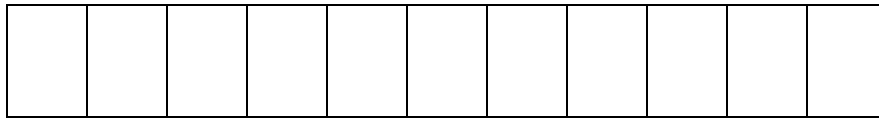


Pushdown Automata

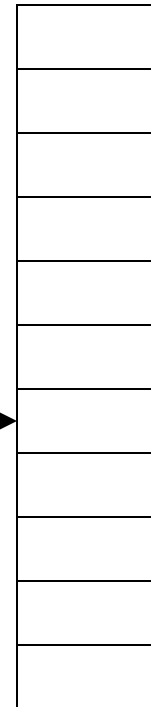
PDA's

Pushdown Automaton -- PDA

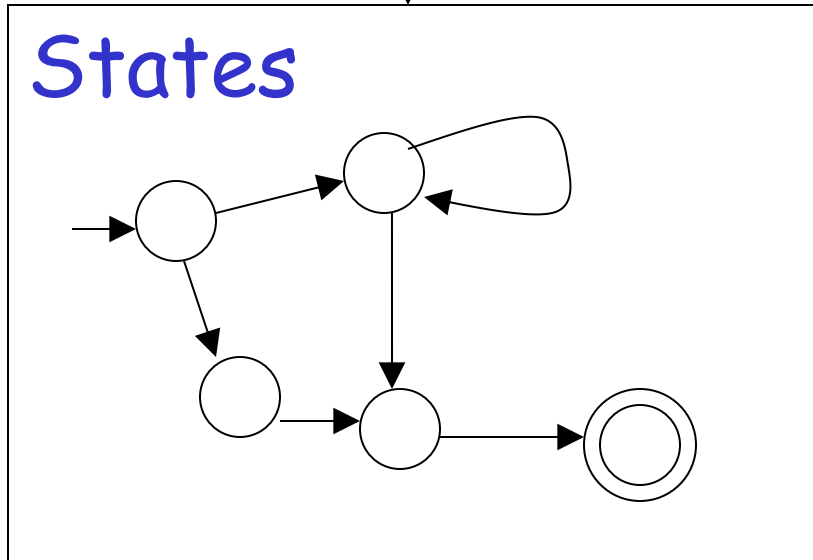
Input String



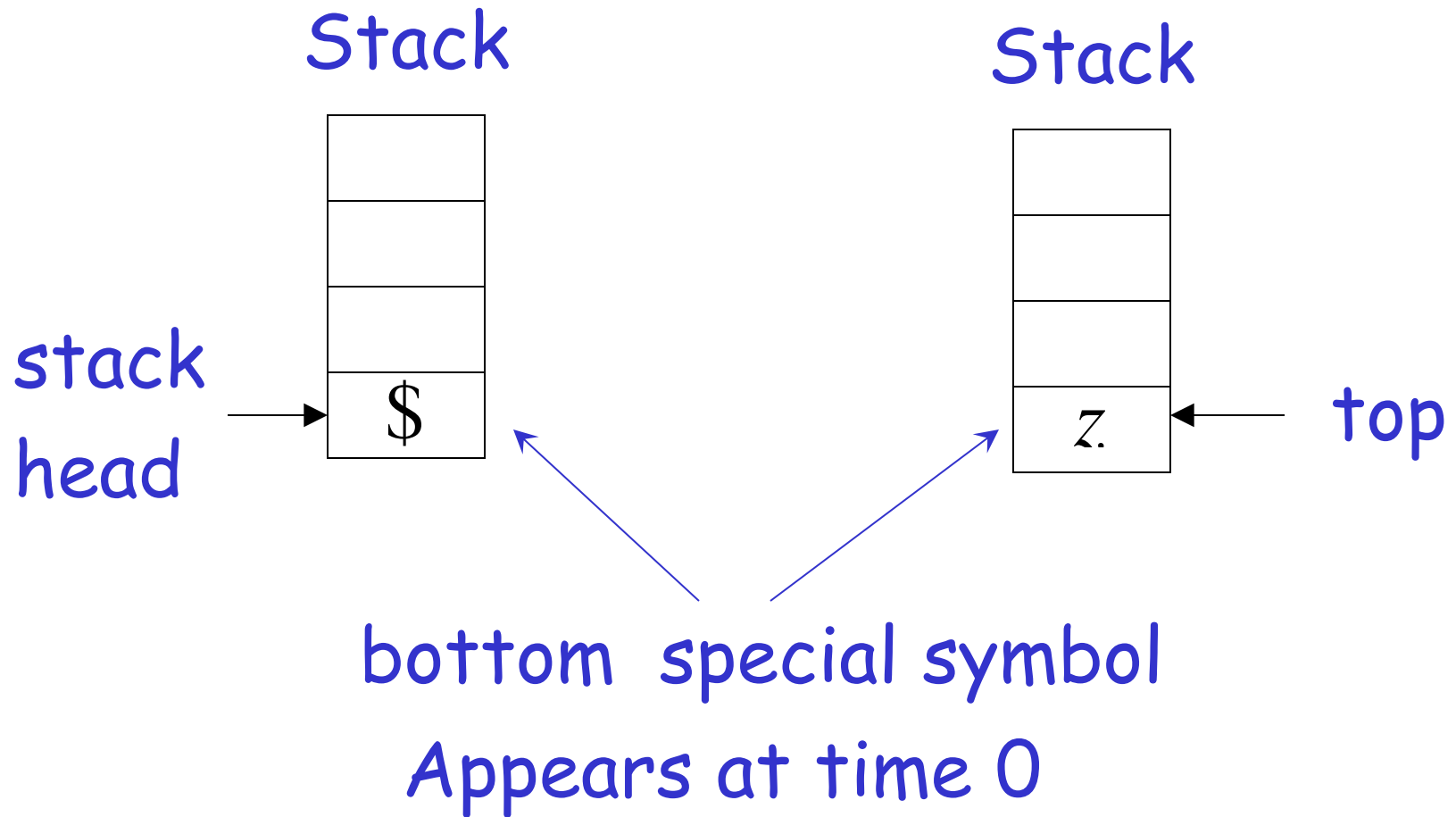
Stack



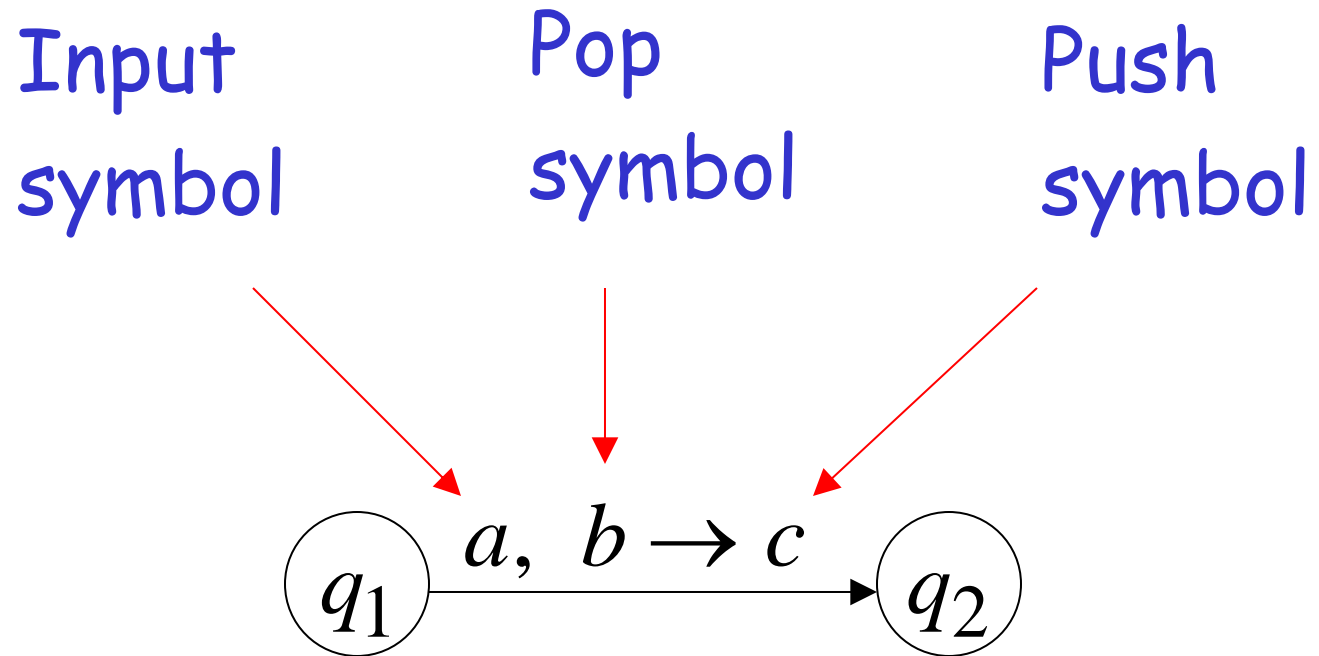
States

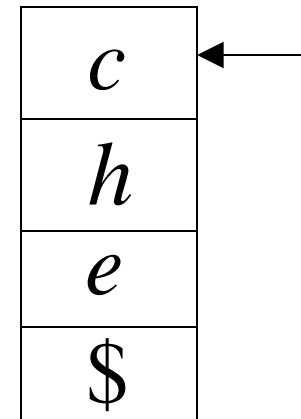
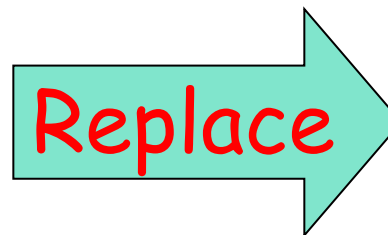
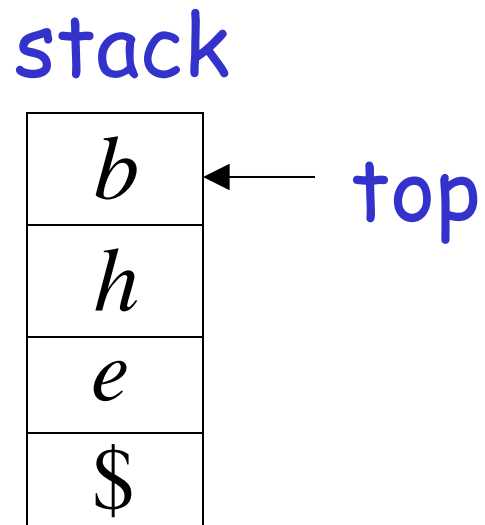
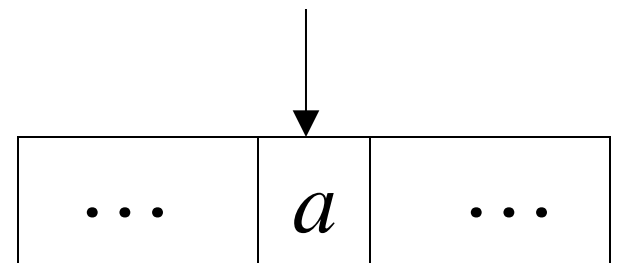
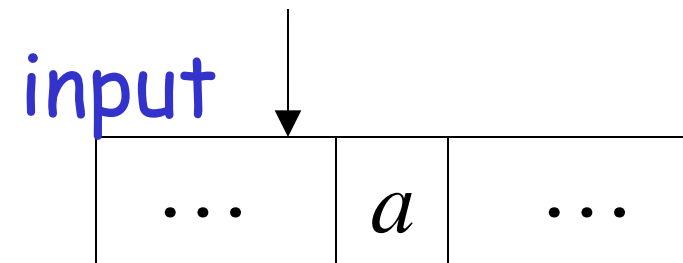
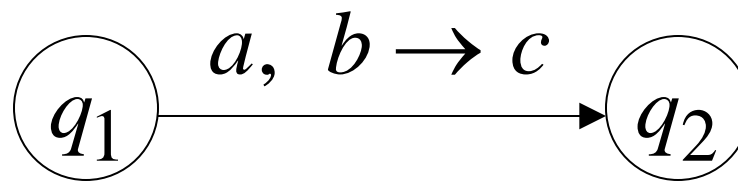


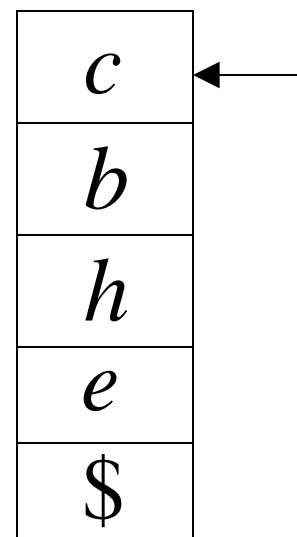
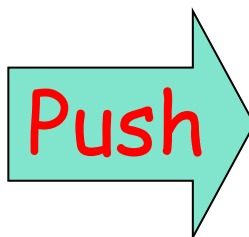
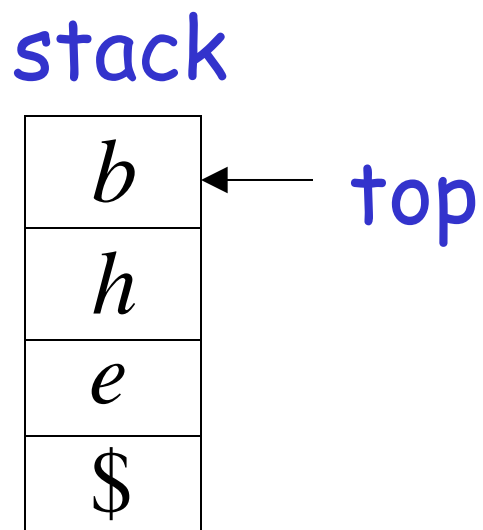
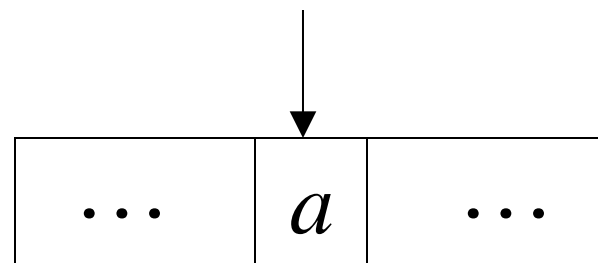
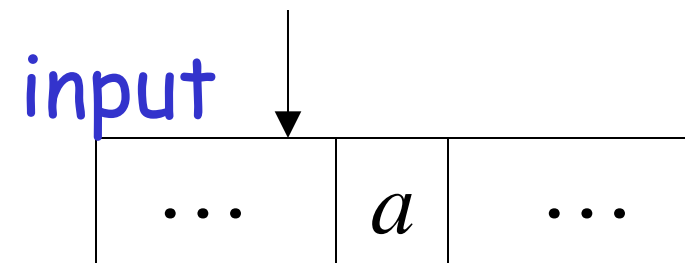
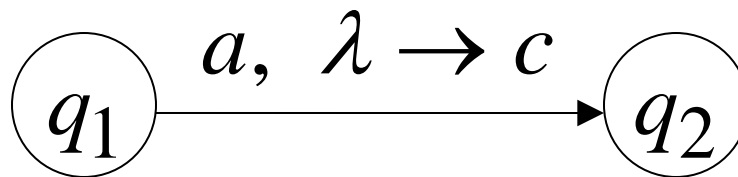
Initial Stack Symbol

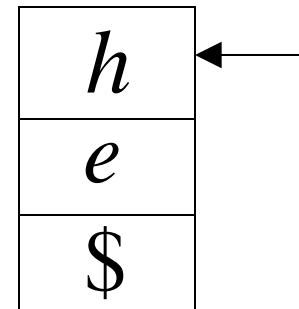
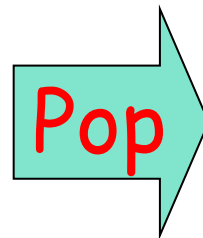
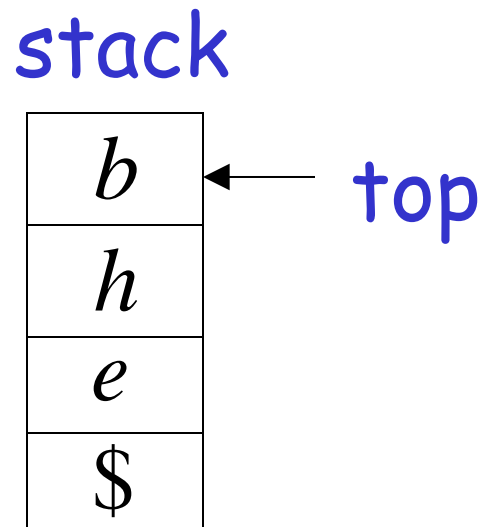
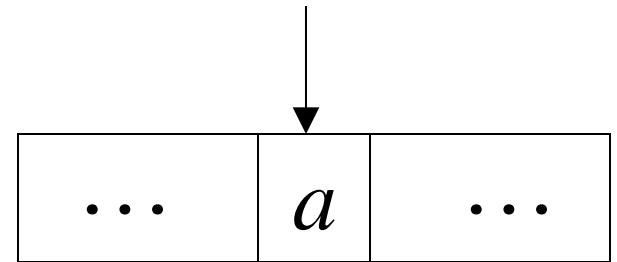
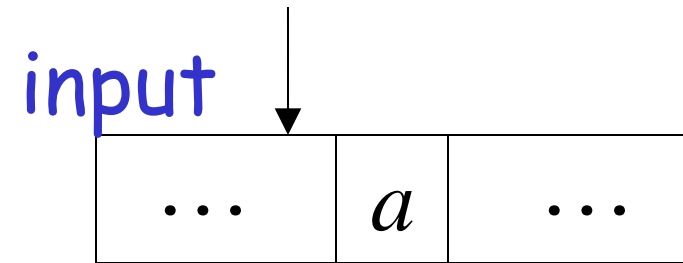
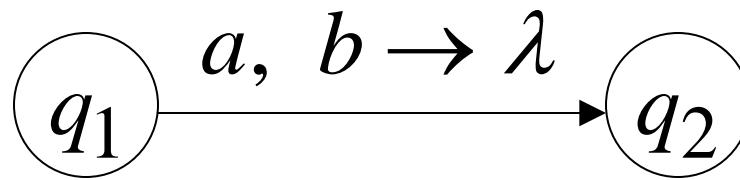


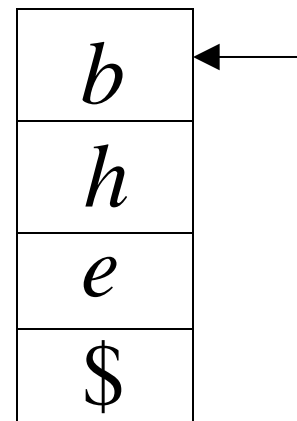
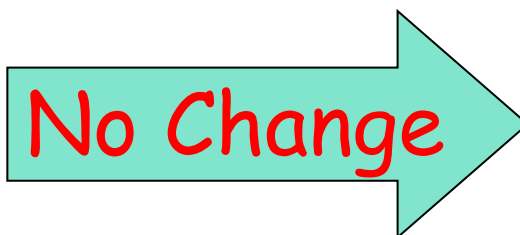
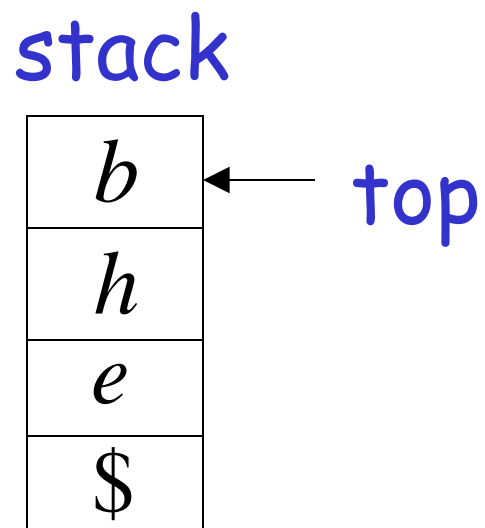
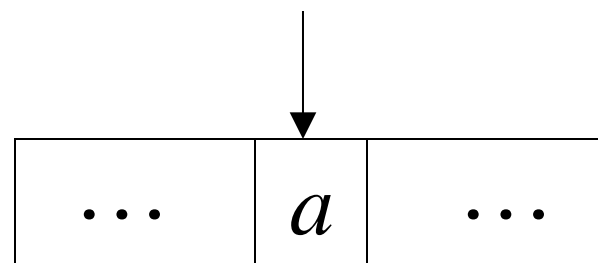
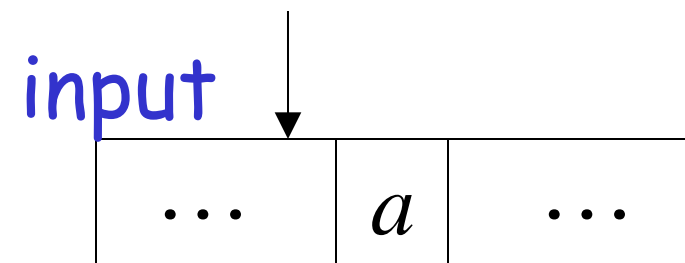
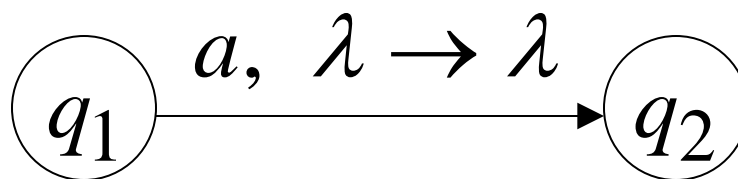
The States



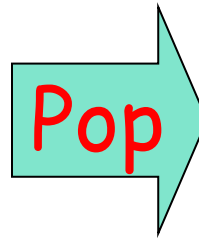
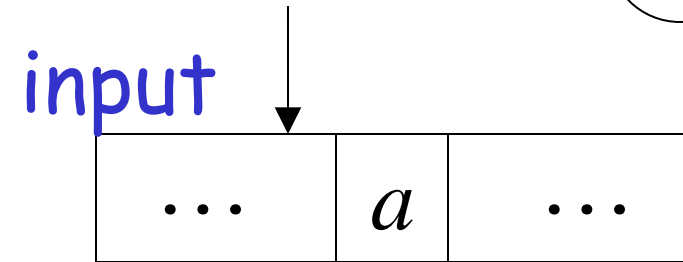
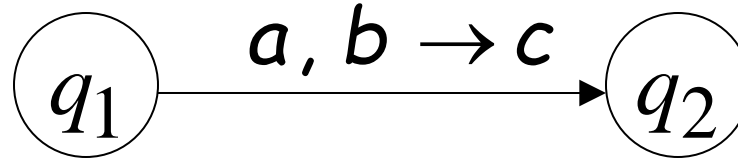








Pop from Empty Stack



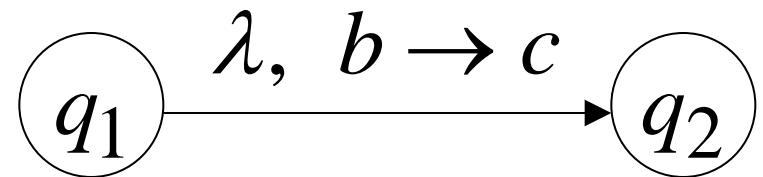
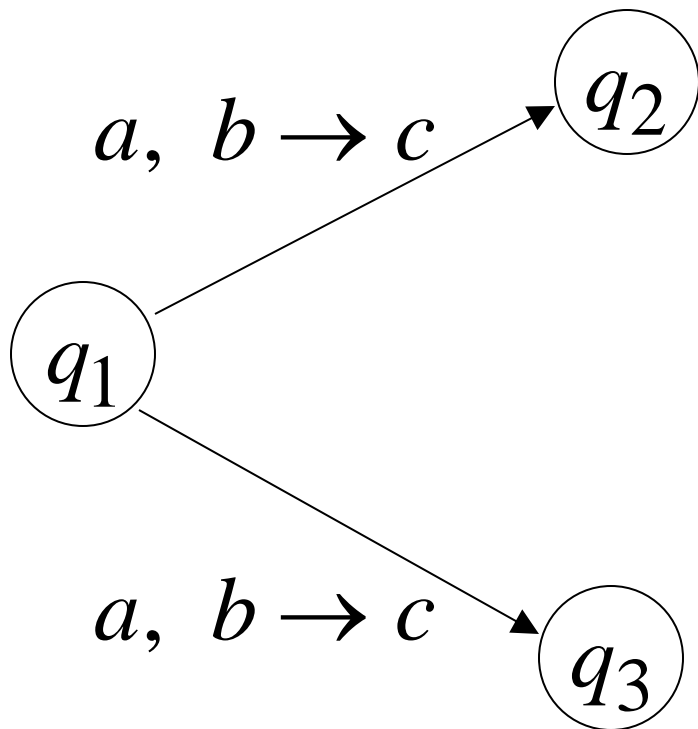
Automaton halts!

