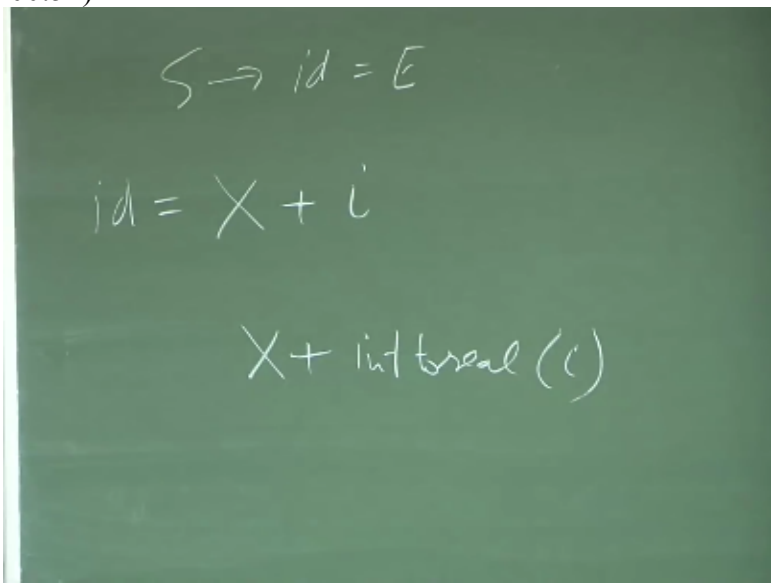
Indian Institute of Technology
Kanpur
NP – TEL
National Programme
On
Technology Enhanced Learning
Course Title
Compiler Design
Lecture – 19

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

Good morning so let us start our discussion of in topic which is a type conversion so it shows from previous class and the discussion towards and end of the previous class start looking at what will be now we will check my presents of ten we have type casting will be the issues that the time of cogeneration which is increase at that to be done so we have looking at some kind of source.

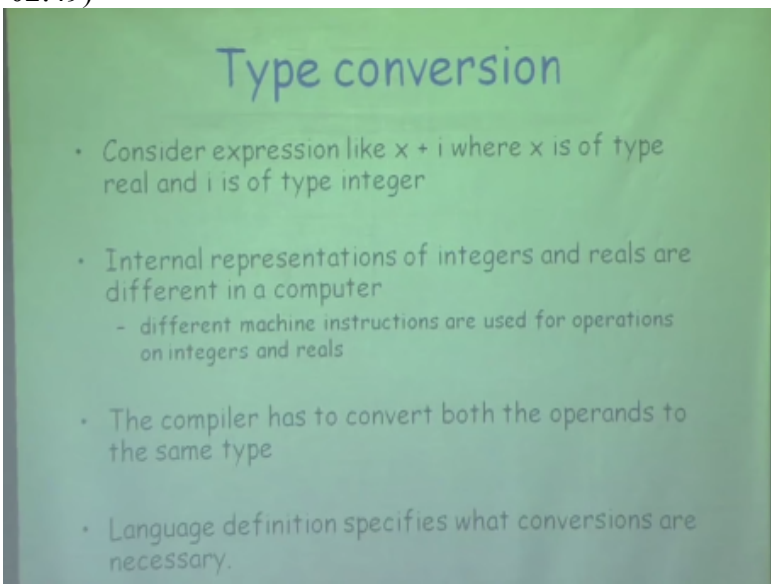
(Refer Slide Time: 00:51)



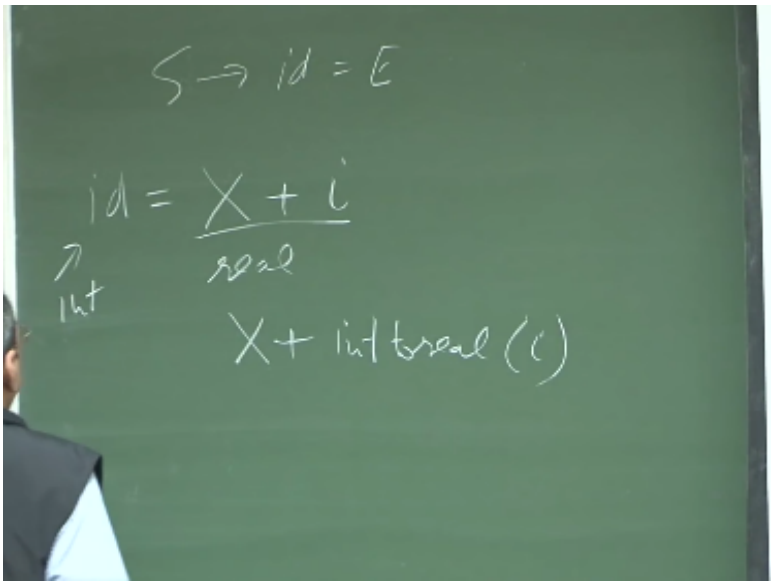
Where we have and expression of this form and I of X now the two issues there one that wish to same representation of so what I need to represent condition of I finally I want type expression to have some kind of type which is consistent for both these variables and for that expression and second I want this that I should be able to find out what is the type expression which is associated with this okay.

So it becomes very clear at least from whatever we know of a programming languages that shall we try to this type of floating point and the S both in converting in floating point so that means

basically interval you look at I what will happen is something like this type of conversion will increase it will happen and once this conversion happens then.
I know that this type is floating point and therefore type expressions that is associated with both type expression which is of X plus I becomes floating point and this is the information I carry because finally if I say that I am going to say two an assignment of this form okay before I do an assignment as I should be able to find out what is type of X plus I and what is type of ID and whether this is consistent assignment a lot remember that.
We have rule like this that we said S goes IDE is the final expression there I wrote the rule same if I define an expression and there are the rule is if ID is E type than expression I decide this you type that remember this okay so this background let us look at what kind of rule of generating what other things.
(Refer Slide Time: 02:49)



We need to worry about okay so if we take an expression like x plus I here X is type real integer and I type integer the I need to internally came representation of I let me sure that is it compatible with real and because the operations which are going to move the machines and are really to diff different and the compiler has to convert both operation to same time okay so this operand is now being converted to this particular time and who defines it obviously the language is telling us that what kind of conversions is possible and what kind of conversions are not possible.
(Refer Slide Time: 03:26)

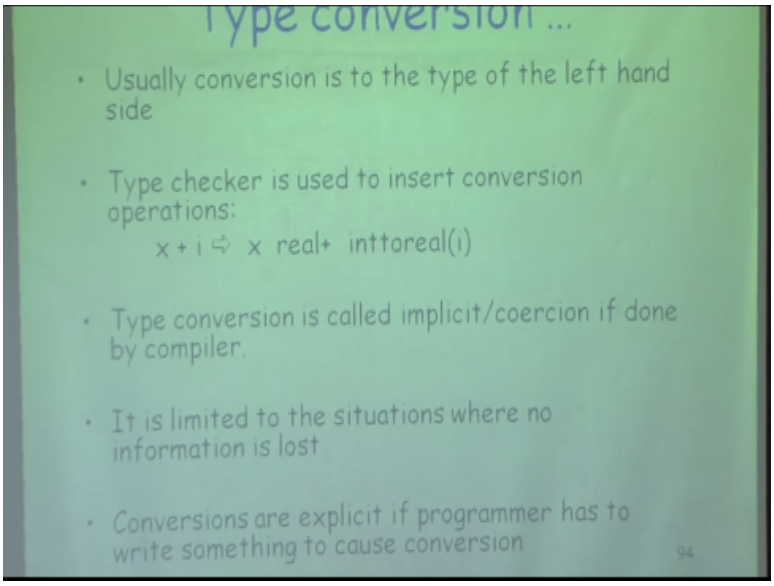


So type conversion normally then worship will not have a situation like this conversion usually happens from this time to this time but there is a small calculate and that is that if I am trying to do this fellowship it is possible that in certain conversions.

