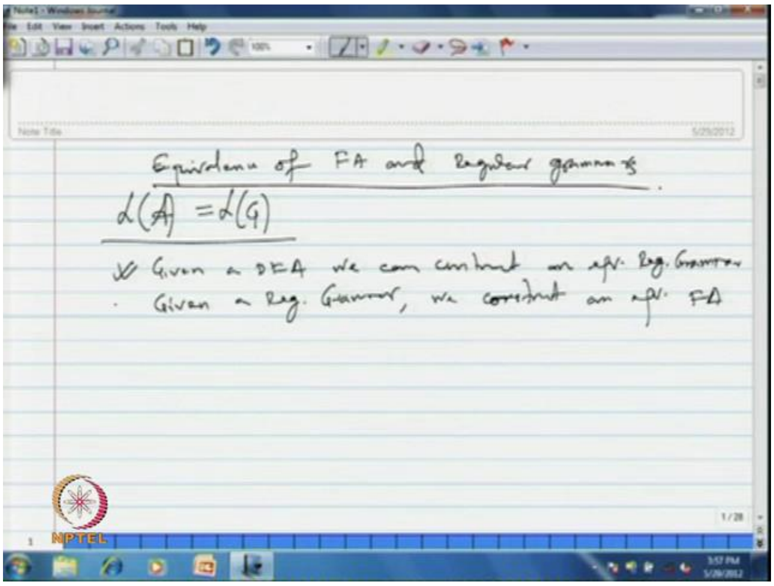
Formal Languages and Automata Theory
Prof. Diganta Goswami
Department of Computer and Engineering
Indian Institute of Technology, Guwahati

Module - 5
RL – RG - RF
Lecture - 3
FA – RG

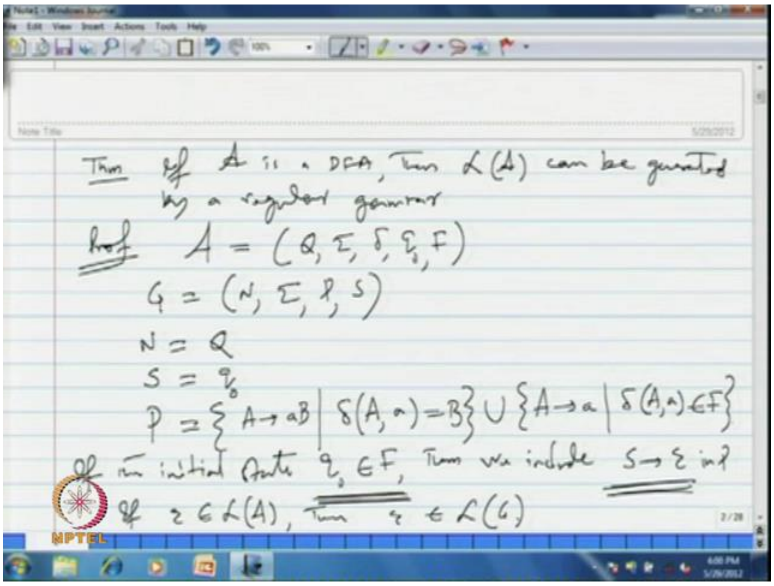
In today’s lecture we will show that final automata and regular grammars are equivalent.

(Refer Slide Time: 00:25)



That means equivalence of finite automata and regular grammars; we say that a finite automaton A is equivalent to a regular grammar G, if the language accepted by the finite automaton A is precisely the language generated by the regular grammar G. We say that automaton A and the grammar G are equivalent. In order to establish this equivalence we first prove that given a DFA; we can construct an equivalent regular grammar. And then for converse given a regular grammar, even a regular grammar we construct an equivalent finite automata. So, we prove that given a DFA you can construct equivalence regular grammar, and for a converse given a regular grammar we construct equivalent finite automata.

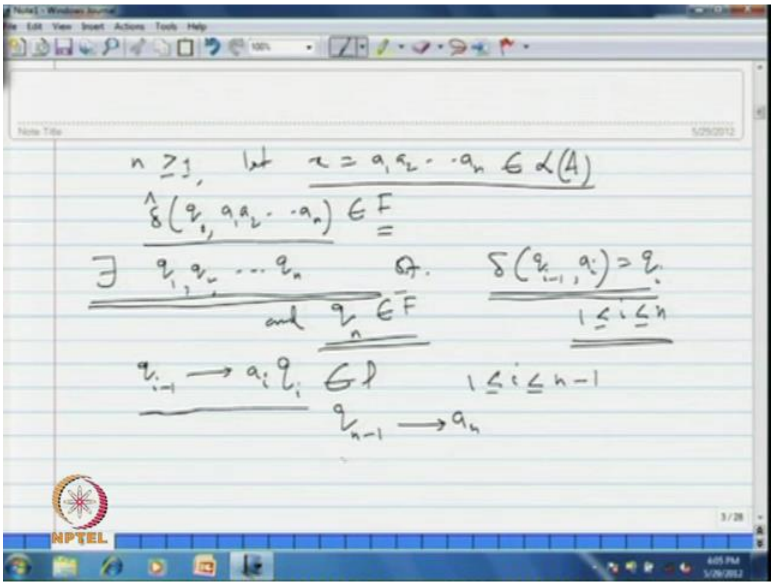
(Refer Slide Time: 02:12)



So, first we prove that if A is a DFA; then the language of the DFA can be generated by a regular grammar. So, we prove this by constructing a regular grammar for any given D F A. Suppose, A is the DFA contain the elements Q, sigma, delta, Q 0 and F. Now, we construct a regular grammar G, which is N, sigma, P, S. But the set of long term nose l N is exactly the set of state of the DFA, A; then the state symbol of the grammar S is nothing the state set of the DFA, A. And the set of production is of the form A goes to a B; capital letters indicate non terminals and smaller indicates terminal symbols from the alpha bet.

