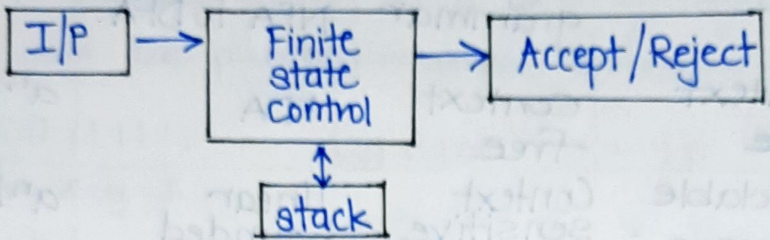
UNIT 4 : PUSHDOWN AUTOMATA (PDA)

- Push down automata in essence is a non deterministic finite automata with ε transitions permitted and one additional capability i.e a stack on which it can store a string of stack symbols
- PDA can only access the information on its stack in LIFO way
- PDA's can recognize all CFL's



$L_{wwr} = \{ww^r \mid w \in (0+1)^*\}$

Procedure: Read every symbol & put it onto the stack after the mid of string pop each symbol & match each reading symbol with top of stack. If match then pop top of stack

The PDA can remember an infinite amount of information.

PDA involves $\{Q, \Sigma, \Gamma, \delta, q_0, z_0, F\}$

Q - all the states ~~set~~ finite set

Σ - finite set of input symbols or string

Γ - A finite stack alphabet that are allowed to be pushed onto stack.

δ - Transition function.

$\delta(q, \Sigma \mid \Gamma)$ ... q - state, Σ - input symbol., Γ - stack operation

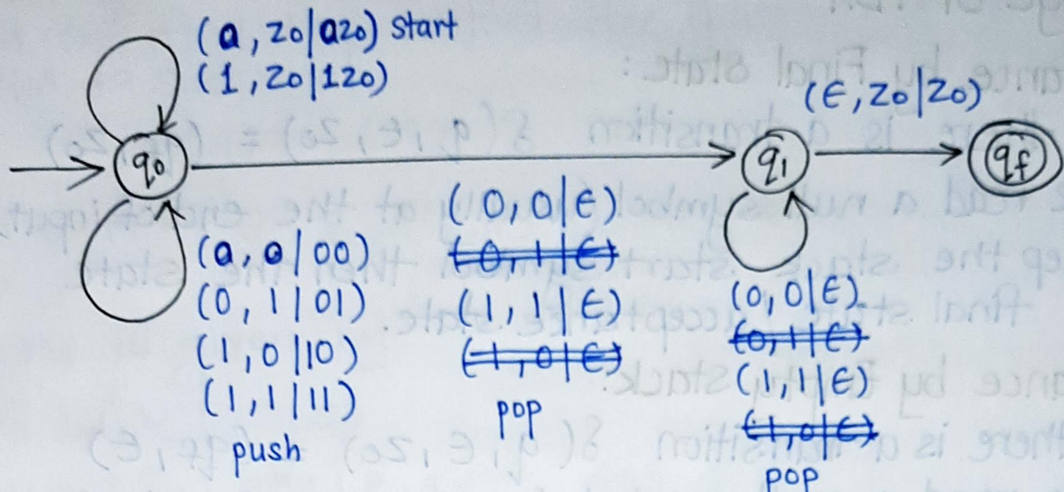
$\delta(q, \Sigma \mid \Gamma) = (p, \gamma)$ ... p - new state, γ - stack string.

q_0 - initial state.

z_0 - bottom of stack / start of stack

F - final state / set of accepting states.

PDA for wcw^R



L_{ww^r} = 010 010

$\delta(q_0, 0, z_0) \longrightarrow (q_0, 0z_0)$

$\delta(q_0, 1, 0) \longrightarrow (q_0, 10)$

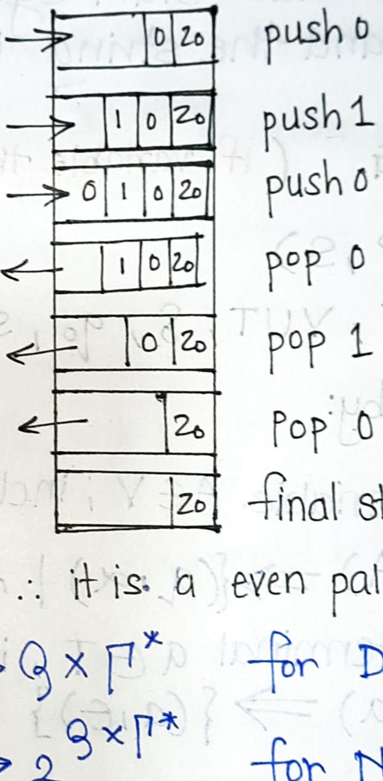
$\delta(q_0, 0, 1) \longrightarrow (q_0, 01)$

