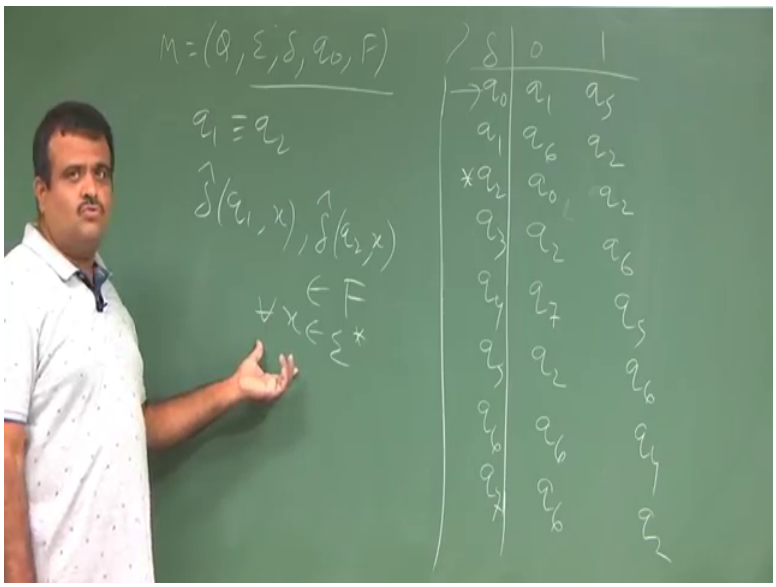
Introduction to Automata, Languages and Computation
Prof. Sourav Mukhopadhyay
Department of Mathematics
Indian Institute of Technology, Kharagpur

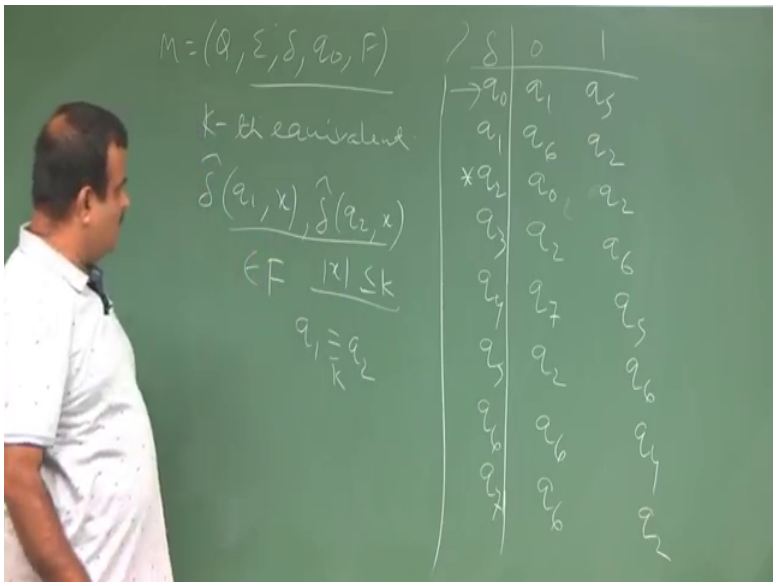
Lecture – 32
Minimization of FA (Contd)

(Refer Slide Time: 00:15)



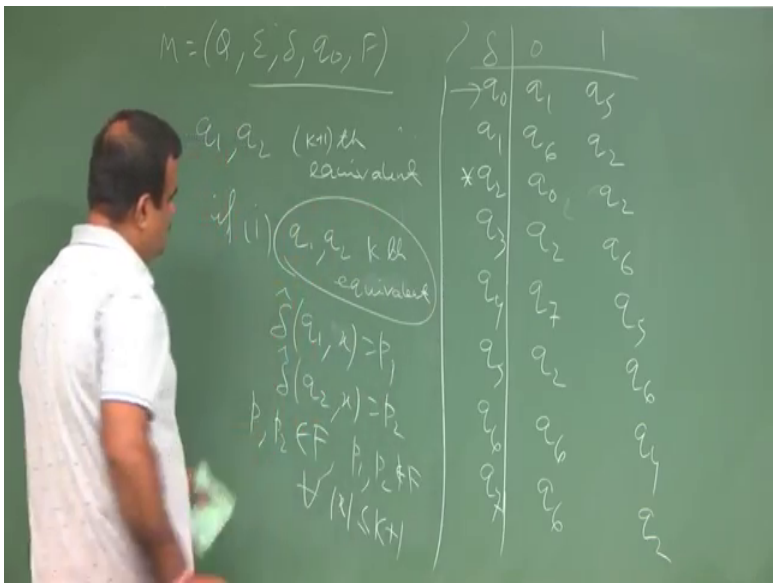
So we are talking about the Minimization of finite automata specially the DFA. So, in the last class, we have seen how we can define the equivalency between two state, so, you given a DFA. So, this is the example you are talking in the last class we will elaborate more here. So, q 0 f this is a given DFA. Now, now we are defining the equivalence between two class a q 1 and q 2 are equivalent, if the if it is reading a, if it is reading a string of length of x and q 2 if they are going to either accepted state or rejected state both then we say they are equivalent. And if this is true for all x, then we have seen that you it is not possible to examine for all x because this is infinite set, so instead of that we define the kth equivalent between two state.

