Automata Formal Languages & Logic

Preet Kanwal

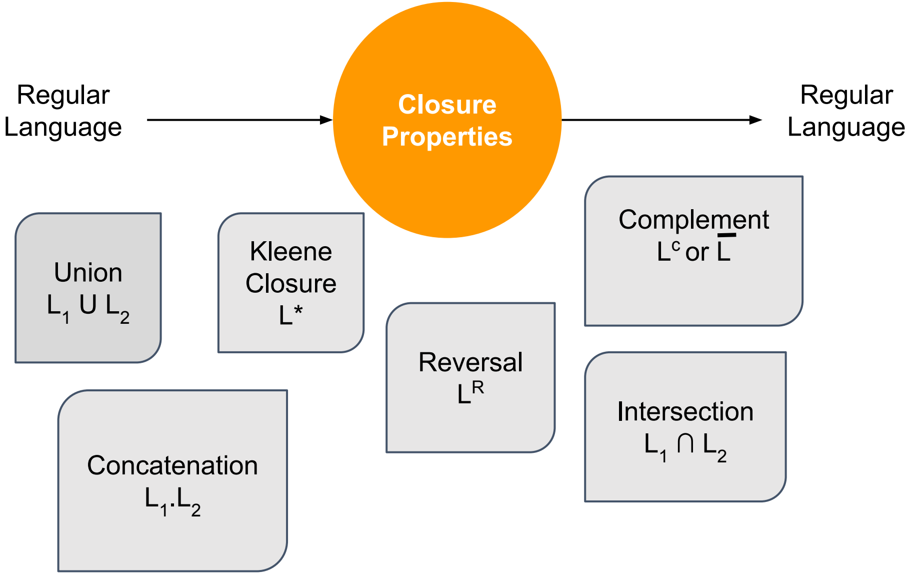
Department of Computer Science & Engineering

Automata Formal Languages & Logic

Unit 2

Preet Kanwal

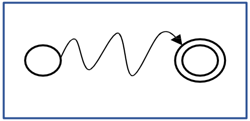
Department of Computer Science Engineering


Regular Language
Closure Properties
Regular Language
Union
$L_1 \cup L_2$
Kleene Closure
L^*
Complement
L^c or $\overline{L}$
Reversal
L^R
Intersection
$L_1 \cap L_2$
Concatenation
$L_1.L_2$

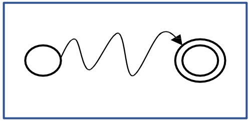
Unit 2 - Regular Languages are closed under Union

