Digital Communication
By Professor Surendra Prasad
Department of Electrical Engineering
Indian Institute of Technology, Delhi
Module 1
Lecture 37
Elements of Error Correcting Codes

If we can finish it today it will be fine, otherwise we will continue it in the next class. Obviously it is going to be a very brief introduction to the subject because it is a full fledge subject by itself and can be taught in a full semester course so we will essential dealing with the elements of error correcting codes, what are the basic ideas behind, okay. What we have learnt from our discussion so far is that use of redundancy is a clue on which Shannon's breakthrough result is based that is if you want error free communication of information we have to add redundancy to your (()) (1:54) to our message that we are transmitting.

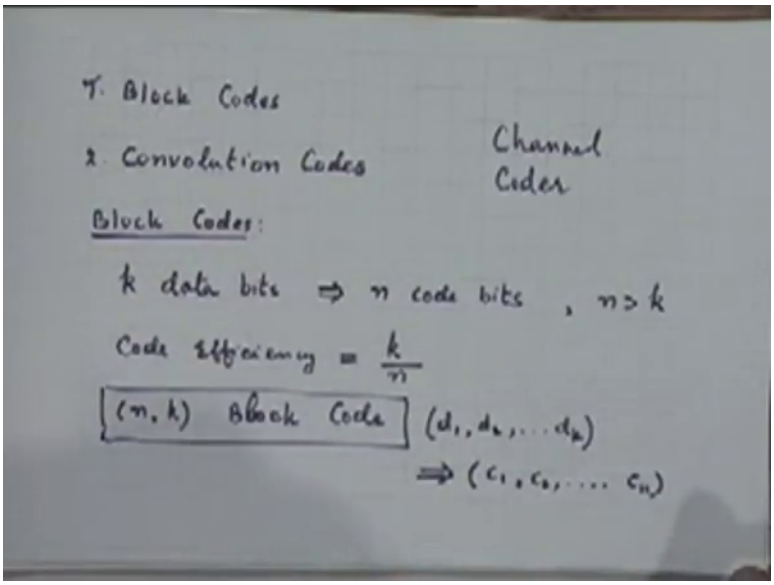
So here is a also a some (())(2:00) there on the one hand we have the source coding problem in which our effort is to remove redundancy to as much as possible, right? Because we want t represent the source digital representation of its source output efficiently you know what start putting redundant which are necessary in the channel. On the other if you want our transmission to be robust against errors in the presence of noise we want redundancy to be present there, right? It of course does not mean that we can do away with source code, right? Because the redundancy that is present by inefficient representation of the source is not something that you can utilize, right?

So what you really have to do is do source coding and then also introduce redundancy in a controlled manner in a manner that will allow you to make use of redundant information for removing errors separately. So really speaking in an overall digital communication system you have two kinds of encoders and decoders present, right? Two kinds of encoders at the transmitter and the corresponding decoders at the receiver. The source encoder its job is to obtain or give you an efficient digital representation of the underlying messages being emitted by a source.

On the other hand the so called the channel coder is a one which will introduce further redundancy in the trans wave message so that you can control the errors or if we able to detect

and correct errors that take place so that you can ultimately hope to get transmission of information with as few errors as possible, okay theoretically even error free information transmission is possible. So this is the basis for going for error correcting codes that we must have a channel coder which is able to add redundant information in a controlled or desired manner, okay.

(Refer Slide Time: 4:22)



Now there are two kinds of lock codes, we will talk about two kinds of codes, one are called block codes and the other kind of convolution codes. For lack of time we will not even touch convolution codes, okay but let me just give you an idea as to where they are useful convolution codes are some people according to (())(4:56) are really a more powerful class of codes because we are able to take care of not only additive white codes in noise channel situations but also in certain other kinds of channels, for example what are called bust error channels, okay but in addition to this there are some difference in opinion as to whether block codes or convolution codes are really superior with respect to each other or ultimately they are all the same.

But other than just this few words we will not really touch upon convolution codes at the moment and we will be essentially dealing with block codes where basically what we do is your coding is carried out on a clock by block cases if you remember the argument that we used for proving the concept of channel capacity Shannon's concept of channel capacity the argument was

asymptotic in nature, right? We had to we are making we are accumulating data bits the information bits before we carry out the coding, right?

And ideal situation in the asymptotic situation we are accumulating infinite number of bits before you carry out the coding because we have you know taking alpha t bits information bits and coding it into the block of beta t bits. Of course that means working with infinite block lengths infinite information sequence lengths which is in practical because it means infinite coding (()) (6:34) corresponding to coding delay, right?

So more realistic block codes accumulates information bits up to a finite number, right? Say you take k data bits or information bits I will be using this word data bits or information bits rather interchangeably from now onwards but when I say data now I am talking of information data information carrying data or information bits. And these will be encoded into a block of this will be you will be taking a block of data bits or information bits and coding it into a block of code bits so called code bits which will obviously be larger in number, right?

