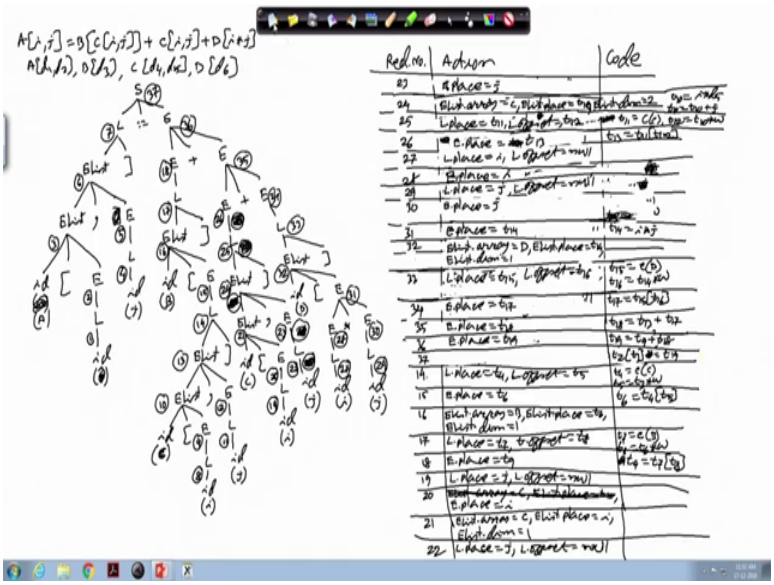
Compiler Design
Prof. Santanu Chattopadhyay
Department of E & EC Engineering
Indian Institute of Technology, Kharagpur

Lecture - 61
Intermediate Code Generation (Contd.)

(Refer Slide Time: 00:17)



So, at reduction number 13, we will have the next action. So, reduction number 13. So, it will have this E list dot array E list dot array equal to C then E list dot place equal to t 3 and E list dot place equal to t 3. And this E list dot dimension equal to 2. And then it also generate some code, the t 3 equal to i into d 5, then t 3 equal to t 3 plus j ok. So, that is 13 at 14 a reaction 14. So, it will do that a at reduction 14 is this one E list bracket close.

So, it will do L dot place equal to a new temporary t 4. And L dot offset equal to another new temporary t 5. So, t 4 is equal to the constant part of c and t 5 is equal to t 3 into w. So, this two lines of code will be generated at reduction 15. So, it will have this E dot place equal to t 6 ok. And it will generate the code t 6 equal to t 4 t 5. This is a one dimensional array axis t 6 equal t 4 t 5. Then at reduction 16 so, it will have the action that E list dot array; E list dot array equal to B and then E list dot place equal to t 6 E list dot place equal to t 6 and E list dot dimension equal to 1 ok.

Then at reduction 17, it will have L dot place equal to t 7 another new temporary and L dot offset equal to t 8. So, another new temporary, where t 7 is the constant part of B and

t 8 is equal to t 6 into w. And so, that is 17 at 18 at reduction 18 so, E producing L. So, it will generate a new it will take a new temporary and E dot place equal to t 9, where t 9 is equal to t 7 t 8 t 9 equal to t 7 t 8.

So, that is 18. Then 19 at 19 so, you will have L producing i d. So, this L dot place equal to i d dot place that is j and L dot offset equal to null; L dot offset equal to null. So, this is the action at 19, at 20 so, it will have this E list dot array equal to C E list dot place equal to t 10, a new temporary. And E list dot dimension sorry at 20 sorry at 20 it is a E producing L sorry so, this is not correct. So, E producing L so, E dot place equal to i; so, E dot place equal to i.

Then at 21 at reduction number 21 we have this one this E list bracket close E list comma E. So, that part 21 we have E list dot array equal to C E list dot place equal to i E list dot place equal to i, because this id is C and this E dot a dot place is already i. So, that is E list dot place is i and E list dot dimension equal to 1; E list dot dimension equal to 1. Then at reduction number 22; at reduction number 22 so, it will do L reduction number 22 is this one, L producing id. So, will have L dot place equal to id dot place that is j and L dot offset equal to null.

Then comes production number 23 at reduction number 23. So, we have got this E producing L. So, this E dot place equal to L dot place so, L dot place is j. So, E dot place is equal to j. And no code is generated, at reduction number 24, at reduction number 24 we have this one, E list comma E, so, E list comma E. So, for that E list dot array has to be copied. So, E list dot array is same as 21 E list dot array.

So, E list dot array equal to C. Now E list dot place is a new temp. E list dot place is a new temporary t 10 and E list dot dimension E list E list dot dimension will be equal to 2. And the code that will be generated is t 10 equal to i into d 5 and then t 10 equal to t 10 plus j. So, this is the code generated at 24.

