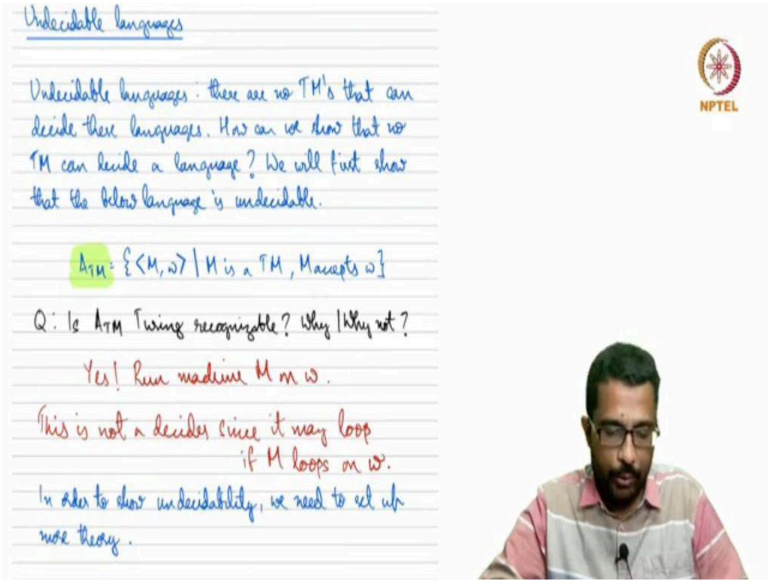
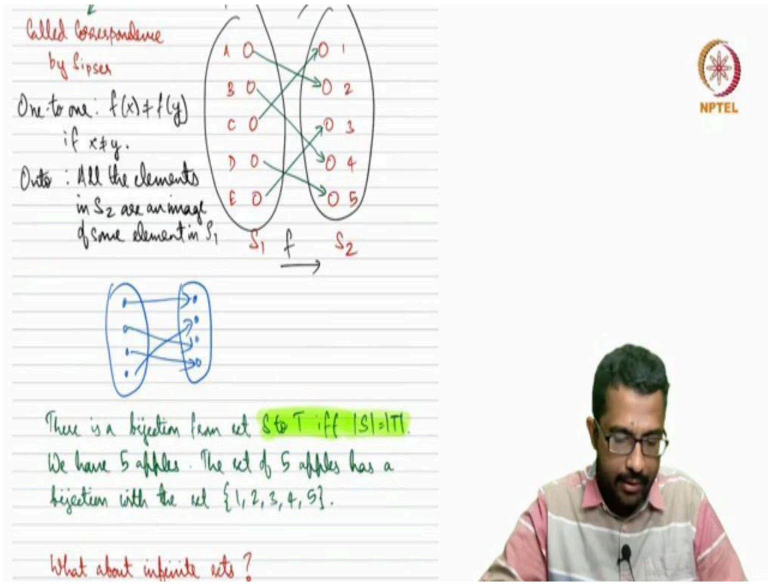
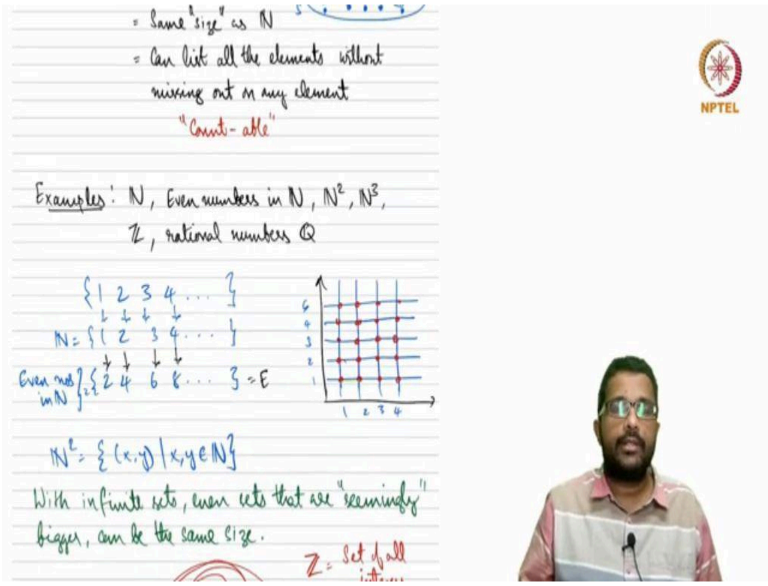
Theory of Computation
Professor Subrahmanyam Kalyanasundaram
Department of Computer Science and Engineering
Indian Institute of Technology, Hyderabad
Lecture 41
Proof of Existence of Undecidable Languages

(Refer Slide Time: 00:16)

