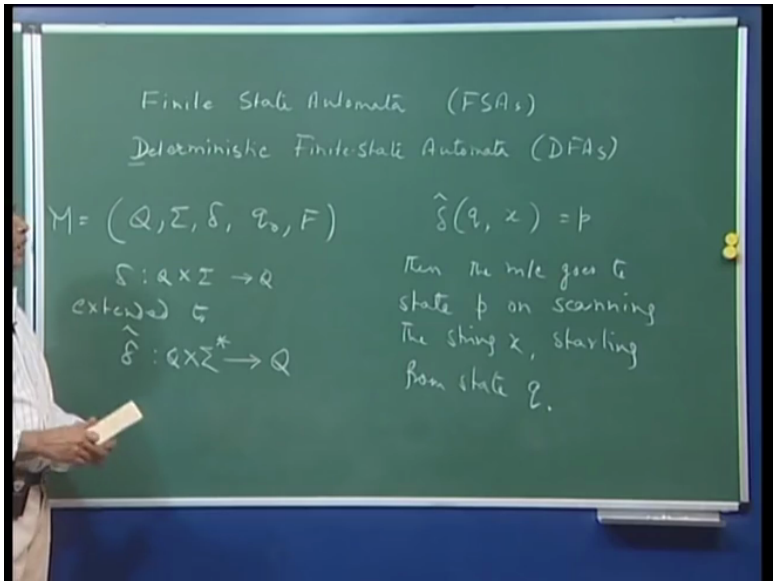
Theory of Computation
Prof. Somenath Biswas
Department of Computer Science and Engineering
Indian Institute of Technology Kanpur
Lecture 3
Finite Automata Continued, Deterministic Finite Automata (DSFA's)
Language Accepted by DFA

(Refer Slide Time: 0:34)



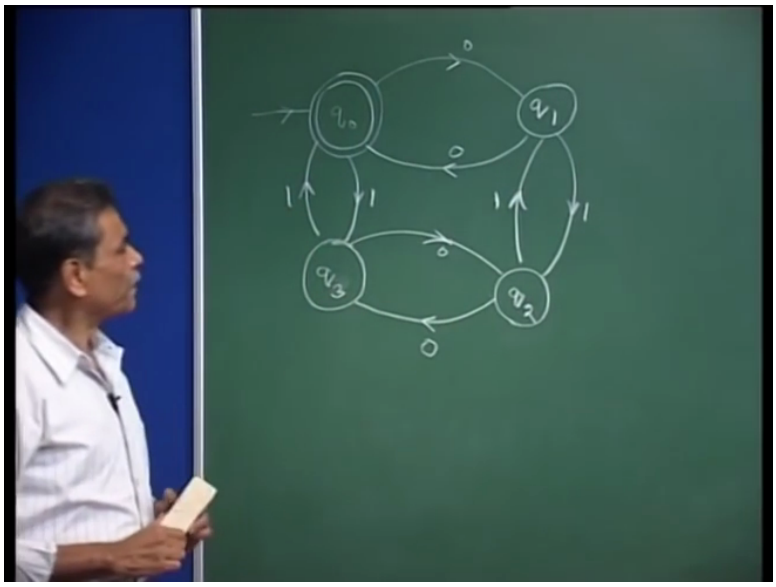
We are discussing finite state automata finite state automata these are abbreviated as FSA's. The particular form we are discussing now they are also called deterministic finite automata deterministic finite state automata (DFA's) and as we have mentioned either we call them automata or plain simply machines this FSA or DFA we can call these now and I will tell you later on while it is called deterministic but for the time being our model recall is such a machine M has 5 components Q, Sigma, Delta, q0 and F.

Q is a finite set of states, Sigma is the alphabet, Delta is the transition function which is a map from Q cross Sigma to Q and q0 is one particular element of the set of states, this is the initial state and F is a subset of Q which we call final states, set of final states. Last time we also saw how the function Delta which was a map from Q cross Sigma to Q. In other words which tells me if the present state is something symbol being scanned is something then what will be the next state?

This extended to Delta hat which was a map from Q cross Sigma star to Q and the idea was Delta hat will take as the first component first argument state and the second argument is a

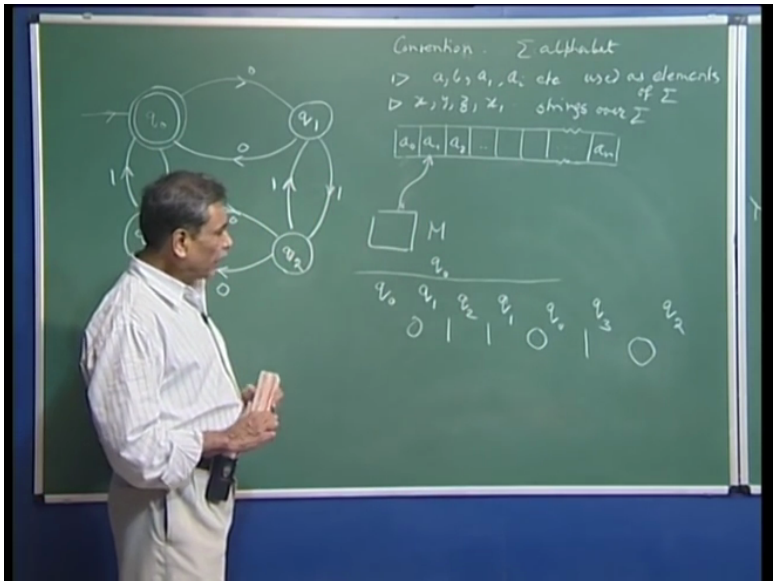
string, so remember x is a string which is an element of Sigma star and if it is p then it means the machine goes to state P on scanning the string x starting from the state q. We give an inductive definition for this function Delta hat.

(Refer Slide Time: 4:59)



Now a very important use of Delta hat is to formally define what is the language accepted by a machine? We will come to that concept but before that let us also keep in mind what is our mental picture of such a machine? We think of, by the way this was the example that we had seen and we will discuss from the concepts which we are going to give now in terms of this machine that we have seen earlier, this edge was missing. So this is of course this is called the transition diagram of the FSA transition diagram of the machine.

(Refer Slide Time: 5:15)



But imagine this machine one way of thinking of this, is that the input is given written on a tape and this tape has number of squares, so this tape is nothing but 1 square after another, so there can be something and the symbol here maybe a0, this is a1, a2 and this is an, incidentally at this point we should make one standard notational convention clear that when I use.

So let me write this convention here, remember Sigma is the alphabet and when we write symbols or you know let us like a, b, a1, ai etc these are supposed to stand for elements of my alphabet as elements of Sigma, okay. So lowercase letters from the beginning of the English alphabet a, b maybe subscripted like a1, ai these are elements of Sigma, so that means these are symbols whereas when I use smaller case letters x, y , z possibly subscripted etc these are strings over Sigma, okay.

So now this is using this convention you should be able to immediately see when I write this what I mean is these are symbols each square in my tape holds a symbol. So this first symbol we are calling it a0, second symbol is a1 etc and now remember we said that the machine starts scanning from the first symbol. So supposing this is my machine M.

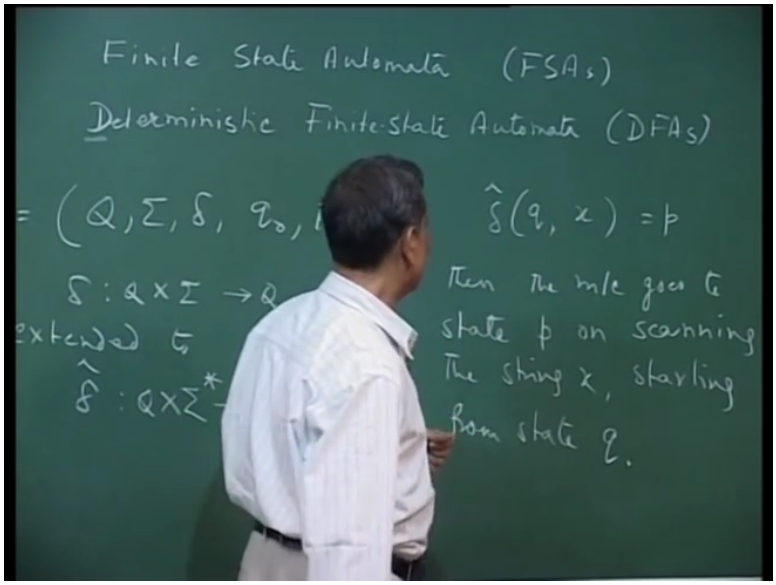
So this is a mental picture that there is some machine which has a head this is called the reed of the machine which at any given time will be at one of the squares of the tape and when it is at that square it is reading that symbol which is held in that square. So what happens when we start the machine like this because that time the machine is in its initial state right? Which is q0 and it scans a symbol goes to the next state, right?