At 25 so, E list L producing E list bracket close. So, it will do L dot place, L dot place equal to E list L dot place equal to t 11 a new temporary. And L dot offset is another new temporary t 12. And it will generate some code. So, it will generate code like t 11 equal to the constant part of the array C of c and t 12 equal to t 10 into w. It will generate this code at 25. Now t 26 I have E producing L. So, this E dot place; so, this E dot place will be equal to L dot place. So, L dot place is t 11. So, E dot place equal to sorry E it will be

at E dot place will be a new temporary t 13 s this an array. So, t 13 is equal to the array axis t 11 t 12 the array axis.

Now, reduction number 27 at reduction, 27 we have this one L producing i d. So, this L dot place equal to i L dot offset equal to null. Now comes reduction number 28 at reduction 28, we have this E producing l. So, this E dot place equal to i. So, that is reduction number 28. At reduction number 29 we have this mm at reduction number 29 we have L producing id. So, L dot place equal to i d dot place that is j and L dot offset equal to null. Now at reduction number 30 E producing l.

So, this E dot place equal to L dot place. So, E dot place equal to j. Now reduction number 31; at reduction number 31 we have 31 so, this E producing E star E. So, for that it has to generate some code that is a new temporary has to be generated and E dot place will be a new temporary t 14, where the code generated will be t 14 equal to i into j ok. Multiplication of this mm the this E, E 1 dot place and E 2 dot place.

So, 28 and 30 so, 28 is i and 30 is j. So, their product is taken as 30. Now reduction number 32. So, in reduction number 32 we have got this rule E list producing id open parenthesis E. So, for that we have this L dot place; L dot place equal to sorry a 32 is E list something. So, this E list dot array has to be created. So, E list dot array equal to this new array d an E list dot place equal to t 14 and this E list dot dimension equal to 1 so, that is 32.

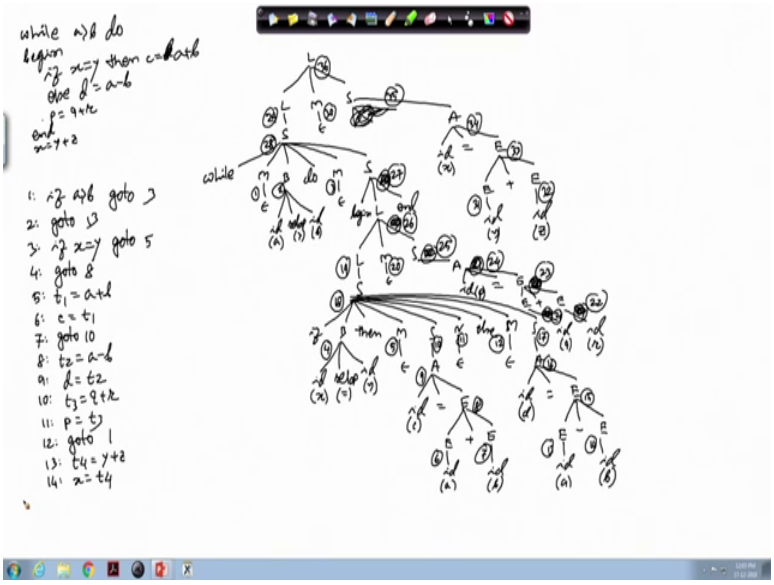
Now, at 33 so, it will 33 is this one, L producing E list bracket close. So, E list dot array equal to E list dot array equal to d, then E list dot place equal to t 14. And E list dot dimension equal to; E list dot dimension equal to 1 sorry this is 32 is reduction 33 is this one L producing E list bracket close. So, it will be L dot something. So, 33 L dot place is equal to a new temporary t 15 and this L dot offset is equal to another new temporary t 16.

Where t 15 code is equal to the constant part of the d array and t 16 code is equal to t 14 into w. Now comes reduction number 34 at reduction number 34 we will do that E list dot place sorry 34 is E producing l. So, E dot place equal to a new temporary t 17. Where this new temporary t 17 is equal to t 4ty t 15 t 16 is an array axis, so, t 15 t 16.

Now, reduction number mm 35; now reduction number 35. So, this is E plus e. So, 35 it will be doing E, E dot place equal to a new temporary t 18. And then it will generate code the t 18 equal to t 13 plus t 17. That is 35 at 36, so, it is again E plus E. So now, it will generate another new temporary E dot place equal to a new temporary, t 19, where t 19 is equal to t 9 plus t 18. That is 36 and 37, so, it is L pa s this assignment statement. So, there is no action only that id dot place. So, that is L dot place of reduction 7. That is t 2 t 3 L dot place L dot offset that is assigned as t 19.

So, in this way the final code that is generated will be final code that will that is generated. So, you can just club all these parts to get the final code. So, next we will be looking into another example, where we will try to generate the code for some statement of some language.

