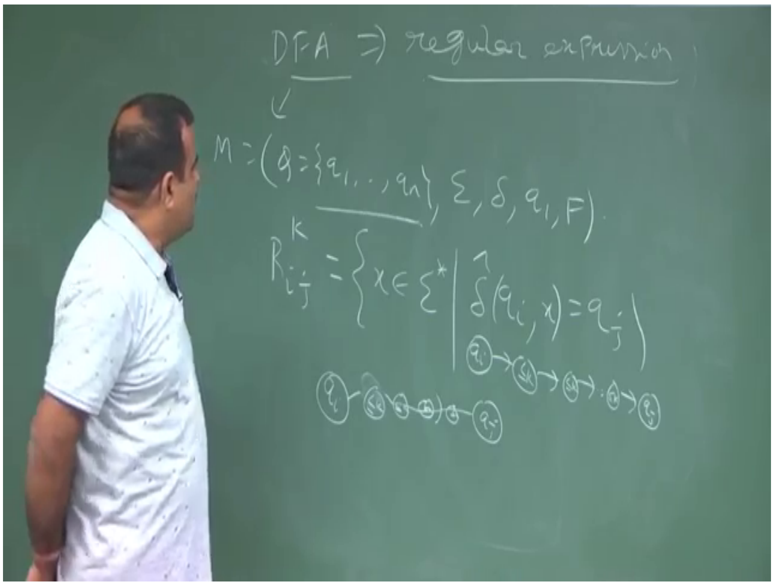
Introduction to Automata, Languages and Computation
Prof. Sourav Mukhopadhyay
Department of Mathematics
Indian Institute of Technology, Kharagpur

Lecture – 22
DFA to Regular Expression (Contd.)

(Refer Slide Time: 00:25)

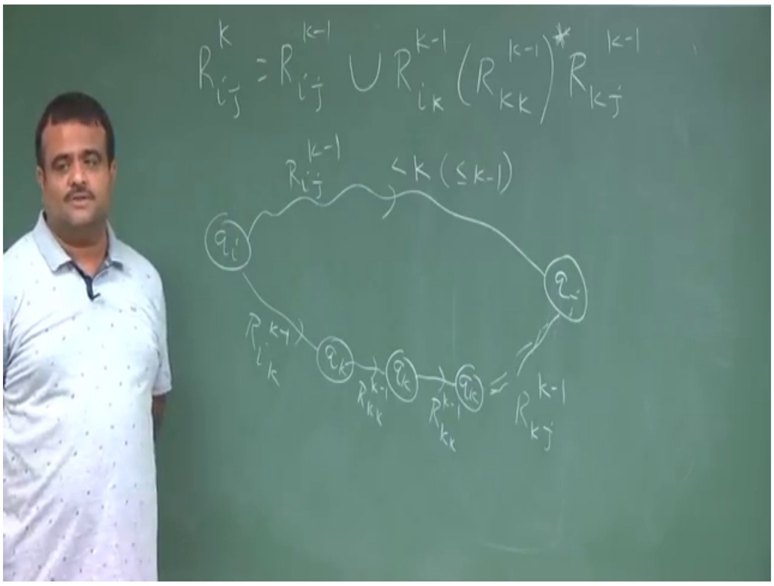


So, we are talking about the regular DFA to regular expression. So, we have seen given a DFA we can rename this states y, if there are q n state we can always rename the states by q 1 to q n and we have this sigma q 1 F ok

So, this is the so now, this is the set of all states which are finite state. So, we can always number the states by 1 to n. Now in the last class we have seen if we denote i j n is equal to set of all string such that delta had of q i comma x is equal to q j. It is starting from q i and it is going to q j and in the intermediate it is not seeing any node which is labeled by more than k. So, all the nodes over here which is seen here all the nodes which is seen here are less than equal to k, all the nodes are less than equal to k so that is the sorry this is k.

So, all the intermediate nodes which is seeing over here are less than equal to k. So, that restriction is there. So, we are starting from q i and we have visiting some of the notes finally, we are reaching to q j and all the nodes labeled here are must be less than equal to dot dot dot less than equal to k; that means, we are not seeing any note which is level more than k and there is no restriction on i and j, i j could be greater than k or equal to k there is no restriction on that ok. And we have seen the relationship recurs recursive relation for this R ij k as follows.

