

# Turing Machines

# The Language Hierarchy

$a^n b^n c^n$  ?

$ww$  ?

Context-Free Languages

$a^n b^n$

$ww^R$

Regular Languages

$a^*$

$a^* b^*$

Languages accepted by  
**Turing Machines**

$a^n b^n c^n$

$ww$

Context-Free Languages

$a^n b^n$

$ww^R$

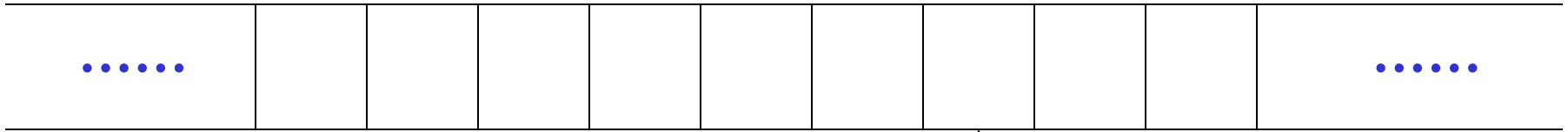
Regular Languages

$a^*$

$a^* b^*$

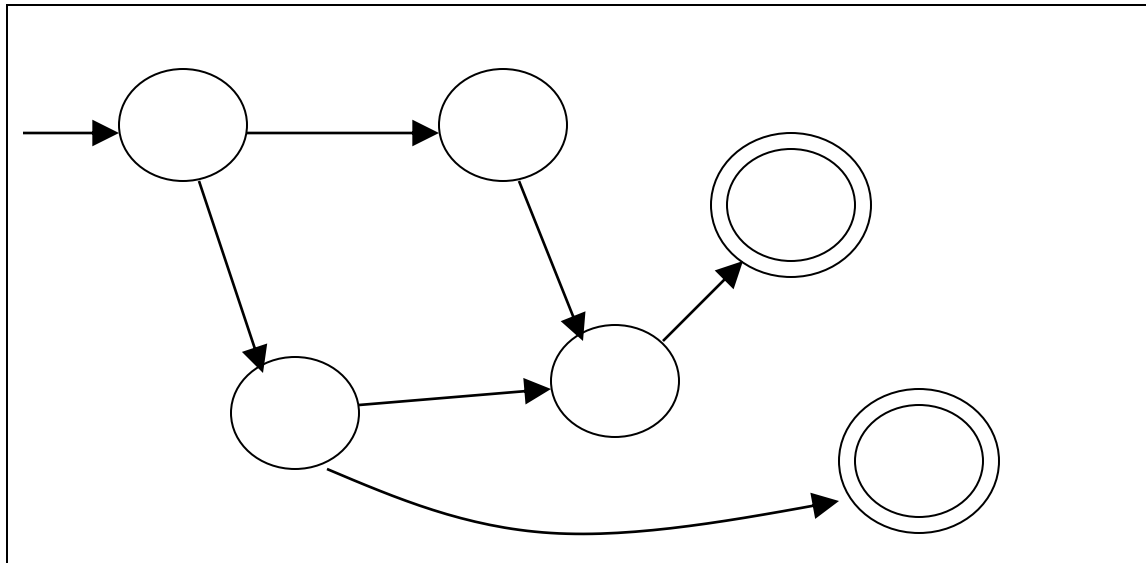
# A Turing Machine

Tape



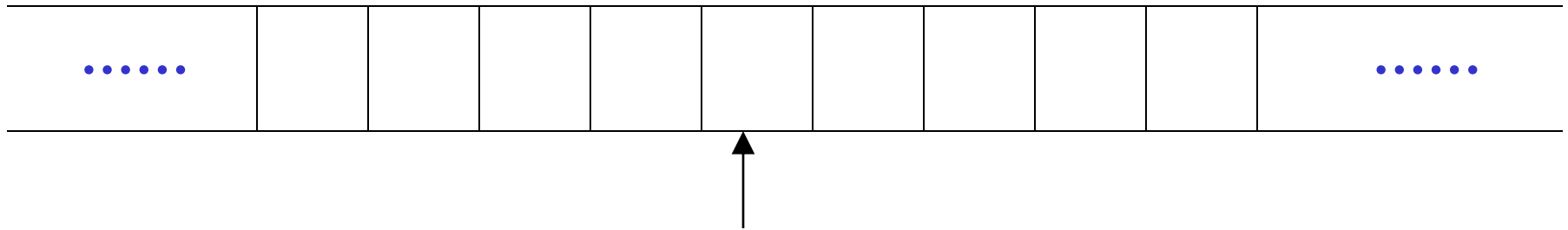
Read-Write head

Control Unit



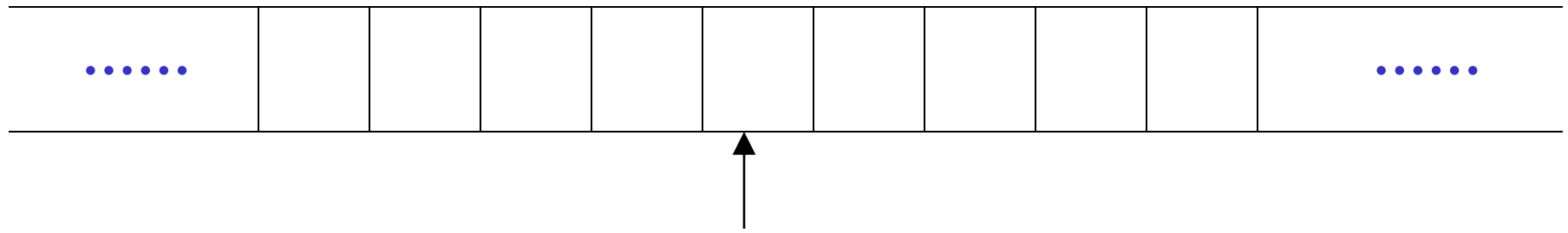
# The Tape

No boundaries -- infinite length



Read-Write head

The head moves Left or Right



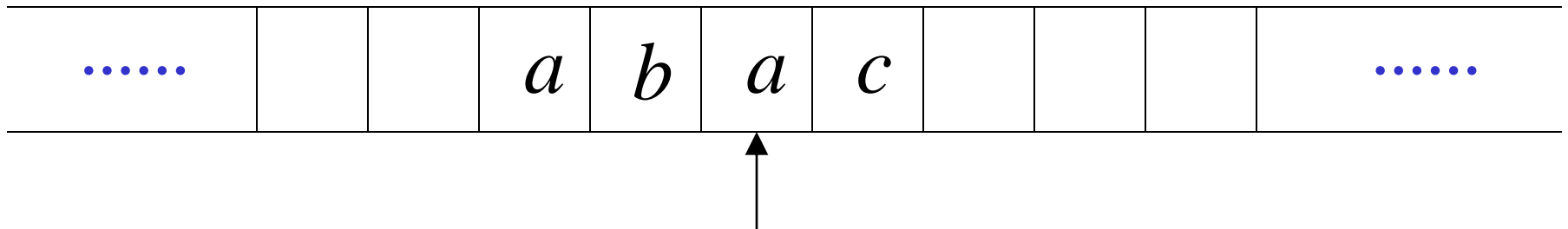
Read-Write head

The head at each time step:

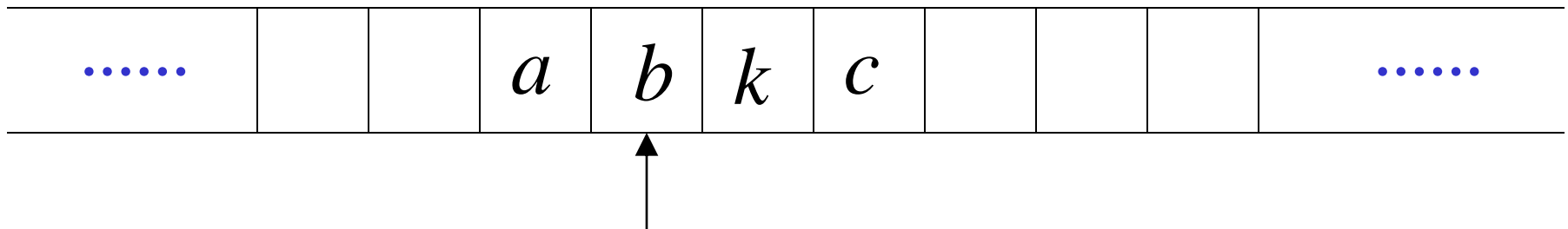
1. Reads a symbol
2. Writes a symbol
3. Moves Left or Right

Example:

Time 0

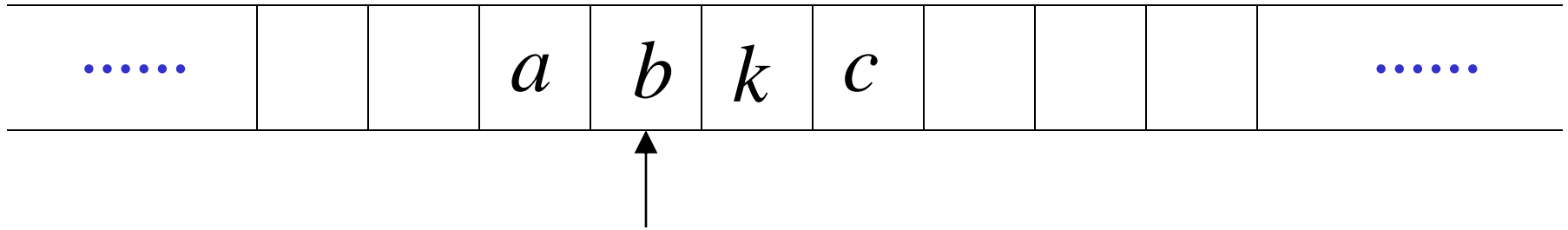


Time 1

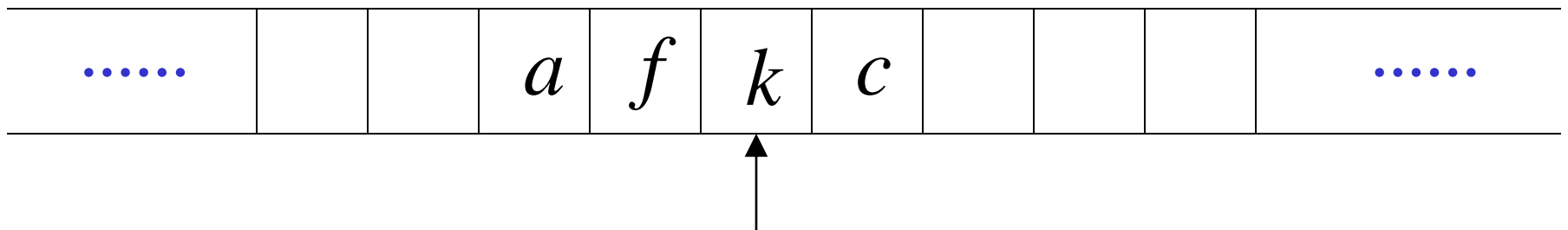


1. Reads  $a$
2. Writes  $k$
3. Moves Left

Time 1



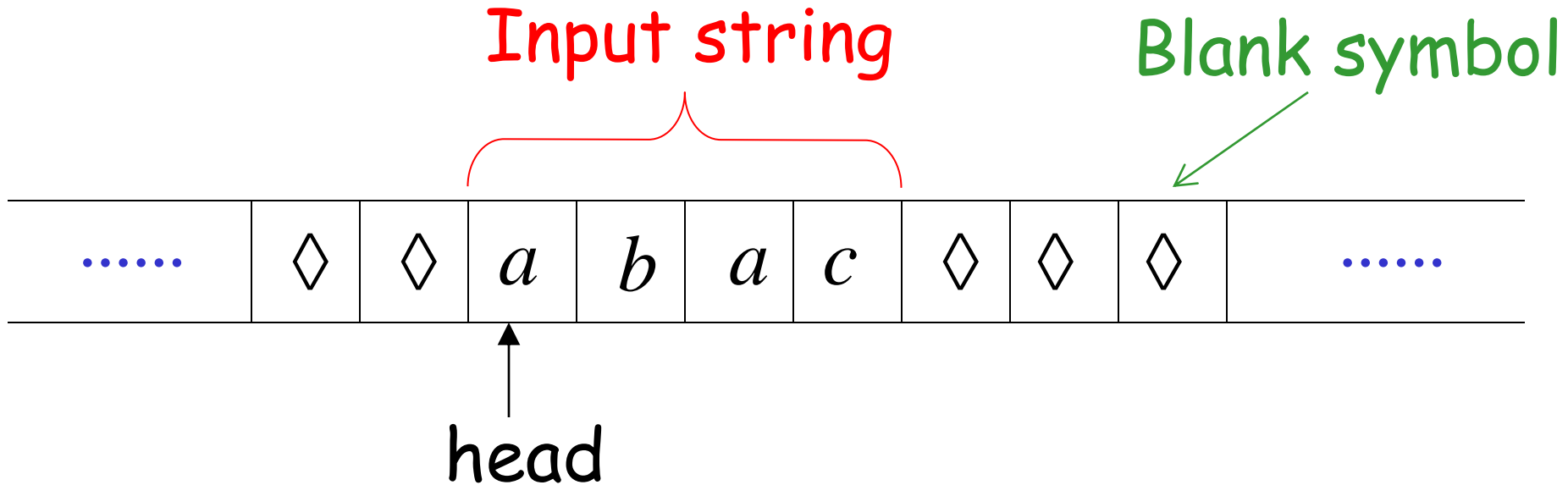
Time 2



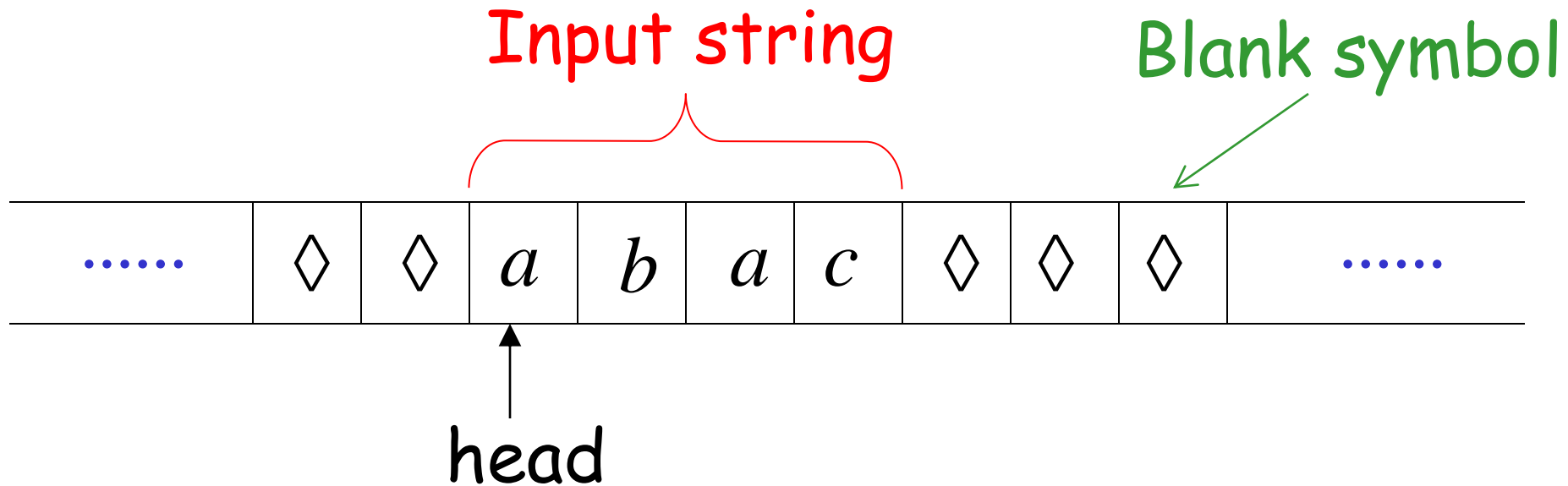
1. Reads  $b$
2. Writes  $f$
3. Moves Right



# The Input String

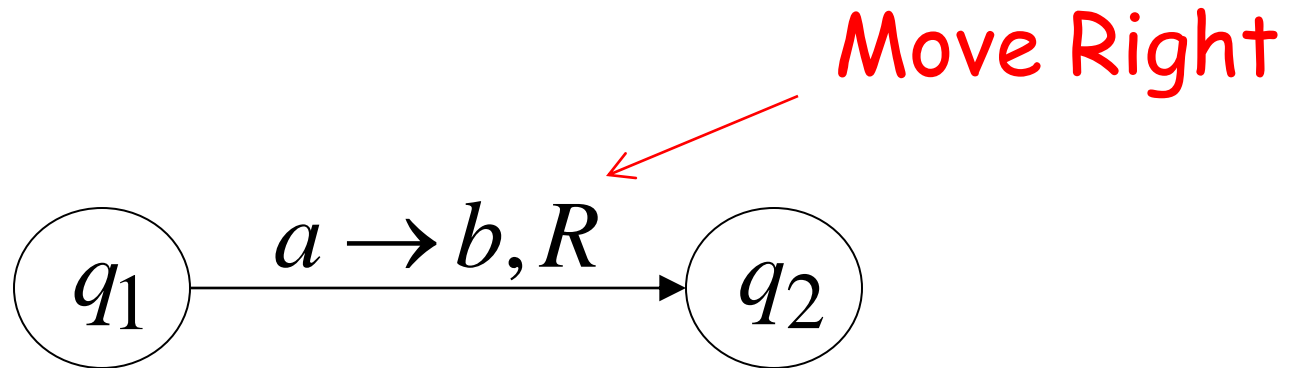
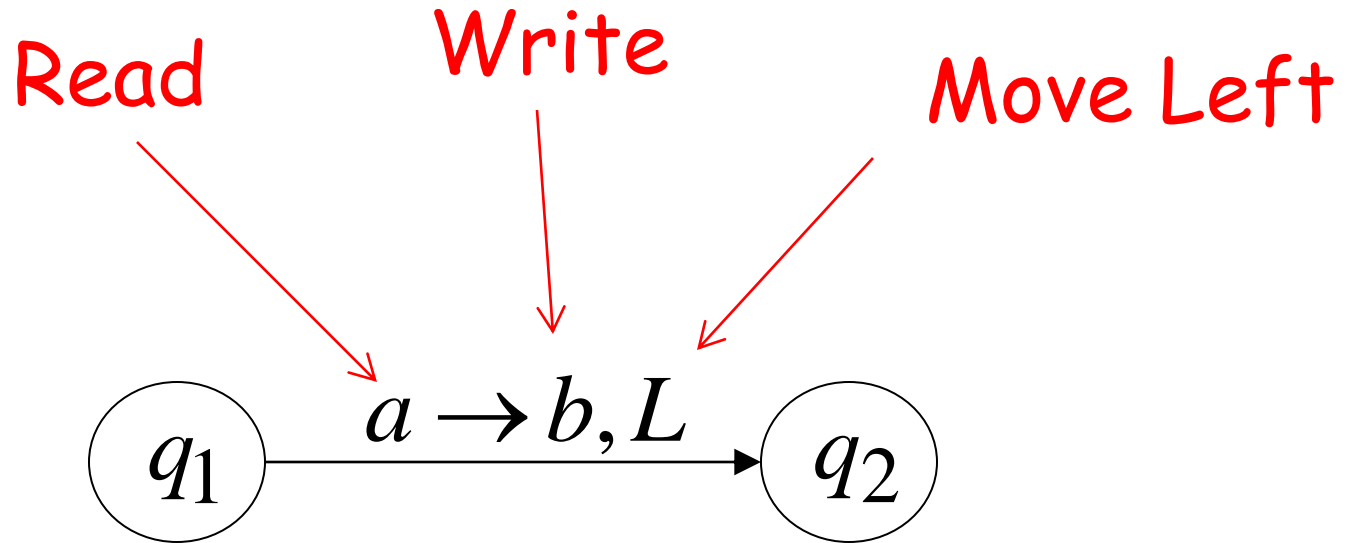


Head starts at the leftmost position of the input string



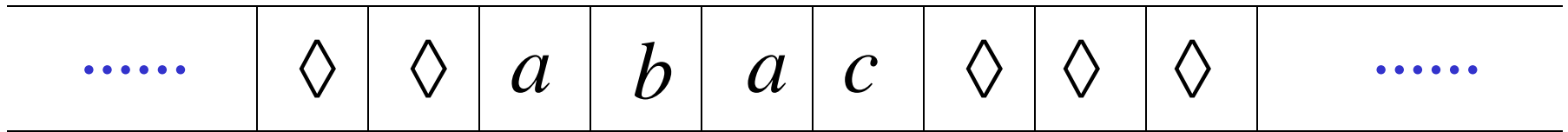
Remark: the input string is never empty

# States & Transitions



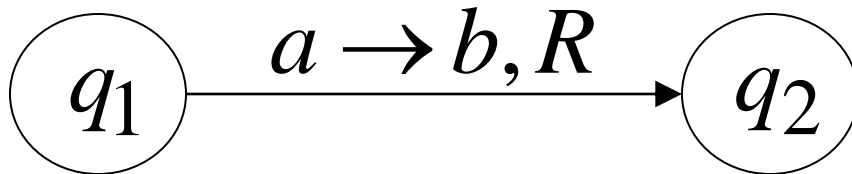
Example:

Time 1

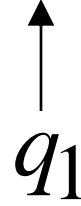
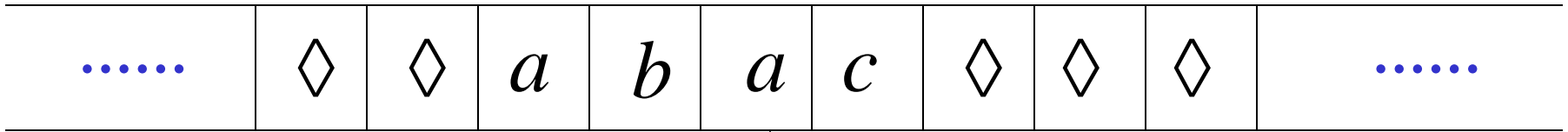


$q_1$

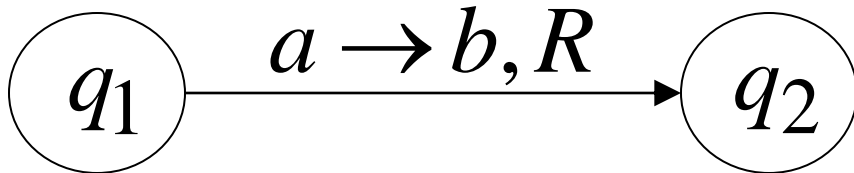
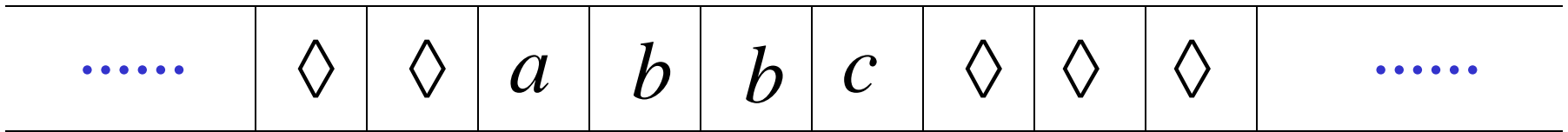
current state



Time 1

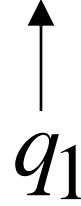
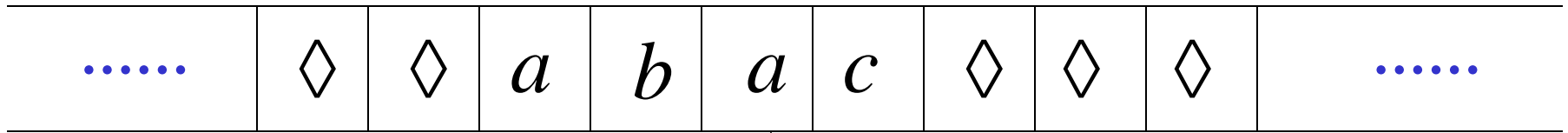


Time 2

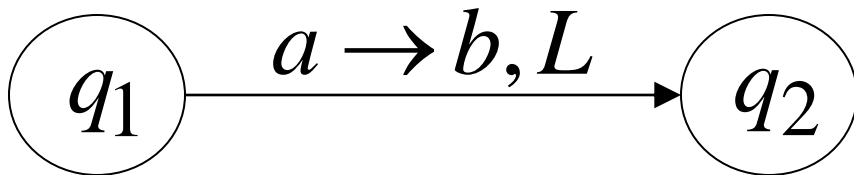
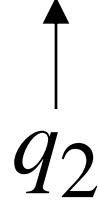
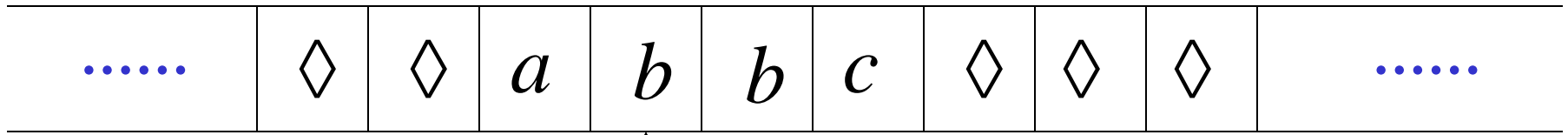


Example:

Time 1

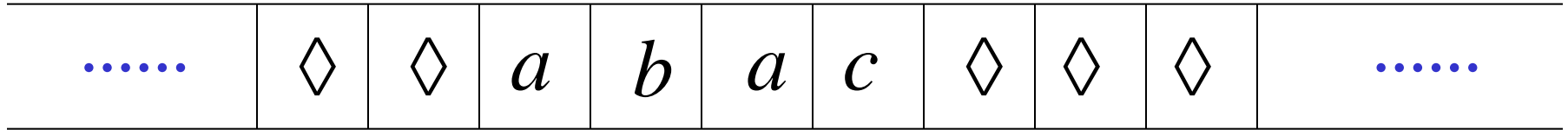


Time 2

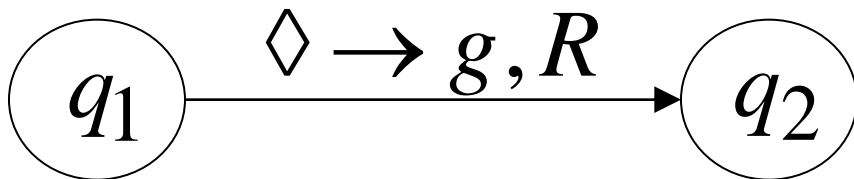
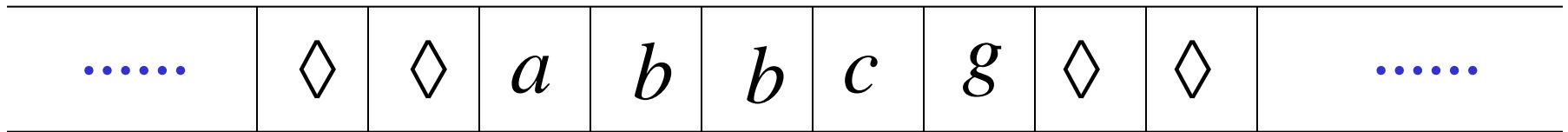


Example:

Time 1



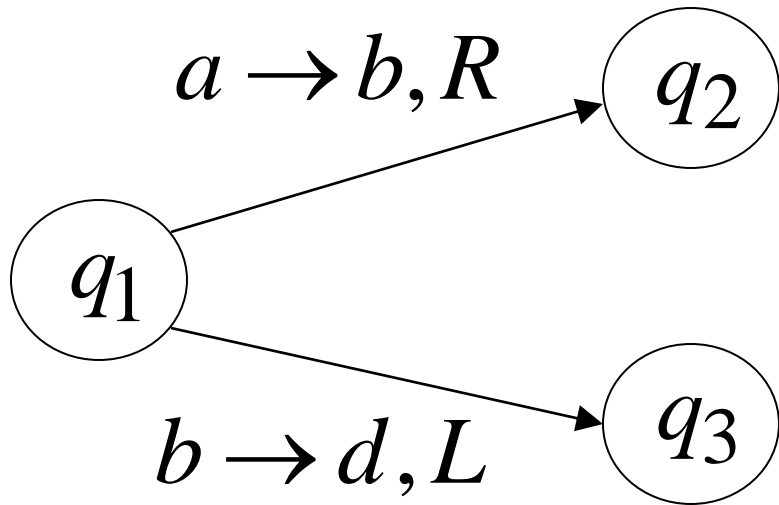
Time 2



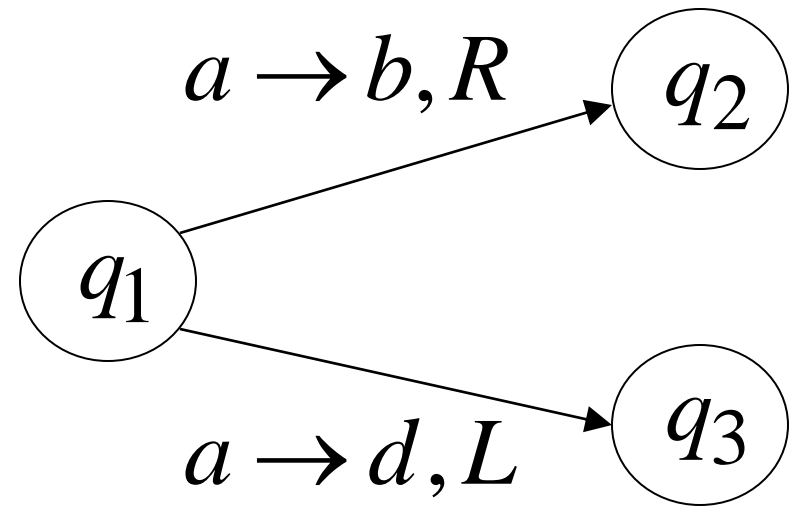
# Determinism

Turing Machines are deterministic

Allowed



Not Allowed

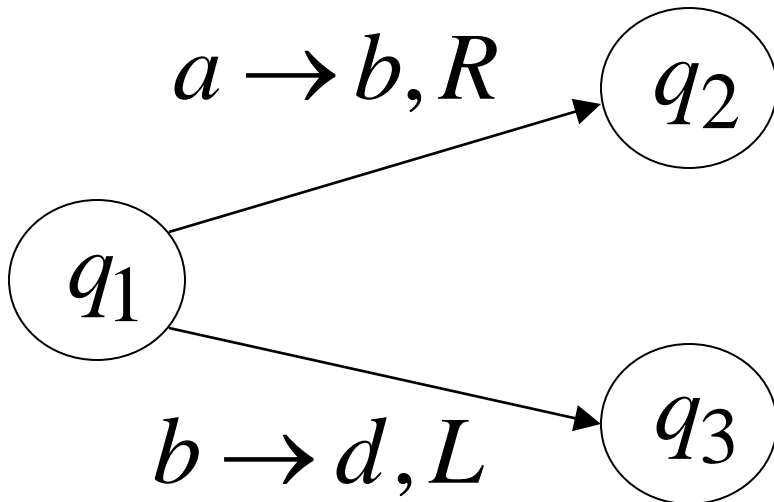
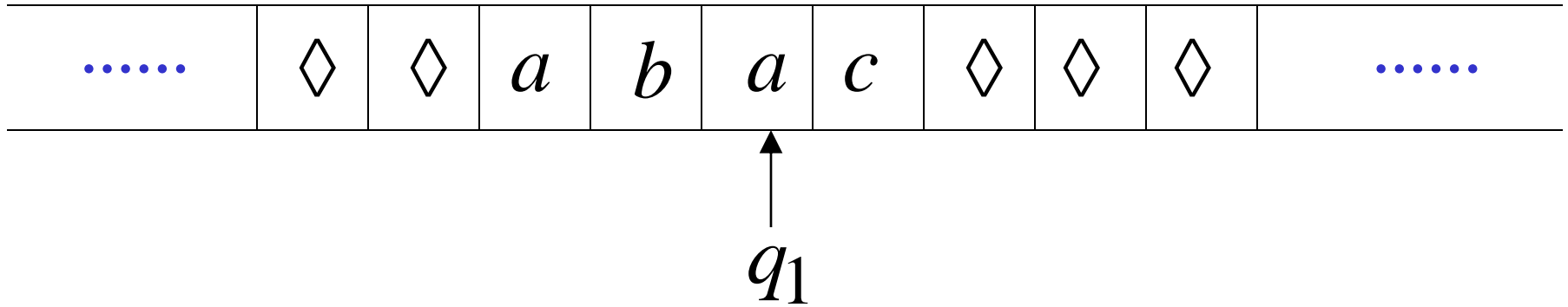


