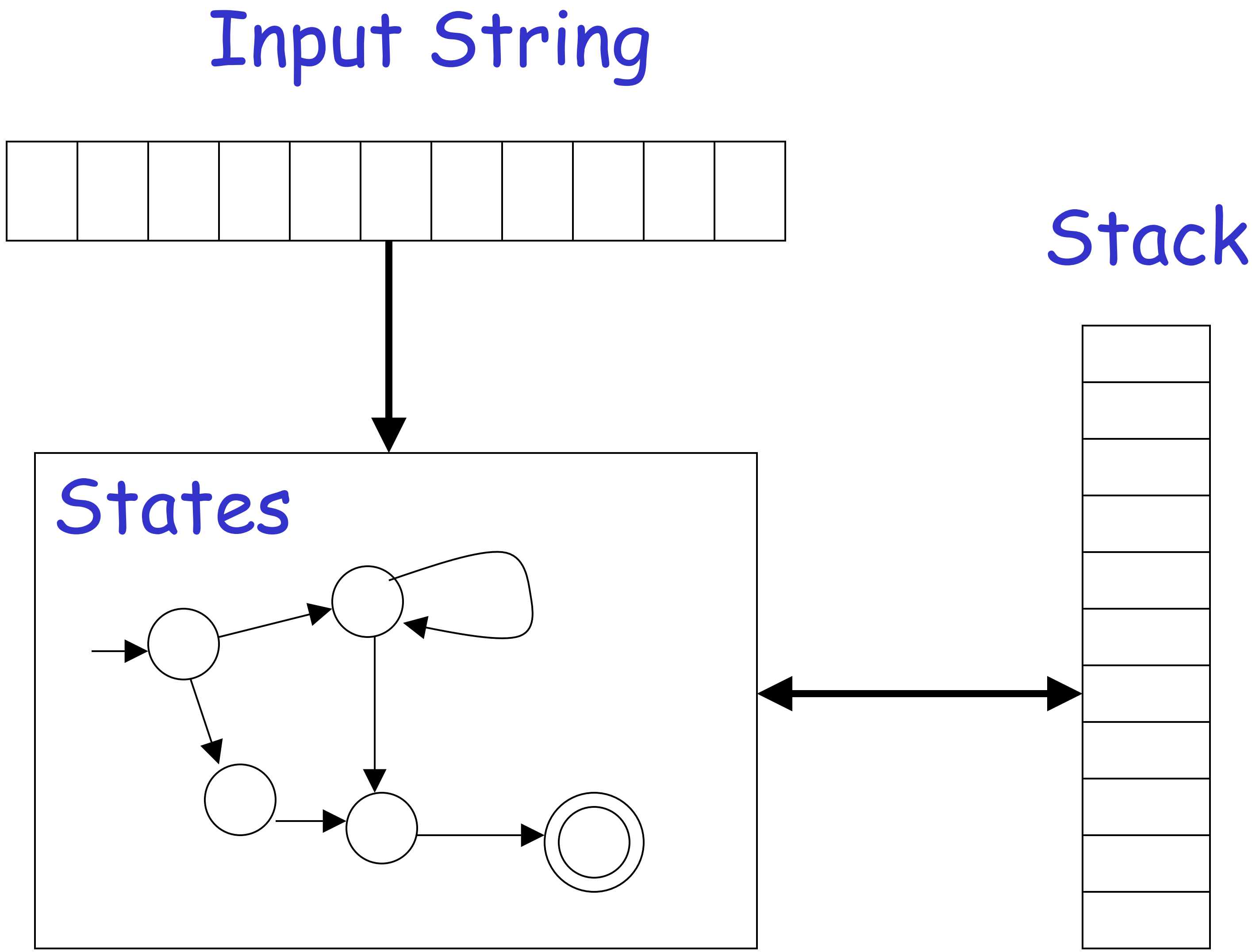
Pushdown Automata
PDAs

Pushdown Automaton

- A Pushdown Automata (PDA) is a way to implement a Context Free Grammar in a similar way we design Finite Automata for Regular Grammar
  - It is more powerful than FSM
  - FSM has a very limited memory but PDA has more memory
  - PDA = Finite State Machine + A Stack

Pushdown Automaton - components

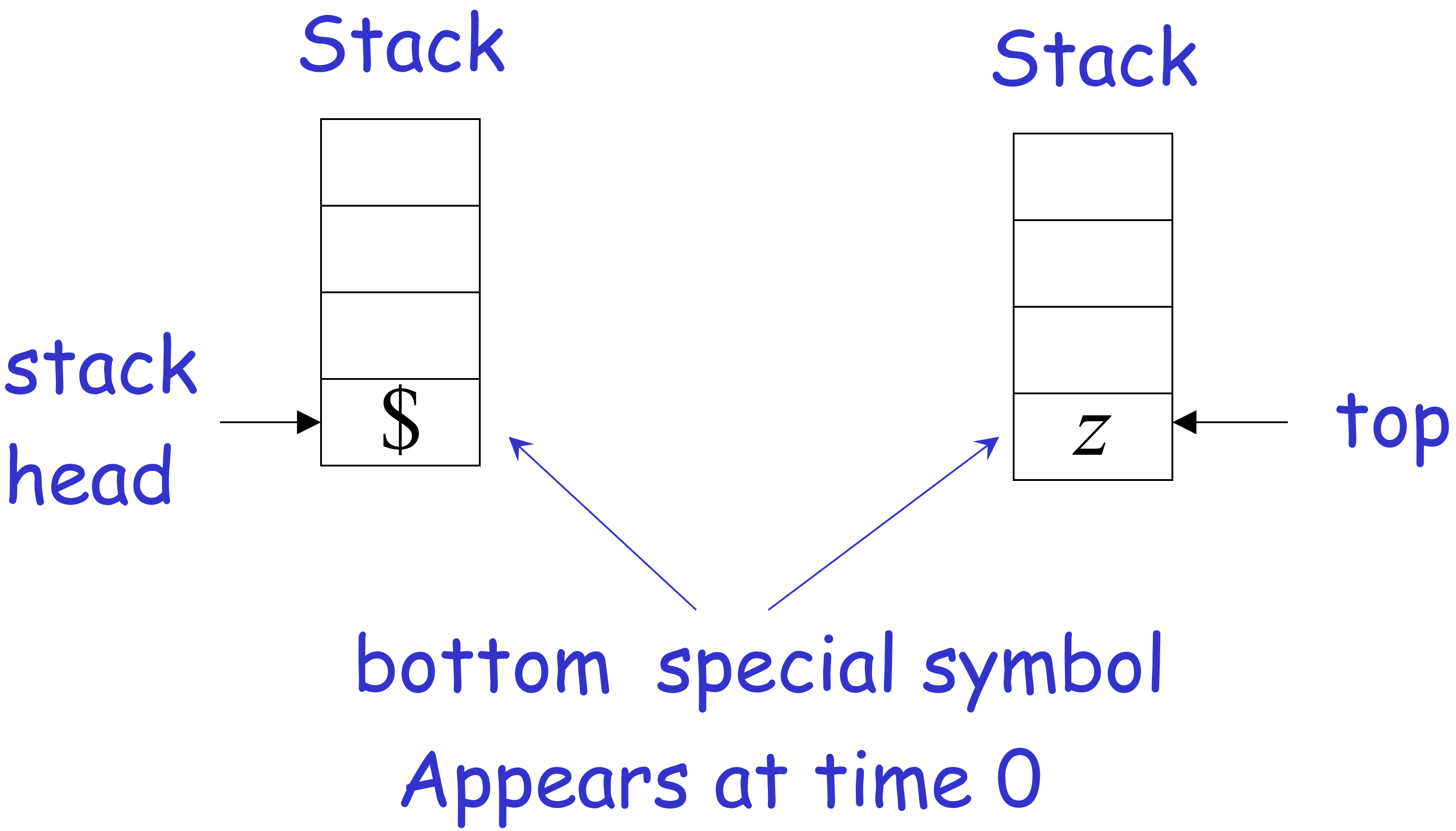


FORMAL DEFINITION OF A PUSHDOWN AUTOMATON

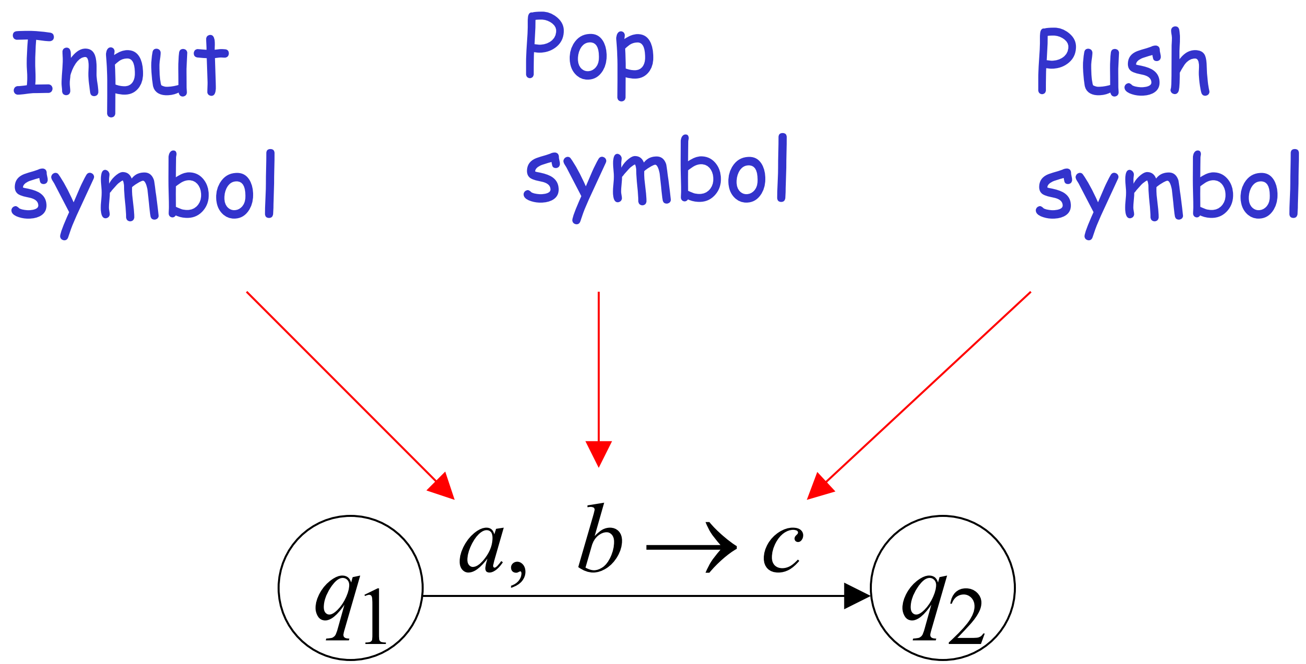
- A *pushdown automaton* is a 6-tuple $(Q, \Sigma, \Gamma, \delta, q0, F)$, where Q, Σ, Γ, and F are all finite sets, and

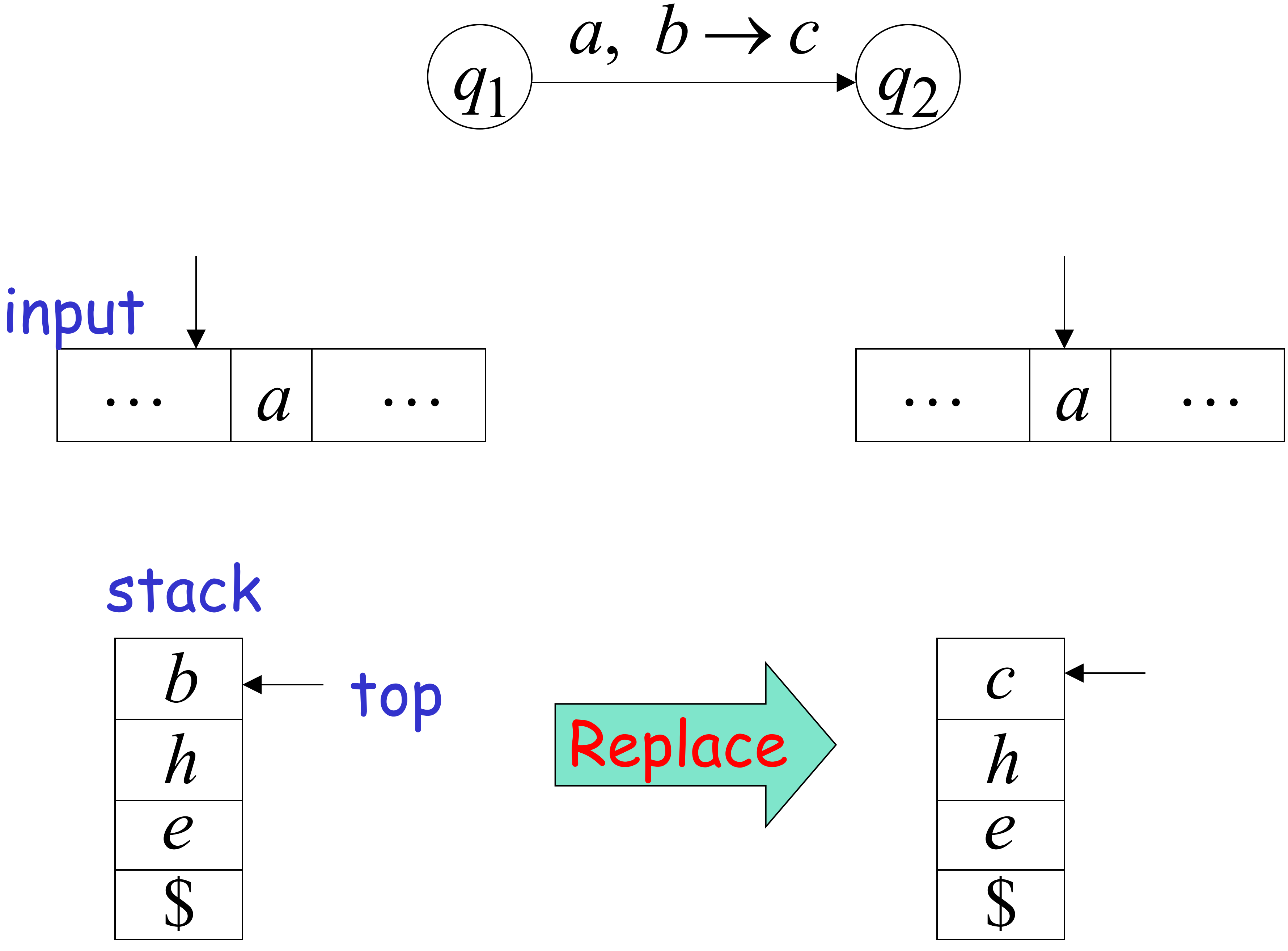
1. Q is the set of states,
2. Σ is the input alphabet,
3. Γ is the stack alphabet,
4. $\delta: Q \times \Sigma\varepsilon \times \Gamma\varepsilon \rightarrow P(Q \times \Gamma\varepsilon)$ is the transition function,
5. $q0 \in Q$ is the start state, and
6. $F \subseteq Q$ is the set of accept states.

Initial Stack Symbol

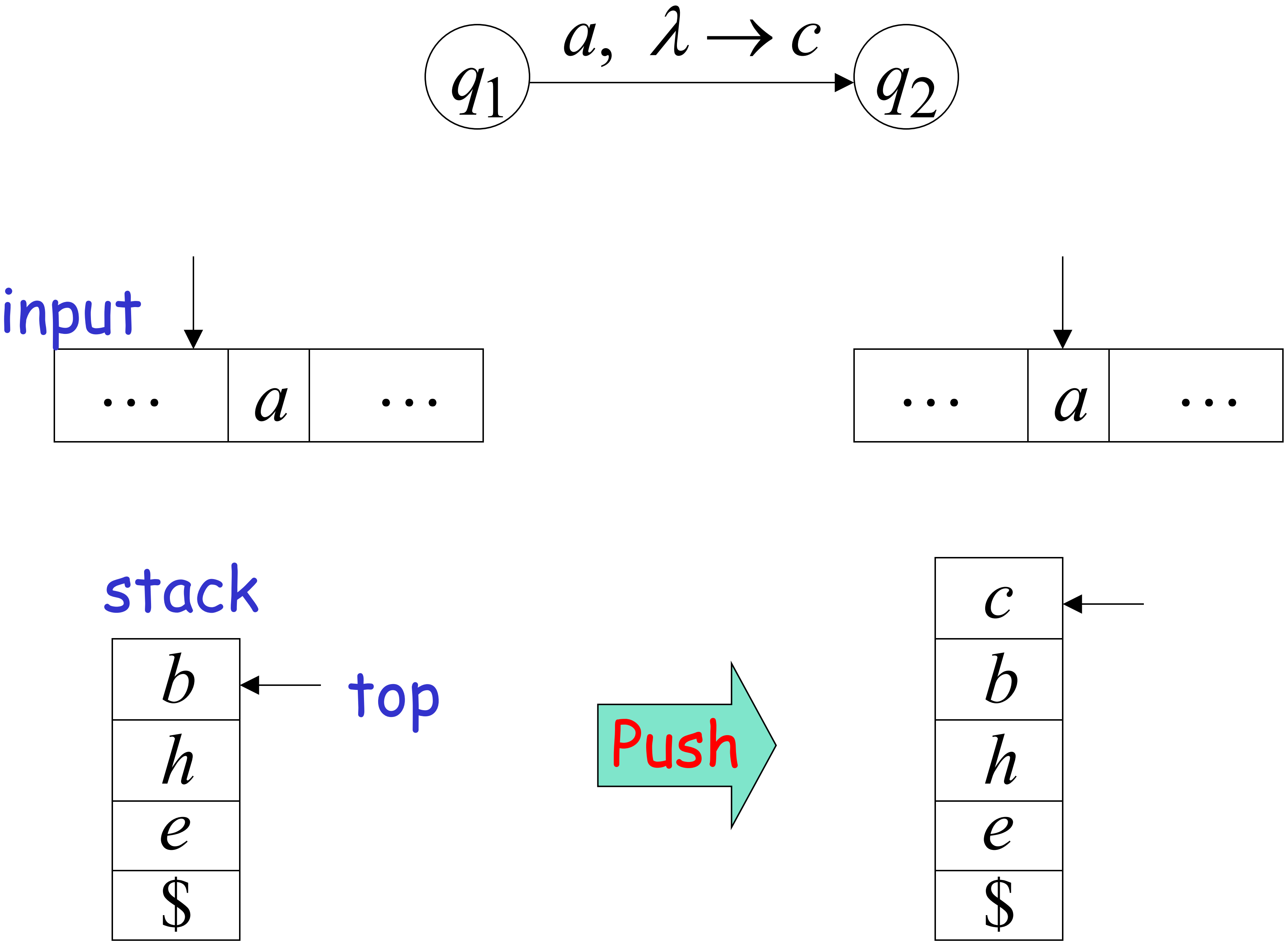


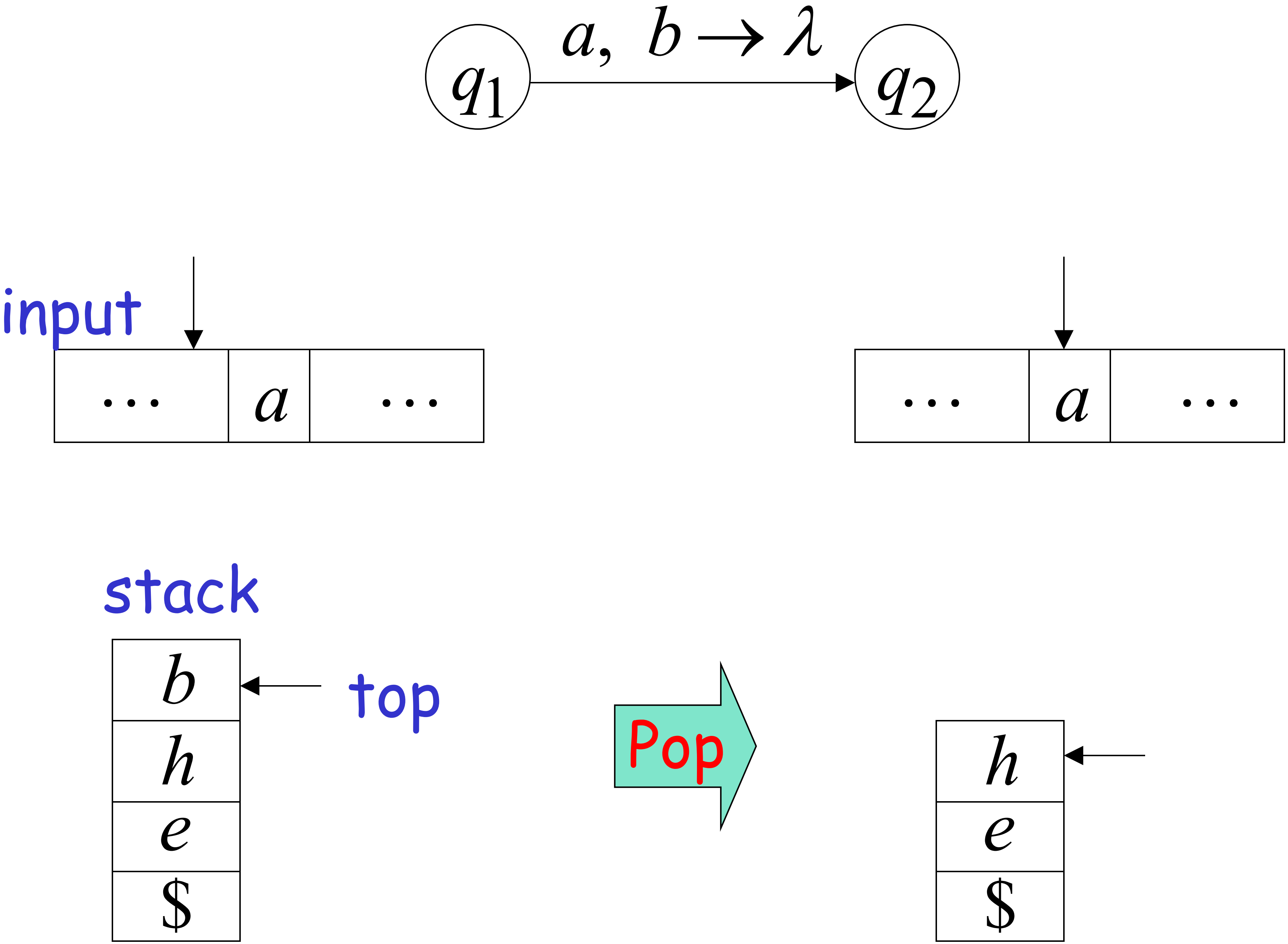
The States



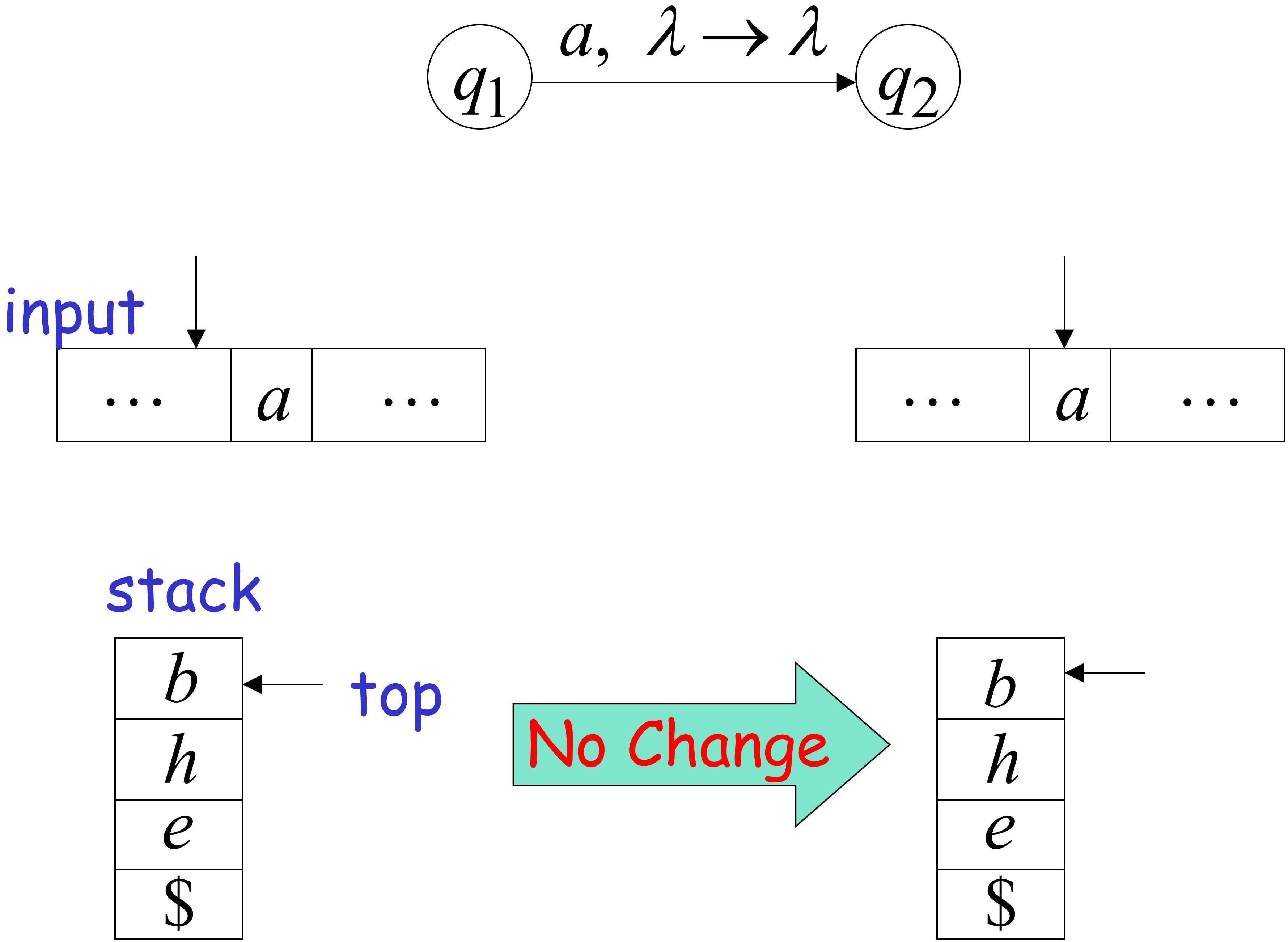


q_1
$a, \ b \to c$
q_2
input
$\cdots$ a $\cdots$
$\cdots$ a $\cdots$
stack
b
h
e
$\$$
top
Replace
c
h
e
$\$$

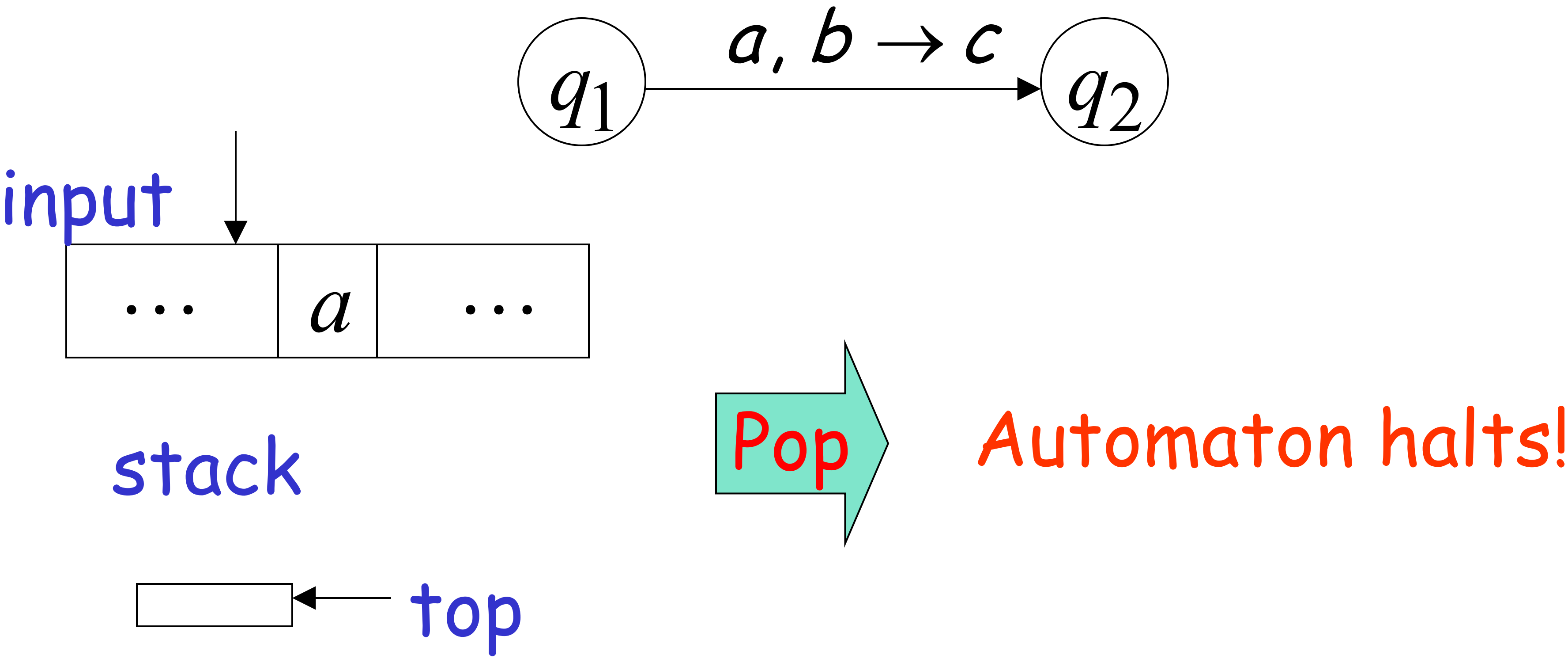

q_1
$a, \lambda \to c$
q_2
input
$\cdots$ a $\cdots$
$\cdots$ a $\cdots$
stack
b
h
e
$\$$
top
Push
c
b
h
e
$\$$



q_1
$a, \; b \rightarrow \lambda$
q_2
input
$\cdots$ a $\cdots$
$\cdots$ a $\cdots$
stack
b
h
e
$\$$
top
Pop
h
e
$\$$


