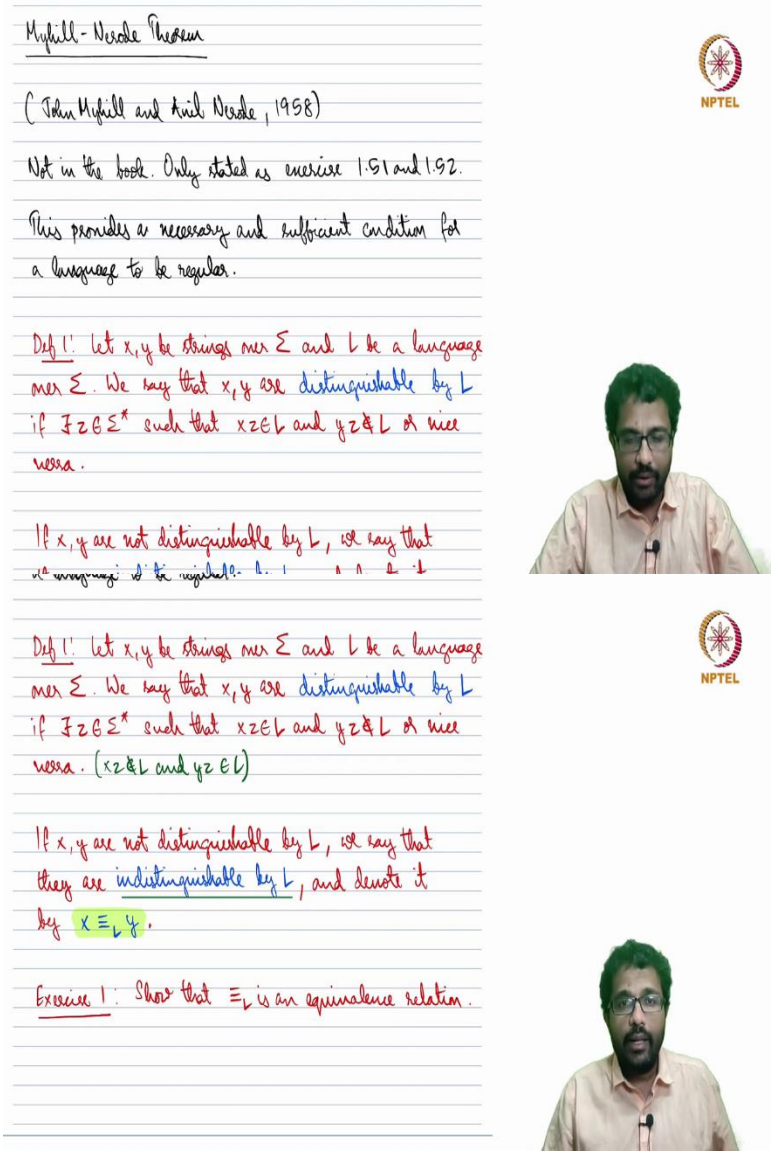
Theory of Computation
Professor Subrahmanyam Kalyanasundaram
Department of Computer Science and Engineering
Indian Institute of Technology Hyderabad
Lecture 14
Distinguishability of Strings and Myhill-Nerode Theorem

(Refer Slide Time: 00:15)



Hello and welcome to Lecture 14 of the course Theory of Computation. In the previous lectures, we saw pumping lemma for regular languages, which was a way to show that certain languages are not regular. Pumping lemma was a necessary condition for languages to be regular. So, if a

certain language does not satisfy those conditions, then one could infer that, that language is not regular.

So, as mentioned during the lecture, we cannot use pumping lemma to show that a language is regular. We can only use pumping lemma to show that languages are not regular. So, in this lecture, we will see Myhill-Nerode Theorem, which was discovered by John Myhill and Anil Nerode in 1958, which is a necessary and sufficient condition.

