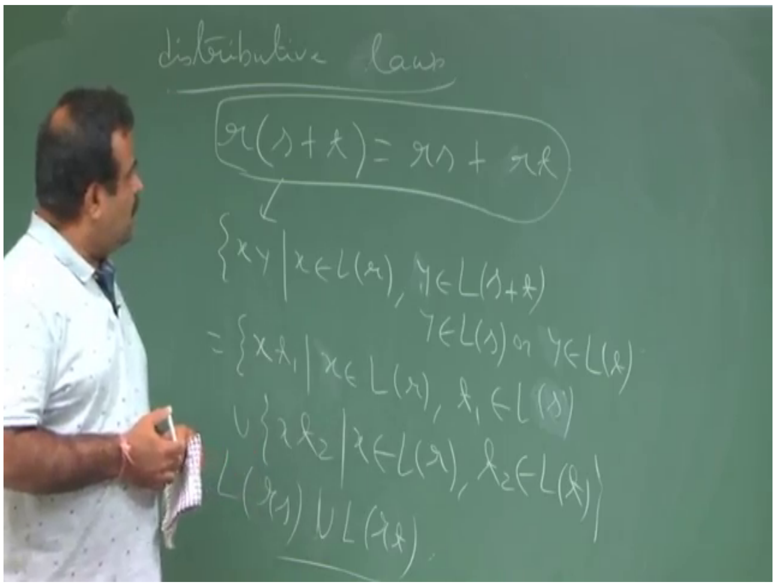
Introduction to Automata, Languages and Computation
Prof. Sourav Mukhopadhyay
Department of Mathematics
Indian Institute of Technology, Kharagpur

Lecture -18
More on Regular Expression

(Refer Slide Time: 00:15)



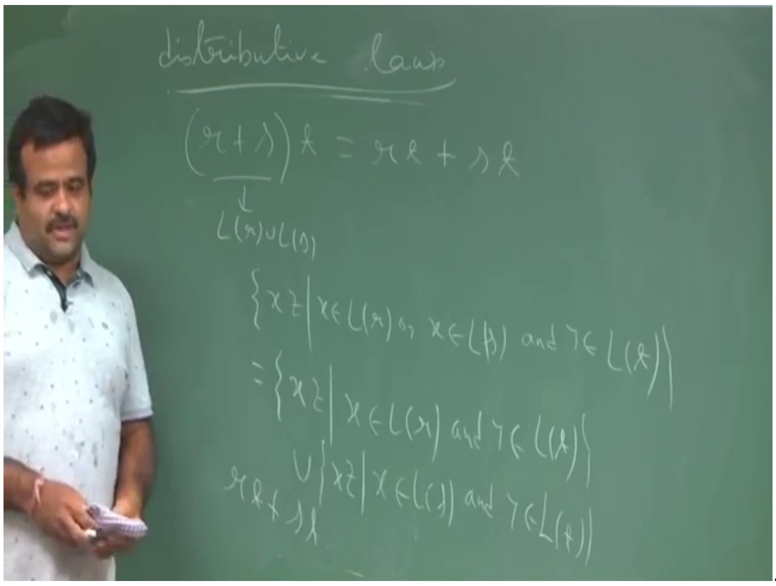
distributive laws

$$r(s+t) = rs + rt$$

$$\{xy \mid x \in L(r),\ y \in L(s+t)$$
$$y \in L(s) \text{ or } y \in L(t)$$
$$= \{xt_1 \mid x \in L(r),\ t_1 \in L(s)\}$$
$$\cup \{xt_2 \mid x \in L(r),\ t_2 \in L(t)\}$$
$$L(rs) \cup L(rt)$$

So, we are talking about Distributive Law of the Operator, the operator use in a regular expression. So, this is the, this is we discussed in the last class, this the left distributive law.

(Refer Slide Time: 00:39)



distributive laws

$$(r+s)t = rt + st$$
$$L(r) \cup L(s)$$
$$\{xz \mid x \in L(r) \text{ or } x \in L(s) \text{ and } y \in L(t)\}$$
$$= \{xz \mid x \in L(r) \text{ and } y \in L(t)\}$$
$$\cup \{xz \mid x \in L(s) \text{ and } y \in L(t)\}$$
$$rt + st$$

