AUTOMATA FORMAL LANGUAGES AND LOGIC

Lecture Notes on
Conversion of NFA/λ-NFA to DFA

Prepared by :
Prof. Kavitha K N
Assistant Professor

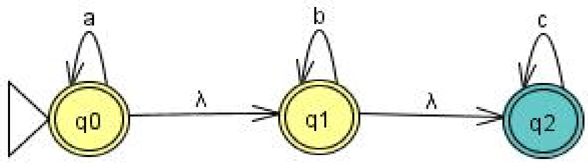
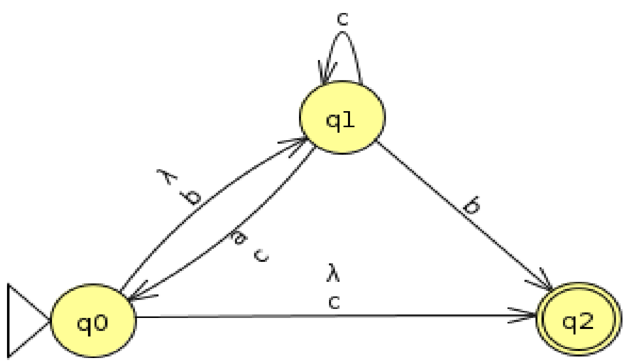
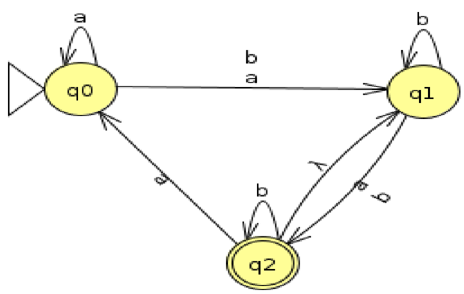
Department of Computer Science & Engineering

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

4	a b c q0 λ q1 λ q2	**13**
5	c q1 λ b a c b λ c q0 q2	**19**
6	a b b a q0 q1 a b λ a b q2	**20**

1.Conversion of NFA or λ-NFA to DFA

DFAs can be viewed as a special case of NFAs; i.e., those for which the there is precisely one start state ($S = \{q0\}$) and for which the transition function always returns singleton (one-element) sets ($\delta(q, x) = \{q'\}$ for all $q \in Q$ and $x \in \Sigma$). The opposite is also true, however: NFAs are really just DFAs "in disguise"

2.Why is the conversion required?

A non-deterministic finite automaton may have transitions to any number of states for a given input and state. It may also have NULL transition(transition without input) This may be a problem for some computer programs because it needs precisely one transition for a given input from a given state. The process of converting NFA to DFA eliminates this ambiguity.

While constructing an automata, it is quite easy to form an NFA instead of DFA later that NFA can be converted to its equivalent DFA and it is sometimes necessary to convert an NFA into a DFA as DFA makes the recognition faster and DFA may also result in less number of states making the execution faster and its implementation will be efficient in terms of time and space and this is also said to be equivalence of two automata. Two finite automata M and N are said to be equivalent if $L(M) = L(N)$

NFA / λ-NFA and DFA are equivalent in power, that means all these machines are capable of accepting the same class of m languages, every NFA and λ-NFA can be converted to DFA, this is also said to be the equivalence of two automata,

.Key idea: Build a DFA whose states represent sets of states in the NFA

3.Subset Construction method:

The method that will be used to convert NFA / to DFA is known as Subset construction method
The subset construction: Given an NFA $A = (Q, \Sigma, \delta, S, F)$ we construct the equivalent DFA:
The basic idea of this construction is to define a DFA whose states are sets of NFA states. A set of possible NFA states thus becomes a single DFA state. The DFA transition function is given by considering all reachable NFA states for each of the current possible NFA states for each input symbol. The resulting set of possible NFA states is again just a single DFA state. A DFA state is final if that set contains at least one final NFA state.
It follows the following steps:

Note: You can convert the given DFA transition diagram to its equivalent transition table as it makes the conversion easier while doing the conversion manually

Step 1: Make the start state of NFA as the start state of DFA

Step 2: Find the transition of the start state to another set of states on a set of alphabets(set of input symbols).

