4/3/13 UNIT-3 (Starting)

→ Let G $\{V, T, P, S\}$ be a CFG. A tree is a derivation tree (or) parse tree for G if

(i) Every vertex has a label, which is a symbol of "$V \cup T \cup \{\epsilon\}$".

(ii) The label of the root is 'S' (start symbol)

(iii) if a vertex is internal and has label A then A must be in V.

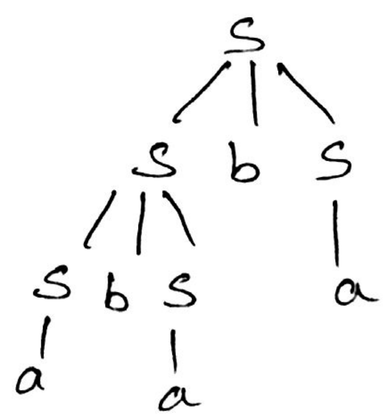
(iv) if a vertex 'n' has a label ϵ, then n is a leaf and is the only son of its father.

(v) if 'n' has label 'A' and vertices $n_1, n_2 \ldots n_k$ are the sons of vertex 'n', in order from the left with labels $x_1, x_2, \ldots x_a$ respectively, then $A \to x_1 x_2 x_3 \ldots x_a$ must be a production in P.

Q: If G is a grammer, $S \to SbS \mid a$ S.T. G is ambiguous.

Let $w = ababa$.

$S \to SbS$
$\to SbSbS$
$\to abSbS$
$\to ababS$
$\to ababa.$



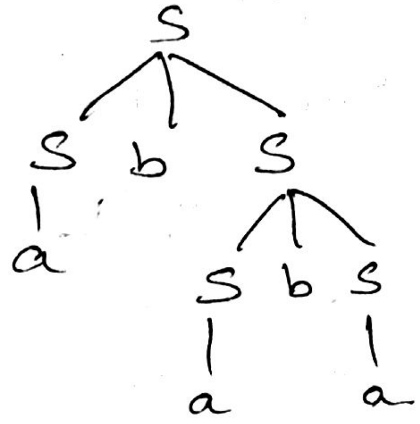
$S \rightarrow SbS$

$S \rightarrow abS$

$S \rightarrow abSbS$

$S \rightarrow ababS$

$S \rightarrow ababS$



∴ There exists two left most derivation trees for given string, then the given grammer is ambigous.

Q: Check whether the given grammer is ambigous or not.

$$E \rightarrow E+E \mid E*E \mid (E) \mid id.$$

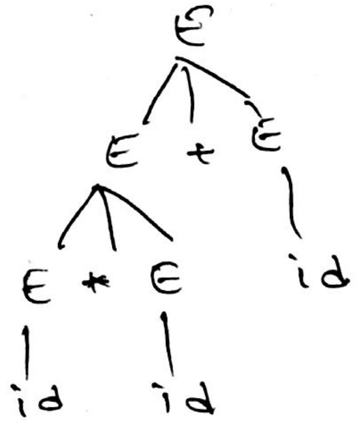
$E \rightarrow E+E$

$\rightarrow E*E+E$

$\rightarrow id*E+E$

$\rightarrow id*id+E$

$\rightarrow id*id+id.$



$E \rightarrow E*E$

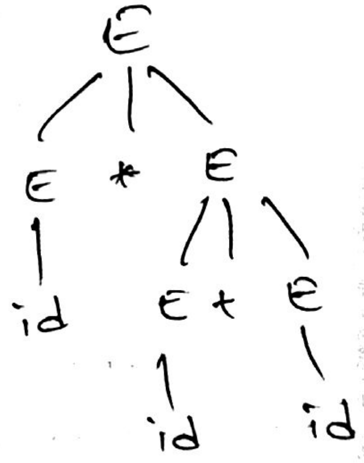
$\rightarrow id*E$

$\rightarrow id*E+E$

$\rightarrow id*E+E$

$\rightarrow id*id+E$

$\rightarrow id*id+id.$



Left most derivation:-

A derivation $S \overset{*}{\Rightarrow} w$ is called left most derivation if we apply the production rules only to the left most non-terminal at each step.

Ex:- $S \rightarrow aAS \mid a$

$A \rightarrow SbA \mid SS \mid ba$

derive the string aabbaa.