And then it moves 1 square to the left, correct? So you can imagine the head is moving 1 square to the left and so on and also we said that when it finishes scanning the entire string it is obviously it will be in some state and that is the state in which the machine is after scanning the string on the input paper. So you see there are 2 sequences, one is of course the sequence of symbols which is in the string which is your input the other is the sequence of states through which the machine is going as it is scanning the symbol one after another.

So let's take this machine in particular and just right some input let us say 0 1 101 0, right? So this is supposing you give this as the input to this machine we see this input as the sequence of symbols and it is convenient to think of that the machine is going through a sequence of states and those states we will write in this manner, initially the machine is in q0, right?

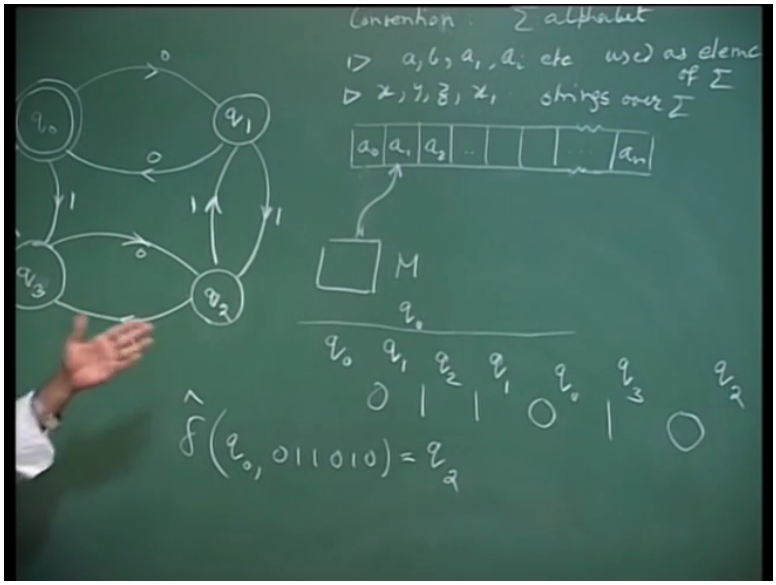
In the state q0 it is seeing the symbol 0 and therefore it goes to state q1, right? Now in q1 it is seeing 1 and therefore it comes to q2, right? In q2 it is seeing 1, so this is going back to the state q1. In q1 it sees 0, so it goes to q0 and in q0 it sees 1, so it comes to q3, in q3 it sees 0 so it comes to q2, right? So this is how of course you can define the sequence of states the machine is going through as it is scanning the input. This machine it initially started in x initial state and you see that as the input is over it finds itself in the state q2, right?

(Refer Slide Time: 12:17)



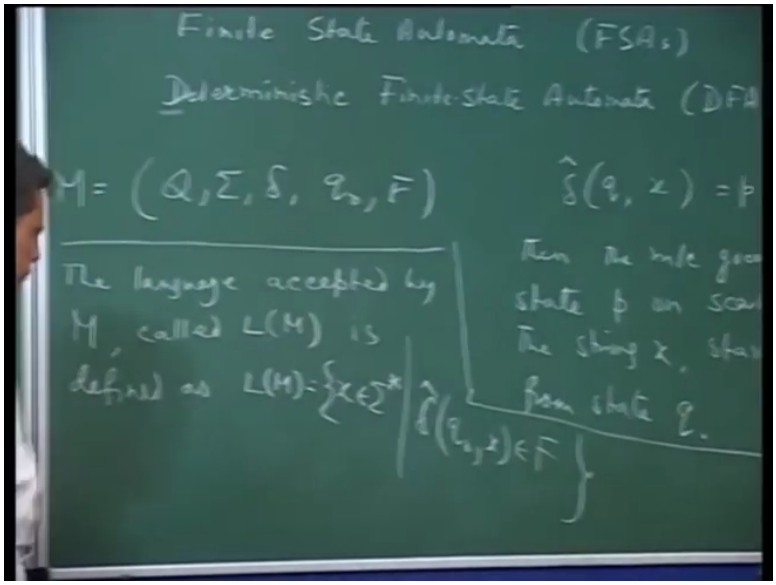
And in fact this function Delta hat tells me, so in this particular example if I use Delta hat because the purpose of Delta hat was to tell me which is the state the machine would be after scanning a string x starting from some particular state q.

(Refer Slide Time: 12:48)



So in this case in this example what we had just gone through, we can say Delta hat for this machine if it takes q0 and the string is 0 1 101 0 then it will be in state q2, right. I would like to point out to you the number of states in the sequence corresponding to a sequence of symbols which is in the input that number of states in the state sequence is exactly one more than the number of symbols and the input that's easy to see, why so? From this example or this illustration.

(Refer Slide Time: 14:01)

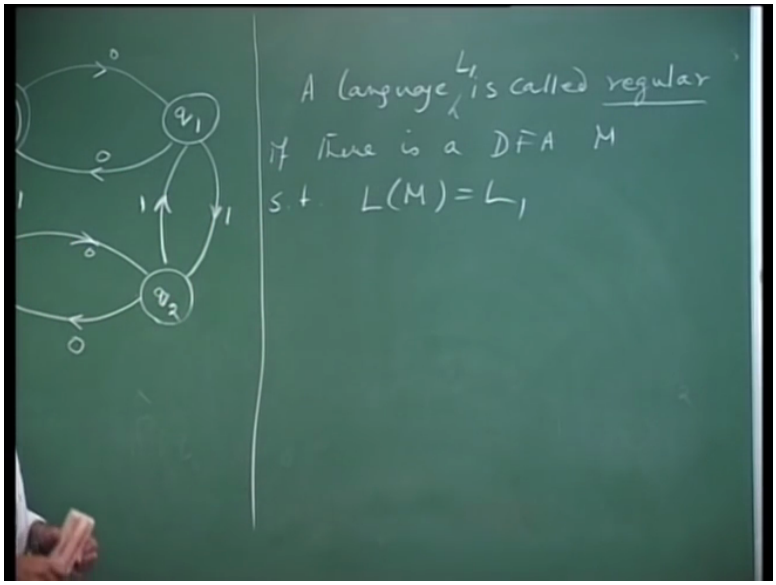


One very important definition is the language acceptant by a DFA or an FSA and that is done in terms of this function Delta hat. So let me say it right here, the language accepted by M, M is as this called LM is defined as LM all those x and because of our convention x immediately means I'm talking of a string over Sigma, all those x in Sigma star such that Delta hat of q0 , x is in F, okay.

So what is this definition is saying? LM is a language; language means a set of strings which are the strings which are in this language? It is precisely all those strings which will take the machine from q0 to one of the states which are designated as final states. Now either a string takes a machine from q0 to a final state or it doesn't. So this is one specific language, this is a unique language associated with the machine M.

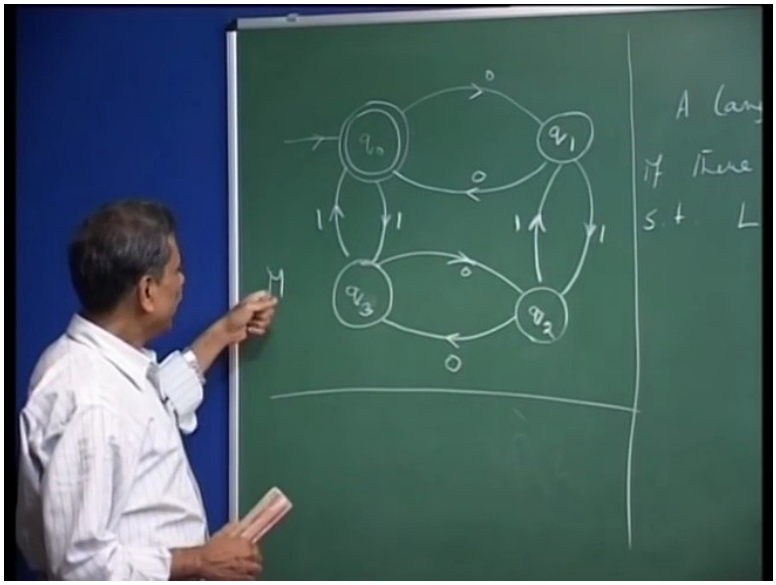
Now another definition which is very important which is the notion of regular languages, it is very simple, the regular language is a language for which there is a DFA to accept it. So in fact let me write down the definition we have done with this or here let me write it down.

