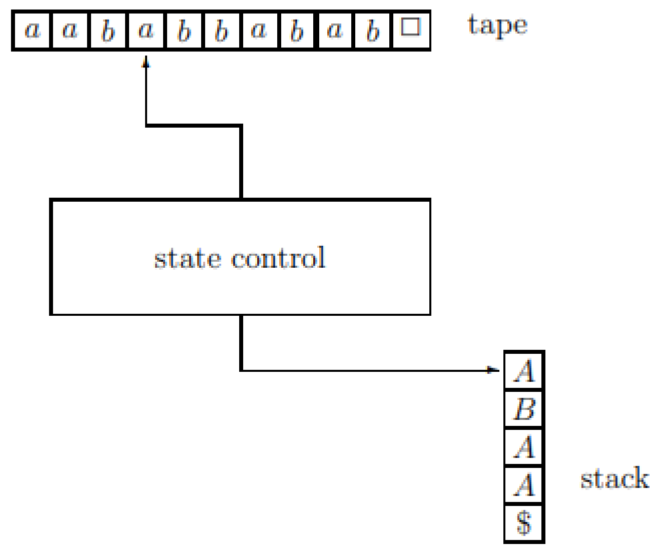
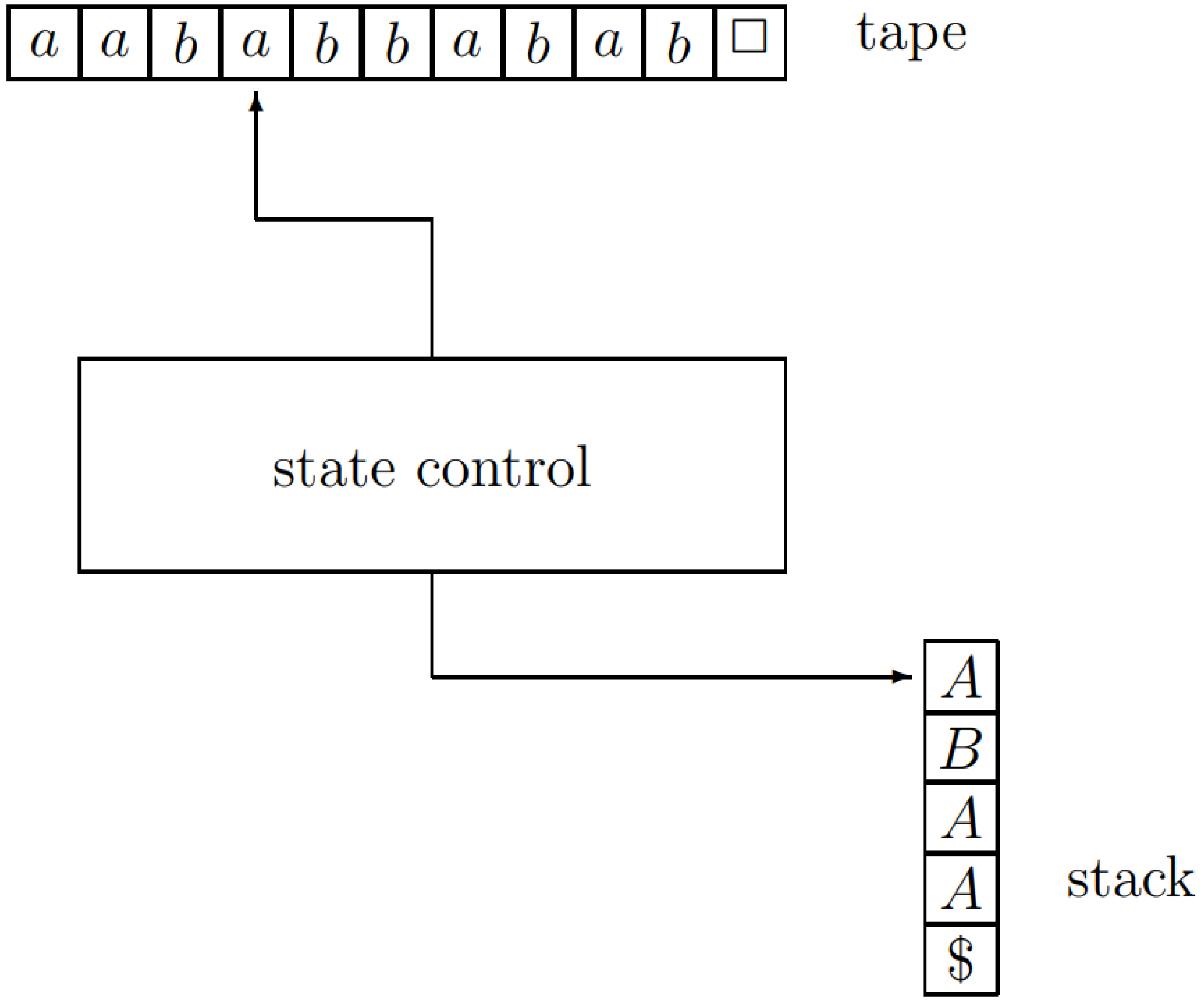