So corresponding to K data bits we will have n code bits this is a basic ideal of block coding where n is typically greater than K and this will bring down your effective information rate, right? That is the whole idea of Shannon channel capacity says in channel capacity if you are wrong which says that if you keep information rate below a certain value then there is a possibility of error free communication, right?

So this obviously will bring in that required reduction in information rate and your resulting what we call code efficiency which we also talked about earlier will be obviously K upon n because total message length is of n bits out of which information carrying bits are only K in number so your code efficiency is K by n and such a block code is usually denoted or called an n, k n, k block code this is a usual terminology for such a block code, more precisely you will the coding procedure consist of taking a sequence of K bits data bits which we can denote as a vector d1, d2, d sub k and map it this is about of coder we will do into a vector or sequence of code bits C1, C2, C sub n, okay this mapping is what is going to be done by the so called channel coder or channel encoder.

Now the first thing that we must answer or discuss before we proceed further is what should be the relative values of K and n because this is the first crucial thing to..

Student: (())(9:36).

Professor: No, code efficiency is not same as such channel capacity, the code efficiency will have to be less than channel capacity, right? But it is attribute of a code, is it clear? It is an attribute of the and the coder that you are designing but it design requires that you keep it channel capacity in (())(10:03) designing, okay.

Student: Sir we have source encoder and (())(10:08) encoder both are separate, so the product of the efficiency should be less than channel capacity or both, code efficiency and the maximum can be equal to channel capacity.

Professor: No, as far as see as far as these K bits are concerned they are supposed to representing source output it is assumed that you have done the best possible thing as far as your source coding is concerned in the matter of this representation, right? From now onwards we are not we are kind of segregated the two problems we are only looking at one problem at a time because they are really separate problems, right? In some redundancy is still there that will go on but we are not going to able to make use that.

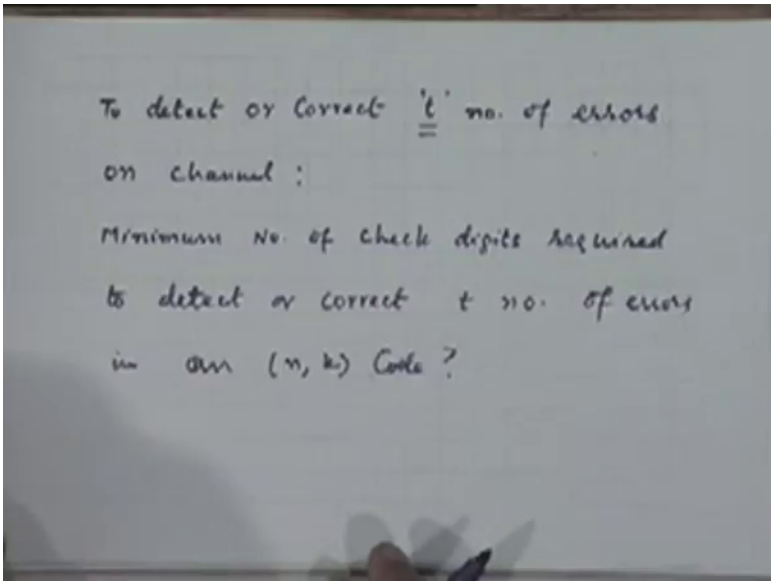
Student: We are not going to make that.

Professor: That is right, I mean there are some proposals and schemes (())(10:57) where people have tried to argue why not use the redundancy that is inherently present in the source itself to carry out error correction and detection but that kind of scheme is not has not being perfected yet I mean the theory of those kinds ideas has not been worked out very nicely, okay. So at the moment we have kind of segregated the two problems and this is how most of the real life systems work, the source encoding process is decoupled from the channel encoding process and the corresponding decoding process is also decoupled, okay.

So does that answer most of the questions that were raised? Fine, so we will take up this question of what should be the relationship between K and n how we should this is really effective you see in real life we will we will not really try to design a encoder or decoder our purpose will not be able to obtain error free communication because that we know is only possibly a theoretical possibility rather than a absolutely real possibility but we will try to go as near to it as possible and one way to do that would be to make sure that if a certain number of errors occur whose

probability is relatively large or significant then we will like to make sure that any error pattern up to a certain length we are able to detect and correct it, right? Which will effectively take care of most of the commonly occurring kind of error situations and therefore a effectively reduce your error information transmission, okay.