So, A goes to small a B is in the production set; if delta A, a capital A, small a where is a rate small a is a symbol is equal to B also we contains all those productions of firm A goes to capital A goes to small a; such that delta capital A small a belongs to F. So, these are constructing on that in use for a given grammar from the given D F A, A. In addition if the initial rate q 0 belongs to F; then we include the production S goes to epsilon n p. So, for q 0 being a final state; we include this particular production S goes to epsilon in the set of production for a grammar. Now, from the construction is quite clear that for this particular rule; we can show that if epsilon belongs to the language of the automaton A; then epsilon must also belong to the language of the grammar G. Similarly, if epsilon belongs to L of G; then epsilon mass also belongs to L of A.

(Refer Slide Time: 05:58)

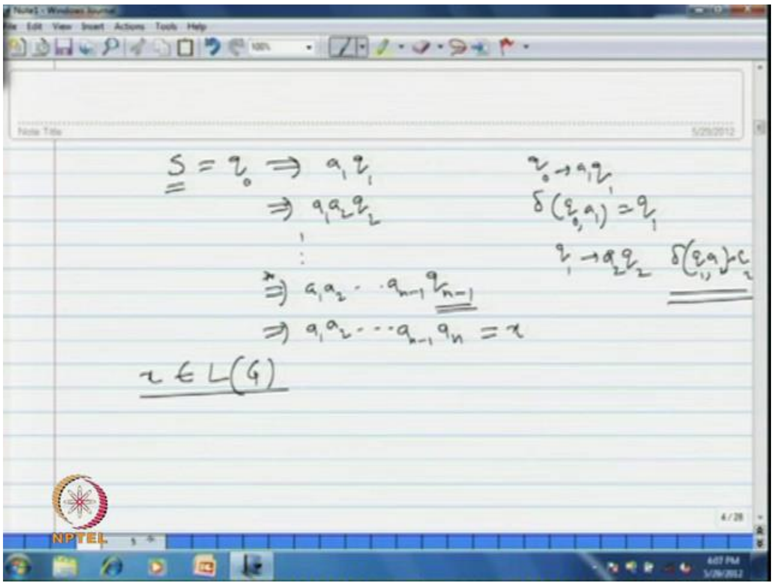


Now, from the construction since, we clear that if epsilon belongs to L of A; it mass also belongs to L of G and vice versa. Now, for any n greater than equal to 1; just consider the string x it is a 1, a 2 up to say a n; containing n symbols from the alphabet. So, x a string; so these string belongs to the language of the D F A is accepted by the D F A. So, where x is any arbitrary string; that means delta hat q 0, a 1, a 2 up to a n belongs to phi. So, up to processing the string starting a start set star state q 0; eventually arrive 1 of final states of the D F A F.

Now, this implies that there exist a sequence subsets q 1, q 2, up to say q n such that. So, there must exist some sets such that delta q i minus 1 a i equal to q i; for all i greater than equal to 1 and less than equal to n and q n belongs to F. So, if these strings to be accepted by the automaton show the process the string eventually arrive at the final stage. So, in such a case there must sequence of states q 1, q 2 up to q n such that delta q i minus 1 a i equal to q i; for all i greater than equal to 1 and less than equal to n and such that q n eventually belongs to F.

So, as per the construction of grammar; we must have q i minus 1 goes to a i, q i is a production of the grammar for all i greater than equal to 1 and less than equal to n minus 1 and q n minus 1 goes to a n. So, this is as per the construction of the grammar that helps us this write.

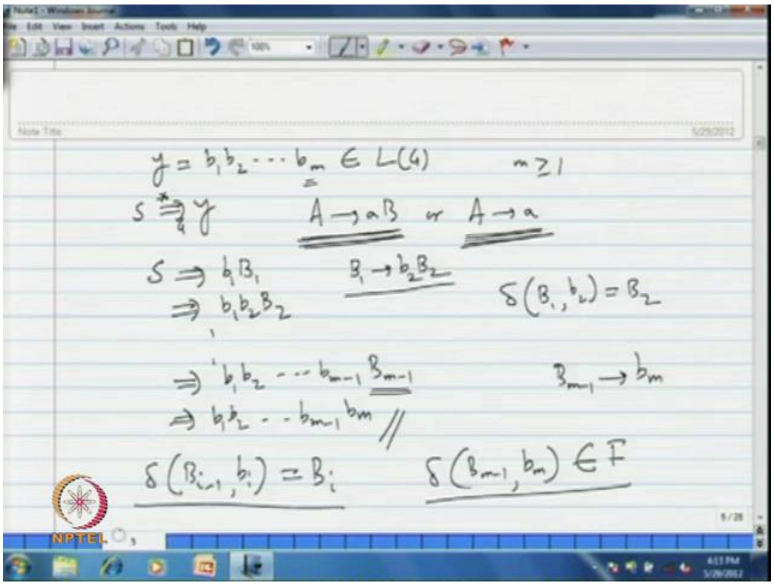
(Refer Slide Time: 08:38)



Now, using this production rules we can eventual derive the string x in g to derive x in g; which starts with the star small s which nothing but the star symbol initial state of the D F A which is q 0. So, this derives in 1 step a 1, q 1; because according to construction we have a production q 0 goes to a 1, q 1. Since, delta q 0, a 1 equal to q 1 must be applied to process a string starting in a star set q 0. So, this again in 1 step gives us a 1; since, q 1 goes to a 2, q 2 must also be a production.

Since, eventually delta q 1, a 2, q 2; must be a production, sorry. So, we removed in the D F A; so it is a 1, a 2, q 2 and so on. Eventually, we must have in if you steps a 1, a 2 up to say a n 1, q n minus. And, finally applying the production q n minus 1 goes to a n; will have a 1, a 2, a minus 1 a n. And, which is nothing but x the string x does x must belong to L of G. Since, we can derive this string x starting the star symbol of the grammar conversely to show that if x belongs to L of G; then x we can derive or the automaton A can accept the string x.