(Refer Slide Time: 03:05)



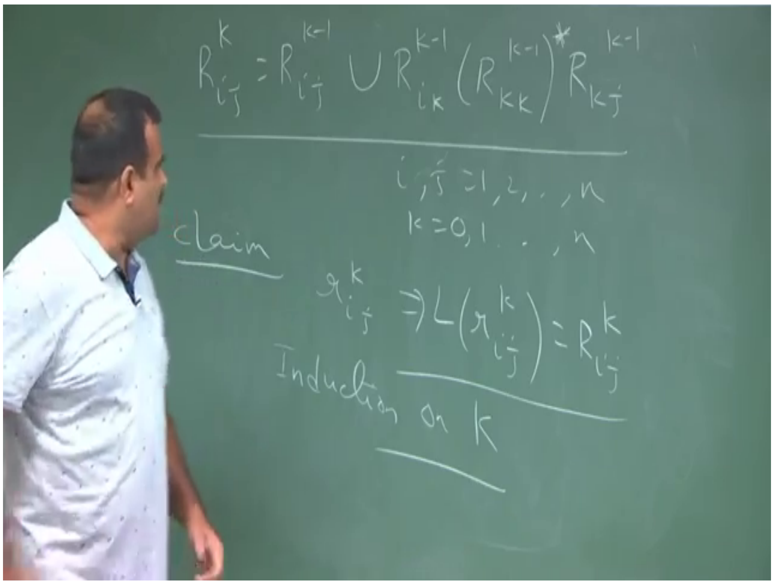
So, we this is R ij k minus 1 Union of R ik R ik k minus 1, R kk k minus 1, R kj k minus 1. So, this is the star. So, this we have proved in the last class this is basically the idea is we are starting from i to j and we do not want to see any nodes which is more than level more than k. So, there are 2 possibilities the kth level node may come q k or may not come. So, this is the path in which we are just seeing no loads. So, which is labeled by so, we are starting from q i we have to go to q j now this is one path in which we are not seeing any node which is q k.

So, that means, all the nodes here our label strictly less than k; that means, less than equal to k minus 1. So, this is denoting by this set of all string is denoted by R ij k minus 1, this is one possibilities now in the path we have no q k no kth node and another possibilities is we have a q k maybe one, maybe twice, maybe thrice in the path. So, we have q k, q k, q k like this. So, there is no q k in this path. So, this is set of all string denoted by R ik k minus 1, there is no q k here so, R kk k minus 1, R kk k minus 1 like, this if there is no q k over here this is R k j k minus 1.

So, this is this path and this is union set of all string this path. So, how it is so, we take the concatenation. So, we take x we go here then this will consider as a star, star means if it is z; that means, there is 1 q k only like that. If it is more q k then we have a star like this and then it is the last q k then from here we are going to q j without saying any q k in that corresponding path. So, this is the recursive relation of R ij which you have seen in the last class.

Now, we are going to prove that for each of these R i this is R i is basically a language set of all string which is starting from q i which is going to q j without seeing any notes which is labeled by more than k ok.

(Refer Slide Time: 06:26)

