Context-Free Grammars and Langugages

Formal Languages and Abstract Machines

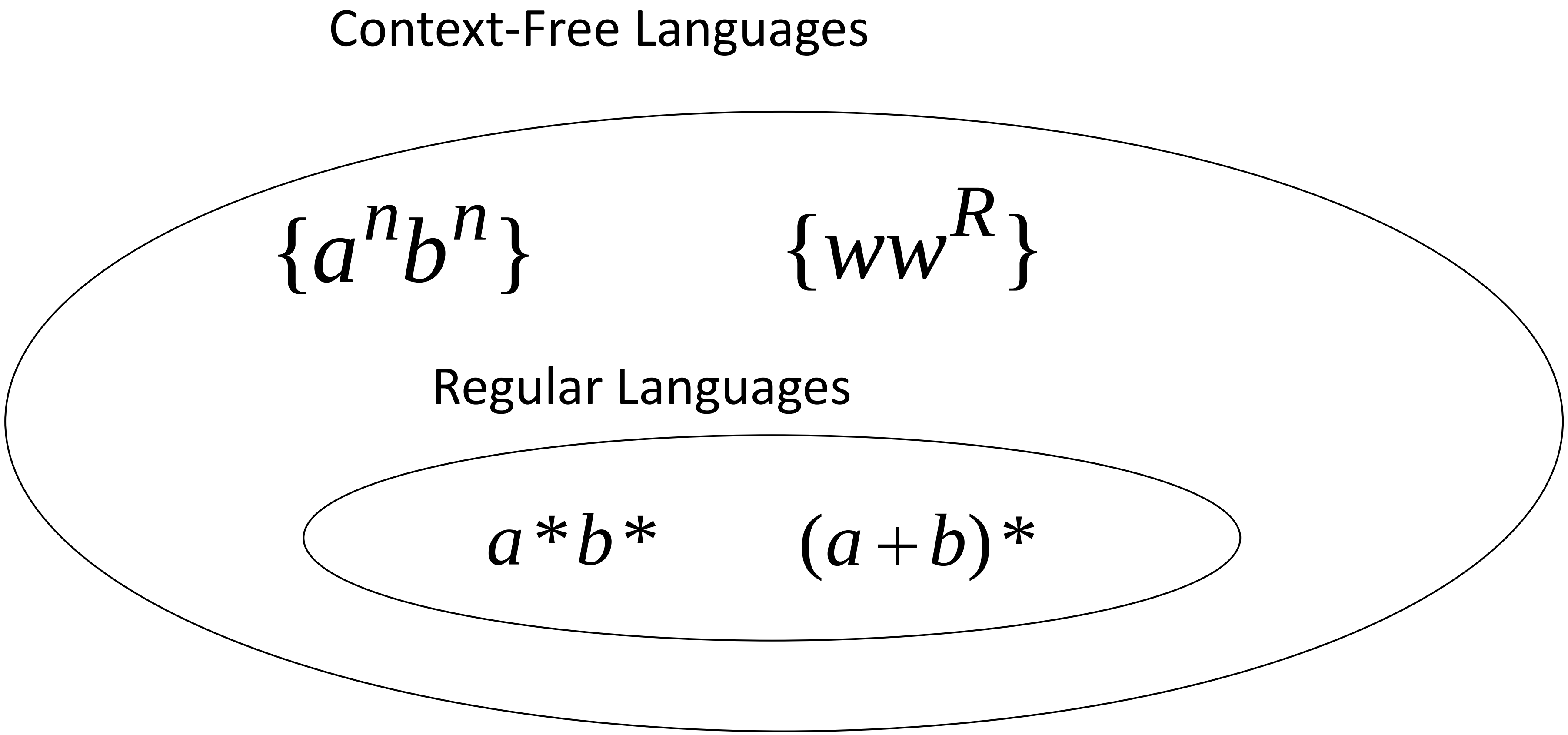
Week 07

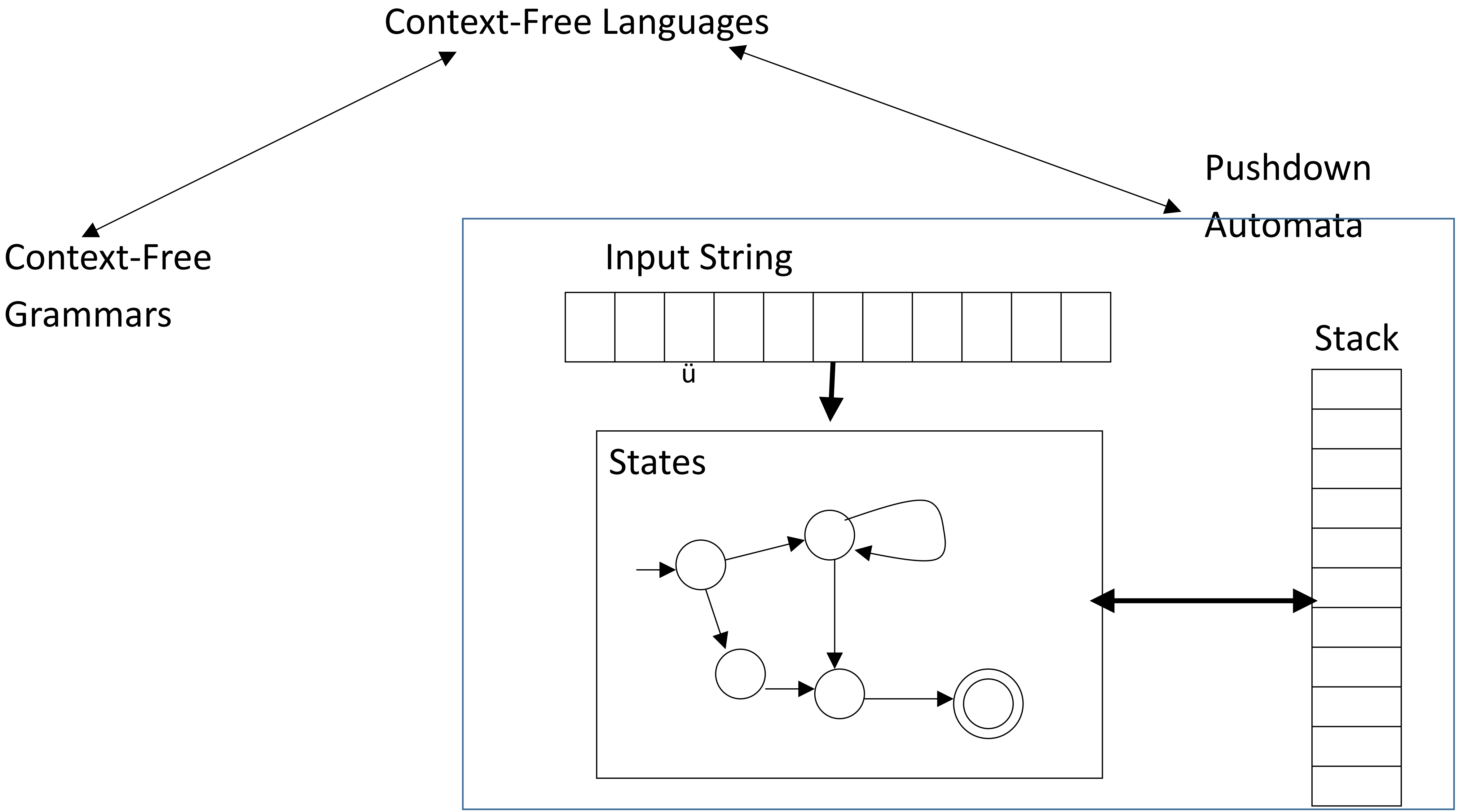
Baris E. Suzek, PhD

Outline

- Last week
- Pushdown automata
- Properties of Context-free Languages

Context-Free and Regular Languages




Context-Free Languages
Context-Free
Grammars
Pushdown
Automata
Input String
ü
Stack
States

Phase 2 $\quad S \to SS \mid aSb \mid bSa \mid \lambda$

$S \Rightarrow SS \Rightarrow SSS$ $\qquad aabb$

$S \Rightarrow SS \Rightarrow aSbS$

Phase 1 $\qquad \cancel{S \Rightarrow SS \Rightarrow bSaS}$

$S \Rightarrow SS$ $\qquad S \Rightarrow SS \Rightarrow S$

$S \Rightarrow aSb$ $\qquad S \Rightarrow aSb \Rightarrow aSSb$

$S \Rightarrow aSb \Rightarrow aaSbb$

$\cancel{S \Rightarrow aSb \Rightarrow abSab}$

$\cancel{S \Rightarrow aSb \Rightarrow ab}$

Total time needed for string w:

$$k + k^2 + \cdots + k^{|w|}$$

phase 1 $\qquad$ phase 2 $\qquad$ phase |w|

Pretty bad!!!

Chomsky Normal Form

Each productions has form:

$$A \to BC$$

variable variable

or

$$A \to a$$

terminal

and no useless productions.

Greinbach Normal Form

All productions have form:

$$A \to a\,V_1 V_2 \cdots V_k \qquad k \geq 0$$

symbol variables

Final grammar in Chomsky Normal Form:

Initial grammar

$$S \to ABa$$
$$A \to aab$$
$$B \to Ac$$

$$S \to AV_1$$
$$V_1 \to BT_a$$
$$A \to T_a V_2$$
$$V_2 \to T_a T_b$$
$$B \to AT_c$$
$$T_a \to a$$
$$T_b \to b$$
$$T_c \to c$$

The CYK(Cocke–Younger–Kasami) Membership Algorithm

Input: Grammar G in Chomsky Normal Form

String w

Output: Find if string $w \in L(G)$

$S \to AB$

$A \to BB$

$A \to a$

$B \to AB$

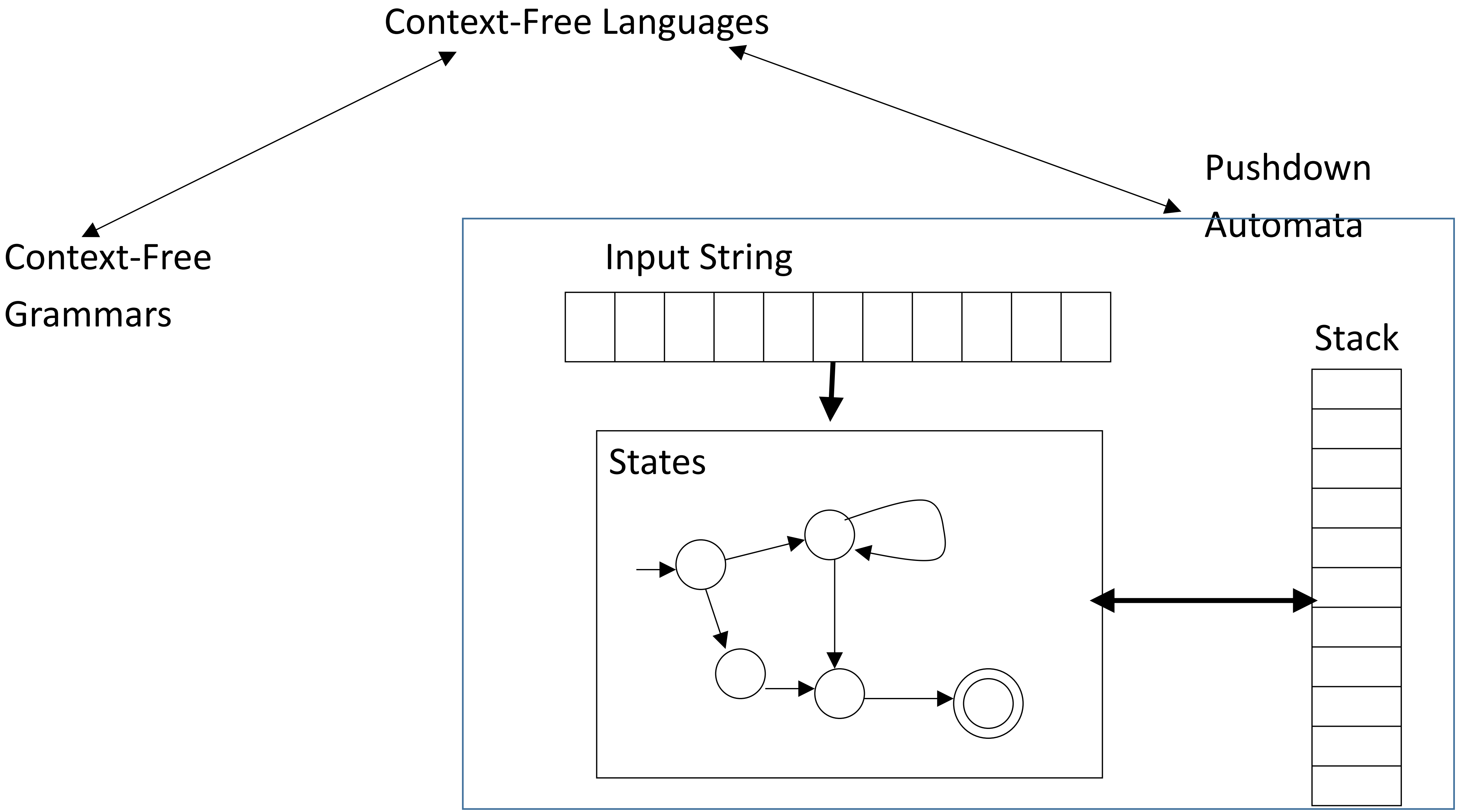
$B \to b$

a A	a A	b B	b B	b B
aa	ab S,B	bb A	bb A	
aab S,B	abb A	bbb S,B		
aabb A	abbb S,B			
aabbb S,B				

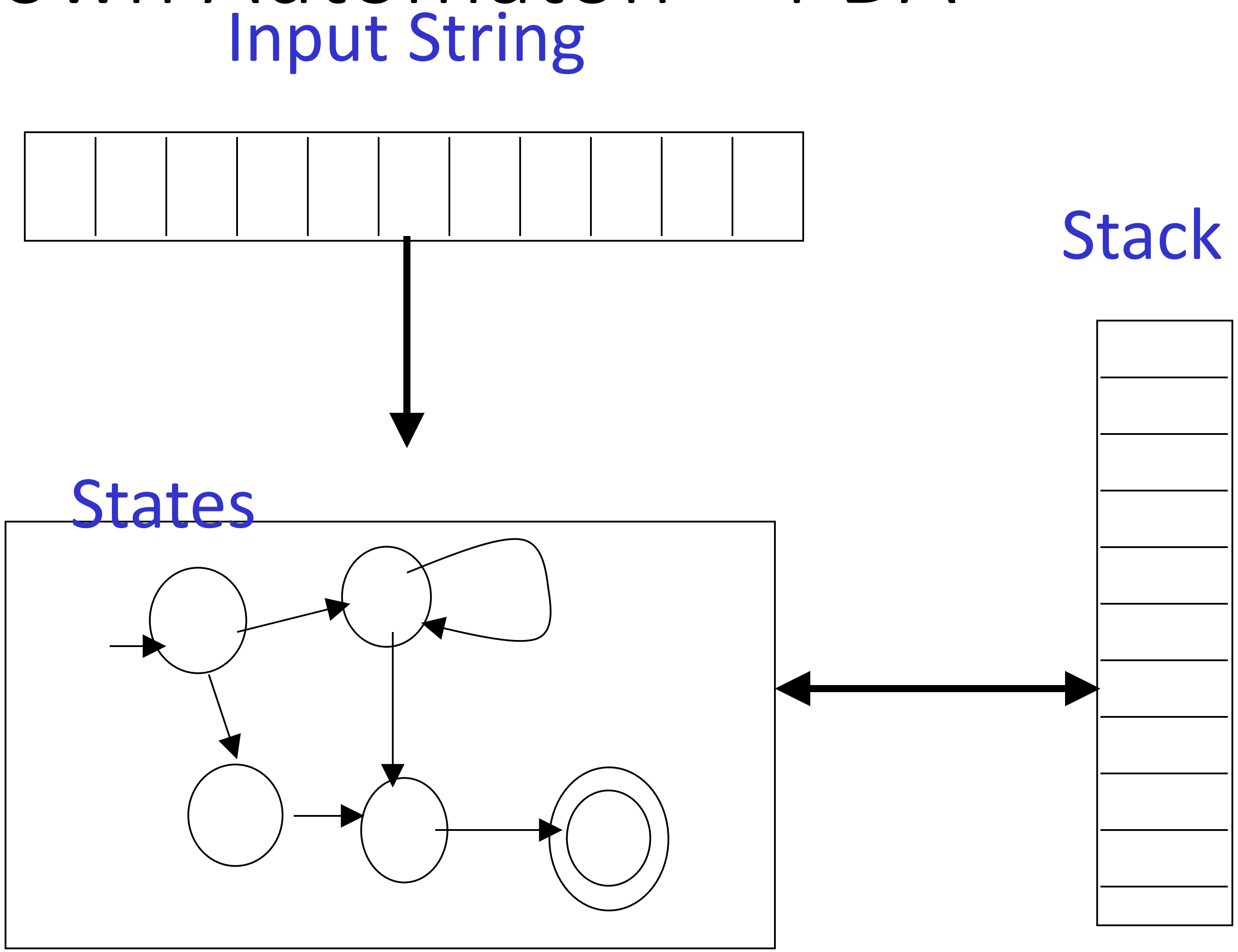
What is the computation complexity?

Outline

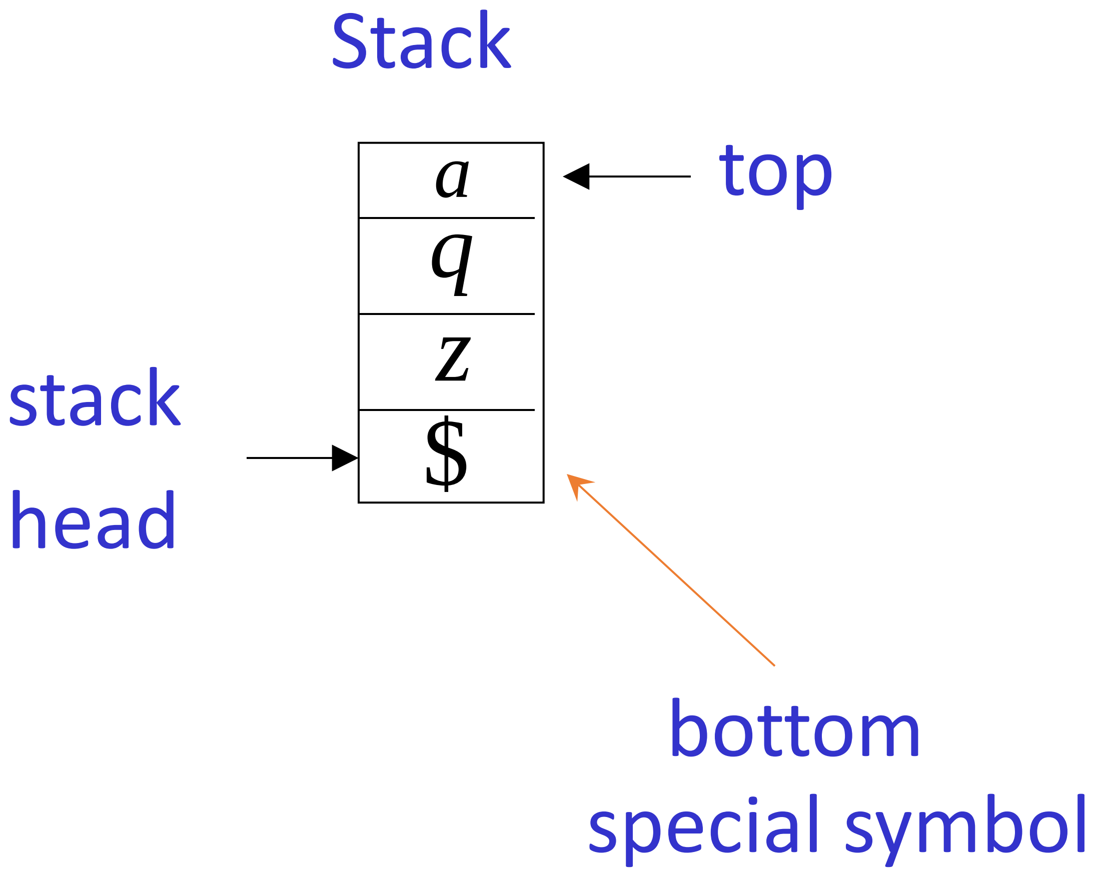
- Last week
- Pushdown automata
- Properties of Context-free Languages


Context-Free Languages
Context-Free Grammars
Pushdown Automata
Input String
States
Stack

