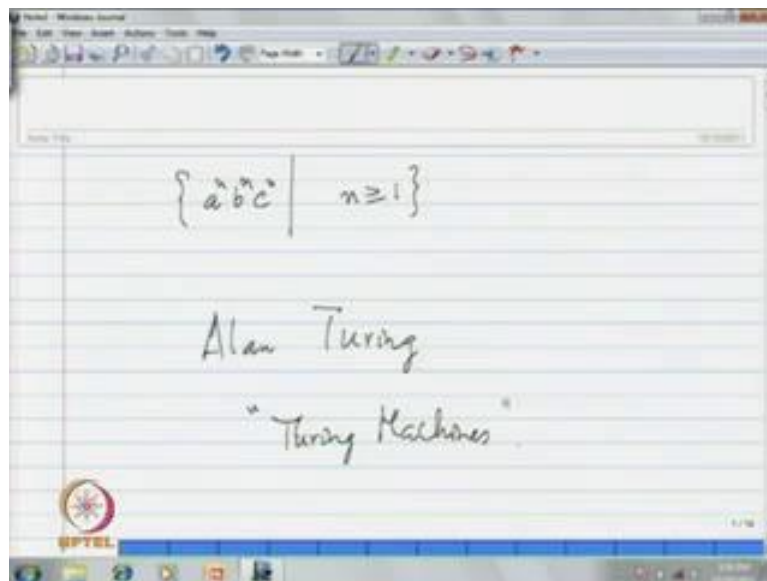


**Formal Languages and Automata Theory**  
**Prof. Dr. K. V. Krishna**  
**Department of Mathematics**  
**Indian Institute of Technology, Guwahati**

**Module - 11**  
**Turing Machines**  
**Lecture - 1**  
**Definition and Examples**

We have discussed about certain types of automata, particularly finite automata and its variants deterministic finite automata, non deterministic finite automata, etcetera, pushdown automata. We have observed that these automata have certain limitations and we have identified there are certain languages, which are not accepted by these automata.

(Refer Slide Time: 00:50)



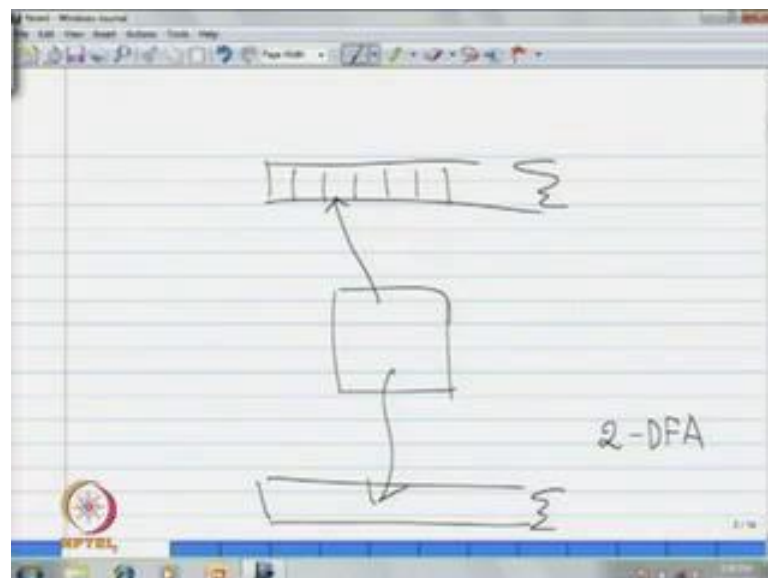
Particular if I as an example you consider this  $a^n b^n c^n$  such that  $n$  greater than or equal to 1. So, this language we understood that it is not context, there is no pushdown automata accepting that. And in the usual a competition using your routine computer. It is very clear that there are much more complicated languages than this particular example, you can compute it and see that the computer is handling such languages all that.

Now, when one thinks for modeling this computation or the notion of algorithm, essentially now these types of automata, so far we have introduced or I have or many

restricted and one looks forward for a better model, which can accept much more complicated languages than whatever, so far we have seen through this push down automata or finite automata. So, in that direction in this lecture, we discuss about an automata, so called Turing machine, this is named after Alan Turing mathematician is called Turing machines in this lecture.

And, we understand that these are having enough computational facilities and indeed that this will work for modeling the computation. And this Turing machine has having the features of an automata, there will be a tape, finite controller and all those things, here is very important point is that in case of finite automaton we have a left just finite right side infinite tape and a finite control.

(Refer Slide Time: 03:08)



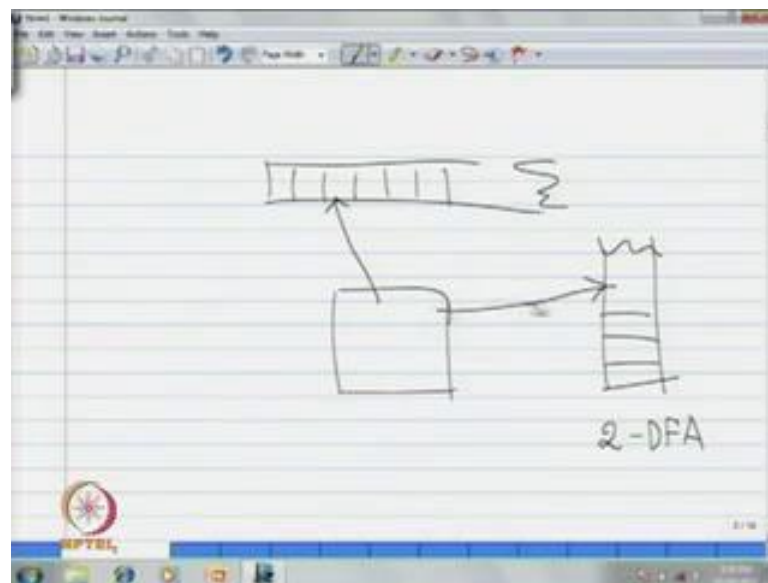
So, we are taking the symbols the input from the tape and with respect to the internal state, what is the transition is given. Accordingly it will take the transitions and finally, if it is reaching to a final state, we are saying it is a input is accepted, otherwise rejected. That is how in finite automaton we have followed and we have certain variants for the class of languages accepted deterministic finite automata, finite automaton this a regular languages.

That is you know we have discussed about 2 DFA, that is you are allowed to move this reading head not only to right may be left. That way we have discussed 2 DFA and it is not having much computational power than a typical deterministic finite automaton

DFA. So, essentially the class of language actually 2 DFA is same as regular languages, although you allowed to move it and we have discussed machines, which give output there you have an output tape and you are allowed to print some symbols on this corresponding to each symbol there more mille sort of machines we have discussed.

