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

Module - 11
Turing Machines
Lecture - 3
Combining Turing Machines

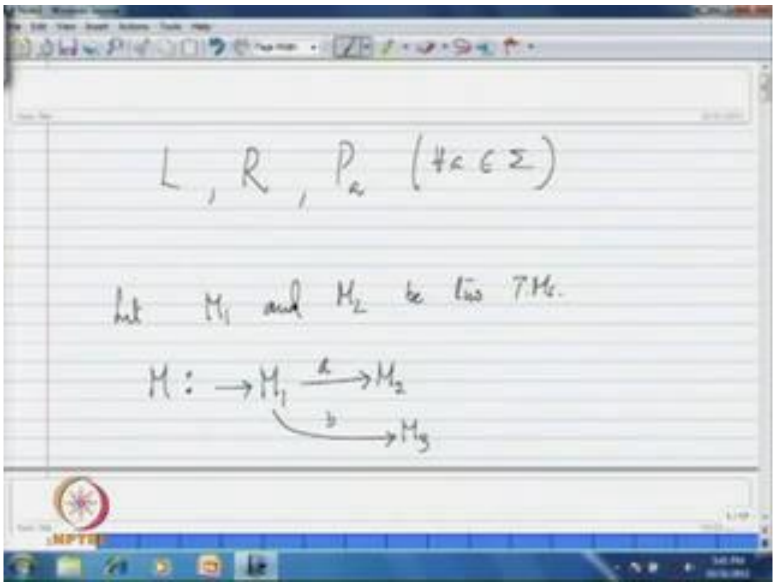
Hi, now we this is third lecture on Turing Machines, in a very first I have introduce the notion of Turing machine and as a language acceptor, we have understood like how a language can be accepted by Turing machines certain examples I gave. And how to give the output the notion of giving output in Turing machines are also discuss, we have fixed some notations for the and a computable function is introduced in the last class.

So, using Turing machine we have two things that one is computing a function from strings to strings, so called computable function are Turing computable function more precisely or as a language acceptor, where we talk like Turing acceptable functions. Turing acceptable languages are recursively enumerable languages this is what I have introduced. Then I started discussing are about how actually a complicated Turing machine a bigger Turing machine can be constructed through some of the Turing machines that we have already constructed.

So, for which we have started looking at very basic machines, I have fix the notation for the basic Turing machines; namely left the left moving Turing machine right, moving Turing machine and printing Turing machines; these are the basic three important steps that we pursue with the Turing machine. That means, at a given position, the Turing machine whatever it is reading it will take one left move and simply halt. So, the Turing machine, such during machine I am representing by L.

Similarly, the Turing machine whatever it is reading, it will take one right move and simply halt, so that we are representing it by R. And I am calling for a given a in sigma the alphabet the P a the Turing machine that print simply a, whatever it is reading in the current cell there it will print a, in the symbol a and it will halt their.

(Refer Slide Time: 02:34)



$L, R, P_a \ (\forall a \in \Sigma)$

Let M_1 and M_2 be two TMs.

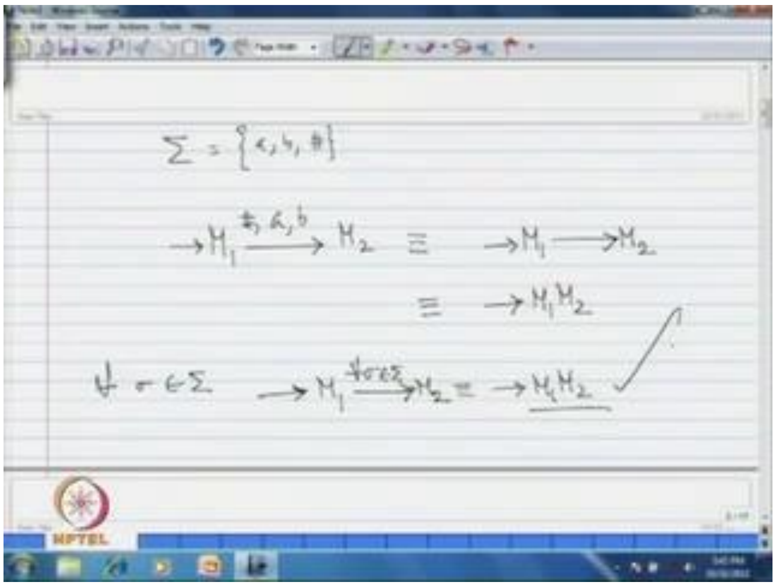
$$\begin{tikzcd} M : \arrow[r] & M_1 \arrow[r, "a"] \arrow[rr, bend right, "b"] & M_2 & M_3 \end{tikzcd}$$

So, this are the three basic Turing machines and L R P a for this is for all a in sigma, so these are three basic Turing machines that we have liberated. Now, when I am talking about combination of some smaller Turing machines, first let me introduce a notation what it indicates the meaning of that and those things, let us first discuss; that is as follows, take two Turing machines. Now, if I use the notation, this M 1 say symbol a, let me right here M 2, the meaning of this is, this arrow indicates that this; we start the process at the machine M 1, this arrow indicates that…

And the Turing machine M 1 with what are the input it will pursue and then whenever it is halting if it is reading a, that is what is indicated on this arrow. Whenever it is halting if it is halting by reading a in wont halt rather it will connect to the Turing machine M 2 and pursue the job what are that M 2 is doing. Now, whenever M 2 halls, this entire Turing machine will halt, so that means, essentially this Turing machine, if I write M the behavior of M is M 1 followed by M 2 with a condition that. Whenever M 1 is halting, if the current reading symbol is a; then it will proceed to connect to M 2 and process the input with that, so that is how the process is…

Now for example, when it is reading now let me give say for example, whenever it is reading b for example, if you want to do it as M 3. Now, the you have a choice here when M 1 is halting if it is reading a, it will simulate M 2 and whenever M 2 is halting this will halt and whenever M 1 is halting if it is reading b, then it will simulate M 3 on the rest of the input, what are the current input evaluable on the tape. And then whenever m three is halting then in which case this will halt; so this is what is the phenomena here the composition, when I am writing this is how it is the arrow be the condition here.

