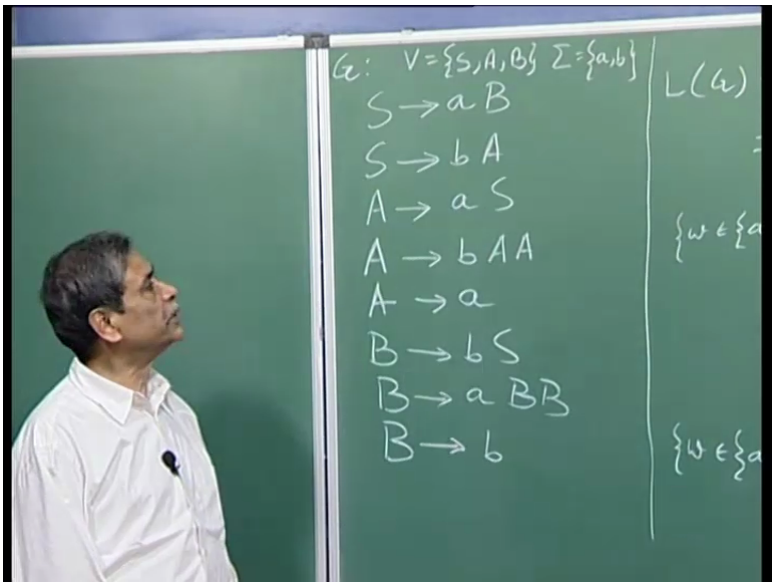
Theory of Computation
Professor Somenath Biswas
Department of Computer Science and Engineering
Indian Institute of Technology Kanpur
Lecture 22
Parse Trees, Inductive Proof that L is L (G)
All Regular Languages are Context Free

We will continue our discussion on context free grammars and languages in this class too. First of all what I would like to do is to complete an argument which partially I did last time and that is about recall this grammar G which had three nonterminals S, A and B. And the terminal symbols where these two, small a and small b and we had these production rules.

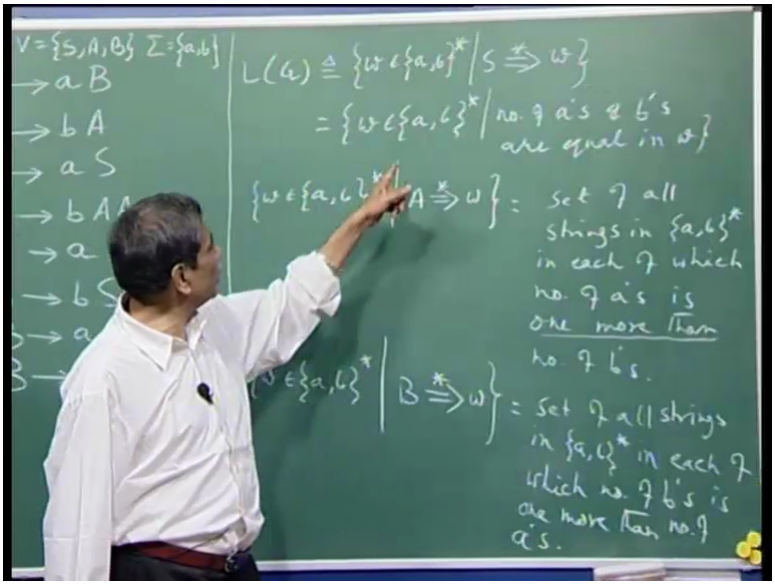
(Refer Slide Time: 00:54)



And what we said was that the language generated by G, this grammar which of course by definition and that is why I put this symbol which reads as language generated by G is by definition is (eq) equal to the set of all terminal stringseach of which is derived. That is all the terminal strings that you can derive from thestart symbol S that is of course the language associated with the grammar G which we call the language generated by G.

And the claim was that this language is precisely those strings over a and b where the number of a's is equal to the number of b's, right? This is what I am writing here.

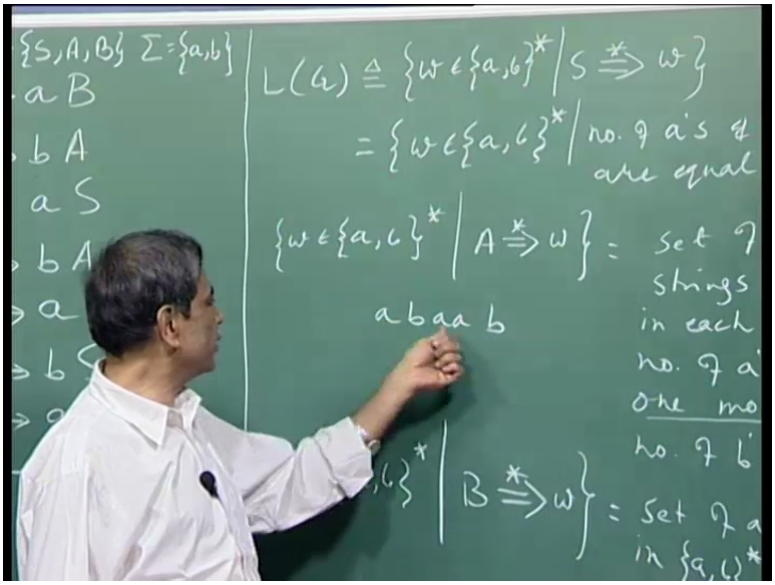
(Refer Slide Time: 01:57)



So this is of course is something would be interested in proving because that is what we say is about the language, right? Just looking at the grammar it may not be immediately clear. You would like to prove what the language is. Now we also said last time that to do so I need to provetwo other claims and there is a derives all strings, right?

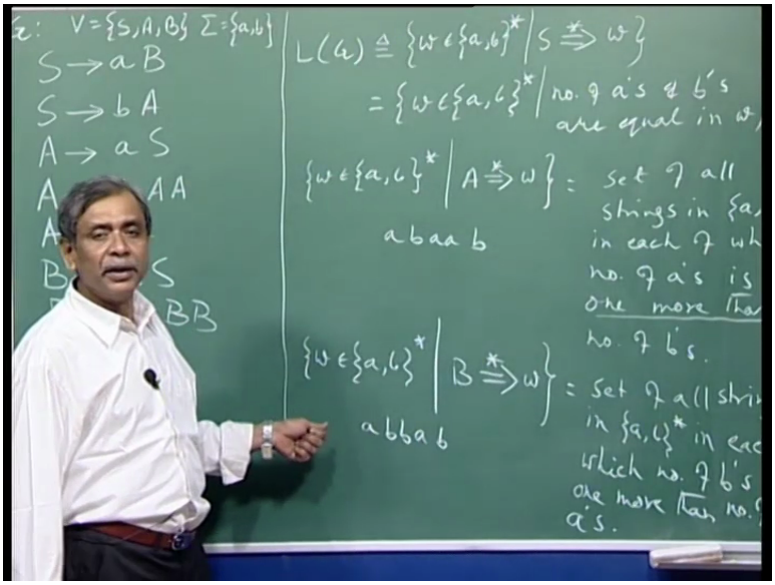
See there are two nonterminals other than S,capital A and capital B and we would like to claim that this nonterminal generates all strings over terminal symbols small a and small b where the number of a's is exactly one more than number of b's. So for example a string like a b a a b, here the number of a's is exactly one more than number of b's because there are three a's and two b's.

(Refer Slide Time: 03:03)



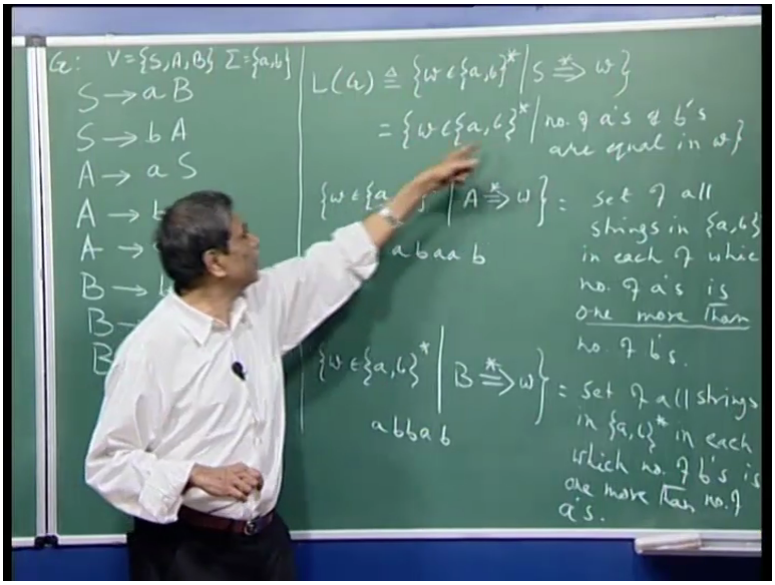
And similarly the set of strings which are derived starting from the nonterminal is precisely those terminal strings where number of b's is exactly one more than the number of a's. So for example if I had written something like this, a b ba b. Here this particular string if you notice that there are three b's and two a's. This string therefore should be in the set generated from the nonterminalB.

(Refer Slide Time: 03:42)



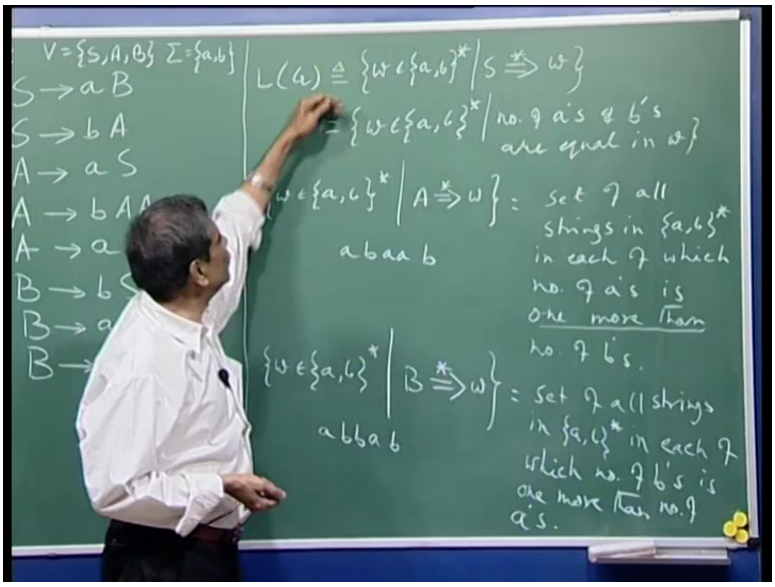
AndI have said this even in the context ofwhen we discussed regular languages that you see that I would like to prove the equality of these two sets. That is the set of all strings which are generated from the start symbol which is by definition in this case of course language generated by G and in some manner I am describing a set of strings over and b which is this, right?

(Refer Slide Time: 04:28)



This is the predicate which is satisfied by w and therefore I can talk of that this is a set of strings over a and b. And point is that we would like to show the equality of these two sets.

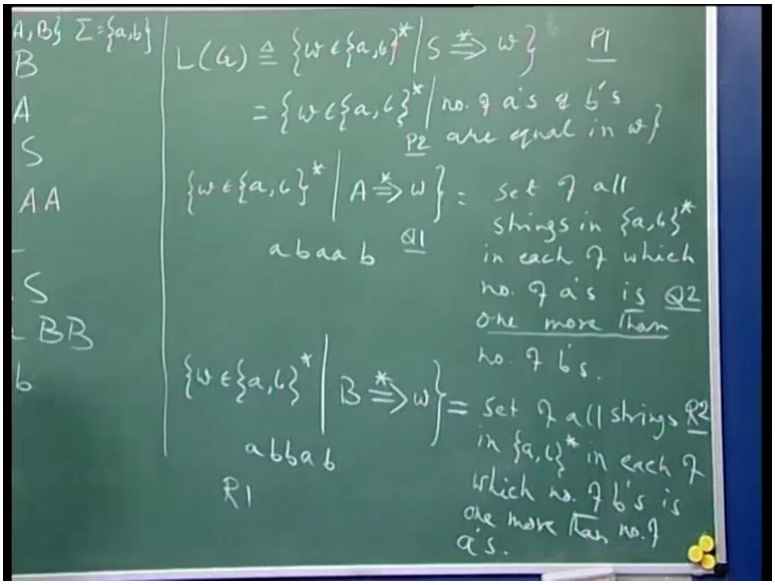
(Refer Slide Time: 04:42)



And when you show equality of two sets then this is the point we have made in our discussion many times that if this is of the form that I have this set let us say S 1 and another set S 2 and we would like to prove the equality, typically this is done by showing S 1 is a subset of S 2 and S 2 is a subset of S 1. What I definitelyhope I have been able to convince you last time that it is the case that this set of all strings which have equal number of a's and b's, they would be generated by S.

How did we do that? We said you know similarly ifyou look at this thatlet me call this set as, you know I am just giving a name P 1 and this right hand side is P 2. And let me say this left hand sideset is Q 1, right? And right hand side set is Q 2, right? And let me say left hand side of this equality, this is also an equality between two sets. This is R 1 and right hand side let me just call it R 2 for the time being.