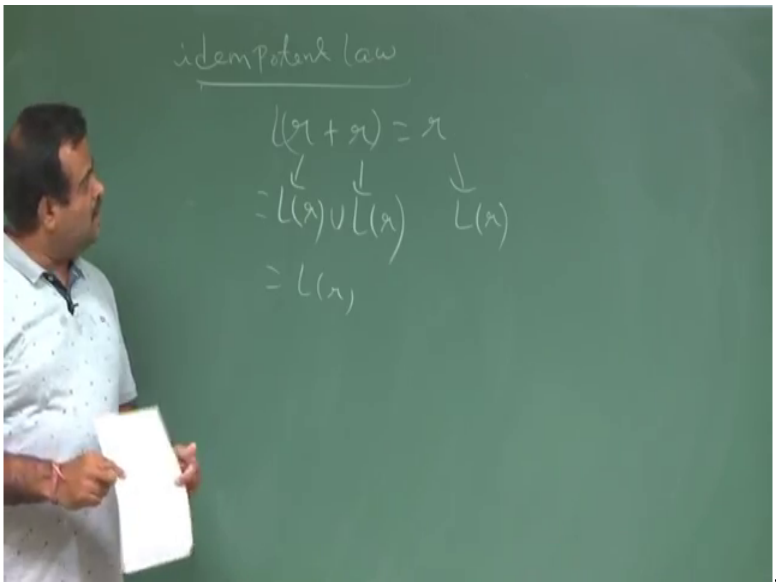
.

So, similarly we have the Right distributive law, like if we have two regular expression r and s this is same as, ok. So, this also we can easily verify this the right distributive law because this is this is nothing if the language corresponding to this is L of s. Now, if you take the concatenation with t, then it will be like x z where x is coming from either L of r or L of s and z is coming from L of p.

Now, this as I as we discussed in the last class. This we can break it up to this. This is x z such that x is coming from L of r and y is coming from L of t, this union of x z again x is coming from L of r and y is coming from, sorry x is coming from L of s and y is coming from L of t. So, this is the same language corresponding to the regular expression r t plus s t, ok. They are same. I mean once the regular expression same means if their language is same, then we say that regular expressions are same.

So, this type of result we need for this type of property we need to simplify the regular expression, ok. So, just to check another few properties, then we will move to the equivalency of the regular expression and the epsilon NFA. Another property is Idempotent property on the plus.

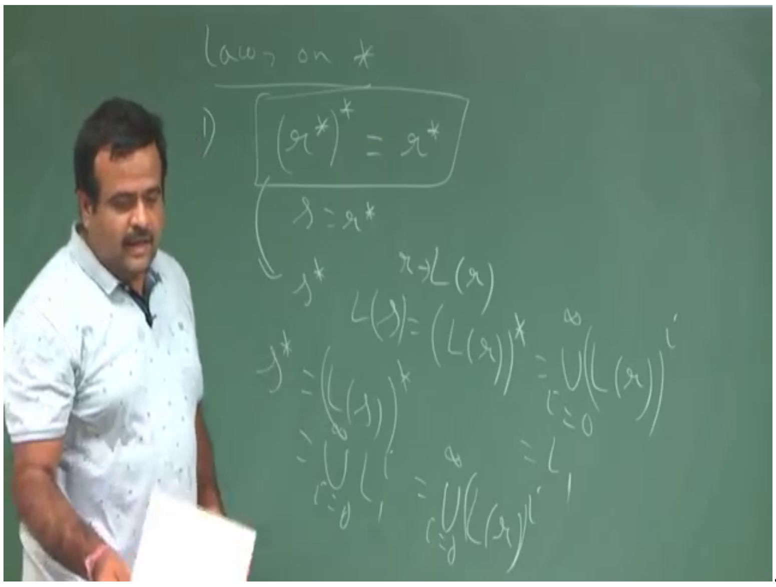
(Refer Slide Time: 02:49)



.

So, it is telling r plus r equal to r. This is also obvious because the r is language. This is corresponding language, this is corresponding to language this. Now, what is the language corresponding to this? This is basically union which is same as L of r. It is quite obvious. So, this is mean corresponding to the language L of r, r plus r is r. Then we see with the some properties with star.

