converted to an equivalent DFA

⇩

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

Convert NFA to DFA

NFA M

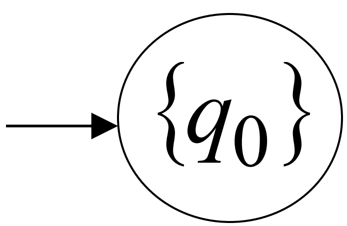


Transition table for NFA M

	input a	input b
q0	{q1, q2}	ไม่มี state ที่จะไปได้ ∅
q1	{q1, q2}	{q0}
q2	∅	{q0}

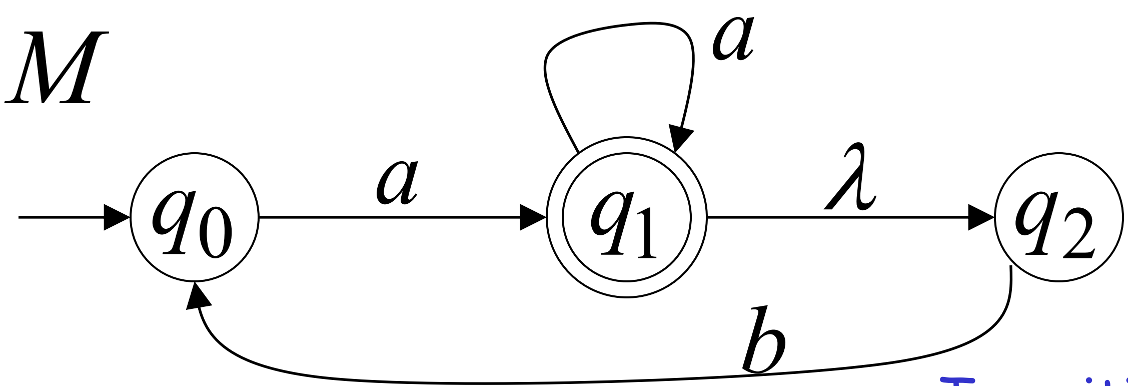
Design DFAs จาก Table →

DFA M'



Convert NFA to DFA

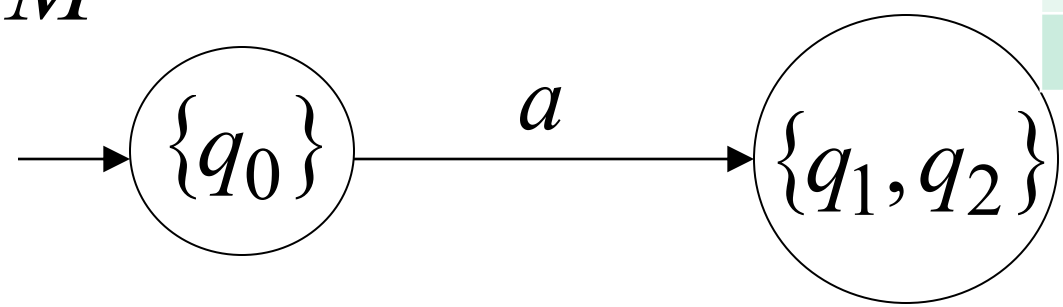
NFA M



Transition table for NFA M

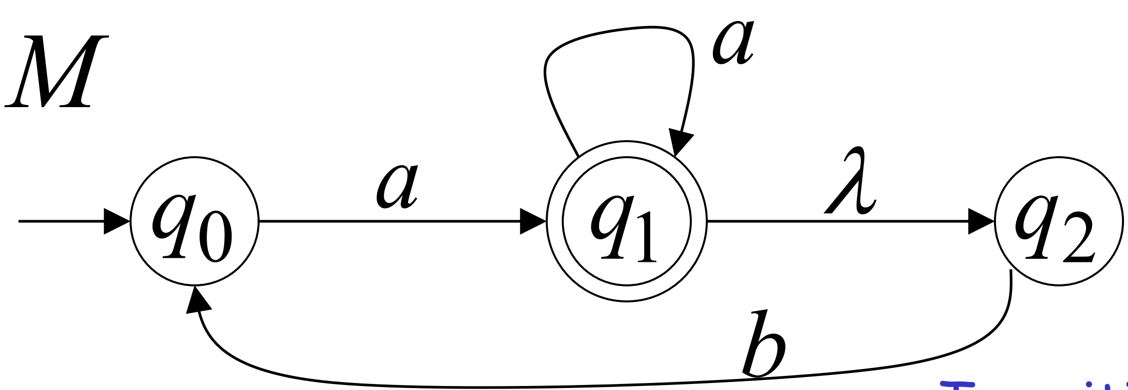
	a	b
q0	{q1, q2}	∅
q1	{q1, q2}	{q0}
q2	∅	{q0}