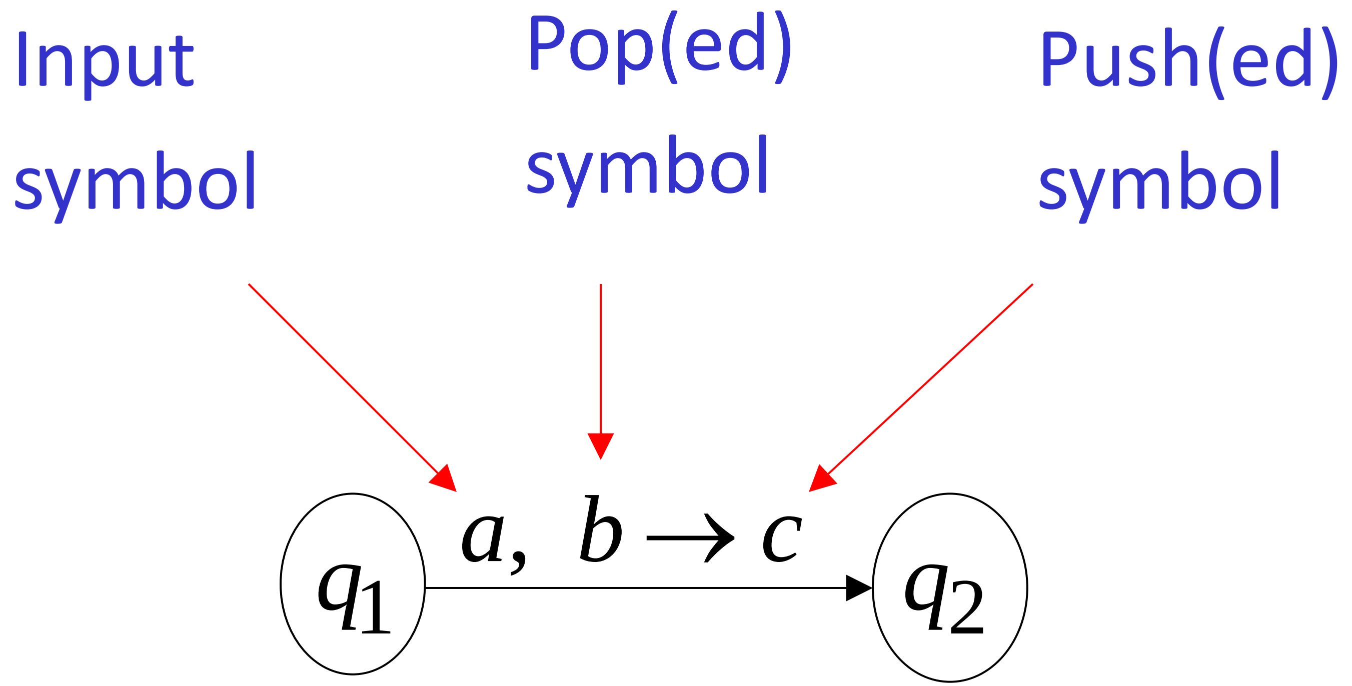
Pushdown Automaton -- PDA

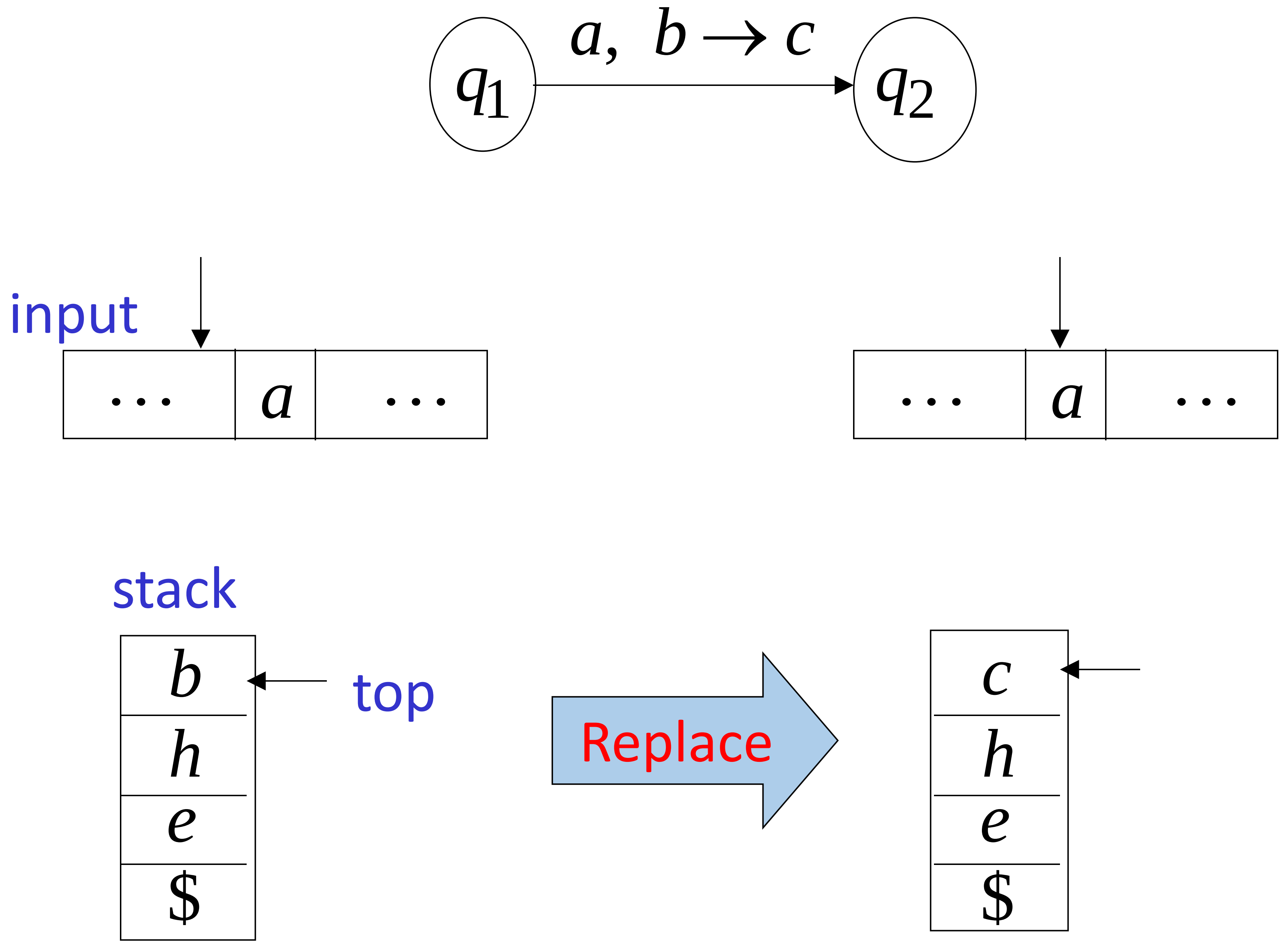


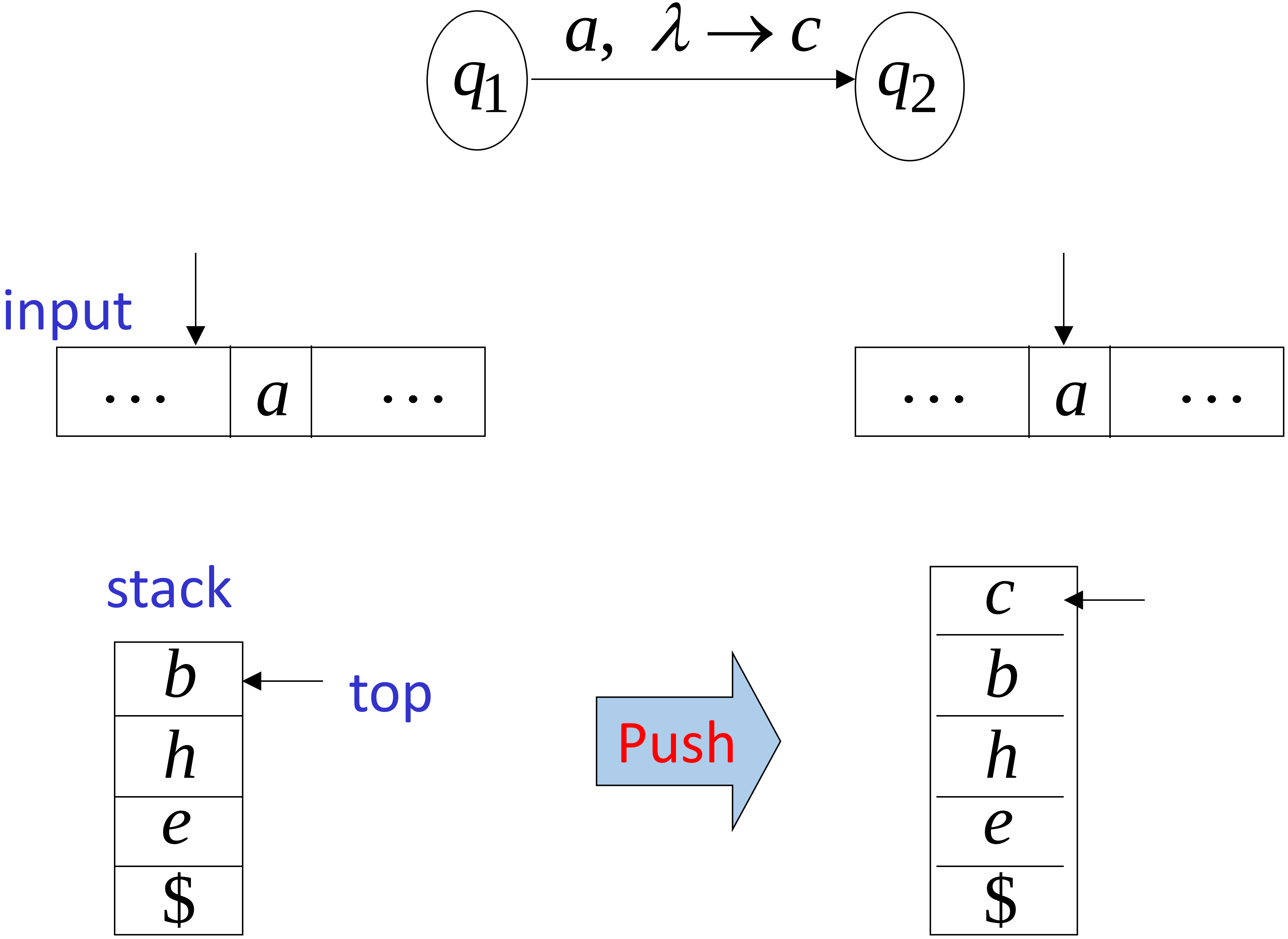
Initial Stack Symbol

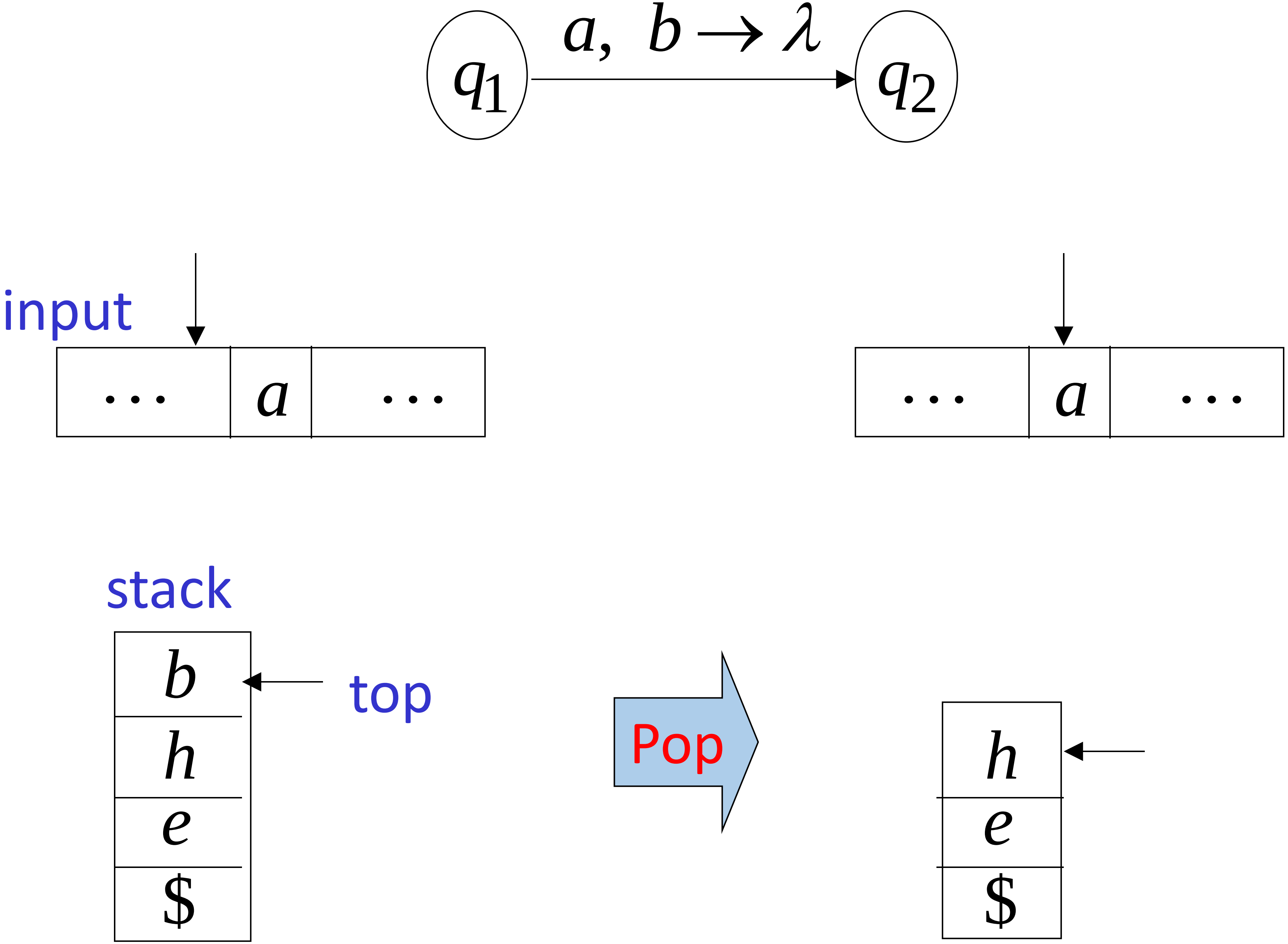


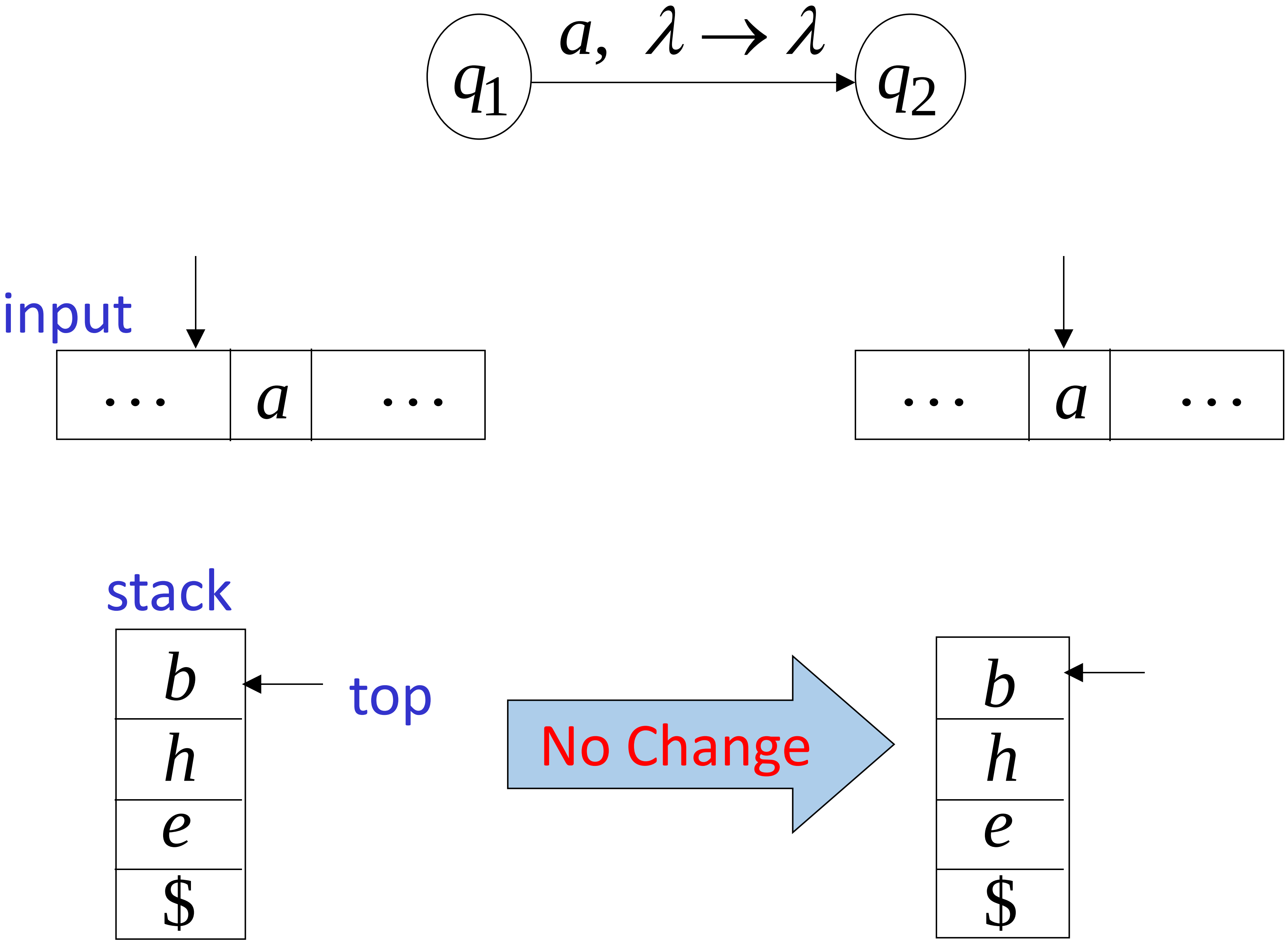
The States



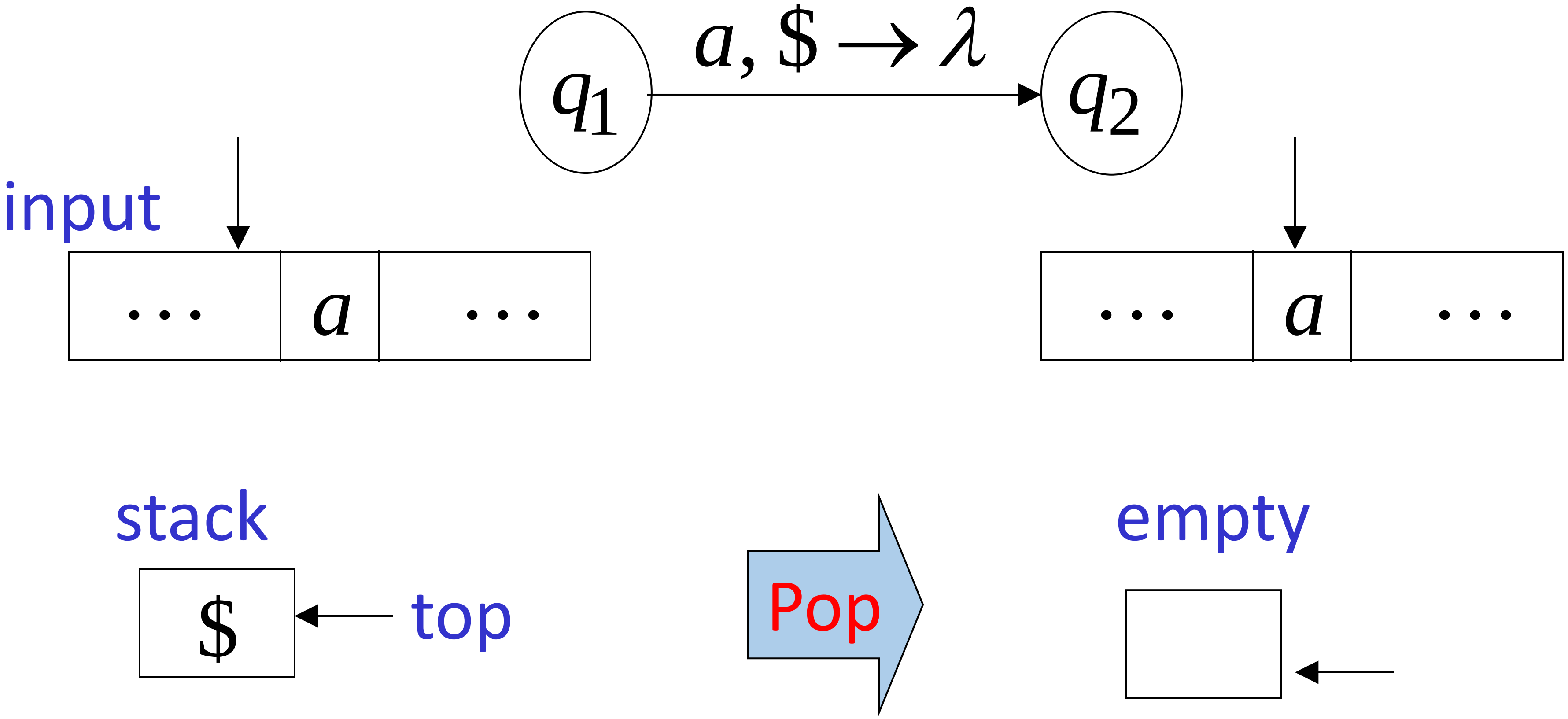

q_1
$a, \; b \rightarrow c$
q_2
input
$\cdots$ a $\cdots$
$\cdots$ a $\cdots$
stack
b
h
e
$\$$
top
Replace
c
h
e
$\$$


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

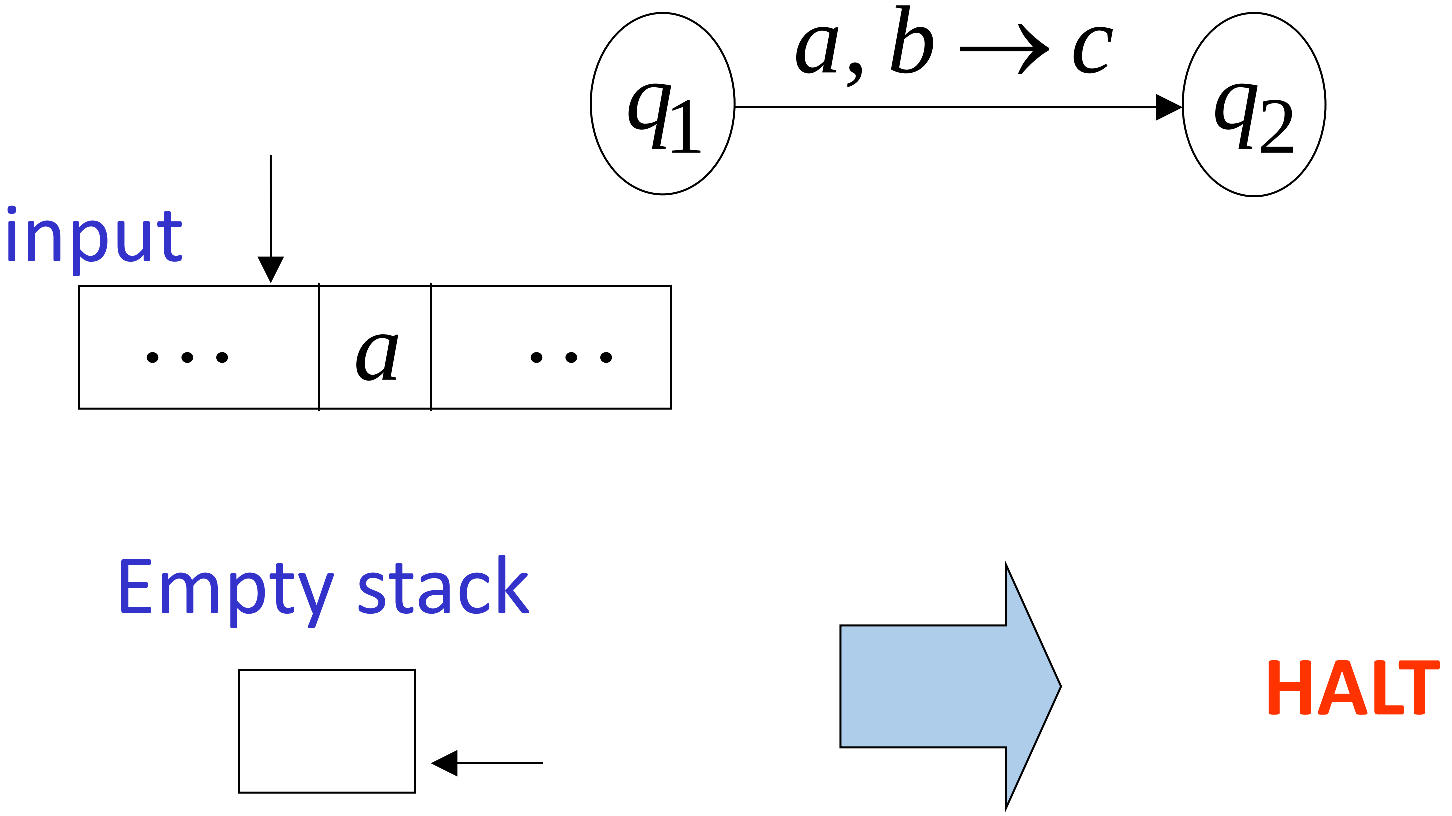

q_1
$a, \; b \rightarrow \lambda$
q_2
input
$\cdots$
a
$\cdots$
$\cdots$
a
$\cdots$
stack
b
h
e
$\$$
top
Pop
h
e
$\$$


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

A Possible Transition



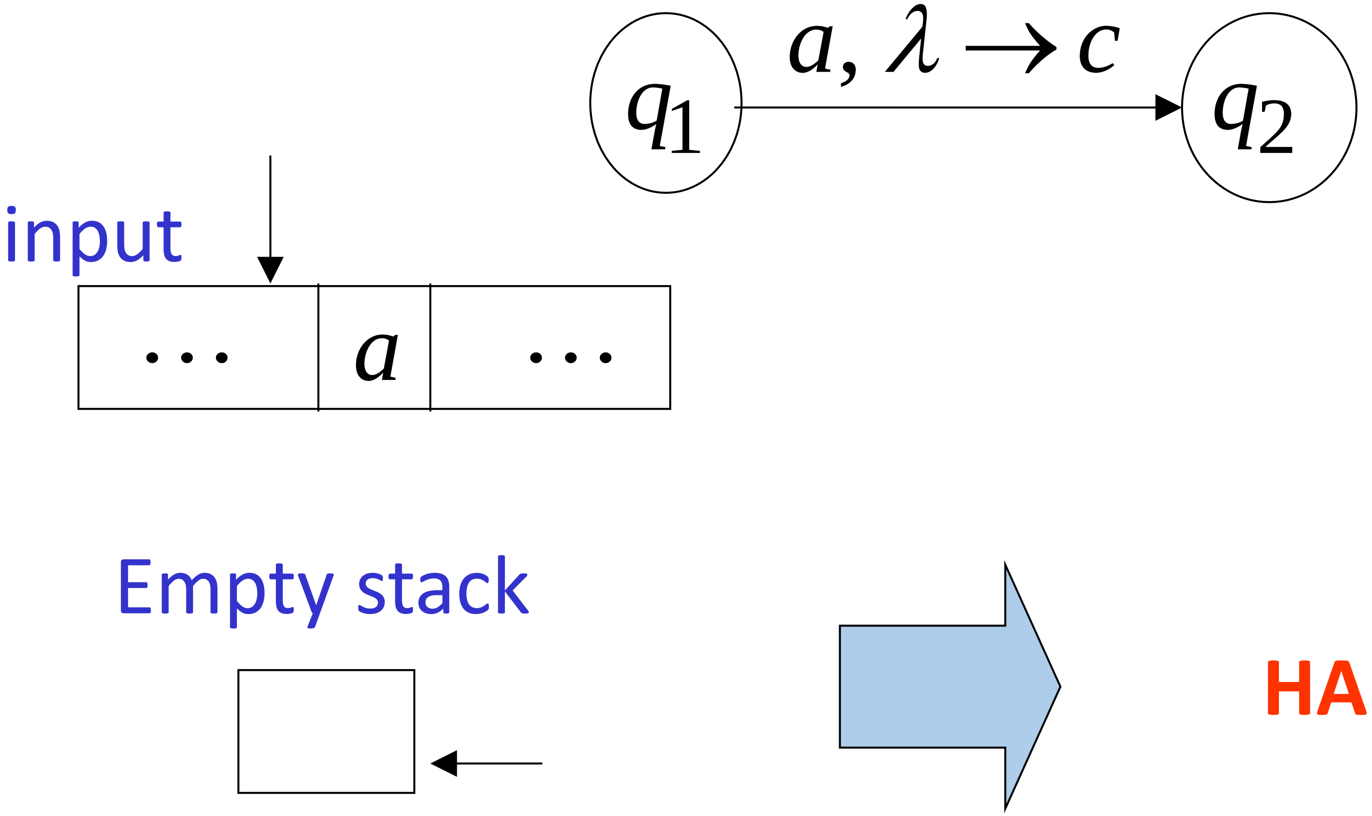
A Bad Transition



The automaton **Halts** in state q_1

and **Rejects** the input string

A Bad Transition



HALT

The automaton **Halts** in state q_1

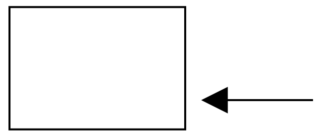
and **Rejects** the input string

No transition is allowed to be followed

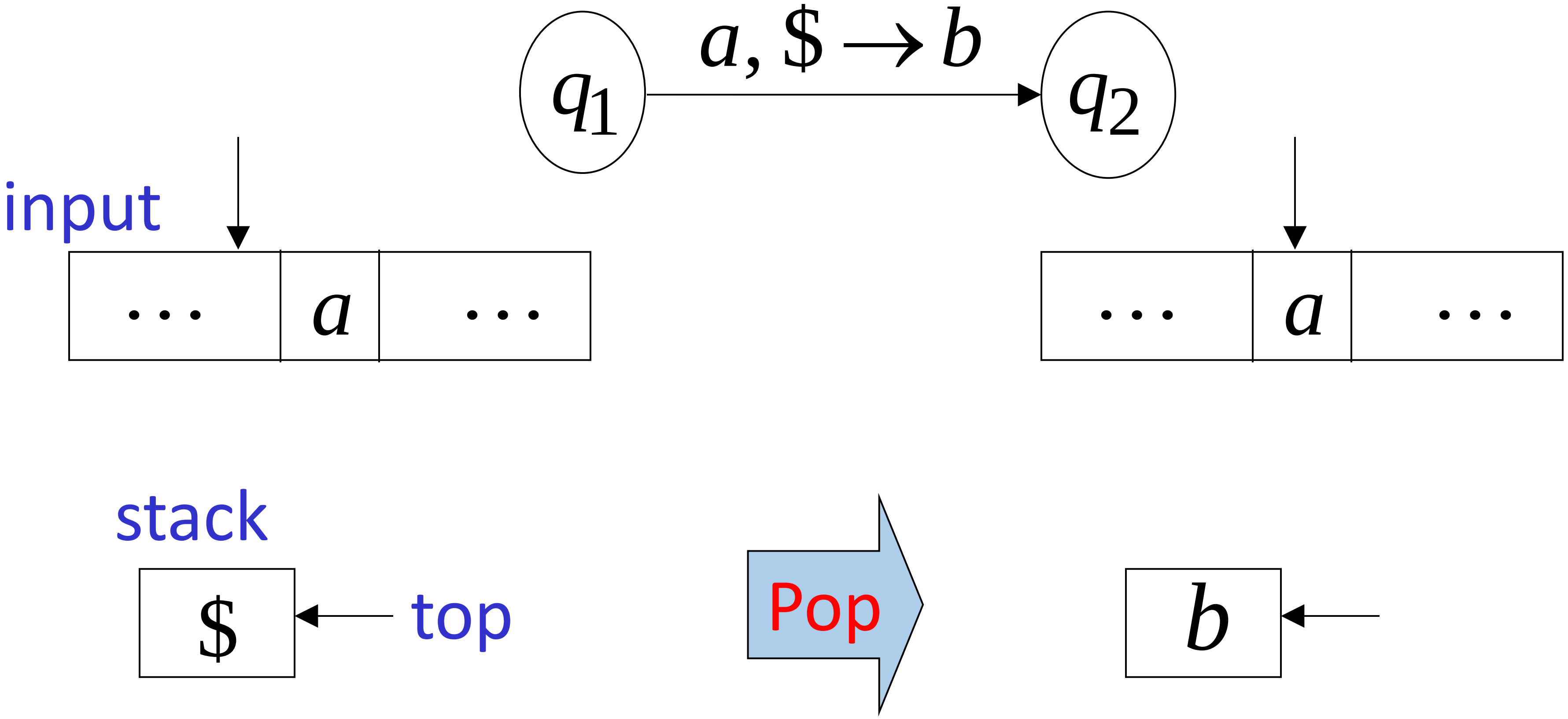
When the stack is empty

$$q_1 \xrightarrow{x,\ y \to z} q_2$$

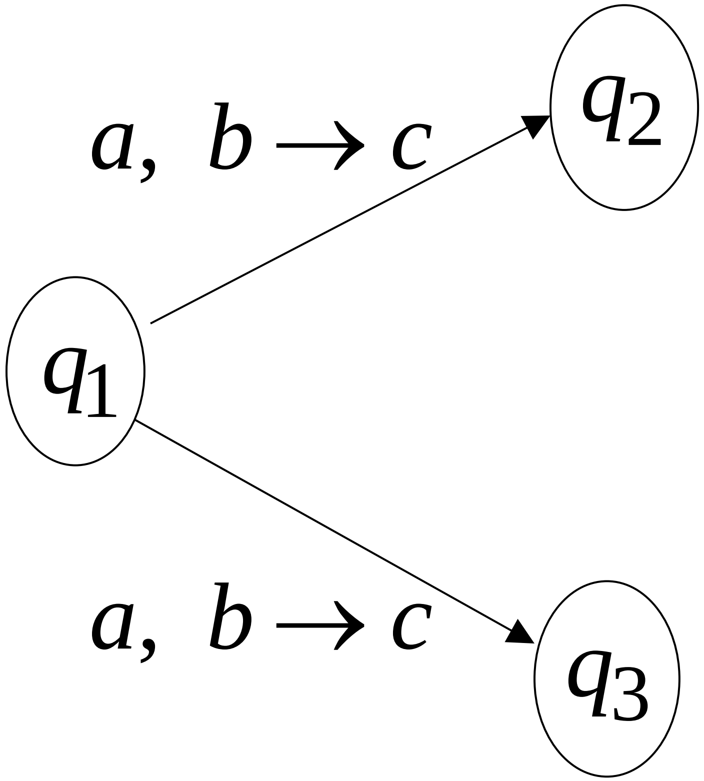
Empty stack

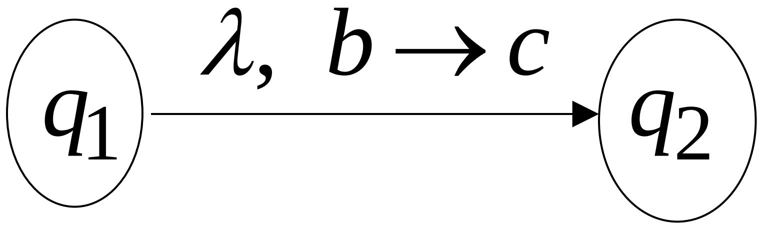


A Good Transition



Non-Determinism





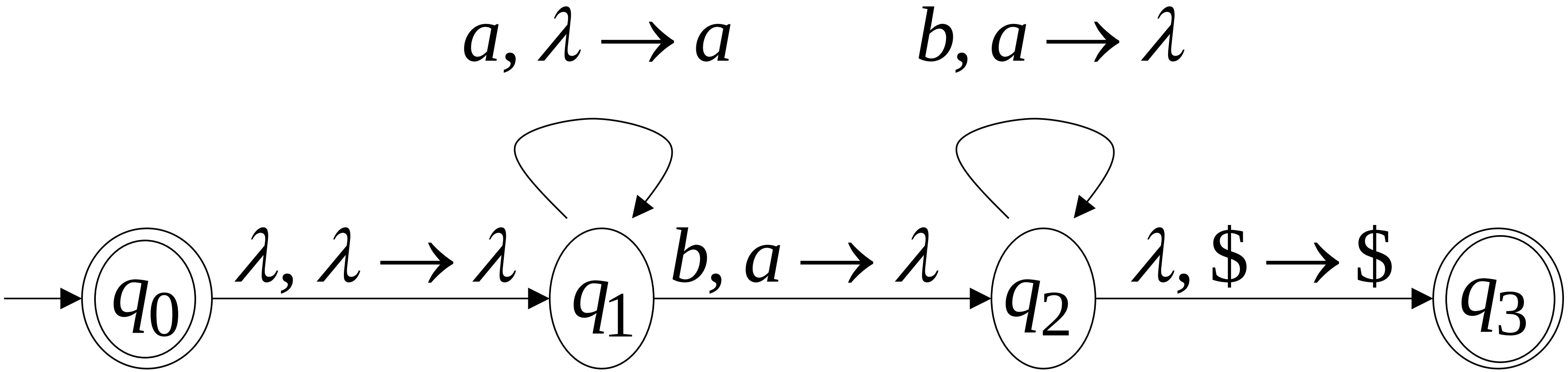
λ – transition

These are allowed transitions in a

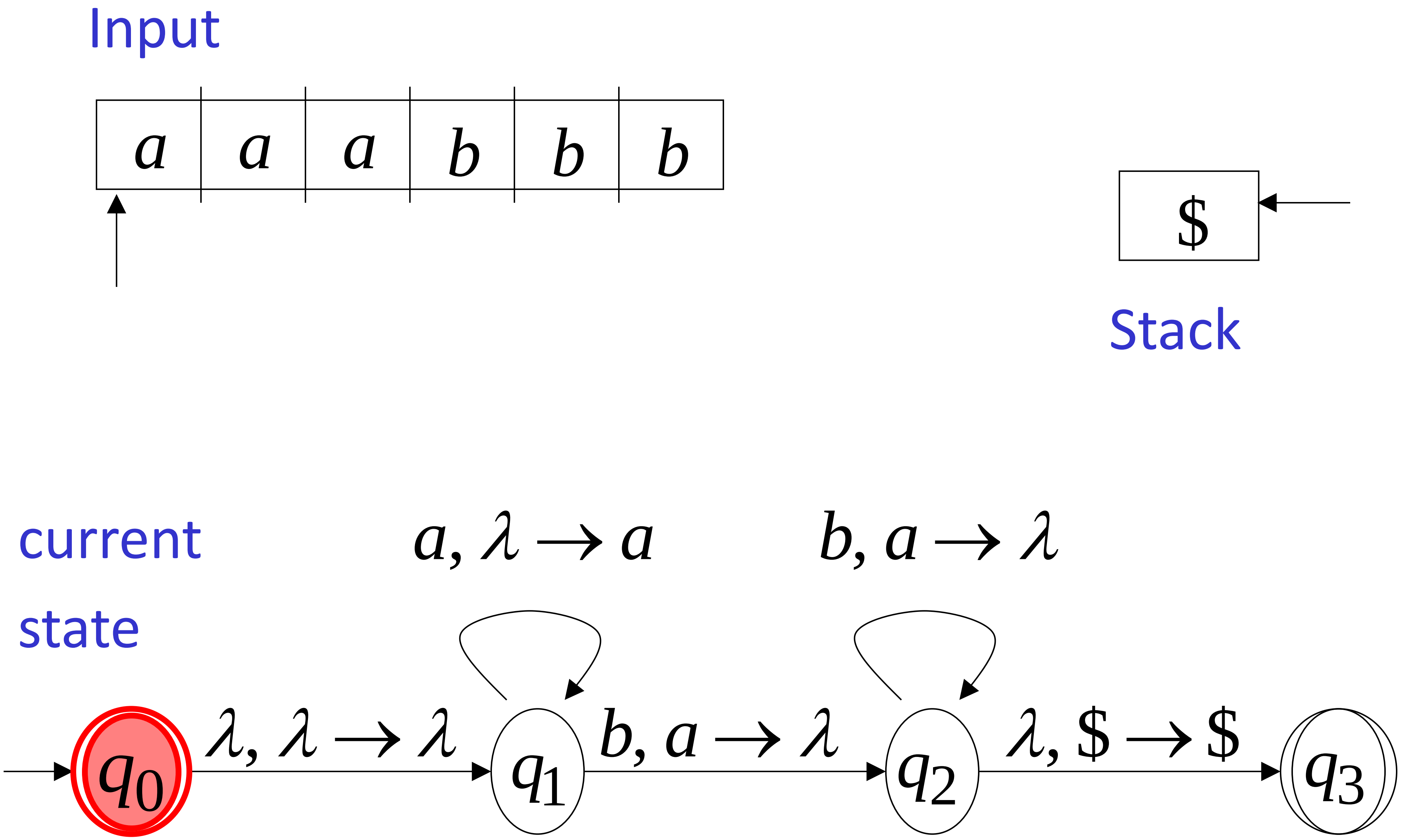
Non-deterministic PDA (NPDA)

NPDA: Non-Deterministic PDA

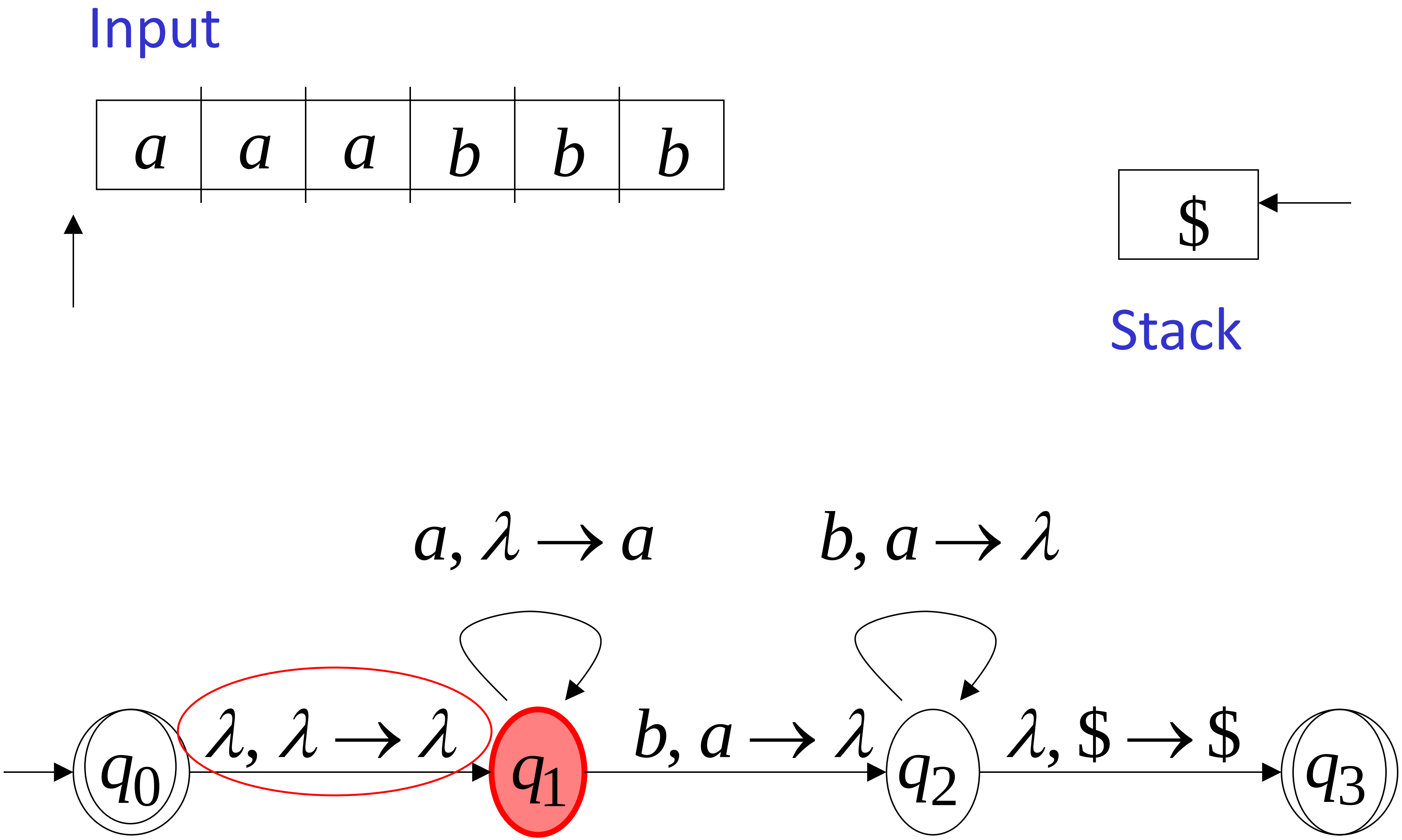
Example:



Execution Example: Time 0

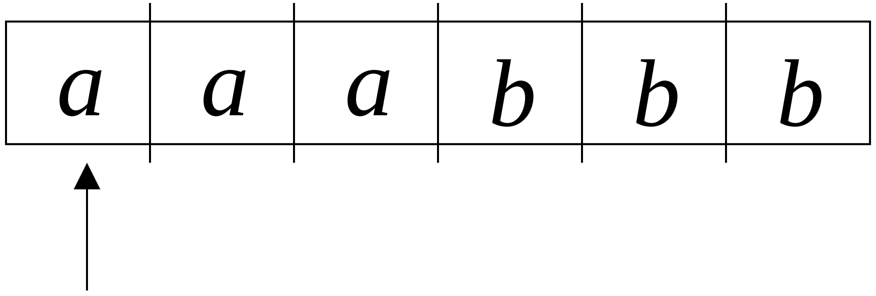


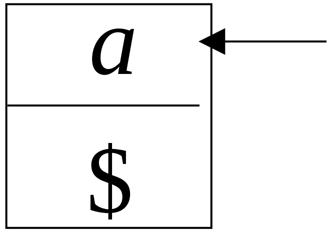
Time 1


Input
a a a b b b
\$
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

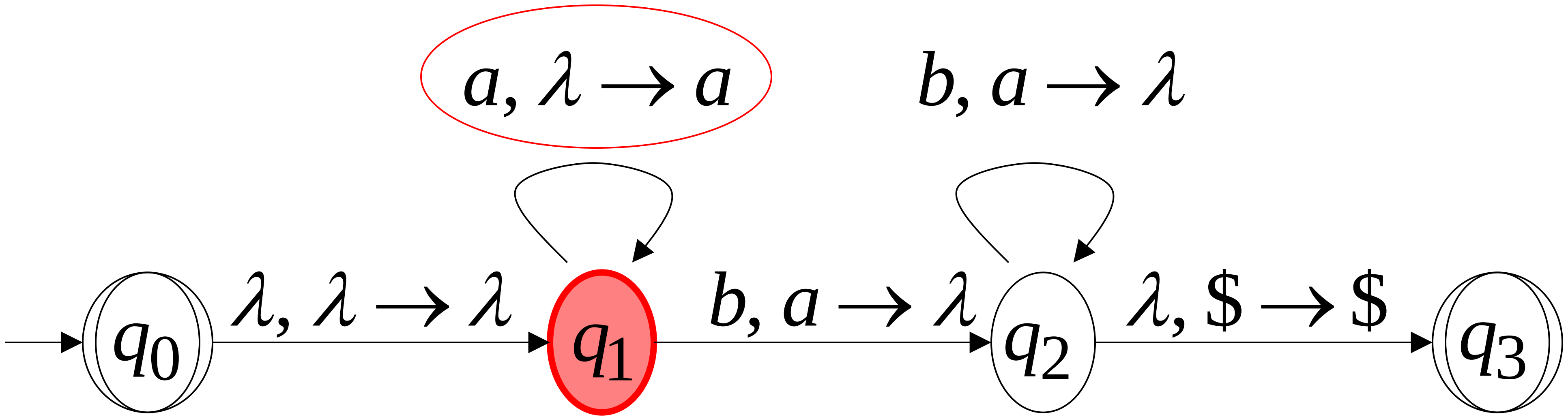
Time 2

Input

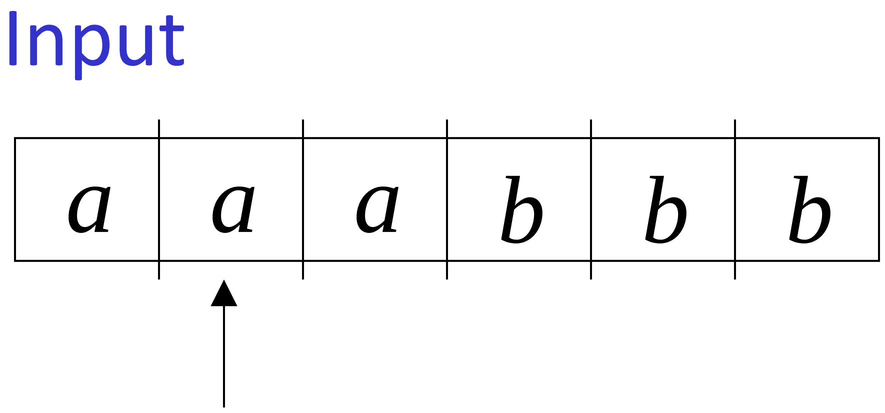




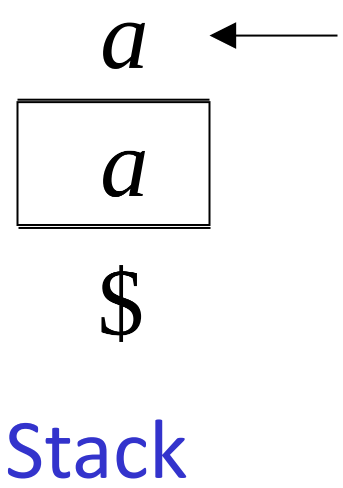
Stack



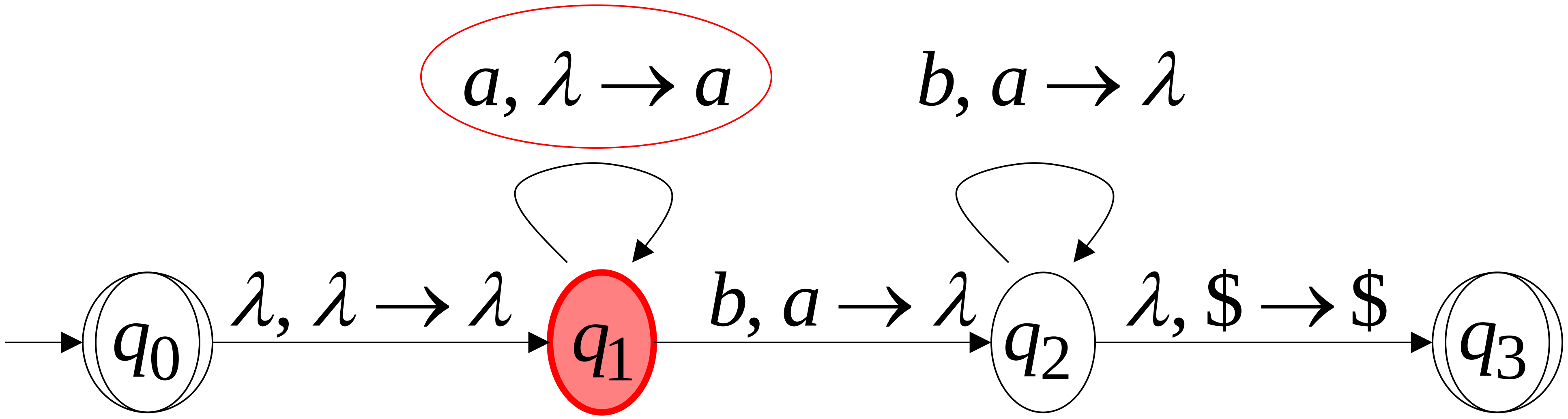
Time 3



Input
a
a
a
b
b
b

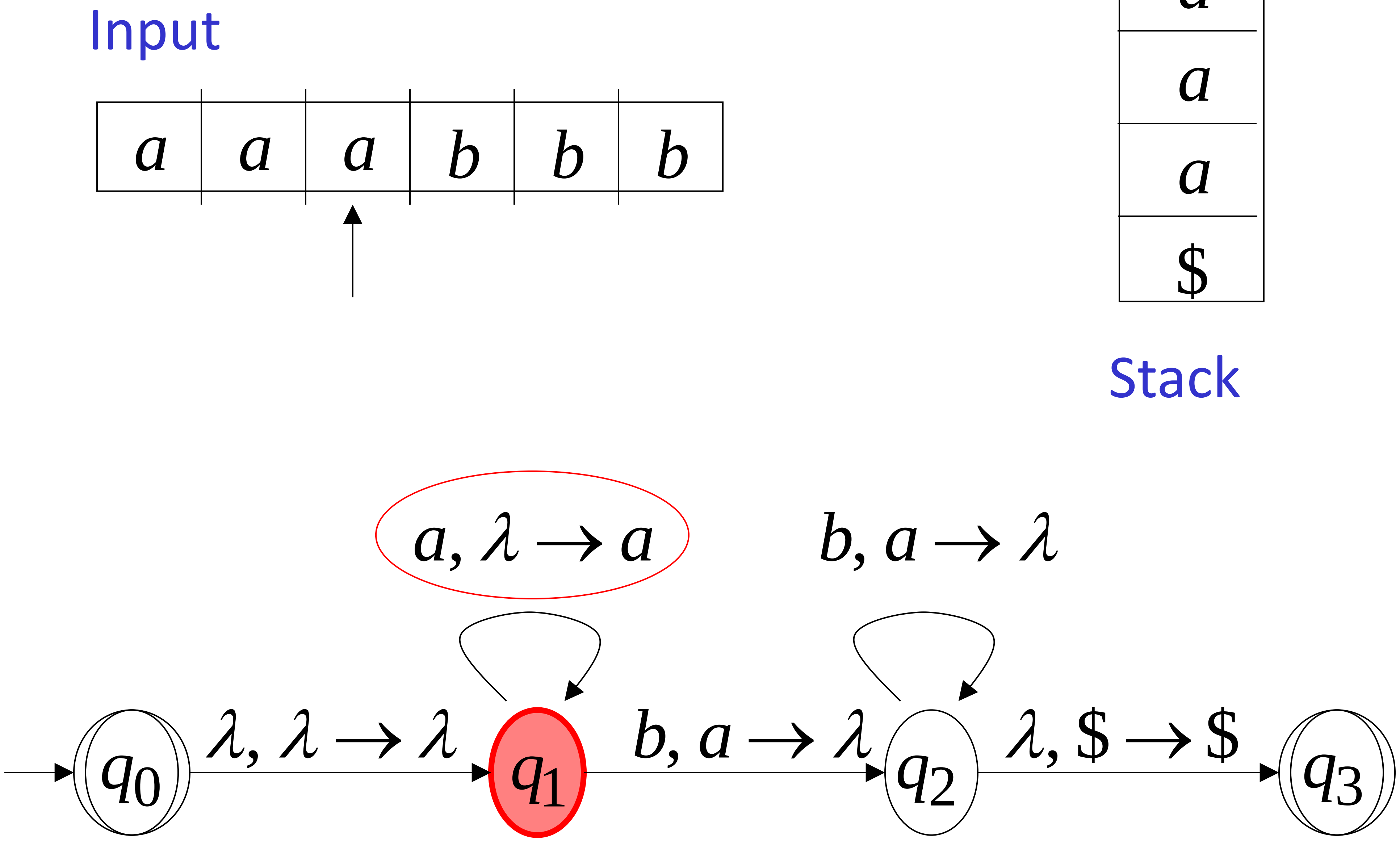


a
a
$
Stack

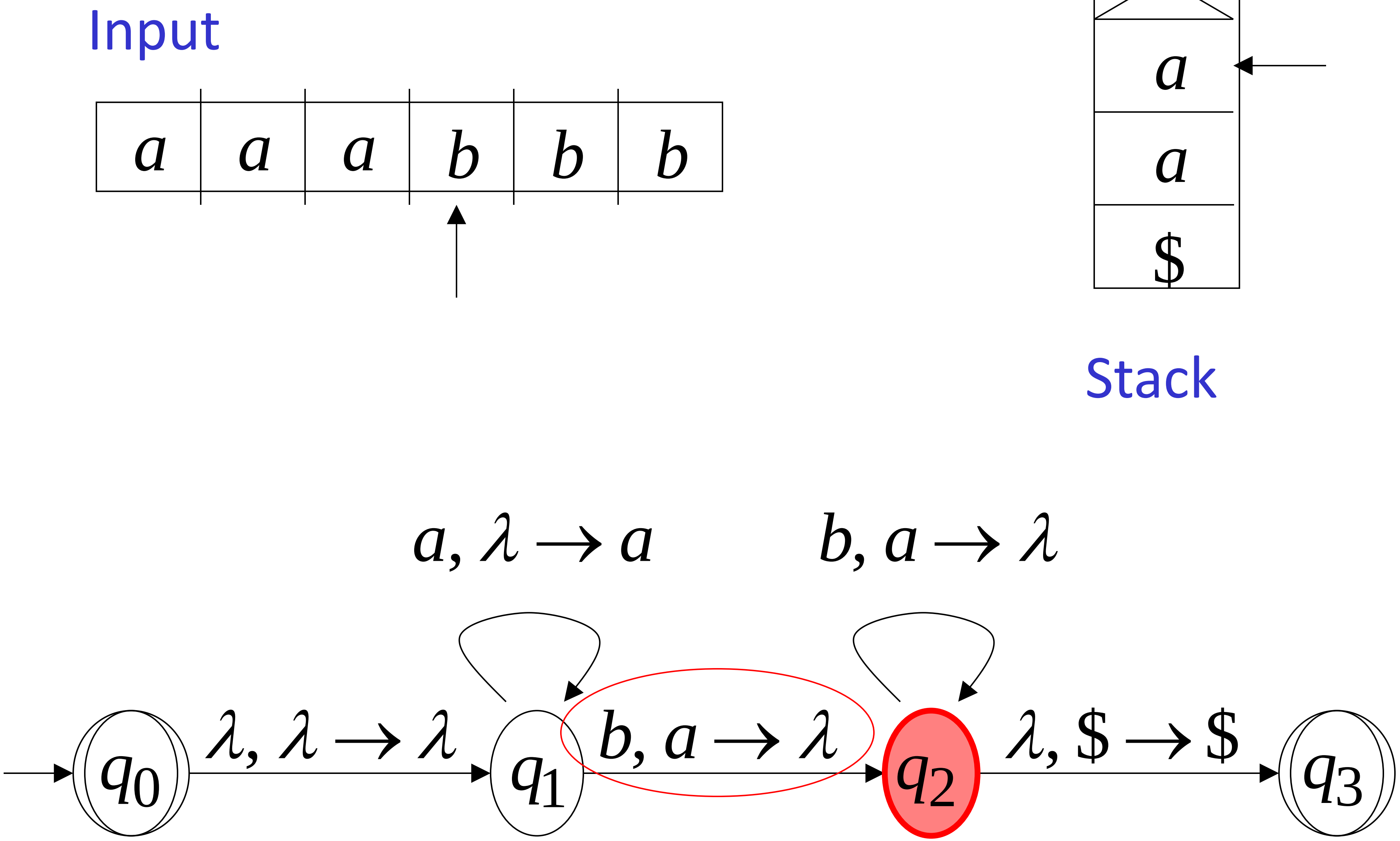


$a, \lambda \to a$
$b, a \to \lambda$
$\lambda, \lambda \to \lambda$
$b, a \to \lambda$
$\lambda, \$ \to \$$
q_0
q_1
q_2
q_3

Time 4


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

Time 5


Input
a a a b b b
a
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

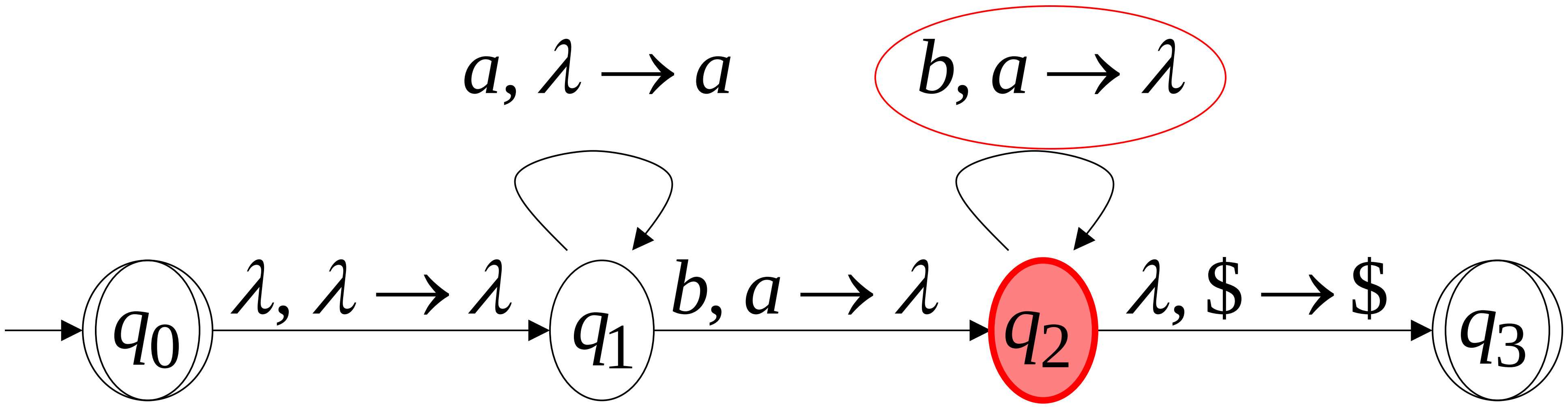
Time 6

Input

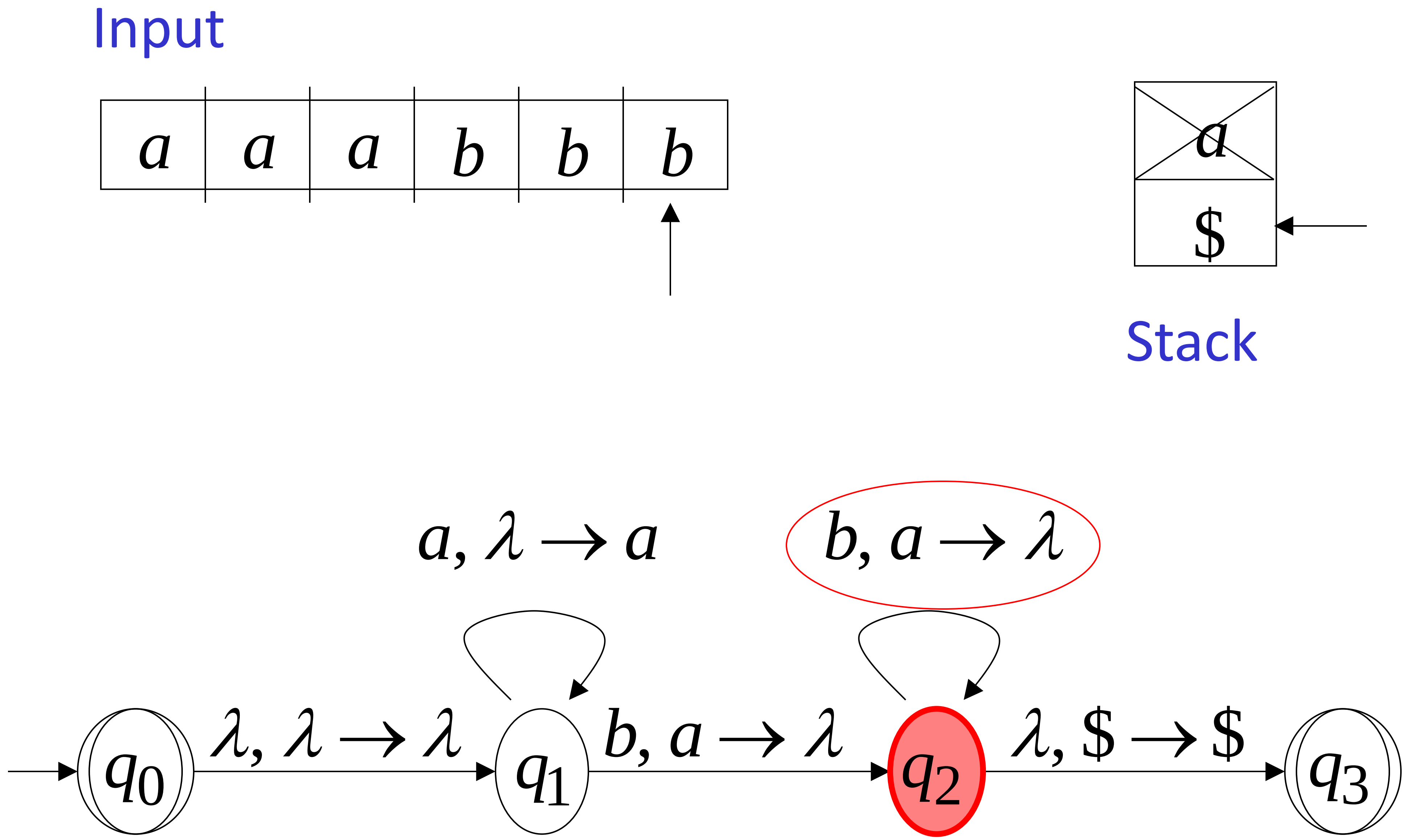
a	a	a	b	b	b

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

Stack

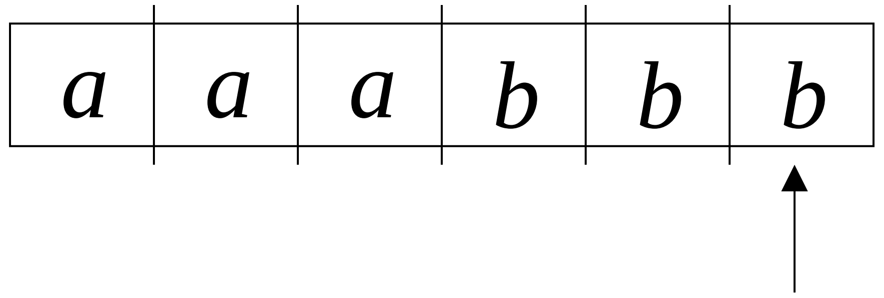


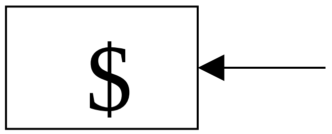
Time 7



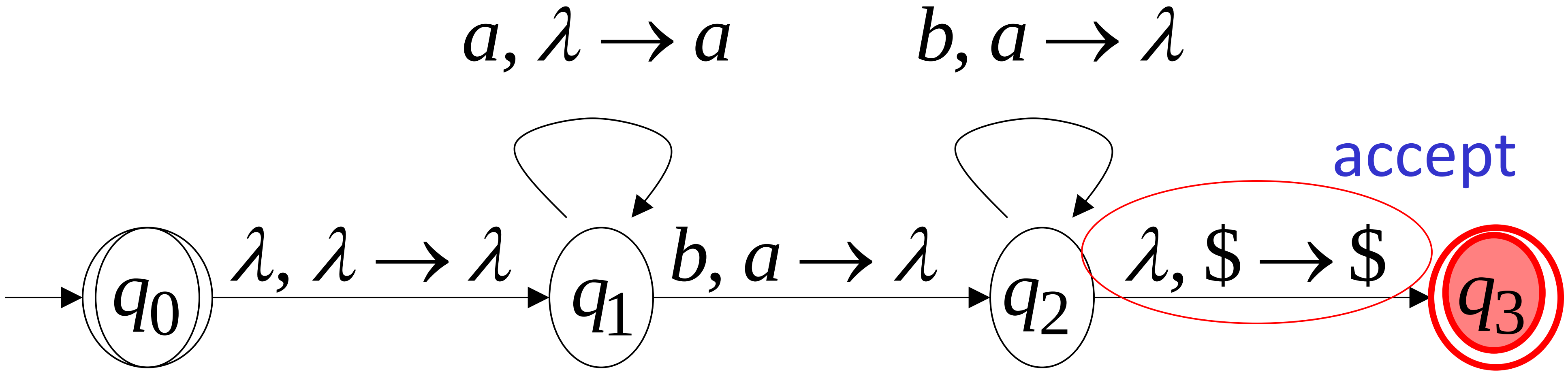
Time 8

Input





Stack



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

All the input is consumed

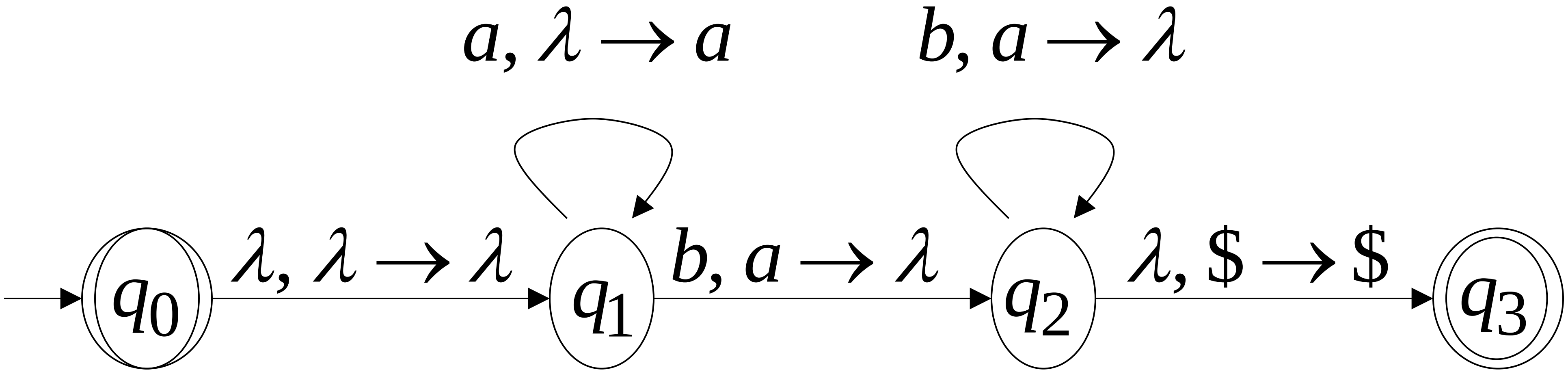
AND

The last state is a final state

At the end of the computation,
we do not care about the stack contents

The input string $aaabbb$

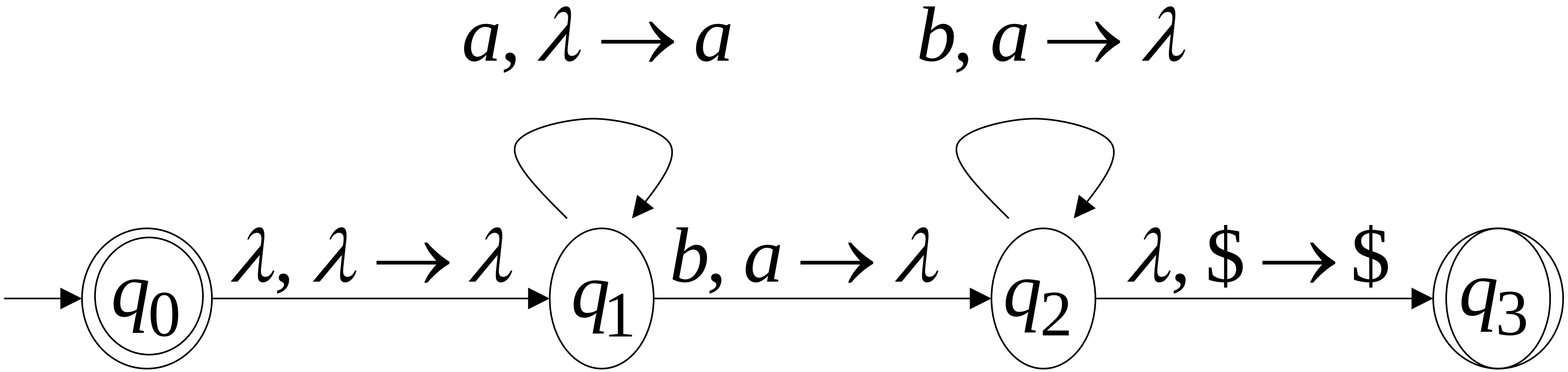
is accepted by the NPDA:



In general,

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

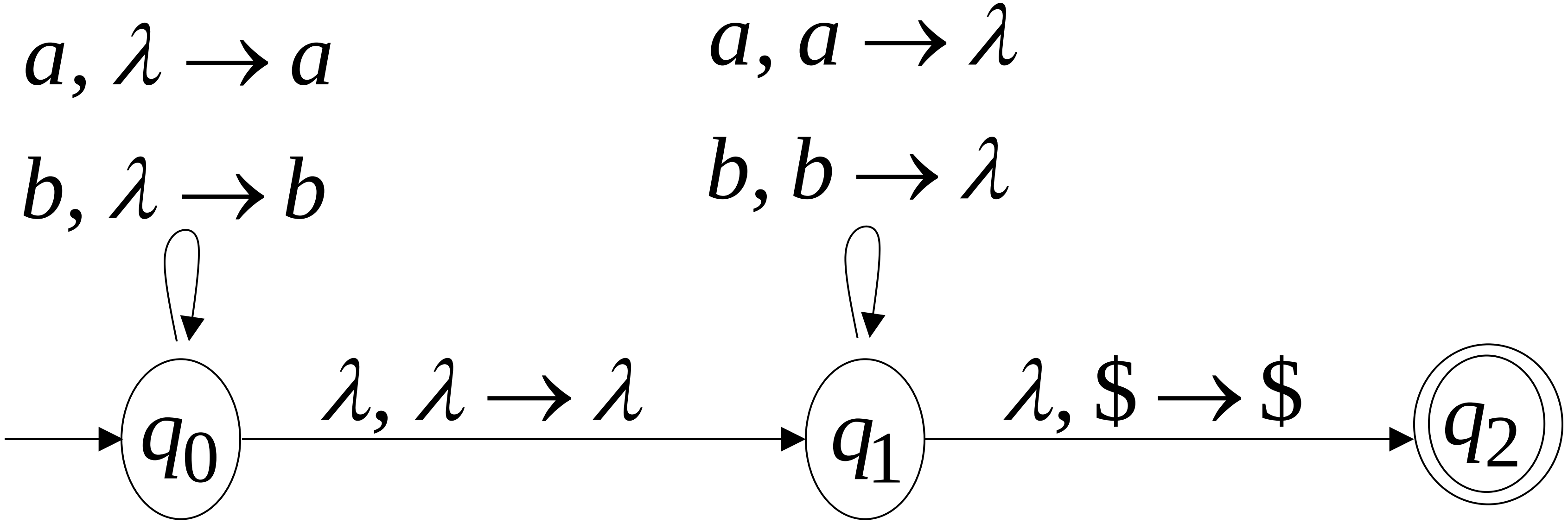
is the language accepted by the NPDA:



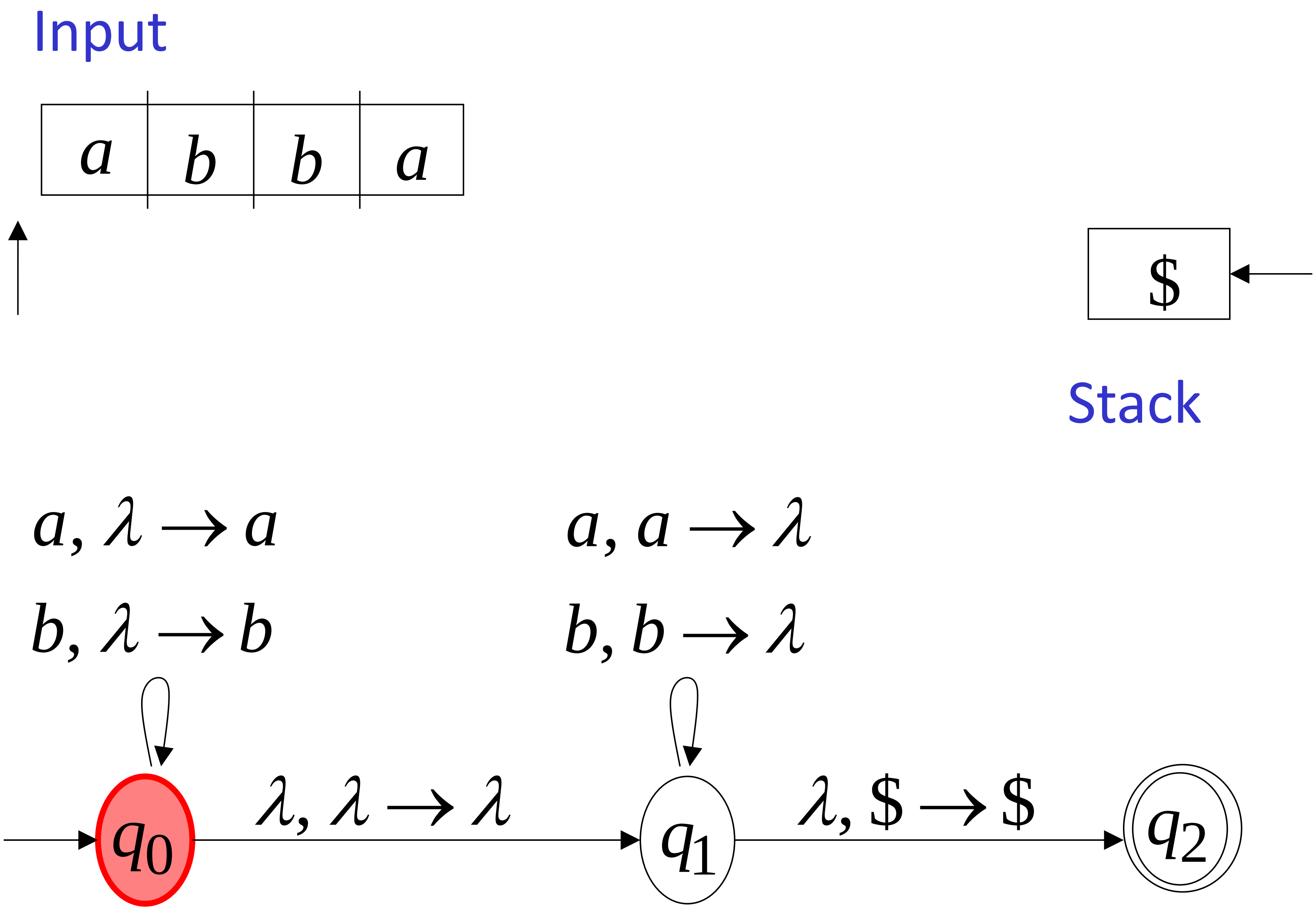
Another NPDA example

NPDA M

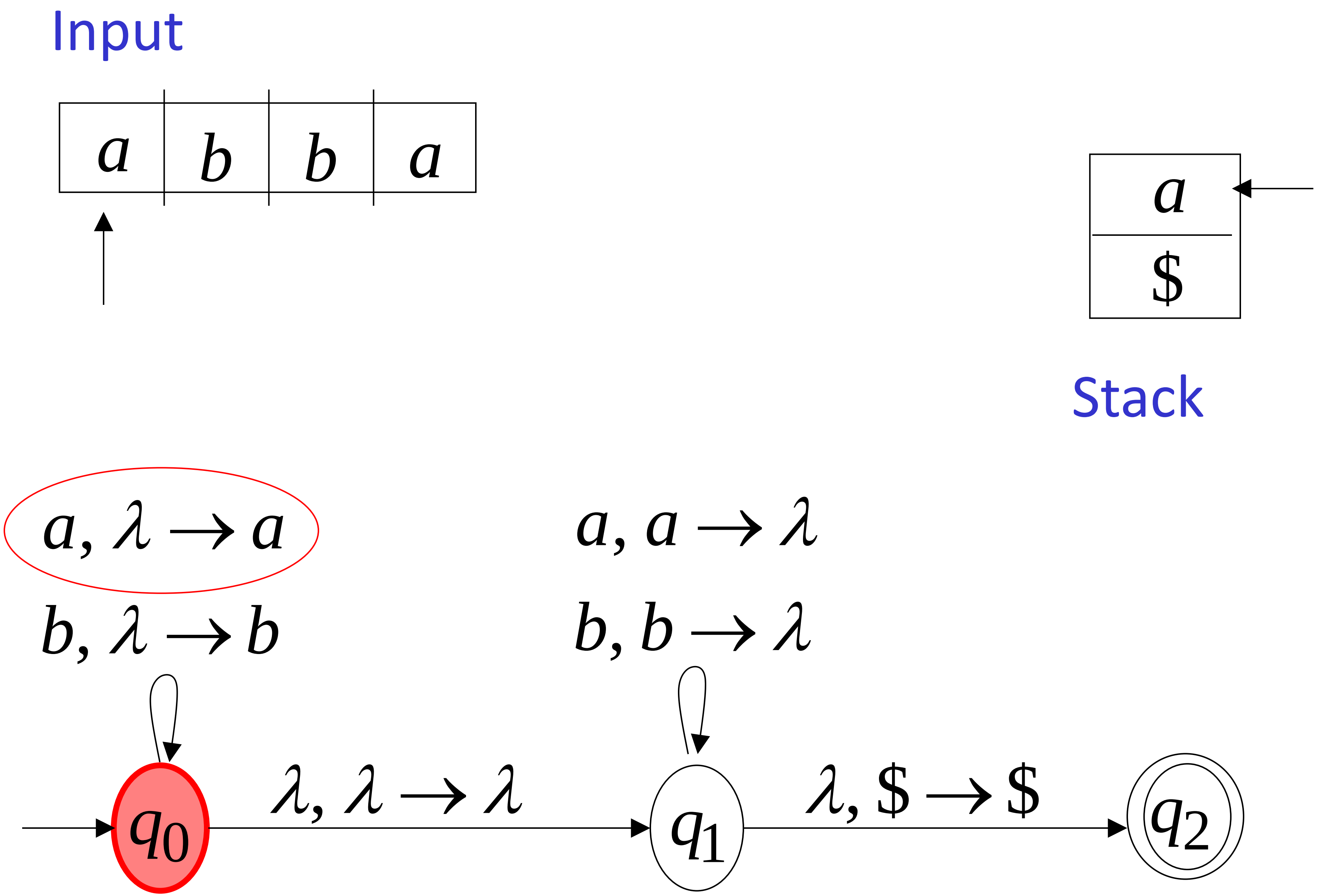
$$L(M) = \{ww^R\}$$



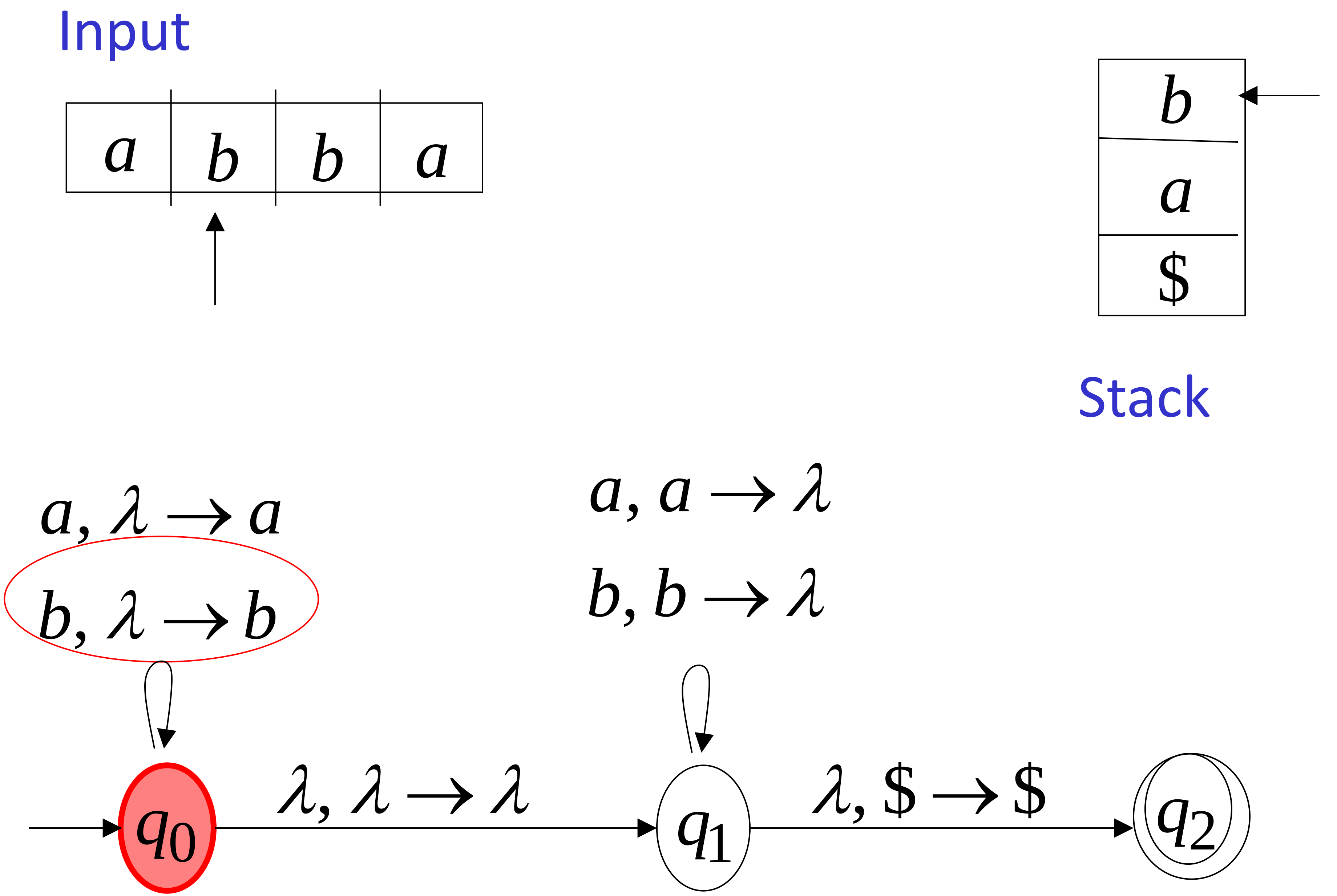
Execution Example: Time 0



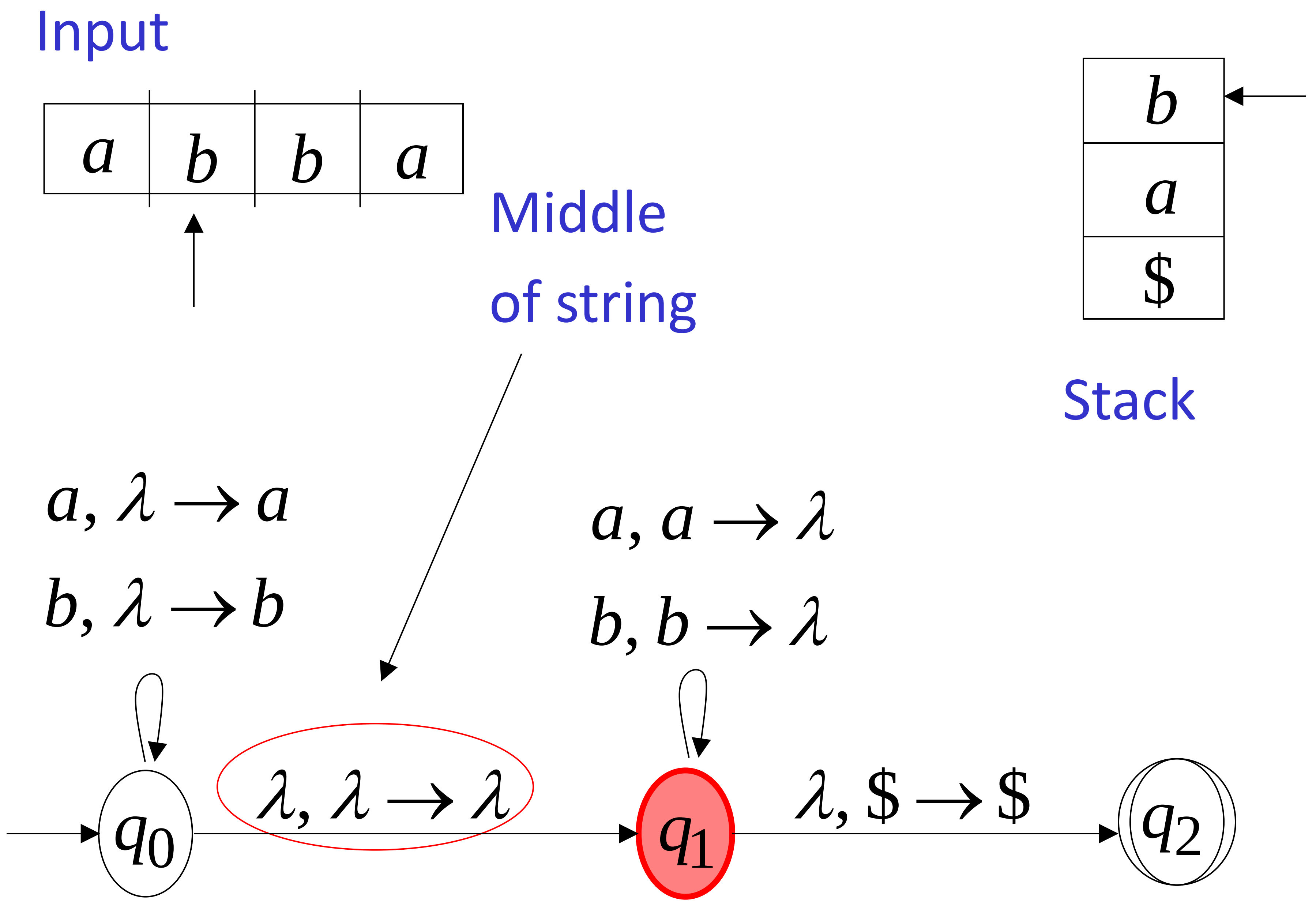
Time 1


Input
a b b a
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 2

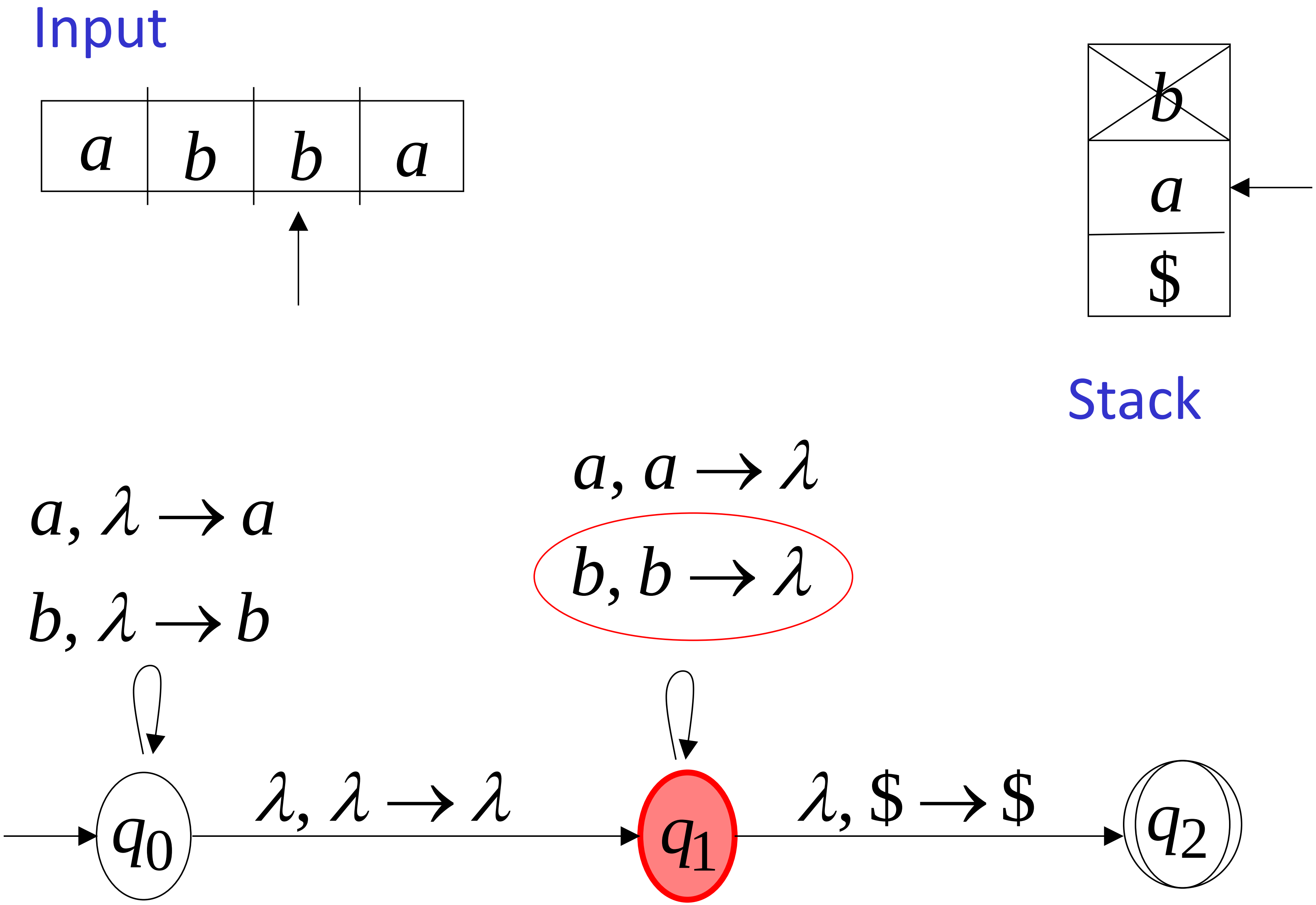


Time 3

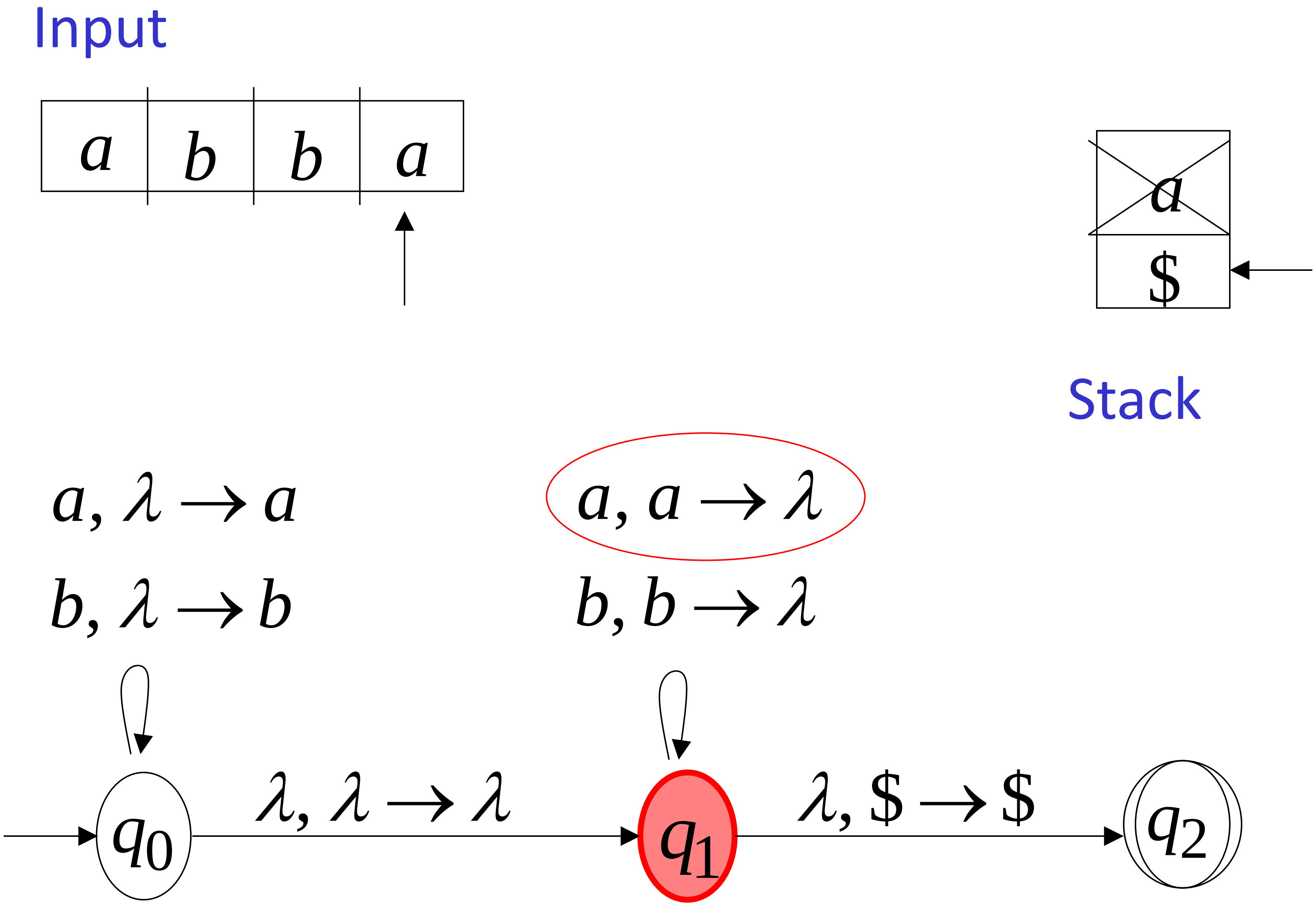


Input
a b b a
Middle
of string
b
a
$
Stack
a, λ → a
b, λ → b
a, a → λ
b, b → λ
λ, λ → λ
λ, $ → $
q0
q1
q2

Time 4


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

Time 5


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

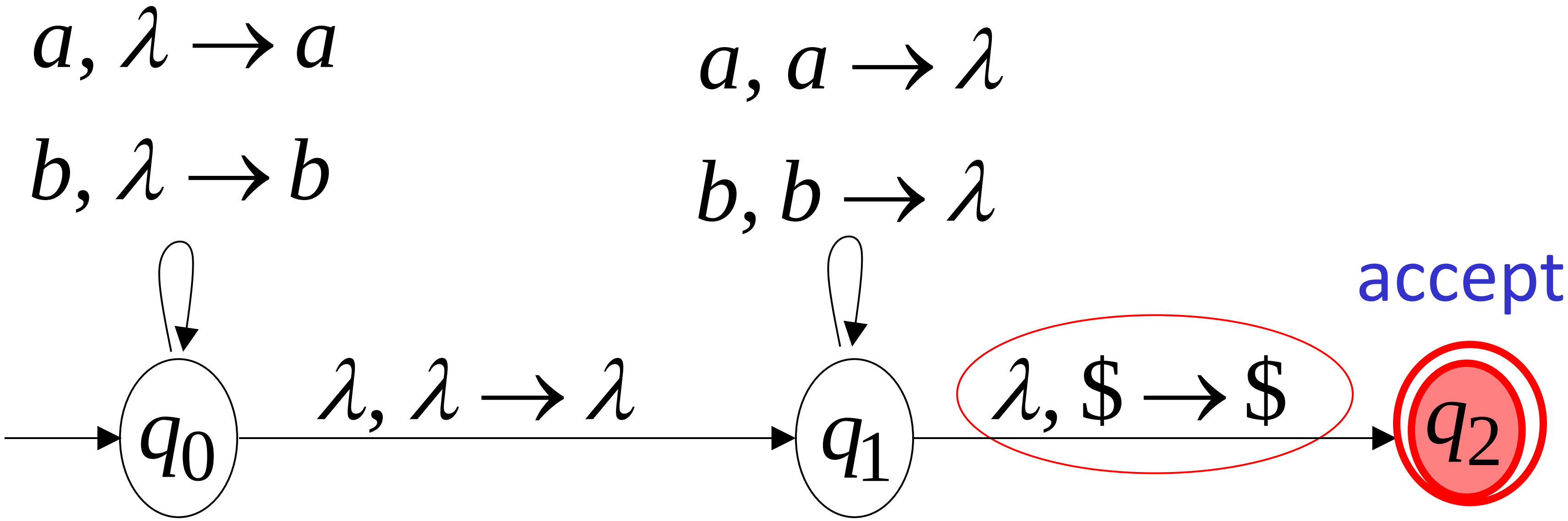
Time 6

Input

a	b	b	a

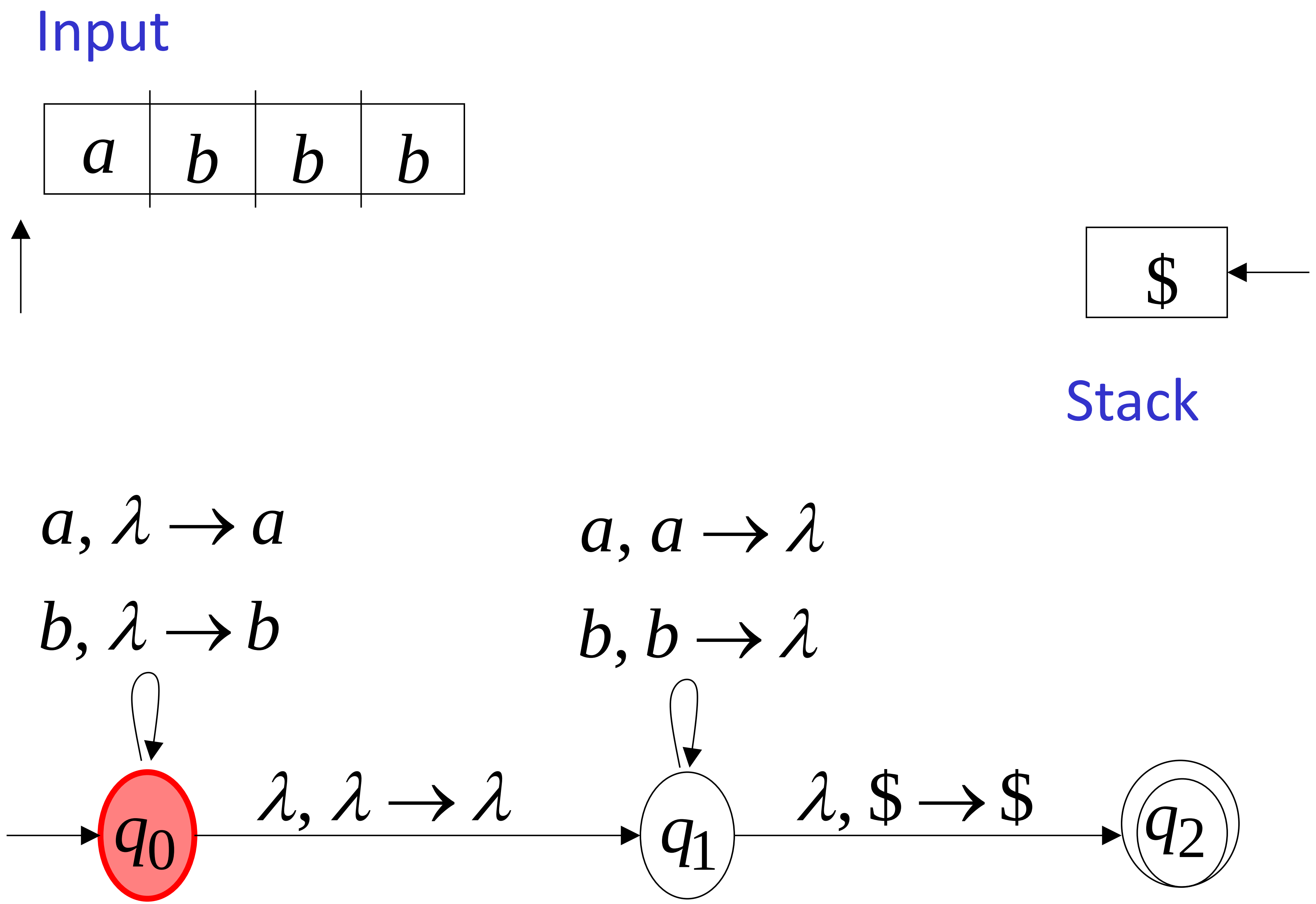
$\$$

Stack



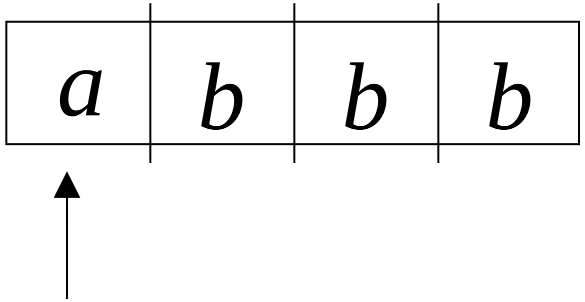
Rejection Example:

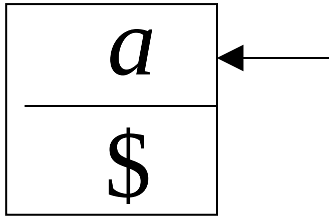
Time 0



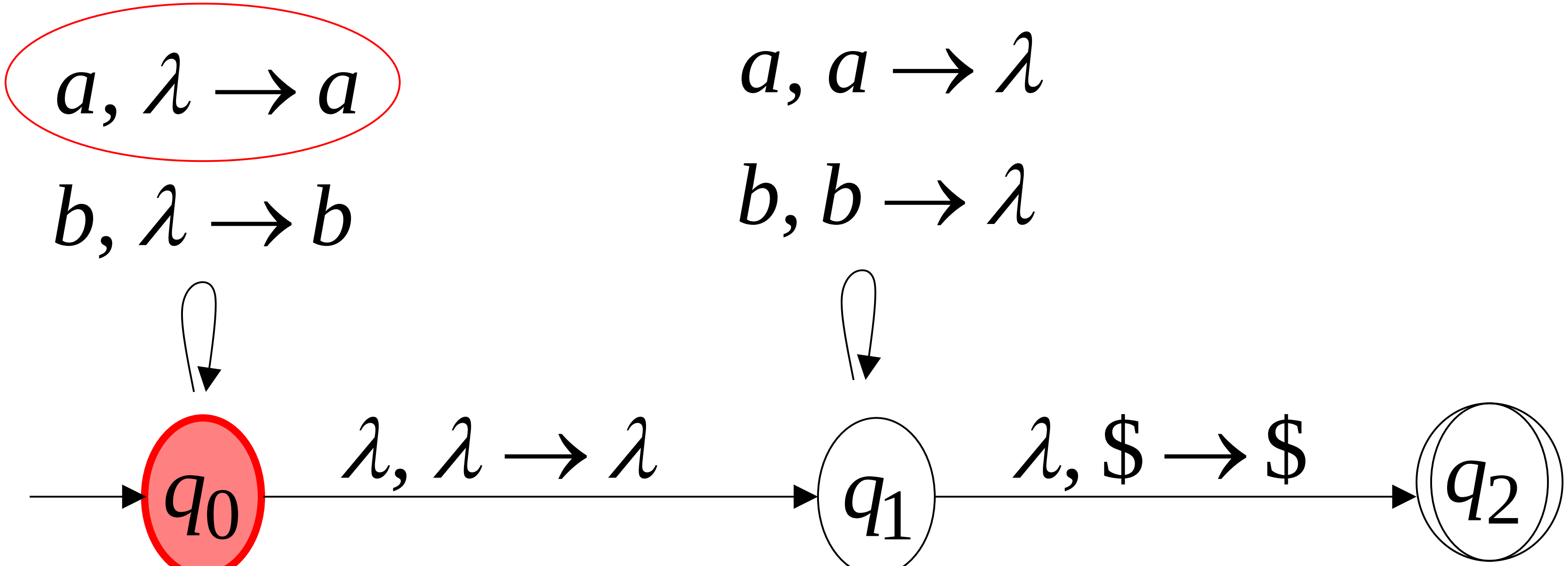
Time 1

Input

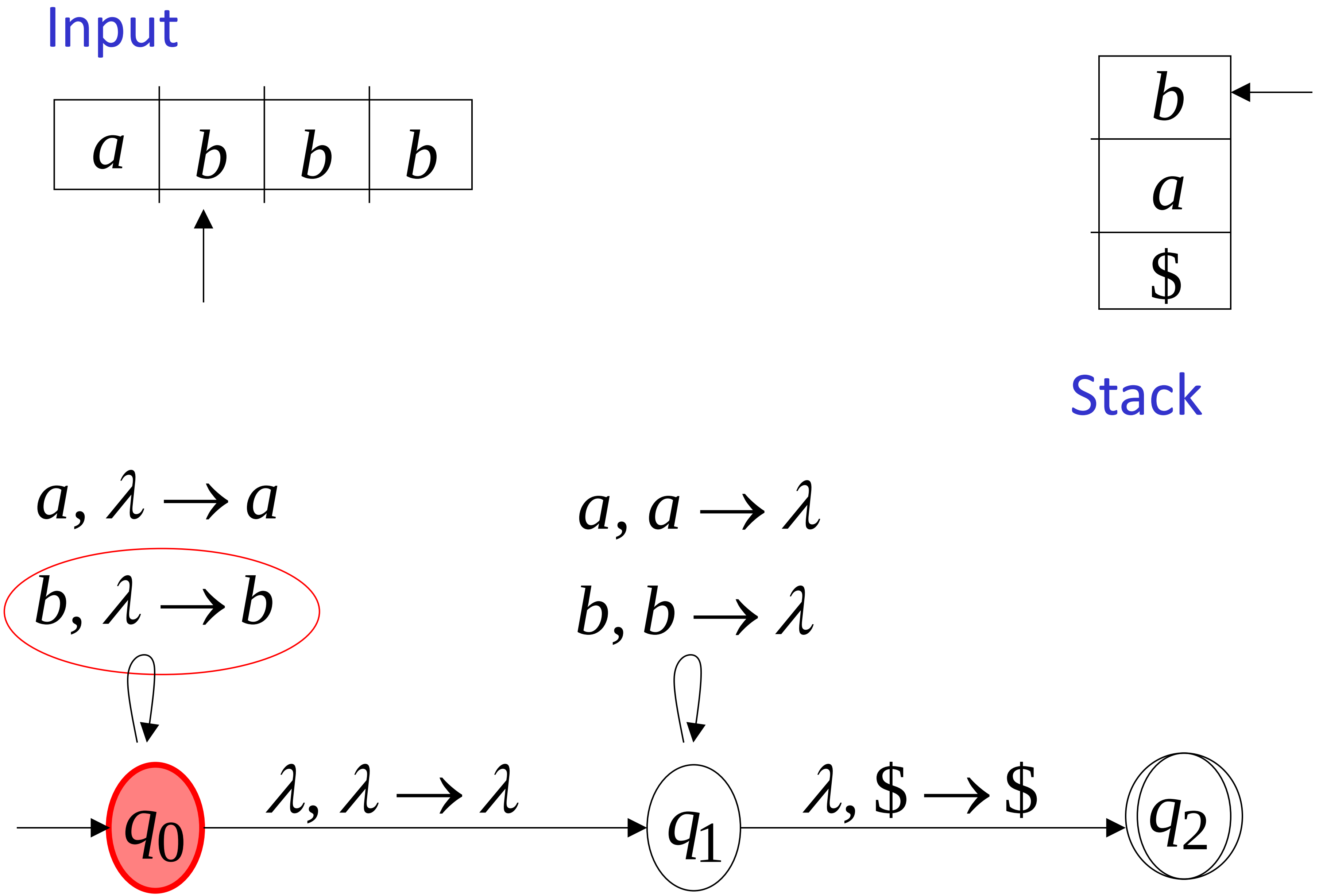




Stack

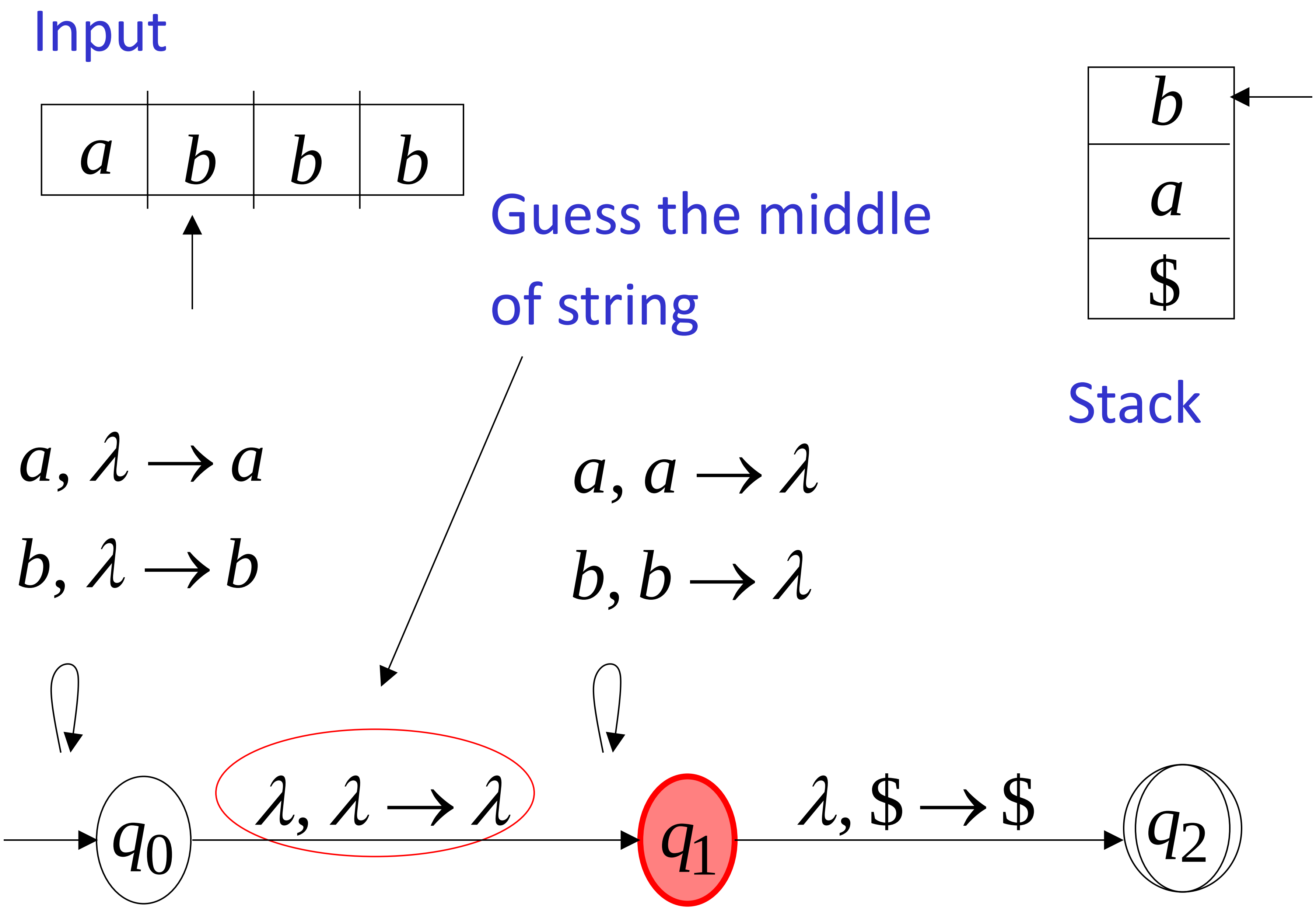


Time 2

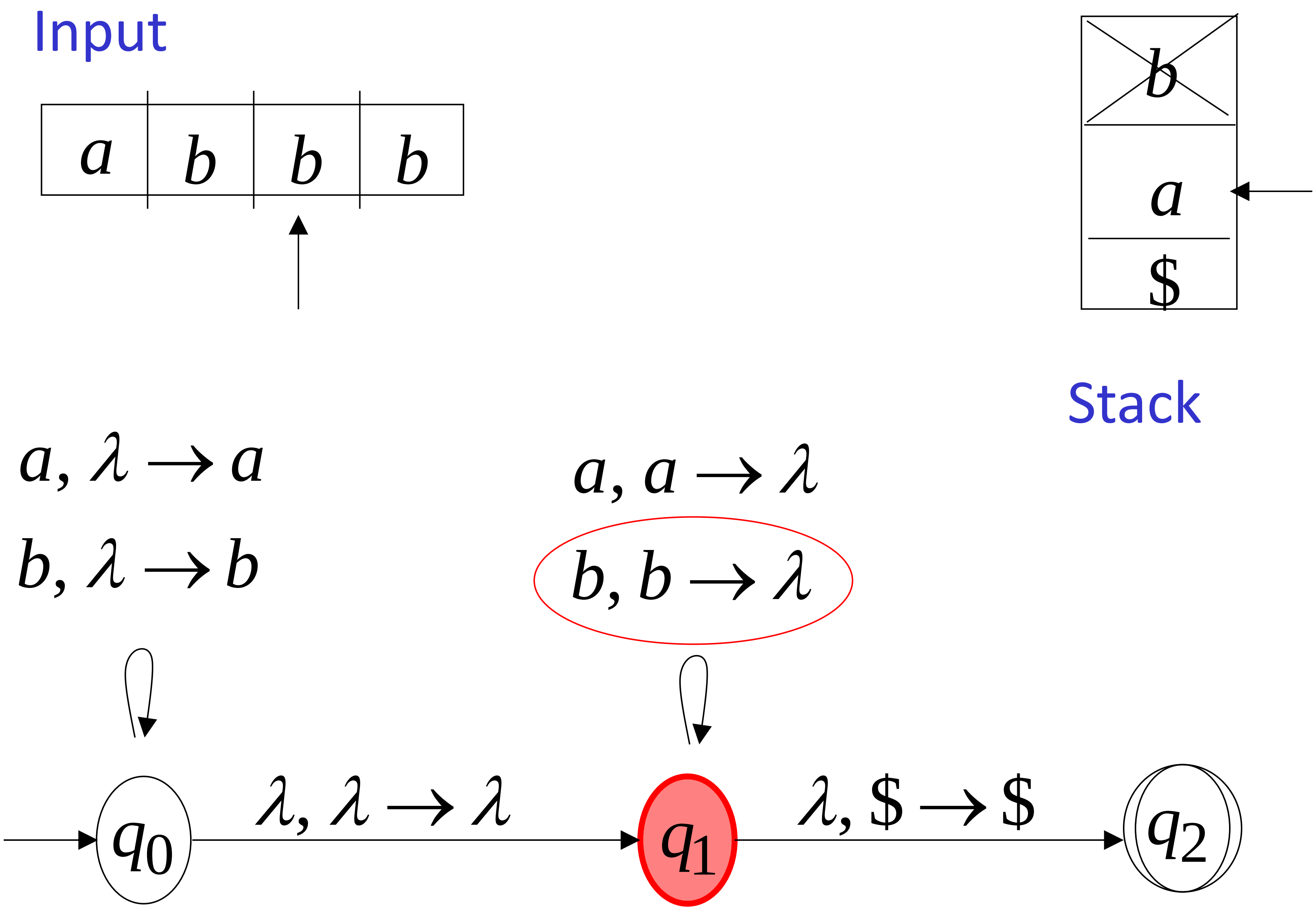


Input
a b b b
b
a
$
Stack
$a, \lambda \to a$
$b, \lambda \to b$
$a, a \to \lambda$
$b, b \to \lambda$
$\lambda, \lambda \to \lambda$
$\lambda, \$ \to \$$
q_0
q_1
q_2

Time 3


Input
a b b b
Guess the middle of string
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

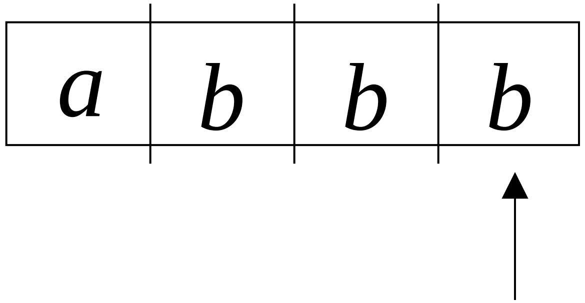
Time 4


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

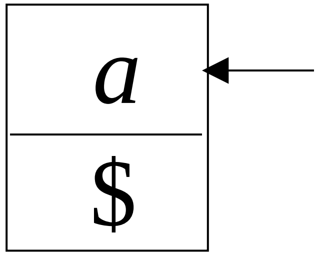
Time 5

Input

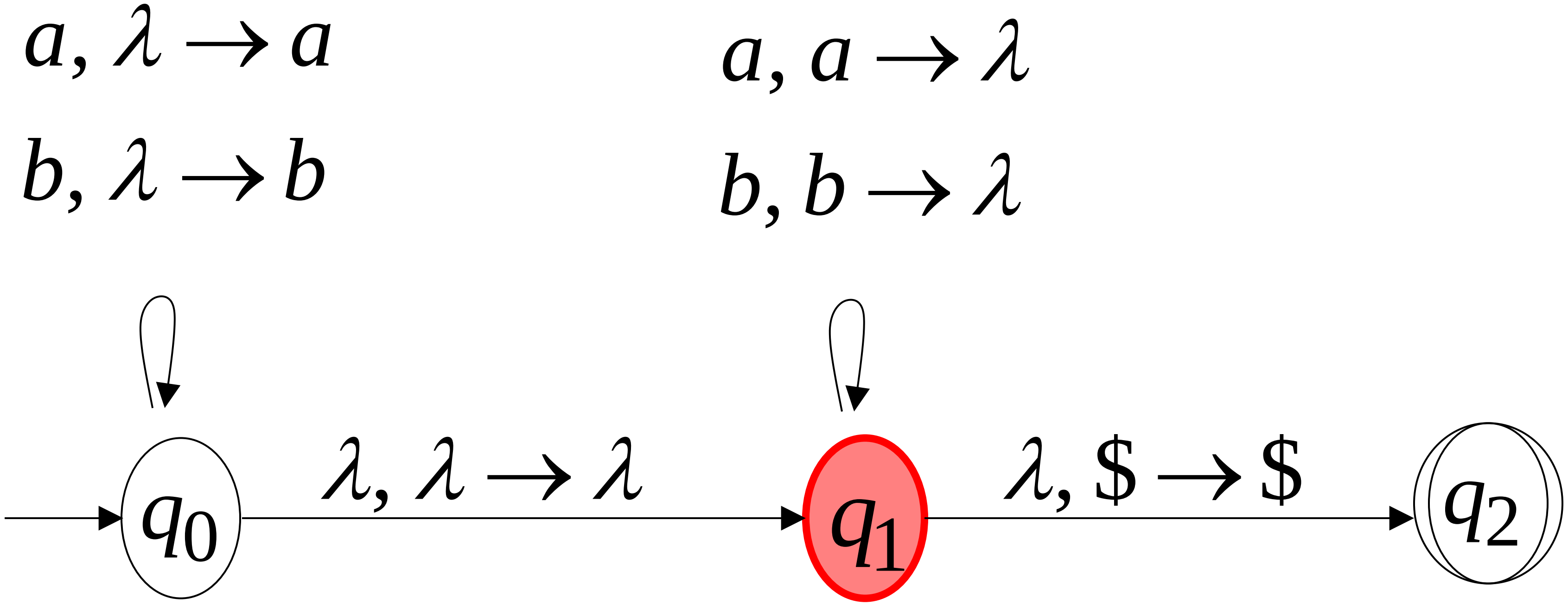
There is no possible transition.



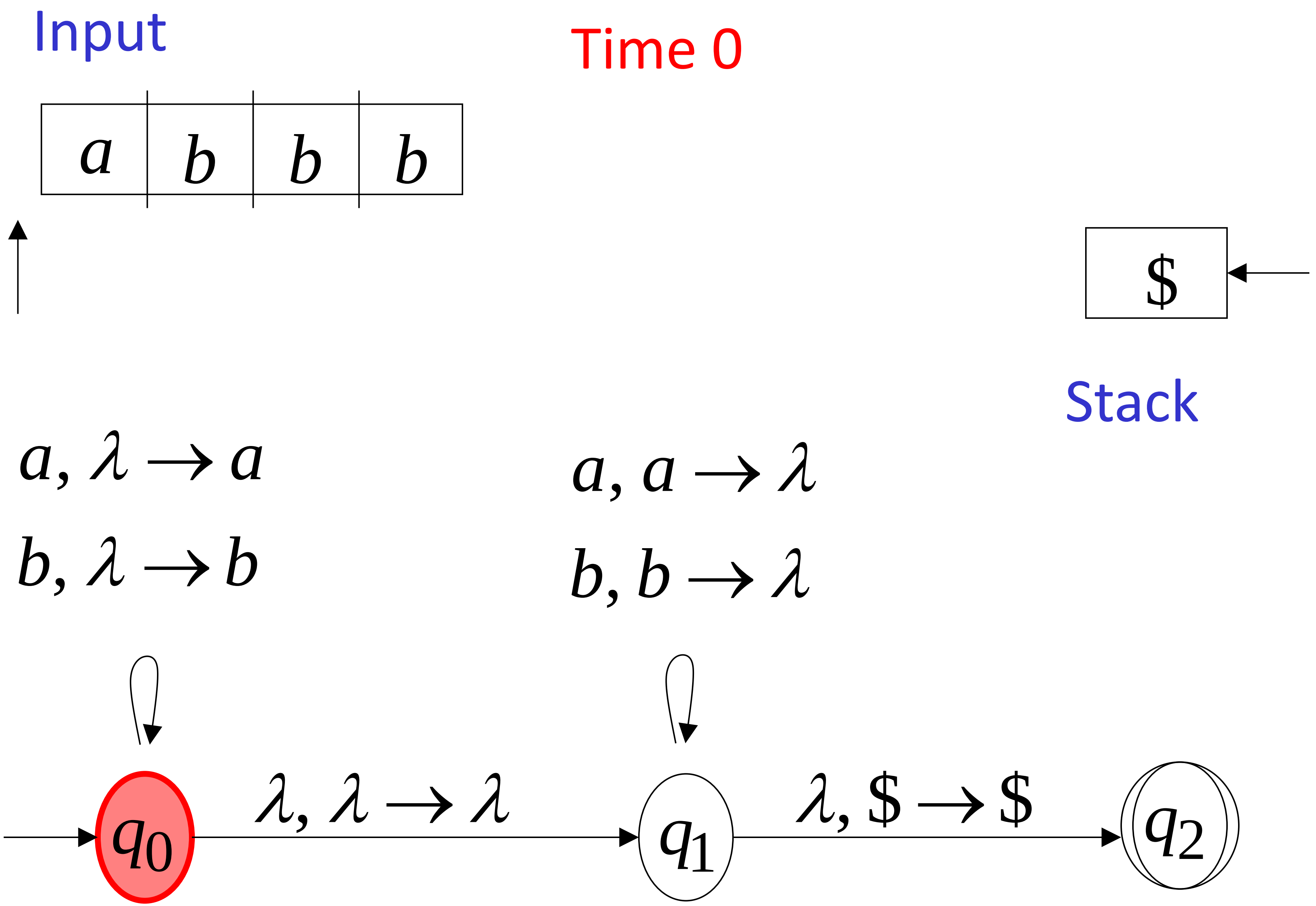
Input is not consumed



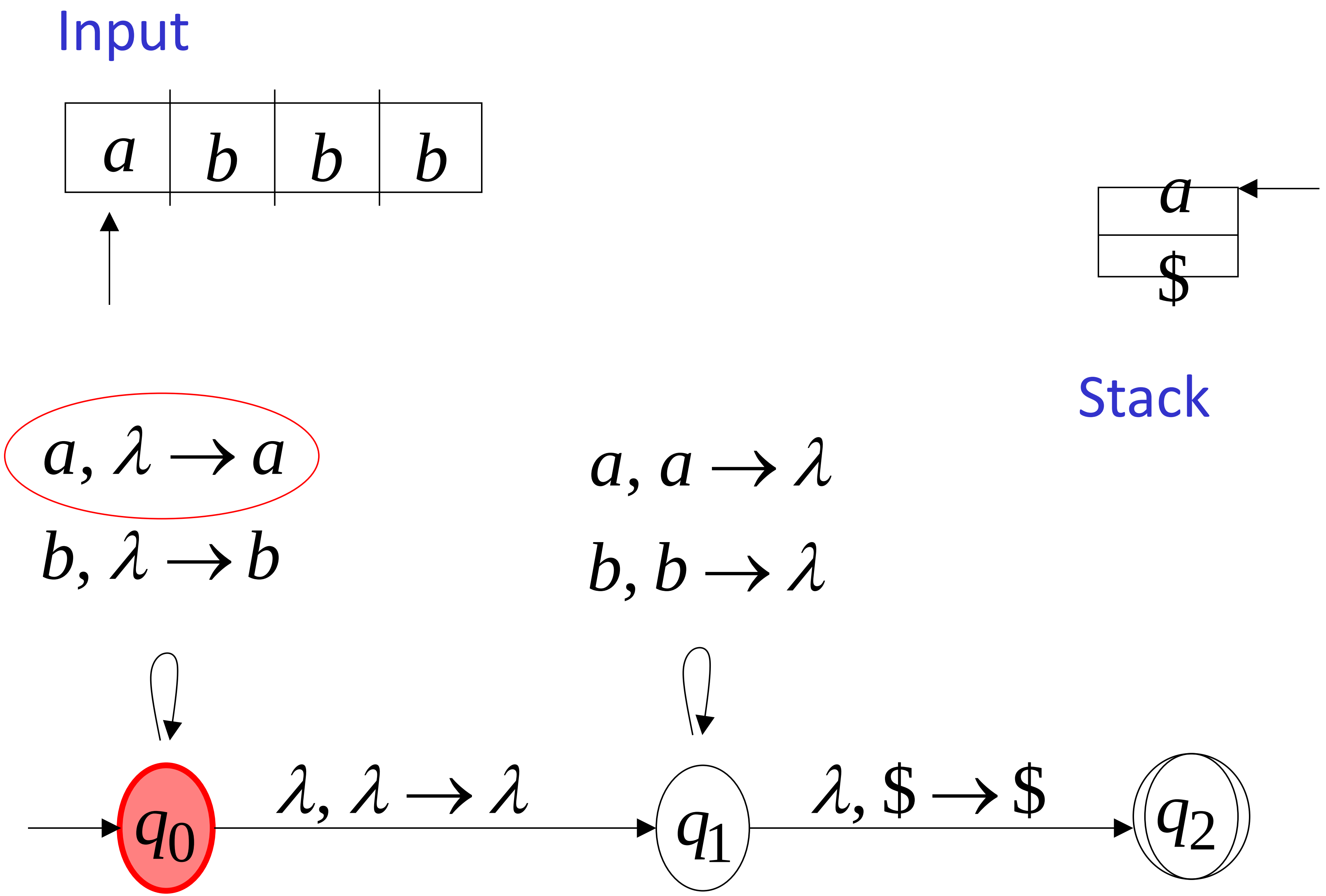
Stack



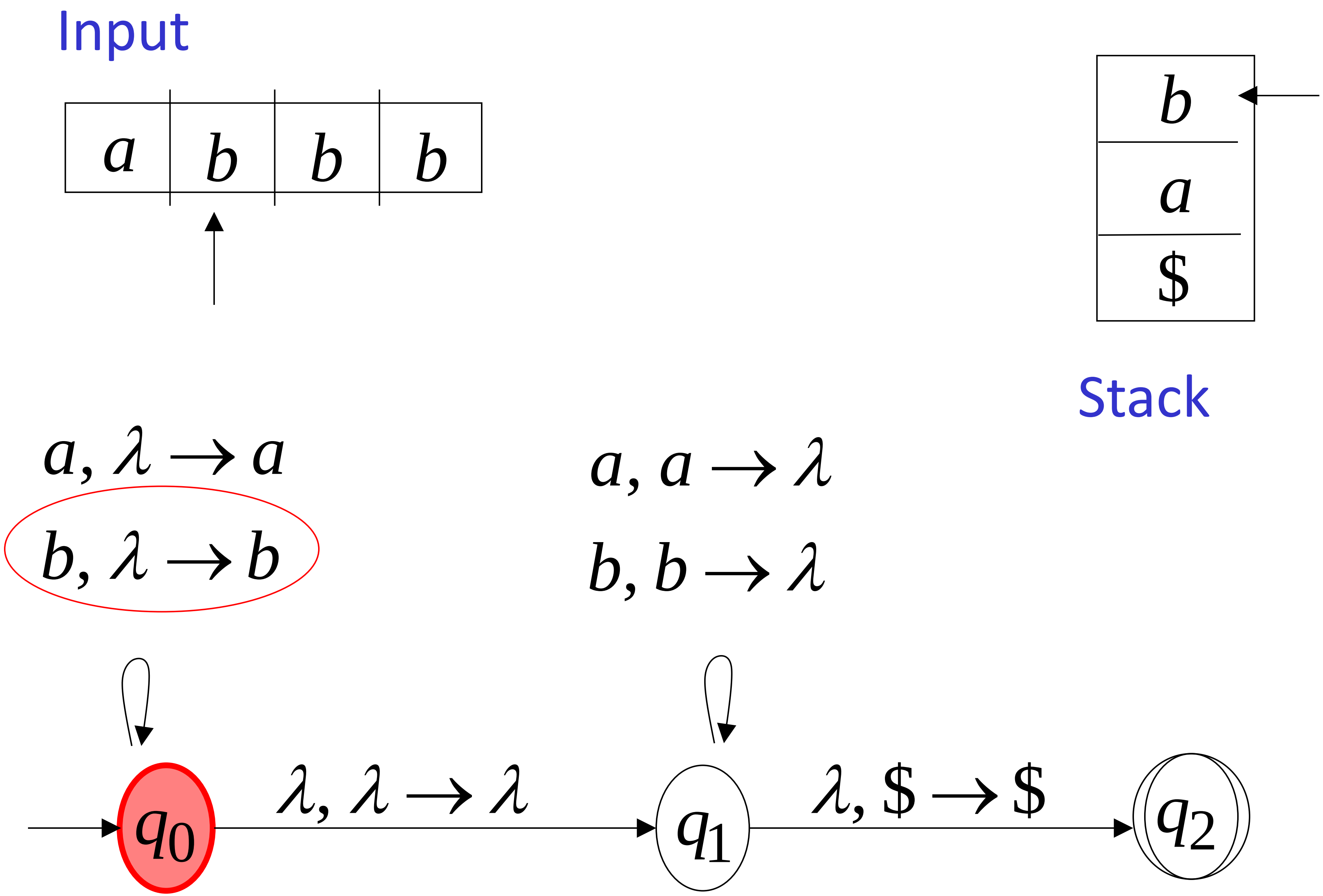
Another computation on same string:



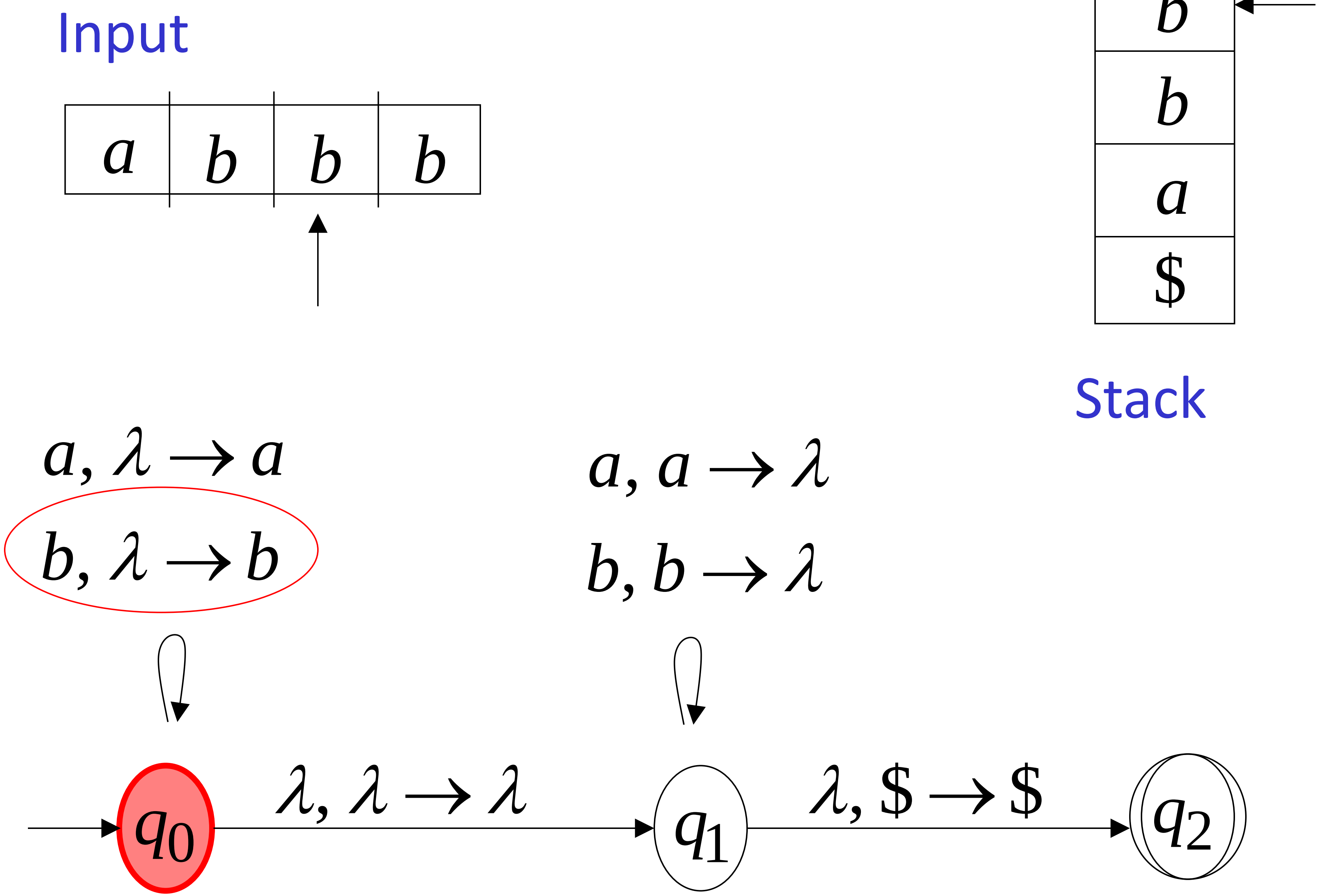
Time 1


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


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

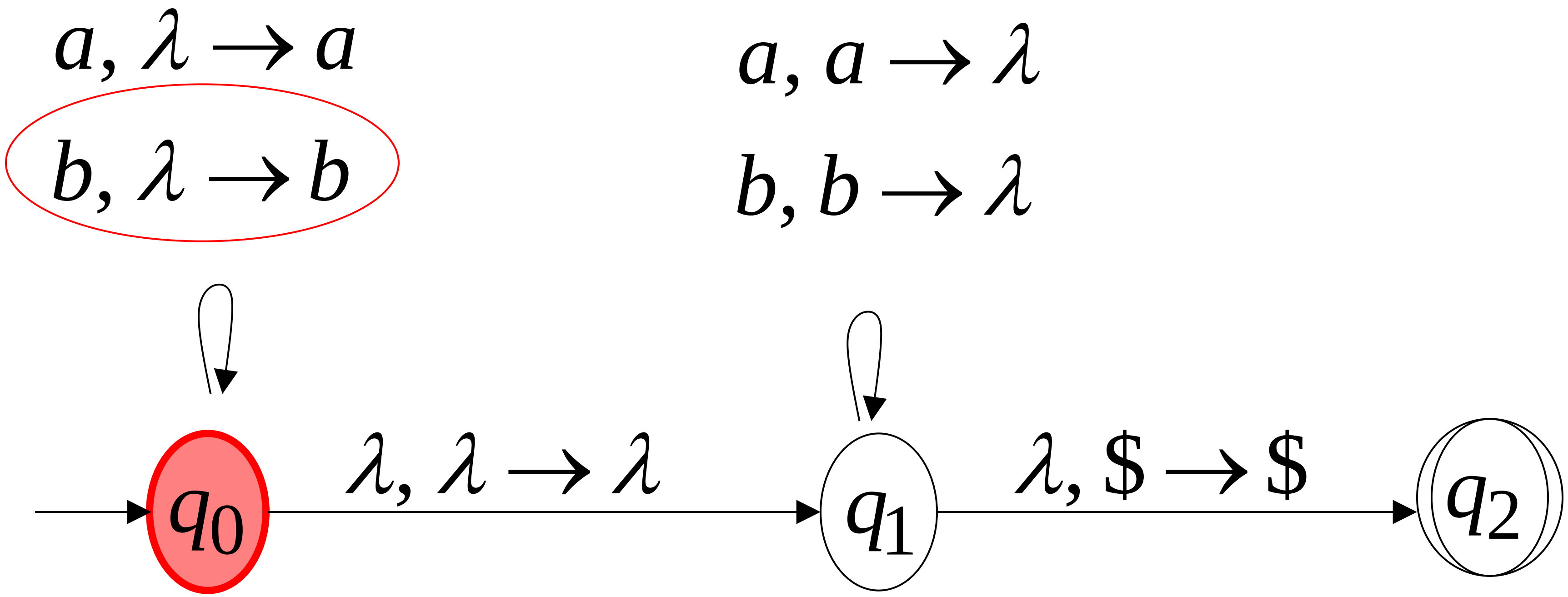
Time 4

Input

a	b	b	b

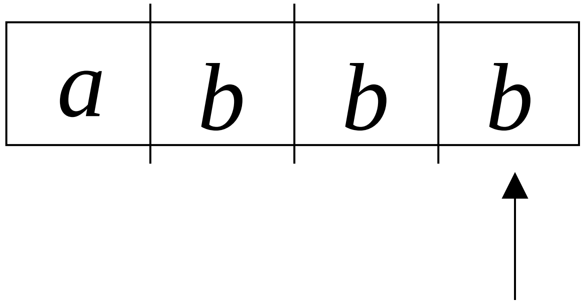
b
b
b
a
$\$$

Stack

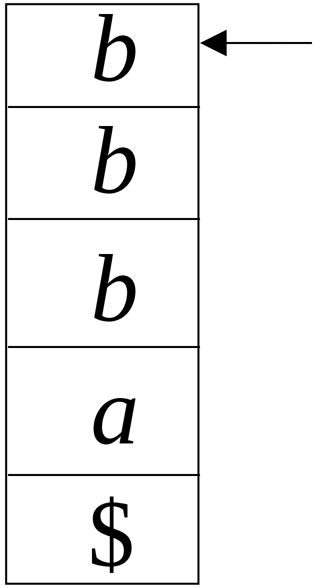


Time 5

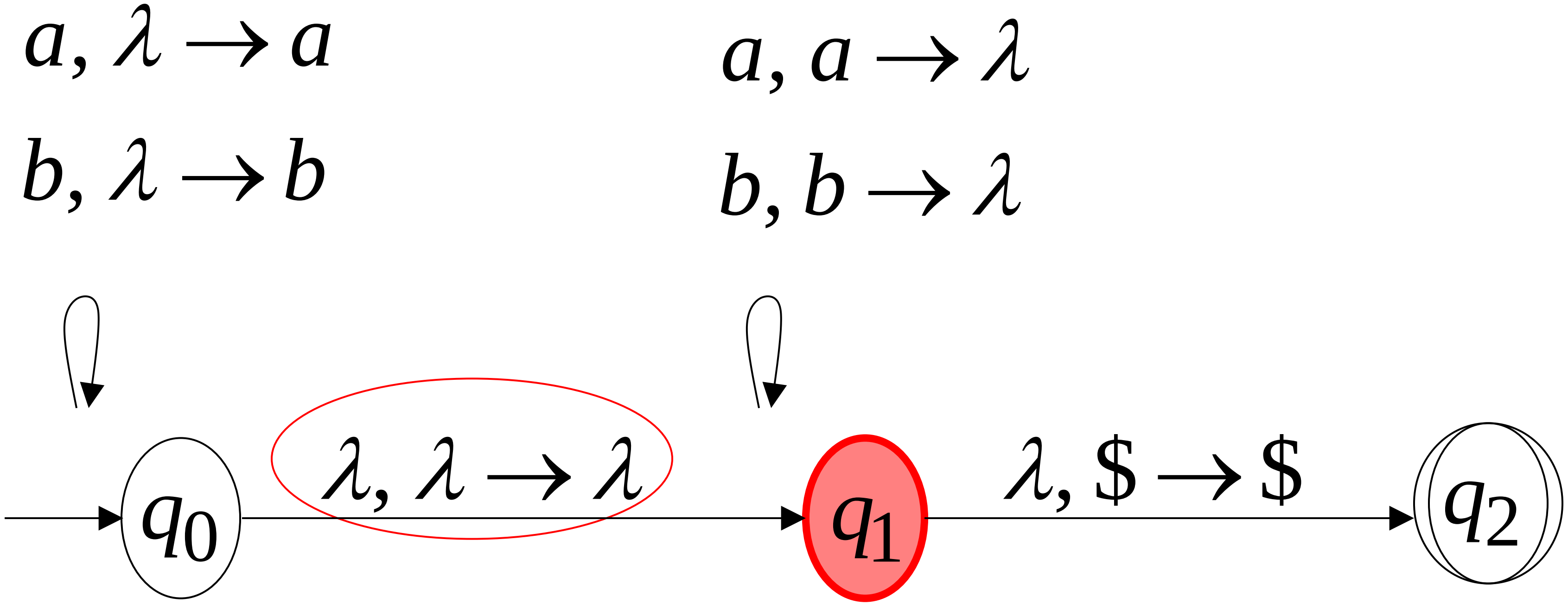
Input



No final state is reached



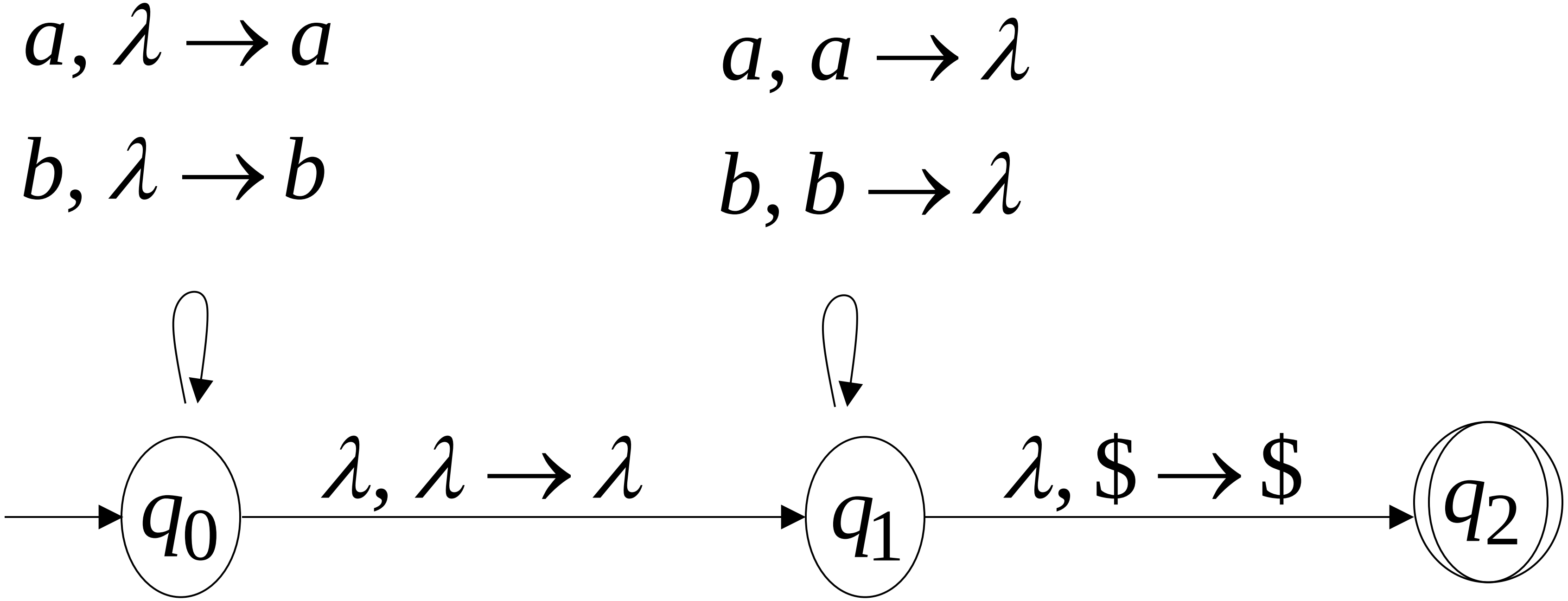
Stack



There is no computation

that accepts string $abbb$

$$abbb \notin L(M)$$



A string is rejected if there is
no computation such that:

All the input is consumed

AND

The last state is a final state

At the end of the computation,
we do not care about the stack contents

In other words, a string is rejected

if in every computation with this string:

The input cannot be consumed

OR

The input is consumed and the last state is not a final state

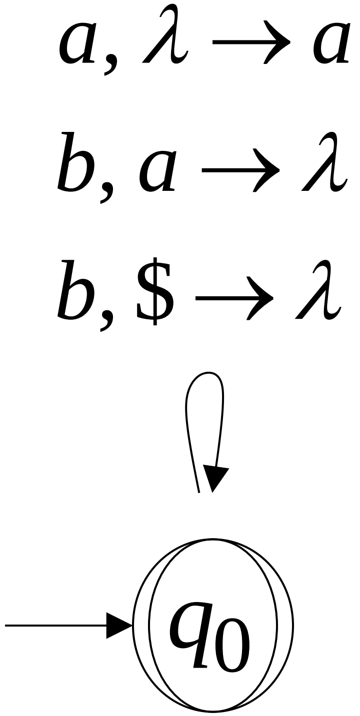
OR

The stack head moves below the bottom of the stack

Another NPDA example

NPDA M

$$L(M) = \{a^n b^m : n \geq m-1\}$$



Execution Example:

Time 0

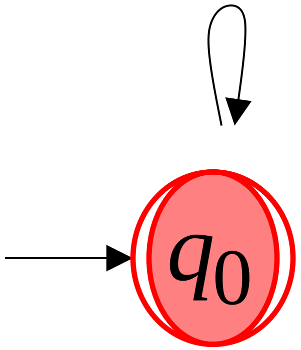
Input

a	a	b

$$a, \lambda \rightarrow a$$

$$b, a \rightarrow \lambda$$

$$b, \$ \rightarrow \lambda$$



$\$$

Stack

Time 1

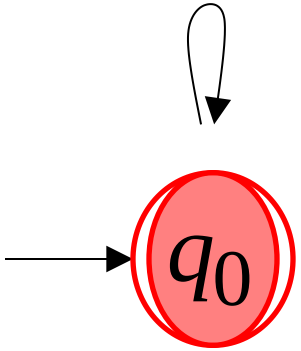
Input

a	a	b

$$a, \lambda \rightarrow a$$

$$b, a \rightarrow \lambda$$

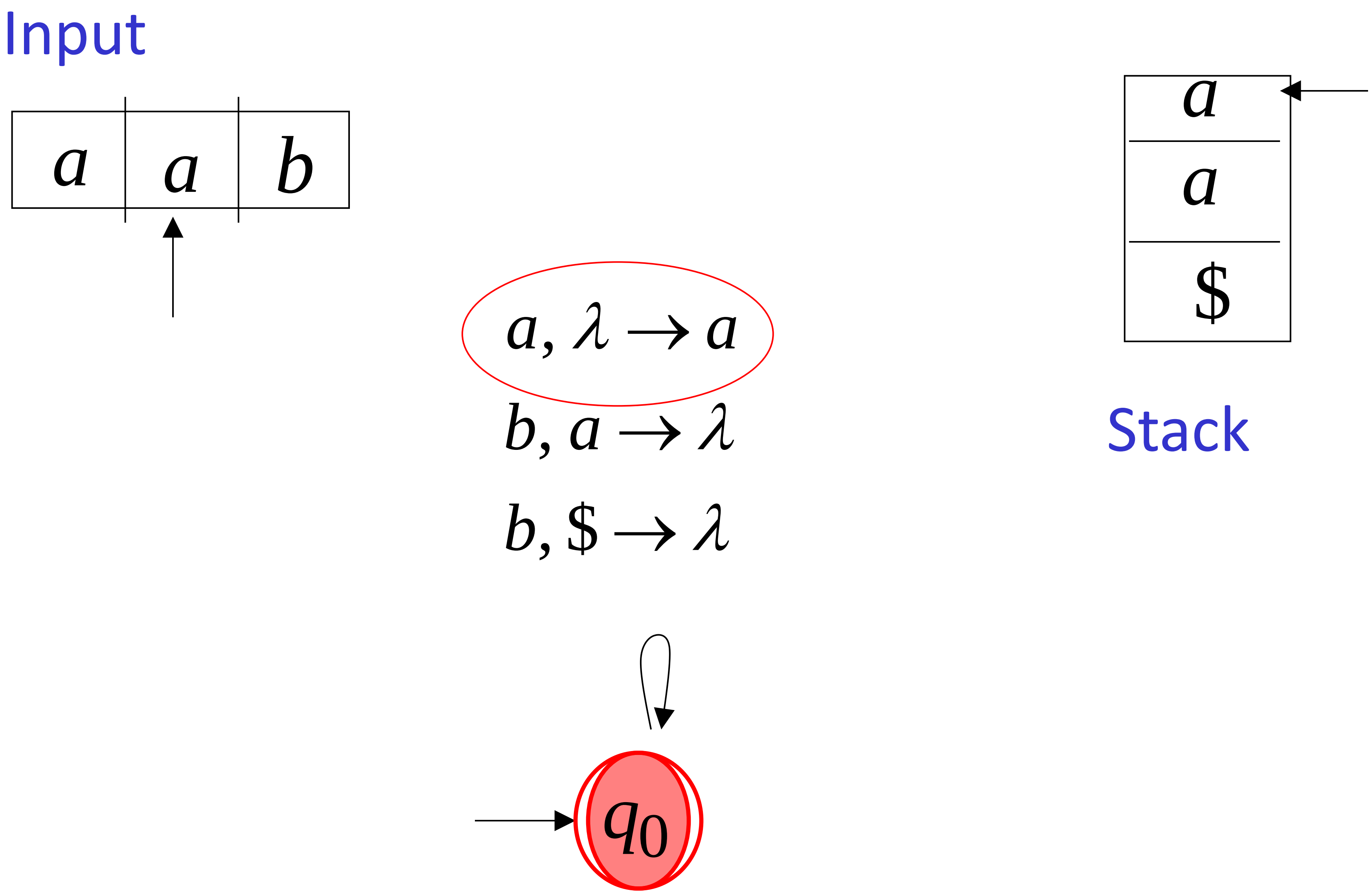
$$b, \$ \rightarrow \lambda$$



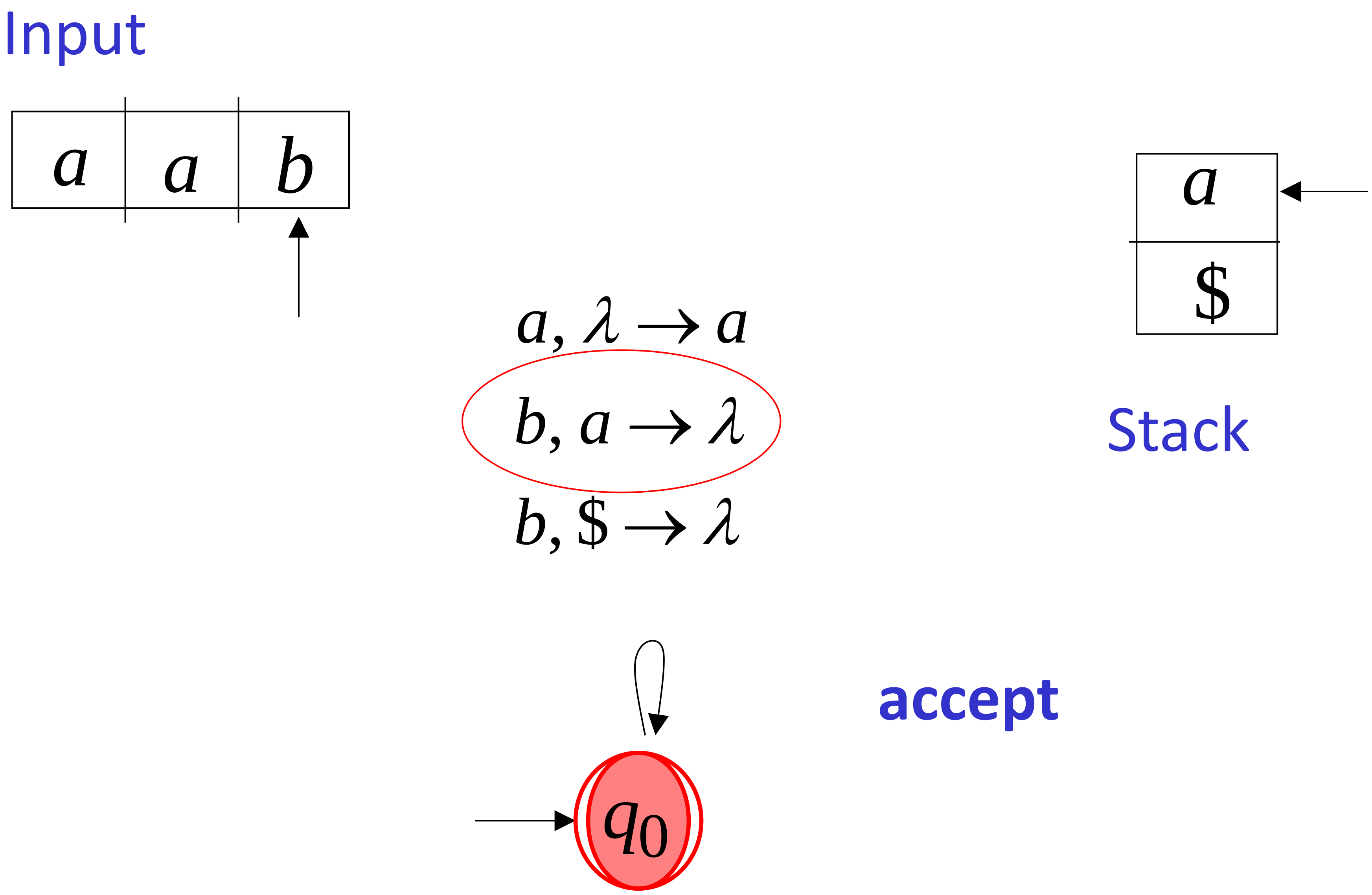
a
$\$$

Stack

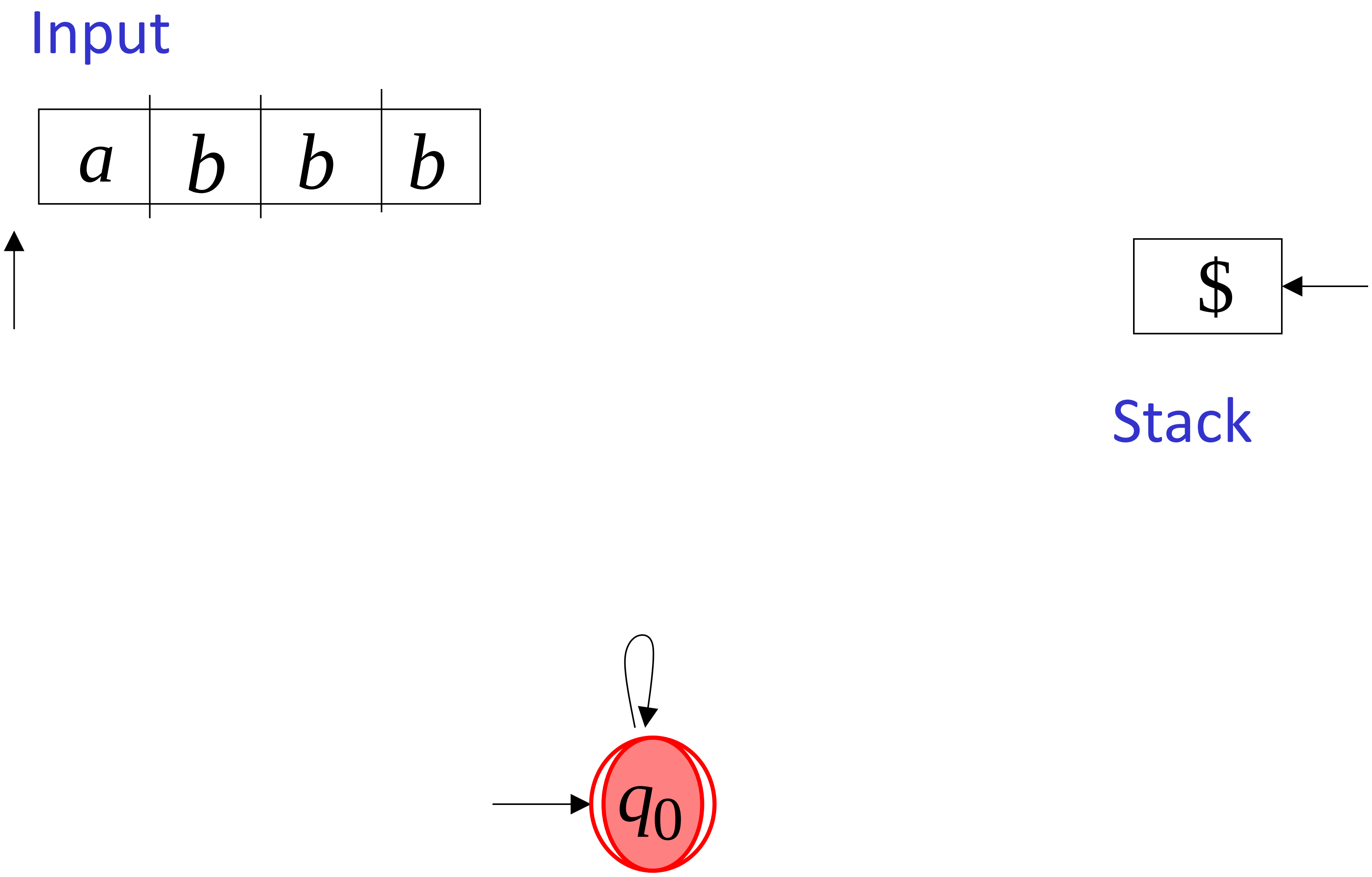
Time 2



Time 3



Rejection example: Time 0



Time 1

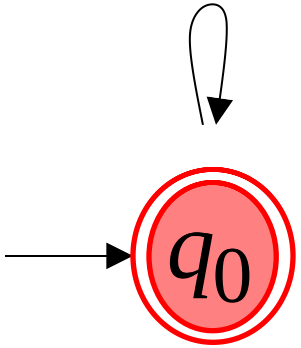
Input

a	b	b	b

$$a, \lambda \rightarrow a$$

$$b, a \rightarrow \lambda$$

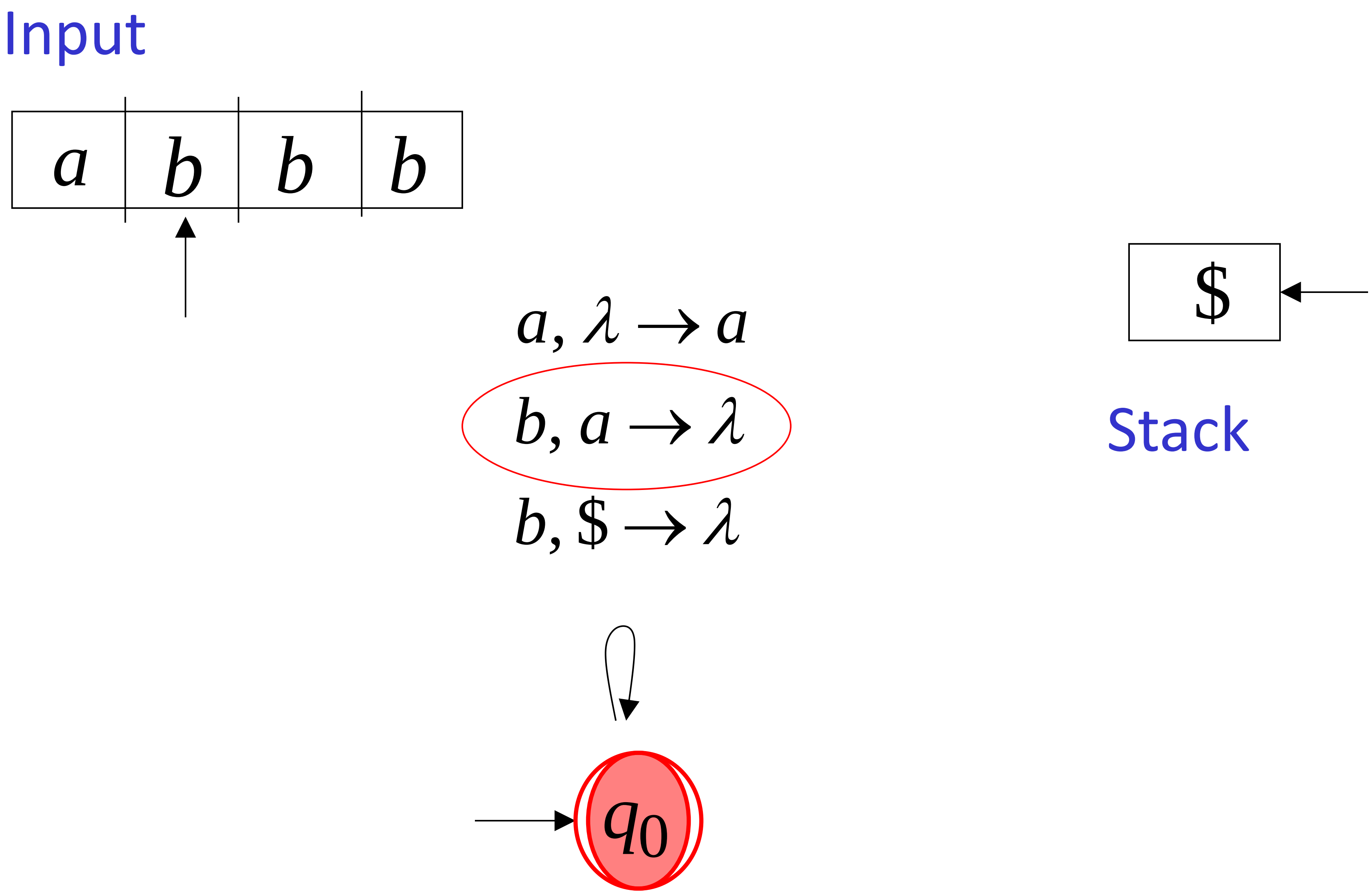
$$b, \$ \rightarrow \lambda$$



a
$\$$

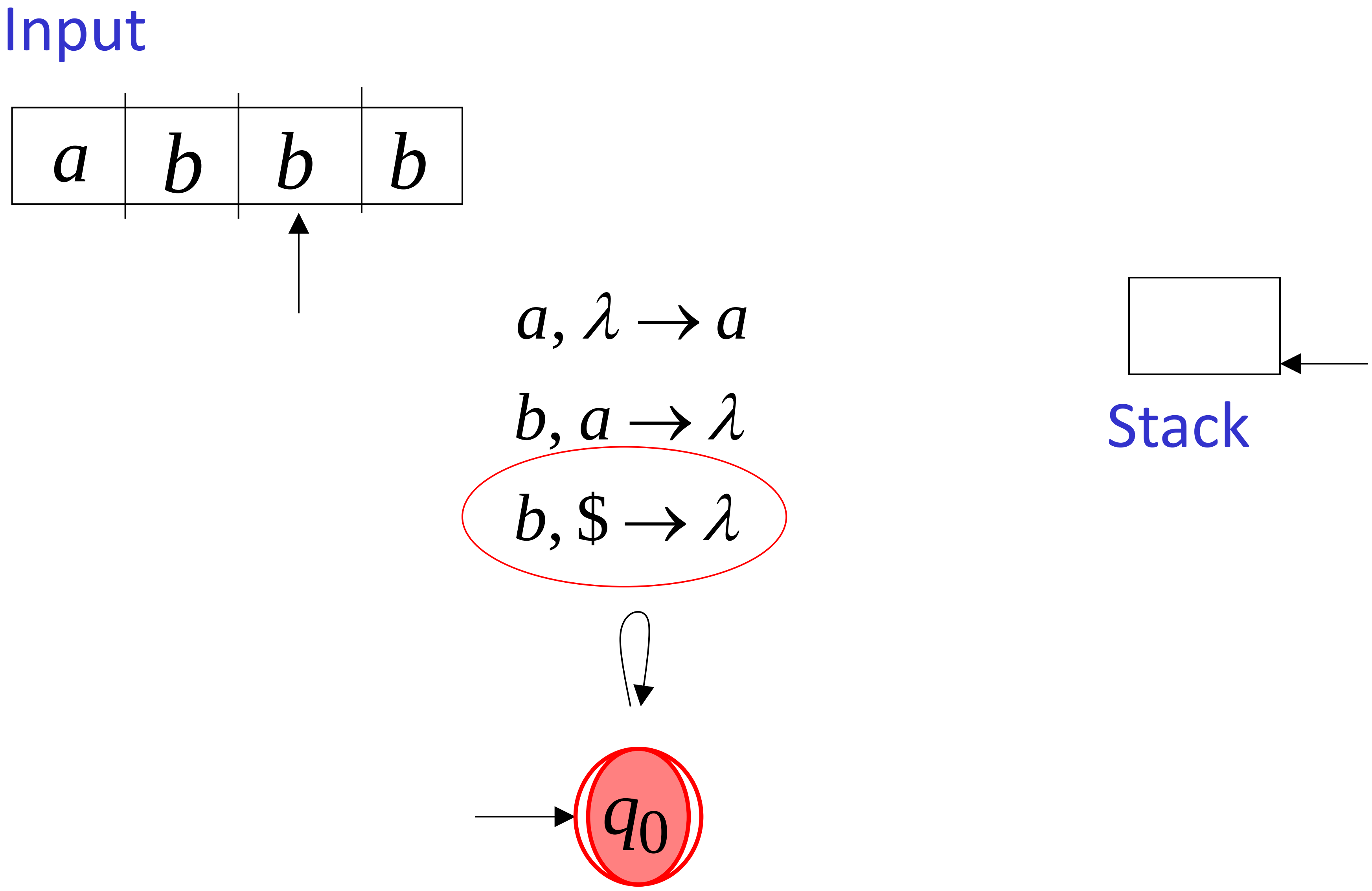
Stack

Time 2



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

Time 3



Time 4

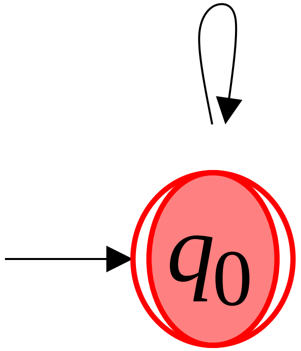
Input

a	b	b	b

$$a, \lambda \rightarrow a$$

$$b, a \rightarrow \lambda$$

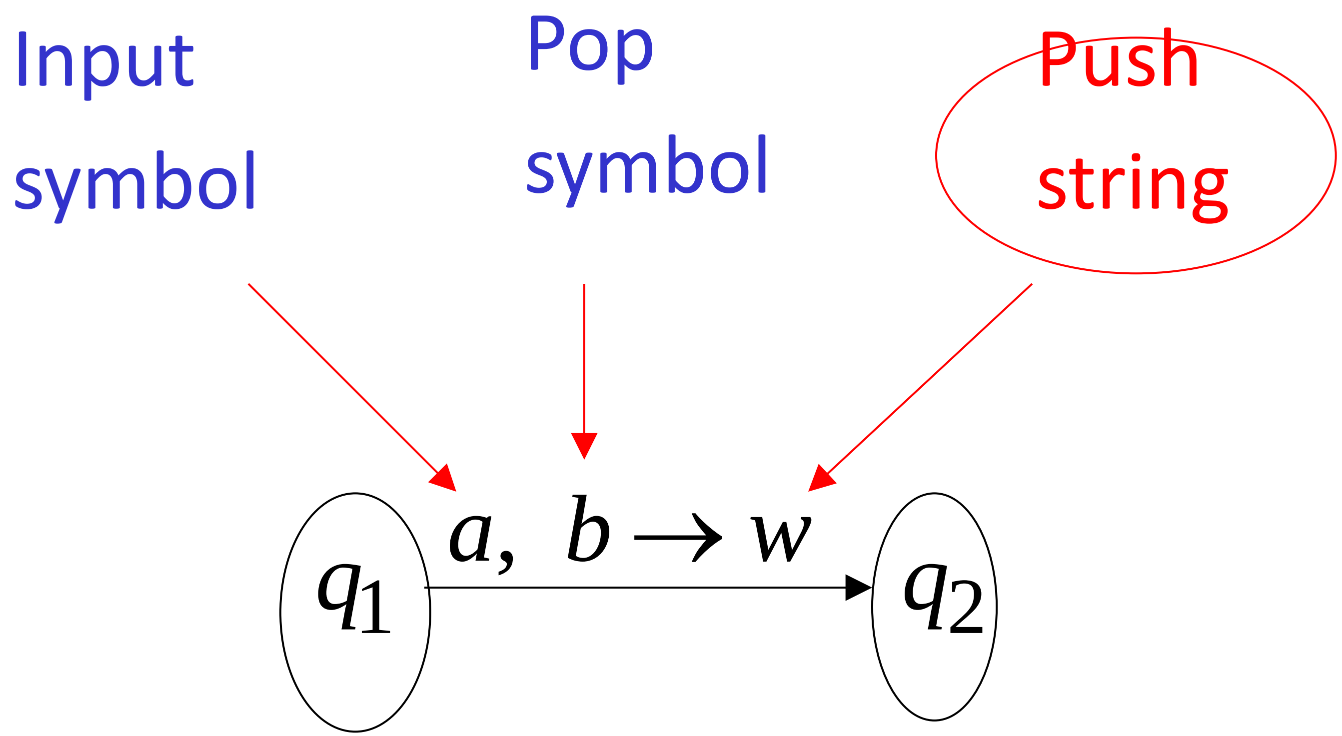
$$b, \$ \rightarrow \lambda$$



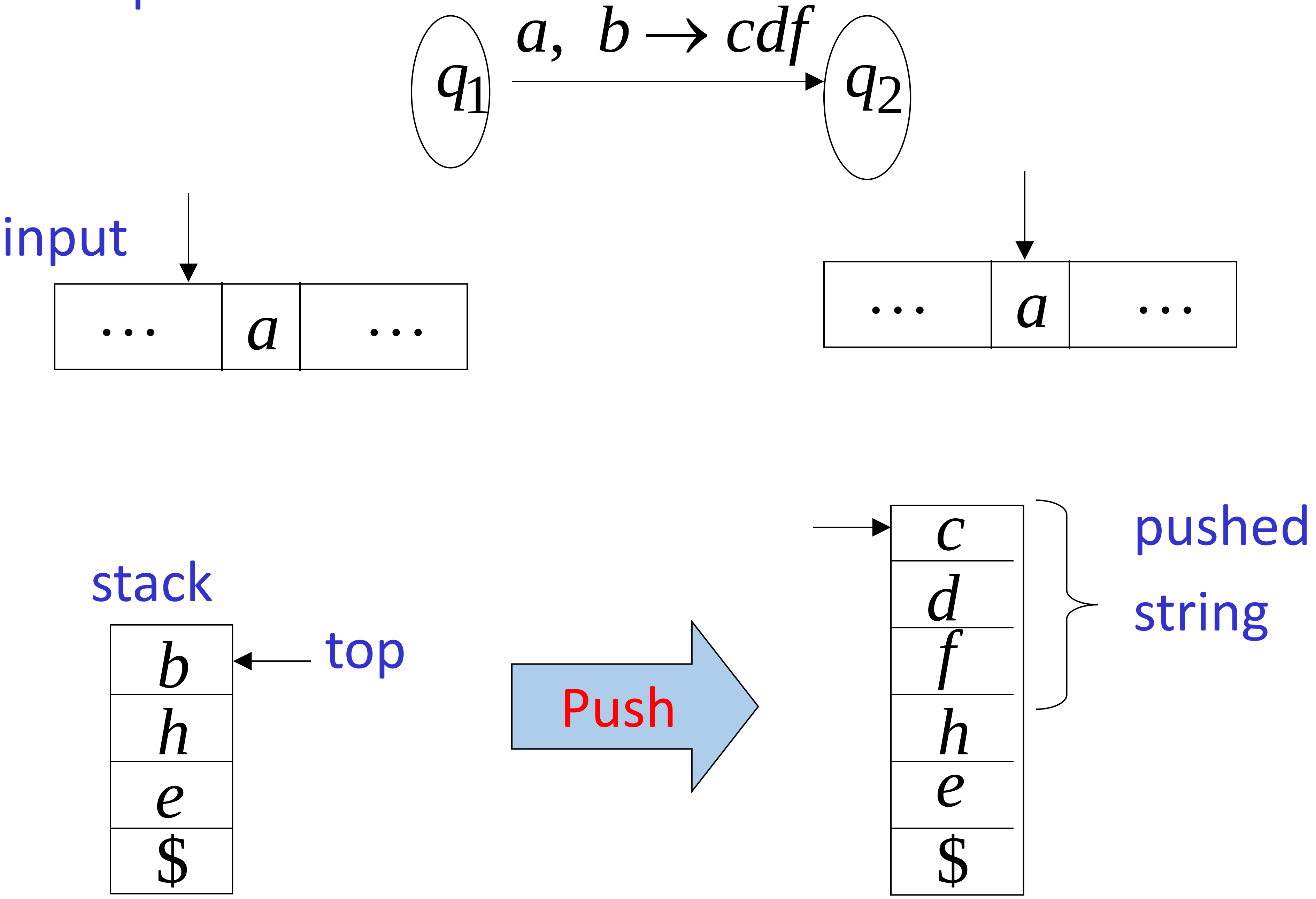
Stack

Halt and <u>Reject</u>

Pushing Strings



Example:

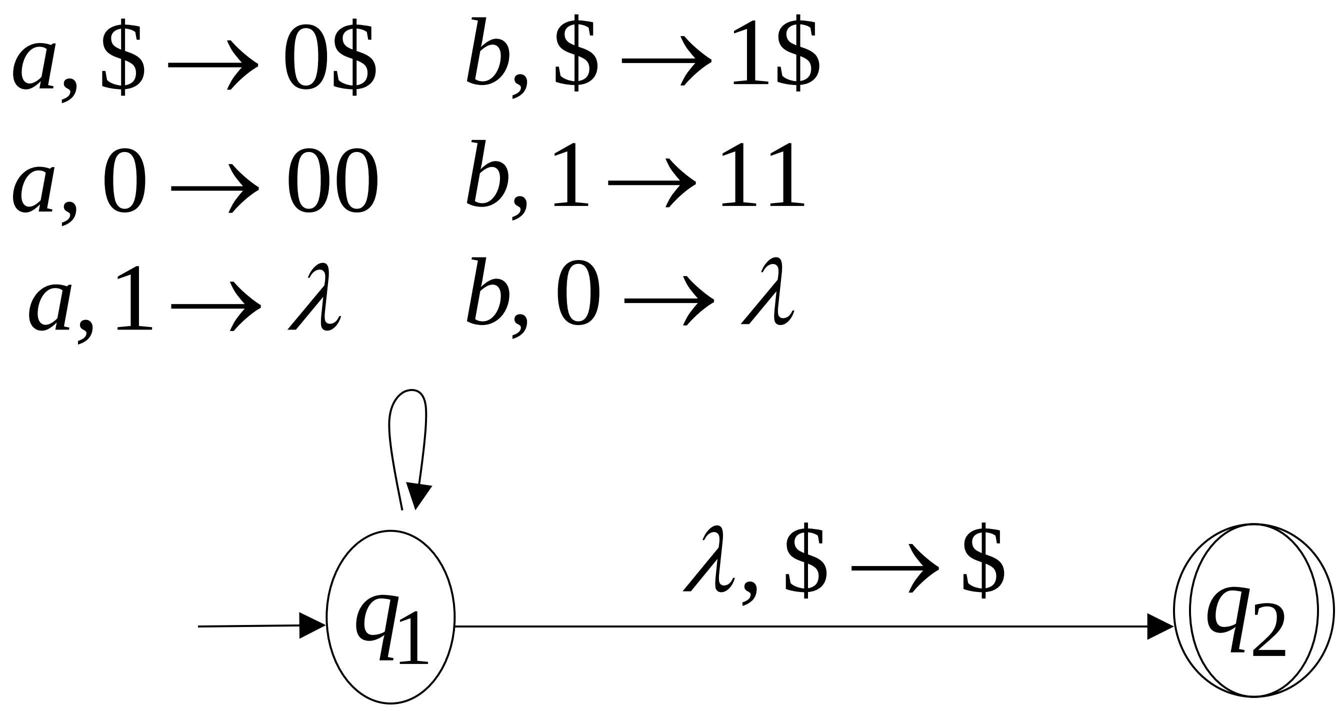


q_1
$a, \; b \rightarrow cdf$
q_2
input
$\cdots$ a $\cdots$
$\cdots$ a $\cdots$
stack
b
h
e
$\$$
top
Push
c
d
f
h
e
$\$$
pushed
string

Another NPDA example

NPDA M

$$L(M) = \{w: \ n_a = n_b\}$$



Execution Example: Time 0

Input

a	b	b	a	a	b

Stack

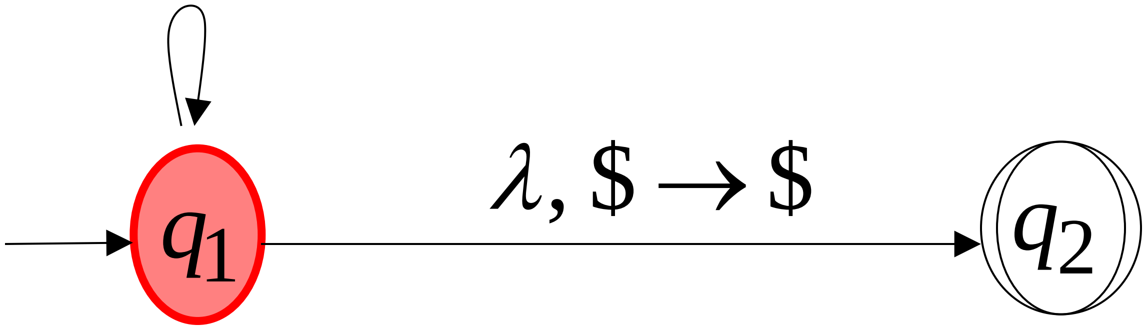
$\$$

$$a, \$ \rightarrow 0\$ \qquad b, \$ \rightarrow 1\$$$

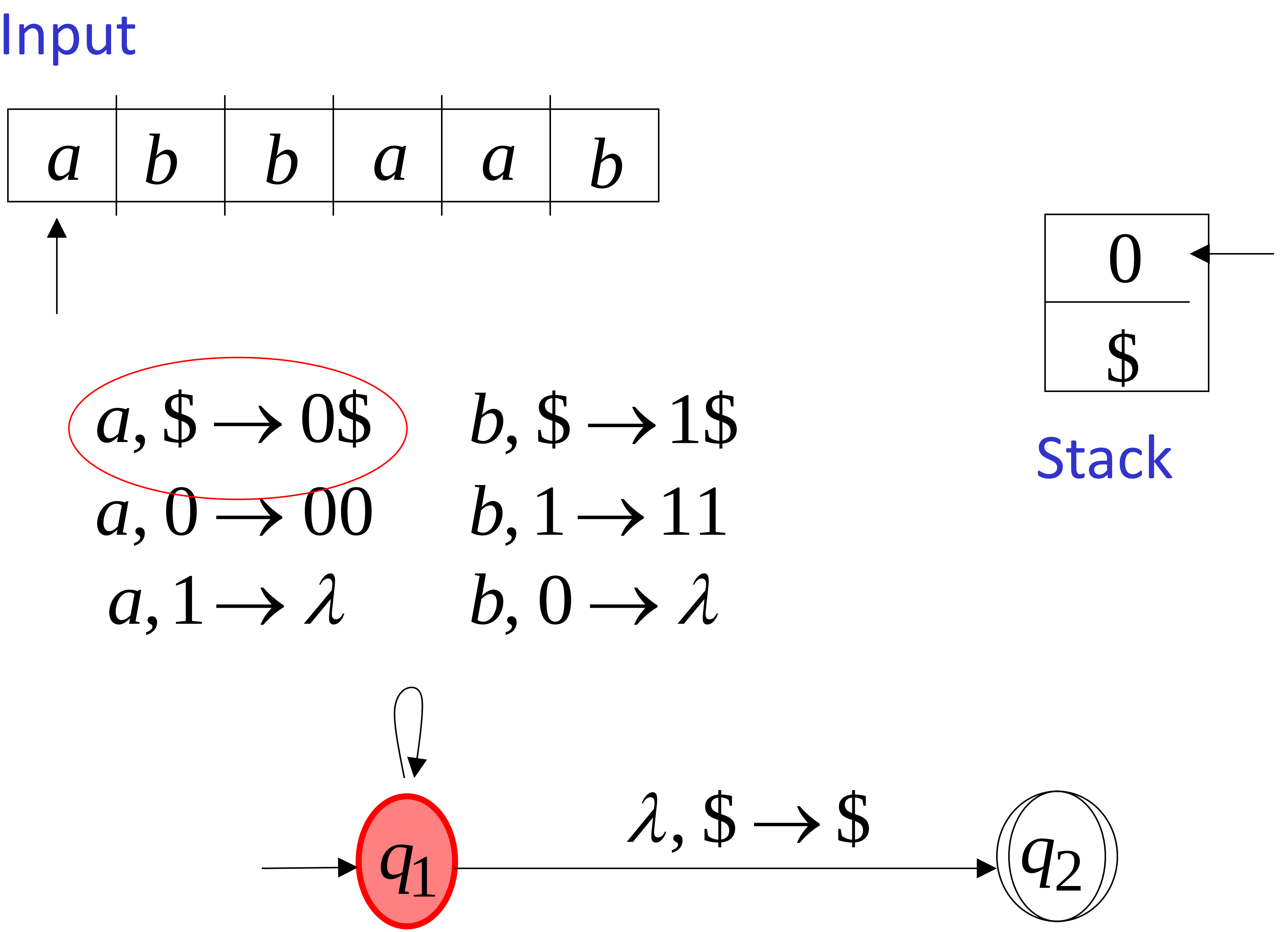
$$a, 0 \rightarrow 00 \qquad b, 1 \rightarrow 11$$

$$a, 1 \rightarrow \lambda \qquad b, 0 \rightarrow \lambda$$

current state

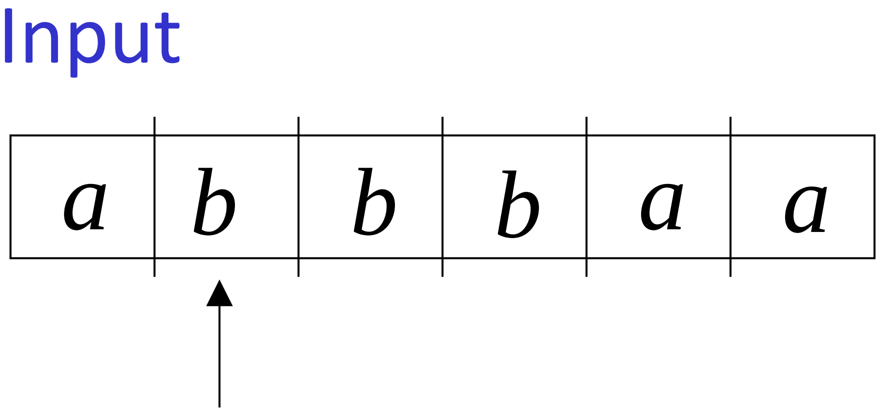


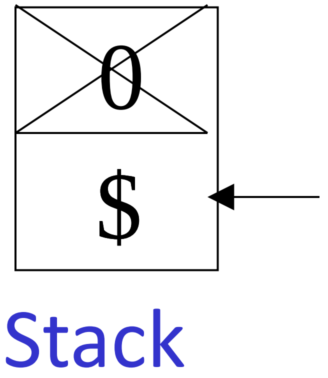
Time 1



Input
a b b a a b
0
$
Stack
$a, \$ \rightarrow 0\$$
$b, \$ \rightarrow 1\$$
$a, 0 \rightarrow 00$
$b, 1 \rightarrow 11$
$a, 1 \rightarrow \lambda$
$b, 0 \rightarrow \lambda$
$\lambda, \$ \rightarrow \$$
q_1
q_2

Time 3

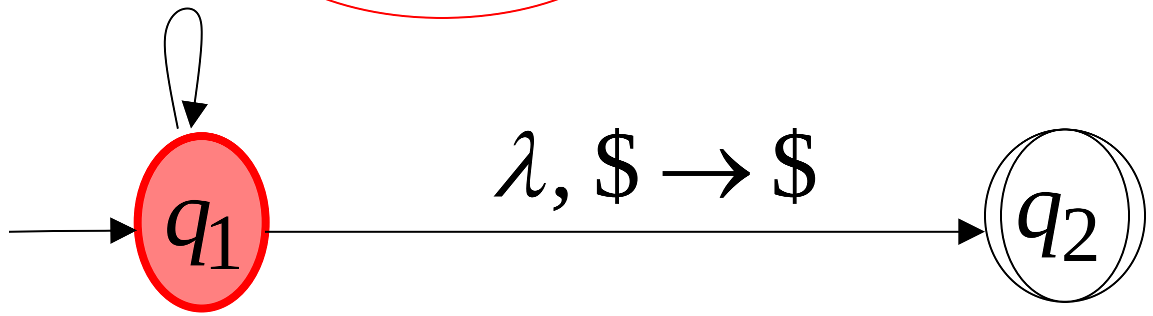




$$a, \$ \rightarrow 0\$ \qquad b, \$ \rightarrow 1\$$$

$$a, 0 \rightarrow 00 \qquad b, 1 \rightarrow 11$$

$$a, 1 \rightarrow \lambda \qquad b, 0 \rightarrow \lambda$$



Time 4

Input

a	b	b	b	a	a

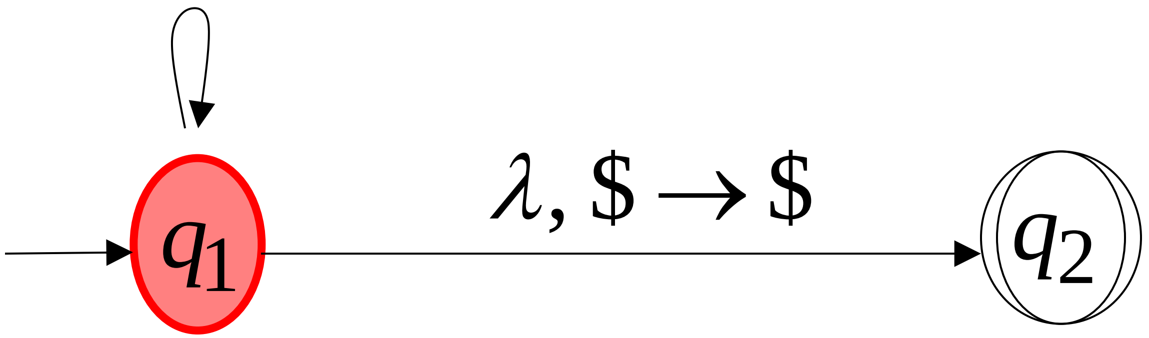
1
$\$$

Stack

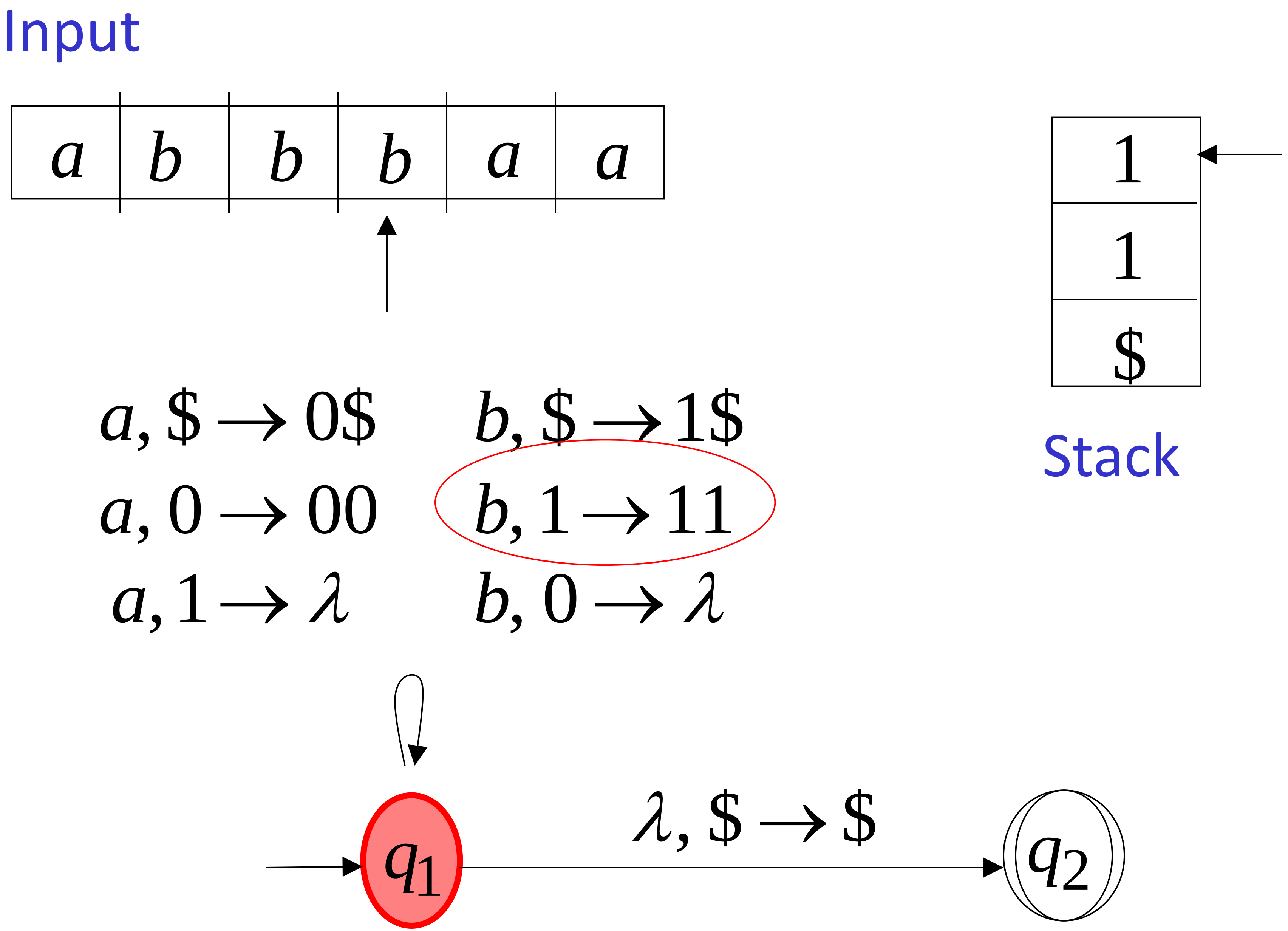
$$a, \$ \rightarrow 0\$ \qquad b, \$ \rightarrow 1\$$$

$$a, 0 \rightarrow 00 \qquad b, 1 \rightarrow 11$$

$$a, 1 \rightarrow \lambda \qquad b, 0 \rightarrow \lambda$$

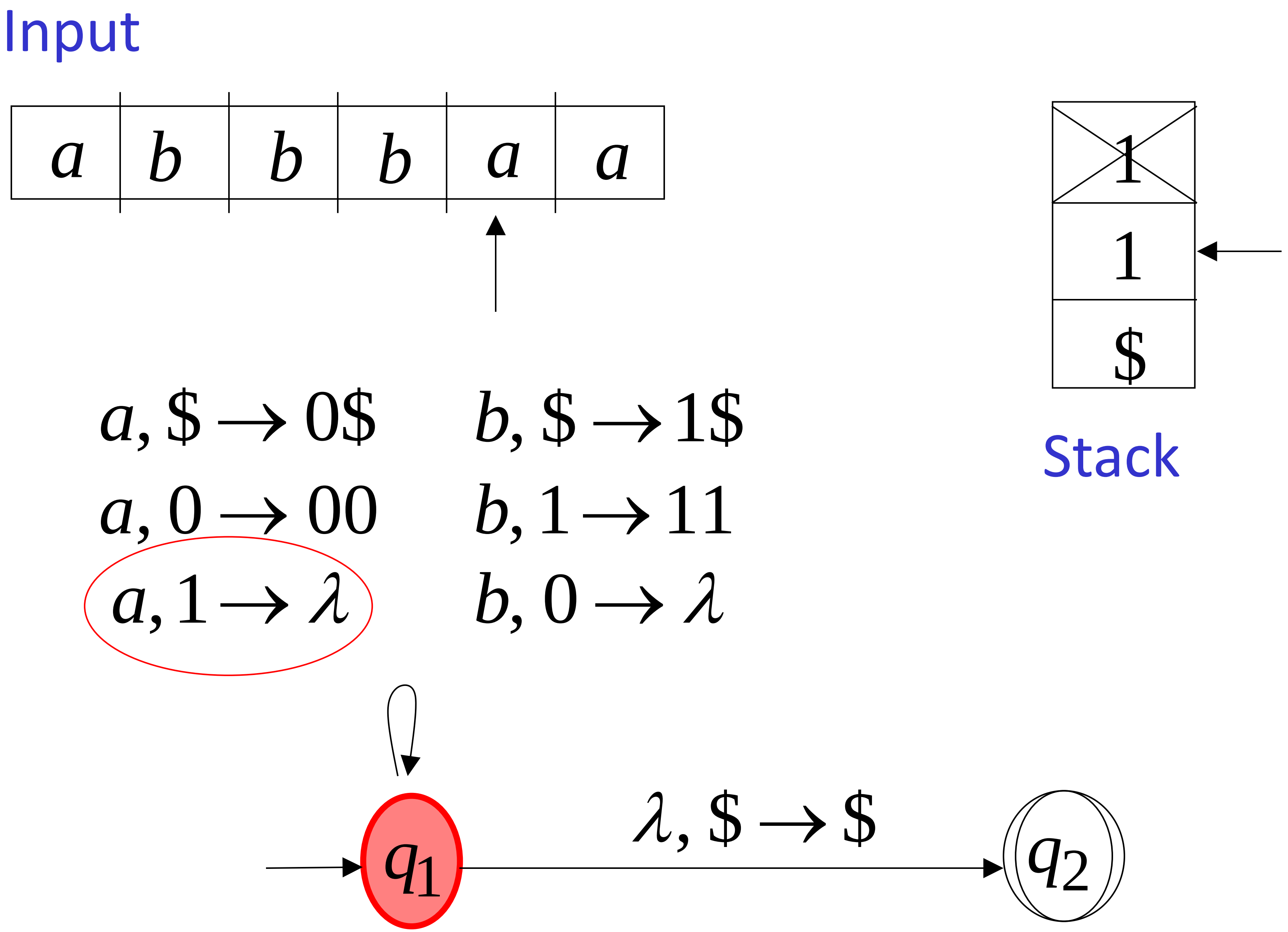


Time 5



Input
a b b b a a
1
1
$
Stack
$a, \$ \to 0\$$
$a, 0 \to 00$
$a, 1 \to \lambda$
$b, \$ \to 1\$$
$b, 1 \to 11$
$b, 0 \to \lambda$
$\lambda, \$ \to \$$
q_1
q_2

Time 6


Input
a b b b a a
1
1
$
Stack
$a, \$ \rightarrow 0\$$
$b, \$ \rightarrow 1\$$
$a, 0 \rightarrow 00$
$b, 1 \rightarrow 11$
$a, 1 \rightarrow \lambda$
$b, 0 \rightarrow \lambda$
q_1
$\lambda, \$ \rightarrow \$$
q_2

Time 7

Input

a	b	b	b	a	a

$\cancel{1}$

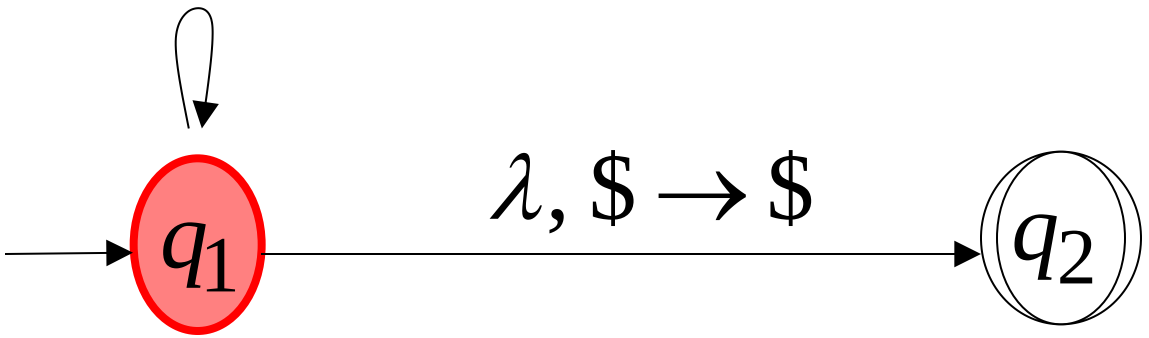
$\$$

Stack

$a, \$ \rightarrow 0\$$ $\quad$ $b, \$ \rightarrow 1\$$

$a, 0 \rightarrow 00$ $\quad$ $b, 1 \rightarrow 11$

$a, 1 \rightarrow \lambda$ $\quad$ $b, 0 \rightarrow \lambda$



Time 8

Input

a	b	b	b	a	a

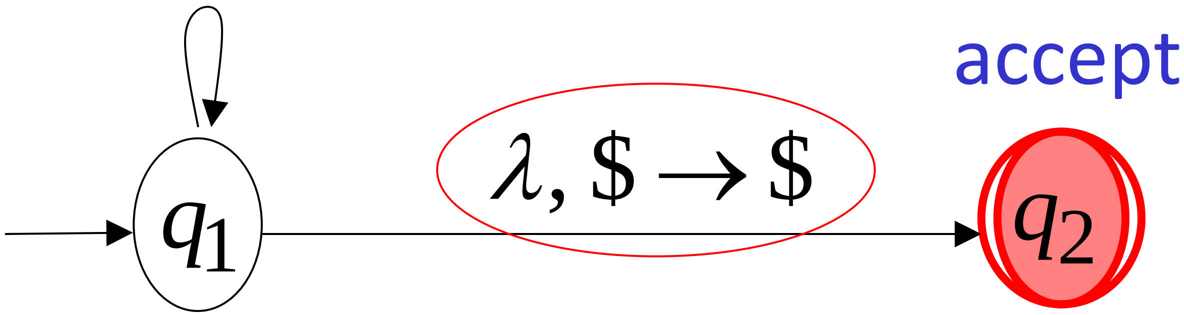
$$a, \$ \rightarrow 0\$ \qquad b, \$ \rightarrow 1\$$$

$$a, 0 \rightarrow 00 \qquad b, 1 \rightarrow 11$$

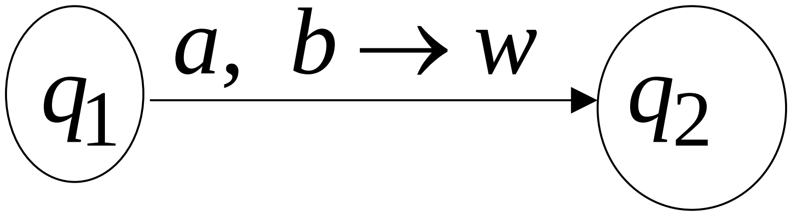
$$a, 1 \rightarrow \lambda \qquad b, 0 \rightarrow \lambda$$