And we see that this type of machines are also very restrictive to handle some complicated languages or to give certain outputs as for the requirement, now in case of pushdown automata we have a stack.

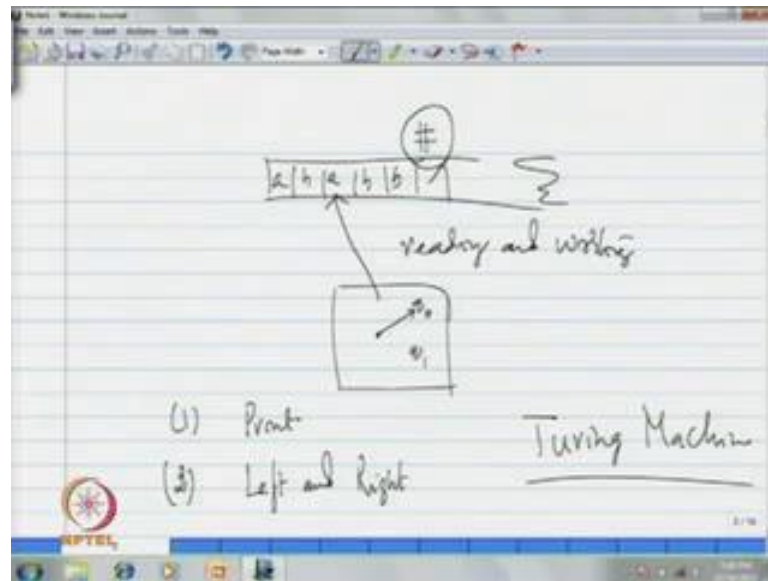
(Refer Slide Time: 04:55)



Let me call for example, a pushdown stack sort of we have discussed in case of pushdown automata, there is a stack. And in which case we have seems like this and now what are the symbol that we are writing corresponding to that you may push something some information to the stack and read using stack principle on this and, so on. Now, in case of Turing machine, the situation is if you are allowed to right I mean printing the symbol and allowed to move both left and right.

In which case we actually introduce the concept with that we introduced the concept called Turing machine, we introduced this concept called Turing machine and a Turing machine is the at most powerful device for the computation. In fact, whatever that you can do using routine computer that you can do using this Turing machine such a powerful this thing.

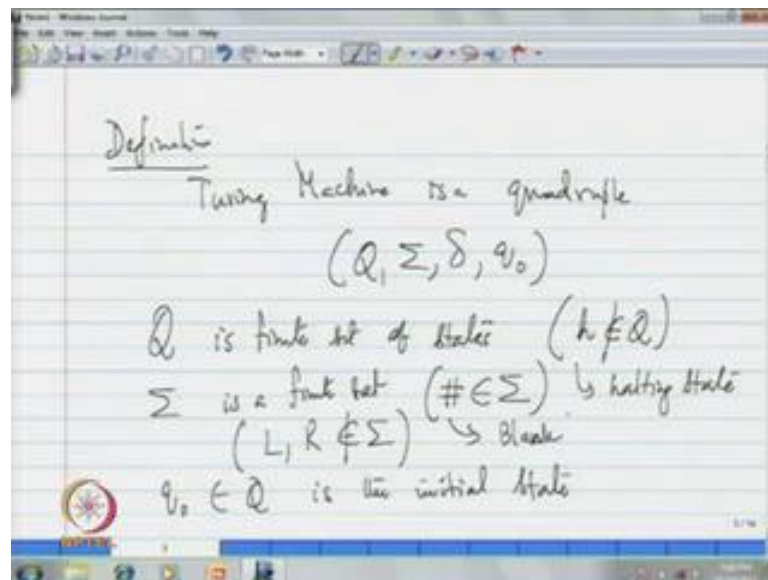
(Refer Slide Time: 06:04)



Only two things extensions for a DFA, that we can clearly see that you are allowed to print on the tape. And number 2 we are allowed to move left and right, so with this two we will gain the ultimate power to represent computation; that means, whatever we can do with the routine computation in computer that you can do it using Turing machine. Now, informally the things is like this a left just fate right side infinite tape and a tape is divided into cells.

Now, here this is not only reading head in like in case of DFA, this is reading and writing head and you have finite control certain states say  $q_{\text{naught}}$ ,  $q_1$  and so on, in the pointer pointing to this. Now, the transitions are something like this given a state and a symbol you are allowed to you know print also or you are allowed to move not only to right you can move one cell to left also. So, this kind of thing is allowed in case of this Turing machine, so with this we will get the ultimate power as I had mentioned.

(Refer Slide Time: 07:52)



Now, let me give you a formal definition for this a definition of a Turing machine ((Refer Time: 07:59)) as earlier you require a set of states and certain symbols that you will put it on the tape. And here in contrast to the DFA, here we designate one particular special symbol that is like final state we call it as halting state, that once you reached to that particular halting state you will not have any further computation; that means, there no transition is defined.

So, you will reach there and you will stop the computation there, that is how one special state is introduced. And once you give the input on in case of DFA on the tape and after that, you see this is the right just left justified right side infinite tape there are several cells. For example, if I give the input a b a b b now these are all the cells, which are available and in case of DFA we simply go till the end of the input and we stop it there, the computation this is what automatically stops over there.

Since, you are allowed to go left and right and you have that facility, now once you reach to this, if you want to go further and if you have this situation to come back and read the input and what are the changes that you do in the input. So, based on that this input for example, here there are 5 symbols, since you are allowed to right you may go and at the end you may print something and you may go further and print something. So, that you can always do such modifications and therefore, you are visiting this cells, which are blank where you are not giving the input.

So, this blank cell we may represent with the symbol this a hash, so we include a special symbol, you know when we are talking about input this special symbol will also be considered. So, now there are two things, one for the final state are I call it as halting state, the computation when the machine halts the such a state is called halting state, the machine comes to that particular state and after that there we do not define any transitions.