(Refer Slide Time: 10:40)



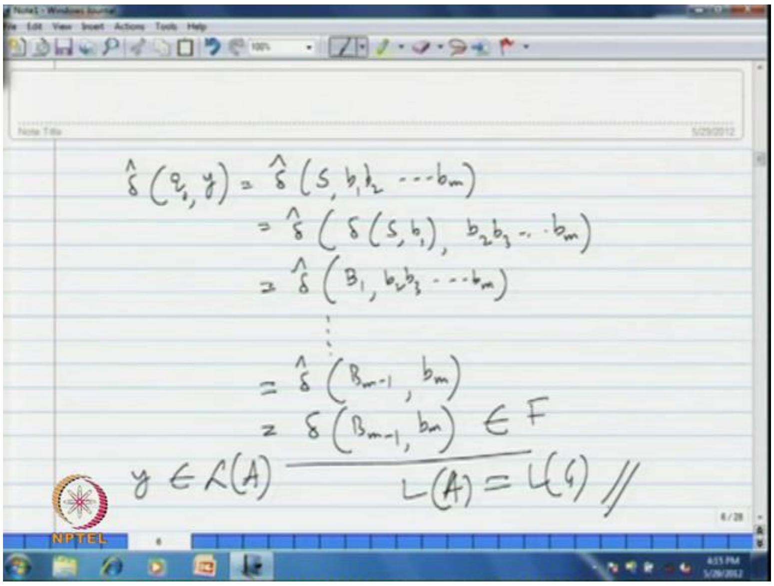
Suppose, the string y which is b 1, b 2 up to say b n is generated by the grammar L of G; some m greater than equal to 1. That is there must exist it derivation in 0 more state the star symbol. So, derive the string y in g the grammar G; which we are able to derive the string y starting a star symbol of the grammar. Now, since every production rule of G is of the form A goes to a B or A goes a; because regular grammar the derivation A goes to y has exactly anon steps.

And, A first m minus 1 steps; because the lengthier m numbers of symbols is m. So, the production the sequence must have exactly anon steps. And, a first m minus 1 steps or the production should use the rule of the form A goes small a B for small a is a terminal symbol. And, eventually in the final step we have to produce this kind of production; precisely, the as starting with the star symbol S. So, first of small b 1 capital B 1 the small b 1 is a terminal symbol from the sigma. And, b 1 is a non terminal; then b 1 goes to b 2, b 1 goes to some production of this form b 2, capital B 2.

In the next step we will get it to be b 1, b 2, B 2 and so on. Eventually, in m minus 1 step we will have b 1, b 2 up to b m minus 1; capital B m minus 1. In every step we have used a production of this form A goes to small a B. In the final step which is m th step this non terminal b m minus 1; can be substitute by a terminal by using this kind of production A goes to small a. That mean it is b 1, b 2 up to b m minus 1 and then this is b m.

Now, from this derivation and if you consider the construction of the grammar G; then see that in the automaton we must have for each production for every production of form say b 1 goes to b 2, B 2; we must have a move of kind. So, delta B 1, b 2 equal to B 2. Now, from the construction we know that delta b i minus 1, delta b n i minus 1 small b i must be equal to b i. According, to our construction, and also since we have use in last step the production B m minus 1 goes to small b m. Therefore, delta b m minus 1 b m must belong to final state it must be final state. So, it has production of this form and this must be a final state.

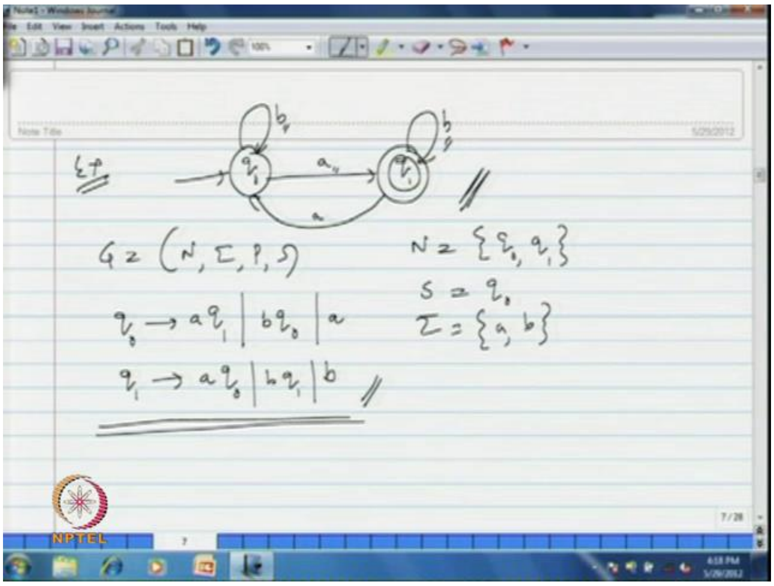
(Refer Slide Time: 14:28)



Now, using this set of transition; so what you can do? You can consider the processing the string starting in a star set q 0. The string y which is nothing but delta hat q 0 is nothing but s it is b 1, b 2 up to b m. So, if we consider it to be delta hat delta s, b 1 first to take the first symbol. And, then consider remaining string b 2, b 3 up to b m. So, using the first move delta s; v 1 is nothing but astical q 0. So, q 0, v 1 will get it to be b 1, b 2,b 3 up to b m; following this events into hotel gate it to be delta hat b m minus 1 b m.

But this is nothing not delta; since, the single state and single symbol and delta b m minus 1 you can replace this hat by simple transition b m. But these must belong to a function according to our construction. So, therefore y must also belong to the language of the automaton A. And, hence we have found it L of A is equal to L of G; so this proofs this theorem.

