PUSHDOWN AUTOMATA

Regular language can be charaterized as the language accepted by finite automata. Similarly, we can characterize the context-free language as the langauge accepted by a class of machines called "Pushdown Automata" (PDA).

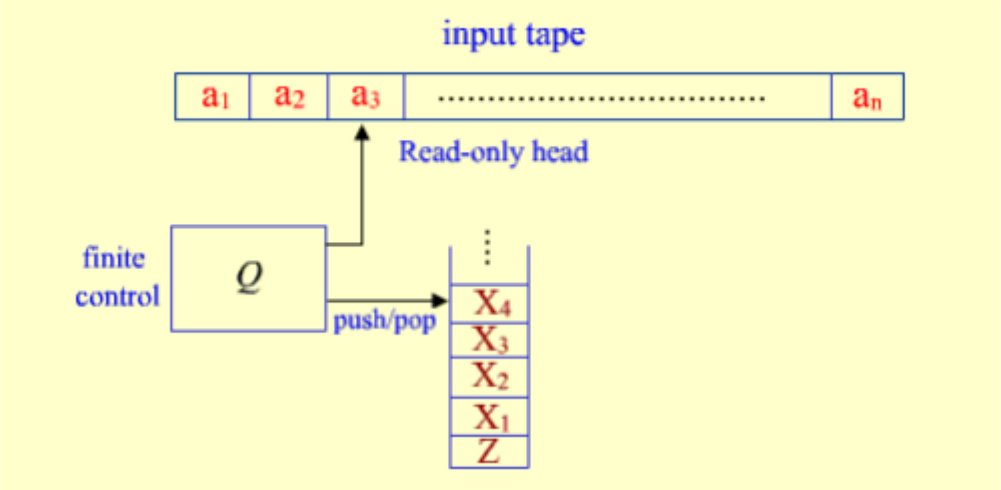
It is observed that FA have limited capability. This is due to the "finite memory" and "no external memory" involved with them.

A PDA is simply an ε- NFA augmented with an "external stack memory". In this Transitions are modified to accommodate stack operations.

This "Stack" or "pushdown store" can be used to record potentially unbounded information. It is due to this memory management capability with the help of the stack that a PDA can overcome the memory limitations that prevents a FA to accept many interesting languages like $\{a^n b^n / n \geq 1\}$

A PDA can store an unbounded amount of information on the stack, its access to the information on the stack is limited. To read down into the stack the top elements must be popped off and are lost. Due to this limited access to the information on the stack, a PDA still has some limitations and cannot accept some other interesting languages.

we can view the pushdown automata informally as shown in the figure.



As shown in figure, a PDA has three components: an **input tape** with read only head, a **finite control** and a **pushdown store or stack**.

The input head is read-only and may only move from left to right, one symbol (or cell) at a time.

Its moves are determined by:

1. The current state
2. The current input symbol (or ε), and
3. The current symbol on top of its stack

In each choice, the PDA can:

1. Change state, and alsc
2. Replace the top symbol on the stack by a sequence of zero or more symbols.

In each step, the PDA, based on the symbol at the top of the stack, the input symbol it is currently reading, and its present state, it can push a sequence of symbols onto the stack, move its read-only head one cell (or symbol) to the right, and enter a new state, as defined by the transition rules of the PDA.

Formally, A pushdown automaton (PDA) is a seven-tuple

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

Where i) Q is a finite nonempty set of states

ii) Σ is a finite nonempty set of input symbols (input alphabet)

iii) Γ is a finite nonempty set of stack symbols called stack alphabet

iv) $q_0 \in Q$ is the initial state

v) $z_0 \in \Gamma$ is a initial stack symbol

vi) $F \subseteq Q$ is a set of final states

vii) δ is a transition function, where

$$\delta: Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \rightarrow \text{finite subsets of } Q \times \Gamma^*$$

.

Ex: Let the PDA is

$$M = (Q, \Sigma, \Gamma, \delta, q, z_0, F)$$

Where $Q=\{q,p,f\}$ $\Sigma=\{0,1\}$ $\Gamma=\{Z_0,X\}$ $F=\{f\}$ and δ is given as follows.