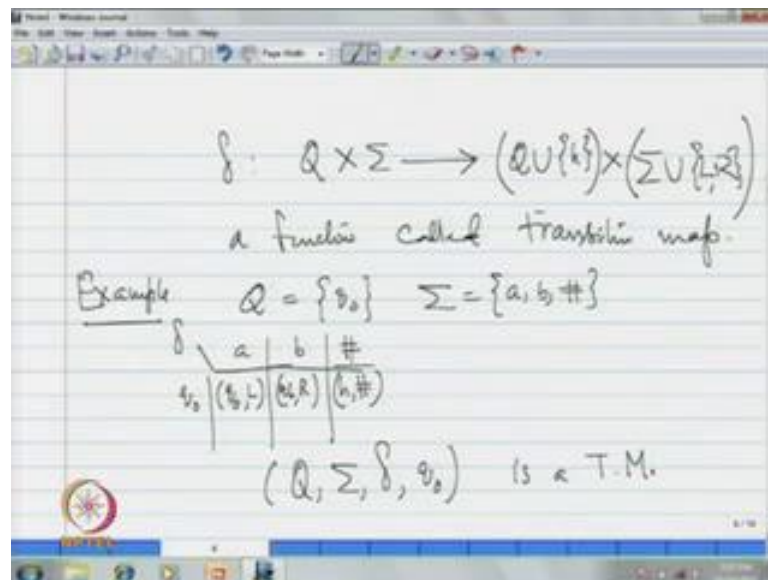
And this new special symbol for the in case of input that we are considering is blank to represent the blank cells on the tape. So, these are the two important new things that we have not, so far discussed, so this Turing machine we introduce it as a quadruple. Therefore, components, let me write  $Q$ ,  $\Sigma$ ,  $\delta$ ,  $q_0$  as you know  $Q$  be normal state for the states,  $Q$  is a finite set of states and here we assume that this symbol  $h$  this is writing for halting state, this is not in  $Q$  this is we write for a halting state.

So, this is one and  $\Sigma$  is again a finite set there is a alphabet, from which you give some input and what are the symbols that you want to type. So, all these symbols should be available here, this is a finite set this include this particular symbol hash, so including this symbol. But, of course, here now moving left and right because, in case of DFA or finite automaton what is the situation there, automatically the reading and writing head in a particular transition takes a right move.

And in case of 2 DFA if you remember we have defined the transition we are defining it, whether it has to move to left or it has to move to right we are defining it. So, here also we have to define, so those symbols we assume that they are not part of this  $\Sigma$ , so further you assume that this symbols  $L$  and  $R$  which are indicate which we are using for the reading head to move to right or left, they are not elements of  $\Sigma$ . So, this care you take, so blank symbol has to be there in  $\Sigma$ , this is representing blank cell on the tape.

So, this will be there and now as usual  $q_0$ , this is a initial state ((Refer Time: 13:02)) now let us look a transition. As we have discussed given is symbol and a state, you may be allowed to print in a particular cell or you may be allowed to now move left right, so these are the possibilities that we have discussed.

(Refer Slide Time: 13:18)



So, I give you delta the transition function given a state and a symbol of sigma of course, you change this state, it can be halting state you may finally, halt after reaching to this state. So, this h is not part of q, so there are no further transitions defined for this because, delta we are defining from q cross sigma h is not in Q once you reached to the halting state, then we do not have further definition. So, that is how we taken care here, this is a function from Q cross sigma, but you may have you know in a particular transition you may reached to this states.

So, a state component is this and you may print in which case that is in element of sigma or you may move to left, so that is L or you may move to right. So, this is the transition function of the Turing machine, so this is the quadruple, so you take a finite set of states, you take some alphabet from which you may give the input and the symbols from which you may print on the tape and, so on. And now blank symbol has to be there in that and a halting state is not part of the state set Q naught is the designated initial state.

And the transition function say from this set Q cross sigma to the set Q union singleton h cross sigma union L R the meaning of this I have mentioned that is, if you take the element of sigma in which case in that particular transition you have to print some symbol on the tape. And if it is L if you are assigning L; that means, you are asked to move the reading and writing head one cell to it left, if it is r it is asked to go to right one cell, this is the transition map.

Now, let us look at some example just for the sake of simplicity, let me take it this set  $Q$  just one state. Because, I want initial state and  $\Sigma$  say a b a blank symbol has to be there, and I designate  $q_0$  as initial state because, there is only one state here and now I define it like this. So, for this symbol a, symbol b, blank in initial state if am reading I will change this state to  $q_0$  we continue in the same state and go to left if I encounter this b is only one state I encounter this and when I am reading this, I will halt and say take a right move R let me give it this way print blank.

So, suppose if I define transition map like this because, this is a state  $q_0$  a b blank for these symbols I have defined. Now, by definition  $Q, \Sigma, \delta$  and the initial  $q_0$  is a Turing machine, likewise you know you take a finite set of states and some alphabet and for every state and a symbol you give some transition here. That means, here state, either you asked to move to left or may be move to right, let me choose for example, may be here right, so that you have all the things represented.

So, move to right or you print here am printing blank, so likewise you define and you see by definition this quadruple is a Turing machine. Now, what is the behavior of this Turing machine, what type of jobs like that you can pursue and all those things that first you have talk about the language acceptance and all those things. Now, in case of DFA or pushdown automaton you have understood that is defined as a language acceptor, as I had mentioned some time back that this Turing machines will be shown as ultimate computational device.

That means, the at most powerful device for computation because, in case of computation it is not only thus language acceptance. It is not only language pursue it is that you may be given some input and you may expect some output in your program in your algorithm. So, that kind of computation we do, so here we have the facility of writing in the tape, so this Turing machine will be introduced is not only as a language acceptor. But, also to see like Moore and miller machines like Moore and miller machines are to give some output give some input and expect some output on the tape.

So, this kind of computation that we will at pursue with this Turing machine, any way since we are already familiar with the language acceptance part. That means, given a string on the tape with the initial state, this is an automaton it takes the transitions automatically and it will coming to the final state by the time of you know this



transitions after completing that input, but here the situation is you are allowed to move to left right all the directions.

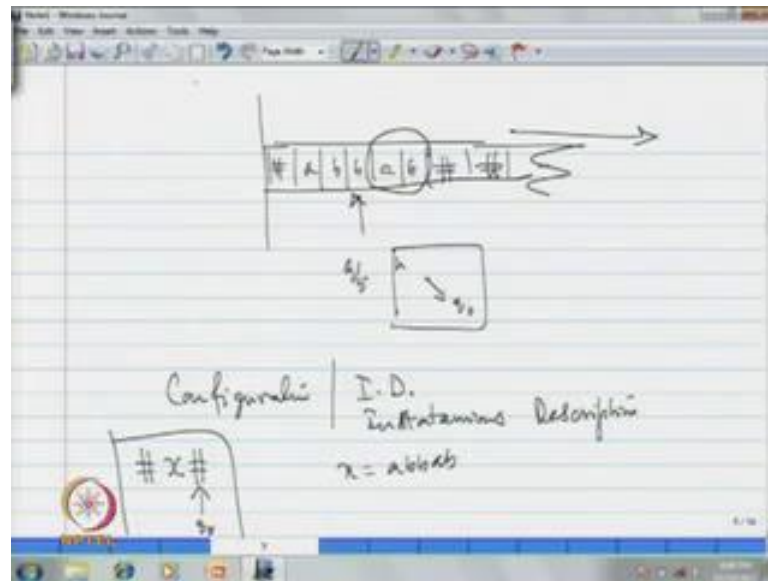
So, it is not like completing the input, in case of finite automata or pushdown automata you see that the reading and writing head on the tape is moving only one direction and end of the input, if you see the state is final state you are saying it is acceptor. But, in case of Turing machine we are allowed to move it to both left and right, now you see when you are going to finish or you know and more tricky thing here is you are allowed to print also. What are the input is given end of the thing you may go and print some more, so it is may be part of input or something newly you have printed.

