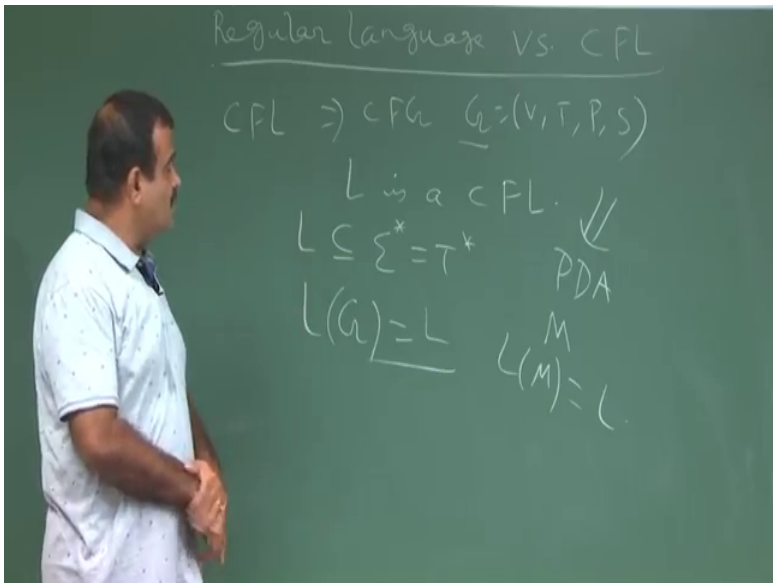
Introduction to Automata, Languages and Computation
Prof. Sourav Mukhopadhyay
Department of Mathematics
Indian Institute of Technology, Kharagpur

Lecture – 56
Relation between Regular Language and CFL

(Refer Slide Time: 00:24)



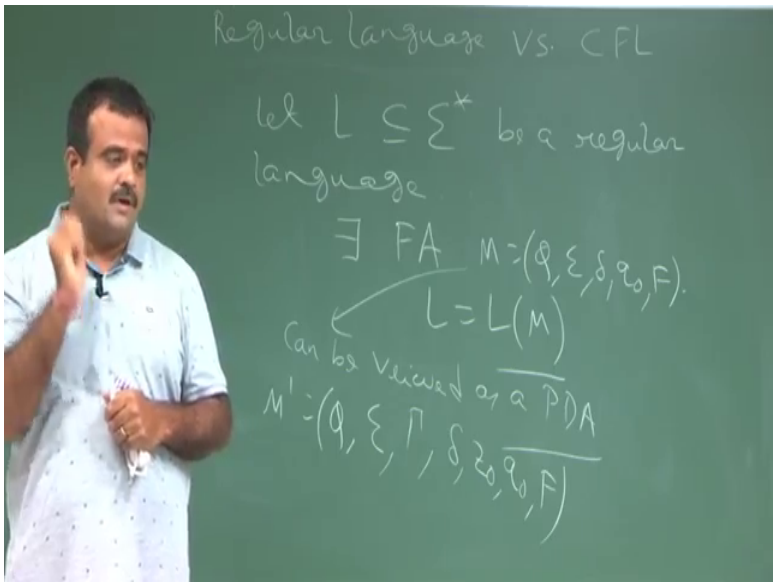
So we are talking about the context free language for which we have corresponding a context free grammar G which is V, T, P, S such that it is accepting the; it is generating the language. So, if L is a CFL Context Free Language then, we called L to be CFL; L is coming from sigma star which is nothing but T; star T sigma over here T is the set of input alphabet or in the grammar sense we call T is the terminals.

So, it is a collection of string of terminals or string of input alphabet we can see that in the terms of the automata so ok. So, when we say this is a CFL; Context Free Language if we have a G, grammar which is generating this language. Then we call this is a context free language if such a grammar exist. And we have seen for every context free grammar, we have it will correspond they are; we have a corresponding PDA push down automata.

So, push down automata is to accept the; which is called context free language. So, for this we have a PDA M which accept this; that means, the language of this, language there are two way of we can define the language accepted by PDA; either in terms of the final state or in terms of the empty stack; so this will be L ok. Now then we have this is a

context free language, now we will discuss what is the, I mean is the; what is the difference or what is the relationship between the regular language under context free language? Ok.

