Formal Definition of a Pushdown Automaton (PDA)

Recall that for any set S, the power set 2^S is the set of all subsets of S.

Also, given any finite set A (representing an alphabet), let $A_\varepsilon = A \cup \{\varepsilon\}$ (representing the same alphabet but with ε added to it).

A *pushdown automaton* P is a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$, where Q, Σ, and Γ are finite sets and

- Q is the set of states;
- Σ is the input alphabet;
- Γ is the stack alphabet;
- $\delta : Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \longrightarrow 2^{Q \times \Gamma_\varepsilon}$ is the transition function;
- $q_0 \in Q$ is the start state; and
- $F \subseteq Q$ is the set of accept states.

Computation Performed by an PDA

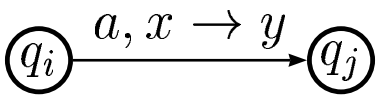
A PDA P *accepts* input $w \in \Sigma^*$ if w can be written as $w = w_1 w_2 \ldots w_m$, with each $w_i \in \Sigma_\varepsilon$, and there exists a sequence of states $r_0, r_1, \ldots, r_m \in Q$ and a sequence of strings $s_0, s_1, \ldots, s_m \in \Gamma^*$ satisfying:

1. $r_0 = q_0$ and $s_0 = \varepsilon$ [says P starts in its start state with an empty stack];
2. for $i = 0, 1, \ldots, m-1$ we have $(r_{i+i}, b) \in \delta(r_i, w_{i+1}, a)$, where $s_i = at$ and $s_{i+1} = bt$ for some $a, b \in \Gamma_\varepsilon$ and $t \in \Gamma^*$ [says state and stack contents are updated properly based on the current state, stack contents, and next input symbol]; and
3. $r_m \in F$ [says ends in an accept state when input runs out].

The *language recognized by* P is $L(P) = \{w \mid P \text{ accepts } w\}$.

Graphical Representation of a PDA

Generalization of the graphical representation of an NFA.



$$\begin{aligned} q_i, q_j &\in Q \\ a &\in \Sigma_\varepsilon \\ x, y &\in \Gamma_\varepsilon \end{aligned}$$

Means:

When in state q_i, if a read from the input tape and popping the stack yields x, then push y onto the stack and make transition to state q_j. (But this may be only one of several possibilities because of nondeterminism.)

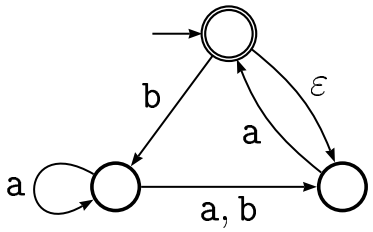
Formally, this corresponds to $(q_i, y) \in \delta(q_i, a, y)$.

Special cases:

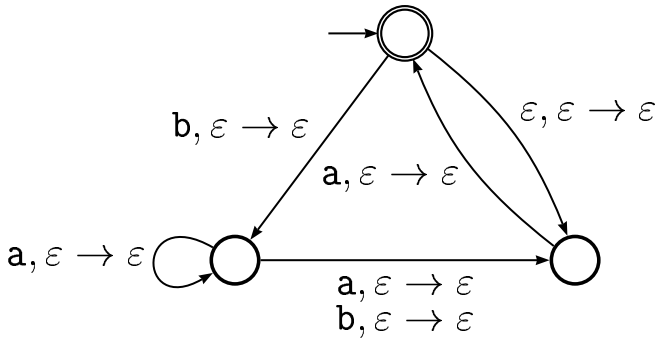
- $a = \varepsilon$ means don't read input
- $x = \varepsilon$ means don't pop stack
- $y = \varepsilon$ means don't push anything onto stack

PDA Example 1

Consider this NFA:



Equivalent PDA:

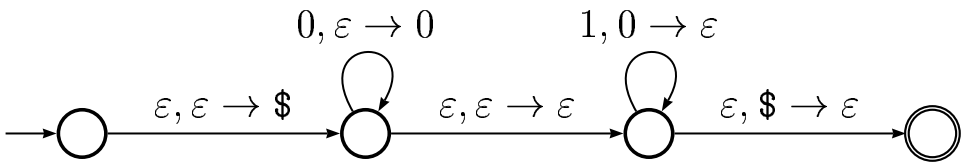


Any NFA can be viewed as a PDA that ignores its stack.

PDA Example 2

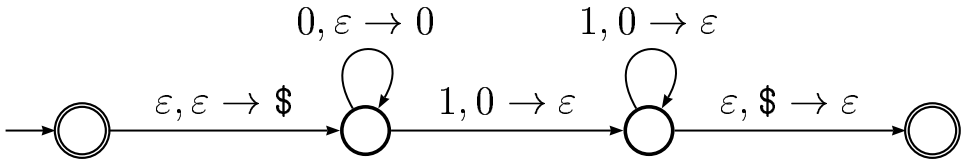
PDA that recognizes $\{0^n 1^n \mid n \geq 0\}$.

$\Sigma = \{0, 1\}$, $\Gamma = \{0, \$\}$



Standard convention: Use a special stack alphabet symbol to mark the bottom of the stack.