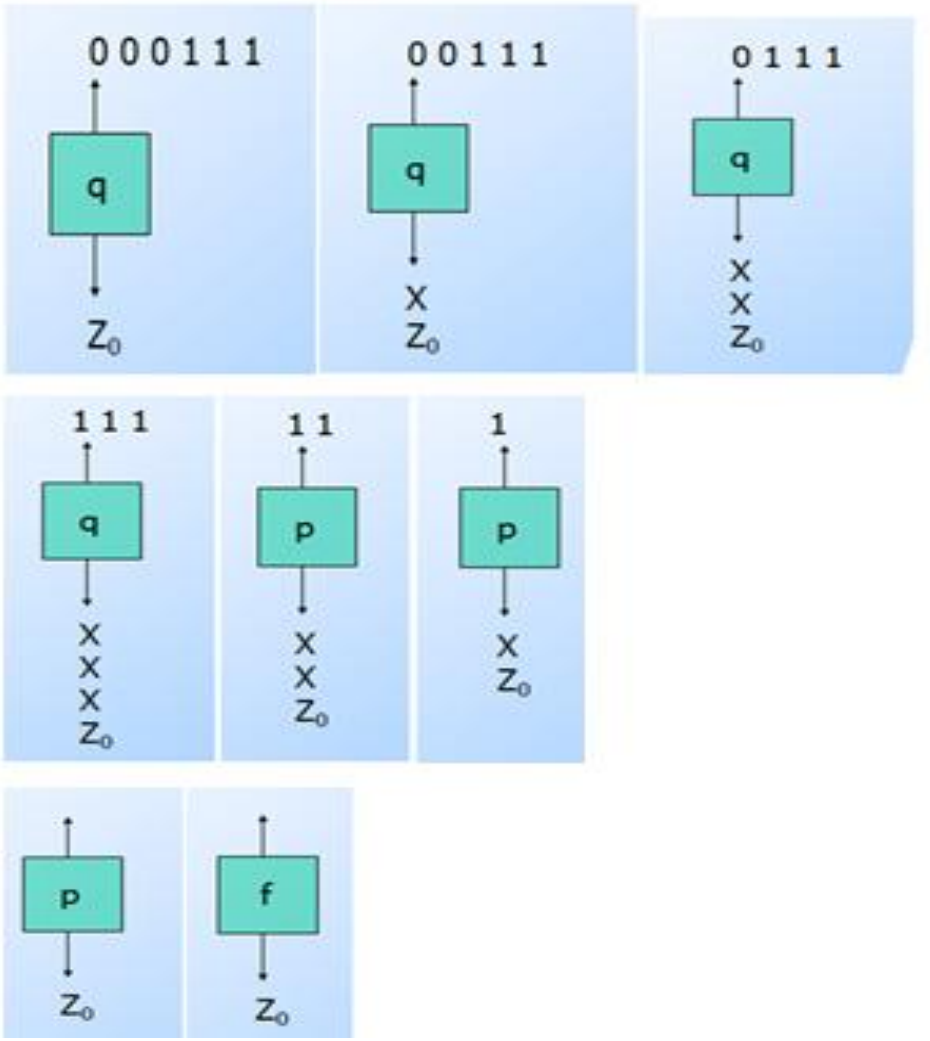
1) $\delta(q, 0, Z_0) = \{(q, XZ_0)\}$. These two rules cause one X to be
2) $\delta(q, 0, X) = \{(q, XX)\}$. pushed onto the stack for each 0 read

3) $\delta(q, 1, X) = \{(p, \varepsilon)\}$. When we see a 1, go to state p and pop one X.

4) $\delta(p, 1, X) = \{(p, \varepsilon)\}$. Pop one X per 1.

5) $\delta(p, \varepsilon, Z_0) = \{(f, Z_0)\}$. Accept at bottom

The above example is a PDA that accepts the language $\{0^n1^n \mid n \geq 1\}$ by final state


000111
q
Z0
00111
q
X
Z0
0111
q
X
X
Z0
111
q
X
X
X
Z0
11
p
X
X
Z0
1
p
X
Z0
p
Z0
f
Z0

Ex2: consider the PDA

$P = (Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$

Where $Q=\{q_0,q_1,q_2\}$

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

$\Gamma =\{0,1,z_0\}$

$F=\{q_2\}$

and δ is given as follows.

1) $\delta(q_0, 0, Z_0) = \{(q_0, 0Z_0)\}$.
2) $\delta(q_0, 1, Z_0) = \{(q_0, 1Z_0)\}$.

3) $\delta(q_0, 0, 0) = \{(q_0, 00)\}$.
4) $\delta(q_0, 0, 1) = \{(q_0, 01)\}$.
5) $\delta(q_0, 1, 0) = \{(q_0, 10)\}$.
6) $\delta(q_0, 1, 1) = \{(q_0, 11)\}$.

7) $\delta(q_0, \varepsilon, Z_0) = \{(q_1, Z_0)\}$.
8) $\delta(q_0, \varepsilon, 0) = \{(q_1, 0)\}$.
9) $\delta(q_0, \varepsilon, 1) = \{(q_1, 1)\}$.

10) $\delta(q_1, 0, 0) = \{(q_1, \varepsilon)\}$.
11) $\delta(q_1, 1, 1) = \{(q_1, \varepsilon)\}$.

12) $\delta(q_1, \varepsilon, Z_0) = \{(q_2, Z_0)\}$.