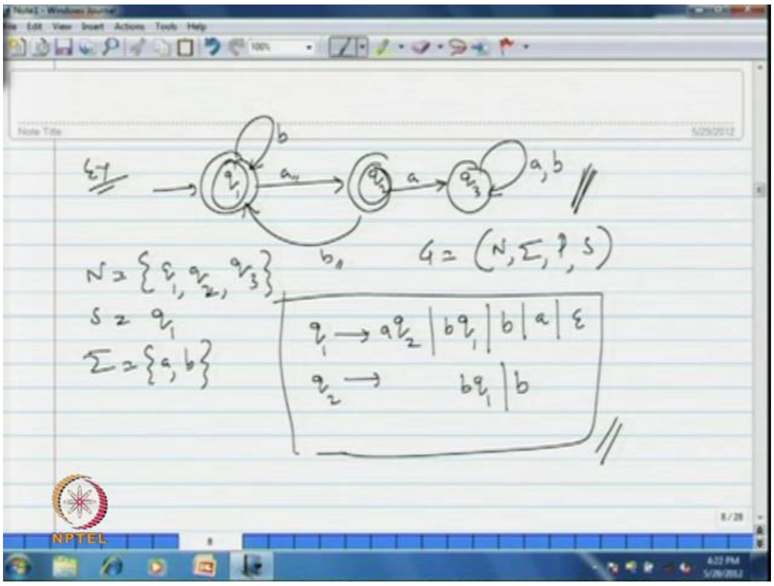
(Refer Slide Time: 16:10)



So, let us illustrate this theorem by given example; just consider the D F A containing 2 states and q 0 and q 1; where q 1 is your final state and q 0 is a initial state q 0 on a goes to the final state q 1. And, on b it remains in a same state b q 1 on b remains on the same state b and on a it comes back to the initial state. So, for this D FA; let us construct the equivalent regular grammar using our construction rules. So, the grammar G contains N sigma p s; where N is the set of states which is nothing but q 0 and q 1. And, a stars symbol s is nothing but q 0; sigma is obviously the 2 symbols a, b.

And, where a set of production tools can be found from the moves of the D F A like this; so q 0 since q 0 on a goes to q 1. So, q 0 a q 1 will be a production rule q 0, q 1, b goes to q 0; therefore, q 0 goes to b q 0 is also a production rule. And, since q 0 on a goes to a final state which is q 1; therefore, q 0 goes to a this also a production rule according to construction. Similarly, and these are possible for q 0; similarly, for q 1; if you consider all the transients all the moves q 1 on a goes to q 0. Therefore, there is a q 0 is a production rule q 1 on b goes to q 1 again. So, q 1 on b goes to q 1. And, since q 1 on b goes to q 1 and q 1 is your final state; therefore, q1 goes to b must also b a production. So, these are possible production that we have for this grammar. So, this is the equivalent regular grammar corresponding to this DFA.

(Refer Slide Time: 18:35)



Let, us consider another example say this is the D F A q 1 is a star state q 1 on a goes to q 2; where, q 2 is also a final state q 1 is star state and final state q two is a final state. And, q 1 on b remains the same state q 2 on a goes to state q 3. And, q 2 on b say goes to q 1 and q 3 on a and b remain in the same state a, b. So, therefore we can construct according to our considering the rules. An equivalent regular grammar say G containing the elements and sigma p s; where, N is set of states which is nothing but q 1, q 2, q 3, s is the star symbol which is the star state of the D FA which is q 1, sigma is the set containing a and b.

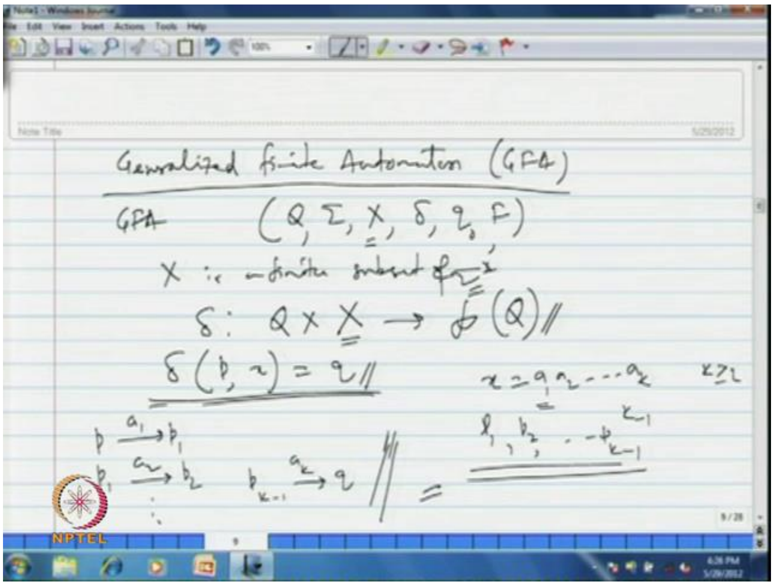
And, the set of production will be can be derived from all internal sum moves of the D F A say if you started q 1. So, q 1 on a goes to q 2; therefore, q 1 goes to a q 2 will be a production rule. Similarly, q 1 b goes to q 1; therefore, q 1, b q 1 must be a production rule. Again, since q 1 b goes to q 1 and q 1 is a final state; therefore, q 1 goes to b will also be a production rule. Similarly, q 1 a goes to q 2; where, q 2 is a final state; therefore, q 1 goes to a must also be a final state. And, since q 1 is the initial state and this also a final state; therefore, q 1 goes to epsilon is also a production rule according to our construction.

And, we have consider all possible; all the transition rules all the moves of the automata from state q 1 on every in 2 symbol, similarly on state q 2; if you consider all the moves q 2 on a goes to q 3. Therefore, it is a q 3, q 2 goes to a q 3 will be a production.

