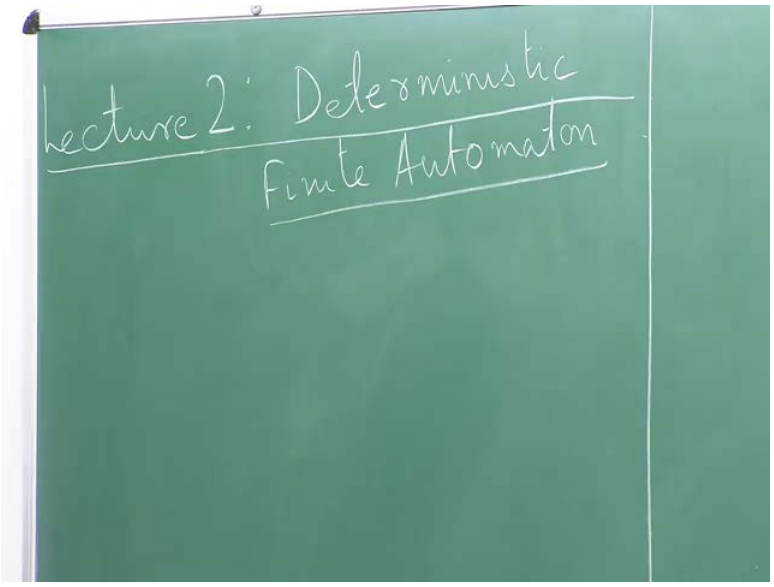
Theory of Computation
Prof. Raghunath Tewari
Department of Computer Science and Engineering
Indian Institute of Technology, Kanpur

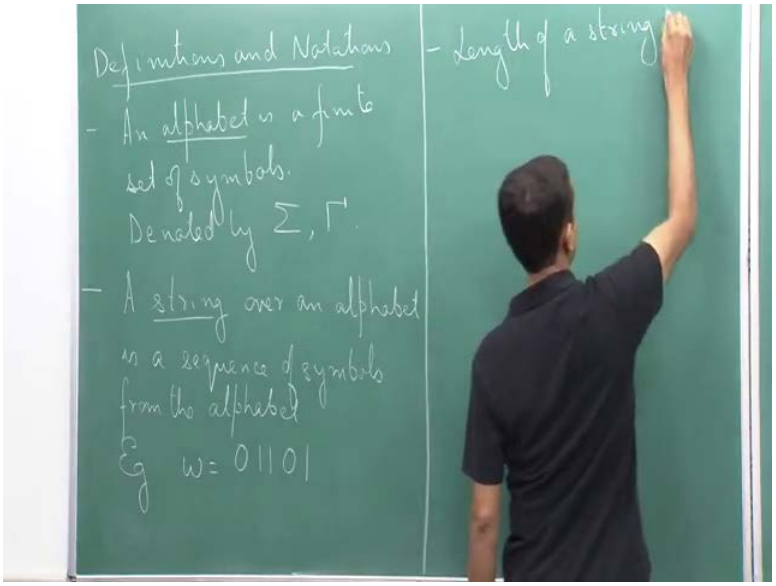
Lecture – 02
Basic Notation and Convention, DFA Edit Lesson

(Refer Slide Time: 00:14)