(Refer Slide Time: 05:32)



$\Sigma = \{a, b, \#\}$

$\rightarrow M_1 \xrightarrow{\#, a, b} M_2 \equiv \rightarrow M_1 \longrightarrow M_2$

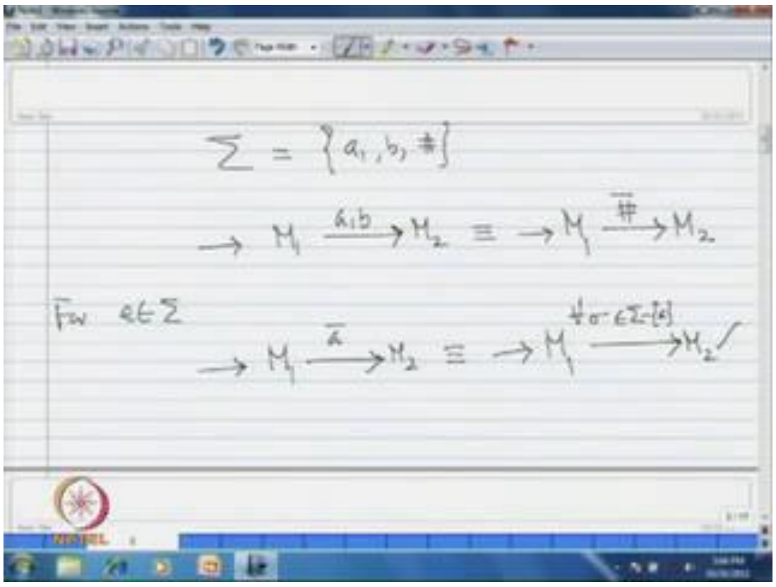
$\equiv \rightarrow M_1 M_2$

$\forall \sigma \in \Sigma \quad \rightarrow M_1 \xrightarrow{\forall \sigma \in \Sigma} M_2 \equiv \rightarrow \underline{M_1 M_2}$ ✓

For instance if sigma is equal to a b blank, then when I am combining M 1 this is a initial machine let me call and for a, I have define say for example, M 2 and for b also I have define this is M 2 and for blank also if this is a situation; that means, for all symbols of sigma if I am doing, this I am a simply write M 1 is a starting machine. Then without any condition here on the arrow I do not write anything I will simply write like this, this also I will write a this as just simply M 1 M 2, that may follow this conventions.

So, for that means, essentially here I wrote sigma is a b blank, so in general for all symbols in sigma if I have the situations these two machines are connected for all symbols of sigma then this is what is M 1, M 2, this is for all sigma in sigma. In the situation, we are writing M 1, M 2, we do not put that arrow in between unconditionally. Whenever it is halting M 1, it will simply connect to M 2, it will simulate and it because for all input this is doing, therefore this is the situation.

(Refer Slide Time: 07:14)



$$\Sigma = \{a, b, \#\}$$

$$\rightarrow M_1 \xrightarrow{a, b} M_2 \equiv \rightarrow M_1 \xrightarrow{\overline{\#}} M_2$$

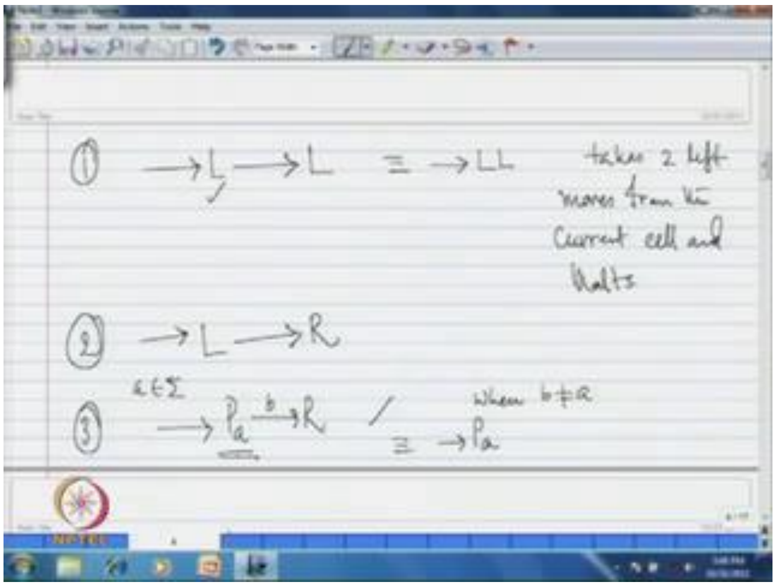
For $a \in \Sigma$

$$\rightarrow M_1 \xrightarrow{\overline{a}} M_2 \equiv \rightarrow M_1 \xrightarrow{\forall \sigma \in \Sigma - \{a\}} M_2 \checkmark$$

Now, let me give one more notation also here, which we may very frequently use, that is for example, I have situation like this, I have a, b, blank and I am connecting M 1 to M 2 for a and b. So; that means, for nonblank symbols of sigma if I have the situation, then I may also write this is, because here only two symbols writing two symbols is not a problem. If I have 10 symbols here writing a 1 gamma, a 2 gamma and so on a 10 and other then the blank essentially, this I may abbreviate are use the notation that, which are not blank here M 2, this is how I will write.

This bar f I am giving other then blank for all symbols of sigma this is connected to M 2, that means for fixed a in sigma, if other then sigma if this is connected. So, this M 1 a bar M 2, I mean this is M 1 connected to M 2 for all symbols of sigma minus that particular symbol a; expect a for all other symbols this will be connected to M 2, that is the notation. So, I will use this notations very frequently, when I am composing the some smaller Turing machines, so let me just look at some examples.

(Refer Slide Time: 08:58)



① $\rightarrow L \rightarrow L \equiv \rightarrow LL$ takes 2 left moves from the current cell and halts

② $\rightarrow L \rightarrow R$

③ $a \in \Sigma$ $\rightarrow P_a \xrightarrow{b} R$ / $\equiv \rightarrow P_a$ when $b \neq a$

You know the Turing machine L, if I connect this unconditionally to for example, L then what it does this Turing machine, from the current reading cell it will take one left move, it will halt that is the first machine. But unconditionally whatever it is reading again since it is connect to the Turing machine L, what will happen it will take another left move and then as L is halting after one left move, so this is the Turing machine. So, that means, the Turing machine, if I write L L is a Turing machine that takes two left moves from the current cell and halt, so that is what it does.

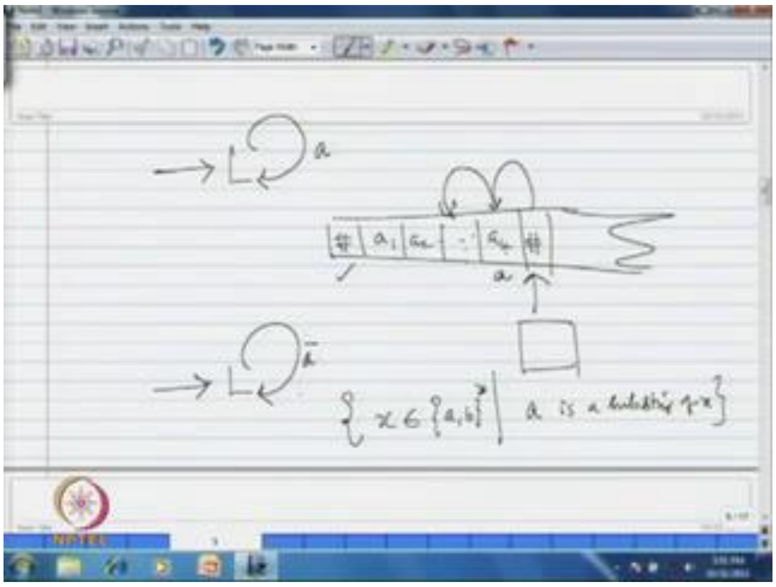
And for example, it you take this machine, this the effect of this Turing machine is null, because from the current cell it will take one left move and since unconditionally it is connected it R it will take one right move and it will halt. That means, from the current cell it will take one left move and one take one right move and halt, but actually here is a problem when I am saying left move if it is…

