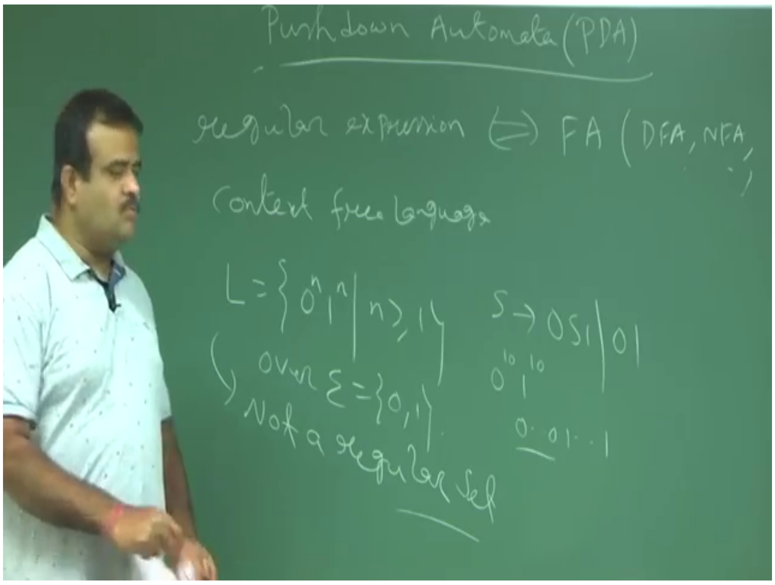
Introduction to Automata, Languages and Computation

Prof. Sourav Mukhopadhyay
Department of Mathematics
Indian Institute of Technology, Kharagpur

Lecture – 48
Pushdown Automata (PDA)

So we have to; we will introduce the new finite machine which is called Pushdown Automata, which is to accept the context free language we need the pushdown automata. We have seen to accept the regular language, we need the finite automata like DFA, NFA and this is the pushdown automata.

(Refer Slide Time: 00:40)



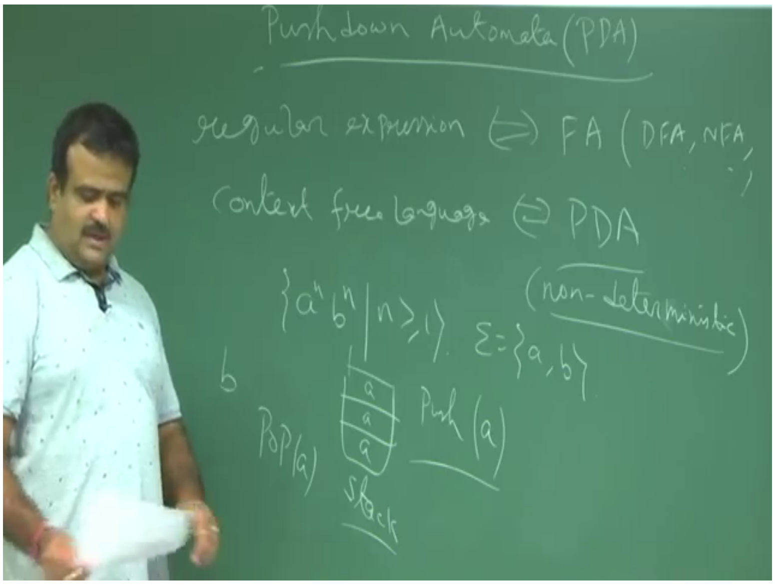
So, for regular expression or regular state, we have corresponding finite automata, for example, DFA, NFA, epsilon NFA, all this finite automata is there. Now, we have seen for context free language languages like for example, if you take this language L 0 to the power n 1 to the power n over 0 1. So, we know this is not a regular language, because there is no finite automata which can accept this, not a regular language.

We have not having any regular expression which is corresponding to this. Even we have proved this using the pumping lemma that this language cannot be regular ok, because in the pumping lemma that is the necessary condition of to be a language to be regular. But we have seen this language is a context free language, that means we have a grammar for this language context free grammar that grammar we know, this is the grammar like s is going to 0 s 1 or 0 1 like this ok. So, this can generate this.