(Refer Slide Time: 16:23)



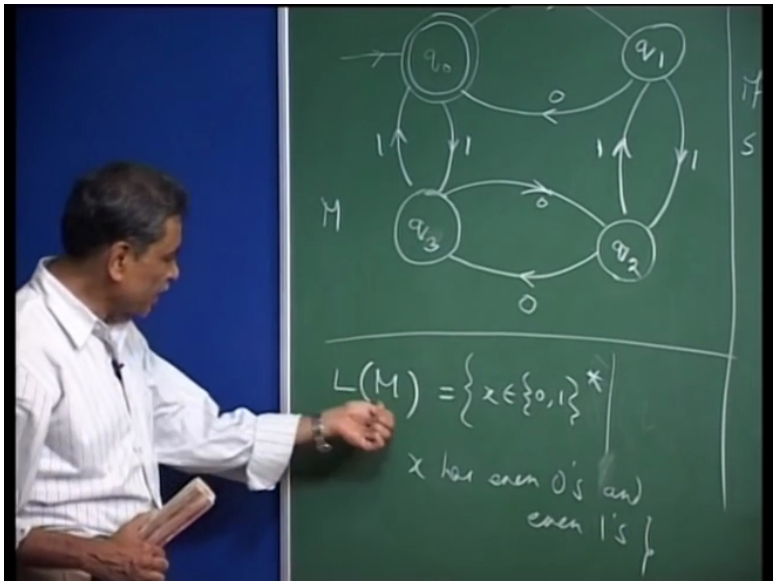
A language is called regular, so let me name this language, a language L is called regular if there is a DFA M such that language accepted by M is L. If these 2 L's are confusing, so let me call it L1, so this is L1. So what is the definition for regular languages? That a language is regular, if you can find a DFA M which will accept that language.

(Refer Slide Time: 17:29)



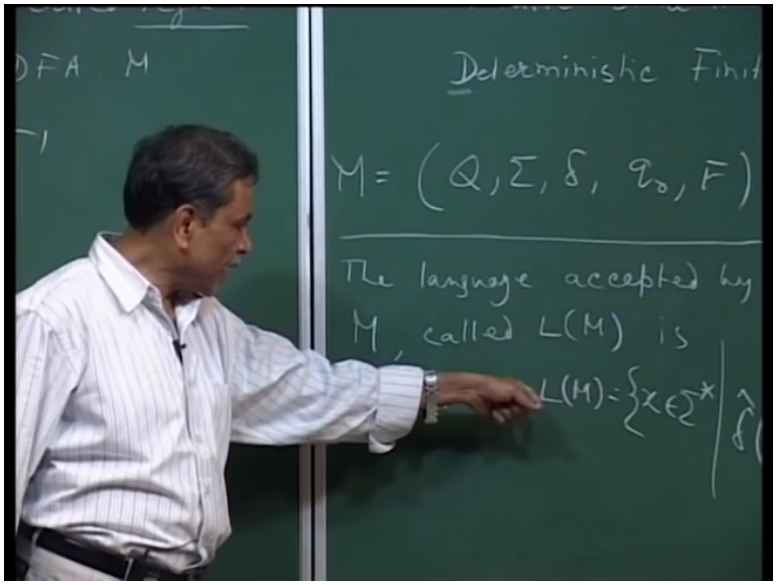
For example what is that language which is associated with this machine M, we know already, right? If you think of any string which has even number of 0's and even number of 1's that string will take the machines from q0 back to q0. String which does not have even number of 0's and even number of 1's will take the machine finally to one of the state's q1, q2, q3 those are not marked as final states. So therefore those strings are not in the language.

(Refer Slide Time: 18:10)



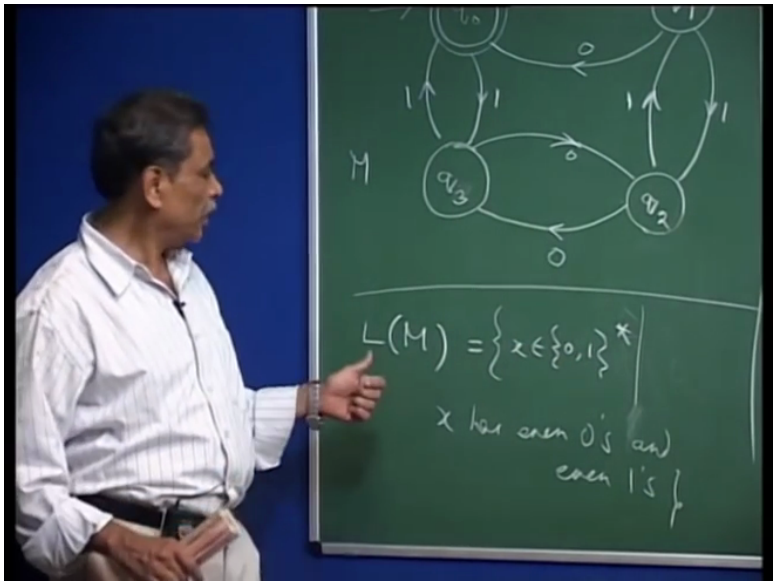
You can prove it very simply that the language accepted by this machine M is precisely the language that we wanted to wanted to accept. In fact let me quickly go to that proof because there is such a simple thing but these proof ideas, so I want to, these proof ideas are useful, I want to show that LM is such that x has even 0's and even 1's.

(Refer Slide Time: 19:12)



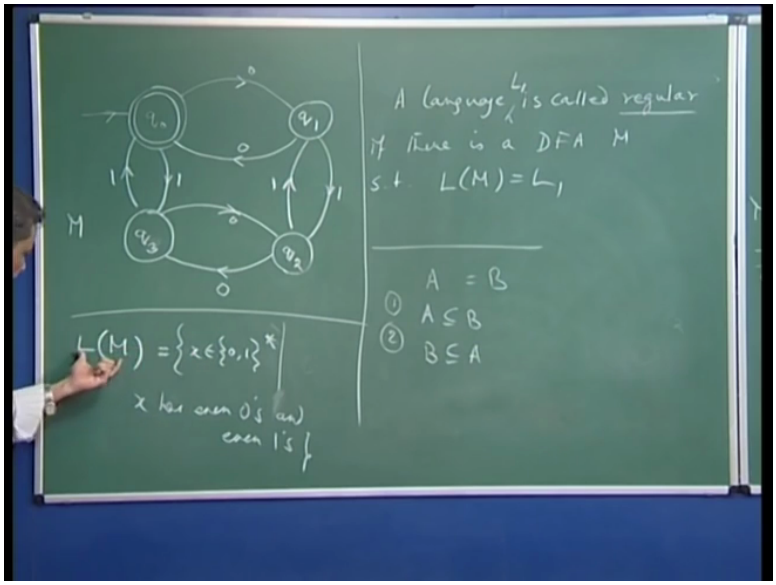
So this is an assertion that language accepted by M, remember the notion of acceptance we had defined here that language accepted by M is all those strings which take the machine from the initial state to one of the final states.

(Refer Slide Time: 19:36)



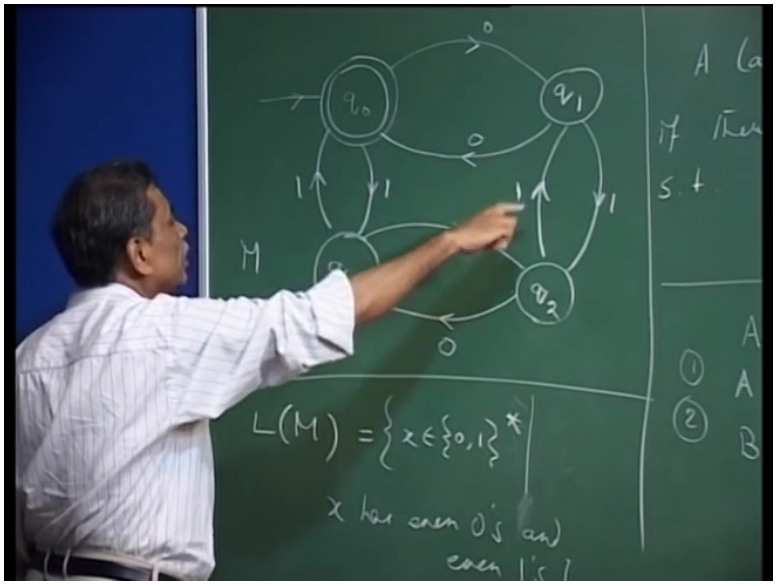
Now what we are making an assertion, we are saying that this set of strings is equal to this set of strings. How do you prove this equality? Since these are 2 sets and that is a very standard way of doing this.

(Refer Slide Time: 19:48)



Suppose A and B are 2 states and I am interested in proving A is equal to B then normally I will do 2 separate proofs that A is the subset of B as well as B is the subset of A. Now that is fairly easy to see here. So A is a subset of, this says is a subset of this that means what? I take an element from here and show that it is in this set also, take an element from here what is an element from here?

