AUTOMATA FORMAL LANGUAGES AND LOGIC

Lecture notes on equivalence of Regular Grammar & Finite Automata

Prepared by:

Prof.Sangeeta V I

Assistant Professor

Department of Computer Science & Engineering

PES UNIVERSITY

(Established under Karnataka Act No.16 of 2013)

100-ft Ring Road, BSK III Stage, Bangalore – 560 085

Table of contents

Examples Solved:

Examples Solved:

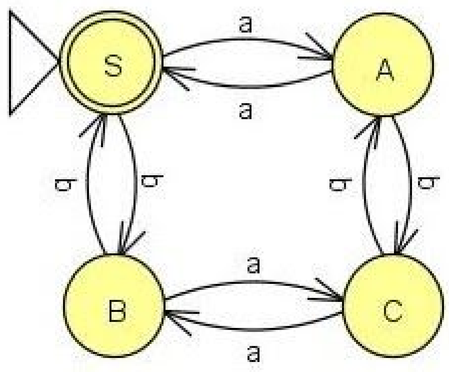
#	Converting Regular grammar to finite automata.	Page number
1	**Convert given regular grammar to finite automata.** $S \rightarrow bS \mid aA$ $A \rightarrow aA \mid bA \mid aB$ $B \rightarrow bbB \mid \lambda$	7
2	**Convert given regular grammar to finite automata.** $S \rightarrow 01A$ $A \rightarrow 10B$ $B \rightarrow 0A \mid 11$	8
3	**Convert given regular grammar to finite automata.** $A \rightarrow aB \mid bA \mid b$ $B \rightarrow aC \mid bB$ $C \rightarrow aA \mid bC \mid a$	9

1. Conversion from Finite automata to regular grammar.

Regular grammar and Finite Automata are equivalent in power.

Example 1:

Converting a given finite automata accepting L={n_a(w)mod 2=0 and n_b(w)mod 2=0 } to regular grammar.



L={even number of a's and b's}

Start state of automata will be the start symbol of the grammar.

We start with S,S on seeing terminal a it moves to state A(S→ aA) and on seeing terminal b it moves to state B (S→ bB).

Since S is also the final state,we introduce the production S→ λ.

S→ aA|bB| λ

We will repeat the same for other states and terminal symbols.

A→ aS|bC   B→ aC|bS   C→ aB|bA

So the grammar is,

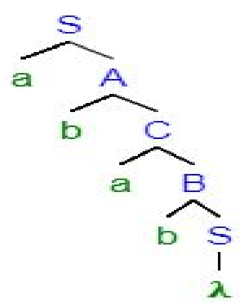
S→ aA|bB| λ

A→ aS|bC

B→ aC|bS

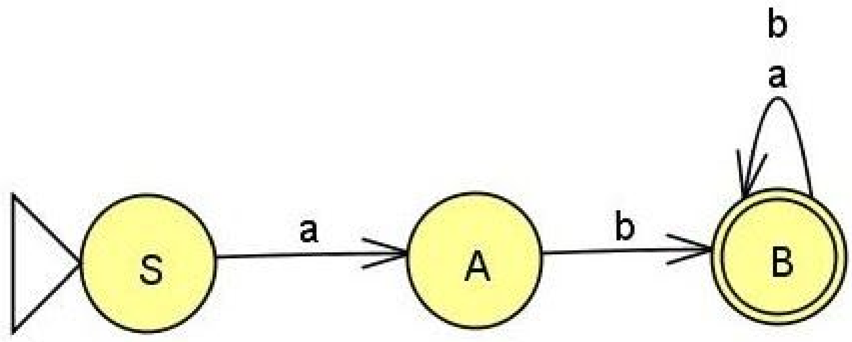
C→ aB|bA

Parse tree for the string abab.



Example 2:

Converting a given finite automata accepting L={abw,w∈{a,b}*} to regular grammar.



Start state of automata will be the start symbol of the grammar.

State S on seeing terminal a it goes to state A. So we introduce the production S→ aA

State A on seeing terminal a it goes to state B. So we introduce the production S→ bB

State B on seeing terminal a,b remains in state B. So we introduce the production B→ aB|bB.

Any final state we introduce the production B→ λ.

So the grammar is,

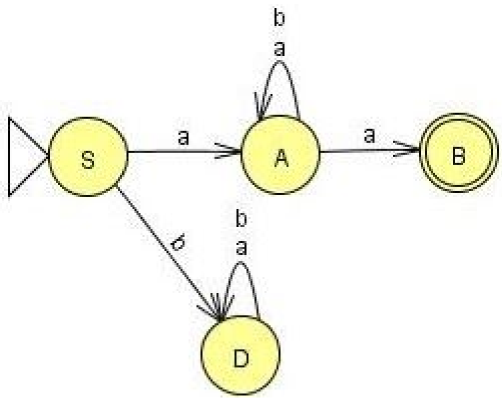
S→ aA

A→ bB

B→ aB|bB|λ

Example 3:

Converting given finite automata accepting L={awa,w∈{a,b}*} to regular grammar.



Start state of automata will be the start symbol of the grammar.

State S on seeing terminal a it goes to state A. So we introduce the production S→ aA

State A on seeing terminal a,b remains in state A. So ,we introduce the production

A→ aA|bA.

State A on seeing terminal a it goes to state B. So we introduce the production A→ aB

Since B is the final state we introduce the production B→ λ**.**

We will not encode the production to state D ,as D is a dead state and it will never lead us to the terminal.It is an useless production.