DFA M'



Convert NFA to DFA

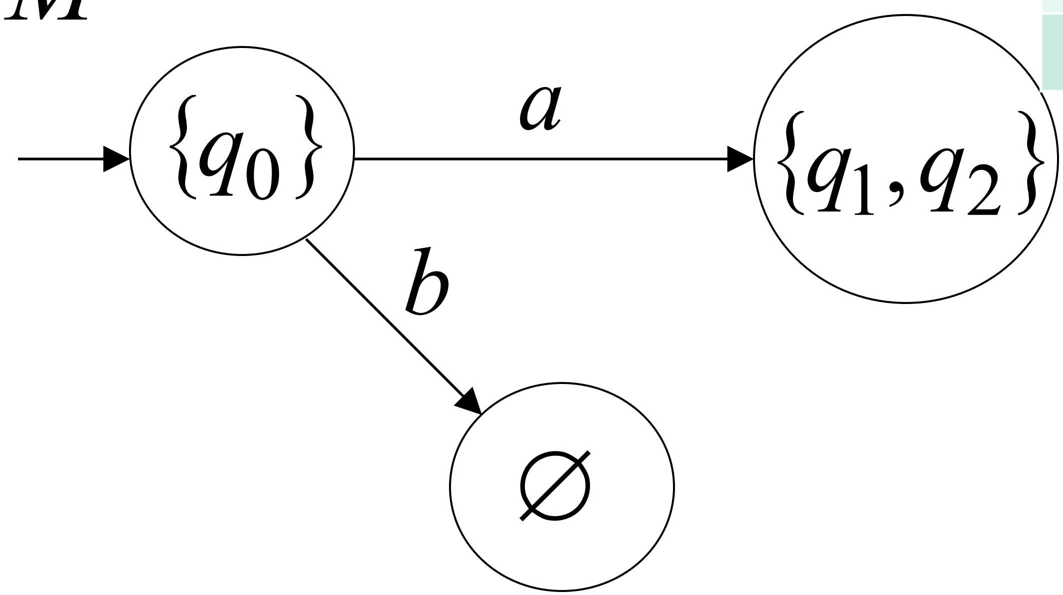
NFA M



Transition table for NFA M

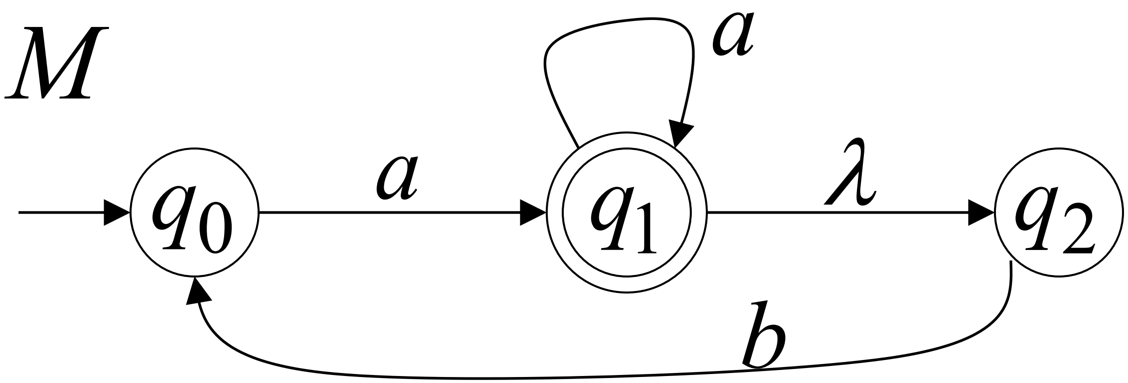
	a	b
q0	{q1, q2}	∅
q1	{q1, q2}	{q0}
q2	∅	{q0}

DFA M'