So, now, the question is why it cannot accept by a finite automata, because the if for acceptance of these we need to count the how many ones we have because it has to say for example, it has to accept 0 to the power 10 1 to the power 10 like 10 0 followed by 10 1s, any form of this 0 to the power 100, 1 to the power 100. Now, in the finite automata there is no way we can count how many 0s we have encountered so far, so that we will encounter the same number of 1s in the when we read the input as 1. So, that is not possible here.

But that is for that we will introduce a concept called pushdown automata, where we will use a stack we will use a stack to I mean sort of memory we can see I mean how many 1s, how many 0. So, whenever we have a 0, we can push 0 into the stack. Whenever we see a 1, we push we pop 1 into this. So, there is will be a stack where the push and pop operation will be there, so that is the new automata which will having extra stack to in the definition like this is.

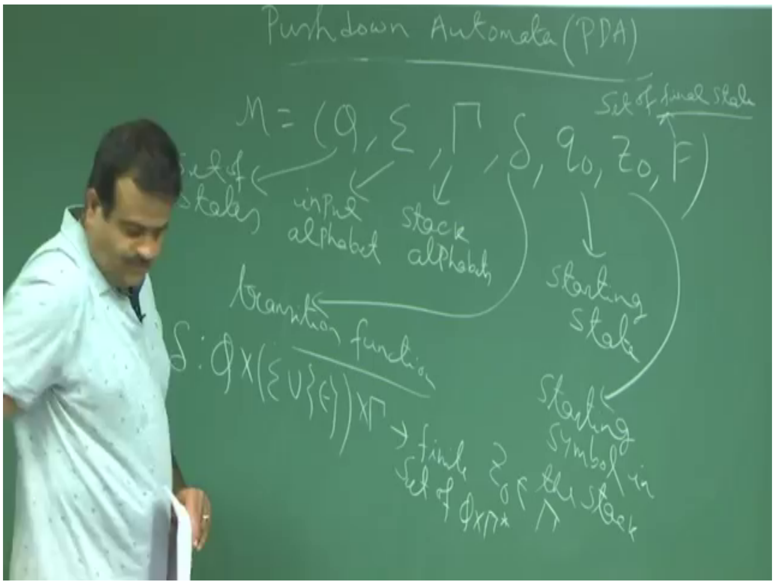
(Refer Slide Time: 03:51)


Push down Automata (PDA)
regular expression ⟺ FA (DFA, NFA, ...)
Context free language ⟺ PDA
(non-deterministic)
$\{a^n b^n \mid n \geq 1\}$. $\Sigma = \{a, b\}$
b
a
a
a
Push (a)
Pop (a)
Stack

So, this will be corresponding to PDA or push down automata ok. And this is the first we learn the non-deterministic version of it, then we have a deterministic version of it. So, the concept is in order to accept this language, what we need in a finite machine, we need to have like we need to remember how many 1s, how many 0s, you have how many here it is a, b, how many a's we have encountered.

So, this is over a, b, so that means, whenever we see a, a we push into the stack, we have a stack we push into the stack ok, push stack kind of thing, this is stack. And whenever we see a b, we pop a from the stack, so that means, if ultimately if the stack is empty then we can accept the string, so that is the idea. So, let us formally define what do you mean by a pushdown automata ok.

