

AUTOMATA FORMAL LANGUAGES AND LOGIC



Lecture notes on **Converting Finite Automata to Regular Expression**

Prepared by

**Ms. Kavitha K N
Assistant Professor**

**Ms. Preet Kanwal
Associate Professor**

**Department of Computer Science & Engineering
PES UNIVERSITY**

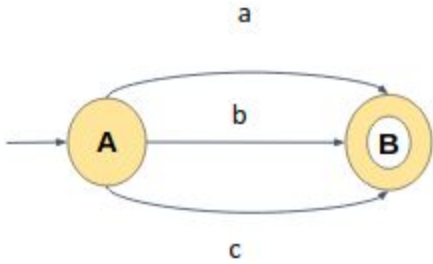
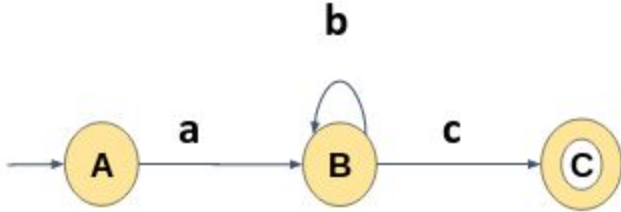
(Established under Karnataka Act No.16 of 2013)

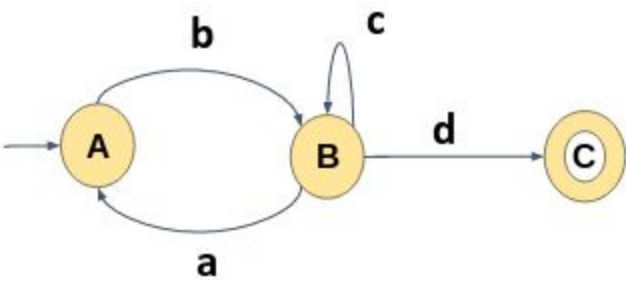
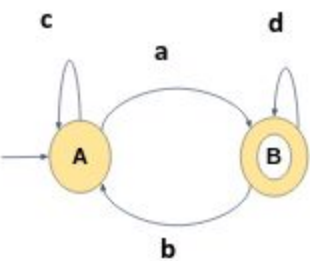
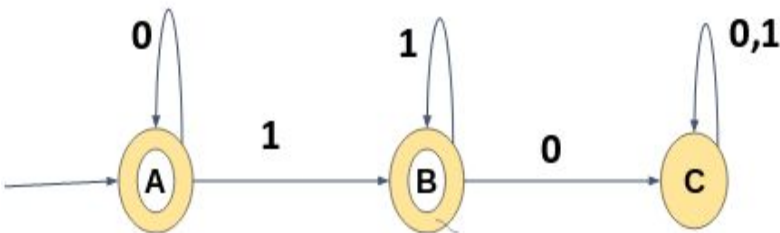
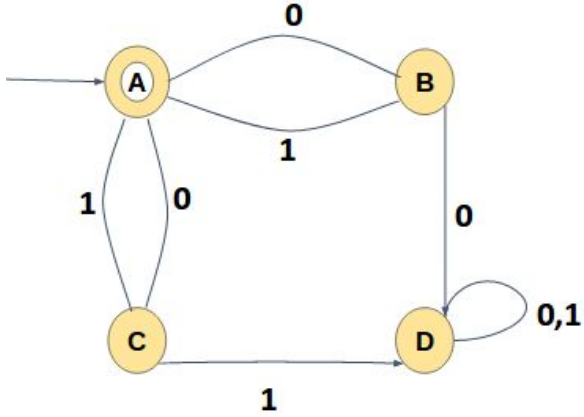
100-ft Ring Road, BSK III Stage, Bangalore - 560 085

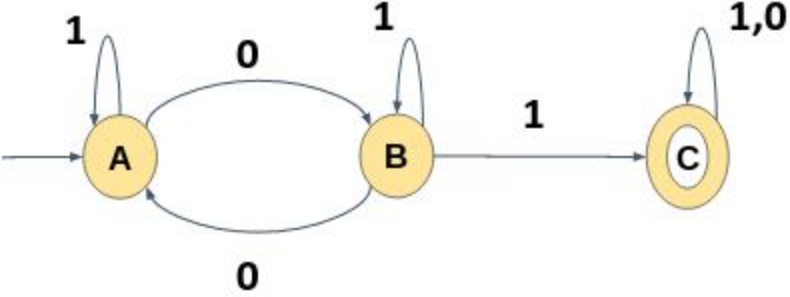
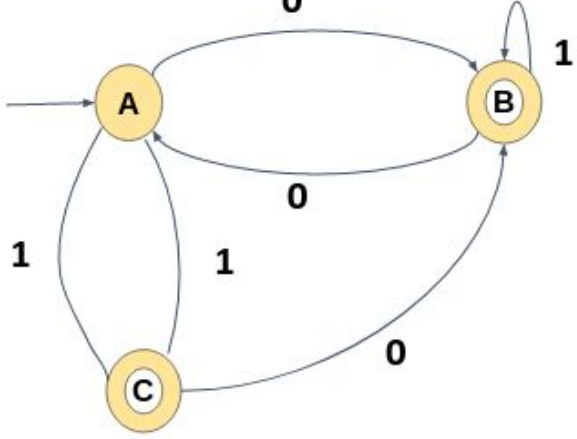
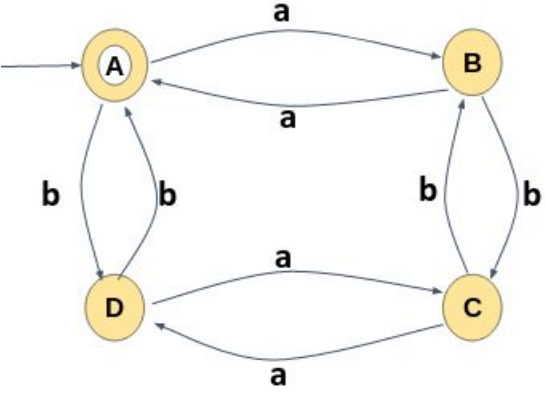
Table of Contents:

Section	Topic	Page number
1.	Introduction	5
2.	State Elimination method:	5
3.	Example	8

Examples Solved:

#	Problems on Conversion of Finite Automata to Regular expression	Page number
1		8
2		9

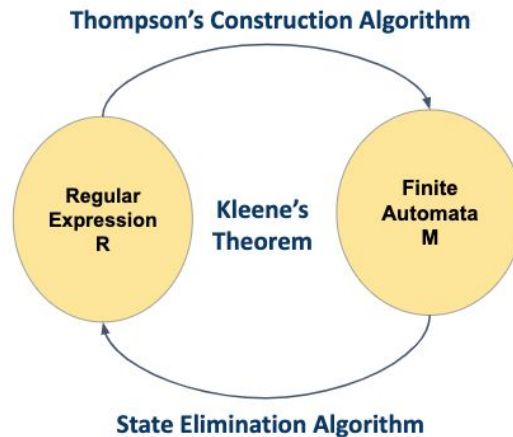
3	 <pre> graph LR Start(()) --> A((A)) A -- b --> B((B)) B -- a --> A B -- c --> B B -- d --> C(((C))) </pre>	10
4	 <pre> graph LR Start(()) --> A((A)) A -- c --> A A -- a --> B(((B))) B -- b --> A B -- d --> B </pre>	11
5	 <pre> graph LR Start(()) --> A(((A))) A -- 0 --> A A -- 1 --> B(((B))) B -- 1 --> B B -- 0 --> C(((C))) C -- "0,1" --> C </pre>	17
6	 <pre> graph LR Start(()) --> A((A)) A -- 0 --> B((B)) B -- 1 --> A A -- 1 --> C((C)) C -- 0 --> A C -- 1 --> D((D)) D -- 0 --> B D -- "0,1" --> D </pre>	21

7	 <pre> graph LR start(()) --> A((A)) A -- 1 --> A A -- 0 --> B((B)) B -- 0 --> A B -- 1 --> B B -- 1 --> C(((C))) C -- "1,0" --> C </pre>	24
8	 <pre> graph LR start(()) --> A((A)) A -- 0 --> B((B)) B -- 0 --> A B -- 1 --> B C(((C))) -- 1 --> A C -- 0 --> B </pre>	27
9	 <pre> graph LR start(()) --> A((A)) A -- a --> B((B)) B -- a --> A A -- b --> D((D)) D -- b --> A B -- b --> C((C)) C -- b --> B D -- a --> C C -- a --> D </pre>	31

1. Introduction:

Given any DFA M , we have to construct a regular expression r (over the alphabet of input symbols of M) satisfying $L(r) = L(M)$.

To construct a regular expression from a DFA, we make use of the State Elimination Algorithm in which we replace each state in the DFA one by one with a corresponding regular expression.



2. State Elimination method:

This method is used for constructing Regular Expression from the Finite Automata. In this method, states will be removed from the automaton, restoring the labels of arcs, so that after removing all the states except first and last state, at the end only a single regular expression is formed.

This single regular expression will be the label on an arc that goes from start state to end state, and this will be the only arc in the automata and the label will be the final regular expression for the given automaton.

This method follows the following simple steps in constructing Regular Expression from the Automaton:

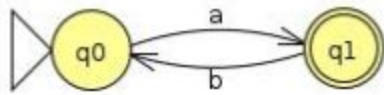
Note: Before eliminating the states, Dead state/ trap states, if any, need to be removed.

Start with an FA for the language L .

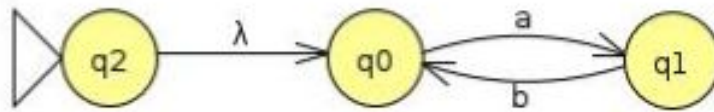
- **Add a new start state q_s to the FA.**

If the start state itself is an accepting state or has any transition in (incoming edge), then create a new start state having no transition in and make a transition from new start state to an old start state with λ transition.

Example:

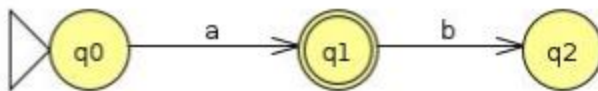


The start state q_0 have incoming edge, This will be modified as below after adding new start state q_2 with λ transition

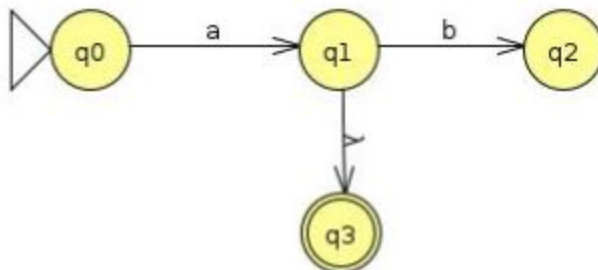


- **Add a new accept state q_f to the FA**

If there exists any outgoing edge from the final state, then create a new final state having no outgoing edge from it.

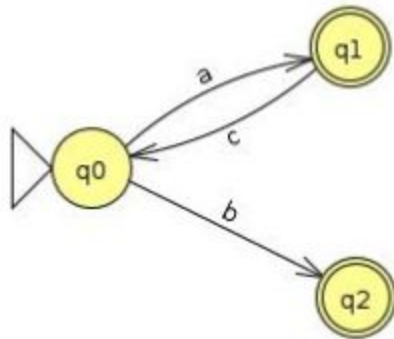


The final state q_1 has an outgoing edge, so create a new final state q_3 and make q_1 as a non final state and have λ transition from q_1 to q_3 as shown below.

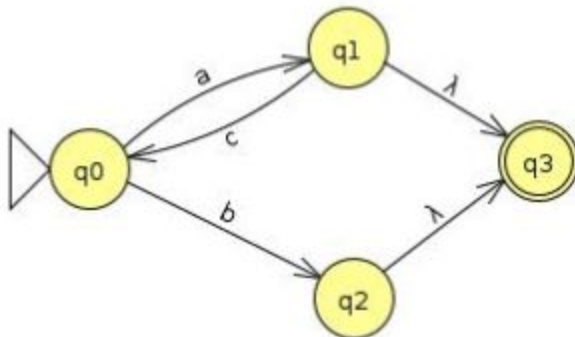


- Add λ -transitions from each original accepting state to q_f , then mark them as not accepting.

