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

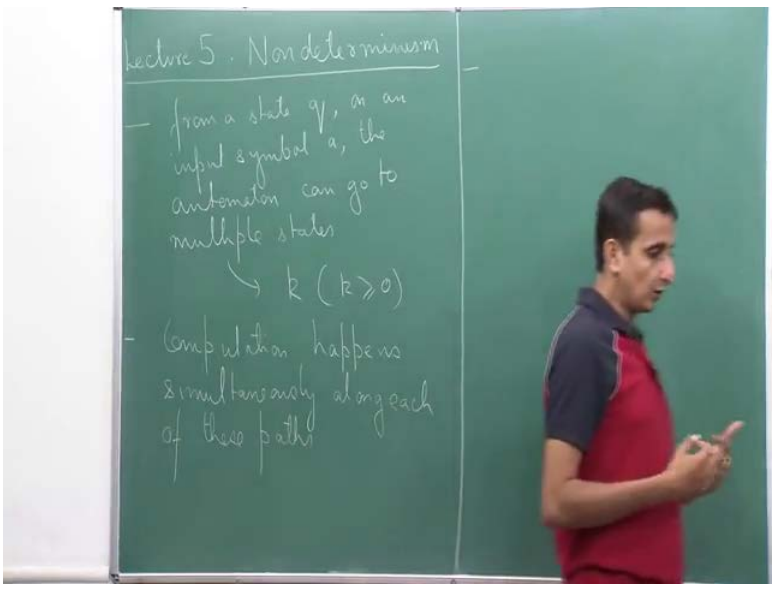
Lecture – 05
Introduction to Non-determinism

Welcome everybody. This is the 5th lecture. And today, we will talk about Non-determinism. So far we have talked about the computational model deterministic finite automata. In deterministic finite automata, from a given state on a given symbol there was a unique state in which the computation proceeded to. So, from a state q 1, a symbol a computation always went to a unique state, what if we try to generalise this theme.

Today we will talk about the concept of non determinism. What happens in non-determinism is that from a state on a input symbol computation can proceed along multiple different paths. So, it can go to multiple states, and then on each of those states computation can proceed as before.

Let us try to understand what we mean by non-determinism and how is a non-deterministic finite automaton different from that of a deterministic finite automata. So, today we will not give you the formal definition of what a non-deterministic finite automata is will do that in the next lecture, but our goal today is to understand non-determinism.

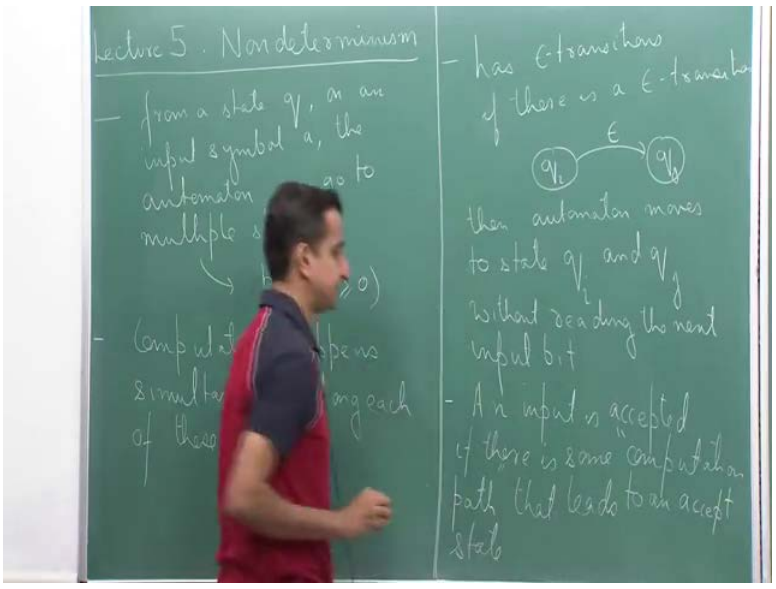
(Refer Slide Time: 01:59)



Basically, the key points in non-determinism are as follows. So, in non-determinism from a state q on an input symbol ‘a’ the automaton can go to multiple states. And what we mean by the terms multiple here is it can go to k different states, where the number k is greater than or equal to 0. In other words, the automaton can go to 0 states in which basically means that it does not go anywhere on seeing a particular symbol or it can go to one state, or two or three or any number that is more than that. So, computation happens simultaneously along each of these paths. Now let us try to understand what we mean here. As I said that let say the computation is at a particular state q, and on seeing a symbol ‘a’ it move to several different states, let say it move to three different states.