Similarly, q 2 on b goes to q 1; therefore, b q 1, q 2 goes to b q 1 must be a production. But q 2 on b goes to q 1; where, q 1 is a final state. Therefore, q 2 goes to b must also be a production according to our construction.

So, this are only moves from q 2 on every input symbol. Similarly, considering q 3 will find at q 3 goes to a q 3 and b q 3 are also production of this grammar. Of course, here q 3 is a trap state in the D F A. And, therefore we can remove or delete all the production rules involving q 3 without disturbing the language generated by the grammar; that means we can remove this production rules, and this production rule. And, the result in simplified grammar will be this 1 only. So, this is the equivalent regular grammar corresponding to this given D F A.

(Refer Slide Time: 23:22)



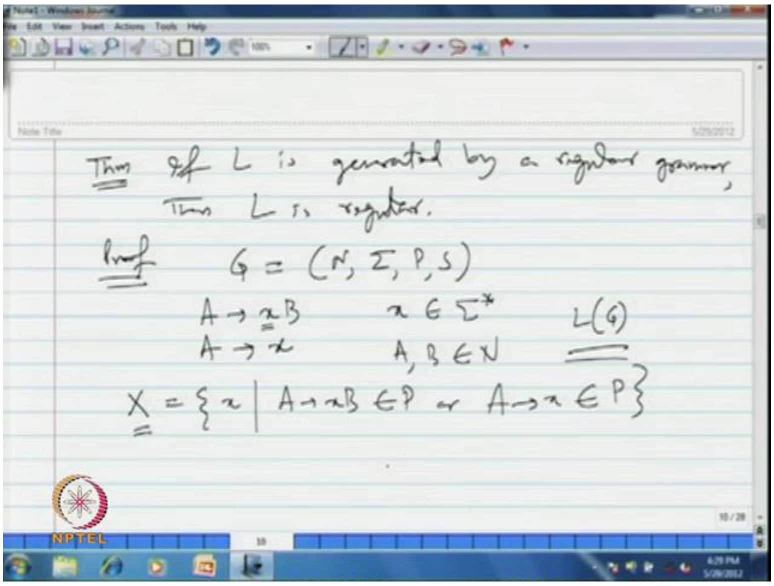
After swing that given any D F A we can construct an equivalent regular grammar; we now move to show that if L is generated by regular grammar. Then, L is regular; that means for L we can construct an equivalent D F A. To be full going to construct a D F A we will first introduce a concept of generalized finite automaton or simply G F A. So, generalized final automaton or G F A is a non final automaton in which the transition may be given via strings from a finite state instead of just symbols.

That is formally G F A is a stakes to pull containing 6 elements Q, sigma, X, delta, q 0, F. All other elements except for X are identical to a D F A; here, X is basically a subset from or finite subset. X is a finite subset of strings from sigma star, finite subset of sigma star. And, here the most are of the form delta the transition from delta is basically from Q cross X to the power set of Q. So, instead of a single symbol from sigma we take a string from sigma star. So, that is how we define a G F A.

So, even though it may be seems to be more powerful; but again so that G F A is no more powerful than an N F A. Instant given any move say delta p, x say equal to q. So, we have this kind transition or move in a G F A. So, we have a transition like this or move like this. We can always replace this transition by sequence of transition introducing a few in terminal state; such that if X equal to say a 1, a 2 up to say a k that means it is of length k for some k greater than equal to 2.

In such a case we will introduce k minus states in between which are say p 1, p 2 up to say p k minus 1. So that from p on a 1 on the first symbol it goes to say p 1; from p 1 on the second symbol a 2 it goes to p 2. And, like that eventually from p k minus 1 it goes to a k state q. So, this move delta p x equal to q can be replace by sequence of moves introducing a few finite number of states. And, we can show that or you can easily be proved the language of this final automaton N F A is equivalent or identical to the original G F A ok.

(Refer Slide Time: 27:25)

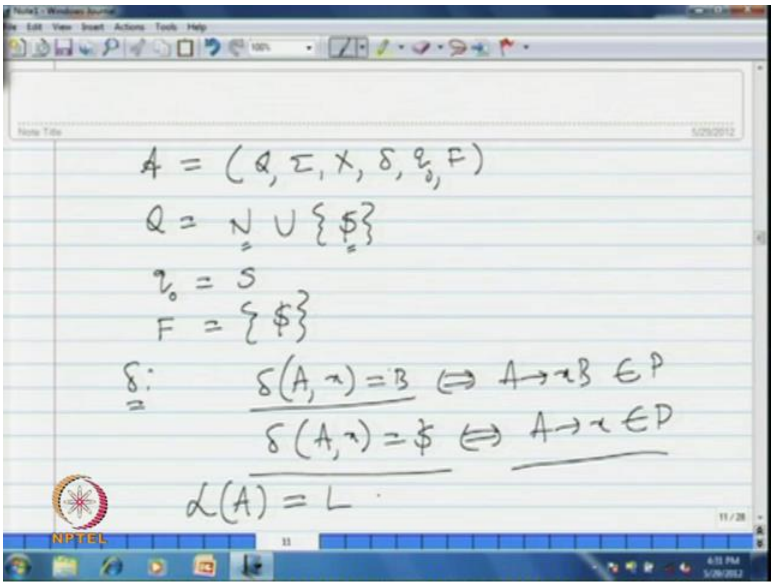


So, similarly all our transition can be replaced by a sequence of transition in the in an equivalent N F A. Hence, for every G F A we can construct on equivalent N F A removing the sequence; I mean transition on strings. Now, we show that if L is generated by a regular grammar then L is regular; that means we can construct an equivalent D F A or equivalent final automaton to accept the language generated by the grammar G. So, let us give you construct some to give a set of rules to construct an equivalent final automaton from the given regular grammar. Suppose, the given regular grammar is G containing the 4 triples, containing 4 elements and sigma, p, s.

