Lecture 7 – Pushdown automata

NTIN071 Automata and Grammars

Jakub Bulín (KTIML MFF UK)

Spring 2024

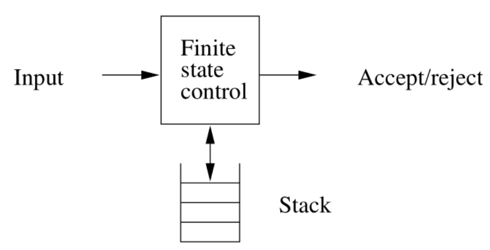
** Adapted from the Czech-lecture slides by Marta Vomlelová with gratitude. The translation, some modifications, and all errors are mine.*

Recap of Lecture 6

- Reducing a grammar: removing ϵ-productions, unit productions, useless symbols
- Chomsky Normal Form of a context-free grammar
- Pumping lemma for context-free languages, application: proving non-context-freeness
- Testing membership in a context-free language: the CYK algorithm

2.9 Pushdown automata

Pushdown automaton (PDA)



- an extension of ϵ–NFA, additional feature: a stack memory
- the stack has its own stack alphabet Γ (can contain Σ or not)
- at each step we pop the top stack symbol X, make a decision based on (q, a, X), push some word $\gamma \in \gamma^*$
- the stack can rememeber an infinite amount of information
- PDA define context-free languages, nondeterminism is important: deterministic PDA only recognize a proper subset of context-free languages (unlike DFA vs. NFA)

The definition

A pushdown automaton (PDA): $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, where

- Q is finite, nonempty set of states
- Σ is a finite, nonempty input alphabet
- Γ is a finite, nonempty stack alphabet
- δ is the transition function,

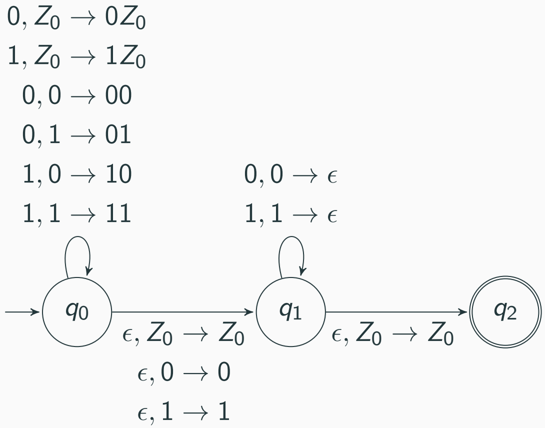
  $$\delta \colon Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma \to \mathcal{P}_{FIN}(Q \times \Gamma^*)$$

  $\delta(q, a, X) \ni (p, \gamma)$ where p is the new state and γ a finite string of stack symbols that replace X on top of the stack
- $q_0 \in Q$ is the initial state
- $Z_0 \in \Gamma$ is the initial stack symbol (bottom of the stack); the only symbol on the stack at the beginning
- F is a set of accepting (final) states; may be undefined if our PDA accepts by empty stack

One transition of a PDA

- read one input letter ($a \in \Sigma$) or do an ϵ-transition ($a = \epsilon$)
- pop X from the top of the stack
- based on a, X, and the current state q nondeterministically choose one of finitely many options $(p, \gamma) \in \delta(q, a, X)$
- switch to the new state p
- push the finite string γ to the stack (the first symbol of Γ is now on top)
- pop: $\gamma = \epsilon$, read only: $\gamma = X$, push: $\gamma = \gamma' X$

An example: $L = \{ww^R \mid w \in \{0,1\}^*\}$



q_0 read input letters pushing them onto the stack; guess the middle (nondeterministically), jump to q_1

q_1 compare input with stack, consuming both; if empty stack (we see the bottom), accept by jumping to q_2; no input can remain

Example cont'd: full description of the PDA

$$P = (\{q_0, q_1, q_2\}, \{0, 1\}, \{0, 1, Z_0\}, \delta, q_0, Z_0, \{q_2\})$$

$\delta(q_0, 0, Z_0) = \{(q_0, 0Z_0)\}$ $\delta(q_0, 1, Z_0) = \{(q_0, 1Z_0)\}$	push input onto stack, leave the bottom
$\delta(q_0, 0, 0) = \{(q_0, 00)\}$ $\delta(q_0, 0, 1) = \{(q_0, 01)\}$ $\delta(q_0, 1, 0) = \{(q_0, 10)\}$ $\delta(q_0, 1, 1) = \{(q_0, 11)\}$	stay in q_0, push input onto stack
$\delta(q_0, \epsilon, Z_0) = \{(q_1, Z_0)\}$ $\delta(q_0, \epsilon, 0) = \{(q_1, 0)\}$ $\delta(q_0, \epsilon, 1) = \{(q_1, 1)\}$	jump to q_1 without changing stack
$\delta(q_1, 0, 0) = \{(q_1, \epsilon)\}$ $\delta(q_1, 1, 1) = \{(q_1, \epsilon)\}$	pop stack and match with input
$\delta(q_1, \epsilon, Z_0) = \{(q_2, Z_0)\}$	we have ww^R, go to accepting state