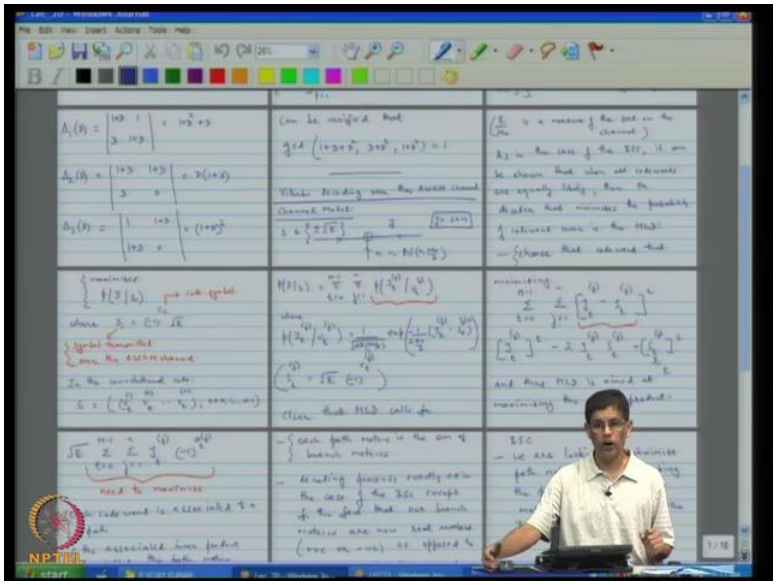
Error Correcting Codes
Prof. Dr. P. Vijay Kumar
Department of Electrical Communication Engineering
Indian Institute of Science, Bangalore

Lecture No. # 21
The Generalized Distributive Law

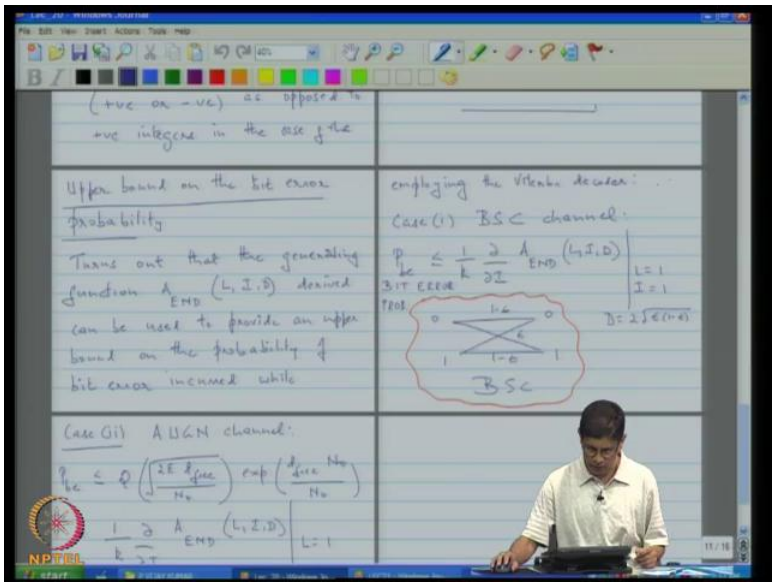
Good afternoon; so actually this will be our twenty first lecture, so that means we have crossed the midpoint. Since, courses oppose to is made up of forty lecture, so we roughly half way through the course almost exactly. So today, perhaps in keeping with that will actually start a new topic, but first as usual let just recap what we were doing last time, we will looking at...

(Refer Slide Time: 00:51)



So last time, we were finishing our discussion on convolutional codes, and I spoke of the condition needed to ensure that catastrophic error propagation does not take place for a general k byte convolutional code. And since this was in the nature of mode of cataloguing rather than actually, detailed derivation I skip the proofs. And then after that I showed you how you can decode the viterbi, excuse me how we can apply viterbi decoding, even when you want to decode convolutional code over an additive white gaussian channel. So we went through that.

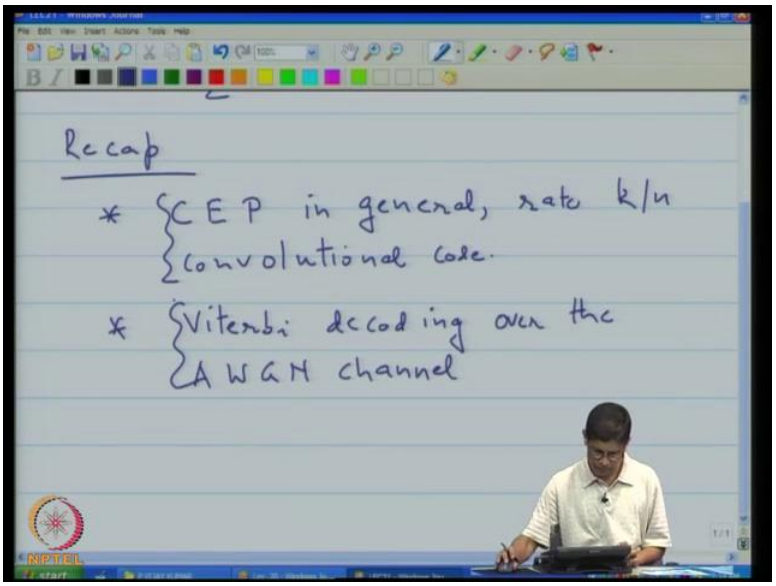
(Refer Slide Time: 01:37)



And then then I turned then I give a circular formula use actually upper bound bit error probability of the viterbi decoder. And this the two formula correspond respectively to the two different types of channels that we will be the encountering; so that is about where we were. And there was just one item there are thought, I should clarify here is that is one in this formula you have this in the formula, you have, so the AWGN channel you have this d free term that actually appears in the expression; and I just start I would spent in couple of minutes clarifying that.

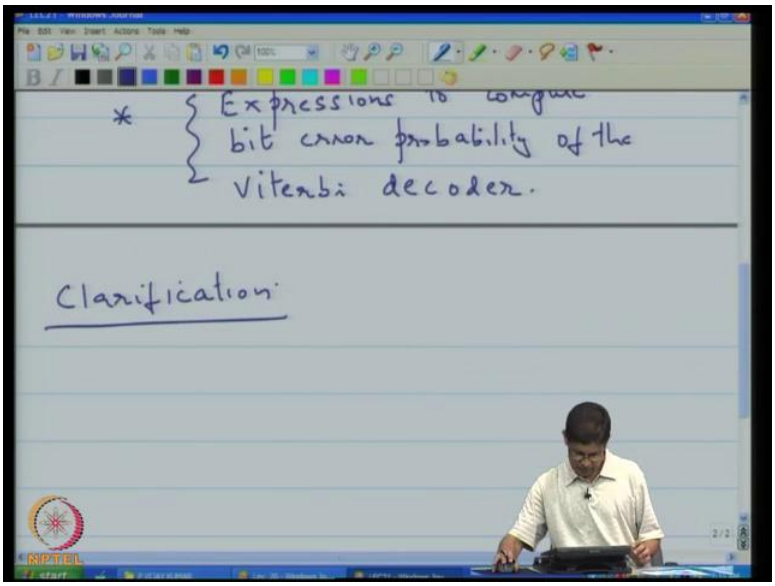
So little just put on title of our today's lecture, which is the generalized distributive law. Now could I am doing today is I am actually swiping switching tablet pc s. So I am tried to get use to the new writing surface, it is a little bit different from one I using from earlier. So I am trying to like slow until I get used to it.

(Refer Slide Time: 02:58)



Let us begin with recap. We talk about CEP catastrophic error propagation in general rate k by n convolutional code. And then I talked about viterbi decoding over the AWGN channel.

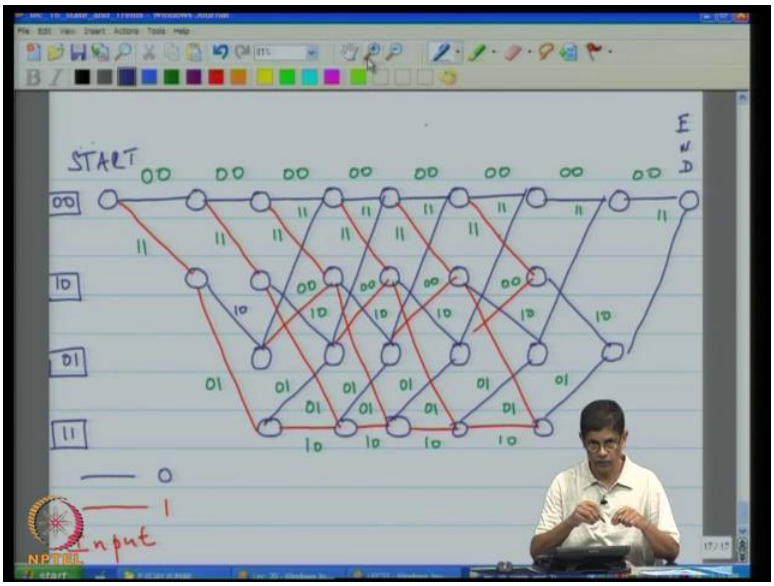
(Refer Slide Time: 04:00)



And then finely I gave you expressions to compute bit error probability of the viterbi decoder. And as I said earlier, before I begin just a clarification; so in this, in a previous expression, we

find in to this term d free, what is that code exactly that mean and I will just keep the explanation very brief.

