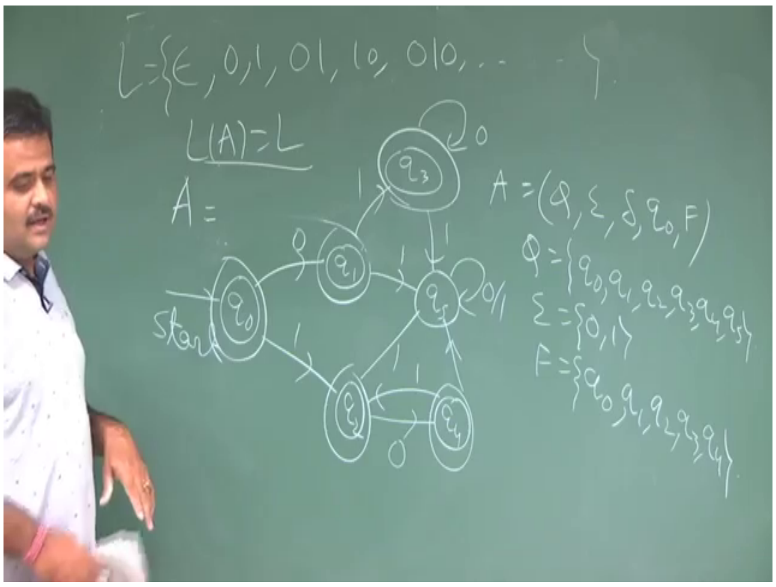
Inroduction to Automata, Languages and Computation

Prof. Sourav Mukhopadhyay
Department of Mathematics
Indian Institute of Technology, Kharagpur

Lecture – 06
Building DFA (Contd.)

So we are talking about DFA and we are talking about that how we can build a DFA from a given language.

(Refer Slide Time: 00:26)



So, this is one example in we discussed in the last class, suppose you have a language which is having alternate 0 and 1’s. So, this is the null string this is having nothing. So, this is also should be accepted this is 0, this is 1, this is 0 1 1 0 0 1 0. So, lie like this alternate 0 and 1. So, we are looking for a DFA deterministic finite automata which accept this type of strings, all this type of strings.

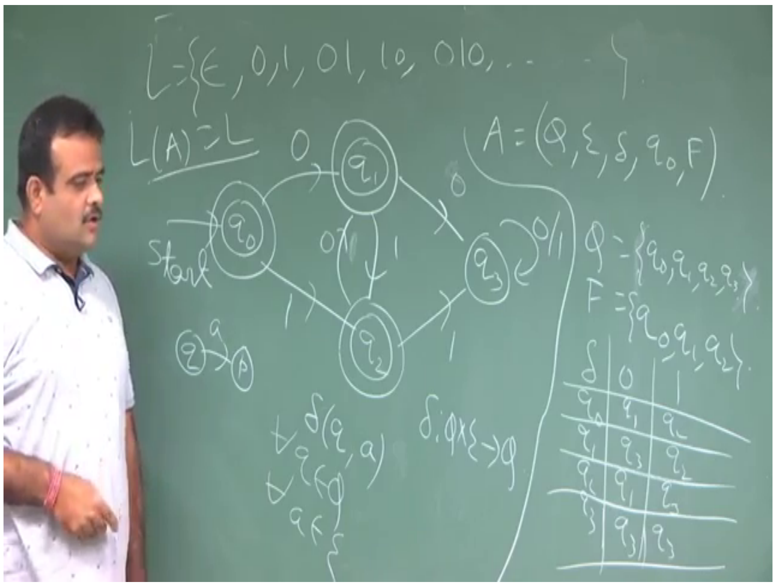
So, in the last class we have seen this is one example where to accept the epsilon the null string, we need to put that the starting state as a final state because, DFA has no epsilon move it is a deterministic finite automata. So, it has no epsilon move. So, that is why we need to put the null string null and the starting state as a final state in order to make it in order to accept this epsilon. So, this is the example where it is accepting the whole string, where 0 has no other coming in alternative way.

So, this has having. So, this is one example where F is having. So, what is Q over here? So, this is our a Q sigma delta of q 0, F what is Q? How many state we have? We have q 0, q 1, q 2, q 3, q 4, q 5, we have 6 states, we have 6 states over here and we have sigma s 0 1 binary string, binary input and q 0 is starting state, but F is here final state is not singleton here we have many states as a final state. So, the here this is one example where final state is not a single (Refer Time: 02:17) state, who are the final state? All the state except the q 5, q 0, q 1, q 2, q 3, q 4.

