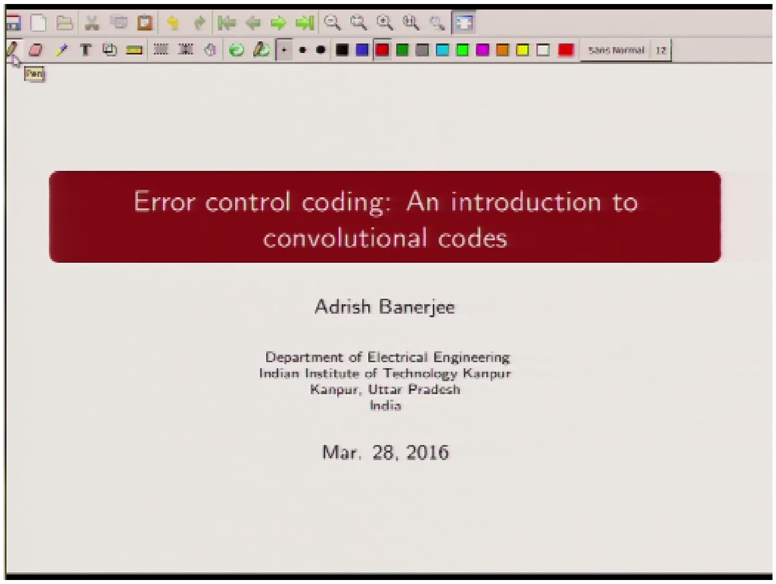
Indian Institute of Technology Kanpur
National Programme on Technology Enhanced Learning (NPTEL)
Course Title
Error Control Coding: An Introduction to Convolutional Codes

Lecture-6
Performance Bound for Convolutional Codes

by
Prof. Adrish Banerjee
Dept. Electrical Engineering, IIT Kanpur

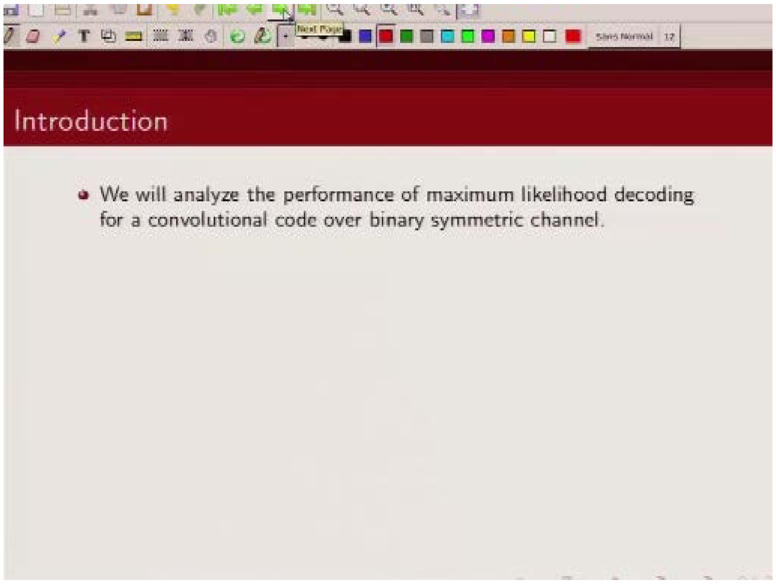
Welcome to the course on error control coding, an introduction to convolutional code.

(Refer Slide Time: 00:20)



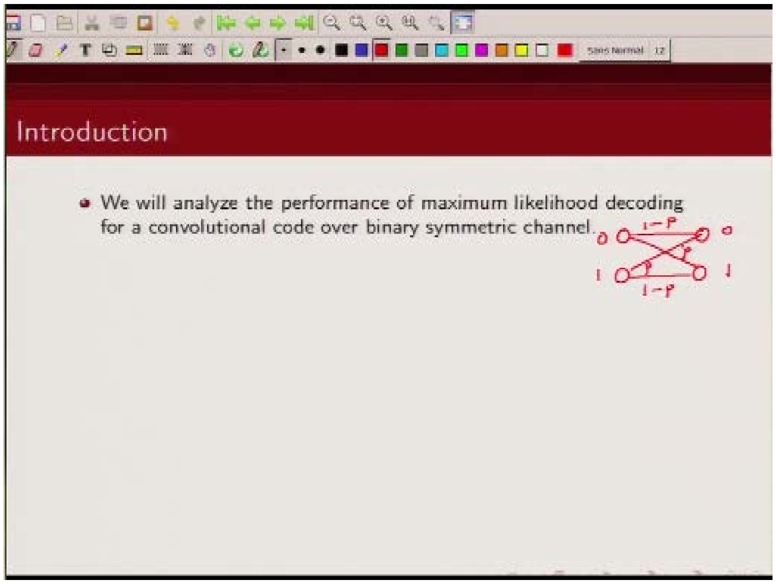
So in this lecture we are going to talk about performance bounce for convolutional codes.

(Refer Slide Time: 00:28)



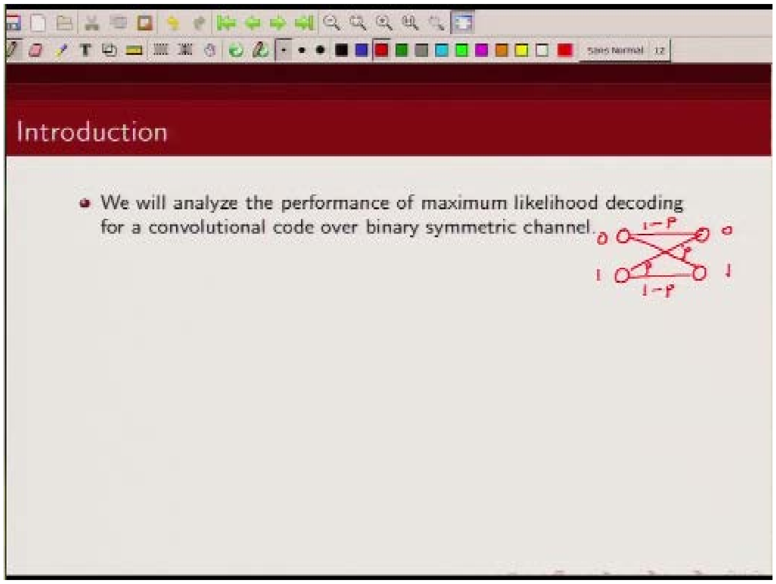
In particular we are going to analyze the performance of viterbi decoding which is maximum likelihood decoding for convolutional code. And as an example we will consider a simple binary symmetric channel. Now if you recall what is a binary symmetric channel?

(Refer Slide Time: 00:46)



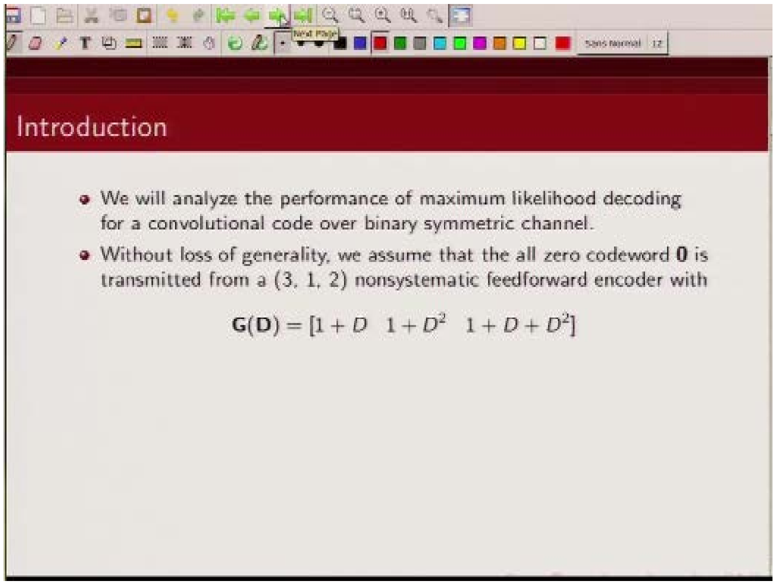
So there are two inputs 0 and 1, 0 and 1 and with some probability 1-P we receive the bits correctly and with the crossover probability of P the bits get flipped, that is our binary symmetric channel. So we are going to analyze the probability of error, a probability of error.

(Refer Slide Time: 01:17)



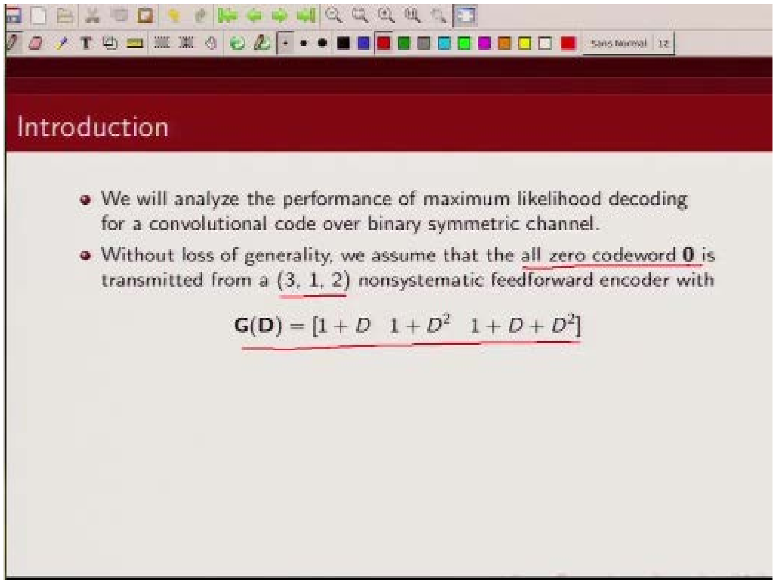
When we are doing Viterbi decoding of convolutional code over a binary symmetric channel.

(Refer Slide Time: 01:23)



So as an example we will consider one particular convolutional code.

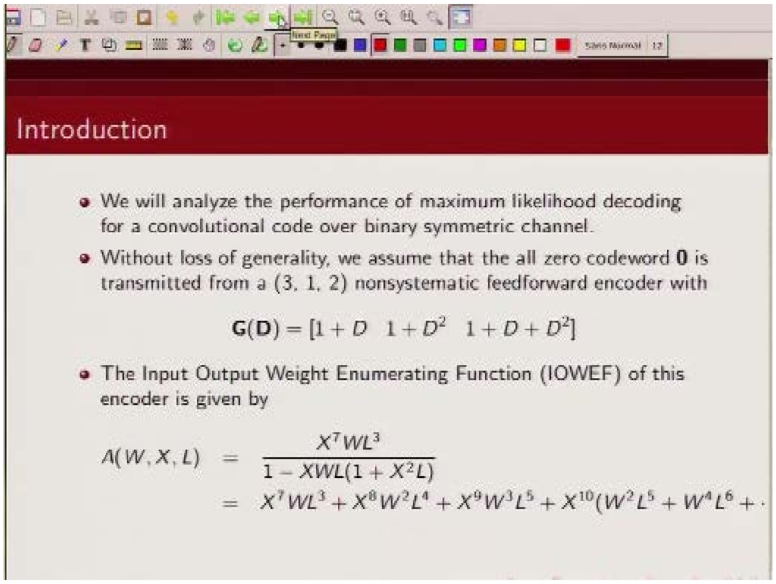
(Refer Slide Time: 01:35)



In this case we are considering a rate 1/3 convolutional code which has memory 2, so it is a four state convolutional code whose generator matrix is basically given by this. So these are my generators 1+D $1+D^2$ and $1+D+D^2$. And without loss of generality we will assume that all zero codeword was transmitted.

So all zero codeword is transmitted over a binary symmetric channel and we receive the bit, we applied maximum likelihood decoding, Viterbi decoding, and now we are interested to find, characterize a performance of the Viterbi decoding algorithm.

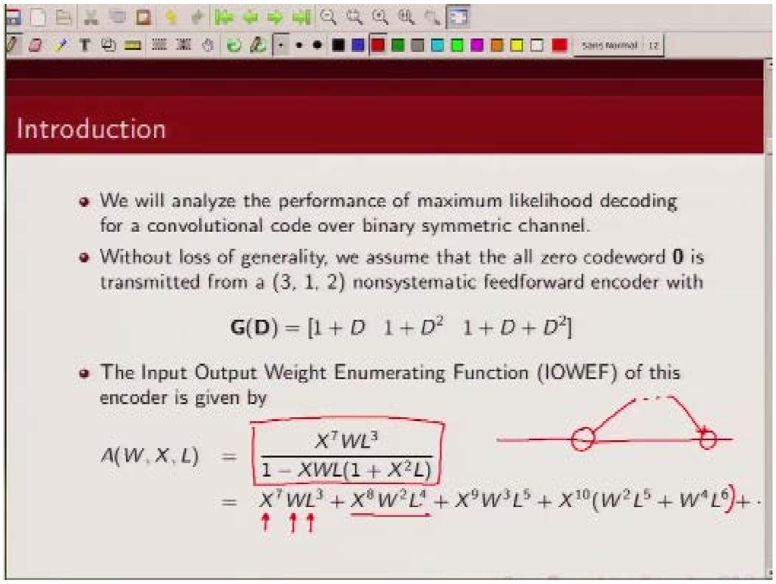
(Refer Slide Time: 02:17)



Now we would require one more concept which we discussed in lecture 2C which is input output weight enumerating function. So what does input output weight enumerating function tells us, it tells us about what is a input that will cause, or what is the corresponding output that it will cause and what is the length of that particular sequence.

So as you can see these all zero parts through the trellis are all valid code words. So this weight enumerating function essentially enumerates all nonzero code words.

(Refer Slide Time: 02:59)

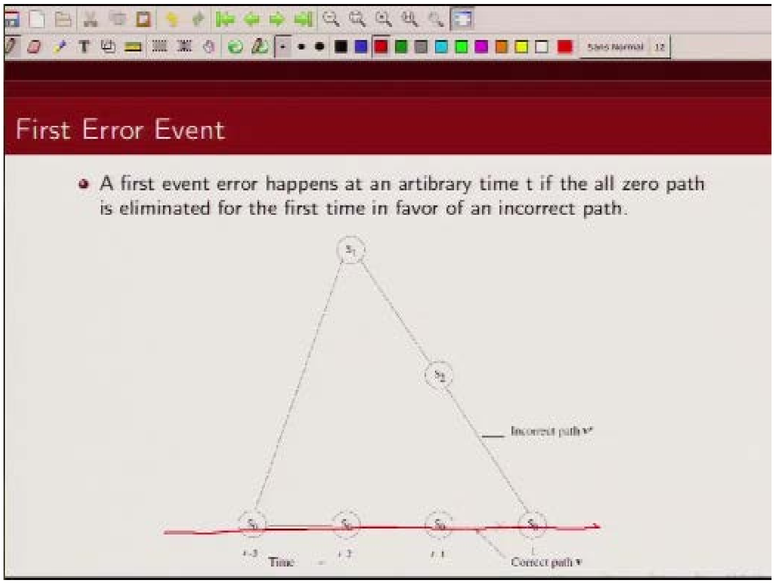


So for this particular example the input output weight enumerating function is given by this expression and which we can expand, we can divide this by this. So this will – we will get our terms. Now here X is my, the exponent of X will give me the overall weight, the exponent of W will give me the information sequence weight and this length will give me the length of this path like of this.

So the time weight diverges from all zero state to the time, it demerges at all zero state. So X^7WL^3 means there exist a code word of distance 7 which has information weight 1 and the time it diverges from all zero state to the time it merges back to all zero state that is 3. And there is one such code word. This can be interpreted as there is a code word of weight 8 and corresponding input weight is 2 and it has length 4, length 4 meaning again by this L denotes.