So, even in the first case or in the second case here the situation is if it is already at the left boundary for example, give the if it is are the already are the left boundary then the left move will make the machine hangs, so the effective Turing machine will hang. So, if you have an assumptions that at least one cell is available left to the current cell, then of course, this machine the effective is nothing. And in this case, if you have first case it you have two cells available to move to left then it will take two left move.

So, the current cell and halts, otherwise it hang of course, two cells are not available in the first case that hangs. So, that is what is here if you write this machine, so I let me fix a for a in sigma some fixed a, so I print a and take a right move, so that may be written like this, because from the current cell in the current cell it will print a and it will take a right move and halt. So, I hope the things are clear this is unconditionally I gave here, but if I give something here. For example, if I give b here, where b is different from a, when b is different from a, then what will happen here this will simply print in the current cell a and halts.

The reason why, when b is different from a you know in the current cell it is printing and simply halting and this condition will not be satisfied and it will not take the right move and does the effect of this Turing machine is simply printing a in the current cell and halt. So, it is nothing has, but just p a if b is different from a this machine is nothing else, but you know the effect of this is p a, because b is different from a in the current cell you have a. So, it is not going to take, it is not going to connect this Turing machine.

(Refer Slide Time: 12:41)



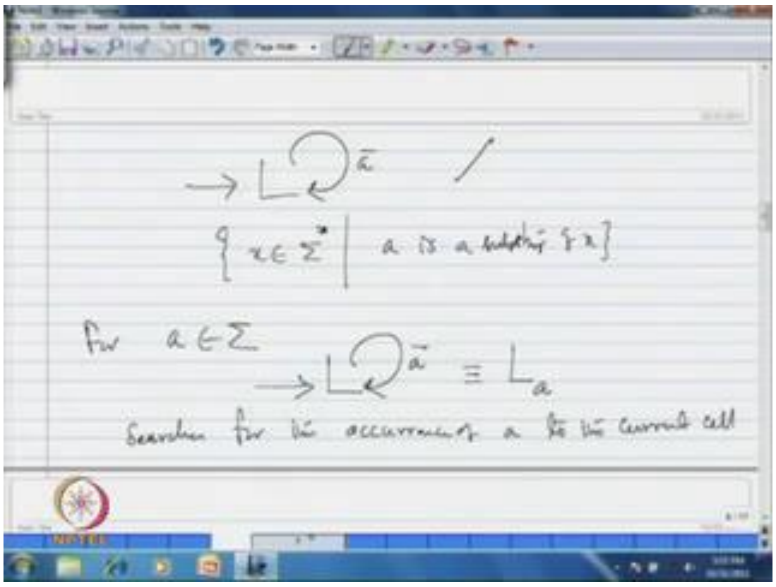
Now, let us look at this if I give this with symbol a, for a in sigma first, the effect of this Turing machine is in the current cell, very first it will take one left move on the tape, say let me say a 1, a 2 and so on, a k is the input given to you. And assume this is the input suppose if I am like this, now it will take one left move here, it will see if it is a, it will take one more left move another a k minus 1 if it is also a, then it will take one move left move and so on.

So, if the input is sequence is of just a’s, then it will come till the end and it will come to here and since it is not a, it will simply halt here; see this composition with a condition is very important to understand. Once again this is take the left move you come here and now if the current available symbol in the cell if it is a, then it will take it is connect to the same machine that is L, so it will take another left move, so it will come here. So, when this is a, this takes one more left move and suppose here the a k minus 1, that the a k minus 1 the symbol if it is not a then header point it will halt.

So, when I am giving the input like this whenever your encountering non a; that means, a symbol which is not equal to a on the tape, then the machine the halts, otherwise as long as you are getting a’s you will keep moving to the left, that is how it is. For example, if I give the machine this with a bar this is on a negation of that; that means, on the tape, if it is getting other symbols, than a it will move to the left keep moving and whenever it encounters a, in it will halt.

So; that means, this Turing machine, now we have constructed earlier, so this actually accepts those strings x for example, let me consider here a b star is a fixed for example, where a is a substring of x. Now, you look at here I have fix the alphabet to be a b star. So, here the symbols in a Turing machine that I am going to construct you may consider a, b, blank like that. Now with this notation I need not say that it is over the alphabet a, b, you can have some another 10 symbols or 100 symbols or whatever.

(Refer Slide Time: 15:56)



$$\begin{tikzcd}\arrow[r] & L \arrow[loop right, "\bar{a}"]\end{tikzcd}$$

$\{x \in \Sigma^* \mid a$ is a substring of $x\}$

for $a \in \Sigma$

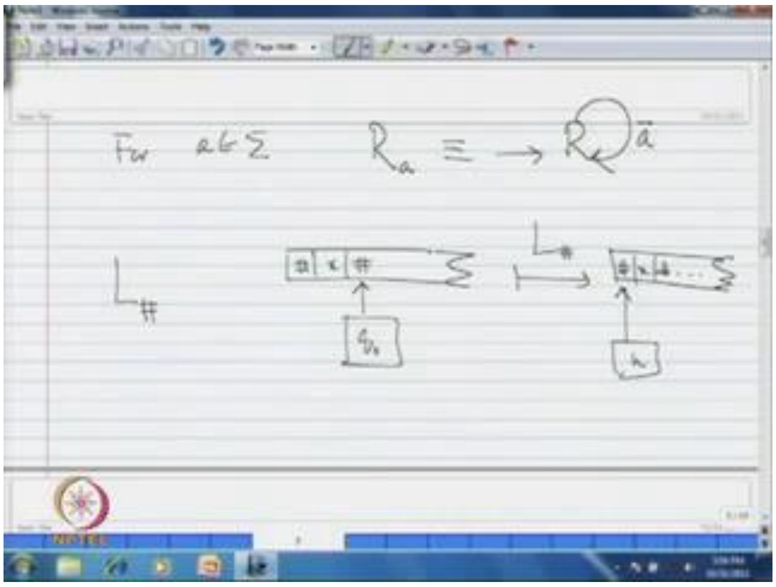
$$\begin{tikzcd}\arrow[r] & L \arrow[loop right, "\bar{a}"]\end{tikzcd} \equiv L_a$$

Searches for the occurrence of a to the current cell

So, I can simply say in this situation this Turing machine, If I write with this notation, the advantage is I can simply say x belongs to say some sigma star, such that a is a substring of x. So, those strings which are having a that you are collecting in this language, so that is how we can represent this. Now, similarly you can think of you know the connecting the Turing machine which makes one right move to itself conditionally unconditionally are whatever.