(Refer Slide Time: 02:26)



So, let us take a language which is regular; let L be a regular language over a input alphabet sigma or we can refer this to be a T; T is the set of terminals. But we usually use the sigma for the automata when we discuss the automata. So, this is regular means what? The regular means there is just a finite automata it could be DFA, it could be NFA, it could be epsilon NFA all are equivalent. So, we can take any finite automata M which is Q, sigma, delta, q 0, f, such that it is accepting the language, such that L is nothing, but the language of that finite automata.

Then we call this is a basically a regular language. Any language is called regular if we have a corresponding finite automata, which is; whose language is the language of that L ok. Now, we know the finite automata can be for example, DFA it can be viewed as a PDA, Push Down Automata. Why? So, this can be viewed as a this is can be viewed as a PDA Pushed Down Automata, in the sense of anyway both are same accepted by the final state.

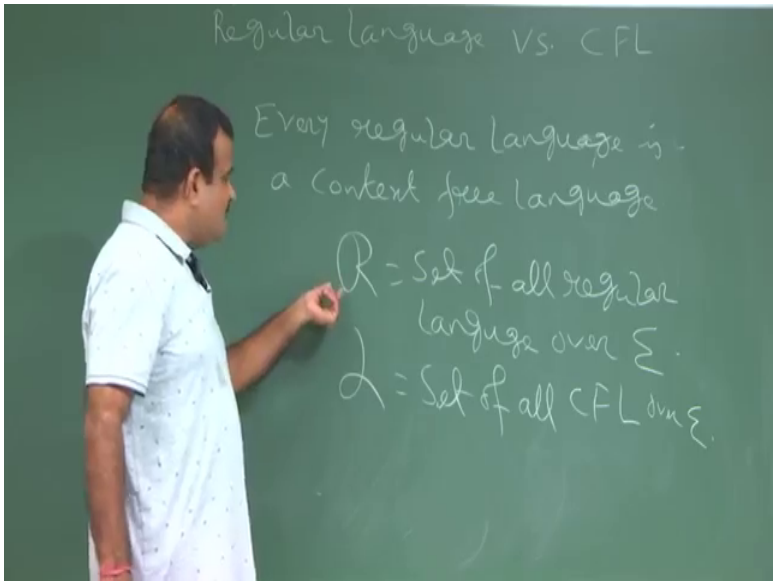
Because in pushed down automata what we have? We have the all are same Q and we have a stack symbol over here and we have delta then z 0, q 0 and F. Now everything is same except we have a new concept which is called stack symbol, but that we can

anyway this is we can ignore we can put anything in the stack. So, make sure that stack will be never be empty through that way.

So, we can keep on putting something on the stack in each step; so that way because we are accepting in that sense of final state. So, that way we do not care about stack, but only thing we have to may be careful, when the stack got empty, then there is no move our automata stuck, our PDA stuck, so that we have to be careful on that.

So, given a finite automata we can viewed as a push down or PDA so; that means, what? That means, this is a regular language is and push down automata means it is corresponding to a grammar. This will corresponding to a grammar G which is V sigma and then P and S which will sigma is the this same sigma and which will generate the same language. So, basically the regular language is a context free language. So, every regular language is a context free language, because it is a finite automata which is accepting that and it will give us the; so every regular language is a context free language ok.

(Refer Slide Time: 05:58)

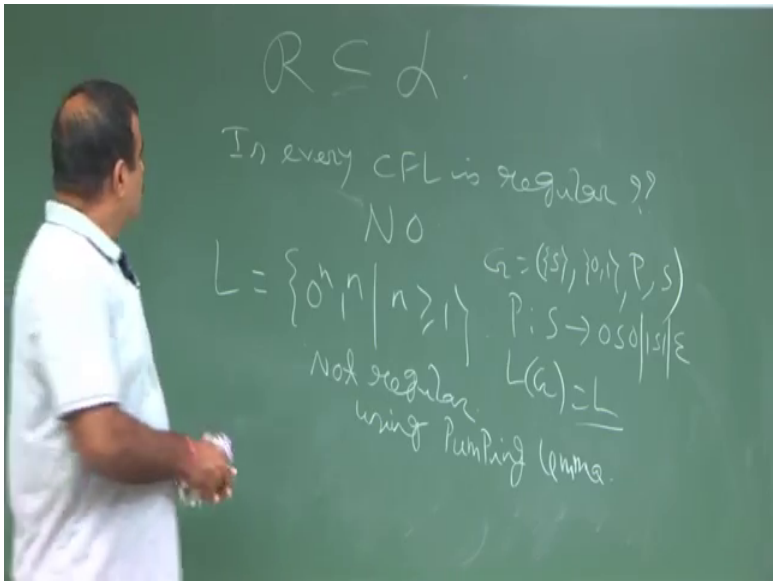


Because it is nothing but a pushdown automata which is accepting that and pushdown automata is nothing but a context free grammar and so on hence that is a regular.

