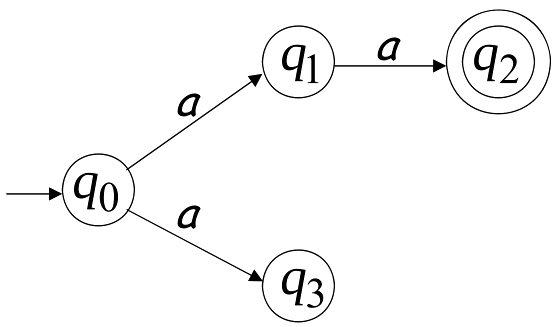
Non Deterministic Automata

1

Nondeterministic Finite Accepter (NFA)

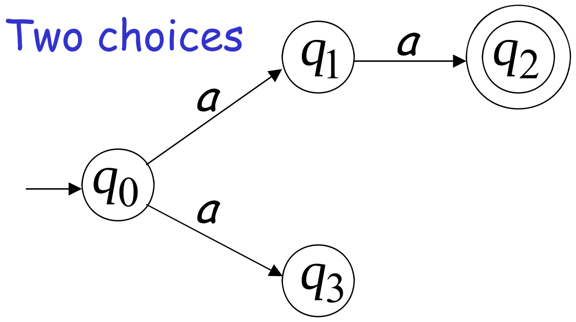
Alphabet = $\{a\}$



2

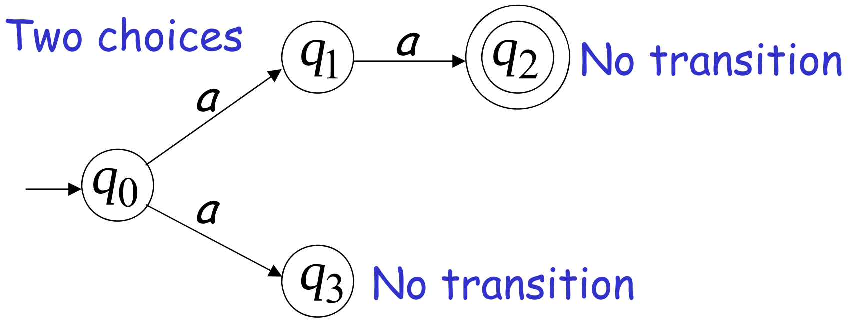
Nondeterministic Finite Accepter (NFA)

Alphabet = $\{a\}$

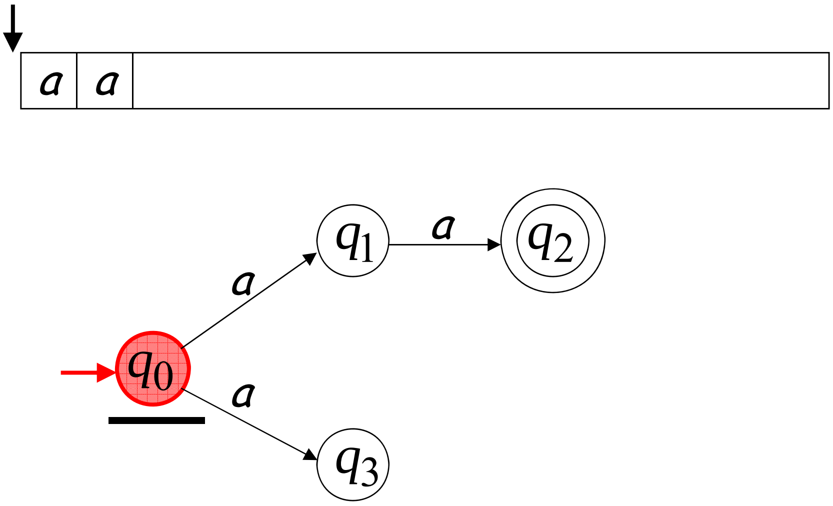


Nondeterministic Finite Accepter (NFA)

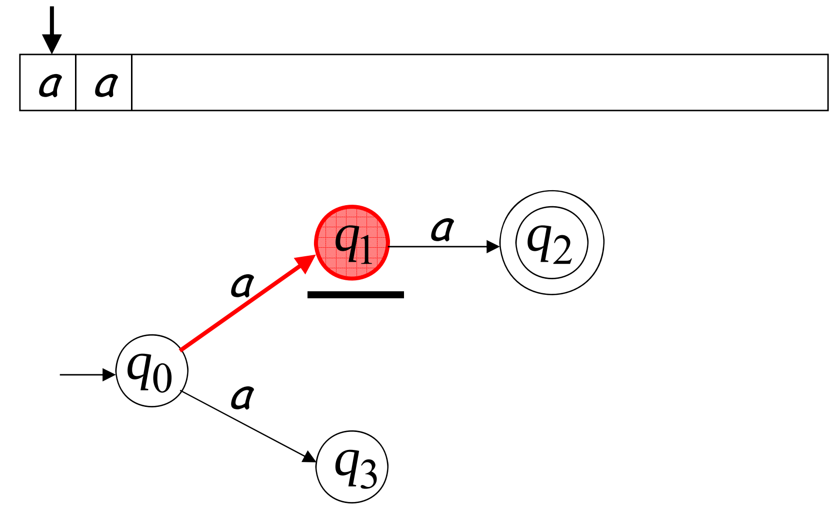
Alphabet = $\{a\}$



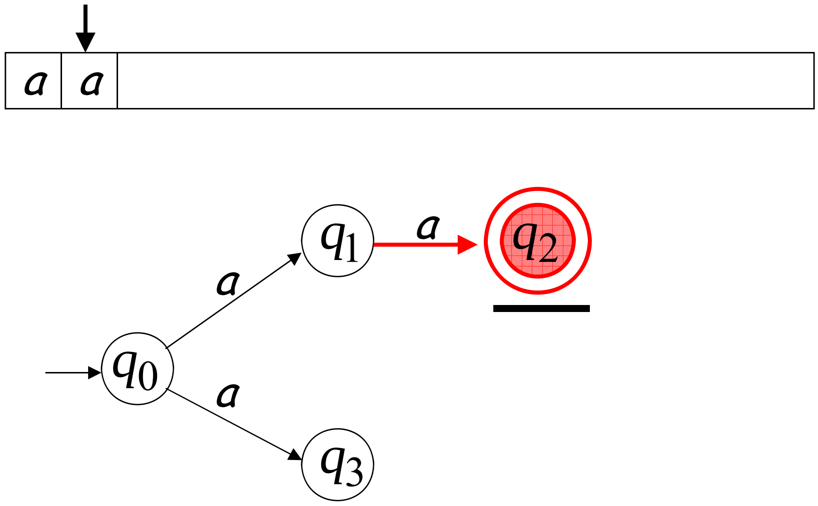
First Choice



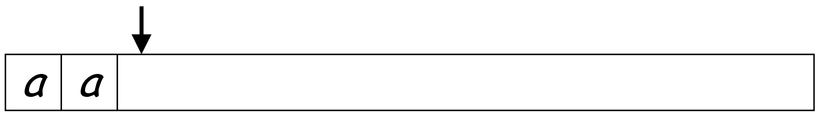
First Choice



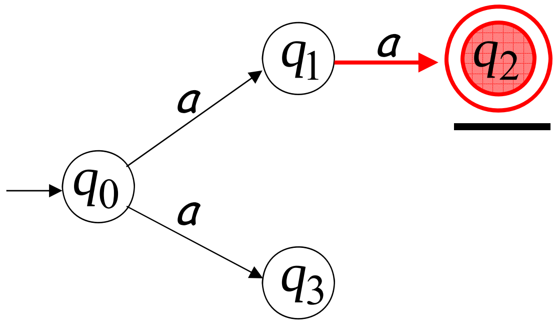
First Choice



First Choice

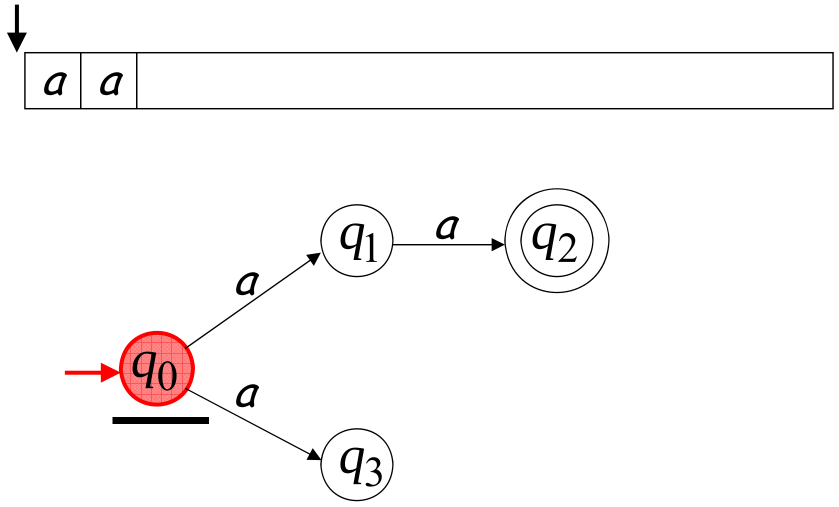


All input is consumed



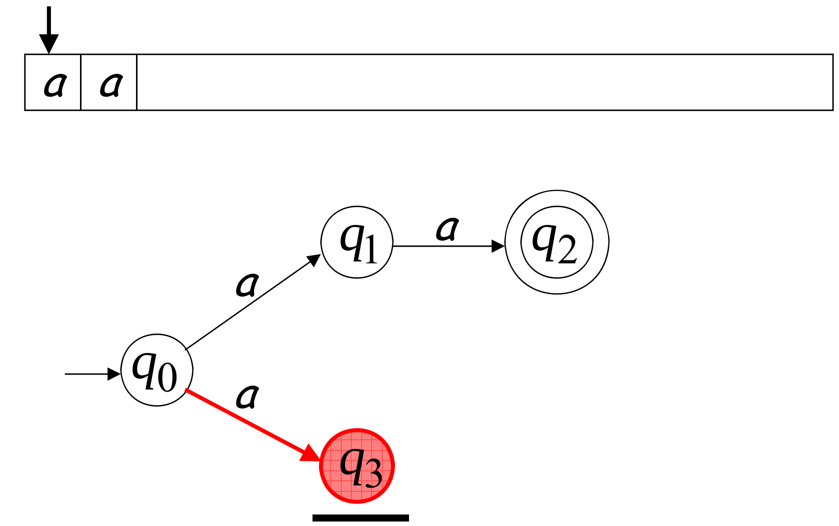
"accept"