So, once we try to verify the conditions in the theorem, if the conditions are met, we know that the language is regular and if a condition is not met, then we know that the language is not regular. So, this is both necessary and sufficient. So, in that sense, this is superior to the pumping lemma, but then checking the conditions themselves is not that easy as you will see.

One point is that, this is not there in the Sipser book. It is not explained in the book. It is listed as two exercises 1.5 and 1.52. However, we will go through it in detail and we will also give all the proofs that are necessary. So, Myhill-Nerode Theorem is in necessary and sufficient condition for languages to be regular.

Before we get to stating then theorem itself, we need some definitions. We need to set up some definitions. So, the first definition is being distinguishable by a language. Let x and y be strings over alphabet Σ and let L be language over the same alphabet. We say that x and y are distinguishable by the language L if there is a z that we can append to x and y such that one of them belongs to the language and the other one does not belong to the language.

So, either $xz \in$ L and $yz \notin L$ or vice versa. Vice versa means the opposite, $xz \notin L$ and $yz \in$ L. So, we say that it is distinguishable by L if you can append some string z to x. So that, upon appending that z, xz belongs to the language and yz does not or the opposite, yz belongs to the language and xz does not.

If this happens, we say that x and y are distinguishable by L. This is stated for any two strings, x and y over the alphabet and where L is any language over the alphabet. In this definition, I am not saying L is regular or anything like that. All I am saying is that x and y are two strings, L is a language, when do we say that x and y are distinguishable by the language L.

For instance, suppose x is in the language and y is not in the language. Then automatically, they are distinguishable by the language because one z that works is the empty strings ϵ. So, if x is in the language and y is not in the language, then x appended by the empty string is x itself, which is in the language and y appended by empty string is y itself, which is not in the language.

So, if one of them is in the language and the other one is not in the language, we are automatically done. However, x, y maybe both in the language. But there may be a z such that when you append that z, xz belongs to the language and yz does not belong to the language or the opposite. So, that is the definition distinguishable by the language L.

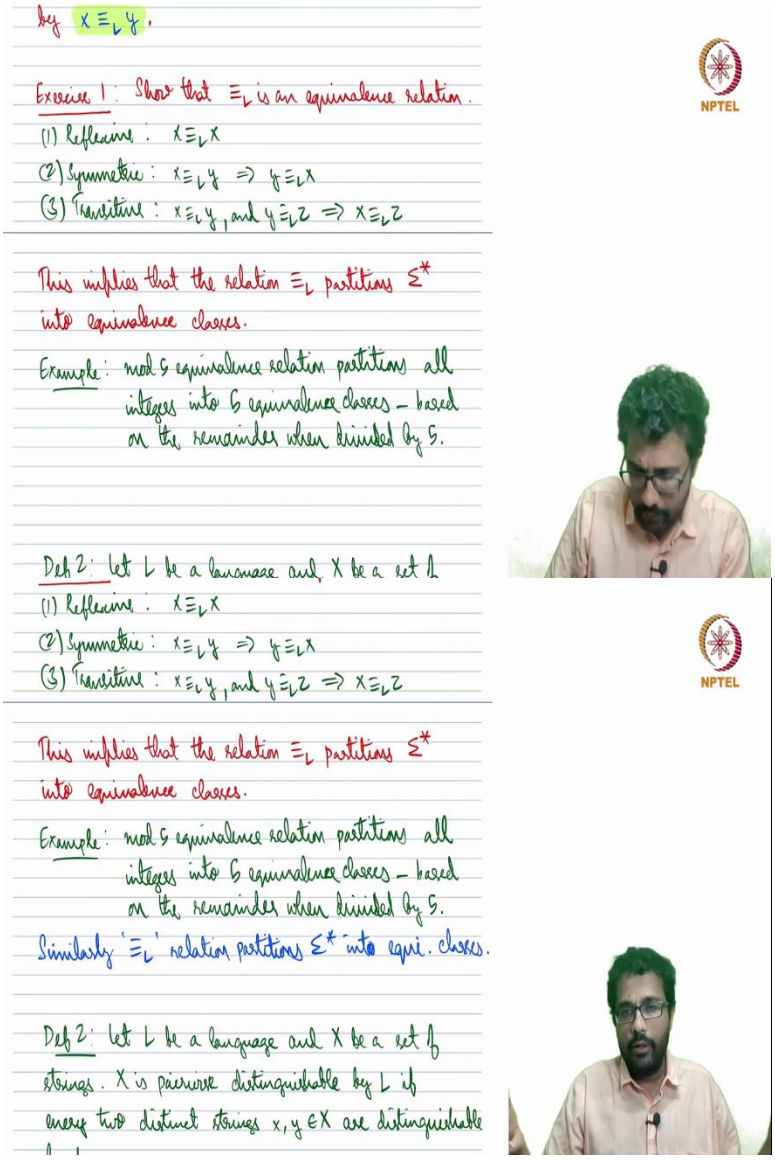
So, x and y are distinguishable if there is some string, which when you append to the strings x and y, one of them belongs to the language the other one does not. If they are not distinguishable, meaning whatever z you append, either both of them belong to the language or both of them do not belong to the language.