I am going to lose information so for example this s of type and this of type we are upload okay then I know that it is going to lead to truncation which is equal to information I do not want that okay implicit conversion also puts right now saying that I will not do a conversion if there is loss of information so usually we do the left hand side there is no loss of information now is it possible that I still want to type faster real into integer.

But in that case I say that I will not do it the compiler known that the programmer design so those functions like c and float and so right so those kind those are kind of explicit functions and where I will not will the compiler in any conversion what I let the user do this conversion okay so type checker is used to insert conversion of the operations like.

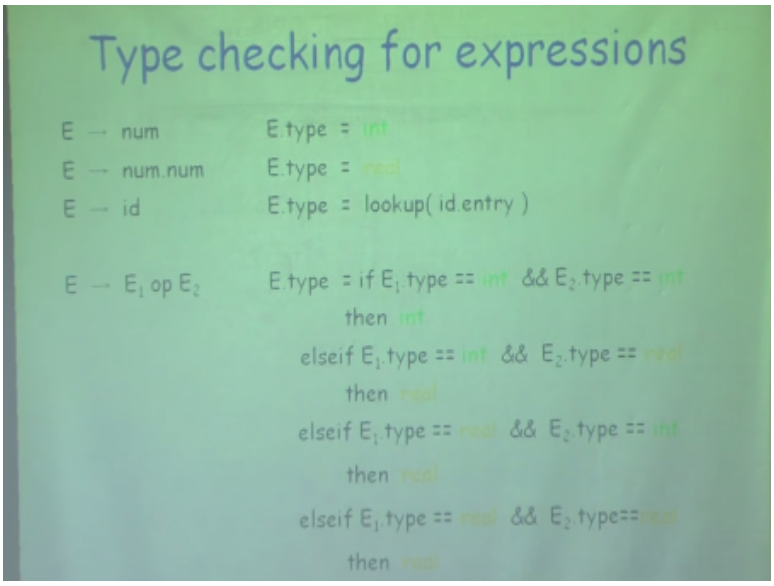
(Refer Slide Time: 04:48)



What I have certain here when say express I what it is converted into is S and plus than a real plus and I will became is to B to A okay and type conversion is also called implicit coercion with this is will became by compiler liking this case that is only limited to situations where is going know lose of this so this point is very important that that compiler does not want to take so the sort of possibilities for what to grammar.

Will having in there so compiler say I will only to conversion if it is only a matter of changing the presentation of change of information if you want expressions will change certain information going expression in the topic so do not depend upon the compiler to be okay so Conflicts are expressively a programmer as to write some people call the conversions so the example is this conversion Is to be done then we being to a programmer that will go to be now and explicit conversions will happen only where only the matter of changes the representation and lose of informations this right here and this is general principle which is form okay.

(Refer Slide Time: 05:52)



Now let us get deeper in the board so we are and suppose I am just worried about this kind of expressions so if I now have to capture basically now I have both integer and real types and I type I know will be there so of this I know that E type is intend for this e type is real and for this we type is nothing but just saying that to the table whatever is type get that interesting thing that happens is here and for the time.

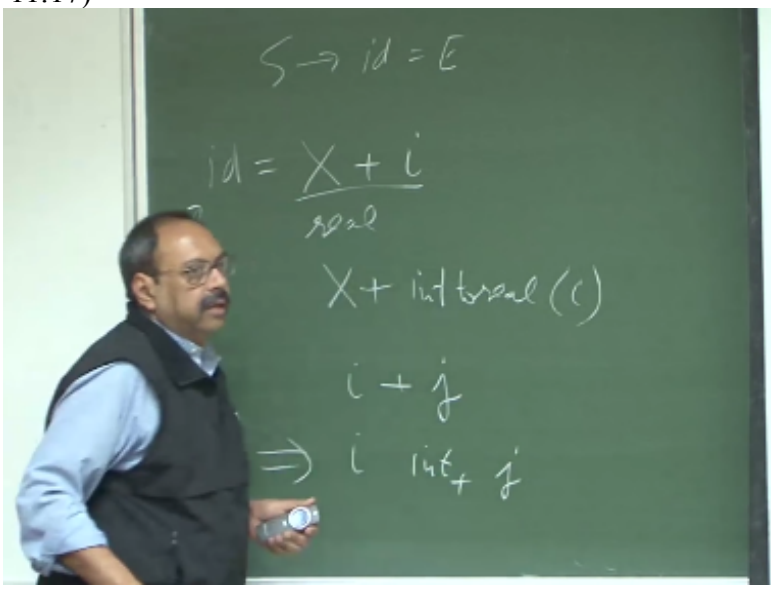
Being let us assume that I only have type looking between see as expanded to whatever define the language but assuming that this point of time language only permits with the string of integer of the real type and if I know E1 type and E2 type t want to find out what is the type expression for E okay so earlier when I wrote this we have not worry about type conversion what is the kind of code I wrote for this so earlier.

When I just had root might me single out then what was the code I wrote so there is no type conversion around then we said this both have a type integer when it positive right and if both are of type float then it was slow but now I will implicit of type conversion happen is I will convey missing this type of and how many combinations are possible both can be real so both possibilities I have so now.

You say that if both are in a then this type is going to be INT okay now if you buy a very short rotation assuming that only types which are possible of e1 and e2 are either integer or real so I just need to write else to this so right but I do not going to do that that the rule is rather than we just say else real I will now say let me enumerate all possibilities and it will become evident why I am and emulating all positives.

So now you see that if the first is not in the second is real then this becomes real and then the first is real and second is in nothing these colors are not very bright I will change it so wrote This

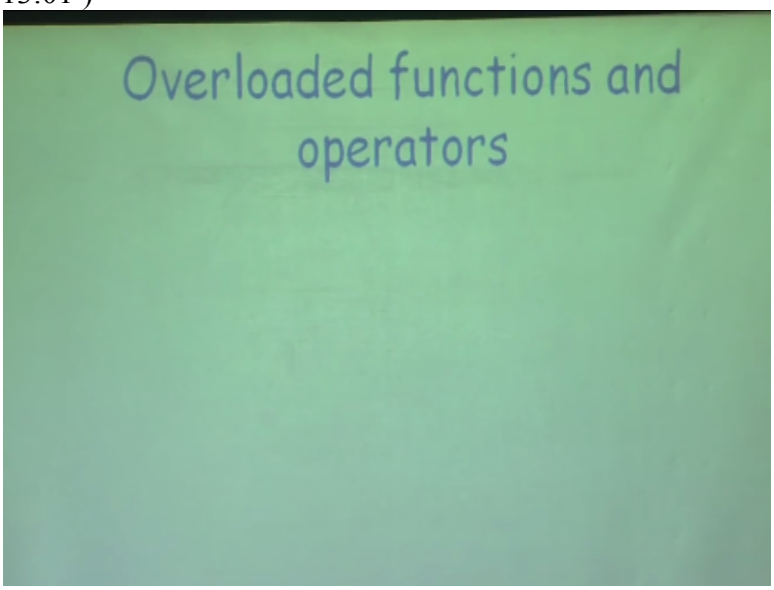
is finding that on display it is not very clearly and then is real if both are here then this is real now you can see that in these three cases.
Because outcome is real I would have just eliminated this as could have to this real right okay now I am doing this and why I am eliminating everything here any reason for particular reason does not give any additional information so let us just come in now look at this I mean even if it is not commutative or does it matter I am not even worried about that I am spending all to come from that.
Some point of time I am going to integrate my code generator to the time way and I integrated to the code generator my code generator with the type checking and this point I will say that the operand needs to be converted in to real and therefore I will insert a small piece of code here saying that to convert whatever the variable is into a real number and here I will have to insert the code so really what will happen.
Is that no type conversion here is the type is it no type conversion here but type is real and in this case is type of real but this or this have to be implicitly towards and then therefore all the options because result is some point of time this will get expanded and the code generated become part of this otherwise if I am just doing type checking then I do not need to do this then I will give the values and equation of the type right okay so this is not giving here an initial nodes now let us issue meaning to deal with this over loading a form okay.
(Refer Slide Time: 11:17)



