Module - 3

Context Free Languages

1) **Method to find whether a string belongs to a Grammar or not**

- $S \to 0B \mid 1A$, $A \to 0 \mid 0S \mid 1AA \mid \wedge$, $B \to 1 \mid 1S \mid 0BB$

00110101

$S \to 0B$ ($S \to 0B$)
$\to 00BB$ ($B \to 0BB$)
$\to 001B$ ($B \to 1$)
$\to 0011S$ ($B \to 1S$)
$\to 00110B$ ($S \to 0B$)
$\to 001101S$ ($B \to 1S$)
$\to 0011010B$ ($S \to 0B$)
$\to 00110101$ ($B \to 1$).

- $S \to aAb$ $A \to aAb \mid \wedge$ generates string aabb

$S \to aAb$ ($S \to aAb$)
$\to aaAbb$ ($A \to aAb$)
$\to aabb$ ($A \to \wedge$).

◇ **Derivation Tree**

Left Derivation Tree (It is applied to the production at leftmost variable in each step.)

For generating string aabaa

aabaa

Grammar

~~$S \to aAS$~~ $S \to aAS \mid aSS \mid \epsilon,$

$A \to SbA \mid ba.$

$S \to aSS$ $(S \to aSS)$

$\to aaASS$ $(S \to aAS)$

$\to aabaSS$ $(A \to ba)$

$\to aabaaSSS$ $(S \to aSS)$

$\to aabaaSS$ $(S \to \epsilon)$

$\to aabaaS$ $(S \to \epsilon)$

$\to aabaa$ $(S \to \epsilon)$

Ambiguous Grammar

A Grammar is said to be Ambiguous if there exists two or more derivation tree for a string w (that means two or more left derivation tree).

eg: $G = (\{S\}, \{a+b, +, *\}, P, S)$

$S \to S+S \mid S*S \mid a \mid b$

$S \to S+S$

$\to S+S*S$

$\to a+$ ~~$b*b$~~ $a*S$

$\to a+a*b.$

Simplification of Context free Grammar

Steps :

1) Reduction of CFG.
2) Removal of Unit productions
3) Removal of Null Productions

(Reduction of CFG)

- W_1 $i = 1$
  W_{i+1} derives W_i
  $W_{i+1} = W_i$

Example $P: S \to AC \mid B,\ A \to a,\ C \to c \mid BC,\ E \to aA \mid e$

Ans Phase - 1 $T = \{a, c, e\}$

$W_1 = \{A, C, E\}$.

~~$W_2 = \{S, C, E\}$.~~

$W_2 = \{A, C, E, S\}$.

$W_3 = \{A, C, E, S\}$.

$G' = \{(A, C, E, S), \{a, c, e\}, P, (S)\}$.

$P:$ $S \to AC$
$A \to a$
$C \to c$
$E \to aA / e$.

Phase-2

$V_1 = \{ S \}$
$V_2 = \{ \cancel{AB}\ S, A, C \}$
$V_3 = \{ S, A, C, a, c \}$.

$V_4 = \{ S, A, C, a, c \}$.

$G'' = \{ (A, C, S), \{a, c\}, P, \{S\} \}$.

$$P: \left.\begin{array}{l} S \to AC \\ A \to a \\ C \to c \end{array}\right\} \text{Reduced C.F.G.}$$

(Removal of Unit Productions)

Any Production Rule of form $A \to B$ where $A, B \in$ Non-terminals is called unit production.

Step 1 To remove $A \to B$, add production $A \to x$ to grammar rule ~~wherever~~ whenever $B \to x$ occurs in the grammar. ($x \in$ terminal, x can be NULL).

Step 2 Delete $A \to B$ from the Grammar.

Step 3 Repeat from step 1 until all unit productions are removed.

Example: Remove unit productions from the Grammars whose production rule is given as:

P: $S \to XY$, $X \to a$, $Y \to Z|b$, $Z \to M$, $M \to N$, $N \to a$

$Y \to Z$, $Z \to M$, $M \to N$

(i): Since $N \to a$, we add $M \to a$

P: $S \to XY$, $X \to a$, $Y \to Z|b$, $Z \to M$, $M \to a$, $N \to a$

(ii) $\because$ $M \to a$ we add $Z \to a$

P: $S \to XY$, $X \to a$, $Y \to Z|b$, $Z \to a$, $M \to a$, $N \to a$.

iii) $\because$ $Z \to a$, we add $Y \to a$.

P: $S \to XY$, $X \to a$, $Y \to a|b$, $Z \to a$, $M \to a$, $N \to a$.

Remove the Unreachable Symbols.

P: $S \to XY$, $X \to a$, $Y \to a|b$.

(Removal of Null Productions)

In CFG, a Non-terminal Symbol 'A' is a nullable variable if there is a production $A \to \epsilon$ or there is a derivation that starts at 'A' and leads to ~~ϵ~~ ϵ.

Procedure:

Step-1: To remove $A \rightarrow \epsilon$, look for all productions whose right side contains A.

Step-2: Replace each occurences of 'A' in each of these productions with ϵ.

Step-3: Add the resultant productions to Grammar.

