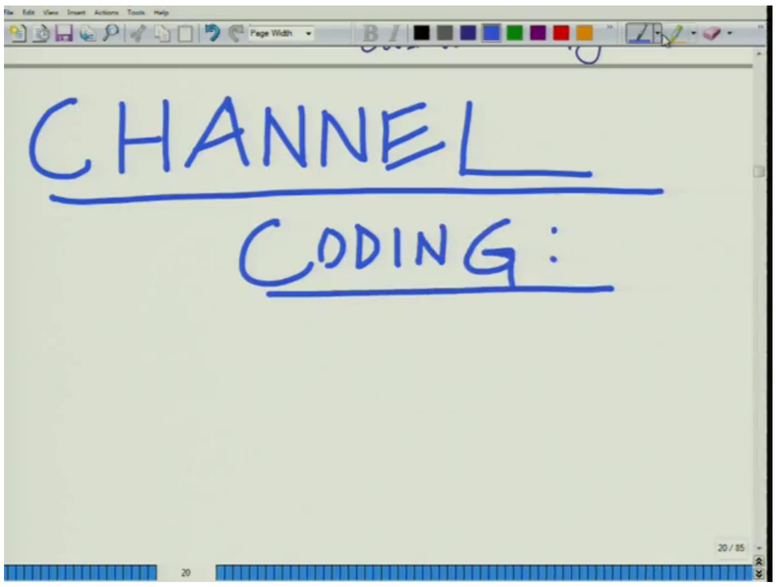
Principles of Communication Systems - Part II

Prof. Aditya K. Jagannatham
Department of Electrical Engineering
Indian Institute of Technology, Kanpur

Lecture - 48
Introduction to Channel Coding, Rate of Code, Repetition Code, Hamming Distance

Hello, welcome to another module in this massive open online course. So, in this module will start looking at a different topic which is channel coding all right. So, so far we have seen source coding and now which is basically we said to represent the symbols of a source efficiently in terms of bits.

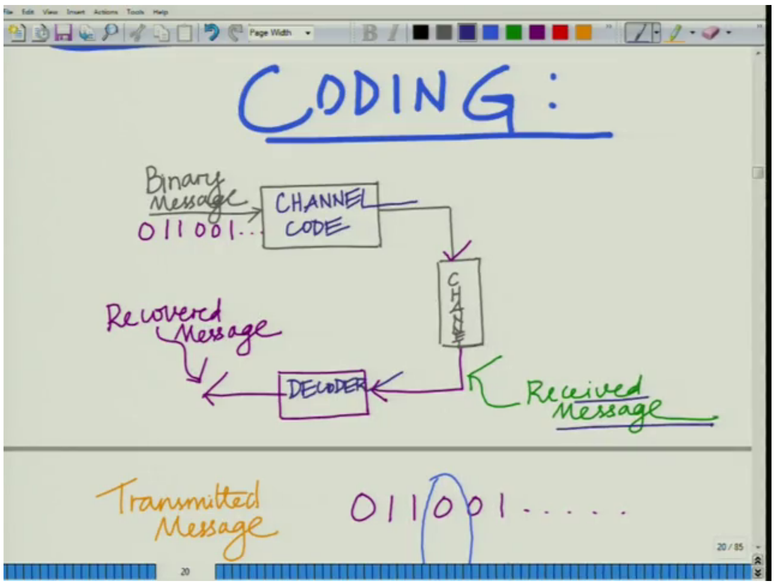
(Refer Slide Time: 00:40)



CHANNEL CODING:

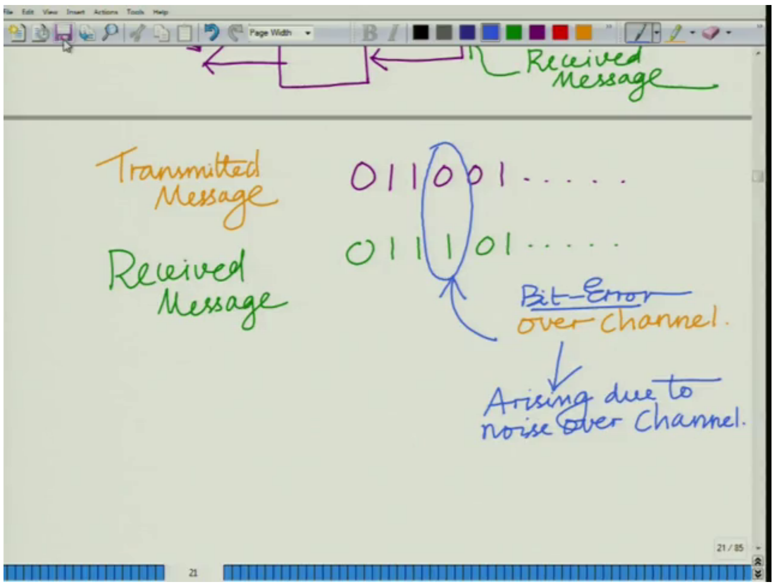
Now, we will start looking at a different aspect of communication that is communication over a digital communication channel which is channel coding. So, you want to start looking at a new topic which is channel coding. Now as we have seen in a communication system.

