Context free grammar :-

Ex:- String has even no. of a's. $\Sigma = \{a\}$

$[NT \to TNT/T]$

$[NT \xrightarrow{or} NT\ T/T]$

$V = \{q_0, q_1\}$ $T = \{a\}$

P: $q_0 \to aq_1/\epsilon$

$q_1 \to aq_0/a$

General construction of grammar

$S \to \underset{A}{aaS}/\underset{B}{Saa}/\epsilon/\underset{B}{aSa}$

$S \to aA/Ba/\epsilon/aB$

$A \to aS$

$B \to Sa$

Ex:- $L = \{a^n b^n / n \geq 0\}$

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

$w = aabb.$

$\underset{x}{aa}\underset{y}{b}\underset{z}{b}$ $y \neq \epsilon$.

$|xy| \leq n$

$xy^2z = aababb \notin L$

$\underset{y}{aa}\underset{z}{bb}$ $y \neq \epsilon$

$|xy| \leq n$

$xy^2z = aaaabb \notin L$

L is not regular.

→ Context free grammar:-

A grammar $G = (V, T, P, S)$ is said to be context free if all productions in P have the form $A \to \alpha$ where $A \in V$ and $\alpha \in (V \cup T)^*$

The language generated by context free grammar is called context free language or context free generator.

Ex:- Palindrome:

$S \to aSa \mid bSb \mid \in \mid a \mid b$ → remove even no. of digits

↓ remove for odd no of digits

→ it has no rules

Ex:- RE : $0+1^+$

$S \to 0 \mid 1A$

$A \to 1A \mid \in$

Ex:- RE = $(0+1)^*$

$S \to \in \mid 0S \mid 1S$

Ex:- $\underbrace{(011+1)}_{C,D}{}^+ \underbrace{(01)^*}_{E}$ (A, B)

$S \to AB$

$A \to CA \mid DA \mid C \mid D$

$B \to EB \mid \in$

$C \to 011$

$D \to 1$

$E \to 01$

Ex:- RE = $\underbrace{(0^* + 1^+)}_{C,D,A}\ \underbrace{01}_{B}$

$S \to AB$

$A \to C \mid D$

$C \to 0C \mid \in$

$D \to 1D \mid 1$

Ex:- $\underbrace{(011+1)^*}_{C,D,A}\ \underbrace{(01)^*}_{E,B}$

$S \to AB$

$A \to CA \mid DA \mid \in$

$B \to EB \mid \in$

$C \to 011 \to \in D$

$E \to 01$

$D \to 1$

Ex:- RE = $\underbrace{0^*}_{A}\ \underbrace{1}_{B}\ \underbrace{(0+1)^+}_{C}$

$S \to ABC$

$A \to 0A \mid \in$

$B \to 1$

$C \to 0C \mid 1C \mid 01$

Ex:- RE = $\underbrace{(0101)}_{A}\ \underbrace{(0+1)^*}_{B}\ \underbrace{(\underbrace{011}_{D}{}^+)}_{C,E}$

$S \to ABC$

$A \to 0101$

$B \to 0C \mid 1C \mid \in$

$C \to DE$

$D \to 01$

$E \to 1E \mid \in$

Ex:- strings having atleast one 'a' $\Sigma = \{a, b\}$

$$RE = \underbrace{(a+b)^*}_{A}\underbrace{a}_{B}\underbrace{(a+b)^*}_{A}$$

$S \rightarrow AaA$

$A \rightarrow aA / bA / \epsilon$

Ex:- Substring should be 110

$$RE = \underbrace{(0+1)^*}_{A}\underbrace{110}_{B}\underbrace{(0+1)^*}_{A}$$

$S \rightarrow ABA$

$A \rightarrow 0A / 1A / \epsilon$

$B \rightarrow 110$

Ex:- $L = \{ a^n b^m / n \neq m ; n > m \text{ or } m > n \}$

$L = \{ a, b, aab, abb, aaab, aabb, abbb, aabbb, \ldots \}$

$w = \underbrace{aa}_{x}\underbrace{a}_{y}\underbrace{bb}_{z}$. $\quad xy^1z = aaaabb$. ✓

$\underbrace{aa}_{x}\underbrace{ab}_{y}\underbrace{b}_{z}$ $\quad xy^1z = aaabab$ ✗

$\underbrace{aa}_{x}\underbrace{abb}_{y}\underbrace{b}_{z}$ $\quad xy^1z = aaabbb$ ✗

L is not a regular language.

$n > m \quad m > n$

$S \rightarrow S_1 / S_2$

$S_1 \rightarrow$

Ex:- Construct CFG for $11(0+1)^*0$.

$S \rightarrow ABC$

$A \rightarrow 11$

$B \rightarrow 0B / 1B / \epsilon$

$C \rightarrow 0$

* Derivation Trees :- It is a graphical representation for the derivation of given production rule for a given CFG. It is a simple tree to show how the derivation can be done to obtain string from a given set of production rules. It is also called parse tree.

→ Properties of derivation Tree :-

1. The root node is always a node indicating the start non-terminal symbol.
2. The derivation is read from left to right.
3. The leaf nodes are always terminal nodes.
4. The interior nodes are non-terminal nodes

→ Generally we have 2 types of Derivation Trees

① Leftmost :-

It is a derivation in which the leftmost non terminal is replaced first.

② Rightmost :-

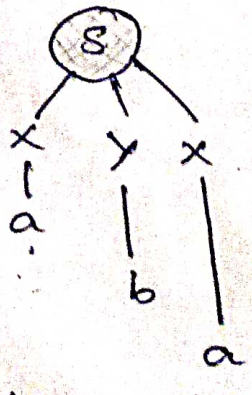
It is a derivation in which rightmost non terminal is replaced first.

Ex :- $S \rightarrow XYX$

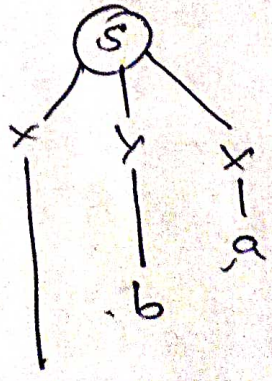
$X \rightarrow a$

$Y \rightarrow b$.

aba



leftmost (LMD)



rightmost (RMD)

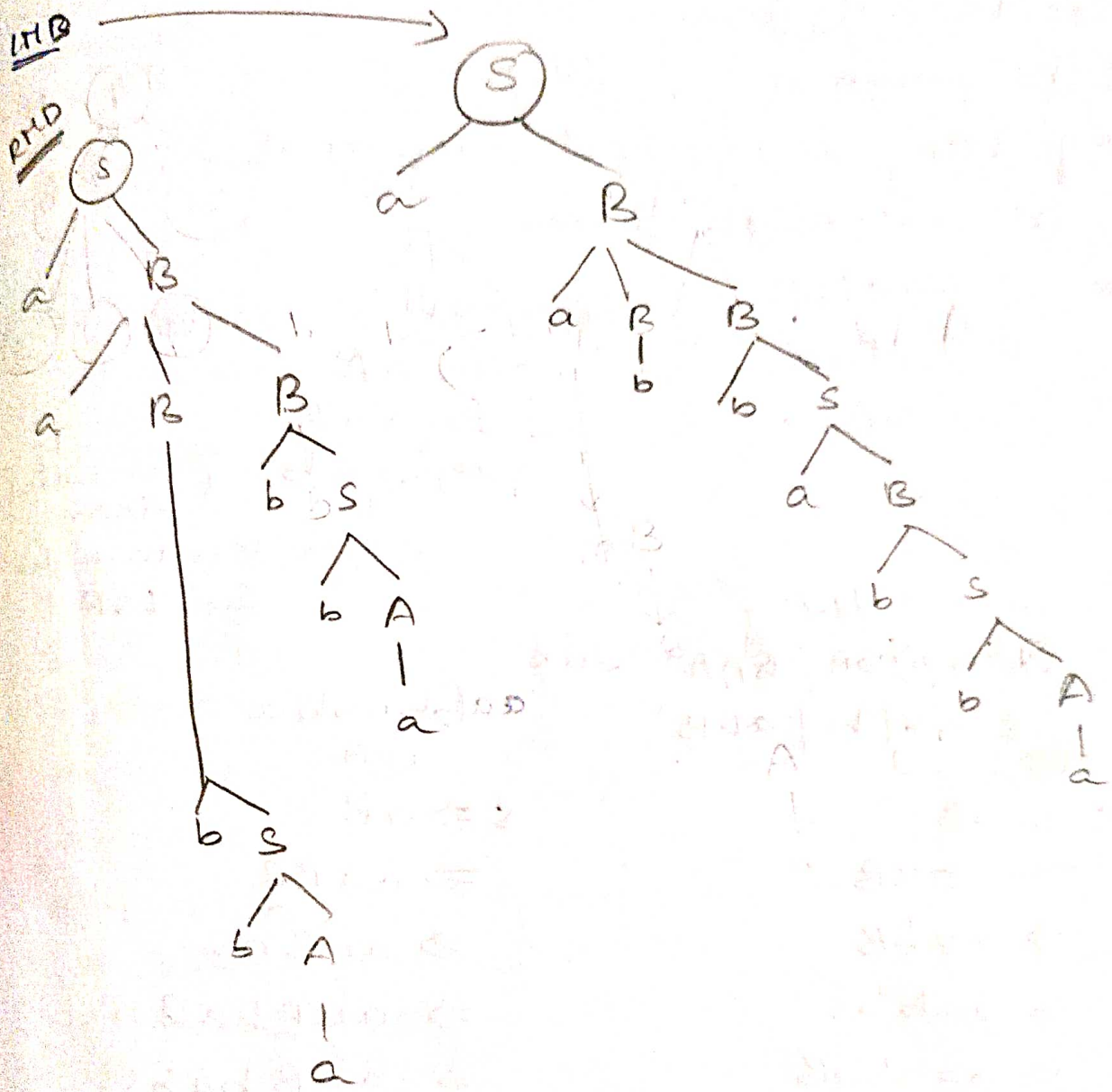
Eg:-

$S \rightarrow aB \mid bA$

$A \rightarrow a \mid as \mid bAA$

$B \rightarrow b \mid bs \mid aBB$

aabbabba - i/p.



Ex:-

$S \rightarrow aA \mid bB$

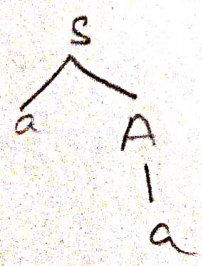
$A \rightarrow aB \mid a$

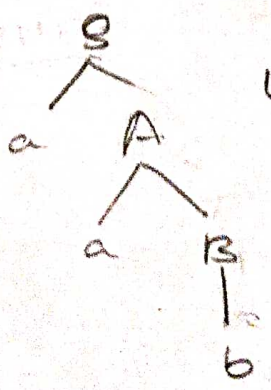
$B \rightarrow bA \mid b$.

i/p - aa

i/p - abab

LMD = RMD





LMD ≠ RMD

→ For single non-terminal production rules, more than one derivation tree can't be constructed.

* Sentential form :-

If the grammar $G = (V, T, P, S)$ is a CFG, then any string $\alpha \in (V \cup T)^*$ such that $S \overset{*}{\Rightarrow} \alpha$ is a sentential form.

