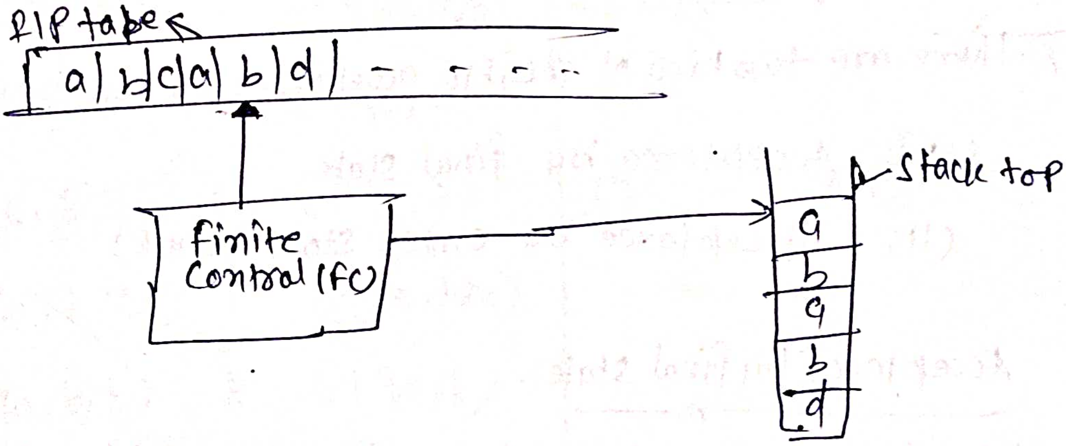
Pushdown Automata :

- The FA that we have studied earlier are not capable to recognise the Context free language (CFL) such as $\{WcW^R / W \in \Sigma^*\}$. for Maching With W^R, W required Some storage

- The PDA Will have 3-things : an I/P tape, a finite Control, and a stack
- The device Will be non-deterministic, having some finite no. of moves in each situation.



Definition of PDA :

A Pushdown Automata is a system. Which is mathematically define as follows.

$$P = (Q, \Sigma, \Gamma, \delta, S, q_f, Z_0)$$

Where Q : non-Empty set of states

Σ : " " " I/P alphabate

Γ : Finite set of Push-down Symbols

δ : Transition function show the mapping between
$(Q \times \Sigma^* \times \Gamma^*) \vdash (Q \times \Gamma^*)$

S : initial state $S \in Q$.

$q_f \subseteq Q$ final state

Z_0 = Stack Symbol

PDA

DPDA (Deterministic PDA)

NDPDA (Non-deterministic PDA)

$\delta \rightarrow Q \times \Sigma^* \times \Gamma^* \rightarrow Q \times \Gamma^*$

$Q \times \Sigma^* \times \Gamma^* \rightarrow 2^{Q \times \Gamma^*}$

Acceptance by PDA:

There are two kind of possible acceptance by PDA for any string

(I) Acceptance by final state

(II) Acceptance by empty store (stack)

(I) Acceptance by final state:

A string accepted by PDA starting with initial state and ending with final state. using stack.

Let PDA, $P(Q, \Sigma, \Gamma, \delta, q_0, q_f, Z_0)$

Let $W \in \Sigma^*$ is a string, which is accepted by PDA 'P' by final state. It will be represented by as follows.

$$(q_0, W, Z_0) \vdash (q_f, \epsilon, Z_0)$$

(ii) Acceptance by empty stack:

A string accepted by PDA starting with initial state and ending with Empty stack.

Let $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, q_f)$ be a PDA, A given string $W \in \Sigma^*$ ~~Let~~ is accepted by empty stack.

$$(q_0, W, Z_0) \vdash (q_0, \epsilon, \epsilon)$$

Q., Design a PDA which accepts the language.