Second Choice



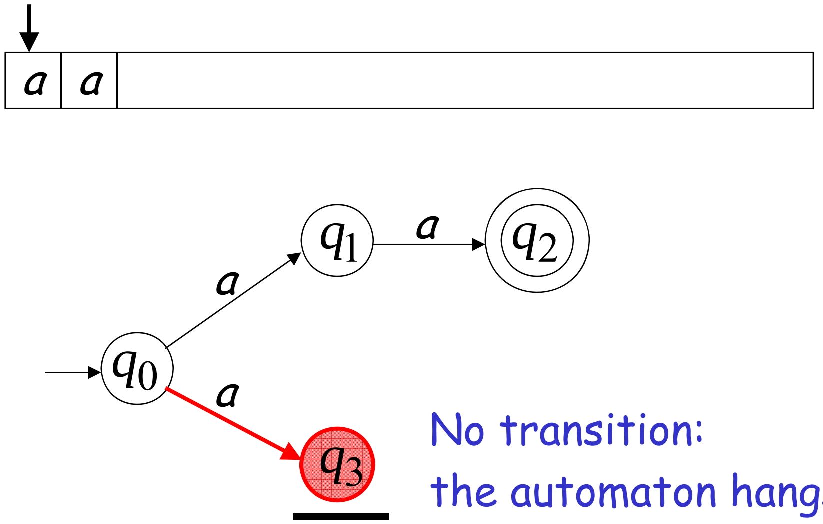
9

Second Choice



10

Second Choice

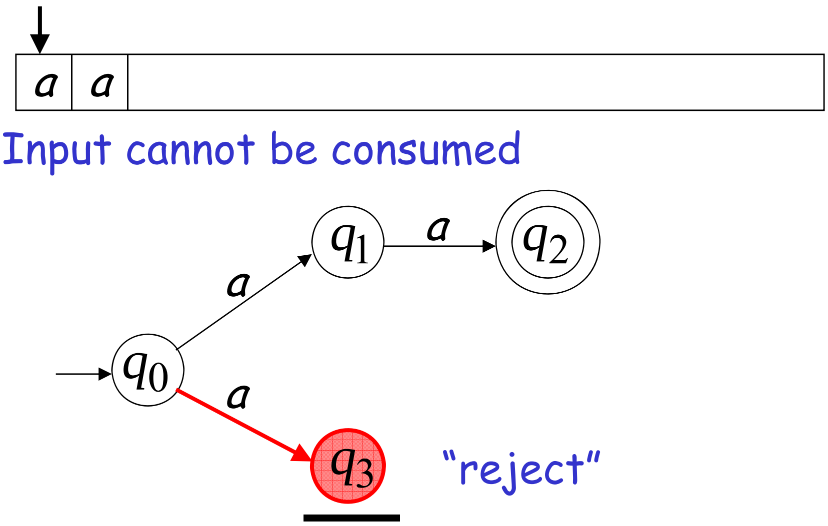


No transition:
the automaton hangs

11

Second Choice

Input cannot be consumed



"reject"

12

An NFA accepts a string:

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

AND

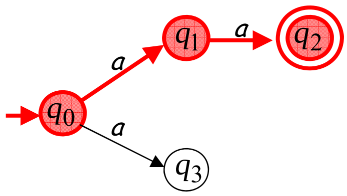
all the input is consumed and the automaton is in a final state

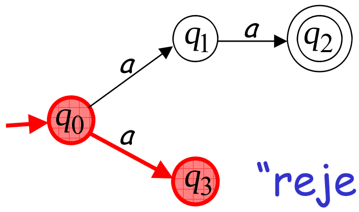
13

Example

aa is accepted by the NFA:

"accept"



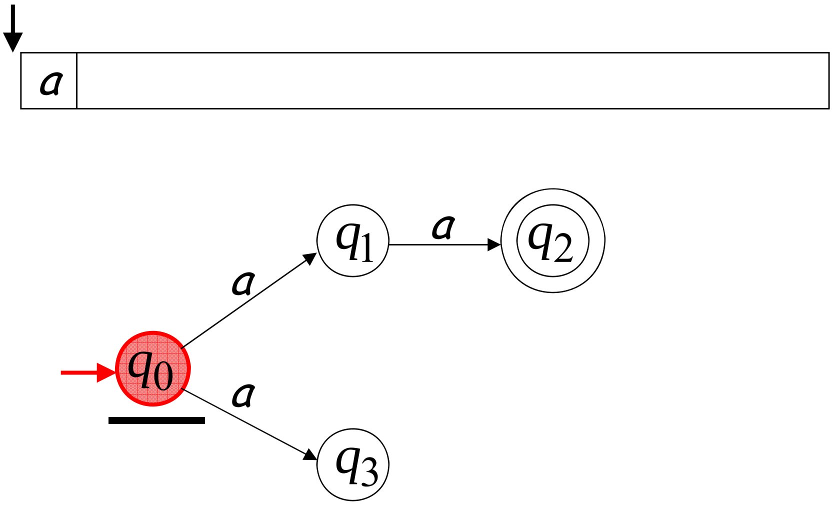


"reject"

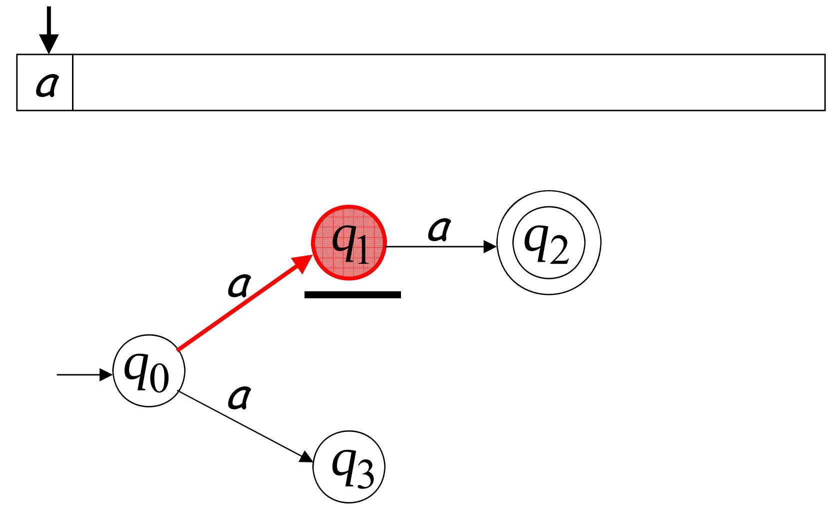
because this computation accepts aa

14

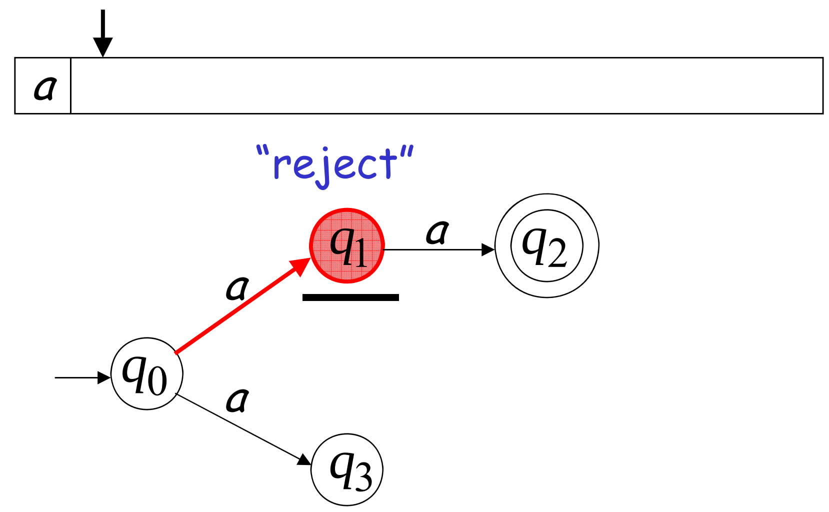
Rejection example



First Choice

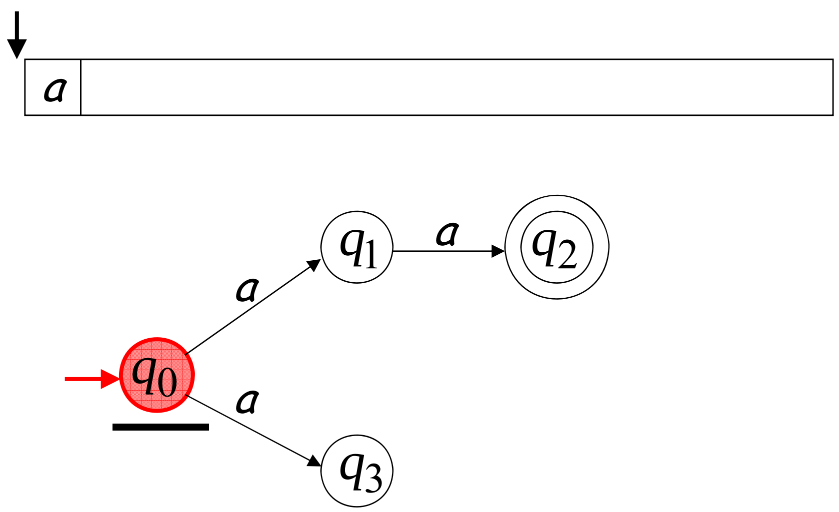


First Choice



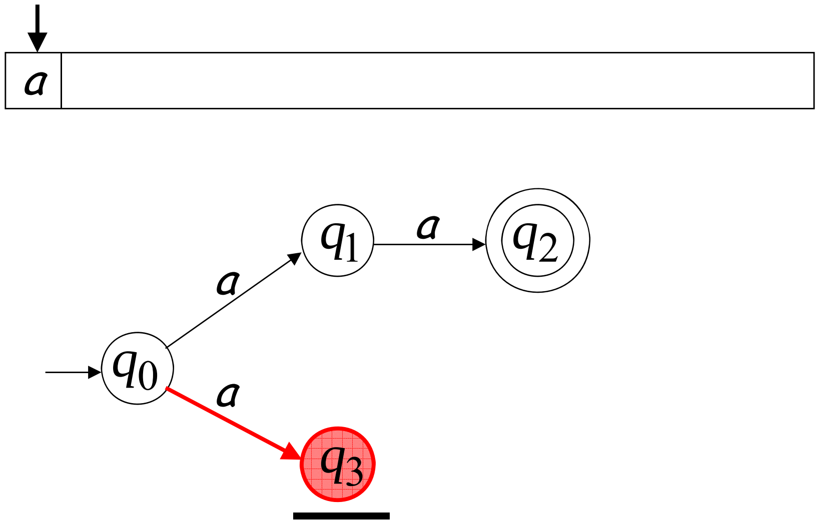
17

Second Choice



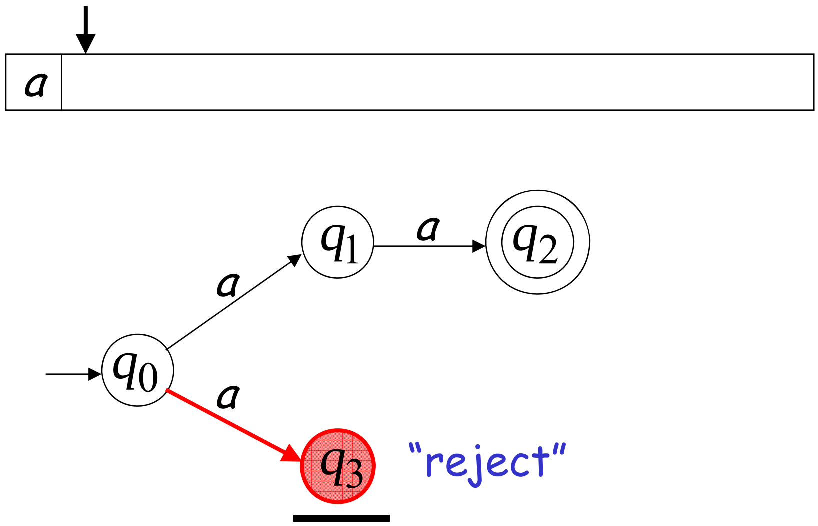
18

Second Choice



19

Second Choice



20

An NFA rejects a string:

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

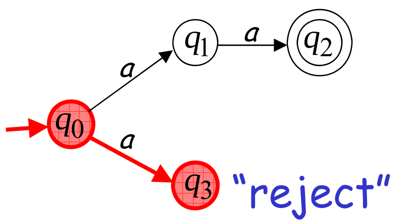
- All the input is consumed and the automaton is in a non final state