(Refer Slide Time: 12:55)

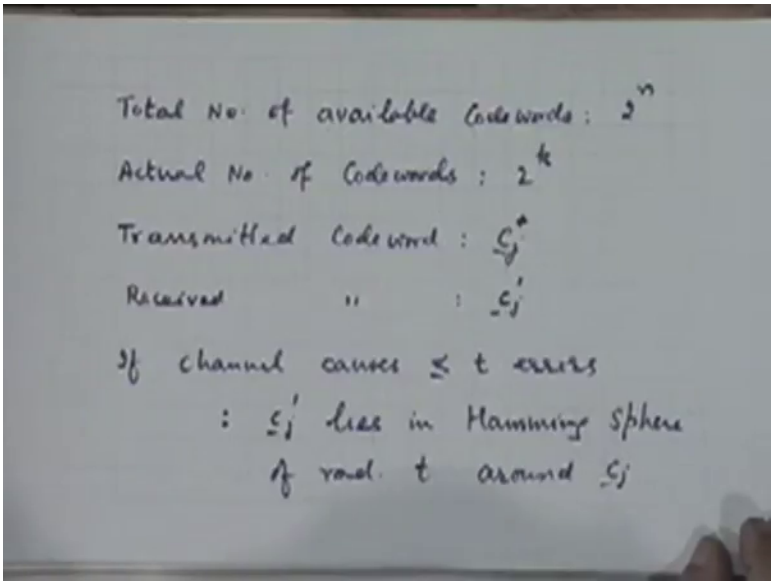


So one of the most commonly used criteria will be to able to detect or correct a certain number that you have decided say t that take place on the channel. So you are transmitting a code of length m and there is a certain small probability that up to t errors can take place because let us say the probability of having more in t errors it becomes so negligible small that it is of practically no importance to us. So we decide on this number t which is important to us it will depend on the channel conditions, right? If it is a very bad channel then t can be large if it is a relatively good channel t can be very small, right? We can have very small value of t and that is good enough for us.

So once we decide on this value of t that is important this becomes our crucial design parameter for the encoders channel encode. So we will like to have a relationship between K and n in terms of this parameter t, okay how many errors up to how many errors we want to be able to either detect or correct, right? Depending on whether you are going for error detection only or error detection and correction.

To answer this question let us first pose a more limited question of what is a minimum number of in fact this a question I am rephrasing here many of number of checked digits required to detect or correct t number of errors in an empty code and this is the question of interest to us but to answer this question let me do a bit of commentarial counting of a different kind of question.

(Refer Slide Time: 15:23)



Start from the let us start by counting the total number of available code words, when I say available code words you please imagine the same n dimensional hypercube that we have been talking about in the past here n bits n code bits we can think of this n code bits as vertices of appropriately chosen vertices of an n dimensional hypercube, right? Fine?

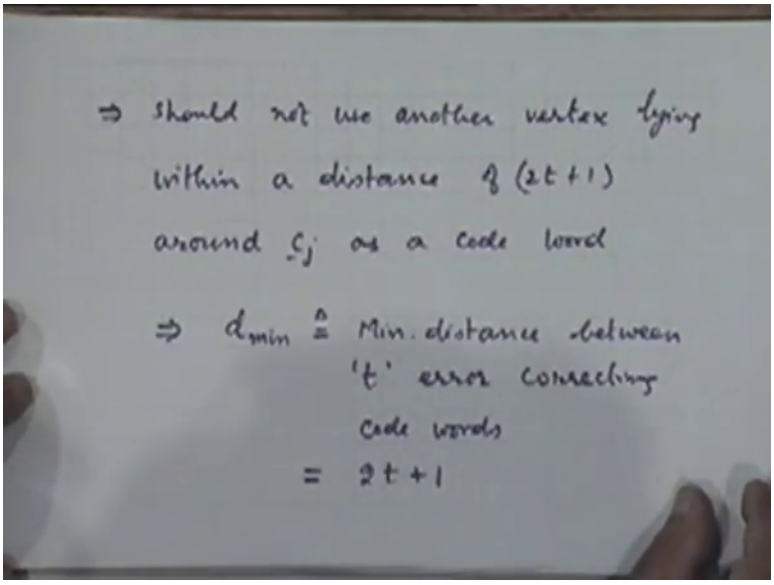
How many vertices are available? This is what I mean by total number of available code words the vertices which we can use as code words sequences which we can use as code words we have a choice of from 2 to the power n possible vertices, right? Out of which how many we should really select as code words? Actual number of code words will be 2 to the power K, let us say a particular code word is transmitting let the transmitted code word be some code word which we will denote by Cj prime, sorry Cj and let us say the received code word please ignore the prime here the received code word is Cj prime if no errors occur on the channel Cj prime would be precisely the same as what you transmitted Cj, right?

What we know have is a situation where we have agreed that we can aspect up to a certain number t of errors in Cj prime with respect to Cj, okay. So if the channel introduces or causes

less than or equal to t errors, okay it implies that our Cj prime will lie within the hamming sphere of radius t around the code vertex Cj transmitted code vertex Cj transmitted code vertex Cj, alright? Cj prime in that case lies in within the hamming sphere of radius t centered around the transmitted code word C sub j, right?