So, there is nothing like you know completing the input here, so this is an automaton transitions takes place instantaneously it will not take any time. So, in the initial state you start the Turing machine and see eventually, if it is coming to the halting state that is the final state as I had mention. If it is coming to a halting state we define the notion, like the input is accepted, what are the changes it is doing let it do, it may print certain symbols, it may erase certain symbols, it may print some symbols and, so on.

But, eventually if it comes to the halting state, we say that the input is accepted and when we are expecting some output on the tape, we will expect that this particular output need to be printed and eventually it helps to halt. Because, once again I emphasize the point that this is an automaton, automaton each transition takes instantaneously even if it is 100 transitions or 1 crore transitions whatever, the total time it takes is 0 time; that means, instantaneously you give the input and eventually.

That means, immediately right after pursuing that, if it is halting it will come to the halt state with 0 time or you know if it not halting, it will go to some sort of infinite loop one can say. So, that kind of situation that you will see you with the Turing machine, now how do we give the input and all those things we will discuss of one we fix some pattern here and accordingly I will define the formal notions based on that.

(Refer Slide Time: 21:44)



So, I fix this notation that further Turing machine of course, there is a left justified right side infinite tape divided into cells, I always give the input by leaving, one cell in the beginning we will understand that why I am leaving this. So, this is blank cell, so I am representing and the input whatever is that  $x$  may be  $a b b a b$  whatever, so I give the input like this and we start the computation from the right most side of this.

So, the reading head and writing head initially will be and we set the state to the initial state that is how it is starts, this is convention we follow. And now as for the transitions on the finite control do you automaton works and eventually if the pointer shows the halting state, if it shows the halting state then I say that input is accepted, this we formulize. So, this notion I will formulize it here, so as in case of finite automaton or pushdown automaton given a particular situation that is. So, called instantaneous description is what the word we are using instantaneous description.

So, that means, at a given point of time what is there on the tape, what is the current state, where is the reading and writing head. So, this particular picture need to be represented, so this is called configuration, so the initial configuration; that means, how we are starting, you are having blank on the tape and what is the input  $x$  is given and the reading and writing head is here on the tape and what are is the current state that is shows.

So, this particular picture I need not draw it tape and whatever this reading and writing head are the finite control and all that, this is the information that I am seeing on the tape with x is equal to a b b a b here x is equal to a b b a b. So, this is a information if I somehow mention, then I know how to draw this diagram, for a given point of time say for example, reading and writing head is pointing this particular b. And say for example, state is say for example, q 5 we know that by just mentioning what is the tape content and where is the reading and writing head and what is the current state, if I give this I can always draw this picture. So, that is what the notion of instantaneous description or configuration, you know already in case of finite automaton that other places we have discussed, so this instantaneous description or configuration we define it formally.

(Refer Slide Time: 24:57)

Formally, configuration / I.D. is an element

$$Q \times \Sigma^* \times \Sigma \times (\Sigma^* \cup \{\epsilon\})$$

I.D.  $(q_0, \#abab, \#, \epsilon) \equiv (q_0, \#abab\#)$

I.D.  $(q_5, \#ab, b, ab) \equiv (q_5, \#ab\text{---}bab)$

---

Initial Configuration:  $(q_0, \#x\#)$

Final Configuration:  $(h, y\text{---}z)$

Halting

That is the current state we have to mention it is a state component ((Refer Time: 25:01)) and you see from the reading and writing head, the left side to this there will be some tape content, this is a left justified tape. So, the finite amount of information will be here that is the maximum because, this is a left justified tape, so this is an element of sigma star it is a combinational symbols of you know elements of sigma. So, the left to the reading head is the an element of sigma star and the reading head always points out some symbol of sigma.

And then when you see ((Refer Time: 25:34)) I cannot say that is just say element of sigma star. The reason is am interested about the input what is there on this, there are

infinitely many blank cells on right side because, it is right side infinite tape. And therefore, I have to take care of this situation, that is I cannot simply write sigma star here because, right to the current particular symbol it is an element of sigma star because, blank is an element of sigma and therefore, writing sigma star means, so many blanks which I am not interested in ((Refer Time: 26:08)).

Unless the reading and writing head is an blank cell, I am not going to look at this, so till this we write. So, we restrict writing leading blanks here on the right side in a given strings, so how does that set how it can be represented that is an element of sigma star and at the end I have to remove this blank; that means, sigma minus this blank. So, all those strings which are not ending in blank represented like this and of course, ((Refer Time: 26:45)) if you are at the end of this I have only blanks here, since we have restricted that.

So, for example, if this is a situation here all blanks; that means, empty string here, so that is that has gone here because, I have concatenated this set with this. So, I include that separately, so this is epsilon, so configuration or instantaneous description is an element of this set, another state component you declare what is left side to the current symbol and what is the current symbol and what is right side to the current symbol, this is to be declared.

For example, in this particular situation I have these two descriptions, in case of the first one q naught blank, we have written a b b a b. So, is blank a b b a b this is the string and the current symbol is blank and right side to that is epsilon, this is the configuration in this picture or as I had mentioned second time that is q 5 is a current state as I have mentioned and left side to the current symbol is blank a b and the current symbol is b and right side to this a b, now you see this of course, assuming that I do not have this to point out.

So, there are several blanks of course, we are not going to write that only the part of sigma star excluding means, which is not ending in blank, so this is the second one. So, these are the configurations in that I mean representing that particular this pictures, whatever I have drawn here. So, this is how the configurations are formerly defined now, what I follow is whenever the input is given on the tape I leave one blank here and the

input will be given and I start from the right side of the input, you will understand why I am normally, so far in the finite automaton.

