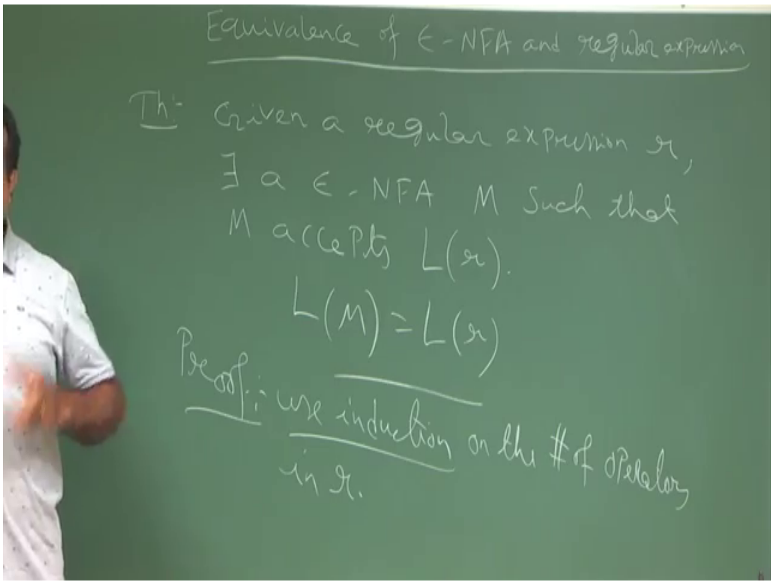
**Introduction to Automata, Languages and Computation**
**Prof. Sourav Mukhopadhyay**
**Department of Mathematics**
**Indian Institute of Technology, Kharagpur**

**Lecture - 19**
**Equivalence of Epsilon -NFA and Regular Expression**

So, we are talking about the Equivalence between the Epsilon NFA and a Regular Expression.

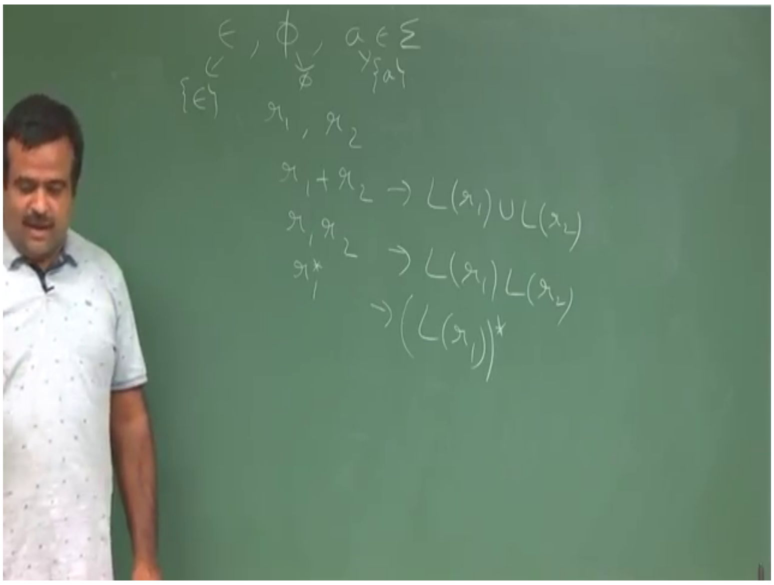
(Refer Slide Time: 00:22)




So, last class you have given the statement of the theorem; given a regular expression r, there exists a NFA epsilon, NFA M such that such that the language of the regular expression M accept such that M accepts L of r. In particular, the language of M is same as language of r. So, this theorem we are going to prove by induction, induction on the number of operator on r ok.

So, to prove this theorem we will use mathematical induction on the number of operators in r; r is a regular expression; so, which consists of some operator like there are only three operator concatenation, plus and star ok. So, we will prove this by mathematical induction on the number of operator on r. So, in that case, what is the base case? Base case means where number of operator is 0; that means, so you just recall the definition of regular expression in the we recursively define the regular expression.