And this also implies that as we discussed earlier that we should not use the other another code word that we chose as another vertex which we chose as a code word should also similarly be how far (())(18:29) should be from this code word, no if you want to be able to have Cj prime corresponding to Cj lie within its own hamming sphere within the hamming sphere of Cj and similarly all t are less distant received code words from other code words also to lie within their corresponding hamming spheres and it means that this various code words must be separated from each other by a distance of 2t plus 1, right?

(Refer Slide Time: 19:21)



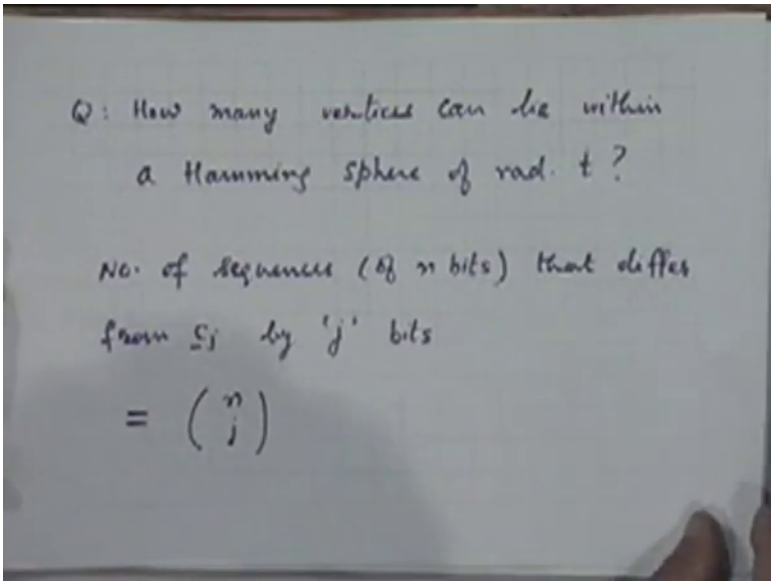
One the code word and two all 2t distant around it should not be code words, right? So it implies that we should not use another vertex lying between a distance of 2t plus 1 around Cj as a code word, fine? Because you want this 2t radius hamming spheres to be non-overlapping with respect to each other basically that is the consideration, therefore that implies that the minimum distance this is what we call minimum actually when I say distance here the measure of distance is in the sense of what we call hamming distance, right? Which we have been talking about in this context

is defined as the minimum distance between t error correcting code words and that should be equal to 2t plus 1 this is the important result.

So we have not yet answered the question of what should be the relationship between n and K we started with a more limited understanding of what should be the relationship between various code words, right? If we want t error correction capability then the various code words that we select should be such that the minimum distance with respect to each other in the hamming sense is at least 2t plus 1, okay. Now this will take us further to the relationship between n and K that we are looking for, we see further is that understood is there any question about this conclusion that we (())(21:42) for error correction up to t errors the required code should be such that minimum distance between any two code words of the code should be at least 2 t plus 1 you have 2 spheres you are drawing sphere in the centers or somewhere and you are drawing mind you t this t is an integer remember that t is an integer, right? You are drawing 2 spheres of radius t around each of them what is the distance between the centers, obviously they cannot touch each other so it is 2t plus 1 that is the basic idea, okay.

Therefore to answer what should be the relationship between n and K we first now take a, very good we first look at how many vertices will lie in each of these spheres then only you will be able to see what should be the value of n, right?

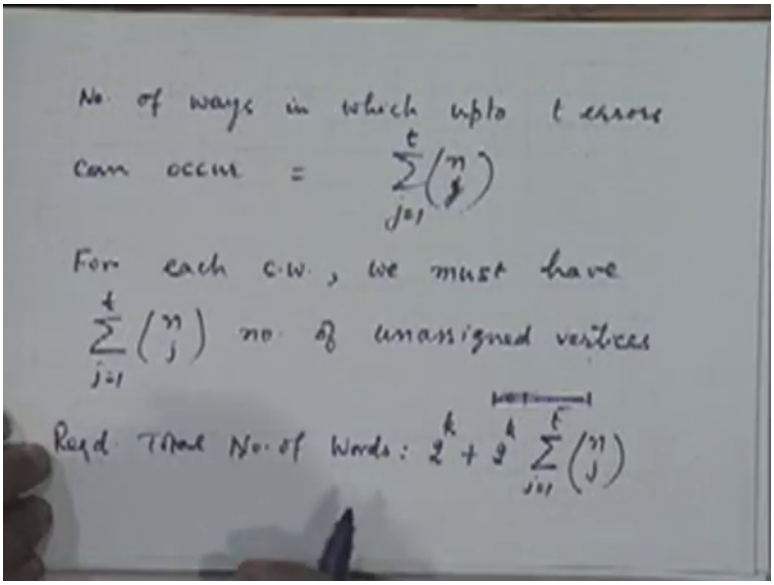
(Refer Slide Time: 23:04)



So the next question we address is how many vertices can lie within a hamming sphere of radius t, okay on it, what is a characteristic of all these vertices which lie within a hamming sphere of radius t we can think of them as if they are obtained because of errors in the channel on the basic code word, right? And errors up to minimum maximum of t.