$\$$

Stack

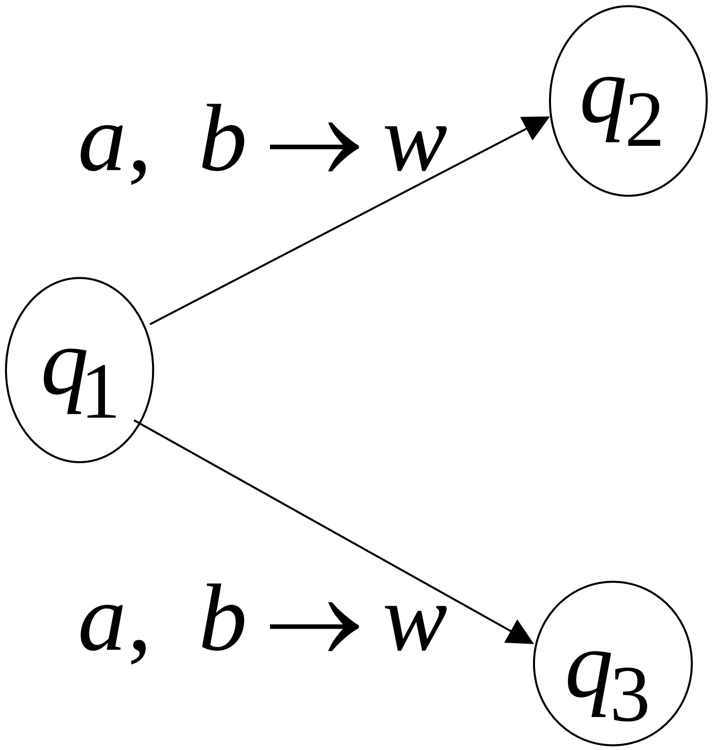


Formalities for NPDAs



Transition function:

$$\delta(q_1, a, b) = \{(q_2, w)\}$$



Transition function:

$$\delta(q_1, a, b) = \{(q_2, w), \ (q_3, w)\}$$

Formal Definition

Non-Deterministic Pushdown Automaton

NPDA

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

States

Input alphabet

Stack alphabet

Transition function

Initial state

Stack start symbol

Final states

Instantaneous Description

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

Current state

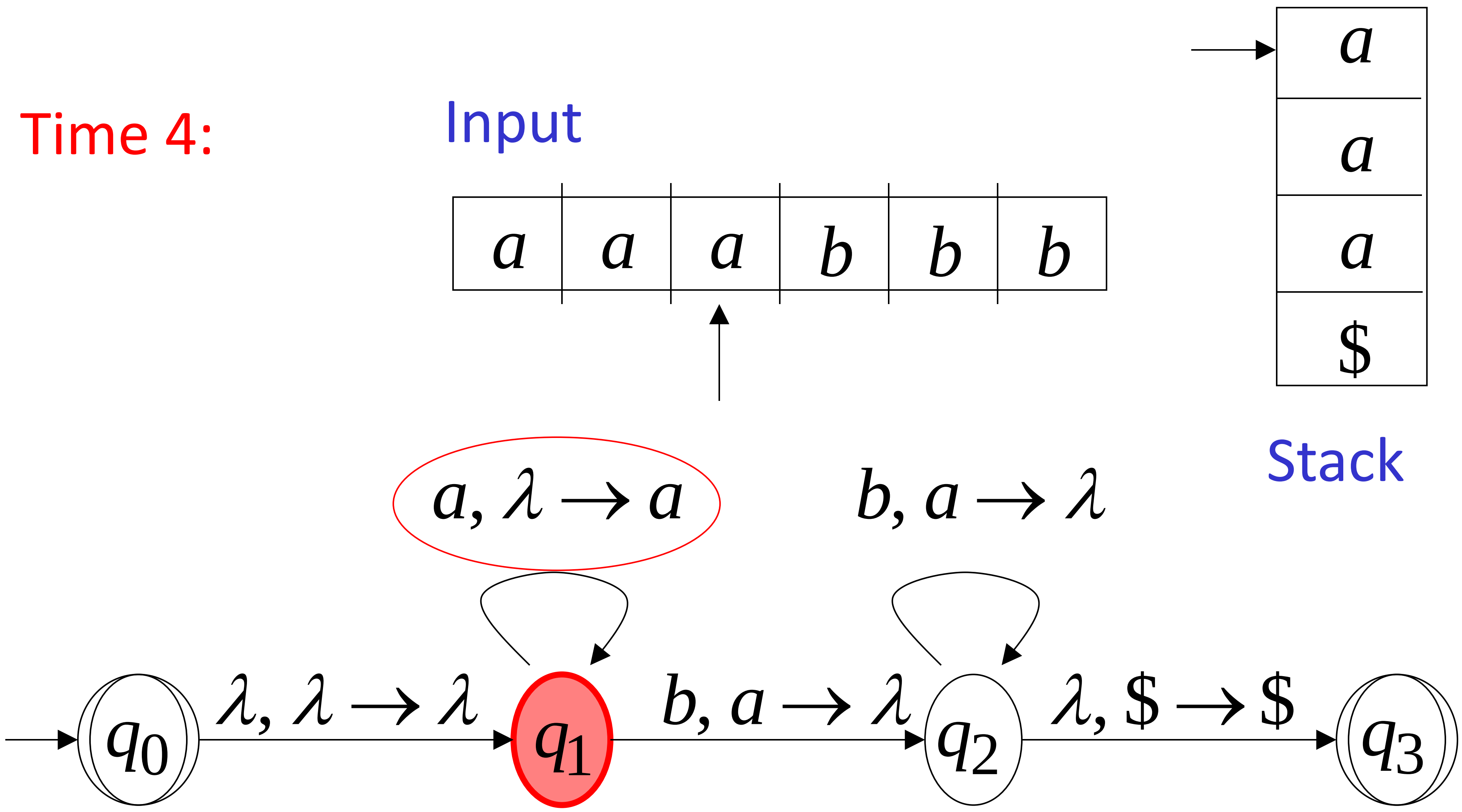
Remaining input

Current stack contents

Example: **Instantaneous Description**

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

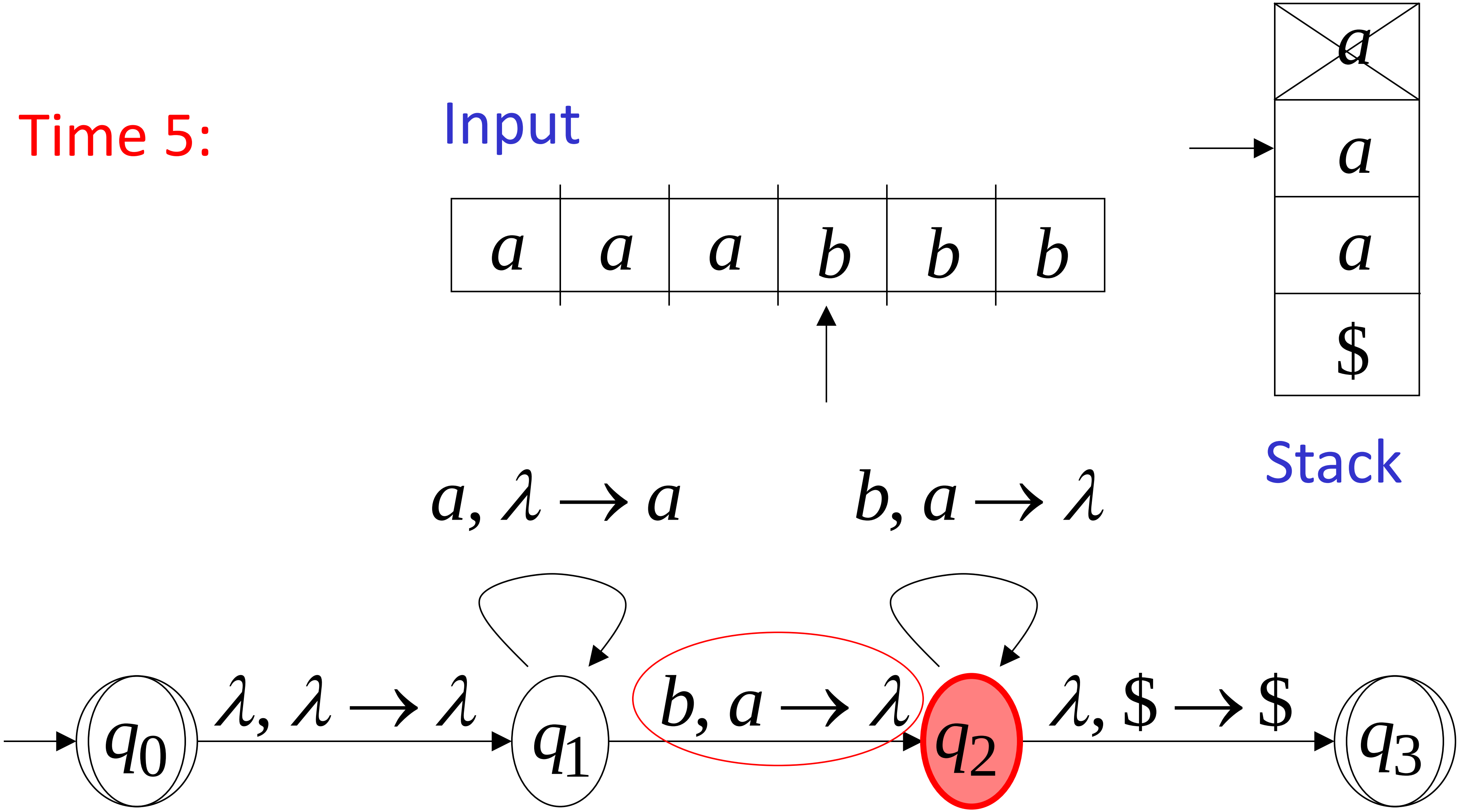
Time 4:



Example: **Instantaneous Description**

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

Time 5:



We write:

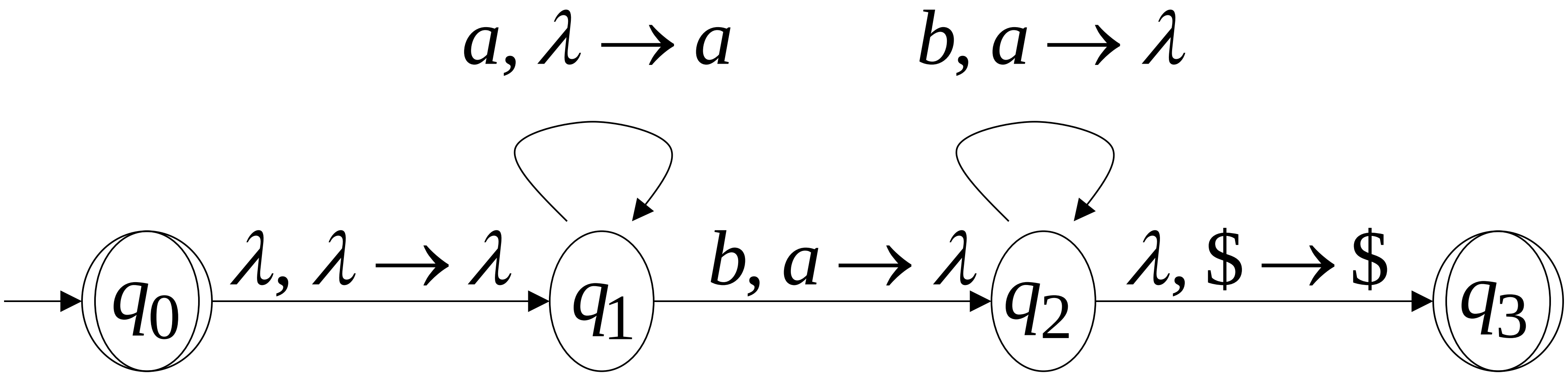
$$(q_1, bbb, aaa\$) \succ (q_2, bb, aa\$)$$

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:

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

Formal Definition

Language $L(M)$ of NPDA : M

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

Initial state

Final state

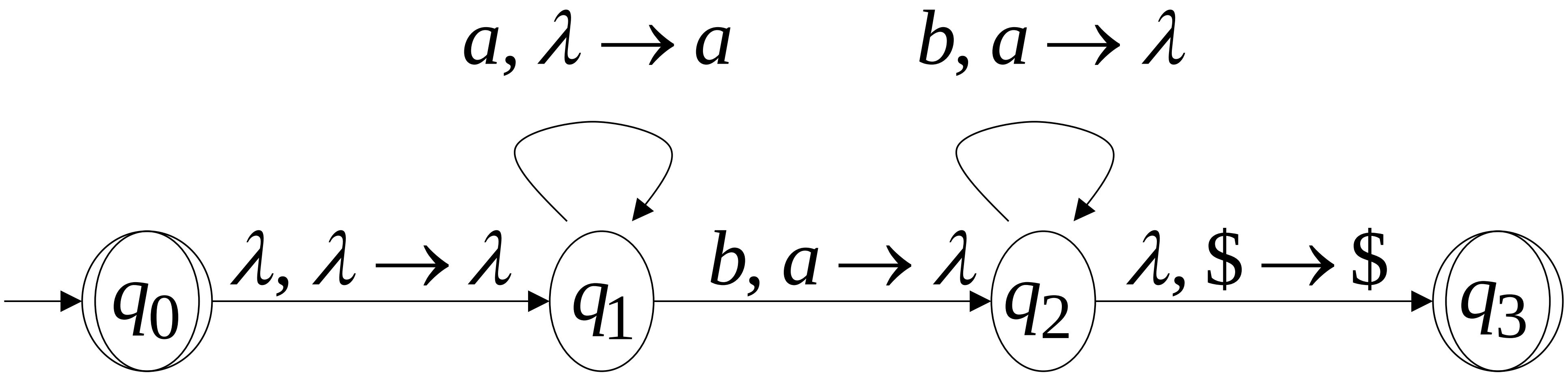
Example:

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

$$\Downarrow$$

$$aaabbb \in L(M)$$

NPDA M:

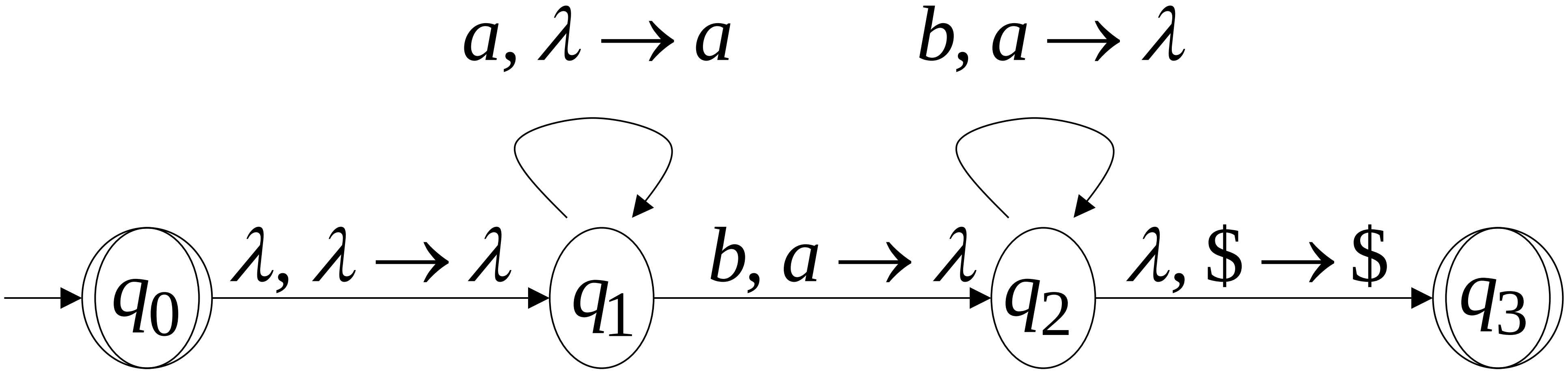


$$(q_0, a^n b^n, \$) \overset{*}{\succ} (q_3, \lambda, \$)$$

$$\Downarrow$$

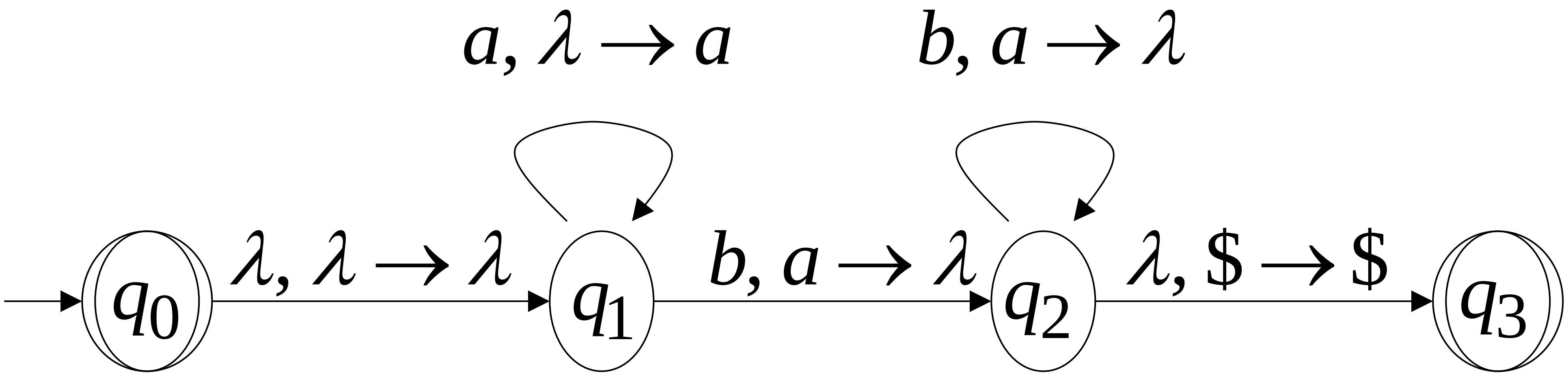
$$a^n b^n \in L(M)$$

NPDA M:



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

NPDA M:



Outline

- Last week
- Pushdown automata
- Properties of Context-free Languages

Positive Properties
of
Context-Free languages

Union

Context-free languages

are closed under: **Union**

L_1 is context free

L_2 is context free

$\Rightarrow$ $L_1 \cup L_2$ is context-free

Example

Language

Grammar

$$L_1 = \{a^n b^n\}$$

$$S_1 \to aS_1b \mid \lambda$$

$$L_2 = \{ww^R\}$$

$$S_2 \to aS_2a \mid bS_2b \mid \lambda$$

Union

$$L = \{a^n b^n\} \cup \{ww^R\}$$

$$S \to S_1 \mid S_2$$

In general:

For context-free languages $L_1,\ L_2$

with context-free grammars $G_1,\ G_2$

and start variables $S_1,\ S_2$

The grammar of the **union** $L_1 \cup L_2$

has new start variable S

and additional production $S \to S_1 \mid S_2$

Concatenation

Context-free languages
are closed under: **Concatenation**

L_1 is context free

L_2 is context free

$\Longrightarrow$ $L_1 L_2$ is context-free

Example

Language | **Grammar**

$$L_1 = \{a^n b^n\}$$

$$S_1 \rightarrow aS_1b \mid \lambda$$

$$L_2 = \{ww^R\}$$

$$S_2 \rightarrow aS_2a \mid bS_2b \mid \lambda$$

Concatenation

$$L = \{a^n b^n\}\{ww^R\}$$

$$S \rightarrow S_1S_2$$

In general:

For context-free languages $L_1,\ L_2$

with context-free grammars $G_1,\ G_2$

and start variables $S_1,\ S_2$

The grammar of the **concatenation** L_1L_2

has new start variable S

and additional production $S \to S_1S_2$

Star Operation

Context-free languages
are closed under: **Star-operation**

L is context free $\Rightarrow$ L^* is context-free

Example

Language

Grammar

$$L = \{a^n b^n\}$$

$$S \rightarrow aSb \mid \lambda$$

Star Operation

$$L = \{a^n b^n\}^*$$

$$S_1 \rightarrow SS_1 \mid \lambda$$

In general:

For context-free language L

with context-free grammar G

and start variable S

The grammar of the **star operation** L^*

has new start variable S_1

and additional production $S_1 \rightarrow SS_1 \mid \lambda$

Negative Properties
of
Context-Free Languages

Intersection

Context-free languages
are **not** closed under: **intersection**

L_1 is context free

L_2 is context free

$\Rightarrow$ $L_1 \cap L_2$

not necessarily
context-free

Example

$$L_1 = \{a^n b^n c^m\}$$

Context-free:

$$S \to AC$$

$$A \to aAb \mid \lambda$$

$$C \to cC \mid \lambda$$

$$L_2 = \{a^n b^m c^m\}$$

Context-free:

$$S \to AB$$

$$A \to aA \mid \lambda$$

$$B \to bBc \mid \lambda$$

Intersection

$$L_1 \cap L_2 = \{a^n b^n c^n\}$$ **NOT** context-free

Complement

Context-free languages

are **<u>not</u>** closed under: **complement**

L is context free $\Rightarrow$ $\overline{L}$ **<u>not</u>** necessarily context-free

Example

$$L_1 = \{a^n b^n c^m\}$$

Context-free:

$$S \to AC$$

$$A \to aAb \mid \lambda$$

$$C \to cC \mid \lambda$$

$$L_2 = \{a^n b^m c^m\}$$

Context-free:

$$S \to AB$$

$$A \to aA \mid \lambda$$

$$B \to bBc \mid \lambda$$

Complement

$$\overline{\overline{L_1} \cup \overline{L_2}} = L_1 \cap L_2 = \{a^n b^n c^n\}$$

NOT context-free

Intersection
of
Context-free languages
and
Regular Languages

The intersection of

a context-free language and

a regular language

is a context-free language

L_1 context free

L_2 regular

$$\Rightarrow$$

$L_1 \cap L_2$

context-free

Machine M_1

Machine M_2

NPDA for L_1

context-free

DFA for L_2

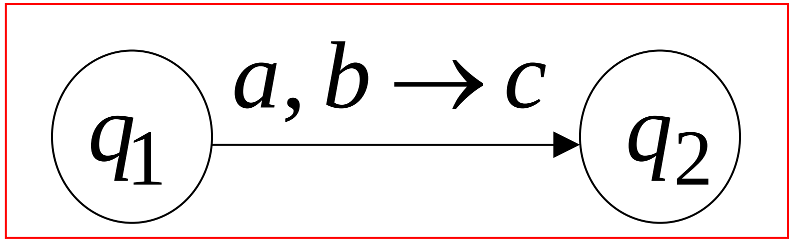
regular

Construct a new NPDA machine M

that accepts $L_1 \cap L_2$

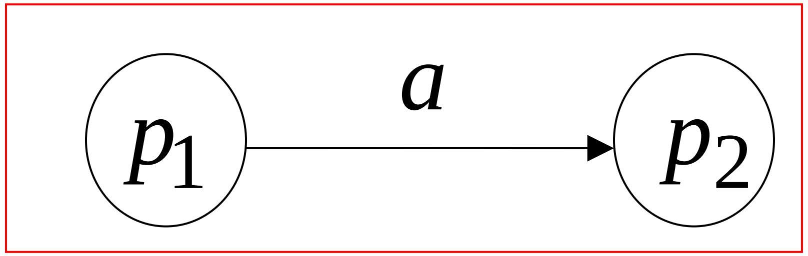
M simulates in parallel M_1 and M_2

NPDA M_1



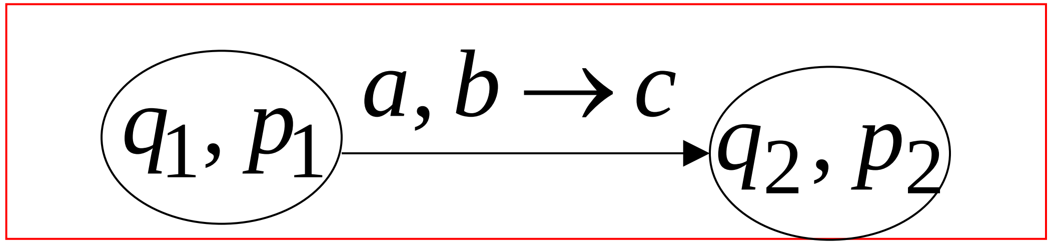
transition

DFA M_2



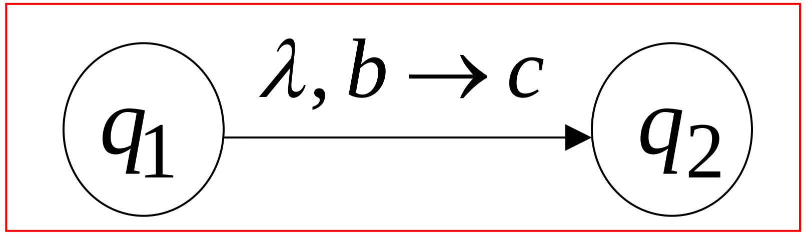
transition

NPDA M



transition

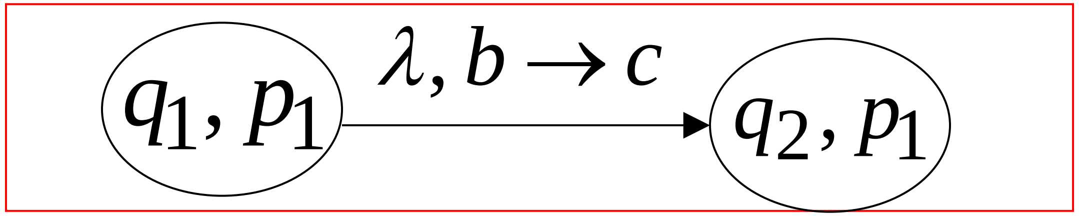
NPDA M_1



transition

DFA M_2

NPDA M



transition

NPDA M_1

DFA M_2

q_0

initial state

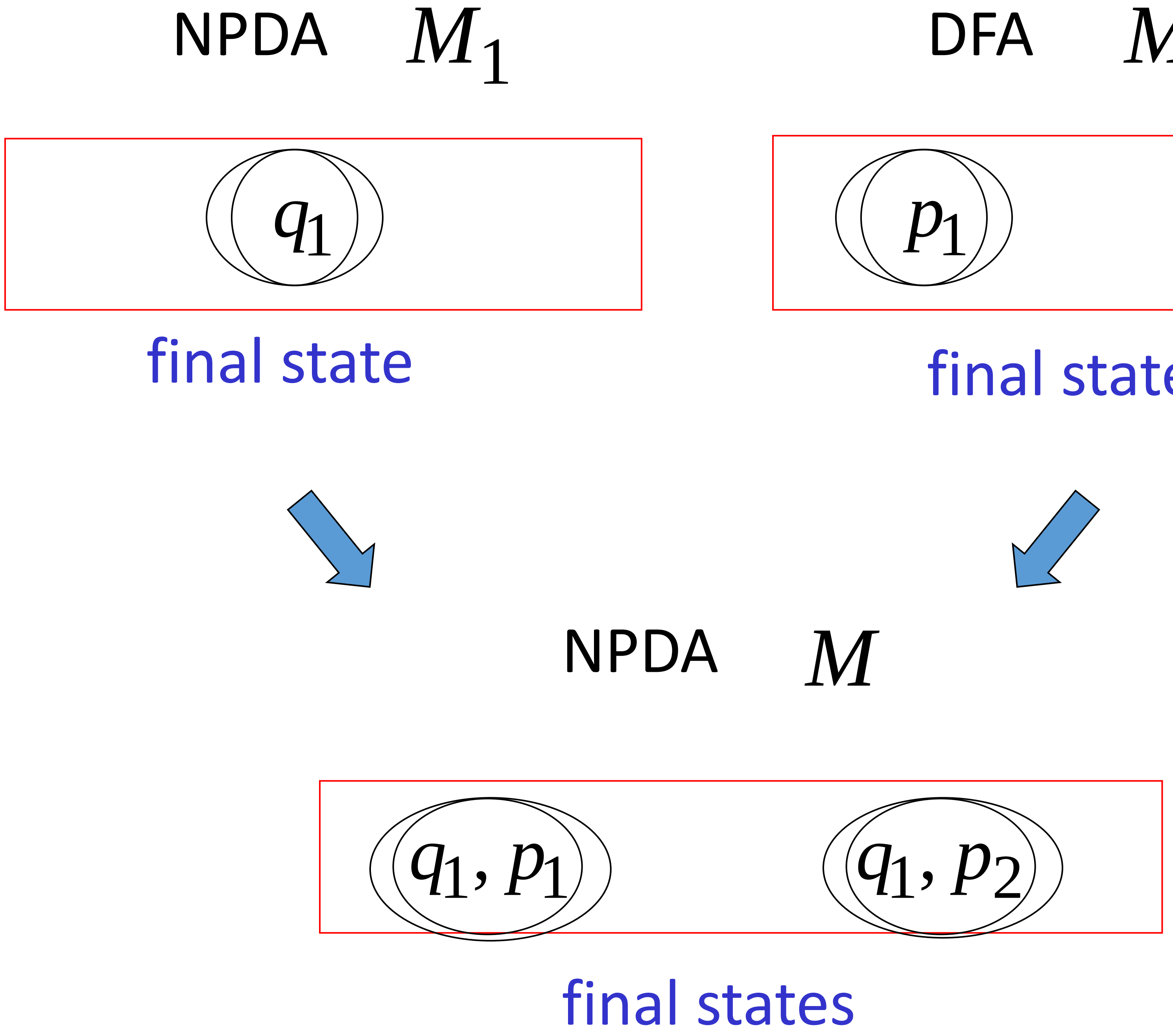
p_0

initial state

NPDA M

q_0, p_0

Initial state

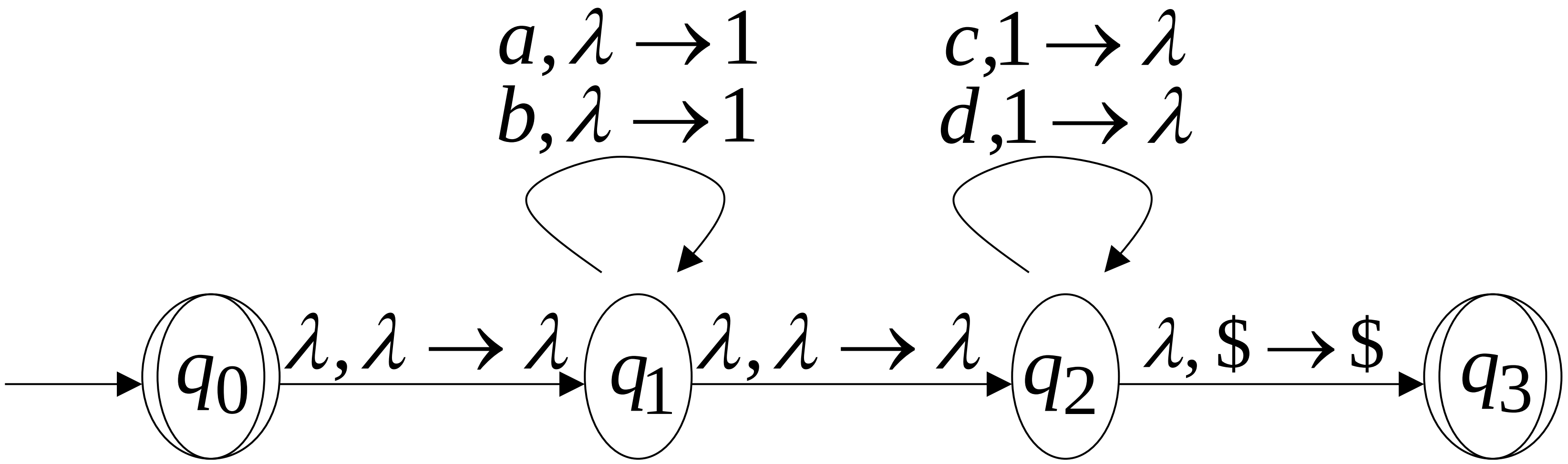

NPDA M_1
DFA M_2
q_1
p_1
p_2
final state
final states
NPDA M
q_1, p_1
q_1, p_2
final states

Example:

context-free

$$L_1 = \{w_1 w_2 : |w_1| = |w_2|,\ w_1 \in \{a,b\}^*,\ w_2 \in \{c,d\}^*\}$$

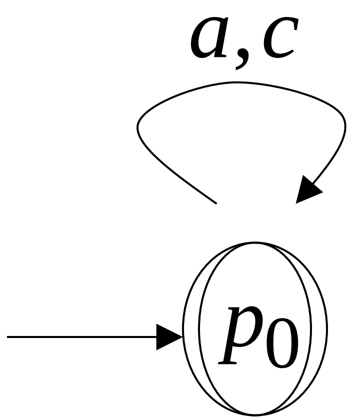
NPDA M_1



regular

$$L_2 = \{a, c\}^*$$

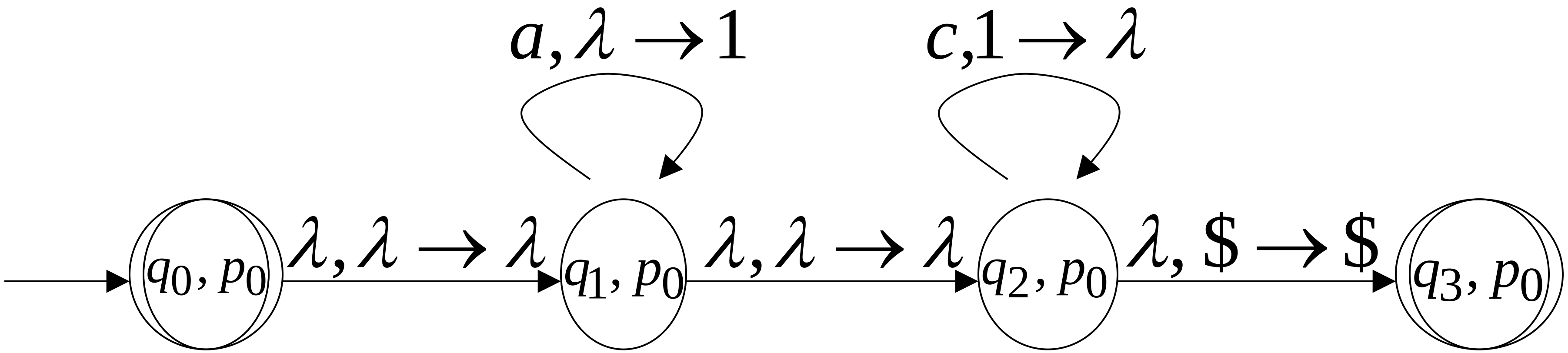
DFA M_2



context-free

Automaton for: $L_1 \cap L_2 = \{a^n c^n : \ n \geq 0\}$

NPDA M



In General:

M simulates in parallel M_1 and M_2

M accepts string w if and only if

M_1 accepts string w and

M_2 accepts string w

$$L(M) = L(M_1) \cap L(M_2)$$

Therefore:

M is NPDA

$\Downarrow$

$L(M_1) \cap L(M_2)$ is context-free

$\Downarrow$

$L_1 \cap L_2$ is context-free

Applications
of
Regular Closure

The intersection of

a context-free language and

a regular language

is a context-free language

L_1 context free

L_2 regular

$\Longrightarrow$

Regular Closure

$$L_1 \cap L_2$$

context-free

An Application of Regular Closure

Prove that: $L = \{a^n b^n : \quad n \neq 100, n \geq 0\}$

is context-free

We know:

$\{a^n b^n : n \geq 0\}$ is context-free

We also know:

$$L_1 = \{a^{100}b^{100}\}$$ is regular

$$\Downarrow$$

$$\overline{L_1} = \{(a+b)^*\} - \{a^{100}b^{100}\}$$ is regular

$$\{a^n b^n\}$$

context-free

$$\overline{L_1} = \{(a+b)^*\} - \{a^{100} b^{100}\}$$

regular

(regular closure)

$$\{a^n b^n\} \cap \overline{L_1}$$

context-free

$$\{a^n b^n\} \cap \overline{L_1} = \{a^n b^n : \quad n \neq 100, n \geq 0\} = L$$

is context-free

Another Application of Regular Closure

Prove that: $L = \{w: \quad n_a = n_b = n_c\}$

is **not** context-free

If $L = \{w: \ n_a = n_b = n_c\}$ is context-free

(regular closure)

Then $L \cap \{a^* b^* c^*\} = \{a^n b^n c^n\}$

context-free regular context-free

Impossible!!!

Therefore, L is **not** context free