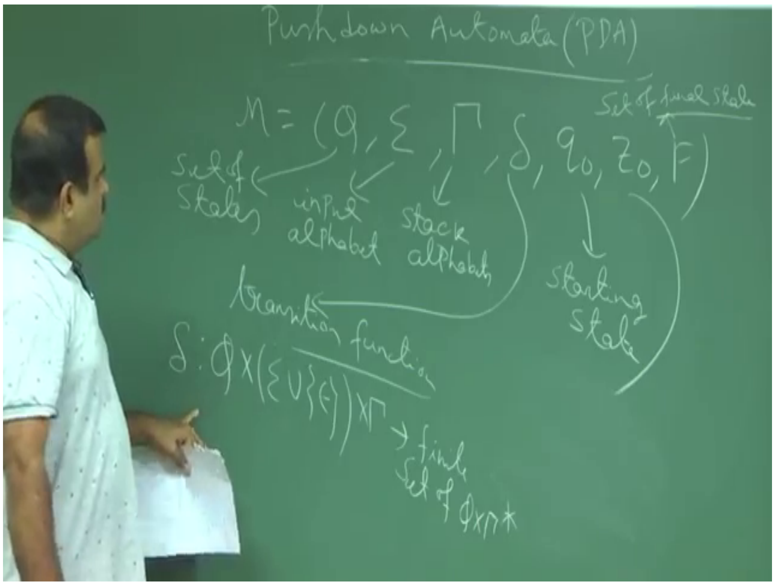
(Refer Slide Time: 05:22)


Pushdown Automata (PDA)
$M = (Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$
Set of states
input alphabet
stack alphabet
Set of final state
starting state
transition function
Starting symbol in the stack
$\delta : Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma \rightarrow$ finite set of $Q \times \Gamma^*$
$z_0 \in \Gamma$

So, it is a tuple consist of Q, sigma, delta, q 0, z 0, F ok. So, this is the this we know this is the set states, set of states, this is finite. And this is the input alphabet which is on the tape and this is the stack alphabet which will be there in the stack, all are finite this is the stack alphabet and this is the transition rules. So, here we have a our machine is reading a symbol from the stack and that symbol is coming from this set ok. And this is the transition rule I will define this formally this is the initial state or the starting state of the machine, and this is the starting symbol in the stack. So, we put this z 0 in the stack, starting symbol in the stack symbol in the stack.

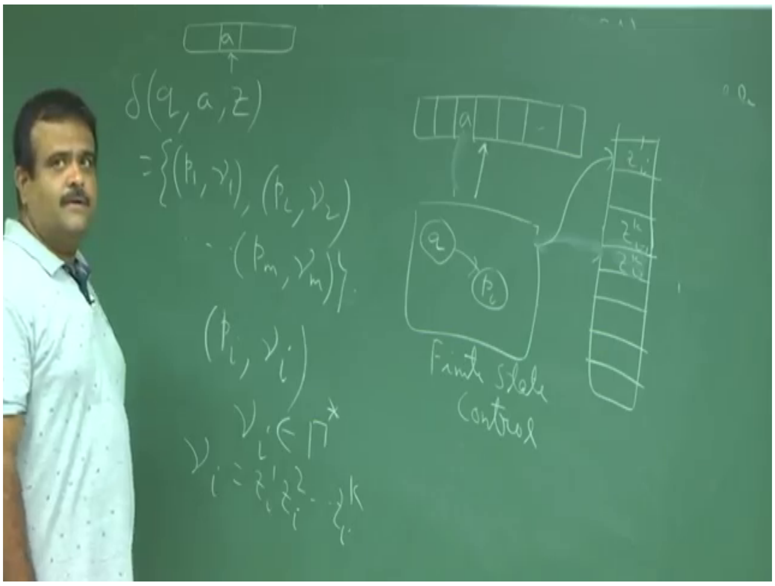
So, z 0 is basically coming from this set starting symbol z 0 is the stack alphabet, and this is the final state, this is the set of final state, which is a subset of Q. And this delta this is the transition rule or transition function, so this is now is a sub is coming, so it is just considering a state where we are at now and it is reading a input symbol in the tape and also it is reading a it is seeing a stack ok.

(Refer Slide Time: 08:38)



So, this is it is reading the stack symbol. So, this is a function of Q our current state and sigma the systems this is the epsilon NFA sorry this is this is a non-deterministic and an epsilon move is allowed here and we have a stack. And it is going to finite set of it is going to finite set of set of Q cross star.

(Refer Slide Time: 08:48)



So, basically it is taking a so we are at say q at some state and our tape is say reading at that is say a, we are reading the input symbol as a, a could be epsilon also. And then suppose here our stack symbol is say z, this will be a some kind of subset of this set Q cross this. So, it will read a symbol from the stack, suppose this is the stack symbol and it will replace a string of that stack symbols in the stack.