Theory of Computation

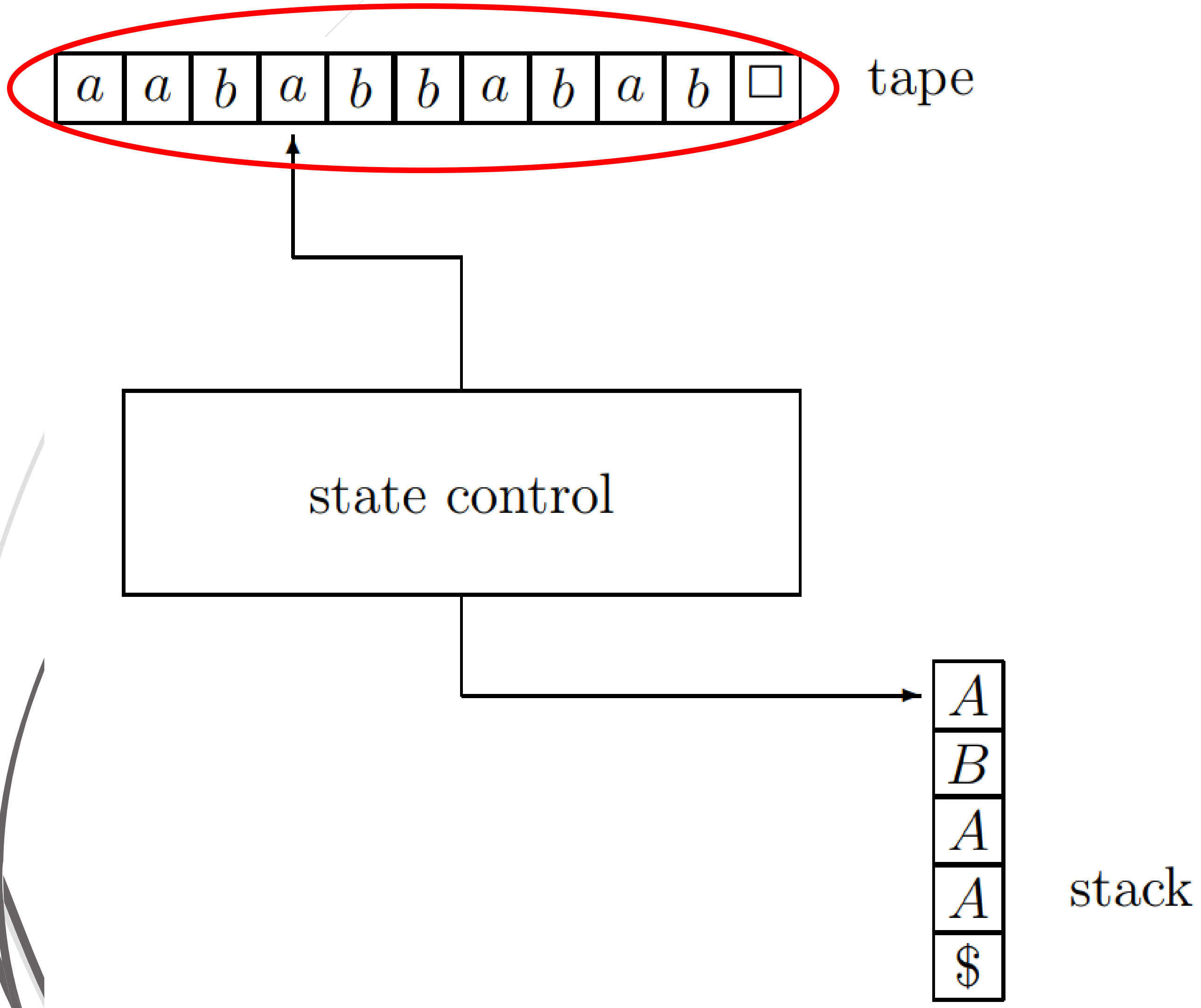
Lesson 9

Push-Down Automata

Push Down Automata

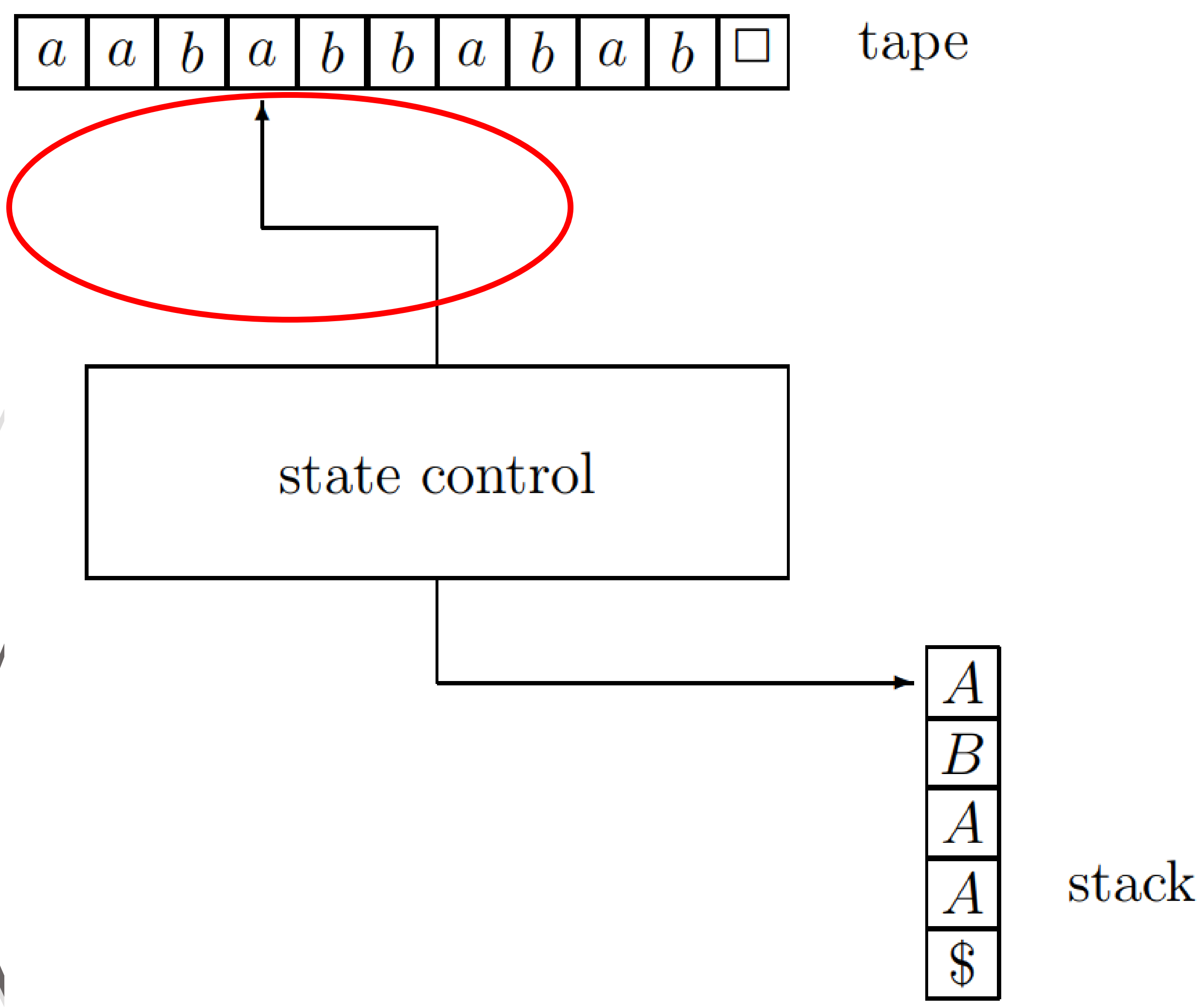


Push Down Automata



1. There is a tape which is divided into cells. Each cell stores a symbol belonging to a finite set Σ, called the tape alphabet. There is a special symbol # that