2021

Theory of Computation

Kun-Ta Chuang
Department of Computer Science and Information Engineering
National Cheng Kung University

Outline

1. The Standard Turing Machine
2. Combining Turing Machines for Complicated Tasks
3. Turing's Thesis

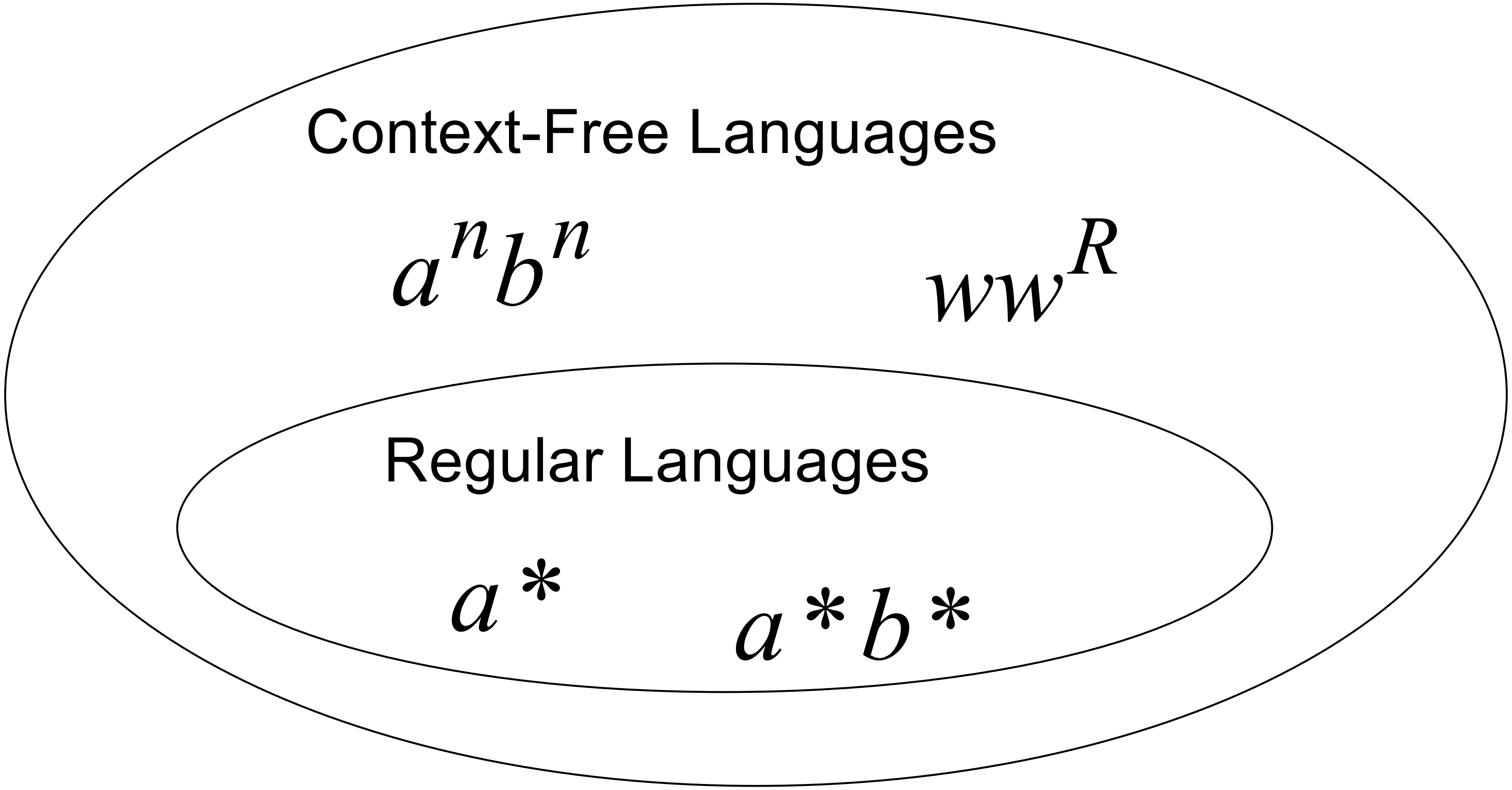
Turing Machines

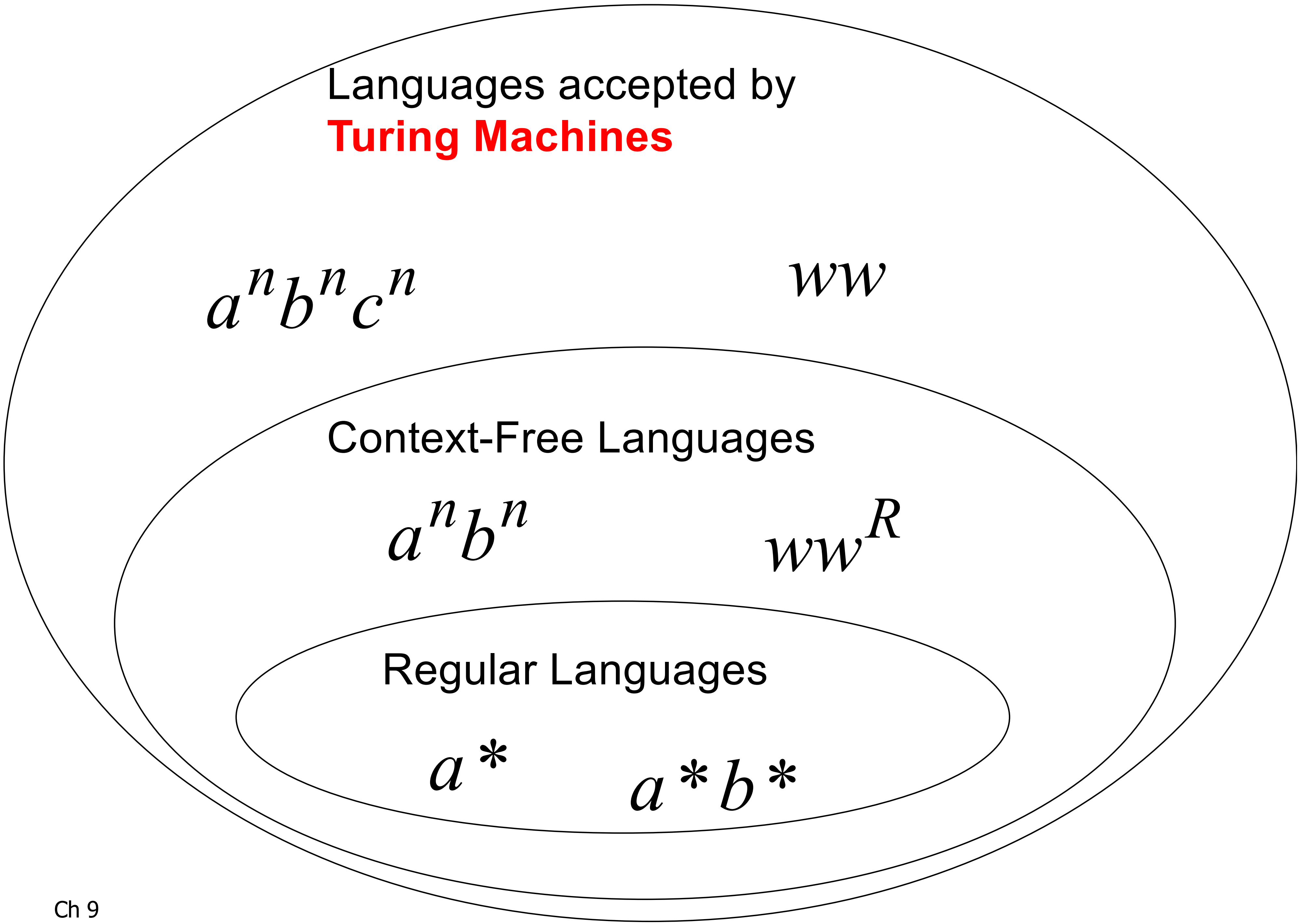
- Developed by Alan Turing in 1936
- More than just recognizing languages
- Foundation for modern theory of computation

- Alan Turing
  - 1912 – 1954
  - b. London, England.
  - PhD – Princeton (1938)
  - Research
    - Cambridge and Manchester U.
    - National Physical Lab, UK

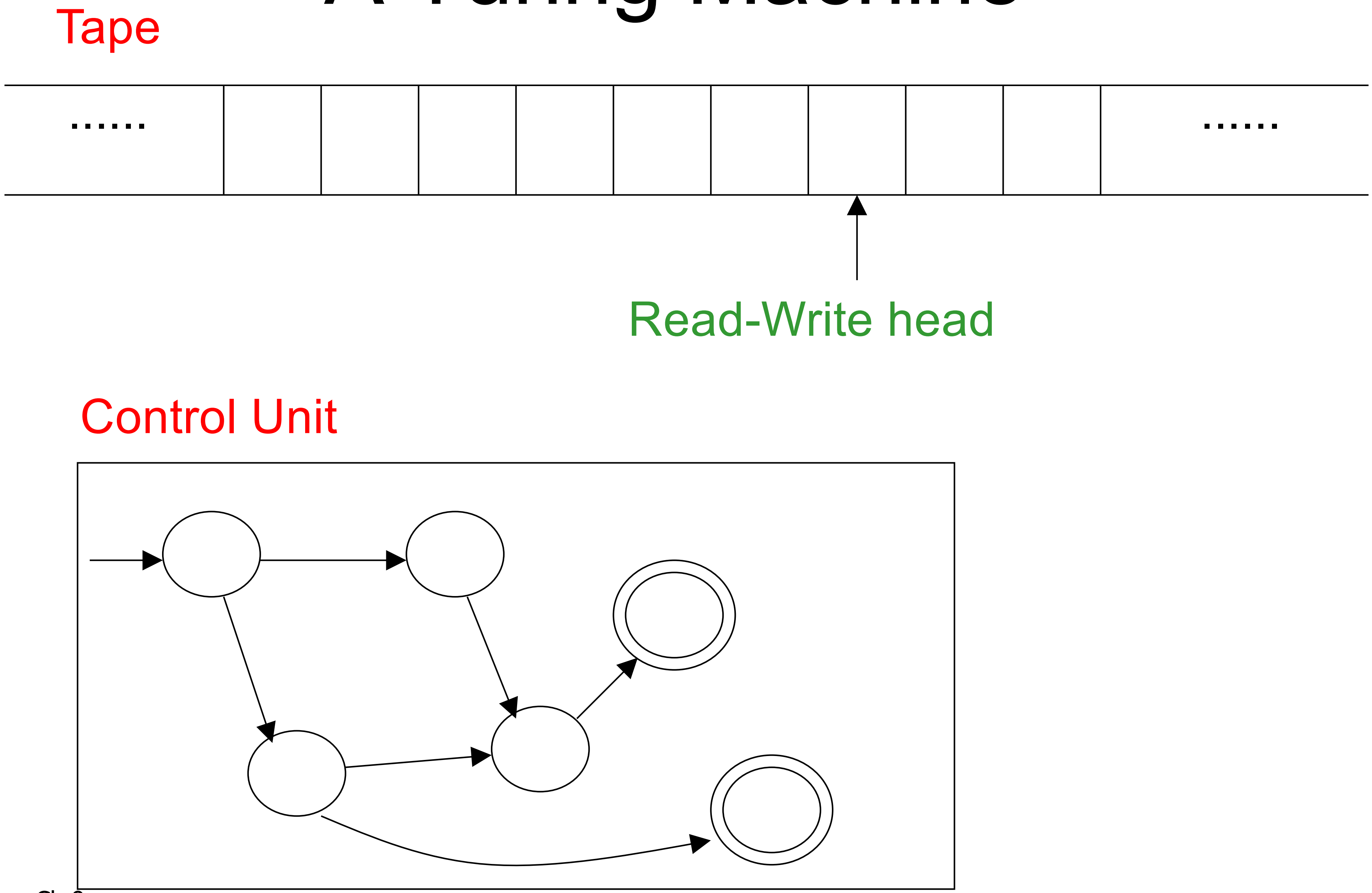
The Language Hierarchy

$a^n b^n c^n$ **?** ww **?**



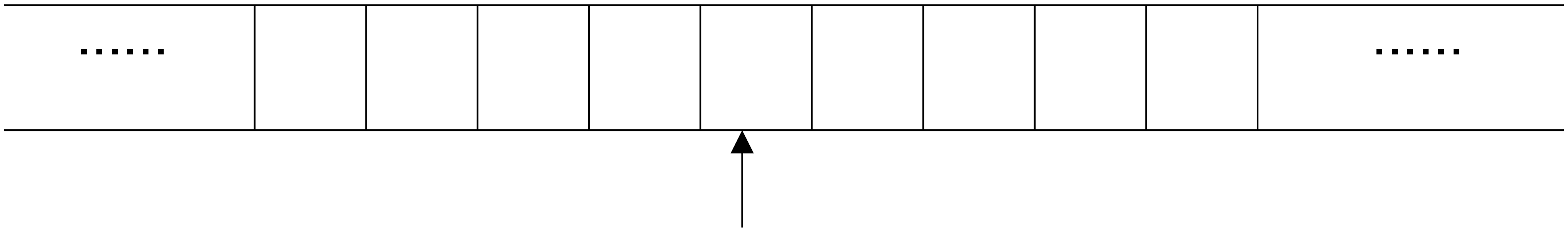

Languages accepted by
Turing Machines
$a^n b^n c^n$
ww
Context-Free Languages
$a^n b^n$
ww^R
Regular Languages
a^*
$a^* b^*$

A Turing Machine



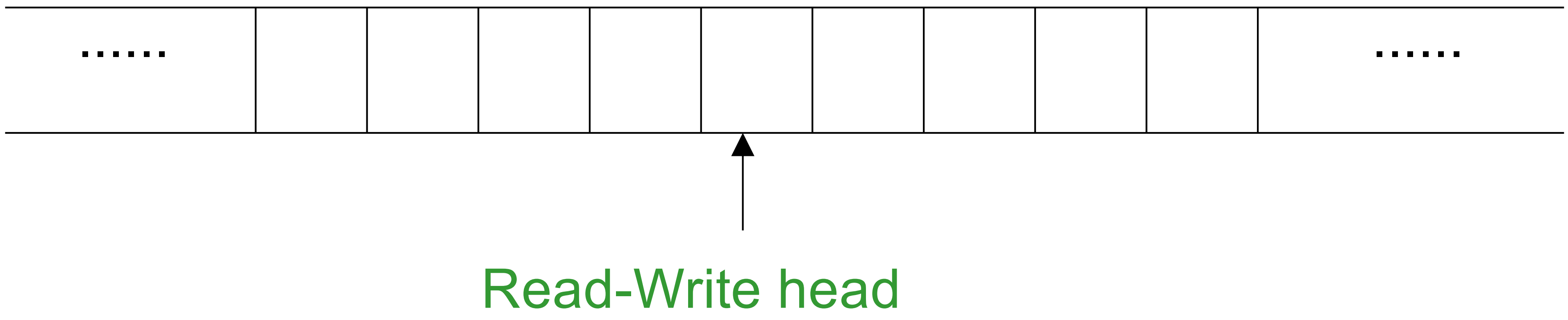
The Tape

No boundaries -- infinite length



Read-Write head

The head moves Left or Right

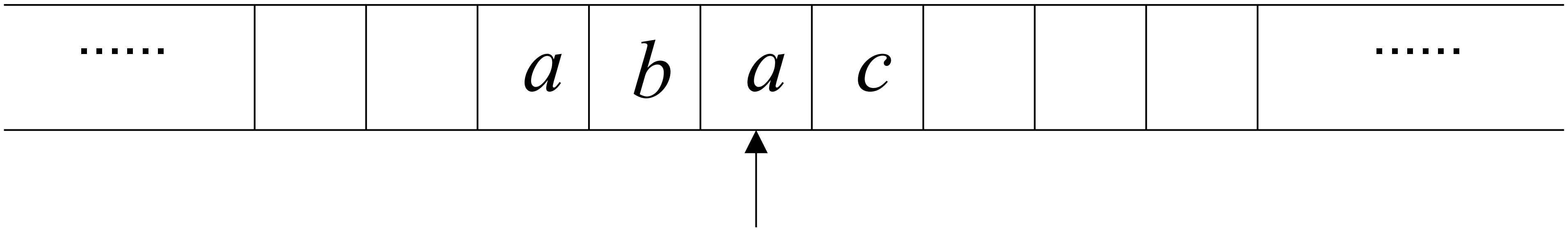


The head at each time step:

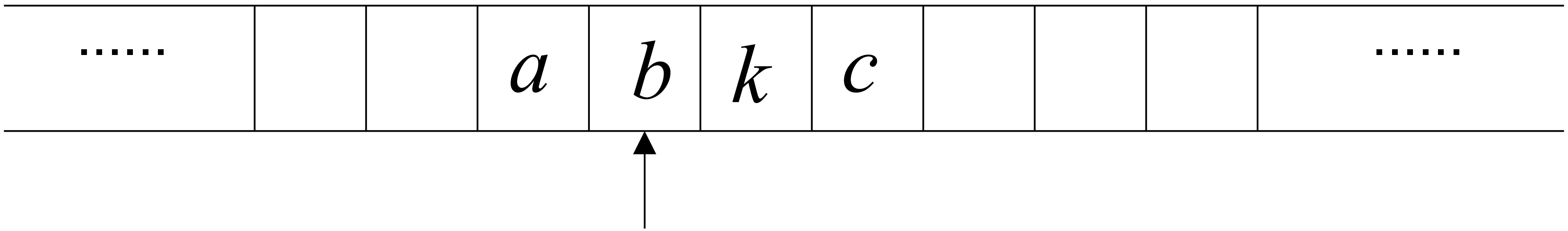
1. Reads a symbol
2. Writes a symbol
3. Moves Left or Right

Example:

Time 0



Time 1

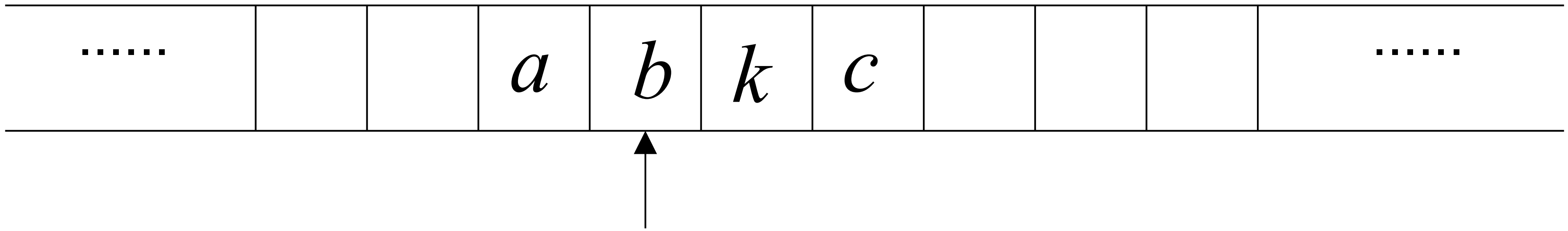


1. Reads a

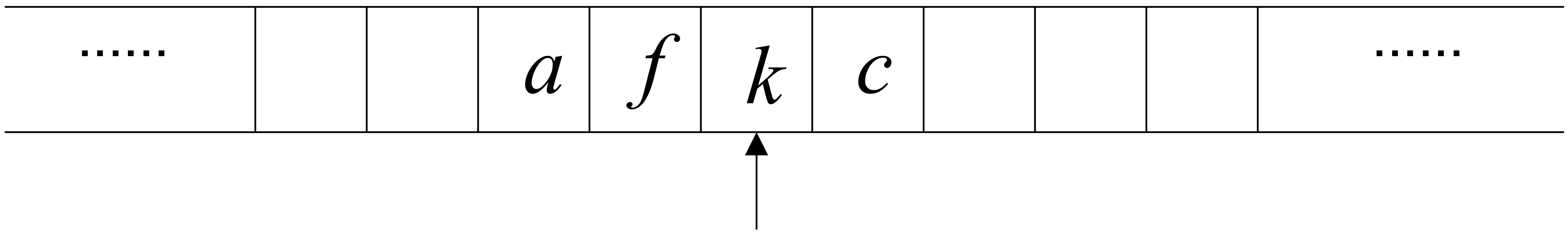
2. Writes k

3. Moves Left

Time 1

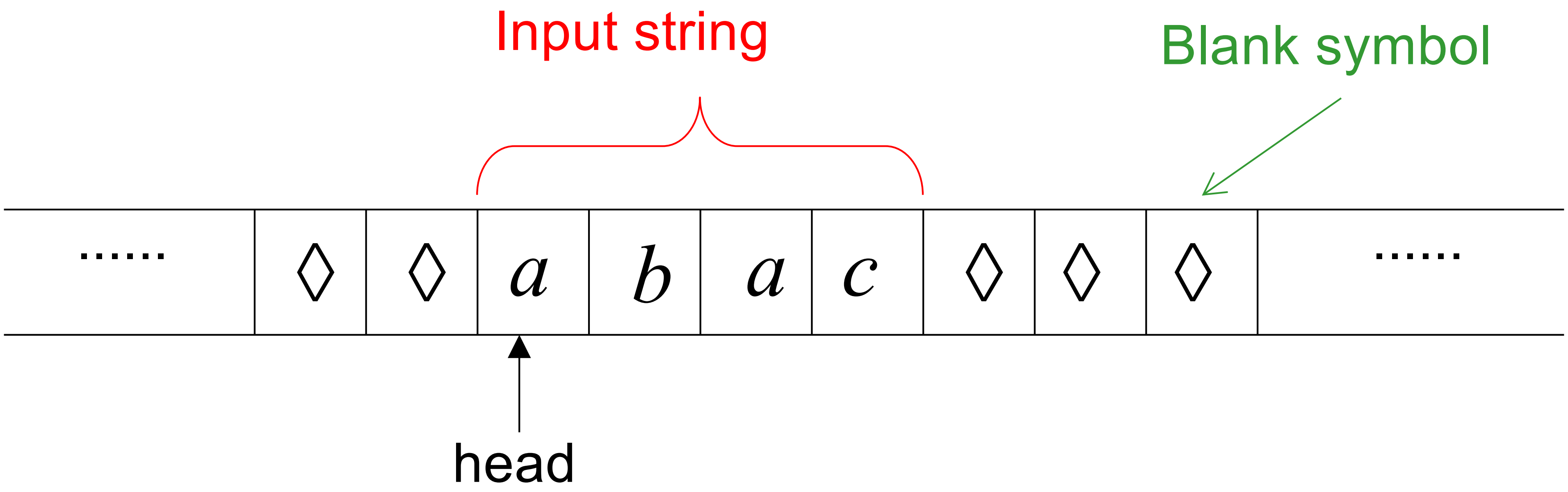


Time 2

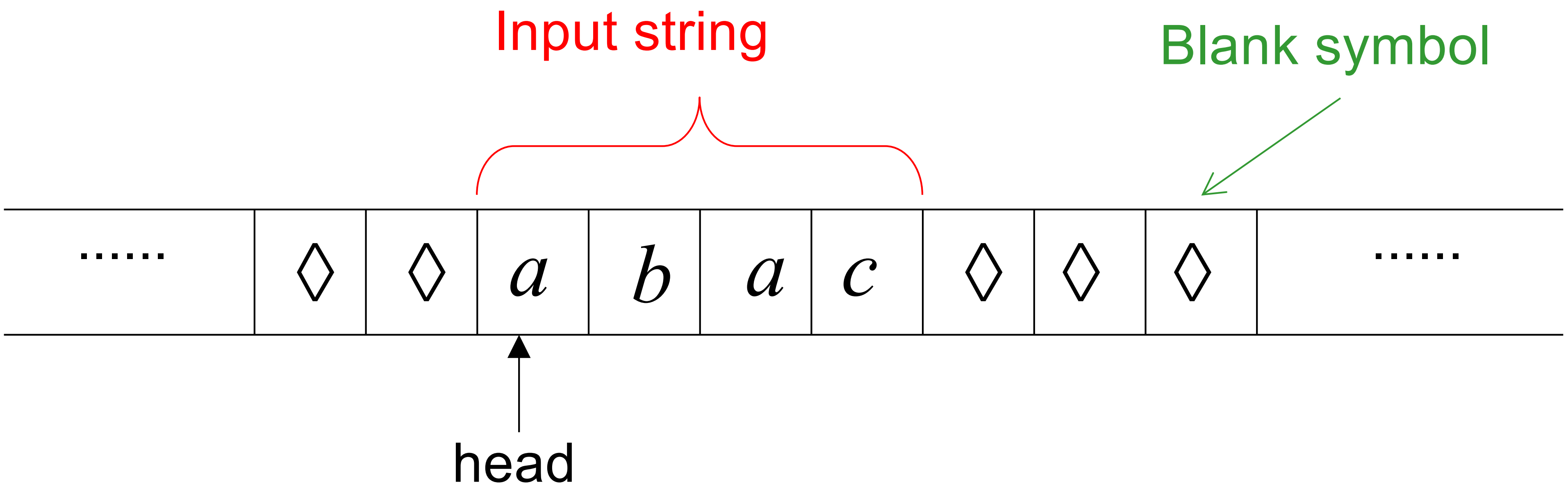


1. Reads b
2. Writes f
3. Moves Right

The Input String



Head starts at the leftmost position
of the input string



Remark: the input string is never empty

Turing Machine:

States

Input alphabet

Tape alphabet

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

Transition function

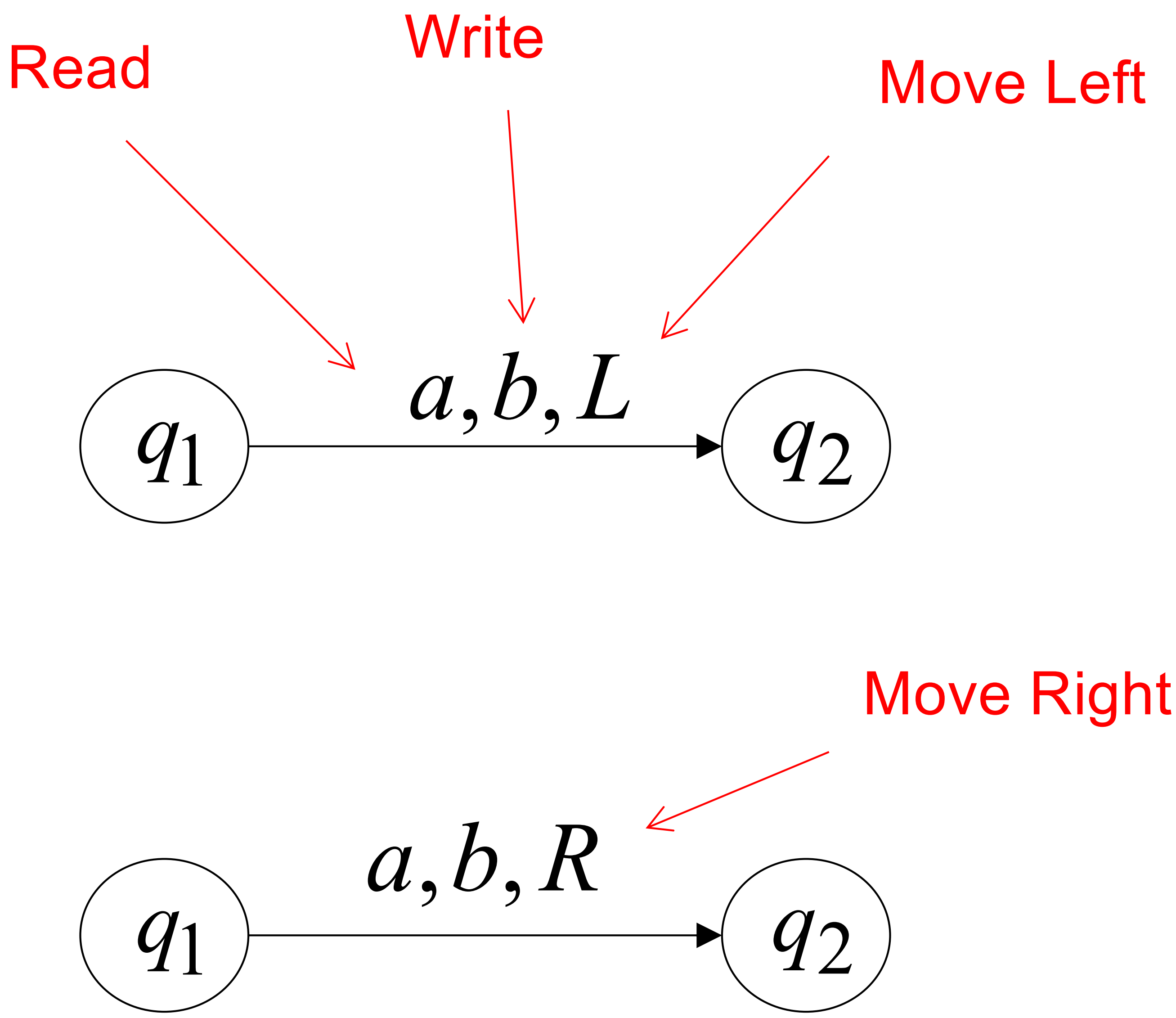
Initial state

blank

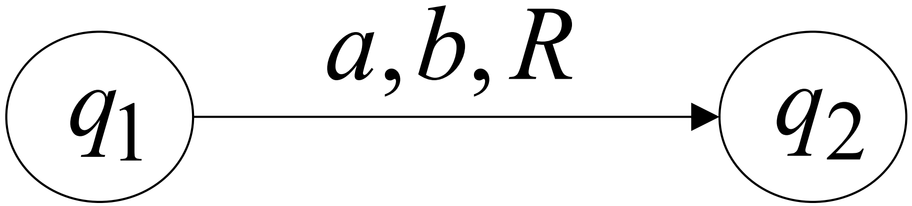
Final states

Ch 9

States & Transitions

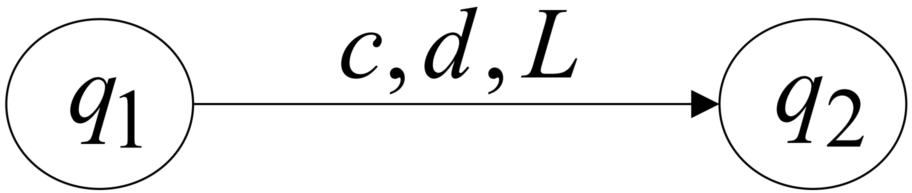


Transition Function (program)



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

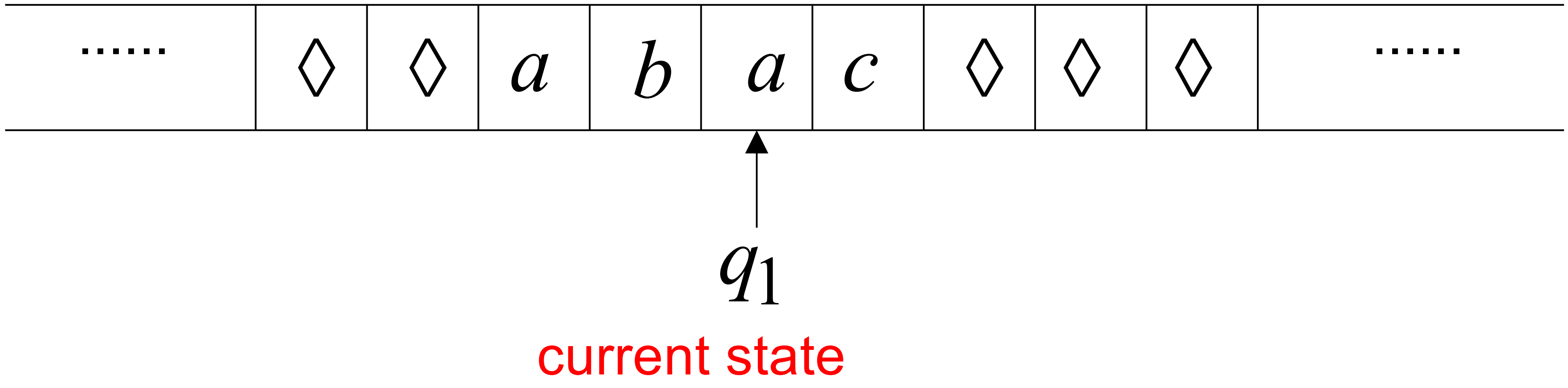
Transition Function

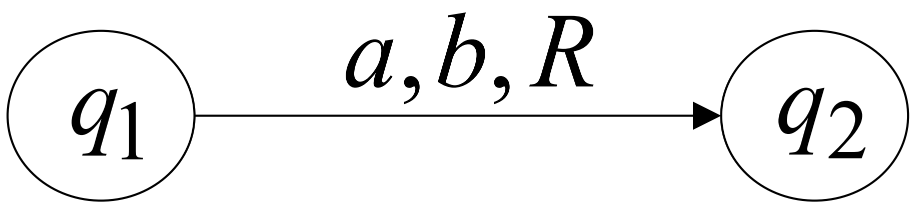


$$\delta(q_1, c) = (q_2, d, L)$$

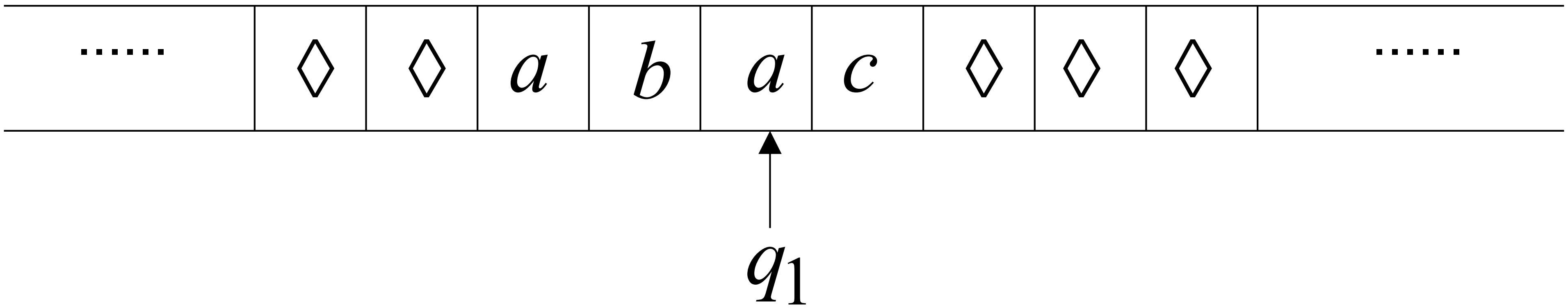
Example:

Time 1

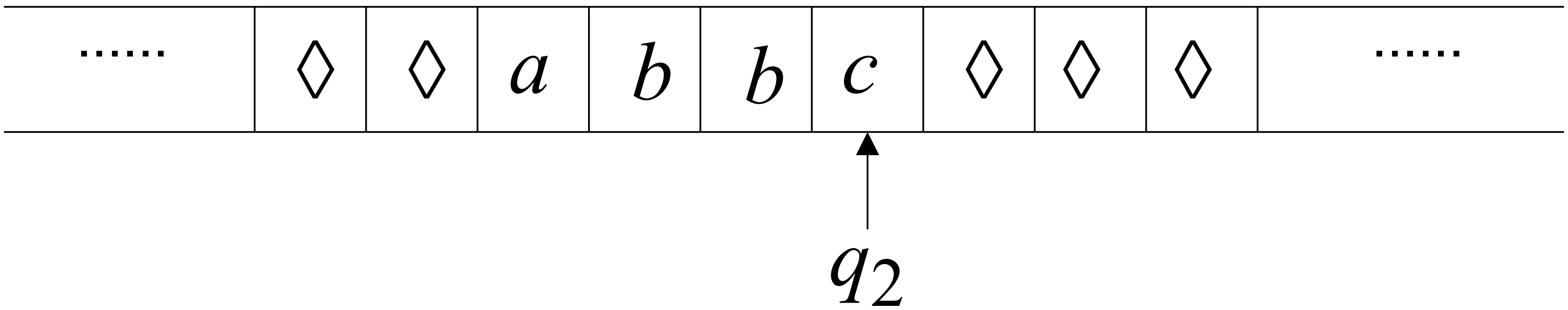


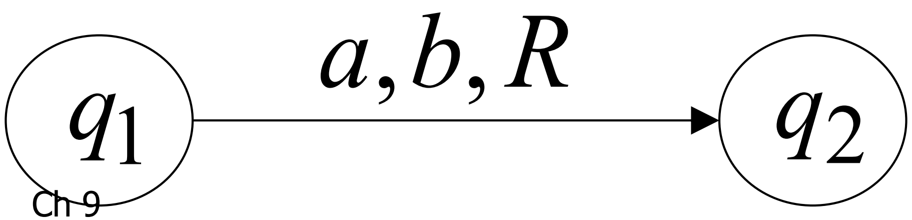


Time 1



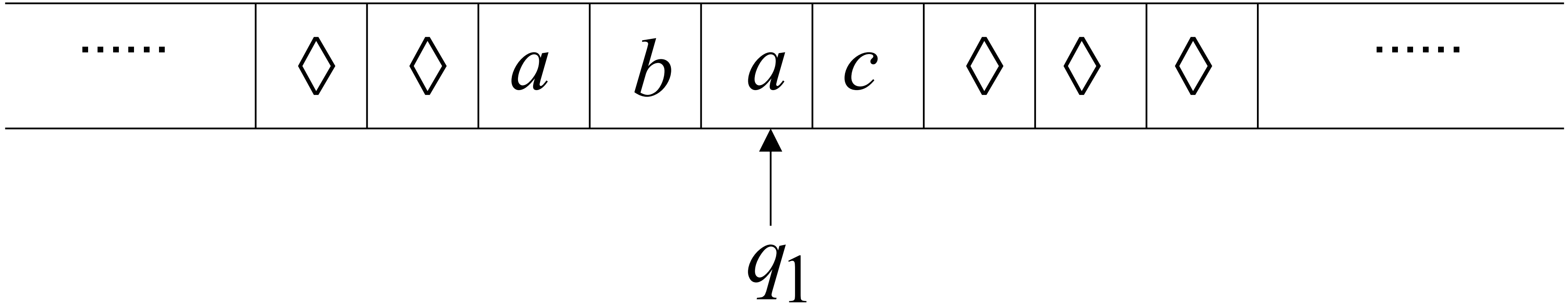
Time 2



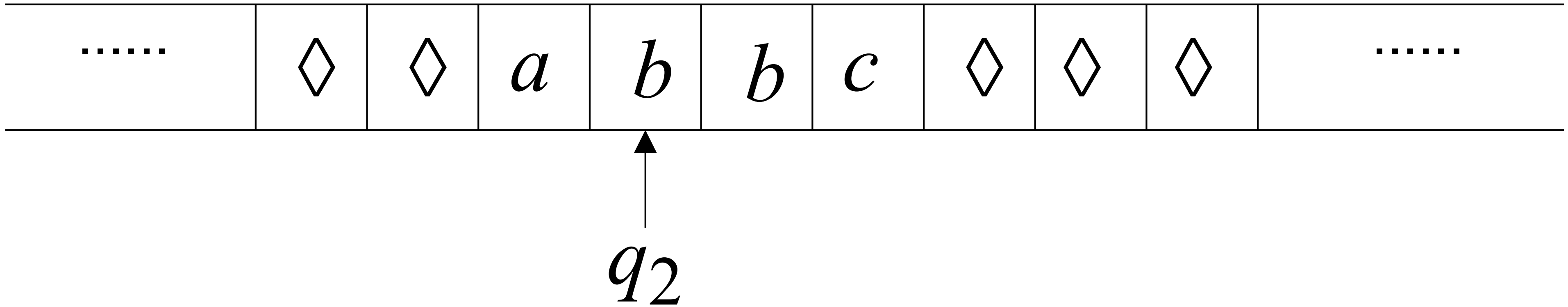


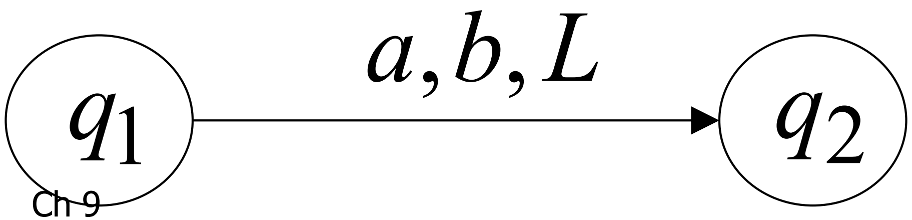
Example:

Time 1



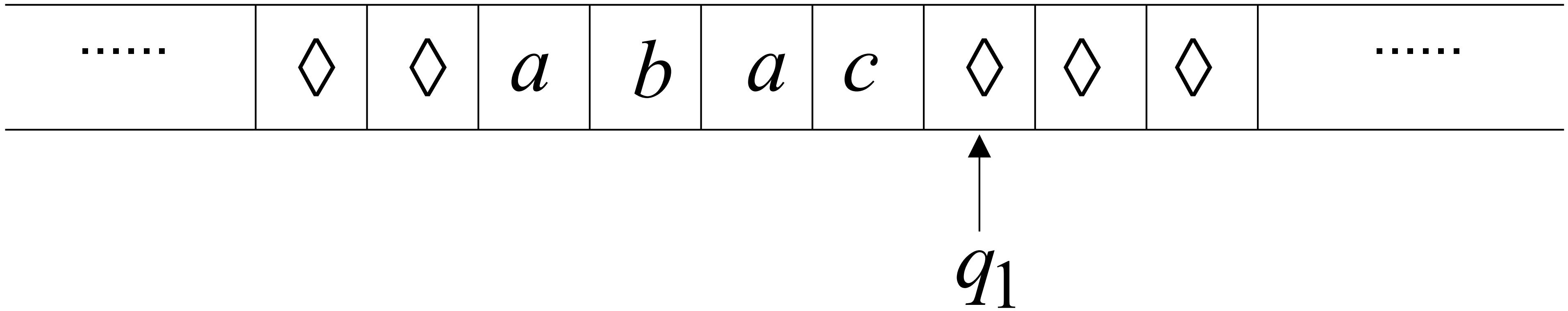
Time 2



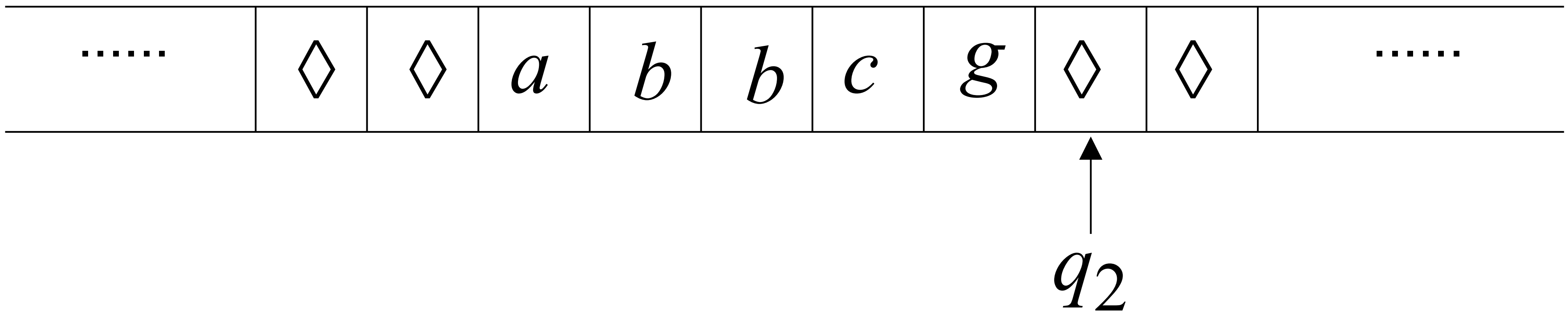


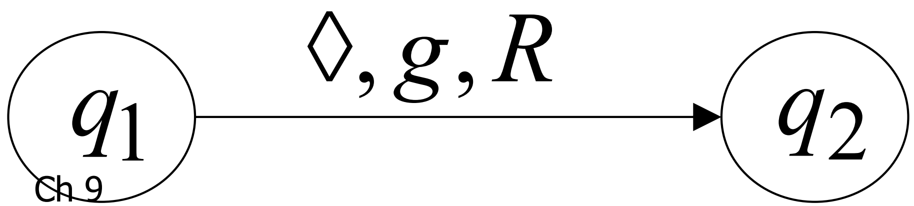
Example:

Time 1



Time 2

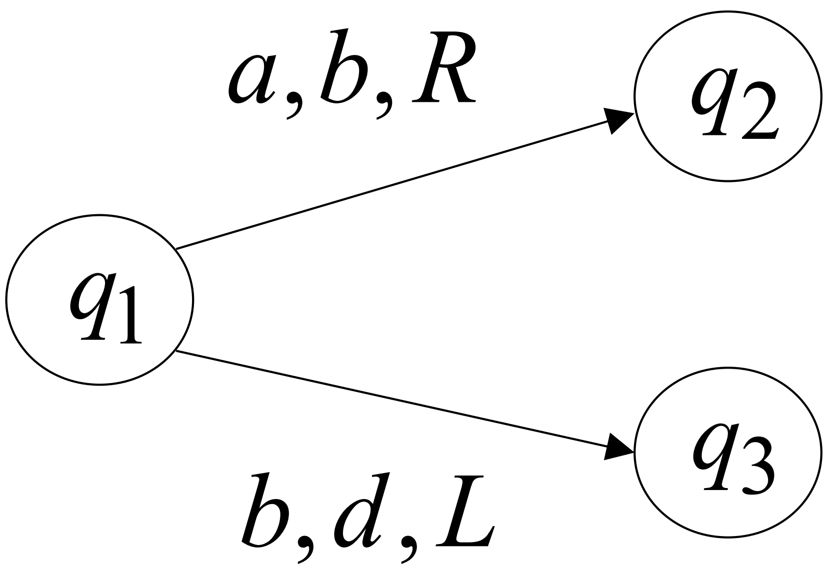




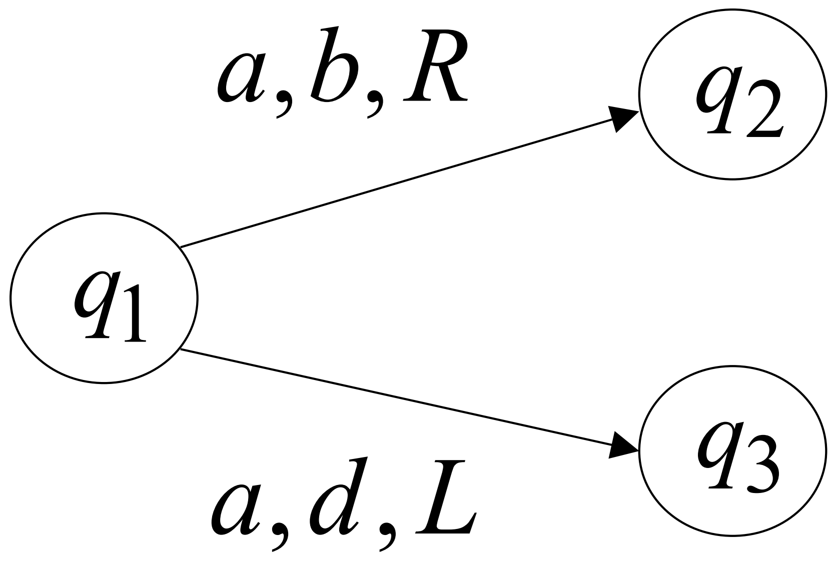
Determinism

Turing Machines are deterministic

<u>Allowed</u>



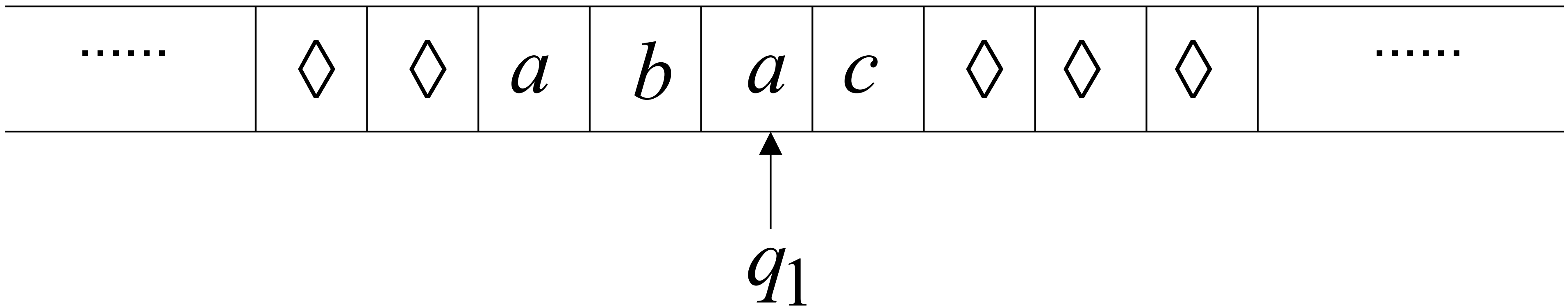
<u>**Not** Allowed</u>

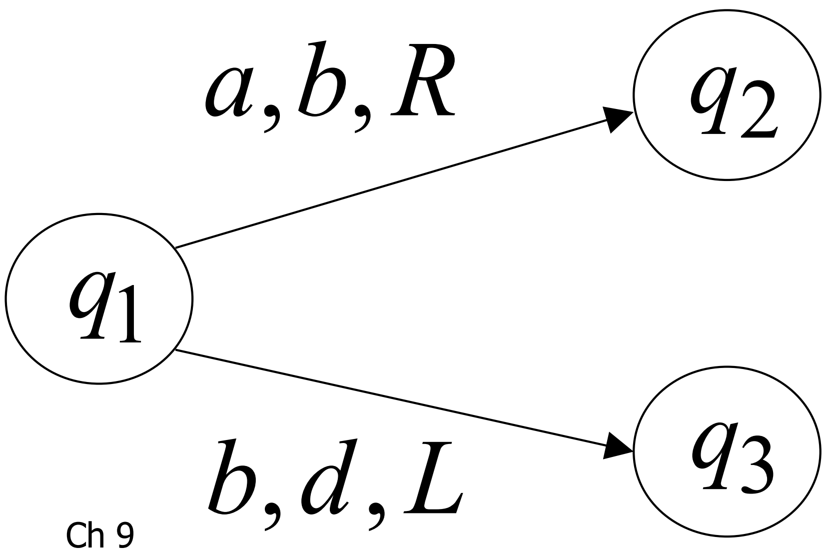


No lambda transitions allowed

Transition Function Is Partial

Example:



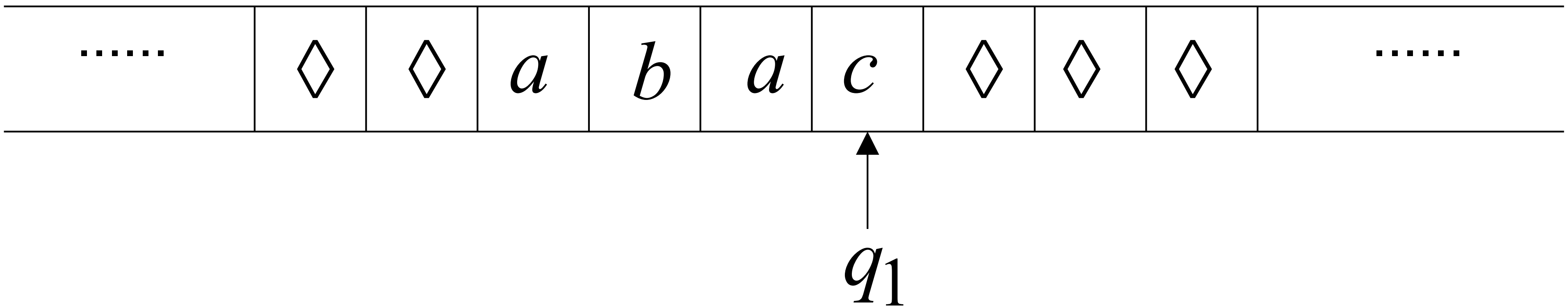


Allowed:

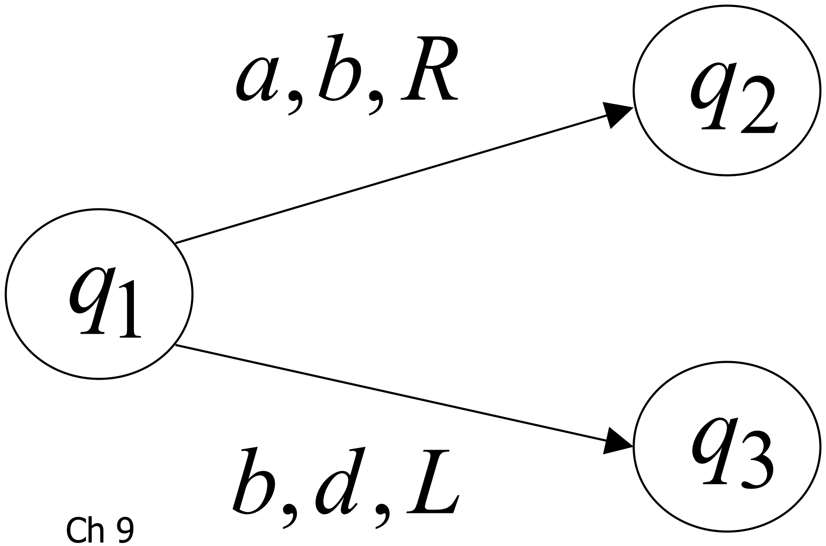
Halting

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

Example:



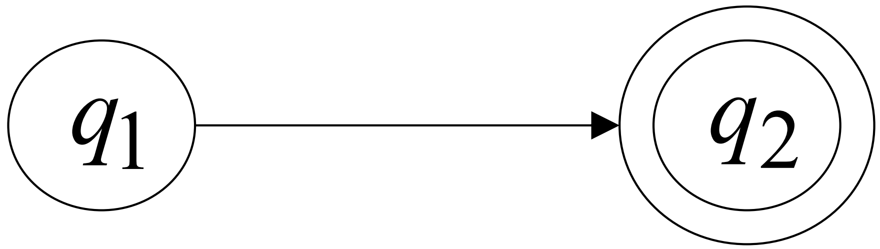
No transition
for input symbol c



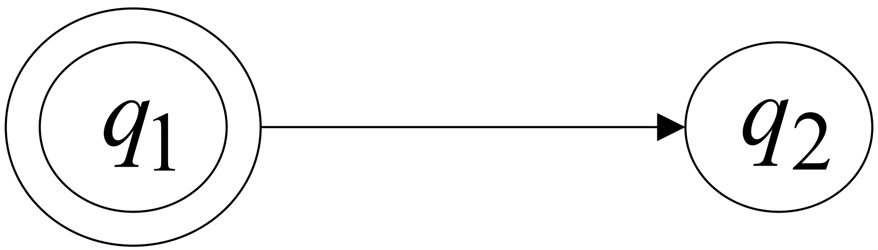
No possible transition

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

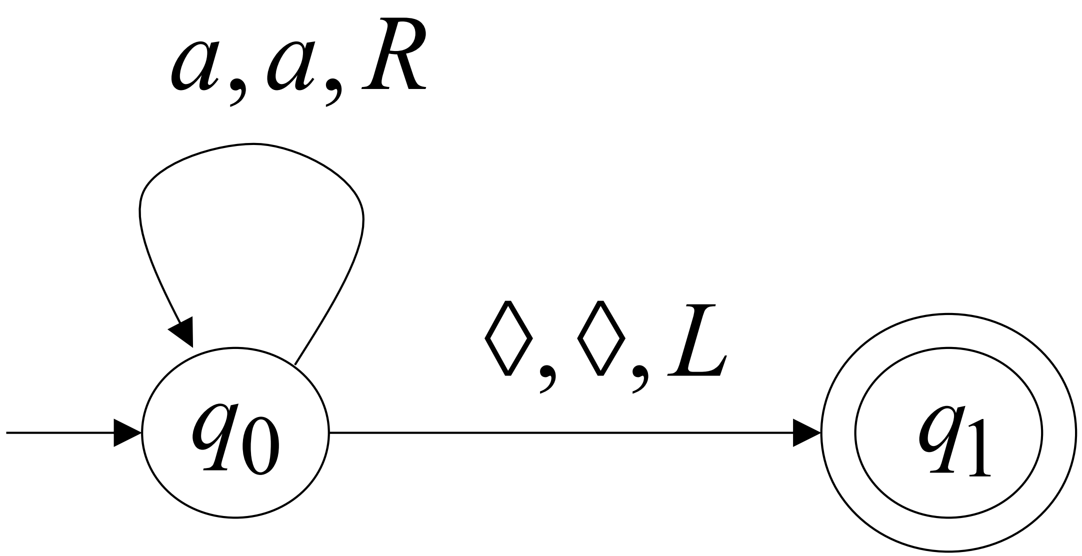
If machine enters an *infinite loop*

Turing Machine Example

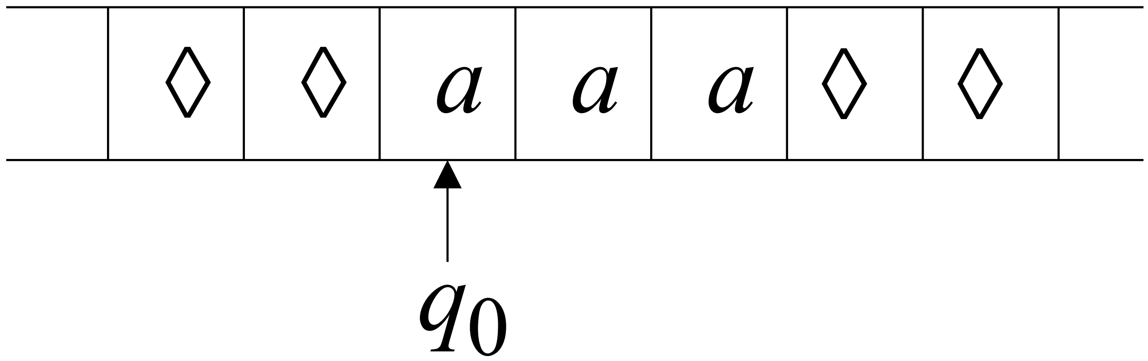
A Turing machine that accepts the language:

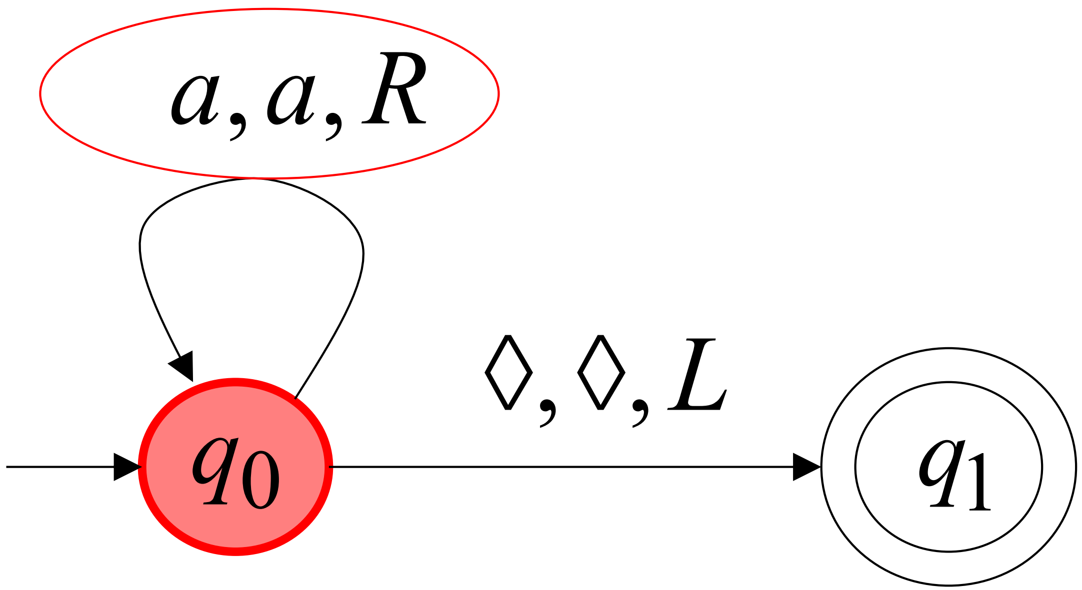
$$aa^*$$

$\Sigma = \{a, b\}$

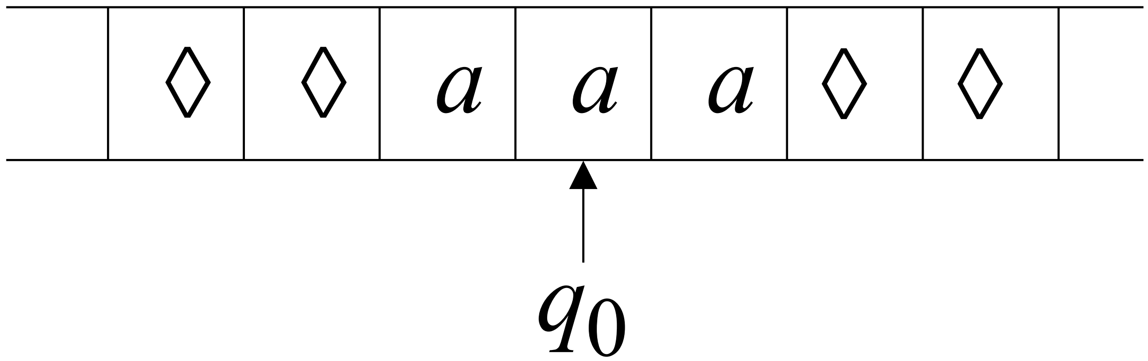


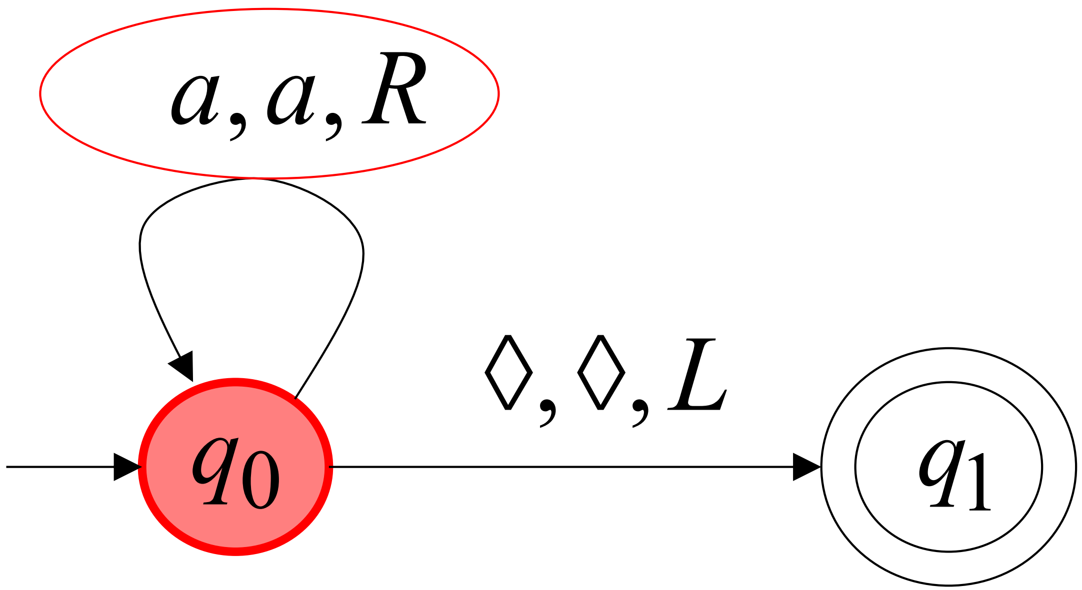
Time 0



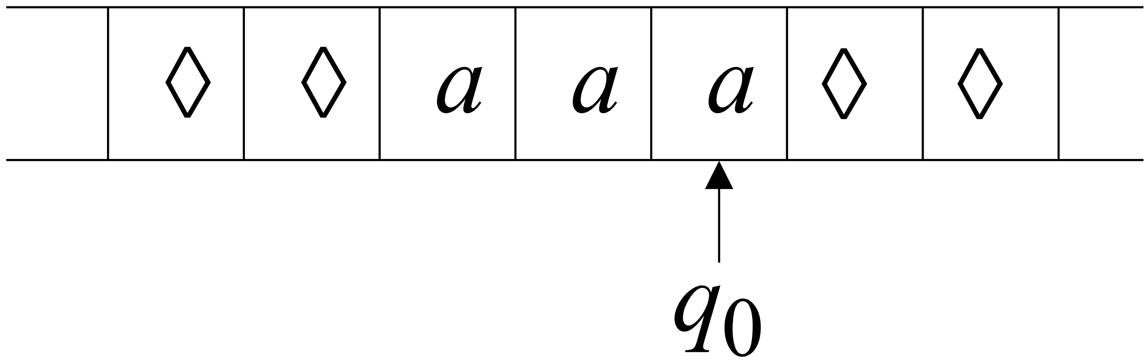


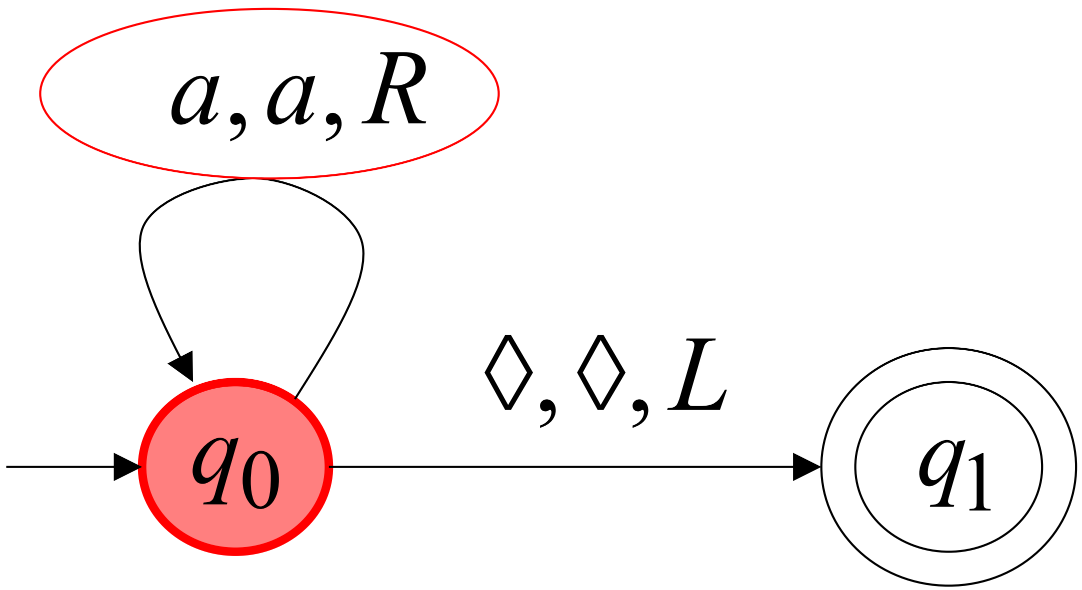
Time 1



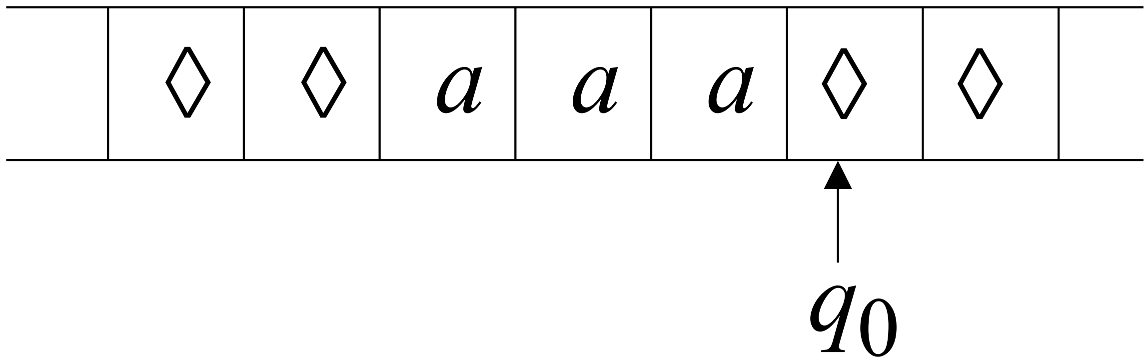


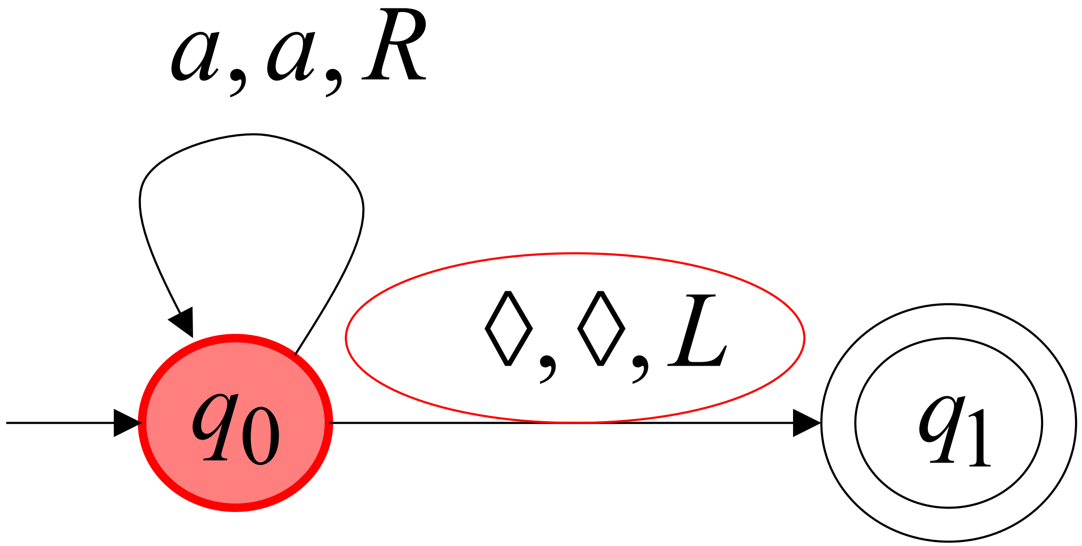
Time 2



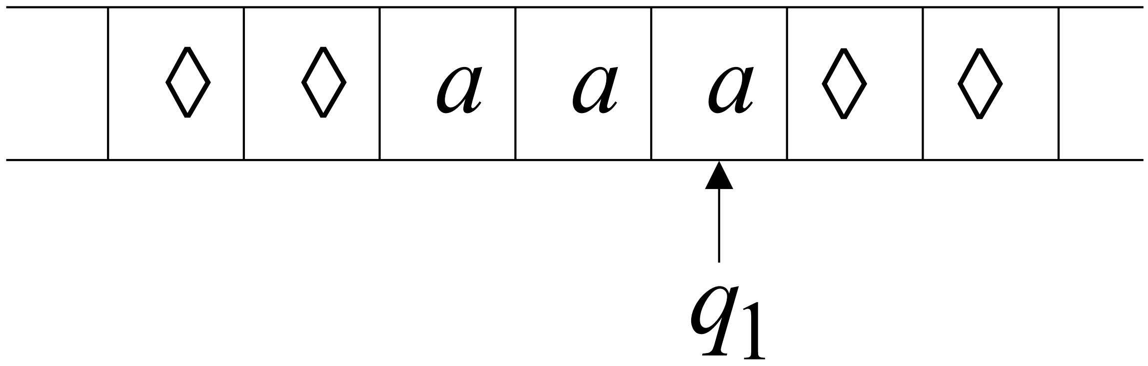


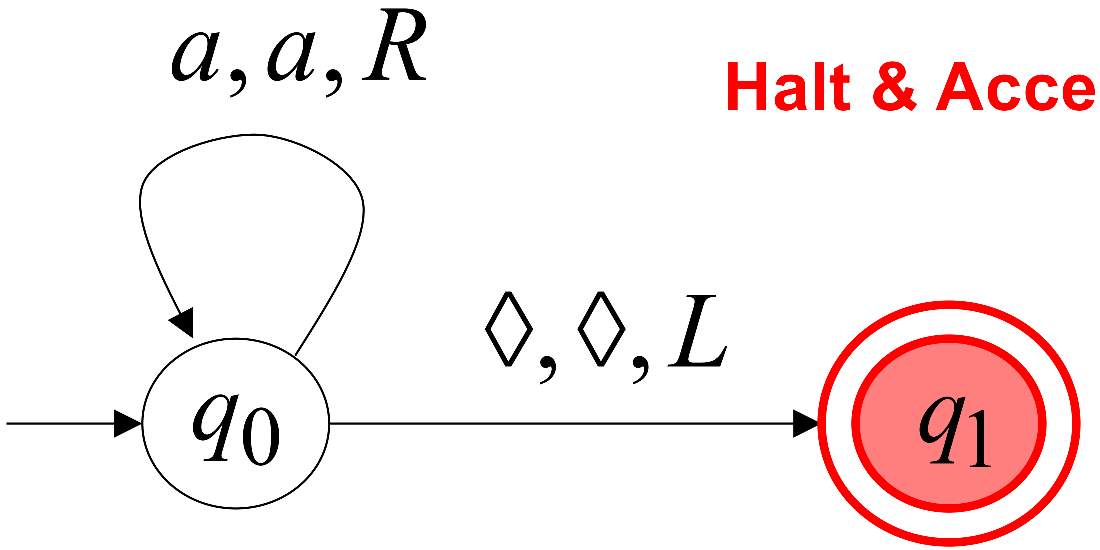
Time 3





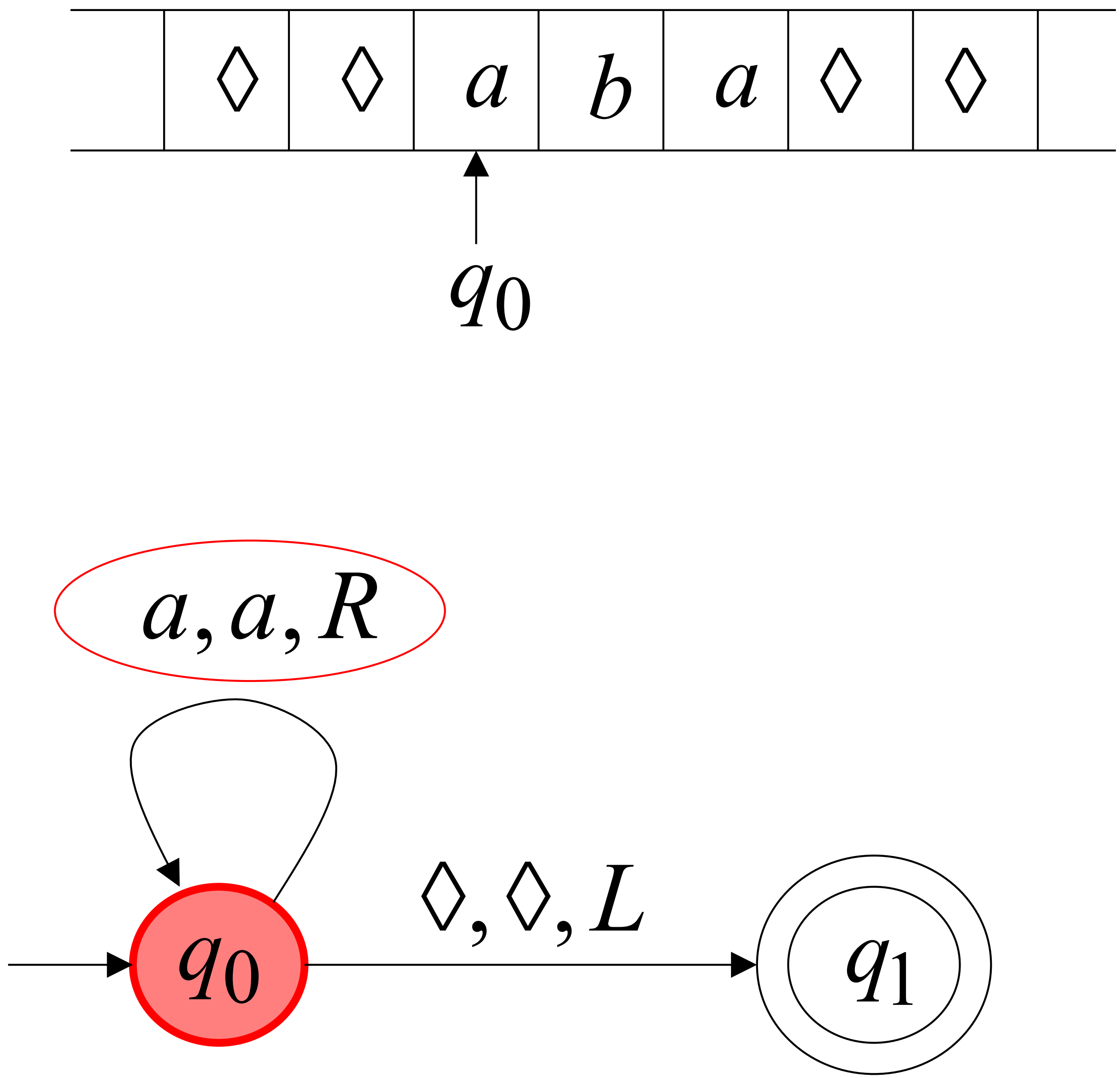
Time 4



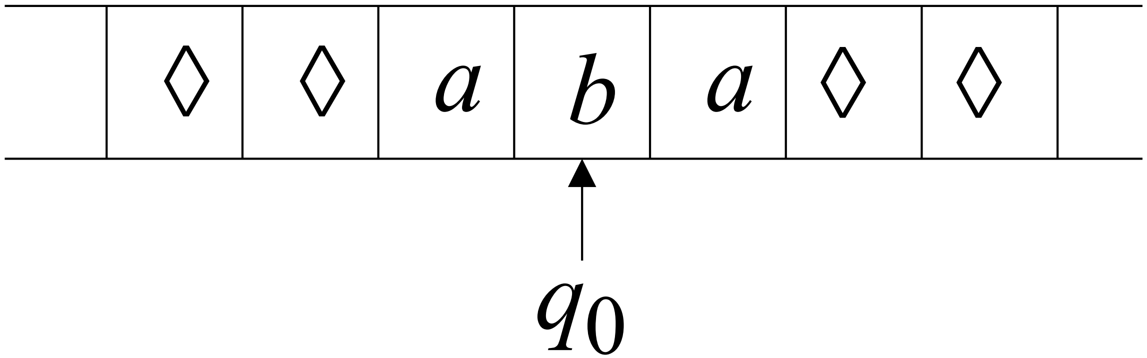


Rejection Example

Time 0

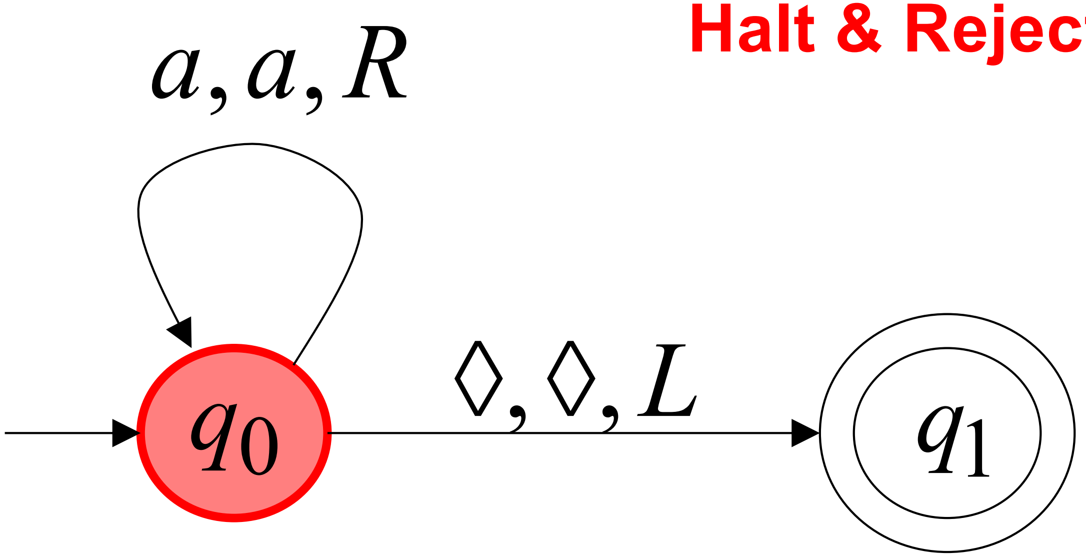


Time 1

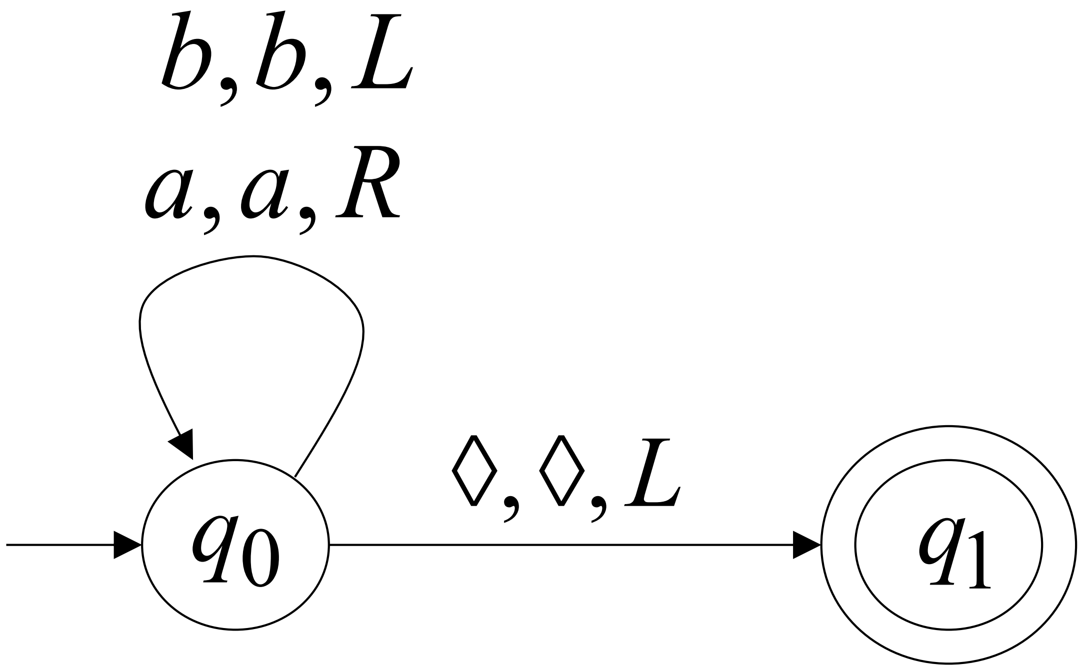


No possible Transition

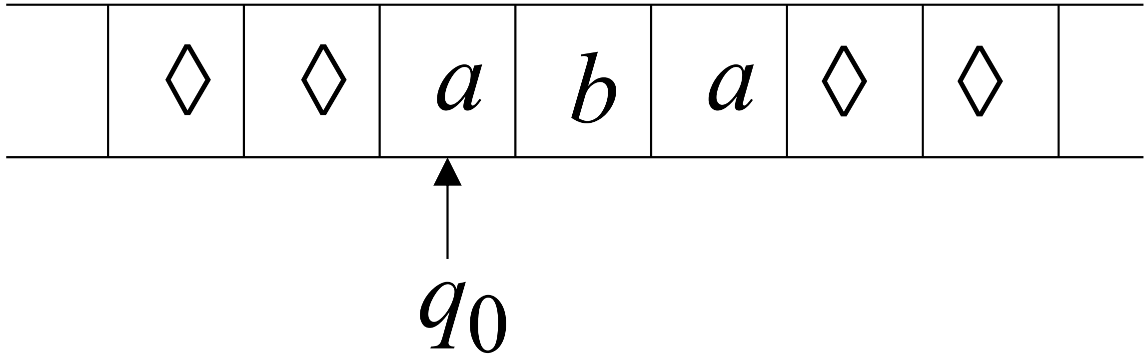
Halt & Reject

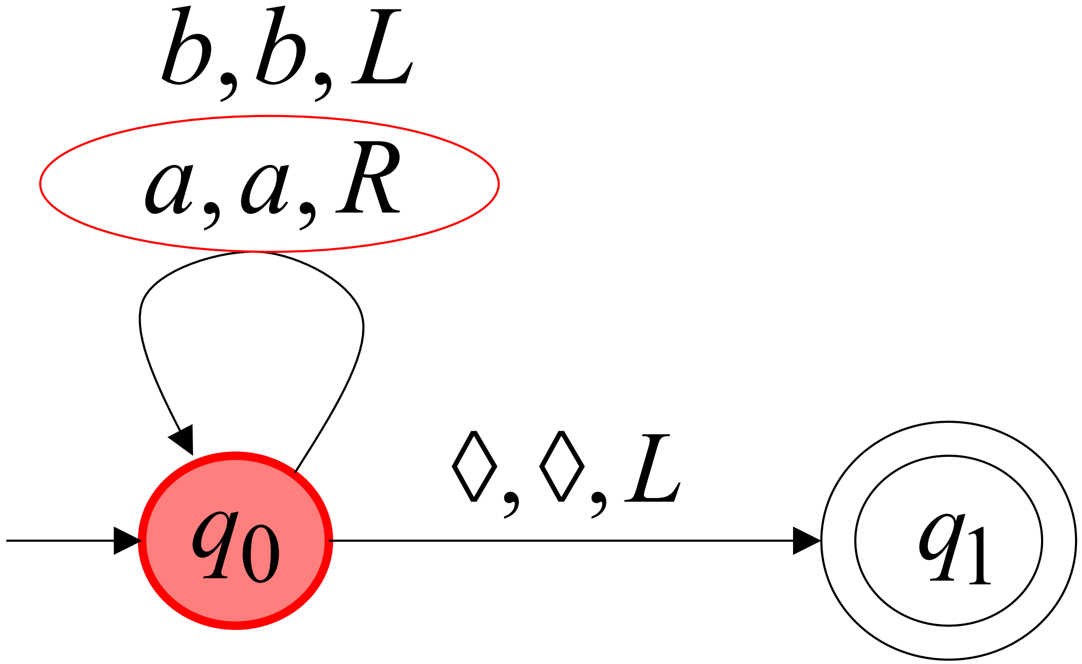


Infinite Loop Example

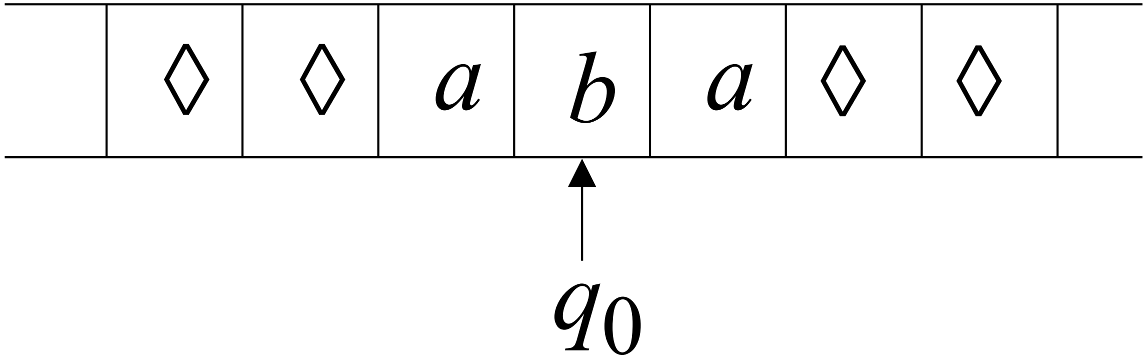


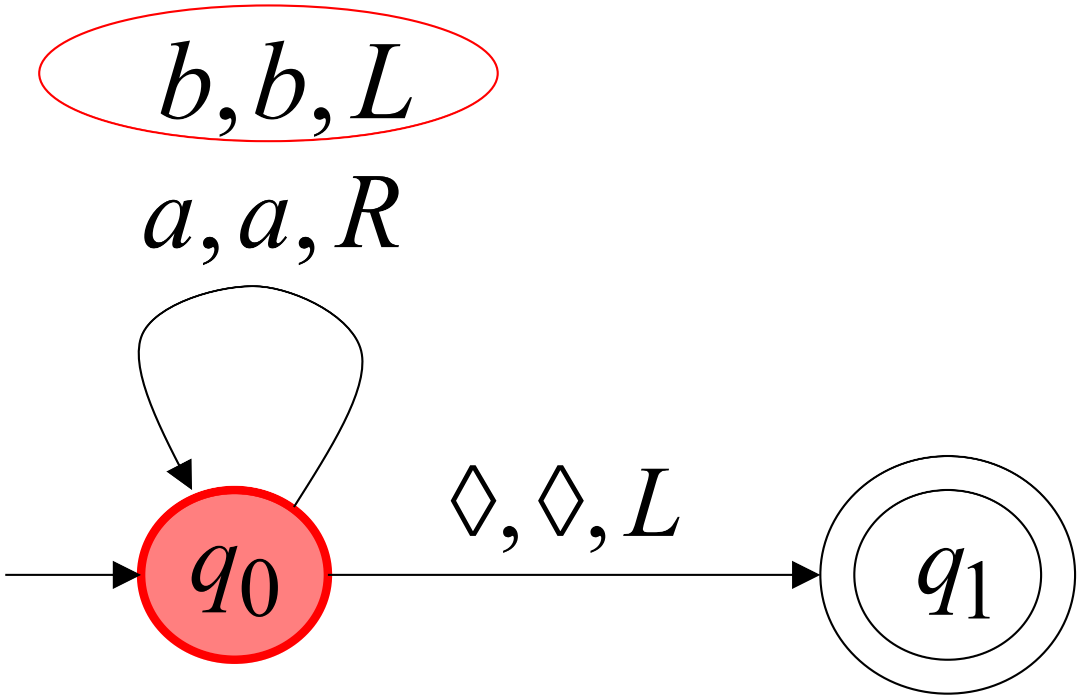
Time 0



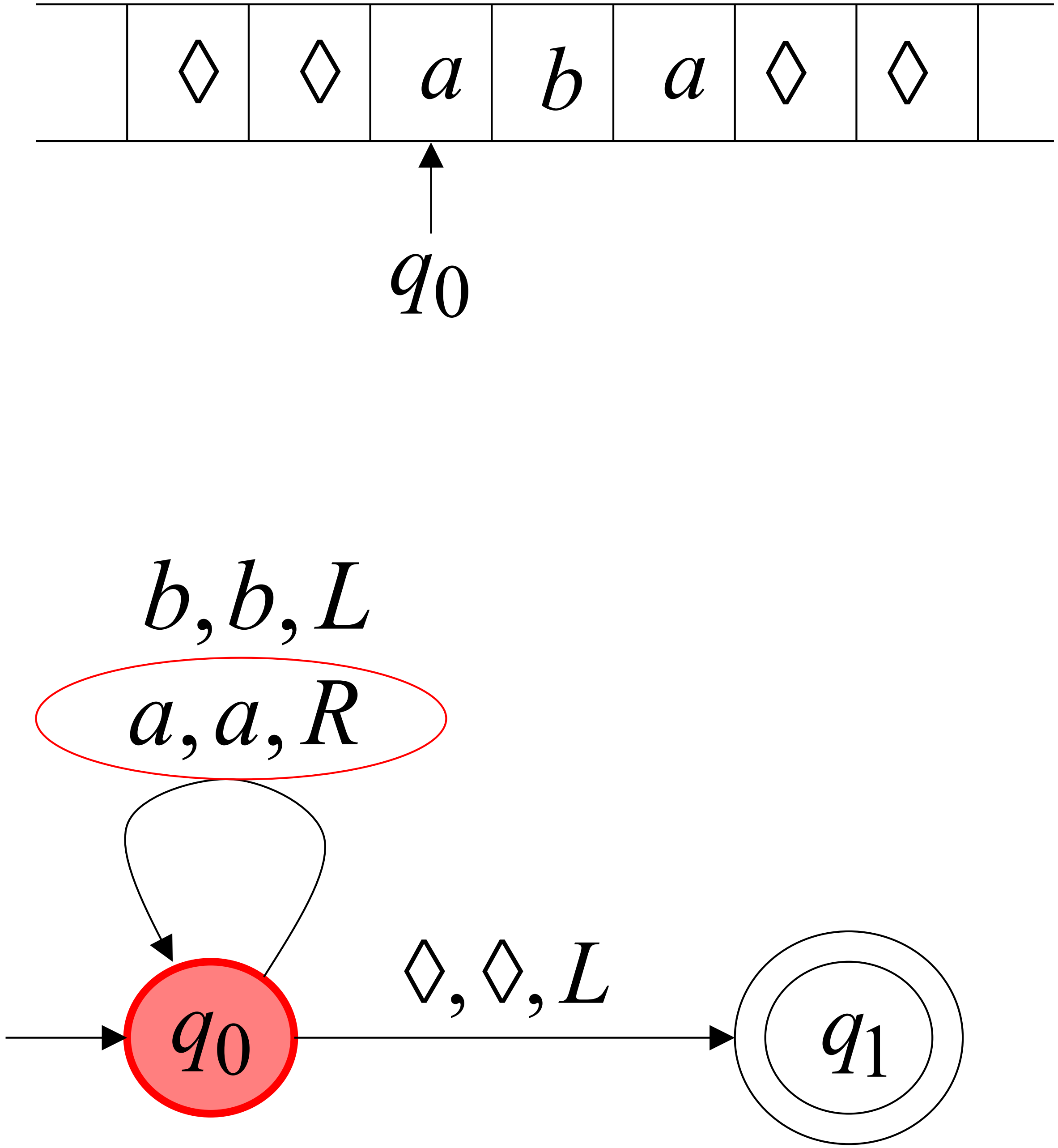


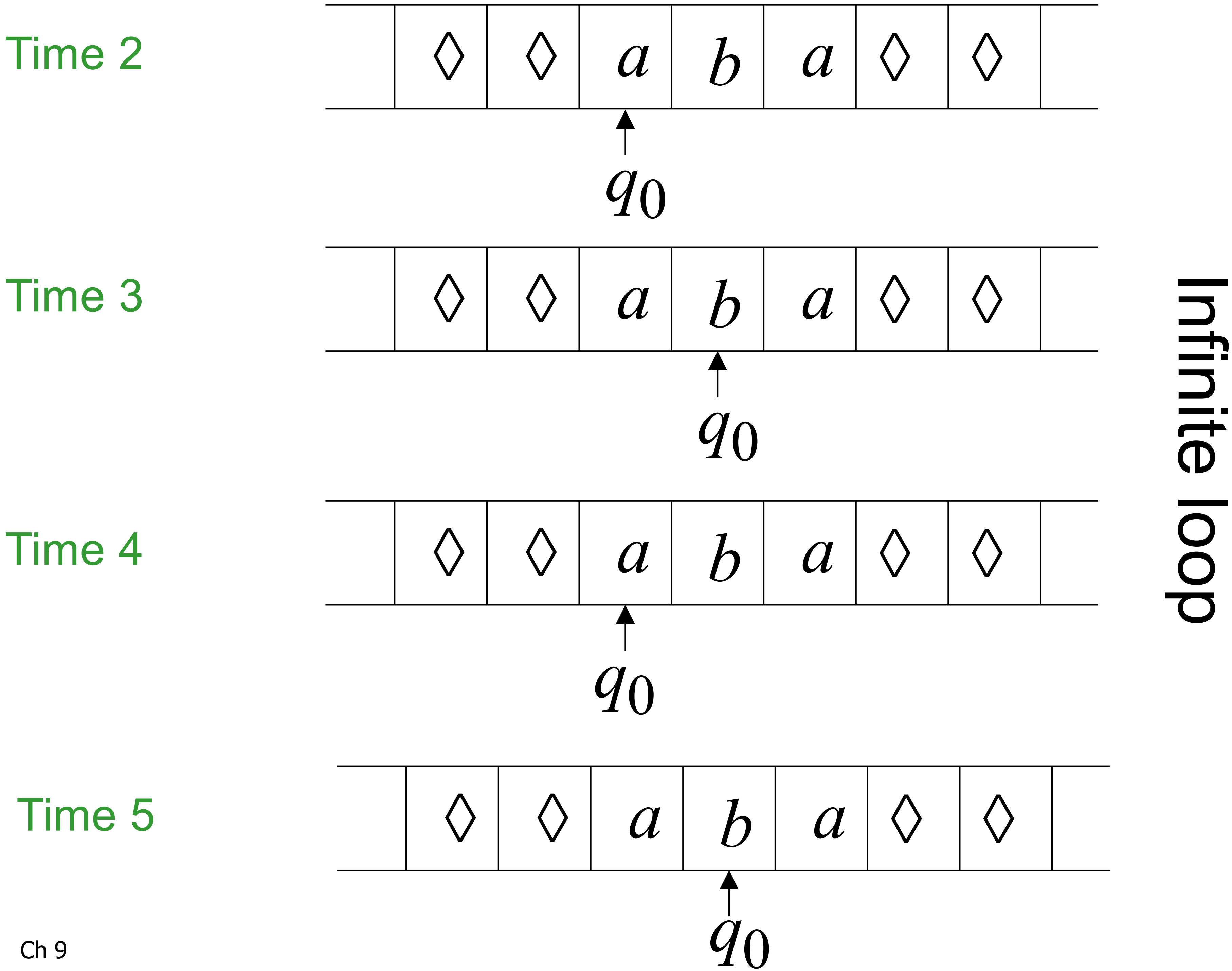
Time 1





Time 2




Time 2
◊ ◊ a b a ◊ ◊
q0
Time 3
◊ ◊ a b a ◊ ◊
q0
Time 4
◊ ◊ a b a ◊ ◊
q0
Time 5
◊ ◊ a b a ◊ ◊
q0
Infinite loop

Because of the infinite loop:

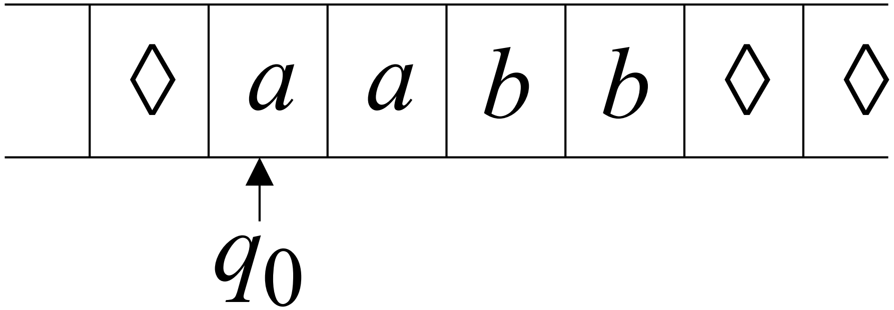
- The final state cannot be reached

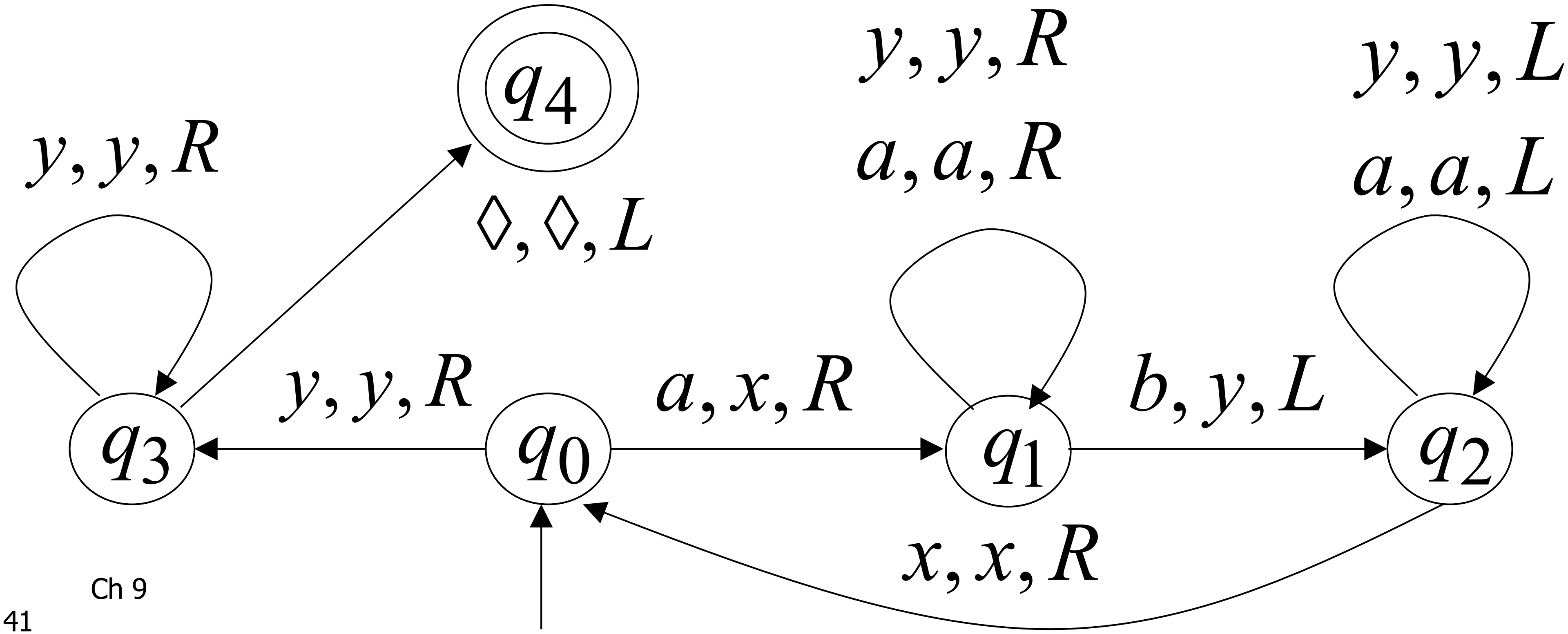
- The machine never halts

- The input is not accepted

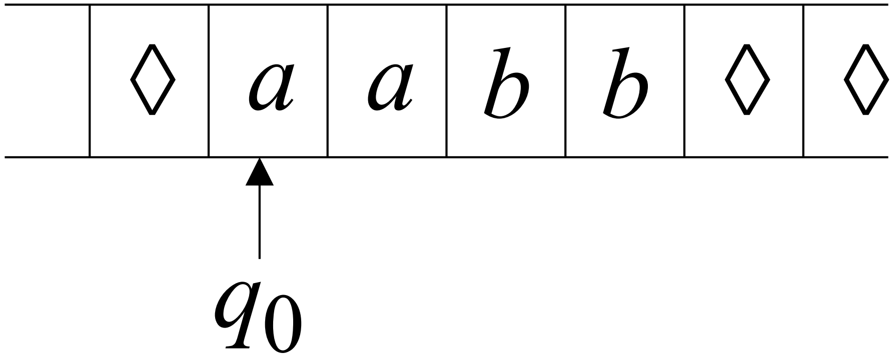
Example 9.7

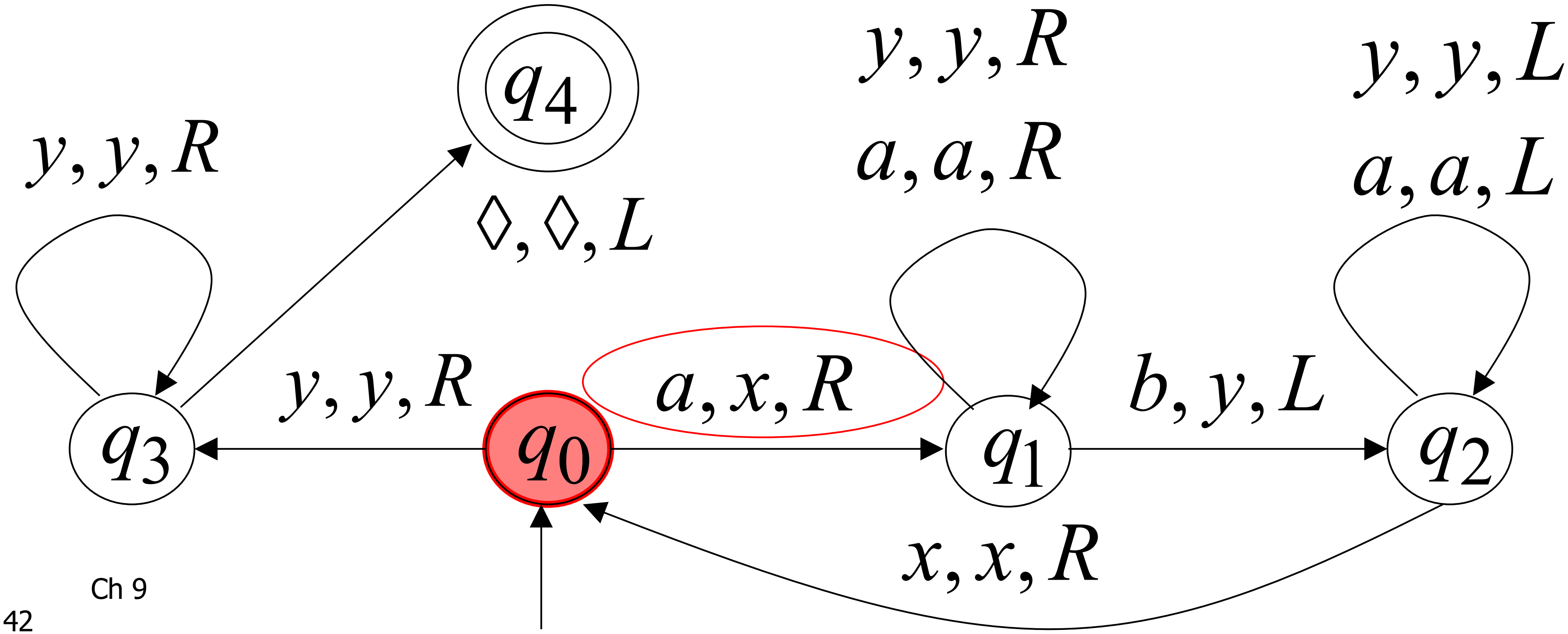
Turing machine for the language $\{a^n b^n\}$



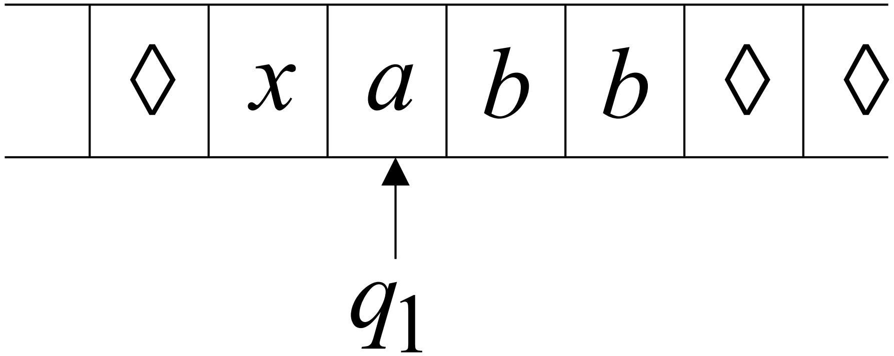


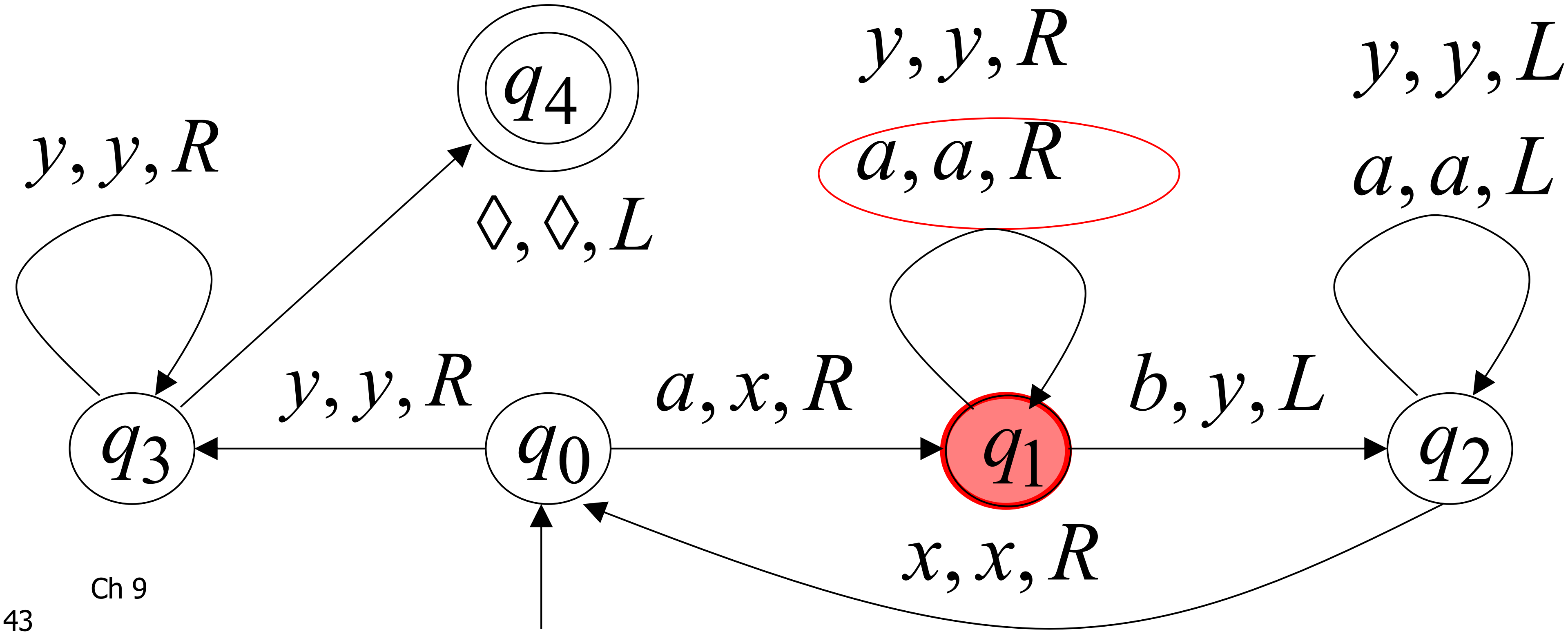
Time 0



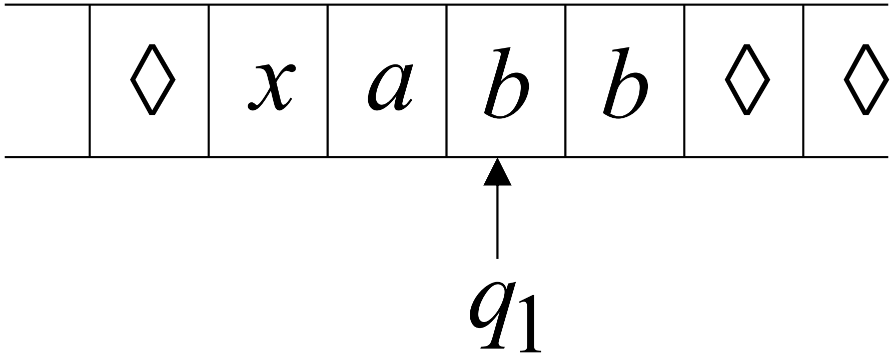


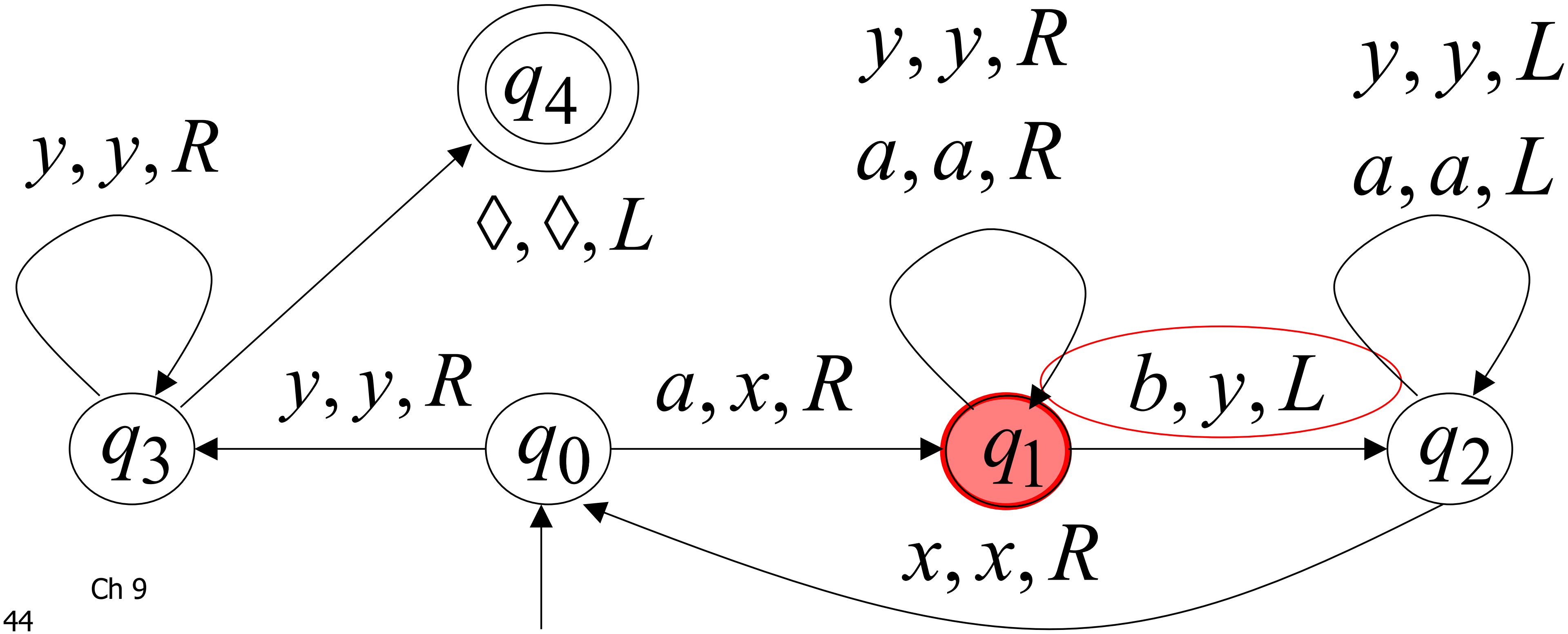
Time 1



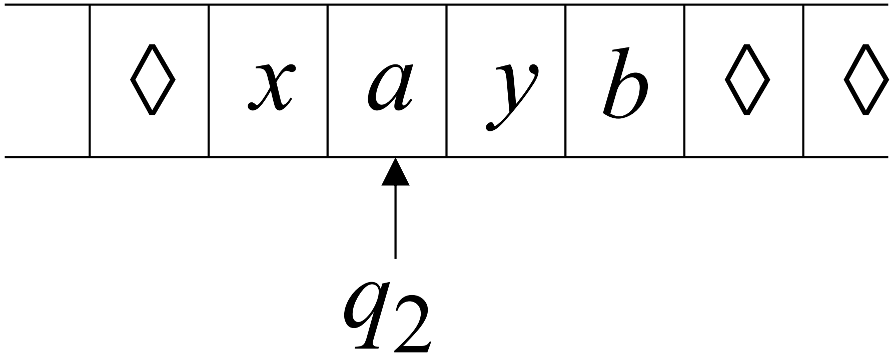


Time 2

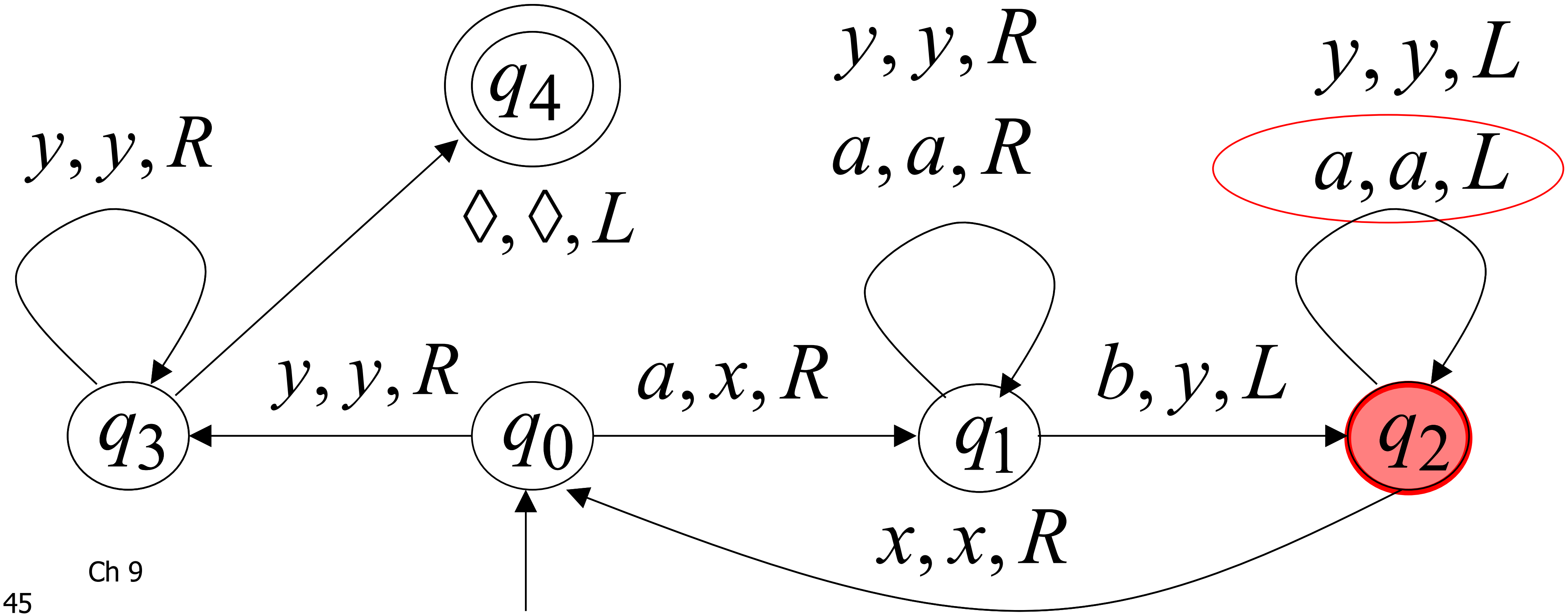




Time 3

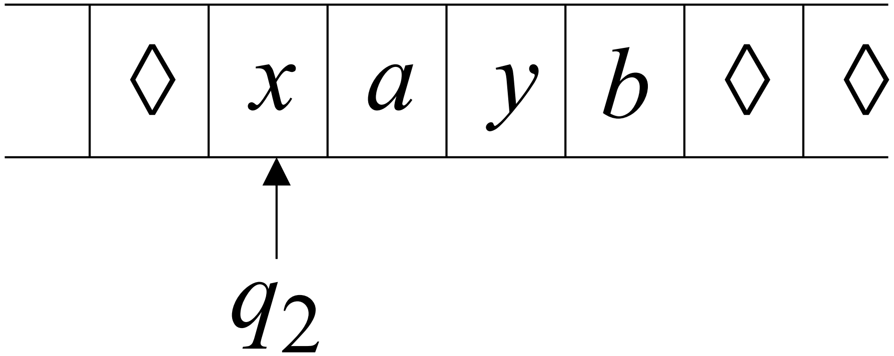


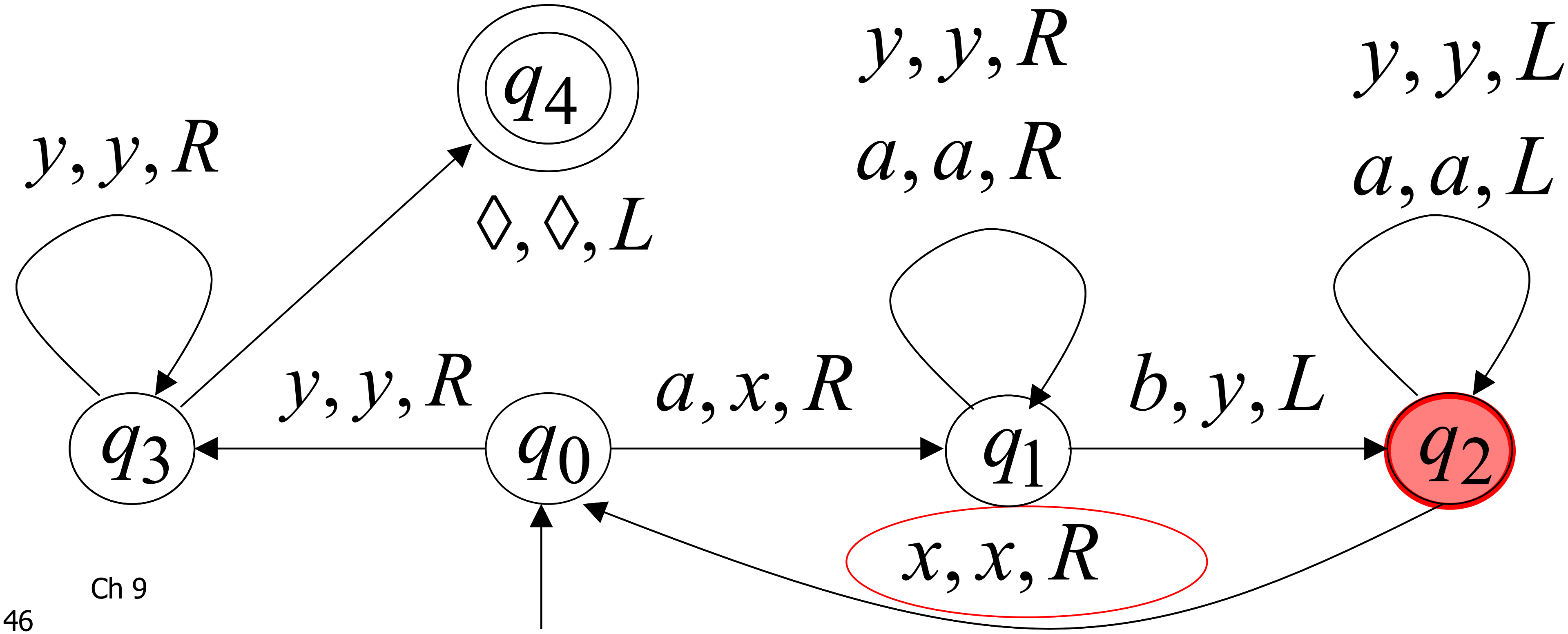
◊
x
a
y
b
◊
◊
q_2



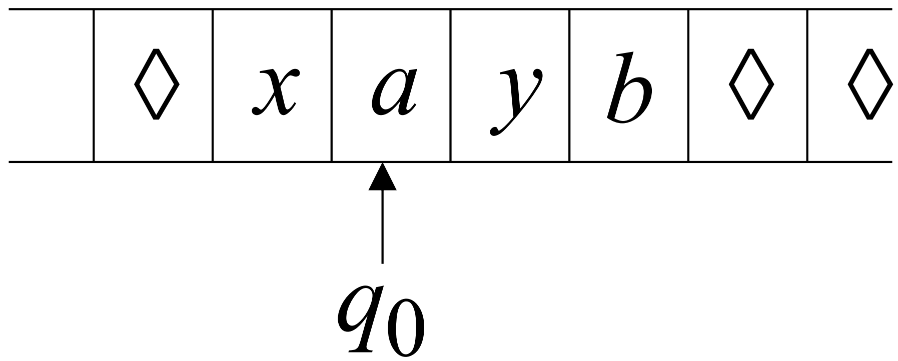
y, y, R
q_4
$\Diamond, \Diamond, L$
y, y, R
a, a, R
y, y, L
a, a, L
q_3
y, y, R
q_0
a, x, R
q_1
b, y, L
q_2
x, x, R

Time 4

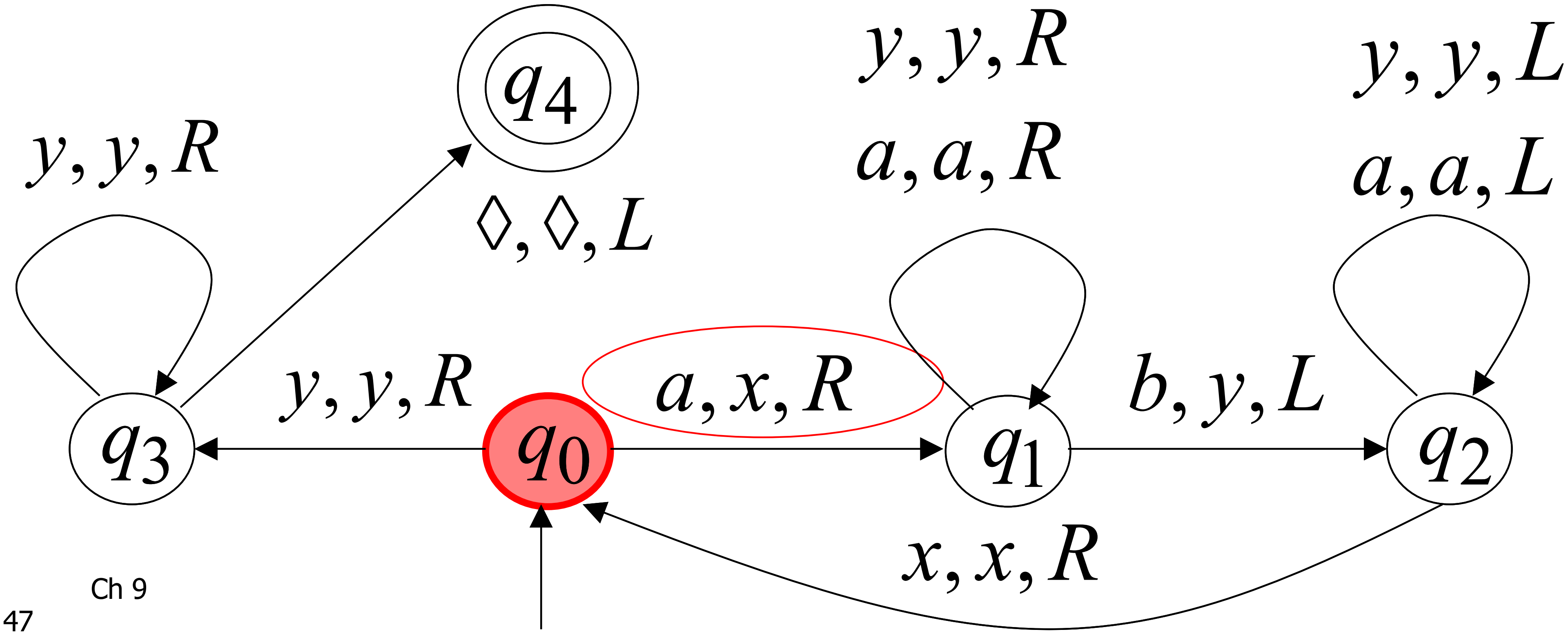




Time 5

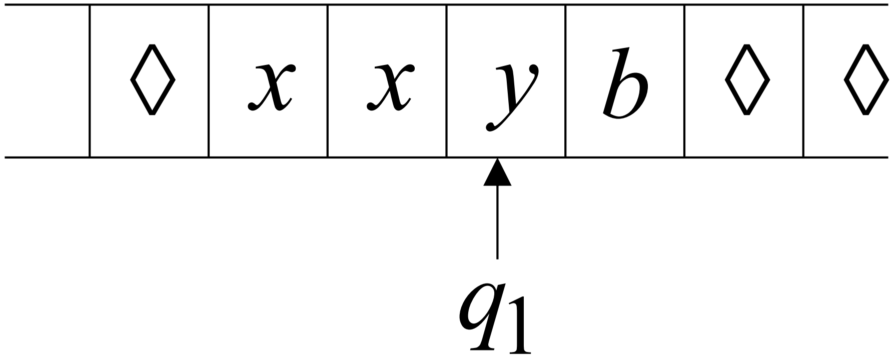


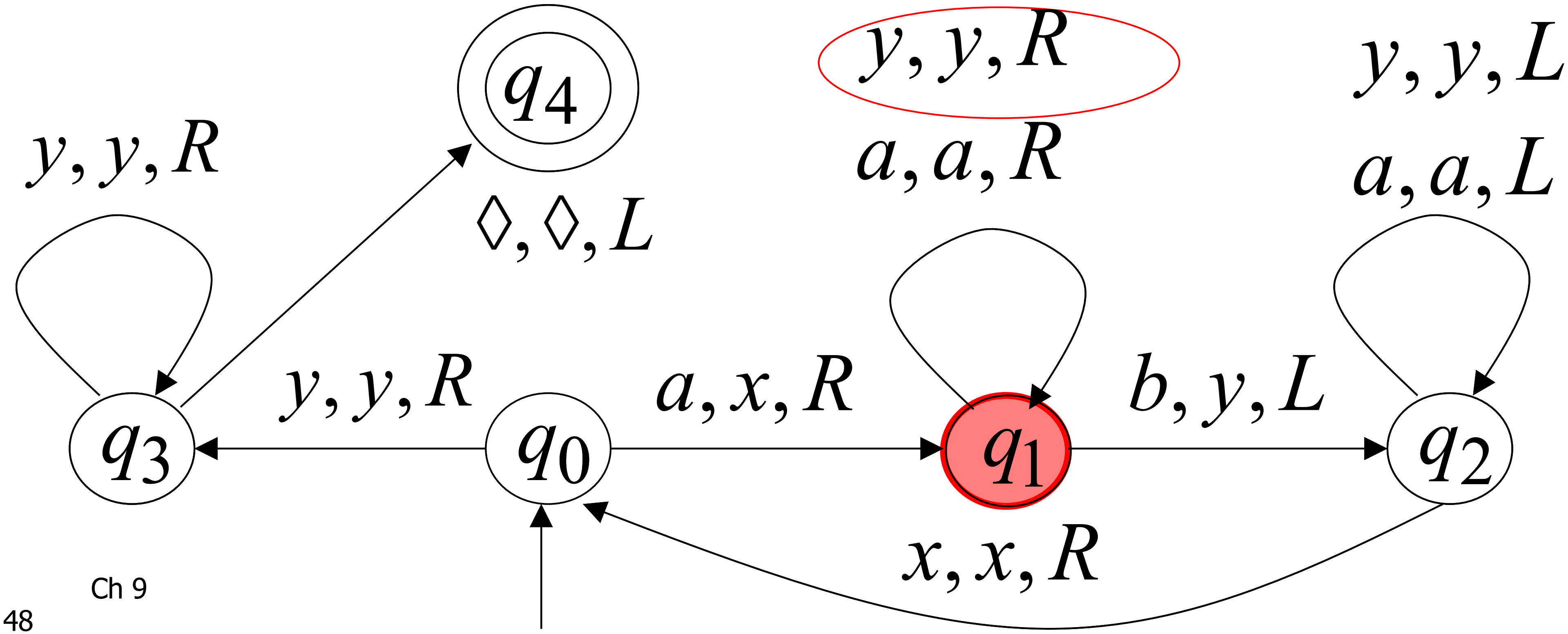
◊ x a y b ◊ ◊
q_0



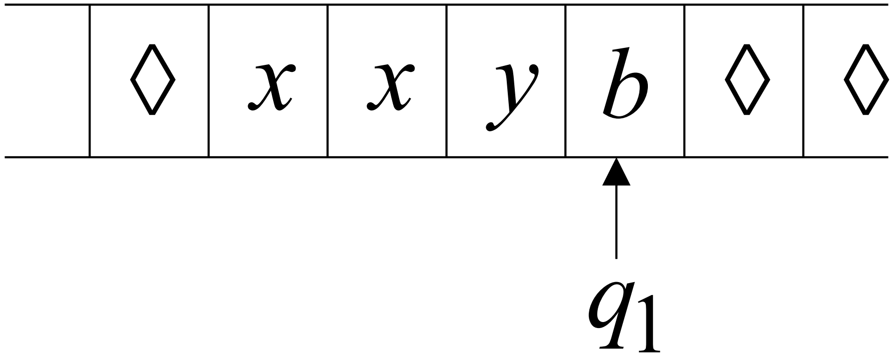
y, y, R
q_4
$\Diamond, \Diamond, L$
y, y, R
a, a, R
y, y, L
a, a, L
y, y, R
a, x, R
b, y, L
q_3
q_0
q_1
q_2
x, x, R

