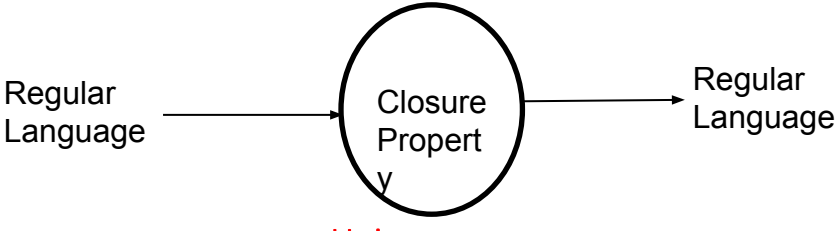
Automata Formal Languages & Logic

Preet Kanwal
Department of Computer Science & Engineering

Automata Formal Languages & Logic

Unit 2

Preet Kanwal
Department of Computer Science Engineering



Union
Concatenation
Star
Reversal
Complement
Intersection

For regular languages L_1 and L_2 we will prove that:

Union : $L_1 \cup L_2$

Concatenation: $L_1 L_2$

Star: L_1^*

Reversal: L_1^R

Complement: $\overline{L_1}$

Intersection: $L_1 \cap L_2$

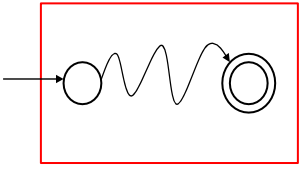
Are regular languages

Take two languages

Regular language
L_1

$L(M_1) = L_1$

NFA M_1

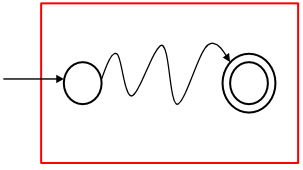


Single accepting state

Regular language
L_2

$L(M_2) = L_2$

NFA M_2

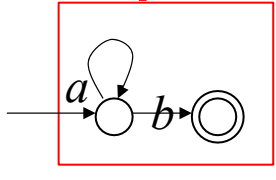


Single accepting state

Example

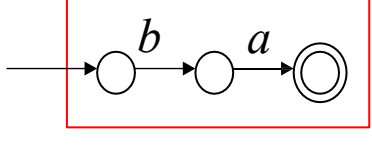
$L_1 = \{a^n b\}$

M1



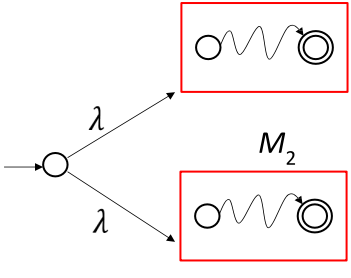
$L_2 = \{ ba\}$

M2



Union

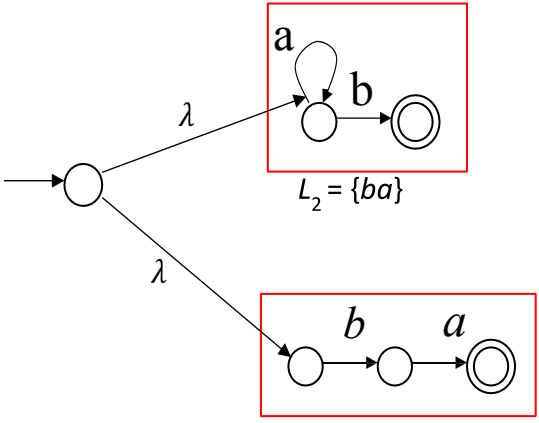
NFA for $L_1 \cup L_2$



$w \in L_1 \cup L_2 \iff w \in L_1$ or $w \in L_2$

NFA for $L_1 \cup L_2 = \{a^n b\} \cup \{ba\}$

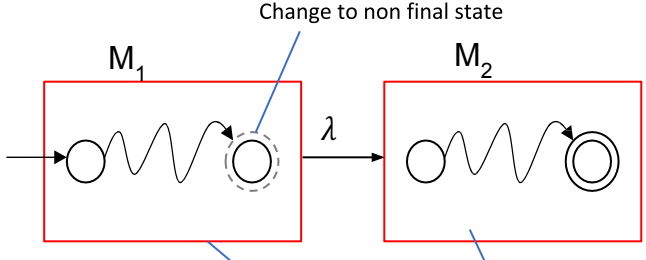
$L_1 = \{a^n b\}$



$L_2 = \{ba\}$

Concatenation

NFA for $L_1 L_2$

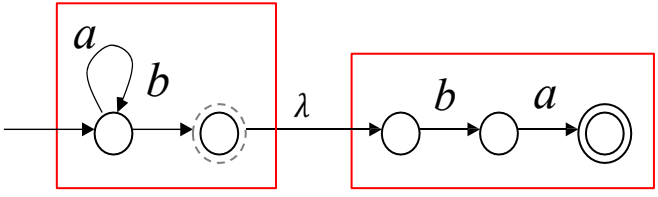


$w \in L_1 L_2 \iff w = w_1 w_2 : w_1 \in L_1 \text{ and } w_2 \in L_2$

Example

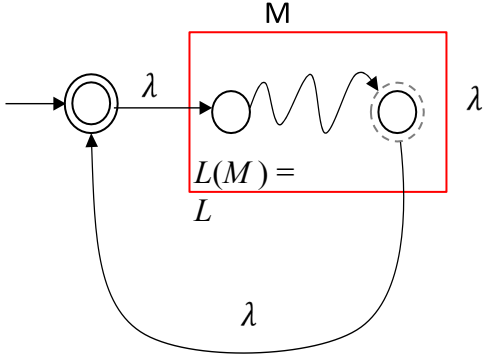
NFA for $L_1 L_2 = \{a^n b\}\{ba\} = \{a^n bba\}$

