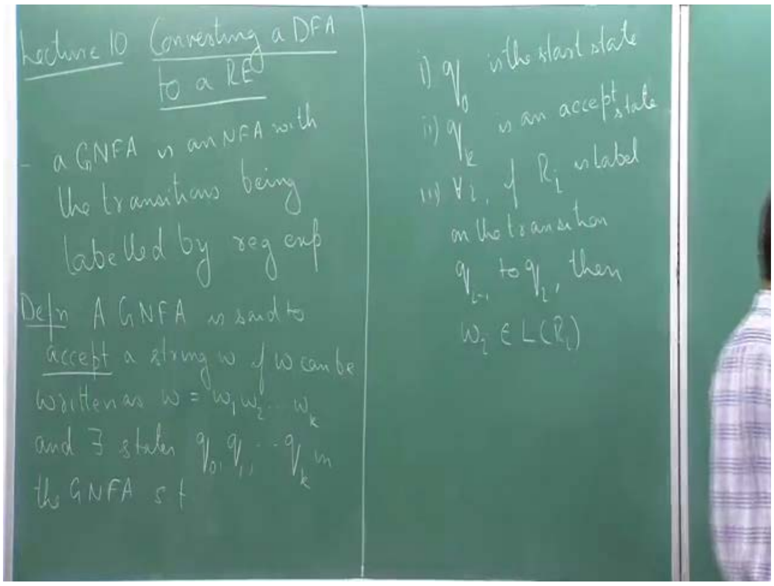
Theory of Computation
Prof. Raghunath Tewari
Department of Computer Science and Engineering
Indian Institute of Technology, Kanpur

Lecture – 10
GNFA to RE Conversion

Welcome to the 10th lecture of this course. Today, we will see how to convert deterministic finite automaton to a regular expression.

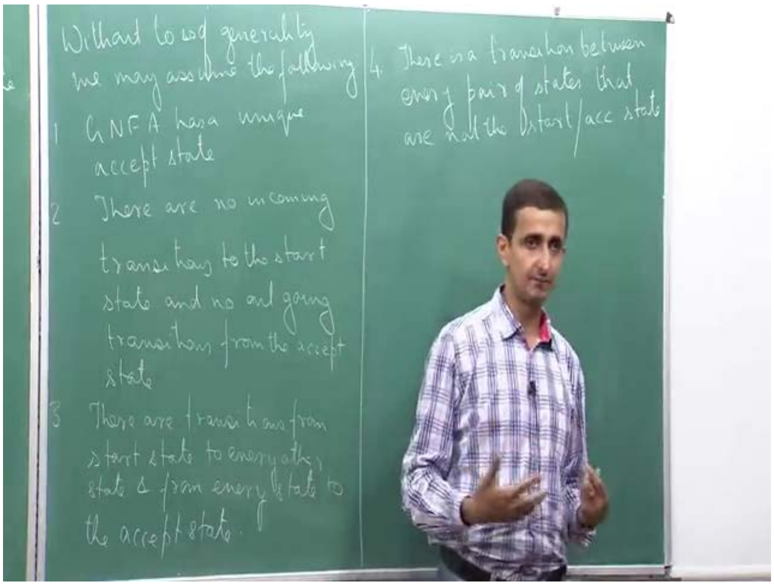
(Refer Slide Time: 00:39)



Last time, at the end of the last lecture, we introduced what a generalized non-deterministic finite automaton is. Let me give a brief recap of what it was. So, generalized non-deterministic finite automaton or GNFA in short is an NFA with the transitions being labeled by regular expressions. So, we saw informally what it means for an NFA to accept a string, but let us look at a formal definition of when we say that a string is accepted by an NFA. Let w be a string, so a GNFA is said to accept a string w, if w can be written as w equals w 1, w 2 up to some w k. So, w can be written as a concatenation of k strings; and there exist states q 0, q 1 up to q k in the GNFA.

