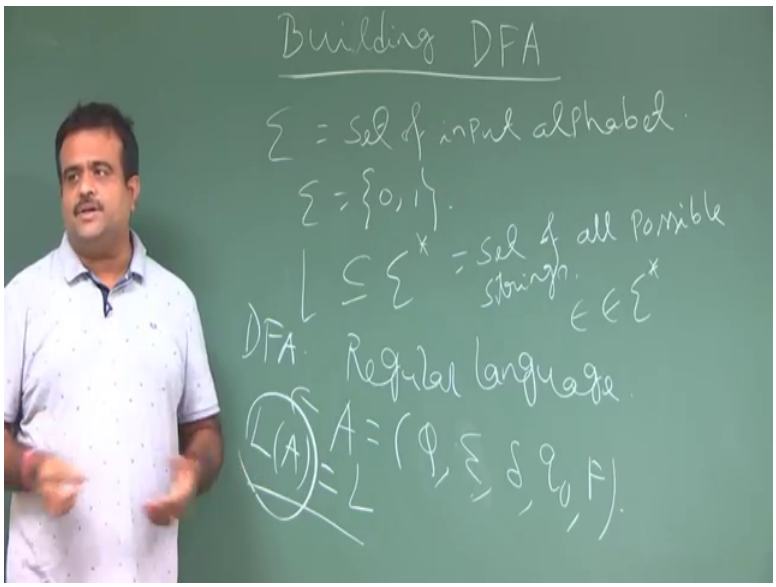
Introduction to Automata, Languages and Computation
Prof. Sourav Mukhopadhyay
Department of Mathematics
Indian Institute of Technology, Kharagpur

Lecture – 05
Building DFA

(Refer Slide Time: 00:17)



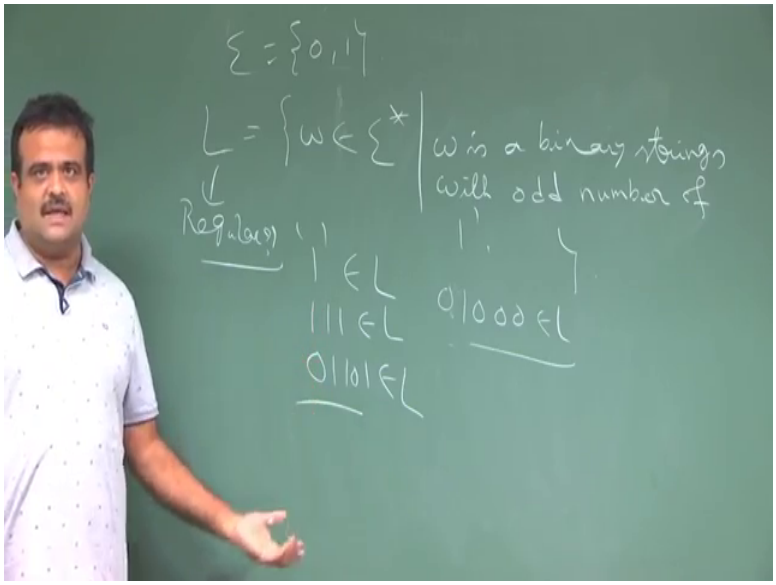
So we are talking about how to construct the DFA, I mean we know the regular language means say so a language suppose you have a this is the sigma so set of input alphabet. And if it is binary input, binary alphabet it is 0, 1. Then the language is a subset of L star this is the set of all possible string of any length including the length 0, set of all possible string including the null string that is epsilon. So, epsilon also belongs to including the null string.

So, any subset of sigma star is called a language. Now, this language will be called a regular language, a regular language. If we have a DFA which is the 5 tuple Q, sigma, delta - the rule, and the start state, and we have final state; if we have a DFA such that the language accepted by this DFA is this. So, this is the meaning is we strike a string if the string is we read the string starting from the state q 0, and if it is reaching to the final state then that string will be accepted by this DFA. So, this is L of A means the language of DFA, that means, the all possible string which are accepted by this DFA which is end with a accepted state after the end of the string input.