(Refer Slide Time: 01:09)



What we have is we have a binary message, we have a binary message which will pass through some block, I am going to name this block later and that passes through the channel. It passes through the channel followed by some other block we are going to name these blocks later and followed by the output. So, naturally here at this point we have to have a again we have to be able to estimate all right, so this is the recovered message. So, let us say we are transmitting the binary message 0, 1, 1 let us say any arbitrary message.

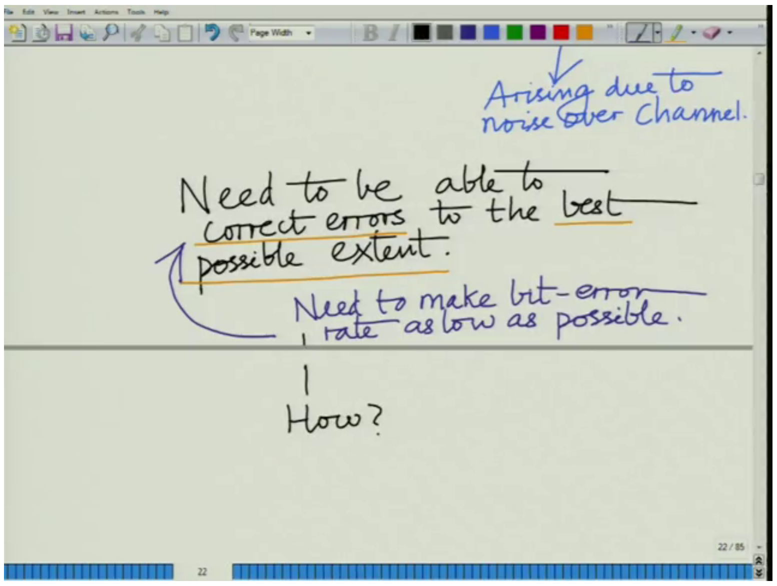
(Refer Slide Time: 02:23)



So, we have this is the transmitter message. And the corresponding received message or the reconstructed. Let us say the corresponding received message not the reconstructive, we are talking about let us say at this point. Let us look at let us make this fine distinction, let us make this as the received message. And let us say the received message or the equivalent received string, remember we are looking at digital messages therefore; we are looking at digital sequence of digital information bits binary information bits that constitutes a digital message. The received message can be something like 0, 1, 1, 1, 0, 1. And if you observe this thing what you will see is while the transmitted messages is 0, 1, 1, 0, 0, 1 followed by the rest of the message. The received messages 0, 1, 1, 1, 0, 1. So, if you can look at this position there is an error in the received message the transmitted bits the 0, received message or received bit is 1.

So, there is an error or more specifically there is a, there is a bit error. So, this bit error arises over the channel. So, there is a bit error over channel or arising due to basically the noise in the channel. This arises basically due to the noise over the, this arises due to noise over the channel. So, we need to be able to, correct? Such errors which are arising over the channel right, because we are transmitting a stream of digital communication digital bits which constitutes the digital message that is transmitted over the channel at the receiver, correct? Due to noise over the channel, correct? Due to the noise effects of the channel the received message or the received sequence of bits received digital message is not in accordance with or is not identical to the message that has some bits or an error all right, which arises due to the which these bits errors arise due to the channel imperfections and the receiver we have to be able to recover from these (Refer Time : 05:54) bits that is recover the errors which are arising recover from these bit errors or errors is the message which are arising due to the channel imperfections.

(Refer Slide Time: 06:08)



Arising due to noise over Channel.

Need to be able to correct errors to the best possible extent.