No lambda transitions allowed



# Partial Transition Function

Example:



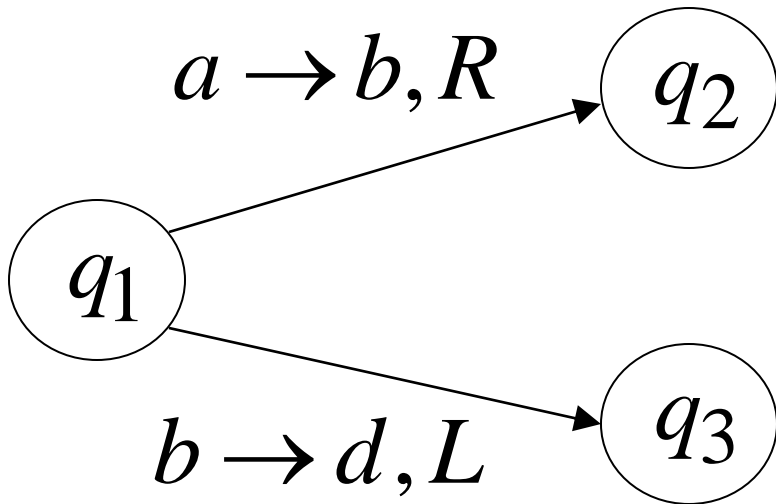
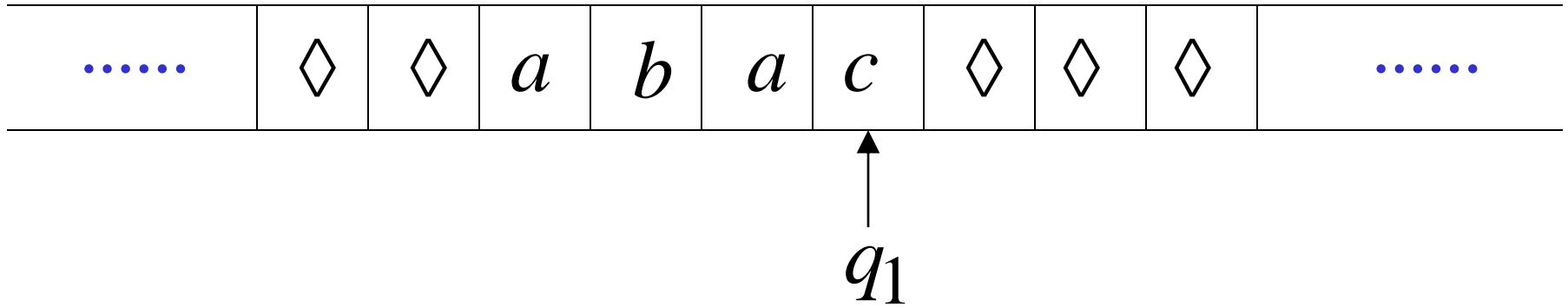
Allowed:

No transition  
for input symbol  $c$

# Halting

The machine *halts* if there are no possible transitions to follow

Example:



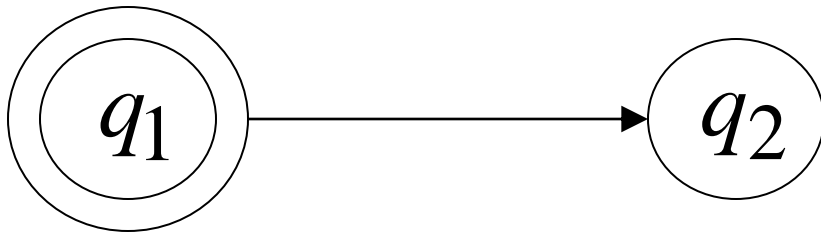
No possible transition

**HALT!!!**

# Final States



Allowed



Not Allowed

- Final states have no outgoing transitions
- In a final state the machine halts

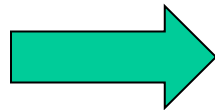
# Acceptance

Accept Input



If machine halts  
in a final state

Reject Input



If machine halts  
in a non-final state

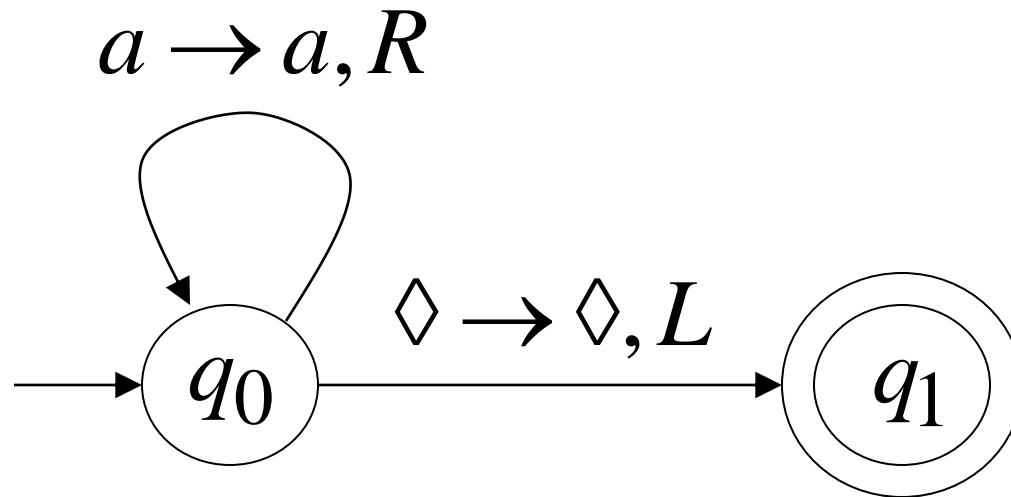
or

If machine enters  
an *infinite loop*

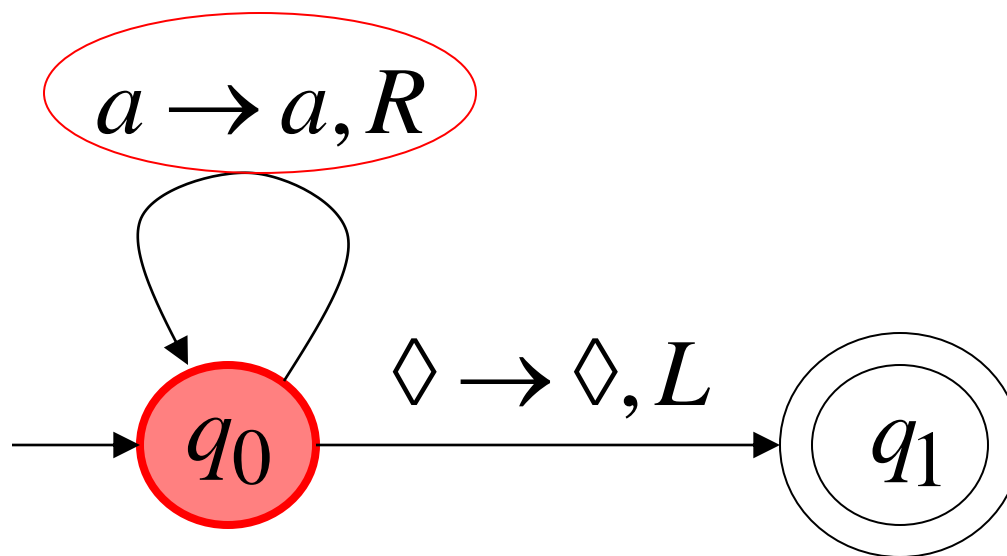
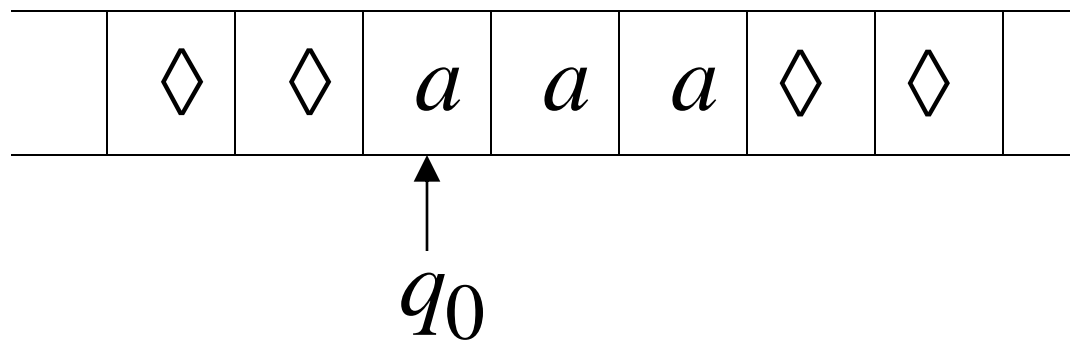
# Turing Machine Example

A Turing machine that accepts the language:

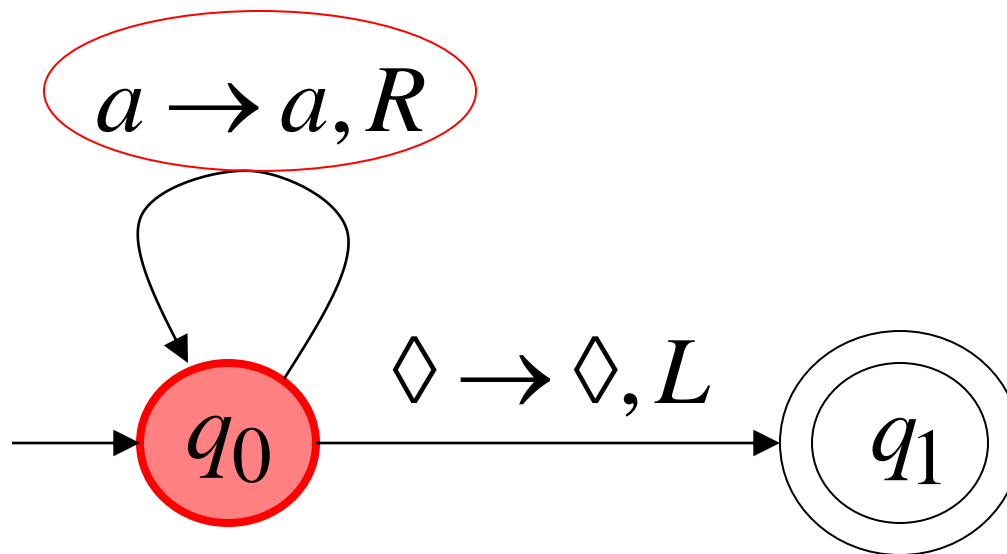
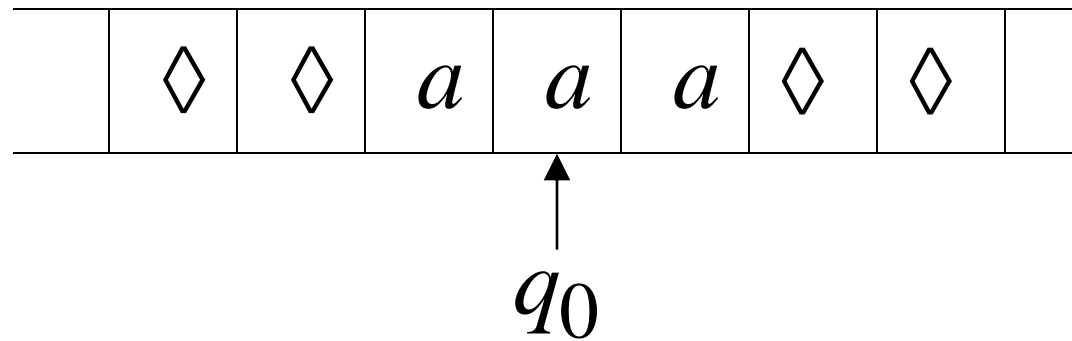
$aa^*$



Time 0

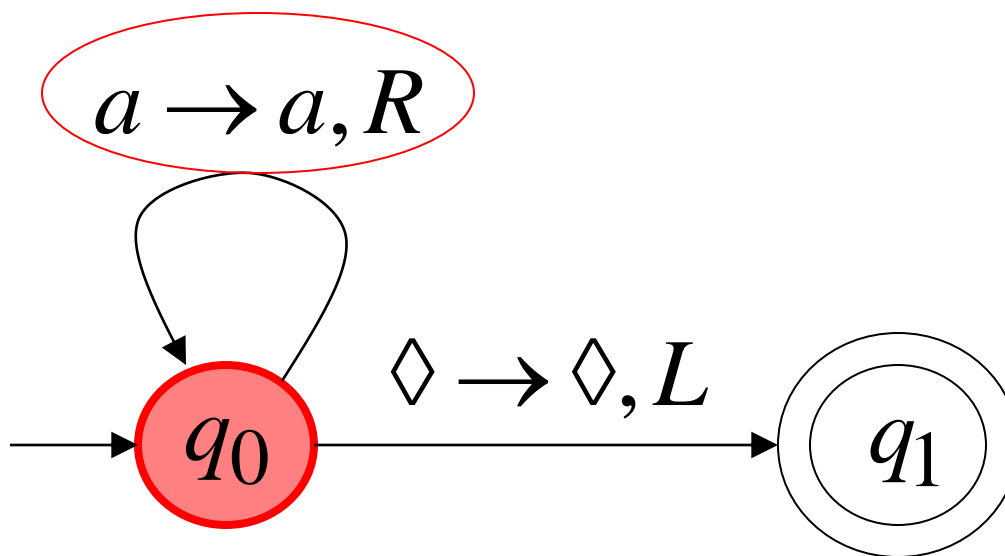
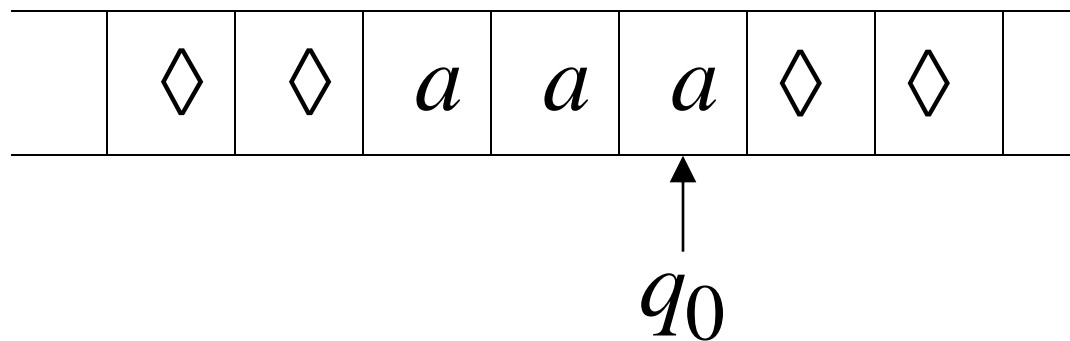


Time 1

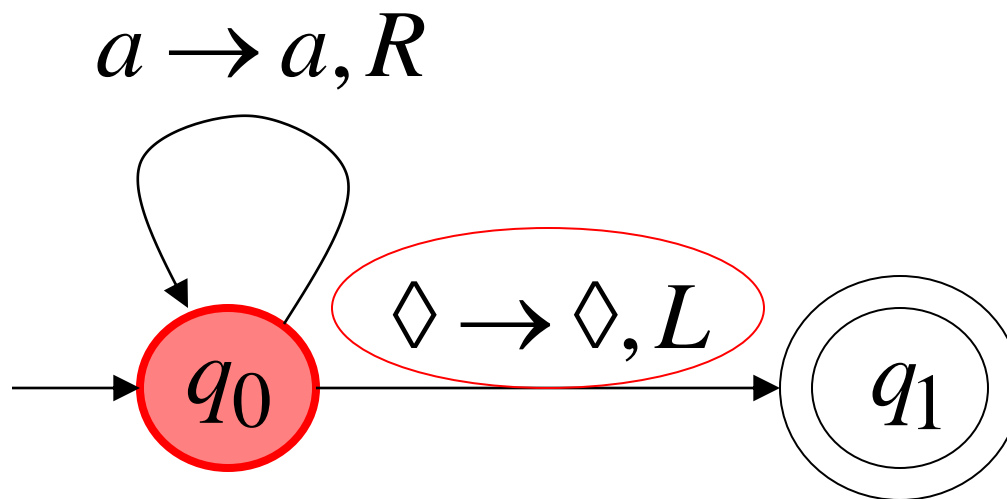
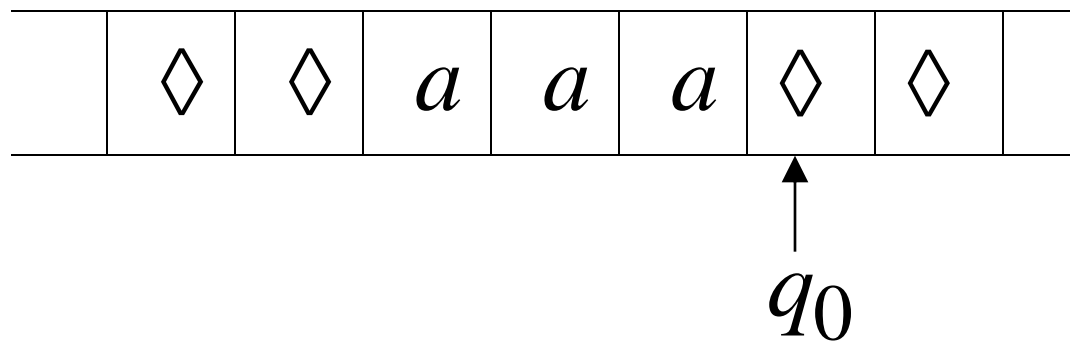




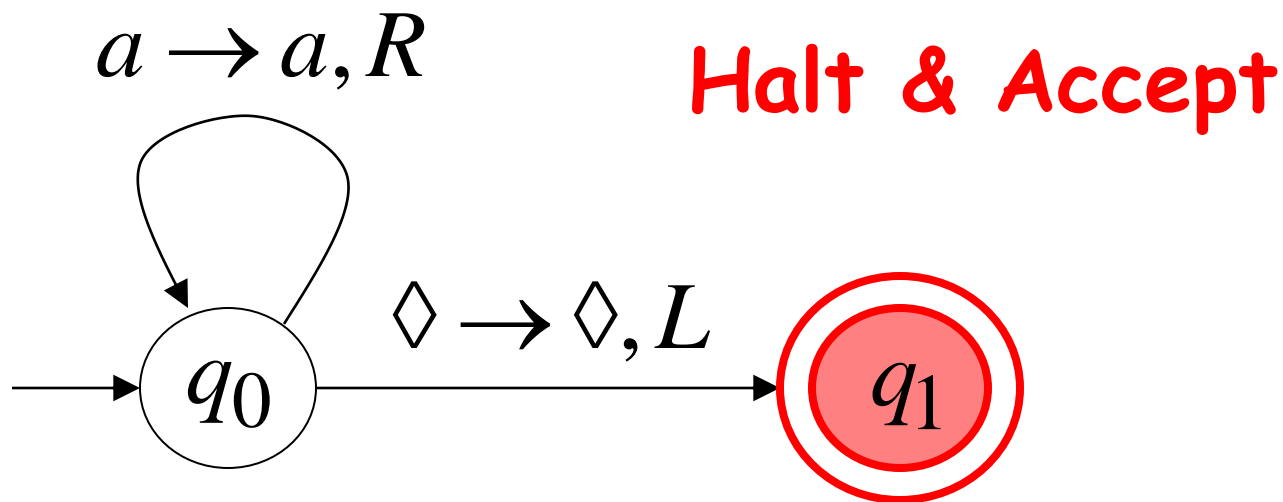
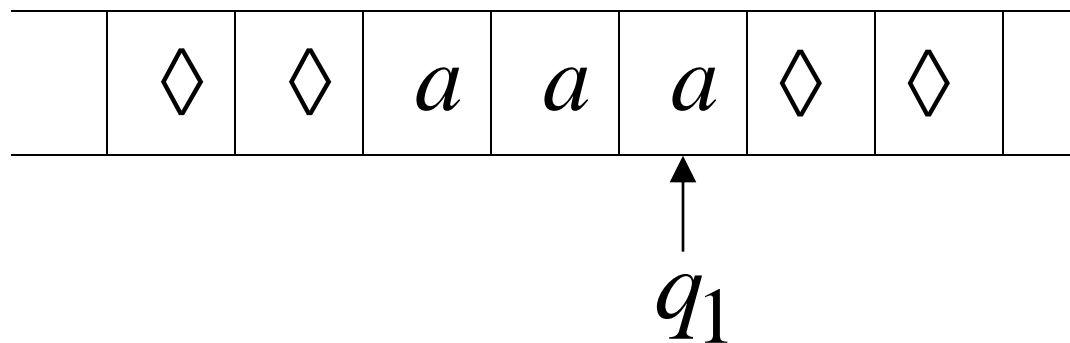
Time 2



Time 3

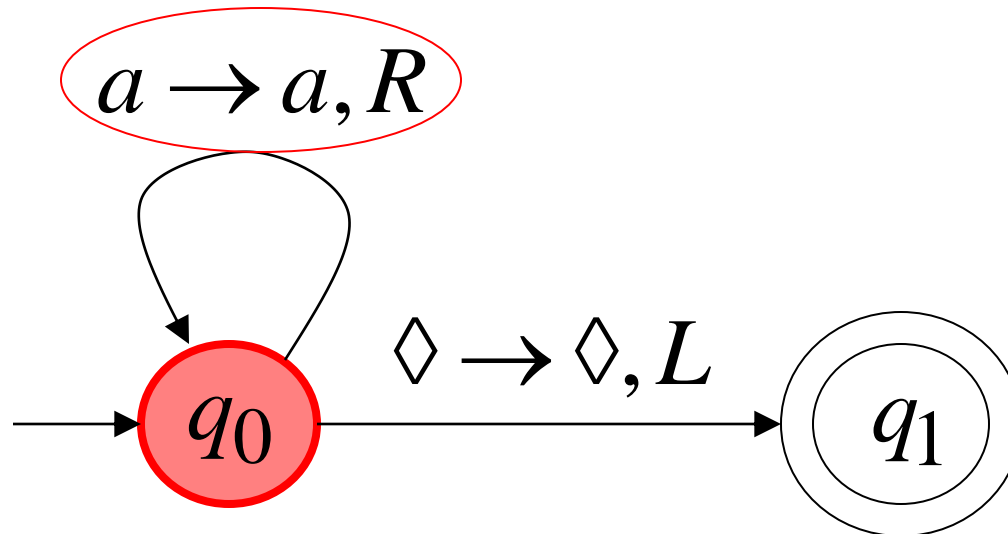
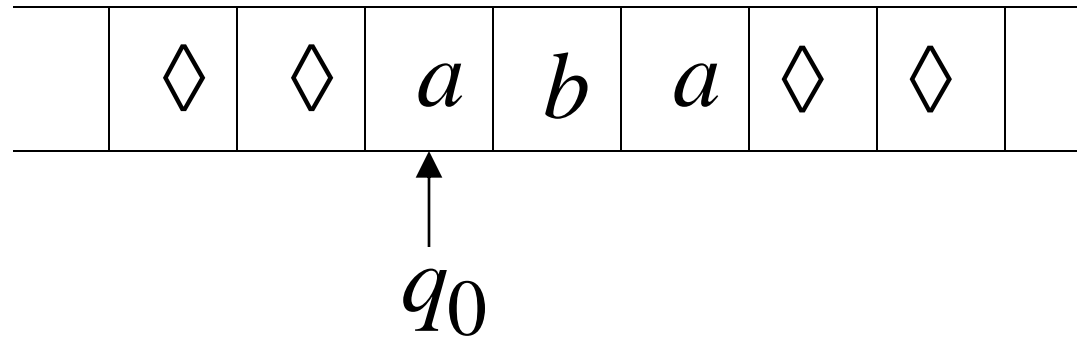


Time 4

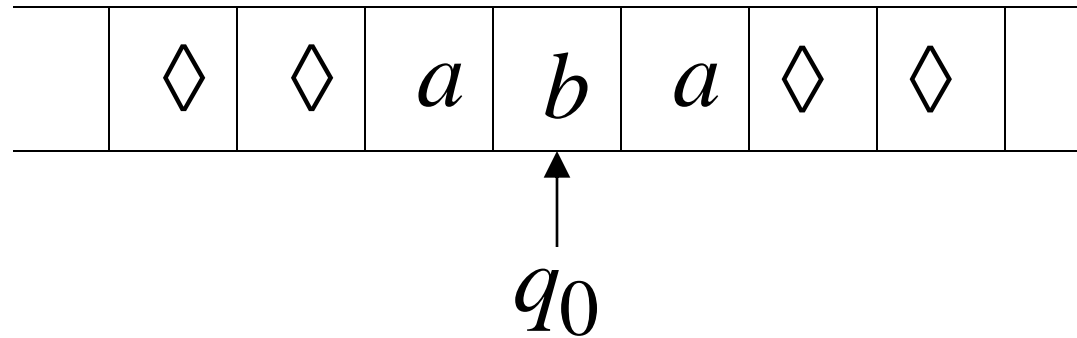


# Rejection Example

Time 0



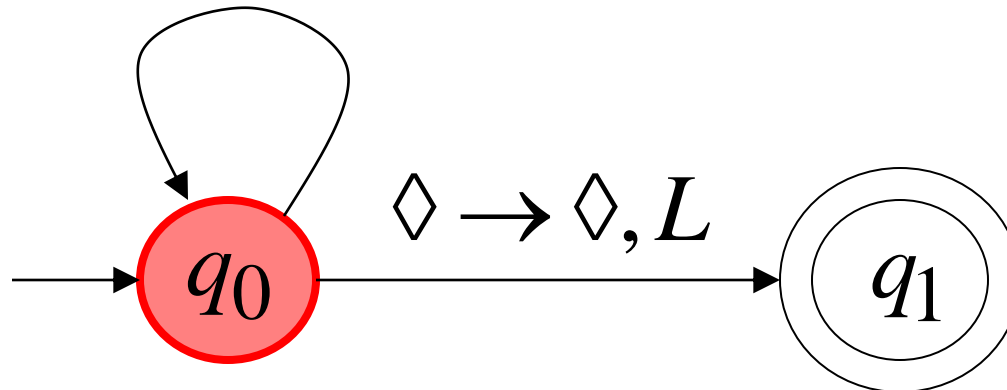
Time 1



No possible Transition

**Halt & Reject**

$a \rightarrow a, R$



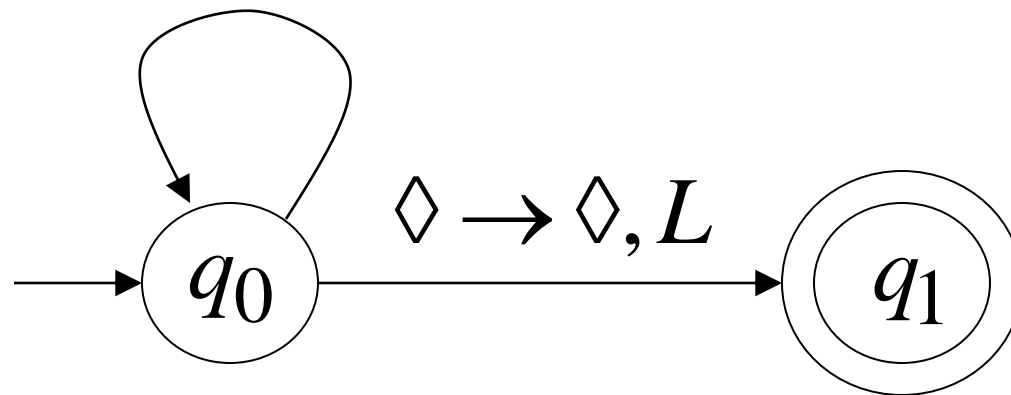
# Infinite Loop Example

A Turing machine

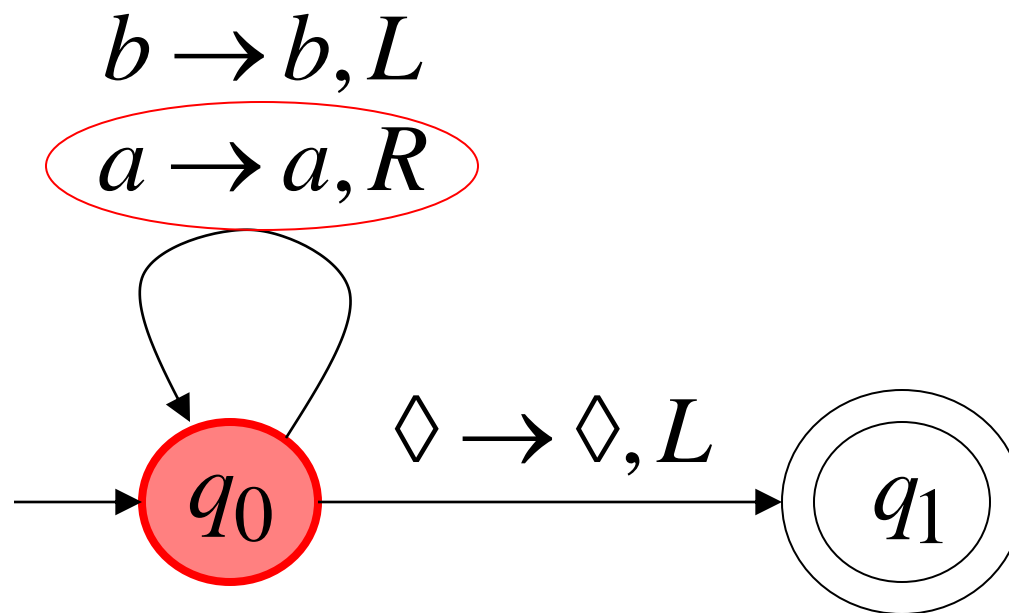
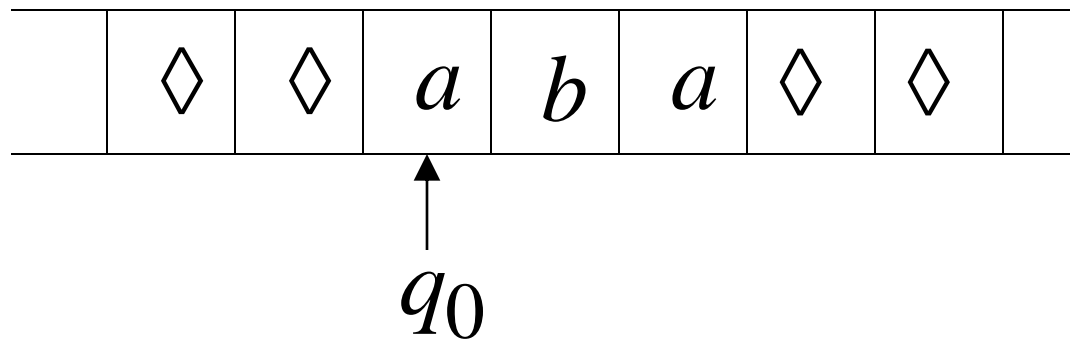
for language  $aa^* + b(a + b)^*$