So, if this is happening then we called L is a regular language. So, basically to prove that a given language is a regular, we need to construct a DFA. So, we need to build a DFA which is accepting the which is the whose language is same as the language L, then we call this is a regular language, so that is the that is we are going to do now. How to given a language we want to see whether we can build a DFA for that language.

So, this we will do by trial and error method there is no magic I mean there is no we have to do the trial and error method. In the later stage we will see if we can write this language in a regular expression, then we can construct a NFA, in particular epsilon NFA, and then from there we can construct a DFA, but for the time being we will treat this as a trial and error method to build up.

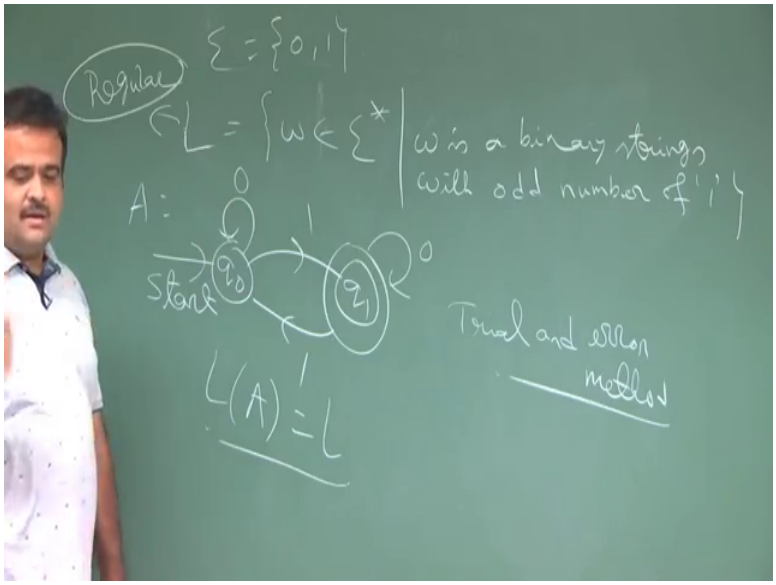
(Refer Slide Time: 03:23)



So, suppose we take a alphabet and we take a language like this. So, this is the set of all string such that w is a binary string, binary strings with odd number of 1s odd number of 1s. So, this is our L, so that means, this is belongs to L, 1 1 1 belongs to L, 0 1 1 0 1 belongs to L. So, we just count the number of 1s in that string if it is number of 1s is odd then this is belongs to L. So, even 1 0 0 0 sorry 0 this is also belongs to L. So, these are the language. So, we have to check whether this is a regular or not. So, this is the question. So, to be a regular we need to have a finite automata, deterministic finite automata which is accepting this which whose language is same as this language. So,

these we have to try. So, we have to build an automata which is accepting this. So, this is we will do for the timing by trial and error method.

(Refer Slide Time: 05:03)