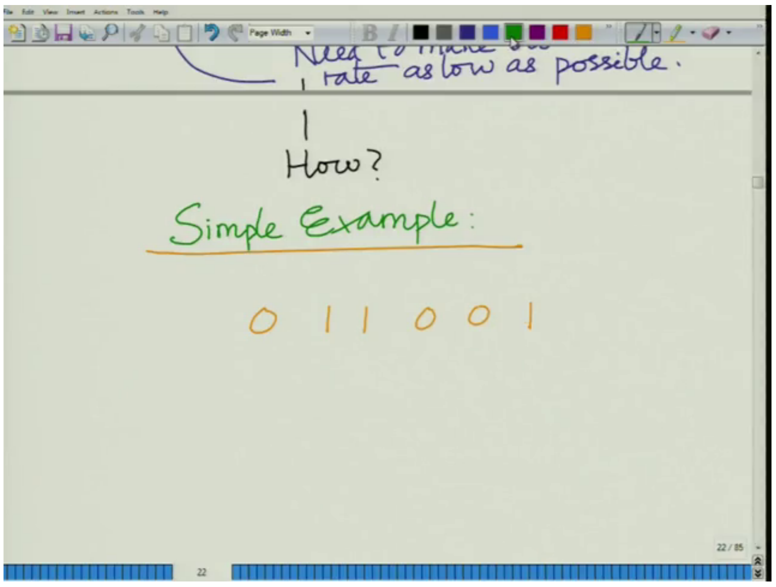
Need to make bit-error rate as low as possible.

How?

So, at receiver so, we need to be able to correct the errors, correct the errors to the best we need to be able to correct the errors to the best possible extent. Now remember we are not looking at correcting, we are not looking at correcting all the errors it might not be correct to it might not be possible right. In fact, that is what Shannon's capacity theorem tells us that is it is only possible to communicate with a diminishingly lower that is with the diminishingly lower probability of error. It might not be able to recover from all the errors bit errors that occur over the channel, but we will try to attempt right it is our aim to recover that is correct this digital message to the extent to the maximum extent that is possible that is to recover from as many bit errors as possible. That is I we might not be able to make the bit error rate exactly zero, but idea is to make it as small as to make this probability of error as small as possible, that is the idea.

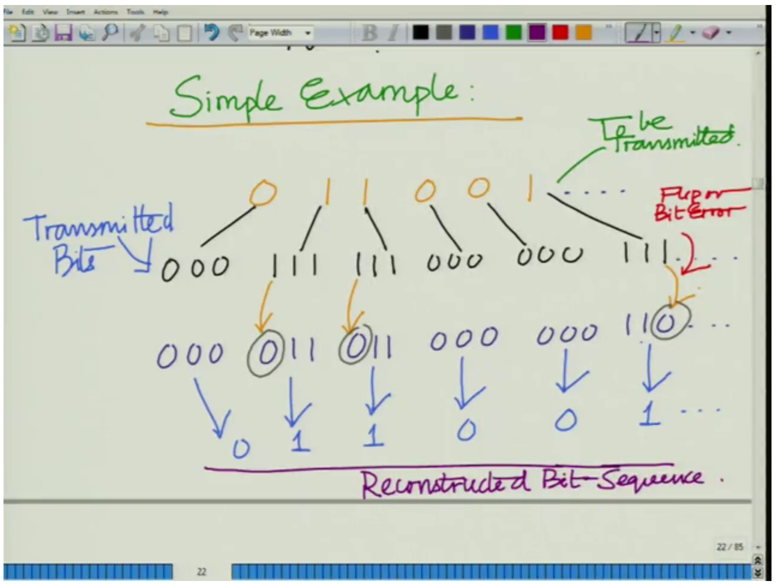
So, we need to be able to correct the errors to the best possible extent that is need to make it bit error rate as low as possible, need to make the bit error rate as low as possible. Now the question now arises how does one do this? How does one do this? Let us look at a simple example to eliminate bit error rate.

(Refer Slide Time: 08:24)



Let us look at a very simple example to understand this better I am going to look at a very simple example which might almost be something that is rather inefficient as you are going to see shortly correct, but for the sake of understanding let us go back again to our transmitter sequence of bits 0, 1, 1, 0, 0, 1. What I am not going to.

(Refer Slide Time: 08:56)



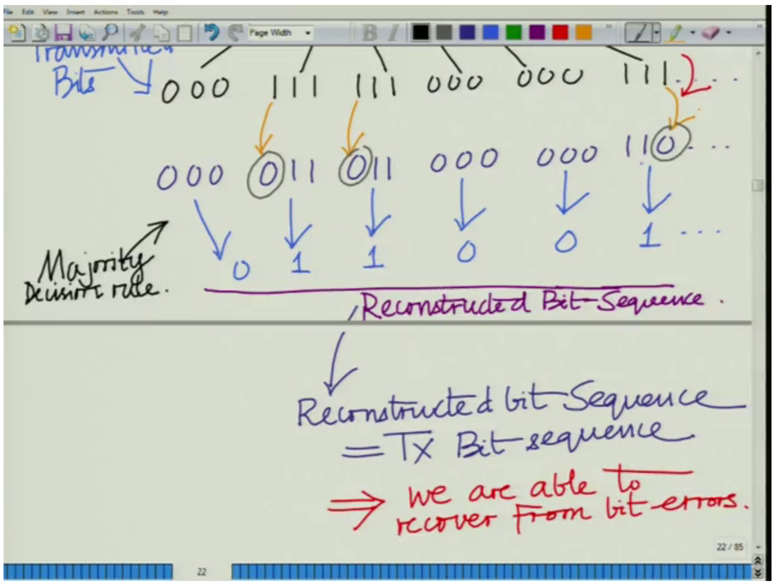
So, this is the transmitted 6 that is sequence of bit 2 bit transmitter not transmitted, this is now to be transmitted. What I am going to do now is I am going to repeat each bits of transmitting 0, I am going to repetitively transmit each bit 3 times 0.