Ex:- $S \to aA \mid bB$
$A \to aB \mid a$
$b \to bA \mid b$

i/p = aaab

$S \Rightarrow aA$
$\Rightarrow aaB$
$\Rightarrow aaab$. [sentential form]

from left to right
So LSF

Ex:- $S \to aB \mid bA$
$A \to a \mid aa \mid bAA$
$B \to b \mid bS \mid aBB$

aabbabba

LMD

$S \Rightarrow aB$
$\Rightarrow aaBB$
$\Rightarrow aabB$
$\Rightarrow aabbS$
$\Rightarrow aabbaB$
$\Rightarrow aabbabS$
$\Rightarrow aabbabbA$
$\Rightarrow aabbabba$

RMD

$S \Rightarrow aB$
$\Rightarrow aaBB$
$\Rightarrow aaBbS$
$\Rightarrow aaBbaB$
$\Rightarrow aaBbabS$
$\Rightarrow aaBbabbA$
$\Rightarrow aaBbabba$
$\Rightarrow aabbabba$

Ex: $S \to T00T$

$T \to 0T \mid 1T \mid \epsilon$

i/p - 100111

LMD

$S \Rightarrow T00T$

$\Rightarrow 1T00T$

$\Rightarrow 1\epsilon00T$

$\Rightarrow 100\underline{1}T$

$\Rightarrow 10011\underline{T}$

$\Rightarrow 100111\underline{T}$

$\Rightarrow 100111\epsilon$

$\Rightarrow 100111$

RMD

$S \Rightarrow T00\underline{T}$

$\Rightarrow T00\,0T$

$\Rightarrow T000\underline{1}T$

$\Rightarrow T0001\underline{1}T$

$\Rightarrow T00011\underline{1}T$

$\Rightarrow T000111\epsilon$

$\Rightarrow 1T000111$

$\Rightarrow 1\epsilon000111$

$\Rightarrow 1000111$

LMD

$S \Rightarrow T00T$

$\Rightarrow 1T00T$

$\Rightarrow 10T00T$

$\Rightarrow 10\epsilon00T$

$\Rightarrow 10001T$

$\Rightarrow 100011T$

$\Rightarrow 1000111T$

$\Rightarrow 1000111\epsilon$

$\Rightarrow 1000111$

Ex: $S \to AA$.

$A \to AAA \mid a \mid bA \mid AL$.

i/p - aba.

LMD

$S \Rightarrow AA$

$\Rightarrow aA$

$\Rightarrow abA$.

$\Rightarrow aba$

RMD

$S \Rightarrow AA$

$\Rightarrow AbA$

$\Rightarrow Aba$

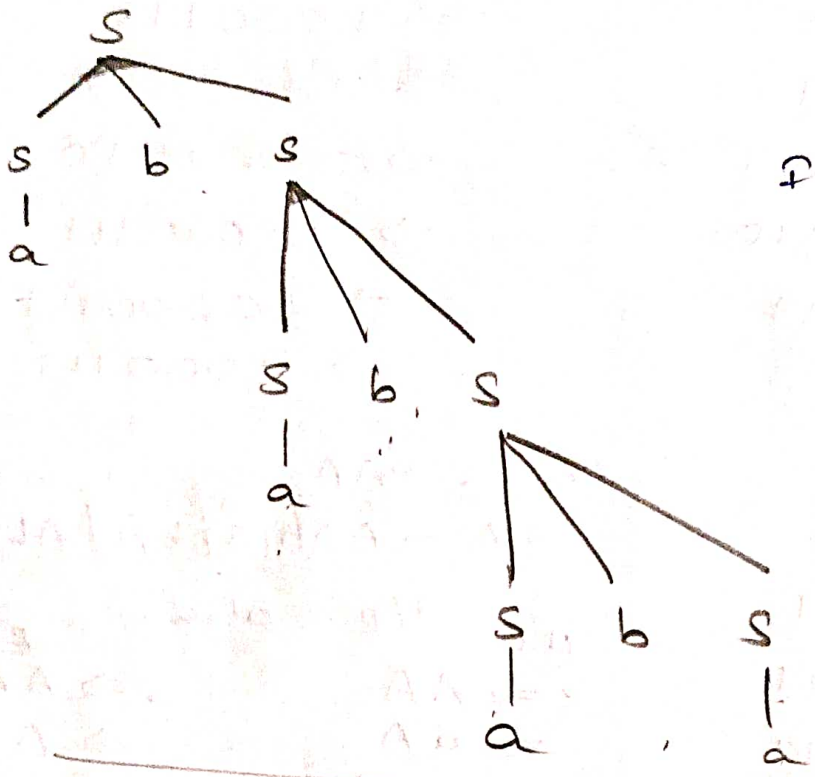
$\Rightarrow aba$

→ if we can construct 2 or more Left most (or) 2 or more right most trees for a string then we can say that the string is ambiguous

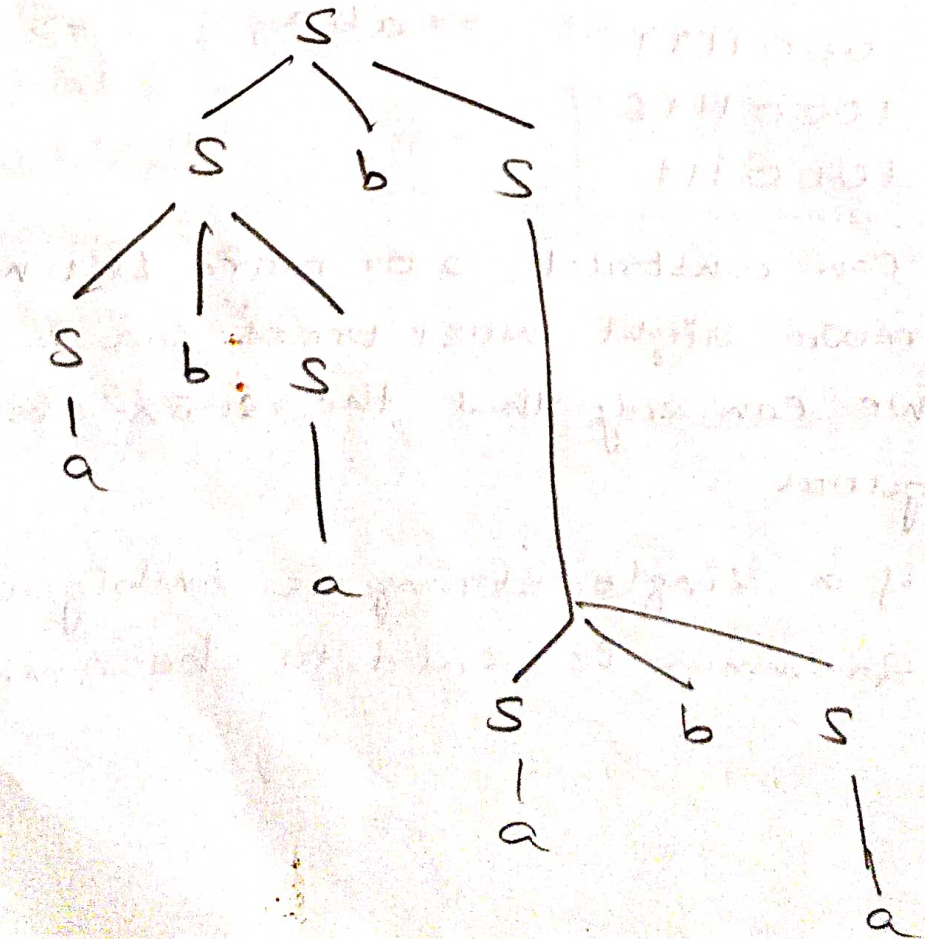
→ Even if a single string is ambiguous, the Grammar is said to be ambiguous.

→ A terminal string 'w' ∈ L(G) is ambiguous if there exists two or more derivation trees for 'w'. (or their exist two or more left most derivations for w).

Ex:- S → SbS/a i/p - abababa



It is ambiguous



Simplification of CFG

We say a non-terminal is useless when
i. there is no direct path from the initial Symbol.
ii. There is no specified production rule for a particular non-terminal.

Ex:- $S \to AB$
$A \to a$
$B \to bS$
$C \to b$ — useless
as there is no direct path

Ex:- $S \to AB$
$A \to a/C$ useless
$B \to bS$
as it has no further productions

→ steps for simplification of CFG:-

1. Construction of reduced grammar (remove useless symbols)
2. Elimination of null productions
3. Elimination of unit productions.

Ex:- $S \to AB$
$A \to a$
$B \to b$
$B \to C$
$C \to C$

$V = \{S, A, B, C\}$
$T = \{a, b, c\}$
P
$S = S$.

$S \to aB$
$B \to b/C$
$C \to C$ useless

$S \to aB$
$B \to b/c$

$V = \{S, B\}$
$T = \{a, b, c\}$
$S \to S$

Ex:-
$S \to AB/CA$
$B \to BC/AB$
$A \to a$
$C \to aB/b$.

$S \to aB/Ca$
$B \to BC/AB$
$C \to aB/b$.

B → useless symbol as it doesn't produce any terminal symbol.

Reduced grammar –

$S \to CA$

$C \to b$

$A \to a$

ex: $S \to aB \mid bA \mid C$, $A \to AC \mid b$, $C \to Bb \mid A$

$V = \{S, A, B, C\}$, $T = \{a, b\}$

$\Rightarrow$ $S \to bA \mid C$, $A \to b \mid AC$, $C \to A$ { B is useless

$V = \{S, A, C\}$, $T = \{b\}$

→ Elimination of null productions

$S \to as \mid A$

$A \to \lambda$

↓

transformed as

↓

$S \to as \mid \lambda$

$S \to as \mid a$ → one substitution of λ we get a new production

---

$S \to as \mid aA \mid a$

$A \to \lambda$

$S \to as \mid aA \mid a \mid \lambda$

↑

as there are no productions for → we can delete it

$S \to as \mid a \mid \lambda$

substitute $\lambda \Rightarrow$ $S \to as \mid a \mid a$

$S \to as \mid a$

$S \to ABAC$

$A \to aA \mid \epsilon \Rightarrow A \to aA \mid a$

$B \to bB \mid \epsilon \Rightarrow B \to bB \mid b$

$C \to c$.

$S \to ABAC \mid BAC \mid ABC \mid AAC \mid BC \mid C \mid AC$

$A \to aA \mid a$

$B \to bB \mid b$

$C \to c$.

Ex:- $S \to a \mid xb \mid aYa$

$x \to Y \mid \epsilon$

$Y \to b \mid \epsilon$

eliminate $x \to \epsilon$

$S \to a \mid xb \mid aYa \mid b$

$x \to Y$

$Y \to b \mid \epsilon$

eliminate $Y \to \epsilon$

$\Rightarrow S \to a \mid xb \mid aYa \mid b \mid aa$

$x \to Y$ x

$Y \to b$

$\Rightarrow S \to a \mid bb \mid aYa \mid aa \mid b$

$Y \to b$.

