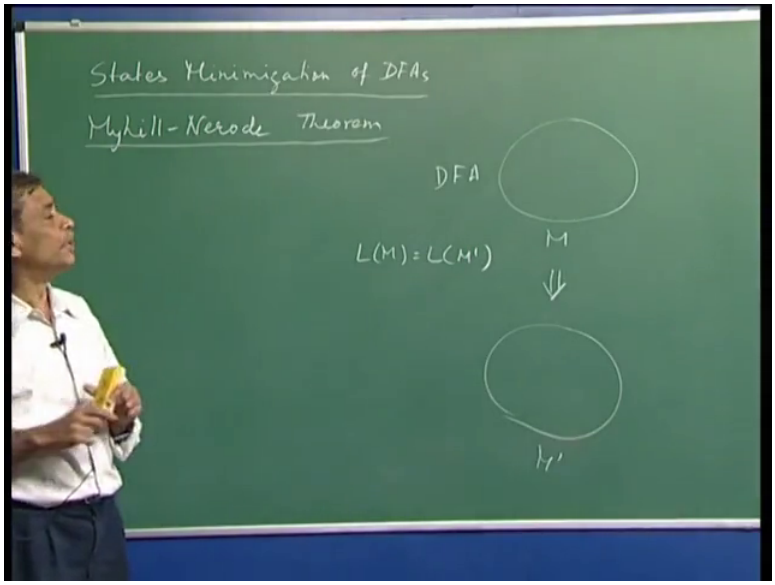
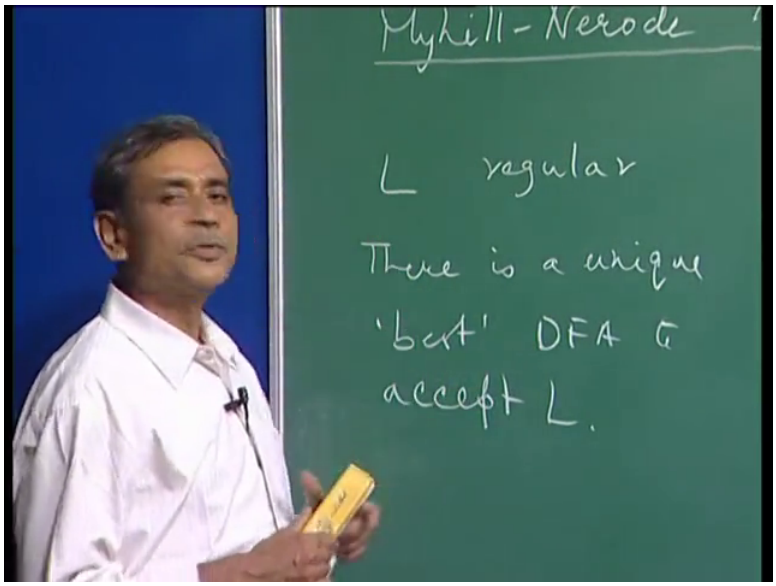
Theory of Computation.
Professor Somenath Biswas.
Department of Computer Science & Engineering.
Indian Institute of Technology, Kanpur.
Lecture-16.
About Minimisation of States of DFAs, Myhill-Nerode Theorem.

(Refer Slide Time: 0:31)





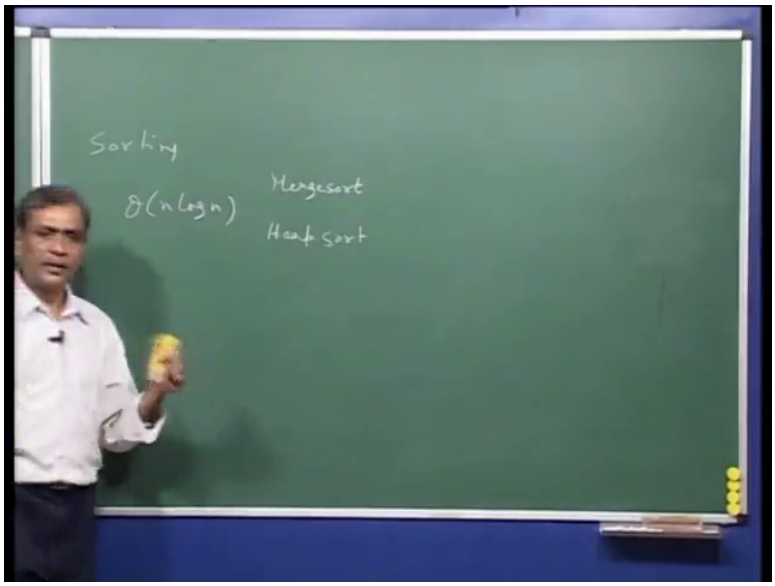
Our final topic for regular languages will be minimisation of states of dfas, what we mean by that is, suppose that I have a DFA M and it has a number of states. Then we will provide an algorithm to obtain another DFA M dash, such that dfas are equivalent in the sense both of these dfas accept the same language. Further this new DFA M dash is such that there can be no other DFA which states smaller than the number of states in this M dash to accept the

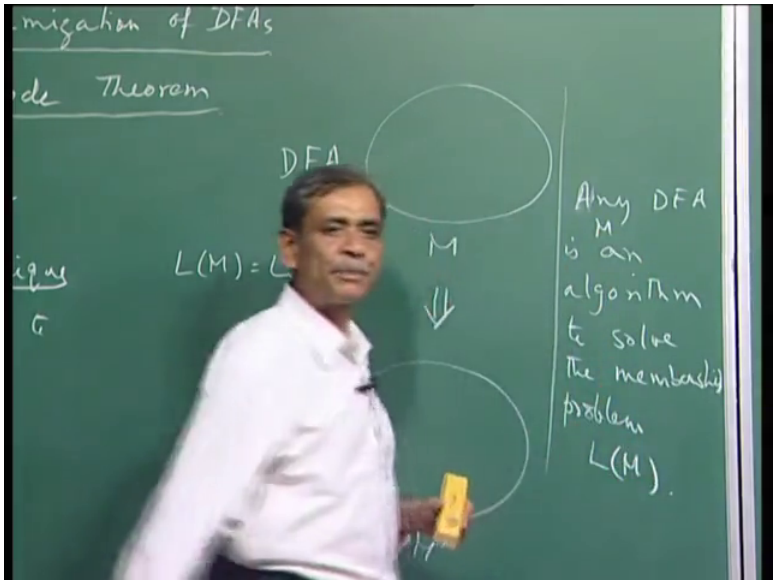
same language. In that sense we will be able to minimise the number of states required for a DFA to accept a particular language.