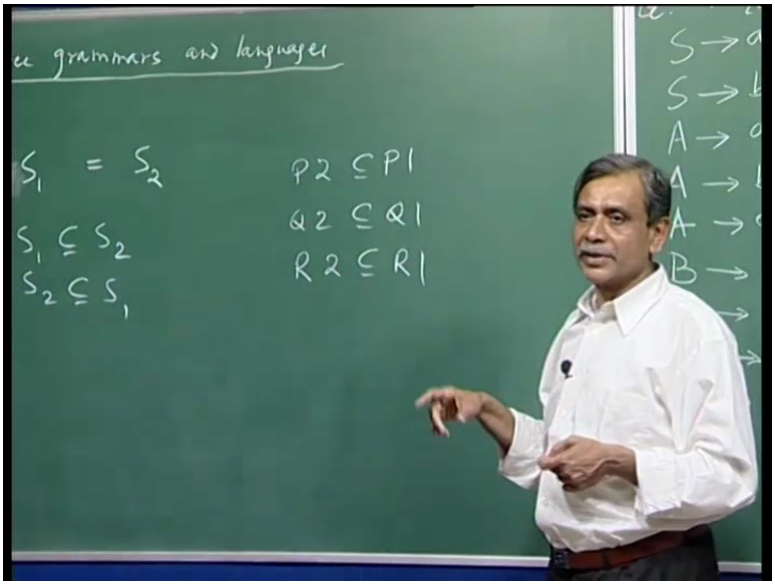
(Refer Slide Time: 06:41)



I think you would agree that we showed last time was that P 2 is a subset of P 1. Similarly Q 2 is a subset of Q 1 and R 2 is a subset of R 1. This we proved. How did we proved it? Justwe will remind ourselves what we said was that we proved all these three assertions simultaneously by means of what is called simultaneous induction.

So we are using an induction but all these three are been proved by induction but simultaneously we are doing it. So let us say that this something is true upto a certain length. All these three statements are true upto a certain length.

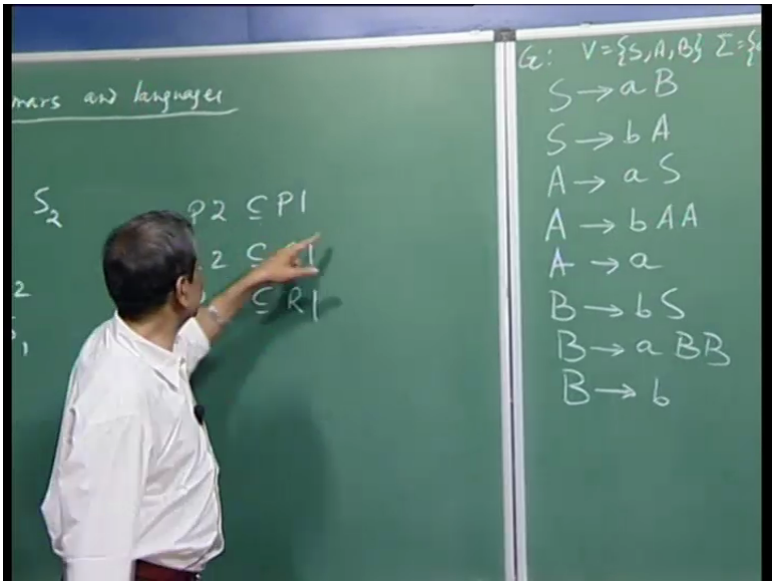
(Refer Slide Time: 07:41)



Let us see k. What it means is all string of length k or less which have equal number of a's and b's, they will be generated bythe start symbol S and so on. Then the induction is over this k. We showed that ifthat I take as the induction hypothesis then you know the next biggerstring also is something I will be able to generate. That is you know these assertions I can prove for the next you know assuming about k and then I can prove for k plus 1 and so on, right?

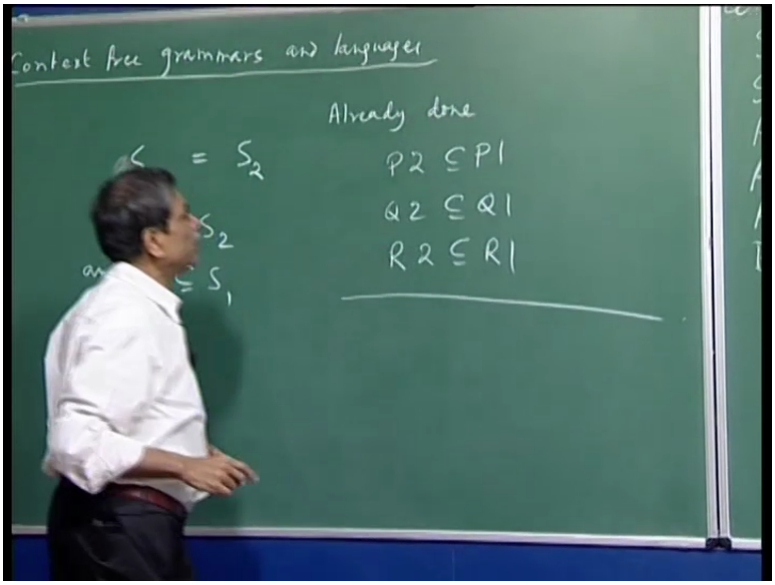
And it is not difficult to see in order to prove the assertion. This assertion for the next length I needed to use these two also and so onto prove this. For the next length I needed to use the other two further.

(Refer Slide Time: 08:54)



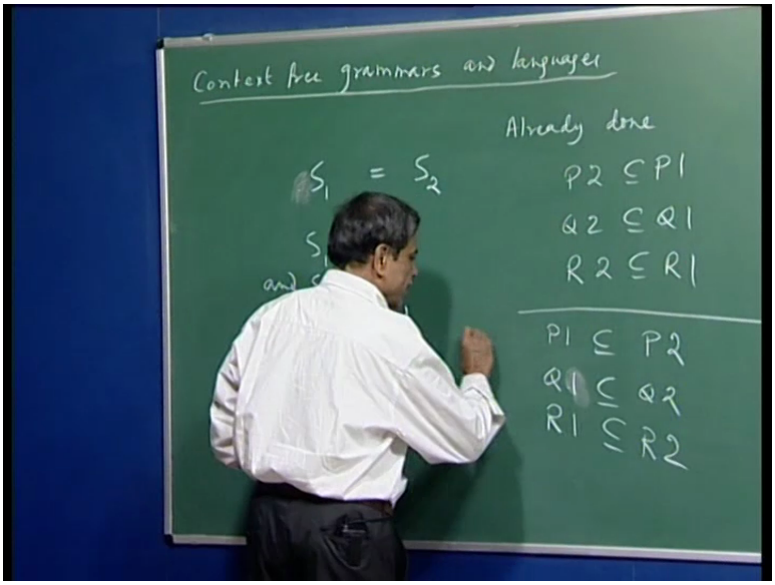
Sothat is whatsimultaneous induction is all about. That simultaneously we proved all these three assertions. So here our point was that the induction is over length of the strings that we are talking about. You know these sets have strings and each string has a length and our induction we are inducting over lengths of these strings. Nowso can I say that this we have already done? This part.

(Refer Slide Time: 09:44)



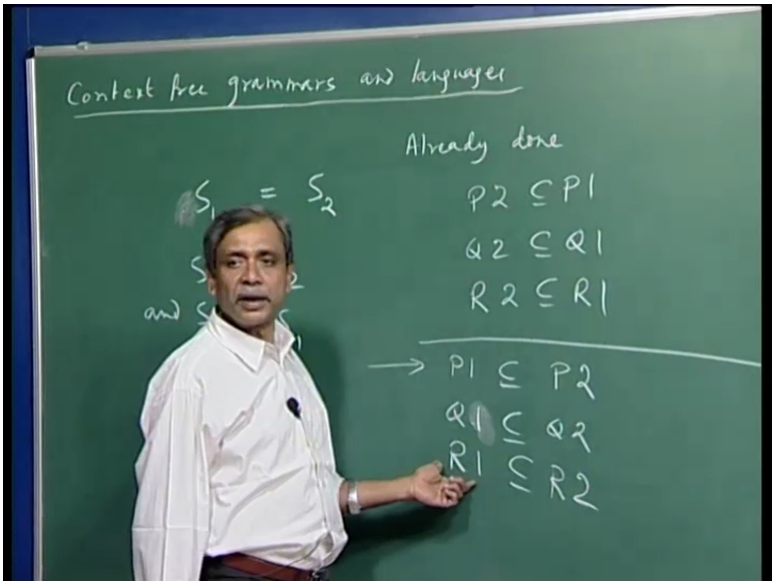
Now today let me just indicate how do I do the other part so that I can prove the equality. So basically today I need to at least indicate.Now Isaid that P 2 (con) contains as a subset P 2, this is something we proved. Now therefore I need to prove assertions like this, right? R 1 is a subset of R 2. If I manage to prove these three also then all six together means P 2 is equal to P 1 and which of course means what we wanted to assert about the language generated by G.

(Refer Slide Time: 10:51)



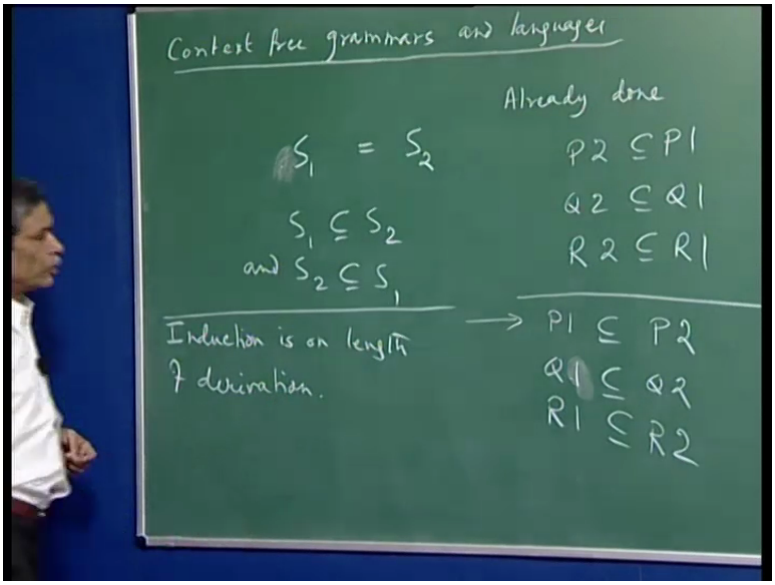
Nowso what is this saying for example? This is saying that every string which is derived from S, every string over the terminal alphabet which is over the symbolssmall a and small b, every such string has the property that it has equal number of a's and b's, is not it? This is what this means. That every left hand side is aboutterminal strings which are generated from S. Similarly this is about terminal strings generated from A and this is about terminal strings generated from R.

(Refer Slide Time: 11:37)



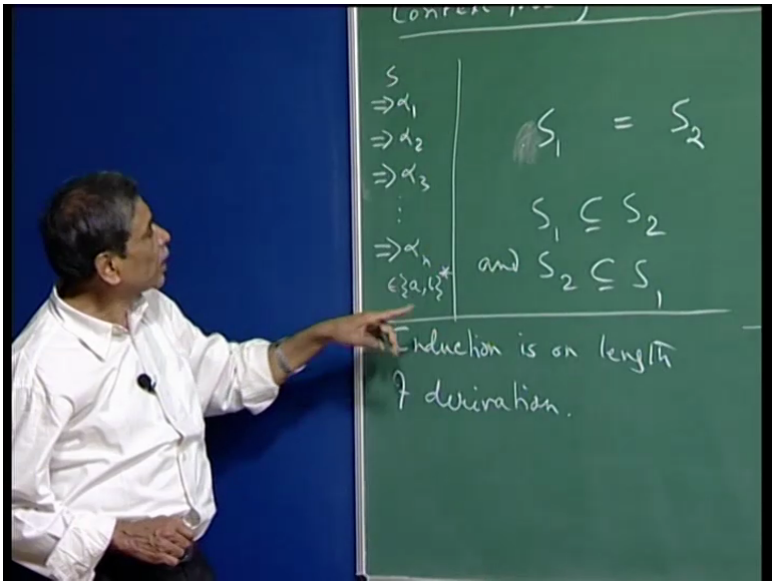
Now againin order to prove these three assertions we would again prove them simultaneously, again we will use induction, but now we shall use induction on the length of derivation, okay. Solet me say for example the induction is now, what do we mean by this?

(Refer Slide Time: 12:25)



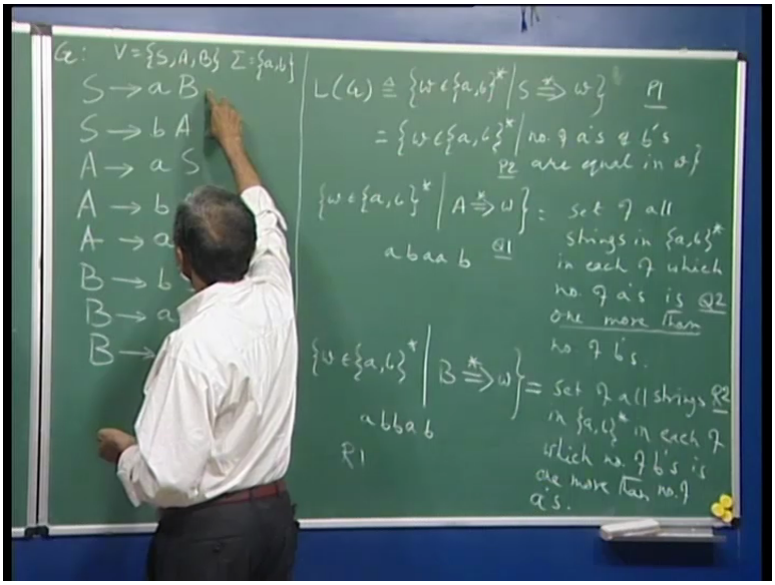
Say for example starting from S, I get alpha 1 then alpha 2. That is from alpha 1 I get alpha 2, then alpha 3 and so on alpha n, right? And this is an element of the terminal alphabet suppose I have.So what is the length of this derivation? This is the derivation starting from S, ending in alpha n.