Example (Remove Null productions)

$$S \rightarrow ABAC, \quad A \rightarrow aA \mid \epsilon, \quad B \rightarrow bB \mid \epsilon, \quad C \rightarrow c$$

$$A \rightarrow \epsilon \quad , \quad B \rightarrow \epsilon$$

i) To eliminate $A \rightarrow \epsilon$.

$S \rightarrow ABAC$
$S \rightarrow ABC \mid BAC \mid BC$
$A \rightarrow aA$
$A \rightarrow a$

New Production: $S \rightarrow ABAC \mid BAC \mid BC \mid ABC$
$A \rightarrow aA \mid a, \quad B \rightarrow bB \mid \epsilon, \quad C \rightarrow c.$

ii) To eliminate $B \rightarrow \epsilon$.

$S \rightarrow AAC \mid AC \mid C$, $B \rightarrow b$

New Production: $S \rightarrow ABAC \mid ABC \mid BAC \mid BC \mid AAC \mid AC \mid C$
$A \rightarrow aA \mid a.$
$B \rightarrow bB \mid b.$
$C \rightarrow c.$

(Chomsky Normal Form & CFG to CNF).

In CNF elements in R.H.S should either be two variables or a Terminal.

A C.F.G is in Chomsky Normal form if the productions are:

$$A \to a$$
$$A \to BC$$

A, B, C are Non-terminals & 'a' is terminal.

Steps to Convert CFG to CNF

Step-1 If Start symbol occurs on right side then

$$S' \to S$$

Step 2 Remove Null Productions

Step 3 Remove Unit Productions

Step 4

Convert CFG to CNF

Eg: 1 P: $S \to ASA \mid aB$, $A \to B \mid S$, $B \to b \mid \epsilon$.

Ans

1) Since S appears on Right Hand Side we add a new state S' & $S' \to S$ is added production.

P: $S' \to S$, $S \to ASA \mid aB$, $A \to B \mid S$, $B \to b \mid \epsilon$.

2) Remove Null Productions

$B \to \epsilon$ & $A \to \epsilon$:

After Removing $B \to \epsilon$: P: $S' \to S$, $S \to ASA \mid aB \mid a$, $A \to B \mid S \mid \epsilon$, $B \to b$.

After Removing $A \to \epsilon$ P: $S' \to S$, $S \to ASA \mid aB \mid a \mid SA \mid AS \mid S$, $A \to B \mid S$, $B \to b$.

3) Remove Unit Productions

$S \to S$, $S' \to S$, $A \to B$, $A \to S$

After Removing $S \to S$:
P: $S' \to S$, $S \to ASA \mid aB \mid a \mid SA \mid AS$,
$A \to B \mid S$, $B \to b$.

After Removing $S' \to S$.
P: $S' \to ASA \mid aB \mid a \mid SA \mid AS$
$S \to ASA \mid aB \mid a \mid SA \mid AS$
$A \to B \mid S$
$B \to b$.

After Removing $A \to B$.

P: $S' \to ASA \mid aB \mid a \mid SA \mid AS$
$S \to ASA \mid aB \mid a \mid SA \mid AS$
$A \to b \mid S$
$B \to b$.

After Removing $A \to S$.

P: $S' \to ASA \mid aB \mid a \mid SA \mid AS$
$S \to ASA \mid aB \mid a \mid SA \mid AS$
$A \to b \mid ASA \mid aB \mid a \mid SA \mid AS$
$B \to b$.

4) Find Productions that has more than 2 variables in R.H.S.

$S' \to ASA$, $S \to ASA$ $A \to ASA$

After Removing these, we get:

P: $S' \to AX \mid aB \mid a \mid SA \mid AS$
$S \to AX \mid aB \mid a \mid SA \mid AS$
$A \to b \mid AX \mid aB \mid a \mid SA \mid AS$
$B \to b$
$X \to SA$

5) ~~Find~~ Now change productions $S' \to aB$, $S \to aB$, $A \to aB$.

$$S' \to AX \mid YB \mid a \mid AS \mid SA,$$
$$S \to AX \mid YB \mid a \mid AS \mid SA,$$
$$A \to b \mid AX \mid YB \mid a \mid AS \mid SA,$$
$$B \to b,$$
$$X \to SA,$$
$$Y \to a.$$

(Greibach Normal Form)

A CFG is in Greibach normal form if the productions are in the following forms:

$$A \to b$$
$$A \to b\, C_1 C_2 \ldots C_n$$

where $A, C_1, C_2, \ldots C_n$ are Non-terminals & b is terminal

Convert CFG to GNF

Step 1: Check if CFG have any Unit Productions or Null Productions and remove if there are any

Step 2: Check whether if CFG is already in CNF & convert it into CNF if it is not.

Step 3: Change the names of Non-Terminal symbols into some A_i in ascending order of i.

Example $S \to CA \mid BB$
$B \to b \mid SB$
$C \to b$
$A \to a.$

Already in CNF Form.

Replace S with A_1
C with A_2
A with A_3
B with A_4.

We get. $A_1 \to A_2 A_3 \mid A_4 A_4$
$A_4 \to b \mid A_1 A_4$
$A_2 \to b$
$A_3 \to a.$

Step 4 Alter rules so that non-terminals are in ascending order, such that, If the production is of the form $A_i \to A_j x$, then, $i < j$ and should never be $i \geq j$.