So, suppose it is say it could be any set of p 1, gamma 1 like this, p 1, p 2, gamma 2 dot dot dot say p m gamma m, this is non-deterministic. If it is if it is deterministic, then we have only one of this. Since it is non-deterministic, we have a choice. So, it can go to say any, so it can be any one of this. So, it is a subset of this is option we can move any one of this. So, that means, if it is say so you have a option to go anyhow.

So, suppose, we this is the situation, this is our tape, we are reading this tape. At some point of time, we had our tape head. And this is our machine. And we are such as this is our finite machine finite state control. And suppose we are at state q and we are reading this symbol and there is a stack position also there this is a stack. And suppose this is the top of the stack which is currently we are reading.

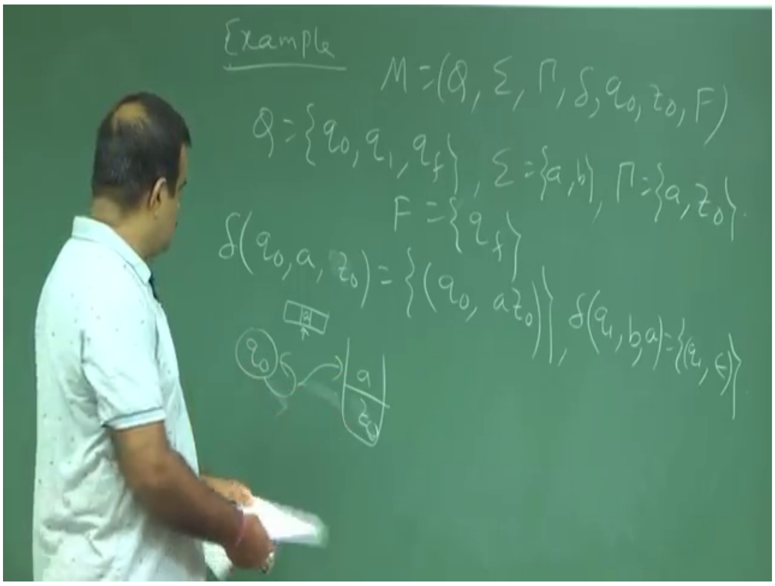
So, currently our position is this is a snapshot of the machine, we are taking a snapshot of the machine, this is the current situation of the machine a final state control. So, that means, we are currently we are at state q and our tape header input is reading the symbol a, and our stack symbol it is seeing z. Now, if it is it have a option, it can take any one of this suppose it is taking the option p i gamma i.

So, this is a, this is coming from this is a symbol stack symbol. So, this could be say some z i 1, z i 2 like this, z i k. See if it is like this, and what it is so that means, it is going to, so it will go into some state p i and it will move further and then this will be replaced by this. So, z 1, z 2, z i k z i minus 1 k, so z i 1 like this and this in the pointing to like this ok. So, it can take one of this option.

So, this that since the same as it is called non-deterministic, we have a non-deterministic this is not deterministic. If it is deterministic, then it will be it will be fixed one of this. So, it will go to some p state and it will replace that z to some of the string of the stack symbols that is the way ok. So, this is the definition of a pushdown automata. So, now that z will be replaced by these symbols like this.

So, we take a example of a pushdown automata then it will more clear. So, the difference from a finite automata which we have seen the DFA, NFA their difference is that we have a stack here I mean we are reading another symbol which is a stack symbol, and the movement the delta transition will depend on the current state, the input alphabet and the stack symbol. So, it will go to some new state or may be the same state and it will replace the stack by some string of the stack symbol ok. It could be epsilon also that is why it is string, string of the stack signal. We will take an example.