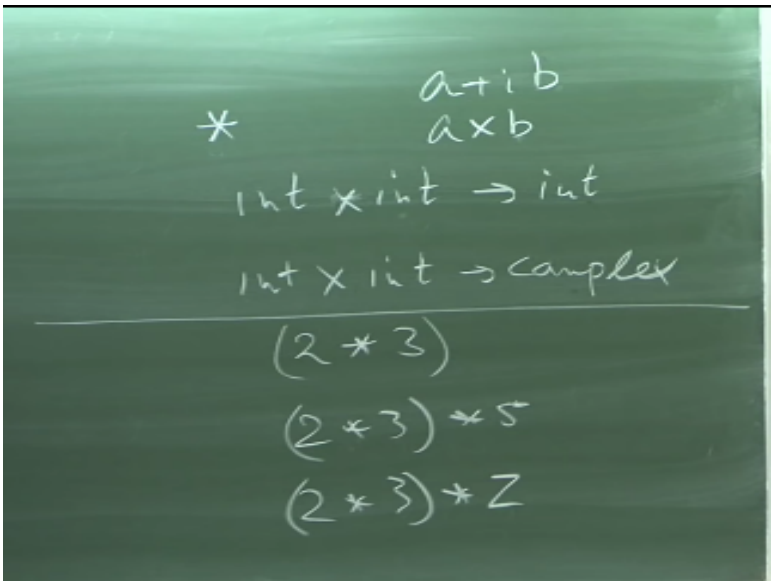
Now what is over loading functions is so let you go back okay and let me look at this okay and let me say that I started with X plus I but then when I was doing type checking I said this plus is

available now it is possible that if I am just saying I plus J where both I and J happen to me integers then my course or type checker will say this is I in p f + k okay.
So it is the same plus operator in one case this is a real plus in another case this will be okay so what I have done is over loaded in this particular function and depending upon the context I am finding out which is the real operator remember on machine this operator with this integer addition and real addition when I map on the different outputs okay I need to here type of type taking and find out what will all the operators.
Actually situation is far can this okay so there some operators is which by designation by mathematical definition by over rule okay so many time we will notice all are the operators are going to over loaded for example take multiplication okay I can multiply to integer I can multiply on combination of integers and the floating point numbers I can multiply are two floating point numbers and of course are going to be a okay both are that major what are the over loaded function so there is a situation where I say that not only overloading of operating system.

(Refer Slide Time:13:01)



Function okay and the language is like data which are important to over load function we can take standard function so I can take a standard function like bus and overload it with something else the operation in additional to the built in operation and request is not come in that if I now write an expression like this what is the definition of the function I associated this how do I figure the how so what we want to do here.
(Refer Slide Time: 14:09)



Is we want to see that if I have overloaded functions okay now I give two integers should I use this function or this function so let us now look at a situation that I have overloaded a function and let us say a table which is function star and these are the two prototypes are available.

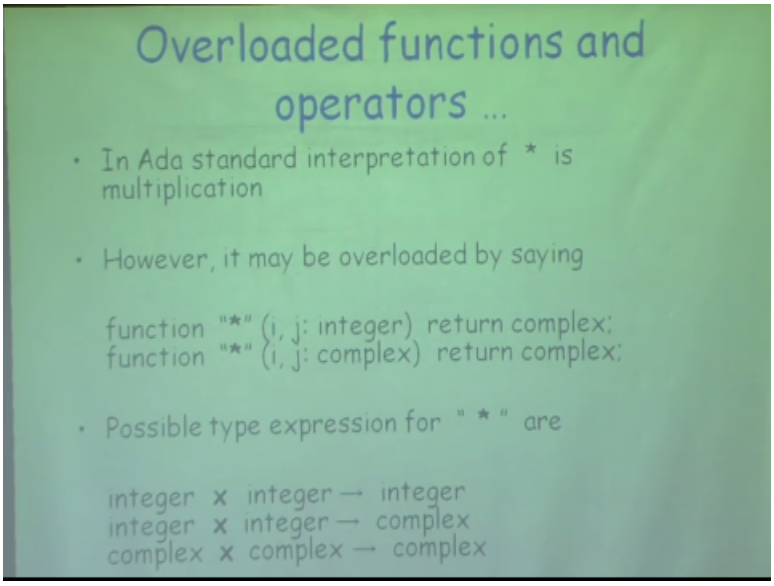
So suppose I take two integers here it is going to return a complex number to me it might be or this is just going to give me a multiplication of a B now in this case let the outcome is integer in this case given to the number to the complex number .

So suppose I say that I have so let us look at a situation like this that I have let us say 2 & 3 and I apply this function here question I want to ask is what will be the type of to stop me now just by looking at this sub expression I will not be able to find out what is the I suppose now I say that I put it in a context which where I say that two star three multiply by x let us say five in that is it I only know that what will be the definition of this particular function.

Which prototype will applied this one right because you know that there is no operation which is defined on integer complex and integer that is invalid yes but suppose I have something like this where it happens to be a complex number then which proto type will I apply then obviously because again there is no operation which is defined on integer and complex and for this to be valid this must be complex.

So I may have to have added another prototype so let us go through this example first and then see resolve issues here so overloaded symbols they have different meanings depending upon the context.

(Refer Slide Time: 17:13)



The context is the one so it is not this the left hand side the right one so therefore I use another term that is the context which determines and if addition is overloaded this is used for a period the complex matrices.