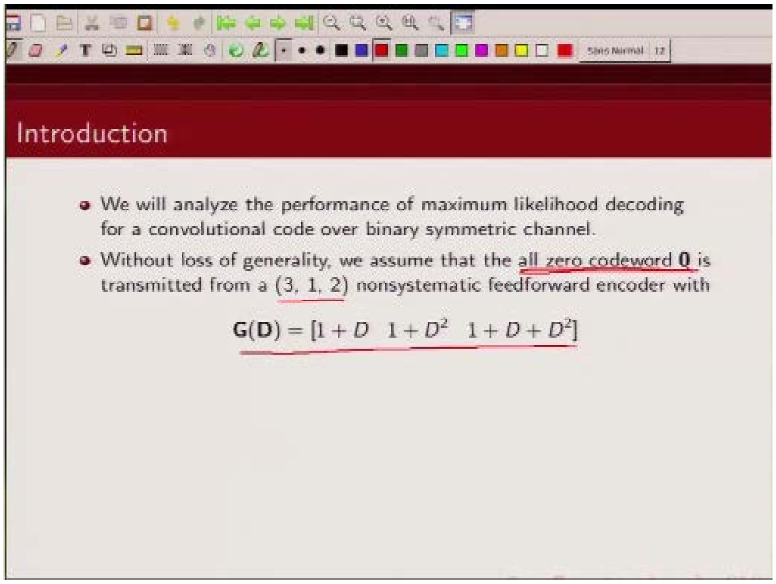
So you have your code word which diverging from all zero state and then after staying in some nonzero state it is merging back into your all zero state. So this L denotes the time from which it moves away from all zero state to the time it takes to come back to all zero state.

(Refer Slide Time: 04:56)



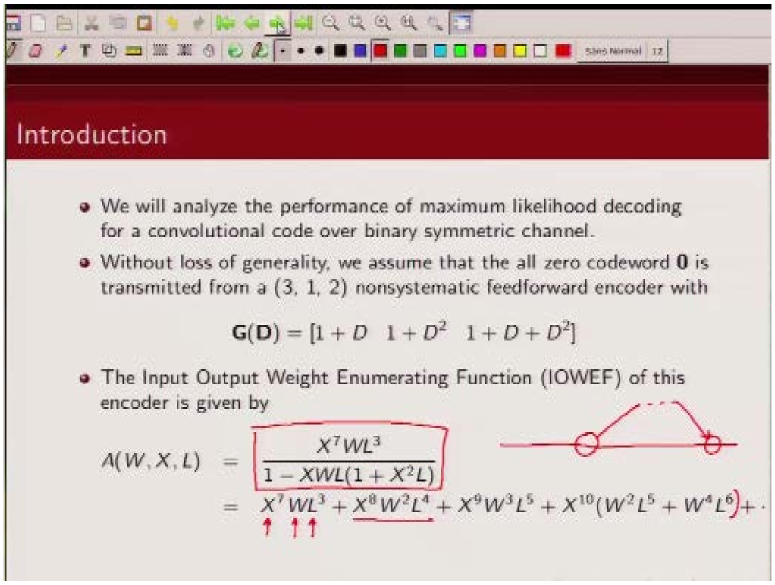
So let us talk about, what do we mean by error here. So since we considered an all zero sequence we expect that our correct path should be the one which goes through all zero state, because our transmitted code sequence that we assumed.

(Refer Slide Time: 05:17)



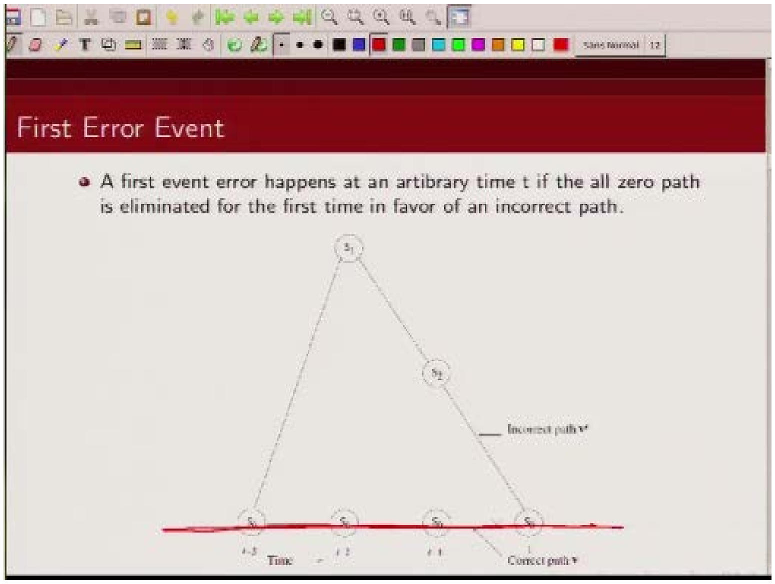
If you recall we assumed that an all zero code word was transmitted.

(Refer Slide Time: 05:25)



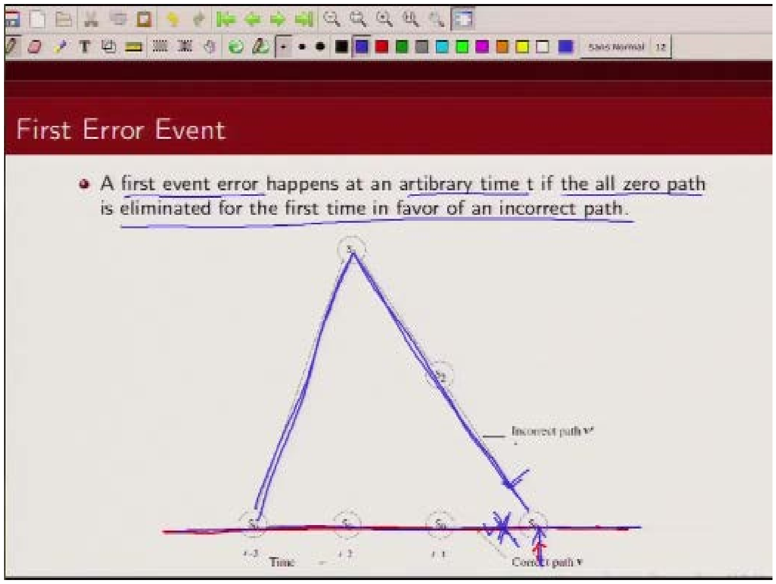
So what is the correct path?

(Refer Slide Time: 05:26)



The correct path is 1 that goes through all zero state, so this is the correct path.

(Refer Slide Time: 05:40)

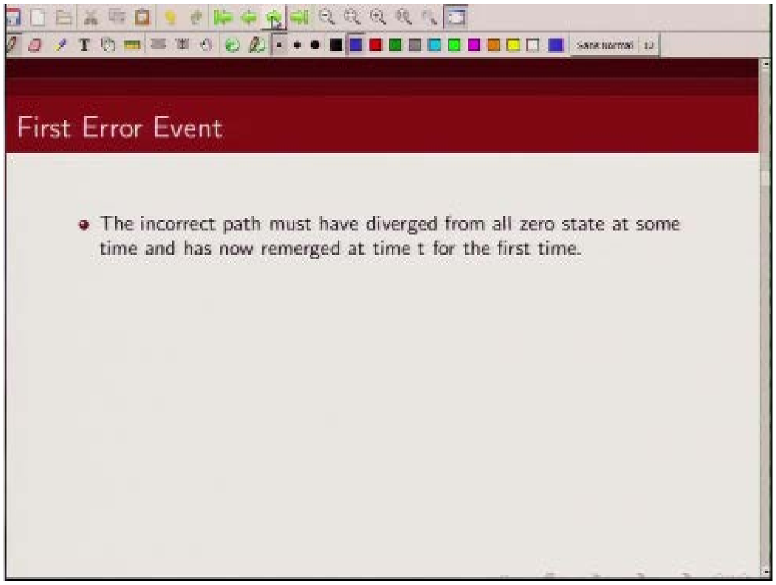


Now when will error happen, an error will happen if at a particular time instance let us just say this is the time instance I am looking at. If at this time instance instead of deciding for this path, instead of deciding for this path if I go for this path, then error will happen. Why? Why is this an incorrect path, why is this an incorrect path?

This is an incorrect path, because my transmitted code word was all zero sequence. So if I decide in favor of this path instead of this path, then I make an error. So I say a first event error happens at some arbitrary time t, if the all zero path is eliminated in favor of a nonzero path which is our incorrect path.

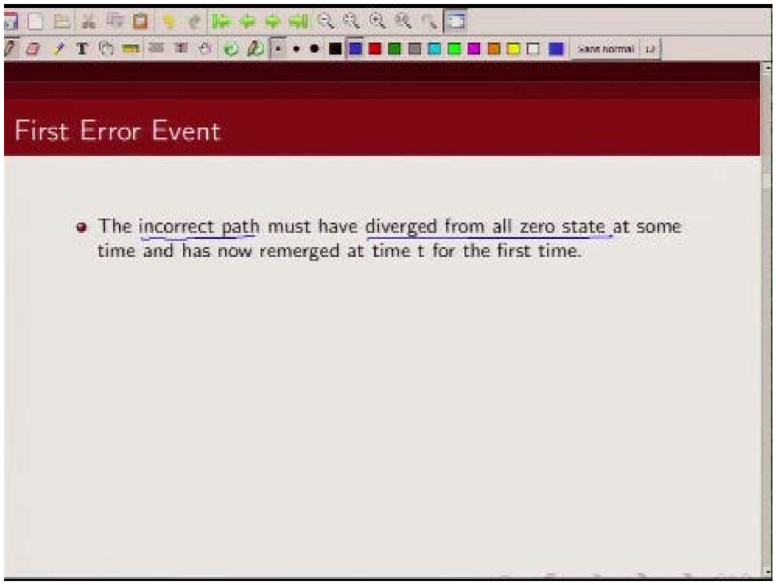
So the first instance when I do that, so the instance when I do that, that is basically my event error, because I should have chosen all zero path, but at this instance what happened was, the metric corresponding to this path was better, so I chose this and I eliminated this path. But this is a wrong path, because my transmitted code word was all zero sequence.

(Refer Slide Time: 07:01)



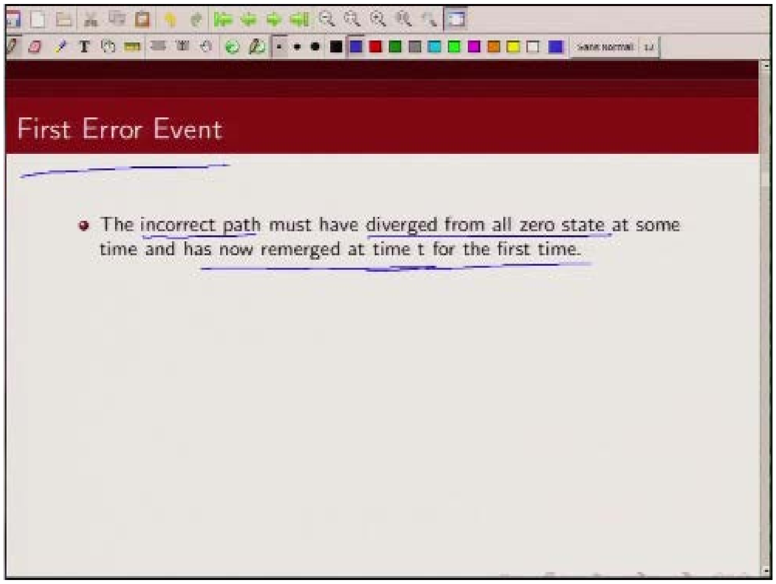
Now what does in, we know that all paths through this trellis are essentially valid code word, now what does this

(Refer Slide Time: 07:12)



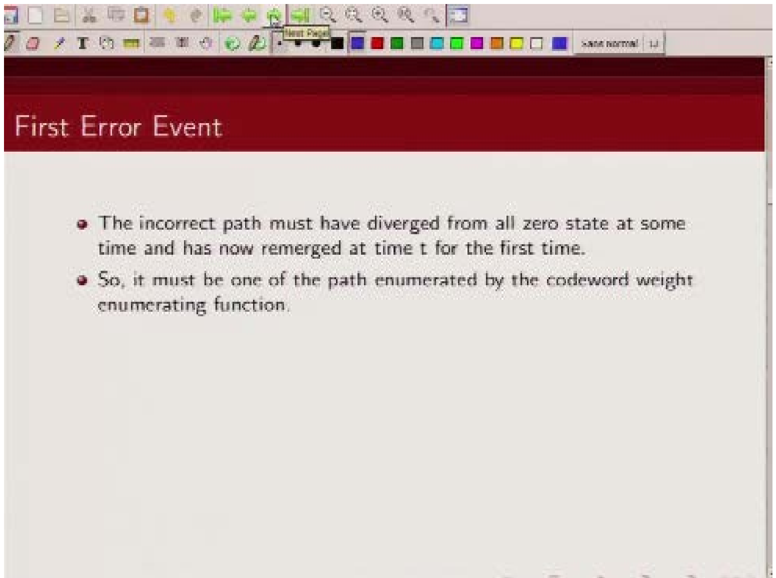
Incorrect path does it diverges from an all zero state at some other time before this time t and has merged back

(Refer Slide Time: 07:12)



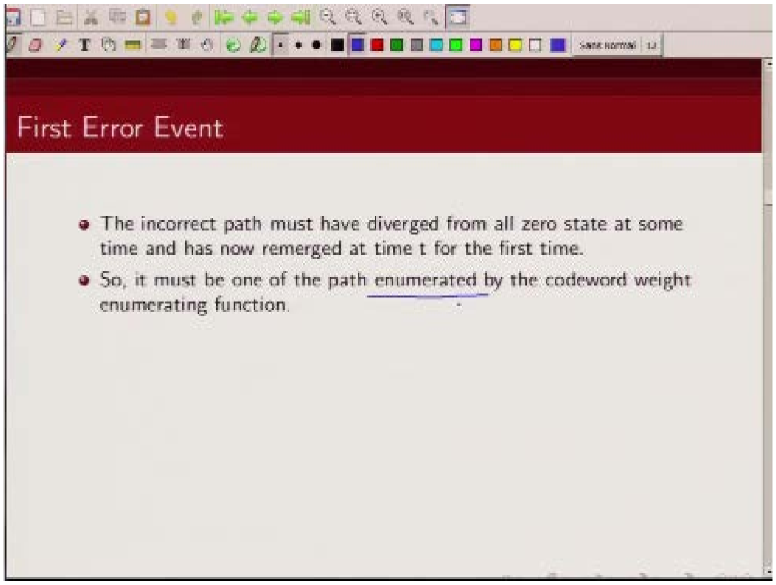
Into all zero state at time t for the first time so that is what we call first error event, so at the time t when we are deciding in favor of some other path other then the all zero path so that means this is the time when this other path the incorrect path has merged into all zero state, so at some time in the past it would have diverged from all zero state and now at time t it is merging back into all zero state.