And will, that is an algorithm but what we will 1st do give a very interesting Theorem called Myhill-Nerode theorem and this theorem is the theoretical underpinning for the algorithm that we are going to describe for minimisation of states. But this theorem says something beyond, that what I just proved. Roughly what it says is that for any regular language L let us say is regular, what Myhill-Nerode theorem states, there is a unique 'best' DFA to accept L. Now best in the sense that there cannot be a DFA which has lesser number of states that the DFA that our theorem is going to suggest.

However more importantly or equally importantly that DFA is unique. In other words what we are saying, it is not possible to have 2 different dfas, essentially different dfas. Of course you can have different dfas by changing the names of the state but if you do not bother yourself with such trivial changes, there is precisely one DFA with minimum number of states to accept regular language. Now let me spend a minute to understand what I am saying. Take for example a problem like sorting.

(Refer Slide Time: 3:38)

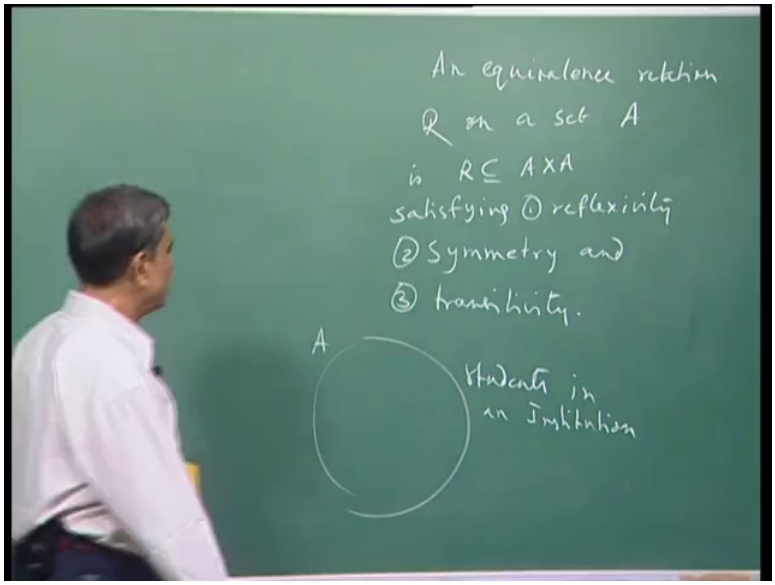


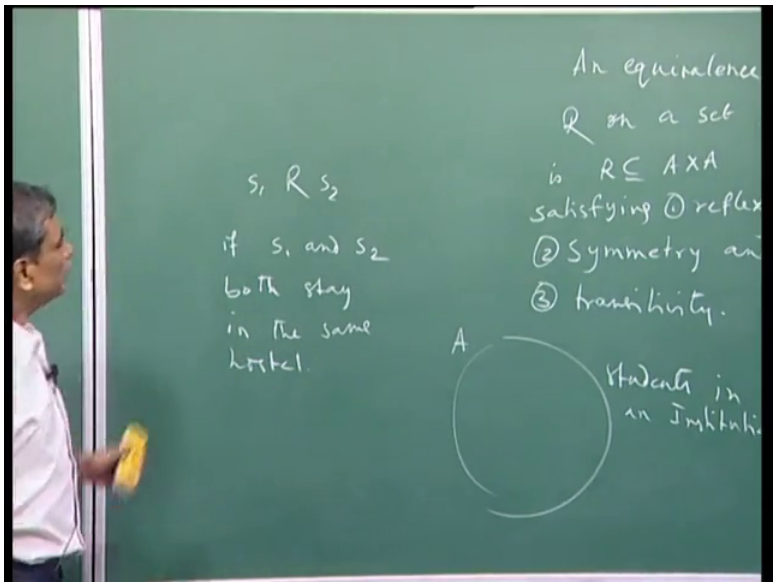


We know that for sorting I have an optimal algorithm which is order nlogn time complexity, such an optimal algorithm will have nlogn time complexity, where N is the size of the input that you are you are interested in sorting. Now you will remember that we just do not have one optimal algorithm for sorting, there are essentially different algorithm which will give me the same nlogn time complexity and yet they are very different. Example is mergesort is one nlogn algorithm and heapsort where you create a heap and that heap essentially, repeatedly tells you what is the smallest number of the largest number. Both these algorithms are optimal but yet they are very different, you cannot say that merge sort is same as heap sort, although they are doing the same job of sorting.

Here what we are trying to say when we are saying there is a unique DFA, 1st of all you can think of a DFA also as an algorithm which solves the, you know what I am saying is that any DFA is a algorithm to solve the membership problem, so any DFA let us say M, it solves the membership problem accepted by M. Now if you think of this class of algorithms which can be given by DFA, for one, once you fix a language L and if you restrict yourself to dfas and the class of algorithms, then there is a unique optimal algorithm in the sense, not so much in the sense of time complexity but in the sense of the number of states that you would use.