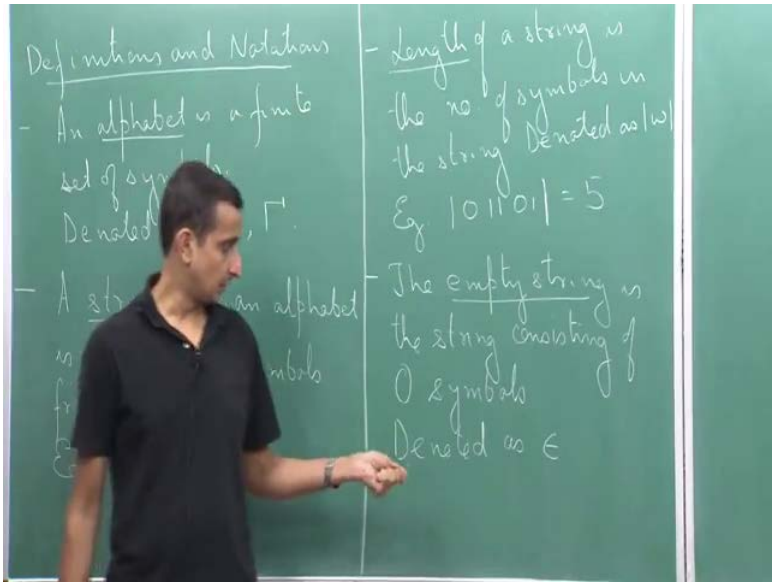
Welcome everybody. This is the second lecture. In this lecture, we will talk about Deterministic Finite Automaton. We will give the formal definition and will talk more about its properties.

(Refer Slide Time: 00:27)



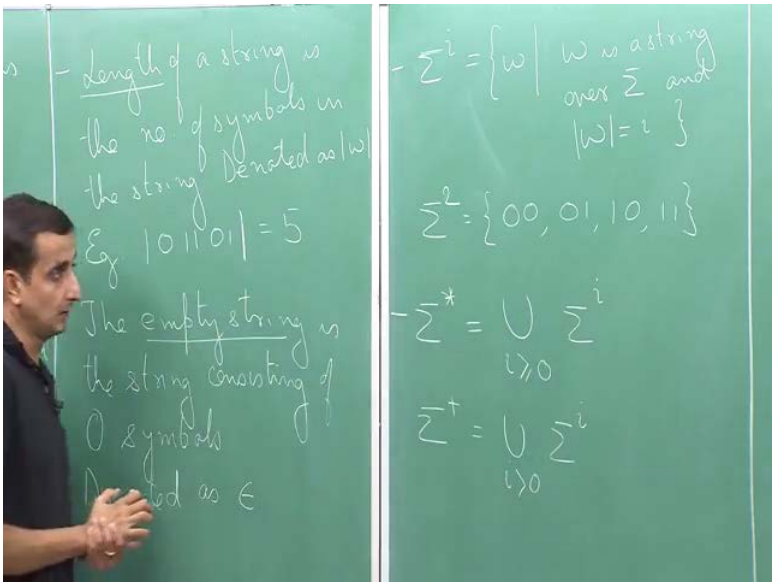
So, First thing that we define is alphabet. An alphabet is a finite set of symbols. So, this is usually denoted by the symbols capital sigma or capital gamma. So, it is a finite set containing symbols. A very common example is the binary alphabet consisting of two symbols let say 0 and 1 or you can have more than two symbols and so on. A string over an alphabet basically sequence of symbols from the alphabet. For example, if you look at the primary alphabet consisting of the symbol 0 and 1, we have a string let say w, which is equal to 01101. This is an example.

(Refer Slide Time: 02:49)



Length of a string is the number of symbols in the string. So, this is for a string, let say w, we denote its length as w with a modular sign around it. For example, the length of the string 0 1 1 0 1 is 5, it has five symbols in it. We also have a special type of string call the empty string or which is denoted as epsilon. The empty string is the string consisting of 0 symbols. So, this is denoted as a small epsilon. The point to be noted here is that this string epsilon does not have any symbol. This is not the same as the symbol 0. Here, this is a string which has zero symbols in it, and it is denoted using this character epsilon.