is not contained in Σ; this symbol is called the blank symbol. If a cell contains #, then this means that the cell is actually empty.

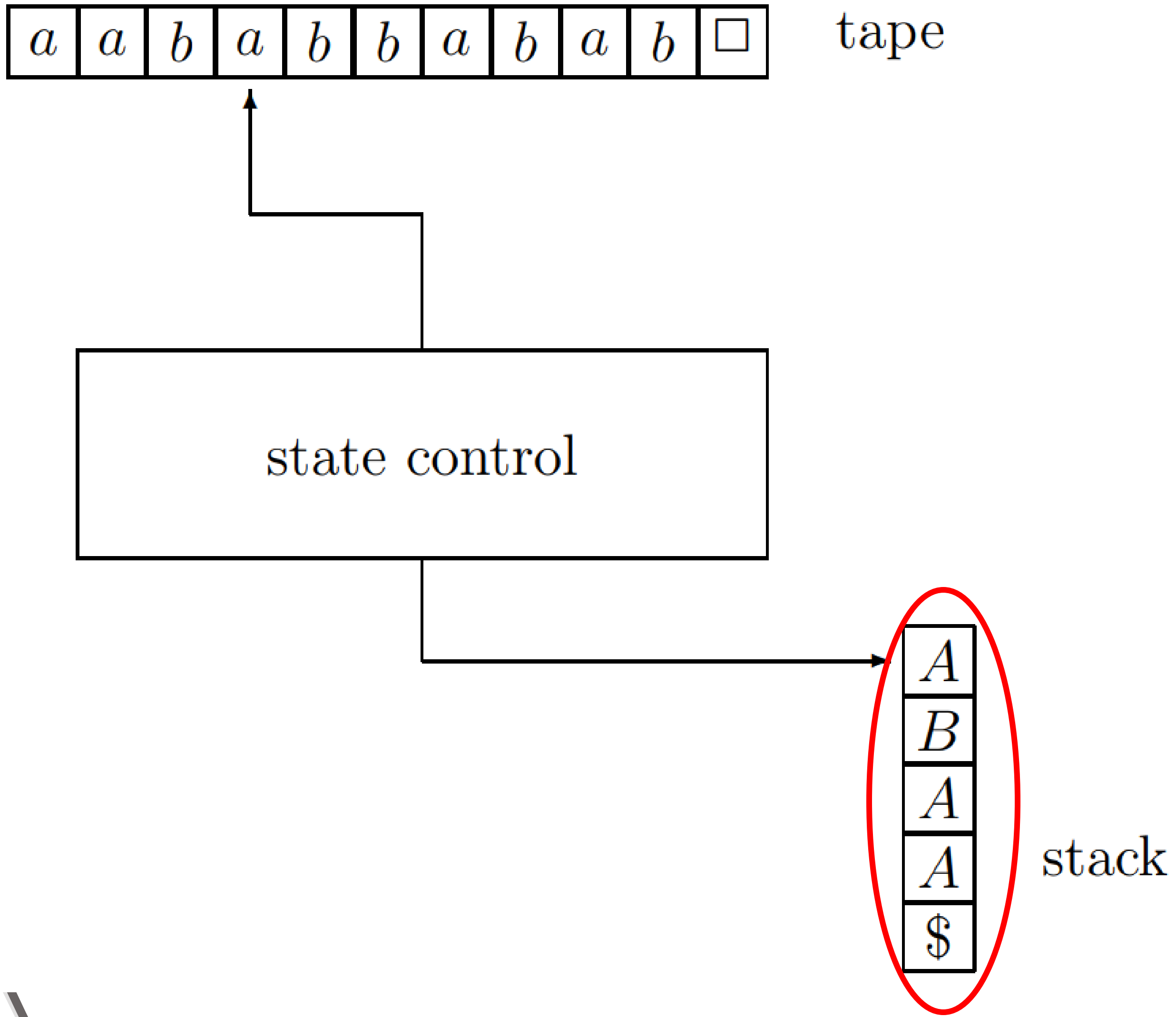
Push Down Automata



2. There is a tape head which can move along the tape, one cell to the right per move.

This tape head also read the cell it currently scans.

