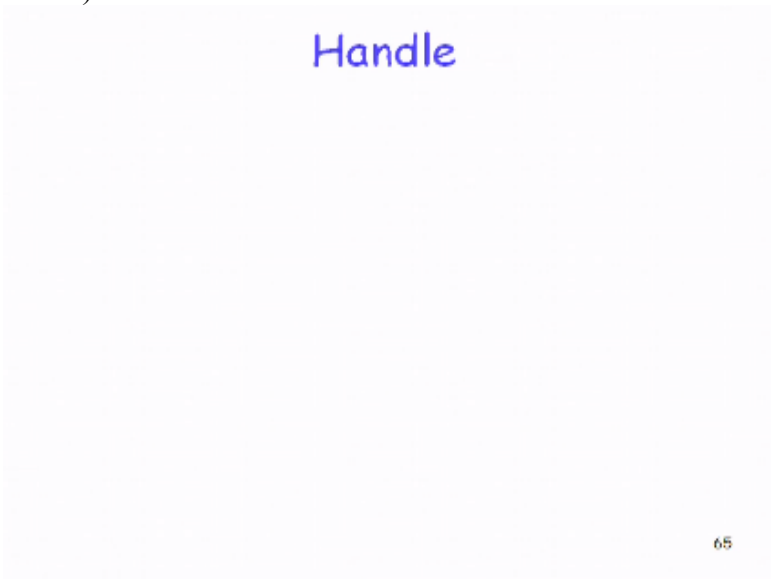
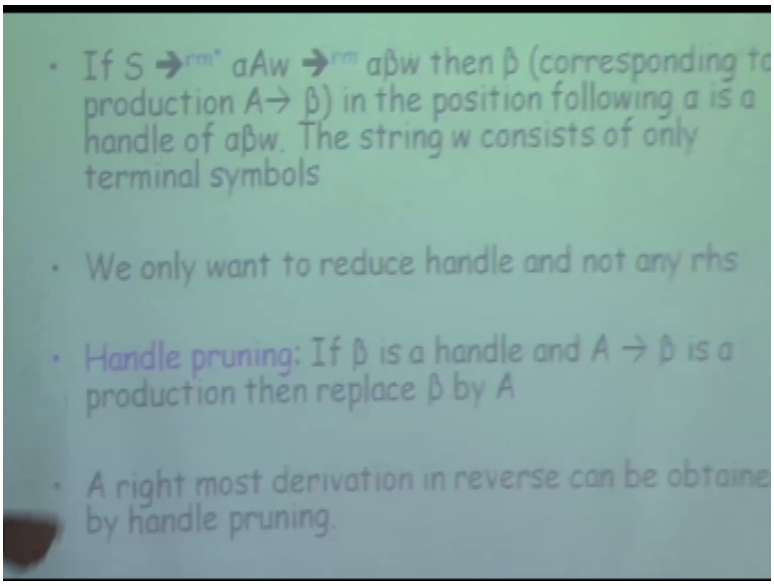
Indian institute of Technology
Kanpur
NP – TEL
National Programme
On
Technology Enhanced Learning
Course Title
Compiler Design
Lecture – 10

By…
Prof. S. K. Aggarwal.
Dept. of Computer Science and Engineering.

(Refer Slide Time: 00:15)



Good morning so for PPT recapture here what were here discuss towards end of the last class and then we will continue so first axial words construction of part of the looking handle and handle is try to production if I find to the pattern if I also this we symantaniousely aspects.
In the reverse cyclone derivation and I continue hope we have to the start symbol and if the string is write then I will start with the string is incorrect then I will be asked so notation wise we are saying that star symbol in some steps of right most derivation.
(Refer Slide Time: 01:15)



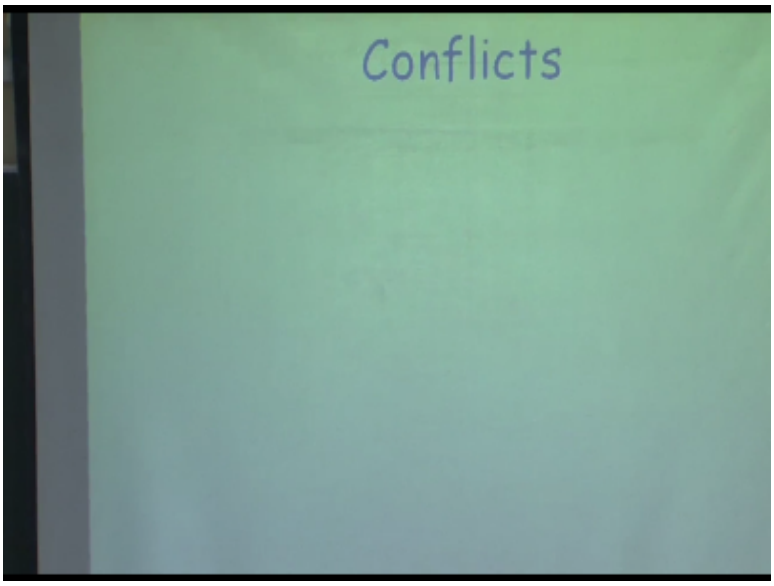
Gives me a sentential form with the alpha W the W is string of terminals alpha is a string of terminals alpha do is string of terminals and non-terminals and a is non terminal and then one more rightmost derivation gives me a pattern which is alpha beta W which means that there is a production which say it goes to beta which is part of the grammar and then if I find such a pattern and beta is the handle.

Then I can use beta way and this is what is giving here static is the reverse of high potential so a means that starting from W I should keep on finding such handles and even to the effects to reach the particle so you only want to reduce handles and not any right hand side and this is very important that so the first topic we have to discuses in the preview part are the important are in the classy where are the just matching.

The pattern without worrying about whether it was reversible step that where to a situation where I should not move I could not go any further and therefore came to a wrong conclusion back the string did not belong to the language the language so pencil tuning is so there the first thing we do and this is really where we are moving towards handle and when we are saying that handle pruning it is a step.

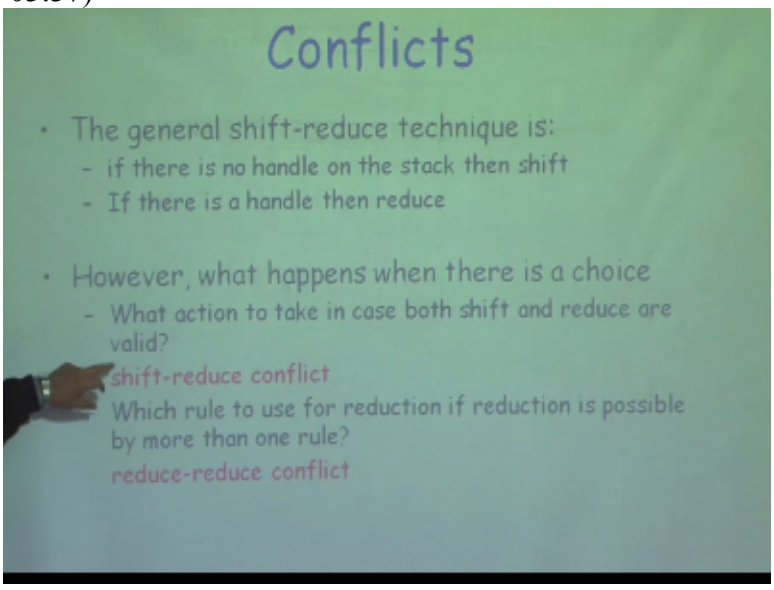
Where if I find the handle and I use this rule to reduce handle to this left hand side and then eventually parsing is nothing but it will take me to the start symbol and derivation because on replace just in terminals so this is what you were discusses towards end of the previously right .