(Refer Slide Time: 01:19)



So, for that what we did? We did like this. So, if delta hat of q 1 x and delta hat of q 2 x, now here x is of length k. Then if they are going to if both of these are going to same accepted state or rejected state same, then we say they are kth equivalent, then we say q 1 is equivalent to q 2, but kth equivalent here, this is true for all x of length k. Then from here we define the k plus 1the equivalent recursively.

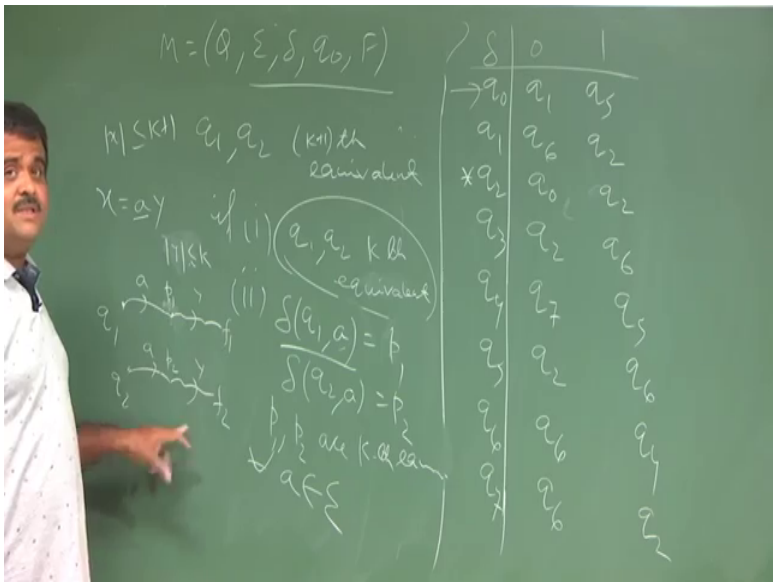
(Refer Slide Time: 02:07)



And we observe that that q 1 and q 2 will be kth equivalent a k plus 1th equivalent if they are kth equivalent because yeah, so I am they are kth equivalent. So, k plus 1th

equivalent means what kth equivalent. So, k plus 1th equivalent means what? K plus 1th equivalent means delta hat of q 1 x is p 1 and delta hat of q 2 x is p 2, where p 1 and p 2 either both its final state or both irrigated state. And this is true for all x less than equal to k plus 1 length. Now, length is less than equal to k plus 1 means length is less than k. So, this has to be true for all the string of length k, so that will eventually give us this here kth equivalent, this is one condition.

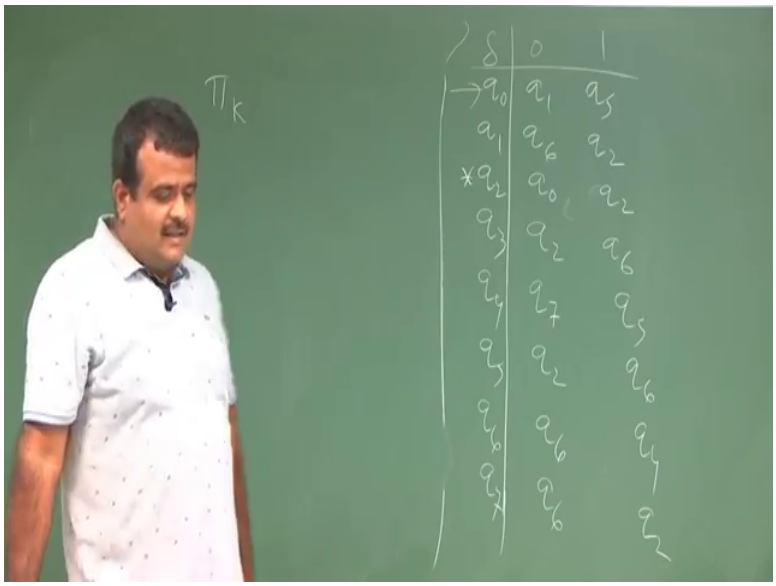
(Refer Slide Time: 03:23)



Another is there so delta of q 1 a this is a state suppose which is going to sum p 1 and delta of q 2 a it is going to p 2. Now, if there are k plus 1th equivalent then this p 1 p 2 has to be are kth equivalent kth equivalent because a is here. And this is true for all input symbol because a is one a symbol. So, if k plus 1th equivalent means if you take a string of length k plus 1 at most, so this can be written as a y where y is a, so if x is k plus 1, then y is of length y is of length less than k, because a is a input symbol, so that means, if this is kth equivalent this has to be k because we start with q 1 we take a, we are going to p 1. We start with q 2 we take a we are going to p 2.