Because, the convention is we are always moving to right and case of pushdown automaton also. But, here since we are allowed to both move left and right for some reasons that you will realize little later, after few lectures that I always start from the right side of the tape I do not have any problem. Because, I am allowed to move left and right, so I can always go to left and read this input, so I start from the right side of the input; that means, this particular blank from here I will start this is the convention we follow.

And then of course, we have to start with the initial state, now let me give some abbreviation here. So, instead of writing 1, 2, 3, 4 components like this just by distinguishing where is the reading and writing head I can always simply write this as say state component let me write like this. And this I will write it as blank a b b a b blank and the current symbol is this, I will represent this by underscore. So, this is a short hand that I use for this, you know instead of writing this many components.

Similarly, for this using that notation I may simply write it as  $q_5$  blank a b b a b this is how I will write, when by giving the underscore in a at a particular symbol, the reading and writing head is at this particular symbol that is a meaning. Now, I give some more notations here as I had mention, the initial configuration is if  $q_0$  is initial state it is of the form blank x blank here we start up, when x is the input and  $q_0$  is the initial state this is the initial configuration, any Turing machine I start, I start with this configuration.

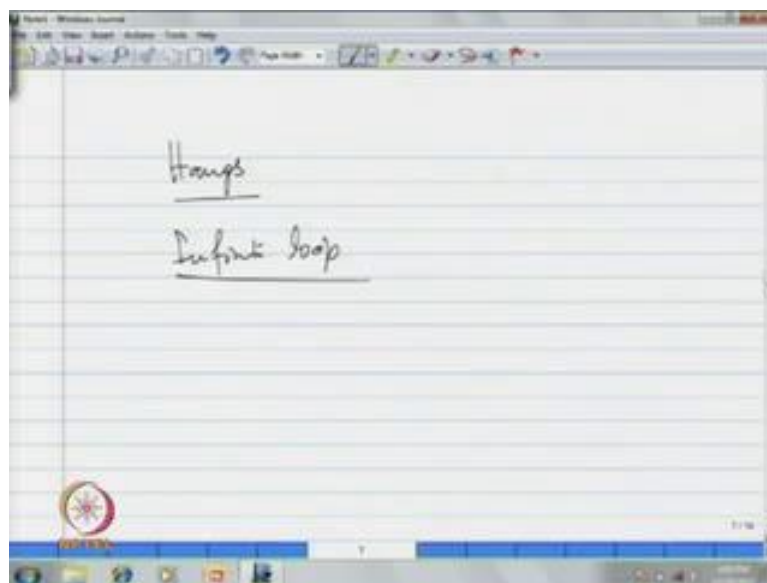
And now final configuration if exist; that means, give an input on that particular input the machine may halt or may not halt. Whenever it is halting you get the final configuration or you may not have a final configuration, so the final configuration is essentially the one, in which the state component is halting state that is how we define, so and where we will be the reading and writing head, it may be somewhere on the tape.

So, I may simply write it as say since I have used x as for input, let me write some y some current symbol a z, for some  $y z \in \Sigma^*$  and a is the current reading symbol some because, configuration should be of this form. So, if the state component is halting state, we call it as final configuration or halting configuration and now let me discuss

some more situations. For example, the machine you may be asked you know to keep moving to left.

For example, from here I have started on this input and we are moving left, left, left, as per the transition. Suppose, we have defined the transitions like this, it will come and this is left justified tape let me call this as the boundary of this tape, so it will come here, if you are further asked to move to left what will happen. We are not going to read any symbol and such configuration you cannot actually represent, which is going beyond that tape. I may say that whenever it is coming and hitting this left boundary I may call it as the machine hangs. So, we call Turing machine hangs if you are asked to in a particular transition, if you are asked to cross this left boundary in a Turing machine, so that is called I may call this situation of hanging.

(Refer Slide Time: 33:30)



So, hanging situation is that the machine hangs Turing machine hangs, if you are asked to if a particular transition, we are asking means particular transition is force the machine to move to left. Further when it is at the left boundary, that is what is hanging and of course, one more thing that you can quickly realize is where you will not have this sort of halting configuration. Because, here in a such a situation this machine that particular input cannot have halting configuration because, the machine is hanging.

And one more instance is you know, you may be asked to go to right in which case you will never come back. So, if this is a right side infinite tape, so you can always find a

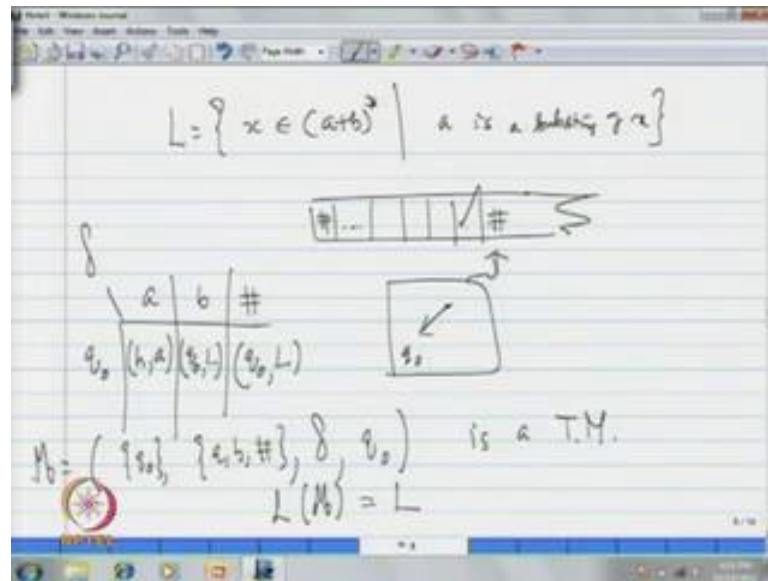
new cell and, so on and, so on and thus in the situation, it will go through that infinite side. So, you will not have halting configuration or you can always create on a finite tape you can go back and forth for example, the transitions are defined such a way that.

For example, if I am here I may be asked to go to somewhere for example, 2 cells left and then you I may be asked to go to some 3 cells right, 2 cells left, 3 cells right and, so on. For example, if I keep doing this again you see that you cannot have halting configuration because, 2 cells left, 3 cells right; that means, from a current position finally, I see that I am 1 cell to right and in each such loop am going 1 cell, 1 cell to right and, so on.

In such situation I cannot have halting configuration, even if I change this idea that 3 cells to left, 2 cells to right. That means, from the current cell I may go 1 cell to right and, so on, after some time it will hit the left boundary because, this is left justified tape wherever I am left to that there are only finitely many cells and thus the machine hangs. So, if you do not want the machine to halt; that means, if you do not want the halting configuration.