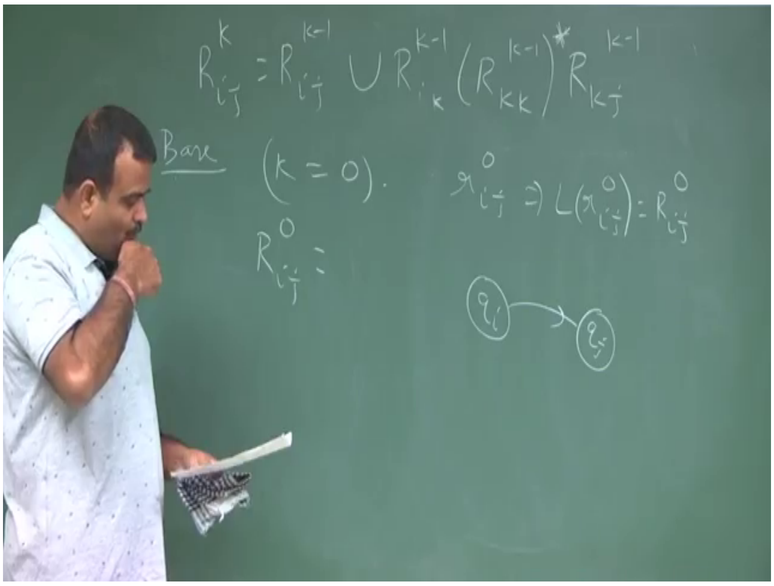


Now our claim is for each of this ij so, this is i is i and j both are 1 to n and k is, now our claim is for every ij and k we have a regular expression R ij k such that language of that regular expression is same as R ij k. This is our claim we are we are trying to prove that, I mean not trying we are going to prove that we have a regular expression there exist a regular expression R ij k such that the language of the regular expression is same as this set of all strings R ij k ok.

So, we will prove this by method of induction on k. So, we will prove this by induction on k for a given i j, we fix the ij now this is a this is something which is talking about on n, n is a natural number n if starting from here starting from z to n. So, by mathematical induction if you have to prove some statement is true what we need to show, we need to show the statement is true for this is a property. The statement is true for some lower value of k that is the base case say for k is equal to 0.

So, if we can. So, that we have a r i 0 small r i 0 and then there is induction hypothesis, we assume that the statement is true for k is equal to some value say k and then if we can prove that this statement is true for k plus 1. Then by the method of mathematical induction we can say that statement is true for all k so, that we are going to. So, first of all we need to show the statement is true for k is equal to 0 that is the base case.

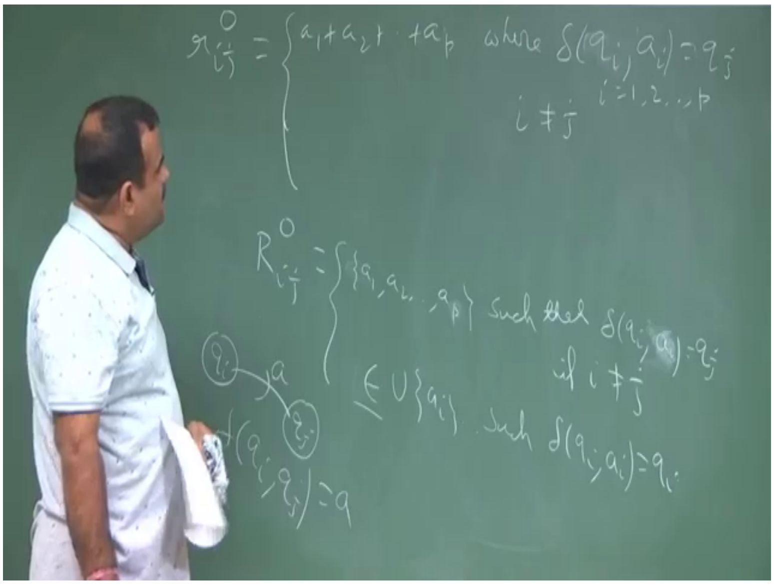
(Refer Slide Time: 08:44)



So, for k is equal to 0 we are trying to get a r i we are trying to get a we fix ij r i 0 such that is the R ij 0

So, what is R i 0, capital R i 0? Capital R i 0 is nothing, but we have to be just. So, we are at q i we have to go to q j with the intermediate nodes which is labeled by at most 0, but there is no nodes labeled by 0 now our states are starting from 1 to n. So, that means, this meaning is that there is no intermediate node, either there is a direct H direct arc if at all there is or it is empty or it is empty so; that means, how to write this. So, this is basically if i is not equal to j.

(Refer Slide Time: 10:15)