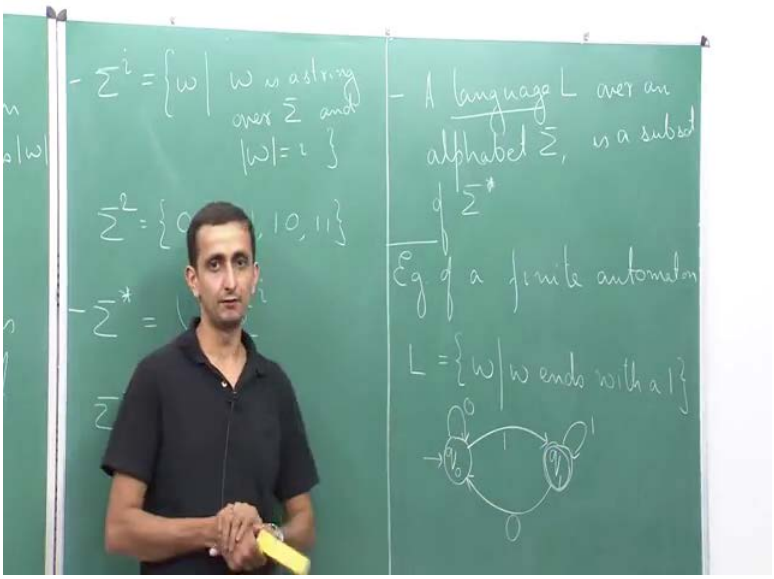
(Refer Slide Time: 05:19)



Now given in alphabet sigma, we defined the set sigma to the power i as the set of all strings w such that so w is a string over sigma; and the length of w is equal to i. So, for example, if we take sigma as the alphabet consisting of symbols 0 and 1, then sigma square will be set of strings 00, 01, 10 and 11, so these are the only four strings which have length equal to 2. Now sigma star is the set of all possible strings that can be constructed over the alphabet sigma. This is written as union over all i greater than or equal to 0 of sigma raise to the power i. It contains sigma 0, sigma 1, sigma 2 and so on.

An important point that must be noted here is that, what is sigma 0. What are the strings that have length 0? So, sigma 0 is the set consisting of only the string epsilon, because as we defined epsilon is the string the only which has length 0. And all though the following notation is usually defined, but it is very sparsely used, so sigma plus is the set of all strings that have length strictly greater than 0.

(Refer Slide Time: 07:43)



Now we can define what the language is. A language L over an alphabet sigma is a subset of sigma star. So, language is basically any collection of strings from sigma star. In particular note that it can be the entire set sigma star as well or it can be the empty set or it can be any other set in between the empty set and the complete set sigma star. These are some of the basic notations that we will be constantly using, and we will be referring to for the rest of this course.

So, now that we have this frame work let us get back to what we mean by deterministic finite automaton. So, last time, we gave an informal description of what deterministic finite automata is, what are finite automaton and then what kind of sets or what kind of languages such computational models are able to accept. Today, we will give a formal definition of the following.

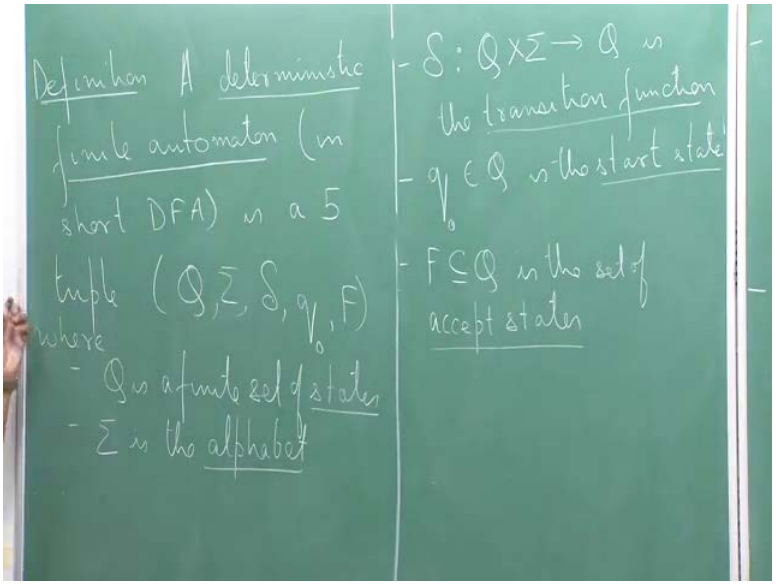
Once again, before we give a formal definition let us look at an example. Consider the language L, which consists of all strings of the form w such that w has w ends with a 1. All strings that end with the 1. So, how can we give a finite automaton that accepts all strings that are in this language? We will construct automata that will have two states, so the two states will intuitively contain the following information. So, one state will

contain the information of all those string that end with 0 and the other state will correspond to all those stay or all those strings that end with a 1.