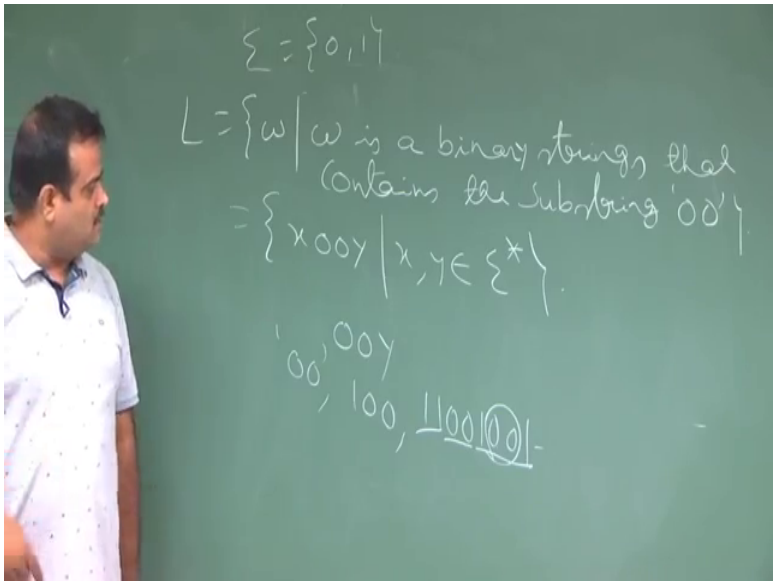


So, how to do that? So, we start with the q 0, this is the starting state. Now, if we have a we were just counting the number of 1. So, number of 1, if it is 1, if it is odd, then we are in a odd number of 1s, then we are in final state. So, if we having a 1, we reach to the final state.; but if we having a 0, we have over here. So, like this. Once we reach there, if we have 0, we do there because 1 0 is a accepted string, 1 0 0 is again a accepted string, 1 0 0 0 is again acceptance string. Now, if we put a 1, then we should not accept that, because 1 1 is not a belongs to L. So, we just if we having a 1, we will come back here so, this is by trial and error method, trial and error method.

So, later stage we will see how we can effectively construct a DFA from a given, if we know the language is regular because that has that information has to be there, s, that will be that will be discussed in the in maybe third week or so. When we talk about regular expression if a language is regular it we should able to express in a that there are some regular expression. So, those thing we can have a for regular expression, we can have NFA - non deterministic finite automata, then we can convert that NFA to the DFA. But first of all that information has to be there, the language is regular. But suppose we do not know anything, then we have to do the trial and error method to I have this ok.

Now, we can have another example. So, this is the this is our this is our DFA which is accepting whose language is same as this language, so that means, this is a regular language. So, this language is regular ok. We will take another example. Say if we consider all the string which contained the sub strings as 0 0. So, we will check whether that collection of string is regular or not.

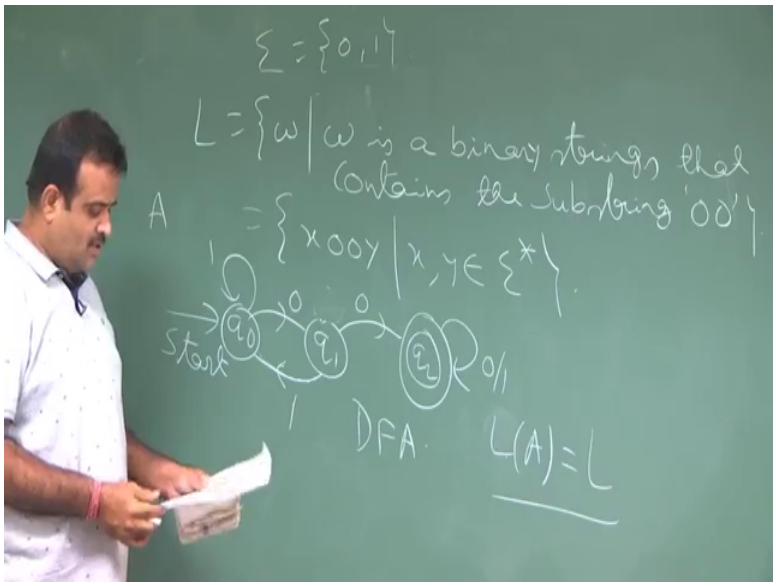
(Refer Slide Time: 07:37)



So, again alphabet is this. We consider the language w, where w is a binary string, binary strings that contain that contains the substring 0 0. So, 0 0 will be the part of the string anywhere. So, like this should be written as like this, x 0 0 y and this x y is belongs to sigma star, so that means, x y could be null also epsilon see if x is epsilon. So, if x is epsilon, this is of the form 0 0 y y could be epsilon also, so 0 0 is remember. Then say 1 0 0, the in this case x is 1 and y is a epsilon. So, we can have 1 1 0 0 1 0 0 like this. So, this is our, this is our x, this is our y. Even in the y we have so any kind of thing, but in general we can write in this form, where this x y is coming from sigma star. So, this could be x y could be epsilon also.

