Theory of Computer Science

B9. Context-free Languages: Push-Down Automata

Gabriele Röger

University of Basel

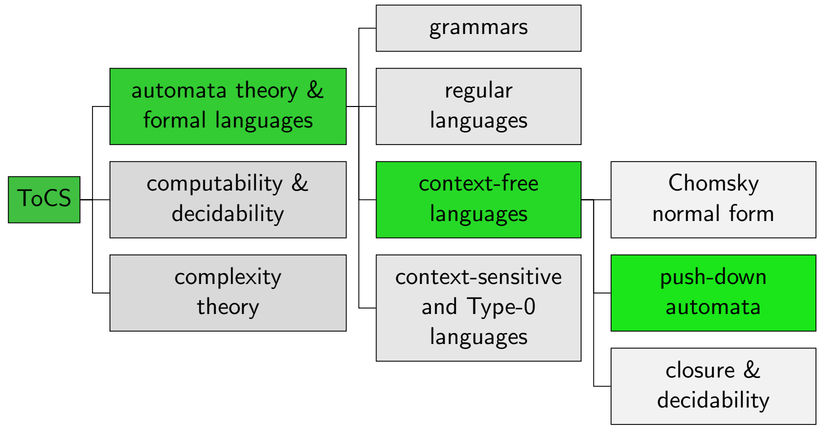
March 31, 2025

Theory of Computer Science
March 31, 2025 — B9. Context-free Languages: Push-Down Automata

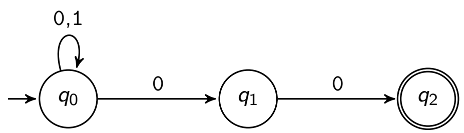
B9.1 Push-Down Automata

B9.1 Push-Down Automata

Content of the Course



Limitations of Finite Automata

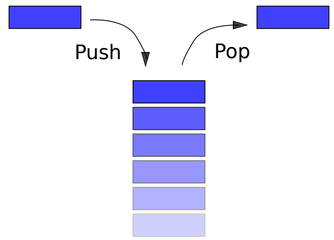


- Language L is regular.
  $\Longleftrightarrow$ There is a finite automaton that recognizes L.
- What information can a finite automaton "store" about the already read part of the word?
- Infinite memory would be required for $L = \{x_1 x_2 \dots x_n x_n \dots x_2 x_1 \mid n > 0, x_i \in \{\mathtt{a}, \mathtt{b}\}\}$.
- therefore: extension of the automata model with memory

Stack

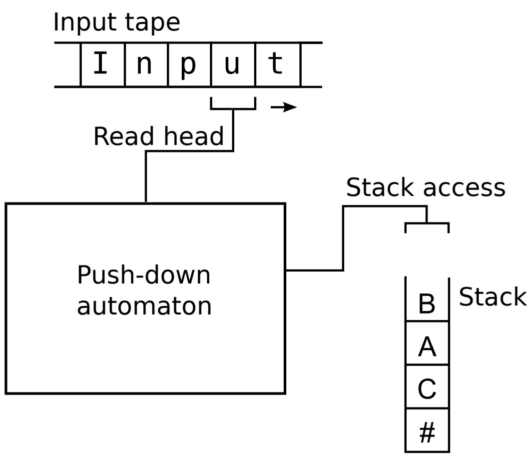
A stack is a data structure following the last-in-first-out (LIFO) principle supporting the following operations:

- push: puts an object on top of the stack
- pop: removes the object at the top of the stack
- peek: returns the top object without removing it



German: Keller, Stapel

Push-down Automata: Visually



German: Kellerautomat, Eingabeband, Lesekopf, Kellerzugriff

Push-down Automaton for $\{\mathtt{a}^n\mathtt{b}^n \mid n \in \mathbb{N}_0\}$: Idea

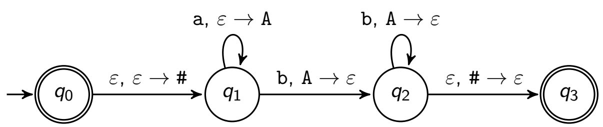
- As long as you read symbols a, push an A on the stack.
- As soon as you read a symbol b, pop an A off the stack as long as you read b.
- If reading the input is finished exactly when the stack becomes empty, accept the input.
- If there is no A to pop when reading a b, or there is still an A on the stack after reading all input symbols, or if you read an a following a b then reject the input.

Push-down Automata: Non-determinism

- PDAs are non-deterministic and can allow several next transitions from a configuration.
- Like NFAs, PDAs can have transitions that do not read a symbol from the input.
- Similarly, there can be transitions that do not pop and/or push a symbol off/to the stack.

Deterministic variants of PDAs are strictly less expressive, i. e. there are languages that can be recognized by a (non-deterministic) PDA but not the deterministic variant.

Push-down Automaton for $\{a^n b^n \mid n \in \mathbb{N}_0\}$: Diagram



Push-down Automata: Definition

Definition (Push-down Automaton)

A push-down automaton (PDA) is a 6-tuple $M = \langle Q, \Sigma, \Gamma, \delta, q_0, F \rangle$ with

- Q finite set of states
- Σ the input alphabet
- Γ the stack alphabet
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\}) \to \mathcal{P}(Q \times (\Gamma \cup \{\varepsilon\}))$ the transition function
- $q_0 \in Q$ the start state
- $F \subseteq Q$ is the set of accept states

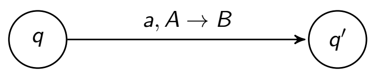
German: Kellerautomat, Eingabealphabet, Kelleralphabet, Überführungsfunktion

Push-down Automata: Transition Function

Let $M = \langle Q, \Sigma, \Gamma, \delta, q_0, F \rangle$ be a push-down automaton.

