13/8/25

# AT Tut-2

A grammar has –

i) Variables
ii) Terminals
iii) Production
iv) Start

## Converting RLG ⟶ Finite Automata

- $Q_0$ is the start var of grammar and also initial state of the DFA.

- For rules of the form $Q_i \longrightarrow w Q_j$ the corresponding transition defined will be of the form

$$\delta^*(Q_i, w) \longrightarrow Q_j$$

- For each rule of the form
  $q_i \longrightarrow w$ the transition will be
  $\delta^*(Q_i, w) \longrightarrow Q_f$ ⟶ final state.

**Example:** Take the following grammar.

$$S \longrightarrow aA \mid \epsilon$$
$$A \longrightarrow aA \mid bB \mid \epsilon$$
$$B \longrightarrow bB \mid \epsilon$$

$S \longrightarrow aA \Rightarrow (S, a) \longrightarrow A$

$S \longrightarrow \epsilon \Rightarrow (S, \epsilon) \longrightarrow Q_f$

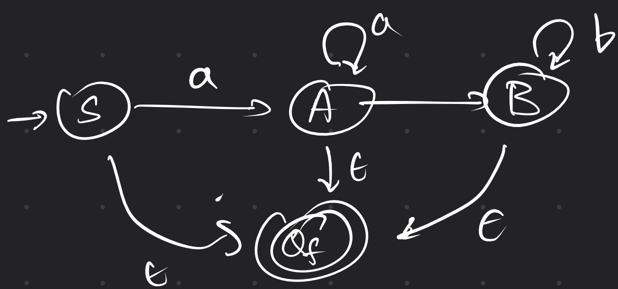
$A \longrightarrow aA \Rightarrow (A, a) \longrightarrow A$.

$A \longrightarrow bB \Rightarrow (A, b) \longrightarrow B$

$A \longrightarrow \epsilon \Rightarrow (A, \epsilon) \longrightarrow Q_f$

$B \longrightarrow bB \Rightarrow (B, b) \rightarrow B$

$B \rightarrow \epsilon \Rightarrow (B, \epsilon) \rightarrow B$



## Converting Finite Automata to RLG:

DFA $\Rightarrow$ RLG

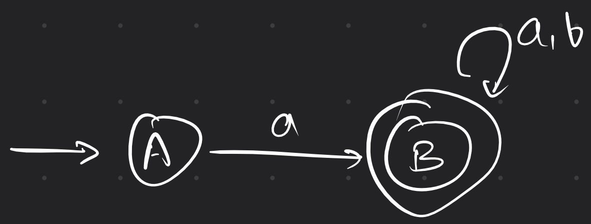
- $\delta(Q_i, w) \to Q_j$ is captured by $Q_i \to w Q_j$

- For final states of the DFA, just add rule $Q_f \to \epsilon$

We can then eliminate the $\varepsilon$ production.

RLG $\Rightarrow$ Reg.

**Example:**




$$A \to aB$$
$$B \to aB \mid bB \mid \epsilon$$

$$\equiv \quad A \to aB \mid a$$
$$B \to aB \mid bB \mid a \mid b.$$

| |
|---|
| **Try it later:** R Linear Grammar $\equiv$ Left Linear Grammar<br>this means showing RLG $\Rightarrow$ LLG and<br>LLG $\Rightarrow$ RLG |

**Definition:** Linear Grammar:

Any CFG that has at most one non-terminal in the RHS of each production is called linear grammar.

If the variable is at the right most place, then it is right linear grammar.

Showing RLG = Regular Languages

Reg $\Longrightarrow$ RLG

RLG $\Longrightarrow$ DFA

} If both are shown

Reg $\equiv$ RLG.