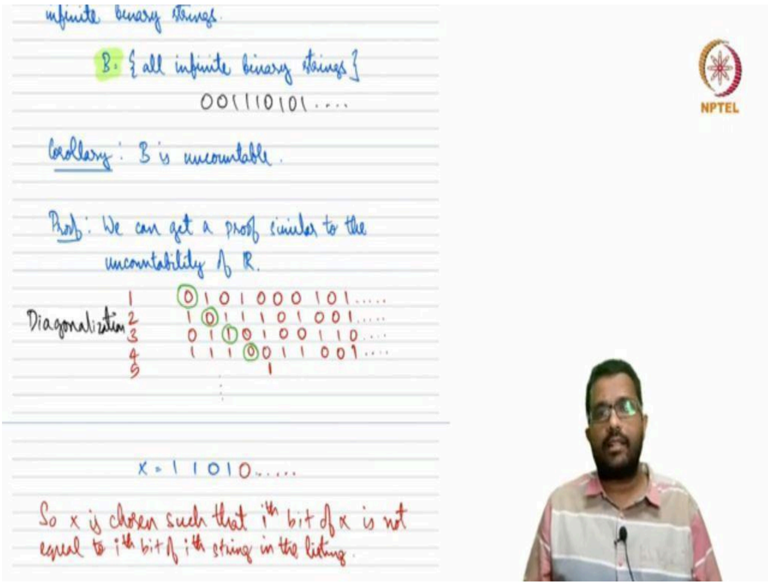


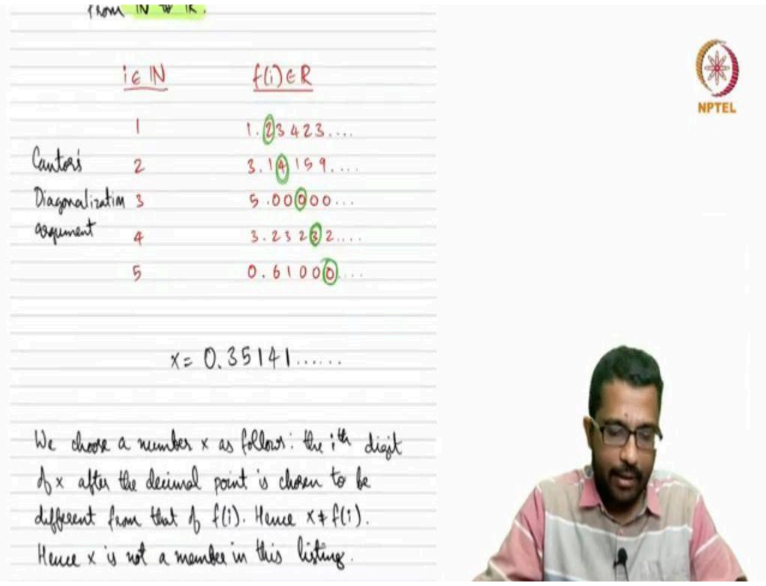




Hello and welcome to lecture 36 of the course Theory of Computation. In lecture 35, we started building towards the theory of undecidable languages. We defined bijection as the correspondence in Sipser. We defined countable numbers, countable sets; we saw that N, Q, the set of all natural numbers, the set of all integers, and a set of all rational numbers are countable. We saw that R the set of all real numbers is uncountable. Now, we will see further, we will move further ahead. We will see more sets that are countable and uncountable, and as we build towards the theory of undecidable languages.

(Refer Slide Time: 01:11)





So, just like we saw R by using this diagonalization argument, we will see another set which is uncountable. So, let us set B be the set of all infinite binary strings; in binary strings, infinitely long binary strings. So, I mean strings like 0110, some something, some in; so each element in the set is an infinite binary string. So, B is a set of all infinite binary strings, so not a finite binary string; so for instance, 00 or 1, or 101, these strings are not there in B. So, the elements of B are infinitely long binary strings; so this B is uncountable. Why is that? We can pretty much use the same argument as the diagonalization argument used in the uncountability of R.

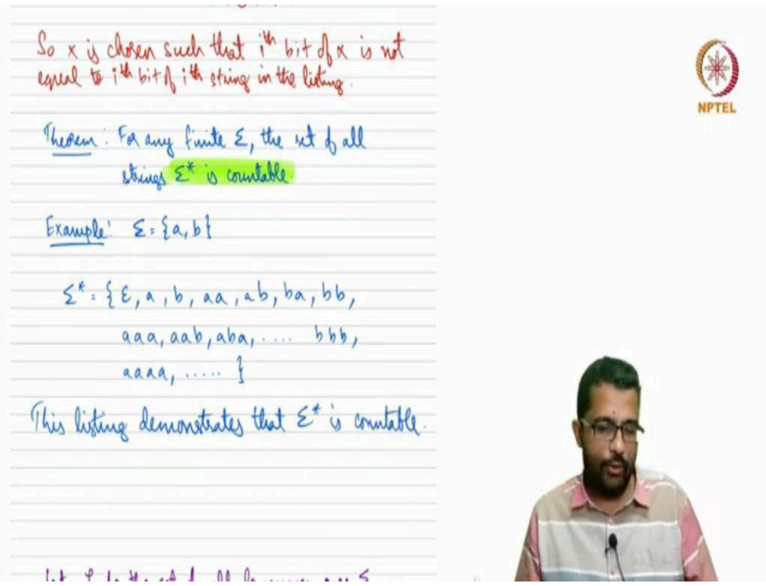
So suppose, we suppose it is indeed countable; so which means there is a listing of the elements of B. So, let us say the first number, in the first number the listing was 0101000101 something; second number in the listing was 1011101001; third number was 0110100110; fourth number is 1110011001, something like this. Suppose it is some listing that is available or some ordering that is possible of all the numbers in B. So, each are not numbers in B, all the elements of B; so, each element is a infinitely long binary string. Now, just like in the uncountability of R, we produce an x which is not, which does not feature the listing. Here also we can use the same technique to produce such an x.

So, look at the first element or first bit of the first string it is 0, second bit of the second string 0, third bit of the third string 1, fourth bit of the fourth string is 0; so it is 0010. So, I take the opposite, so I define x to be 1101, and so on. So, if the fifth bit of the fifth number was 1, then the fifth bit of x would be 0. So, I make, I choose x in such a way that the fifth bit of x or the i^{th} bit of x, differs from the i^{th} bit of the i^{th} string in this listing. So, x is chosen, such that i^{th}

bit of x is not equal to i^{th} bit of i^{th} string in the listing; the rest are not, rest the same as in the uncountability of R.

So, this means that x is different from all the numbers in the listing. So, x is different from the 50^{th} number 50^{th} string in the listing, because the 50^{th} bit of x is different from the 50^{th} bit of the 50^{th} string in the listing; and so on for 100 string or 200 string and so on. So, x is different from all the strings in the listing, which means the listing was not proper to begin with. How come x is different from all of them? That means x does not get captured at all; which means, the listing is missing numbers. So for instance, x is such a number. So, this means the set B of all infinitely long binary strings, this set is uncountable. So, the set of all infinite binary strings is uncountable. Now, let us see another set which is countable.