(Refer Slide Time: 02:26)

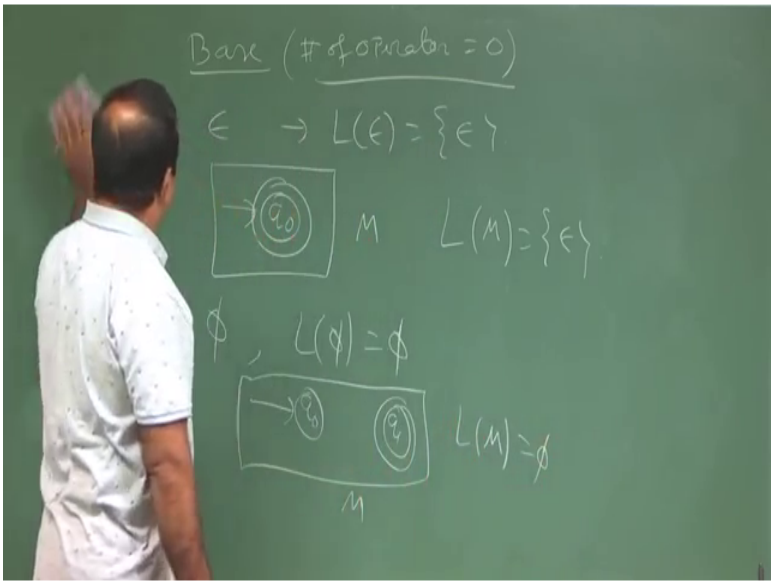



There is a base case we define this epsilon, then empty and a where a is coming from the sigma. These are all regular expression with 0 number of operator, this is the starting and the recursively we defined the regular expressions have suppose r 1, r 2 is a regular expression; r 1, r 2 could be one of this. Then, if r 1 r 2 is a regular expression then r 1 plus r 2 is a regular expression, then r 1 into r 2 is a regular expression, r 1 star is a regular expression.

So, now what are the language corresponding to this? So, this will corresponding to the language, language of r 1 in your language of r 2. This is language of r 1, language of r 2 and this is language of r 1 star, this we all know this is the recursive definition of regular expression and the base case is; what is the language corresponding to this? This is the singleton set epsilon and this is empty set and this is the language corresponding to the singleton set a.

So, this is the definition of regular expression. Now these are operator this is called concatenation, this is plus and this is the star ok. So now, we will prove this on the now by using the mathematical induction on the number of operator.

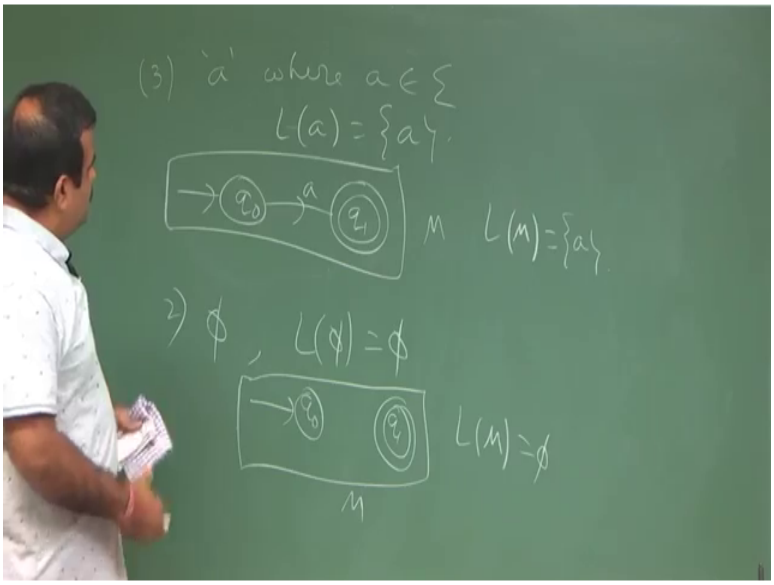
(Refer Slide Time: 04:05)




So, for that it such first prove the base case because, every mathematical induction method we have a base case. We prove the result is true for n is equal to 0 or 1, that is the base case. Now after that, we assume the result is true for n is equal to k up or up to k and then if we can prove the result is true for k plus 1, then we are done. So, that is the way we will do this. So, base case so, base case is number of operator is 0 ok. So, there are three base case; one is epsilon is a regular expression which is denoting the language of this is singleton set epsilon. So, this is, now we want to find a automata for this which is accepting this. So, what is the automata? This is q 0 here q 0. So, this is a this is our M, this is our M such that L of M is nothing but epsilon ok.