So, now this becomes my actual transmitted or let us put it this way, this is the transmitted series of bits. Now let us say if there are some bit flips for instance let us say the first bits are 0 0 0 this bit 1, 1, 1 the first bit 1 flips to 0 rest are intact 0, 1, 1. Next also let us say this bit flips 0, 1, 1 let us say next received correctly 0, 0, 0. Let us say this bit flips the last one flips to a 0 and so on of course. So, what we have seen is this bits flip, this bits flip this bit is. So, I am using the arrows flip or basically bit flips arising due to errors, correct? These are basically you will realize bit flip or error. Basically a flip or a bit error. So, what we have is we have bit errors. But while decoding, you can see that there will be no, there wouldn't be any problem, because if you use a majority rule or majority logic decoding, by majority logic decoding you mean that we are repeating each bit 3 times.

So, if 2 or more of the bits are 1 then you decide a 1 or 2 or more are 0 then you decide a 0 this is known as a majority logic. It is a very simple rule that is you look at each bit group of 3 bits assign the majority assign this assign the decoded bit to the bits which are in a majority. If the zeros are in a majority then assign a 0 if the ones are in the majority then assign 1. For instance here we have 3 zeros. So, this will; obviously, be a 0 correct. So, this will be 0, 0, 1, 1, 2 bits are 1. So, this will be 1 0, 1, 1 one again this will be 1, 0 0 0 again this will be 0 majority 0 0 0 0 no question this is 0, 1, 1 0 majority is 1. So, this is a 1. So, basically we have got our this is our decoded so, this is our not use the word decoded this is our reconstructed.

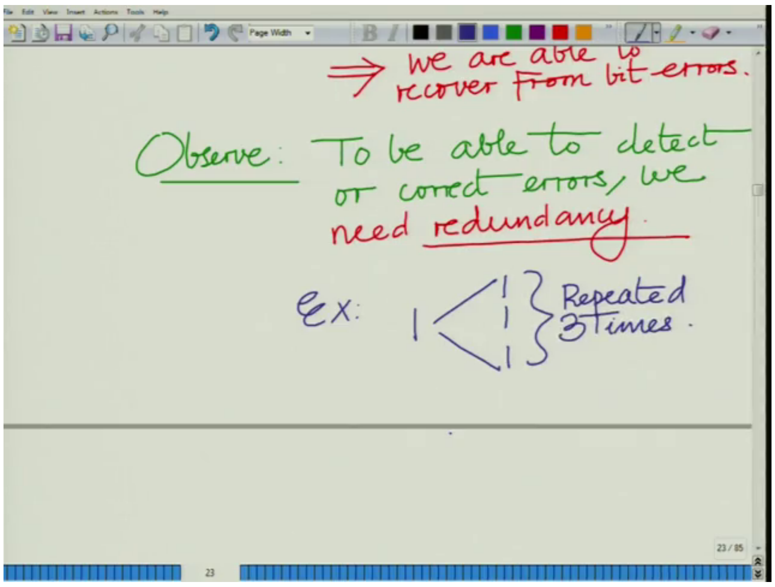
So, now you can see this reconstructed despite the errors, the reconstructed bit sequence is identical to the transmitted bite sequence.

(Refer Slide Time: 12:47)


Transmitted Bits
000 111 111 000 000 111 ...
000 011 011 000 000 110 ...
0 1 1 0 0 1 ...
Majority Decision rule.
Reconstructed Bit-Sequence.
Reconstructed bit Sequence = Tx Bit-sequence
⇒ we are able to recover from bit-errors.