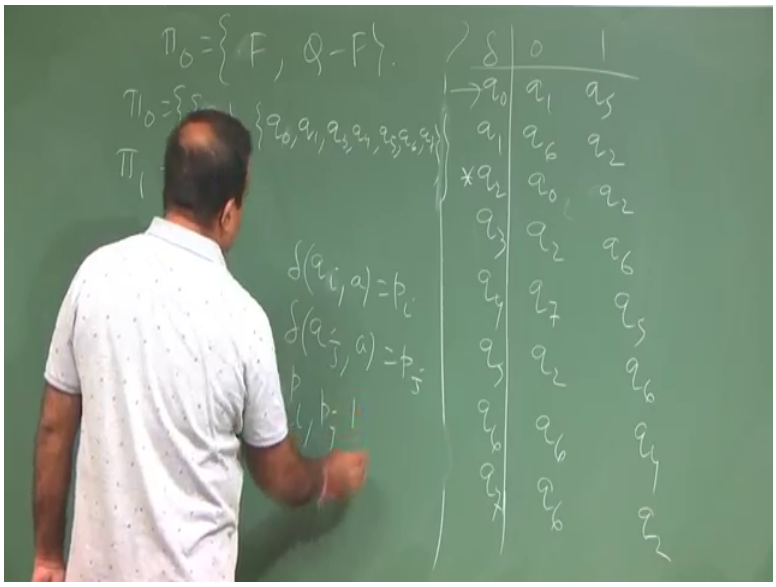
Now, then we apply this y; then we apply this y. So, they are going to the same state. If it is f 1, this is f 2 that means, these two has to go to the same state, same state means either accepted state or rejected state not same state, but same nature of the state then these has to these two has to kth equivalent. So, this is the recursive definition. And we know that this is an equivalence relation which will partition this into the equivalence classes.

(Refer Slide Time: 05:07)



And the partition we denoted by pk for the pi k for the kth equivalent class partitions.

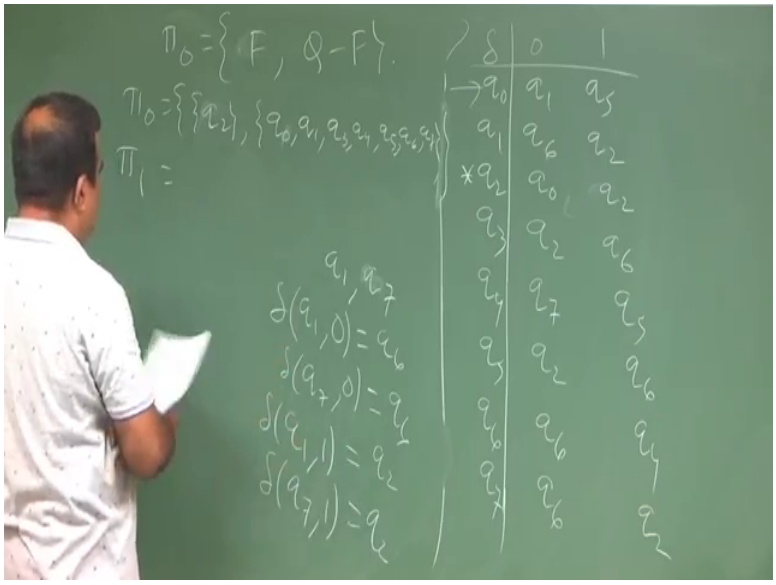
(Refer Slide Time: 05:17)



And then we have seen the pi 0 is basically the 1st equivalence class; that means, 0th equivalence class, 0th equivalent class means without reading any symbol those are those states are equivalent that means, here if this is a DFA it cannot have epsilon move, so that means, 0th equivalent means the all the final state which are remaining final state. If you take the input epsilon, it will be remain at final state. So, this is nothing but two state, one is final state I mean 2 class F minus 0.