Graphical Notation for PDA:

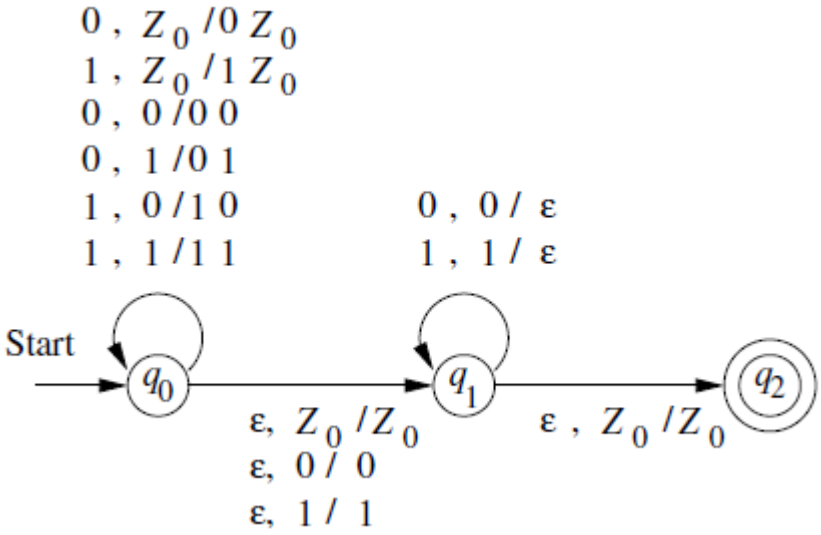
A PDA can also be represented by means of transition diagram, where

a) the nodes correspond to the states of the PDA

b) An arrow labeled start indicates the start state, and doubly circled states are final/accepting states

c) The arcs correspond to transitions of the PDA as follows.

An arc labeled a, X/α from state q to state p means that δ(q,a,X) contains the paid (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



Note:

1. In general, the behavior of a PDA is non deterministic

2. The PDA cannot take a transition when stack is empty.

Instantaneous Descriptions(ID's) of a PDA:

Let M = (Q, Σ, Γ, δ, q_0, z_0, F) be a given PDA. We use an ID to describe the current condition of a PDA and the working of a PDA can be described in terms of change of ID's

An ID is triplet (q,x,α) where:

1. q Є Q is the current state.
2. x Є Σ* is the input to be processed
3. α Є Γ* is the content of the stack

Ex:

Consider the ID (q, $a_1a_2a_3$…….a_n , $z_1z_2z_3$……z_m). This describes the PDA when the

a) current state is q ,

b)the input string to be processed is $a_1a_2a_3$…….a_n and

c) the stack has $z_1z_2z_3$……z_m with z_1 at the top , z_2 is the second element from the top etc. and z_m is the lowest element in stack.

Definition Let A be a pda. A move relation, denoted by $\vdash$, between IDs is defined as

$$(q,\ a_1a_2\ \ldots\ a_n,\ Z_1Z_2\ \ldots\ Z_m) \vdash (q',\ a_2a_3\ \ldots\ a_n,\ \beta Z_2\ \ldots\ Z_m)$$

if $\delta(q,\ a_1,\ Z_1)$ contains $(q',\ \beta)$.

Note:

We also use the symbol $\vdash^*$ to represent zero or more moves of the PDA

The language of a PDA:

A PDA has final states like a nondeterministic finite automaton and has also the additional structure. namely pushdown store(PDS or stack). So we can define **acceptance of input strings by PDA** in terms of final states or in terms of PDS.

Acceptance by Final state:

Let $P = (Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$ be a PDA. Then Language accepted by P by final state is denoted as L(P) and is defined as

$$L(P) = \{ w \in \Sigma^* \mid (q_0, w, Z_0) \vdash^* (q_f, \varepsilon, \alpha)$$

for some final state q_f and any $\alpha \in \Gamma^*\}$

Acceptance by Empty Stack:

Let $P = (Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$ be a PDA. Then Language accepted by P by empty stack is denoted as N(P) and is defined as

$$N(P) = \{ w \in \Sigma^* \mid (q_0, w, Z_0) \vdash^* (q, \varepsilon, \varepsilon) \text{ for any state } q\}$$

These two methods are equivalent in the sense that a language L has a PDA A that accepts it by final state if and only if L has a PDA B that accepts it by empty stack.

However for a given PDA P, the languages that P accepts by final state and by empty stack are usually different.

*) Construct a pda accepting $L = \{\omega c \omega^T \mid \omega \in \{a,b\}^*\}$ by final state

Sol: The required pda is

$$P = (\{q_0, q_1, q_f\}, \{a,b,c\}, \{a,b,z_0\}, \delta, q_0, z_0, \{q_f\})$$

where δ is given as

$\delta(q_0, a, z_0) = \{(q_0, az_0)\}$
$\delta(q_0, b, z_0) = \{(q_0, bz_0)\}$
} By these rules, pda P pushes the first symbol of input string on stack if it is a (or) b

$\delta(q_0, a, a) = \{(q_0, aa)\}$
$\delta(q_0, b, a) = \{(q_0, ba)\}$
$\delta(q_0, a, b) = \{(q_0, ab)\}$
$\delta(q_0, b, b) = \{(q_0, bb)\}$
} By these rules, pda P pushes the symbols of the input string to the stack until it sees the symbol C

$\delta(q_0, c, a) = \{(q_1, a)\}$
$\delta(q_0, c, b) = \{(q_1, b)\}$
$\delta(q_0, c, z_0) = \{(q_1, z_0)\}$
} on seeing c, the pda moves to state q_1 without making any changes to stack