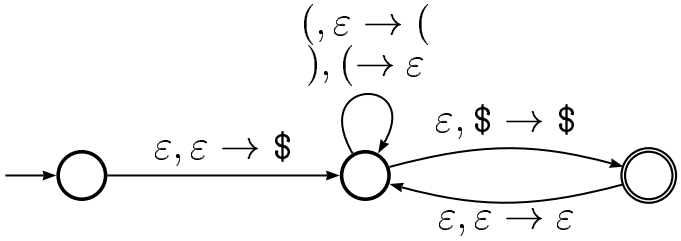
Here's another PDA for the same language that behaves deterministically (except when it rejects because its only computation path dies):



PDA Example 3

Recognizes matching-parentheses language.

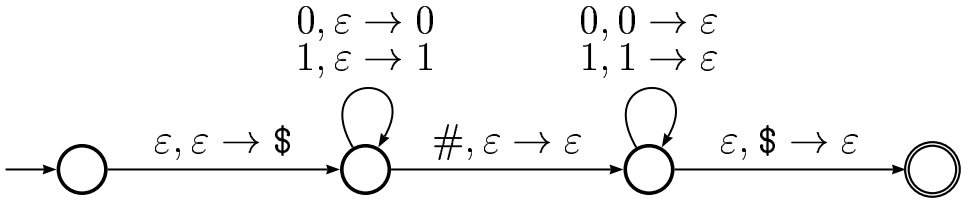
Here, $\Sigma = \{(,)\}$.



PDA Example 4

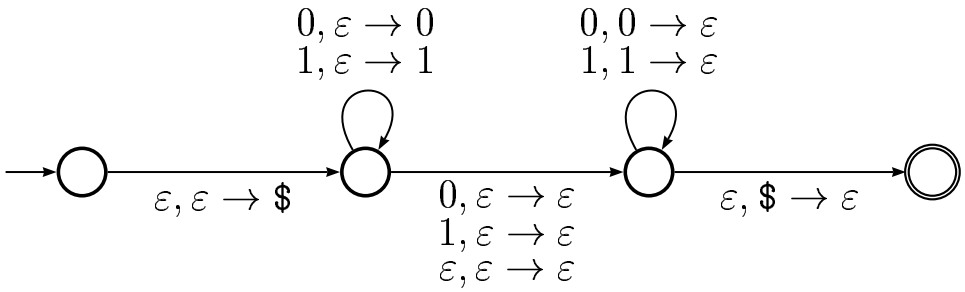
Recognizes the language $\{w\#w^{\mathcal{R}} \mid w \in \{0,1\}^*\}$.

Here, $\Sigma = \{0, 1, \#\}$.



PDA Example 5

Recognizes the language $\{w \mid w \in \{0,1\}^* \text{ and } w \text{ is a palindrome}\}$.



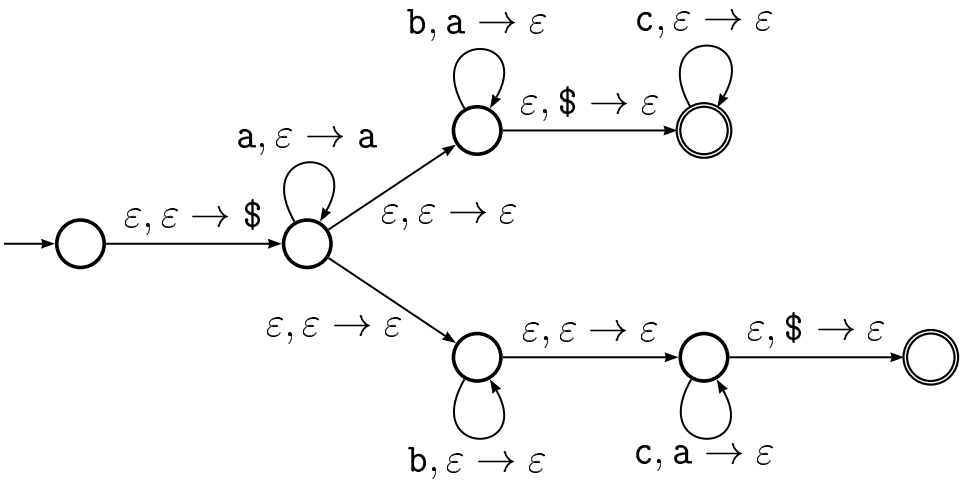
Behaves almost like the previous PDA except that it nondeterministically "guesses" when it's at the middle of the input string.

3 possibilities:

- length is odd and middle symbol is a 0
- length is odd and middle symbol is a 1
- length is even

PDA Example 6

Recognizes the language $\{\mathtt{a}^i\mathtt{b}^j\mathtt{c}^k \mid i, j, k \geq 0 \text{ and } i = j \text{ or } i = k\}$.



The upper branch checks that the number of a's equals the number of b's.

The lower branch checks that the number of a's equals the number of c's.

Informal description:

First, push all the a's onto the stack. Then, nondeterministically

- check that the number of b's matches the number of a's on the stack and skip over all the c's at the end; or
- skip over all the b's in the middle and check that the number of c's matches the number of a's on the stack.

Failure at any point in this process leads to rejection.