$L = \{ W \in \{a,b\}^* \mid W$ has the equal no. of a's and b's $\}$

Ans:

$P = (Q, \Sigma, \Gamma, \delta, q_0, q_f, Z_0)$

$Q \to \{q_0, q_f)$

$\Sigma \to \{a, b\}$

$\Gamma \to \{a, b\}$

$\delta(q_0, a, Z_0) \vdash \delta(q_0, aZ_0)$

$\delta(q_0, a, a) \vdash \delta(q_0, aa)$

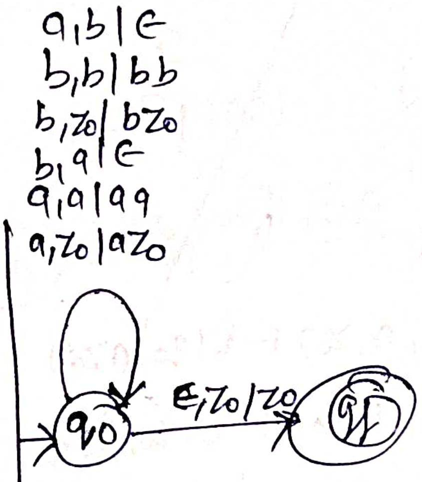
$\delta(q_0, b, a) \vdash \delta(q_0, \epsilon)$

$\delta(q_0, b, Z_0) \vdash \delta(q_0, bZ_0)$

$\delta(q_0, b, b) \vdash \delta(q_0, b)$

$\delta(q_0, a, b) \vdash \delta(q_0, \epsilon)$

$\delta(q_0, \epsilon, Z_0) \vdash \delta(q_f, Z_0)$



Q. Design a PDA for language $L = \{ WcW^R \mid W \in \{a,b\}^* \}$

Ans.

$P \to (Q, \Sigma, \delta, \Gamma, q_0, q_f, Z_0)$

$Q \to \{q_0, q_1, q_f)$

$\Sigma \to \{a, b\}$

$\Gamma \to \{a, b, \bullet\}$

$\delta(q_0, a, Z_0) \vdash (q_0, aZ_0)$

$\delta(q_0, b, Z_0) \vdash (q_0, bZ_0)$

$\delta(q_0, c, Z_0) \vdash (q_f, Z_0)$

$\delta(q_0, a, a) \vdash (q_0, aa)$

$\delta(q_0, a, b) \vdash (q_0, ab)$

$\delta(q_0, b, a) \vdash (q_0, ba)$

$\delta(q_0, b, b) \vdash (q_0, bb)$

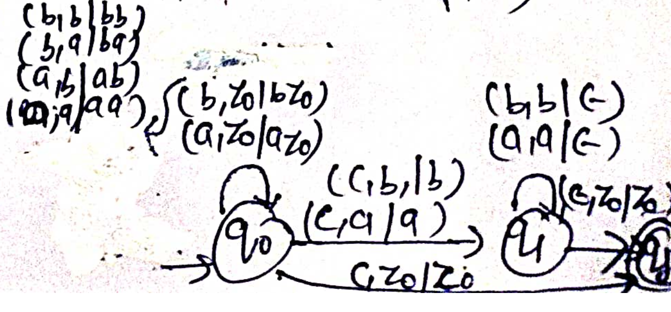
$\delta(q_0, c, a) \vdash (q_1, a)$

$\delta(q_0, c, b) \vdash (q_1, b)$

$\delta(q_1, a, a) \vdash (q_1, \epsilon)$

$\delta(q_1, b, b) \vdash (q_1, \epsilon)$

$\delta(q_1, \epsilon, Z_0) \vdash (q_f, Z_0)$



Q. Design PDA for the language $L = \{a^n b^n : n > 0\}$

Ans.

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

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

$\Sigma = \{a, b\}$

$\Gamma = \{a\}$

$\delta \rightarrow$

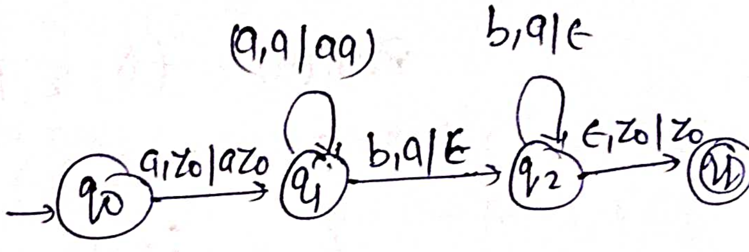
$\delta(q_0, a, z_0) \vdash (q_1, az_0)$