Step 3: Add the newly emerged state as a state of DFA and find the combination of states which the current state transit on the input alphabets.

Step 4: This process is repeated until no new states emerge.

Step 5: Make the state that contains any of the final states of NFA as the final state of its equivalent DFA.

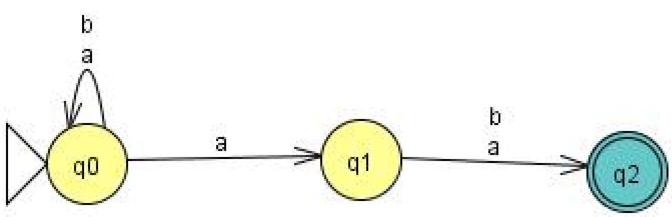
4.Example on converting NFA to DFA.

Example 1:

Consider the following NFA,

Σ={a,b}

L={Strings where the second symbol from RHS is a}



Note: Convert the above DFA transition diagram to its equivalent transition table

States{q0,q1,q2}

Input symbol{a,b}

δ(q0,a)={q0,q1}

	a
q0	{q0,q1}

δ(q0,b)=q0

	a	b

q0	{q0,q1}	q0

δ(q1,a)=q2

	a	b
q0	{q0,q1}	q0
q1	q2	

δ(q1,b)=q2

	a	b
q0	{q0,q1}	q0
q1	q2	q2

δ(q2,a)=Φ
δ(q2,b)=Φ
no transition so specify that an phi(Φ) and the resulting final transition table is given below:

NFA transition table:

	a	b
q0	{q0,q1}	q0
q1	q2	q2
q2	Φ	Φ

Now construct the transition table for DFA from the transition table of NFA
DFA transition table:
Step 1: Make the start state of NFA as the start state of DFA

	a	b
q0		

Step 2: Find the transition of the start state to another set of states on a set of alphabets(set of input symbols).
Note: Look at the NFA transition table to convert to DFA

δ(q0,a)={q0,q1}

	a
q0	{q0,q1}

δ(q0,b)=q0

	a	b
q0	{q0,q1}	q0

Whenever we encounter a state that has not been considered before, we add that state to the table. In this case, one new DFA state {q0,q1} emerge
Step 3: Add the newly introduced state as a state of DFA and find the combination of states which the current state transit on the input alphabets
The newly introduced state is {q0,q1}, this will be treated as one state in DFA

	a	b
q0	{q0,q1}	q0
{q0q1}		

δ({q0q1},a) =
δ(q0,a)={q0,q1}
δ(q1,a)={q2}
δ({q0q1},a) =**δ**{q0,a} U **δ**{q1,a}
= {q0,q1} U {q2}
={q0,q1,q2}

	a	b
q0	{q0,q1}	q0

{q0q1}	{q0,q1,q2}	

δ({q0q1},b) :
δ(q0,b)= q0
δ(q1,b)=q2
δ({q0q1},b) = **δ**(q0,b) U **δ**(q1, b)
=q0 U q2
= {q0,q2}

	a	b
q0	{q0,q1}	q0
{q0q1}	{q0,q1,q2}	{q0,q2}

Two new states has been emerged , they are {q0,q1,q2} and {q0,q2} , add these two states as states of DFA and find the transition on input alphabet

	a	b
q0	{q0,q1}	q0
{q0q1}	{q0,q1,q2}	{q0,q2}
{q0,q1,q2}		
{q0,q2}		

δ({q0,q1,q2},a):
δ(q0,a)={q0,q1}
δ(q1,a)={q2}
δ(q2,a)={ Φ}
δ({q0,q1,q2},a)= **δ(**q0,a} U **δ(**q1,a} U **δ**(q2,a}
={q0,q1} U {q2} U { Φ}
={q0,q1,q2}

	a	b
q0	{q0,q1}	q0
{q0q1}	{q0,q1,q2}	{q0,q2}

{q0,q1,q2}	{q0,q1,q2}	
{q0,q2}		

δ({q0,q1,q2} , b):
δ(q0 ,b)={q0}
δ(q1 ,b)={q2}
δ(q2 ,b)={ Φ}
δ({q0,q1,q2} ,a):= **δ(** q0,a) U **δ(**q1,a) U **δ**(q2,a)
={q0} U {q2} U { Φ}
={q0q2}

	a	b
q0	{q0,q1}	q0
{q0q1}	{q0,q1,q2}	{q0,q2}
{q0,q1,q2}	{q0,q1,q2}	{q0,q2}
{q0,q2}		