$V = \{S, A\}$

$T = \{a, b\}$

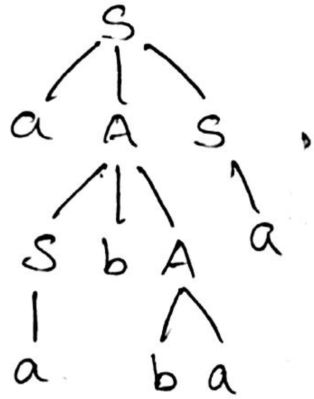
$S \rightarrow aAS$

$\rightarrow aSbAS$

$\rightarrow aabAS$

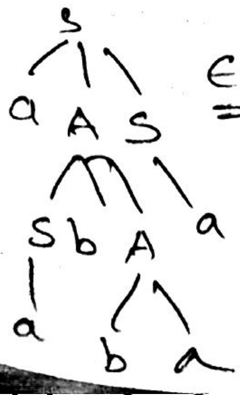
$\rightarrow aabbaS$

$\rightarrow aabbaa.$



Right most derivation:-

A derivation $S \overset{*}{\Rightarrow} w$ is called right most derivation if we apply the production rules only to the right most non-terminal at each step.



Ex:- derive aabbaa.

$S \rightarrow aAS$

$\rightarrow aAa$

$\rightarrow aSbAa$

$\rightarrow aSbbaa$

$\rightarrow aabbaa.$

Ambiguity in CFG:-

A terminal string $w \in L(G)$ is ambiguous if there exists two (or) more left most or right most derivations for w.

Ex:- $E \to E+E$
$/ E*E / (E) / id$

$w = id*id+id$

$E \to E+E$	$E \to E*E$
$\to E*E+E$	$\to id*E$
$\to id*E+E$	$\to id*E+E$
$\to id*id+E$	$\to id*id+E$
$\to id*id+id$	$\to id*id+id$

$\therefore$ It is ambiguos.

Context free language:-

The language generated by some grammar G, $L(G)$

$$L(G) = \{ w \mid w \in T^* \text{ and } S \overset{*}{\Rightarrow} w \}$$

→ A string is in $L(G)$ if

(i) The string consists of terminals only.

(ii) The strings can be derived from S.

→ A string of terminals and variables of 'α' is called a centinental form if $S \overset{*}{\Rightarrow} \alpha$.

Q: Construct CFG generating all integers (with sign).

A:

S → <sign><Integer>

<sign> → + | −

<Integer> → <digit><Integer> | <digit>

<digit> → 0|1|2|3|4|5|6|7|8|9.

Q: CFG for generating floating point numbers with sign.

A:

S → <sign><float>

<float> → <Integer>.<Integer>

<sign> → + | −

<Integer> → <digit><integer> | <digit>

<digit> → 0|1|2|3|4|5|6|7|8|9.

Q: CFG for $a^n b^n \mid n \geq 1$.

A:

$a^n b^n \mid n \geq 1$

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

$S \to ab \mid aSb.$

$\boxed{S = \epsilon \mid aSb \text{ when } n \geq 0}$

Ex:- $S \to aSb$, $w = aaabbb$

$\to aaSbb$

$\to aaabbb$

25/2/13

Regular Languages:-

→ A grammer is regular if it is either left linear (or) right linear.

Regular grammer from finite automata :-

Model-1:-

Construction of right linear for the given finite automata.

* Let the right linear grammer by $G = \{V, T, P, S\}$

where V – set of all non-terminals.

T – set of all terminals.

P – set of all productions.

$$A \to wB \mid w.$$

S – start symbol.

* To obtain, the productions of the grammer ie, P, we apply the following rules:

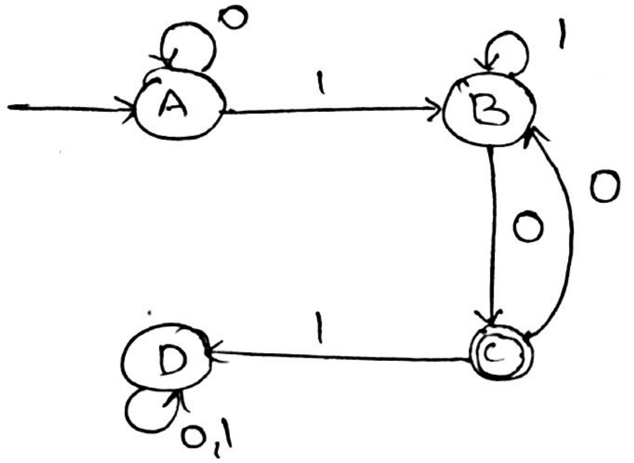
1. If there is a transition $\delta(q_i, a) \to q_j \notin F$, then include a production. $q_i \to a q_j$ (P)
2. If there is a transition $\delta(q_i, a) \to q_j \in F$, then include the productions

$$q_i \to a q_j \mid a.$$

3. An initial state of finite automata is the start symbol of G.
4. If the initial state ∈ final state, then add a production $S \to \epsilon$.