Push Down Automata

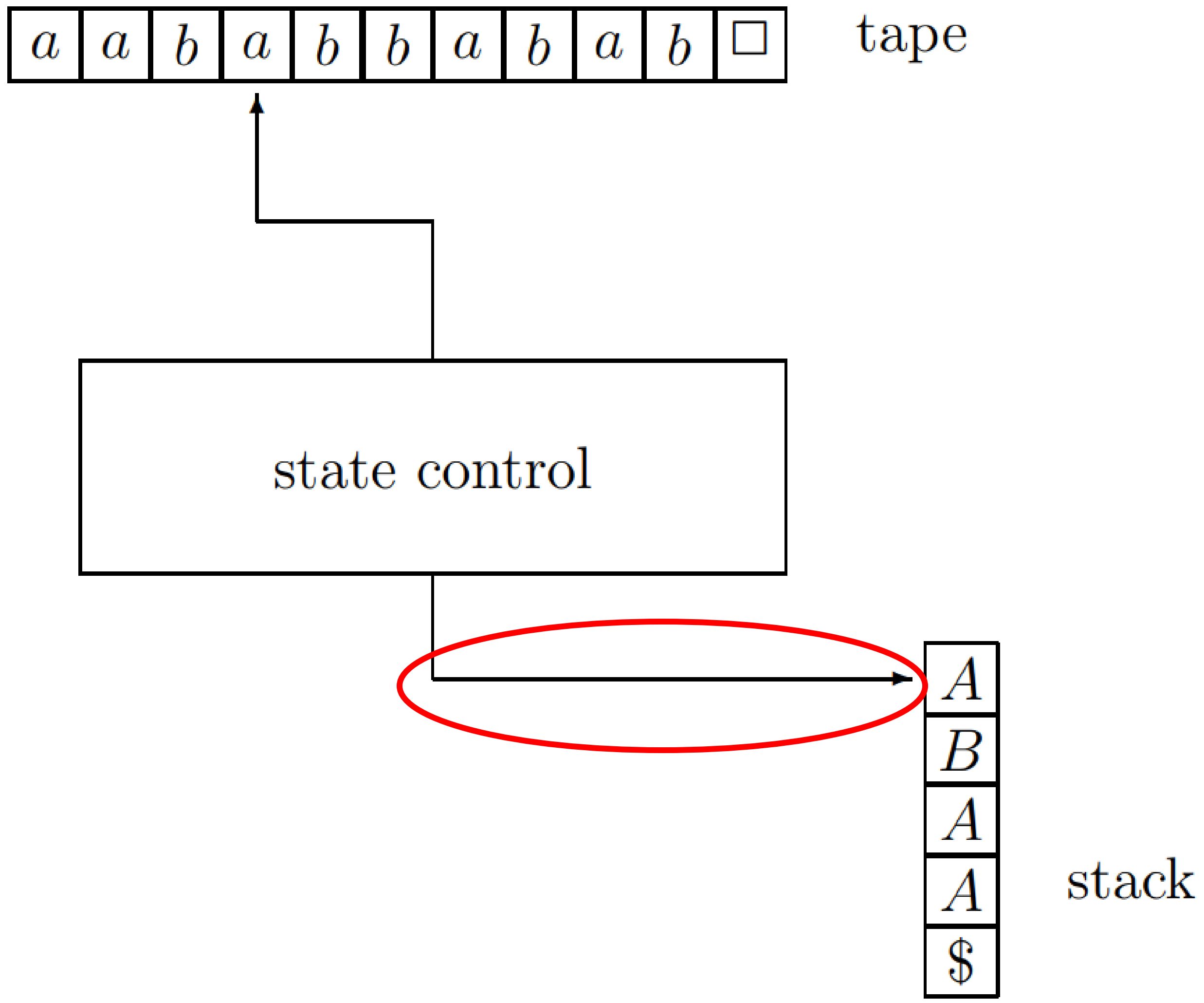


3. There is a stack. So we can store something now.

But we can use only symbols defined in Γ, called the stack alphabet.

This set contains a special symbol $.