δ({q0,q2}, a):
δ(q0,a)={q0q1}
δ(q2,a={Φ}

δ({q0,q2},a)= **δ(**q0,a) U **δ(**q2,b)
= {q0q1} U {Φ}
= {q0q1}

	a	b
q0	{q0,q1}	q0
{q0q1}	{q0,q1,q2}	{q0,q2}
{q0,q1,q2}	{q0,q1,q2}	{q0,q2}
{q0,q2}	{q0q1}	

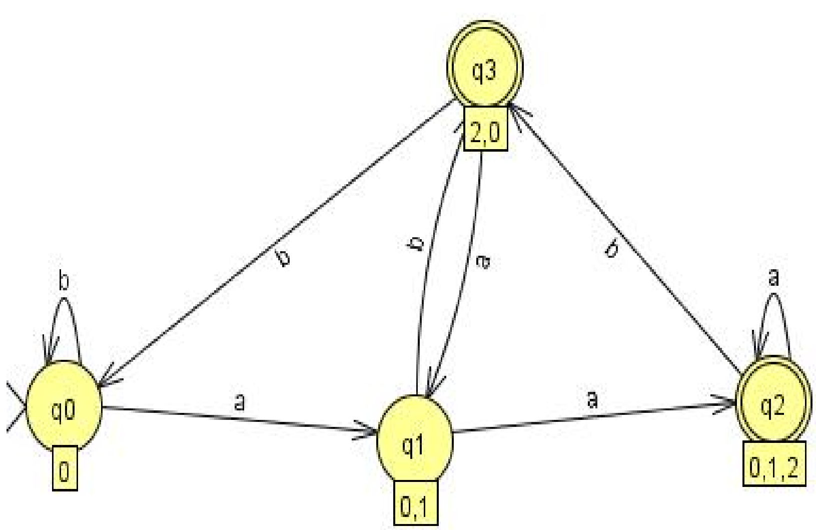
δ({q0,q2},b):
δ(q0,b)={q0}
δ(q2,b)={Φ}

δ({q0,q2},a)= **δ(**{q0,b} U **δ(**q2 on b}
={q0} U {Φ}
=q0

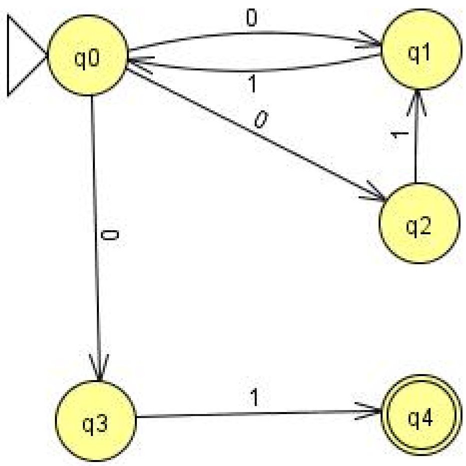
	a	b
q0	{q0,q1}	q0
{q0q1}	{q0,q1,q2}	{q0,q2}
{q0,q1,q2}	{q0,q1,q2}	{q0,q2}
{q0,q2}	{q0q1}	q0

No new states are introduced. Stop here and convert this transition table to DFA transition diagram
States= [{q0}, {q0q1},{q0q1q2},{q1q2}]
Can rename the states as q0=q0, {q0q1}=q1, {0,q1,q2}=q2, {q0,q2}=q3
Input alphabet= a,b

Transition diagram:



Example 2:



NFA Transition table:
δ(q0 , 0) ={q1,q2,q3}
δ(q0 ,1)= Φ
δ(q1 ,0)= Φ
δ(q1 ,1) =q0
δ(q2 ,0)= Φ
δ(q2 ,1)= q1
δ(q3 ,0)= Φ
δ(q3 ,1)= q4
δ(q4 ,0)= Φ
δ(q4 ,1)= Φ