Now, what happens from each of these three states computation will precede at the same time to other different states. So, it is not that one computation halts, and whereas the other one proceeds a first and so on; the way they should be pictured that is along all these paths from all the states that that the computation went to when going from q an symbol a, it will move forward in a simultaneous manner. It will move forward at the same time. Again, when we will very soon look at an example and this point will be clearer at that point.

(Refer Slide Time: 05:43)

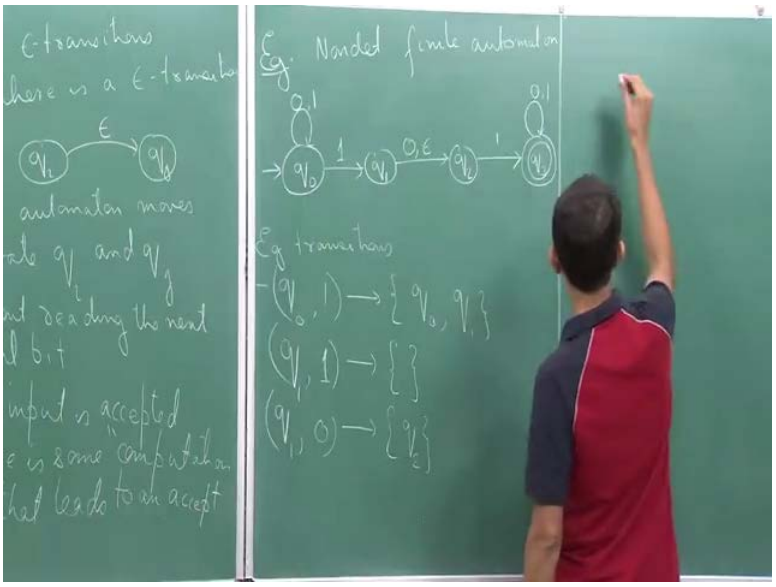


The third difference the third feature of non-deterministic computation is that of epsilon transitions. So, so far when we talked about transitions, when we went from one state to another it was always on a symbol of the alphabet, but now in an non-deterministic

model, we can also have transitions which are labelled by epsilon. So, has epsilon transitions which means that from let say that if there is a epsilon transition of the form let say from some q i to some q j, there is a transition like this, then the automaton moves to state q i and q j without reading the next input bit. If there is a transition of this form, then the automaton would move to states q i and q j, moving to state q i essentially means that it is stays at state q i. So, one computation stays at q i, the other moves to q j, and it does this without reading the next input bit, even without reading a bit it will do this. And from there on, the computation will proceed as earlier.

And the last point is that of acceptance and input is accepted, if there is some computation path that leads to an accept state. So, what is a computation path? The computation path is just one of these paths that I am considering. So whenever the computation branches off, it gives rise to more than one computation path. If any 1 of those computation path leads to an accept state then we accept input. If all the computation paths end up at a rejects state, then we do not accept the input, so that is the idea.

(Refer Slide Time: 09:13)

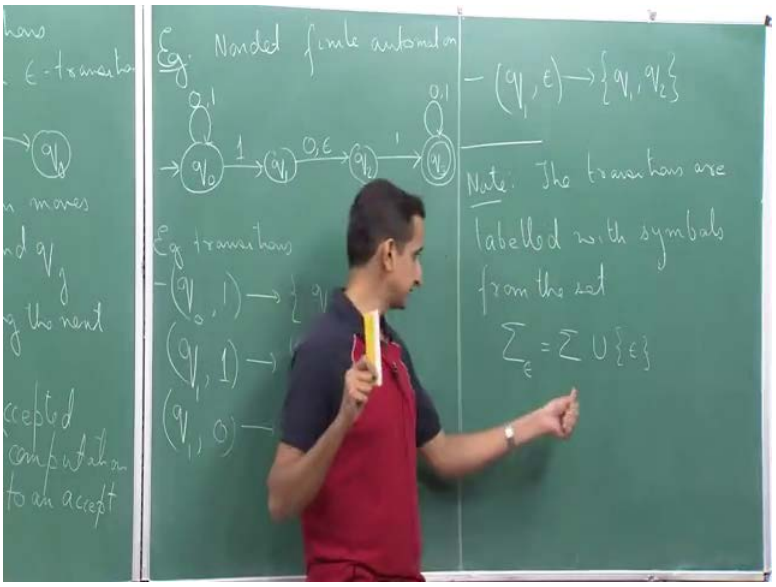


Let us look at an example. Here, is an example of a NFA. We have state q 0, which is the start state. So, from q 0 on a 0, it stays at q 0; and from q 0 on a 1, it stays at q 0 as well as it go to the state q 1. Here you can see that here is an example, where from q 0 on seeing the input bit 1, the machine goes to two states that is q 0 itself and q 1. Now come