(Refer Slide Time: 7:39)





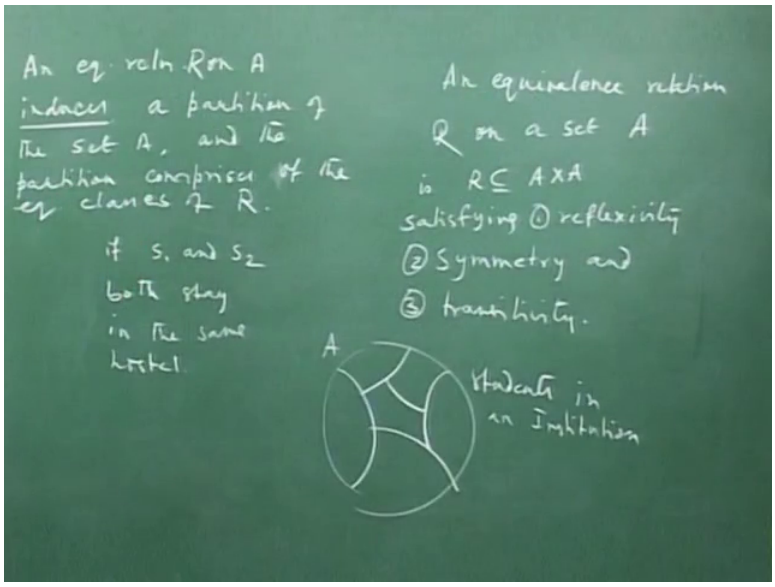
So this is a very interesting Theorem Myhill-Nerode theorem and to understand Myhill theorem, even to state it, I will require certain terminology in terms of equivalence relations. We will need to refresh our understanding of equivalent relations and we all remembered an equivalence relation R on a set, let us say A is what, is a subset of A cross A, in other words any relation for that matter is a set of pairs satisfying reflexivity, symmetry and transitivity. Right. For example you think of the students in your institution. So our capital A is the set of students in an institution, in let us say and then we say that 2 students here, let us say S1 and S2 are related by the relation if S1 and S2, both stay in the same hostel. Okay.

So clearly this relation satisfies reflexivity because if we are talking of S1 related to S1, of course that is true, S1 lives in the same hostel as S1, that is true of everybody. Then symmetry

is the student S1 lives in the same hostel as the student S2, then of course means the student S2 also lives in the same still as student S1, so symmetry is satisfied and transitivity is also similarly satisfied. Suppose S1 and S2, they live in the same hostel and S2 and S3, they also live with the same hostel, therefore all 3 live in the same hostel, therefore S1 and S3 also live in the same hostel. So transitivity is satisfied.

Important aspect of an equivalence relation is that it partitions the set one which the relation is defined in a number of equivalence classes. Okay. By partition of course, what means, that together partitions are subsets of A, such that we subsets together will become the entire set A and no 2 such subsets have anything in common. So again in this example it is easy to see what the partition that will be induced by this particular relation on the set of students in that institution of our interest. By the way, these particular classes will be what?

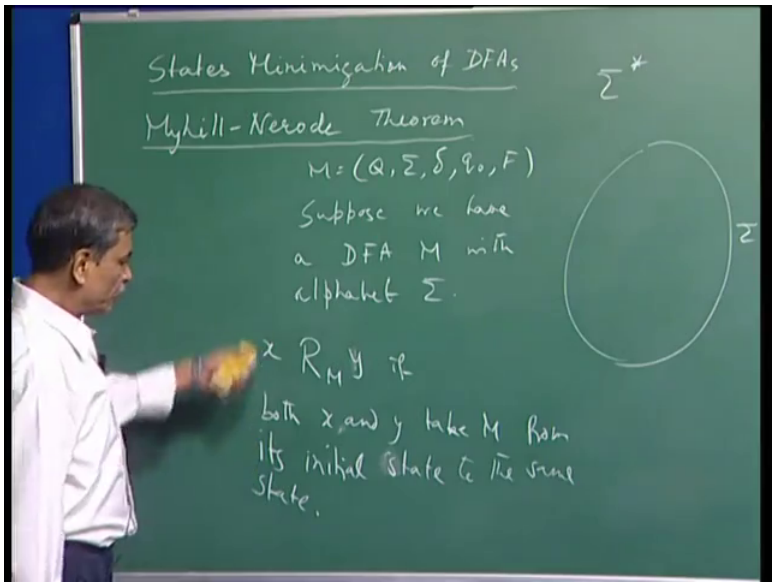
(Refer Slide Time: 12:10)

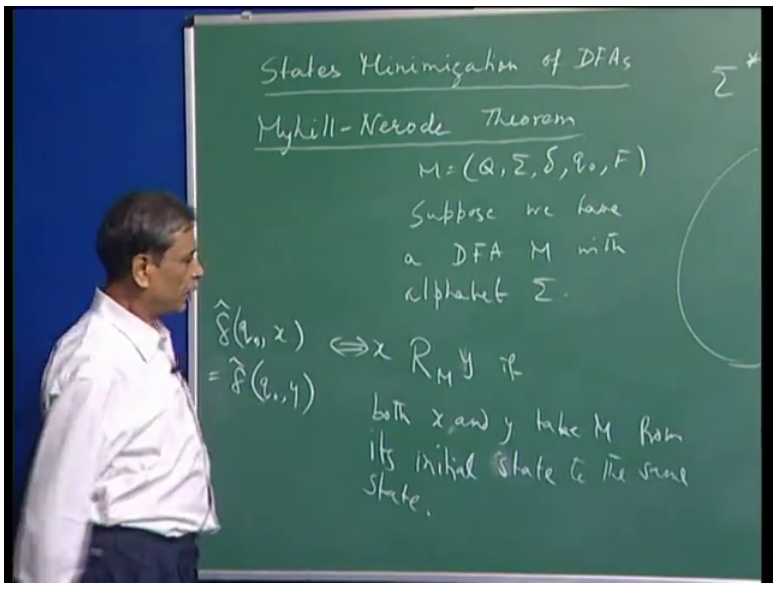


This these are called equivalence classes and what is the equivalence class, the set of all elements which are related to each other. So in this case all the students living in one particular hostel, they are of course related, they form one equivalence class. So let us say another hostel, hostel number 2, the students who are there in that hostel, they form the 2nd you know the another equivalence class and so on. And together if you take these equivalence classes, that means if we just take the hostels, then of course they are they will cover the set of all students in distribution, of course assuming the institution is residential, all students are supposed to live on campus in hostel or the other.