$A_1 \to A_2 A_3 \mid A_4 A_4$

$A_4 \to b \mid \underline{A_1} A_4$

$A_4 \to b \mid \underline{A_2} A_3 A_4 \mid A_4 A_4 A_4$

$A_4 \to b \mid b A_3 A_4 \mid A_4 A_4 A_4$

left Recursion.

Step 5 Remove left Recursion.

Introduce a New Variable to remove the Left Recursion

$A_4 \rightarrow b \mid bA_3 A_4 \mid A_4 A_4 A_4$.

(New Variable) $Z \rightarrow A_4 A_4 Z \mid A_4 A_4$

$$\boxed{A_4 \rightarrow b \mid bA_3A_4 \mid bZ \mid bA_3A_4Z}$$

$\rightarrow$ GNF form

Now the Grammar is.

~~$A_1 \rightarrow A_2 A_3 \mid A_4 A_4$~~
~~$A_4 \rightarrow b \mid b A_3 A_4$~~

$A_1 \rightarrow A_2 A_3 \mid A_4 A_4$
$A_4 \rightarrow b \mid b A_3 A_4 \mid b Z \mid b A_3 A_4 Z$

$Z \rightarrow A_4 A_4 \mid A_4 A_4 Z$

$A_2 \rightarrow b$
$A_3 \rightarrow a$

Now

$A_1 \rightarrow b A_3 \mid b A_4 \mid b A_3 A_4 A_4 \mid b Z A_4 \mid b A_3 A_4 Z A_4$

$A_4 \rightarrow b \mid b A_3 A_4 \mid b Z \mid b A_3 A_4 Z$

$Z \rightarrow b A_4 \mid b A_3 A_4 A_4 \mid b Z A_4 \mid b A_3 A_4 Z A_4 \mid$

$b A_4 Z \mid b A_3 A_4 A_4 Z \mid b Z A_4 Z \mid b A_3 A_4 Z A_4 Z$

$A_2 \rightarrow b$

$A_3 \rightarrow a$

New Grammar with 'Greibach Normal Form'.

3.4 Pumping lemma for Context free languages

Pumping lemma (for CFL) is used to prove that a language is NOT Context free

If A is a context free language then A has a pumping length 'p' such that any string 's', where $|s| \geq p$ may be divided into 5 pieces $s = uvxyz$ such that following conditions must be true:

(1) uv^ixy^iz is in A for every $i \geq 0$.
(2) $|vy| > 0$
(3) $|vxy| \leq p$

To Prove that a language is not Context free ~~language~~ using pumping lemma (for CFL) follow the given steps

→ Assume that A is Context free ~~language~~
→ It has to have pumping length (say p)
→ All strings longer than p can be pumped (say $|s| \geq p$)

→ Now find a string 's' in A such that $|s| \geq p$
→ Divide s into $uvxyz$.
→ Show that $uv^ixy^iz \notin A$ for some i.

→ then consider the ways that s can be divided into uvwxyz.
→ Show that none of these can satisfy all the 3 pumping conditions at the same time.

→ s cannot be pumped == CONTRADICTION

Example 1 Show that $L = \{a^N b^N c^N \mid N \geq 0\}$ is NOT context free

Ans → Assume that L is context free
→ L must have a pumping length (say P)
Now we take a string s such that $S = a^P b^P c^P$
→ We divide s into parts $uvxyz$.

Eg: $P = 4$ So, $S = a^4 b^4 c^4$

Case I (v and y each contain only 1 type of symbol)

$\underbrace{a}_{u}\underbrace{aa}_{v}\underbrace{abbbbc}_{x}\underbrace{c}_{y}\underbrace{cc}_{z}$ $uv^i xy^i z$ ($i = 2$)

uv^2xy^2z

It is a Contradiction to assumption that L is context free

a aaaa abbbbc cccc

$a^6 b^4 c^5 \notin L$

New string does not belong to Language

Case II (Either v or y has more than one kind of symbols.)

$a^4 b^4 c^4$

$\underbrace{aa}_{u}\underbrace{aabb}_{v}\underbrace{b}_{x}\underbrace{b}_{y}\underbrace{cccc}_{z}$ $uv^i xy^i z$ ($i = 2$)