$a, \lambda \to \lambda$
q_1
q_2
input
$\cdots$ a $\cdots$
stack
top
b
h
e
$\$$
No Change

Pop from Empty Stack

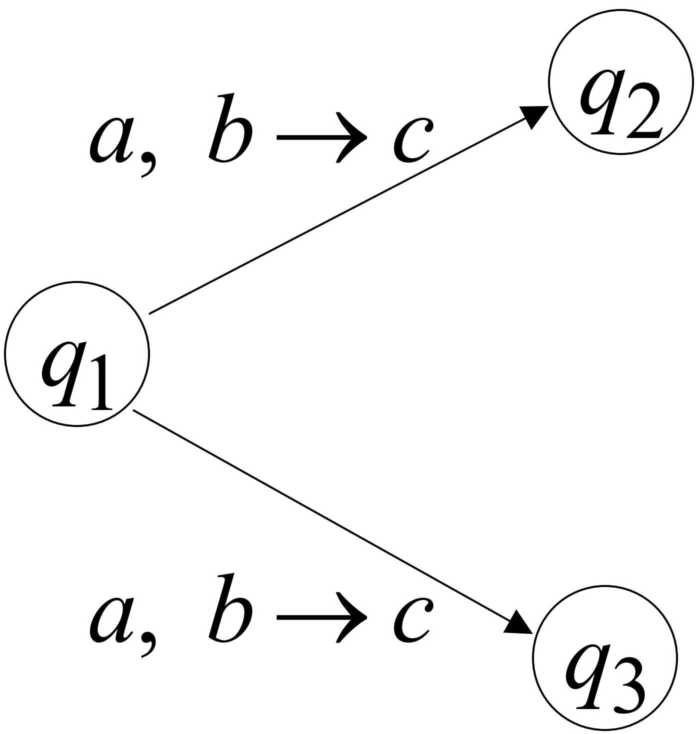


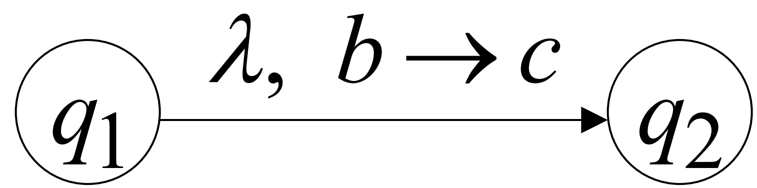
If the automaton attempts to pop from empty stack then it halts and rejects input

Non-Determinism

PDAs are non-deterministic

Allowed non-deterministic transitions



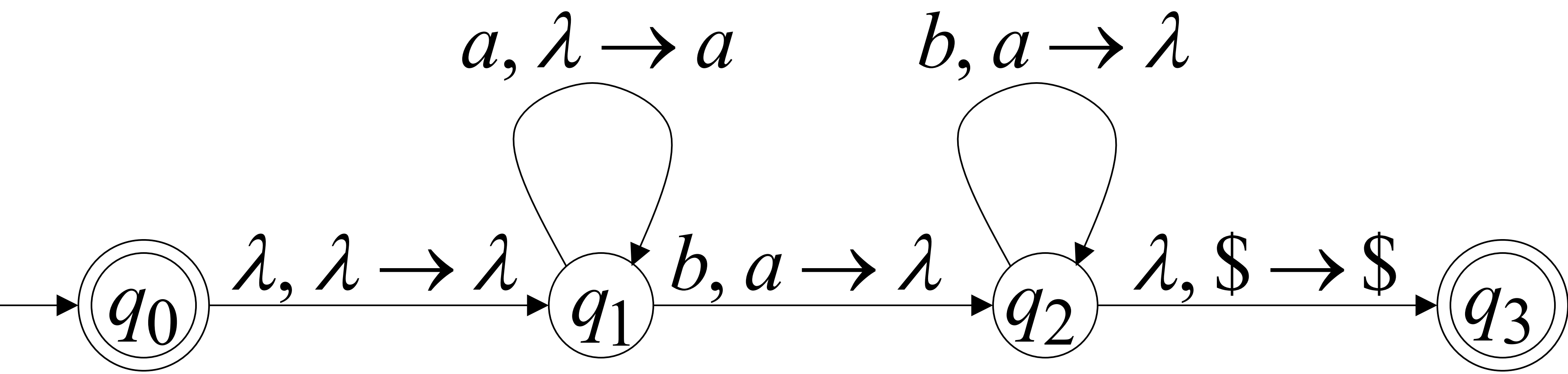


$\lambda -$ transition

Example PDA

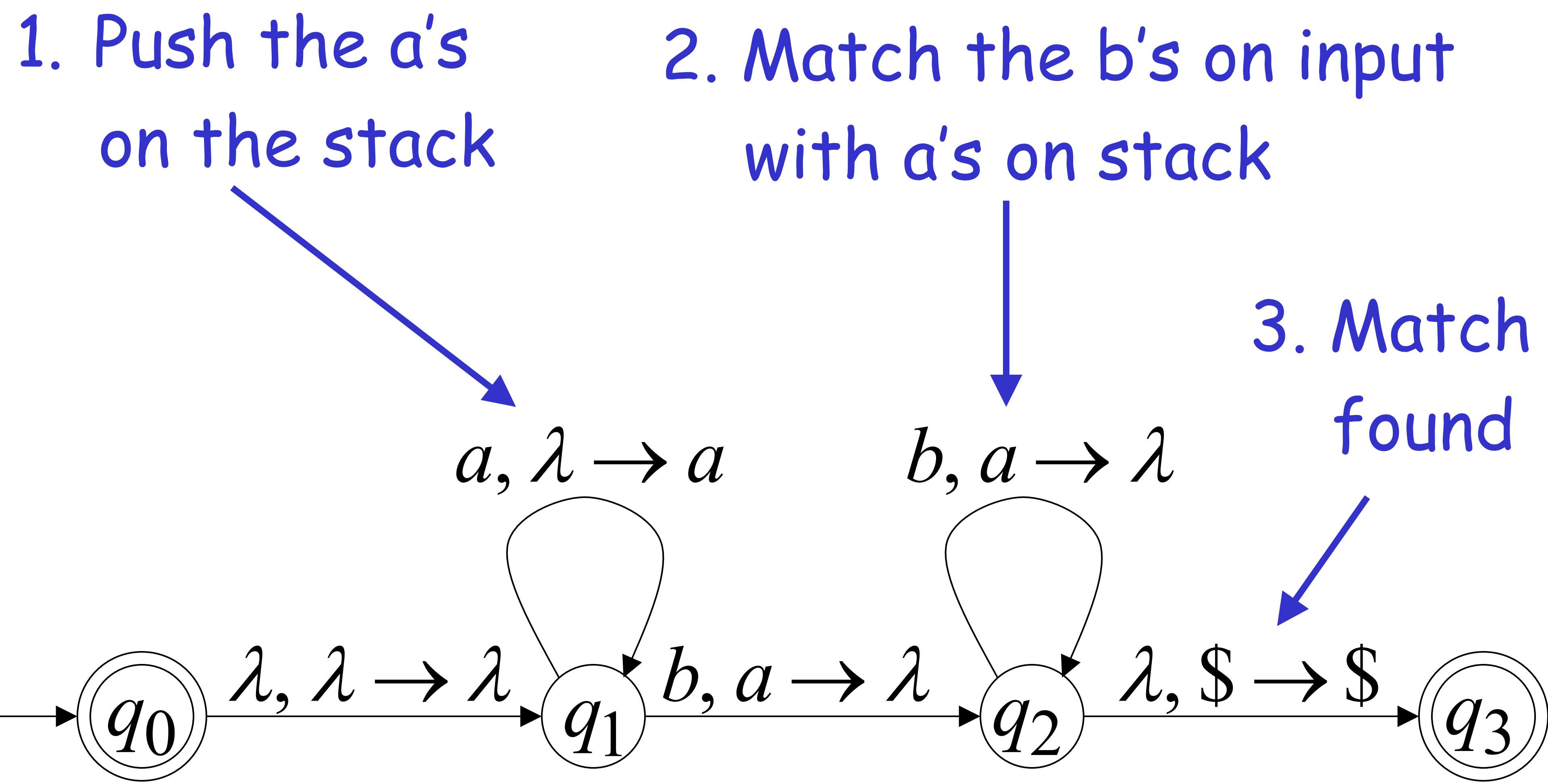
PDA M:

$$L(M) = \{a^n b^n : n \geq 0\}$$



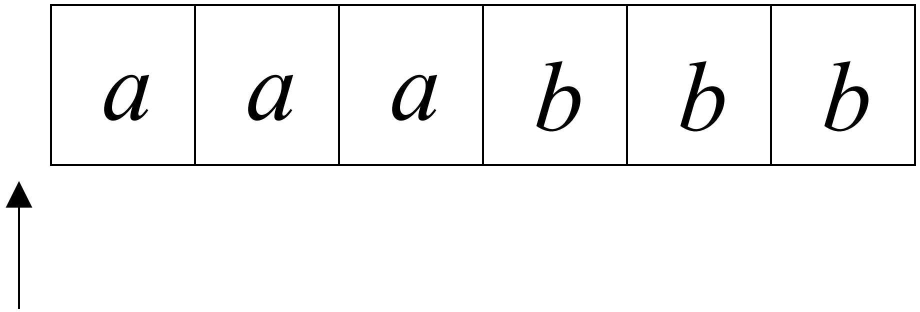
$$L(M) = \{a^n b^n : n \geq 0\}$$

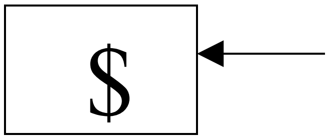
Basic Idea:



Execution Example: Time 0

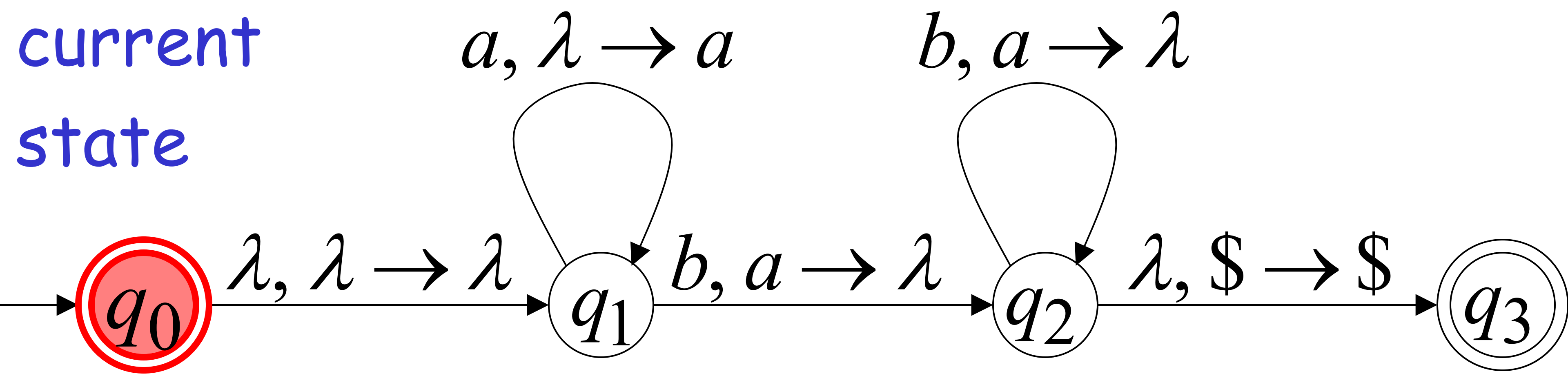
Input



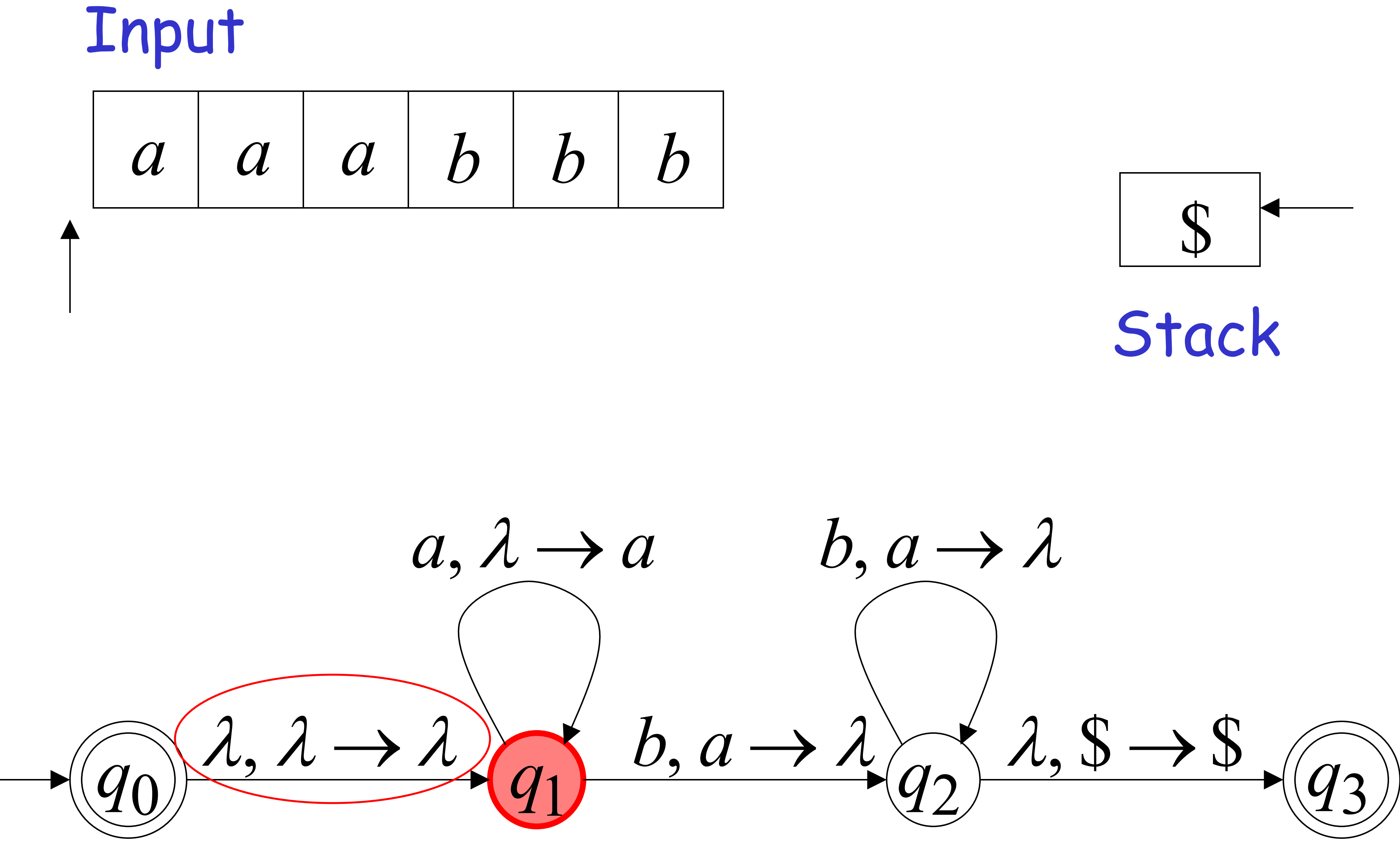


Stack

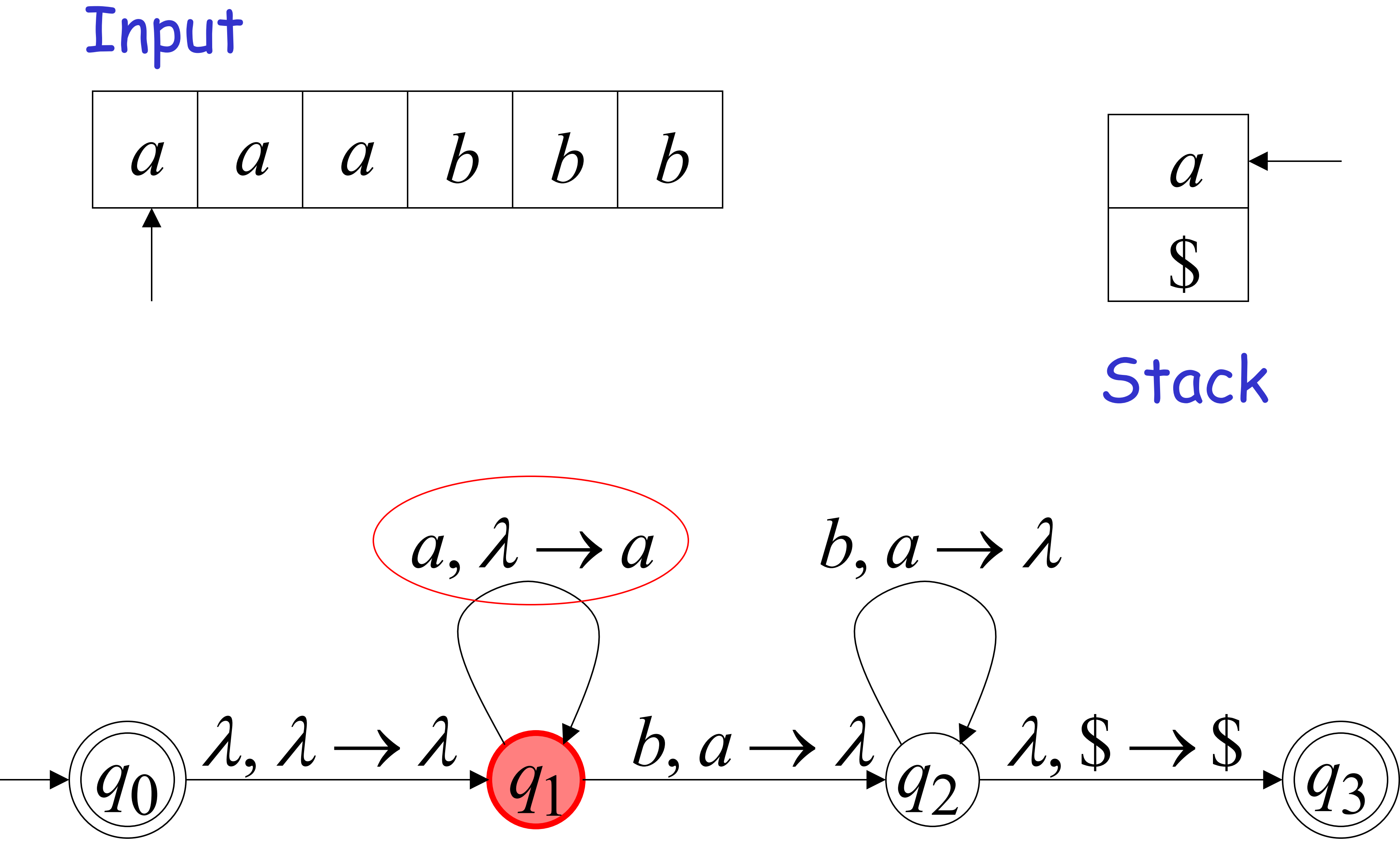
current state



Time 1



Time 2



Input
a a a b b b
a
$
Stack
$a, \lambda \to a$
$b, a \to \lambda$
$\lambda, \lambda \to \lambda$
$b, a \to \lambda$
$\lambda, \$ \to \$$
q_0
q_1
q_2
q_3

