



# Computing Theory

COMP 147

Chapter 2: Context-Free Languages  
Section 2.2: Pushdown Automata

# Announcements

- Assignment 3 due tomorrow night
  - no late submission allowed
- Exam1 on Tuesday October 13
- Review Session next Class

# Pushdown Automata

Definition  
Moves of the PDA  
Languages of the PDA

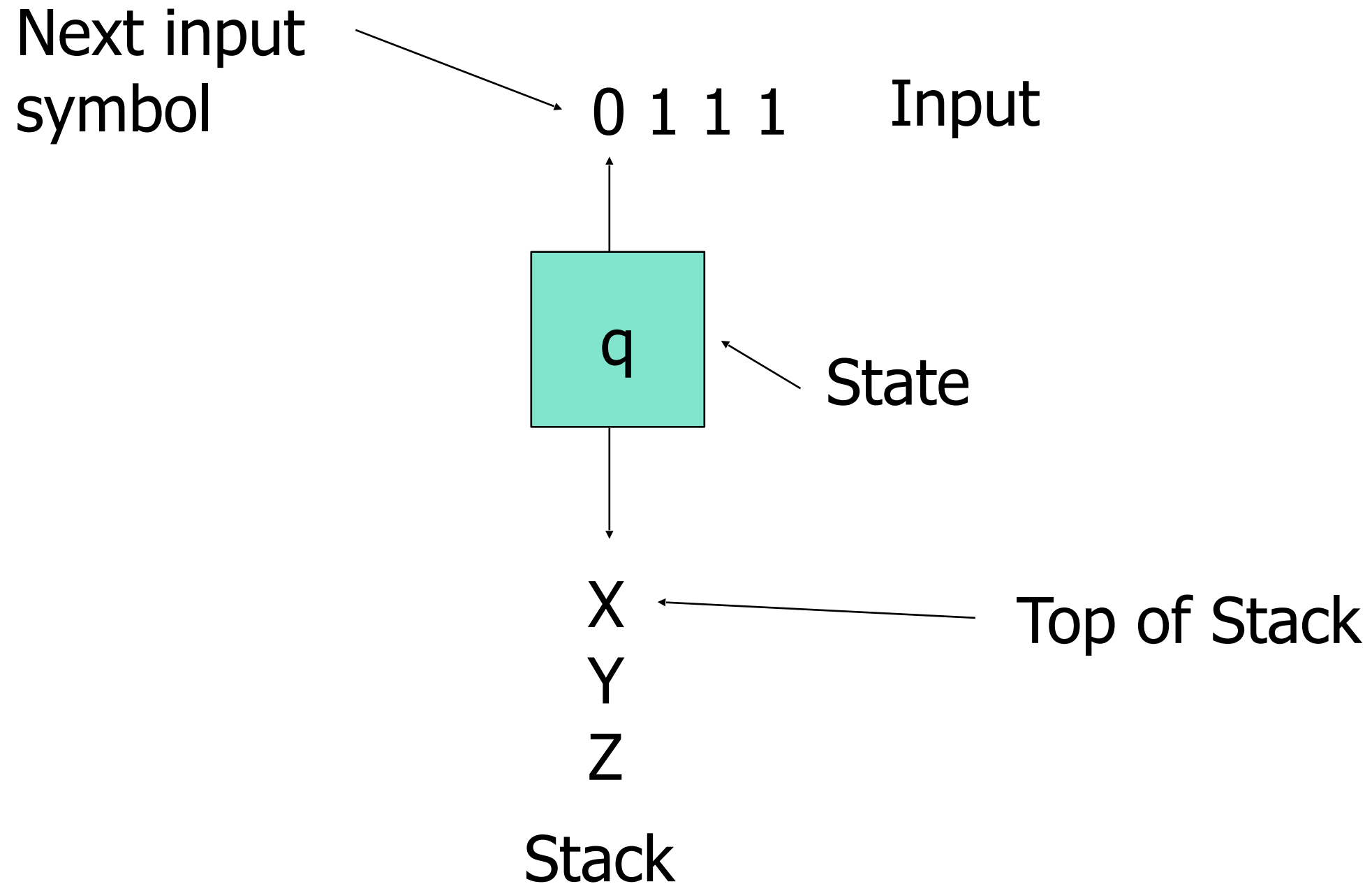
# Pushdown Automata

- The PDA is an automaton equivalent to the CFG in language-defining power.
- Only the nondeterministic PDA defines all the CFL's.
- But the deterministic version models parsers.
  - Most programming languages have deterministic PDA's.