(Refer Slide Time: 13:03)



Such a derivationclearly how many times I have used this (deriva) one step derivation? You can see we have used it n times. So length of this derivation is n, okay. Solet us try to look at the base case which is kind of veryobvious. Suppose by the way thus do you getany terminal string in one stepfrom S? You actually do not. If you see what happens isif you start from S, you can only use these two.

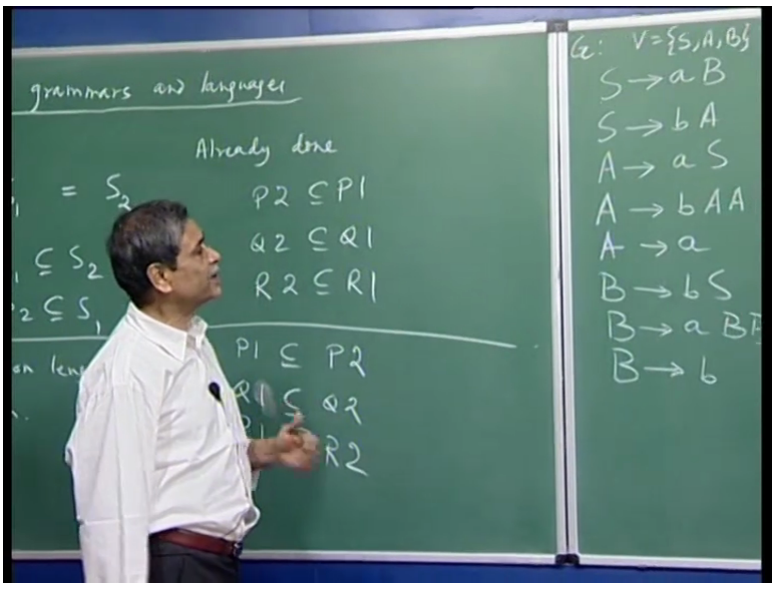
(Refer Slide Time: 13:55)



And in one step you can write this or you can write this and in either case you do not get any terminal string. Sofor S the base case will be let us say two. So there are two strings of length two which have equal number of a's and b's which are a b and b a. And you can verify that both of these can be derived from S, right? Andif we want to do everything from length so let us say that forcapital A also we can derive in one stepterminal strings small a.

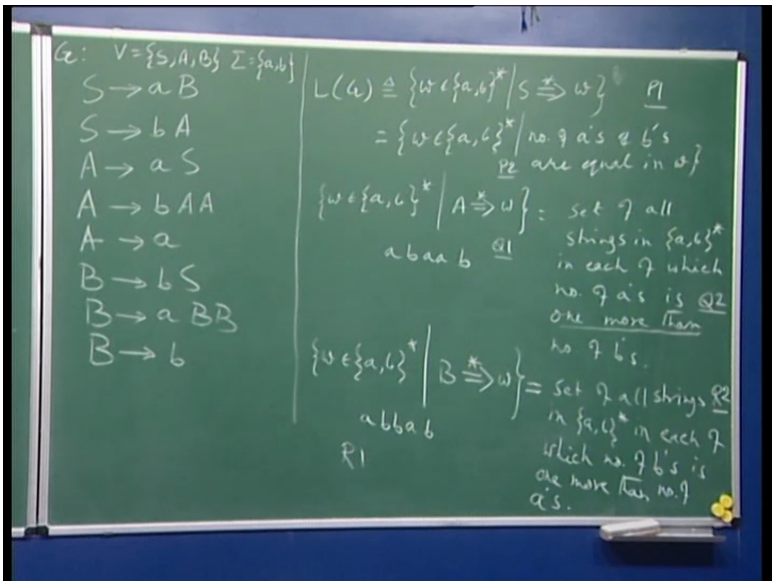
Similarly forB one step I can derive a terminal string b. In two step starting from B can you get anything? I guess you cannot becauseif you start from A in one step of course you can get this but you will see the two steps you cannot.

(Refer Slide Time: 15:21)



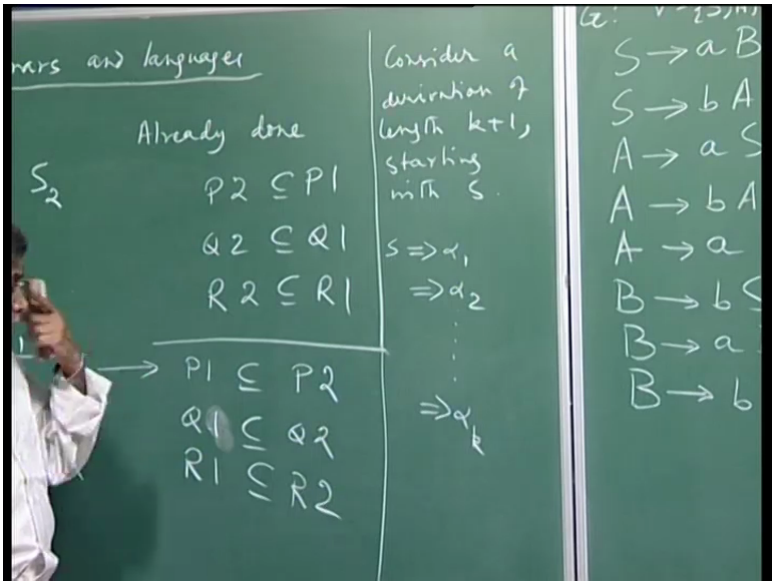
But whatever it is, the basecase can be handled. And I leave it as an exercise, what should be the proper base? Which you can do this. So let us say that base we have handled and nowwhat is the assertion thatI want to say? The assertion is that induction hypothesis is thatall derivation whose lengths are k or less they forthese three properties will be satisfied.

(Refer Slide Time: 15:54)



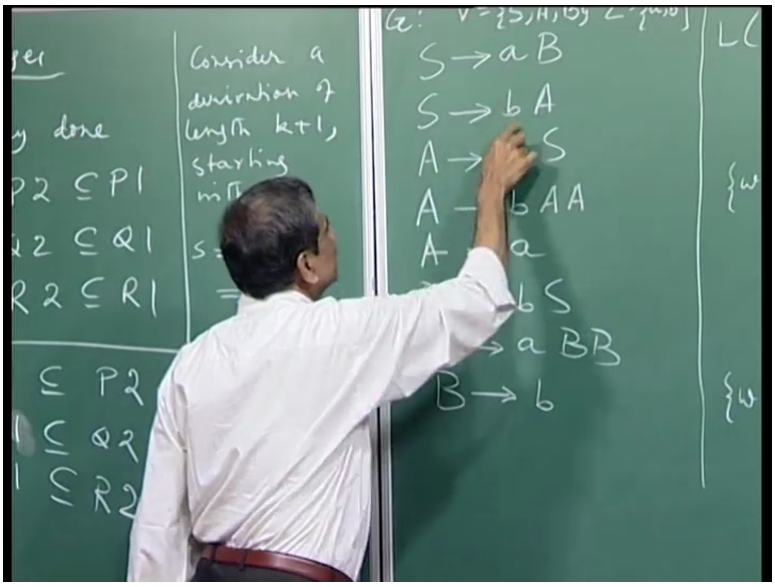
Which these properties are? We of coursehave said here. Now how do I go one more step? So let us say I am talking of a derivation whose length is k plus 1 for S, right? Solet us say considerderivation of length k plus 1 starting with S, okay. So you start with S. Now the point and that is something which is very simple to observe but once we observe this the proof becomes very clear, right? Sothe first is something alpha 1 and then I have alpha 2 from alpha 1 and here I will have alpha k.

(Refer Slide Time: 17:14)



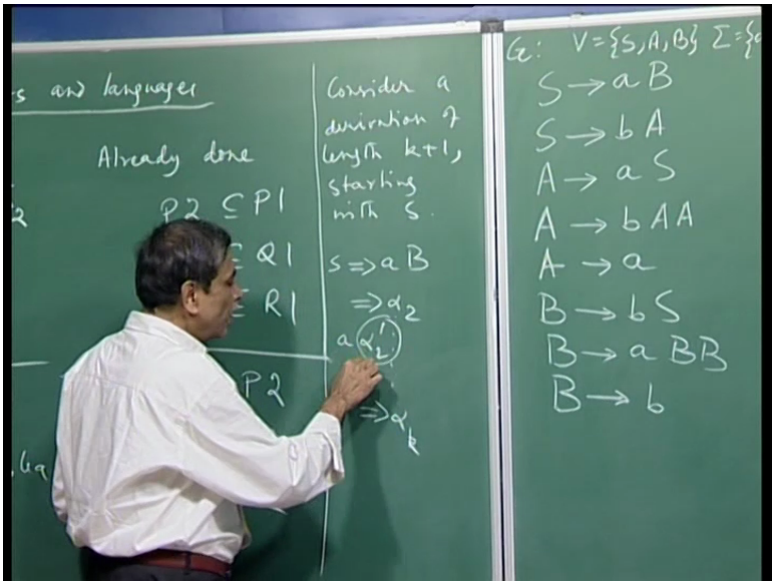
Now here what I have? That here is first step. Whatcould it have been? Either let us saythis or this.

(Refer Slide Time: 17:30)



So let us say I have my first thing is first step in that derivation. So take whichever derivation of length k plus 1 and then I have this, right? Now whatwe are saying you see that alpha 2, how did Iget alpha 2? Alpha 2 therefore necessarily is something a string which is of the form. Can you see this that alpha 2 has to be a string which is of the form a and then let us say alpha 2 dash. And what is this alpha 2 dash? Alpha 2 dash is a string which can be derived from the nonterminalB.

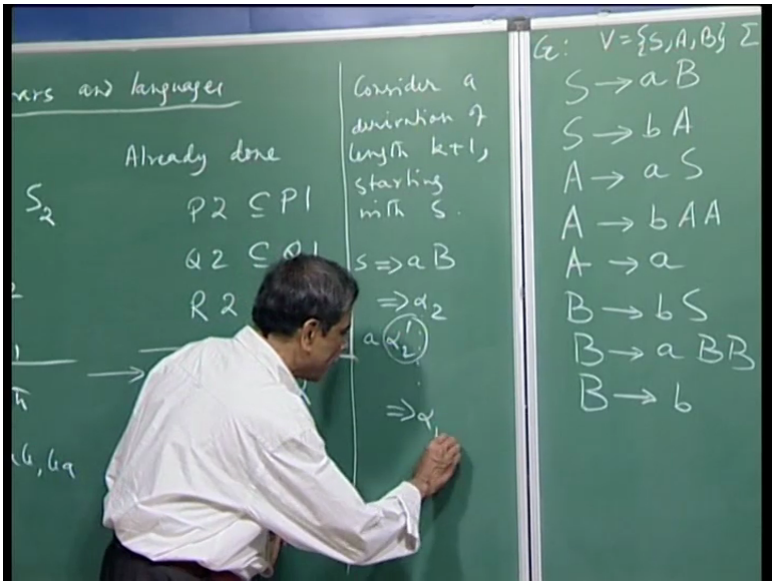
(Refer Slide Time: 18:10)



And now I used the induction hypothesis. This string necessarily will bethis derivation if I just take out this a which will always appear in alpha 2all the way up to alpha k. If I take these a's out then I am talking of length k derivation starting with the nonterminal B, right? By induction hypothesis this will be some string, right, which has one more b's than the number of a's. Which will be string over a's and b's because that is what we are assuming that alpha k is a string over a's and b's.

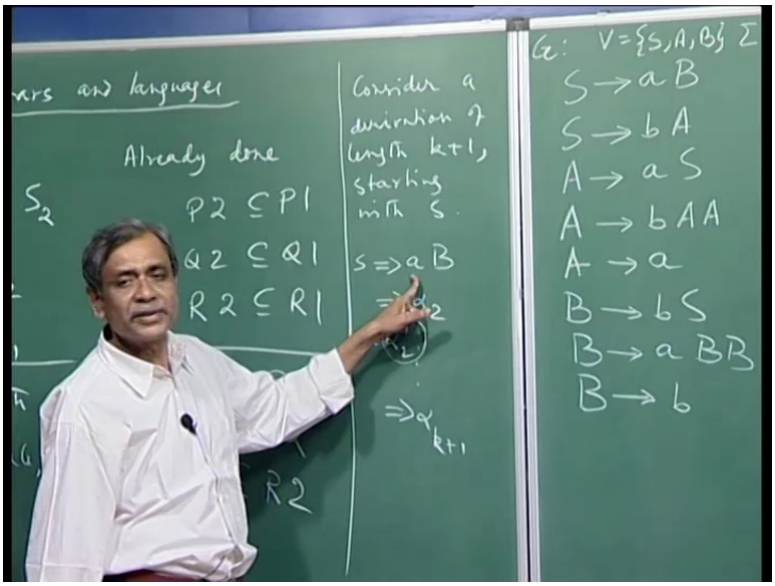
And alpha k is of the form a alpha k. So if Ileave that first a out what is left? If I call that alpha subscript k dash. That stringis something which is derived from B in steps k becausestarting from here we are doing it. So that is total thing was, I am sorry this was let us say k plus 1.

