Push Down Automaton (PDA)

Like ε-NFA but also operates a stack. Action in one step includes

1) Consumption of a symbol from input (may be ε)
2) Change of state
3) Pop and push on the stack

A PDA is a 7-tuple $P=(Q,\sum,\Gamma,\delta,q_0,Z_0,F)$ where

Q = finite set of states

$\sum$ = alphabet

Γ= stack alphabet

$\delta : Q \times \sum_e \times \Gamma \rightarrow P(Q \times \Gamma^*)$ non-deterministic

$q_0 \in Q$ start state

$Z_0 \in \Gamma$ bottom of stack symbol

F - a subset of Q : set of final states

If $\delta(q,a,\beta)$ contains (p,u) then the PDA in state q with top of stack β while consuming $a \in \sum_e$ can change state to p, pop β and then push $u \in \Gamma^*$. Because of non-determinism there can be zero or more such actions. Configuation of PDA P is given by (p,w',γ) where p = current stack, w' = remaining input and γ = current stack contents.

If $(p,u) \in \delta(q,a,\beta)$ then the configuration

$(q,aw',\beta\gamma)$ yields in one step $(p,w',u\gamma)$. We indicate this by $(q,aw',\beta\gamma) \rightarrow (p,w',u\gamma)$. If $c_1=c_2$ or $c_1 = c_1' \rightarrow c_2' \rightarrow .... \rightarrow c_k' = c_2$ we say that c_1 yields c_2 ($c_1 \rightarrow^* c_2$). Starting configuration is (q_0,w,Z_0) where w = input string. We associate two languages with PDA P.

Language accepted by P using final state :

$L(P) = \{w \in \sum^* \mid (q_0,w,Z_0) \rightarrow^* (q_f,\varepsilon,\gamma)\}$ for some $q_f \in F$ and any $\gamma \in \Gamma^*$

and language accepted by P using empty stack

$N(P)=\{w \in \Sigma^* \mid (q_0,w,Z_0) \rightarrow^* (q,\varepsilon,\varepsilon)\}$ for any $q \in Q$.

We now state four theorems and their corollaries about these languages without proof.

Theorem 1 : If $L=N(P_N)$ for some PDA P_N then we can construct a PDA P such that $L = L(P)$.

Theorem 2 : If $L = L(P)$ for some PDA P then we can construct a PDA P_N such that $L = N(P_N)$.

Theorem 3 : Given a CFG G, we can construct a PDA P_N such that $N(P_N) = L(G)$.

Corollary 3.1 : Given a CFG G, we can construct a PDA P such that $L(P) = L(G)$.

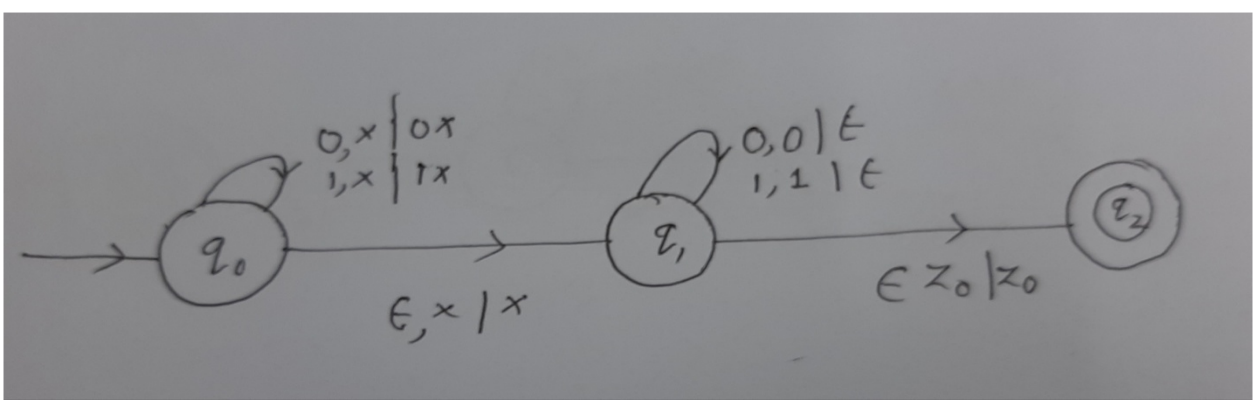
Theorem 4 : Given a PDA P_N, we can construct a CFG G such that $L(G) = N(P_N)$.

Corollary 4.1 : Given a PDA P, we can construct a CFG G such that L(G) = L(P).