from q 1, it goes to q 2 on the symbol 0 as well as on epsilon. So, there is an epsilon transition from q 1 to q 2 as well as there is a transition labelled by 0. From q 2, if we go to q 3 on a 1; and q 3 is our accept state and let say that at q 3, I stay at q 3, if I see a 0 or 1. So, this is an example of NFA. Let us look at some of the transition.

As I said that from the state q 0 on a 1, we go to both these states q 0 and q 1. Similarly, if you look at another example from the state q 1, if we see a 1 where do we go to? So, look at the state q 1, there is no transition coming out of q 1 on the symbol 1. Therefore, it is just the empty set. As I said earlier at the beginning that we can have we can go to multiple states from a given state q on input symbol which does include the number 0 as well. So, it this is an example where we go to no state, hence the empty set. Another example, let us take may be q 1 on 0, we go to a single state that is q 2. These are some example transitions.

(Refer Slide Time: 13:16)



Now let us look at an example of the epsilon transition also lets try to see what happens. We have an epsilon transition from q 1 to q 2. So, from q 1 on this epsilon transition, where do we go to? Again as defined earlier, let us look at the definition of this epsilon transition. When we have an epsilon transition, the automaton simultaneously moves to the state to which there is an epsilon transition as well as it stays at the same state without reading an input bit. So, that is what happens here. So, from q 1, we have an epsilon transition to q 2. So, the set of states that the automaton goes to is q 1 comma q 2.

Here, these are some of the example transitions. So, one point that I should make at this juncture is that the transitions are labelled with symbols from the set. This is usually denoted as sigma epsilon, which is basically sigma union with epsilon. Here, we have so if we look at the labels that are there on these states that are there on these transitions they can either be symbols from sigma or they can be the label epsilon and this set is called usually referred to as sigma with subscript epsilon. So, we will be using this notation frequently.

(Refer Slide Time: 15:55)

Now let us look at an example of an input that is accepted by this automaton. So, consider an input let say w equal to 0 1 0 1 1 0. So, how does this automaton. So, we have our automaton over here. So, how does this automaton proceed with its computation on this input? Let us try to look at this. So, we will view the computation by drawing a table; first, we will have, on the left hand side, we will have the set of states the automaton is in; and on the right hand side, we will have symbol read. Initially at the beginning, even before to begin our computation by definition the automaton is in the initial state or the start state. We will have q 0 here, and this is the initial state that we have.

Now what is the first bit that is being read, so the first bit that is being read is 0 or the first symbol that is being read is 0, so we read 0. Now what happens to the automata. So, on reading 0, the automaton from q 0 it states at q 0. So, there is only one transition on 0,

and that is to q 0. On reading zero the automaton goes to the set q 0 itself. The second symbol is 1. What happens when it reads a 1. When the automaton reads a 1 from q 0, note that it stays at q 0, because there is a 1 over here, it goes to q 1 also because there is a 1 over here. So, it goes to the states q 0 comma q 1, they are still not done. Because look at the state q 1, so from the state q 1, there is an epsilon transition, and by the definition of epsilon transition the automaton takes an epsilon transition without reading the next input bit.

So, from the state q 1, it sees an epsilon transition the definition of epsilon transition is that if there is an epsilon transition the automaton will go to q 1 as well as q 2. So, there is already q 1 present here, so we do not need to write it again. So, the automaton will go to the state q 2 as well. So, basically from q 0 on seeing the input bit 1, it goes to state q 0, q 1 and to q 2.

The next bit is zero. So, we will have to analyse. Now observe that, there are three different computations that are taking place; so there are three things that happen in simultaneously. We have one computation at q 0, one at q 1, and one at q 2. So, what happens to each of this computation? So, we will look at them one by one. So, from q 0 on a 0, so here we have q 0. Again on a 0, we will stay at q 0. What happens from q 1, from q 1 on a 0, we go to q 2, so we go to q 2. And what about q 2, what happens at q 2. From q 2 on a 0, there are no transitions, so we go to the empty set, which means that I do not write anything more here. We just have q 0 comma q 2. So, essentially what this means is that from the set of states q 0, q 1 and q 2 in the next step the automaton goes to q 0, q 2.