$\delta(q_1, a, a) = \delta(q_1, b, b) = \{(q_1, \epsilon)\}$ — The pda erases the topmost symbol of stack if it coincides with the current input symbol

$\delta(q_1, \epsilon, z_0) = \{(q_f, z_0)\}$ — The pda reaches final state q_f when the entire string is processed

Consider the string ω = bacab and test for acceptance for the above pda

$(q_0, bacab, z_0) \vdash (q_0, acab, bz_0) \vdash (q_0, cab, abz_0)$

$\vdash (q_0, ab, abz_0) \vdash (q_1, b, bz_0)$

$\vdash (q_1, \epsilon, z_0) \vdash (q_f, \epsilon, z_0)$

$\therefore (q_0, bacab, z_0) \vdash^* (q_f, \epsilon, z_0)$ $\therefore$ bacab $\in L(P)$

ie, it is accepted by pda P.

Test the acceptance of the string abcbb by the above pda

$(q_0, abcbb, z_0) \vdash (q_0, bcbb, az_0)$

$\vdash (q_0, cbb, baz_0)$

$\vdash (q_0, bb, baz_0)$

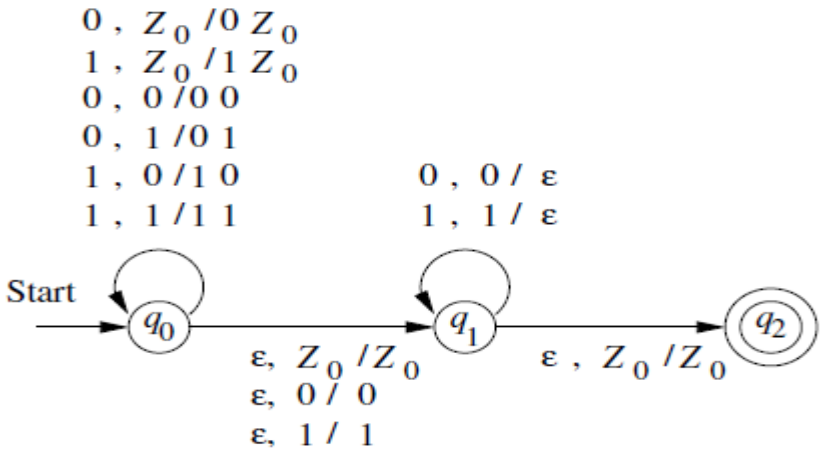
$\vdash (q_1, b, az_0)$

Since $\delta(q_1, b, a) = \phi$, the pda halts.

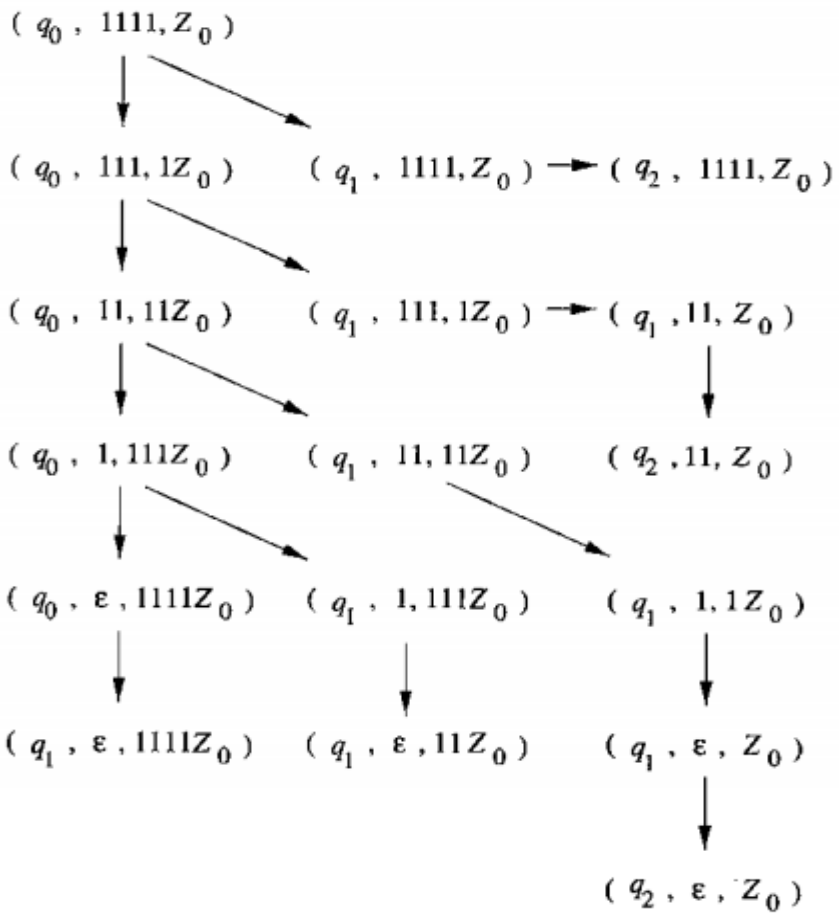
Hence the string abcbb is not accepted

Ex:

Consider the following automata



Test the acceptance of the string 1111 by the above PDA.