uv^2xy^2z

It is a contradiction to assumption that L is context free language

aaaabbaabb b b b cccc

$a^4 b^2 a^2 b^5 c^4 \notin L$

Pattern is not being followed

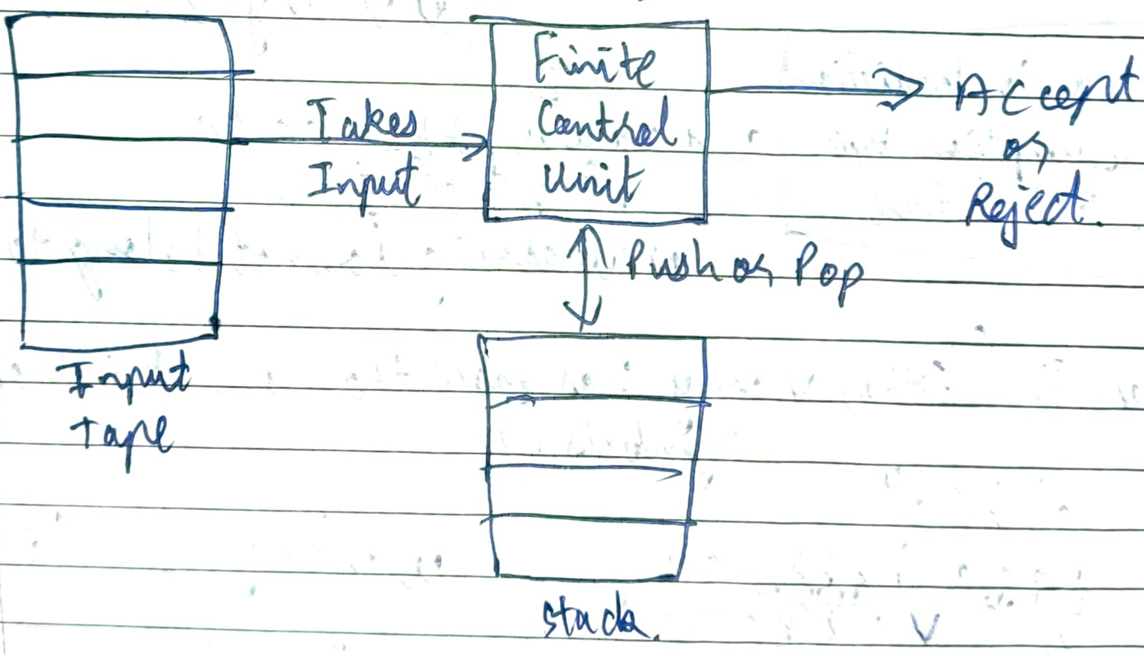
(Pushdown Automata)

A pushdown Automata (PDA) is a way to implement a context free grammar in a similar way

→ It is more powerful than FSM.
→ FSM has limited memory but PDA has more memory
→ PDA = FSM + stack.

A PDA has 3 components.

i) An input tape
ii) A finite Control unit
iii) A Stack with infinite size



Formal definition

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

Date

δ takes an argument a triple $\delta(q, a, X)$ where

(i) 'q' is a state in Q.
(ii) 'a' is either an input symbol in Σ or $a = \epsilon$
(iii) X is a Stack Symbol, that is a member of Γ.

The output of δ is a finite set of pairs (p, γ) where:
'p' is a new state
'γ' is a string of stack symbols that replaces X at top of stack.

Eg: If $\gamma = \epsilon$ then stack is popped.
$\gamma = X$ then stack is unchanged.
$\gamma = YZ$ then X is replaced by Z & Y is pushed onto the stack.

Ex-1 Design PDA to accept
$L = \{0^n 1^n \mid n \geq 1\}$.

Sol^n (i) $L = \{a^n b^n \mid n \geq 1\}$

$L = \{ab, aabb, aaabbb, \ldots\}$

(ii) Logic

Consider $n = 3$
so string is $w = aaabbb$.

- Push "n" no of 'a' into the stack.
- Now for every 'b' Pop out 'a' from the stack.
- At the end of string, the m/c stops as it reaches its final state.

(iii) <u>Implementation</u>

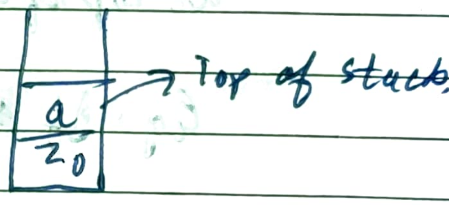
PDA $M = (Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$

$Q = \{q_0, q_1, q_2\}$ $\quad \Gamma = \{a, z_0\}$ $\quad z_0 = z_0$

$\Sigma = \{a, b\}$ $\quad q_0 = q_0$ $\quad F = \{q_2\}$.

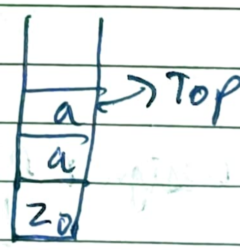
(iv) <u>Transitions Functions</u> $\quad w = aaabbb$

Push a $\delta(\cancel{az}\, q_0, a, z_0) = \delta(q_0, a z_0)$

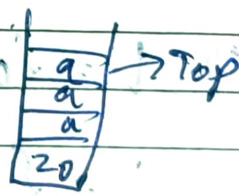


~~δf~~

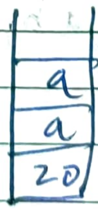
Push a $\delta(q_0, a, a) = \delta(q_0, aa)$



Push a $\delta(q_0, a, a) = \delta(q_0, aa)$.



