A stack based machine (PushDown Automaton)

In addition to a finite state transition system, we have an (infinite capacity) stack.

\[ (Q, \Sigma, \delta, \emptyset, \delta_0, \Gamma, \Delta) \]

Two separate terminating conditions:

1. The stack is empty when input string is exhausted.
2. We are in a final state when input is exhausted.
\[ L = \{ \omega \in \omega^* \mid \omega \in (0+1)^* \} \]

\[ M = (\{q_1, q_2\}, \{0, 1, c\}, \{R, B, G\}, s, q_1, \emptyset) \]

1. \( \delta(q_1, 0, R) = \{(q_1, BR), \_\} \) \( \delta(q_1, 1, R) = (q_1, GR) \)
2. \( \delta(q_1, 0, B) = (q_1, BB) \) \( \delta(q_1, 1, B) = (q_1, GB) \)
3. \( \delta(q_1, 0, G) = (q_1, BG) \) \( \delta(q_1, 1, G) = (q_1, GG) \)
4. \( \delta(q_2, 0, B) = (q_2, \varepsilon) \) \( \delta(q_2, 1, G) = (q_2, \varepsilon) \)
5. \( \delta(q_2, \varepsilon, R) = (q_2, \varepsilon) \)
6. \( \delta(q_1, c, R) = (q_2, R) \) \( \delta(q_1, c, G) = (q_2, G) \)
7. \( \delta(q_1, c, B) = (q_2, B) \)
Instantaneous Description (1D) of a PDA

The complete information about a PDA can be obtained from:
1) Current state
2) the current symbol it is scanning
3) the stack contents

1D: \((q, a \cdot w, \alpha)\) \(q \in Q\)
\(a \in \Sigma, w \in \sum^*\)
\(\alpha \in T^*\)

\(I_0\) : \((q_0, \varepsilon, Z_0)\)

\(I_0 \rightarrow I_1 \rightarrow I_2 \rightarrow \cdots \rightarrow I_f\)

\(I_0 \vdash^* I_f\)

\(I_j \vdash^* I_{j+1}\)

\((p, a \cdot w, A \alpha) \rightarrow (q, w, \beta \alpha)\)

\(\delta(p, a, A)\) must contain \((q, \beta)\)
PDA's that accept by empty stack

\[ w \text{ is accepted by the PDA iff} \]

\[ (q_0, w, Z_0) \xrightarrow{*} (p, \epsilon, \epsilon) \quad p \in Q \]

PDA's that accept by final state

\[ (q_0, w, Z_0) \xrightarrow{*} (q_f, \epsilon, \lambda) \]

\[ q_f \in F \quad \lambda \in \Sigma^* \]

Thus let \( L \) be a language accepted by a PDA using empty stack. Then \( L \) is also accepted by some PDA that accepts using final state.

and vice versa
Theorem: Suppose \( L \) is a CFL generated by a CFG \( G = (V, T, S, P) \). Then we can design a PDA \( M \) (accepting using empty stack) s.t.
\[
L(M) = L
\]
The proof uses Greibach Normal Form.

Theorem: Given a PDA \( M \) that accepts a language \( L \). Then we can design a CFG \( G \) s.t.
\[
L(G) = L
\]
The deterministic version of PDA is not equivalent to PDA (non-det)
\[ L = \{ w w | w \in (0+1)^* \} \]

\[ L' : \text{all strings not of the form } w w \]

\[ \overline{01011} \]