Since q2 is a final state, the string 1111 is accepted by the PDA.

From Empty Stack to Final state:

Theorem:

If L=L(P_N) for some PDA

$$P_N = (Q, \Sigma, \Gamma, \delta_N, q_0, Z_0, \phi)$$

Which accepts by empty stack, then there exists another PDA P_F which accepts by final state such that L=L(P_F)

Construction steps:

We will construct a PDA P_F which accepts by final state from a given PDA

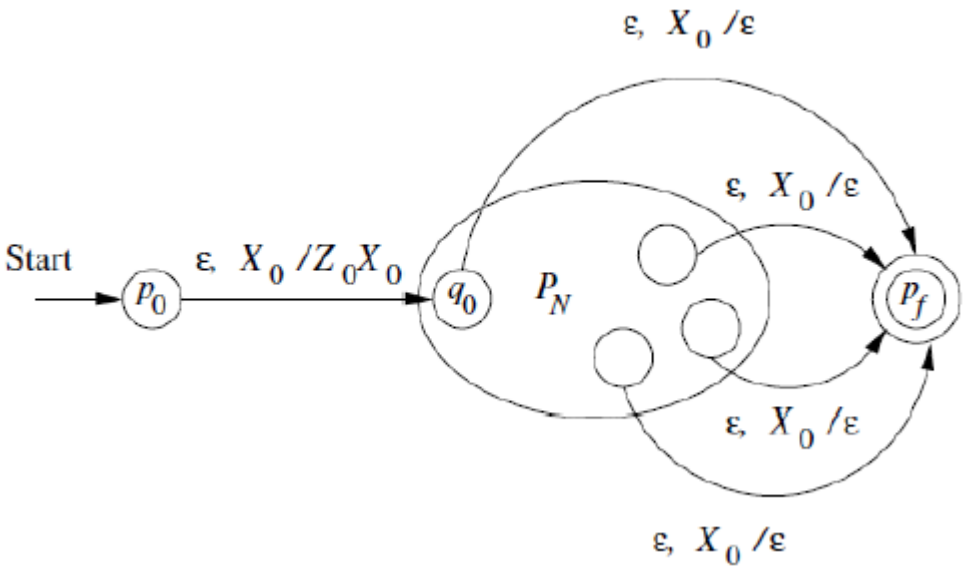
$P_N = (Q, \Sigma, \Gamma, \delta_N, q_0, Z_0, \phi)$ which accepts by empty stack as follows

The PDA P_F is given as

$$P_F = (Q \cup \{p_0, p_f\}, \Sigma, \Gamma \cup \{X_0\}, \delta_F, p_0, X_0, \{p_f\})$$

- The states of P_F will contain two new extra states p_0 and p_f.
  - The state p_0 is the start state of P_F.
  - The state p_f is only final state of P_F.
- P_F will have a new stack symbol X_0, and X_0 will be the initial stack symbol of P_F.
  - The new stack symbol X_0 is to guard the stack bottom against accidental emptying.
- The transition function δ_F will contain everything in the transition function δ of P_N and the following new transitions:
  - $\delta_F(p_0,\varepsilon,X_0)=\{(q_0,Z_0X_0)\}$ i.e. The new start state p_0 pushes the initial stack symbol Z_0 of P_N into the stack and moves to the state q_0 which is the start state of P_N.
  - For any state q of P_N, $\delta_F(q,\varepsilon,X_0)=\{(p_f,\varepsilon)\}$. i.e. Every state q of P_N can move to the new final state p_f when the stack symbol is X_0 and erases the top of stack symbol.

This is illustrated in the following figure.



Problem:

Construct a PDA accepting the set of all strings over{0,1} with equal no.of 0's and 1's by empty stack.

Also convert it to PDA that accepts by final state.

Sol:

The PDA that accepts the given language by empty stack is

$$M = (\{q\}, \{0,1\}, \{0,1,z_0\}, \delta, q, z_0, \Phi)$$

Where δ is given as

1) $\delta(q, 0, Z_0) = \{(q, 0Z_0)\}$.

2) $\delta(q, 1, Z_0) = \{(q, 1Z_0)\}$.

3) $\delta(q, 0, 0) = \{(q, 00)\}$.

4) $\delta(q, 1, 1) = \{(q, 11)\}$.

5) $\delta(q, 0, 1) = \{(q, \varepsilon)\}$.

6) $\delta(q, 1, 0) = \{(q, \varepsilon)\}$.

7) $\delta(q, \varepsilon, Z_0) = \{(q, \varepsilon)\}$.

Now the PDA that accepts the required language by final state is given as

$$P_F = (\{q, p_0, p_f\}, \{0,1\}, \{0,1,z_0,x_0\}, \delta_F, p_0, x_0, \{p_f\})$$

Where δ_F is given as

1) $\delta_F(p_0, \varepsilon, x_0) = \{(q, Z_0 x_0)\}$

2) $\delta_F(q, 0, Z_0) = \{(q, 0Z_0)\}$.

3) $\delta_F(q, 1, Z_0) = \{(q, 1Z_0)\}$.