So here, we have a start state, so we refer to the start state with an arrow pointing in to it. This is the state that will correspond to all strings that end with a 0. Whenever we see as, if you are sitting at this state, and we see as 0, we remain at this state; and we will have another state that corresponds to all strings that end with a 1. So let us give them some names as well; so will call this state as q 0, and will call this state as q 1.

So, from q 0, if we see a 1, we go to q 1. Now, when we are at q 1 and if we again see a 1, it means that the string is has a one at the end, which means that we will stay at the state q 1 itself. On the other hand, from q 1, if we see a 0, we will come back to q 0, because that means, that now the last symbol of the string that we have seen is a 0, hence we should be at q 0. And now what is the string that needs to be accepted, so according to the language l we should accept all strings that end with a 1. So, our accept state would be the state q 1, so we denote a accept state with a double circle, so that is the notion that we will follow. Now that we have this example, we will formally define what we mean by a deterministic finite automaton.

(Refer Slide Time: 12:37)



A deterministic finite automaton or which we will in short refer to as a DFA, in short DFA is a 5 tuple Q, sigma, delta, q naught comma F. Before I proceed, before I proceed and I explain what each of these symbols mean, let us intuitively try to understand what exactly we are trying to quantify here. So, if I get back to this example as so coming back to this example that I give. A deterministic finite automaton is basically a computational model that needs to capture whatever is happening here. So, it needs to capture the structure of this object, this mathematical object. And if you look at this mathematical object, this is in some sense it looks like a directed graph.

So, you can imagine this state as vertices and you can imagine these transitions that are happening as directed edges. It is a directed graph which in addition has labels on the edges. So each edge has a fix label. And in addition, here we have specific state which is the starts state; and we have some states which are our accept states. This is what we need to quantify in our definition of a DFA so that is what we are going to do here.

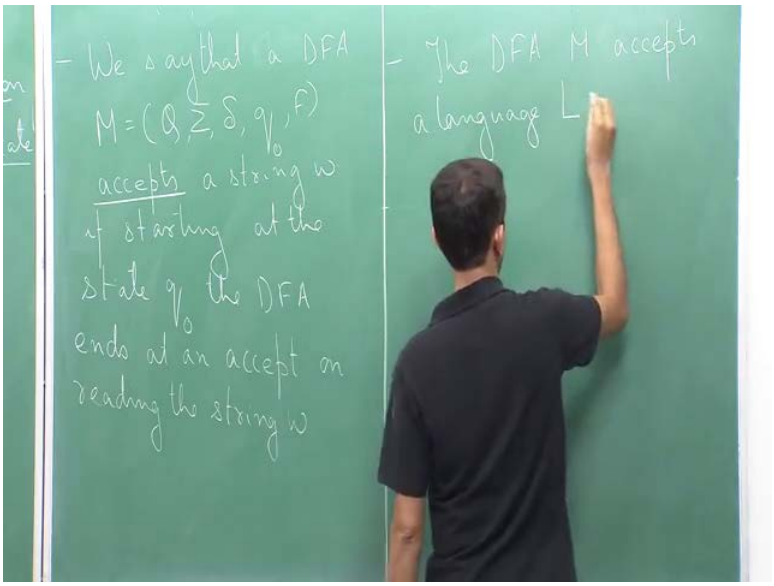
Where, capital Q is a finite set of states sigma is the alphabet. This small delta is what is known as the transition function, so delta is the let me just take it over here. So delta which is a map from Q cross sigma to Q is the transition function; q naught which is an element of q, so it is a special state, it is the start state. And F is a subset of q is the set of accept states. So, now observe that once we have such a description, it exactly quantifies what the example that we gave earlier that of a deterministic finite automaton. So, again let us go over it, so Q represents the states that we have. So, in this directed graph it is the set of vertices that are there. Sigma is the alphabet So these are the labels that we have on the edges of our DFA.