$\delta(q_0, 0, 0) \longrightarrow (q_1, \epsilon)$

$\delta(q_1, 1, 1) \longrightarrow (q_1, \epsilon)$

$\delta(q_1, 0, 0) \longrightarrow (q_1, \epsilon)$

$\delta(q_1, \epsilon, z_0) \longrightarrow (q_f, z_0)$



Thus the string got accepted ∴ it is a even palindrome.

PDA : $Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma \rightarrow Q \times \Gamma^*$ for DPDA

$Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma \rightarrow 2^{Q \times \Gamma^*}$ for NPDA

$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad}_{\text{for single stack PDA}}$

Language of PDA

1) Acceptance by Final State:

when there is a transition $\delta(q, \epsilon, z_0) = (q_f, z_0)$ i.e we read a null symbol (usually at the end of input) and keep the stack start symbol then the state q_f is final state / acceptance state.

2) Acceptance by Empty stack:

when there is a transition $\delta(q, \epsilon, z_0) = (q_f, \epsilon)$ i.e we read a null symbol (eos - end of string) and pop the stack start symbol then the stack is empty and the string is accepted.

PDA from CFG (if variable then push if terminal then pop)

$G = (V, T, P, S)$

$M = \{\{q\}, T, V \cup T, \delta, q_0, S, \phi\}$

δ are given by:

1. For each variable $A \in V$, include a transition

$$\delta(q, \epsilon, A) \Rightarrow \{(q, \alpha) \mid A \rightarrow \alpha \text{ is a production}\}$$

2. For each terminal $a \in T$, include a transition

$$\delta(q, a, a) \Rightarrow \{(a, \epsilon)\}$$

example: $S \rightarrow 0S1 \mid 00 \mid 11$

$$\therefore \delta(q, \epsilon, S) \rightarrow \{(q, 0S1), (q, 00), (q, 11)\}$$

$$\delta(q, 0, 0) \rightarrow \{(q, \epsilon)\}$$

$$\delta(q, 1, 1) \rightarrow \{(q, \epsilon)\}$$

CFG from PDA

we can find the context free grammar G for any PDA M such that

$$L(G) = L(M)$$

PDA $M = (Q, \Sigma, \Gamma, \delta, q_0, z_0, \phi)$

then

CFG is given by

$G = (V, T, P, S)$

$V = \{S, [q X p]\ p, q \in Q \text{ and } X \in T\}$

P for CFG are:

1. Add the following productions for the start symbol

$S \rightarrow [q_0\, z_0\, a_i]$ for each $a_i \in Q$ where z_0 is start symbol.

2. For each transition of form:

$\delta(q_i, a, B) \Rightarrow (q_j, C)$

where a) $q_i, q_j \in Q$

b) a belongs to $(\Sigma \cup \epsilon)$

c) B and C $\in (\Gamma \cup \epsilon)$

then for each $q \in Q$ we add production.

$[q_i\, B\, q] \rightarrow a[q_i\, C\, q]$

3. For each transition of form.

$\delta(q_i, a, B) \Rightarrow (q_j, C_1 C_2)$

where a) $q_i, q_j \in Q$

b) $a \in (\Sigma \cup \epsilon)$

c) B, C_1 and $C_2 \in \Gamma$

then for each $p_1, p_2 \in Q$ we add production

$[q_i\, B\, p_1] \rightarrow a[q_j\, C_1\, p_2][p_2\, C_2\, p_1]$

DPDA :

A push down auto mata for which reading one input on a state with current top of stack will only lead to 1 operation on stack.

i.e when on $(q_0, a, z_0) \rightarrow (q_0, az_0)$ is the only transition for a on q_0 when top of stack is z_0.

Hence the CFL parsing doesnt have ambiguity.

NPDA :

A push down automata for which reading one input signal on a state with current top of stack can lead to multiple transitions / operation on stack

i.e when on $(q_0, a, z_0) \rightarrow (q_0, az_0)$ & (q_0, z_0)
thus leads to ambiguity.

Hence CFL parsing has ambiguity.