(Refer Slide Time: 05:47)

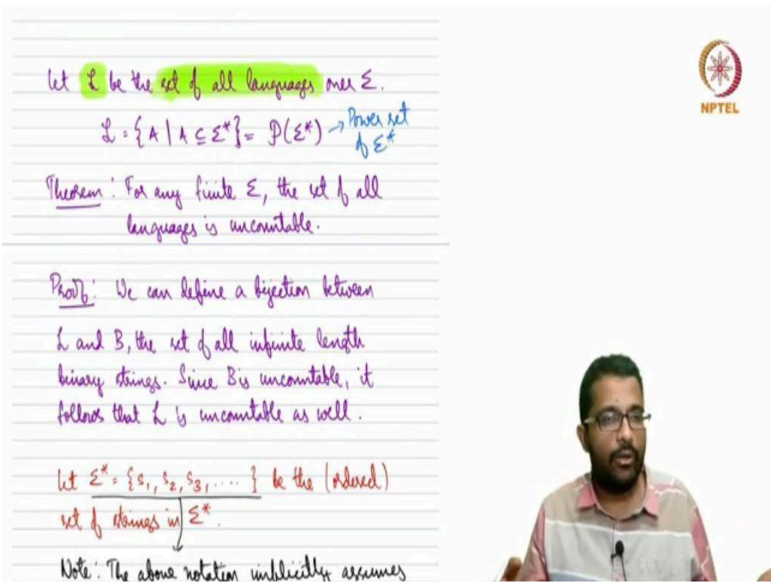


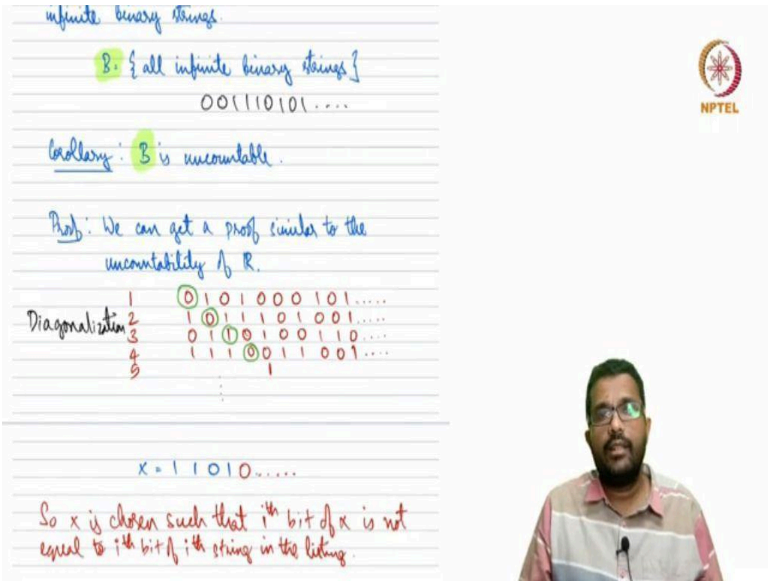
So, consider a finite Σ. So, all the alphabets that we have seen. We have seen many languages in the like while seeing DFAs, NFAs, Turing machines et-cetera; all of these languages were built on some alphabet. So, it could be the binary alphabet, it could be the English alphabet, it could be the decimal alphabet; all of these were finite. So, let us take a finite Σ, finite alphabet, then the set of all strings. So, given any finite Σ, the set of strings will be infinite; the set of all, it could have because you could have infinitely long strings also. Or you could have finite strings, but of infinitely many possible lengths.

So, this theorem says that the set of all strings is countable, not uncountable, countable. So, this I will not, it is easy to just give the listing. So, suppose Σ is a b, a, b it is binary; but not 0 1, but a, b. So the listing is very simple. If we first write an epsilon which is an empty string, which is a string of length zero, then write a, b. So, a and b are strings of length 1, which are only single symbols. Then, we write the four strings of length 2, aa, ab, ba, and bb; then we write the strings of length 3. So, if you think about it, there are 8 of them, aaa, aab, aba and so on up to bbb. Then, we write strings of length 4, then we write strings of length 5 and so on. So, it should be evident that all the possible strings are captured in this listing.

So, if you ask me for some long string ab, bb, aa or something, then although it may be difficult; but I can, I can tell you that this string will appear as a number, let us say 526 in this listing, some numbers. I do not know, I did not calculate it; but it should be possible to tell where that number appears, where that string appears. So in other words, every string is guaranteed to be captured in this listing. So, this listing demonstrates that a Σ^* is countable; so we are just directly giving the listing itself. So Σ^* is countable. Now given, now let us move on to the next thing, next set.

(Refer Slide Time: 08:26)