(Refer Slide Time: 20:44)



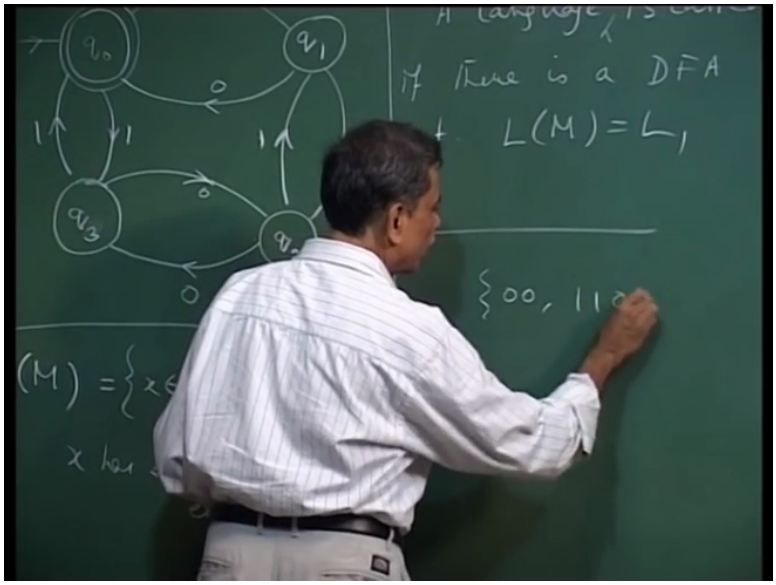
Is a string, that is given by here that in this particular thing that it's a string that takes the machine from q0 to back to itself. Now if you, you can argue that to come from here back to hear, it is it is necessary that the string has even number of zeros and even number of ones, right? And you can go back to what we said earlier that this is the state in which the machine will be after scanning even 0's and even 1's, right?

And you can go back to what we said earlier that this is a state in which the machine will be after scanning even 0's and even 1's and this is odd 0's and even 1's and so on, remember we have done that and if you do that little carefully then it is not difficult to see that any string accepted by this machine has even number of zeros and even number of ones, so therefore LM is a subset of this.

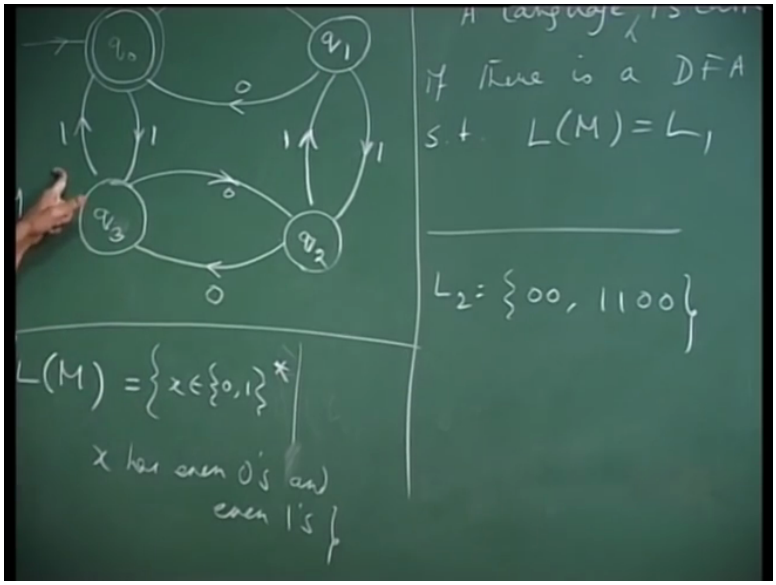
And now the other way take a string which has even number of 0's and even number of 1's, clearly if I had if it has even number of 0's and even number of 1's again if you if you think about the details about the states the assertions that we have made then it would be clear that such a string will take the machine back to here. So basically any string here is also an element of this.

Any string in this set is an element of this set, so this set is also a subset of this set. So therefore this is a subset of this and right-hand side set is also a subset of left-hand side set. So these 2 sets are equal, right.

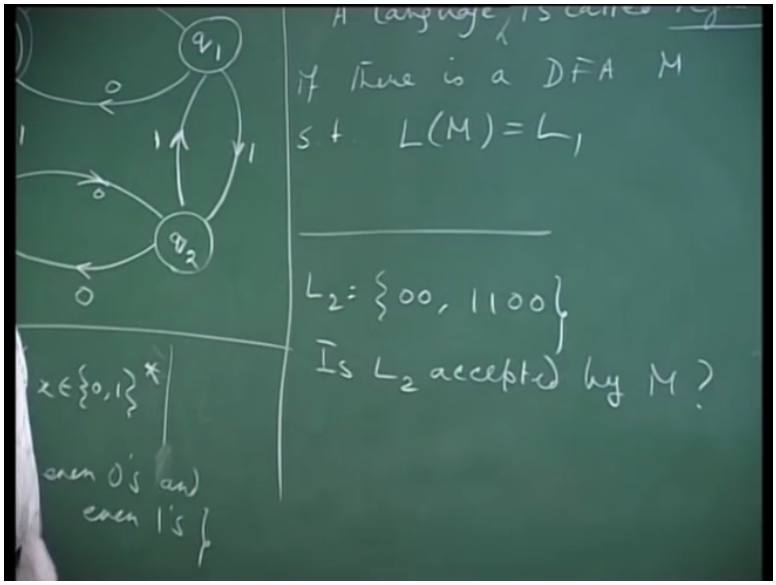
(Refer Slide Time: 22:34)



(Refer Slide Time: 22:44)

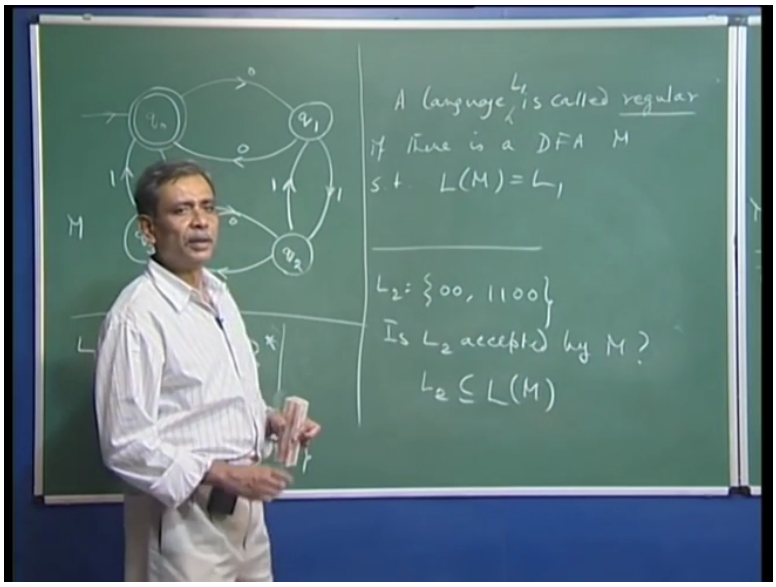


(Refer Slide Time: 22:59)



I warn against a misconception many students have in the beginning when they start reading these things, let's just take this case, this is also the language, so let's say this is the language L2, is this accepted by this is L2? To question asking, is L2 accepted by M? Now students sometimes argue that look every string here has even number of 0's and even number of 1's. So therefore L2 is accepted by M.

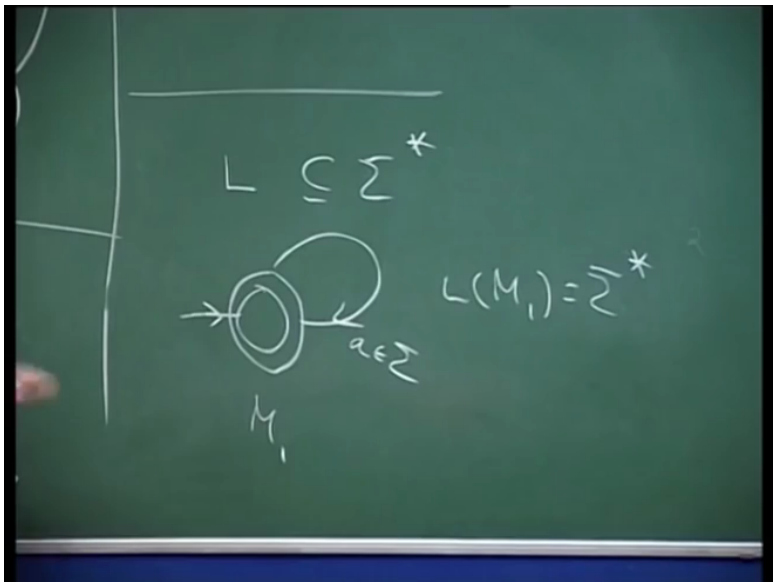
(Refer Slide Time: 23:53)



Is that the correct argument? It is not because you see, so LM is the language accepted by M and L2 is this language. Now the student just said that look every string in L2 has even number of 0's and even number of 1's, you told me LM is the language which is even number

of 0's and even number of 1's all that shows his argument just shows that L2 is a subset of LM is it clear?