Such that firstly, q 0 is the start state, q k is an accept state. And finally, for all i, if R i is the label on the transition q i minus 1 to q i, then w i belongs to L of R i. For example, if I have from q 0 to q 1, if I have a label if I have the label as the regular expression R 1 and w 1 must belong to L of R 1. Then from q 1 to q 2 if I have a regular expression R 2 then w 2 must belong to L of R 2 and so on. So, basically I can partition a string into k parts such that each part is accepted, or each part belongs to the language of the corresponding regular expression. So, if that happens we say that the GNFA accepts the string w. So, without any loss of generality, we can assume that a GNFA has the following properties.

(Refer Slide Time: 05:26)



So, without loss of generality, we may assume the following. So, firstly, we may assume that the GNFA has a unique accept state, because if not, if there are multiple accept state, I can always add epsilon transition from the old accept state to a new accept state. Secondly, there are no incoming transitions to the start state, and no outgoing transitions from the accept state. Again why can we assume this? Suppose this is not true suppose there are some incoming transitions to the start state of my GNFA.

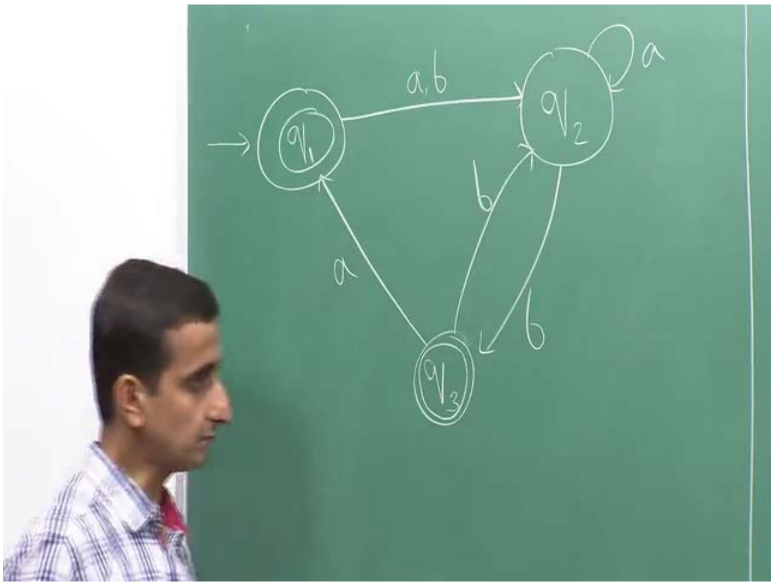
What I can do is that I can add a new start state, let us call it q 0 prime. And what I will do is that, from q 0 prime I will add a epsilon transition to my old start state that is a q 0, so that will ensure that there are no incoming transitions to q 0 prime, and the new GNFA accepts the same language as the old GNFA. And the same thing I can do with the accept state as well, similar thing.

The third assumption is that there are transitions from start state to every other state; and from every state to the accept state of my GNFA. So, here we make use of the fact that I can put the label phi on a transition. So, suppose from the start state, there is no transition to some state, some state let say q. What I can do is that I can put a transition from the start state to q, and I can label it as phi, which means that I cannot take that transition basically to construct a string. So, and the same thing I can do from every state to the accept state.

And the final point is what I will assume is that there is a transition between every pair of states that are not the start of accept state. So, basically take any two states in your GNFA that are not the start or accept state; we will assume that there is a transition between them. Once again if there is no transition, I will put a transition with the label phi on it, so that is the idea. So, basically this is a way to take arbitrary GNFA, and convert GNFA to, a GNFA which has the following list of properties.