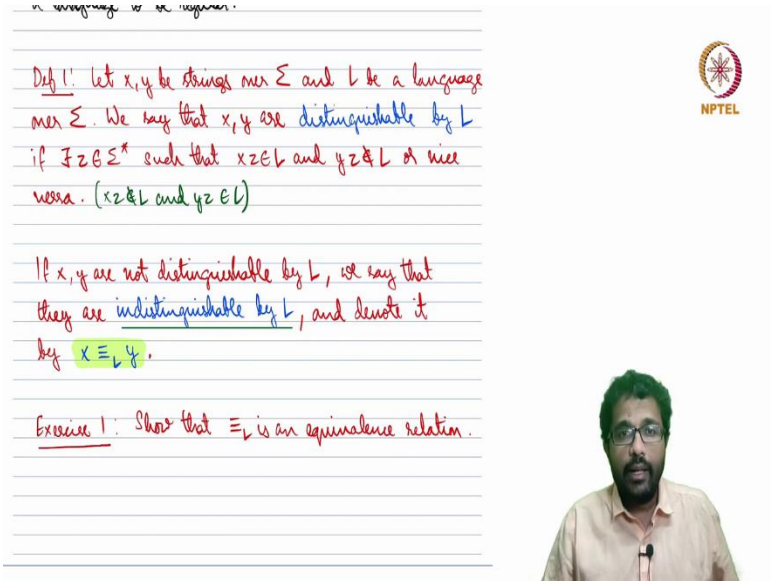
We want to create this opposite situation where one of them belongs and other one does not. If it is not the case, meaning whatever z you append, both of them xz and yz either belong to the language together or do not belong to the language together. When that happens, we say that it is indistinguishable by L.

So, indistinguishable by L and this has a certain notation. The notation is that $x \equiv_L y$. So, equal to symbol has two lines. This has three lines. It is an equivalent symbol subscript L. x equivalent to y subscript L. This can be thought of as an L equivalence. So, this is the symbol for x equivalent to y, or x and y are indistinguishable by L.

So, notice we first define distinguishable if you can append a z such that one of them xz is in the language and yz is not, or the opposite. When the strings are not distinguishable, meaning whatever z you append, either xz and yz belong to the language or xz and yz do not belong to the language, then we say they are indistinguishable by the language. For indistinguishability we have this notation, the equivalence notation. So, the notation is for indistinguishability.

(Refer Slide Time: 06:44)





And the one interesting thing is that being indistinguishable is an equivalence relation. So, what is an equivalence relation? We need to show three things. I will not fully show them but I will explain them. So, 1 is Reflexive, 2 is Symmetric and 3 is Transitive.

So, reflexive means x is equivalent to itself this is true a trivially because x you can just verify this. So, xz is indistinguishable from itself. So, if xz is in the language xz also has to be in the language. Symmetric means, basically it is like commutativity, x equivalent to y implies that y equivalent to x. This is also trivially true because, the way we have defined this there is no it is not like x and y play exactly the same role.