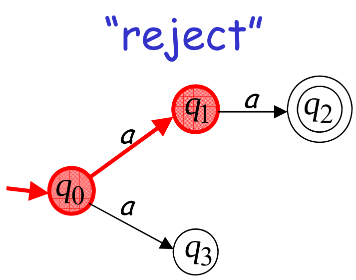
OR

- The input cannot be consumed

Example

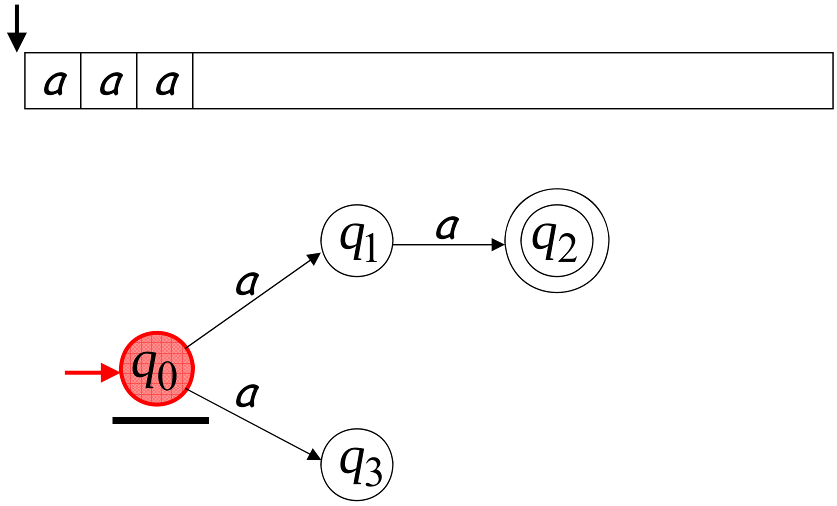
a is rejected by the NFA:





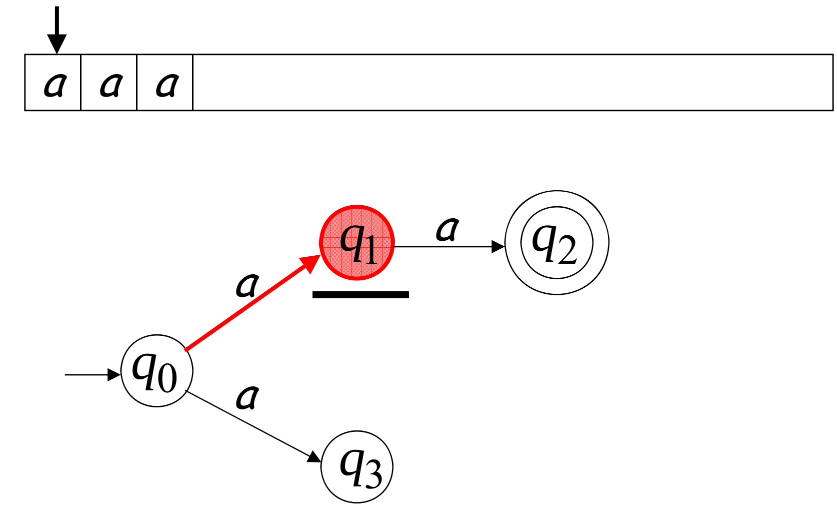
All possible computations lead to rejection

Rejection example



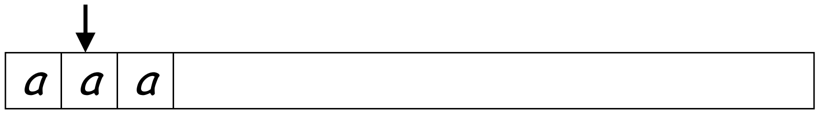
23

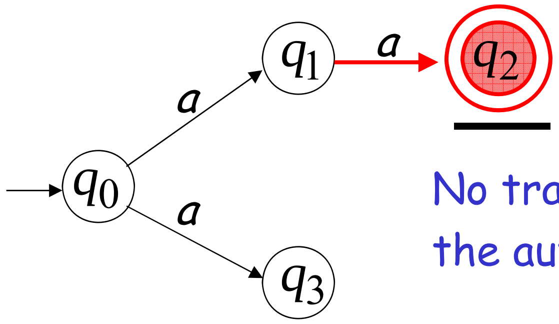
First Choice



24

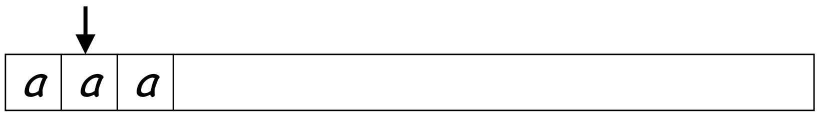
First Choice



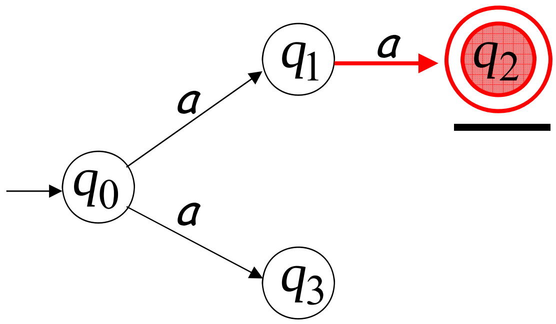


No transition:
the automaton hangs

First Choice

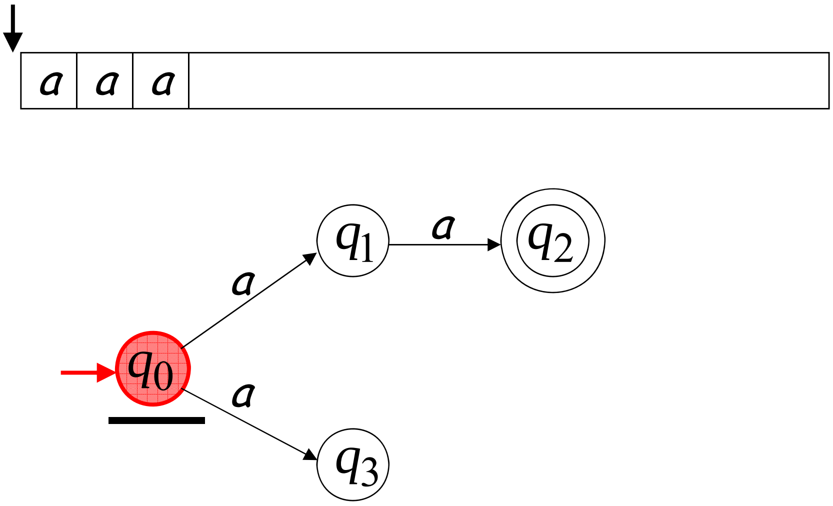


Input cannot be consumed



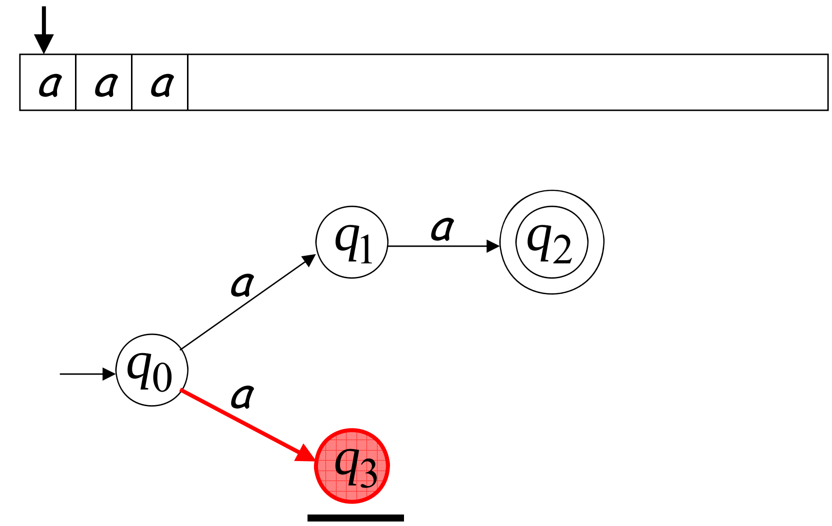
"reject"

Second Choice



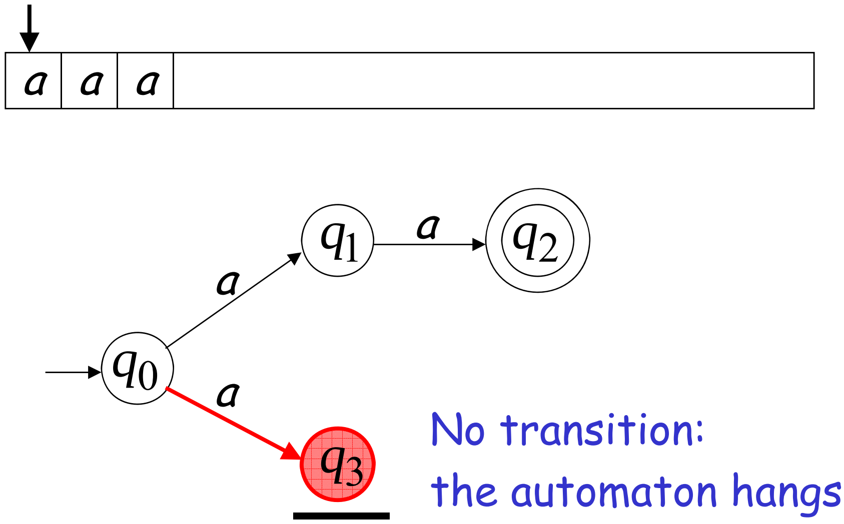
27

Second Choice

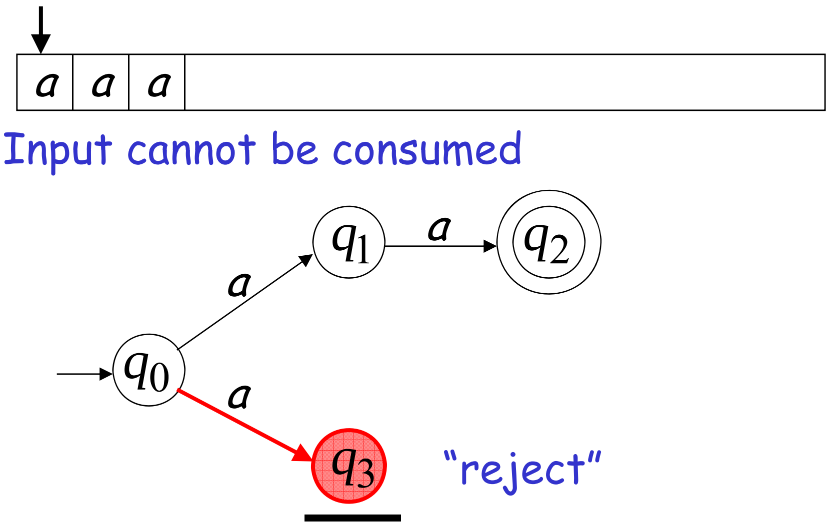


28

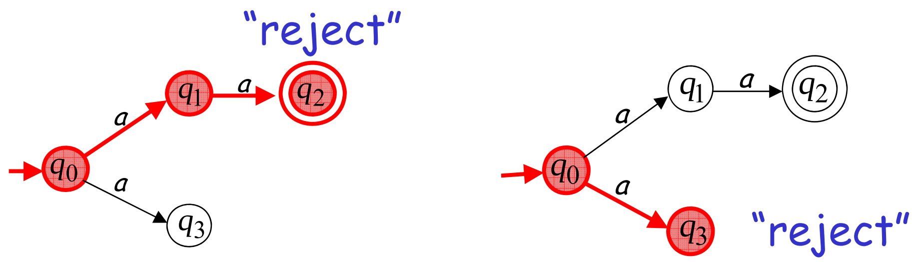
Second Choice



Second Choice



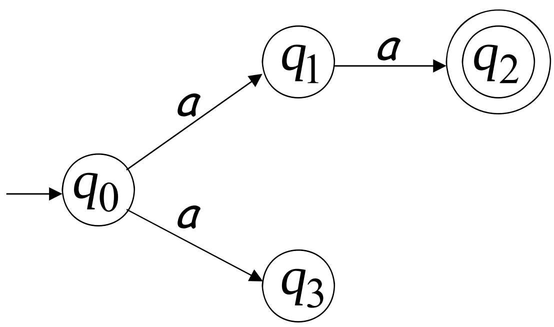
aaa is rejected by the NFA:



All possible computations lead to rejection

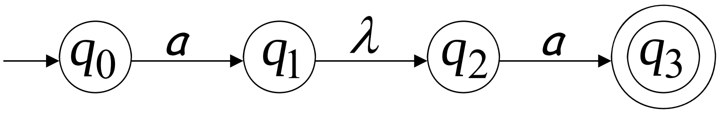
31

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

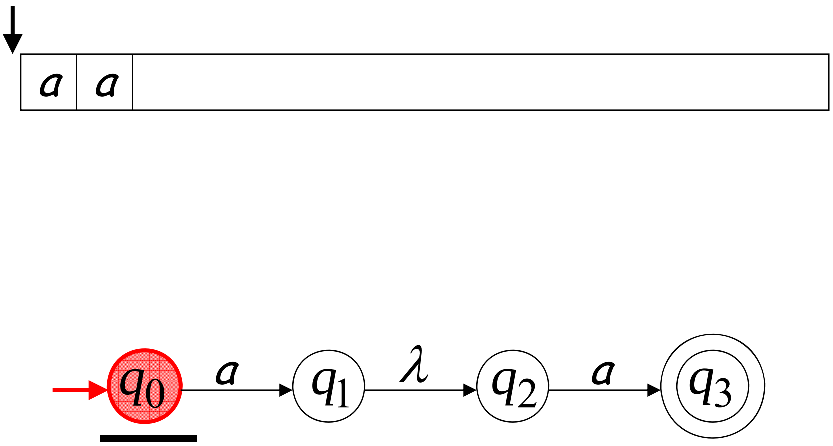


32

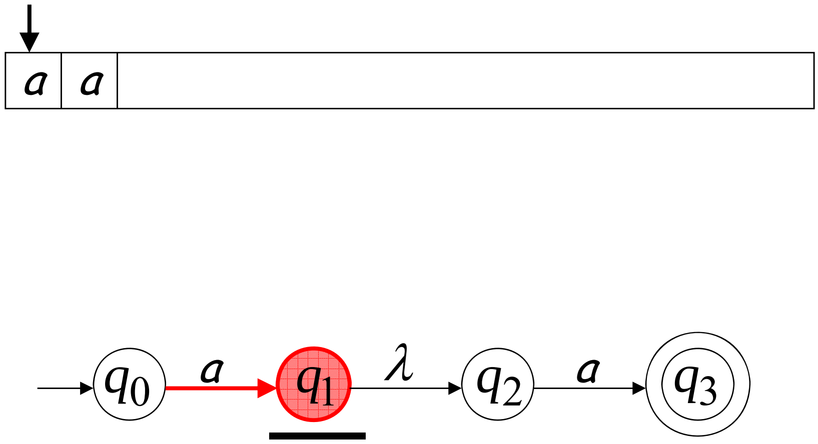
Lambda Transitions



33

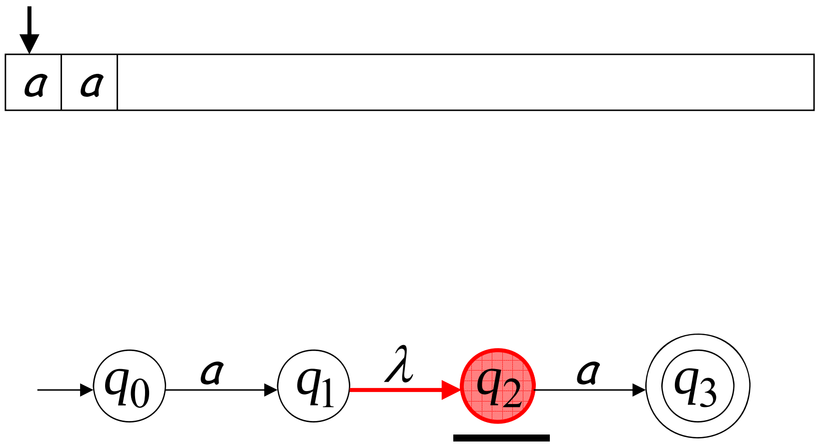


34

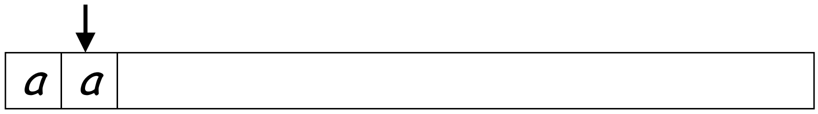


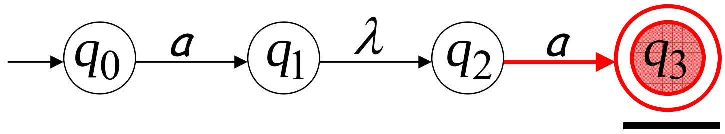
35

(read head does not move)

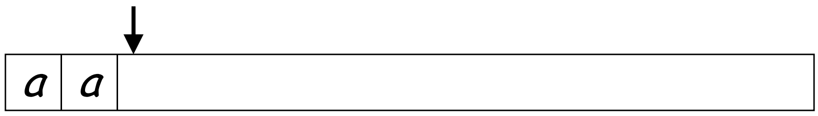


36

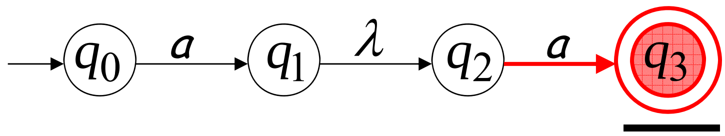




all input is consumed

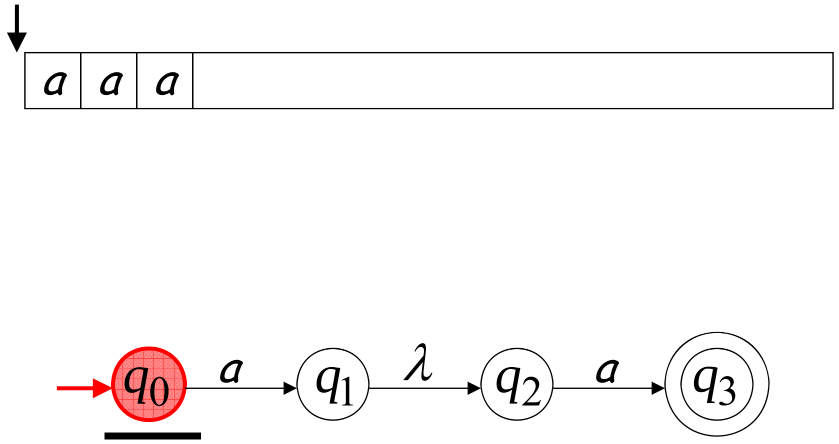


"accept"

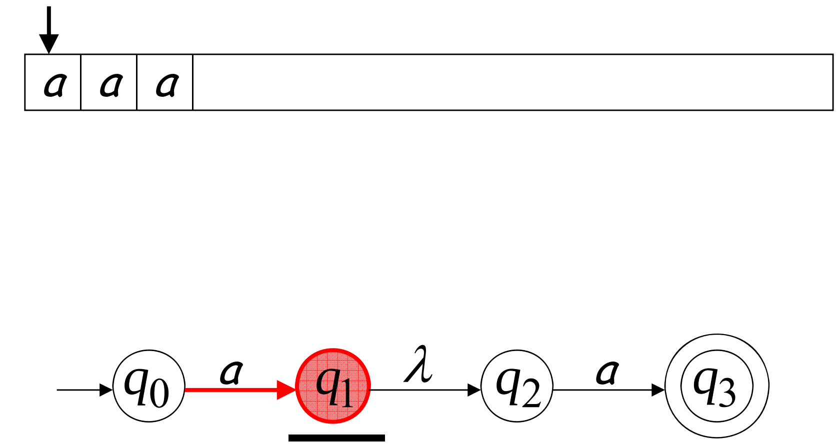


String aa is accepted

Rejection Example

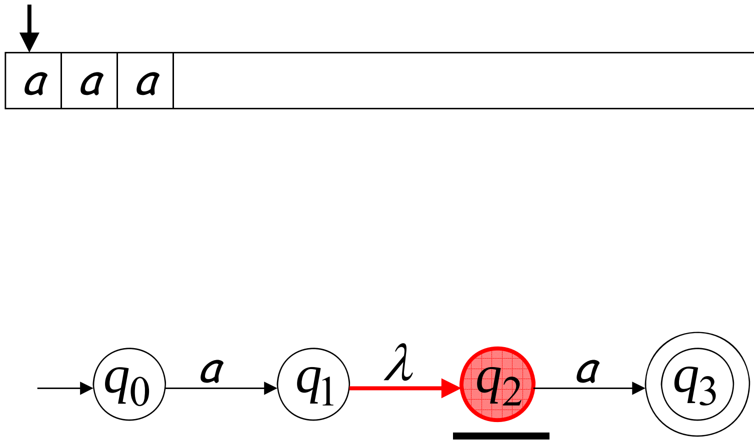


39

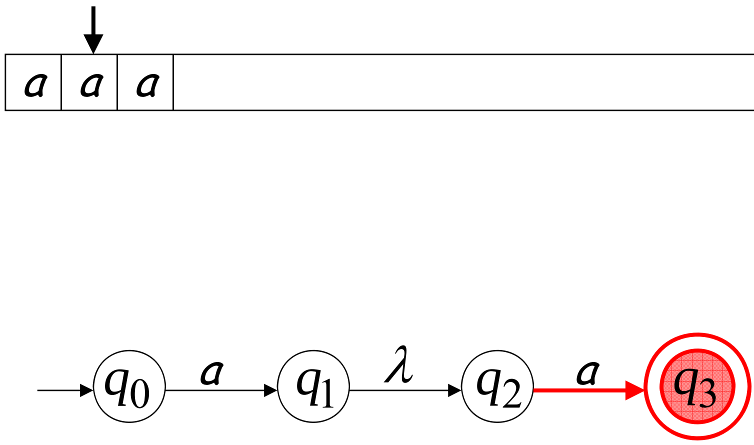


40

(read head doesn't move)