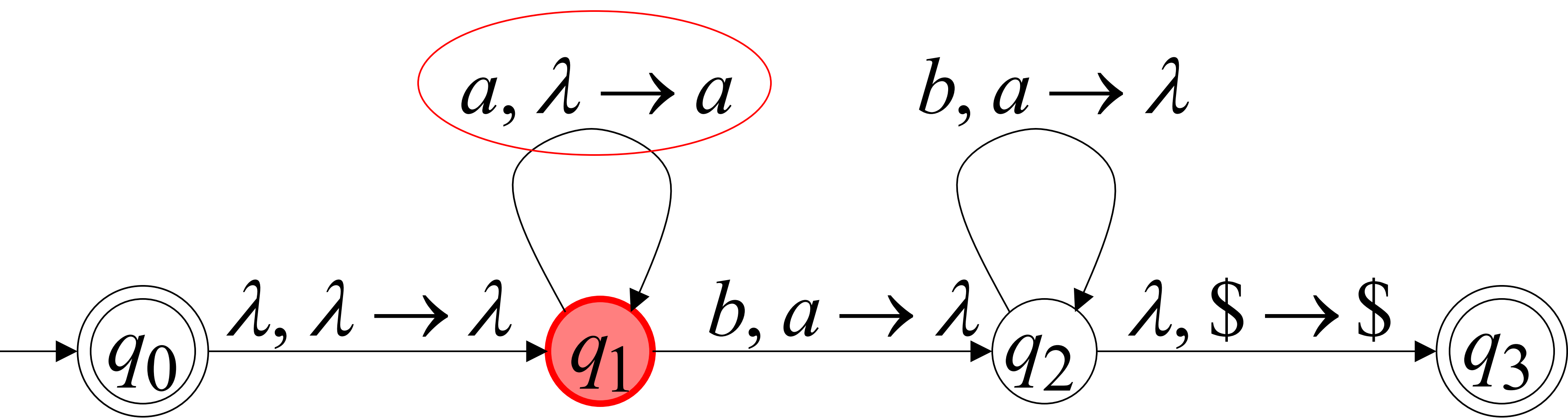
Time 3

Input

a	a	a	b	b	b

a
a
$\$$

Stack



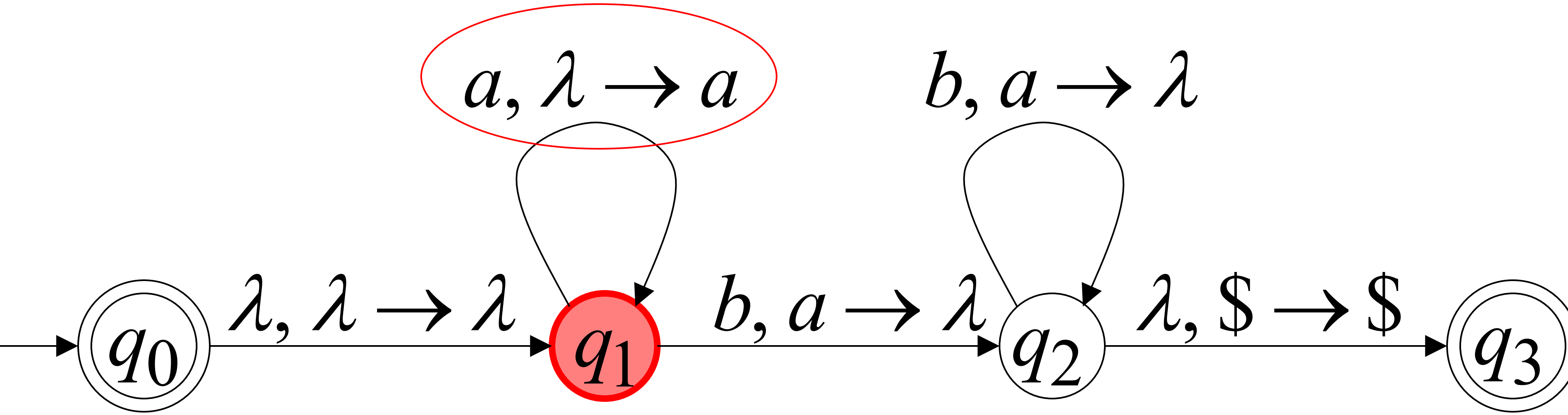
Time 4

Input

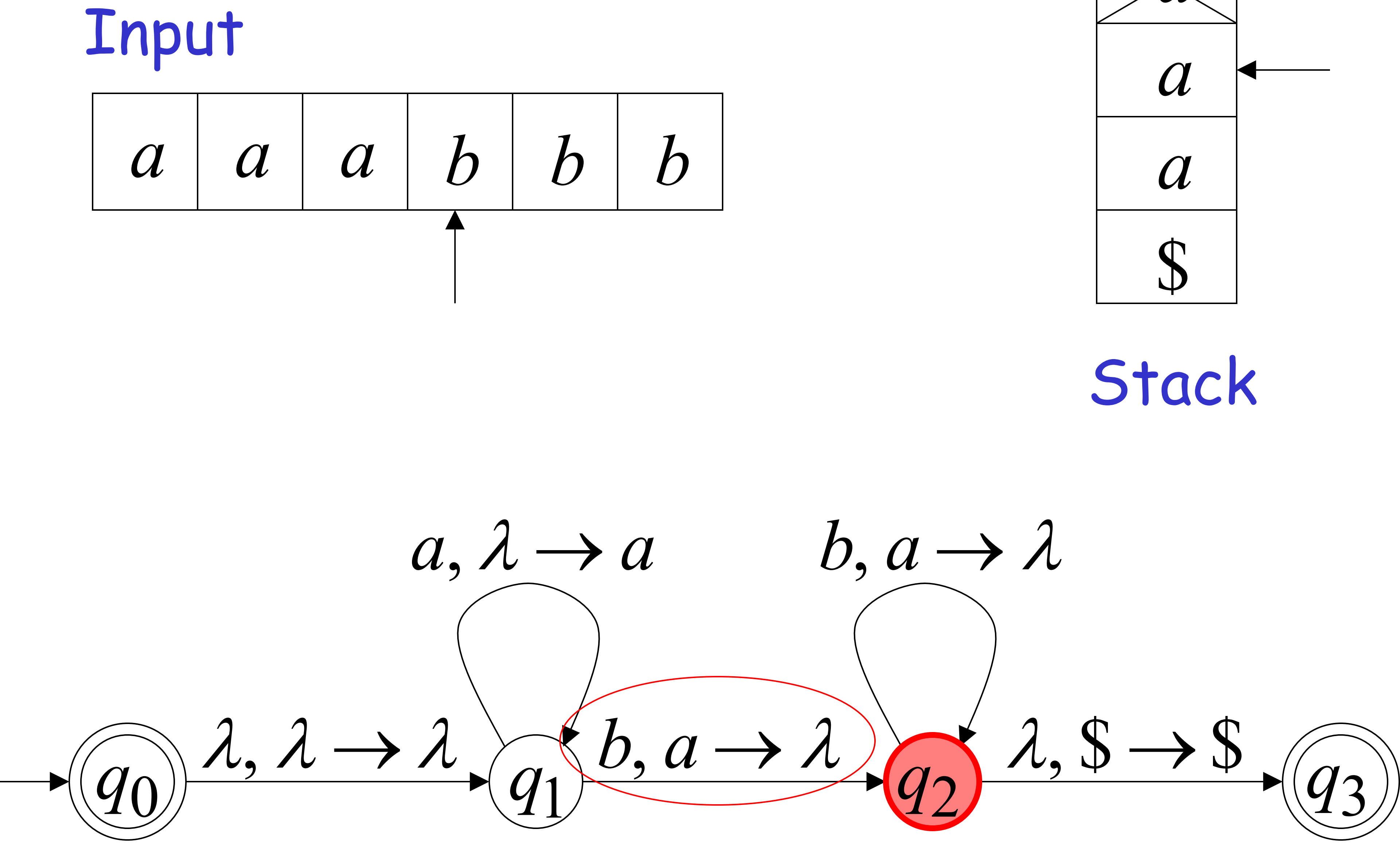
a	a	a	b	b	b

a
a
a
$\$$

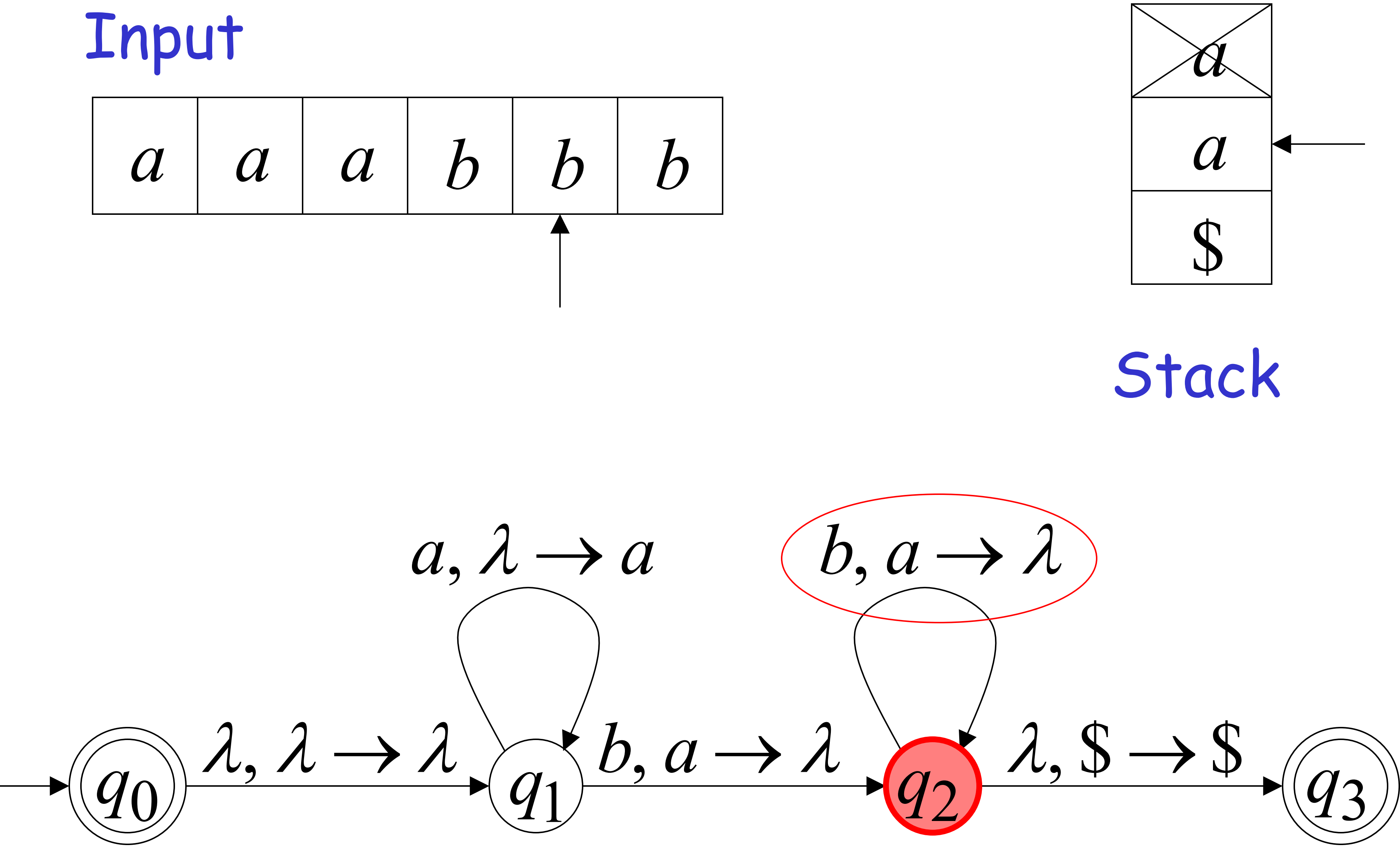
Stack



Time 5



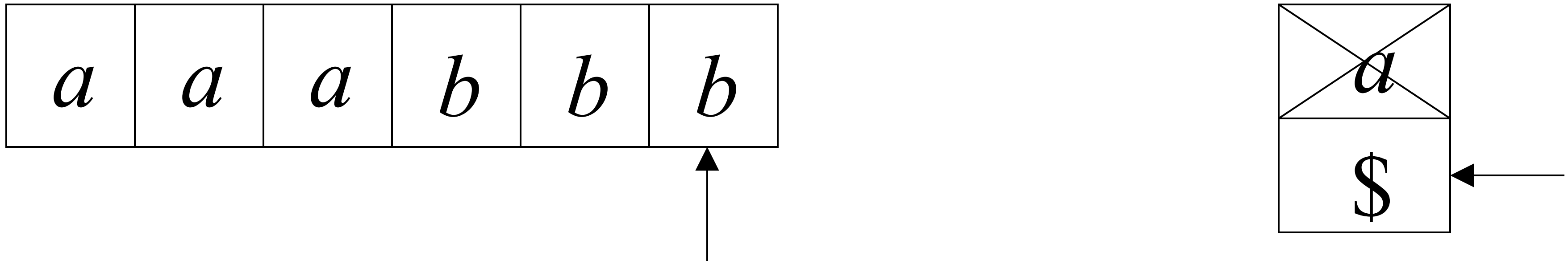
Time 6



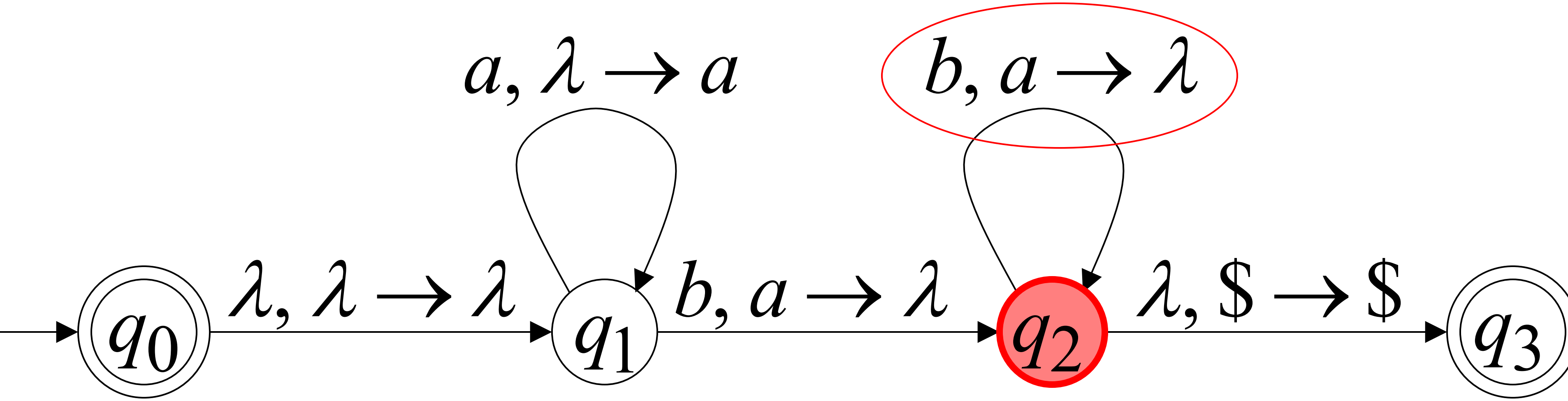
Input
a
a
a
b
b
b
a
a
$\$$
Stack
$a, \lambda \rightarrow a$
$b, a \rightarrow \lambda$
$\lambda, \lambda \rightarrow \lambda$
$b, a \rightarrow \lambda$
$\lambda, \$ \rightarrow \$$
q_0
q_1
q_2
q_3

Time 7

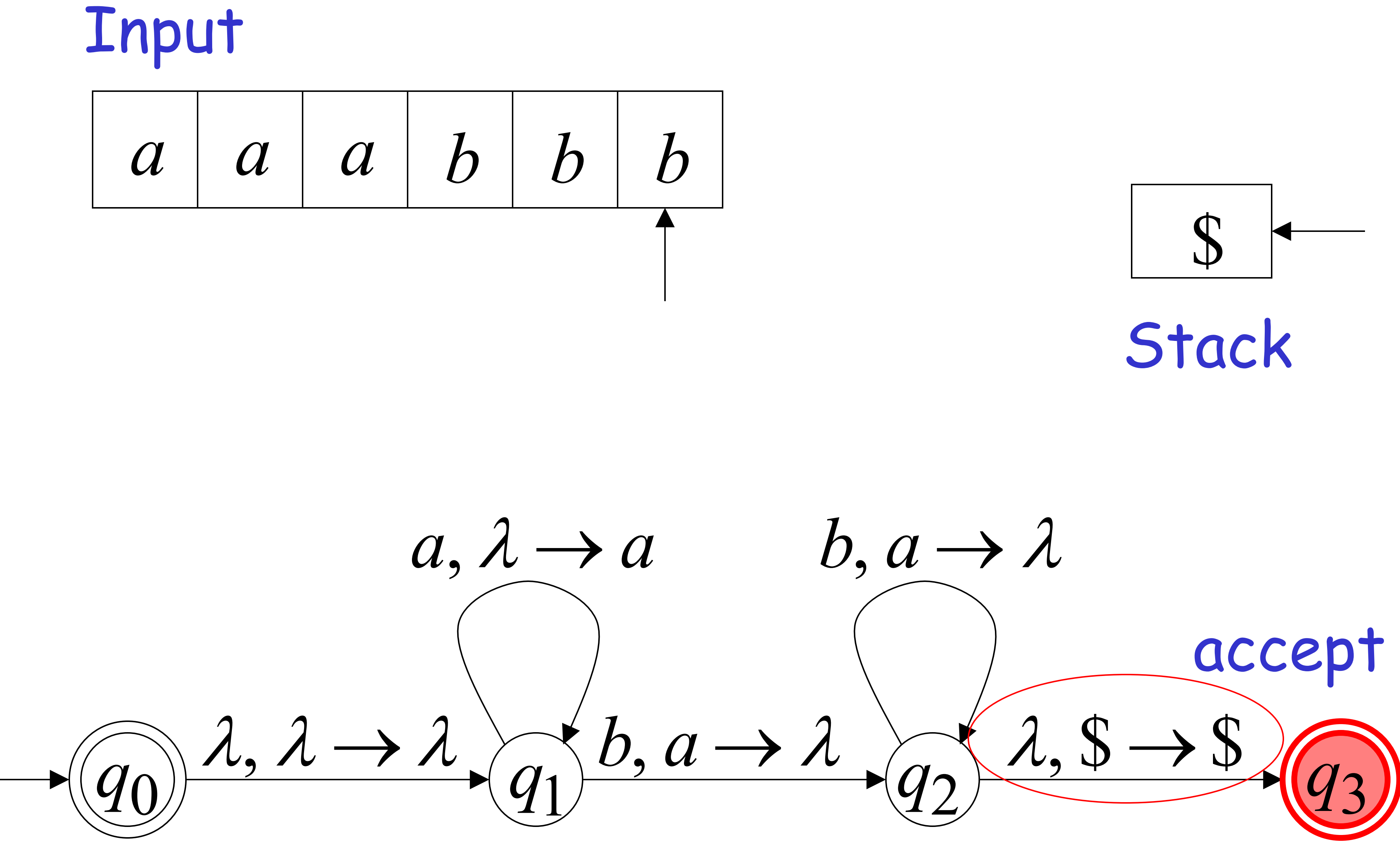
Input



Stack



Time 8


Input
a
a
a
b
b
b
$\$$
Stack
$a, \lambda \rightarrow a$
$b, a \rightarrow \lambda$
accept
$\lambda, \lambda \rightarrow \lambda$
$b, a \rightarrow \lambda$
$\lambda, \$ \rightarrow \$$
q_0
q_1
q_2
q_3

A string is accepted if there is a computation such that:

All the input is consumed

AND

The last state is an accepting state

we do not care about the stack contents at the end of the accepting computation

Rejection Example: Time 0

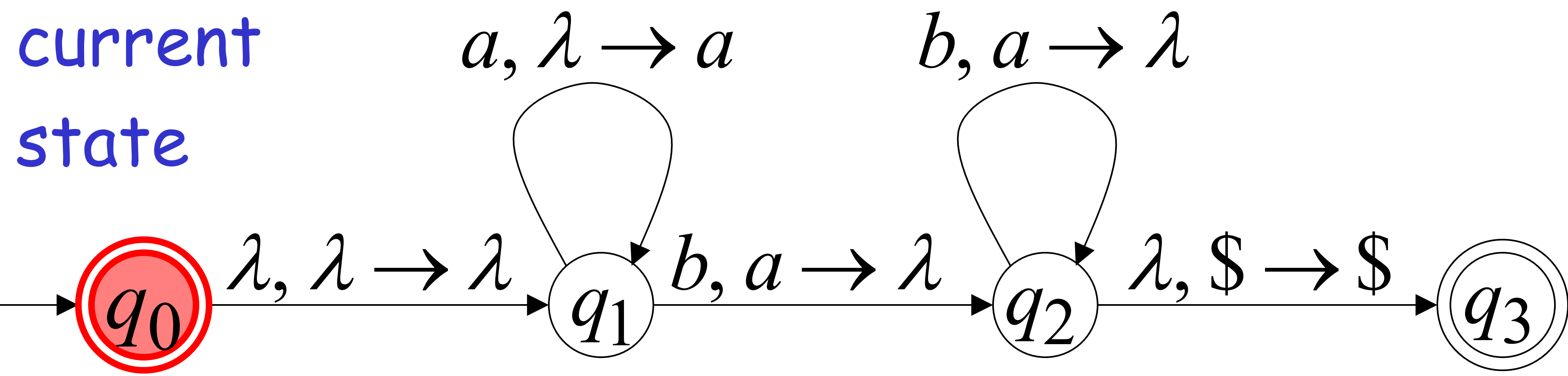
Input

a	a	b

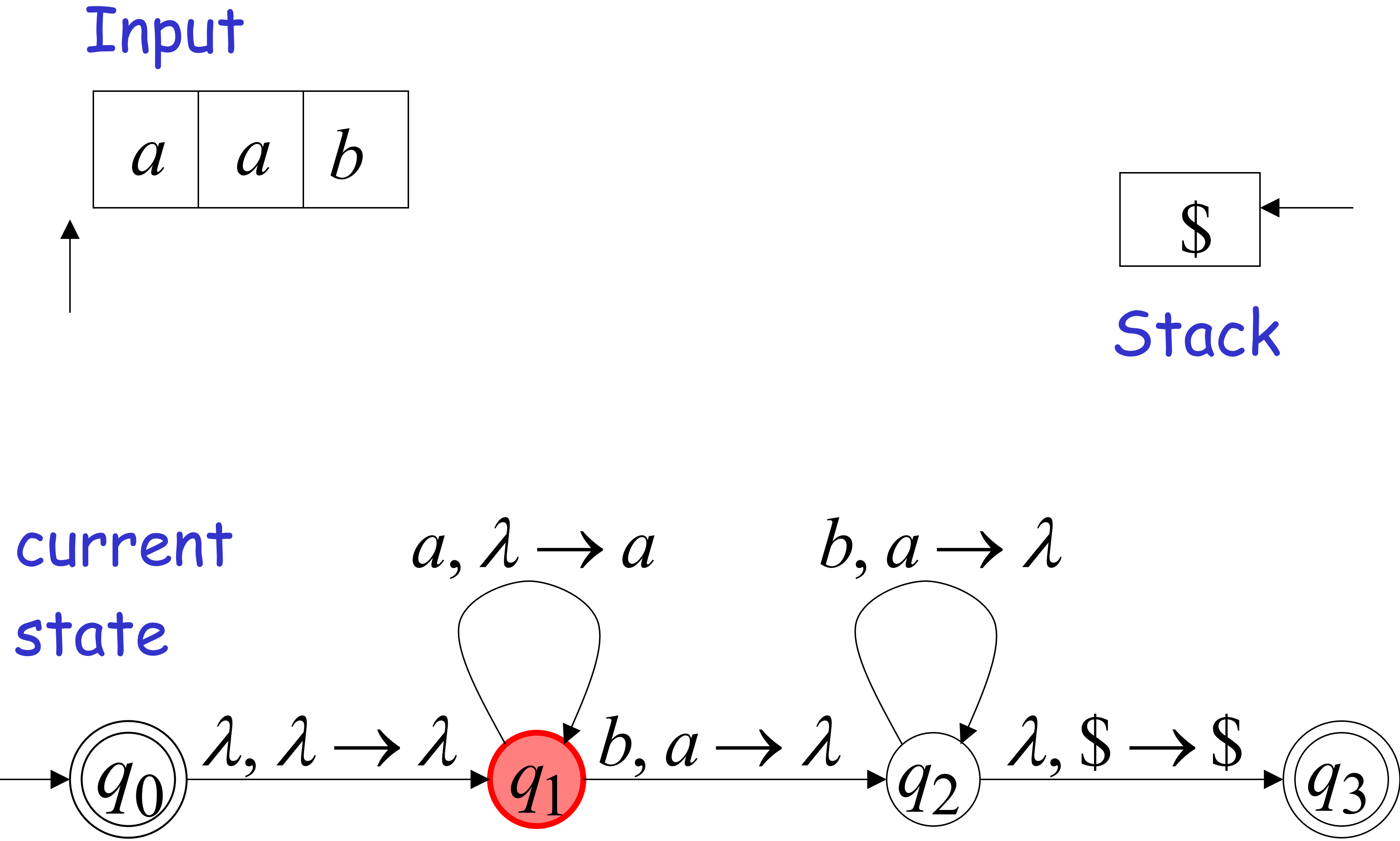
$\$$

Stack

current state

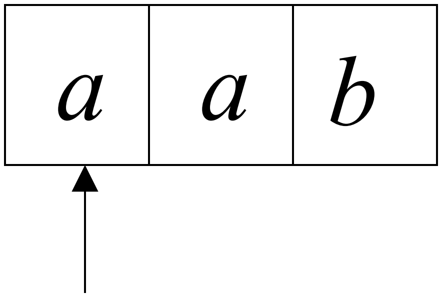


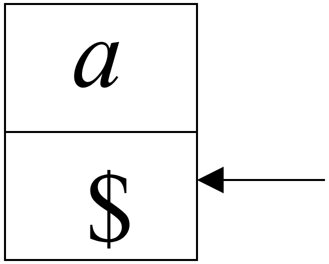
Rejection Example: Time 1



Rejection Example: Time 2

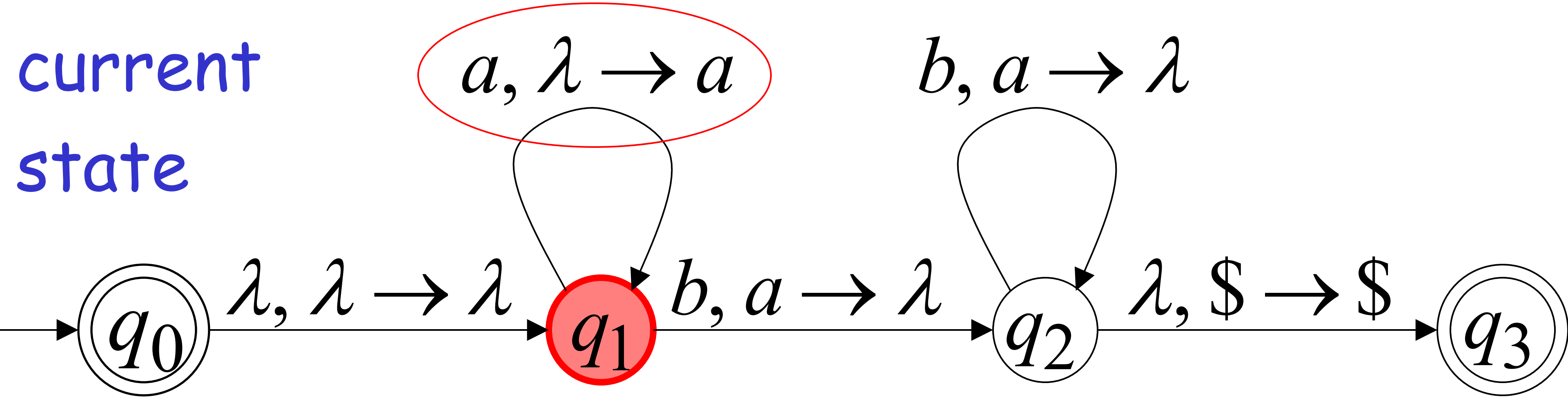
Input





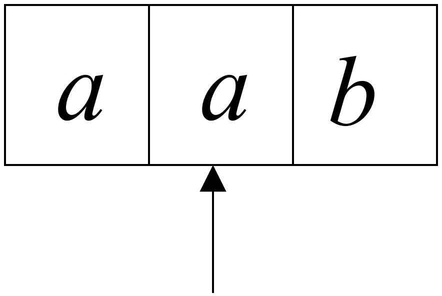
Stack

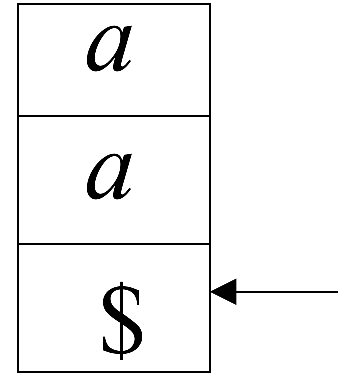
current state



Rejection Example: Time 3

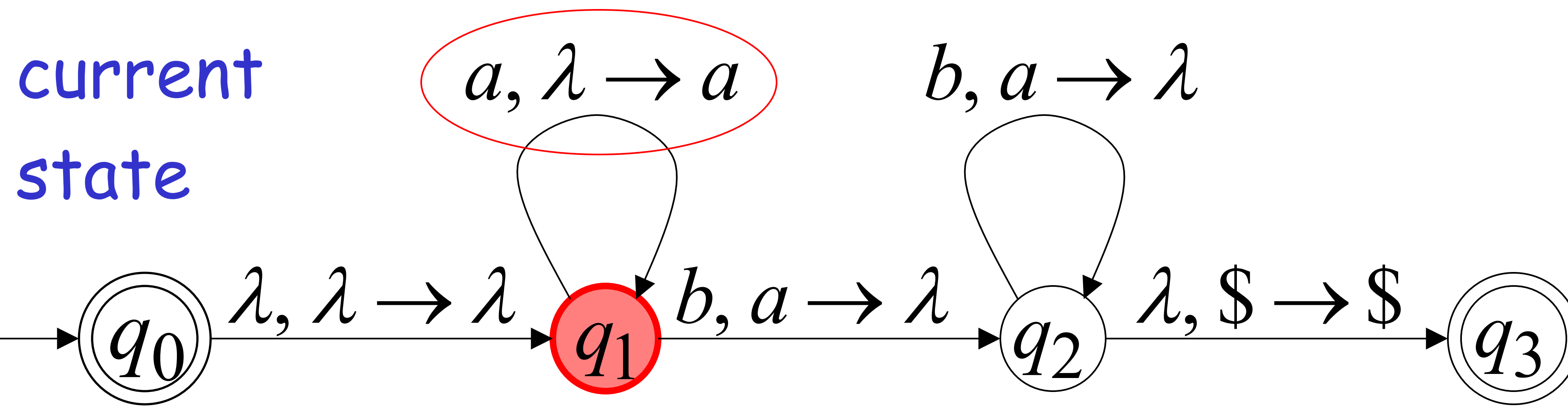
Input





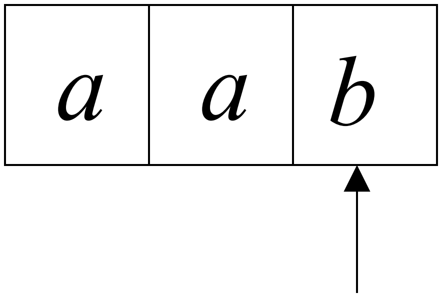
Stack

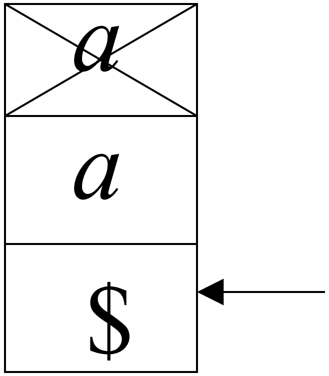
current state



Rejection Example: Time 4

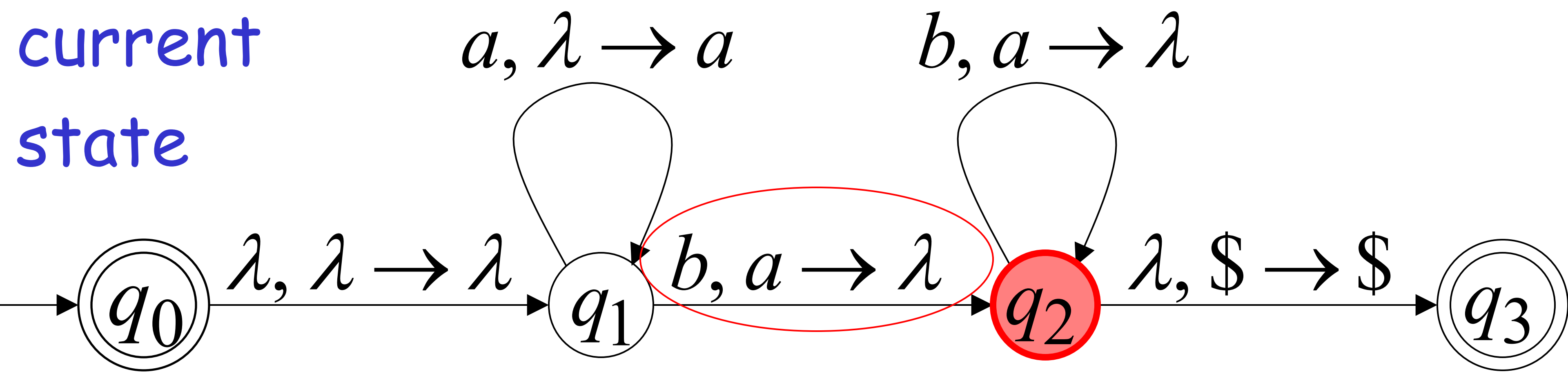
Input



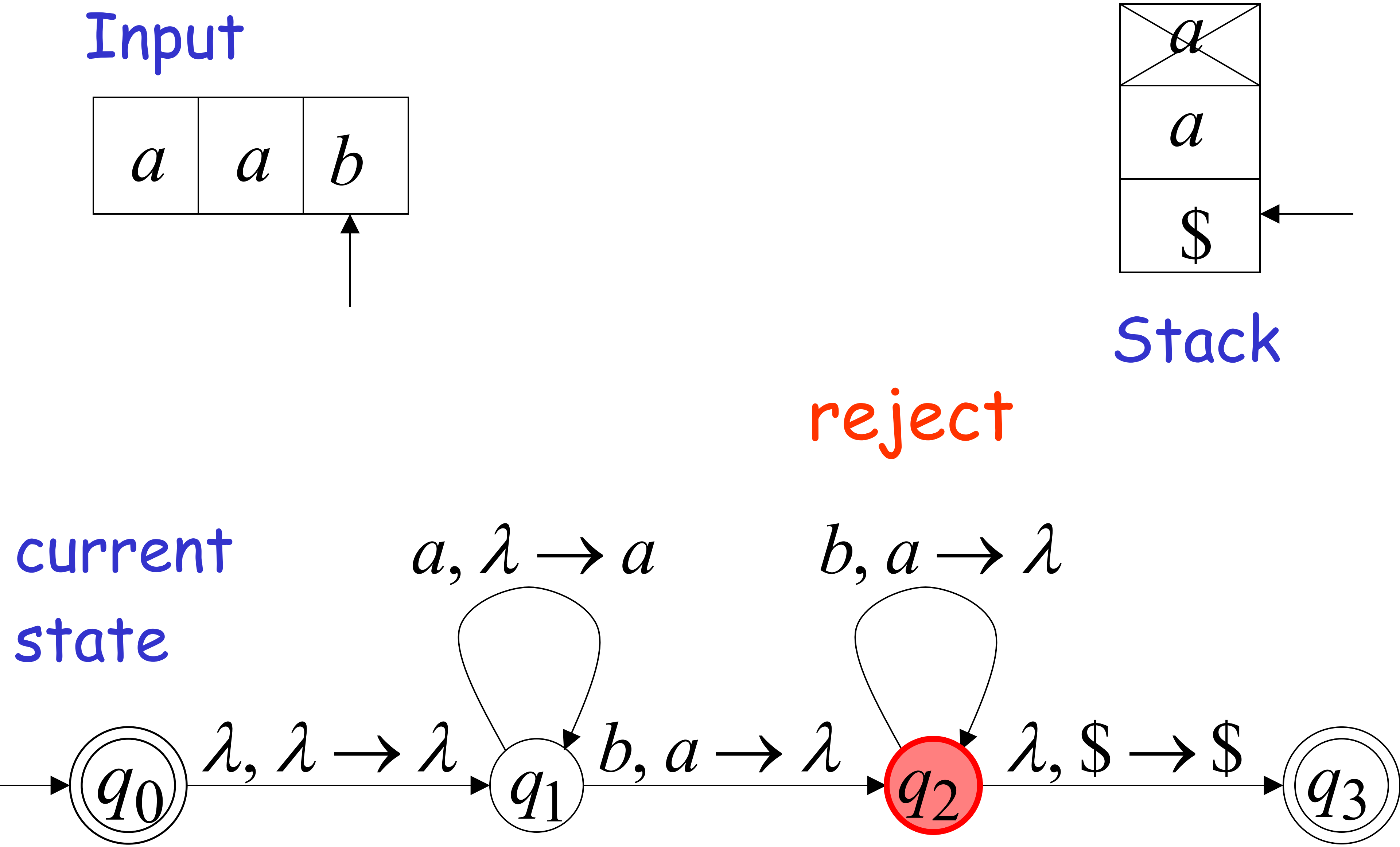


Stack

current state

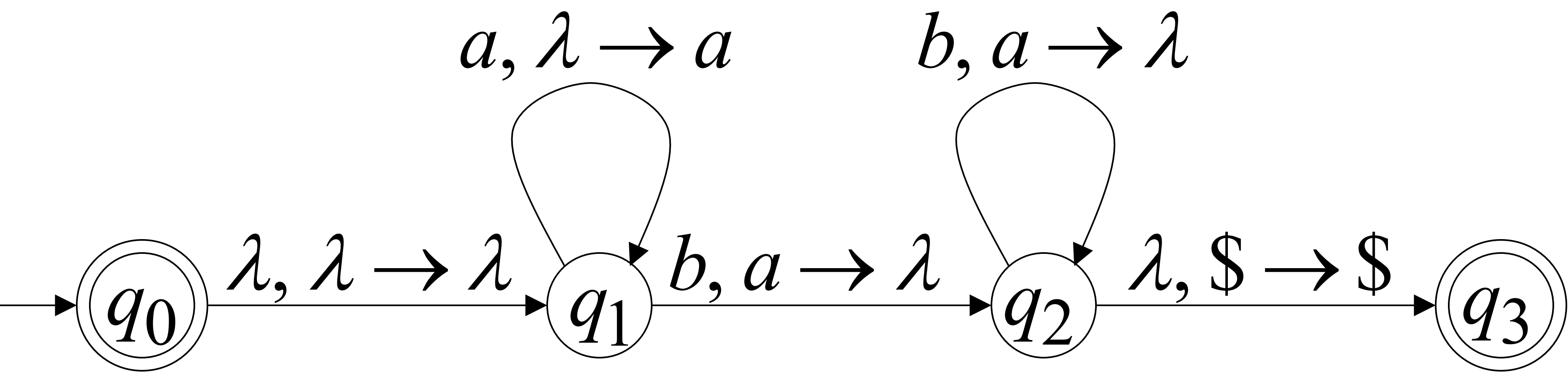


Rejection Example: Time 4

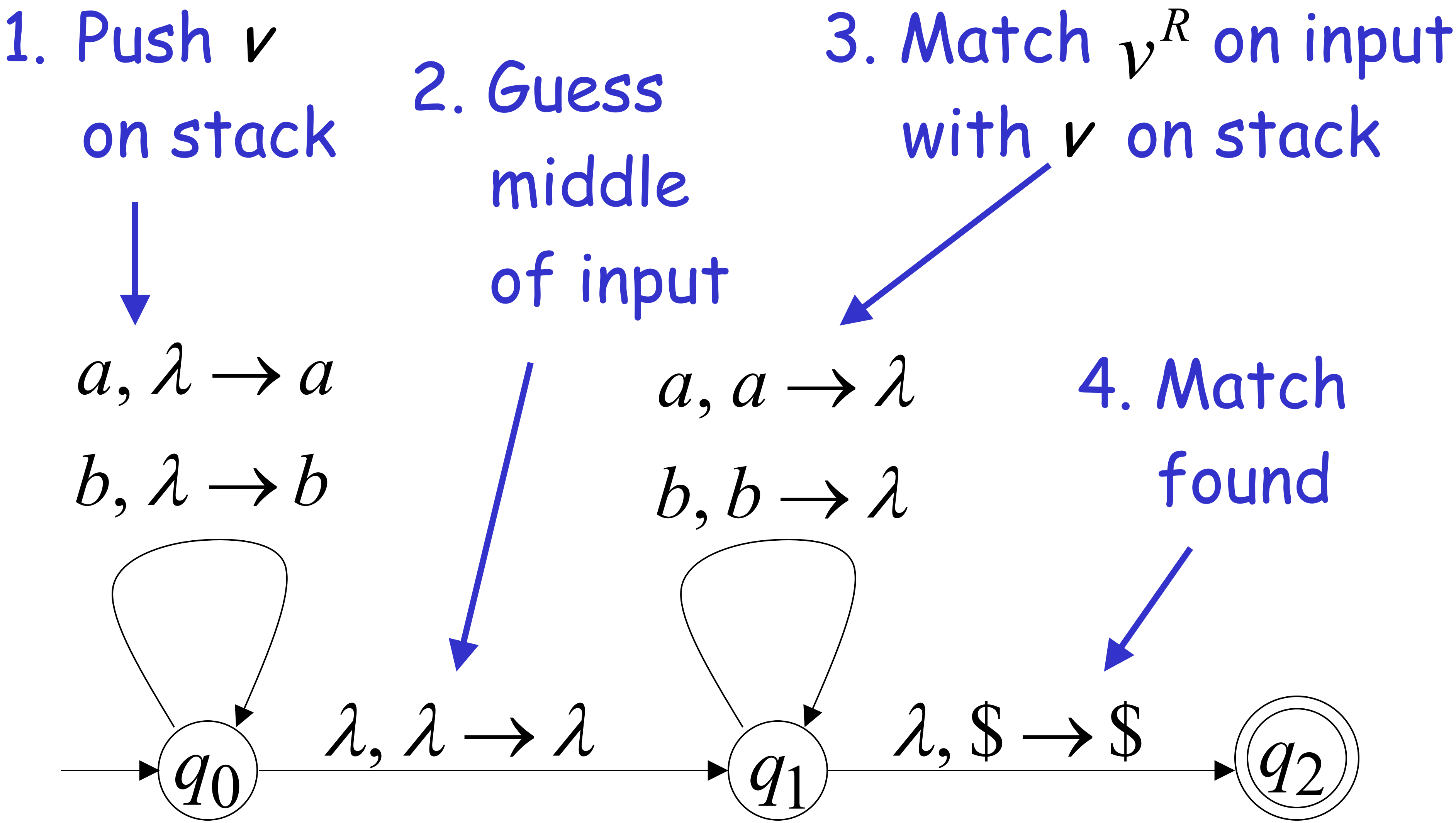


There is no accepting computation for aab

The string aab is rejected by the PDA

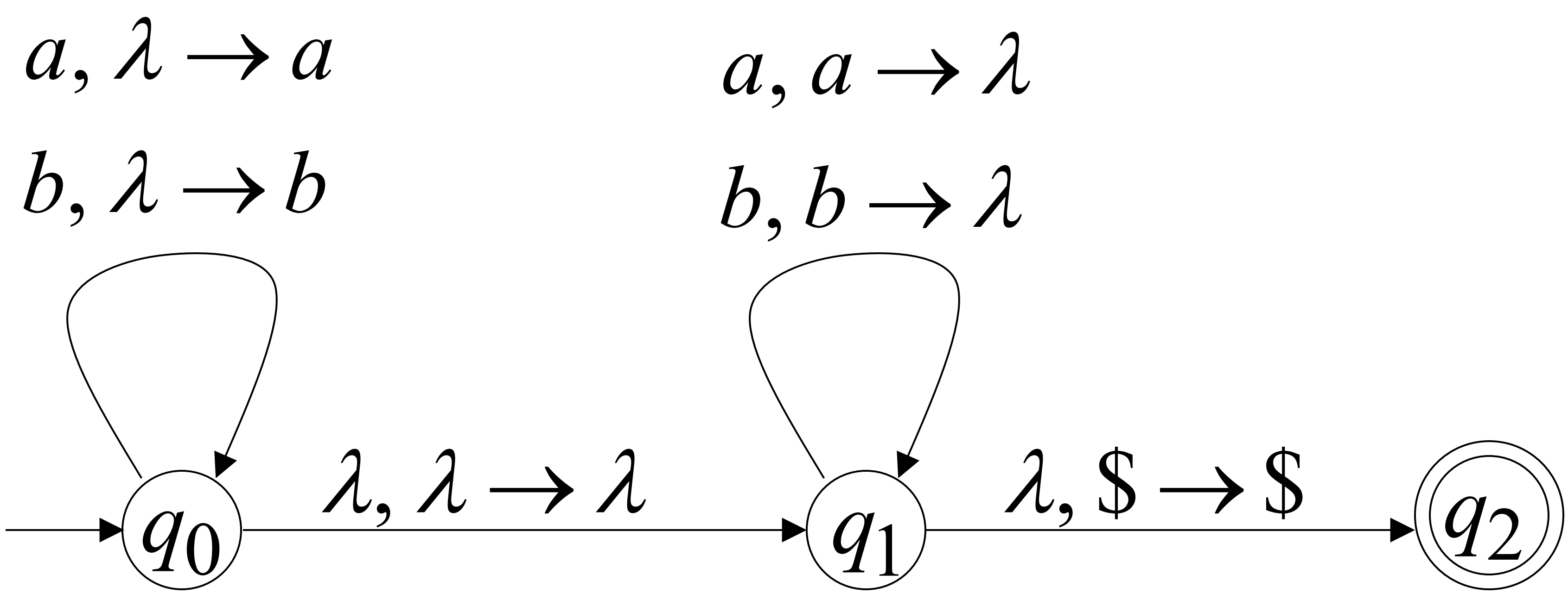


Basic Idea: $L(M) = \{vv^R : v \in \{a,b\}^*\}$

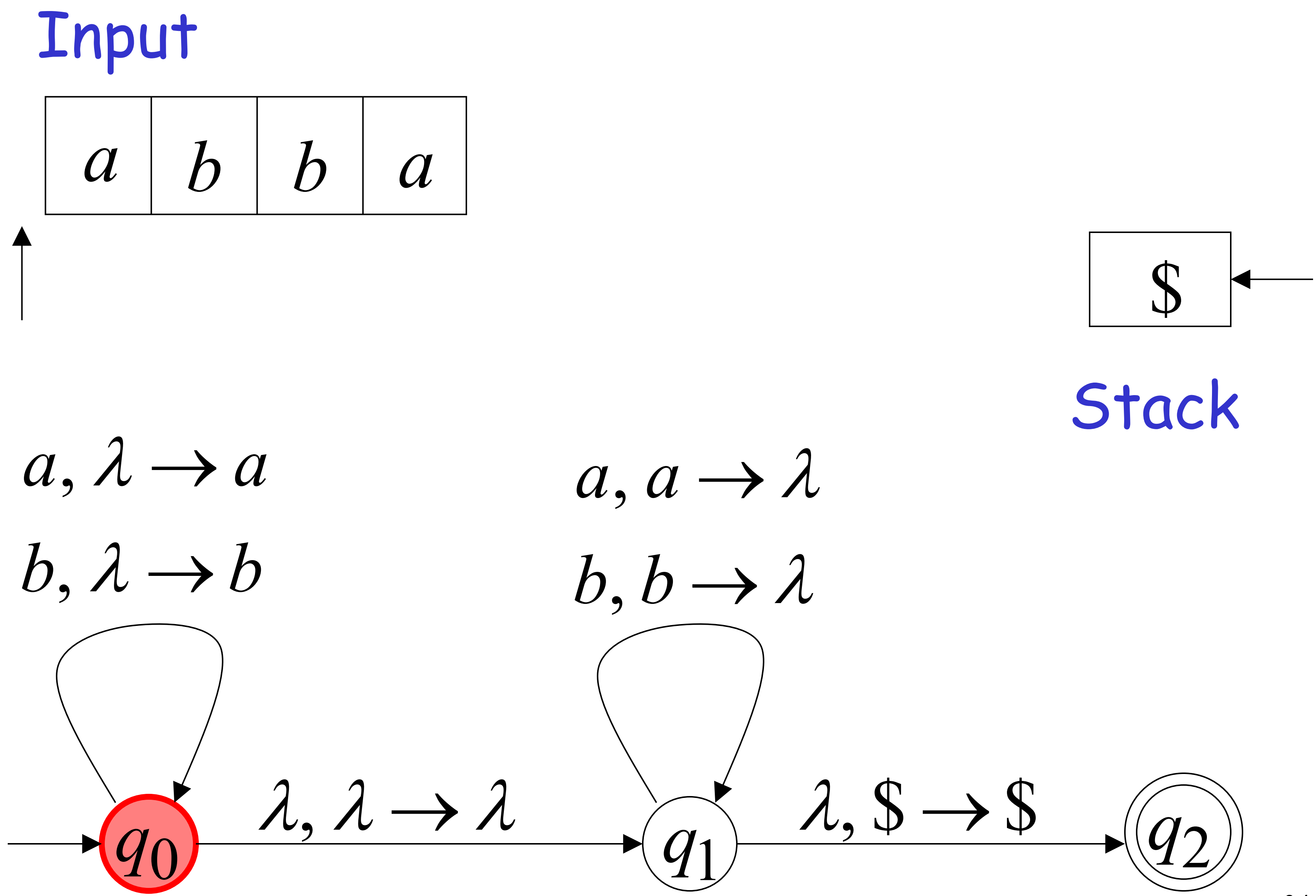


Another PDA example

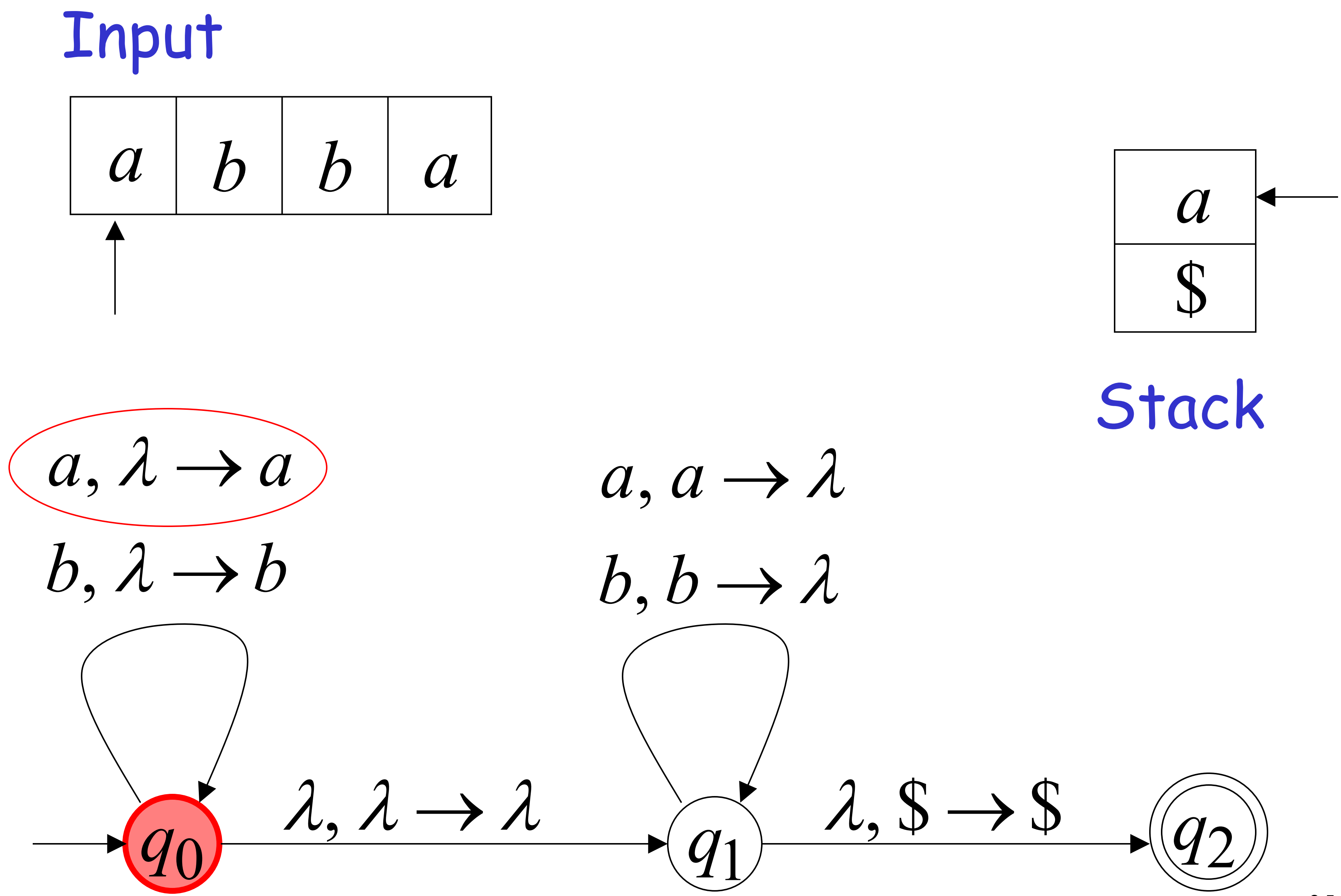
PDA M: $L(M) = \{vv^R : v \in \{a,b\}^*\}$



Execution Example: Time 0

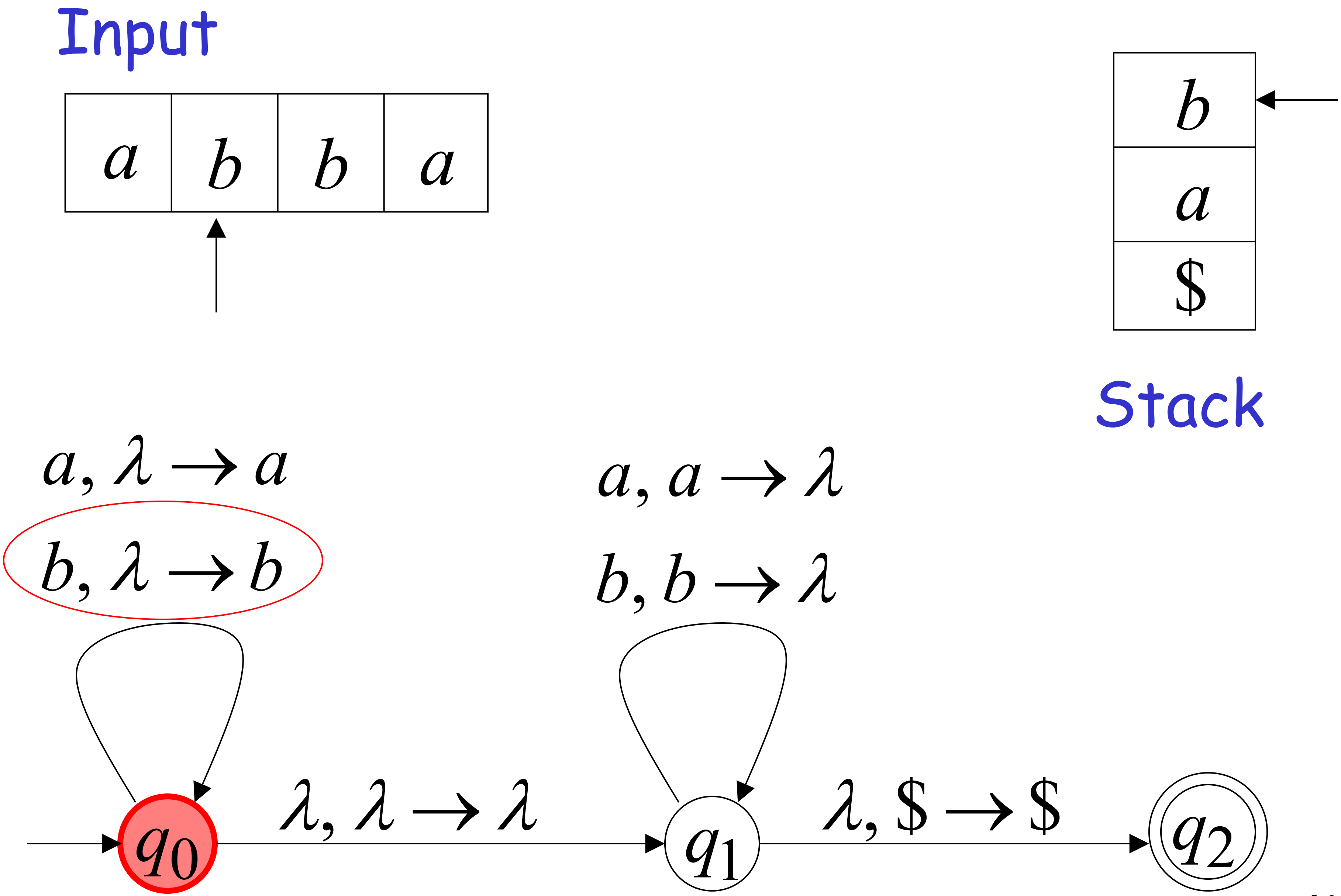


Time 1

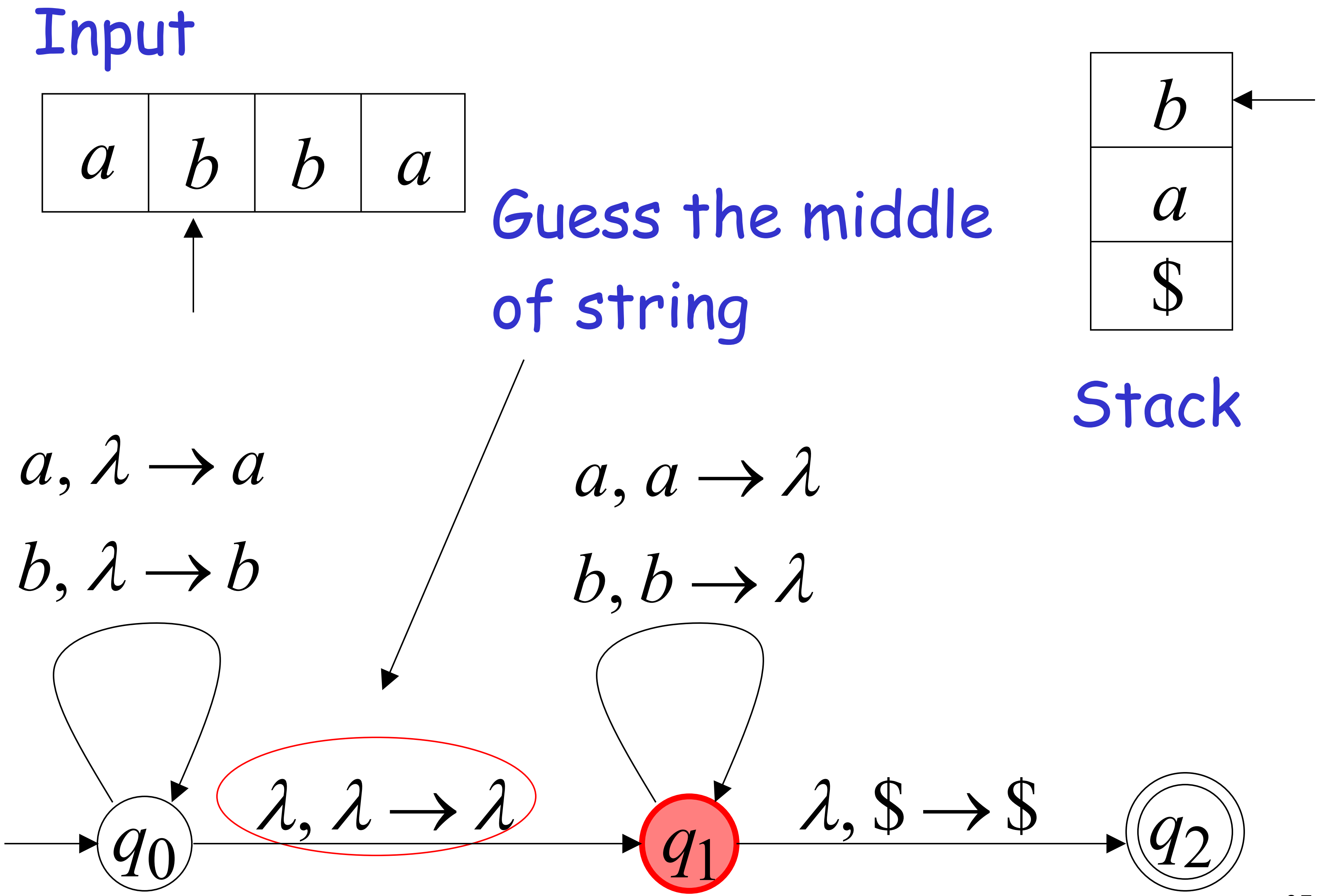


Input
a b b a
a
$\$$
Stack
$a, \lambda \to a$
$b, \lambda \to b$
$a, a \to \lambda$
$b, b \to \lambda$
$\lambda, \lambda \to \lambda$
$\lambda, \$ \to \$$
q_0
q_1
q_2