So, even if you change the role of x and y, that is change the order of x and y, it does not really matter so, this also easily follows. Transitivity is if x equivalent to y and y equivalent to z, so, if x and y are indistinguishable, and y and z are indistinguishable, together imply that x and z are indistinguishable. If you want to show equivalence relation, this is the only one, the third one, that takes some work. The other two are almost immediate. So, these are the three conditions for equivalence relation.

So, here a relation meaning, equal to is a relation and here the indistinguishability is a relation. So, once any relation satisfies these three conditions, we say it is an equivalence relation. So, another example of an equivalence relation is, let us say, over the set of all natural numbers, so 0, 1, 2, 3 up to infinity. We say that two numbers are equivalent if they have the same remainder when divided by 5.

So, they share the same remainder when divided by 5. That is, 1 and 6 are equivalent because when they are divided by 5, 1 and 6 leave the same remainder. 6 and 11 are equivalent, 7 and 12 are equivalent, 7 and 22 are equivalent. And one thing that any equivalence relation does is it partitions the entire space into equivalence classes.

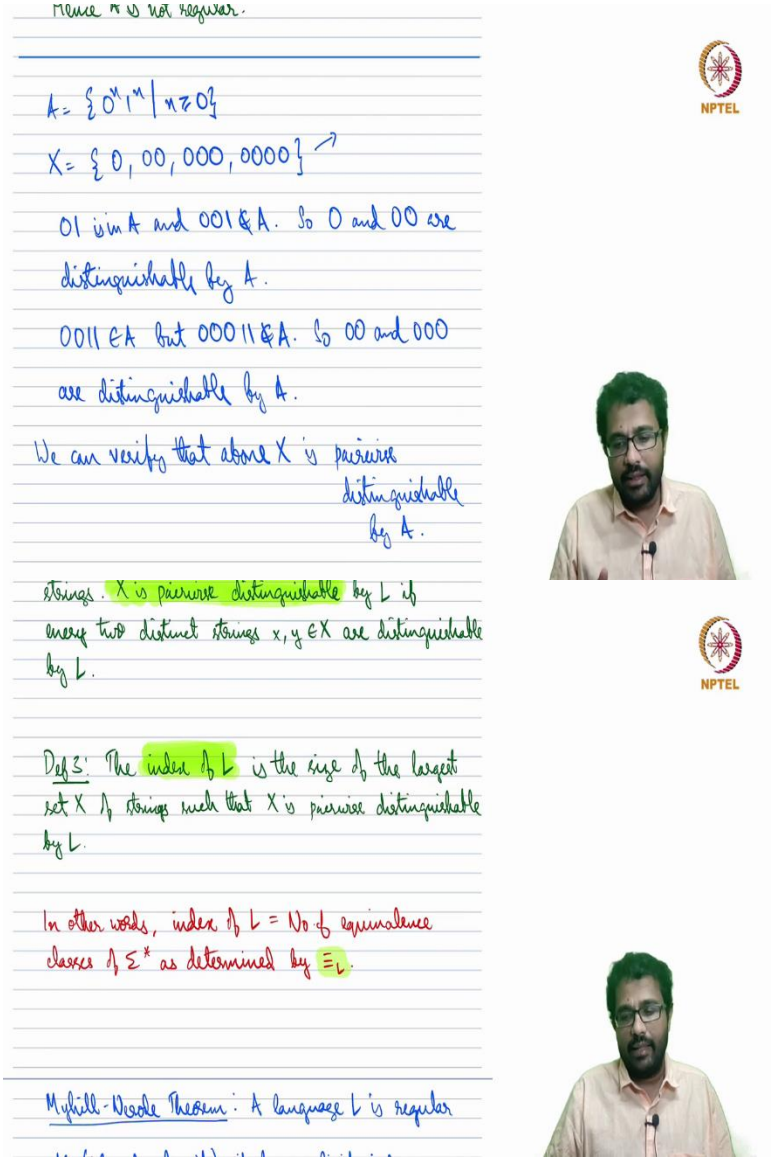
For instance, if you take the 'remainder when divided by 5' equivalence relation, the entire space of natural numbers gets partitioned into 5 classes, the multiples of 5, the numbers that leave a remainder 1 when divided by 5, the numbers that leave a remainder 2 when divided by 5, those who leave a remainder 3, and those who leave a remainder 4. There are five classes. I said natural numbers, but it is also true for integers.