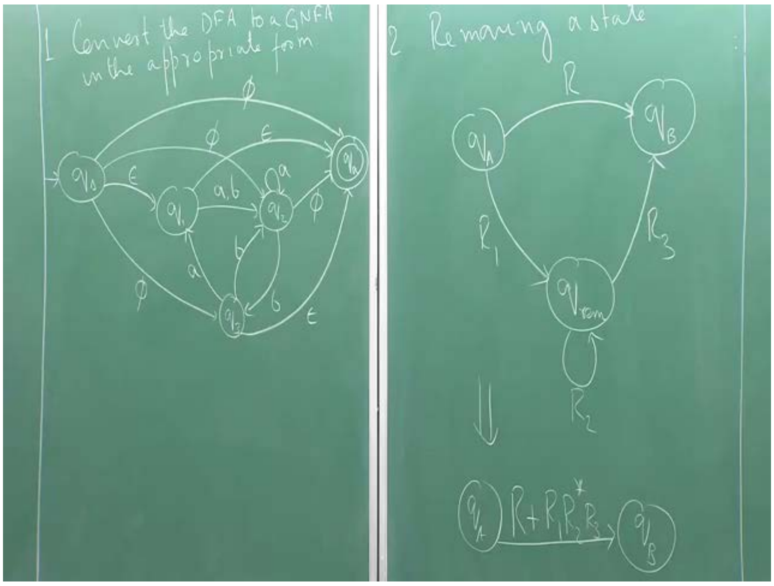
Now, I think we are in a position to describe how to convert a DFA to a regular expression. And the way we will convert is in a constructive manner; we will actually give a algorithm which takes a DFA as input and it produces a regular expression. So, before I describe the algorithm in a formal way, I will actually motivate the algorithm by giving an example, in fact, we will do both parallelly. So, we will solve the example as well as we will write the algorithm side-by-side. So, we will form the algorithm simultaneously.

(Refer Slide Time: 11:46)



Let us start by looking at a DFA. Let say - we have a DFA which has three states. So, we have a start state, I will call it q 1, my start state is also my accept state. From q 1, I go to q 2 on both a and b. From q 2, I go to q 3 on a b; and I stay at q 2 on a a. From q 3, I go to q 1 on a a, and I go to q 2 on a b. And let us assume that this DFA has so this is a DFA and this DFA has 2 accept states q 1 and q 3.

(Refer Slide Time: 13:07)



The first step of the algorithm is to convert the DFA to a GNFA in the appropriate form. So, we will convert this DFA into a GNFA in the form that I described earlier. So, what do we do? So, first I add a start state, I will call it q start; I add a global accept state, I will call it q a; and I have my q 1, q 2 and q 3 that is present. Let me draw the transitions for between q 1, q 2 and q 3, this is what was given to us.

Now we need to complete the GNFA. So, from the start state, I draw epsilon transition to the start state of my DFA; and to every other state, I will put a transition with the label phi. So, from q s to q a, I has transition labeled phi; from q s to q 2, I have a transition labeled phi; from q s to q 3, I have a transition labeled phi. Similarly, I have a transition from the accept states of the DFA to q a. So, from q 1, I have a transition to q a with label epsilon; from q 3, I have a transition to q a labeled epsilon; and from q 2, I have a transition to q a with label phi, because q 2 was not an accept state.

So, I have these transitions. Now there are also transitions between these states. So, I am just due to space constraint, I am not putting in all the transitions that are there, but so this is has a label b. So, I am not putting in all the transition. So, wherever we have a missing transition that basically corresponds to a transition with a label phi on it. So, I am just ignoring for the time being.