If there exists multiple final states in the FA, create an equivalent automaton that has a single accepting state with no transitions out.



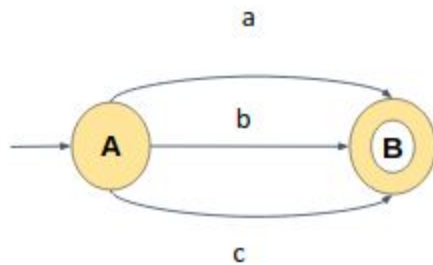
The above automata have two accepting states, create a new single accepting state q_3 and have a λ transition from q_1 and q_2 to q_3 by changing them to non final state. The automata after introducing new final state is shown below:



- Repeatedly remove states in any order other than start state and final state from the FA by “shortcutting” them until only two states remain: start and final state .
- The transition from q_s to q_f is then a regular expression for the FA.

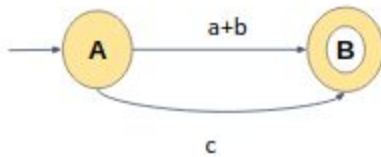
Example:

1) Conversion of Finite Automata to Regular Expression

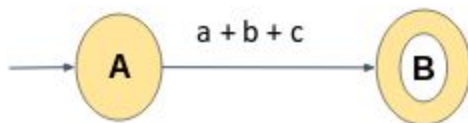


Solution:

Start state does not have any incoming edge and Accepting state does not have any outgoing edge so there is no need to introduce any new start or final state, there are three direct transition for state A to state B with the input a, or with the input b, or with the input c, So the edges can be replaced by a single edge as shown below, input a or b can be replaced with a single edge with the expression $a+b$.

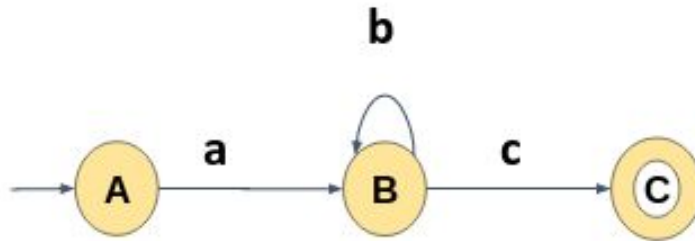


then we have two edges (transition), one for $(a+b)$ and the other for c can be replaced with a single edge with the label (expression $(a+b+c)$) as shown below:



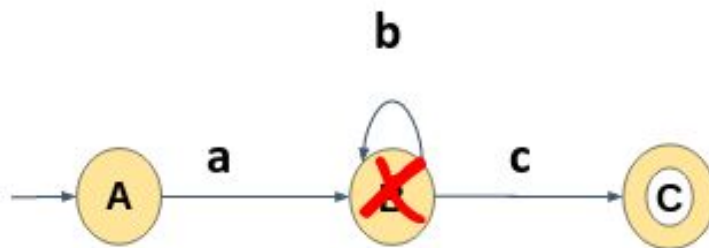
The final regular expression for the given automaton is **$(a+b+c)$**

2) Convert the following FA to RE



For the given DFA there is no incoming edge to the start state and no outgoing edge from the final state, so we can start eliminating the intermediate states (states between start and final state)

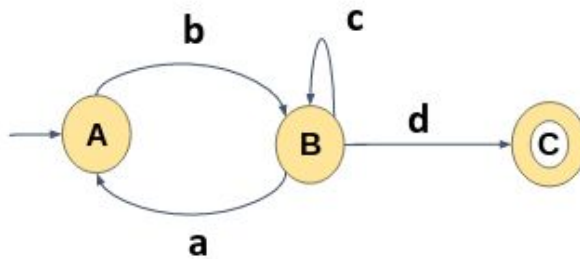
The only state between start state A and the final state C is B, so let us eliminate the state B



there is a path going from state A to state C through B, and State B has a self loop, So after deleting state B, there will be a direct path from state A to state C having the cost (ab^*c)

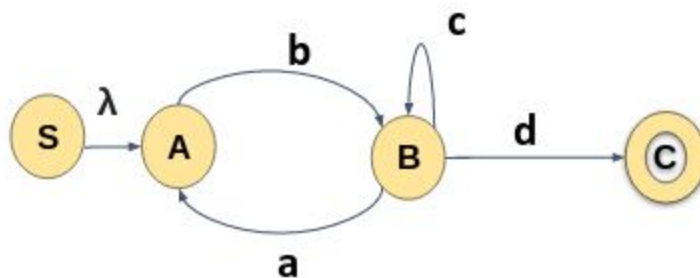
The final Regular Expression is **(ab^*c)**

3) Convert the following Finite Automata to Regular Expression



Start state A is having an incoming transition, so we create a new start state with the name S, (from now state S will be the start state)

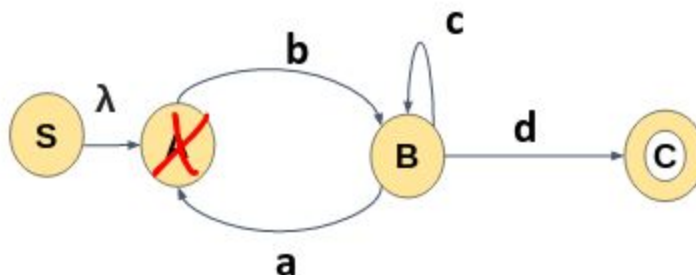
Make a transition from new start state to state A with λ as shown



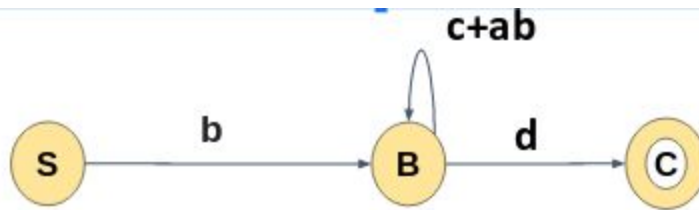
We first eliminate state A,

-> there is a path going from state S to state B through state A, so after eliminating state A there will be direct path from state A to state B with the cost/expression ' $\lambda.b$ = b'

-> There is a loop on state B through state A, So after deleting state A, state B will have direct path with the cost / expression $ab+c$

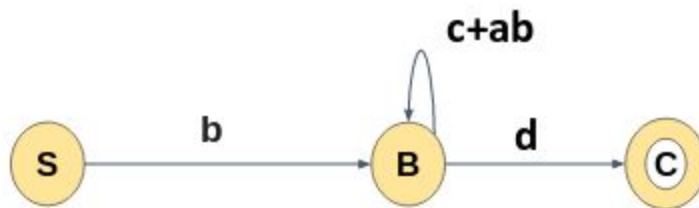


After Eliminating state A, the automata will look like this:

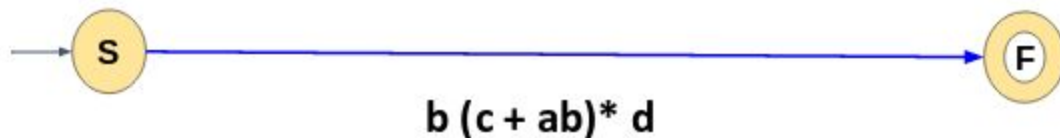


Next, we will eliminate state B, the only intermediate state

-> there is a path from state S to state C through B and B have self loop, so after eliminating state B there will be a direct path from state A to state C with the cost / expression $b(c+ab)^*d$

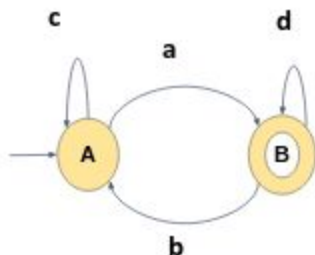


After eliminating state B the automata will look like this, there are no intermediate states to eliminate

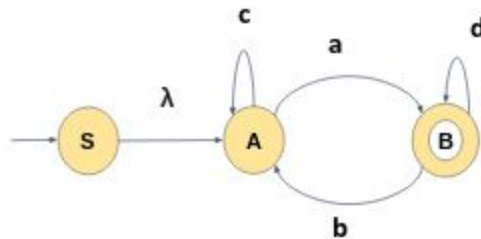


The final regular expression for the given finite automata is $b(c+ab)^*d$

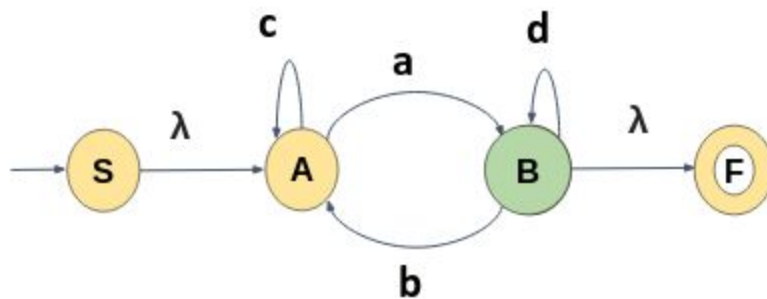
4) Convert the following Finite Automata to Regular Expression



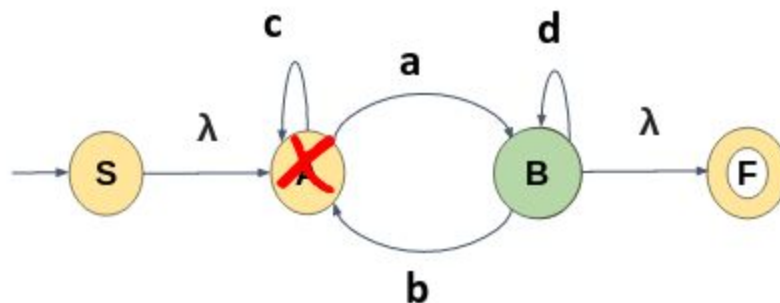
Start state A is having an incoming transition, so we create a new start state with the name S, (from now state S will be the start state). Make a transition from new start state to state A with λ as shown



Final state B has an outgoing edge so, we create a new final state F and make the state B as non final state, also have a transition from state B to state F with λ as shown below:



First eliminate state A,

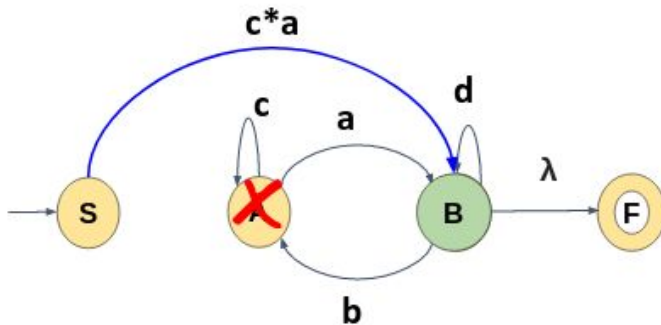


-> There is a path going from state S to state B through state A and state A have a self loop.

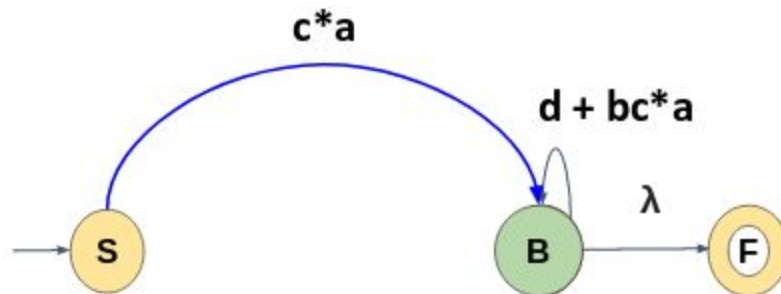
So after deleting state A, We will have a direct path from State S to state B having the cost $\lambda . c^* a = c^* a$.