So, if we see this R is a set of all regular language, so if we denote this R is a set of all regular language over sigma and C or we denote this L. Anyway this is just a symbol L is

the set of all CFL; Context Free Language over sigma. So that means, this any regular language is a context free language, so R is a subset of this L.

(Refer Slide Time: 07:31)

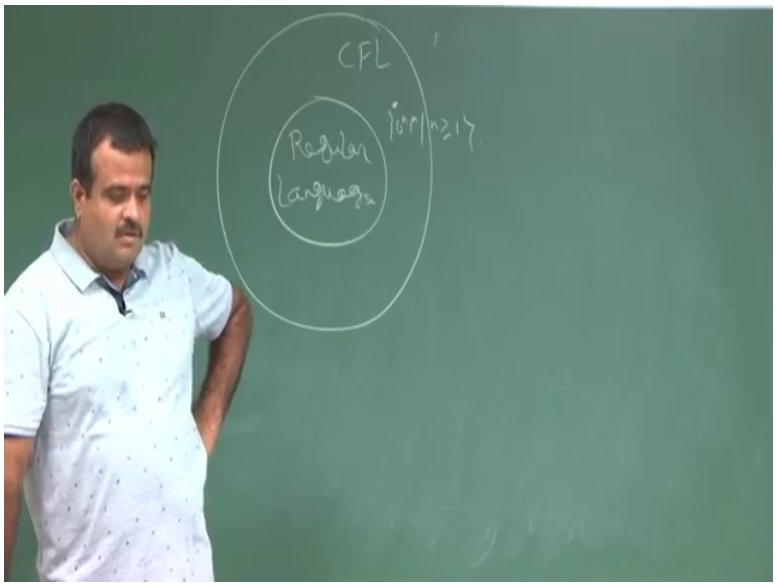


Now, the question is the other one is true. So, R is a subset of L. Now the question is the reverse I mean is the every context free language CFL is a regular, is every CFL is we know this answer regular? So, what is the answer of this we know this answer, so the answer is no. So, then you have to take a counter example which is CFL, but it is not regular; so who can remember this.

So, we know this 0 to the power n, 1 to the power n, n is greater than equal to 1 this language. If you consider this language this is a CFL, why? Because we can construct a grammar G, V, 0, 1 this is our sigma or T and P, S. So, V is only S say only one symbol. So, what are the rules? Rules are basically S is going to 0 S 0 or 1 S 1 or epsilon.

So, if you take that grammar it will generate the all the string like this, so this grammar will generate the L so; that means, this is a CFL, Context Free Language. But is this the regular language? We know this; no, this is not a regular language. Why? Because it is not satisfying the pumping lemma, pumping lemma is a necessary and necessary condition to be a language to be regular. So, this is not satisfying the pumping lemma. So, this is not regular by we can say by using pumping lemma, we can by using pumping lemma ok; so this is not regular.

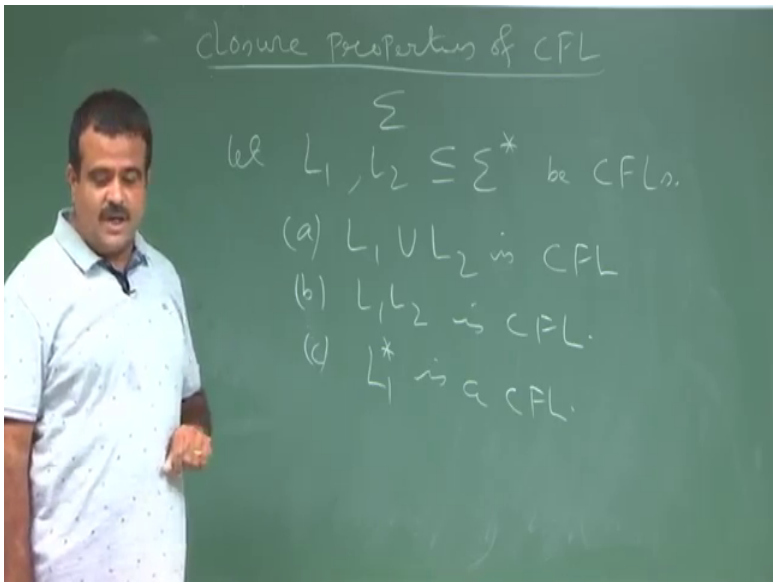
(Refer Slide Time: 09:56)



So, now the question is so this is not a sub so then the picture is like this, we have a set of all regular languages; say this is set of all regular languages ok. And then we have other set which is CFL.