So, it is only accepting epsilon, this is the epsilon NFA. Only there is no epsilon over here, but this is the automata and this is a epsilon NFA. Epsilon NFA it is not mandatory to have a epsilon move, we can have a epsilon move, we may not have a epsilon move also. In fact, there is no move from this. So, that is why it is accepting and this is the starting state it is also the final state q 0, that is why it is accepting the epsilon. Number 2 empty. So, what is the language of this? It is just a empty set. Now, what is the automata for this? We can have automata like this q 0 and we can have final state q 1 which is not reachable from q 0 that is all, this is our M which is not reachable from q 0 by any move any means, even not by the epsilon move. So, NFA means empty.

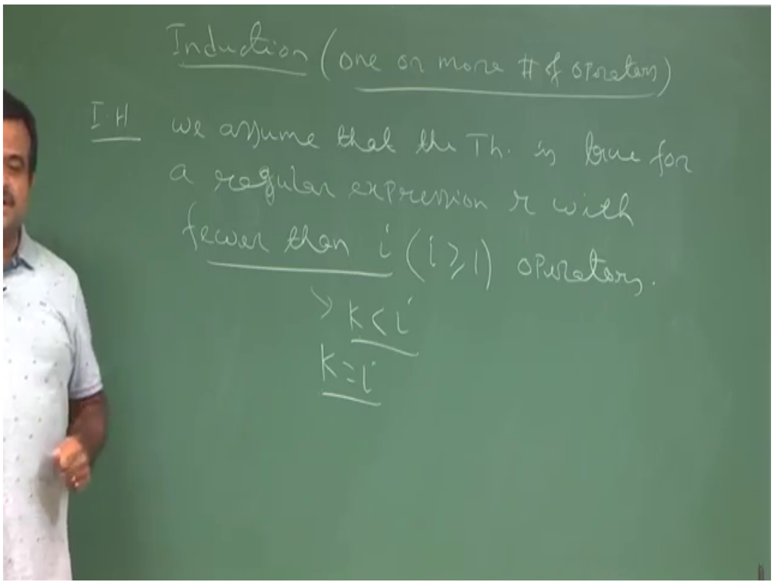
(Refer Slide Time: 06:39)




So, this is now we have another base case number 3 which is the regular expression a where a is coming from, where a is any alphabet coming from that sigma, our set of all possible finite alphabet ok. So, what is the language of a? Language of a is the singleton set a. Now, what is the automata? Simple, you start with q 0; q 1 is the final state we have a, this is a NFA in particular this is the epsilon NFA ok. So, this is our M. So, L of M means a. So, the result is the theorem the statement is true for 0 the base case, that is the number of operator is 0 ok.

Now, the inductive step; so, these are all the base epsilon NFA for the base case.

(Refer Slide Time: 07:45)

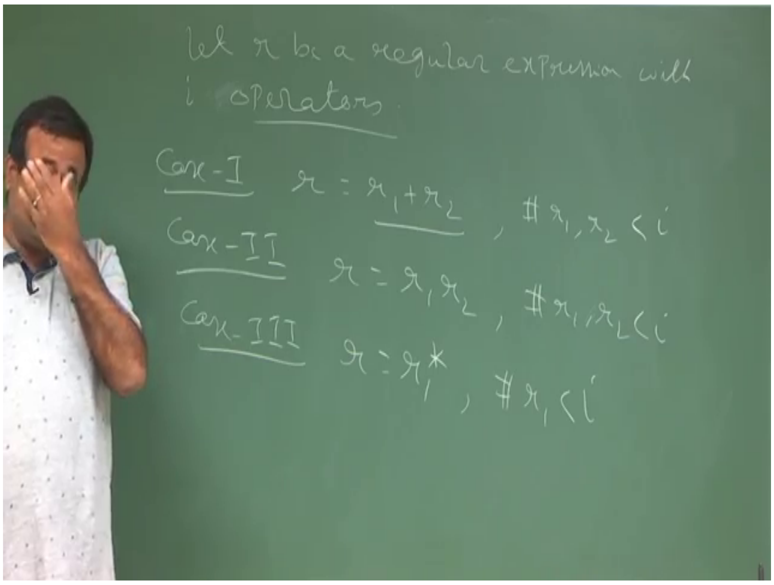



Now, the induction: so one or more operator, more number of operators ok. So, we assume that given the regular expression having one or more number of operator. So, they are not the base case. So, they are so, in that case they are either of this form if r is such a regular expression which is having that ok. So, for that we need to have a assumption or hypothesis, the induction hypothesis or assumption. We assume that, we assume that the theorem is true for a regular expression; for a regular expression r with fewer than i where, i is greater than equal to 1 operator.