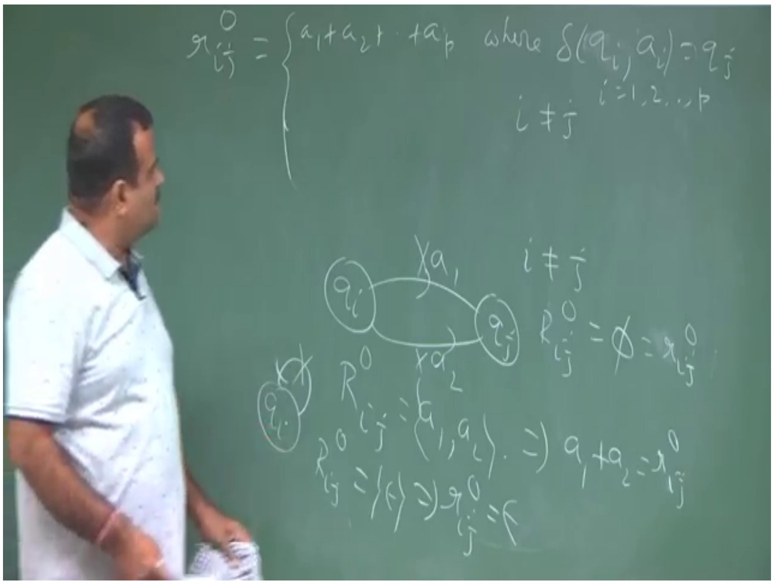


So, this is also all a 1. So, this is the set a 1, a 2, a r or a l what notice can I have used a p such that such that such that such that delta of delta of q i a i is equal to q j, delta of a i q i is equal to q j, when i not equal to j it is epsilon union of all they see is a j if same thing is happened such that delta of a i q i ok.

So that means, this is q i if i is equal to j then if there is some a i it will just a i otherwise if there is no strength loop it is just epsilon or else or else it is just if i not equal to j is this q i and this is q j if i not equal to j if there is a which is taking delta of q i q j equal to a then that a will come. So, all union of all such a is basically R i j.

So, then what is the small r ij 0. So, small r ij 0 is nothing, but. So, this union means this are union means that in the regular expression it is basically the plus all such a 1 plus a 2 a p where delta of a i sorry q i a i is equal to q j this i is from 1 to p and here i is not equal to 0; that means, if there is a arc form.

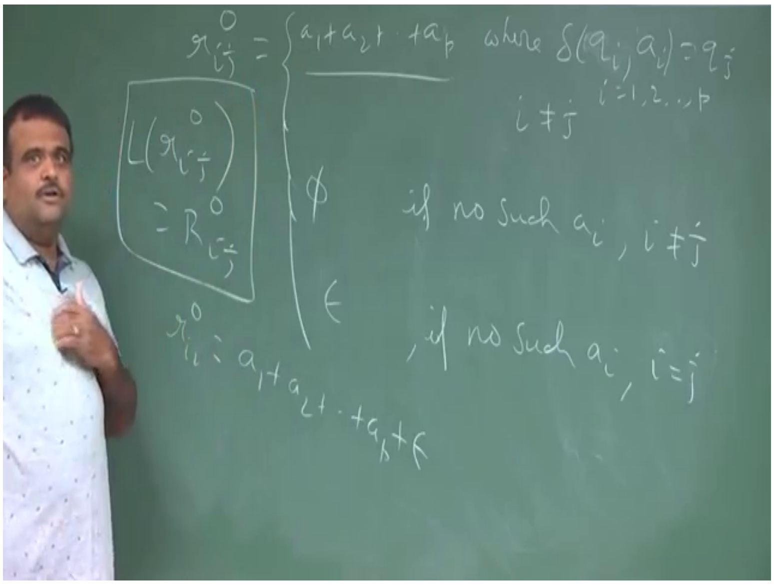
(Refer Slide Time: 13:29)



Say this is q i this is q j. So, there are 2 possibilities one is there is a arc say a and a 1, a 2 say this is one chance a 1 a 2, then what is our what is our R ij 0, R ij 0 is nothing, but a 1 union a 2, which is corresponding to the regular expression a 1 a 2 this is r ij small r ij 0 very simple.

And if there is no sieve i not equal to j and if there is no such a i then what it is, it is basically epsilon. Then if there is no such a i then R ij 0 sorry R ij 0 is empty and in that case this is the regular expression ok. And if i is equal to j so, if q i is equal to q j if there is no self loaf then it is simply epsilon say epsilon which is corresponding to the r ij 0 is equal to epsilon, because epsilon will give us the regular expression epsilon will give us the corresponding the singleton epsilon. So, this is the yeah this is the base case. So, for base case let us write informal a.