Ex:- $S \to ABAC \mid B \mid AC$

$A \to aA \mid \epsilon$

$B \to \epsilon$ —— eliminate

$C \to c \mid bc \mid \epsilon$

$S \to AAC \mid AC$

$\Rightarrow A \to aA \mid \epsilon$

$C \to c \mid bc \mid \epsilon$.

eliminate $A \to \epsilon$

$S \to AAC \mid AC \mid C$

$A \to aA \mid a$

$C \to c \mid bC \mid \epsilon$

eliminate $C \to \epsilon$

$S \to AAC \mid AC \mid AA \mid A$

$A \to aA \mid a$

$C \to c \mid bC \mid b$.

Q:- $C \to D$, $D \to a$ $\Rightarrow$ $C \to a$ & $B \to C$, $C \to a$ $\Rightarrow$ $B \to a$

So reduced production rules

$S \to AB$
$A \to a$
$B \to c \mid b$
$C \to D$
$D \to a$

$S \to AB$
$A \to a$
$B \to a \mid b$.

Q:- $S \to A \mid bb$
$A \to B \mid b$
$B \to S \mid a$.

$A \to B$, $B \to S$ $\Rightarrow$ $A \to S$

$S \to A \mid bb$
$A \to S \mid b$
$B \to b \mid a$

$A \to B$, $B \to b \mid a$ $\Rightarrow$ $A \to b \mid a$ $\Rightarrow$

$S \to A \mid bb$
$A \to b \mid a$

$S \to A$, $A \to b \mid a$ $\Rightarrow$ $S \to b \mid a$ $\Rightarrow$ $S \to b \mid a \mid bb \mid S$

Q:- $S \to ABaC$
$A \to BC$
$B \to b \mid \in$
$C \to D \mid \in$
$D \to d$.

$C \to D$, $D \to d$ $\Rightarrow$ $C \to d$.

$S \to ABaC$
$A \to BC$
$B \to b \mid \in$
$C \to d \mid \in$

$\Rightarrow$

$S \to ABaC \mid ABa$
$A \to BC \mid B$
$B \to b \mid \in$.
$C \to d$.

$S \rightarrow ABaC \mid ABa \mid AaC \mid Aa$

$A \rightarrow BC \mid B \mid \Lambda.$

$B \rightarrow b$

$C \rightarrow d$

$S \rightarrow Bac \mid ABac \mid ac \mid Aac \mid Ba \mid ABa \mid a \mid Aa$

$A \rightarrow C \mid BC \mid B$

$B \rightarrow b$

$C \rightarrow d.$

$A \rightarrow C$
$C \rightarrow d \quad \Rightarrow \quad A \rightarrow d$

$S \rightarrow Ba \mid ABa \mid a \mid Aa \mid Bad \mid ABad \mid ad \mid Aad$

$A \rightarrow d \mid Bd \mid b$

$B \rightarrow b$

→ Unit production :- Non-terminal derives only one non-terminal.

Normal forms:-

1. chomsky normal form (CNF)
2. Greibach normal form (GNF)

CNF:-

Rules of CNF:-

1. one non-terminal derives only two non-terminals or derives only one terminal.

$S \rightarrow AB/a$
$A \rightarrow a$
$B \rightarrow b$

→ Ex1- $S \rightarrow ABA$, $A \rightarrow a$, $B \rightarrow b$ } CFG

Step-1:- identify the production rules that are not in CNF format.

Here, we have $S \rightarrow ABA$.

Step-2:- convert to CNF

let $C \rightarrow AB \Rightarrow S \rightarrow CA$.

$S \rightarrow CA$
$A \rightarrow a$
$B \rightarrow b$
$C \rightarrow AB$.

check again whether all are in CNF format or not

→ Ex2- $S \rightarrow ABA$
$A \rightarrow aA/\epsilon$
$B \rightarrow bB/\epsilon$

Rule-1 :- Simplify the CFG

Rule-2 :- apply CNF rules for simplified CFG.

Sol:- Simplification.

Step-1:- find useless symbols

Here no useless productions

Step-2 :- eliminating null productions

→ A has null production

S → ABA | AB | BA | B

A → aA | a

B → bB | ε.

→ B has null production

S → ABA | AB | BA | B | AA | A ~~| ε~~

A → aA | a

B → bB | b.

Step-3 :- eliminating unit productions

S → B

S → A.

S → ABA | AB | BA | ~~B~~ | A | aA | a | bB | b

A → aA | a

B → bB | b

Step-4 - again check for useless symbols

ii, CNF

S → ABA(×) | AB(✓) | BA(✓) | AA(✓) | aA(×) | a(✓) | bB(×) | b(✓)

A → aA(×) | a(✓)

B → bB(×) | b(✓)

⇒ S → CA | AB | BA | AA | aA | a | bB | b

A → aA | a

B → bB | b

C → AB.

⇒ $S \to CA \mid AB \mid BA \mid AA \mid A'A \mid a \mid B'B \mid b$

$A \to aA \mid a$ $\quad A' \to a$

$B \to bB \mid b$ $\quad B' \to b.$

$C \to AB$

$S \to CA \mid AB \mid BA \mid AA \mid A'A \mid a \mid B'B \mid b$

⇒

$A \to A'A \mid a$ $\quad A' \to a$

$B \to B'B \mid b$ $\quad B' \to b.$

$C \to AB$

$S \to AB \mid aB$

$A \to aab \mid \in$

$B \to bbA.$

i) No useless symbols

ii) eliminate null productions

→ A has null production

$S \to AB \mid aB \mid B$

$A \to aab$

$B \to bbA \mid bb.$

iii) eliminate unit production

$S \to \overset{\checkmark}{AB} \mid a\overset{\times}{B} \mid bb\overset{\times}{A} \mid bb\overset{\times}{}.$

$A \to aa\overset{\times}{a}b$

$B \to bb\overset{\times}{A} \mid bb\overset{\times}{}.$

iv) $S \to AB \mid A'B \mid bbA \mid bb$

$A \to aab$

$B \to bbA \mid bb.$

$A' \to a$

v) $S \to AB \mid A'B \mid B'B'A \mid B'B'$

$A \to A'A'B'$

$B \to B'B'A \mid B'B'$

$A' \to a$

$B' \to b$

vi. $S \to AB \mid A'B \mid CA \mid B'B'$

$A \to DB'$

$B \to CA \mid B'B'$

$A' \to a$

$B' \to b$

$C \to B'B'$

$D \to A'A'$

Ex:- $S \to ASB \mid \epsilon$

$A \to aAS \mid a$

$B \to SbS \mid A \mid bb.$

i. No useless productions

ii. eliminate null productions

→ S has null production.

$S \to ASB \mid AB$

$A \to aAS \mid a \mid aA$

$B \to SbS \mid A \mid bb \mid Sb \mid bS \mid b.$

iii. eliminate unit productions.

$B \to A$

$S \to ASB \mid AB$

$A \to aAS \mid a \mid aA$

$B \to SbS \mid aAS \mid a \mid aA \mid bb \mid Sb \mid bS \mid b.$

iv. $S \to CB \mid AB$

$A \to aAS \mid a \mid aA$

$B \to SbS \mid aAS \mid a \mid aA \mid bb \mid Sb \mid bS \mid b$

$C \to AS.$

$S \to CB \mid A$

$A \to A'C \mid a \mid A'A$

$B \to SbS \mid A'C \mid a \mid A'A \mid bb \mid Sb \mid bS \mid b$

$C \rightarrow AS$.

$A' \rightarrow a$

$S \rightarrow CB/A$

$A \rightarrow A'C/a/A'A$

$B \rightarrow SB'S/A'C/a/A'A/B'B'/SB'/B'S/b$.

$C \rightarrow AS$

$A' \rightarrow a$

$B' \rightarrow b$.

$S \rightarrow CB/A$

$A \rightarrow A'C/a/A'A$

$B \rightarrow DS/A'C/a/A'A/B'B'/SB'/B'S/b$

$C \rightarrow AS$

$A' \rightarrow a$

$B' \rightarrow b$

$D \rightarrow SB'$

GNF:-

A context free language is said be GNF is all productions are of the form $A \rightarrow aX$, where $a \in T$ $X \in V^*$.

Ex:- $S \rightarrow aA/a$

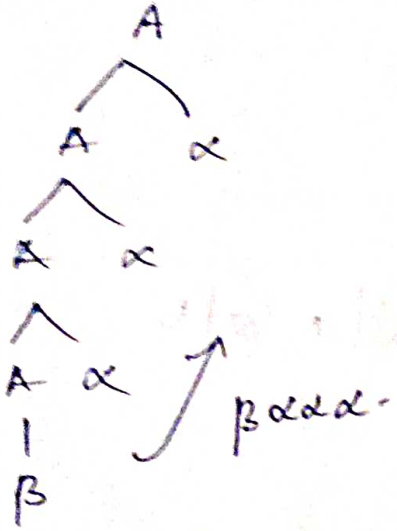
$A \rightarrow a$

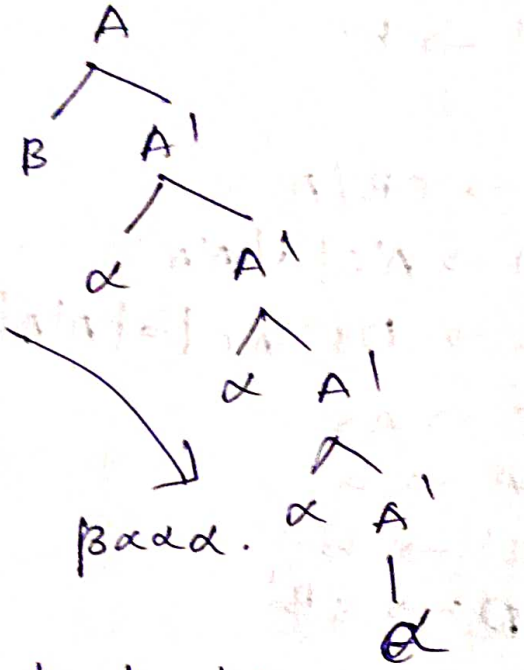
Lemma 1:- [substitution theorem] Let $G=(V,T,P,S)$ be a CFG. Let $A \rightarrow B\alpha$ be a production in P and $B \rightarrow B1/B2/B3/B4$---. The equivalent grammar can be obtained by substituting B in A then the resulting grammar is $A \rightarrow B1\alpha/B2\alpha/B3\alpha/B4\alpha$---

→ Lemma-2 [elimination of left recursion]
Grammar of the form $A \to A\alpha | \beta$ is called left recursive grammar.

To eliminate left recursion, we need to substitute the following.

$$A \to \beta A' | \beta$$
$$A' \to \alpha A' | \alpha$$





Ex:- $A \to A\alpha_1 | A\alpha_2 | A\alpha_3 | \beta_1 | \beta_2 | \beta_3$

Sol:- $A \to \beta_1 A' | \beta_2 A' | \beta_3 A' | \beta_1 | \beta_2 | \beta_3$

$A' \to \alpha_1 A' | \alpha_2 A' | \alpha_3 A' | \alpha_1 | \alpha_1 | \alpha_3$

→ Converting given grammar to GNF:-

Rules

1. eliminate null productions and unit production then construct CNF.
2. Rename variables as $A_1, A_2, A_3, \ldots$ with $S = A_1$
3. For each production of the form $A_i \to A_j\alpha$ apply the following.

a) if $j > i$, leave production as it is

b) if $j = i$, apply lemma 2

c) if $j < i$, apply lemma 1

for each production of the form

$A_i \rightarrow A_j\alpha$, where $i < j$ apply substitution lemma if A_j is in gnf to bring A_i to GNF

$S \rightarrow AA \mid a$

$A \rightarrow SS \mid b.$

d) no null & unit productions and is in CNF.

e) Rename.