Fewer than i; that means, the result is true for up to i, then we have to show this is the assumption this is the induction hypothesis. Then you have to show that if a result is true for i, then we are done. So, result is; that means, it is true for the all k up to i ok. Now, we have to prove using this induction hypothesis now to prove that the result is true for i is equal to k is equal to i. If we can do that, as you have already have the base case result is true for i is equal to 0 in. So, if we can prove this then the result is true for i is equal to 0 plus 1 that is 1, i is equal to 1, i is equal to 2, i is equal to 3. So, result is true for all i, but with the help of, so this is the induction hypothesis this is our assumption ok.

Now, we assume this result is true for all regular expression whose number of operator is less than up to less than i. Now, we take a regular expression with number of operator is i and then we will see whether we can construct a epsilon NFA a for that or not. So, let us try that and this assumption we need.

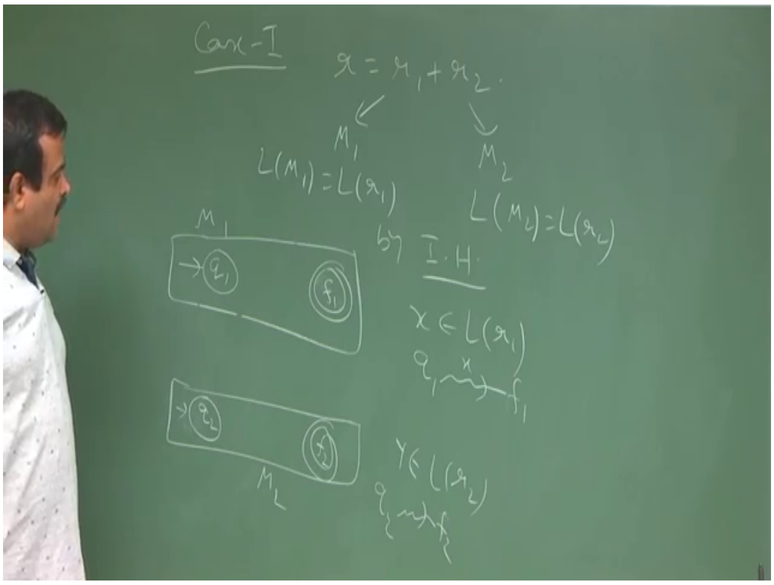
(Refer Slide Time: 10:48)




So, let r be a, r be any regular expression with i number of i number i operator and here i is greater than 1. So, if i is greater than 1, it is not the base case. Then r must consist of the recursive definition. So, r will be of this form there are three cases are here, case I. So, r will be either some r 1 and r 2 where both the r 1 because we have used one operator over here. So, then r 1 and r 2 are both are, so number of operator in r 1 and r 2 are both less than i. And case II, r may be r 1 concatenation r 2 where number of operator in r 1 r 2 because we have used one operator that is called concatenation or it is a unitary operator r is some r 1 star. So, we have used one operator. So, in that case the number of operator in r 1 is less than i, ok.

So, these are all less than i. So, on the if it is number of operator in any regular expression is less than i, then we have a assumption or we have a hypothesis that we have a epsilon NFA for that. So that means, we have a epsilon NFA for r 1 and r 2 for each of these cases. Now, from that we need to construct epsilon NFA for r for each of the cases ok. So, let us try that. So, let us try one by one. First we will take the case I.

(Refer Slide Time: 13:03)




Case I, case I is r is r 1 plus r 2. Now r 1 plus r 2, number of operator in r 1 and r 2 are both less than i. So, by the using the hypothesis the assumption, our assumption; in that case, we have a regular expression n 1 sorry we have a epsilon NFA M 1 and we have epsilon NFA M 2 such that L of M 1 is L of M 2 is this is by induction hypothesis or the induction assumption. We that is the assumption because this if r is having I number of operator we have used one operators. So, r 1 is having will be having less than i operator and r 2 must be having less than i operator. Once it is less than i operator, then we have a regular we have a epsilon NFA for this which is accepting this and we have epsilon NFA for this which is accepting this ok.