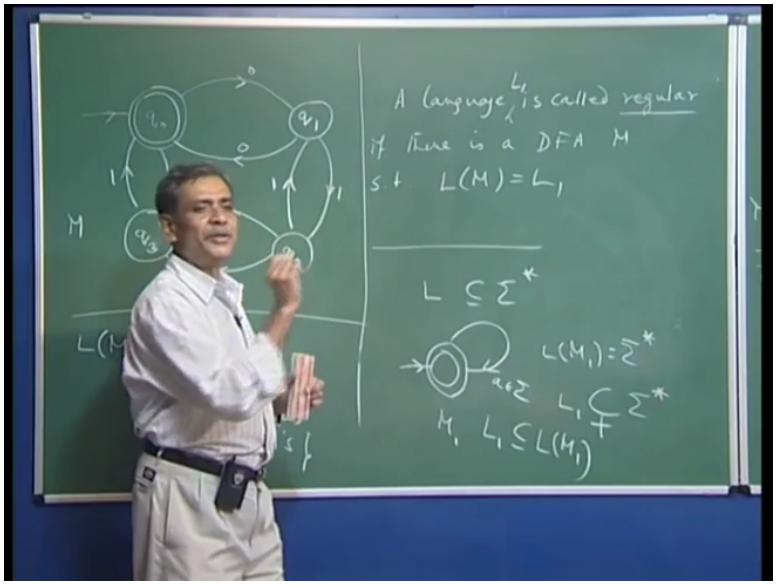
(Refer Slide Time: 24:27)



So just because all the strings of L2, also take the machine from initial state to a final state that does not mean L2 itself, L2 is not the language which is accepted by LM, LM is much more, LM is in fact an infinite set and this kind of difficulty students have been another way also, the other way also for example every language L is of course a subset of Sigma star by definition and think of this trivial machine.

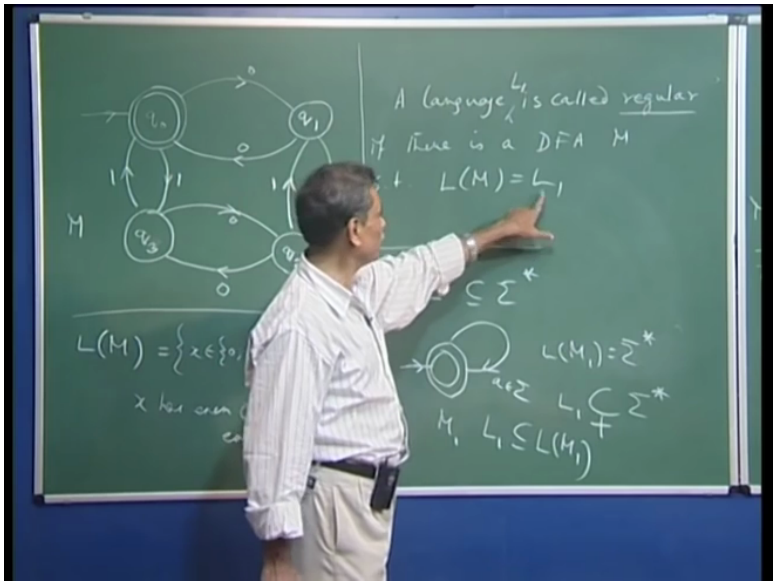
What is the language accepted by M1? If I look at that definition, LM one is of course Sigma star because it has exactly one state and that is the final state, so this is any string a in any symbol, a in Sigma, whatever be the symbol it comes back to this only state the machine has. So clearly every string takes the machine to this state to itself and final that is the final state and therefore Sigma star is the language accepted by this.

(Refer Slide Time: 25:32)



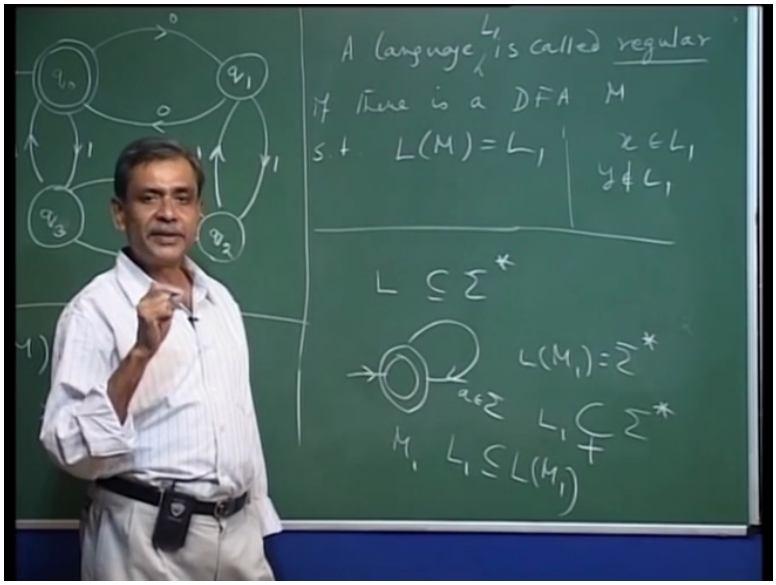
And now think of another language L1 which is a strict subset of Sigma star. Of course every string of L1, also takes this machine from initial to its initial to its final state that does not mean however that L1 is accepted by this machine. It just shows the same way that L1 is a subset of LM1, of course that is true. When I say and the point I'm trying to make is that with a machine there is a precisely one language which is accepted by that machine.

(Refer Slide Time: 26:48)



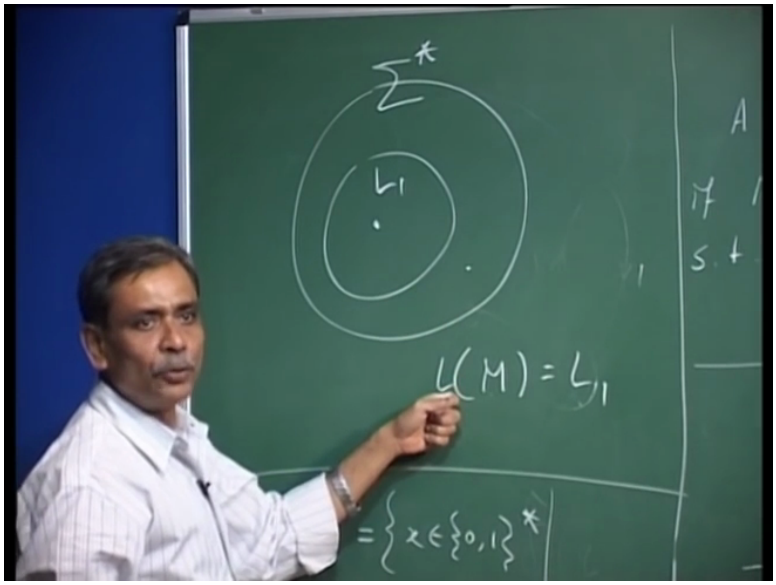
And when you say some language is regular you mean that I have a DFA, it is that unique set of strings that forms the language, not a subset of it not a superset of it. So please remember this that when I say a language L1 is regular, if there is a DFA such LM is equal to L1, I'm not saying LM is a subset of L1 or LM is a superset of L1 I'm saying equal to the language L1 and LM they are identical, okay.

(Refer Slide Time: 27:15)



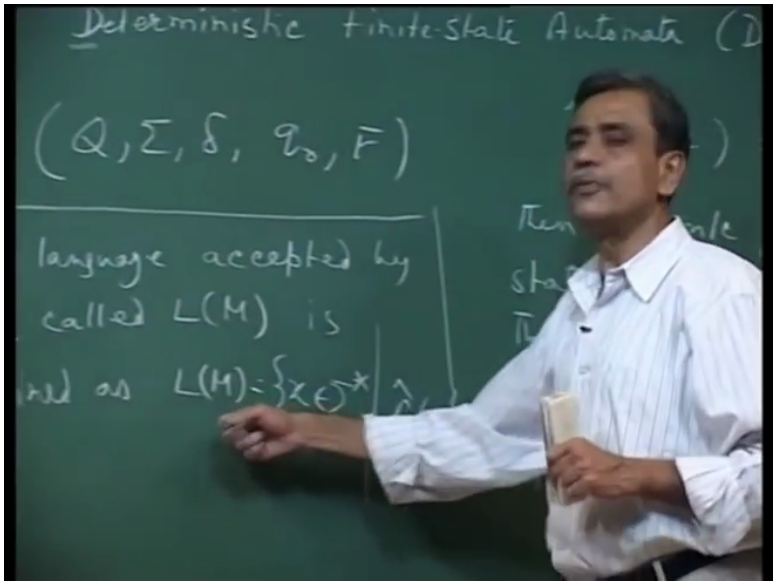
Another way of saying this that this machine M, if you take a string which is in L1 it will take the machine from initial to final state that is clear and if you take a string y which is not in L1 than that string will take the machine from its initial state to a state which is not final. So is it clear what I am saying? If we say that if M is the machine which accepts L1 that means.