$L_1 = \{a^n b\}$ $\quad$ $L_2 = \{ba\}$



Star Closure

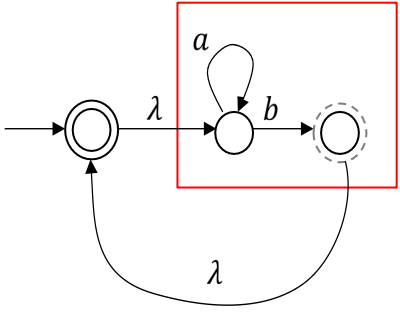
NFA for L^*



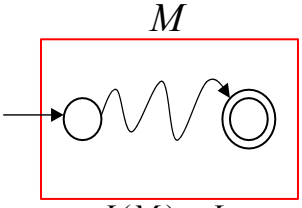
Example

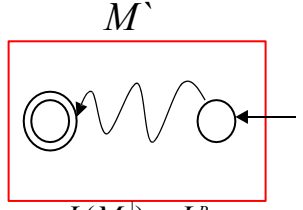
NFA for $L^* = \{a^n b\}^*$

$$L_1 = \{a^n b\}$$

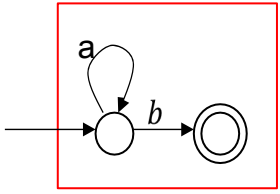


Reverse



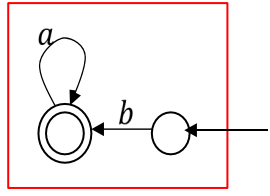


$L_1 = \{a^n b\}$

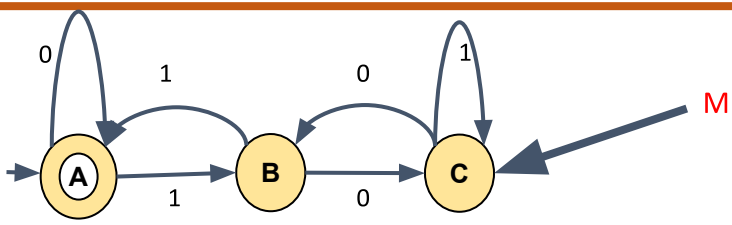


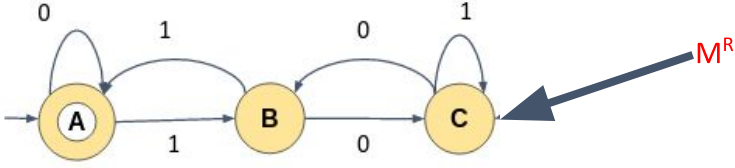
M_1

$L^R = \{ba^n\}$

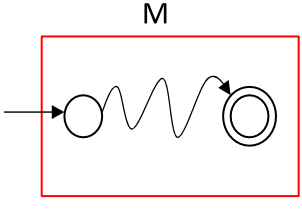


$M_1^{\ l}$

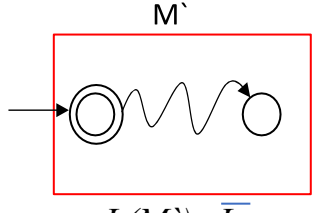

0
1
0
1
A
B
C
1
0
M


0
1
0
1
A
B
C
1
0
M^R

Complement



$L(M) = L$

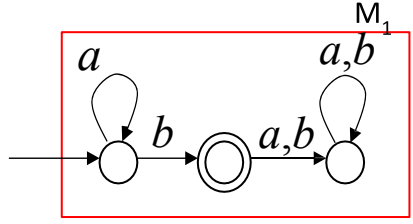


$L(M`) = \overline{L}$

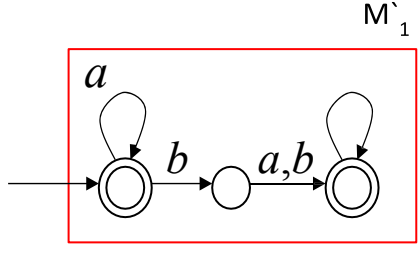
1. Take the DFA that accepts L
2. Make accepting state has non accepting and vice versa