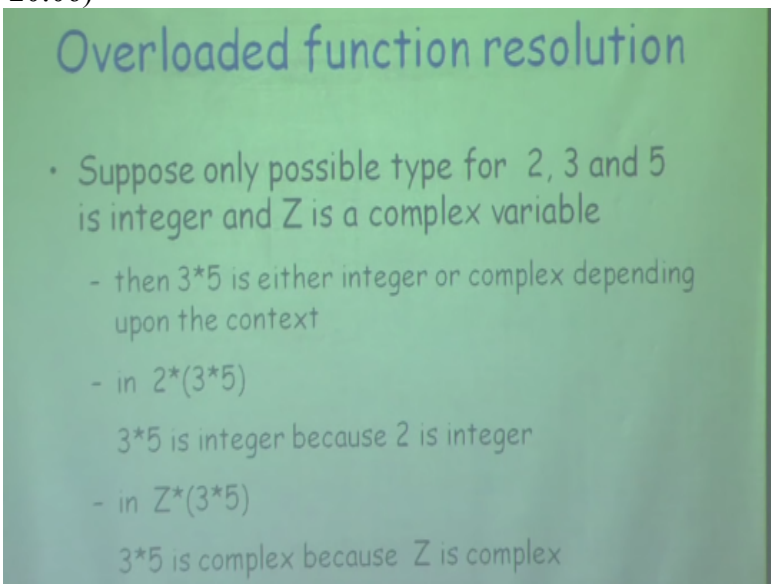
In fact yeah this is another interesting thing so there languages where you may have data fantasies like we like it here they became a integer floating point number and the matrix okay and multiplication of the matrix I do not have two factors A to b and semantic to do if I take a 2 factors and additional of two factors while I do saying all of these factor k and b the semantic of this operators is define that.

So data not that means the overloading which is used for array functions cal type numbers and so on okay if you look at the static is here over loaded when these context and the over loading is resolve can you find the using this okay this last point is very important that to beginning with when I just look at this okay I do not know what is the function which is being is slider we can start looking at the complex okay.

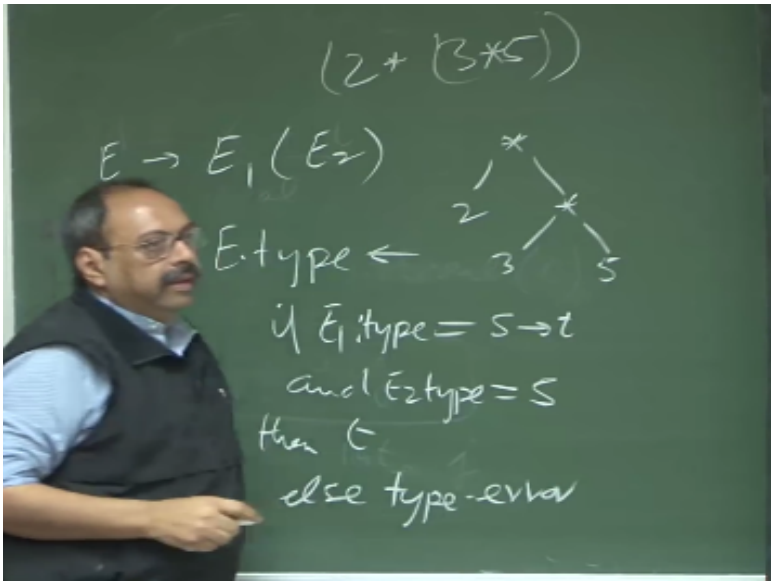
And I when I start looking at the context the type is always to keep on resolving okay of order this right it okay but if I find a unique meaning when I associate I said his is an add because this is an program should dominate it cannot be interpreted that be a form right the interpreter obviously compare you can but if I find a unique meaning then I associate that meaning with this and that is the function to replay to this going to do so let us look at a situation where we say that in and the standard interpretation of star is multiplication.

But it can be overloading is resolve type by having multiple type multiple arguments and return types so I can now have this and this is the kind of syntax it says that function star takes two integers and returns a complex and can also take two complex numbers and the complex number

so basically this is saying that I can do be a multiplication I can take two integers and return the complex number.
Which is of this form a plus IB or I can take two complex numbers and complex now if that happens and now if that happens state that has possible type expressions for the star of this integer of integers I can take integer complex after so alluring suppose I say two three and five these are envious and that is complex variable.
(Refer Slide Time: 20:06)



Then what will happen three Star five is either interior of complex depending upon the context context so if I say my context is like saying two star three star five okay then what is the context here context is that I am taking in integer here and therefore since I do not have a prototype to say that integer because integer x by complex just not there then I say that this is only going to be in two integer but if I have something like this then.
I know that this has to be complex will go be again then issue that terms is how do that in terms of writing the code so let me go back and remind you what is the code we had for function resolution and then we try to modify that.
(Refer Slide Time: 21:12)



So I was something wrote this form whereas e1 kind of syntax we had yeah and what is the code I wrote for this for type checking I wanted to find out each side and what was E type here so we said that it's even five is equal to a mapping complex from s to T and E 2 5 is s then P else type is s then t is type end right now what might happen here is that when I start applying C so I don't distinguish between function and operator.

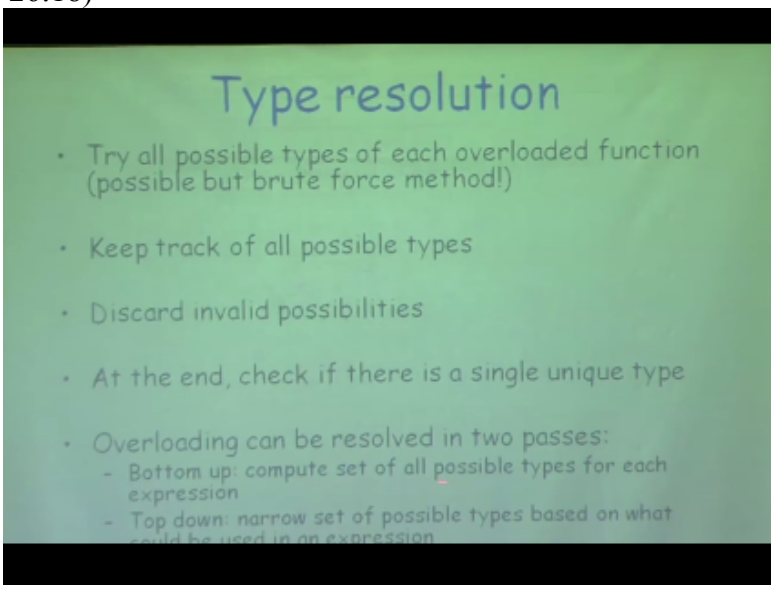
Just a matter of s that means you say now what may happen here is that when I say e type in time if I say that I am taking the first function which is 3 star five then what could be type what would be E type here this will be right there well it would be this type error or it will be P and if it is T then what will be values so I just look at only this selects testing to start 3 here or just take three star function here just go by this you got it.

So what are the changes I made you want you to make any change oh I will just lose against a manner so only change only I need to make here is rather than saying T if you accept thing is that even type remember that I know e to type right e to type in both cases is not me but even type will say that is going and in cross it also max on to complex so all I need to say here is this P T instead of a single value will be now.

Set that is the only came that really nothing else right so this is the kind of things will do and now once we start doing this at some point to time to need to have a unique value so remember that this will occur now in certain contexts okay so when I started by saying that I have this 3 star 5 at some point of time this 3 star 5 and that reduction happen with E this is occur certain context and that context is going to be 2 star 3 star 5 okay.