(Refer Slide Time: 02:52)



So let us look at now some of the property because being kind of formals and we phase in the partial and therefore aware of this so one thing that happens is that there are two things which are happening initially to possible they are two possible actions one is you introduced to other actions I will not worry about at this point of time which is saying that we can say must have been have a recovery and the potential that have we should accept this derivers and say that belongs to handle so either I can shift something on the stack or I can do a reduction and in cases where I have more than one choice so suppose I have a situation and again both shift and I can do reduce okay.

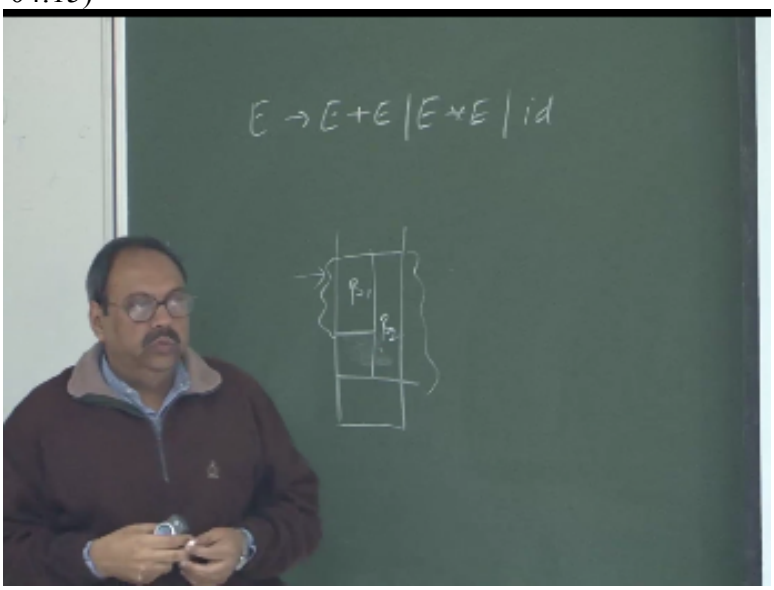
(Refer Slide Time: 03:37)



And again just call if I write this grammar when I was getting in to the situation I a head both shifted as possibilities and then you have to make a choice like what to do known as shift

responding and normally have a make the line grammars so they have get into conflict like this and similarly at we have getting into a reduce to reduce conflict okay so that what that means is that if I have a step and this is top of a step either I can find a beta1 once which matches some right-hand side or I can find a beta2 okay.

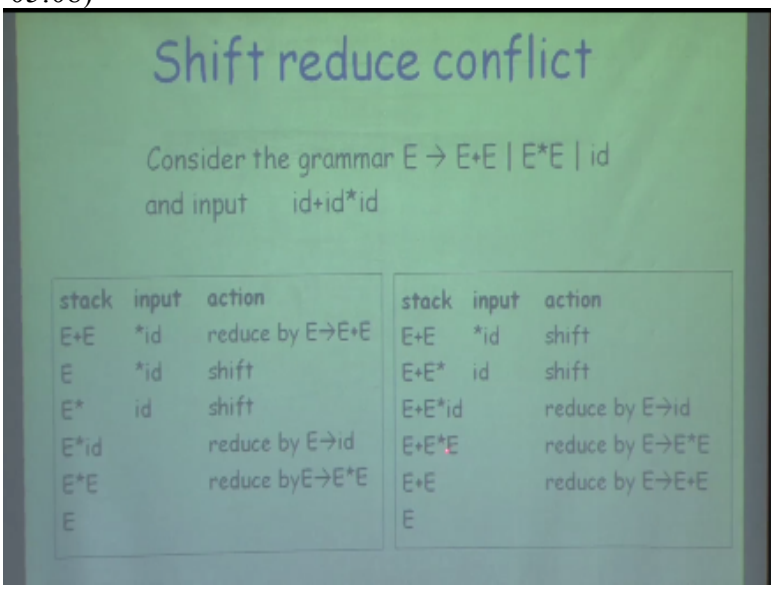
(Refer Slide Time: 04:15)



So this is right properly so this may be the γ and this like be the beta2 and both match like inside of a production and therefore I can get to work on fixing which reduction.

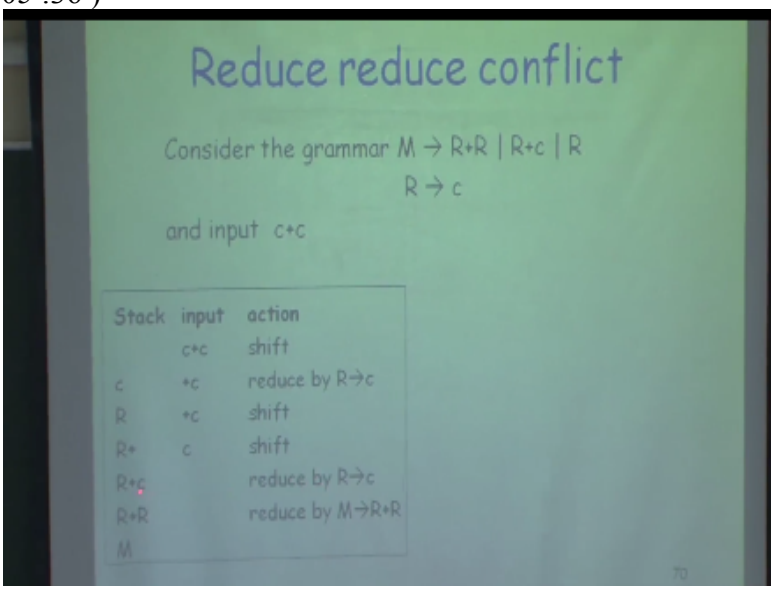
So normally we would like to design our grammar so that I never content and compromise the kind of form so these conflict are leading to a situation so there are the interpretation so conflict can come because of two reasons either my grammar is not is improperly or the language is such that it does not fall into this class of languages whatever.

(Refer Slide Time: 05:08)



So that is so let me give you an example of both shift reduce and reduce to reduce conflicts so here is the grammar we discussed okay and we have already seen this example that at this point of time I was able to reduce but also I was able to I could have started slightly differently and I could have first shifted this and could have left to this reduction so either I can reach start symbol by two different set of actions and this is what the form conflicts similarly so this example we discussed expensively yesterday.

(Refer Slide Time:05 :36)



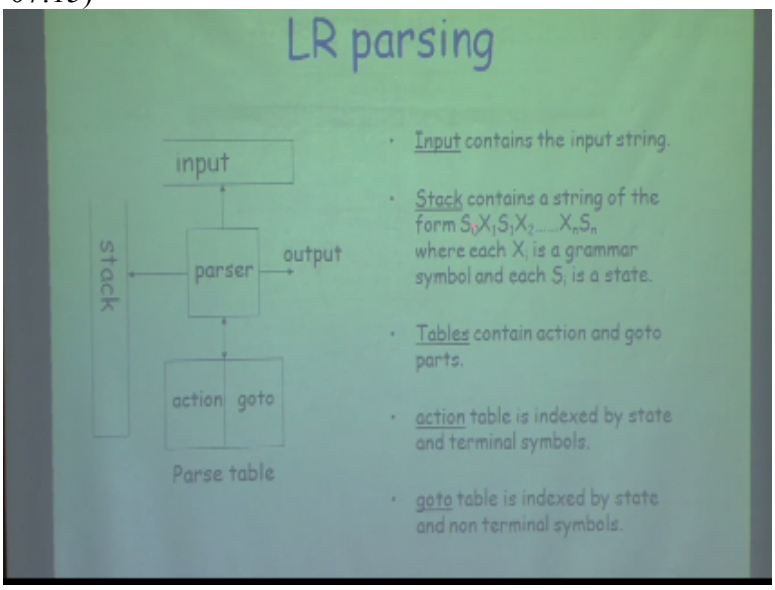
Similarly I can have a situation where I have a grammar like this and this grammar is interesting because this grammar actually corresponds to machine instructions and what we are what their vision will distinguish that even the machine instructions vowels so this is saying that in many locations can be completed by adding either two registers or a register plus the constant or the resistor then we loaded into memory location.