Union
L_1 U L_2

M_1



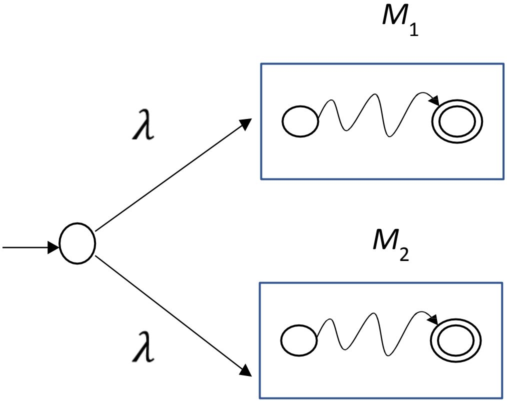
M_2



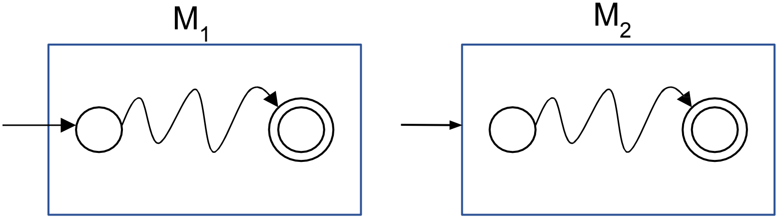
Automata Formal Languages and Logic

Unit 2 - Regular Languages are closed under Union

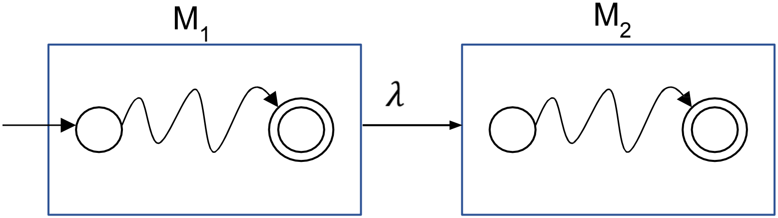
Union
$L_1 \cup L_2$



Concatenation
$L_1.L_2$

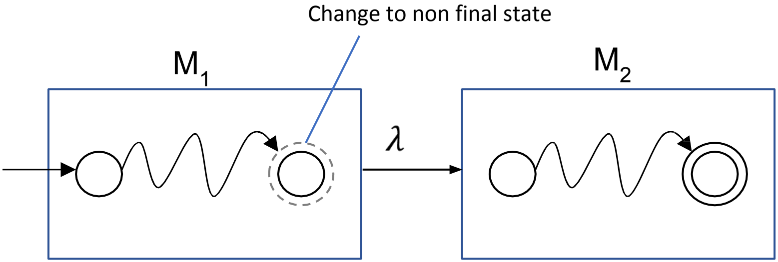


Concatenation
$L_1.L_2$



Unit 2 - Regular Languages are closed under Concatenation

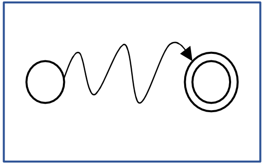
Concatenation
$L_1.L_2$



Unit 2 - Regular Languages are closed under Kleene Closure

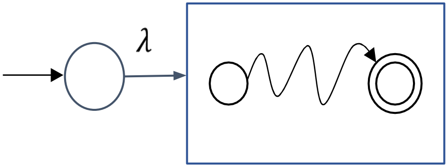
M

Kleene Closure
L*



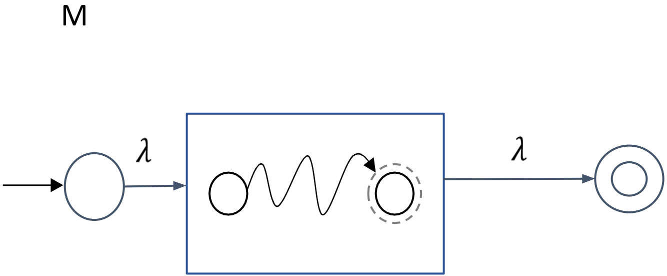
Kleene Closure L*

M



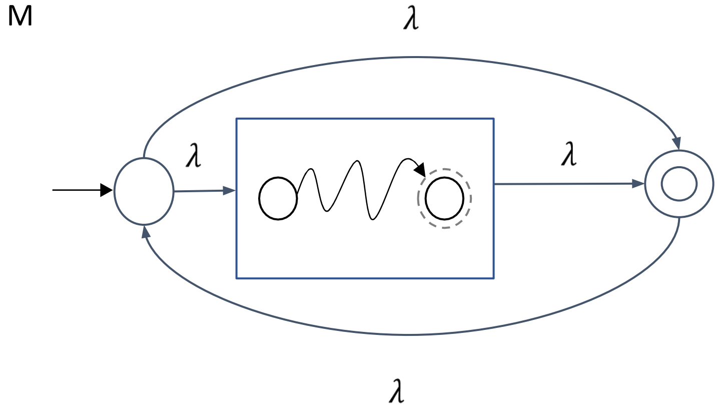
Unit 2 - Regular Languages are closed under Kleene Closure

Kleene Closure L*



Unit 2 - Regular Languages are closed under Kleene Closure

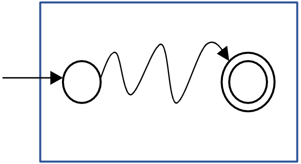
Kleene Closure L*



Unit 2 - Regular Languages are closed under Reversal

Reversal
L^R

M



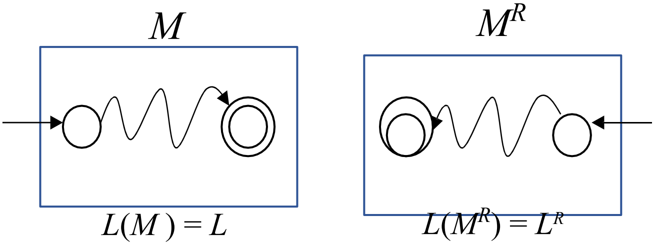
$L(M) = L$

How to reverse of a Regular Language :

- Make Start State as Final State
- Final State as Start State
- Reverse all the transitions
- If there is more than one final state in the machine M , a new start can be introduced with lambda transitions leading from it to these multiple final states.