(Refer Slide Time: 03:51)

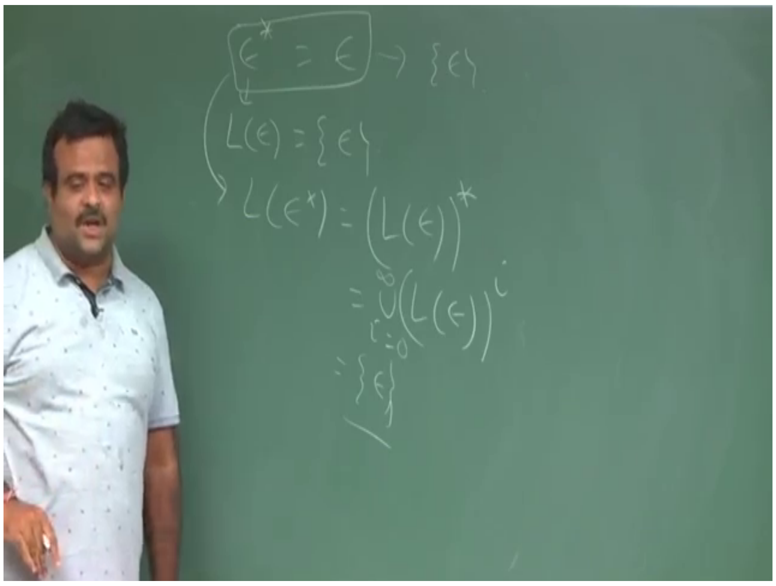

laws on *
1) $\boxed{(r^*)^* = r^*}$
$s = r^*$
s^*
$r \to L(r)$
$L(s) = (L(r))^* = \bigcup_{i=0}^{\infty} (L(r))^i$
$s^* = (L(s))^* = \bigcup_{i=0}^{\infty} L_1^i = \bigcup_{i=0}^{\infty} (L(r))^i = L_1$
.

So, first property is r is a regular expression, r star star is equal to r star, ok. Now, what is r star star? If we denote s is equal to r star, then this is nothing, but s star. So, what is r star I mean what is s star? So, r is so this is the language corresponding to r, r is corresponding to this language, then what language s is corresponding s is corresponding to. So, this is the language of s and this is nothing, but union of i is equal to 0 to infinity. Now, if you take if this is say this is say some language L1.

Now, what is s star? Then s star is basically L1 star which is nothing, but again union of this L1 to the power i, i is equal to 0 to infinity, but L1. So, this is basically double union. So, which is basically same as say L1 is that union. So, which is basically same as union of L r to the power i, i is equal to 0 to infinity which is same as this.

So, this is also quite obvious. Now, what is epsilon star? What is epsilon star?

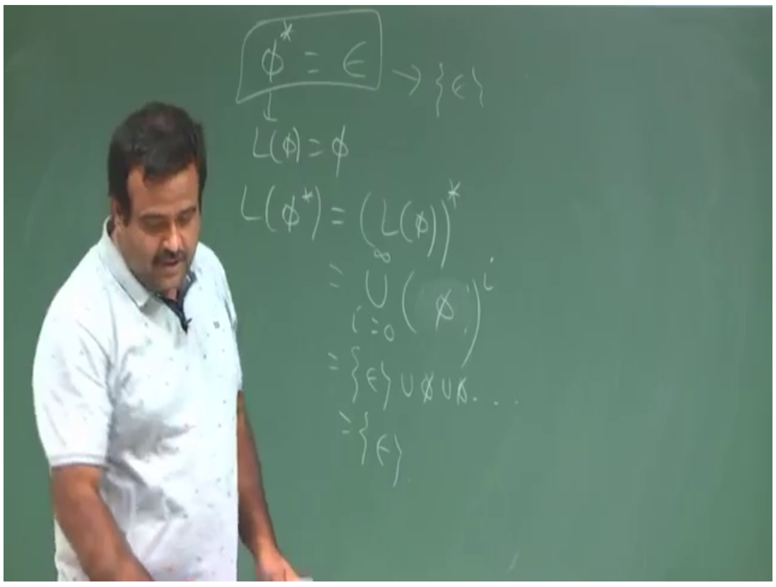
(Refer Slide Time: 06:03)


$\boxed{\epsilon^* = \epsilon} \to \{\epsilon\}$
$L(\epsilon) = \{\epsilon\}$
$L(\epsilon^*) = (L(\epsilon))^*$
$= \bigcup_{i=0}^{\infty} (L(\epsilon))^i$
$= \{\epsilon\}$

This we want to see is this same as epsilon. We need to check this. So, what is epsilon? Epsilon is language corresponding to epsilon is just a singleton set epsilon I mean that the null string this contain only the null string I mean no string is there, but this is not empty, this is containing null string. So, this is not same as phi. This is a singleton set containing the null string, ok.

Now, what is epsilon to the power star that the language corresponding to this is then language of epsilon to the power star. Now, this is nothing but union of L of epsilon to the power i, i is equal to 1 0 to infinity. Now, i is equal to 0. So, this is basically epsilon, then we take epsilon with the epsilon that is also epsilon, then epsilon epsilon that is also. So, this is nothing, but the epsilon only. So, this is nothing, but epsilon. So, this is also corresponding to epsilon. So, this is same ok, but what about phi to the power star?

