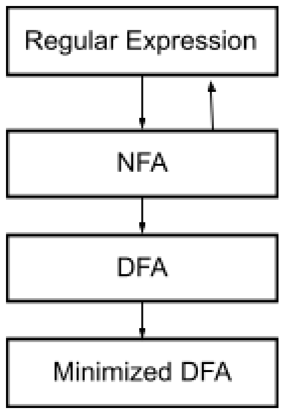
Assignment-8

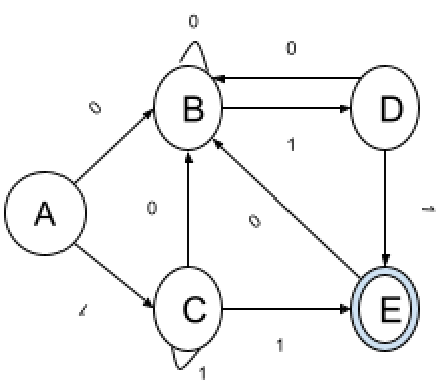
TOPIC: **DFA minimization.**

It is the mechanism by which we create DFA with a minimum number of states required to create a DFA.

Suppose a language is defined to us so considering what we have learnt from the past lectures it is as follows:



Now we will look into **example-1** to further understand the concept.



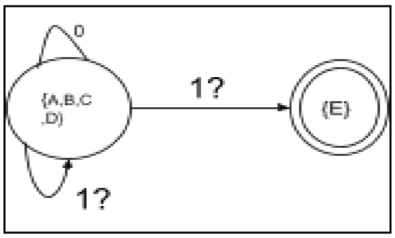
Here we have a DFA and we will find the minimized version of this DFA that means we will find the least possible number of states required to reach the final state and create a DFA out of it.

To begin with, to make it easier for us we will create a table in which we will note down all the possible transitions that each state does by consuming each action/symbol.

	0	**1**
A	B	C
B	B	D
C	B	C
D	B	E
E	B	C

*We will make this table up by looking at the given DFA. We also label the final state in the table
*Orange highlight represents the final state

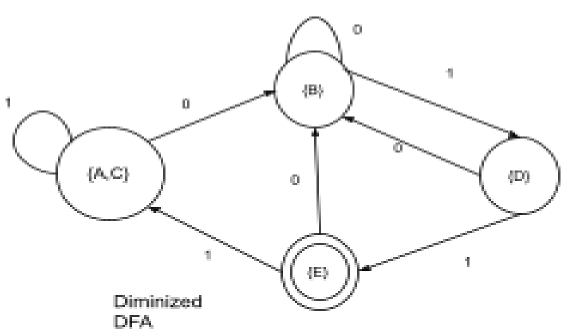
Now we assume that to make a DFA at minimum two states are required, a non final state and a final state. This is known as the **0-equivalence.**
At 0-equivalence there exist only two states.
NON Final State : {A,B,C,D} Final state: {E}



However when we do this, we see that there is some confusion because of D because when D consumes 1 it transits to the state E as shown in the table above. However, The rest of states transits within the {A,B,C,D} state. We cannot have any “confusion'' while creating DFA thus we introduce **1-equivalence** in which we separate the state which is causing confusion from the set. In this case, D is separated from {A,B,C,D} state. Therefore,
NON Final State : {A,B,C}{D} Final state: {E}. Here if we create a table for this we get,

	0	**1**
{A,B,C}	{A,B,C}	…….?
{D}	{A,B,C}	{E}
{E}	{A,B,C}	{A,B,C}

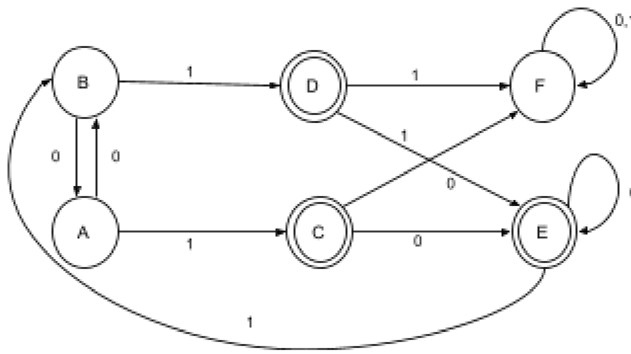
We made this new table using the first table. Here, when {A,B,C} consumes 1, some confusion is created by B because when B consumes 1 it transits to D however, D is not present in {A,B,C} and another loop forms by itself. Thus, we will have to separate B as well. To do so, we have to go for **2-equivalence.** In here, after following the same process just like before we will get a DFA just like below,



To summarise, first we created a transition table from the given DFA which will include transitions from each state using each action. Then we wrote the

0-equivalence and if any confusion was found then we separate the state thus creating 1- equivalence. We will keep on repeating the process (means finding 2,3-equivalence) until there is no confusion left and until the number of states remains the same consequently. Therefore, at the end we receive our final DFA.

Now we will look into another **example-2** and by using the same process as mentioned above.



We will make a transition table using the DFA above.

	0	1
A	B	C
B	A	D
C	E	F
D	E	F
E	E	B
F	F	F

Now as we know 0-equivalence has two states, Non-final and final so,

0-equivalence:

Non-final: {A,B,F} Final : {C,D,E}

So when the state {A,B,F} consumes 0 it transits to non final state however it consumes 1 we see that A and B goes to final states however F goes to itself(Non final state) thus, F is creating confusion so we remove it from the group. Therefore 1-equivalence comes in.

1-equivalence:

Non-final: {A,B} {F} Final : {C,D,E}

Now that F got separated we see that when {C,D,E} consumes 1, C and D transits to non-final {A,B} state however E transits to F state therefore E needs to be separated. We keep the rest as it is because none of the other states causes any confusion.

2-equivalence:

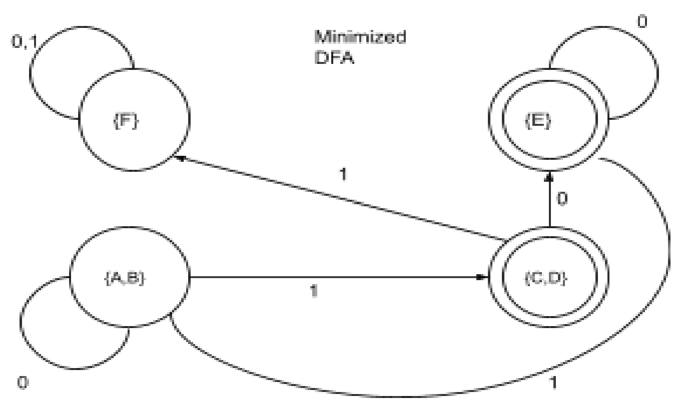
Non-final: {A,B} {F} Final : {C,D} {E}

Here everything seems alright and no confusions are detected; however for the sake of checking, we will move forward to 3-equivalence. If the number of states remains the same then we will directly draw our DFA.

3-equivalence:

Non-final: {A,B} {F} Final : {C,D} {E}

The number of states are the same consequently so we can finally draw our DFA.



We have to remember that it is not compulsory to draw transition tables for each equivalence. We draw tables just to make things easier for us to navigate and understand.

Next, in the lecture we solved another problem (**example-3**) using the same method as mentioned above.