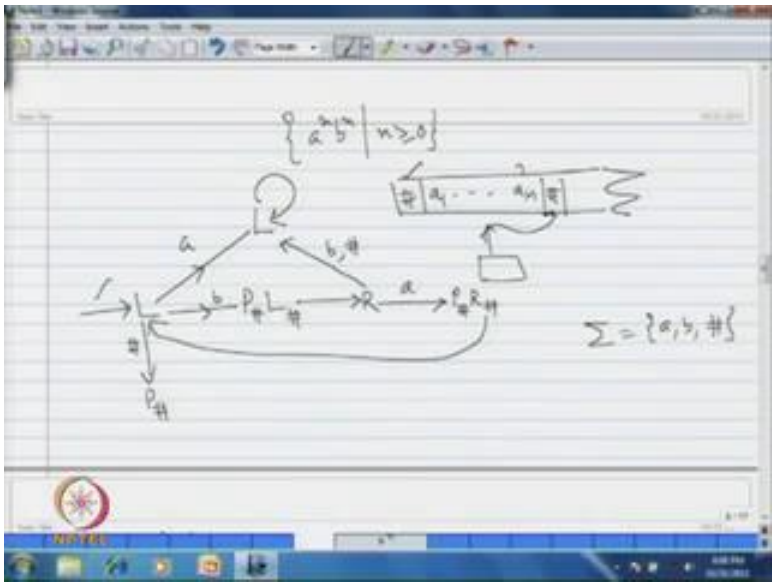
Let me give a short notation for this Turing machines, which will often use, here for a fixed symbol, let me right that we a in sigma, if I consider the Turing machine a bar from L is connected to itself, I may write this Turing machine by the symbol L a suffix a. That means, this searches for a from the current cell to it is left whether there is in a, whenever a comes that halts. So, this Turing machine is searches for the occurrence a to the current cell, so if it is available it will halt there, if it is not available of course, when it is going to left it will hanger.

(Refer Slide Time: 17:55)



Similarly for a fixed a in sigma, I will write R a to denote the Turing machine which takes a right move as long as it is encountering some other symbol then a, so this is keep going to write when till it receives this. So, in particular if I say L hash is a Turing machine on the input, if you give like this suppose some input your giving normally we do not give blank in the input, if this is the input tape, this machine this performs this L hash performs. So, here it will halt of course, this are all blanks in after words need let us to mention, so that is how the situation L a I mean L hash. Similarly what are is the symbol you write and if it is the suffix is what is it is performing.

(Refer Slide Time: 19:24)



Now, using for example, these Turing machines let me discuss a Turing machine that constructing for example, a power n b power n such that n greater than equal to 0. For which you remember that I have consider I think about 7 8 states to draw the to write the Turing machine, the transition I have the Turing machine. Now, let us follow the same logic and see you like how we combine the Turing machines x is input given to you and this is how we are starting are you can write a 1, a 2, a n this is the current cell.

Now, we take a left move and see whether the symbol is a or b, so based on that we take a decision, so we have to encounter b to cross check whether there is a. So, take a left move now you have two possibilities if you are encountering a then what to do that will see that we should not we do not want to halt the machine, in which case am I take left move keep taking left move unconditionally it will go on hanger. Now for example, if I get b, then we are printing their blank, so this is what the machine that prints blank and then go till end of this that is L hash as per our convention and then take one right move.

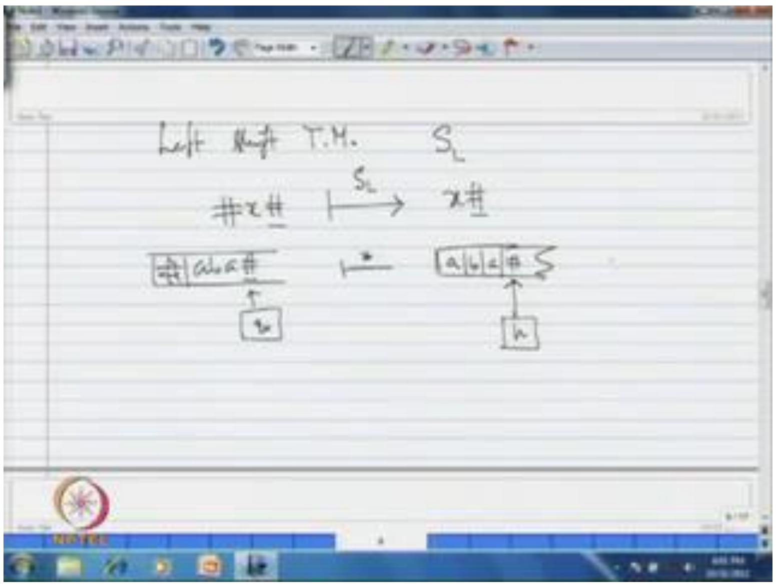
So, take one right move and then see whether you are getting a there, if you are getting a your happy you print blank there, then take r hash; that means, it will come to till this position, because here you may this is blank, so let me make R hash. So, from R hash again you pursue the same job; that means, connect back to this if in the beginning I mean the beginning or after few hydrations, if you are getting blank then you are happy you may just halt. So, halting means you whatever you perform even if I do not defining here. So, any way let me say print blank.

Now, the situation is then I am not getting a here when I take right move; that means, it can be blank or b these are the possibilities I can write a bar. So, in which case again if I put this loop that it will keep going to left and hangs and you see now how elegant is this picture. So, now you can quickly understand that how the Turing machine is working, now this are the Turing machines that in which we have use the Turing machine L, L hash, P hash, R.

So, these are the Turing machines that we have use the basic Turing machines and what are the Turing machine their after we have constructed and we have constructed this Turing machine. Now, this is the starting Turing machine indicated by you know this arrow this is where you have to start and now give the input in the prescribe the format and pursue the job. Now, you see that this Turing machine precisely accepts the language a power n, b power n such that n greater then equal to 0.

So, instated of you know giving the transition map very that huge number of states by following this composition of the composition this kind of notation I have followed to construct such an elegant representation of the Turing machine which accepts a power n b power n. Now, let me give one more some useful Turing machine, that is right shift Turing machine or left shift Turing machine let me discuss.