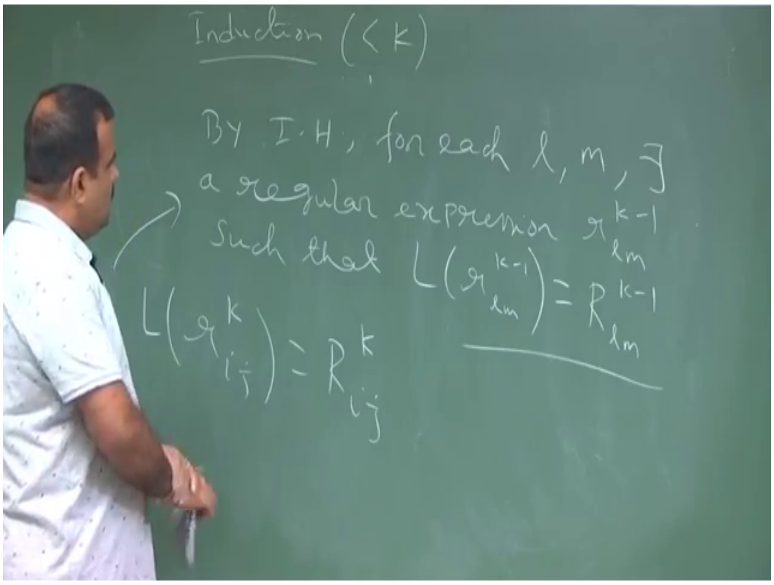
(Refer Slide Time: 15:29).



So, r ij 0 is this a ij this for i not equal to 0 or it is empty if no such a i this is also for i not equal to j or it is epsilon if no such a i and i equal to j, this is true for i equal to j also. So, if i is equal to j and these are all self loop then the r i is basically r 1 plus r p plus epsilon this is our r ij 0 if all this self loop with the epsilon. So, that is claim is true for so, L of r ij 0 is nothing, but R ij capital R ij 0. So, this we can easily verify. So, the result is true for r k is equal to 0 that is the base case ok.

Now, we are going to assume that result is true for some value k for up to k and then if we can show that result is true for k plus 1 then here done. So, that step we are going to.

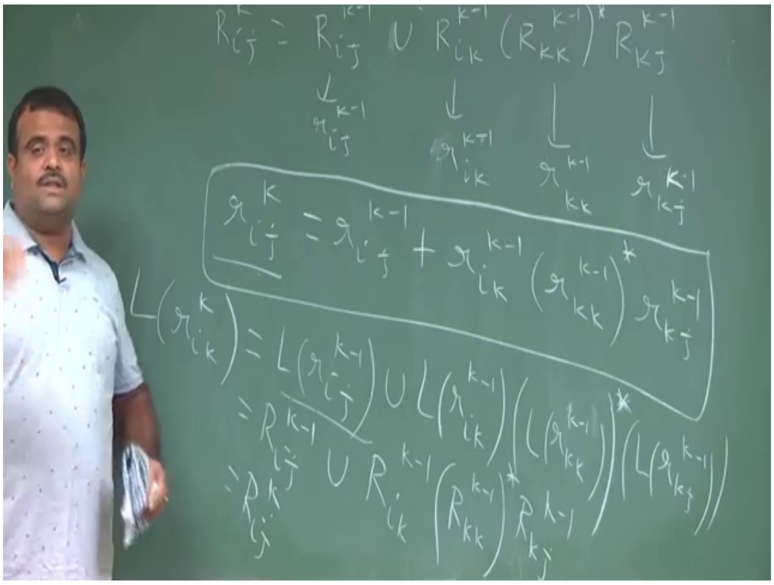
(Refer Slide Time: 17:04).



So, now o, this is called induction step induction step or induction hypothesis. So, we assume that is all is true up to k ok. So, we assume given R ij we are fixing we have fixed in i and j given R ij for all R ij sorry for all i and j ah. This is k this result is true for up to k minus 1 that means, what; that means, this assumption is telling us by the induction hypothesis for each yeah we may not fix ij also if each lm there exist a regular expression r l m k minus 1 such that language of r lm k minus 1 is same as R lm k minus 1 this is really this is our assumption ok.

Now, from here we need to find a r or we can we need to find a r ij k such that the language accepted by this is R ij k with the help of this and then we can we have already proved the base case we can say that this result is true for all of all k ok. So, this is our assumption, now what is that therefore, how to construct small r ij k that will coming from the recursive definition of R capital.