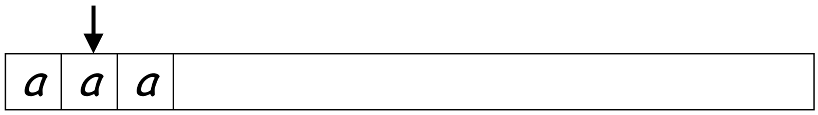
41



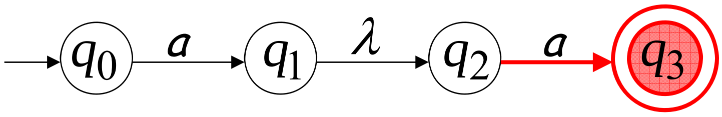
No transition:
the automaton hangs

42

Input cannot be consumed

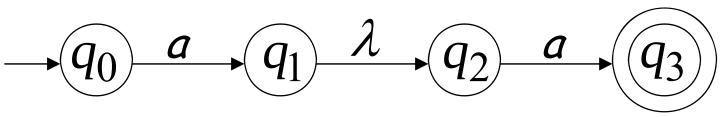


"reject"

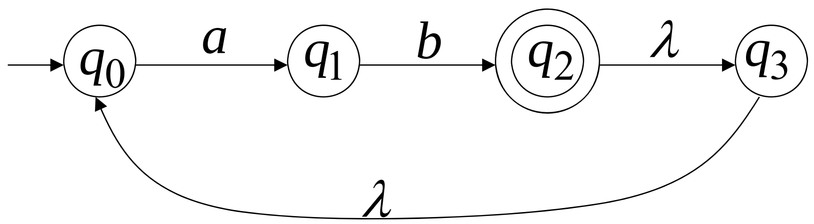


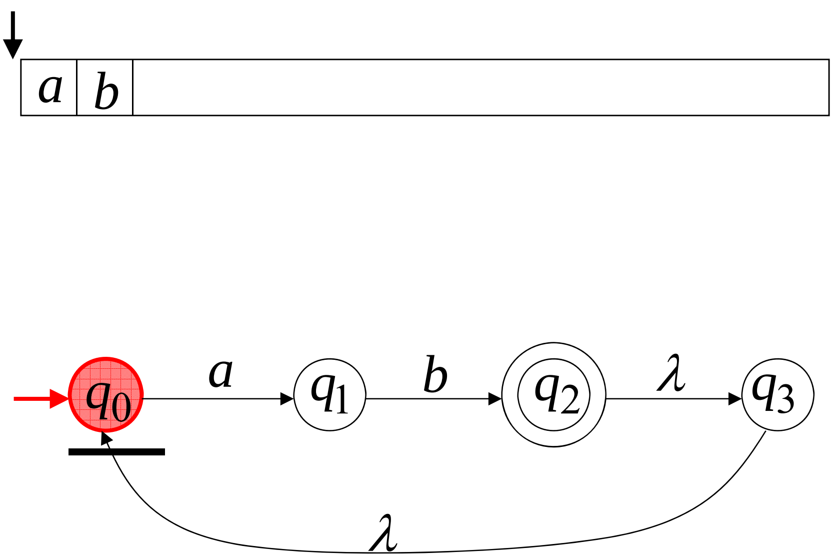
String aaa is rejected

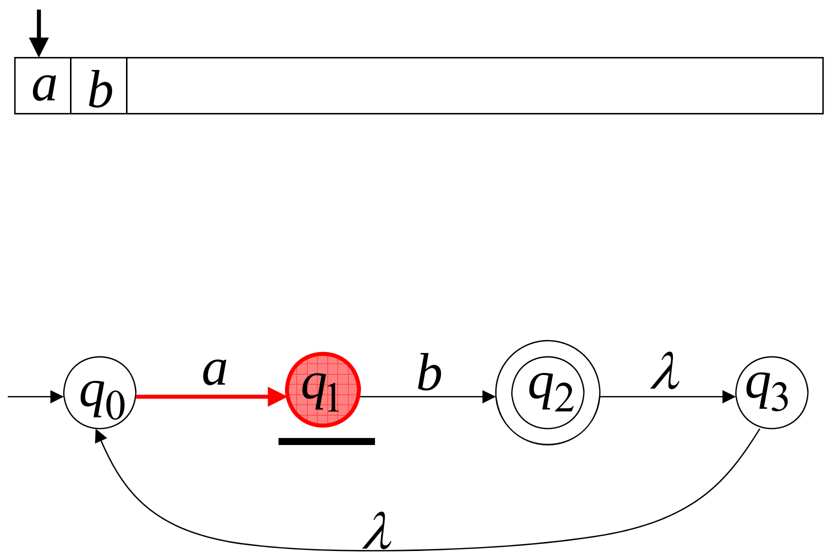
Language accepted: $L = \{aa\}$



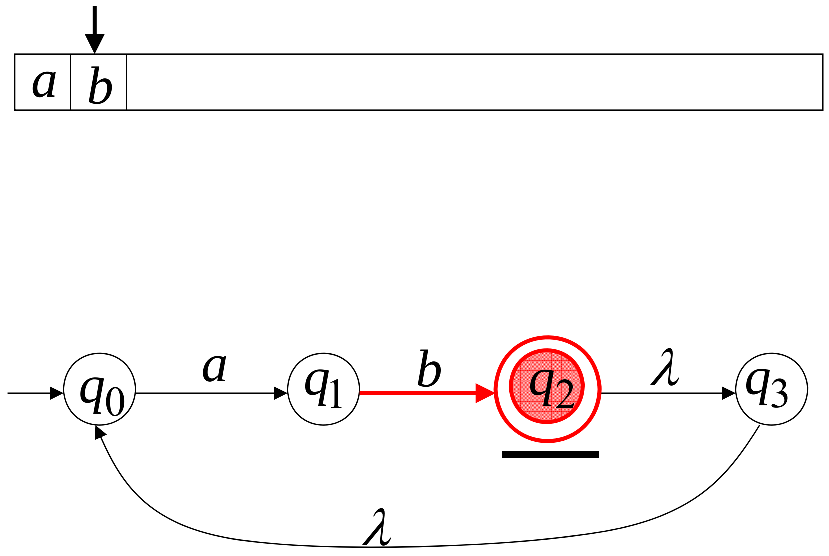
Another NFA Example



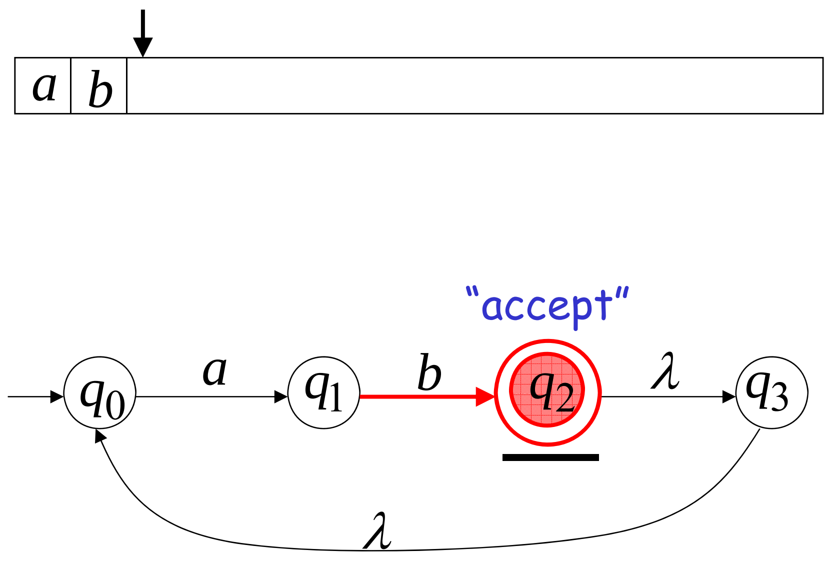




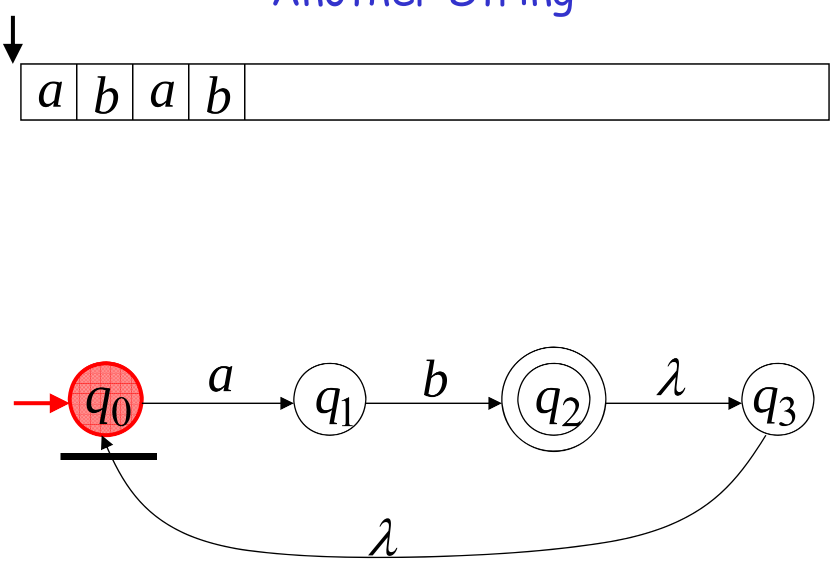
47

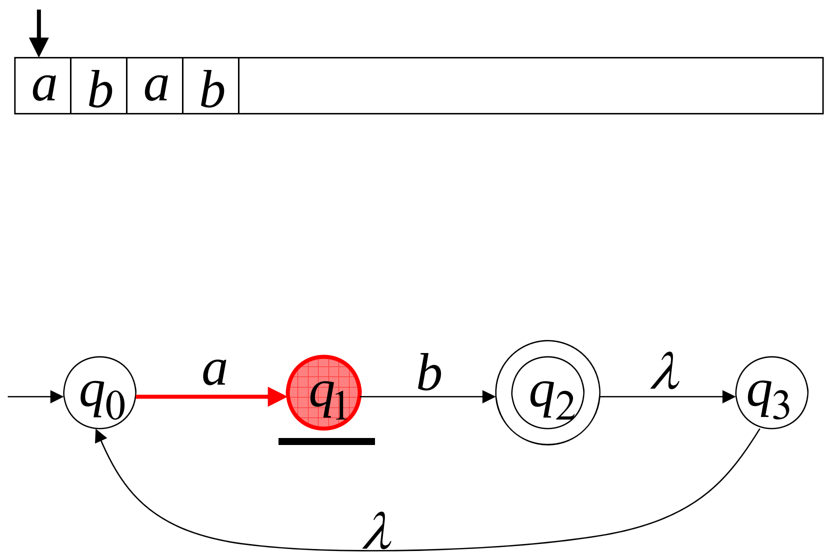


48

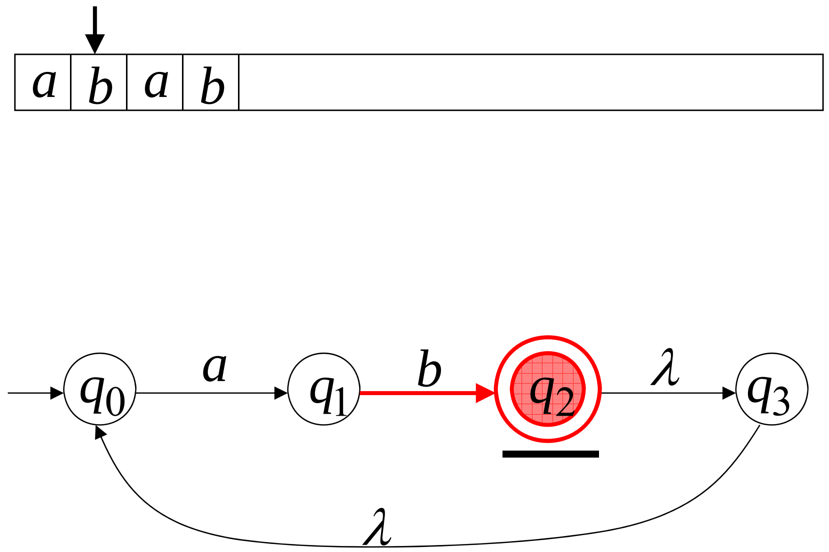


Another String

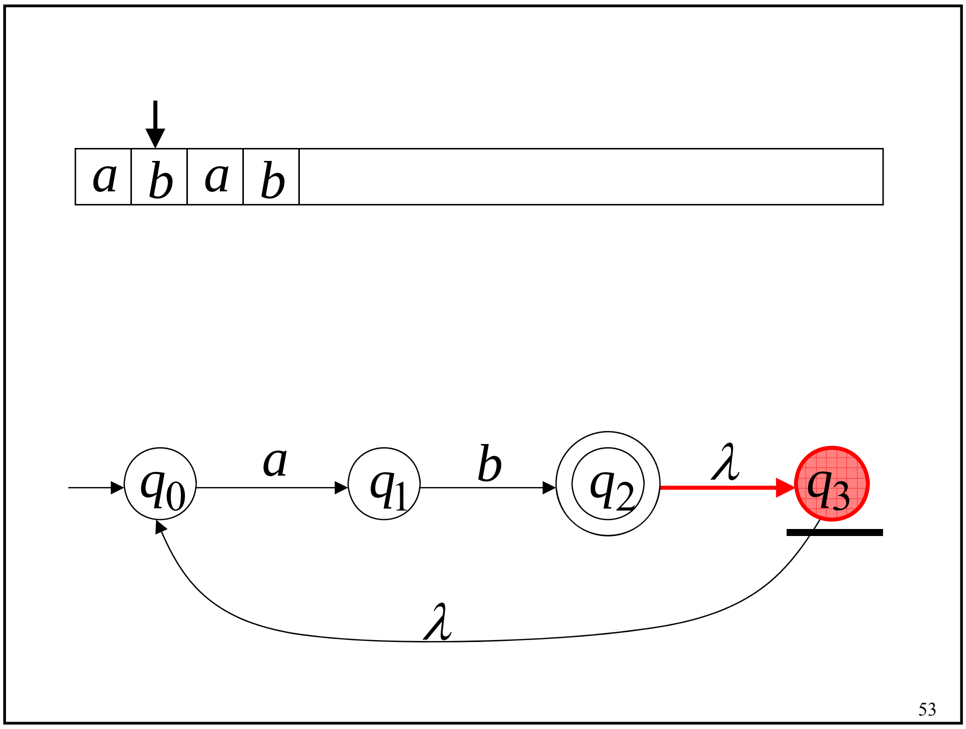


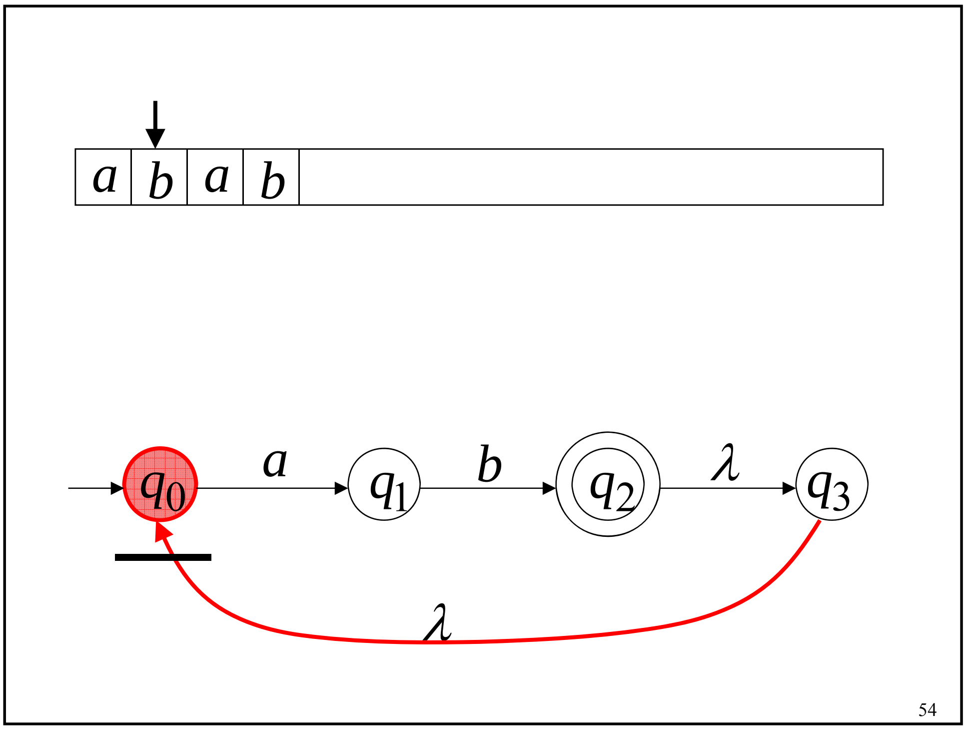

a b a b
q_0
a
q_1
b
q_2
λ
q_3
λ

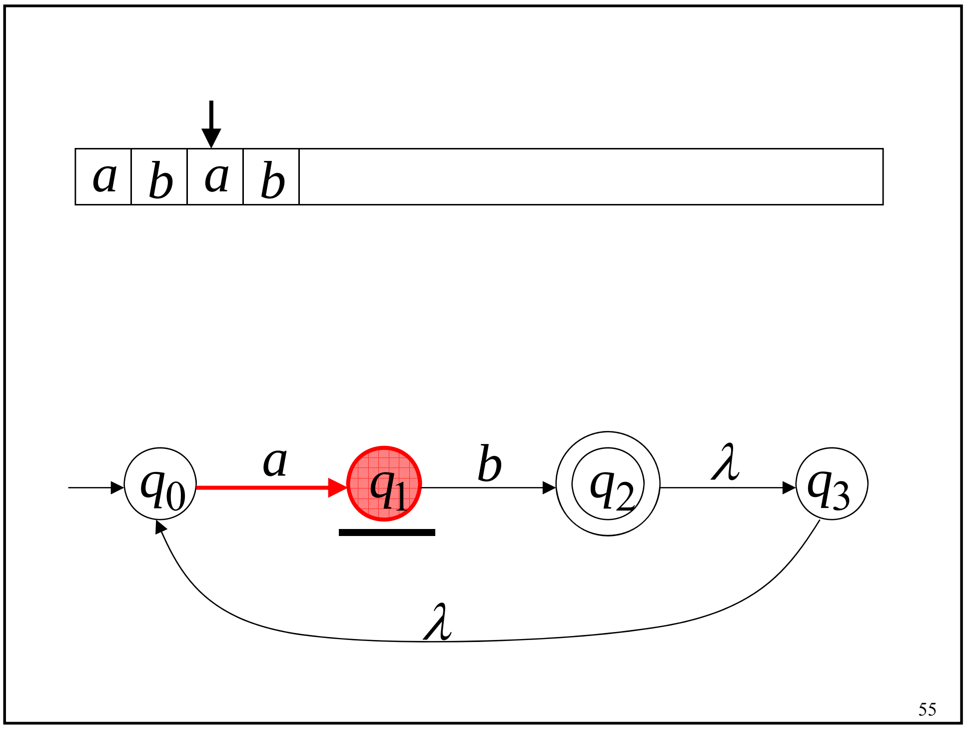
51

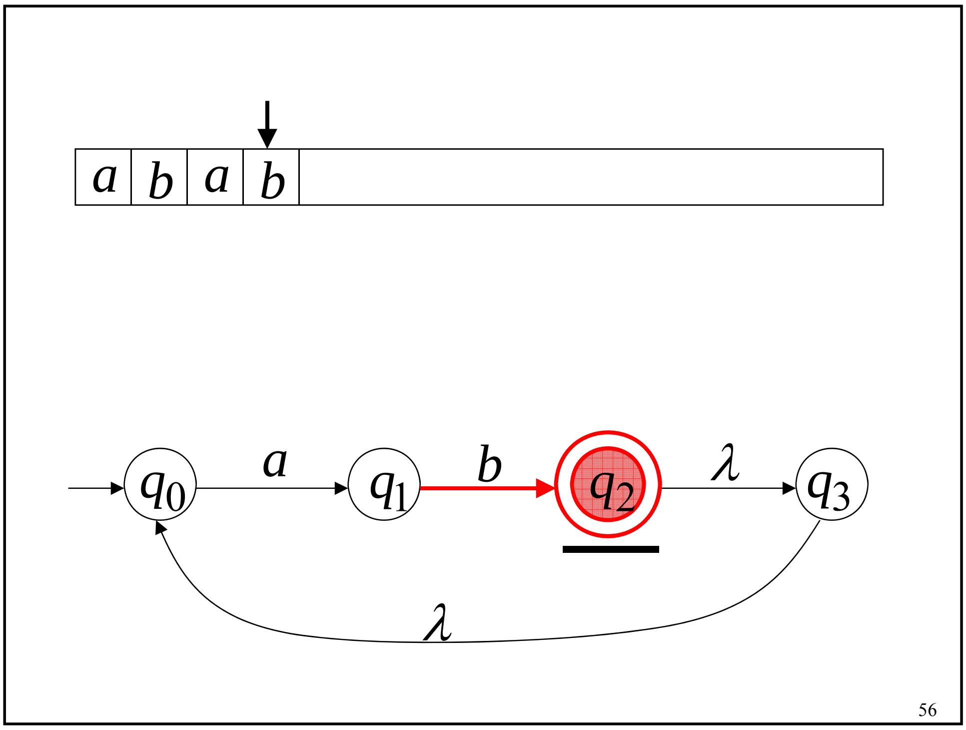

a b a b
q_0
a
q_1
b
q_2
λ
q_3
λ

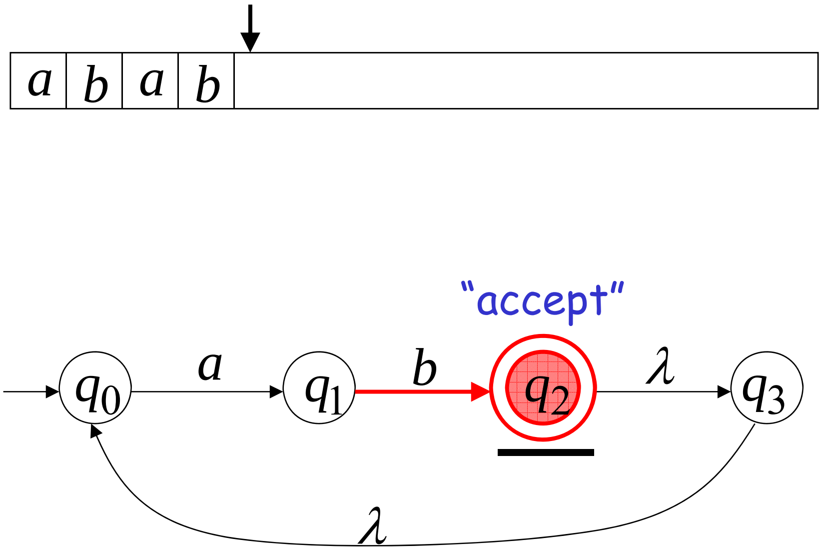
52


a
b
a
b
q_0
a
q_1
b
q_2
λ
q_3
λ
53


a
b
a
b
q_0
a
q_1
b
q_2
λ
q_3
λ
54


a b a b
a
b
λ
λ
q_0 q_1 q_2 q_3
55

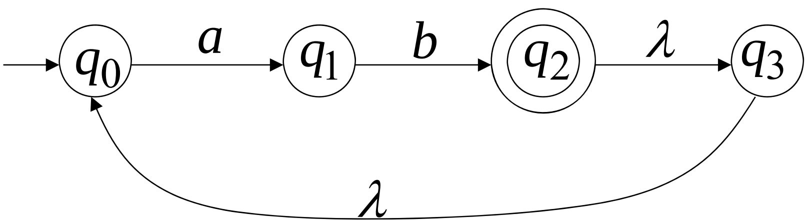

a b a b
a
b
λ
λ
q_0 q_1 q_2 q_3
56