Push b $\delta(q_0, b, a) = \delta(q_1, \epsilon)$



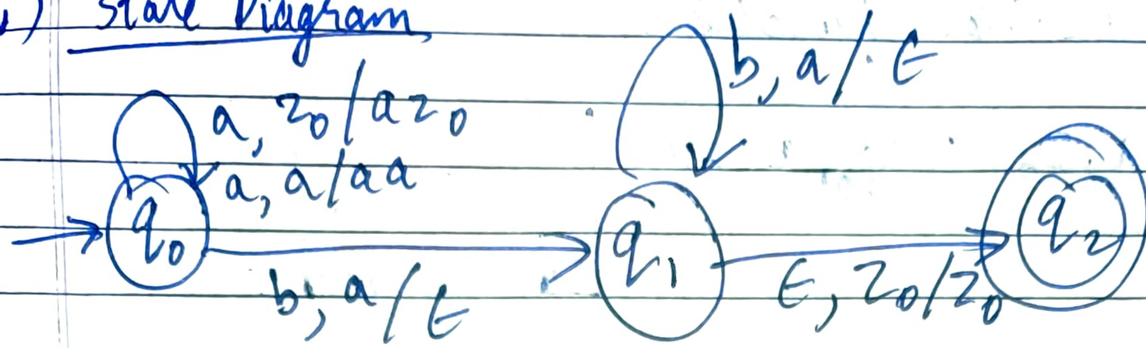
Push b $\delta(q_1, b, a) = \delta(q_1, \epsilon)$

Push b $\delta(q_1, b, a) = \delta(q_1, \epsilon)$

Stack empty.

$\delta(q_1, \epsilon, z_0) = \delta(q_2, \epsilon)$

(v) State Diagram



Exp-2 Design PDA for Language

$$L = \{a^n b^{2n} \mid n \geq 1\}.$$

(i) $L = \{a^n b^{2n} \mid n \geq 1\}$

Strings accepted are

$L = \{abb, aabbbb, aaabbbbbb, \ldots\}$.

ii) Logic (w = aaabbbbbb).

Put $n = 3$ in $a^n b^{2n}$

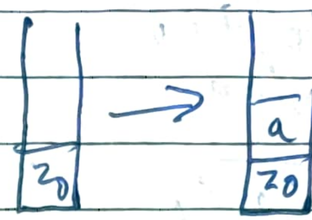
- Push n no of 'a' in stack.
- For every 2 'b's pop out one 'a' out of the stack
- At the end of the string the m/c stops as it reaches the final state.

iii) Formal definition

$w = aaabbbbbb$

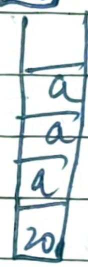
iv) Transition function

Push a $\delta(q_0, a, Z_0) = (q_0, aZ_0)$



Push a $\delta(q_0, a, a) = (q_0, aa)$

Push a $\delta(q_0, a, a) = (q_0, aa)$



Push b $\delta(q_0, b, a) = (q_1, a)$ ⟷ no element pushed or popped.

Push b $\delta(q_1, b, a) = (q_2, \epsilon)$

Push b $\delta(q_2, b, a) = (q_1, a)$

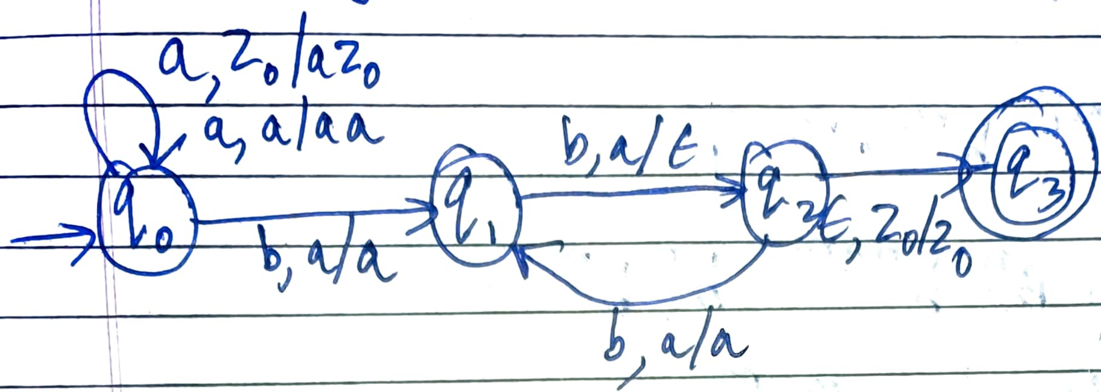
Push b $\delta(q_1, b, a) = (q_2, \epsilon)$