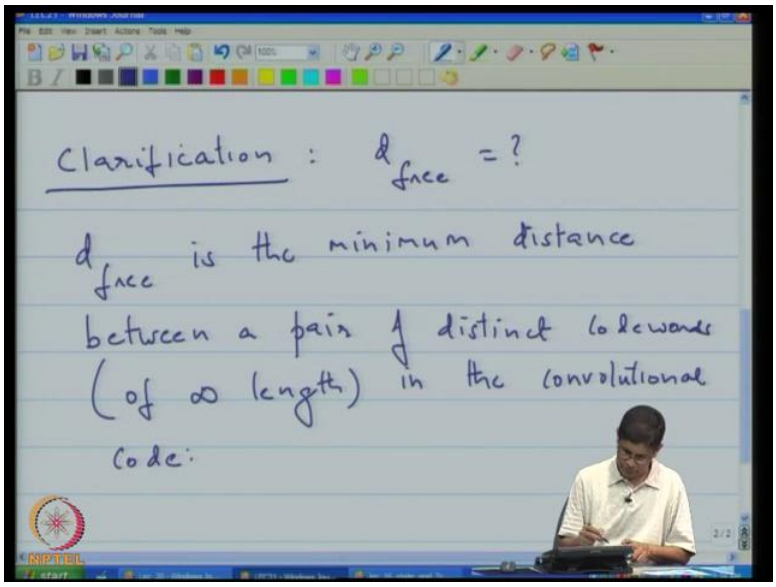
(Refer Slide Time: 05:33)



Now one way of explaining is that since as you know every path in the trellis corresponds to code word; and if this trellis series is of finite length that is by you look at the number of branches that you actually see; so one, two, three, four, five, six, seven, eight. But if theorem infinite number of branches, and that will be infinite trellis; and what we have talking about in d free is the minimum distance, minimum hamming distance between a pair of distinct code word in expanded trellis. And assume that imagine the minimum distance since it is a linear code, you can always try to find code word of minimum hamming weight. And as it you can imagine, if you just keep going among stay among 0 state, then you will get hamming weight 0.

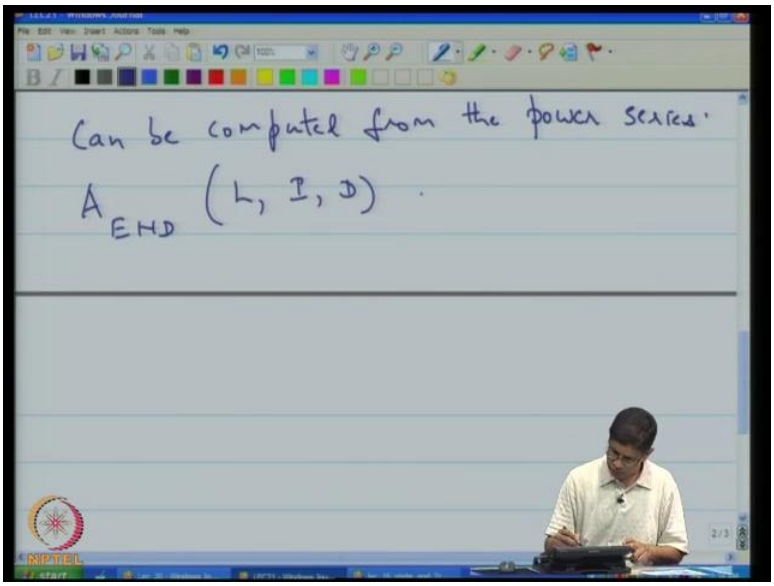
So any 0 code word must deviate from this all 0 path and as you can imagine it will try you get minimum distance you actually deviate little as possible. For example here deviation that gives you very quickly back to the all 0 path, and hamming weight that will actually encourage two plus one plus two, which is five. So this is suggest minimum hamming distance between a minimum pair of distinct code words, which is the minimum hamming weight of a non zero code word is actually five. So that is actually in the case and in fact the way in verify it is through the power series.

(Refer Slide Time: 07:11)



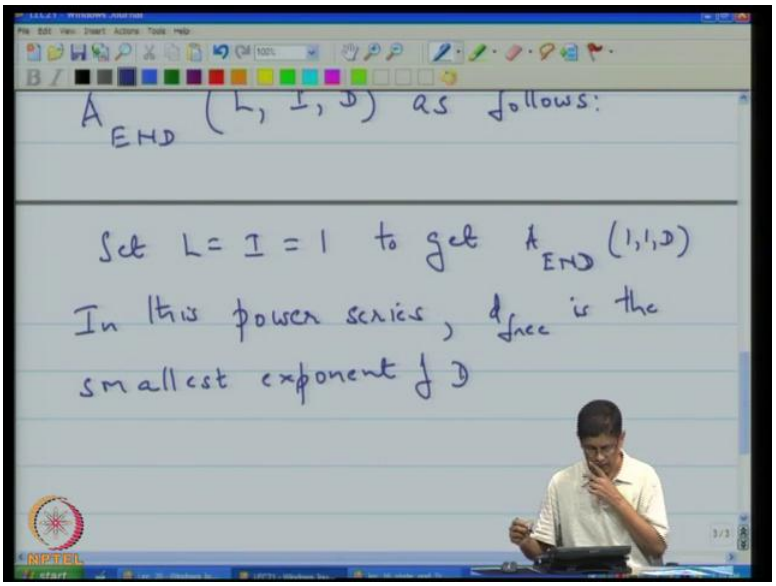
So let me just put that down in writing, d free is the minimum hamming distance between a pair of distinct code words, and in brackets of infinity length, in the convolutional code.

(Refer Slide Time: 08:15)



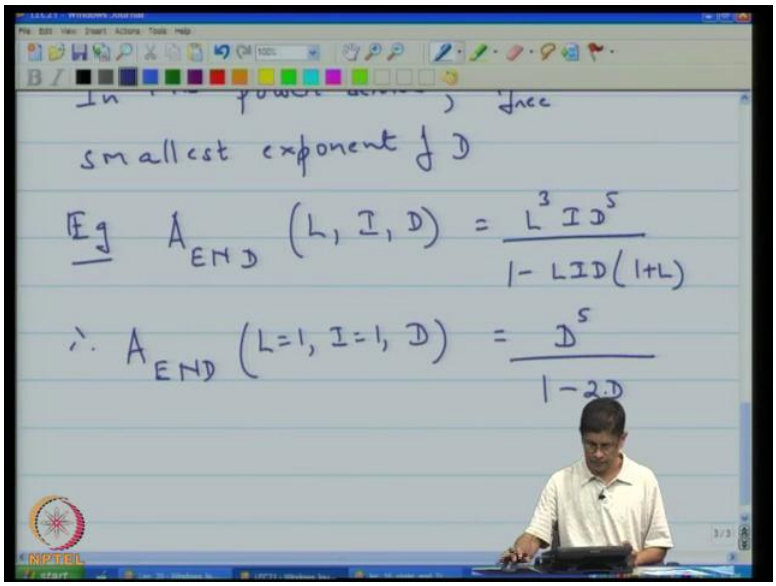
As a matter of fact is the way you have actually computed is from the power series, so it can be computed from the power series A End (L, I, D) as follows.

(Refer Slide Time: 08:53)



Set L equal to I equal to 1 to get, to get A End 1, 1, D. In this power series, d free is the smallest exponent of D. There appears with non-zero coefficient, but I guess that is understood.

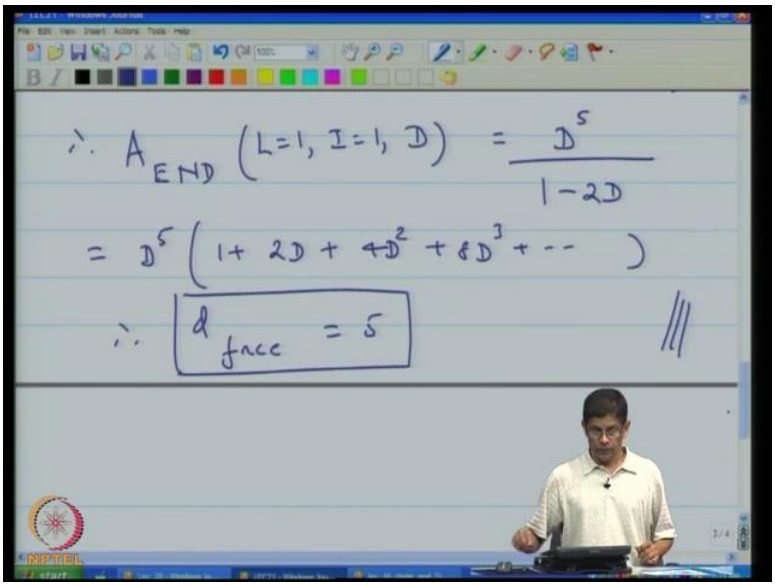
(Refer Slide Time: 09:50)



So we will look at example, in the example, this is our prototype convolutional code; and we would actually come across this before. So this was L cube I D to the 5 divided by 1 minus L I D

into 1 plus L. Therefore A End L equal to 1, I equal to 1, D is D to the 5 divided by 1 minus 2 D. Once you make the substitution, here you see L is 1, I is 1 and this is 2, so this 1 minus 2 D.

