Pushdown Automata

- *Pushdown Automata* is an extension of the nondeterministic finite automaton with ε-transitions.
- A pushdown automaton is essentially an *ε-NFA* with the addition of a *stack*. The stack can be read, pushed and popped only at the top, just like the stack data structure.
- The presence of a stack means that, unlike the finite automaton, the pushdown automaton can *remember* an infinite amount of information.
- However, unlike a general-purpose computer, which also has the ability to remember arbitrarily large amounts of information, the pushdown automaton can only access the information on its stack in a last-in-first-out way.
- *Two different versions of the PDA*: One that accepts by entering an accepting state, like finite automata do; and another version that accepts by emptying its stack, regardless of the state it is in.

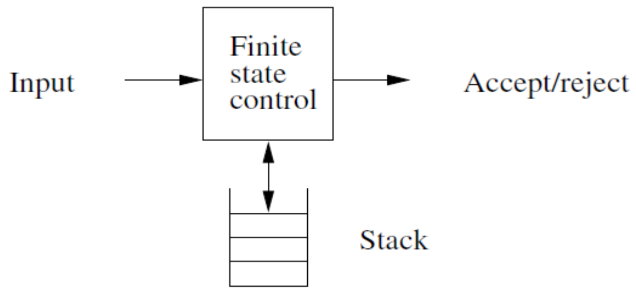


Fig: A pushdown automaton is essentially a finite automaton with a stack data structure.

- A *finite-state control* reads inputs, one symbol at a time.
- A pushdown automaton is allowed to observe the symbol at the top of the stack and to base its transition on its *current state*, the *input symbol* and the *symbol at the top of stack*. Alternatively, it may make a *spontaneous* transition, using ε as its input instead of an input symbol.

In one transition, the pushdown automaton:

1. Consumes from the input the symbol that it uses in the transition. If ε is used for the input, then no input symbol is consumed.
2. Goes to a new state, which may or may not be the same as the previous state.
3. Replaces the symbol at the top of the stack by any string.
   - The string could be *ε*, which corresponds to a *pop of the stack*.

- It could be the same symbol that appeared at the top of the stack previously; i.e. no change to the stack is made.
- It could also replace the top stack symbol by one other symbol, which in effect changes the top of the stack but does not push or pop it.
- Finally, the top stack symbol could be replaced by two or more symbols, which has the effect of (possibly) changing the top stack symbol, and then pushing one or more new symbols onto the stack.

Formal Definition of Pushdown Automata

A Pushdown Automaton (PDA) is a *7-tuple*, that is, a system which consists of 7 elements. We describe a PDA, M as follows:

$M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, where

Q - finite nonempty set of *states*;

Σ - finite nonempty set of input symbols, i.e., *input alphabet*;

Γ- finite nonempty set of stack symbols, i.e., *stack alphabet*. It is the set of symbols that are allowed to push onto the stack, usually $\Sigma \subset \Gamma$;

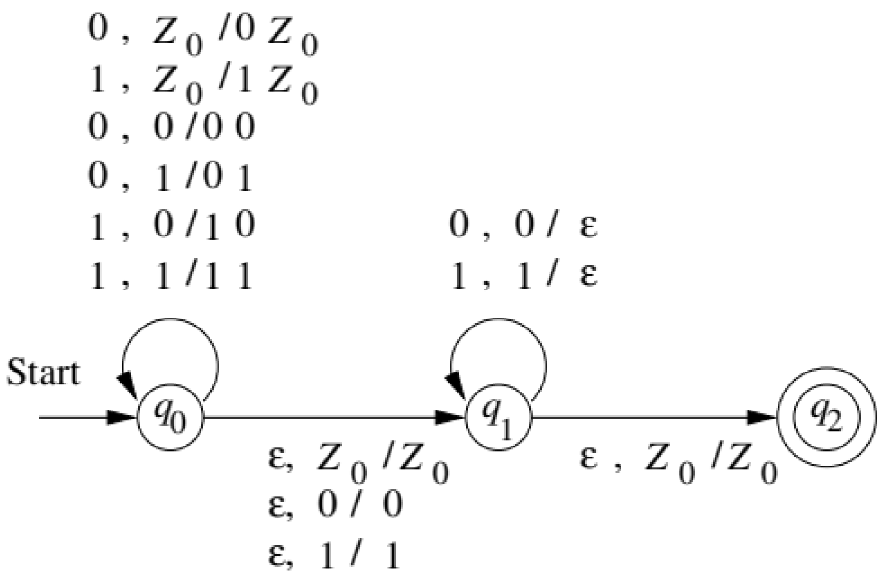
δ - transition function, it takes as argument a *triple (q, a, X)* where $q \in Q$, $a \in (\Sigma \cup \{\varepsilon\})$ and $X \in \Gamma$. The output of δ is a finite set of pairs (p, γ), where *p* is the *new state*, and γ is the *string of stack symbols* that replaces X at the top of the stack. For instance, if $\gamma = \varepsilon$, then the stack is *popped*, if $\gamma = X$, then the stack is *unchanged*, if $\gamma = YZ$, then X is replaced by Z, and Y is pushed onto the stack.

q_0 – initial/start state, $q_0 \in Q$;

Z_0 – initial/start stack symbol, $Z_0 \in \Gamma$; Initially, the PDA's stack consists of one instance of this symbol, and nothing else.

