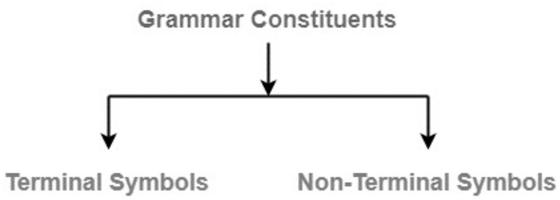
Grammar in theory of computation is a finite set of formal rules that are generating syntactically correct sentences.

The formal definition of grammar is that it is defined as four tuples –



G = (V,T,P,S)

- G is a grammar, which consists of a set of production rules. It is used to generate the strings of a language.
- V is the final set of non-terminal symbols. It is denoted by capital letters.
- T is the final set of terminal symbols. It is denoted by lower case letters.
- P is a set of production rules, which is used for replacing non-terminal symbols (on the left side of production) in a string with other terminals (on the right side of production).
- S is the start symbol used to derive the string.

Example

Grammar G1:

({S, A, B}, {a, b}, S, {S →AB, A →a, B →b})

Here,

S, **A**, and **B** are Non-terminal symbols; **a**

and **b** are Terminal symbols

S is the Start symbol, S ∈ N

Productions, **P : S →AB, A →a, B →b**

Example:

Grammar G2:

({S, A}, {a, b}, S,{S → aAb, aA →aaAb, A→ε })

Here,

S and **A** are Non-terminal symbols. **a**

and **b** are Terminal symbols.

ε is an empty string.

S is the Start symbol, S ∈ N

Production **P : S → aAb, aA →aaAb, A→ε**

Example

Production rules P = { S → ABa , A → BB , B → ab , AA → b }

V = { S , A , B } ⇒ Non-Terminal symbols
T = { a , b } ⇒ Terminal symbols
S = { S } ⇒ Start symbol

Example

Production rules P = { S→A1B, A→0A| ε, B→0B| 1B| ε }

V= {S, A, B} ⇒ non terminal symbols
T = { 0,1} ⇒ terminal symbols
S= {S} ⇒ start symbol.

If $G = (\{S\}, \{\ \}, \{S \to SS\}, S)$, find the language generated by G.

$L(G) = \emptyset$, since the only production $S \to SS$ in G has no terminal on the right-hand side.

Let $G = (\{S, C\}, \{a, b\}, P, S)$, where P consists of $S \to aCa$, $C \to aCa \mid b$. Find $L(G)$.

$S \to aCa \mid aCa$

$\quad\;\; b \qquad aaCaa$

$\quad\;\; \underline{aba} \qquad \underline{aabaa} \quad \ldots$

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

Let $G = (\{S, A_1, A_2\}, \{a, b\}, P, S)$, where P consists of

$$S \to aA_1A_2a,\ A_1 \to baA_1A_2b,\ A_2 \to A_1ab,\ aA_1 \to baa,\ bA_2b \to abab$$

Test whether $w = baabbabaaabbaba$

is in $L(G)$.

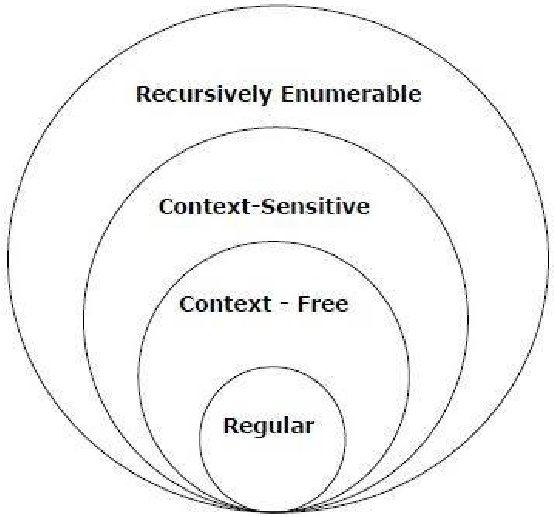
$$\begin{aligned}
S &\Rightarrow \underline{aA_1}\, A_2a \\
&\Rightarrow baa\, \underline{A_2}\, a \\
&\Rightarrow baa\, \underline{A_1}\, aba \\
&\Rightarrow baab\, \underline{aA_1}\, A_2baba \\
&\Rightarrow baabbaa\, \underline{A_2}\, baba \\
&\Rightarrow baabba\, \underline{aA_1}\, abbaba \\
&\Rightarrow \underline{baabbabaaabbaba} = w
\end{aligned}$$

of grammar

ent types of grammar –

ar	Language	Automata	Production rules
0	Recursively enumerable	Turing machine	No restriction
1	Context-sensitive	Linear-bounded non-deterministic machine	$\alpha A\beta \rightarrow \alpha\gamma\beta$
2	Context-free	Non-deterministic push down automata	$A \rightarrow \gamma$
3	Regular	Finite state automata	$A \rightarrow aB$ $A \rightarrow a$

