CS 4410

Automata, Computability, and Formal Language

Dr. Xuejun Liang

Chapter 7

Pushdown Automata

1. Nondeterministic Pushdown Automata
   - Definition of a Pushdown Automata
   - The Language Accepted by a Pushdown Automaton
2. Pushdown Automata and Context-Free Languages
   - Pushdown Automata for Context-Free Languages
   - Context-Free Grammar for Pushdown Automata
3. Deterministic Pushdown Automata and Deterministic Context-Free Languages
4. Grammars for Deterministic Context-Free Languages*

Learning Objectives

At the conclusion of the chapter, the student will be able to:

- Describe the components of a nondeterministic pushdown automaton
- State whether an input string is accepted by a nondeterministic pushdown automaton
- Construct a pushdown automaton to accept a specific language
- Given a context-free grammar in Greibach normal form, construct the corresponding pushdown automaton
- Describe the differences between deterministic and nondeterministic pushdown automata
- Describe the differences between deterministic and general context-free languages

Nondeterministic Pushdown Automata

- A pushdown automaton is a model of computation designed to process context-free languages
- Pushdown automata use a stack as storage mechanism

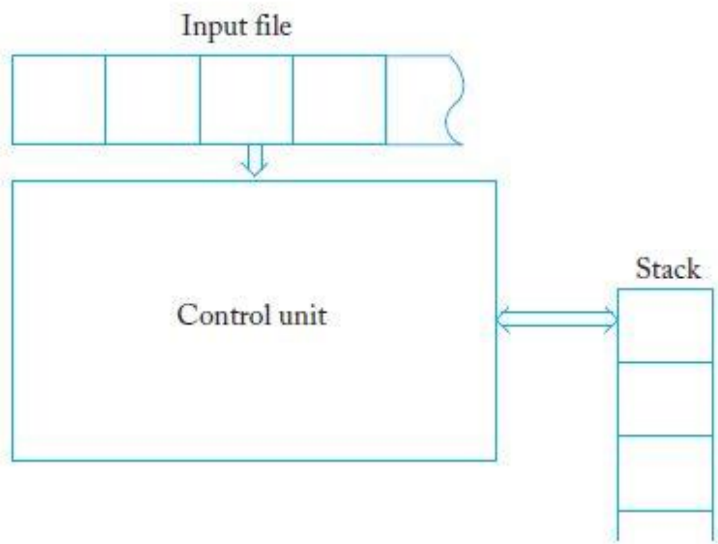

FIGURE 7.1

Nondeterministic Pushdown Automata

Definition 7.1: A **nondeterministic pushdown accepter** (npda) is defined by the sep-tuple $M = (Q, \Sigma, \Gamma, \delta, q_0, z, F)$
where

Q is a finite set of internal states of the control unit,
Σ is the input alphabet,
Γ is a finite set of symbols called the **stack alphabet**,
$\delta: Q \times (\Sigma \cup \{\lambda\}) \times \Gamma \rightarrow$ finite subsets of $Q \times \Gamma^*$ is the transition function,
$q_0 \in Q$ is the initial state of the control unit
$z \in \Gamma$ is the **stack start symbol**
$F \subseteq Q$ is the set of final states

Input to the transition function δ consists of a triplet:

A state, input symbol (or λ), and the symbol at the top of stack

Output of δ consists of a set of pairs:

A new state, and new top of stack

Note: Transitions can be used to model common stack operations

Sample NPDA Transition

- **Example 7.1** presents the sample transition rule:

  $\delta(q_1, a, b) = \{(q_2, cd), (q_3, \lambda)\}$
- According to this rule, when the control unit is in state q_1, the input symbol is a, and the top of the stack is b, two moves are possible:
  - New state is q_2 and the symbols cd replace b on the stack
  - New state is q_3 and b is simply removed from the stack

An Example NPDA

Example 7.2: $Q=\{q_0, q_1, q_2, q_3\}$, $\Sigma=\{a, b\}$, $\Gamma=\{0, 1\}$, $z=0$, $F=\{q_3\}$