Push b $\delta(q_2, b, a) = (q_1, a)$

Push b $\delta(q_1, b, a) = (q_2, \epsilon)$

Push ϵ $\delta(q_2, \epsilon, Z_0) = (q_3, Z_0)$

State Diagram



Exp-3

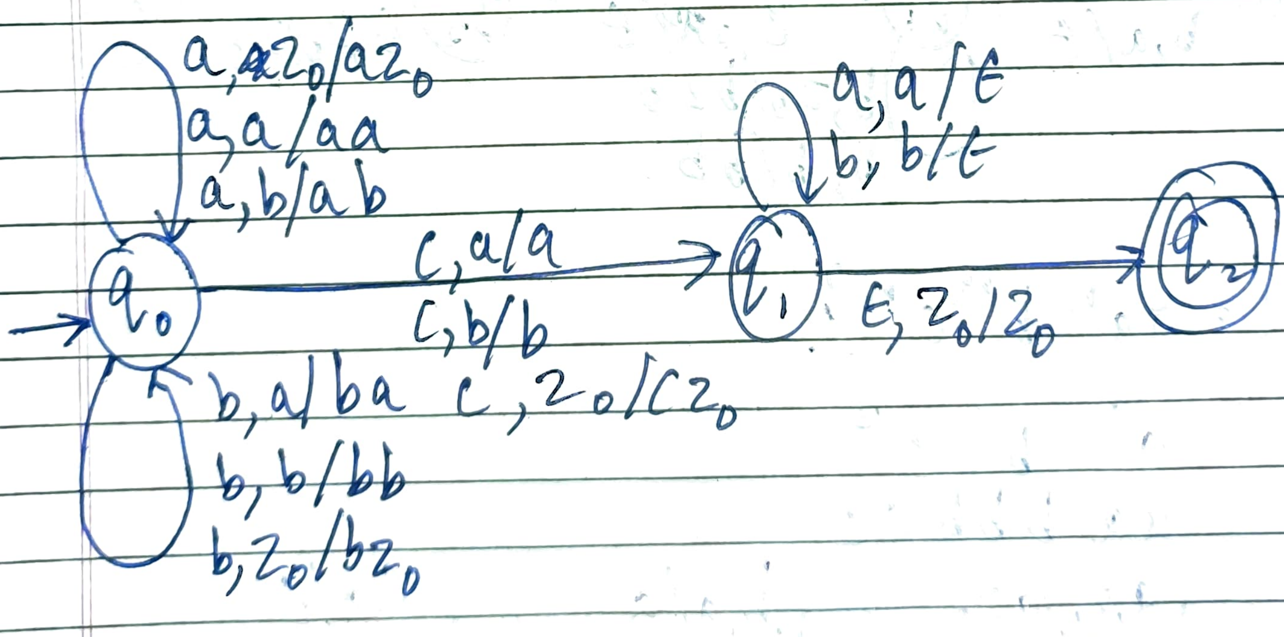
$L = \{ wcw^R \mid w \in \{a,b\}^* \}$.

let

a	b	b	c	b	b	a	ε

↑ string separator

- stack is always popped in reverse.



Exp4 $L = \{a^n b^{m+n} c^m \mid n, m \geq 1\}$.

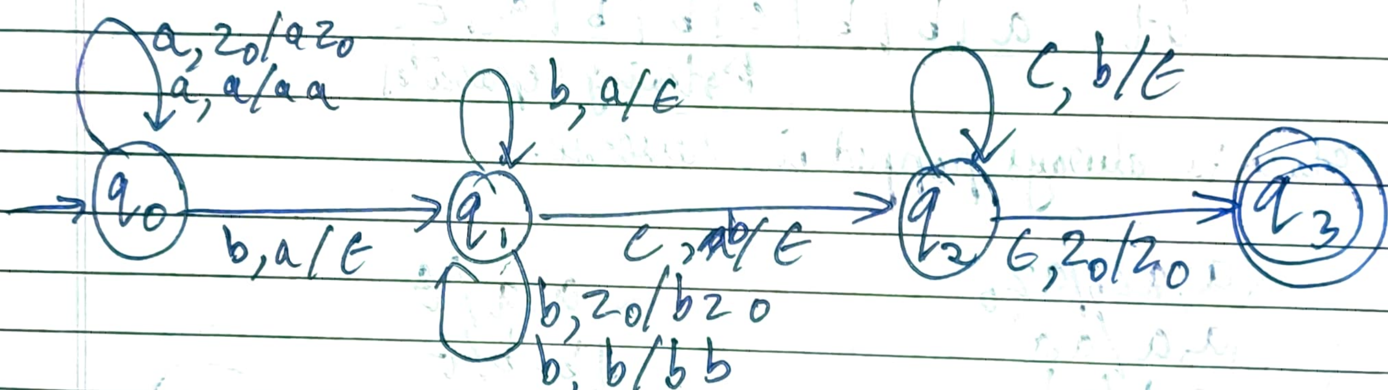
$L = \{a^n b^n b^m c^m\}$

Breaking $b^{n+m} = b^n b^m$

$\underbrace{a^n b^n}_{\text{compare}}$ $\qquad$ $\underbrace{b^m c^m}$

aa bb bb cc

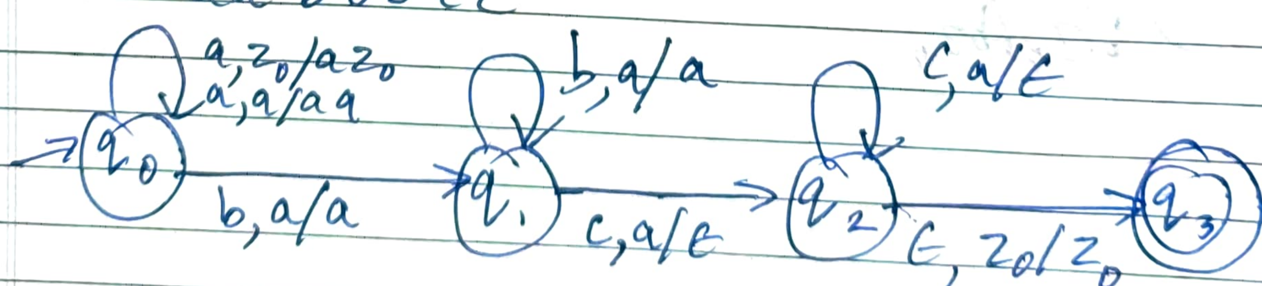
$a^2 b^2 b^2 c^2$



Exp5 $L = \{a^n b^n c^n\}$.

