Lecture 8: Pushdown Automaton

Instructor: Ketan Mulmuley
Scriber: Yuan Li

January 29, 2015

1 Nondeterministic Pushdown Automaton

Informally, nondeterministic pushdown automaton (NPDA) is a machine with a read-only input tape, a head of finite states reading input string one character by one character, and a stack where only the top element can be read/pushed/poped.

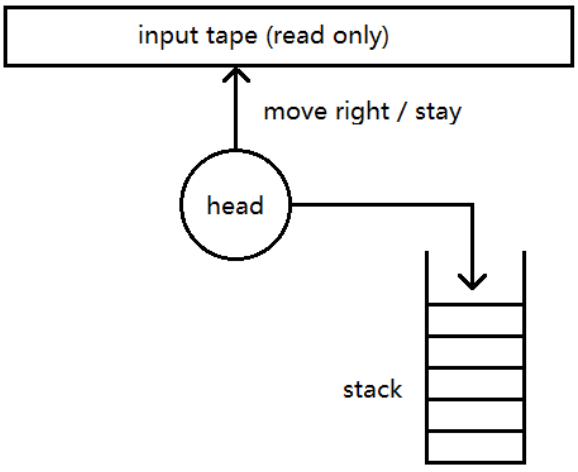

Figure 1: Pushdown Automaton

Depending on the current state, the input symbol, the top element in the stack, the machine nondeterministically moves to a new state, and pop

or push the top element in the stack. It also can ignore the input symbol, nondeterministically move to a new state and changing the top element of the stack, which is called ϵ-move.

There are two types of rules for acceptance, which turns out to be equivalent.

- Acceptance by null-stack: after processing the last symbol, stack becomes empty.
- Acceptance by going into final state: machine goes into an accepting state after processing of the input.

Formally, a nondeterministic pushdown automaton (NPDA) is

$$M = (Q, \Sigma, \Gamma, \delta, q_0, z_0, F \subseteq Q),$$

where Q is a finite set of states, Σ the input alphabet, Γ stack alphabet, δ transition function, q_0 initial state, $z_0 \in \Gamma$ start symbol (at the bottom of stack), F the set of accepting states, and

$$\delta : Q \times (\Sigma \cup \{\epsilon\}) \to \mathcal{P}(Q \times \Gamma^*).$$

If $\delta(q, a, z) = \{(p_1, \gamma_1), (p_2, \gamma_2), \ldots\}$, where $q, p_1, p_2, \ldots \in Q$, $a \in \Sigma$, $z \in \Gamma$ and $\gamma_1, \gamma_2, \ldots \in \Gamma^*$, then if the state is q, input symbol is a, top of stack is z, the machine nondeterministically picks i, and goes into state p_i, replace z by γ_i (the first symbol of γ_i will be the top of the stack), and move input-tape head right by one step.

If $\delta(q, \epsilon, z) = \{(p_1, \gamma_1), (p_2, \gamma_2), \ldots\}$, then if the state is in q, the top of the stack is z, the machine picks i nondeterministically, and goes into state p_o, and replace z by γ_i. (The input symbol is ignored, and the input-tape head does not move.)

If the state is q, input symbol is a, and top of the stack is z, and if both $\delta(q, a, z)$ and $\delta(q, \epsilon, z)$ are nonempty, then the machine nondeterministically choose one of the two options.

2 Instantaneous Description

In order to formally define the language accepted by some NPDA, we need the concept of *instantaneous description (ID)*. At any time, an ID records

state, stack contents, input head position and input string, that is, (q, w, γ), where $q \in Q$ is the current state, $w \in \Sigma^*$ is the unprocessed part of the input, and $\gamma \in \Gamma^*$ is the current stack content (the leftmost symbol of γ is the top of stack).

Define the relation $\longmapsto_M$ between instantaneous descriptions as follows: $\mathrm{ID}_1 \longmapsto_M \mathrm{ID}_2$ if ID_2 *can* follow from ID_1 as an instantaneous description of M, .i.e.,

$$(q, aw, z\alpha) \longmapsto_M (p, w, \beta\alpha)$$

if $\delta(q, a, z)$ contains (p, β), where $p, q, \in Q$, $a \in \Sigma \cup \{\epsilon\}$, $w \in \Sigma^*$ and $\alpha \in \Gamma^*$.

Let $\longmapsto_M^*$ be the reflexive transitive closure. Then the language accepted by NPDA M is formally defined as

- (acceptance by final state) $L(M) = \{w : (q_0, w, z_0) \longmapsto_M^* (p, \epsilon, \gamma)\}$.
- (acceptance by null-stack) $N(M) = \{w : (q_0, w, z_0) \longmapsto_M^* (p, \epsilon, \epsilon)\}$.

3 Deterministic PDA

We say that M is deterministic if

(1) whenever $\delta(q, \epsilon, z)$ is nonempty, $\delta(q, a, z)$ is empty for all $a \in \Sigma$.

(2) $\delta(q, a, z)$ can not contain more than one element for all $a \in \Sigma \cup \{\epsilon\}$.

By default, we always assume the machine is nondeterministic.

For DFA, nondeterminism does not help, in the sense that for any NFA, there is a DFA accepting the same language. What about PDA? The answer is not obvious, which is left as a challenging exercise.

Exercise 3.1 (*)**.** *Prove that* $L = \{ww^R : w \in \{0, 1\}^*\}$ *is accepted by NPDA not not DPDA.*

4 PDA and CFL

The motivation to invent PDA is to give a machine characterization of context-free grammar. The equivalence between CFL and languages accepted by some PDA is our next theorem.

By the way, why the language is called context-free? In the definition of context-free grammar, if there is a variable at some place, we can apply production rules to expand, which does *not* depend on the context. In other words, the meaning of word does not depend on the context. English is not context-free. When I first came to US from India, I was amazed by the meaning of the word "interesting", which could mean good, bad, or anything. When someone says that, you need to look at his face to figure our what does it mean.

Before proving the theorem, let us prove that languages accepted by PDAs by final state acceptance are exactly languages accepted by empty stack.

Proposition 4.1. *If $L = L(M_2)$, then $L = N(M_1)$ for some M_1, where $L(M_2)$ denotes the language accepted by final state, and $N(M_1)$ denotes the language accepted by empty stack.*

Proof. M_1 simulates M_2, and when M_2 enters final state, M_1 goes into "flush" state, and flush the stack. □

Proposition 4.2. *If $L = N(M_1)$, then $L = L(M_2)$ for some M_2.*

Proof. M_2 simulates M_1, and if M_1 removes z_0, M_2 goes into final state. □

Theorem 4.3. *NPDAs accept precisely CFLs.*

Proof. We need to prove two directions: every CFL is accepted by some NPDA; every language accepted by NPDA is CFL.

Let L be a CFL. Without loss of generality, assume $\epsilon \notin L$. By previous class, let $G = (V, T, P, S)$ be a grammar for L in Greibach normal form, where V is the set of variables, T the set of terminals, P the set of productions, S the start symbol. Let

$$M = (\{q\}, T, V, \delta, q, S, \emptyset),$$

where T is the input alphabet, V the stack alphabet, q the start state, and S the start symbol.

M simulates the leftmost derivation of G. Transition function $\delta(q, a, A)$ contains (q, γ) if G has a production of the form $A \to a\gamma$, where $\gamma \in V^*$.

Let us continue the proof next class. □