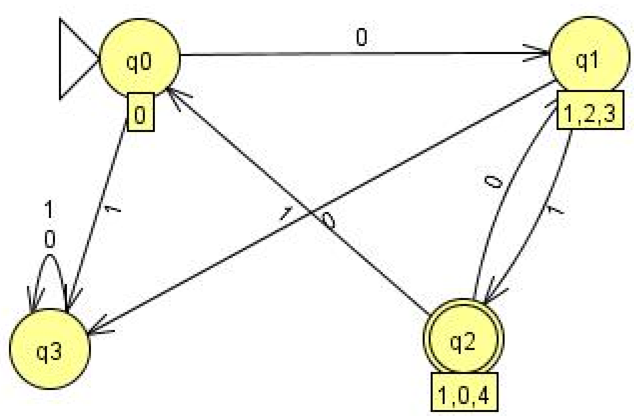
	0	1
-->q0	{q1,q2,q3}	Φ
q1	Φ	q0
q2	Φ	q1
q3	Φ	q4
*q4	Φ	Φ

Transition table for NFA is given below: as Φ is also introduced as one of the state it is considered as dead state as trap state in DFA and need to be included as a state in DFA

δ(q0,0)={q1,q2,q3}
δ(q0 ,1)=Φ
δ({q1,q2,q3} ,0)=Φ
δ({q1,q2,q3} ,1)={q0,q1,q4}
δ({q0,q1,q4},0)={q1,q2,q3}
δ({q0,q1,q4},1)=q0
δ(Φ,0)=Φ
δ(Φ,1)=Φ

	0	1
---> q0	{q1,q2,q3}	Φ
{q1,q2,q3}	Φ	{q0,q1,q4 }
*{q0,q1,q4}	{q1,q2,q3}	q0
Φ	Φ	Φ

Transition diagram for the above DFA transition table is given below:
States: [{q0},{q0q1q2},{q0q1q4},Φ}
Rename the states as q0=q0,q0q1q2=q1, q0q1q4=q2, Φ= q3



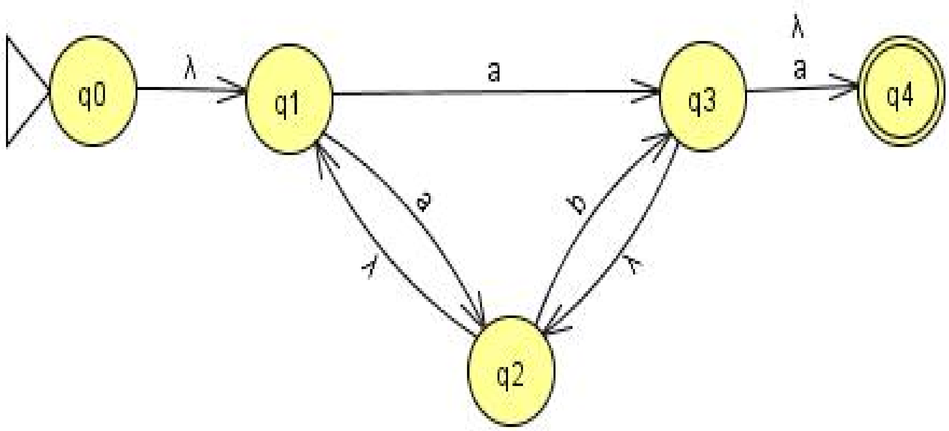
Note: Every NFA is aDFA but not vice versa, but there is an equivalent DFA for every NFA
Note: NFAs and DFAs recognise the same class of languages.

5.Conversion of λ-NFA to DFA

NFA with λ move: If any FA contains λ transaction or move, the finite automata is called NFA with λ move.

λ-Closure: Needed to handle subset construction for NFAs containing ε-transitions
for all NFA states s, the λ closure of s, written λ-closure(s), is defined as set of states reachable from s by traveling along 0 or more λ-edges

5.1 λ-Closure Computation Example



Transition table

	a	b	λ
q0	Φ	Φ	q1
q1	{q2q3}	Φ	Φ
q2	Φ	q3	q1
q3	q4	Φ	{q2q4}
q4	Φ	Φ	Φ

λ-closure of any state will be the state itself and other states which is reached without any input with λ-moves
λ-closure(s)={λ-closure(s)} U {states which is reached without any input with λ-moves}