(Refer Slide Time: 19:32)



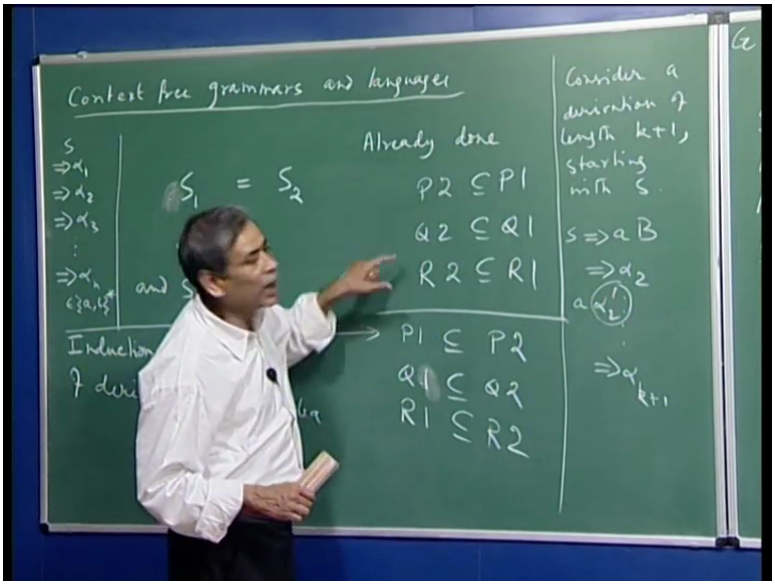
So the length was k plus 1. And now we are doing from here alpha 2 to, so which isfrom here to this which is the length k derivation of B. When Iput a dash taking the small a out and that is the string which has got one more b's than a's. And now in front of that there is a small a.

(Refer Slide Time: 20:00)



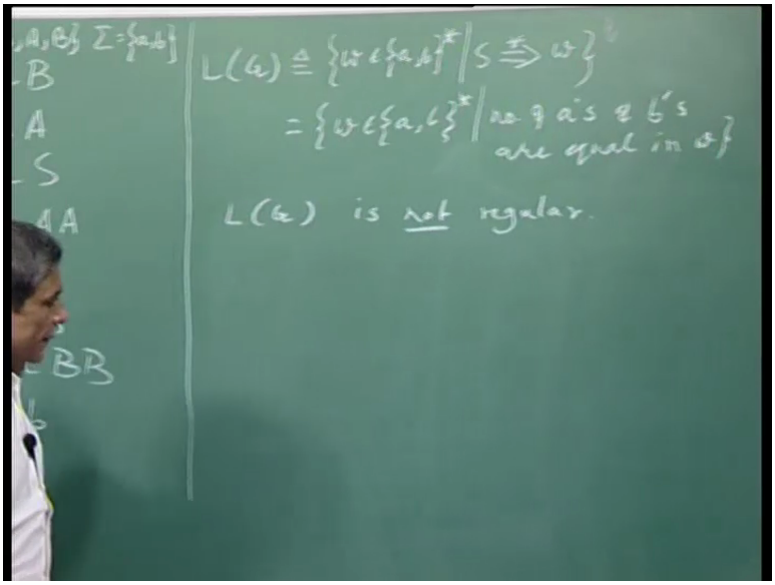
So total number of a's and b's will be equal. Soyou can see thatin this manner simultaneously taking all these three assertions at the same time and we can do an induction. Point I am saying is now we do the induction on lengths of derivation whereas in this partagain we did simultaneous induction but on lengths of strings.

(Refer Slide Time: 20:25)



Here these an example of a context free language which if you think for a second you will realize that this is not a regular language, right? This language is a set of all strings over a b where the number of a's is equal to number of b's. Very easily you can show that this set is not a regular set. So L G is not regular.

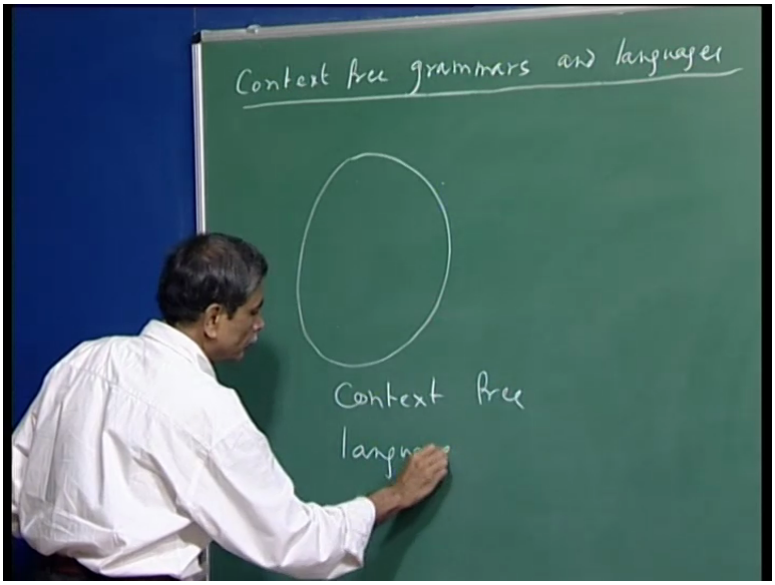
(Refer Slide Time: 21:15)



So I have at least one example of the language which is not regular but which is context free because here is this context free grammar which generates that language. What about regular languages themselves? Can we generate every regular language by a context free grammar? You see the reason is suppose, why this question is important? The reason for that, the

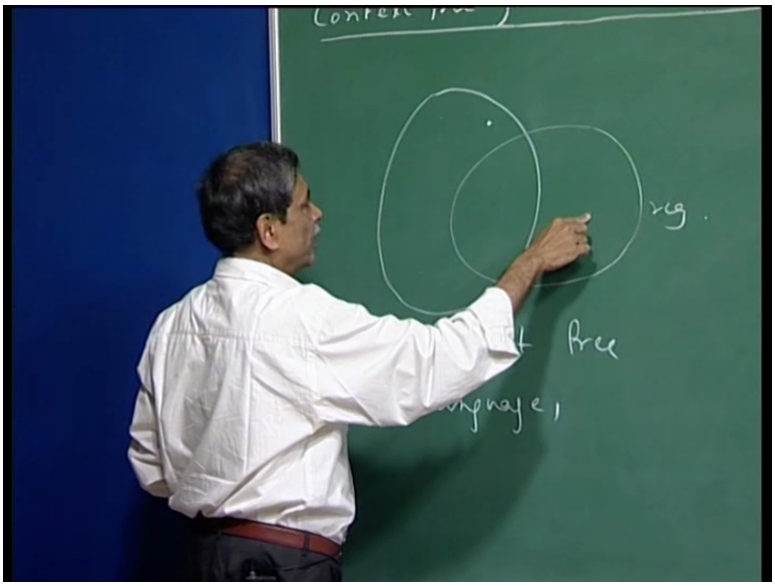
importance is if this is the set of all languages which are context free. So this is the class of context free languages.

(Refer Slide Time: 22:11)



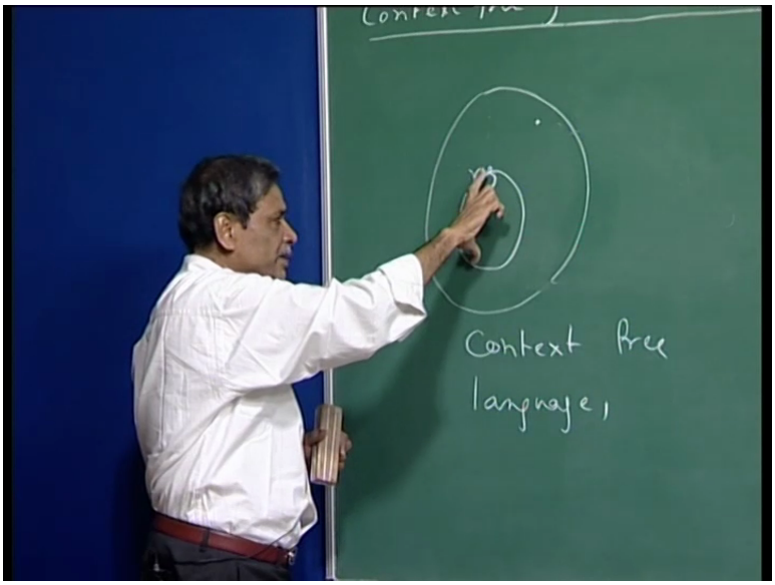
And here for example I got this set which is not a regular set. Now there could have been two possibilities when some elements here are not regular is either that this is also a possibility if I say that this set is regular. Could be that there are some regular languages which are not context free. For example if this is the picture then this is the regular languages which is not context free.

(Refer Slide Time: 22:47)



What we are going to show that is not the case. In fact all regular languages will be context free languages. So that statement in picture what does it mean? Here is the class of regular languages. When I say every regular language is also context free that meansthis class is a subset of context free class.

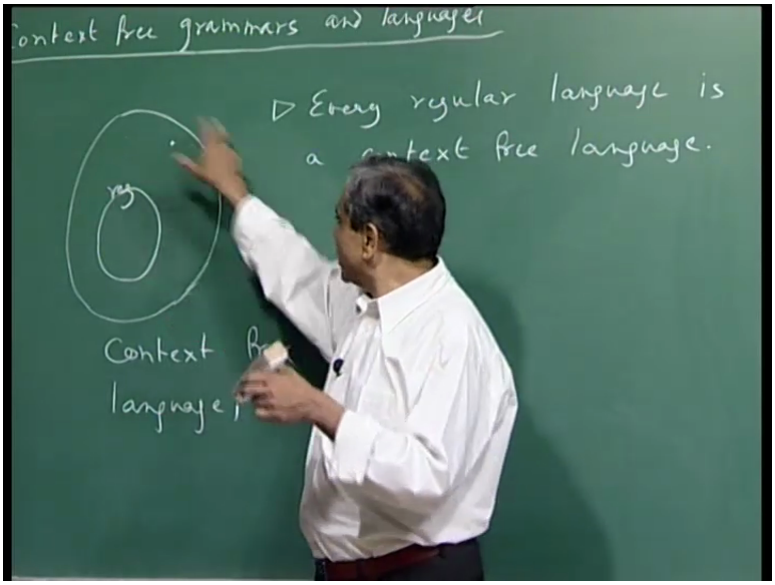
(Refer Slide Time: 23:17)



What more I know there are examples of languages which are context free but not regular. So this containment is actually proper and this is really the right picture. I need to show you this. So therefore I need to show that this is the claim that every regular language is a context free language. If I managed to prove this along with the example that I have here of one context free language which is not regular that means the picture or the relationship between these two classes isthat the class of context free languages properly contains regular language.

In that sense we have progressed. If I manage to show that you know we had some set. Wedescribed its properties, we did many things with regular languages but we also saw that you know certain simple languages could not be regular languages and here I have a larger class.

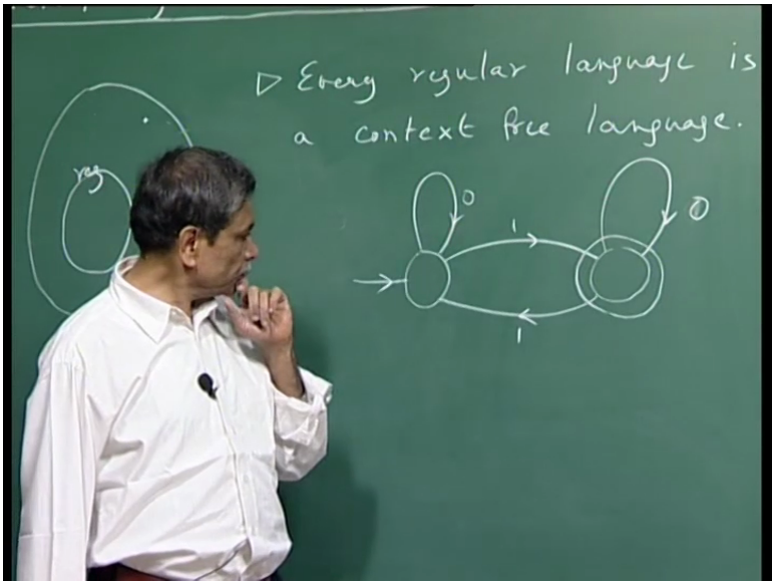
(Refer Slide Time: 24:54)



That you know overall whatthis course is trying to do is to ultimately be able to capture all languages which are computable, the most general sets, alright.So you would like to prove this. Let me give you an example of how would the proof go and thenwe can formalize that example to see the proof exactly. So letus actually take a context free language. And of course context free language you can describe by a DFA. So why not takea DFA.