(Refer Slide Time: 07:41)



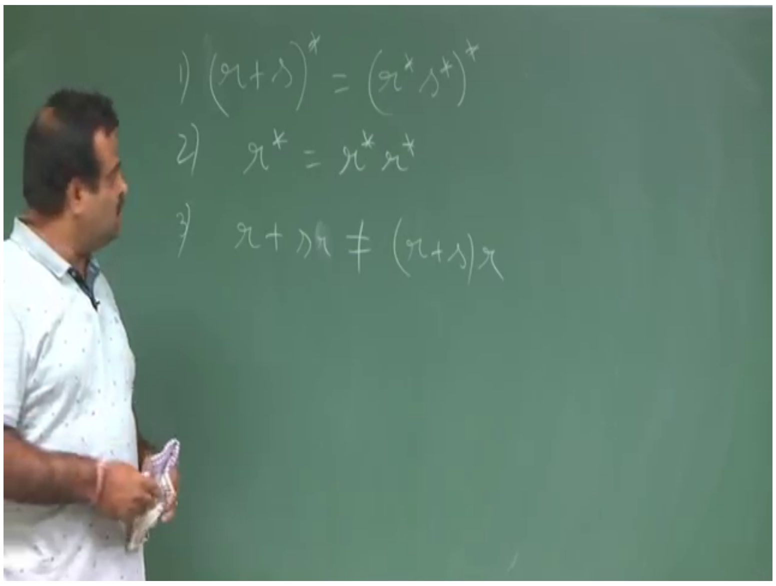
$\boxed{\phi^* = \epsilon} \rightarrow \{\epsilon\}$
$L(\phi) = \phi$
$L(\phi^*) = (L(\phi))^*$
$= \bigcup_{i=0}^{\infty} (\phi)^i$
$= \{\epsilon\} \cup \phi \cup \phi \ldots$
$= \{\epsilon\}$

So, that we have to, so what about phi to the power star is this phi. No this is epsilon. Why? Because phi is what phi is basically language corresponding to empty set nobody is there.

So, now what is the language corresponding to phi to the phi to the power star? This is the language corresponding to phi to the power star. Now, once we take the star it is union from language corresponding to the phi. So, this is phi. So, phi to the power i, i is equal to 0 to infinity. Once we have 0, then epsilon will be there and remaining are all phi because nobody is there in that set. So, this is basically epsilon and this will also corresponding to epsilon.

So, these two, these are the properties we will use for simplification the regular language.

(Refer Slide Time: 08:57)



1) $(r+s)^* = (r^* s^*)^*$

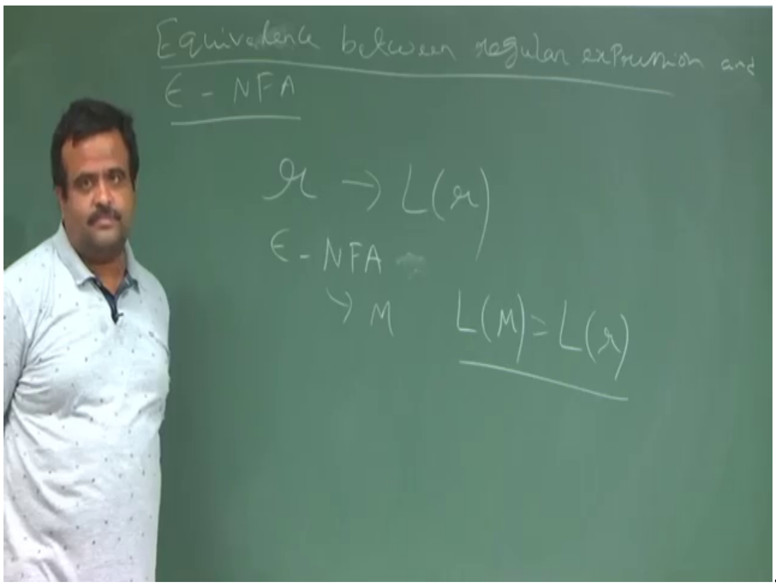
2) $r^* = r^* r^*$

3) $r + sr \neq (r+s)r$

There are other few more properties are there like say r plus s to the power star is same as r star s star to the power star, ok. So, this we are not proving now, then r star can be written as r star r star, ok. So, these are the properties. Another thing is r plus s t is not same as. So, sorry r plus s r is not same as r plus s r, ok.

So, these are the properties we are not going to prove it, but these are the properties we will we may use for simplification, ok.