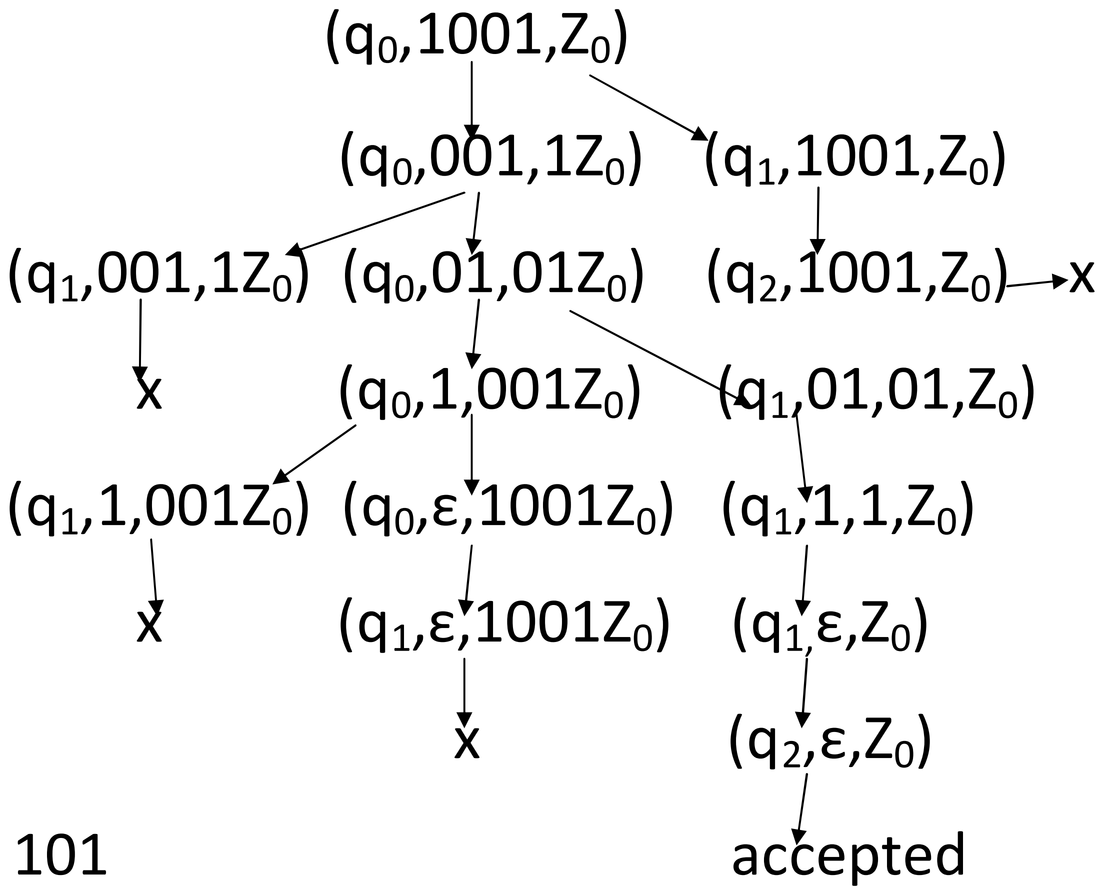
Design of a PDA : We can use transition diagrams. A transition with label a,β/u means a Є $\sum_e$ will be consumed from input, β is the top of the stack which will be popped and after that u Є $\sum^*$ will be pushed. If β = X, it will mean any symbol of Γ.

Design a PDA P such that L = L(P) for L = $\{ww^R \mid w \in \{0,1\}^*\}$, the language of even-length palindromes.

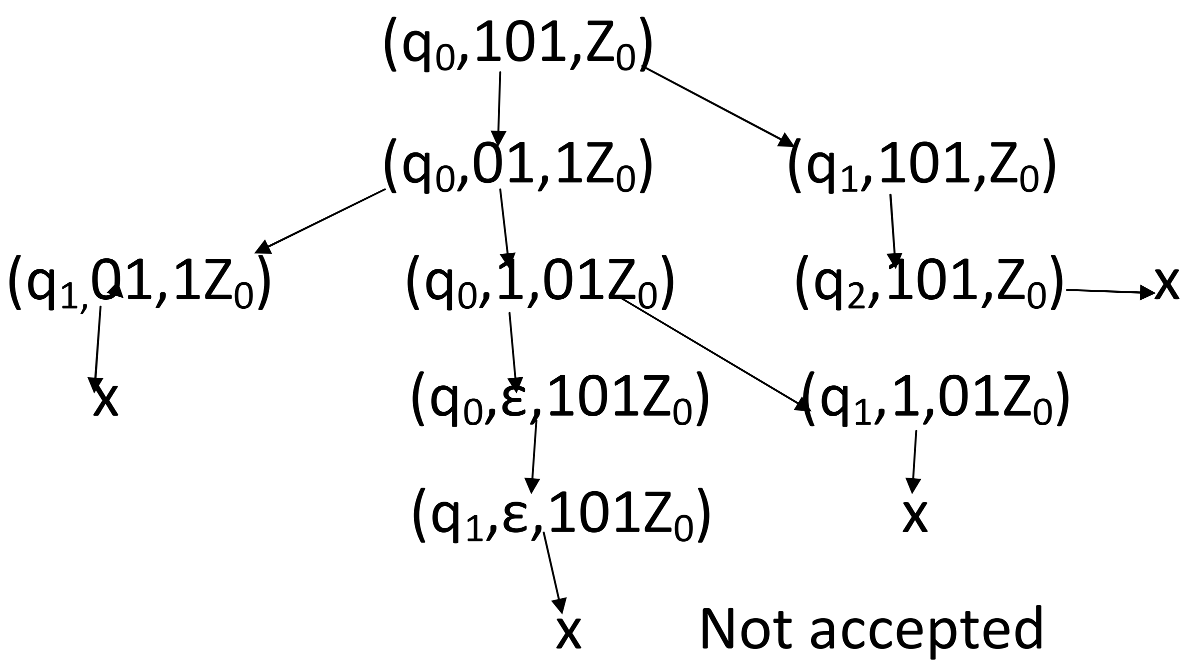
$\sum = \{0,1\}$, $\Gamma = \{0 , 1, Z_0\}$



Computation with 1001



input 101



HW Design PDA P s.t. L(P) ie the language accepted by P using final states is

a) Palindromes of odd length
b) All palindromes