$b \rightarrow b, L$

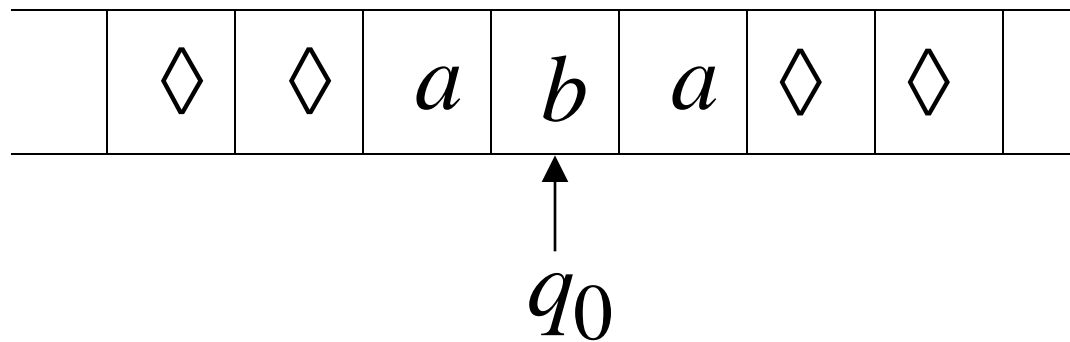
$a \rightarrow a, R$



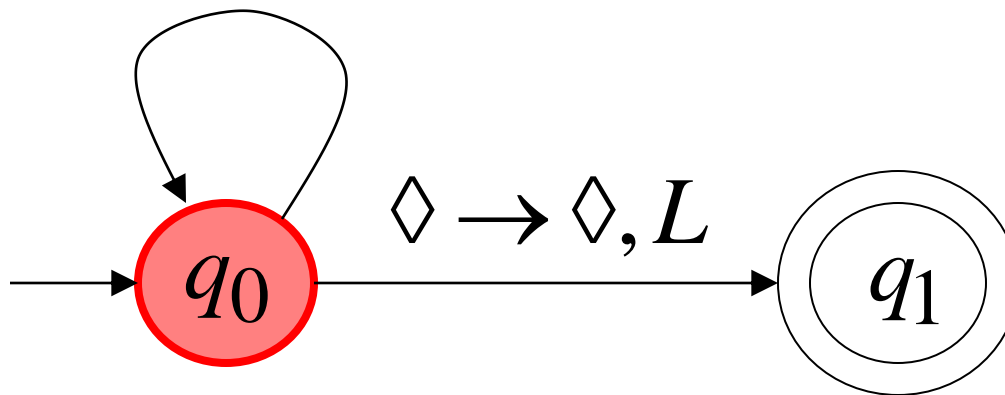
Time 0



Time 1

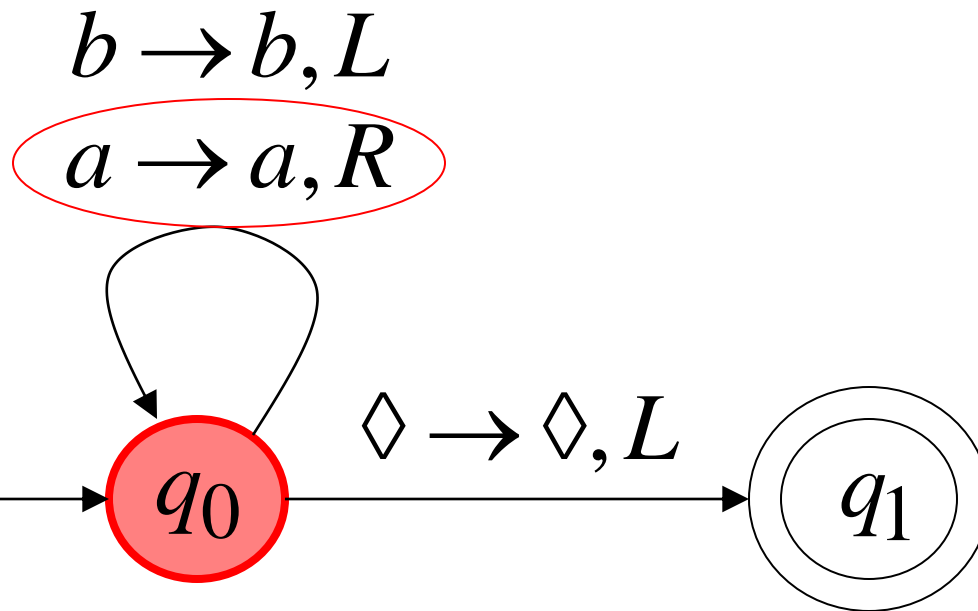
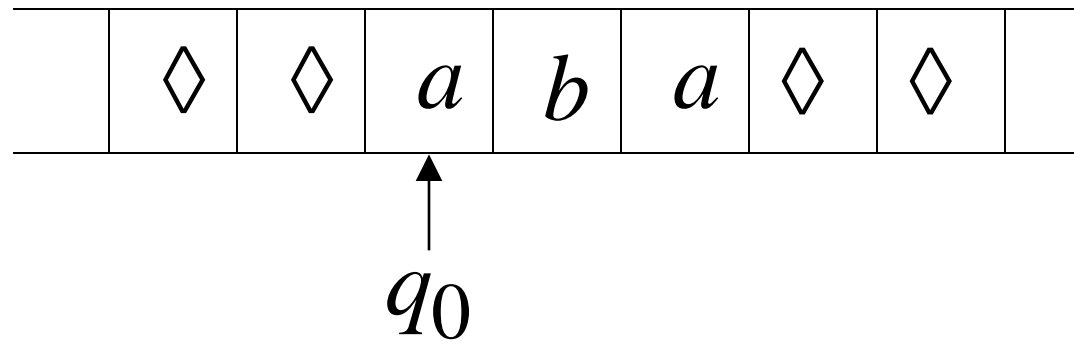


$b \rightarrow b, L$   
 $a \rightarrow a, R$

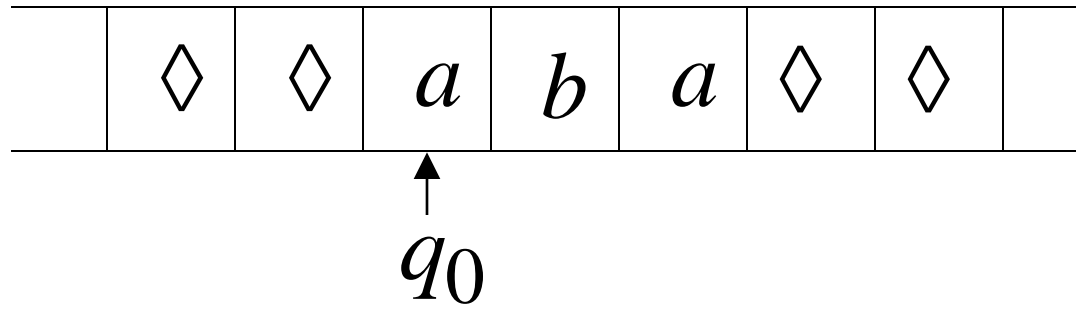




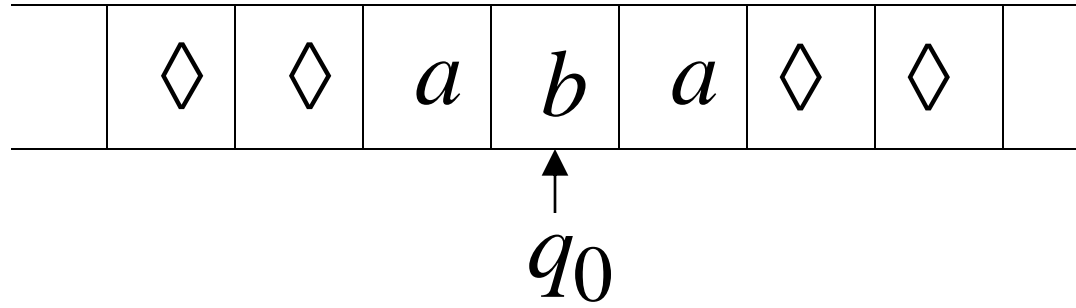
Time 2



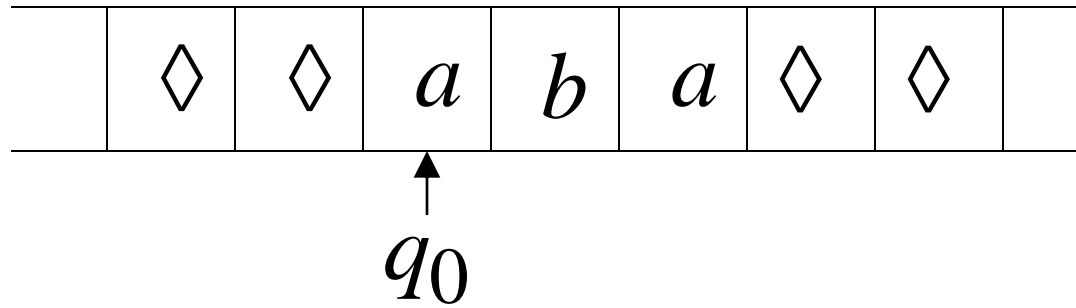
Time 2



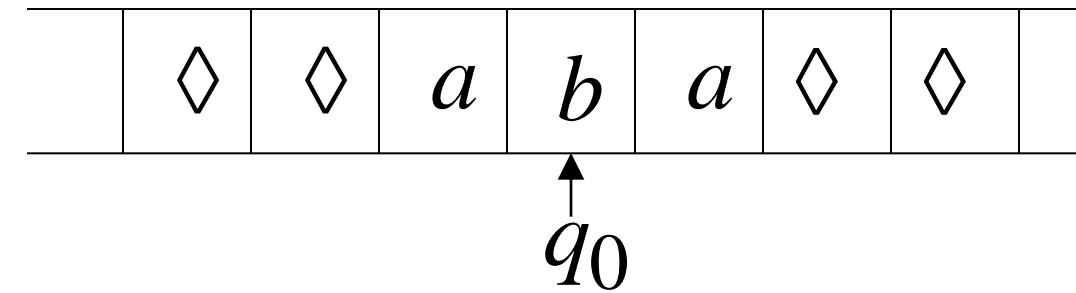
Time 3



Time 4



Time 5



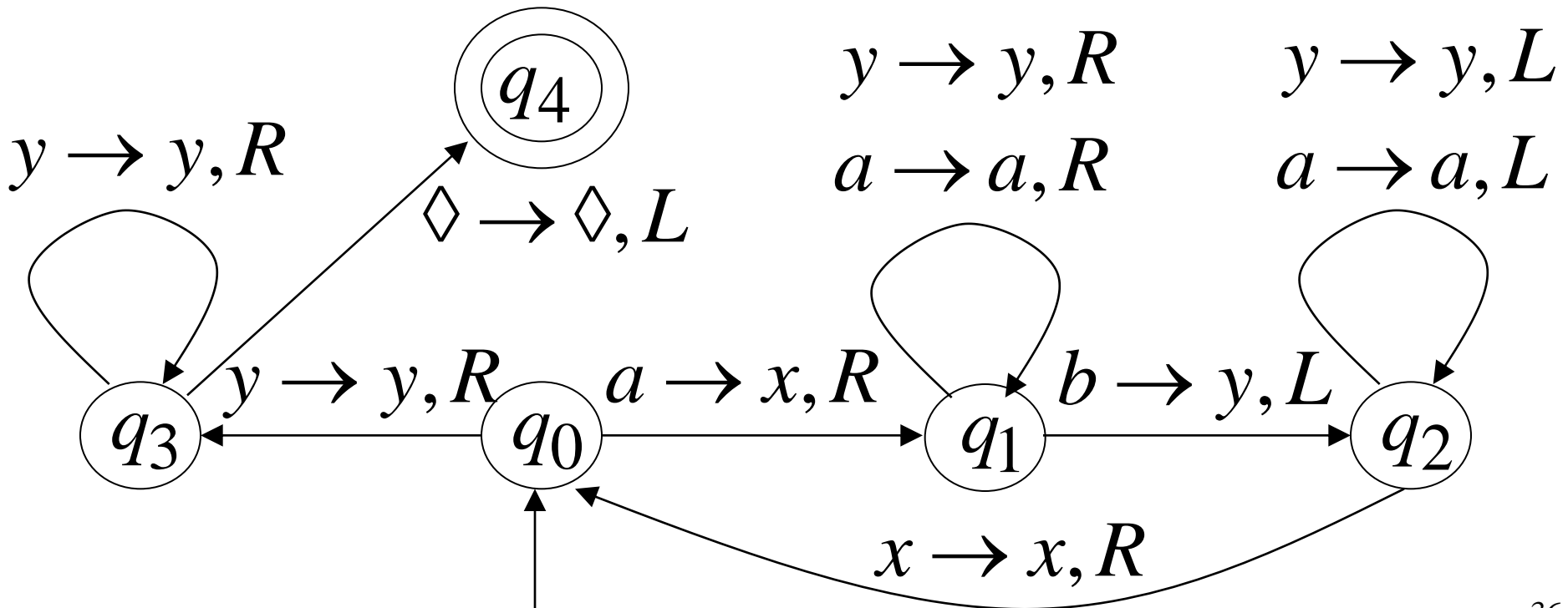
Infinite loop

Because of the **infinite loop**:

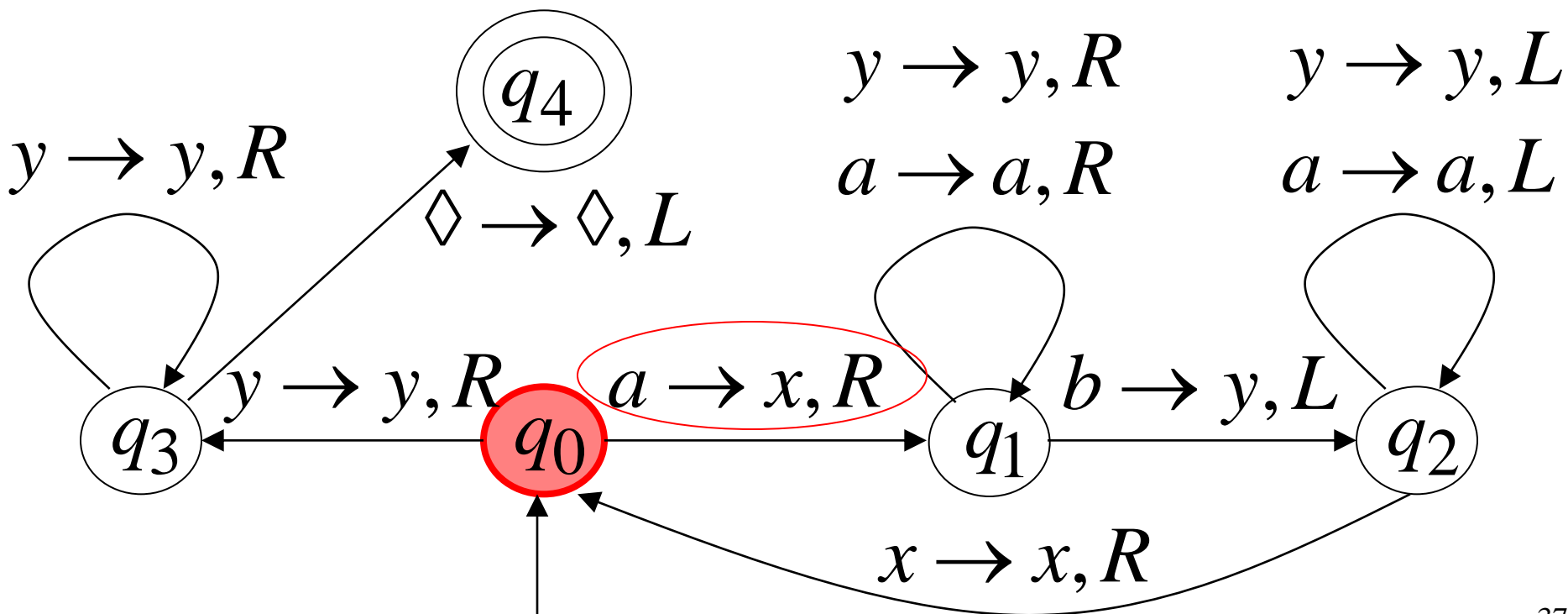
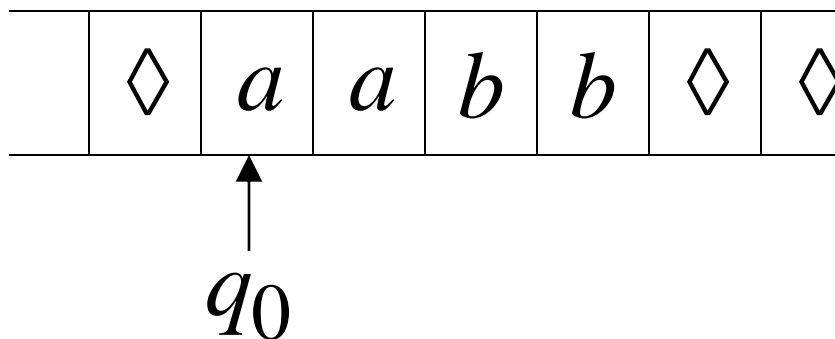
- The final state cannot be reached
- The machine never halts
- The input is **not accepted**

# Another Turing Machine Example

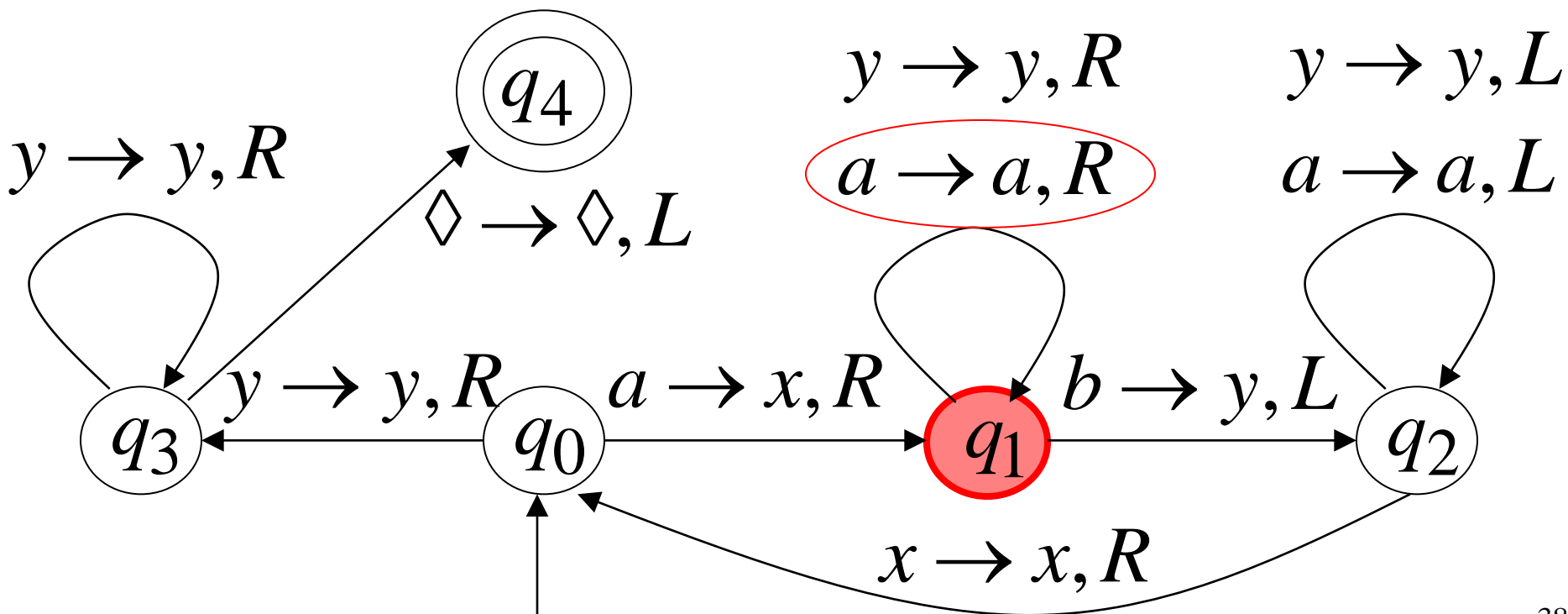
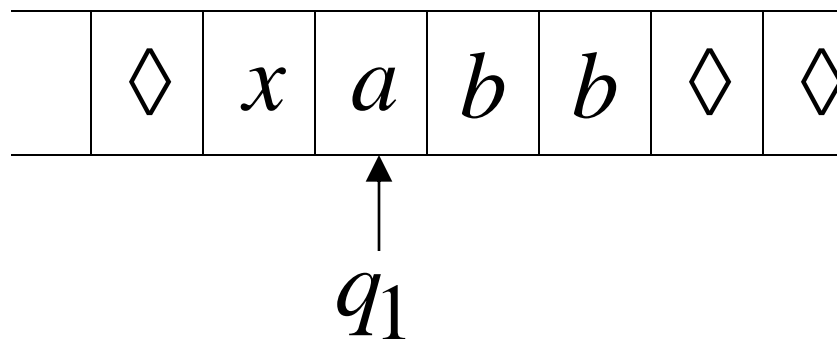
Turing machine for the language  $\{a^n b^n\}$



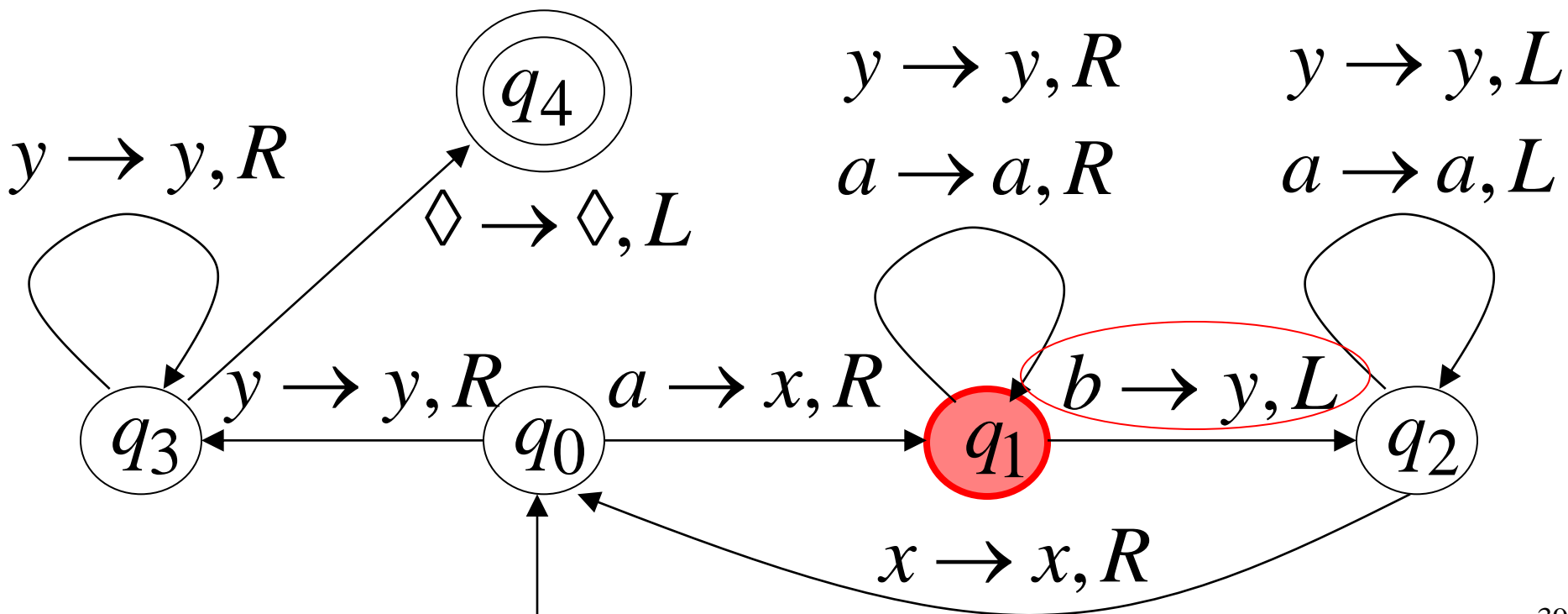
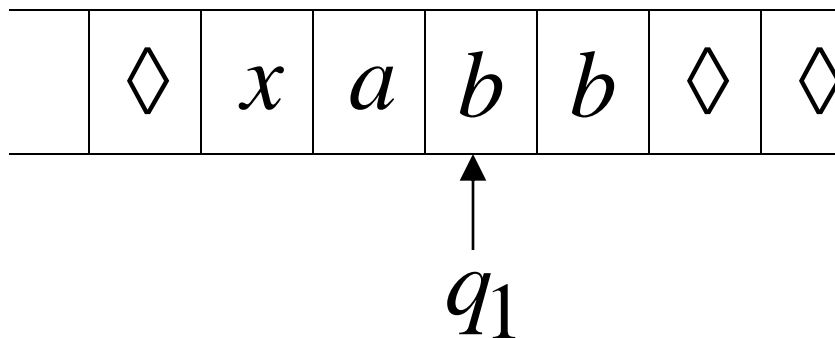
Time 0