Example

$L_1 = \{a^n b\}$



$\overline{L_1} = \{a,b\}^* - \{a^n b\}$

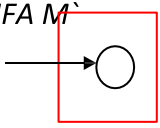


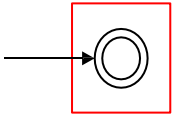
NFAs cannot be used for complement

Make accepting state has non accepting and vice versa

NFA M

NFA M`





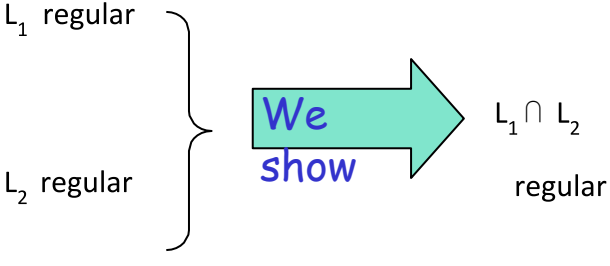
L(M) = { }

$L(M') = \{\lambda\} \neq \overline{L(M)}$

$\overline{L(M)} = \sum^* = \{a,b\}^*$

It is not the complement

Intersection



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

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

Example

$L_1 = \{a^n b\}$ regular

$L_2 = \{ab, ba\}$ regular

$\Rightarrow L_1 \cap L_2 = \{ab\}$

regular

