Finite Automata

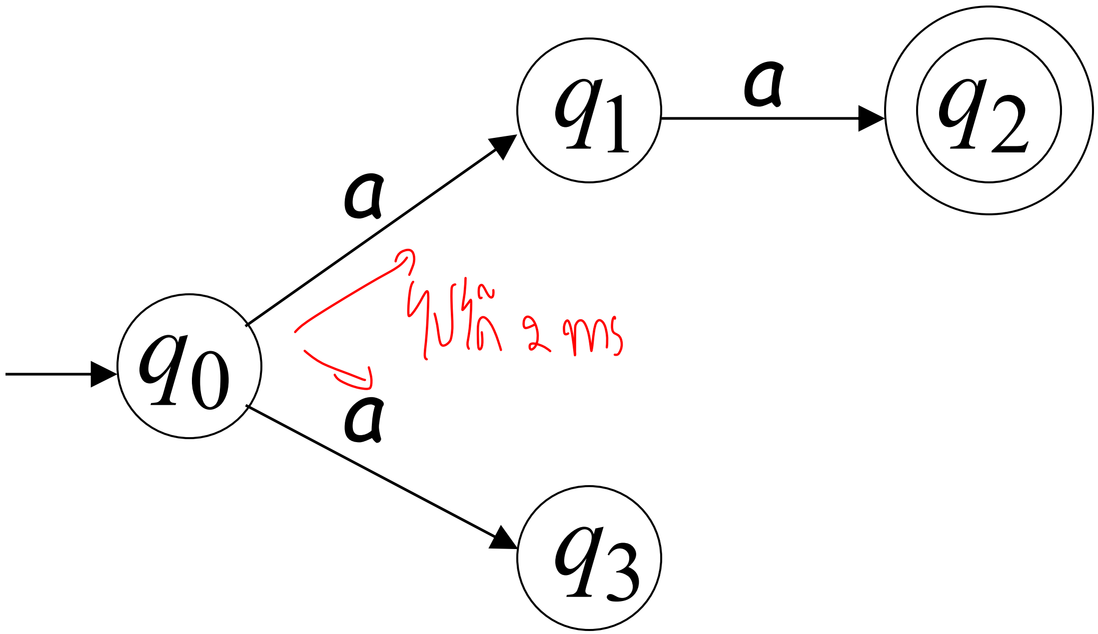
Deterministic FA

Nondeterministic FA

Nondeterministic Finite Automata

Nondeterministic Finite Automata (NFA)

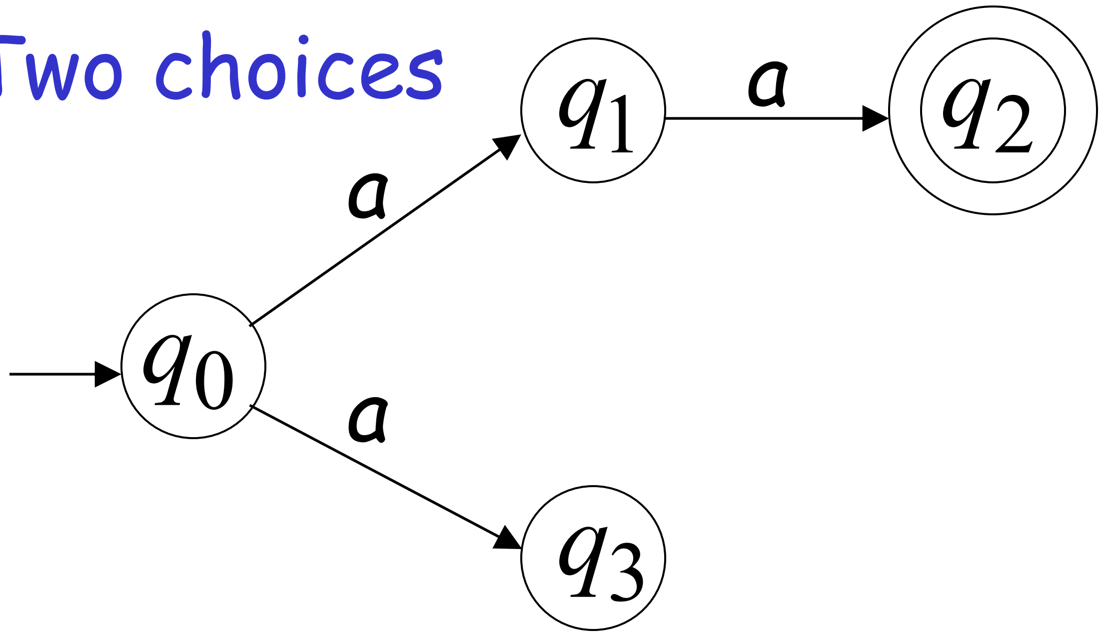
Alphabet = $\{a\}$



Nondeterministic Finite Automata (NFA)

Alphabet = $\{a\}$

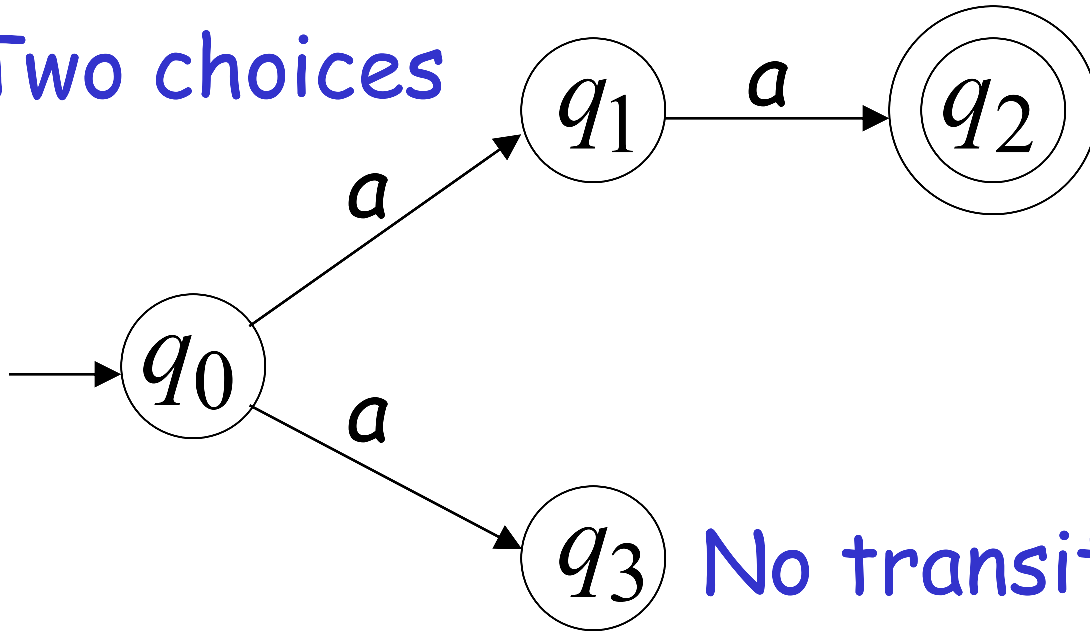
Two choices



Nondeterministic Finite Automata (NFA)

Alphabet = $\{a\}$

Two choices

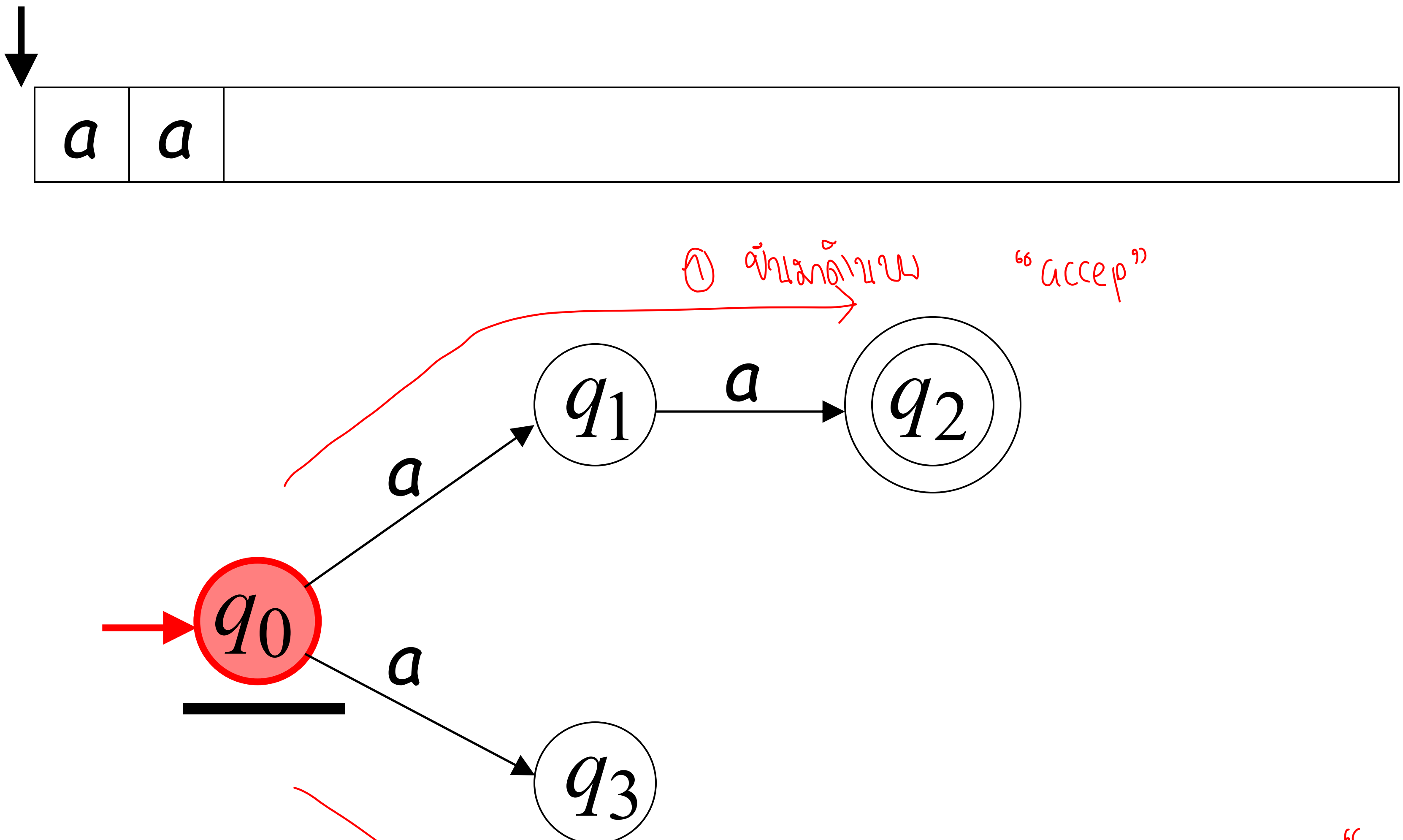


No transition

ทางตัน

No transition

First Choice

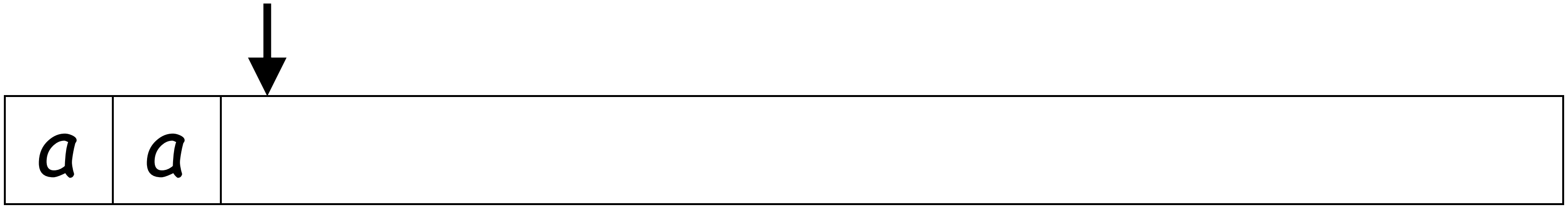


① ขึ้นหมดด้านบน "accep"

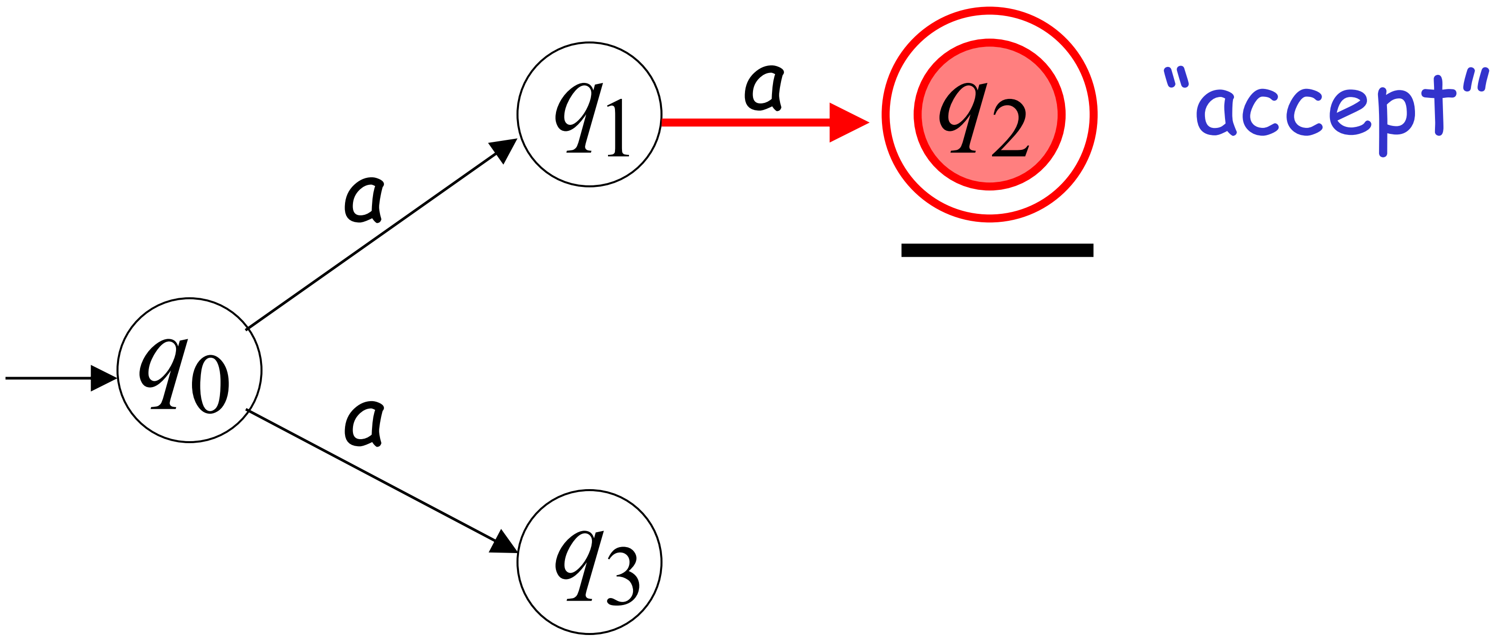
"reject"