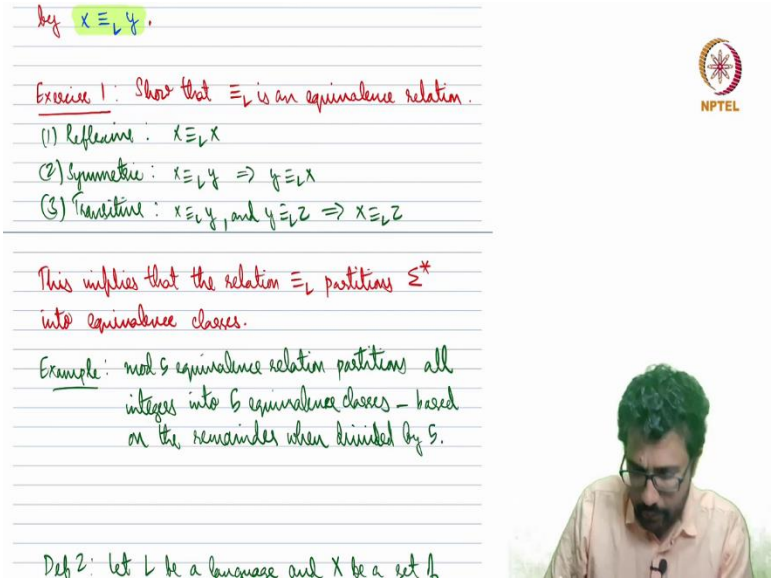
So, for example mod 5 equivalence relation partitions all integers into 5 equivalence classes, those based on the remainder when divided by 5. So, this is an example for an equivalence relation partitioning the integers, partitioning meaning each integer goes into exactly one of these classes. It is not like these classes overlap there, meaning they are disjoint classes and together they divide the entire set of integers or natural numbers.

So, in this case we have indistinguishable by L, as an equivalence relation. And the things that you can compare using this relation are the set of all strings over a certain alphabet, Σ^*. In this case, what happens is this relation partitions Σ^*. So, maybe I will just use another color. Similarly, this relation partitions Σ^* into equivalence classes.

So, when it partitions, you get some number of equivalence classes. If you look at a certain class, anything within the class is indistinguishable, but two different classes are not distinguishable. So, that is what this relation will do. Maybe, I will just give a very quick example of indistinguishable by L. So, for a moment, you can ignore whatever I have written over here.

(Refer Slide Time: 13:15)





But, consider A to be $\{0^n1^n \mid n \geq 0\}$. We know this is not a regular language. And consider this set of strings. Let me call it S. It is $\{0, 00, 000, 0000\}$. I claim that they are all distinguishable by L. So, you take any two. So, for instance, if you take the string 1, $01 \in A$ and $001 \notin A$. So, this means that 0 and 00 are distinguishable by A, because you append 1.

Similarly, $0011 \in A$, but $00011 \notin A$. So, 00 and 000 are distinguishable, just to give you an example of what distinguishable means. Anyway, coming back to what we were saying. So, being not distinguishable is an equivalence relation. And that equivalence relation partitions the entire set of strings into equivalence classes. So, now two more definitions. Suppose L is a language again, I am not insisting that L is regular, and X be some set of strings. We say that X is pairwise distinguishable by L.