So, now, we want to check whether this is a regular or not, that means, whether we can build a DFA. So, if you do not know whether it is regular or not, then we have to try by trial and error method. But if we know that this is a regular language, then there is a technique by doing by regular expression ok. So, how to pass it? So, you want to have a DFA which is accepting this language.

(Refer Slide Time: 09:37)

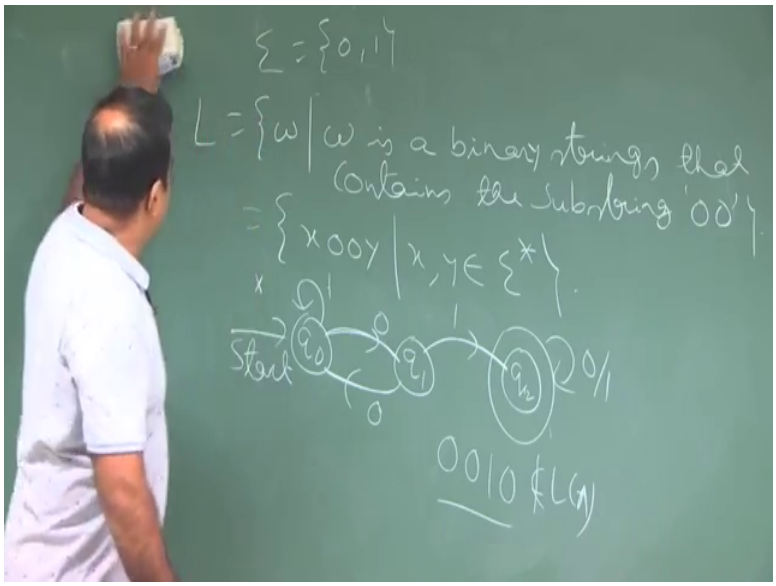


So, again we will start with q 0 starting state, and you have a q 1 another state. One state is not sufficient, because so we are reading a 0. So, we are going to q 1. So, from q 1 here, so from q 0, if you see a 1 will be remain here, because we need to read 0 0 to reach to the final state. So, suppose q 2 is the final state. Then if we see a from here if we see a 0, then we end up with the final state.

But, if you see a 1, then we must come here we must come here. This is one possibilities or we can half here, because if we half now, we must come here. Because, if we must come here, if we see a 1 ok. Because, 0 1 then we are here then again 0 0 is there so yeah that is ok. This is trial and error method. There is no I mean we are just trying to reach to the final state, if we see that type of string ok.

So, then from here does not matter whether 0 or 1, because you already we already encountered two 0's. So, this is the this is the DFA. This is our DFA a which is accepting, which is the language of this DFA same as L. So, again as you can see this we were just trying by trial and error method.

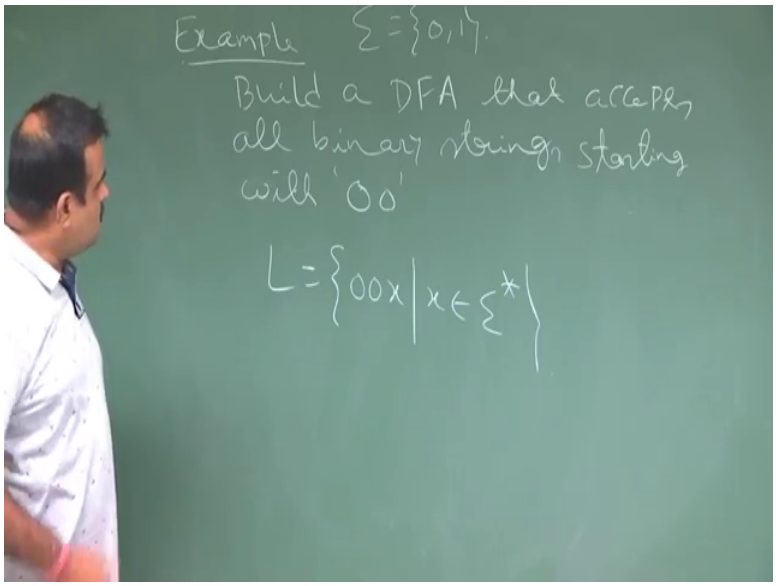
(Refer Slide Time: 11:41)