② การเดิน → รับ input ไม่หมด เหลือ "a" อีกตัว

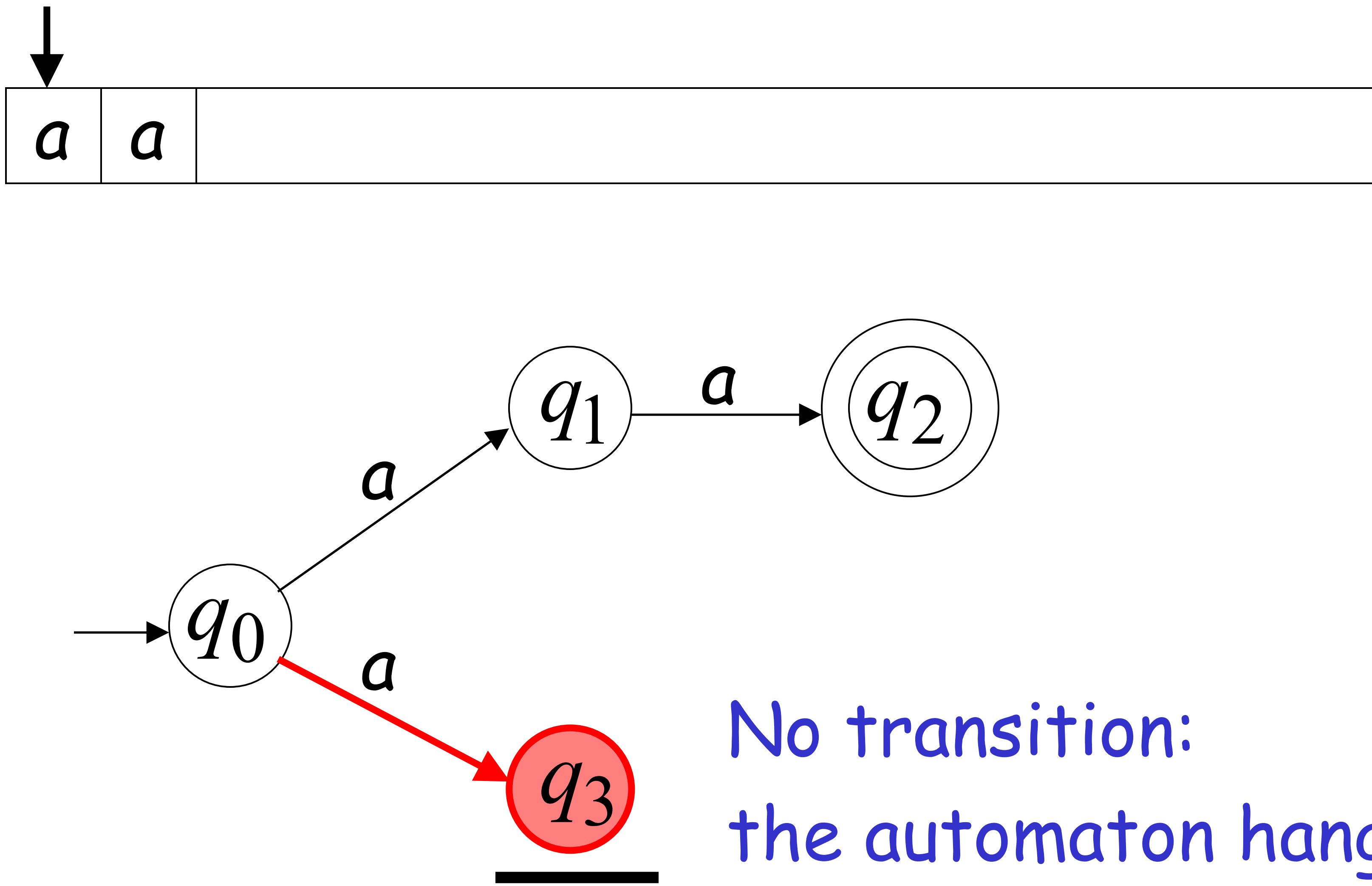
First Choice



All input is consumed

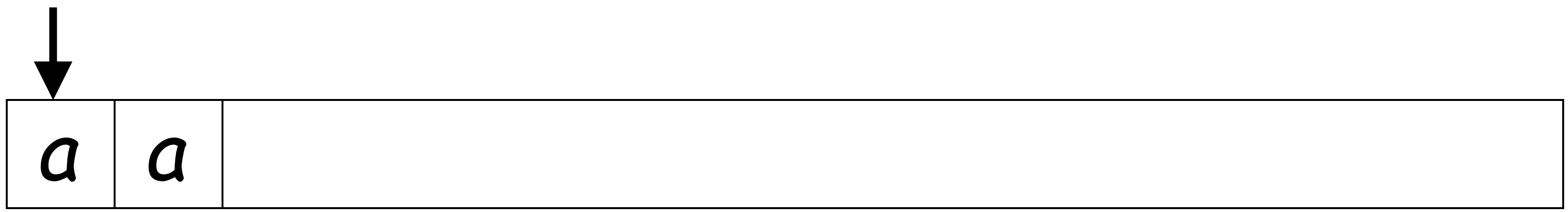


Second Choice

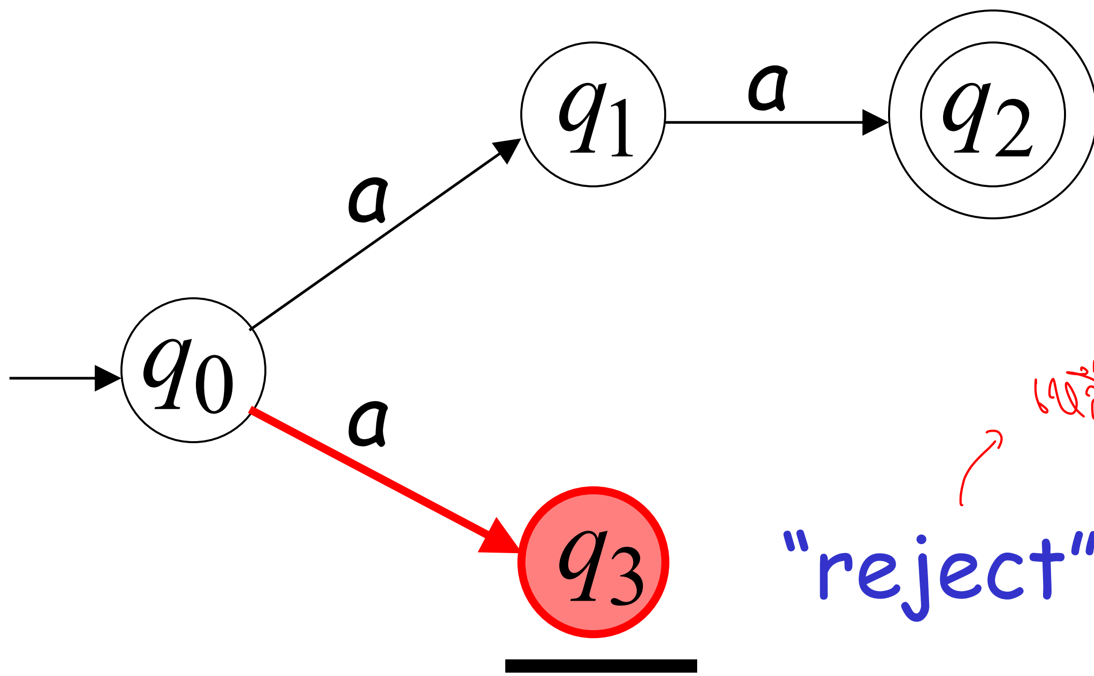


No transition:
the automaton hangs

Second Choice



Input cannot be consumed



เหลือ input อีกตัวนะจ้า

"reject"

An NFA accepts a string:

If there is a computation such that:

All the input is consumed

AND

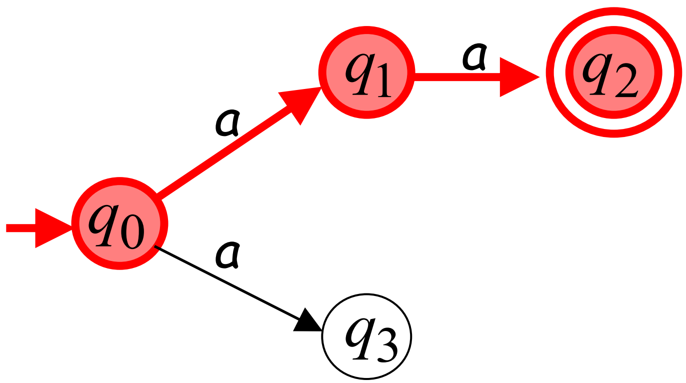
The automata is in a final state

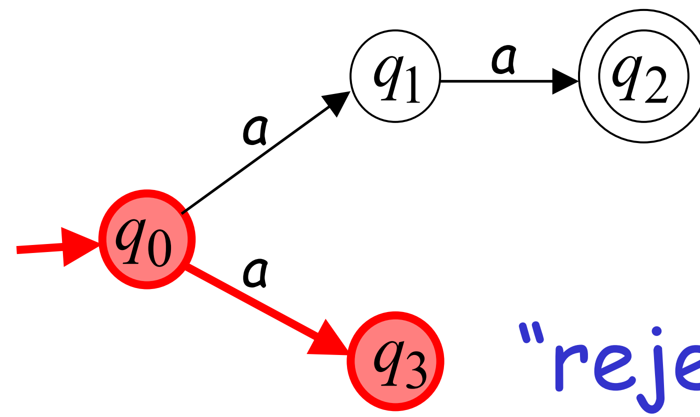
Example

aa is accepted by the NFA:

ต้องมีทางใดทางหนึ่งที่ accept

"accept"

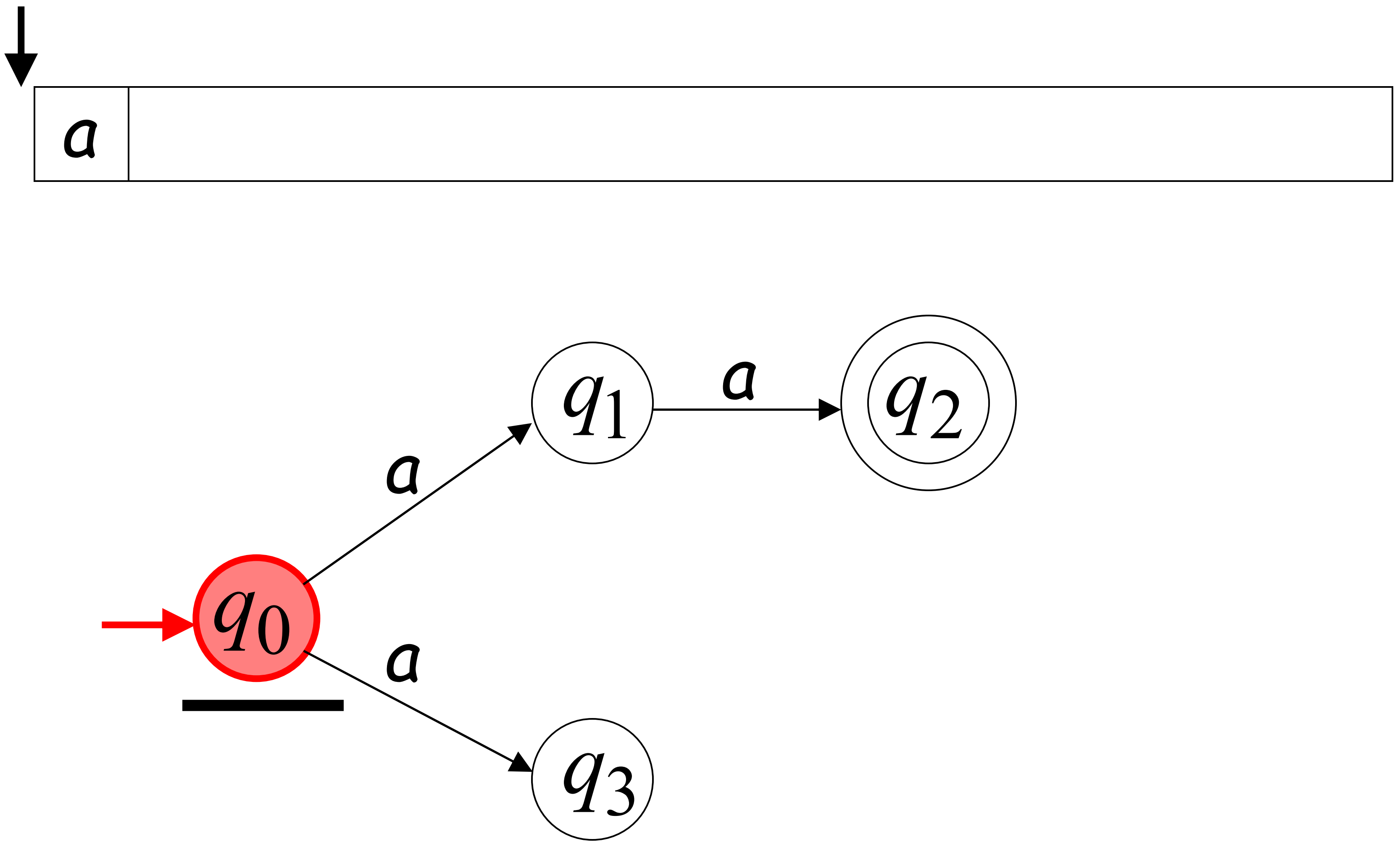




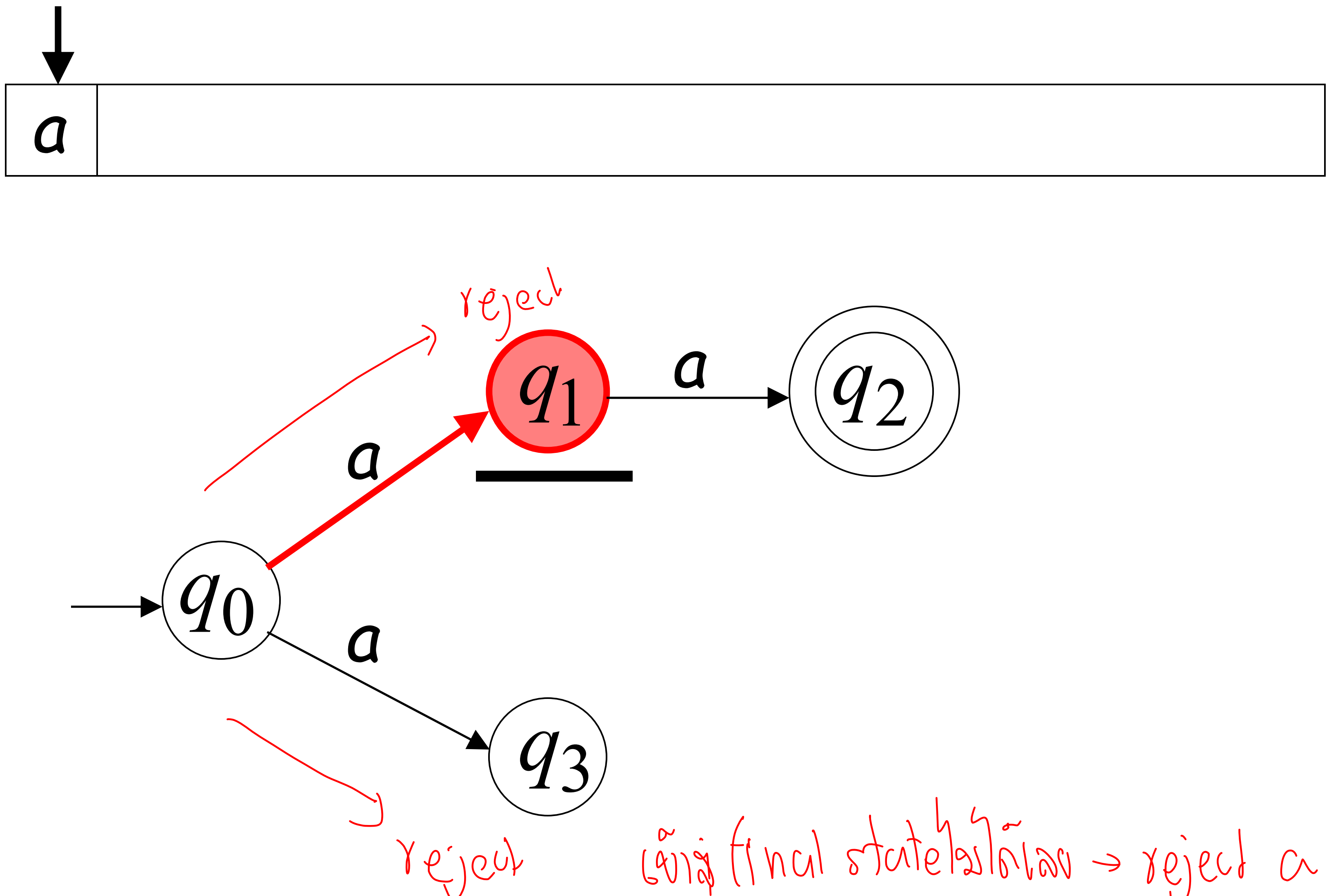
"reject"