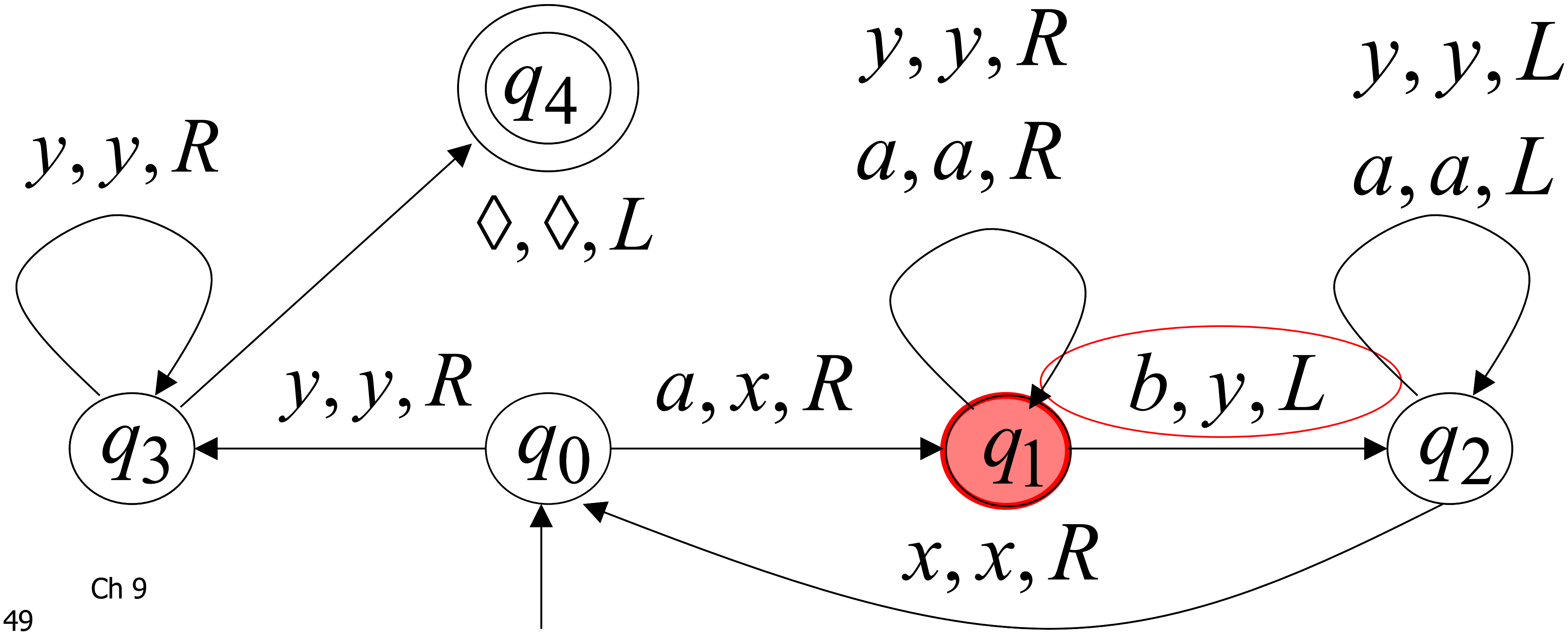
Time 6


◊
x
x
y
b
◊
◊
q_1

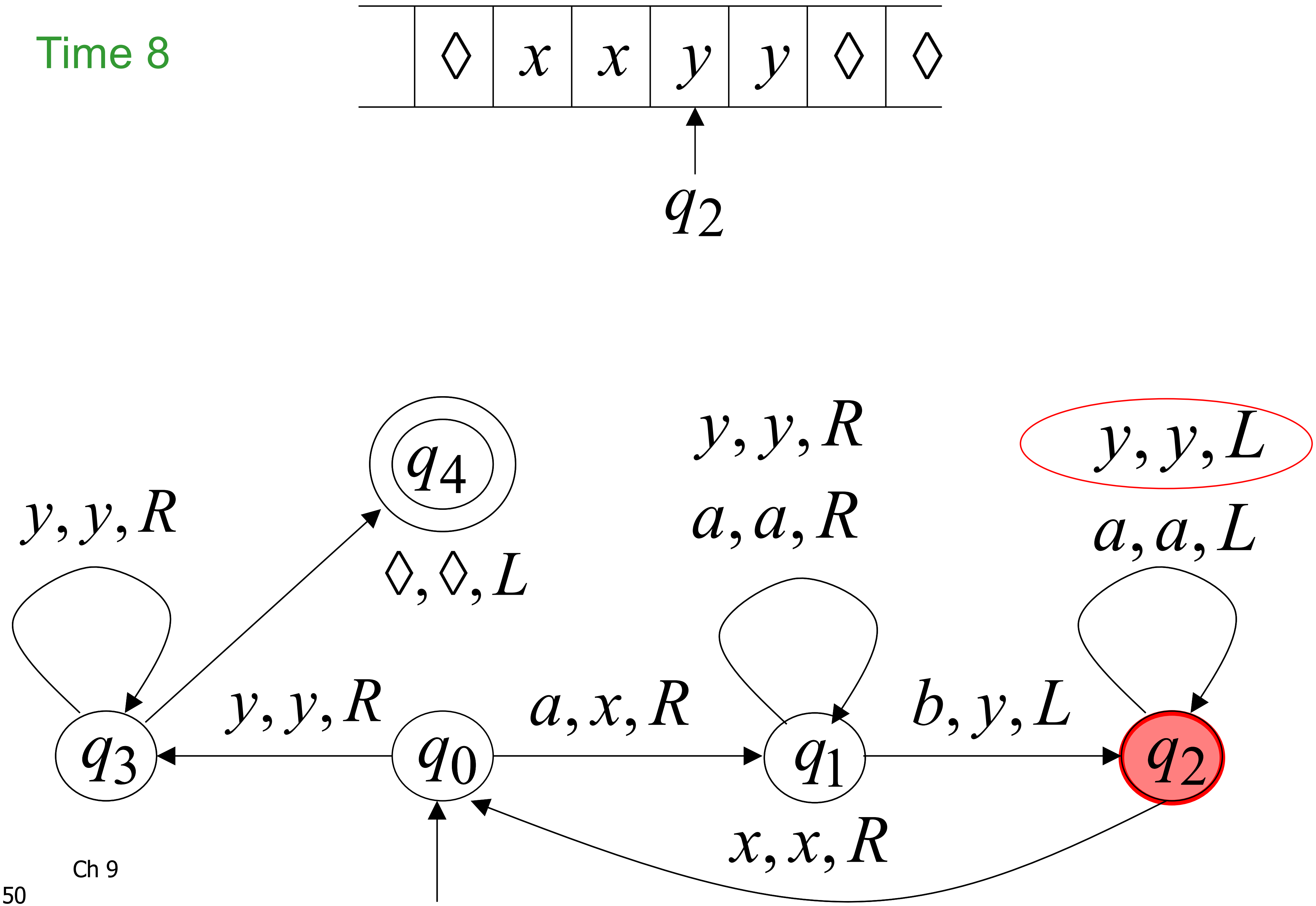

q_4
y, y, R
a, a, R
y, y, L
a, a, L
y, y, R
$\Diamond, \Diamond, L$
y, y, R
a, x, R
b, y, L
q_3
q_0
q_1
q_2
x, x, R

Time 7

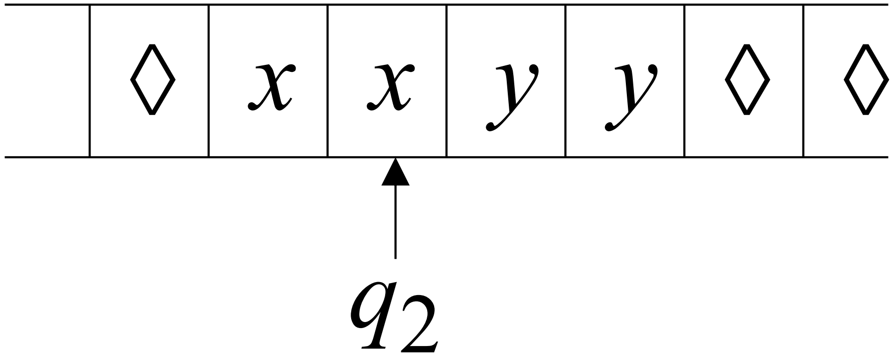

◊
x
x
y
b
◊
◊
q_1

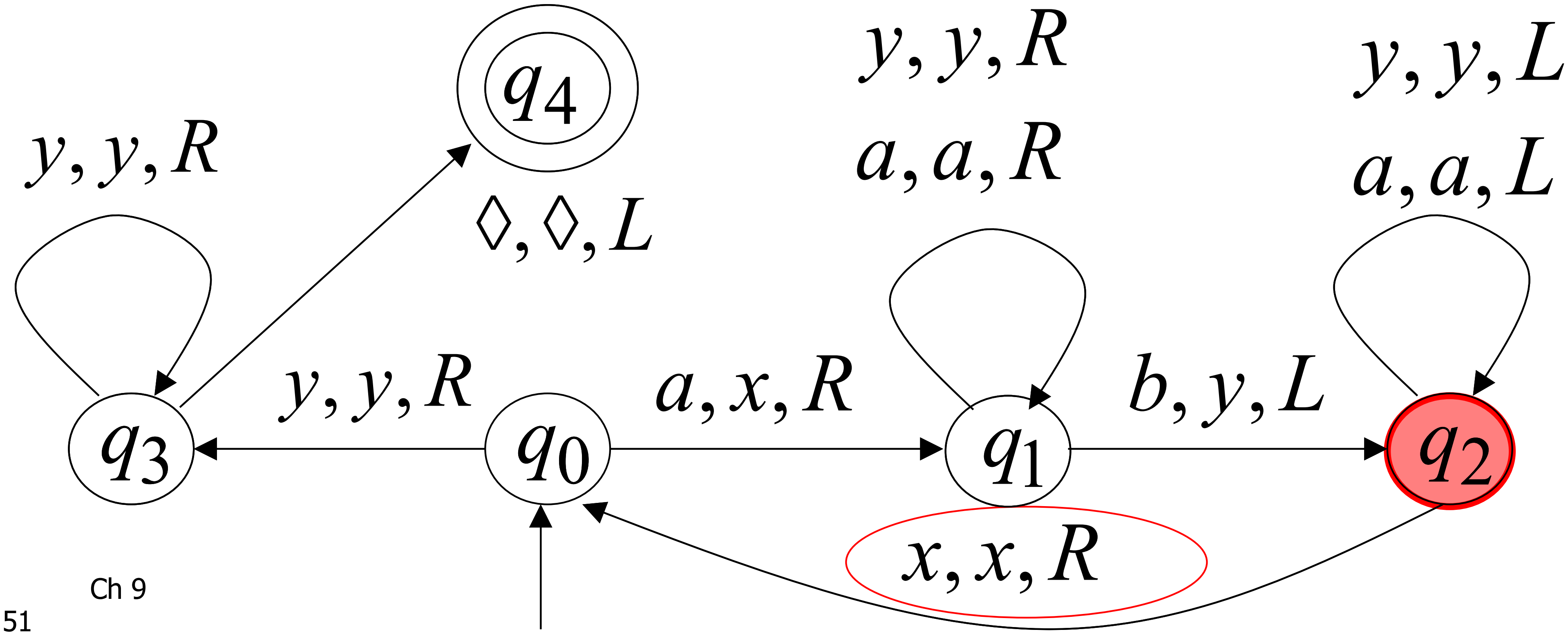

q_4
y, y, R
$\Diamond, \Diamond, L$
y, y, R
a, a, R
y, y, L
a, a, L
q_3
y, y, R
q_0
a, x, R
q_1
b, y, L
q_2
x, x, R

Time 8

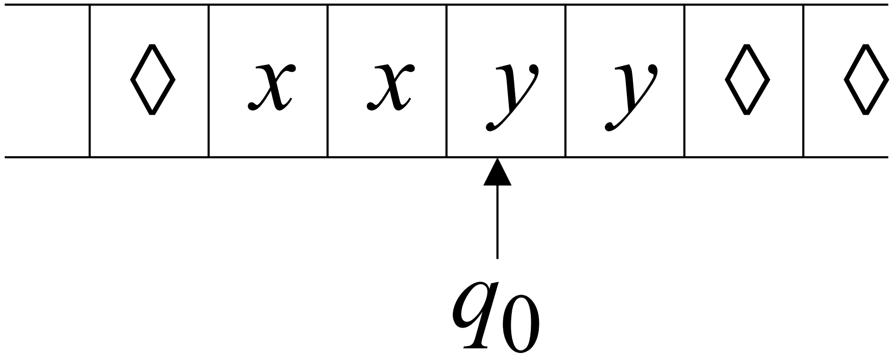


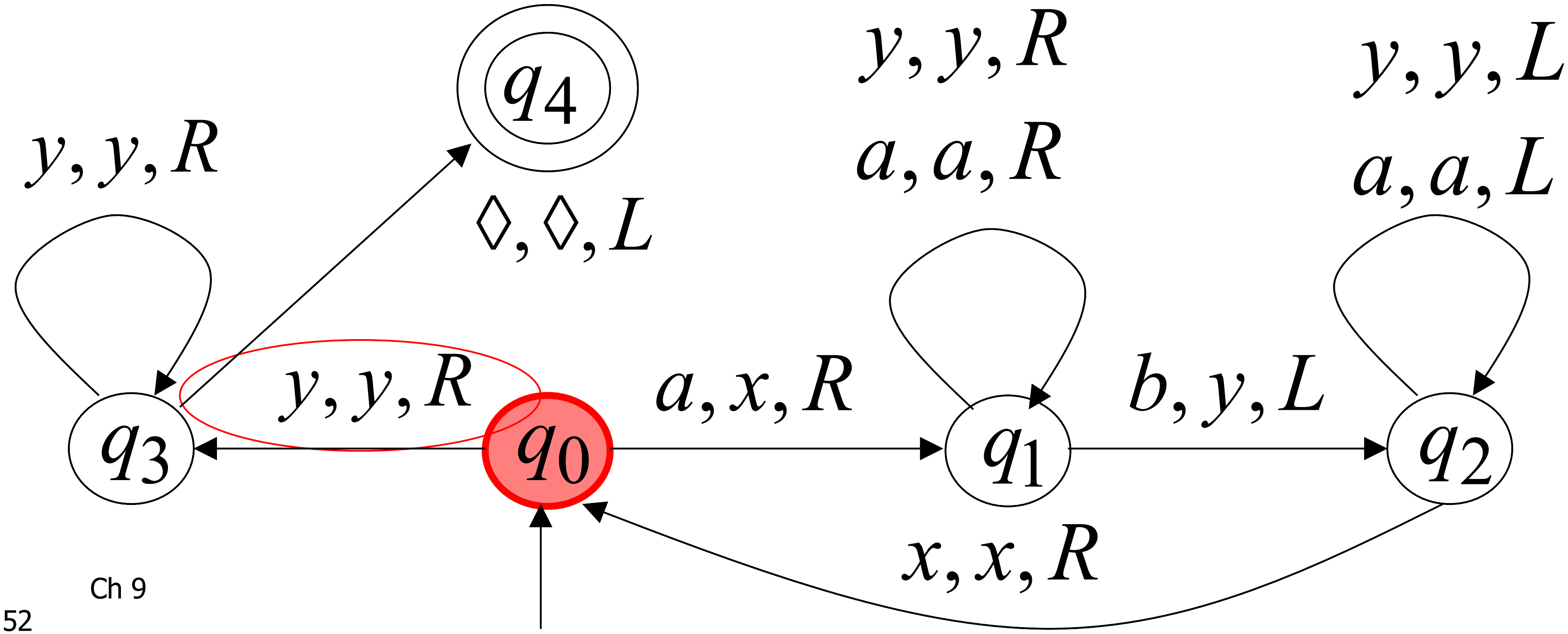
Time 9



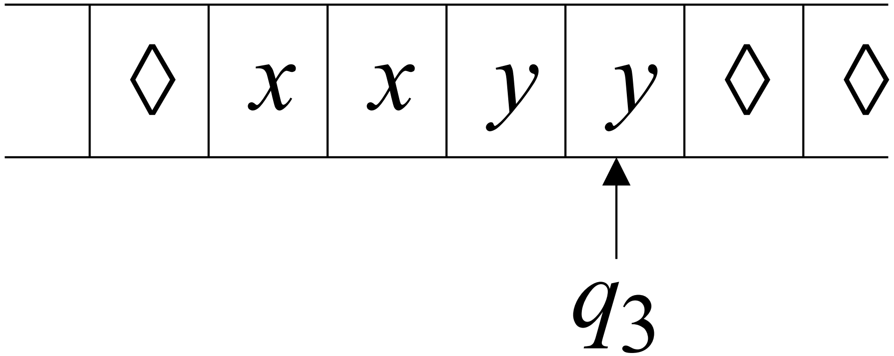


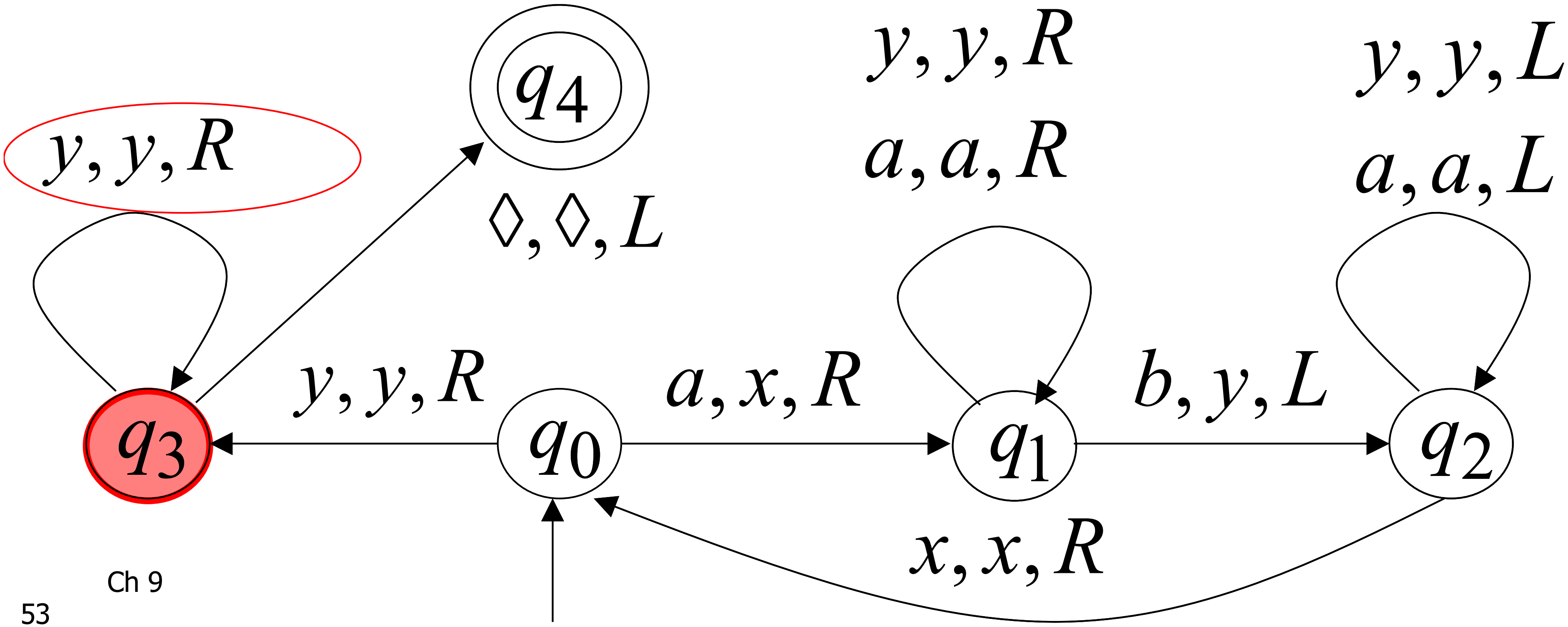
Time 10



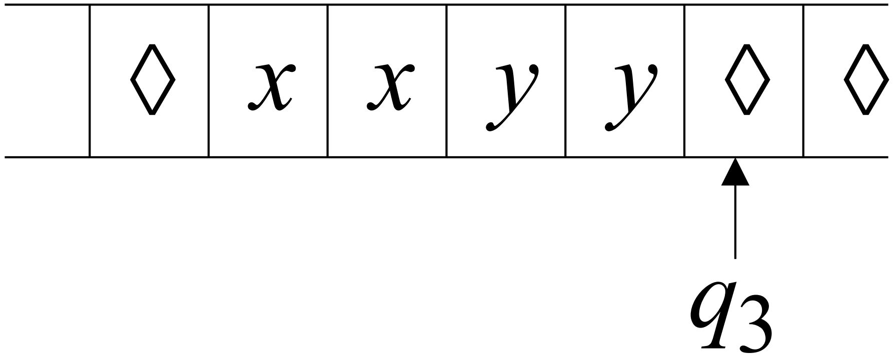


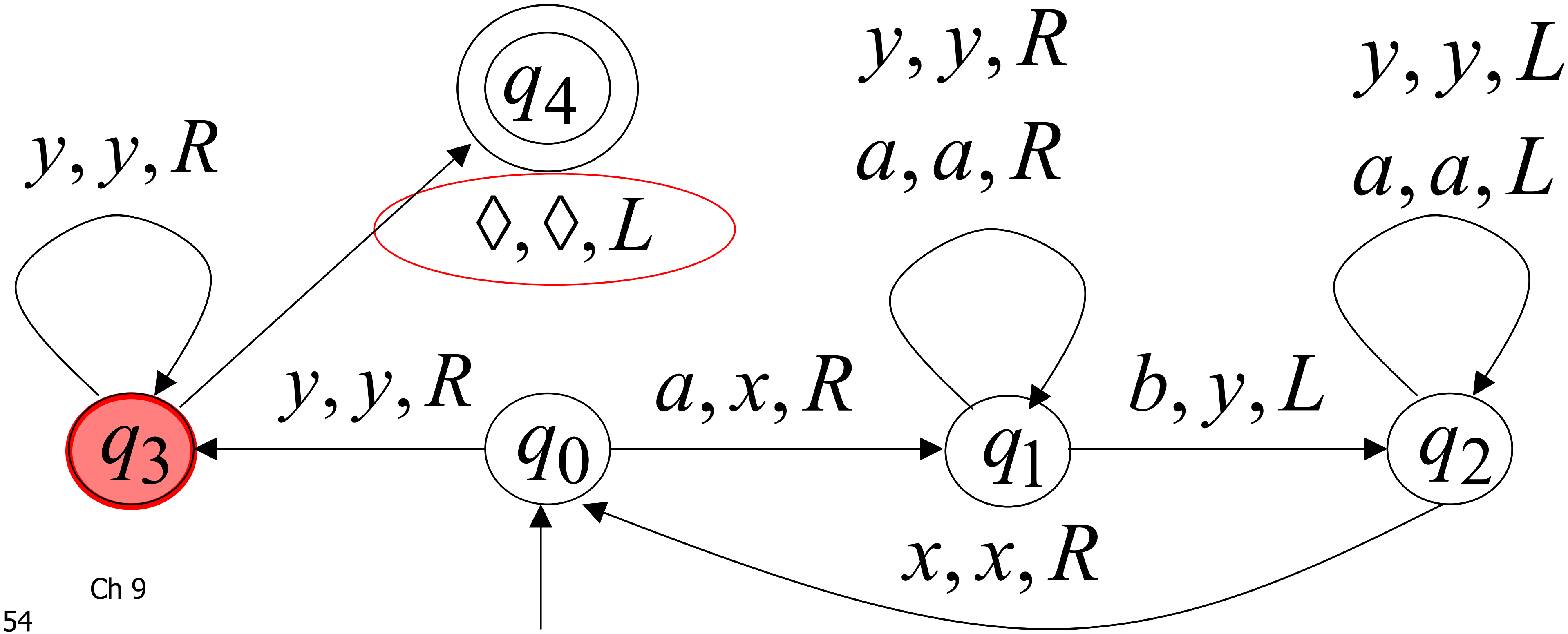
Time 11



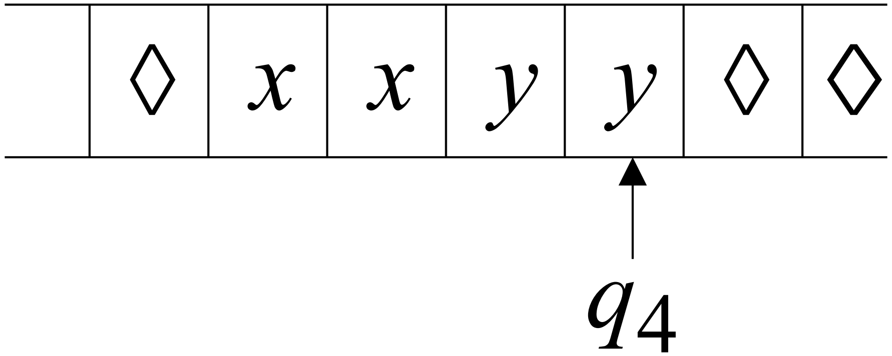


Time 12

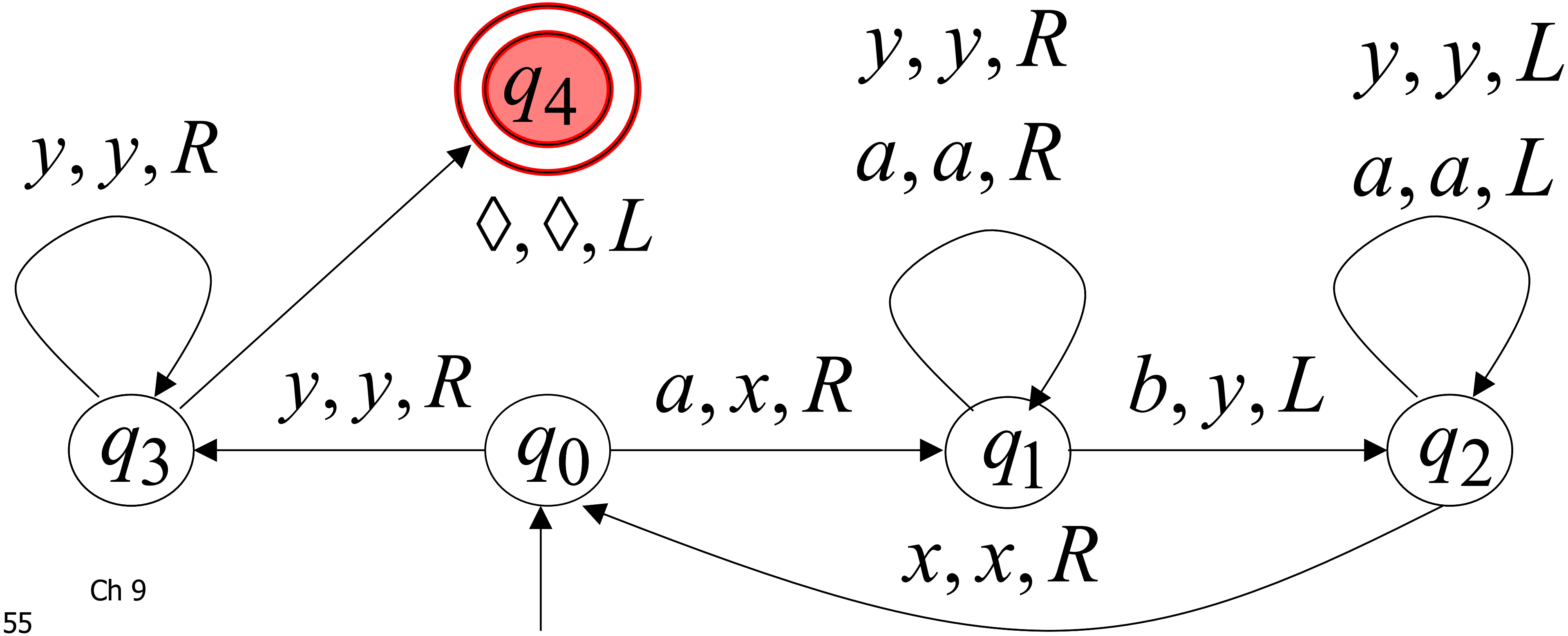




Time 13



Halt & Accept

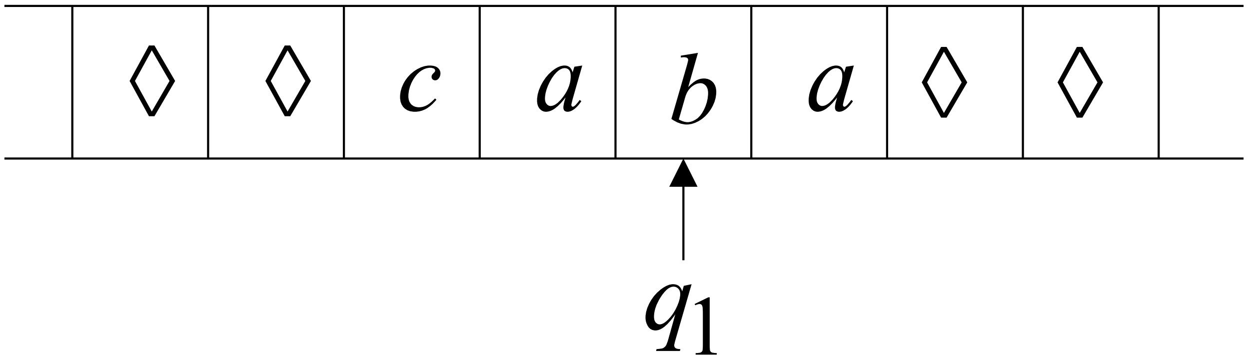


Observation:

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

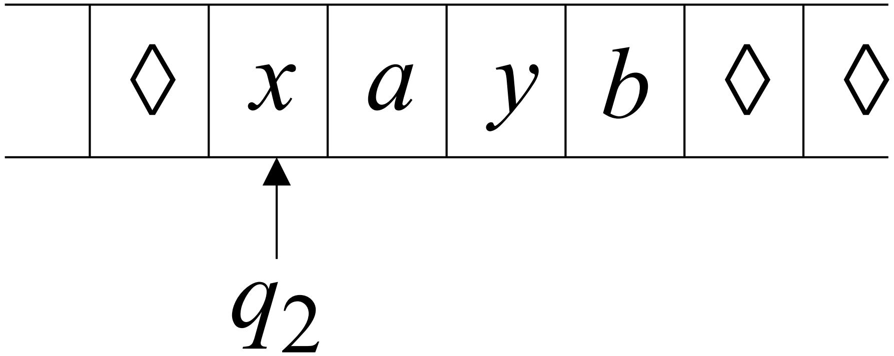
we can also construct
a machine for the language $\{a^n b^n c^n\}$

Configuration

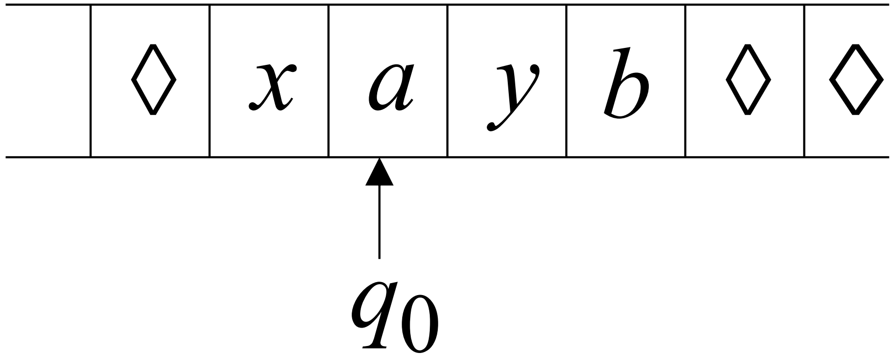


Instantaneous description: $ca\ q_1\ ba$

Time 4

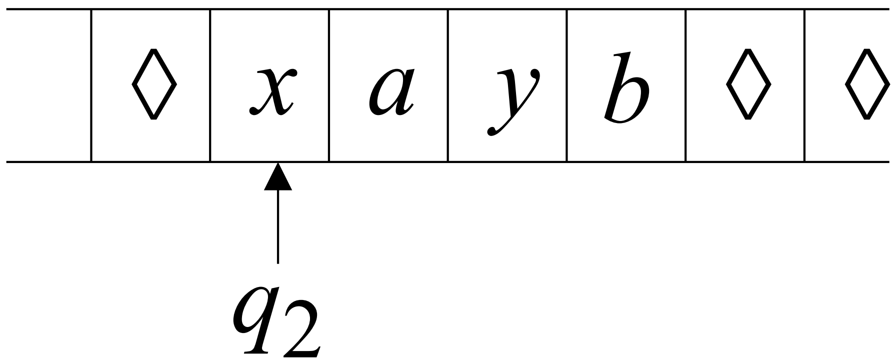


Time 5

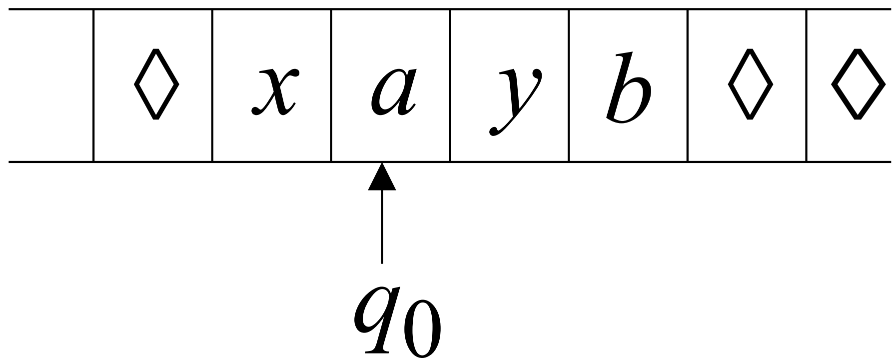


A Move: $q_2\ xayb \vdash x\ q_0\ ayb$

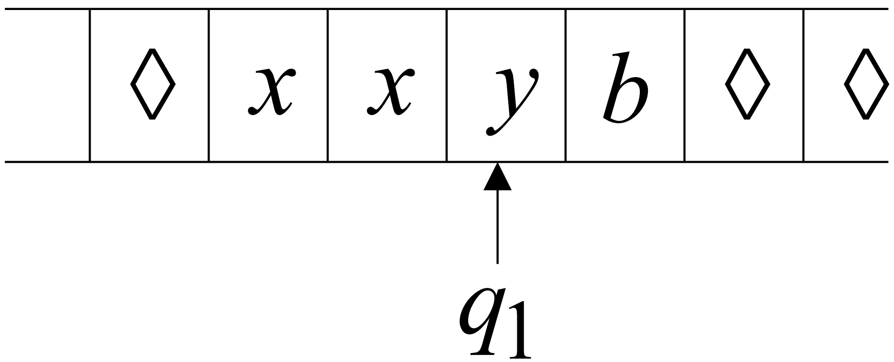
Time 4



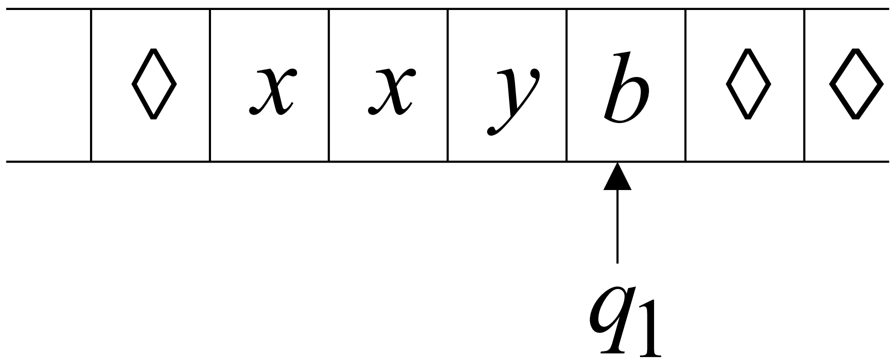
Time 5



Time 6



Time 7



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

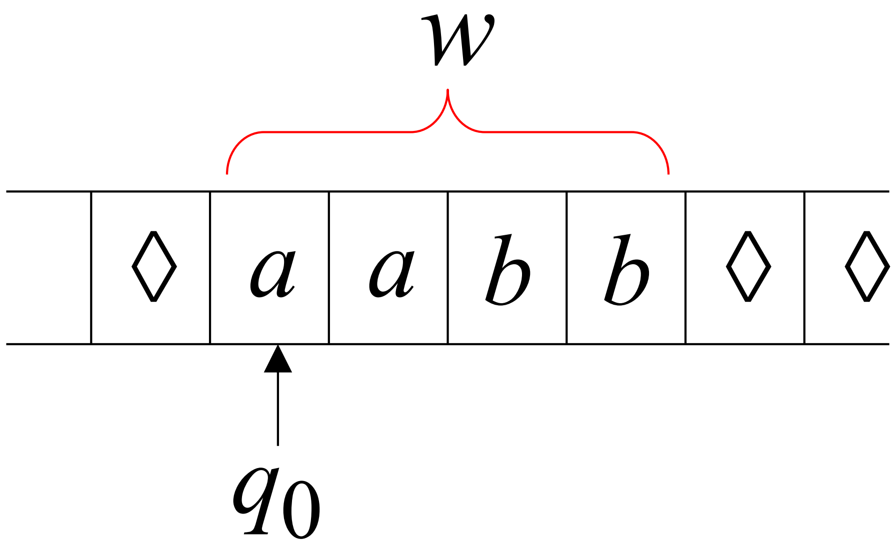
$$q_2\ xayb\ \succ\ x\ q_0\ ayb\ \succ\ xx\ q_1\ yb\ \succ\ xxy\ q_1\ b$$

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

Infinite loop (TM never halt): $$x_1 q x_2\ \overset{*}{\succ}\ \infty$$

Initial configuration: $q_0 \; w$

Input string



The Accepted Language

For any Turing Machine M

$$L(M) = \{w : \quad q_0\, w \overset{*}{\vdash} x_1\, q_f\, x_2\}$$

Initial state

Final state

The sequence of configurations leading to a halt state will be called a **computation**.

