Lecture 19

Turing Machines

COT 4420

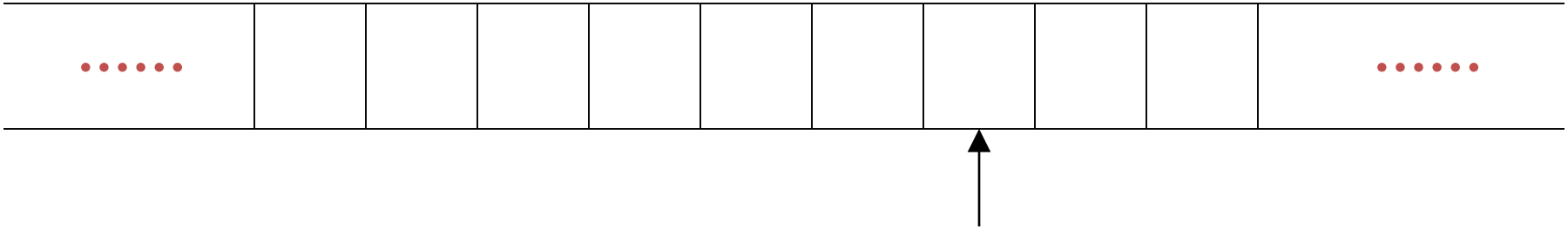
Theory of Computation

Chapter 9

Slides from Prof. Busch

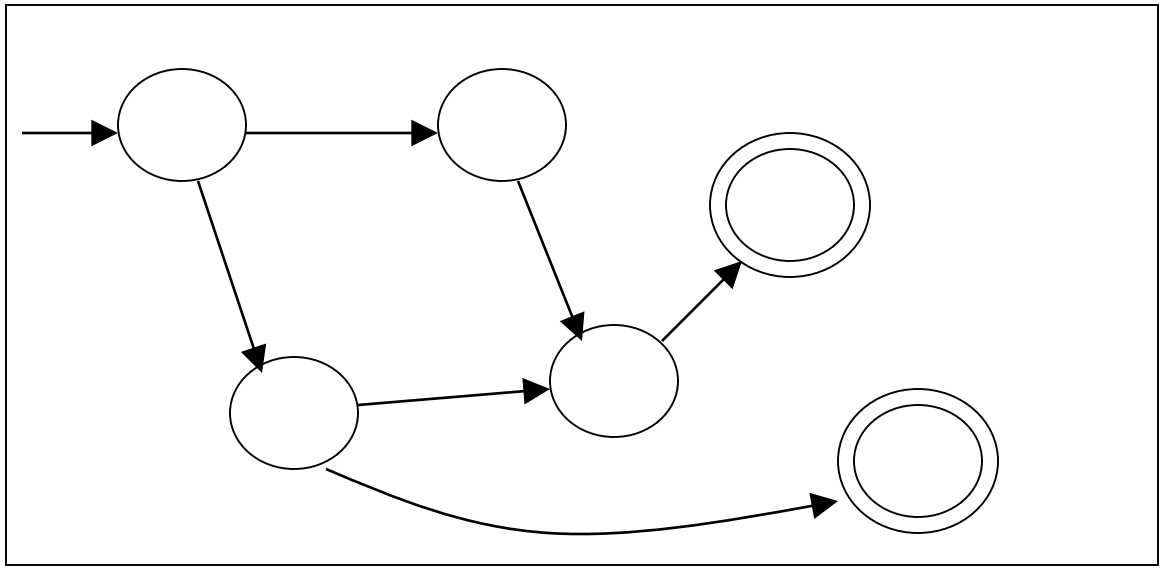
A Turing Machine

Tape



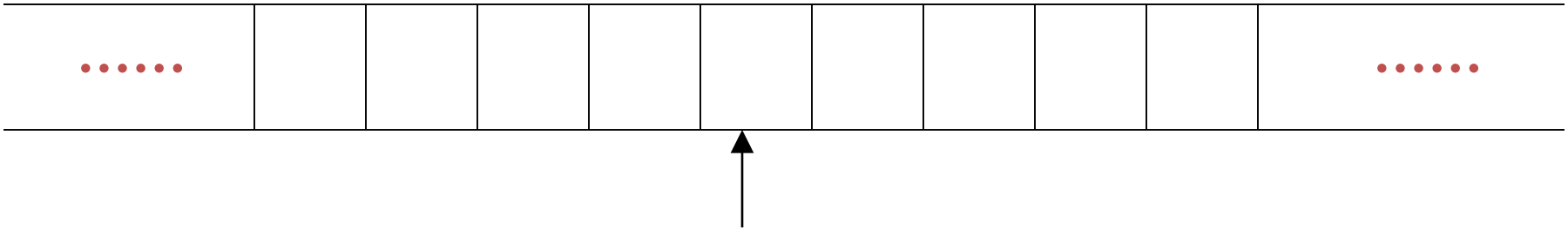
Read-Write head

Control Unit



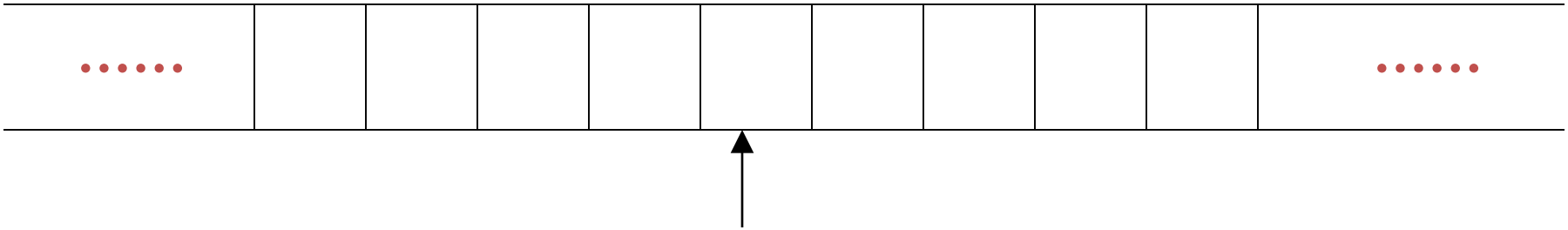
The Tape

No boundaries -- infinite length



Read-Write head

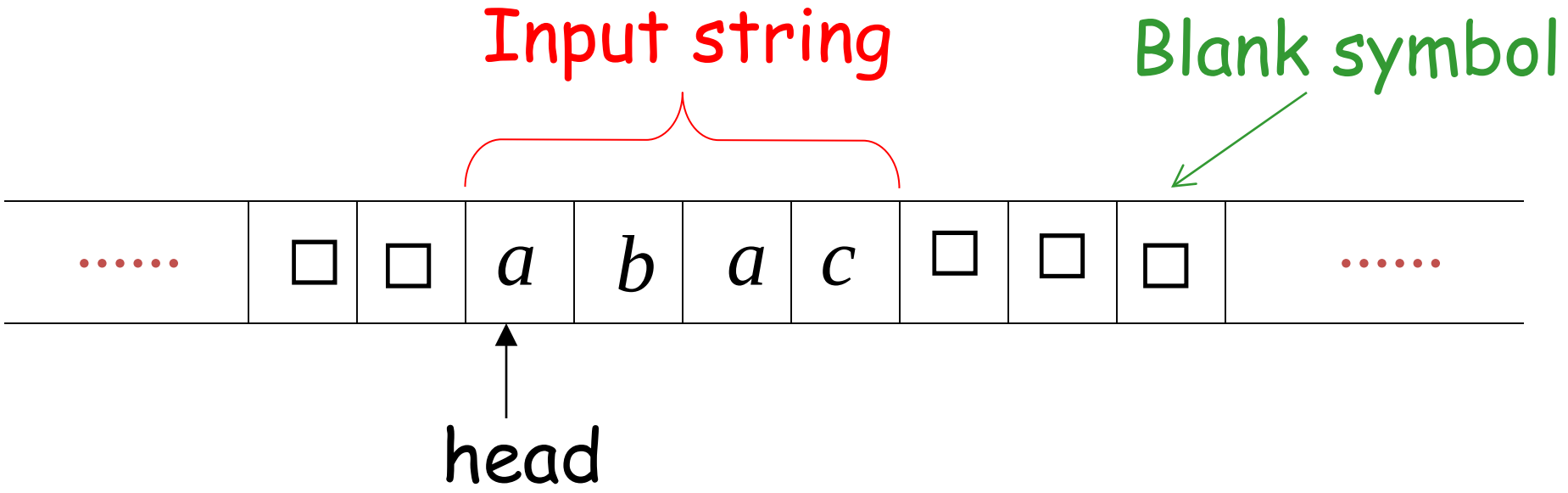
The head moves Left or Right



The head at each transition:

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

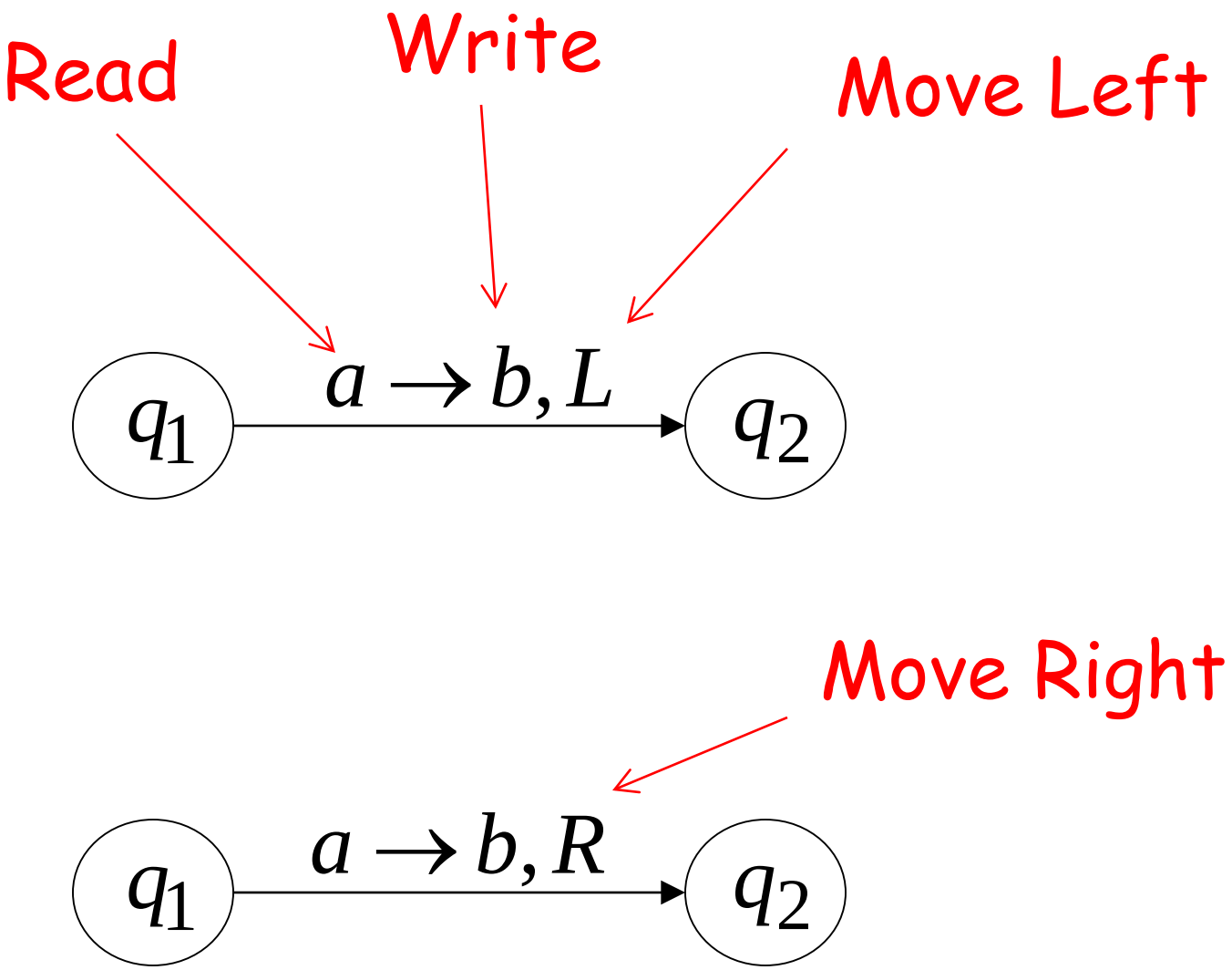
The Input String

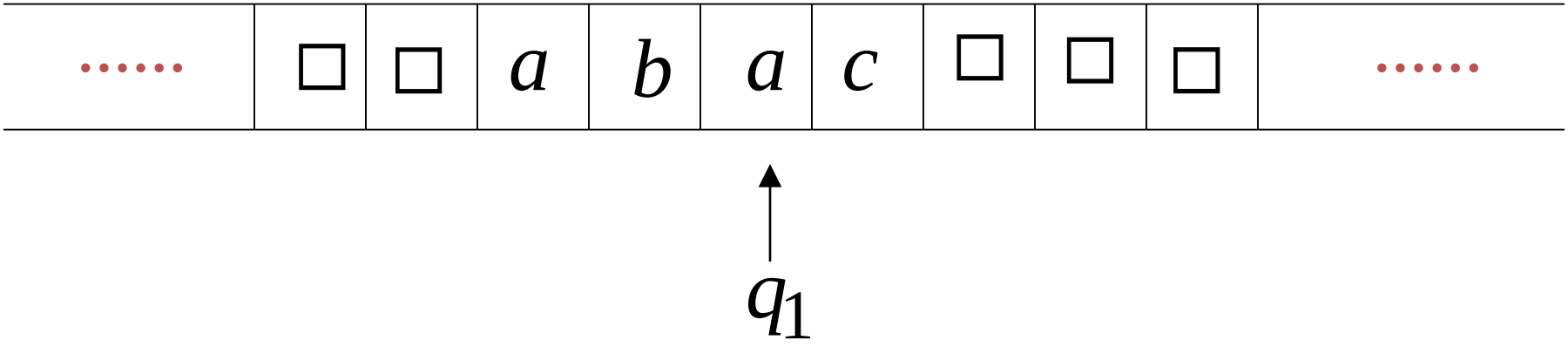


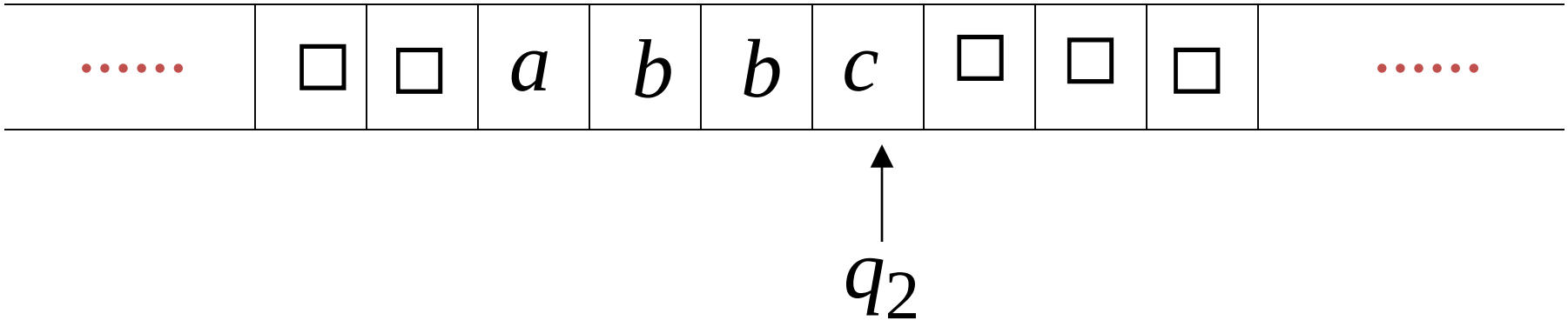
Head starts at the leftmost position

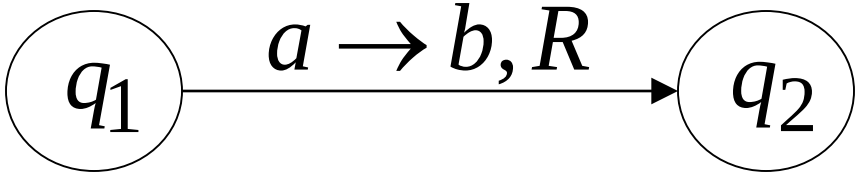
of the input string

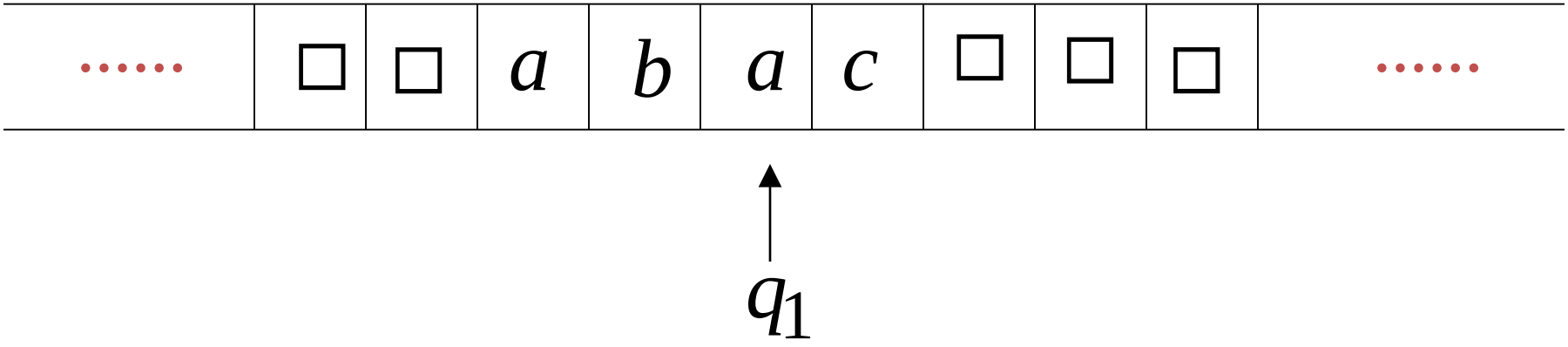
States & Transitions

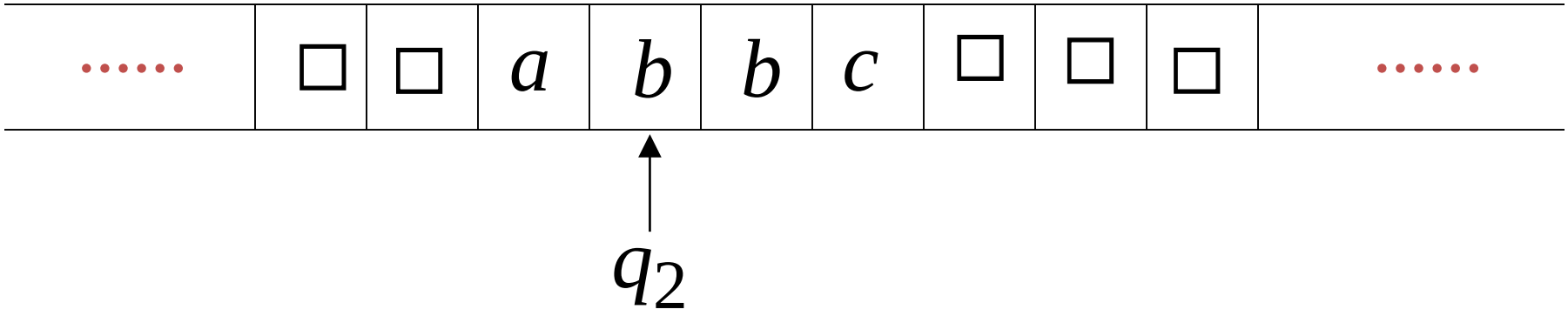


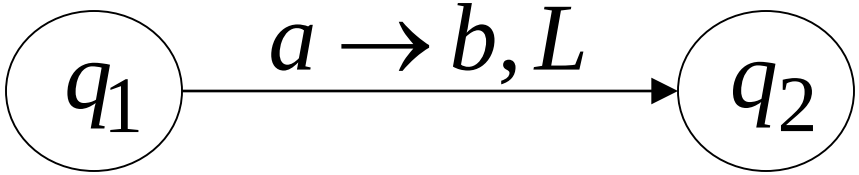

.....
□ □ a b a c □ □ □
.....
q_1

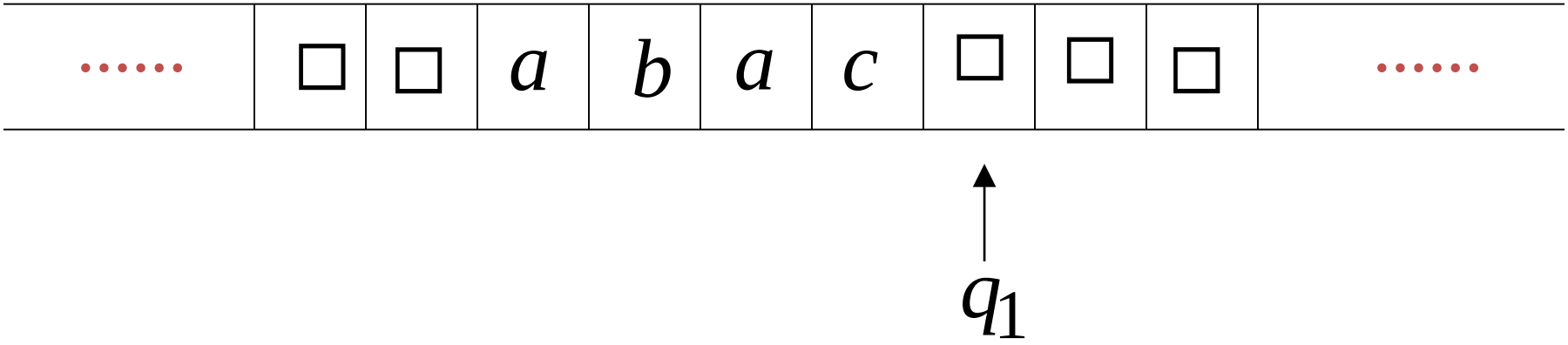

.....
□ □ a b b c □ □ □
.....
q_2

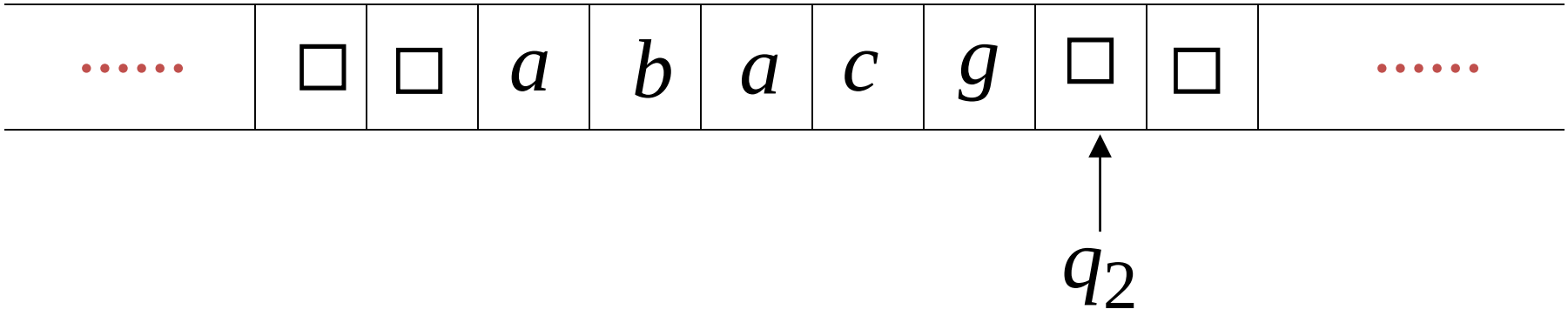

q_1
$a \to b, R$
q_2

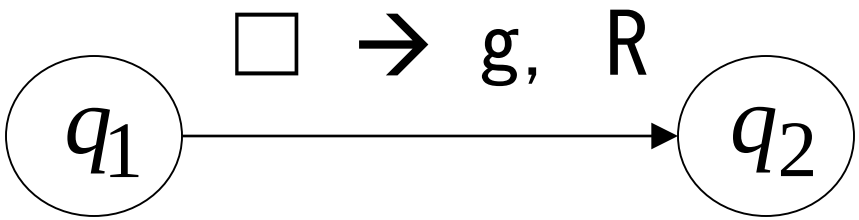

.....
□ □ a b a c □ □ □
.....
q_1


.....
□ □ a b b c □ □ □
.....
q_2


q_1
$a \rightarrow b, L$
q_2


.....
□ □ a b a c □ □ □
.....
q_1


.....
□ □ a b a c g □ □
.....
q_2

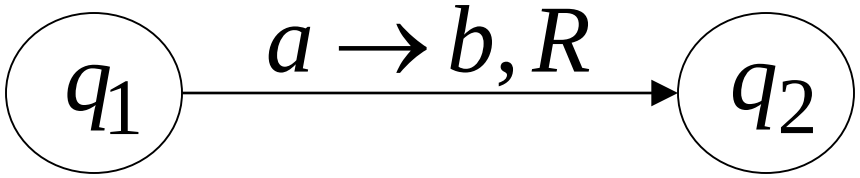

□ → g, R
q_1
q_2

Formal Definition of Turing Machine

$M = <Q, \Sigma, \Gamma, \delta, q_0, \square, F>$

- Q: a finite set of internal states
- Σ: is the input alphabet ($\Sigma \subseteq \Gamma - \{\square\}$)
- **Γ**: finite set of symbols called the tape alphabet
- δ: transition function ($Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$)
- q_0: the initial state ($q_0 \in Q$)
- $\square$: is a special symbol called blank ($\square \in \Gamma$)
- F: a set of final/accepting states ($F \subseteq Q$)