So, the definition is pairwise distinguishability: we say that X is pairwise distinguishable by L if any two distinct strings in the set X are distinguishable. For instance, in the example that we just saw, if you take the set, maybe I will just call it X instead of S. Any two strings from the set X, you can see that you can indeed verify that they are distinguishable by the language. So, back by the language A. So, for instance, $01 \in A$ but $001 \notin A$.

So, 0 and 00 are distinguishable. $0011 \in A$ but $00011 \notin A$, so 00 and 000 are also distinguishable and you take any two, so, if you take 0 and 0000, the string 1 distinguishes them. This means that

we can verify that above X is pairwise distinguishable by A. Pairwise distinguishable by A means, in that set, any two strings you take must be distinguishable.

So, this is what pairwise distinguishable by a language means. It is a set. The set is pairwise distinguishable if any two distinct elements of that set are pairwise distinguishable by the language. The third definition is that of the index of a language. So, notice that so far, we have just that L is some language. We have not been saying anything above whether L is regular or not. All these definitions are for a language, for some language.

So, the index of L is the size of the largest set X such that X is pairwise distinguishable. Just to give the same example again, here we had A, which was not a regular language, like 0^n1^n. We have a set X here, which is pairwise distinguishable, because any two distinct elements are distinguishable. So, this set is of size 4.

Now, can you make a bigger set, maybe a superset of this but not necessarily that, that is pairwise distinguishable by A? So, what is the biggest set that you can make that is pairwise distinguishable by A? The size of that set is the index of A. So, here, we know that the index is at least 4 because there is a set of size 4 that is pairwise distinguishable, but it could be 5, it could be 6.

So, that is the definition of index. It is the size of the largest set of strings that is pairwise distinguishable by the language. So, earlier I mentioned the equivalence class thing. So, any equivalence relation partitions the entire set of strings into equivalence classes. So, basically, if there are let us say 10 equivalence classes indistinguishable by the language, this partitions the set of all strings into 10 equivalence classes.

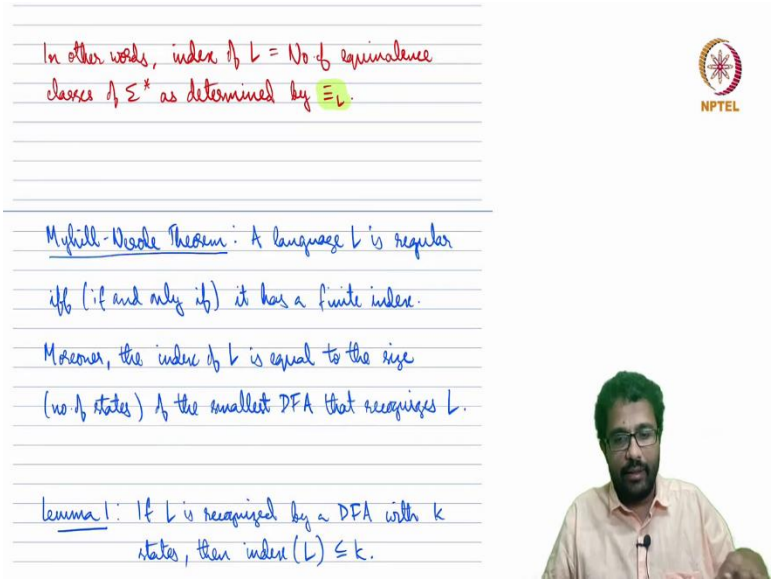
That means you can pick one string from each of the equivalence classes and this is the maximum that you can take. If there are 10 classes you can pick one from each because if you pick 11, then by pigeonhole principle you must be picking two from some class and they will not be distinguishable.

So, the index is actually asking what is the number of equivalence classes? How many equivalence classes? The relation 'indistinguishable by the language L' partitions Σ^* into equivalence classes, then what is the number of equivalence classes that we get? If it partitions Σ^* into 10 equivalence classes, then I can get a pairwise distinguishable set of size 10.

But if it partitions Σ* into 6 equivalence classes then the largest set that I can construct is of 6. From each equivalence class I can pick one representative and that is the best that we can do. For instance, in the case of integers module 5, I can only pick one number that is a multiple of 5, one number with remainder 1, one number with remainder 2 and so on.