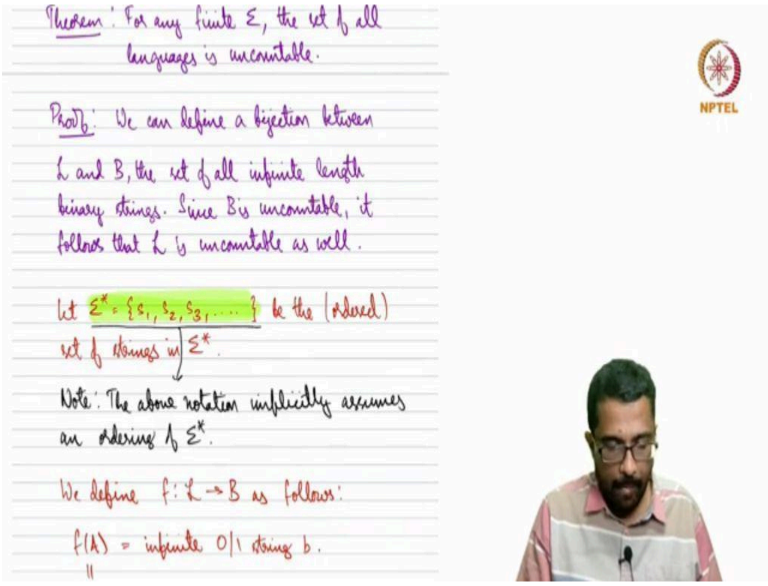


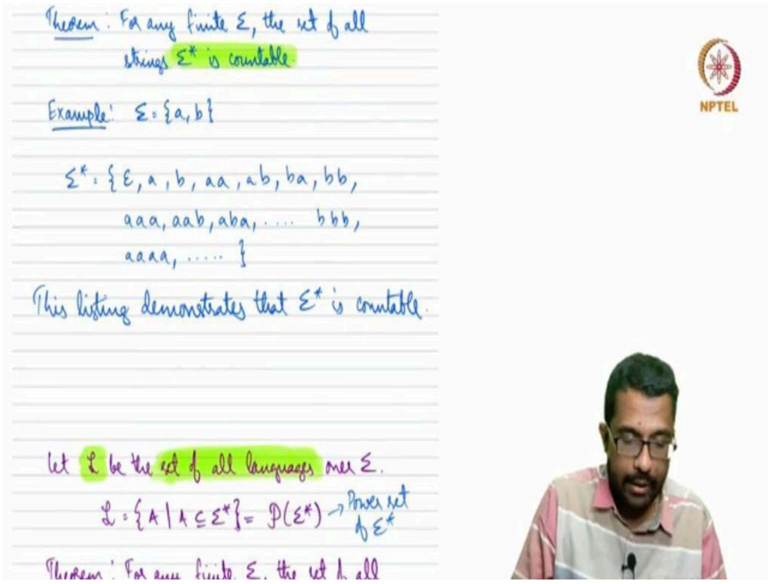
So given Σ, we talked about Σ^*; Σ*is a set of all strings. Now, let me talk about the set of all languages, the set of all languages over Σ. So, what is the language? A language is a subset of* Σ^*. So, now I am talking about a set of all segments subsets of Σ^*; so L is a set of all languages over Σ. So, L is defined to be the set of all subsets of Σ^*; because a language is merely a subset of Σ^*. Or in other words, the power set is a power set of; so the set of all subsets is called the power set. So, L is a set of all languages of over Σ which is the set of all subsets of Σ^*. The theorem states that for any finite Σ, the set of all languages is uncountable.

So, Σ^* itself is countable, the set of all strings is countable; but the set of all languages is uncountable. So, how we will show this is by defining a bijection between the set of all languages and the set B, sorry. So, we showed that B the set of all infinite binary strings was uncountable. What we will do now is to demonstrate a bijection between the set of all languages and the set B. Now, B is uncountable, and if I can have a mapping or bijection between L and B; the set of all languages and B, that means this also has to be uncountable. It cannot be that L can be countable and B can be uncountable and then we can have a bijection.

Because that would imply that B also is countable, which you know is not the case. So, we will show the bijection. So, the bijection automatically implies that since B is uncountable, it implies that L is uncountable, let us see how.

(Refer Slide Time: 10:57)



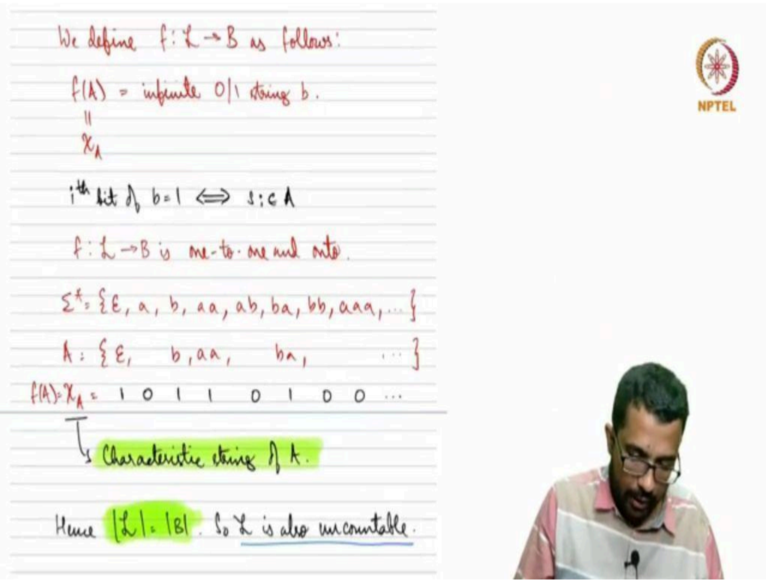


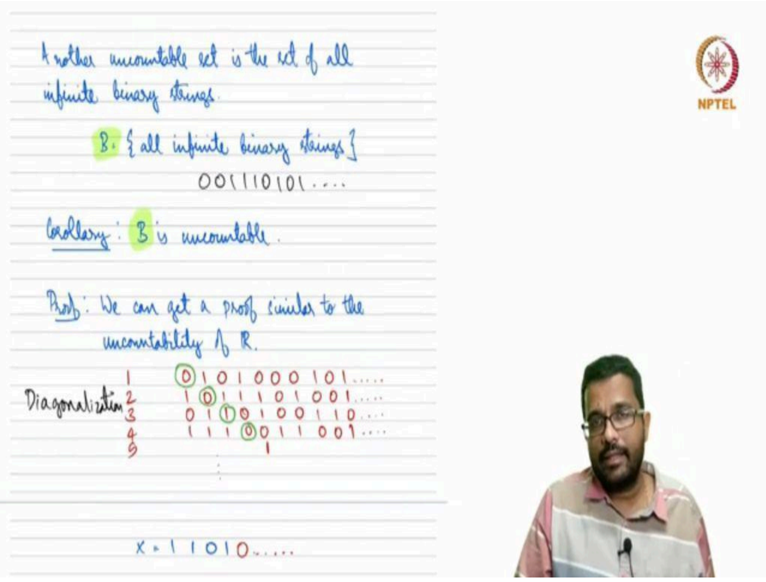
So, let Σ^* be the set of all strings of over Σ, Σ^* that of all strings over Σ. So, let me let Σ^* be represented like this s1, s2, s3, meaning the first string, second string, third string and so on; so notice this. So, even though this may seem like a innocuous or simple notation, actually, this actually carries a meaning; and this actually the way the fact that I can write the Σ^* as the first string comma second string comma third string, itself means that the set of strings in Σ^* is countable. So, this notation, what is it telling us? It is telling us that there is a first string, then there is a second string, then there is a third string and so on.