Transition Function

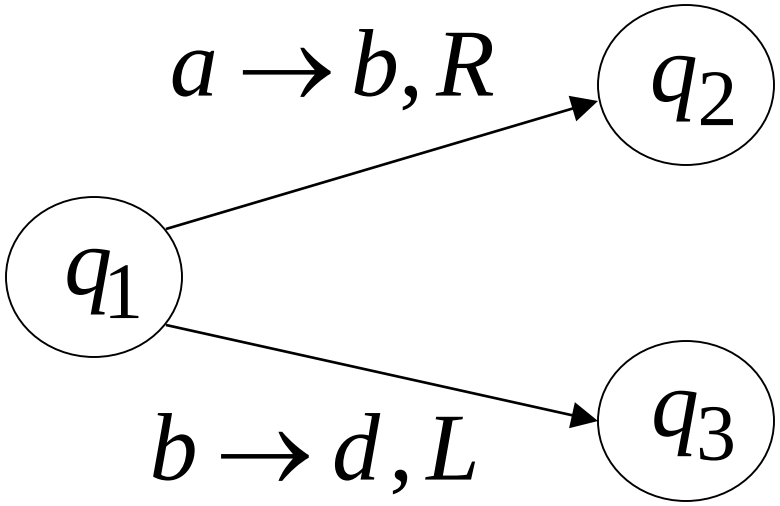


$$\delta(q_1, a) = (q_2, b, R)$$

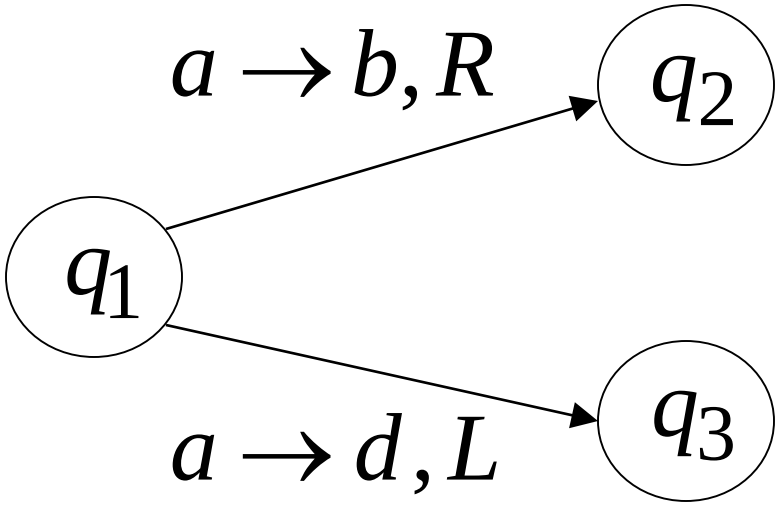
Determinism

- Turing Machines are deterministic

Allowed



Not Allowed

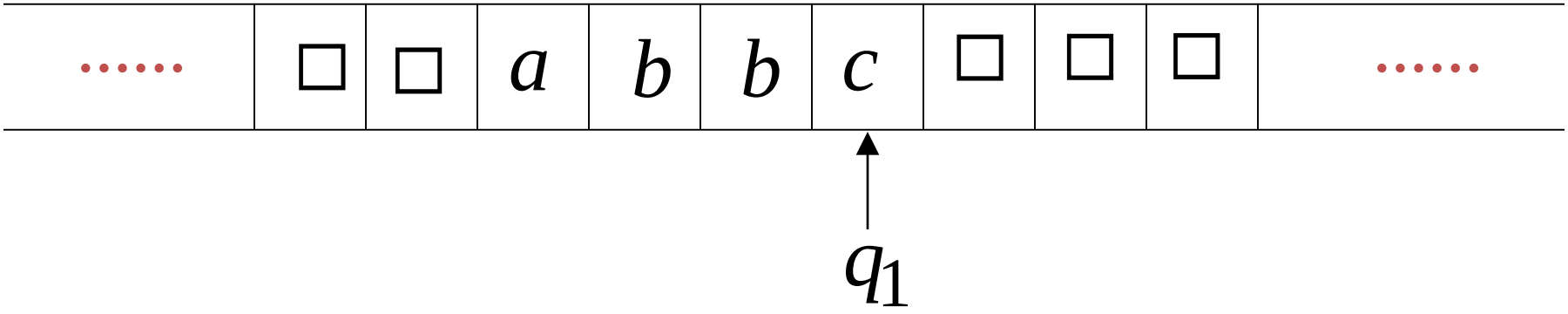


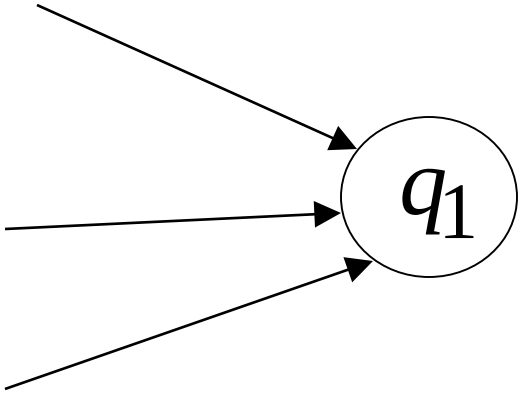
- No lambda transitions allowed
- It is okay to not have transitions for some alphabet

Halting

The machine ***halts*** in a state if there is no transition to follow

Halting Example 1:

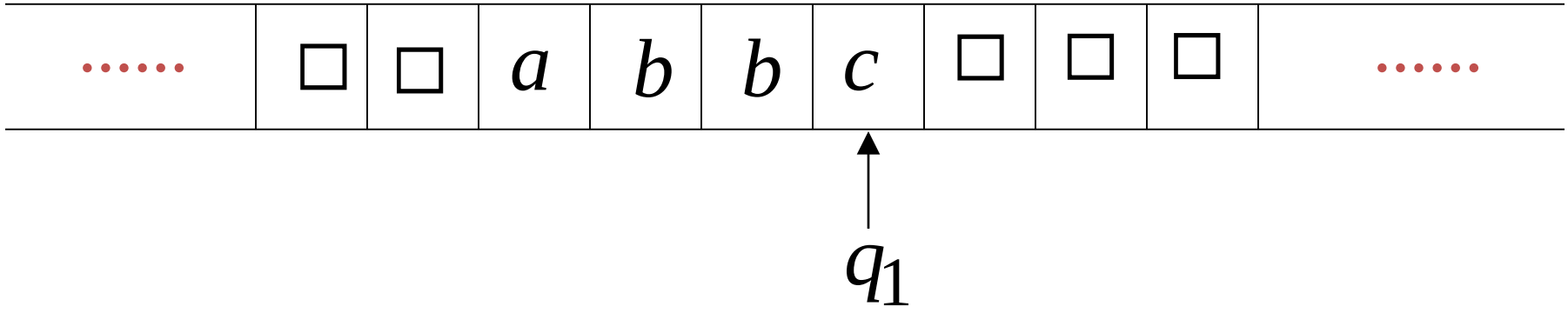


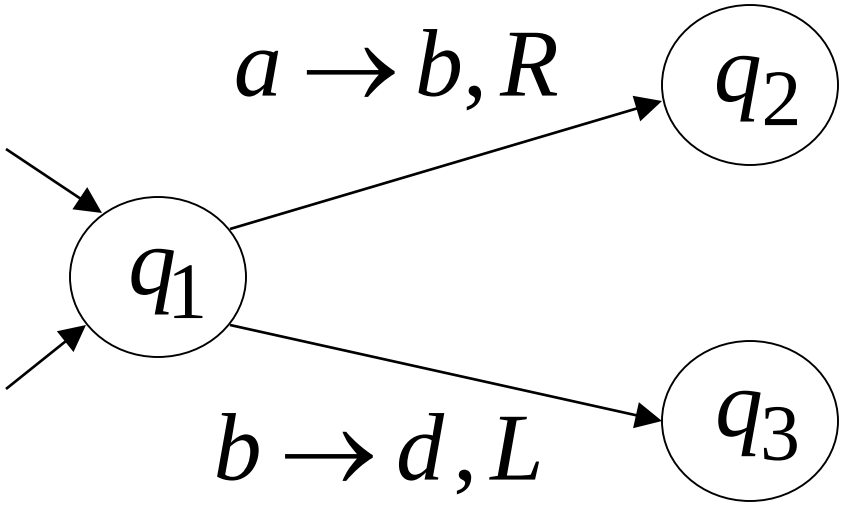


No transition from q_1

HALT!!!

Halting Example 2:



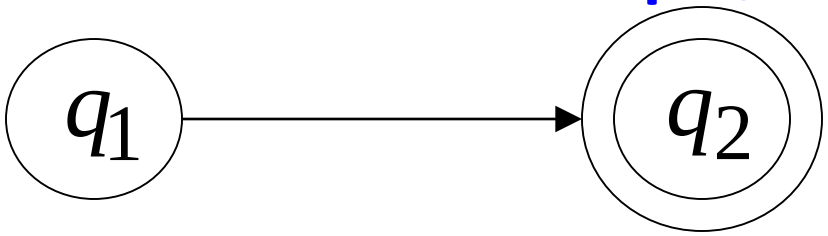


No possible transition

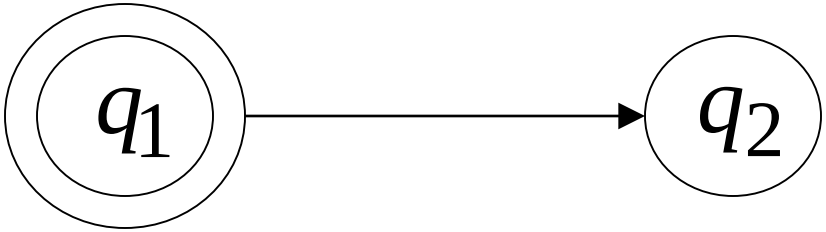
From q_1 and symbol c

HALT!!!

Accept/Final States



Allowed

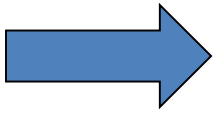


Not Allowed

- Accepting states have no outgoing transitions
- The machine halts and accepts

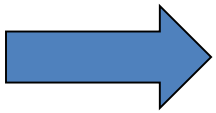
Acceptance

Accept Input string



If machine halts in a final state

Reject Input string



If machine halts in a non-final state

or

If machine enters an *infinite loop*

Observation:

In order to accept an input string,

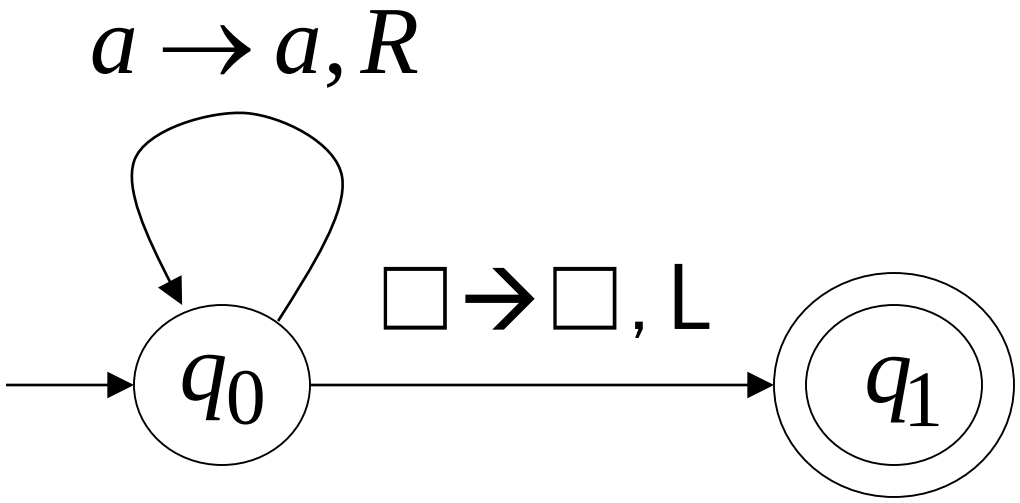
it is not necessary to scan all the

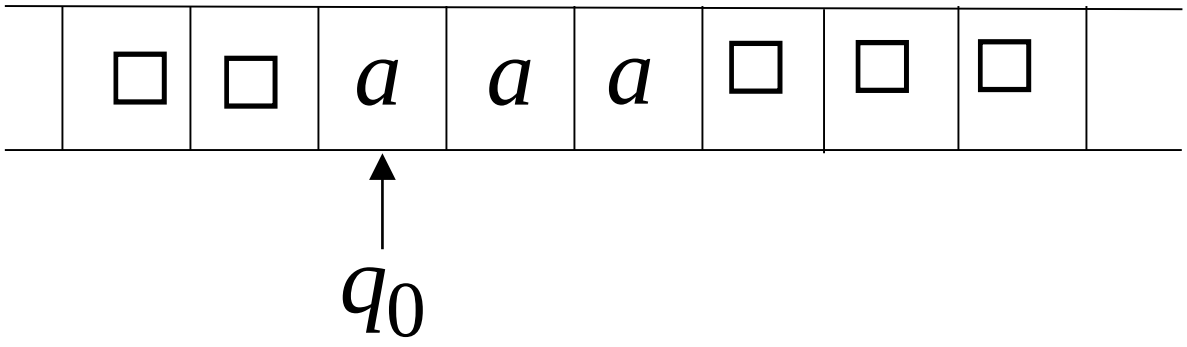
symbols in the string

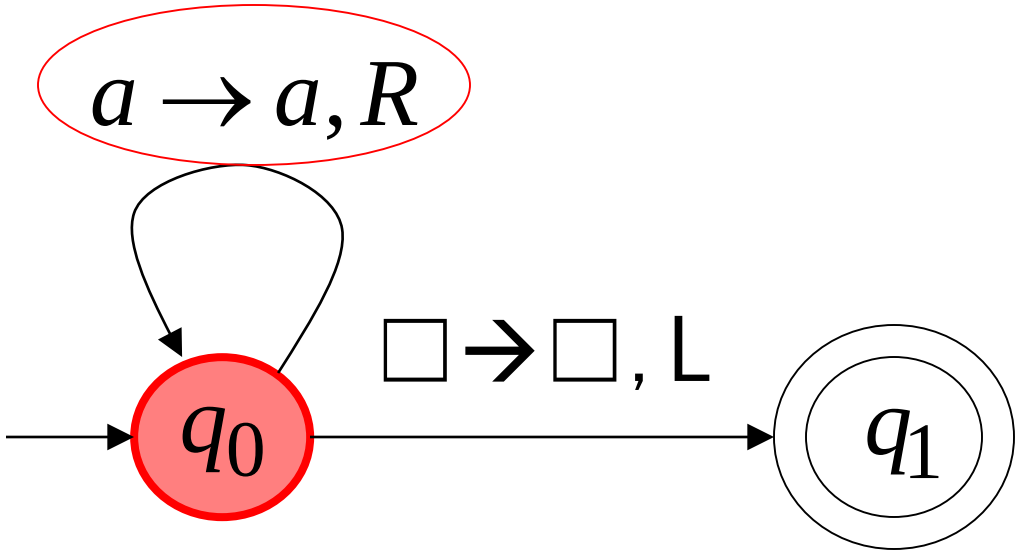
Turing Machine Example

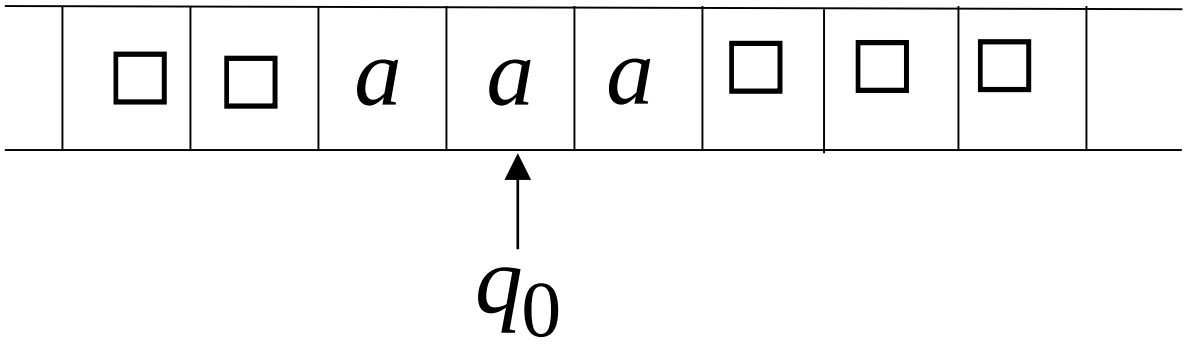
Input alphabet {a}

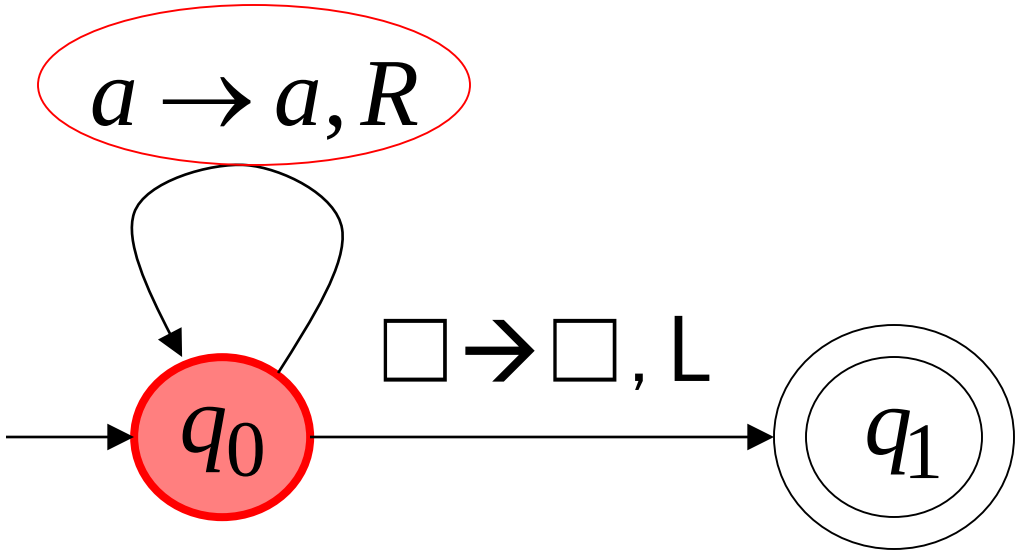
Accepts the language: a*

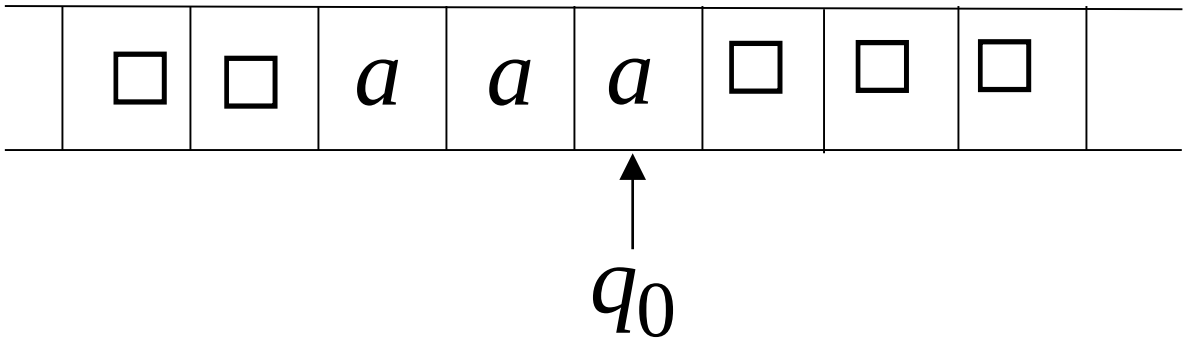


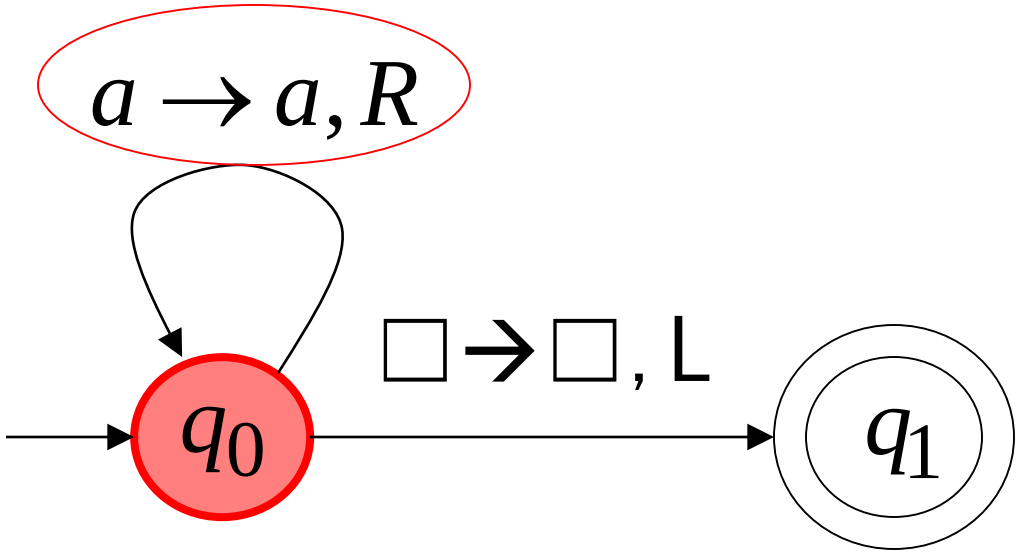

□ □ a a a □ □ □
q_0

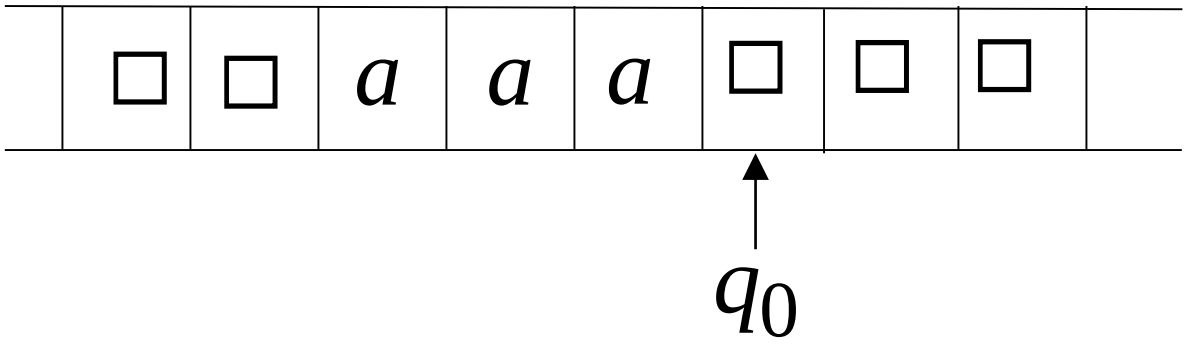

$a \rightarrow a, R$
□→□, L
q_0
q_1

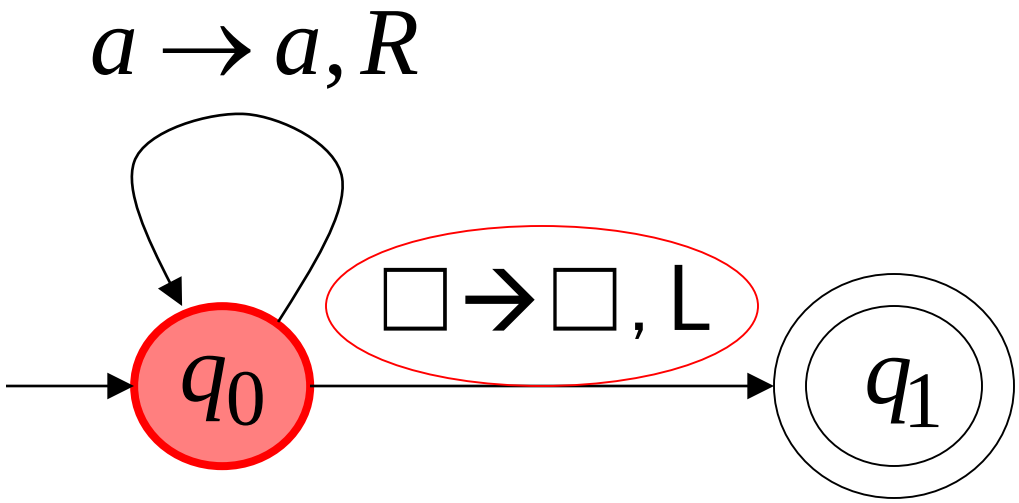

□ □ a a a □ □ □
q_0

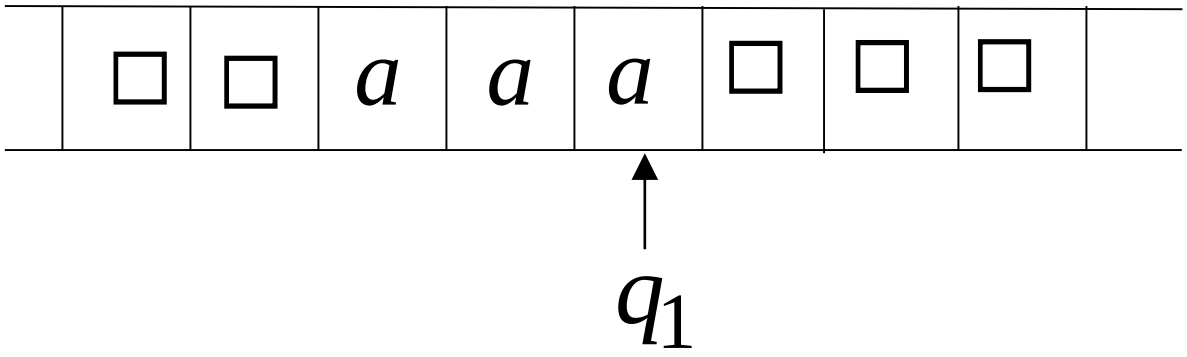

$a \rightarrow a, R$
□→□, L
q_0
q_1

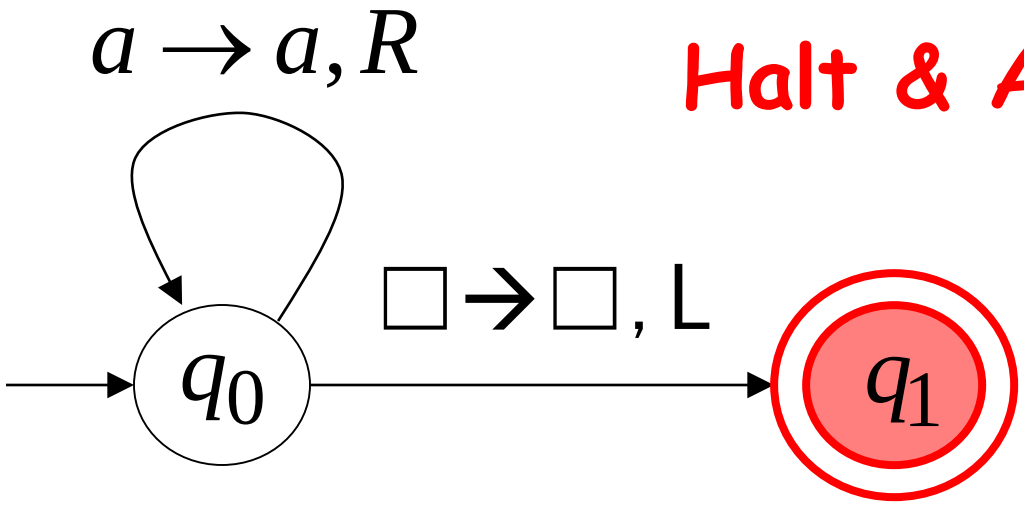

□ □ a a a □ □ □
q_0


$a \rightarrow a, R$
□→□, L
q_0
q_1


□ □ a a a □ □ □
q_0

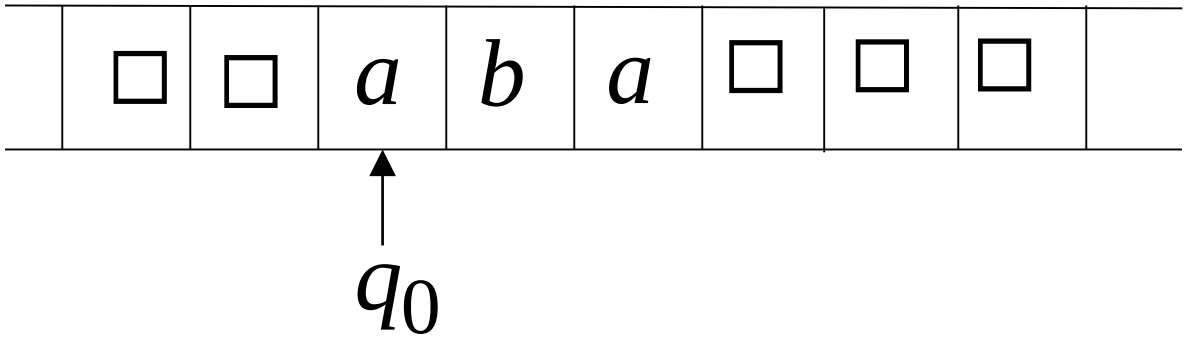

$a \rightarrow a, R$
□→□, L
q_0
q_1

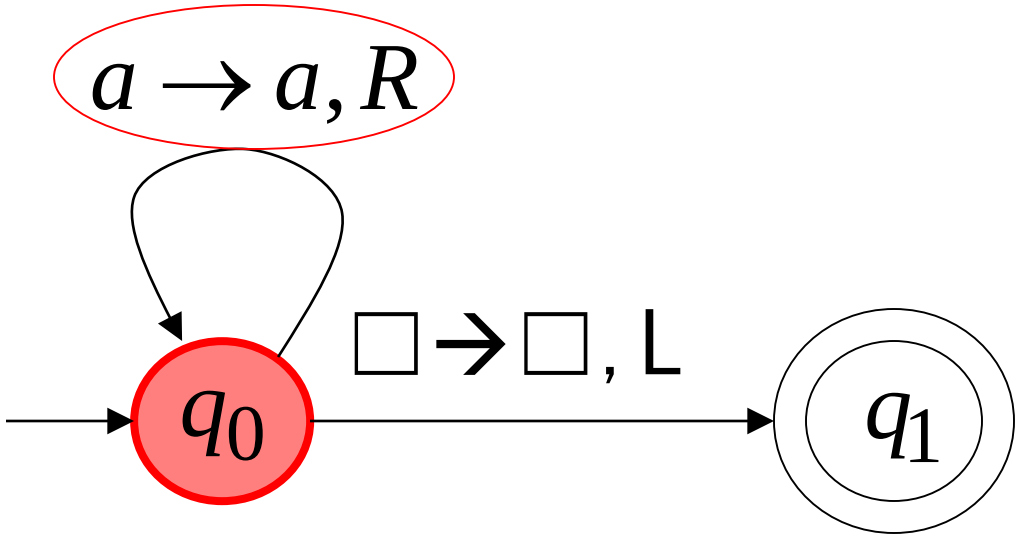


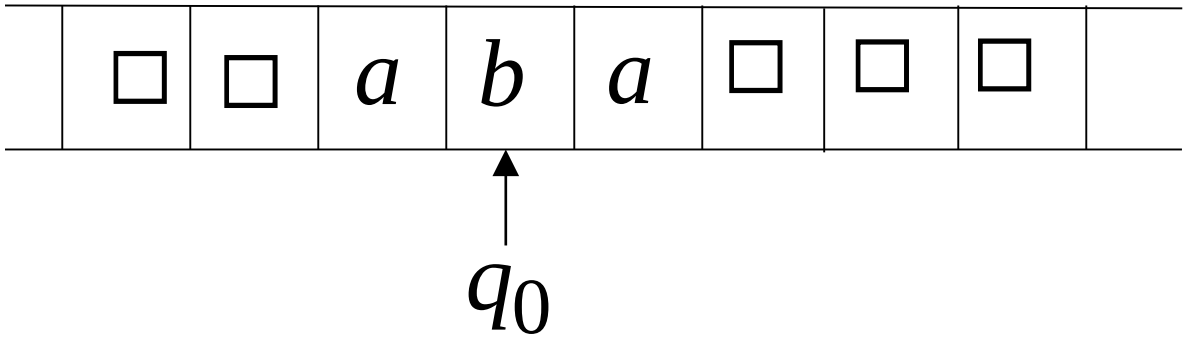


Halt & Accept

Rejection Example

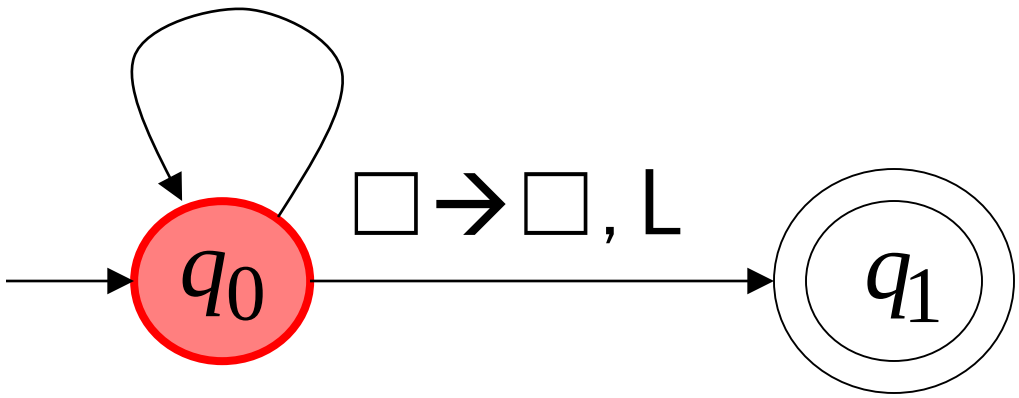






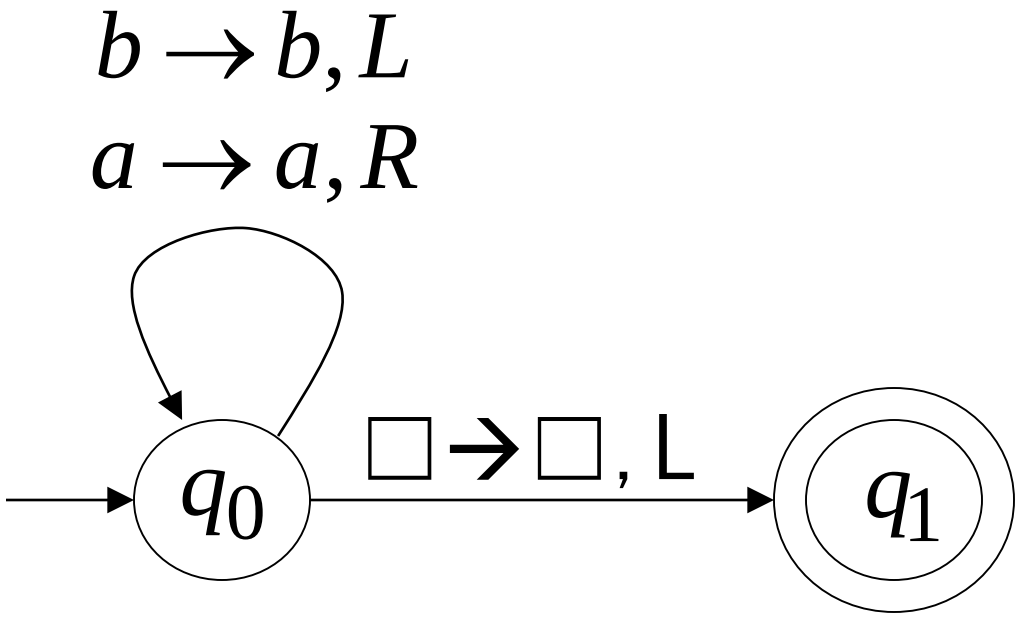
No possible Transition

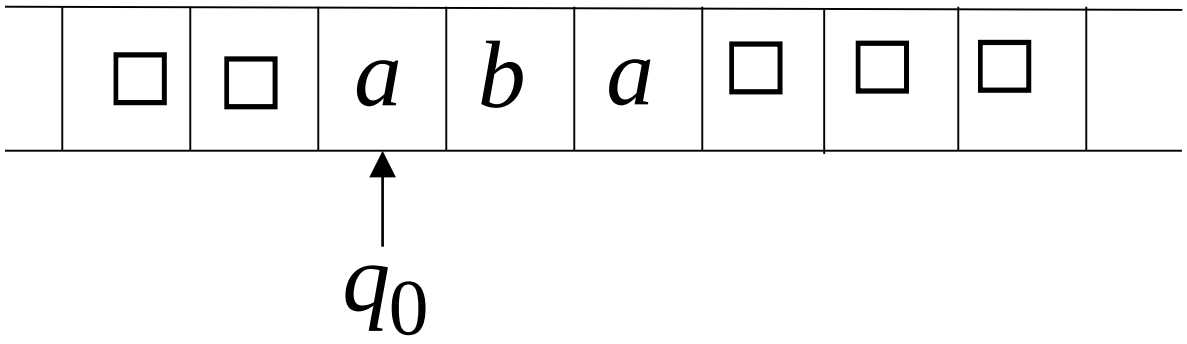
Halt & Reject

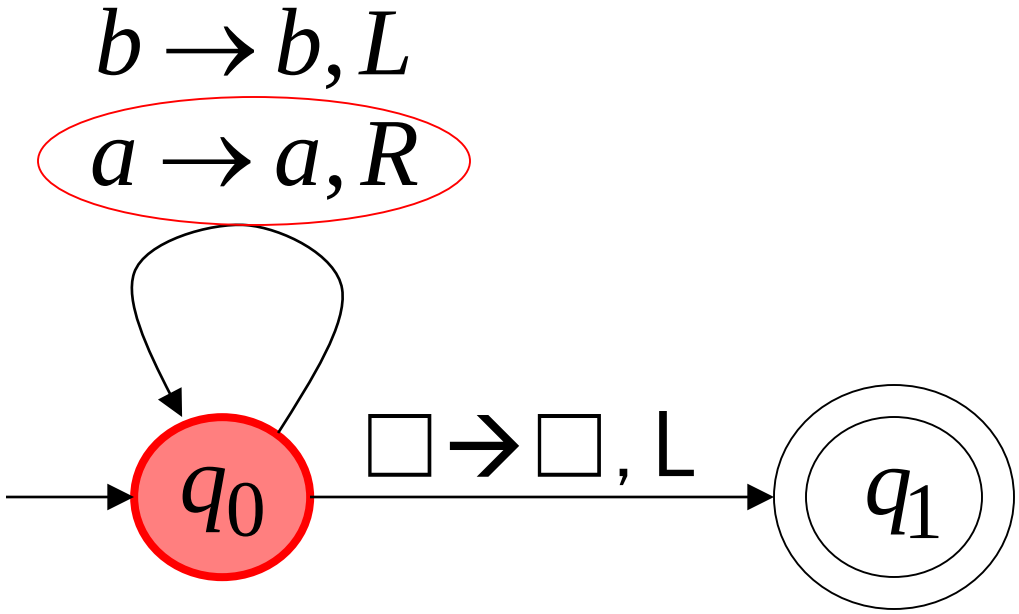


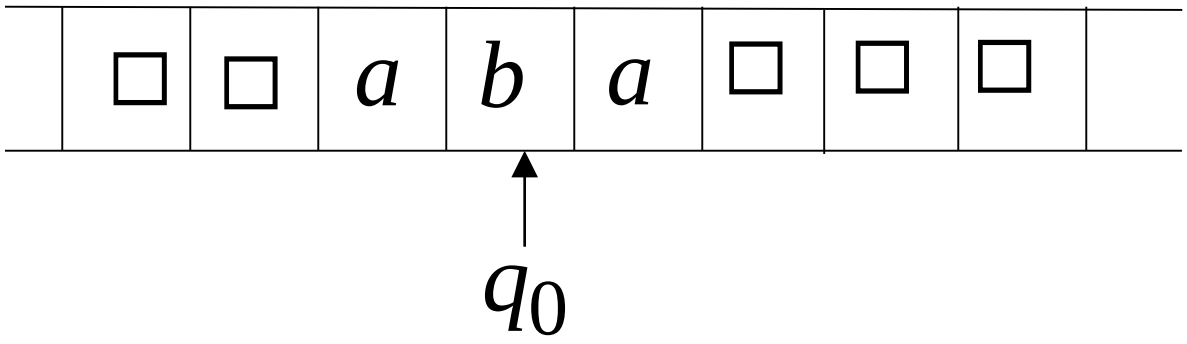
Infinite Loop Example

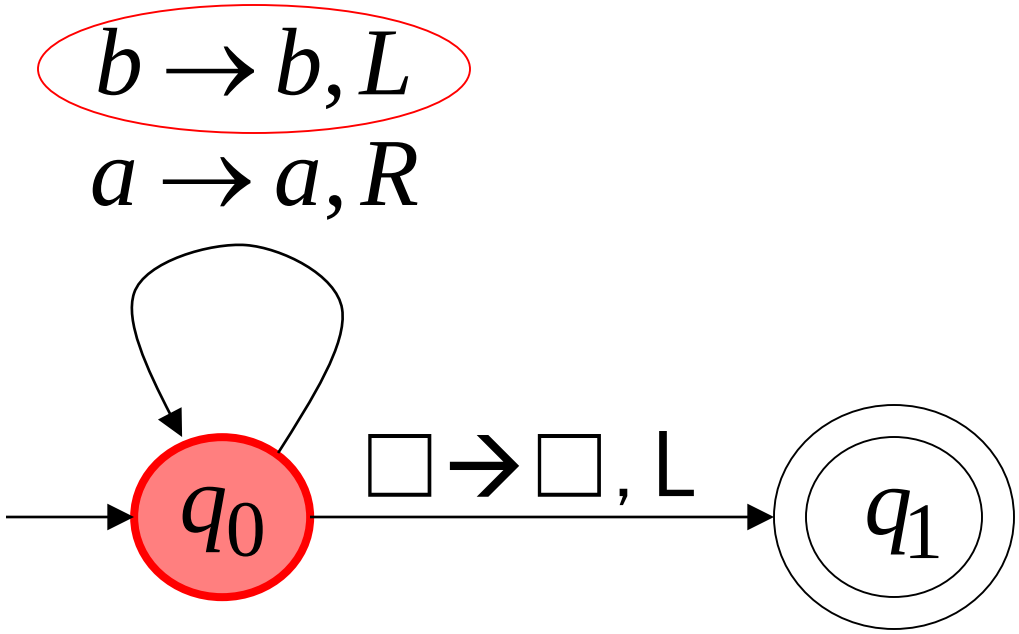
A Turing machine for language $a^*+b(a+b)^*$

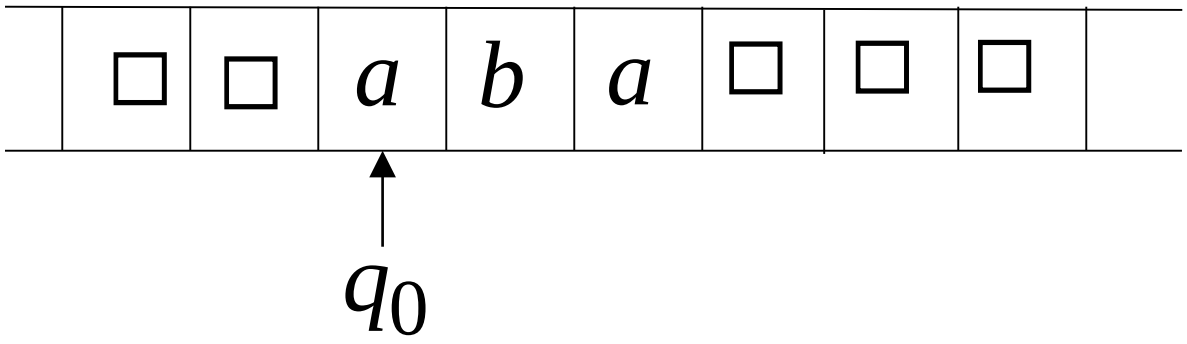


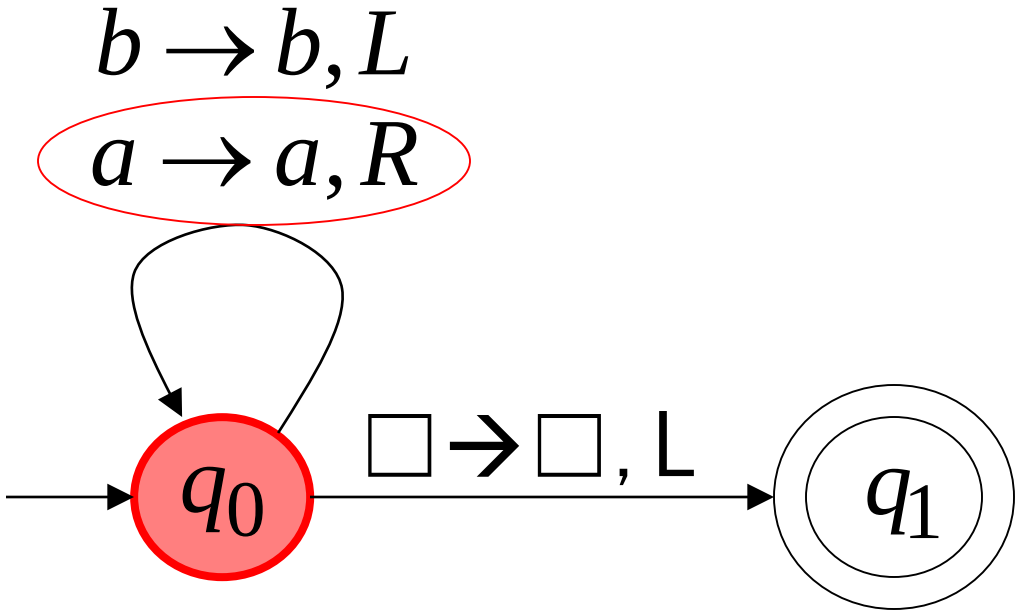

□ □ a b a □ □ □
q_0


$b \to b, L$
$a \to a, R$
q_0
□ → □, L
q_1


□ □ a b a □ □ □
q_0


$b \to b, L$
$a \to a, R$
q_0
□ → □, L
q_1


□ □ a b a □ □ □
q_0

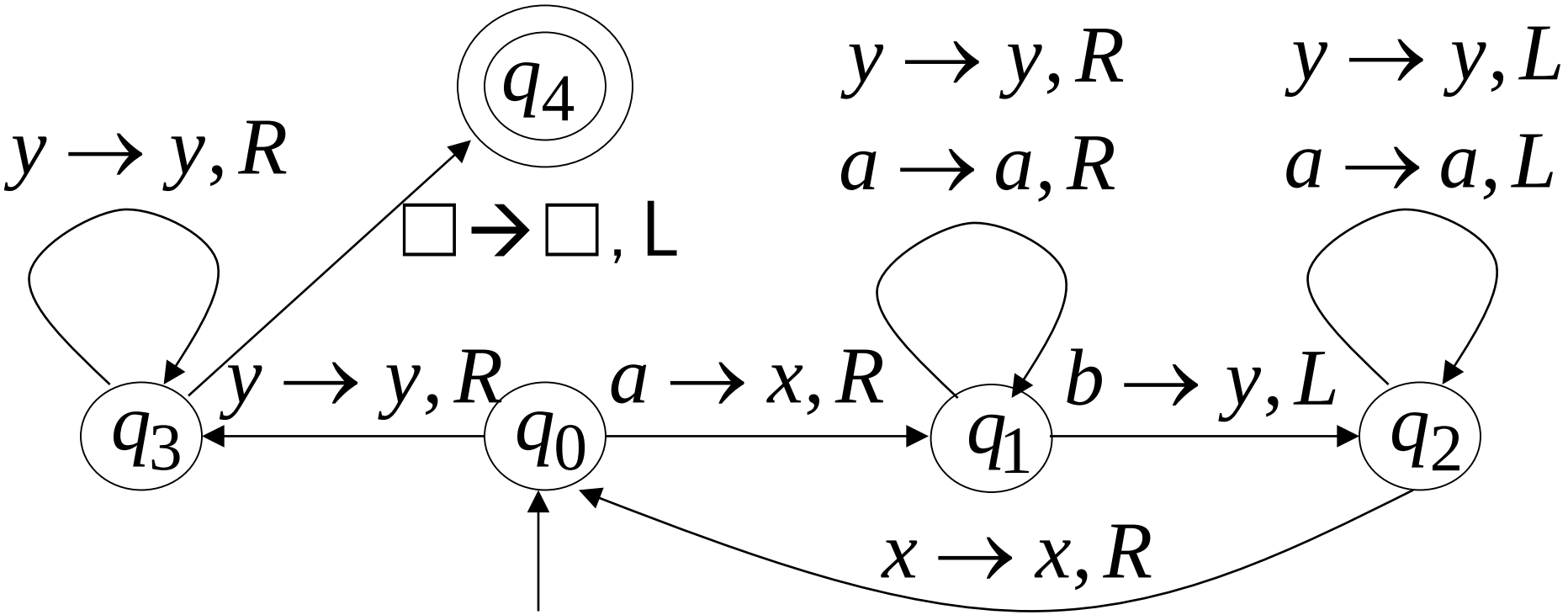

$b \to b, L$
$a \to a, R$
q_0
□ → □, L
q_1

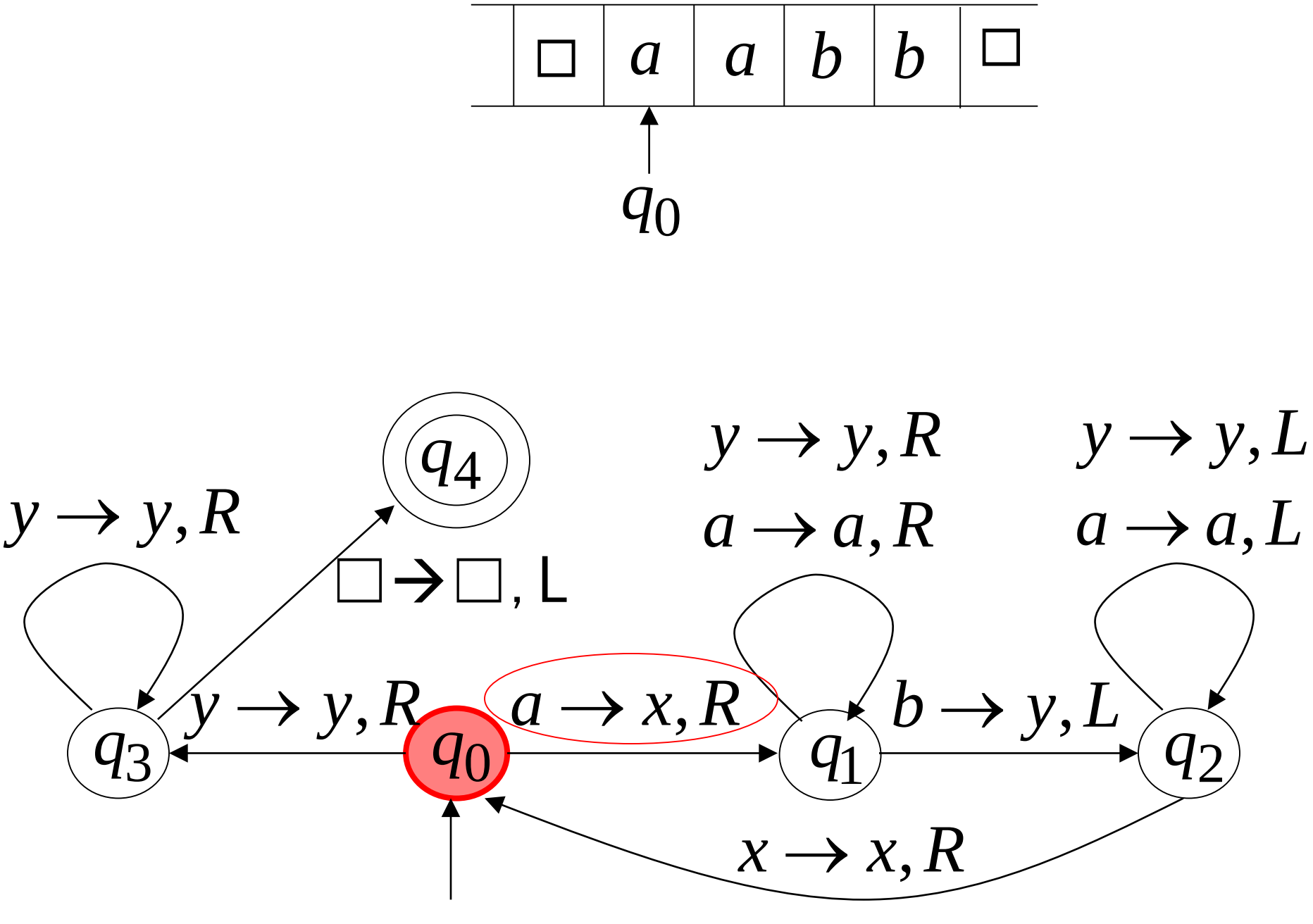
Because of the infinite loop:

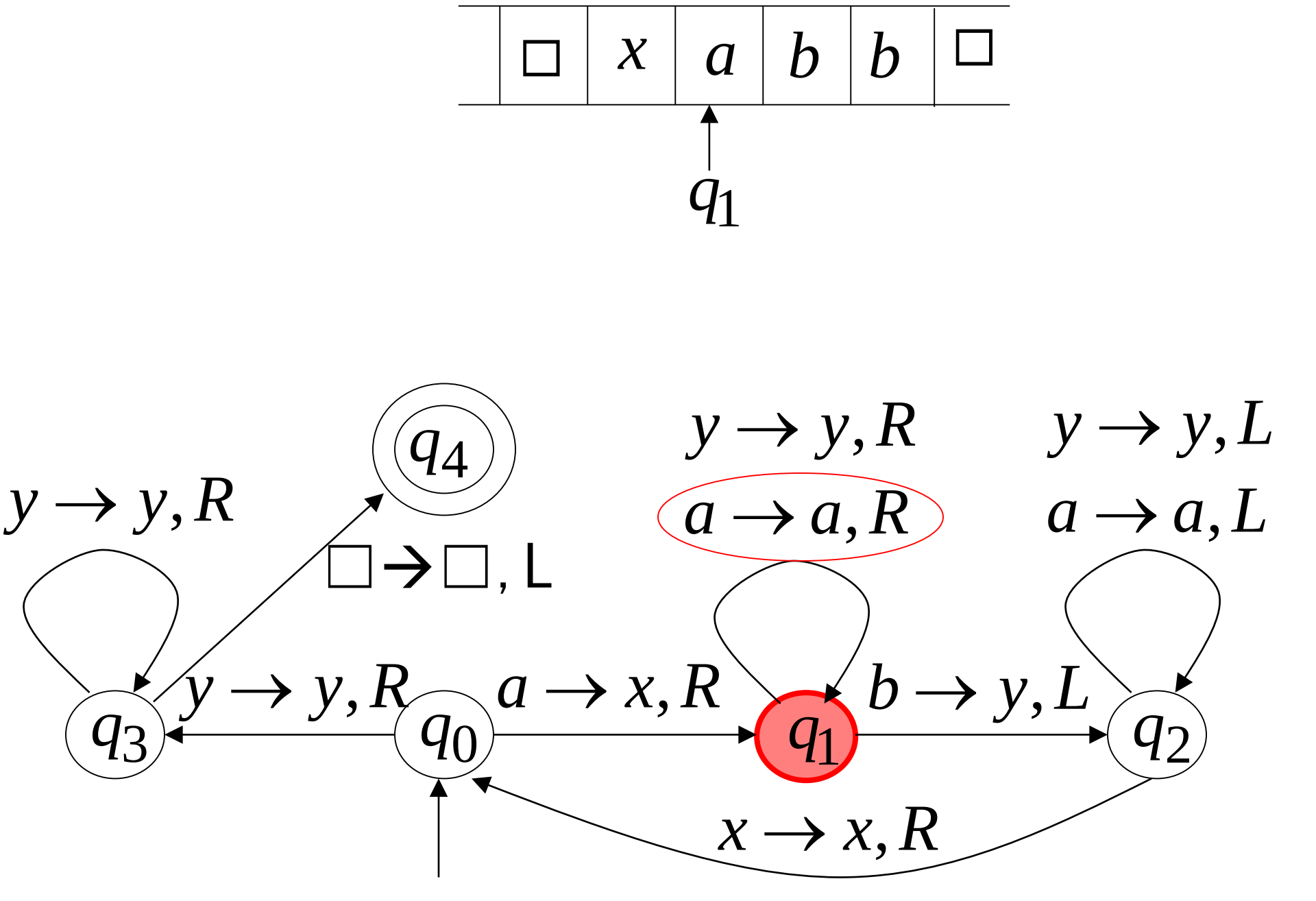
- The accepting state cannot be reached
- The machine never halts
- The input string is rejected