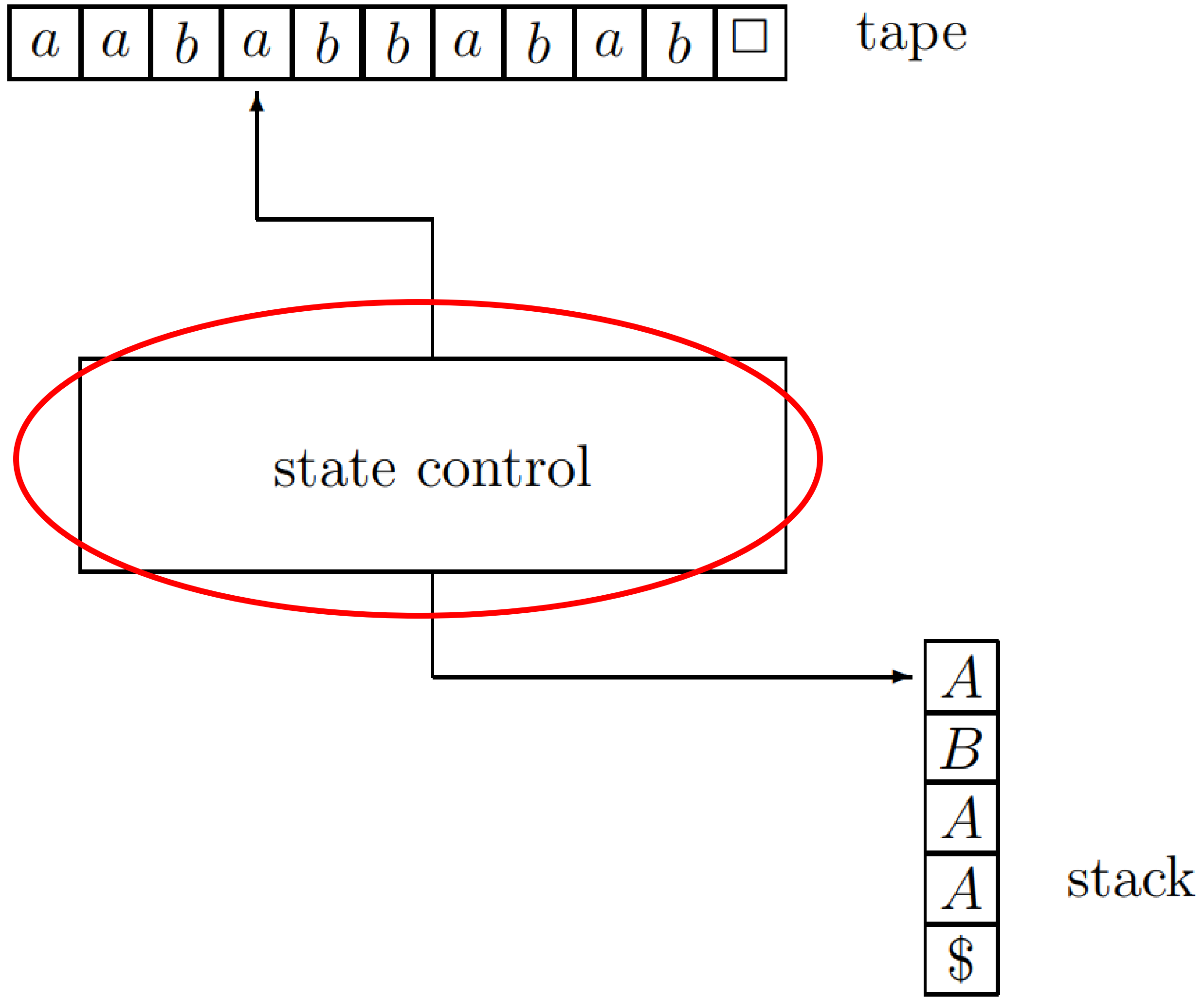
Push Down Automata



4. There is a stack head which can read the top symbol of the stack.

This head can also pop the top symbol, and it can push symbols of Γ onto the stack.

Push Down Automata



5. There is a state control, which can be in any one of a finite number of states.

The set of states is denoted by Q. The set Q contains one special state q, called the start state.

Formal Definition

A deterministic PDA is a 5-tuple $M = (\Sigma, \Gamma, Q, \delta, q)$, where

- Σ is a finite set, called the tape alphabet; the blank symbol # is not contained in Σ,
- Γ is a finite set, called the stack alphabet; this alphabet contains the special symbol \$,
- Q is a finite set, whose elements are called states,
- q is an element of Q; it is called the start state,
- δ is called the transition function, which is a function

$$Q \times (\Sigma \cup \{\#\}) \times \Gamma \to Q \times \{N, R\} \times \Gamma^*$$

A Sample Delta Rule

$$\underset{\text{states}}{q}a\$ \rightarrow \underset{\text{states}}{q}R\$S$$

When we focus on the states here, we will see that the state has not changed. Thus, we can read it as 'stay in the same state'.

A Sample Delta Rule

$qa\$ \rightarrow qR\$S$

tape

When the tape head reads an ‘a’ symbol from the tape, the tape head must go one cell to the right.

A Sample Delta Rule

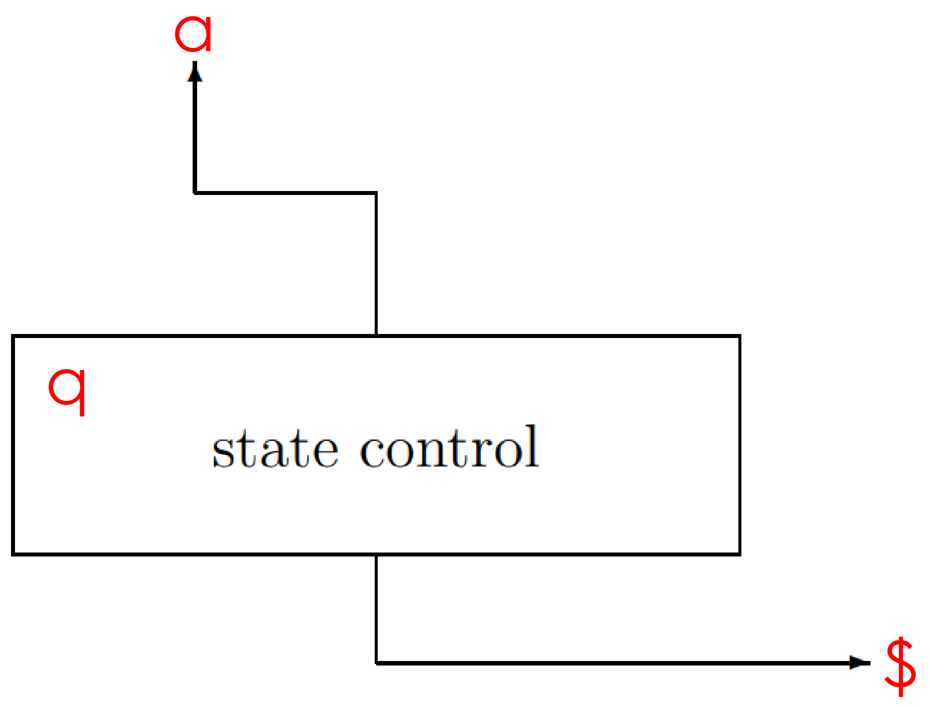
$$qa\$ \rightarrow qR\underbrace{\$S}$$

stack

When stack head pop the dollar symbol from the stack, the stack head must push $S symbols into the stack in order.