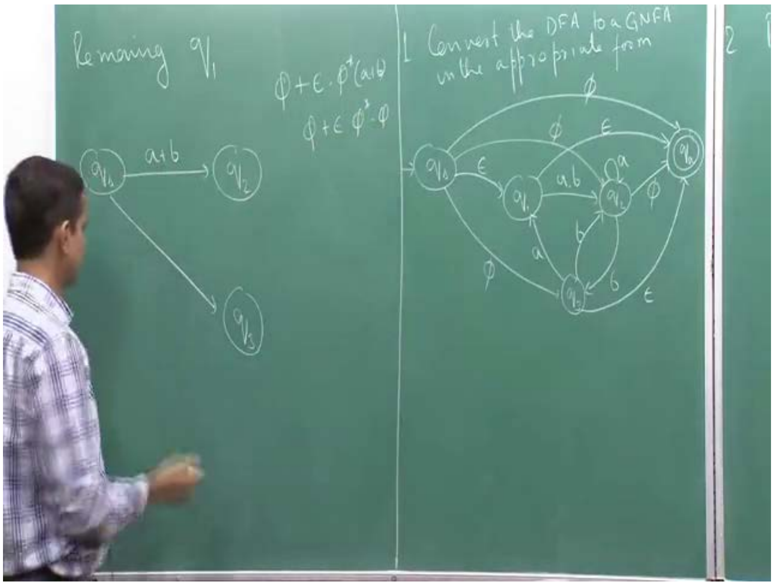
The idea of the algorithm is to remove each internal state of this GNFA one state at a time. So, the internal states are the states which are not the start state and the accept state. So, I will keep the start state and the accept state as they are. But for the other state, so other states in this GNFA are q 1, q 2, and q 3 in each iteration of my algorithm, I will remove one state from this GNFA and construct another GNFA which has the property that whatever language is accepted by this entire GNFA is the same language that is accepted by my new GNFA. And which state I remove in a particular iteration is not important I can remove any state.

For example, I can choose to remove q 1, then q 2, then q 3 or whatever order you want, it does not change the answer. And then finally, when I have removed all the internal states that is when I have removed all q 1, q 2 and q 3. Whatever is the transition that is going from q s to q a, I look at the label on that transition whatever is the regular expression labeling the transition is the regular expression that corresponds to my input DFA, so that is the regular expression that I want to construct. So, that is the entire process.

Let us do this. So, the important point is that how do we remove a particular transition. Let say that or how do we remove a state. So, removing a state, so let say that I have a state some may be q 1 or let me call it q A. I have another state q B. And I want to remove a state q let me call it rem, that I want to remove. Such that there is a there is some transition from q A to q rem, there is some transition from q rem to q B, so this has a regular expression R 1 on it this has a regular expression R 3 on it. Maybe there is a transition to q rem to q rem itself with R 2 on it; and may be there is a transition from q A to q B with that label R on it. So, suppose if I want to remove a state q rem, I want to answer what will be the new label on this transition from q A to q B, if I remove this, what will be the new label.

So, the idea is that if I remove this then in the next step, I have q A I have q B, I have a transition from q A to q B, and this transition will have the label R plus R 1 which is from q A to q rem, R 2 star that is the label on the self loop on q rem concatenated with R 3 that is the label on q rem to q B. So, this is the entire algorithm. So, this allows us to remove the state q rem when I do this for each thing.

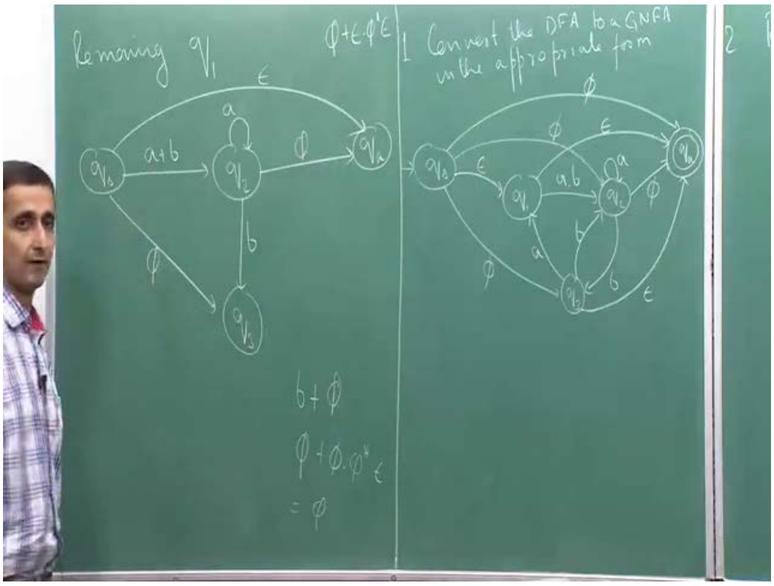
(Refer Slide Time: 20:18)