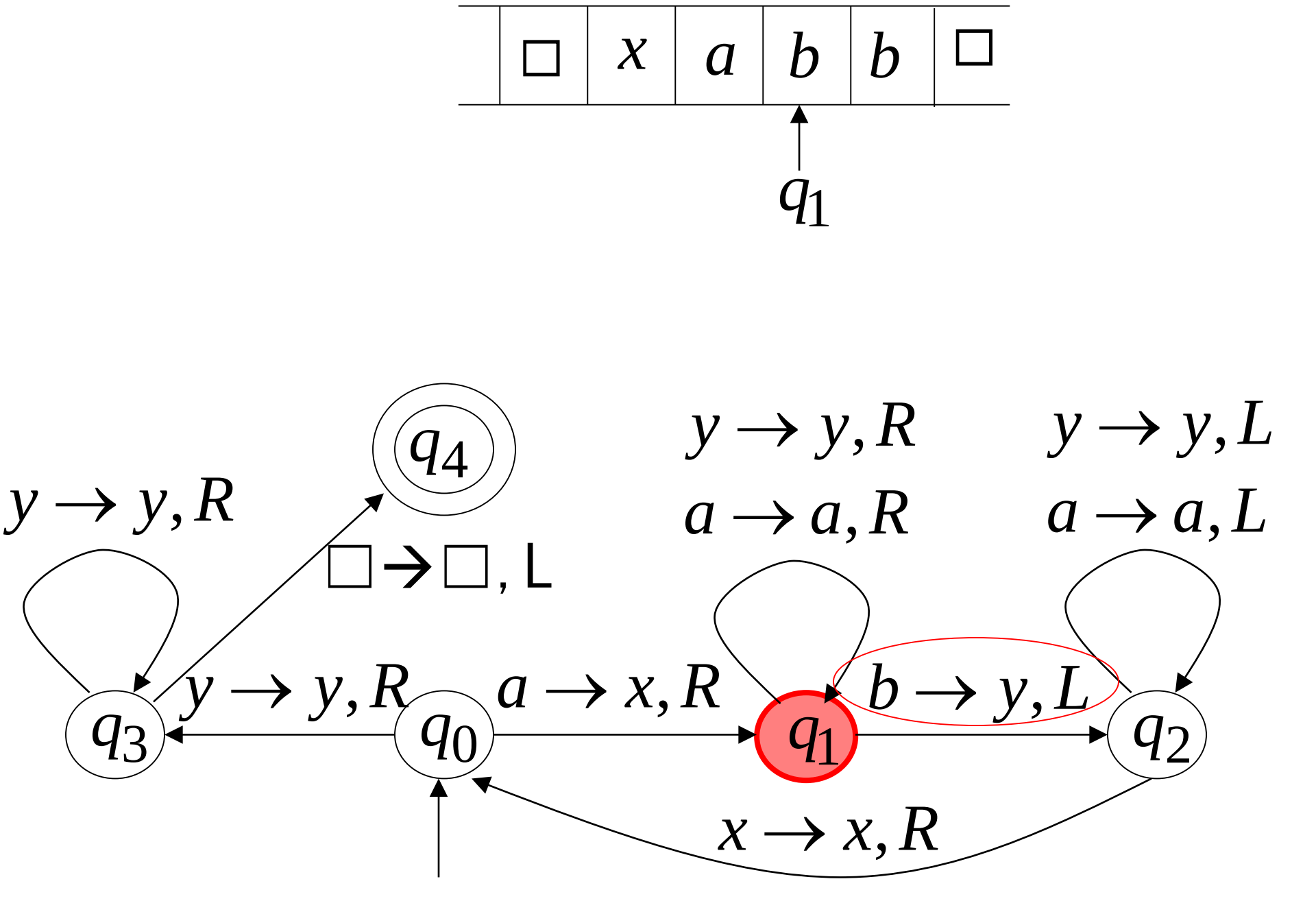
Turing Machine Example

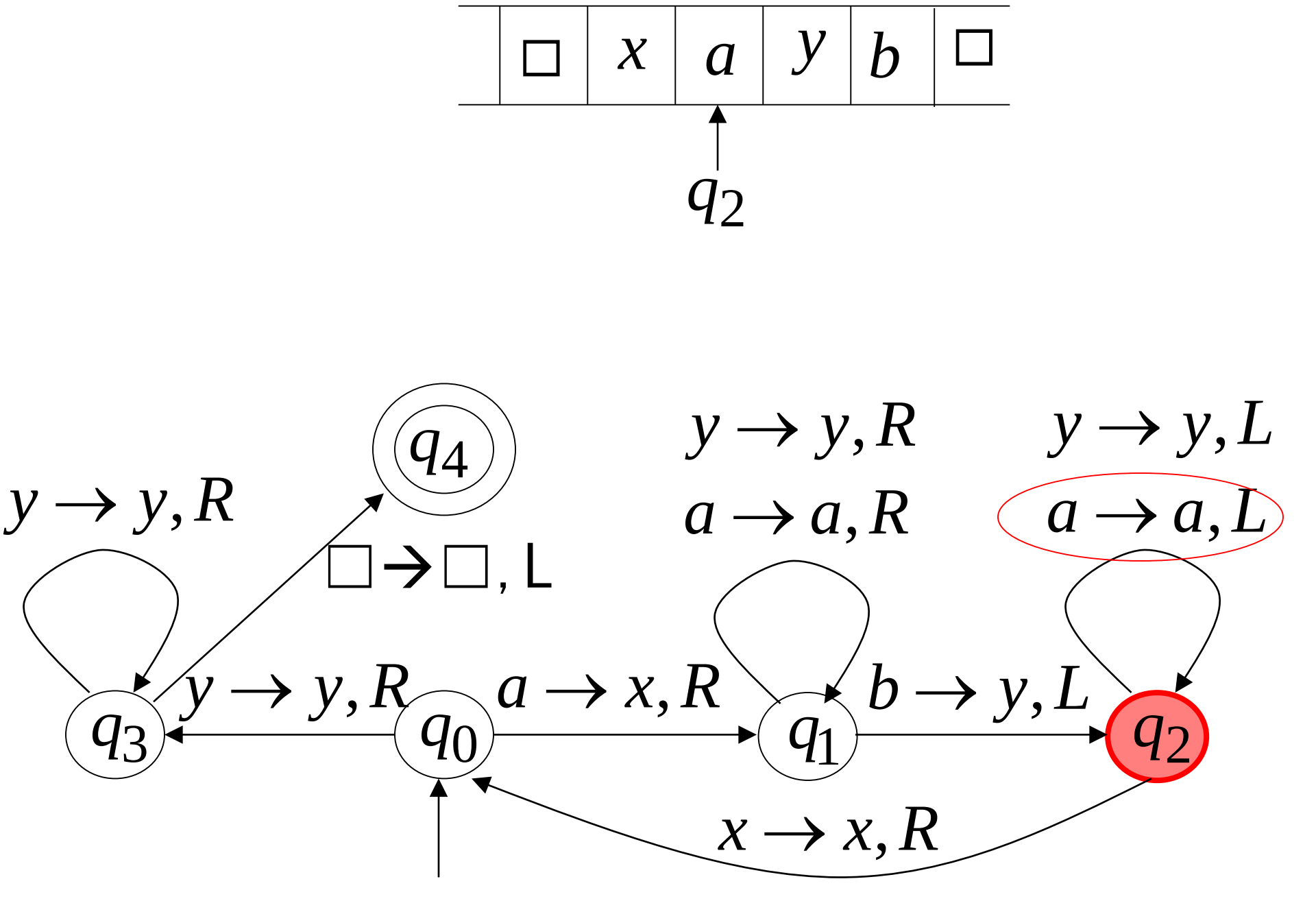
Turing machine for the language $\{a^n b^n\}$

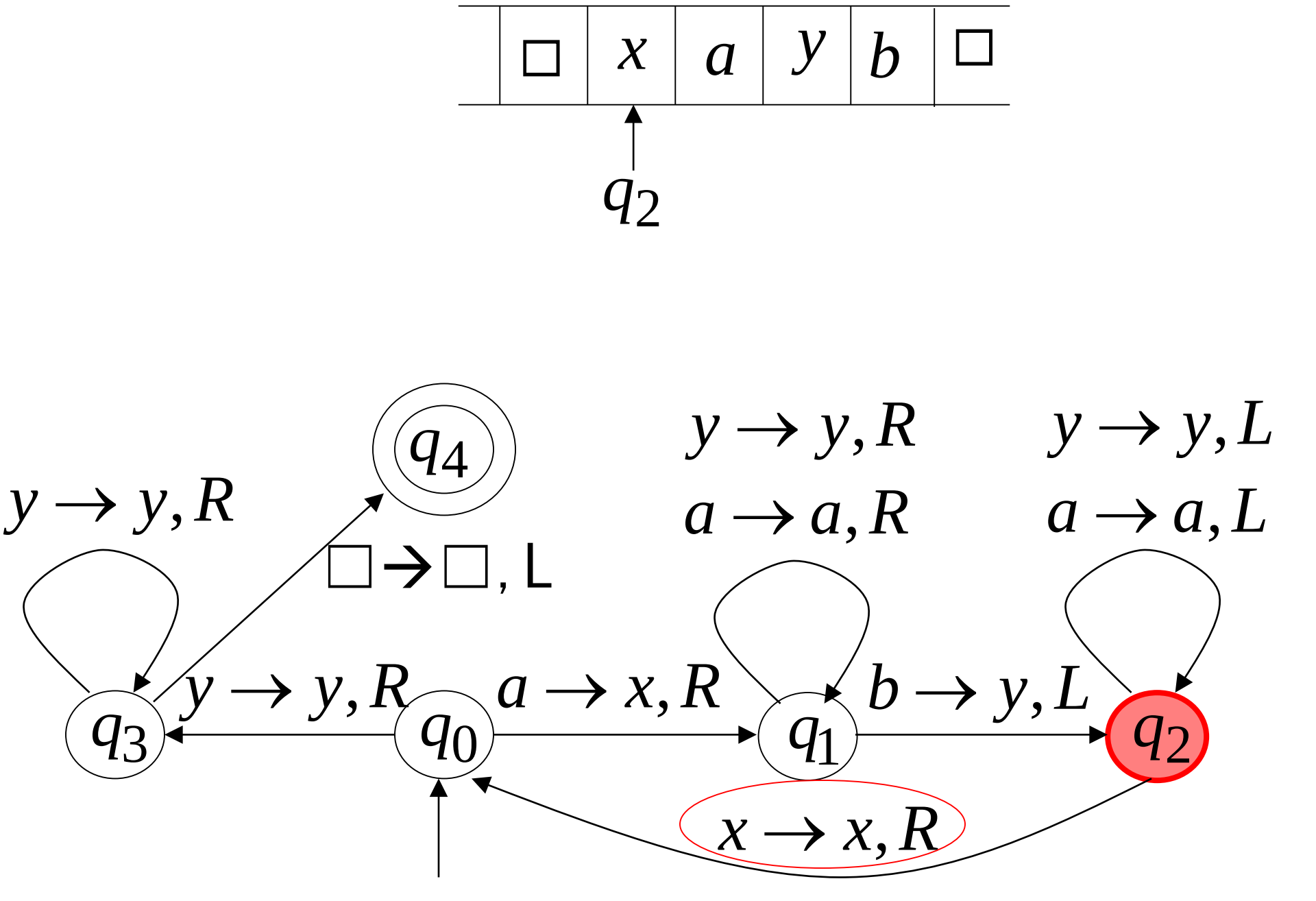


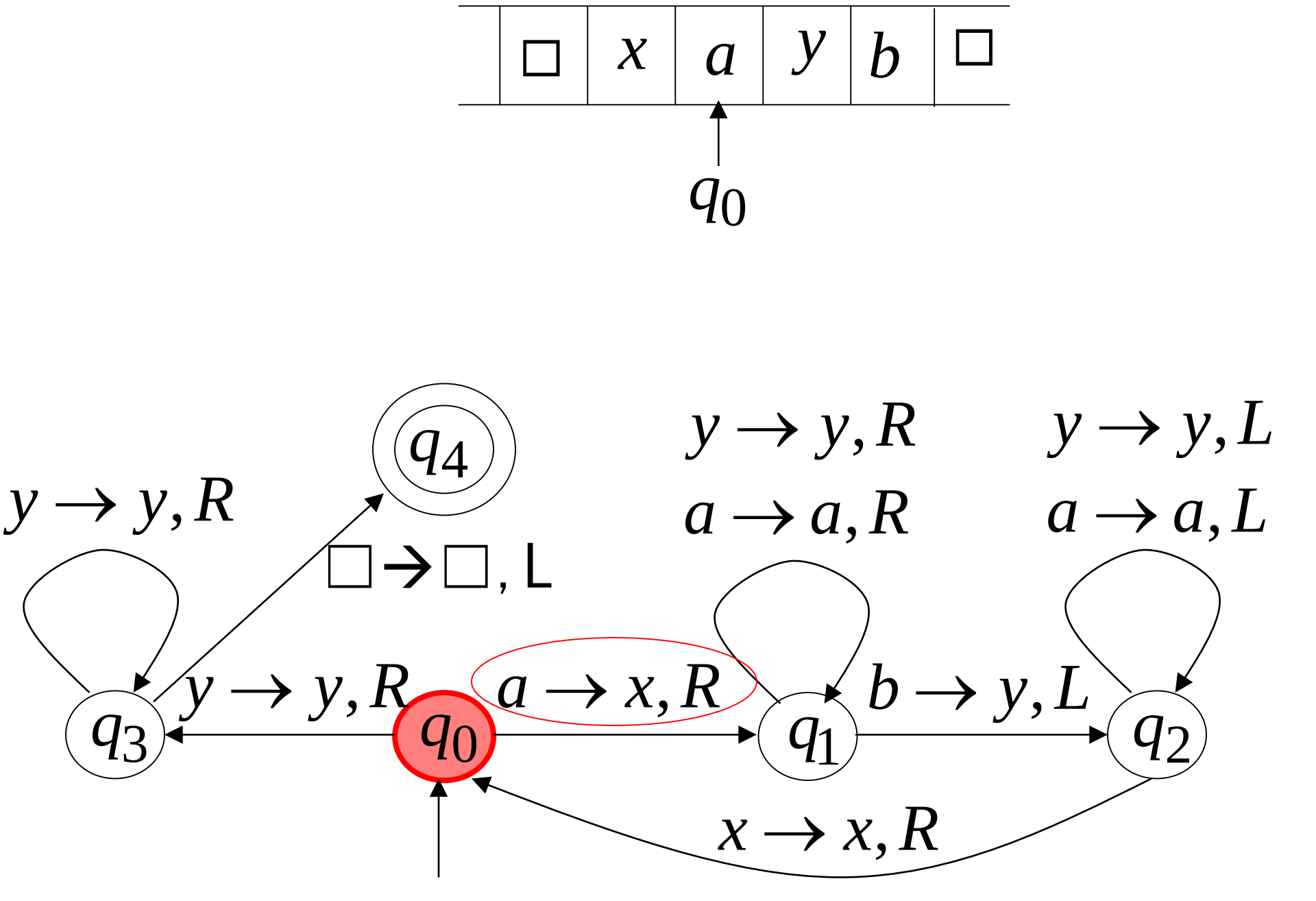

□ a a b b □
q_0
q_4
$y \to y, R$
$y \to y, R$
$a \to a, R$
$y \to y, L$
$a \to a, L$
□ → □, L
$y \to y, R$
$a \to x, R$
$b \to y, L$
q_3
q_0
q_1
q_2
$x \to x, R$

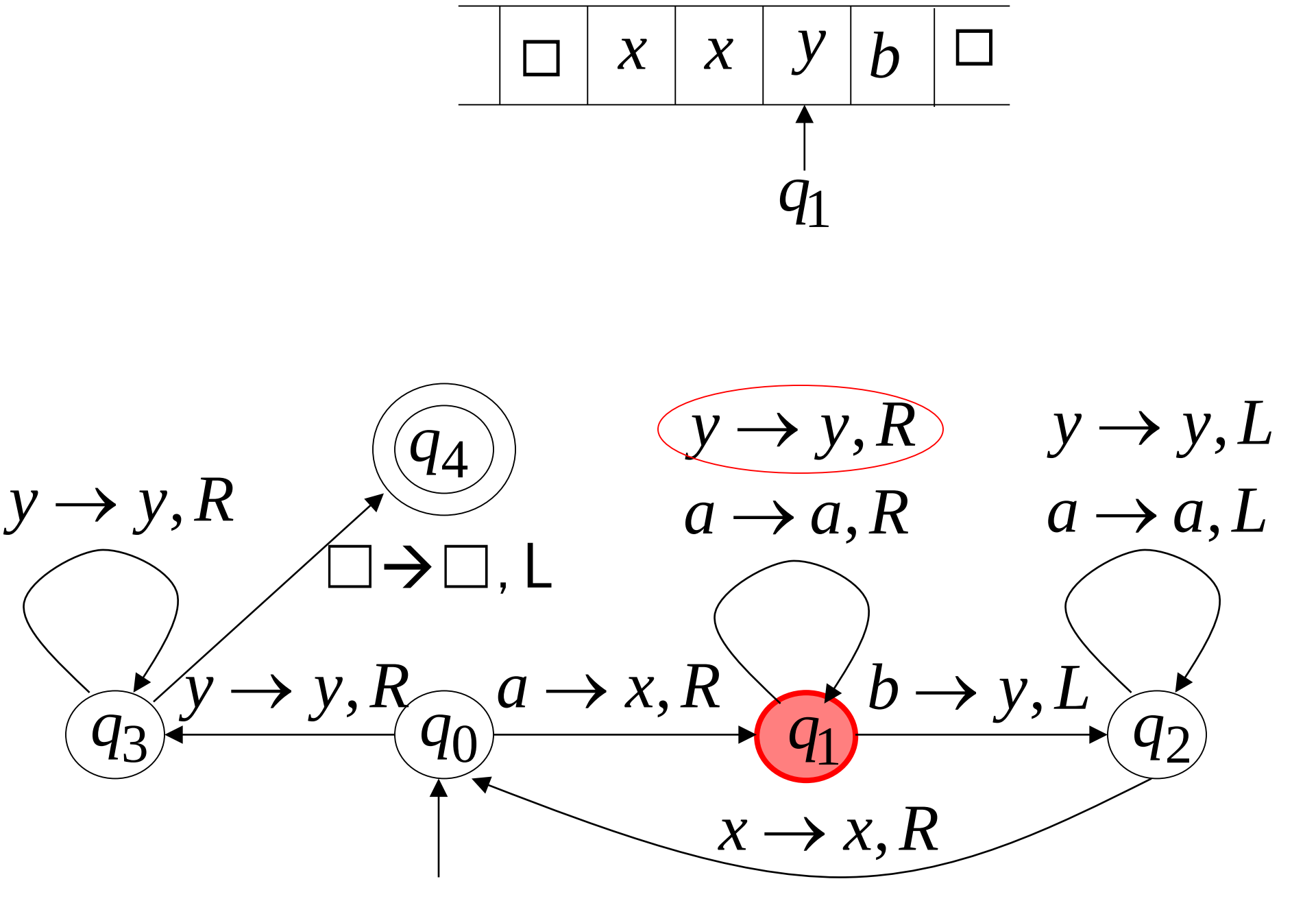

□ x a b b □
q_1
q_4
$y \to y, R$
$y \to y, R$
$y \to y, L$
$a \to a, R$
$a \to a, L$
□ → □, L
$y \to y, R$
$a \to x, R$
$b \to y, L$
q_3
q_0
q_1
q_2
$x \to x, R$

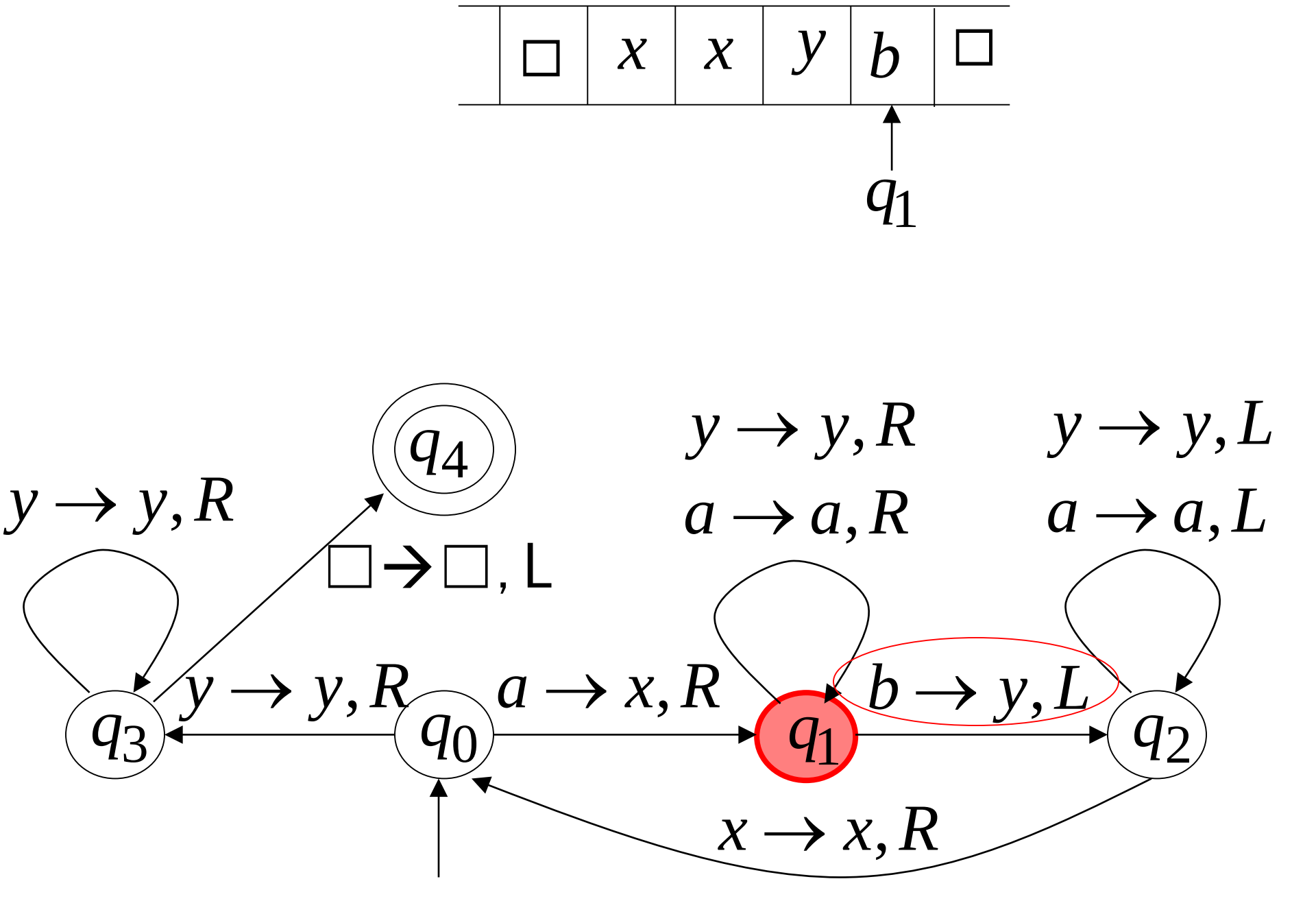

□ x a b b □
q_1
q_4
$y \to y, R$
$y \to y, R$
$a \to a, R$
$y \to y, L$
$a \to a, L$
□ → □, L
$y \to y, R$
$a \to x, R$
$b \to y, L$
q_3
q_0
q_1
q_2
$x \to x, R$