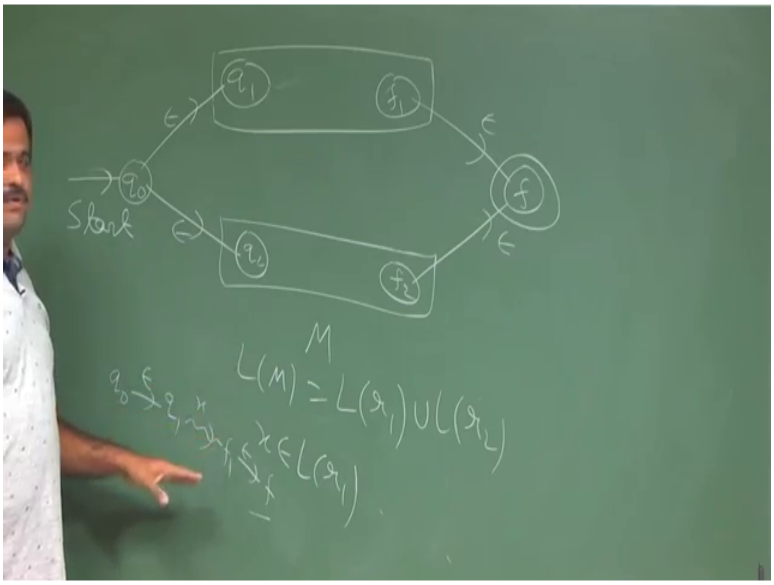
Now, from here how we can have a epsilon NFA for r how we can have a epsilon NFA which is accepting the L of r ok. So, let us let us try that. So, suppose we have M 1. So, suppose M 1 is like this, we are starting with some q 1 q 1 and then say q 2. And then so, this is a not q 2 q there q q f 1 M 1 a q 1 f 1 say that two steps. So, this is the final state. So, this is the starting state like this. So, this is suppose M 1, this is our M 1. So, this is the epsilon NFA for r 1; that means, any string of L of r 1 will be accepted by this. So, if you start with that q 0, q 1 it will be reached to the.

So, if you take any string from L of L 1, then from q 1 if we start x will moves to f 1. So, this is the ok good. So, now, this is the M 1, now corresponding similarly we have M 2, for M 2 we take q 2 and then you have a final state f 2. And here we are assuming the only one final state for each this and there is no move from this final state to any other state that assumption is there in the theorem yeah. So, that assumption is there. So, we are assuming we are going to construct epsilon NFA which is having only one final state and there is no transition from this final state ok.

I mean, this assumption is not required, but to make the proof simply simple, we have to do otherwise we can have because if you have many final state we can have different construction for this. But, anyway we can have because the base case we are not violating this assumption that we have only one final state and there is no transition from the final state to any other state, good. So, now, this is also M 2, now if we have a y which is L of r 2 then, so, if we start from q 2 with this y we can reach to f 2 ok. So, this is the, this is by the induction method. So, we do have a epsilon NFA for r 1, we do have epsilon n f a for r 2.

Now, the question is from here how we can construct a epsilon NFA for r 1 and r 1 plus r 2. So, that is, that is the construction you are going to do now.

(Refer Slide Time: 17:28)