Right so the important fact about the equivalence relation, let me write it, an equivalence relation on A induces a partition of the set A and this partition constitutes or comprises of the equivalence classes of R. So as you can see in this example.

(Refer Slide Time: 13:45)

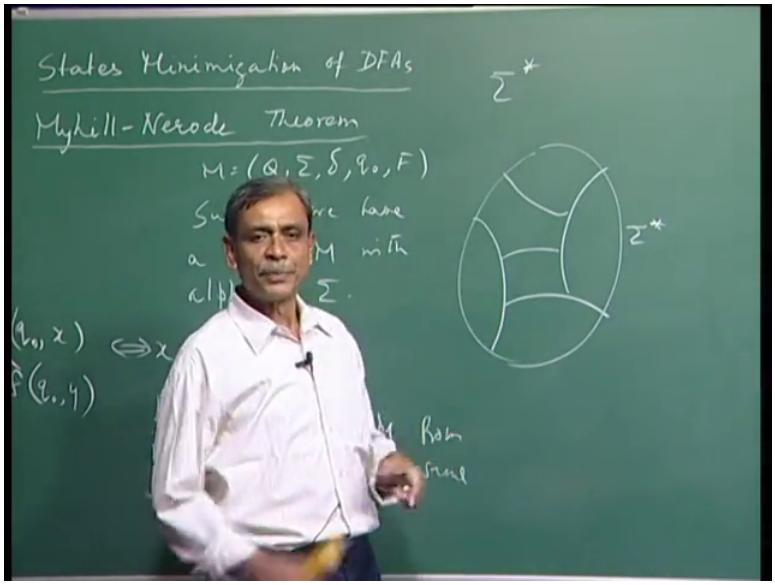




Now our theorem, Myhill-Nerode theorem will be about some equivalence relation but on Sigma star. You remember Sigma is your Alphabet, Sigma star is the set of all finite strings that we can build using the symbol Sigma. And So I can in a similar way I can think of this Sigma star in the set, of course it is an infinite set and we will consider certain equivalence relations on Sigma star. Now let us start with something fairly familiar, which is that suppose we have a DFA M. Right and M uses alphabet Sigma. Let me define this relation RM as, we

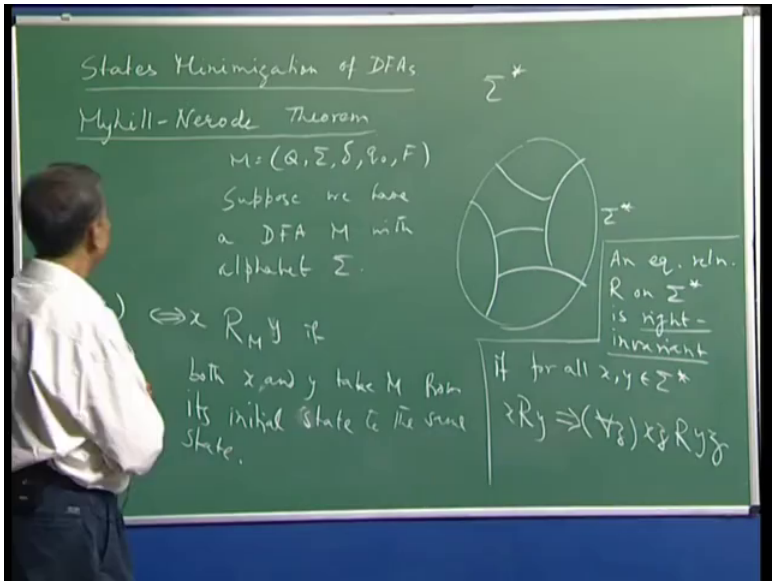
say that x is related by this relation RM to y, if both x and y take the machine M from its initial state to the same state.

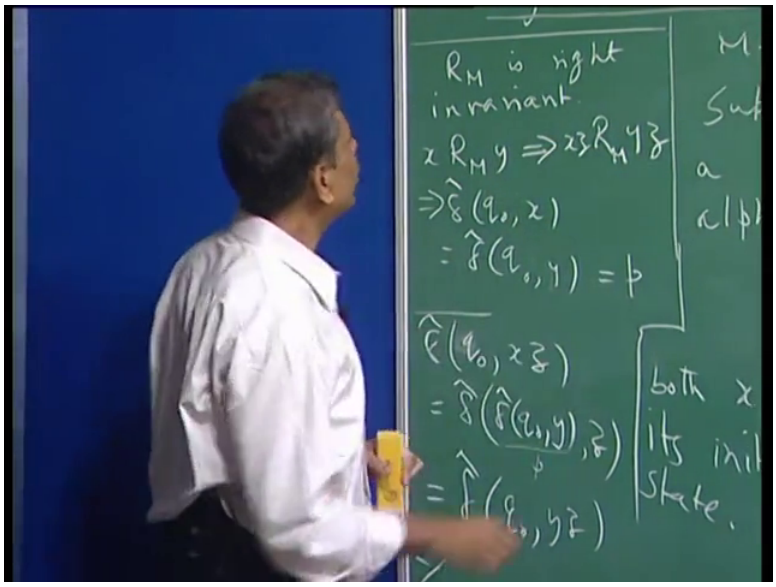
(Refer Slide Time: 17:04)



So if M was q, Sigma, Delta, q0 and F, then of course what we mean is, another way x rmy, if and only if Delta hat of q0 which is the initial state of x which gives me a state is same as the Delta hat of q0 y. Now it is fairly simple to see that this relation RM is also an equivalence relation, we just have to verify that the relation RM will satisfy reflexivity, symmetry and transitivity. Further if of course, being an equivalence relation and being defined on the set Sigma star, the relation RM induces a partition on Sigma star and what are these equivalence classes, the partition classes? Clearly these are, this is the set of all strings which takes the machine from the initial state to one particular state, this is to another state from the initial state and so on.