Now, we know that since G is regular; every production is of the form A goes to a B. Sorry, A goes to x B; where x belongs to sigma star or it may be of the form A goes to x. So, x may be any string from sigma star. And, here A and B are non terminals it belongs to the set of non terminals. Now, what will do we will construct a G F A from these given grammar G. So, that it accept same language generated by the grammar; that means L G.

Now, what I will do let X be the set of strings get here under right hand side of the production. That means if this is a production; so this string X will belong to this state X. So, A goes to x; this will belong this set x that means x is set of all strings x such that A goes to x B belongs to the set of production or a goes to x belongs to the set of production in the grammar. So that how it define the set of strings x from sigma star. Now, since p is finite, p is a finite state this x is a finite subset of sigma star; so it must finite.

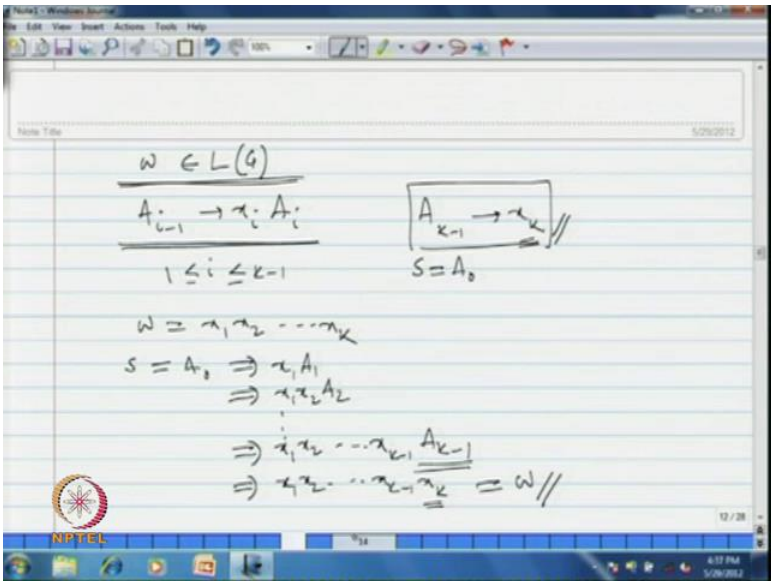
(Refer Slide Time: 30: 18)



Now, let us construct the G F A. So, A is a G F A containing the 6 elements is a state triple; where is set of states of the G F A, Q is basically the set of non terminals union; 1 special state say it is dollar. So, number of states will be equal to number of non terminals in the grammar plus 1 special state; which indicate why is dollar? And, a star state is nothing but a star symbol of the grammar. And, a final state, set of final state is nothing but the only state containing the symbol dollar. And, we define the transition function like this.

So, delta A x equal to B; if and only if A goes to x B is a production of the grammar. So, if A goes x B is production grammar; then we include delta x equal to b as a transition or a move in the G F A. Similarly, delta A, x equal to dollar; if and only if A goes to x is a production of the grammar. So, there are 2 rules that we have introduced to construct the moves of the G F A from the production of the grammar. Now, we claim that the language of the G F A, A is nothing but the language generated by the grammar G; that is why we have to proof.

(Refer Slide Time: 32:34)



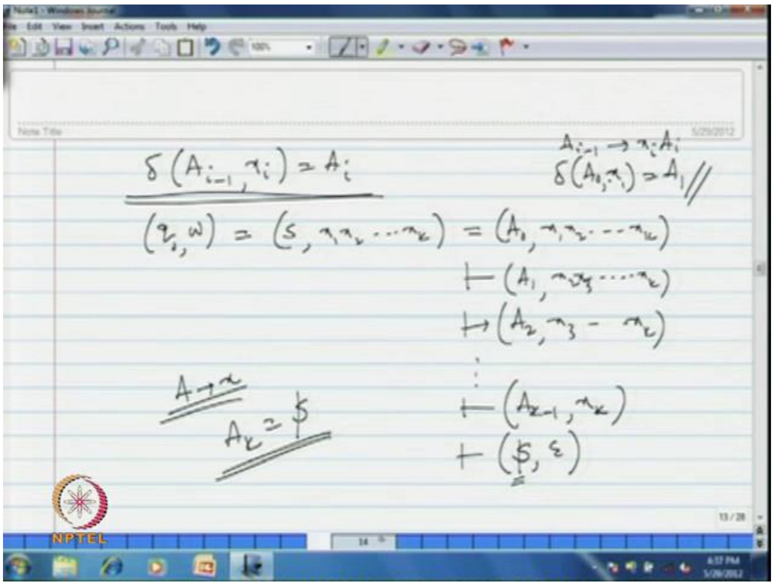
Say, let w belongs to L of G. So, since w is generated by the grammar G; therefore, there must derivation say which has suppose k steps. So, G must have a derivation which has k steps, which is obtain by following k minus 1 production rules; in the first k steps and eventually in the k step. So, in a first k steps will be using the production rule of the form

A minus 1 goes to x i, A i. And, the k step we must use a production of form say A k minus 1 goes to x k. So, if w belongs to L, G well; obviously, must have a derivation.

And, the derivation suppose if it has of line k, in the first k steps it must be we must use this kind of production rule; for i greater than equal to, less than equal to k minus 1. And, the final step we have to use to replace the last non terminal, we must use the production rule of the form a q minus 1 goes to s k. So, in this case of course, the star symbol S is A 0; does for w equal to x 1, x 2 up to say x k we can show that derivation like say S started S which is equal to A 0. In 1 step it must derive x 1, A 1, in the next step A 1 goes to x 2, A 2 has to be use. So, it is x 2, A 2 and so on.