$\delta(q_1, a, a) \vdash (q_1, aa)$

$\delta(q_1, b, a) \vdash (q_2, \epsilon)$

$\delta(q_2, b, a) \vdash (q_2, \epsilon)$

$\delta(q_2, \epsilon, z_0) \vdash (q_f, z_0)$



Q. Design a PDA for the language $L = \{a^n b^{2n} \mid n \geq 1\}$

Ans

$PDA = \{Q, \Sigma, \Gamma, \delta, q_0, q_f, z_0\}$

$Q \rightarrow \{q_0, q_1, q_2, q_f\}$

$\Sigma \rightarrow \{a, b\}$

$\Gamma \rightarrow \{a\}$

$\delta \rightarrow$

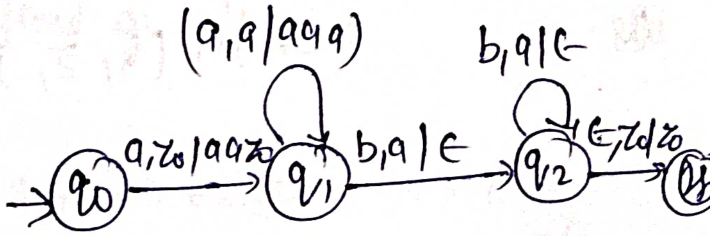
$\delta(q_0, a, z_0) \vdash (q_1, aaz_0)$

$\delta(q_1, a, a) \vdash (q_1, aaa)$

$\delta(q_1, b, a) \vdash (q_2, \epsilon)$

$\delta(q_2, b, a) \vdash (q_2, \epsilon)$

$\delta(q_2, \epsilon, z_0) \vdash (q_f, z_0)$



Q. Construct a PDA for the regular Expression

$r = 0^* 1^+$

Ans. Regular Expression is $r = 0^* 1^+$, Let us write the language for RE.

$$L = \{0^m 1^n \mid m \geq 0, n > 0\}$$

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

$Q = \{q_0, q_f\}$

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

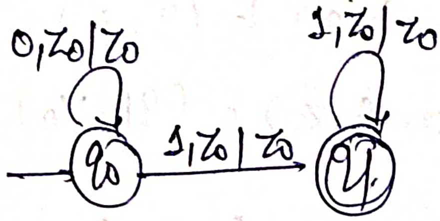
$\Gamma = \{z_0\}$

$\delta \rightarrow$

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

$\delta(q_0, 1, z_0) \vdash (q_f, z_0)$

$\delta(q_f, 1, z_0) \vdash (q_f, z_0)$



Q. Construct the PDA for the language $L = \{a^n b^{n+1} \mid n \geq 1\}$

Ans: $PDA(M)\ (Q, \Sigma, \Gamma, \delta, q_0, q_f, z_0)$

$Q \rightarrow$

$\Sigma \rightarrow$

$\Gamma \rightarrow$

$\delta \rightarrow$

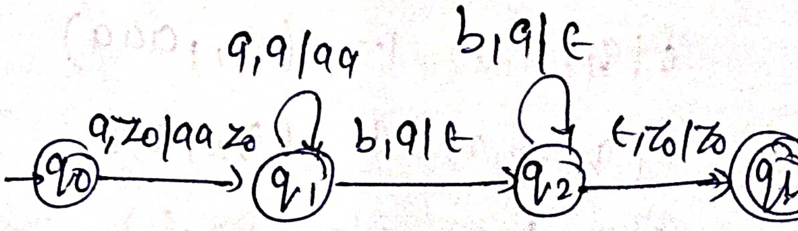
$\delta(q_0, a, z_0) \vdash \delta(q_1, aaz_0)$

$\delta(q_1, a, a) \vdash \delta(q_1, aa)$

$\delta(q_1, b, a) \vdash \delta(q_2, \epsilon)$

$\delta(q_2, b, a) \vdash \delta(q_2, \epsilon)$

$\delta(q_2, \epsilon, z_0) \vdash \delta(q_f, z_0)$



Pushdown Automata & CFG :

It should be clear now that PDA can recognise any language for which there exist a CFG. " That is class of language accepted by Pushdown automata is exactly the class of CFL"

(i) **Construction of PDA equivalent of a CFG :-**

Let $G \cdot (V, T, P, S)$ be a CFG, We must construct a PDA 'P' such that $L(P) = L(G)$. The machine we construct has only two states, P & q, and remains permanently in state q after it's first move. Also, P uses V the set of variables and T the set of terminals, as its stack alphabate, Let, $P = (Q, \Sigma, \Gamma, \delta, S, q_f)$