λ-closure(q0)= {q0,q1}
λ-closure(q1)={q1}
λ-closure(q2)={q1,q2}
λ-closure(q3)={q1,q2,q3,q4}
λ-closure(q4)={q4}

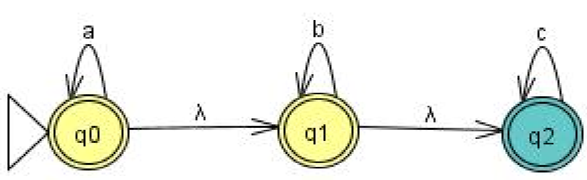
5.2 Examples of Converting λ-NFA to DFA using subset construction method

Convert the given λ-NFA state transition diagram to state transition table and compute the λ-closure of all the states (makes the conversion easier while converting manually)
Step 1: λ-closure of state will be the start state of the DFA
Remaining steps will remain similar except that λ-closure of the states need to be considered instead of just the states

<u>Example 1:</u>

L={$a^n b^m c^k$, n,m,k>=0}
Language consists of strings that can have any number of a's followed by any number of b's followed by any number of c's. Since the number of a's ,b's and c's can be 0 λ isa part of the language

The below shows the λ-NFA transition diagram:



Transition table for λ-NFA :

	a	b	c	λ

->*q0	q0	Φ	Φ	q1
*q1	Φ	q1	Φ	q2
*q2	Φ	Φ	q2	Φ

Note: λ is also considered as one of the input symbols in λ-NFA as the states make a state transition on λ also.
λ-Closure:
λ-Closure(q0):{q0,q1,q2}
λ-Closure(q1):{q1q2}
λ-Closure(q2):{q2}
DFA should constructed using the λ-Closure:

DFA Transition table:
λ-Closure(q0) will be the start state of DFA

	a	b	c
->{q0q1q2}			

δ({q0q1q2} ,a):
λ-Closure({**δ(**q0,a) U **δ(**q1 ,a) U **δ(**q2,a)})
λ-Closure(q0 U Φ U Φ)
= λ-Closure(q0)
=q0,q1,q2

	a	b	c
->{q0q1q2}	{q0q1q2}		

δ({q0q1q2},b):
λ-Closure({**δ(**q0,b) U **δ(**q1,b) U**δ(**q2,b)})
λ-Closure(Φ U q1 U Φ)
λ-Closure(q1)
={q1q2}

	a	b	c
->{q0q1q2}	{q0q1q2}	{q1q2}	

δ({q0q1q2},c):

λ-Closure(**δ**(q0,c) U **δ(**q1,c) U **δ(**q2 ,c))
λ-Closure(Φ U Φ U q2)
λ-Closure(q2)
=q2

	a	b	c
->{q0q1q2}	{q0q1q2}	{q1q2}	q2

Two new states have emerged ({q1,q2} and q2)and need to be added as a state in DFA

	a	b	c
->{q0q1q2}	{q0q1q2}	{q1q2}	q2
{q1q2}			
q2			

δ({q1q2},a):
λ-Closure(**δ**(q1 ,a) U **δ(**q2,a))
λ-Closure(Φ U Φ}
= Φ

	a	b	c
->{q0q1q2}	{q0q1q2}	{q1q2}	q2
{q1q2}	Φ		
q2			

δ({q1q2},b):
λ-Closure(**δ(**q1,b) U **δ(**q2,b))
λ-Closure(q1 U Φ)
λ-Closure(q1)
={q1q2}

	a	b	c
->{q0q1q2}	{q0q1q2}	{q1q2}	q2
{q1q2}	Φ	{q1q2}	
q2			

{q1q2} on input 'c':
λ-Closure({q1 on input 'c'} U {q2 on input 'c' })
λ-Closure(Φ U q2)
λ-Closure(q2)
=q2