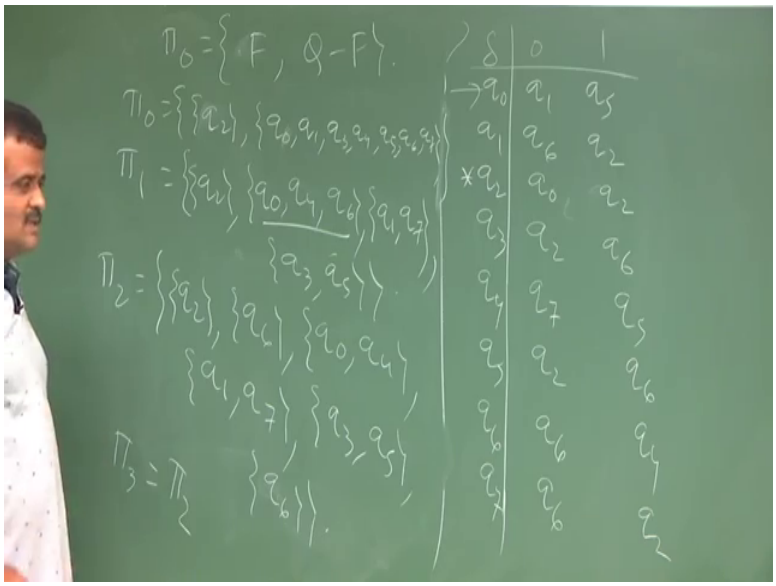
So, on this example pi 0 is nothing, but final state is q 2 and the all remaining state q 0, q 1, q 2 is there, q 3, q 4, q 5, q 6 and then q 7, all the remaining states are there ok. So, this is our 1st level. Then the 2nd level is pi 1. So, pi 1 so now, we are going to break this. So, to check this, we have to check whether some state can be 1th equivalent here. So, for 1th equivalent what they have to, so they have to come from 0th equivalent class. And for them delta of q 1 I mean qi a if it is p i and delta of q j a if it is p j then p i p j must be kth equivalent I mean 1th, 0th equivalent. So, this is the difference this is the recursive way.

(Refer Slide Time: 07:07)



So, if we check that we will be getting say for q 1 and q 7 we can check this we did in the last class, but we elaborate here. So, for q 1 and q 7, so q 1, 0, it is going to q 6 and q 7 this is q, q 7, 0, it is going to q 6, so same set. And q 1 1, q 1 1, it is going to q 2 and q 7 1 this is also going to q 2, so that means, they are in the same class. So, if you check this repeatedly for other states, we will be getting like this.

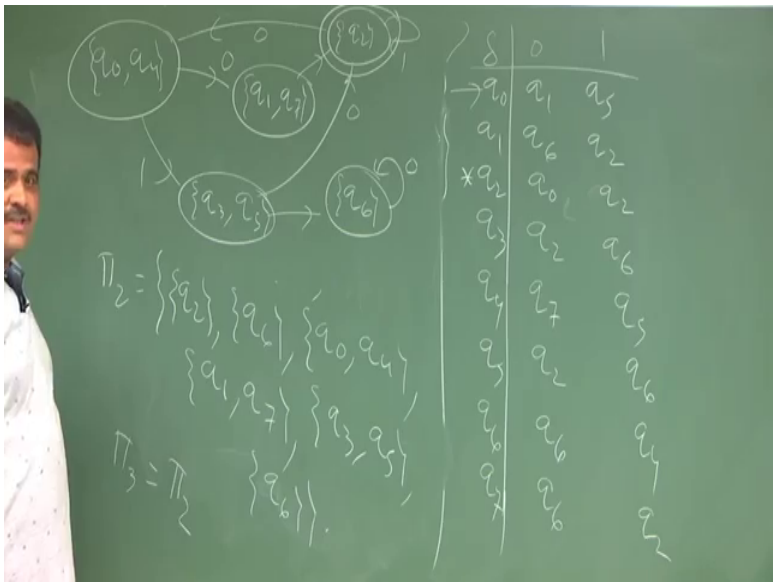
(Refer Slide Time: 07:49)



So, q 2 is cannot be furthered decomposed. So, q 0, q 4, q 6 - one class and q 1, q 7 is another class, then we have q 3 q 5 is another class this we can easily check. So, these are all one equivalent. Now, we have to find the two equivalence. For this again we check whether this can be further decomposed this can be decomposed q 6 will be separated, we can verify this. Then q 0, q 4 and then q 1, q 7, q 3, q 5 and then q 6, so, this is the final state.

And now if we want to further decompose, we can easily check there is no further decompose is possible and then it is converging. So, eventually this is the q 2 is the equivalence final equivalence class. So, once we have this final equivalence class, we can draw the we can march, we can collapse the states. Why you can collapse the state we will see.