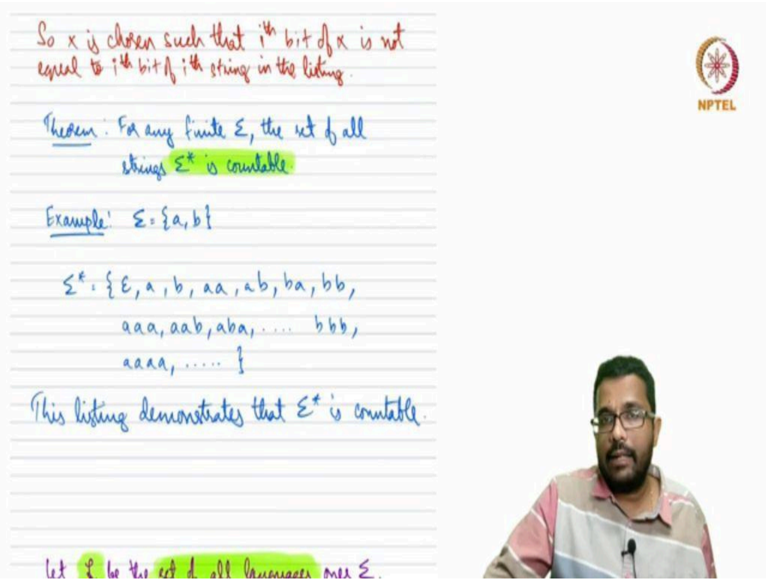
So, it implicitly assumes that I am listing the strings of the Σ^* as the first string, second string, third string, which is kind of saying that the set is countable. So, we are implicitly using the fact that it is countable; we know that Σ^* is countable, so it is okay. But, I am just kind of alerting you to the assumptions just inherent in this notation. Sometimes you just have a notation and that notation itself, just because we are writing in a certain way that itself carries an assumption with it. So, that is what I want to kind of highlight here. Just the fact that I am writing it as first string, second string, third string, itself assumes that Σ^* is countable.

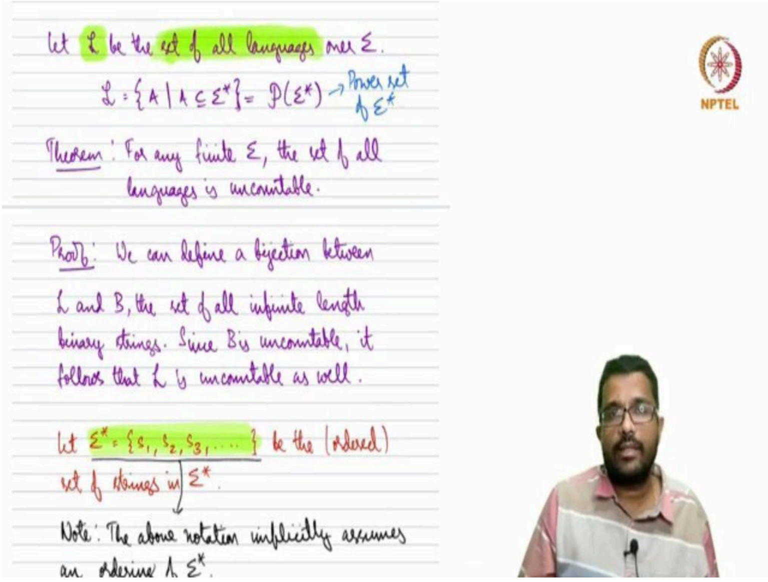
We know Σ^* is countable, so it is fine; this assumption is fine, but one has to be careful in general. So, now let me define the mapping or the bijection from L to B. So, this bijection is going to be like this.

(Refer Slide Time: 12:55)




Another uncountable set is the set of all infinite binary strings.
B = {all infinite binary strings}
001110101....
Corollary: B is uncountable.
Proof: We can get a proof similar to the uncountability of R.
Diagonalization
1 0 1 0 1 0 0 0 1 0 1.....
2 1 0 1 1 1 0 1 0 0 1....
3 0 1 1 0 1 0 0 1 1 0....
4 1 1 1 0 0 1 1 0 0 1....
5
x = 1 1 0 1 0.....
NPTEL


So x is chosen such that i^th bit of x is not equal to i^th bit of i^th string in the listing.
Theorem: For any finite Σ, the set of all strings Σ* is countable.
Example: Σ = {a, b}
Σ* = {ε, a, b, aa, ab, ba, bb,
aaa, aab, aba, ... bbb,
aaaa, ... }
This listing demonstrates that Σ* is countable.
Let L be the set of all languages over Σ.
NPTEL



So, suppose so Σ^*, let us take a two symbol alphabet, a and b. So, let me order these elements of Σ^* like here; so I have the empty string a, b, aa, ab, ba. So, first strings of length zero, then strings of length one, then strings of length two, then strings of length three and so on. So, how will I map? So, the point is that we are going to give a mapping from L to B. Meaning we are for any language, L is a set of languages given any language, B is a set of infinite binary strings. Given any language, we are going to map it to an infinite binary string. So, let us say we give this language; so now, I can, so let a be this.