So, because regular language is a CFL, every regular language is a context free language, but not every context free language is regular. For example, we have some member over here like 0 to the power n 1 to the power n, there are many examples of the language which are context free language, but which are non regular using pumping lemma we can show that ok.

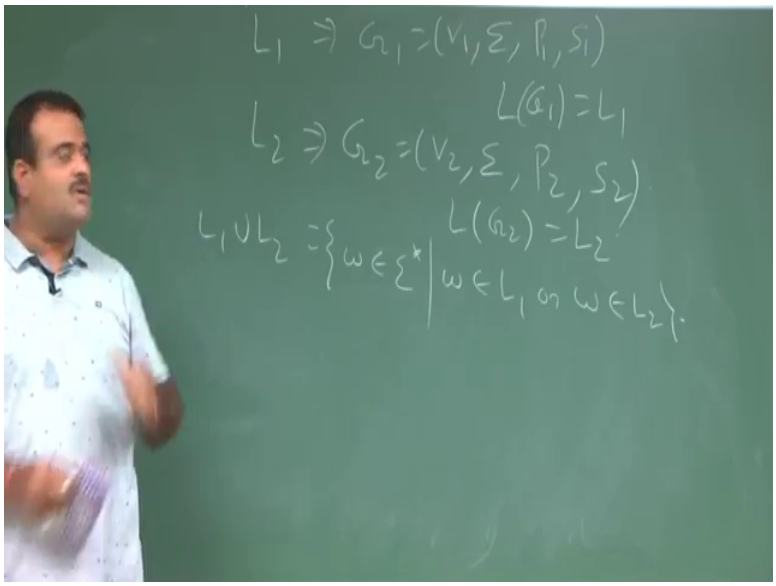
(Refer Slide Time: 10:48)



So, now we will talk about some closure properties of the CFL closure properties of CFL. We know the closure property of a regular language; that means, if we have two language L 1 and L 2. So, given a input alphabet it could be 0 and it could be a b, it could be anything. So, it is a finite set of symbols input symbols.

So, given this input symbol if we take two language L 1 and L 2 which is subset of sigma star, then if this both are CFL. Let this be CFLs to context free language, then we can show this these are the properties like it is closure under union. Then L 1 union L 2 is CFL and it is closure under concatenation. L 1 L 2 is CFL and it is closure and under (Refer Time: 12:05) closure like this L star, L 1 star is a CFL. So, we will slowly check this; these properties of the regular context free language. So, first we will check the how this is closure under union.

(Refer Slide Time: 12:38)



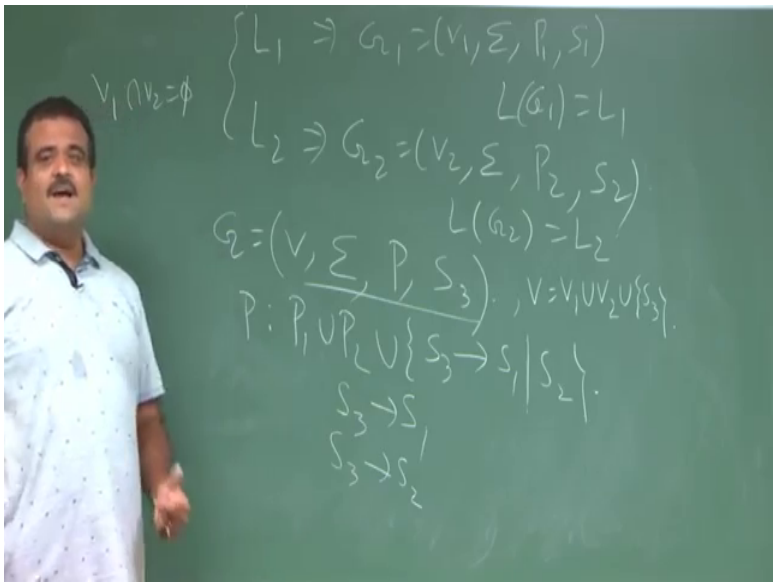
So, suppose we have two language which is regular sorry which is CFL, L 1 is CFL means what? Means we have a context free grammar which is generating this so; that means, we have a grammar G 1 which is V 1 sigma we can take same; I mean it is T 1 basically T 1 and T 2 for the other, but we can without any loss of generality we can take the same sigma or we can take union of that for, but anyway does not matter.