In this case, the example that we had earlier is 0 and 1. Delta is the transition functions, so this kind of describes so this describes what the edges out going to be and what are the labels on these on each of these edges. So, what delta takes as input is a pair is a tuple consisting of a state and an input symbol and it goes to another state.

Let us look at this example. Suppose, if I give the state q 0 and I give the symbol 1, then it maps q 0 comma 1 to the state q 1. If I give the state q 0 and the symbol 0, then it maps q 0 comma 0 to the state q 0. Similarly, if I give q 1comma 0, it maps q 1 comma 0 to the

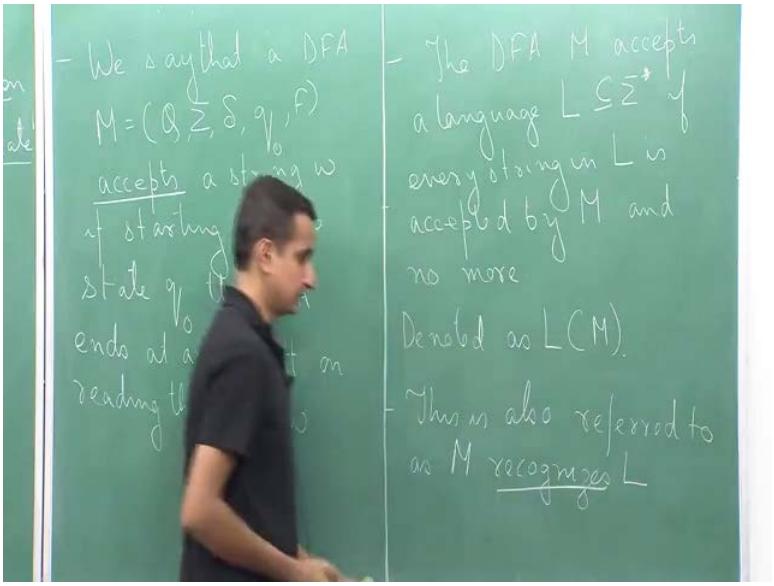
state q 0 and so on, so that is our transition, so this is the transition function. So, for every pair of consisting of a state and I input symbol it produces state that the automaton goes to. Then we have a unique state q 0 which is our start state and we have a sub set of states which are the accept state. At the end of reading the entire input, if the automaton is in one of the accept state, we accept that input; and if at the end of reading the input the automaton is in a state which is not an accept state, then we do not accept that input.

(Refer Slide Time: 20:00)



Now, let us introduce some notation related to DFA. So, we say that a DFA M accepts a string w if, so let us given informal description for the time being, so if starting at the state q 0 the DFA ends at an accept state on reading the string w. If starting from the state q 0, so we say that this DFA accepts a string w. If starting at q 0 on reading the string w the DFA ends at an accept state, and if it does not end at an accept state then we say that it does not accept the string w.

(Refer Slide Time: 22:28)



Now we can define language. So, the DFA M accepts a language l subset of sigma star, if every string in L is accepted by M and no more; so this important point. So, we see that m automaton a finite automaton M accepts a certain language L; if every spring that is present in the language is accepted by M, and every string that is not present in the language is not accepted by M. So, every string that is presented is accepted by M, and no more; so no other strings. It accepts exactly those strings that are in the language L, so this is denoted as L of M.

So, when we say l of m we mean the set of all strings that are accepted by a DFA M. Again as I said that this is a common mistake that people tend to make, when they say that L is accepted by M is the construct an automata which possibly accepts every string that is in L and also some more strings, but that should not be the case. So, when we say that an automata accepts us to a language L then it should accept exactly those strings that are in the language L and no more. So, some text also use the term recognizes in place of accepts. Let me just mention that. This is also referred to as M recognizes the language L. As far as theory of computation is concern, we use both these terms accepts and recognizes synonymously.

Thank you.