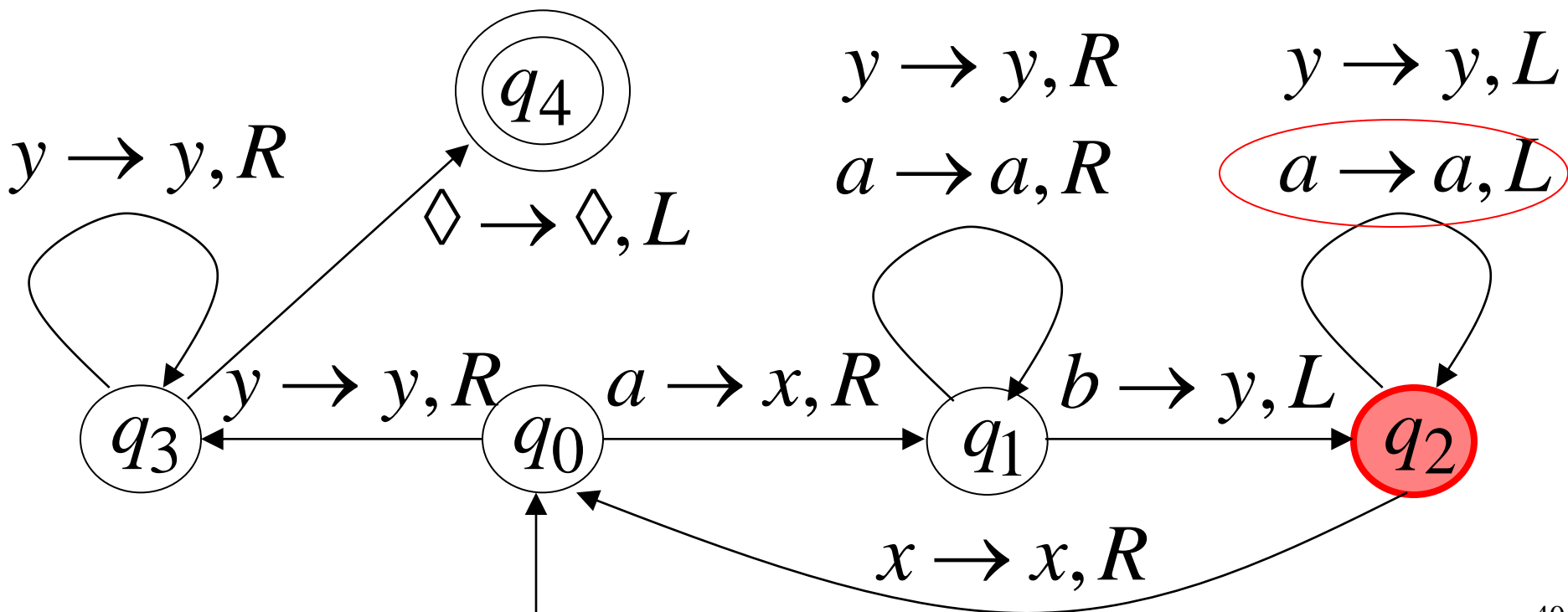
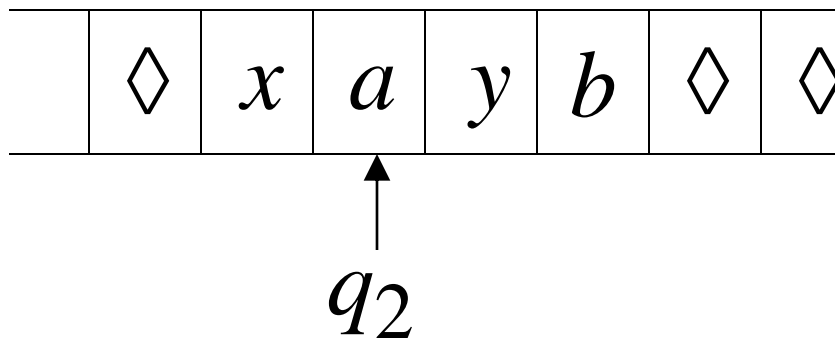
Time 1



Time 2

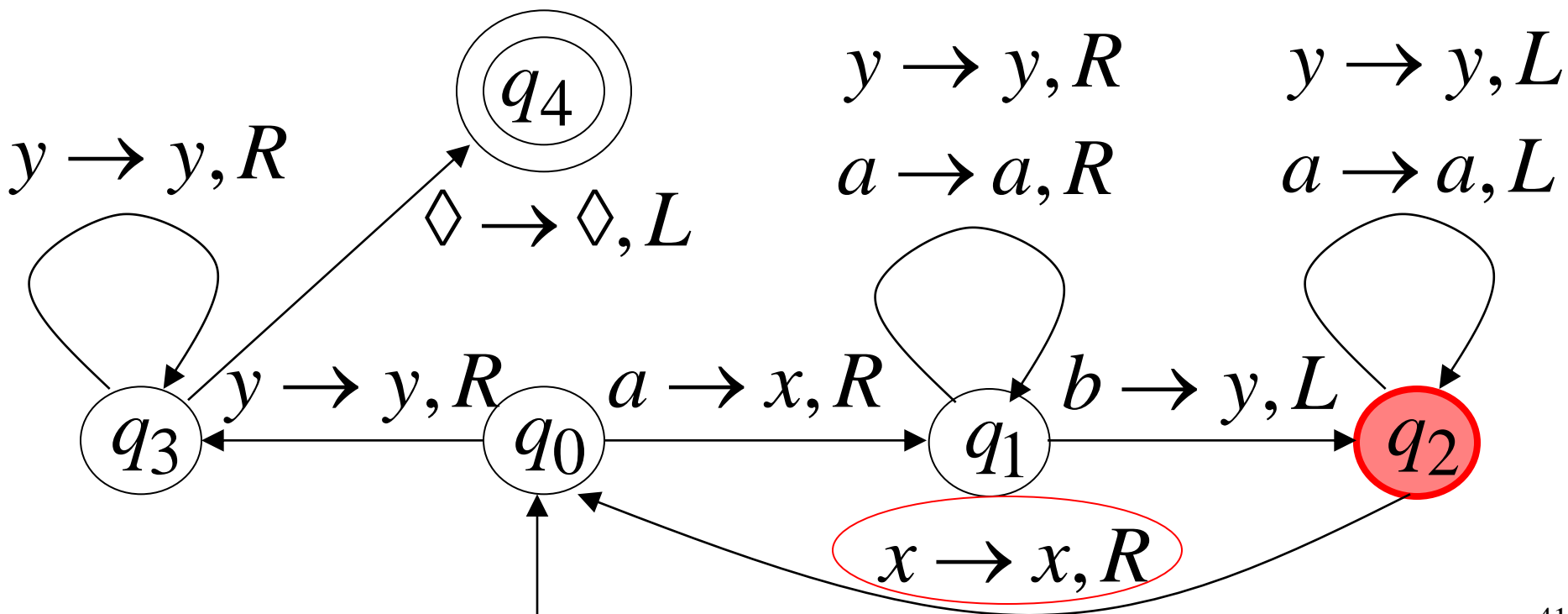
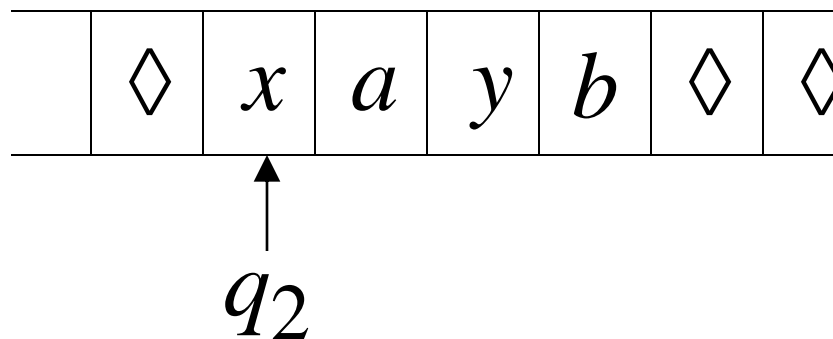


Time 3

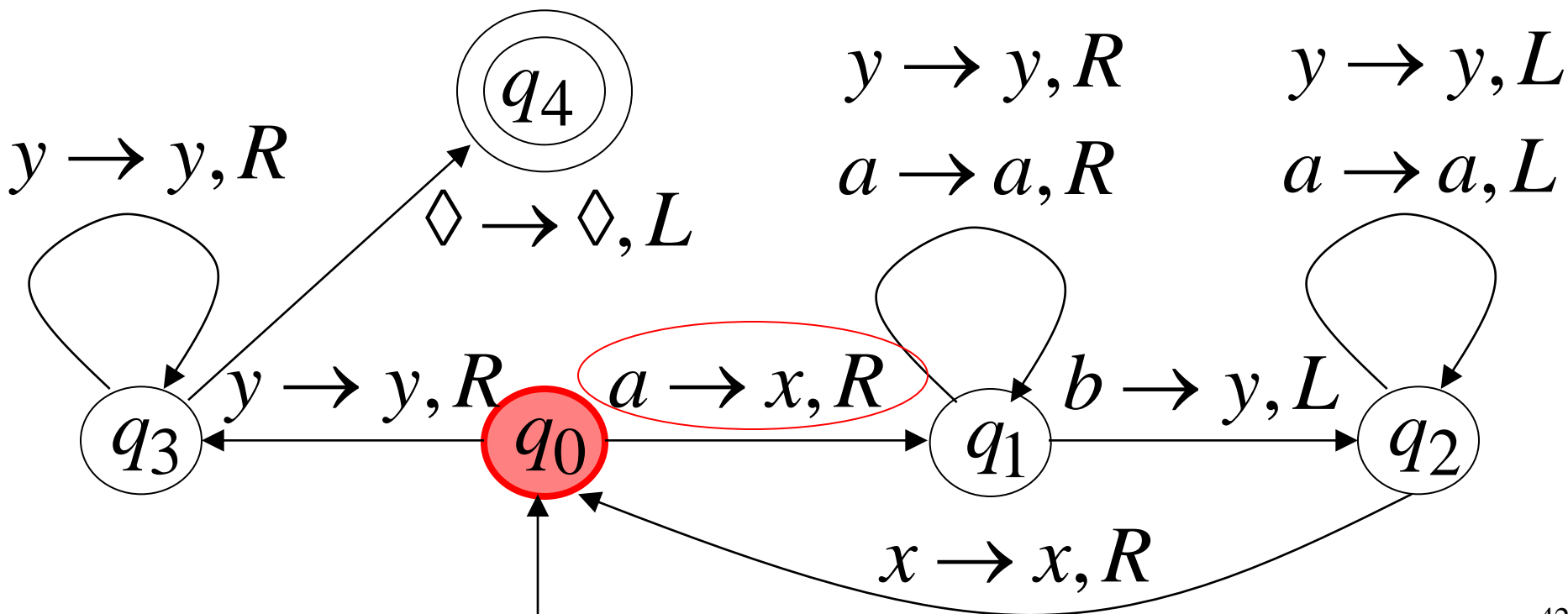
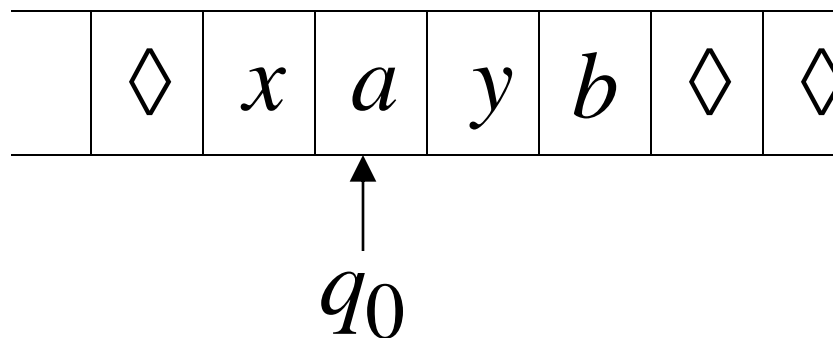




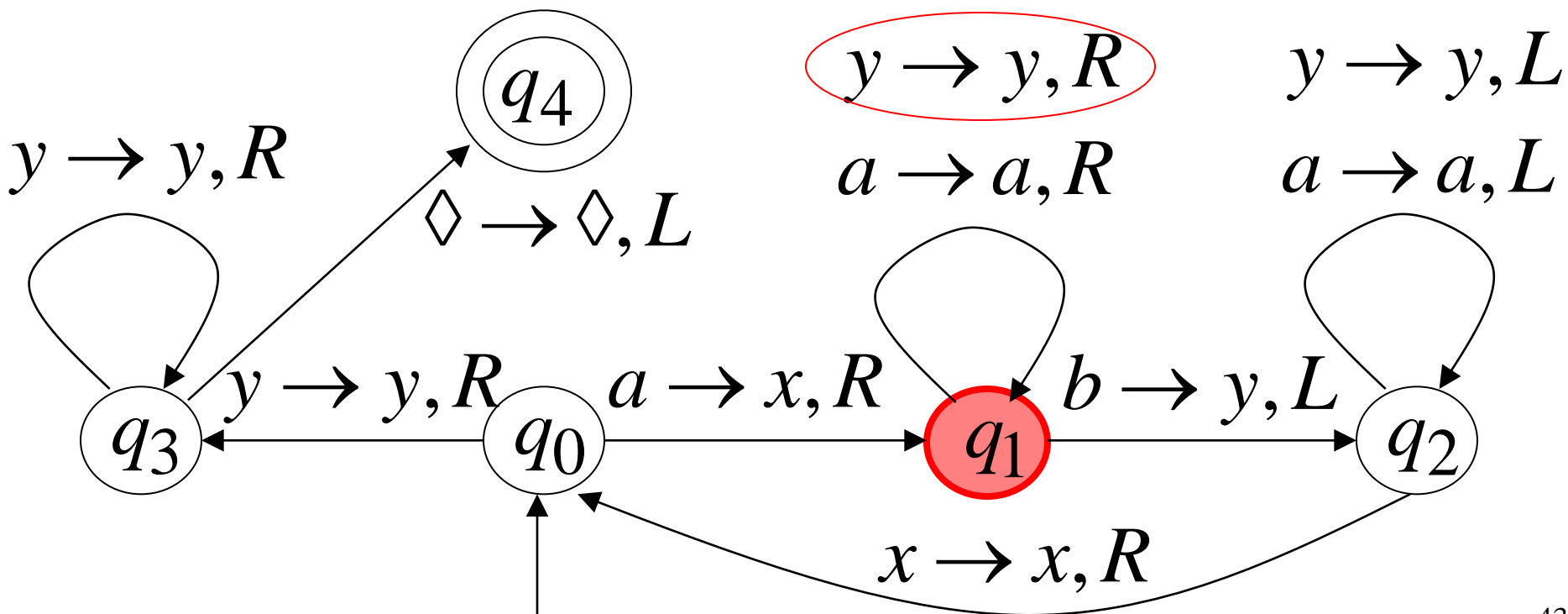
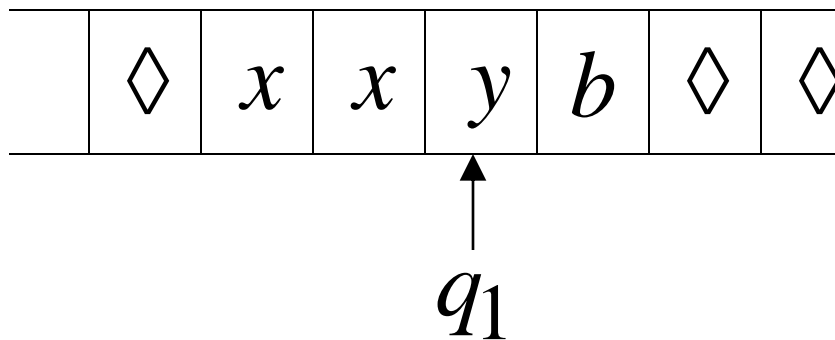
Time 4



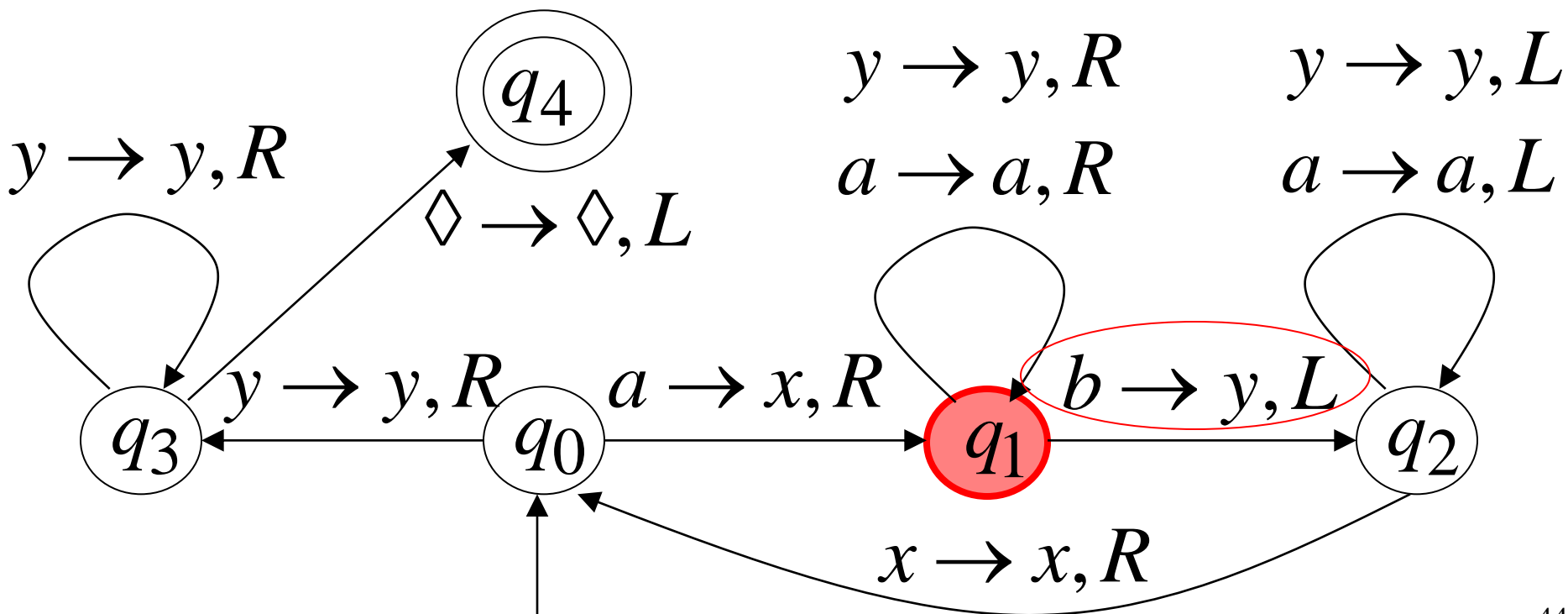
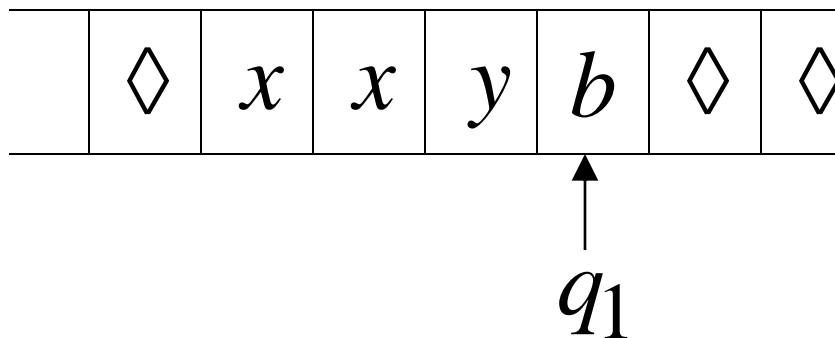
Time 5



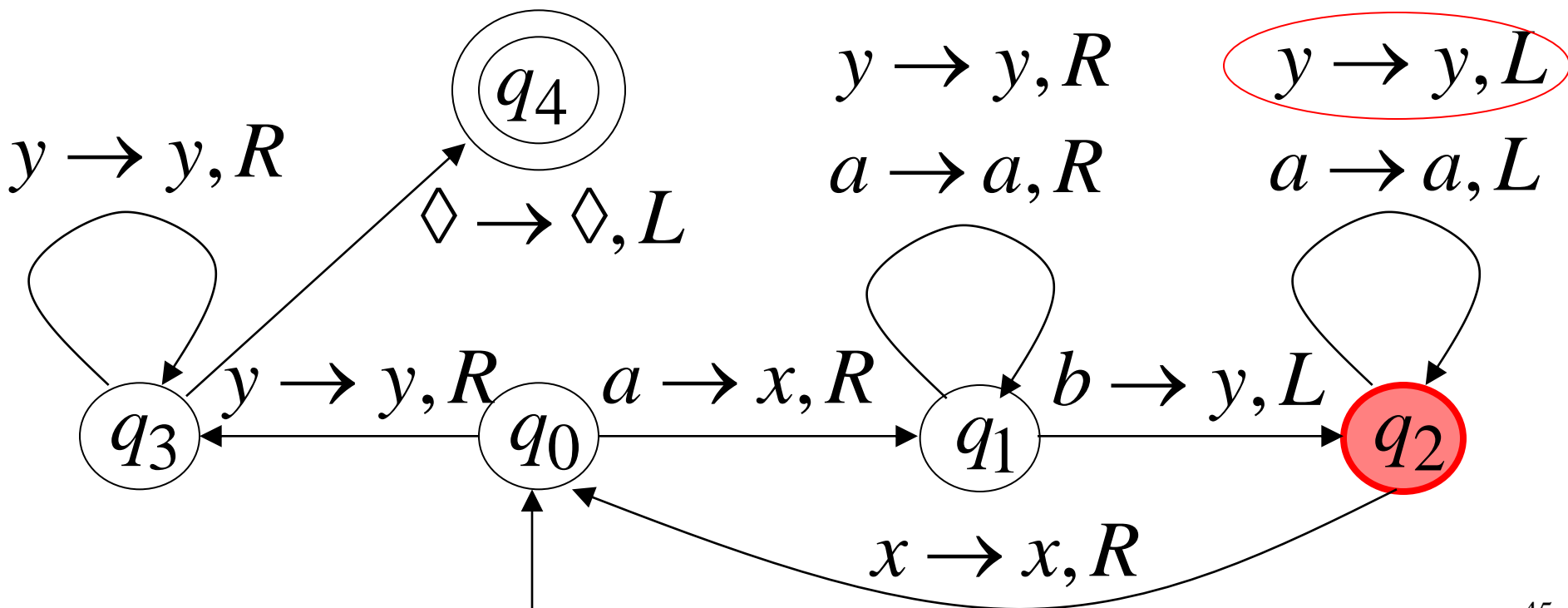
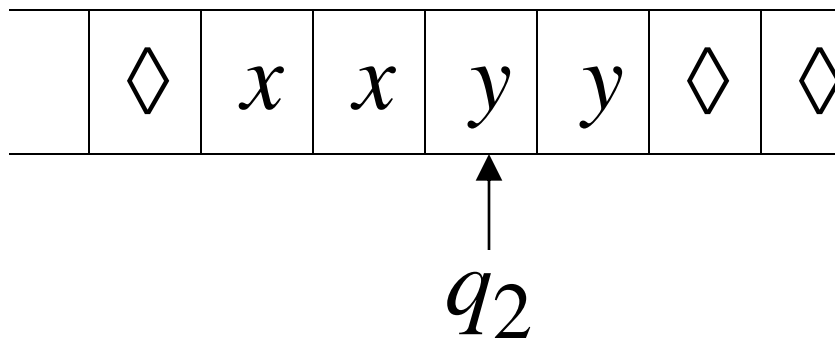
Time 6



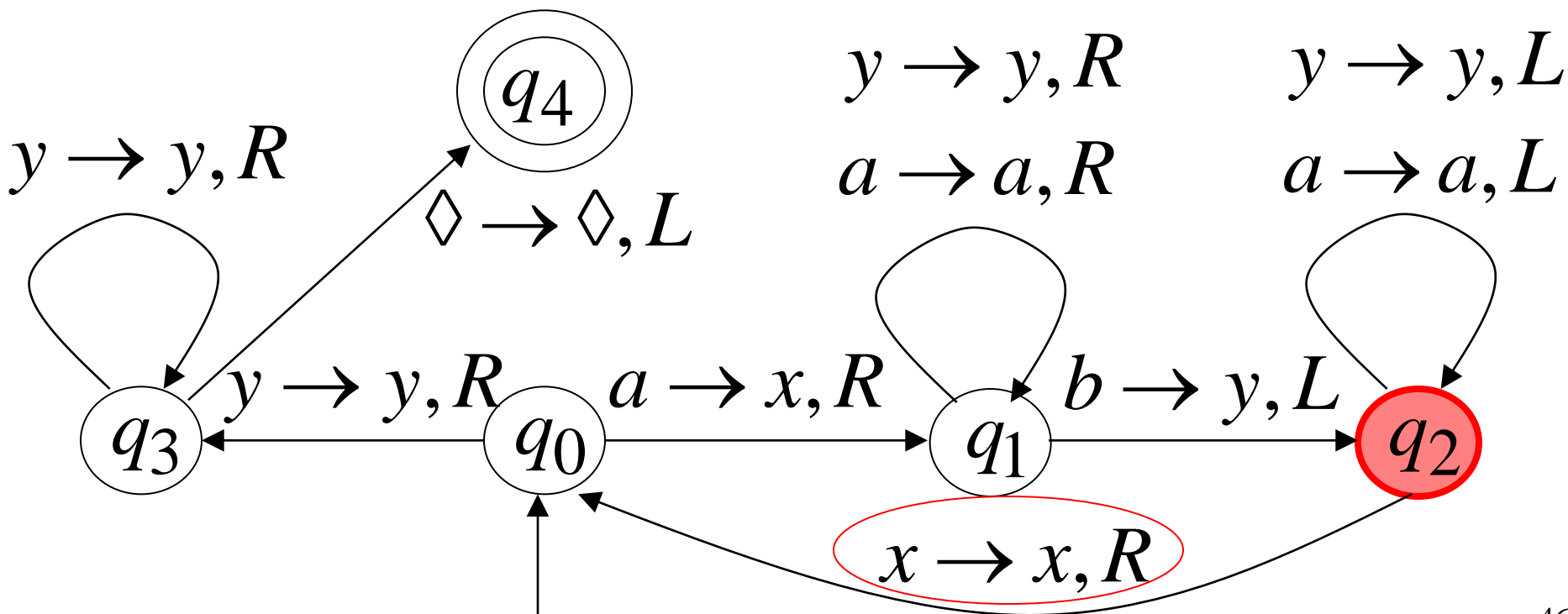
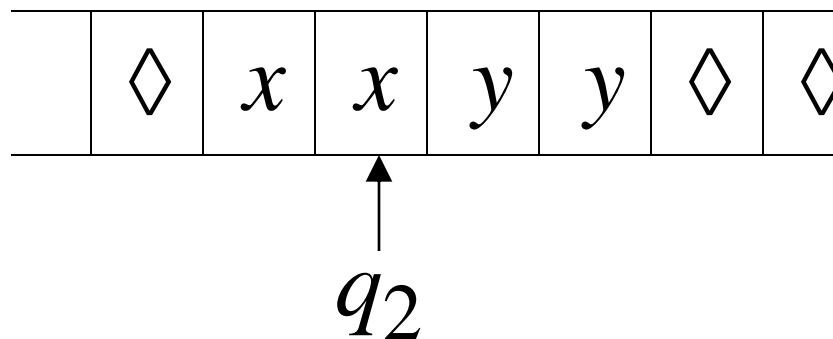
Time 7



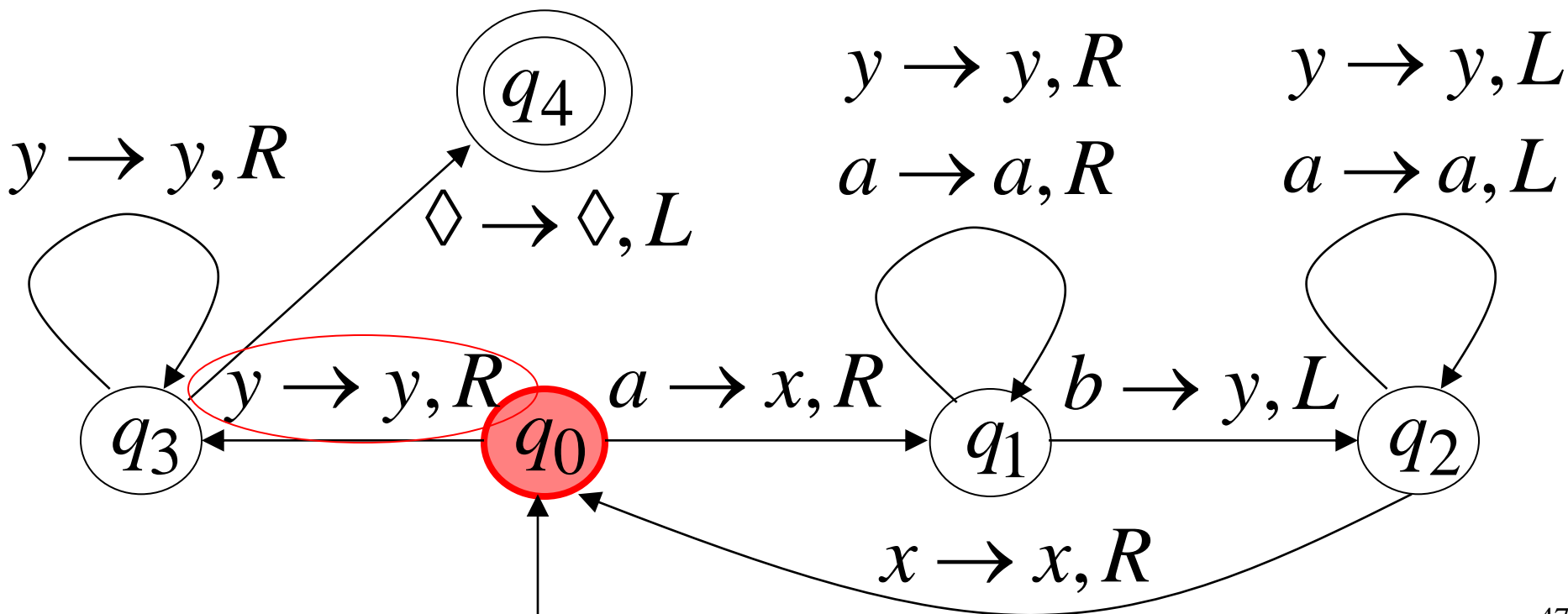
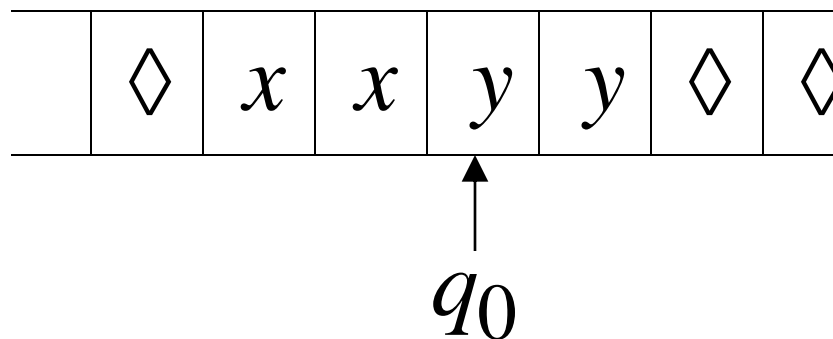
Time 8