So, then P 1 and then S 1, such that L of G 1 is L 1 this is the first CFL which is corresponding grammar is G 1. And since L 2 is also a CFL we have corresponding we should have a grammar G 2 which is V 2; we can take the same input. But again I am telling we can take T 1 T 2 then we can take the union of that, but that is ok because we are talking about the language over a set input set.

So, P 2 and S 2 such that S 2 is the starting variable of the new of the second grammar; such that L of G 2 is L 2 the language generated by this grammar G 2 is nothing but L 2; so this is the situation. Now we want to show that L 1 L 2 is a CFL L 1 union L 2 is CFL. So, for that what we need to do we need to construct a grammar from this two grammar such that these two are I mean that will accept the; that will generate the L 1 union L 2.

So what is L 1 union L 2? L 1 union L 2 is the set of all strings, such that w is either in L 1 or w in L 2 that is the union definition we know. So, now the question is how we can construct a grammar L which will generate this so this is not tough to construct ok.

(Refer Slide Time: 14:53)



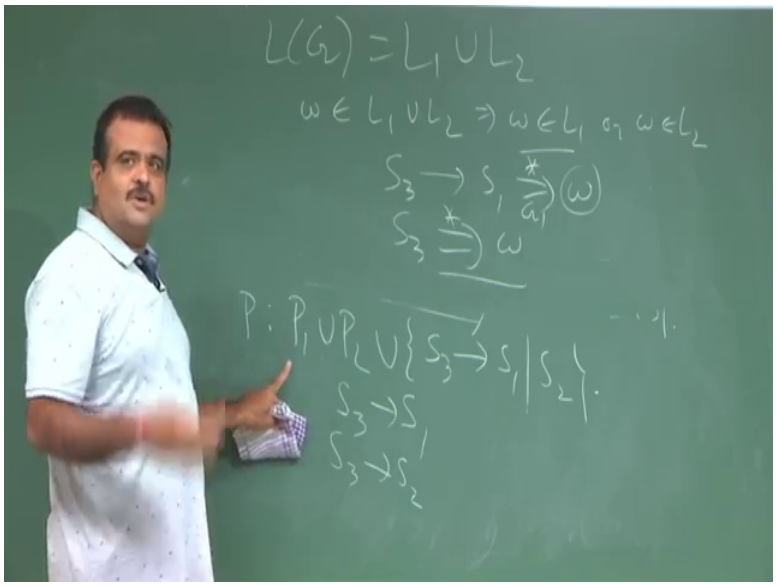
So, we will try to construct a G, we take help of say V is a variable said sigma we take same otherwise you have to take T 1 union T 2, if we take the different terminals. But we are we are talking about over the same input alphabet we so we are taking sigma over here and then we take a P, P we have to define and we take new starting variable S 3 ok.

Now, what is V? V is V 1 union V 2 union S 3. And here without loss of generality we can assume that there is no common variable in V 1 V 2, because if there is common name we can always rename that; there is no harm of renaming the variable. So, this is ok, this assumption is that V 1 V 2 are not containing; I mean their intersection is null, I mean they are not containing any common variables ok.

Now, how to define the rules for P? So, rules for P, P will be P 1 will keep the rules for v G 1 rules for G 2 and we add extra rule which is to accept that union. So, which is basically we starting with S 3 it is going to either S 1 or S 2 that is all.

So, these will make us I mean this is the rule whether we are going for I mean, we are taking all the rules from all the transition of sorry all the productions of G 1 and G 2, that is P 1 P 2. And apart from that we are taking two extra production, that S is going to S 3 the starting variable of the new grammar is going to S 1 or and S 3 going to S 2. And S 1 S 2 are nothing but the starting variable of this. So, this will make us acceptance of L 1 union L 2 so how? So we check that how this can be this can accept a.