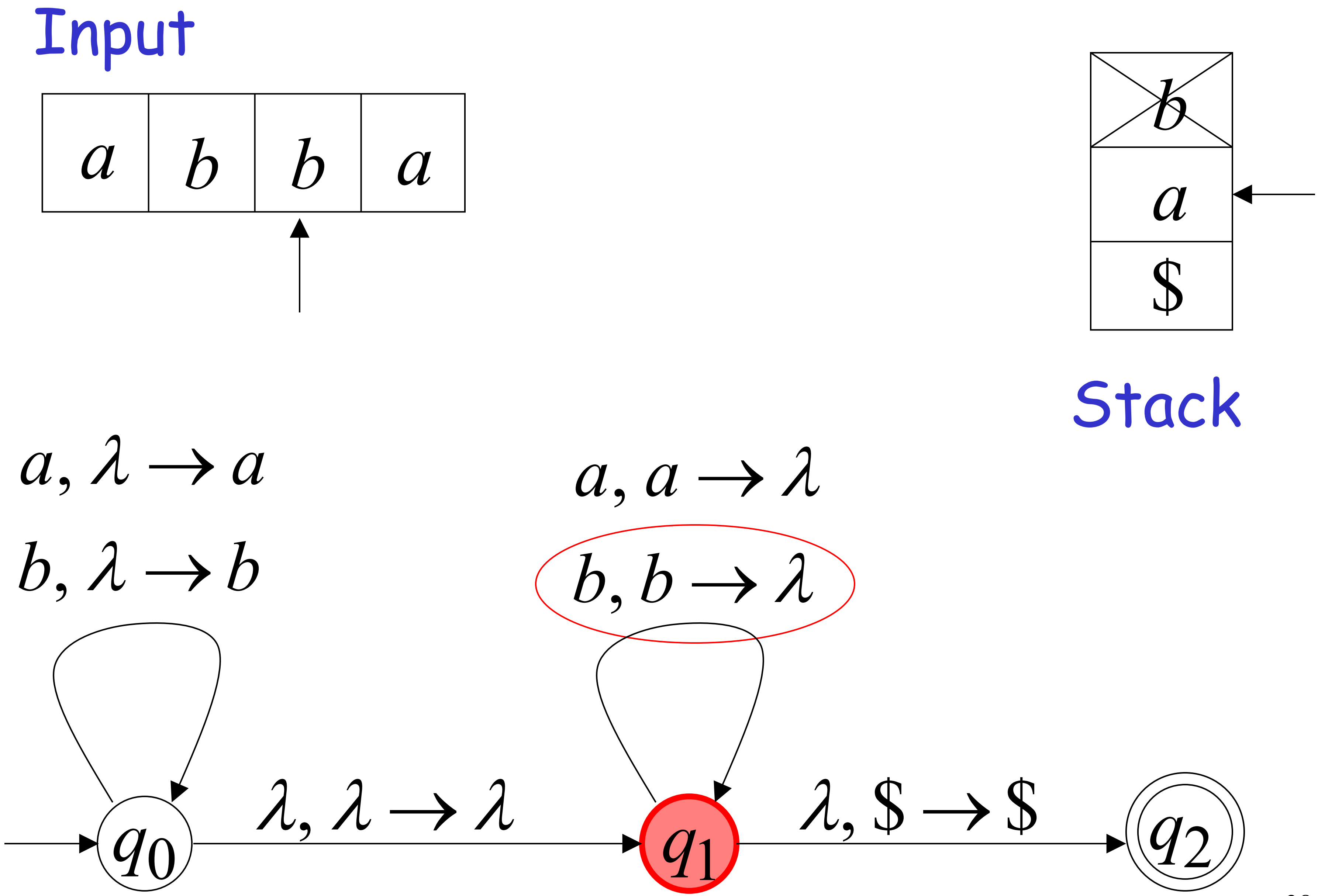
Time 2



Time 3

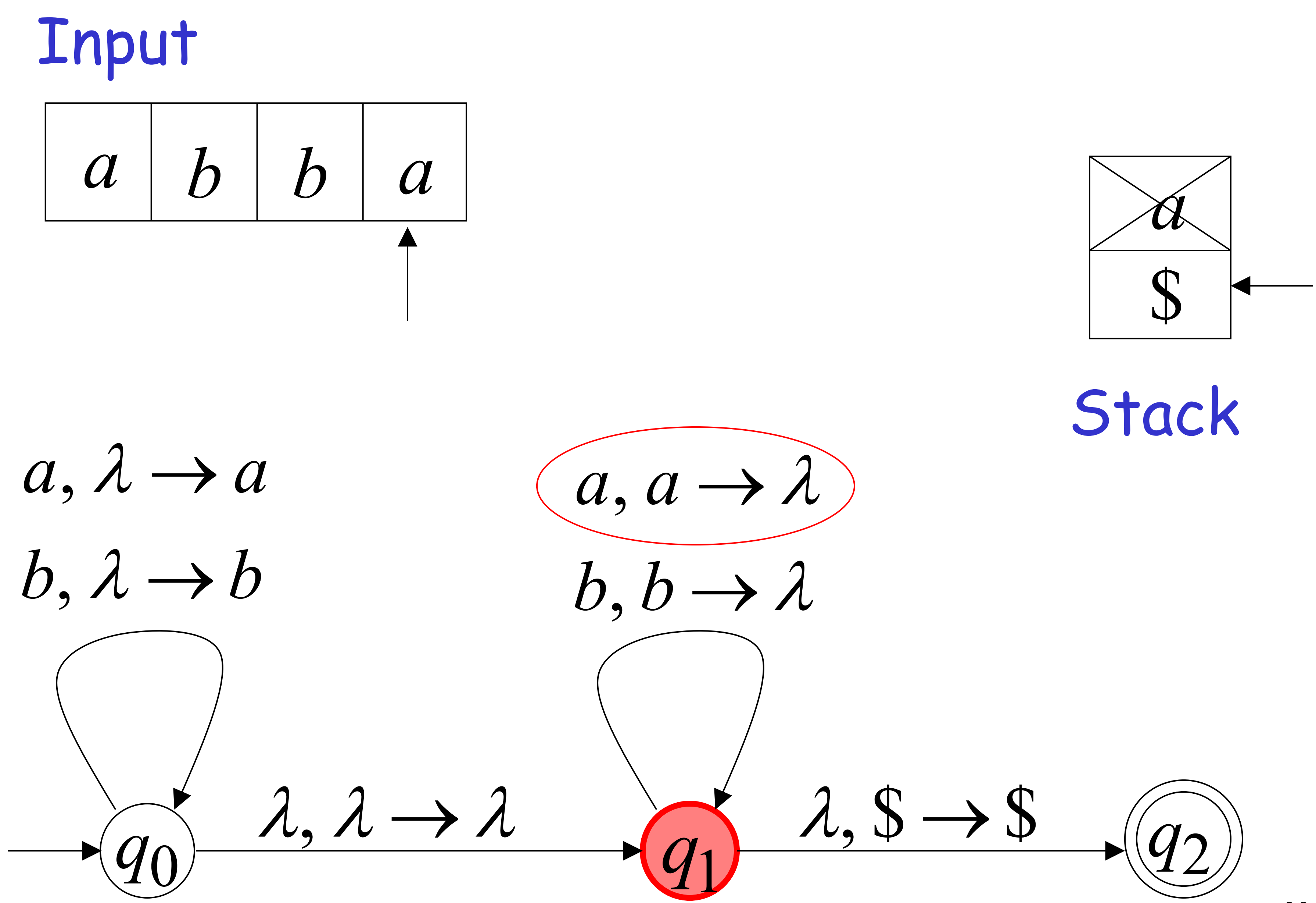

Input
a
b
b
a
Guess the middle
of string
b
a
$
Stack
a, λ → a
b, λ → b
a, a → λ
b, b → λ
λ, λ → λ
λ, $ → $
q0
q1
q2

Time 4

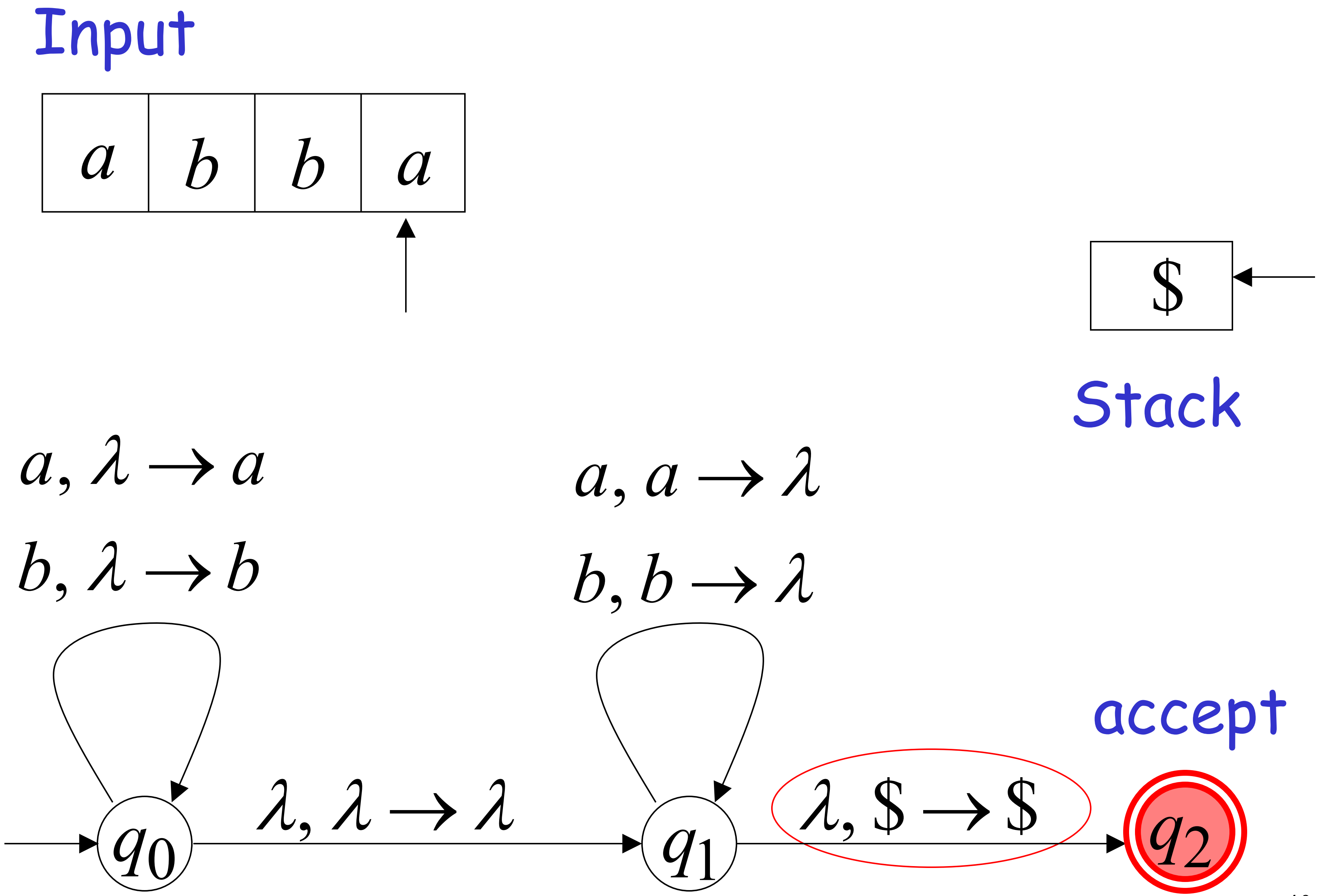


Input
a
b
b
a
b
a
$
Stack
$a, \lambda \rightarrow a$
$b, \lambda \rightarrow b$
$a, a \rightarrow \lambda$
$b, b \rightarrow \lambda$
$\lambda, \lambda \rightarrow \lambda$
$\lambda, \$ \rightarrow \$$
q_0
q_1
q_2

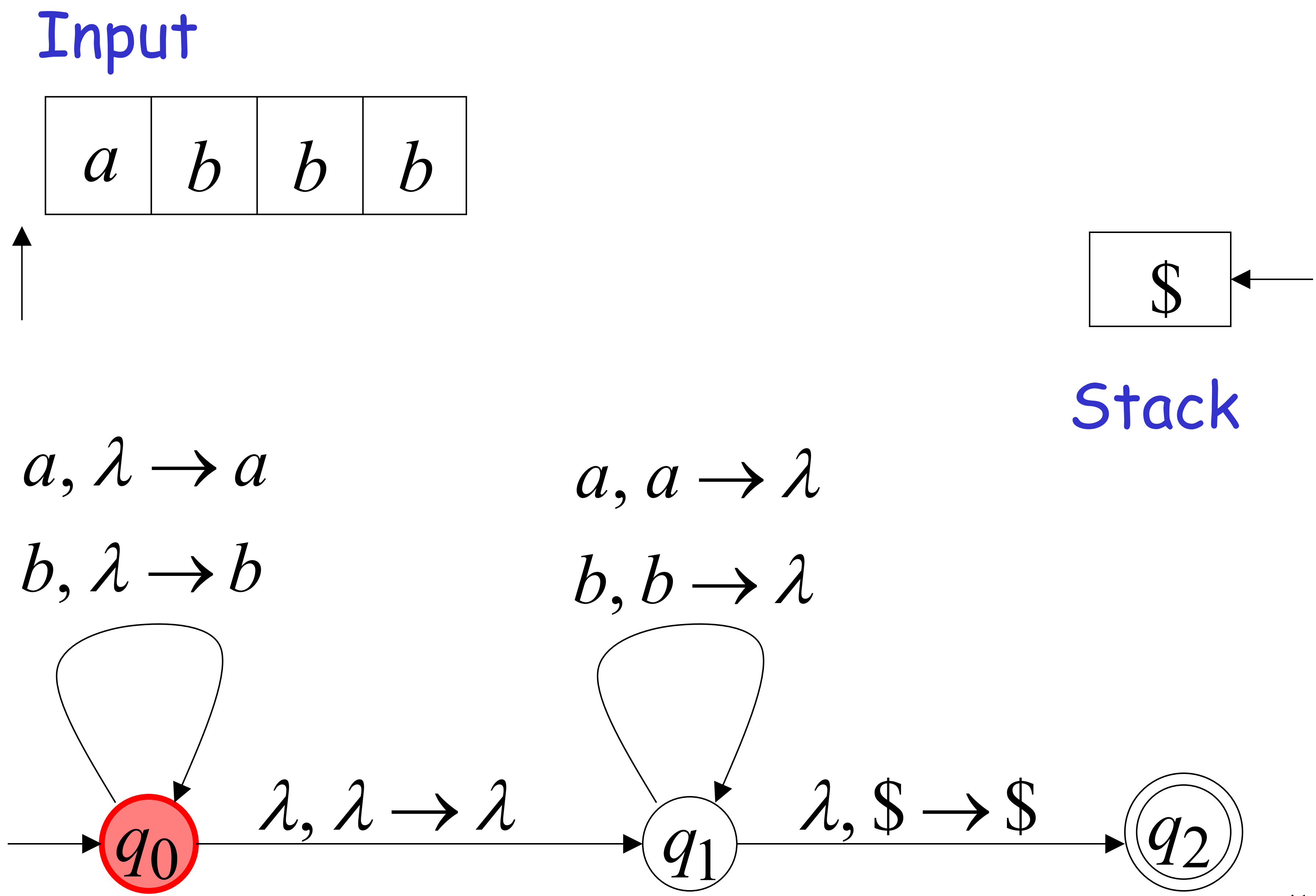
Time 5



Time 6


Input
a b b a
$\$$
Stack
$a, \lambda \rightarrow a$
$b, \lambda \rightarrow b$
$a, a \rightarrow \lambda$
$b, b \rightarrow \lambda$
$\lambda, \lambda \rightarrow \lambda$
$\lambda, \$ \rightarrow \$$
accept
q_0
q_1
q_2

Rejection Example: Time 0



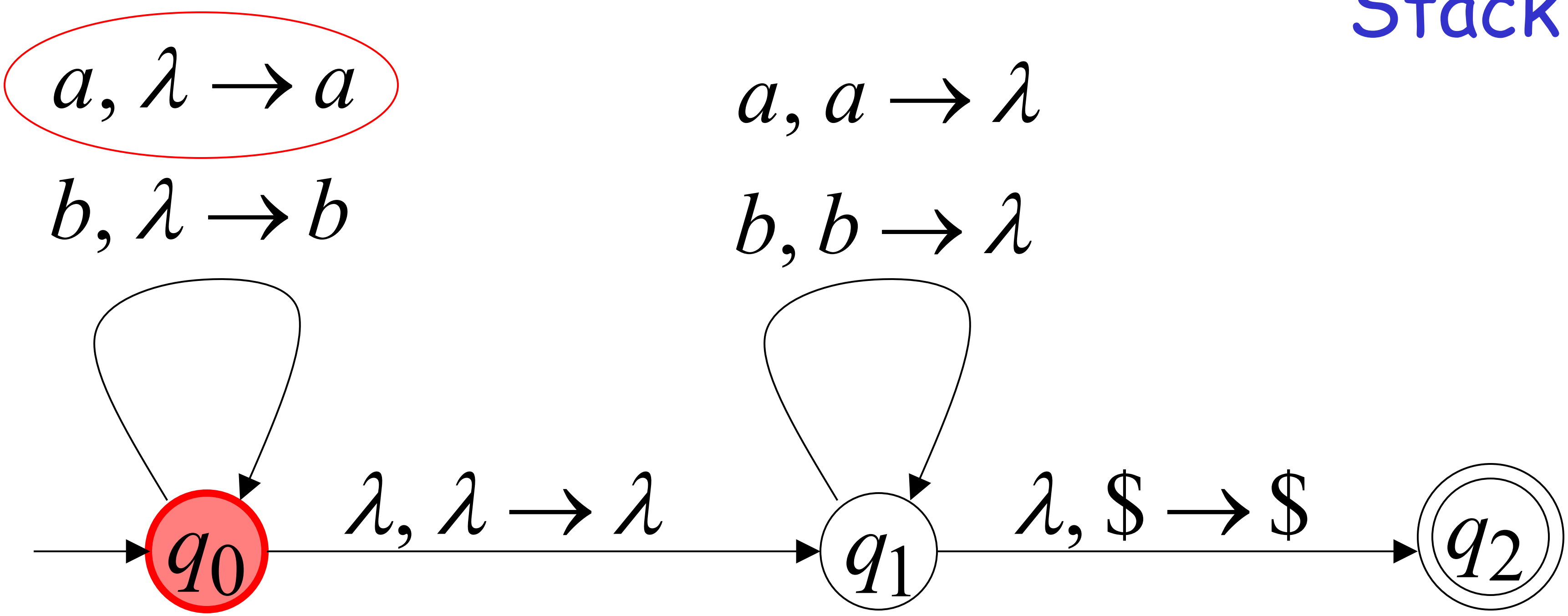
Time 1

Input

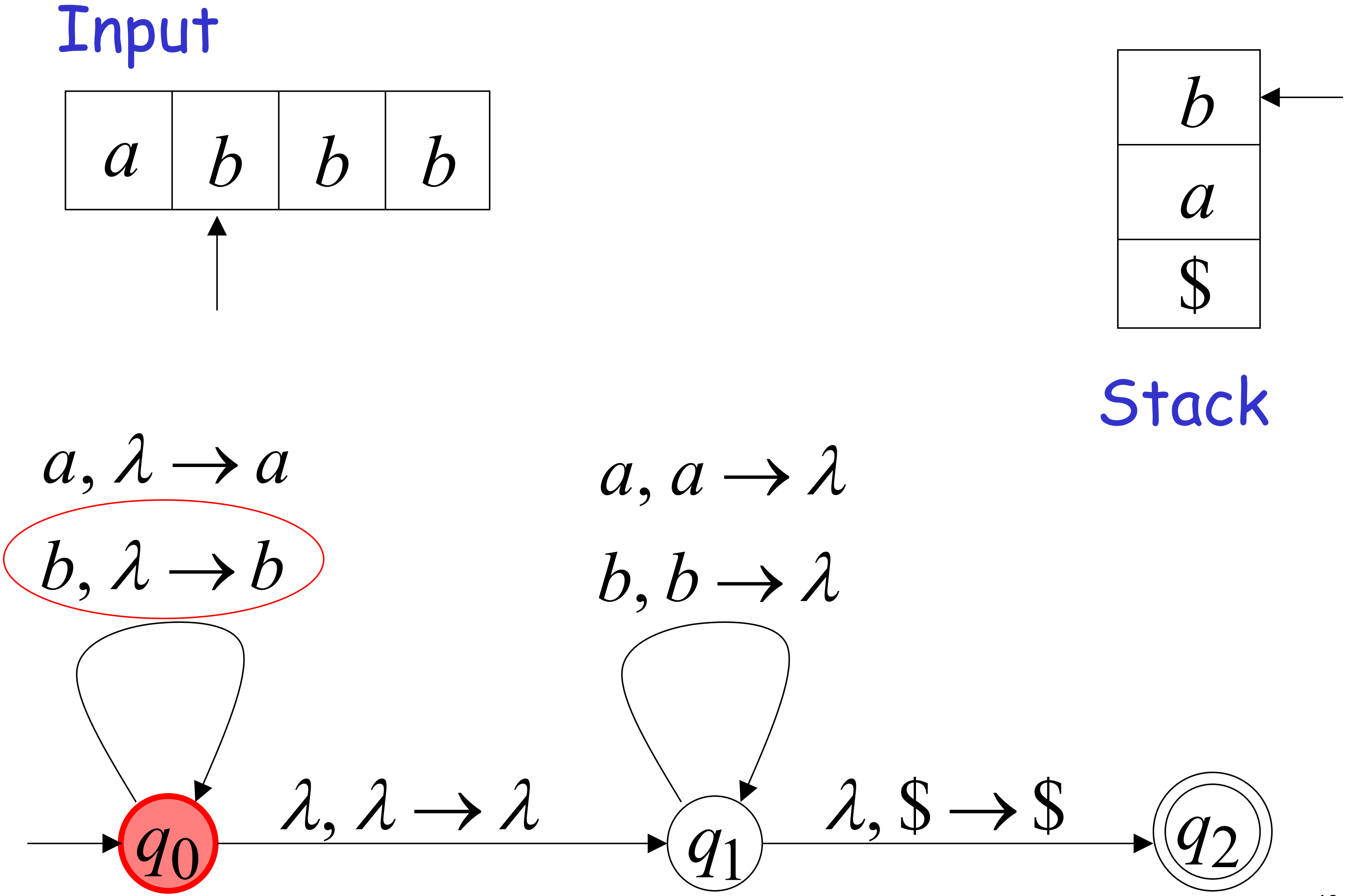
a	b	b	b

a
$\$$

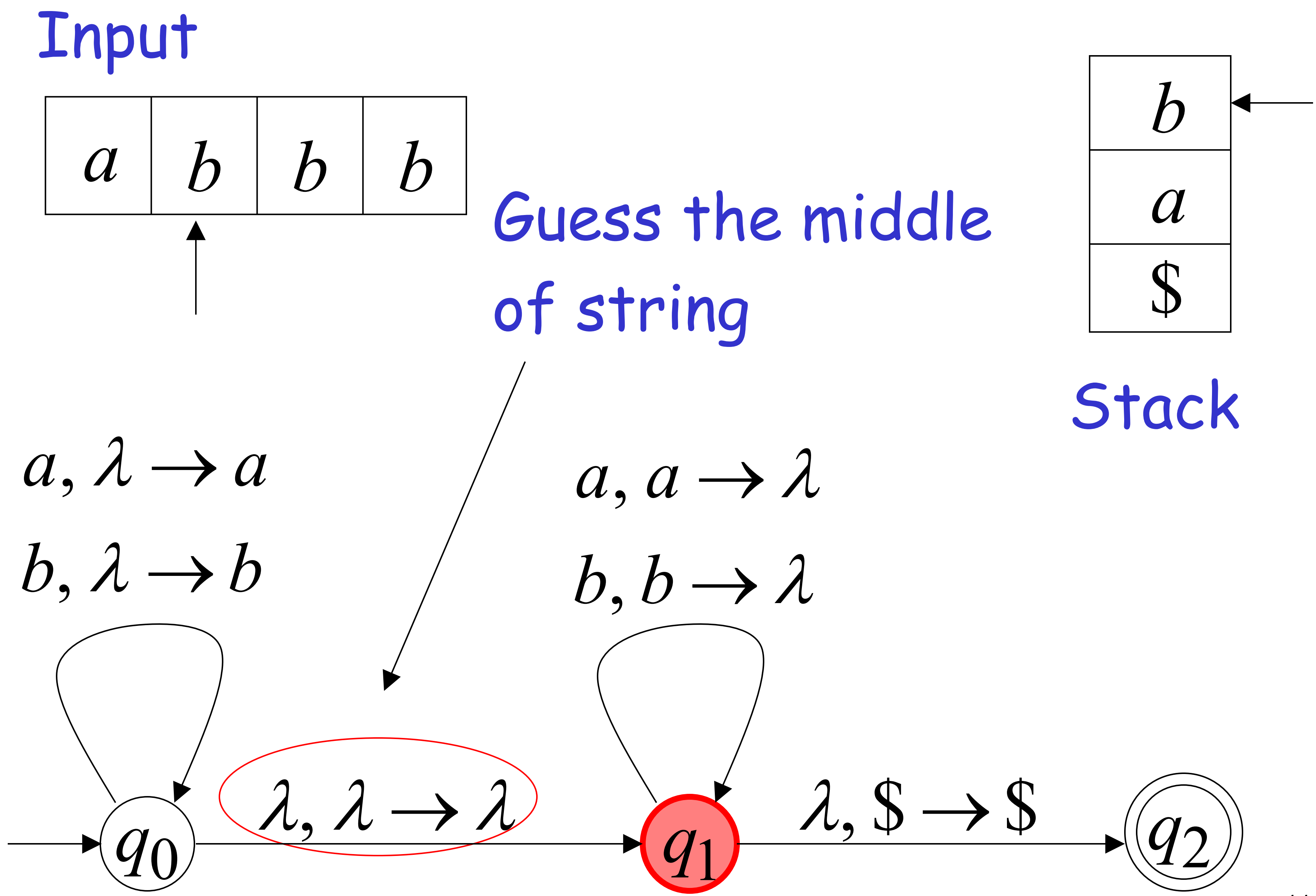
Stack



Time 2

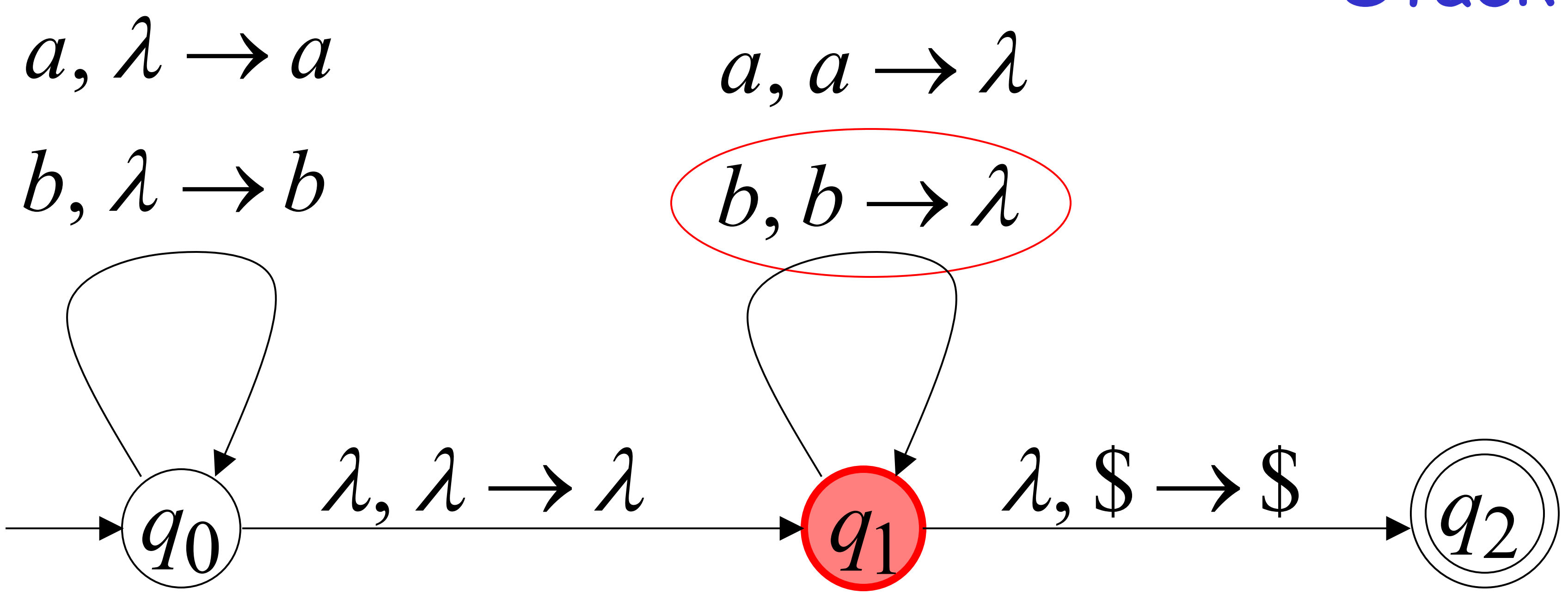

Input
a b b b
b
a
$
Stack
$a, \lambda \rightarrow a$
$b, \lambda \rightarrow b$
$a, a \rightarrow \lambda$
$b, b \rightarrow \lambda$
$\lambda, \lambda \rightarrow \lambda$
$\lambda, \$ \rightarrow \$$
q_0
q_1
q_2

Time 3


Input
a
b
b
b
Guess the middle
of string
b
a
$
Stack
a, λ → a
b, λ → b
a, a → λ
b, b → λ
λ, λ → λ
λ, $ → $
q0
q1
q2

Time 4

Input
a
b
b
b
b
a
$
Stack



$a, \lambda \rightarrow a$
$b, \lambda \rightarrow b$
$a, a \rightarrow \lambda$
$b, b \rightarrow \lambda$
$\lambda, \lambda \rightarrow \lambda$
$\lambda, \$ \rightarrow \$$
q_0
q_1
q_2

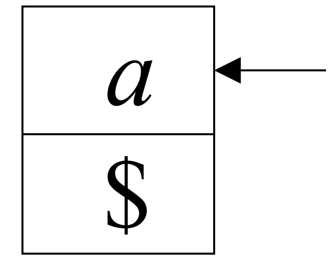
Time 5

There is no possible transition.