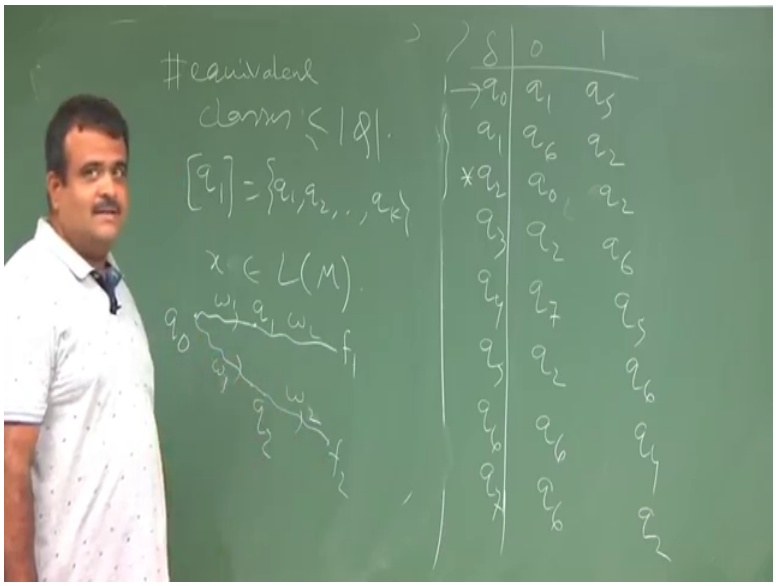
(Refer Slide Time: 09:27)



So, if you collapse these states that means we will have a state q 3 q 4 we can combined this collapsing this is one state and then q 1 q 7 this is another state q 1 q 7. And then q 2, this is another state. And then q 3 q 5 this is these 2 states are collapsing and then we have q 6 these are the 5 state. There are 8 state, it reduce to 5 state. And then we can have the r. Now, q 0, with q 0, while we are going with 0 we are going to q 1, so, where is q 1 q 1 here. So, even for q 4 also where we are going with 0 we are going to q 7. So, this can be checked because that is the way it is constructed.

And with 1 we can check this is going here. Now, from here with 0, we are going here; with 1, we can go here and from here with 1 we go here, with 0, it come here; with 1, we go here. This is the final state, because it contained the final state and then we can have 0 over here and with 1 it is going here any way we can complete this. So, this is the minimization of this DFA using this equivalence techniques. Now, why it is working I mean why we can able to collapse these states, so that idea we need to you know I mean and moreover if in general what is will be the number of equivalence classes.

(Refer Slide Time: 11:29)



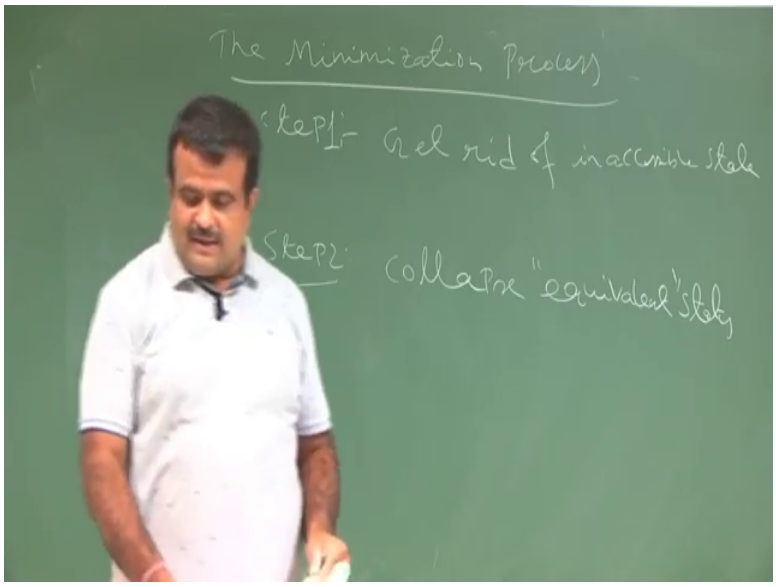
So, at most we can have all the states will be divided like at most we can have number of equivalence classes will be will be less than equal to cardinality of the number of states, because at most it will divided into single term state for all ok. Now the now let us consider the equivalence class say q 1 class; q 1 class consist of say q 1 q 2 q k. Now, concept is why we are merging this. why we are collapsing this and representing this to a single class single node in the minimization process.

Because so let us consider a string x which is a accepting string of this given DFA. Now, suppose we are starting from q 0, suppose in this process suppose somehow we are reaching to q 1 some w 1 and then we have w 2 we are going to say final state some f 1, which is same as from this if we can reach to with some w, I mean w 3 if we can reach to some q 2 say.

Now, if you reach to q 2, then also with the remaining w 4, we should able to reach to the final state because they are equivalent. They are equivalent means with a string x, if we start from here with a string x, wherever we are going with the same string or so with the same strings sorry the w 2, this is then we must go to the same status of the state. So, if it is reaching to a final state or it will both will reach to a rejected state, accepted state or rejected state that is the idea. So, somehow in the path of the execution of x, if we can encountered, any one of these states they are basically behaving same because from there, we can reach to either final state or rejected state, so that is the that is how we are

collapsing this we are merging the states to get a single state in the modified DFA ok. So, that is the idea and this is one example. So, we will do more on this. We will do more on the minimization process.