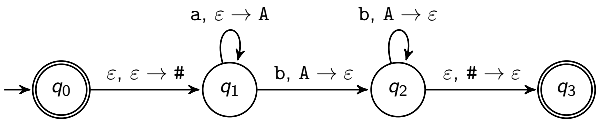
What is the Intuitive Meaning of the Transition Function δ?

- $\langle q', B \rangle \in \delta(q, a, A)$: If M is in state q, reads symbol a and has A as the topmost stack symbol, then M can transition to q' in the next step popping A off the stack and pushing B on the stack.



- special case $a = \varepsilon$ is allowed (spontaneous transition)
- special case $A = \varepsilon$ is allowed (no pop)
- special case $B = \varepsilon$ is allowed (no push)

Push-down Automaton for $\{a^n b^n \mid n \in \mathbb{N}_0\}$: Formally



$M = \langle \{q_0, q_1, q_2, q_3\}, \{a, b\}, \{A, \#\}, \delta, q_0, \{q_0, q_3\} \rangle$ with

$\delta(q_0, a, A) = \emptyset$	$\delta(q_0, b, A) = \emptyset$	$\delta(q_0, \varepsilon, A) = \emptyset$
$\delta(q_0, a, \#) = \emptyset$	$\delta(q_0, b, \#) = \emptyset$	$\delta(q_0, \varepsilon, \#) = \emptyset$
$\delta(q_0, a, \varepsilon) = \emptyset$	$\delta(q_0, b, \varepsilon) = \emptyset$	$\delta(q_0, \varepsilon, \varepsilon) = \{(q_1, \#)\}$
$\delta(q_1, a, A) = \emptyset$	$\delta(q_1, b, A) = \{(q_2, \varepsilon)\}$	$\delta(q_1, \varepsilon, A) = \emptyset$
$\delta(q_1, a, \#) = \emptyset$	$\delta(q_1, b, \#) = \emptyset$	$\delta(q_1, \varepsilon, \#) = \emptyset$
$\delta(q_1, a, \varepsilon) = \{(q_1, A)\}$	$\delta(q_1, b, \varepsilon) = \emptyset$	$\delta(q_1, \varepsilon, \varepsilon) = \emptyset$
$\delta(q_2, a, A) = \emptyset$	$\delta(q_2, b, A) = \{(q_2, \varepsilon)\}$	$\delta(q_2, \varepsilon, A) = \emptyset$
$\delta(q_2, a, \#) = \emptyset$	$\delta(q_2, b, \#) = \emptyset$	$\delta(q_2, \varepsilon, \#) = \{(q_3, \varepsilon)\}$
$\delta(q_2, a, \varepsilon) = \emptyset$	$\delta(q_2, b, \varepsilon) = \emptyset$	$\delta(q_2, \varepsilon, \varepsilon) = \emptyset$

and $\delta(q_3, x, y) = \emptyset$ for all $x \in \{a, b, \varepsilon\}$, $y \in \{A, \#, \varepsilon\}$

Push-down Automata: Accepted Words

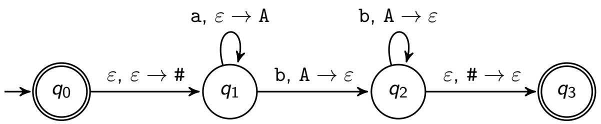
Definition

A PDA $M = \langle Q, \Sigma, \Gamma, \delta, q_0, F \rangle$ accepts input w
if it can be written as $w = w_1 w_2 \dots w_m$ where each $w_i \in \Sigma \cup \{\varepsilon\}$
and sequences of states $r_0, r_1, \dots, r_m \in Q$ and
strings $s_0, s_1, \dots, s_m \in \Gamma^*$ exist
that satisfy the following three conditions:

1. $r_0 = q_0$ and $s_0 = \varepsilon$
2. For $i = 0, \dots, m-1$, we have $(r_{i+1}, b) \in \delta(r_i, w_{i+1}, a)$, where $s_i = at$ and $s_{i+1} = bt$ for some $a, b \in \Gamma \cup \{\varepsilon\}$ and $t \in \Gamma^*$.
3. $r_m \in F$

The strings s_i represent the sequence of stack contents.

Push-down Automaton for $\{a^n b^n \mid n \in \mathbb{N}_0\}$



The PDA accepts input aabb.

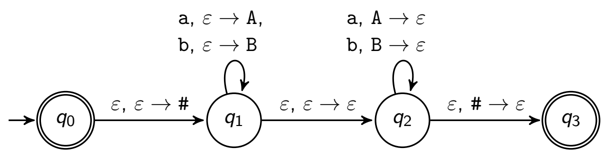
PDA: Recognized Language

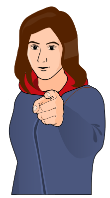
Definition (Language Recognized by an PDA)

Let M be a PDA with input alphabet Σ.

The language recognized by M is defined as
$\mathcal{L}(M) = \{w \in \Sigma^* \mid w \text{ is accepted by } M\}$.

Recognized Language: Exercise





What language does this PDA recognize?

PDAs Recognize Exactly the Context-free Languages

Theorem
A language L is context-free if and only if L is recognized by a push-down automaton.

PDAs: Exercise (if time)

Assume you want to have a possible transition from state q to state q' in your PDA that

- processes symbol c from the input word,
- can only be taken if the top stack symbol is `A`,
- does not pop `A` off the stack, and
- pushes `B`.

What problem do you encounter? How can you work around it?

Summary

- Push-down automata (PDAs) extend NFAs with memory (only stack access)
- The languages recognized by PDAs are exactly the context-free languages.