4) $\delta_F(q, 0, 0) = \{(q, 00)\}$.

5) $\delta_F(q, 1, 1) = \{(q, 11)\}$.

6) $\delta_F(q, 0, 1) = \{(q, \varepsilon)\}$.

7) $\delta_F(q, 1, 0) = \{(q, \varepsilon)\}$.

8) $\delta_F(q, \varepsilon, Z_0) = \{(q, \varepsilon)\}$.

9) $\delta_F(q, \varepsilon, x_0) = \{(p_f, \varepsilon)\}$

From Final state to Empty Stack:

Theorem:

If L=L(P_F) for some PDA

$$P_F = (Q, \Sigma, \Gamma, \delta_F, q_0, Z_0, F)$$

Which accepts by final state, then there exists another PDA P_N which accepts by empty stack such that L=L(P_N).

Construction steps:

We will construct a PDA P_N which accepts by empty stack from a given PDA

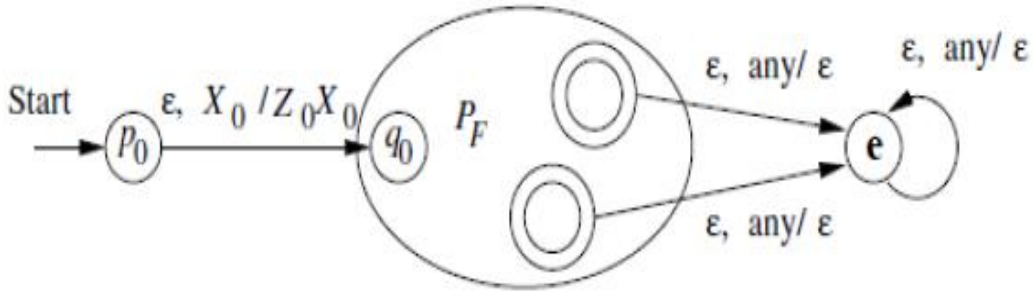
$P_F = (Q, \Sigma, \Gamma, \delta_F, q_0, Z_0, F)$ which accepts by final state as follows

The PDA P_N is given as

$$P_N = (Q \cup \{p_0, e\}, \Sigma, \Gamma \cup \{X_0\}, \delta_N, p_0, X_0)$$

- The states of P_N will contain two new extra states p_0 and **e**.
  - The state p_0 is the start state of P_N.
  - The state **e** is an **erase state** which will empty the stack.
- P_N will have a new stack symbol X_0, and X_0 will be the initial stack symbol of P_N.
  - The new stack symbol X_0 is to guard the stack bottom against accidental emptying.
- The transition function δ_N will contain everything in the transition function δ of P_F and the following new transitions:
  - $\delta_N(p_0, \varepsilon, X_0) = \{(q_0, Z_0 X_0)\}$ i.e. The new start state p_0 pushes the initial stack symbol Z_0 of P_F into the stack and moves to the state q_0 which is the start state of P_F.
  - For any final state **f** of P_F, $\delta_N(f, \varepsilon, Y) = \{(e, \varepsilon)\}$ for any stack symbol **Y**. i.e. A final state of P_F can move to the new erase **e** state by erasing the top of stack symbol.
  - $\delta_N(e, \varepsilon, Y) = \{(e, \varepsilon)\}$ i.e. The new erase state **e** erases all symbols from the stack.

This is illustrated in the following figure.



Pb:

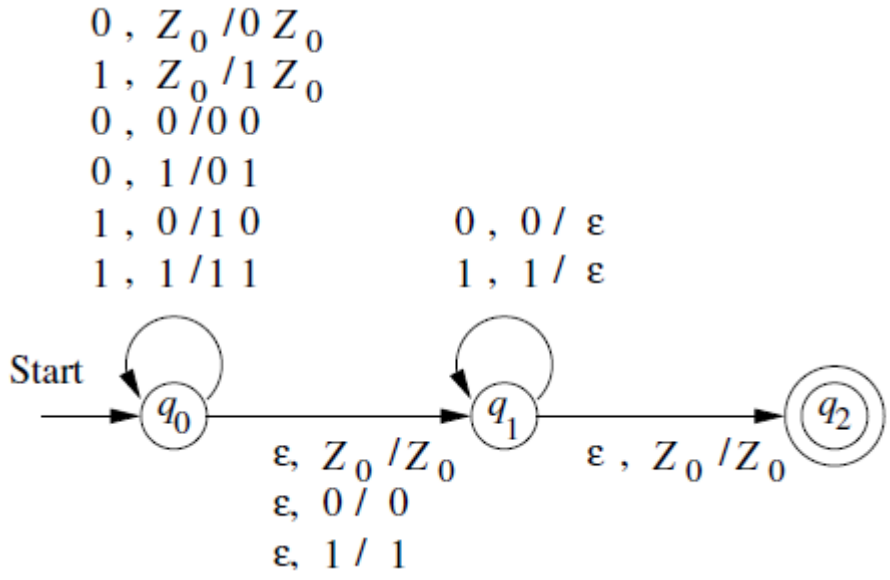
construct a PDA accepting L={ ww^R/ w Є {0,1}*} by final state. Hence construct a PDA that accepts the same language L by empty stack.