Let us look at number of sequences that we are possibly likely to get number of sequences of obviously n bits because we are talking of code words that differ from C sub j the center code word around which the sphere has been drawn that differ from C sub j by j bits, okay where j is obviously less than t because when you say up to t errors remember we can have t errors or less so I am considering other moment a number j of errors which is less than t then we will sum over j. What is this number? Consider this number it is the number of n things taken j at a time, right? So obviously the answer is n Cj, okay is it clear? We have n things and you are looking at how many ways you can take out j at a time j things at a time in how many ways you can take out j things.

(Refer Slide Time: 25:08)



Now therefore that implies that the number of ways in which t error up to t errors can occur, what will be that? That is right, we take n Cj j equal to 1 to t not 0 because 0 will include the code word.

Student: That is also to be included that is in the sphere that has to be included.

Professor: I was going to bring it in later let us at the moment I am only counting the number excluding the code word, okay. And this is the therefore this is the number within one sphere and we have how such spheres, 2 to the power K, right? So now let us count our total number of vertices, right? So for each code word we must have that is each vertex that you select as a code word you must leave out so many vertices in the neighborhood unassigned as code words, is it clear? We must have n Cj j equal to 1 to t number of unassigned vertices, right? Yes.

Student: The code word original Cj it was already included in like for any given value of j where n Cj the number of possible other sequences which can come around, if j is differ and it includes that Cj also.

Professor: But I am not taking j equal to 0, that would include that if I include j equal to 0 which I will do later because this is the number of other sequences which differ from it in j places if it is differing from it cannot included unless j is equal to 0.

Student: What I meant was (())(27:15) j equal to 1 you have value assigned and you have the (()) (27:20) so it differs from (())(27:23) it could be in any (())(27:26).

Professor: No, when I say n the sequence is this means they are in the neighborhood n sequences that differ from it in one place, total sequences, why you are getting confused? We have let me draw a picture we have a center we have all kinds of sequence what is in the neighborhood, right? Those which are at distance unity that is they differ from it only in 1 bit though it differs from it in two bits those which are differ from it 3 three bits, right?

Student: I am saying is if j is equal to 1 that means there are n different combinations.

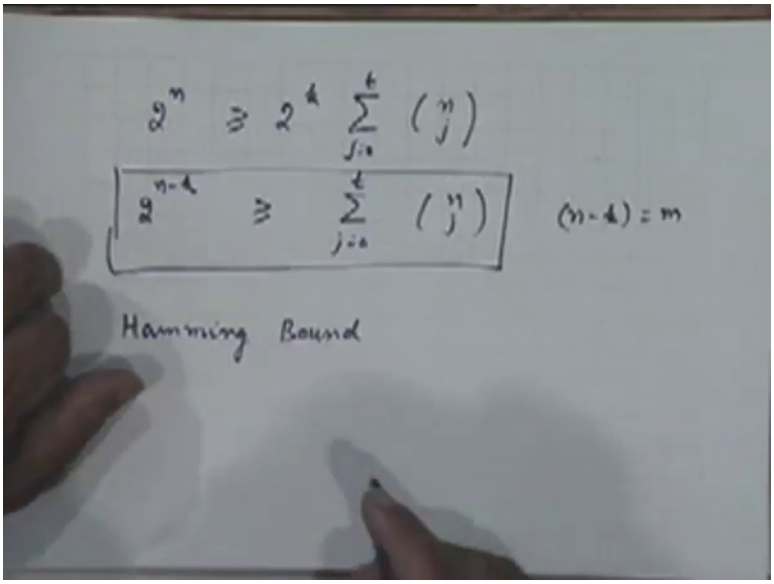
Professor: There are n other sequences it differ from Cj in one place, why n minus 1? It should be n only, suppose I have a particular sequence, right? I can construct n other sequences by replacing one of each this where opposite bit, so what is the problem? I can construct n other sequences by replacing n bit by its complimentary bit not n minus 1, okay this is okay I mean this is basically using combinatorial theory but you can work it out otherwise, alright? Does anybody else have this confusion now? Fine?

Let us do the counting, the counting is our total number of code words that we must have total number of vertices that we must have must be the total number of code words plus what we have

to leave unused, right? So total required total number of words or total number of vertices words or vertices you can use into changeably this will be equal to 2 to the power K this is again this 2 to the power K is the number of code words themselves because I require 2d power K code words plus for each code word I require so many unassigned vertices, right?

This is the total number of required vertices in the end dimensional hypercube that I have it must have at least so many vertices for such a code to be feasible such that minimum distance between any two code word is 2d plus 1, okay for a code of this kind to be possible to make I must have at least so many vertices in my hypercube, is it clear? This 2 to the power K term is the number of code words itself and for each of this code words I have so many unassigned vertices, so for 1 multiply by 2 to the power K code words this is the number of unassigned vertices this is the number of assigned vertices total number of vertices must be the sum of these two, okay.