(Refer Slide Time: 10:19)



Equivalence between regular expression and ϵ - NFA

$r \to L(r)$

ϵ - NFA

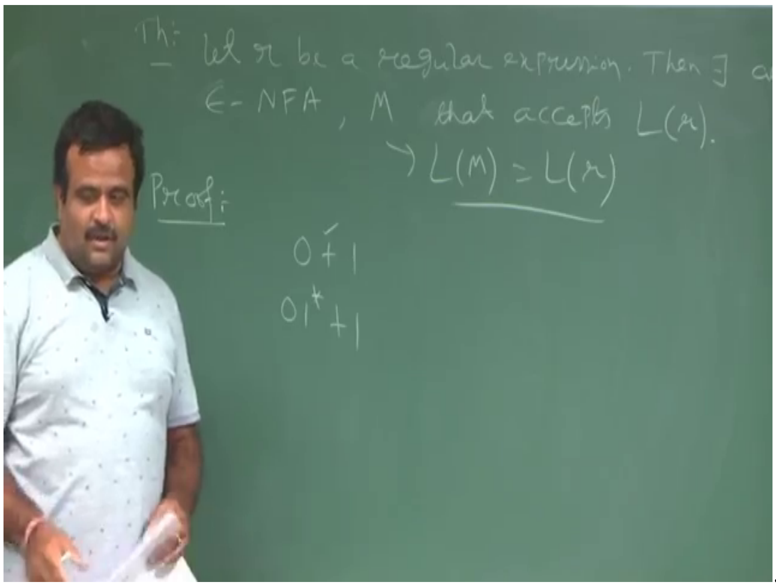
$\to M$ $\quad$ $\underline{L(M) = L(r)}$

.

So, now we move to the equivalency of equivalence between regular expression and epsilon NFA. So, what is the meaning of this? Meaning of this is suppose we have given a this will prove by a using a theorem suppose you have given a regular expression r. So, it will corresponding to a language L of r depending on the regular expression it will corresponding to a language that language is denoted by L of r.

Now, we can show that there exists NFA epsilon NFA which is more flexible automata. So far we know because DFA have some restriction only NFA, but epsilon NFA has we have epsilon moved there. So, it is easy to construct the epsilon NFA, then the NFA than the DFA, ok. So, there is a epsilon NFA m such that language of that epsilon NFA is same as language of L. So, this we have to see we have to prove by a theorem. So, let us write that theorem. So, this is a statement of this theorem.

(Refer Slide Time: 12:15)


Th: let r be a regular expression. Then $\exists$ a ϵ-NFA, M that accepts $L(r)$.

$\rightarrow L(M) = L(r)$

Proof:

$0+1$

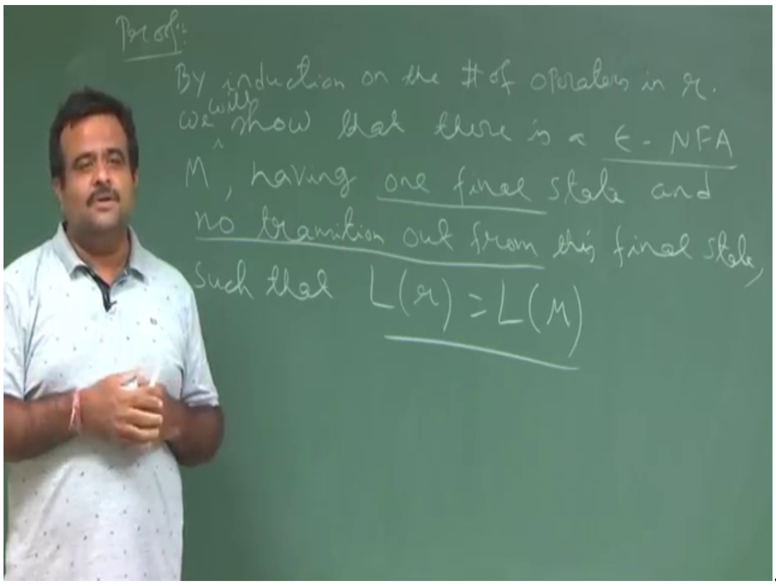
01^*+1

Let r be a regular expression, then there exists a an there exists an epsilon NFA M such that that accepts L of r. That means, the language of this NFA is same as language of the regular expression.

So, these are we have to prove this, ok. So, to prove this we use the method of induction. So, how to use a method of induction. So, for method of induction we should have something on the natural number. So, here natural number is the number of operator in the regular expression how many operators are there like if we have say 0 plus 1, there is only one operator. So, if you have 0 1 star plus 1, there are two operator, a three operator; one is concatenation, one is this star and one is plus.