CHOMSKY CLASSIFICATION OF LANGUAGES



Derivation

- L= $a^n b^n$ Where n>=1.

$$P = \left\{ \begin{array}{l} S \rightarrow aSB \\ S \rightarrow aB \\ B \rightarrow b \end{array} \right.$$

Derivation

$S \rightarrow aSB$

$\rightarrow aaSBB$

$\rightarrow aaaBBB$

$\rightarrow aaabBB$

$\rightarrow aaabbB$

$\rightarrow aaabbb$

G:

$S \to aSb \mid a$

Lan.

$(a^{n+1} b^{n})$ $n \geq 0$

$S \to 1S \mid 0A$

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

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

$((0+1)^* 00)$

1S

10A | 101

100

$L = \{aa,ab,ba,bb\}$

Regular Expression = (a+b)(a+b)

Grammar = S -> aa | ab | ba | bb or

S-> AA
A-> a/b

$L = \{a^n \mid n>=0 \}$

S -> aA / ϵ or

S -> Aa / ϵ

L = (a+b)*

S -> aS | bS| ϵ

Atleast two –

(a+b)(a+b)(a+b)*

S -> AAB
A-> a/b
B-> aB /bB/ ϵ

Atmost two –

(a + b + ϵ) (a + b + ϵ)

S -> AA
A-> a/b/ϵ

S. No.	Grammar	Rule		
1	Context Free Grammer (CFG)		A -> x where A ∈ V and x ∈ (V ∪ T) *	
2	Linear Grammar	(A linear grammar is a CFG that has **at most one non-terminal / variable** in the right hand side of each of its productions.(Having ε in the RHS also counts). 1. Right Linear 2. Left Linear		
	Right Linear		A → ε A → a A → aB	
	Left Linear		A → ε A → a A → Ba	
3.	Regular Grammar	A regular grammar is one that is either left-linear or right-linear.	A -> xB A -> x where A, B ∈ V and x ∈ T *	

- Example

- Now, consider the grammar G = ({S,A}, {a, b}, S, P) with productions:

- $S \to aA$
- $S \to Sb$
- $S \to \lambda$ ϵ

- This grammar is Context-free for sure. Since, LHS of the production consists of a variable and the RHS consists of $(V \cup T)^*$.

- This grammar is linear because it has at most one variable on the right.

- But if you observe closely the productions are the combination of both left-linear and right-linear grammar (1st and 2nd one respectively). So, it is not a regular grammar. It had to be either left-linear or right-linear.

$$
\left.
\begin{aligned}
S &\to aA \\
A &\to Bb \\
B &\to a
\end{aligned}
\right\}
$$

This is linear grammar but it consists of both left linear & right linear productions. Therefore, this grammar will not have an equivalent regular grammar.

Regular Grammar

The productions of the form

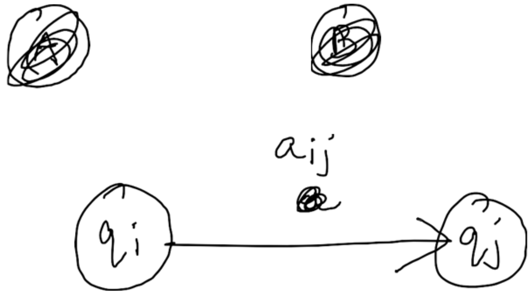
$$A \to xB \qquad \Big| \quad x \in \Sigma^* \quad \text{and} \quad A, B \in V$$
$$A \to x$$

will generate a regular language.

Alternatively, Regular Grammar will have productions of the form $A \to xB$ & $A \to x$ $\Big|$ $x \in \Sigma$ & $A, B \in V_N$

Note: $S \to \epsilon$ is allowed in RG but in that case S should not appear on the right side of any production.