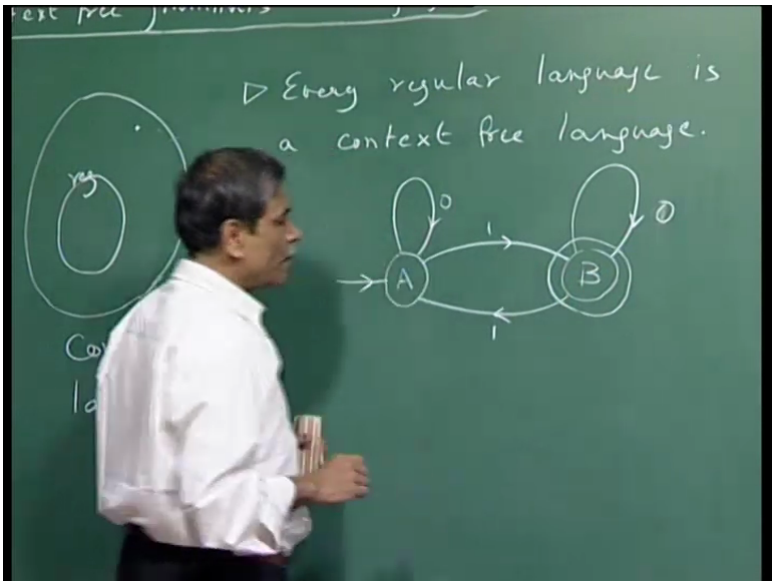
So letmein particular consider this DFA, okay. It has two states andyou should be able to almost look at this and immediately see whatis the language accepted by this DFA. And that language is very clear that it is the set of all binary strings where the number of 1's is odd, right?

(Refer Slide Time: 26:50)



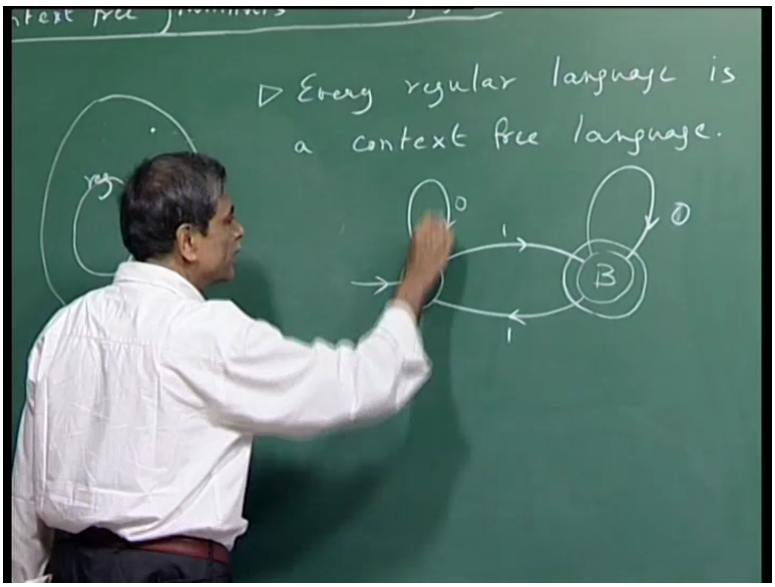
Alright, soI will show you how to get acontext free grammar for that language that is the set of all binary strings where the number of 1's is odd. Do not care anything about the 0's if you notice in this example.And the way we are going to do this is we will associate anonterminal with every state of this DFA. So let me call this state as A, this state as B, right?

(Refer Slide Time: 27:30)



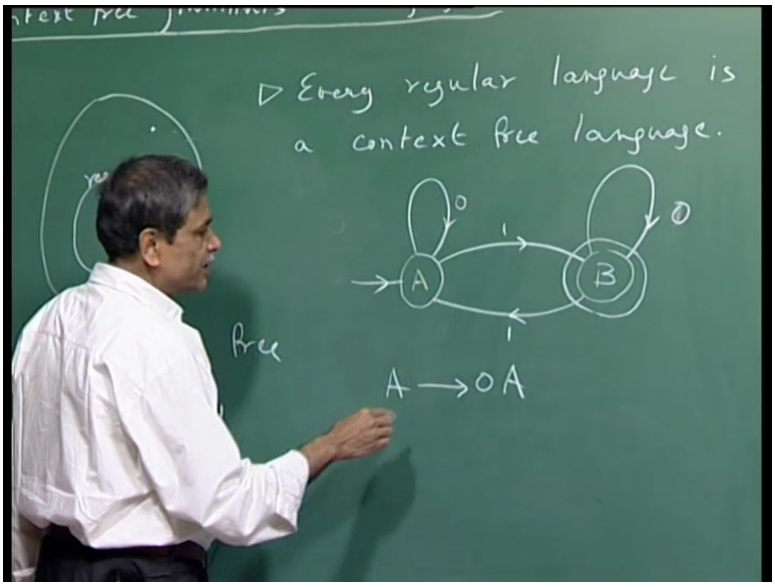
And what we are going to do is if you see what does it say? Solet me first write somethingthat from every nonterminal that I have written which is of course corresponding to the state, my rules are going to be if you consider a transition, right? So here one transition is this, right?

(Refer Slide Time: 28:02)



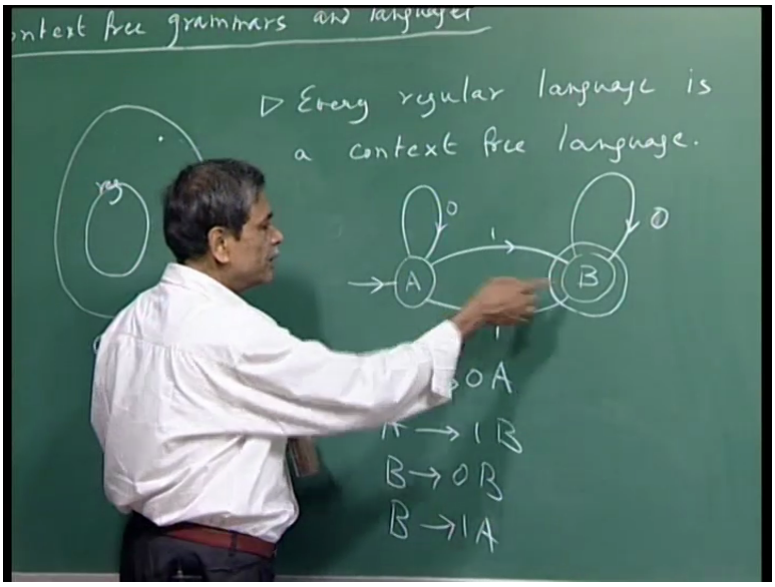
So I will put like that. So A 0 A. We willjustify this a little later.And again starting from A I have another arrow going out which is 1 followed by B. So, and similarly I would write for B, right?B is you know one arrow is like that so 0 B. B is also 1 A.

(Refer Slide Time: 28:15)



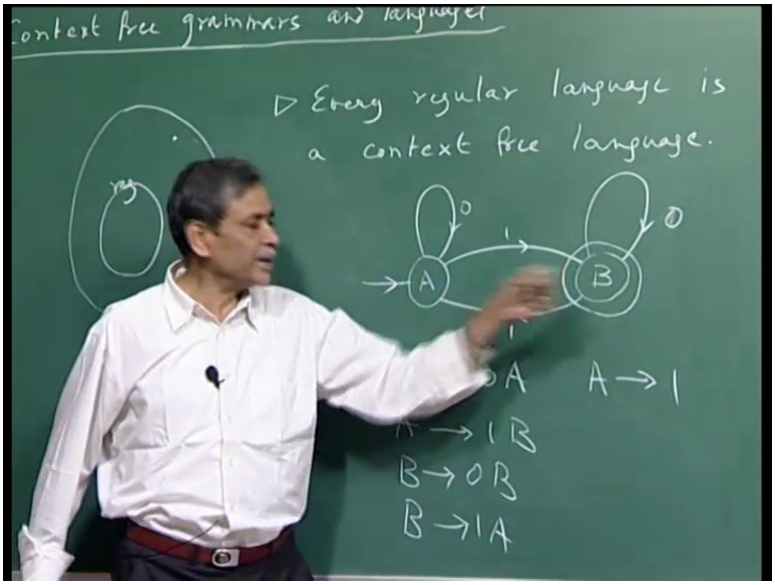
Now that is what I have taken. There are four transitions, four arrows you know four transitions which were there. I got these four rules plus I need toconsider one more kind of production rule which will correspond to all those arrows which are ending in a final state. Here there is of course one final state you see.

(Refer Slide Time: 29:17)



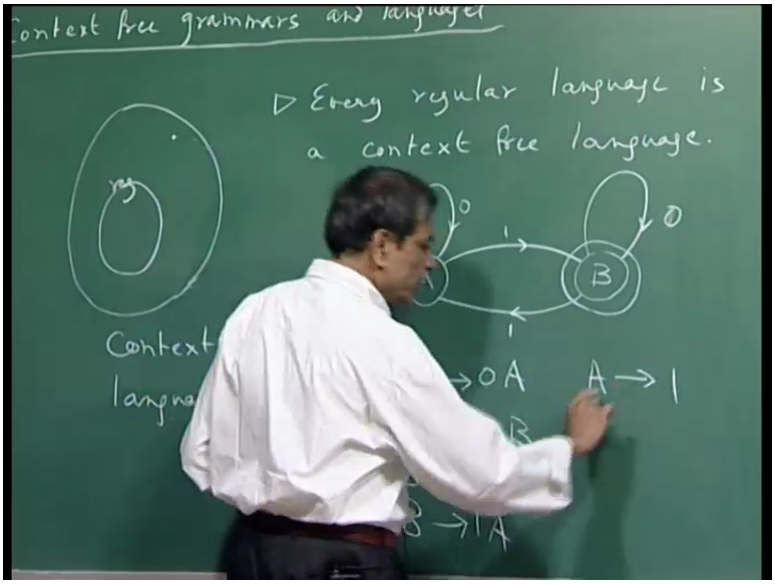
So starting from A using 1 I could have gone to B which is a final state. In such a caseA I will just write 1 and not write thisstate.

(Refer Slide Time: 29:37)



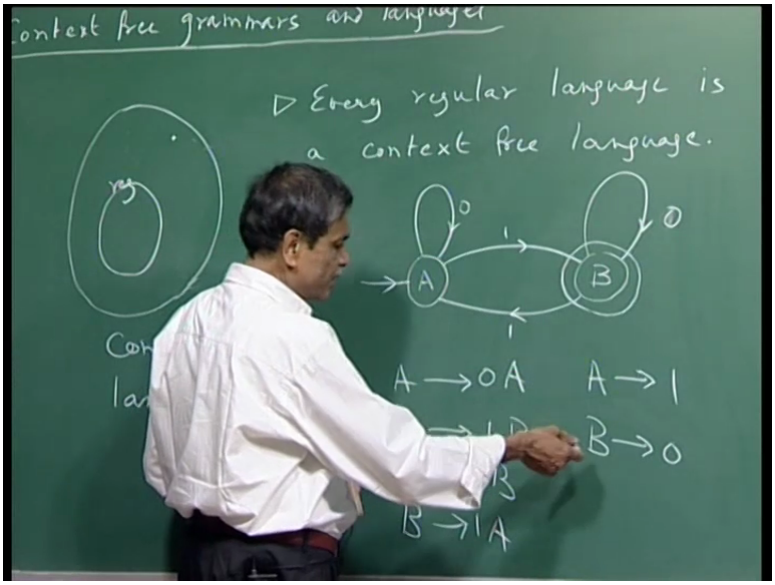
You see this arrow, this transition I used to define this particular rule, A goes to 1 B. But since Bhappens to be a final state of the DFA,I will also define another rule A goes to 1.

(Refer Slide Time: 30:13)



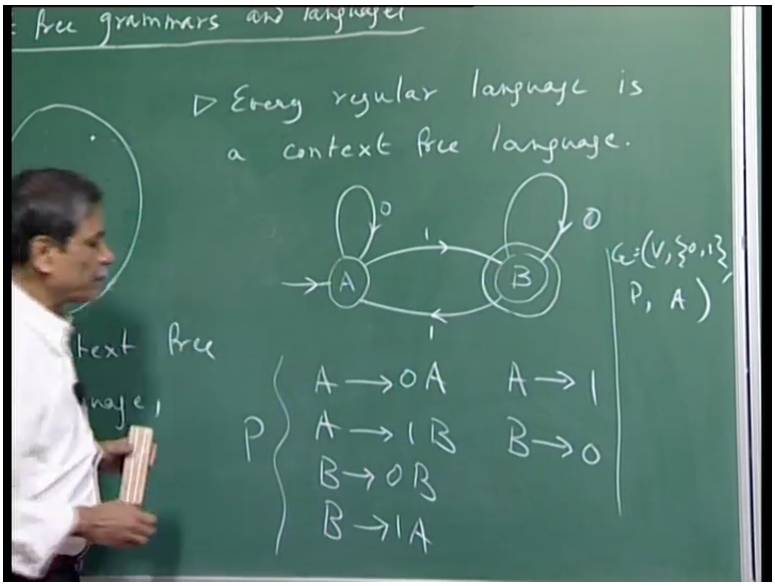
Sothis is something whichon the right hand side there is no nonterminal. Similarly you see this B.So B can go to, right? Because B can be rewritten as 0 B but Bin this this kind of removal of the state or the nonterminal symbol I doonly on the right hand side. That makes sense, right?

(Refer Slide Time: 30:49)



Andlet me define the grammar now. G is V and then of course 0 1 P. And the start symbol here is A because that is the statewhich was the initial state, okay. And these are my productions.