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

Non-Determinism

PDAs are non-deterministic

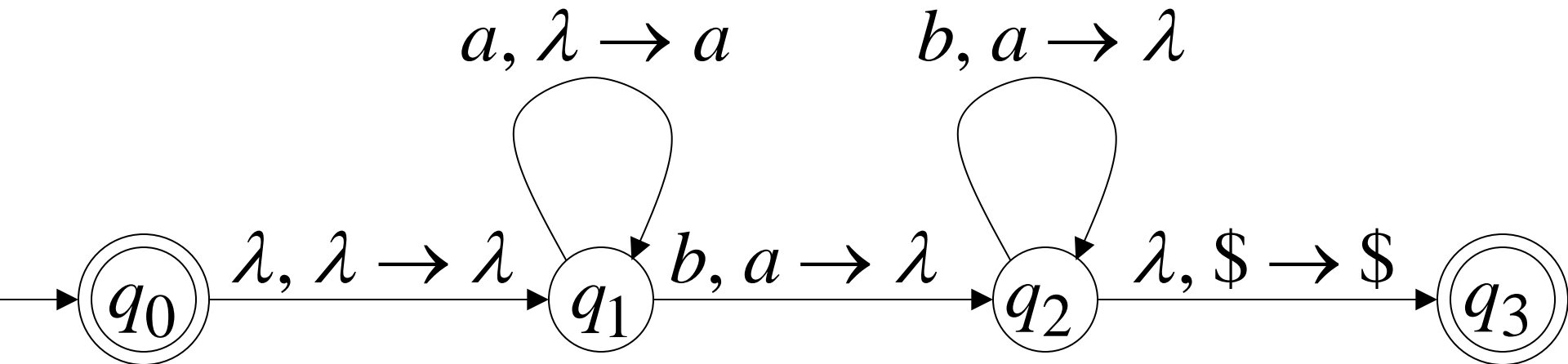
Allowed non-deterministic transitions



λ – transition

Example PDA

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



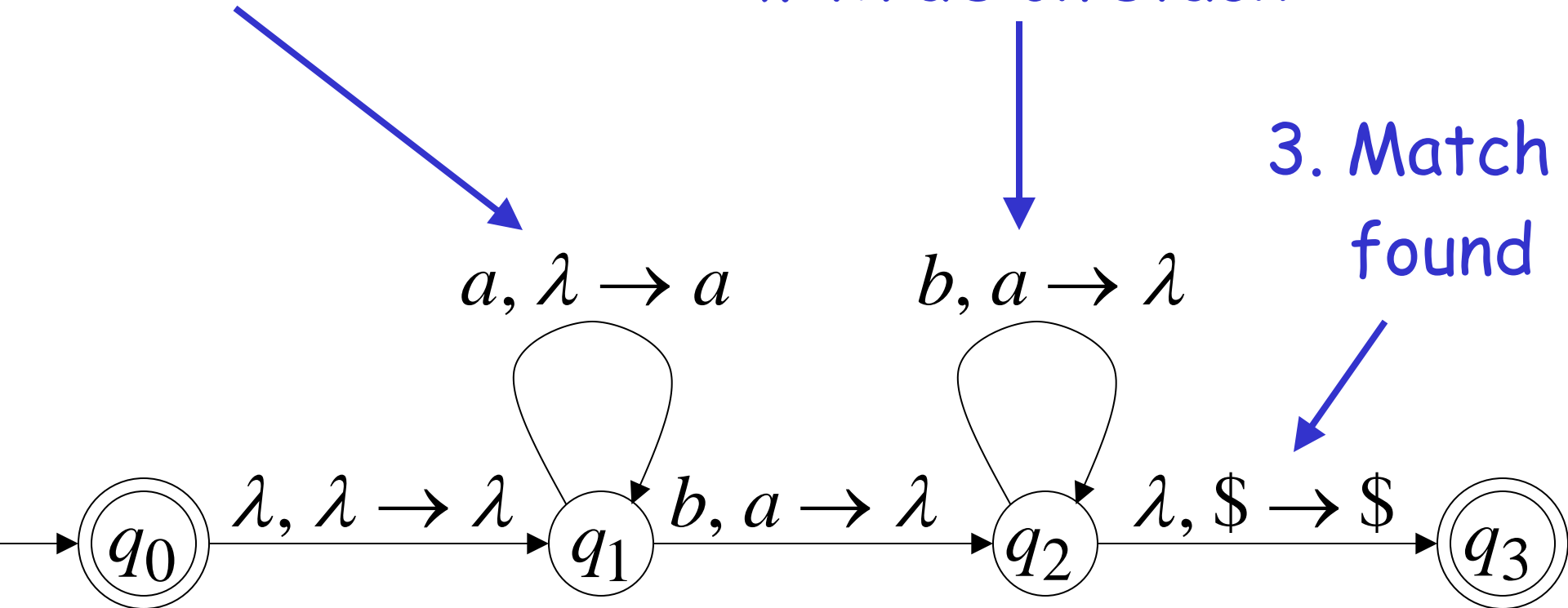
$$L(M) = \{a^n b^n : n \geq 0\}$$

Basic Idea:

1. Push the a's
on the stack

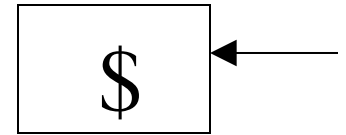
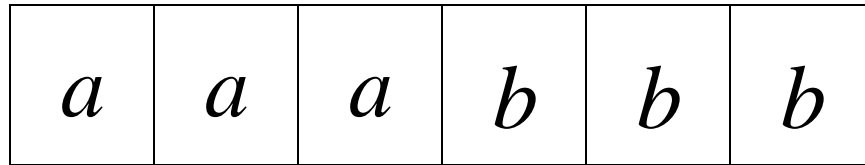
2. Match the b's on input
with a's on stack

3. Match
found

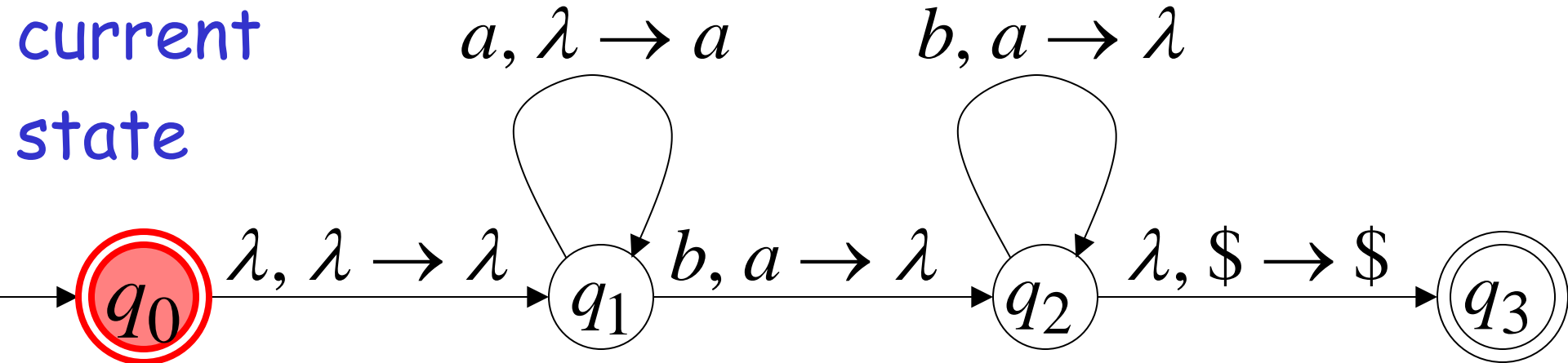


Execution Example: Time 0

Input

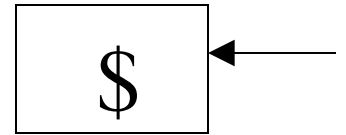
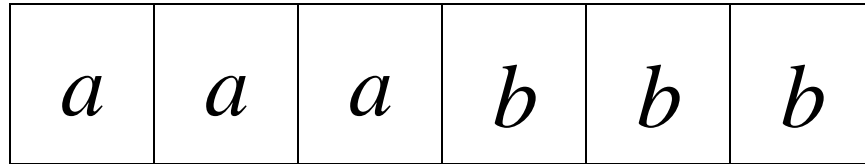


Stack

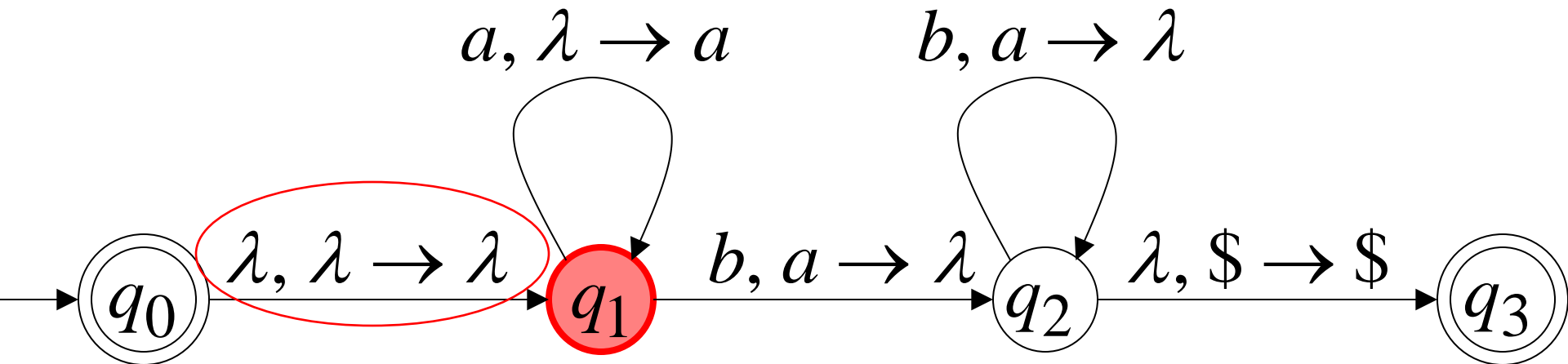


Time 1

Input

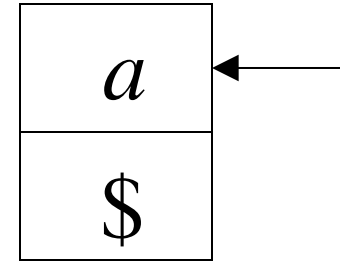
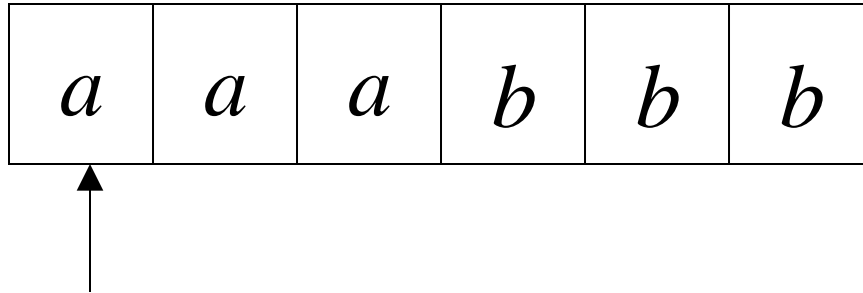


Stack

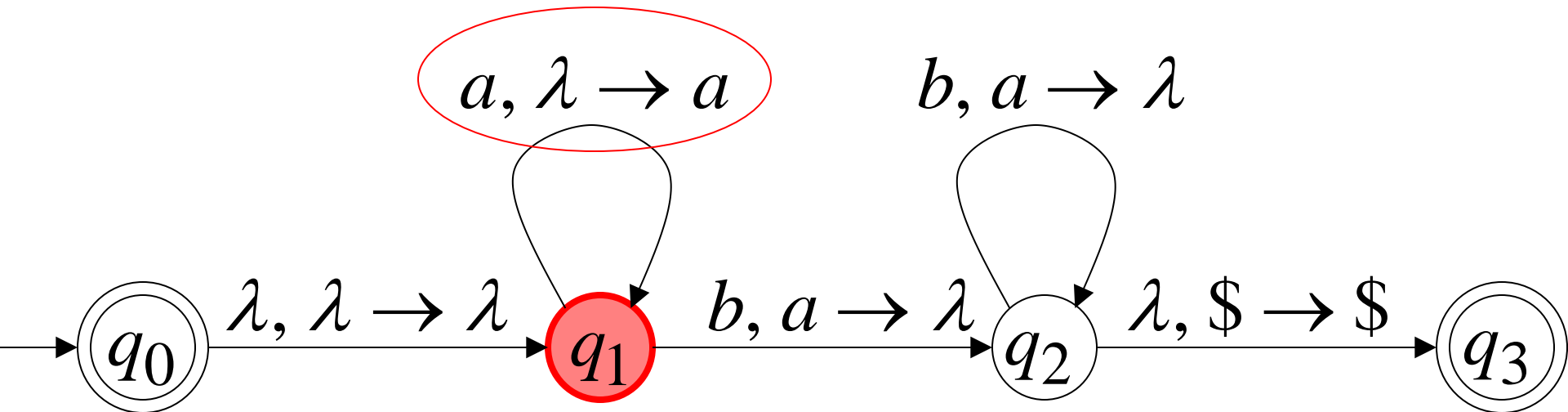


Time 2

Input

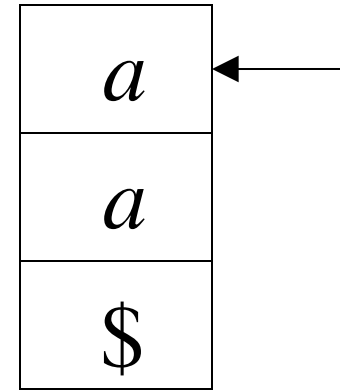
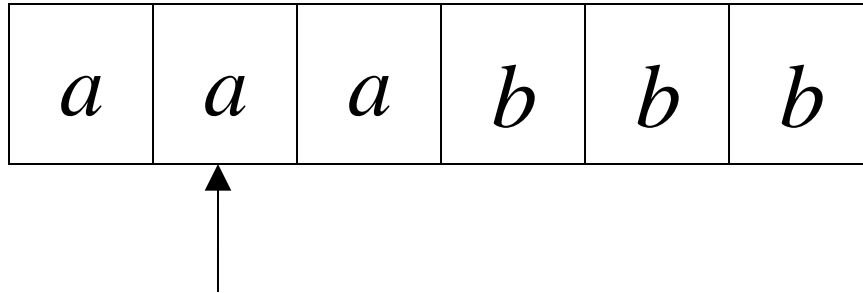


Stack

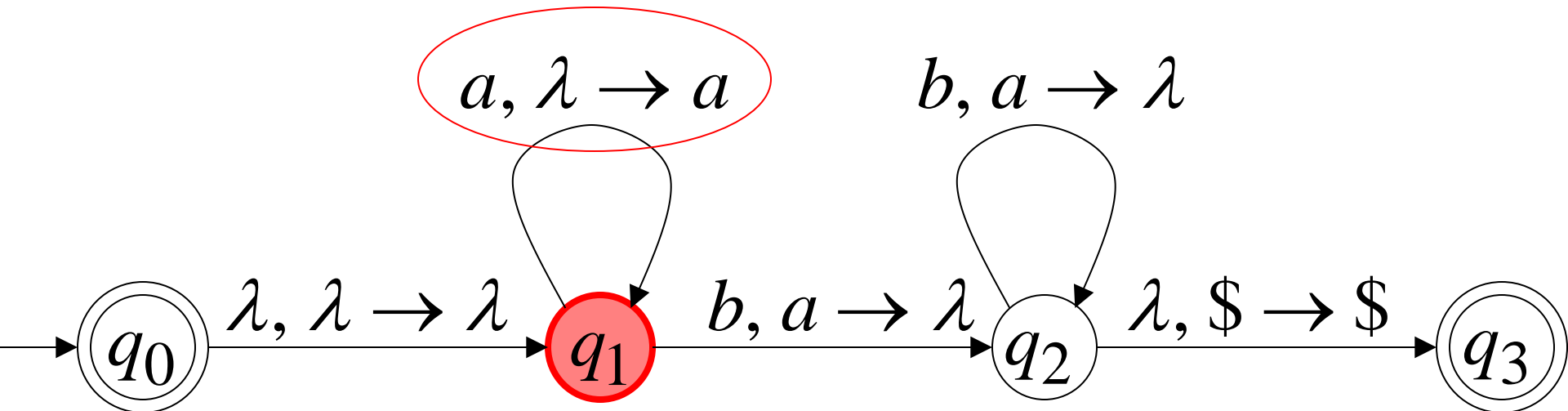


Time 3

Input

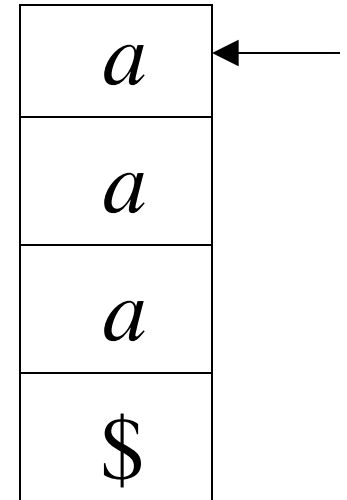
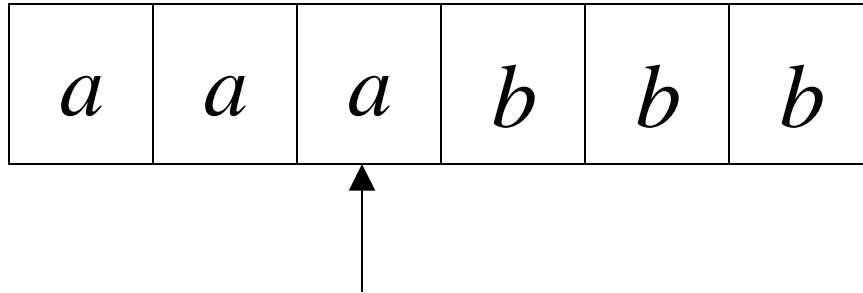


Stack

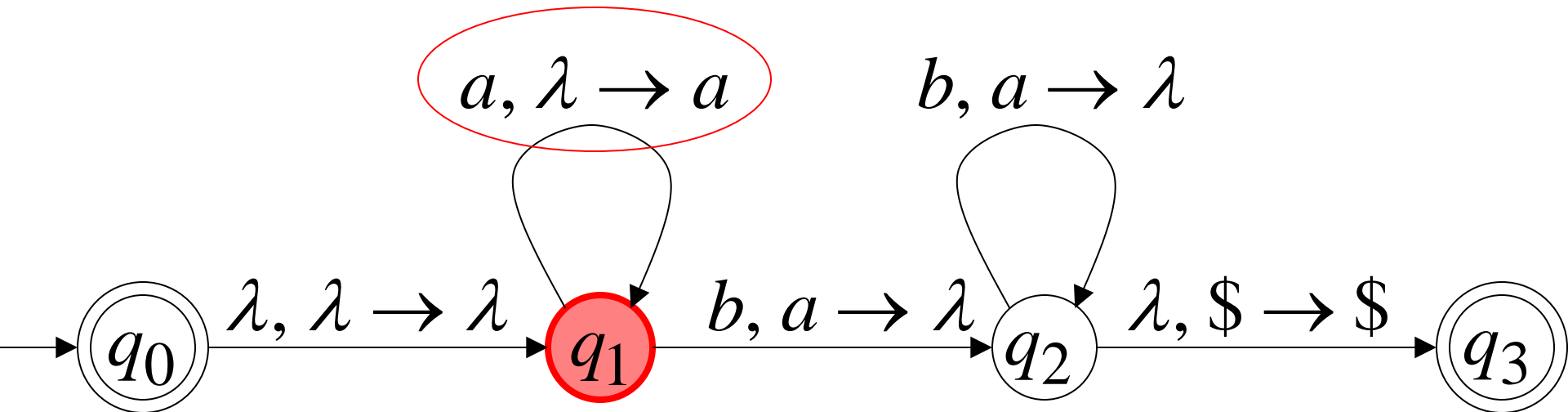


Time 4

Input

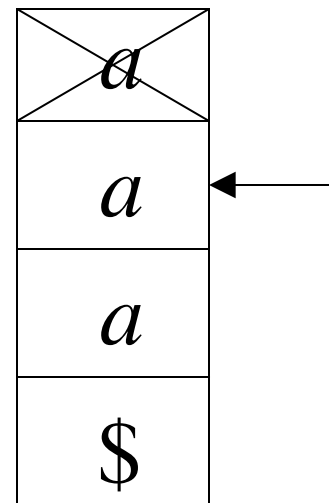
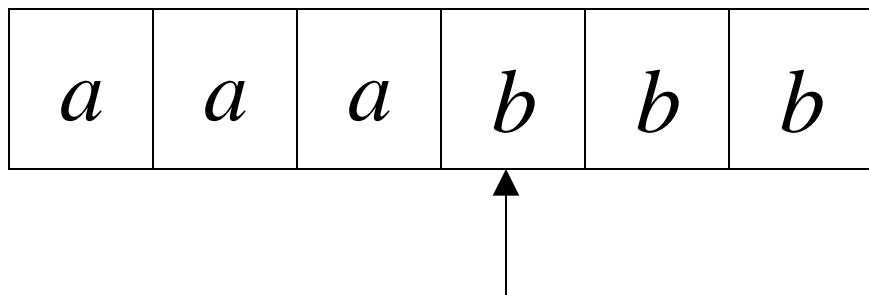