So, can you have a another DFA, which is accepting this. So, like if we construct this DFA, so q 0, we start from here and q 1 if you see a 0, you go there. And then if we see a 1, you half here then the q 2 so this is if you see a 1, we go there. Now, but this is not this is not accepting, this is not accepting this because, if we see a 1 over here, if we go there and if we see a 0, if we come here.

And then if we make this is final state, this is not a DFA, which is accepting this, because this is accepting the string like 0 0 1 0. So, 0 0 means we go here, then we go here yeah this is this is not a. So, this is one example, where this is although this is having two 0 0 consecutive substring, but it is not accepting. So, this is not belongs to this L of A. So, this is not a DFA we are looking for. So, this type of trial and error method we need to do.

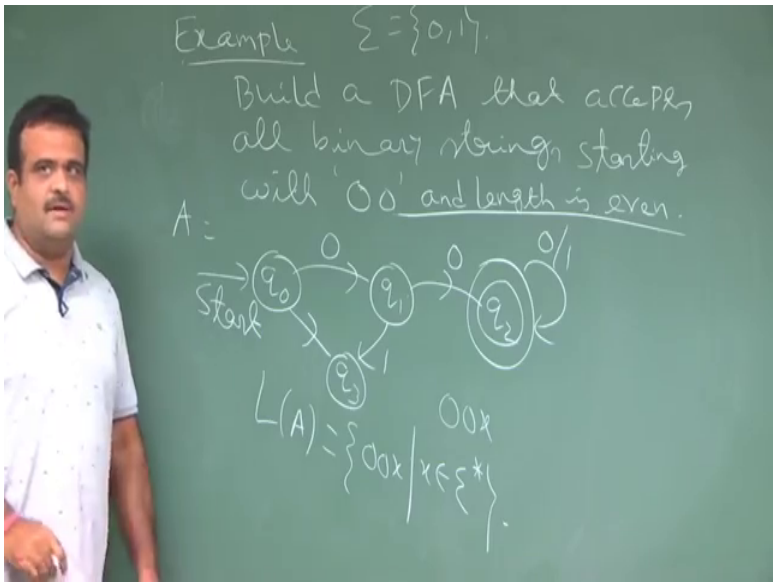
(Refer Slide Time: 13:29)



So, we will take some more example. So, suppose some more example to build a DFA. Suppose, you want to build a DFA that accept that accept all binary string all binary strings with 0 0 as a starting with 0 0 as a substring with 0 0 by there a string starting with 0 0 binary string starting with 0 0 starting with 0 0 ok.

So, suppose you want to construct a DFA. So, this is our language, our language is like this. So, it is 0 0 starting with 0 0, then any x and x belongs to sigma star. Here sigma is also binary string. So, we want to see whether this language is regular or not that that means, we want to construct a DFA, which can accept this language.

(Refer Slide Time: 15:11)



So, how to do that? So, now we can try to have a DFA like this. So, q 0 is the starting state. And then a can from q 0, we can go to with 0 move, we can go to q 1. And if it is 1, then we are we should not reach to a final state. So, this is q so this is a q 2 final state. So, if we just take another 0, we should able to reach to a final state, because 0 0 is the starting.

Now, if it is 1, then we should not able to reach two so this is something like dead state. So, this is q 3 ok, now from here also. If you see a 0, we are going to final state. If you see a 1, we are going to death state. And from here you can just or any input. So, this is the language this is the language, which is this is the DFA deterministic finite automata, which is accepting this all the string like this. All the string ending as 0 0 starting with 0 0 0 0 x ok; so, this is our A so this is our A. So, L of A is nothing but 0 0 x, x belongs to sigma star. So, this is a regular language.