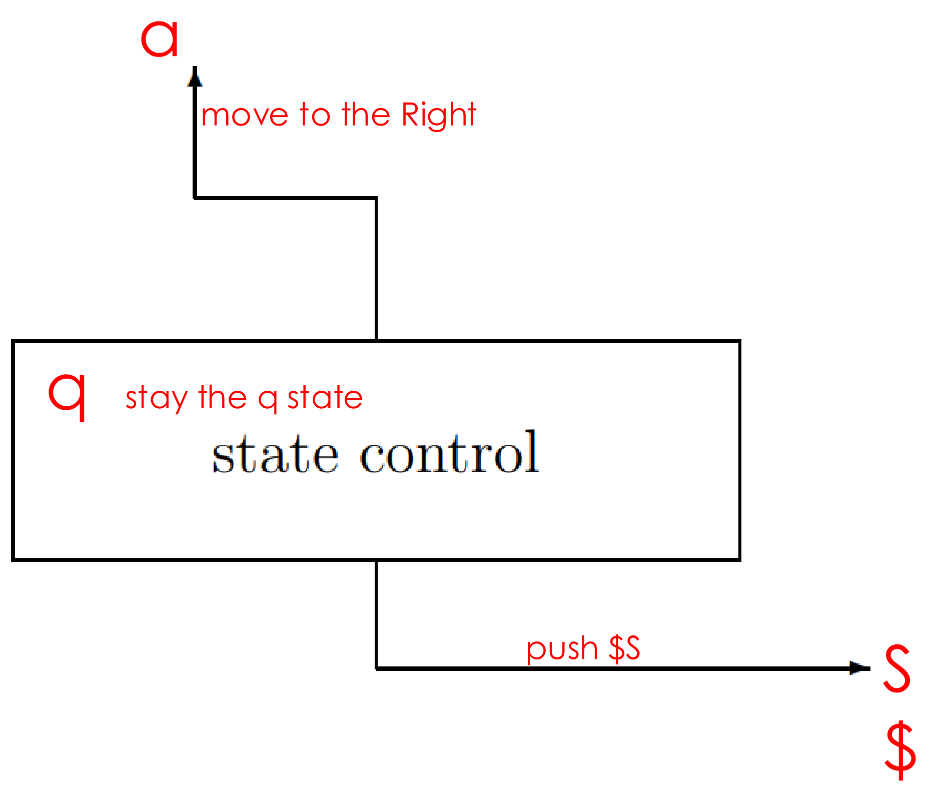
A Sample Delta Rule

qa\$ → qR\$S

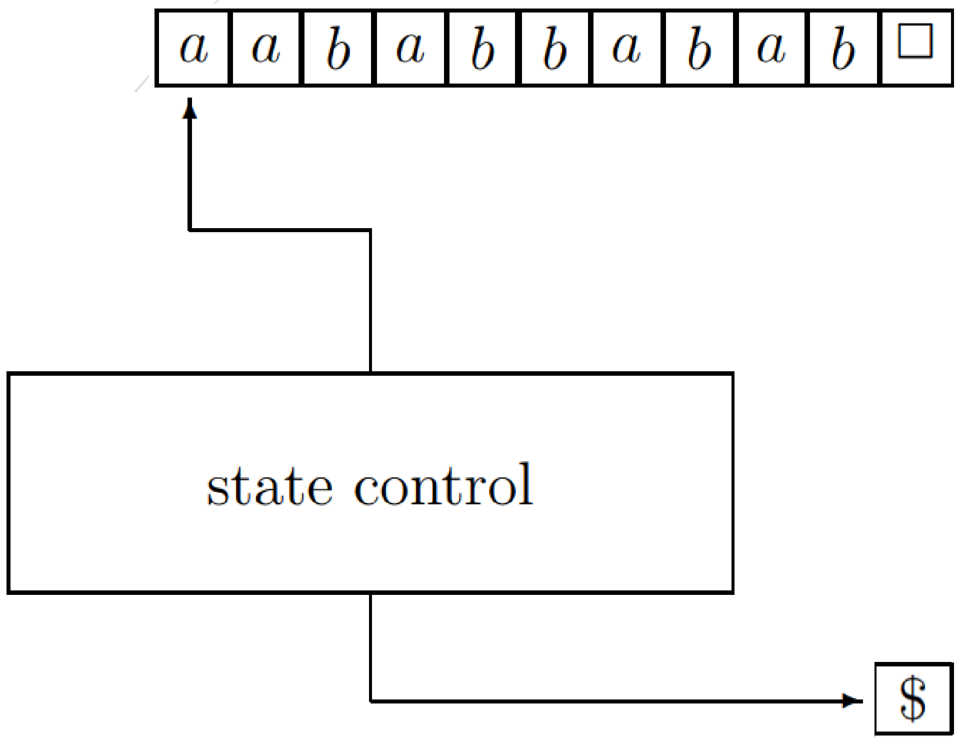


A Sample Delta Rule

qa$ → qR$S



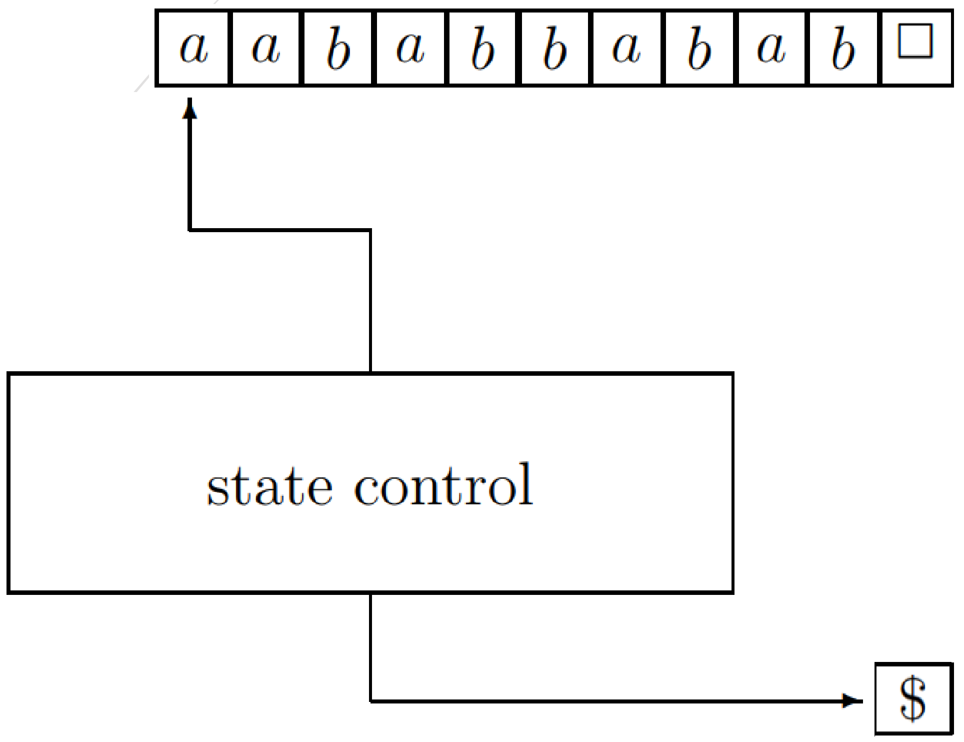
Execution of PDA



User can enter a string into the tape, but cannot see the stack. Only state controller can use the stack.