Sol:

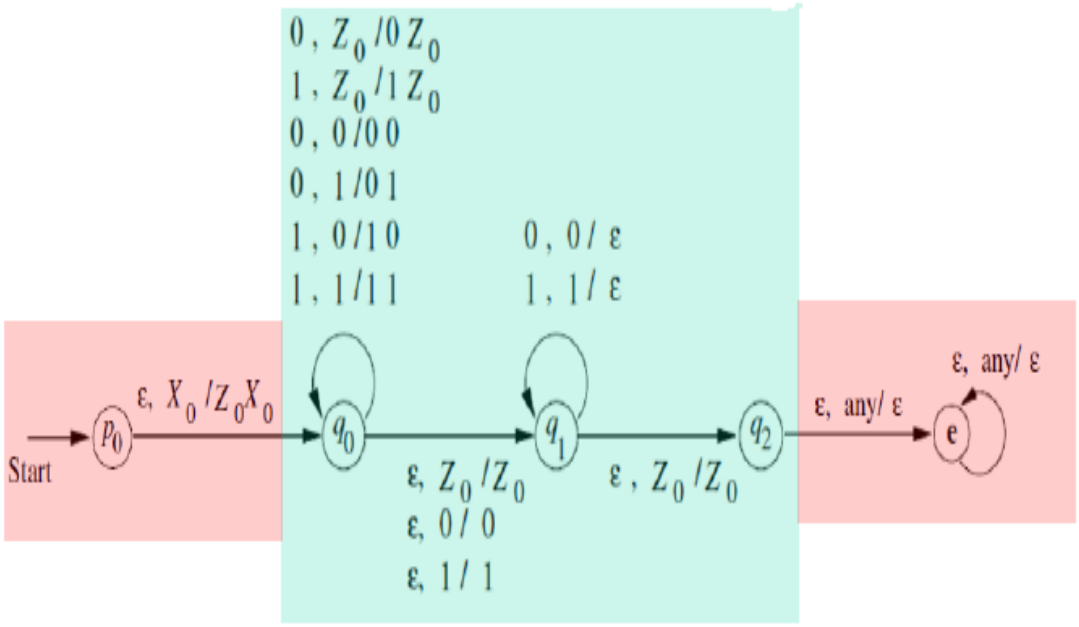
The PDA accepting the language L by final state is given as

$$P = (\{q_0,q_1,q_2\}, \{0,1\}, \{0,1,Z_0\} ,\delta ,q_0 ,Z_0, \{q_2\})$$

Where δ is given as follows.



Now the PDA that accepts the language L by empty stack is given as follows.



Equivalence of PDA's and CFG's:

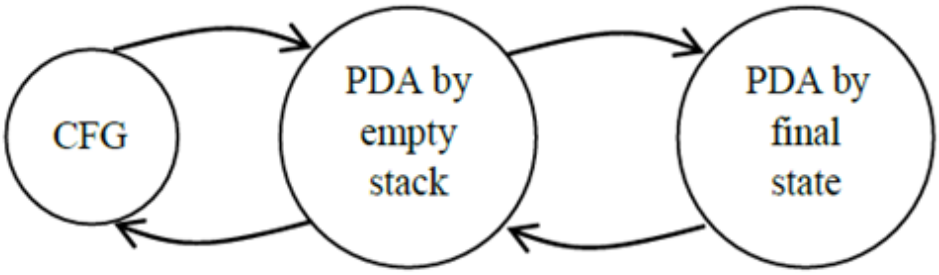
We can prove that the following are all equivalent.

1. The context-free-languages, i.e., the languages defined by CFG's.

2. The languages that are accepted by final state by some PDA.

3. The languages that are accepted by empty stack by some PDA.

We can prove this by



FROM CFG TO PDA:

Let G = (V, T, P, S) be a CFG. We can construct PDA P_N that accepts L (G) by empty stack as follows.

The PDA P_N is given as

$$P_N = (\{q\}, T, V \cup T, \delta, q, S, \phi)$$

Where δ is given as defined as follows.

1. **For each variable A,**

   $\delta(q,\varepsilon,A) = \{ (q, \beta) \mid A \rightarrow \beta$ is a production of G $\}$

2. **For each terminal a, $\delta(q,a,a) = \{(q,\varepsilon)\}$**

Problem:

Find a PDA for the given grammar S →0S1 | 00 | 11

Sol:

In the given grammar S is a variable and the terminals are 0 and 1.

Now the required PDA P_N is given as

$P_N = (\{q\}, T, VUT, \delta, q, S, \Phi)$

$= (\{q\}, \{0, 1\}, \{0, 1, S\}, \delta, q, S, \Phi)$

Where δ is given as

$\delta (q, \in, S) = \{(q, 0S1), (q, 00), (q, 11)\}$

$\delta (q, 0, 0) = \{(q, \in)\}$

$\delta (q, 1, 1) = \{(q, \in)\}$

Problem:

Let G be the grammar given by

$S \rightarrow 0BB$

$B \rightarrow 0S \mid 1S \mid 0$

Construct a PDA. Test if 010^4 is in the language.