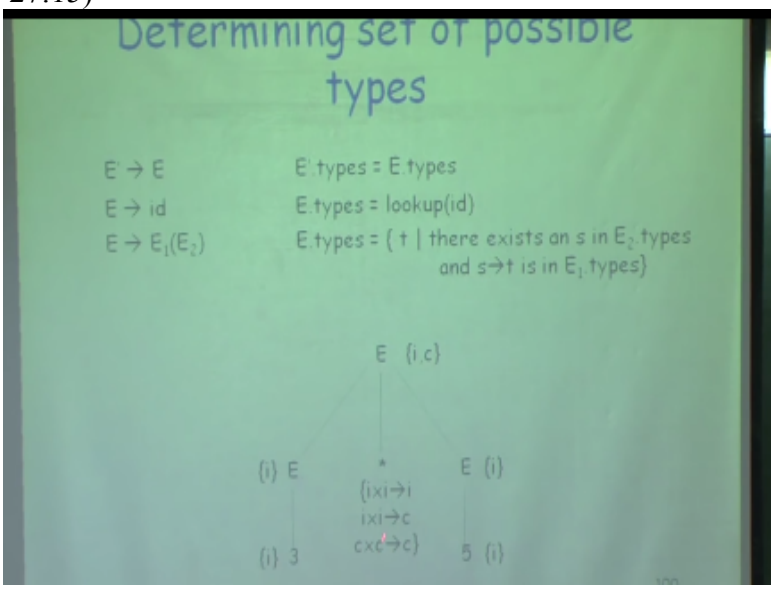
So again I will do this and say that when this resolution happens again I will have a set of types but at some point of time I will know the complete context and at that point of time I need to say

that only one type is possible invest from the star then the star hold the types so one way to look at this is that if I am building my expression tree and in expression tree if I have an expression tree like this so let us say I have an expression tree of this form.
Then I start finding out type so if I say what is type of 3 this is INT and this is INT and what is the type of this stock this could either e value here for could have either e either B here for the complex and then I say this is a ok I know that E is not defined here so this can only be INT so I can come down and discard this and add F is on okay.
So actually it is a to pass process when I first say that I start bottom up and start building all the types and find out the sets here rather than unique value which was possible when I did not have overloading and then once.
I have reached the top at that point for fine I must know that it must have a unique type and if it does not have a unique type that is a type error type but if it has a unique type then I use that information and start diverting down once again and say that certain types will be discarded and at every node I need to find a unique site because that is the only way I can generate code that is the only way I can call the functions right so let us look at an example of this okay so type resolution is that try all possible types of each overloaded function.
(Refer Slide Time: 26:18)

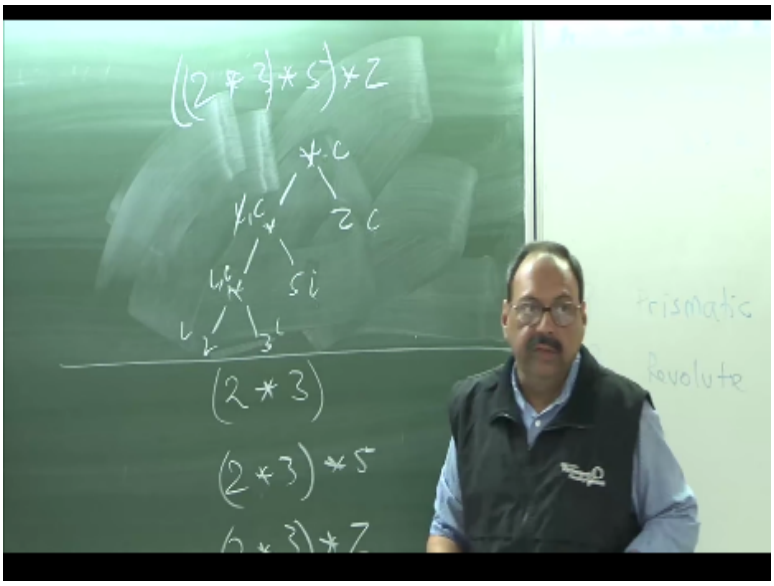


In some cases it may be possible but absolutely brute force method we are not interested in those kind of situations so keep track of all possible types and this is what I am going okay so when I start diverging down and I will start discarding this kind of information and then at the end check if there is a single unique type done.
If it is not then it is all finite code for this so overloading can be resolved in two passes so first pass is the bottom-up compute set of all possible types for each expression and pop down this

when I start narrowing down the set of possible types based on what could be used in an expression and that comes from the context right makes sense okay.
(Refer Slide Time: 27:15)

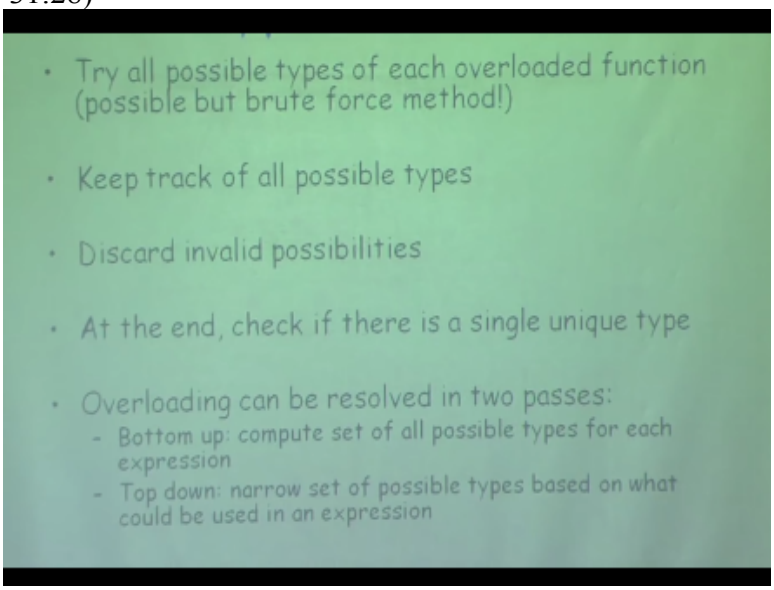


So this is precisely what we do that now my code will look something like this that when I say E types = E types I say that so let me first write the full code and then show you so basically what is happening here is that if I say 3 and 5 okay these are integers and this is star and if I have a type expression like this which says he goes to e star II then I know that types of start I cause I point why I cross I going to C and C cross C going to see these are the three over root types but since these are two eyes then I know that is possible but these two types are possible in my outcome is going to be I is.
In and then I will Traverse down once I have resolved for the unique type of this I will Traverse Town and discard one of those types so suppose I come down and say in this context in the context this occurs C is not possible then I say this is I and therefore this I cross I going to see and only the first type in use so basically what happens said is that if I look at these types I look up I D and now I say what is T type here this is a set T where we say that there exist sub s in E 2 times N SD e I bar types so I will not say this is just T but it could be have multiple range. So rather than having a single so first let me make an expression P for this and especially save that I am going to it has certain rules for Matsu right I have start from left to right it is very compact so suppose I were to write.
(Refer Slide Time: 29:20)



(2 * 3) *5)*Z this is what now if this is I and this is I then you know that it can be either I or see what about five is I now what are the types of cysts or C so if this is because if I think I stopped I see Ci combination will not be permitted but I know that if this is I here this is I can still get I can see there and this is type of this C okay but what I see as soon as you say this is because I and C are not permissible only C and C are permissible you will be start this. I already said that you multiple task do not worry about now although it can be done you got in a single parts okay but the pattern if we have two multiple process here.