You can actually define in two ways, one making some sort of infinite loop going to the right or oscillating on a finite tape. In such situation you may say that it is going to infinite loop or the machine, which is touching the left boundary in which case we may say that the machine hangs, now infinite loop. So, this situations I hope now you understand with this, now let me give an example for accepting example of a Turing machine accepting this language.

(Refer Slide Time: 36:01)



A simple language that I considered, that is set of all strings  $x$  in or the set  $a$   $b$  such that  $a$  is a symbol of  $x$ . So, that means,  $a$  is a substring of  $x$ ; that means, there is an  $a$  in  $x$ , you see this is a regular language we have constructed finite automaton for this language. Let me construct a Turing machine, as I had mention that this is ultimate a powerful device, so what are all the regular languages you can always constructed Turing machine, what are all the contrastive languages for which also you can construct a Turing machine.

So, this is such a general device that we are introducing and as a more automaton, which can accept a any of the languages that we have discussed, so for regular languages, contrastive languages, etcetera. Now, for this let us first discuss on this picture and then corresponding to that we can write the transitions, so the finite control let me call  $q_0$  is the initial state. So, the point is pointing to this what I do, we start with here am leaving one blank here, now I keep searching for the symbol  $a$  on the tape.

So, input is given here assume input is given here, you come to left in  $q_0$  just to distinguish that you have started. Because, this is a blank and after sometime you are encountering one more blank, if you do not for example, encounter  $a$ , so let me change a state, so that I understand that the computation is started. So, come to left from this blank and change this state for example, to  $q_1$  and now you see that this particular symbol is  $a$  or not in this particular cell.



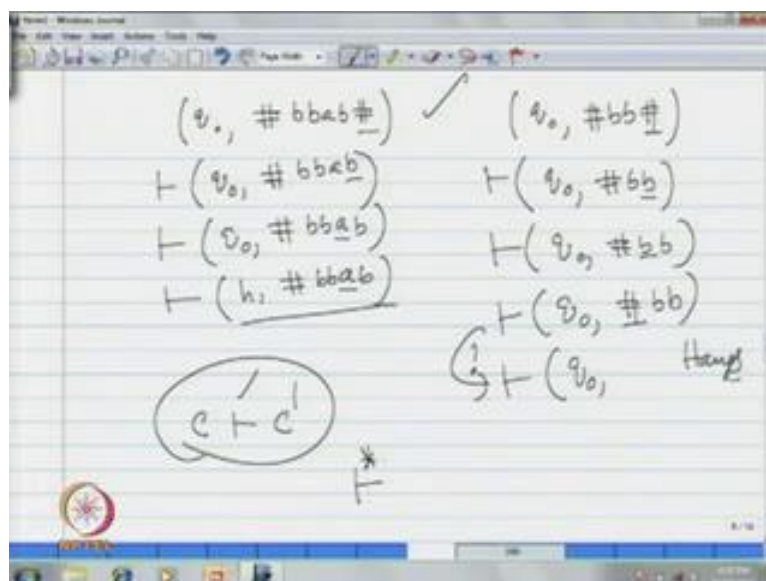
If it is a then I am happy that there is an a in the input and simply you change it to halt state and then the computation completes and that particular input will be accepted. If this is not a further in that state  $q_1$ , you may ask to go to left, in this situation you need not actually change the state. Because, state we define in case of automaton to remember something to create such some sort of memory, here I need not remember anything because, the competition started that is only signal with  $q_1$  am having.

So, if a is there I am halting, if a is not there I go on further left and search, if a is there and, so on. But, only thing is I have to cross check, whether I have encountered you know this blank, if you do not want to hang the machine or you know you keep going to left and if it hangs there is no problem. Because, if you do not want to accept a string, you can hang it or you can put it in infinite loop or if you are searching for this blank, you can search for this blank and you can go to right side further infinite loop.

So, in which case of course, I do not require any new state as well, so I am in  $q_{naught}$  if I want to hang the machine, I may simply define it like this. So, the symbols are a, b this is a special symbol always there I am in  $q_{naught}$ , assume  $\Delta$  at this blank, so I have to go to left I will continue in  $q_{naught}$  go to left. If I am encountering b, I will be in  $q_{naught}$  and go to left and if I am encountering a I will halt it when you are halting, you may take one cell to left or you may take one cell to right there is no problem or you can print something and halt may be some a is there, so I will say print a.

So, this is how if I define  $\delta$ , now you see this quadruple only one state I have chosen  $q_{naught}$  and  $\Sigma$  is now a, b blank this is always there blank and  $\delta$  is defined here and there is only one state. So, this is initial state is a Turing machine and we see, the language accepted by let me call this is  $L$ , since we are calling machines let me use  $M$  the language accepted by  $M$  we see as desired. If I call this as  $L$  this is what is the language, let us do some computation on this and under and see how this is happening.

(Refer Slide Time: 40:46)



For example, you are given the input say blank b b a b for example, so this is the initial configuration. As earlier with this notation in one step let me use this one step relation, so in  $q$  naught if you are reading blank, it will be in  $q$  naught it will change to  $q$  naught and go one cell to left there is a definition. So, the current state is again  $q$  naught blank b b a b I am not going to represent this blank now because, the leading the right side this blanks will not be represented in a configuration.

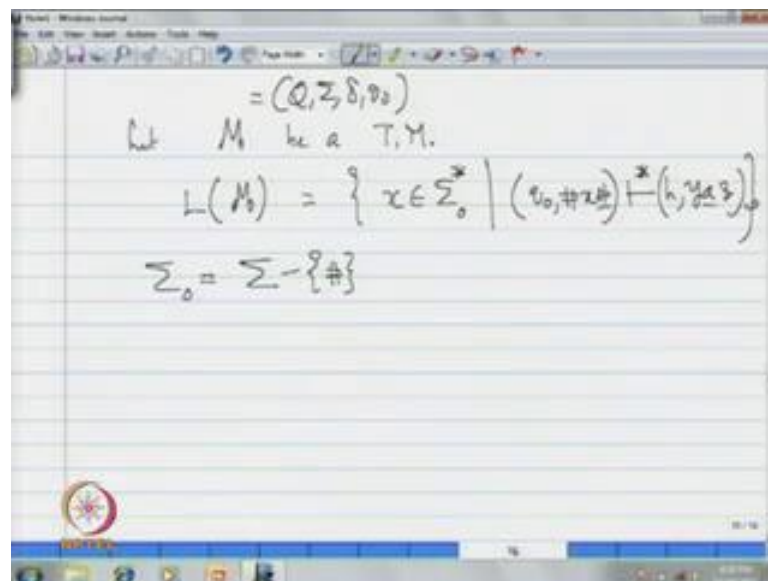
So, this is a situation again now in  $q$  naught as for the definition if you are reading b it will go to left changing to the state  $q$  naught. So, I will continue to the state  $q$  naught and now this is b b a, this is a current symbol of course, this is b will be there we have not changed anything. Now, we look at in  $q$  naught if you are reading a you changed to halting state and print a, so at this place we are printing that this is or writing, so whatever ultimately you write that will be shown, so what you shown here again a.