$\delta(q_0, a, 0)=\{(q_1, 10), (q_3, \lambda)\}$,

$\delta(q_0, \lambda, 0)=\{(q_3, \lambda)\}$,

$\delta(q_1, a, 1)=\{(q_1,11)\}$,

$\delta(q_1, b, 1)=\{(q_2, \lambda)\}$,

$\delta(q_2, b, 1)=\{(q_2, \lambda)\}$,

$\delta(q_2, \lambda, 0)=\{(q_3, \lambda)\}$

Transition Graphs

- In the transition graph for a npda, each edge is labeled with the input symbol, the stack top, and the string that replaces the top of the stack
- The graph below represents the npda in Example 7.2:

$\delta(q_0, a, 0)=\{(q_1, 10), (q_3, \lambda)\}$,
$\delta(q_0, \lambda, 0)=\{(q_3, \lambda)\}$,
$\delta(q_1, a, 1)=\{(q_1,11)\}$,
$\delta(q_1, b, 1)=\{(q_2, \lambda)\}$,
$\delta(q_2, b, 1)=\{(q_2, \lambda)\}$,
$\delta(q_2, \lambda, 0)=\{(q_3, \lambda)\}$

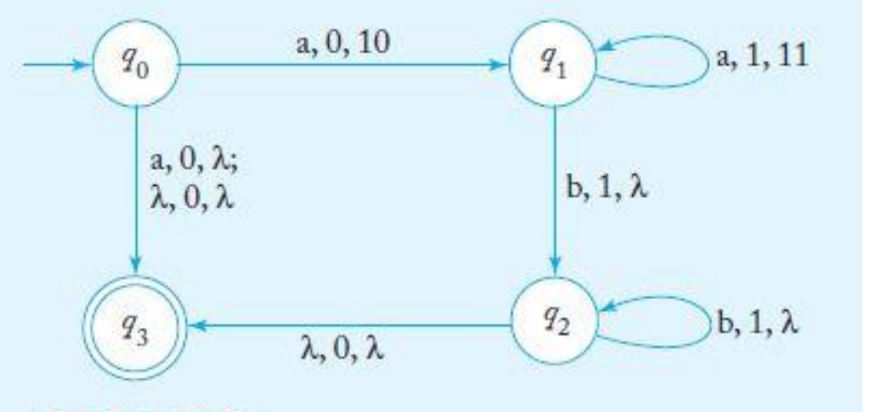

FIGURE 7.2

Language accepted by a Pushdown Automata

An ***instantaneous description*** is a triplet (q, w, u), where

q: current state, w: unread part of input string, and u: stack content (with the top as the leftmost symbol)

A ***move*** from (q_1, aw, bx) to (q_2, w, yx) denoted by

$(q_1, aw, bx) \vdash (q_2, w, yx)$

is possible, if and only if $(q_2, y) \in \delta(q_1, a, b)$.

Definition 7.2: Let $M=(Q, \Sigma, \Gamma, \delta, q_0, z, F)$ be a nondeterministic pushdown automaton. The language accepted by M is the set

$$L(M) = \{w \in \Sigma^* : (q_0, w, z) \vdash_M^* (p, \lambda, u), p \in F, u \in \Gamma^*\}$$

In words, the language accepted by M is the set of all strings that can put M into a final state at the end of the string. The final stack content u is irrelevant to this definition of acceptance.

Language accepted by a Pushdown Automata

Example 7.4: Construct an npda for the language

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

Show the ndpa moves in processing the string baab

Idea

z is on stack if #a's = #b's

#0's = #a's - #b's if more a's

#1's = #b's - #a's if more b's

Example 7.5: Construct an npda for the language

$$L=\{ ww^R : w \in \{a, b\}^+ \}$$

Show the ndpa moves in processing the string abba

Pushdown Automata for Context-Free Languages

Theorem 7.1: For every context-free language L, there exists an npda M such that L=L(M). (Assume the L is generated by context-free grammar G.)