And the cost and can be loaded into a variable okay and if I now try to report generation for or if I try to reduce this string which is saying at constants it is basically come to the machine says that fallow the kind of instructions to sequence are want to get it so I try to sparse C plus T using the grammar lines okay now you can clearly see even without going to parsing that I can generate but multiple instructions for this so I can say load the first constant into a register then enter the register to a constant and store.

That memory and I can say that load this constant go register and the two registers and then user enable so these two patterns basically give me the project where I can get into a situation where I do should reduce parsing so I can hear say that reduce by this rule and then have our purpose we

saw and then I can say reduce by this rule and here I have another possibility where I directly reduce by this rule okay.

(Refer Slide Time: 07:15)



So I can get into multiple situations but at this point of time this example is not important what is important is that I should be able to non design process and write grammar which is going clearly okay so this basically idea of just bringing these examples to say that some issues and how resolve this issues is going to conflict the top of parser now here is the same structure or a similar structure to what we saw what I have is a stack.

I have an input then I have a parser and then I have parse table only difference is going to be that the structure of the path is going to be different and then my parser is going to different and then but basically I need to do certain manipulations and manipulations are going to be now looking at something which is at the top of stack and my input symbol of look at itself now the first thing that comes here is that the way these parts of some design .

and that's why I said that in the I have does not appear to as top of the parser and I have some concept of aspect of the parser so here my top of stack was only symbol which does not W symbol or a terminal symbol is determining that and going to terminal match now I say that my parser goes into certain states and these are states which I can just give a unique number to each stage then the matter of the number is as long as this is unique and I say that when I start pushing symbols understand at the same time.

I pushed more information on the stack which is saying that what is this stable of the part we come to this point slightly later what state means what is it that is actually okay but at we come to this points slightly later what means what is that capturind okay but this of time just to

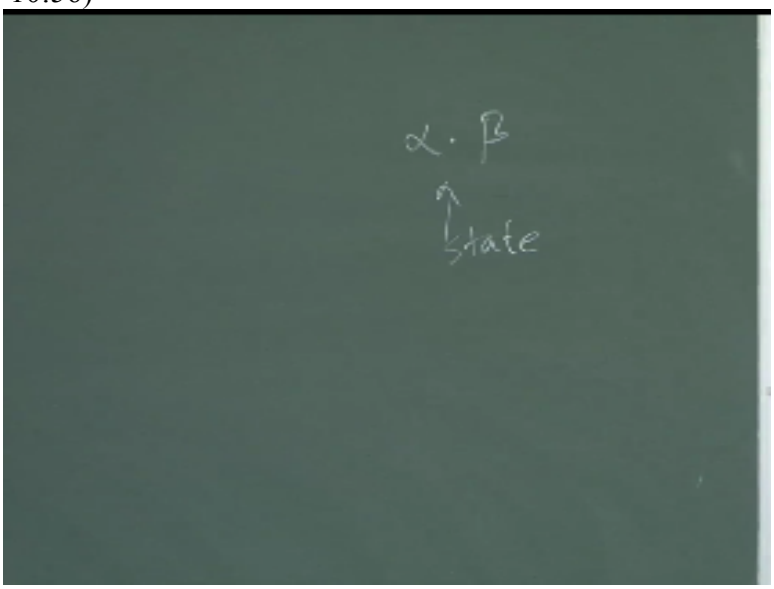
understand remember the state of the parcel is older stack of the past okay so what we are doing here is that my input is only going to contain of input strings and what does my stack contain so what is stack contains is state symbols and grammar symbols all these s0 s1 of we were send these are the state symbols and X1,X2,Xn etcetra and the γr symbols.

so these grammar symbols put identity for terminal or non terminal so what we are doing there is my input is only going to contain input strings and what is my stack contain so stack contains states symbols and the grammar symbols all these s0,S1,Sn there are the state symbols and x1,x2,xn etc grammar symbols to these grammar symbols these grammar symbols could identy a terminal or non terminal so to begin with my parts.

of is in certain state and whenever a symbol is pushed on the stack then the parcel goes into a new state so state symbol is always on the top of stack table basically this also should give you a hint of what state symbol is captain if you think of what is being captured by the state symbol we don't worry about how we capture system but does this string give you some hint of what state what state symbol is captain no so just step out there with me.

and when I say my parcel is in state I see you what does that mean that is a T that is the initial state and then I say some state s1 what that means is that part of the symbols below it so basically is defined at some point of time or possible prefixes in my language my grammar and then I say that corresponding to each of the prefix I actually have a spacing so s1 says that if X 1 if below s1 as symbol X 1 and then earlier state for the slope that is missing actually something unique which is known as state.

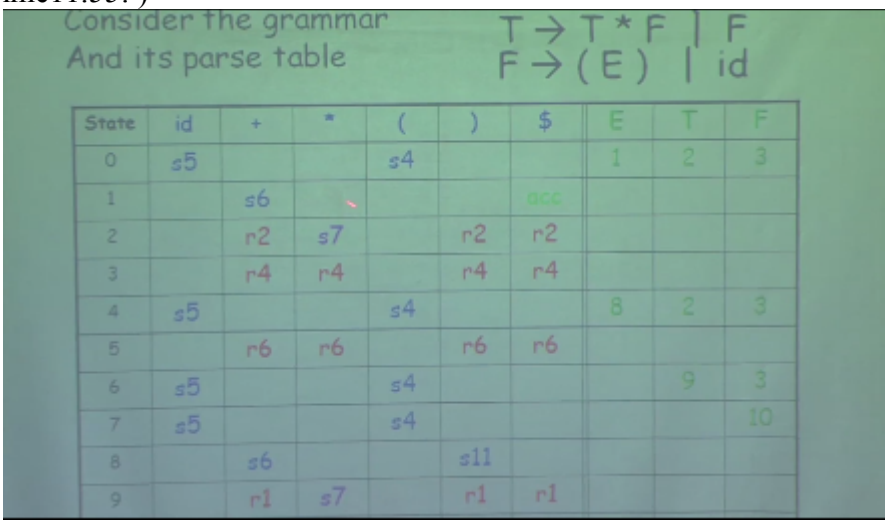
(Refer Slide Time: 10:56)



So basically if you recall now once again I said that I am going to introduce a special symbol dot and something I have this alpha dot beta okay this term called actually corresponds to the state and this say that one of the critics it has seen exactly what I try to capture in each of the state symbols okay that as we go along and start designing these states then it will become more clear that at this point of time.

You can clearly see that if I know the state some state number Then I can find out immediately just by looking at the states s what is the content of the stack and my table contains from two parts one is action part and one is go to part action part is indexed by the state numbers and all the terminal symbols and go to part is indexed by the states and the non terminals and we shot you see an example of this and this is how we do parsing so let's first look at an example and then go through this is part of this.