So, I write all the strings of length zero in a, which is only the empty string; then I write all the strings of length one. So, in a is not in A, small a is not in A; but small b is in A. Then, I write all the strings of length two, which is aa and ba; ab is not in A, bb is not in A. then, I write the strings of all the strings of length three; so aaa is not A, perhaps the next string is an A or not. So, this is how I will write the language; now this itself gives me a binary string. So the way is this, so in the ordering of Σ^*; so the first thing was empty, empty string, and A contains the empty string.

So, now we put the first bit to be 1, because empty string is in A. The second string in Σ^* was small a, which is not there in the language A; so, the second bit of the string is zero. Small b is a third number which is in A, so the third bit is 1. Fourth bit is one because a is in the language; fifth bit is zero, because ab is not in the language, six bit is one, seventh and eight bits are zero, and so on. So, now notice that this string 10110100 indicates this language A, so this is an infinite string; so, this will extend up to infinity. So, this will tell us which strings are in, which strings of Σ^* are in A, and which strings are not in A.

So, this infinitely long binary string corresponds to one language and only one language which is A; and a corresponds to only this binary string. So, this is a bijection from the set of all languages to the set of all infinite binary strings. So, sometimes this kind of representation is called the characteristic string of A, because I have all the strings in Σ^*; and I am telling which strings of Σ^* are in A and which things are not in A. So, 1 indicates that string is in A, 0 indicates that string is not in A. So, this χ, sometimes it is called χ_A. So, now, this mapping of looking at each string whether it is in a or not, gives us a mapping from the set of all languages to the set of all infinite binary strings.

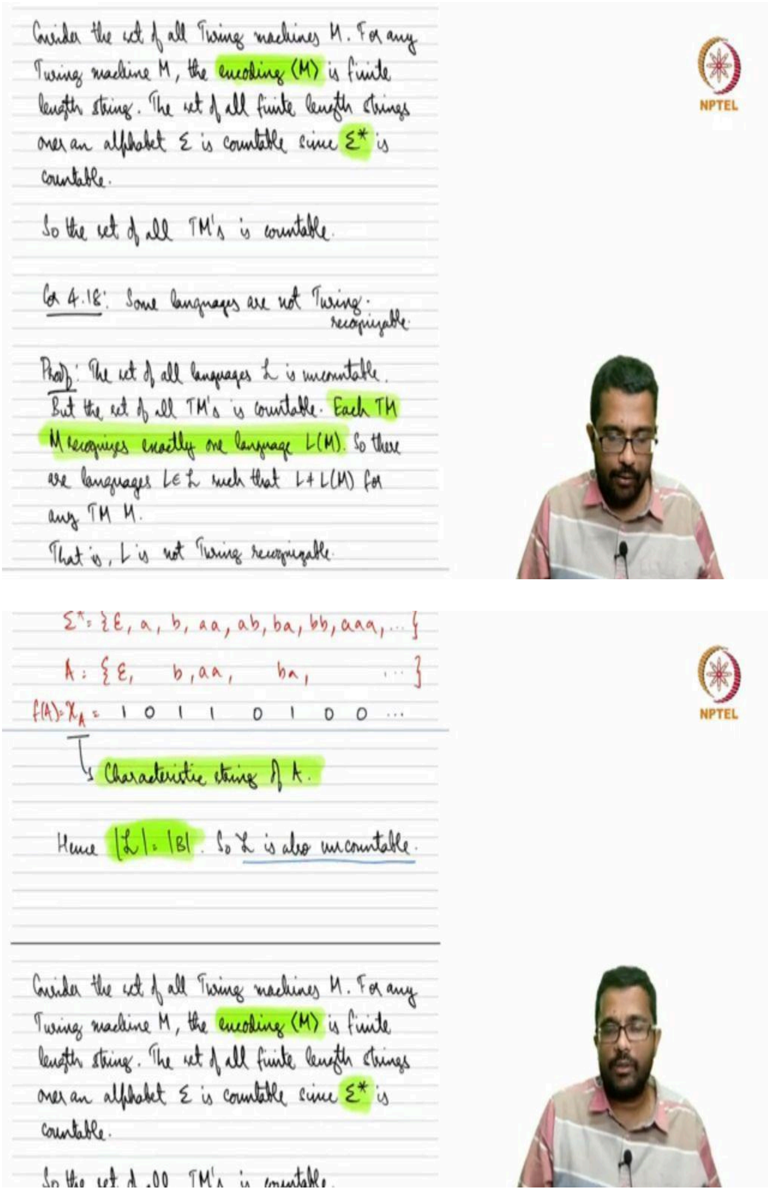
So, given a language, I can give you the infinite binary string; given an infinite binary string, I can tell you which language it corresponds to. Hence, this is a bijection. This means, the size of L which is the set of all languages and the sight of all, size of B which is the set of all binary strings is the same. This means that since B we have already seen that B is uncountable; this implies that L is also uncountable. So, that is the proof that the set of all languages is uncountable. So, just to recap, we showed that the set of all infinite binary strings is uncountable. For a finite alphabet, the set of all strings is countable; but, the power set of all strings which is a set of all languages is not countable.

Now, this is going to be used like the fact that the set of all languages is uncountable is going to be used to show that there are undecidable languages. So, we are not going to show that a specific language is undecidable, but this will imply that languages are undecidable; or there are undecidable languages. The proof is fairly simple, but it is not constructive; or in the sense that it is not directly going to tell us which language is undecidable. But it is really simple. So, it's like this. Suppose, you have 10 mangoes and there are 12 people, or 11 people; then there has to be like you distribute the mangoes, there has to be at least one person who will not get mangoes.

So, that mango is not mapped to anybody; so, it is something like that. So, the point is this, the proof is going to be this. We have seen that the set of all languages uncountable, we will say that the set of Turing machines is countable. But, each Turing machine recognizes only one language; so there is exactly a set of strings that that Turing machine accepts. So, given a Turing machine, there is a language corresponding to it; the language recognized by it. But, there are only countably many Turing machines, but there are uncountably many languages; which means, uncountable is far more than countable.