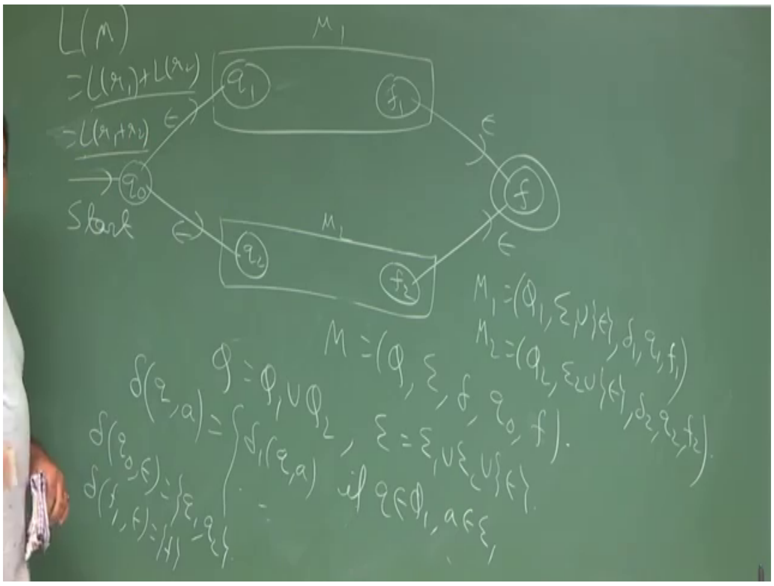

So, we are now we are constructing a new epsilon NFA with the help of by taking. So, we have this M 1, this is 2 sorry this is not a single, there are many states over here f one this is our M 1 and we have M 2 also q 2 f 2. Now, we have a q 0 this is we are constructing the new M which is which could accept the r which is r 1 plus r 2, then this is f meaning f. So, we have a epsilon move from here to here, then we have then we have we have a the move for M 1, then from here to here we have a epsilon move, from here to here we have a epsilon move and this is our final state and this is our starting state ok.

So, this is our M. So, this is the epsilon NFA which is accepting the language. So, this is our m. So, L of M is nothing but L of r 1 union of L of r 2 which is accepting the language L of r 1 union L of r 2. How to show this, because if you take any x from L of r 1, then what we do? We first start with this, we take the epsilon move we go here then x is a accepted string then with the x will go to the this thing f 1, then from f 1 with the take of help of epsilon move we go there.

So, like we start from q 0, then by epsilon move will go to q 1 and then with this x we go to f 1 and then again by epsilon move we go to final f. This is, this f is our final state; there is only one final state that is the part of the theorem done. Now, similarly if you take a y from here, we take this path we go there ok. So, this L of M is basically, so any element here and similarly we can this is equality.

So, if we have if there is a, the if there is a string which is accepting by this new NFA, then it has to go either this path or this path. Now, if it is going to this path, then it has to reach to x by f 1, final state of this. So that means, that string is accepted by this M 1 or it has to reach to this that string is accepted by this. So, this is a subset of this, that is a subset of this. So, from here we can say this is equal ok.

(Refer Slide Time: 20:36)

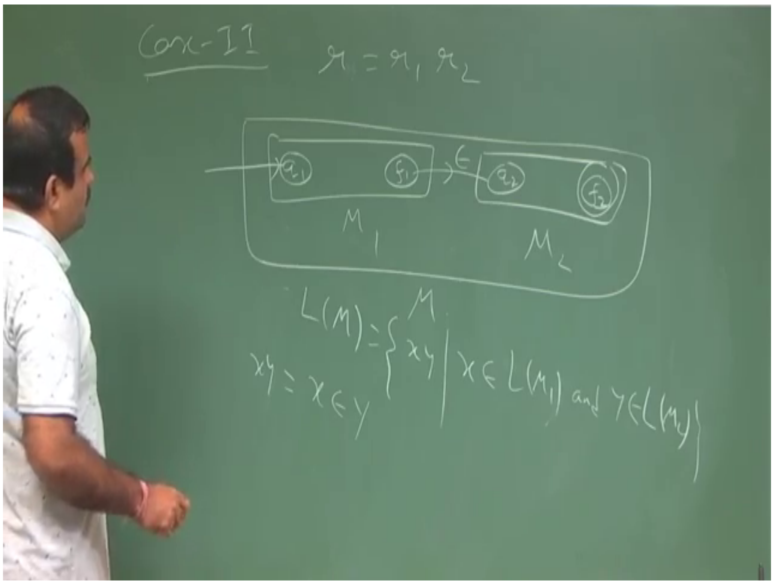


So, we can formally define this M with the help of tupple like if say M 1 is q 1 and q this is small q 1 sorry, delta 1, delta 1 is the transition small q 1 f 1. So, if this is M 1 and q 2 sigma 2, sigma 2 and sigma 1 this could be say or this could be different we are taking a general 1. So, we can make it different then what is our M? M is nothing but q sigma delta q 0 and f. So, what is sigma? Sigma, what is q; q is nothing but q 1 union q 2 and sigma is nothing but, sigma 1 union sigma 2 along with the epsilon and q 0 and f we have introduced the new state for this.

And what is delta? The transition delta of delta of q comma a is equal to delta L of q comma a if q is in q 1 and a is in sigma 1; that means, if we take this path will follow the transition rule of M 1 if we take this path will follow the transition rule of M 2. Similarly, this and we have 2 epsilon move over here. So, delta of q 0 epsilon is nothing but q 1 and q 2 and there also we have epsilon move, delta of f 1 epsilon which is f similarly f 2 epsilon which is also f. So, anyway formally we can define this, this formal thing will be available in the lecture note this is case I.

