CS3110 Formal Language and Automata

Tingting Chen
Computer Science
Cal Poly Pomona

Pushdown Automaton: Formal Definition

A nondeterministic pushdown automaton (NPDA) is a 7-tuple

Q is a finite set of states

Σ is the input alphabet (a finite set)

Γ is the stack alphabet (a finite set)

$\delta : Q \times (\Sigma \cup \{\lambda\}) \times \Gamma \rightarrow$ (finite subsets of $Q \times \Gamma^*$) is the transition function

$q_0 \in Q$ is the start state

$z \in \Gamma$ is the initial stack symbol

$F \subseteq Q$ is the set of accepting states

Example

So we can fully specify any NPDA like this:

- Q = {q0, q1, q2, q3}
- Σ = {a, b}
- Γ = {0, 1}
- q_0 is the start state
- z = $ (the empty stack marker)
- F = {q_3}
- δ is the transition function:

Example

- $\delta(q0, a, \$) \rightarrow \{(q1, 1\$), (q3, \lambda)\}$
- $\delta(q0, \lambda, \$) \rightarrow \{(q3, \lambda)\}$
- $\delta(q1, a, 1) \rightarrow \{(q1, 11)\}$
- $\delta(q1, b, 1) \rightarrow \{(q2, \lambda)\}$
- $\delta(q2, b, 1) \rightarrow \{(q2, \lambda)\}$
- $\delta(q2, \lambda, \$) \rightarrow \{(q3, \lambda)\}$

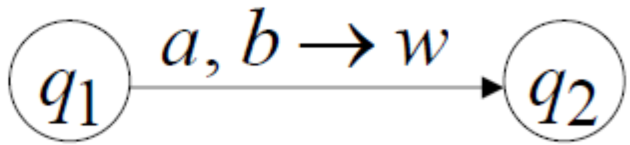
- This PDA is nondeterministic. Why?
- What is the language accepted by this PDA?
- $L=\{a^n b^n : n \geq 0\} \cup \{a\}$

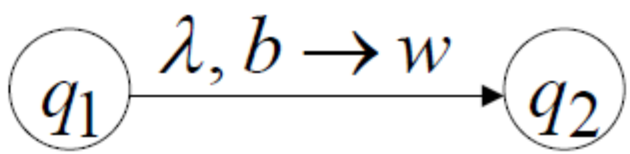
Working with a Stack

- You can only access the top element of the stack.
- To access the top element of the stack, you have to POP it off the stack.
- Once the top element of the stack has been POPped, if you want to save it, you need to PUSH it back onto the stack immediately.
- Characters from the input string must be read one character at a time. You cannot back up.
- The current configuration of the machine includes: the current state, the remaining characters left in the input string, and the entire contents of the stack

Deterministic PDA: DPDA

Allowed transitions:

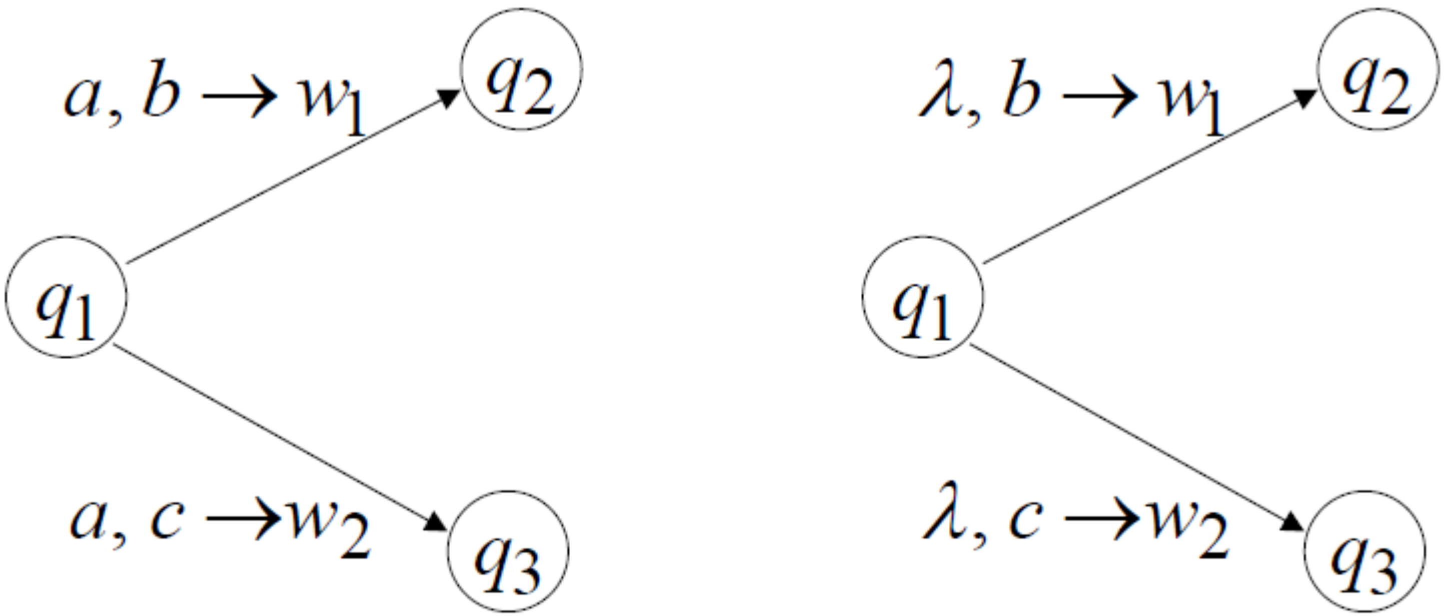




Deterministic choices

Deterministic PDA: DPDA

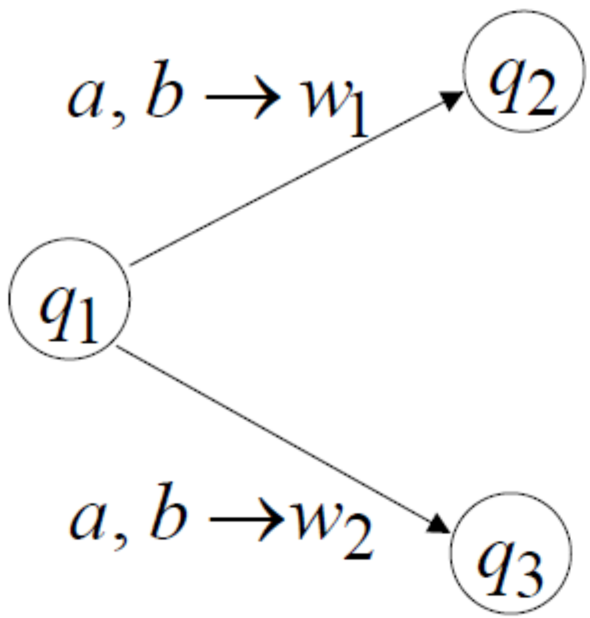
Allowed transitions:

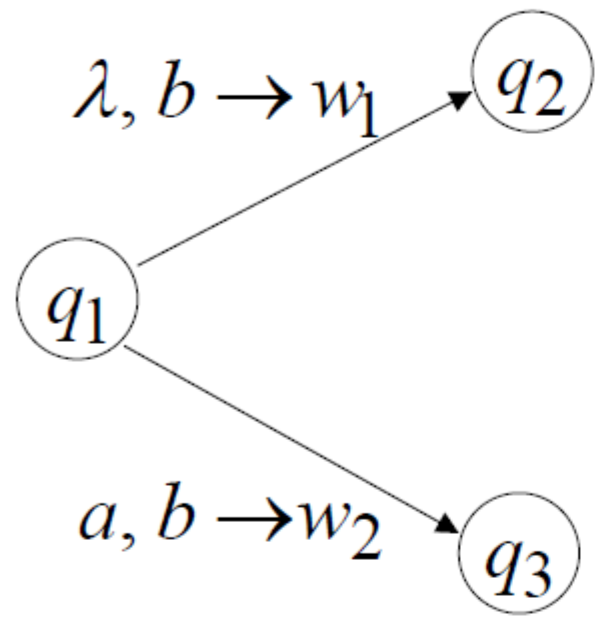


Deterministic choices

Deterministic PDA: DPDA

Not-Allowed:

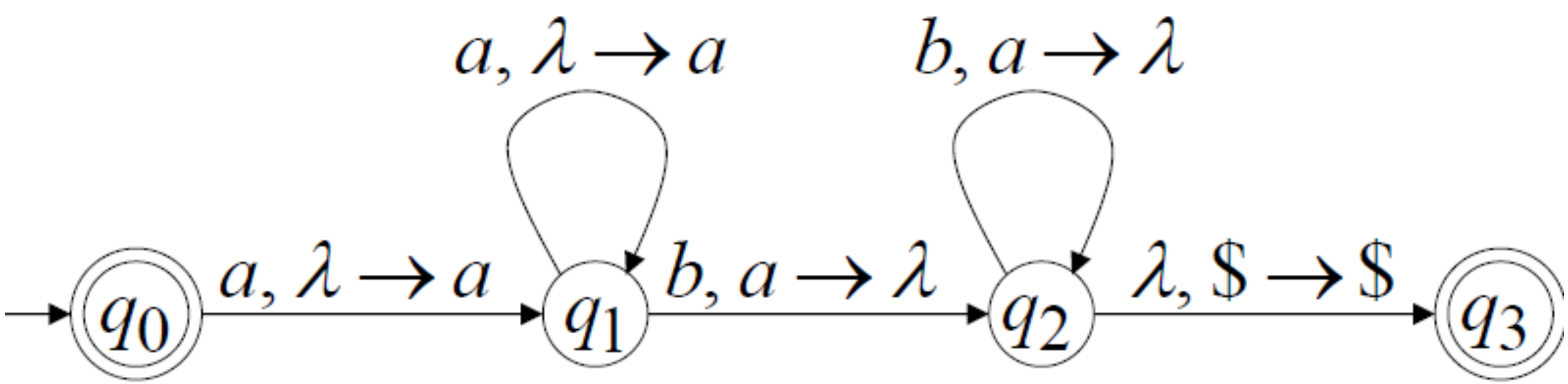




Non-Deterministic choices

DPDA -- Example

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



Instantaneous description

Given the transition function

$\delta : Q \times (\Sigma \cup \{\lambda\}) \times \Gamma \to$ (finite subsets of $Q \times \Gamma^*$)

a configuration, or *instantaneous description,* of M is a snapshot of the current status of the PDA. It consists of a triple:

(q, w, u)

where:

$q \in Q$ (q is the current state of the control unit)

$w \in \Sigma^*$ (w is the remaining unread part of the input string), and

$u \in \Gamma^*$ (u is the current stack contents, with the leftmost symbol indicating the top of the stack)

Instantaneous description

- To indicate that the application of a transition has caused our PDA to move from one state to another, we use the following notation:
  - $(q_1, aw, bx) \vdash (q_2, w, yx)$
- To indicate that we have moved from one state to another via the application of several transitions, we use:
  - $(q_1, aw, bx) \vdash^* (q_2, w, yx)$
    or
  - $(q_1, aw, bx) \vdash_M^* (q_2, w, yx)$ to indicate a specific PDA

Definition: Acceptance

If $M = (Q, \Sigma, \Gamma, \delta, q_0, z, F)$ is a push-down automaton and $w \in \Sigma^*$, the string w is *accepted* by M if:

$$(q_0, w, \#) \ \vdash_M^* \ (q_f, \lambda, u)$$

for some $u \in \Gamma^*$ and some $q_f \in F$.

This means that we start at the start state, with the stack empty, and after processing the string w, we end up in an accepting state, with no more characters left to process in the original string. We don't care what is left on the stack.

This is called *acceptance by final state*.

Language Accepted by a PDA M

Let $M = (Q, \Sigma, \Gamma, \delta, q_0, z, F)$ be a non-deterministic push-down automaton. The language accepted by M is the set