So, which means there are many languages not even, not just one, there are many languages that are not recognized by Turing machines. So, this shows that there are languages that are not Turing recognizable, which means there are languages that are not decidable; because being decidable first requires you to be Turing recognizable. So, let us go over the proof very quickly; anyway, I have told the high level picture.

(Refer Slide Time: 19:06)

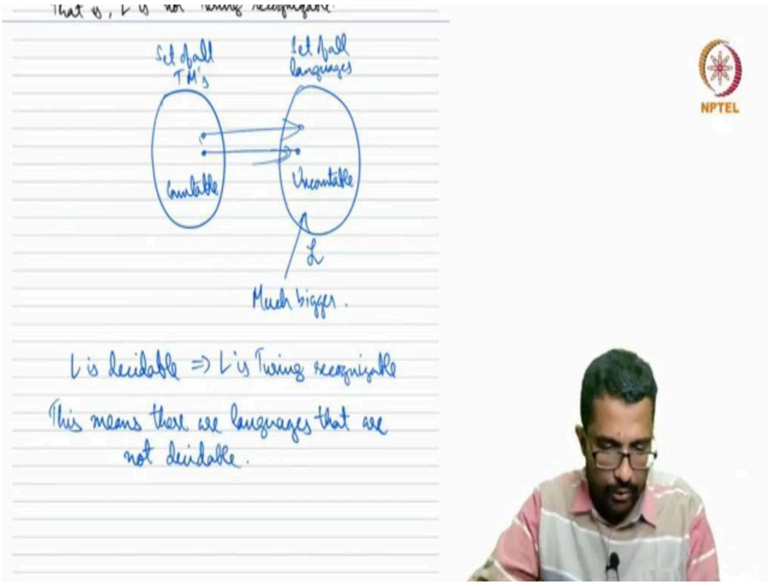


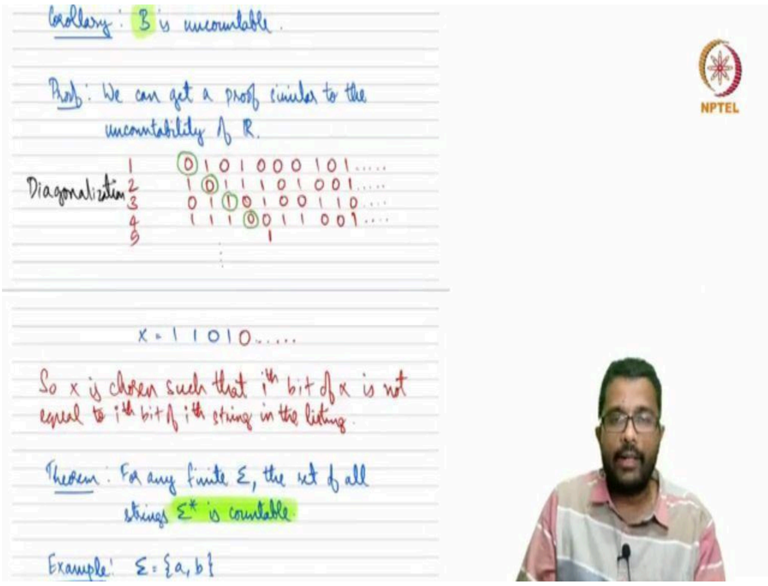
So, consider all Turing machines. So, given any Turing machine, as we saw in the previous week, or in the last lecture of chapter 3; any Turing machine has an encoding. So, given a Turing machine, we can list down all the alphabet, at least all the rules in zeros and ones; and it gives a finite length string. So, suppose we encode the Turing machine into a finitely long string, finite string. Because, it is a fixed set of rules, a fixed set of alphabet, everything is fixed; so, the description is going to be finite. Now, since the description is going to be finite, every Turing machine has a finitely long description.

Over any alphabet Σ, we have the set of all finitely long strings is actually a Σ^*. So, all the encoding of all the Turing machines is going to be a member of Σ^*. We know Σ^* is countable; so the set of all Turing machines is also countable. So, not it is not necessary that all this, all the strings correspond to all the finite strings correspond to encodings. But, whatever the encodings are, they are finitely long strings; so it is going to be a subset of a Σ^*. So, since Σ^* is countable, the set of all Turing machines is also going to be countable.

But, each Turing Machine recognizes exactly the set in exactly one language; so give, you made a Turing machine. There will be some strings that it accepts, some strings that it does not accept. So, what I am saying is that every Turing machine M recognizes exactly one language. So, this means that there are languages, but we know that the set of all languages are countable. That is what we saw just now, just before coming here. So, the set of all languages uncountable and the set of all Turing machines is countable; so it is like this just to. And the set of all uncountable sets is much bigger than the countable set.

(Refer Slide Time: 21:28)





Theorem: For any finite Σ, the set of all strings Σ^* is countable.

Example: $\Sigma = \{a, b\}$

$\Sigma^* = \{\varepsilon, a, b, aa, ab, ba, bb, aaa, aab, aba, \ldots\, bbb, aaaa, \ldots\}$

This listing demonstrates that Σ^* is countable.

Let $\mathcal{L}$ be the set of all languages over Σ.

$\mathcal{L} = \{A \mid A \subseteq \Sigma^*\} = \mathcal{P}(\Sigma^*)$ → Power set of Σ^*

$f(A)$ = infinite 0/1 string b.

$\;\;\| \\ \chi_A$

i^{th} bit of $b = 1 \iff s_i \in A$

$f : \mathcal{L} \to B$ is one-to-one and onto.