$a^2 b^3 c^2$

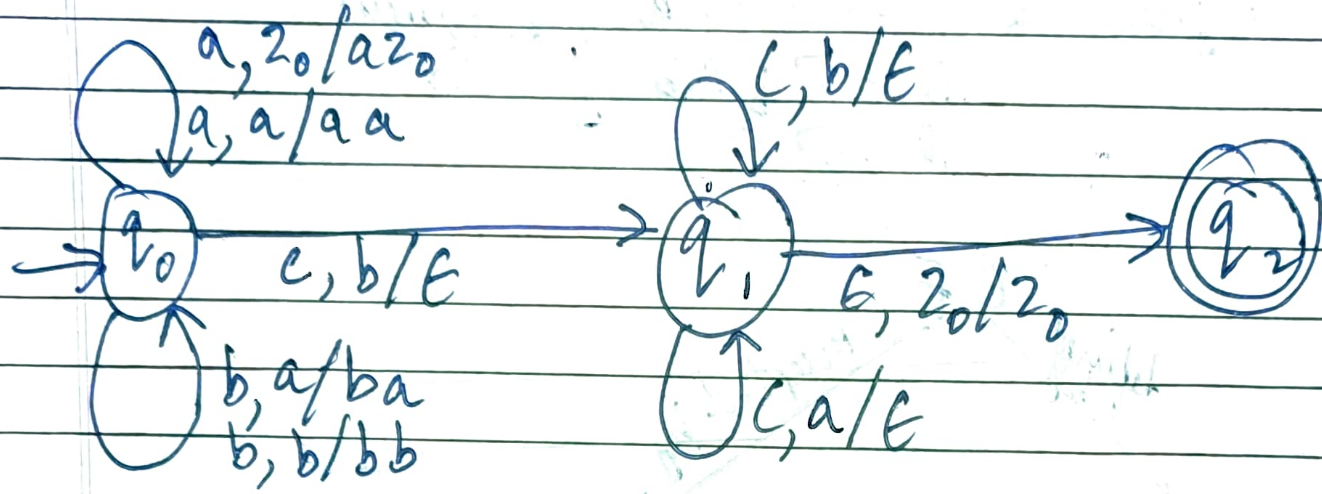
aa bbb cc



Exp 6 $L = \{a^n b^m c^{n+m}\}$.

$n = 2 \qquad m = 3$

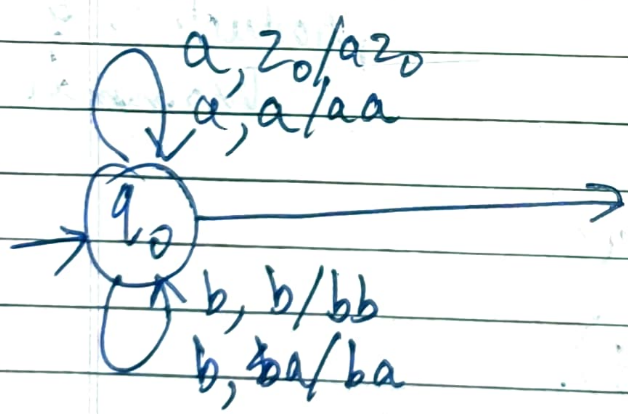
$\underline{aa}\ \underline{bbb}\ \underline{ccccc}$



Exp 7 $L = a^n b^m c^n d^m$

$n = 2 \qquad m = 3$

aa bbb cc ddd



Equivalence with Context free Grammars

* Convert ~~to~~ ~~Eq~~ from CFG to PDA

Ex CFG given is $S \to aSa$, $S \to bSb$ } production CFG, $S \to c$

$$\begin{tikzcd}
 & CF \arrow[dl, "\text{defined by}"'] \arrow[dr, "\text{Accepted by}"] & \\
CFG \arrow[rr, bend left] & & PDA \arrow[ll, bend left]
\end{tikzcd}$$

Rule-1 For each variable A.

$\delta(q, \varepsilon, A) = (q, \beta)$ where $A \to \beta$ is a production of Grammar

Rule-2 For each terminal 'a'

$\delta(q, a, a) = (q, \varepsilon)$.

Ex $S \to 0S1 \mid 00 \mid 11$

Equivalent

$\delta(q, \varepsilon, S) = (q, 0S1), (q, 00), (q, 11)$ — 1

$\delta(q, 0, 0) = (q, \varepsilon)$ — 2

$\delta(q, 1, 1) = (q, \varepsilon)$. — 3

0111, 0001

Date

$\delta(q, 0111, S)$ using ①.

$\delta(q, 0111, 0S1)$ —— ②

$\delta(q, 111, S1)$ —— ①

$\delta(q, 111, 111)$ —— ③

$\delta(q, 11, 11)$ —— ③

$\delta(q, 1, 1)$ —— ③

$\delta(q, \epsilon, \epsilon)$ Accept.