So the grammar is,

S→ aA

A→ aA|bA|aB

B→ λ

2. Converting Regular grammar to finite automata.

Example 1:

Convert given regular grammar to finite automata.

$S \rightarrow bS|aA$

$A \rightarrow aA|bA|aB$

$B \rightarrow bbB|\ \lambda$

State S on seeing b remains in S,we will have a self loop on S.

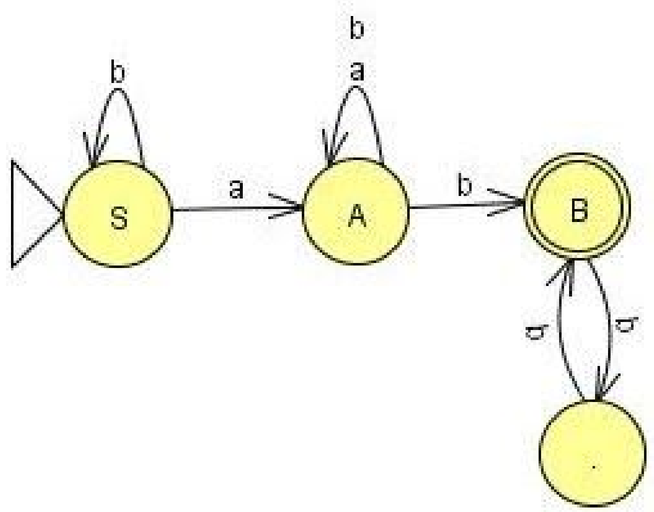
State S on seeing a moves to state A.

A on seeing a and b remains in state A ,we will have a self loop on A.

A on seeing a also moves to state B.

B loops on bb,we introduce a new state and loop on bb.

$B \rightarrow\ \lambda$ indicates B is a final state.



Example 2:

Convert given regular grammar to finite automata.

S→ 01A

A→ 10B

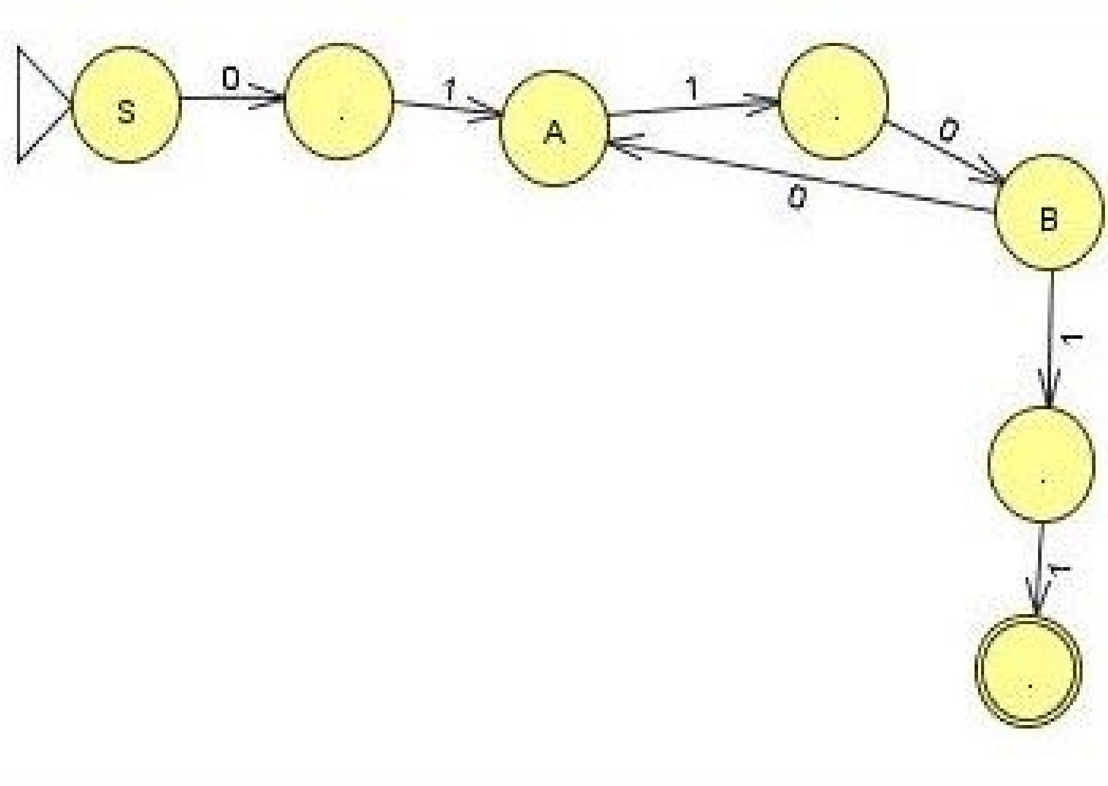
B→ 0A|11

State S on seeing 01 is moving to state A.

State A on seeing 10 is moving to state B.

State B on seeing 0 moves to state A.

B ends with 11.



Example 3:

Convert given regular grammar to finite automata.

A→ aB|bA|b

B→ aC|bB

C→ aA|bC|a

State A seeing terminal a is moving to state B.

State A seeing terminal b remains in state B.

State B seeing terminal a is moving to state C.

State B seeing terminal b remains in state B.

State A accepts only b as well.

State C on a moves to final state.