The direct path what we get after deleting A is shown below

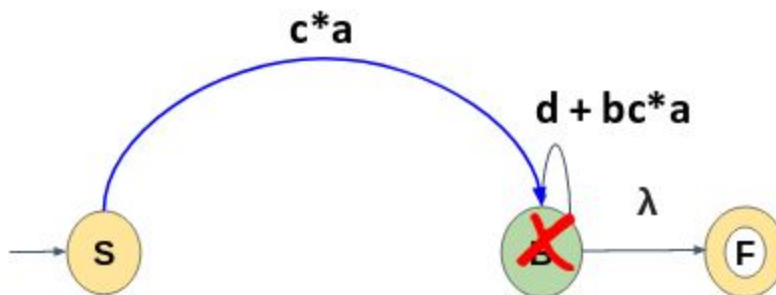
-> It also have a loop on state B through state A , B also have a self loop, and state A also have a self loop



So after deleting state A , we get a direct self loop on state B having the expression $d + bc^*a$ as shown below:



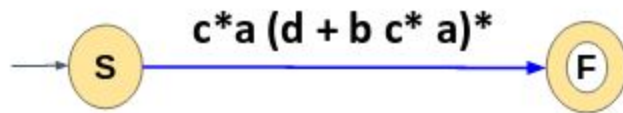
Next eliminate state B,



There is a path from state S to state F through B, and B have a self loop

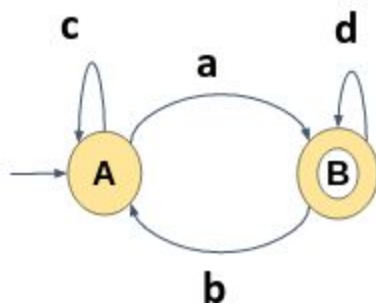
So after eliminating there will be a direct path from state S to state F with the cost $c^*a.(d+bc^*a)^*.\lambda = c^*a.(d+bc^*a)^*$

We have only two states remaining and the label on the path from state S to state F is the final regular expression.

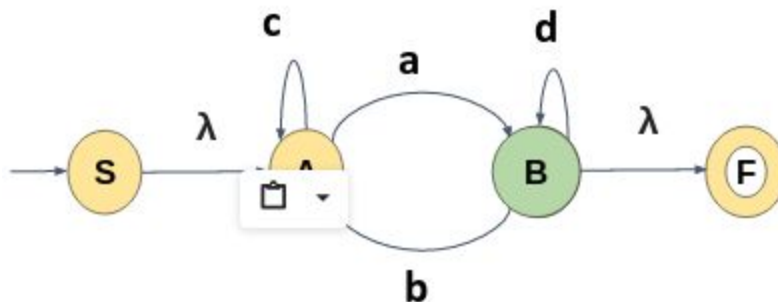


The final regular expression for the given Automaton is : $c^*a (d + b c^* a)^*$

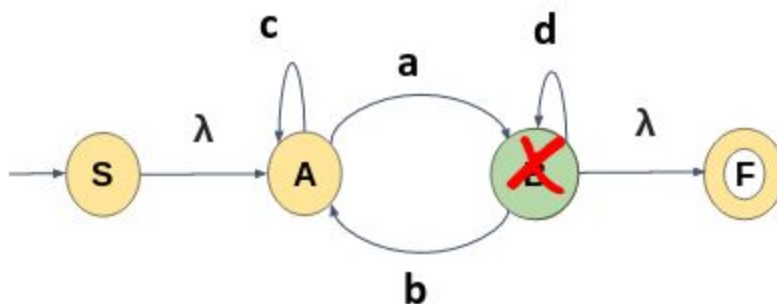
-> Consider the same example but different order in deleting the states



Adding a new start state and final state remain the same.



Now first eliminate state B instead of state A,



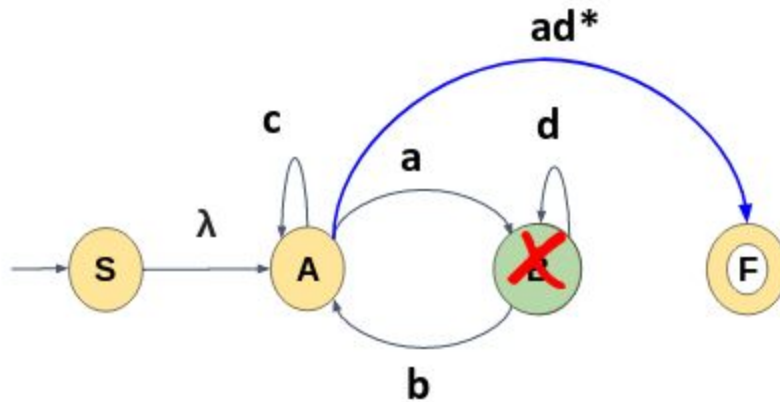
-> There is a transition from state A to state F through B and B has a self loop,

So, after eliminating state B there will be a direct path from state A to state F having a cost $(ad^* \cdot \lambda) = ad^*$

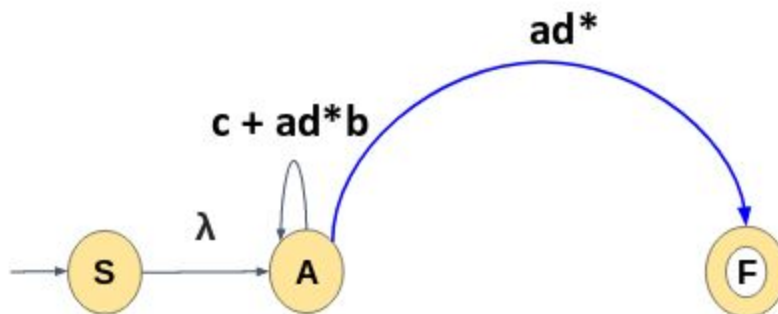
Direct path from state A to state F is shown below

-> there is a loop on state A through state B, A also have a self loop, and B also have a self loop,

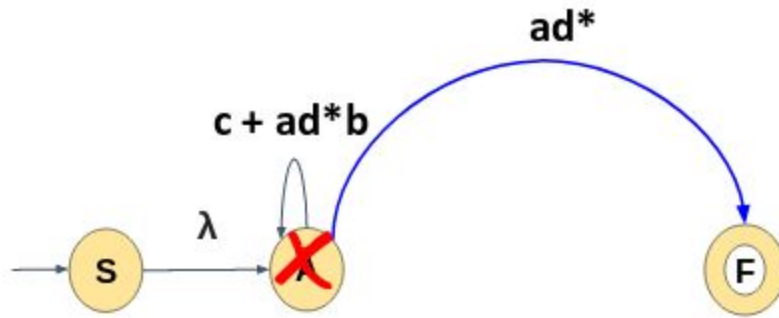
so after eliminating state B there will be a direct self loop on state A having the cost $c^* + ad^*b$



Path after deleting state B is shown below along with the expressions

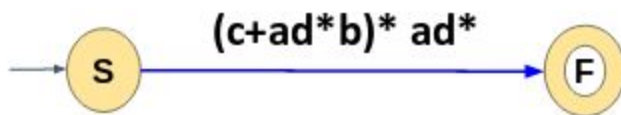


-> next eliminate the only intermediate state A (state between start state and accepting state)



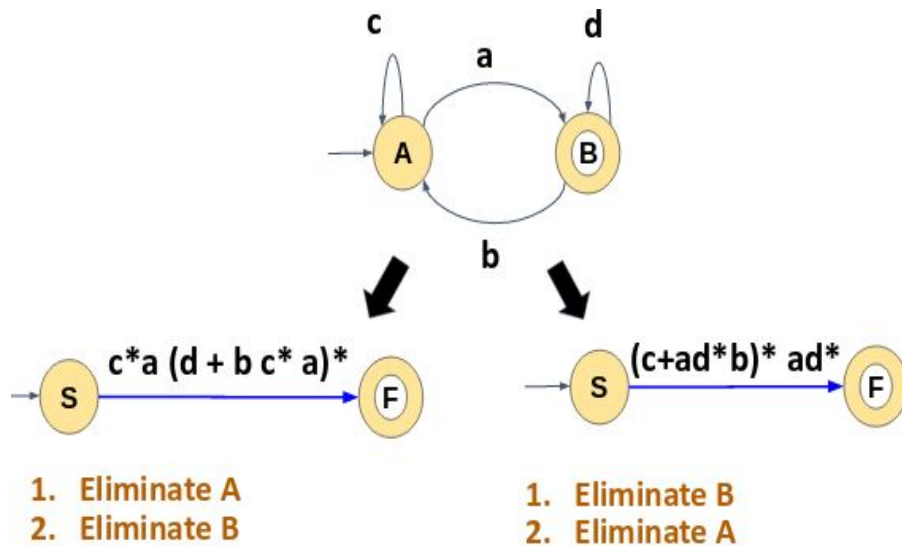
There is a path from state S to state F through state A and state A have a self loop, so after deleting state A, we get a direct path from state S to state F having the cost λ .
 $(c+ad*b)^* ad^* = (c+ad*b)^* ad^*$

We have only two states remaining and the label on the path from state S to state F is the final regular expression

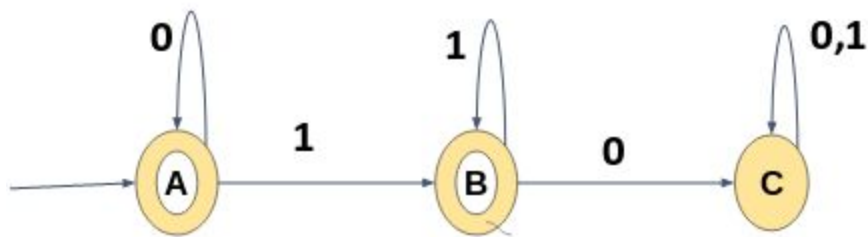


The final expression is $(c+ad*b)^* ad^*$

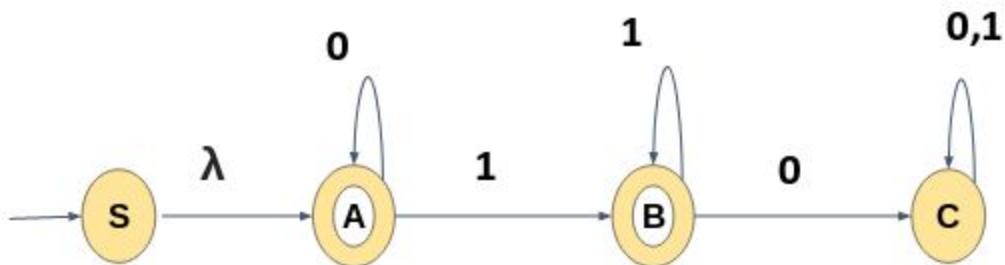
The below are the two different regular expression obtained based on the order of eliminating the states



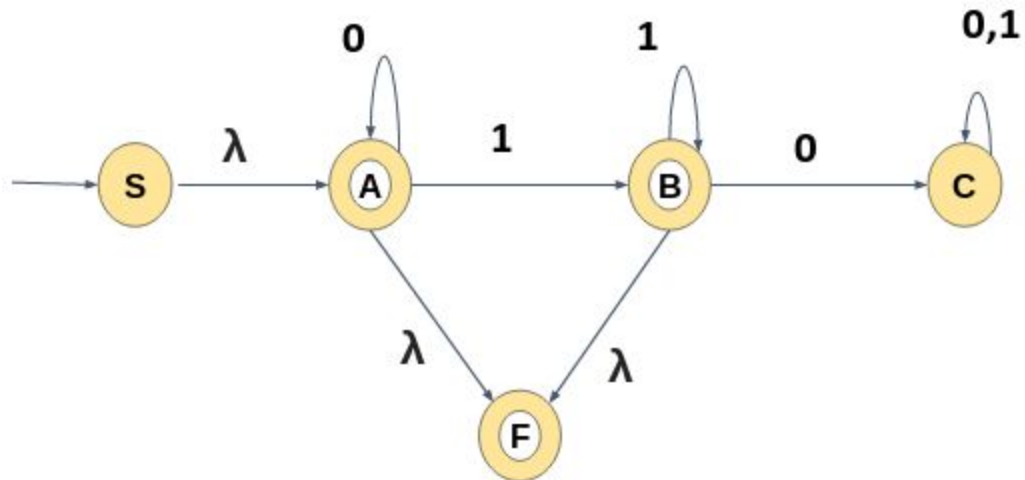
5) Convert the following Finite Automata to Regular Expression