(Refer Slide Time: 16:50)



So, we will take this example while a greater than b do begin, if x equal to y then c equal to d c equal to a plus b, else d equal to a minus b, b equal to q plus r and then x equal to y plus z. So, we will draw the parts tree and then you will be you should be able to follow the code generation portion that we have seen previously and then we will see the final code. So, that will be like this. So, it is L M S because their multiple statements. So, we have to do it like this then the L gives me S and this L give this S gives me that while loop. So, while M the Boolean condition B then another M and S then do then another M then S.

Now, this M gives me epsilon. So, this B should give me the Boolean condition, that id bellow id it. So, this id is A this (Refer Time: 18:30) is greater than, so, this is your B. Then this b gives us epsilon and then this S should give me this begin end part, so, this is begin L end. And then this L again there are 2 statements the if statement. And then this p equal to cube plus L statement. So, it should be L M S like that this L should give me S. And this S should give me if then else statement. So, it is if B then M S N else S sorry L M then S. So, this is if then else statement. Now this B will give me that id (Refer Time: 19:32) id that is your x equal to y.

This M should give me epsilon ok. Then this S should give me this c equal to a plus b this assignment statement. So, this S give me A which is id equal to E. This id is c and this E is E plus E this E gives me id which is A this E gives me id which is B. Now this n will give me epsilon this m will give me epsilon now this S. So, this s should give me the else this else part. So, that is d equal to a minus b. So, this S should give me A which is id equal to E.

So, this id is d and this should give me E minus E this E is id this E is also id, so, this is a and this is b. So, this is the d equal to a minus b portion. Now this m gives me epsilon now this S will give me the next statement p equal to q plus r ok. So, this p equal to q plus r, so, it will have S producing A, S produce A producing id equal to E and this E giving me E producing E plus E. So, this is id this is id, so, this id is your c. sorry this id is your p, this is q and this is r. This is drink p equal to q plus r.

So, we are done with this begin end part. So, n gives me epsilon then this S gives me sorry this S gives me the assignment statement A and this A gives me id equal to E. So, this id is x and then this is E plus E this E gives id which is y and this E gives id which is z. So, this is the whole parts tree.

Now, if we think in terms of reductions then this is the reduction number 1, then this is reduction number 2, this is reduction number 3. Now after that we have got reduction number 4 as this one. Then this is reduction number 5 then we have got this one as reduction number 6, reduction number 7, 8 9 10 11 12. Now this one 13 14, 15 16 17 18, 19, 20 21, 22, then this is 23, this is 24 ok. So, this is 25 oh in between some numbers we have missed.

So, this is 12; so, once this if the end part is over that is 17. So, 18 onwards we did mistake 18 19,s, these are wrong. 18 17 is this reduction. So, then this reduction should be 18. This should be 19 this should be 20. After that this should be 21 22 23 24 25 26 27, then this will be 28 this is 29 30. Now this is 31 this E producing id this should be 31, this is 30 2 this is 30 3 this is 34 this is 35 and this is 36 fine.

So, if you use this part tree and follow the grammar rules that we have for this for this high level language statements. So, it will generate a code and the final code that will be generated will look something like this. So, at offset 1 so, it will generate code if A greater than b goto 3. At 2 it will generate goto 13 at 3 it will generate if x equal to y goto 5. At 4 it will generate goto 8. At 5 it will generate t 1 equal to a plus b 6 it will generate c equal to t 1 7 goto 10.

At 8 t 2 equal to a minus b, at 9 d equal to t 2, at 10 t 3 equal to q plus r, at 11 p equal to t 3. At 12 generate code 2 1, then at 13 it will generator t 4 equal to y plus z 14 it will generate x equal to t 4. So, you have to follow the rules that we have this for different production rules that we have for the grammar. And in fact, all this rules that we have, so, all these rules for assignment statement sementic action. So, this is for array, so, for other statement. So, we can do that. So, that way if we follow those rules then you will get this piece of code and if you check this piece of code c.

So, the if a greater than b goto 3. So, it comes to c. So, this is the body part and other if a is not greater than 3 then it will come to goto 13. At 13 it is outside the loop. So, it is computing y plus z and then x equal to t 4. So, x equal to y plus z, so, this computation is being done. So, if comes to 3. So, it is a body. So, it checks if x equal to y goto 5. So, it move 5 it computes at a plus b into t 1. And then it c it is a t 1 is assigned to c then it goes it says goto 10 because it is keeping over the else part.

So, it comes to the goto 10. So, t equal to t 3 equal to q plus r then p equal to t 3 then it is goto 1. So, it is restarting the while loop. And if x is not get a not equal to y then it will goto 8. So, here it will do this a minus b then d equal to t 2. So, d equal to a minus b and then it comes to the then comes out of if then else statement. So, p equal to q plus r. So, that is done and then again goto 1. So, it will be branching to this, so, this way. So, this piece of code faithfully replicates the code that we have for the while loop. And in this

way we can translate any set of statements following the rules that we have discussed in our class for generating the 3 address code.