Regular Grammar to FA Conversion

Bachelor of Technology
Computer Science and Engineering

Submitted By

ARKAPRATIM GHOSH (13000121058)

MARCH 2023

Techno Main
EM-4/1, Sector-V, Salt Lake
Kolkata- 700091
West Bengal
India

TABLE OF CONTENTS

1. Introduction:

Regular grammar generates regular language. They have a single non-terminal on the left hand side and a right-hand side consisting of a single terminal or single terminal followed by a non-terminal.

A → xB

A → x

A → Bx

where A, B ∈ Variable(V) and x ∈ T* i.e. string of terminals.

Types of regular grammar:

- **Left Linear grammar(LLG)**
- **Right linear grammar(RLG)**

1. Left linear grammar(LLG):

In LLG, the productions are in the form if all the productions are of the form

A → Bx

A → x

where A,B ∈ V and x ∈ T*

2. Right linear grammar(RLG):

In RLG, the productions are in the form if all the productions are of the form

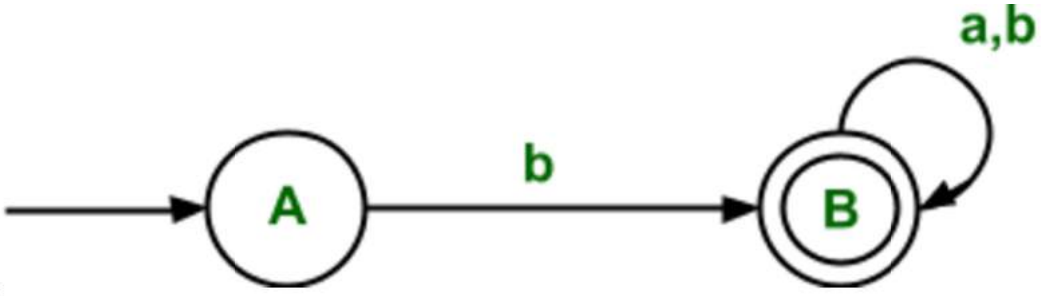
A → xB

A → x

where A,B ∈ V and x ∈ T*

The language generated by type-3 grammar is a regular language, for which a FA can be designed. The FA can also be converted into type-3 grammar

Example: FA for accepting strings that start with b



∑ = {a,b}

Initial state(q0) = A

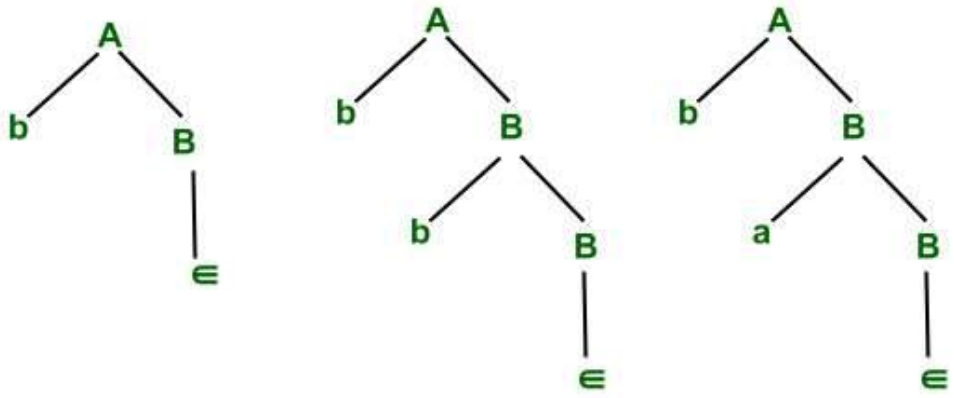
Final state(F) = B

The RLG corresponding to FA is

A → bB

B → ∈/aB/bB

The above grammar is RLG, which can be written directly through FA.



The above RLG can derive strings that start with b and after that any input symbol(i.e. ∑ ={a, b} can be accepted).

The regular language corresponding to RLG is

L= {b, ba, bb, baa, bab ,bba,bbb ..... }

If we reverse the above production of the above RLG, then we get

A → Bb

B → ∈/Ba/Bb

It derives the language that contains all the strings which end with b.

i.e. L' = {b, bb, ab, aab, bab, abb, bbb .....}

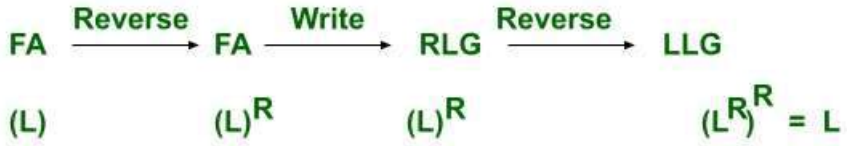
So we can conclude that if we have FA that represents language L and if we convert it, into RLG, which again represents language L, but after reversing RLG we get LLG which represents language L'(i.e. reverse of L).For converting the RLG into LLG for language L, the following procedure needs to be followed:

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.



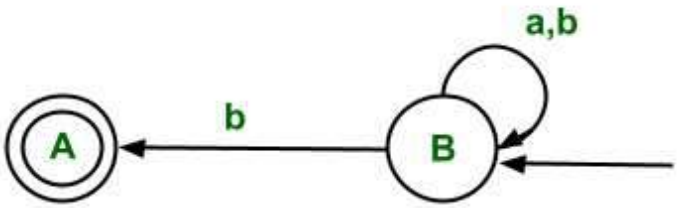
Here L is a language for FA and LR is a reversal of the language L.

Example:

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.