Stack

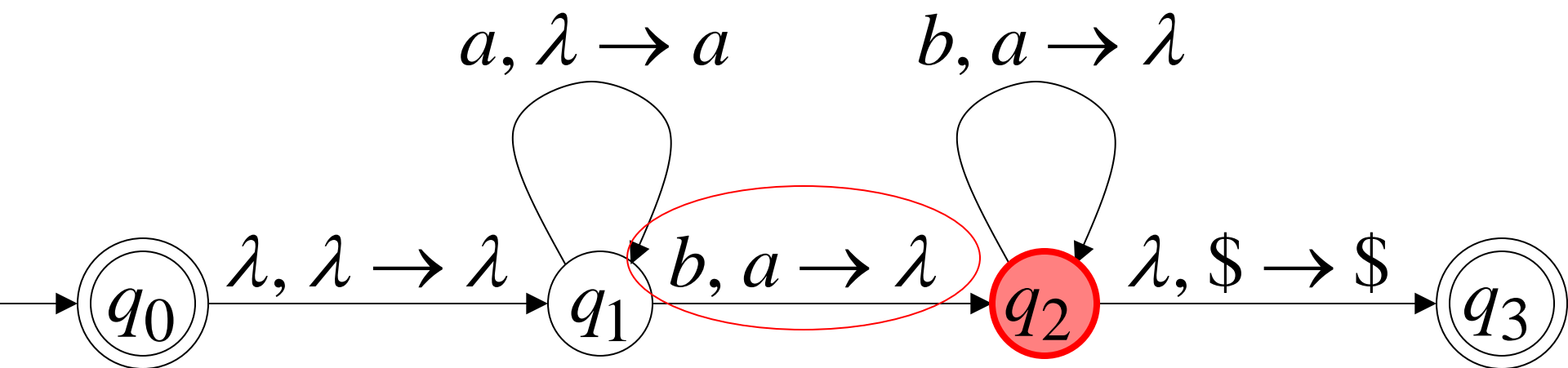


Time 5

Input

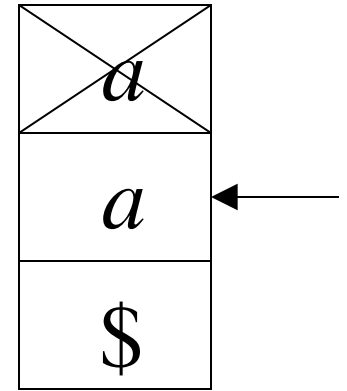
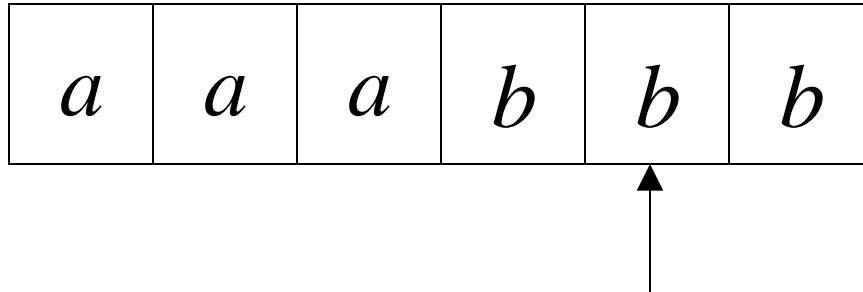


Stack

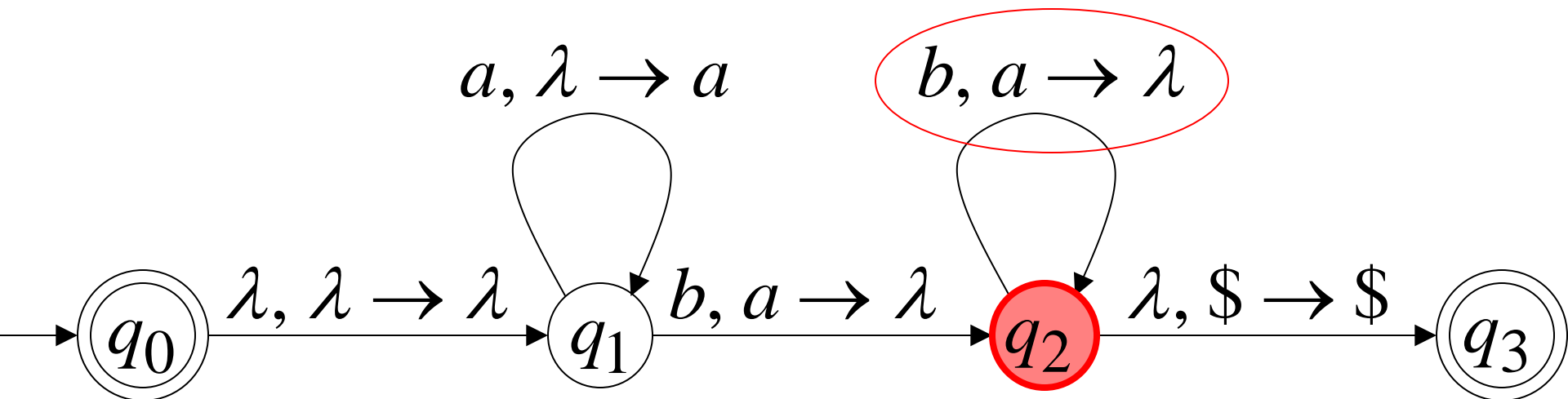


Time 6

Input

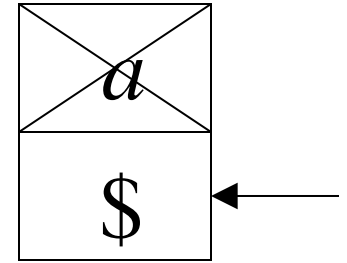
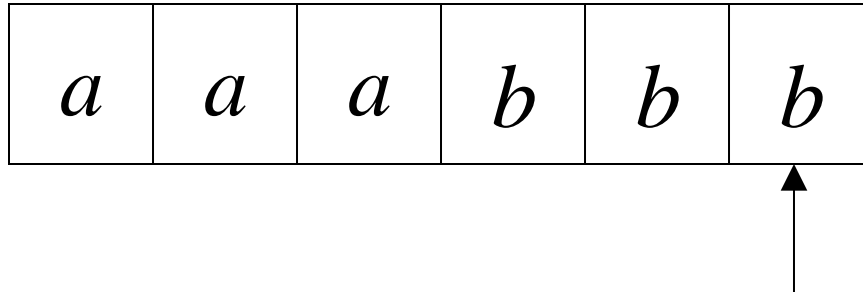


Stack

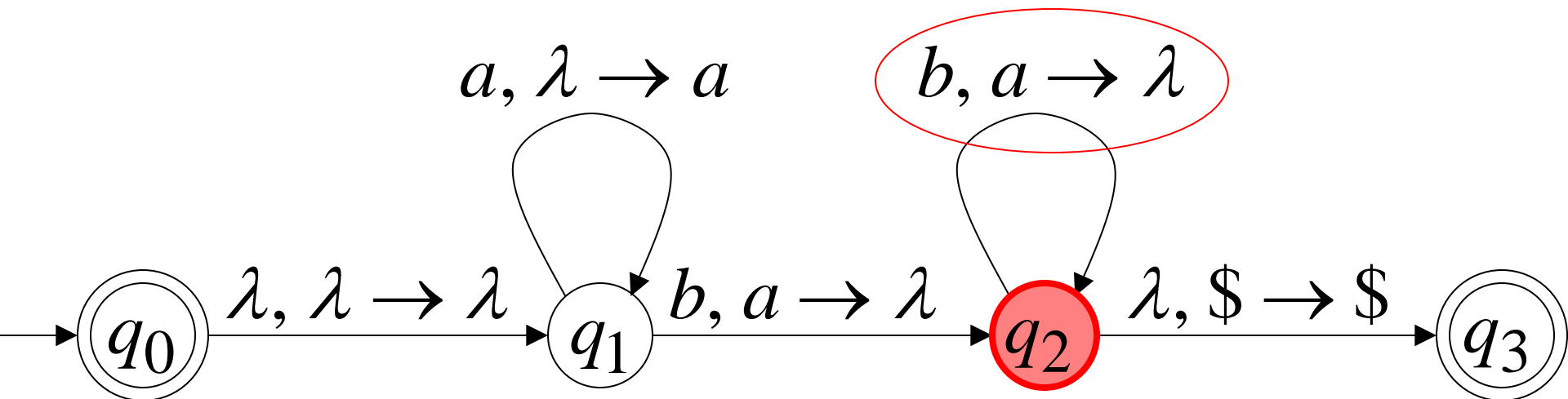


Time 7

Input

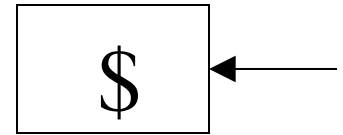
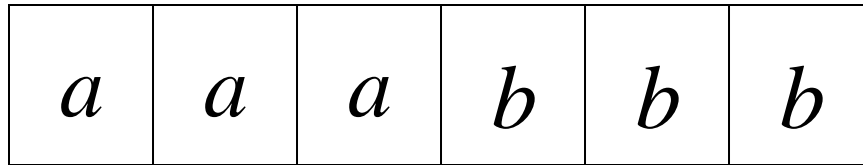


Stack

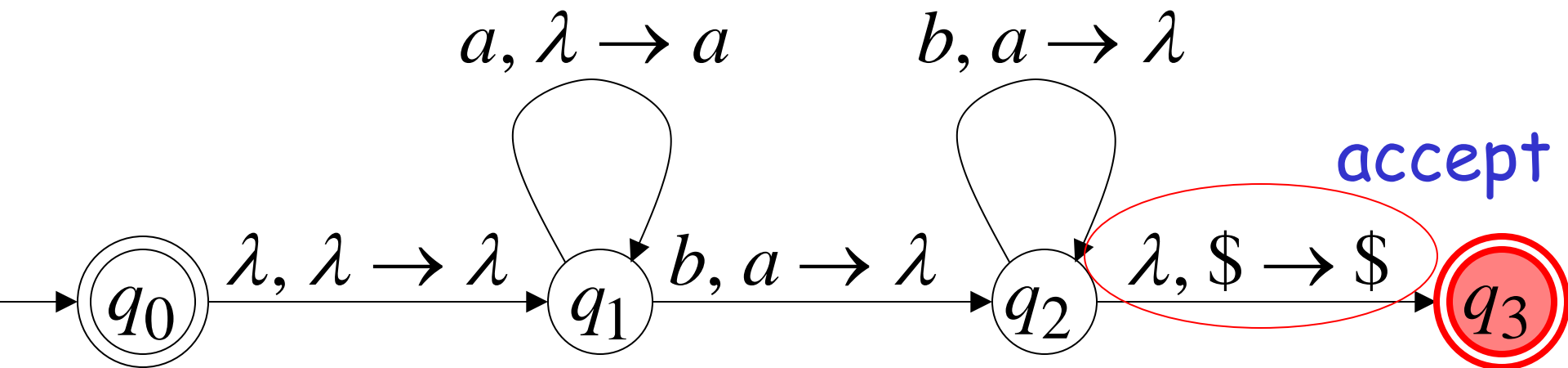


Time 8

Input



Stack



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

All the input is consumed

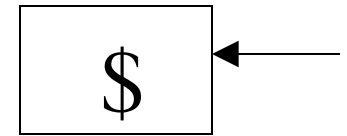
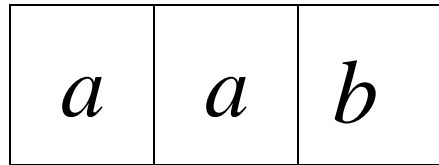
AND

The last state is an accepting state

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

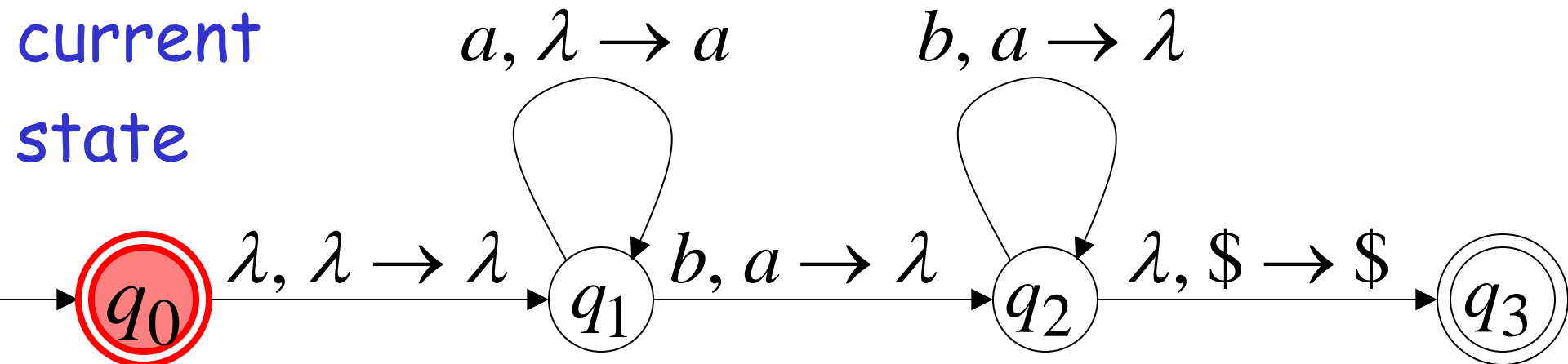
Rejection Example: Time 0

Input



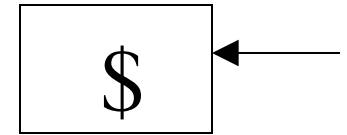
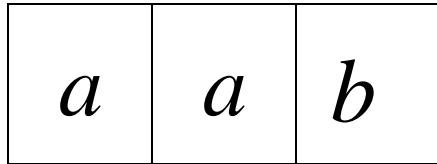
Stack

current
state



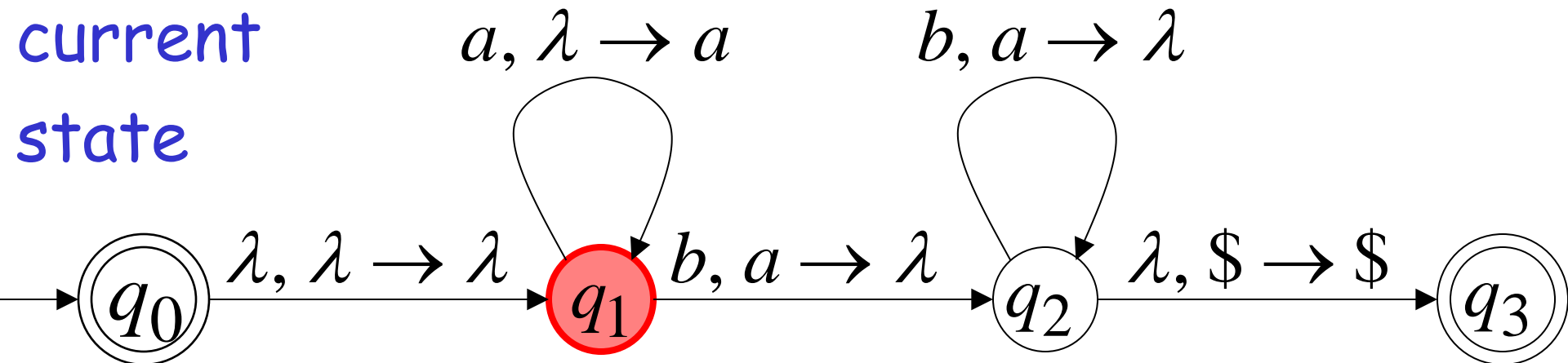
Rejection Example: Time 1

Input



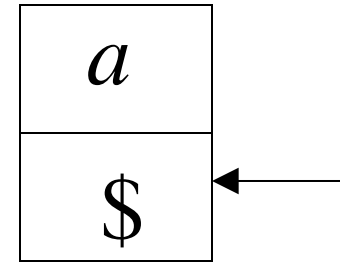
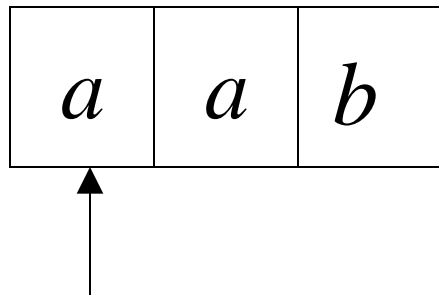
Stack

current
state

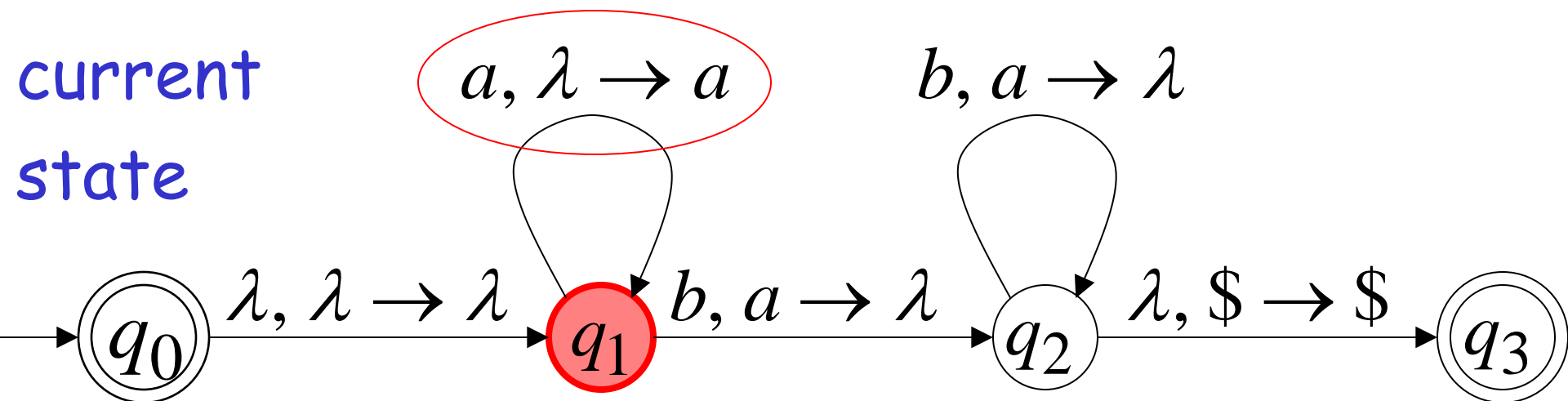


Rejection Example: Time 2

Input

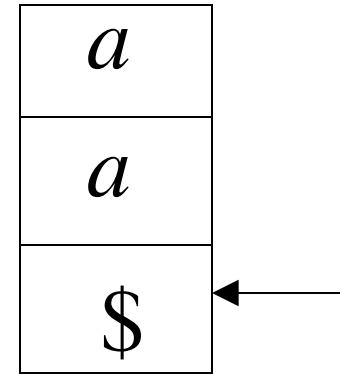
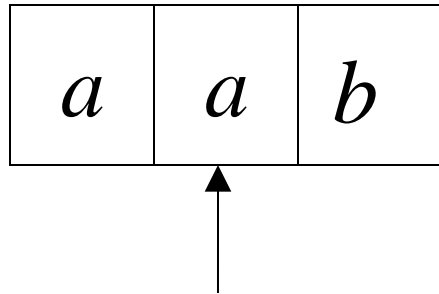


Stack

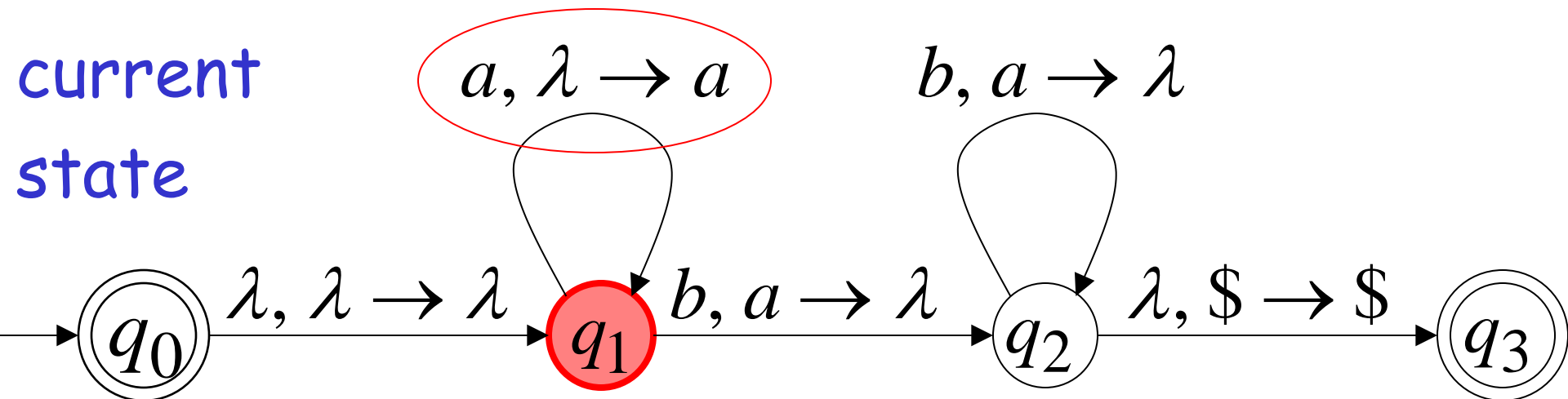


Rejection Example: Time 3

Input

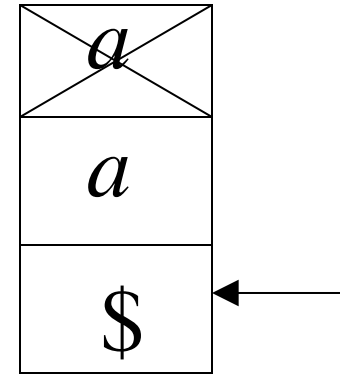
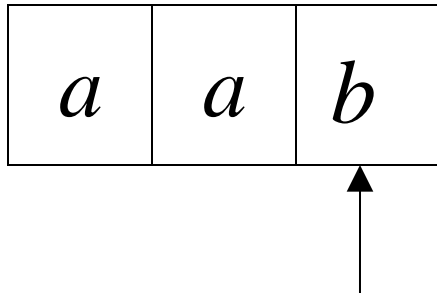