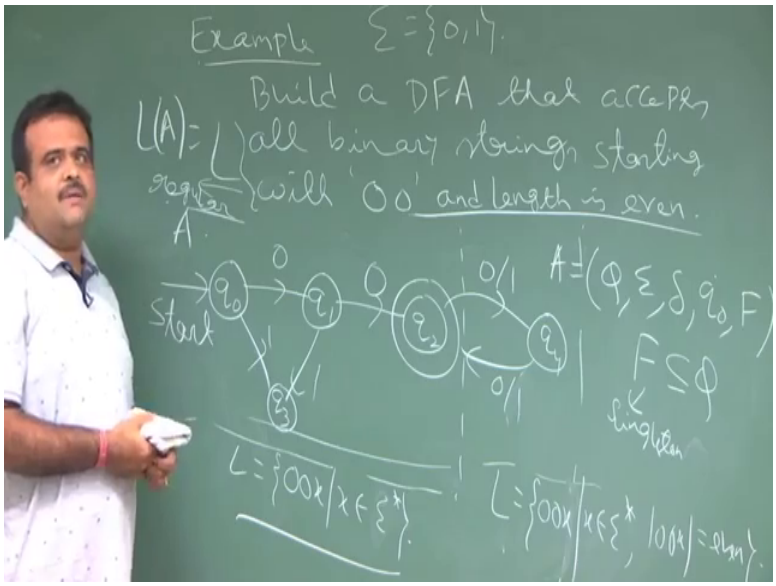
Now, can we extend this like we need to so this is with starting is 0 0. And also we want then length should be even, we want to extend this little more. And the length of the string is even this you want to see whether we can have a DFA to accept this. So, this will not tell us length is even. So, can you extend this yeah, so this will have a sub part of that?

(Refer Slide Time: 17:31)

So, we start with this state, length is even we know we want. So, if we have a 0 we reach here, now then if we have this is the final state 0, we reach to the final state, because this is 0 0 as a starting and also length is 0 0, so 0 0 is even so you reach. But, if you see a 1 here, so we must go to this q 3 and from here which we have a 1; because that is anyway that is not 0 0 starting. Now, here you want length also.

So, for that what we do, we will take so after this we have full filled, but we may have we need to count the length. So, like if we have a 0 0 1, this should not be in this L, because length of this is 3 which is not even, so we need to have that. But, 0 0's 1 0 this should be accepted by this. So, we are looking for a DFA, which accept the all the string which is starting with 0 0, and which is having even number of string. So, for that what we do, we just add another state say q 4. Now, if you have any input after this 0 0 will go there, and then with any input we will come back here. We will come back to here, just to make sure that the length is this. So, yeah that is all.

(Refer Slide Time: 19:17)