So, all the states we mark as a accepted state of the final state other than q 5, because q 5 is the dead state, if you go to q 5, I mean that string is not accepted and then we can write the deltas rules over here. So, this is one x one df 1. So, this is accepting a L of A it is nothing, but this L. So, this is a regular language. So, now, the question is, is this unique, I mean whether we can have a better DFA in the sense of number of states is less. So, we will see we can have another we can construct another DFA with 4 states, which is also accepting this language. So, let us try with that.

So, we can have another example where it is accepting this type of stream.

(Refer Slide Time: 03:33)



$L = \{\epsilon, 0, 1, 01, 10, 010, \ldots\}$
$L(A) = L$
$A = (Q, \Sigma, \delta, q_0, F)$
$Q = \{q_0, q_1, q_2, q_3\}$
$F = \{q_0, q_1, q_2\}$
Start
$\delta : Q \times \Sigma \to Q$
$\delta(q, a)$

So, q 0. So, in order to accept the epsilon, we have to make this to be final state, then we can go to q 1 with the move 0 and we can go to q 2 with the move 1 and both are accepted. And from q 1 from q 1, if we see a 1 then it is also accepted, because this is 0 1, but if we see a 0 after the reading this then we must go to a state, which is here refer to be a gate state this is 0, because 0 0 is not accepted. So, we should not accept the 0 0, it should be alternative ok. Now from q 2, if we see a 0.

So, this is. So, from q 2, if you see a 0 we must go here, because this is kind of 1 0 or it may come from here 0 1 0 like this, but if you see a 1 we should (Refer Time: 04:48). So, this is and from here we can because the deterministic finite automata should have moved for every resist this sigma is. So, this is one example. So, this is Q sigma delta q 0 F. So, we have our Q is. So, our Q is this sets in 0 q 1 into q 2 q 3, we have 4 states, we have 4 state among this 4 state we have a final state, all except the q 3 which is that get state. If you wish to q 3 that those string will never accepted, because those strings are not alternating 0 and one's and delta we can have the rules.

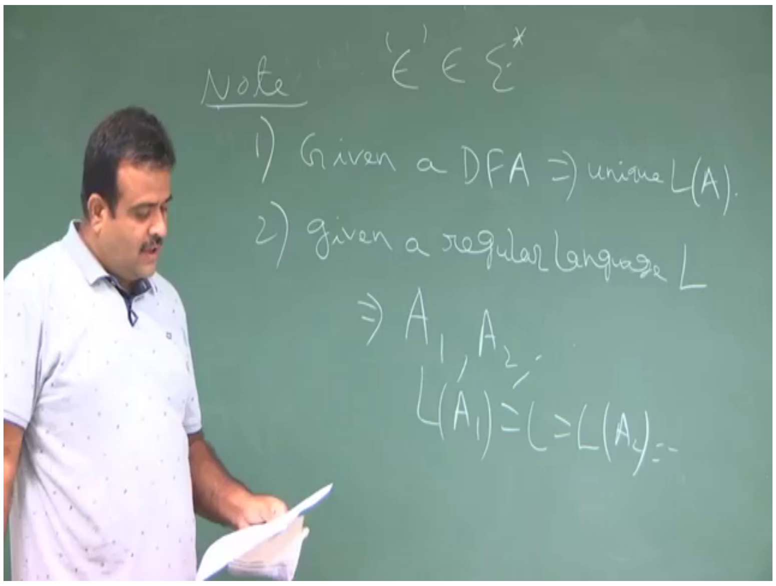
So, delta is like this. So, we have 0 and 1 move and we have p 0 q 1 q 2 q 3. So, deltas should be delta is a function from Q cross sigma to Q. So, it is a deterministic function. So, you should have. So, this is the input. So, we should have. So, for all pr of. So, for all pr of this we should have output. So, this is. So, for all q and for all a we should able to define delta that is the meaning of deterministic, it is deterministic finite automata finite, because this number of states are finite and deterministic because, it has a move I mean for every see every we are at a state we have an input we have a move ok.

So, this is a function form Q cross sigma to Q. So, for a given state we are at some state q, we have a move a we should go to some other state or maybe same state, but this is in a deterministic way. So, that we have we clearly have this function. So, this is from q 0 we can go to if it is 0, you go to q 1 if it is q 1, you go to q 2 from q 1, if it is 0, you go to q 3 if it is 1, we go to q 2 and from q 2, if it is 1, we go to if it is 0, we go to q 1 and if it is 1, you go to q 3 and from q 3 will be remain at q 3.