# Intuition: PDA

- Think of an NFA with the additional power that it can manipulate a stack.
- Its moves are determined by:
  - The current state (of its “NFA”),
  - The current input symbol (or  $\epsilon$ ), and
  - The current symbol on top of its stack.

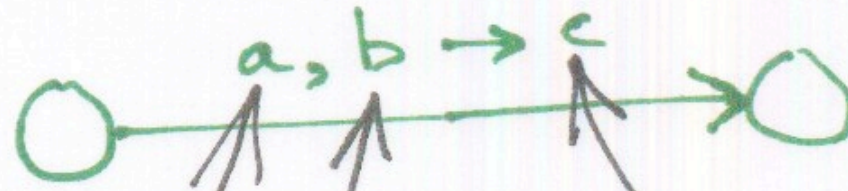
# Picture of a PDA



# FINITE STATE MACHINE



# PUSHDOWN AUTOMATON



Input  
symbol

Symbol on top of  
the stack.  
This symbol is  
popped.

This symbol  
is pushed  
onto the stack.

May "ε" be

"ε" means the  
stack is neither  
read nor popped

"ε" means  
nothing  
is pushed.



# PDA Formalism

- A PDA is described by:
  1. A finite set of **states** ( $Q$ ).
  2. An **input alphabet** ( $\Sigma$ ).
  3. A **stack alphabet** ( $\Gamma$ ).
  4. A **transition function** ( $\delta$ ).
  5. A **start state** ( $q_0$ ).
  6. A set of **final states** ( $F \subseteq Q$ ).



# The Transition Function

- Takes three arguments:
  - A state, in  $Q$ .
  - An input, which is either a symbol in  $\Sigma$  or  $\epsilon$ .
  - A stack symbol in  $\Gamma$ .
- $\delta(q, a, Z)$  is a set of zero or more actions of the form  $(p, \alpha)$ .
  - $p$  is a state;  $\alpha$  is a string of stack symbols.

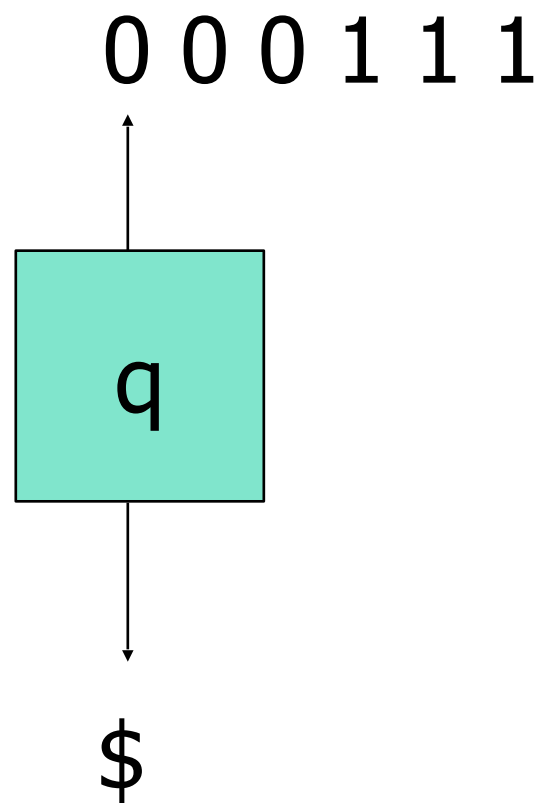
# Example: PDA

- Design a PDA to accept  $\{0^n 1^n \mid n \geq 1\}$ .
- The states:
  - $q$  = start state. We are in state  $q$  if we have seen only 0's so far.
  - $p$  = we've seen at least one 1 and may now proceed only if the inputs are 1's.
  - $f$  = final state; accept.