(Refer Slide Time: 27:46)



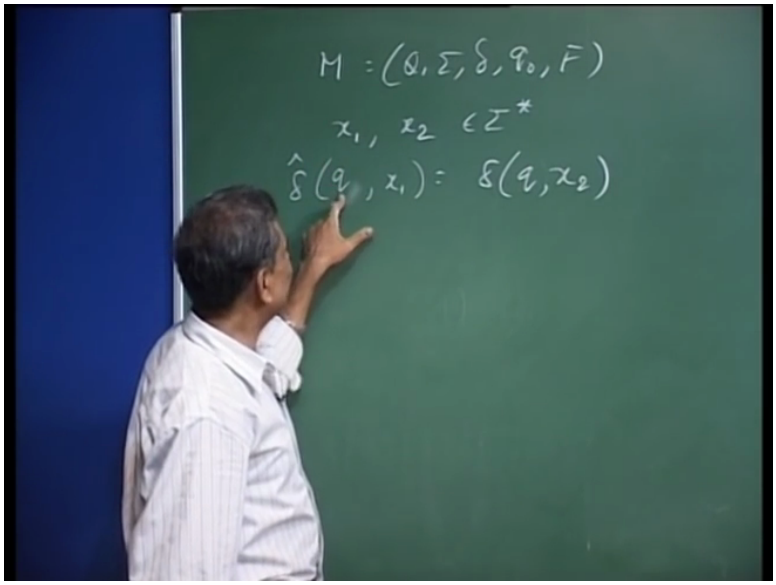
Now L1 you see this is the set of all strings Sigma star and L1 is here, if you say you're machine M is such that LM is equal to L1, so that means if you take a string from here then that string takes your machine from the its initial state to one of its final state and if you take a string which is not in L1 then that string will take the machine from its initial state to one of the non-final states and this is something we must keep in mind.

(Refer Slide Time: 28:48)



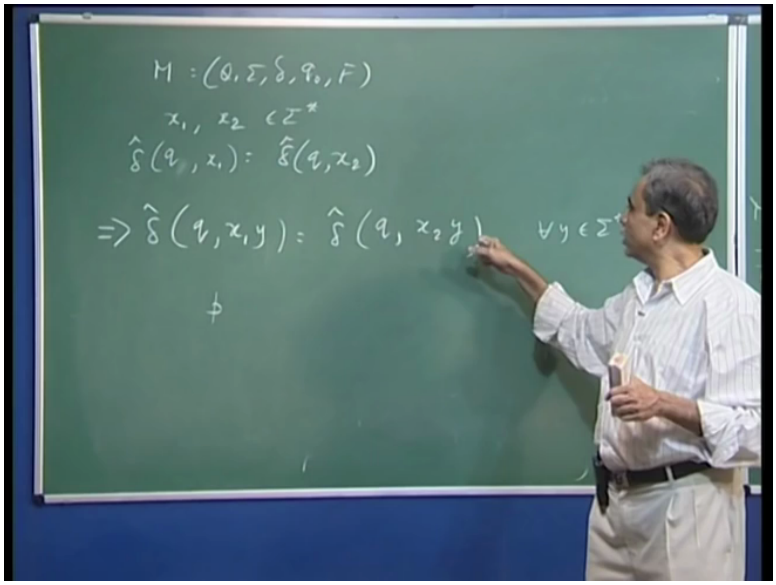
So I'm repeating again because this is one misconception people have initially sometimes and that misconception will not arise in your mind if you remember that with every machine there is a unique language associated with it which we call is the language accepted by that machine, okay.

(Refer Slide Time: 29:08)



So little more on what we're doing here, couple of things follow from this definition of in fact Delta hat. Suppose I have a machine M which is again Q, Sigma, Delta, q0, F and I have 2 strings x1 and x2 in Sigma star such that Delta hat of q0, x1 is same as Delta hat of in fact this is got nothing to do with the initial state what I want to say q, x2. So what I'm trying, this is asserting in symbols that both strings x1 and x2 take the machine M from the state q 2 some same state, right?

(Refer Slide Time: 30:20)

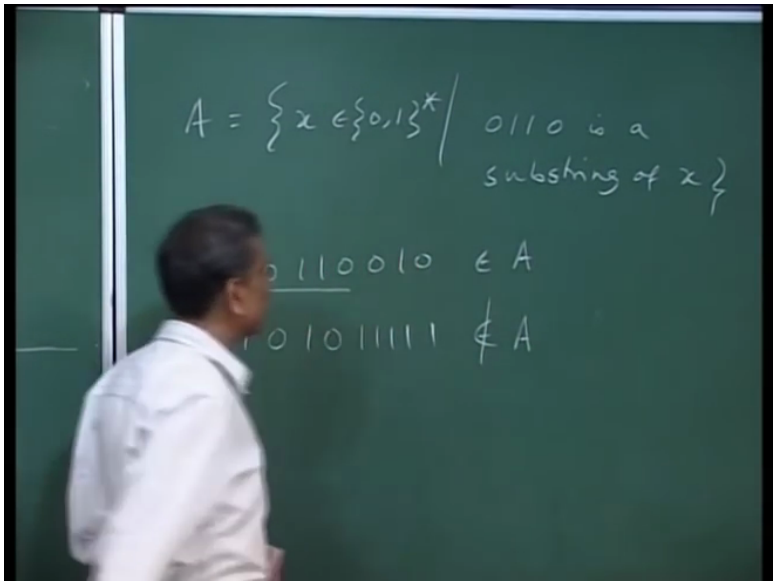


Supposing this is P, this is also P the state is P. Now do you see that this implies that Delta hat of q, x1, y is equal to this is Delta hat, Delta hat of q, x2, y for all y, it's not very difficult to see because what is happening is q, the machine M starting from q, x1 let us see to into P on x2 also it will go to the same state because of this quality assertion. So now from P, in either case it is seeing the string y and therefore this final state is going to be that state from which results starting from P and scanning this string y, right?

Then this is a very simple observation it follows directly from other definition of Delta hat but it has you know it says something very very important, it says that look if 2 strings took the machine to the same state then the machine has no way of remembering which of these 2 strings brought me in this particular state. So the machine after because the future of the machines behaviour will depend on the state, current state the machine is in and what is the string ahead?

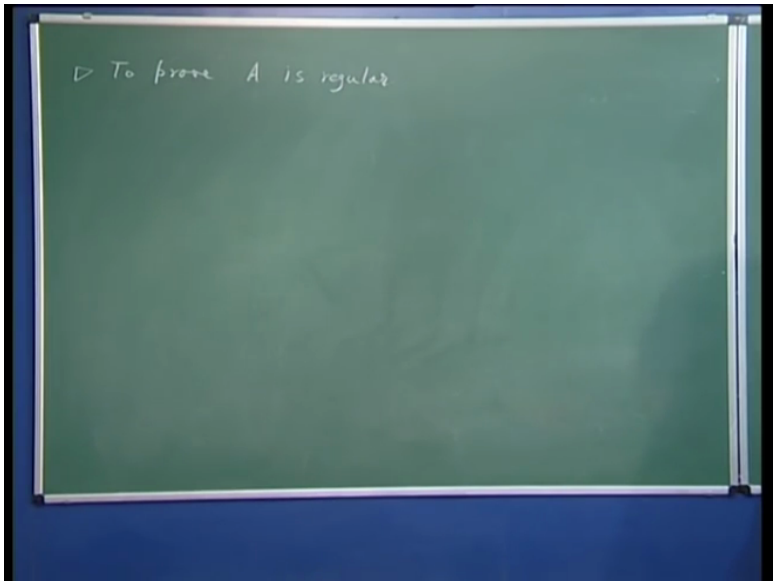
In both these cases the string ahead is y, so you see that it could not distinguish between x1y and x2y, okay. And that something we have to keep in mind when we design a finite state machine that what the machine can keep in its head is a finite amount of information and that information is encoded or kind of there in the state where the machine itself now is. So let's keep this in mind and try to design 1or 2 examples and then we will see is more properties of or properties of regular languages, okay.