Stack



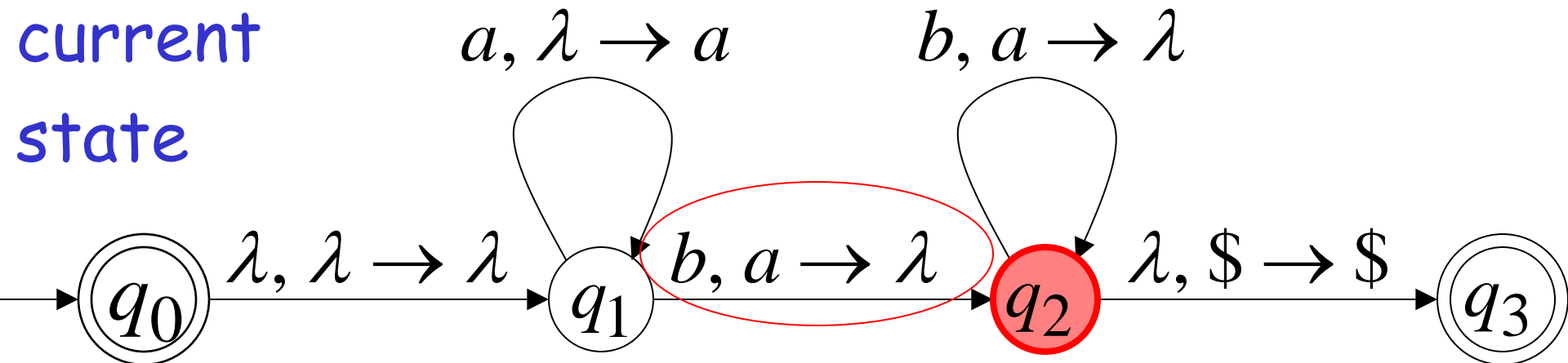
Rejection Example: Time 4

Input



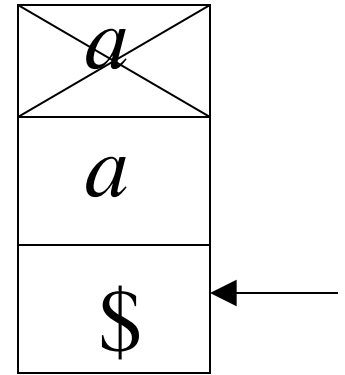
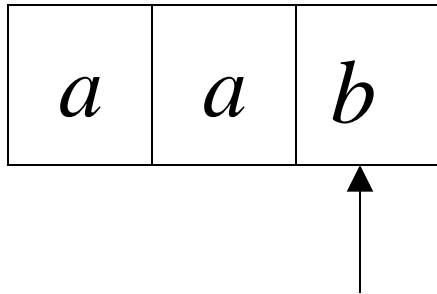
Stack

current
state



Rejection Example: Time 4

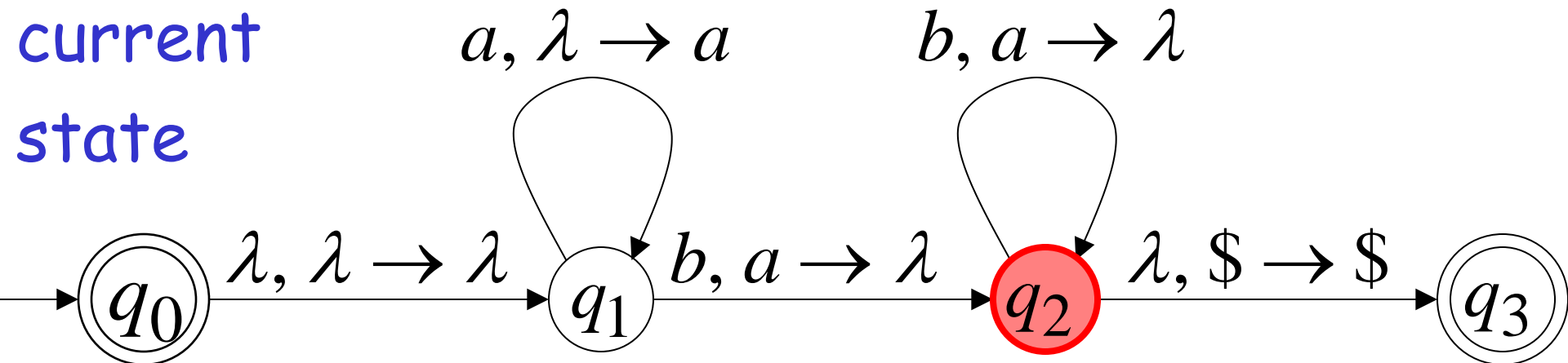
Input



Stack

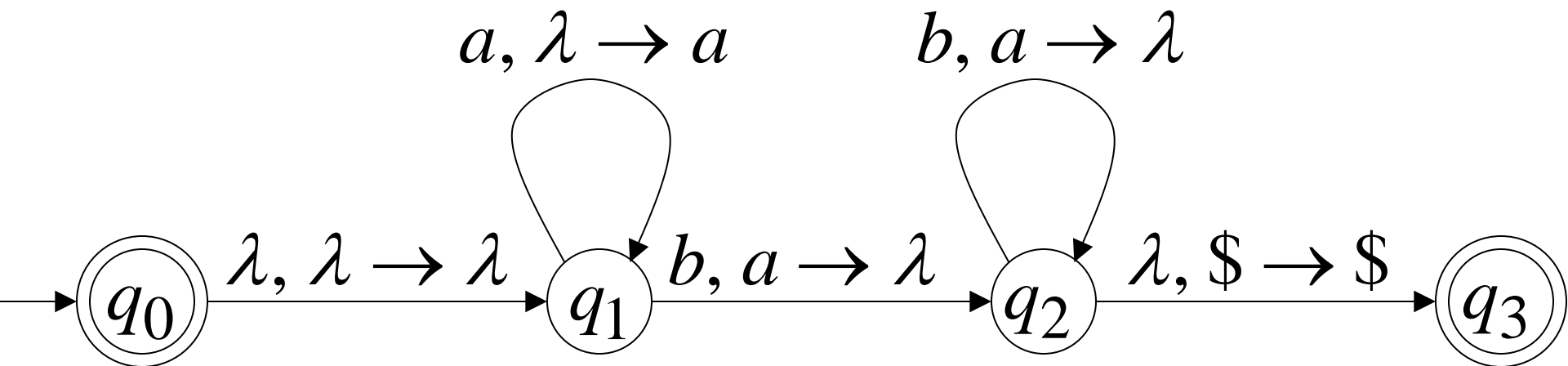
reject

current
state



There is no accepting computation for aab

The string aab is rejected by the PDA



Another PDA example

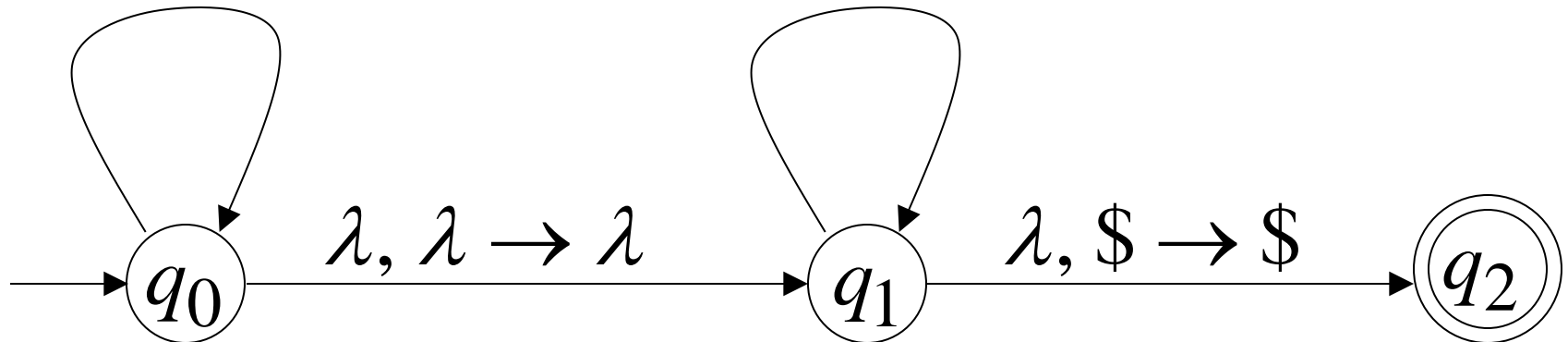
PDA M : $L(M) = \{vv^R : v \in \{a,b\}^*\}$

$a, \lambda \rightarrow a$

$a, a \rightarrow \lambda$

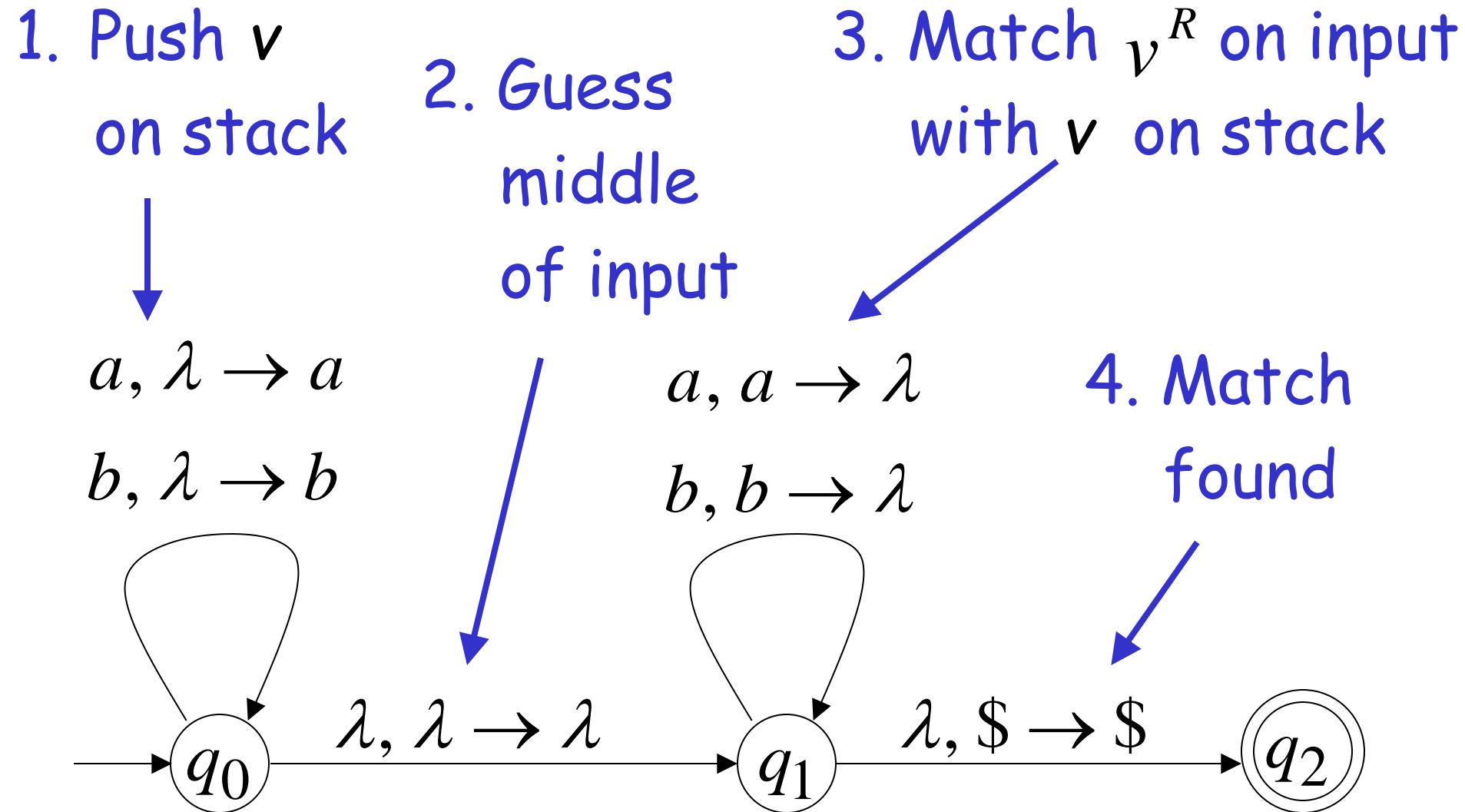
$b, \lambda \rightarrow b$

$b, b \rightarrow \lambda$



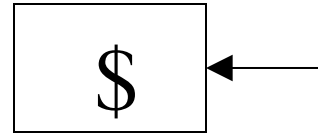
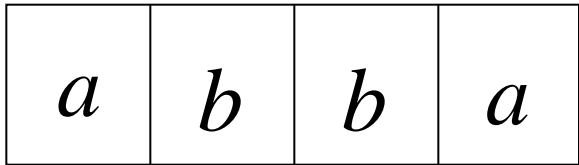
Basic Idea:

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



Execution Example: Time 0

Input



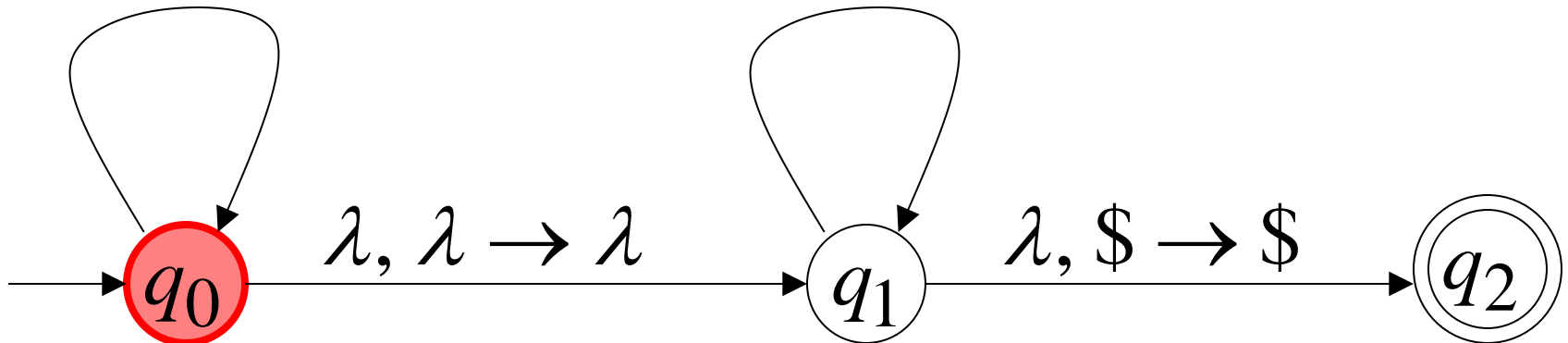
Stack

$a, \lambda \rightarrow a$

$a, a \rightarrow \lambda$

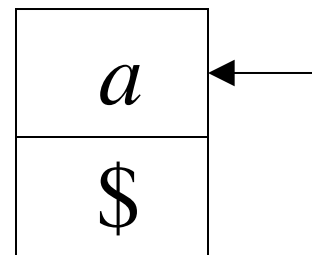
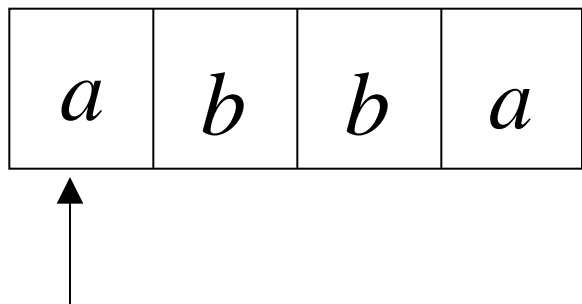
$b, \lambda \rightarrow b$

$b, b \rightarrow \lambda$



Time 1

Input



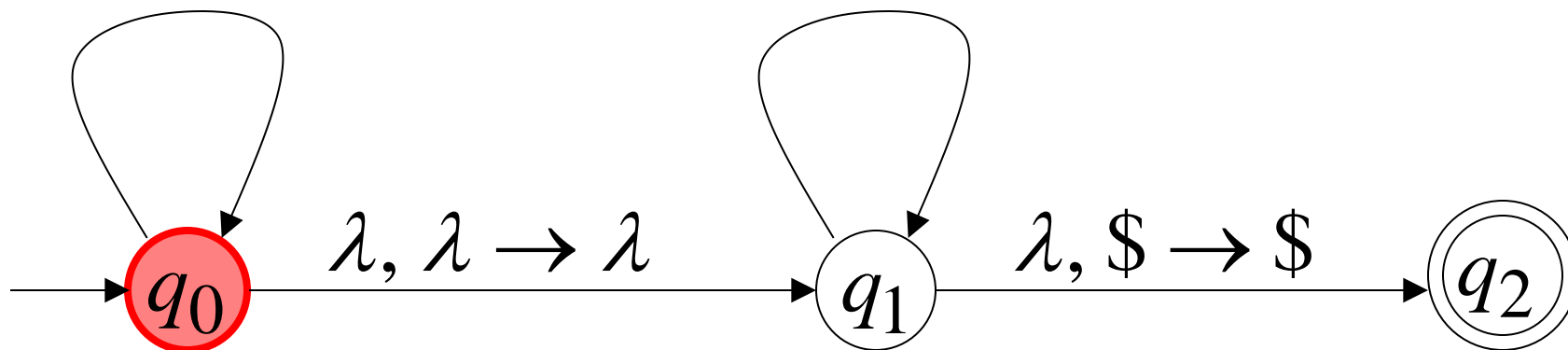
Stack

$a, \lambda \rightarrow a$

$b, \lambda \rightarrow b$

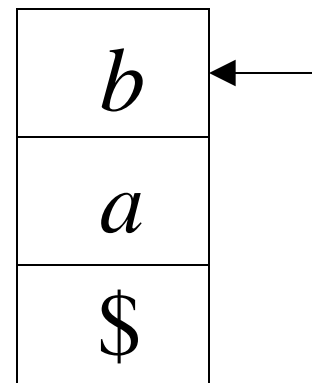
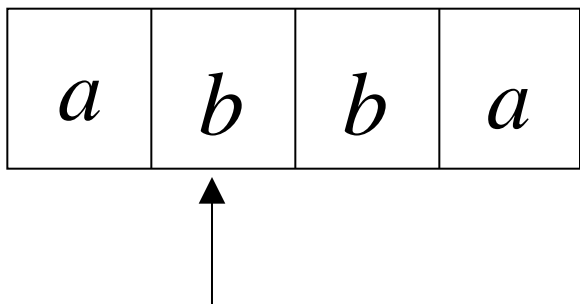
$a, a \rightarrow \lambda$

$b, b \rightarrow \lambda$



Time 2

Input



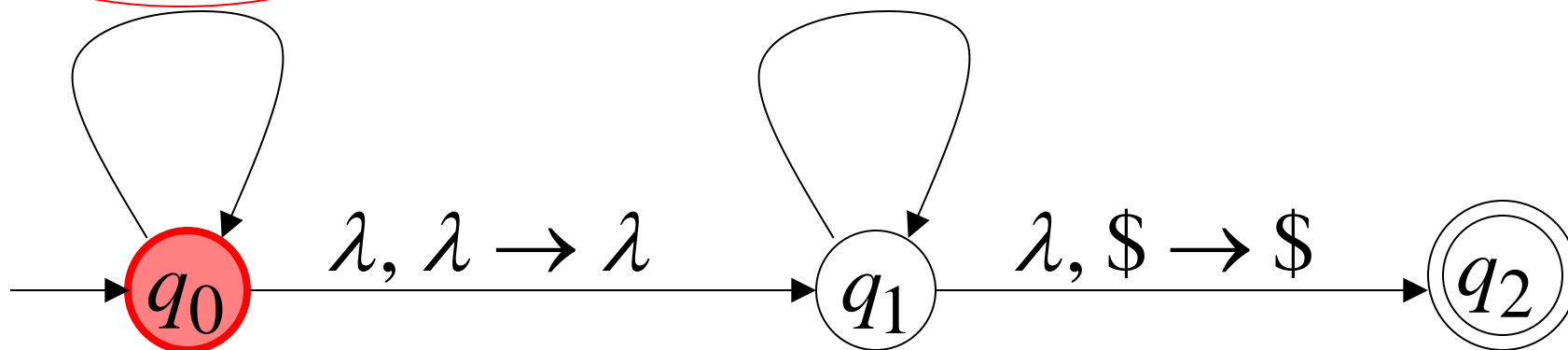
Stack

$a, \lambda \rightarrow a$

$b, \lambda \rightarrow b$

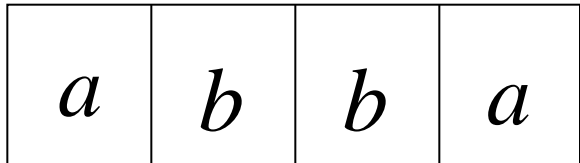
$a, a \rightarrow \lambda$

$b, b \rightarrow \lambda$

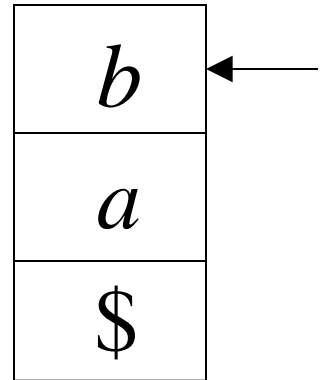


Time 3

Input



Guess the middle
of string



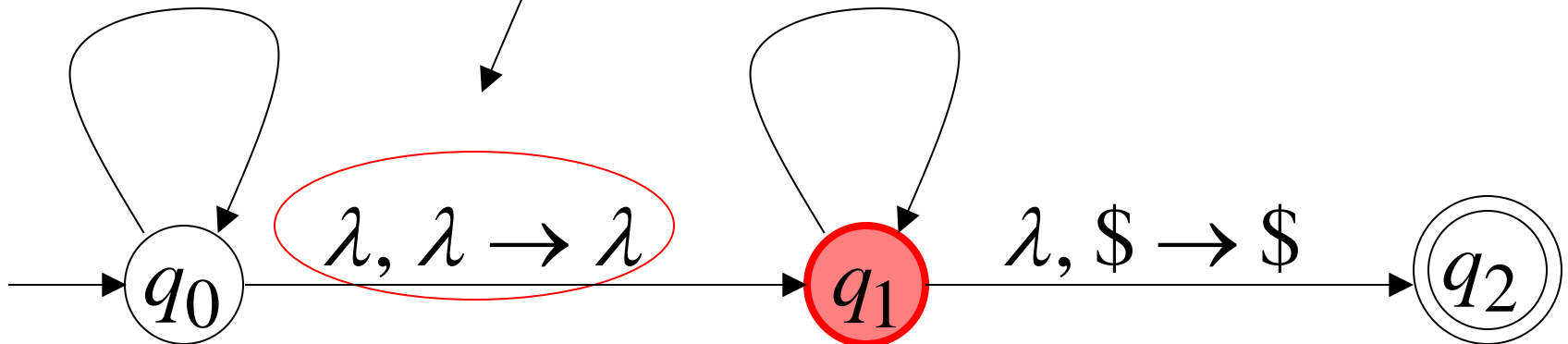
Stack

$a, \lambda \rightarrow a$

$b, \lambda \rightarrow b$

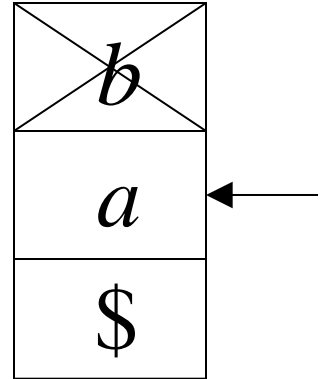
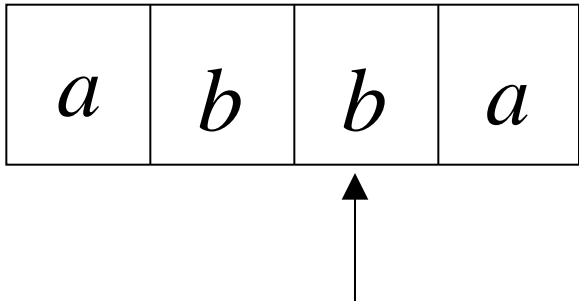
$a, a \rightarrow \lambda$

$b, b \rightarrow \lambda$



Time 4

Input



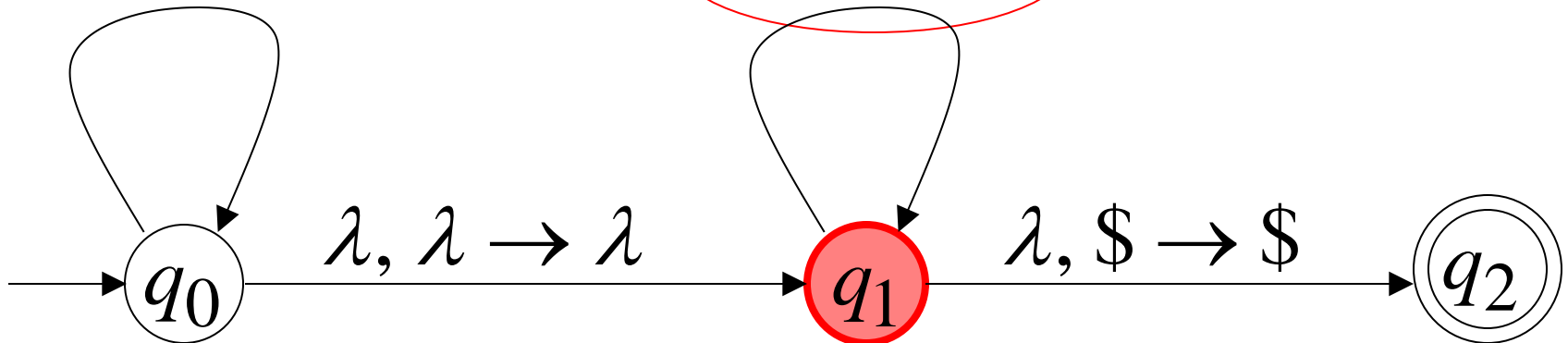
Stack

$a, \lambda \rightarrow a$

$b, \lambda \rightarrow b$

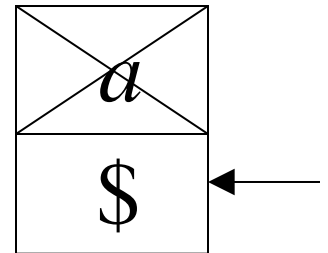
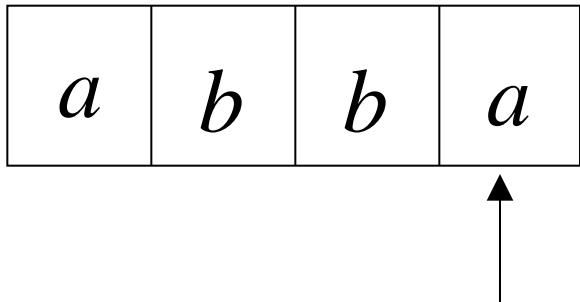
$a, a \rightarrow \lambda$