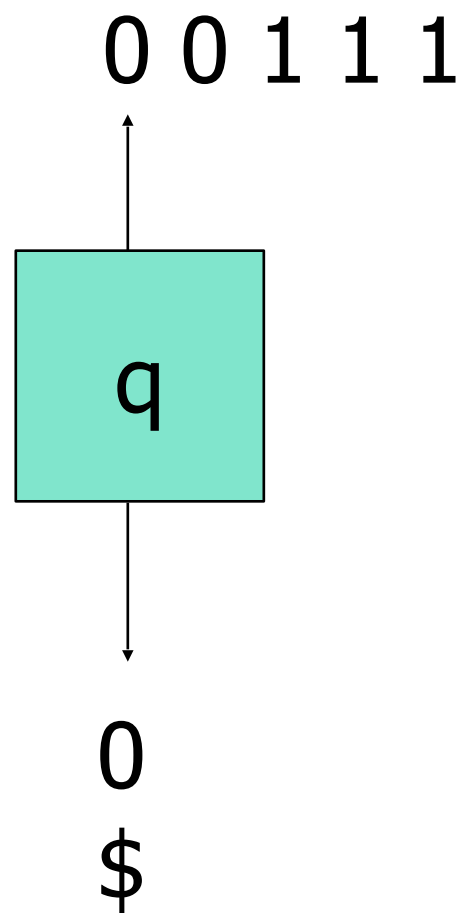
# Example: PDA – (2)

- The stack symbols:
  - \$ = start symbol. Also marks the bottom of the stack, so we know when we have counted the same number of 1's as 0's.
  - 0 = used to count the number of 0's seen on the input.