Unit 2 - Regular Languages are closed under Reversal

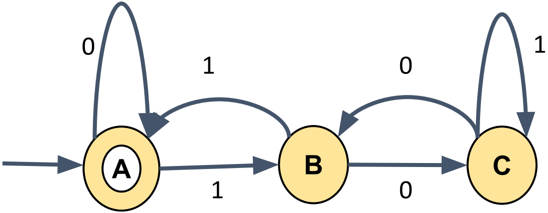
Reversal L^R



How to reverse of a Regular Language :

- Make Start State as Final State
- Final State as Start State
- Reverse all the transitions
- If there is more than one final state in the machine M , a new start can be introduced with lambda transitions leading from it to these multiple final states.

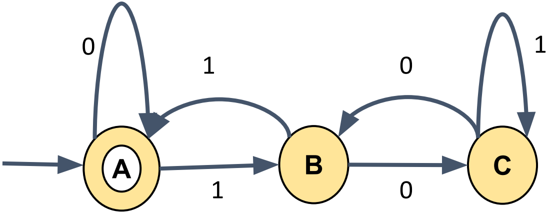
Unit 2 - Example where L and L^R is a same language



L = L(M)

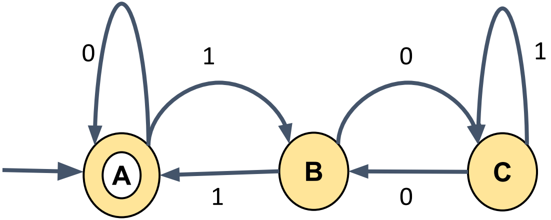
= Binary Strings divisible by 3

Unit 2 - Example where L and L^R is a same language



L = L(M)

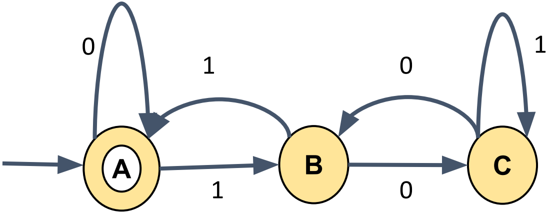
= Binary Strings divisible by 3



L^R=L(M^R)

= Binary Strings divisible by 3

Unit 2 - Example where L and L^R is a same language



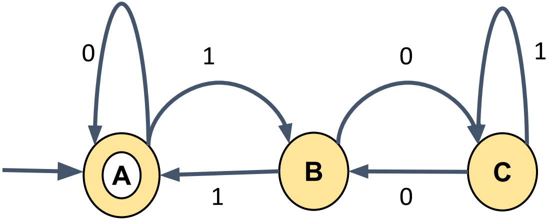
$L = L(M)$

= Binary Strings divisible by 3

Each String is a palindrome

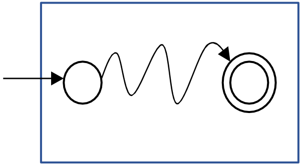
$L^R = L(M^R)$

= Binary Strings divisible by 3



Complement
L^c or $\overline{L}$

DFA - M



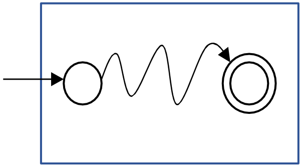
How to Complement a Regular Language :

- Construct a DFA.
- Toggle the final and non-final states.
- Works only for the DFAs and not for the NFAs

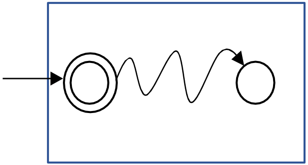
Unit 2 - Regular Languages are closed under Complement

Complement
L^c or $\overline{L}$

DFA - M



M^c

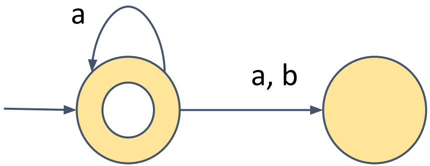


$$L(M^c) = \overline{L}$$

How to Compliment a Regular Language :