(Refer Slide Time: 33:17)



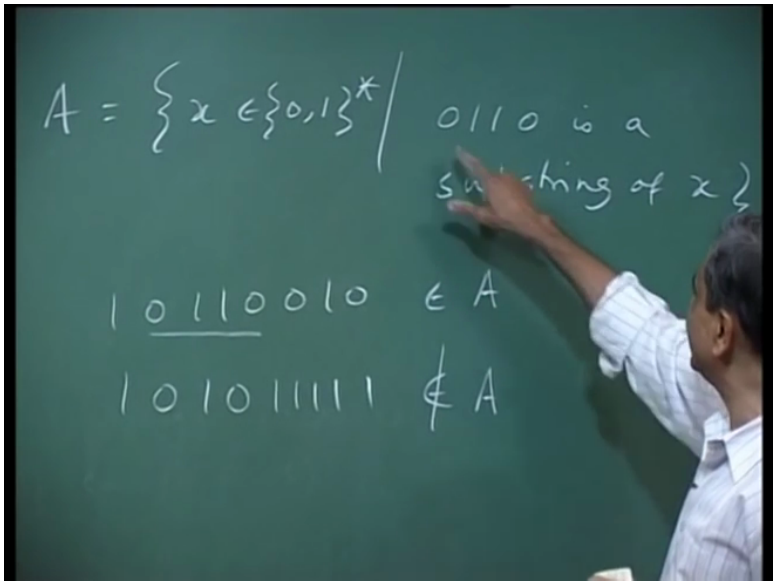
So let's take, so consider this language that is say A which is strings over binary strings such that 0 1 1 0 is a substring of x, okay. So this is a language A. So this is, is this in A or not? Yes because 0 1 1 0 is occurring and if this as a substring in this string and therefore it is language A. On the other hand if I take something like 101 0 1 1 1 1 this is not in the language because the substrings are you know folding, substrings are this, this and so on and none of the 0 1 1 0 is not occurring here as a substring. So this string is not in the language A.

(Refer Slide Time: 34:59)



And what I wish to do? Is to show; now we can use that particular word we want to show that this language A is regular. So let us say, remember what is the definition of a regular language? A Language is regular if and only if there is a finite state machine to accept that language. So in order to proof that this language A is regular, one way of doing it there are other ways of doing it which you will learn later.

(Refer Slide Time: 36:02)

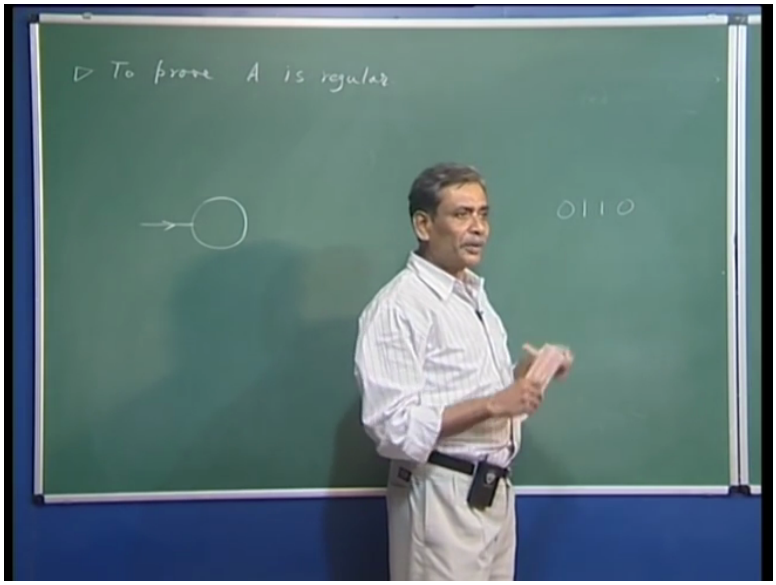


But one way of doing this would be to actually design a , a finite state machine, finite state automaton which accepts this language A and remember the notion of acceptance that DFA which I design should be such that every string which is the language A that means every string in which 0 1 1 0 is a substring, that string should take the machine from its initial state to one of its final states and a string which is not which does not have a substring 0 1 1 0 has to take the machine from its initial state to one of its non-final states, only such a DFA is will be considered to be accepting this language.

I am belabouring this point because that is an initial problem which some students, just this notion that what is the language accepted by a machine? So let us do this problem. So of course we must start with an initial state that is clear but how do I go ahead and design DFA, one way of doing it would be one one-way that most people find it convenient that imagine you yourself as that DFA, what you are going to do?

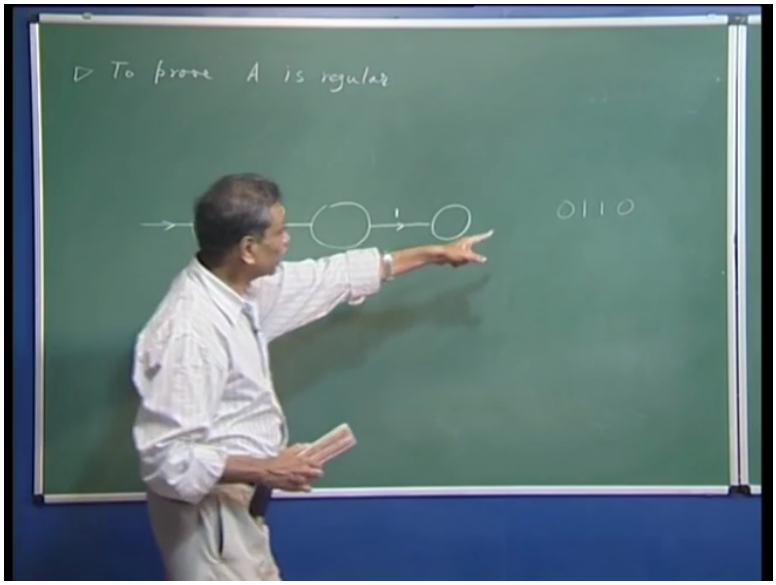
And in front of you, you are starting at the left end of a huge string possibly in this case the alphabet is binary, so it is a huge binary string and you are stepping from one symbol to the next symbol keeping only finite amount of information in your head and when you finish scanning the entire string you should be in a position to say whether that string is in the language or not in the language, okay.

(Refer Slide Time: 38:09)

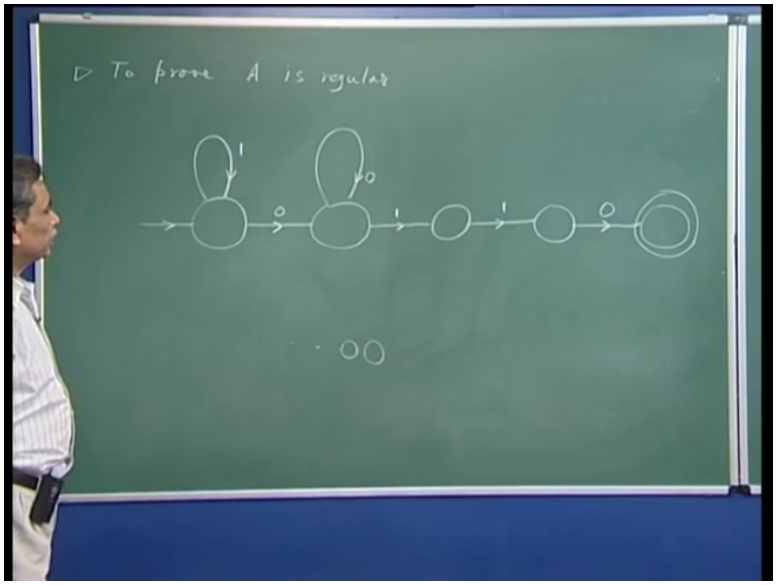


So alright this is the initial state and now let us say 1 comes, by the way what do I, what is the kind of information I should keep in my head in order to be sure whether or not the string has the substring 0 1 1 0 in it, so one way of doing it would be, clearly the most obvious way of doing it is whenever I see a prefix of this interesting string. So 0 1 1 0 is the string of my interest whose presence I am trying to find out the input, so if I see 0 then I should say oh! Maybe the next 3 bits will be 1 1 0 and then my string is in the language, right?

(Refer Slide Time: 39:57)



(Refer Slide Time: 40:12)



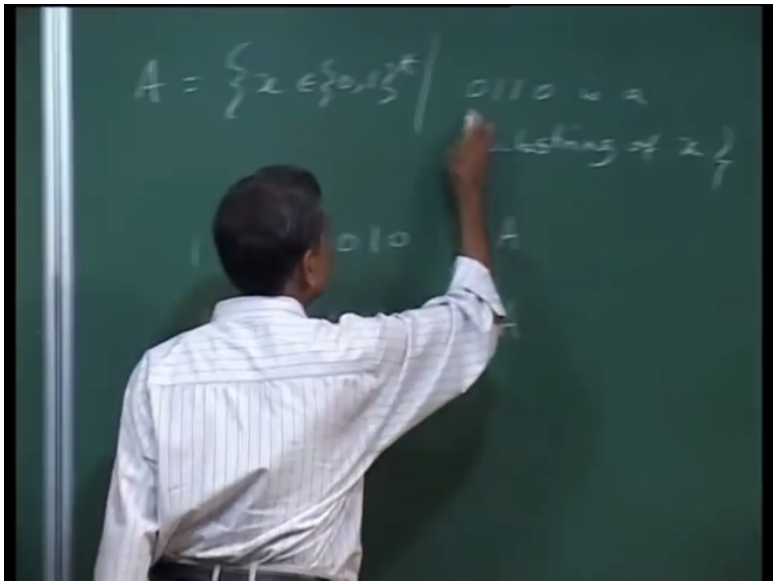
So clearly in that case my states can be something like this, yes I have seen 0, Oh! I have seen not just 0 but 0 1, Oh! I have seen 0 1 1 and oh! I have seen all 4 bits 0 1 1 0. So let us think in that manner and so 0 came you are in this state which remembers that I have seen 0 then 1 came in this state which remembers that I have just seen 0 and 1 and now let me look for the other to bits 1 and 0 and now 1 and 0.

