CSC-257
Theory Of Computation
(BSc CSIT, TU)

Ganesh Khatri
kh6ganesh@gmail.com

Left and Right Linear Grammars

- A grammar G = (V, T, P, S) is said to be left linear grammar(LLG) if all the productions are of the form

  $A \rightarrow Bx \mid x$

- And it is said to be right linear grammar(RLG) if all the productions are of the form

  $A \rightarrow xB \mid x$

  where, $A, B \in V$ and $x \in T^*$

- It means, RHS of LLG productions start with non-terminals followed by zero or more numbers of terminals and

  RHS of RLG productions start with zero or more numbers of terminals followed by a non-terminal

Regular Grammar

- A regular grammar is a CFG which may be either :
  - Left Linear
  - Right Linear
- A language generated by regular grammar is called Regular Language.
- A regular grammar represents a language that is accepted by some finite automata called regular language
- Note : All the CFGs are note regular i.e. all the CFGs can not be converted into left linear or right linear.

Equivalence of Regular Grammar and Finite Automata

- **A right linear grammar** can be converted into equivalent finite automata using the following rules :
  - Note : Number of states of finite automata will be one more than number of non-terminals in regular grammar and, each non-terminal represents a state in finite automata. Such one additional state is the final state.
  - **Transitions of equivalent automata :**

1. The start symbol of regular grammar is the start state of the finite automaton.
2. For every production of the form A → aB, make δ(A, a) = B : make an edge labelled with 'a' from A to B.
3. For every production of the form A → a, make δ(A, a) = Final state
4. For every production of the form A → ε, make δ(A, ε) = Final state

Equivalence of Regular Grammar and Finite Automata

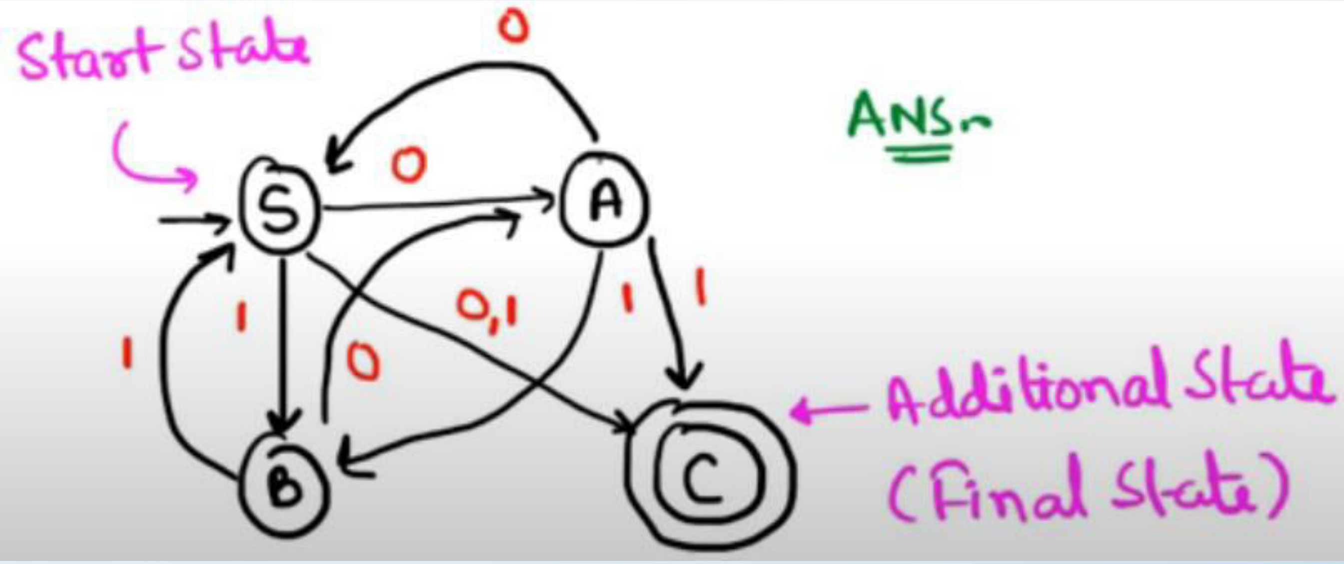
- **Example :** Convert the following RLG into equivalent finite automata.