$A_1 \rightarrow A_2A_2 \mid a$ $\quad [S = A_1, A = A_2]$

$A_2 \rightarrow A_1A_1 \mid b.$

f) $A_1 \rightarrow A_2A_2 \mid a$ – leave it

$i < j$

$A_2 \rightarrow A_1A_1 \mid b$ – apply lemma 1

$i > j$

$A_2 \rightarrow A_2A_2A_1 \mid aA_1 \mid b.$ – apply lemma 2

$i = j$ (A: A_2; α: A_2A_1; β_1: aA_1; β_2: b)

$A_2 \rightarrow aA_1A' \mid bA' \mid aA_1 \mid b.$

$A' \rightarrow A_2A_1A' \mid A_2A_1$ $\quad [A' = A_3]$

$i > j$

and A_2 is in GNF, so apply substitution

$A' \rightarrow aA_1A'A_1A' \mid bA'A_1A' \mid aA_1A_1A' \mid bA_1A' \mid$

$aA_1A'A_1 \mid bA'A_1 \mid aA_1A_1 \mid bA_1$

Final productions

$A_1 \to A_2A_2 \mid a$ — not in GNF

$A_1 \to aA_1A_2' \mid bA'A_2 \mid aA_1A_2 \mid bA_2 \mid a$

$A_2 \to aA_1A' \mid bA' \mid aA_1 \mid b.$

$A' \to aA_1A'A_1A' \mid bA'A_1A' \mid aA_1A_1A' \mid bA_1A' \mid$
$aA_1A'A_1 \mid bA'A_1 \mid aA_1A_1 \mid bA_1$

Ex:- $S \to XA \mid BB$ [wrong]

$B \to b \mid SB$

$X \to b$

$A \to a$

∴ no null & unit productions & it is CNF.

2. Rename.

$A_1 \to A_2A_3 \mid A_4A_4$

$A_4 \to b \mid A_1A_4$

$A_2 \to b$

$A_3 \to a$

$S = A_1$
$X = A_2$
$A = A_3$
$B = A_4$

$A_1 \to A_3A_4 \mid A_2A_2$

$A_2 \to b \mid A_1A_2$

$A_3 \to b$

$A_4 \to a$

3. $A_1 \to A_3A_4 \mid A_2A_2$ $\quad [i < j_1, i < j_2]$ leave
$\quad i \quad j_1 \quad j_2$

$A_2 \to b \mid A_1A_2$ $\quad [i > j]$
$\quad i \quad j$

apply lemma 1

$A_2 \to b \mid A_3A_4A_2 \mid A_2A_2 \mid A_2A_2$ $\quad [i < j_1, i = j_2]$
$\quad A \quad \beta_1 \quad \beta_2 \quad A \quad \alpha$
$\quad i \quad j_1 \quad j_2$

apply lemma 2

Ex:-

$S \to ABA$

$A \to aA \mid \epsilon$

$B \to bB \mid \epsilon$

1. Remove null productions.

$A \to \epsilon \quad B \to \epsilon$.

$S \to ABA \mid BA \mid AB \mid B$

$A \to aA \mid a$

$B \to bB \mid \epsilon$.

$S \to ABA \mid BA \mid AB \mid AA \mid B \mid A$

$A \to aA \mid a$

$B \to bB \mid b$.

remove unit productions

$S \to A \quad S \to B$

$S \to \overset{\times}{ABA} \mid \overset{\checkmark}{BA} \mid \overset{\checkmark}{AB} \mid \overset{\checkmark}{AA} \mid \overset{\times}{aA} \mid \overset{\checkmark}{a} \mid \overset{\times}{bB} \mid \overset{\checkmark}{b}$.

$A \to \overset{\times}{aA} \mid \overset{\checkmark}{a}$

$B \to \overset{\times}{bB} \mid \overset{\checkmark}{b}$

Convert to CNF.

$S \to CA \mid BA \mid AB \mid AA \mid A^1A \mid a \mid B^1B \mid b$

$A \to A^1A \mid a$

$B \to B^1B \mid b$

$C \to AB$

$A^1 \to a$

$B^1 \to b$

rename.

$A_1 \rightarrow A_4A_2 \mid A_3A_2 \mid A_2A_3 \mid A_2A_2 \mid A_5A_2 \mid a \mid A_6A_2 \mid b$

$A_2 \rightarrow A_5A_2 \mid a$

$A_3 \rightarrow A_6A_3 \mid b$

$A_4 \rightarrow A_2A_3$

$A_5 \rightarrow a$

$A_6 \rightarrow b$.

First production $[i < j]$ = leave.

$A_2 \rightarrow A_5A_2 \mid a$ $[i < j]$ = leave.

$A_3 \rightarrow A_6A_3 \mid b$ $[i < j]$ = leave.

$A_4 \rightarrow A_2A_3$ $[i > j]$ remove.

$A_4 \rightarrow A_5A_2A_3 \mid aA_3$ $[i < j]$ leave.

Final productions.

$A_1 \rightarrow A_5A_2A_3A_2 \mid aA_3A_2 \mid A_6A_3A_2 \mid bA_2 \mid A_5A_2A_3 \mid aA_3 \mid A_5A_2A_2 \mid aA_2 \mid aA_2 \mid a \mid bA_3 \mid b$.

$A_1 \rightarrow aA_2A_3A_2 \mid aA_3A_2 \mid bA_3A_2 \mid bA_2 \mid aA_2A_3 \mid aA_3 \mid aA_2A_2 \mid aA_2 \mid a \mid bA_3 \mid b$

$A_2 \rightarrow aA_2 \mid a$

$A_3 \rightarrow bA_3 \mid b$

$A_4 \rightarrow aA_3$

$A_5 \rightarrow a$

$A_6 \rightarrow b$.

* Pumping lemma for Context free languages:-

Let L be any context free language, then there is a constant 'n' which depends only upon 'L'. $\exists$ a string z belongs to L and $|z| \geq n$ where $z = uvwxy$ such that

→ Rule-1:- $|vwx| \leq n$

2:- $|vx| \neq \epsilon$ or $|vx| > 1$

3:- for all $i \geq 0$ $uv^iwx^iy \in L$

Ex:- $L = \{a^nb^n / n \geq 1\}$

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

$z = \underset{u}{a}\underset{v}{a}\underset{x}{b}\underset{y}{b}$.

$i=0$ $uv^0wx^0y \rightarrow ab$ → its a CFL

$i=1$ $uv^1wx^1y \rightarrow aabb$

Ex:- $L = \{a^nb^{2n} / n \geq 1\}$

$L = \{abb, aabbbb, aaabbbbbb, ---\}$

$z = \underset{u}{a}\underset{v}{a}\underset{w}{b}\underset{x}{b}\underset{y}{bb}$

$i=0$ $uv^0wx^0y = abb$ → it's a CFL

$i=1$ $uv^1wx^1y = aabbbb$

* closure properties:-

① CFL's are closed under union

$a^nb^n \rightarrow L_1 \rightarrow CFL$

$a^{2n}b^n \rightarrow L_2 \rightarrow CFL$ + $a^{3n}b^{2n} \rightarrow CFL$

$S \rightarrow S_1 | S_2$

CFL's are closed under concatenation

$S \rightarrow S_1 \cdot S_2$

CFL's are under closure

$S^* \rightarrow S \mid \epsilon$

CFL's are not closed under intersection

$a^n b^n c^n \rightarrow$ CFL

$a^m b^n c^n \rightarrow$ CFL

$\cap$ ———————————

$a^n b^n c^n \rightarrow$ not in CFL.

CFL's are not under complement

$L = a^m b^m c^m \rightarrow$ CFL

L^* has $a^n b^n c^n \mid b^n c^n a^n \mid c^n b^n a^n \mid a^n c^n b^n \mid \ldots$

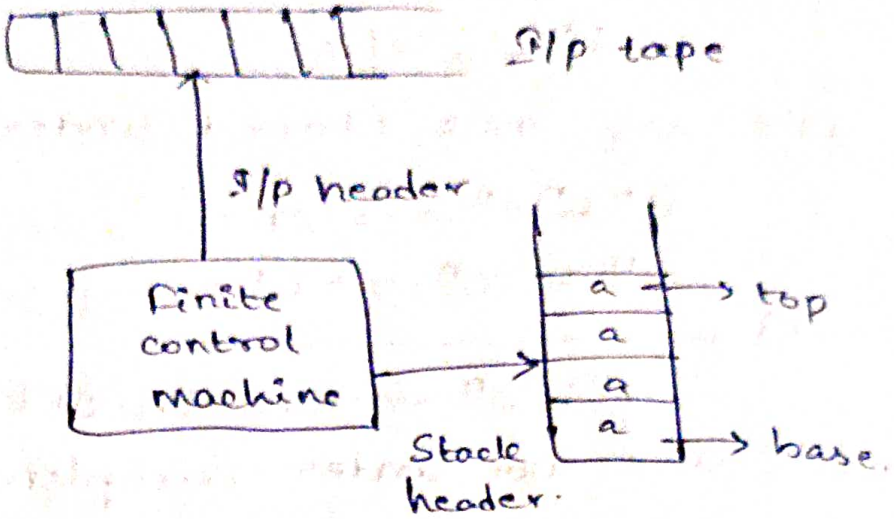
all these are not CFL's.

Decision algorithms:-

1) CFL is finite or not
2) CFL is empty or not
3) Given word is in CFL or not
4) Are the 2 CFG's equivalent or not
5) Whether the CFG is ambiguous or not.

Unit-4

Push Down-Automata



PDA:- It is a 7 tuple machine

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

Q = Finite no. of states

Σ = Finite no. of input symbols

Γ = Finite no. of stack symbols [push down]

q_0 = initial state

z_0 = stack initial symbol

F = Final states

δ = Transition function.

Stack - infinite

Mapping function: $Q \times \Sigma^* \times \Gamma^* \rightarrow Q \times \Gamma^*$

→ **Rules:-**

1) $\delta(q_0, a, z_0) = (q_1, az_0)$ indicates that in the state q_0 on seeing 'a' from input tape, a is flushed onto the stack and gets the output state as q_1

2) $\delta(q_0, a, z_0) : (q_0, \epsilon)$ → popping. deleting the top of stack and move the I/p header forward.

$\delta(q_0, a, z_0) = (q_1, z_0)$

we do no action but just skip the input variable.

Ex: $L = \{a^n b^n / m \geq 1\}$

string = aabbb → on a push and on b pop.