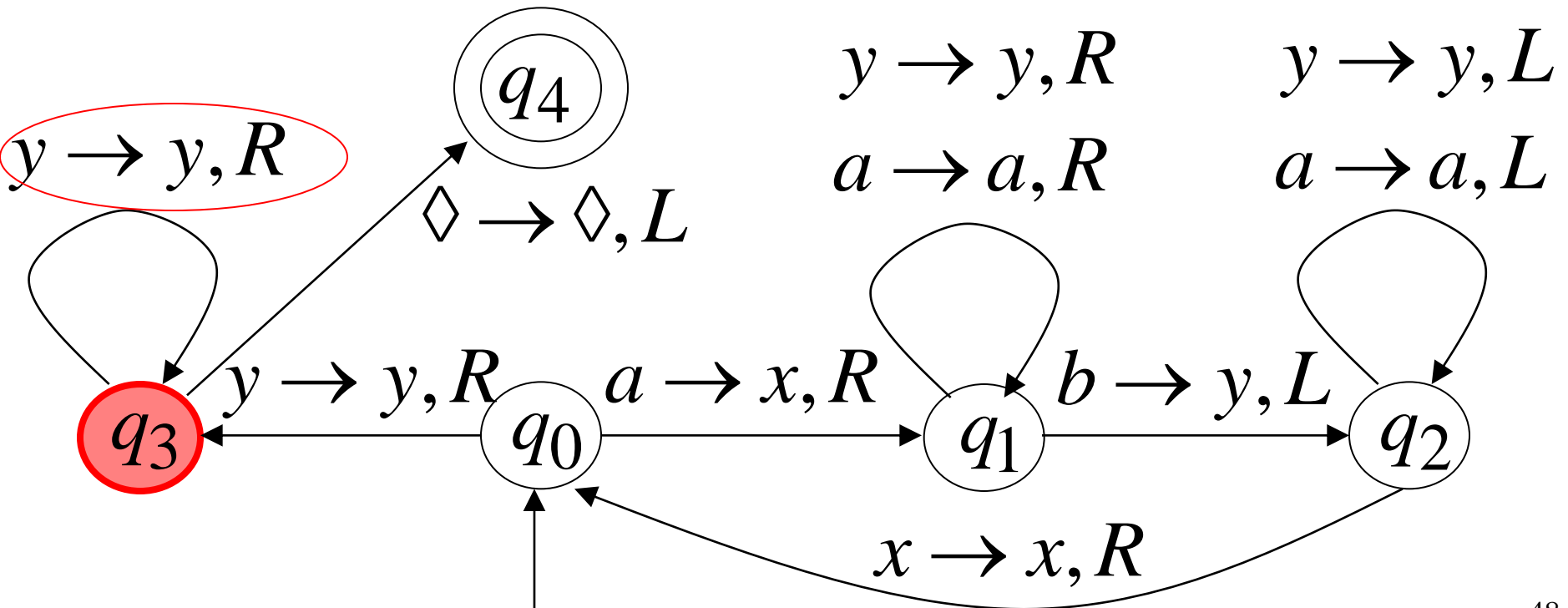
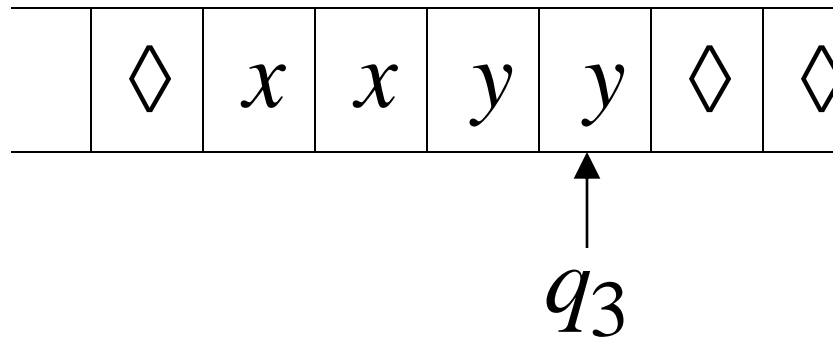
Time 9



Time 10

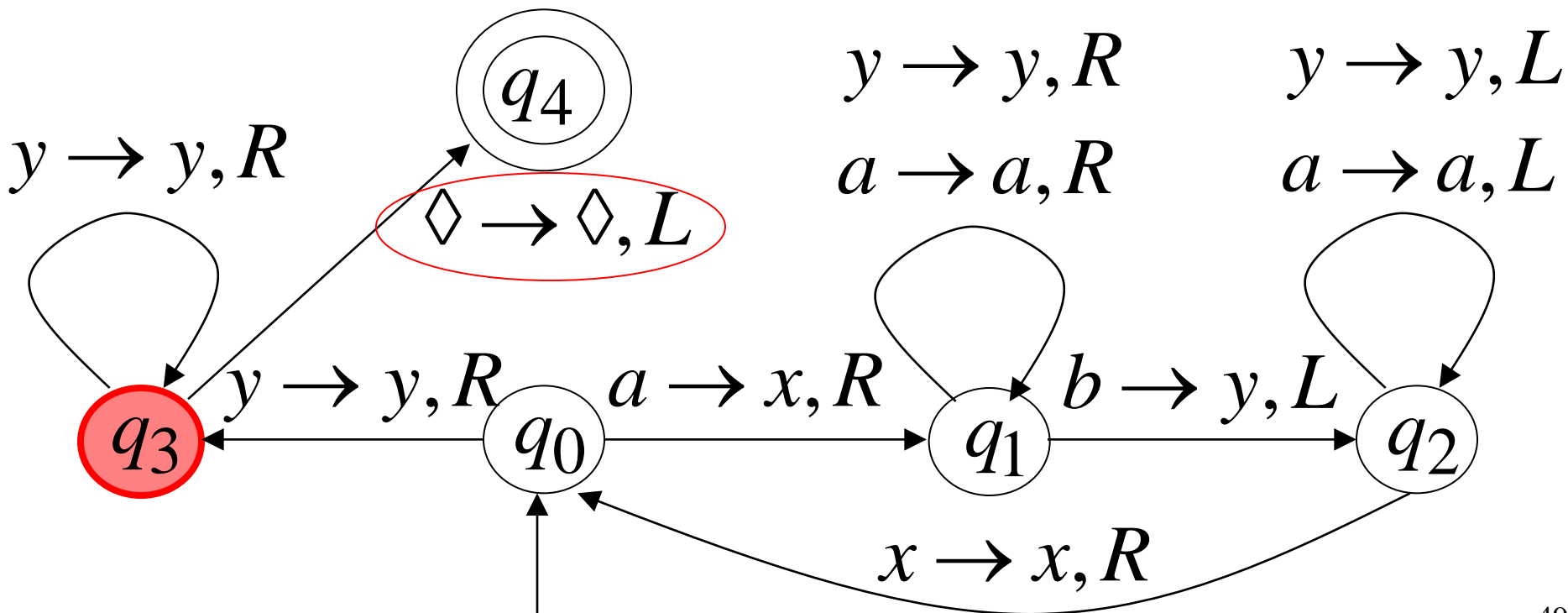
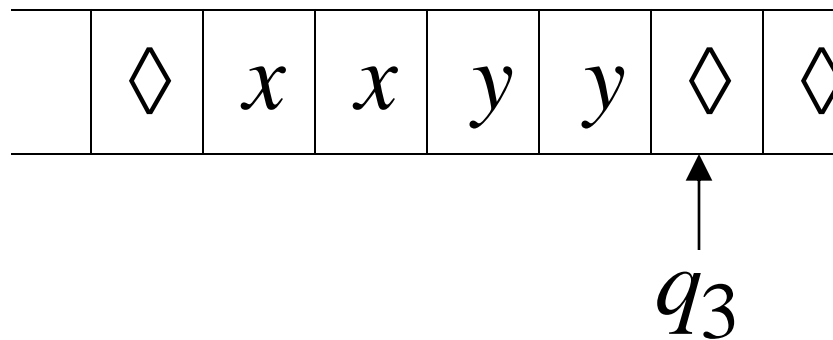


Time 11

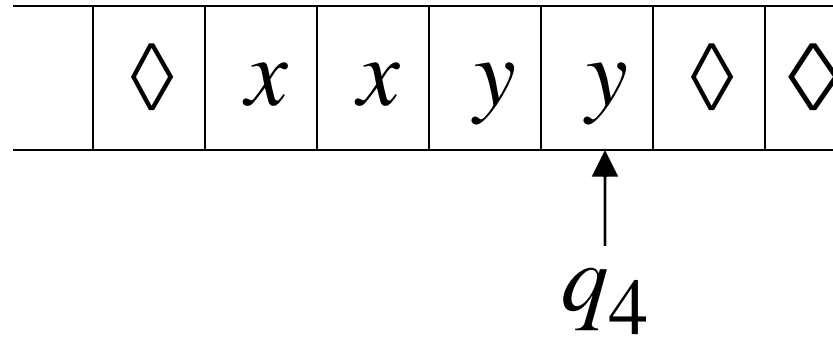




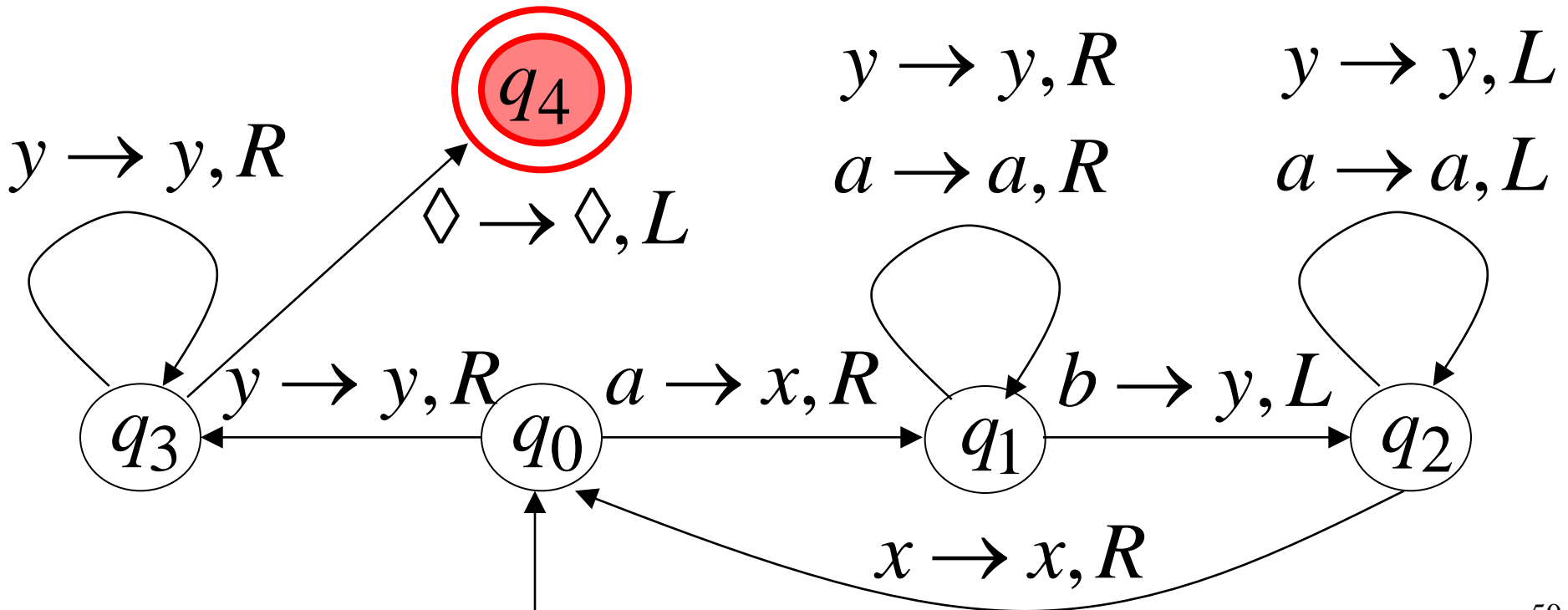
Time 12



Time 13



**Halt & Accept**



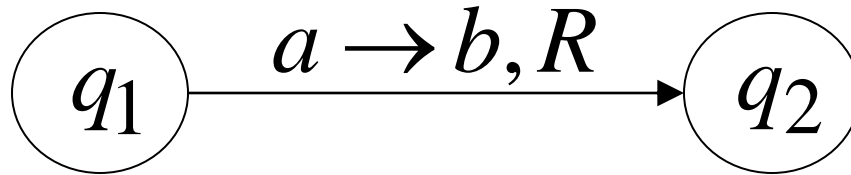
## Observation:

If we modify the machine for the language  $\{a^n b^n\}$

we can easily construct a machine for the language  $\{a^n b^n c^n\}$

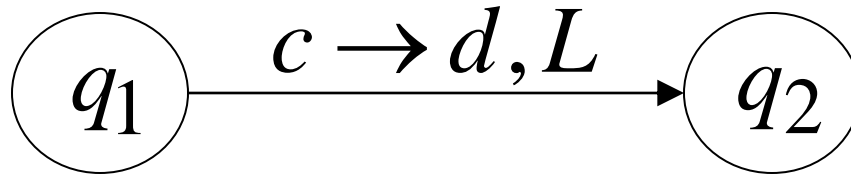
# Formal Definitions for Turing Machines

# Transition Function



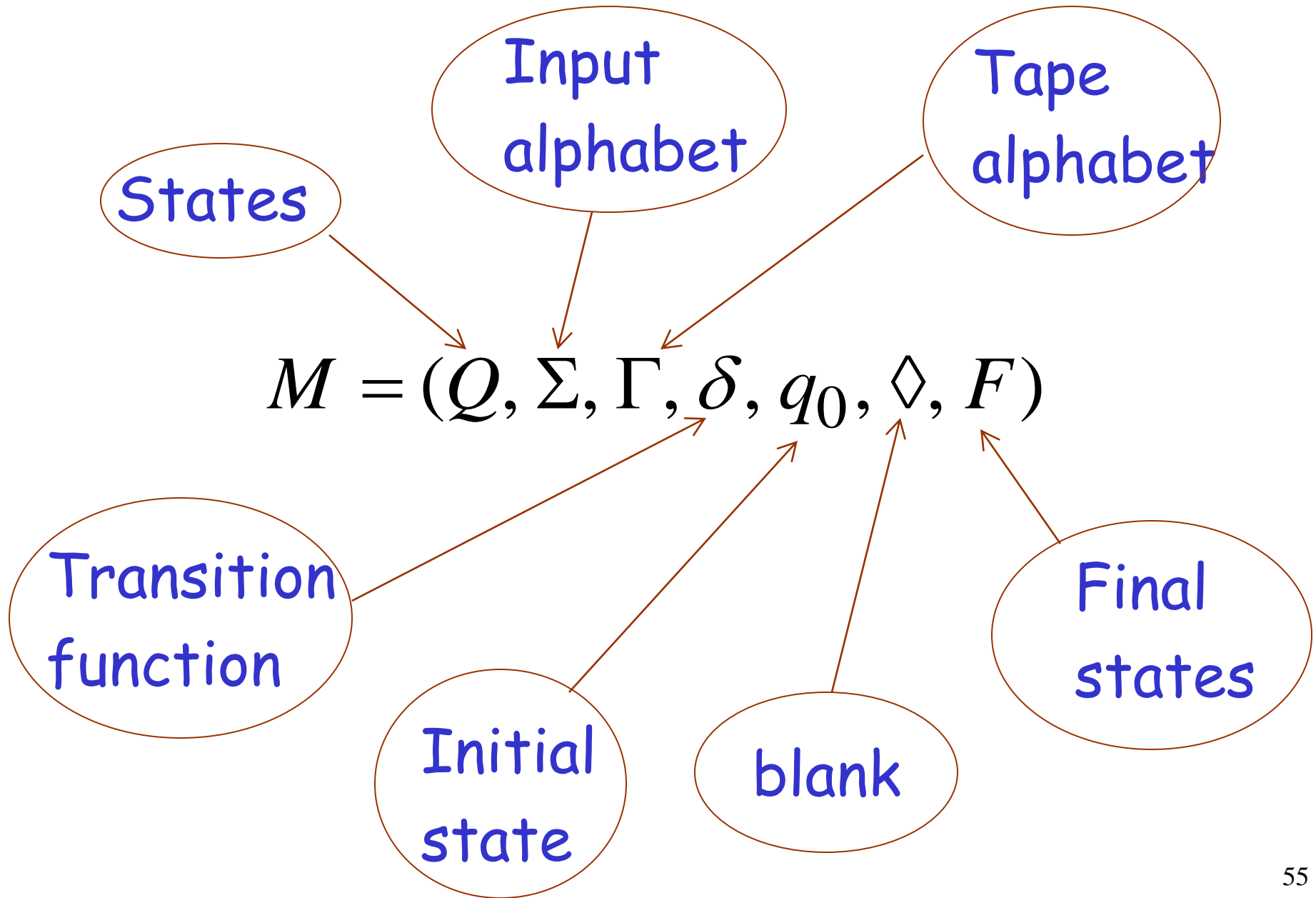
$$\delta(q_1, a) = (q_2, b, R)$$

# Transition Function

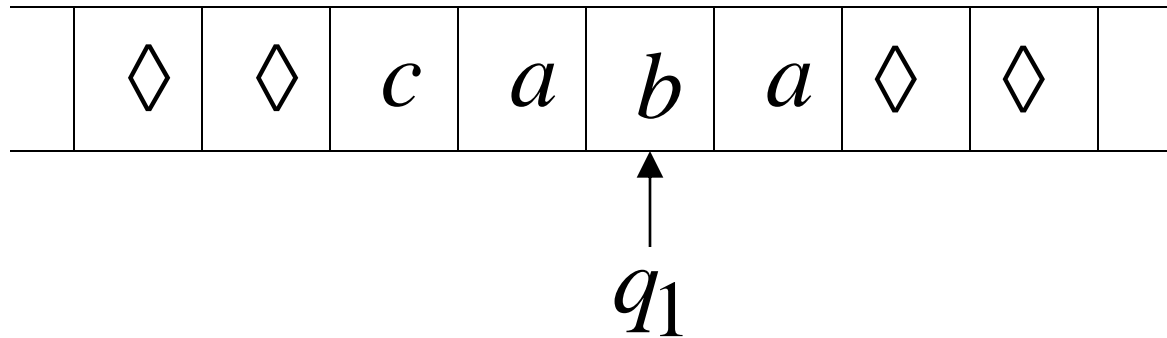


$$\delta(q_1, c) = (q_2, d, L)$$

# Turing Machine:



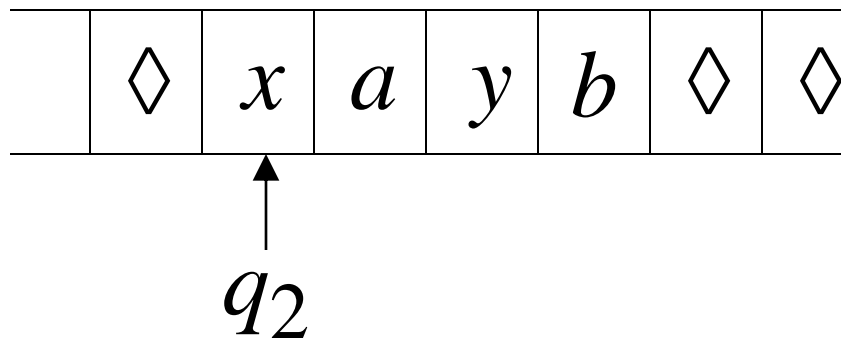
# Configuration



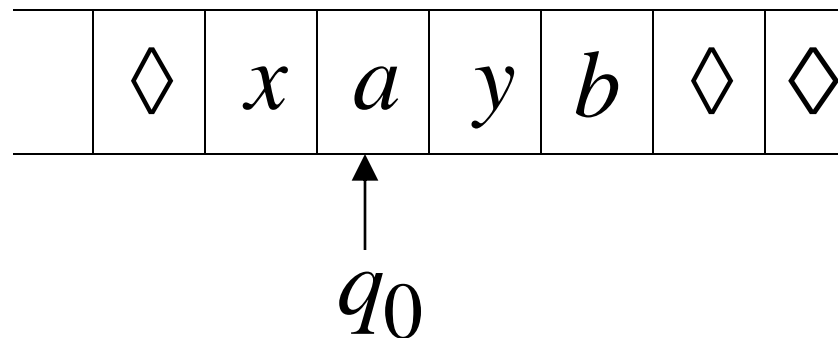
Instantaneous description:  $ca q_1 ba$



Time 4

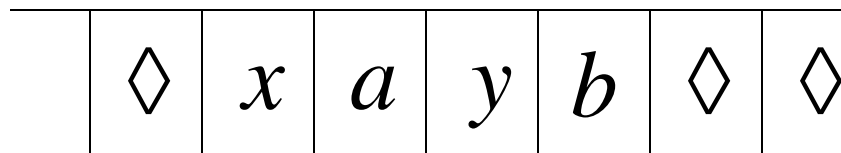


Time 5



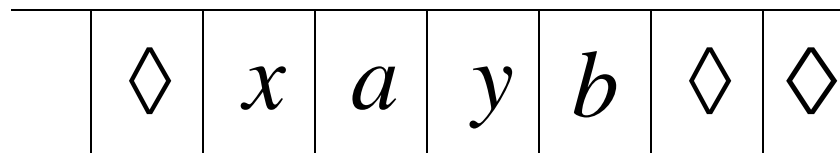
**A Move:**  $q_2 \ x a y b \succ x \ q_0 \ a y b$

Time 4



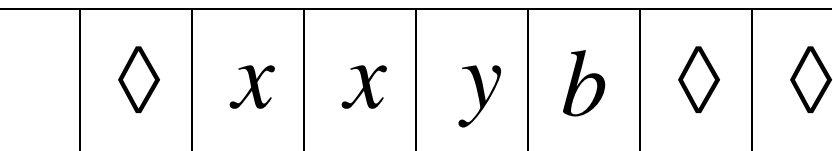
$q_2$

Time 5



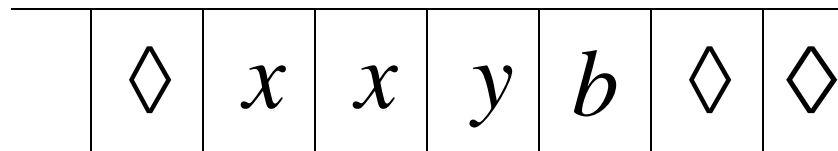
$q_0$

Time 6



$q_1$

Time 7



$q_1$

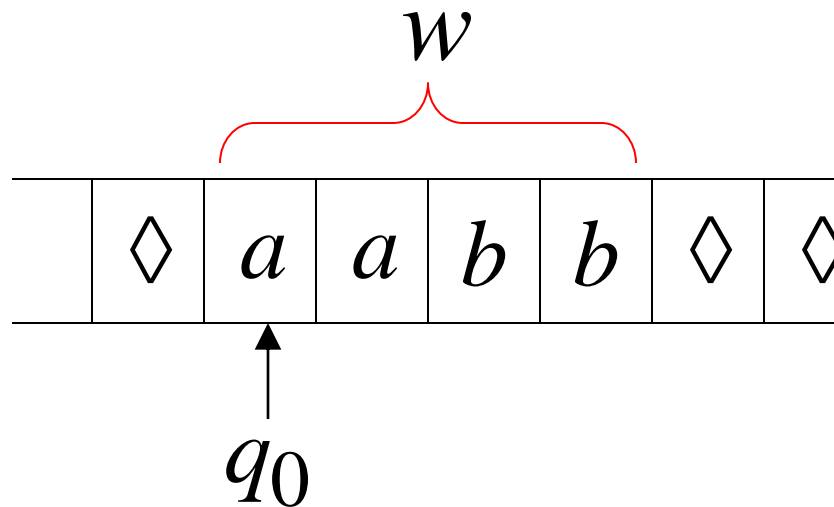
$q_2 \ x a y b \succ x \ q_0 \ a y b \succ x x \ q_1 \ y b \succ x x y \ q_1 \ b$

$$q_2 xayb \succ x q_0 ayb \succ xx q_1 yb \succ xxy q_1 b$$

Equivalent notation:  $q_2 xayb \overset{*}{\succ} xxy q_1 b$

Initial configuration:  $q_0 w$

Input string



# The Accepted Language

For any Turing Machine  $M$

$$L(M) = \{w : q_0 w \xrightarrow{*} x_1 q_f x_2\}$$

Initial state

Final state

# Standard Turing Machine

The machine we described is the standard:

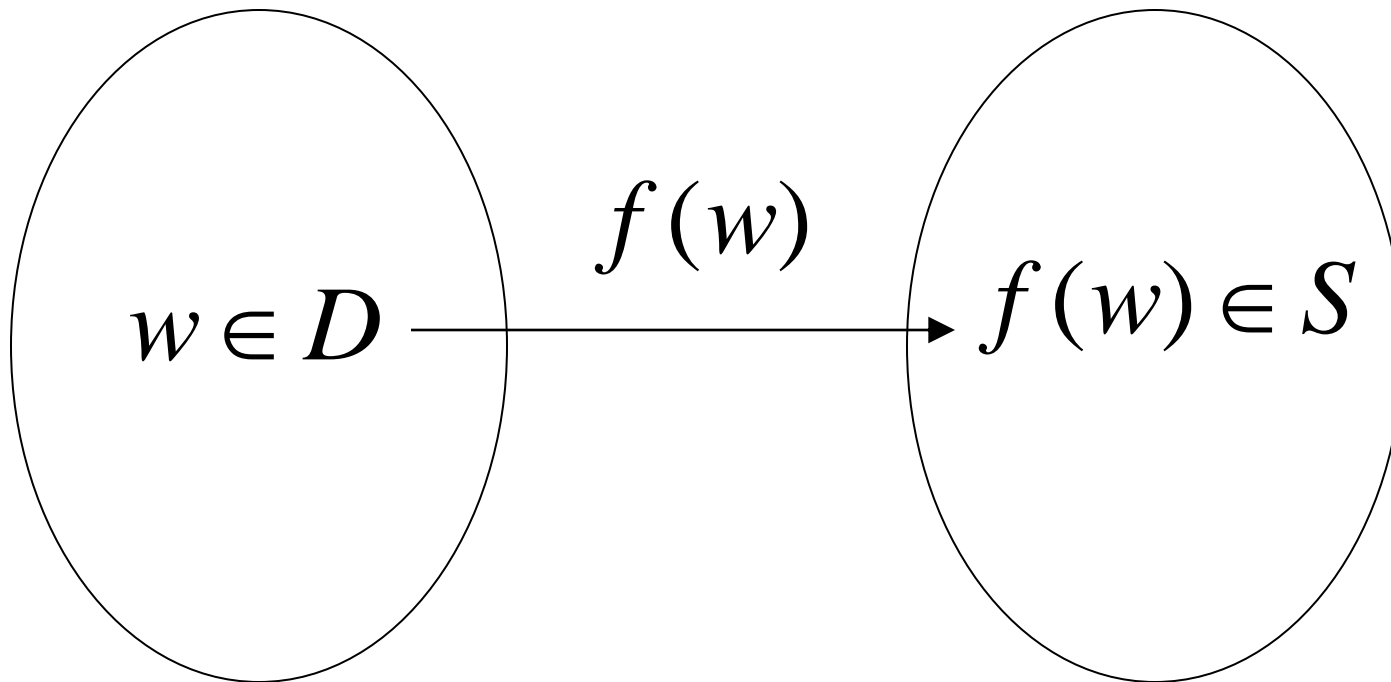
- Deterministic
- Infinite tape in both directions
- Tape is the input/output file

# Computing Functions with Turing Machines

A function  $f(w)$  has:

Domain:  $D$

Result Region:  $S$





A function may have many parameters:

Example: Addition function

$$f(x, y) = x + y$$

# Integer Domain

Decimal: 5

Binary: 101

Unary: 11111

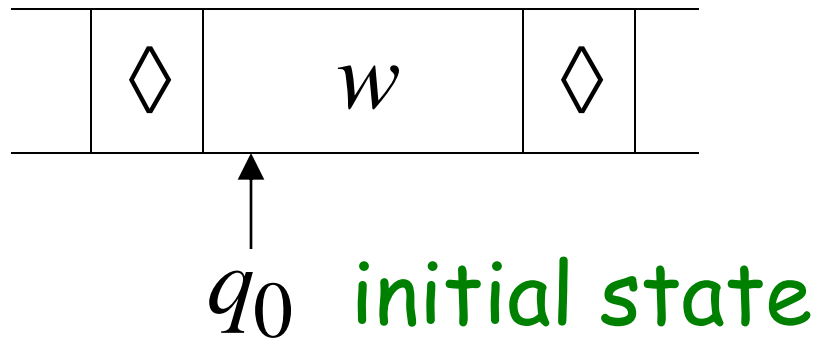
We prefer **unary** representation:

easier to manipulate with Turing machines

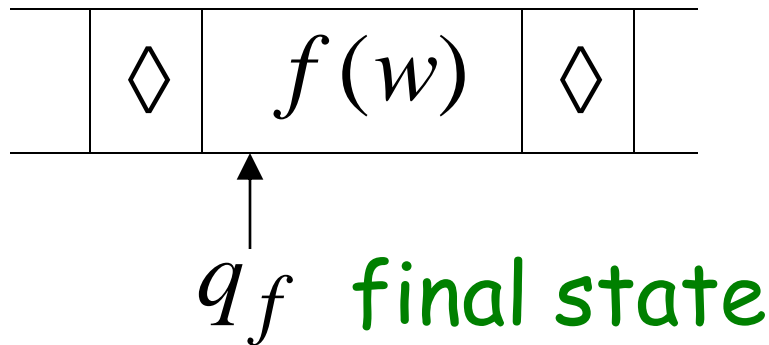
# Definition:

A function  $f$  is computable if there is a Turing Machine  $M$  such that:

Initial configuration



Final configuration



For all  $w \in D$  Domain

In other words:

A function  $f$  is computable if there is a Turing Machine  $M$  such that:

$$q_0 w \xrightarrow{*} q_f f(w)$$



Initial

Configuration



Final

Configuration

For all  $w \in D$  Domain

# Example

The function  $f(x, y) = x + y$  is computable

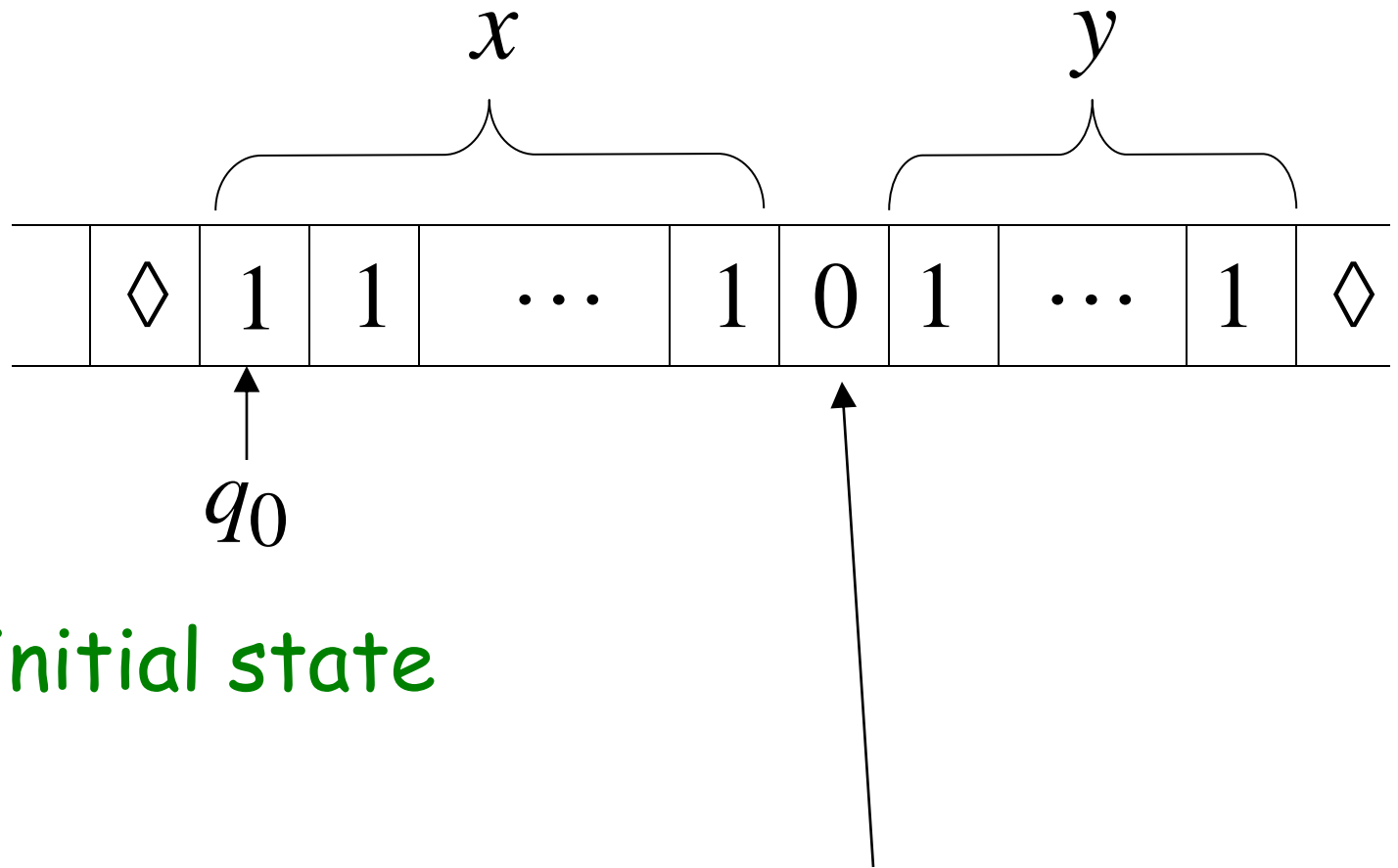
$x, y$  are integers

Turing Machine:

Input string:  $x0y$  unary

Output string:  $xy0$  unary

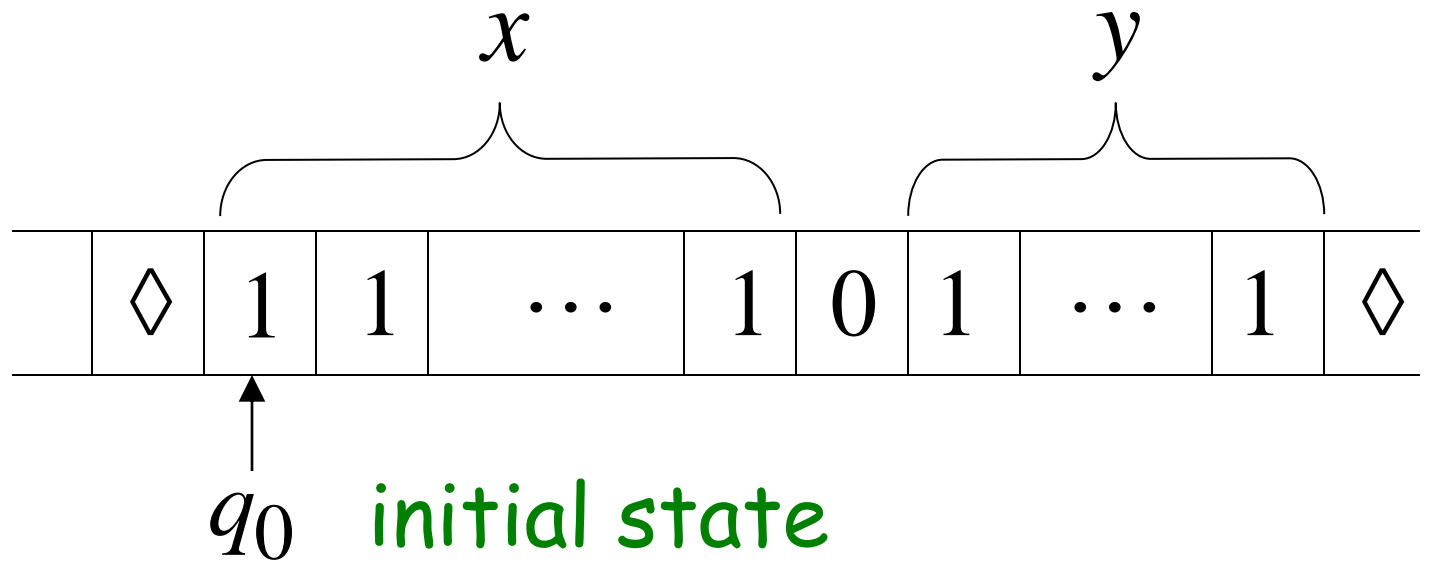
Start



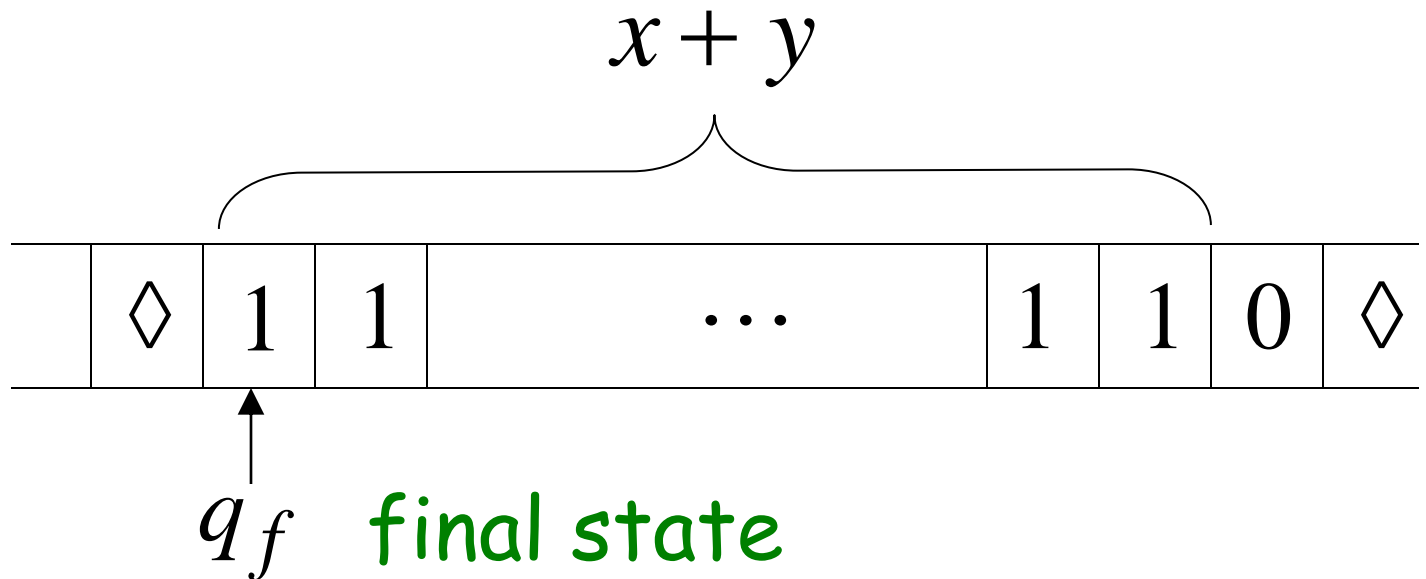
initial state

The 0 is the delimiter that separates the two numbers

Start

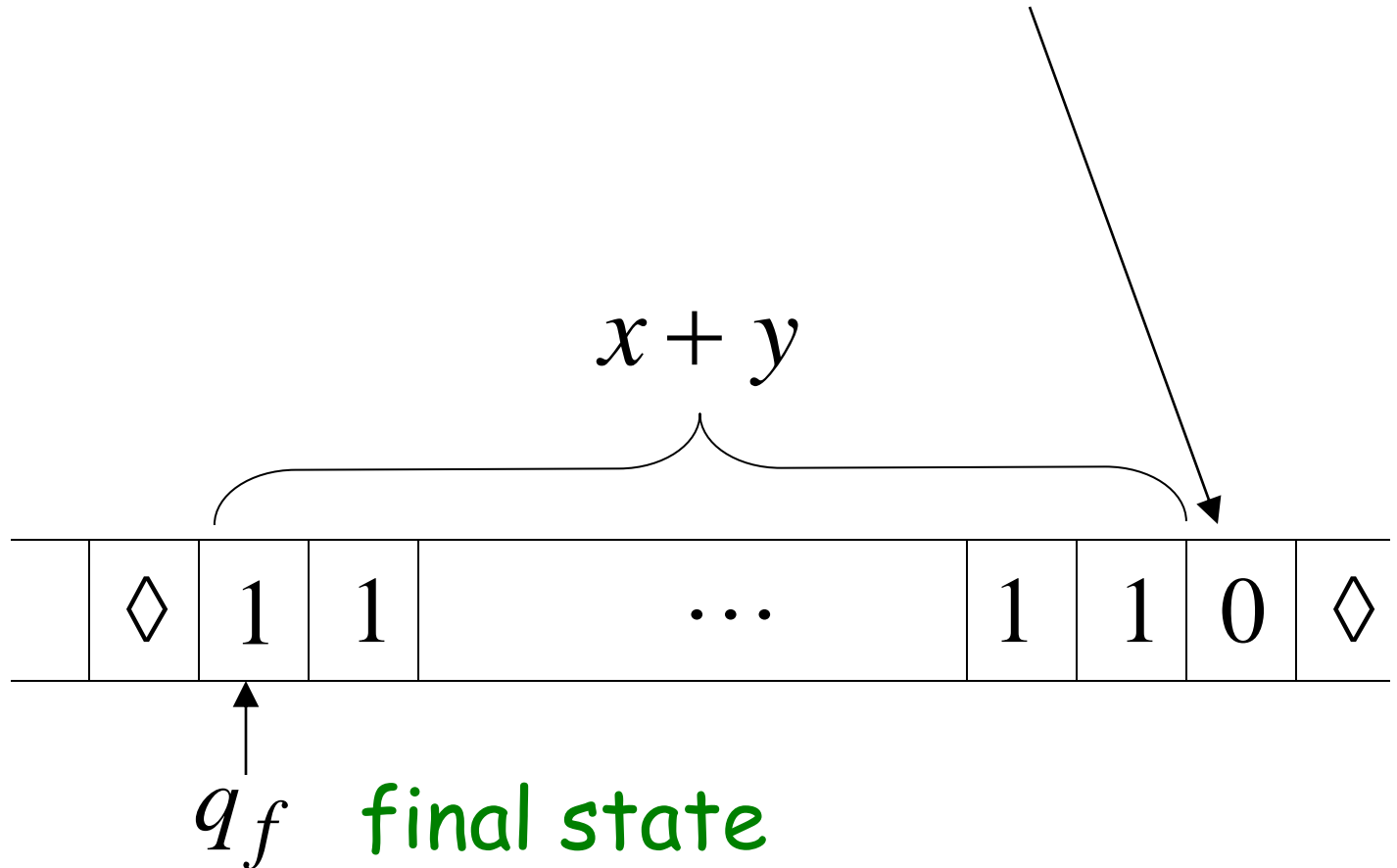


Finish



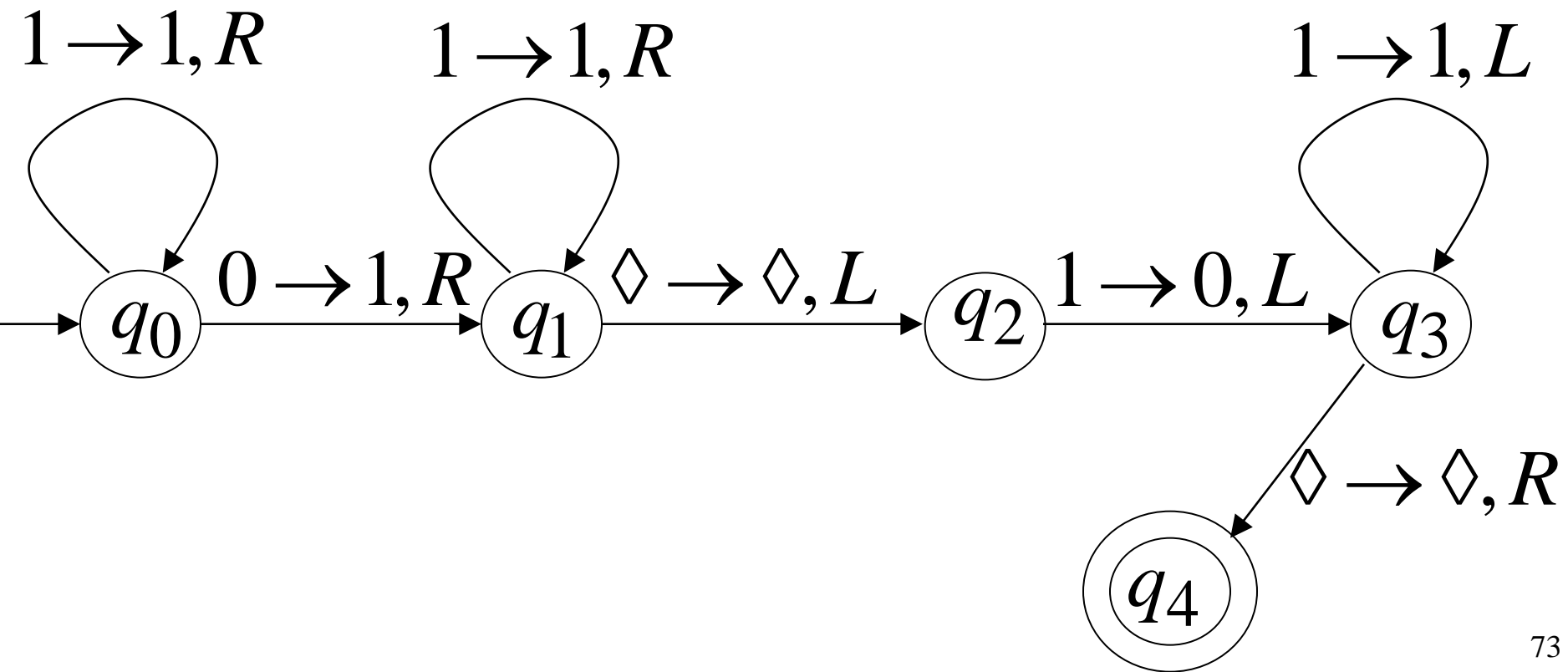
The 0 helps when we use the result for other operations

Finish





# Turing machine for function $f(x, y) = x + y$

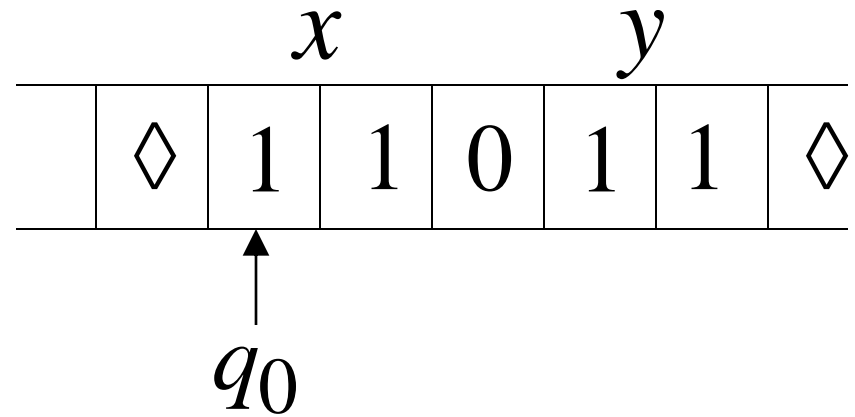


# Execution Example:

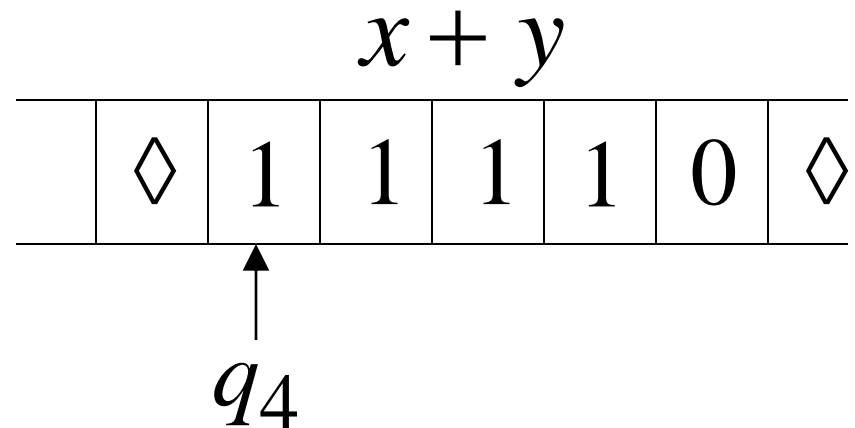
$$x = 11 \quad (2)$$

$$y = 11 \quad (2)$$

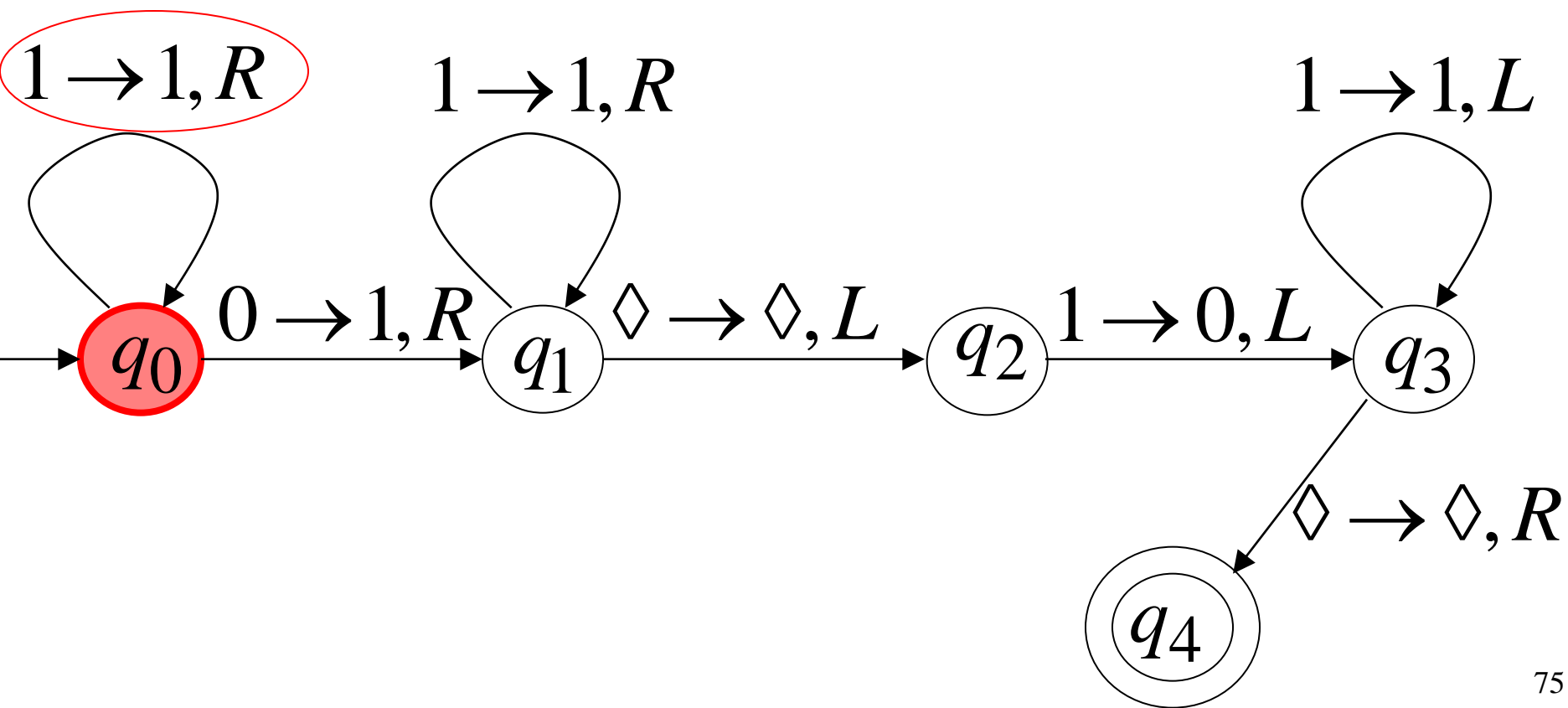
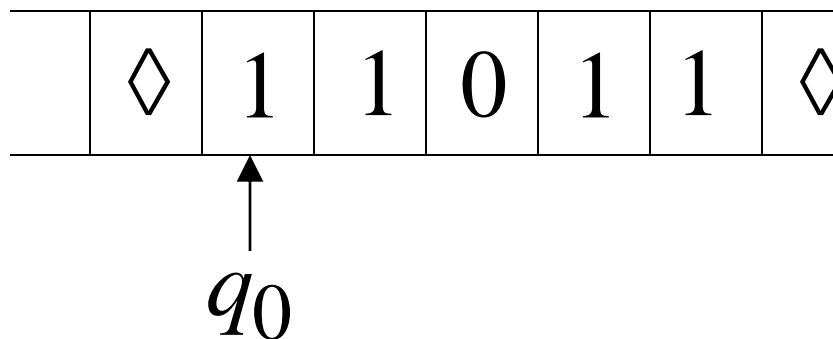
Time 0



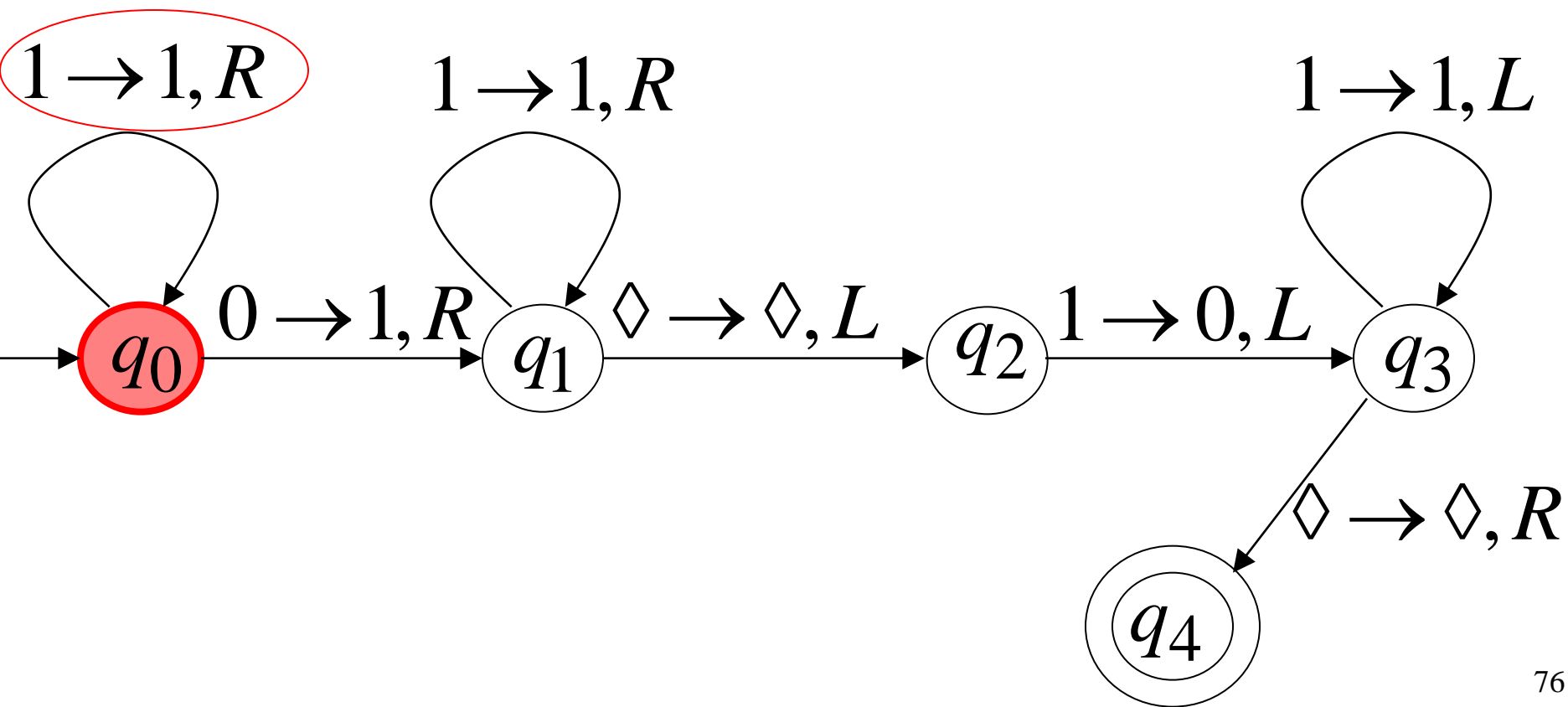
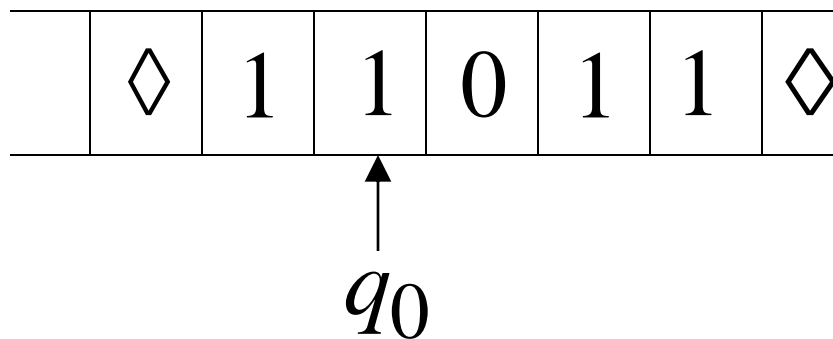
Final Result



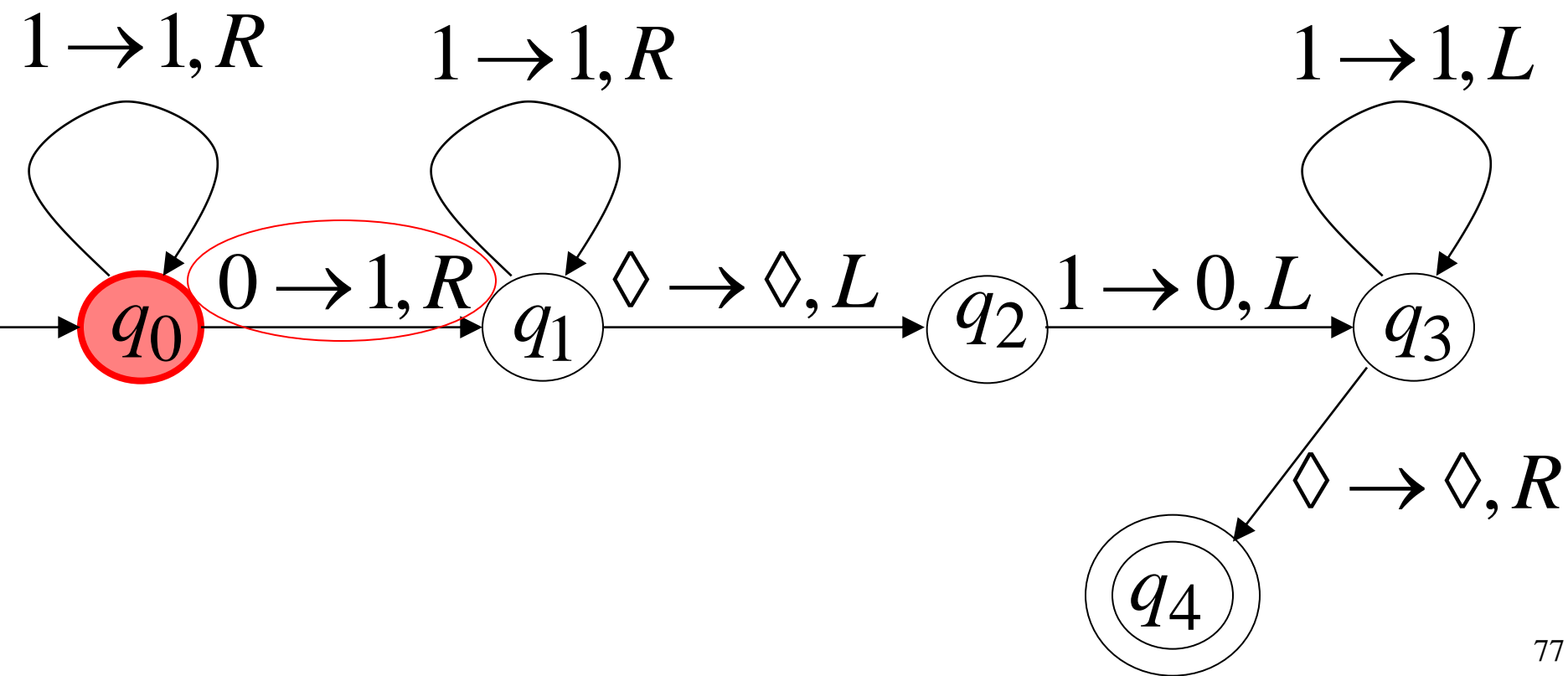
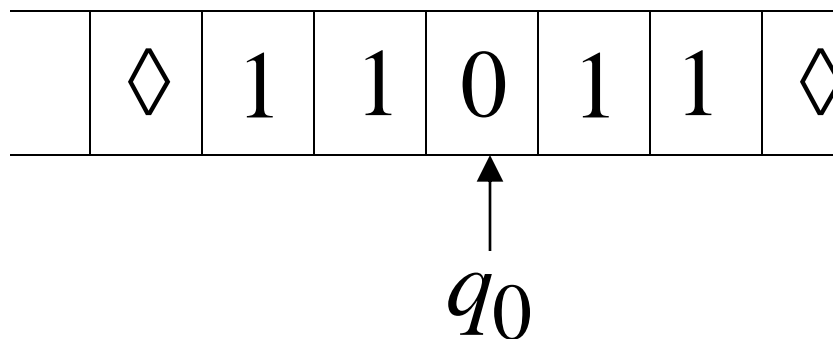
Time 0