(Refer Slide Time11:55:)



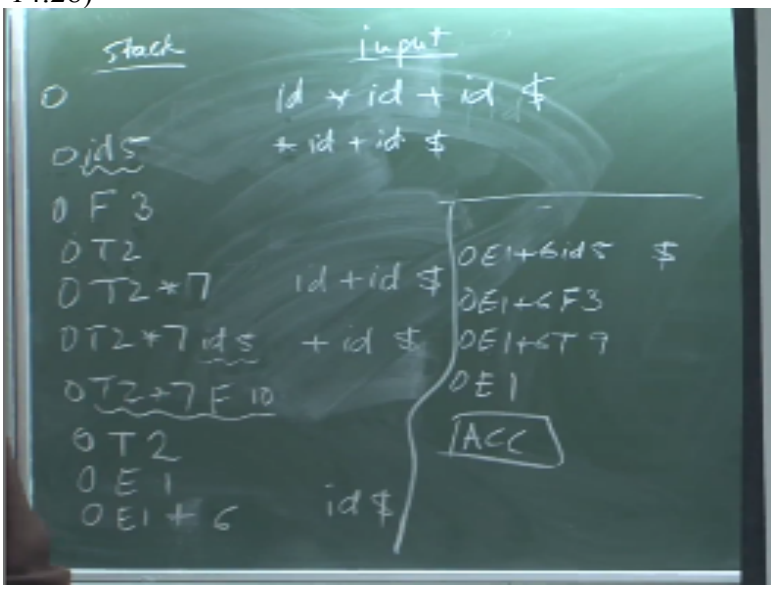
State	id	+	*	(	)	$	E	T	F
0	s5			s4			1	2	3
1		s6				acc			
2		r2	s7		r2	r2			
3		r4	r4		r4	r4			
4	s5			s4			8	2	3
5		r6	r6		r6	r6			
6	s5			s4				9	3
7	s5			s4					10
8		s6			s11				
9		r1	s7		r1	r1			

So it is a parser or a parser table form grammar language so this grammar is left recursive expression grammar I am not keeping minus and division and so on so this day is equals to e plus T goes E started for us there goes to states okay now this is the factional table these are the go to table these are my possible States this grammar can have only nine states 102 because those are the prefixes which are possible in this grammar.

And then I have these entries and go to this entries means what will these entries means what is entries mean not that if I am in state zero and I see ID then ID need to shift a simple on the stack and go to state five either have these entries as s and some number which is saying shift and go to the new state or I will have entries like saying that reduced by rule number four so this is if I will take p + IC + s symbol that means I have identified.

The handle and reduce by rule number four and rule number four is this is one two three four five and six these are the six okay so you can see that all my entries the pass table are either and

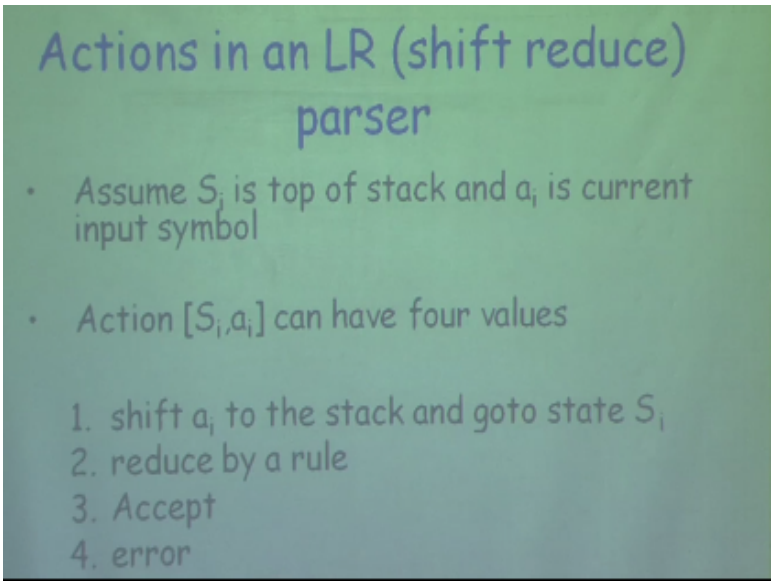
T which means that it is correct or error either a shift action or a you do section except one place where it says that I have reached accept state which means that this is not valid state so and what does this table tell me this says that if I have in state 0 and on top of state 0 the non terminal is e then I must go to state 1 okay or if I'm in state 4 and stake the non-terminal.
On top of State four is e then my parser should be state and so we do not know how to construct this table we do not even know how to use this table or that is the interpretation of this table okay now first thing I want to do is without really worrying about how this table has been constructed I would like to take an example and take this the two steps of shifted to do this possible just keep this table in mind.
(Refer Slide Time: 14:28)



So let us write parse id star id plus id okay and the dollar indicates that's the special symbol I have put at the end of my input so this is what my input is my initial configuration stack okay and what is my stack a stack should begin with is always in state symbol okay that is where my initial configuration starts and then what I need to now check is by going through these actions of this table will I ever reach a situation.
Where my state of the parser is 1 and the symbol is dollar if I reach that particular state is that clear what I trying to do so let us look at this table now also this table should tell you something like what we were looking in the top down parser if I am in state 0 which is the initial state then all the valid strings can start either with an identifier or with the left notices anything else gives me a okay so if I try to attempt to start.
A string by a plus or a multiplication or right parenthesis that is I have okay I am not worry about and T why not even say that in state 0 if I see an empty string then I should be excepts it mean

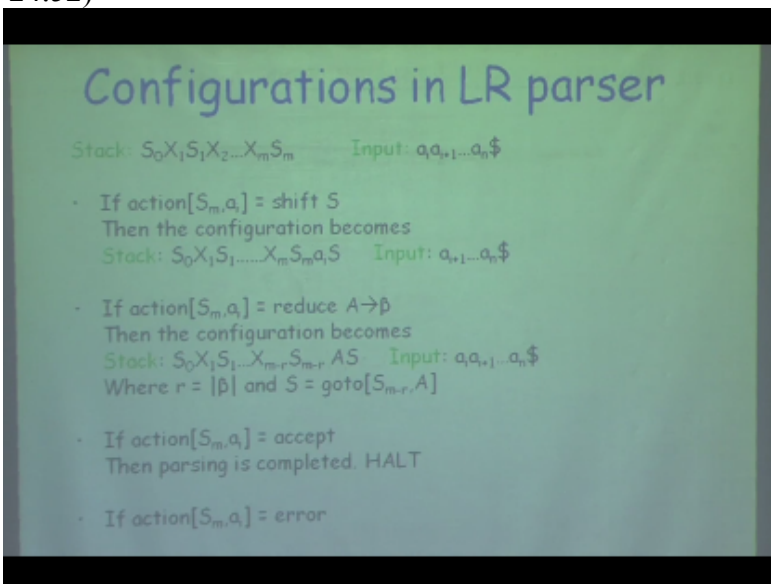
that is initial must be on all on to a grammar okay so if I am in state 0 and I see ID then what doing I shift the symbol on the set and I go to a new state which is side.
So my stack becomes 0 Id 5 so you can see that initially I was in some state and after this shifting the simple I have reached a new stage.
And what is my input now my input is id plus id dollar so and now I see that instead star if I see star in state 5 if I see star this is reduced by the number 6 my number 6 is at this why exists so when I say that I am reducing this symbol essentially what that means is that both these symbols will have to be popped from the stack okay so after I pop these symbols and how many symbols why pop now.
So how did I tried box this is one two three and whatever so I need to pop why is the length of the right-hand side the number of symbols which have to be box I slice the length of the right hand side in white because between two grammar simple they do is state symbol so for every grammar signify the length of the right hand side and a grammar symbols and top to N symbols and therefore I love to be pop another n state symbols.
So if the right hand side of one I just popped from the stack and I go into configuration like this and then what do I put on the stack the left hand side of rule so I push a different side of the rule on the stack and now I must becoming a what is that used of my parser and how I find out new state of parser and telling what is this by looking at go to table which says it's top of stack simple is this non-terminal and the state symbol.
Just below this is zero and then refer to 0 F which says that now parser must be in state T you understood how I arrived from 0 ID 5 to 0 f3 is that clear to everyone yes no yes okay so now how do I continue okay yeah now I say let me refer to three star because input has not changed and when I refer to three star this is reduced by rule number four okay so what even my set configuration somebody tell me this configuration not.
When I use rule number reduce by rule number four rule number four says e go square 0 and 0t and what will be the state two okay so my parts are we going to be state so nothing in this state right you understood what has happen so I have got two symbols I push that left hand side then I refer to table corresponding to 0 T and go into a new state now okay now I look at to stop what is two star okay what is two star says shift and go to exceptional state okay.
And now I can say that I will shift this symbol and put my exceptional what is my input my input id is ID plus id dollars so what I have done is I have shifted this symbol of the stack and because this empty set said I was in to star right so to start was taking me to sit sir now I look at seven ID