	a	b	c
->{q0q1q2}	{q0q1q2}	{q1q2}	q2
{q1q2}	Φ	{q1q2}	q2
q2			

δ(q2,a):
λ-Closure(**δ(**q2,a))
λ-Closure(Φ)
=Φ

	a	b	c
->{q0q1q2}	{q0q1q2}	{q1q2}	q2
{q1q2}	Φ	{q1q2}	q2
q2	Φ		

δ(q2 , b:
λ-Closure(**δ(**q2 ,b)
λ-Closure(Φ)
=Φ

	a	b	c
->{q0q1q2}	{q0q1q2}	{q1q2}	q2
{q1q2}	Φ	{q1q2}	q2
q2	Φ	Φ	

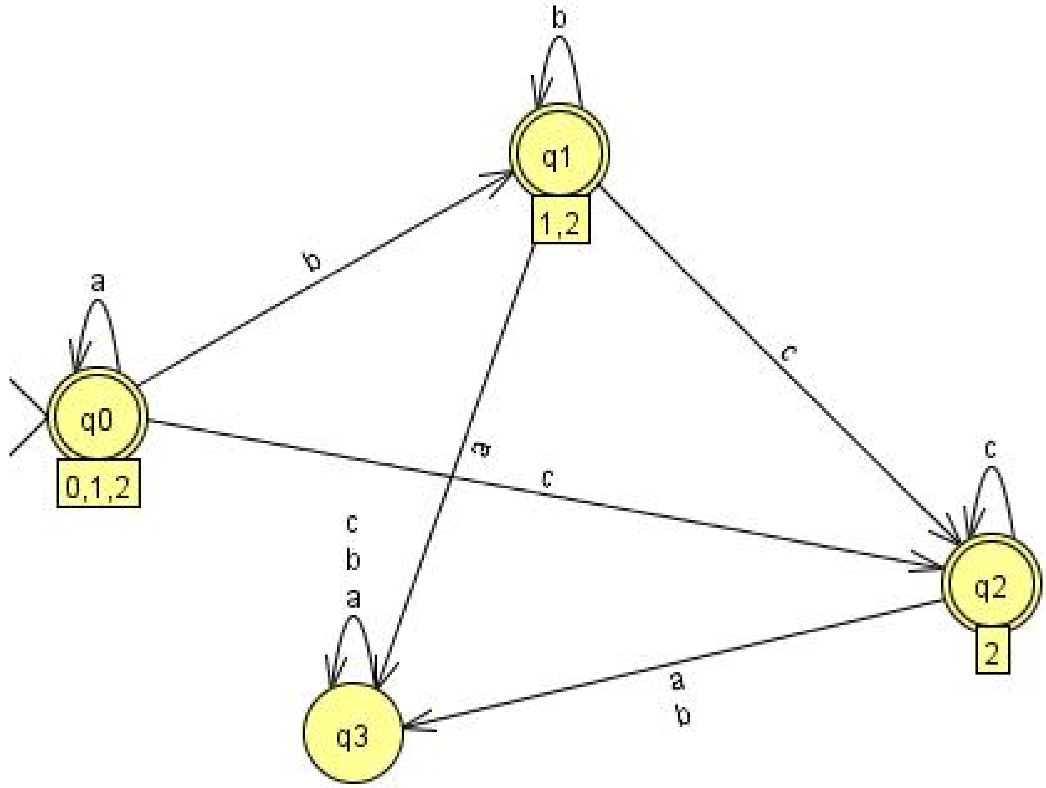
δ(q2 ,c):
λ-Closure(**δ(**q2 , c)
λ-Closure(q2)
=q2

	a	b	c
->{q0q1q2}	{q0q1q2}	{q1q2}	q2
{q1q2}	Φ	{q1q2}	q2
q2	Φ	Φ	q2

Since Φ is also emerged as one of the state , Φ should also be included in DFA and it is treated as dead state or trap state

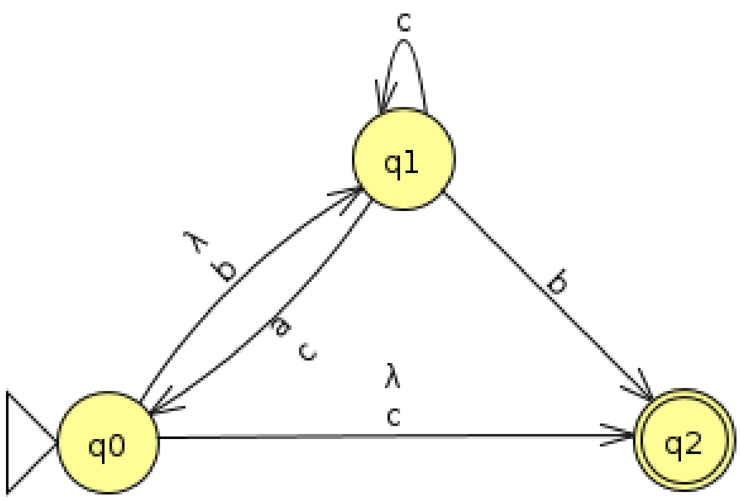
	a	b	c
->*{q0q1q2}	{q0q1q2}	{q1q2}	q2
*{q1q2}	Φ	{q1q2}	q2
*q2	Φ	Φ	q2
Φ	Φ	Φ	Φ