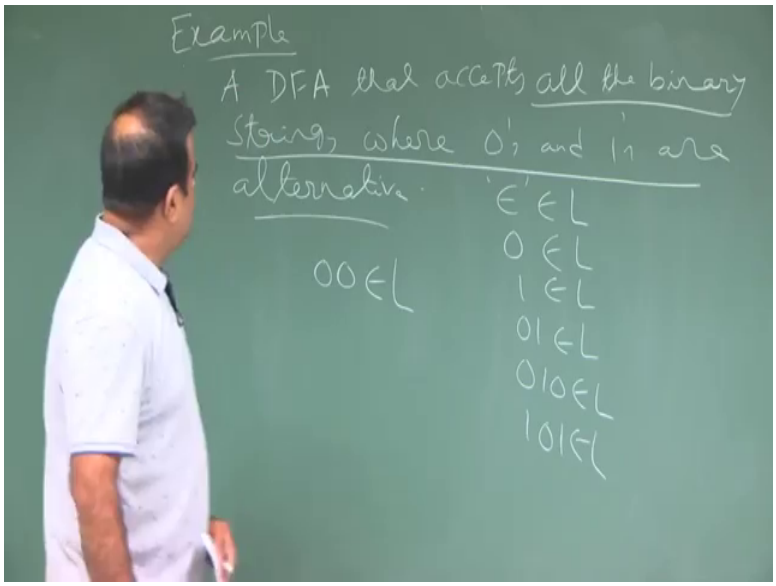


So, this is so this part you have seen. So, this part is the string, which is just this is the language, which is starting with 0 0, we do not have any restriction on the length. This is the string this is the DFA, which is accepting the all the possible string, which is starting with 0 0. Now, if we extend up to this, then this is the language which is having the 0 0 as a starting and so x belongs to sigma star, and the length is length of this is even. This is another restriction, so length is this is even number.

So, up to this the DFA, which is accepting the all the string who is starting with the string alphabet 0 0, now up to that up to that now we are looking for the that is that property is there along with that the length of the string is even. So, this is the DFA, which is accepting this language. So, this language is a regular language ok. So, this is another example to construct how we can construct a DFA.

Now, the question is whether we have a unique DFA I mean to for a given string that we will see. And so far we have consider the final state as a single strong state said F is a say the so far the DFA we have consider Q, sigma, delta, q 0, F. So, far you have consider F is a single term set single element single term that means, this is q 2. But, F could be I mean F could F could contain I mean many states as a final state, so that example we will see now. So, where F is containing more than one states. So, we have more than one states, which are which are accepted state or the final states ok. So, these will see now.

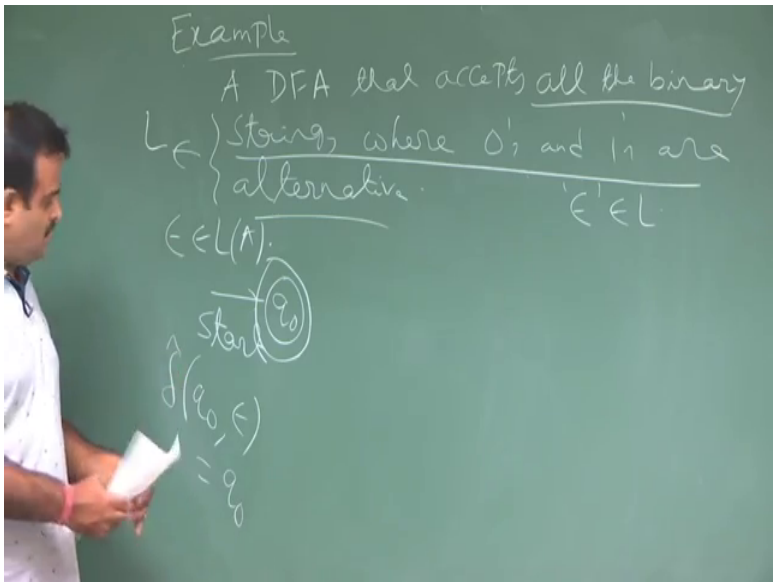
(Refer Slide Time: 21:51)



So, suppose you want to have a DFA we want to have a DFA, which that accepts that accept all the binary strings all the binary strings where 0, 1's are alternative and 0's and 1's are alternative. It is coming 0 1 0 1 like this are alternative. So, this is the language. So, we want to this is the language all the binary string, where this is alternative.

So, null string is also a member of this language, then 0 is member of language, one is also member of language. So, 0 1 like this alternate 0 1 0, 1 0 1 alternative this is a happening. So, this is not belongs to this is not alternative. So, if we have a zero, then the next thing will be must be should not be 0 again, it has to be 1. If you have a 1, the next thing should not be 1 again, so that is the meaning of this. So, you want to have a we want to construct a DFA, which is accepting this.