But that move is also explicitly given from q 3 also if we have a 0 input will be at q 3, if we have 1 it fully you have will be at q 3. So, this is another DFA which is accepting this. So, this is L of A L which is accepting this alternate 0 1. So, it is not that we have a unique DFA, I mean from a giving language we can have many DFA, which is accepting this, but the other converse is like if we have a DFA, we have a unique language.

But if you have a regular language then we can have many DFA's which is accepting this ok. So, that we can write is, but if you have many DFA it is desirable to have the DFA, which is smaller like in the terms of number of states.

(Refer Slide Time: 09:02)

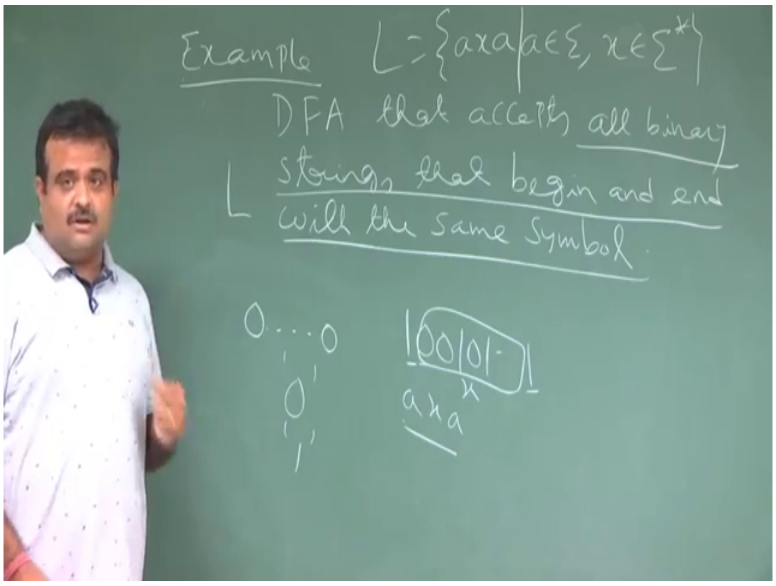

Note 'ϵ' $\in \Sigma^*$
1) Given a DFA $\Rightarrow$ unique L(A).
2) given a regular language L
$\Rightarrow A_1, A_2, \ldots$
$L(A_1) = L = L(A_2) = \ldots$

So, let me write this. So, this we can say note on observation. So, first note is given a DFA we have a unique language, I mean given a DFA we know the language of that, but the given language, given a regular language, language L, we know this is regular then we may have different different DFA, we may have A 1, A 2 may be more than something such that L of A 1 is A which is same as L of A 2 like this.

So, in the last example we have seen, we can have 2 DFA, but whichever is desirable with the fewer number of states are desirable having, we should have a DFA which is having less number of state. So, we should be careful about the epsilon string, because this is also a sigma star and DFA is not having any epsilon move. So, to accept this we need to put the, we need to make the starting state as a accepted state ok.

So now, we take one more example, to construct DFA and then we move to the finite automata, which is non deterministic finite automata which is called NFA.

(Refer Slide Time: 10:58)


Example $L = \{axa \mid a \in \Sigma, x \in \Sigma^*\}$
DFA that accepts all binary strings that begin and end with the same symbol.
L
0 . . . 0
0
1
100101 . 1
x
axa

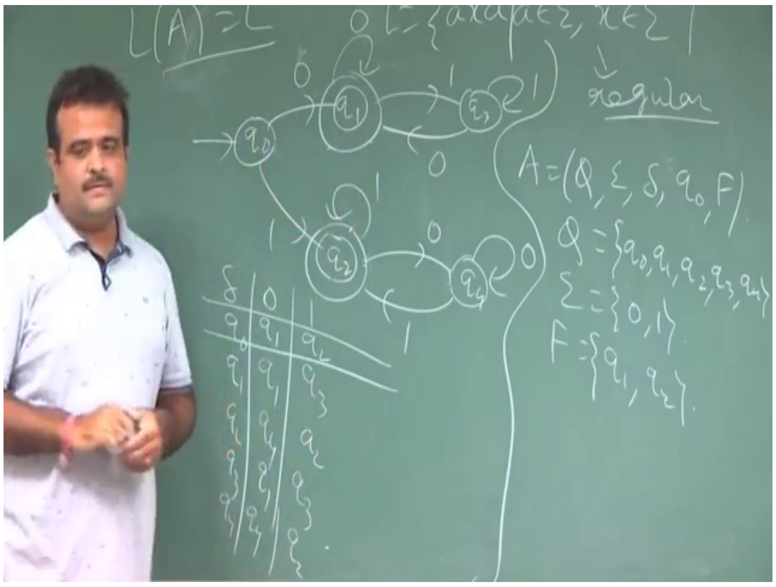
So, let us just quickly take one more example on to build the DFA. So, we want to have a DFA that accept the binary string. So, DFA accepts all binary strings that begin and end with the same symbol that begin and end with the same symbol, begin and end with the same symbol. So, this is our language, our language L is like this.