So, these are the number of operator on the regular expression. So, we will prove this by the induction on the number of operator in the regular expression. So, let us just ok.

(Refer Slide Time: 14:03)



Proof:
By induction on the # of operators in r.
We will show that there is a ϵ - NFA
M, having one final state and
no transition out from this final state,
Such that $L(r) = L(M)$

.

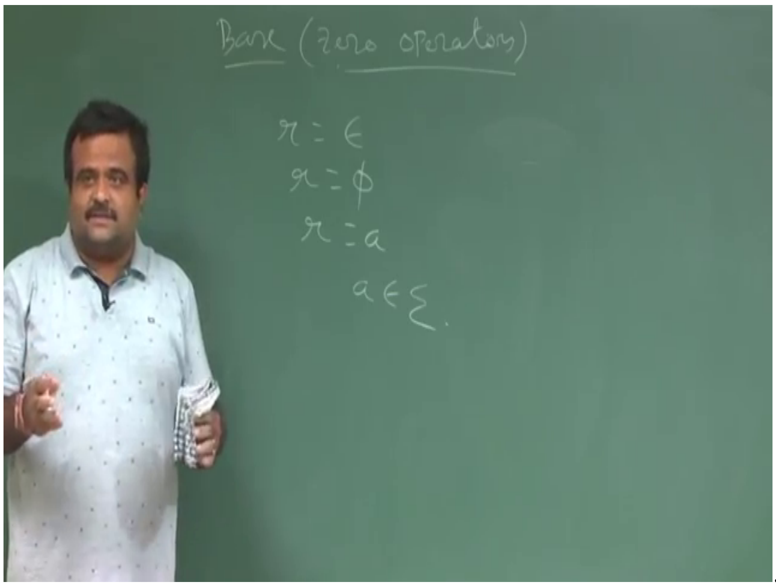
So, prove is by induction on the number of operator operator in r, then we show that there exist there is an epsilon NFA M. Next we are restricting this NFA to some extent like this M has only one final state, and there is no move from the final state to any other states. So, that restriction is there.

So, no transition from the final state M having it can prove in general also, but we are using this proof having one final state, final state and no transition from that no transition out from the final state. So, that means there is no move from the final state to any other state transitions out from the from the final state from this final state. So, this we have to understand. Then, we will show that such that L of r is same as L of M, ok. So, this we have to understand the logic. Logic is we want to show this by induction. For induction we need to have something on the natural number.

So, here what is natural number? The number of operator. We know the regular expression is made by these three operators. So, we just check the number of operators how many times they are coming. So, that is our that is our natural number n for the induction and we will show that will show will show that we have a regular x will have NFA epsilon NFA M which is having only one final state and which is having no transition from the final state to any other state that is the meaning of this no transition out from the final state and only one final state such that the language accepted by that NFA is same as the language of this regular expression. So, that will prove.

So, this since this proof is by induction, so for induction we need to assume this is true for some i up to i or i minus 1, then you have to prove this is true for i and also you have a base step base case. Base case means it should true for i is equal to i is equal to 0 or i is equal to 1 we start with. So, let us just first try to see the base case the zero operator and we will see for the base case we have the epsilon NFA, then we will go for the inductive step, ok.

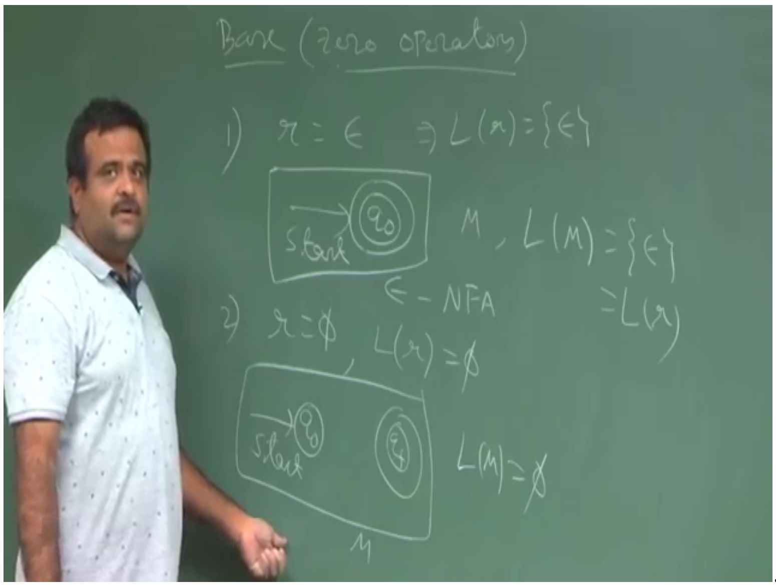
(Refer Slide Time: 17:57)



So, this is the base step base case that is 0 operator zero operations. So, zero operation means we have all the three options. We know r is equal to epsilon is a regular expression and there is no other I mean. So, if we have like this 0 plus 1. This is one operator, but we do not want any operator. So, there are three possibilities r is equal to either 0 or r is equal to empty or r is equal to some alphabet where a is coming from sigma. Sigma is the set of alphabet.