(Refer Slide Time: 31:28)



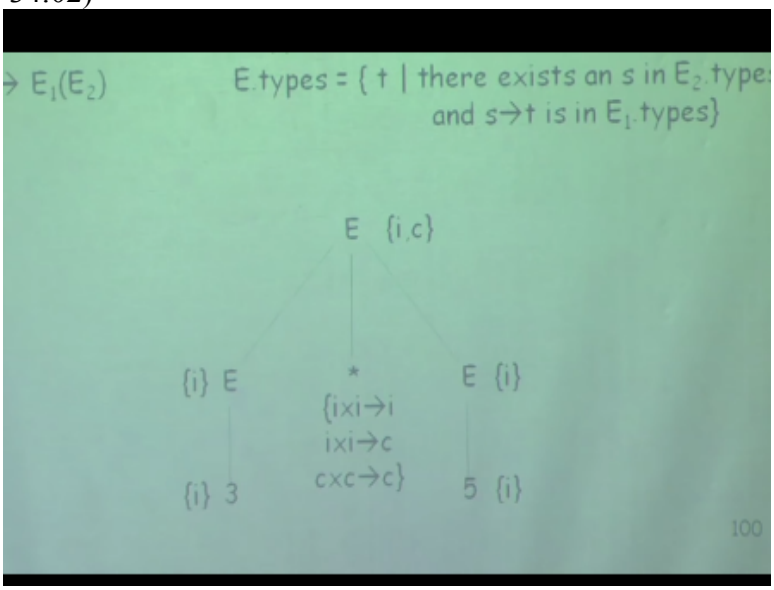
We have a bottom of pass you would here am going to type a single pass although I lot of generally and moving about the rules in the possible going a pass. But next a type to find out worry about whether let us say that even if I have to have multiple so look at this list that I say

that if this is C okay then is if I say I and I think then is not it clear that what I am trying to get here is that I am taking here interior value okay because remember that C and I e combination is not permitted.

I do not have such a profit on him so this C anyway is not enough okay then I know that what I am doing here is I am starting with India giving an India taking another India now giving accomplice and then taking too complex and moving on second cause they are in this one so what happens here is so look at it this way they cannot keep going up till initiative last completed at the root and the group I say that not white house a unique type if I go that is a type error if I do then I lose that information to come down and for each node.

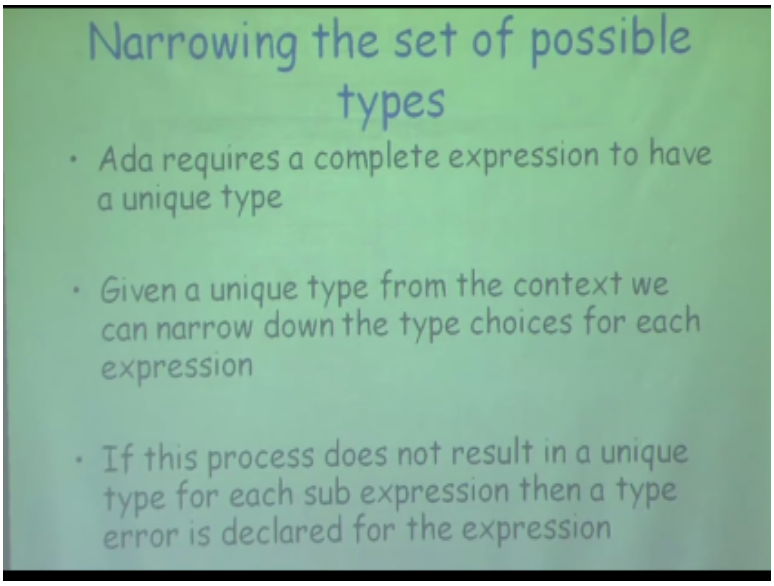
If I am able to do that that system what is possible that you can over motor function and I know that I never be able to resolve it and then I want to give a tighter because I do not know which one can we use and in any case if I can use more than one contract after implementing it look at it this way that if I cannot resolve it there is no way we are any more questions on this him so this is what we do here again that we keep on resolving these types.

(Refer Slide Time: 34:02)



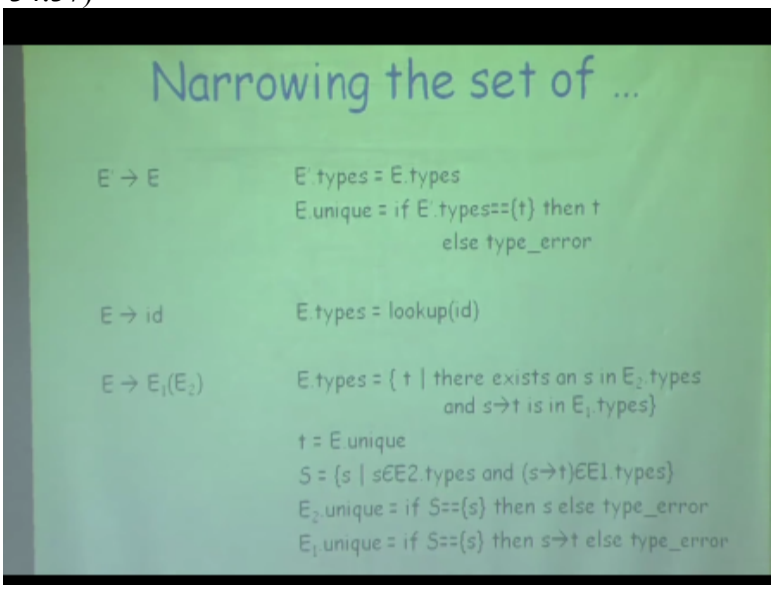
So first thing that happens is rather than having a single friend who I will now have a set of plans and how do I never have this set of values okay.

(Refer Slide Time: 34:12)



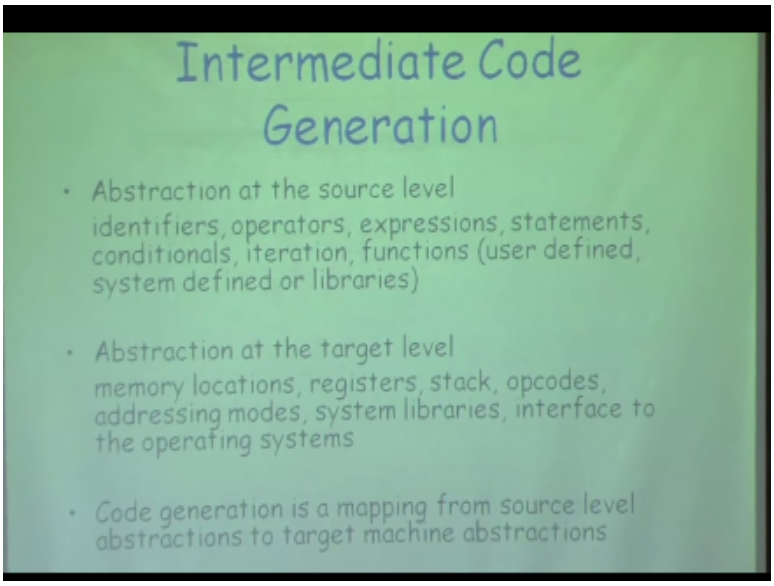
Had this requires now programming language says that a complete expression must have a unit that anything else is not completed so what we do is given a unique type from the context we narrow down the type of choices for each of the expressions which are below English and if this process does not result in a unique type for each sub expression. So how do I write now?

(Refer Slide Time: 34:37)



The narrowing down the types so basically let me again okay full code and see what happens here okay so what we have is when I say E I D says so look at who kind of moves actually I could have color coded it buts okay so when I say that E goes to I D so if I say he goes not that I know that E I is if E goes numb that he type is real and if E goes ID and when I pass this when I pass using this particular rule which says that I have a function then I say that II type.