(Refer Slide Time: 23:23)

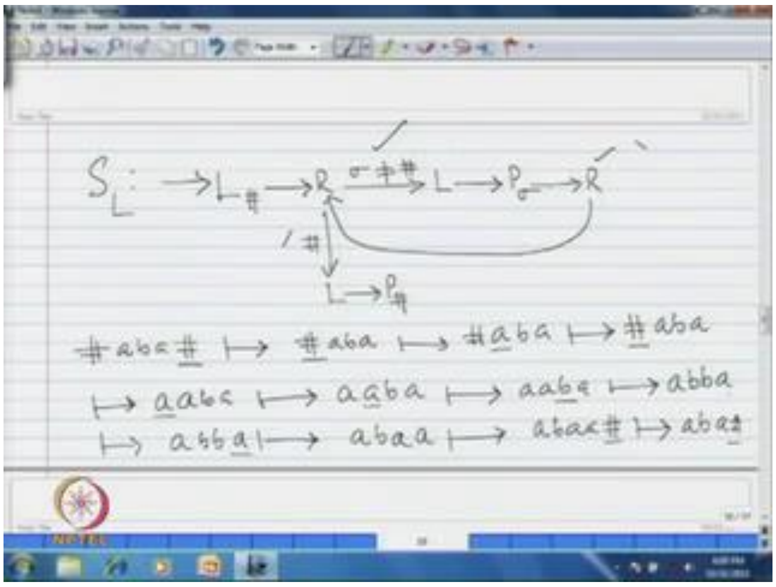

Left Shift T.M. S_L
$\#x\underline{\#} \xrightarrow{S_L} x\underline{\#}$
$\# | a b a \underline{\#}$
$\vdash^*$
$a | b | a | \# \dots$
s
h

Left shift Turing machine, let me right this as S L may be what is the job of these, you give an input of the form x, this and what is does, it transforms S L transforms this one. That means, the entire input what are that you are giving it will shift one cell left for example, here say a b a this is input given to you and suppose this is available on the tape for example, like this are whatever it is. So, this after finitely many steps you will see of course, with the initial state whatever you call this a b a is like this and finally, it will halt again here.

So, when I am talking about this Turing machines, I am not the prescribed input is for the entire to the machine, but a particular Turing machine when I am talking about if the respect to places are available, then it performers the desired operation. Otherwise depending on the input given to us it may hanger, it may not pursue and all those things, but for the entire Turing machine for a particular task if you are constructing Turing machine the input format and output format they are fixed.

Now, you see for example, instated of this S L when we are performing here I said I am starting with this with a blank here are whatever. Now, here up there are two blanks for example, in the beginning then the in the rest it will be you know one blank will be available, because the entire input is moved one cell to it is left where it is blank. So, that is how we have to consider here. So, to indicate, now you can ask me the question that for example, here if there is no blank that is not the hyposis for this Turing machine because their as to be a blank that indicate that, the input in what are the string to be shifted there is one cell available.

(Refer Slide Time: 25:49)



$S_L: \rightarrow L_{\#} \rightarrow R \xrightarrow{\sigma \neq \#} L \rightarrow P_{\sigma} \rightarrow R$
$/\#$
$L \rightarrow P_{\#}$
$\#abc\underline{\#} \mapsto \underline{\#}aba \mapsto \#\underline{a}ba \mapsto \underline{\#}aba$
$\mapsto \underline{a}abc \mapsto a\underline{a}ba \mapsto aa\underline{b}a \mapsto abba$
$\mapsto abb\underline{a} \mapsto abaa \mapsto abac\underline{\#} \mapsto aba\underline{\#}$

So, now, to perform this task let me constructed Turing machine and see. So, the input assume you are here and say a 1, a 2, a n is given a r is a blank, what do you do you just go to this end and you keep shifting one, one symbol to it is left this have you can do and to get the desired output. A 1 will be shifted first a, then in a 1 position you can move it like this a 2 can be moved and a 2 position you can move, a 3 in a three position you can move a 4 and so on that is the process here I pursue. This is L hash I will come first to the, this position and then take a right move.

Now, whatever that I am reading if it is not equal to blank, this is the notation very important you have to follow here. So, if what are the input that sigma I am reading if it is not blank, then I take a left move and then at the place I print this particular sigma. So, the notation here is what are the input I am reading that symbol to be printed here. So, that I am taking in the variable sigma, the sigma, which is not equal to blank, the same sigma will be printed here. And then take a right move and now further one more right move you have to take to continue in the loop. So, when you are receiving blank; that means, you have reached to the end of the input, then what do you do come one cell to it is left and their you print blank and halt this is how we can pursue you see.

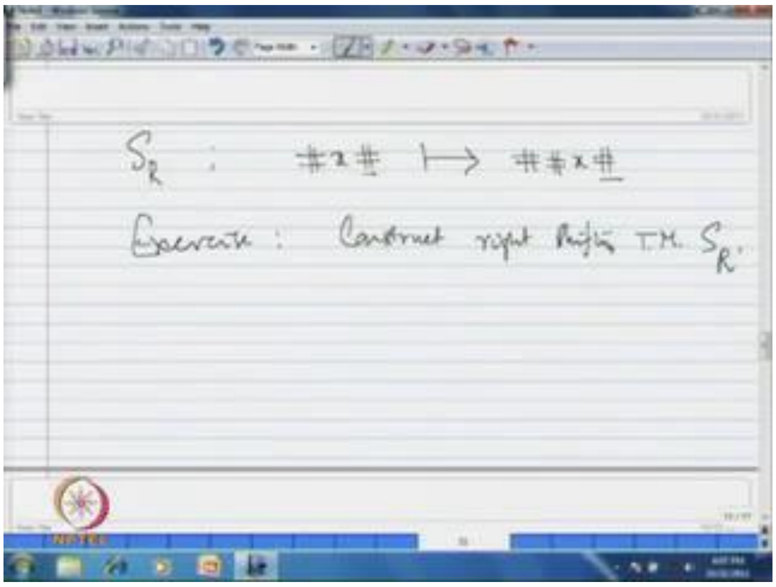
Once again what we are doing then the input, let me just take an example say a b a blank I am here of course, let me use this symbol, because here we require finitely many operations. So, this L hash in the beginning that takes the cursor to the first blank symbol to it is left; that means, it will come this place and of course, normally it halts L hash, but it has connected to R, it has connected to r unconditionally. So, from here by taking this step, it will come to this potion a b a, this is the situation R means.

Now, the current reading symbol is not blank that is a in fact, so it will take one left move so; that means, now with is l this condition has been cross checked and it is carrying the input in the variable sigma you take one left move; that means, it will come to this position a b a now this is situation. Now, there in the current position it will print what are the sigma that it is carrying it is carrying a. So, it will print a, so now, a comes here, so a b a this is the situation, now it is takes one right move with this machine.

So, that is now a b a, so one right move, now in this loop again I have connected back to this R so; that means, it take one more right move. So, it goes to this place a a b a this is the place with this are now this is not blank again. So, it will take one left move their it prints the current reading symbol that is carrying in sigma that is b. So, now it will be a b b, a at prints and takes one right move and further one more right move, so that comes here, so a b b a and again this is not equal to blank, this is not a blank symbol.