$\delta[q_0, \epsilon, \epsilon) = (q_0, z_0)$

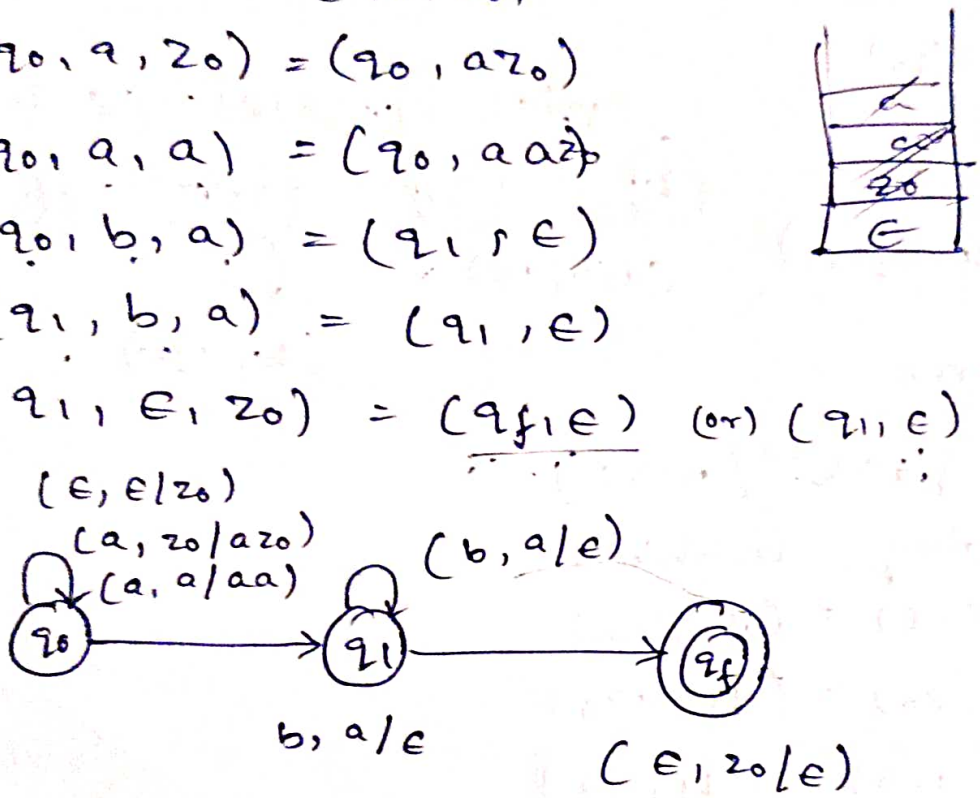
$\delta(q_0, a, z_0) = (q_0, az_0)$

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

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

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

$\delta(q_1, \epsilon, z_0) = (q_f, \epsilon)$ (or) (q_1, ϵ)



Ex: $L = \{a^{2n} b^n / n > 1\}$

string = aaaabb

$\delta(q_0, \epsilon, \epsilon) = (q_0, z_0)$

$\delta(z_0, a, z_0) = (q_1, az_0)$

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

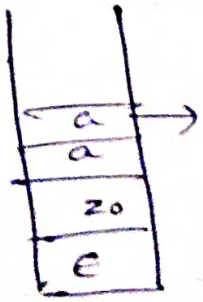
$\delta(q_2, a, a) = (q_1, aa)$

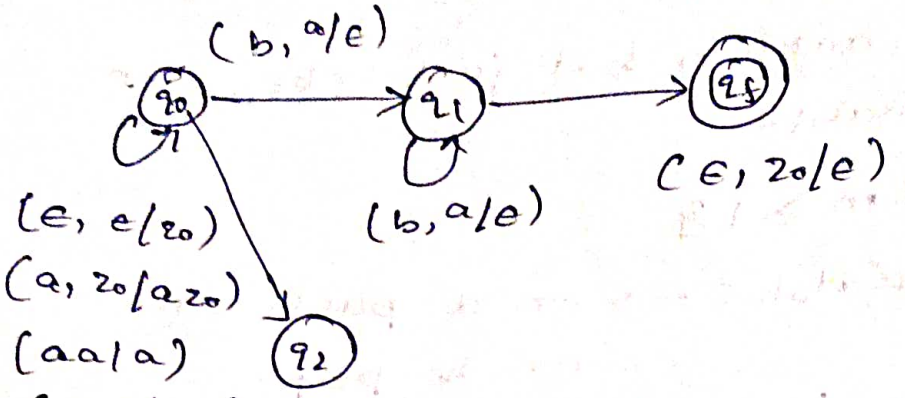
$\delta(q_1, a, a) = (q_2, a)$

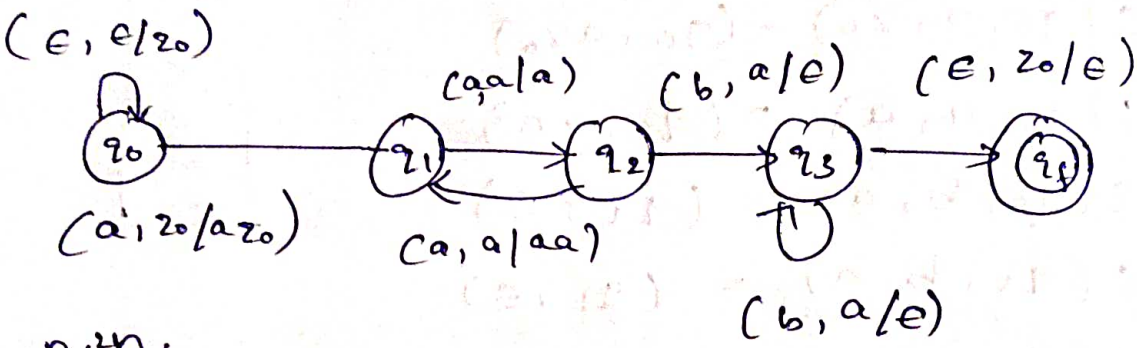
$\delta(q_2, b, a) = (q_3, \epsilon)$

$\delta(q_3, b, a) = (q_3, \epsilon)$

$\delta(q_3, \epsilon, z_0) = (q_f, \epsilon)$







Q1- $a^n b^{2n}$.

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

string = aabbbb

$\delta(q_0, \epsilon, \epsilon) = (q_0, z_0)$

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

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

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

$\delta(q_2, b, z_0) = (q_2, z_0)$

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

Acceptance of PDA

Acceptance by final state $(q, w, z_0) \vdash^* (p, \epsilon, \gamma)$ (i/p string; final state; stack symbols)

Acceptance by empty stack. $(q, w, z_0) \vdash (p, \epsilon, \gamma)$

$\left[\begin{matrix} p \in Q \\ \gamma \in \epsilon \end{matrix}\right]$

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

If at last transition

$\delta(q_1, \epsilon, z_0) = (q_2, z_0)$ — use it for acceptance by final state

$\delta(q_1, \epsilon, z_0) = (q_1, \epsilon)$ — use for acceptance by empty stack.

Let us consider string aabb

$(q_0, aabb, z_0) \vdash (q_0, abb, az_0)$

$\vdash (q_0, bb, aaz_0)$

$\vdash (q_1, b, az_0)$

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

$\vdash (q_2, z_0)$ or $[(q_1, \epsilon)]$.

entire string is completed and we reached final state. So this is language is accepted by PDA by final state.

⇒ if in above we just change the last step then we say the language is accepted by PDA by empty stack.

Converting grammar to PDA

$S \to AA/0A$

$A \to a$

$\delta(q, \epsilon, S) \to (q, AA)$

$\delta(q, 0, S) \to (q, A)$

$\delta(q, a, A) \to (q, \epsilon)$

Ex: $S \to aAA$

$A \to bS/aS/a$

$\left[\begin{array}{l}\delta(q,a,a) = (q,\epsilon) \\ \delta(q,b,b) = (q,\epsilon)\end{array}\right]$ – rules

↓ pop

$\delta(q, a, S) = (q, AA)$

$\delta(q, b, A) = (q, S)$ ← add.

$\delta(q, a, A) = (q, S)$

$\delta(q, a, A) = (q, \epsilon)$

Ex: $S \to abAB/ba$

$A \to aaa$

$B \to aA/bb.$

$\delta(q, a, S) = (q, bAB)$

$\delta(q, b, S) = (q, a)$

$\delta(q, a, A) = (q, aa)$

$\delta(q, a, B) = (q, A)$

$\delta(q, b, B) = (q, b)$

$\delta(q, a, a) = (q, \epsilon)$

$\delta(q, b, b) = (q, \epsilon)$

* Constructing CFG for a given PDA:-

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

We will construct a grammar G such that

$L(G) = L(M)$

→ Rules :-

1. The productions for the start symbol 'S' are given by $S \to [q_0, z_0, q]$ for each state q in Q.

Ex: $Q = \{q_0, q_1\}$

$\delta \rightarrow [q_0, z_0, q_0] \,/\, [q_0, z_0, q_1]$

2. For each (pop) move that pops a symbol from stack with transition $\delta(q, a, z_0) = (q, \epsilon)$

$[q, z_0, q'] \rightarrow a$ for each q' in Q.

Ex:- $\delta(q_0, a, A) = (q_0, \epsilon)$ $\quad Q = \{q_0, q_1\}$

$[q_0, A, q_0] \rightarrow a \,/\, [q_0, A, q_1] \rightarrow a$

3. For each move that doesn't pop symbol from stack with transitions as $\delta(q', a, z_0) = (q_1, z_1 z_2)$

$[q', z_0, q''] \rightarrow a[q_1, z_1, q_2][q_2, z_2, q_3][q_3, z_3, q_4]$ ---- $[q_m, z_m, q'']$

Examples:-

1. PDA:-

1) $\delta(q_0, b, z_0) = (q_0, zz_0)$
2) $\delta(q_0, \epsilon, z_0) = (q_0, \epsilon)$
3) $\delta(q_0, b, z) = (q_0, zz)$
4) $\delta(q_0, a, z) = (q_1, z)$
5) $\delta(q_1, b, z) = (q_1, \epsilon)$
6) $\delta(q_1, a, z_0) = (q_0, z_0)$

$Q = \{q_0, q_1\}$ $\quad \Sigma = \{a, b\}$ $\quad z_0 = z_0$

1) $\delta(q_0, b, z_0)$ — apply rule 3. $= (q_0, zz_0)$

$[q_0, z_0, q_0] = b[q_0, z, q_0][q_0, z_0, q_0]$

$[q_0, z_0, q_0] = b[q_0, z, q_1][q_1, z_0, q_0]$

$[q_0, z_0, q_1] = b[q_0, z, q_0][q_0, z_0, q_1]$

$[q_0, z_0, q_1] = b[q_0, z, q_1][q_1, z_0, q_1]$

3) $\delta(q_0, b, z) = (q_0, zz)$ apply 3rd rule.

$[q_0, z, q_0] = b[q_0, z, q_0][q_0, z, q_0]$

$[q_0, z, q_1] = b[q_0, z, q_0][q_0, z, q_1]$

$[q_0, z, q_0] = b[q_0, z, q_1][q_1, z, q_0]$

$[q_0, z, q_1] = b[q_0, z, q_1][q_1, z, q_1]$

4) $\delta(q_0, a, z) = (q_1, z)$ apply 3rd rule

$[q_0, z, q_0] = a[q_1, z, q_0]$ ~~$[q_0, z, q_1]$~~.

$[q_0, z, q_1] = a[q_1, z, q_1]$

6) $\delta(q_1, a, z_0) = (q_0, z_0)$ apply 3rd rule.

$[q_1, z_0, q_0] = a[q_0, z_0, q_0]$

$[q_1, z_0, q_1] = a[q_0, z_0, q_1]$

2) $\delta(q_0, \epsilon, z_0) = (q_0, \epsilon)$ apply 2nd rule.