Now, let us try to see, so in fact let me do it here that will be closer. So, removing let say I want to remove the state q 1, what will be the transitions? So, first I have q s, and then I have let say q 2. So, what will be the new label on q s to q 2? Let us look at this GNFA. So, if I remove q 1, the new label on q 2 will be the old label on q s to q 2 that is phi plus the label epsilon concatenated with the self loop on q 1 star concatenated with the label from q 1 to q 2. So, what is the self loop on q 1? There is no self loop on q 1; and I said if there is no edge then it means it is phi. We have seen earlier that phi star is nothing but 1.

Basically the label from q s to q 1, if I just write it down here will be phi plus q s to q 1 is epsilon concatenated with phi star this is the label on the self loop on q 1 concatenated with the label from q 1 to q 2. So, the label q 1 to q 2 is nothing but basically a plus b. So, the label from q s to q 2 is if I just simplify this, this is nothing but a plus b. So, epsilon dot epsilon dot a plus b. Similarly, from q s to q 3, I have phi plus epsilon dot phi star which is epsilon dot phi, because I have no edge going from q 1 to q 3, it means that it is so, in this case it will be phi plus epsilon dot phi star dot phi and phi dot anything is just phi. So, the label from q s to q 3 is basically phi.

(Refer Slide Time: 22:53)

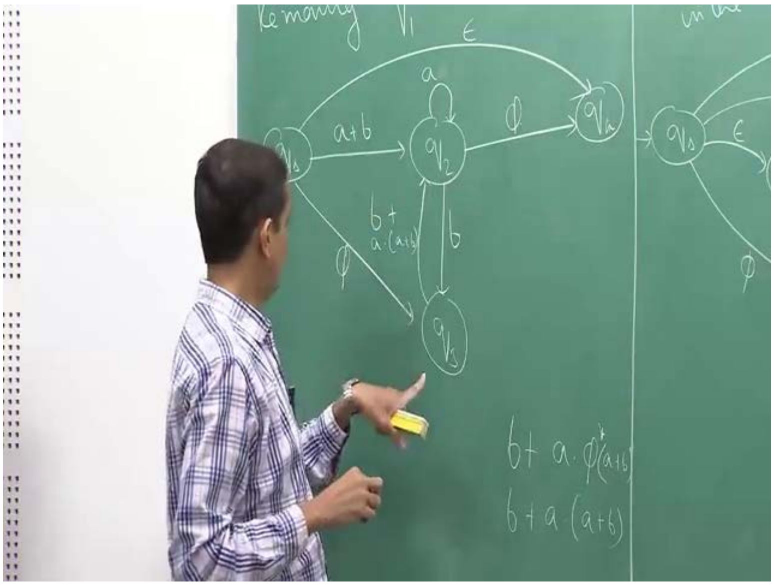


What about q s to q accept? So, I will erase this. Let me put q accept over here. So, q s to q accept will have a label. So, I have phi. So, I have phi over here plus q s to q 1 is epsilon. So, I have epsilon dot phi star dot epsilon. So, from q 1 to q a, I have - a label epsilon, so it will be epsilon. So, what does this simplify to? So this simplifies to just epsilon. So, this gives me the transitions from q s to these 3 states.

Now let us look at these states, let us look at a q 2 from q 2 to itself I have to put an edge. So, from q 2 to itself already I have a, so I will have a plus something what is that something. So, q 2 to q 1, but I do not have anything from q 2 to q 1. So, it is phi, and phi dot anything will anyway be phi. So, it is a plus phi which means that I have only a.