The next symbol the input bit to be read is 1. And I would not go into the details of the analyses, you can figure it out. So, from q 0 and q 2 on a 1, so we know that from q 0 on a 1, we go to q 0, q 1 and q 2. And from q 2 over here on a 1, if you look at the automaton we go to the state q 3, so we go to q 3. The next input bit is again a 1. So, again from q 0, q 1, q 2, q 3 on a 1, we go to q 0, q 1, q 2 and q 3. So, only one more bit remains - the last bit that is a 0. So, on reading a 0, let see what happens. So, from q 0 on a 0, we stay at q 0; from q 1 on a 0, we go to q 2; from q 2 on a 0, we do not go anywhere because that is the empty set; and from q 3 on a 0, we stay at q 3. So, we have q 0, q 2, and q 3. So, these are the final set of states that the automaton is in.

And, now does this input get accepted or not. The condition for acceptance was that if some computation path if there is one computation path that leads to an accept state then we accept. Another we way of stating is that in this last set of states that I have, is there an accept state that is present here. The answer is yes, because the only accept state in this NFA is q 3, and we have q 3 over here, which means that the input is accepted. So, before I proceed, I mean this concept of non-determinism is a very important concept; and later on when we will see more powerful computational models, again we would be studying at the non-deterministic versions of those computational models. It is very important we spend some time and we understand the idea of non-determinism, because it is basically the same for those other models as well.

So, one very useful way of understanding or getting the intuition of what non-determinism is that, so think of a non-deterministic automaton as an automaton that is placed on the ground, and there is an given an input, imagine that there is some extra terrestrial being some very powerful being who is able to create clones of himself or herself. This person is able to clone him or herself into as many clones as needed. So, whenever so this person on given an input, starts at the starts state; and whenever it sees an input bit it does the following. If on that input bit, there are came many transition; this person creates k copies of itself; and in the next step each of those copies will go to these k different states.

It has these k different copies and each of those copies goes to the k different states, and it keeps on proceeding in this manner. Suppose at some point, it reaches a state; and from that state on a particular symbol, there are no outgoing transitions, for example, in this case, we had that from q 2 on a 0 they are no outgoing transition I mean such a thing can happen. In which case, that particular clone just dies out, so that particular clone does not survive. Well there are the other clones, which are of course, present there. So, this is what happens when we have a regular transition.

In the case of an epsilon transition, something slightly different happens. So, if there is a from a given state, if there is an epsilon transition that let say that from the state q 1 we have an epsilon transition to q 2. What the this person will do is that particular clone will do is that so that particular clone if there are k epsilon transition is going out it will create k plus 1 copies of itself, or it will create k plus 1 many clones. One of these copies stays

at the current state, whereas the other k copies will go to the k states to which we have an epsilon transition. It is basically the same as.

Here we have an epsilon transition to q 2. So, it creates two copies one stays at q 1, whereas the other goes to q 2. So, each of this clones basically work independently and they proceed in the same manner. So, one clone does not know what is happening with the other clone, whenever it is at a state and it sees a symbol it just works in this manner.

Finally, if some clone ends up at an accept state at the end of reading entire input if there is some clone as ended at an accept state then that input is accepted; otherwise it is not. This is basically a very intuitive way of understanding how non-deterministic computation happens. So, non-deterministic computation is not a natural computing phenomenon in the sense that we cannot implement it using a standard computer because in a standard computer we cannot create different computations at the same time. So, we have a processor, a processor at any given point of time can process only one thing; it cannot process several things, so that is why it is not a natural.

But then it is very that the reason why we studied because it gives there are several problems for which we get an efficient solution if we use non determinism. So, we can solve certain problems in a better manner using non determinism. And also from a theoretical point of view it is interesting because it allows us to compare with determinism.

Basically it gives rise to the question that how powerful is non-deterministic computation, is it more powerful then determinism, is it the same is it does it have the same power as determinism. Maybe for I mean what happens for different models for finite automata for maybe models, which have more resources; it allows us to compare how determinism and non-determinism behave. So, whether one is more powerful than the other or not. So, I will stop here.

Thank you.