(Refer Slide Time: 07:58)



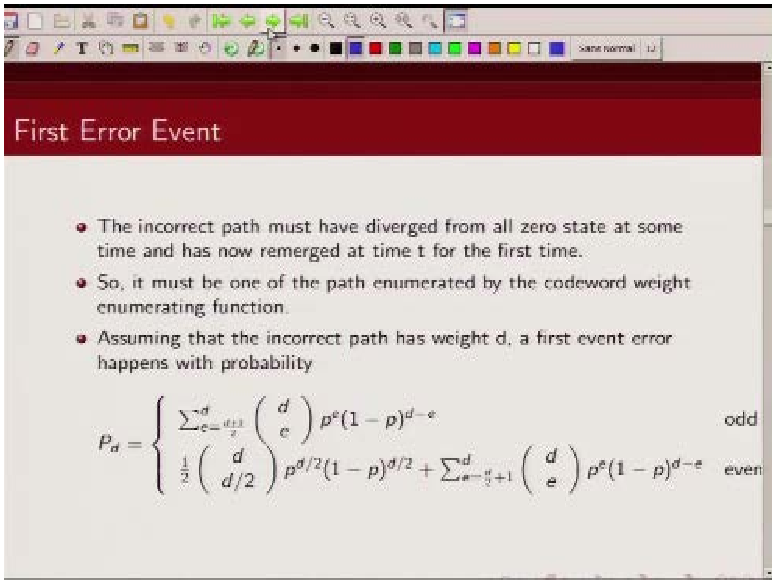
Now can we find out what is the weigh corresponding to this incorrect path, yes we can because when we write the weight enumerating function of a convolutional code it enumerates weight distribution of all valid code word. Now any path through, through this trellis diagram is a valid code word so we can

(Refer Slide Time: 08:28)



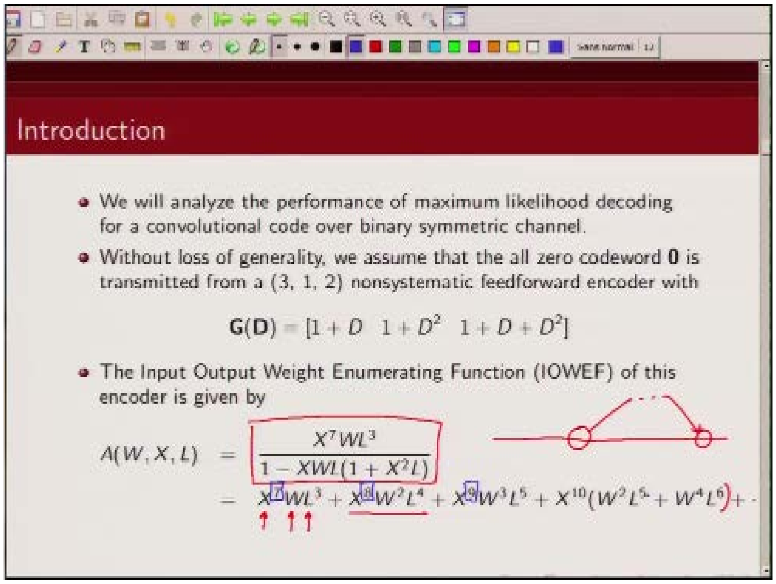
Easily enumerate what is a code weight corresponding to any path through this trellis.

(Refer Slide Time: 08:36)



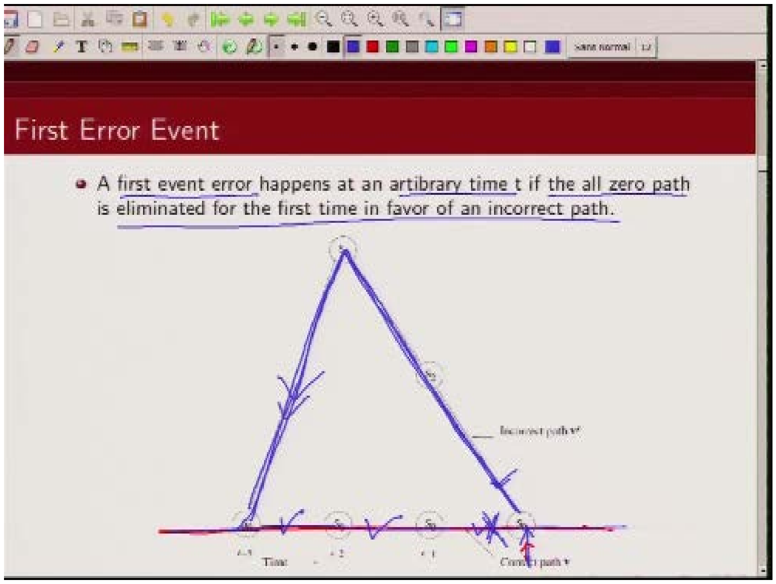
Now when will we decide in favor of this incorrect path and not the all zero path? We can take this example of the same code that we are considering.

(Refer Slide Time: 08:50)



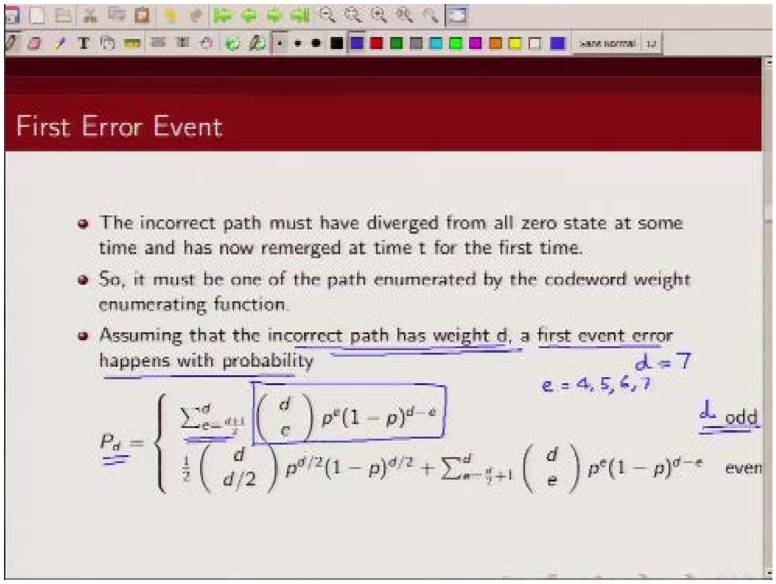
Let us just this code has dfree seven, then it has one code word of weight eight, one code of weight nine, there are two code words of weight ten like that, so minimum distance dfree is seven here so let us say

(Refer Slide Time: 09:07)



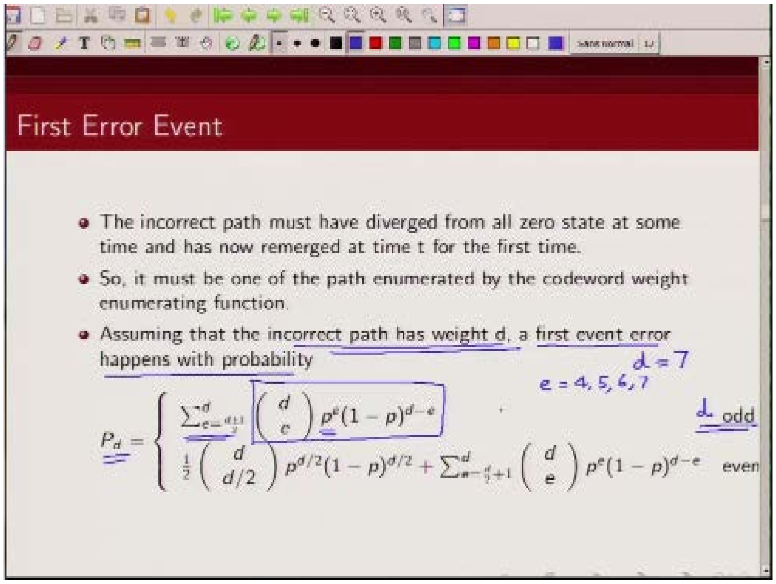
Let us assume that this incorrect path had weight seven, now when will you decide in favor of this path rather than this path? When the number of errors are such that when they are more than four errors such that your receiver sequence is closer to this sequence rather than this right then you will make an error or let us say if this path was weight eight then of course when your error is more than four bits you will decide in favor of this path, and whenever error has happened in four bit location there is a equal probability you can either choose this path or you can choose this path.

(Refer Slide Time: 09:53)



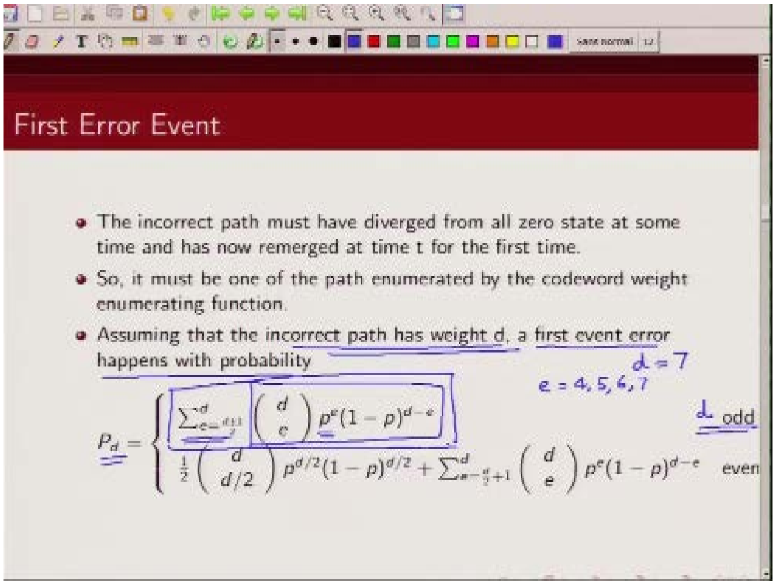
So we can calculate the probability of this first event error, we can find out its probability, what is this probability let us say this incorrect path has weight d so d is the weight of this incorrect path which has diverged from all zero state at some other time and has merged back at all zero state at time t. Now if d is odd then this probability is given by sum of this probability and where Σ takes place over all error pattern which are greater than (d +1/ 2)^{d} so in the example we just considered let us say if we had a incorrect path of weight seven then we should look at error pattern of 4, 5, 6 and 7. If 4 errors happen maybe it will bring us closer to this incorrect path rather than all zero path okay? Now if

(Refer Slide Time: 11:18)



P is the cross over probability of changing, flipping of the bits, so P times e will give us a probability that e number of bits have flipped and

(Refer Slide Time: 11:18)



1- P^{d-c} will tell us the probability that d-e bits have not flipped and d e d choose e is number of possible combinations of in which we can have these error patterns, e error patterns among these set of d weight so this will give us the probability of first event error. Again we will make a decision in favor of incorrect path if our received sequence is closer to the incorrect path and we know what is the maximum likely would role for binary symmetric channel, it is we choose a code word such that the hamming distance between the received code word and this chosen cord word is minimized so

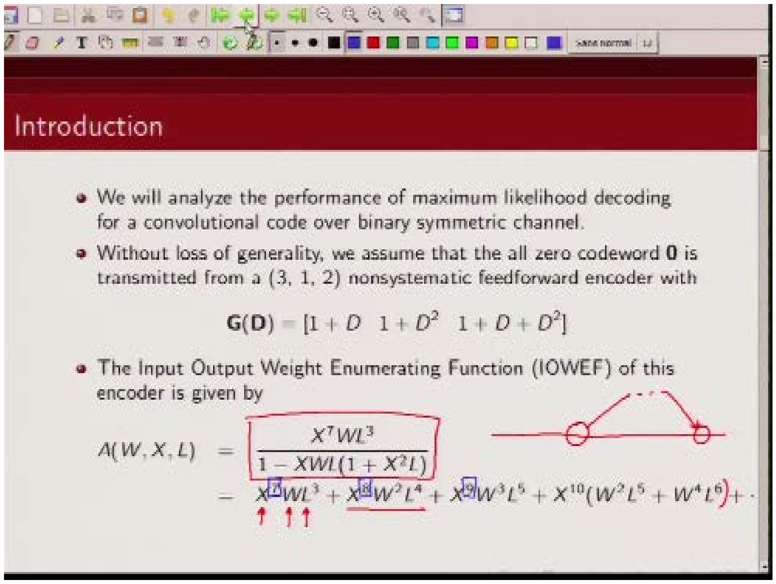
(Refer Slide Time: 11:18)

If there are more than, there are four or more errors for the case when incorrect path has weight seven then you are going to choose, basically go for this incorrect path. What about when d is even when d is even for all error patterns from $d/2+ 1$ to d you will decide in favor of incorrect path and whenever the error is

(Refer Slide Time: 11:18)

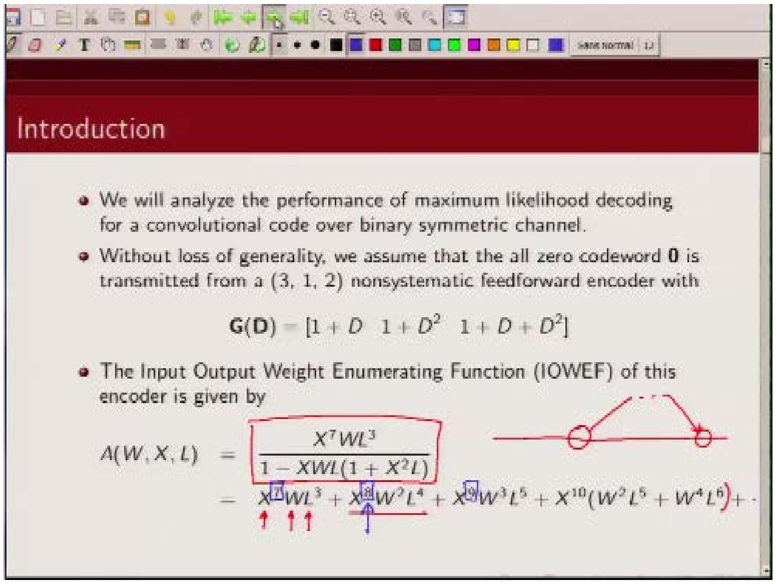
d/2 then there is a 50/50 chance because you can just then the, there is a tie in the matrix so you can just flip and choose either of the path and that is why.

(Refer Slide Time: 13:09)



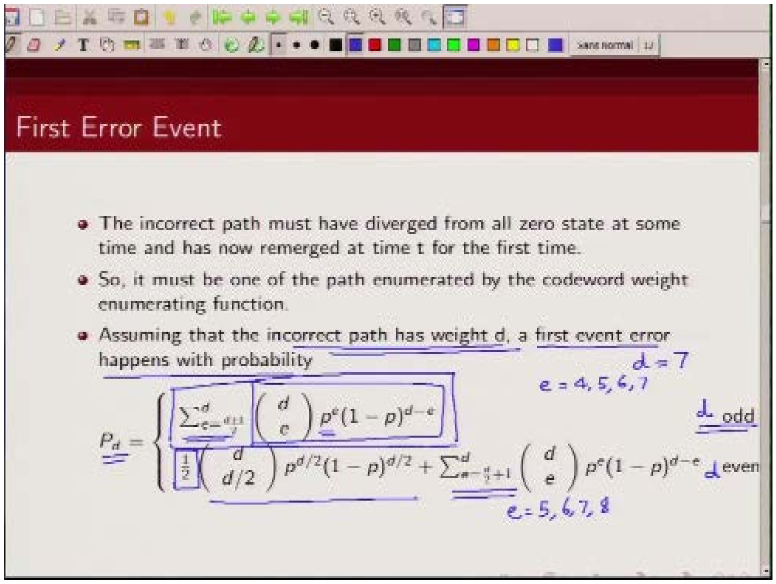
For d/2 I have written here it is half times this probability plus all error patterns of weight more than d/2. So for example in this case we had one path

(Refer Slide Time: 13:27)



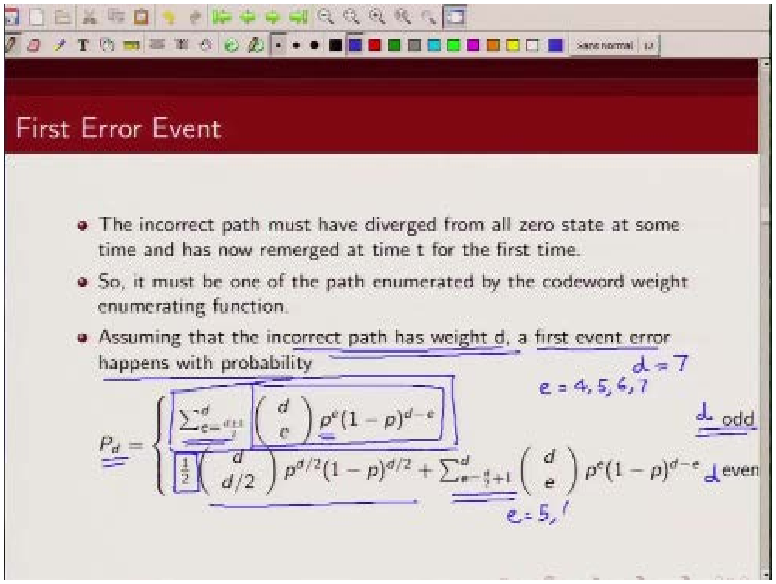
Of weight eight right so if our incorrect path was of weight eight

(Refer Slide Time: 13:27)



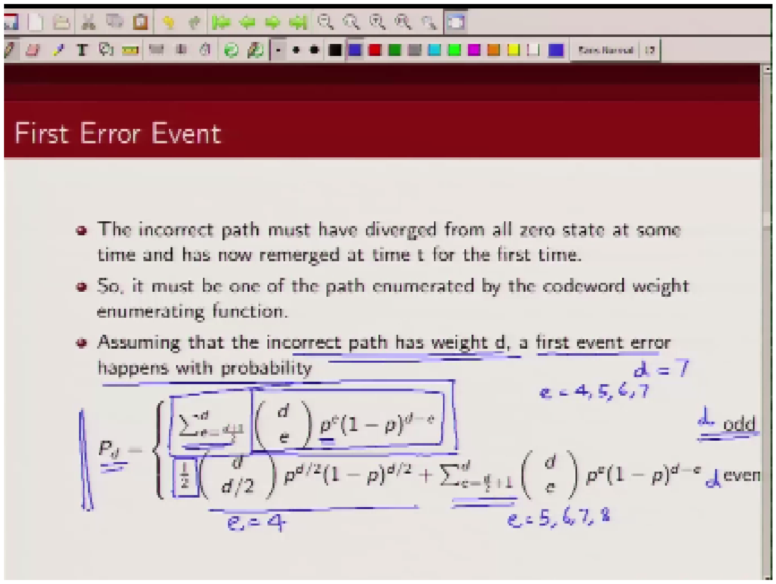
Then all error patterns of 5, 6, 7, 8 they would have cost decision in favor of this incorrect path and whenever there is an error pattern of.