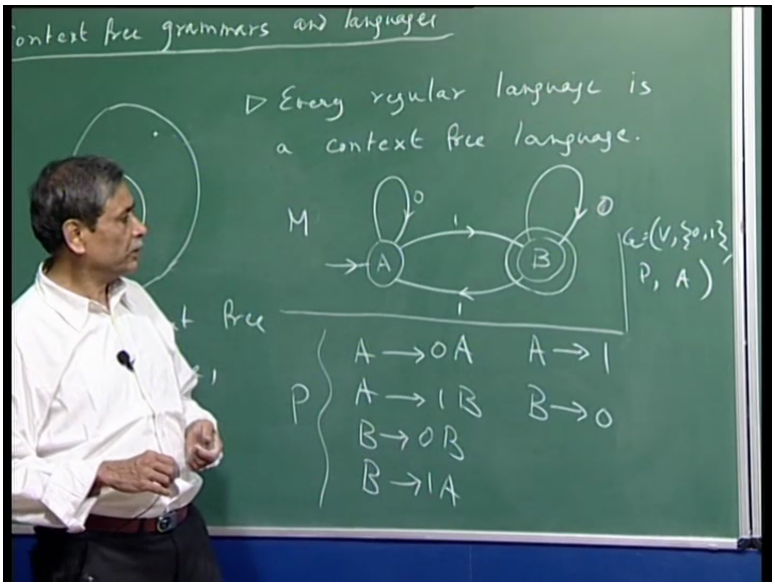
(Refer Slide Time: 31:21)



Nownotice this.In this examplehow can I claim that this grammar, the language (cor) corresponding to that grammar is actually the set of all strings which are accepted by this (()) (31:46)? So idea of that is not difficult.What is our problem? We would like to show that the language which is accepted by this DFA is equal to the language generated by this context free grammar.

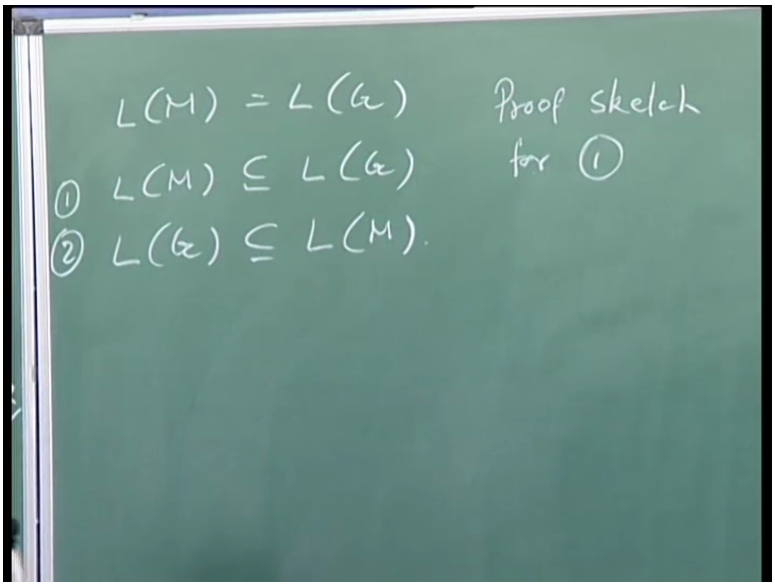
By the way that it is a context free grammar it is clear because every production rule is of that form that left hand side is a nonterminal and the right hand side is a string over the union of the two sets, terminal and nonterminal, okay.

(Refer Slide Time: 32:25)



So formally we would like to show that the language accepted by the DFA M is same as the language generated by the grammar G. And as before again there are two parts in this.Proof normally that language accepted by M is a subset of language generated by G. So and this is the other ways. You have L G is a subset of L M. So let us say proof sketch forpart 1.

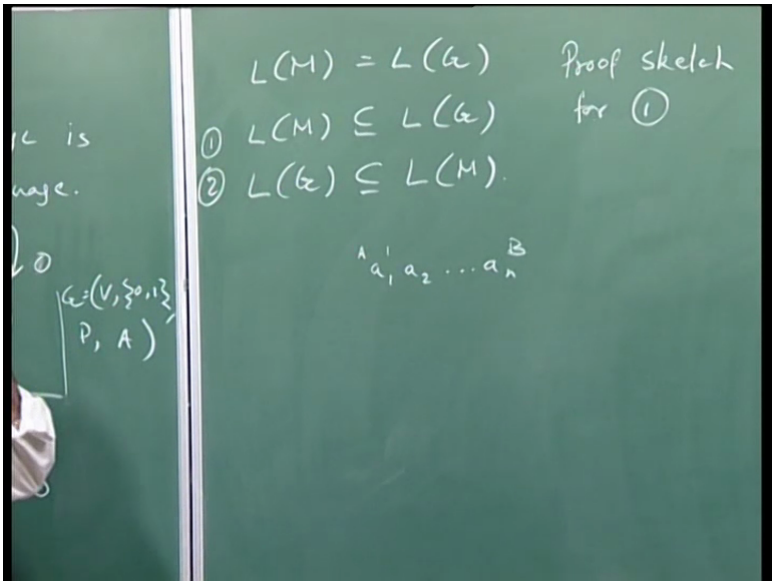
(Refer Slide Time: 33:21)



So I need to show that every string which is accepted by the DFA M can be also generated by this grammar G. Let us do that. So if I have a string which is accepted by this machine M then such a string supposing that string is let me say a 1, a 2, a n which iswhere each a 1is 0 or 1. And where do you start?In the initial state is A.

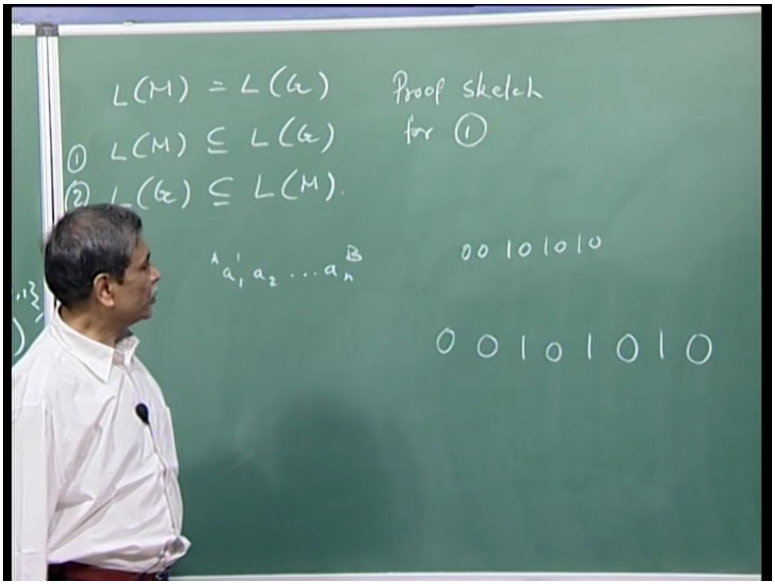
So let me just write it A and then this a 1 came. You will be in some state here either in your B. So this is you know whatever state you are here and finally there is only one accepting step and here you could have gone toB.

(Refer Slide Time: 34:27)



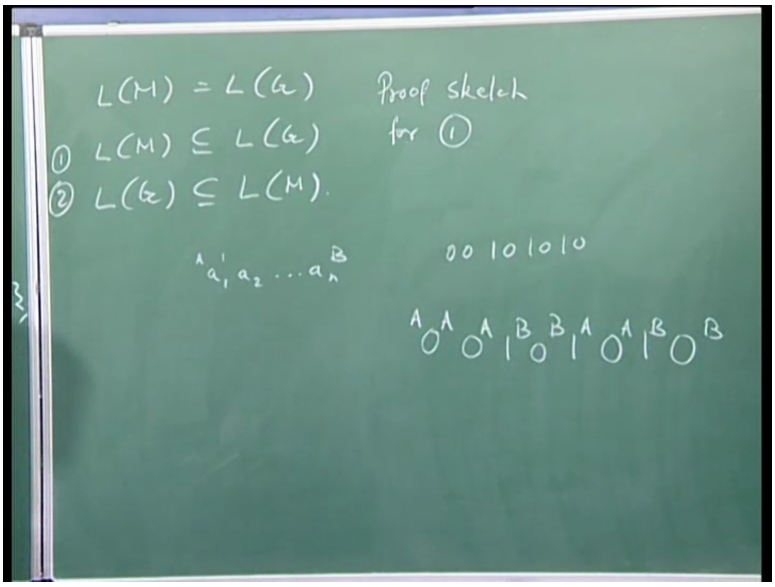
So let me show you the idea witha simple example maybe. Then you should be able to do it yourself. So let us say I have this 0 0 1 0 1 0 1 0, okay. Now thisis a string which will be accepted by A B and let us see howfor the string the state transition for the machine is going to be? So let me just writing this as a little separate. I mean the symbolseparated so that I can write out the states clearly, 0 0 1 0 1 0 1 0.

(Refer Slide Time: 35:14)



Here we will start at the initial state A and on 0 remember the machine stays in this state itself, right? So A A and on this one from A it will go to B and then B again if you are in state B and 0 comes, you will remain there. So B and then on this one you will come back to A and then here you are going to remain in A, here you will go to B and on this also you will go to B.

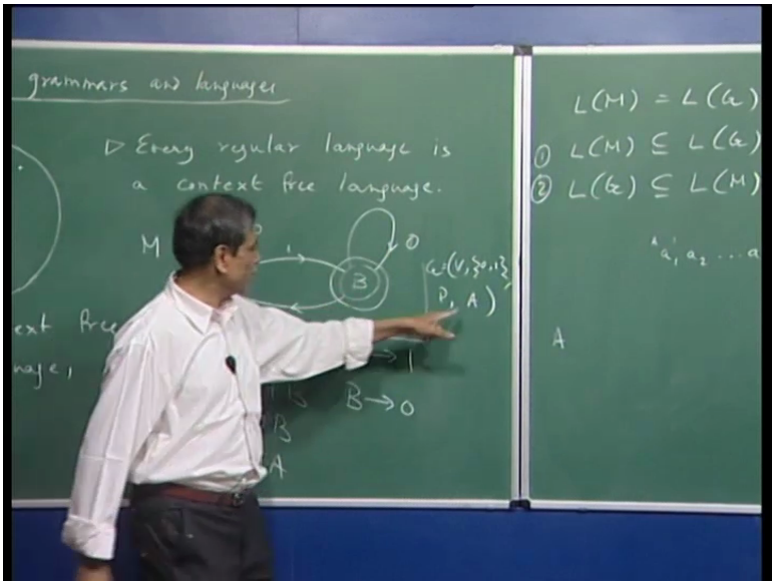
(Refer Slide Time: 35:51)



Now looking at this and looking at this set of production rules I can show you that what will be the corresponding derivation. See I would like to show that since this string is accepted because this string takes the machine from A to the accepting state. Initial state A to the accepting state B. The string is accepted by the machine M and the claim is that the string can

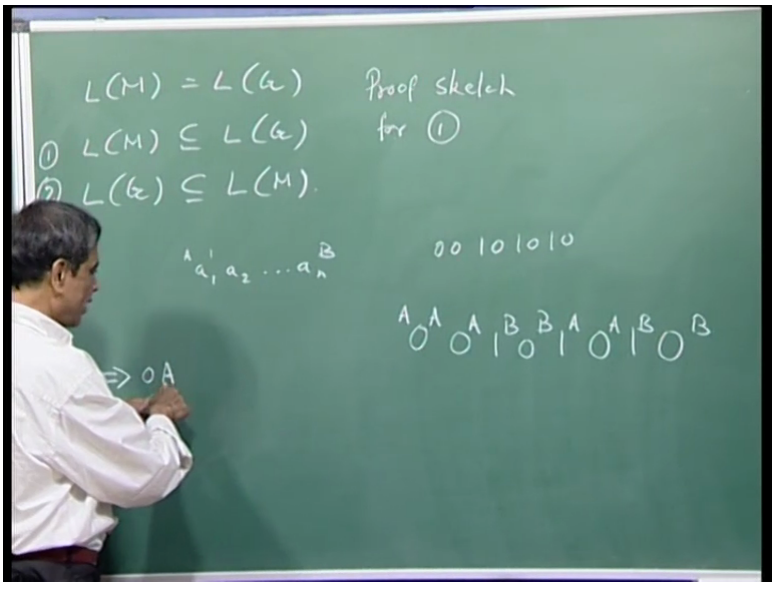
also be generated by the grammar G. And you just follow. Sothe derivation will go like this. A you know we will start because that is the start symbol here, A.

(Refer Slide Time: 36:32)



Now I see what happens here. 0 came and then (go) going to state A. So let me just write, this is allowed because from A I can use this 0 A. And thendo you see what is happening? A, this was the old state, symbol, new state, symbol, new state.