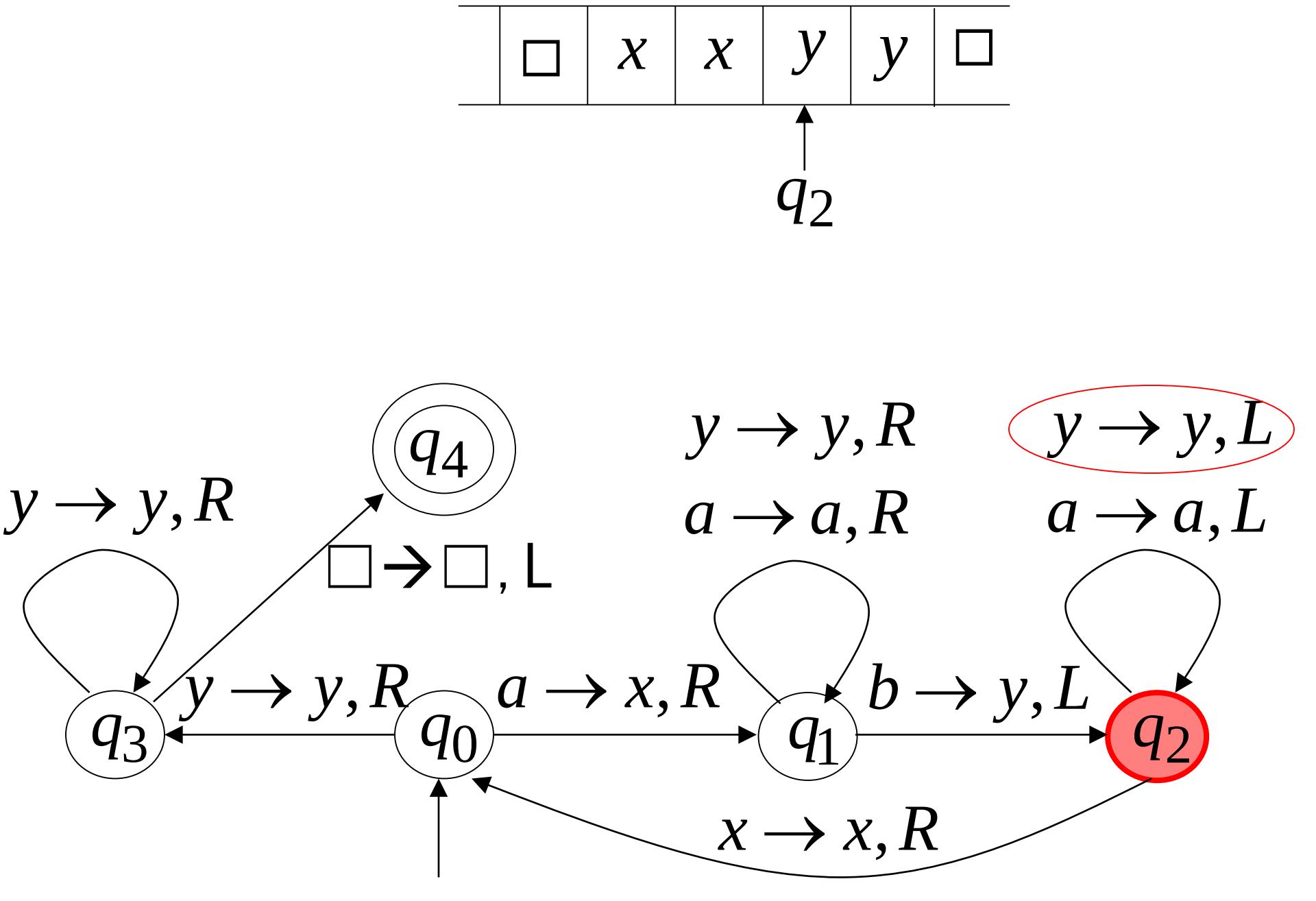

□ x a y b □
q_2
q_4
$y \to y, R$
$y \to y, R$
$a \to a, R$
$y \to y, L$
$a \to a, L$
□ → □, L
q_3
$y \to y, R$
q_0
$a \to x, R$
q_1
$b \to y, L$
q_2
$x \to x, R$

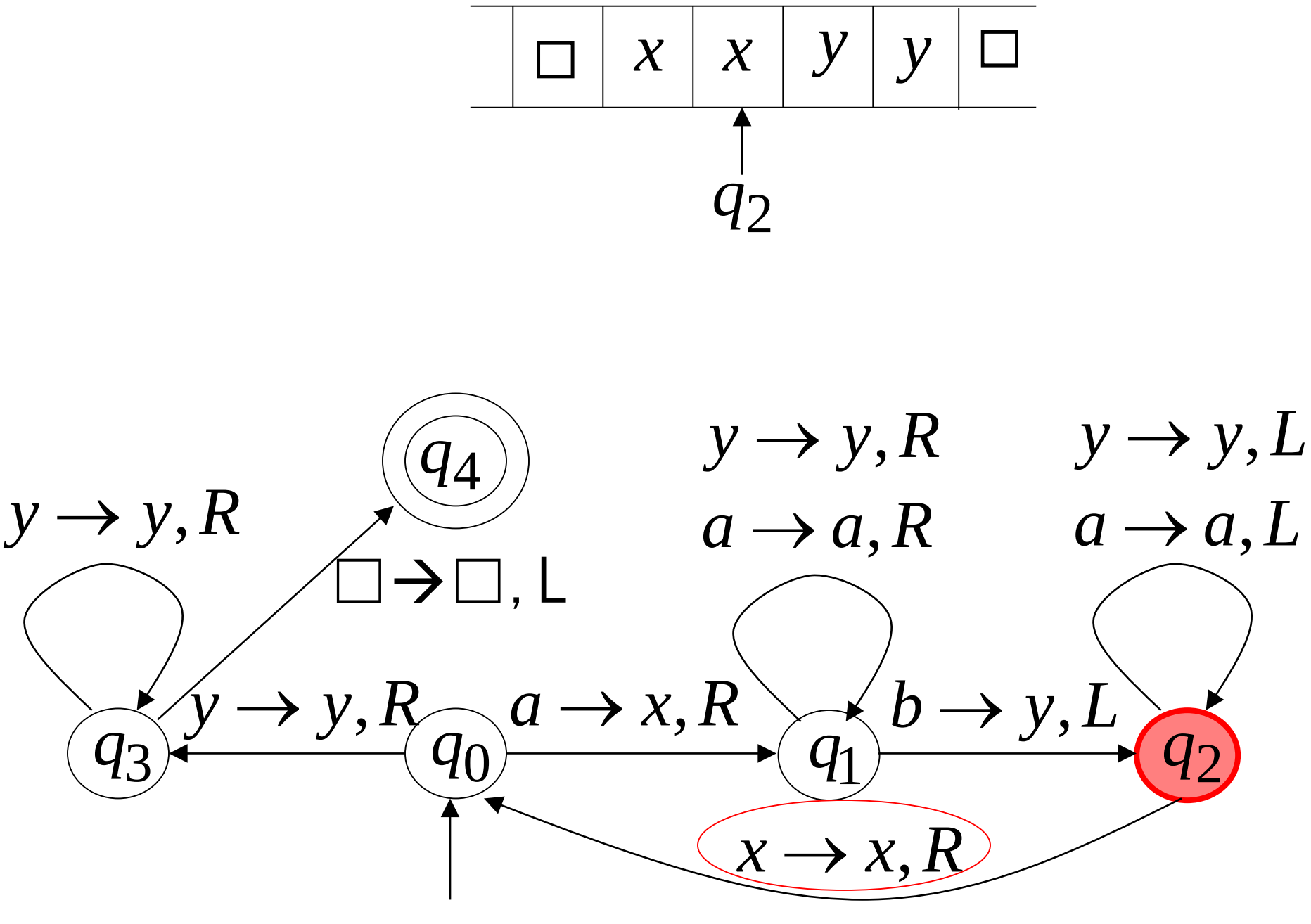

□ x a y b □
q_2
q_4
$y \to y, R$
$y \to y, R$
$a \to a, R$
$y \to y, L$
$a \to a, L$
□ → □, L
q_3
$y \to y, R$
q_0
$a \to x, R$
q_1
$b \to y, L$
q_2
$x \to x, R$

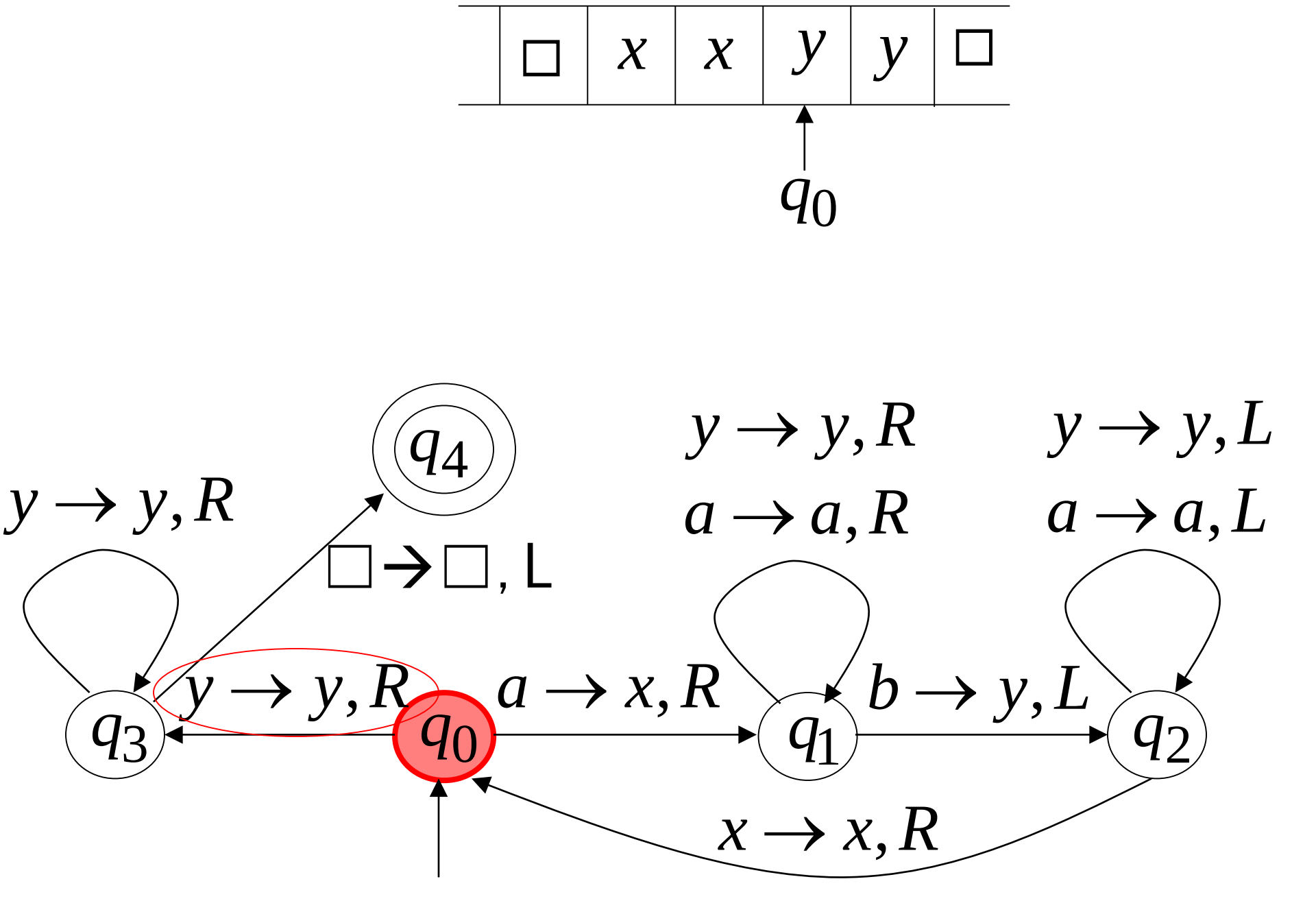

□ x a y b □
q_0
q_4
$y \to y, R$
$y \to y, R$
$a \to a, R$
$y \to y, L$
$a \to a, L$
□ → □, L
$y \to y, R$
$a \to x, R$
$b \to y, L$
q_3
q_0
q_1
q_2
$x \to x, R$

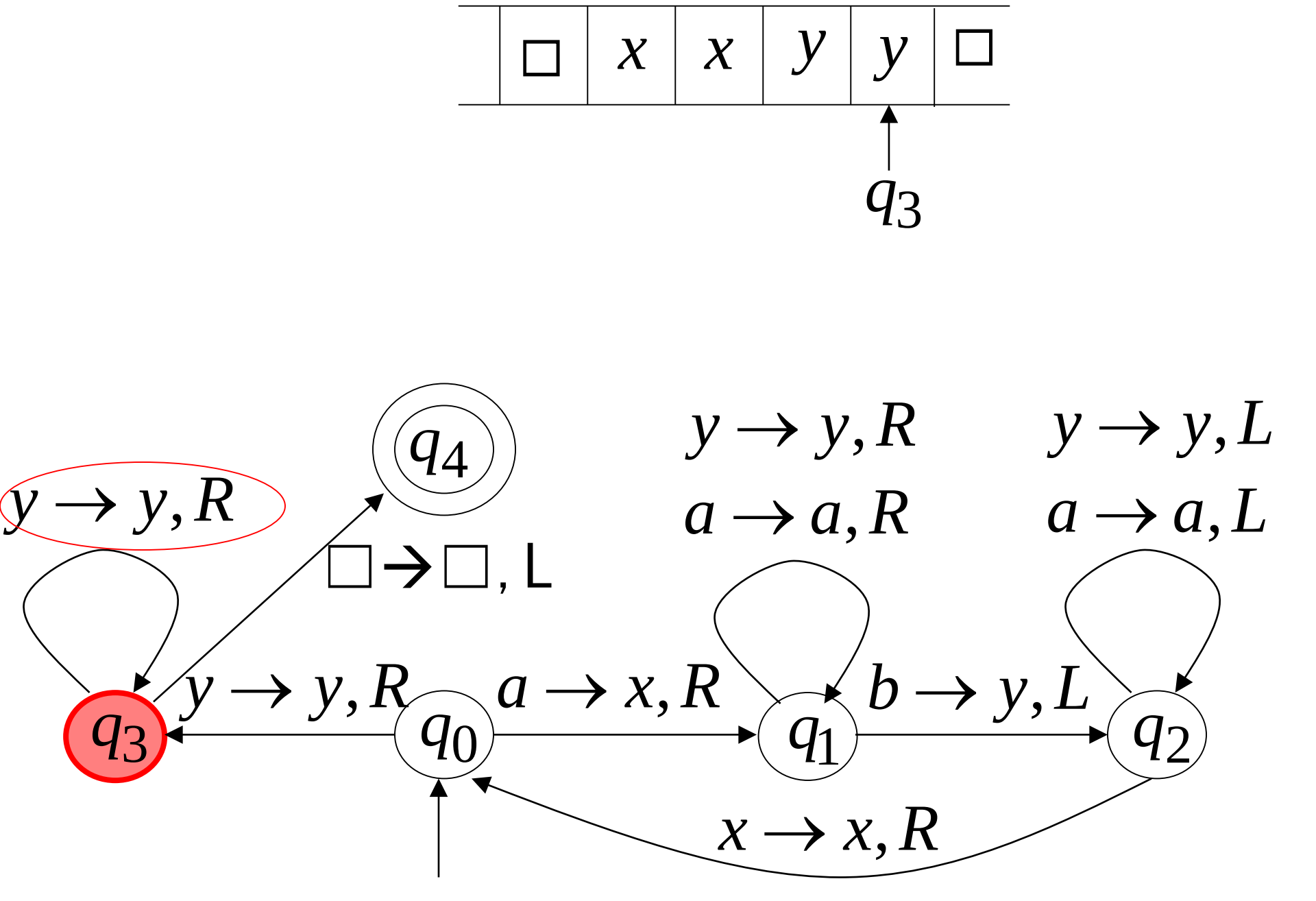

□ x x y b □
q_1
q_4
$y \to y, R$
$y \to y, R$
$a \to a, R$
$y \to y, L$
$a \to a, L$
□ → □, L
$y \to y, R$
$a \to x, R$
$b \to y, L$
q_3
q_0
q_1
q_2
$x \to x, R$

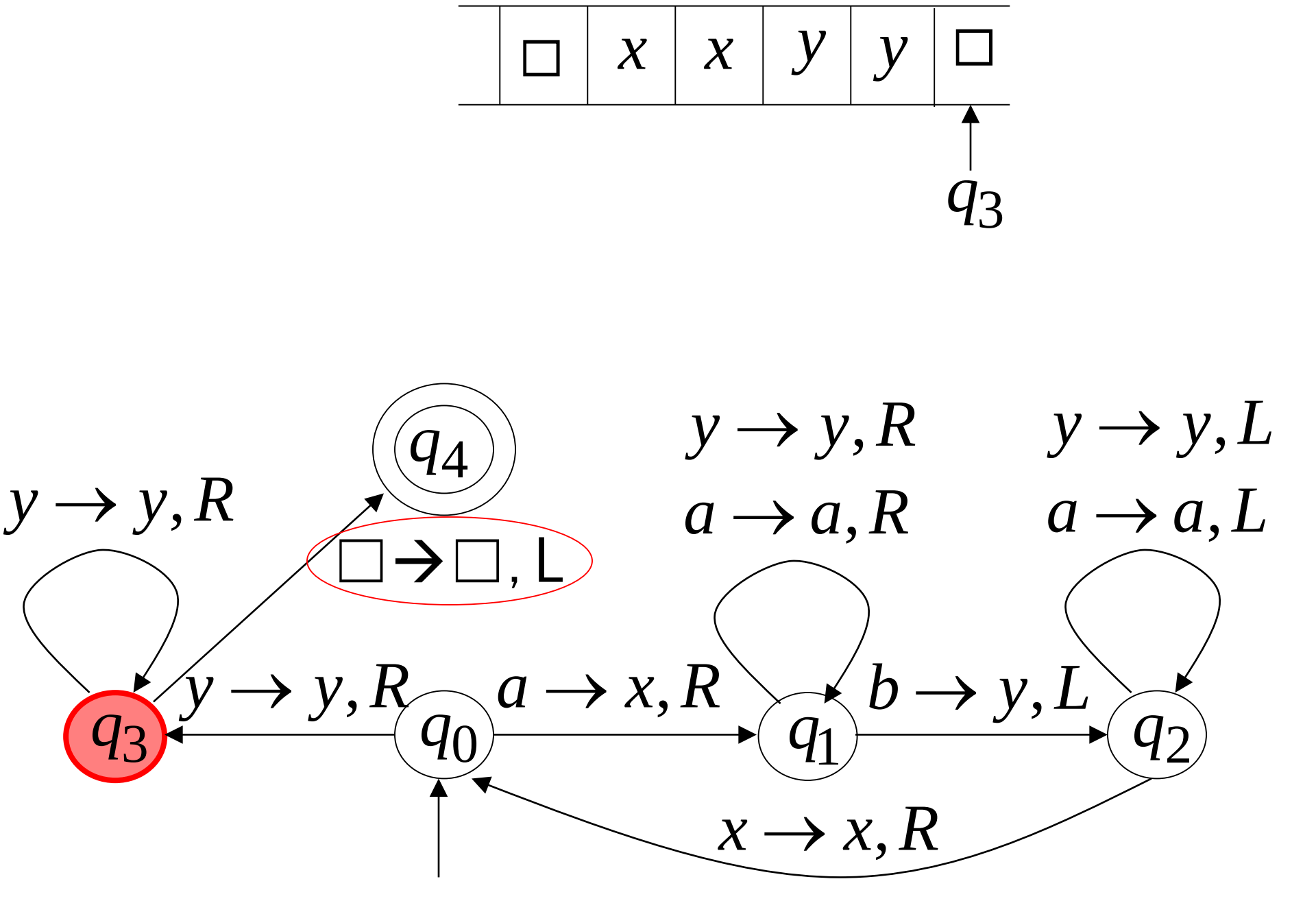

□ x x y b □
q_1
q_4
$y \to y, R$
$y \to y, R$
$a \to a, R$
$y \to y, L$
$a \to a, L$
□ → □, L
$y \to y, R$
$a \to x, R$
$b \to y, L$
q_3
q_0
q_1
q_2
$x \to x, R$

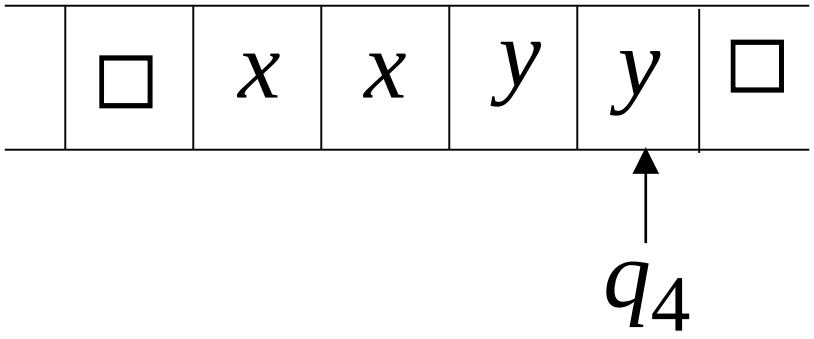

□ x x y y □
q_2
q_4
$y \to y, R$
$y \to y, R$
$a \to a, R$
$y \to y, L$
$a \to a, L$
□ → □, L
q_3
$y \to y, R$
q_0
$a \to x, R$
q_1
$b \to y, L$
q_2
$x \to x, R$


□ x x y y □
q_2
q_4
$y \to y, R$
$y \to y, R$
$a \to a, R$
$y \to y, L$
$a \to a, L$
□ → □, L
q_3
$y \to y, R$
q_0
$a \to x, R$
q_1
$b \to y, L$
q_2
$x \to x, R$

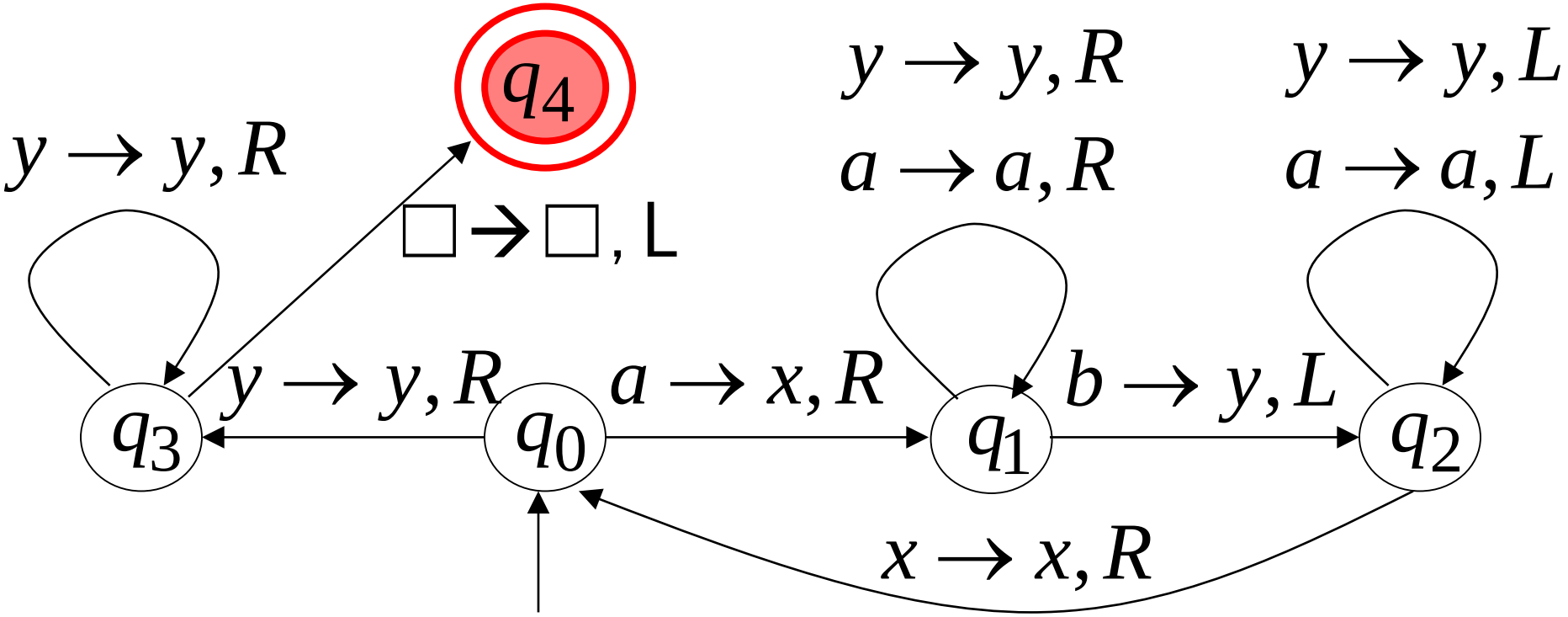

□ x x y y □
q_0
q_4
$y \to y, R$
$y \to y, R$
$a \to a, R$
$y \to y, L$
$a \to a, L$
□ → □, L
$y \to y, R$
$a \to x, R$
$b \to y, L$
q_3
q_0
q_1
q_2
$x \to x, R$


□ x x y y □
q_3
q_4
$y \to y, R$
$a \to a, R$
$y \to y, L$
$a \to a, L$
$y \to y, R$
□ → □, L
$y \to y, R$
$a \to x, R$
$b \to y, L$
q_3
q_0
q_1
q_2
$x \to x, R$


□ x x y y □
q_3
q_4
$y \to y, R$
$a \to a, R$
$y \to y, L$
$a \to a, L$
$y \to y, R$
□ → □, L
$y \to y, R$
$a \to x, R$
$b \to y, L$
q_3
q_0
q_1
q_2
$x \to x, R$


□
x
x
y
y
□
q_4

Halt & Accept


q_4
$y \to y, R$
$a \to a, R$
$y \to y, L$
$a \to a, L$
$y \to y, R$
□ → □, L
$y \to y, R$
$a \to x, R$
$b \to y, L$
$x \to x, R$
q_3
q_0
q_1
q_2

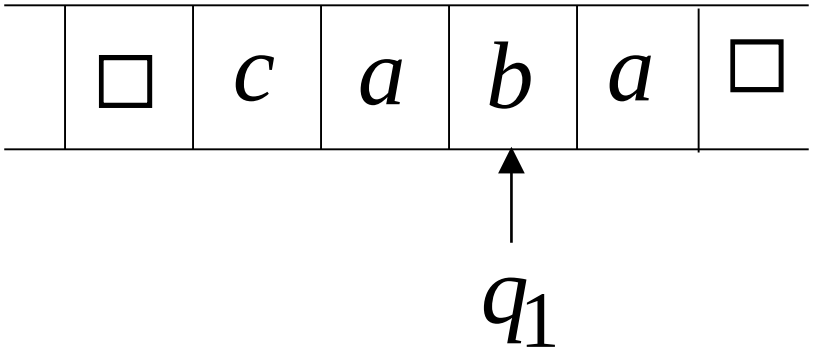
Observation:

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

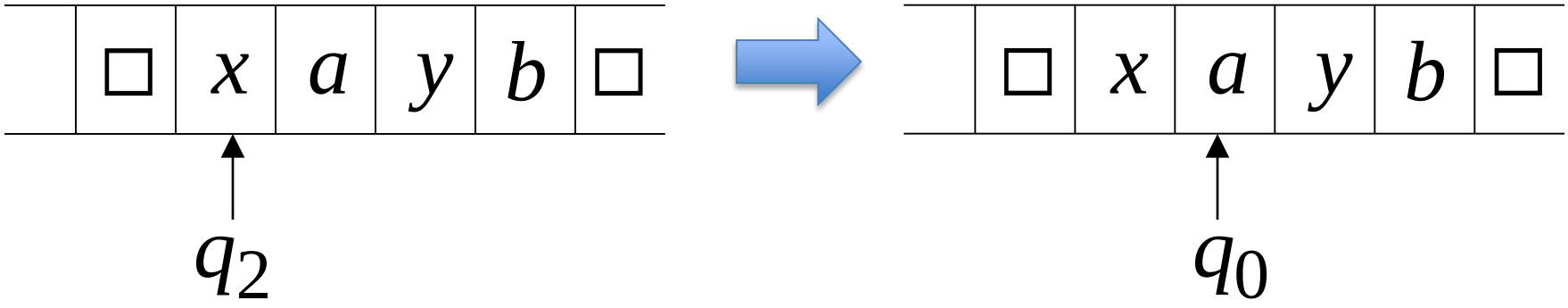
we can easily construct
a machine for the language $\{a^n b^n c^n\}$