If this happens then you must be in a final state that is clear but this is not complete picture of the DFA, why? Because I have not said what happens in this state? For example on input 1 what should you, what is the meaning of this state? Or what is the can you think of what is

the intuition behind this? Let us look I have not seen anything interesting, no prefix I have seen other than the trivial empty prefix of the substring.

So if one comes you remain in that state, right? In this state if one comes of course then that means I have seen 0 and 1 but if 0 came then what? So that means you know there are 2 zeros, so of course 0 0 is not a prefix of this. However this could be, the second 0 could be the prefix of the interesting string, right? So here you should if you see 0, it remains here in this particular state.

(Refer Slide Time: 41:55)



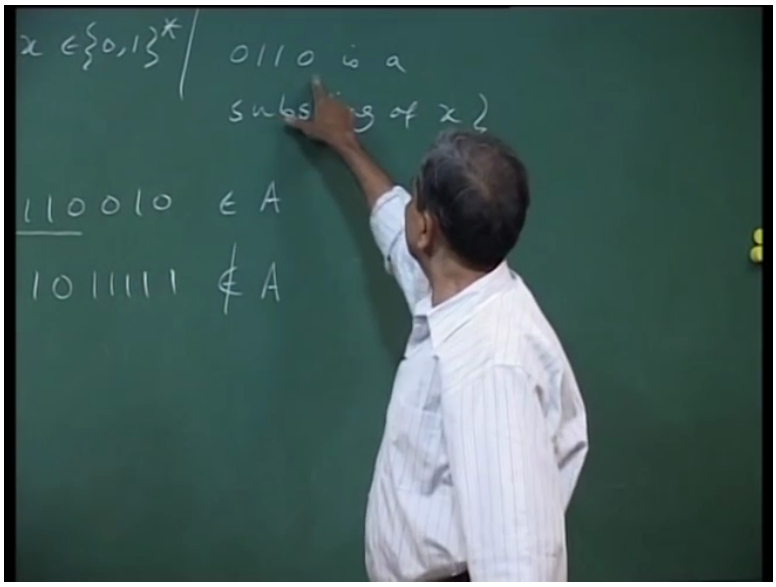
And now these 2 states are completely specified because aside what happens in this state when 0 came and 1 came, in this state when o1 came and 0 came I have specified, what about this state? So in this state if 1 comes, yes you have met progress you have just seen 0 1 1 but what happens if I have seen 0. So you are in this state that means you have seen something and then 0 1, it came 0 1 then in 1, right?

And now if 1comes you will be here, what happens if 0 is there? Well, this 0 could be the beginning, see this thing that you have seen the other 0 1 you have seen is of no value because this is not extending to the interesting substring, so this 0 I have seen now that is indeed a prefix, so I come here. Say this state remember saying that I have just seen 0 which could possibly be extended to 1 1 0.

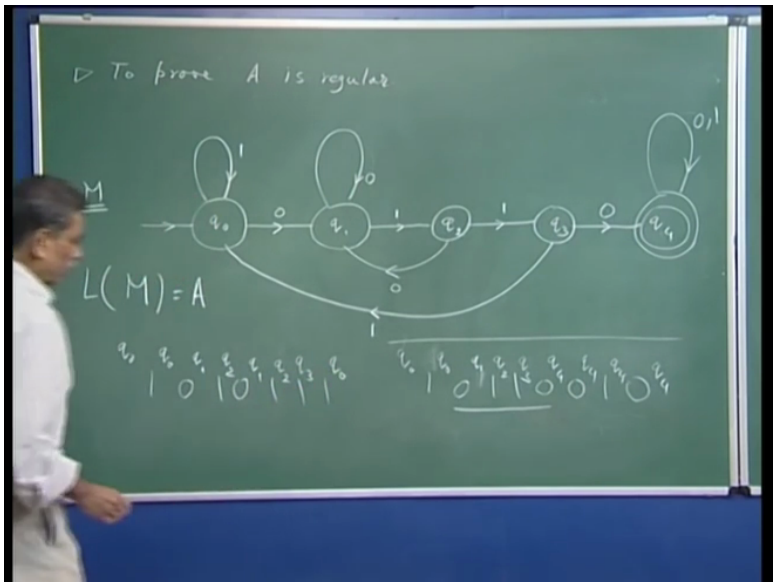
And now in this state if your 1 comes what should you do? You have, so basically the situation is the string is something then 0 1 1 and 1 came then what happens? Really you have

made no progress at all. So if your 1 came here we will go back to this state hoping that something in future will be the string that substring maybe but right now I started got something but then it became bad.

(Refer Slide Time: 44:11)



(Refer Slide Time: 44:29)

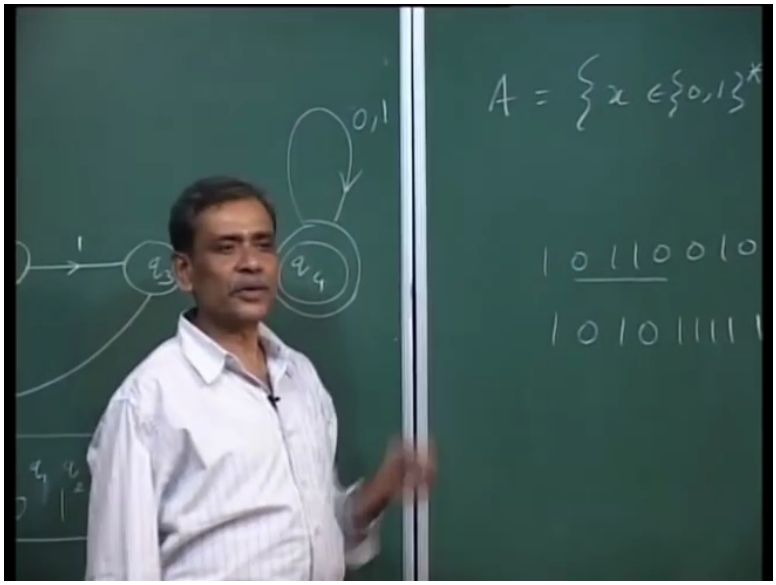


Of course in this state if 0 came then you have 0 1 1 0 and then it is fine but now in this state if 0 comes or 1 comes that means what? But then I have already scanned a first part of a string which has 0 1 1 0, so if any further symbols come that is fine. So both 0 and 1 will remain in this state. Now this machine M is it claim the language accepted if I say this is my machine M the language accepted by this machine is A.

Let us quickly see 1 or 2 examples, so let me call it q0, q1, q2, q3 and q4 and we said, so let us take this same string which is in the language 1, let me write it here 1 0 1 1 0 0 1 0. So this is the sequence of symbols in the input and we start from here q0, q0 on 1 I remain in q0, q0 on 0 I go to q1 from q1 on 1 I go to, it should not confuse this, this is q1 and the next symbol is this 1, q1 on 1, I go to q2, q2 on 1 I go to q2 on 1, I go to q3, q3 on 0 I go to q4, q4 on 0 I remain in q4, q4 1 q4, q4, 0 , q4.

So what has happened? The string 1 0 1 1 0 0 1 0, this string took the machine from q0 to q4, q4 is the final state and therefore this string is accepted by this machine and that is what it should be because this has the substring 0 1 1 0. On the other hand consider the other string which was not in the language 101 0 and let us say 1 1 something. So you started here q0, hereafter 1 it remains in q0 on 0 it goes to q1, q1 on 1 it goes to q2, q2 on 0 it comes back to q1, q1 on 1 goes to q2, q2 on 1 it goes to q3, q3 on 1 it comes back to q0.

(Refer Slide Time: 48:05)



So this string took the machine from q0 to q0 but q0 is not a final state, so therefore this string is not accepted by the machine. So therefore this is what as it should be and the way we proved earlier about the previous example we can formally prove this also that the statement that language accepted by M is precisely this language A.