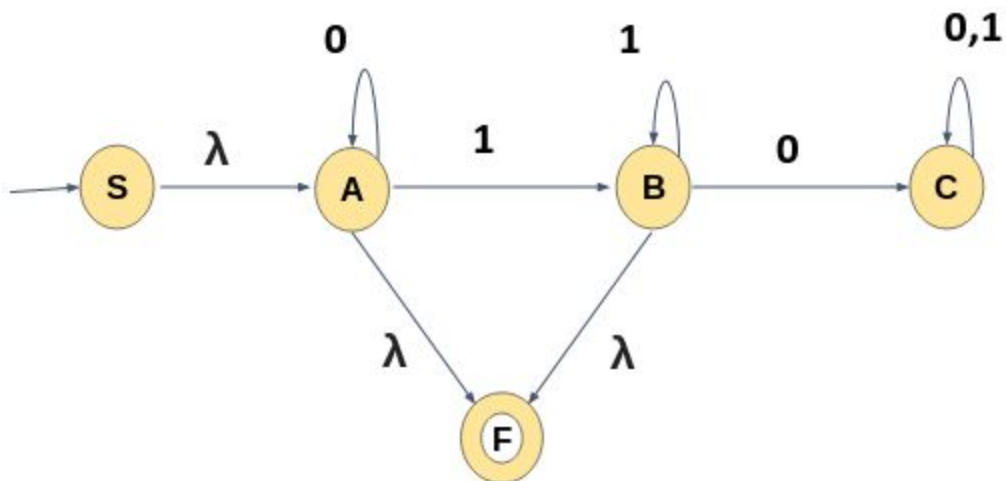
Start state A is having an incoming transition, so we create a new start state with the name S, Make a transition from new start state to state A with λ as shown:



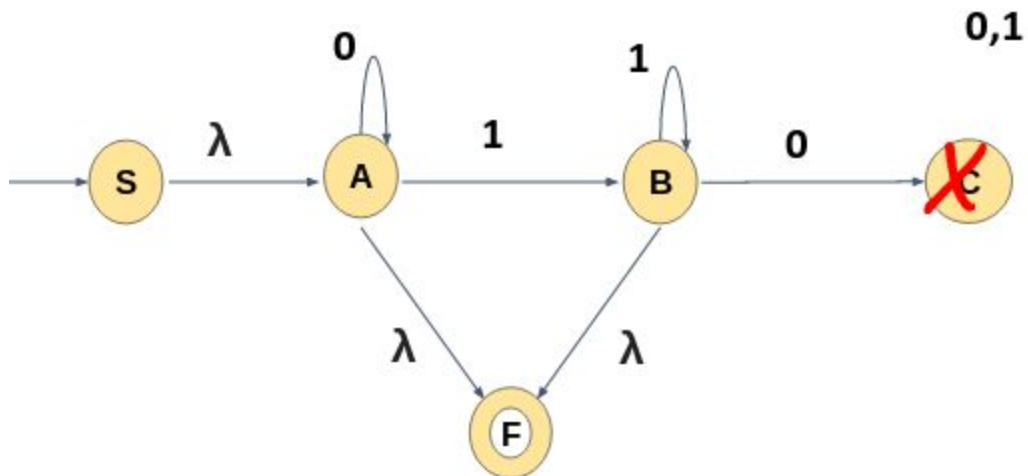
Since there are two final states and an outgoing edge, we create an equivalent automaton that has a single accepting state with no transitions out and Make a transition from old final states to new final state with λ .



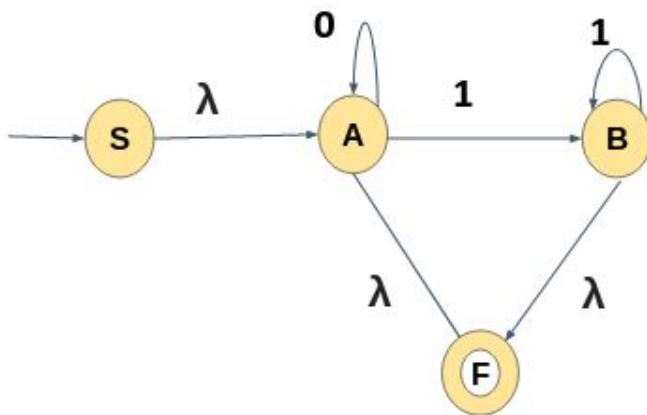
Old final states are made as non final states. once we have a single start state and final state, eliminating the intermediate state.



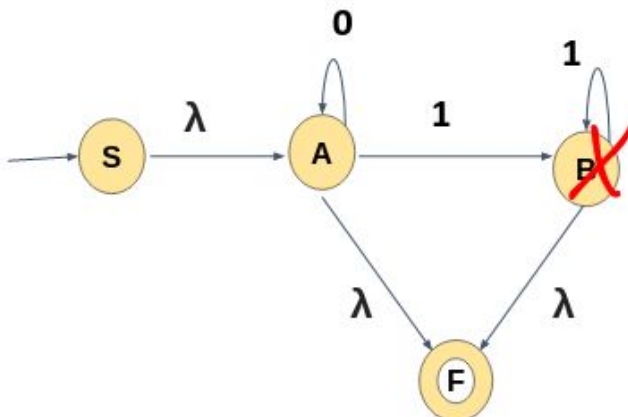
First eliminate state C. Before choosing the order of states to eliminate, it is better to check whether we have any dead /trap state, if we have one eliminate that state first. Here State C is a dead state, so delete state C and its corresponding transition.



The Automata after deleting state C will look like below:

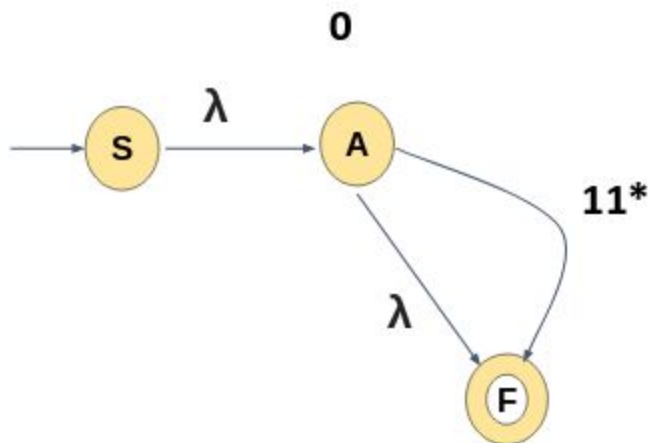


->Next eliminate state B.

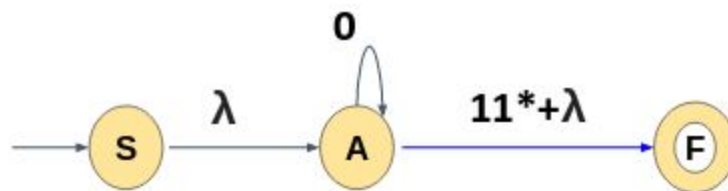


There is a transition from state A to state F through state B, and state B have a self loop. so after deleting state B, there will be a direct path from state A to state F with the cost **11***.

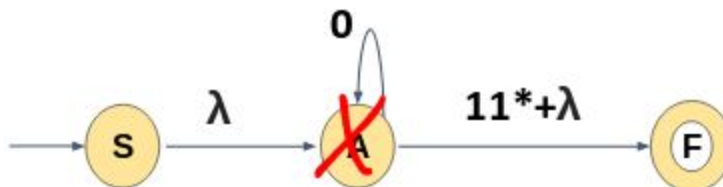
Now, there is two path from state A to state F, one is through λ and the other one is through 11^* . we can represent this with a single path with the expression $\lambda + 11^*$



DFA will look below this after the eliminating state B



-> Next will eliminate state A(the only intermediate state)



There is a path form state S to state F through state A and State A have a self loop, so after eliminating state A, we get a direct path from state S to state F wit the cost λ .

$$0^* 11^*. \lambda = 0^* 11^*$$

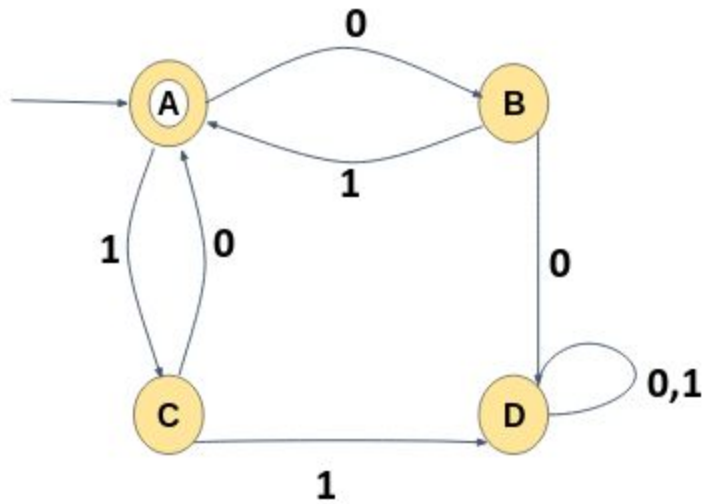
$$11^* = 1+$$

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

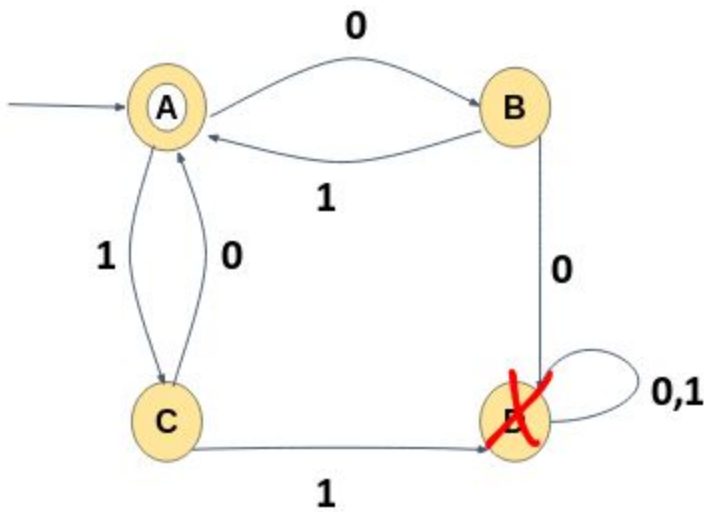
$$\text{so, } 0^*(11^*+ \lambda) = 0^*1^*$$

so the final regular expression for the given DFA is 0^*1^*

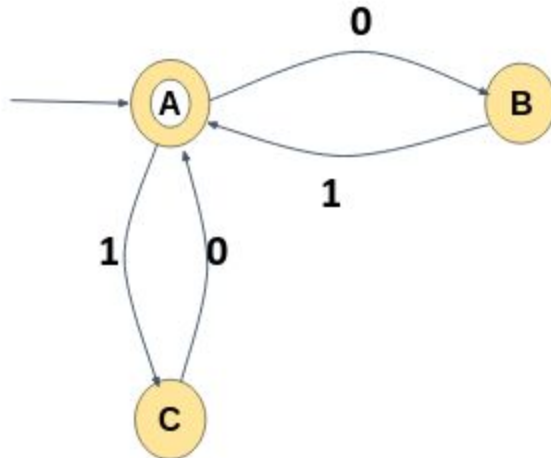
6) Convert the following Finite Automata to Regular Expression



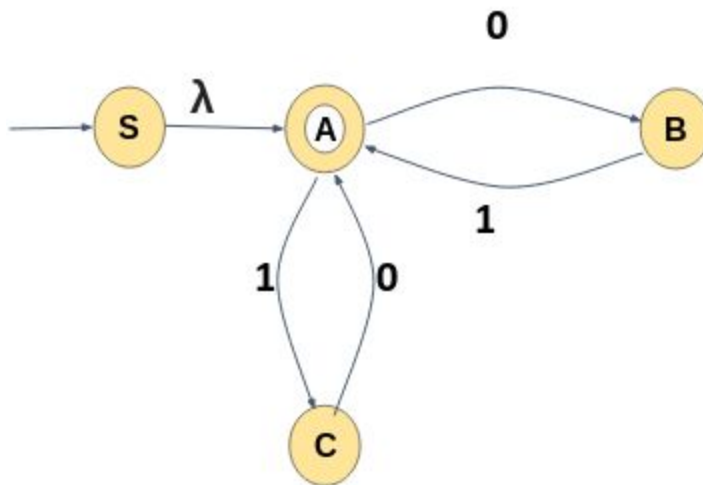
State D is a dead / trap state, so we first eliminate state D and its corresponding transition.