$[q_0, z_0, q_0] \to \epsilon$

~~$[[q_0, z_0, q_1] \to \epsilon]$~~ $[q_0, z_0, q_1] \Rightarrow \epsilon$

5) $\delta(q_1, b, z) = (q_1, \epsilon)$ apply 2nd rule.

~~$[[q_1, z, q_0] = b]$~~ $[q_1, z, q_0] = b$

$[q_1, z, q_1] = b$.

$Q = \{q_0, q_1\}$ $\Sigma = \{a, b\}$ $\Gamma = \{z, z_0\}$

$[q_0, z, q_0] \to A_1$

$[q_0, z, q_1] \to A_2$

$[q_1, z, q_0] \to A_3$

$[q_1, z, q_1] \to A_4$

$[q_0, z_0, q_0] \to A_5$

$[q_0, z_0, q_1] \to A_6$

$[q_1, z_0, q_0] \to A_7$

$[q_1, z_0, q_1] \to A_8$.

1) $A_5 \to bA_1A_5$

$A_6 \to bA_1A_6$

$A_5 \to bA_2A_7$

$A_6 \to bA_2A_8$.

2, $A_5 \to \epsilon$

$A_6 \to \epsilon$

3, $A_1 \to bA_1A_1$

$A_2 \to bA_1A_2$

$A_1 \to bA_1A_3$

$A_1 \to bA_1A_4$

4, $A_1 \to aA_3$

$A_2 \to aA_4$

5, $A_3 \to b$

$A_4 \to b$

6, $A_7 \to aA_5$

$A_8 \to aA_6$.

Q:- 1, $\delta(q,1,z) = (q, xz)$

2, $\delta(q,1,x) = (q, xx)$

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

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

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

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

$Q = \{p, q\}$

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

$\Gamma = \{x, z\}$

1, $\delta(q,1,z) = (q, xz)$ apply rule 3.

$[q,z,q] = 1[q,x,q][q,z,q]$

$[q,z,p] = 1[q,x,q][q,z,p]$

$[q,z,q] = 1[q,x,p][p,z,q]$

$[q,z,p] = 1[q,x,p][p,z,p]$

2, Rule 3

$[q,x,q] = 1[q,x,q][q,x,q]$

$[q, x, p] = 1[q, x, q][q, x, p]$

$[q, x, q] = 1[q, x, p][p, x, q]$

$[q, x, p] = 1[q, x, p][p, x, p]$

3. apply rule 2

$[q, x, q] = \in$

$[q, x, p] = \in$

4. apply rule 3

$[q, x, q] = 0[p, x, q]$

$[q, x, p] = 0[p, x, p]$

5. apply rule 2

$[p, x, p] = \in$

$[p, x, q] = \in$

6. apply rule 3.

$[p, z, q] = 0[q, z, q]$

$[p, z, p] = 0[q, z, p]$

$[q, z, p] \to A_1$

$[q, z, q] \to A_2$

$[p, z, p] \to A_3$

$[p, z, q] \to A_4$

$[q, x, p] \to A_5$

$[q, x, q] \to A_6$

$[p, x, p] \to A_7$

$[p, x, q] \to A_8$.

1. $A_2 \to 1A_6A_2$

$A_1 \to 1A_6A_1$

$A_2 \to 1A_5A_4$

$A_1 \to 1A_5A_3$

2. $A_6 \to 1A_6A_6$

$A_5 \to 1A_6A_5$

$A_6 \to 1A_5A_8$

$A_5 \to 1A_5A_7$

3. $A_6 \to \in$

$A_5 \to \in$

4. $A_6 \to 0A_8$

$A_5 \to 0A_7$

5. $A_7 \to \in$

$A_8 \to \in$

6. $A_4 \to 0A_2$

$A_3 \to 0A_1$

Unit-5

Turing Machines

A turing machine is a 7 tuple $(Q, \Sigma, \Gamma, \delta, q_0, B, F)$ where

- Q - Finite set of states
- Σ - Finite set of input alphabets
- Γ - Tape alphabets $(\Sigma \cup B)$
- δ - mapping function
- q_0 - initial state
- B - Blank state
- F - Final state

→ Mapping function of turing machine.

$$\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$$

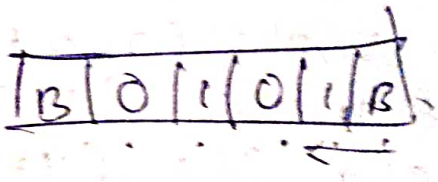
Ex:- $0^{m,n}$ Ex:- 001

Transitions

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

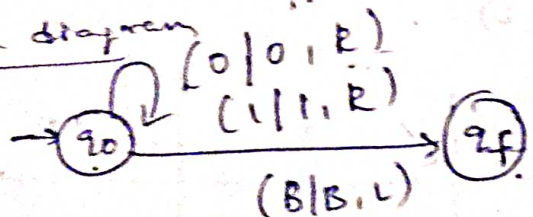
$\delta(q_0, 1) = (q_0, 1, R)$

$\delta(q_0, B) = (q_f, B, L)$



Language - $(0+1)^*$

Transition diagram



Transition table:

	0	1	B
q_0	$(q_0, 0, R)$	$(q_0, 1, R)$	$(q_f, B, L$
q_f	—	—	—

⇒ input tape values can be modified.

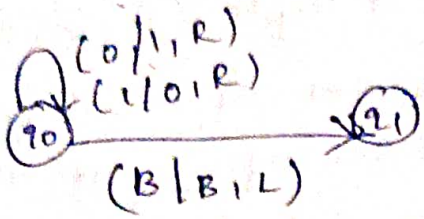
Q:- 1's compliment of a given number.

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

$\delta(q_0, 0) = (q_0, 1, R)$

$\delta(q_0, 1) = (q_0, 0, R)$

$\delta(q_0, B) = (q_f, B, L)$



	0	1	B
q_0	$(q_0, 1, R)$	$(q_0, 0, R)$	(q_f, B, L)
q_f	–	–	–

⇒ turing tape has blanks.

Q:- $0^{m}1^{n}$ where $m, n \geq 1$

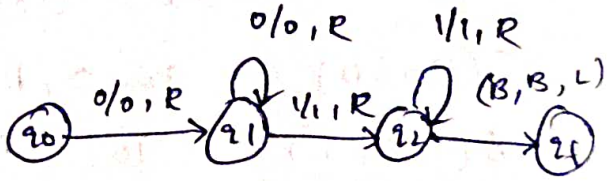
$\delta(q_0, 0) = (q_1, 0, R)$

$\delta(q_1, 0) = (q_1, 0, R)$

$\delta(q_1, 1) = (q_2, 1, R)$

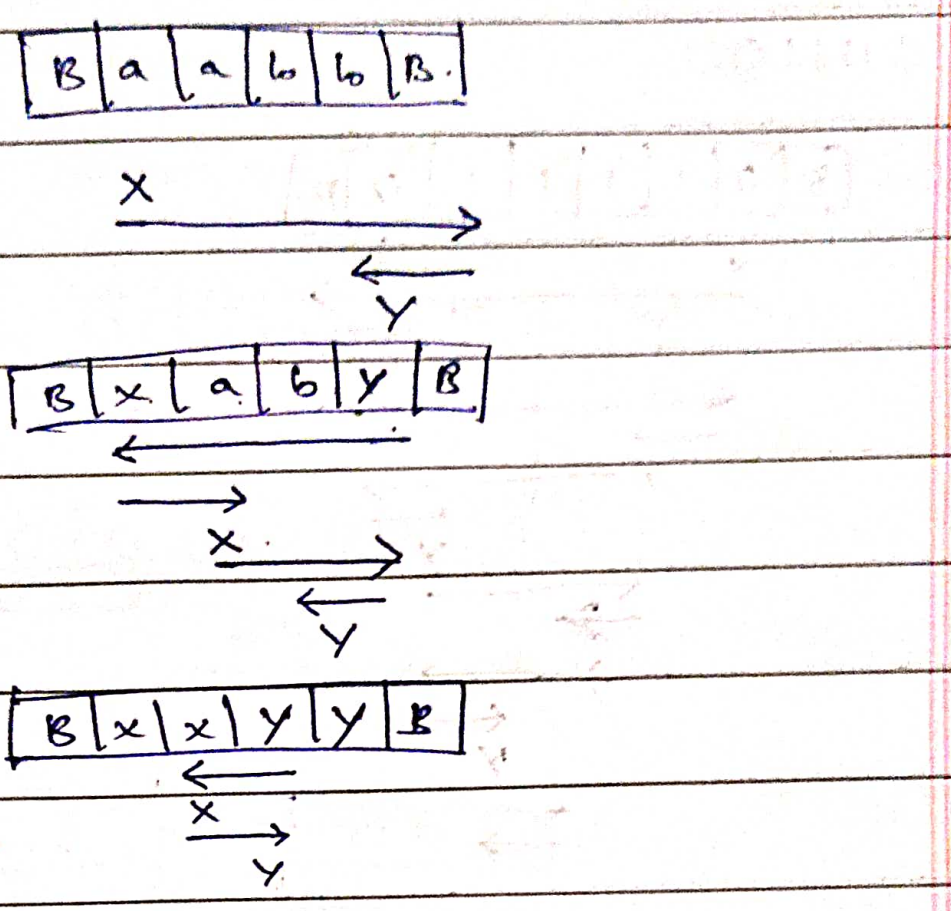
$\delta(q_2, 1) = (q_2, 1, R)$

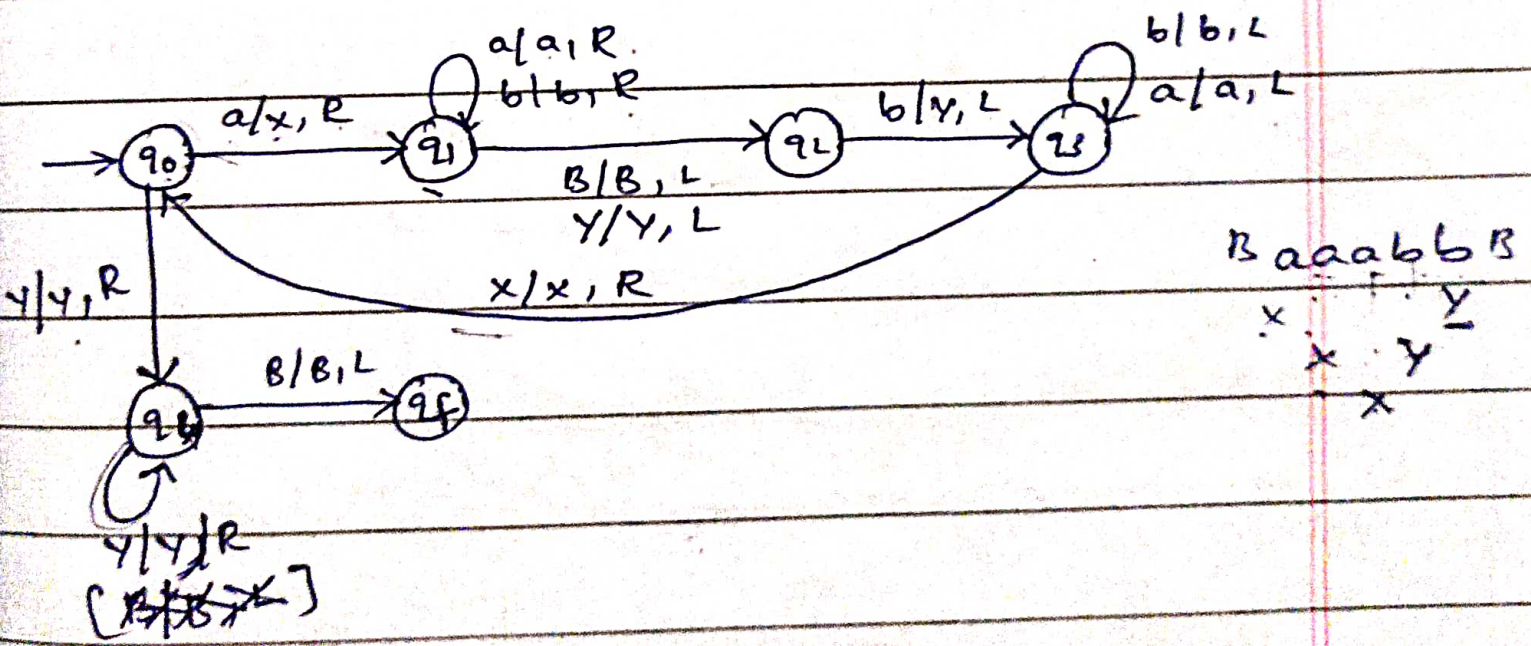
$\delta(q_1, B) = (q_f, B, L)$.