(Refer Slide Time: 11:00)



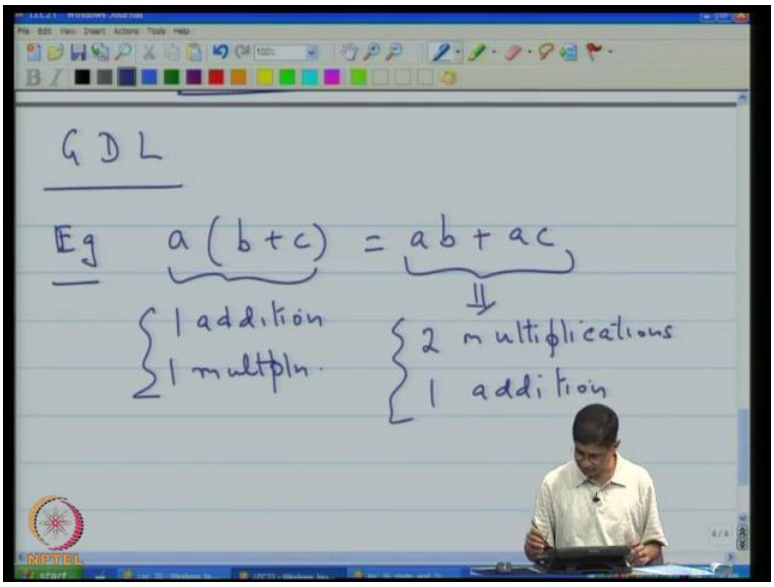
And as a power series, and this power series this expands to D to the 5 into 1 plus 2 D plus 4 D square plus 8 D cube and so on exactly as I explained before. So therefore follows the d free is equal to 5, and we already saw at the trellis this is just verifying the earlier observation. So, that we observe perhaps discussion on convolutional codes, and it was little shorter than we normally do, simply because we have other topics to cover. So today with that we will move on to discussing a new topic; and was given in put down this is the title of our lecture, the generalized distributive law and that might actually possible. What the trellis means is that you are going to apply the distributive law, but in a different setting, but in potentially different setting. And what is the relevant of distributive law decoding theory for this is all today will decoding of code; decoding is per amount from a practical implementation point of view.

From a structural point of view building the code with large hamming distance between distinct code words and large code words these are very desirable things. But on the other hand they affect performance there is true and other hand from practical point of view convolutional code you can actually code without occurring too much of complexity; so recent attention in coding theory has shifted towards codes that can be efficiently decoded. And now, underpinnings some

of the recent decoding algorithm is sometimes called message passing. It also goes by the name belief propagation; sometimes which also you also refer to this message passing that there actually using an, you are carrying out iterative decoding, these are all reference to more all as the same class or same type of decoding algorithm and of course everybody things with respect to area.

I am heavily influence by very nice article written by well known coding theory. I will give you that references give later. And I am going to present that point of view, which is slightly mathematical, but not well as you will see, and then I see about it is that develops theory in very nice way admittedly in idealistic setting. In practice this thing are not ideal, but it at least this will give you a reference point of say well this in some sense is the can be by a phase reference point. This is my starting point, I can use that understand try to understand how things are operated in practice.

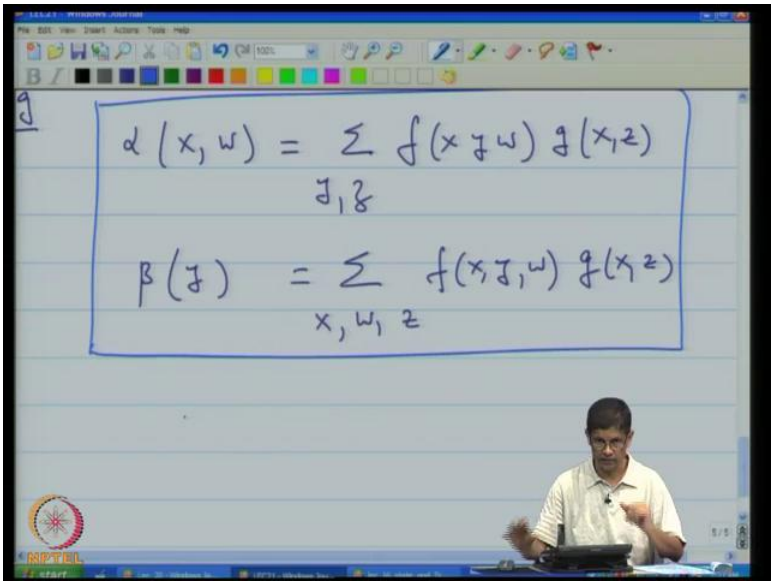
(Refer Slide Time: 14:20)



So let us discussed generalized distributive law, which is abbreviated simplify by GDL. And I just want to give you some examples begin with how the distributive law can actually be used to actually same computation. I mean, consider with simple computation. Supposing I take a times b plus c this is a times b plus a times c. Now if fewer to carry out the computation as this is organized on right hand side, you would actually incurred 2 multiplication and 1 addition. And

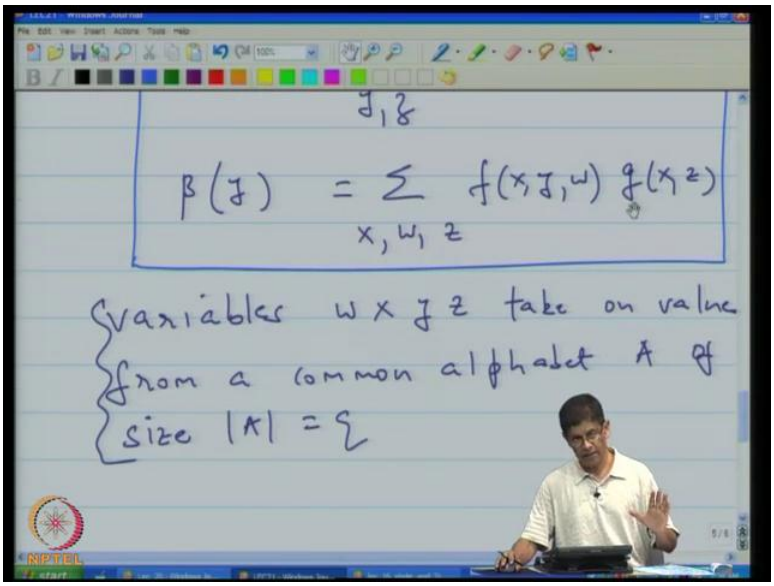
the other hand, if we were to use distributive law to rewrite it this computation this form, then is just 1 addition and 1 multiplication. In this very simple example, you have already say an one operation that may not see like lot but when the number of variables cross and in very large then this is, this becomes increasing more significant.

(Refer Slide Time: 16:00)



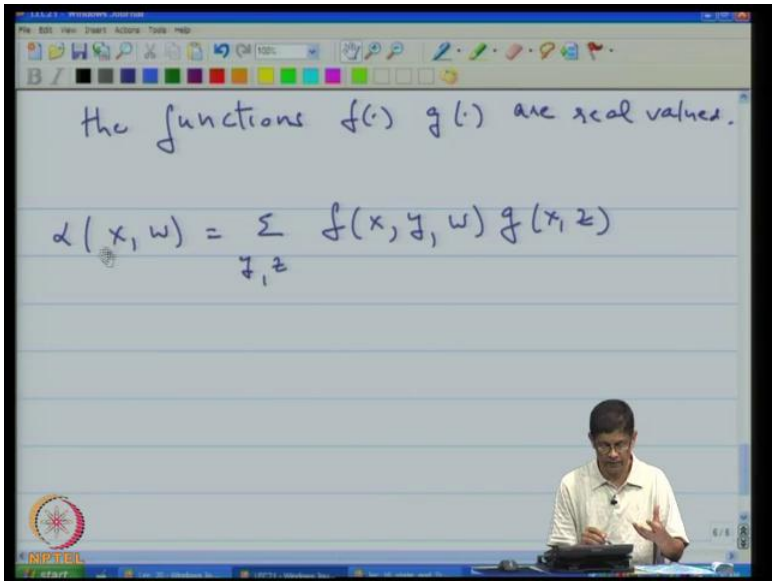
Let us take look another example and in this example I want to carry out pair of computations, which are alpha x, w, which is the sum over y, z of x y w and g of x, z. And the second computation is beta of y, which is the sum over x, w, z again of the same product. So, we want to evaluate both of these quantities; initially we will treat them separately that GDL will look at them together that we want to worry about that.

(Refer Slide Time: 17:44)



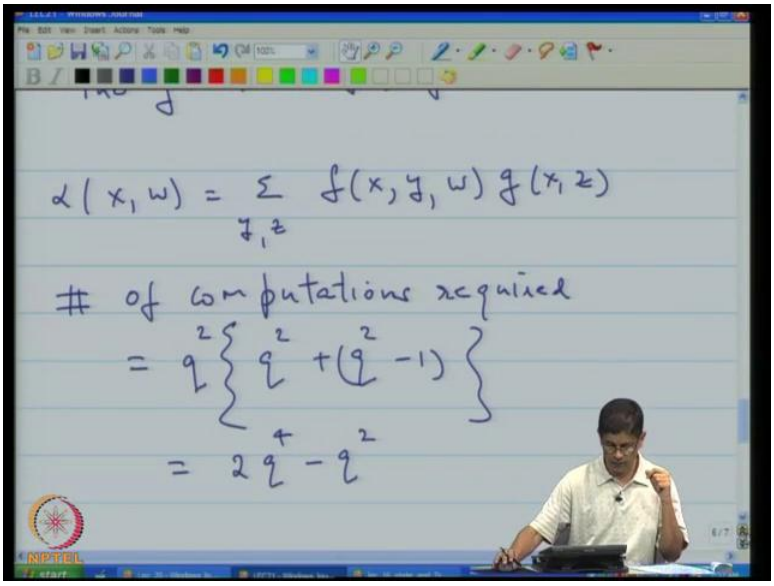
I should give you some additional information. So the variables the variables w, x, y and z take on values from a common alphabet A of size q. So that is the first point, and then so here the summation is over x ranging over all of a x value ranging A w ranging all of a anything with side then might asked what about this functions of g, so f the functions f and g are both real value.