So, L is it should begin and end with the same symbol like if it is beginning with the symbol a in x a. So, a a is belongs to sigma and x is belongs to sigma star ok. So, the epsilon is not belongs to L, because epsilon is we are not beginning with any symbols. So here, explicitly we are telling that we are beginning, if you are beginning to like this string, if we are begin with 1, 0, 0, 1, 0 whatever may be, but the n symbol should be 1. So, this is our a and this part is over x. So, a x a.

So, if we are begin with 0 then after some time, we must end with 0 only 0 also, if you are being only 1 string, I mean length of 1 we are beginning with 1 0 1, we are bringing with 1 ending with 0 1, you are beginning with 1 ending with 1. So, this is the way. So, all type of all types of this thing. So, this is of this form. So, we want to check whether this language is regular or not in other word, we need to build a DFA Deterministic Finite Automata, which is accepting this language so, let us try that.

So, can you have a DFA here, which can accept that. So, this is again the trial and error method, because so far we don't know this language is regular or not ok. So, it is kind of I mean, if it to be regular we need to have a DFA. So, anyways later stage will see the regular expression and it will give us some light inside that. So, a way to construct a NFA of automata ok.

(Refer Slide Time: 14:20)



So, let us try that. So, we start with here.

So here, we do not need to make this as a final state, because epsilon is not a accepted it, after this null string is not accepted over here ok. So, we put a q 1 if we have 0, if you have 0 then also we are going to final state. If you have 1, there also we are going to final state. So, these 2 are the final state ok. So now, from here, if we are reading a 0. So, if we are having a 0, we can just go here, because 0 0 which is same and from here, if we have a 1 then where we go? Then we must go to the here, yeah this is also possible.

So, 1. So that; that means, if you see 0 1 we come here again, if you see a 0 will only go there, but, but if we see a 1 will come. So, this is this is again as I said this is not (Refer Time: 15:46) method. So, this is not possible because, if we put a 1 over here then what is the problem? Problem is if we say is 0, we go there if we see a 1, we come here then again you see 1, we go there and this is the final state, but this string is not belongs to L, it has to be 0. So, this is not the way. So, we can have like this.

So, we need to take him from another state maybe we can go to q 3. Now if we see a 1 over here, you must go to q 3 and if we see a 0 over here, we can because 0 0 is accepted fine, now from here also if we see from q 3. So, 0 1 and it again, if we see a 1 0 over here, we can go here, but if we see a 1 then we must hop over here until you see a 0 and then we can have q 4 for this note. So, if we see a 1 over here. So, this is starting it 1, if we see a 1, you must go here, but if we see a 0 over here.

So, for 1 we must hop there, because that is the accepted string. So, for 1 we here, but if we see a 0 then we must go there with a hope to get a another one. So, now, from here if we see a 1, we can come here and if we see a 0 again then we must go there ok. So, this is one example we can minimize this I think, I mean those techniques we will discuss in the later stage, but this is our A. This is our A, we have this may not be unique, because we I am sure that we can combine this instead of 2 state, we can have 1 state can you try that like we can go from.