and what does 7 ID say shift and go to state 5 okay so now I say 0T to star 7id 5and id it becomes dollar this okay and what is the next thing now.
I say I mean State 5 and IC plus so if I am in state final state and IC plus what does it mean reduce number 5 with 6 state with the id so I am going to form two symbols on the stack okay so I am going to pop these two symbols and this is going to give me a configuration like this and then I am going to push left hand side on the stack.
Which is F and then what will be the new state of my parcel seven F and seven F is 10 and now if I say what is ten plus so I look at the 10 plus intense plus says reduced file rule number 3 and what is rule number three rule number three says T equals T star F so how many symbols are form to pop all these six symbols from the stack and I am going to first remain its low and then what you like push on the left hand side what you like push on the stack.
The left hand side which is T and what will be my state now 2 right so 0 is 2 so I go to state and then I will look at 2 plus n 2 plus phase reduced by rule number 2 and 1 is rule number 2 equal to T so that means I am going to split is 0 1 or 0 1 ok now in 1 if I see Plus this is shift and go to state 6 so I am going to know 0e1 b6 so this should be plus and not if I sign my input is not ID T okay and now if I see so let me continue on this part okay.
Now if I see six Id and what is the six id says shift and go to state fight so that we take me to 0 e 1 plus 6 Id 5f and my put will be us okay and I fight you get $5 $5 says reduced by rule number six and six is at most yd so that takes me to the stage this in what is 6 X 6 F is 3 so I much parser goes into state tree and I get t dollars and p dollar to reduce the rule number four and rule number 3 is t goes to F so my part now becomes 0 e 1 plus 6 T and what is 669 okay.
And now I look at $9 $9.00 reduce the rule number one and rule number one is E goes suite plus T so I am going to pop six symbols from the stack so I pop one two three four five and six and then I push left-hand side on the stack and then I am going to state one and I look at one dollar and what is one dollar accept so we should find this thing that term you understand how this parser will being interpreted now right okay.
So let us now, so we still do not know how to take this grammars and encode it into a pasta bit like this but what we definitely know now is that given a pass table and given an input how do I go through the process possible. So let me just take you through the same thing you have already seen this if this part. So what are the actions? What are the things I am doing? So action swap that I was looking at the top of the state symbol which was the stack state symbol of my parser and I am looking at the input symbol and based on these two combinations.
(Refer Slide Time: 24:37)



I was taking actions we said possibly I can shift I can reduce I can accept or I can go in and what are the possibilities?

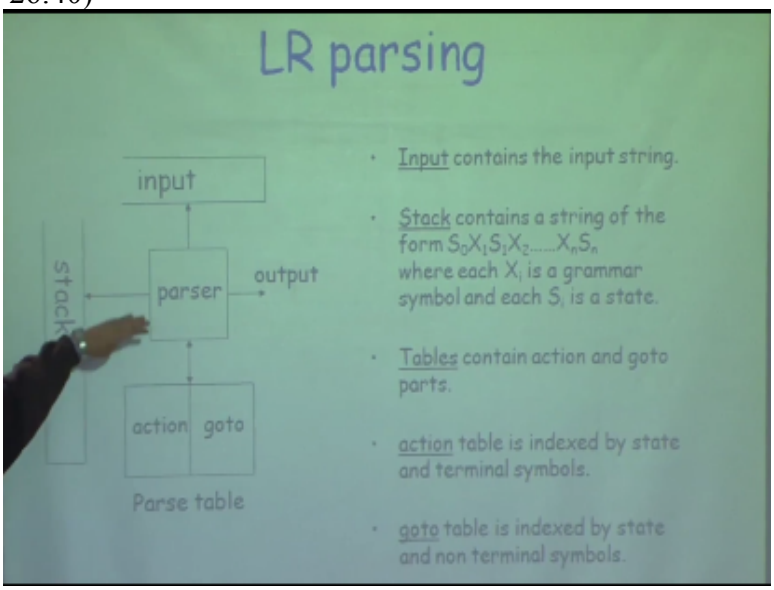
(Refer Slide Time: 24:52)



In terms of if I use these symbols if I say my stack is in certain operation like this we have top of stack is some state then action will say that so I am looking at this page symbol and I am looking at AI and I refer to the action table and if I look at action table and entries corresponding to shift and go to a new states. What that means is that after this configuration is going to be that I am going to push here me on top of stack and go to new state S and my input will now have one less symbol. Are if it is a reduce section which is corresponding to some rule then what will I do I find out length of β I will fall twice the number of symbols and β.

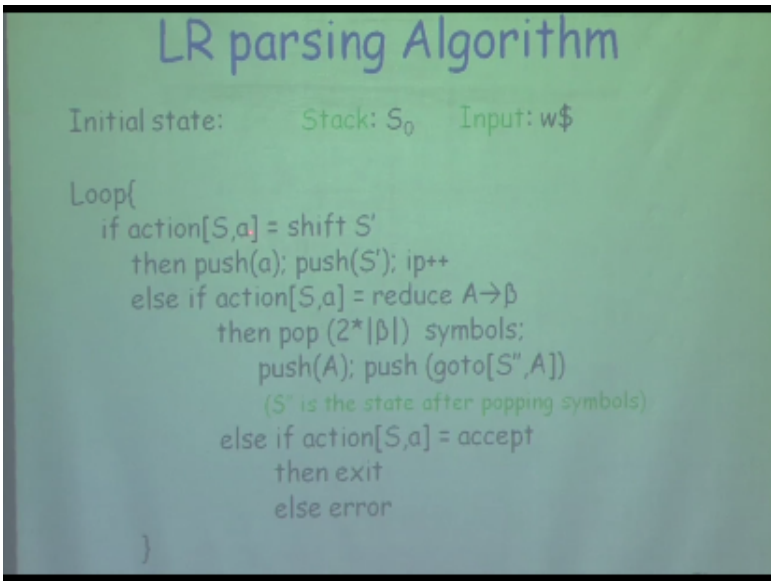
So what happens in is length of β is R then I am going to pop to our symbols from this and after talking to our symbols I will be left with top of stack has Sm - R1 A then I push left hand side and then I am going to go to a new state which will be corresponding to boots walk SN - r and D. Right okay and then either be in the accept state so if this is accept then what why do I just respond that I have a listing and if this is error state then we like to do some invocation of parsing action here to everyone. Next if I go back once again to my parser let me take you back to the same figure okay.

(Refer Slide Time: 26:40)



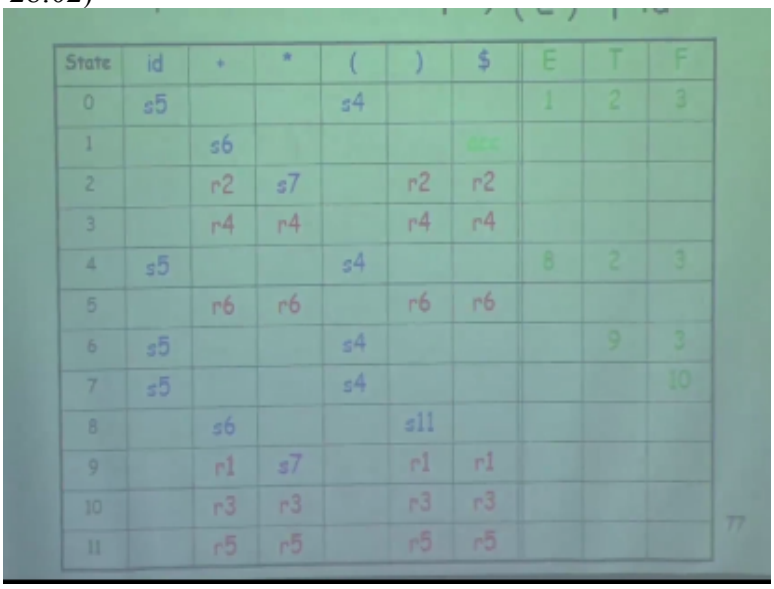
We at least understand how I manipulated stacked or do a manipulate inputs and what does my password do okay. But I am not understood so far this that is how does i construct action table to the parse table. Write this figure this is the only box now which is unexplained. Everyone is in sync everyone understands how the parser is working it okay. So let us then move on,

(Refer Slide Time: 27:16)



So again we can leave the same thing if I try to write as a block my initial state is s 0 or 0 and my input is W dollar and if action is shift then I put am the sky pad push s pan on the stack I am the PI bond if action is reduced by this rule of symbols. So once again okay let us jump to this table and try to see this table.