	0	1	B
q_0	$(q_1, 0, R)$	–	–
q_1	$(q_1, 0, R)$	$(q_2, 1, R)$	–
q_2	–	$(q_2, 1, R)$	(q_f, B, L)
q_f	–	–	–

Ex:- $L = \{a^n b^n / n \geq 1\}$ aabb.

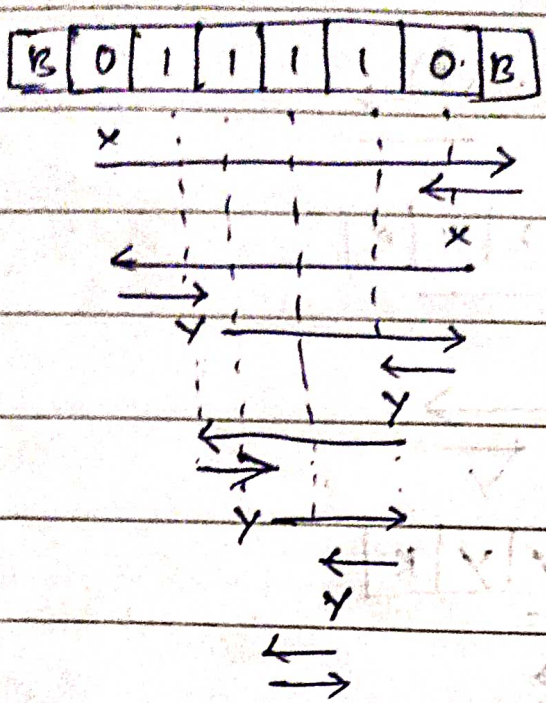


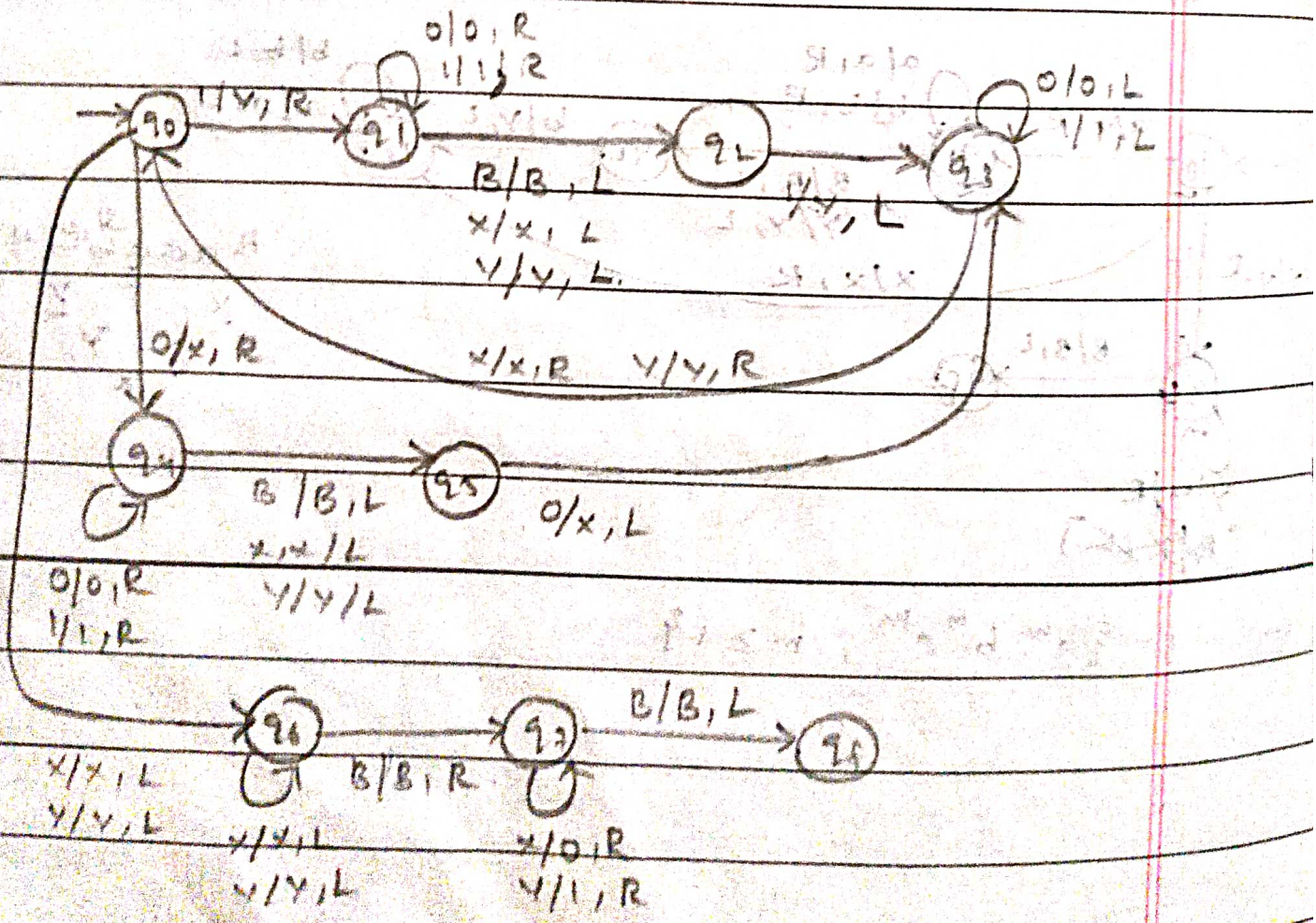


Ex:- $L = \{a^m b^m c^m, m \geq 1\}$

ex1. $L = \{ww^R / w \in \Sigma\}$ ... palindrome.

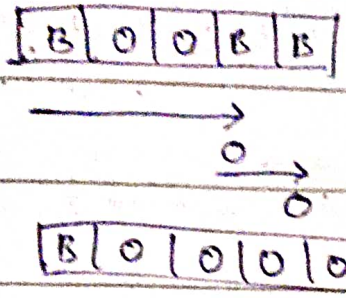
011110





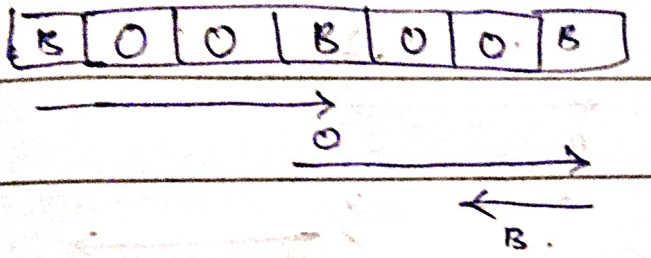
Recursive functions :-

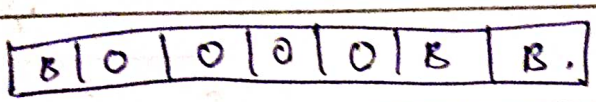
Ex1. $f(x) = x+2$



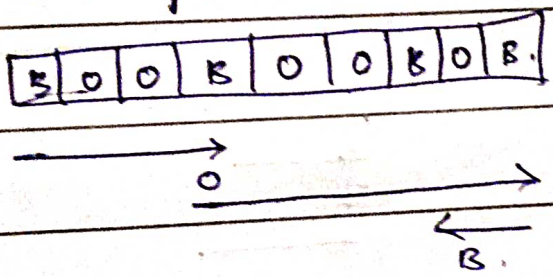
o/p = 4.

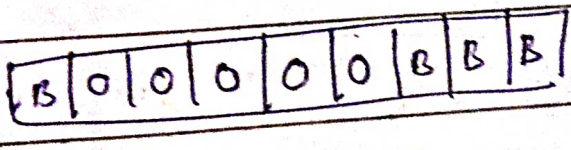
Ex1- $f(x) = x+y$.