(Refer Slide Time: 31:15)



So that implies that 2 to the power n should be greater than or equal to 2 to the power K as you can see I can take 2 to the power K factor the out I will be left with 1 plus this and I can include that is now in this summation by putting j equal to 0 which is the same thing that you are saying more or less that is our result, okay which is sometimes also put in this way 2 to the power n minus K is greater than or equal to.

This tells me if I want a t error correcting code how many check bits n minus K is the number of additional check bits or redundant bits I am adding, isn't it? n minus K let me call this m, what

should be the value of m such that I get t error correction capability? Okay. This result is known by Hamming Bound by the name of Hamming Bound, okay. So it is not an explicit relationship it is somewhat involved relationship what one can (())(32:08) out, right? For a given value of t one can work the combinations of values of n and K for which this relationship will be satisfied. For a given value of t and K you can determine a corresponding value on n, right? Or in (())(32:25) in an different in any combination that we (())(32:30) interest, any question on this result?

Student: But can a code bit (())(32:43).

Professor: That is a very deep question, I cannot answer it straight away but codes do exists for such values of n, he is asking me whether I can guarantee the existence of a suitable code of this thing the existence is obviously there because, it depends on what you mean by finding a code if you say whether you can find a code with a particular nice structure then it is a difficult question to ask for but if I just say whether it is possible to find a code I already know that it exist because this bound has been obtained basis of a construction and the construction procedure is I will chose an arbitrary vertex, leave out so many vertices unassigned, right? Then go to the next vertex and you know by construction I can always go and do it but it may not be possible to form in a nice linear structured code or any kind of structure code that will depend on the kind of structure that can imply, okay.

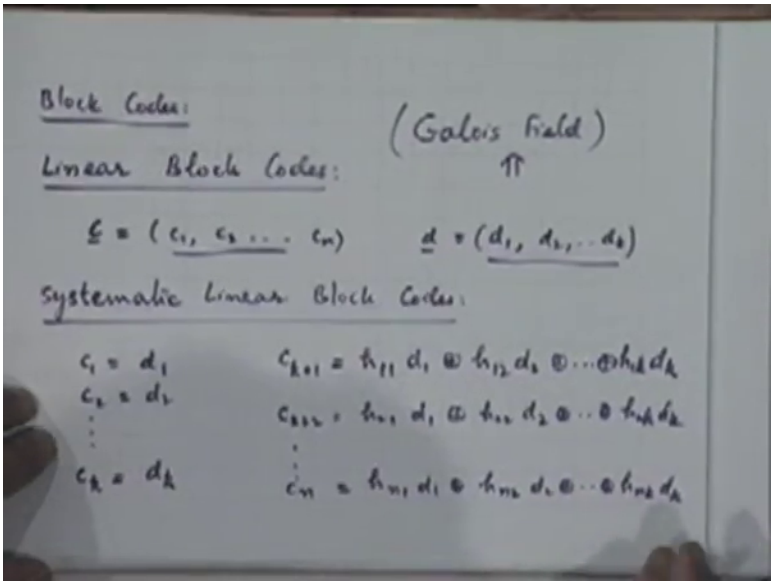
So at the moment the answer is yes it is possible to find a code how easy (())(33:58) difficulties to implement it is a different matter. Now similarly if you are on interested in error detection and not error correction then there are certain kinds of communication systems particularly where a liability is poor where one does not really entire is lie on error correction automatic error correction where you rather retransmit a message if errors have been too many errors have been detected then I will then just automatic error correction because there can be number lot of problems in that situation this kind of communication systems are called ARQ system automatic repeat request are systems.

They prefer own error detection if it is rather then error correction capability, okay. For that you require code with minimum distance team in prime equal to this answer should be obvious, if I only want to be able to detect up to t errors and not have a capability of correcting that t plus 1, right? Is it obvious?

Student: Because we cannot have overlapping (())(35:13).

Professor: All you want to make sure is that you are able to detect it, so the answer is t plus 1, okay.

(Refer Slide Time: 35:35)



Now let me come to some basic theory about block codes, there construction and there decoding. Now a subject of block coding is a very highly developed subject today and essentially what it really biased on to is finding mathematical structures which will enable simple or reasonably simple implementations of code generation as well as decoding for codes which have good properties, good error correcting properties, okay.

So basically what I am trying to say now is so far we are only talked in an unstructured manner, that we have a n dimensional hypercube and somehow we have to assigned those vertices as some of this vertices as code words and leave rest of them unused. The question is how do I go about selecting words, which word should be code words and which should be left unassigned that is what coding theories really all about, right? How do you carry out that design of assigning the proper vertices as code words and leaving (())(36:50) I mean that is what theory of error correcting codes are I recon a control coding is all really about.