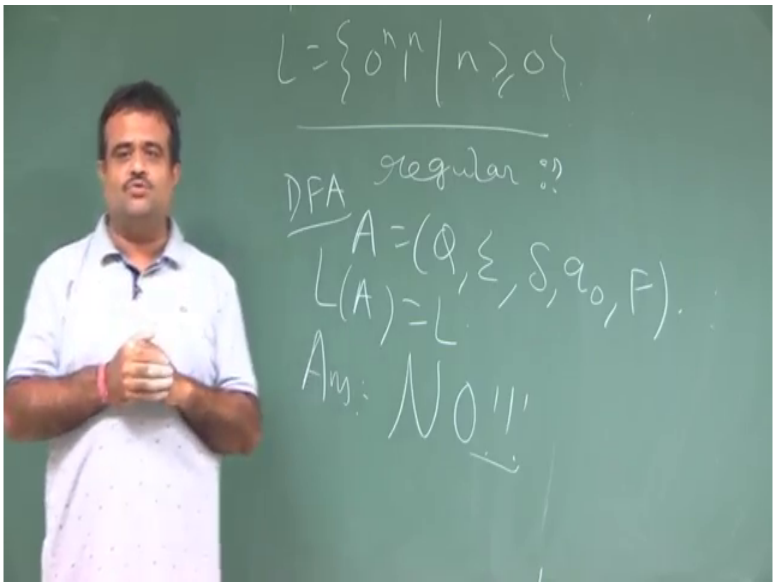
So, 0. So now, if we have from q 2, if you see 0 we can go here, but problem is it is coming alternatively like here we are 0 input reaching to another 0 input reaching here to the final state, anyway as the exercise you can check with that this is a only one way, I mean this is the only DFA or not or we can have a another DFA with the minimum with the number of. So, this is this is our A, A is begin Q sigma delta q 0 F. So, Q is nothing but Q is nothing, but how many states? We have q 0, q 1, q 2, q 3, q 4, there are 5 states and sigma is just a alphabet input binary input 0 1 and the q 0 is the starting state and here F is that we have many the F is not a single strong set then we have many accepted state. So, 2 accept q 1 and q 2 and what is delta? Delta is the rule. So, for 2 input and we have q 0 q 1 q 2 q 3 q 4 also we have.

So, for q 0 if we see a 0 it is going to q 1 if we see a 1, it is going to q 2 and for q 1 if we see a 0 it is remain at q 1 and if we see a 1 it is going to q 3, there is no dead state over here, it can able to come back then from q 2, if we see a 0, it is going to q 4. If you see a 1 this remain at q 2 come from q 3, if we see a 0 it is coming to q 1 and if you see a 1 it will be remain at q 3 with a hope that it will get a 1 until I sorry until it get a 0, it will be there because, we need to end up with a 0 because, we started with a 0 we started with 0.

So; that means, we need to end up with a 0. So, q 3 and from q 4 we have. So, if it is 0 will be here, because this branch is started with 1 so; that means, it should end up with 1. So, after a few hoping hopping, we can if we get a 1 then we go to the q 2. So, this is the deterministic, I mean rule like this is the transition rule for this DFA and. So, this is our

L. So, this is our A. So, L of A is nothing, but. So, this language is regular, this is regular ok.

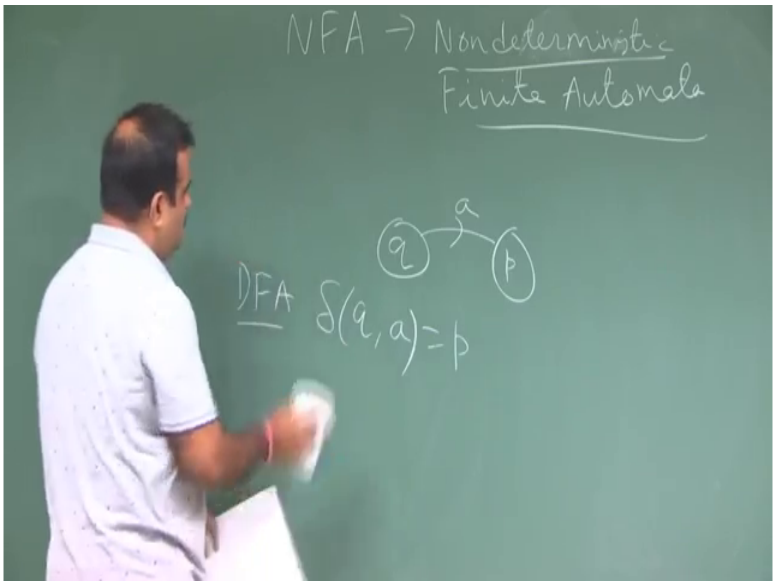
(Refer Slide Time: 21:54)


$L = \{0^n 1^n \mid n \geq 0\}$
regular ?
DFA
$A = (Q, \Sigma, \delta, q_0, F)$.
$L(A) = L$
Ans: NO !!