Instantaneous Description

Configuration

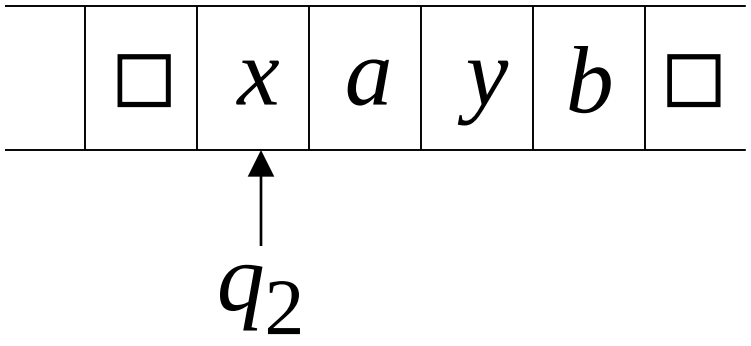


Instantaneous description: $ca\ q_1\ ba$

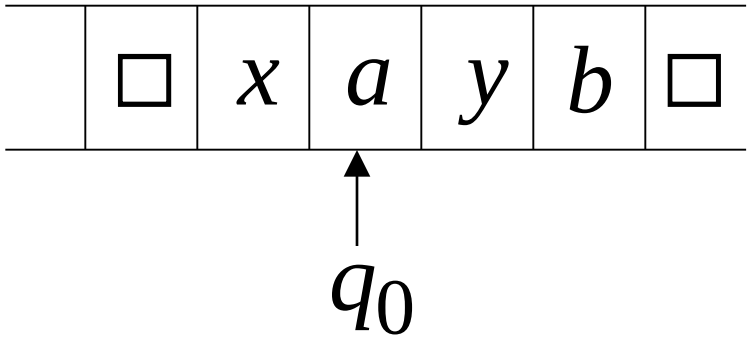


A Move: $q_2\ xayb \vdash x\ q_0\ ayb$

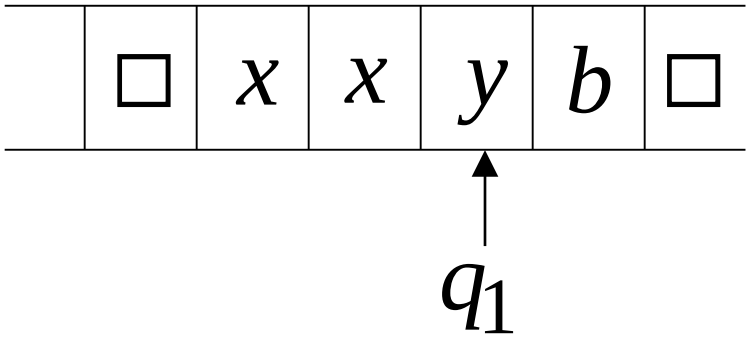
Time 4



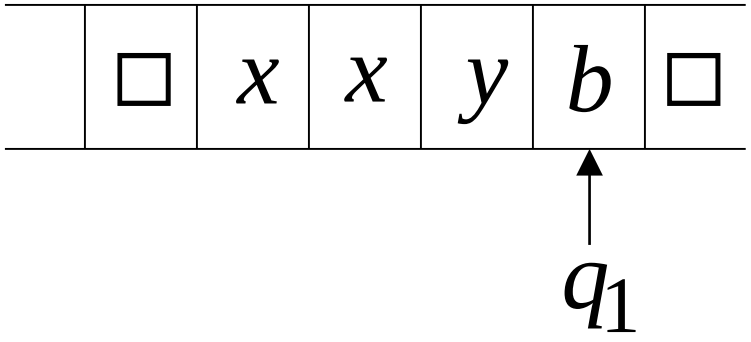
Time 5



Time 6



Time 7



A computation

q_2 xayb ⊢ x q_0 ayb ⊢ xx q_1 yb ⊢ xxy q_1 b

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

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

The Accepted Language

For any Turing Machine M, the language accepted by M is

$L(M) = \{ w \in \Sigma^+ : q_0 w \vdash x_1\, q_f\, x_2$ for some $q_f \in F$, $x_1, x_2 \in \Gamma^* \}$

If a language L is accepted by a Turing machine M then we say that L is:

Turing Recognizable

Other names used:

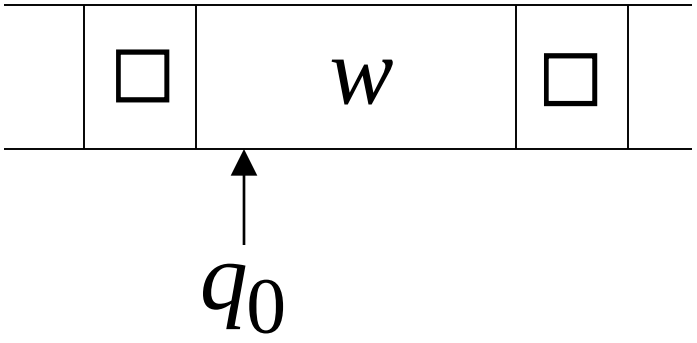
- Turing Acceptable
- Recursively Enumerable

Computing Functions with Turing Machines

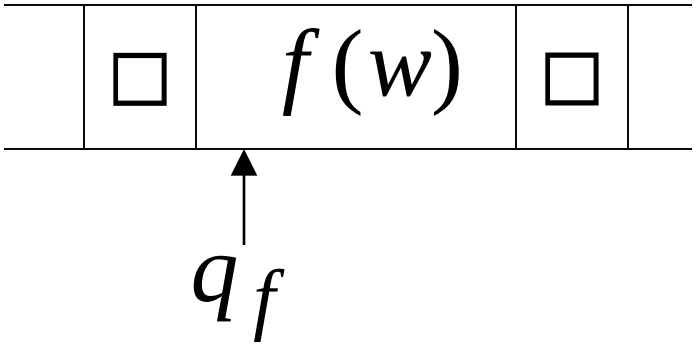
Definition:

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:

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

Initial Configuration

Final Configuration

For all $w \in D$ Domain

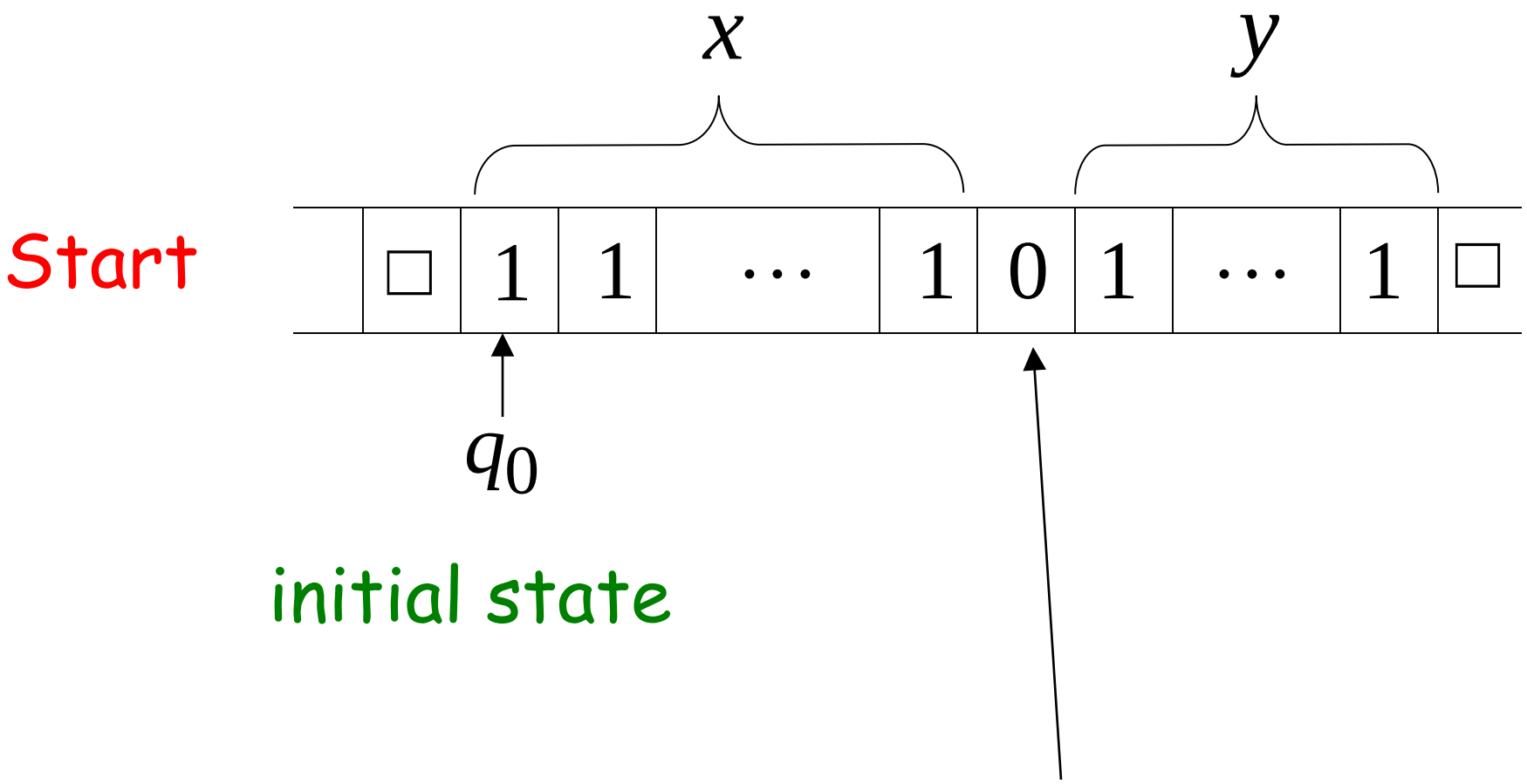
Example

Design a Turing machine that computes:

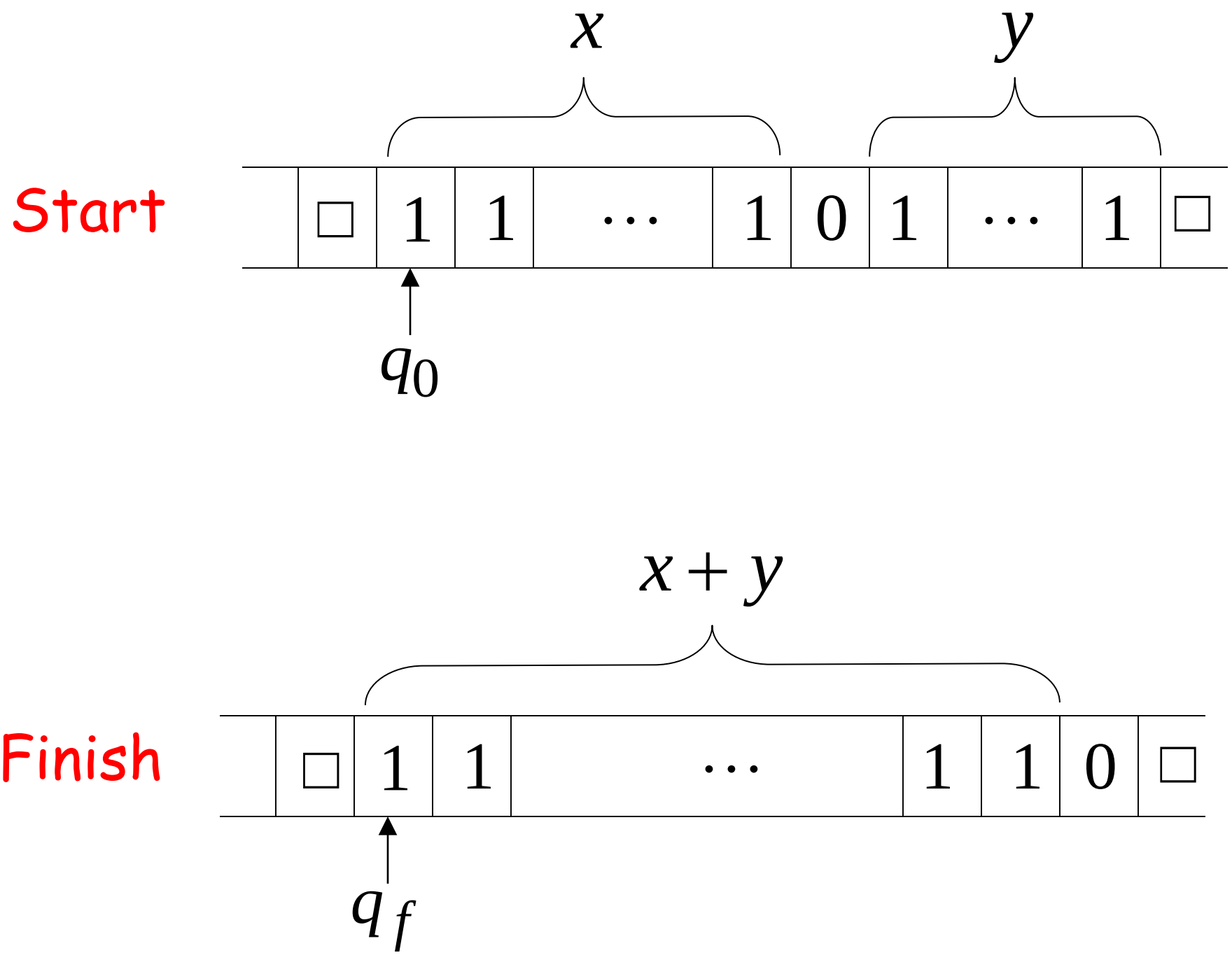
$$f(x, y) = x + y$$

- How to represent x and y on the tape?
  For simplicity we will use unary notation

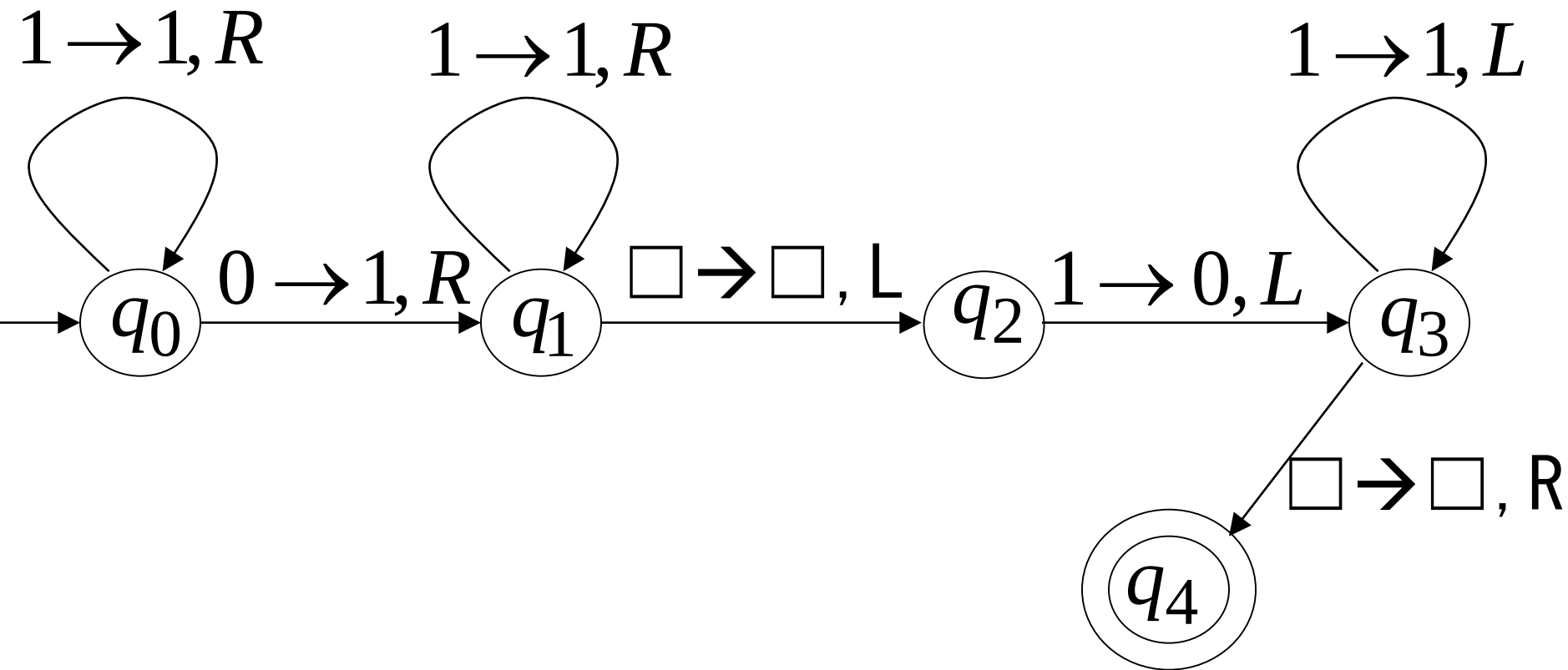
  For example: x = 5 11111

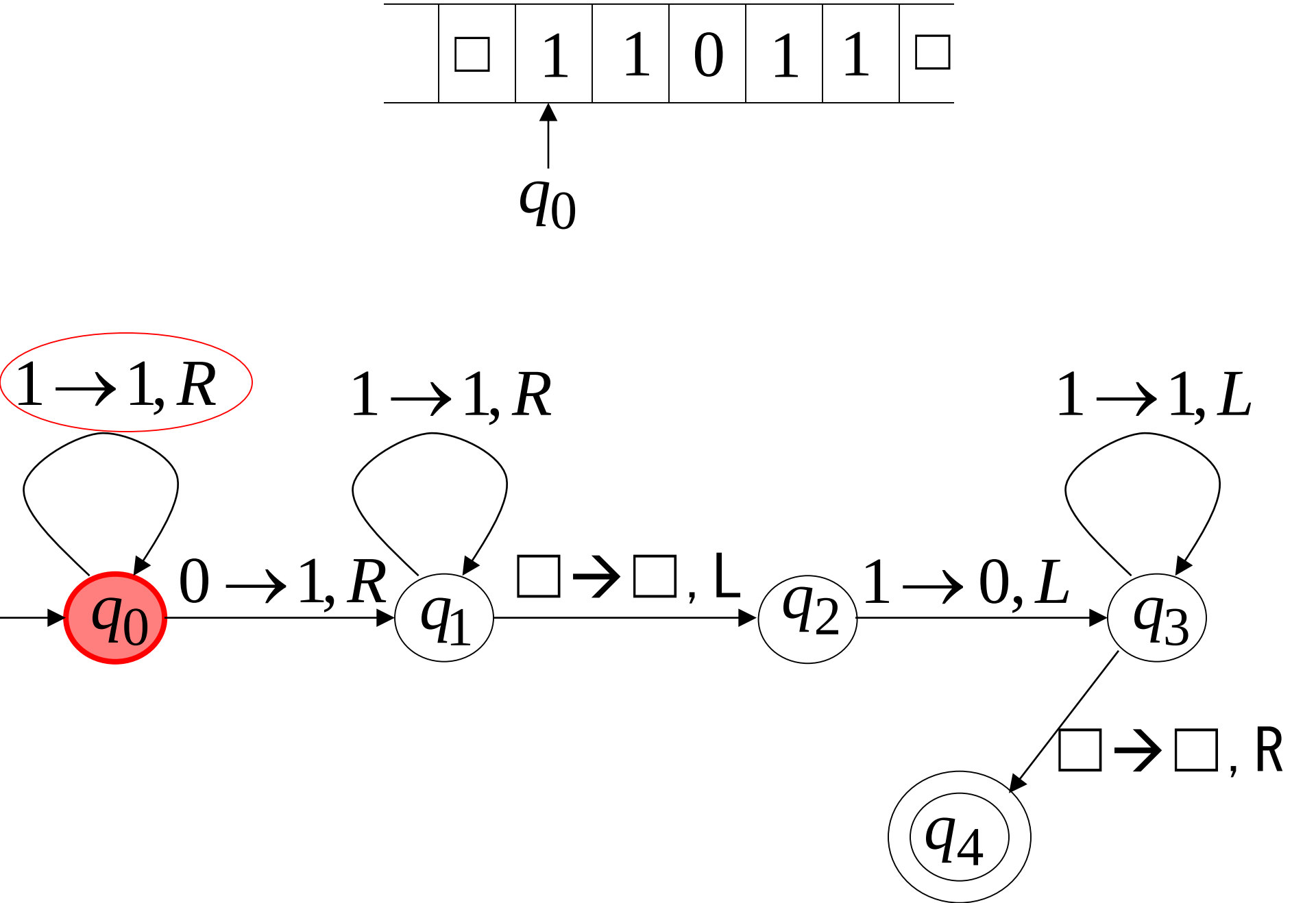


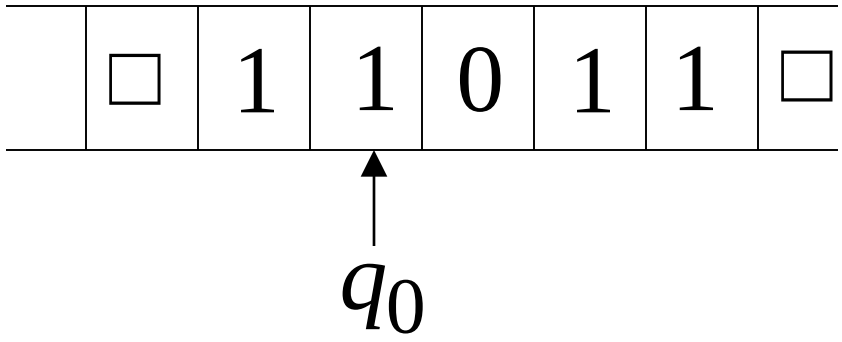
The 0 is the delimiter that separates the two numbers

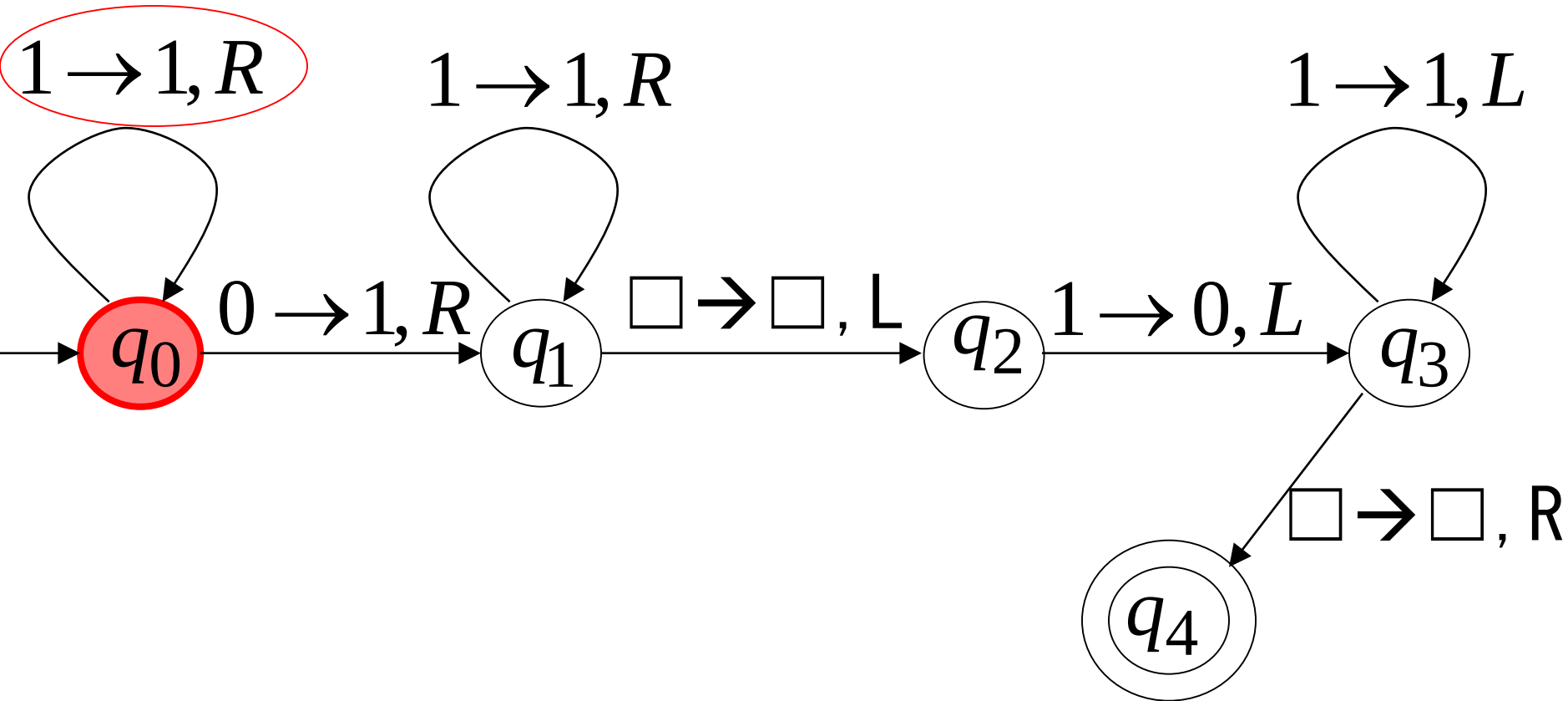

Start
x
y
□ 1 1 ⋯ 1 0 1 ⋯ 1 □
q_0
Finish
$x+y$
□ 1 1 ⋯ 1 1 0 □
q_f

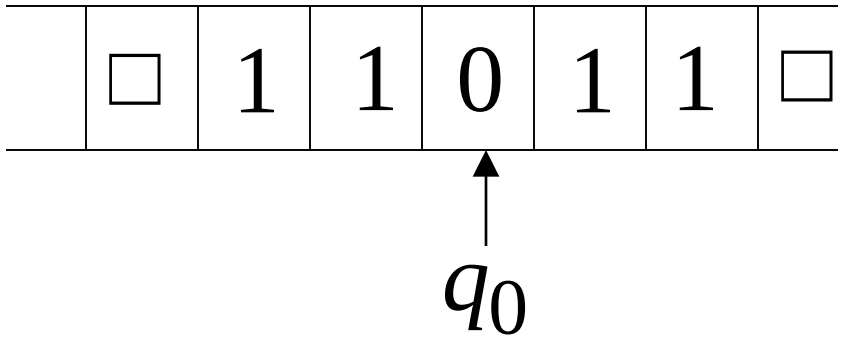
Turing machine for function $f(x, y) = x + y$