Another Proof for Intersection

Machine M_1

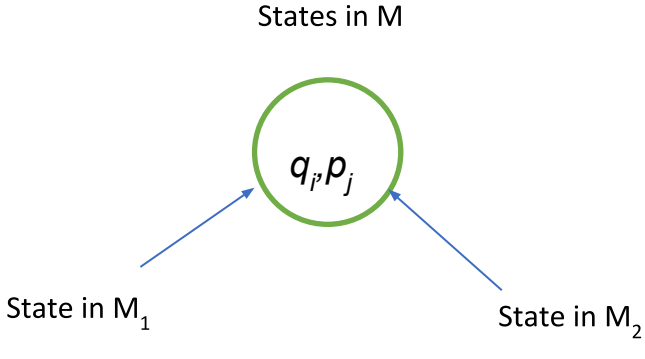
DFA for L_1

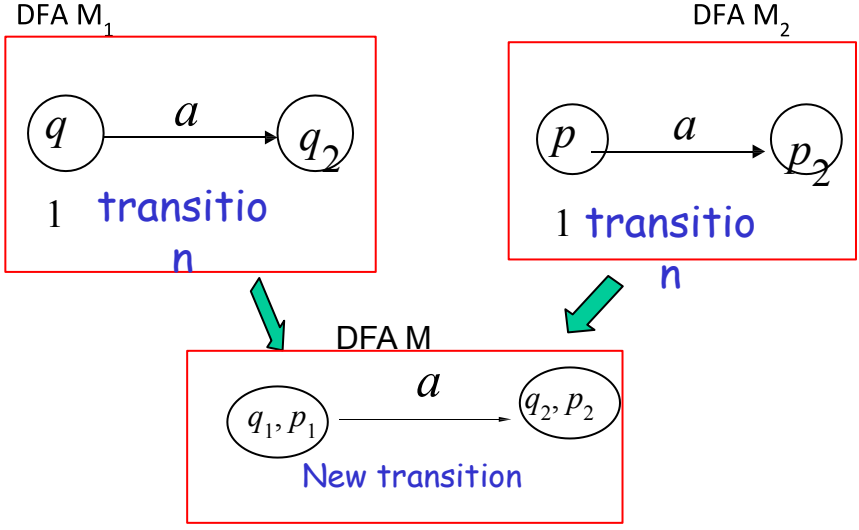
Machine M_2

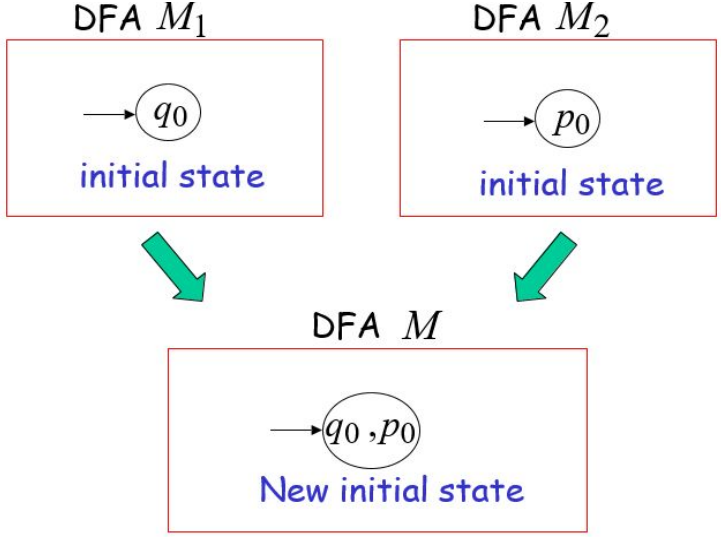
DFA for L_2

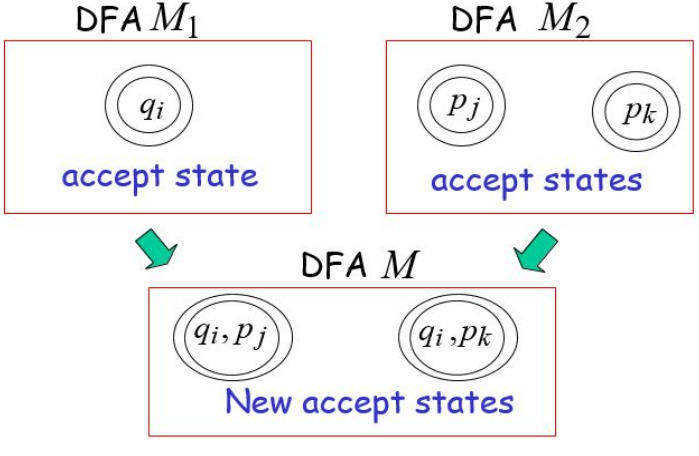
Construct a new DFA M that accepts $L_1 \cap L_2$

M simulates in parallel M_1 and M_2


States in M
q_i, p_j
State in M_1
State in M_2

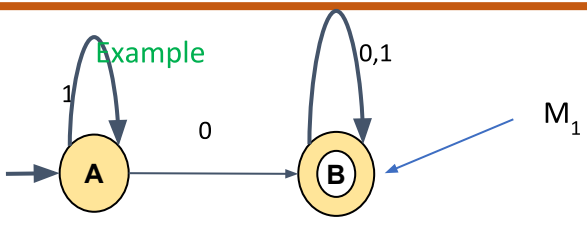

DFA M_1
q
a
q_2
1 transitio
n
DFA M_2
p
a
p_2
1 transitio
n
DFA M
a
q_1, p_1
q_2, p_2
New transition

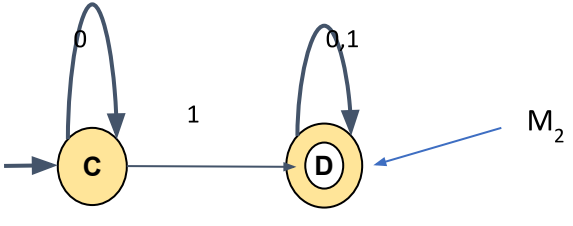

DFA M_1
q_0
initial state
DFA M_2
p_0
initial state
DFA M
q_0, p_0
New initial state