(Refer Slide Time: 13:52)



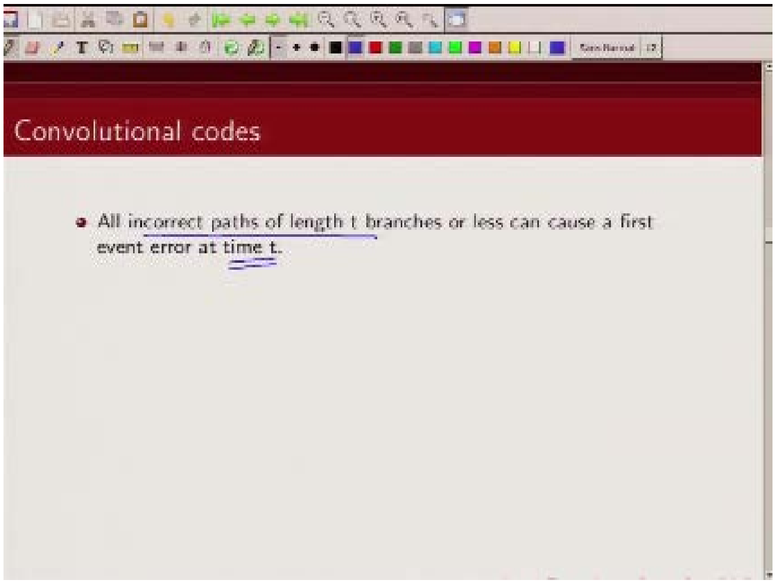
Weight four there is a 50/50 probability that I may choose an incorrect path or I may choose a all zero path because the hamming distance is same from all zero path or this incorrect path

(Refer Slide Time 14:09)



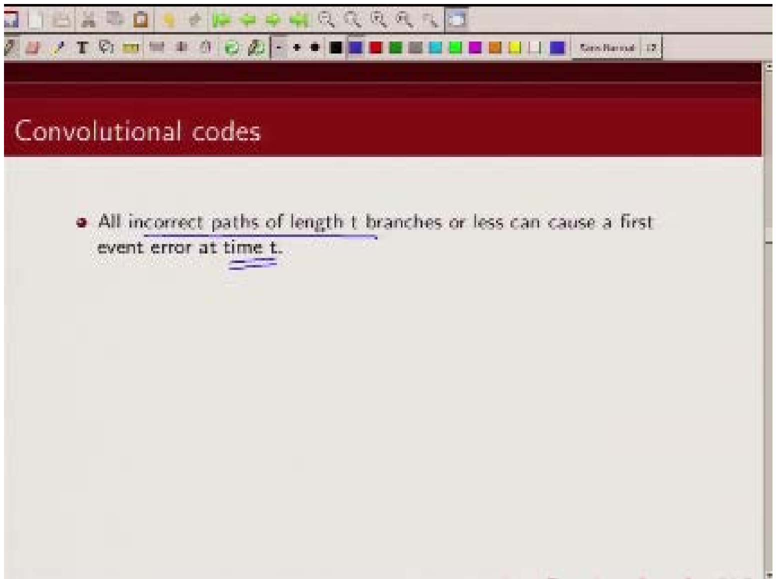
So that is why I have a half here, so this is the probability of first event error.

(Refer Slide Time 14:17)

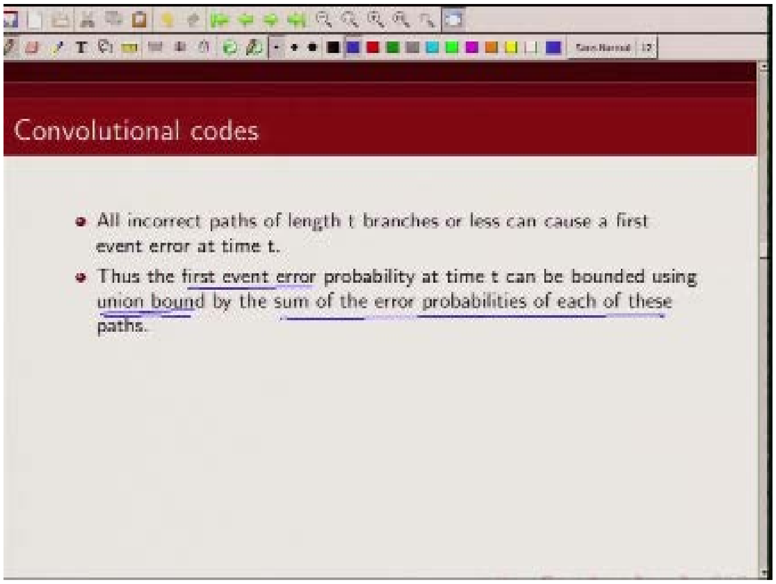


Now if we are looking at time t then all incorrect path lengths of length t or less can cause first event error.

(Refer Slide Time 14:29)

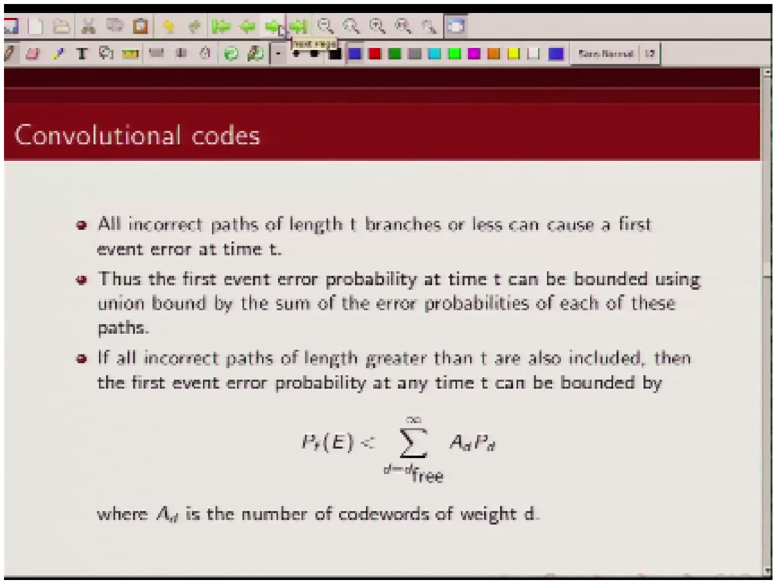


(Refer Slide Time 14:31)



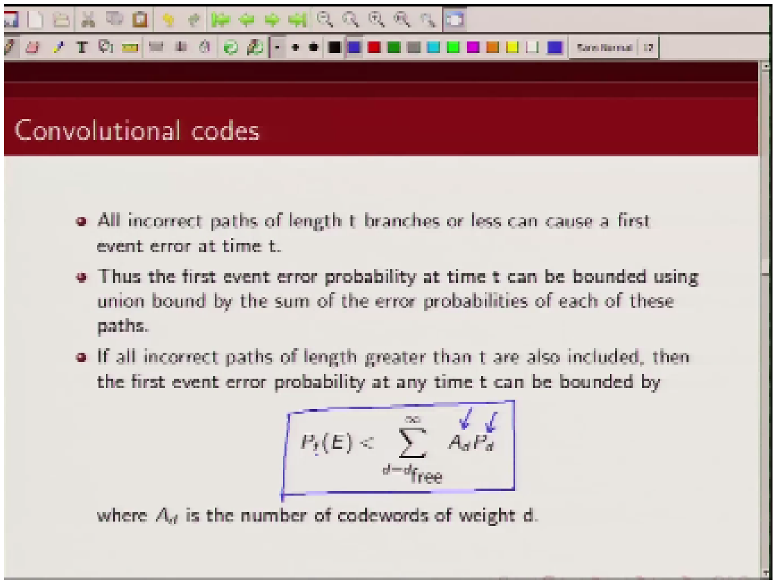
So we will now use union bound to essentially upper bound this probability of first event error probability. So what does union bound says probability of union of errors is basically upper bounded by union of probability some error so we are going to upper bound this first event error probability by sum of error probabilities of all these incorrect paths.

(Refer Slide Time 15:06)



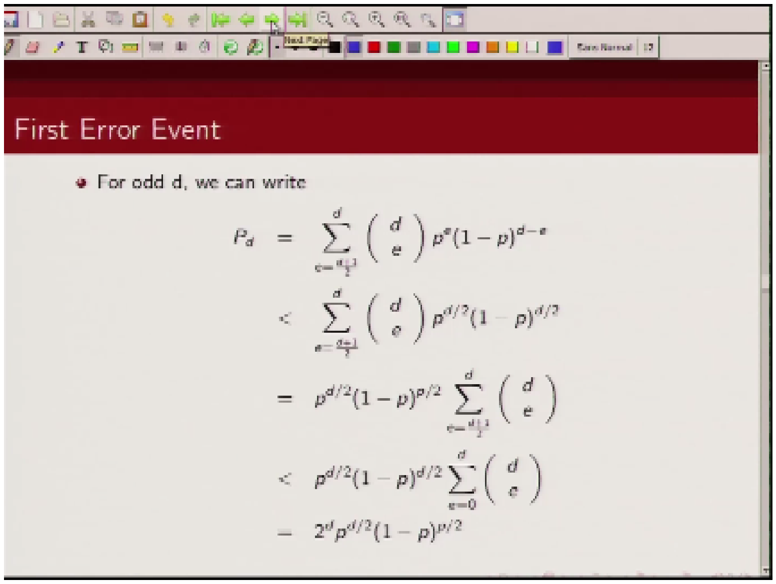
Now if we also allow incorrect path of any length even greater than t.

(Refer Slide Time 15:13)



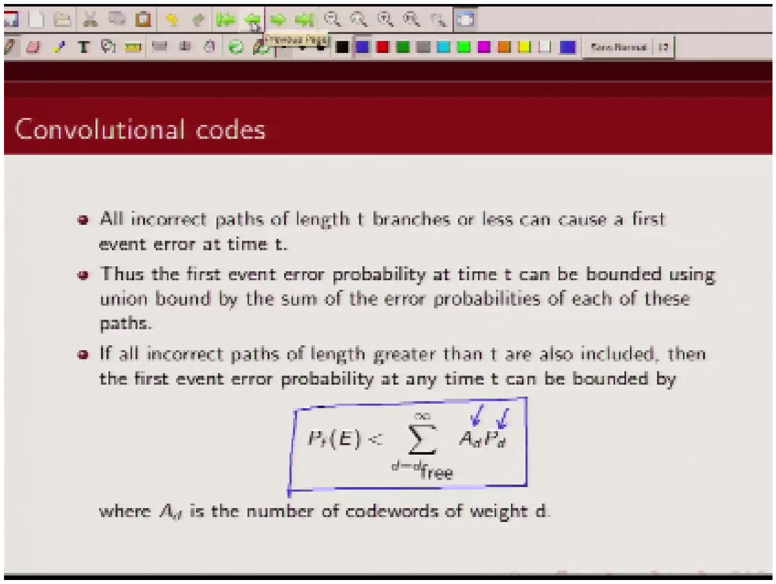
Then in that case the first event error probability can be upper bounded by this, so this is the probability of first event error of weight d and how many such weight D paths exists, that is given by this and you sum a word, your D which goes from free distance of initial code to infinity. So this will give you an upper bound on probability of first error event probability, now can we make use of the weight enumerating function of the convolutional code to calculate this, and this is what

(Refer Slide Time 16:11)

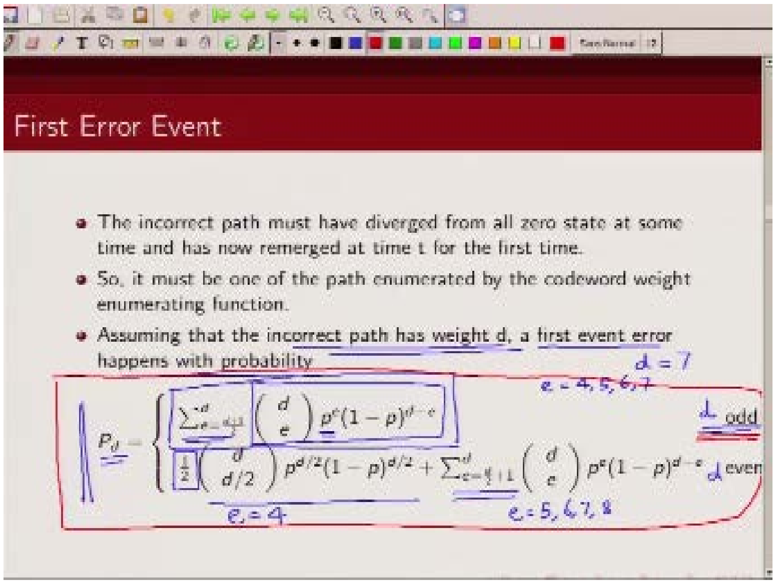


We are going to show so let us first try to simplify the expression for probability of first event error.

(Refer Slide Time 16:20)

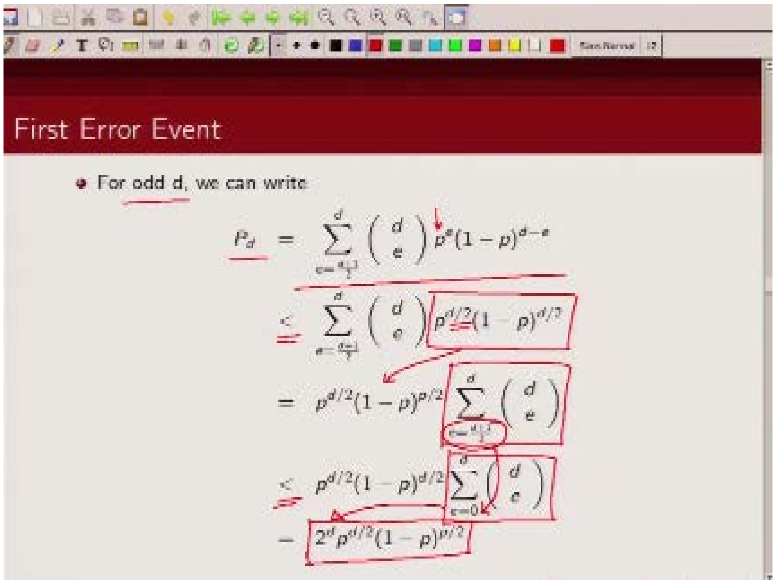


(Refer Slide Time 16:22)



So this was the expression for, this was the expression for first event error probability so let us try to simplify this expression, so we will first do it for when D is odd and then we will do it for when D is even.