When the program is started, we will assume that the tape head is on the first letter of the string written by the user.

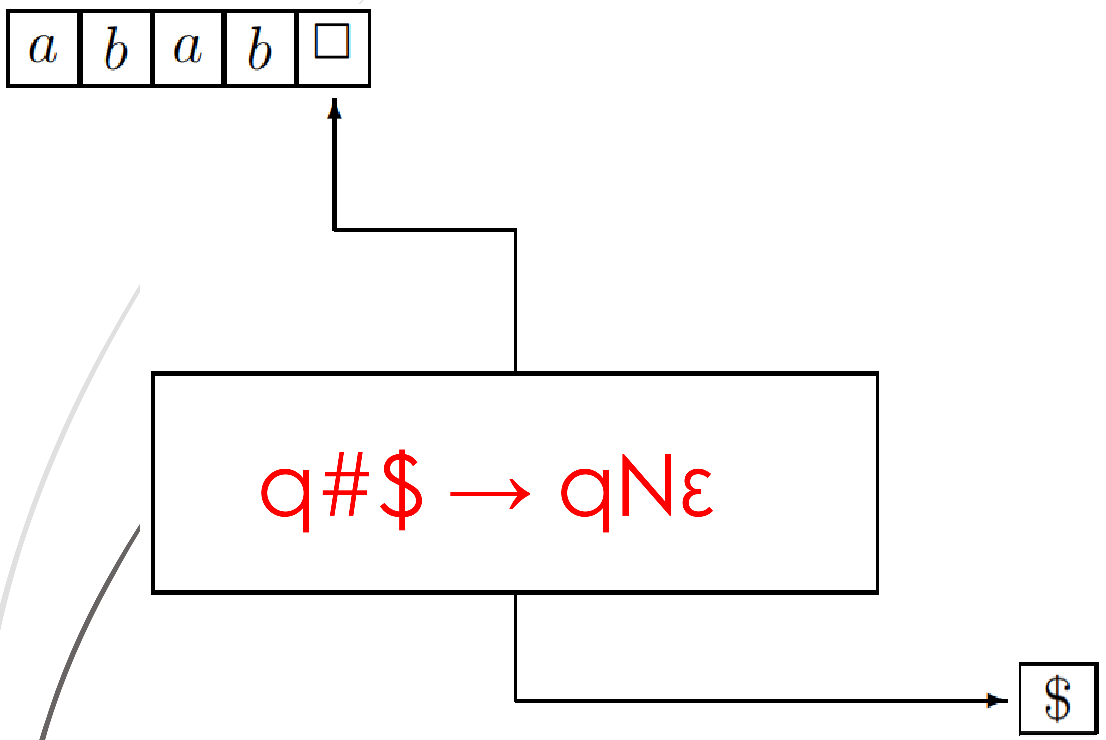
Execution of PDA



State controller starts on the start state q and the stack has only one dollar symbol.

A command is selected according to the symbol to be read from the tape, and the operation on the right hand side is performed.

Execution of PDA



When the stack gets empty, PDA terminates.

When PDA terminates, if the tape head is on the cell immediately to the right cell of the last symbol of the string, this is good news.

Because the string on the tape is accepted.

A first example: Properly nested parentheses

Algorithm: remember the number of ‘a’s, then match them with ‘b’s.

Use the stack to remember the a's. For each ‘a’ reading, push a symbol (let's say S) into the stack. Delete one from the stack each time ‘b’ is read.

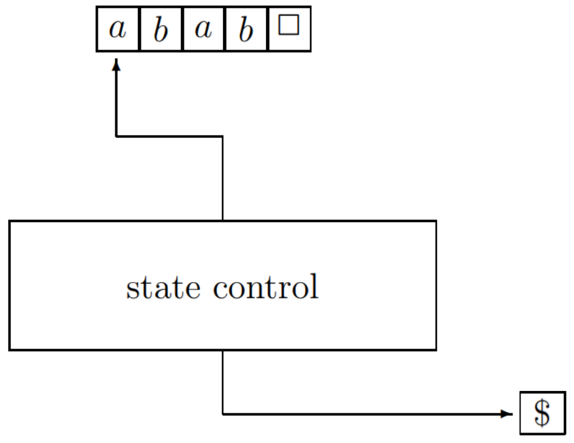
$(3*(x^{(x-2)}+1))-(y+1)*(x+5)$

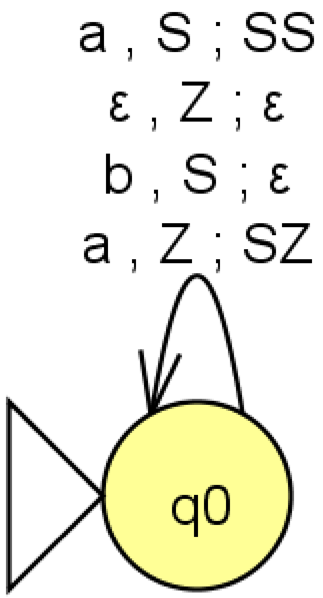
(((   ))) ( ) ( )

aaabbb ab ab

A first example: Properly nested parentheses

qa\$ → qR\$S	remember
qb\$ → qNε	reject
q#\$ → qNε	accept
qaS → qRSS	remember
qbS → qRε	forget
q#S → qNS	reject with loop-forever





A first example: Properly nested parentheses

$qa\$ \to qR\$S$	remember
$qb\$ \to qN\varepsilon$	reject
$q\#\$ \to qN\varepsilon$	accept

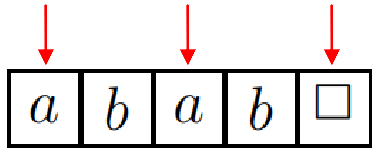
$qaS \to qRSS$	remember
$qbS \to qR\varepsilon$	forget
$q\#S \to qNS$	reject with loop-forever

a	b	a	b	$\square$

When do you think the top part will work?