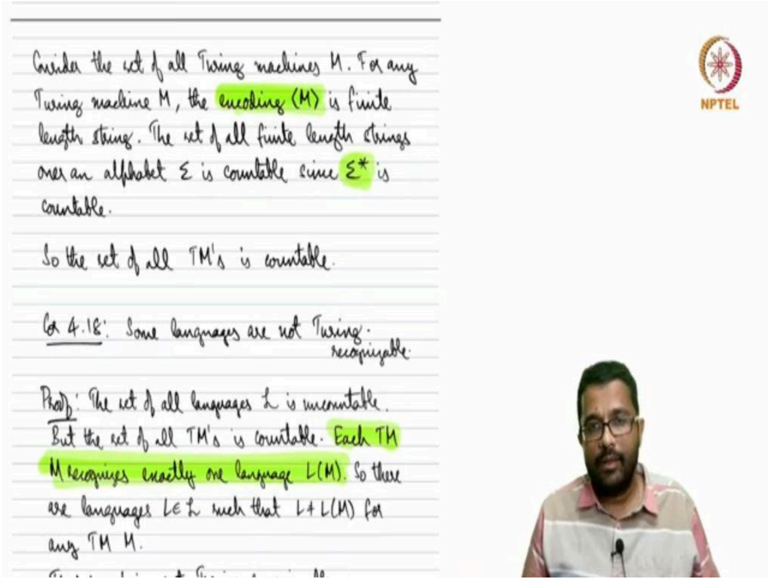
$\Sigma^* = \{\varepsilon, a, b, aa, ab, ba, bb, aaa, \ldots\}$

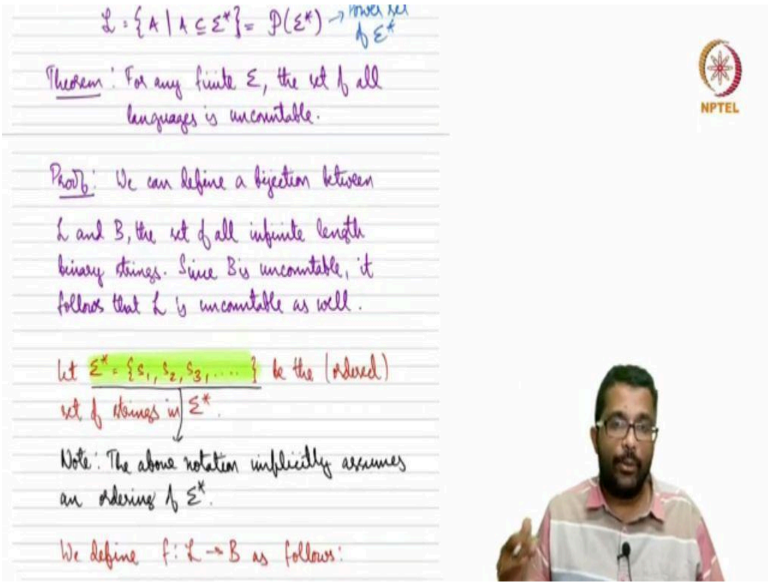
$A = \{\varepsilon, \;\; b, aa, \;\; ba, \;\; \ldots\}$

$f(A) = \chi_A = 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0 \ldots$

↳ Characteristic string of A.

Hence $|\mathcal{L}| = |B|$. So $\mathcal{L}$ is also uncountable.





So, this is the set of all Turing machines, this is the set of all languages which we called, which we call L; so this is uncountable, this is countable. So, so, you cannot have a bijection. In fact, even though I drew it kind of similar size, this is much bigger, this is much bigger. So, which means there are many languages; in fact, there are many many, there are a huge number of languages. In fact, most of the languages will be not recognizable by Turing machines. So, given any Turing machine, it corresponds to only one language. And hence, you can have a mapping from the set of all Turing machines to the set of all languages; but many languages will not be, it will not be a bijection.

So, many languages will not have Turing machines corresponding to it. Which means it is the same as saying that there are many languages for which there is no Turing machine that recognizes it. This means that there are languages that are not Turing recognizable. So, if L is decidable or L is decidable, implies that L is Turing recognizable. So, for a language to be decidable, we want it to be recognizable; and halt on every input. So, if it is decidable, it is certainly recognizable. This means that, since there are languages that are not Turing recognizable; that means there are languages that are not decidable.

This means there are languages that are not decidable. So, and that completes all the things that I want to say in this lecture, lecture number 36. So, what we said is that the set of all binary infinite binary strings is uncountable. Set of all finitely long strings over any alphabet is countable, the set of all languages uncountable. And because of a mapping between the set of all infinitely long binary strings, but the set of all Turing machines is countable. But, every Turing Machine corresponds to a language, but the number of languages is uncountable; which means there are way more languages than the set of Turing machines.

Which means there are many languages that are not recognized by a Turing Machine; which means there are languages that are not recognizable, which means there are languages that are not decidable. So, it is an interesting way of proving it. So, we said that there are languages that are not decidable or languages that are not Turing recognizable; but the proof did not produce any specific language. It just said that there are such languages, just by this counting or bijection argument. You have this set that is countable, this set is uncountable. So, there are way more elements in the uncountable set as compared to the countable set.

So, you cannot have a mapping. So, this means there are languages that are not Turing recognizable; hence, there are languages that are not decidable. So, we saw that B the set of all infinitely long binary strings is uncountable. Set of all finite strings over a language, over an alphabet is countable. Set of all languages is uncountable; the set of all Turing machines is countable. And hence there are languages that are not Turing recognizable; and hence, there are languages that are not decidable. And that completes what I had to say in this lecture, lecture number 36. In the next lecture, we have built all these theories. In the next lecture, we will actually take a specific language which is ATM and then show that this language is undecidable.

So, so far, the argument is kind of based on this set being small, the set is big. So, there are elements in the big set that cannot be mapped to the small sets. So now, we will actually take a specific language and show that it is undecidable; meaning no Turing machine can be a decider for this language. So, that we will see in the next lecture, so see you there. Thank you.