Eventually, in k minus 1 step; we will have x 1, x 2 up to x k minus 1. And, in the k step we have to use replace this A k minus 1 by s k, by using this kind of rule x 1, x 2 up to x k minus 1; x k which is nothing but w. So, the derivation must be like this. So, if these are sequence of the derivation or sequence steps of the derivation.

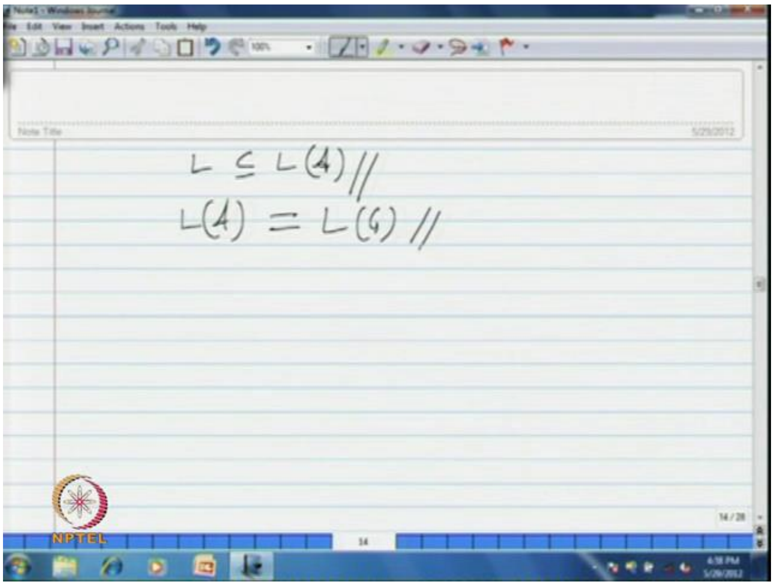
(Refer Slide Time: 35:26)



So, according to our construction of the G F A; we must have production rule of the form A i minus 1, x i equal to A i that means if we start at q 0 and process w we nothing but s x 1. Because q 0 is the star symbol of the grammar s, w is x 1, x 2 up to say x k. So, this will give since s is let us say is A 0, x 1, x 2 up to x k. So, in 1 step we will get it to be A 1, x 2, x 3 up to x k; because according to construction delta A 0, x 1 equal to A 1 is a transition or move in the G F A. Since, we have a move like this for every production of the form say A i minus 1 goes to x i, A i.

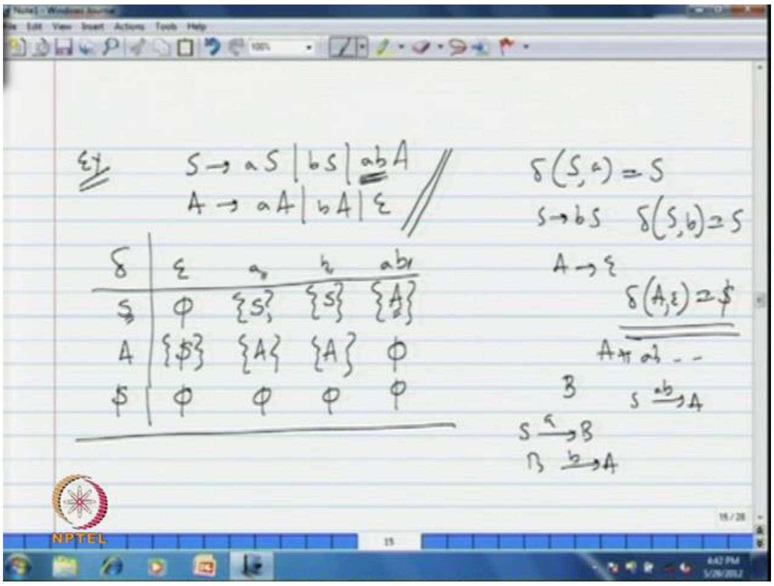
So, in a next accordingly what we will have from A1, x 2, delta, A1, x 2; it will get A 2, x 3 up to say x k. And, eventually in k minus 1 move we will have A k minus 1, x k. And, finally since in the derivation the final step we have used this production A k minus 1 goes to x k to replace A k minus 1 by x k. So, accordingly we must have A k minus 1 x k; because since this is a production of the form A goes to x. So, this must be a final state and eventually we will consume the whole string w. And, we will arrive at final state which is dollar; because A k is basically must be a final state according the construction.

(Refer Slide Time: 38:15)



Therefore, since we have consumed the whole string we know that the language L is a subset of L A see what we can showed a converse. So, converse can be shown exactly in a similar way therefore, eventually we can show that L of A equal to L of G.

(Refer Slide Time: 38:45)



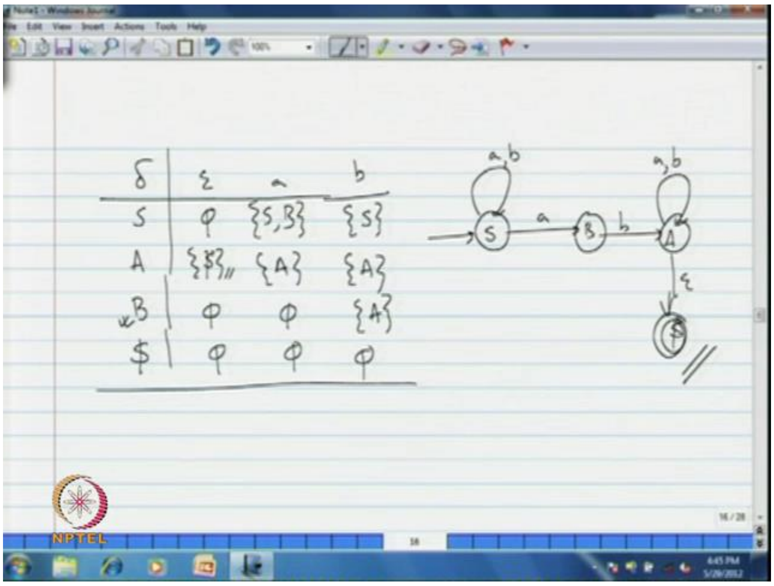
Now, let us explain the construction by giving 1 or 2 examples. And, the first example let us consider the grammar S goes to small a s, small b s, a b A. And, say A goes to small a b A and epsilon suppose we have 6 production in the grammar. So, we will first construct from this is a regular grammar it follows all the rules of a regular grammar. So, we will first construct a G F A; the transition map of the G F A is delta can we construct like this for epsilon symbol a, b. And, we are adjusting a, b in the right hand side of the production; those are only possible strings that we have a, b, a b and epsilon.

