Learning Outcomes

By the end of this session you should be able to...

• ... *define* pushdown automata

• ... *design* pushdown automata
Another look at FA

- a finite automaton (DFA, NFA or $\varepsilon$NFA) can be thought of as a machine that scans a string;
- a string $x$ is accepted if there exists a computation that ends in a final state.
Pushdown Automata

- adding an ingredient:
  - the control unit has access to memory: a stack;
  - the stack has no pre-set size limit.
PDAs, formally

- a PDA is a 7-tuple \( M = (Q, \Sigma, \Gamma, \delta, s, \bot, F) \)
  - \( Q \) is a finite set of states;
  - \( \Sigma \) is the input alphabet;
  - \( \Gamma \) is the stack alphabet;
  - \( \delta \) is the transition relation (see next slide);
  - \( s \in Q \) is the start state;
  - \( \bot \in \Gamma \) is the initial stack symbol;
  - \( F \subseteq Q \) is the set of final states.
PDA transitions

\[ \delta \subseteq (Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma) \times (Q \times \Gamma^*) \]

- \((q, \sigma, \gamma), (q', \gamma_1 \gamma_2 \cdots \gamma_k)\) \in \delta \text{ intuitively means:}
  - if in state \(q\), reading \(\sigma\) and \(\gamma\) on top of the stack, then pop \(\gamma\) off the stack and
  - go to state \(q'\) pushing \(\gamma_1 \gamma_2 \cdots \gamma_k\) on the stack (\(\gamma_k\) first, \(\gamma_1\) last)

\[ \sigma; \; \gamma \; \gamma_1 \gamma_2 \cdots \gamma_k \]

\(q\) \; \(q'\)
Configuration

• A complete description of a computation at a point in time:
  o the current state;
  o the part of input still to be read;
  o the contents of the stack.

• it is an element of $Q \times \Sigma^* \times \Gamma^*$

• eg: $(q, \sigma_1 \ldots \sigma_k, \gamma_1 \ldots \gamma_l \perp)$
**Semantics**

- we define a relation $\rightarrow$ between configurations;
- let $s \in \Sigma^*$ and $g, h \in \Gamma^*$, write:
  $$ (q, \sigma s, \gamma g) \rightarrow (q', s, hg) \text{ when } ((q, \sigma, \gamma), (q', h)) \in \delta $$
- write
  $$ (q, s, \gamma g) \rightarrow (q', s, hg) \text{ when } ((q, \epsilon, \gamma), (q', h)) \in \delta $$
- note that $\rightarrow$ is nondeterministic! (this is a consequence of how we defined $\delta$)
Acceptance

- Given configurations $C_1$, $C_2$, write $C_1 \Rightarrow C_2$ if there exist configurations $C'_1, \ldots, C'_k$ for some $k \in \mathbb{N}$ such that

$$C_1 \rightarrow C'_1 \rightarrow \cdots \rightarrow C'_k \rightarrow C_2$$

- $x \in \Sigma^*$ is accepted by $M$ if there exists $f \in F$ such that

$$(s, x, \bot) \Rightarrow (f, \epsilon, g)$$

- Note: all the input has to be consumed!
Definition. Suppose $M = (Q, \Sigma, \Gamma, \delta, s, \bot, F)$ is a PA.

$L(M) = \{ x \mid \exists f \in F. (s, x, \bot) \Rightarrow (f, \epsilon, g) \}$
Example 1

- PA that accepts the language $\{a^n b^n \mid n \geq 1\}$.

- Let $\Sigma \defeq \{a, b\}$, $\Gamma \defeq \{a, \bot\}$, $Q \defeq \{0, 1, 2\}$ with 0 the initial state and 2 the final state;

![Diagram](attachment:image.png)
Example ctd

• \( \delta \) contains the following pairs:

\[
((0, a, \bot), (0, a \bot)) \quad (1)
\]

\[
((0, a, a), (0, aa)) \quad (2)
\]

\[
((0, b, a), (1, \epsilon)) \quad (3)
\]

\[
((1, b, a), (1, \epsilon)) \quad (4)
\]

\[
((1, \epsilon, \bot), (2, \epsilon)) \quad (5)
\]
Example 1 - Simulation

\[(0, \quad aaabbb, \quad \bot) \rightarrow (0, \quad aabbb, \quad a \bot) \quad (1)\]

\[(0, \quad aabbb, \quad a \bot) \rightarrow (0, \quad bb, \quad aa \bot) \quad (2)\]

\[(0, \quad bb, \quad aa \bot) \rightarrow (1, \quad b, \quad a \bot) \quad (3)\]

\[(1, \quad b, \quad a \bot) \rightarrow (1, \quad \epsilon, \quad \bot) \quad (4)\]

\[(1, \quad \epsilon, \quad \bot) \rightarrow (2, \quad \epsilon, \quad \epsilon) \quad (5)\]

COMP 2210: Lecture 7 – p. 13
Other langs accepted by PAs

- all the regular languages
  - any $\varepsilon$ NFA can be considered as a PA that doesn’t use its stack;
- palindromes
- balanced parentheses
Acceptance by empty stack

- Recall that we accept a string $x$ when there exists $f \in F$ such that:

$$ (s, x, \perp) \Rightarrow (f, \epsilon, g) $$

- refer to this as acceptance by final state.

- there is another possible definition: $M$ accepts $x$ if there exists some $q \in Q$ such that:

$$ (s, x, \perp) \Rightarrow (q, \epsilon, \epsilon) $$

- this is acceptance by empty stack.
Final state vs Empty stack

• the two notions of PA can simulate each other, i.e.
  - any PA $M$ that accepts by final state can be altered to a PA $M'$ that accepts by empty stack;
  - any PA $M$ that accepts by empty stack can be altered to a PA $M'$ that accepts by final state.
Learning Outcomes

you should be able to . . .

• . . . define pushdown automata

• . . . design pushdown automata