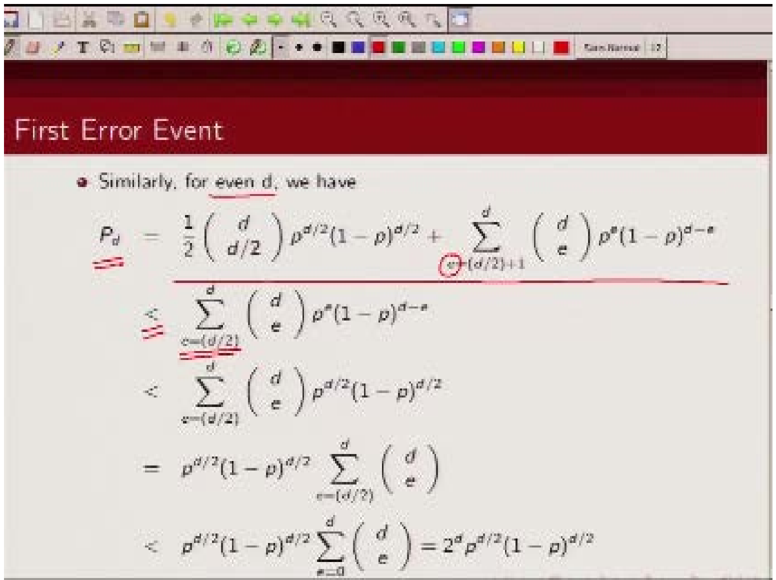
(Refer Slide Time 16:41)



So when D is odd the expression is given by this, now note that this crossover probability is typically small. So this is our, again this is our number between zero and one so if you raise it by a higher number you get smaller quantity. So the first upper bounding that we are doing is instead of raising it to E we are raising it to D/2 and again because this is a number between zero and one, see the actual expression would have been this P raised to power d+1/2 then d+3/2/2 and like that it basically this number will go on decreasing because

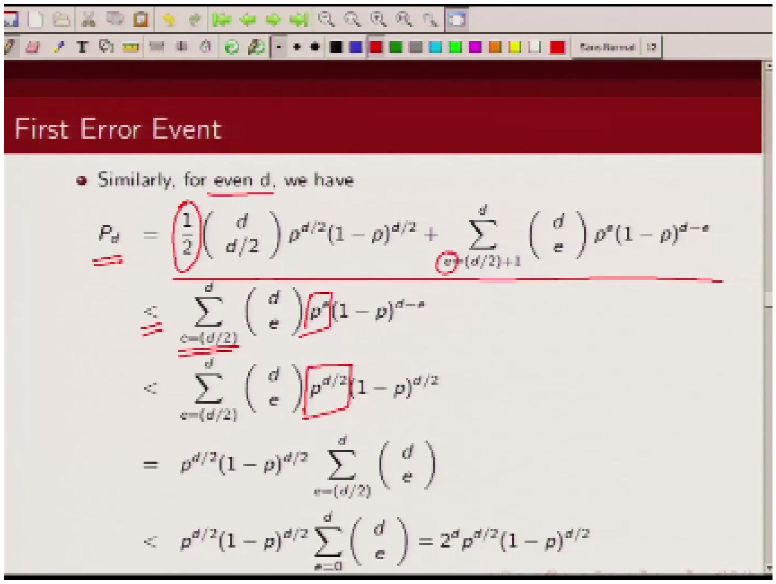
You are raising it to a higher value now I can just fix instead of varying it as fixed I keep it fixed and I kept it as d/2 so that is the first upper bound that we are getting. Next this term does not depend on E so I can take this out, so what I am left with this, this D choose E where E goes from D+1/2, 2D. Now next upper bounding what I did was I replaced it by equal to zero so instead of E= D+1/2 I am adding more terms I added from equal to zero. So that is why this upper bound came and what is this, this is nothing but to this P^d so then for odd D I can write this first event error probability as upper bounded by this quantity.

(Refer Slide Time 18:47)



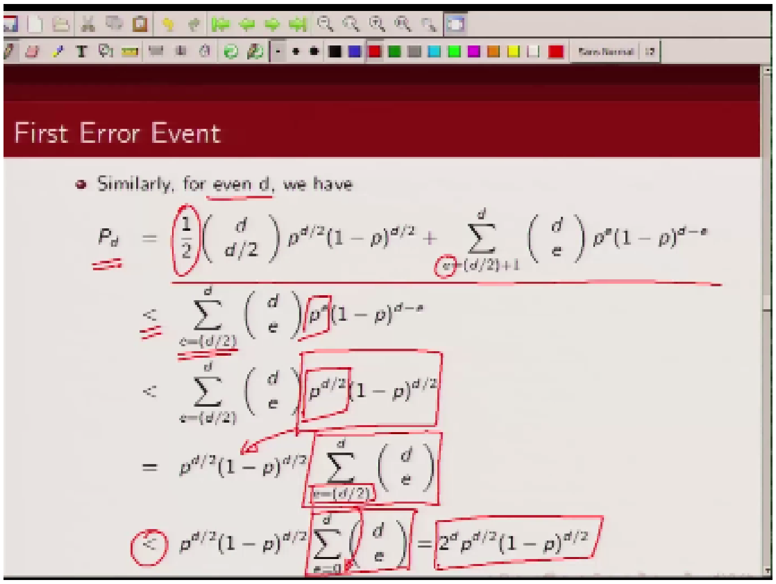
Next let us do the same thing for even D, so for an even D the expression for first event error is given by this expression, so first thing that I do was I upper bound this by changing this exponent from E/2+ one two E/2. Note that in this expression when the error pattern is D/2 I consider a 50% probability of making an error, now I just remove that and I added that here.

(Refer Slide Time 19:27)



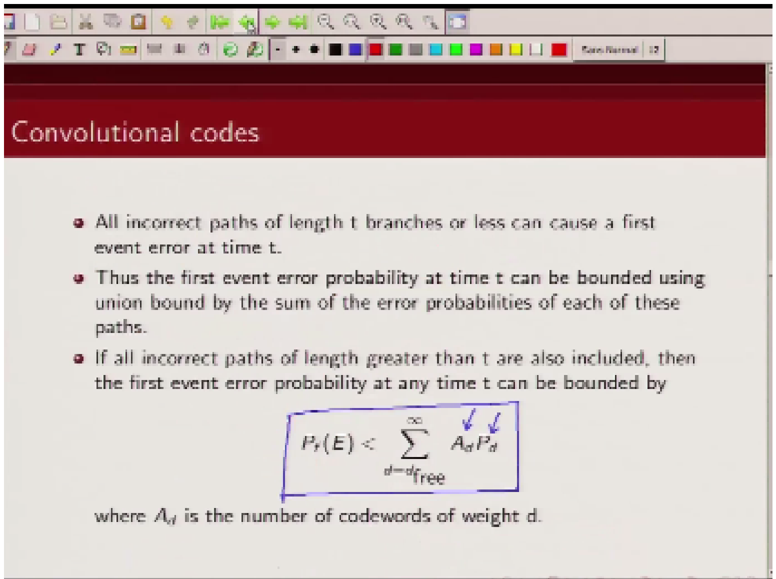
I removed that 50% thing here so that is why I am getting an upper bound here, next same as infuse expression I replace previous power of E by previous power D/2 and again because P is a number between zero and one if you raise it to a higher number it decreases so I just.

(Refer Slide Time 19:46)



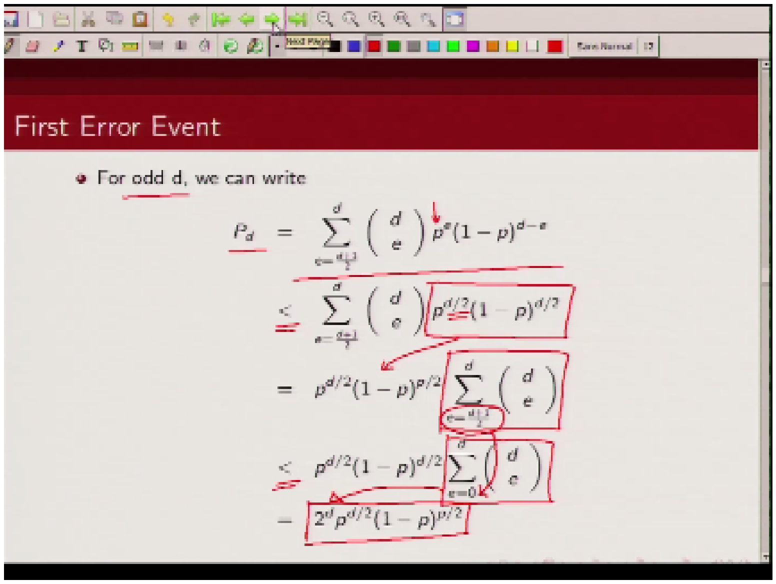
Kept it at D/2 this is the smallest value of E, third thing which I do is again like in for the odd case this term does not depend on E, so I can bring it out so I brought it out and what I am left is with this right and then I add terms from E zero to up to D/2, so I replace this by this so again then I am upper bounding, I am adding additional terms here so this quantity would be less than this quantity, and finally this is 2 is P^d so I get my expression for first event error probability for D being odd given by this expression, so PD is upper bounded by this for when D is even.

(Refer Slide Time 20:50)



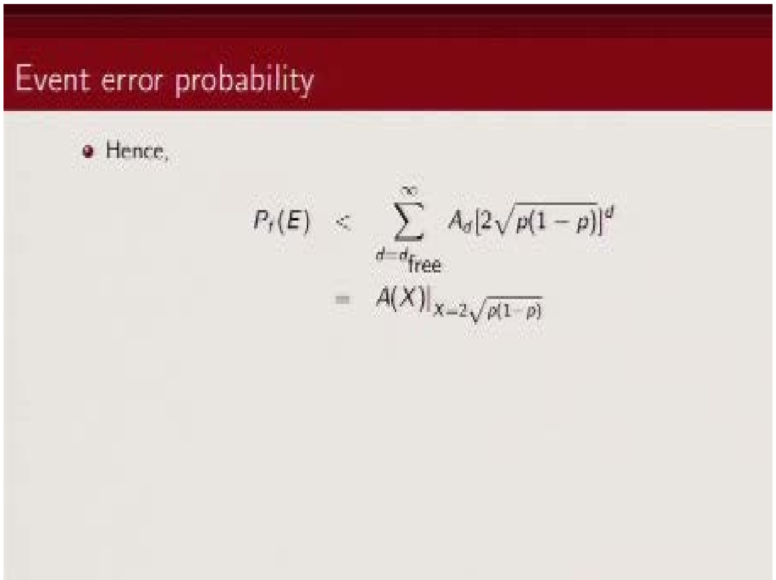
And we showed

(Refer Slide Time 20:51)



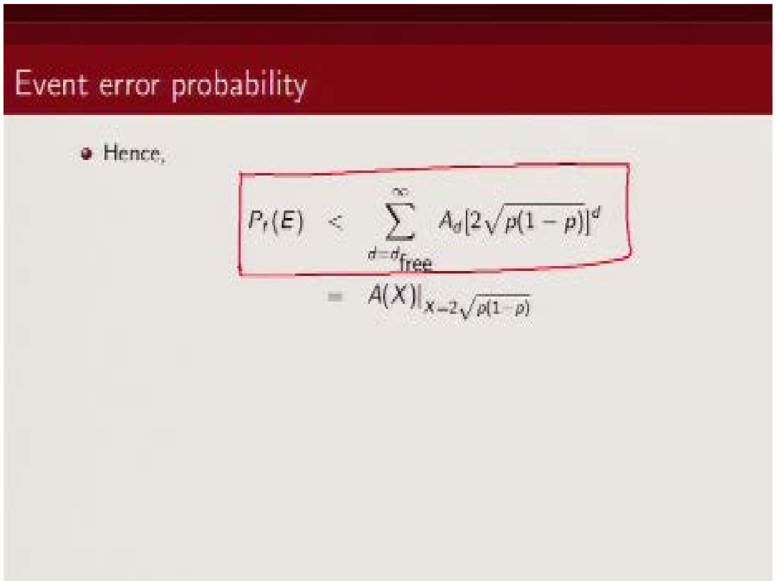
Even for when D is odd it is upper bounded by the same quantity.

(Refer Slide Time: 21:00)



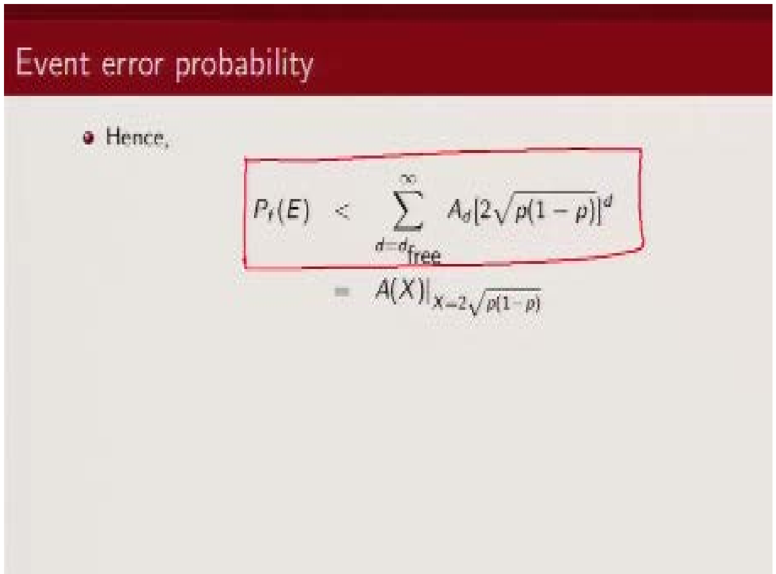
So then what we can do is we can write that our first event error probability is upper bounded by this.

(Refer Slide Time: 21:04)



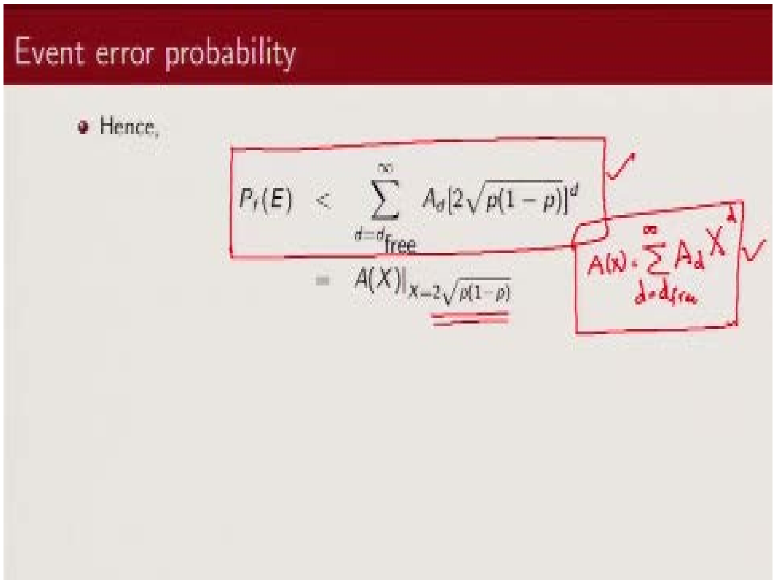
We just showed this for d equal to odd and d equal to even separately. So we can upper bound this probability by this.

(Refer Slide Time: 21:18)



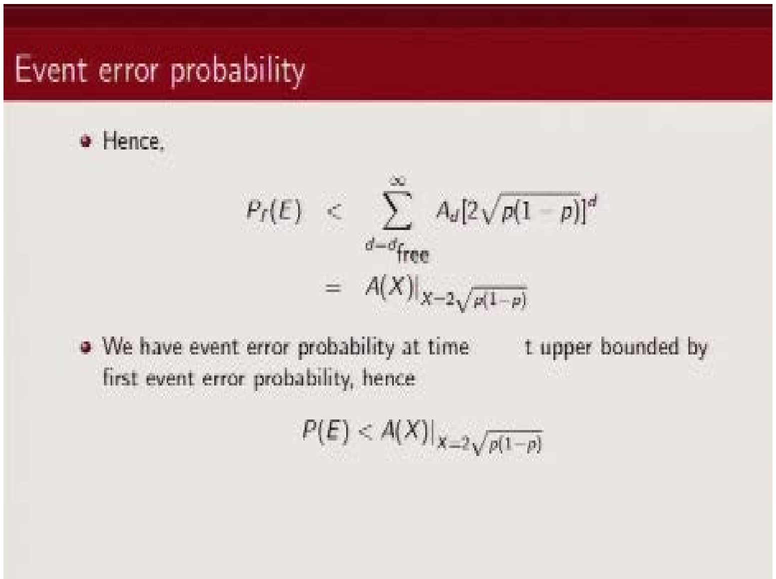
Now what we are going to do is we know our weight enumerating function so we will give us the distribution.