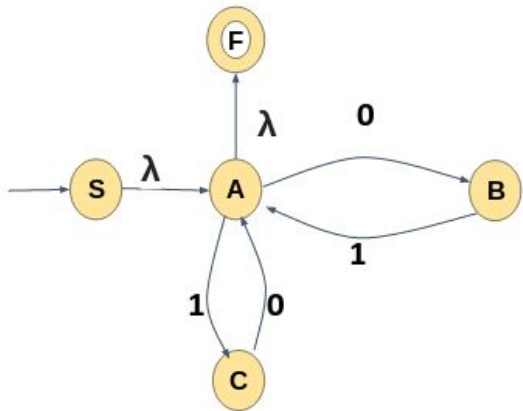
After deleting the dead state the automata will look like below.



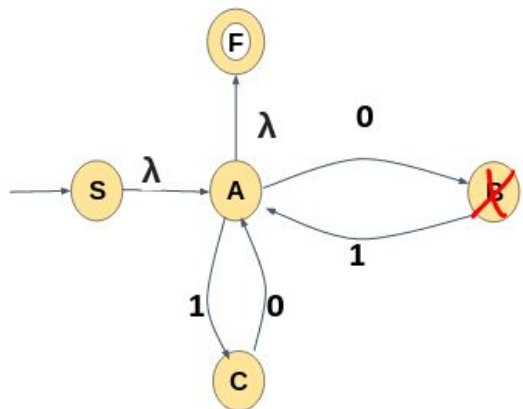
Start state A is having an incoming transition, so we create a new start state with the name S, and Make a transition from new start state S to state A with λ as shown.



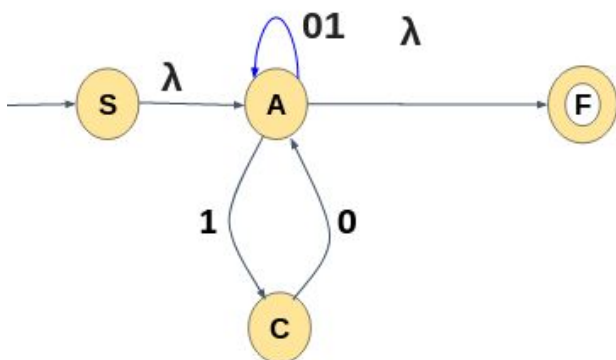
The accepting state have outgoing edge, so create a new final state with the name F and have a transition from old final state A to new final state F with λ . and make the old final state A an non final state



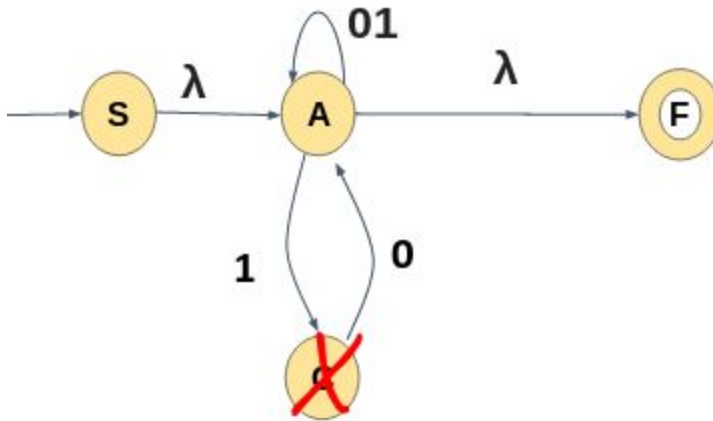
->Next eliminate state B,



There is a loop on state A to state A through B,
 After eliminating state B, State A will have a direct self loop with the cost 01
 After deleting state B automata will look the following:

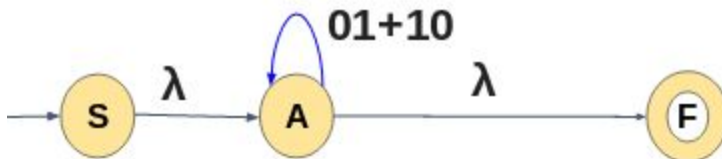


Next delete state C

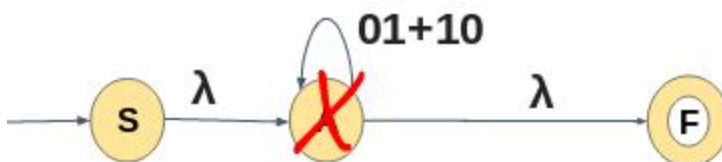


There is a loop on state A through state C, and State A also have a self loop
 so after eliminating state C there will be a direct self loop on state A having the cost
10+01

After deleting state C automata will look the following:



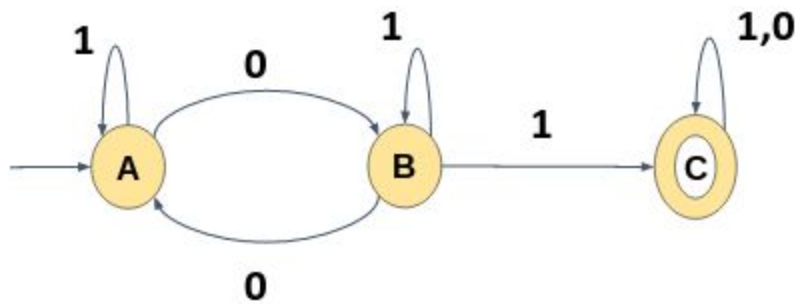
-> Next will eliminate state A(the only intermediate state)



There is a path form state S to state F through state A and State A have a self loop, so
 after eliminating state A, we get a direct path from state S to state F wit the cost
(01+10)*

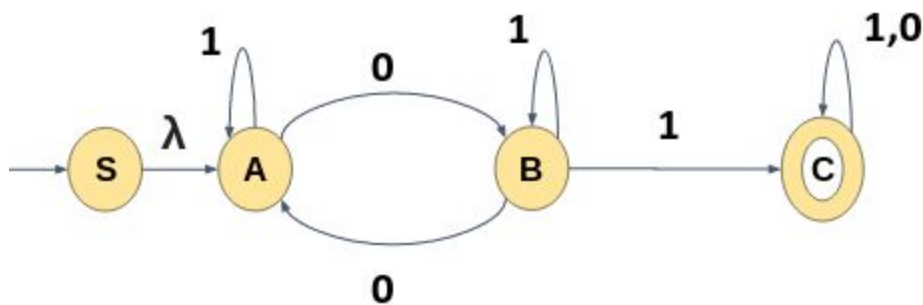
So the final expression after eliminating all the intermediate states is: **(01+10)***

7) Convert the following FA to RE

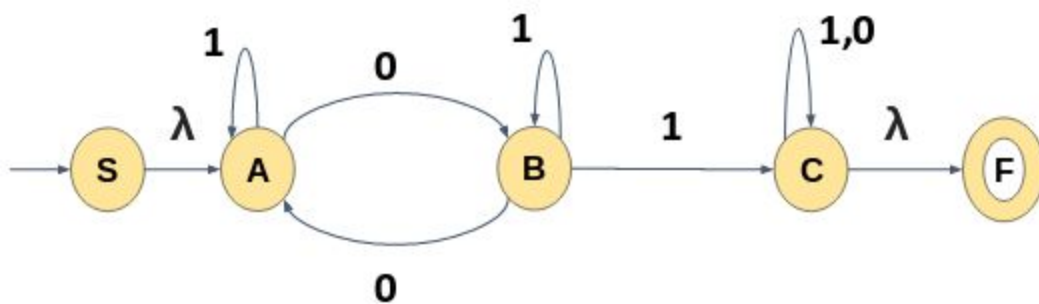


There are no dead / trap states,

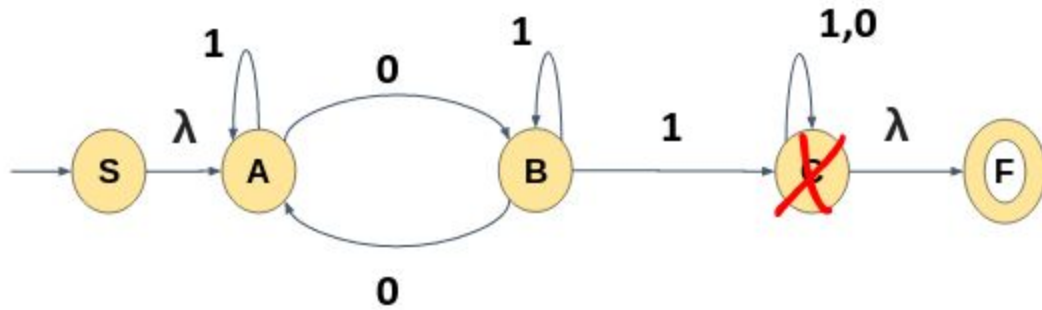
Start state A is having an incoming transition, so we create a new start state with the name S, Make a transition from new start state S to state A with λ as shown



The accepting state have outgoing edge, so create a new final state with the name F and have a transition from old final state C to new final state F with λ . and make the old final state C as a non final state.

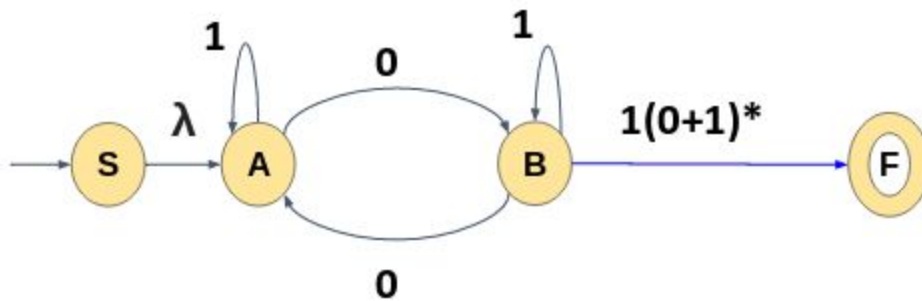


-> Eliminate state C

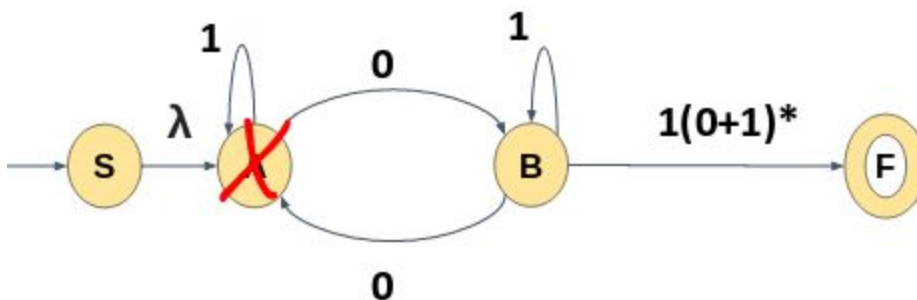


there exists a path from state A to state F through state S and state C have a self loop,
 So after deleting state C we will have a direct path from state B to state F with the
 cost $1 \cdot (0+1)^* \cdot \lambda = 1(0+1)^*$