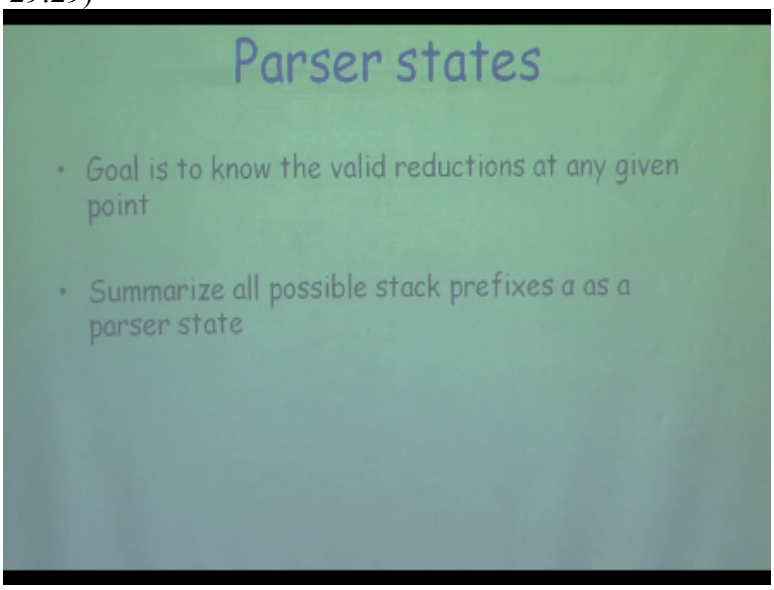
(Refer Slide Time: 28:02)



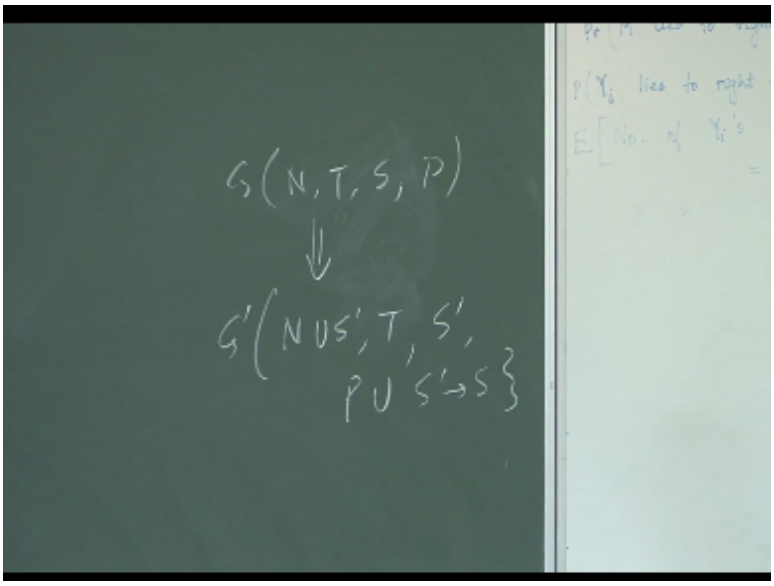
State	id	+	*	(	)	$	E	T	F
0	s5			s4			1	2	3
1		s6				acc			
2		r2	s7		r2	r2			
3		r4	r4		r4	r4			
4	s5			s4			8	2	3
5		r6	r6		r6	r6			
6	s5			s4				9	3
7	s5			s4					10
8		s6			s11				
9		r1	s7		r1	r1			
10		r3	r3		r3	r3			
11		r5	r5		r5	r5			

This table I need to have few more definitions and some more information so first information that is going to come is that what these stakes in loosely these take symbols. Has I said what they mean is that what is on the stack so it is actually looking at what is understand and it cannot be an infinitely long straight because all the time I will keep on doing reduction what is the maximum height of the fact. Number of twice the number of focus +1tight. So if my length of input in hand and it is possible that across all shift before I do first macadam.

You can have in the worst case so and so this is actually defined by this so except state are symphonic six machines are nothing but boney productions some is I came to uniquely find which is the right production whose views so for example here for example I was able to say I want to reduce by when I was in this state here I was keep on shifting.
(Refer Slide Time: 29:29)



But some point of time i was doing reduction and I need to do what is deduction rule I need to use and state summarizes all possible access okay in any part of this okay. So for example in 0 what are means is that my passers state is not happening. And similarly these states we capture something move. So this is the parses state actually defines by again a final statement sheet okay. And it is reads stack this final statement I was doing a reduction in so okay so there are some states in which I was able to find absolute reduction and these reductions. Who are finally telling me that I can eventually reach a start state okay so let us start slowly inspecting the pasta whatever definitions we need to know for that we will introduce the first step I do is a modified grammar itself .
(Refer Slide Time: 31:06)
And what I do is I say that I want to augment my γ okay and what does augmentation of grammar me computation of γ means is,
(Refer Slide Time: 31:12)

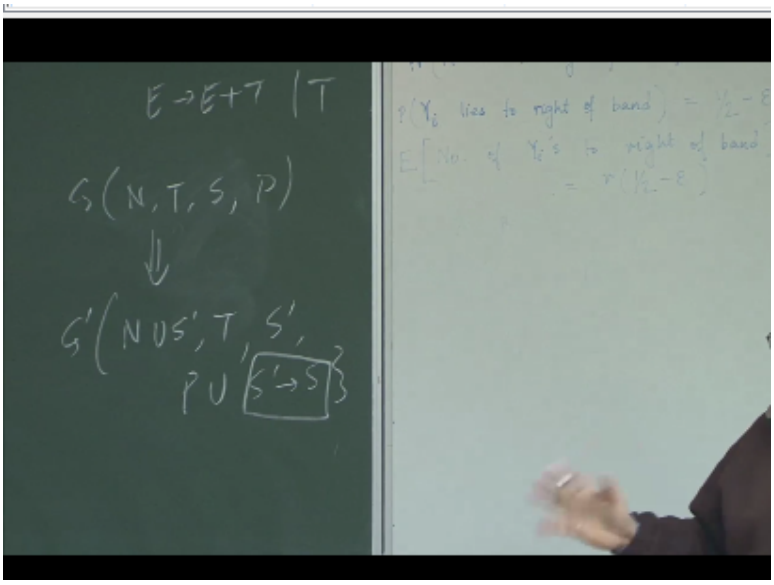


That if I started with some context-free grammars and this context-free grammar consists of a set of non terminals. Some start symbol and the set of productions then I am going to modify this and one with the modification so let me call this Omega γ as G Prime and now I am going to say that I have one more non terminal which I am going to change as finally I am going to not change the set of my terminals and change my start symbol to s Prime and I change my productions to say that whatever productions are had and I have one processor.

So what is a new non terminal which I am going to call the new start symbol and I have introduced one more rule which say that as fine goes place does it anyway changes the string does not right. Then why I am doing these manipulations but that is not really the real meaning that this man should not be I mean what you are saying is direct observations that s fine will never be on the right hand side of any production.