because this computation accepts aa

Rejection example

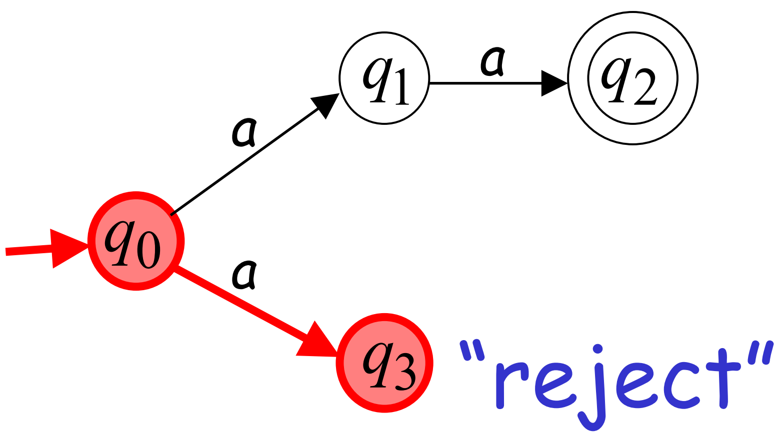


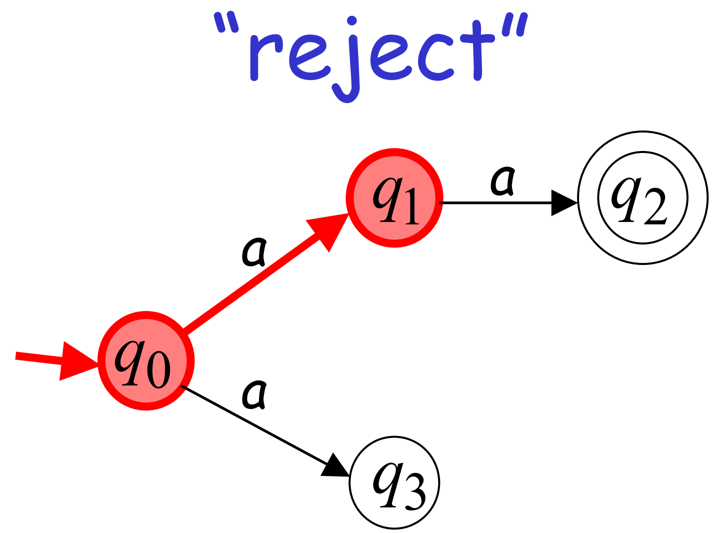
First Choice



เข้าสู่ final state ไม่ได้เลย → reject a

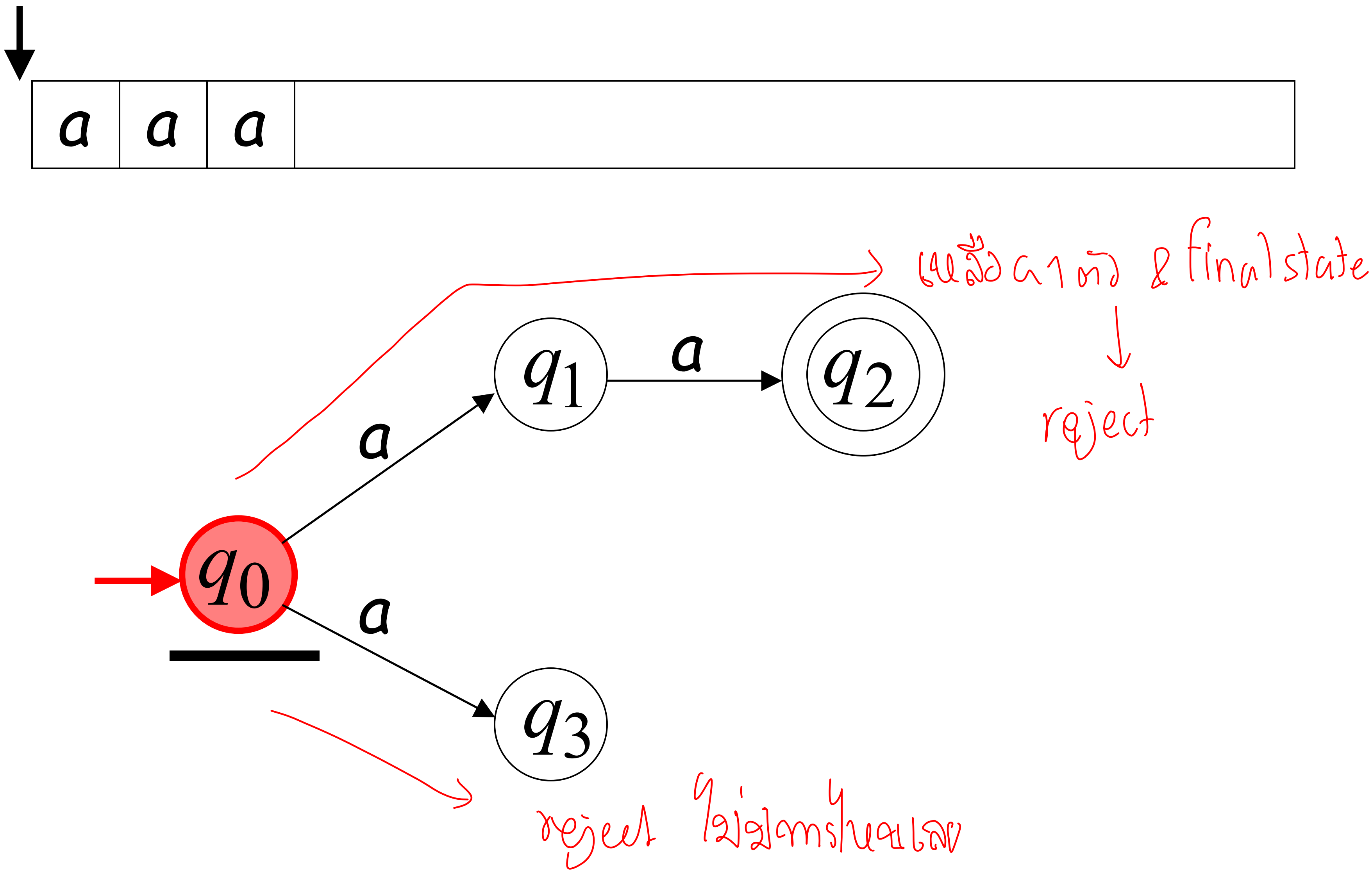
a is rejected by the NFA:



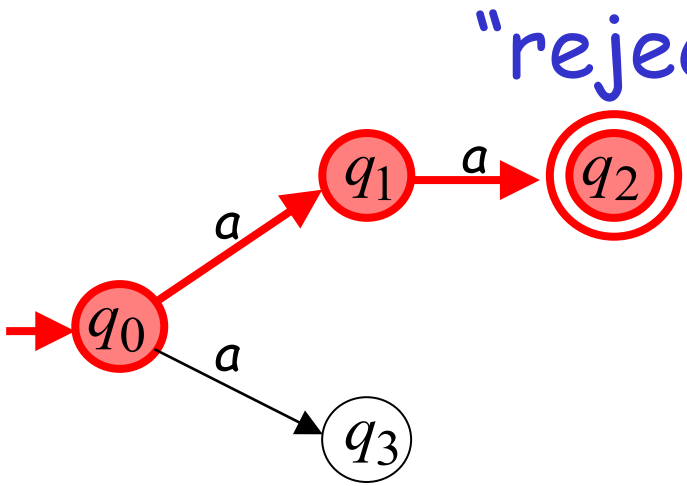


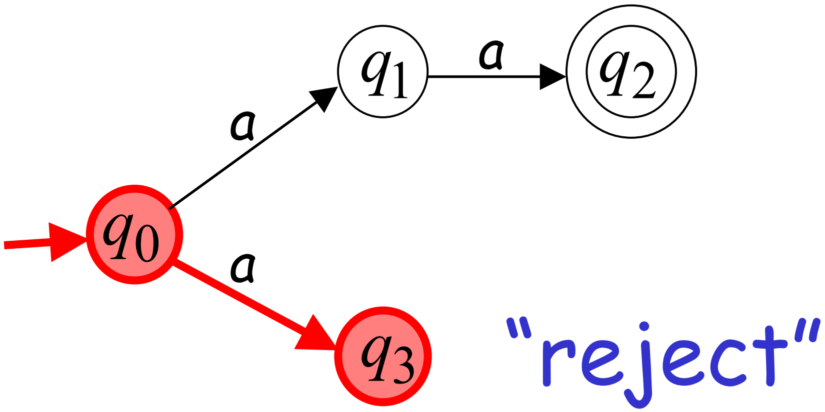
All possible computations lead to rejection

Rejection example



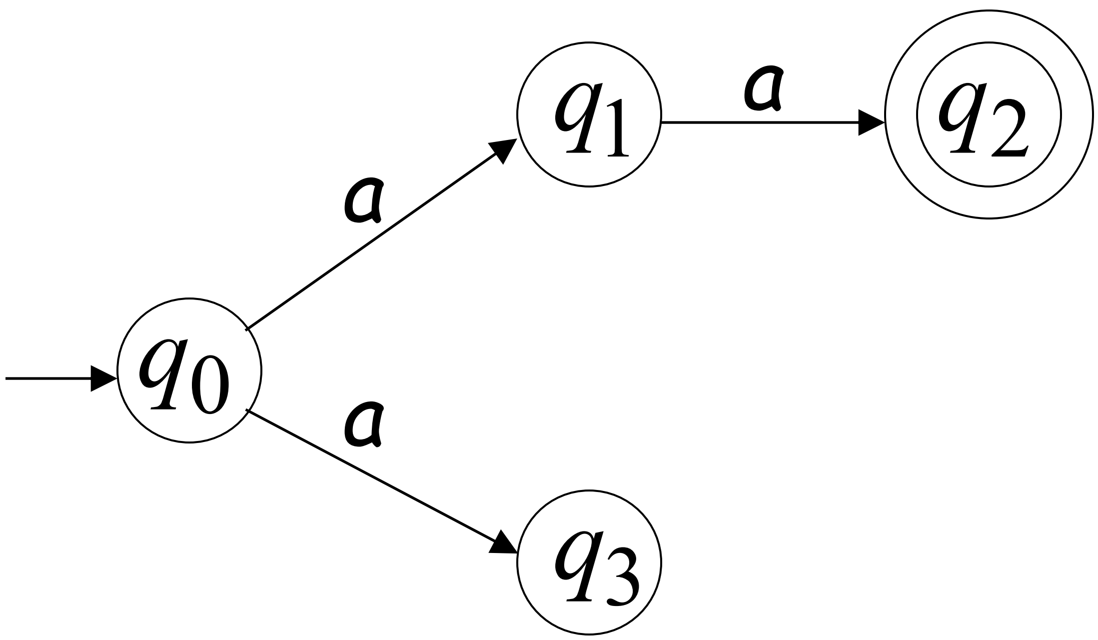
aaa is rejected by the NFA:





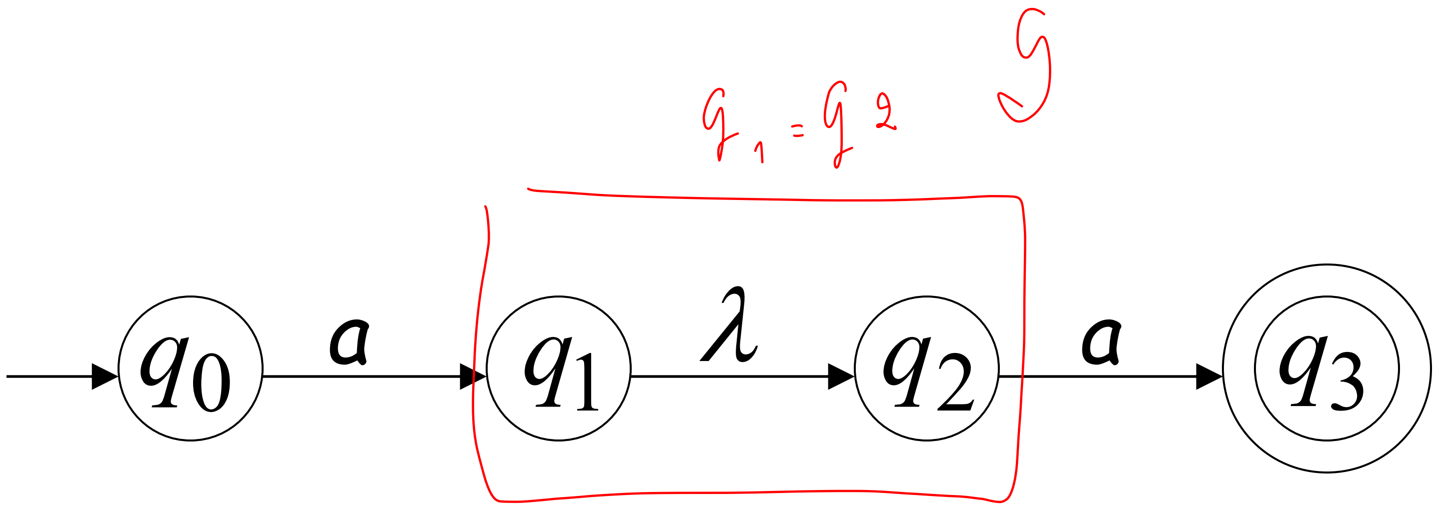
All possible computations lead to rejection