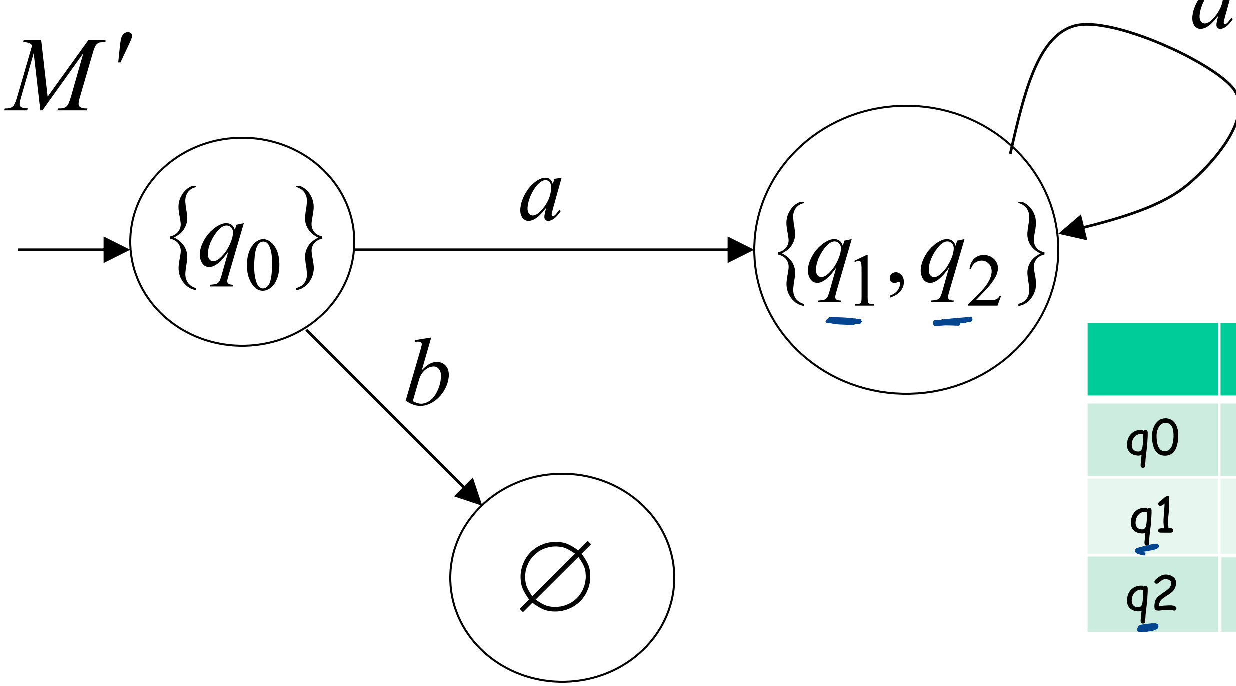


Convert NFA to DFA

NFA M



DFA M'