FA to RG

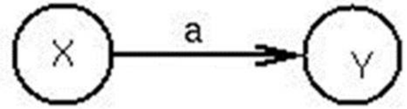


$A_i \rightarrow a_{ij} A_j$

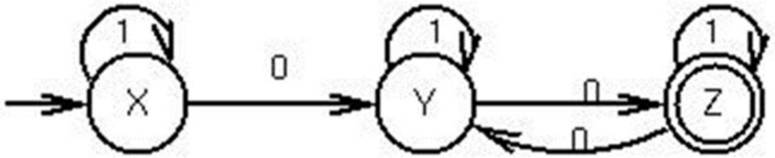
$A_i \rightarrow a_{ij} A_j$

$A_i \rightarrow a_{ij}$

Equivalence of FSA and regular grammars

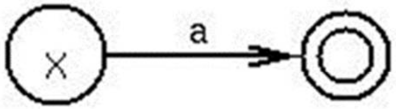


same as $X \longrightarrow a\,Y$



$X \longrightarrow 1\,X \mid 0\,Y$
$Y \longrightarrow 1\,Y \mid 0\,Z$
$Z \longrightarrow 0\,Y \mid 1\,Z$

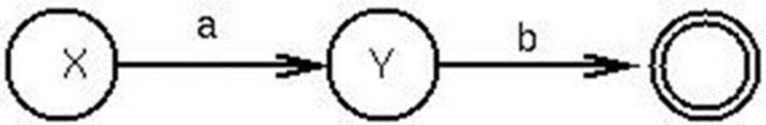
$Y \longrightarrow 0$
$Z \longrightarrow 1$



yields $X \longrightarrow a$

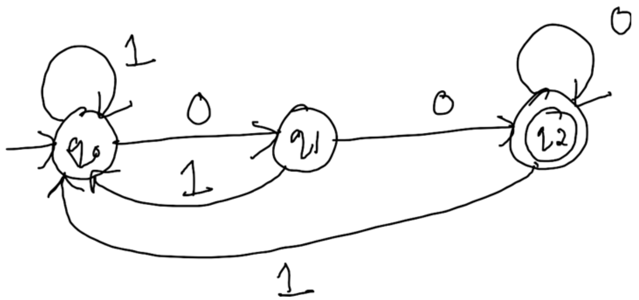
To go from regular grammar to FSA, make the following transformations:

$X \longrightarrow aY$
$Y \longrightarrow b$



Q11 Find a RG for the language that represents ~~a~~ all the binary strings ~~end~~ ~~ends~~ ending with 00.

Ans



Initial state represented by start symbol.

Let S, A, and B correspond to the states q_0, q_1, and q_2.

The Regular Grammar is as follows:

$S \rightarrow 1S$	$S \rightarrow 0A$	$B \rightarrow 0B$
$A \rightarrow 1S$	$A \rightarrow 0B$	$B \rightarrow 0$
$B \rightarrow 1S$	$A \rightarrow 0$	

Example

1000

$\therefore$ The final grammar is

$S \rightarrow 1S \mid 0A$

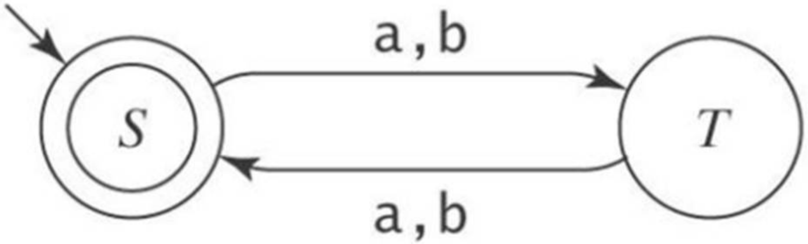
$A \rightarrow 1S \mid 0B \mid 0$

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

$S \rightarrow 1S$	$(S \rightarrow 1S)$
$\rightarrow 10A$	$(S \rightarrow 0A)$
$\rightarrow 100B$	$(A \rightarrow 0B)$
$\rightarrow 1000$	$(B \rightarrow 0)$

FA to G

$L = \{w \in \{a, b\}^* : |w| \text{ is even}\}$ $\quad ((aa) \cup (ab) \cup (ba) \cup (bb))^*$



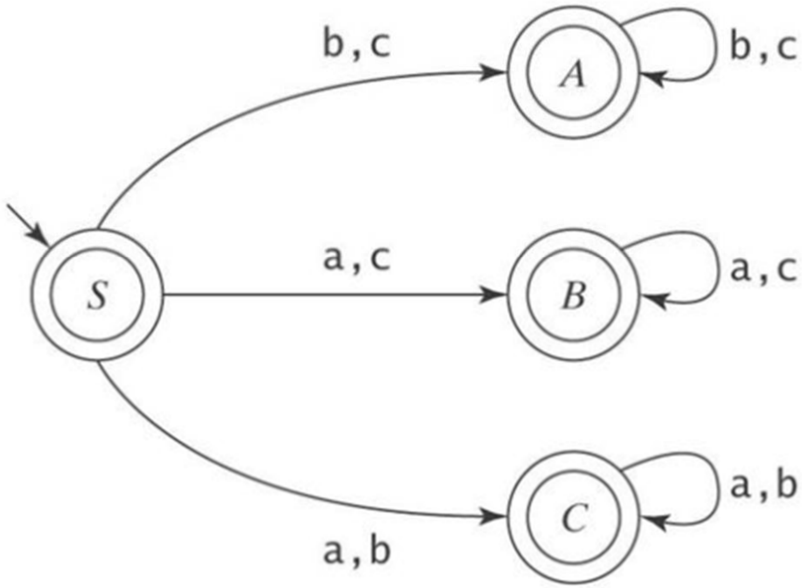
$S \rightarrow \varepsilon$
$S \rightarrow aT$
$S \rightarrow bT$
$T \rightarrow a$
$T \rightarrow b$
$T \rightarrow aS$
$T \rightarrow bS$