  S → 0A | 1B | 0 | 1

  A → 0S | 1B | 1

  B → 0A | 1S

- **Solution :** Following the steps, we get the finite automata as :



Equivalence of Regular Grammar and Finite Automata

- **Exercise :** Convert the following right linear grammars into finite automata

1.

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

$A \rightarrow 0A \mid 1A \mid 0 \mid 1$

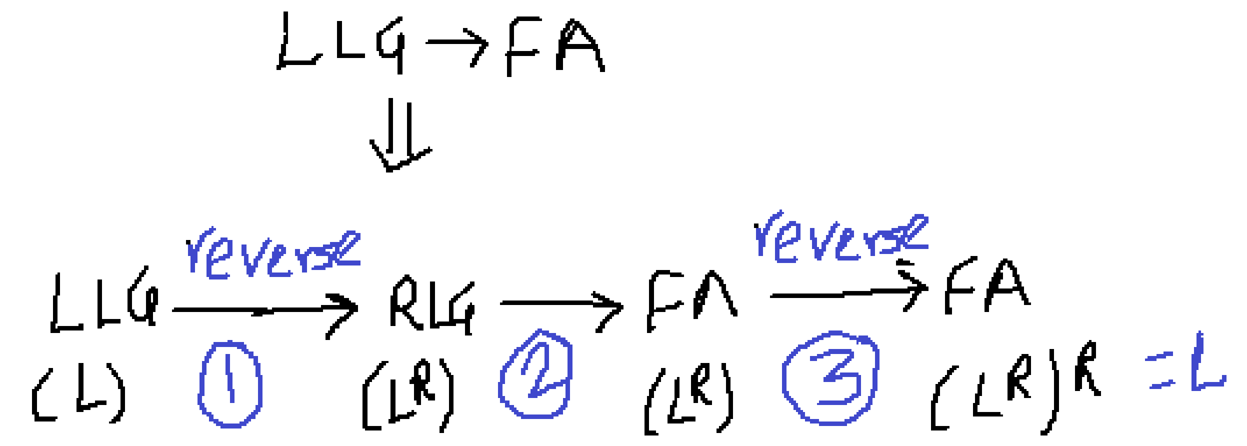
2.

$S \rightarrow aaA \mid B \mid baB \mid \varepsilon$

$A \rightarrow bS \mid aS \mid b$

Equivalence of Regular Grammar and Finite Automata

- **A left linear grammar** can be converted into equivalent finite automata using the following rules :
  1. Reverse the left linear(LLG) grammar to right linear grammar(RLG). (**Note :** reverse the combinations of terminals and non terminals)
  2. Convert RLG obtained in step 1 into finite automata.
  3. Reverse finite automata obtained in step. [**Note :** exchange start and final states and reverse the directions of the transitions]. This will be our required finite automata for LLG.
- **Note :** The regular grammar obtained from finite automata of step 3 will be RLG and is equivalent to original LLG.



Equivalence of Regular Grammar and Finite Automata

- **Example :** Convert the following LLG into equivalent finite automata.