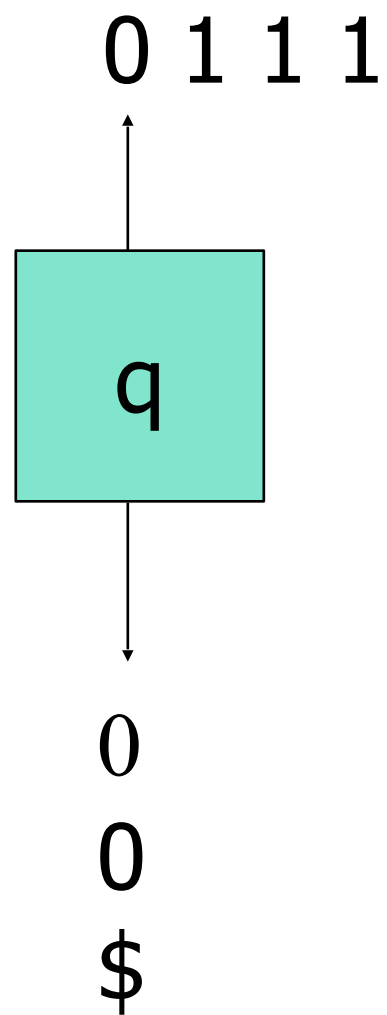
# Actions of the Example PDA



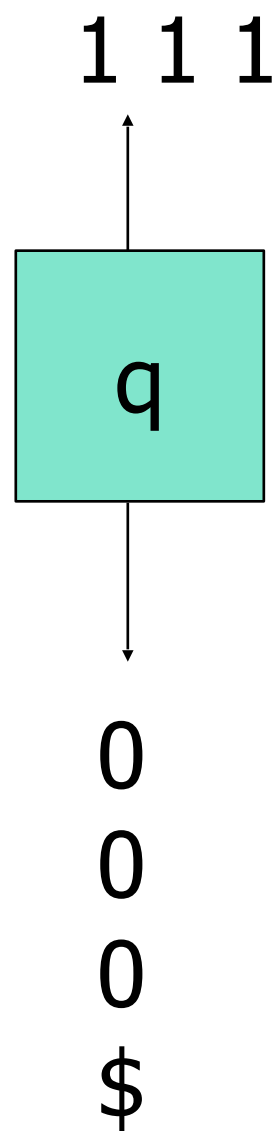
# Actions of the Example PDA



# Actions of the Example PDA

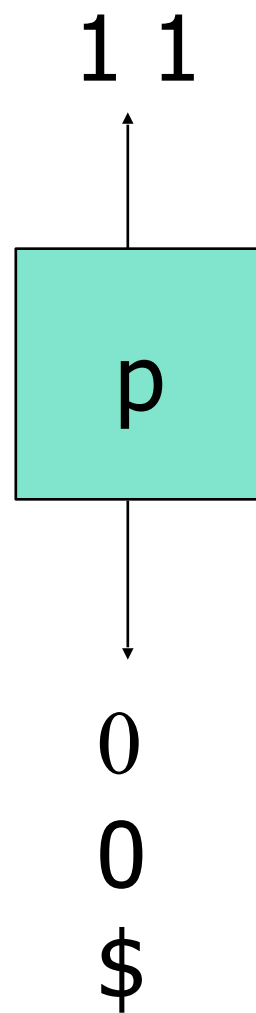


# Actions of the Example PDA

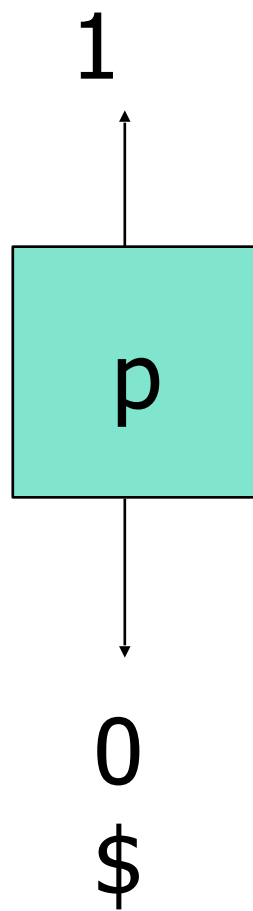




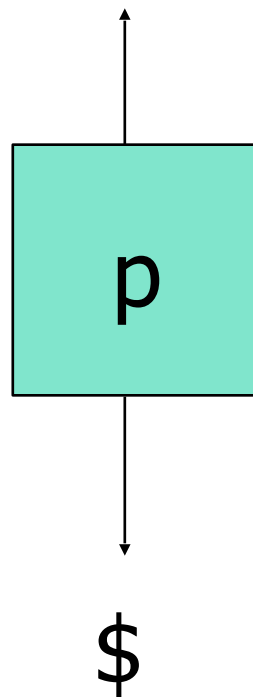
# Actions of the Example PDA



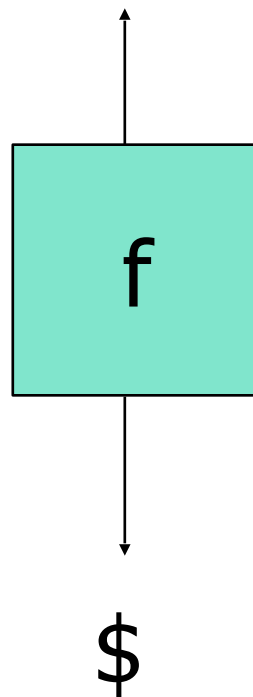
# Actions of the Example PDA



# Actions of the Example PDA

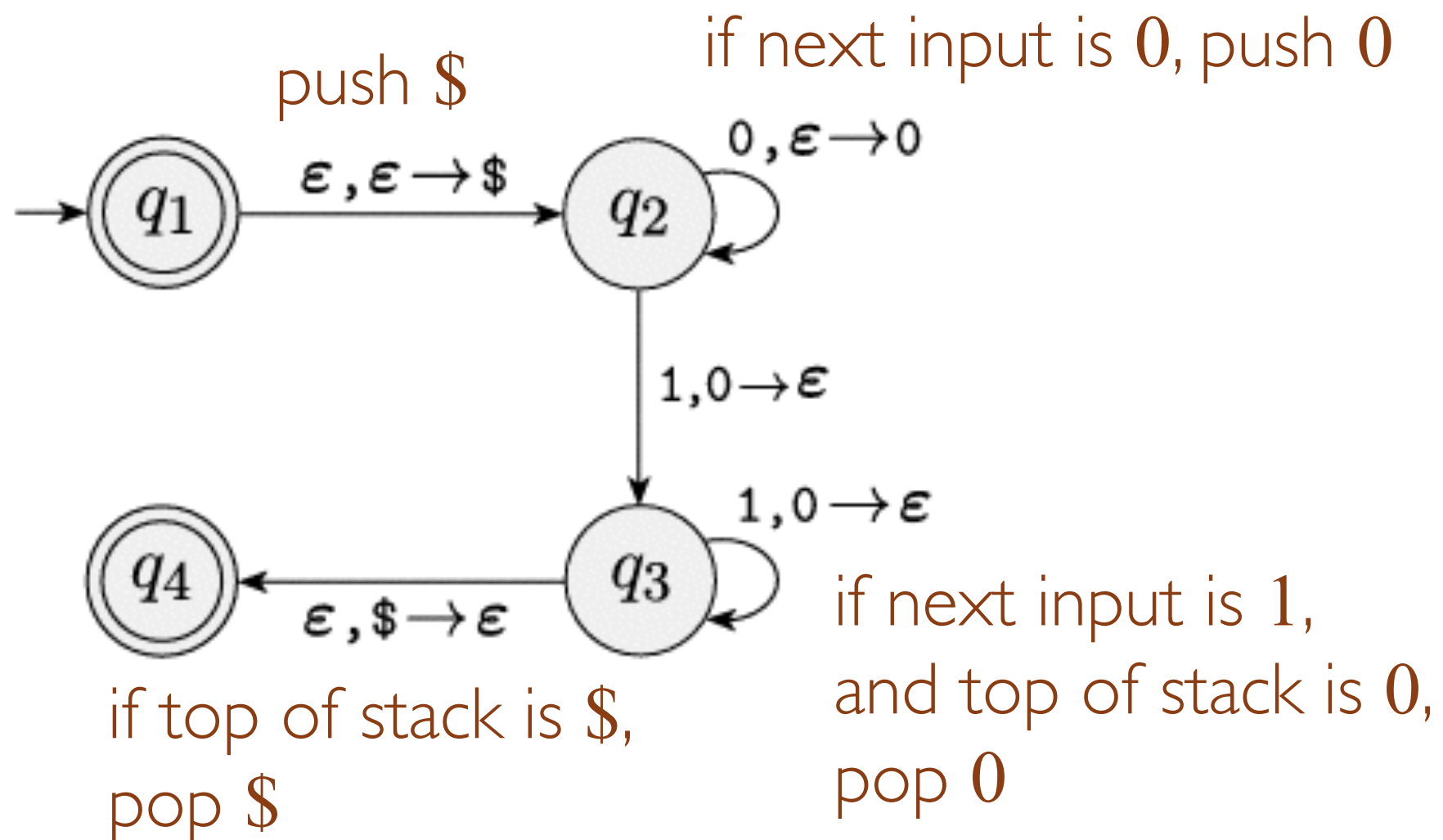


# Actions of the Example PDA



# PDA Example

$$L = \{ 0^n 1^n \mid n \geq 0 \}$$



# PDA: Transition Notation

- $\epsilon, \epsilon \rightarrow \epsilon$       transition immediately  
do not consume input  
do not modify stack
- $x, \epsilon \rightarrow \epsilon$       transition if next input is  $x$   
consume  $x$  from input  
do not modify stack
- $\epsilon, y \rightarrow \epsilon$       transition if top of stack is  $y$   
do not consume input  
pop  $y$  from stack
- $x, y \rightarrow \epsilon$       transition if next input is  $x$  and top of stack is  $y$   
consume  $x$  from input  
pop  $y$  from stack