So, now you can see we have record in principle of course, there are lot of problems with whatever talked about in principle equals the transmitter bit sequence, reconstructed bit sequence is equal to the transmitted bit sequence. It this part the bit errors which implies that we have been able to recover from the bit errors which implies that we are able to recover from the bit errors. We are using a majority decision rule now here we are using a majority decision rule. That is also something that you should remember majority decision rule and that is very obvious. Whatever now observe therefore,

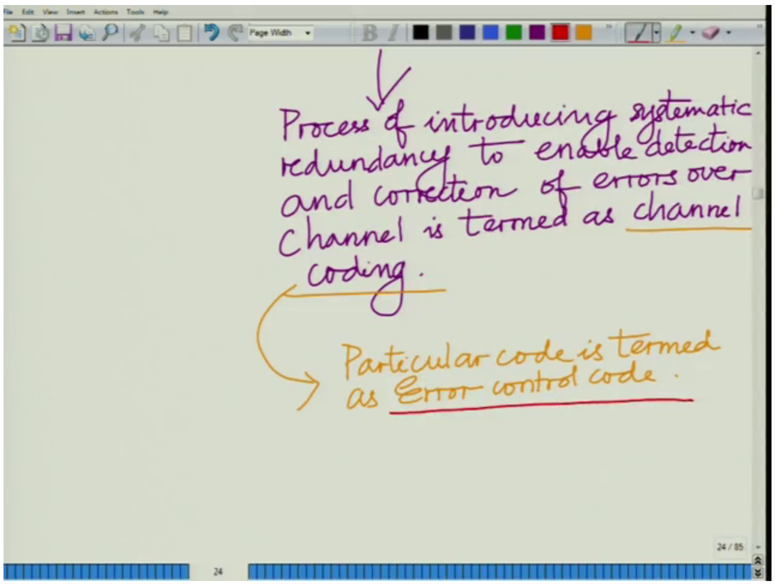
(Refer Slide Time: 14:08)


⇒ we are able to recover from bit-errors.
Observe: To be able to detect or correct errors, we need redundancy.
Ex: 1 < 1 1 1 } Repeated 3 times.

Now the first observation that you can make is observe that to be able to detect or, correct errors this is important to be able to, correct errors we need some form of redundancy. For instance here we are repeating each bit 3 types already we have do not need to do it we can transmit each bit a single time.

But here to be able to recover from the errors we are repeating each bit 3 times. So, we are introducing redundancy in the system. So, to be able so, that is the first point to be able to detect, to be able to detect, detect or correct errors we need, we need redundancy. For instance example you are taking the bit 1 we are repeating it 3 times, repeated 3 times. Now this process right this process of systematically introducing redundancy. To be able to detect or, correct errors introduced over the channel right introducing systematic redundancy in the transmitted bits stream to be able to detect or correct error is transmission is known as channel coding and the particular code, that is the particular format of redundancy that is employed that is known as the channel code or an error control code.

(Refer Slide Time: 16:17)

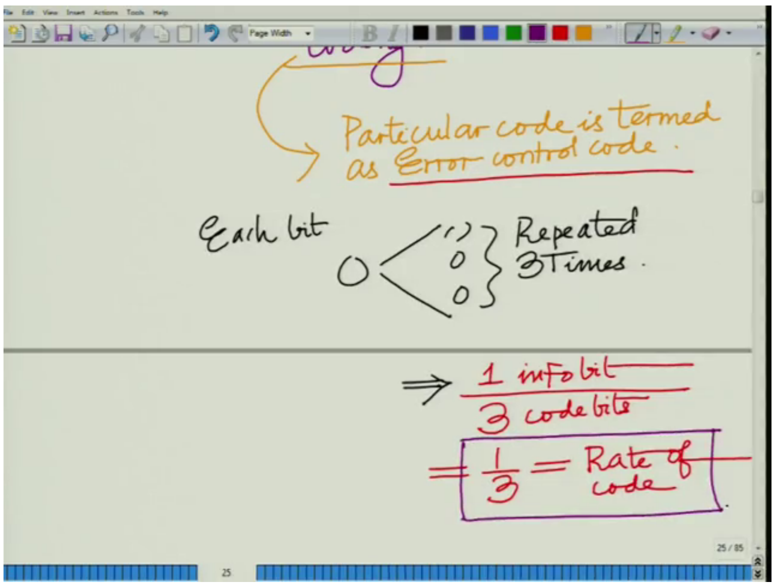

Process of introducing systematic redundancy to enable detection and correction of errors over Channel is termed as channel coding.

Particular code is termed as Error control code.

So, the point is this process, this process of introducing systematic redundancy to enable detection or correction for that matter and detection and correction of errors over channel or introduced by the channel is termed as, is termed as channel coding. So, this is basically termed as channel coding. So, this process of systematically introducing is a or and the particular code is termed as, and the particular code is termed as an error control code. So, the particular code this is termed as an error control coding. So, this process is known as channel coding that is introducing the systematically introducing the redundancy, the code that is employed is termed as an error control code.