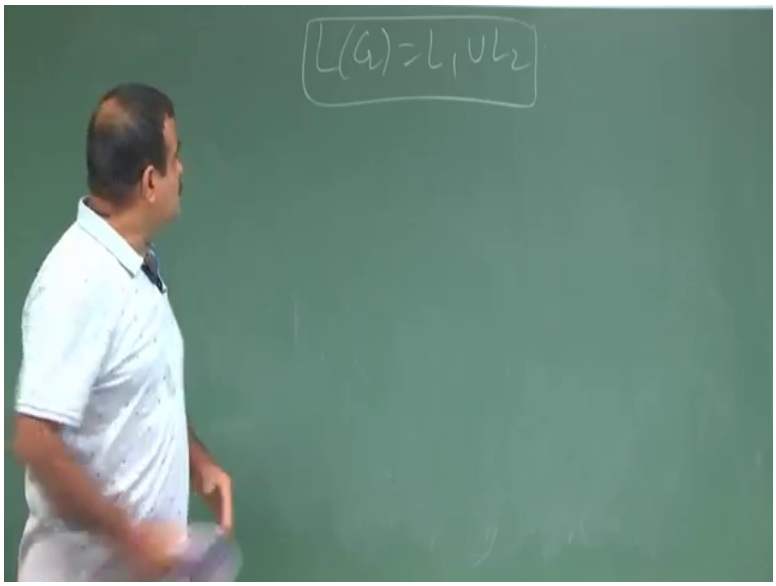
(Refer Slide Time: 17:24)



So, if we take a w. So, what is L of G then our claim is L of G is L 1 union L 2 ok. Now if we take a w which is L 1 union L 2; that means, this imply w is either belongs to L 1 or w belongs to L 2. If w belongs to L 1 so, what we do? We first take this move S 3 is going to S 1 and then as w belongs to and then as w belongs to L 1; then we know this is going to w.

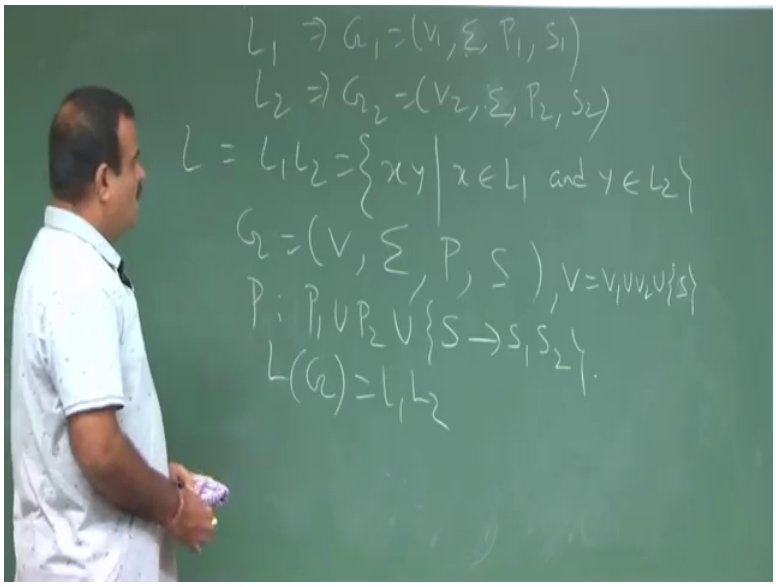
So that means, because this is the accepted string of G 1 so; that means, we have a derivation either leftmost derivation or parts tree anyway, but we have a derivation from S 1 to that string w. So that means, eventually we have a derivation from S 3 to w in G. So this is in G 1 because G 1, the production of G is consists of production of G 1 and production of G 2 so like this. So, similarly we can show that other way that if a string is accepted by a L of G so it has to be accepted by either a L 1 if it has to be in L 1 or L 2.

(Refer Slide Time: 18:50)



So; that means, L of G is nothing but ok, so this is the construction very simple.

(Refer Slide Time: 19:07)