Ex:- Obtain the regular grammer from the following DFA.

PS	NS 0	NS 1
$\rightarrow$ A	A	B
B	C	B
(C)	B	D
D	D	D



1 $\delta(A,0) \rightarrow A$, 5 $\delta(A,1) \rightarrow B$

2 $\delta(B,0) \rightarrow C$, 6 $\delta(B,1) \rightarrow B$

3 $\delta(C,0) \rightarrow B$, 7 $\delta(C,1) \rightarrow D$

4 $\delta(D,0) \rightarrow D$, 8 $\delta(D,1) \rightarrow D$

$A \rightarrow 0A.$

$5 \Rightarrow A \rightarrow 1B$

$2 \Rightarrow B \rightarrow 0C \mid 0$

$6 \Rightarrow B \rightarrow 1B$

$3 \Rightarrow C \rightarrow 0B$

$7 \Rightarrow C \rightarrow 1D$

$4 \Rightarrow D \rightarrow 0D$

$8 \Rightarrow D \rightarrow 1D$

$\Longrightarrow$

$A \rightarrow 0A \mid 1B$

$B \rightarrow 0C \mid 0 \mid 1B$

$C \rightarrow 0B \mid 1D$

$D \rightarrow 0D \mid 1D$

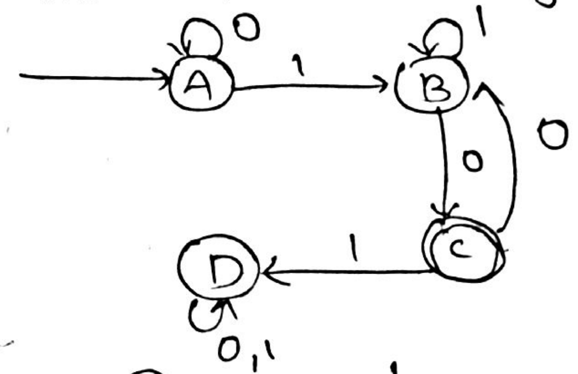
Construction of left linear grammer for given automata :-

* To constuct left linear grammer, first we need to construct a FA as follows:
  1. Reverse the direction of all edges.
  2. Make the initial state as final state and final state as initial state.

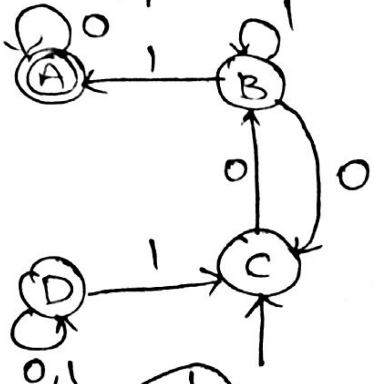
* After construction of new FA, find the right linear grammer for the new FA and then reverse the right side of all prodⁿ of the resulting right linear grammer.

Note - If there are more than 1 final state, then there is no left linear grammer.