57

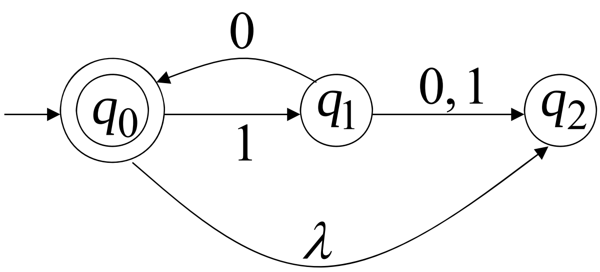
Language accepted

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



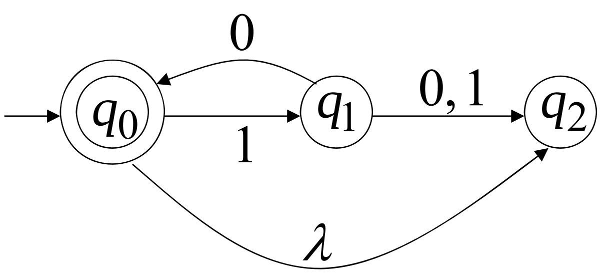
58

Another NFA Example



Language accepted

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



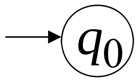
(redundant state)

Remarks:

- The λ symbol never appears on the input tape

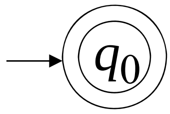
- Simple automata:

M_1



$L(M_1) = \{\}$

M_2

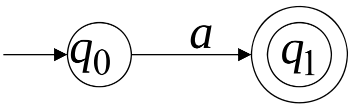


$L(M_2) = \{\lambda\}$

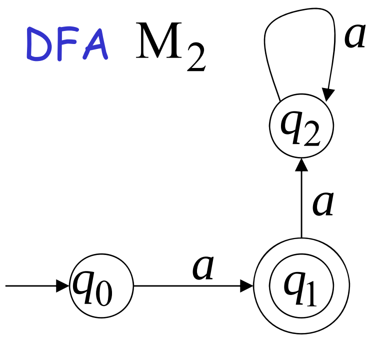
61

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

NFA M_1



DFA M_2



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

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

62

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 aplhabet, i.e. $\{a, b\}$

δ : Transition function

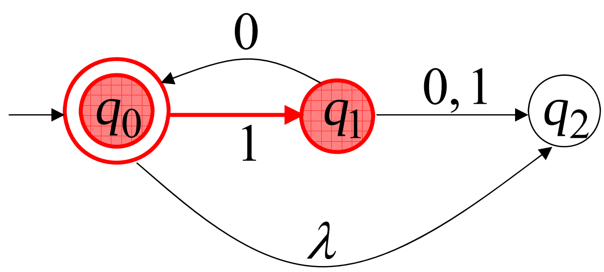
q_0 : Initial state

F : Final states

63

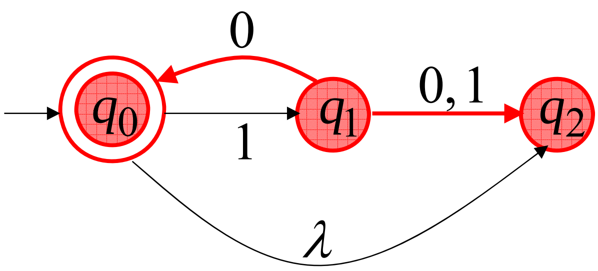
Transition Function δ

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



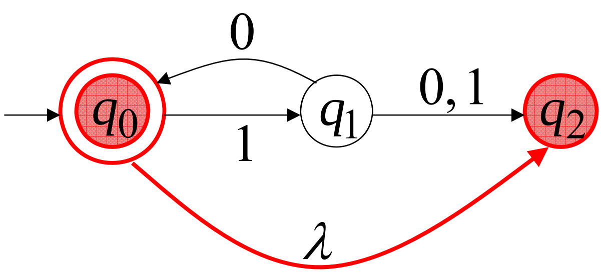