So, this is the current place, this is the configuration, this is the initial configuration and now the computation. Because, this is the one step relation I have written for each transition what is happening I have updated on this and see we have got a halting configuration. And thus this particular string we say is accepted, if you take a string in which there only b's let me takes a two b's, there is a initial configuration I have to start with, so what will happen since it in reading blank, it will come here b b and since in  $q$  naught again this is in  $q$  naught, it will come to this b and in  $q$  naught again b.

So, it will come to this place, so b b and in q naught when you are reading blank that will come to left, so but coming to left means the here is a boundary. So, I do not have anything to write here, so after what situation will come I do not have any configuration here. So, this machines hangs here, thus if you have only b's in a string that will not be accepted, let me give you the formal notion of acceptance using this notation, as earlier the reflexive ((Refer Time: 43:38)) of this one step relation.

The one step relation is on the given and say this is or again as usual, we define this is a relation or configuration the set of configurations. If we apply one transition on a particular configuration to give another configuration we say for example, say this is c and c dash, these two are related by this one step relation, if you can get c dash the configuration c dash from c by applying one transition. So, this is definition as earlier and we use this to represent the reflexive trans to closer of this one step relation.

(Refer Slide Time: 44:24)



Handwritten mathematical definitions for a Turing machine  $M$ :

$$M = (Q, \Sigma, \delta, q_0)$$

Let  $M$  be a T.M.

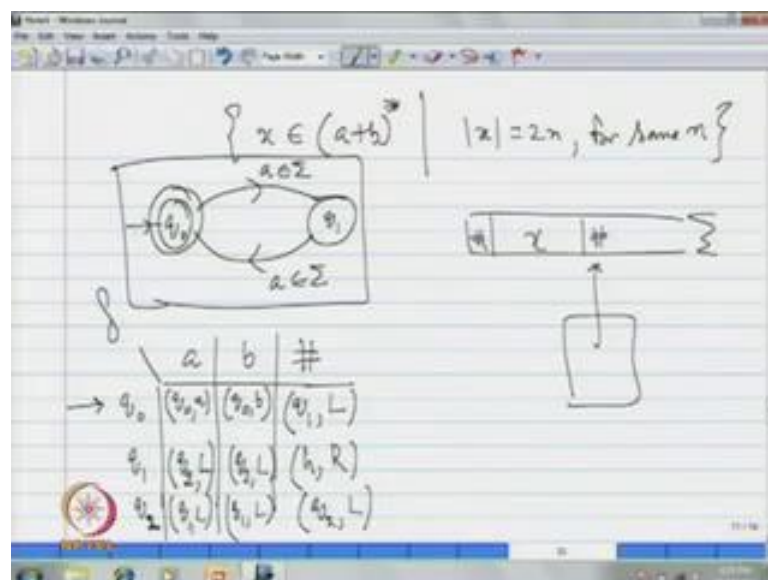
$$L(M) = \left\{ x \in \Sigma_0^* \mid (q_0, \#x) \vdash^* (h, y\#) \right\}$$

$$\Sigma_0 = \Sigma - \{\#\}$$

And using this, let me define formally let B be a Turing machine language accepted by M denoted by L of M is set of all strings in let me write sigma naught star. So, where m is a Q, sigma, delta, q naught as such that you start with this as input, the initial configuration. After finitely many steps you will get a halting configuration, in which case you may have something on the tape, so from the initial configuration you get a halting configuration.

So, all those strings, so now, what is sigma naught because, let me write where sigma naught is except this blank symbol, whatever is the input alphabet you take this string. So, this is what is formerly let me write the language accepted by the Turing machine M, the language accepted by M L of M, now using this definition you see that any string, which is having a ((Refer Time: 45:55) if you give to this it will halt, once it encounters that particular a, if it is not encountering a it will keep on going to left and you see the machine hangs. And thus only those strings which are having a at least one a is if it is existing in the input on the string, so only this Turing machine halts and thus the language accepted by M here is L.

(Refer Slide Time: 46:37)



Now, let me give one more example, the set of all strings these are all regular languages that we have observed  $x$  in say or  $a$   $b$  does not matter, such that  $2n$  for some  $n$ . So, all those strings are of even length, so what I have to do here, once again let me discuss with this tape and then once you are familiar with this because, I am reading from right to left from this side. So, then you will be familiarize with, so the input is given to you I have to see whether the it is of length  $2n$  or not.

As earlier in case of finite automaton, you have two states in the beginning you will be maintaining say for example, in  $q$  naught when you encounter a symbol we are going to say for example,  $q_1$  which is understanding, which is recognizing this is a hard line string. And once you read one more symbol we change it back to  $q$  naught, this is how

the finite automaton you have maintained their let me write that, so that. So, the initial final state is considered to be this, under all the symbols from this you may change to say  $q_1$  and all the symbols you may change back to this if you are reading a symbol here.

So,  $a$  belongs to  $\Sigma$  for all  $a$  in case of finite automaton, so we follow it this way, the same thing we can write and see. Here I have the symbols  $a$ ,  $b$  blank, in  $q_0$  naught let me start looking it, in the beginning I have a blank, so I will come to this the first symbol after reading, then I will change it to say for example, another state. So, to distinguish between this blank and this blank, you can change the state for example,  $q_1$  and go to left.

And when I am in  $q_1$  I am encounter  $b$ ,  $a$  or blank because, if this is empty string I am a straight away encounter blank. For example, am encountering blank in  $q_1$ ; that means, this is even line string and therefore, it should be accepted, so in  $q_1$  if I encounter this blank. So, I should halt may be I will print something or I may take a right move does not matter, so this is how you can do, if we are encountering a symbol say for example, I have encounter  $a$  or  $b$  I will count that now change it to for example, say  $q_2$ .