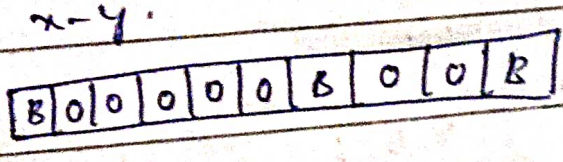


Ex1- $f(x) = x+y+z$

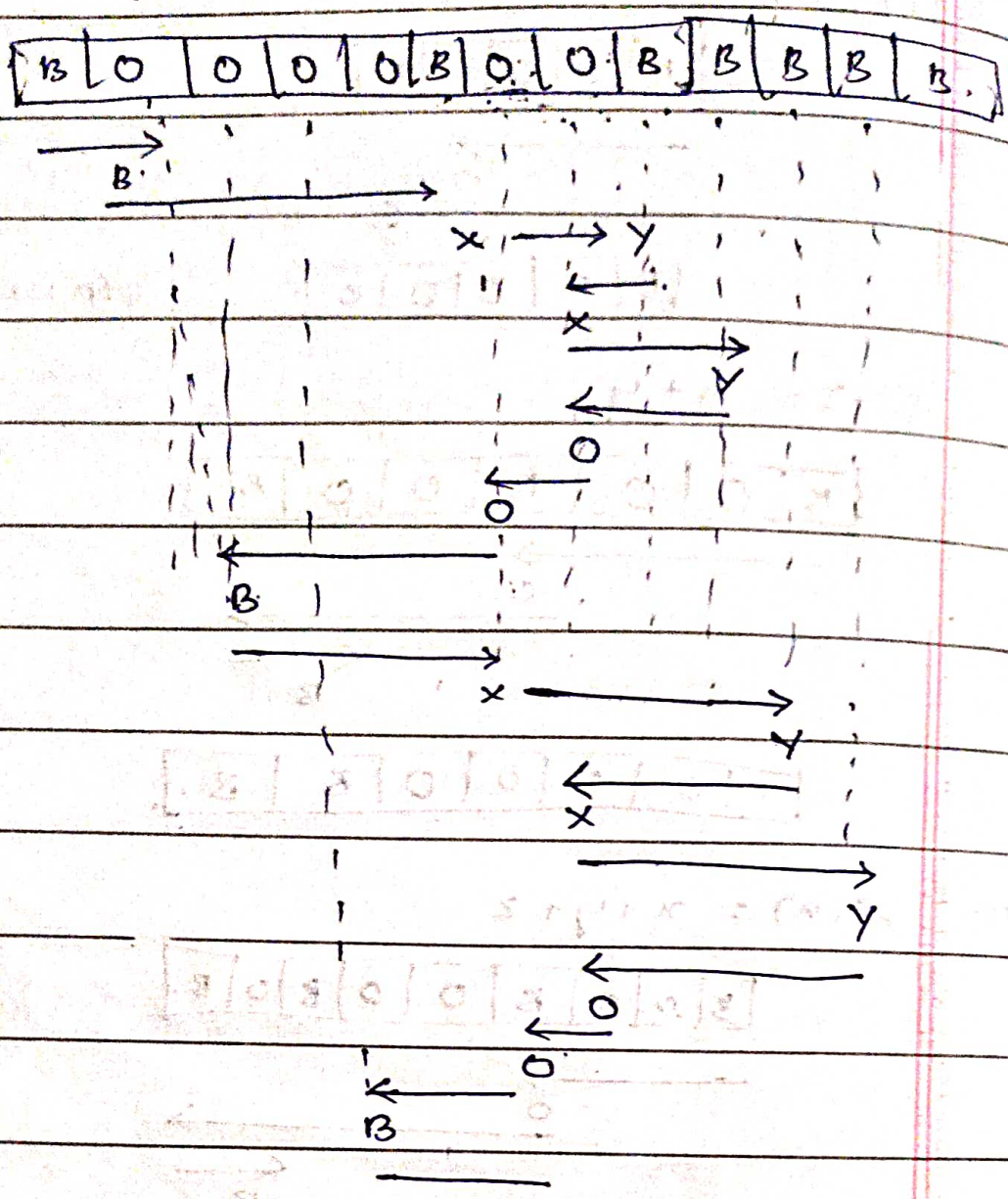




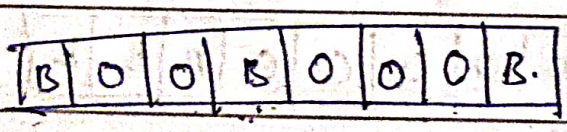
Ex1- $f(x) = x-y$.

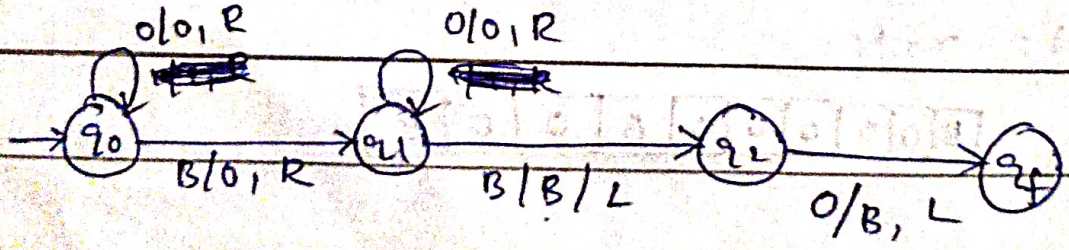


Ex1- $f(x) = x^* y$

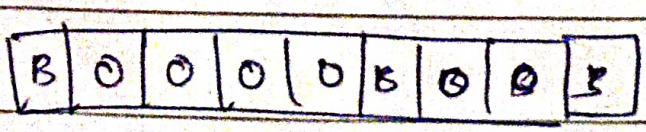


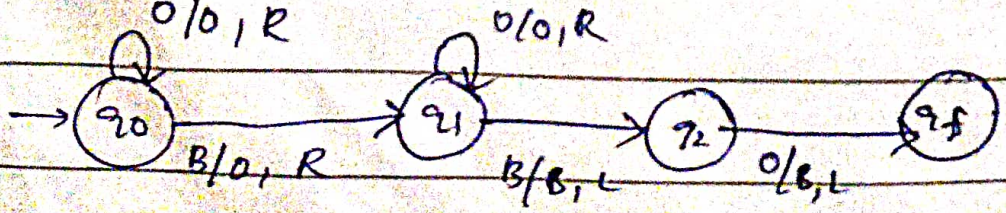
Ex2- $x+y$





Ex3- $x+2$





accepting strings should be accepted

(has halt) for non-accepting strings

* closure properties for Recursive languages:

→ union

→ intersection

→ compliment

→ Concatenate

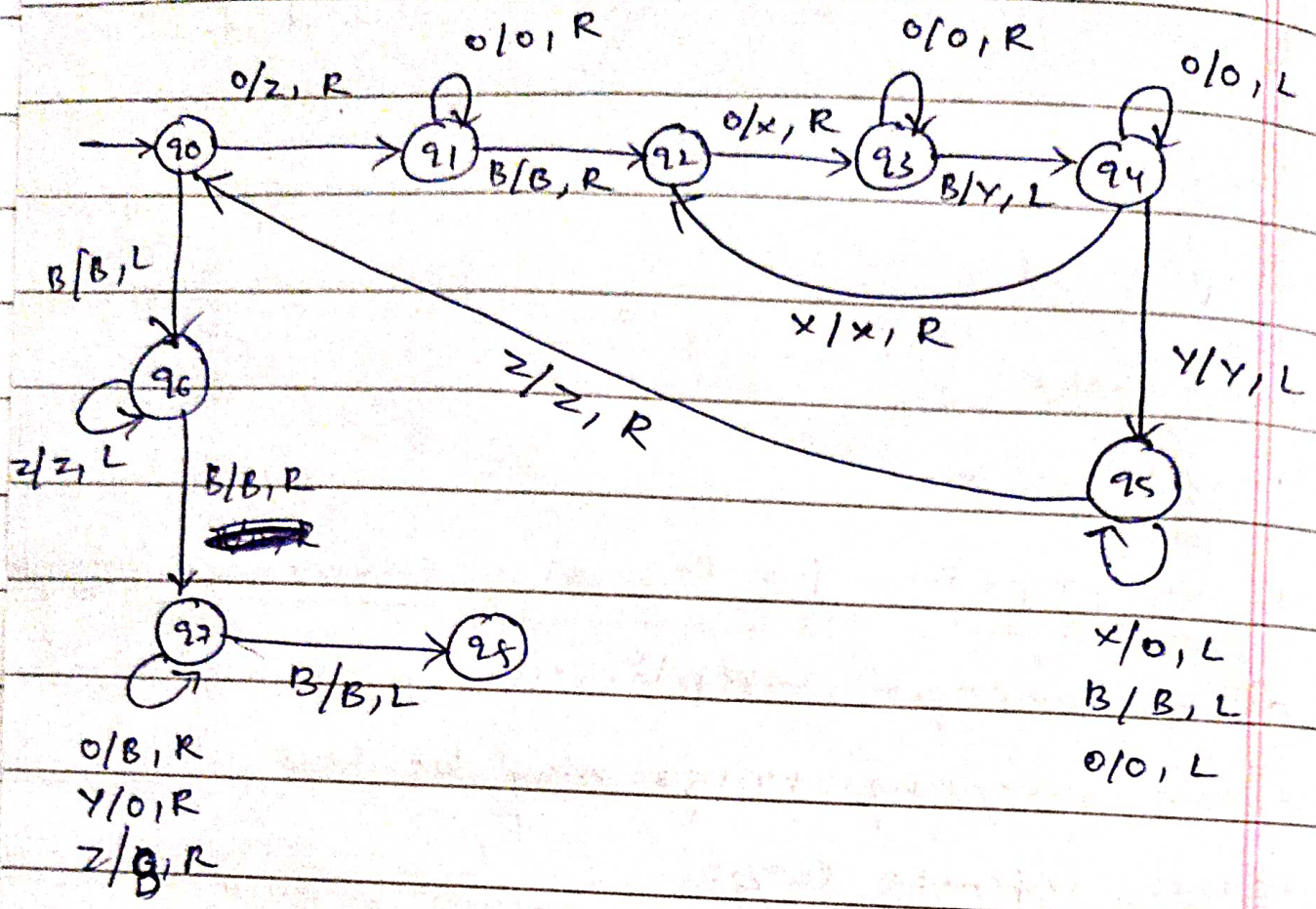
→ clean closure.

* closure properties for Recursive enumerable languages

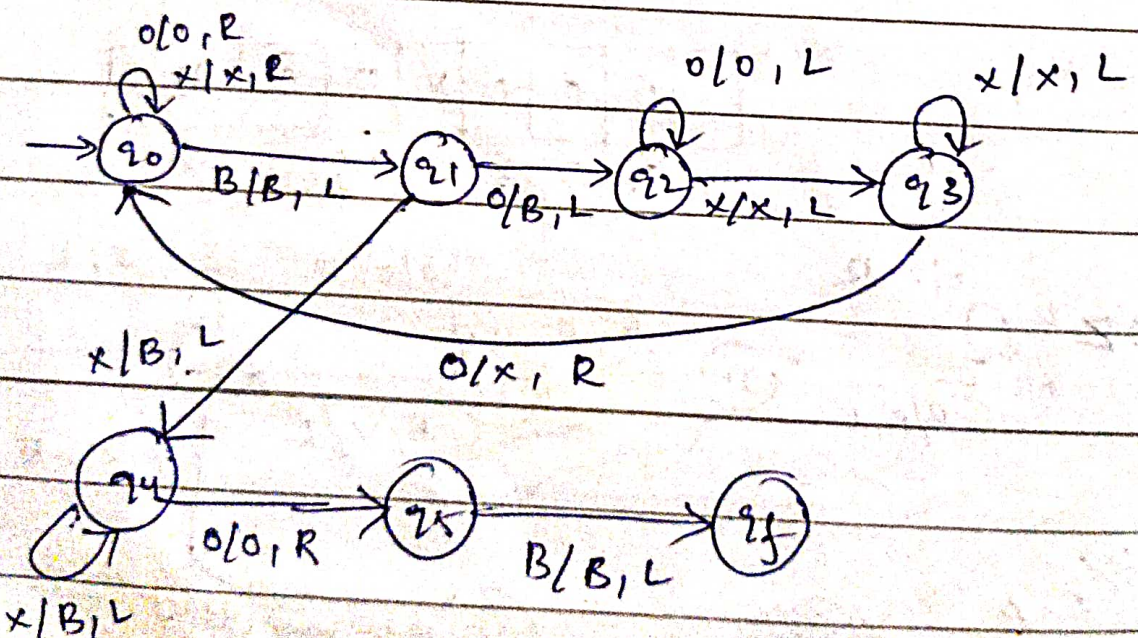
All above except compliment.

Here non-accepting strings may be halt or get into infinite loop.

Ex:- $x * y$.


0/0, R
0/Z, R
0/0, R
0/0, L
0/X, R
B/B, R
B/Y, L
B/B, L
X/X, R
Y/Y, L
Z/Z, R
Z/Z, L
B/B, R
X/0, L
B/B, L
0/0, L
B/B, L
0/B, R
Y/0, R
Z/B, R

Ex:- $x - y$.


0/0, R
X/X, R
0/0, L
X/X, L
B/B, L
0/B, L
X/X, L
0/X, R
X/B, L
0/0, R
B/B, L
X/B, L