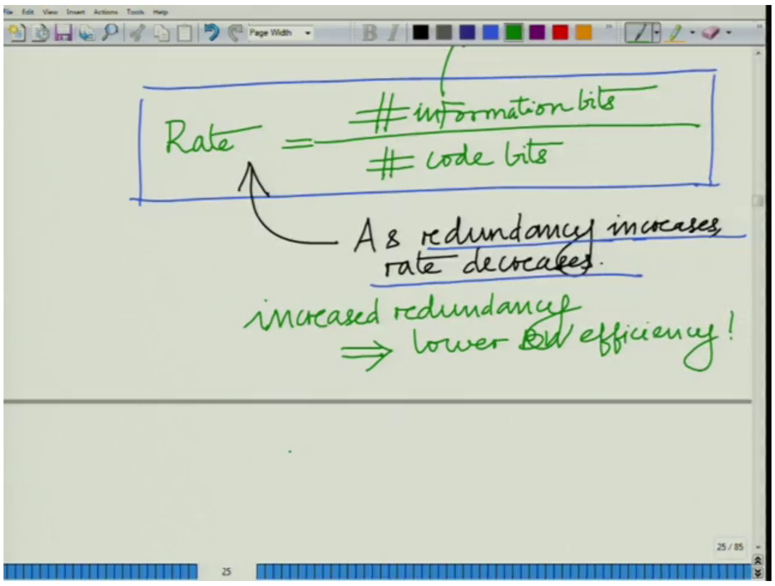
So, naturally when you employee encoding at the transmitter you have to employed decoding at the receiver all right. So, that is what these blocks are. So, these blocks constitute basically your channel code let us now fill in these blocks, these blocks constitute your channel code that is your encoder and this is a corresponding decoder and out of the decoder you get the, so you have the received message you decoded you get the recovered message or the reconstructed message, you can also think of this as the reconstructed you also called this as the reconstructed message.

(Refer Slide Time: 19:28)


Particular code is termed as Error control code.
Each bit
0
0
0
0
Repeated 3 Times.
⇒ 1 info bit / 3 code bits
1/3 = Rate of code
25 / 85

Also observe that each bit for example, 0 is repeated 3 times in this simple example each bits repeated 3 times implies for everyone information bit there are 3 code bits which is equal to 1 over 3 this is termed as the rate of code. So, the rate of the code is basically 1 over 3. What is this? This is basically your number of information bits.

(Refer Slide Time: 20:20)



$$\boxed{\text{Rate} = \frac{\#\,\text{information bits}}{\#\,\text{code bits}}}$$

As redundancy increases rate decreases.

increased redundancy $\Rightarrow$ lower ~~BW~~ efficiency!

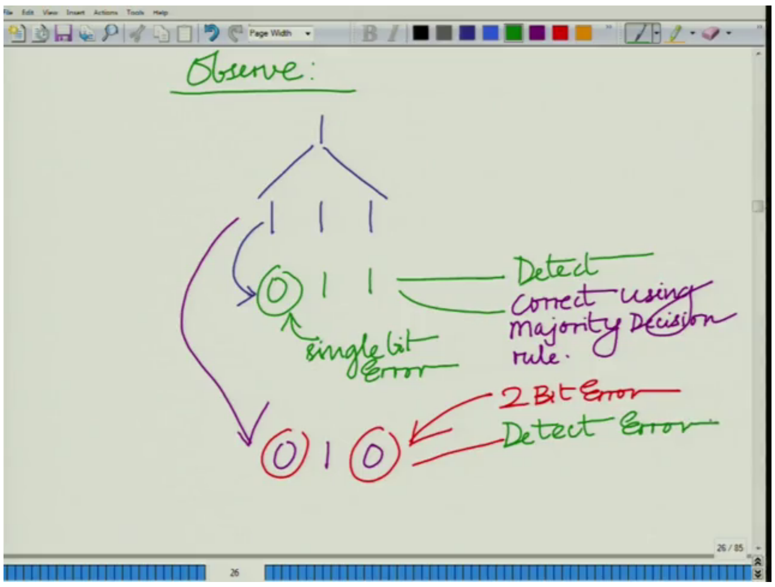
You can see what is this, this is the number of information bits divided by the number of output code bits, this is equal to the rate. This is equal to the rate of the code number of information bits divided by the number of code bits, this is equal to the rate of the code.

Also observe that this is an important aspect code bit that is it shows that at what rate are we introducing the redundancy we remember we have to introduced re redundancy. So, for e information bit there is certain larger number of code bits that is generated that is the meaning of redundancy right. So, this rate is basically shows as what is the number of code bits number of redundancy, what is the number of redundant bits that are introduced for each information bit all right. So, we simply transmit the information bit there is a maximum possible rate is 1, as the redundancy increases the code rate decrease; obviously, the number of code bits keeps increasing redundancy keeps increasing therefore, rate of the code keeps decreasing ok.