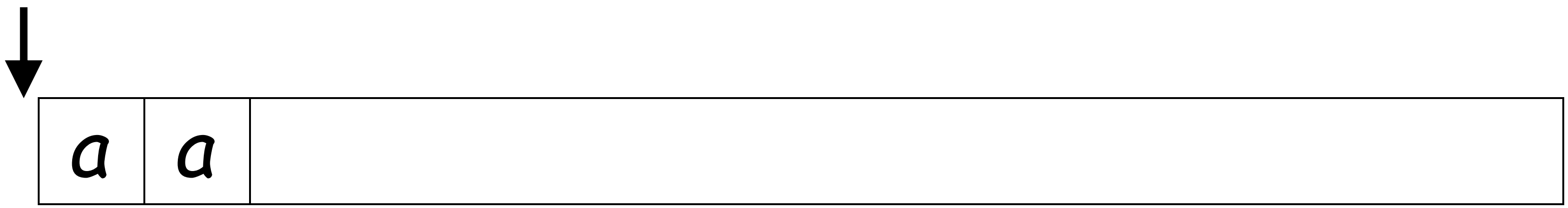
Language accepted: $L = \{aa\}$



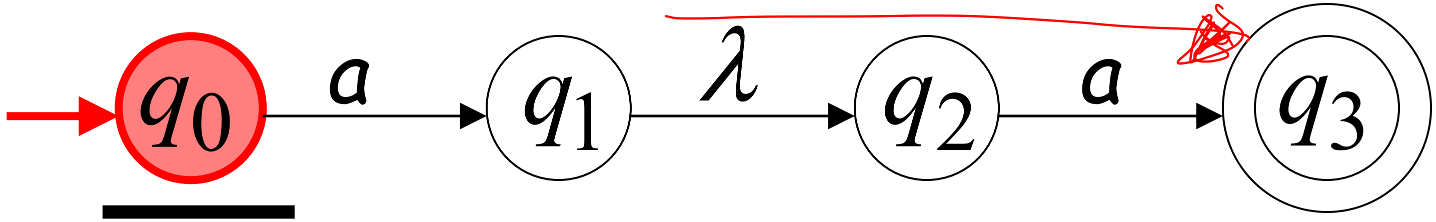
Lambda Transitions

empty string



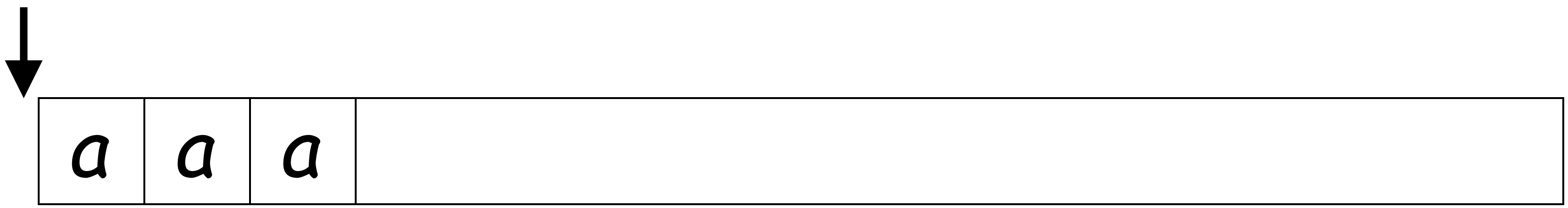


ต้องใช้ a แต่ λ มันเหมือนทางเดียวกัน $a\lambda a = a$

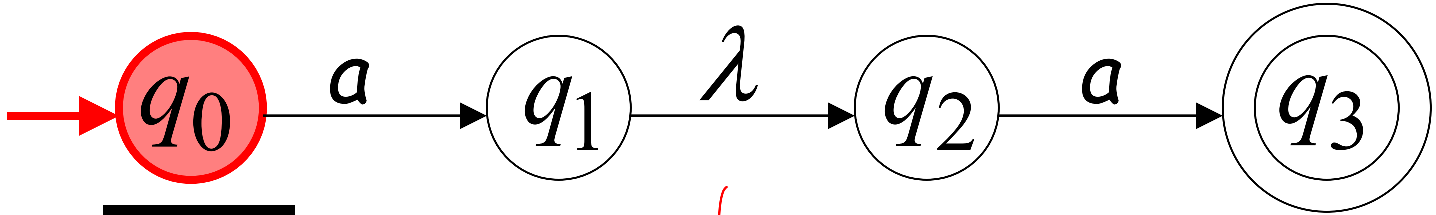


accept aa

Rejection Example

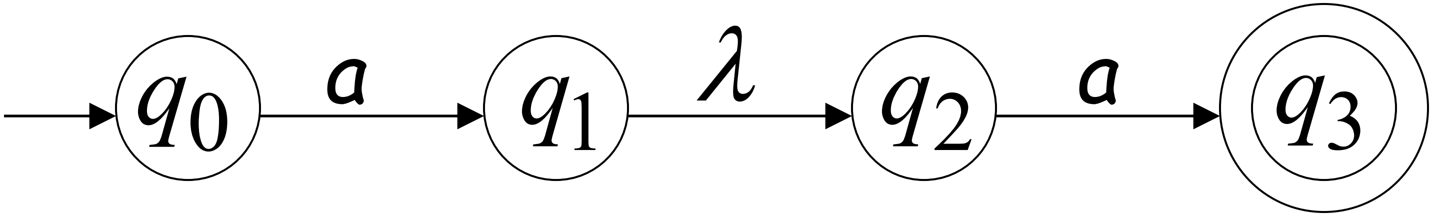


a_1 → ใช้ λ ฟรีๆ → a_2 → เหลือ a อีกตัว

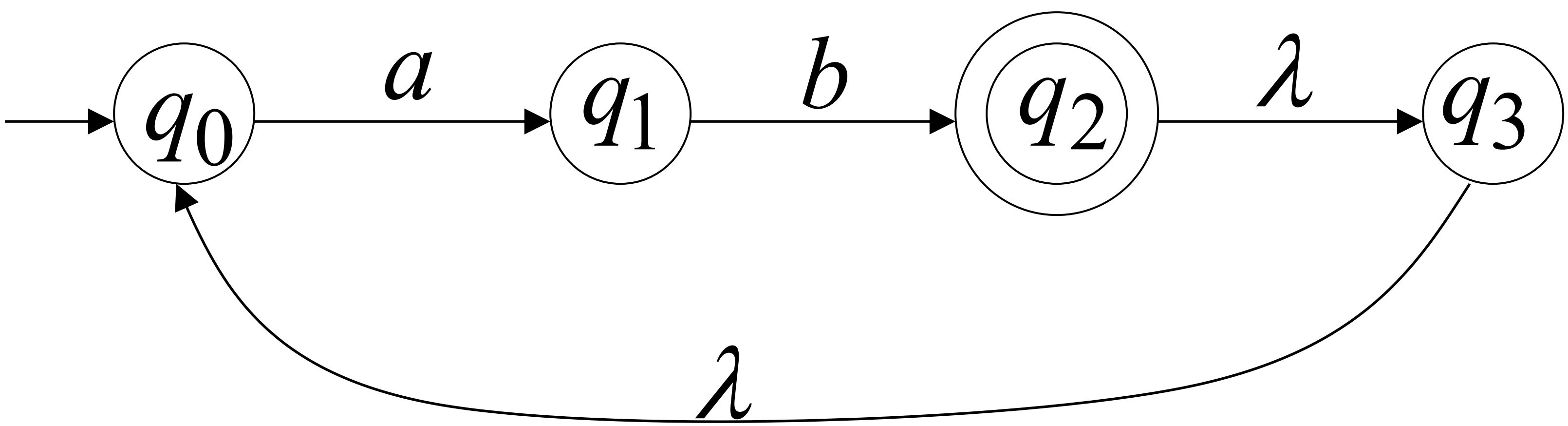


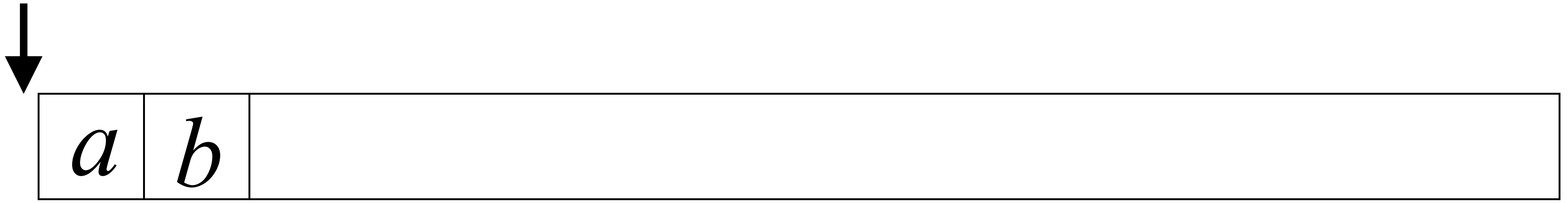
เหมือน transition ที่กระโดดข้ามได้แบบฟรีๆ

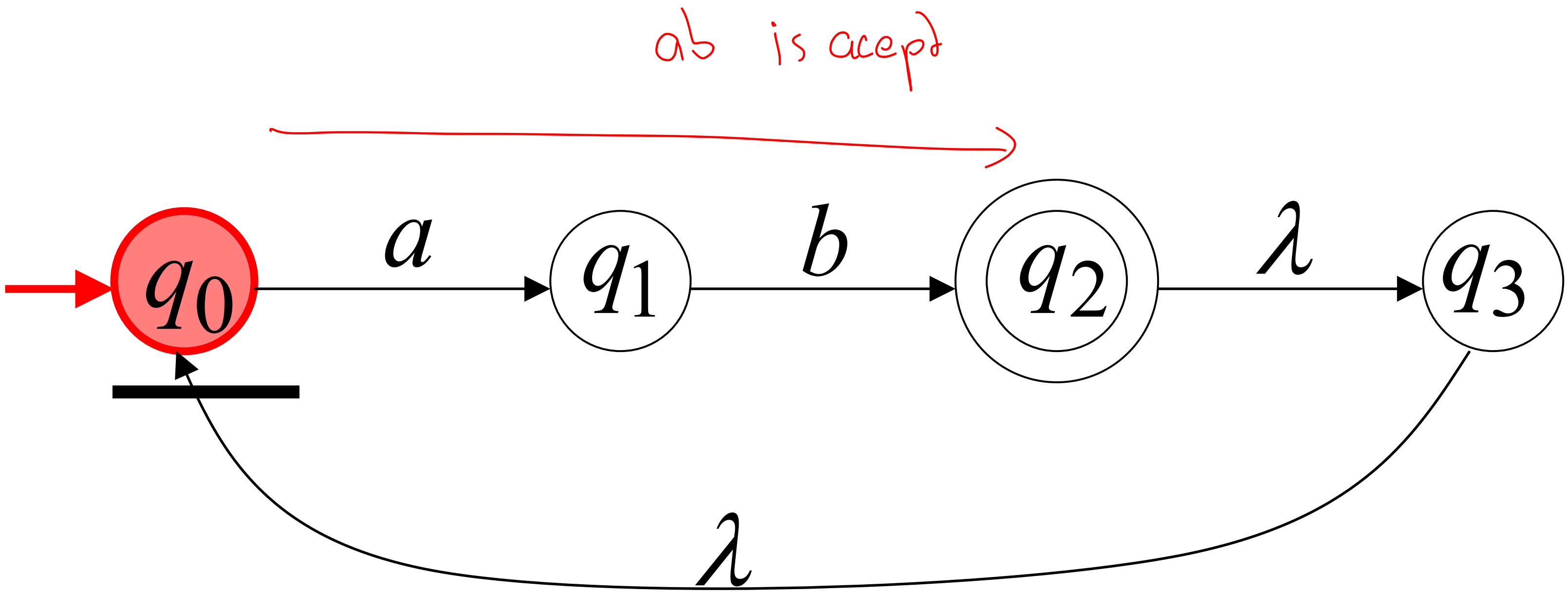
Language accepted: $L = \{aa\}$



Another NFA Example

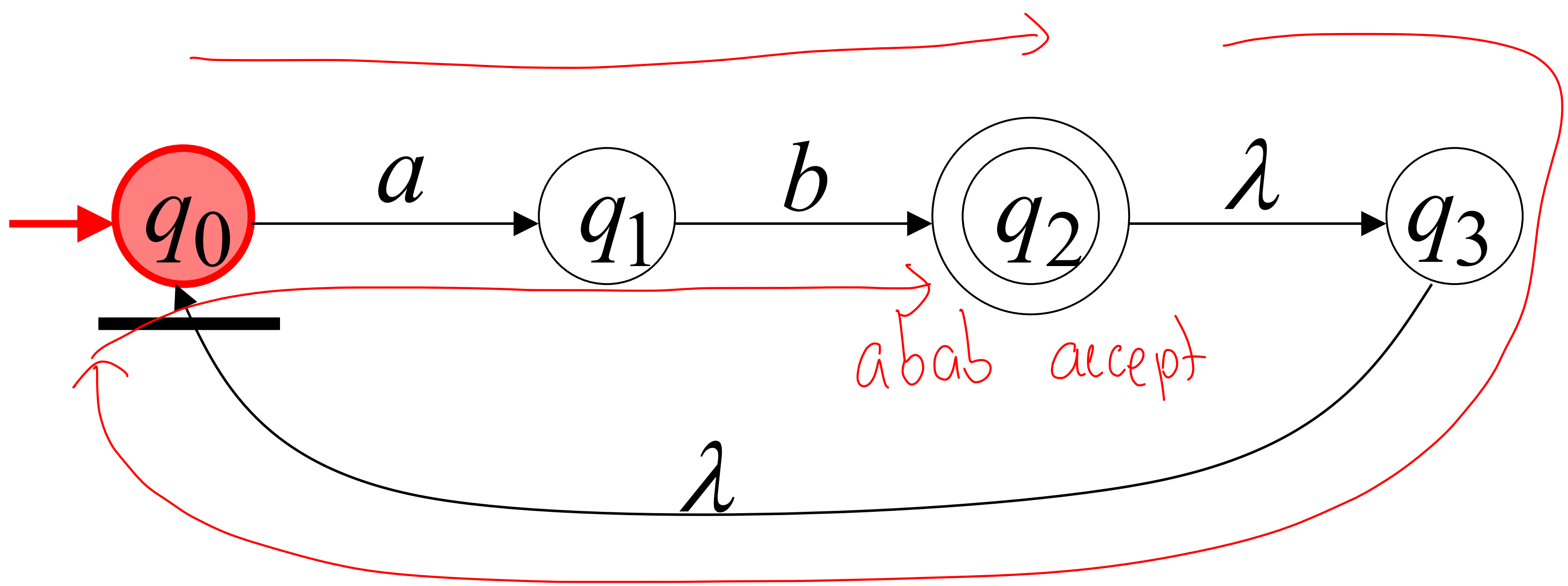



a
b
ab is acept
q_0
q_1
q_2
q_3
λ


a
b
λ
q_0
q_1
q_2
q_3
λ

Another String

a	b	a	b	

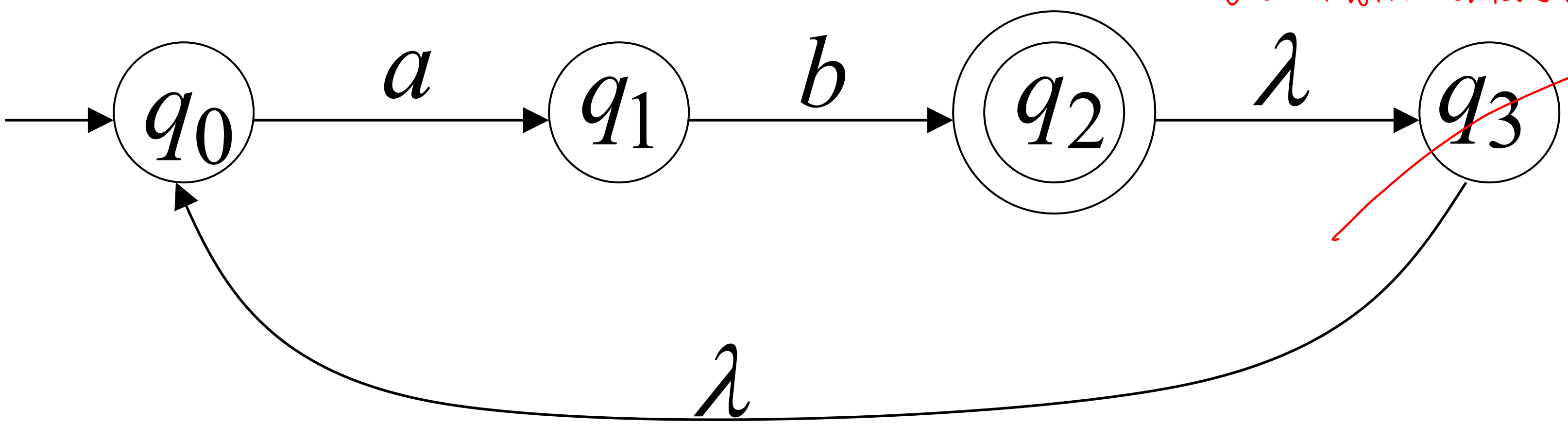


Language accepted

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

$$= \{ab\}^{+}$$ ไม่รวม λ

ต้องได้ทำงานกลับเองเลย

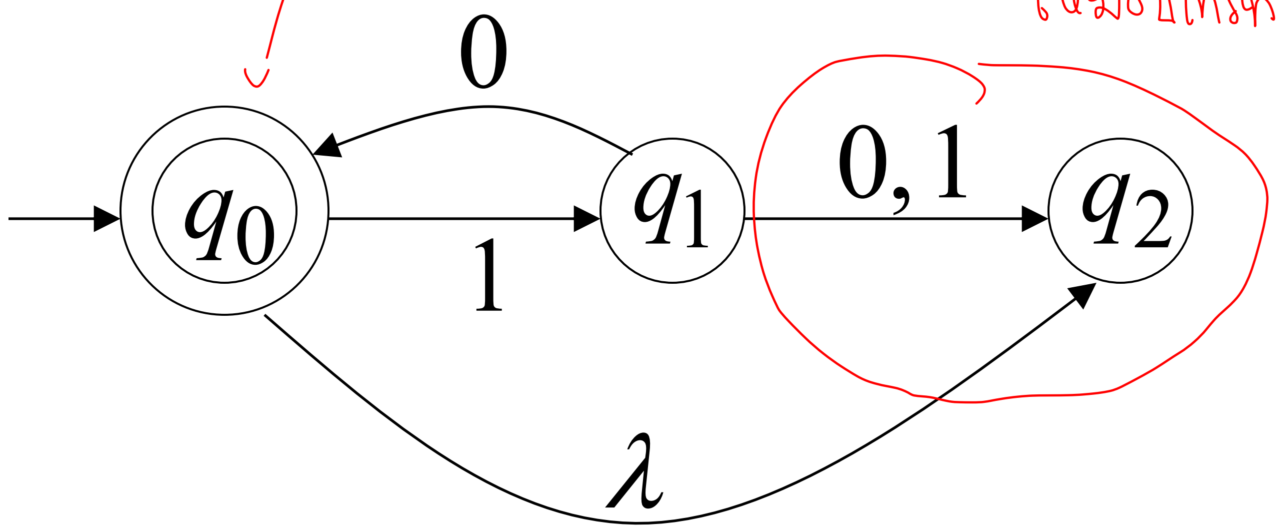


Another NFA Example

{λ, 10, 1010, ...}

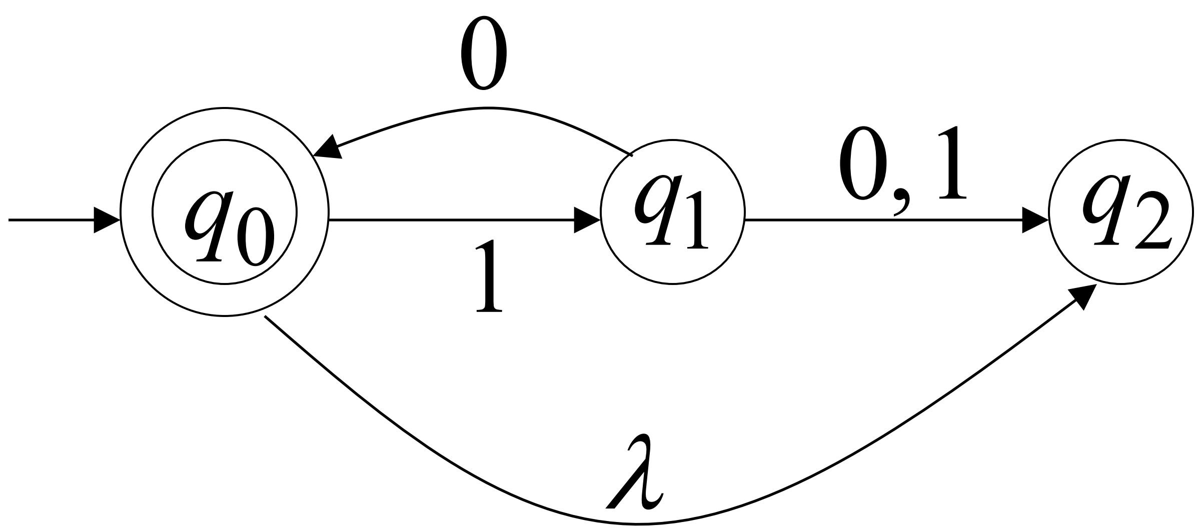
ที่สำคัญ

เหมือนการทำ trap state



Language accepted

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



(redundant state)

Remarks:

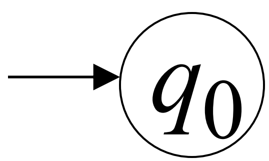
- The λ symbol never appears on the input tape

จริงๆแล้วเราจะไม่เห็น " " แต่แทนด้วย λ

- Simple automata:

M_1

ภาษา set ว่าง

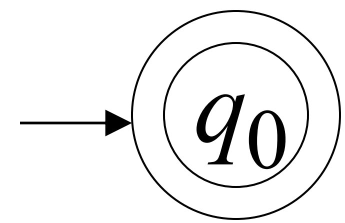


ไม่มี string ที่ยอมรับ

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

M_2

ภาษาที่มี string ว่าง

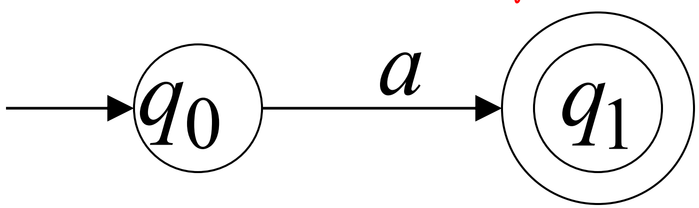


$$L(M_2) = \{\lambda\}$$

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

อธิบายได้ง่ายกว่า DFA → ซับซ้อนกว่า

NFA M_1



ละเลย trap state ได้

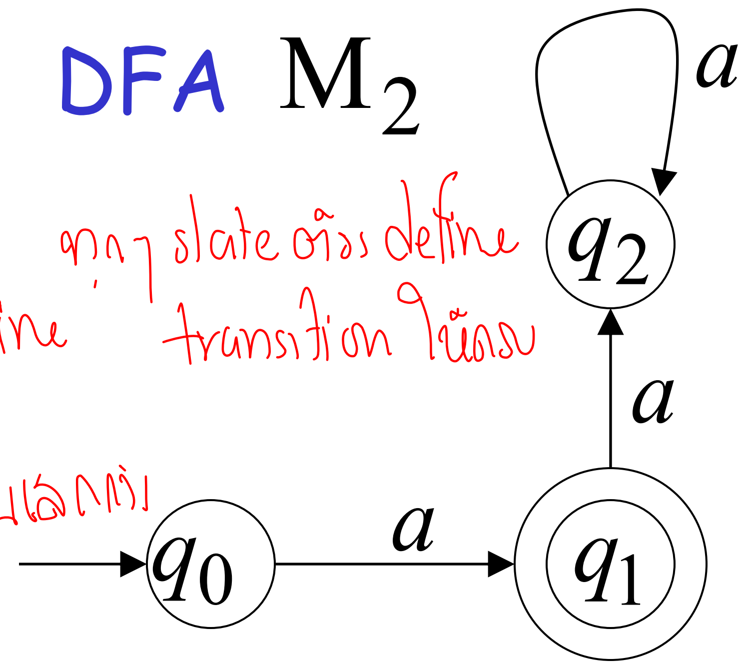
↓

กรณีไม่ได้ define ไว้

↓

ก็ไม่สนใจเลยก็ได้

DFA M_2



ทุกๆ state ต้อง define transition ให้ครบ

ถ้า aa → input เหลือ

↓

reject

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

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

Formal Definition of NFAs

เหมือนเป็น DFA

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

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

Σ : Input aplhabet, i.e. $\{a, b\}$

δ : Transition function ต่างแค่การอธิบายตัวนี้

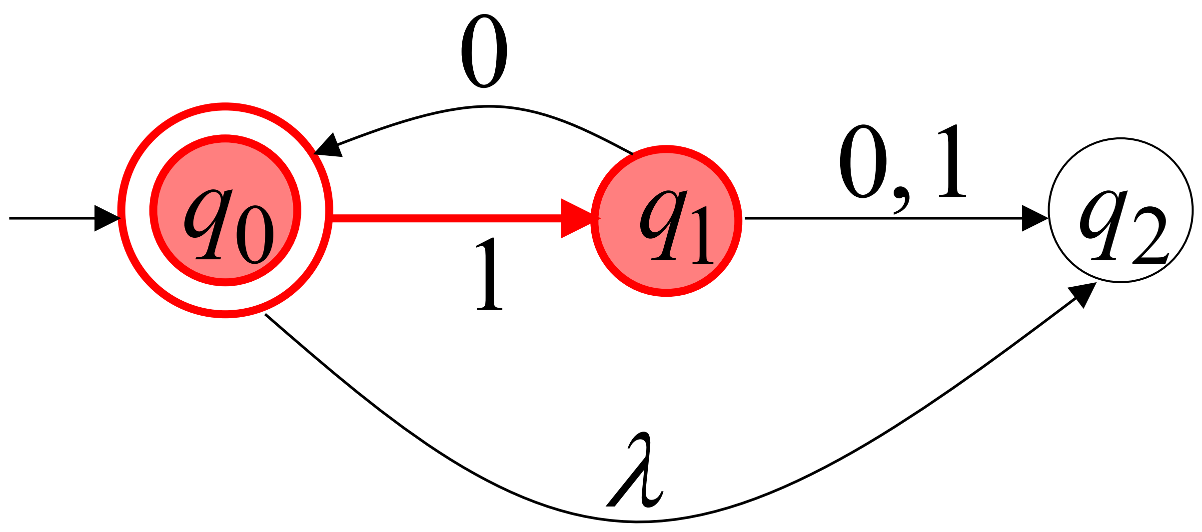
q_0 : Initial state

F : Final states

Transition Function δ

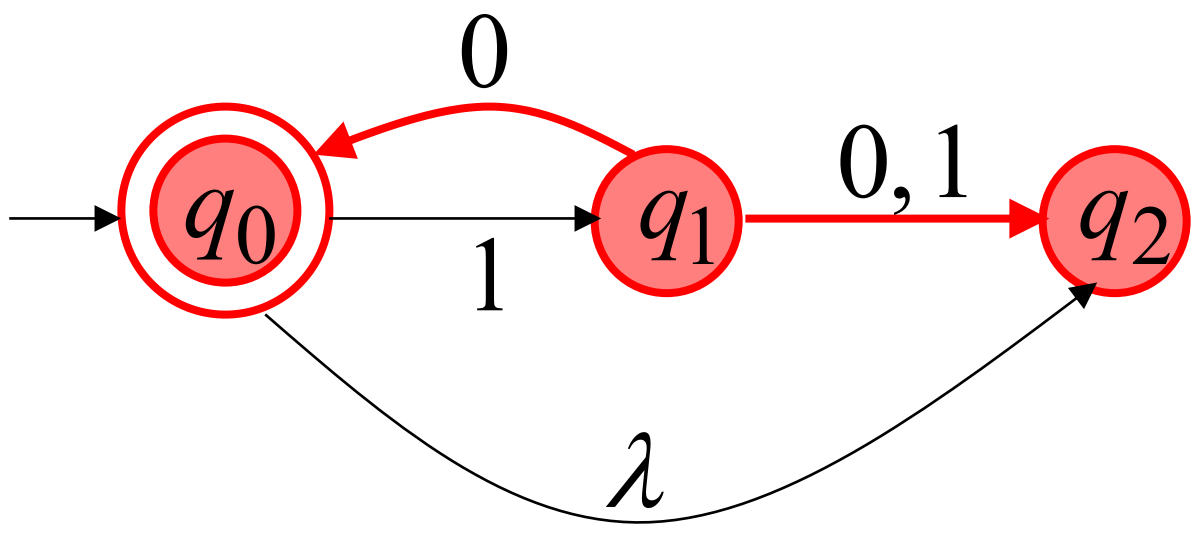
set vos state

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



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

ไปได้ ๒ ทางชั้นเอง

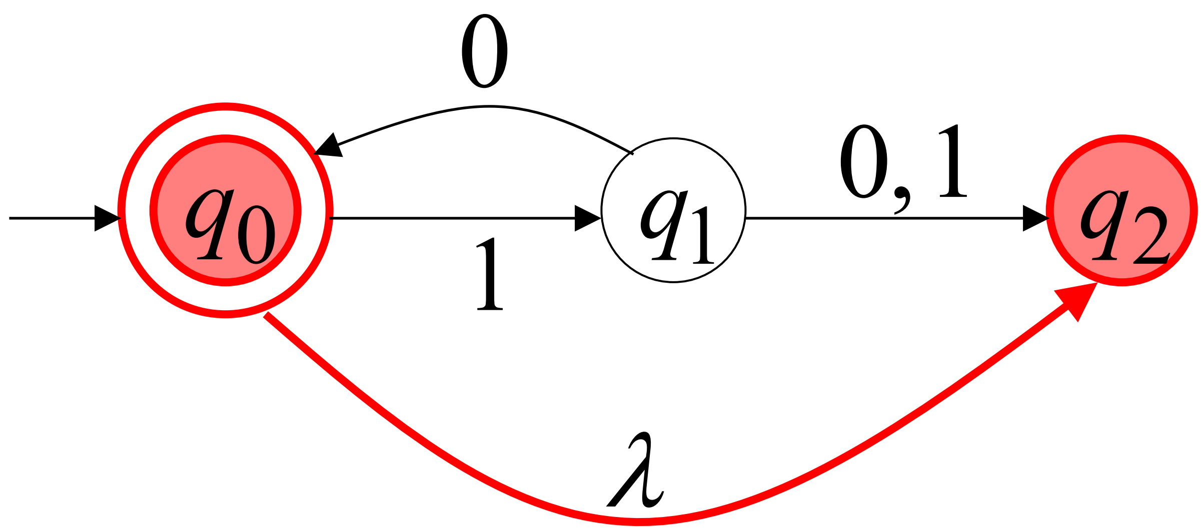


กรณีไม่อ่าน

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

กรณีอ่าน

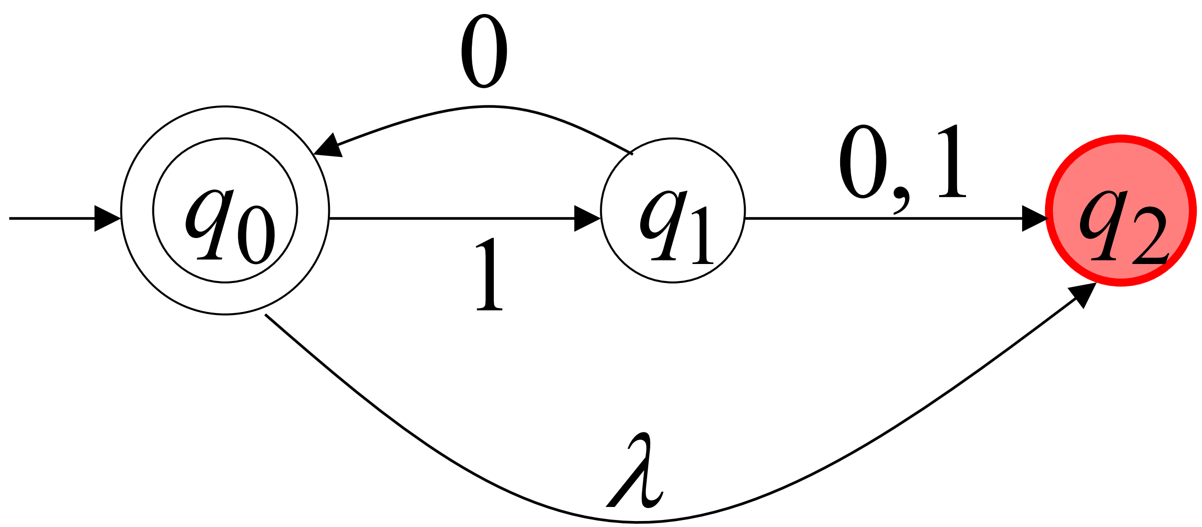
แล้วแต่ใจ



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

ไม่มีไปที่ไหนเลย

ไม่มี transition ใดๆเลย NFA จะเลยรับ

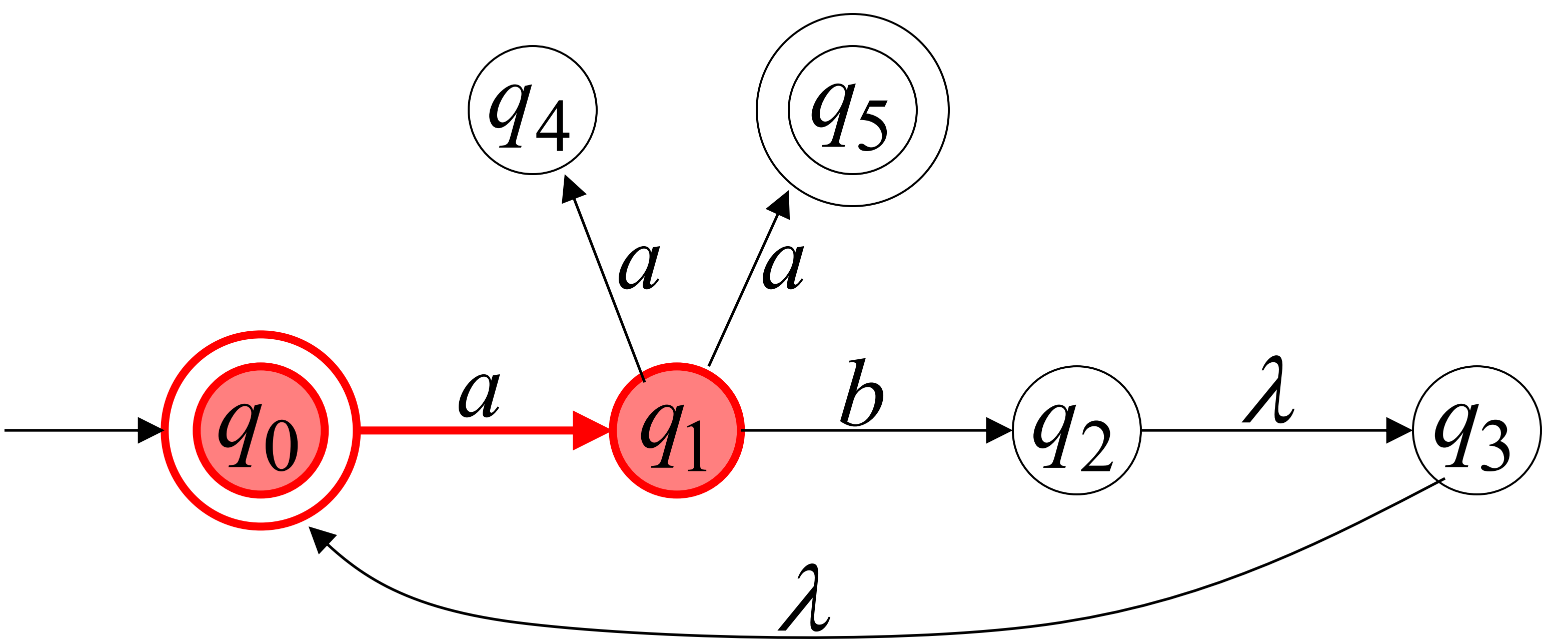


Extended Transition Function δ^*

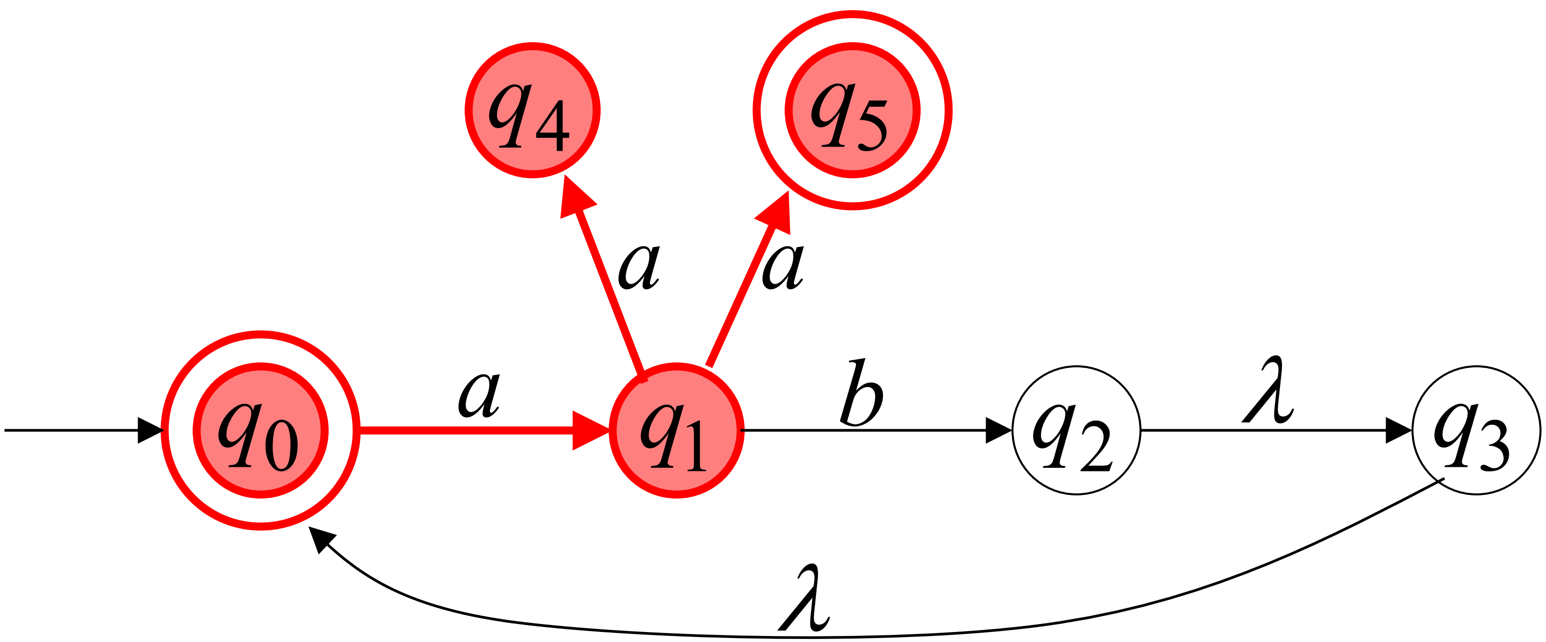
ที่ผ่านอะไรไปยาวเลย

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

string แทน alphabet

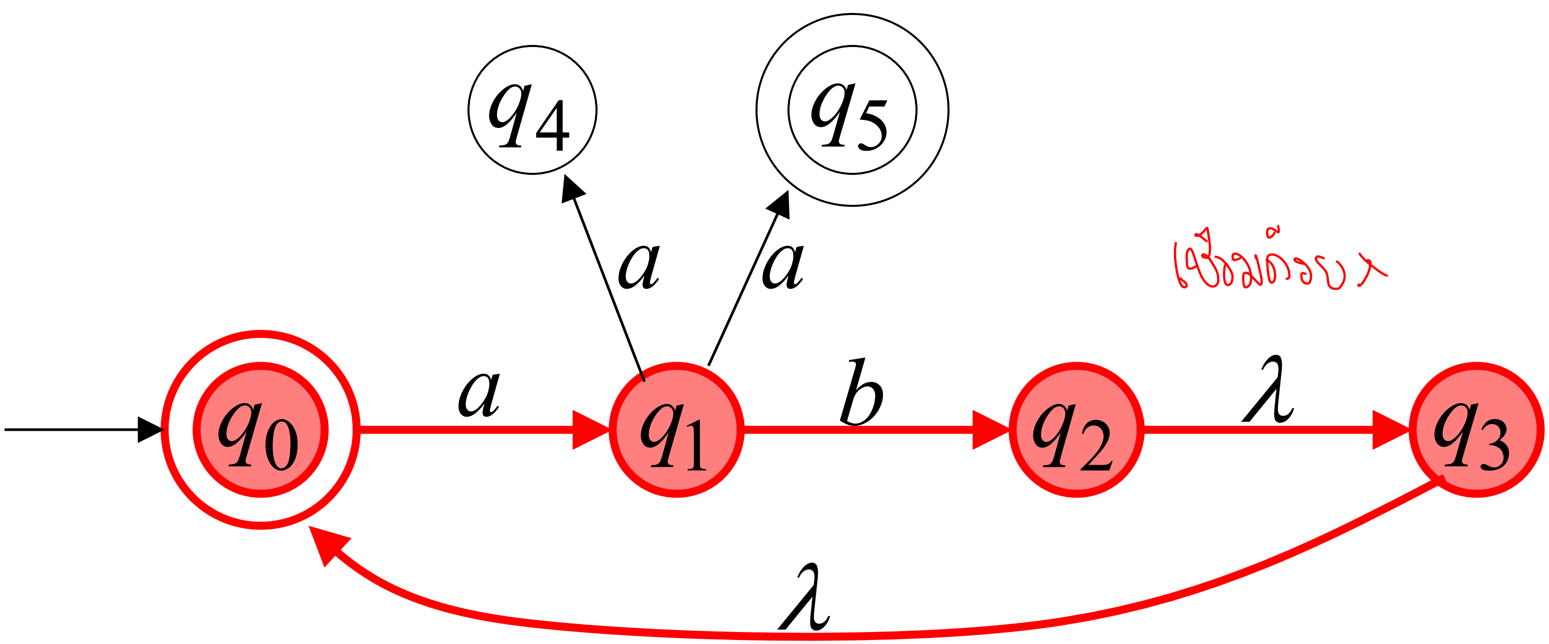


$$\delta^*(q_0, aa) = \{q_4, q_5\}$$



ทุกเส้นทาง possible

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

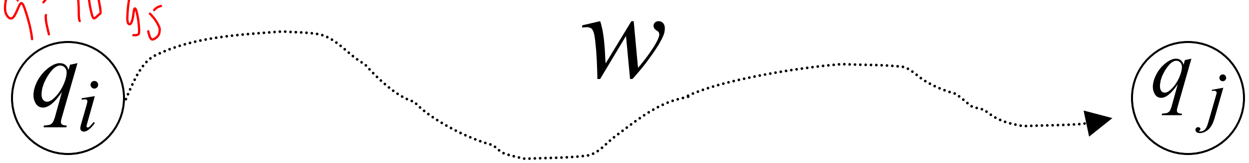


Formally

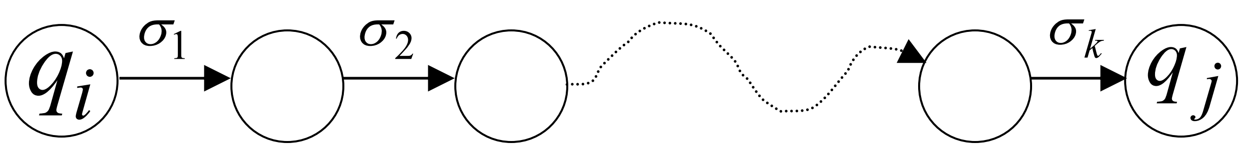
$q_j \in \delta^*(q_i, w)$: there is a walk from q_i to q_j with label w