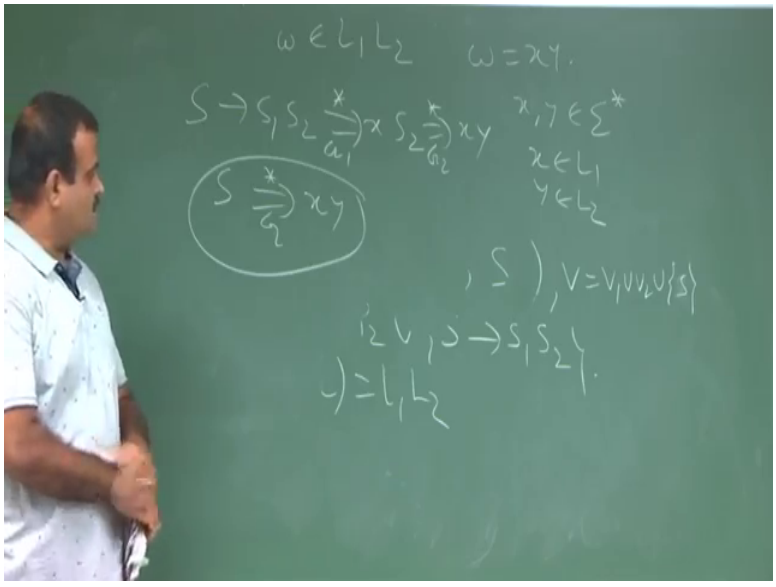
Now, how to construct the concatenation like we know, if L 1 is a context free language so, we have a G 1 V 1, sigma we are keeping same over here, P 1, S 1 and L 2 is another context free language. So, we have corresponding this thing, corresponding grammar context free grammar which is generating this. Here we could take T 1 T 2, then we have to take their union, but since they are coming from sigma; we can take sigma for simplicity there is no harm with that.

Because sigma we can consider the T 1 union T 2 so that is fine ok. So now L 1 concatenation L 2 means, this is set of all strings, so this is a L; all string of the like form like x y such that x is coming from L 1 and y is coming from L 2 ok; x is coming from L 1 and y is coming from L 2. Now the question is this regular; is this CFL?

So, answer is yes, but for that we need to construct a G context field grammar, which can generate this; so how to get a G, yes. So, let us construct a G which is V set of variables sigma will be same and P and now some new starting variable S 3 or we can say S anyway we have used S 3 this is S or S ok. So now what is V? V is nothing but set of all variables of G 1, set of all variables of G 2 and the new starting variable S.

Now, what are the rules? Rules are basically rules of P 1 rules of P 2 this is the productions and we are introduced a new rules where because we have to accept y x like this. So, that will be like this S is going to S 1 S 2 that is all. This is the concatenation I mean, this is the new rule we are adding. So, we have the productions of P 1, we have the productions of P 2 and then we are introducing the extra rule the production that S is going to S 1 S 2; so, that will give us the concatenation will be accepted.

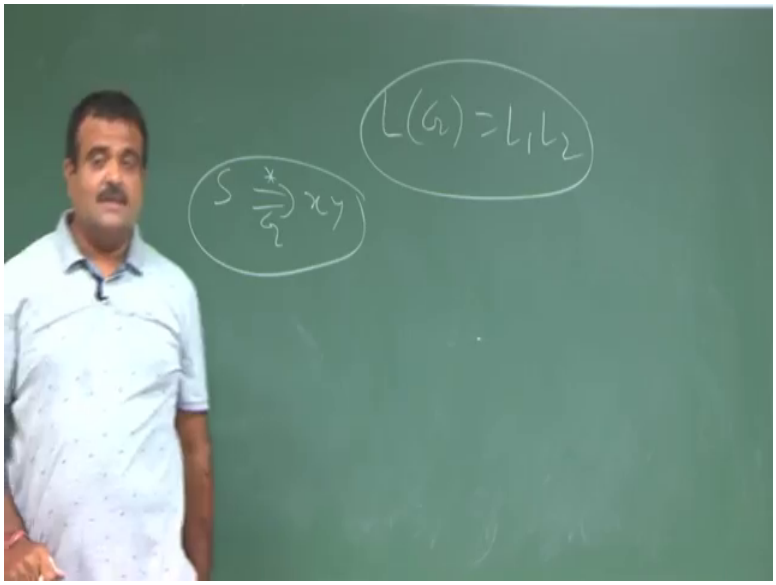
(Refer Slide Time: 22:34)



So, we can show that L of G is nothing but L 1 L 2. How? Because if we take this, if you take any w, belongs to L 1 to L 2; that means, w will be of this form x y. Now, x y are all coming from and x y, x is belongs to L 1 and y is belongs to L 2 now to accept x we need to.