(Refer Slide Time: 17:55)



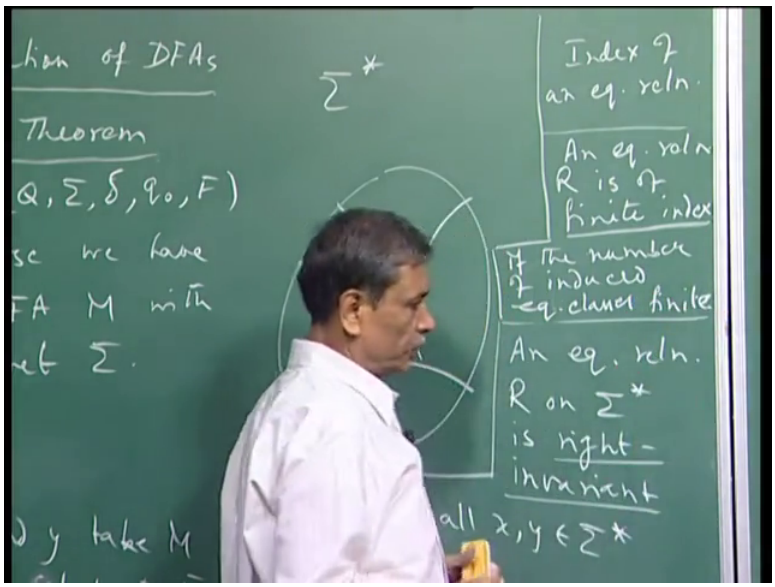


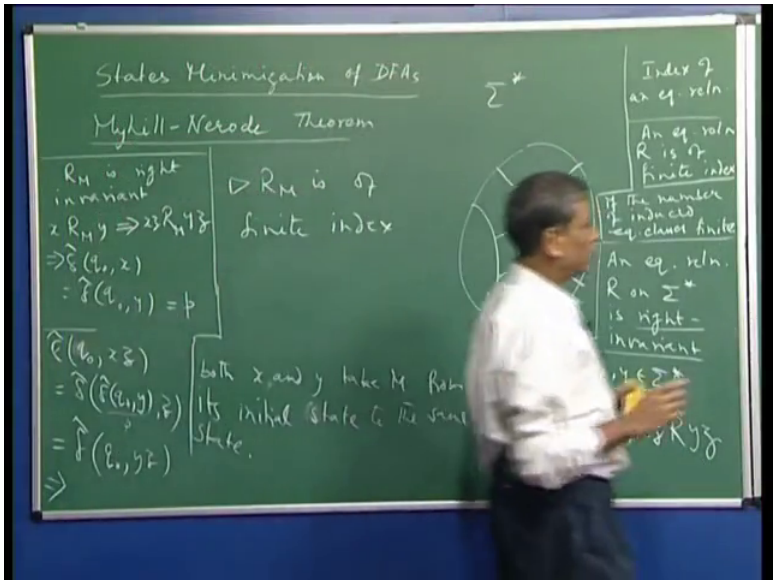
So essentially if you take Sigma star and take the set of states, then all strings which take the machine to one state constitutes one of them and so on. Now there are 2 other definitions, 3 other actually definitions that I need concerning equivalence relation on Sigma star. So 1st of all one thing is, another definition is, let me write it here. An equivalence relation R on Sigma star, from now on we are just concerned with certain equivalence relation on Sigma star, is Right invariant, this is the definition, if for all x, y in Sigma star, x, if x and y are related, then for all z, xz will also be related to yz, is it clear? That what we are saying is if x and y are related, then you take any string z, you concatenate that string z to x and then to y, you will get 2 new strings xz and yz.

You will find that xz and yz, they are also related in case of course the relation is Right invariant. Now you can check that this relation RM is Right invariant. Why? Let us say, let me take this out, suppose xrmy, that is what, that both x and y take the machine from q0 to the same state, right. So in fact we have written that here. And now you consider, so this implies of course that Delta hat of q0x is same as Delta hat of q0 y. And let say that this state is the state P and now you concatenate a string z to both x and y, right. So what is Delta hat of xz, suppose Delta hat of xz, q0 x, z, we know it is going to be Delta hat of, Delta hat of q0, x. Firstly which state the machine goes to on x, then from that state, which is of course is B, user string z to come to a particular state.

But since Delta hat q0 x is same as Delta hat q0 y, I could have replaced this q0 x part by q0 y. And that would mean, this is of course them as Delta hat of q0 yz, so therefore both xz and yz take the machine from q0 to some same state. And this of course implies that xz and yz are related. So I can say xz related at this relation RM to yz. So the important point is the relation RM is Right invariant.

(Refer Slide Time: 22:16)

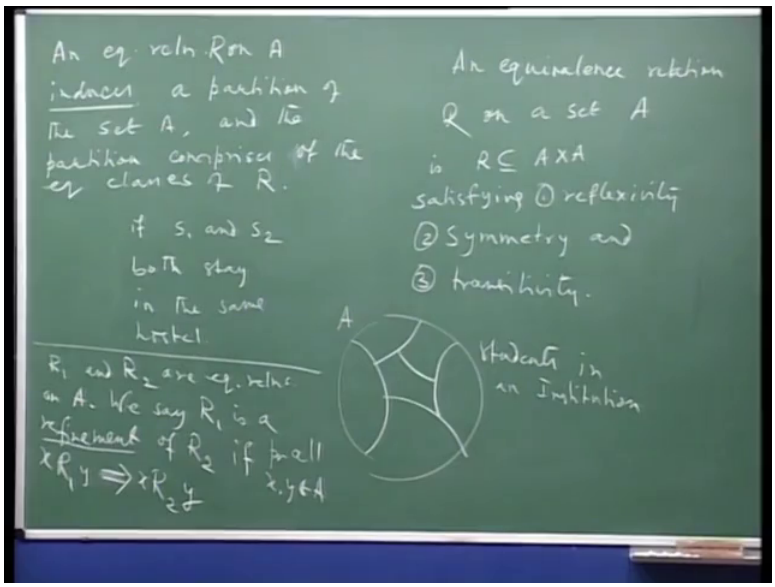




There is another definition I need and which is index of an equivalence relation. Now this is a very simple definition, it says an equivalence relation R is of finite index, if the number of induced equivalence classes is finite. So what it is saying that remember, you start with an equivalence relation that will partition the set on which the relation defined is defined on a number of equivalence classes and this number could be finite, it could be infinite. If it is finite, then we see the relation is of finite index, that is what we said. Clearly in this example, the students in an institution and they being in the same hostel is our relation, since the number of hostels in an institution is finite, so this relation is of finite index.