After eliminating state C the Automata look like the following

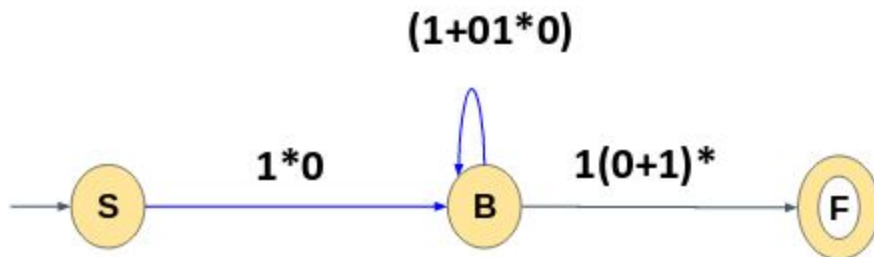


Next eliminate state A

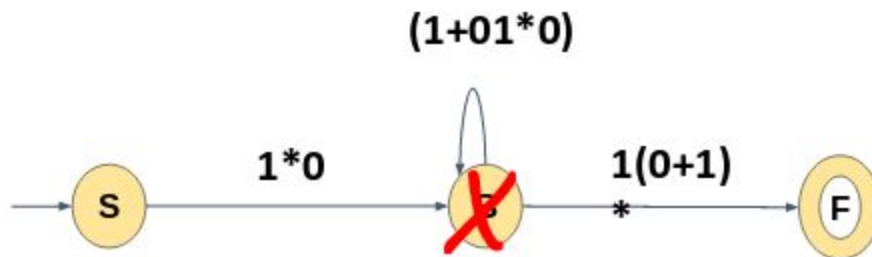


There exists a path from state S to state B through state A and state A have a self loop
 So after deleting state A will have a direct path from state A to state B with the cost
 $\lambda \cdot 1^* 0 = 1^* 0$

State B we have a loop on state B through A, B also have a self loop so,
After eliminating state A state B have a self loop with the cost **$(1+01^*0)$**



-> Next will eliminate state B(the only intermediate state)

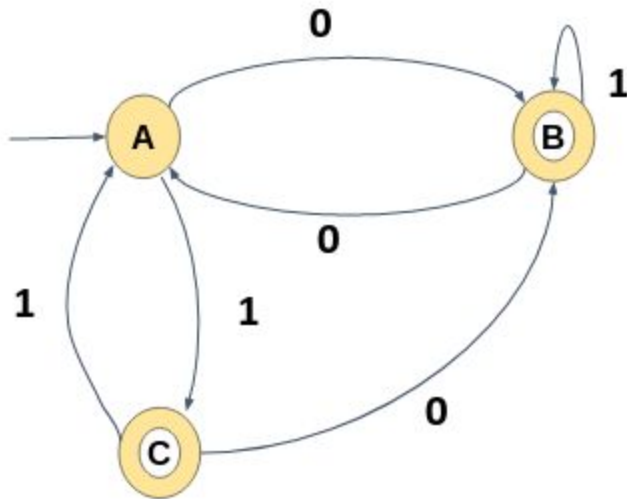


There is a path from state S to state F through state B and State B have a self loop, so
after eliminating state B, we get a direct path from state S to state F with the cost
 $1^*0.(1+01^*0)^*1(0+1)$

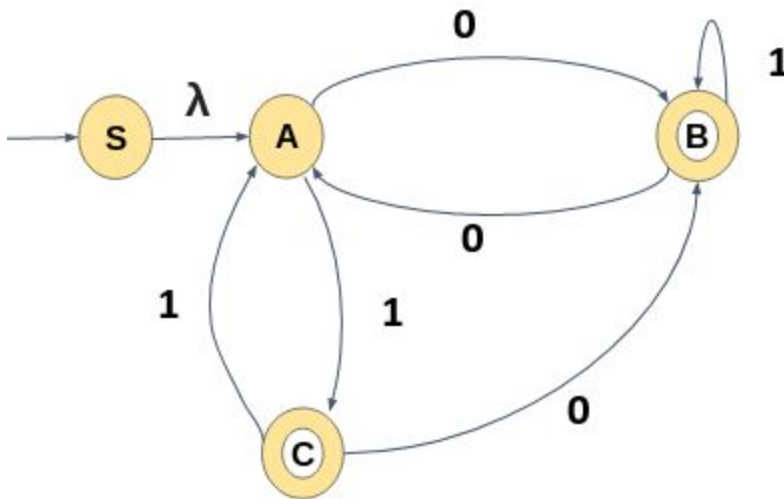
So the final expression after eliminating all the intermediate states :

$1^*0(1+01^*0)^*1(0+1)^*$

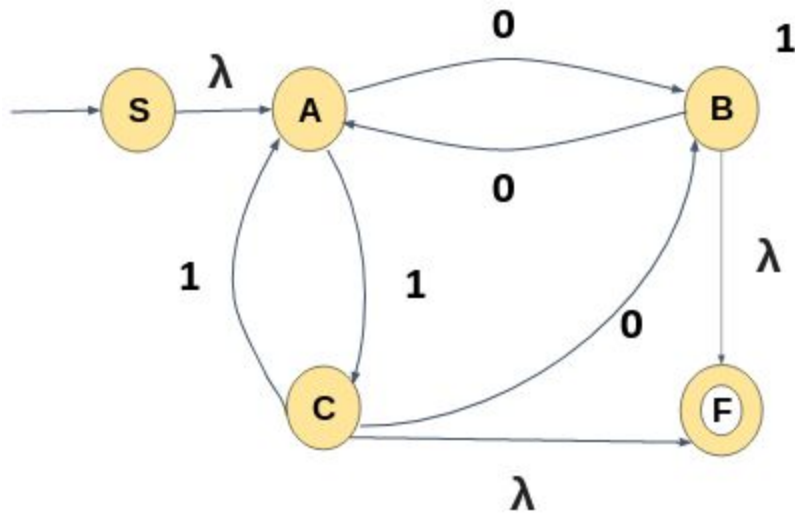
8) Convert the following FA to RE



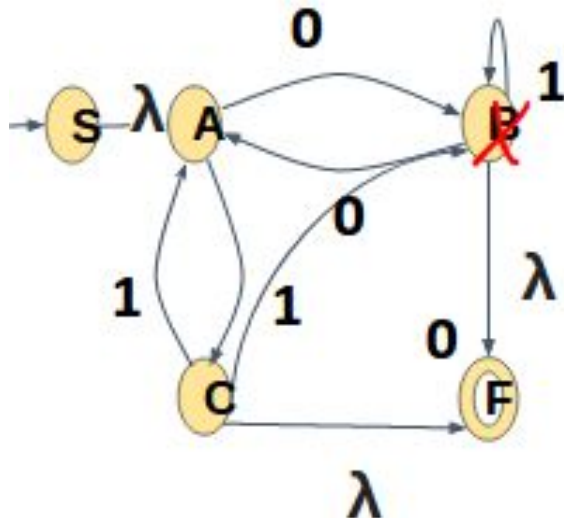
Start state A is having an incoming transition, so we create a new start state with the name S, Make a transition from new start state S to state A with λ as shown.



There are two accepting states and it has outgoing edge, so create a new final state with the name F and have a transition from old final state A and B to new final state F with λ . and make the old final state C and B as a non final state



Eliminate state B,

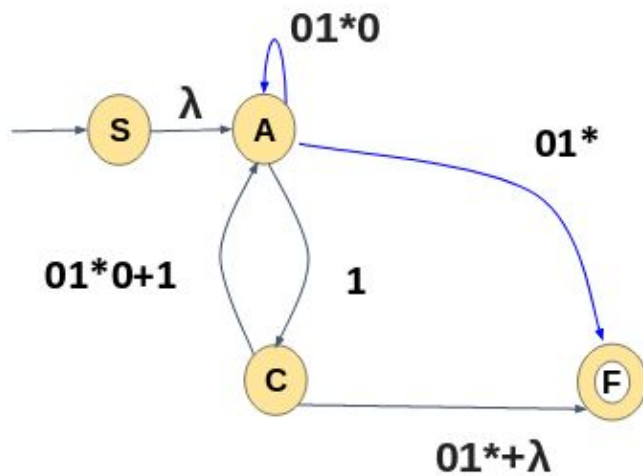


-> there is a loop on state A through B, so after deleting state B we will have a direct self loop on B with the cost **01*0**

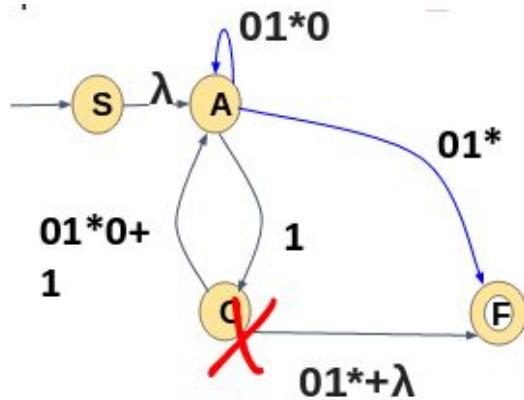
-> There is a direct path from state C to state A and indirect path to state A through state C and state B. so after eliminating B there will be a direct path from state C to state A with the cost **1+01*0**.

-> there is a path from state A to state F through B, so after deleting state B there will be a direct path from state A to F with the case **01*.λ = 01***

After eliminating state B, Automata looks like the following



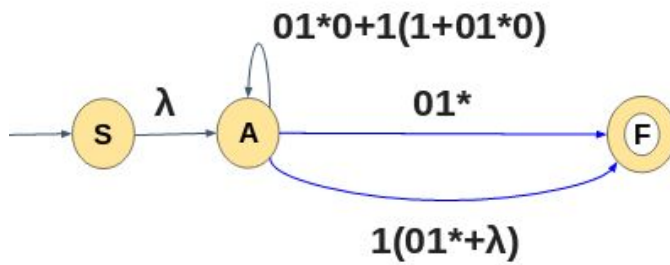
Next eliminate state C,



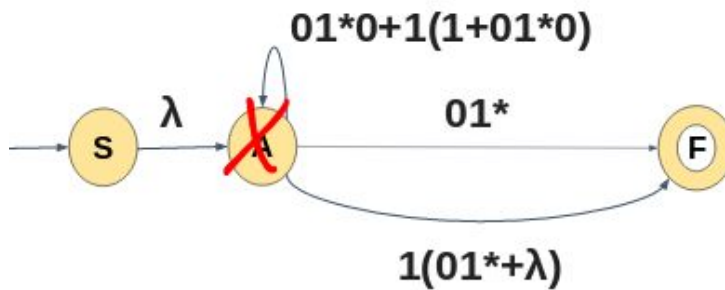
-> There is a loop on state A through state C and A have a self loop, so after deleting state C we will have a direct self loop on A with the cost $01^*0+1(1+01^*0)$

-> There is a path from state A to state F through C, so after deleting state C there will be a direct path for state A to state F having the cost $1(01^*+\lambda)$

After deleting state C the automata looks like:



Next will eliminate state A(the only intermediate state)



There is a path from state S to state F through state A and State A have a self loop and have two different path from state A to state F , so after eliminating state A, we get a direct path from state S to state F with the cost $(01^*0+1(1+01^*0))^* (01^*+1(01^*+\lambda))$