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

มีเส้นทางจาก q_i ไป q_j

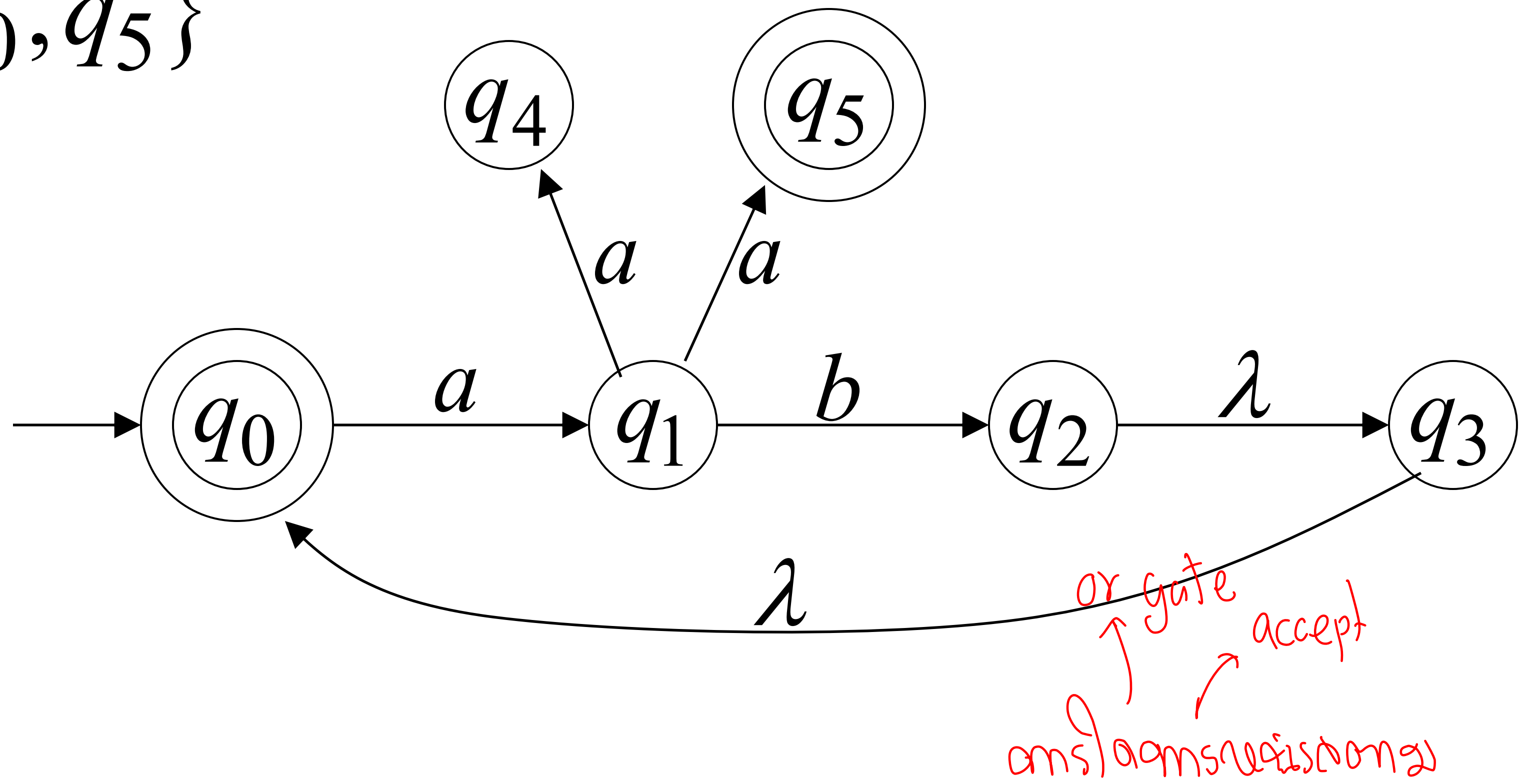


$$w = \sigma_1 \sigma_2 \cdots \sigma_k$$



The Language of an NFA M

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

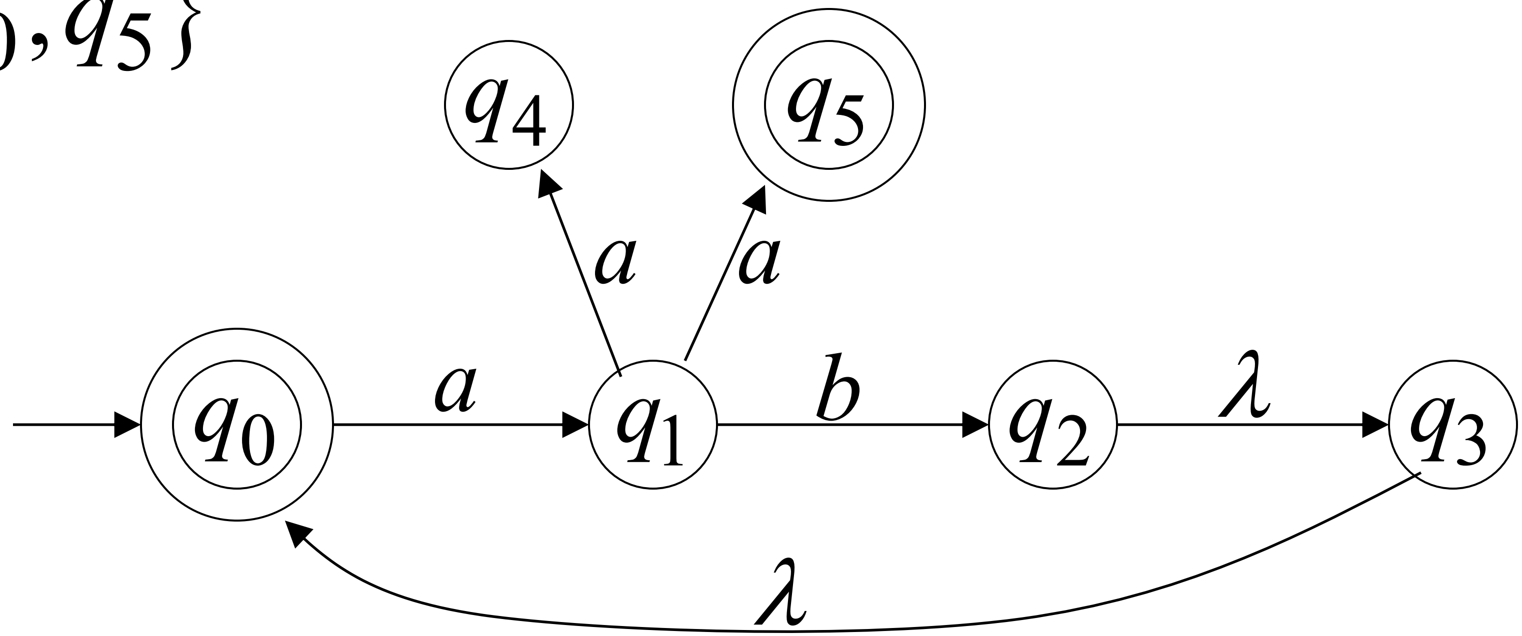


$$\delta^*(q_0, aa) = \{q_4, \underline{q_5}\}$$

$\longrightarrow \in F$

$$\underline{aa \in L(M)}$$

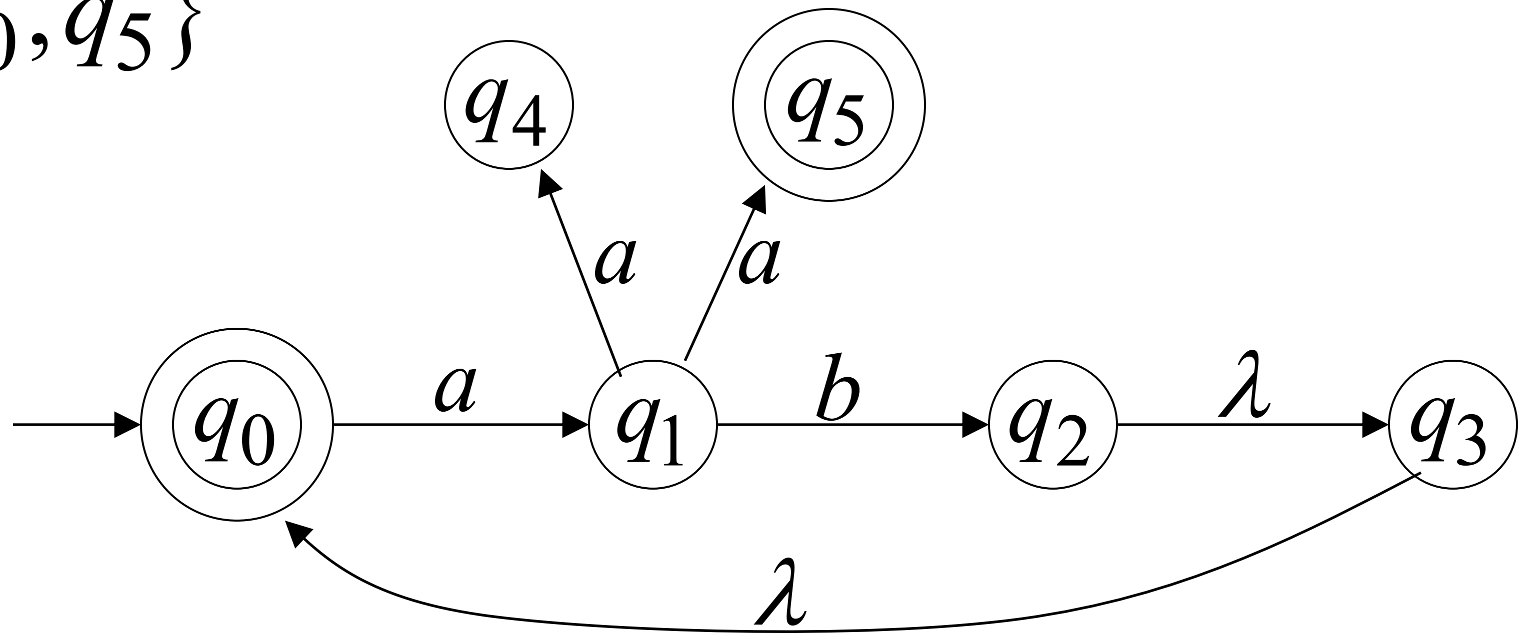
$$F = \{q_0, q_5\}$$



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

$$\searrow \in F$$

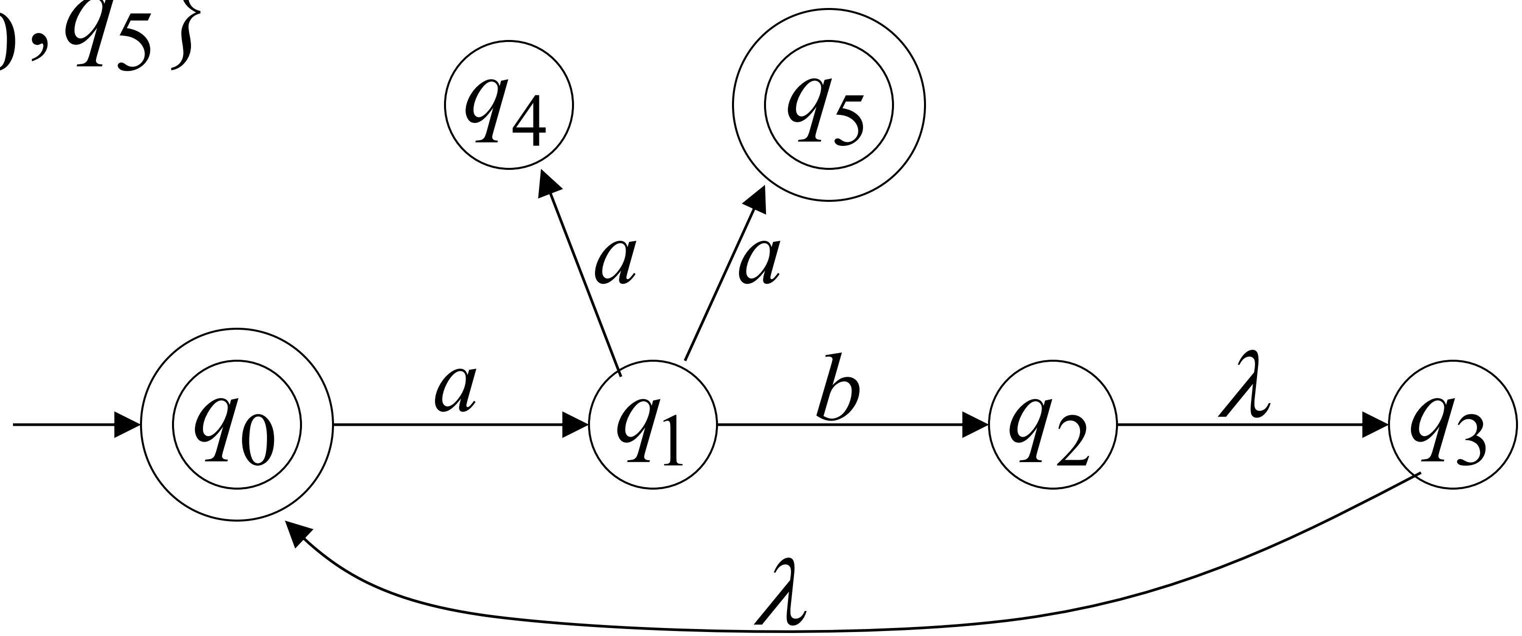
$F = \{q_0, q_5\}$



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

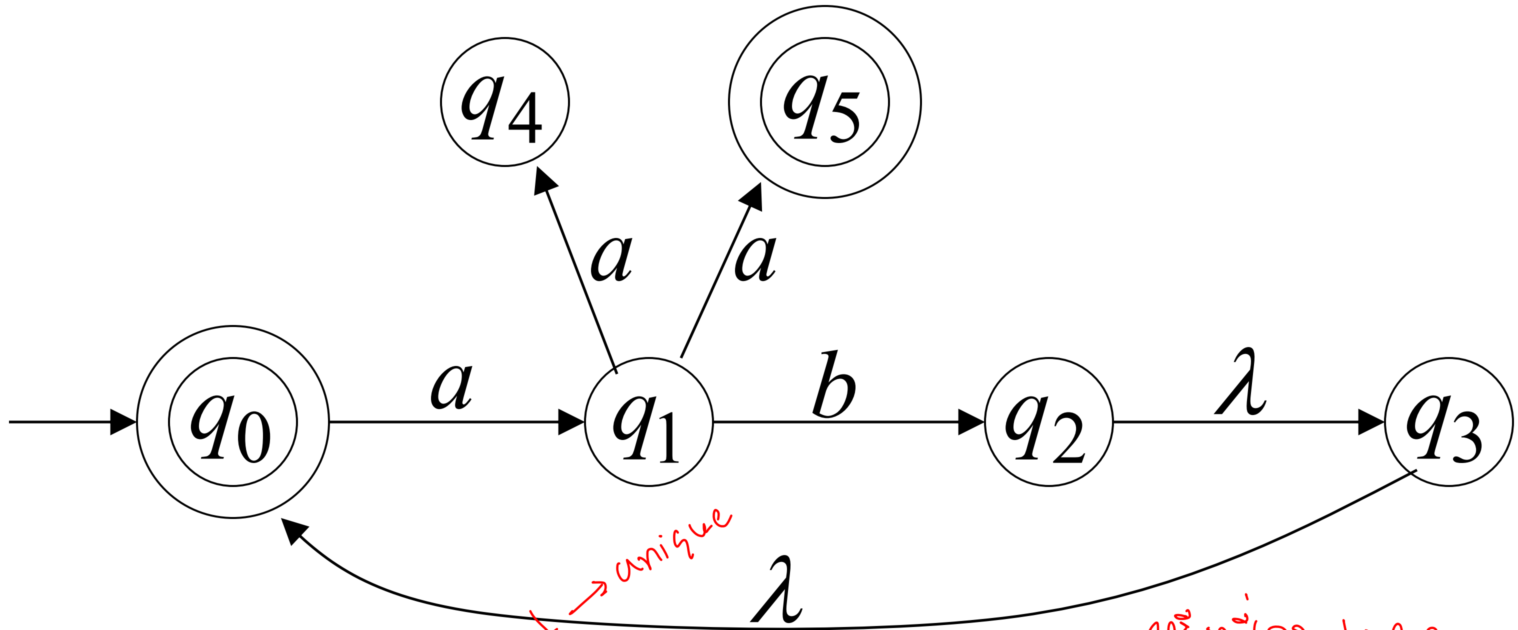
$\searrow \in F$

$F = \{q_0, q_5\}$



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

$aba \notin L(M)$



set → unique

loop

หรือที่เจอจะต่อ aa

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

ภาษาที่สั้นที่สุด

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

การconcat

Formally

The language accepted by NFA M is:

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

มีแค่อันเดียวก็พอแล้ว

ฟังก์ชัน

any string

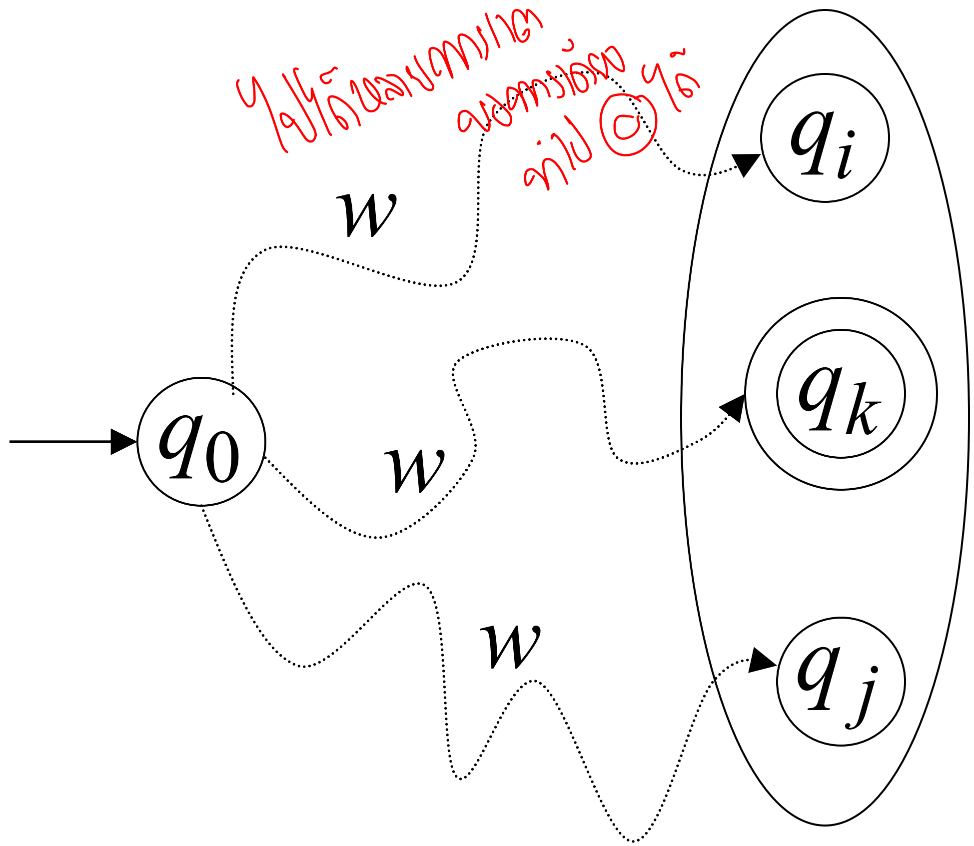
where $\delta^*(q_0, w_m) = \{q_i, q_j, \ldots, q_k, \ldots\}$

เริ่มจาก initial

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

$$w \in L(M)$$

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



$$q_k \in F$$

NFAs accept the Regular Languages ?

Equivalence of Machines

Definition for Automata:

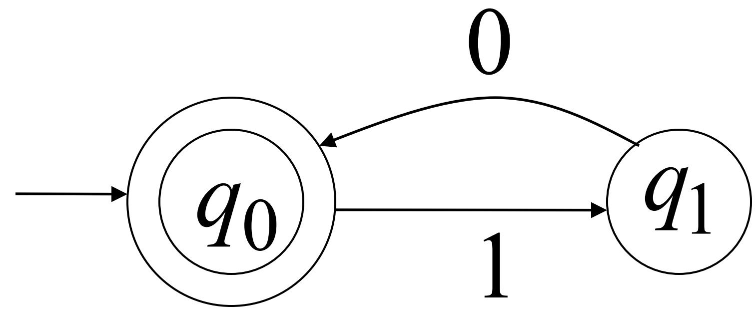
Machine M_1 is equivalent to machine M_2

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

Example of equivalent machines

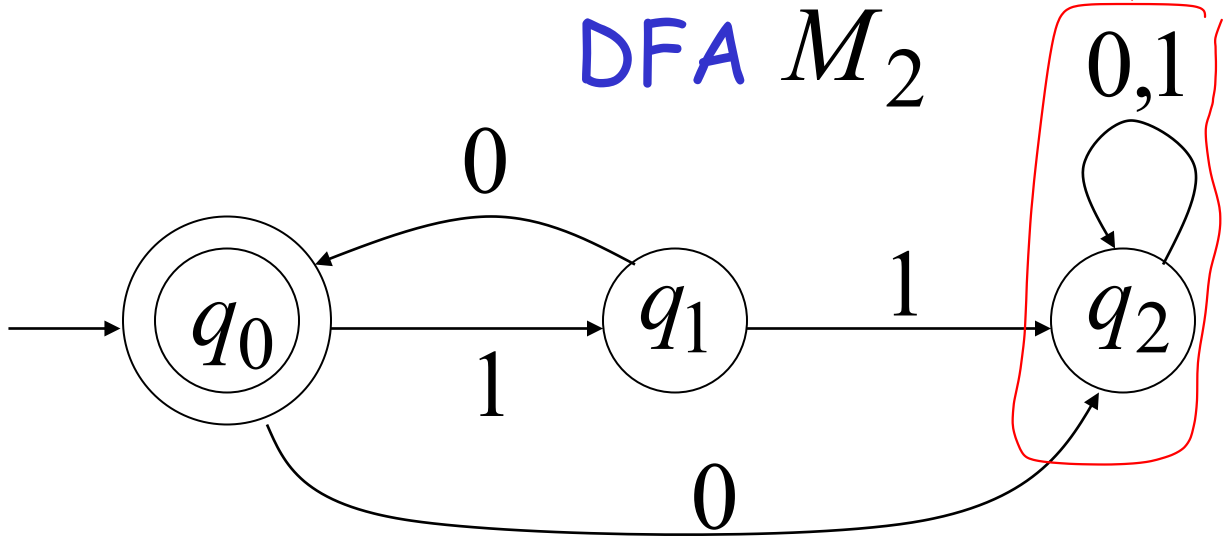
NFA M_1

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



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 ที่สามารถสร้าง DFA ขึ้นมารองรับได้

Languages
accepted
by DFAs

NFAs and DFAs have the same computation power

Step 1

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

Proof: Every DFA is trivially an NFA

ไม่เป็นสาระ

DFA: $\delta(q_i, w) = q_j$

NFA: $\delta(q_i, a) = \{q_k, q_j\}$

$\Downarrow$

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

DFA $\subseteq$ NFA

DFA $Q \times \Sigma \to Q$

NFA $Q \times \Sigma \to 2^Q$

จริง $Q \subseteq 2^Q$ (แน่นอน

Step 2

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

Proof: Any NFA can be converted to an equivalent DFA

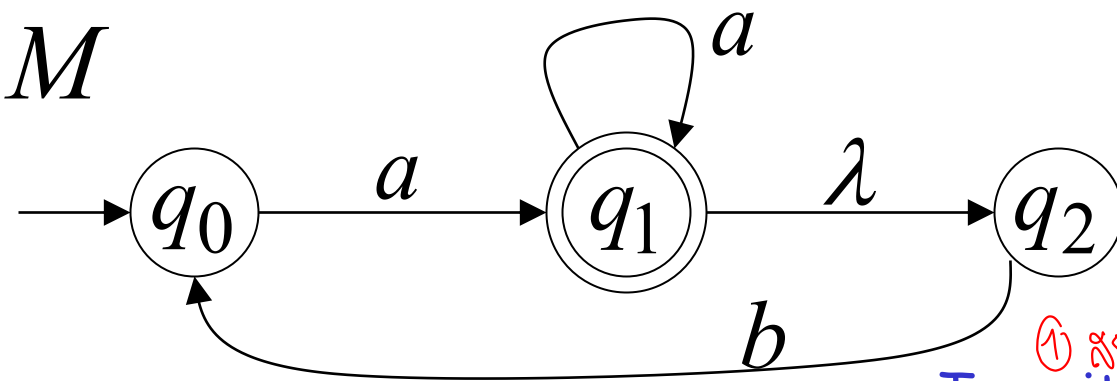
เราจะแปลง NFA → DFA

⇓

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

Convert NFA to DFA

NFA M

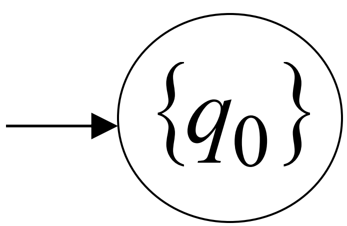


① สร้าง table transition ของ

Transition table for NFA M

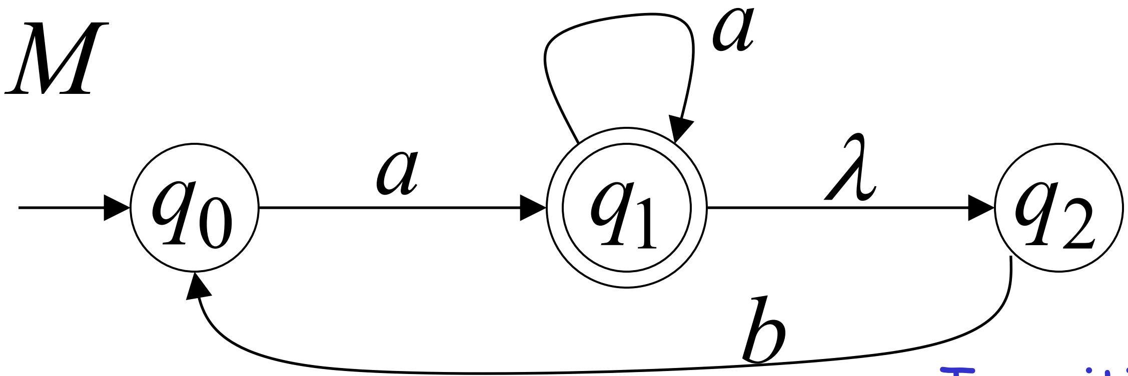
	a	b
q0	{q1, q2}	Ø
q1	{q1, q2}	{q0}
q2	Ø	{q0}

DFA M'



Convert NFA to DFA

NFA M

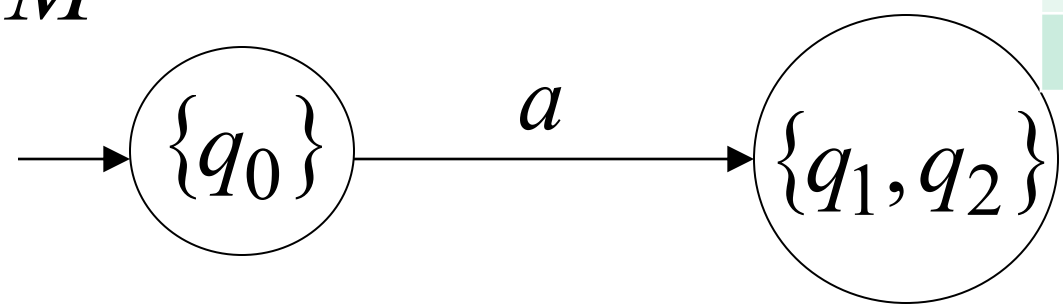


เราค่อยๆ สร้างตาม transition table

Transition table for NFA M

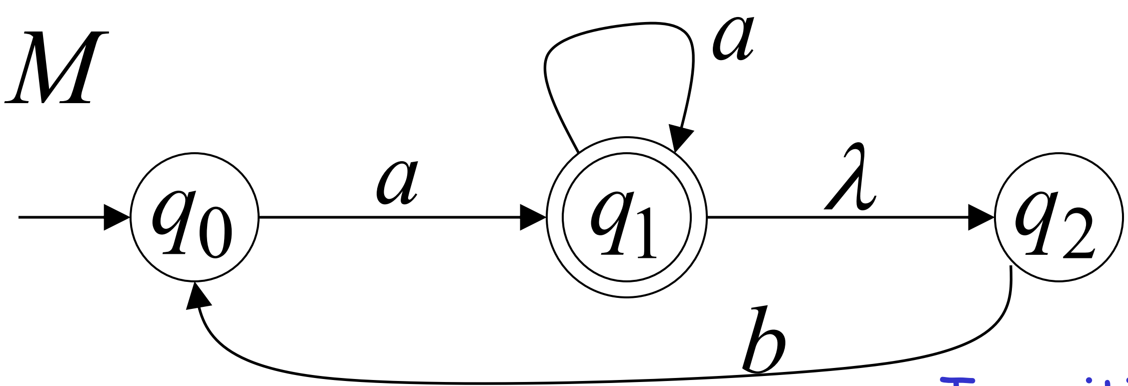
	a	b
q0	{q1, q2}	∅
q1	{q1, q2}	{q0}
q2	∅	{q0}

DFA M'