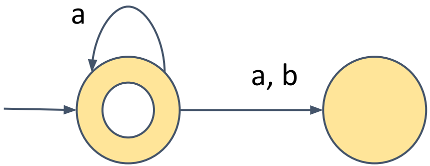
- Construct a DFA.
- Toggle the final and non-final states.
- Works only for the DFAs and not for the NFAs

Complement of a NFA **may not** be complement of the regular language

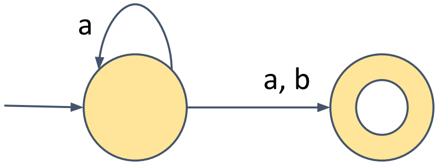


$L = \{ \lambda, a, aa, aaa .. \}$

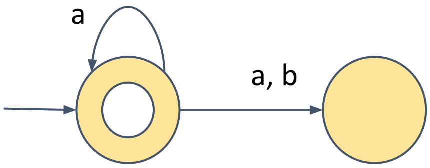
Complement of a NFA **may not** be complement of the regular language



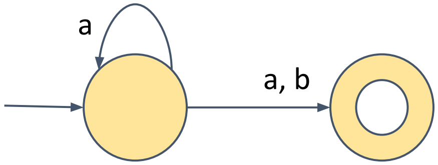
L = { λ, a, aa, aaa .. }



Complement of a NFA may not be complement of the regular language

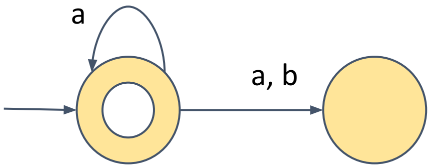


L = { λ, a, aa, aaa .. }



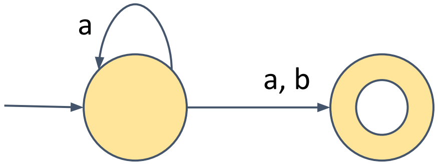
L = a*(a+b)

Complement of a NFA **may not** be complement of the regular language



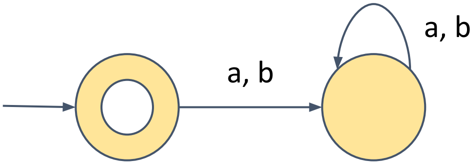
L = { λ, a, aa, aaa .. }

Complement of NFA is a not a complement of the language



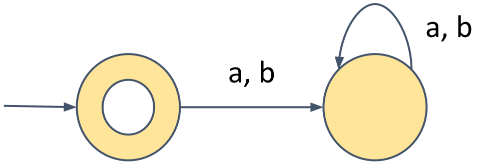
L = a*(a+b)

Complement of a NFA may not be complement of the regular language

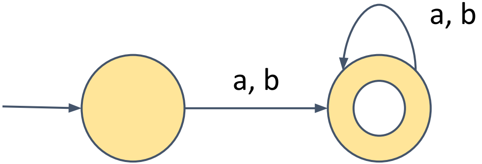


$L = \{ \lambda \}$

Complement of a NFA may not be complement of the regular language



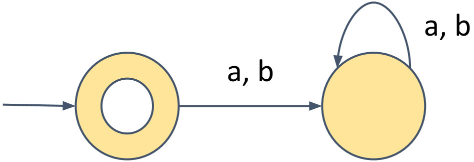
$L = \{ \lambda \}$



$L = (a+b)^+$

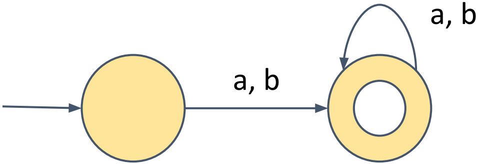
Unit 2 - NFAs cannot be complemented

Complement of a NFA may not be complement of the regular language



$L = \{ \lambda \}$

Complement of NFA is a complement of the language

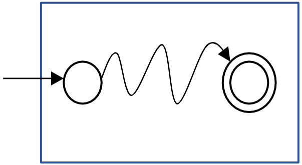


$L = (a+b)^{+}$

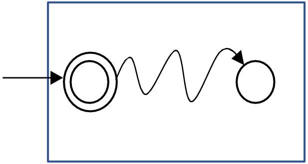
Unit 2 - Regular Languages are closed under Intersection