(Refer Slide Time: 14:04)

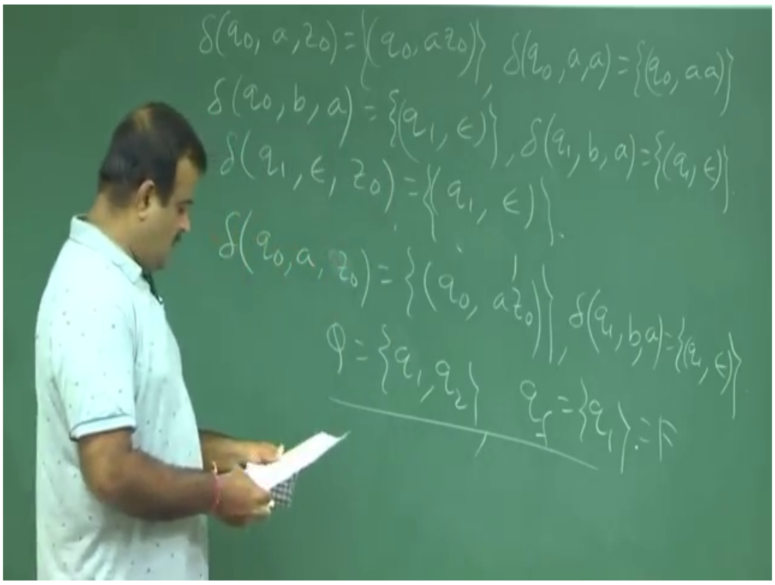

Example
M = (Q, Σ, Γ, δ, q0, z0, F)
Q = {q0, q1, qf}, Σ = {a, b}, Γ = {a, z0}.
F = {qf}
δ(q0, a, z0) = {(q0, az0)}, δ(q1, b, a) = {(q1, ε)}
q0
a
z0

We will take an example. So, let us define a. So, this is a tuple, you know this is a pushdown automata z q 0 z 0 F. So, where Q is we have how many states we have say three state q 0 q 1 q f final state, we have one final state and suppose sigma is a, b and stack alphabet is say a z, 0, z 0 is the initial position of the stack that is the starting symbol in the stack. And F is q f, we have all the one final state.

So, delta we have to define. So, delta we write like this q 0, a z 0 sorry. So, if we see z 0 in the stack. So, this has only one option q 0. So, it will be remain that q 0 and the stack sigma will be it is z 0, so it will be replaced by a z 0. So, we are pushing a kind of thing then delta of q 1 b a; that means a sorry q 1, so it has three input q 1, if you see a b and if we see a in the top of the stack, so it will just go to some q 1 and it will erase that a from the stack. So, epsilon will be replaced so, it is ok.

So, this means what, this means we are at q 0 and our stack is say z 0. If our input tape we are reading a layer, then it will going to it will be remain at q 0 and this will be replaced by this will be replaced by a z 0 like this and this is pointing here. And now again if we see a b sorry if we see a, so this is the things now if we see a b now then it will just go to q 1 and it will replace this. So, we are trying to accept a to the power n, b to the power n. So, let us write this.

(Refer Slide Time: 17:00)



$\delta(q_0, a, z_0) = \{(q_0, az_0)\}, \delta(q_0, a, a) = \{(q_0, aa)\}$

$\delta(q_0, b, a) = \{(q_1, \epsilon)\}, \delta(q_1, b, a) = \{(q_1, \epsilon)\}$