$b, b \rightarrow \lambda$



Time 5

Input



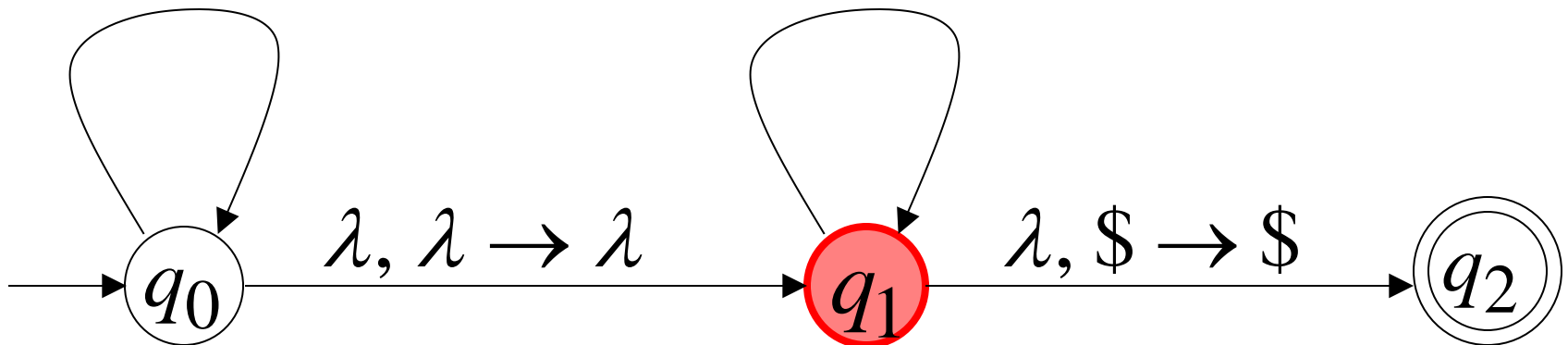
Stack

$a, \lambda \rightarrow a$

$b, \lambda \rightarrow b$

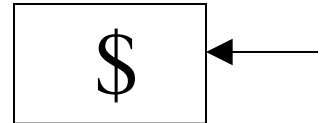
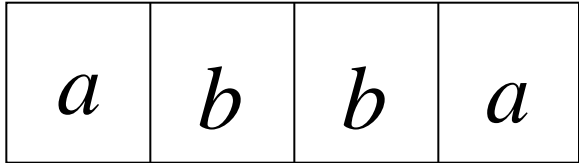
$a, a \rightarrow \lambda$

$b, b \rightarrow \lambda$



Time 6

Input



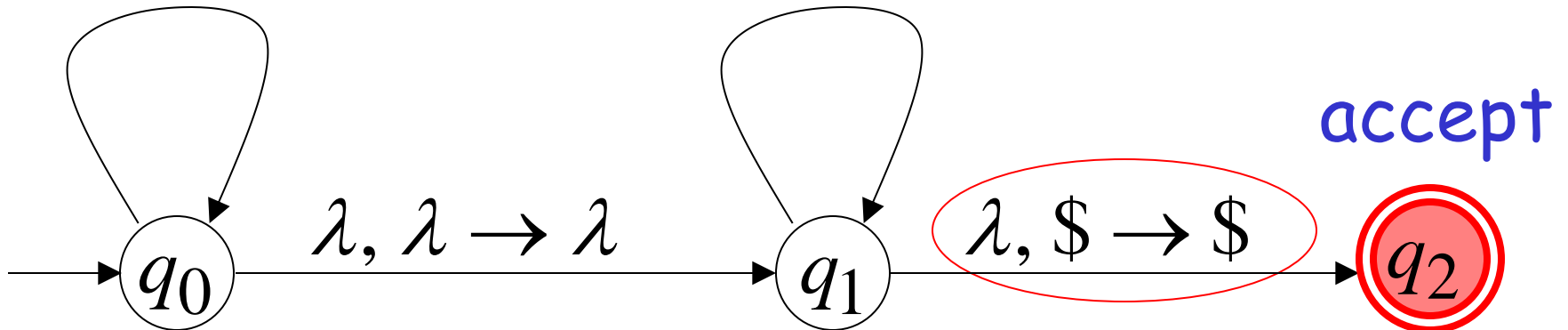
Stack

$a, \lambda \rightarrow a$

$a, a \rightarrow \lambda$

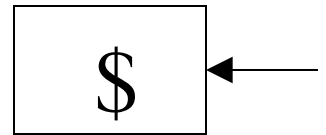
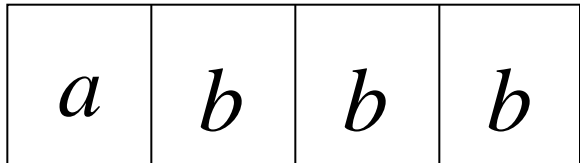
$b, \lambda \rightarrow b$

$b, b \rightarrow \lambda$



Rejection Example: Time 0

Input



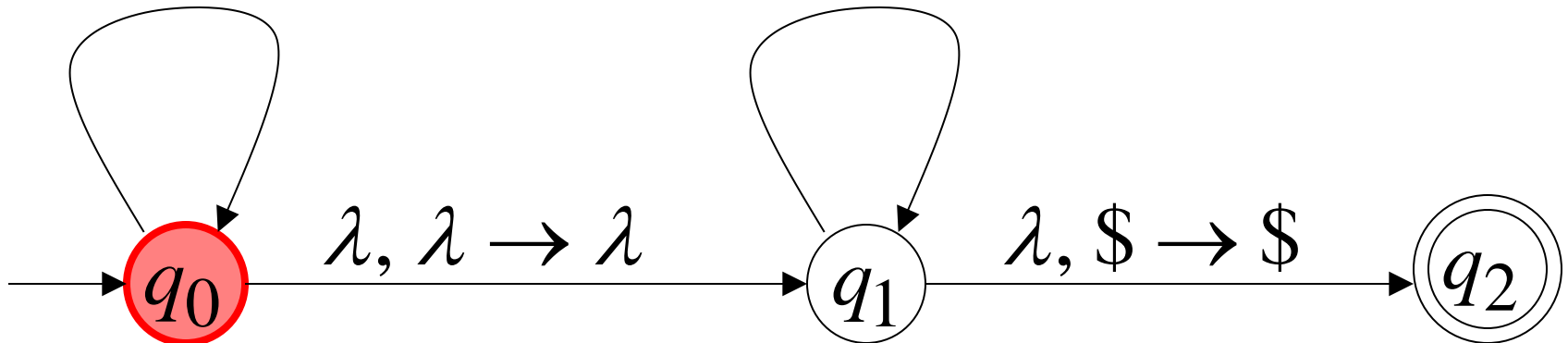
Stack

$a, \lambda \rightarrow a$

$a, a \rightarrow \lambda$

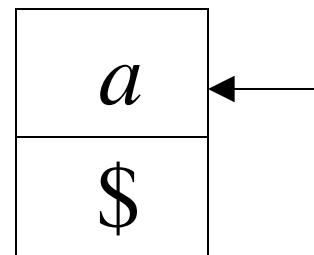
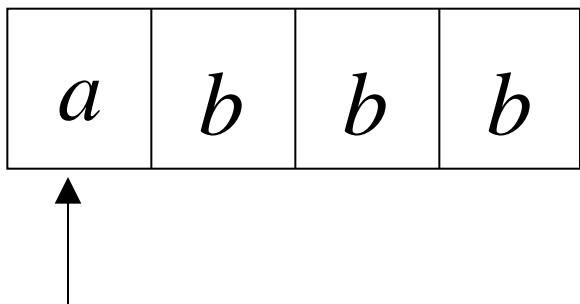
$b, \lambda \rightarrow b$

$b, b \rightarrow \lambda$



Time 1

Input



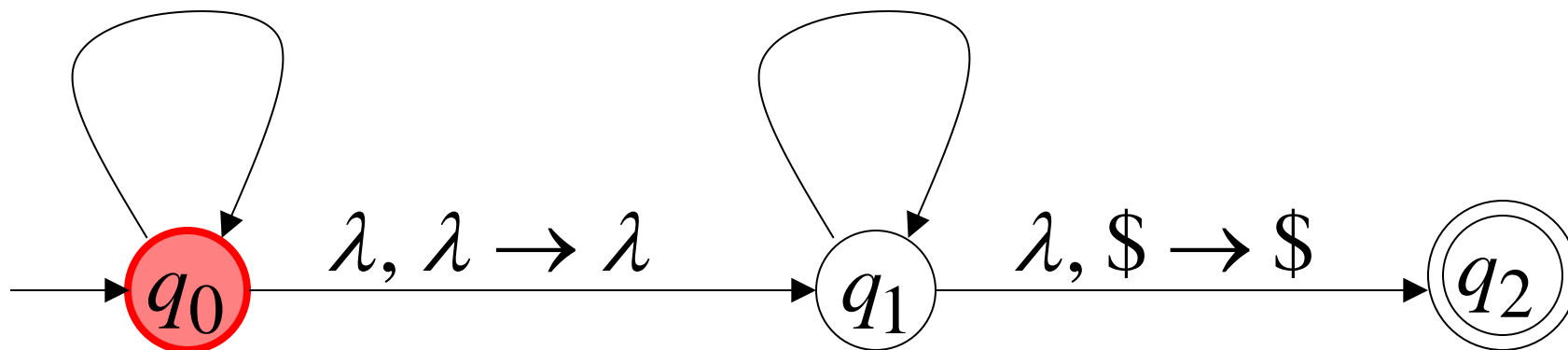
Stack

$a, \lambda \rightarrow a$

$b, \lambda \rightarrow b$

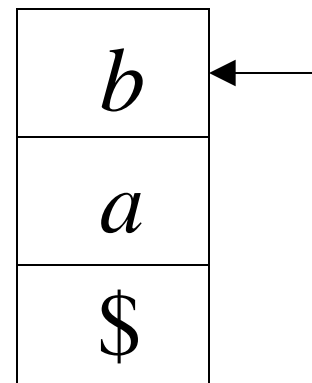
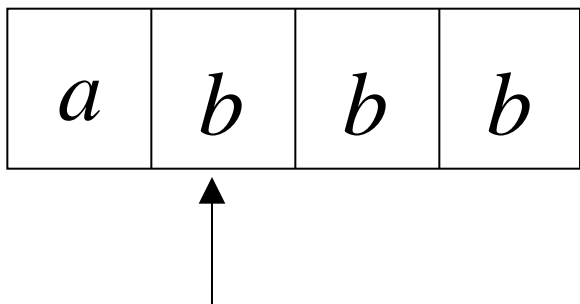
$a, a \rightarrow \lambda$

$b, b \rightarrow \lambda$



Time 2

Input



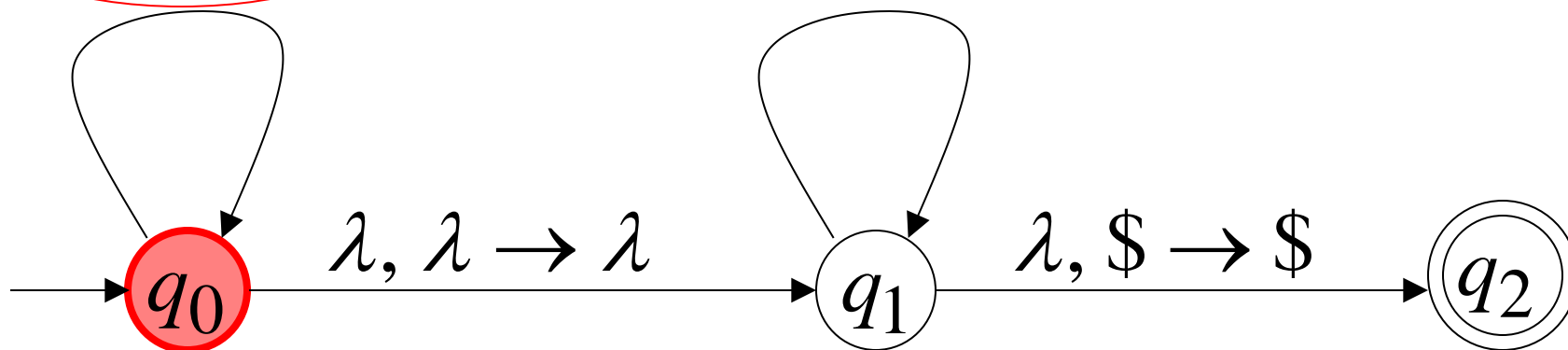
Stack

$a, \lambda \rightarrow a$

$b, \lambda \rightarrow b$

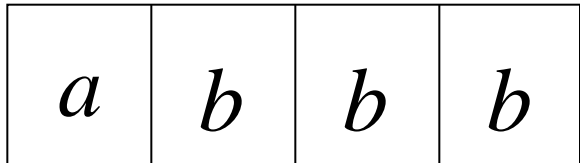
$a, a \rightarrow \lambda$

$b, b \rightarrow \lambda$

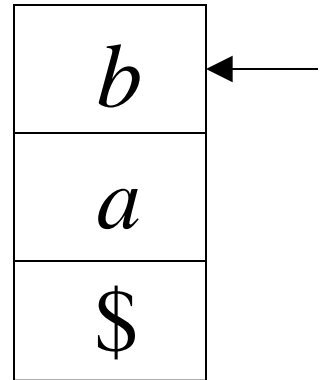


Time 3

Input



Guess the middle
of string



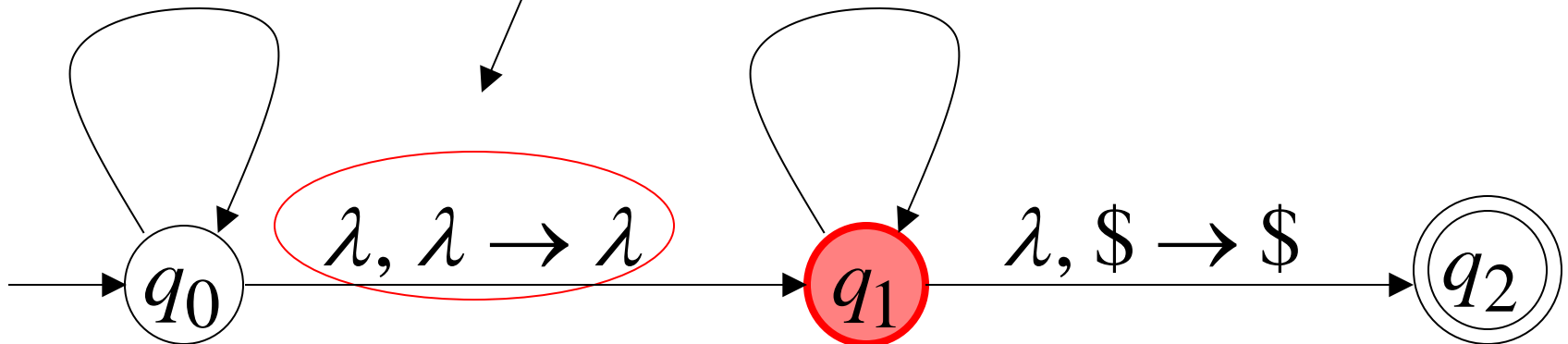
Stack

$a, \lambda \rightarrow a$

$b, \lambda \rightarrow b$

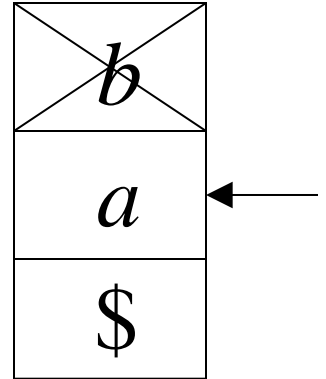
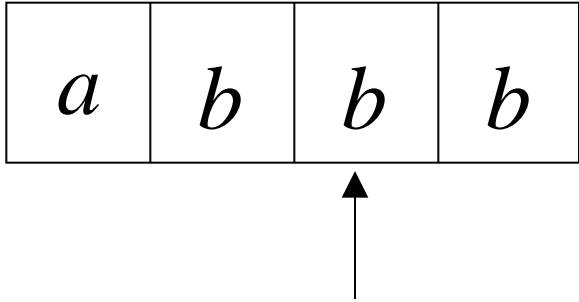
$a, a \rightarrow \lambda$

$b, b \rightarrow \lambda$



Time 4

Input



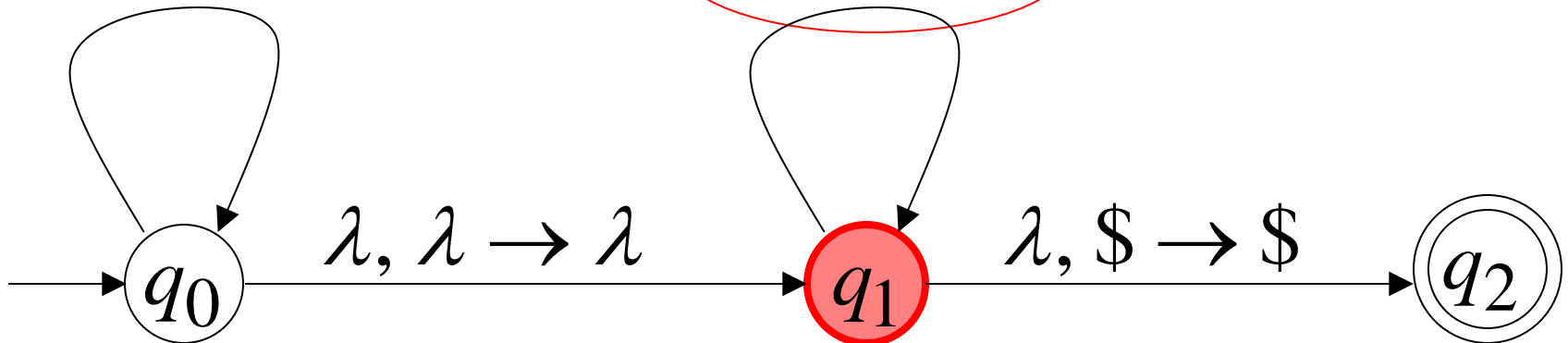
Stack

$a, \lambda \rightarrow a$

$b, \lambda \rightarrow b$

$a, a \rightarrow \lambda$

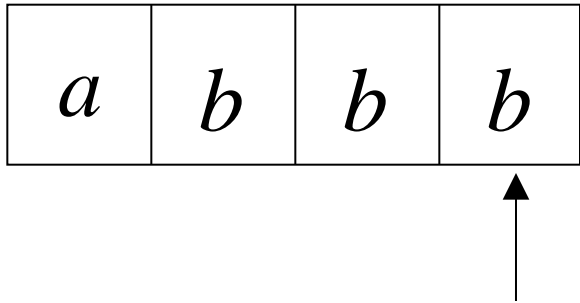
$b, b \rightarrow \lambda$



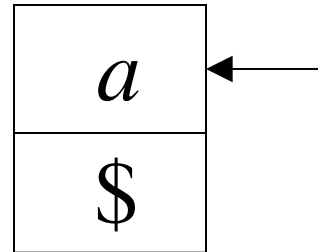
Time 5

Input

There is no possible transition.



Input is not consumed



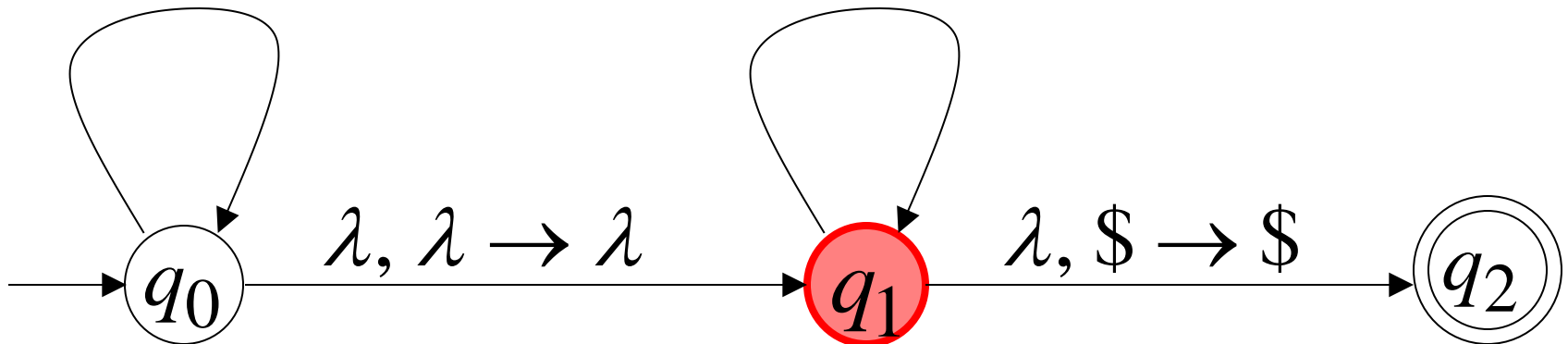
Stack

$a, \lambda \rightarrow a$

$a, a \rightarrow \lambda$

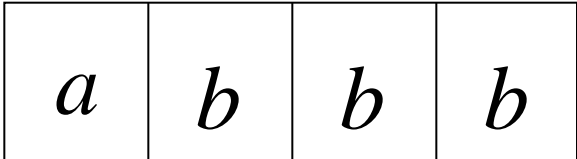
$b, \lambda \rightarrow b$

$b, b \rightarrow \lambda$

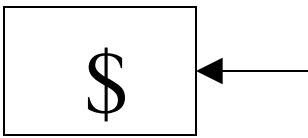


Another computation on same string:

Input



Time 0



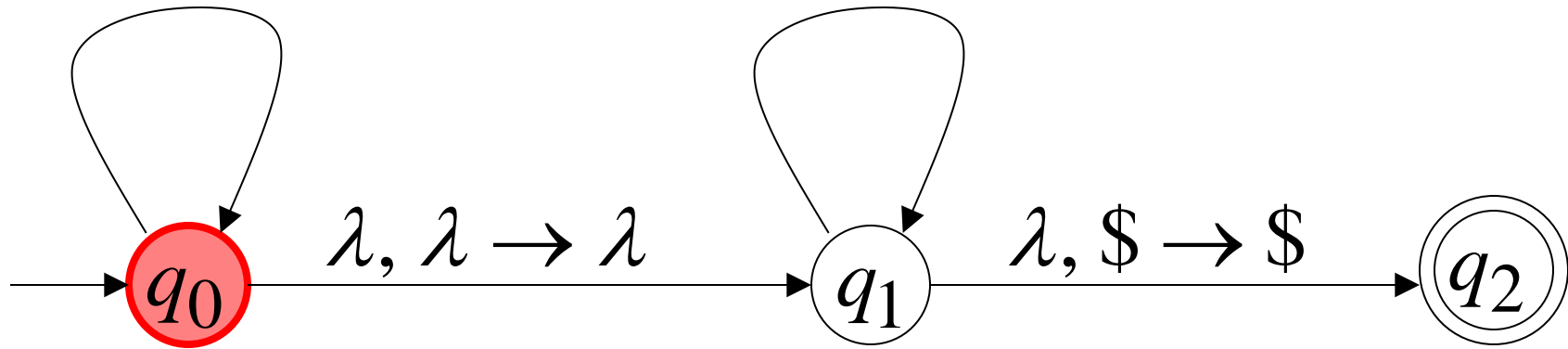
Stack

$a, \lambda \rightarrow a$

$a, a \rightarrow \lambda$

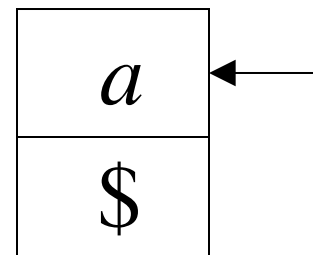
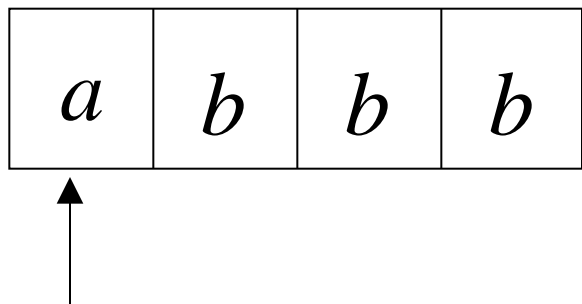
$b, \lambda \rightarrow b$