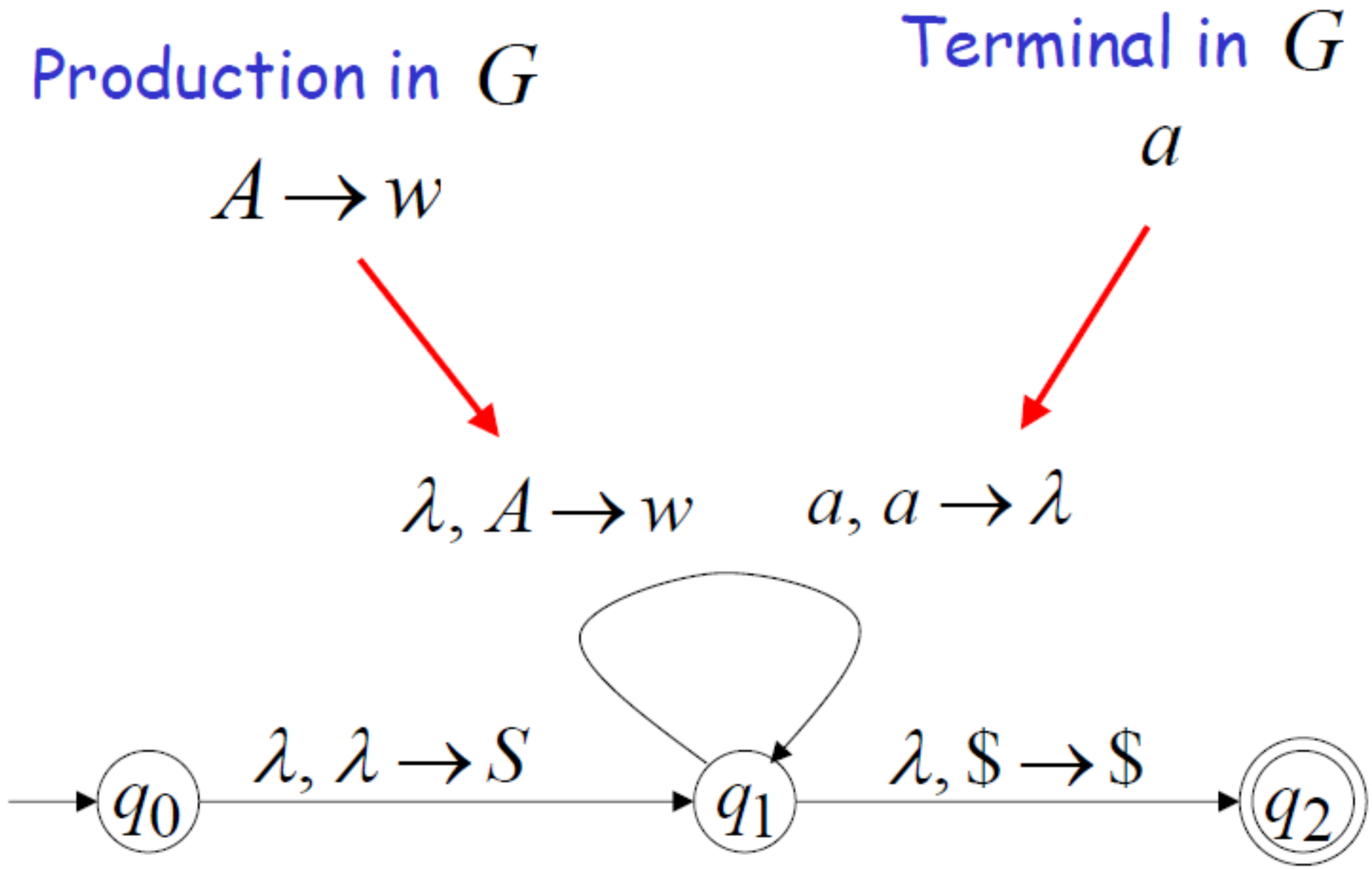
$\{w \in \Sigma^*: (q_0, w, \#) \mid\!-_M^* (q_f, \lambda, u), u \in \Gamma^*, q_f \in F\}$.

PDAs accepts Context-Free Languages

- Any context-free grammar G can be converted to a PDA M with L(G) = L(M).
- Any PDA M can be converted to a context-free grammar G with L(G) = L(M).

Convert CFG to PDA

- Convert grammar G to PDA M



Example

Here is a grammar in GNF:

$G = (V, T, S, P)$, where

$V = \{S, A, B, C\}$,

$T = \{a, b, c\}$,

$S = S$,

and $P =$

$S \rightarrow aA$

$A \rightarrow aABC \mid bB \mid a$

$B \rightarrow b$

$C \rightarrow c$

Let's convert this grammar to a PDA.

Example

The steps that the PDA would go through to process the string *aaabc*, starting with the initial precondition:

$(q_0, aaabc, \$)$

$(q_0, aaabc, \$) \vdash (q_1, aaabc, S\$)$
$\vdash (q_1, aaabc, aA\$)$
$\vdash (q_1, aabc, A\$)$
$\vdash (q_1, aabc, aABC\$)$
$\vdash (q_1, abc, ABC\$)$
$\vdash (q_1, abc, aBC\$)$
$\vdash (q_1, bc, BC\$)$
$\vdash (q_1, bc, bC\$)$
$\vdash (q_1, c, C\$)$
$\vdash (q_1, c, c\$)$
$\vdash (q_1, \lambda, \$)$
$\vdash (q_2, \lambda, \$)$

$S \rightarrow aA$
$A \rightarrow aABC \mid bB \mid a$
$B \rightarrow b$
$C \rightarrow c$

Example – Solution 2

The steps that the PDA would go through to process the string *aaabc*, starting with the initial precondition:

$(q_0, aaabc, \#) \vdash (q_1, aaabc, S\#)$

$\vdash (q_1, aabc, A\#)$

$\vdash (q_1, abc, ABC\#)$

$\vdash (q_1, bc, BC\#)$

$\vdash (q_1, c, C\#)$

$\vdash (q_1, \lambda, \#)$

$\vdash (q_2, \lambda, \#)$

Notice that this corresponds to the following leftmost derivation in the grammar:

$$S \Rightarrow aA \Rightarrow aaABC \Rightarrow aaaBC \Rightarrow aaabC \Rightarrow aaabc$$

Exercise

Given a Context-free grammar:

$$S \to a \mid aS \mid bSS \mid SSb \mid SbS$$

Convert it to a PDA