  S → Ca | Aa | Bb

  A → Ab | Ca | Bb | a

  B → Bb | b

  C → Aa

- **Solution :**

- Step 1 : Reversing the LLG, we get

  S → aC | aA | bB

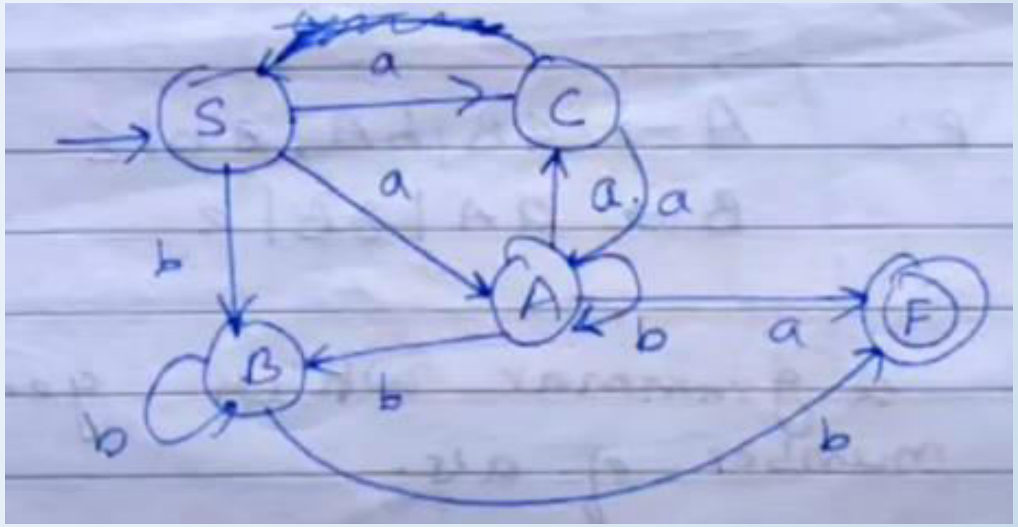
  A → bA | aC | bB | a

  B → bB | b

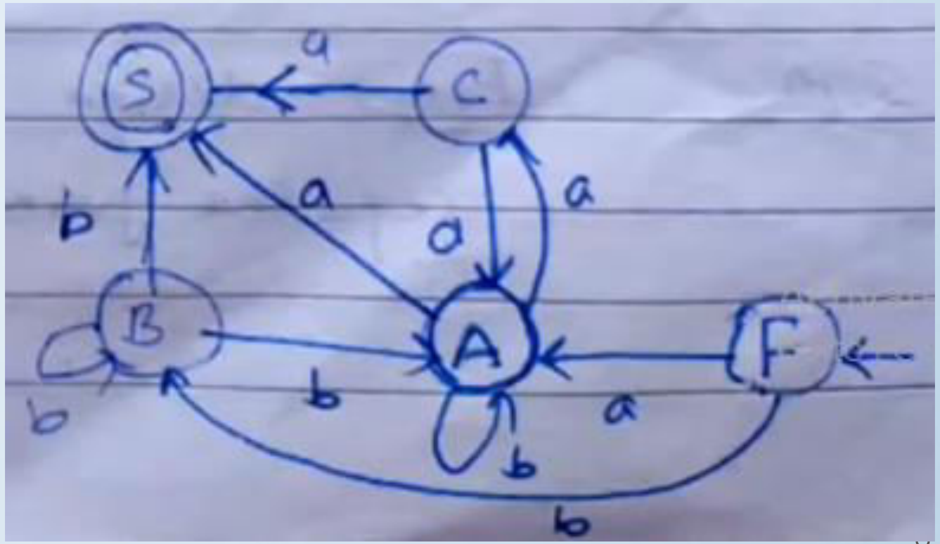
  C → aA

Equivalence of Regular Grammar and Finite Automata

- Step 2 : Now, convert the grammar into FA, we get



- Step 3 : Now, reversing the FA, we get



- Now, equivalent RLG will be

  F → aA | bB

  A → bA | aC | a

  B → bB | bA | b

  C → aA | a

Equivalence of Regular Grammar and Finite Automata

- **Exercise :** Convert the following left linear grammars into finite automata
  1. $S \rightarrow Aab$

     $A \rightarrow Aab \mid B$

     $B \rightarrow a$