Where. $Q = \{P, q\}$

$\Sigma = V$

$\Gamma = (V \cup T)$ (set of variables & Terminals)

$S = P$

and Transition relation δ is defined as fallows

(i) $(P, \epsilon, \epsilon) \vdash (q, S)$

(ii) $(q, \epsilon, A) \vdash (q, x)$ for each rule $A \to x$ in CFG

(iii) $(q, a, a) \vdash (q, \epsilon)$ for each a in T

The PDA P start by pushing S, the start symbol of grammar G, on it's initially empty pushdown store & entering state q (transition 1). On each subsequent step, it either replace the top most symbol A on the stack, provided that it is a non-Terminal (Variable) by the RHS x of some rule $A \to x$ in grammar or Pop top most symbol from the stack, provide that it is terminal that match the next symbol.

Q. Design a PDA for the CFG.

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

$V = \{S\}$

$T = \{(,)\}$

$P \rightarrow$

~~soln~~

$S \rightarrow \epsilon$

$S \rightarrow SS$

$S \rightarrow (S)$

Ans!

PDA = $(Q, \Sigma, \Gamma, \delta, S, q)$

$Q = \{q\}$

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

$\Gamma = \{S, (,)\}$

$\delta \rightarrow$

(1). $\delta(q, \epsilon, S) \vdash (q, \epsilon)$

(2) $\delta(q, \epsilon, S) \vdash (q, SS)$

(3) $\delta(q, \epsilon, S) \vdash (q, (S))$

(4) $\delta(q, (, () \vdash (q, \epsilon)$

(5) $\delta(q,),)) \vdash (q, \epsilon)$

Lets apply this Transition on String $W = ()()$

State	Unread I/P	Stack	Transition used.
q	( ) ( )	S	—
q	( ) ( )	SS	2
q	( ) ( )	(S)S	3
q	) ( )	S)S	4
q	) ( )	)S	2
q	( )	S	5
q	( )	(S)	3

P.T.O.

State	Unread I/P	Stack	Transition used
q	c)	S)	4
q	)	)	1
q	—	—	5

Q. Design PDA for the grammar $G(V, T, P, S)$ where.

$V = \{S\}$

$T = \{a, b, c\}$

Production (P) —

$S \rightarrow aSa$

$S \rightarrow bSb$

$S \rightarrow c$

Ans.

PDA (A) $\rightarrow (Q, \Sigma, \delta, \Gamma, S, q)$

$Q = \{q\}$

$\Sigma = \{a, b, c\}$

$\Gamma = \{S, a, b, c\}$

$\delta \rightarrow$

$(q, \epsilon, S) \vdash (q, aSa)$

$(q, \epsilon, S) \vdash (q, bSb)$

$(q, \epsilon, S) \vdash (q, c)$

$(q, a, a) \vdash (q, \epsilon)$

$(q, b, b) \vdash (q, \epsilon)$

$(q, c, c) \vdash (q, \epsilon)$

(ii) Construction of CFG equivalent of a PDA:

Let $P=(Q, \Sigma, \Gamma, \delta, q_0)$ be a PDA that accepts a language L by empty stack. then a CFG, $G=(V, T, P, S)$ that generates L can be constructed using following rules.

- Let's we assume that stack bottom is indicated by a special symbole Z_0.

Let S be the start symbol of grammar.

(i) for every $q \in Q$, add a production $S \to [q_0, Z_0, q_i]$ in P, thus if there are n states in PDA P, then we will add 'n' new production in P,

(ii) for every $q, r \in Q$, $a \in \{\Sigma \cup \{\epsilon\}\}$, $X \in \Gamma$, if $\delta(q, a, X) \vdash (r, \epsilon)$, then add a production

$$[q, X, r] \to a$$

(iii) for every $q, r \in Q$, $a \in \{\Sigma \cup \epsilon\}$, $X \in \Gamma$, if $\delta(q, a, X) \vdash (r, X_1, X_2, \ldots X_k)$ where $X_1, X_2, \ldots X_k \in \Gamma$, then for every choice of $q_1, q_2, \ldots q_k \in Q$, add the production.

$$[q, X, q_k] \mapsto a[r, X, q_1][q_1, X_2, q_2] \ldots [q_{k-1} X_k q_k] \text{ in } P.$$

The basic idea behind this construction is to recognize that the current string in the derivation will consist of two part, the string of I/P symbols read by the PDA so far and a remaining portion corresponding to the stack contents. The variables of the proposed grammar in the form $[r, X, q)$ where r and q are states of the PDA. for variable $[r, X, q]$ to be replaced by a symbol 'a' (a may be ϵ also), it may be the case there is a move in PDA that reads 'a', pop X from the stack, and takes machine from state r to q moves.