(Refer Slide Time: 19:33)



We know the capital R ij k is nothing, but R ij k minus 1 Union of R ik k minus 1 R kk k minus 1 star R kj k minus 1 so, this recursive relationship we know ok. Now we know that we have a regular expression for this we know we have regular expression for this here we are not fixing ij we for every i and j we have a regular expression when the power is less than k. So, we all have regular expression for this.

So, we have a regular expression for this r ij k minus 1 we have regular expression for this small this is this is from the assumption induction hypothesis, we are assuming that we do have a regular expression for all these which is whose power is less than k, I mean less than equal to k minus 1. Then we have to construct we have to have a regular expression which is true for k also ok. So, we have a regular expression for this, you also have a regular expression for this, is coming from the induction assumption or induction hypothesis we do have a regular expression from this.

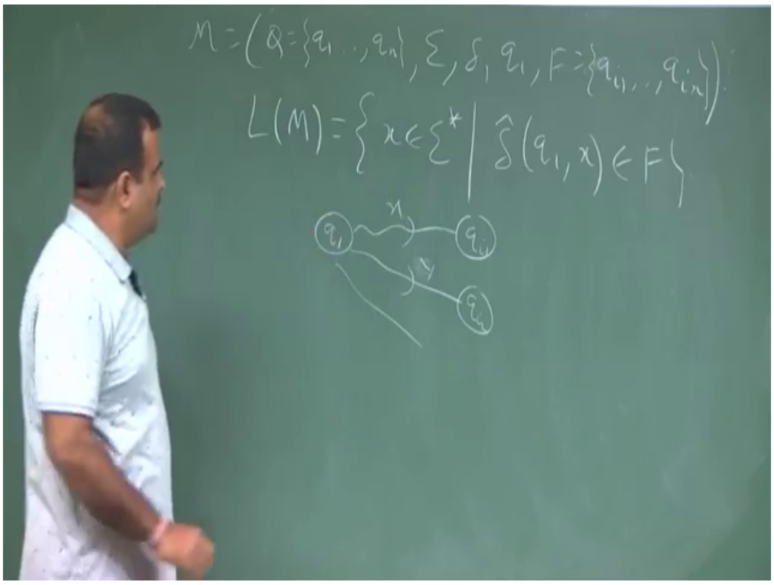
Now, using this regular expression we have to get a regular expression for this and that is nothing, but because we know the properties of regular expression r ij k we just simply defined because this union is nothing, but plus when you talk about the language, r ij k minus 1 plus r ik k minus 1 concatenate r kk k minus 1 star r kj k minus 1 that is all that is all. So, r ij k is equal to r ij k minus 1 plus r ik k minus 1 because we are assuming that we do have a regular expression for this.

Now, what is the language accepted by this. So, language accepted by these is nothing, but by the definition of regular expression and if we consider the precedence of this operator this is nothing, but language accepted by this union of language accepted by this concatenated with the language accepted by this language accepted by this star then concatenated with the language accepted by this.

Now, what is this, this is basically this is the regular expression corresponding to this. So, language of this is nothing, but this that is the assumption R ij k minus 1 union of what is this, similarly this is the regular expression corresponding to this language. So, this will be R yeah this will be R ik k minus 1 similarly this will be R kk k minus 1 star and this is this is again R kj k minus 1 and from this recursive relation this is nothing, but R ij k. So, this language this regular expression is corresponding to the language R ij k this is the proof.

So, our statement is true for all k and we have already our statement is true for k we assume the statement is 2 power k minus 1. Now here we have a regular expression which is accepting the key and we have seen that base case the k is equal to 0 it is true. So, that means, for k is equal to 1, k is equal to 2, k is equal to 3, it will be true. So, we can keep on constructing this by this relationship. So, this way we can ah. So, this is the proof this is the proof by induction ok. So, this will use ah. So, now, suppose we have this. So, we have this now from here how we can say that given a DFA we have a regular expression which is accepting the language corresponding to DFA. So, that we have to see.