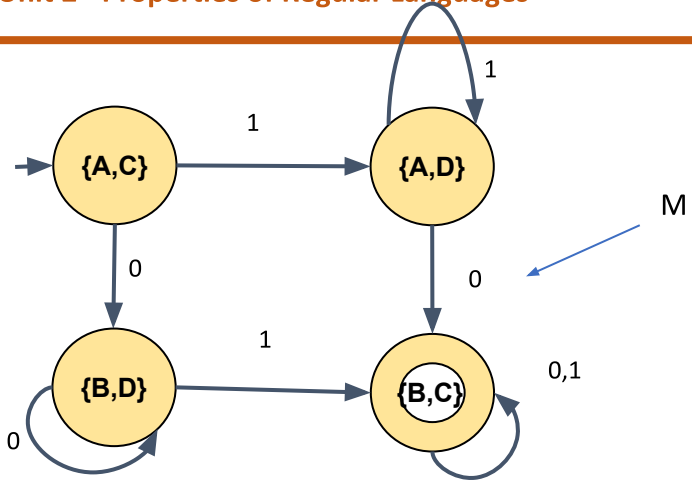


Both constituents must be accepting states

Example



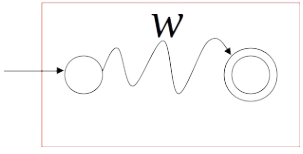



1
1
{A,C}
{A,D}
M
0
0
1
{B,D}
{B,C}
0,1
0

Decidable properties of regular language

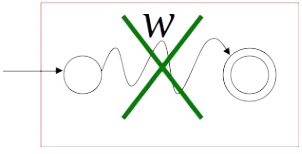
1. Membership Question

DFA



$w \in L$

DFA



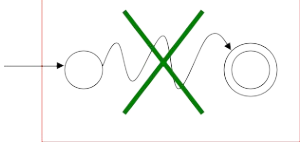
$w \notin L$

2. Testing Emptiness

DFA

$L \neq \emptyset$

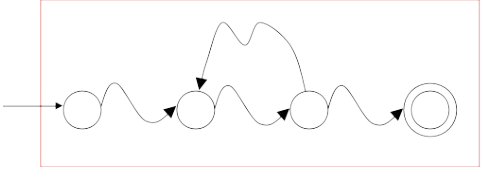
DFA



$L = \emptyset$

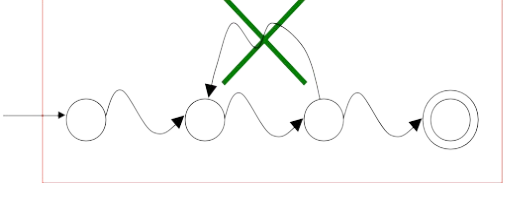
3.Is a given regular language finite or Infinite?

DFA



L is infinite

DFA

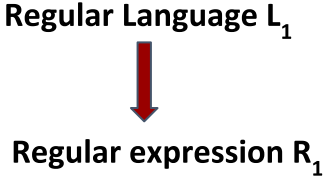


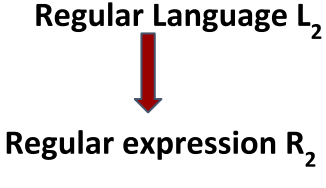
L is finite

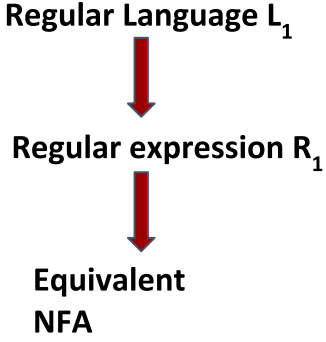
4.Given a regular language L_1 and L_2 how can we check if $L_1 = L_2$?