Q. Construct a CFG (G) which accept the PDA where

$$A = (\{q_0, q_1\}, \{a, b\}, \{Z_0, Z\}, \delta, q_0, Z_0, \phi)$$

& δ is given by

$$\delta(q_0, b, Z_0) = \{(q_0, ZZ_0)\}$$
$$\delta(q_0, \Lambda, Z_0) = \{(q_0, \Lambda)\}$$
$$\delta(q_0, b, Z) = \{(q_0, ZZ)\}$$
$$\delta(q_0, a, Z) = \{(q_1, Z)\}$$
$$\delta(q_1, b, Z) = \{(q_1, \Lambda)\}$$
$$\delta(q_1, a, Z_0) = \{(q_0, Z_0)\}$$

Ans:

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

Where V consist of S, $[q_0, Z_0, q_0]$, $[q_0, Z_0, q_1]$, $[q_0, Z, q_0]$, $[q_0, Z, q_1]$, $[q_1, Z_0, q_0]$, $[q_1, Z_0, q_1]$, $[q_1, Z, q_0]$, $[q_1, Z, q_1]$

The Production are.

$P_1: S \to [q_0, Z_0, q_0]$

$P_2: S \to [q_0, Z_0, q_1]$

$\delta(q_0, b, Z_0) \vdash (q_0, ZZ_0)$

$P_3 = [q_0, Z_0, q_0] \vdash b[q_0, Z, q_0][q_0, Z_0, q_0]$

$P_4 = [q_0, Z_0, q_0] \vdash b[q_0, Z, q_1][q_1, Z_0, q_0]$

$P_5 = [q_0, Z_0, q_1] \vdash b[q_0, Z, q_0][q_0, Z_0, q_1]$

$P_6 = [q_0, Z_0, q_1] \vdash b[q_0, Z, q_1][q_1, Z_0, q_1]$

$\delta(q_0, \Lambda, Z_0) \vdash \{(q_0, \Lambda)\}$ gives.

$P_7 = [q_0, Z_0, q_0] \vdash \Lambda$

$\delta(q_0, b, Z) \vdash \{(q_0, ZZ)\}$ gives

$P_8 = [q_0, Z, q_0] \vdash b[q_0, Z, q_0][q_0, Z, q_0]$

$P_9 = [q_0, Z, q_0] \vdash b[q_0, Z, q_1][q_1, Z, q_0]$

$P_{10} = [q_0, Z, q_1] \vdash b[q_0, Z, q_0][q_0, Z, q_1]$

$P_{11} = [q_0, Z, q_1] \vdash b[q_0, Z, q_1][q_1, Z, q_1]$

$\delta(q_0, a, Z) \vdash (\{(q_1, Z)\})$ yields

$P_{12} = [q_0, Z, q_0] \vdash a[q_1, Z, q_0]$

$P_{13} : [q_0, Z, q_1] \vdash a[q_1, Z, q_1]$

$\delta(q_1, b, Z) = \{(q_1, \Lambda)\}$ gives

$P_{14} : [q_1, Z, q_1] \vdash b$

$\delta(q_1, a, Z_0) \vdash \{(q_0, Z_0)\}$ gives

$P_{15} : [q_1, Z_0, q_0] \vdash a[q_0, Z_0, q_0]$

$P_{16} : [q_1, Z_0, q_1] \vdash a[q_0, Z_0, q_1]$

Two Stack PDA:-

Two stack PDA required two stacks for solving the problem of single stack PDA.