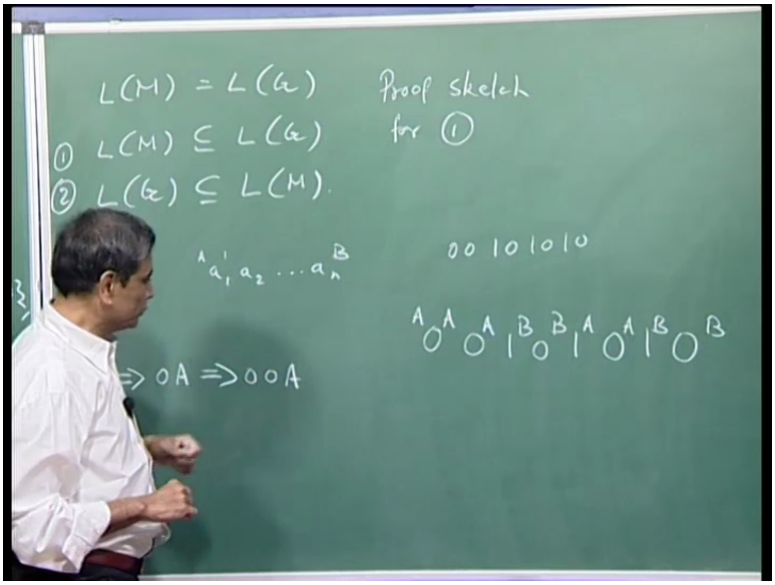
(Refer Slide Time: 37:04)



This is the old state where do you go? 0 A, right? But this is already there sonow you can see what is happening. Basically we are traversing this string and keeping track. This derivation

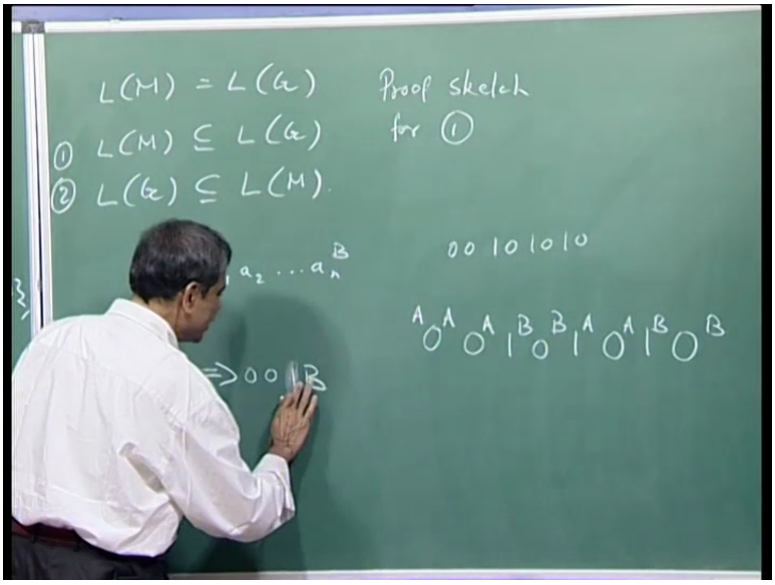
is keeping track in a way what is the state in which the machine would have been having scan the string 0 0.

(Refer Slide Time: 37:33)



It would be in state A. Now comes A, so from A on 1 the machine M goes to B andto capture that I have thatnonterminalA can generate one B. SoI rewrite this A as one B.

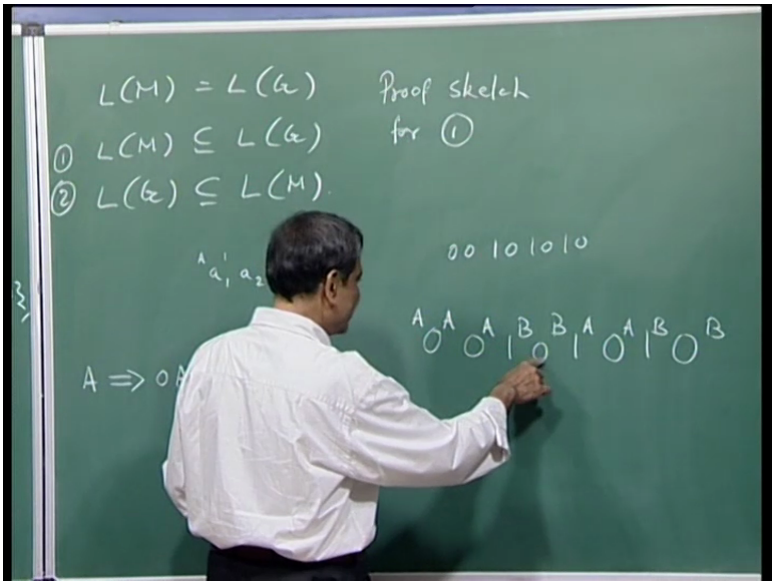
(Refer Slide Time: 37:59)



Again you can see the same invariant which is in our mind now as we do it, hold. And that invariant isyou know that partially if youcome up to a point thenwherever the state of the machine M is then the grammar also generates the first part of the string and that

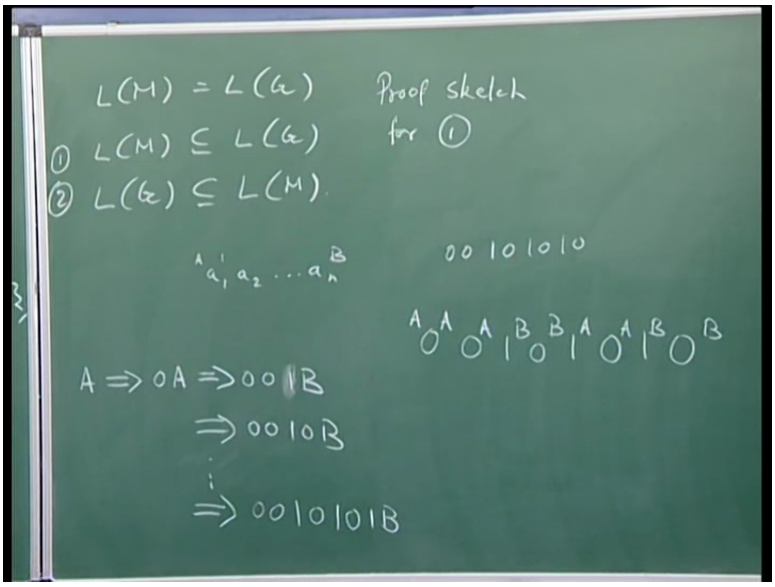
corresponding nonterminal. So this is how it goes. So 0 0 1 B and now 0 0 1 and this B remember that this is seeing 0 so it is 0 B.

(Refer Slide Time: 38:44)



Can you see what is happening? So this way it will just go on and when you are here, right, after scanning 0 0 1 0 1 0 1, easy to see the machine is in state B, right? And you will see thatour derivation will generate 1 0 1 and now the nonterminal at the end is B.

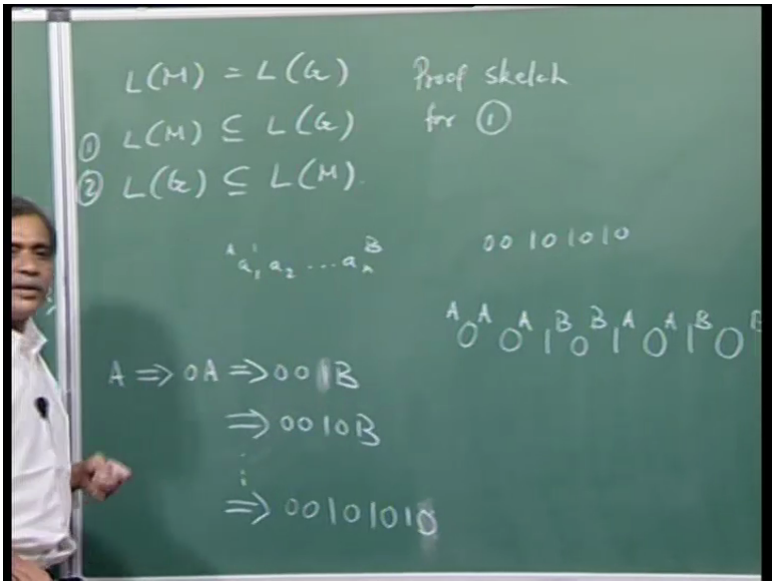
(Refer Slide Time: 39:28)



And you see what is happening is here from B on 0 we are going to B, just fineso far as the machine is concerned. But now the derivation must in from this(ru) B I will just derive. Since

B is a final state of the machine M and corresponding nonterminalB if you notice I have this production. So this B I will rewrite it asonly 0and then that derivation stops.

(Refer Slide Time: 40:08)

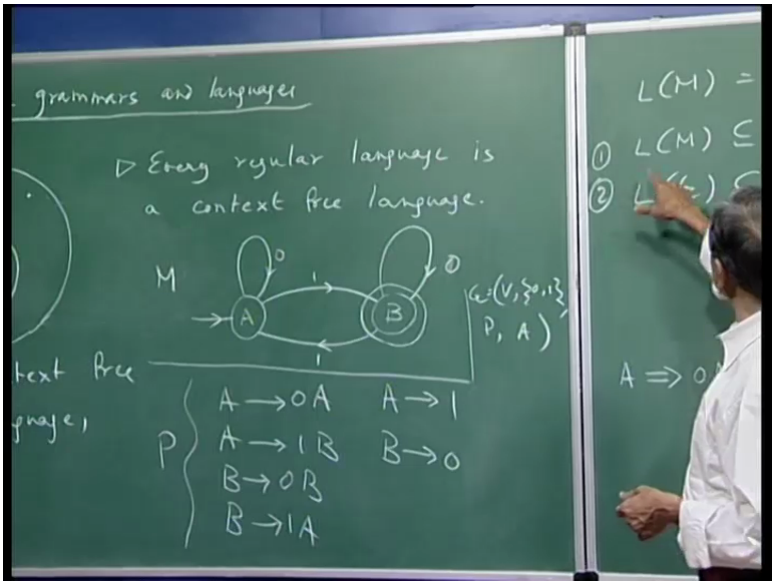


So basically this derivation also mimics the way the string is recognized.I meanas you present the string to the machine, the machine is going from state to state, right?

And after scanning some first initial part of the stringthis machine is in some state and what we are claiming is ourgenerative device this grammar starting from its start symbol, it would generate that prefix and thenlast symbol of that generation is a symbolnonterminal which corresponds to the state in which the machine would have been after generating that prefix.

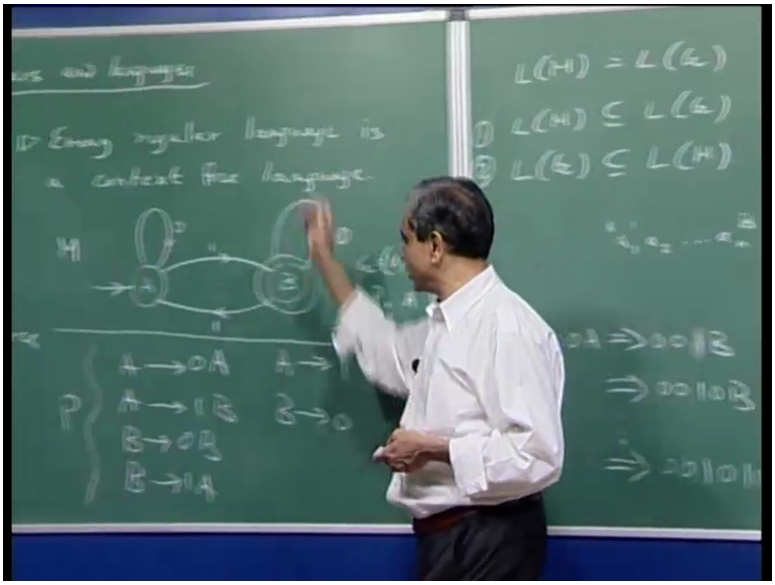
But we have to end somewhere. When we end, the final state iswe end in the final state so far as machine is concerned and that we would use a production like this to say that you know there is,as in this case no more nonterminalin the string that we have. So the generationsof this particular case stops, right? So we have a terminal string, right? I will not formally prove this. This is not too difficult to prove both these parts.

(Refer Slide Time: 41:47)



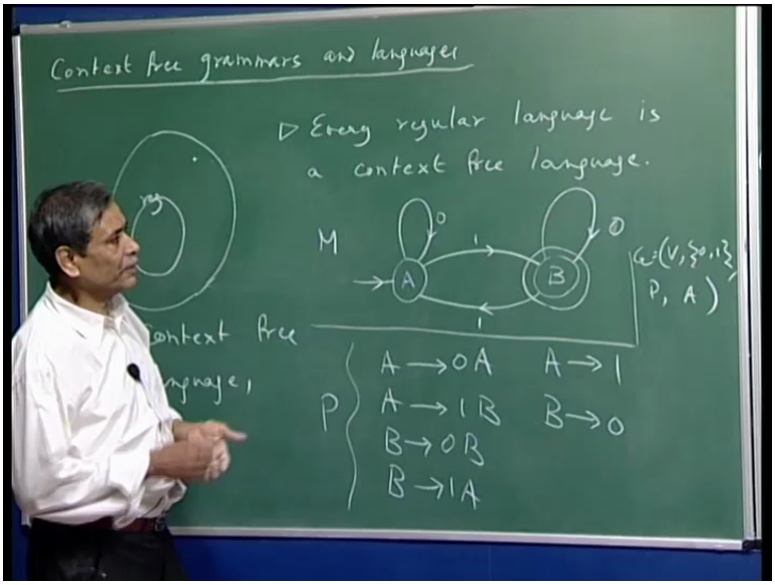
So Ikind of sketched thatevery string which is accepted by the machine M can also be generated by the grammar G and it is not too difficult using the same intuition that every string which is generated by the grammar G is also accepted by machine M for this particular example.

(Refer Slide Time: 42:09)



I will not prove that but let me indicate what I should do in general, right? This is the statement which we would like to show, which we would like to prove and I give you one example, right,given a regular language.In this example at least we know how we could get a context free grammar which generates the same language.