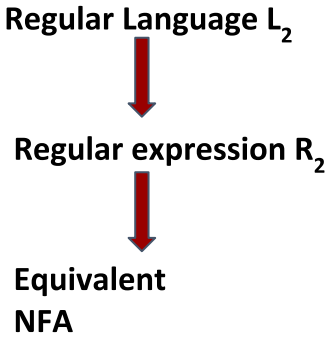
Regular Language L_1

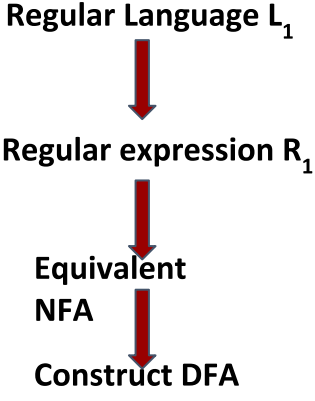
Regular Language L_2

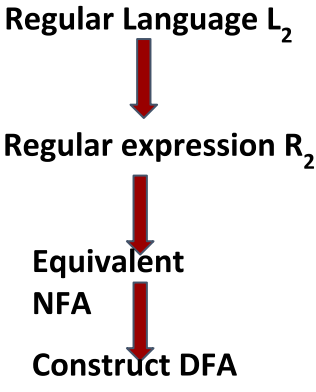

Regular Language L_1
Regular expression R_1

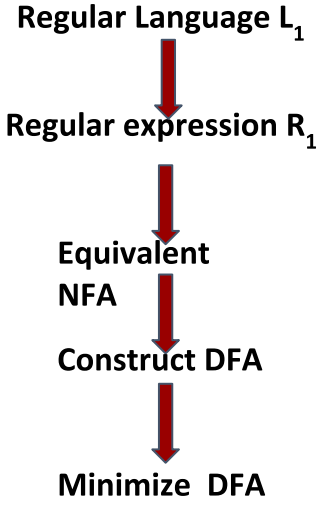

Regular Language L_2
Regular expression R_2

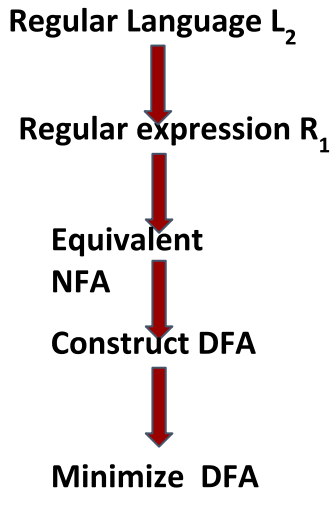

Regular Language L_1
Regular expression R_1
Equivalent
NFA

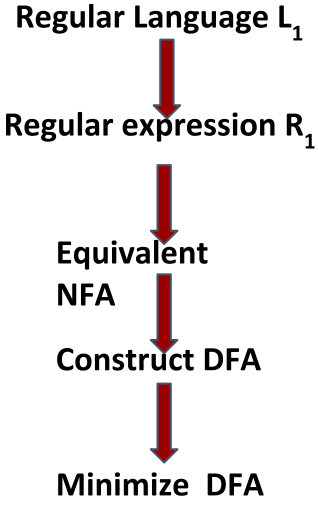

Regular Language L_2
Regular expression R_2
Equivalent
NFA

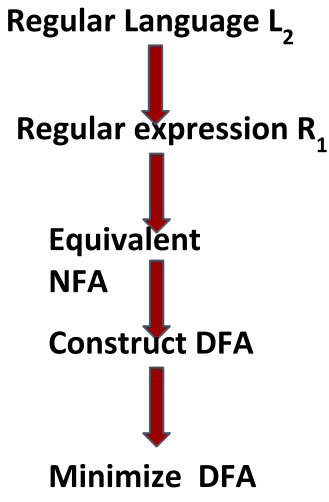

Regular Language L_1
Regular expression R_1
Equivalent
NFA
Construct DFA


Regular Language L_2
Regular expression R_2
Equivalent
NFA
Construct DFA


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

PES UNIVERSITY ONLINE

Example 1:

$L_1(R_1) = R_1 = (0+1)^*(0+\lambda)$ $L_2(R_2) = R_2 = (1+\lambda)(1+0)^*$

PES UNIVERSITY ONLINE

Example 1:

$(0+1)^*(0+\lambda)$

R_1

$(1+\lambda)(1+0)^*$

R_2

PES UNIVERSITY ONLINE

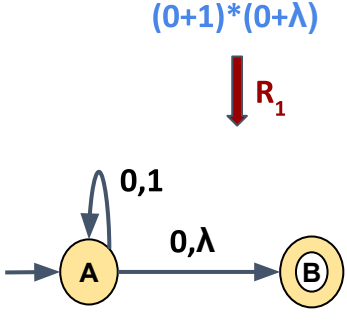
Example 1:

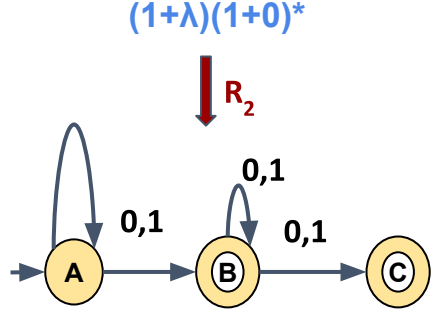
$(0+1)^*(0+\lambda)$ → R_1

$(1+\lambda)(1+0)^*$ → R_2

Convert the regular expression R1 and R2 to its equivalent NFA

Example 1:

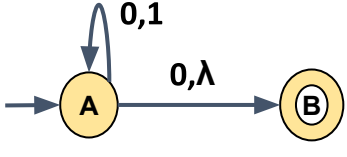




Example 1:

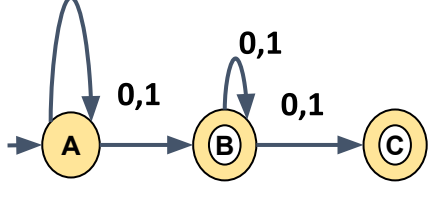
$(0+1)^*(0+\lambda)$

R_1

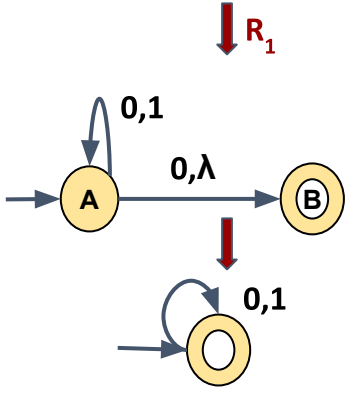


$(1+\lambda)(1+0)^*$

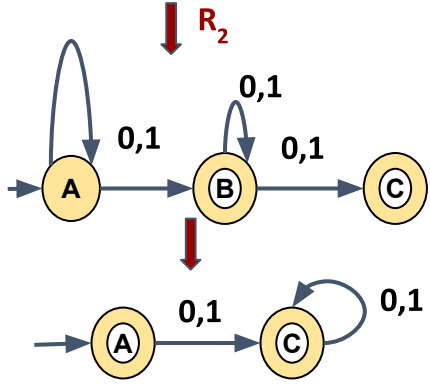
R_2



Convert the R_1 NFA and R_2 NFA to its equivalent DFA

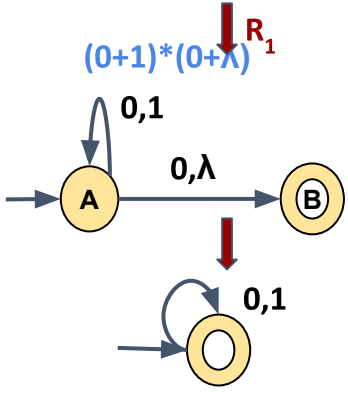


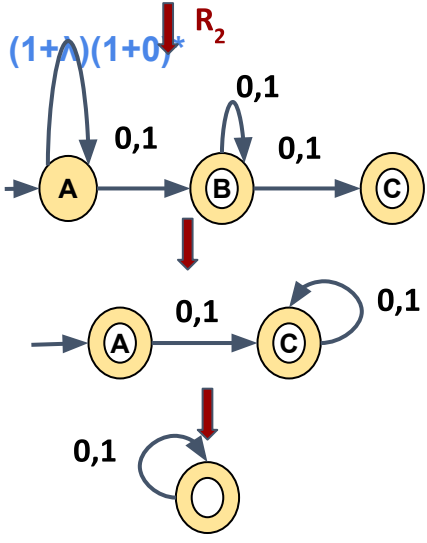
R_1 DFA is is minimized



Minimize R_2 DFA

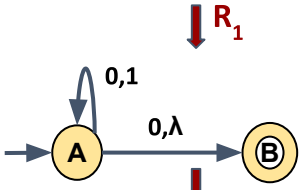
Example 1:



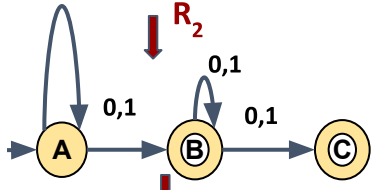


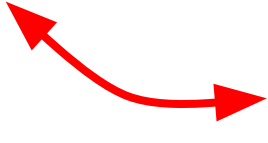
PES UNIVERSITY ONLINE

Example 1: (0+1)*(0+λ)



(1+λ)(1+0)*





R_1 DFA = R_2 DFA

$L_1(R_1) = L_2(R_2)$

THANK YOU

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