(Refer Slide Time: 21:41)



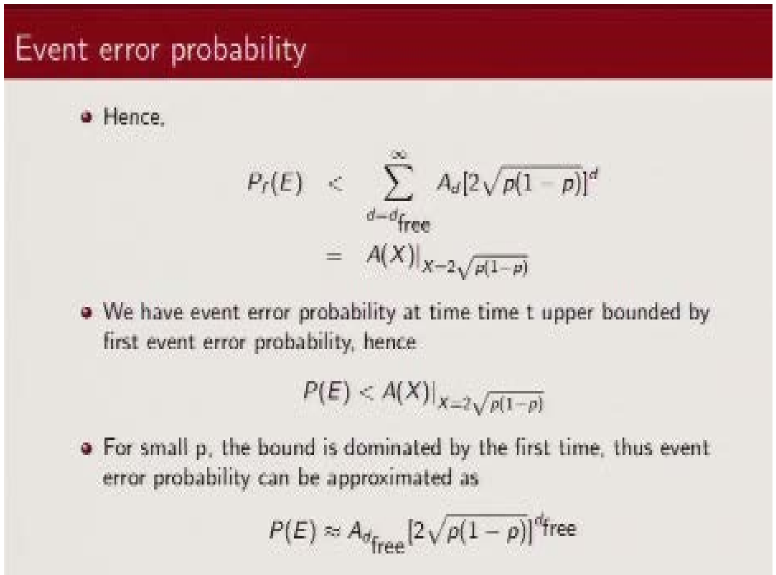
So if we have some weight enumerating function it is essentially will give us from d = d_{free} to infinity it will give us how many code words of weight d are there and x^d okay, so we are essentially making use of this weight enumerating function and we look at the form of our first event error probability so looking at these two we can write then that first event error probability can be computed from the weight enumerating function by replacing the weight that x by this quantity. So this we obtain by comparing this equation with this equation, okay?

(Refer Slide Time: 22:35)

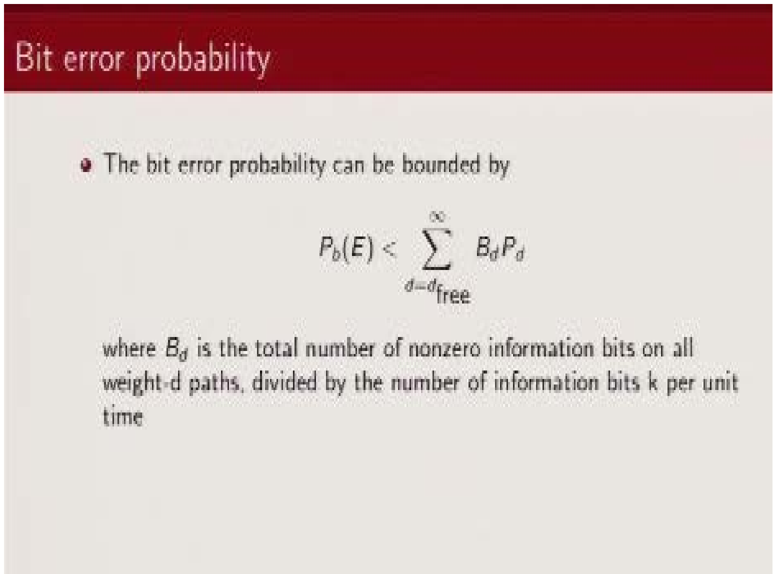


Now what sort of error events can happen? So let us just spend some time on that.

(Refer Slide Time: 22:43)

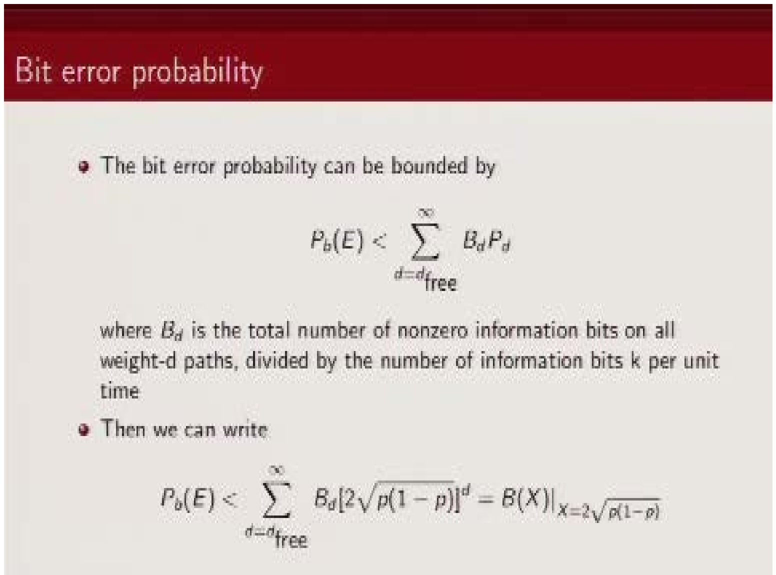


(Refer Slide Time: 22:45)

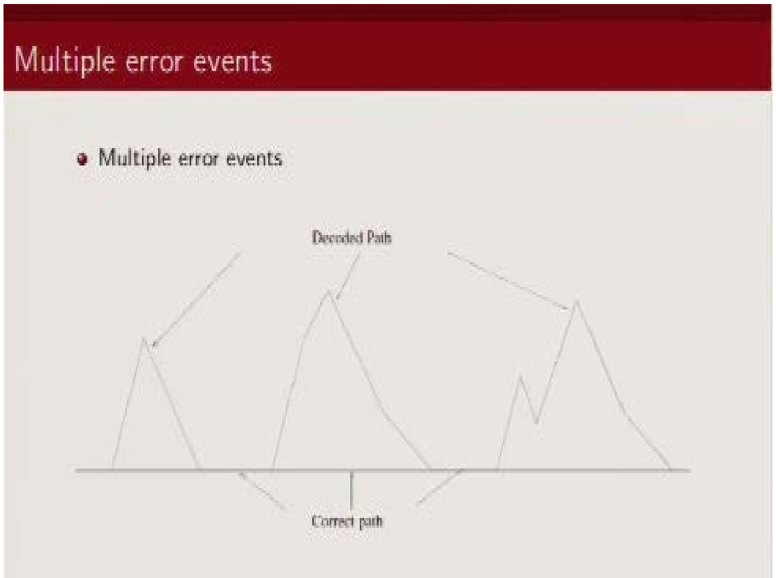


What is the type of error events can happen?

(Refer Slide Time: 22:46)

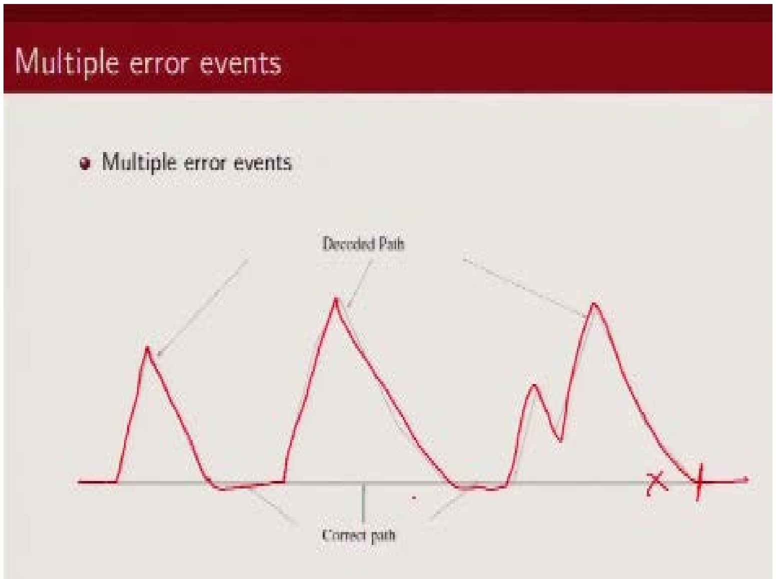


(Refer Slide Time: 22:47)



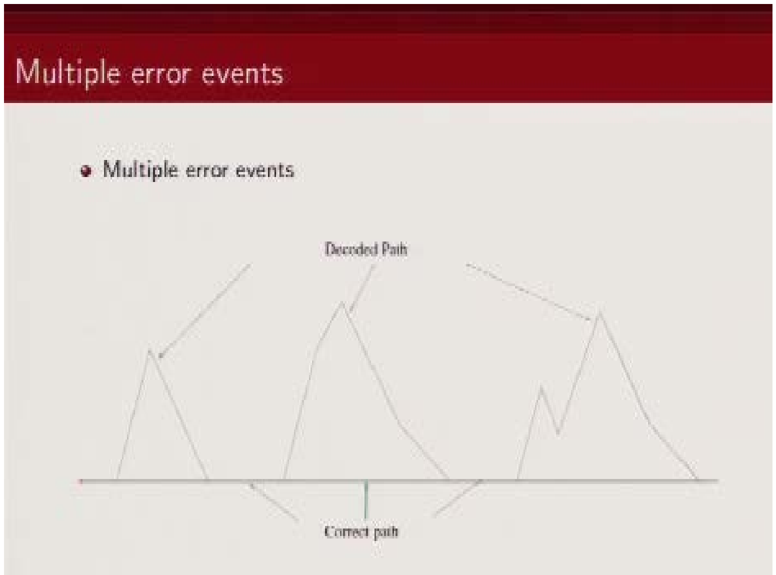
So of course you can have a scenario where you have multiple error event.

(Refer Slide Time: 22:50)



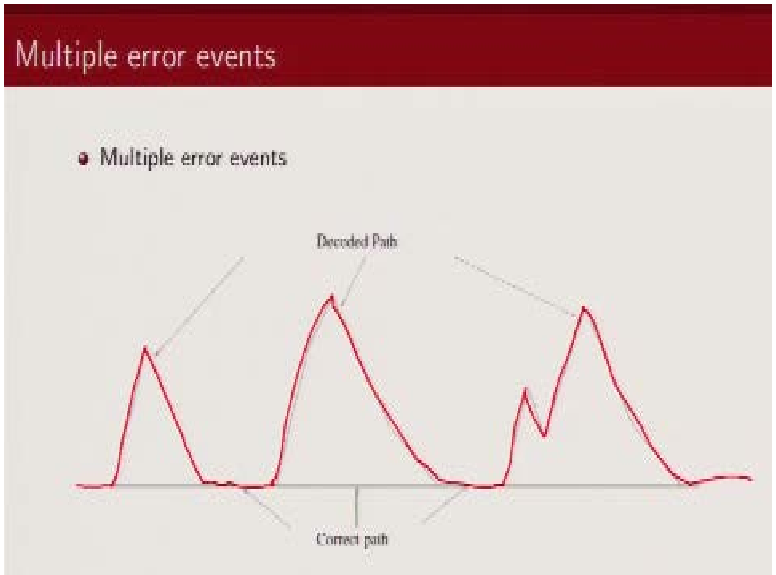
So you had, you diverged from all zero state you merge back then you again diverged you merge back, you diverged you merge back this can happen, now when will you, so if your decoded path is this that means the metric that you are getting here is better than the metric value for all zero state and that is why you are deciding in favor of this.

(Refer Slide Time: 23:19)



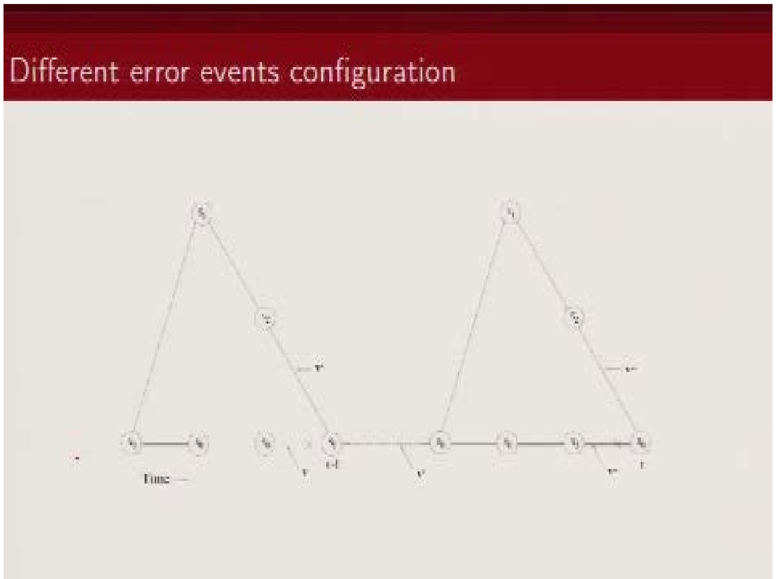
Now there are multiple ways in which these errors can happen, so error can happen error events can happen in multiple ways so let us say.

(Refer Slide Time: 23:30)



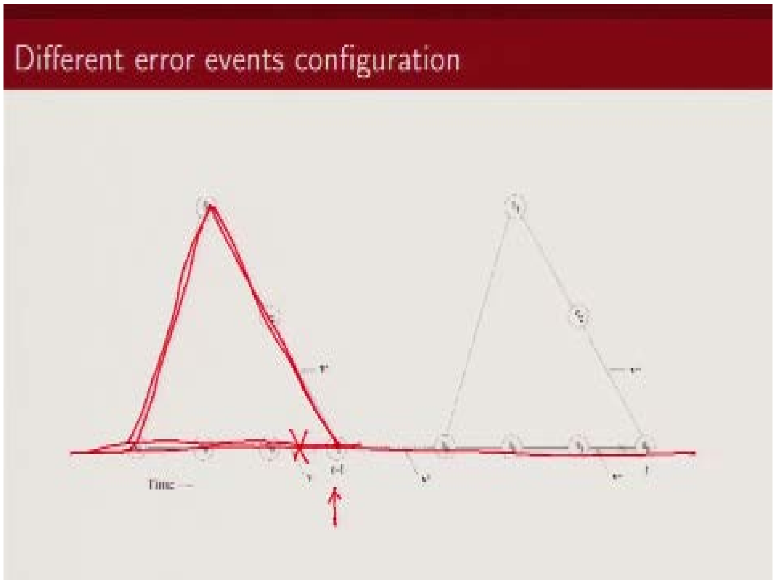
There might be situation when you have diverged from all zero state and then merged back into all zero state and then after some time you again diverge from all zero state, stay away from all zero state and then you merge back and again then you stay away from all zero state diverge and then again merge back, so you can have multiple error events happening.

(Refer Slide Time: 23:59)



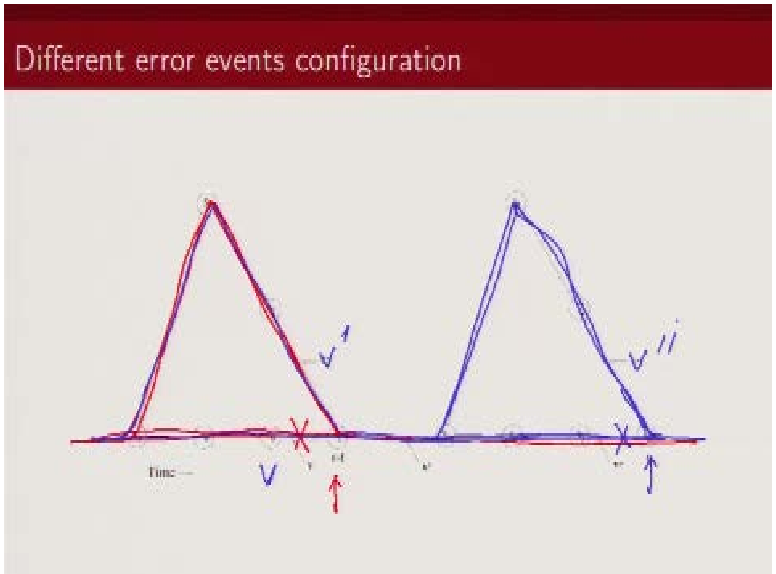
Now let us look at whatever various types of error configuration you can have.