If I am in  $q_2$ ; that means, I have right one symbol, you may change this symbol or you know let me go to left. Similarly, even if I read  $b$  in, so let me just to  $q_2$  and go to left, now in  $q_2$  the meaning of this I have write one symbol, if you encounter blank you see this is essentially only one symbol say  $r$  length here. So, I may asked to go to further say left, so that it will hang, if I am encountering one more symbol I will put it back to  $q_1$ .

So,  $q_1$  and go to left and  $a$  I am not going to encounter  $a$  and  $b$  in case of  $q_0$  naught, so I can arbitrarily define anything. So, let me for example, define say  $q_0$  naught  $a$  does not matter because, I am not going to encounter  $a$  for our convention, so this is the transition map I have defined here. The states I have considered 3 states here,  $q_0$  naught,  $q_1$ ,  $q_2$ , the symbols are  $a$ ,  $b$ , blank and the initial state let me declare  $q_0$  naught as initial state.

So, this quadruple accepts I claim that this quadruple accepts the desired language, what do you do as an exercise take various types of inputs. Like for example, you take empty string or may be some string, non empty string, which is of even length some non empty string, which is of odd length you take. And see the way that I have done the computation, you do it and realize that this accepts this language or the desired language.

You know let us discuss one more example, if you consider because, when I am talking about a sub string. Because, we have seen that when you are reading from, in case finite automaton we read from left to right.

(Refer Slide Time: 51:33)

$\{x \in (a+b)^* \mid ab \text{ is a substring of } x\}$

	a	b	#
$\rightarrow q_0$	A	A	$(q_1, L)$
$q_1$	$(q_1, L)$	$(q_2, L)$	$(q_1, L)$ ✓
$q_2$	$(q_1, L)$	$(q_2, L)$	$(q_1, L)$ ✓

$A = (q_2, b)$

When I say for example, set of all x in say a b star such that a b is a sub string of x, if you consider this example. In case of finite automaton I see that whether a is encountered when a is encountered, immediately after that whether b is encountered that we have to see. But, as for our conventions since we are coming from right side, first we have to see whether b is encountered, then immediately whether a is encountered this is how we have to design.

So, let me do this construction also, say we have started with initial state  $q_0$  in the blank. So, on a tape when the input you are coming let me changed to  $q_1$  conventionally in coming to left, if I have encountering this I am not worried I will continue in  $q_1$  going to left I do not remember these things. So, when I am in  $q_1$  if I encounter a I will continue in  $q_1$  and go to left. But, where if I got b then I will remember this so; that means, I will changed to  $q_2$  and go to left, if I am in  $q_2$ ; that means, I have encountered 1 b.

So, in the state  $q_2$  if I have encountered some more b's I can safely continue in  $q_2$  itself, so; that means, any way I have at least one b, I have read. Now, after that left to this whether I have a that is what is important, so in  $q_2$  if I get a I can halt now, so let

me print b a itself, so that the machine will halt their itself. Now, let us look at the situations, in the  $q$  naught I am not going to encounter this a and b because, in the beginning, in the initial state I will start with this I can arbitrarily define anything here.

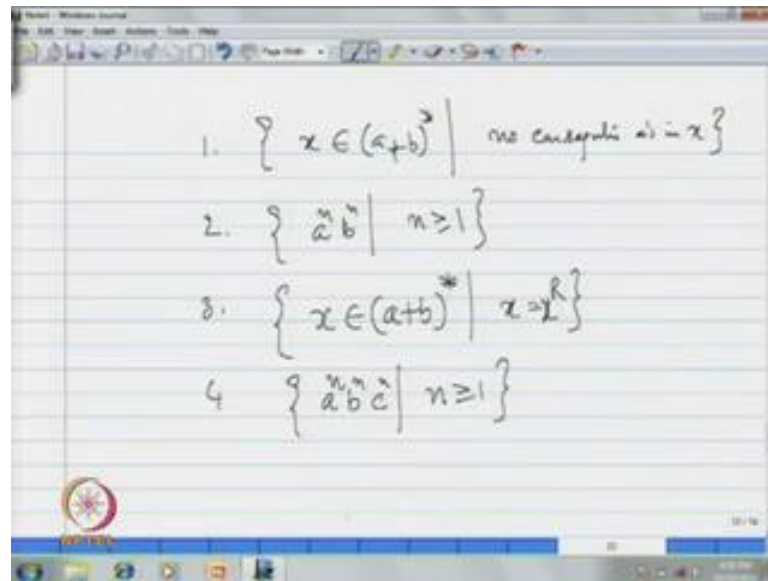
Because, I have to define because, the transition function is a total map from  $q$  cross  $\Sigma$  for every state and symbol I have to define something, let me write A for the time being arbitrarily I can define anything. So, now, in  $q_1$  if we encounter blank; that means, in  $q_1$  I encountered blank means I would have, so many a's or may be empty string. Because, in if I have encountered a I continue in  $q_1$ , so that way the input may be only a's or may be empty string, in which case I am continuing in  $q_1$ , so in that case I should not accept this string.

So, what I do you can asked to go to further left, so that it will hang or you can asked to go to infinite loop whatever it is. So, what we do say let me write  $q_1$  left it will hang, now in  $q_2$  I have encountered blank, if this is the situation what is the meaning of this I have encountered a b or, so many b's from when I am coming from the right side, but I have not encountered a. So, in which case also the same thing, so I have encountered blank means I can ask further to go to left, so  $q_1$  left.

So, from blank if it goes further it will hang, so when or we can define anything here, but it has to hang or go to infinite loop in these situations, that is how you have to define. But, here I can define anything arbitrarily, so where A I have whatever we can give say for example,  $q_2 b$  let me give something I have to give, so arbitrarily, so this and this I may fill with this. Now, you has again as earlier what you do you take the initial configuration with what are the input that you want to consider and do the computation and see, like what are the respective final configuration if they exist.

If there is no final configuration; that means, either it will hang or it will go to infinite loop, whatever is happening you just observe it. And observe that this particular machine accepts this language and in this line up may be some more regular or contrastive languages, you may try to construct Turing machine let me give one or two exercises as a home work.

(Refer Slide Time: 55:53)



Set of all those strings for a b star such that a a is not a sub string, so or there is no consecutive a's, or you can try this. You know this is a contrastive language a power n and b power n such that say n greater than or equal to 1 or you can try you will discuss some of these things. In next class x equal to x power r is a palindromes we have discussed and what are these a language I have mentioned, which a power n b power n c power n, which is you cannot have pushdown automaton for this you can try this also. How to remember, because you know in case of finite automata how to create the memory and do I mean this computation. So, you try these examples means constructed Turing machine that accepts these things and will discuss further on this in a next class.