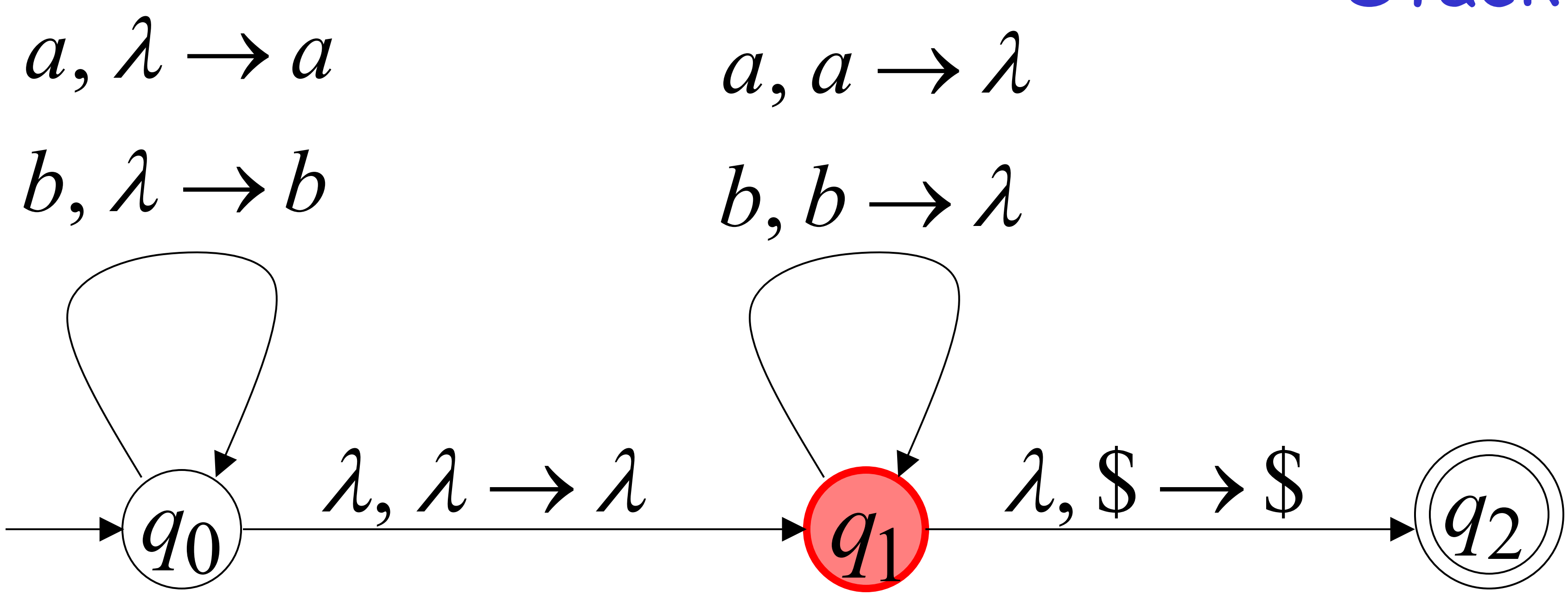
Input

a	b	b	b

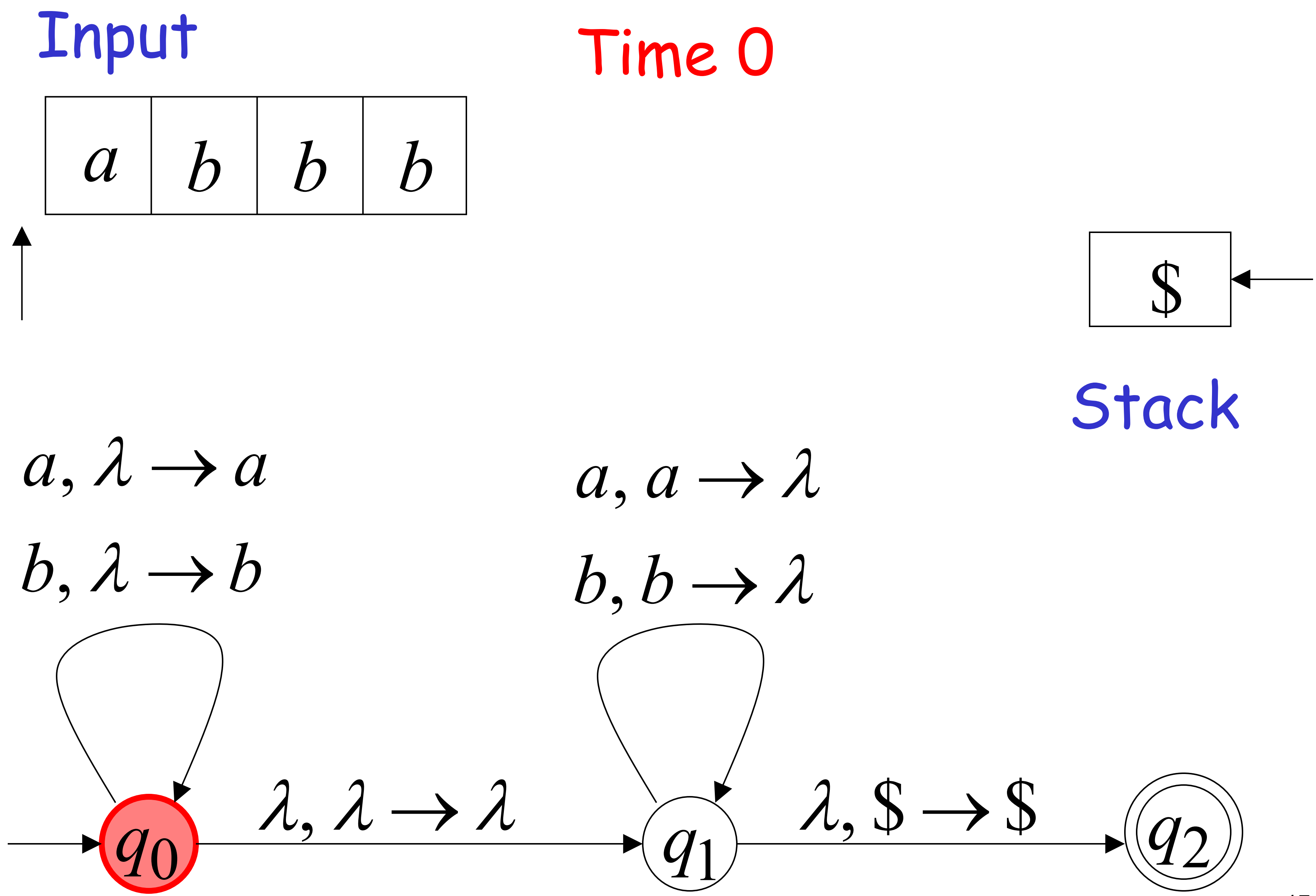
Input is not consumed



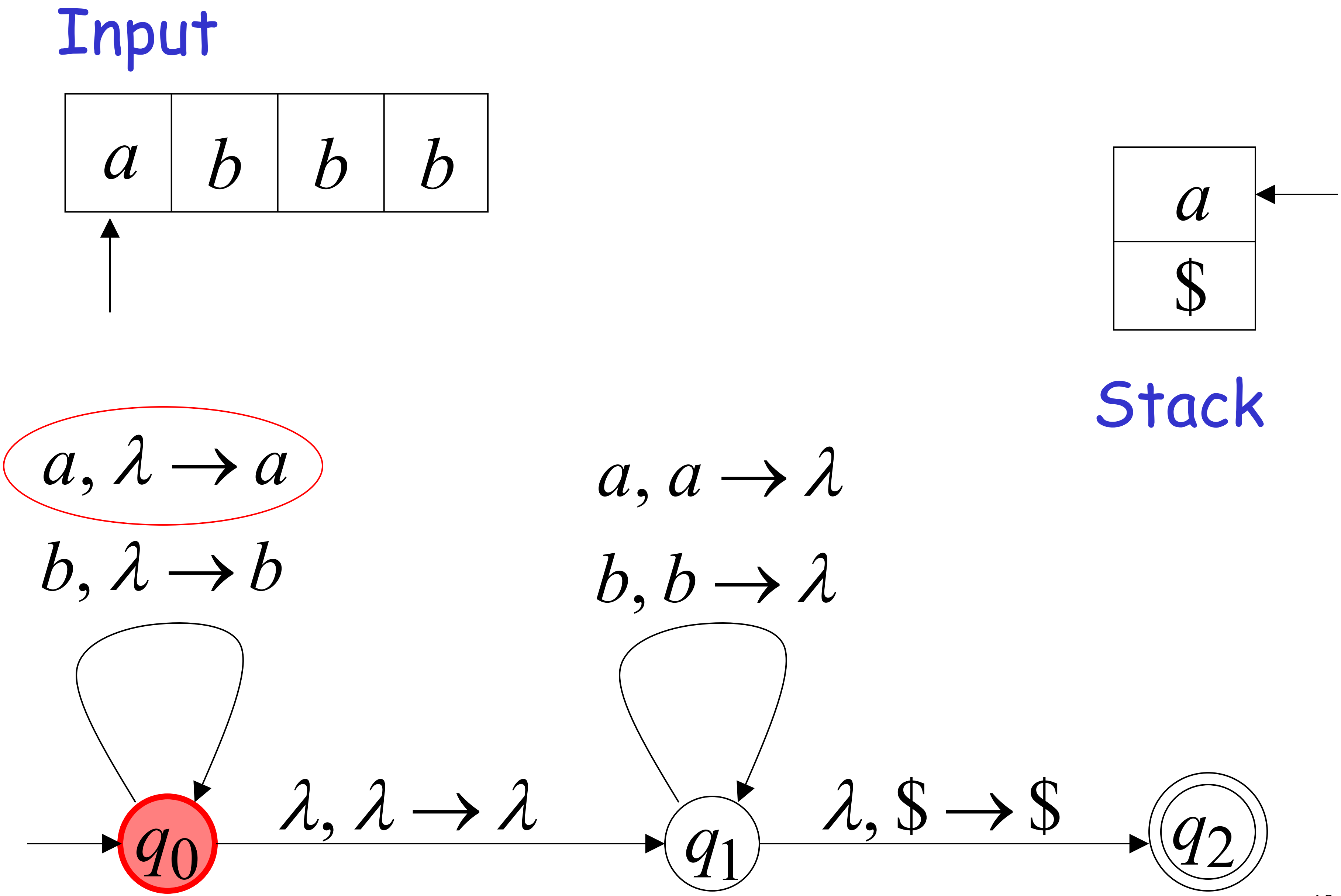
Stack



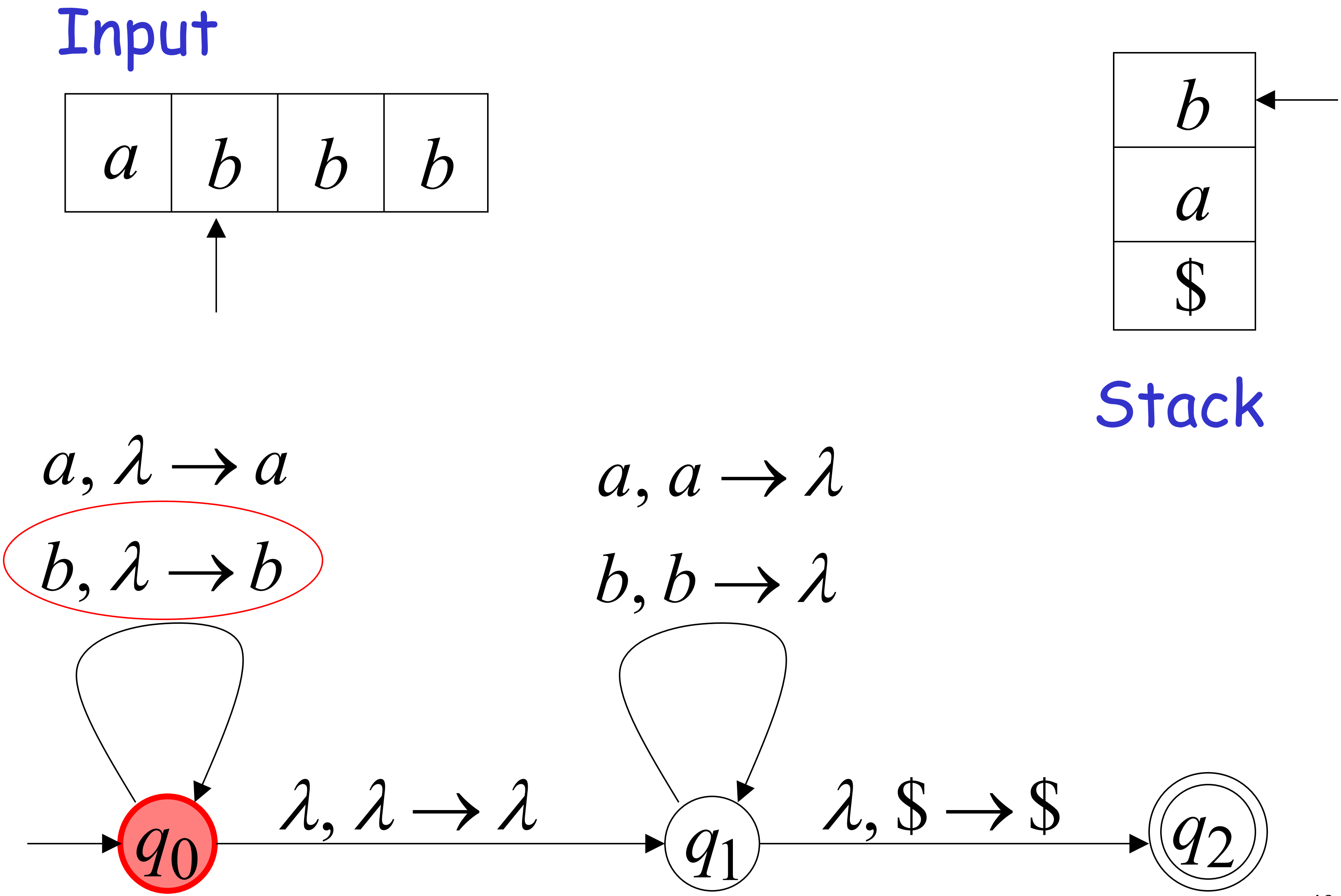
Another computation on same string:



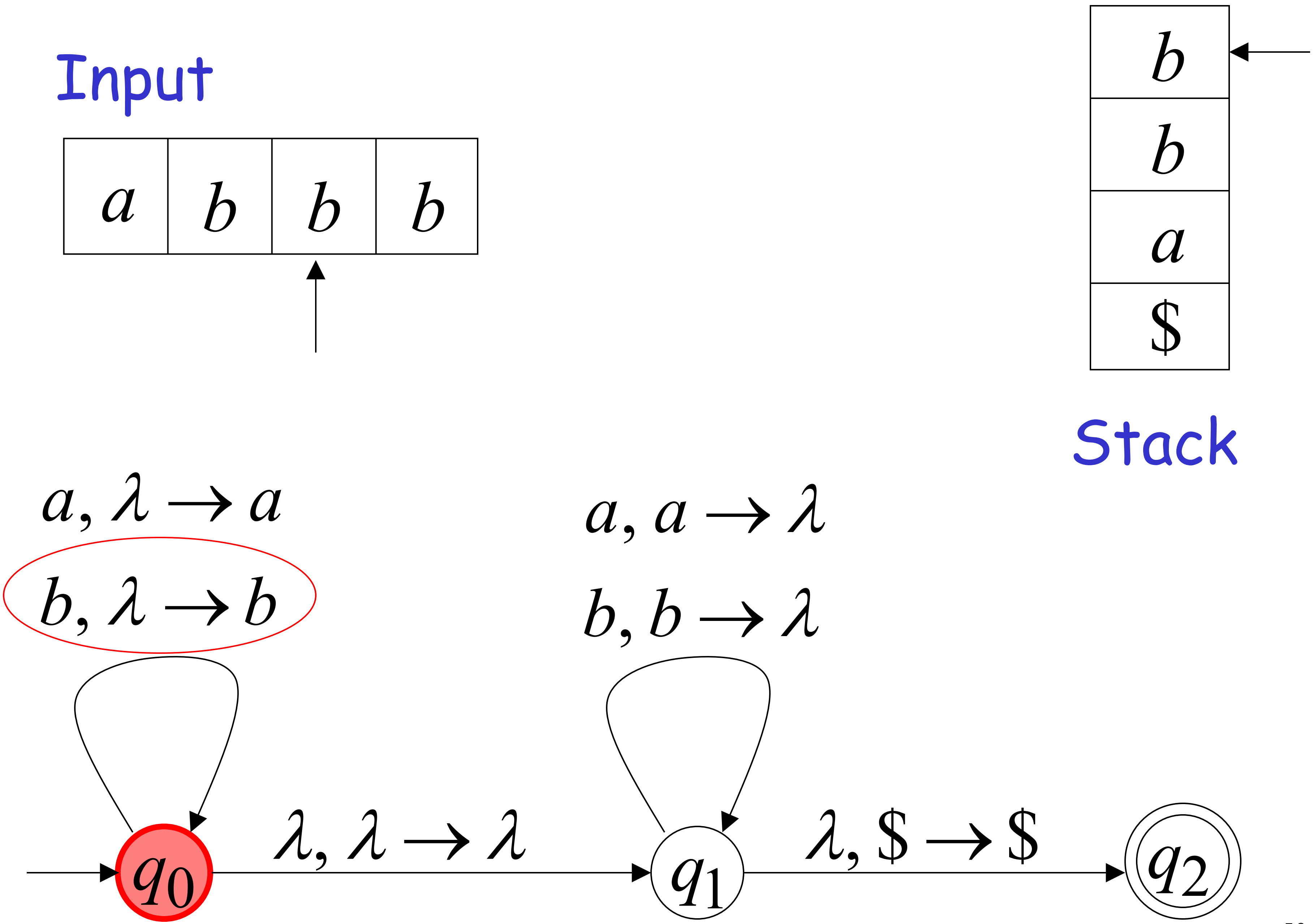
Time 1


Input
a b b b
a
$\$$
Stack
$a, \lambda \to a$
$b, \lambda \to b$
$a, a \to \lambda$
$b, b \to \lambda$
$\lambda, \lambda \to \lambda$
$\lambda, \$ \to \$$
q_0
q_1
q_2

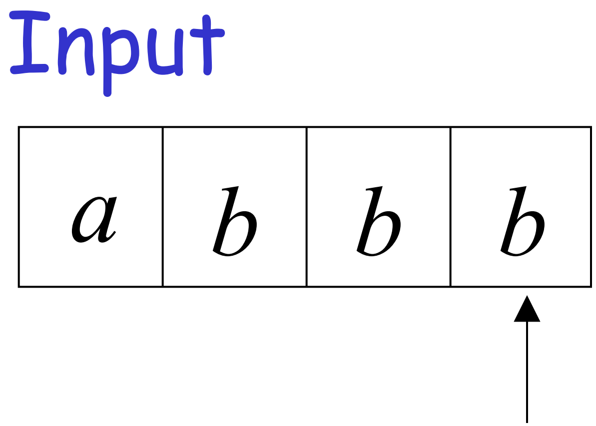
Time 2

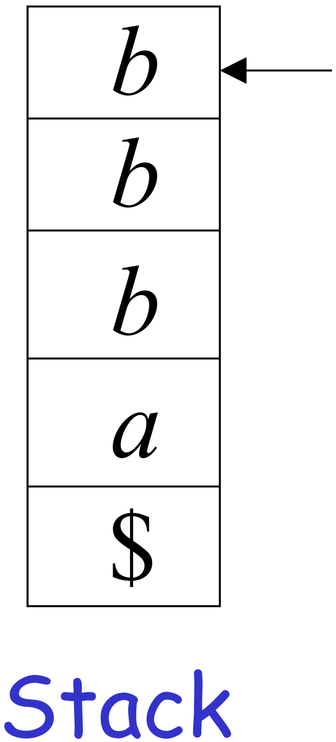

Input
a
b
b
b
b
a
$
Stack
$a, \lambda \to a$
$b, \lambda \to b$
$a, a \to \lambda$
$b, b \to \lambda$
$\lambda, \lambda \to \lambda$
$\lambda, \$ \to \$$
q_0
q_1
q_2

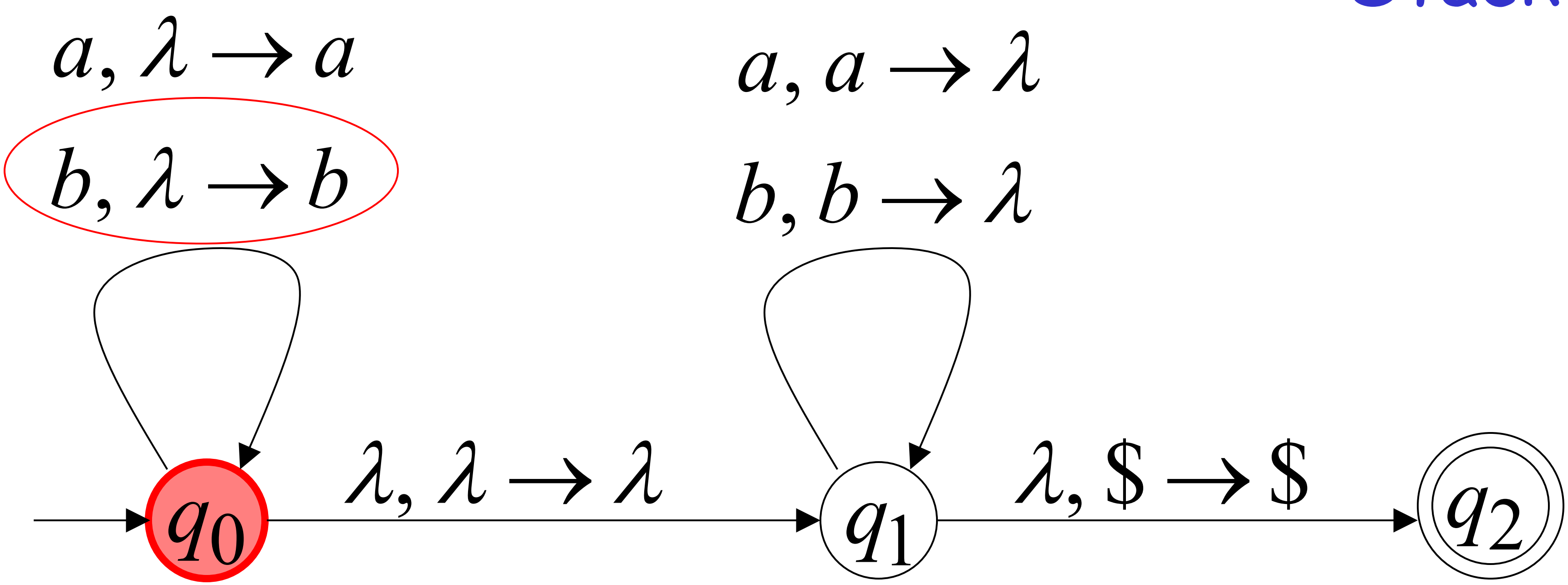
Time 3



Time 4


Input
a
b
b
b


b
b
b
a
$
Stack


$a, \lambda \rightarrow a$
$b, \lambda \rightarrow b$
$a, a \rightarrow \lambda$
$b, b \rightarrow \lambda$
$\lambda, \lambda \rightarrow \lambda$
$\lambda, \$ \rightarrow \$$
q_0
q_1
q_2