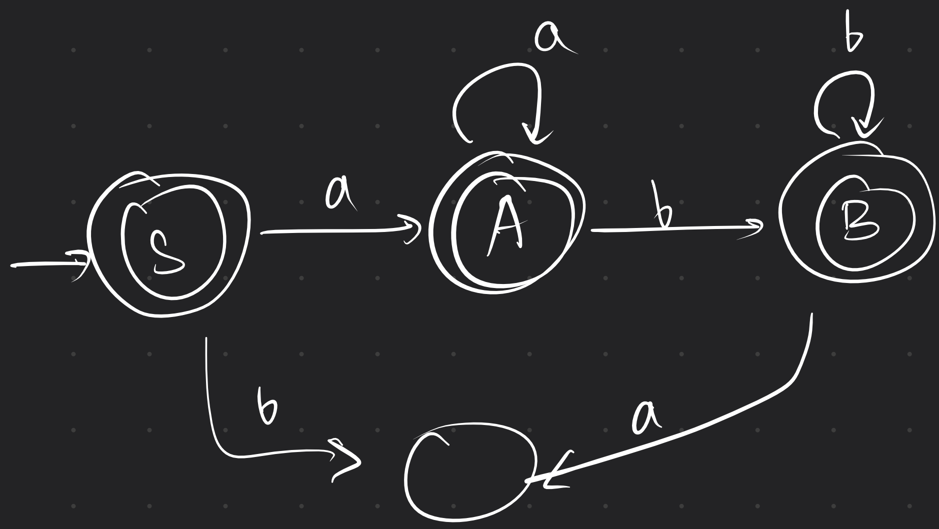
Variables, Terminals, Production, Start

$A \to aA \mid \epsilon$

$A \to Aa$

$A \to AB$

Linear Grammar.




A DFA should not have missing transitions!

# Pumping lemma

If $A$ is regular, $\exists\, p$ s.t.

$s \in A$, $|s| \geq p$. $s = xyz$ s.t.

(i) $|xy| \leq p$

(ii) $|y| > 0$

(iii) $xy^iz \in A \quad \forall\, i \geq 0$

$\rightarrow \quad L = \{ 0^n 1^n \mid n \geq 0 \}$

$n' \rightarrow$ pumping length.

$s = 0^{2n'} 1^{2n'} \in L$

$= xyz$

$$\underbrace{\boxed{\underbrace{00\cdots0}_{n'}}}_{xy}\,\underbrace{0\cdots0}_{n'}\,\underbrace{1\cdots}_{2n'}$$

$xy^2z$

$\rightarrow \quad L = \{ ww \mid w \in \{0,1\}^* \}$

$p \rightarrow$ Pumping length

$s = 0^p 0^p$

$xyz$

$|xy| \le p$

$\boxed{y = 0^{n'}}$

$x = 0^a \quad a \geq 0$

$xy^i z = 0^a 0^{in'} 0^{p + (p - a - n')}$

$\notin L$

$s = 01^p 01^p$

$|xy| \le p$

$xy = 0 \mid x = \varepsilon$

$y = 0$

$xy^i z = 0^i 1^p 01^p$

$\notin L \quad \forall i \neq 1$

$xy = 01^{n'} \mid x = \varepsilon \mid x$

$\begin{cases} (01^{n'})^i \, 1^{p-n'} \, 01^p \\ 01^{n''} (1^{n'''})^i \, 1^{p-n''-n'''} \, 01^p \end{cases}$

$\rightarrow \quad L = \{ 0^i 1^j \mid i > j \}$

Consider the pumping length to be p.

For the string $s = 0^p 1^{p-1}$

Since we know $|xy| < p$,

this means $xy$ part of the string contains only 0s

$\Rightarrow$ $y$ contains only 0s.

but since we also know $|y| > 0$, this means

$x = 0^{l}$ $\quad y = 0^{m}$ $\quad z = 0^{p-l-m} 1^{p-1}$

$(m > 0)$

Now, we construct the string

$$w' = x y^{i} z$$

$$= 0^{l} \, 0^{mi} \, 0^{p-l-m} \, 1^{p-1}$$

$$= 0^{p+m(i-1)} \, 1^{p-1} \qquad \{\text{Note} : m > 0\}$$

If we now choose $i=0$, i.e. we never take the loop $y$, then

$w' = 0^{p-m} 1^{p-1}$ Pumping Down.

Since $m > 0 \Rightarrow p-m \leq p-1$

$\therefore$ $w' \notin L$ since number 0s is not more than the number of 1s.

$\therefore$ Using pumping lemma, we have shown that there does not exist a DFA which decides the language.

$\therefore$ The language is NOT regular.