2 stack PDA (M) Tuple $(Q, \Sigma, \delta, \Gamma, q_0, q_f, \_, Z_1, Z_2)$

Q - Non-Empty set of states

Σ - " " " " I/P alphabate

$\delta \rightarrow$ Transition function $(Q \times \Sigma^* \times \Gamma^* \times \Gamma^*) \rightarrow (Q \times \Gamma^* \times \Gamma^*)$

Γ - Stack alphabate

$q_0 \in Q$ initial state

$q_f \subseteq Q$ final state

Z_1 Stack symbole of 1st stack

Z_2 Stack symbole of 2nd stack

Q. Construct the 2-stack PDA for the language.

$L = \{a^n b^n c^n \mid n \geq 1\}$

Ans:- 2 Stack PDA(M) $\rightarrow (Q, \Sigma, \Gamma, \delta, q_0, q_f, Z_1, Z_2)$

$\delta(q_0, a, Z_1, Z_2) \vdash (q_1, aZ_1, Z_2)$

$\delta(q_1, a, a, Z_2) \vdash (q_1, aa, Z_2)$

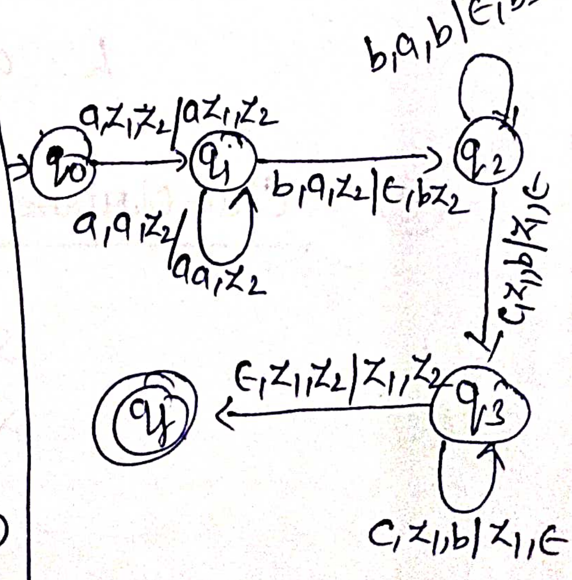
$\delta(q_1, b, a, Z_2) \vdash (q_2, \epsilon, bZ_2)$

$\delta(q_2, b, a, b) \vdash (q_2, \epsilon, bb)$

$\delta(q_2, c, Z_1, b) \vdash (q_3, Z_1, \epsilon)$

$\delta(q_3, c, Z_1, b) \vdash (q_3, Z_1, \epsilon)$

$\delta(q_3, \epsilon, Z_1, Z_2) \vdash (q_f, Z_1, Z_2)$



(Lecture-08)

Closure of CFG :-

13

CFL is closed on Union, Concatination and clean closure:

① $L_1 \cup L_2 = CFG$

$\Downarrow \quad \Downarrow$

CFG CFG

Lets. $L_1 = a^n b^n \mid n \geq 0 \Rightarrow S_1 = aS_1b \mid \epsilon$

$L_2 = c^m d^m \mid m \geq 0 \Rightarrow S_2 = cS_2d \mid \epsilon$

$S \to S_1 \mid S_2$

$S_1 \to aS_1b \mid \epsilon$

$S_2 \to cS_2d \mid \epsilon$

$L = \{a^n b^n\} \cup \{c^m d^m \mid n, m \geq 0$

② Contatination:. $L = L_1 * L_2$

CFG CFG CFG

$S \to S_1 \cdot S_2$

$S_1 \to aS_1b \mid \epsilon$

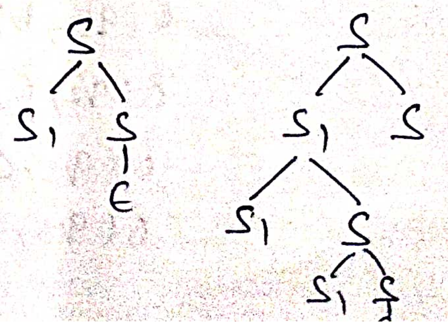
$S_2 \to cS_2d \mid \epsilon$

$L = a^n b^n c^m d^m \mid m, n \geq 0$

③ Clean closure: if L_1 is CFG then L_1^* is CFG

$S \to S_1 S \mid \epsilon$

$S_1 \to aS_1b \mid \epsilon$



∴ CFL is not closed on Complement and Intersection.