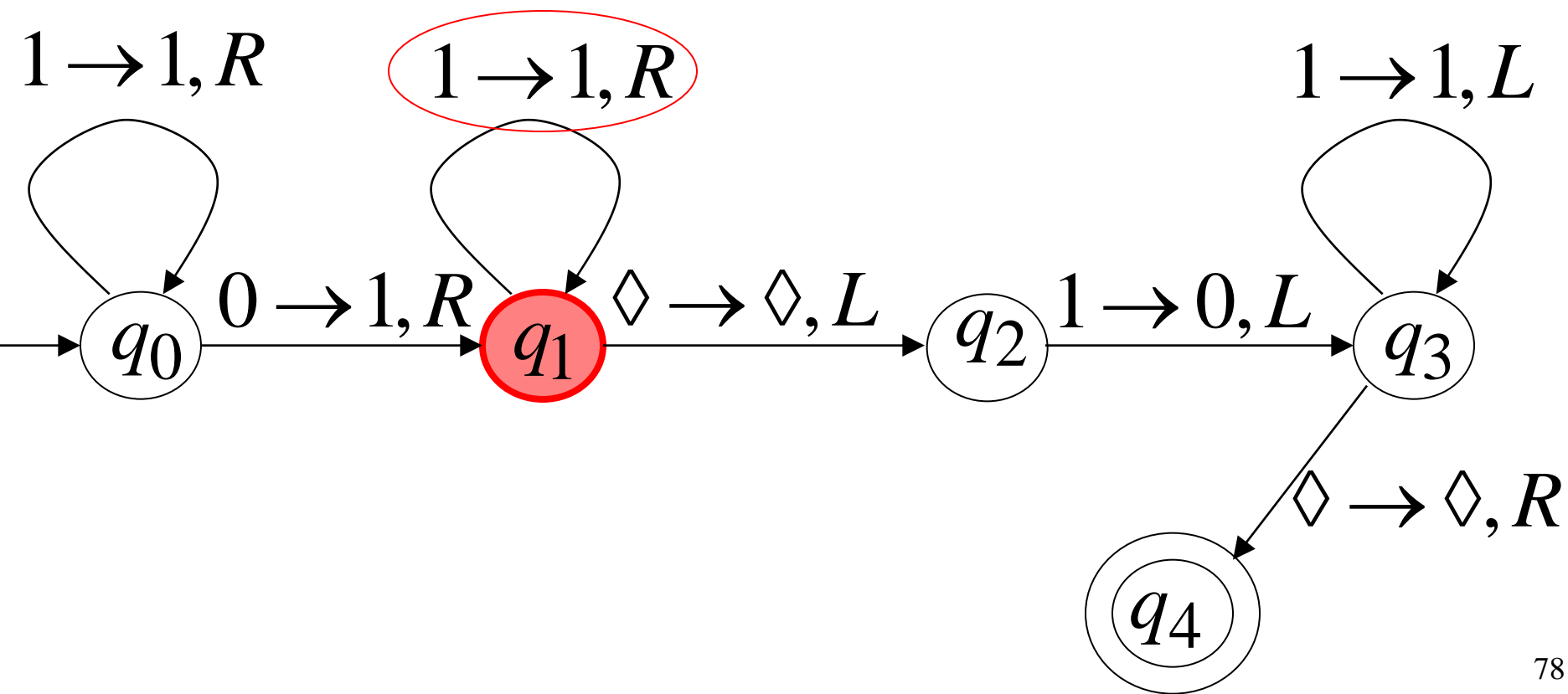
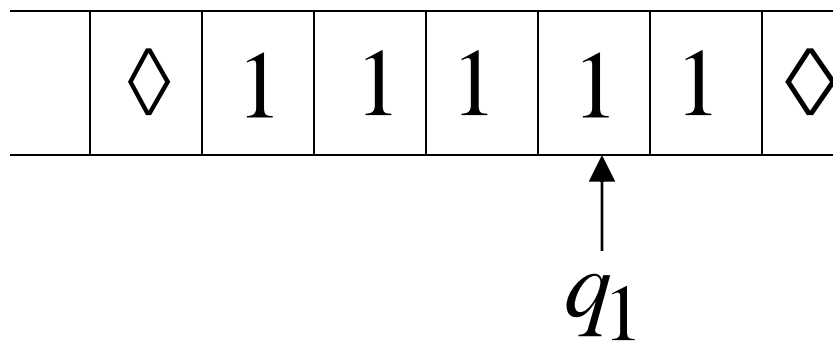
Time 1



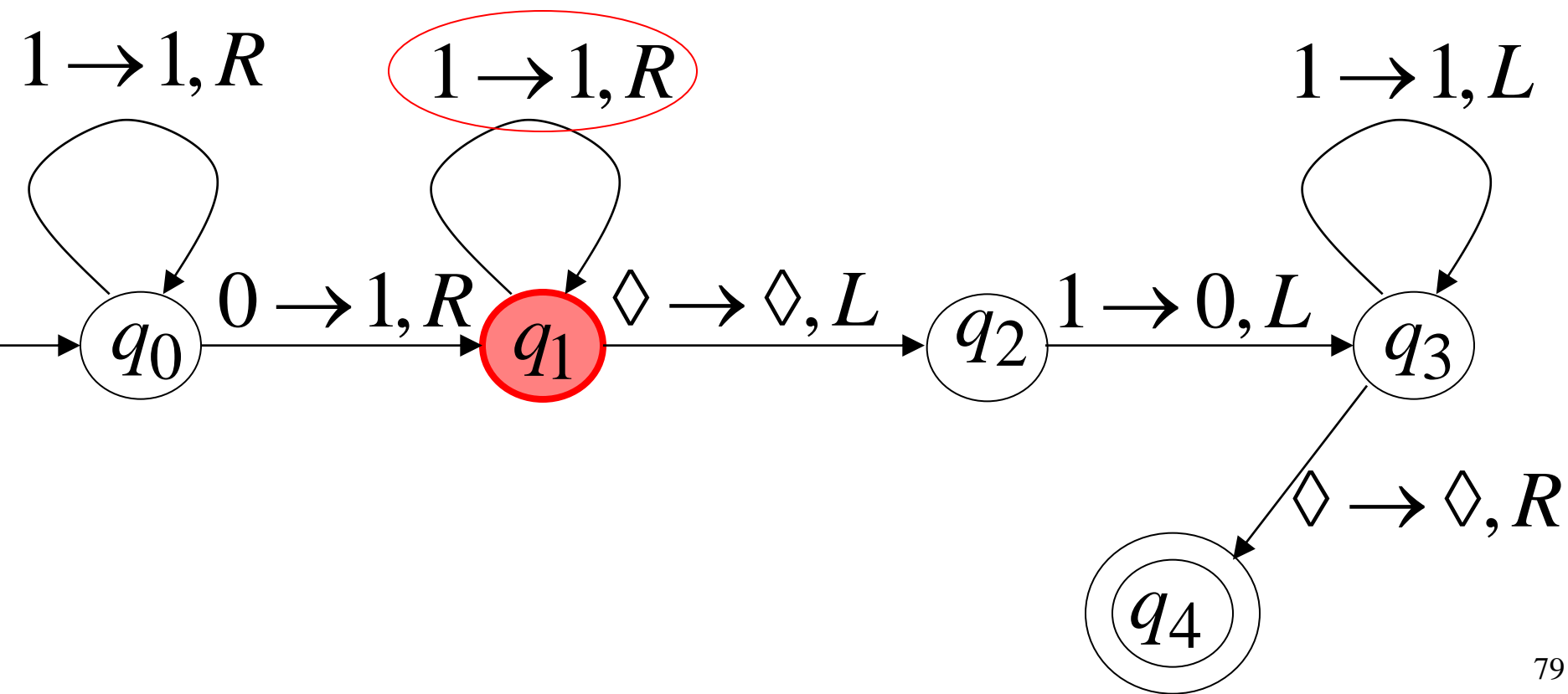
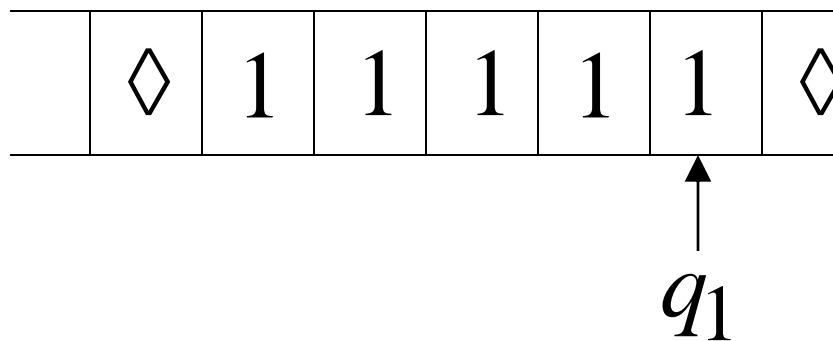
Time 2



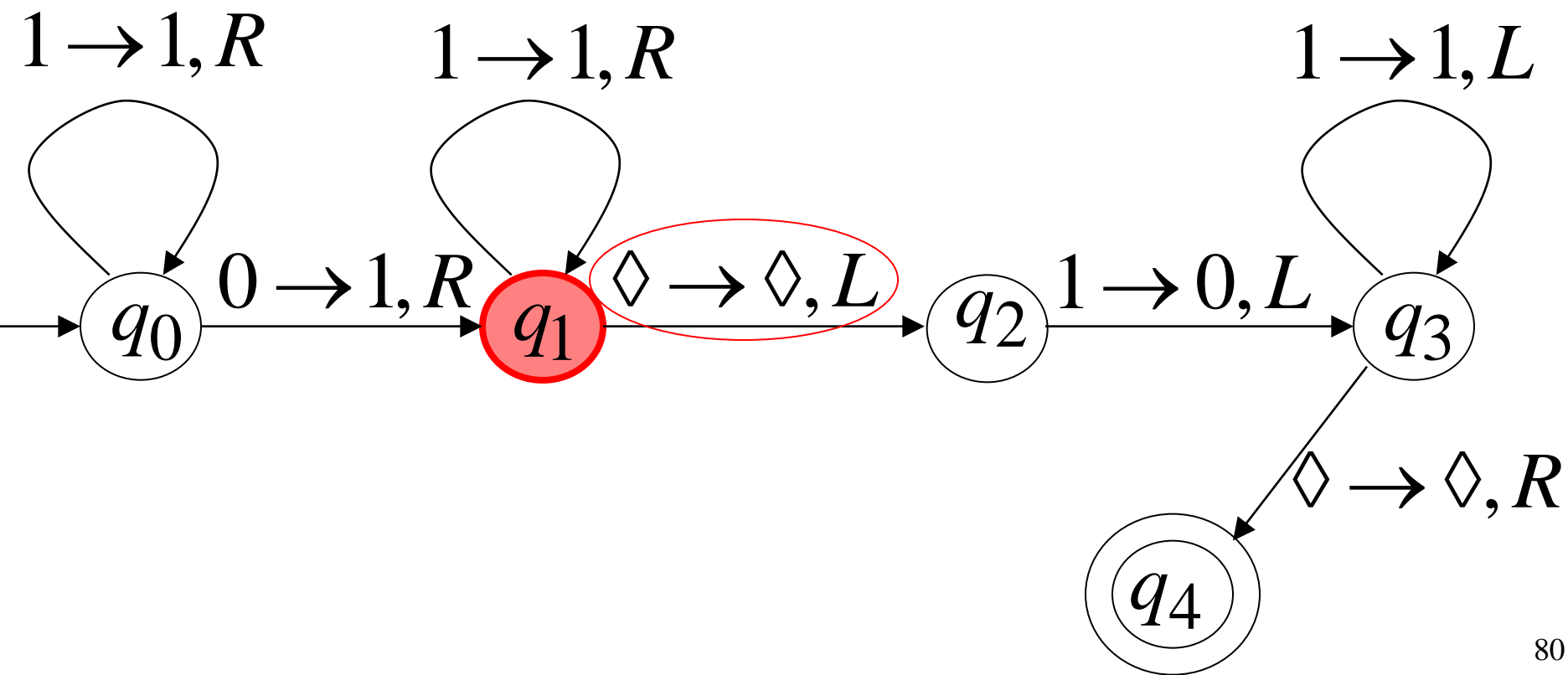
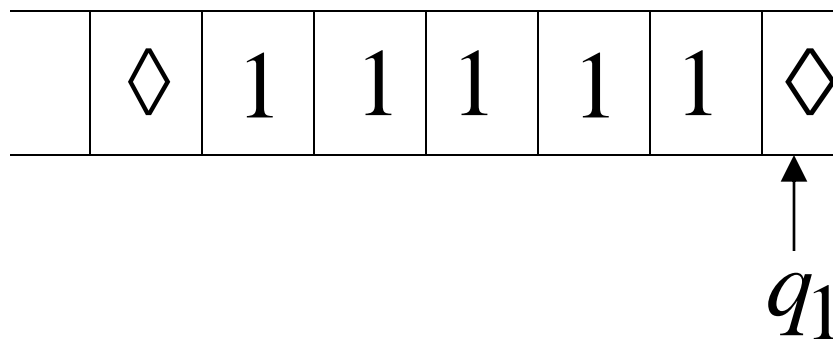
Time 3



Time 4

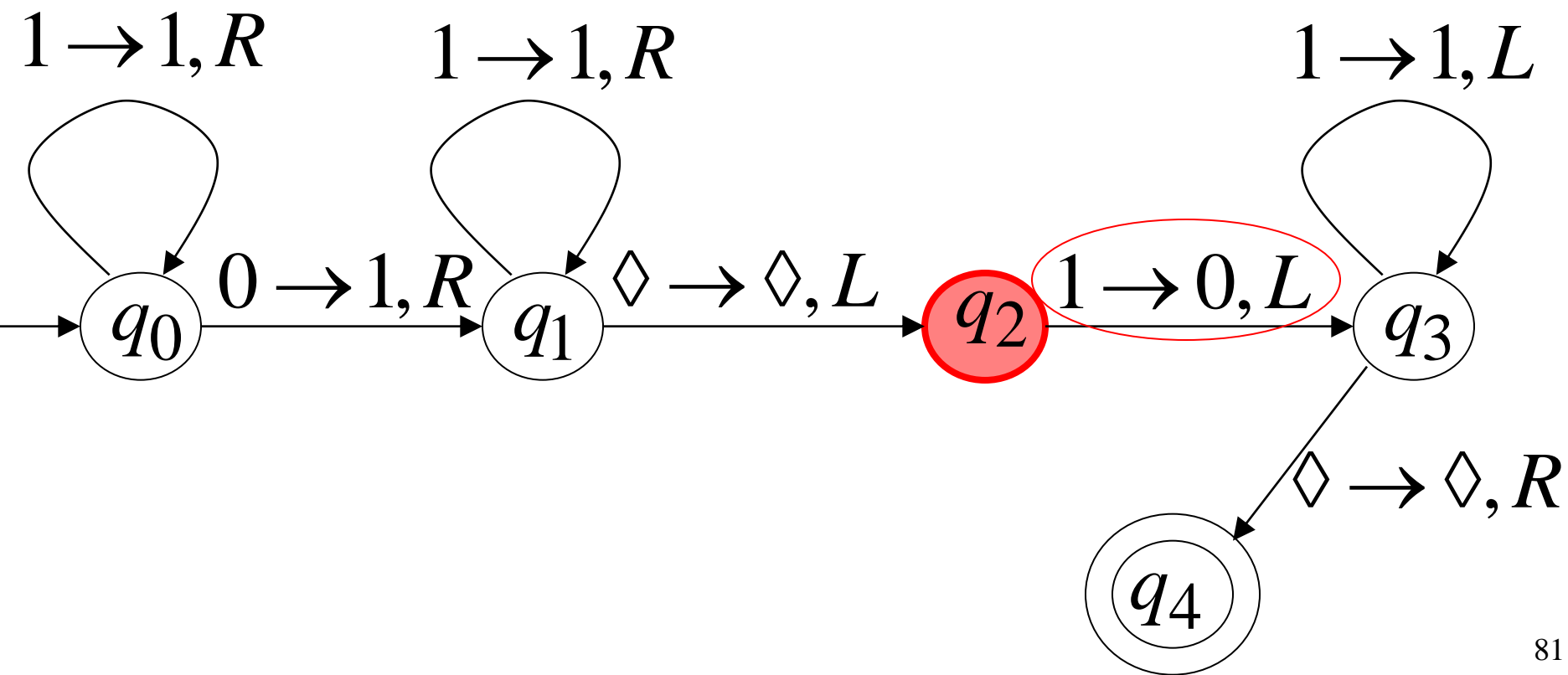
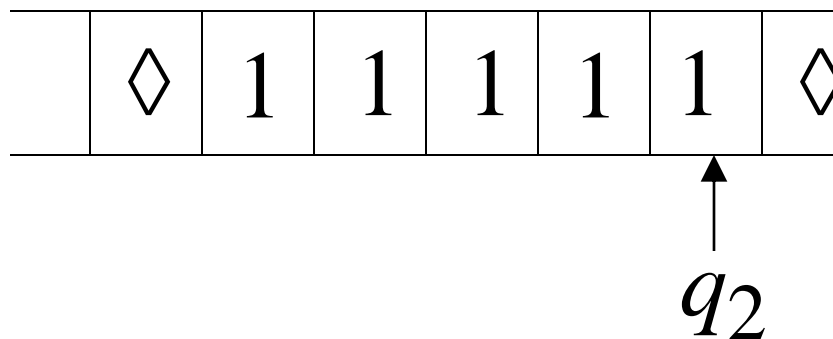


Time 5

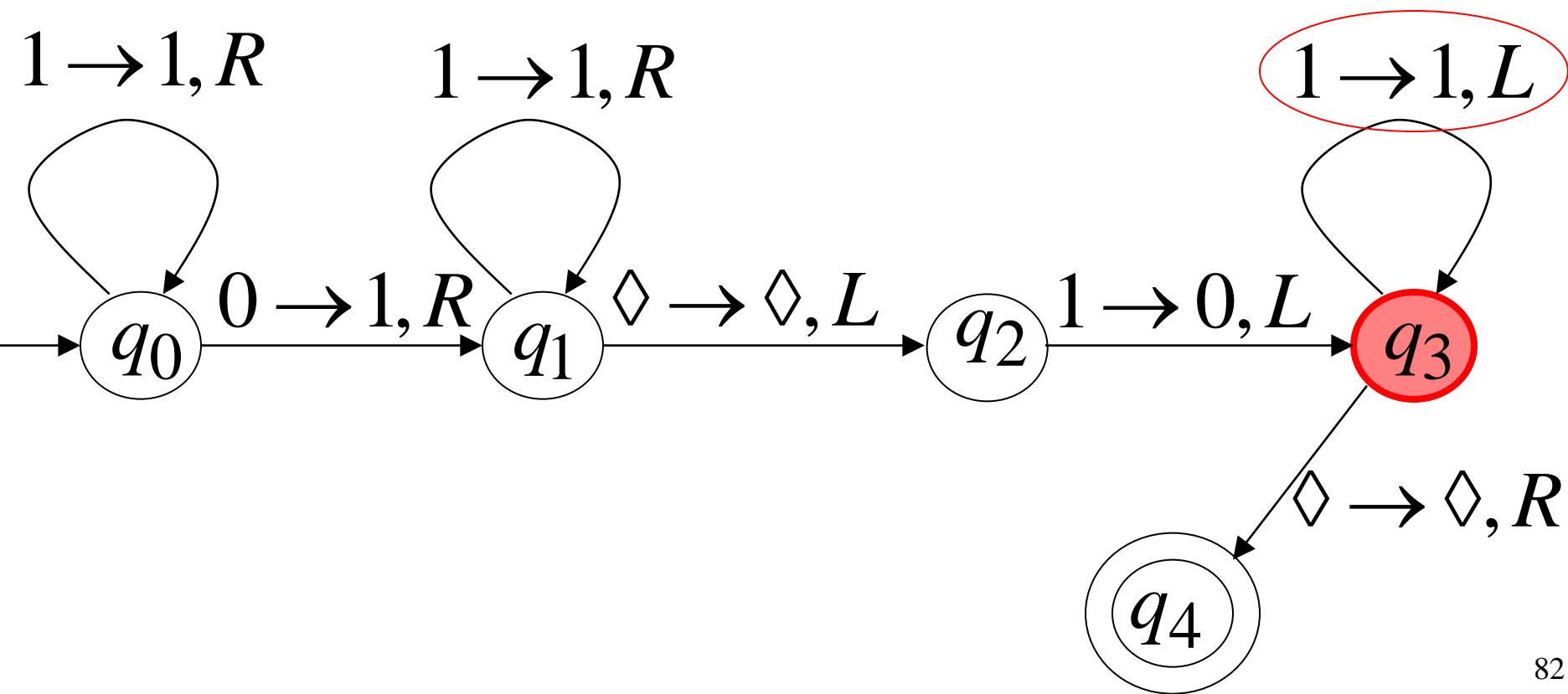
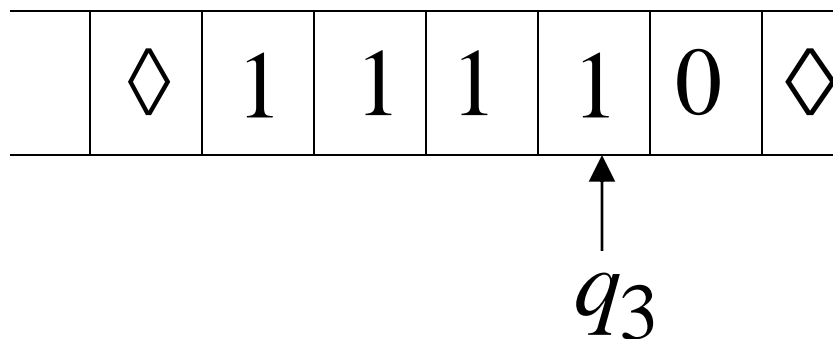




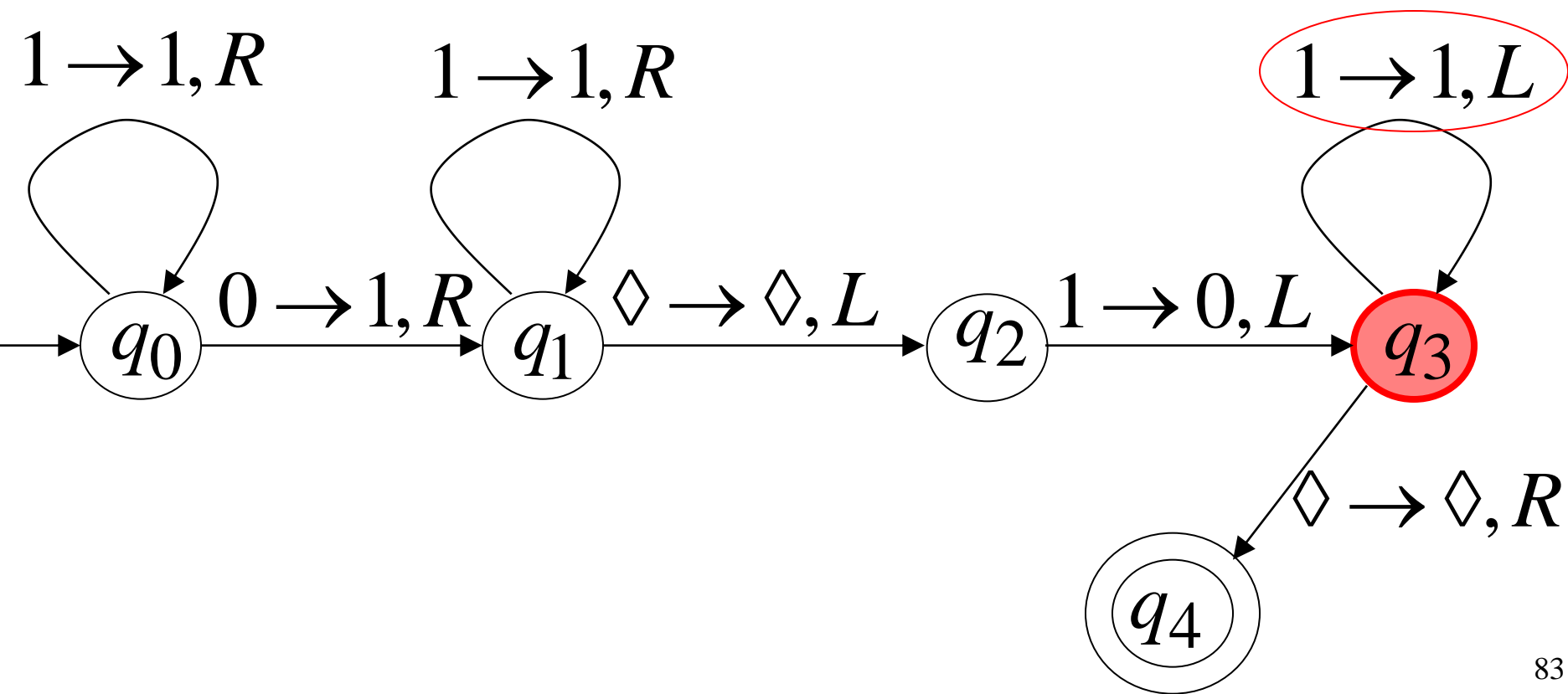
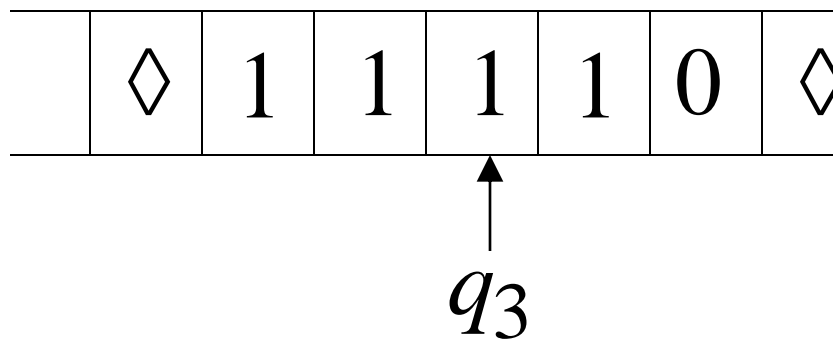
Time 6



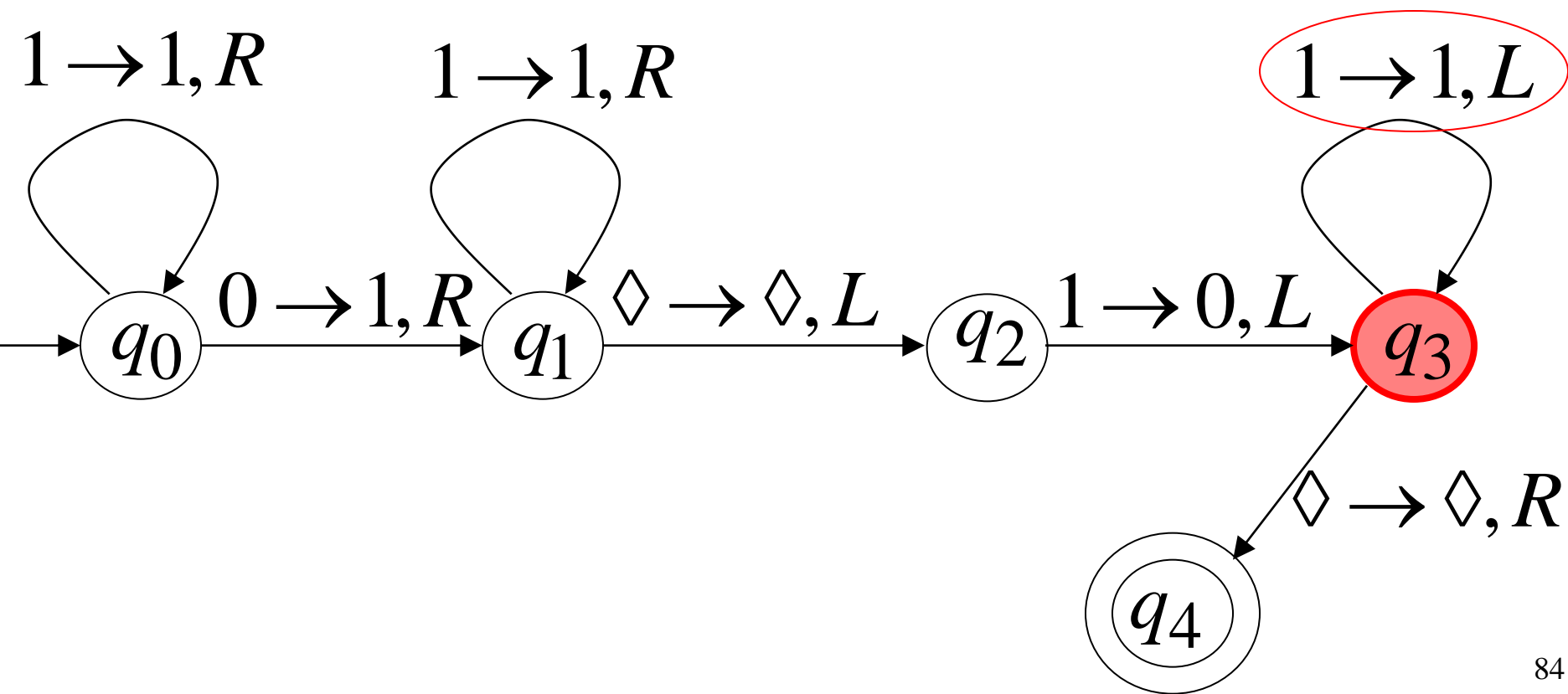
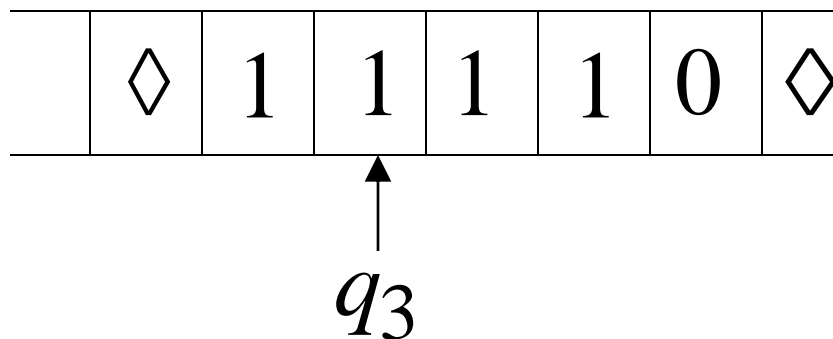
Time 7