The final regular expression for the given automata is:

$$RE = (01^*0 + 1(1 + 01^*0))^* (01^* + 1(01^* + \lambda))$$

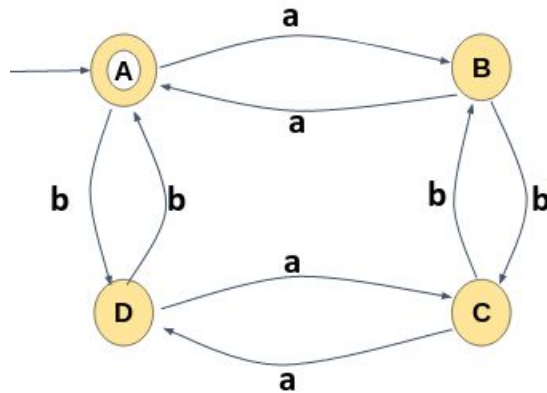
$$= (01^*0 + 11 + 101^*0)^* (01^* + (01^* + 1))$$

$$= (101^*0 + 01^*0 + 11)^* (01^*(\lambda + 1) + 1)$$

$$= (01^*0(1 + \lambda) + 11)^* (01^*(\lambda + 1) + 1)$$

$$RE = (01^*0(1 + \lambda))^* (11)^* (01^*(\lambda + 1) + 1)$$

9) Convert the following FA to RE

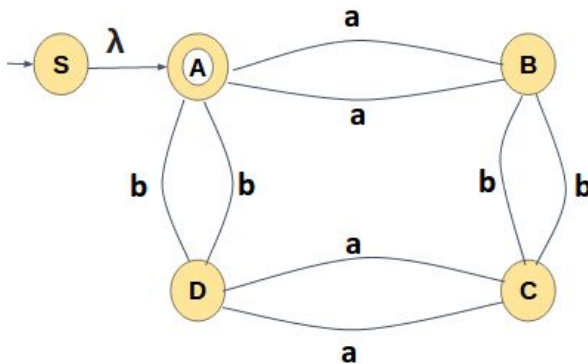


10)

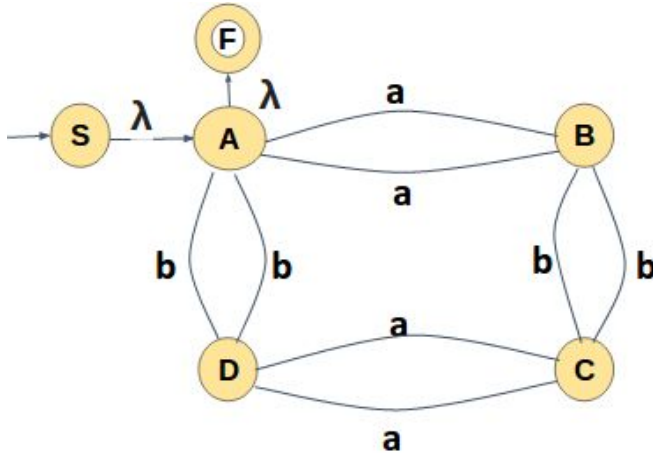
order of state elimination(B,D,C,A)

There are no dead / trap states,

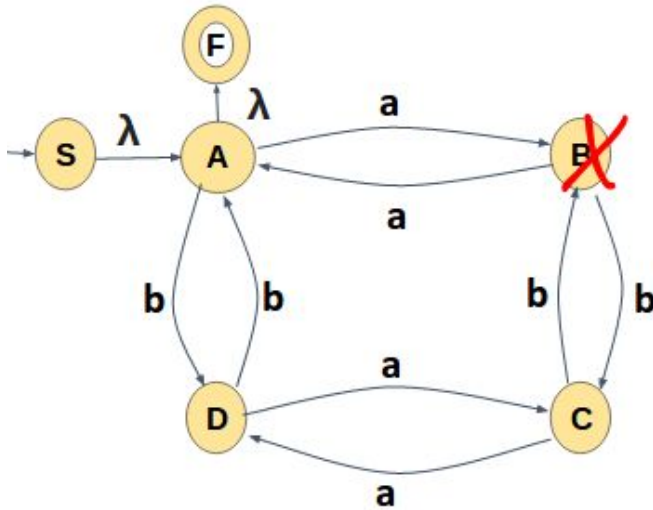
Start state A is having an incoming transition, so we create a new start state with the name S, Make a transition from new start state S to state A with λ as shown



There is an outgoing edge from accepting state, so create a new final state with the name F and have a transition from old final state A to new final state F with λ . and make the old final state A as a non final state



-> Eliminate state B



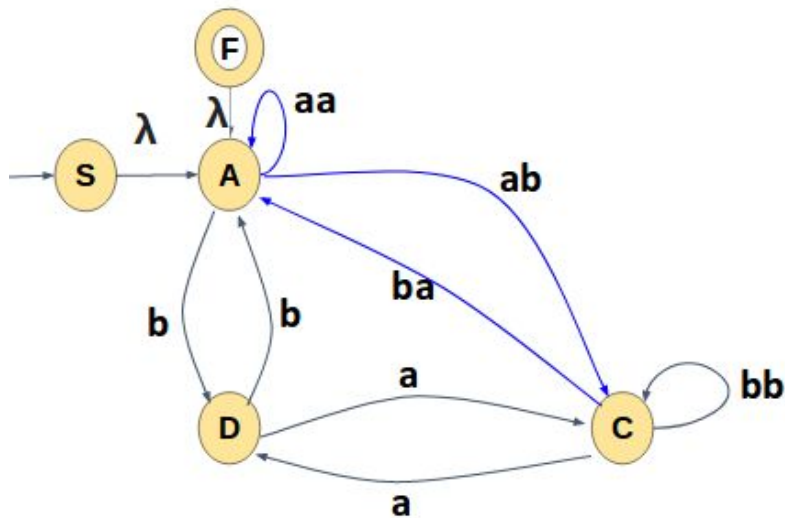
There is a loop on State a through B, so after deleting state B , State A will get a direct self loop with the cost aa

There is a path from state A to state C , so after deleting state B there will be a direct path form state A to state C with the cost ab

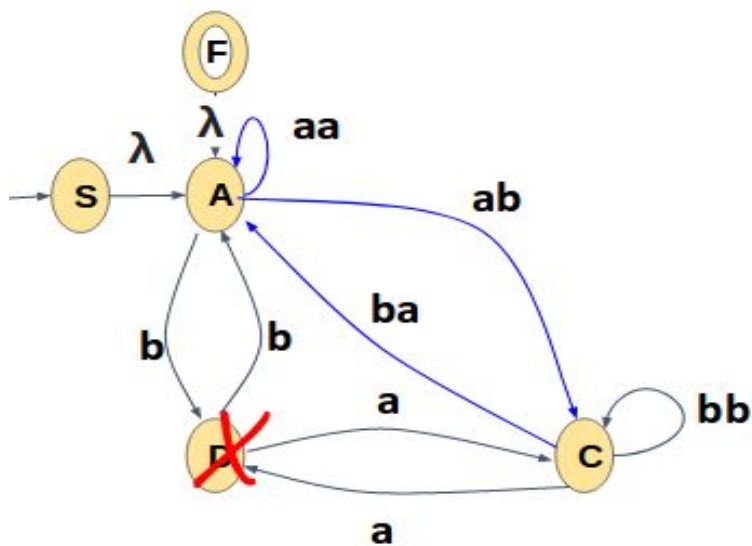
There is a path from State C to state A through state B, so after eliminating state B there will be direct path from state C to state A with the cost ba

There is a loop on state C through B, so after eliminating state B , state C will have a self loop with the cost bb

After deleting state B the automata will like below:

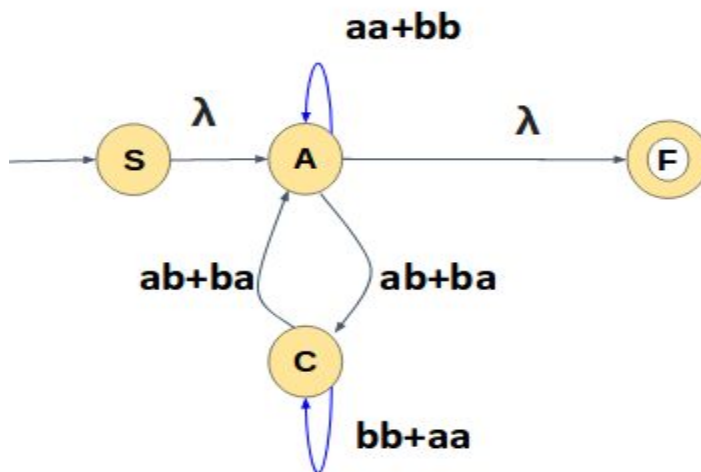


Next eliminate state D:

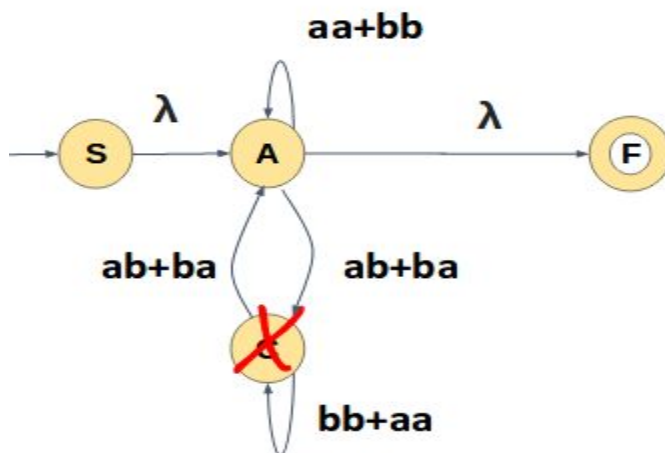


There is a loop on state A through state D and state A have a self loop,
 So after deleting state D there will be a self loop on state A with the cost $aa+bb$
 There are two paths from state A to state C, one is direct path with the cost ab and the other one is through state D with the cost ba ,
 So after deleting state D there will be direct path from state A to state C with the cost $ab+ba$
 There are two paths from state C to state A, one is direct path from with the cost ba , and the other path is through state D, so after eliminating state D there will be one direct path from state C to state A with the cost **$ab+ba$**

After deleting state D the automata will look like :



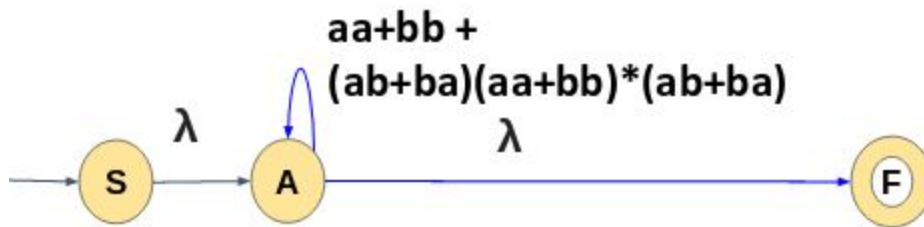
Next eliminate state C:



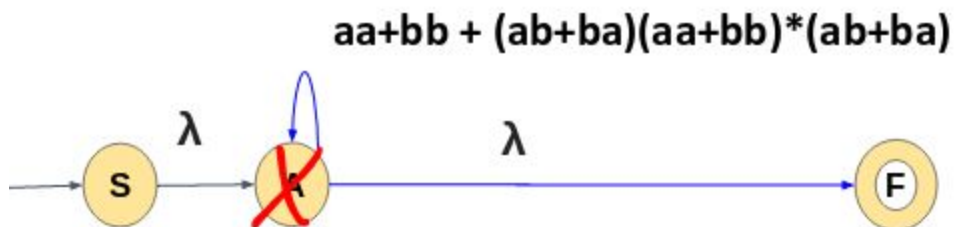
There is a loop on state A through state C and state C has a self loop, state A also has a self loop. so after deleting state C there will be a self loop on state A with the cost

$$(aa+bb) + (ab+ba)(aa+aa)^*(ab+ba)$$

After deleting state C the automata will look like:



-> Next eliminate state A(the only intermediate state)

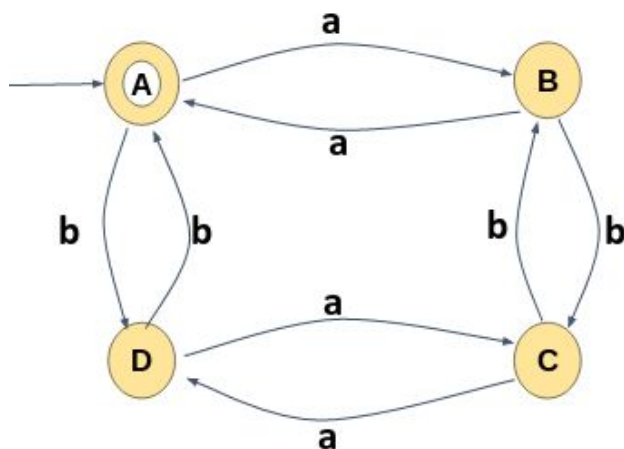


There is a path from state S to state F through state A and State A have a self loop, so after eliminating state A, we get a direct path from state S to state F with the cost $(aa+bb + (ab+ba)(aa+bb)^*(ab+ba))^*$