Intersection
$L_1 \cap L_2$

M_1



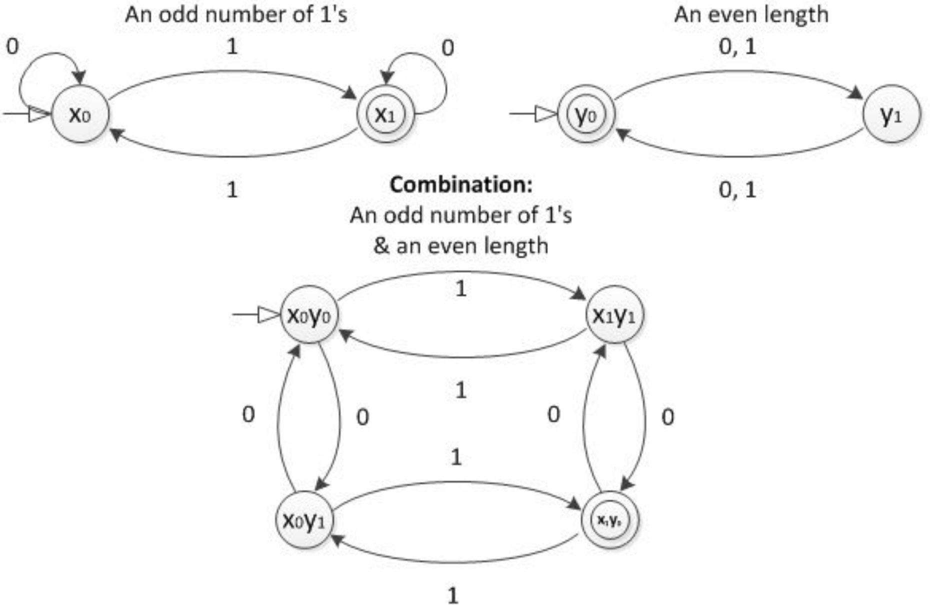
M_2



Two ways to find Intersection of two Regular Languages :

1. Take Cartesian production of the states of the two FA:
   - #of States in new FA will be **m x n** where,
     m - # states in M_1 and n - # states in M_2
   - Start state of new FA will be the pair :
     {Start State of M_1 , Start State of M_2)}
   - Final state of new FA will be the pair :
     {Final State of M_1 , Final State of M_2)}
2. Use De Morgan's Law

Unit 2 - Regular Languages are closed under Intersection



DeMorgan's Law: $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$

$L_1\ ,\ L_2$	regular, regular
$\Rightarrow$ $\overline{L_1}\ ,\ \overline{L_2}$	regular, regular
$\Rightarrow$ $\overline{L_1} \cup \overline{L_2}$	regular
$\Rightarrow$ $\overline{\overline{L_1} \cup \overline{L_2}}$	regular
$\Rightarrow$ $L_1 \cap L_2$	regular

Answer the following :

1. if $L_1 = \{a^n b\}$ and $L_2 = \{ ba\}$ then prove that $L_1 \cup L_2$ is regular
2. if $L_1 = \{a^n b\}$ and $L_2 = \{ ba\}$ then prove that $L_1 . L_2$ is regular
3. if $L = \{a^n b\}$ then prove that L^* is regular
4. if $L = \{a^n b\}$ then prove that L^R is regular
5. if $L = \{a^n b\}$ then prove that L^c is regular
6. Let L1 = {w | w contains '11' as a substring} and

   L2 = {w | w contains '00' as a substring} then prove that L1 ∩ L2 is regular. That is come up with an automaton accepting the strings that contain both '11' and '00' as a substring.

Can you answer the following about Regular language:

1. Membership question : Does String w $\in$ L?
2. Testing Emptiness : Does L = { }?
3. Does L = Σ^* ??
4. Is a given language finite or infinite?
5. Given a regular language L_1 and L_2 can we check if $L_1 = L_2$?

Can you answer the following about Regular language:

1. Membership question : Does String w $\in$ L?

   Construct a FA for L and check whether w lands in a final state.

2. Testing Emptiness : Does L = { }?
3. Does $L = \Sigma^*$??
4. Is a given language finite or infinite?
5. Given a regular language L_1 and L_2 can we check if $L_1 = L_2$?