F - set of final or accepting states, $F \subseteq Q$.

Example: Design a PDA *P* to accept the language L_{wwr} over {0, 1} - often referred to as "w-w-reversed", is the even-length palindromes over alphabet {0, 1}.



Graphical Notation of PDA:

a) The nodes correspond to the states of the PDA.

b) An arrow labeled *Start* indicates the start state, and doubly circled states are accepting, as for finite automata.

c) The arcs correspond to transitions of the PDA in the following sense. An arc labeled $a, X/\alpha$ from state q to state p means that $\delta(q, a, X)$ contains the pair (p, α), perhaps among other pairs. That is, the arc label tells what input is used, and also gives the old and new tops of the stack.

Instantaneous Description of a PDA:

PDA goes from configuration to configuration in response to input symbols (or sometimes ε), but unlike the finite automaton where the state is the only thing that we need to know about the automaton, the PDA's configuration involves both the state and the contents of the stack.
The configuration of a PDA is represented by a triple (q, w, γ), where

1. q is the state
2. w is the remaining input
3. γ is the stack contents

Conventionally we show the top of the stack at the left end of γ and the bottom at the right end. Such a triple is called an *instantaneous description,* or ID, of the pushdown automaton.

Let's find out the ID's of the PDA above on input 1111:

$$\begin{tikzcd}
(q_0, 1111, Z_0) \arrow[d] \arrow[dr] & & \\
(q_0, 111, 1Z_0) \arrow[d] \arrow[dr] & (q_1, 1111, Z_0) \arrow[r] & (q_2, 1111, Z_0) \\
(q_0, 11, 11Z_0) \arrow[d] \arrow[dr] & (q_1, 111, 1Z_0) \arrow[r] & (q_1, 11, Z_0) \arrow[d] \\
(q_0, 1, 111Z_0) \arrow[d] \arrow[dr] & (q_1, 11, 11Z_0) \arrow[dr] & (q_2, 11, Z_0) \\
(q_0, \varepsilon, 1111Z_0) \arrow[d] & (q_1, 1, 111Z_0) \arrow[d] & (q_1, 1, 1Z_0) \arrow[d] \\
(q_1, \varepsilon, 1111Z_0) & (q_1, \varepsilon, 11Z_0) & (q_1, \varepsilon, Z_0) \arrow[d] \\
 & & (q_2, \varepsilon, Z_0)
\end{tikzcd}$$

The Language of a PDA:

There are two approaches for a PDA to accept its input:

1. Acceptance by Final State
2. Acceptance by Empty Stack

Acceptance by Final State:

Let $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ be a PDA. Then L(P), the language accepted by P by final state, is:

$$\{w \mid (q_0, w, Z_0) \overset{*}{\underset{P}{\vdash}} (q, \epsilon, \alpha)\}$$

for some state q in F and any stack string α. That is, starting in the initial ID with w waiting on the input, P consumes w from the input and enters an accepting state. The contents of the stack at that time are irrelevant.

Acceptance by Empty Stack:

For each PDA P = (Q, Σ, Γ, δ, q_0, Z_0, F), we also define,

$$N(P) = \{w \mid (q_0, w, Z_0) \overset{*}{\vdash} (q, \epsilon, \epsilon)\}$$

for any state q. That is, *N(P)* is the set of inputs *w* that *P* can consume and at the same time empty its stack.

Deterministic Pushdown Automata:

While PDAs are by definition allowed to be nondeterministic, the deterministic subcase is quite important.

Definition of a Deterministic PDA: A PDA is deterministic if there is never a choice of move in any situation. These choices are of two kinds. If δ(q, a, X) contains more than one pair, then surely the PDA is nondeterministic. However, even if δ(q, a, X) is always a singleton, we could still have a choice between using a real input symbol, or making a move on ε. Thus, a PDA is defined to be a deterministic (DPDA), if and only if the following conditions are met:

1. $\delta(q, a, X)$ has at most one member for any q in Q, a in Σ or $a = \epsilon$, and X in Γ.

2. If $\delta(q, a, X)$ is nonempty, for some a in Σ, then $\delta(q, \epsilon, X)$ must be empty.