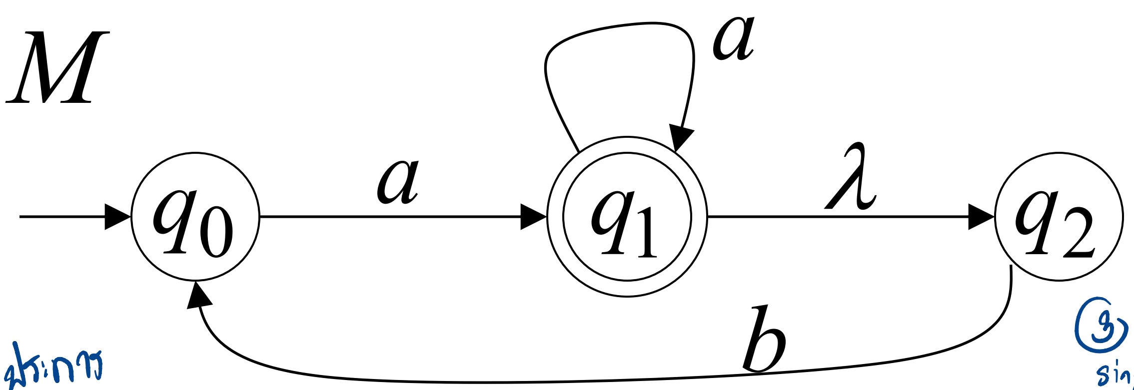


$\{q_1, q_2\} \cup \emptyset = \{q_1, q_2\}$
ได้ชื่อ state

	a	b
q0	{q1, q2}	∅
q1	{q1,q2}	{q0}
q2	∅	{q0}

Convert NFA to DFA

NFA M

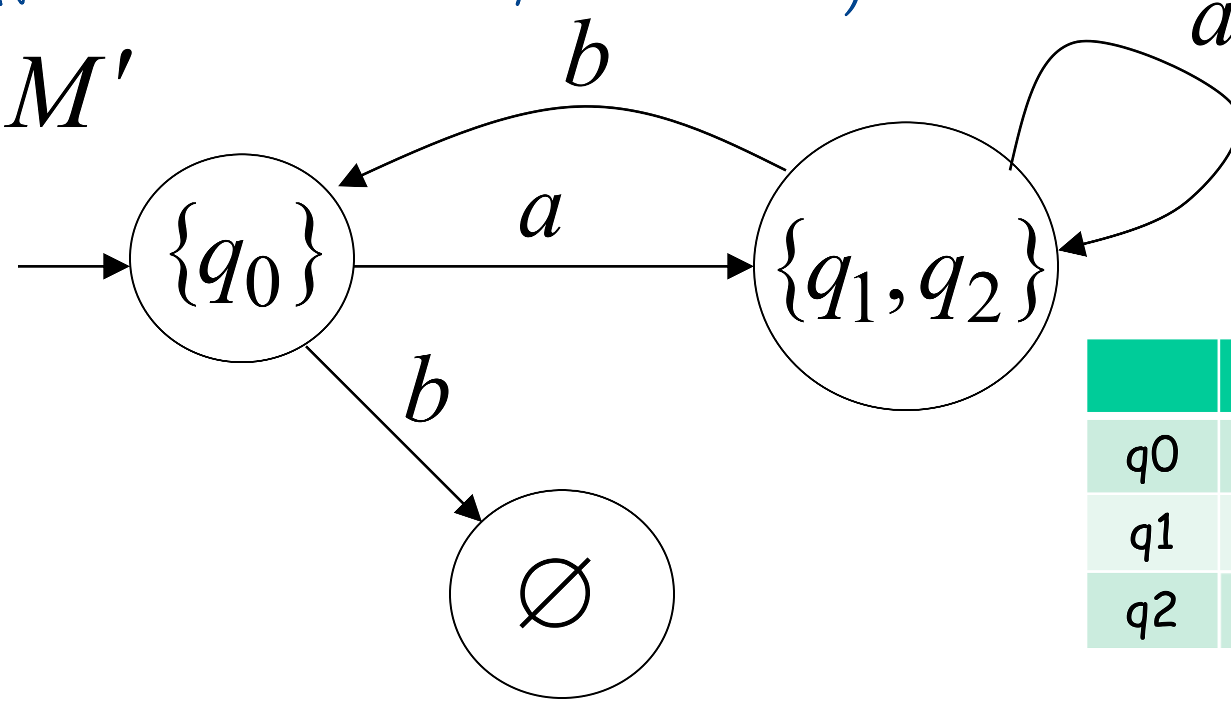


ความต่าง 3 ประการ

① ไม่ซ้ำเฟรม ต้องมี trap (ตัวดัก) , ② ไม่มี λ transition , ③ single state DFA: $Q \times \Sigma \to Q$, NFA $Q \times \Sigma \to 2^Q$

DFA M'

(เก่งกว่า)



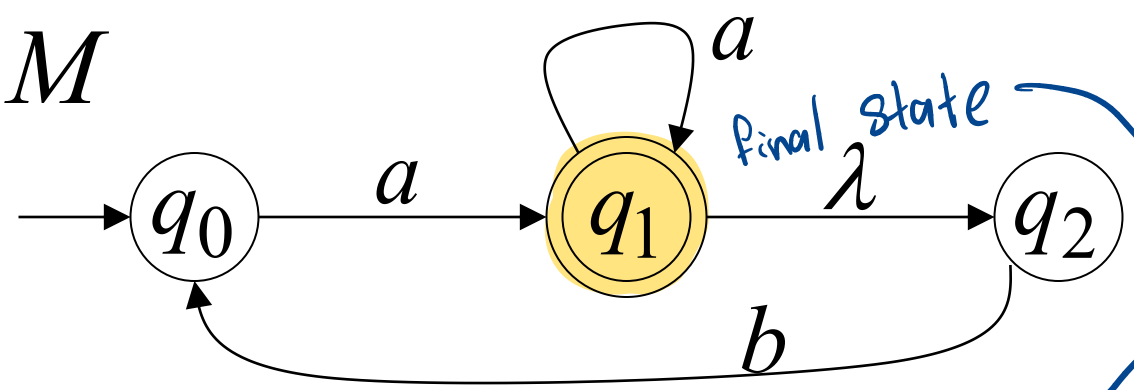
	a	b
q0	{q1, q2}	∅
q1	{q1, q2}	{q0}
q2	∅	{q0}