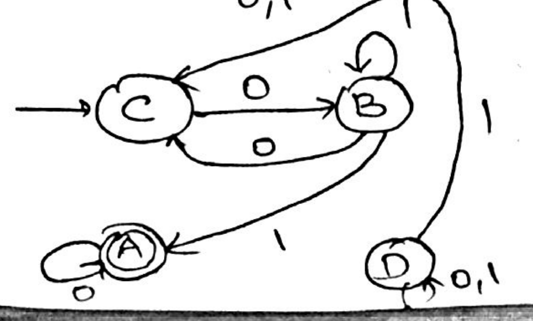
Given,



⇒



⇒



$\delta(C,0) \rightarrow B \notin F \Rightarrow C \rightarrow 0B$

$\delta(B,0) \rightarrow C \notin F \Rightarrow B \rightarrow 0C$

$\delta(B,1) \rightarrow B \notin F \Rightarrow B \rightarrow 1B$

$\delta(B,1) \rightarrow A \in F \Rightarrow B \rightarrow 1A \mid 1$

$\delta(A,0) \rightarrow A \in F \Rightarrow A \rightarrow 0A \mid 0$

$\delta(D,0) \rightarrow D \notin F \Rightarrow D \rightarrow 0D$

$\delta(D,1) \rightarrow C \notin F \Rightarrow D \rightarrow 1C$

$\delta(D,1) \rightarrow D \notin F \Rightarrow D \rightarrow 1D$

$\Rightarrow$ $A \rightarrow 0A \mid 0$

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

$C \rightarrow 0B$

$D \rightarrow 1C \mid 0D \mid 1D$

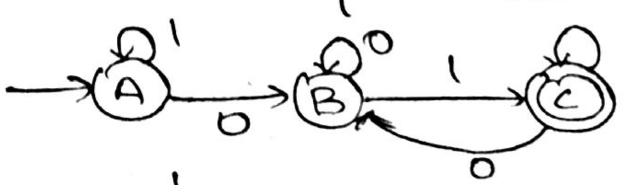
LLG $\Rightarrow$ $C \rightarrow B0$

$D \rightarrow C1 \mid D0 \mid D1$

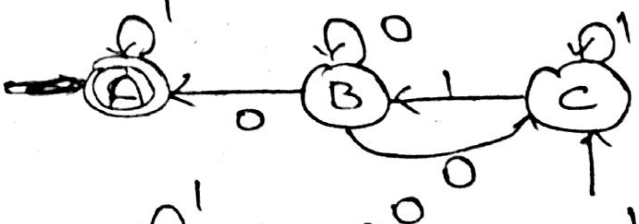
$B \rightarrow C0 \mid B1 \mid A1 \mid 1$

$A \rightarrow A0 \mid 0.$

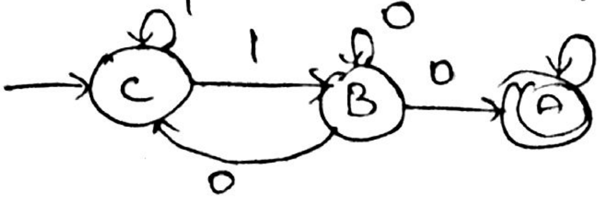
<u>Ex:-</u> Obtain a LLG for the following DFA.



$\Rightarrow$



$\Rightarrow$



$\delta(C,1) \rightarrow C \notin F \Rightarrow C \rightarrow 1C$

$\delta(C,1) \rightarrow B \notin F \Rightarrow C \rightarrow 1B$

$\delta(B,0) \rightarrow C \notin F \Rightarrow B \rightarrow 0C$

$\delta(B,0) \rightarrow B \notin F \Rightarrow B \rightarrow 0B$

$\delta(B,0) \rightarrow A \in F \Rightarrow B \rightarrow 0A \mid 0$

$\delta(A,1) \rightarrow A \in F \Rightarrow A \rightarrow 1A \mid 1$

$\Rightarrow \quad C \rightarrow 1C \mid 1B$

$B \rightarrow 0C \mid 0B \mid 0A \mid 0$

$A \rightarrow 1A \mid 1$

LLG :- $\Rightarrow \quad C \rightarrow C1 \mid B1$

$B \rightarrow A0 \mid B0 \mid C0 \mid 0$

$A \rightarrow A1 \mid 1$.

Generated by CamScanner from intsig.com

1/3/13

Conversion of right linear grammer to finite automata :-

1. If there is a production of the form $A_i \rightarrow aA_j$, then there is a transition of the form
$$\delta(A_i, a) \rightarrow A_j.$$

2. If there is a production of the form $A_i \rightarrow a$, then there is a transition of the form
$$\delta(A_i, a) \rightarrow A_k$$
A_k is final state. (newly introduced).

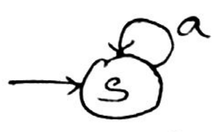
3. If $A \rightarrow \epsilon$, if A is not the start symbol, then that will be the final state.

4. If $A \rightarrow \epsilon$, and A is the start symbol, then that will be the start symbol and final state.

Q: Construct a FA recognizing the following regular grammer.
$$S \rightarrow aS \mid bA \mid b$$
$$A \rightarrow aA \mid bS \mid a$$

Sol:

$S \rightarrow aS \Rightarrow$

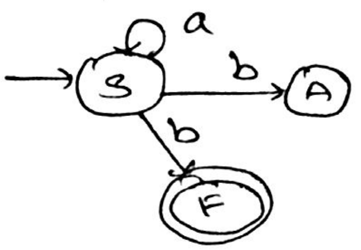


$S \rightarrow bA \Rightarrow$

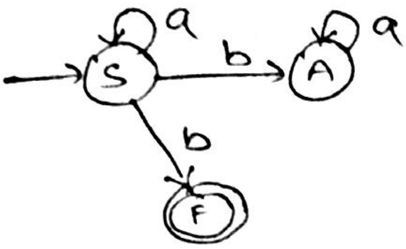
$S \rightarrow b \Rightarrow$

F is the new final state introduced.