Gr. → FA

$S \to \varepsilon$
$S \to \mathtt{a}B$
$S \to \mathtt{a}C$
$S \to \mathtt{b}A$
$S \to \mathtt{b}C$
$S \to \mathtt{c}A$
$S \to \mathtt{c}B$

$A \to \mathtt{b}A$
$A \to \mathtt{c}A$
$A \to \varepsilon$
$B \to \mathtt{a}B$
$B \to \mathtt{c}B$
$B \to \varepsilon$

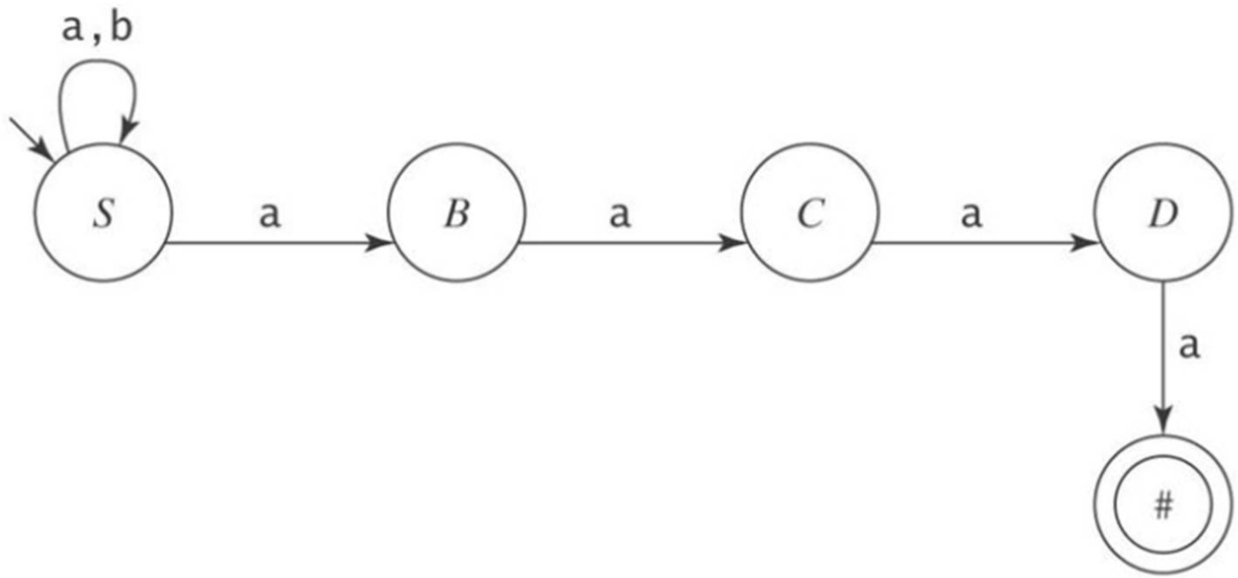
$C \to \mathtt{a}C$
$C \to \mathtt{b}C$
$C \to \varepsilon$



Strings that End with aaaa

$L = \{w \in \{\mathtt{a}, \mathtt{b}\}^* : w$ ends with the pattern $\mathtt{aaaa}\}$.

$$S \rightarrow \mathtt{a}S$$
$$S \rightarrow \mathtt{b}S$$
$$S \rightarrow \mathtt{a}B$$
$$B \rightarrow \mathtt{a}C$$
$$C \rightarrow \mathtt{a}D$$
$$D \rightarrow \mathtt{a}$$



Q// 2(b) Even length binary strings, where odd position contains 1

$(1(0+1))^*$ $(10+11)^*$

$1(0+1)^*$

$S \to 1A \mid \epsilon$

~~A → 0A~~

$A \to 0S \mid 1S \mid 0 \mid 1$

RLG

$S \to 10S \mid 11S \mid \epsilon$

LLG

$S \to S10 \mid S11 \mid \epsilon$

3(c)