So, then this is the M which is accepting the, so this is the M which is accepting the, so that means, it is corresponding to. So, this is nothing but language of ok. So, this is the proof. Now, we take the other case II ok.

(Refer Slide Time: 23:33)



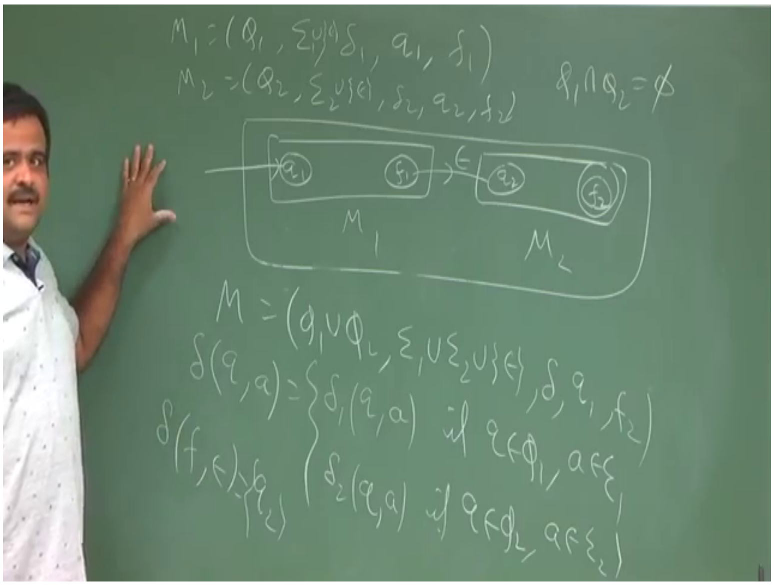

Case II, case II is the concatenation; r is nothing but r 1 r 2 concatenate and we assume there is a because of induction, we assume there is a epsilon NFA for r there is a epsilon

NFA for r 2. Now, we need to construct a epsilon NFA for r 1, r 2. Suppose, this is the epsilon NFA for r 1, q 1, f 1 and this is the epsilon NFA r 2, q 2, f 2. So, this is our M 1, this is our M 2 such that these two are accepting this two regular expression regular language. I mean the language corresponding to these two regular expression. Now how to construct a new NFA epsilon NFA, we do this we take the starting state of M 1 as the starting state of M, then we have a epsilon move from here to here and this is our final state, this is our final state ok.

So, this whole thing is our M. So, then what is the language corresponding to M because if this is nothing but x, y such that x is I mean x is L of M 1 and y is L of M 2. It is it can easily verify because if you take a x, y the string we take start with this then with the x with reach to the f 1, then from there we take a epsilon move we go to q 2, then with the y because this is nothing but x epsilon y, x y can be written as x epsilon y epsilon is a null string ok. Then, with the help of this will go to the final state q 2. So, f 2, f 2 is the final state of this ok.

So, now here also we can have this formally, we can define this M with the help of M 1 and M 2.

(Refer Slide Time: 26:02)




So, if M 1 is q 1 sigma delta 1 q 1 and f 1, if M 2 is q 2 sigma 1 sigma 2 say I mean you could take the same or different because, if it is same also does not matter we can take the union of that and then, but any effort generalized case we are taking these two are different. And q 1 and q 2, we are taking they are disjoint. In the earlier case one also because, if they are if there is some state we can easily rename. So, we can always assume that this q 1 and q 2 are disjoint because by renaming of the states; this one delta 2, then q 2 f 2, then what is our M.

So, M is nothing but q 1 union q 2 sigma 1 union sigma 2 union epsilon delta then q 1 and then f 2 and what is delta? Delta is nothing but, so delta of q a is nothing but delta 1 of q a if q is in q 1 and a is in sigma 1 including epsilon. And otherwise it is q 2 q a if q is in q 2 and a is in sigma 2 and we have a epsilon move over here delta of f comma epsilon is q 2 because there is no other move from these states. So, that is one of the assumption, that is one of the, one of the part in the theorem ok. So, this is done. So, this is the third second case. Now, in the next class we will prove the third case and then you take some example.

Thank you.