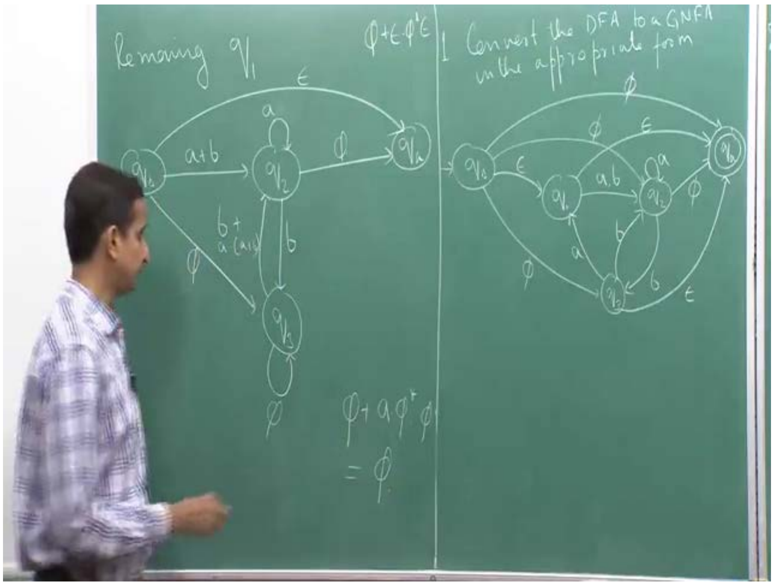
Now q 2 to q 3, what will I have. So, from q 2 to q 3, earlier I had b. Let me work out over here. So, I have b from earlier. Now, q 2 to q 1 is phi. So, I do not have anything. So, anyway I get phi. So, this will just be b. And finally, from q 2 to q a, I have phi from earlier. So, I have phi from q 2 to q 1, I do not have anything, so it is phi. So, it is just phi plus phi dot whatever phi dot phi star dot q 1 to q a is epsilon, but this is nothing but phi. So, here also I have phi. So, q 2 is done. Now the only state remaining is q 3. So, q 3 to q 2, I have to check. So, what do I what will be the label on q 3 to q 2.

(Refer Slide Time: 25:36)



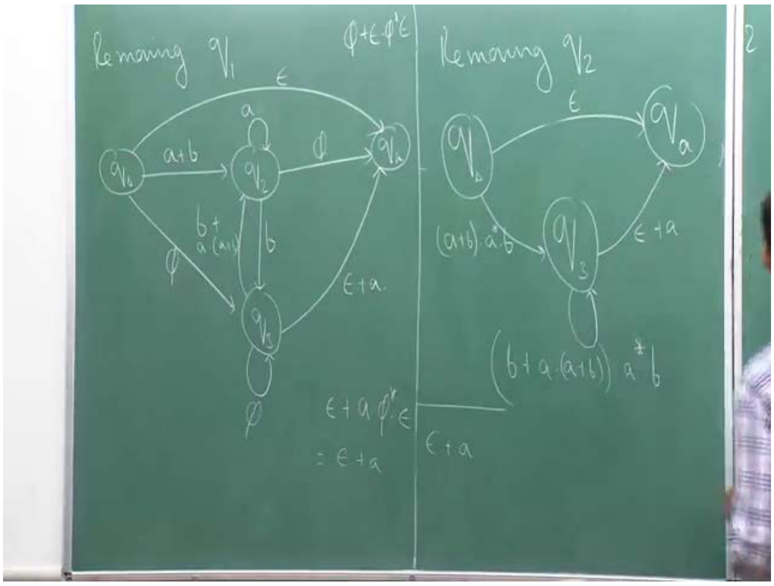
So, the old label from q 3 to q 2 is b. So, I have b now how can I go from q 3 to q 2 via q 1. So, here I have a, so b plus a dot there is nothing on q 1. So, it is phi star dot q 1 to q 2 I have a plus b right, yeah q 1 to q 2, I have yeah a plus b. So, this gives me b plus a dot a plus b. So, I have b plus a dot a plus b. So, this is q 3 to q 2.