$a(aa+bb)^*$

RLG

$S \to aA$

$A \to aaA \mid bbA \mid \epsilon$

OR

LLG

$S \to Saa \mid Sbb \mid a$

4(b)

$(111+11111)^*$

$G = (\{S\}, \{1\}, P, S)$

RLG

$S \to 111S \mid 11111S \mid \epsilon$

Right Linear Grammar	Left Linear Grammar
$A \to xB$ $A \to x$ $x \in \Sigma^*$ $A, B \in V_N$	$A \to Bx$ $A \to x$ $x \in \Sigma^*$ $A, B \in V_N$

$S \to abS \mid a$ $\qquad (ab)^+ a$

$\boxed{\begin{array}{l} S \to aA \mid a \\ A \to bS \end{array}}$

$S \to abS$
$\to ababS$
$\to abababS$
$\ldots$

$a^n b^n$

$aaa \ldots bbb$
$\vdots$

~~RG~~ Right Linear Gramm

RE

$\downarrow$

NFA / DFA

$\downarrow$ Remove dead states if any.

RG / RLG

Left linear grammar

RE

$\downarrow$

NFA / DFA

$\downarrow$

Reverse NFA/DFA

reverse the direction of edges
Initial state ~~...~~ becomes final state
& final state becomes initial state

$\downarrow$

RLG

$\downarrow$

Reverse the right ~~of~~ side of all the productions

$\downarrow$

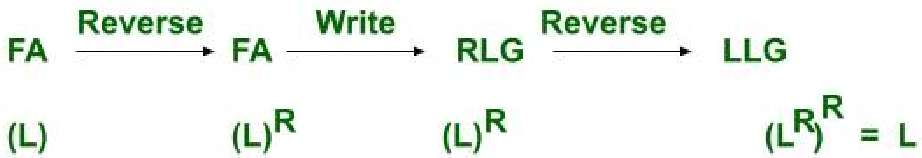
LLG

Rules

RE	RLG	LLG
a	$S \to a$	$S \to a$
$a+b$	$S \to a \mid b$	$S \to a \mid b$
ab	$\boxed{S \to aA,\ A \to b}$	$\boxed{S \to Ab,\ A \to a}$
a^*	$S \to aS \mid \epsilon$	$S \to Sa \mid \epsilon$
a^+	$S \to aS \mid a$	$S \to Sa \mid a$
$(a+b)^*$	$S \to aS \mid bS \mid \epsilon$	$S \to Sa \mid Sb \mid \epsilon$
$(a+b)^+$	$S \to aS \mid bS \mid a \mid b$	$S \to Sa \mid Sb \mid a \mid b$
$(ab)^*$	$S \to aA \mid \epsilon$ $A \to bS$	$S \to Ab \mid \epsilon$ $A \to Sa$
$(ab)^+$	$S \to aA$ $A \to bS \mid b$	$S \to Ab$ $A \to Sa \mid a$

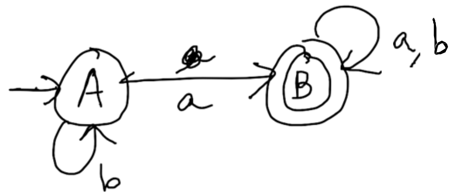
For converting the RLG into LLG for language L, the following procedure needs to be followed:

Step 1: Reverse the FA for language L
Step 2: Write the RLG for it.
Step 3: Reverse the right linear grammar.
after this we get the grammar that generates the language that represents the LLG for the same language L.

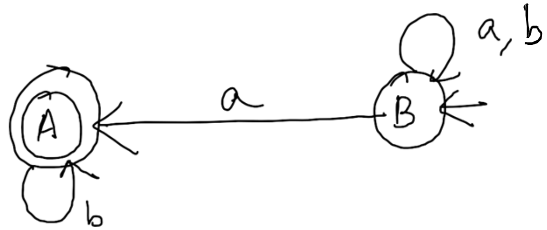


The above FA represents language L(i.e. set of all strings over input symbols a and b which start with b).
We are converting it into LLG.

Q/ Find LLG for the following ~~D~~FA .



Ans : Find the reverse of the given FA



Then find RLG

$B \to aB$	$A \to bA$
$B \to bB$	$A \to b$
$B \to aA$	
$B \to a$	

$\xrightarrow{\text{LLG}}$

$B \to Ba$

$B \to Bb$

$B \to Aa$

$B \to a$

$A \to Ab$

$A \to b$