(Refer Slide Time: 14:25)



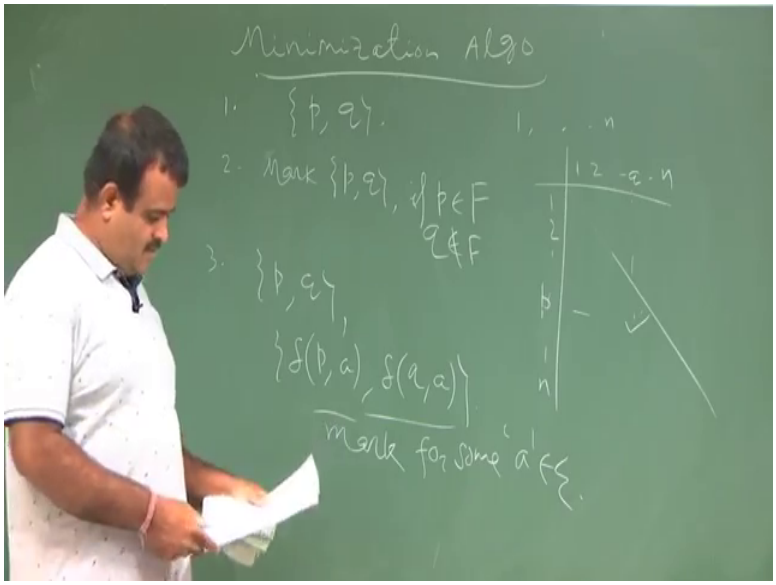
So, now we will be so the formally defined the minimization process. So, minimization process consists of two steps, step 1 we must delete those states which cannot be reachable from q 0, those are called inaccessible state. So, we must get rid of inaccessible state, that means, these are the state which are not reachable from q 0, we can delete it, because ultimately we are looking for a we are looking for the language.

So, we are starting off start state is q 0. If there are few states say p 1 like this, which cannot be reach from q 0 with some x, then the states are no one participating in the language. So, we can straight away delete those states and the corresponding transition of corresponding transition function in that state, because that are not contributing from q 0, because ultimately the language we are talking about in all are starting from the q 0 the string I mean our start state this is one.

And second one is what we did is the equivalence classes. So, step 2, we are just collapsing equivalence states. So, we have seen a method to do that, so k plus a k plus 1, kth equivalence class, then we go to the k plus 1 so this is the way. Now, we will learn another technique minimization technique for doing this collapsation I mean we are

merging those equivalence class to make a new state. So, we will learn more we will take another technique which is called tabular method to do this ok.

(Refer Slide Time: 16:47)



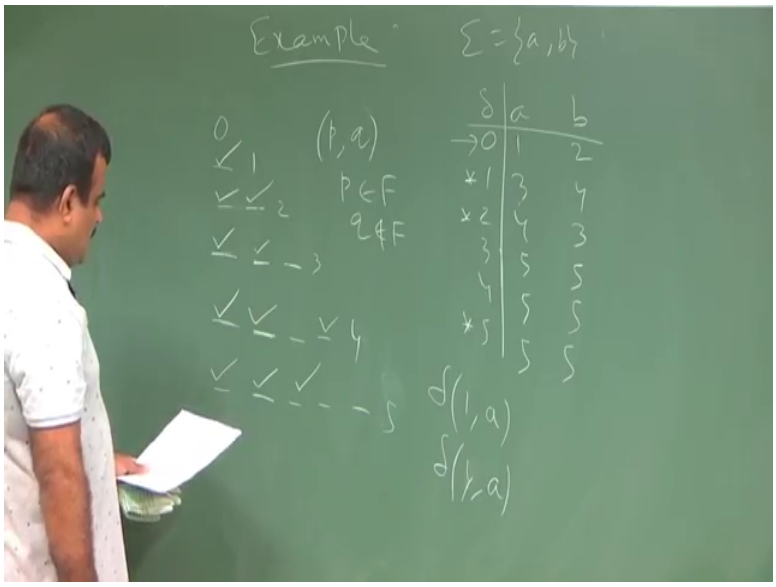
So, this is to find the equivalence classes in the tabular area. This is we can say minimization algo. So, what we do, we write down all tables in a pair of the states like p, q, we will take an example in a pair of the states. So, if we have say state say 1 to n, so we have a table like this 1 to n, 1 to n. So, this is p; this is q. And this is a lower triangular matrix, because this is a symmetric matrix. So, we will only draw the lower power lower triangular one.

And then what we do, so these we form this matrix then we mark. So, this is the first level, we mark tick mark, we will do the tick mark, we mark pq, if both are in the final state or both are in the rejected state sorry we mark p q, if they are in different state I mean if p is this and q is this or vice versa; that means, they are not colliding I mean they are this is kind of 0 equivalence class ok. So, we will check this I mean say p, q. So, if they are indifferent state, we will just put it equivalence here like this. And then we will repeat this, so, this is the 1st level.