So, for these three cases we need to have a epsilon NFA that is the base case we know that there is epsilon. So, let us try that. So, for epsilon is r is equal to epsilon what is the language L of r.

(Refer Slide Time: 18:55)

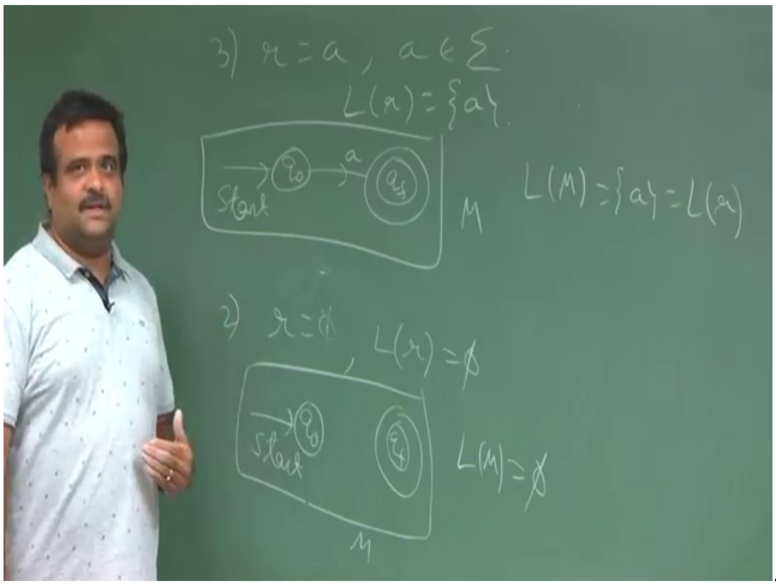

Base (zero operators)
1) $r = \epsilon \Rightarrow L(r) = \{\epsilon\}$
start
q_0
$M,\ L(M) = \{\epsilon\} = L(r)$
ϵ - NFA
2) $r = \phi,\ L(r) = \phi$
start
q_0
q_f
$L(M) = \phi$
M

.

So, we need to construct a epsilon NFA which is accepting that just the singleton set epsilon. So, that is all easy to construct we start with q 0. We make it final step and this is r epsilon NFA, this is r M, this is r M. So, what is the language accepting by this? All the epsilon it is accepting because there is no other move and this is also having only one final state. We have that restriction also and there is no move from the final state to any other state there is only.

So, then L of M is just epsilon which is same as L of r. So, we have the we have for base case. For this case, we have the epsilon NFA for this now the second case for empty set if r is empty, then L of r is the empty set. So, then we need to have a epsilon NFA for this. So, we can have just like this q 0 and then, we can have a final step q f and there is no move. That is all. So, this is the this is our M. This is our M. So, what is the language accepting by this empty not even epsilon because there is no epsilon move over here in this epsilon NFS. Epsilon NFA itself is NFA. So, it is if we do not have any epsilon move also. So, this is the empty set. Now, the third case is the a. So, this is.

(Refer Slide Time: 20:57)



.

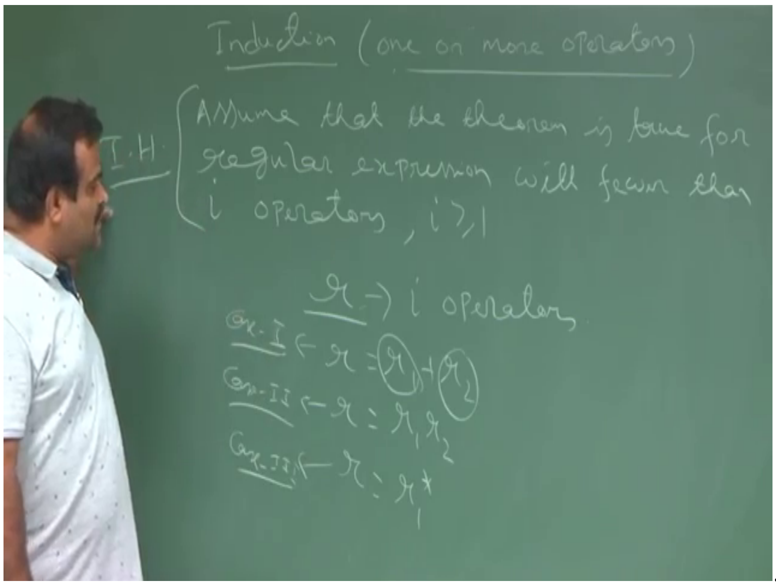
Now, if r is a, where a is an alpha bit any alpha bit if sigma is 0 1, then a is 0 or 1 any one of these, ok.