(Refer Slide Time: 18:47)



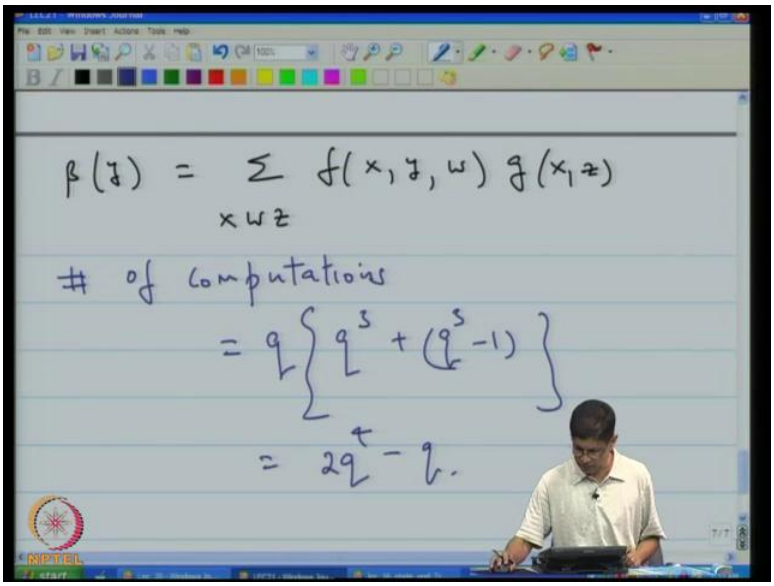
So f the functions f, g are real valued. We take on values in the sort of real numbers; and now the question is the question relevant to asked was how many computations actually need to compute alpha and beta. Let us look once again this computation normally, what I am do is class room setting how should asked class well tell me how the computation squared. And just based on personal experience with little bit careful counting, because one time is your over confidence and I will make mistakes. So, the way we will actually computation so will first look at this expression and say look I need to compute this function alpha for all possible value x and w now since x and w take on values from the set of size q there are q square possible value for this pair. For every value on x and w, then I look at how many competitions are need to actually evaluate expression here. And when I look this I actually that what I need to do is there are this summation over again q square term, because there are q possibilities for y, and q for z. In every one of these terms, there is multiplication involved. And then after multiplication carried out, since there are q square terms in the summation, I need q square minus 1 addition.

(Refer Slide Time: 21:10)



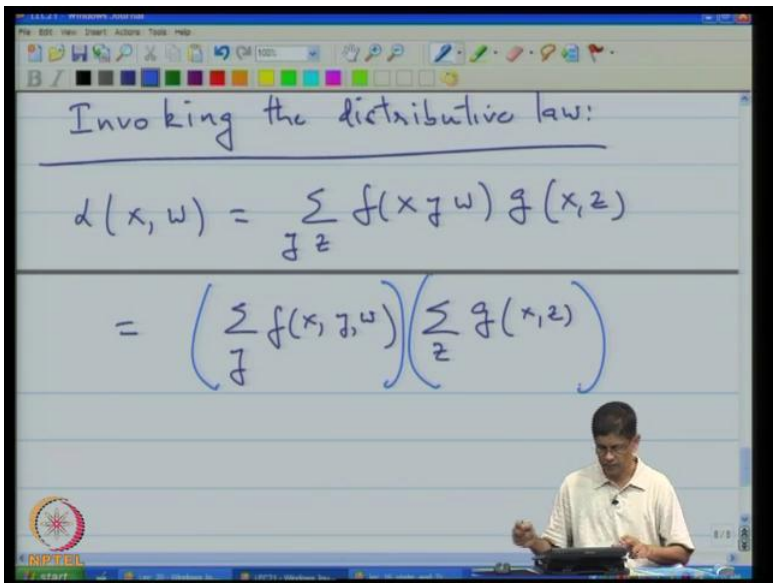
So the number of operations required to complete alpha, so the number of computations required is equal to q square times q square plus q square minus 1, this can here represents multiplication, this represents additions; so this is 2 q to the 4 minus q square. And usually it just a leading term that is people look at, so it is of order q to the 4; this number of computation to alpha.

(Refer Slide Time: 22:30)



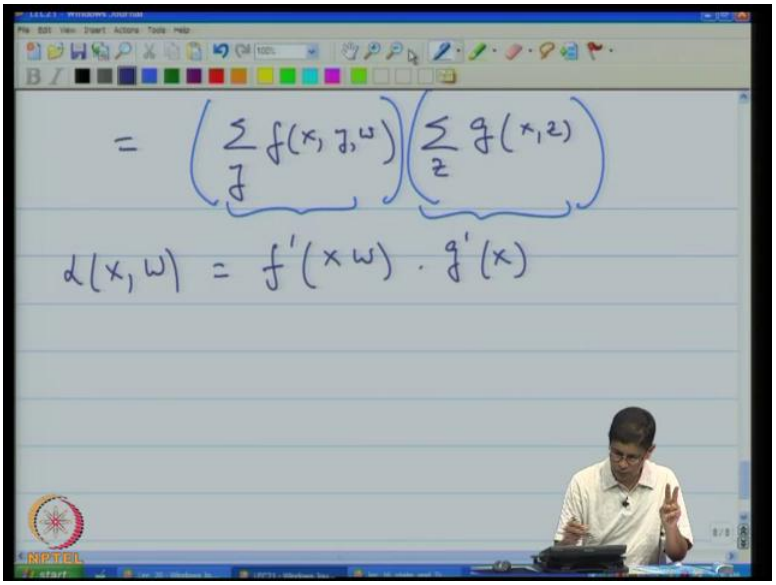
Let us do a similar calculation for beta, so for beta, so beta of y this is sum x w z f of x y w g of x comma z. And again the number of computations is given by q, because there are q values of y for which want to the computations times now in this case, there are q cubed summation terms; for each summation terms, we do one multiplication. So this will be this times q q cubed plus q cube minus 1, so this is 2 q to the 4 minus q. But you can actually compute this in a number of computations, by make in a use of distributive law.

(Refer Slide Time: 24:00)



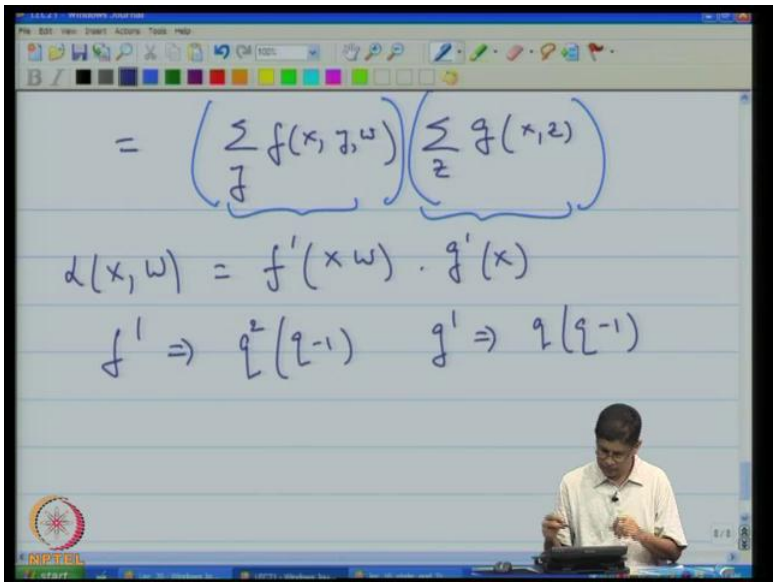
For example let me write invoking the distributive law, so alpha of x w. I can compute by actually summing taking summation on z inside in taking summation on y here, so I can actually compute this by computing this separately, by computing this terms separately that terms separately.

(Refer Slide Time: 25:18)



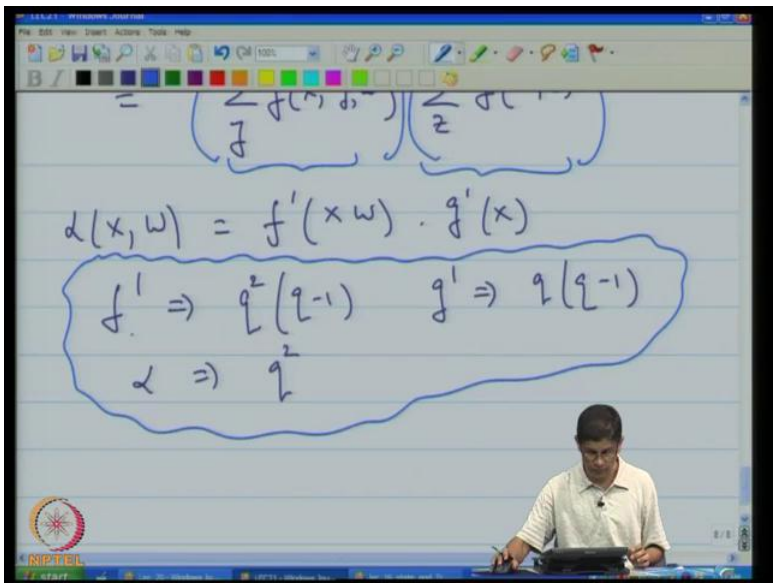
And if I once of computed this is going to end up as is some other function which you can call f prime of x, w, y, you can call this g prime of x, and now your computation is just alpha x, w is equal to this times this. And you can evaluate how much work is involved in computing this, so let us look at this again for much be careful in evaluating number of operation needed, the easiest I will find is to again in such cases is to look, I need to do as sub computation more exactly I need to two sub computations; one can actually calculate is f prime, then calculate g prime. So let us take care of those first.