The equivalent transition diagram is shown below:
Rename the states as:
{q0q1q2} as q0, {q1q2} as q1, q2 as q2, Φ as q3


b
q1
1,2
a
b
c
q0
0,1,2
a
c
c
c
b
a
c
q2
2
q3
a
b

Example 2:

Convert the following λ- NFA to its equivalent DFA



NFA Transition table:

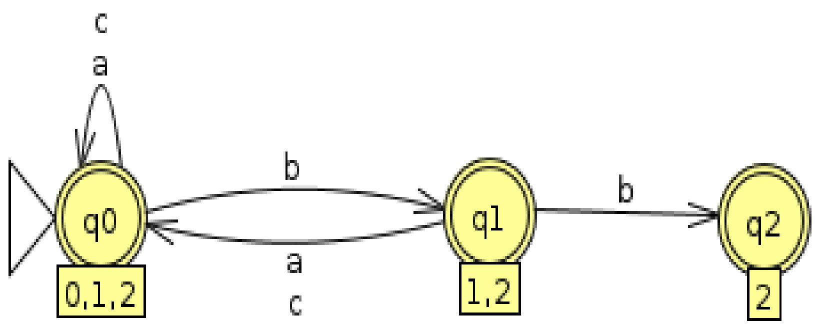
	a	b	c	λ
q0	Φ	q1	q2	{q1,q2 }
q1	q0	q2	{q0,q1}	Φ
q2	Φ	Φ	Φ	Φ

Find the λ closure of q0,q1,q2
λ closure(q0)= q0,q1,q2
λ closure(q1)= q1
λ closure(q2)=q2

DFA transition table:

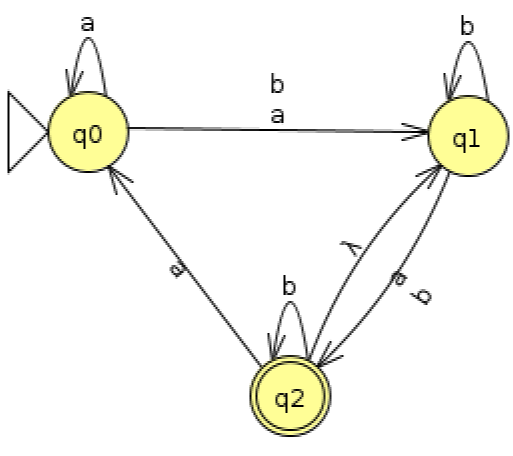
	a	b	c
-> *{q0q1q2}	{q0q1q2}	{q1q2}	{q0q1q2}
*{q1q2}	{q0q1q2}	q2	Φ
*q2	Φ	Φ	Φ
Φ	Φ	Φ	Φ

State transition diagram of the dfa for a given NFA :



Example 3:

Convert the following λ-NFA to its equivalent DFA



λ-NFA State transition table :

	a	b	λ
->q0	{q0q1}	q1	Φ
q1	q2	{q1q2}	Φ
*q2	q0	q2	q1

find the λ - closure for q0,q1,q2

λ - closure(q0)=q0
λ - closure(q1)=q1
λ - closure(q2)={q1q2}

DFA transition table:

	a	b
->q0	{q0q1}	q1
q1	{q1q2}	{q1q2}
{q0q1}	{q0q1q2}	{q0q2}
*{q1q2}	{q0q1q2}	{q1q2}
*{q0q1q2}	{q0q1q2}	{q1q2}

The state transition diagram of DFA for the given λ-NFA