So, it will take a left move a b again, their it prints a this is a, so it takes one right move and further one more right move. That means, at the situation what it does a b a a it comes to this place blank. Now, the current reading symbol is blank, so what we it has to do it will take one left move their it prints blank and simply halt, so that means, this is a b a. So, this is how the input a b a, as been shifted one cell form the current position to it is left the entire input x has been shifted one cell that is what is the machine S L is doing.

(Refer Slide Time: 30:50)



S_R : $\# x \# \mapsto \#\# x \underline{\#}$

Exercise : Construct right shifting T.M. S_R.

Now, similarly one can think of right shifting machine, what is the process of this given input it will leave the input in this format one cell from the current cell. So, it as to shift this x one cell do it is right. So, that no format will be this, now take it an exercise to construct this Turing machine S R right shifting machine, construct right shifting Turing machine S R this is the notation let me use.

Now, let me go back and see when we have constructed to Turing machines ((Refer Time: 31:52)) for a power n b power n, the notation what we have used here some of the important points let us look at here. So, it takes one left move, when a b R blank these are the three possibilities we have numerated here, if the input alphabet as some other symbol here when the first left we have define for a b and blank for all the three symbols whatever in this particular context, we are writing we have written. If the input has some other symbol then the machine halts and the input will be accepted.

So, in this case we have to restrict because we have to specify that the input alphabet as to be a b sigma for which only this will work to accept a power n, b power n. Otherwise has a ad mention in the beginning if any other symbol is encounter when it takes left move it will halt and the input will be accepted as for all definition. Now, this is one important point and you see here we have define for a and a here it is unconditional. So, at every connection you see for every symbol, the thinks are define here and here it is assume that sigma to be a b blank this is the important hyposis.

Now, when I am coming to this Turing machine here some other conventions are being adapted that is we are carrying the input, because what are the symbols that we are reading that you shown on the arrow. And you see here for every symbol first arrow is given for every symbol, this is for non blank symbol second arrow, the third arrow nothing is return; that means, for every symbol and this arrow is for every symbol, this arrow is for every symbol, this is given for non blank and this is for blank symbol.

So, you see an arrow everything for every symbol the current reading symbol whatever it is it will work. So, that is you see this machine is independent of the sigma whatever that it is under concentration, only ring is some alphabet is fixed and on that alphabet this performance a job. You see this is a basic difference between this two constructions there we have to indicate that sigma as to be this accordingly it performance to accept a power n b power n.

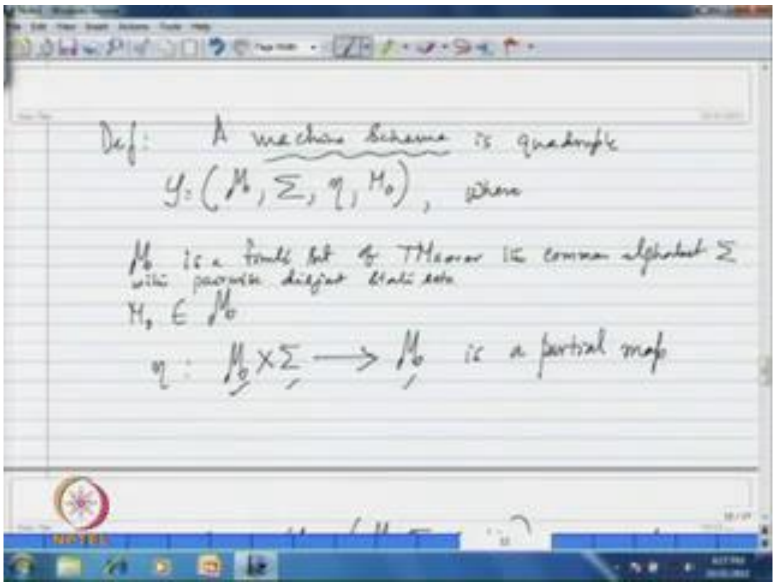
Now, assume in the previous case, if sigma is a, b, c there are three letters and now the you have the choice of giving input with some c is available. Now, in which case what will happen it will accept a power n b power n the strings, if you give a power n, b power n indivertible accept, but if encounters c on the tape on the input then since there is now definition then it will simply halt that is the problem. So, it will accept few more strings now you understand what are those strings that it can accept, if you make it if you make it independent the fixed alphabet.

That means, here when you are combining Turing machines as had mention you have a choice like to what you want to connect under what conditions you wanted to connect the condition will be specified on that arrow. As each you know you see we have started with when we are defining Turing machine we have a formal definition for this Turing machine we have given it is a quadruple their state set like other automata. We have state set alphabet and transition map and where we have to start.

Now, when I started discussing about this diagram and halt that now the question is whether is it loosing that generality; that means, any Turing machine as we have defined this Turing machine the way that I am now introducing this kind of notation whether they are actually as per the definition are not that you may have a doubt, I will address that yes.

This will actually follow the definition what we have define; that means, corresponding to this a such you can give the corresponding state set and right as a mention that this will this gives the advantage of you know not representing the states explicitly. Rather it will capture the entire idea of the basic definition of Turing machine through state transitions. Now, let me give the formal definition how to combine this Turing machines then you will realize yes exactly this diagrams are corresponding to the original definition. So, now for which let me first introduced the notion, because when you are combining Turing machines you require certain Turing machines to be combine that let me call it as machines schema.

(Refer Slide Time: 36:25)



Def: A $\underset{\sim}{\text{machine schema}}$ is quadruple

$\mathcal{S} = (\mathcal{M}, \Sigma, \eta, M_0)$, where

$\mathcal{M}$ is a finite set of TMs over the common alphabet Σ with pairwise disjoint state sets

$M_0 \in \mathcal{M}$

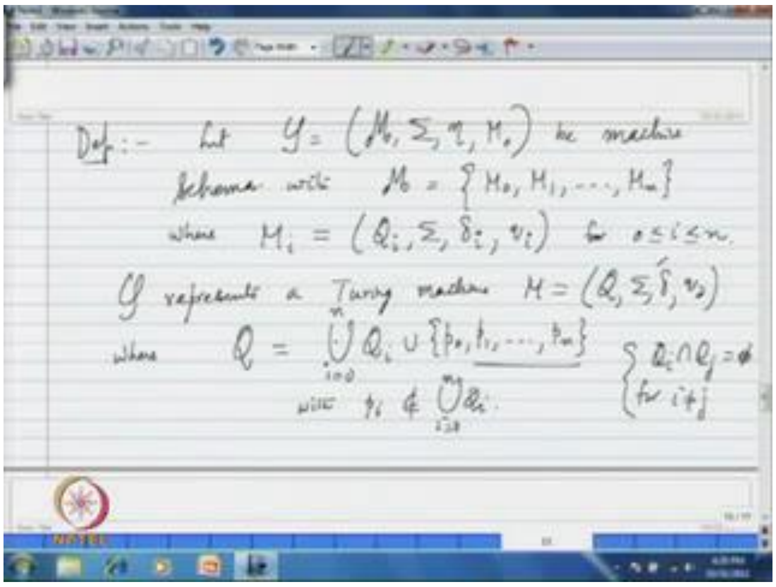
$\eta : \mathcal{M} \times \Sigma \longrightarrow \mathcal{M}$ is a partial map