(Refer Slide Time: 26:18)



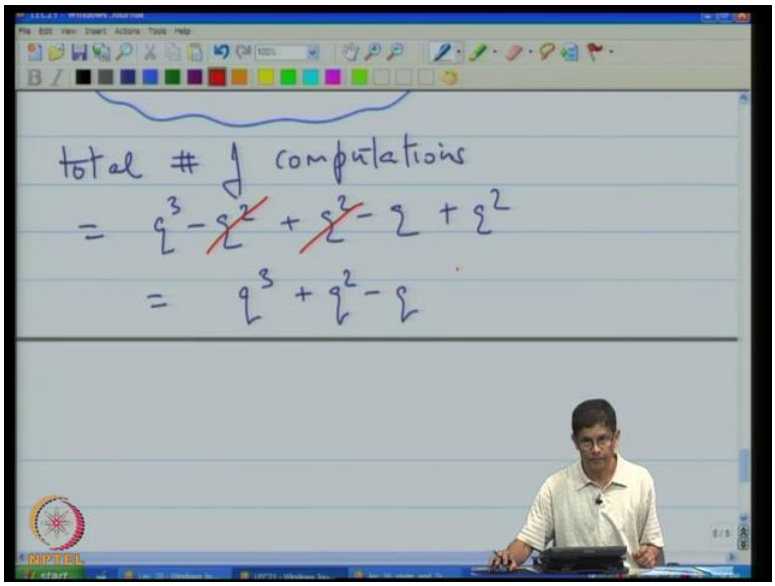
So as to compute a f prime requires, since there are q square values of these arguments requires q square; for each of the q square, I need to sum over q times so that q minus 1. And then to compute g prime, I need here I need to sum there are q values of x and for every value of x have a sum over q terms .So that is q into q minus 1, but I have not done here, because then and that compute f prime g prime. Now I need compute alpha; and there are q square values of x w, so which of the value of this; for every pair here, I need to do one multiplication.

(Refer Slide Time: 27:10)



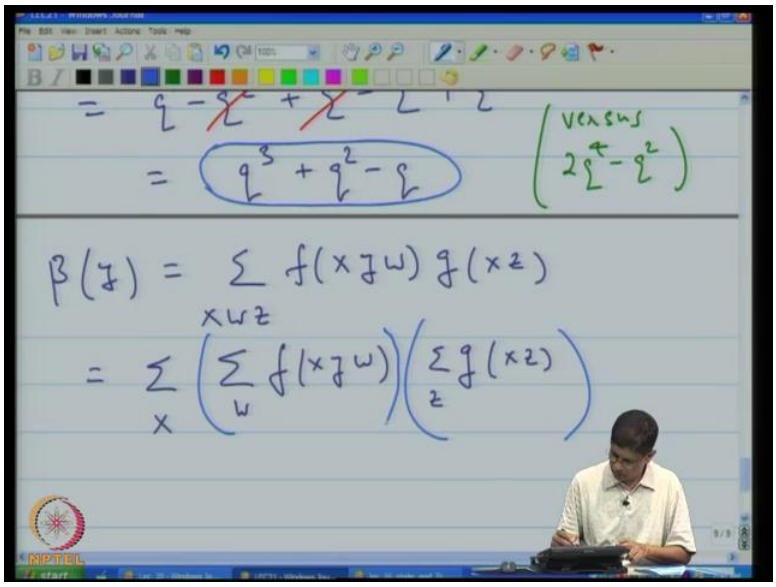
For alpha therefore I need in divisional q square operation. So if I put all of this together, so I need, so if I need all of these together, and add it up.

(Refer Slide Time: 27:30)



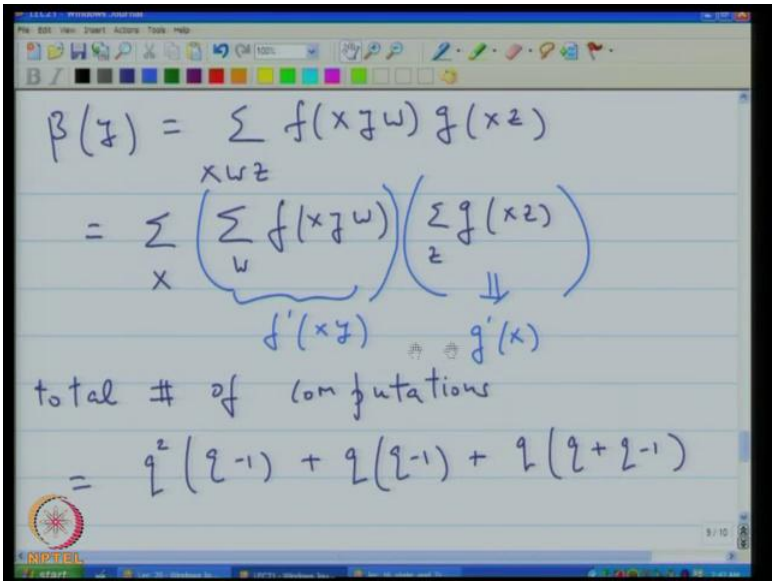
So the total number of computations is now equal to q cubed minus q square plus q square minus q plus q square, so I am just adding the three terms, and what I end up with these this is cancelling. So I am end up with in that with q cubed plus q square minus q, because these terms cancelled. Now this should be compared against so this result here should be compared against earlier count, which was 2 cube 4 minus q square, so we can see that will reduce in number of computations.

(Refer Slide Time: 28:48)



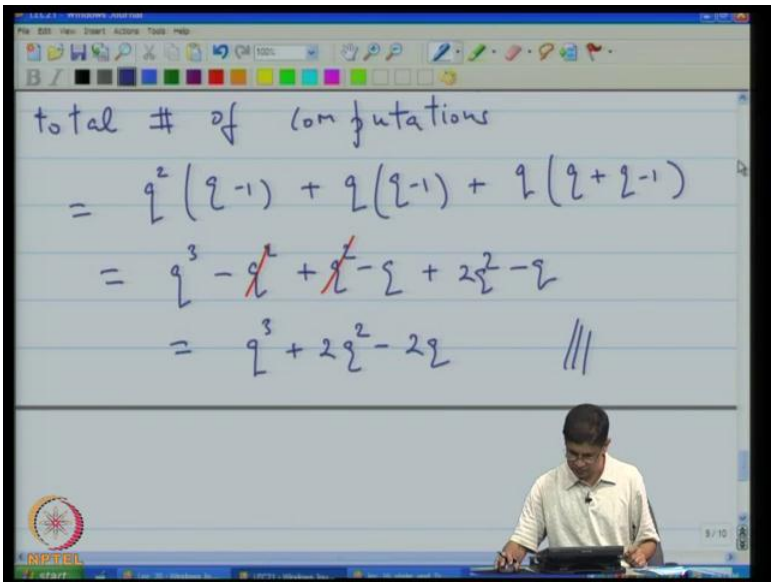
So I will write in brackets versus 2 q 4 minus q square value. So now (( )) beginning to see how the distributive law comes in to play. Now similarly, one can actually reduce the number of computations needed to compute beta. I will do that little bit quickly, since the idea should be clear, so beta of y x w z f of x y w g of x z. So we will sum over z here, then we will multiply by f of x y w, and then sum over w, and then I have to sum over x.

(Refer Slide Time: 30:28)



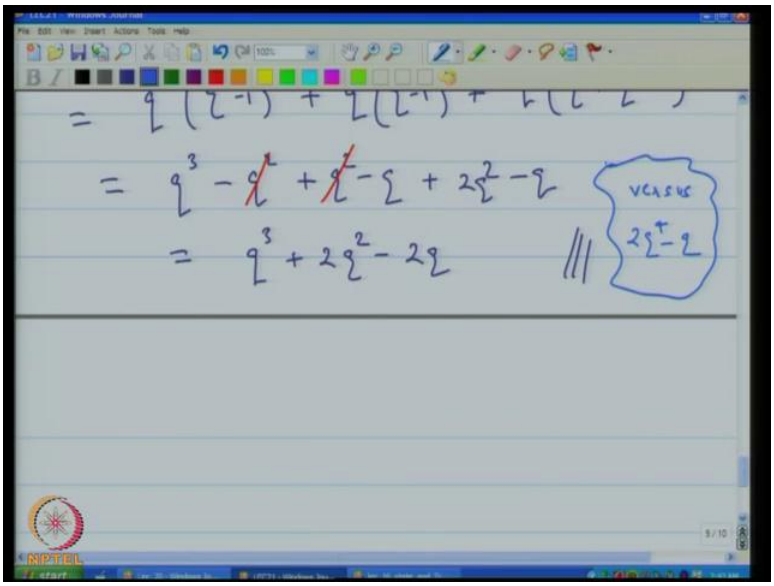
So let me call this, just like we did last time g prime of x, let me call this f prime of x y. And then we have summing over that to get rid of the dependence on x. So the total number of computations, so the total number of computations is equal to, first I am going to count the number of compute as f prime, then g prime then whole term, so this is q squared into q minus 1 plus q into q minus 1 plus q into q plus q minus 1. So basically this is the number of computations that you need to compute f prime, this g prime and once you have both f prime and g prime, this is the single summation which requires q, this many number of computation.