So, this is another example, where we can have this given a language, we can construct a DFA provided itself it is not for that the all language is the regular. For example, if you have a language like this 0 to the power of n 1 to the power n. So, like this is basically the number of 0's and number of 1's are. So, if we have consecutive 3 0's then we should have consecutive 3 1's. So, like this. So, whether this is a regular language or not? So, can you have a DFA? So, if it is a regular then we should have a DFA are to accept this whose language the DFA.

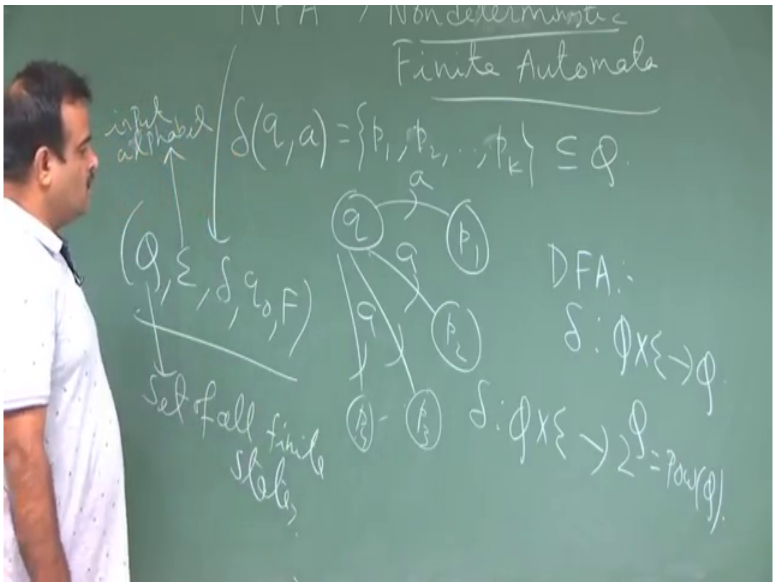
So, can, we have a DFA whose language is L? So, this is a DFA can you have a DFA whose language is L? So, answer is no, we cannot DFA is not possible to construct the accept, because this is a particular pattern like we have to memorize the, how many 0's we have encountered before going to 1? So, that is the things, but in DFA we have no memory, we cannot use a extra stack or something. So, this will come in the pushdown automata that is also finite state machine, we will discuss that how this can be accepted, but DFA no ok.

(Refer Slide Time: 23:46)



So, now we will move to another finite automata which is called NFA or Non deterministic Finite Automata. NFA this is called non deterministic sorry, ministic finete automota or automation. So, I will there it was deterministic and this is non deterministic. So, there should be a difference like that non deterministic means the rules are I mean it is a non deterministic is move is not specified it can be from we have given a state q and given input a So, the delta of q earlier it was just a p a fixed state, but here this is not a fixed it could go to the set of states.

(Refer Slide Time: 25:10)


NFA → Nondeterministic Finite Automata
input alphabet
$\delta(q,a)=\{p_1,p_2,\ldots,p_k\} \subseteq Q$
(Q,Σ,δ,q_0,F)
Set of all finite states
q
a
p_1
p_2
p_3
DFA:
$\delta: Q\times\Sigma \to Q$
$\delta: Q\times\Sigma \to 2^Q = Pow(Q)$

So, that is the difference. So, this move is not deterministic, it is a non deterministic in the sense of.

So, this was case of DFA, it was a fixed state, but here it is not fixed it can go to many state like p 1, you can go to p 2, it can go to p 3 like this it can go to some. So, that is the difference. So, it is moves is. So, p 4 like this. So, delta of q comma a is not unique it is a set, this is the set like p 1 p 2 p k and so, this is; this is a subset of Q. So, it could be any subset of Q ok. So, that is the difference. So, move is not deterministic.

So, it can go to any one of these states. So, these are the possible state it can go. So, that is the idea. So, that is non determines other things are same in the non deterministic finite automata, we have again the 5 couple delta q 0, if this is same, this is the set of all states all finite states and this is input alphabet, which is alphabet and q is the starting state F is the final states, delta here is different earlier delta was for DFA for DFA no orders were delta, delta as a function for Q cross sigma to Q, it is going to a fixed state, but here it is not going to fixed state.

So, for here delta is Q cross Q, this is power I mean the subset, the pow set to deliver Q we can say our pow set, pow set means set of all possible subsets of q. So, like if we take q, if we have a this is a subset and you will discuss this in more details in the next class.

Thank you.