So, rate characterizes the redundancy. So, as redundancy, so as the redundancy increases right the rate decreases, this is something that you have keep in mind so; obviously, the best codeword give the maximum possible error recovery or the best possible error protection for the minimum redundancy that is lowest possible redundancy best possible error protection that would be an efficient code the; obviously, if we keep increasing the redundancy without bound. So, that is really the error protection right for instance if you keep repeating a 5 times 10 times and so on as the rate decreases the error protection increase, but we do not want that. We want to have reliable degree of error protection at the same time limit the redundancy because the redundancy decreases the bandwidth efficiency of the system. Observe when your repeating a bit 3 times or 5 times. So, let us say the same bit right the same bandwidth which could have been use for transmission of other information bits is now use for transmission of the redundants bit ok.

So, increase redundancy reduces the bandwidth efficiency of the system that is an important point. Increase redundancy implies lower bandwidth efficiency that is something that one has to give you. Now also observe, observe also that for instance you have this 1 whichever repeating 3 times or Let me write it this way.

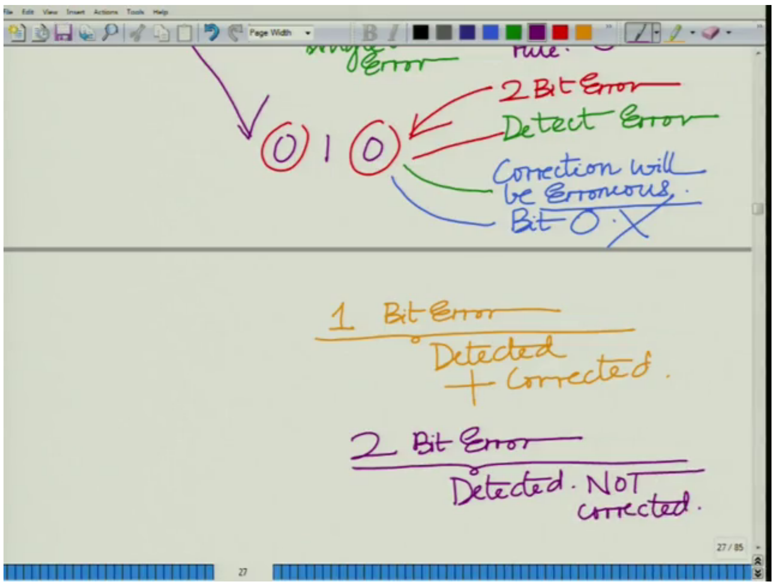
(Refer Slide Time: 23:31)



This one which has been repeated 3 times, now if there is a single bit in error for instance you have 0, 1, 1. The single bit in error you can detect it because the moment you see 0, 1, 1 I can we will for transmitted sequence 3 bits sequence can other be 1, 1, 1 corresponding to repetition of what or 0 0 0 moment you see 0, 1, 1 you can detect an error. So, you can detect in not only that using the majority rule you can corrected because 0, 1, 1 using the majority logic this will be bit 1 because the majority is 1, 2 1s and 1 0.

So, you can also correct it using the majority logic flow, you can also correct this using the majority; however, let us say if you receive 2 bit flips 0 1 0. There are 2 bits in error, you can only detect this. You know, that something is wrong because the moment you see 0 0 0 1 0 you know that there has an error has occurred, but if you try to corrected using the majority logic decision rule you will end up with 0 because there are 2 0s 1 1.

(Refer Slide Time: 25:51)



Error

rule.

0 1 0

2 Bit Error

Detect Error

Correction will be Erroneous.

Bit 0. X

1 Bit Error

Detected + Corrected.

2 Bit Error

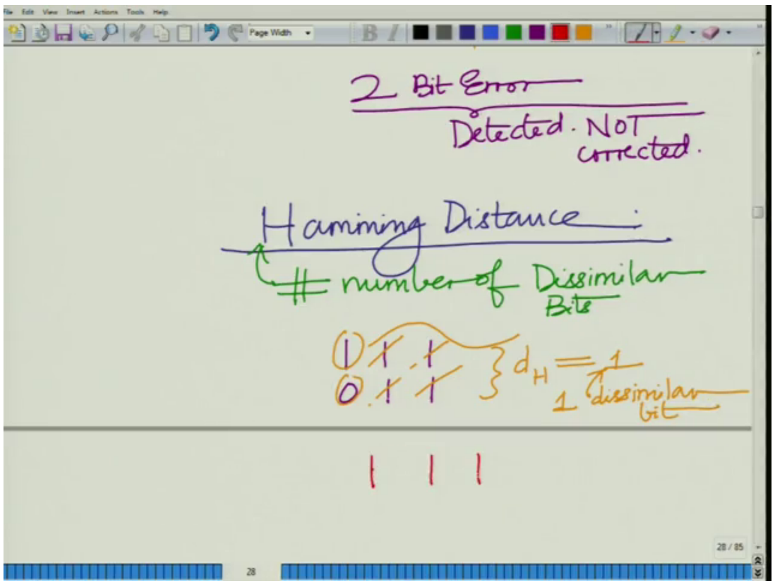
Detected. NOT corrected.