(Refer Slide Time: 32:21)



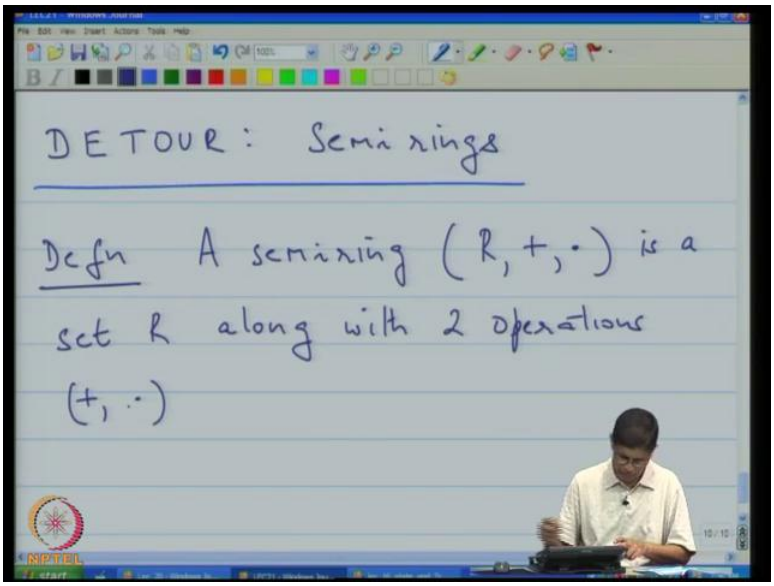
So the total is therefore q cubed thus the q cubed term. So q cube minus q squared plus q squared minus q plus 2 q squared minus q. We cancel this, and to get total of q cubed plus 2 q square minus 2 q. Again we achieved saving in a comparison with the number of computation that we needed on the last, because earlier we had estimated is 2 q 4 minus q.

(Refer Slide Time: 33:38)



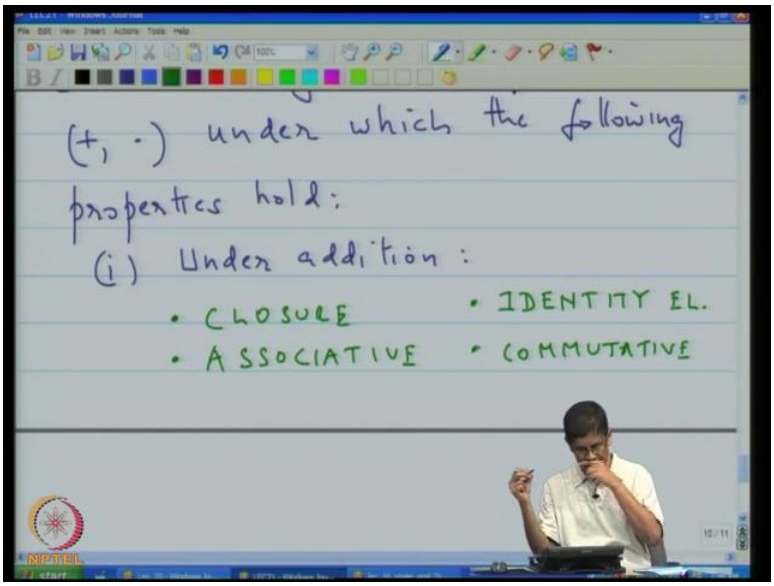
So this is versus 2 q 4 minus q. Now in a in this particular instances, it was fairly easy to see how one could actually organize the computation taking advantage of the distributive law. But if there are very number of variables, then it becomes very difficult to do that and it is not practical. So what the GDL tell you to do is, it tells you that sometimes you can actually organize them by making use of graphs. What you do is, you associate graphs to this computations. And then you use on algorithm on the graph, which automatically carries out the distributive law. So that is what I am going to do next, however just I choose the case with development of coding theory we will spend little bit of time getting little bit of mathematics. How do the way, because that this whole algorithm key point in this whole entire algorithm is that the distributive law can be applied in more other setting other than the one that we are (( )), and that setting in terms of to be setting of algebraic structures which are known as semi rings.

(Refer Slide Time: 35:27)



So let us talk about, so we will take a slight detour and talk about semi rings, so what is the semi rings? So definition a semi ring R plus dot is a set R along with two operations plus and dot.

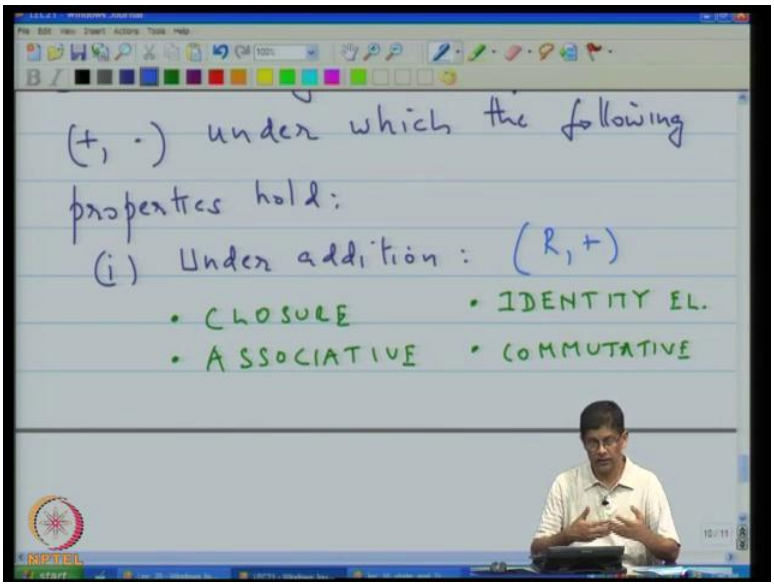
(Refer Slide Time: 36:30)



Under which the following properties hold. And these properties are one we require that under addition we must have closure, we must have closure, we must have that operation is associative,

they must be an identity element, and then the operation must be commutative, so we need this four.

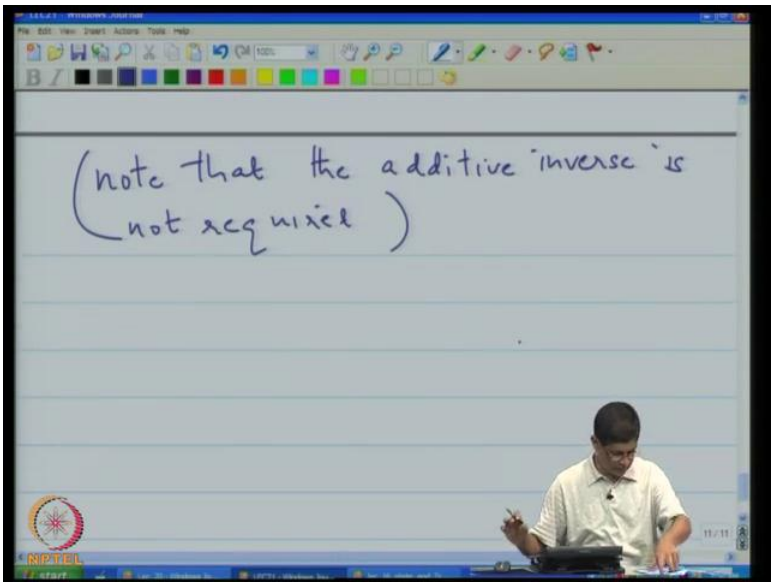
(Refer Slide Time: 38:18)



Now what is closuring? Closure means that we have two elements then it is back in the set. So this applies the addition and R. So that means if you think about the structure R under plus, then this structure must be have the property of closure. That is if you add to the elements in semi rings, you get back something in semi ring, then the way go to elements together violate adding should not make a difference. There is the identity element and we will call that zero and additional should be commutated, there is a plus b is the sign as b plus a.

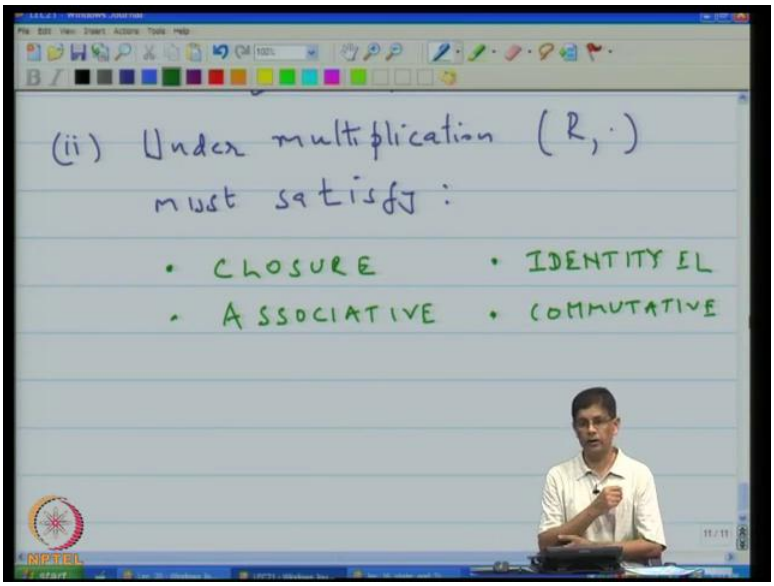
And I am spelling this out, because we already encountered these terms in our earlier discussion on rings. Now you look at this and say this really looks familiar and it does, in fact this when we were actually talking about groups. We encountered this same axiom; there is just one difference and that differences that if you look here done. When we defined groups there was an additional property that is required, and that property was the property of the additive inverse. So except for the fact that is semi ring need not possess the additive inverse, under addition it has all the other remaining axioms that going to making group, an abelian group abelian or commutative group.