Time 5

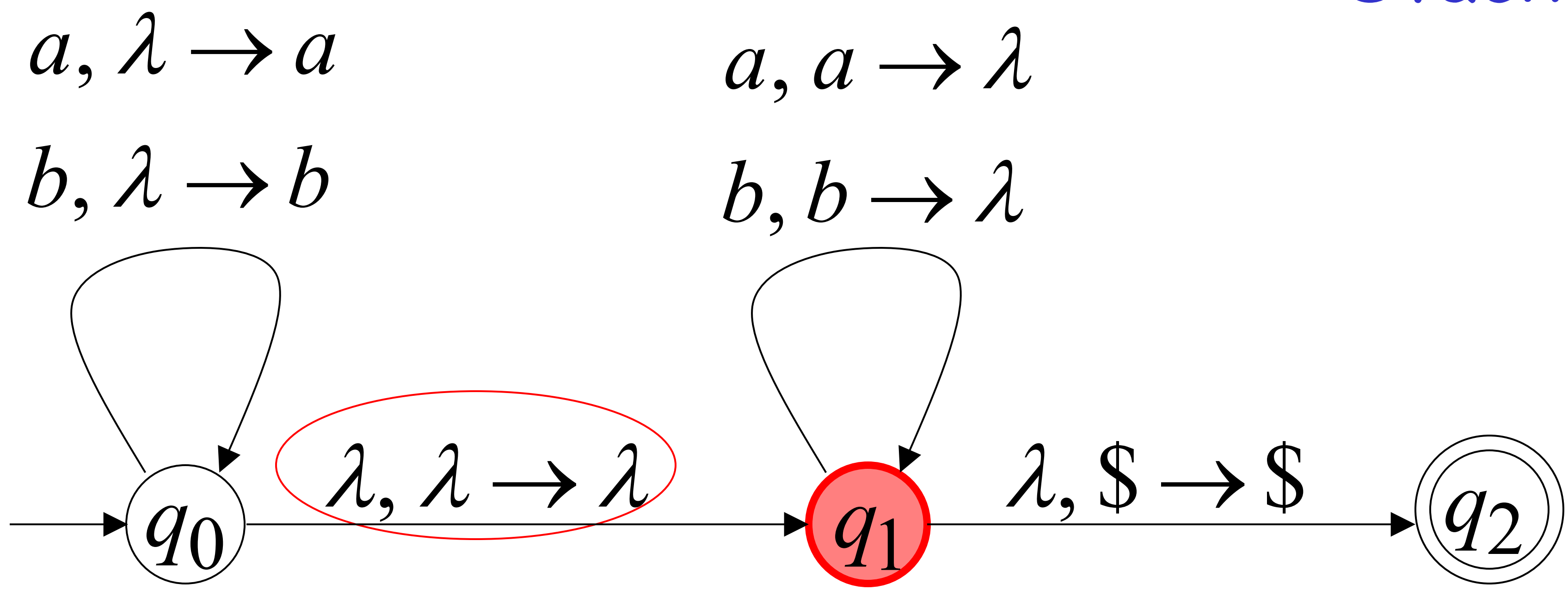
Input

a	b	b	b

No accept state is reached

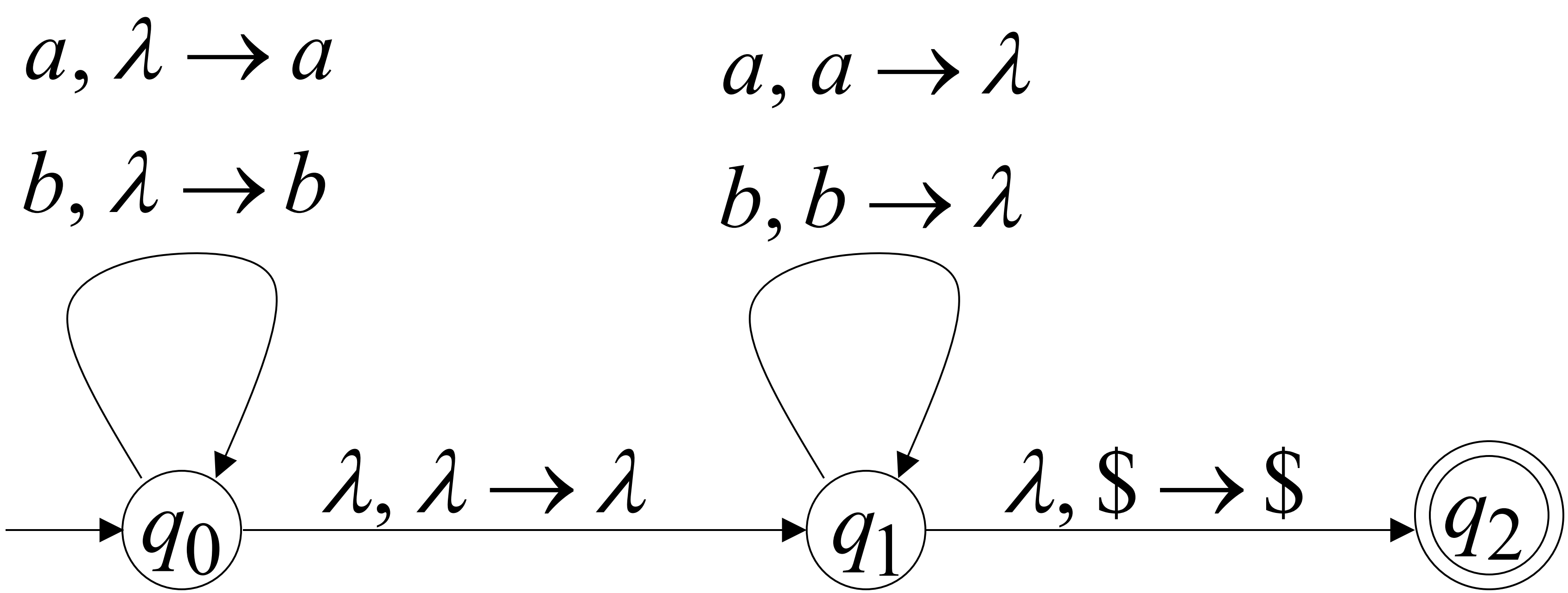
b
b
b
a
$\$$

Stack

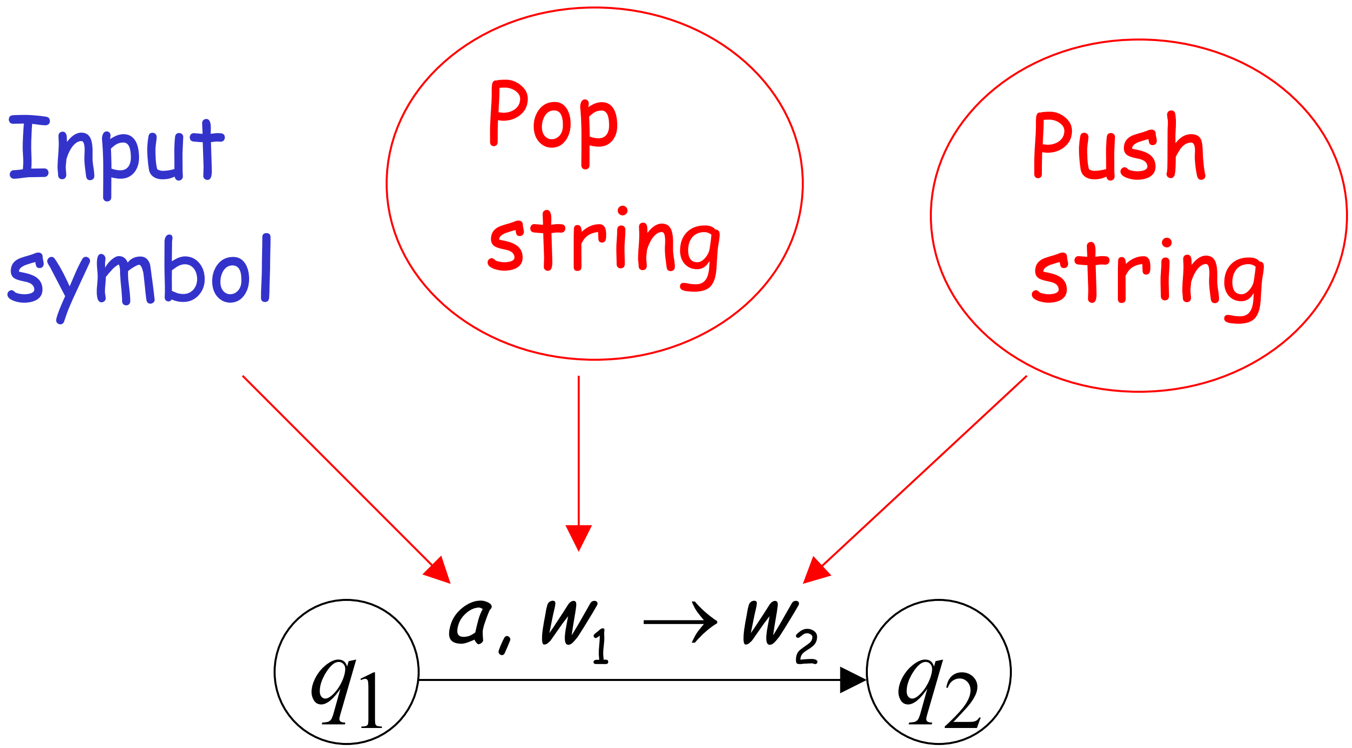


There is no computation that accepts string $abbb$

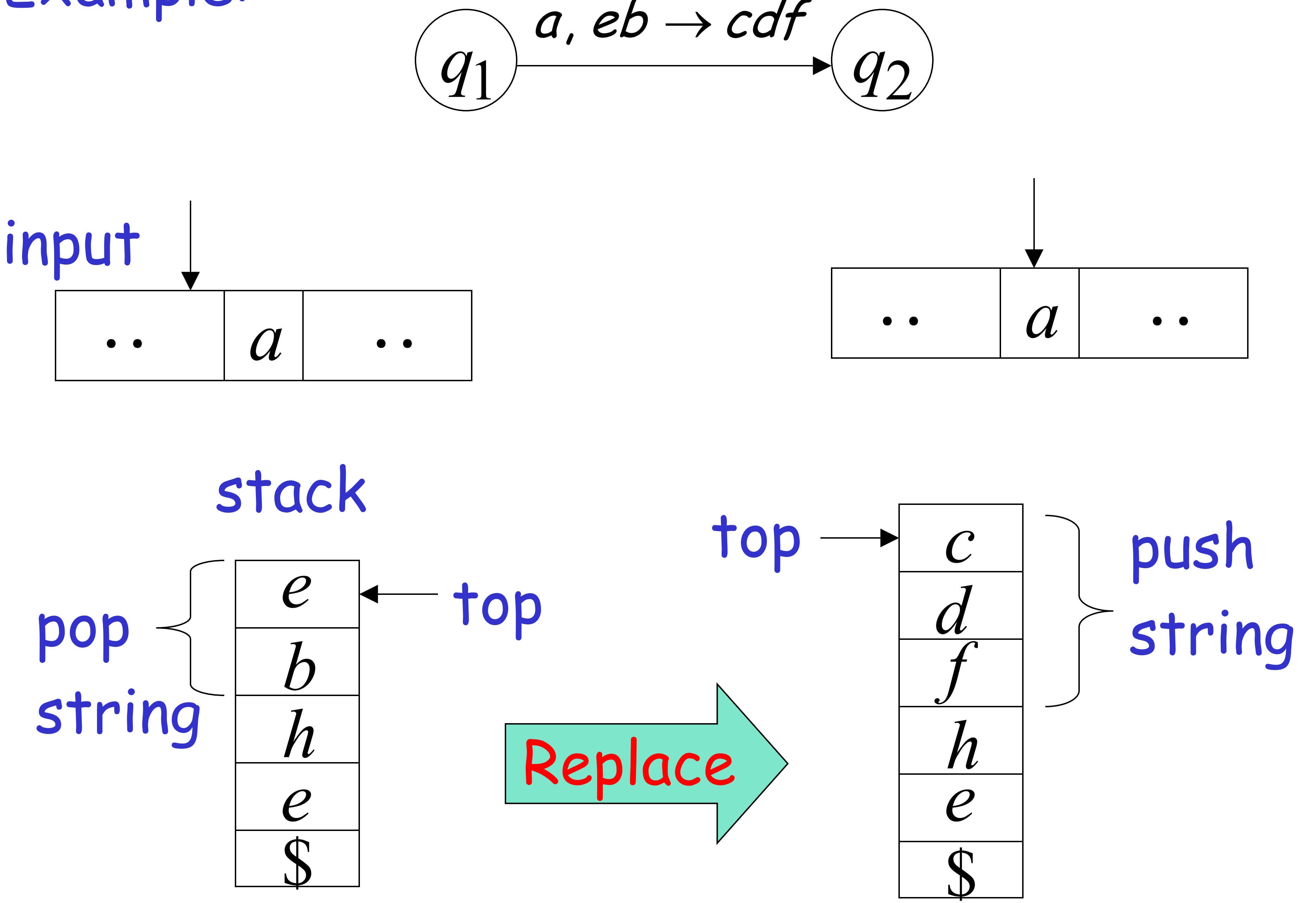
$$abbb \notin L(M)$$



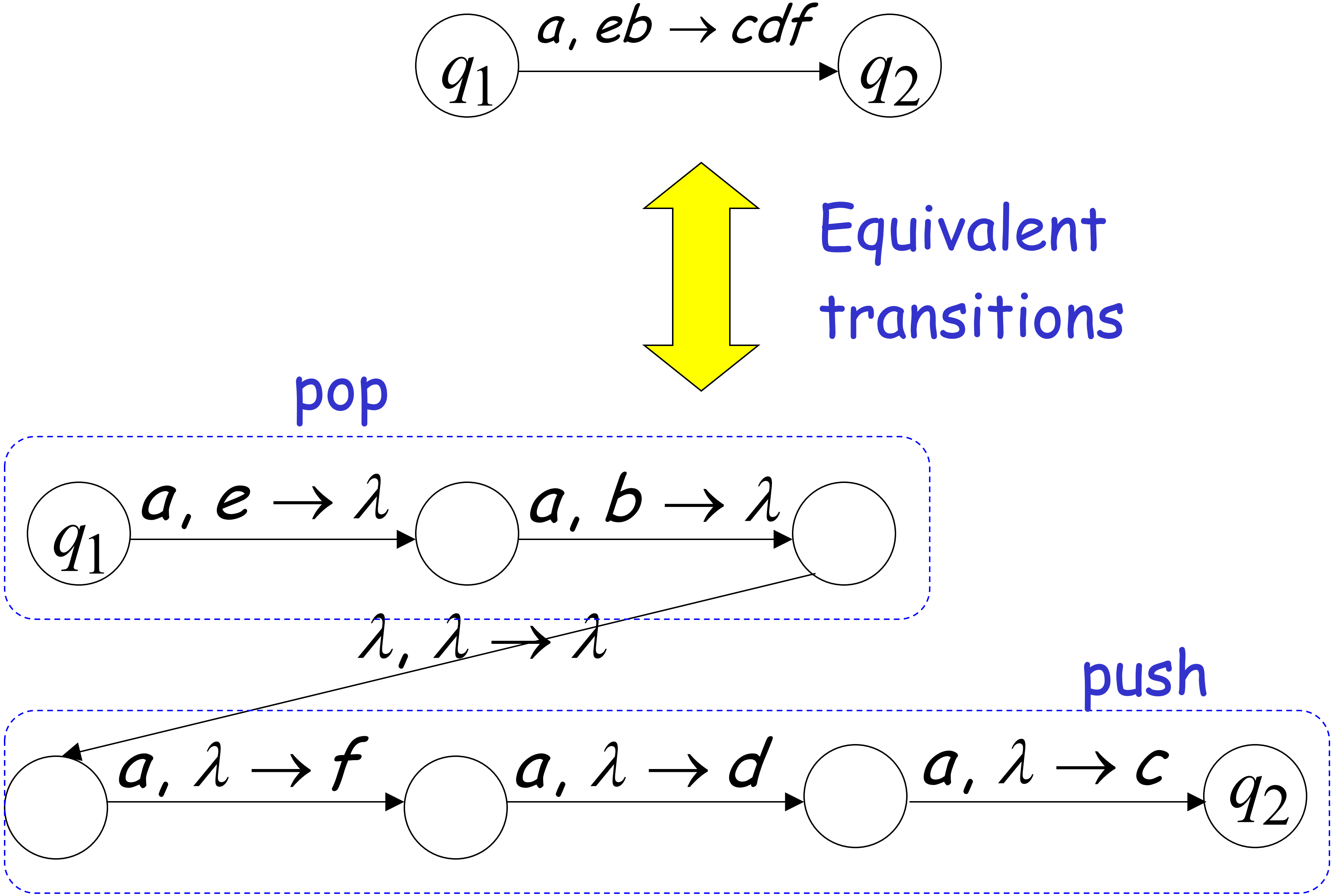
Pushing & Popping Strings



Example:



$a, eb \to cdf$
q_1
q_2
input
stack
pop
string
top
e
b
h
e
$
Replace
top
push
string
c
d
f
h
e
$



q_1
$a, eb \to cdf$
q_2
Equivalent transitions
pop
q_1
$a, e \to \lambda$
$a, b \to \lambda$
$\lambda, \lambda \to \lambda$
push
$a, \lambda \to f$
$a, \lambda \to d$
$a, \lambda \to c$
q_2

Instantaneous Description

$$(q, \; u, \; s)$$

Current state

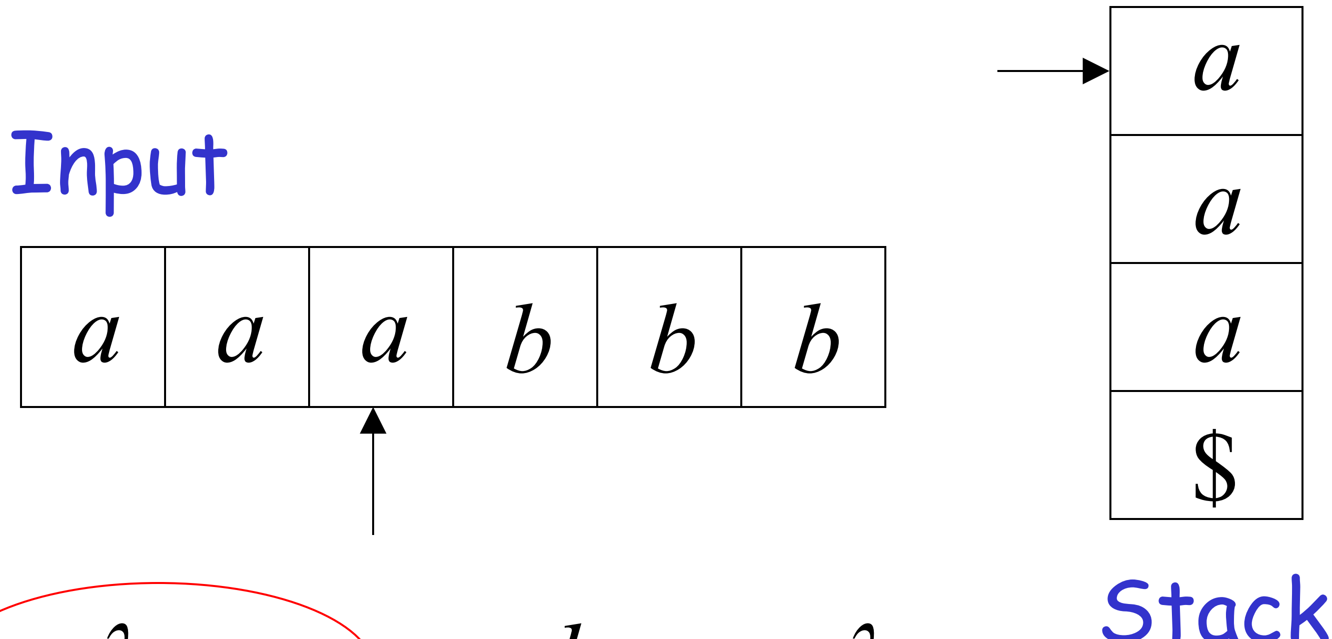
Remaining input

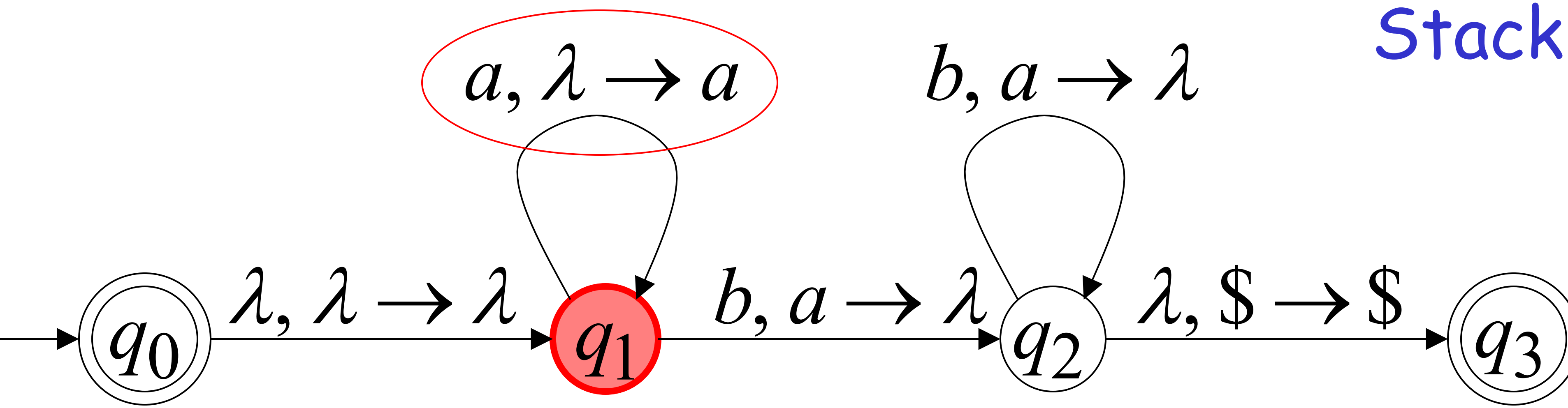
Current stack contents

Example: **Instantaneous Description**

$$(q_1, bbb, aaa\$)$$

Time 4:

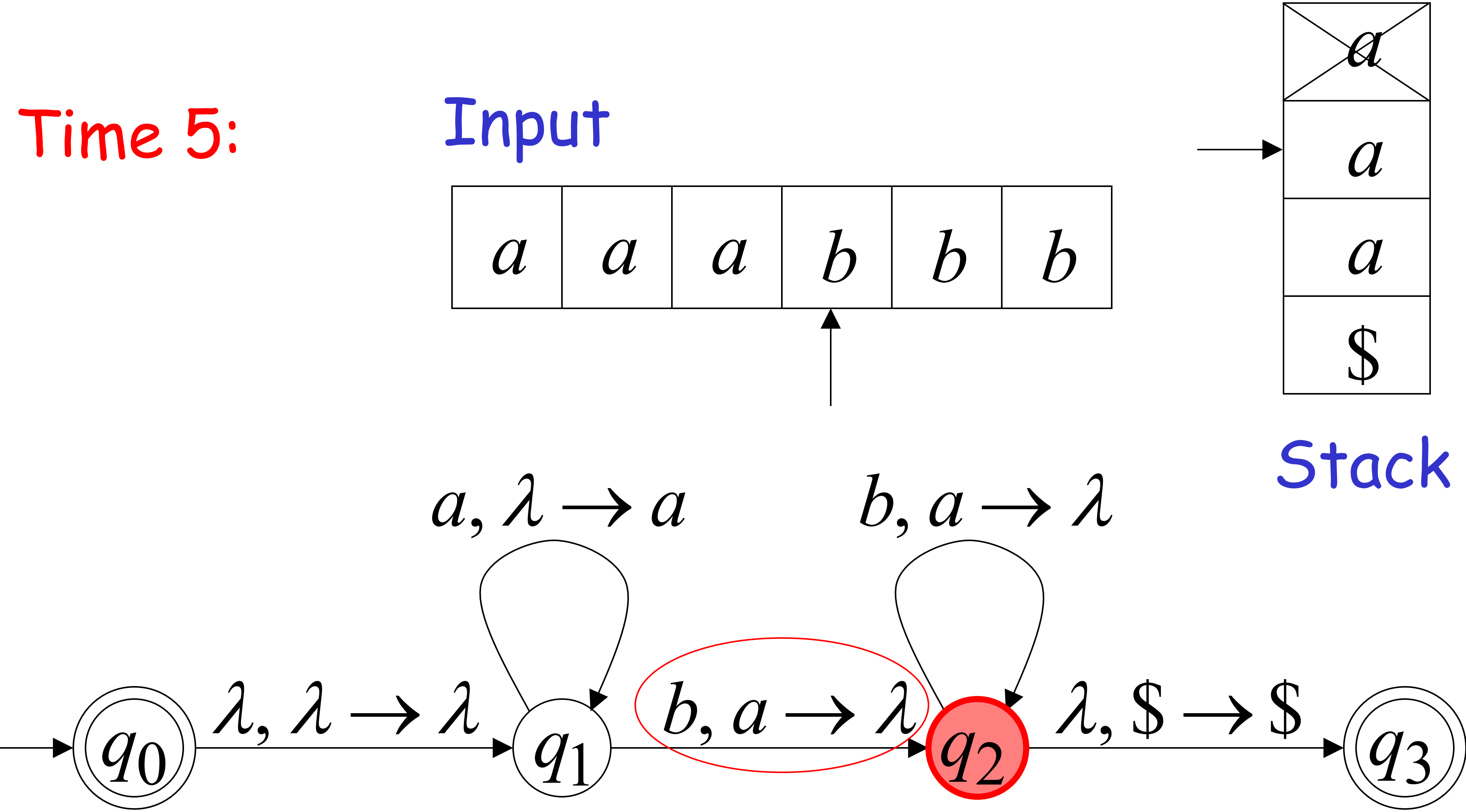




Example: **Instantaneous Description**

$$(q_2, bb, aa\$)$$

Time 5:



We write:

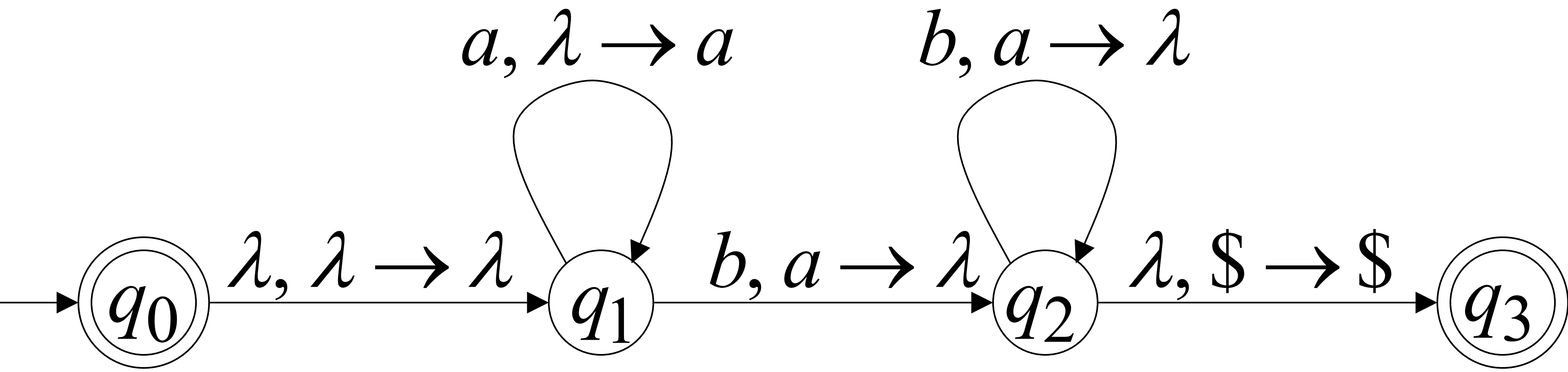
$$(q_1, bbb, aaa\$) \succ (q_2, bb, aa\$)$$

Time 4

Time 5

A computation:

$$(q_0, aaabbb, \$) \succ (q_1, aaabbb, \$) \succ$$
$$(q_1, aabbb, a\$) \succ (q_1, abbb, aa\$) \succ (q_1, bbb, aaa\$) \succ$$
$$(q_2, bb, aa\$) \succ (q_2, b, a\$) \succ (q_2, \lambda, \$) \succ (q_3, \lambda, \$)$$



$$(q_0, aaabbb, \$) \succ (q_1, aaabbb, \$) \succ$$
$$(q_1, aabbb, a\$) \succ (q_1, abbb, aa\$) \succ (q_1, bbb, aaa\$) \succ$$
$$(q_2, bb, aa\$) \succ (q_2, b, a\$) \succ (q_2, \lambda, \$) \succ (q_3, \lambda, \$)$$

For convenience we write:

$$(q_0, aaabbb, \$) \overset{*}{\succ} (q_3, \lambda, \$)$$

Language of PDA

Language $L(M)$ accepted by PDA M:

$$L(M) = \{w : \ (q_0, w, z) \overset{*}{\succ} (q_f, \lambda, s)\}$$

Initial state

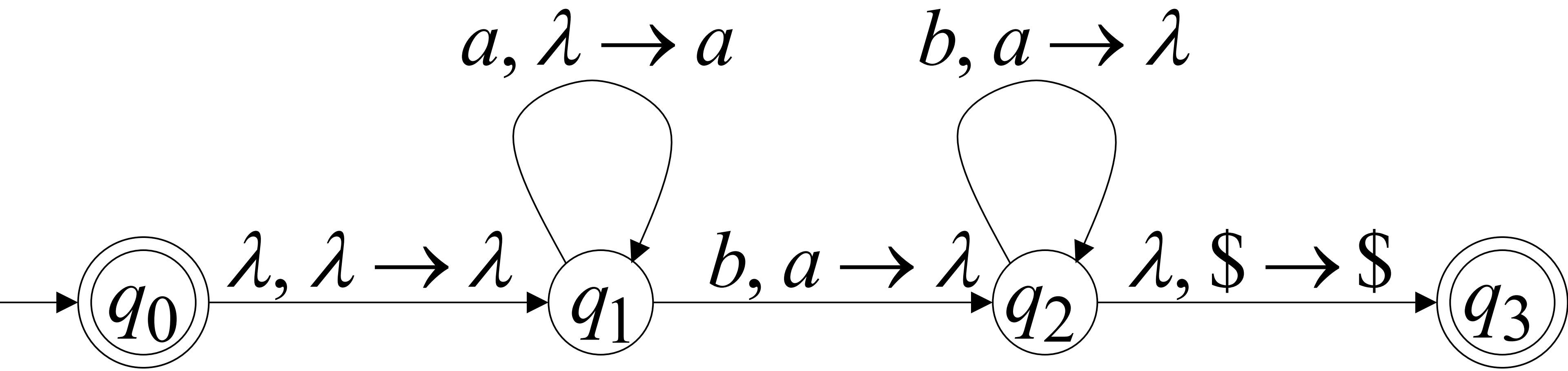
Accept state

Example:

$$(q_0, aaabbb, \$) \overset{*}{\succ} (q_3, \lambda, \$)$$

$$aaabbb \in L(M)$$

PDA M:

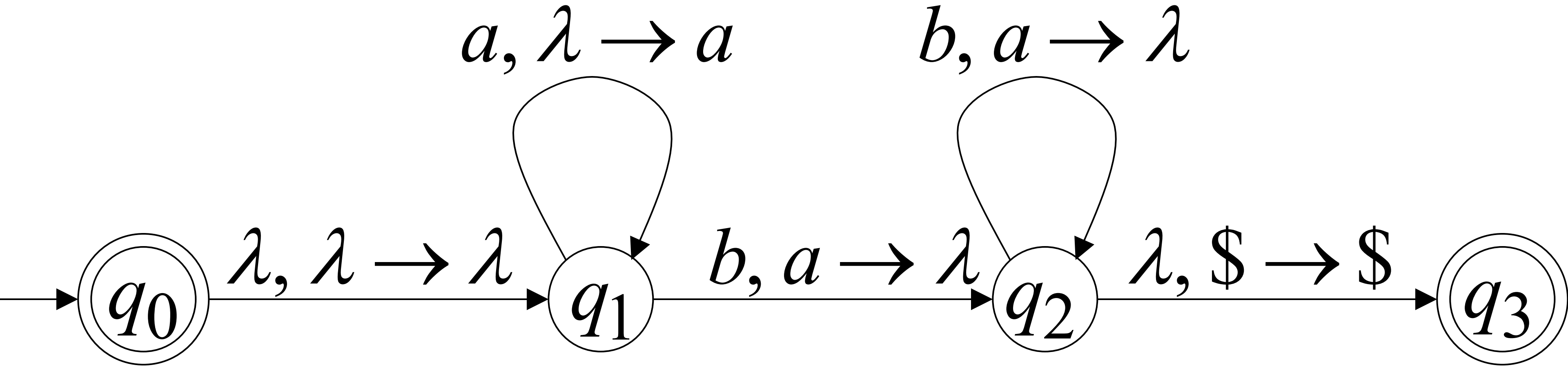


$$(q_0, a^n b^n, \$) \overset{*}{\succ} (q_3, \lambda, \$)$$

$$\Downarrow$$

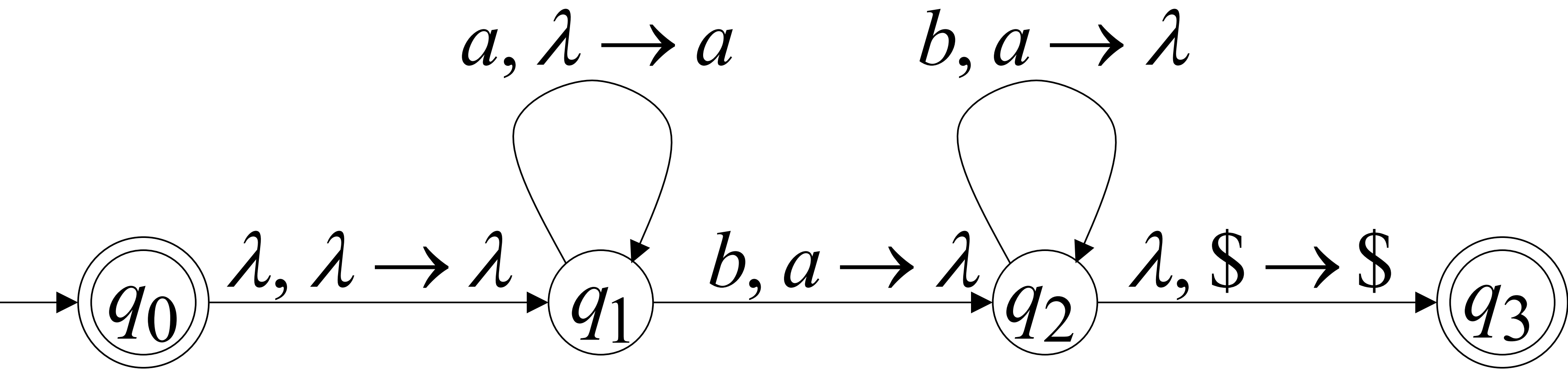
$$a^n b^n \in L(M)$$

PDA M :