$\delta(q_1, \epsilon, z_0) = \{(q_1, \epsilon)\}.$

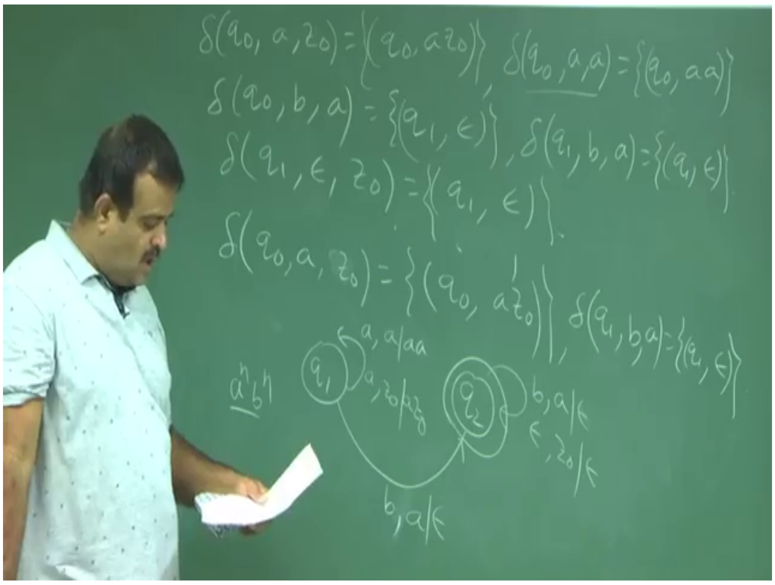
$\delta(q_0, a, z_0) = \{(q_0, az_0)\}, \delta(q_1, b, a) = \{(q_1, \epsilon)\}.$

$Q = \{q_1, q_2\} \quad q_f = \{q_1\} = F$

So, let us write the rules here transition over here. So, we have transition like this delta of q 0, a, z 0 equal to q 0, a z 0, delta of q 0, a, a, we replace this by q 0, a a. So, we are pushing a basically. Now, delta of q 0, b, a, then we have to it is q 1 epsilon. We are erased this a. We are popping a, pushing a popping a, now delta of q 1. So, q 1, b, a, then we will go to the intermediate state before going to the final step q 1 and then delta of q 1, epsilon z 0, this will be q 1 epsilon or we can make it; we can make it this one.

So, this is the sorry we have only 2 state; one is the q f. So, this, sorry so Q is q 1 q 2 only; and q f is basically q 1, yeah it has only 2 states. So, like this ok. So, this is please rectify this. So, this is Q is q 1, q 2 and among this q 1 is the final state that is the F. So, this is our F ok. So, let me write this diagraph for this. So, if you write the diagraph for this.

(Refer Slide Time: 19:17)

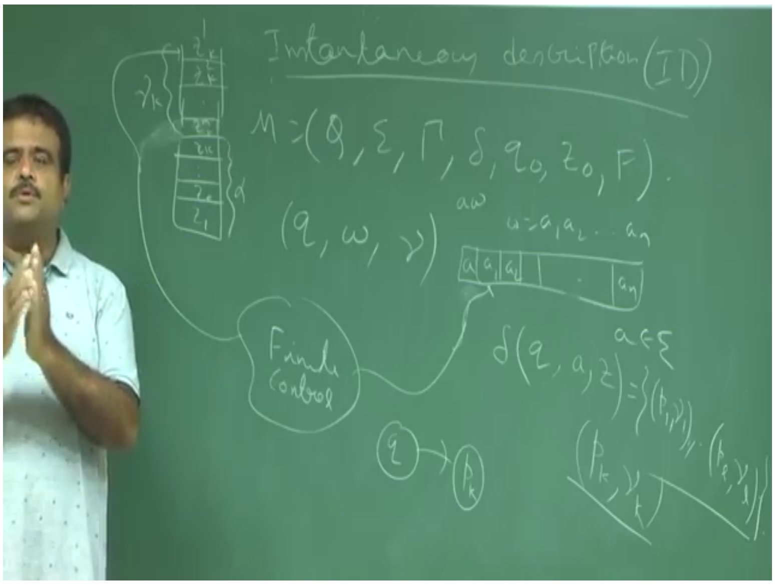

δ(q0, a, z0) = {(q0, az0)}, δ(q0, a, a) = {(q0, aa)}
δ(q0, b, a) = {(q1, ε)}, δ(q1, b, a) = {(q1, ε)}
δ(q1, ε, z0) = {(q1, ε)}.
δ(q0, a, z0) = {(q0, az0)}, δ(q1, b, a) = {(q1, ε)}
aⁿbⁿ
q1
q2
a, a/aa
a, z0/az0
b, a/ε
ε, z0/ε
b, a/ε

So, we have 2 state q 1, q 2. Now, from q 1, if we see a from q 1, if we see a a and if the stack symbol is say a, if the stack symbol is z 0, then it will go to q 1 and the stack will be a z 0. This is the stack symbol string which will be replaced by which will replace this. And if it is b or if it is again we have three option, if it is a, again if it is a and then if it is a again this one, so it will replaced by a a like this ok. And if it is like a z 0, we have this, this one this. And if it is b, then it will go to here. If it is b and the stack symbol is a, then it will replace the stack symbol by epsilon.