Convert NFA to DFA

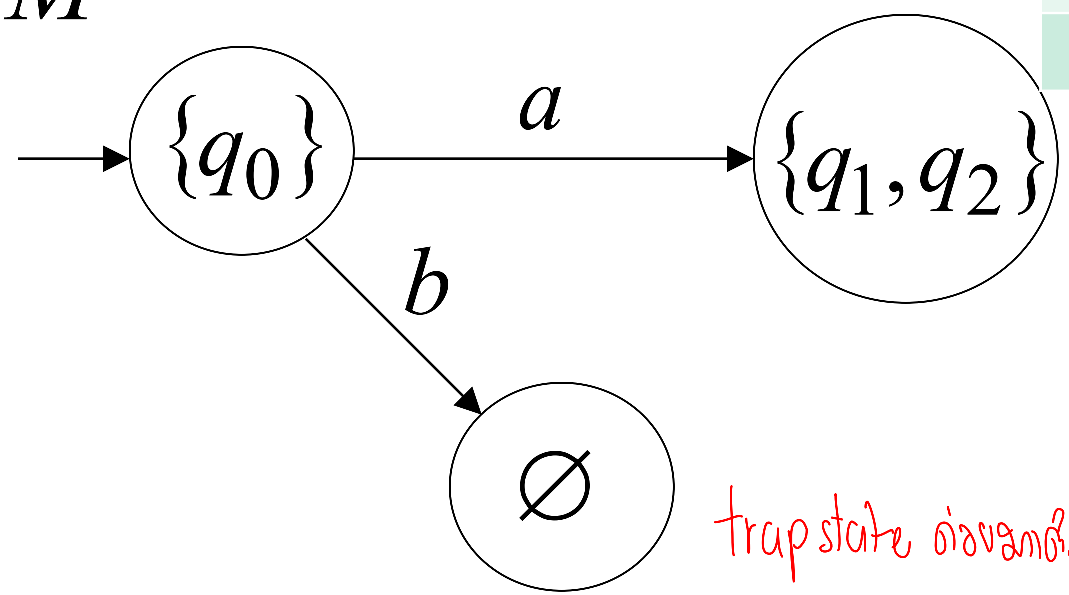
NFA M



Transition table for NFA M

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

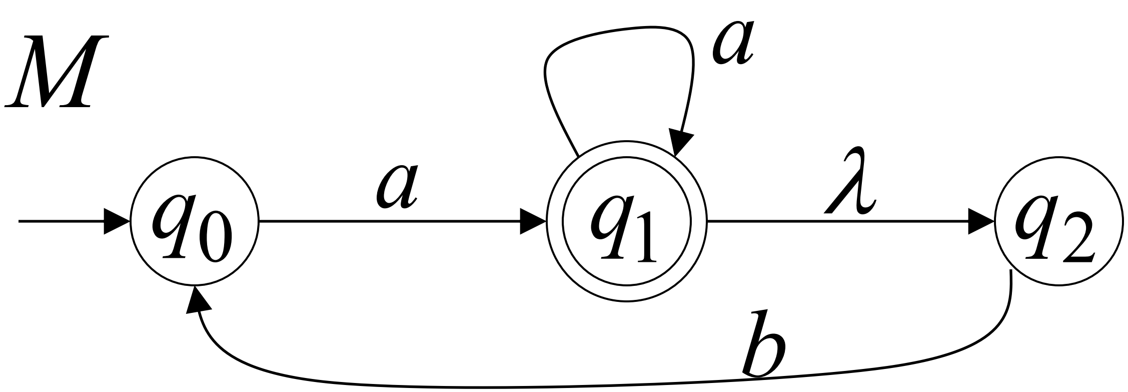
DFA M'



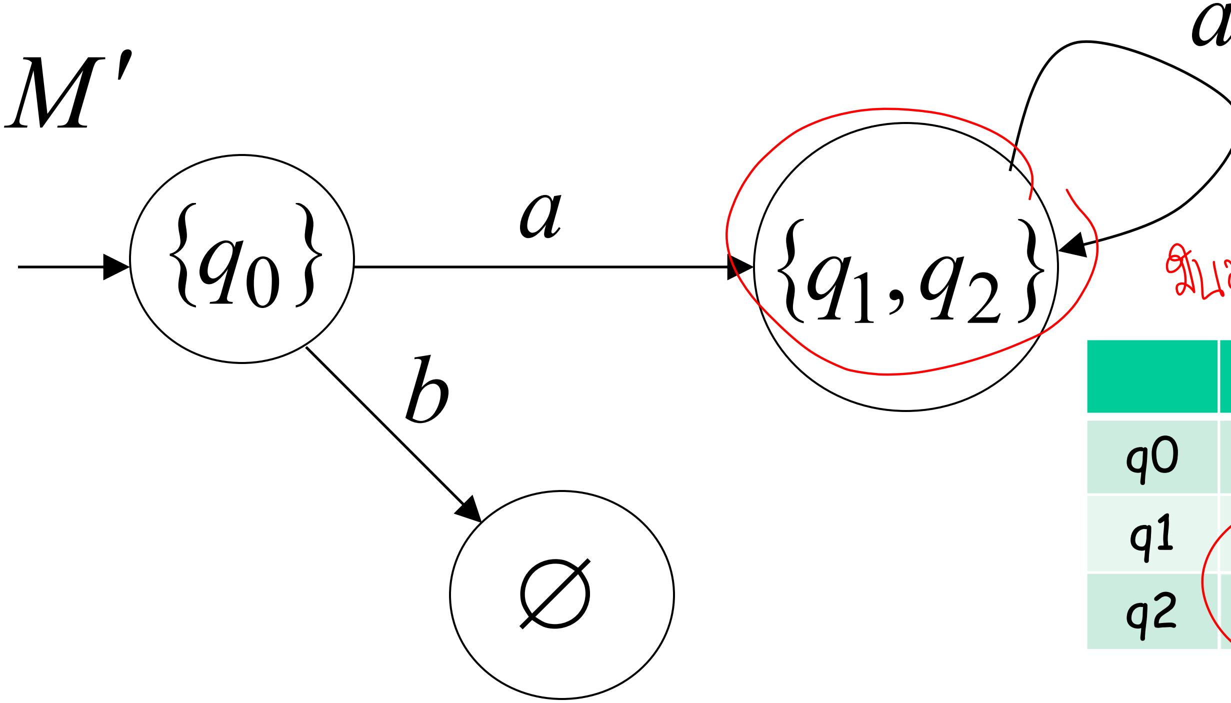
trap state ต้องมีด้วย

Convert NFA to DFA

NFA M



DFA M'

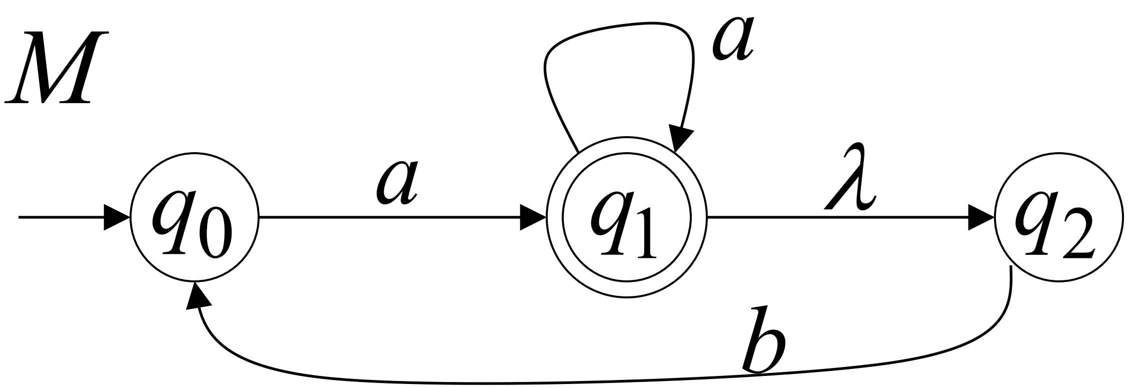


มันอยู่เป็นคู่ เลยได้ไม่เป็นคู่

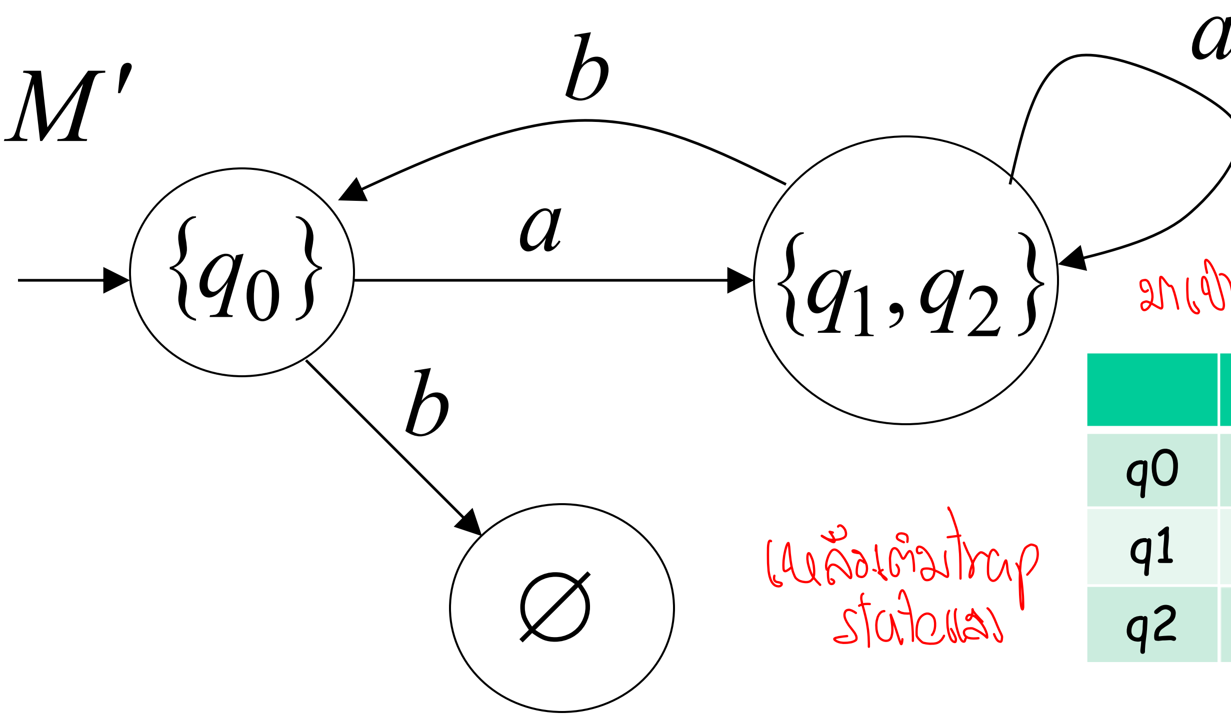
	a	b
q0	{q1, q2}	ø
q1	{q1,q2}	{q0}
q2	ø	{q0}

Convert NFA to DFA

NFA M



DFA M'



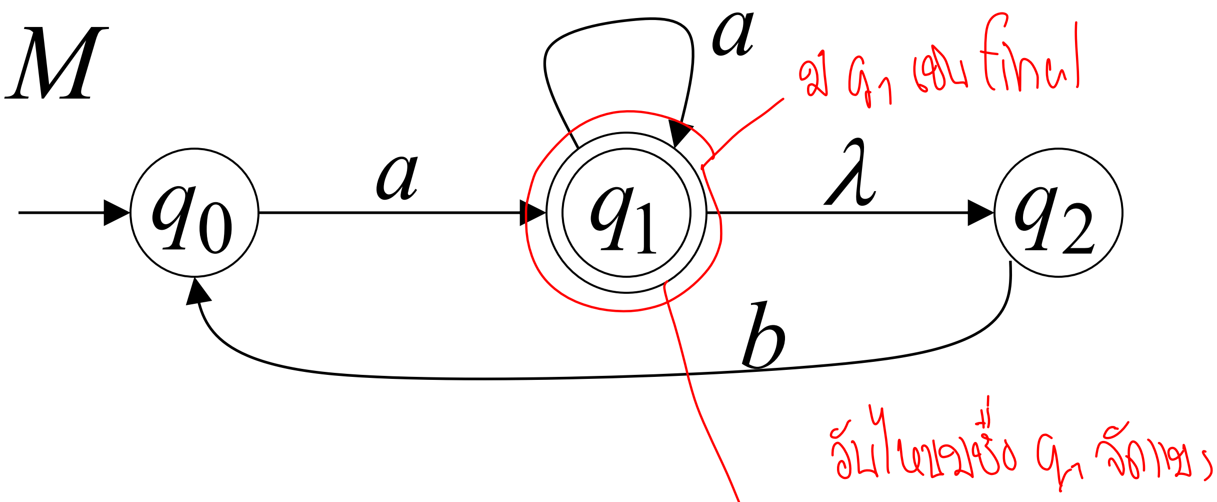
มาเป็นคู่ดูเป็นคู่นคง

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

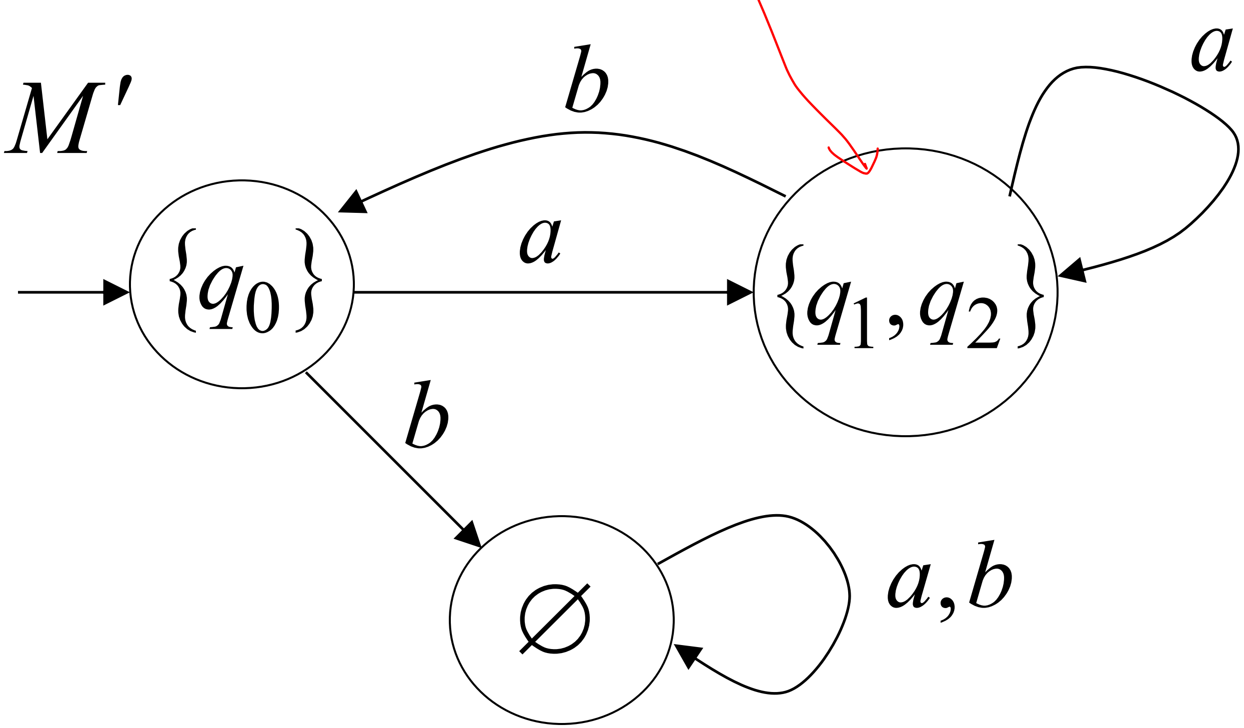
แล้วเติมtrap stateแลง

Convert NFA to DFA

NFA M

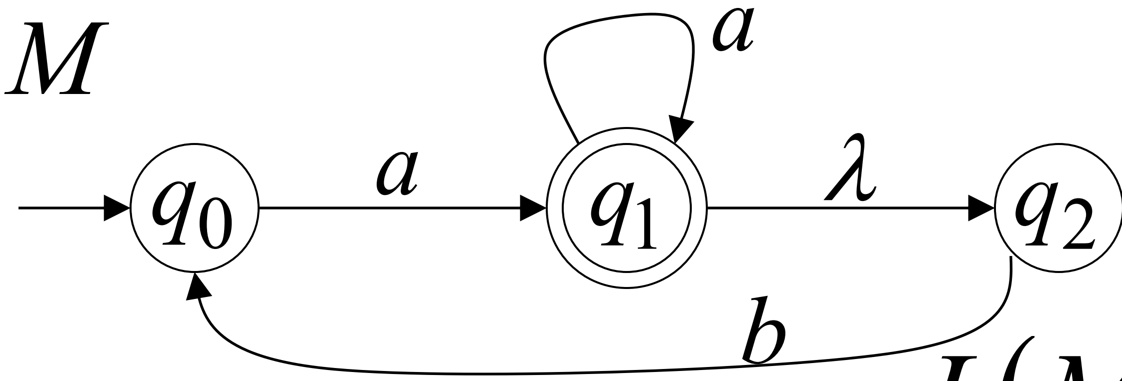


DFA M'



Convert NFA to DFA

NFA M



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

DFA M'