(Refer Slide Time: 26:40)



Now, on q 3 itself I did not have any self-loop earlier, and if I try to work out what I will get here. So, earlier I had only phi. So, it will be phi plus q 3 to q 1 I had a, a dot q 1 there is no self loop. So, phi star and from q 1 to q 3 again there was nothing so phi. So, this is equal to phi. So, I will have just phi over here.

(Refer Slide Time: 27:16)

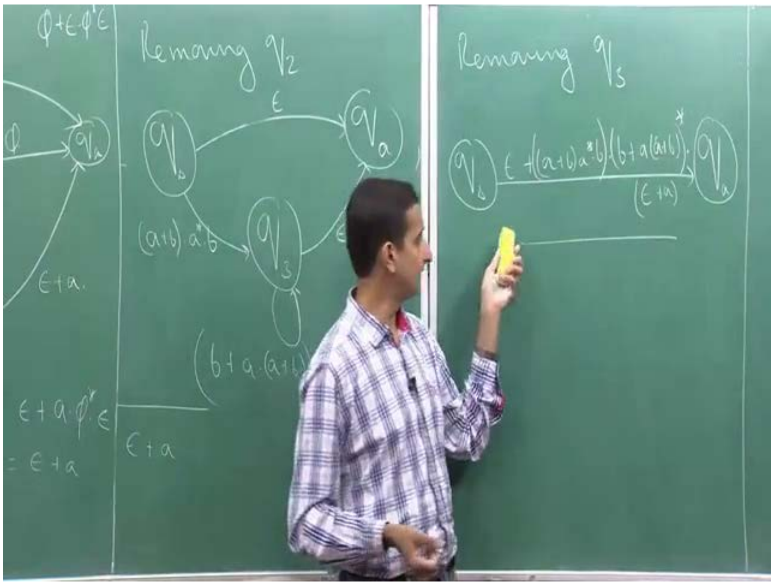


And finally, from q 3 to q a, we will have epsilon plus a dot phi star dot epsilon. So, earlier I had epsilon, now I will have epsilon plus a dot phi star dot epsilon, this is nothing but epsilon plus a, the new label will be epsilon plus a. So, this is a bit, it can get a bit complicated, but once we actually start shortening it, it gets slightly easier.

Now let us remove the next state, let us remove q 2, so removing the state q 2. So, if I remove q 2, what I will get is we have q s and then let say we have q 3, and we have q accept. So, from q s to q accept, I will get epsilon plus a plus b dot a star dot phi which is the same as saying it is epsilon plus phi, which is the same as saying just epsilon. From q s to q 3, I will have phi plus a plus b dot a star dot b, so q s to q 2 to q 3, so this is done.

Now from q 3 what would be the label on it is self-loop. So, it will be phi plus q 3 to q 2 is b plus a dot a plus b whole concatenated with a star concatenate with b. So, here I go from q 3 to q 2, which is b plus a, a plus b, then self-loop on q 2 is a star and then I come back from q 2 to q 3 which is just b. And from q 3 to q a, what we will get is epsilon plus a. If I just let me just work it out first. So, we will have epsilon plus a plus q 3 to q 2 is this - the self-loop on q 2 is a star dot phi which will just give me phi. So, it will just be epsilon plus a and that is it. So, this is the next automaton.

(Refer Slide Time: 31:07)



And finally, I have to remove the last state which is q 3. So, what we get is, so I have q s and I have q a. So, the expression will be epsilon plus this which is nothing but a plus b a star b concatenated with this whole star, so b plus a dot a plus b whole star concatenated with epsilon plus a, so concatenated with epsilon plus a. And if you want, you can put this also in a bracket just to emphasize that.

First we have this, then we have this star and then we have epsilon plus a, and this is basically the answer. So, this is the regular expression corresponding to our initial DFA, and this is the algorithm. So, it is iterative process; where in each step, you remove one state; and by removing the state, you construct a new GNFA which has new set of labels on its transition. Until and unless you get rid of all the internal states except for q s and q a, whatever label the edge from q s to q a, has is your final expression.

Thank you.