So, correction not possible or rather correction will be erroneous this will be corrected to bit 0, correct? Which is basically wrong using correction will be erroneous that is the whole point because it will be bit 0 which is correct. So, 2 bit error so, the point is that I want to drive is one bit error can be detected and corrected.

So, single bit error. So, if you look at this repetition code the larger point that I want to have is, larger point I want to make is that 1 bit error this is both detected plus corrected; however, if you have 2 bit error this is only detected, but not corrected. So, this is only detected not detected, but not corrected. So, every code has an error correction capability has an error detection capability. And if you look at here the number of bit errors you can detect is larger than, than number of bit errors that can; obviously, if you bit error can be corrected then it is necessary to be detected you can corrected only if you detected correct, but if you detected it might not be necessary that you will be able to corrected. So, that is the distinct we will talk more about these things later correct.

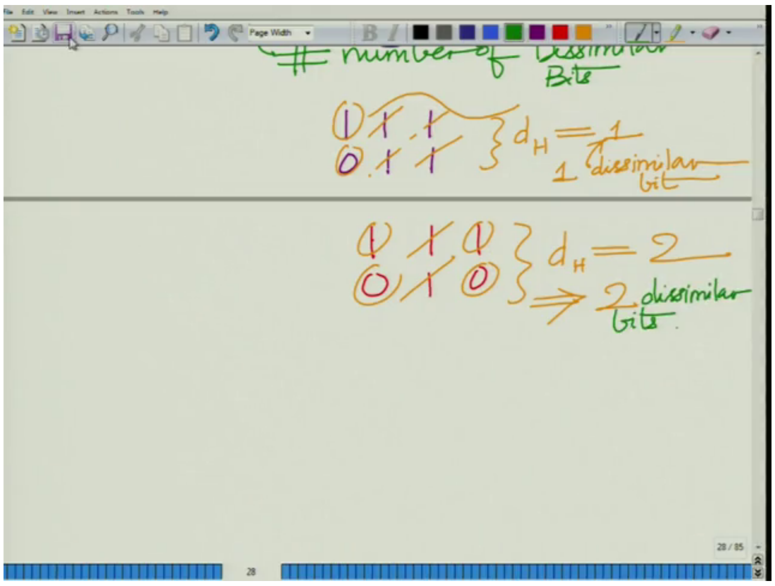
As we progress. So, several other modules in this course, but there are important point to keep in mind is that every code has a bit detection capability, has a bit correction capability. And if you look at the hamming distance between the codewords and another bit characterize this is by what is known as the hamming distance.

(Refer Slide Time: 28:02)



The hamming distance is simply number of dissimilar bits. For instance if you look at our 0 our transmitted code bit 1, 1, 1 1 bit error received 1 is 0. So, there is a one bit difference. So, these are same 1 1 is same this is different implies the hamming distance d H equals 1, that is basically 1 the hamming distance between the transmitted and received word is 2 1. On the other hand if you look at the second one that is 1, 1, 1 and 0 1 0, 1, 1 1.

(Refer Slide Time: 29:10)



And if you look at the second one that is 0 1, 0, 1, 0, now of course, these 2 are identical these are different.

So, hamming distance equals 2 this basically implies 2 different or 2 bits, 2 dissimilar bits let us put it that rate 2 bits differ basically 2 dissimilar this is 2 dissimilar bit. So, the hamming distance. So, the hamming distance between. So, what this says is in terms of hamming distance if there are 2 bits in error what is the hamming distance is 2 then you can detected, but not corrected if there is one dissimilar bit then you can both the detected as well as corrected, this much that is the very high level over view of the concept of channel coding and the terminology. What we have looked at is a very simple channel code there is a repetition code all though it is a code in the strict sense of the world, but it is very inefficient code. What we will see is there are much better and much more efficient codes.

Much more efficient in the sense that is have a very low probability of error as well as very low redundancy, that is give the best possible, this is the lowest possible probability of error for the least possible redundancy. As we said this is metric to characterize the efficiency of the code and we will see one such code or one such class of codes in subsequent models which is termed as convoloutional codes, that is where we are adding towards. So, we will stop here.

Thank you very much.