That is a machine schema, the machine schema is a quadruple what do you give here, let me right by s a set of a finite set of Turing machines let me call it as m and or same alphabet. Let me say that is sigma and a partial map eta from where to where I mention letter this is m naught the initial Turing machine. Where m is a finite set of Turing machines is a finite set of Turing machines over the same alphabet over the common alphabet sigma and M naught is a specified machine in m this is the starting machine that we will.

So, to combine Turing machines we have to give this machines schema and eta this is a partial map that is from given a machine and symbol, which machine to be connected when it is halting a particular machine is a partial map. So, why this is partial map you have a choice that when a machine is halting, when a halting by reading a particular symbol, at the particular symbol if you have a definition as I had mention you will continue to connect to the next machine and it will continue the process.

If there is no definition of a particular symbol then the machine simply halts the composite machine simply halts. So, this machine schema is given as a quadruple, here this says a partial map given a machine and a symbol what is the next machine, if required. So, that is to be given to the partial map, this is machine schema and each machine each such machine schema represents, because depending on that eta, what is the combination will be declared and based on that we will see that, what is the simulation of that representing Turing machine.

(Refer Slide Time: 39:05)



Def:- Let $\mathcal{S} = (\mathcal{M}, \Sigma, \eta, M_0)$ be machine schema with $\mathcal{M} = \{M_0, M_1, \ldots, M_n\}$

where $M_i = (Q_i, \Sigma, \delta_i, q_i)$ for $0 \le i \le n$.

$\mathcal{S}$ represents a Turing machine $M = (Q, \Sigma, \delta, q_0)$

where $Q = \bigcup_{i=0}^{n} Q_i \cup \underline{\{p_0, p_1, \ldots, p_n\}}$ $\left\{ \begin{array}{l} Q_i \cap Q_j = \phi \\ \text{for } i \neq j \end{array} \right.$

with $p_i \notin \bigcup_{i=0}^{n} Q_i$.

So, we are let me give the definition also, a machine schema let this S as M sigma as I am writing eta M naught be machine schema, so we are given a machine schema. Now, S this machine schema represents a Turing machine, so here let me specify machines schema with this M because it is a finite set let me call them as M naught, M 1 and so on. Say m n there are n number of Turing machines are common alphabet where each M i let me fix that to be is the state set Q i as there is a common alphabet sigma, that is what is here and for this M I, let me take delta a to be the transition map and initial state of the Turing machine let me call it as Q i.

So, for 0 less than equal to i less than equal to n, so for all this Turing machines, let me take this. Now, this S the machine schema S represents, the Turing machine let me write that to be m with state set say Q of course, the same alphabet sigma the transition map delta. And the initial state will be the initial state of the initial machine there is q naught here is Q naught and where what is Q? Q is essentially all the state set. So, here very important thing is we have to see that in the machine schema this are, pair wise disjoint sets of states.

So, that is very important here in the machine schema let me put that condition here, this state sets the common of alphabet and pair wise disjoint sates either finite m is a finite set of Turing machines over common alphabets with pair wise disjoint state sets. So, this is important; that means, here if you take Q i intersection Q j this is empty for i different from j this is condition we have, so this is a disjoint union essentially here, so this are all disjoint sets. And to combine this let pick up some new states, they are new states which are not there in any of these capital Q s.

So, this are the new sets with the condition that the P i is not in union Q i equal to 0 to n, so that is how you have to choose this state set. So, essentially when I am talking about the Turing machine that is represents by the machines schema, this has to capture whatever. So, for we have constructed, because the original Turing machines will be lying in that circuit what are the circuit that we are drawing, the diagram that you are drawing.

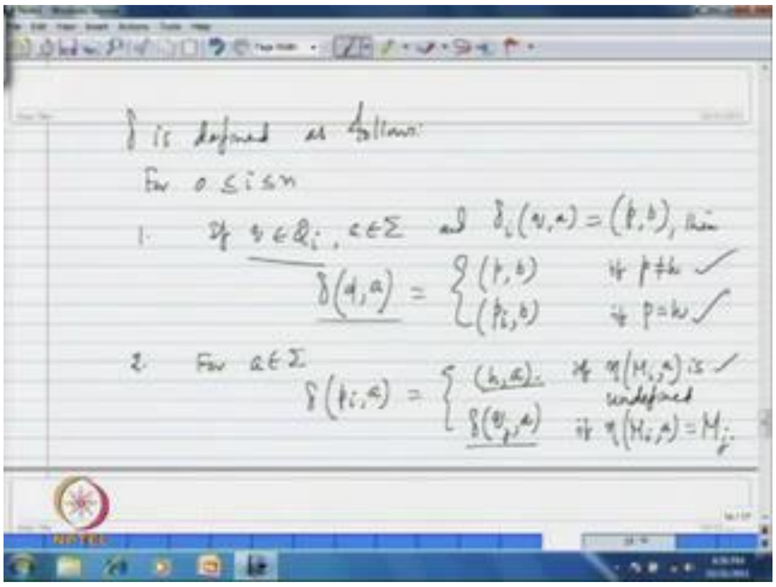
So, those states we also be will be there and this new states P naught, P 1, P n one for each Turing machine that I consider for what purpose, whenever a Turing machine is halting a particularly. For example, if M i is halting as we know that we should not make the machine the composite machine halt, we have to crossing the condition. And the current symbol if there is a definition to connect to some other Turing machine, we have to connect to that, so if we halts then the machine halts, the entire machine will halt.

So, what we have to do a component machine a particular machine whenever it is halting as per its definition delta i here, whenever it is coming to the halting state h, we change the transition to you know by keeping to on auxiliary state for each M i. I take one p i a new state instated of putting in the halting state in put first in the state p I, further purpose we consider this for each machine one auxiliary state P i.

So, that is how these states are consider this states are consider, now you put it there and you cross check eta, whether eta has a definition for that particular symbol at that particular movement. If there is a definition; that means, what is the next machine will be known from eta is there is a definition, then you connected to the particular Turing machine and start simulating the rest on that machine; otherwise then you can halt the machine.

So, that is how essentially the delta works where, now let me give that also. So, this is the state set and now delta is defined as follows, because in this four components M the, Q is mention sigma is a common alphabet and delta I have to define Q naught is initial state of the initial machine, that is way the process will start, now delta is define as follows.