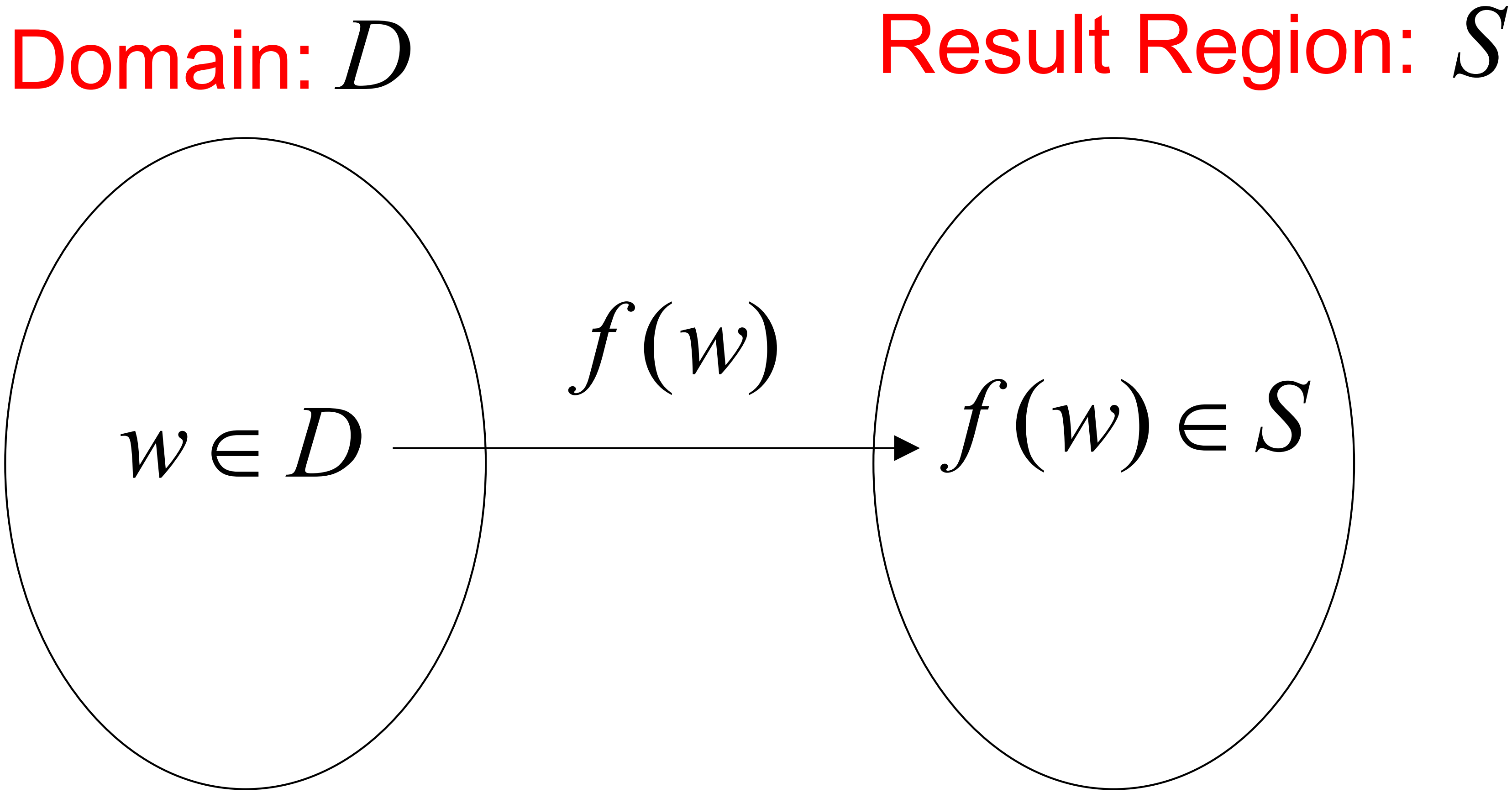
Standard Turing Machine

Main features of the standard Turing machine:

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

Computing Functions with Turing Machines

A function $f(w)$ has:



“Transducer”

A function may have many parameters:

Example: Addition function

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

Integer Domain

Decimal: 5

Binary: 101

Unary: 11111

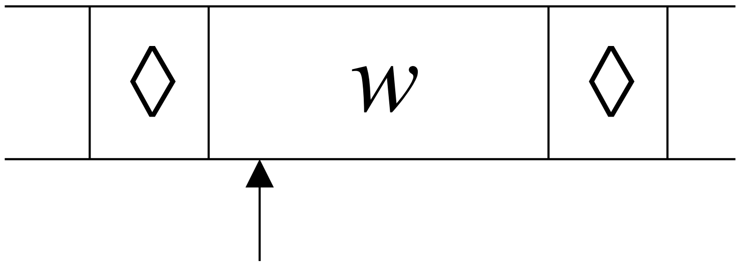
We prefer **unary** representation:

$\because$ easier to manipulate with Turing machines

Definition:

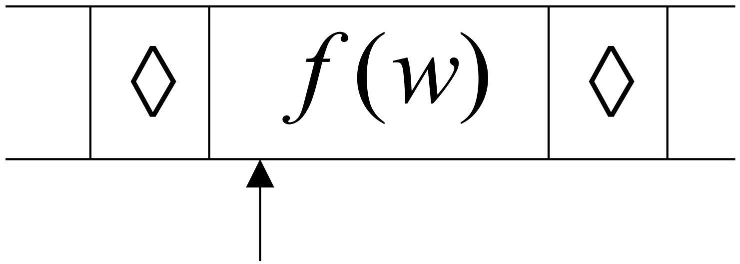
A function f is computable (Turing Computable) if there is a Turing Machine M such that:

Initial configuration



q_0 initial state

Final configuration



q_f final state

For all $w \in D$ Domain

In other words:

A function f is computable (Turing Computable) if there is a Turing Machine M such that:

$$q_0 \, w \ \overset{*}{\vdash}_{M} \ q_f \, f(w)$$

Initial Configuration

Final Configuration

For all $w \in D$ Domain

Example 9.9

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

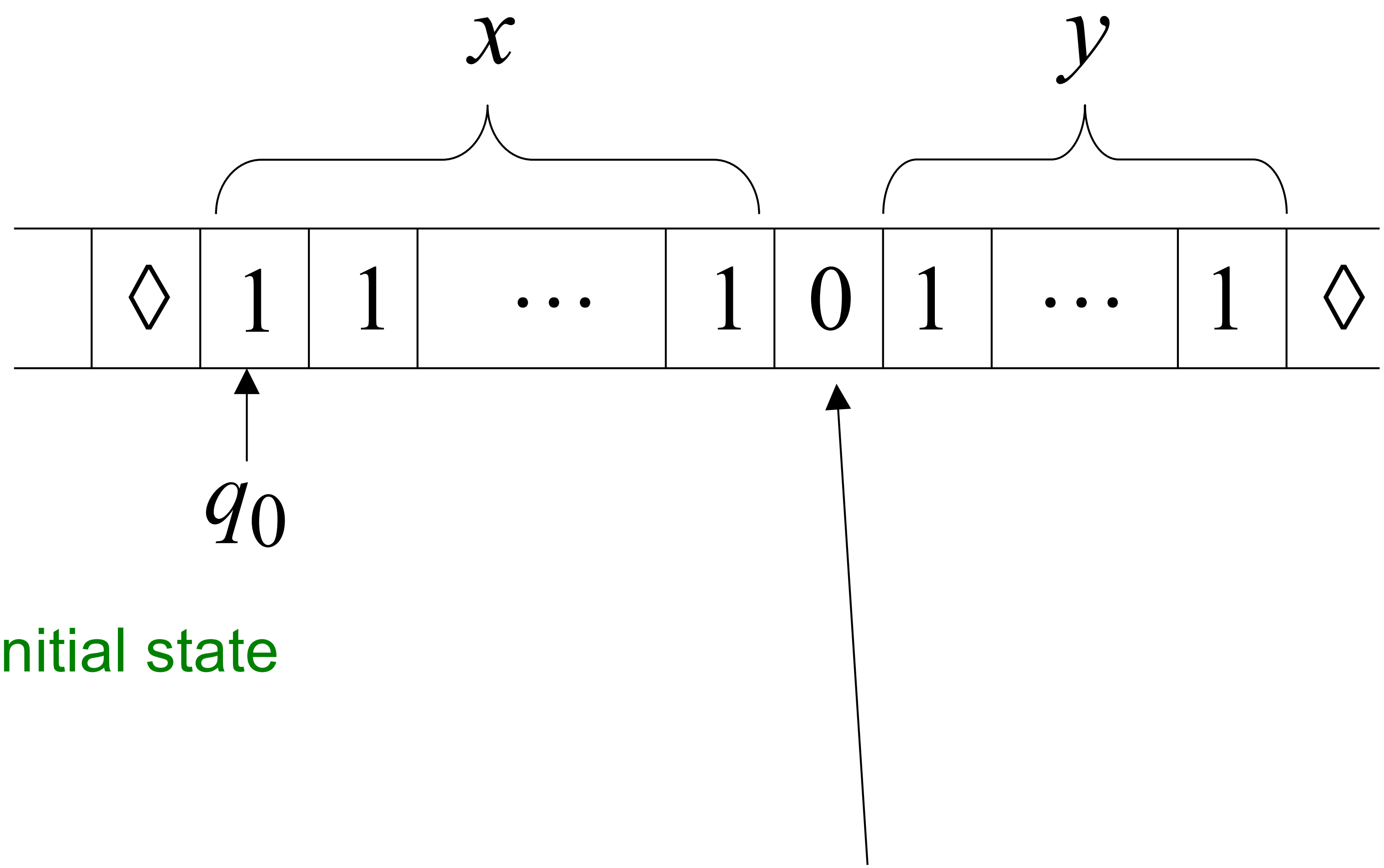
x, y are integers

Turing Machine:

Input string: $x0y$ unary

Output string: $xy0$ unary

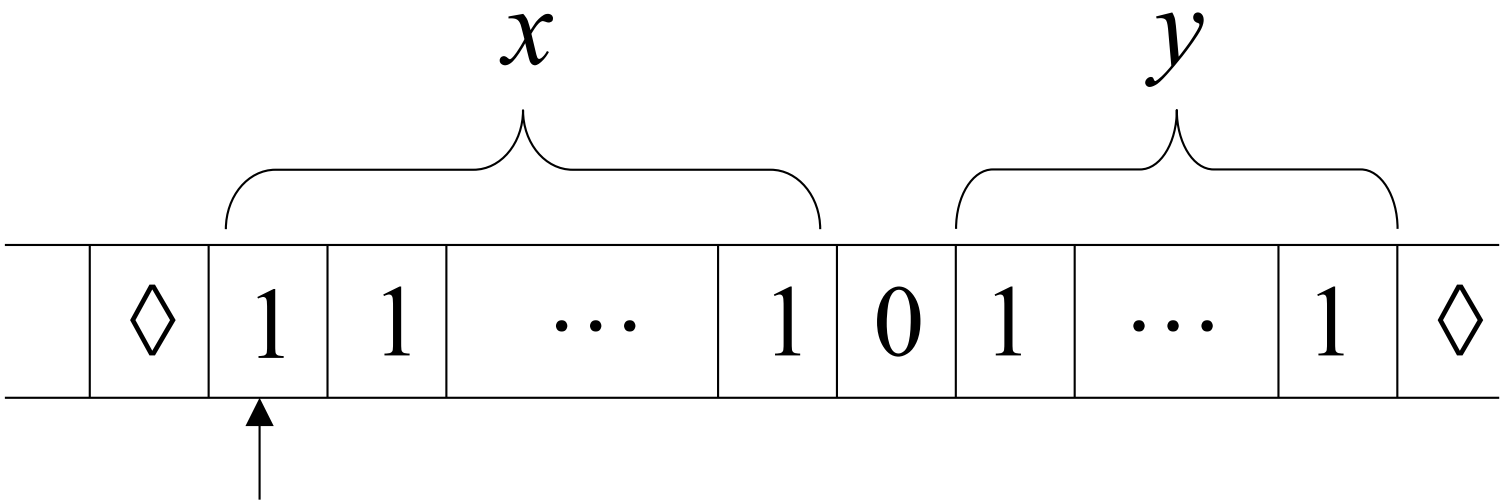
Start



initial state

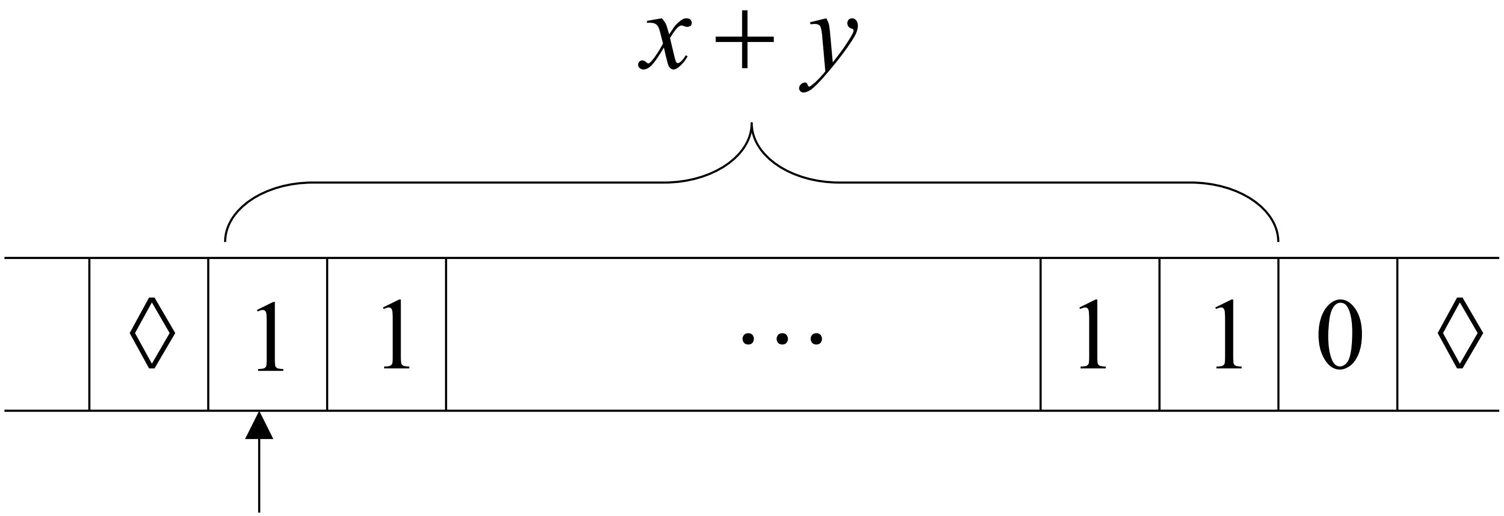
The 0 is the delimiter that separates the two numbers

Start



q_0 initial state

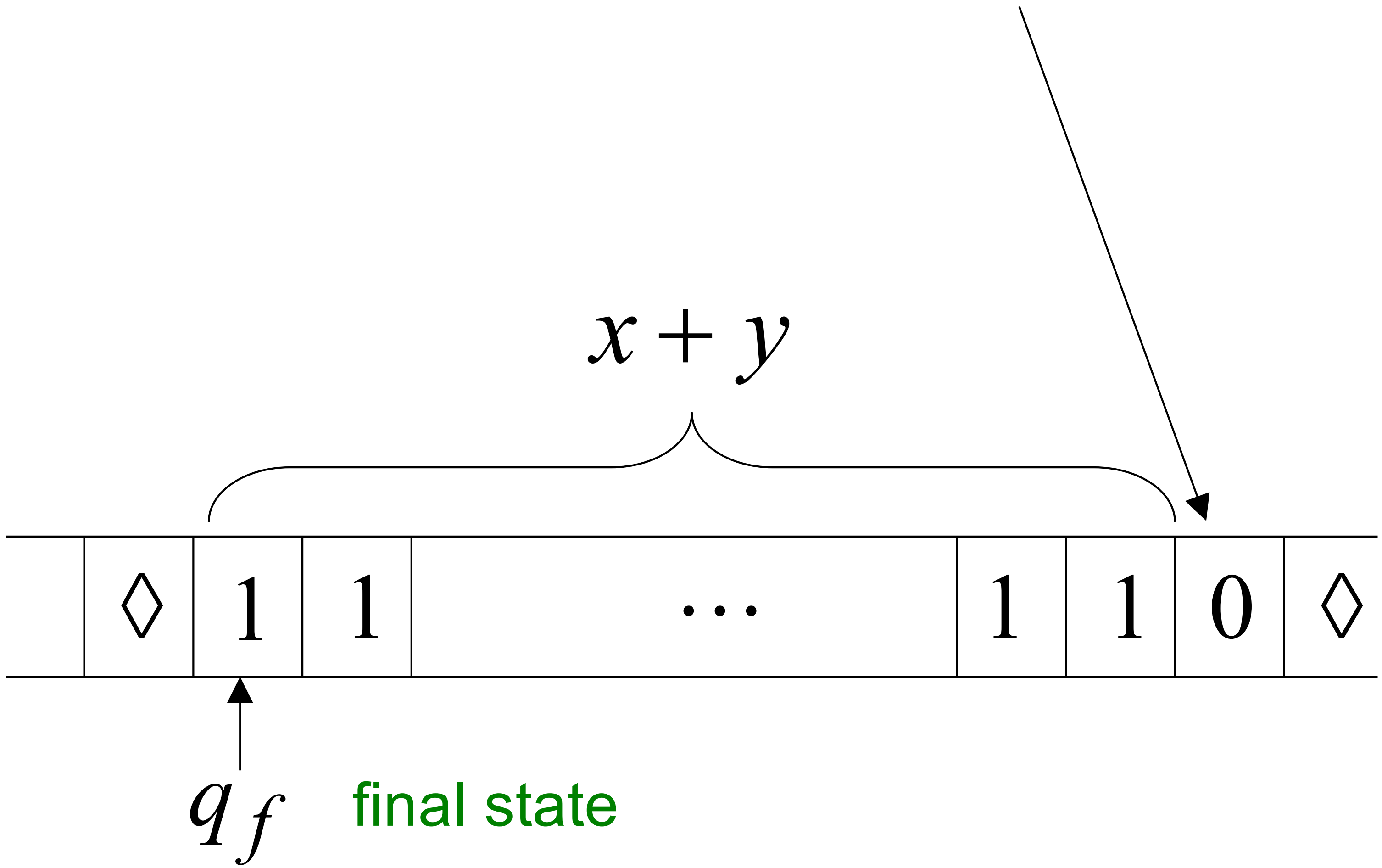
Finish



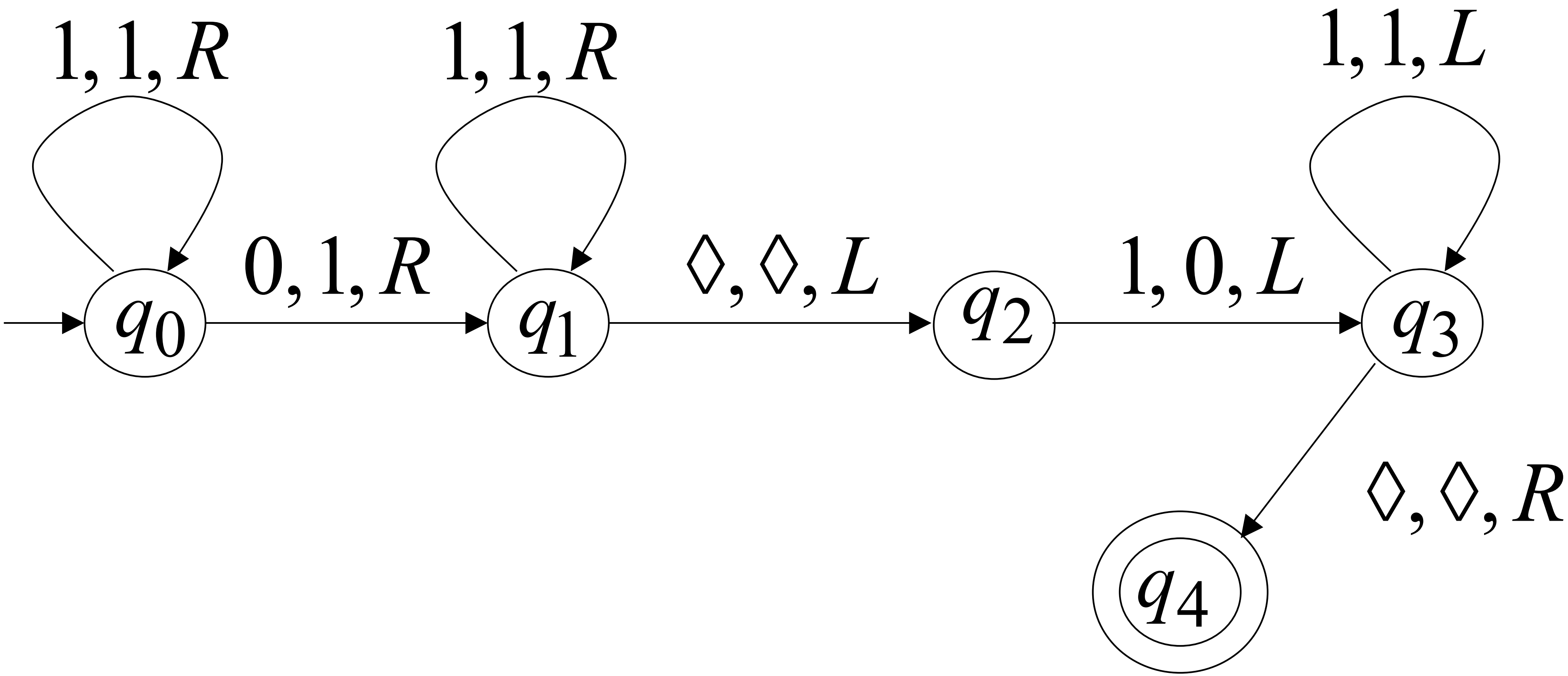
q_f final state

The 0 helps when we use
the result for other operations

Finish



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

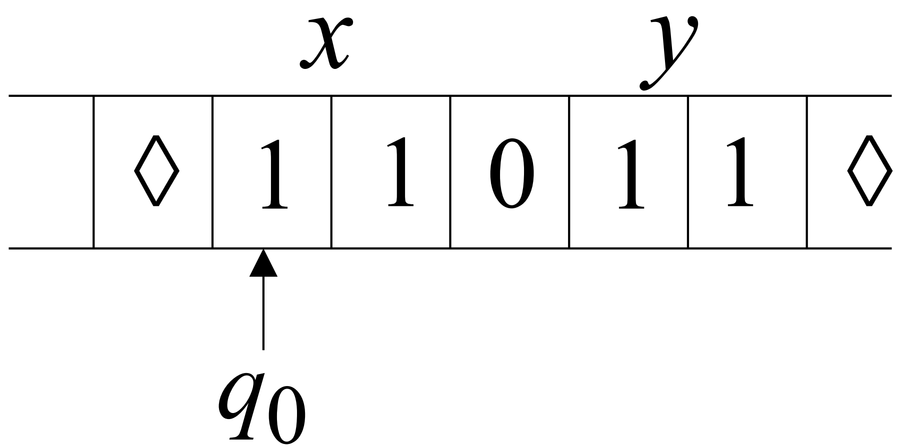


Execution Example:

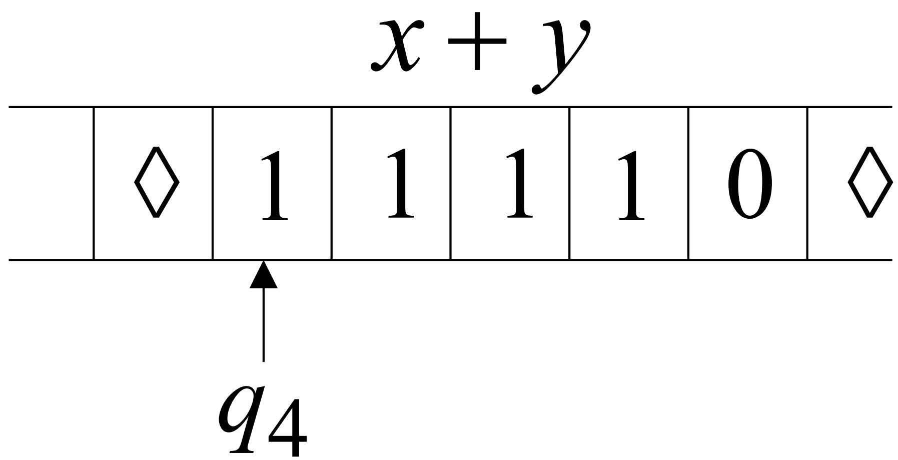
$x = 11 \quad {}_{(2)}$

$y = 11 \quad {}_{(2)}$

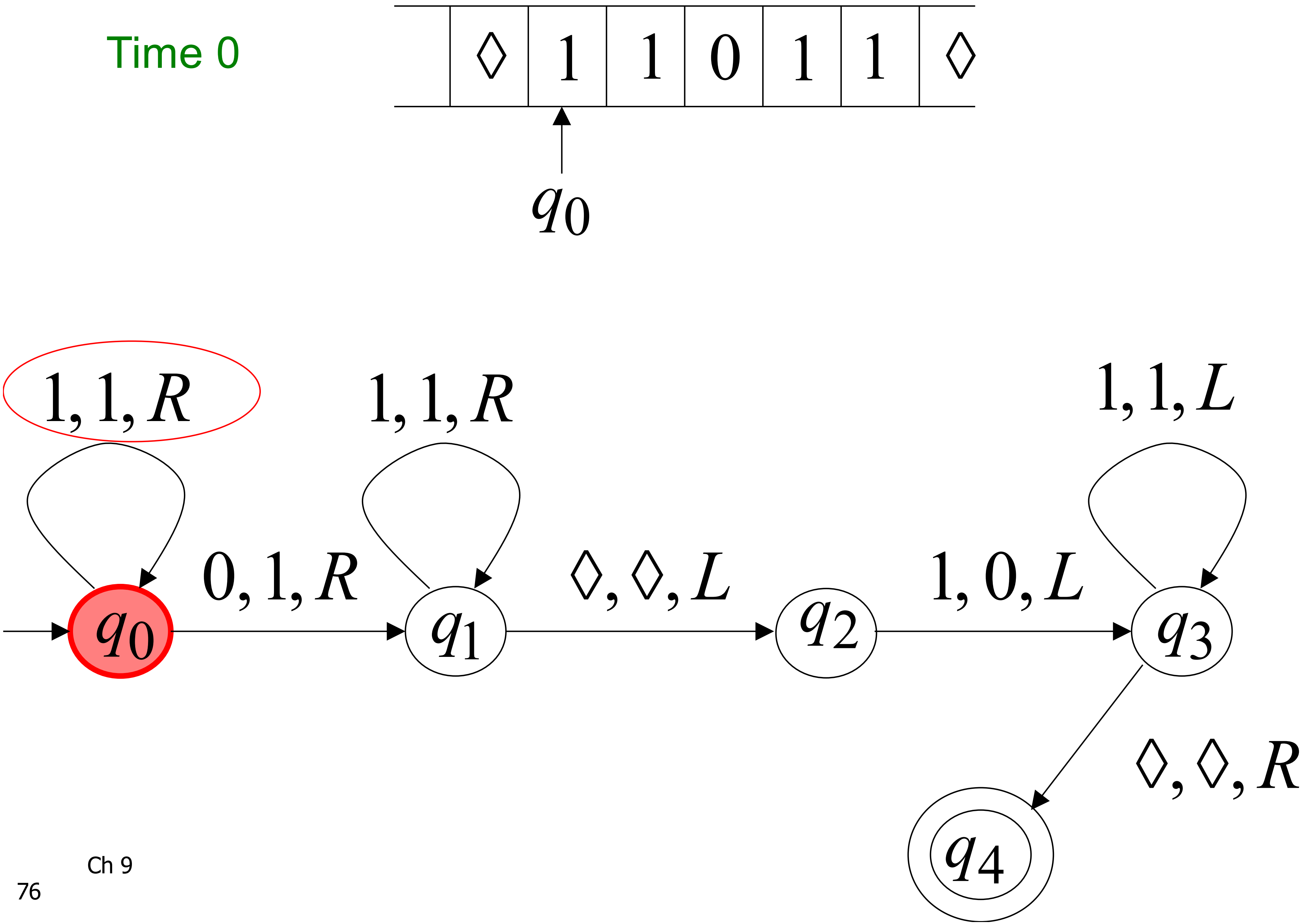
Time 0



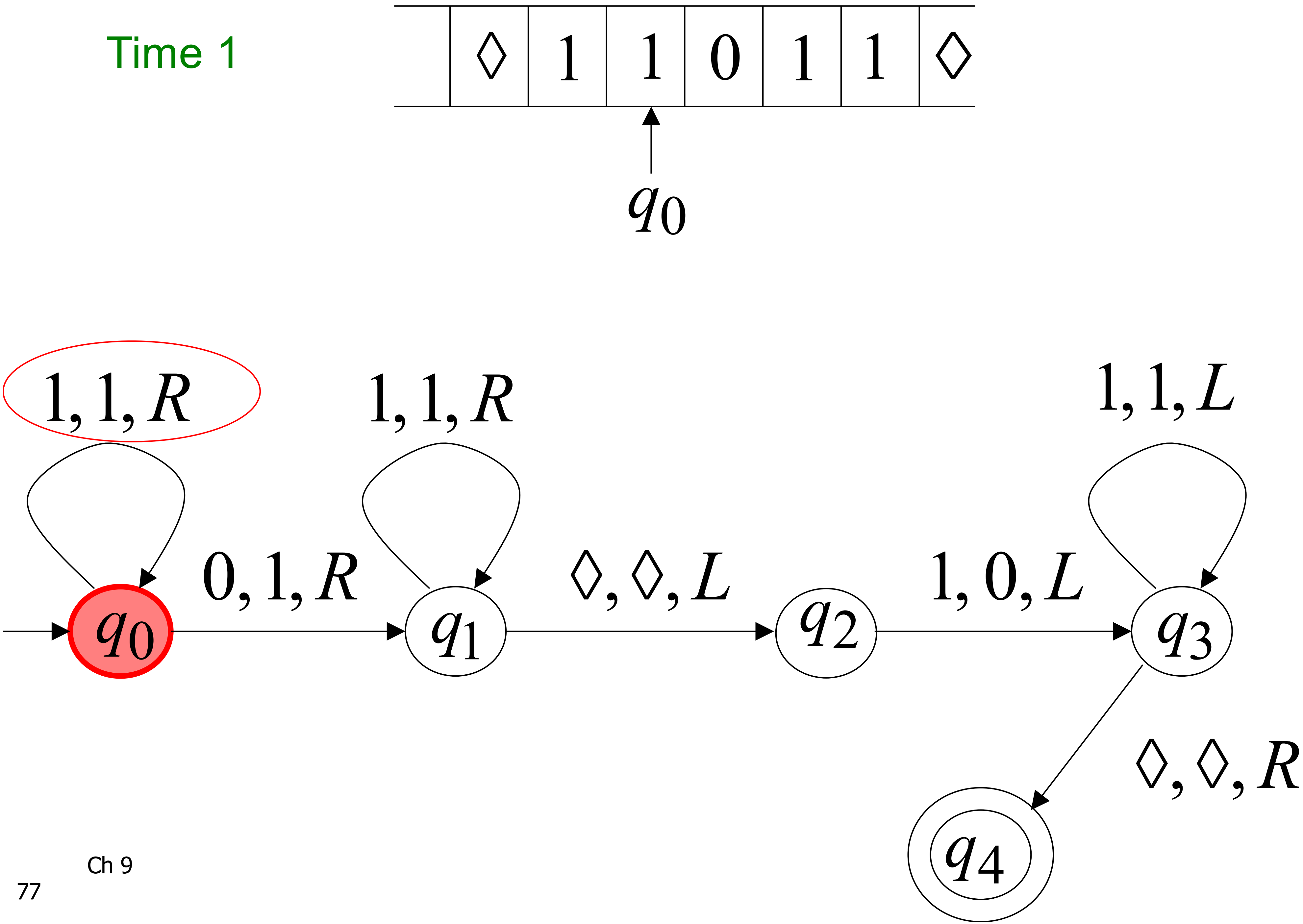
Final Result



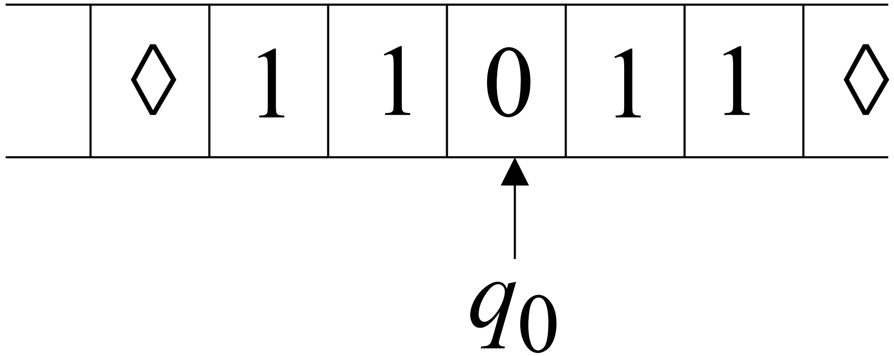
Time 0

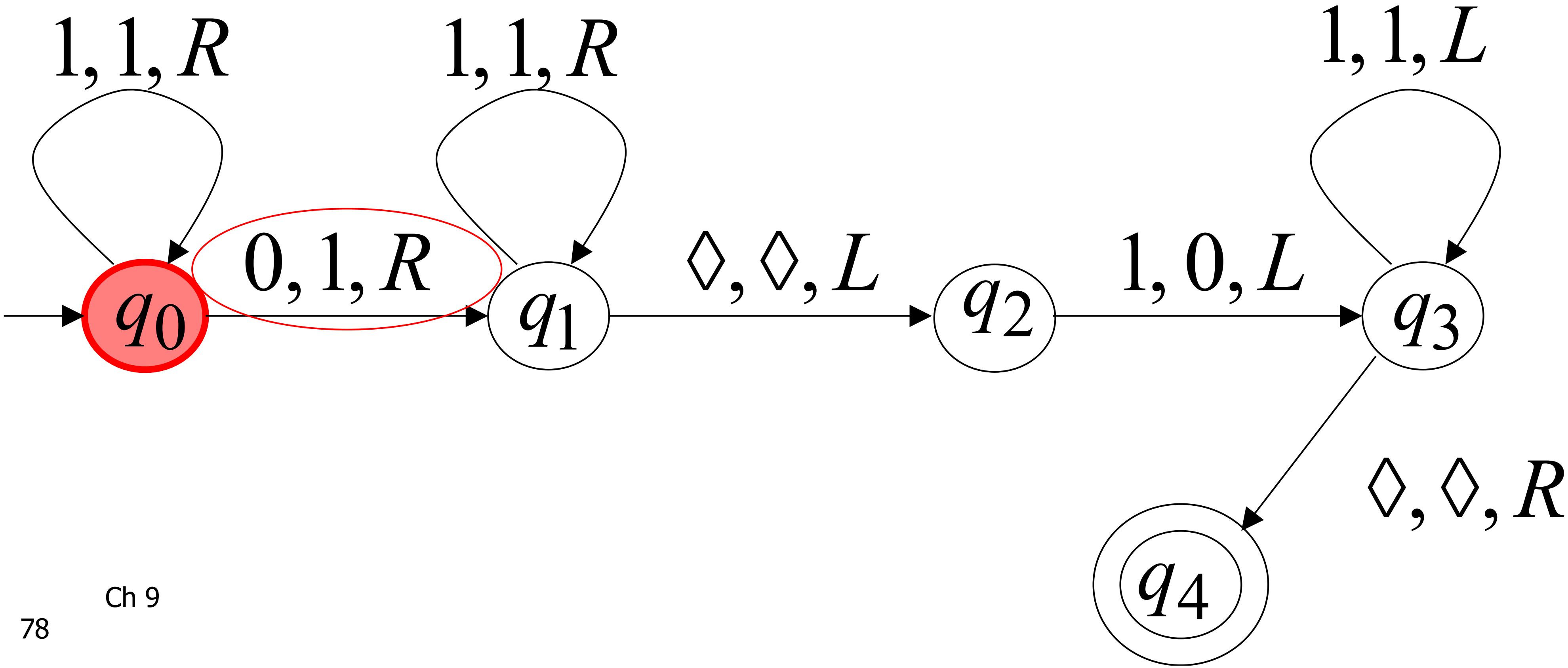


Time 1

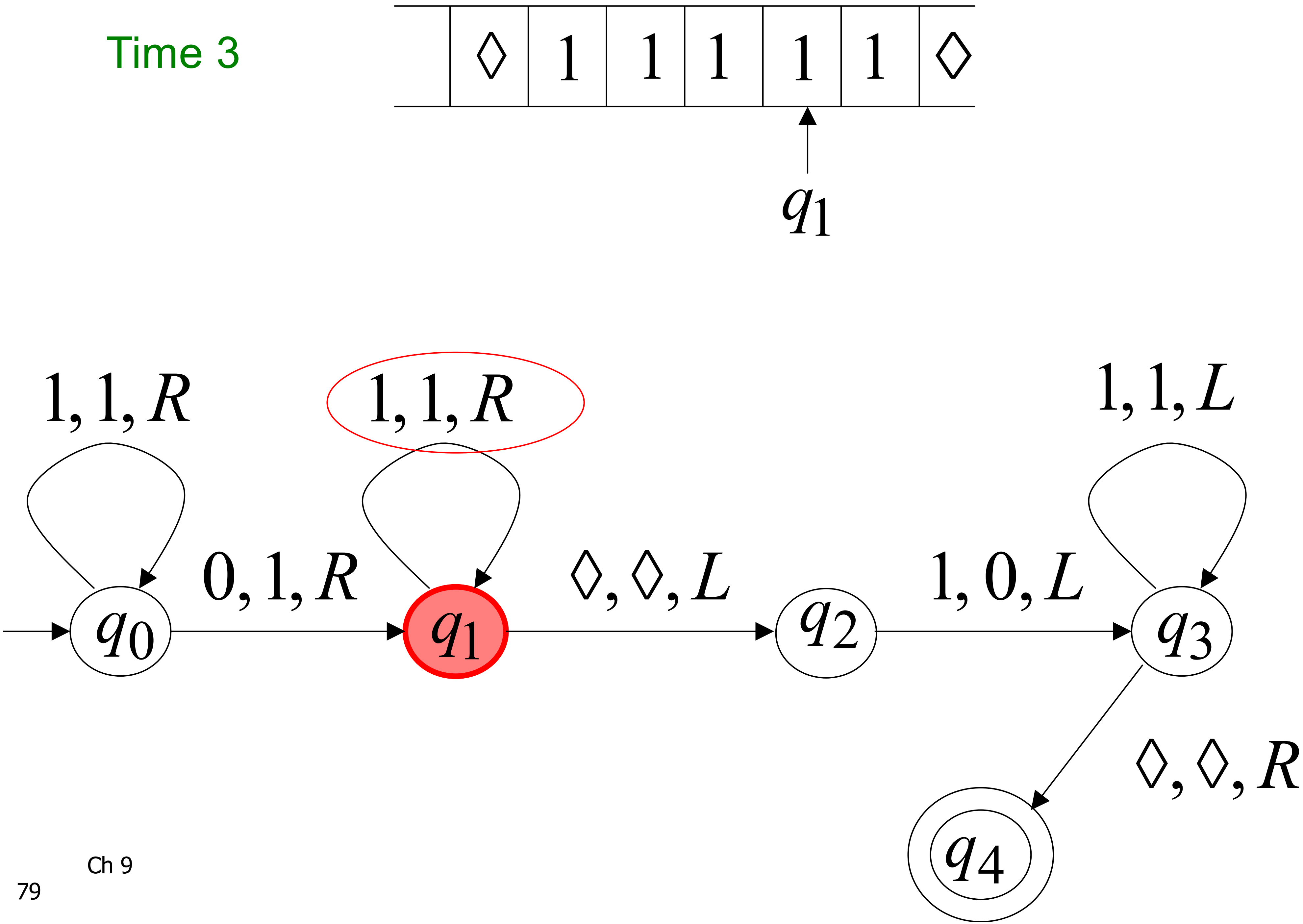


Time 2

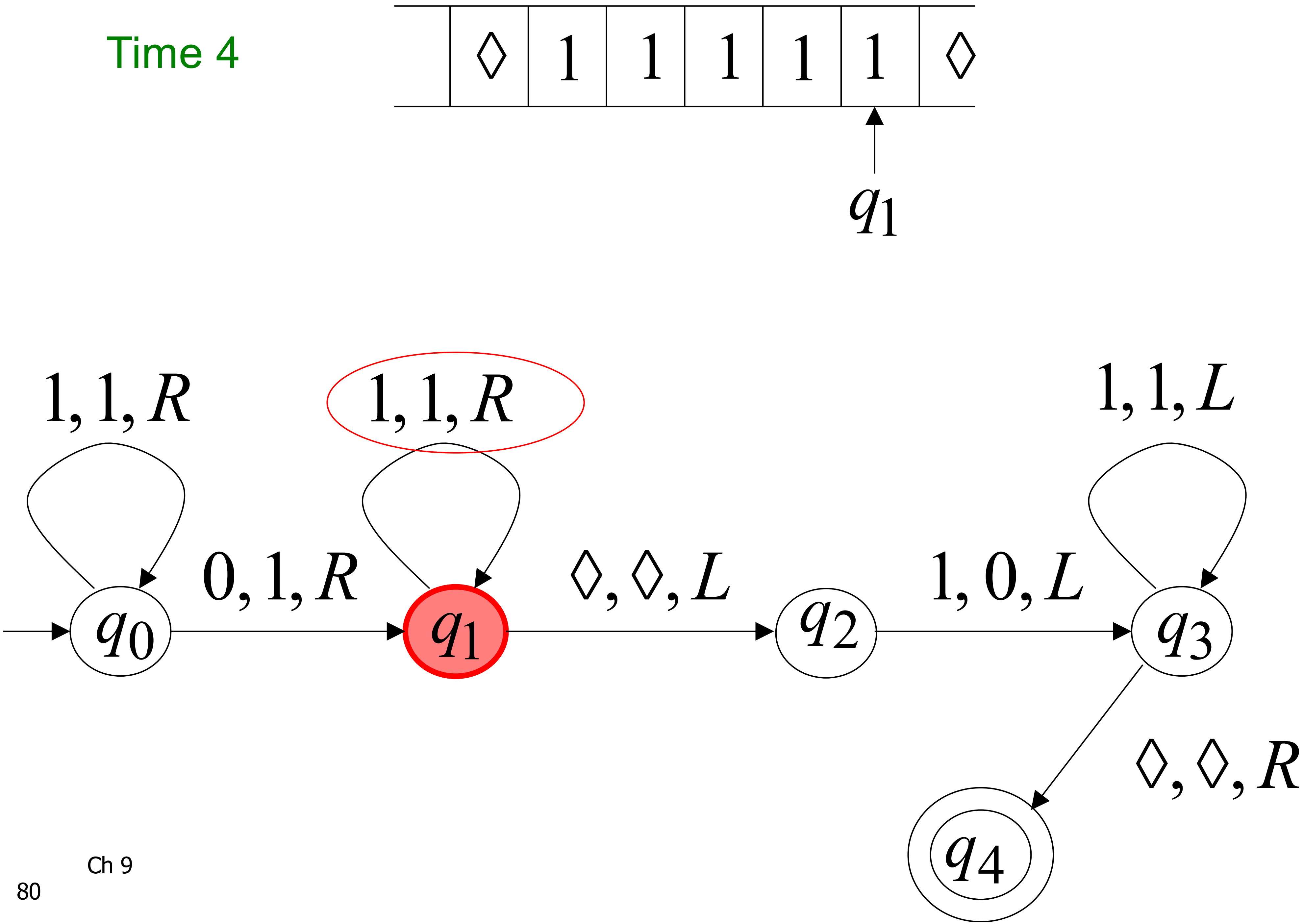




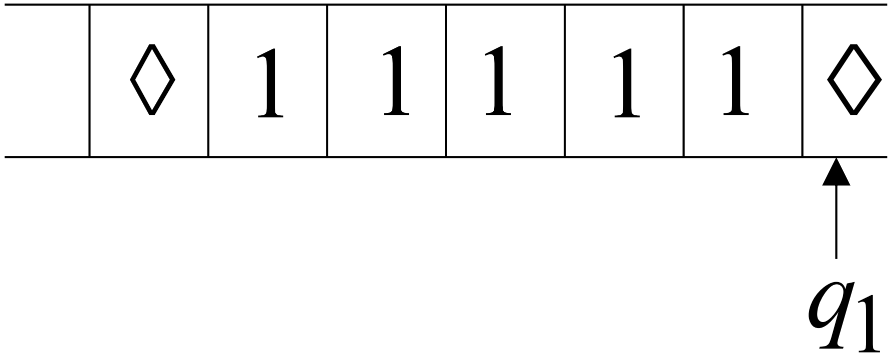
Time 3

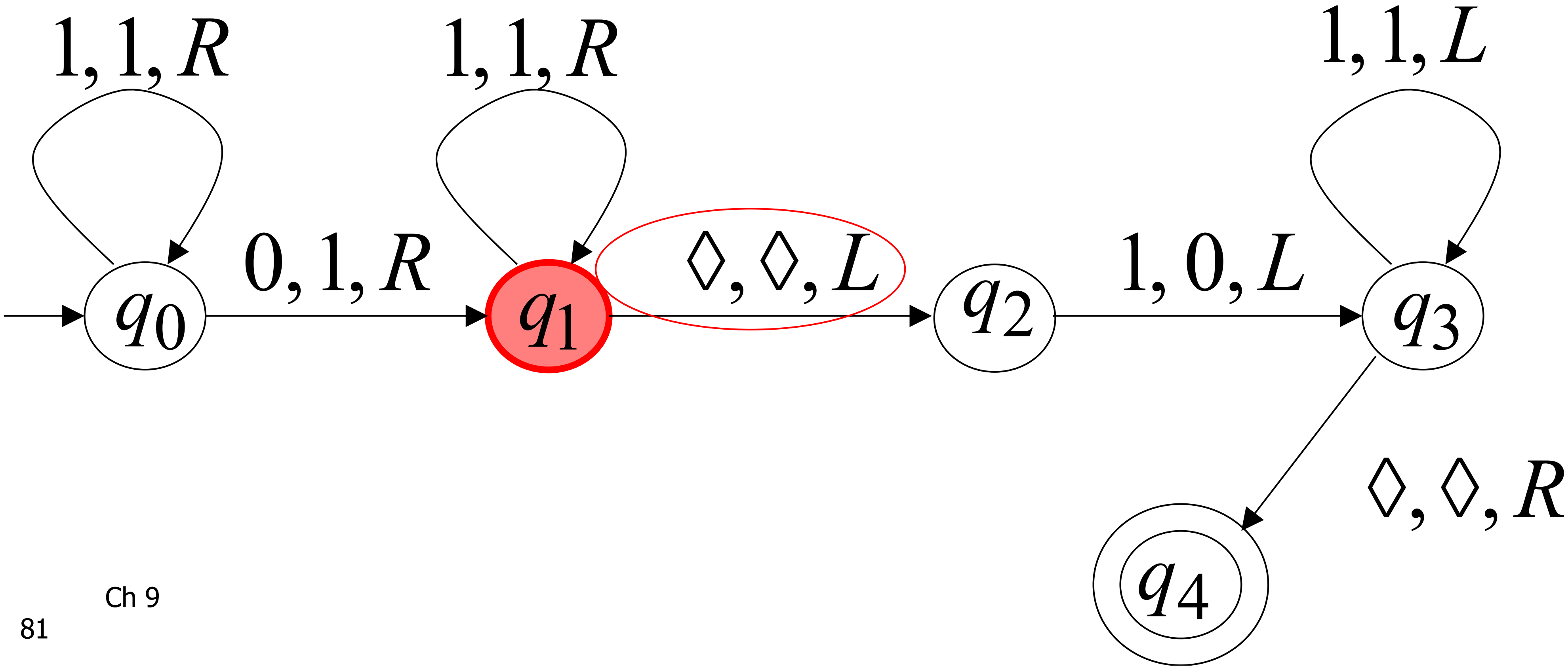


Time 4

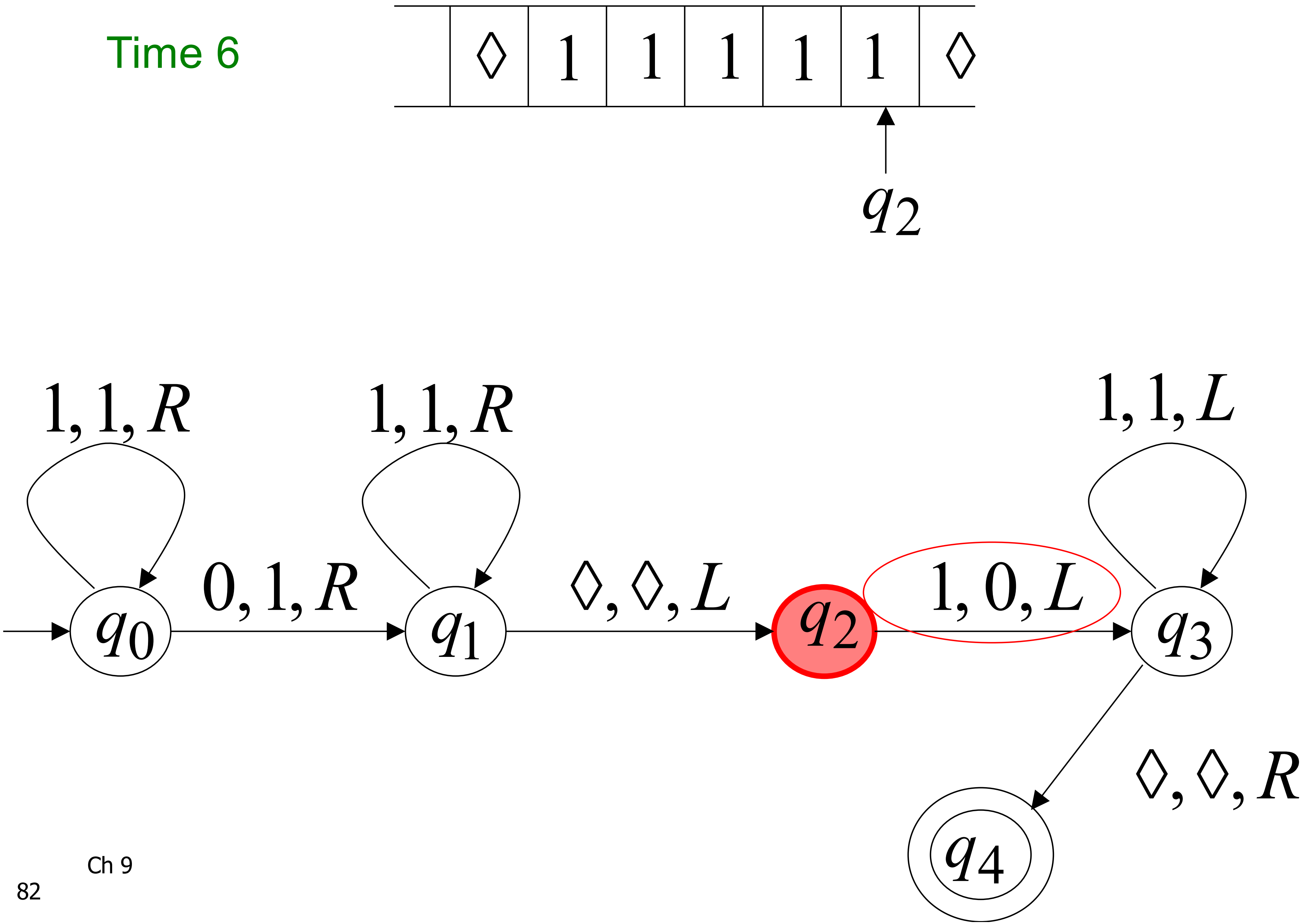


Time 5

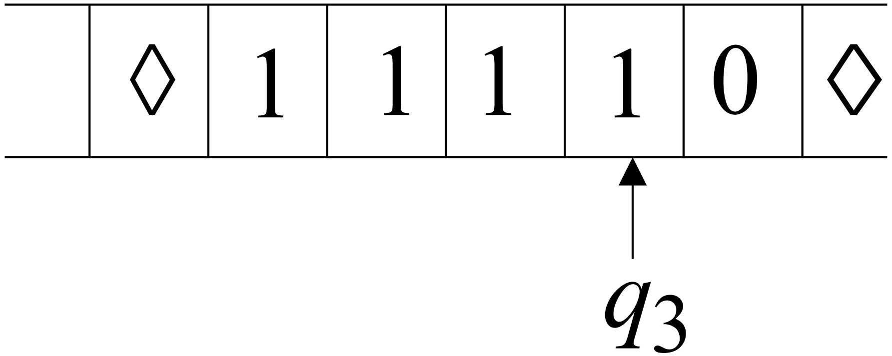

◊ 1 1 1 1 1 ◊
q_1


$1, 1, R$
$1, 1, R$
$1, 1, L$
$0, 1, R$
$\Diamond, \Diamond, L$
$1, 0, L$
q_0
q_1
q_2
q_3
$\Diamond, \Diamond, R$
q_4

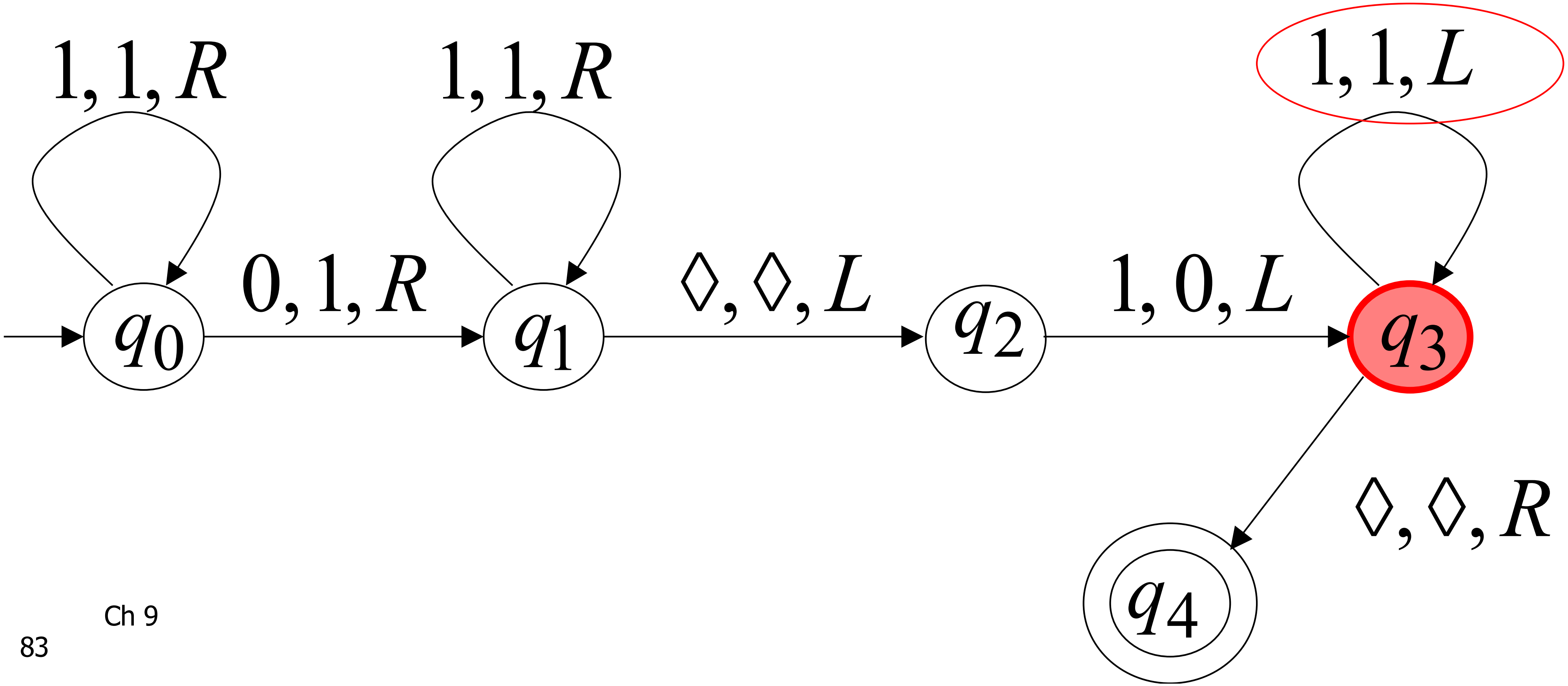
Time 6



Time 7

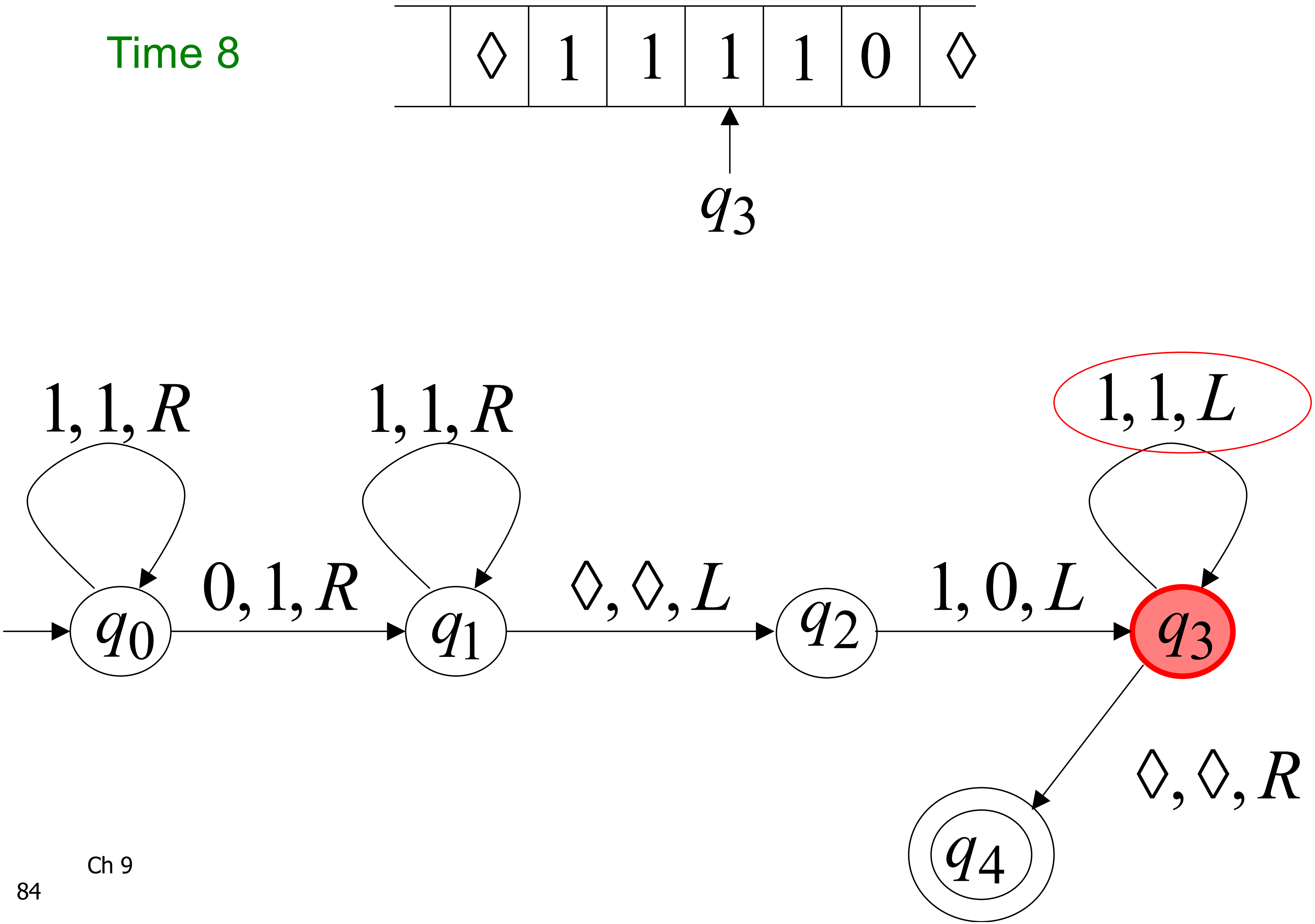


◊
1
1
1
1
0
◊
q_3



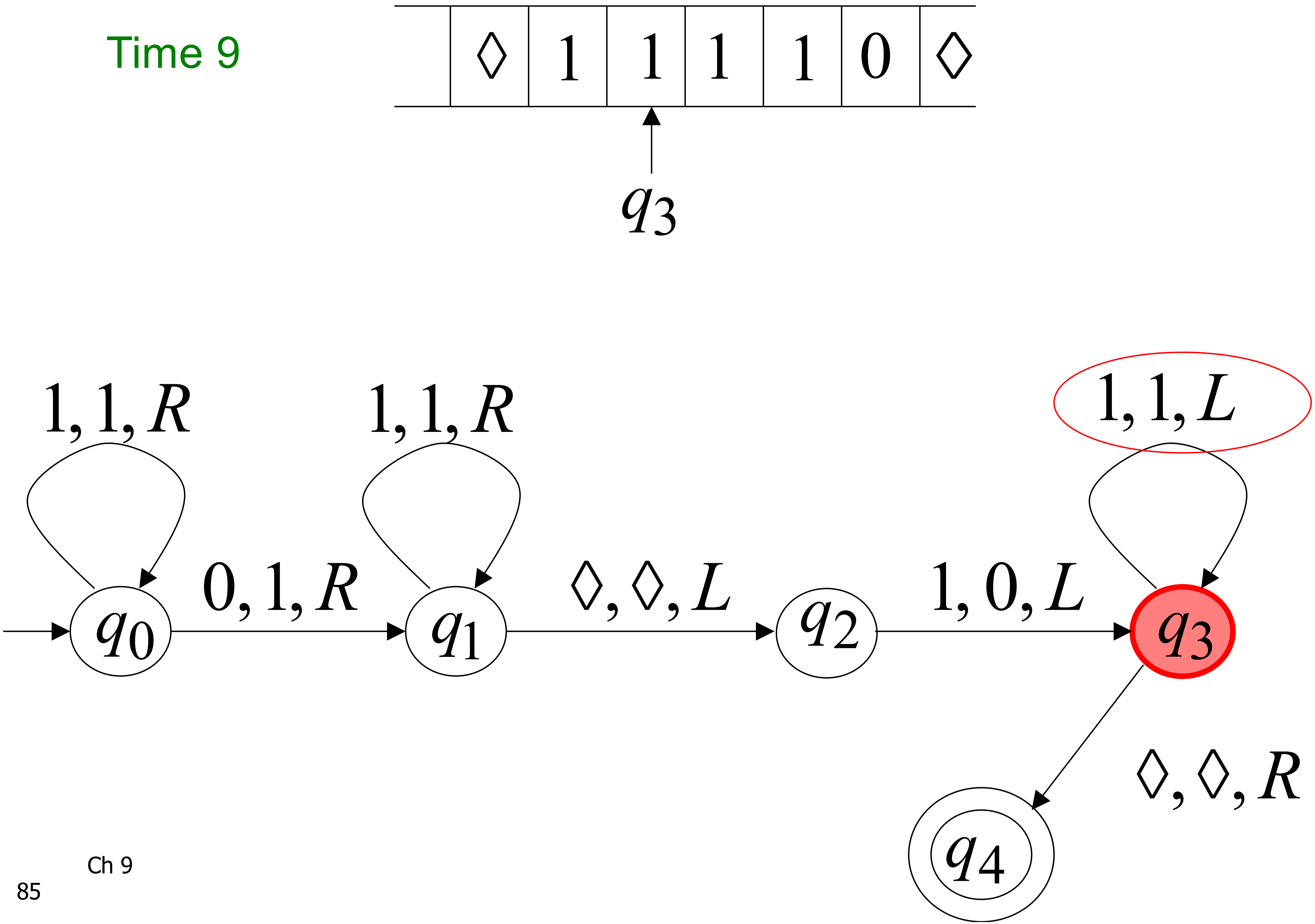
$1, 1, R$
$1, 1, R$
$1, 1, L$
$0, 1, R$
$\Diamond, \Diamond, L$
$1, 0, L$
q_0
q_1
q_2
q_3
$\Diamond, \Diamond, R$
q_4