Step1:The reversal of FA is



Step 2:The corresponding RLG for this reversed FA is
B → aB/bB/bA
A → ∈
Step 3:The reversing the above RLG we get
B → Ba/Bb/Ab
A → ∈
So this is LLG for language L(which represents all strings that start with b). L= {b, ba, bb, baa, bab ,bba, bbb ….. }

2. Body

Conversion of RLG to FA:

•Start from the first production.

•From every left alphabet (or variable) go to the symbol followed by it.

•Start state: It will be the first production state.

•Final state: Take those states which end up with terminals without further non-terminals.

Example: The RLL grammar for Language(L), represents a set of all strings which end with 0.
A → 0A/1B/0B
B → ∈
So the FA for corresponding to RLG can be found out as Start with variable A and use its production.

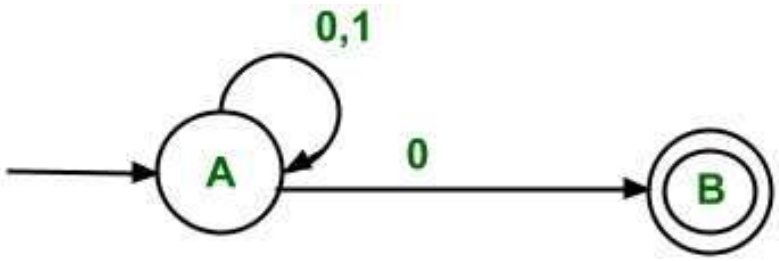
•For production A → 0A, this means after getting input symbol 0, the transition will remain in the same state.

•For production, A → 1B, this means after getting input symbol 1, the state transition will take place from State A to B.

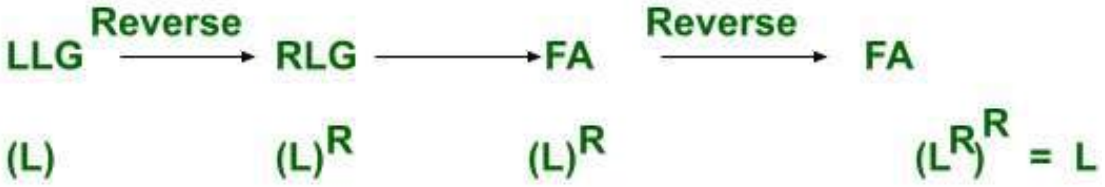
•For production A → 0B, this means after getting input symbol 0, the state transition will take place from State A to B.

•For production B → ∈, this means there is no need for state transition. This means it would be the final state in the corresponding FA as RHS is terminal.

So the final NFA for the corresponding RLG is

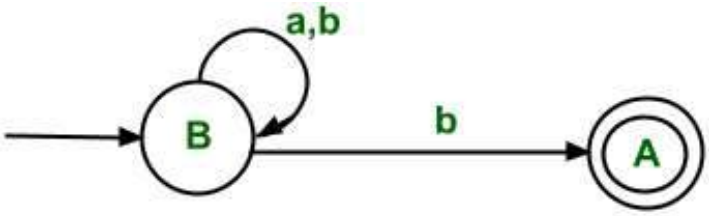


Conversion of LLG to FA:

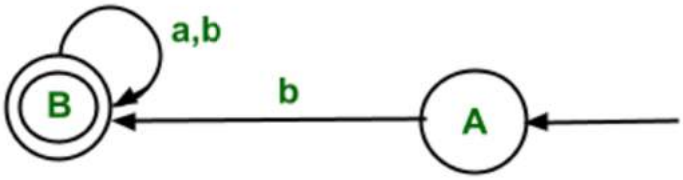


Explanation: First convert the LLG which represents the Language(L) to RLG, which represents, the reversal of language L(i.e.LR) then design FA corresponding to it(i.e. FA for Language LR). Then reverse the FA. Then the final FA is FA for language L).

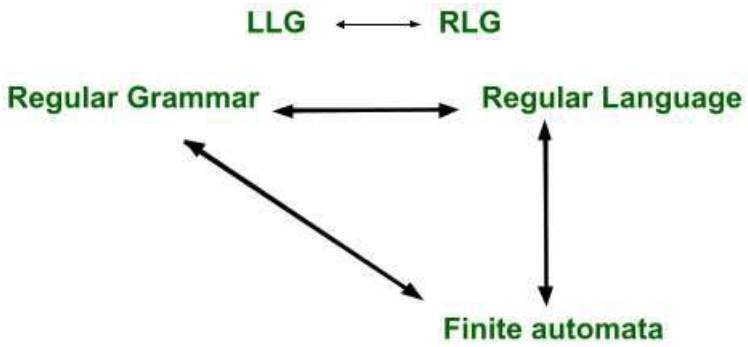
Conversion of LLG to RLG: For example, the above grammar is taken which represents language L(i.e. set of all strings that start with b)
The LLG for this grammar is
B → Ba/Bb/Ab
A → ∈
Step 1: Convert the LLG into FA(i.e. the conversion procedure is the same as above)



Step 2: Reverse the FA(i.e. initial state is converted into final state and convert final state to initial state and reverse all edges)



Step 2: Reverse
the FA(i.e. initial state is converted into final state and convert final state to initial state and reverse all edges)



3. Conclusion

The regular languages can be generated by regular grammar. In regular grammar, the left-hand side always consists of a single non-terminal. The left side cannot have more than one non-terminal or any terminal variable. There can be a single terminal or a single terminal followed by a non-terminal on the right-hand side.

4. References

a. Theory of Computation (With Formal Languages) ~ R.B. Patel, Prem Nath
b. GeeksforGeeks
c. Javatpoint