(Refer Slide Time: 44:57)



δ is defined as follows:

For $0 \leq i \leq n$

1. If $q \in Q_i$, $a \in \Sigma$ and $\delta_i(q, a) = (p, b)$, then

$$\underline{\delta(q, a)} = \begin{cases} (p, b) & \text{if } p \neq h \checkmark \\ (p_i, b) & \text{if } p = h \checkmark \end{cases}$$

2. For $a \in \Sigma$

$$\delta(p_i, a) = \begin{cases} \underline{(h, a)}. & \text{if } \eta(M_i, a) \text{ is undefined} \checkmark \\ \underline{\delta(q_j, a)} & \text{if } \eta(M_i, a) = M_j. \end{cases}$$

Now, delta is define as follows, now for this condition if the current state q is in the component Q i, what are the symbol? It is reading if this and the definition of delta i at the symbol q a is p b whatever it is, then this delta at the state q a, it will consider p b provided p is not halting state. If p is halting state it will not halt rather it will put it in the state corresponding to these machine that p i and whatever it is performing the same thing it will perform p i b, if p is halting state that is how this performs.

And now for states which are not in any of this Q i's we have to define because here we have define the first statement is for any state which is in q i. And now their only two possibilities if you pickup any state rather it will be in the union of any of these states or any of the these P i's. Now, the situation is if the state is from p i then; that means, for just a in sigma, now I have to give the definition for this P i's only that is P i at a is define to b this halting a whatever that a, if eta of M i at a is undefined if this is not define.

If there is no next state from this state at the symbol, then the machine should halt otherwise whatever the next machine if it is define so on. So, if I say q j what are the next machine it is doing. So, you can do this you can connect it to this the first state initial state of the next machine is Q j if it is M j. So, if eta of m i that the symbol is M j.

Look at this transition whether it as captured carefully, there are two possibilities for a state and for each symbol sigma that I have to define because the transition map has to be define for each state and the symbol. So, here the states either it is any of this machines are the new states that I have introduced P naught to P n. Now, the definition here is if the state whatever that I have consider if I have to define delta of q a if the things as to be define, you just look at.

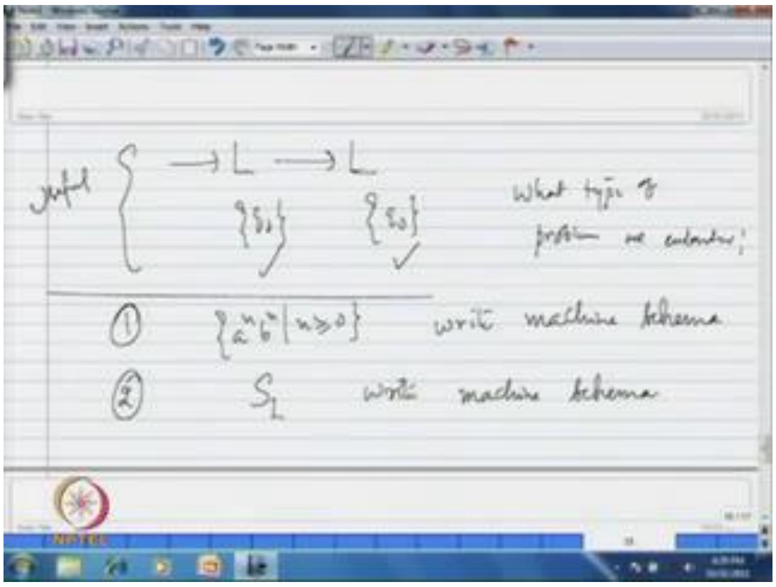
If this q is an any of these Q i's then you simply define as in that particular machine; in that particular machine if delta of q delta of q a is p b then you follow the same provided if that result an state is not a halting state, that is what is the condition here. If the result an state is a halting state as I had mention you do not make it halt, rather you put in that auxiliary state P i corresponding to this M i the machine because when the Q is an Q i; that means, I am simulating the machine M i.

In which case you put corresponding halting state that is what is p I, so this is the condition here, if it is halting state put it in P i. Now, the definition for when you i have somehow come to the any of this P i's; now for each symbol a in sigma how to define it now you just look at, because if I am in P i; that means, this auxiliary state corresponding to the state M i. So, you just see M i at that particular symbol, whether this definition is given that eta is define or not, if it is defined then we have to at one way if it is not define you have to act another way.

What is the, if it is un define then you simply halt; that means, halt in the same position; that means, since I am reading a, I will of course, I have to something. So, I will print the same symbol that is how the definition is given that is h a, if there is a definition eta as a definition for this pair assume it is asked to connect to M j there in which case what we have to do is these current symbol. Whatever the machine m j does the initial state initial state of M j is Q j, so delta of q j a that is how it is define.

So, this is how the total map delta is define for the Turing machine m and the machine schema given a machines schema; that means, a finite set of states and declared as with some initial machine are common alphabet with disjoint set of states, if when eta is given, that eta tells you that how the machines are to be combined. And as per this you this machines schema that represents a Turing machine as define here, so this is the total definition of that. Now, what we can do you just take some machines schema are some combination and you can what are the component Turing machines the respective states you can write and you see. Now, the question is if you are having you know some Turing machines where you have to use them twice or thrice.

(Refer Slide Time: 51:03)



$\rightarrow L \rightarrow L$

$\{q_0\}$ $\{q_0\}$ ✓

What type of problem we encounter?

① $\{a^n b^n \mid n \geq 0\}$ write machine schema

② S_L write machine schema

For example, if you consider the Turing machine L L, now if you consider is say for example, this you have consider the state q naught, because we know the Turing machine will have only one state; here also if I consider q naught then that creates a problem that you have to realize. So, realize the problem if you consider same q naught here, what is the problem as per the result thing. So, if I consider here also q naught the state set and in this component also q naught as I had mention when I am taking two machines you have to have disjoint set of states.

So, by considering same state set what type of problem that you will encounter, so that you just look at and will discuss these things in the next lecture. Now, for some you can write the machine schema and what is the machine schema for the an appropriate combination. For example, so first understand that an accordingly whatever the machines that we have already constructed, one is for a power n, b power n such that n greater than or equal to 0, you have constructed Turing machine for which you write the machine schema. So, this information will be useful to you, this a useful information for this purpose.

So, exercise one is these and write machines schema for whatever the left shifting machine we have constructed, write machines schema, you have seen constructed composition of Turing machines. Now, first you realize what type of problem that you encounter if you take the state set, so what is the difference why we have to consider different this thing. So, that how you have to represent those things that you should discuss in this particular problem that will be useful to answer the this two what are the machines composite machines we have constructed, write the machine schema for that and will discuss the rest in the next lecture.