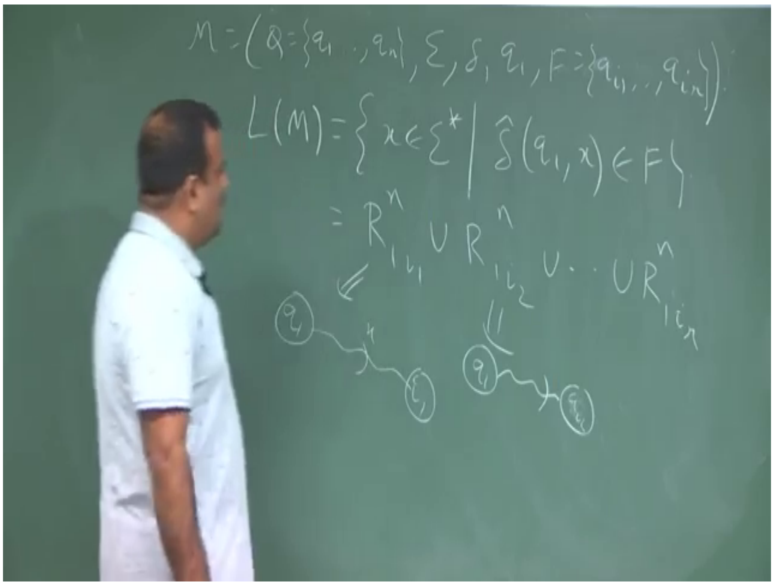
(Refer Slide Time: 25:21)



So, we have given a DFA Q to say q 1, q 2, q n delta q 1 and you have F, F is also say some q i1, q i r ok. Now the language accepted by this DFA is also stream x such that delta hat of q 1, x is belongs to F either one of this. So, we start from q 1 x either it will go to q i1 or you can go to q i 2 all such x y z all such connection of the string this.

Now, we want to get a regular expression for this.

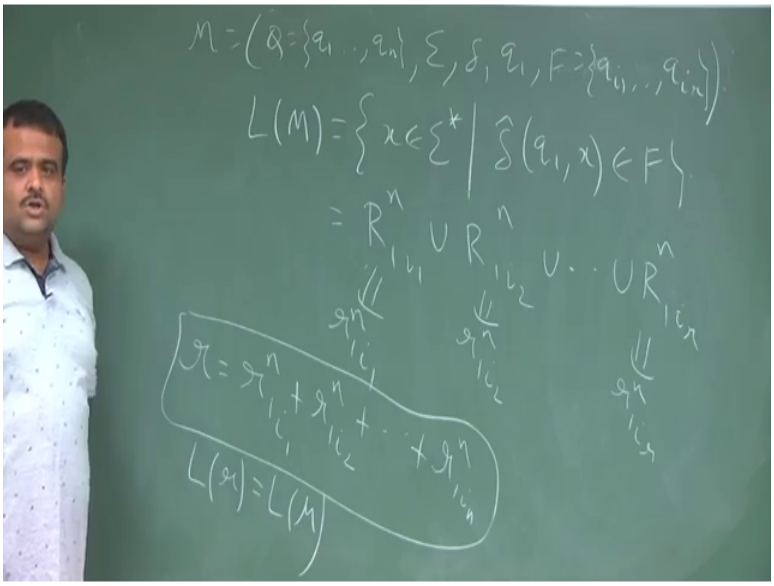
(Refer Slide Time: 26:25)



Now we know this R ij n. So, what is the in terms of these, this can be written as then R 1i1 n union of R 1i2 n union of R 1ir n, because these are all language these are all string they say for example, this what is this set this is the string we start with q 1 with x we reach to i 1 which is one of the final state and there is no restriction on the intermediate nodes because this is n like this.

So, similarly here also we start with q 1 we reached to q i2 q i1 q i2 and there is. So, this is our language in terms of capital R i. Now we know for each of this r i you have a small r i.

(Refer Slide Time: 27:30)



So, that is nothing, but so, this will corresponding to r 1i1 n, r 1i2 n, r 1ir n and these are union. So, basically r is r 1 i1 n plus r 1 i 2 n dot dot dot plus r 1in n. So, this is the regular expression.

So, what is the language, language of this is union of all these things. So, language of this is nothing, but language of all these, these are this basically language of n. So, this is the regular expression for this automata. So, given a DFA we can always construct a regular expression which will corresponding to the same language of that automata. So, in the next class we will give take some example to construct such given a DFA with say few 2 3 states how to construct this. So, we will discuss that we will we will exercise that in the next class.

Thank you.