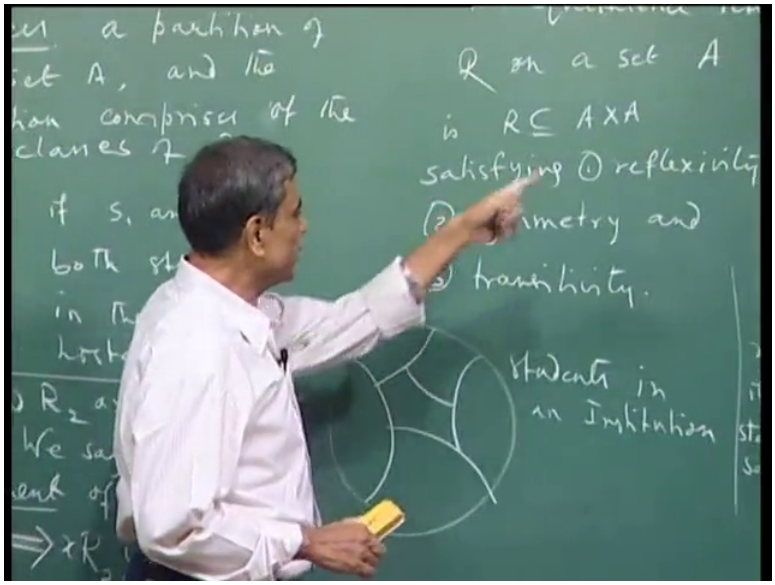
And more importantly or more relevantly for us the relation RM is also finite index. Right. Why? Because you can see that I have an equivalence class for each state in the machine M, right, that is the set of strings, that state is closed and that equivalence class consists of those strings which take the machine from the initial state to that state after fixing a particular state. And so these, you know each one of these equivalence classes as induced by RM, each one is associated, can be associated with the state and since the number of states in a DFA is finite, so therefore the number of equivalence classes is also going to be finite for RM.

(Refer Slide Time: 26:00)



And therefore RM is of finite index. And one more definition before I can state the Myhill-Nerode theorem is, now this concerns 2 equivalence relations on the same state A. Now we say the relation R1, so let us say R1 and R2 are equivalence relations on A, we say R1 is a refinement of R2 if the following condition holds. X R1 y implies x R2 y. So this is the definition of when 2 relations, 2 equivalence relations are such that one equivalence relation is a refinement of the other equivalence relation. Very simple definition, it says that R1 is a refinement of R2 if whenever you have 2 elements related by R1, then they will also be related by R2, so of course true for all x, y for all x, y in the set A. Let me let me give an example of this relation that we have studied or just mentioned for the students in an institution, living in the same still being related by the equivalence relation.

(Refer Slide Time: 28:11)

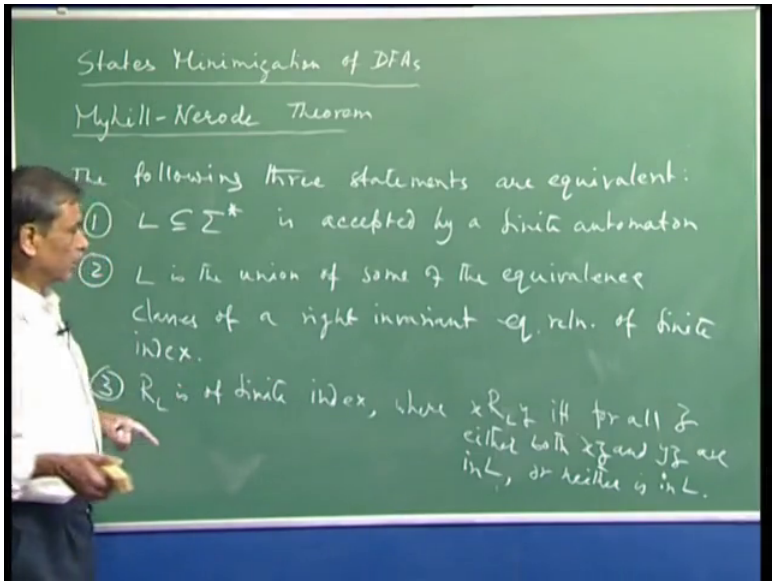


Now let me define another relation R dash, this says that xr dash y, if the 2 students x and y stay in the same block, I am assuming that a hostel typically, as in my institution has a number of blocks, suppose these are numbered A, B, C, D, etc. So let us say this is hostel 5 and this has these blogs, block A, Block B… Block F. Now what we are saying, 2 students, they are related if they are in the same block. To be the same block they have to be in the same hostel. So you see this relation R dash is a refinement of the relation R of being in the same hostel and what happens in such a case?

When the relation R dash in this case R1 is a refinement of R2 in this particular example, R dash is a refinement of R1, all that happens, if you look at, in fact if you look at this, it is saying that suppose you, your relation is such that, new relation that it breaks the old equivalence classes in, maybe number of finer classes. Because old relation, everybody in the same hostel were clubbed together in the same equivalence class. Now we are going down to the level of blocks and that is the reason we are saying or that is the reason it is, the term is used that it is a refinement. You are refining the equivalence classes to finer chunks.

So we have given in terms of Sigma star, so we have seen some terminologies, one is that finite index, the other is Right invariant and now I said another notion, that of one equivalence relation being a refinement of another equivalence relation. And of course we give an example, particular example that RM, which is a very natural equivalence relation associated with a DFA M on the set Sigma star. I am now ready to state the theorem, the theorem states the following.