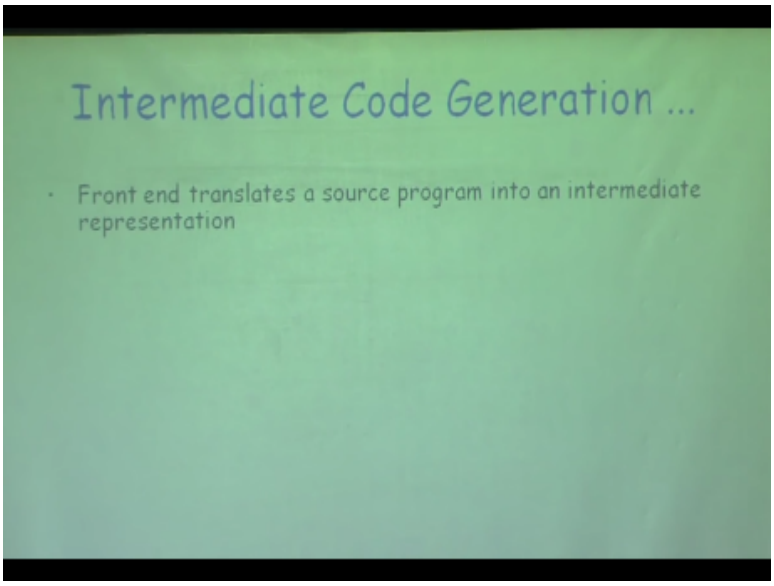
So this is the root we saw that it exists nothing but a stack of times so forget about this part do not even read these lines all the time now once I have constructed this then I say that now I say what is d prime time eat right time here when this is the star tool which says if I mostly eat prime types eat crime types this takes the value now this is the start symbol and not a check whether this site is unique or not.
Whether the cardinality of the set is 1 or it is not 1 but if it is 1 then we say that what is unique that is a new variable I defined and we say if I types is T then this is the otherwise lighter only one time it should have now if it gets resolved if it is not type error then we send our let us come down and that is where we start looking at last four letters we say that yes now I am trying to resolve type so here I now define a new variable which says what is my E unique must be having.
Because I have seen here because if each unique is not a single member then I gave a diaper what do I do now I say let me find out types of arguments so now I say that if S belongs to E to type and S going to P belongs the Eventide then if e to type is s this is the best I can this even type is s then this is s going to T otherwise this is the tiger so basically I will just understand this code what I am doing here is that first I have reached here which is vector menu you pointed out a good example.
Which you think this part you do this that when we reach here I check what is the type of this particular function this is unique so I am happy there can you narrow down earlier when I said I had a coma see here by saying that only I see is not a valid combination only see and see the combination and therefore is coming down but basically the trick remains the same that have a to pass have multiple passes to passes keep on building pipes till you reach the room and the proof if you find you do not have a unique type like.
A type error but if you find the truth you have a unique type then keep coming down and resolve at it for each of the descending notes and wherever you find that you are not able to otherwise yeah questions anything so therefore this actually plays a major role it says that this look always this is where I want to close description so let us move on to then what condition so what we want to do is now we want to start generating code and if you recall once again what we did in order to find the structure.
(Refer Slide Time: 40:57)



We had a front-end so let me just remind you of the structure we had we had a front E we had a back E and in fact as I said that we are going to have at least two phases and others will be we finally the machine code and of course we had an optional optimization phase which we will introduce only towards the end after doing the addition because this part is optional at least we must know this box and this box becomes sweet over compiler and if time permits so now what we want to do is we are here to talk about this place and this box is saying that I have done on the program and this is put that into some kind of code.

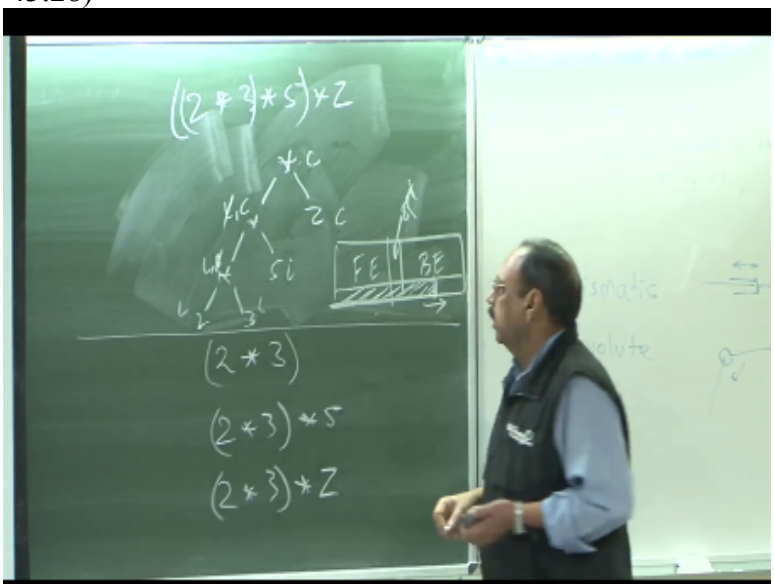
Which is closer to the machine so I am not straightaway generating machine code but I am saying I will do it in two steps and I want to move machine code generation digitalization obvious index of the machine look exactly like cotton SME labels time to go cameras okay so construction once again so this is again I have captured from the introduction in this point specification to that the construction we started with for identifier operator expression and so on and at target level one I am having our again memory locations register. Stack of words at this works and so on and for temptation is going to be backing from here okay and we want to take all the source level abstractions and in this I am m going to use,

(Refer Slide Time: 43:11)



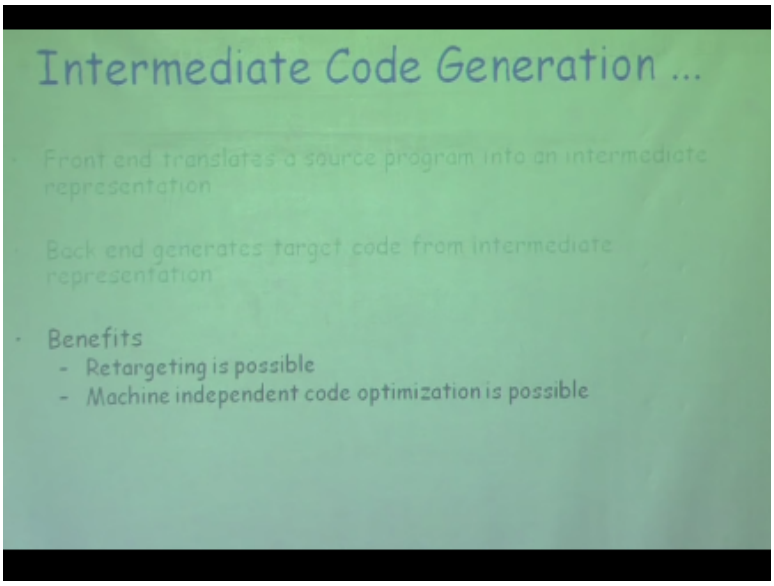
First the front intermediate for gestation forefront and translates source into so normally what happens is logically we may say that I have a front and in the hub sis but actually what happens is that you can implementation wise.