(Refer Slide Time: 24:08)



So for example you could have a scenario like this, so you diverge from all zero state and then you merge back and at this time since, let us say your metric corresponding to this path was better than metric corresponding to all zero state path so you have discarded all zero state path and decided in favor of this path.

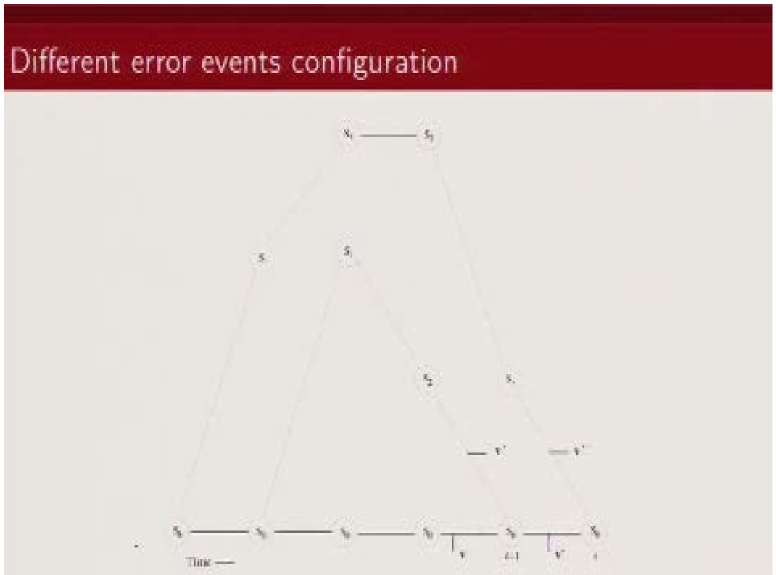
(Refer Slide Time: 24:36)



Now further up when you came here you are here and there was one path which was going like this and when you came to this time is this t then again the metric corresponding to this was worse than the metric value corresponding to this path so you decided to go for this particular code word just calling this all zero state as vector V having code word V this is code word V' and this is code word V".

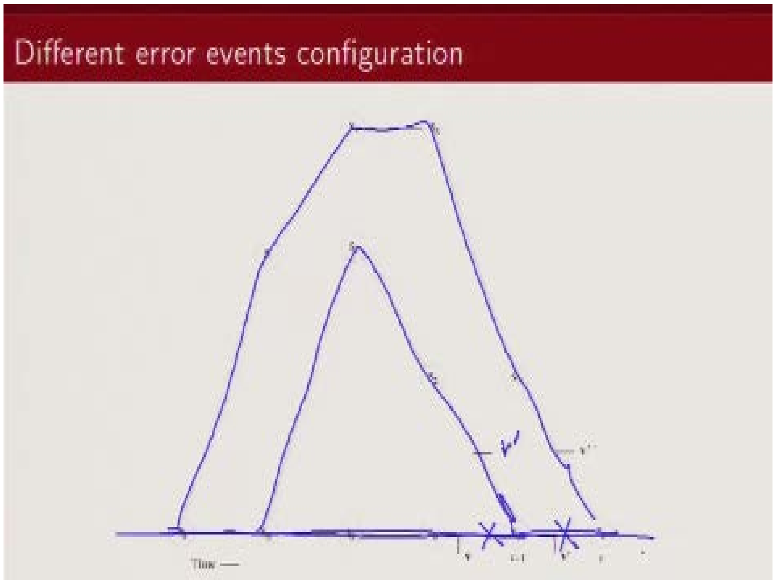
So the point which I am trying to make is whenever you have these kinds of multiple error events you can always upper bound your event probability by error probability of the first event error. So when you decide in favor of this particular code word that means this has better metric corresponding to V' or V, right? So we can upper bound the probability of event error by first event error probability.

(Refer Slide Time: 25:51)



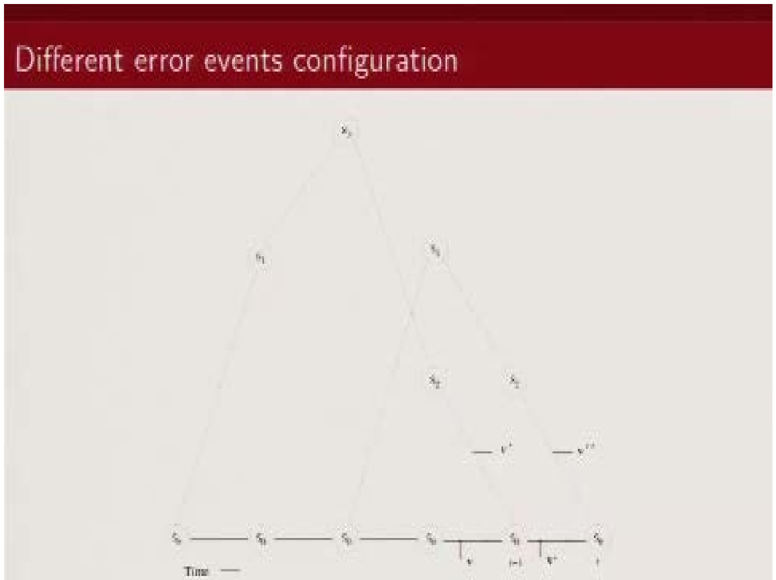
And this could have multiple configuration for example.

(Refer Slide Time: 25:55)



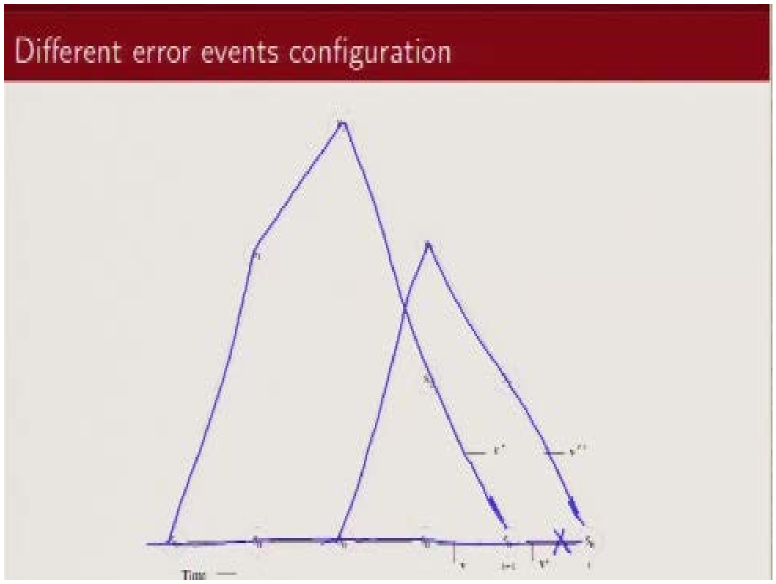
This is your all zero state so you are here all zero state when you came to this point you notice that this metric was better than this metric so you discarded this and you decided in favor of this particular code word, which is your code word V' but then when you came here again this metric was better than this so you discarded this and you decided in favor of this, so this another way multiple error events can happen.

(Refer Slide Time: 26:25)



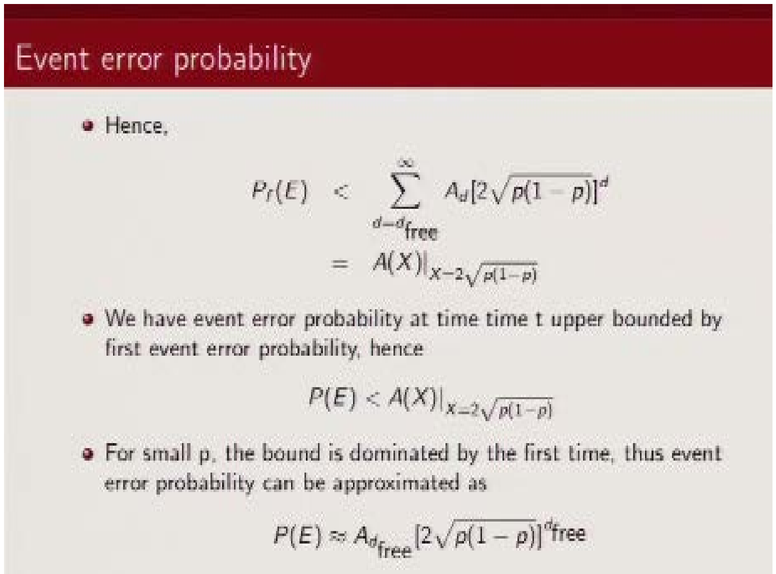
Or you could have a situation like this.

(Refer Slide Time: 26:28)

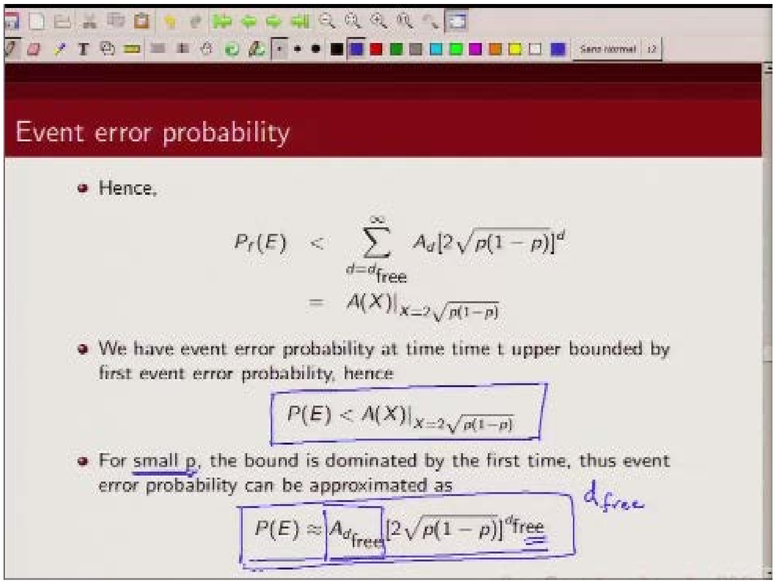


You are at all zero state when you are at this particular instance you notice this has a better metric so you decide in favor of this incorrect path, but then when you reached here you noticed this path has a better metric so you discarded this and then you decided in favor of this, now for each of these cases we can upper bound the event probability by the first event error probability which we have already computed.

(Refer Slide Time: 26:54)



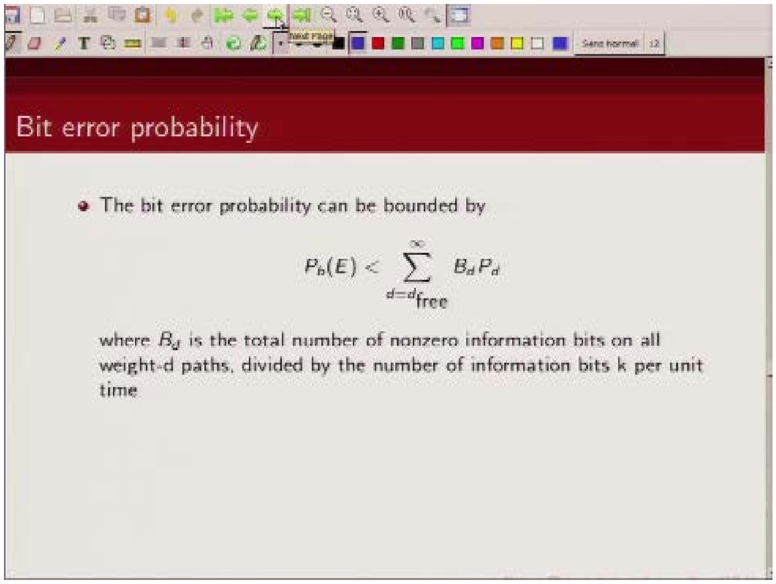
(Refer Slide Time: 27:04)



So this event error probability can be upper bounded by the first event error probability so we can write a probability of error in this particular fashioned, now note here you had terms corresponding to various d's so you had terms correspond to d_{free}, $d_{free,}$ +1, d_{free}+2 so since the value of p is typically very small the most dominating term in this error probability expression is your first term which is the $d_{free,}$ term.

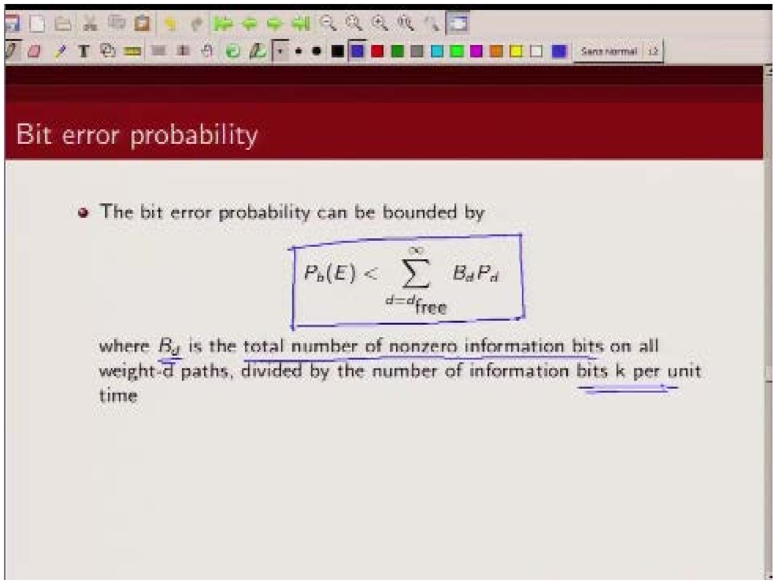
So you can also approximate your probability of event error by how many such events are there which have paths which have A d_{free} and this raised to power $d_{free,}$ So case when p is very small this bound is dominated by the first term which is the term corresponding to d_{free} and in that case we can write down an approximate expression for probability of error like this.

(Refer Slide Time: 28:28)



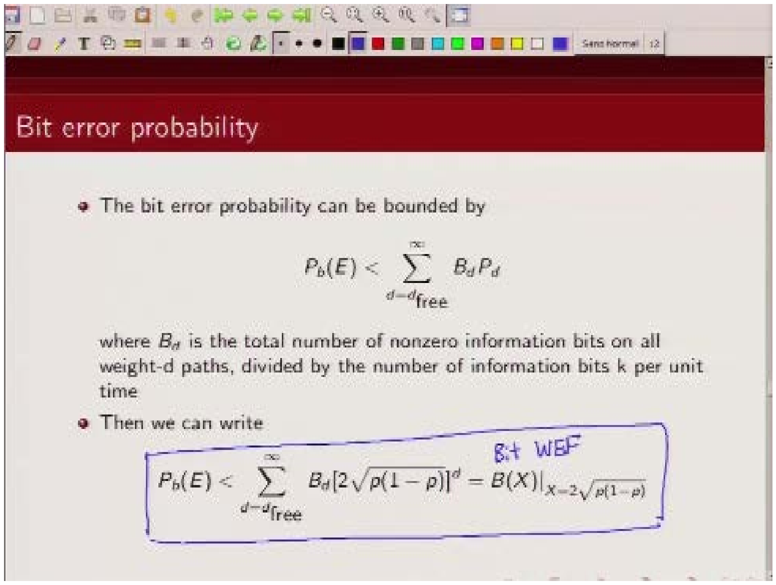
Now can we modify this expression to calculate bit error rate probability the answer is yes. So you know what is a weight corresponding to an incorrect path, now if we can find out what is the information weight corresponding to these incorrect path and then.