Two Steps to build such an npda

(1) Transform productions of G into Greibach normal form

(2) Build an npda from the productions in Greibach normal form

Details of step 2:

$Q = \{ q_0, q_1, q_f \}$, Σ = all terminal symbols, and Γ = all variables,

The moves (transitions) contain the following:

1. $(q_0, \lambda, z) \mid\!- (q_1, \lambda, Sz)$
2. For every production of the form $A \to aX$, a move

   $(q_1, a, A) \mid\!- (q_1, \lambda, X)$
3. $(q_1, \lambda, z) \mid\!- (q_f, \lambda, z)$

Pushdown Automata for Context-Free Languages

Example 7.6: Construct an npda that accepts the language generated by grammar with productions

$S \rightarrow aSbb|a$

production	move
	$(q_0, \lambda, z) \vdash (q_1, \lambda, Sz)$
$S \rightarrow aSA$	$(q_1, a, S) \vdash (q_1, \lambda, SA)$
$S \rightarrow a$	$(q_1, a, S) \vdash (q_1, \lambda, \lambda)$
$A \rightarrow bB$	$(q_1, b, A) \vdash (q_1, \lambda, B)$
$B \rightarrow b$	$(q_1, b, B) \vdash (q_1, \lambda, \lambda)$
	$(q_1, \lambda, z) \vdash (q_f, \lambda, z)$

Example 7.7: Construct an npda that accepts the language generated by grammar with productions

$S \rightarrow aA,$
$A \rightarrow aABC|bB|a,$
$B \rightarrow b,$
$C \rightarrow c.$

Show the ndpa moves in processing the string aaabc and corresponding leftmost derivation of the grammar

Context-Free Grammars for Pushdown Automata

Two properties of an npda

1. Single final state q_f, which is entered if and only if the stack is empty
2. All transitions must have the form

$$\delta(q_i, a, A) = \{c_1, c_2, .., c_n\},$$

where

$$c_j = (q_j, \lambda) \qquad ((q_i, a, A) \vdash (q_j, \lambda, \lambda)) \qquad \text{or}$$
$$c_j = (q_j, BC) \qquad ((q_i, a, A) \vdash (q_j, \lambda, BC))$$

Build the grammar from an npda with the two properties

1. Variable: (q_iAq_j) and Staring variable: (q_0zq_f)
2. Production: $(q_iAq_j) \rightarrow a$ if $(q_i, a, A) \vdash (q_j, \lambda, \lambda)$

$\forall q_l, q_k \in Q$, $(q_iAq_k) \rightarrow a\ (q_jBq_l)\ (q_lCq_k)$ if $(q_i, a, A) \vdash (q_j, \lambda, BC)$

Theorem 7.2: If L=L(M) for some npda M, then L is a context-free language.

Deterministic Pushdown Automata and Deterministic Context-Free Languages

Definition 7.3: A **deterministic pushdown accepter** (dpda) is a pushdown automata as defined in Definition 7.1, subject to the restrictions that, for every $q \in Q$, $a \in \Sigma \cup \{\lambda\}$ and $b \in \Gamma$,

1. $\delta(q, a, b)$ contains at most one element,
2. If $\delta(q, \lambda, b)$ is not empty, then $\delta(q, c, b)$ must be empty for every $c \in \Sigma$.

Definition 7.4: A language L is said to be a deterministic context-free language if and only if there exists a dpda such that $L = L(M)$.

In contrast to finite automata, deterministic and non-deterministic pushdown automata are not equivalent.

Deterministic Pushdown Automata and Deterministic Context-Free Languages

Examples 7.10: (1) $L=\{a^n b^n : n \geq 0\}$ is deterministic context-free language.
(2) $L=\{ww^R: w \in \{a, b\}^+\}$ is not deterministic.

The dpda has
States: $Q = \{q_0, q_1, q_2\}$ and q_0 as its initial and final state
Input alphabet: { a, b }
Stack alphabet { 0, 1 } and z = 0,

The transition rules are

$\delta(q_0, a, 0) = \{ (q_1, 10) \}$

$\delta(q_1, a, 1) = \{ (q_1, 11) \}$

$\delta(q_1, b, 1) = \{ (q_2, \lambda) \}$

$\delta(q_2, b, 1) = \{ (q_2, \lambda) \}$

$\delta(q_2, \lambda, 0) = \{ (q_0, \lambda) \}$