(Refer Slide Time: 40:00)



So we just make a note of that note that. Note that the additive inverse is not required, it could be there, but it is not required. That is it for an additional course.

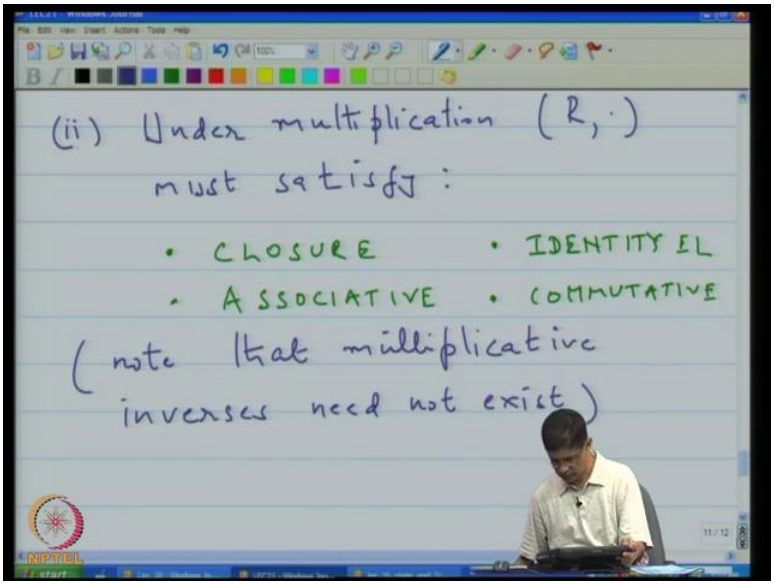
(Refer Slide Time: 40:35)



Then we look at example shortly, secondly under multiplication that is R must satisfy the properties of again. You must have the property of closure under multiplication, multiplication must be associative, they must be an identity element and multiplication must be commutative.

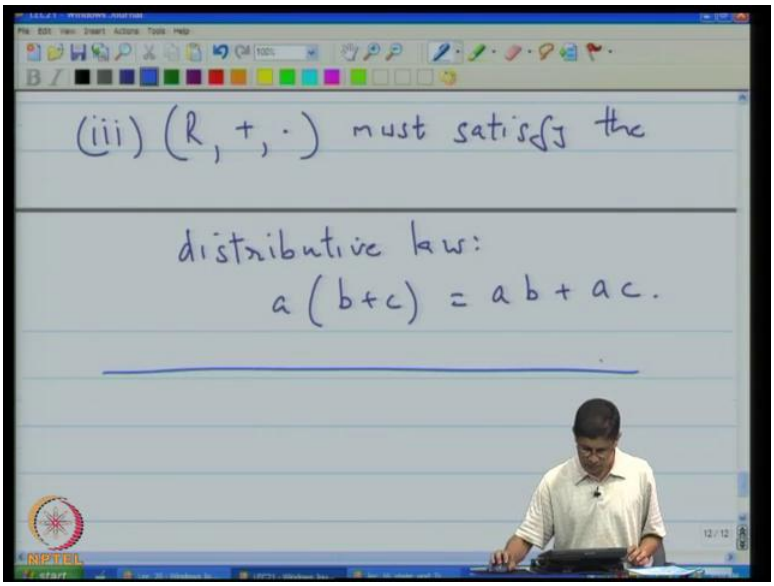
Again this look at say where again it looks like a inverse is not required that is true even on multiplication we do not requiring inverse. But that is perhaps not a surprising as in earlier case, because after all when we studied rings, rings were object which commonly did not contain multiplicative inverse. So we have encountered this kind of situation before.

(Refer Slide Time: 42:14)



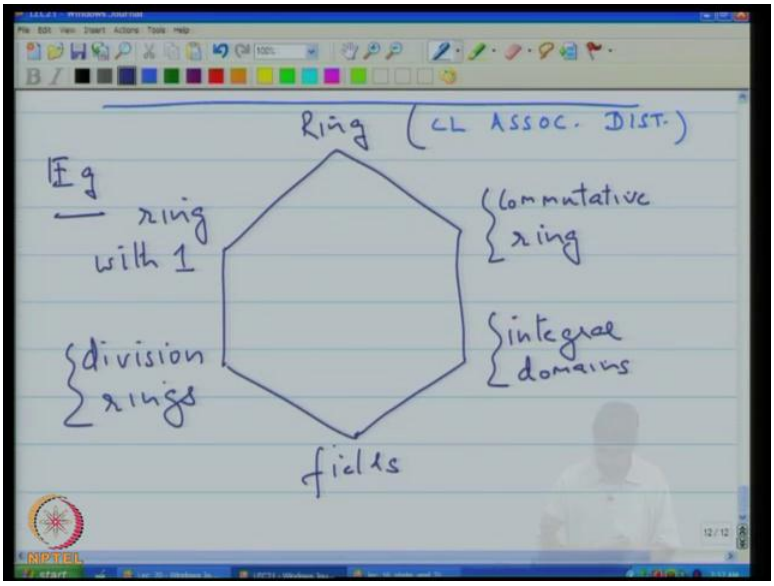
So again note that note that multiplication is not multiplicative inverses need not exist. And then, so these properties are very similar to what would you have in the case of ring, by the way identity element for addition is 0, for the multiplication it is called 1. And another think that I should point out is that semi ring really consists of set in two operations in the definitions, we found in convenient to give them, give these operations in name, so we said the one operation is addition. And other one is multiplication. But in reality, it could be different set, different pair of operations. So for example, we will see semi ring which plus and multiplication are replaced by max and multiplication or min and sum, min and addition where max is maximum and min is minimum, and we both maximum and minimum as operational two elements. So the formal, so they were we listed properties under addition and multiplication and of course since we have labeled these generalized distributive law not surprisingly various requirement that multiplication distribute over addition.

(Refer Slide Time: 44:31)



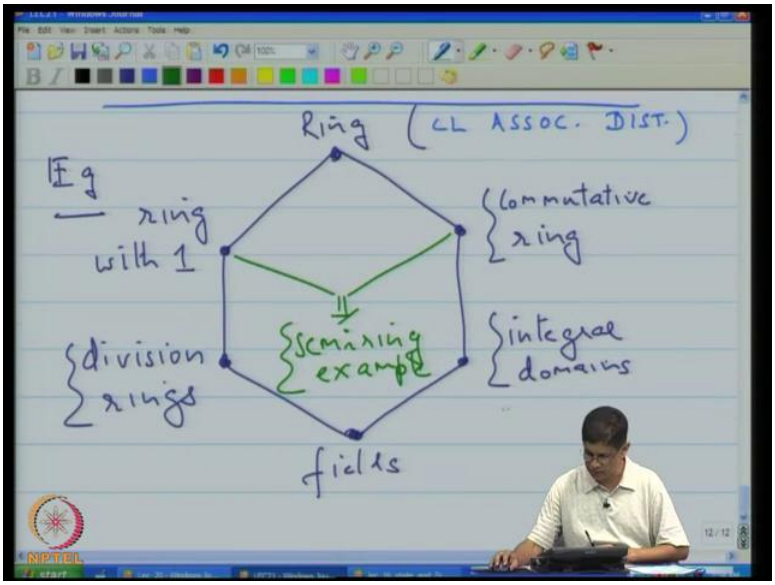
So that is our third property, namely that R and plus and dot must satisfy the distributive law, namely that a times b plus c is a b plus a c. So these are the requirements and semi ring. So again to recap, you need that all the axioms to group with exception auditing inverse are required, and then on the multiplication. Again you required closure, associatively, identity element, commutatively do not require inverse, but we do the required that the distributive law.

(Refer Slide Time: 45:40)



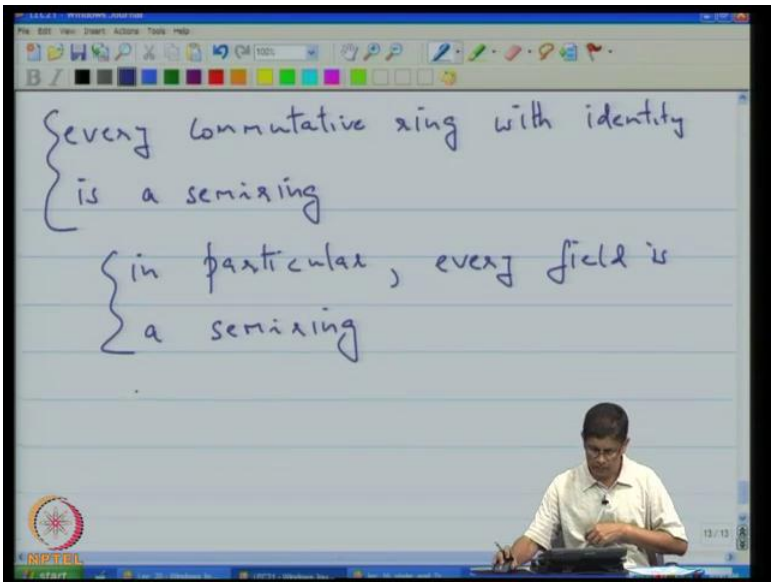
Now so, for this example go, the first obvious common; and if you go back and discussion on ring, remember we drew a figure like this hexagon, in which we placed at the top rings, general ring where we have closure, you have closure, you have associatively, and you have the distributive property. And then we have decided, we have rings with identity, rings with multiplicand identity. We had division rings, we had commutative rings, we had integral domains, and then at the bottom we had fields. Now semi rings can be found within this clause, because every ring has all the properties I mean in a ring and additional since billing groups has all the properties plus additive inverse, which is not required, but is still there. And the multiplication you have all these properties provided your ring is a commutative ring with identity.