Now one popular structure which is more or less basis of all error correcting codes most of error correcting codes I should not say all is the linear structure linear block codes that is the structure

of linearity where basically the idea is that a code vector C C1, C2, up to Cn is obtained from a data vector d d1, d sub 2 to d sub k essentially by taking a linear mapping of this data vector from a k dimensional space to n dimensional space, okay just essentially each code bit is taken as some kind of a linear combination the corresponding information bits but mind you we are now working with not our normal our field of operation is not the normal real number field in which we carry out this combinations because these are binary numbers, okay.

The binary numbers we have a separate field of operation which we are familiar with and it is called the Galus field, okay where basically addition is done modular 2, right? But there are many other properties associated with this field I am not really going it a proper study of coding theory first would require us to study this field theory fairly throw away but we do not have time for doing all that whatever (())(38:39) knowledge about modular to addition and all that we can study our basic theory on that basis, okay.

So in this field the linear combination operation is essentially exclusive of with a bit exclusive of operation, okay. So when I set a linear combination of these bits it only means I am taking some d1 plus d2 plus d5 and things like that this plus being really an exclusive OR operation, right? Because basically your addition is modeled to addition. Now out of all the this bit class of linear block codes there is a subclass without any loss of generality and this subclass is known by the name of systematically linear block codes, we will discuss these.

In the systematic linear block codes the first key bits of the code are taken identically to be equal to the corresponding information bits themselves, okay. So C1 is not taken a linear combination of these but C1 taken as simply equal to d1, C2 is taken as d2 and so on up to C sub k s taken as d sub k. So the first k bits of the code sequence are the same as the corresponding information bits or data bits which we are trying to encode mind you this is the encode to encoder the vector d is a input to encoder and coder is a mapping device which converts this k bit input sequence to a C bit to a n bit output sequence C, right?

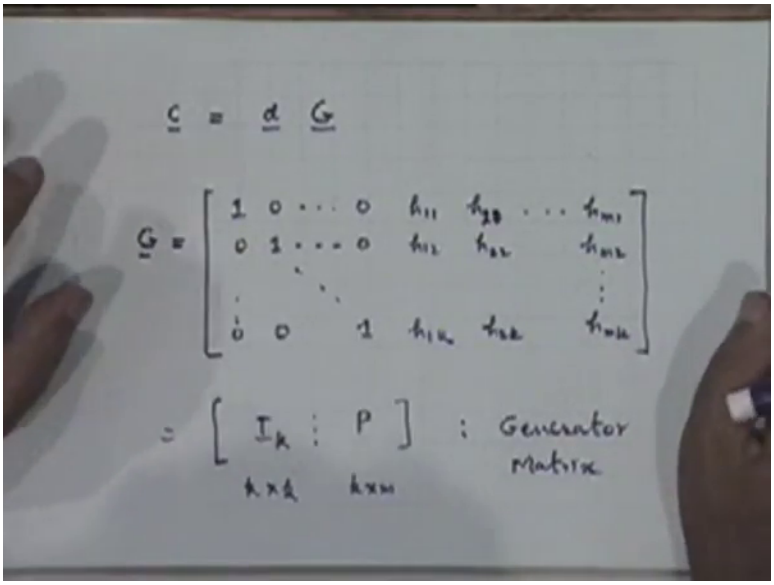
The systematic linear block encoder will first produce first k bits straight away in this manner practically no processing is required there, okay. For the rest of them it does that linear combination so Ck plus 1 for example could be h11 d1, h12 d2, h1k dk, okay where h11, h12, h1k are so we are taking a some linear combination of these k bits these are the coefficients of

linear combination, obviously because you are working in a modular 2 field these coefficients also take on values only between two values 0 and 1 binary values 0 and 1, alright?

d1, d2, dk obviously take 0 and 1 values so these operations are essentially modular 2 operations, these are the some coefficients of linear combination which we have to select, right? It will depend on the design of the code. Similarly Ck plus 2 will be given by h21 d1, h22 d2, h2k dk and so on, finally C sub n will be hn1 d1, yeah this should be m I will call that m (())(42:39) you are quite right because there is only n minus k check bits and these are the check bits these are the data bits, this operator is exclusive OR operation modular 2 addition, right?

Now obviously this mapping from the vector d you can see each of these C's each of these components of C depends on essentially this vector d, right? I can express it in the form of a linear mapping, you know how linear mappings are described mathematically algebraically? By the use of matrices. So best thing is to use a matrix representation for this set of equations which you can write something like this the code vectors C is the data vector d multiplied for the mapping or so called generating matrix g, okay where g structure of g let me see whether you can see I have taken C and d to be what kind of vectors, row vectors, right? These are row vectors and alright let us see whether you can tell me what is structure, right?