$b, b \rightarrow \lambda$



Time 1

Input



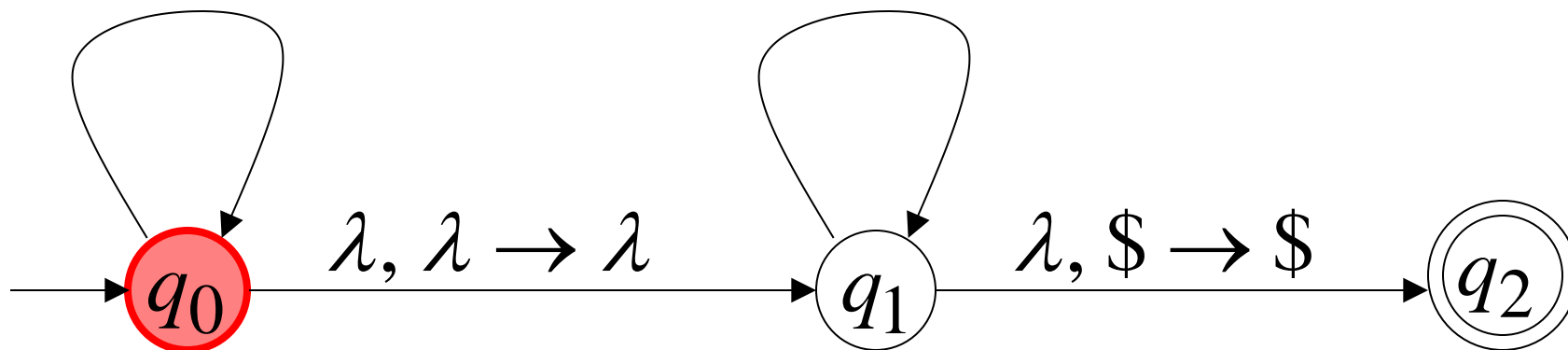
Stack

$a, \lambda \rightarrow a$

$b, \lambda \rightarrow b$

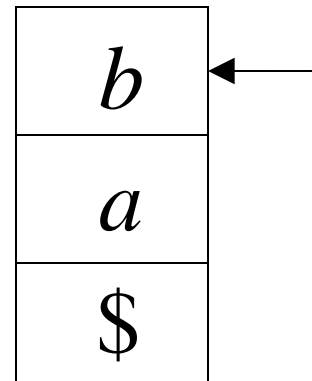
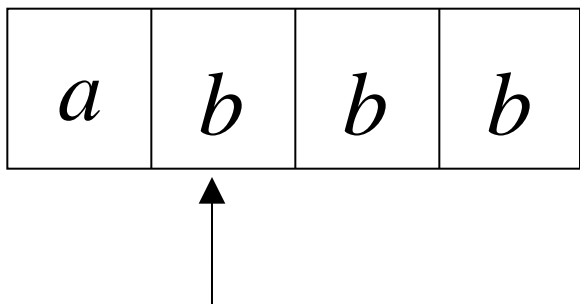
$a, a \rightarrow \lambda$

$b, b \rightarrow \lambda$



Time 2

Input



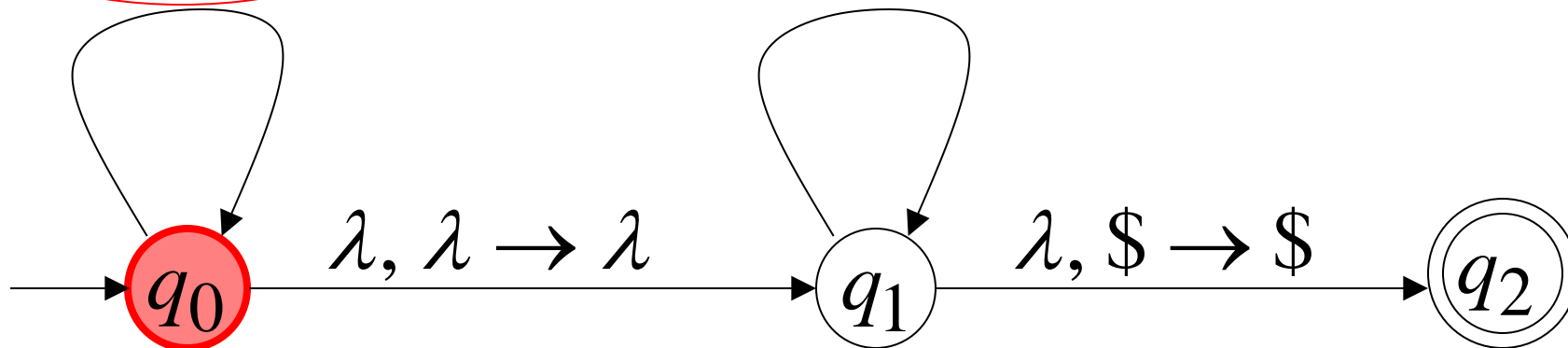
Stack

$a, \lambda \rightarrow a$

$b, \lambda \rightarrow b$

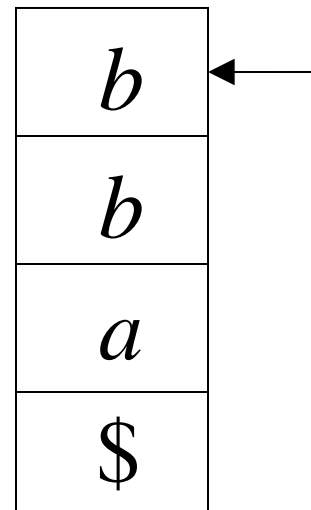
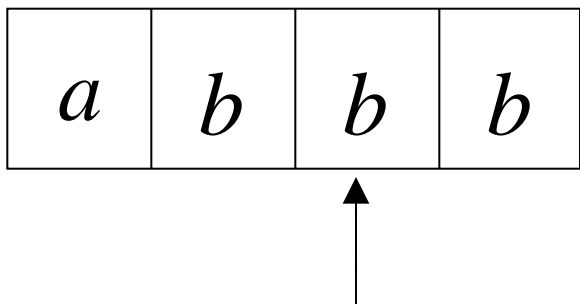
$a, a \rightarrow \lambda$

$b, b \rightarrow \lambda$



Time 3

Input



Stack

$a, \lambda \rightarrow a$

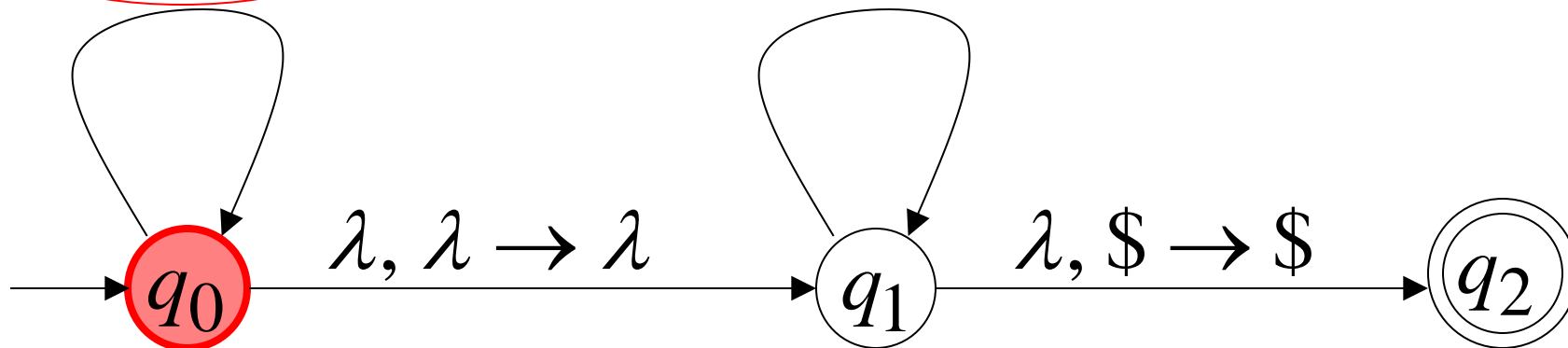
$b, \lambda \rightarrow b$

$a, a \rightarrow \lambda$

$b, b \rightarrow \lambda$

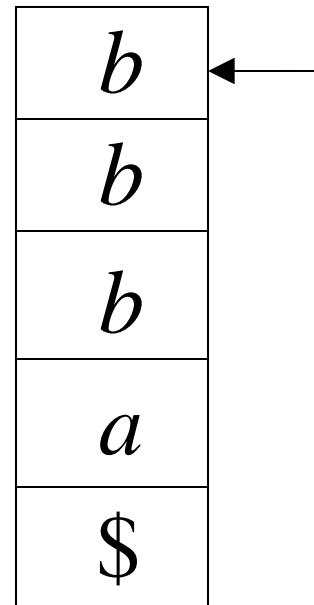
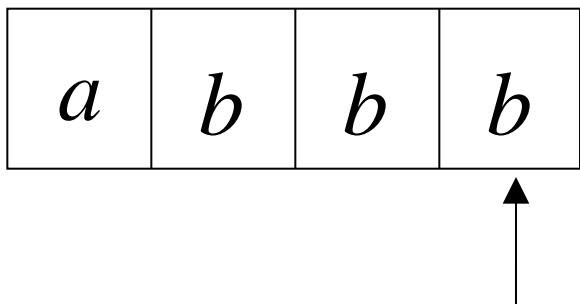
$\lambda, \lambda \rightarrow \lambda$

$\lambda, \$ \rightarrow \$$



Time 4

Input



Stack

$a, \lambda \rightarrow a$

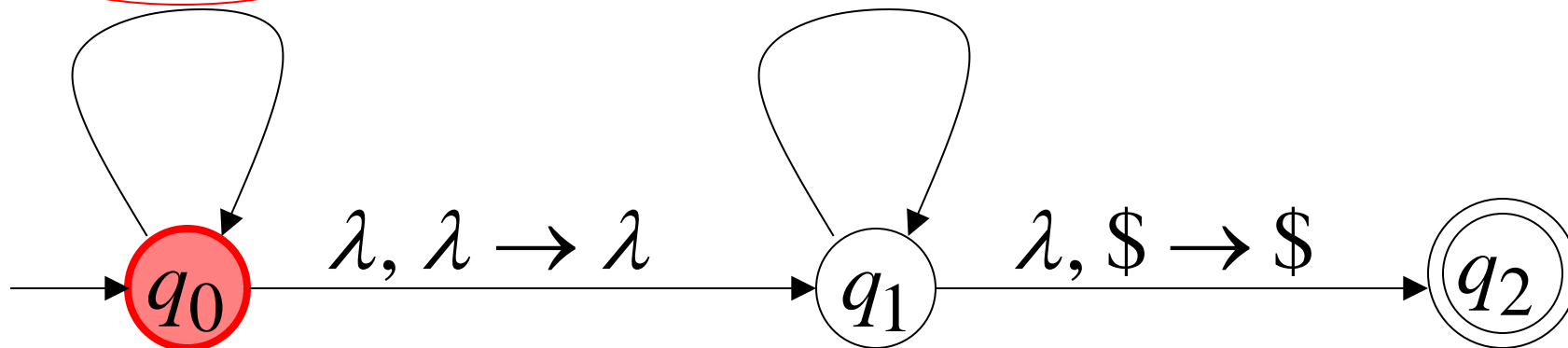
$b, \lambda \rightarrow b$

$a, a \rightarrow \lambda$

$b, b \rightarrow \lambda$

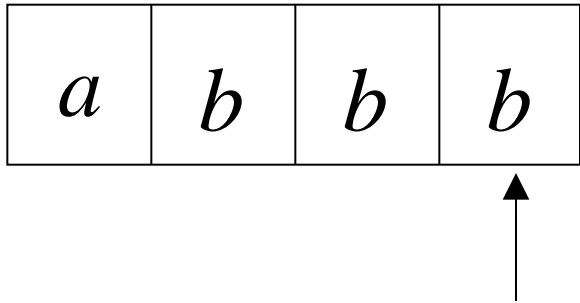
$\lambda, \lambda \rightarrow \lambda$

$\lambda, \$ \rightarrow \$$

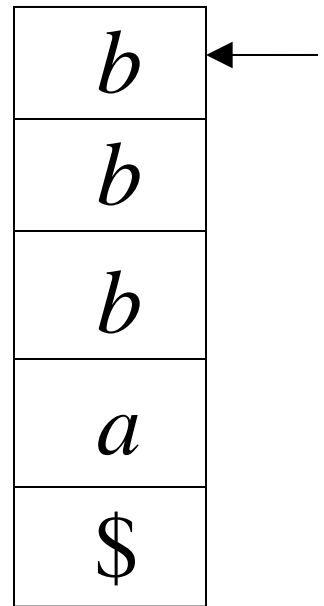


Time 5

Input



No accept state
is reached



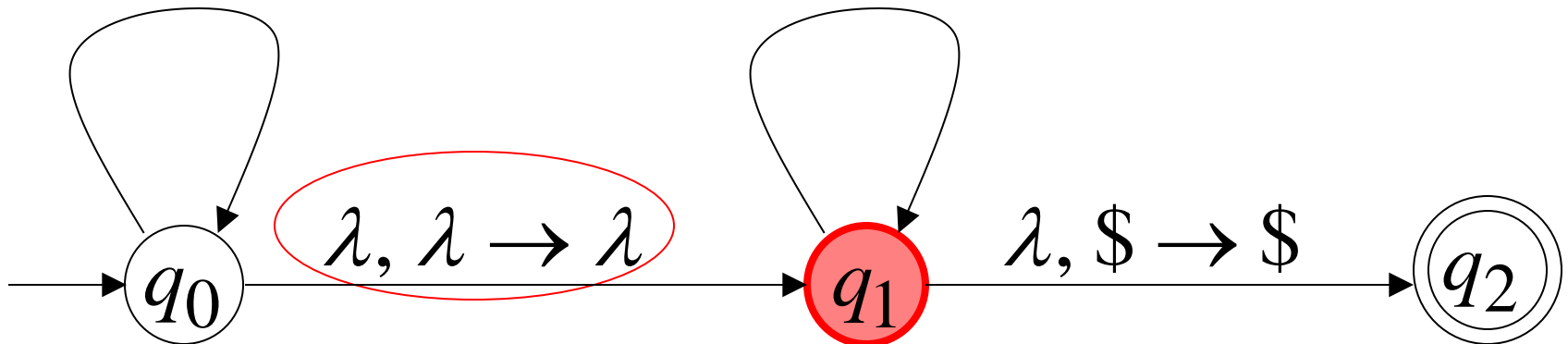
Stack

$a, \lambda \rightarrow a$

$b, \lambda \rightarrow b$

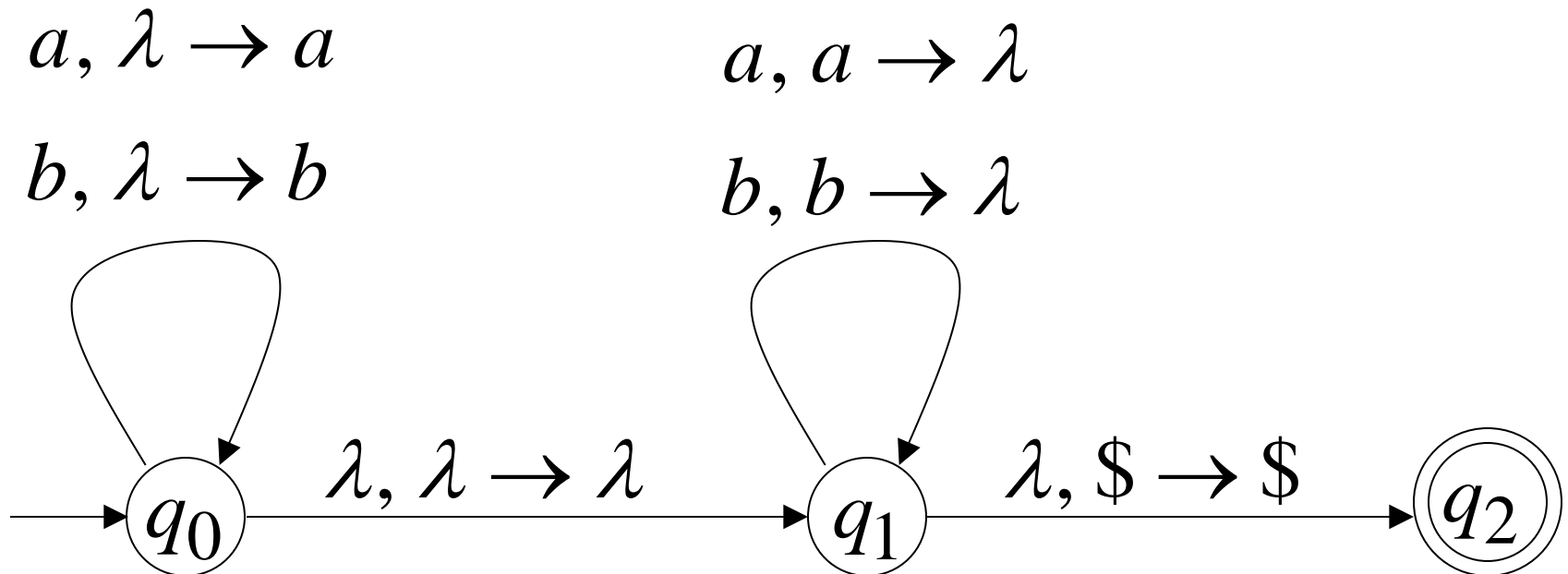
$a, a \rightarrow \lambda$

$b, b \rightarrow \lambda$

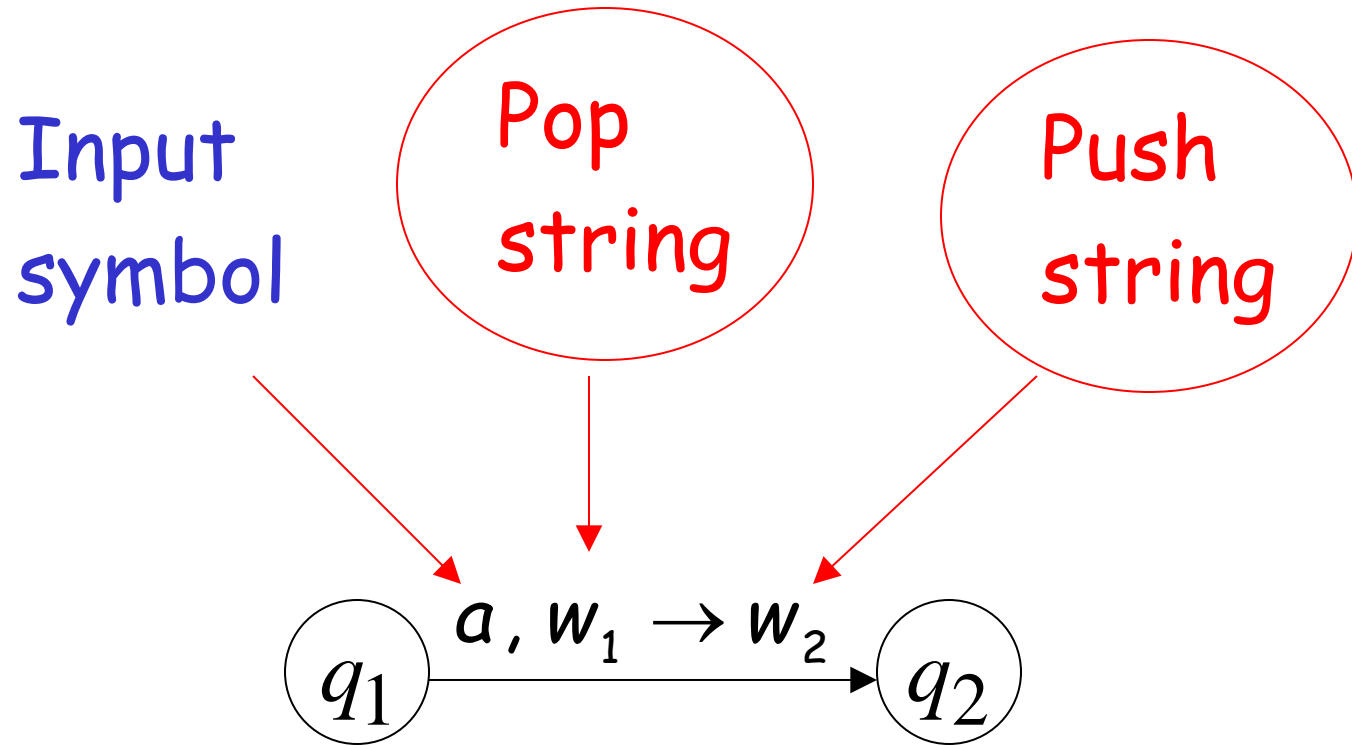


There is no computation
that accepts string $abbb$

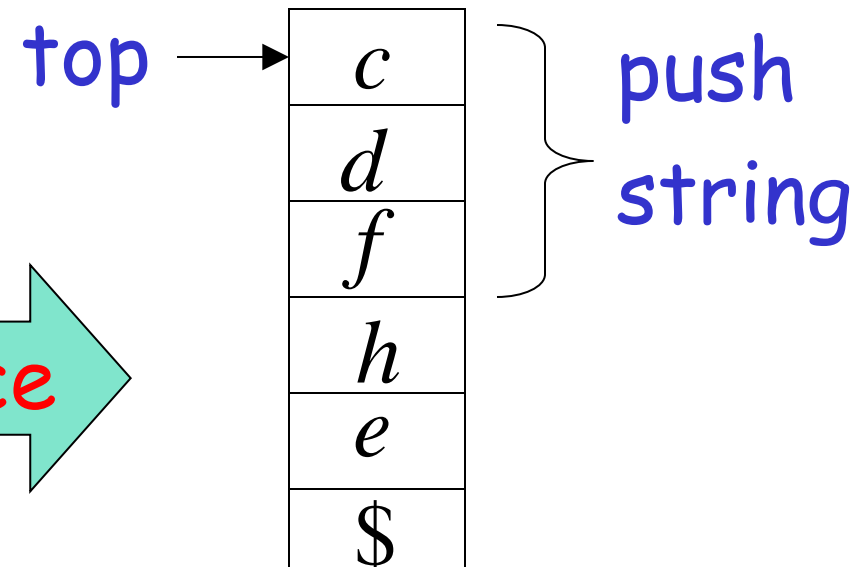
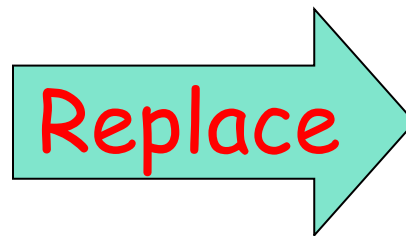
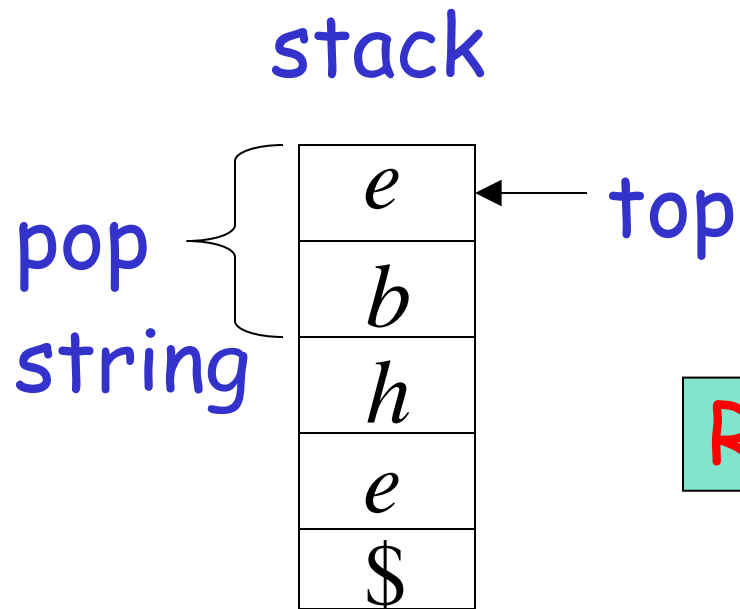
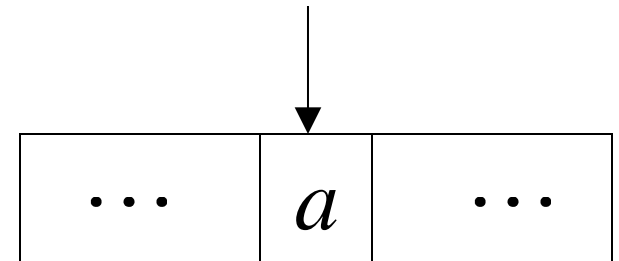
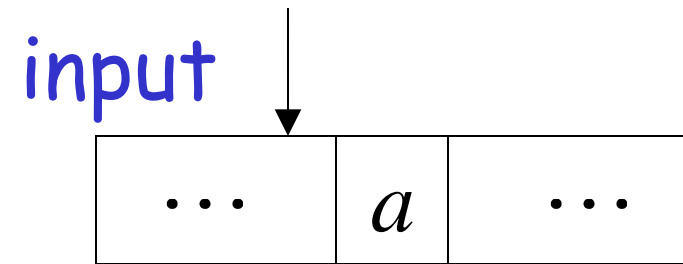
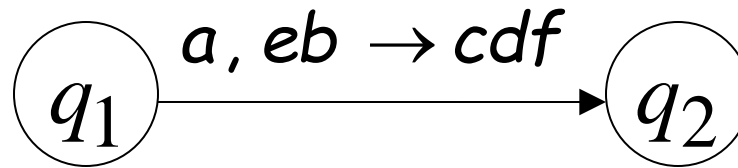
$$abbb \notin L(M)$$