64

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



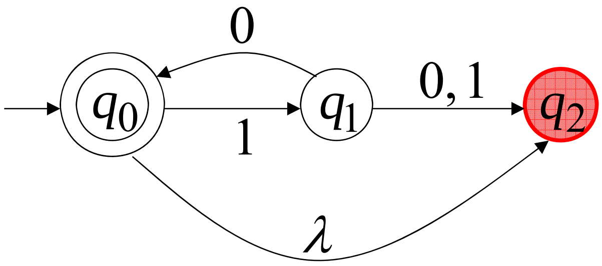
65

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



66

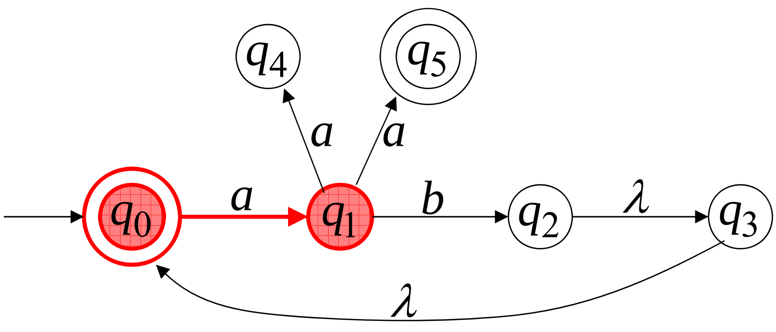
$$\delta(q_2, 1) = \varnothing$$



67

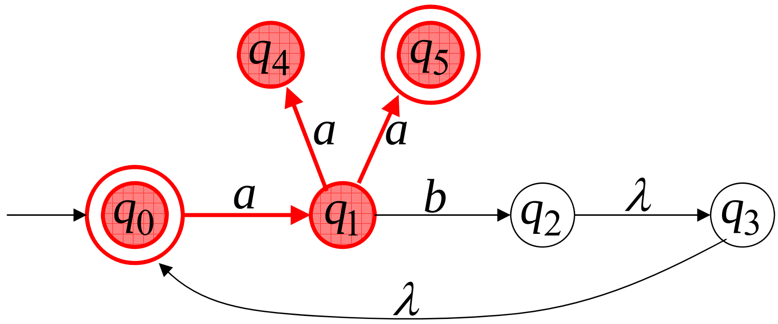
Extended Transition Function δ^*

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



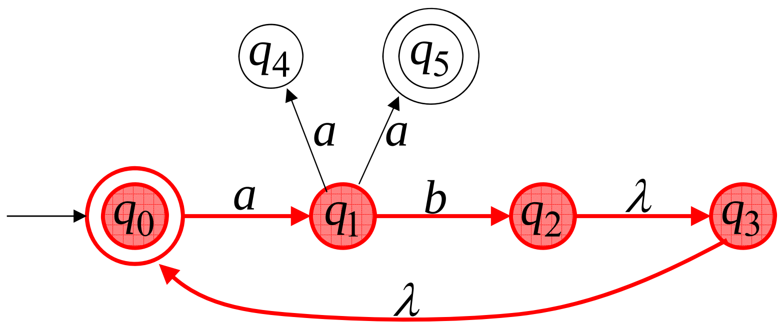
68

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



69

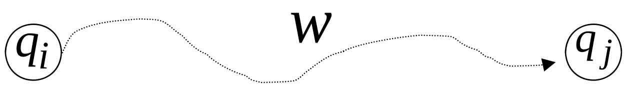
$$\delta^*(q_0, ab) = \{q_2, q_3, q_0\}$$



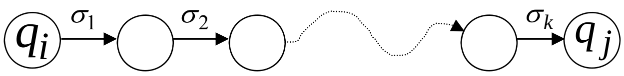
70

Formally

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

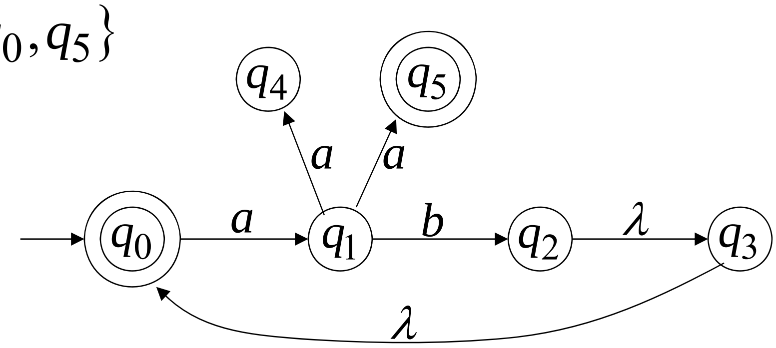


$$w = \sigma_1 \sigma_2 \cdots \sigma_k$$



The Language of an NFA M

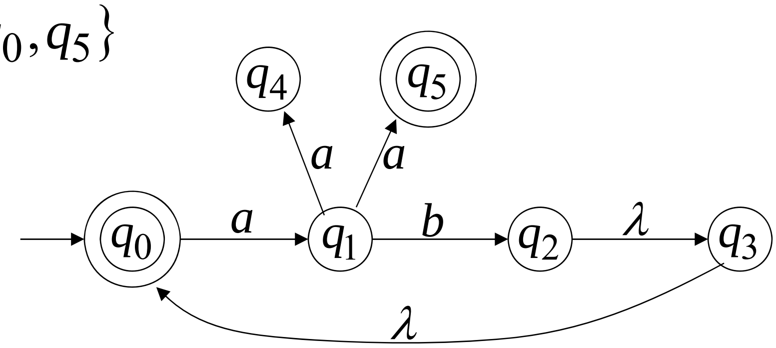
$F = \{q_0, q_5\}$



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

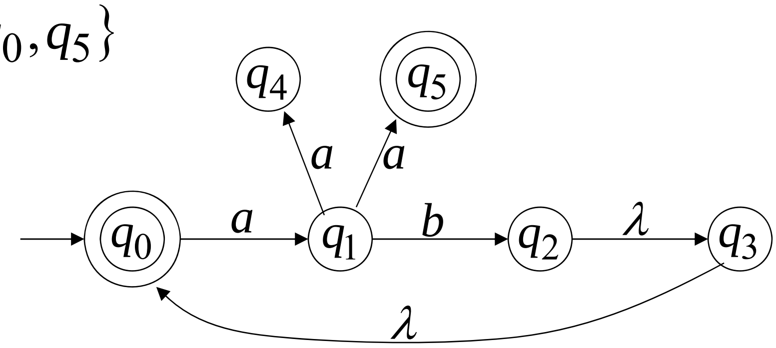
$aa \in L(M)$

$F = \{q_0, q_5\}$



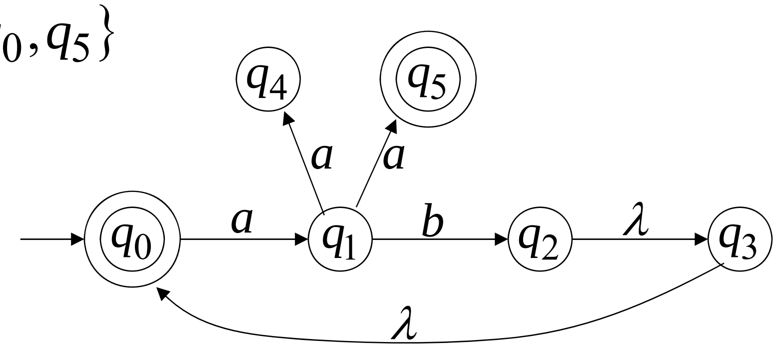
$$\delta^*(q_0, ab) = \{q_2, q_3, \underbrace{q_0}_{\in F}\} \qquad ab \in L(M)$$