(Refer Slide Time: 44:05)



Is it obvious? And this will be h11, h12 up to actually along column I will go h12 to h1k and this will be h12, h22 this should be h21 I am sorry h21 h2k and this is hm1 to hmk because imagine you have a row vector in front of it, right? That is d1 to dk C1 is equal to d1 C2 equal to d2 C3 equal to d3, right? And do I need to explain this I wish that, no I have taken C and d to be row vectors if they are column vectors if they were columns vectors what you are saying is right, right? Remember this C and d are row vectors and the equation is being written like this what you are normally used to it is writing it like this they are being column vectors, okay so just keep that in mind while, unfortunately space is a bit limited I cannot display that two together then becomes obvious but you have the minimum nodes so I think you can see it from there.

So I can write this as a k into k unit matrix this is k into k and this is some matrix which will call matrix P it is known by the name of check matrix its dimensions are k into m k into n minus k and this matrix is known by the name of generator matrix g. So a systematic linear block code is essentially specified by the specification of an appropriate g matrix (())(46:45) matrix, okay that is at far as the basic definition is concerned, any questions?

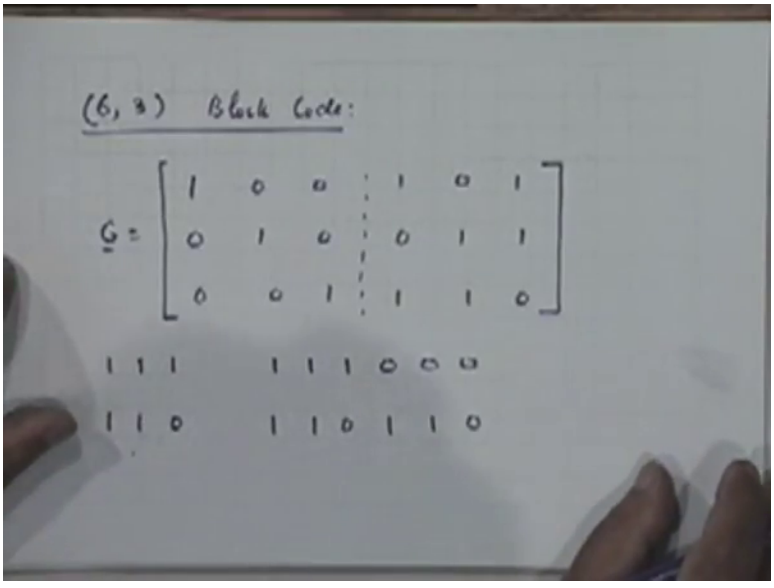
I will give you an example of, limitation in what sense? Okay now you are saying how one should select g, right? That is a matter of designing such codes you are not at all touching the topic here, okay there are lots of things we have to study to be rather making this statements of that kind, you have to really study the structure very very thoroughly and then understand it in

terms of what properties we want from it, right? And then only we are able to say something that it affect.

Yes some of the stipulations you are making may be correct but unfortunately I cannot take up that for discussion here because lack of time, as I said this is a full-fledged subject by itself and proper study of that will would require you to study finite field theory, right? Because you in fact we may not always be working with only binary numbers we are working with armory numbers in armory some power of 2 and first you to understand the theory of numbers in that field and then only we can do a proper study of error correcting codes in a very general way but even for the binary field, right? g of 2 in (())(48:21) base of 2 one has to understand what that field is all about, what are its properties and then how do you study the properties of such matrices g in that field.

So I am not really going to all that detail which for those of you are interested can perhaps hope to take a course in information theory and coding next semester if it is offered next or next to next semester. At the moment let us I only want to give an idea that using such things how can we code word and coding and decoding, right? Basically suppose we have a code like this how do we decode it?

(Refer Slide Time: 49:08)



Those kind of things are all I can cover with this but before doing that I think I will try to finish this talk today but just taking a simple example of a 6, 3 block code where g is 1, 0, 0, 0, 1, 0, 0,

0, 1 this is your i3, right? And the parity check part this p matrix is something is called as parity check matrix we will see a reason for that for one it is obvious that it is generating the check bits, right?

So for that reason it is called the parity check matrix but it has some other physical significance in decoding which will take up when we talk about decoding, okay. So this will be we have to specify the p matrix here let this be given by this 1, 0, 1, 0, 1, 1, 1, 1, 0 so that is a 3 by 6 matrix corresponding to a 6, 3 code. You can now generate a mapping of this kind if the input sequence is 1, 1, 1 your d1, d2, d3 is 1, 1, 1 what will be our code sequence? It will be 1, 1, 1 the first three bits are going to the same and next one is going to be all three will be 0, right?

Similarly if it is 1, 1, 0 this will be 1, 1, 0 this will also be 1, 1, 0 and so on you can construct a table of 8 code words I think I will just stop here and next time we will discuss suppose I have a linear block code systematic linear block code which has this structure how do we do the decoding at the other end, okay. Is tomorrow on or off, there is a class tomorrow is it been announced? Because I do not know what they did. So I think then we will try to finish it by tomorrow and Monday if you want we will meet otherwise we will have to think.