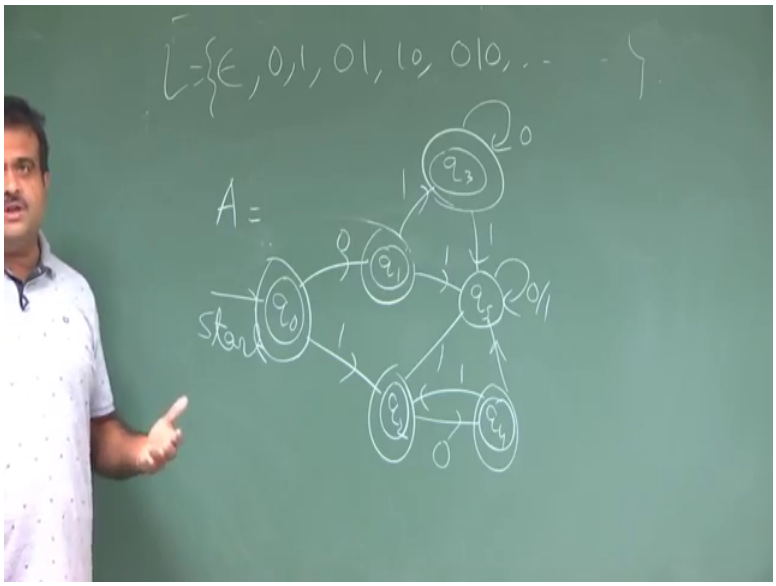
(Refer Slide Time: 23:41)



So, in other word we have to check whether this language is a this particular language L is regular or not. So, for that we need to have a we need to construct a DFA ok. Now, the question is so here epsilon is also a epsilon is also a member of this, because alternating means nothing has started I mean null string. So, this is also we need to accept epsilon, but in DFA there is no move for epsilon. But, in order to accept epsilon, we have to put the starting state also the final state that is the only way we can accept epsilon.

In the DFA, this is our starting state q 0. So, if we have to if we have to include epsilon in this, so we have to make it final state, because epsilon so delta of delta hat of q 0 epsilon is nothing but q 0. This is the definition how you read epsilon, so that means this is because in DFA there is no epsilon move. We will talk about epsilon NFA, where you have epsilon move nobody in the DFA there is no epsilon move, it is deterministic. We have been we have a input 0 or 1, we have move for every move for 0 1 at the present state. So, this is q 0, so q 0 has to be a final state in order to accept this epsilon as a string of this language ok. So, we are making this as a final state.

(Refer Slide Time: 25:31)



Now, we just this is the alternative 0 and 1. So, we can have like this. So, q 0, q 1, then if we have a 1 0 will go there; q 2 if we have this will go there; then q 4 so if we have from q 2 if we so this is 1. So, now we must see a so this is also accepted, this is also accepted, because this is alternate.

Now, if we have 0 over here, we must go here; this is also accepted state. But, if you see a 1 over here, we must go to some state, which is should not be accepted, because this is 1 1. And if we see a from here, if we see a 0, then we must go there. But, if you see a 1, we should go to the accepted state. And from here, if you go 0 and 1 same, because you already destroy the alternate property of that and then we can have a another state q 3 from which we can if we see a 1, we should go here.

So, let me just rub this. So, if you see a 1, we will go there. And this is also so let me just write it here. So, we have a q 3, if we see a 1, we go there, and that should be also accepted state, because this is alternate 0 and 1. And then, but if we see a 1 again, then we must go here that is the dead state. And from here also if you see a 0, we must go here. And if we see a 0 over here, we came from here. So, this is our A; this is our DFA, which is accepting all the string like epsilon 0, 1 then 0 1, 1 0, 0 1 0 like this alternative 0 and 1's. So, this is the language, which is accepting by this DFA ok.

Now, we will see whether this is a unique construction or not? Is there any other DFA, which is accepting these are ha having how many states? These are having 1, 2, 3, 4, 5, 6

states. So, now we want to see whether we can minimize this, I mean minimize in the sense that we can have a smaller states DFA, which is accepting the same language, so that we will discuss in the next class.

Thank you.