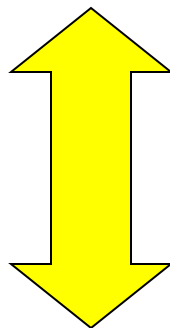
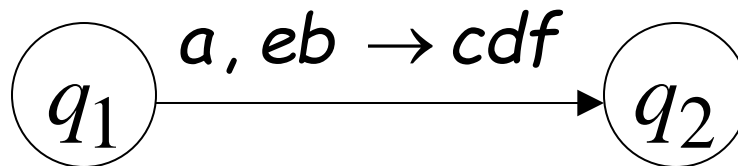


Pushing & Popping Strings



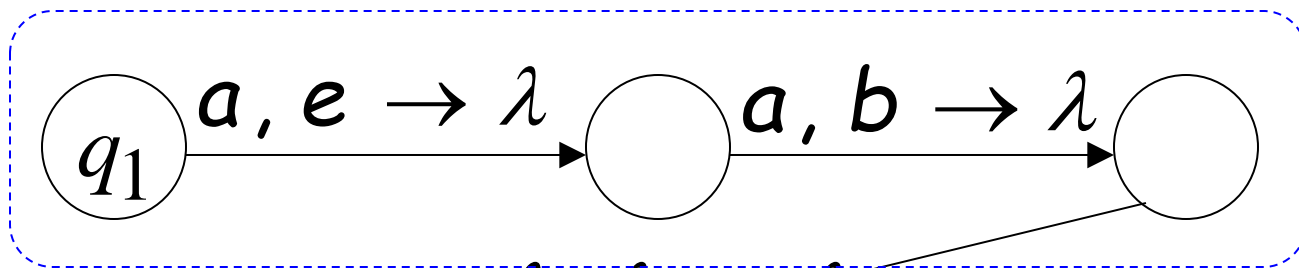
Example:





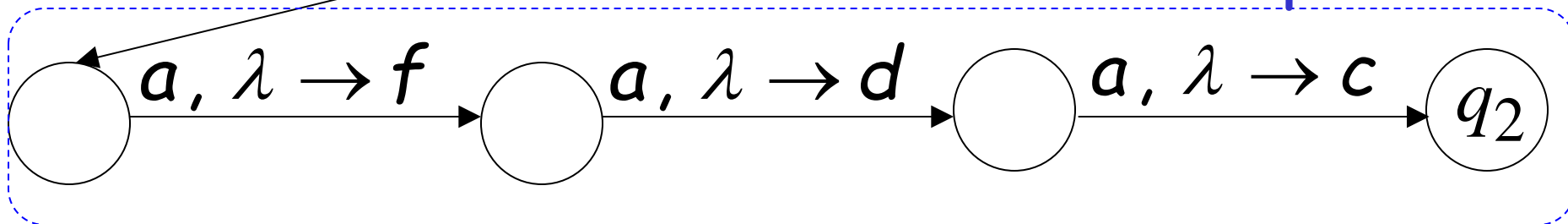
Equivalent
transitions

pop



$\lambda, \lambda \rightarrow \lambda$

push



Another PDA example

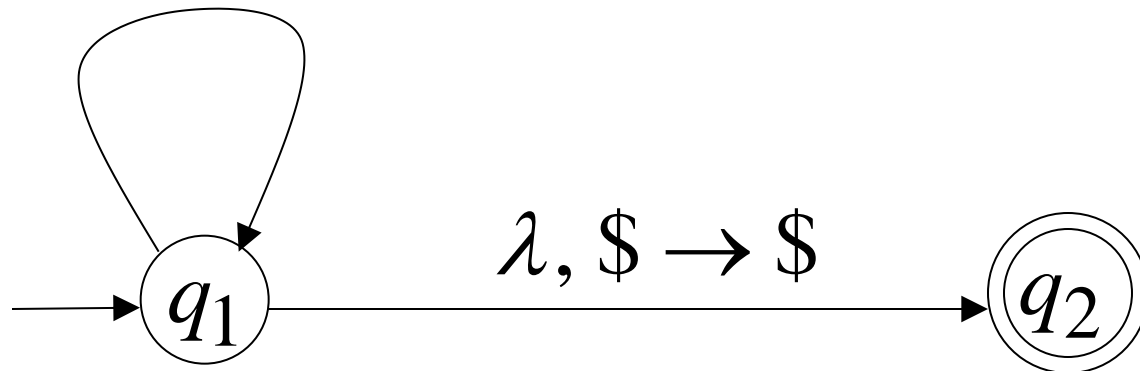
$$L(M) = \{w \in \{a,b\}^* : n_a(w) = n_b(w)\}$$

PDA M

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

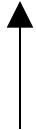
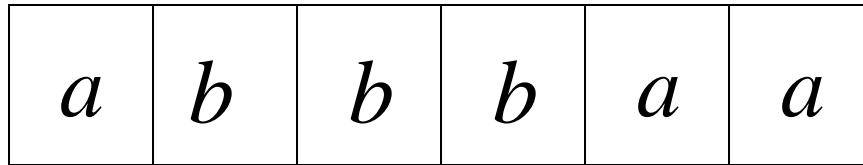
$a, 0 \rightarrow 00$ $b, 1 \rightarrow 11$

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



Execution Example: Time 0

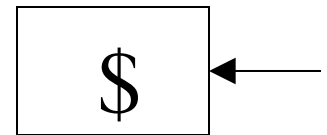
Input



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

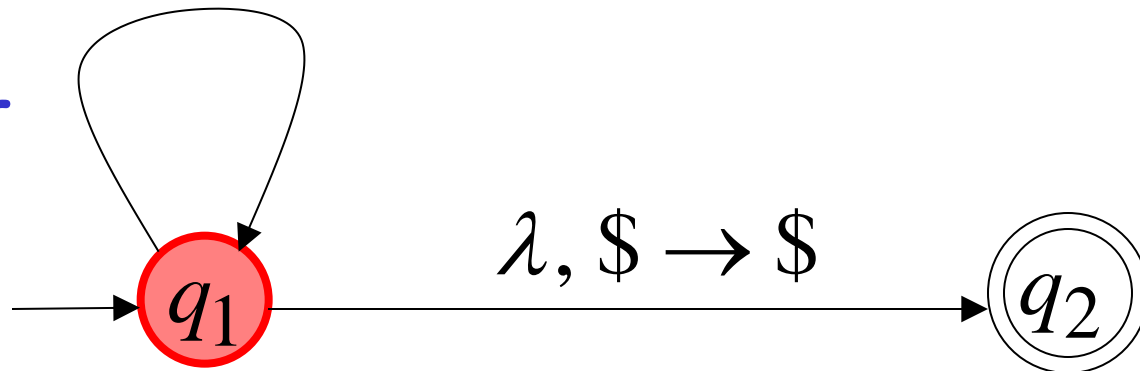
$a, 0 \rightarrow 00$ $b, 1 \rightarrow 11$

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



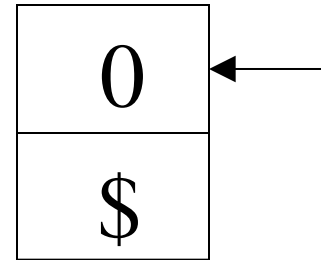
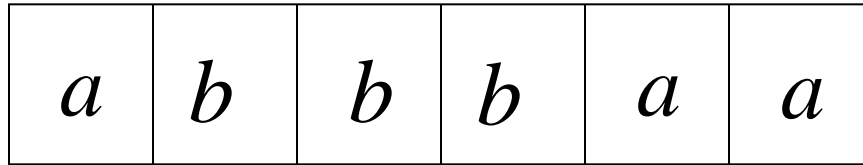
Stack

current
state



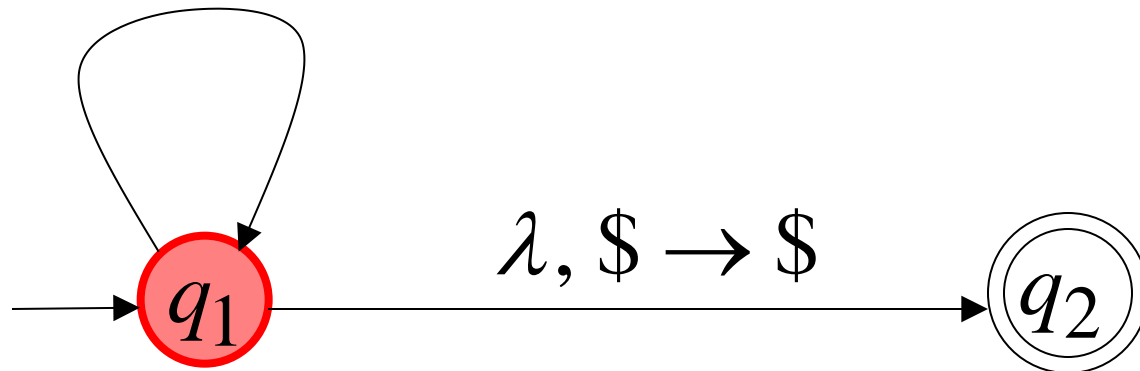
Time 1

Input



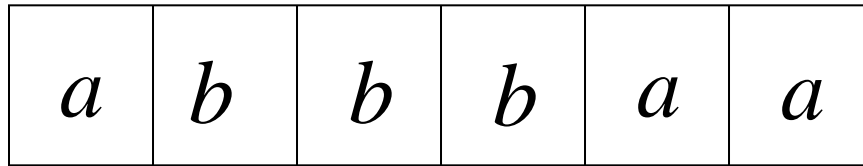
Stack

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



Time 3

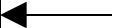
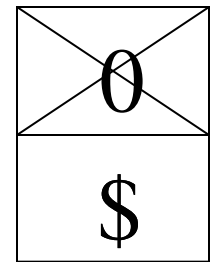
Input



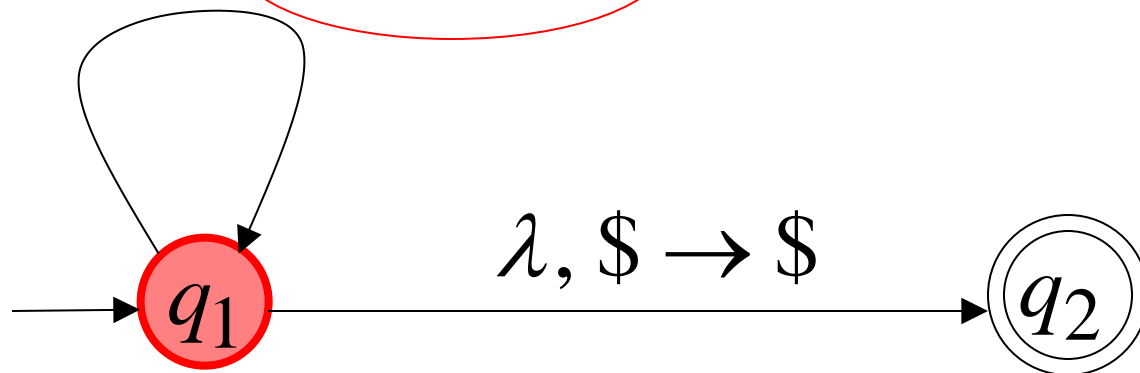
$a, \$ \rightarrow 0\$$ $b, \$ \rightarrow 1\$$

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

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

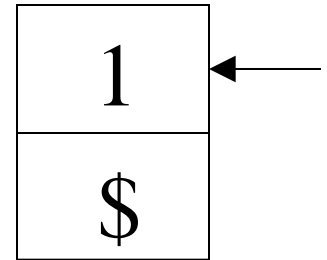
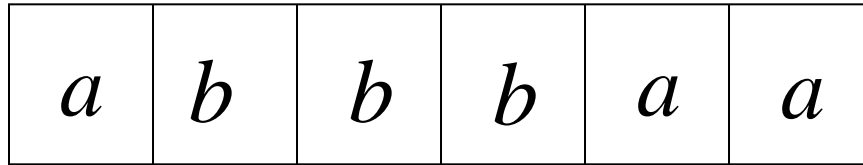


Stack



Time 4

Input

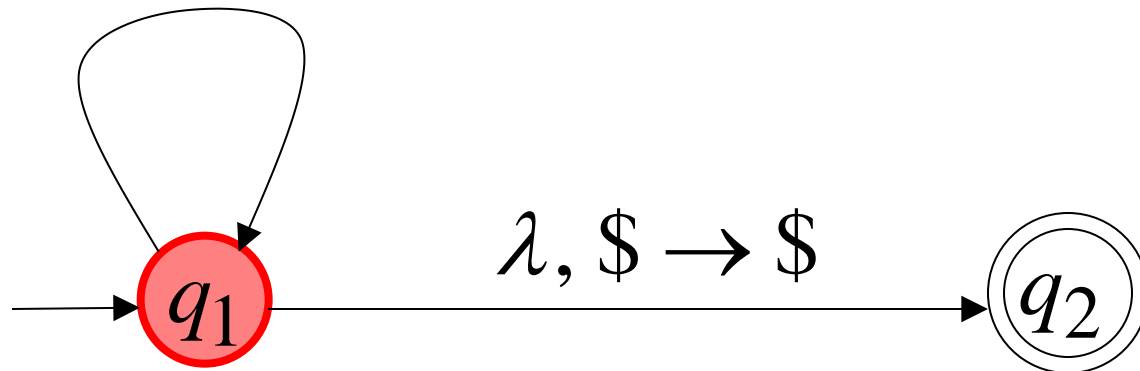


Stack

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

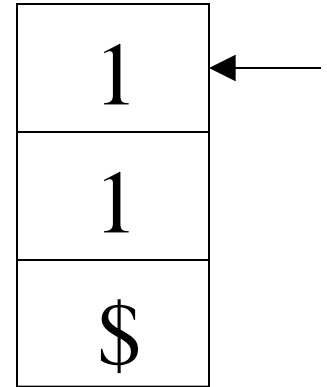
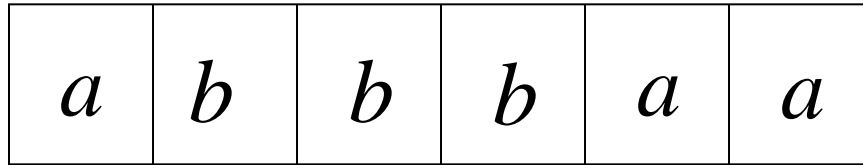
$a, 0 \rightarrow 00$ $b, 1 \rightarrow 11$

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



Time 5

Input

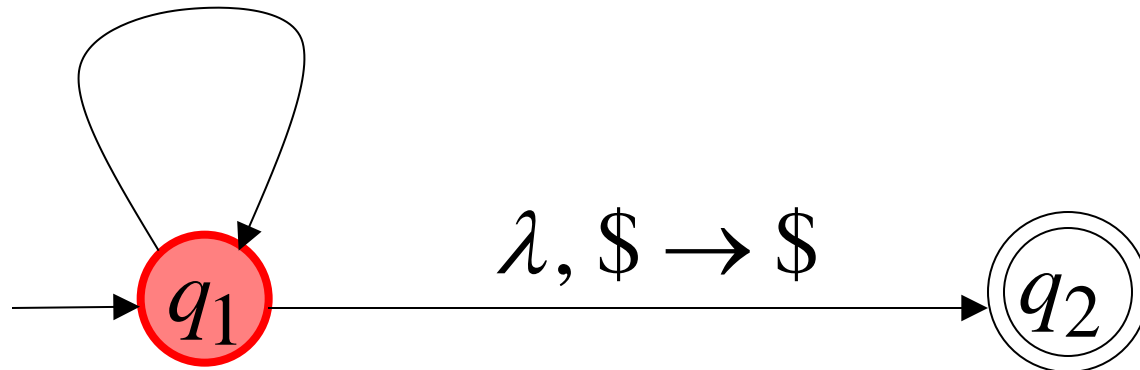


Stack

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

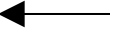
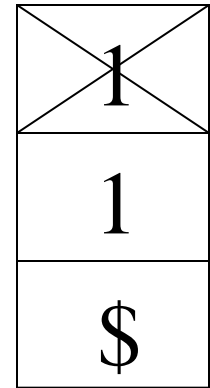
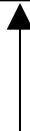
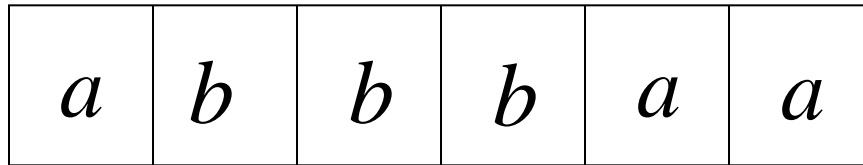
$a, 0 \rightarrow 00$ $b, 1 \rightarrow 11$

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



Time 6

Input

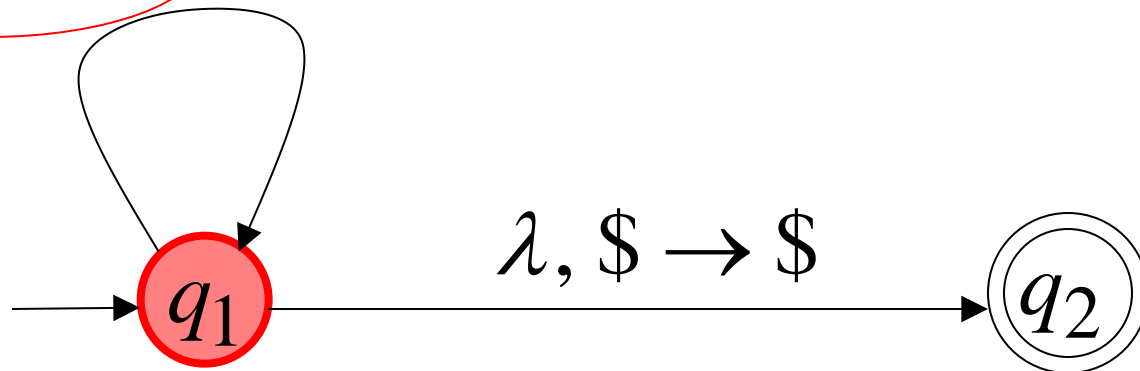


Stack

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

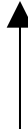
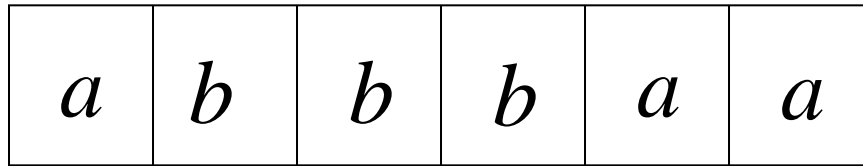
$a, 0 \rightarrow 00$ $b, 1 \rightarrow 11$

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



Time 7

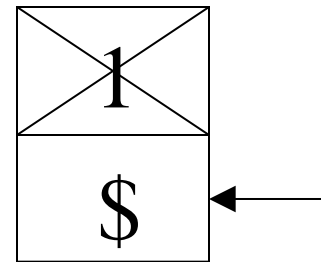
Input



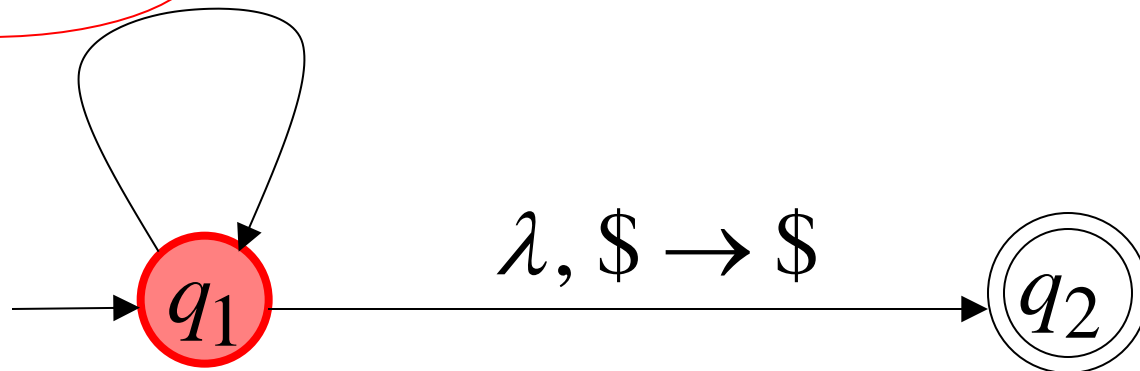
$a, \$ \rightarrow 0\$$ $b, \$ \rightarrow 1\$$

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

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

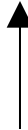
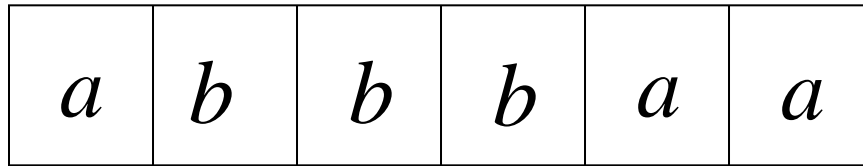


Stack



Time 8

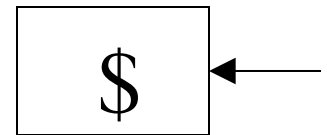
Input



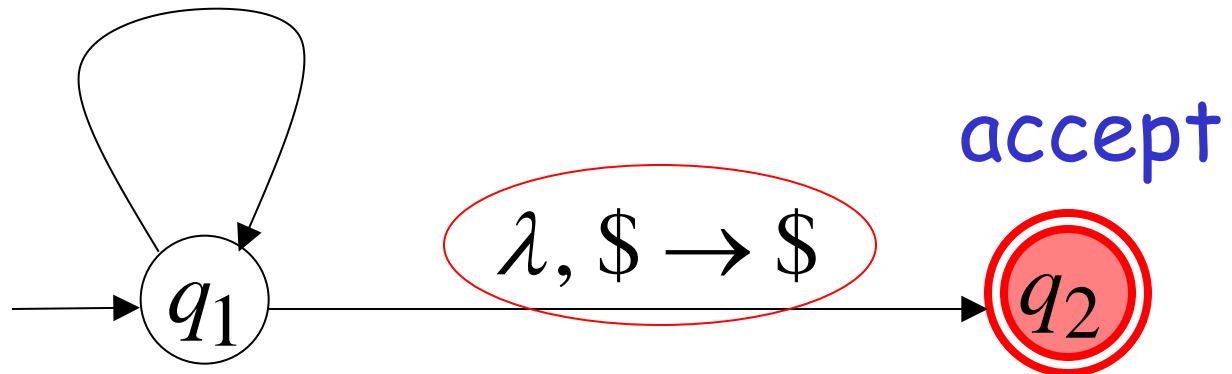
$a, \$ \rightarrow 0\$$ $b, \$ \rightarrow 1\$$

