Formal Languages and Automata, 2024 Fall semester

Lec 09. Pushdown Automata

Eunjung Kim

Pushdown automata

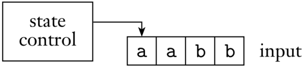

Figure 2.11, Sipser 2012

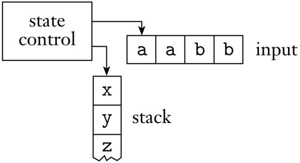

Figure 2.12, Sipser 2012

PUSHDOWN AUTOMATA

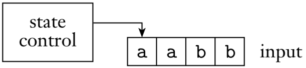

Figure 2.11, Sipser 2012

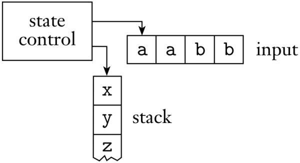

Figure 2.12, Sipser 2012

PDA: INFORMAL DESCRIPTION

A pushdown automata has three components: an input tape, a state (control), and a stack. At each step, PDA does the following

- current state: q.
- read from input: the first symbol a of the input tape; possibly ϵ.
- pops off stack: the first symbol x at (the top of) the stack; possibly ϵ.
- transition: depending on (q, a, x), PDA
  1. updates the current state q to a new state q'
  2. pushes a symbol y to the stack; possibly ϵ.

PDA FOR $L = \{w \cdot w^R : w \in \{0,1\}^*\}$

(INFORMAL) PDA RECOGNIZING A PALINDROME

- It has three states: $q_{start}, q_{push}, q_{pop\&match}, q_{accept}$.
- Starting at q_{start}, it pushes \$ to stack and updates to q_{push}.
- Stays in q_{push} while: it reads the symbol b from input and pushes b to stack.
- "Guess" the middle of the word; implemented as an update from q_{push} to $q_{pop\&match}$
- Stays in $q_{pop\&match}$ while: it reads the symbol b from input, pops the symbol b; they match.
- Moves $q_{pop\&match}$ to q_{accept} if: there is nothing to read in the input when \$ is popped.
- In all other cases (e.g. "mismatch" between the input content and stack symbol): "dies"

FORMAL DEFINITION OF PDA

A PUSHDOWN AUTOMATA IS

a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$, where all entries except for q_0 is a finite set:

- Q is the set of states.
- Σ is the input alphabet.
- Γ is the stack alphabet.
- $\delta : Q \times \Sigma_\epsilon \times \Gamma_\epsilon \to \mathcal{P}(Q \times \Gamma_\epsilon)$ is the transition function.
- $q_0 \in Q$ is the start state.
- $F \subseteq Q$ is the set of accept states.

Language recognized by a PDA

PDA accepting a string

A pushdown automata $P = (Q, \Sigma, \Gamma, \delta, q_0, F)$ accepts an input string $w \in \Sigma^*$ if

- I w can be written as $w_1 w_2 \cdots w_n$, where $w_i \in \Sigma_\epsilon$,
- II there is a sequence $r_0, r_1, \ldots, r_n$ of states, and
- III a sequence of strings $s_0, s_1, \ldots, s_n$ of strings over Γ^*,

such that the following holds:

1. $r_0 = q_0$ and $s_0 = \epsilon$,
2. $s_i = xt$, $s_{i+1} = yt$ and $(r_{i+1}, y) \in \delta(r_i, w_{i+1}, x)$ hold for some $x, y \in \Gamma_\epsilon$ and $t \in \Gamma^*$
3. $r_n \in F$.

INSTANTANEOUS DESCRIPTION (A.K.A CONFIGURATION)

CONFIGURATION OF PDA

For a pushdown automata $P = (Q, \Sigma, \Gamma, \delta, q_0, F)$, an instantaneous description (ID), or equivalently a configuration of P is a triple (q, w, γ). Essentially

- q is the current state.
- w is the remaining input string.
- γ is the string in the stack, read from top to bottom.

If $(q', y) \in \delta(q, a, x)$, then we write

$$(q, aw, x\beta) \vdash (q', w, y\beta).$$

This means that one can reach the configuration $(q', w, y\beta)$ from $(q, aw, x\beta)$ in a single step. The notation $\vdash^*$ is used when one is reach from the other in $n \geq 0$ steps.

Language recognized by a PDA

Language recognized by PDA

For a PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, F)$, the language recognized by PDA P is defined as

- the set of all strings in Σ^* accepted by P, or equivalently
- the set of all strings $w \in \Sigma^*$ such that $(q_0, w, \epsilon) \vdash^* (q, \epsilon, \gamma)$ for some $q \in F$ and $\gamma \in \Gamma^*$.

We write as $L(P)$ the language recognized by PDA P.

PDA FOR $L = \{0^n 1^n : n \geq 0\}$

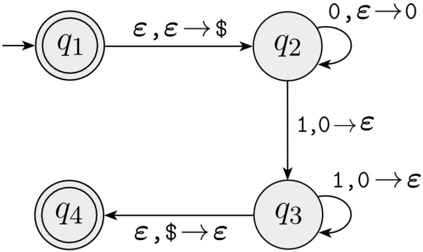

Figure 2.15, Sipser 2012

PDA for $L = \{0^n 1^n : n \geq 0\}$

Formal description of PDA recognizing L: it is a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_1, F)$ as follows.

- $Q = \{q_1, q_2, q_3, q_4\}$
- $\Sigma = \{0, 1\}$
- $\Gamma = \{0, \$\}$
- the transition function δ is given as the transition table below.
- Start state is q_1
- $F = \{q_1, q_4\}$

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

From example 2.14, Sipser 2012

PDA FOR $L = \{w \cdot w^R : w \in \{0,1\}^*\}$

PDA FOR $L = \{w \in \{a, b\}^* : |w|_a = |w|_b\}$

Equivalence of PDA and CFL

Theorem

A language is context-free if and only if some pushdown automaton recognizes it.

Proof outline for ($\Rightarrow$).

- From a context-freem grammar $G = (V, \Sigma, R, S)$, we aim to construct a PDA P such that $L(P) = L(G)$.

Equivalence of PDA and CFL

Theorem

A language is context-free if and only if some pushdown automaton recognizes it.

Proof outline for ($\Rightarrow$).

- From a context-freem grammar $G = (V, \Sigma, R, S)$, we aim to construct a PDA P such that $L(P) = L(G)$.
- Key 1: we design PDA P which accepts w if and only if w has a leftmost derivation in G.

Equivalence of PDA and CFL

Theorem

A language is context-free if and only if some pushdown automaton recognizes it.

Proof outline for ($\Rightarrow$).

- From a context-freem grammar $G = (V, \Sigma, R, S)$, we aim to construct a PDA P such that $L(P) = L(G)$.
- Key 1: we design PDA P which accepts w if and only if w has a leftmost derivation in G.
- Key 2: P simulates a leftmost derivation of $w \in L(G)$ by

Equivalence of PDA and CFL

Theorem

A language is context-free if and only if some pushdown automaton recognizes it.

Proof outline for ($\Rightarrow$).

- From a context-freem grammar $G = (V, \Sigma, R, S)$, we aim to construct a PDA P such that $L(P) = L(G)$.
- Key 1: we design PDA P which accepts w if and only if w has a leftmost derivation in G.
- Key 2: P simulates a leftmost derivation of $w \in L(G)$ by
  1. "matching" the input symbol and the stack symbol if the stack symbol is an element of Σ.
  2. "replacing" the stack symbol A by z if A is a variable of G and there is a rule $A \rightarrow z$.