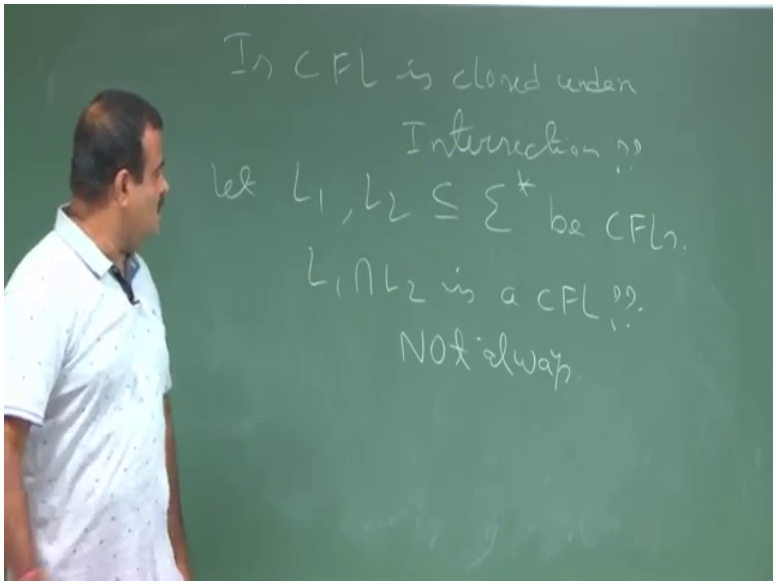
So, we can start with this S 1 S 2 now we can have a derivation for S 1 and it will eventually lead us to. So, this is for G 1 eventually lead us to x S 2 because x S is a terminal strings which is generated by this G 1 and then again y is a terminal string which is generated by G 2. So, this is G 2 those this will eventually lead us to y; that means, this S will be x y.

(Refer Slide Time: 23:54)



So this extra rules will give us this production. So; that means, this will give us L 1 and L 2, the L of G is nothing but L 1 concatenation L 2 this is the concatenation.

(Refer Slide Time: 24:11)

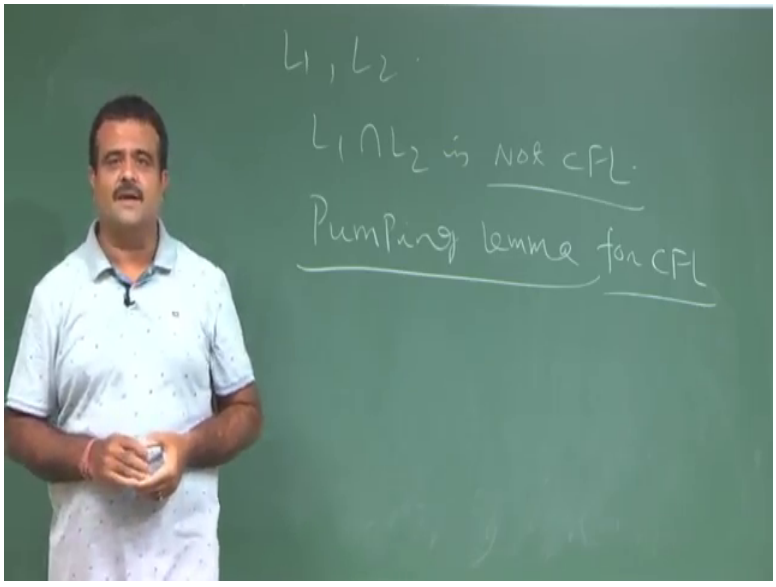


Now the question is the union is the CFL is closed under union; sorry, intersection. In that you have seen close under intersection ok; that means, if we have given to L 1 and L 2. Again I am telling this could be from the different sigma, different T 1 T 2, but we can take the union of T 1 T 2 as a sigma.

So, there is no harm by taking L 1 is coming from sigma star so we can take a one input alphabet there is no harm. So, these are all, these be CFL. Now the question is whether L 1 intersection L 2 is a CFL or not? We know these properties for the regular language right if I have two language which is regular, their intersection may not be regular. So, same thing is followed by here I mean, this intersection not always will be regular. So, this we can; so this is answer is no, not always.

So, if we have two regular language; if we have two context free language, which are I mean then their intersection need not be a context free always. So, to show that we need to take help of a counter example so for that we will show L 1 we will take two regular language sorry, context free language, L 1 and L 2 and we will show their intersection is not regular sorry, intersection is not CFL.

(Refer Slide Time: 26:13)



So, for that we will use, we will take some counter example for that we will use what is called pumping lemma for CFL which is necessary conditions for a language to be a context free language. Then using that pumping lemma we will show this is not regular; so that we will discuss in the next class.

Thank you.