Then, what is now how to show what is L of r. L of r is basically the singleton set a. Now, what is the language? Now, we need to construct epsilon, we need to get epsilon NFA which is accepting this. So, this is also easy to construct. We have q f, then this is the starting state and this is the final state and we have a, Ok. This is our M, this is our NFA epsilon NFA although there is no epsilon a. So, what is the language accepted by this L of M is basically a which is same as L of r, ok.

So, all the three base cases where there is no operator, only those symbols we have the epsilon NFA and also, we have that property that there is only one final state and there is no transition from the final state to there is no transition going out from the final state. That is also we need to take care of, ok.

Now, we will go for the inductive step. So, this is the base case is true. Yes we do have epsilon NFA for the zero operator. Now, we will move to the in inductive induction one or more operation, ok.

(Refer Slide Time: 22:53)


Induction (one or more operators)

I.H. { Assume that the theorem is true for regular expression with fewer than i operators, $i \geq 1$

$r \to i$ operator

Case-I $\leftarrow r = r_1 + r_2$

Case-II $\leftarrow r = r_1 r_2$

Case-III $\leftarrow r = r_1^*$

.

So, for that we need to assume that is called Induction Hypothesis. We need to assume that, we assume that the theorem, that means we have a corresponding epsilon NFA that the theorem is true for regular expression with fewer than i operator. That means, we are assuming that this is true up to i minus 1, ok. I mean fewer than i operator up to i minus 1, our result is true. That means, if you have any regular expression where the number of operator is less than i minus 1, then we have a epsilon NFA corresponding to that. That is the assumption i operator for i is greater than 1. This is the assumption and this is called Induction Hypothesis. In every Inductive method we have such statement.

We assume that our result is true up to i minus 1, then we will prove that our result is true for i. That means, we will prove that if you have a regular expression r with the i operator suppose our r is a regular expression with i operator, then we will show that we have a epsilon NFA for this r with the help of this Induction Hypothesis and then, after that once we can show that, then after that we have already seen this is the base case. We have a regular expression for 0 for i is equal to 0, then by the Induction method we can say that we have regular expression for every i. So, that is the idea of the proof ok.

So, the idea is for i, i operator means, so we can i is more than 1. So, either there will be 3 cases; either r is r 1 plus r 2 because there are only 3 operators or r is some r 1 concatenate r 2 or r is r star or r is some r 1 star. There are 3 possibilities are there. So, in each possibilities, if we see because this is i operator, this is one operator. So, each of this here as fewer operator I mean the number of operate in this r 1 must be less than i and number of operator in r 2 must be less than i number of operator in here. Also r 1 r 2 must be less than i because here everywhere we are using one operator. This plus, this is concatenation and this is star.

So, everywhere we are using one operator. So, when we talk about individual, those things must have one lesser I mean at most i minus 1 operator and for that we have an assumption for that we have a hypothesis that for those we have for r 1 r 2, we have the epsilon NFA, then we will show by these 3 cases. We refer this as case 1 this is case 2. Case 1 case 2 and this is as case 3, we refer this as case 1 case 2 case 3 and for all this, we will prove that we have the epsilon NFA for r. With the help of this assumption if we have the epsilon NFA for r 1 and if you have the epsilon NFA for r 2, then we can construct epsilon NFA for r.

Similarly, in case 2, if we assume when that assumption is valid because of this Hypothesis if we have epsilon NFA for r 1 and if you have epsilon NFA for r 2 because r 1 r 2 has less than i operator. So, we have the assumption. So, we must have epsilon NFA for r 1, we have the epsilon NFA for r 2, then we can show if we can show we have epsilon NFA for r, then we are done. Similarly here also you have epsilon NFA for r 1 which is fewer operator, then i if we can show that we have epsilon NFA for r 1 star, then we are done. So, that we will discuss that will show in the next class.

Thank you.