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

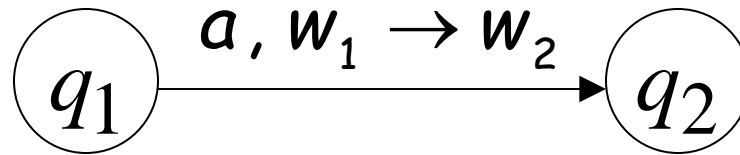
$a, 1 \rightarrow \lambda$ $b, 0 \rightarrow \lambda$



Stack

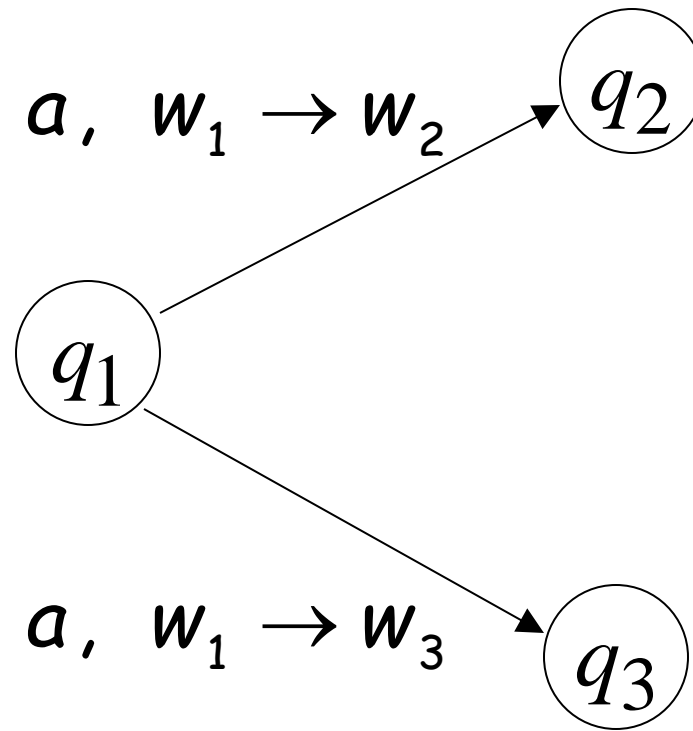


Formalities for PDAs



Transition function:

$$\delta(q_1, a, w_1) = \{(q_2, w_2)\}$$



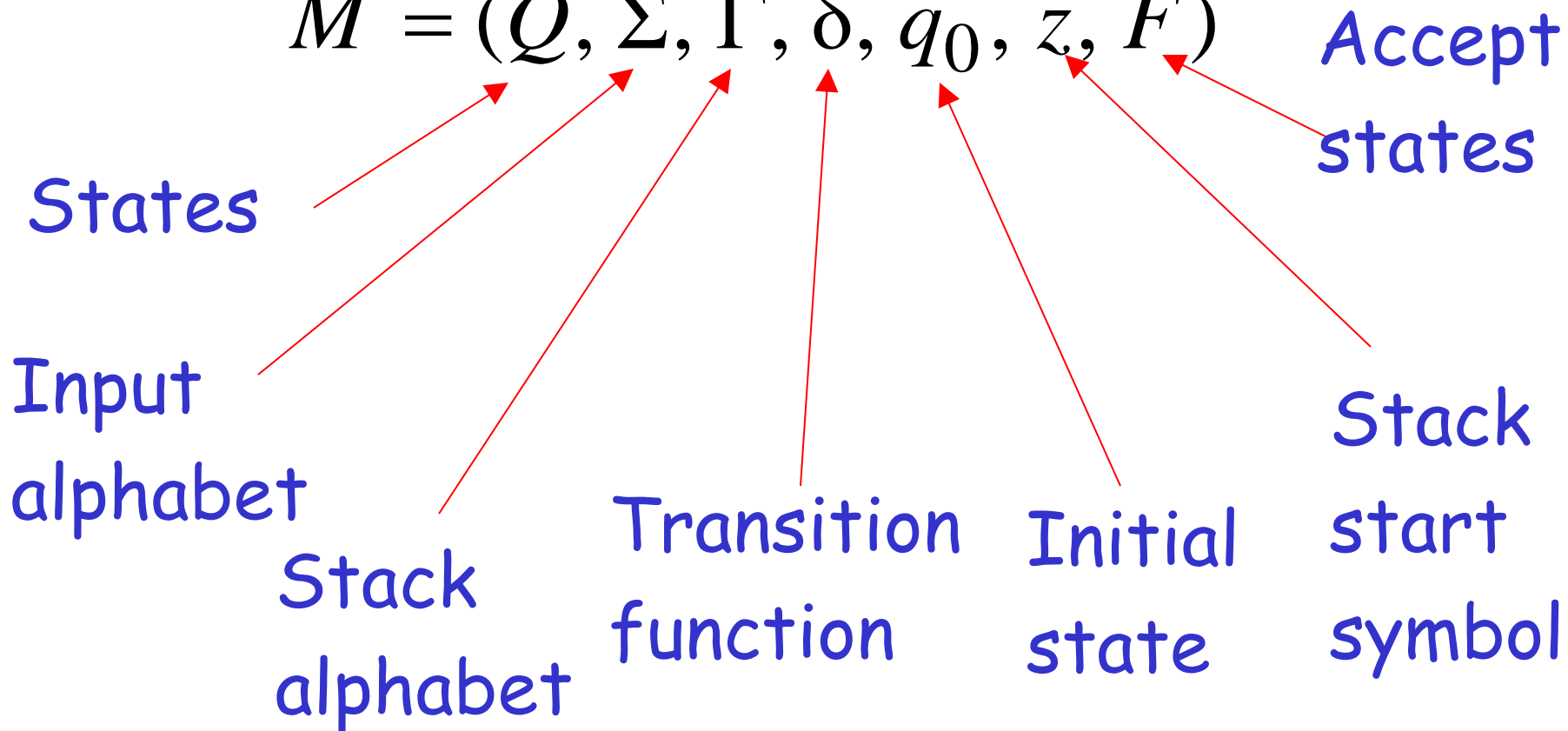
Transition function:

$$\delta(q_1, a, w_1) = \{(q_2, w_2), (q_3, w_3)\}$$

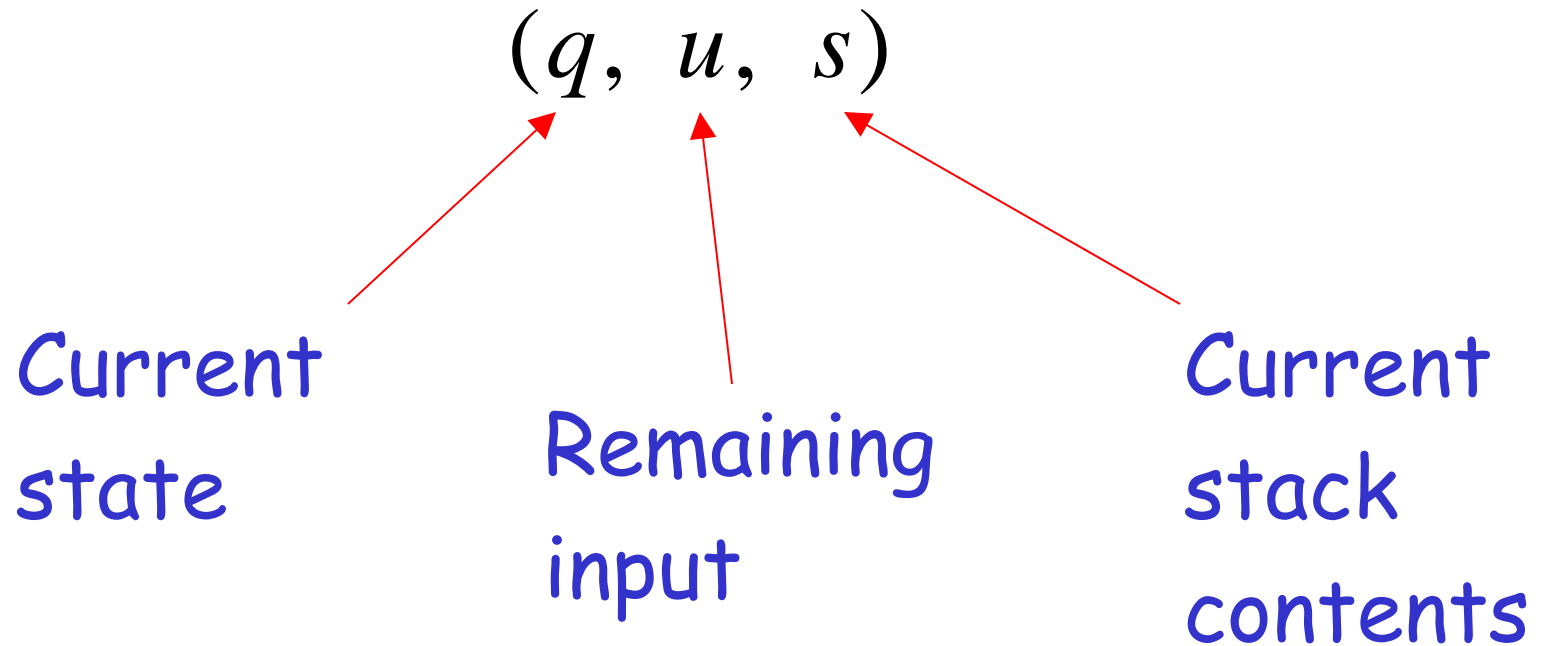
Formal Definition

Pushdown Automaton (PDA)

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



Instantaneous Description



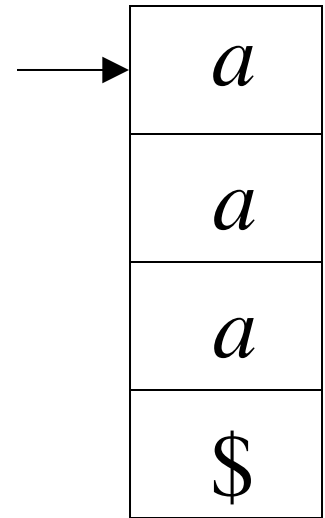
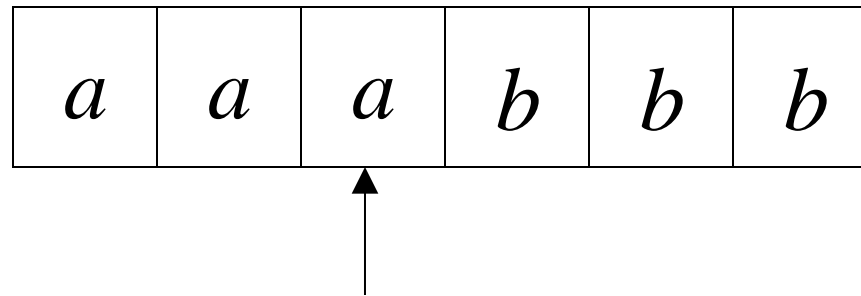
Example:

Instantaneous Description

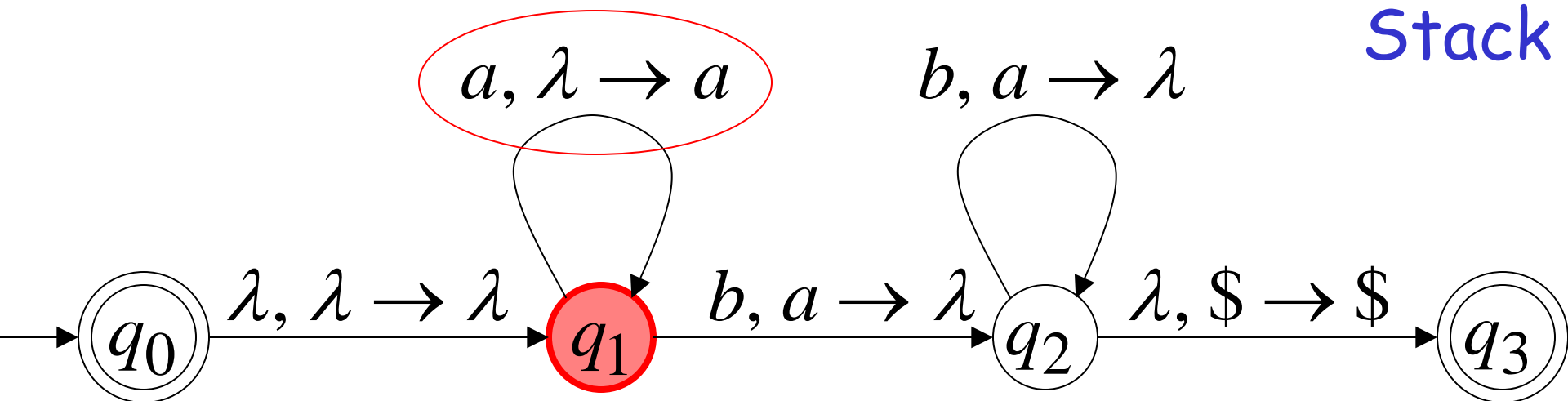
$(q_1, bbb, aaa\$)$

Time 4:

Input



Stack



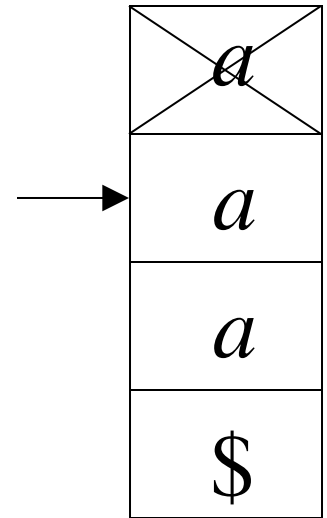
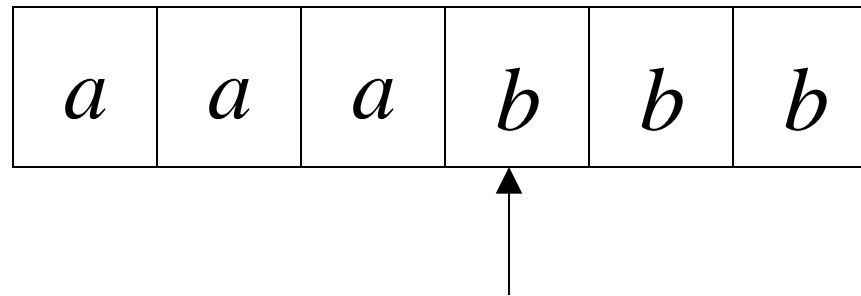
Example:

Instantaneous Description

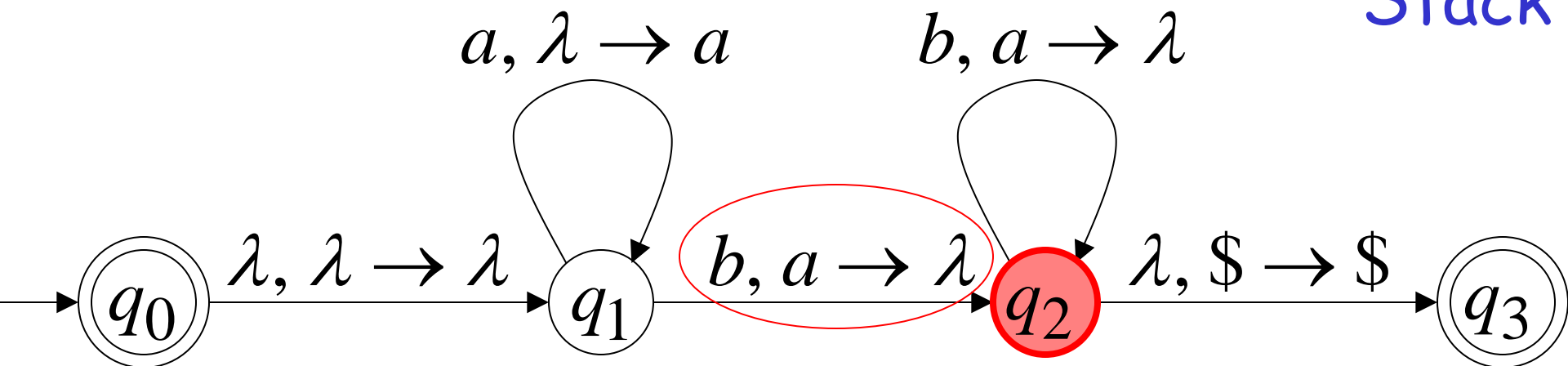
$(q_2, bb, aa\$)$

Time 5:

Input



Stack



We write:

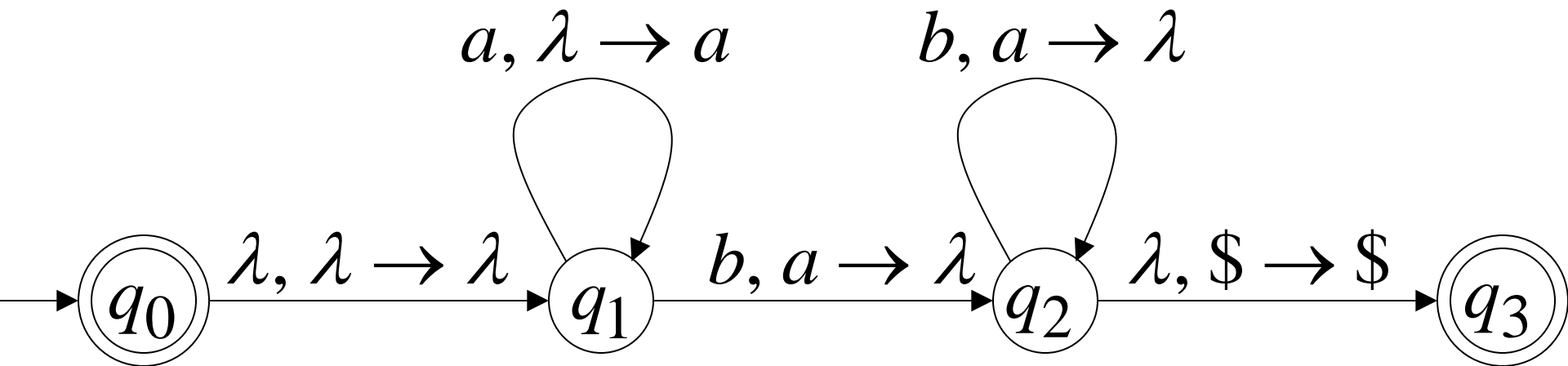
$$(q_1, bbb, aaa\$) \succ (q_2, bb, aa\$)$$

Time 4

Time 5

A computation:

$(q_0, aaabbbb, \$) \succ (q_1, aaabbbb, \$) \succ$
 $(q_1, aabbbb, a\$) \succ (q_1, abbbb, aa\$) \succ (q_1, bbbb, aaa\$) \succ$
 $(q_2, bb, aa\$) \succ (q_2, b, a\$) \succ (q_2, \lambda, \$) \succ (q_3, \lambda, \$)$



$$\begin{aligned}
 & (q_0, aaabbbb, \$) \succ (q_1, aaabbbb, \$) \succ \\
 & (q_1, aabbbb, a\$) \succ (q_1, abbbb, aa\$) \succ (q_1, bbb, aaa\$) \succ \\
 & (q_2, bb, aa\$) \succ (q_2, b, a\$) \succ (q_2, \lambda, \$) \succ (q_3, \lambda, \$)
 \end{aligned}$$

For convenience we write:

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

Language of PDA

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

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

Initial state



Accept state



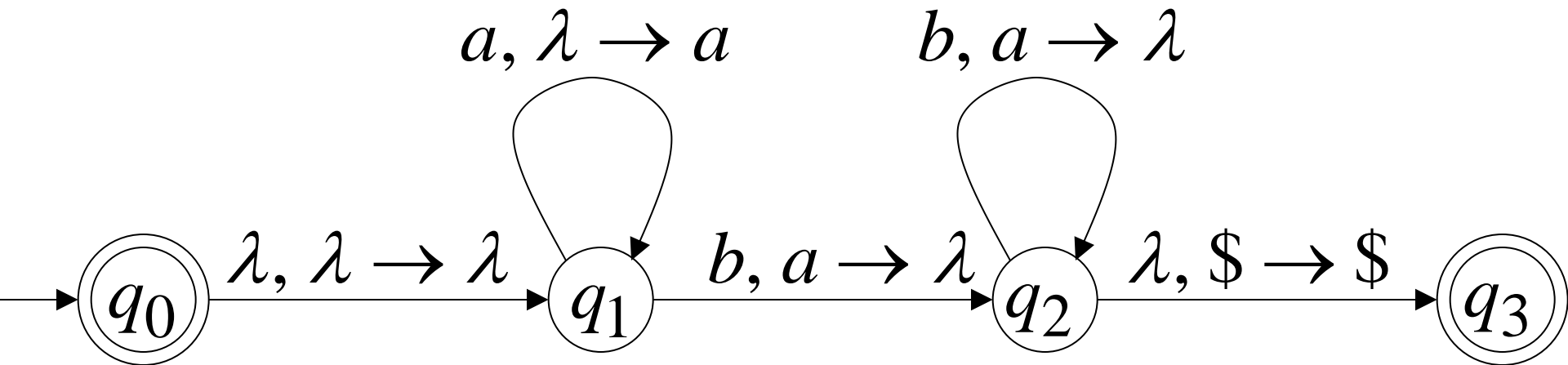
Example:

$$(q_0, aaabbbb, \$) \stackrel{*}{\succ} (q_3, \lambda, \$)$$



$$aaabbbb \in L(M)$$

PDA M :

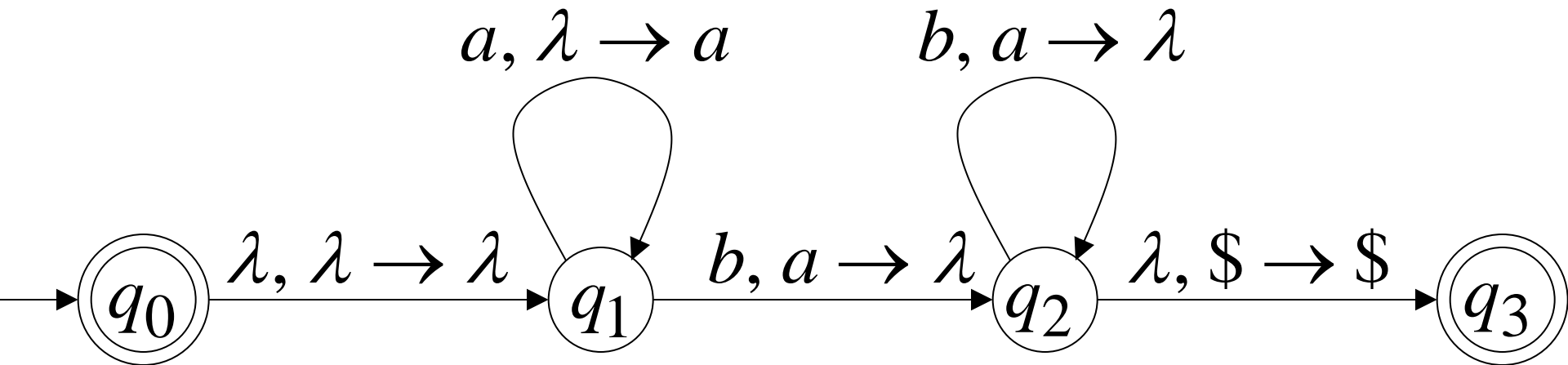


$$(q_0, a^n b^n, \$) \stackrel{*}{\succ} (q_3, \lambda, \$)$$



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

PDA M :



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

PDA M :