I can only pick 5 such representatives. If you pick a sixth number, it will be equivalent or it will have the same remainder as one of the other representatives already chosen. So, index is basically the number of equivalence classes of Σ* for the equivalence relation indistinguishable by *L*. So, how many equivalence classes does it divide Σ* into, that is the index.

(Refer Slide Time: 21:49)



Lemma 1: If L is recognized by a DFA with k states, then index$(L) \leq k$.

Lemma 2: If index$(L) = k < \infty$, then there exists a DFA with k states that recognizes L.

Proof of Myhill Nerode theorem assuming lemmas

($\Rightarrow$) Suppose L is regular. Then there is a DFA that recognizes L. Consider a smallest DFA that recognizes L. Let this DFA be M, and let M have k states. By lemma 1, we have index$(L) \leq k$.

Size of the smallest DFA that recognizes L. Consider a smallest DFA that recognizes L. Let this DFA be M, and let M have k states. By lemma 1, we have index$(L) \leq k$.

$$\text{Index}(L) \leq \text{Size of the smallest DFA that recognizes } L.$$

($\Leftarrow$) Suppose L has finite index, say k. By lemma 2, there exists a DFA with k states that recognizes L. So L is regular.

$$\left.\begin{array}{l}\text{Size of the smallest DFA}\\ \text{that recognizes } L\end{array}\right\} \leq \text{Index}(L)$$

by $x \equiv_L y$.

Exercise 1: Show that $\equiv_L$ is an equivalence relation.

(1) Reflexive: $x \equiv_L x$

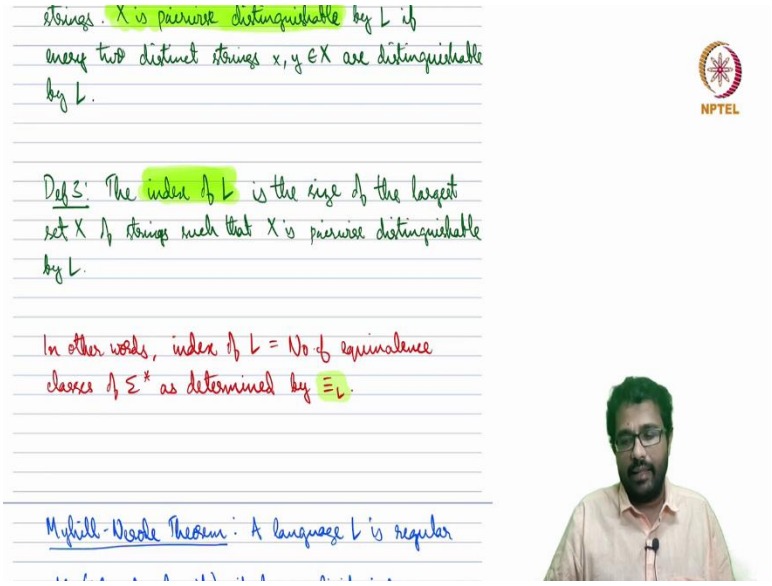
(2) Symmetric: $x \equiv_L y \Rightarrow y \equiv_L x$

(3) Transitive: $x \equiv_L y$, and $y \equiv_L z \Rightarrow x \equiv_L z$

This implies that the relation $\equiv_L$ partitions Σ^* into equivalence classes.

Example: mod 6 equivalence relation partitions all integers into 6 equivalence classes – based on the remainder when divided by 6.

Def 2: Let L be a language and X be a set of



And Myhill-Nerode Theorem now, after all this definition. So, we defined distinguishable by *L*, then we defined indistinguishable by *L*, we said that indistinguishable by *L* is an equivalence relation and then we said that it divides the entire set of strings into equivalence classes. Index is the number of equivalence classes. So, after all this definition, we come to the theorem statement. All it is saying is that a language *L* is regular if and only if it has a finite index.