# PDA: Transition Notation

- $\epsilon, \epsilon \rightarrow z$  transition immediately  
do not consume input  
push  $z$  on stack
- $x, \epsilon \rightarrow z$  transition if next input is  $x$   
consume  $x$  from input  
push  $z$  on stack
- $\epsilon, y \rightarrow z$  transition if top of stack is  $y$   
do not consume input  
pop  $y$  from stack and push  $z$  on stack
- $x, y \rightarrow z$  transition if next input is  $x$  and top of stack is  $y$   
consume  $x$  from input  
pop  $y$  from stack and push  $z$  on stack



# PDA: Formal Definition

## DEFINITION 2.13

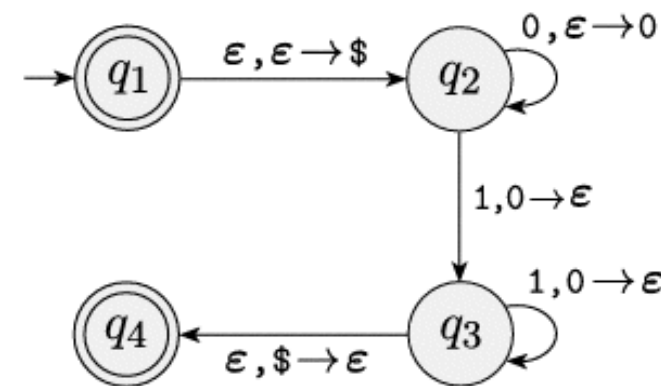
---

A *pushdown automaton* is a 6-tuple  $(Q, \Sigma, \Gamma, \delta, q_0, F)$ , where  $Q$ ,  $\Sigma$ ,  $\Gamma$ , and  $F$  are all finite sets, and

1.  $Q$  is the set of states,
2.  $\Sigma$  is the input alphabet,
3.  $\Gamma$  is the stack alphabet,
4.  $\delta: Q \times \Sigma_\epsilon \times \Gamma_\epsilon \longrightarrow \mathcal{P}(Q \times \Gamma_\epsilon)$  is the transition function,
5.  $q_0 \in Q$  is the start state, and
6.  $F \subseteq Q$  is the set of accept states.

$$\Sigma_\epsilon = \Sigma \cup \{\epsilon\} \text{ and } \Gamma_\epsilon = \Gamma \cup \{\epsilon\}$$

# PDA Example: Formal Definition

$$M = \{ Q, \Sigma, \Gamma, \delta, q_1, F \} \text{ recognizes } L = \{ 0^n 1^n \mid n \geq 0 \}.$$
$$Q = \{q_1, q_2, q_3, q_4\},$$
$$\Sigma = \{0, 1\},$$
$$\Gamma = \{0, \$\},$$
$$F = \{q_1, q_4\}, \text{ and}$$


$\delta$  is given by the following table, wherein blank entries signify  $\emptyset$ .

Input:	0			1			$\epsilon$		
Stack:	0	\$	$\epsilon$	0	\$	$\epsilon$	0	\$	$\epsilon$
$q_1$									$\{(q_2, \$)\}$
$q_2$	$\{(q_2, 0)\}$			$\{(q_3, \epsilon)\}$					
$q_3$				$\{(q_3, \epsilon)\}$			$\{(q_4, \epsilon)\}$		
$q_4$									

# PDA: Nondeterminism

$$L = \{ a^i b^j c^k \mid i, j, k \geq 0 \text{ and } i = j \text{ or } i = k \}$$