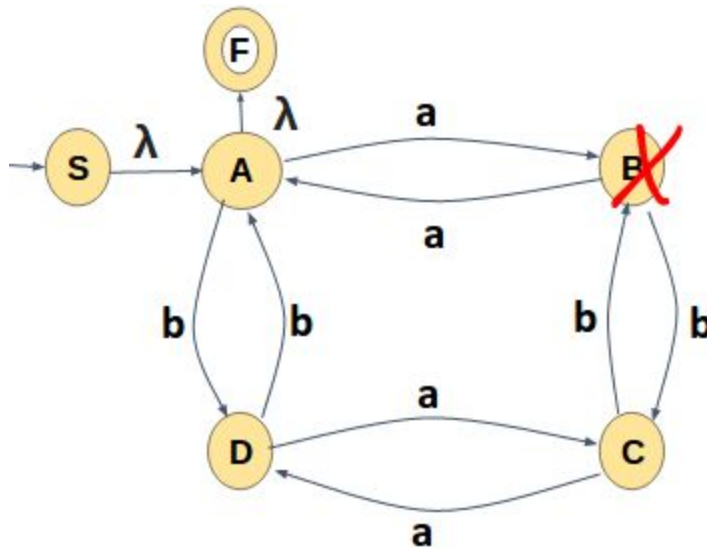
The final regular expression for the given DFA is:

$(aa+bb + (ab+ba)(aa+bb)^*(ab+ba))^*$

-> Consider the same example: Order of state elimination: B,C, D, A



-> Eliminate state B



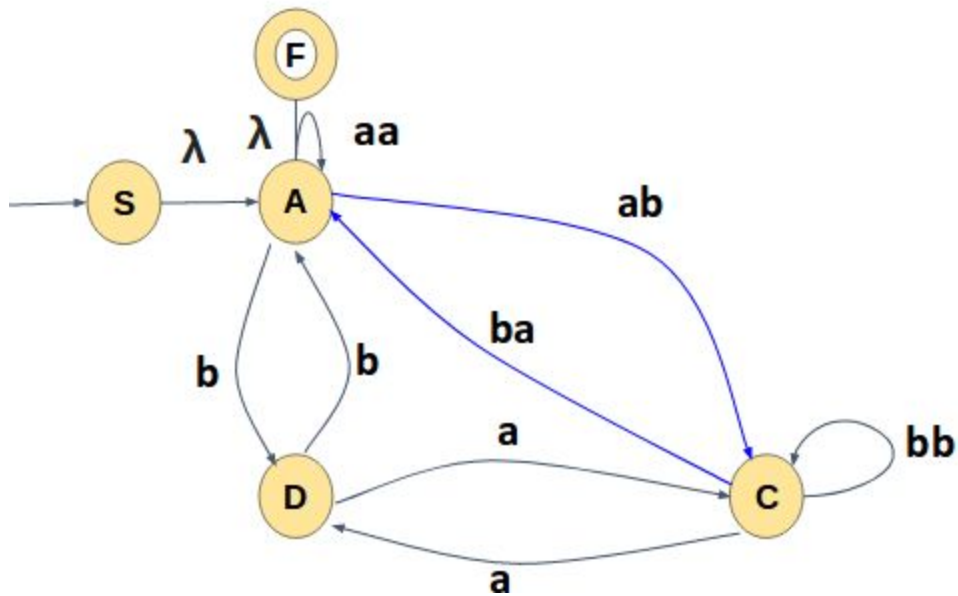
There is a loop on State a through B, so after deleting state B , State A will get a direct self loop with the cost **aa**

There is a path from state A to state C , so after deleting state B there will be a direct path form state A to state C with the cost **ab**

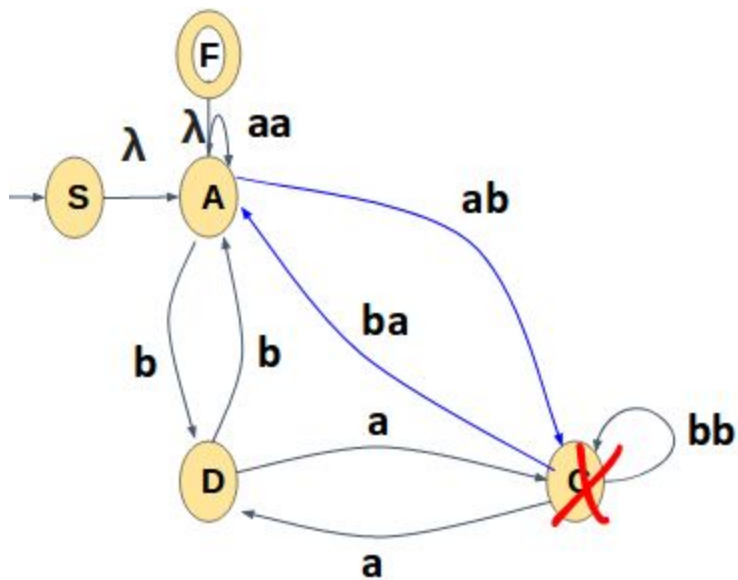
There is a path from State C to state A through state B, so after eliminating state B there will be direct path from state C to state A with the cost **ba**

There is a loop on state C through B, so after eliminating state B , state C will have a self loop with the cost **bb**

After deleting state B the automata will like below:



Next eliminate state C:

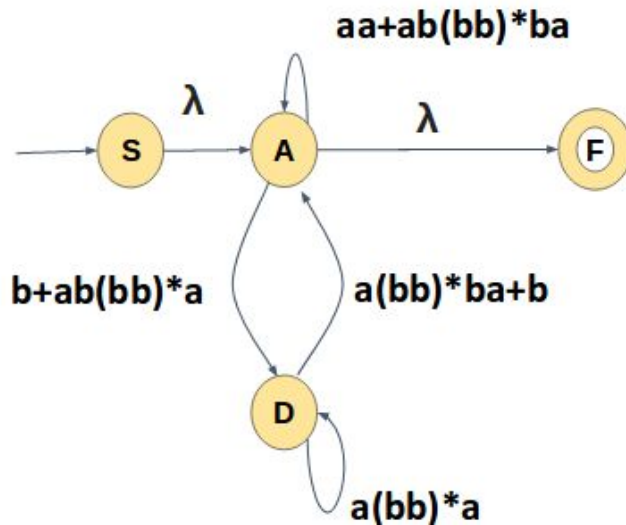


There is a loop on state A through state C, State C have a self loop, and state A also have a self loop, so after deleting state C we will have a direct self loop on state A with the cost $aa + ab(bb)^*ba$

There are two paths from state D to state A, one is direct path the other one is through state C, so after deleting state C we will one direct path with the cost $b + a(bb)^*ba$

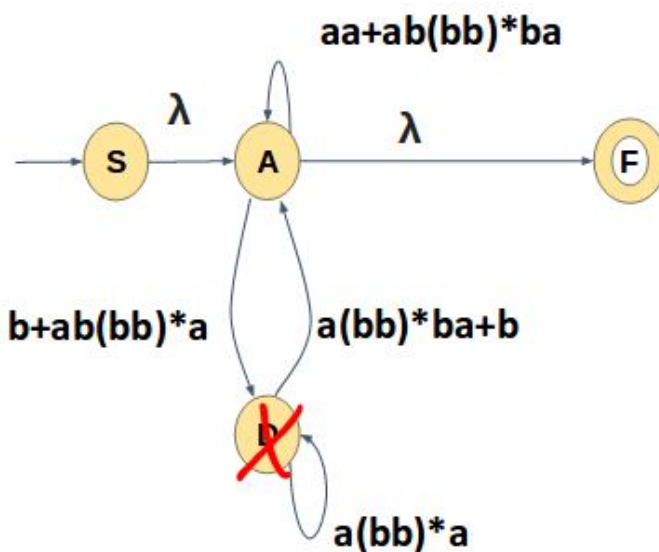
There are two path from state A to state D, one is direct path and the other one is indirect through state C, So after eliminated state C there will one direct path from state A to state D with the cost $b + ab(bb)^*a$

After deleting state C the automata will look like:

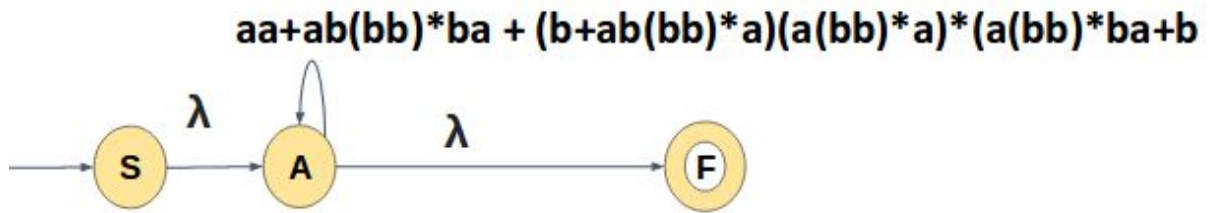


Next eliminate state D,

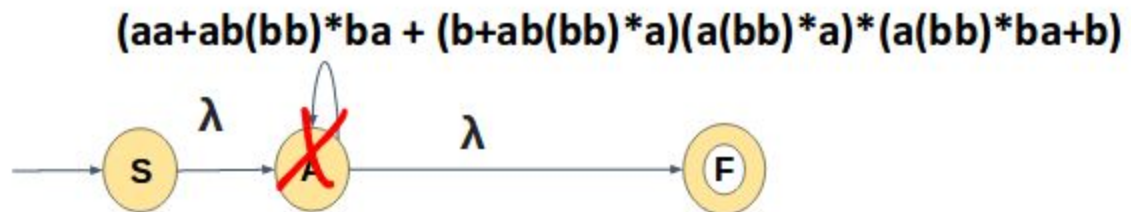
There is a loop on state A through state D, and state A and D have a self loop, so after deleting state D, state A will have a self loop with the cost $(b+ab(bb)^*a).(a(bb)^*a).(a(bb)^*ba+b) + aa+ab(bb)^*ba$



After deleting state D the automata will look like:



> Next will eliminate state A (the only intermediate state)



There is a path from state S to state F through state A and State A has a self loop, so after eliminating state A, we get a direct path from state S to state F with the cost **$(aa+ab(bb)^*ba + (b+ab(bb)^*a)(a(bb)^*a)^*(a(bb)^*ba+b))^*$**

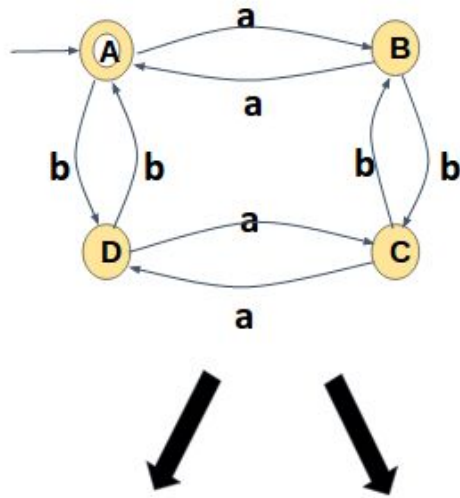
Final regular expression for the given automata is :

$((aa+ab(bb)^*ba + (b+ab(bb)^*a)(a(bb)^*a)^*(a(bb)^*ba+b))^*$

We get two different regular expressions for the same automata based on the order of eliminating states in the automata:

Note: We can eliminate the state in any order and the regular expression may be different but the language accepted by the automata/ regular expression will be same

Example 9 :



$(aa+bb + (ab+ba+(aa+bb)^*(ab+ba))^*$
 Eliminate : B,D,C,A

$((aa+ab(bb)^*ba)^* +$
 $(b+ab(bb)^*a)(a(bb)^*a)^*(a(bb)^*ba+b))^*$
 Eliminate B,C,D,A