That is not the real reason that describe is not being should not I could be given the introduction. We will what saying correct observation that destroys in any production. But not the reason I am changing now these are not changing it is I am now saying that only when I do a reduction by this particular rule that is the exact state in earlier cases when I have something like this so for example when I said,

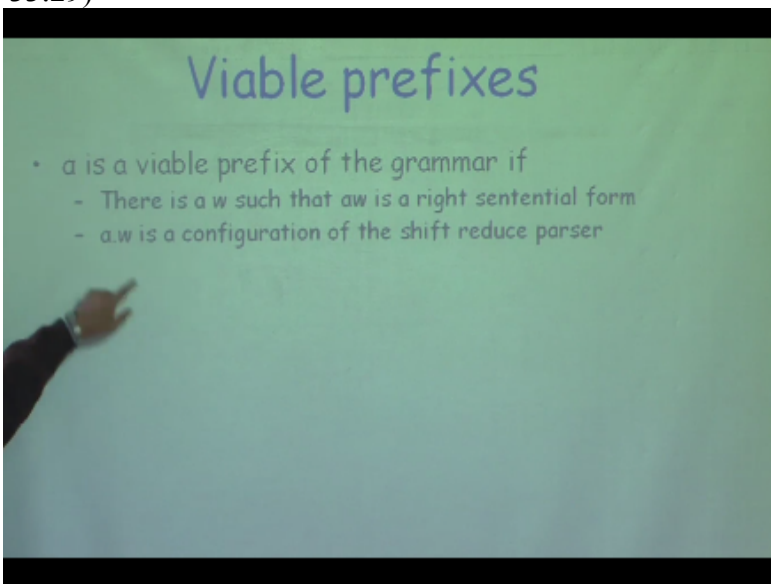
(Refer Slide Time: 32:49)



He goes to E + T or e go sweetie I put breach accept state either by reducing this or by reducing this or by reducing this okay. So therefore I think multiple entries they could have multiple ending states will take that are a state. This is a unique situation where I say that only by reducing this way I reach next the state. So there only a small automatic change does not gain anything in the language I am dealing with.

So if G is a grammar it starts simple less then Augmented grammar G prime as a new start symbol S Prime and you expand west and when parcel reduces by this rule this population where else so this is the first change I make,

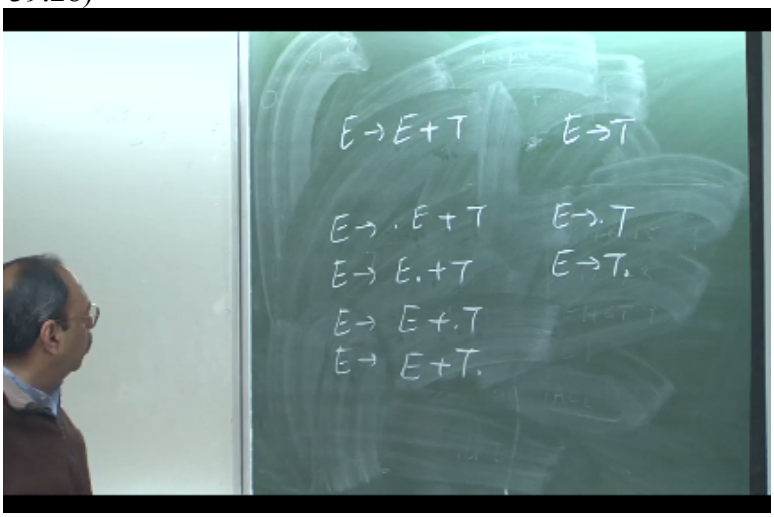
(Refer Slide Time: 33:29)



Now I define viable prefixes okay now the reason I want to go to this replication of viable prefixes because at some point of time I want to capture all this information about States so this

is saying that suppose I am doing these reductions state has certain symbols and stack has certain symbols and a certain grammar and the stations. So prefix is viable prefixes that if I take all possible solution forms in my language then what could be that fat configuration.
I want to find out all possible stack complications because those are the in my values and forms those are the only valid parsing States I can pack everything is visited so this is what I want to be find that one of the value prefixes in any configuration of shift reduce parser and we say that α is the viable and depends on not on the language but on the form of the town those different grammars be giving different state on is that configuration.
So what we are saying here is that as Y is some symbol and finds to a string of terminals and non-terminals and this is the viable prefix in the drama if we have some such W okay such that α w is in right sentential form it means if I do a right most derivation then I get some past thing or something somewhere where α w will be the same and also that valid configuration of my task okay.
And also that α W is shift reduce parser that means if I do the reverse of rightmost derivation then sometime I will be able to see α from the stack and so you can see now what I am doing if I look at this thing okay if you see stack configuration you can see that when I wished state 3 okay what was my prefix my prefix was that my parcel initially was in space 0 and the only symbol .
which put me on the stack was 3 okay or here only simple my parser could have bus T my parcel to have these stops so if I put assign me if I ignore all state symbols or here I could say my parcel could have these * ID I could have T * F if I cannot have something arbitrary like peace ignore that will never be my parser congregation. So what I start counting by looking at the novel is on what I will start para all possible deviations of my pack.
Because this viable prefix is the one which captures all my state information okay so as long as parser has only viable prefixes in the stack then I know that parcel has not seen an error if parser is always in Manistee but a stack confirmation which is not correct then immediately I say that who she produced because this leads to a situation where my posture this does not contain about the purpose.
So set of sacrifice with prefixes is a regular language and now this is not an obvious proof and this is what actually when you say setup vigor to success is like regular language that is where we are actually using it yet. So what we are computing our first start in Atlantic City lover I compute all sets of viable prefixes and then I say since this is actually a regular language.
I can implement it by a finite state machine who devised bottom up parser setup process by the language design any idea no idea better known work in advancer okay with isolation and ignition

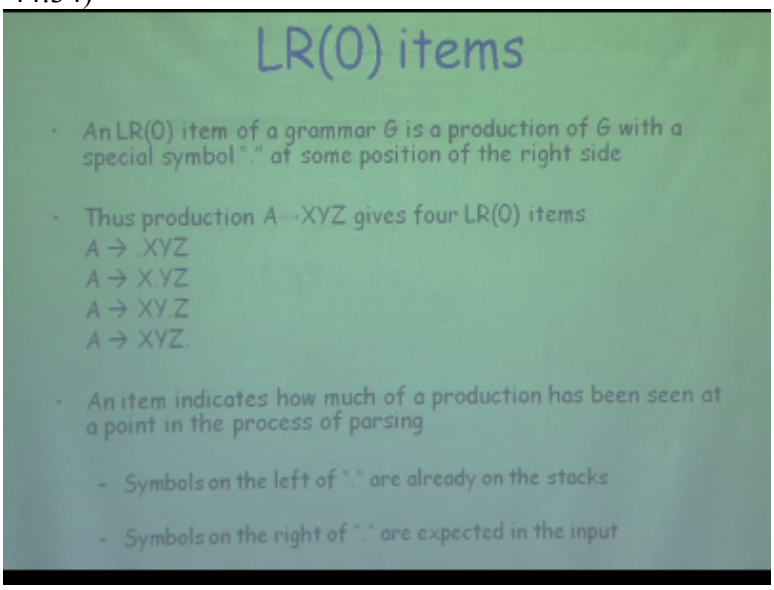
is better known for is benign here it comes but this is something you did it way last in 1963 1964 in fact his port value of art of programming supposed to be like in 7 volumes on you FEMA published.
So this is the proof which is given in his original paper that if I take any kind of 160 table and I try to find by other fixes for this in fact I will put packet of parsing was so difficult that people want you been implemented for a long time failed somebody gave a much better method or Redeemer he gave a method for constructing in always simple as a process so we will go through all simple and our courses.
And then we look at the most complex ones but what we are talking about this is important proof that somebody for visualize that if I construct all viable prefixes that actually gives me a regular man so what we do is once we have constructed all viable prefixes we just need to construct a finite sequence equal in value most night if you now recall the past table what is my possibility that is the finite state.
Machine right it is not doing anything else so first term our customer so first thing I do is I say that and this is my attempt to construct all the viable prefixes I say that I define something known as a log now I say that L R 0 I okay now if I look at any rule look at the grammar.
(Refer Slide Time: 39:28)