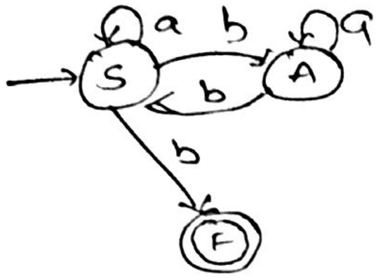
$\Rightarrow$



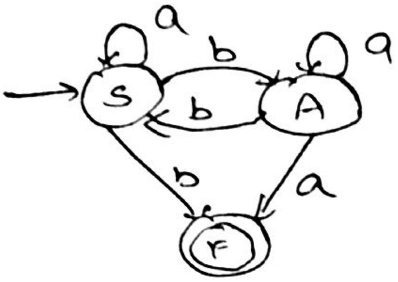
$A \rightarrow aA \Rightarrow$



$A \rightarrow bS \Rightarrow$



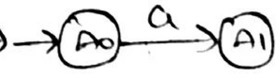
$A \rightarrow a \Rightarrow$



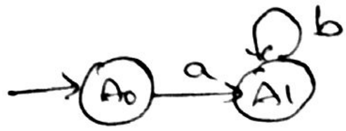
Q: FA recognizing the following regular grammar

$A_0 \rightarrow aA_1$
$A_1 \rightarrow bA_1 \mid bA_0 \mid a$

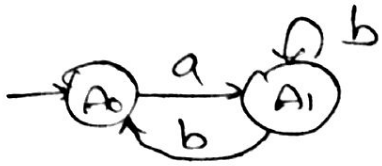
Sol:- $A_0 \rightarrow aA_1 \Rightarrow$



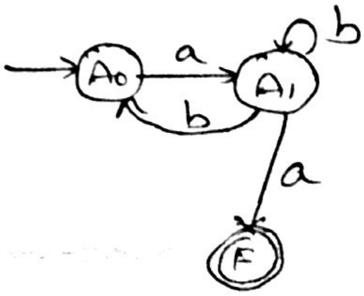
$A_1 \rightarrow bA_1 \Rightarrow$



$A_1 \rightarrow bA_0 \Rightarrow$



$A_1 \rightarrow a \Rightarrow$



Construction of FA from left linear grammer:-

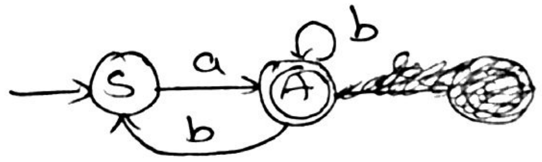
1. Write the right linear grammer by reversing the right hand sides of all productions.
2. Construct FA for the right linear grammer
3. Reverse the edges of FA and interchange the initial and final states. We get the new FA that is the required FA.

Q: FA for the following regular grammer.

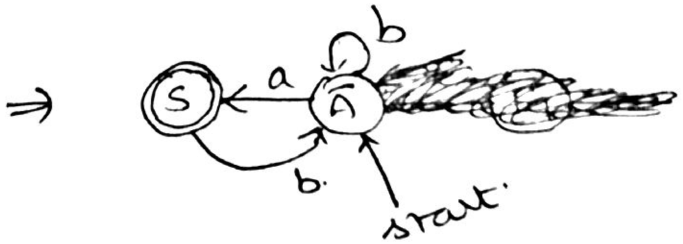
$$S \rightarrow Aa$$
$$A \rightarrow Sb \mid Ab \mid \epsilon$$

Sol:

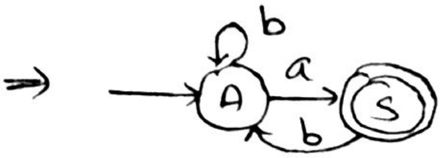
$$S \rightarrow aA$$
$$A \rightarrow bS \mid bA \mid \epsilon$$



$\Rightarrow$



$\Rightarrow$



Conversion of a regular grammer to NFA-ϵ:-

1. Let $L = L(G)$ for some right linear grammer (right linear) $G = \{V, T, P, S\}$. We construct an NFA-ϵ, $M = \{Q, T, \delta, [S], \{[\epsilon]\}\}$ that simulates derivations in G.

Q: Construct Right linear & left linear grammer for the following regular expression

$$0^*.(1(0+1))^*. \rightarrow RL$$

$$\leftarrow$$

$$((1+0)1)^*.0^* \rightarrow LL$$