Ex-2

$S \to 0BB$

$B \to 0S \mid 1S \mid 0$

Test 010^4

010000

Equivalent PDA

$\delta(q, \epsilon, S) = (q, 0BB)$ —— ①

$\delta(q, \epsilon, B) = (q, 0S), (q, 1S), (q, 0)$ ②

$\delta(q, 0, 0) = (q, \epsilon)$ —— ③

$\delta(q, 1, 1) = (q, \epsilon)$ —— ④

Testing for string 010000

$\delta(q, 010000, S) \to$ using ①

$\delta(q, \cancel{0}10000, \cancel{0}BB) \to$ ③

$\delta(q, 10000, BB)$ —— ②

$\delta(q, \cancel{1}0000, \cancel{1}SB)$ —— ④

$\delta(q, 0000, SB)$ —— ①

$\delta(q, \cancel{0}000, \cancel{0}BBB)$ —— ③

$\delta(q, 000, BBB)$ —— ②

$\delta(q, \cancel{0}00, \cancel{0}BB)$ —— ③

$\delta(q, 00, BB)$ —— ②

$\delta(q, \cancel{0}0, \cancel{0}B)$ —— ③

$\delta(q, 0, B)$ —— ②

$\delta(q, \cancel{0}, \cancel{0})$ —— ③

$\delta(q, \epsilon, \epsilon)$ Accepted.

* Convert PDA to CFG

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

Construct CFG, $G = (V, \Sigma, R, S)$

Variables, V

1) Special symbol S
2) $[p\, X\, q]$ where p & q are states in Q & X is in ~~ΣΓ~~ Γ.

Productions, R

1) For all states p,

$S \rightarrow [q_0 Z_0 p]$

2) Let $\delta(q, a, X) = (r, y_1 y_2 \dots y_k)$

$[q\, X\, r_k] \rightarrow a\, [r\, y_1\, r_1]\, [r_1\, y_2\, r_2] \dots\dots [r_{k-1}\, y_k\, r_k]$

For all states $r_1, r_2, \dots, r_k$

Ex5 Convert the following PDA into CFG.

$P = (\{q, p\}, \{0, 1\}, \{x, Z\}, \delta, q, Z)$

$\delta(q, 1, Z) = (q, xZ)$	$\delta(q, 0, x) = (p, x)$
$\delta(q, 1, x) = (q, xx)$	$\delta(q, 1, x) = (p, \epsilon)$
$\delta(q, \epsilon, x) = (q, \epsilon)$	$\delta(p, 0, Z) = (q, Z)$

$V = \{S, \underset{A}{[qxq]}, \underset{B}{[qxp]}, \underset{C}{[px\cancel{p}q]}, \underset{D}{[px\cancel{q}p]},$
$\underset{E}{[qZq]}, \underset{F}{[q\cancel{o}Zp]}, \cancel{[qZ}\, \underset{G}{[pZq]}, \underset{H}{[pZp]}$

- $S \rightarrow [qZp]$ $\quad S \rightarrow F$
  $S \rightarrow [qZq]$ $\quad S \rightarrow E$

i) $\delta(q, 1, Z) = (q, xZ)$ $(2^2 = 4)$

	CFG
$\delta[q, Z, q) = 1[qxq][qZq]$	$E \rightarrow 1AE$
$\delta(q, Z, \cancel{q}) = 1[qxp][pZq]$	$E \rightarrow 1BG$
$\delta(q, Z, p) = 1[qxq][qZp]$	$F \rightarrow 1AF$
$\delta(q, Z, p) = 1[qxp][pZp]$	$F \rightarrow 1BH$

CFG

ii) $\delta(q, 1, x) = \cancel{\delta}(q, xx)$ $2^2 = 4$

$\cancel{\delta[q\,x\,q]} =$

$[q x q] \to 1 [q x q][q x q]$ $A \to 1AA$

$[q x q] \to 1 [q x p][p x q]$ $A \to 1BC$

$[q x p] \to 1 [q x q][q x p]$ $B \to 1AB$

$[q x p] \to 1 [q x p][p x p]$ $B \to 1BD$

---

iii) $\delta(q, \epsilon, x) = (q, \epsilon)$

$[q x q] \to \epsilon$ $A \to \epsilon$

---

iv) $\delta(q, 0, x) = (p, x)$

$[q x q] \to 0 [p x q]$ $A \to 0C$

$[q x p] \to 0 [p x p]$ $B \to 0D$

---

v) $\delta(p, 1, x) = (p, \epsilon)$

$[p x p] \to 1$ $D \to 1$

---

vi) $\delta(p, 0, z) = (q, z)$

$[p z q] \to 0 [q z q]$ $G \to 0E$

$[p z p] \to 0 [q z p]$ $H \to 0F$

Context Free grammar

$S \rightarrow E \mid F$

$A \rightarrow 1AA \mid 1BC \mid 0C \mid E$

$B \rightarrow 1AB \mid 1BD \mid 0D$

$D \rightarrow 1$

$E \rightarrow 1AE \mid 1BG$

$F \rightarrow 1AF \mid 1BH$

$G \rightarrow 0E$

$H \rightarrow 0F$