So, the statement is simple, but then we required some buildup. A language *L* is regular if and only if it has a finite index, meaning, regular implies finite index and finite index implies regular. Moreover, if the language is regular then it has a finite index. Let us say the index is 10 then we can construct a DFA of size 10, size meaning the number of states 10, that recognizes the language. So, index is also the size of the smallest DFA that recognizes *L*. So, size of the smallest or size of a smallest, because the DFA that recognizes *L* may not be unique, let us say a smallest DFA is recognizes *L*.

So, this is the theorem, a language is regular if and only if it has finite index and if it is regular the index is also the size of the smallest DFA that recognizes the language. Basically, like any of these theorems with if and only if, we have to show both directions and both directions are split into lemma 1 and lemma 2.

Lemma 1 says that if *L* is recognized by a DFA with k states, then the index is at most k. If there is a DFA with k states that recognizes the language then the index is upper bounded by k. Lemma 2 states that if the index is some number k, where k is a finite number, then there is a DFA with k

states that recognizes L. First one says that whatever be the number of states, index is at most that. Then it says that if the index is finite, then there is a DFA with that many states. So, putting these two together, we get the theorem. Lemma 1 and Lemma 2 together imply the theorem. Let us quickly see how so. Suppose L is regular.

We have to show that if L is regular, it has a finite index and if it has a finite index, it is regular. Then the second statement, the size of the smallest DFA that recognizes L is equal to the index. Suppose L is regular, which means there is a DFA. Now consider the smallest DFA. Suppose it has 10 states. Now, by Lemma 1, it says that the index is at most 10.

So, if the DFA is M and if M has k states, lemma 1 implies that index is at most k. So, that is what we want to show, if it is regular, then the index is upper bounded by some finite number. And further we have that index is the size of the smallest DFA that recognizes L. Index is upper bounded by the smallest DFA. We got that index is less than or equal to the size of the smallest DFA. It could be smaller. But at least we know that index is not more than that.

Now, the second claim, the other direction, is the opposite. If it has a finite index, we want to show it is regular. Suppose, it has a finite index, let us say k. Lemma 2 says that if it has a finite index, then there is a DFA with k states. So, then there is a DFA with k states, which means L is regular and further we have that the size of the smallest DFA is at most k, which is the index. I am trying to draw a box. So, the size of the smallest DFA that recognizes L is at most the index.

And now, let us compare these two. We have shown that if L is regular, lemma 1 implies that the index is at most the size of the DFA that recognizes L. Secondly, if L has a finite index then lemma 2 says that there is a DFA with number of states equal to the index that recognizes L. So, the size of the smallest DFA cannot be more than that.

Now, comparing these two boxes, it follows that index is equal to the size of the smallest DFA that recognizes L which is the second statement of the theorem. So, even this has been shown. That is how we show the proof of Myhill-Nerode Theorem. Using the lemma, so, I will not show the proofs of the lemma themselves, I am just using the lemmas, but then the lemma itself looks like the statement of the theorem itself.

So, now that we have broken down the theorem statement into these two lemmas. Perhaps, we will see the proof of the lemmas and considering the time, I think I will split the video into lecture 15, where I will show the proof of the lemmas. So, what have you seen here? We saw the definitions that go into the Myhill-Nerode Theorem. The definitions are for x and y. When are x and y distinguishable by a language L, when are they indistinguishable by a language?

So, these are the opposite. Then we saw that indistinguishability is an equivalence relation and this equivalence relation divides the set of all strings into equivalence classes. The index of a language is the number of classes that the indistinguishability relation divides Σ* into. And the statement of the theorem is that a language L is regular if and only if the index of L is finite.

And second, the index of L is also equal to the number of states of a smallest DFA that recognizes that language. And the proof of the Myhill-Nerode Theorem is basically by two lemmas that prove either direction. What I said is that we will complete the proofs in the next lecture, lecture number 15. So, that is it from me for lecture number 14. See you in lecture number 15 where I will complete the proof. Thank you.