A first example: Properly nested parentheses

$qa\$ \to qR\$S$	remember
$qb\$ \to qN\varepsilon$	reject
$q\#\$ \to qN\varepsilon$	accept
$qaS \to qRSS$	remember
$qbS \to qR\varepsilon$	forget
$q\#S \to qNS$	reject without termination



1. At the start of PDA
2. When |a|=|b|

A first example: Properly nested parentheses

a	b	a	b	$\square$

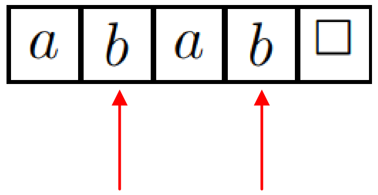
$qa\$ \rightarrow qR\$S$	remember
$qb\$ \rightarrow qN\varepsilon$	reject
$q\#\$ \rightarrow qN\varepsilon$	accept

$qaS \rightarrow qRSS$	remember
$qbS \rightarrow qR\varepsilon$	forget
$q\#S \rightarrow qNS$	reject without termination

So when does this part work?

A first example: Properly nested parentheses

qa$ → qR$S	remember
qb$ → qNε	reject
q#$ → qNε	accept



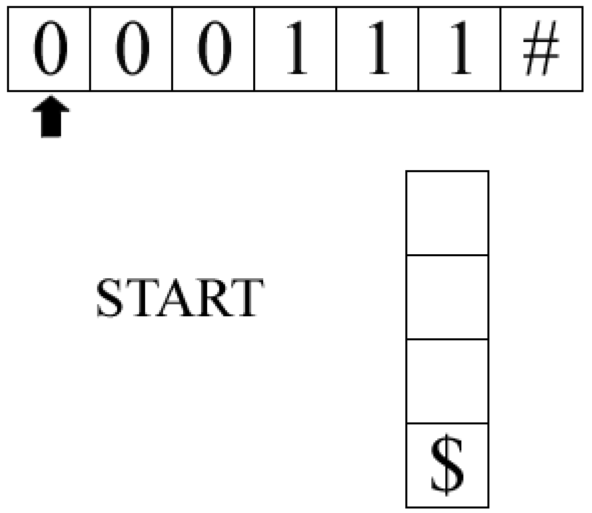
qaS → qRSS	remember
qbS → qRε	forget
q#S → qNS	reject without termination

If |a|>|b|

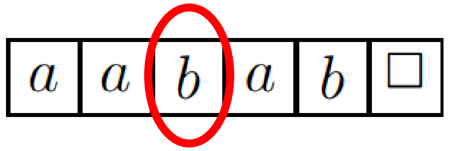
A second example: 0^n1^n

$q_0 0\$ \rightarrow q_0 R\$S$	remember
$q_0 1\$ \rightarrow q_0 N\$$	reject without termination
$q_0 \#\$ \rightarrow q_0 N\varepsilon$	accept
$q_0 0S \rightarrow q_0 RSS$	remember
$q_0 1S \rightarrow q_1 R\varepsilon$	forget
$q_0 \#S \rightarrow q_0 NS$	reject without termination
$q_1 0\$ \rightarrow q_1 N\$$	reject without termination
$q_1 1\$ \rightarrow q_1 N\$$	reject without termination
$q_1 \#\$ \rightarrow q_1 N\varepsilon$	accept
$q_1 0S \rightarrow q_1 NS$	reject without termination
$q_1 1S \rightarrow q_1 R\varepsilon$	forget
$q_1 \#S \rightarrow q_1 NS$	reject without termination

$0^3 1^3$: 000111

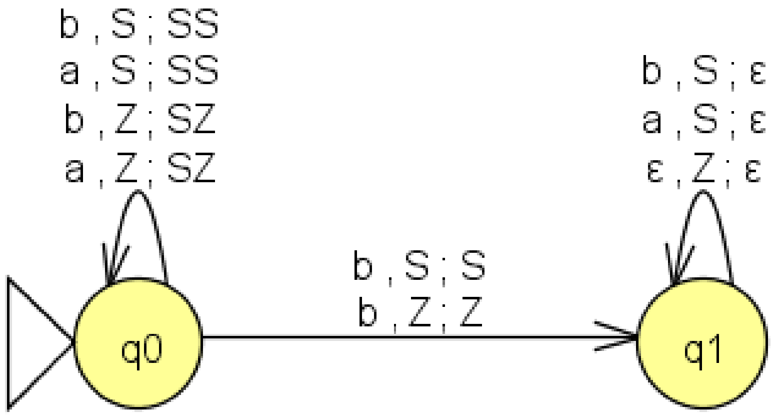


A third example: Find b in the middle



qa\$ → qR\$S	remember
qb\$ → q'R\$	middle b
qb\$ → qR\$S	remember
q#\$ → qN\$	reject without termination
qaS → qRSS	remember
qbS → q'RS	middle b
qbS → qRSS	remember
q#S → qNS	reject without termination
q'a\$ → q'Nε	reject
q'b\$ → q'Nε	reject
q'#\$ → q'Nε	accept
q'aS → q'Rε	forget
q'bS → q'Rε	forget
q'#S → q'NS	reject without termination

q: the state before b in the middle
q': the state after b in the middle



Equivalence of PDA and CFG

Theorem: Let Σ be an alphabet and A ⊆ Σ* be a language. Then A is CFG if and only if there exists a <u>nondeterministic PDA</u> that accepts A.

1. Each **A → BC** rule transforms **qaA → qNCB**, for all a ∈ Σ
2. Each **A → a** rule transforms **qaA → qRε**
3. Each **$ → ε** rule transforms **q#$ → qNε**

- That’s all.
- Thanks for listening.