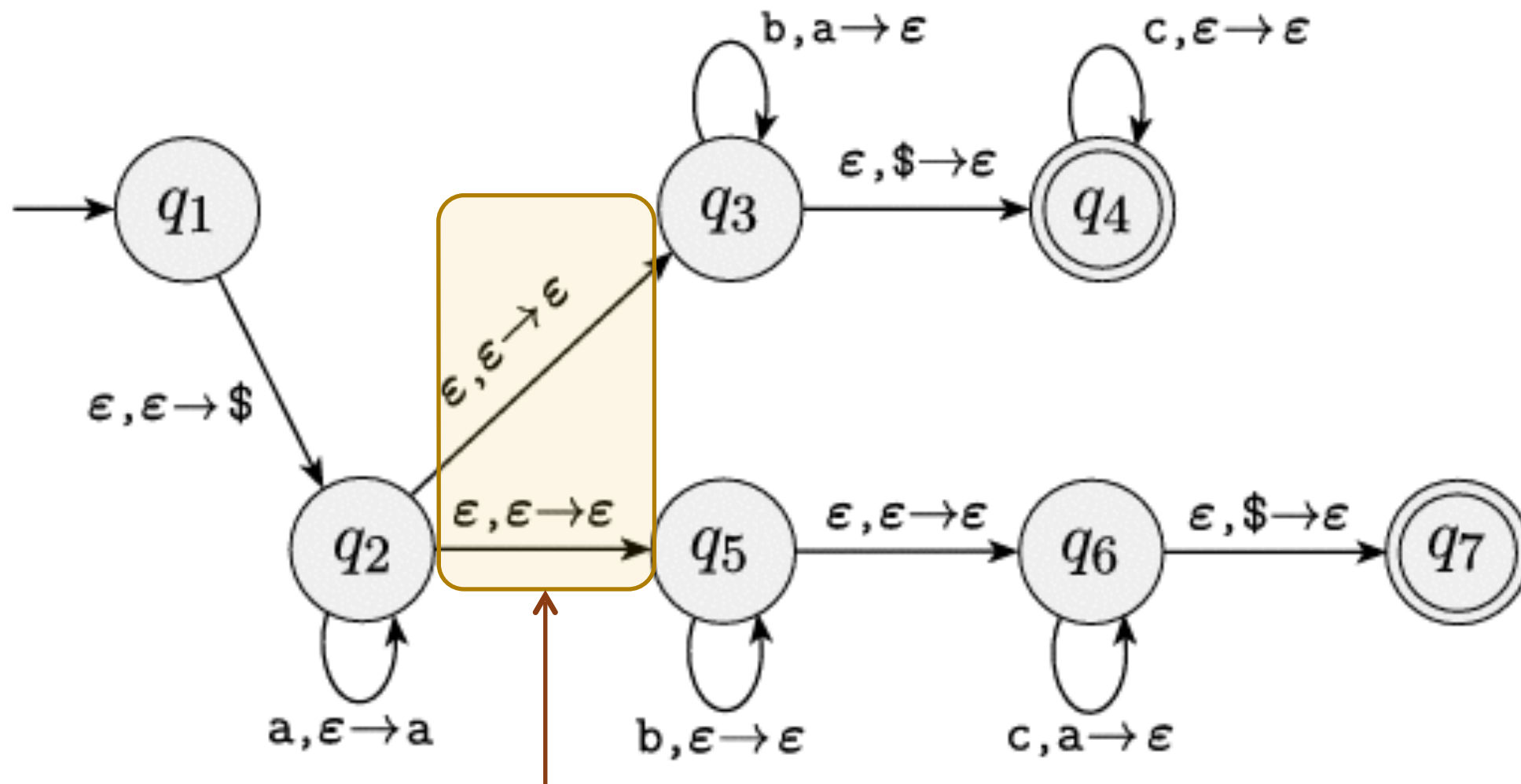
# a's and # b's is the same, or

# a's and # c's is the same

- Read a's and push on stack (as in prev. example)
- Pop the a's to count b's or the c's?

# PDA: Nondeterminism

$$L = \{ a^i b^j c^k \mid i, j, k \geq 0 \text{ and } i = j \text{ or } i = k \}$$



We can view this as a nondeterministic choice.

- (a) Take both paths in parallel, or
- (b) Guess (correctly) which one to take.

# PDA Example: String Reversal

$$L = \{ ww^R \mid w \in \{0, 1\}^* \} \quad (w^R \text{ means reverse of } w)$$