And, the states in the G F A will be the non terminals S A and the new non terminal dollar. So this S on epsilon there is no move on epsilon; so it if phi. So, Son small a it goes to S according to our construction; because S goes to a, S is a production in a grammar. Therefore, delta S, a equal to S; we have a move like this in the G F A. Similarly, since S goes to b, S is a production, delta S, b equal to S is a move. Therefore, delta S on v goes to S itself; similarly, S on A, b goes to state A.

Following the same rules A on epsilon there is no move; sorry, there is a move A on epsilon A goes to epsilon is a production that we have, A goes to epsilon is production. Therefore, A goes to dollar must be included. So, it means delta A epsilon equal to dollar must be move according to our construction that we have or reintroduced. Then, A on a small a it remains in the same state a, A on b again. Since, A goes to b is a production it remains on the same state a. And, on a b there is no move; because there is no production of the form A goes to a b something. Therefore, this is fine there is phi no move and from dollar we are give no move.

So, there is just generalized final automata that we have. Now, we have only 1 transition from the state S to the state A on the string a, b. All the transition are on symbol a i do not symbol a or b. So, the only transition on some string is for A string a b and that is from star set S to the state A. We can remove this by introducing a new step and used to say it is B. So, from S you can go to B on symbol a and from B you can go to state A on symbol b; to replace the move from S to state A on the string a b.

(Refer Slide Time: 42:55)

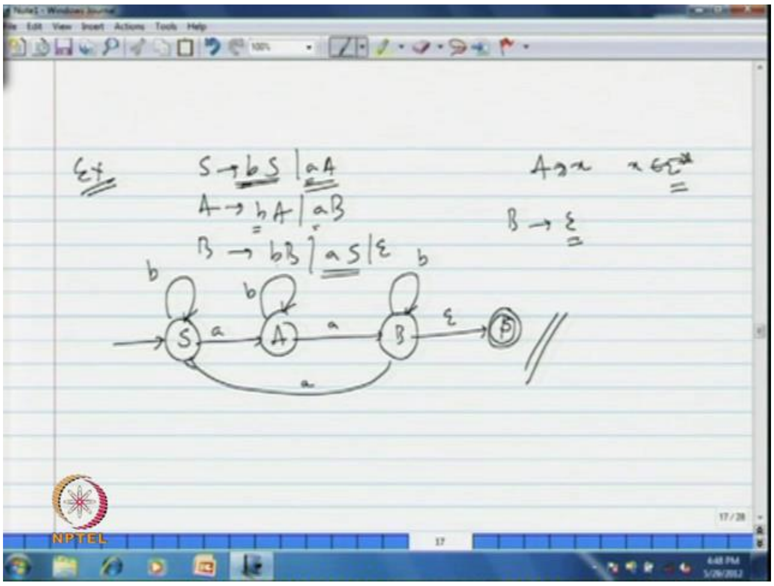


So, introducing this modification will get the new transition map delta, epsilon, a and b. S on epsilon goes to phi and S on a goes to initially it went to star set S now it will also go to b. Therefore, it will be S and B and S on B it will remain on the same state and on a b we have already removed. So, S on A, on epsilon it goes to the final state dollar and on A there is no change, on b there is no change I just copied it. Then, have introduced a new non terminal B on small b goes to A; because according to this move B on small b goes to A. And, there is no other move from B on epsilon or on the symbol A.

Similarly, from S, from dollar there is no transition. So, these are transition map for a given finite automata; that means if we draw the transition diagram start with S there is a start symbol. So, S on a it may go to S or it may go to B. And, then S on b will go to S again; therefore, S on a and b may to may go to S, b on B goes to A, b on small b goes to

A, A on a and b goes to a itself. And, then and on epsilon goes to dollar which is a final state; of course, which on epsilon goes to dollar and dollar does not go this state, does not go to anywhere on any of d input symbol. So, these are required transition diagram for a given regular grammar.

(Refer Slide Time: 46:16)



Let us consider another example say S goes to b, S, a, A; A goes to b, A say a, B and B goes to b, B, a S. Now, here we will find it there is no need production, which contains a string; all the production there is no inside of the production, there is no string involved all are symbol. Therefore, it is easy to draw the transition diagram directly. So, the star set will be the star symbol of the grammar this is S. So, S on b from this production S goes to b, S, S on b remains on the same state. And, S on a small a it goes to the state A from this production S goes to a. Now, A goes to b, A for this we have again S, a goes to b, A; we have a self look. So, whenever A goes a, B; on A it goes to the state B**.**

Then, B on small b goes to b definitely b, A self look on b and b on a goes to A, s from this production B goes a, S, B on a goes to S. And, eventually of course, we need there is another production say epsilon. So, B on epsilon goes to final state; because this is the only production of the kind of the form A goes a goes to x; where x belongs c master. Since, B goes to epsilon; epsilon is a string from sigma star. So, this must be a final state; therefore, B on epsilon goes to dollar. So this is the required fine automata which equivalent to reach regular grammar.