(Refer Slide Time: 43:28)



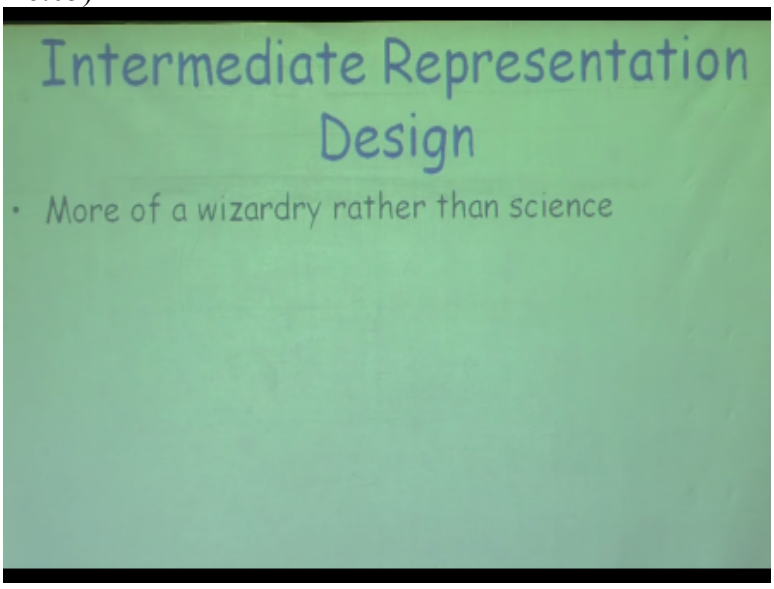
Fit for of this I am leaving the optimization out for the timing you can fit all of this so back and generates target port project is now the target machine from the immediate representations of my back-end released when I talk about my back-end this generates code,

(Refer Slide Time: 43:56)



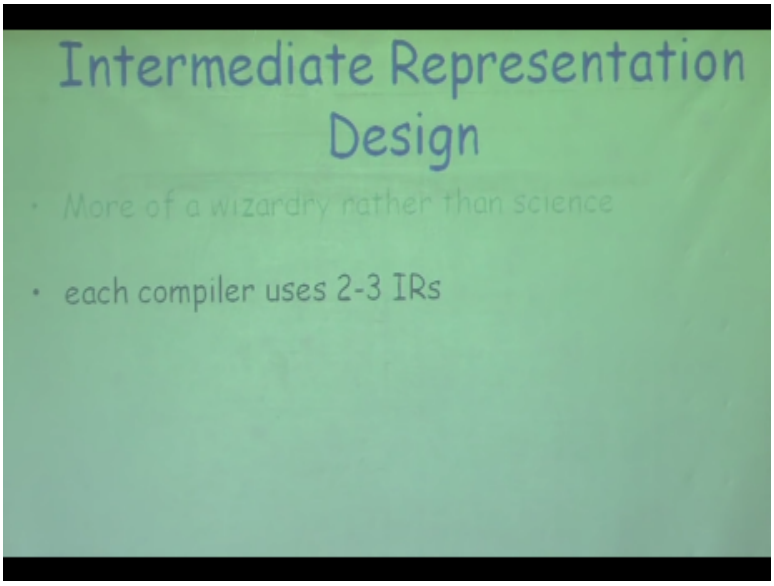
Starting from the together a single physical fits so the benefits of this and because this is talking about this box in the beginning which could then we target it on to any machine. So let us now start looking at what is the kind of intermediate representation I will be talking about so intermediate representation design becomes very important if you first because this is really going to the interface and it is not just interface between a machine code generator but it could be an interface to many machine code. Therefore this design has to be done very carefully okay now first thing that happens is,

(Refer Slide Time: 46:05)



On you have this going back in both they are able to achieve all the comets man it is so first thing is that is an art really,

(Refer Slide Time: 47:33)

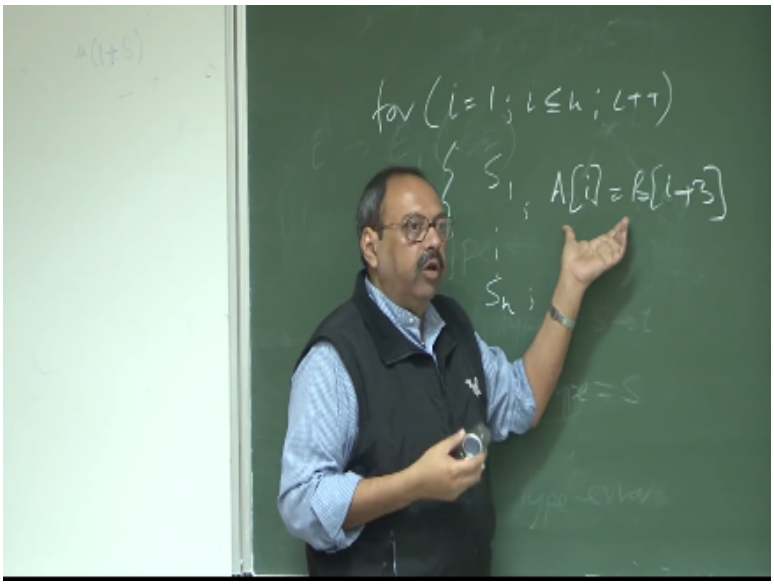


And each compiler uses multiple people I ask if I look at this I am I encountered X if X T which was not resolved we were also talking about parsley but as I said possibly swore off a rotation which is really confined that is not worry about in completion and then you have another ayah which was resorts invest. So this O X Y text is the one where all the operators have been resolved and then we look at otherwise now.

Either could be at multiple levels so we have one this is what we know as H R or high-level intermediate representation where I say that I want to preserve all the loop structures. And all the area now at this point to find it will not begin so let us look at water the other I R H we may have so another yard we may have is what we know as medium level finding where which has a range of features.

Which is basically close to the set of languages we have but still remains independent of the language and is also good for talking about even this kind of fire there is certain optimizations we may want to do and therefore it becomes really an odd let me say that let us take an example suppose I say that I have loop in the program now how do you translate normally loops you will have a sequence of perspectives so suppose I have something like this.

(Refer Slide Time: 49:29)



Or I have this now when I am doing this translation normally. What will happen is that I am going to convert this into a sequence of customs so I say that maybe ice stored in a variable which could be a register variable and then every time this loop has to be executed I will first check whether this is still within the bounds and then I will execute it I will come back I will implement it and I will keep on working now any time suppose I want to do optimization right here I want to recognize.

That there is something wrong because remember that I can convert this into a sequence of fester jumps but not all sequences of testing them and we can were returning. So and I keep on moving from abstraction translation I am moving some information so man I am losing information there are no point say can you go back and start reading something that information because so many times when I want to do optimization at this level and suppose here I have now an address which is of this form.

So this is a very simple form where I say a is a sin B and each for each element that once I convert this what is going to be the conversion here when I convert this into a language which is close to machine language. What is the kind of conversion you can think of load and store is a general technique I might wait to load something from memory and store it back know if you look at the motion that will happen is conversion.

That will happen is that I will say take the base address of B I know the base address of a and I will take the first element then I will increment I and if it takes four bytes to store an element then I will say that whatever is the base address I add 4 to that and keep on doing it for all the elements okay. But suppose I want to do certain optimizations in analysis on I okay now take for example.

Something like I + 3 okay and I want to do this analysis on these index values then I lost that information over okay so what I will try to do therefore is I will try to preserve HR and once I have done this okay then I will be able to do this so depending upon the kind of things I want to do our other want to keep H I R there or I want to go to MIR depending upon the kind of optimizations are so this is will stop our discussion today.

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 Lals
an IIT Kanpur Production