(Refer Slide Time: 48:00)



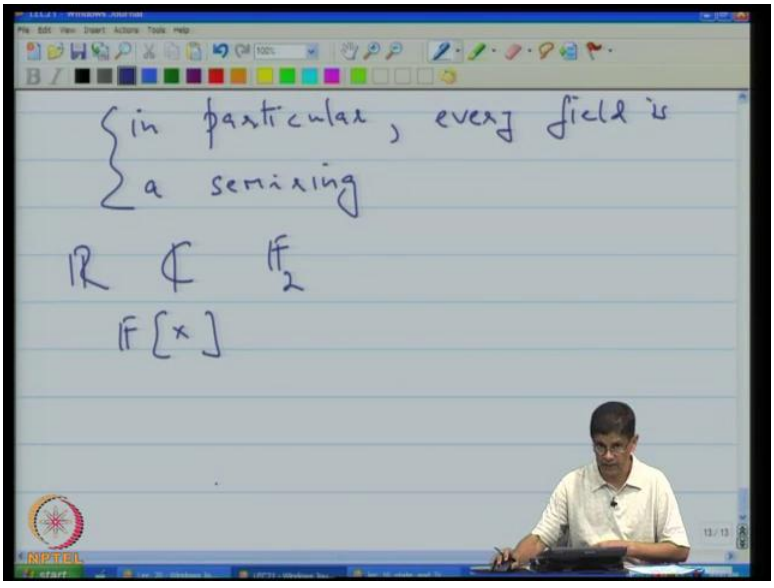
So what that means is that here I can put down that, so if I take this and this, I will get a semi ring example is little bit more than the semi rings, because you do have additive inverse, but certainly these are examples of semi ring. Fields certainly if you look at fields, fields have the property that they have rings are commutative, they have identity and division rings, so certainly fields also are examples of semi ring perhaps not so interesting, because the interesting examples come when you just barley meet the requirements.

(Refer Slide Time: 49:00)



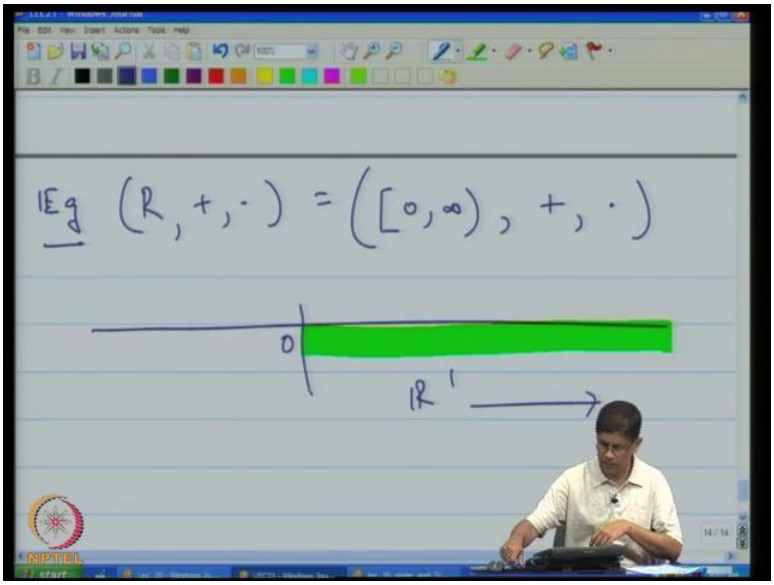
So here on examples will start on see that every commutative ring with identity is a semi ring, In particular, this holds true for fields, in particular every field is a semi ring, so that means that if you look at that set of all real number in the semi ring. Look at set of all complex numbers if you look at finite number field of two elements which is 0 and 1 this semi ring, because value this of example of fields.

(Refer Slide Time: 50:00)



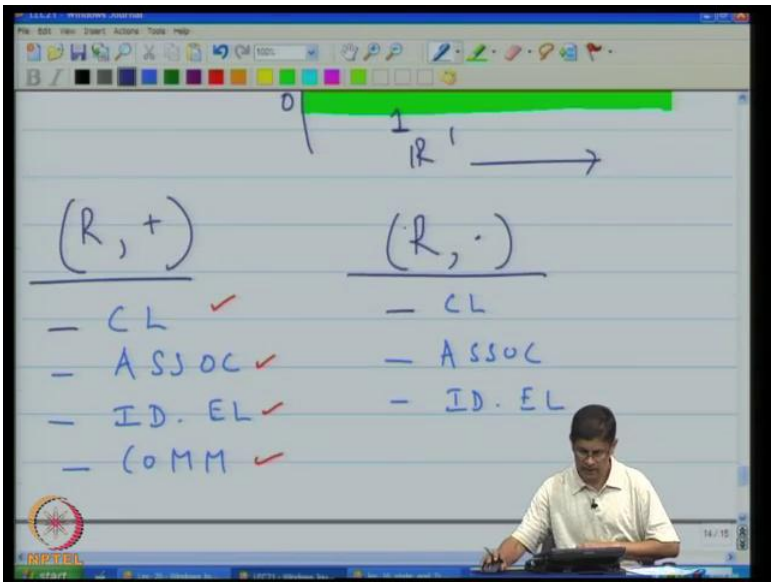
If you look at this set of all polynomial who were a field that is that commutative ring with identity and therefore this semi ring. If you look at the integers, the integers are in field domain it actually in commutative ring with identity and with more 0 divisors. So all of these are examples of semi ring.

(Refer Slide Time: 51:00)



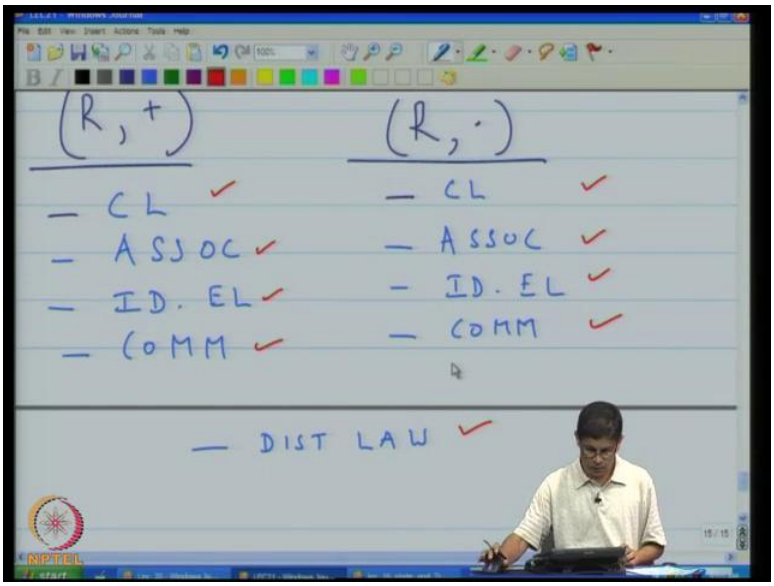
Here is in other somewhat usual example of an semi ring. But very useful, so here R plus dot is sub set of real numbers from 0 to infinity under usual addition and multiplication. So that means the timing real line, we are going to take the set of all positive real number including 0. So this rectangular brace here means here that you are actually including zero. Now why set this is the meant to represent the real line, and this is just nothing but 0. Now this is make that little sort whereas do not think a an axes are anything right. Now you might say y z that you take a sub set of real numbers, it turns out that case in so for an application coding theories I have concerned, what your are working with or either probabilities or else scaled versions of probabilities; and the scaling is done by a positive real number. Now any probability lies is a real number lying between 0 and 1, when you scale it using a positive scaling constant, it lie between 0 and infinity. So there explains why you are restricting a rotation 0 to infinity. Since what with real numbers you addition in and multiplication come in naturally.

(Refer Slide Time: 52:57)



So now how do we actually show this is semi ring so you to this show to this semi ring. We need to make sure that all the axioms are satisfied. Let us check that. So if I look under R and plus, then I notice that I have closure, I have the associative property, do I have the identity element, what is we additive element? Well, the additive element identity is 0 that is why we have taken care to include 0 explicitly, yes we do have the identity element, and then of course additional commutative. So we need all the requirements here, so there is no problem. Then let is look at R under multiplication; let us look at R under multiplication, and again it is easy to check that we have there is closed and multiplication that multiplication is associative, there is an identity element, which is just one real number 1, because after all real number 1 is sitting in here.

(Refer Slide Time: 54:45)



So it has a multiplicative identity. And multiplication is also commutative. We also have commutative property, and we also known that the distributive law holds, so all the axioms are actually satisfied. So this is an example of a semi ring. So I think this is a good place to stop. We have about couple of the minutes left only. So just to recap what we did today is we start it, I just completed discussion on convolution codes by actually clarifying the meaning of d free, which appear in expression in last lecture then after that we started in new topic which is generalized distributive law. And the generalized distributive law is the application of the ordinary distributive law perhaps in setting you would not have thought of earlier. And the application is towards for all purposes for the purposes of this code is for decoding error correcting code. However it turns of that same distributive law can actually be used in other setting as I mentioned the term belief propagation; so belief propagation is application not just decoding of code, but also for sense a networks for example.

So we started talking about GDL, and then I told you how distributive law can achieve a savings and computation through some simple examples. And then we went little bit mathematical, and I said well we would like to put it on a in right algebraic sorting which terms up which from the group of certain people setting and semi rings and semi rings are very much like ring only difference is that you missing the additive inverse. So we define that semi ring is we look at

some example, and we will continue the examples and continue our discussion of the GDL next class, so thank you.