Can you answer the following about Regular language:

1. Membership question : Does String w $\in$ L?

   Construct a FA for L and check whether w lands in a final state.

2. Testing Emptiness : Does L = { }?

   Yes if there is no path from start state to final state.

3. Does $L = \Sigma^*$??

4. Is a given language finite or infinite?

5. Given a regular language L_1 and L_2 can we check if $L_1 = L_2$?

Can you answer the following about Regular language:

1. Membership question : Does String w $\in$ L?
   Construct a FA for L and check whether w lands in a final state.
2. Testing Emptiness : Does L = { }?
   Yes if there is no path from start state to final state.
3. Does $L = \Sigma^*$??
   Check if complement of L accepts nothing.
4. Is a given language finite or infinite?
5. Given a regular language L_1 and L_2 can we check if $L_1 = L_2$?

Can you answer the following about Regular language:

1. Membership question : Does String w $\in$ L?

   Construct a FA and check whether w lands in a final state.

2. Testing Emptiness : Does L = { }?

   Yes if there is no path from start state to final state.

3. Does $L = \Sigma^*$??

   Check if complement of L accepts nothing.

4. Is a given language finite or infinite?

   Construct if FA for L contains a loop. If yes, then the lang is infinite.

5. Given a regular language L_1 and L_2 can we check if $L_1 = L_2$?

Can you answer the following about Regular language:

1. Membership question : Does String $w \in L$?

   Construct a FA and check whether w lands in a final state.

2. Testing Emptiness : Does $L = \{\ \}$?

   Yes if there is no path from start state to final state.

3. Does $L = \Sigma^*$??

   Check if complement of L accepts nothing.

4. Is a given language finite or infinite?

   Construct if FA for L contains a loop. If yes, then the lang is infinite.

5. Given a regular language L_1 and L_2 can we check if $L_1 = L_2$?

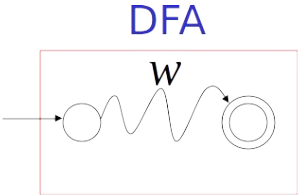
   If minimal DFA for both L1 and L2 is same the languages are equal.

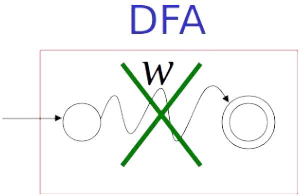
THANK YOU

Preet Kanwal
Department of Computer Science & Engineering
preetkanwal@pes.edu
+91 80 6666 3333 Extn 724

Decidable properties of regular language

1. Membership Question



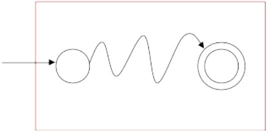


$w \in L$

$w \notin L$

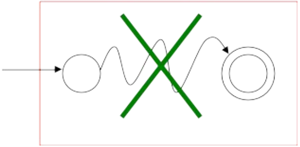
2. Testing Emptiness

DFA



$$L \neq \emptyset$$

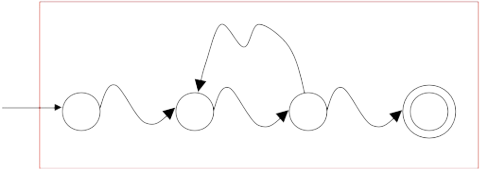
DFA



$$L = \emptyset$$

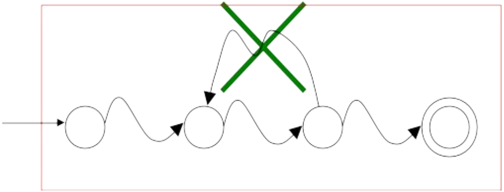
3. Is a given regular language finite or Infinite?

DFA



L is infinite

DFA



L is finite

4.Given a regular language L_1 and L_2 how can we check if $L_1 = L_2$?

Regular Language L_1

Regular expression R_1

Equivalent NFA

Construct DFA

Minimize DFA

Regular Language L_2

Regular expression R_1

Equivalent NFA

Construct DFA

Minimize DFA

RE (A) =RE (B) iff the minimized DFA of both the expression are same as the minimized DFA is unique

THANK YOU

Preet Kanwal
Department of Computer Science & Engineering
preetkanwal@pes.edu
+91 80 6666 3333 Extn 724