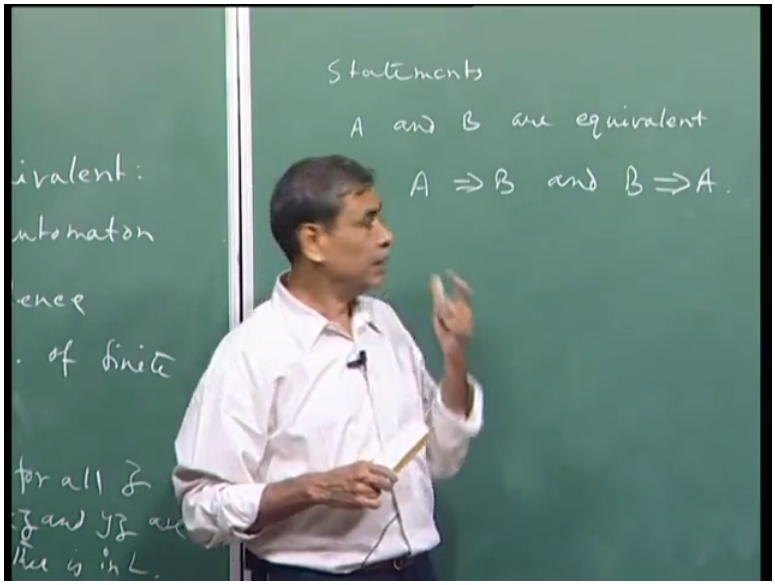
(Refer Slide Time: 31:18)

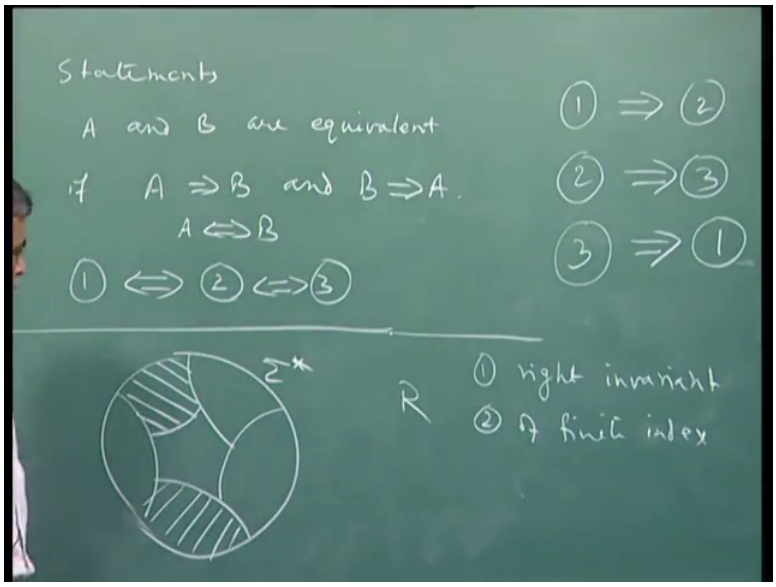


It says the following 3 statements are equivalent. And the 3 statements are, L, the language L over the alphabet Sigma is accepted by a finite automaton, in other words L is regular, L is a language over Sigma and that is regular. The 2nd statement says, L is the union of some of the equivalence classes of a Right invariant equivalence relation of finite index. And 3rd says, RL is of finite index where you define RL, of course I need to see what RL is, where x RL y, if and only if 2 strings, remember that all these things are over, these are, these are relations over Sigma star, if for all z, either both xz and yz are in L or neither is in L.

We will try to understand this definition a little more clearly when we come to this part. Now, this is the theorem and as you shall see that this will imply that there is a unique minimum state automaton, unique to the extent of you know modulo names of the states. 1st of all, I do not know whether you have seen these forms of results, what it is saying is that all these 3 statements are equivalent. What does it mean to say 2 statements are equivalent?

(Refer Slide Time: 35:17)





Remember the little bit of logic that we have learnt in courses like discrete maths. We say statements A and B are equivalent, if A implies B and B implies A. Remember a statement is something which is either true or false, what we are saying is that if A is true then B must be true, as well as if B is true then A is true. In other words, basically it is the same statement, in in its integrity, right. You know many examples of equivalence. Now what this theorem is stating that all these statements are equivalent.

Now if you say the statement 1 is, remember that you also write 2 equal statements as A is equal to B, the statement equivalent, A is equivalent to B, so what this is saying is 1, the statement 1 that a language L is regular is equivalent to saying that this particular language L will discuss what it is saying but look at the way it is and it is saying 1 is equivalent to 2, 2 is

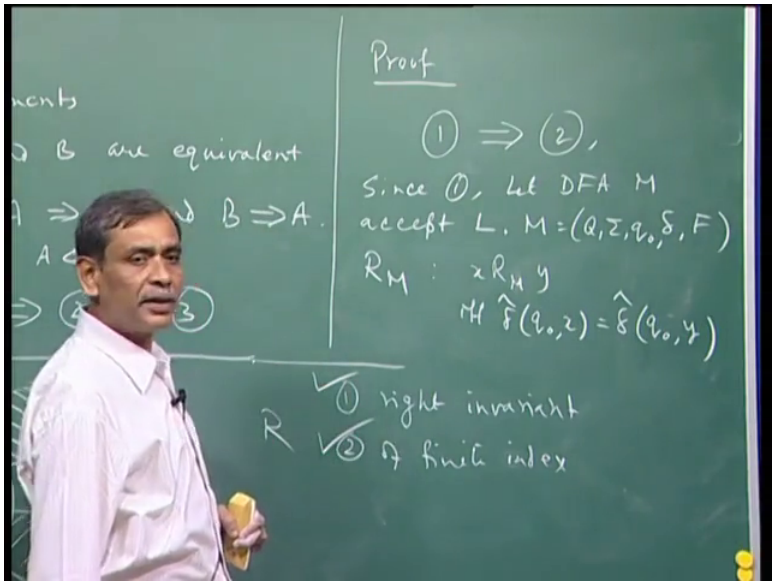
equal to 3 and therefore of course 1 is equal into 3 and so on. So typical way people prove this is this, that they will show that the statement 1 implies statement 2. Then they will show that the statement 2 implies statement 3 and then finally they will show that the statement 3 implies the statement 1.