$F = \{q_0, q_5\}$



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

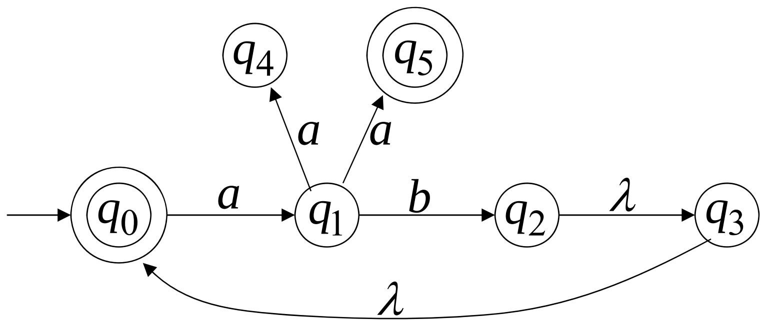
$F = \{q_0, q_5\}$



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

$\searrow \notin F$

75



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

76

Formally

The language accepted by NFA M is:

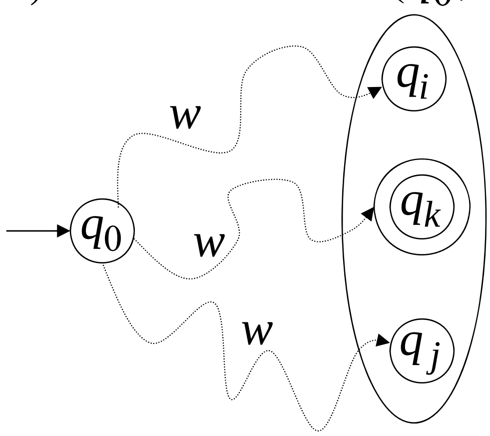
$$L(M) = \{w_1, w_2, w_3, \ldots\}$$

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

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

77

$w \in L(M)$ $\delta^*(q_0, w)$



$q_k \in F$

78

NFAs accept the Regular Languages

79

Equivalence of Machines

Definition for Automata:

Machine M_1 is equivalent to machine M_2

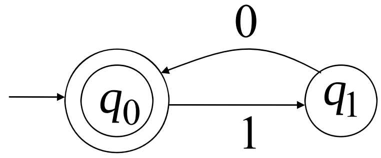
if $L(M_1) = L(M_2)$

80

Example of equivalent machines

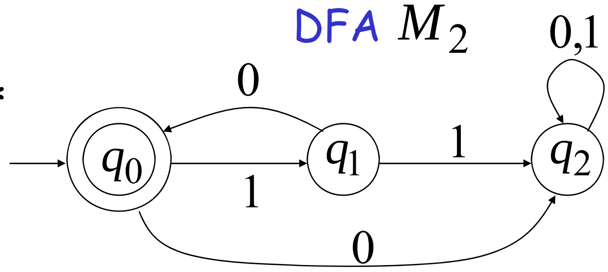
NFA M_1

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



DFA M_2

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



81

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

82

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

⇓

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

Step 2

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

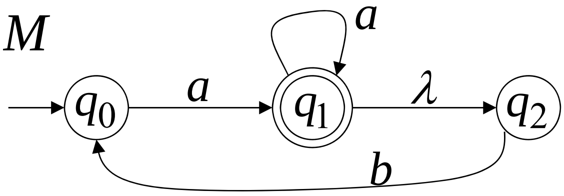
Proof: Any NFA can be converted to an equivalent DFA

⇓

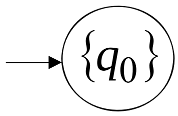
Any language L accepted by an NFA is also accepted by a DFA

Convert NFA to DFA

NFA M



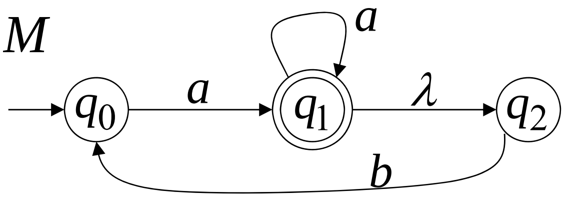
DFA M'



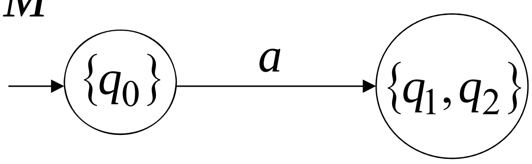
85

Convert NFA to DFA

NFA M



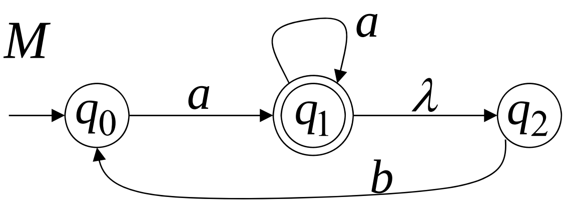
DFA M'



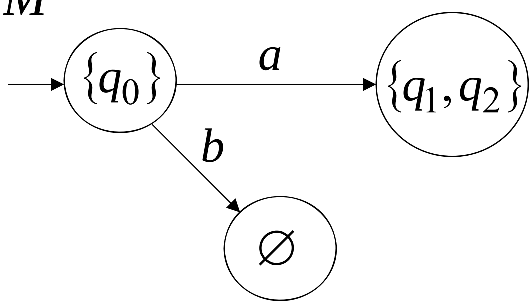
86

Convert NFA to DFA

NFA M

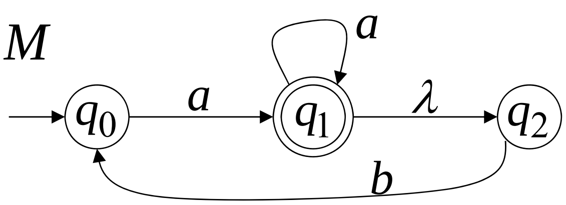


DFA M'

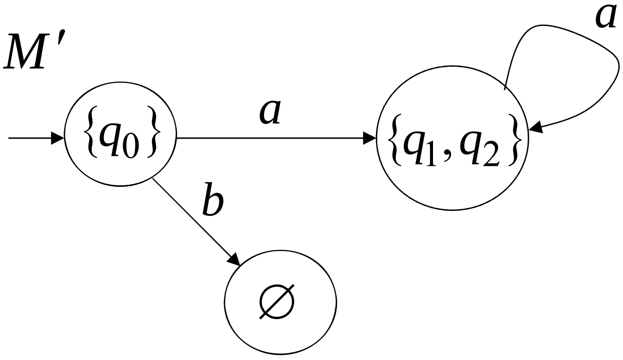


Convert NFA to DFA

NFA M

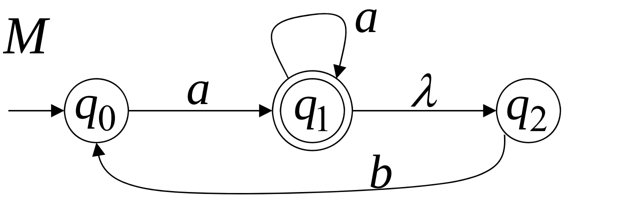


DFA M'

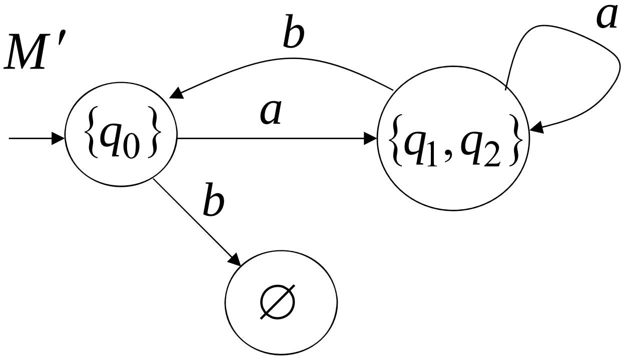


Convert NFA to DFA

NFA M



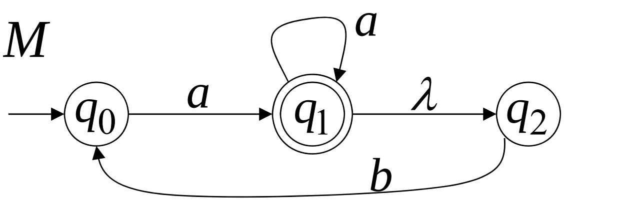
DFA M'



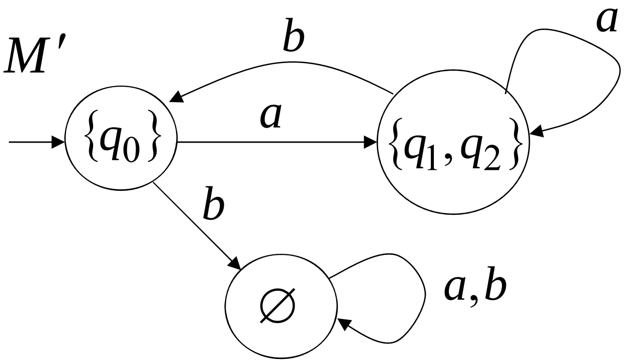
89

Convert NFA to DFA

NFA M



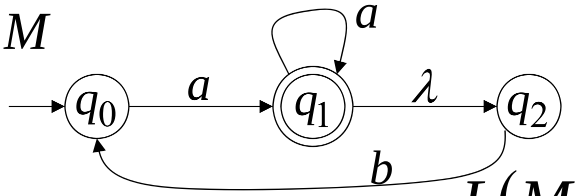
DFA M'



90

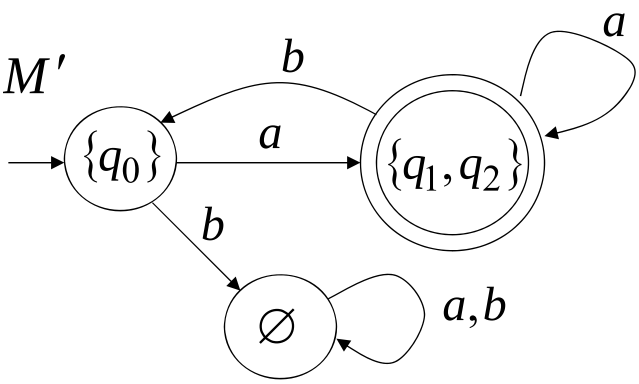
Convert NFA to DFA

NFA M



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

DFA M'



91