And from b what we are doing from sorry from q 1, if it is b and if the stack is a, then it will go to epsilon we are erasing that. Now, if it is epsilon and the stack is z 0, then it is epsilon ok. And this is the final state. So, it is reaching to the final state with the any string like a to the power n, b to the power n. We will formally define the language of a PDA the pushdown automata, but this is one example.

So, our stack is having the symbols like a or z 0. Now, when we read it a we are we push a into the stack; and when you read a b after that we pop a like this, so that we accept this ok. So, now we can have a we can define the instantaneous and description of a this stack of a PDA. Then we will define the language accepted by the PDA.

(Refer Slide Time: 22:01)


Instantaneous description (ID)
M = (Q, Σ, Γ, δ, q0, z0, F).
(q, ω, γ)
aω
ω = a1 a2 ... an
a ∈ Σ
δ(q, a, z) = {(p1, γ1), ... (pk, γk)}
Finite Control
q
pk

Instantaneous description this is called ID of a PDA. This is just a nothing, but a snapshot of the current position of the machine, this is a finite machine. So, snapshot I mean what is the stack would be like this. So, if given a M the pushdown automata z 0, F. So, ID is a tuple like this q, w and this, so q is the current state. So, like this we are at suppose this is our w, we are reading this w is a 1, a 2, a n, this is our input alphabet a n. And this is our we are reading a, suppose before that we have a a. So, a a our current position of the stack is like this. So, we are reading basically input alphabet a w, a is an a is coming from sigma ok.

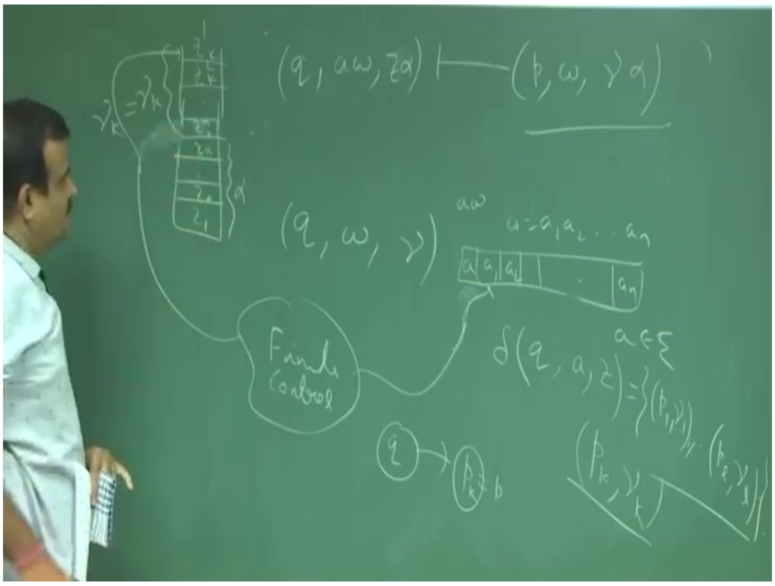
Now, suppose the position of the stack is like this. So, we have this is alpha is there up to this we have say z 1, z 2 z k, this is say alpha and we have at the top of this we have a z, which is currently we are seeing. So, it is a finite control, this is our and we are reading this stack symbol, top of the stack. And we are reading this input symbol a and we have remaining the input s in a string this.

And suppose the delta is like this. So, delta is currently suppose we are at q and we are reading a and our stack symbol is z say. So, this will be; this will be subset of some p i’s delta i by gamma i like this p l gamma l. So, you have option this is a non-determine we have option here. Now, suppose you are taking the move p k gamma k.