In the second level, we will check the unmarked state. So, unmarked state which is not mark p, q, we will mark it if delta of p comma a and delta of q comma a, these two are marked for some a not for all a mark for some a. This is kind of next, next level equivalence class. We just go to the next level equivalence class, we check whether that

can be again mark or not. So, they will be in that two different set. So, we are just partitioning basically. So, this way we will continue until we can able to mark it. So, we will take an example then it will be more clear.

(Refer Slide Time: 19:47)



So, let us take an quick example. So let us take a DFA with two input alphabet a, b and this is the transition rule and these are the states. So, f q 0, q 1, q 2, q 3, q 4, q 5 there are 6 state say, so and these are the say final state q 1, q 2, q 5. Now, the suppose the transition rule is with one we are going to from 0, we are going to q 1, this is q 2, q 3 q 4, q 4 q 3, q 5 q 5, q 5 q 5, q 5 q 5 like this ok.

Now, we will draw the table. So, this is 0, here we have 1, 2, 3, 4, 5 ok. So, initially everything is not explore like this. This is 5 ok. So, now, we check which are the state we can mark, say will mark the state p, q is the first level marking p, q if p is in final state, and q is in non-final state or vice versa. So, now, here 0 is non final, so these can be mark 0 and 1, because 0 is non final state and 1 is final state.

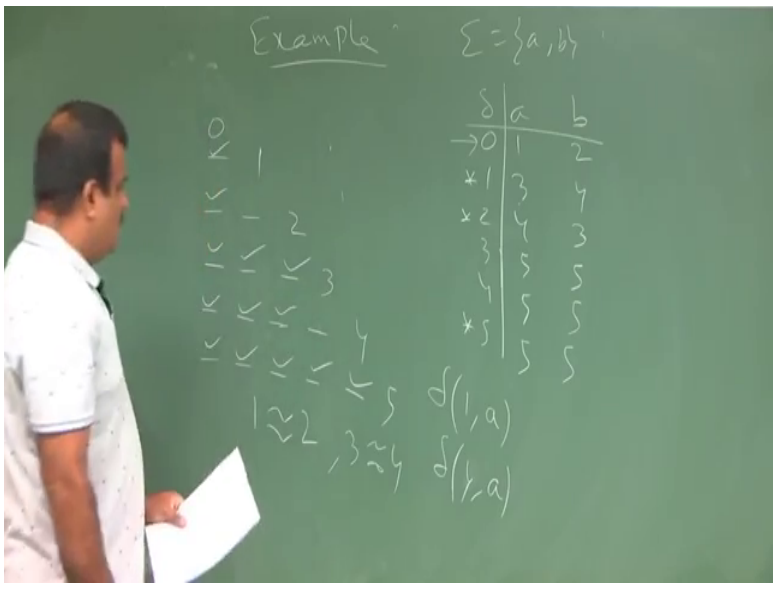
Now, 0 2 also can be mark; 0 3, no because 0 3 both are rejected state; 0 4 – no; 0 5 we can mark like this. So, this is the 1st level done, I know, not 1st level done we have to do for this also. 1 2, 1 2 in the same both are accepted state; 1 3, 1 3 – no, we will not do anything; 1 4, 1 4 is also no; 1 5, yes we will mark. Then 2 3, 2 3, 2 is the accepted state, 3 is the rejected state, we will not do anything 2 4 also not do anything, 2 5 will mark 2 4, just a minute, 2 1 2.

So, these two these two we are marking, 1 2, 1 2, 1 2 we are marking 2 4. So, 1 2 a, so 1 3, 1 3, no, 1 3 they are in two different. And 1 4 – no; 1 5, 1 5, yes, because they are in accepting state ok. Then 2 3, 2 3 is no; 2 4 no; 2 5 is yes. Now, 3 4 is yes; sorry 3 5 – no, then 4 5 – no, this is the situation first one. Now, we will check that un mark one like here. So, we will check this one 0 3. So, if we check 0 3, then we will check 0 is going to have 0 is going to 1 and 1 is going as 0 is going to 1 with a and 3 is going to 5. So, 1 5 is mark, 1 5 is mark. So, we will mark this. Then we have one 0 4, so 0 4.

For 0 4, what we do. So, 0 4, if we take 1, it is going to if we take a it is going to 1. And if you take a with 4, it is going to 5, 1 5 is mark, 1 5 is mark. So, this will also mark. Now, this one this is 1 3, for 1 3 what we check we check delta of 1 a and delta of 3 a, for any a if we have at least 1 a which is going to the mark, then we will mark it. So, 1 a, 1 is going to where, 1 a is going to 1 a is going to 3, q 3; and 3 a is going to 5, 3 5 is mark, yeah, 3 5 is mark. So, we can mark it. So, we can mark this, which one is you are marking 1 3.