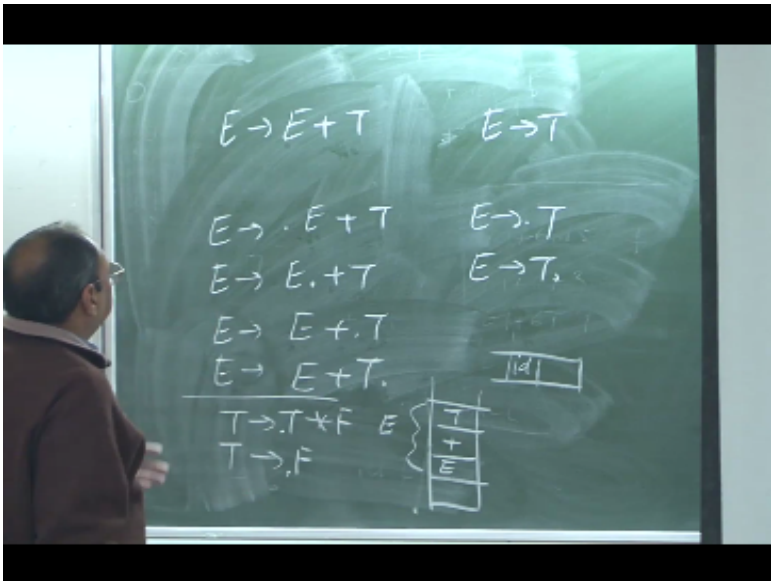
Suppose is my grammar think puts thought anywhere on the right hand side okay so suppose I put got anywhere on the right hand side obviously it has between two symbols so I can get blame is either I can have a dot in the first position or I can have a thought in the second position or I can have a thought in this position all I can have forgotten this position yes then what is in stock what is the Special Master P had.
Which says what is on the stack in what we mean to me so this is that if I want to do reduction by this rule so conceptually what is happening now is I am saying that I am trying to now

compute all viable prefixes so this is of this what is the viable prefix now and for this configuration what is the viable prefix E and for this is E + and this is E + T from this grammar rule I cannot have any other faculties these are the only combinations.
I can tell and similarly I need to do it for all the terms of rules are have so if I have this rule which says equals T then what are the allowed view items from this so this I call as set off a L R E Y so set up lr0 items are that if I think now this is not a production practices in lr0 okay so in another item is that if I take γ and I put a dot anywhere on the right hand side it becomes and this is giving me some unique combinations.
So what did I learn two icons I can have from this I can have either this or I can have this is nothing else I can and what I need to do now is for all some of our symbols all possible this is same fact I can if I want to do this reduction conceptually then they say that I have not seen anything but I must be a derivation of a plus T if I want to reduce this to an E and E + these are handy this is if I have already seen e then I am expecting to see a derivation of + P and this says if I have already seen a plus then I am expecting and this is.
If I have seen a plus T that does not matter what comes in okay I am ready to reduce this to the left hand side okay so what I said in the end is not entirely correct but I do not want to introduce at this point of time okay so what are the situations under which I can do reduction by this rule e so what should be so here says that suppose I am in this configuration okay I have seen e then what I am expecting to see the plus right and if I have in this configuration.
Where I have seen a less then I am seeing that I should be able to see something which is their removal from that means obviously the first symbols of B and if I have seen a plus P I am ready to deduction on e under what circumstances so look at my parcel what will happen is my parcel state I will have E Plus P and this is my input and then say that I am ready to reduce this pool okay now what are the situations.
Under which this reduction and remain valid that this Plus this will be of an extension form in my language is there a property I can attribute to this symbol can it be any symbol can it be any arbitrary simple in my γ phones like this is some difficult question I mean which are not able to even visualize can I put say ID here no so I must have a symbol which is 400 feet because if I do not have a symbol which is in follow of e okay.
Then I have a problem obviously because I not be able to reduce any further so whenever you are ready for reduction by this you say that if I have reached a configuration where dot is in the rightmost position and I have seen the whole of the right hand side of a production I can reduce it

only if so what we need to do therefore is I need to construct an RD and net all possible sets off all so an LR 0 item is.
(Refer Slide Time: 44:34)



In a γ is a production in which God is in some position on the right hand side and if I have a production of this form it says it goes to X Y Z then one of the isle of zero items I can get I can get these possible and of 0 items that dot can be on the leftmost position after next after Y your items okay I can get and conceptually this is that how much of production has been seen at a point and what is it that I am expecting to see symbols of the left hand dot.
Are already on the stack and what I am expecting is that many symbols on the right hand side must be inside the input but derivation of the right sign it because if I say that my right hand side is in P obviously P will never be in input consists only of my terminal symbols so if I see a derivation of T then first time we will push those on the stack reduce them to T and then only do the shape now find out all these possible sets of lr0 items one operation I need to introduce.
Is what we know as loser operation or disclosure oxygen goes of operation is saying and we eat this part here that somewhere if I say that I have seen something in my inverter and I am ready to see something more suppose I reach what this means is that I have seen alpha and I'm expecting to see a derivation of and what I will do is I will say that I am going to define this and I say that conceptually.
(Refer Slide Time: 46:13)



If I am expecting to see an elevation of bb-dub then I must eat first elevations of Δ derivation is corresponding to be right back to the non-terminal okay so now he has some rule of this form a mix is B goes to L E X A T QR something like this then I must add an item which says V goes so this is my this is my saying that if I have enough zero item like this and I want to find explosion this energy of item says that I have seen n bar and I am expected to see a derivation of me be done Δ.

But since P is a non-terminal there must be some rules corresponding to be and if this is the rule corresponding to me then I am expecting to see a derivation of so for example here if I say that I am expecting to see a derivation of T β then what are the rules corresponding to T those corresponding to B goes to Pr * F or t goes to P q r which means that if I am expecting to see a derivation of P then first I must see either this derivation which eventually.

I can reduce to T or I must see this derivation which even today I can reduce to so close and operation is saying that starting from analogue video right now if I want to find its closure then add that particular large item into the set and if there is a non terminal immediately on the right of thought take all these productions foot part in the leftmost position and those are the end of the items I am expecting to see so let us stop here today and tomorrow we will start our discussion from this point onwards. And remember that we know the extra class starting here 11.45.

Acknowledgment
Ministry of Human Resources & Development
Prof. Phalguni Gupta
Co-ordinator, NPTEL IIT Kanpur
Satyaki Roy
Co Co-ordinator, NPTEL IIT Kanpur
Camera

Ram Chandra
Dilip Tripathi
Padam Shukla
Manoj Shrivastava
Sanjay Mishtra
Editing
Ashish Singh
Badal Pradhan
Tapobrata Das
Shuubham Rawat
Shikha Gupta
Pradeep Kumar
K.K Mishra
Jai Singh
Sweety Kanaujia
Aradhana Singh
Sweta
Preeti Sachan
Ashutosh Gairola
Dilip Katiyar
Ashutosh Kumar
Light& Sound
Sharwan
Hari Ram

Production Crew

Bhadra Rao
Puneet Kumar Bajpai
Priyanka Singh

Office

Lalty Dutta
Ajay Kanaujia
Shivendra Kumar Tiwari
Saurabh Shukla

Direction

Sanjay Pal

Production Manager

Bharat Lal

an IIT Kanpur Production