Sol:

In the given grammar S, B are variables and the terminals are 0 and 1.

Now the required PDA P_N is given as

$$P_N = (\{q\}, T, VUT, \delta, q, S, \Phi)$$

$$= (\{q\}, \{0, 1\}, \{0, 1, S, B\}, \delta, q, S, \Phi)$$

Where δ is given as

$$\delta(q, \varepsilon, S) = \{(q, 0BB)\}.$$

$$\delta(q, \varepsilon, B) = \{(q, 0S), (q, 1S), (q, 0)\}.$$

$$\delta(q, 0, 0) = \{(q, \varepsilon)\}$$

$$\delta(q, 1, 1) = \{(q, \varepsilon)\}$$

Acceptance of 010^4

$$\begin{aligned}
\delta(q, 010000, S) &\mapsto (q, 010000\ , 0BB) \\
&\mapsto (q, 10000, BB) \\
&\mapsto (q, 10000, 1SB) \\
&\mapsto (q, 0000, SB) \\
&\mapsto (q, 0000, 0BBB) \\
&\mapsto (q, 000, BBB) \\
&\mapsto (q, 000, 0BB) \\
&\mapsto (q, 00, BB) \\
&\mapsto (q, 00, 0B) \\
&\mapsto (q, 0, B) \\
&\mapsto (q, 0, 0) \\
&\mapsto (q, \in)
\end{aligned}$$

After the string is processed , stack was empty. So the string is accepted by the PDA by empty stack.

PDA to CFG:

Let A = (Q, Σ, Γ, δ, q_0, z_0,Φ) is a PDA, then there is a context-free grammar G = (V, Σ, P, S) such that L(G) = N(A)

Construction steps:

We construct the context-free grammar G = (V, Σ, P, S) as follows.

The set of variables V is given as

$$V = \{S\} \cup \{ [pzq] / p,q \in Q , Z \in \Gamma \}$$

i.e., any element of V is either the new symbol S (or) an ordered triplet whose first and third elements are states and the second element is a pushdown symbol.

The production rules of G are as follows.

a) **For all states p in Q**, G has the production

$$S \rightarrow [q_0 z_0 p]$$

b) **if δ(q, a, z) contains (q^1, ε)** , then this move induces a production of the form

$$[qzq^1] \rightarrow a$$

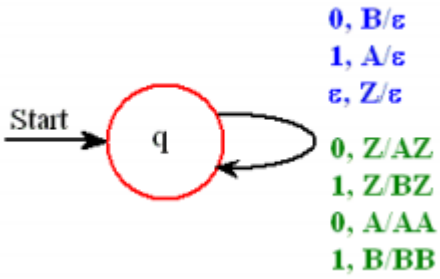
c) **If δ(q, a, X) contains the pair ($r, Y_1Y_2 \ldots Y_k$)** then this induces the productions of the form

$$[qXr_k] \rightarrow a[rY_1r_1][r_1Y_2r_2] \ldots [r_{k-1}Y_kr_k]$$

Where each of the states $r_1,r_2,\ldots\ldots r_k$ can be any state in Q.

Ex:

Construct the equivalent CFG for the following PDA



Sol:

The equivalent CFG G = (V, Σ, P, S) is as follows.

1. The variable set V is given as

$$V = \{S, [qZq], [qAq], [qBq]\}.$$

2. Σ = {0,1}

3. the productions set P is given as follows.

$$S \rightarrow [qZq]$$

$$[qZq] \rightarrow 0[[qAq][qZq] \text{ (since } \delta_N(q, 0, Z) \text{ contains } (q, AZ))$$
$$[qZq] \rightarrow 1[[qBq][qZq] \text{ (since } \delta_N(q, 1, Z) \text{ contains } (q, BZ))$$
$$[qAq] \rightarrow 0[[qAq][qAq] \text{ (since } \delta_N(q, 0, A) \text{ contains } (q, AA))$$
$$[qBq] \rightarrow 1[[qBq][qBq] \text{ (since } \delta_N(q, 1, B) \text{ contains } (q, BB))$$
$$[qAq] \rightarrow 1 \text{ (since } \delta_N(q, 1, A) \text{ contains } (q, \epsilon))$$
$$[qBq] \rightarrow 0 \text{ (since } \delta_N(q, 0, B) \text{ contains } (q, \epsilon))$$
$$[qZq] \rightarrow \epsilon \text{ (since } \delta_N(q, \epsilon, Z) \text{ contains } (q, \epsilon))$$

Renaming the variables as [qzq], [qAq], [qBq] respectively as A,B and C we get the grammar G = (V, Σ, P, S) where V = {S,A,B,C} , Σ ={0,1} and P consists of productions

S →A

A→0BA

A→1CA

B→0BB

C→1CC

C→0

B→1

A→ε

(Or)

S →A

A→0BA | 1CA | ε

B→0BB | 1

C→1CC | 0