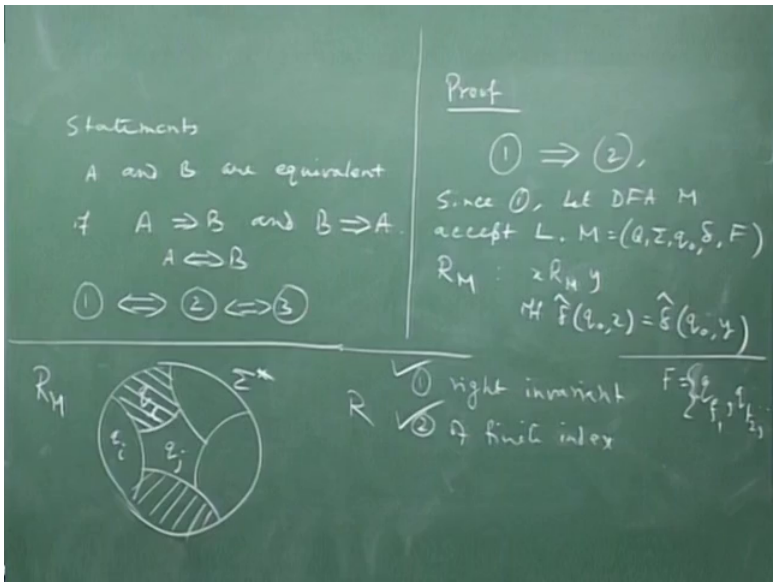
So you know it is a cyclic dependency, 1 implies 2, 2 implies 3, 3 implies 1, therefore these are true of course, that 1 is equal into 2, why, because in the 1 implies 2, 2 implies 3, right and 2 implies 3 and 3 implies 1, so therefore 2 implies 1. So you can see this is how one way of proving a number of statements to be equivalent and this is what we are going to show. Before we start, look at what is the 2nd statement is saying. 2nd statement is saying that the language L, which is over Sigma star is the union, so let me do it here, remember our set on which we are interested is of course Sigma star and what is the language L, L is always a subset of Sigma star.

And now it is saying that L is the union of some of the equivalence classes of some particular equivalence relation which has a certain property. What is that, there are 2 properties, that equivalence relation is Right invariant and it is of finite index. Now see it is saying that L is the union of some of the equivalence classes, so 1st of all that, if that relation is, that the equivalence relation is R, that will induce as we have seen on a on Sigma star partitions. Now what it is saying is, this statement the L is the union of some of the equivalence classes. So maybe L comprises of this equal is class and this equivalence class, all the strings in this equivalence class we take union with this equivalence class and that gives you the language L.

Further it is saying that this equivalence relation R satisfies 2 properties, one it is Right invariant and 2 it is of finite index. Right. So the relation R, since it is of finite index, the number of equivalence classes induced will be finite, there will be some particular number, but only finite. Although the set Sigma star is infinite, the number of equivalence classes will be finite, of which, this of this finite number of equivalence classes we take the union of some of the equivalence classes that gives me the language L. Further we are also saying in this statement that the relation R is of, is Right invariant.

(Refer Slide Time: 41:05)





Actually we have already done this proof, that 1 implies 2, proof of 1 implies 2. What is that proof? We will see, consider RM, remember 2 strings where and that RM is going to give me this. But you see how it is 1 implies 2? Since 1, so we, we say suppose 1 is true, that means L is accepted by a finite automaton, therefore we can say let L is accepted by a DFA M, let me write it properly. Since 1 holds, L, let DFA M accept L and now I claim the relation RM is satisfies this property. We had shown that RM, the relation RM, recall what the relation RM was, we had said that if my DFA M is q, Sigma, q0, Delta, F, we said x RM y, if and only if Delta hat of q0 x is same as Delta hat of q0 y, that was the definition of RM.

And 1st of all it is easy to see that this relation RM is of finite index, right because its equivalence classes, they correspond to the states of the machine M and M being a finite state

machine, it has only finitely many states and therefore this R satisfies this, this we had seen, it is easy to see, we had proved a little earlier that RM is Right invariant, right. And now what is L? So you see each of this, suppose I am now talking of RM, each of these equivalence classes, each of them corresponded to a state of the machine M, so this may be qi, this may be qj, this may be ql and so on.

And these are all the strings which take the machine from its initial state to this state q I, this particular set of strings which constitutes, which will give me this equivalence class, they take the machine from q0 to say state qj and so on. Now the language L is what? Language L we know is the, all those strings which will take the machine to one or the other final states. So you see, suppose this F consists of qf 1, etc. Q F1, qf2 and so on. So now I will have an equivalence class in RM corresponding to q F1 and all those strings in that class, they are in the language because the machine will take all those strings to qf1, starting from the initial state.

But qf 1 is the final state, so therefore they are in the language, and therefore it is easy to see that the language L is nothing but the union of the equivalence classes corresponding to those states which are the final states of the machine, right. So once we understand that, then 1 implies 2, this part of the proof is immediate. So we will continue with the other 2 parts, 2 implies 3 and 3 implies 1 in the next lecture. But let me end by saying what are the concept that we have learnt, essentially we are saying, we are trying to view the regular language in a certain way, in terms of certain equivalence classes of certain equivalence relations and we learnt certain concepts about equivalence relations.

1st of course we remembered that and equivalence relation will induce a partition on the set which it is defined, that is 1st fact that we recalled and then we defined notion of finite index for a equivalence relation. We said an equivalence relation is finite index if it induces a partition where there are only finitely many equivalence classes and then another concept we learnt was that of Right invariant. We said an equivalence class over Sigma star is right invariant, if whenever x and y, the 2 strings are related, then if you add, concatenate another string, same string z to both x and y, then the 2 new strings xz and yz, they will also be related, that was the notion of right invariance.

And finally we talked about refinement which we have not yet used so far but which we will use in the proof of 2 to 3, this statement implying this statement. And we said a relation R is a

refinement of the relation R dash if whenever 2 strings are related by R, they will also be related by R dash.