Therefore: $L(M) = \{a^n b^n : n \geq 0\}$

PDA M :



PDAs Accept Context-Free Languages

Theorem:

$$\left\{\begin{array}{c}\text{Context-Free}\\\text{Languages}\\\text{(Grammars)}\end{array}\right\} = \left\{\begin{array}{c}\text{Languages}\\\text{Accepted by}\\\text{PDAs}\end{array}\right\}$$

Convert

Context-Free Grammars
to
PDAs

Take an arbitrary context-free grammar G

We will convert G to a PDA M such that:

$$L(G) = L(M)$$

Conversion Procedure:

For each production in G

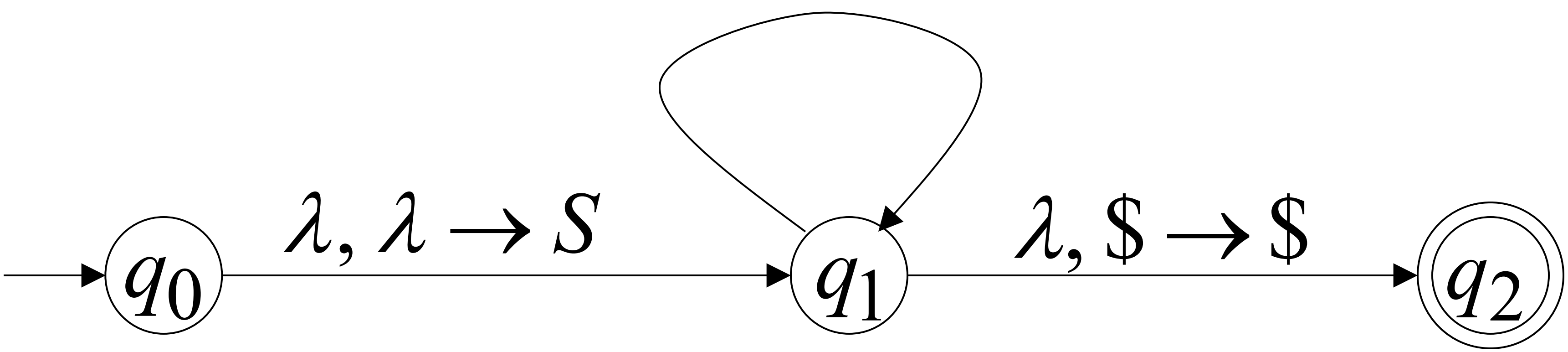
$$A \to w$$

For each terminal in G

$$a$$

Add transitions

$$\lambda, A \to w \qquad a, a \to \lambda$$



Example

Grammar

$S \to aSTb$

$S \to b$

$T \to Ta$

$T \to \lambda$

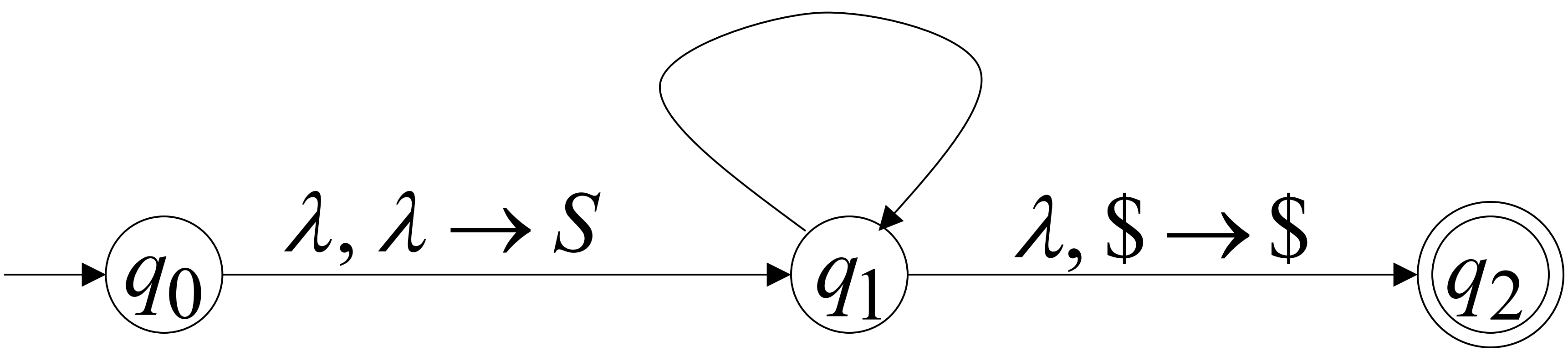
PDA

$\lambda, S \to aSTb$

$\lambda, S \to b$

$\lambda, T \to Ta$ $\qquad a, a \to \lambda$

$\lambda, T \to \lambda$ $\qquad b, b \to \lambda$



Example:

Input

a	b	a	b

Time 0

$\lambda, S \rightarrow aSTb$

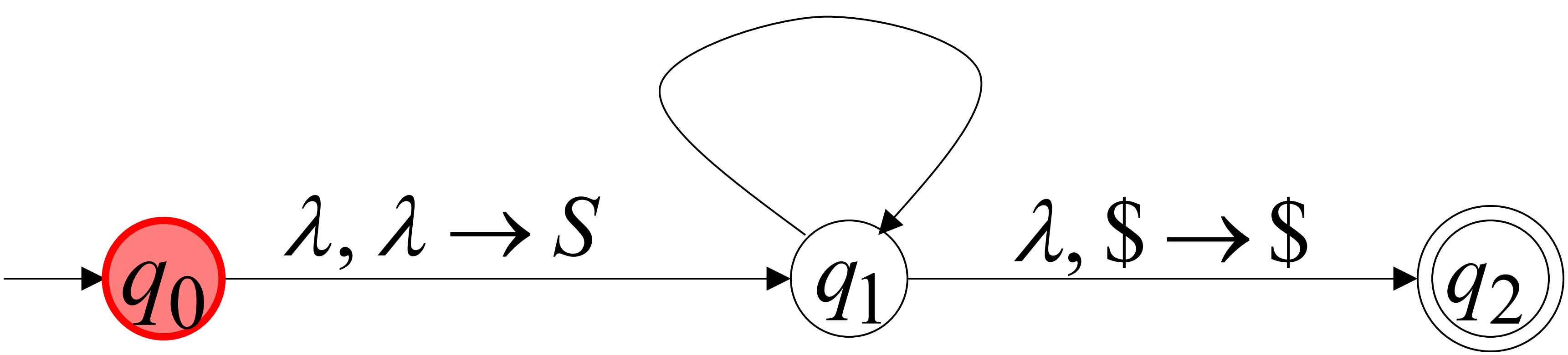
$\lambda, S \rightarrow b$

$\lambda, T \rightarrow Ta$ $\quad a, a \rightarrow \lambda$

$\lambda, T \rightarrow \lambda$ $\quad b, b \rightarrow \lambda$

$\$$

Stack



Derivation: S

Input

a	b	a	b

Time 1

$$\lambda, S \rightarrow aSTb$$

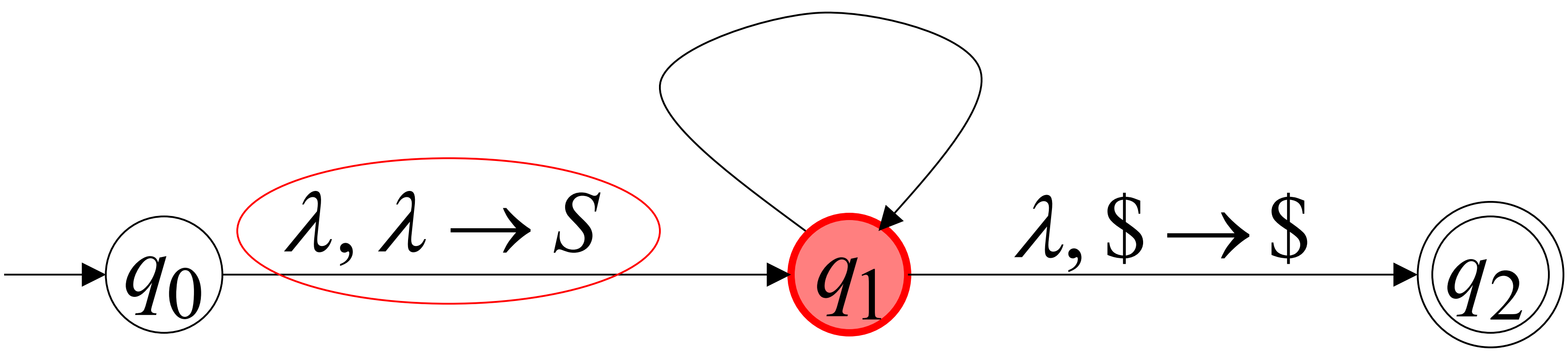
$$\lambda, S \rightarrow b$$

$$\lambda, T \rightarrow Ta \qquad a, a \rightarrow \lambda$$

$$\lambda, T \rightarrow \lambda \qquad b, b \rightarrow \lambda$$

S
$\$$

Stack



Derivation: $S \Rightarrow aSTb$

Input

a	b	a	b

Time 2

$\lambda, S \rightarrow aSTb$

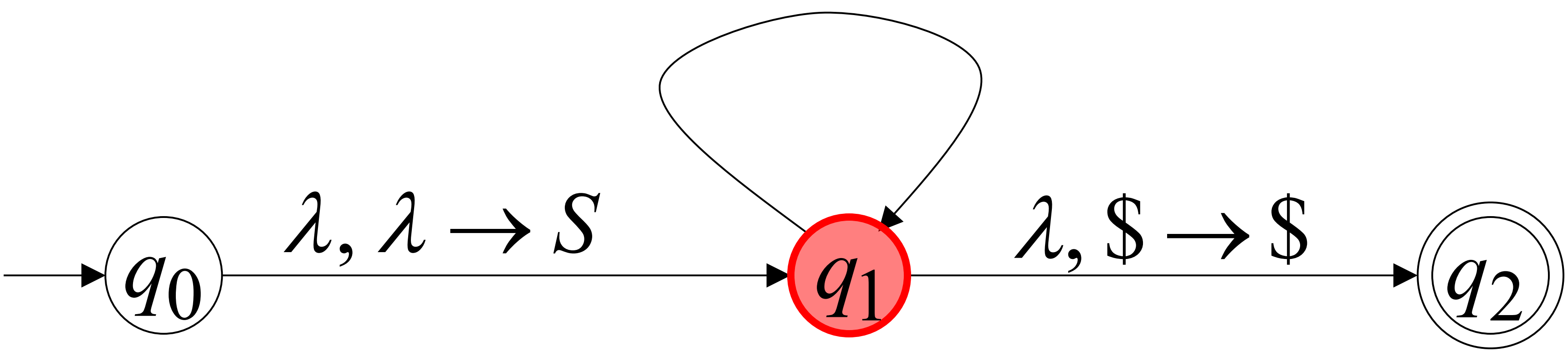
$\lambda, S \rightarrow b$

$\lambda, T \rightarrow Ta$ $\quad a, a \rightarrow \lambda$

$\lambda, T \rightarrow \lambda$ $\quad b, b \rightarrow \lambda$

Stack

a
S
T
b
$\$$



Derivation: $S \Rightarrow aSTb$

Input

a	b	a	b

Time 3

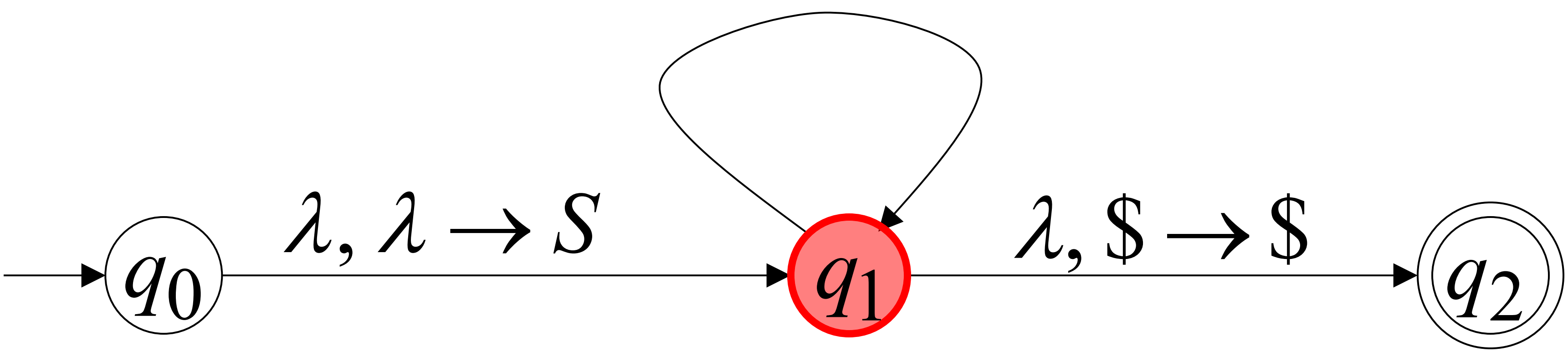
$\lambda, S \rightarrow aSTb$

$\lambda, S \rightarrow b$

$\lambda, T \rightarrow Ta$ $\quad a, a \rightarrow \lambda$

$\lambda, T \rightarrow \lambda$ $\quad b, b \rightarrow \lambda$

Stack
$\cancel{a}$
S
T
b
$\$$



Derivation: $S \Rightarrow aSTb \Rightarrow abTb$

Input

a	b	a	b

Time 4

$\lambda, S \rightarrow aSTb$

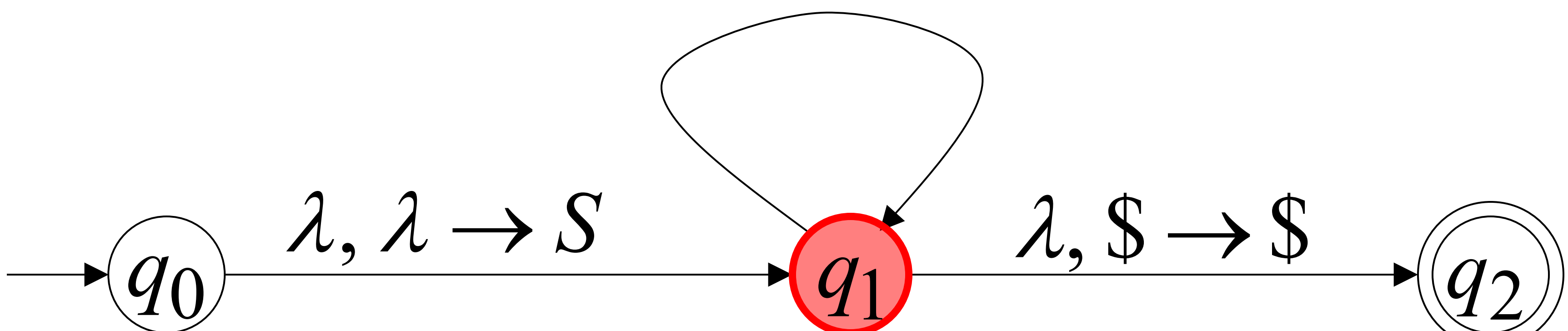
$\lambda, S \rightarrow b$

$\lambda, T \rightarrow Ta$ $\qquad a, a \rightarrow \lambda$

$\lambda, T \rightarrow \lambda$ $\qquad b, b \rightarrow \lambda$

b
T
b
$\$$

Stack



Derivation: $S \Rightarrow aSTb \Rightarrow abTb$

Input

a	b	a	b

Time 5

$\lambda, S \rightarrow aSTb$

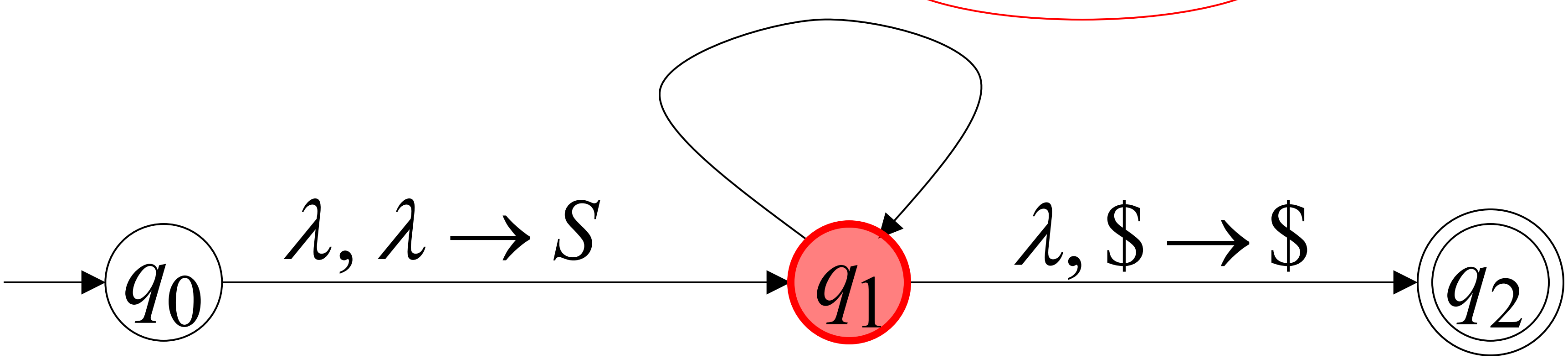
$\lambda, S \rightarrow b$

$\lambda, T \rightarrow Ta$ $\quad a, a \rightarrow \lambda$

$\lambda, T \rightarrow \lambda$ $\quad b, b \rightarrow \lambda$

Stack
$\cancel{b}$
T
b
$\$$

Stack



Derivation: $S \Rightarrow aSTb \Rightarrow abTb \Rightarrow abTab$

Input

a	b	a	b

Time 6

T
a
b
$\$$

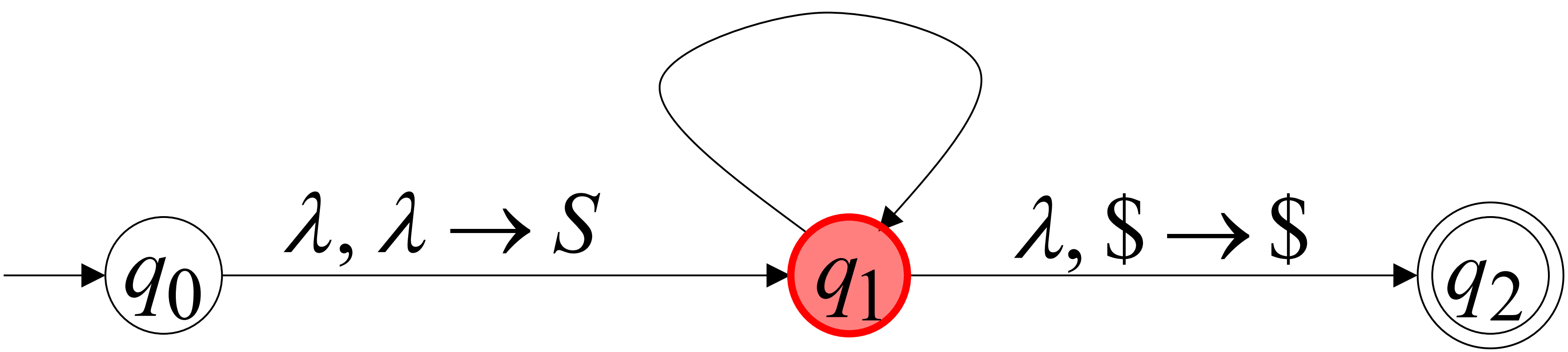
Stack

$\lambda, S \rightarrow aSTb$

$\lambda, S \rightarrow b$

$\lambda, T \rightarrow Ta$ $\quad$ $a, a \rightarrow \lambda$

$\lambda, T \rightarrow \lambda$ $\quad$ $b, b \rightarrow \lambda$



Derivation: $S \Rightarrow aSTb \Rightarrow abTb \Rightarrow abTab \Rightarrow abab$

Input

a	b	a	b

Time 7

$\lambda, S \rightarrow aSTb$

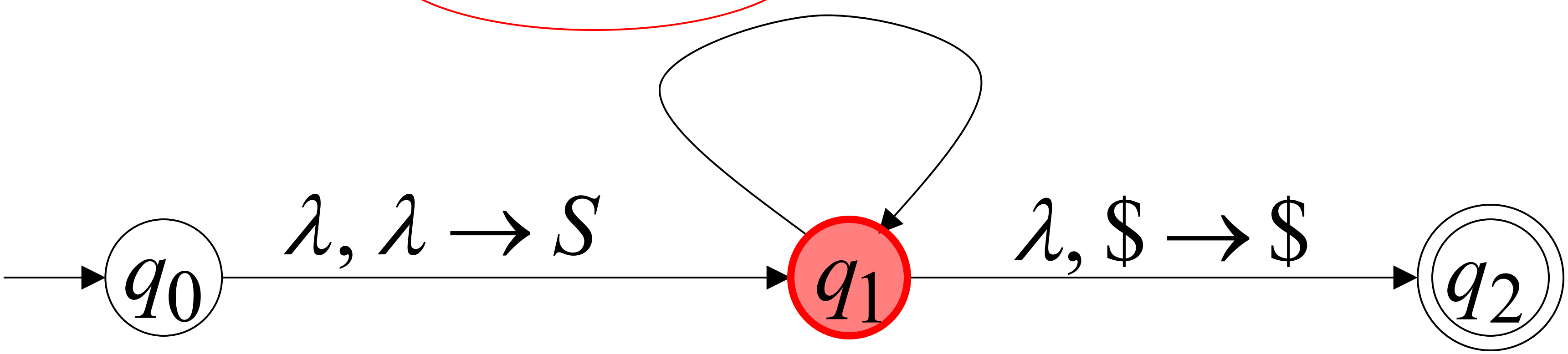
$\lambda, S \rightarrow b$

$\lambda, T \rightarrow Ta$ $\quad a, a \rightarrow \lambda$

$\lambda, T \rightarrow \lambda$ $\quad b, b \rightarrow \lambda$

$\cancel{T}$
a
b
$\$$

Stack



Derivation: $S \Rightarrow aSTb \Rightarrow abTb \Rightarrow abTab \Rightarrow abab$

Input

a	b	a	b

Time 8

$\lambda, S \to aSTb$

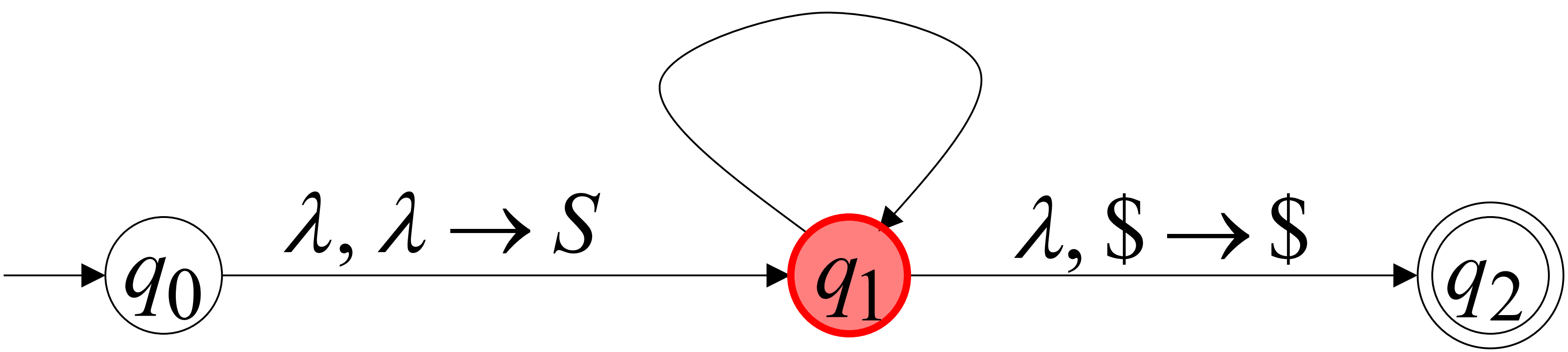
$\lambda, S \to b$

$\lambda, T \to Ta$ $\qquad a, a \to \lambda$

$\lambda, T \to \lambda$ $\qquad b, b \to \lambda$

$\cancel{a}$
b
$\$$

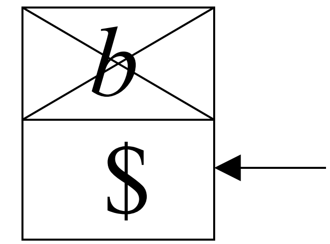
Stack



Derivation: $S \Rightarrow aSTb \Rightarrow abTb \Rightarrow abTab \Rightarrow abab$

Input

a	b	a	b



Stack

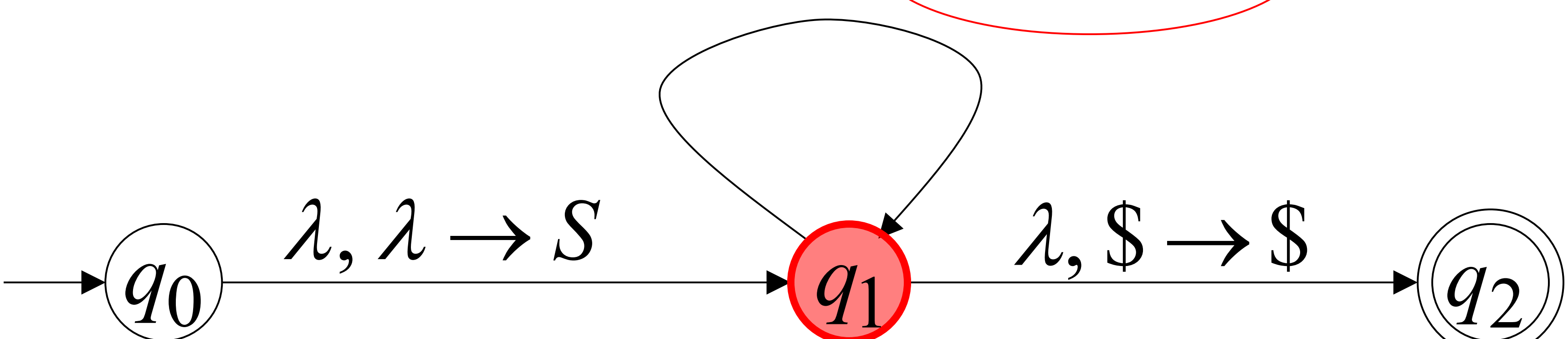
Time 9

$\lambda, S \rightarrow aSTb$

$\lambda, S \rightarrow b$

$\lambda, T \rightarrow Ta$ $\quad a, a \rightarrow \lambda$

$\lambda, T \rightarrow \lambda$ $\quad b, b \rightarrow \lambda$



Derivation: $S \Rightarrow aSTb \Rightarrow abTb \Rightarrow abTab \Rightarrow abab$

Input

a	b	a	b

Time 10

$\$$

Stack

$\lambda, S \to aSTb$

$\lambda, S \to b$

$\lambda, T \to Ta$ $\qquad a, a \to \lambda$

$\lambda, T \to \lambda$ $\qquad b, b \to \lambda$

accept