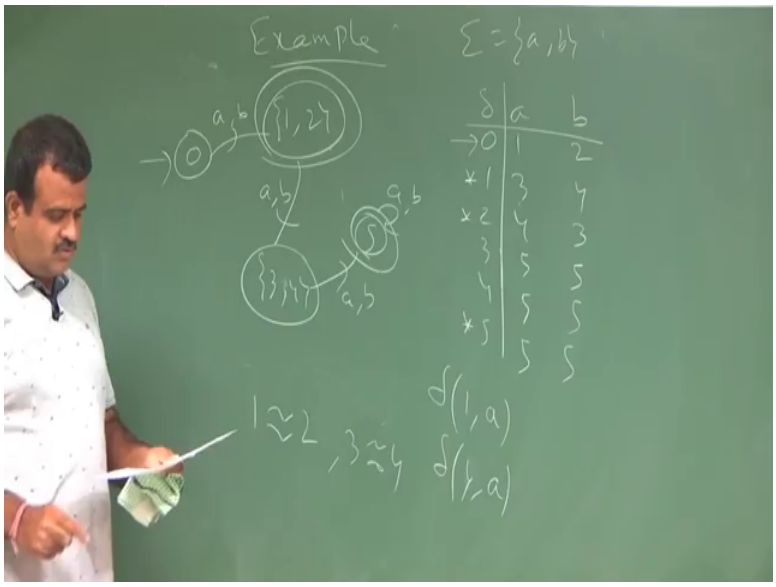
So, 1 3 we have mark. Then if we continue like this, so 1 4, 1 4 calls we can see 1 4. So, this is 1, this is 4. 1 is going to 3; 4 is going to 5 and 1 5 is 1 5 this mark, this is mark. So, if you continue like this, so we will see ultimately this is the final table we can just write the final table over here.

(Refer Slide Time: 25:37)



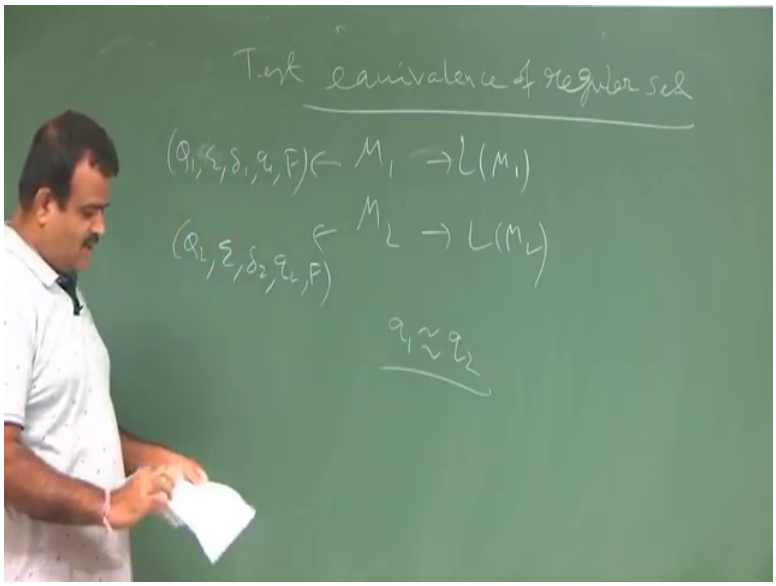
So, this is 0, 1, 2 and 3, 4, 5. Now, we can check this and I am writing the final state. So, this is mark, this is mark. We can easily verify this. This is mark; this is mark; this is mark; this is not mark; this is mark; this is mark. So, we can easily check this 1 2 is not mark and 3 4 is not mark. So, the that means, 1 2 are in the same class and 3 4 in the same class that means, we can remaining are in the different class. So, this is just we can have the corresponding graph.

(Refer Slide Time: 26:31)



So, this is q 0 and 1 2, q 1, q 2 in the same, same; and 3 4 in the same, and here 5. And then we can have this a, b we can draw based on this rules a, b, and this is a, b and which are the final state. So, this is the final one of the final state and 5 is the final state, these two are the final state, and this is the starting state. So, this is one minimization process using the yeah using this thing, using the table tabular method. And we can use this for find the equivalence between two DFA.

(Refer Slide Time: 27:33)



We say we can test this test the equivalency of regular language, equivalence of regular set. So, given two DFA M 1 and M 2, how to test they are accepting same language? So, this is a L of L of M 1, this is L of M 2. So, we want to check there are equivalent or not. So, we can form a DFA combined DFA and then if the both the starting state are equivalent class in the same class. If the start state are you using this tabular method, if the start set says suppose this is say q 1 sigma is a same; q 1 f and this is say q 2 sigma is same delta 2.

Now, if this q 1 is equivalent to q 2, then both are same. So, we have an example for that which will be given in the lecture note. This is the way we can find out that whether two DFAs are equivalent or not by using this tabular method.

Thank you.