Time 8

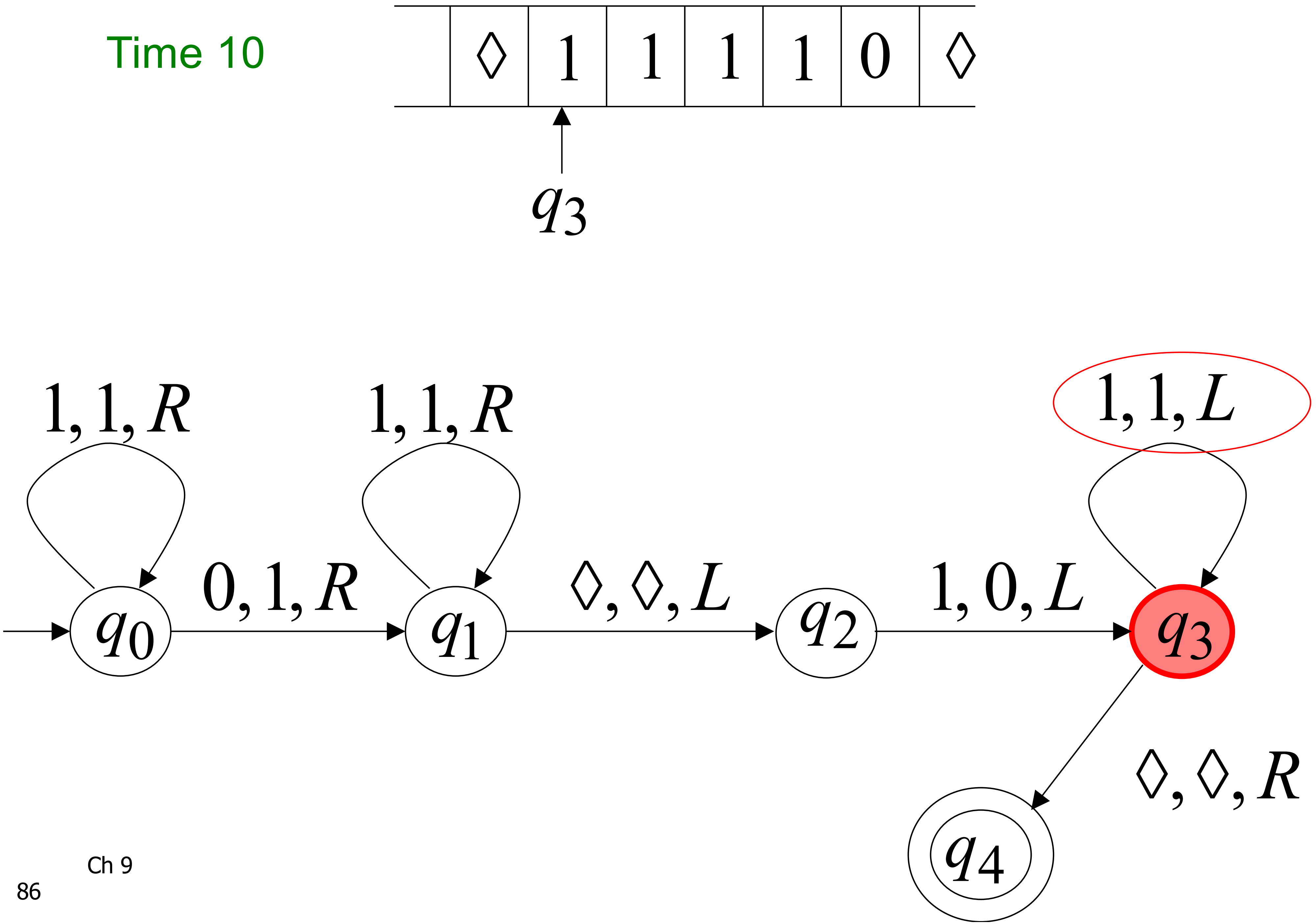


Ch 9

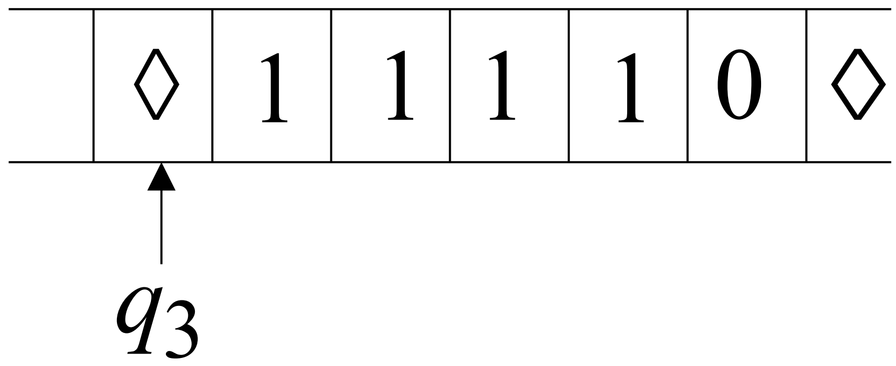
Time 9

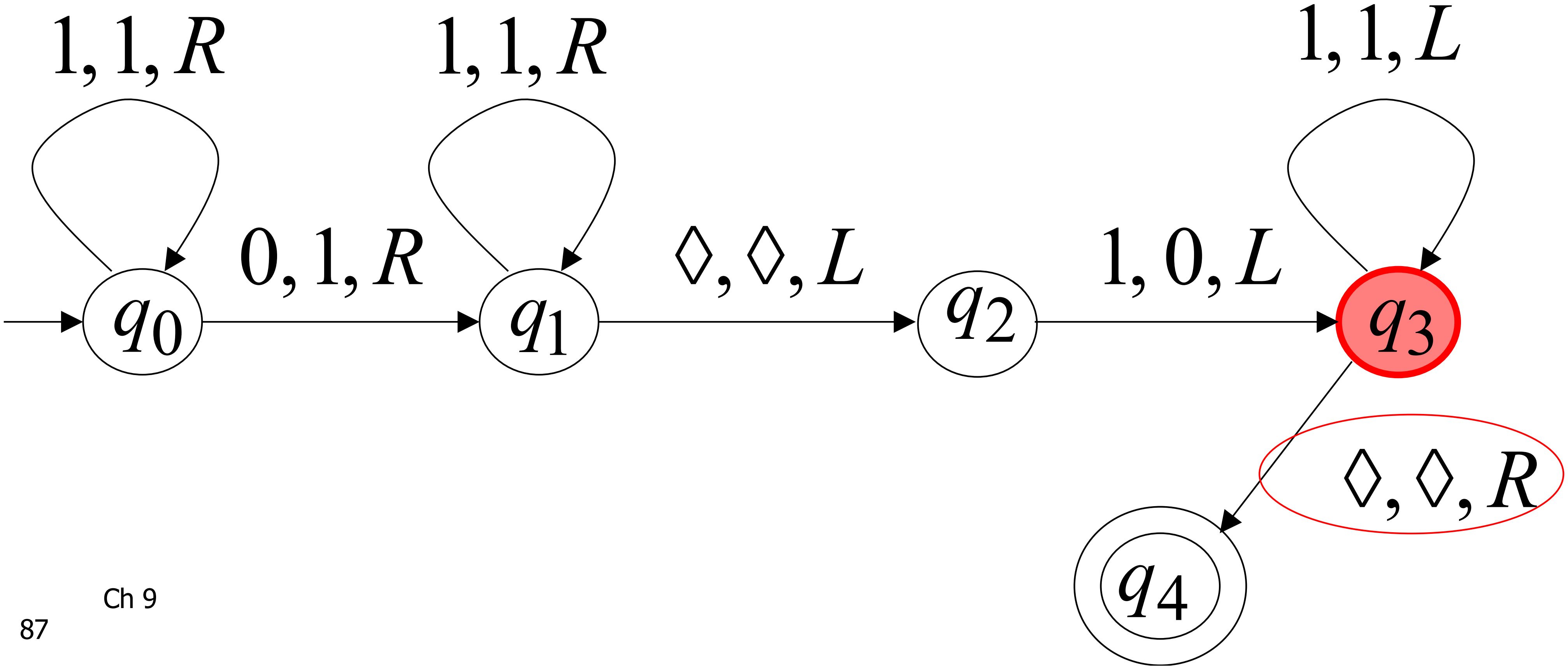


Time 10

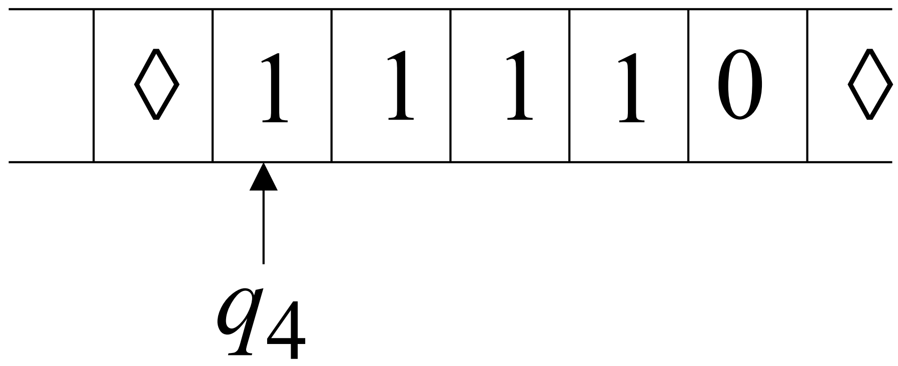


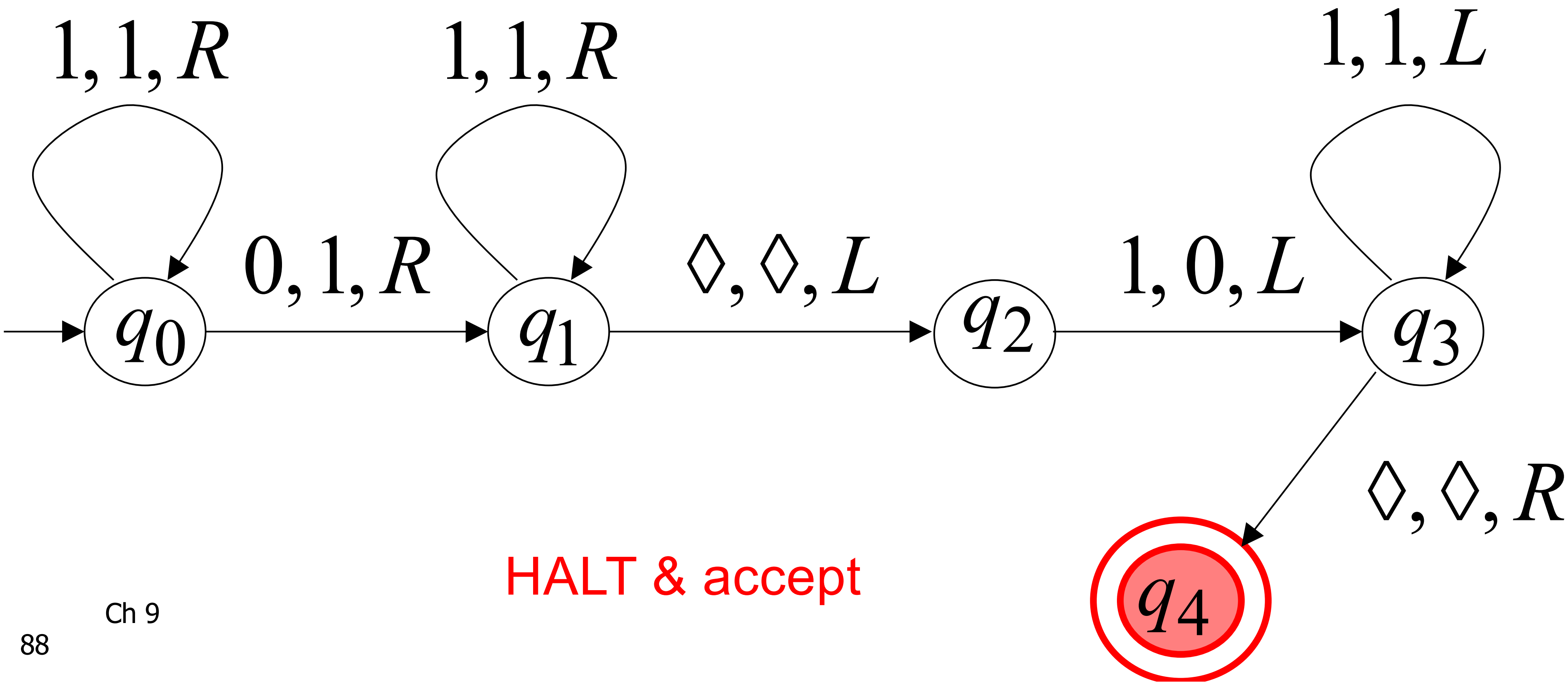
Time 11





Time 12





HALT & accept

Example 9.10

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

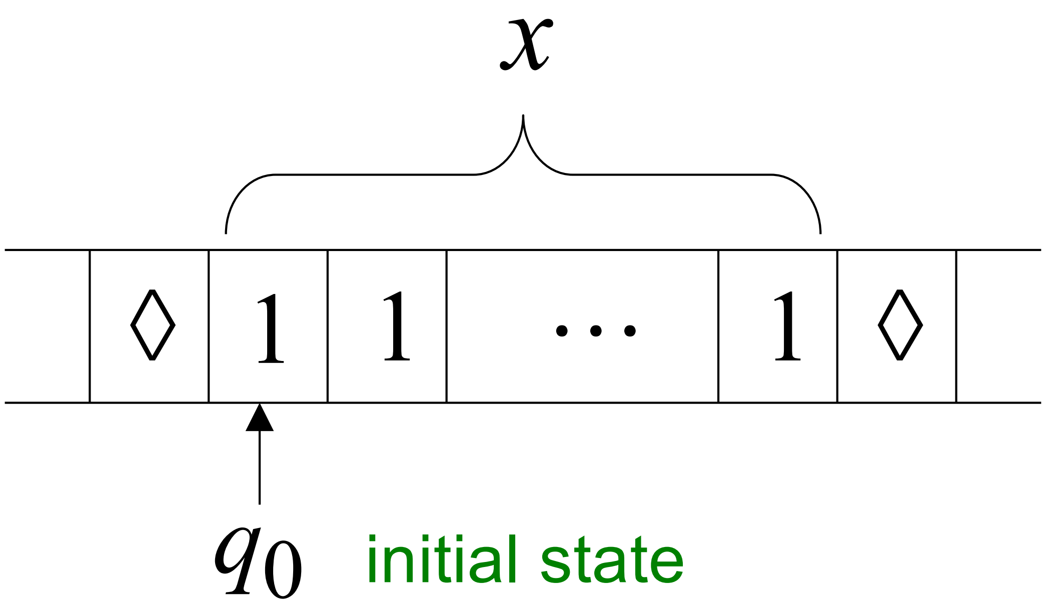
x is integer

Turing Machine:

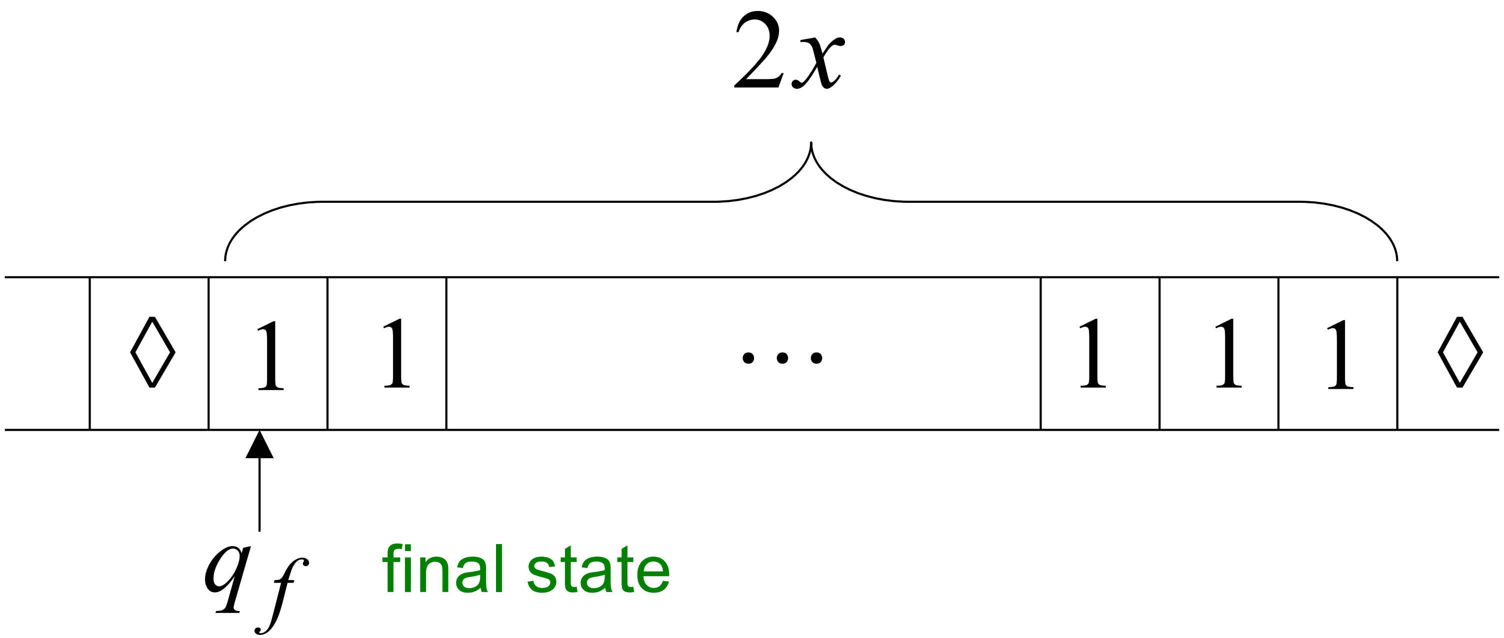
Input string: x unary

Output string: xx unary

Start



Finish

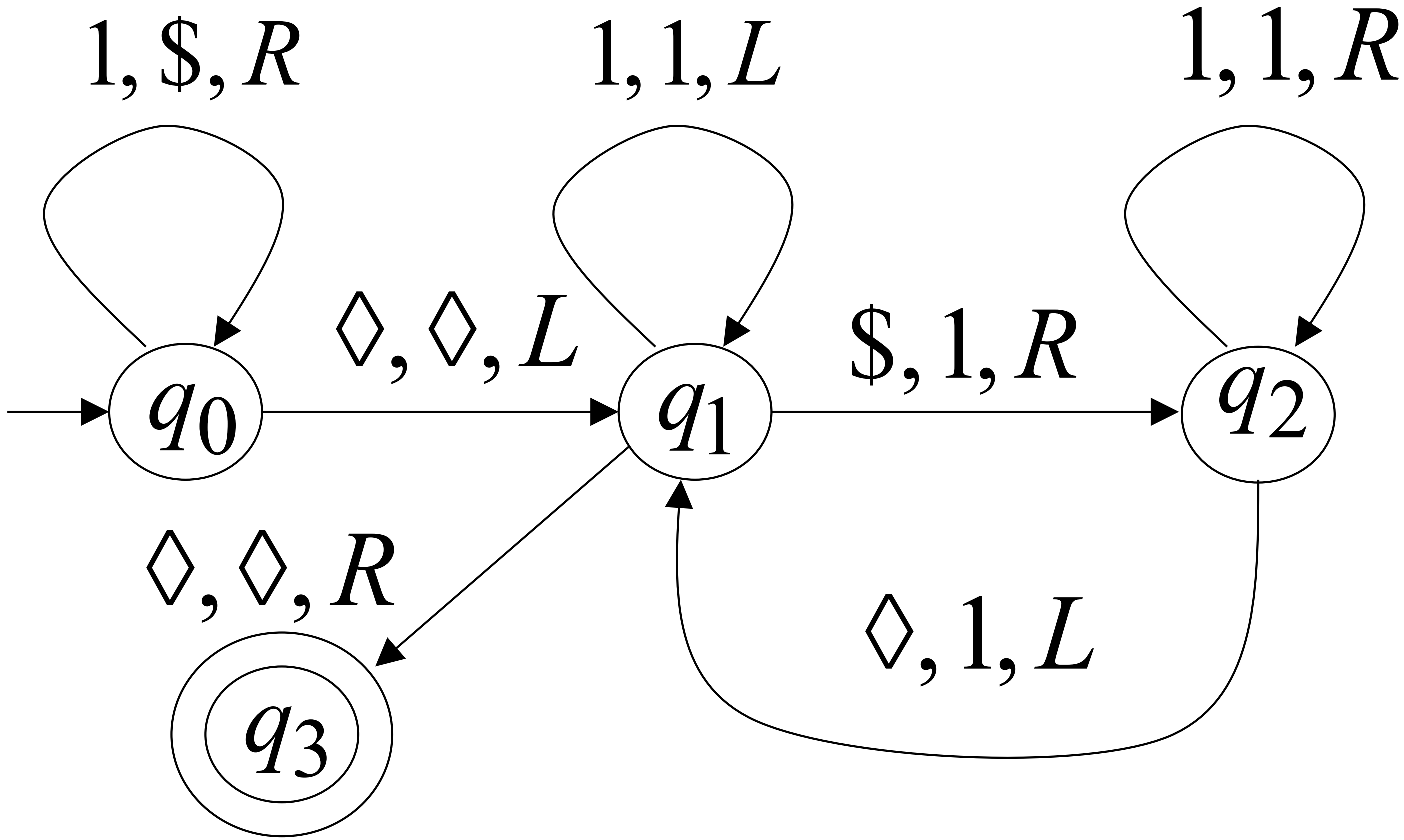


Turing Machine Pseudocode for $f(x) = 2x$

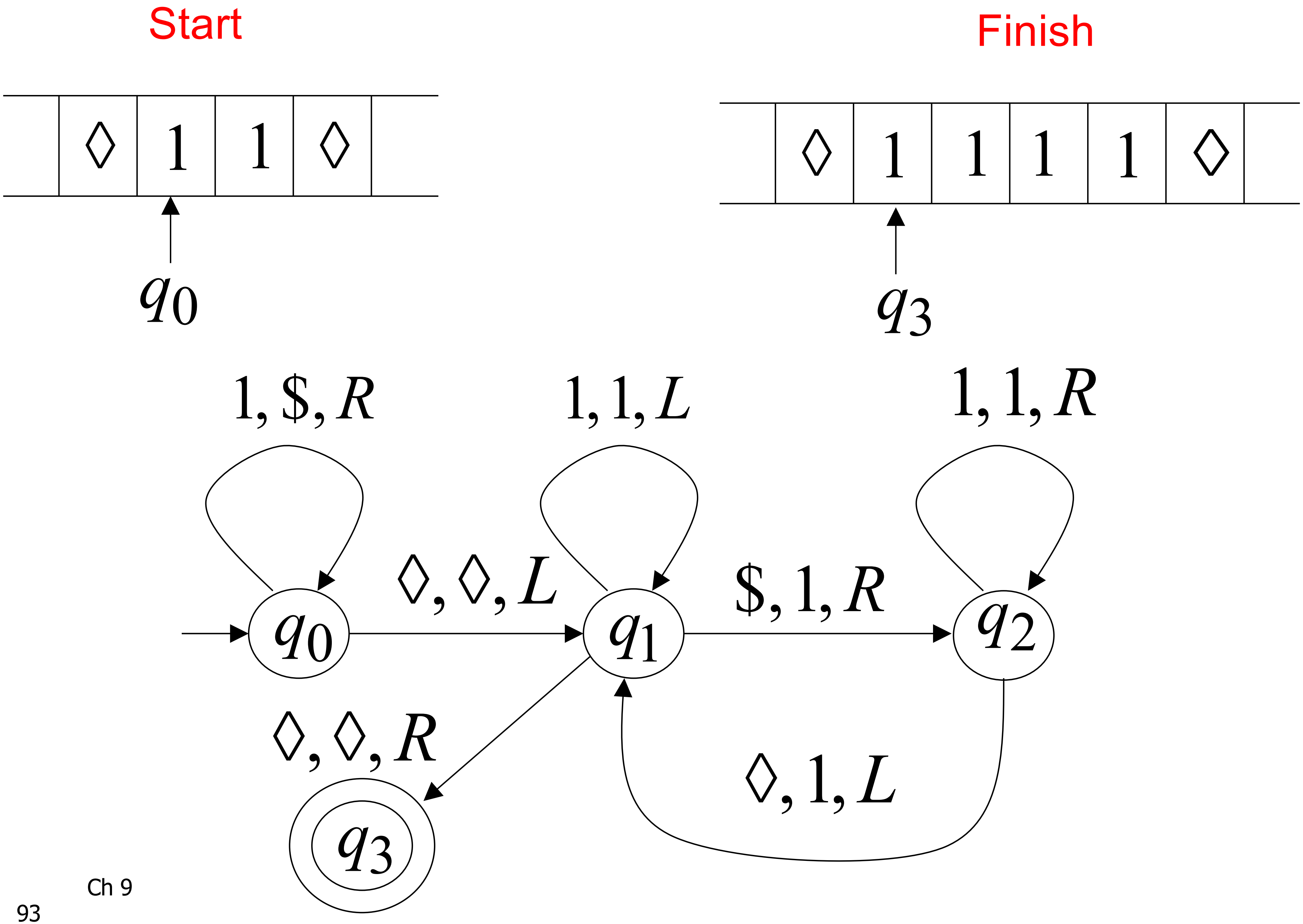
- Replace every 1 with $
- Repeat:
  - Find rightmost $, replace it with 1
  - Go to right end, insert 1

Until no more $ remain

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



Example



Example 9.11

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

is computable

Turing Machine for

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

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 $(x \leq y)$

Outline

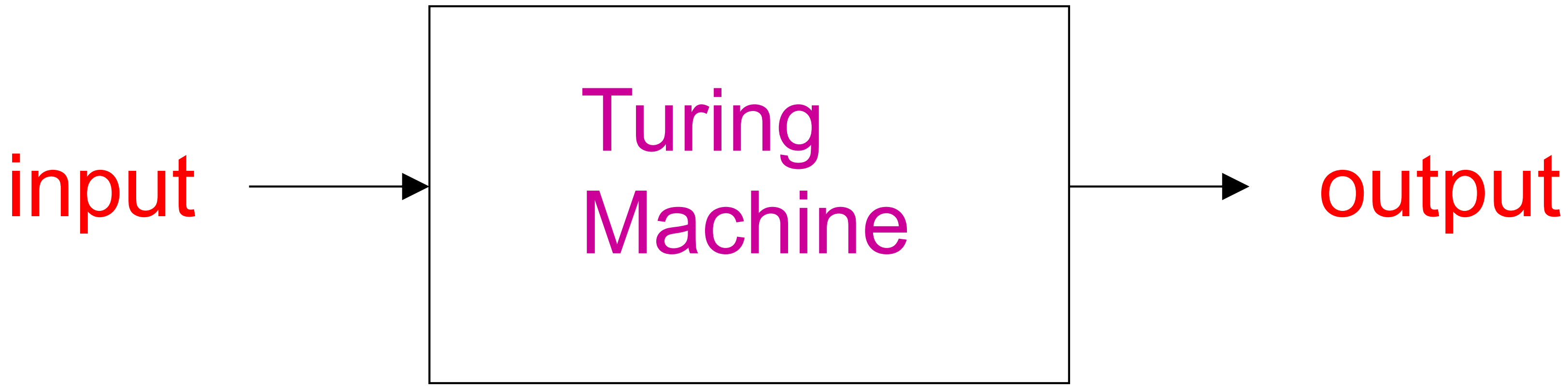
1 The Standard Turing Machine

2 Combining Turing Machines for Complicated Tasks

3 Turing's Thesis

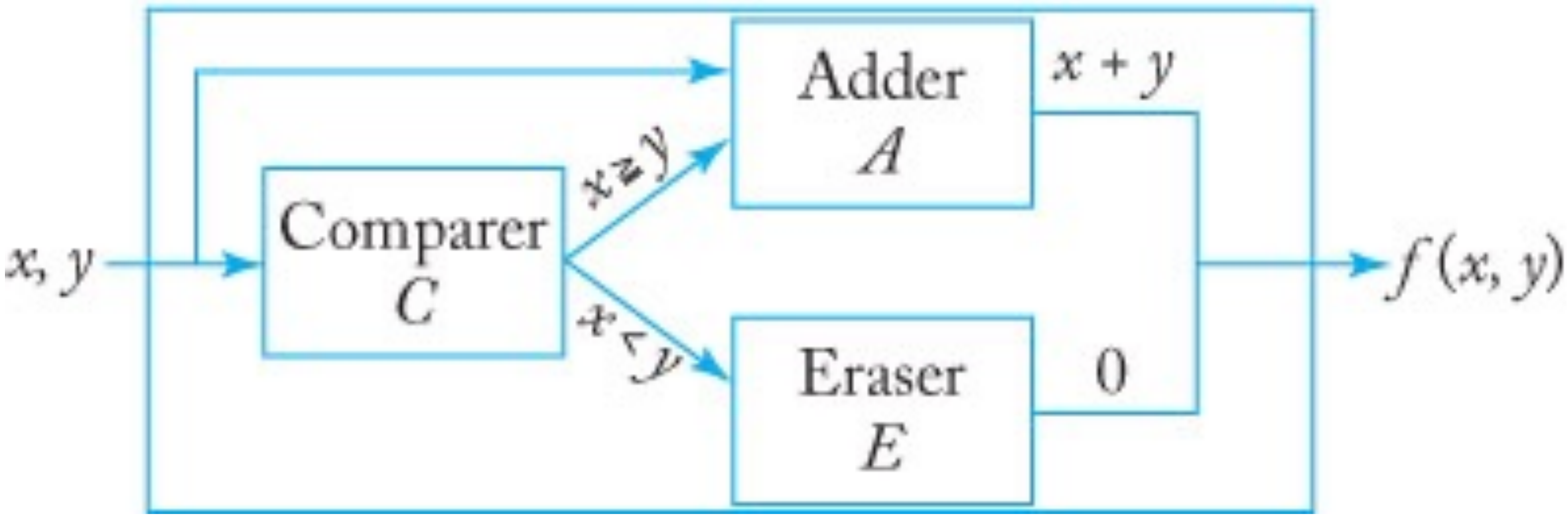
Combining Turing Machines

Block Diagram



Example 9.12:

$$f(x,y)=\begin{cases} x+y & \text{if} \quad x \geq y \\ 0 & \text{if} \quad x < y \end{cases}$$



$q_{C,0}w(x)0w(y) \overset{*}{\vdash} q_{A,0}w(x)0w(y)$ If $x \geqq y$ $\Longrightarrow$ $q_{A,0}w(x)0w(y) \overset{*}{\vdash} q_{A,f}w(x+y)0$

$q_{C,0}w(x)0w(y) \overset{*}{\vdash} q_{E,0}w(x)0w(y)$ If $x < y$ $\Longrightarrow$ $q_{E,0}w(x)0w(y) \overset{*}{\vdash} q_{E,f}0$

Example 9.13

- Consider the macroinstruction

if a then q_j else q_k,

$\delta(q_i, a) = (q_{j0}, a, R)$ for all $q_i \in Q$

$\delta(q_i, b) = (q_{k0}, b, R)$ for all $q_i \in Q$, and all $b \in \Gamma - \{a\}$

$\delta(q_{j0}, c) = (q_j, c, L)$ for all $c \in \Gamma$

$\delta(q_{k0}, c) = (q_k, c, L)$ for all $c \in \Gamma$

Example 9.14

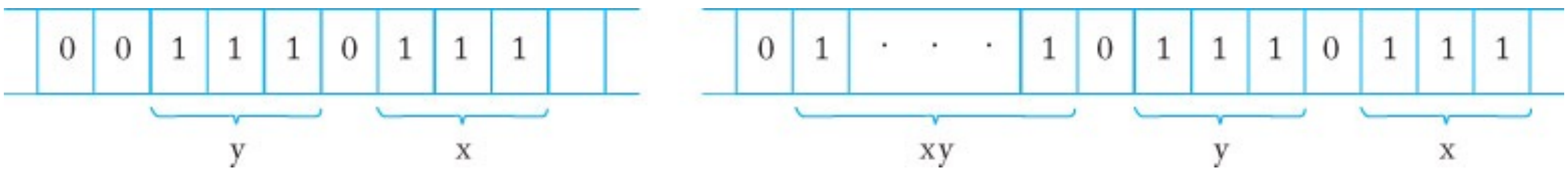
- Design a Turing machine that multiplies two positive integers in unary notation

1. Repeat the following steps until x contains no more 1's

   Find a 1 in x and replace it with another symbol a

   Replace the leftmost 0 by 0y

2. Replace all a's with 1's



Outline

1 The Standard Turing Machine

2 Combining Turing Machines for Complicated Tasks

3 Turing's Thesis

Turing's thesis:

Any computation carried out
by mechanical means
can be performed by a Turing Machine

(1930)

Computer Science Law:

A computation is mechanical
if and only if
it can be performed by a Turing Machine

There is no known model of computation
more powerful than Turing Machines

Definition of Algorithm:

An algorithm for function $f(w)$
is a
Turing Machine which computes $f(w)$

Algorithms are Turing Machines

When we say:

There exists an algorithm

We mean:

There exists a Turing Machine
that executes the algorithm

Turing Thesis

- Anything that can be done on any existing digital computer can also be done by a Turing machine
- No one has yet been able to suggest a problem, solvable by what we intuitively consider an algorithm, for which a Turing machine program cannot be written
- Alternative models have been proposed for mechanical computation, but none of them is more powerful than the Turing machine model

Short Quiz

- Construct Turing machines that will accept the following languages
  - $L=\{a^n b^n c^n : n \geq 0\}$
  - Even-length binary palindromes