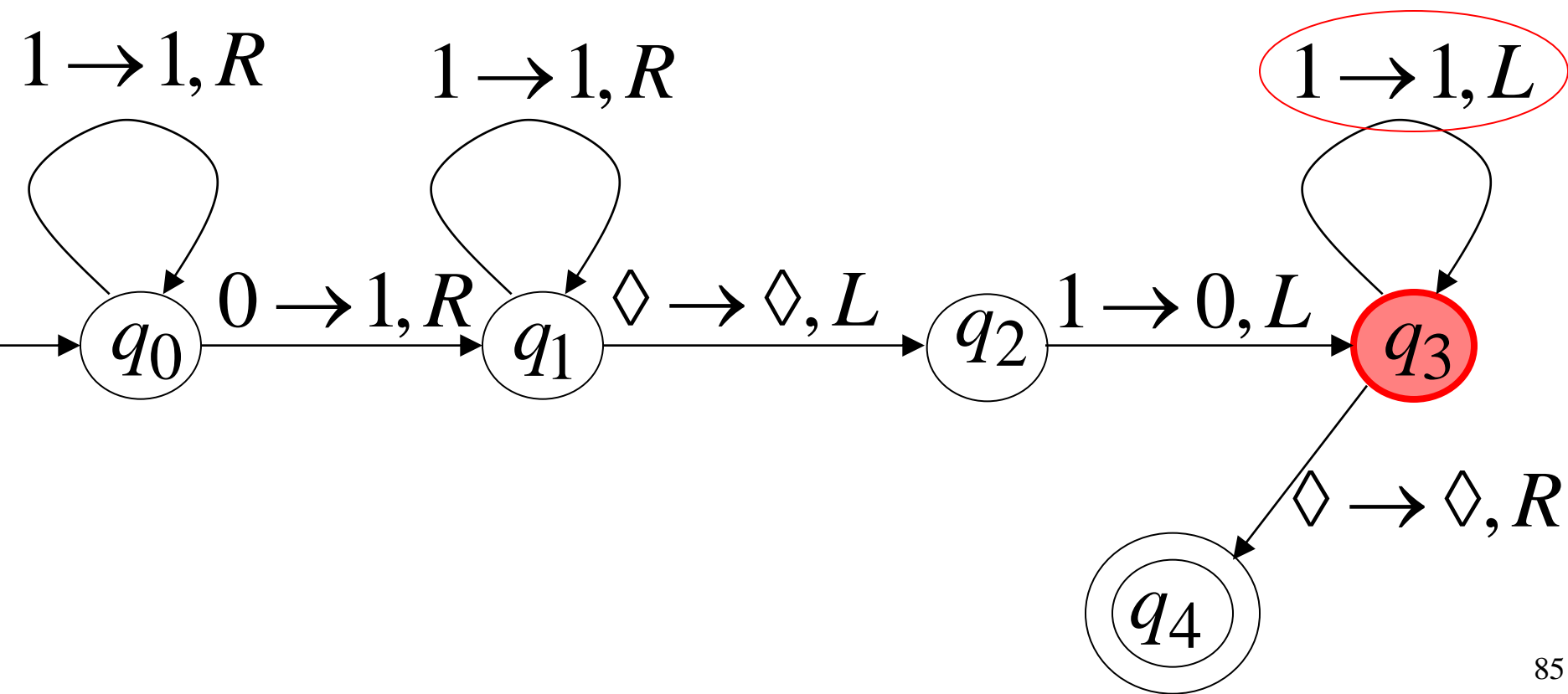
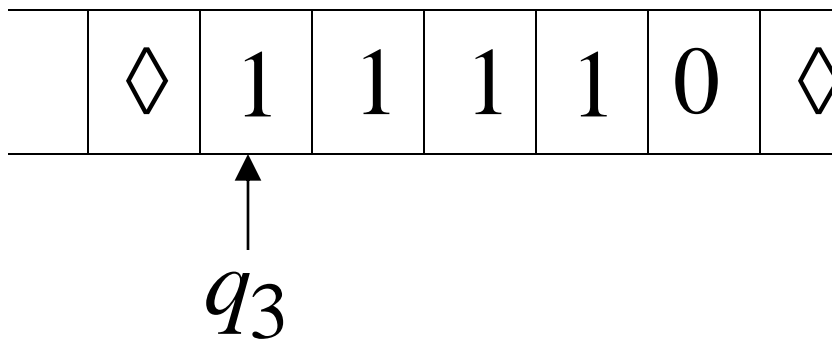
Time 8



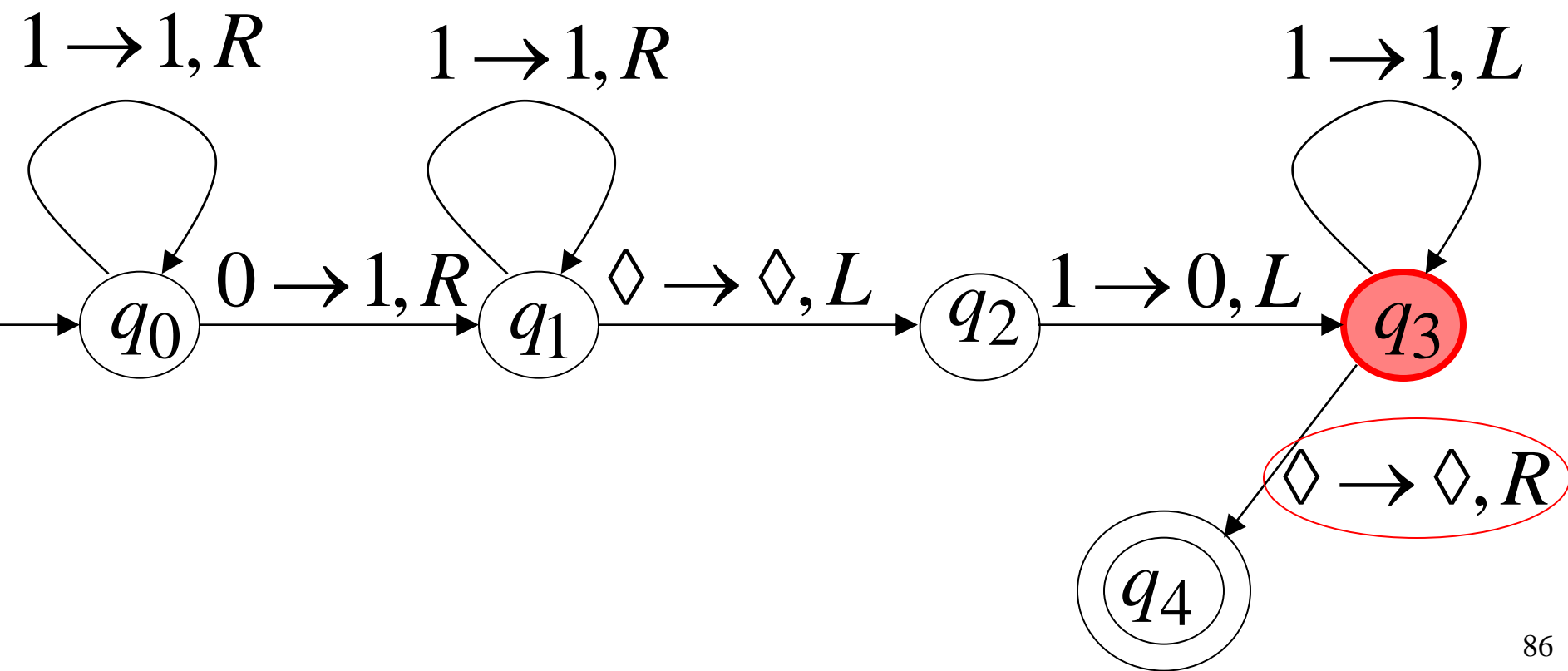
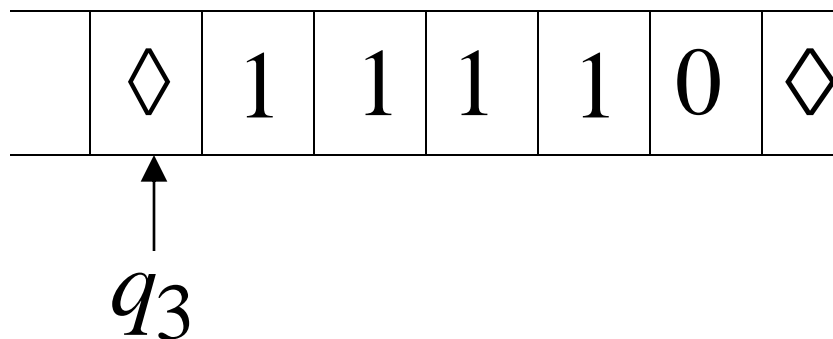
Time 9



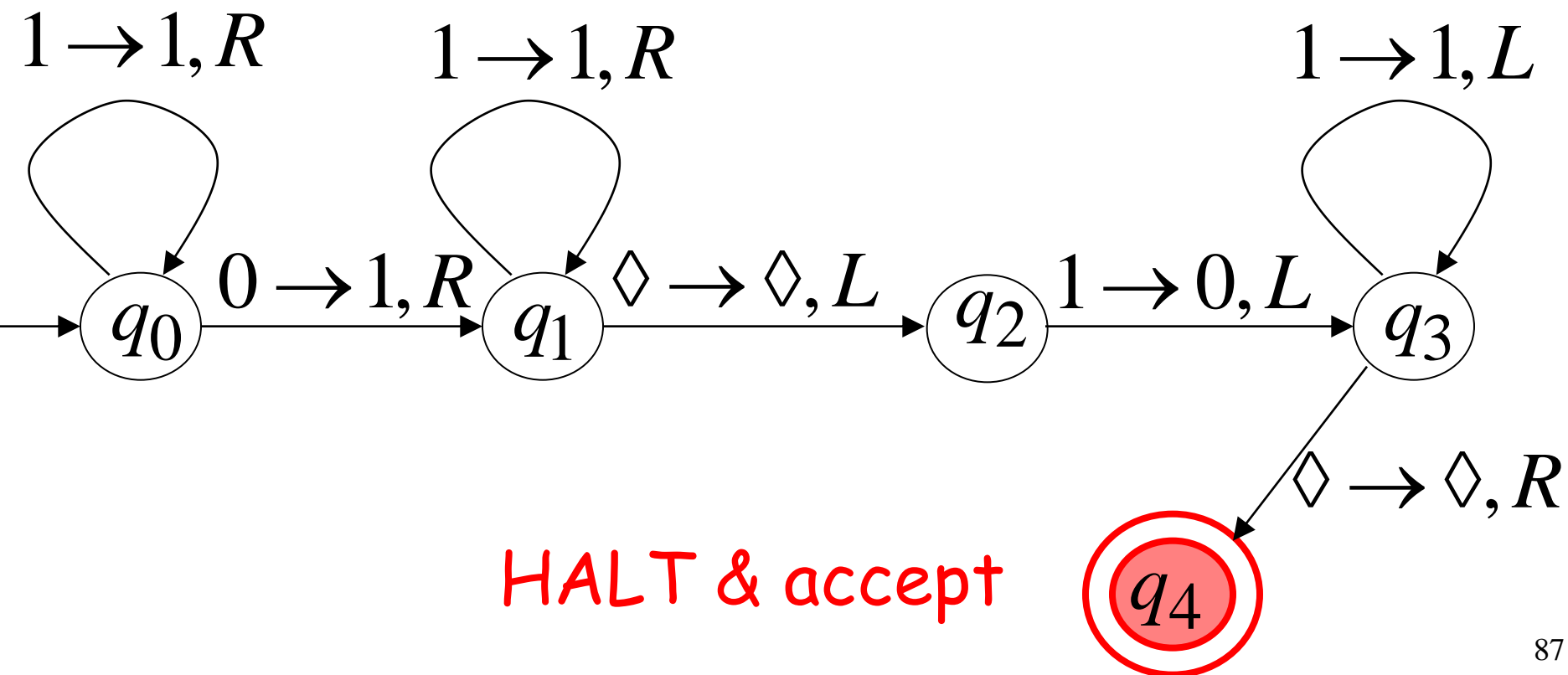
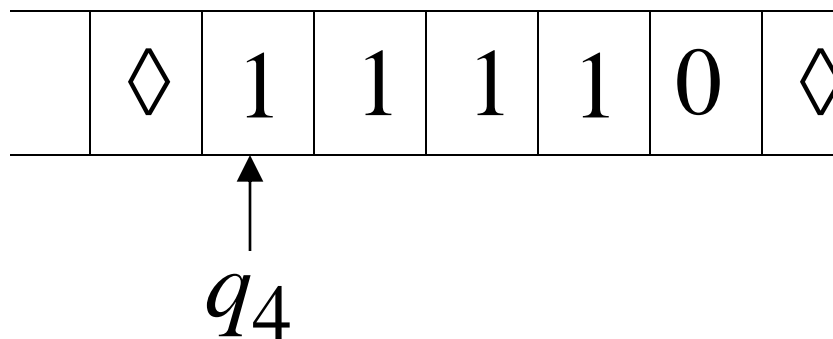
Time 10



Time 11



Time 12



# Another Example

The function  $f(x) = 2x$  is computable

$x$  is integer

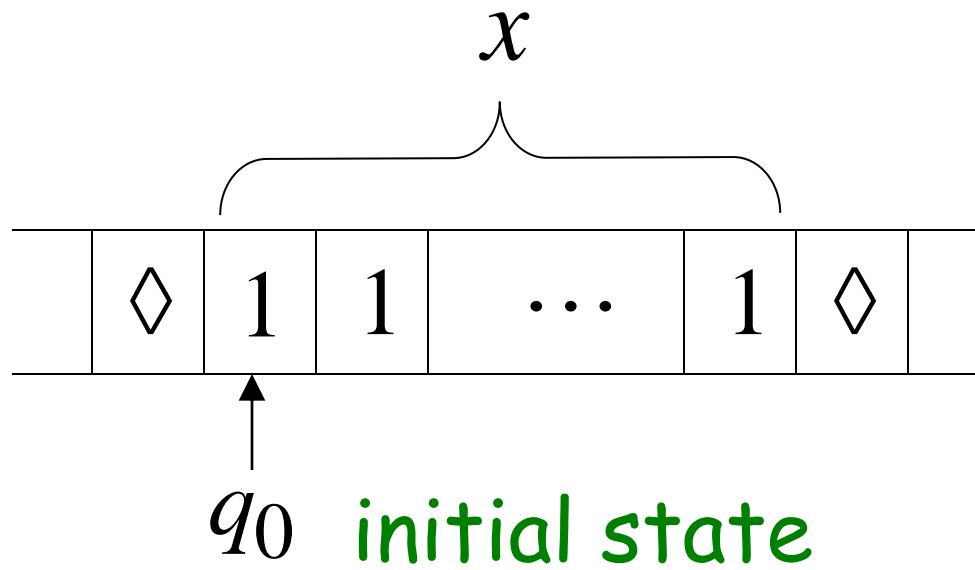
Turing Machine:

Input string:  $x$  unary

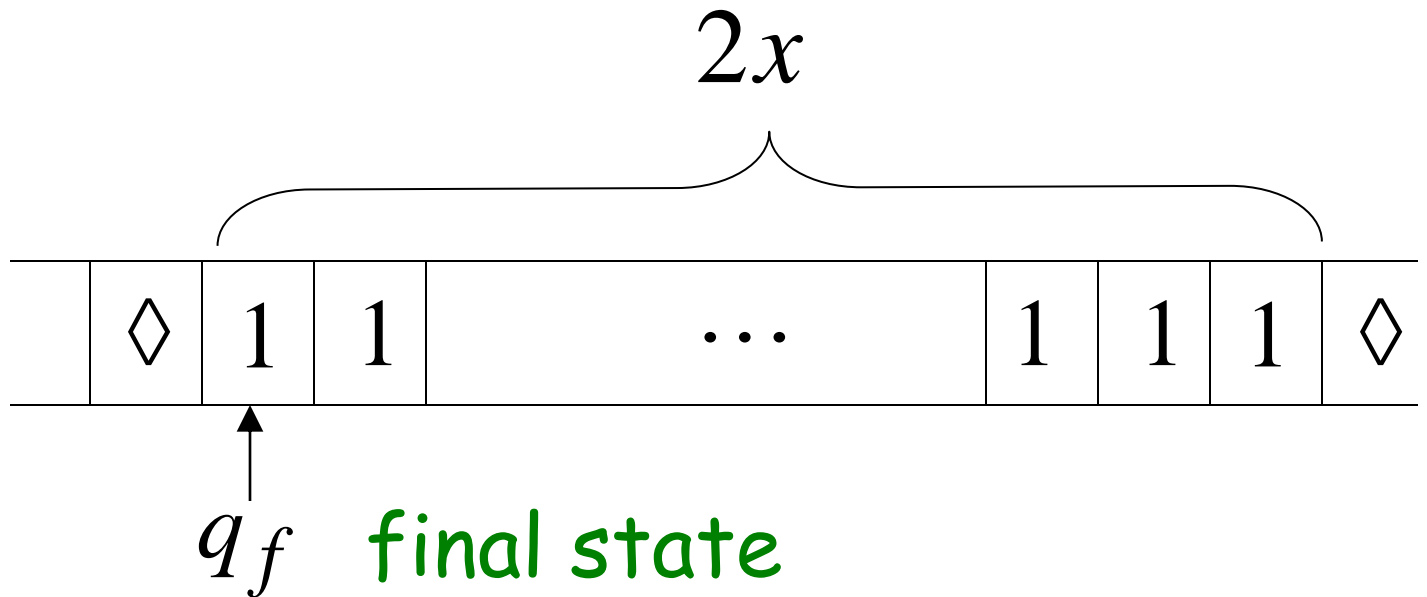
Output string:  $xx$  unary



Start



Finish

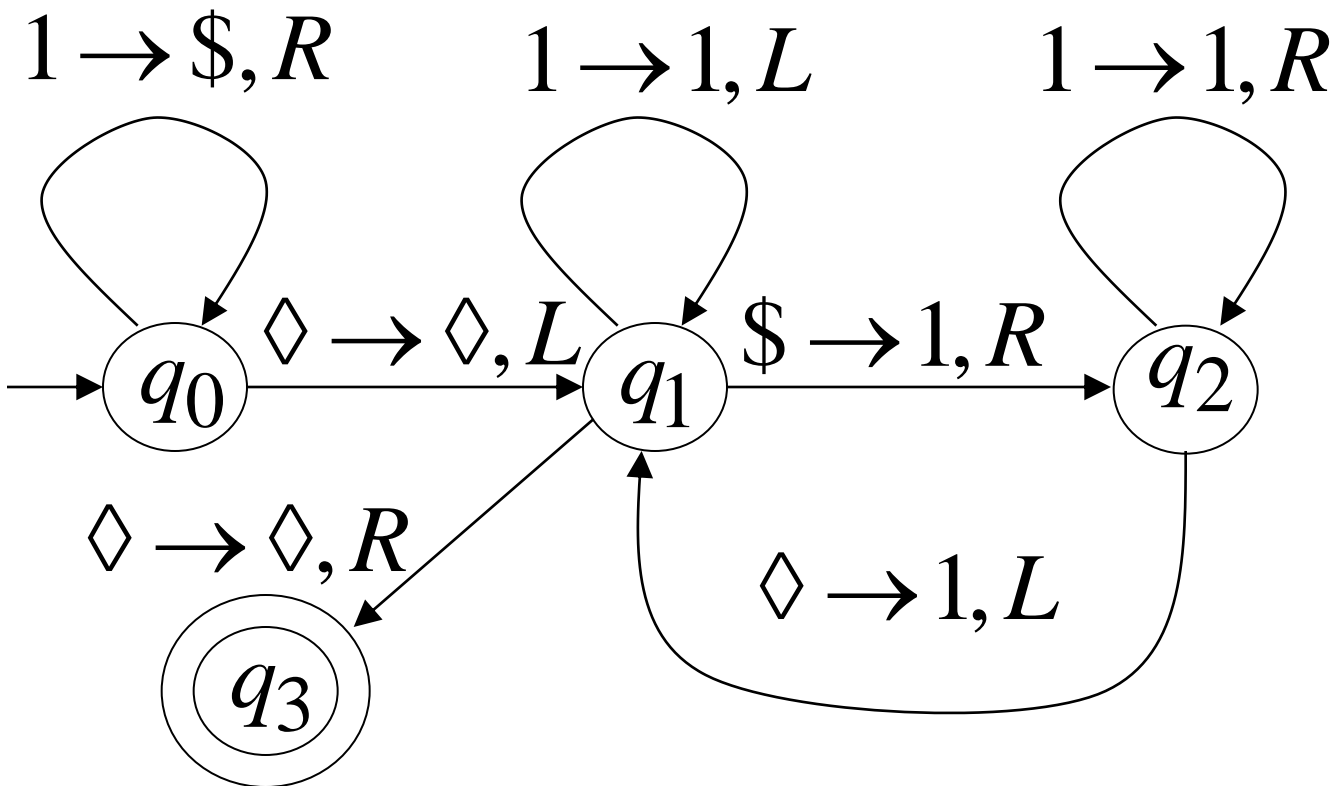


# Turing Machine Pseudocode for $f(x) = 2x$

- Replace every 1 with \$
- Repeat:
  - Find rightmost \$, replace it with 1
  - Go to right end, insert 1

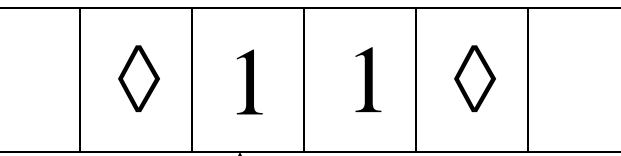
Until no more \$ remain

# Turing Machine for $f(x) = 2x$



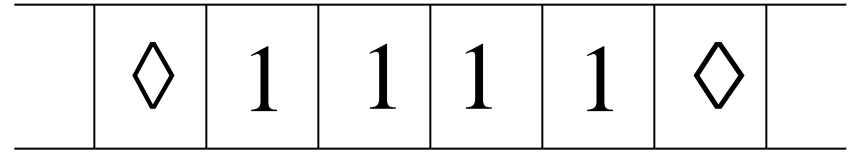
# Example

Start

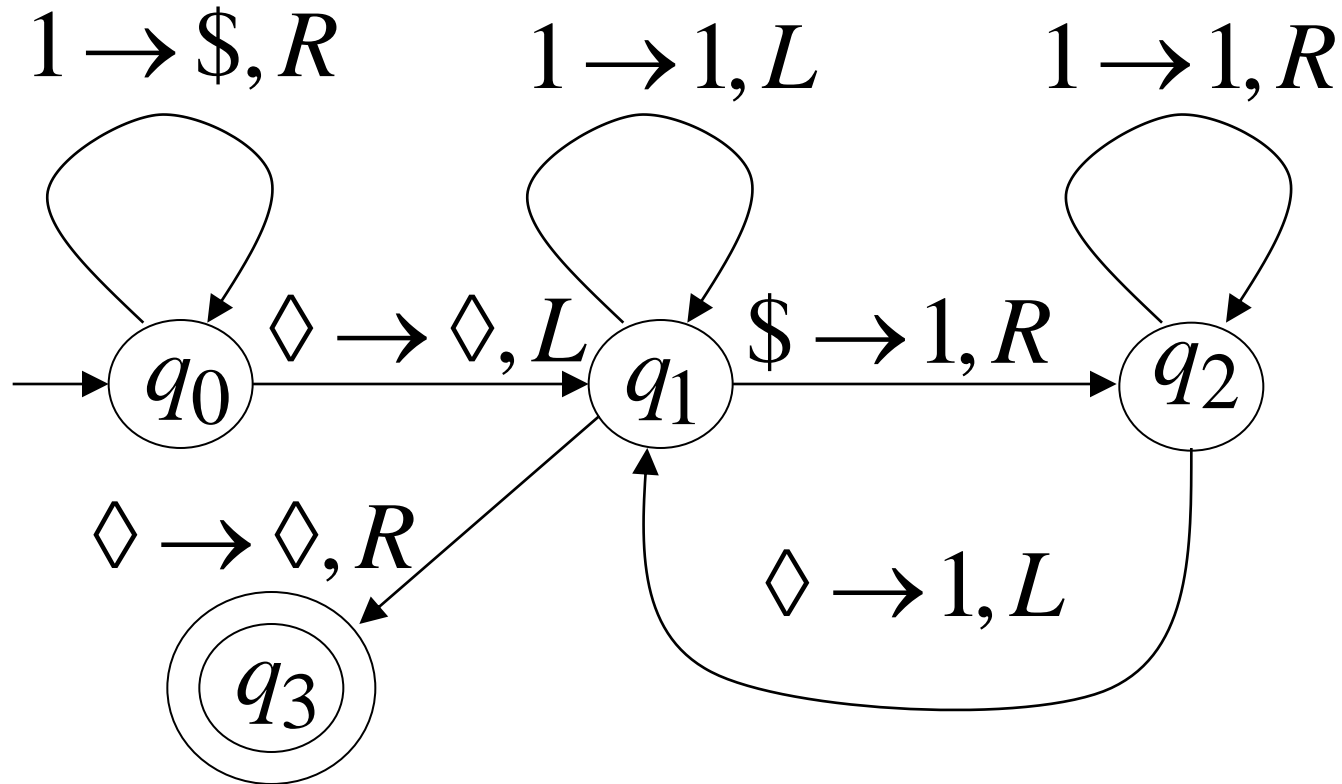


$q_0$

Finish



$q_3$



# Another Example

The function  
is computable

$$f(x, y) = \begin{cases} 1 & \text{if } x > y \\ 0 & \text{if } x \leq y \end{cases}$$

# Turing Machine for

$$f(x, y) = \begin{cases} 1 & \text{if } x > y \\ 0 & \text{if } x \leq y \end{cases}$$

Input:  $x0y$

Output: 1 or 0

# Turing Machine Pseudocode:

- Repeat

Match a 1 from  $x$  with a 1 from  $y$

Until all of  $x$  or  $y$  is matched

- If a 1 from  $x$  is not matched

erase tape, write 1      ( $x > y$ )

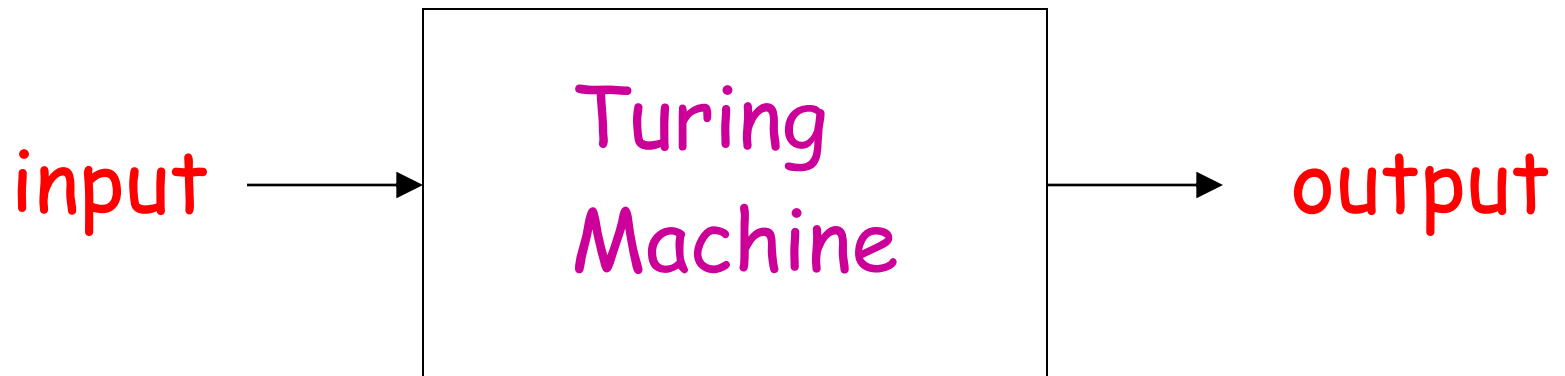
else

erase tape, write 0      ( $x \leq y$ )

# Combining Turing Machines



# Block Diagram



Example:

$$f(x, y) = \begin{cases} x + y & \text{if } x > y \\ 0 & \text{if } x \leq y \end{cases}$$