(Refer Slide Time: 28:54)



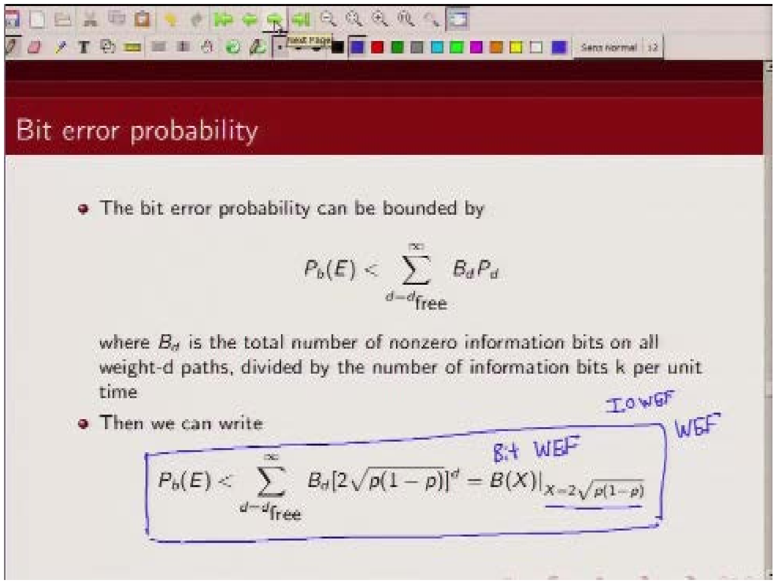
We divide it by total number of information bits, we can find out what is the bit error rate probability. So bit error rate probability can be computed from this first event error probability this is given by this expression where B_d is the total number of nonzero information bits on all these weight-d incorrect paths, divided by k information bits per unit time.

(Refer Slide Time: 29:31)



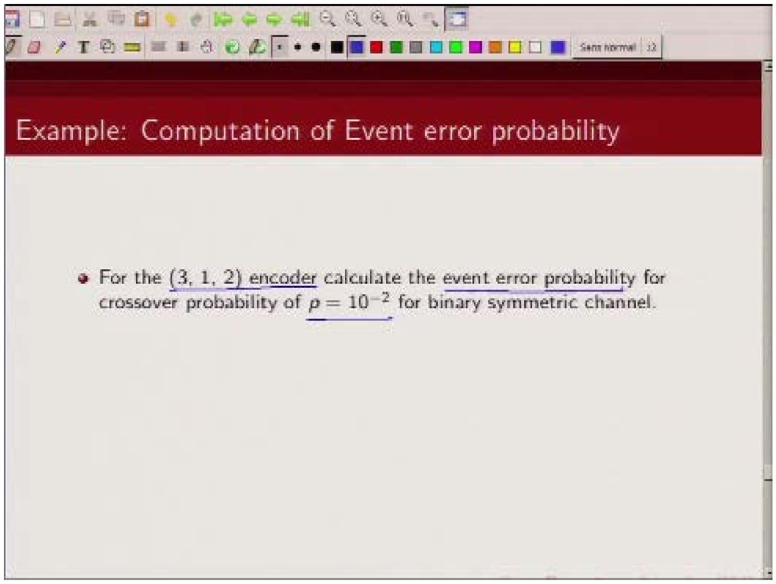
So if we plug that value of bit error rate probability we can see that, we can similarly write the expression for bit error rate probability it is the bit, this is bit weight enumerating function. And in a minute we will talk about how to generate bit weight enumerating function from.

(Refer Slide Time: 30:02)



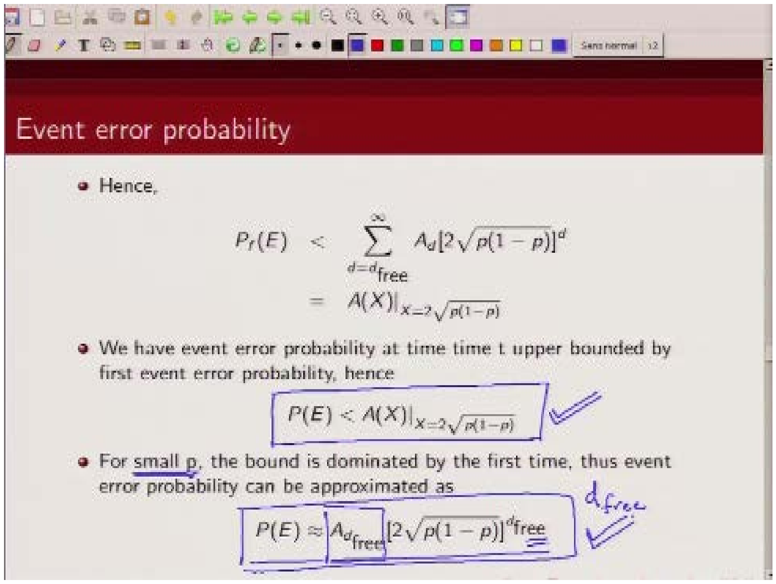
Input, output weight enumerating function or from weight enumerating function we will talk about this so this can be computed from bit weight enumerating function by substituting x equal to this quantity okay. Let us just take an example to illustrate this.

(Refer Slide Time: 30:23)



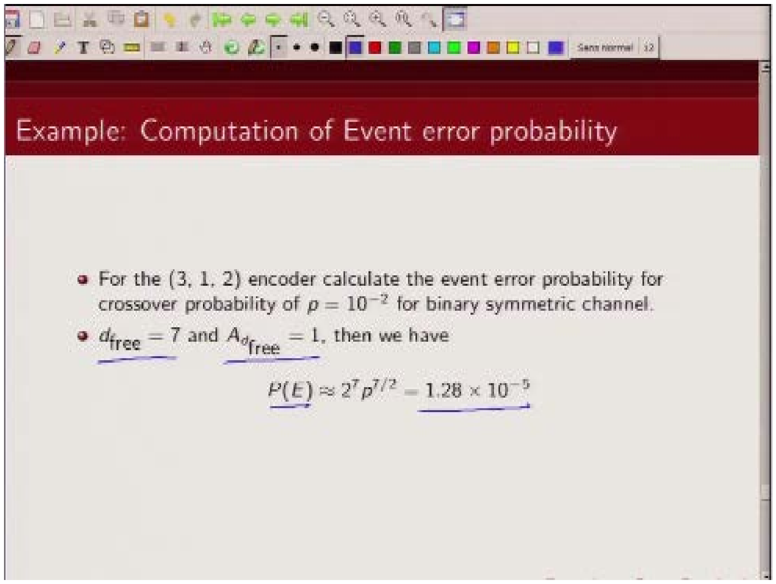
So let us consider the same encoder that we have considered so it is a (3, 1, 2) convolutional encoder it is a feedforward encoder and what is given to us is the crossover probability is point 01, you were asked to compute what is the event error probability. Now we know the expression of event error probability the bound on event error probability that is given by this expression.

(Refer Slide Time: 30:52)



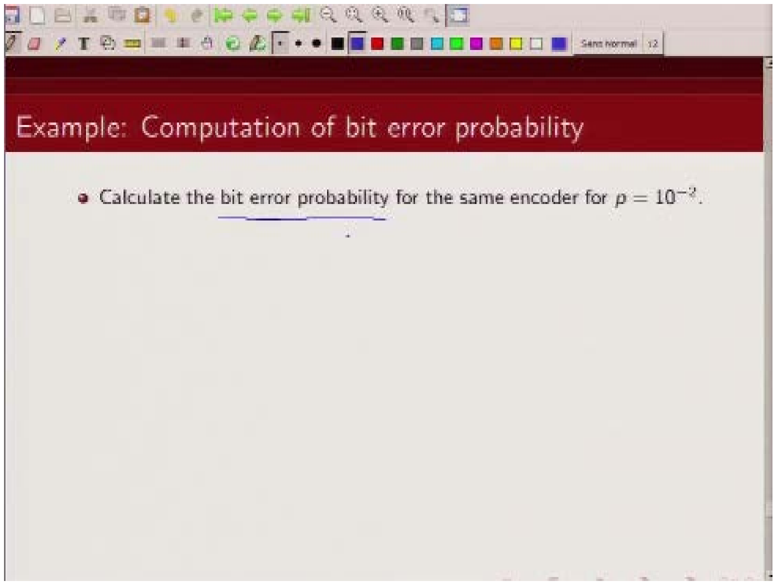
In this particular example p is very small, p is point 01, so we can approximate we can get the approximate expression of event error probability from this. Now what is the free distance of this convolutional code again if you go back the minimum weight sequence was 7. So d_{free} for this particular example was 7.

(Refer Slide Time: 31:21)



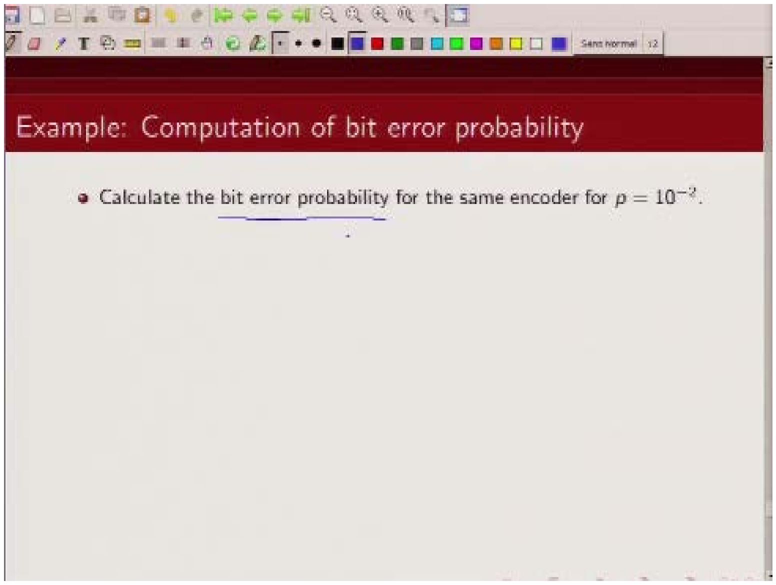
So free distance was 7, and there was only one such sequence of weight 7, so we plug those values of d_{free} and Ad_{free} in the expression for probability of event error and we get probability of event error to be roughly 1.28 x 10^{-5}.

(Refer Slide Time: 31:46)

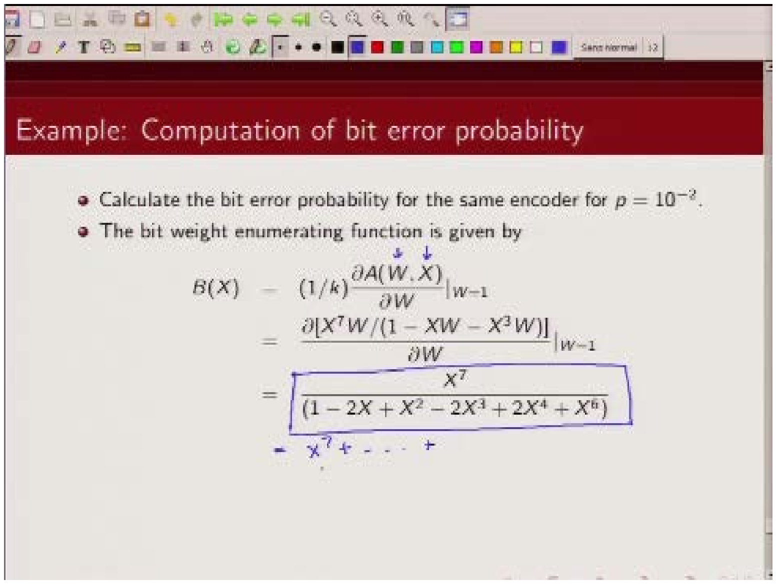


Now how do we compute the bit error probability so first we have to generate bit weight enumerating function. We know how to compute weight enumerating function it has been explained in lecture 2C.

(Refer Slide Time: 32:00)



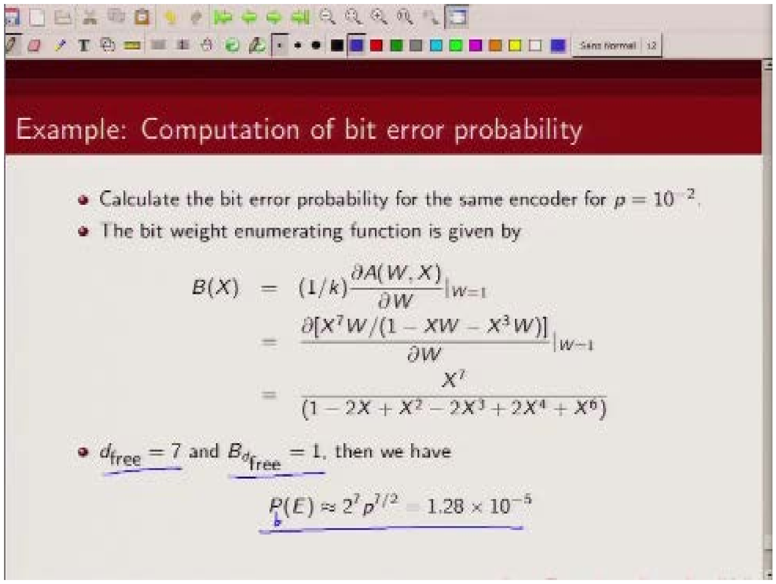
(Refer Slide Time: 32:01)



So assuming you have the weight enumerating function which tells you and you have the input, output weight enumerating function which will tell you what information bit causes what is the corresponding output weight, so you could compute bit weight enumerating function by partial derivative of your input, output weight enumerating function with respect to W and putting W=1 and 1 by k times that, if you do that you will get the expression for bit weight enumerating function.

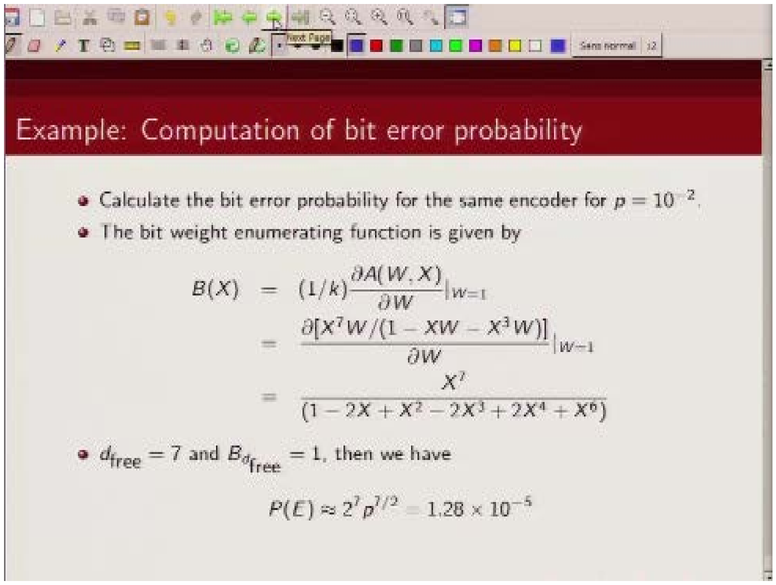
So for the example, that we have considered we already had the expression of input, output weight enumerating function so if we plug that in we get the expression for bit weight enumerating function like this. And then we can divide this by this so we will get X^7 + some, some terms like that. So in this case also what is d_{free}, d_{free} is 7 and Bd_{free} is also 1.

(Refer Slide Time: 33:11)



So this quantity is 1, d_{free} is 7 so we can approximate the expression because the first term corresponding to the free distance will be the dominating term so we can write probability of bit error to be this okay.

(Refer Slide Time: 33:31)



So with this I will conclude this discussion on performance analysis of convolutional code over binary symmetric channel. Now similar analysis can be done for other channels as well. Thank you.

Acknowledgement

Ministry of Human Resource & Development

Prof. Satyaki Roy

Co-ordinator, NPTEL IIT Kanpur

NPTEL Team

Sanjay Pal

Ashish Singh

Badal Pradhan

Tapobrata Das

Ram Chandra

Dilip Tripathi

Manoj Shrivastava

Padam Shukla

Sanjay Mishra

Shubham Rawat

Shikha Gupta

K. K. Mishra

Aradhana Singh

Sweta

Ashutosh Gairola

Dilip Katiyar

Sharwan

Hari Ram

Bhadra Rao

Puneet Kumar Bajpai

Lalty Dutta

Ajay Kanaujia

Shivendra Kumar Tiwari

an IIT Kanpur Production

©copyright reserved