So, gamma k is the so p k gamma k, so that means from here q, it will go to p k and this z will be replaced by the gamma k. So, gamma k may be consists of say some z k 1 I mean something like that z k 1, z k 2, z k k m. So, this is our; this is our say gamma k, and it will next it will pointing with the top of the stack. So, it will if it is epsilon if it is replacing epsilon, then it will erase that symbol like this.

And if we are taking this move and the next will be this one this is the position of the remaining position of the tape input we have to read, so that is the tape position how much yet we have to read. We have given a sequence of input, we are keep on reading after a, we have to read this. So, this is the current position of the tape we have to yet to go. And this is the current state position so that means, this will write as this.

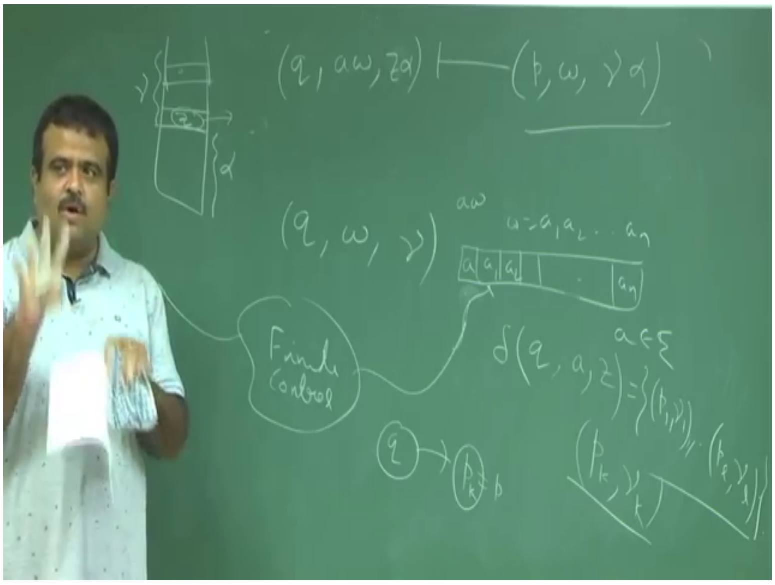
(Refer Slide Time: 26:33)


$(q, a\omega, z\alpha) \vdash (p, \omega, \gamma\alpha)$
$\gamma_k = \gamma_k$
z_k
z_1
α
(q, ω, γ)
$a\omega$
$\omega = a_1 a_2 \ldots a_n$
a a_1 a_2 a_n
Finite Control
$\delta(q, a, z) = \{(p_1, \gamma_1), \ldots (p_k, \gamma_k) \ldots (p_l, \gamma_l)\}$
$a \in \Sigma$
q
p_k
p

So, q, we are at q and w a w is the current input and then gamma. So, gamma is the current state of the this thing. So, gamma is nothing but say sorry z alpha. So, alpha is there, z was there. So, this is going to, so now, the current position is this is a snapshot ID means the snapshot. So, currently current it is going to say p, I mean this is say p. So, p is the current state and what is the current. So, we have already read a, now we have to read w, w is this string.

So, you have to about to read this, this is the position of the tape. And then we have the what is the position of this stack, we have added this. So, position of the status is instead of z, we have added this, so that is basically say some gamma and alpha, some gamma alpha ok. This gamma is nothing but this gamma k. So, we are replacing this. So, stack we are reading in the stack, we are reading the top of the stack.

(Refer Slide Time: 27:54)



This is our z and we have remaining the symbol is alpha. So, these will be replaced by some gamma and then we have a tape here. So, that tape again you have to read, but this is just going by one move. We could have different choice also, but if you take that choice, this is going to one move. So, this is the move relation this is going by one move. Again from here it will go like this.

So, we will keep on going like this. So, but this is the instantaneous description of a push down automata that means we will take a snapshot of that at some at that point what is the situation of the machine like what is the remaining tape it has to read, what is the current state, what is the stack position, so that is the ID. So, we will talk about more on this, we will define the languages accepted by pushdown automata in the next class.

Thank you very much.