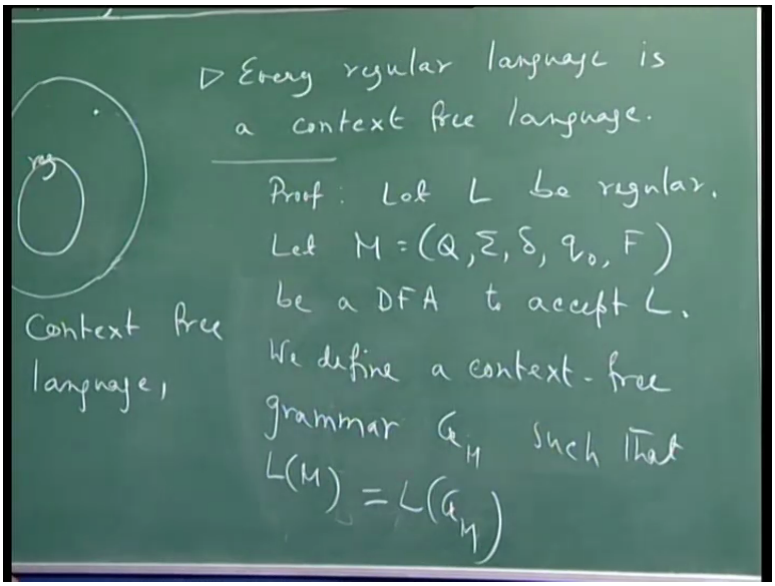
(Refer Slide Time: 42:42)



The way we can prove this statement is by taking a general regular language. So let L be regular.Since it is a regular let MQ sigma delta q 0 F be a DFA to accepted, okay. And what we will show? We will give you a construction that we define the context free grammarG and let me use this subscript M.

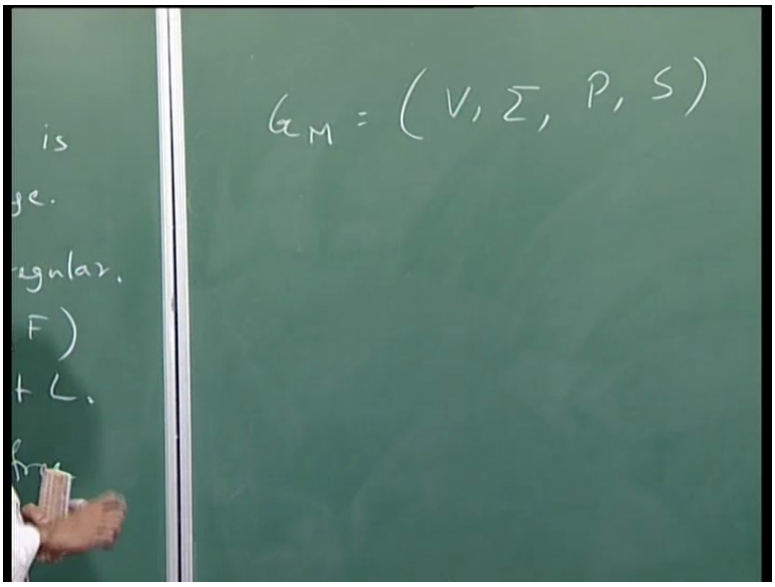
That means this grammar G is defined using the machine M from the definition of M such that the language accepted by the machine M which is of course L is same as the language generated by this context free grammar G subscript M.

(Refer Slide Time: 44:40)



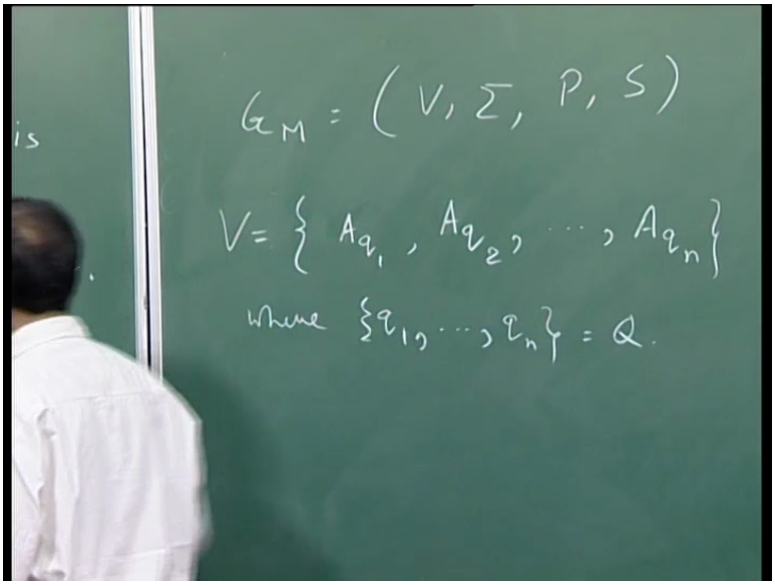
Let me show you the construction and the proof that it is indeed the case is something we can leave because the ideas are very simple for the proof. The construction is nice. So essentially it is the generalization of the earlier example. So this G M, remember it is a context free grammar soI need to define four components V, set of terminals. Notice already I have got one component. The set of terminals for the machine M is sigma and that is a set of terminals for our grammar also, P and S, okay.

(Refer Slide Time: 45:33)



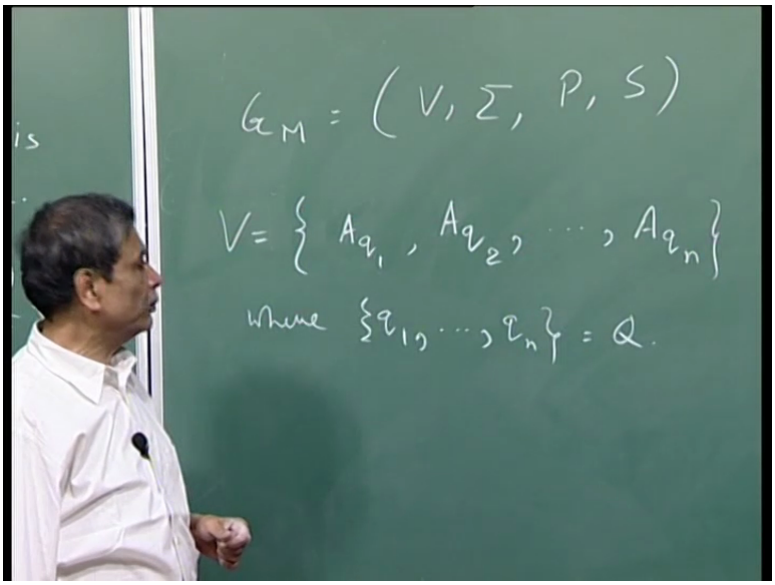
Now what is V, right?V is let mewrite it this way. V q 1, V q 2, right? Or instead of writing V let me just write maybe A q 1, A q 2, A q n. So it has n nonterminals.V is this set where q 1, q n. In fact let me just make it q 1. So the start state is q 1 itself, right? Equal to Q.

(Refer Slide Time: 46:45)



So that means that what I am trying to say is that suppose this machine has n states and I have named q 1 through q n, then for every such state I have anonterminal. Let which simple way ofstating the correspondence would be that you subscript anonterminal name with this state name.

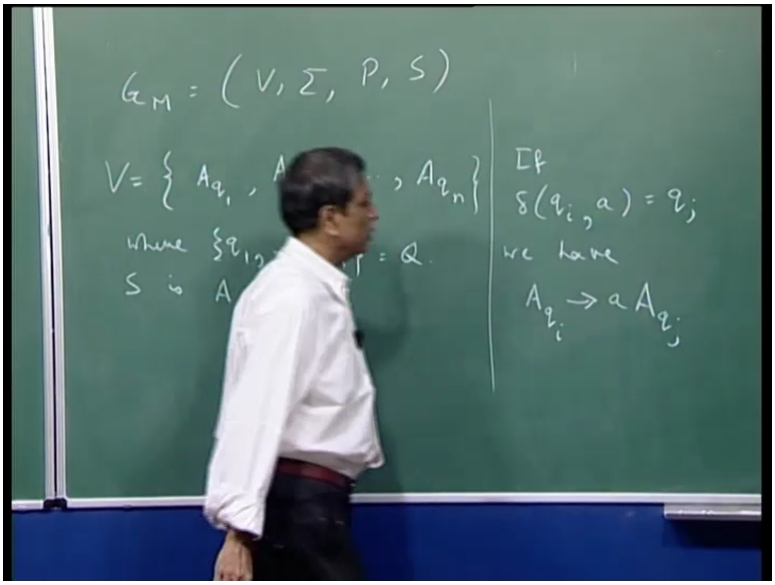
(Refer Slide Time: 47:13)



And S is actually A q 1. What is q 1? Q 1 was the start state of the machine M. And your start symbol of the nonterminal which is the start symbol of the grammar is a nonterminalwhich corresponds to the start state of the DFA. Now I have said what V is, what sigma is, what S is and so I need to say what P is,right? And we will go by what we said that if delta q i a, right, is q j, right?

Delta is the transition function for the DFA. If whenever I have such a thing therefor every state I will have such a thing. We at the productionhave, right, that A q ican bereplaced by a A q j.

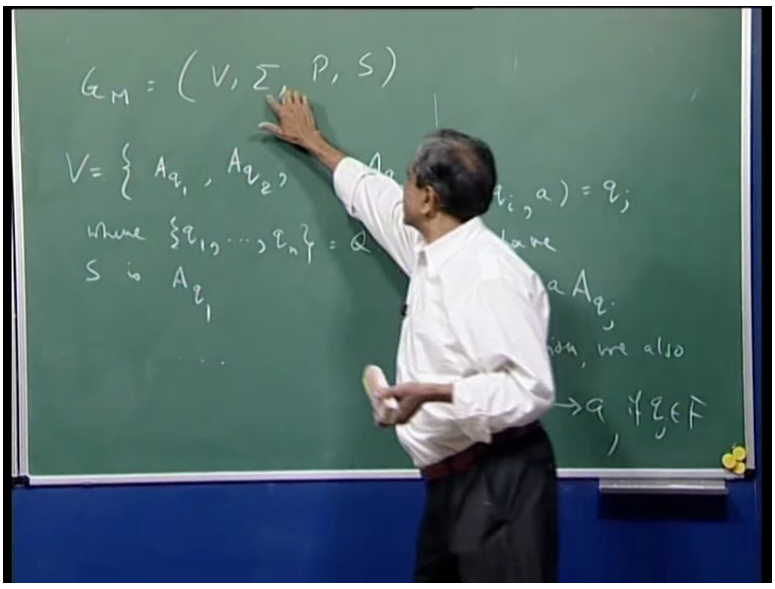
(Refer Slide Time: 48:46)



In addition we also have A q i goes only to a if q jis an element of the set of final states of the machine M. If yousee that is exactlyhow we define that example. Context free grammar is easy to see with these rules. What I have is of course is a context free grammar and the claim is that grammar precisely generates the language accepted by the machine M. Andthe intuition of the proof is again likethat you can consider either a derivation.

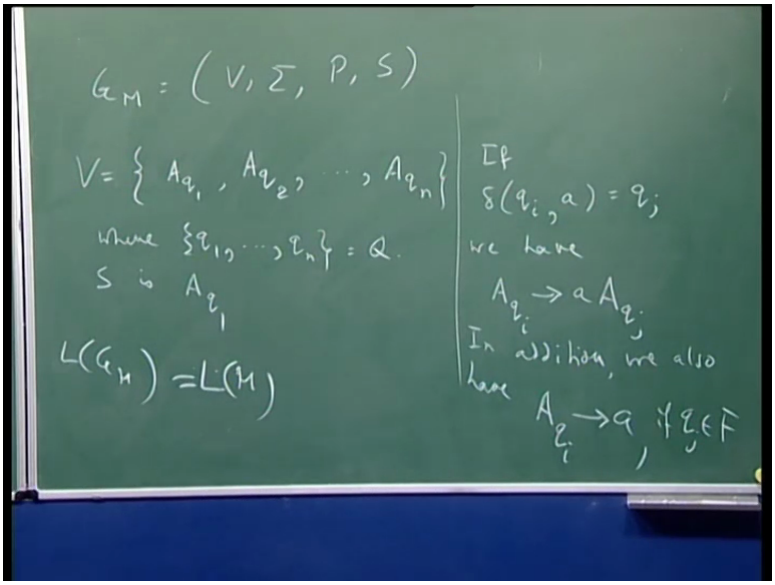
In this grammarthe derivation will be you know you will start A q 1 and you will keep generating strings where there will be anonterminal always at the end and finally we will replace that nonterminal by something. You know that by asymbolof this.

(Refer Slide Time: 50:21)



And that is precisely one way you can see that particular string would have been accepted by the machine M. The machine M would have gone through the same sequence of states as the sequence of nonterminals which appear at the end is in the derivation. I mean aswe had seen an example, okay. So it is not too difficult to prove that this construction, the grammar that we get is this L GM is precisely the language accepted by this machine M.

(Refer Slide Time: 51:07)



Once I prove that then I have proved that you give me any context free language. Ican ask you since L is a context free language.If you give me any regular language I can ask you thatgive me the DFA for that regular language (corres)which accepts that regular language.

Once I have that DFA, here is a constructionthrough which I get that and that grammar is precisely the language accepted by the DFA. And so therefore that regular language is also context free language and therefore we managed to prove this assertion.