Intersection : $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$

let $L_1 = a^n b^n c^m \mid n, m \geq 0$

$L_2 = a^m b^n c^n \mid n, m \geq 0$

$L_1 \cap L_2 = a^n b^n c^n \mid n \geq 0$

Complement : We assume L_1 & L_2 are CFL

$$L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$$

$= \overline{\overline{CFL} \cup \overline{CFL}}$ /* Complement of CFL is CFL

$= \overline{CFL \cup CFL}$ /* Union of CFL is CFL

$= \overline{CFL}$ /* Complement of CFL is CFL

$= CFL$ ✗ /* Not a CFL

Note! In above relation Intersection ~~is not~~ of CFL is not CFL so that due to above relation Complement of CFL is not a CFL.

Pumping lemma for CFL

To prove certain language are not context free language.

* Let L be a CFL
* Let n be a constant
* Any string z in L, $|z| \geq n$
* Split $z = uvwxy$ such that

(i) $|vwx| \leq n$

(ii) $vx \neq \epsilon$ or $|vx| \geq 1$

(iii) for all $i \geq 0$, $uv^iwx^iy \in L$

Q. Show that $L = \{a^n b^n c^n \mid n \geq 1\}$ is not a CFL

Ans:
* Let L be a CFL
* Let n be a constant
* Let $z = a^n b^n c^n$, $|z| \geq n$
* Split $z = uvwxy$

$u = a^n$

$vwx = b^n$, $|vwx| \leq n$

$vx = b^{n-m}$, $|vx| \geq 1$, $m < n$

$y = c^n$

$$uv^iwx^iy = uvv^{i-1}wxx^{i-1}y$$

$$= uvwx(vx)^{i-1}y$$

$$= a^n b^n (b^{n-m})^{i-1} c^n$$

Pick $i = 0$

$uv^iwx^iy = a^n b^m c^n \notin L$ - Hence the given L is not a CFL

UNIT IV: (Important Questions)

1. Construct PDA for following :– L = {an c bzn | n > = 1} = {a,b,c}. Specify the acceptance state.Σover alphabet.
2. Design aPDA fortheLanguage L={WW^R | W={a,b}*}(UPTU 2018-19)
3. GenerateCFGforthe givenPDAMisdefinedas(UPTU 2018-19)

M = ({q_0, q_1}, {0,1} {x, z_0}, δ, q_0, z_0, q_1) where δ is given as follows:$\delta(q_0,1,z_0)=(q_0,xz_0)$

$\delta(q_0,1,x)=(q_0,xx)$

$\delta(q_0,0,x)=(q_0,x)$

$\delta(q_0,\varepsilon,x)=(q_1,\varepsilon)$

$\delta(q_1,\varepsilon,x)=(q_1,\varepsilon)$

$\delta(q_1,0,x)=(q_1,xx)$

$\delta(q_1,0,z_0)=(q_1,\varepsilon)$

4. Prove the Lemma that language recognized by final state PDA machine is also recognized by empty-stack PDA machine and vice-versa. i.e. L(M) = N(M) Language by Final State PDA machine. N(M)→Where L(M) Language by Empty Stack PDA machine.→N(M).(UPTU 2013-14)
5. Prove that the languages L1 and L2 are closed under Intersection and complementation if they are regular, but not closed under the above said two properties if they are context free languages.
6. Construct a PDA that accepts the language L over {0,1} by empty stack which accepts all the strings of 0's and 1's in which number of 0's are twice of the number of 1's.(UPTU 2012-13)
7. Prove that every language accepted by PDA by finite state is also accepted by some PDA by empty stack.
8. Define a deterministic push down automata (DPDA). Write a DPDA which accepts the Language L= {$a^n b^n c^n$ | n and m are arbitrary positive integers}.(UPTU 2015-16)
9. Obtain PDA to accept all strings generated by the language { $a^n b^m a^n$ | m,n>=}.
10. Construct a deterministic PDA for the following language : L = {x ∈ {a, b}* | na (x) ≠ nb (x)} where na (x) : number of a's in the string x nb (x) : number of b's in the string x.(UPTU 2013-14)
11. Show that if L is a language of Deterministic PDA (DPDA) and R is regular then L ∩ R is a language of DPDA.