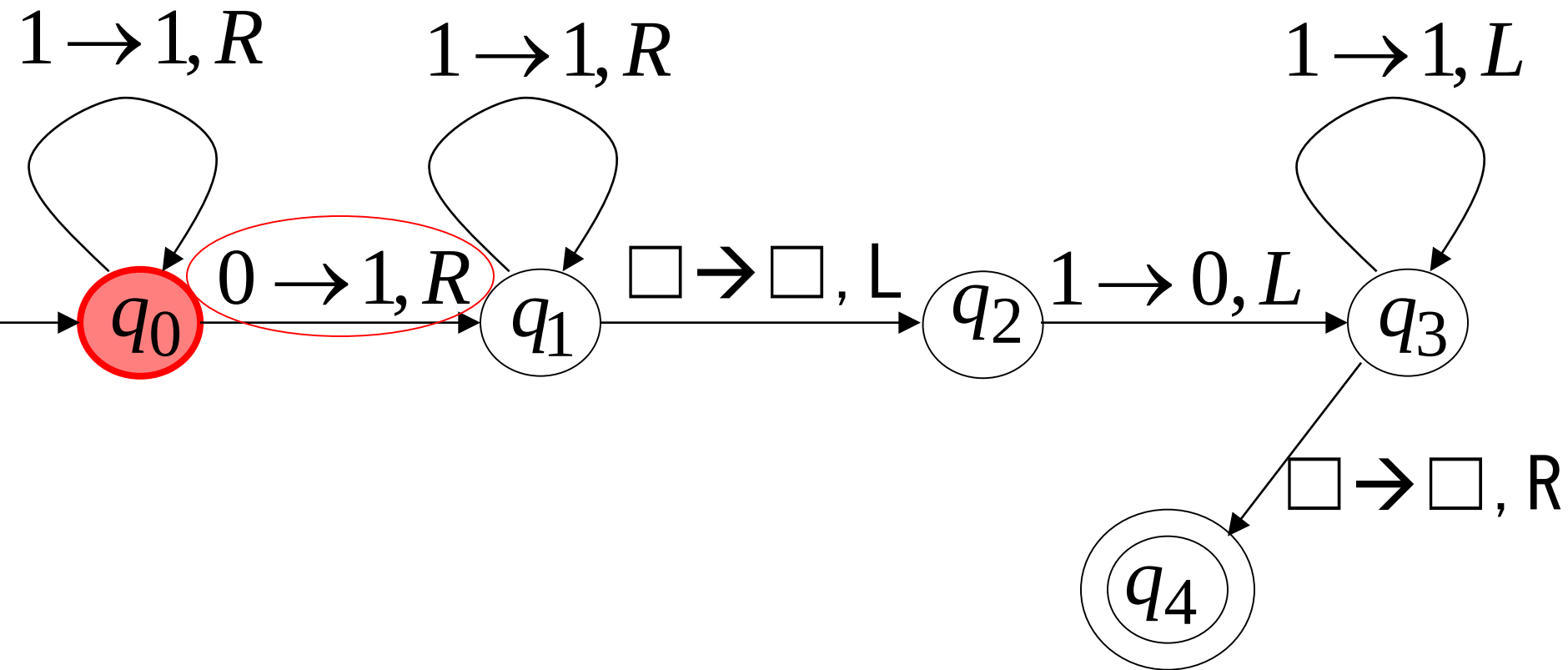


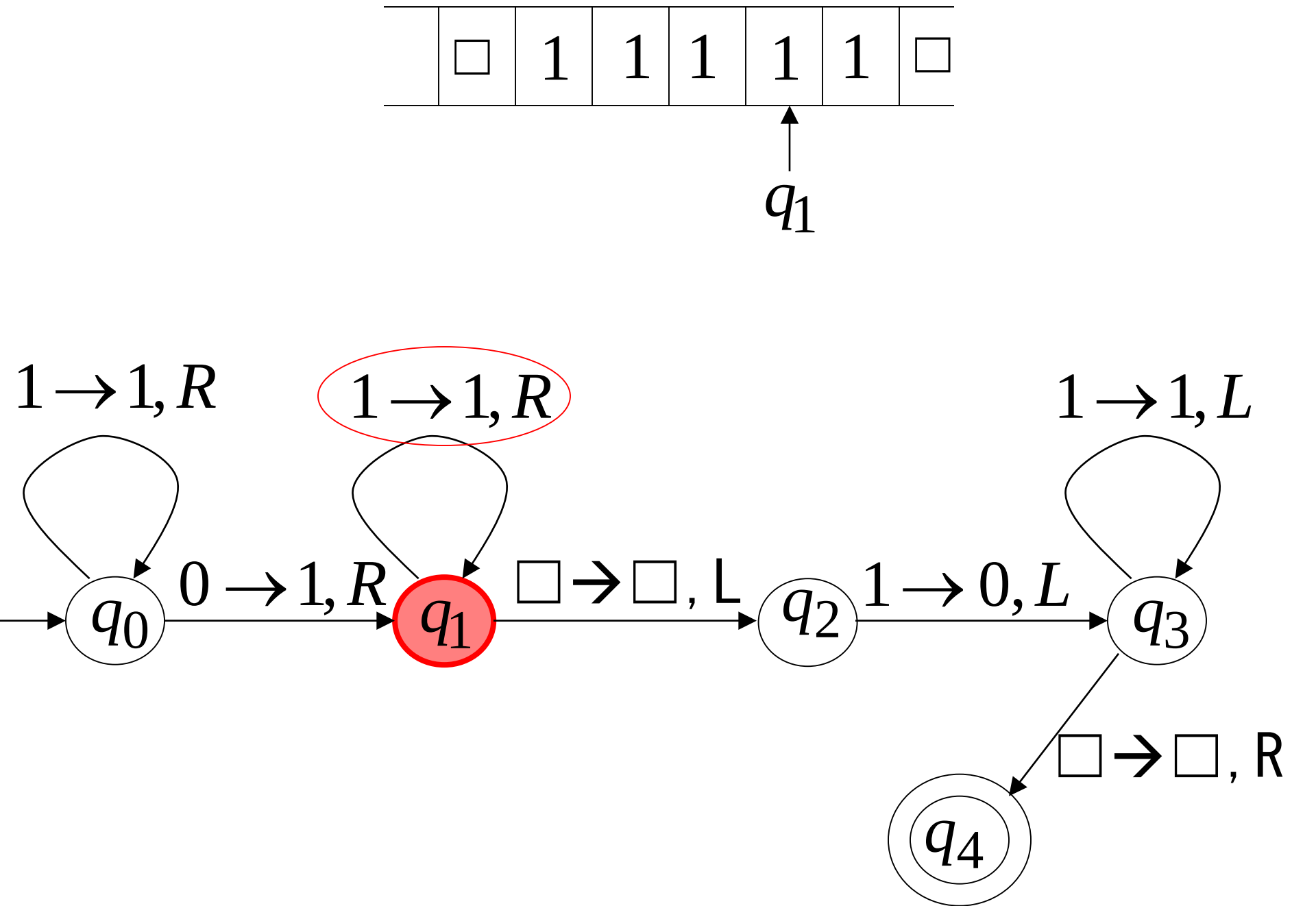

□ 1 1 0 1 1 □
q_0
$1 \to 1, R$
$1 \to 1, R$
$1 \to 1, L$
$0 \to 1, R$
□ → □, L
$1 \to 0, L$
□ → □, R
q_0
q_1
q_2
q_3
q_4

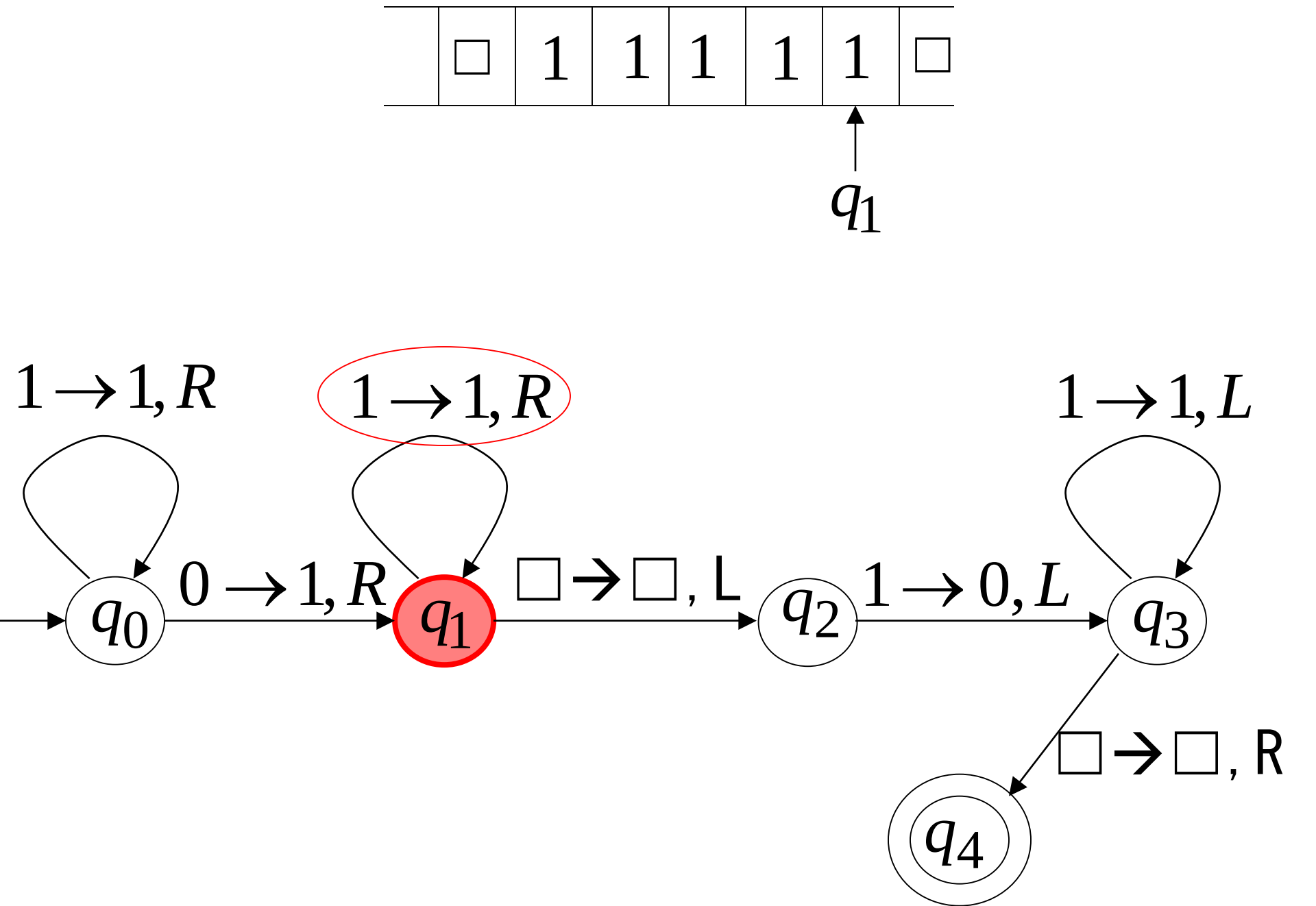

□ 1 1 0 1 1 □
q_0

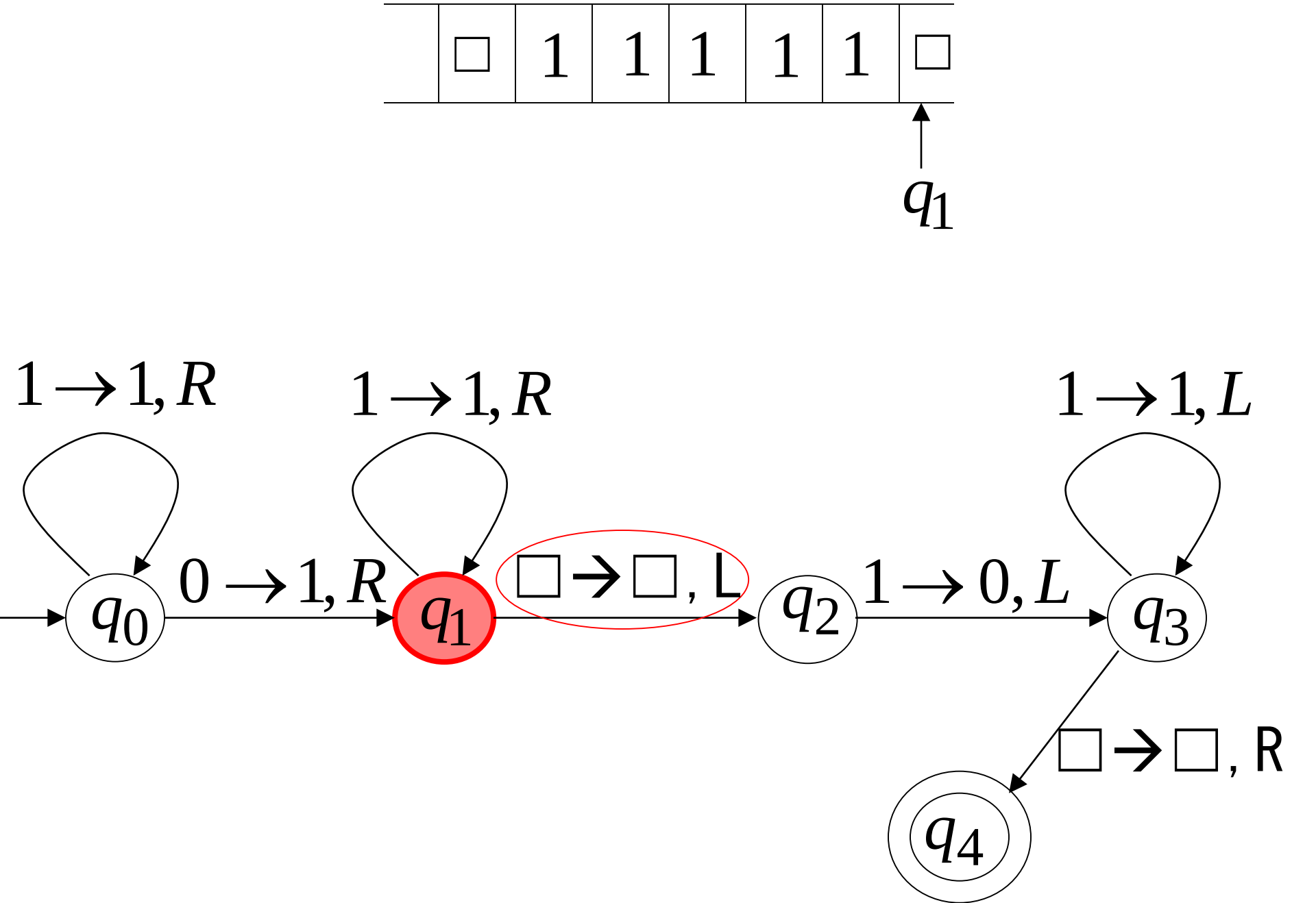

$1 \to 1, R$
$1 \to 1, R$
$1 \to 1, L$
$0 \to 1, R$
□ → □, L
$1 \to 0, L$
□ → □, R
q_0
q_1
q_2
q_3
q_4

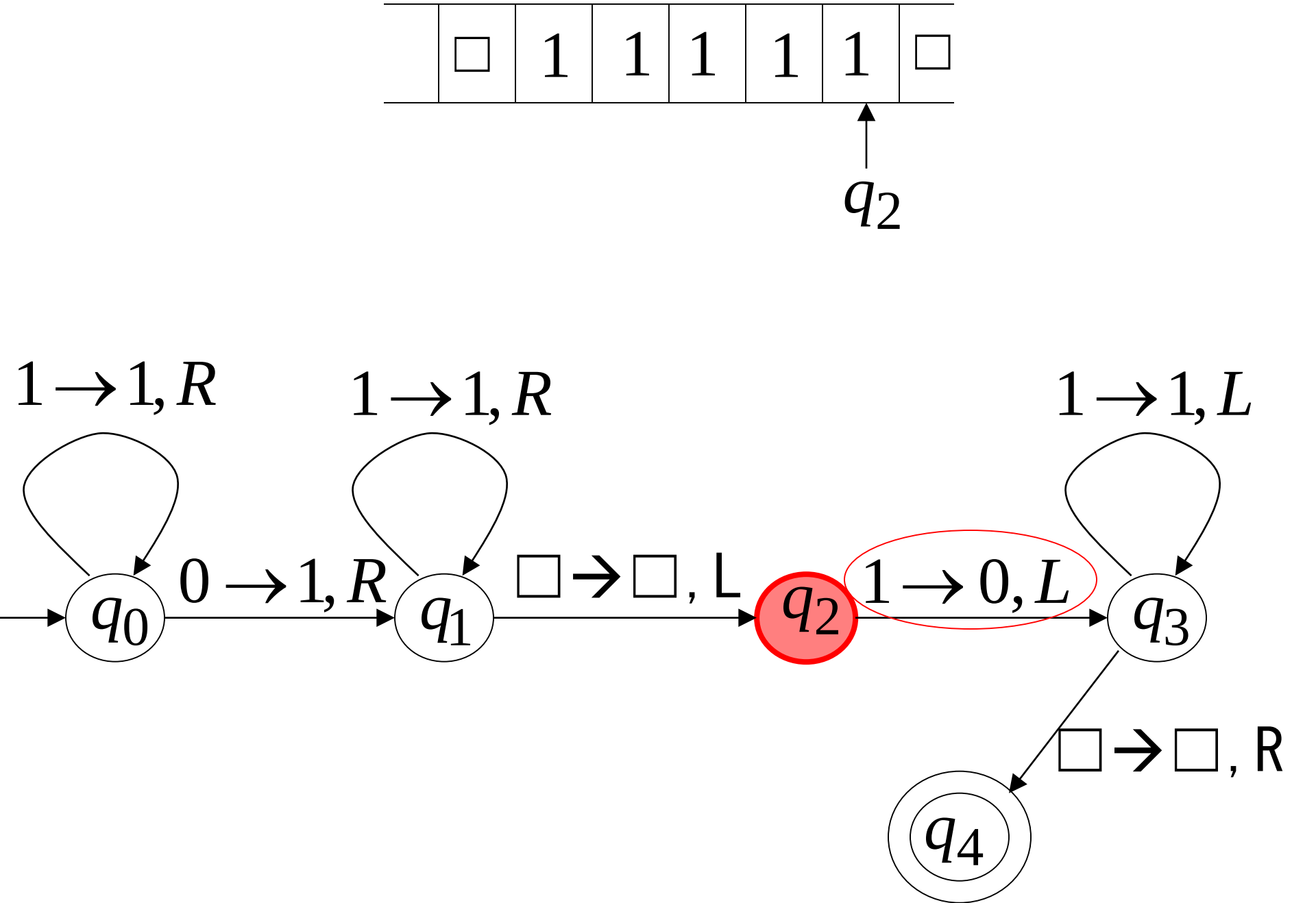

□ 1 1 0 1 1 □
q_0

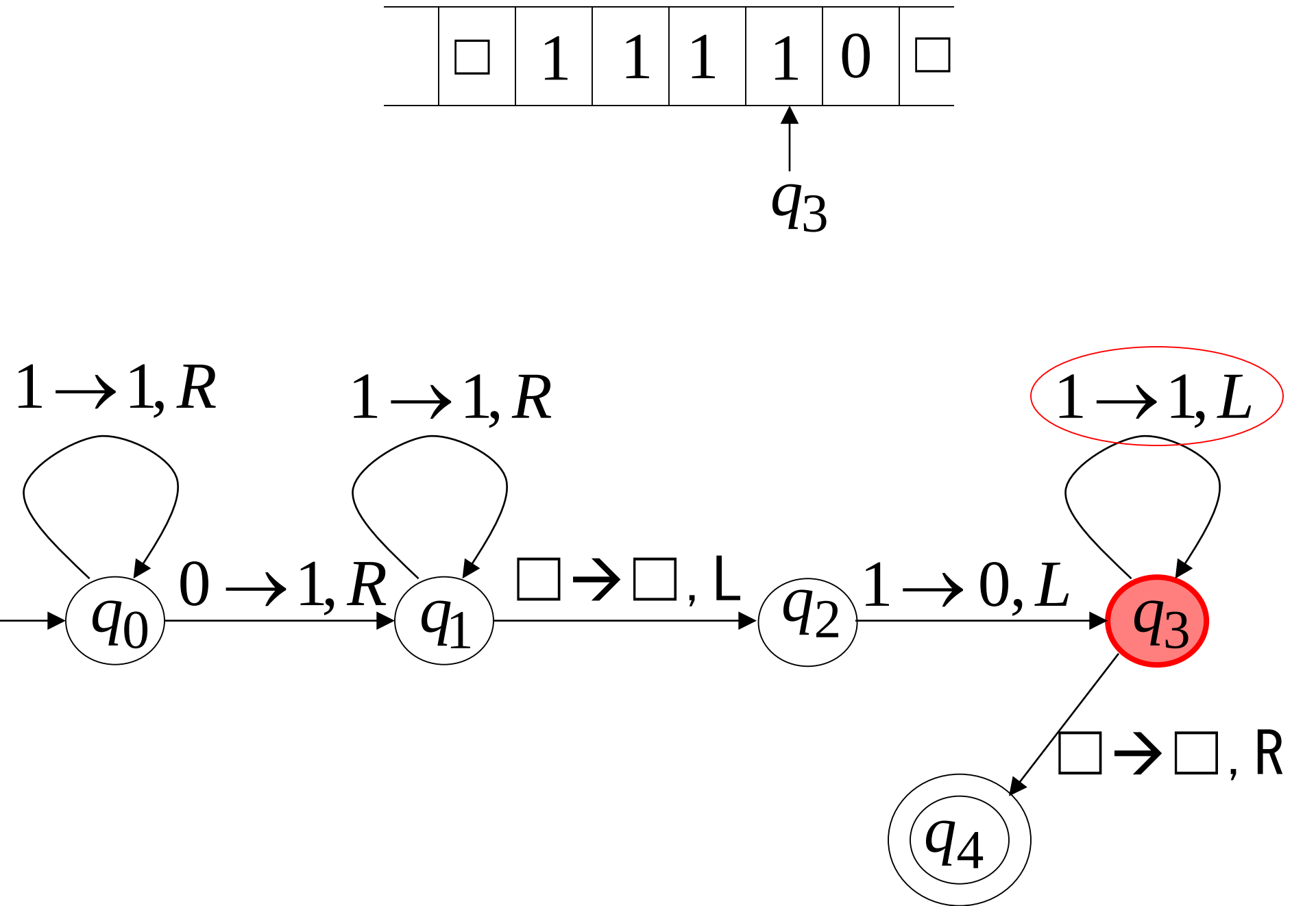

$1 \to 1, R$
$1 \to 1, R$
$1 \to 1, L$
$0 \to 1, R$
□ → □, L
$1 \to 0, L$
□ → □, R
q_0
q_1
q_2
q_3
q_4

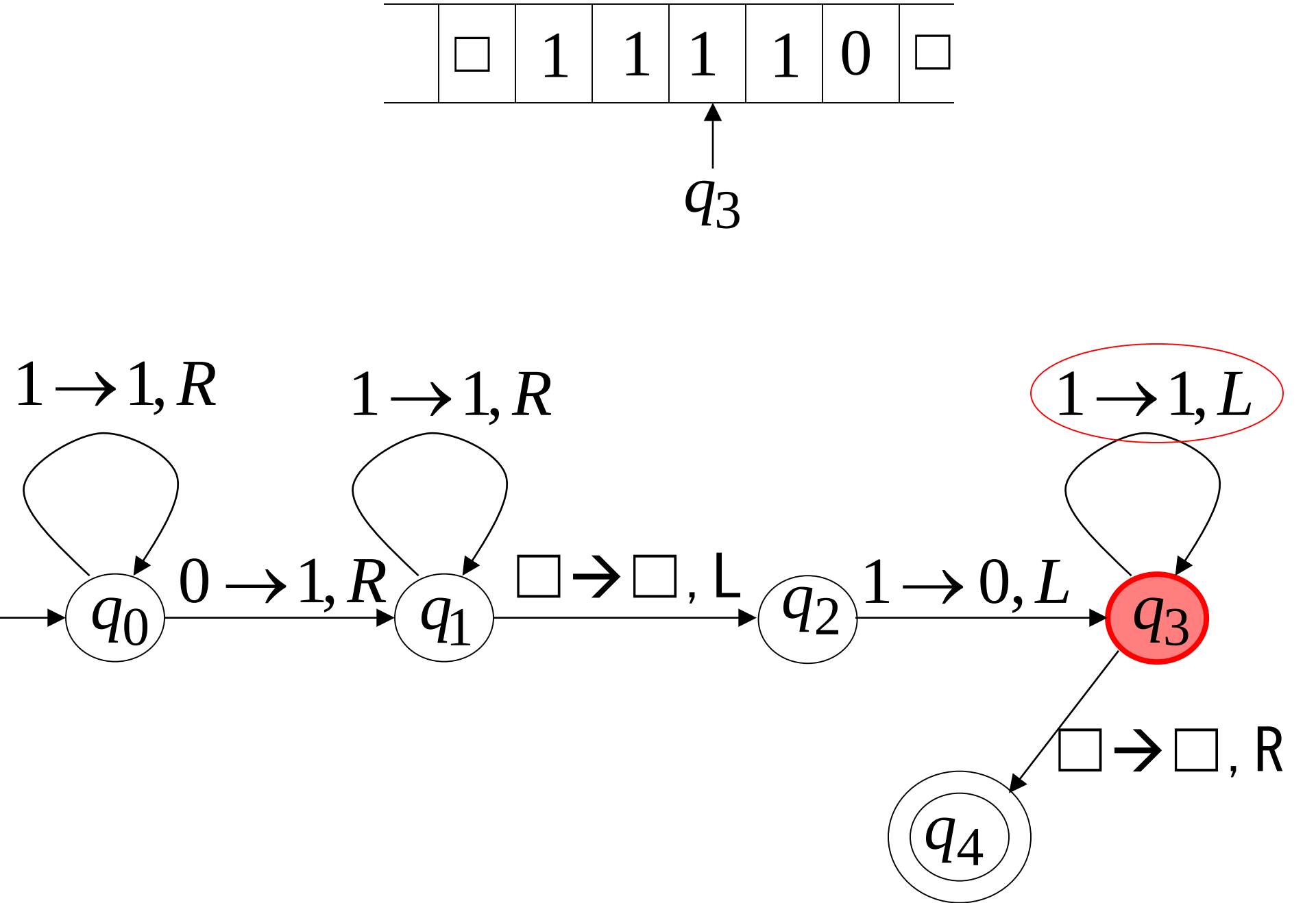

□ 1 1 1 1 1 □
q_1
$1 \to 1, R$
$1 \to 1, R$
$1 \to 1, L$
q_0
$0 \to 1, R$
q_1
□ → □, L
q_2
$1 \to 0, L$
q_3
□ → □, R
q_4