∪

= ชื่อ state

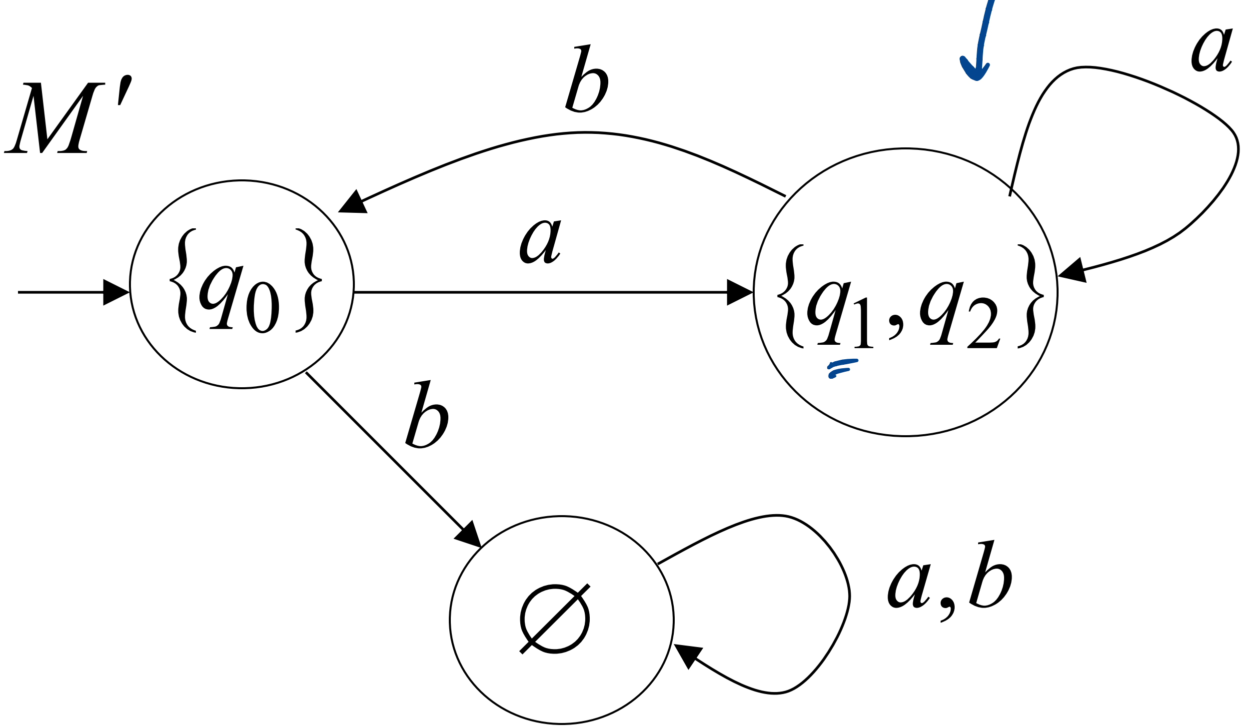
Convert NFA to DFA

NFA M



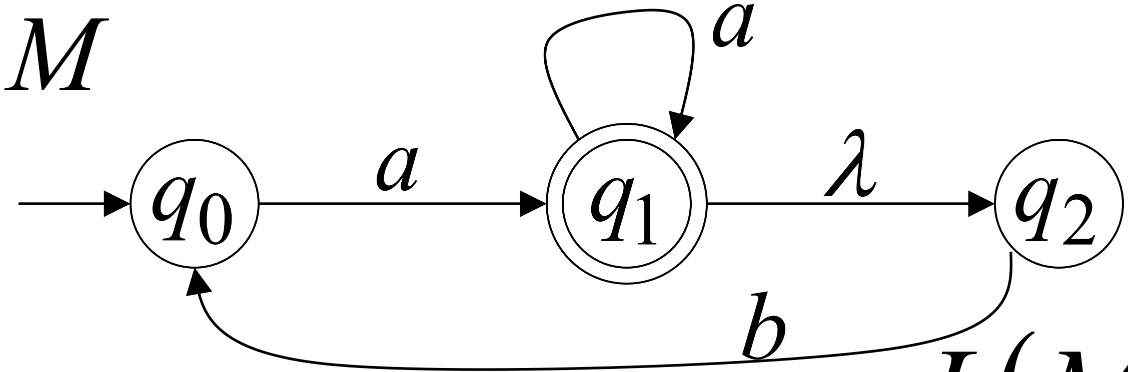
ตัวไหนที่มี final state
= จุดจบ(final state)

DFA M'



Convert NFA to DFA

NFA M



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

DFA M'