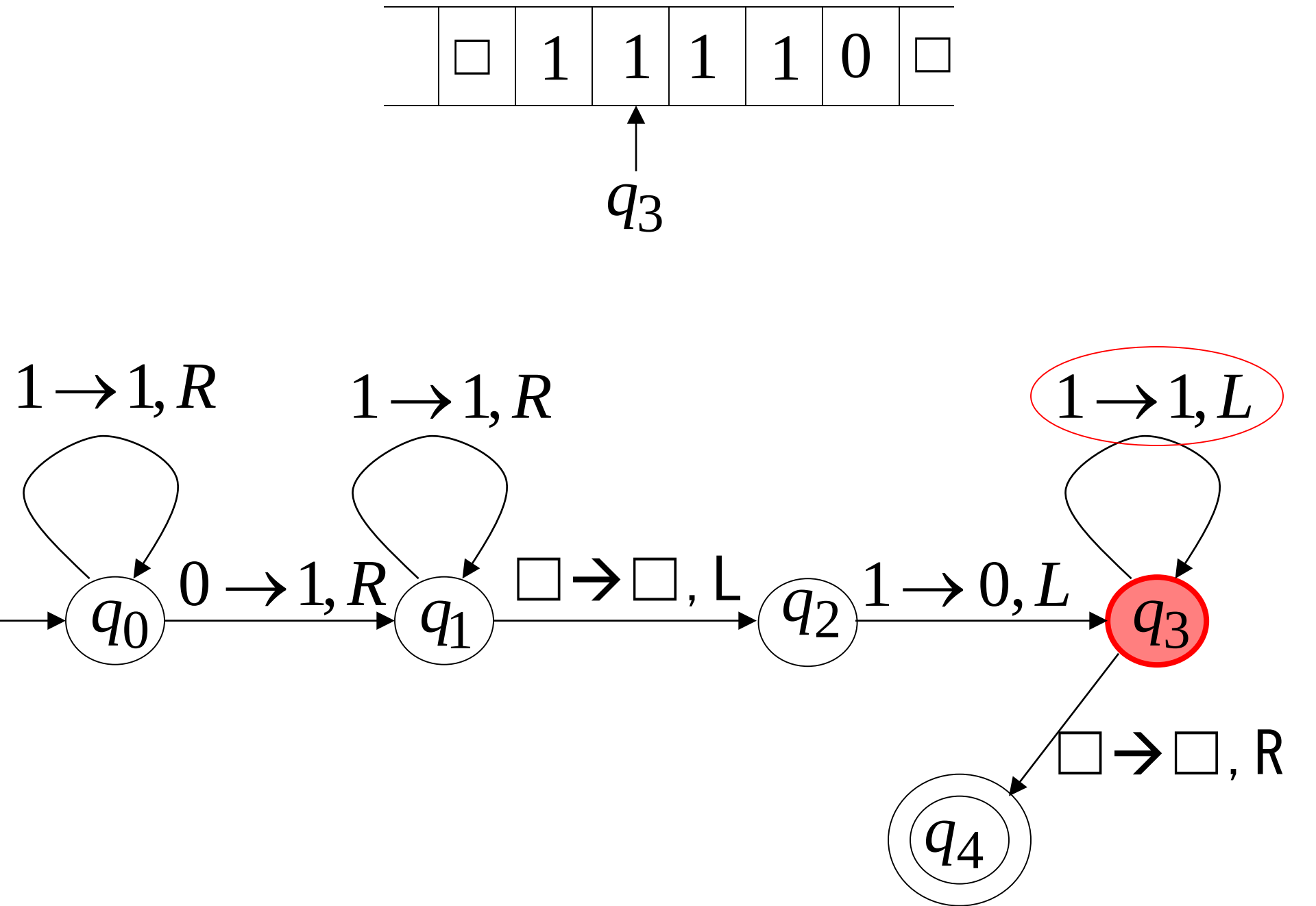

□ 1 1 1 1 1 □
q_1
$1 \to 1, R$
$1 \to 1, R$
$1 \to 1, L$
q_0
$0 \to 1, R$
q_1
□ → □, L
q_2
$1 \to 0, L$
q_3
□ → □, R
q_4

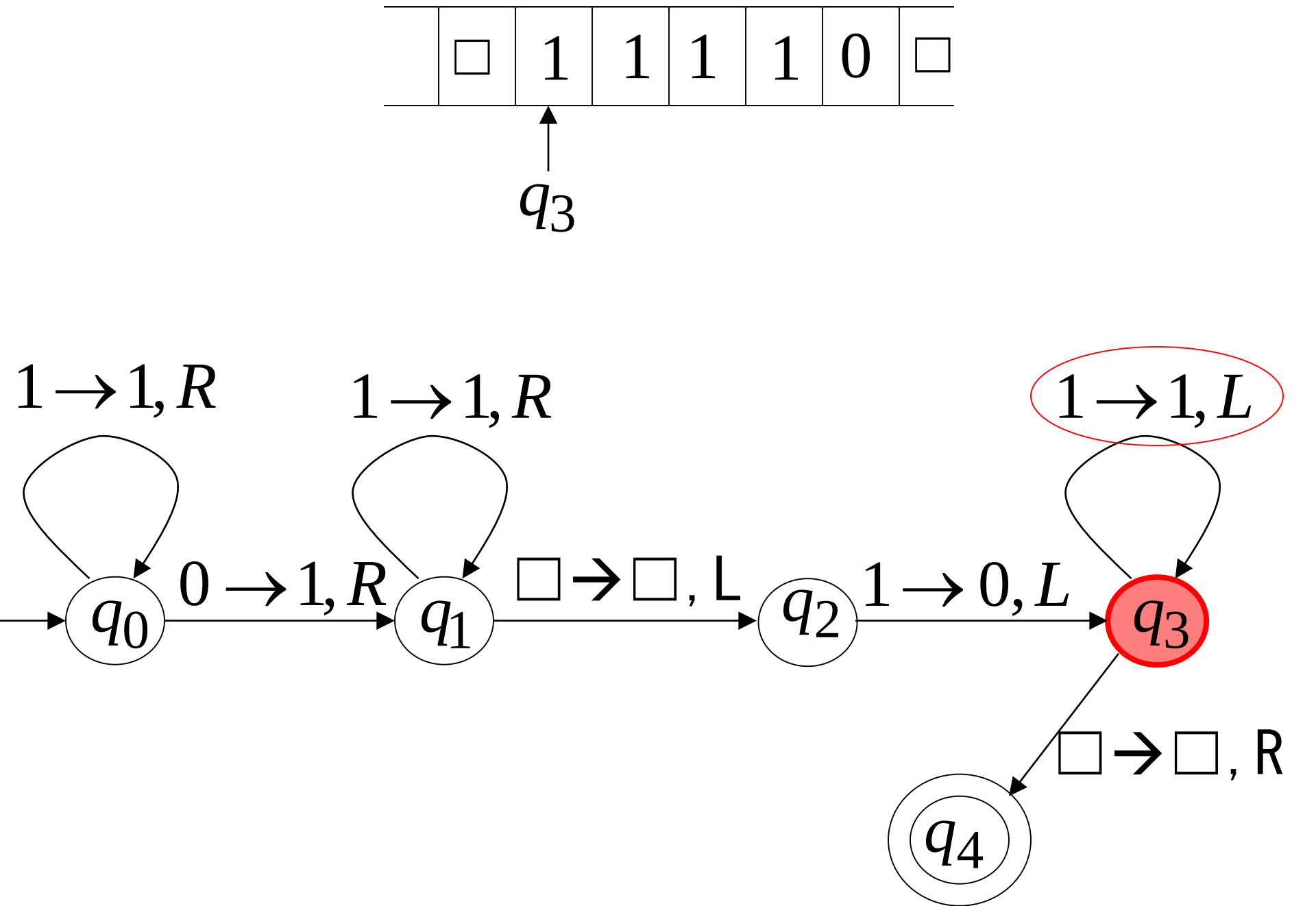

□ 1 1 1 1 1 □
q_1
$1 \to 1, R$
$1 \to 1, R$
$1 \to 1, L$
$0 \to 1, R$
□ → □, L
$1 \to 0, L$
□ → □, R
q_0
q_1
q_2
q_3
q_4

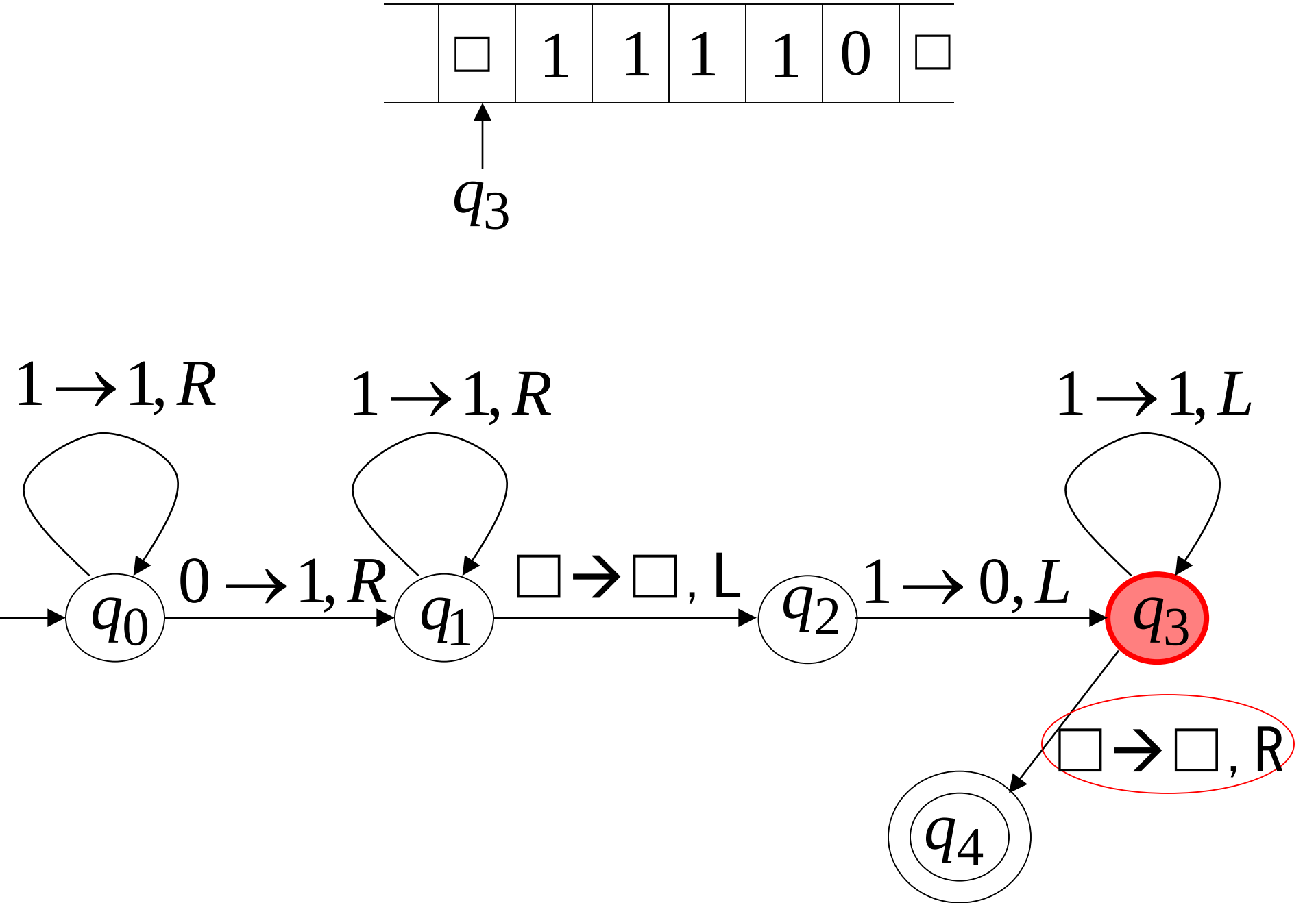

□ 1 1 1 1 1 □
q_2
$1 \to 1, R$
$1 \to 1, R$
$1 \to 1, L$
q_0
$0 \to 1, R$
q_1
□ → □, L
q_2
$1 \to 0, L$
q_3
□ → □, R
q_4

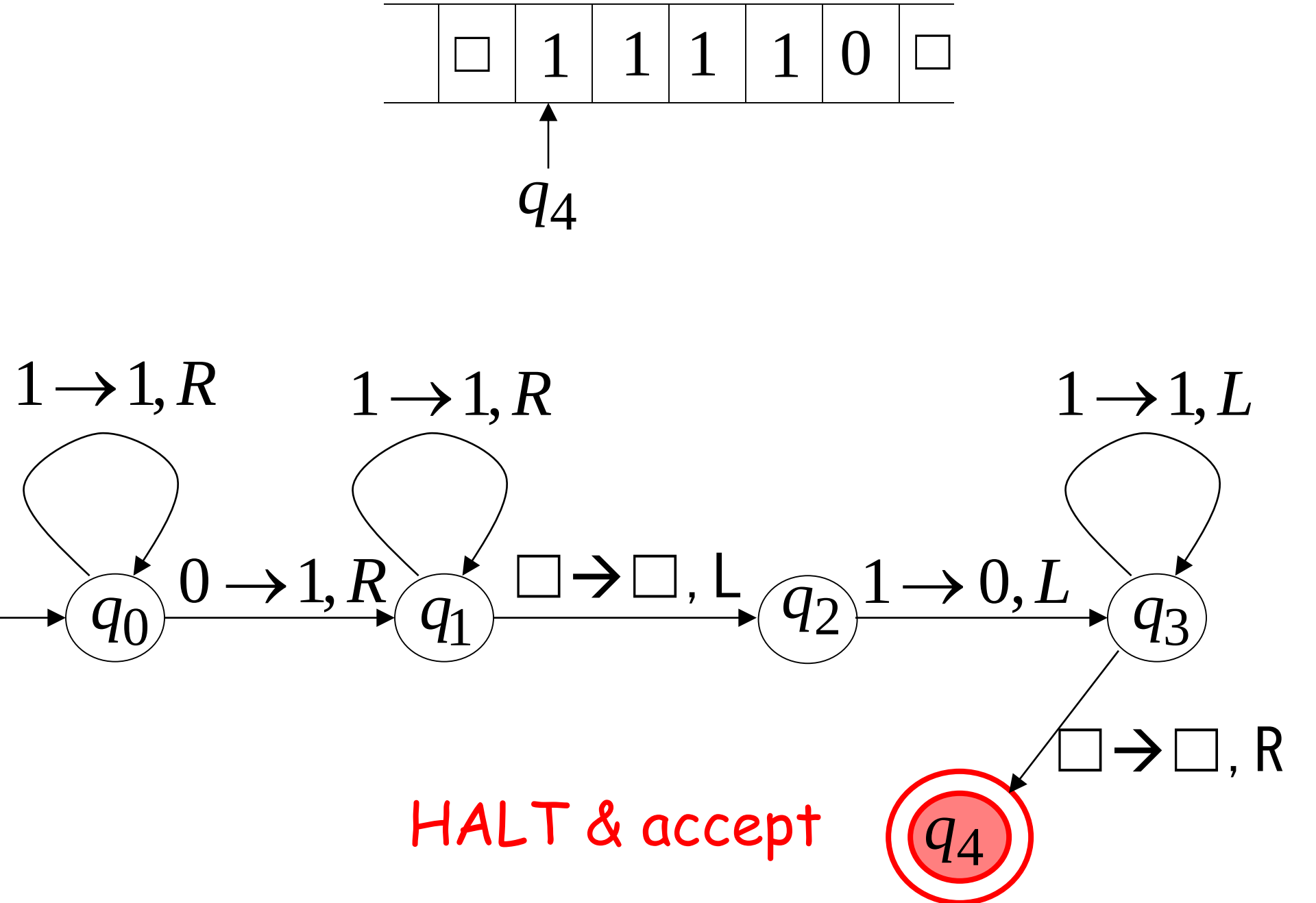

□ 1 1 1 1 0 □
q_3
$1 \to 1, R$
$1 \to 1, R$
$1 \to 1, L$
$0 \to 1, R$
□ → □, L
$1 \to 0, L$
□ → □, R
q_0
q_1
q_2
q_3
q_4


□ 1 1 1 1 0 □
q_3
$1 \to 1, R$
$1 \to 1, R$
$1 \to 1, L$
$0 \to 1, R$
□ → □, L
$1 \to 0, L$
□ → □, R
q_0
q_1
q_2
q_3
q_4


□ 1 1 1 1 0 □
q_3
$1 \to 1, R$
$1 \to 1, R$
$1 \to 1, L$
$0 \to 1, R$
□ → □, L
$1 \to 0, L$
□ → □, R
q_0
q_1
q_2
q_3
q_4


□ 1 1 1 1 0 □
q_3
$1 \to 1, R$
$1 \to 1, R$
$1 \to 1, L$
$0 \to 1, R$
□ → □, L
$1 \to 0, L$
□ → □, R
q_0
q_1
q_2
q_3
q_4


□ 1 1 1 1 0 □
q_3
$1 \to 1, R$
$1 \to 1, R$
$1 \to 1, L$
q_0
$0 \to 1, R$
q_1
□ → □, L
q_2
$1 \to 0, L$
q_3
□ → □, R
q_4


□ 1 1 1 1 0 □
q_4
$1 \to 1, R$
$1 \to 1, R$
$1 \to 1, L$
$0 \to 1, R$
□ → □, L
$1 \to 0, L$
□ → □, R
q_0
q_1
q_2
q_3
q_4
HALT & accept

Another Example

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

Turing Machine:

Input string: x unary

Output string: xx unary

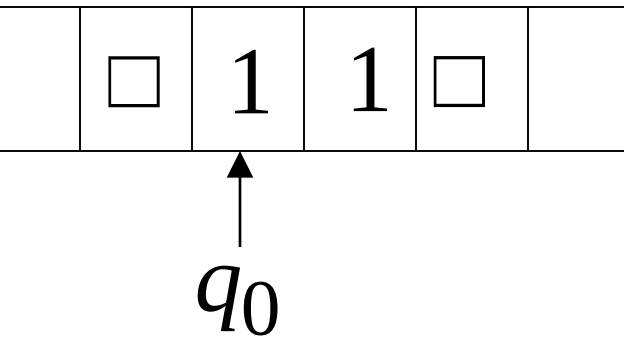
Turing Machine Pseudocode for $f(x) = 2x$

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

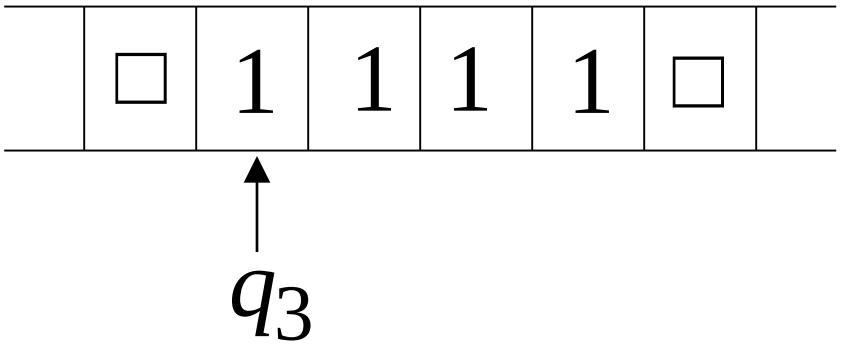
  Until no more $ remain

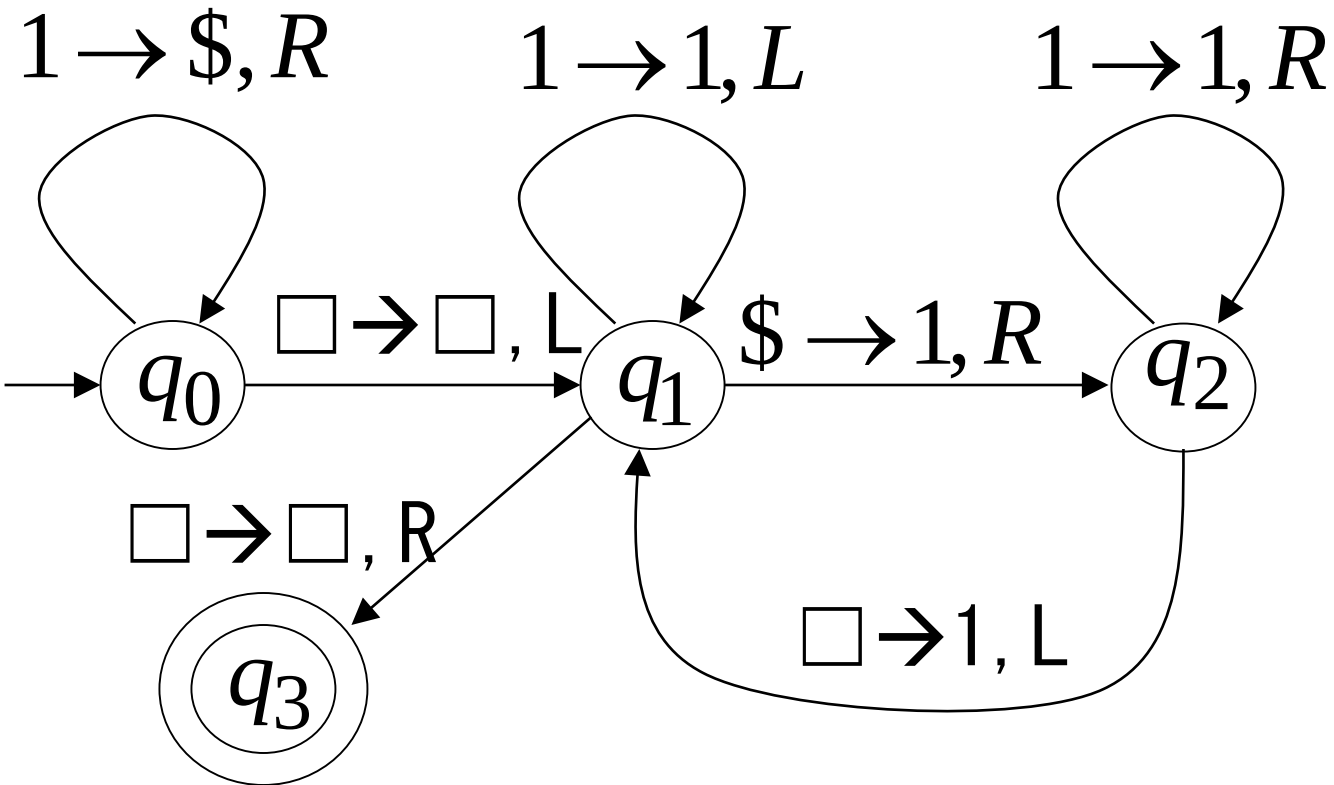
Example

Start



Finish





Another Example

The function is computable

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

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

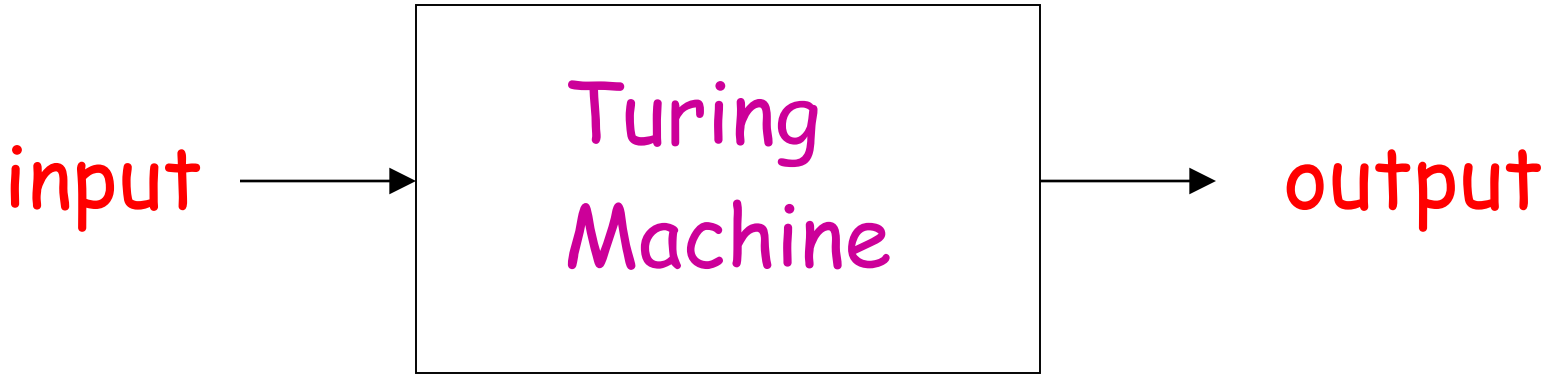
  erase tape, write 1 $(x > y)$

  else

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

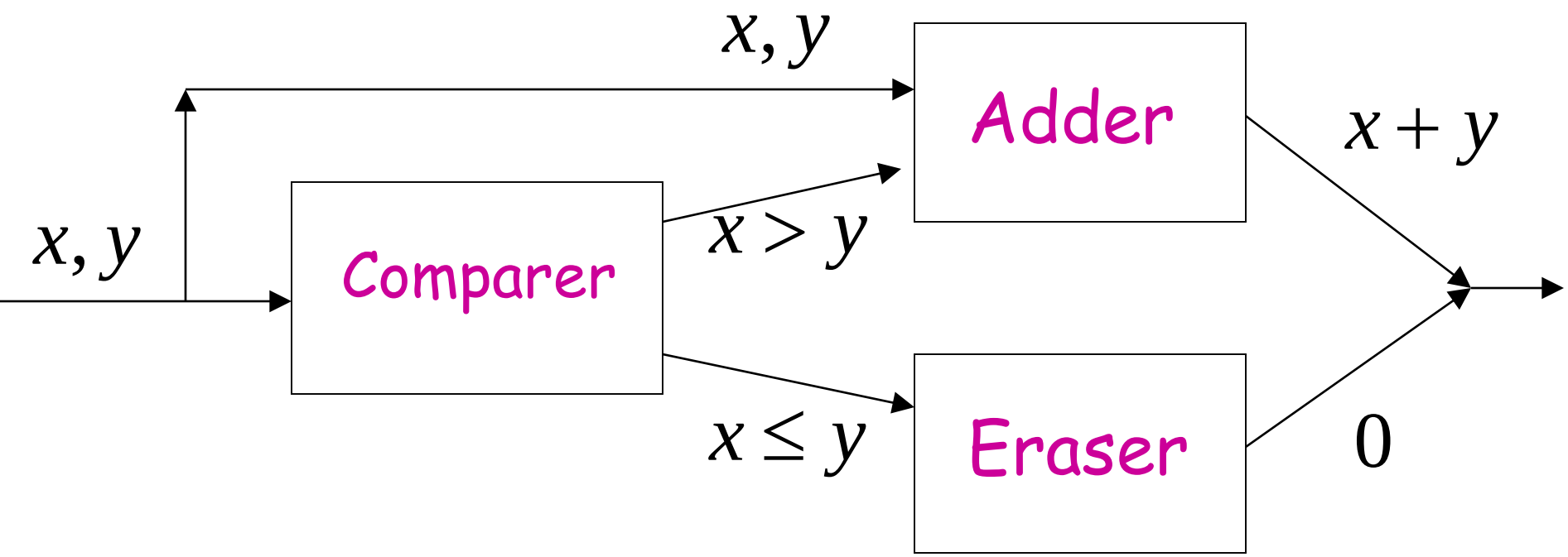
Combining Turing Machines

Block Diagram



Example:

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



Turing's Thesis

Turing's thesis (1930):

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

Arguments that support Turing's 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 TM program cannot be written.
- Alternative models have been proposed but none of them is more powerful than the TM model.

Algorithm:

An algorithm for a problem is a
Turing Machine which solves the problem

The algorithm describes the steps of
the mechanical means

This is easily translated to computation steps
of a Turing machine

An algorithm for a function f: D → R is a Turing machine M, which given as input any $d \in D$ on its tape, eventually halts with the correct answer $f(d) \in R$ on its tape.

$$q_0 d \overset{*}{\vdash}_M q_f\, f(d) \qquad\qquad q_f \in F$$

for all $d \in D$.

When we say: There exists an algorithm

We mean: There exists a Turing Machine that executes the algorithm