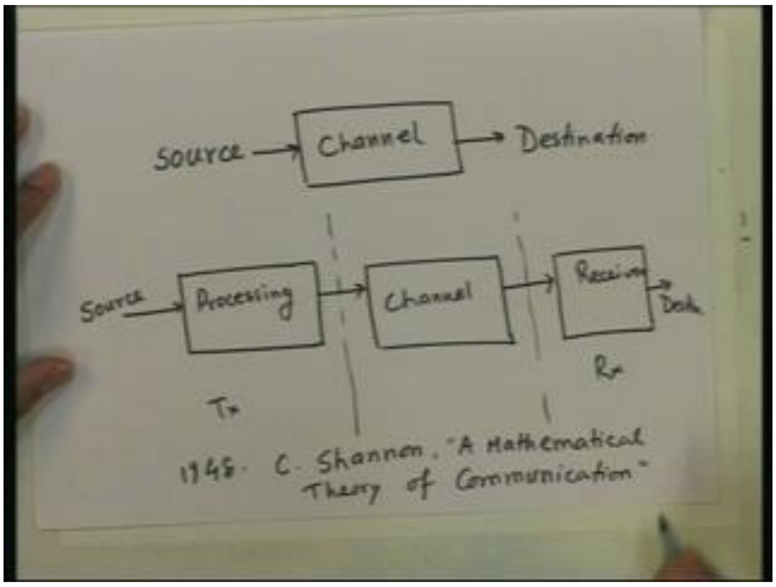
Digital Communication

Prof. Dr. Bikash Kumar Dey
Department of Electrical Engineering
Indian Institute of Technology, Bombay

Lecture - 08
Information Theory
(Part - 1)

Hello everyone. Welcome to the class. In this class we will discuss some elements of information theory, which will give you some idea about what is possible to achieve in a point to point communication channel, and also if we want to compress a signal to what extent we can compress the signal. So, let us see what is the goal of a communication system? The basic goal of communication system is to transmit some information from the source to the destination. So, the source is one which generates some signal a random process and that we want to convey that information to the destination through a channel. So, there is a source and there is a destination and there is a channel in between.

(Refer Slide Time: 01:59)



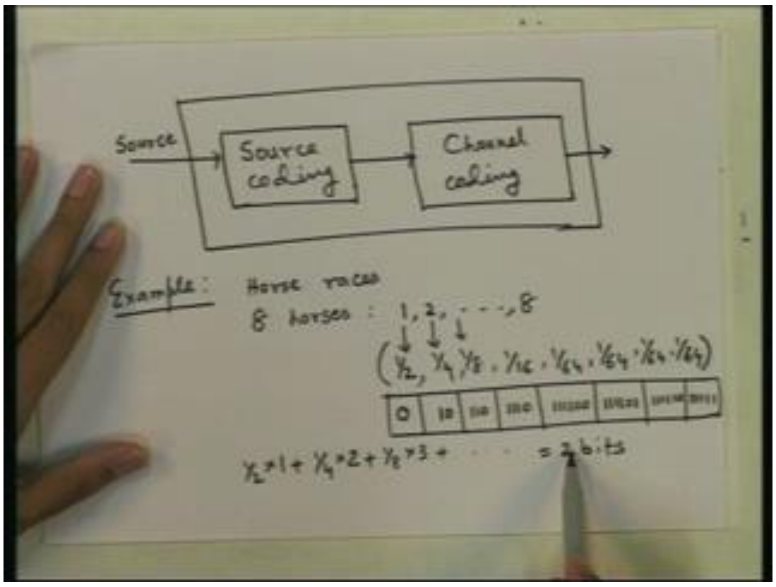
So, usually we would like to do some processing here so that we can transmit maximum amount of information from the source to the destination. So, what kind of processing we will like to do is a question. So, we have the source and we process the signal that is generated by the source and then we pass that signal through the channel. So, this is at the transmitter side. So, this side is transmitter and this is the channel and we have the receiver. So, this is the receiver side and finally, it reaches the destination.

So, what kind of processing is to be done here so that we can transmit information from the source to the destination at the maximum rate is the question. Shannon in his first paper in 1948, Shannon Claude Shannon published paper which is titled a mathematical theory of communication, which made some breakthrough contributions in the area of communication. It totally changed the way communication engineers looked at the looked at communication engineering. So, prior to Shannon, communication engineering was mode of ad hoc in nature, meaning by people used to try different methods with trial and error. People used to see what kind of performance one can get.

But Shannon for the first time tried to analyze it in a rigorous manner and came up with results which gave good; which threw light into the what can be achieved and what cannot be achieved. So, rather than giving techniques for achieving good performance, he tried to analyze what kinds of achievements are possible in communication engineering. What, how much what is the maximum rate at which one can transmit data through a channel? Or what is the minimum rate at which one can compress a source. So, he said that this processing block that we have drawn here as a single block can be divided into two parts. One which tries to compress the source to the maximum extent and the other which tries to transmit the compressed data through this channel in a efficient manner.

So, the problem of designing the transmitter was broken into two parts. One is called the source coding which compresses the source and the other is channel coding which tries to transmit the information through the channel an efficient manner. So, the source coding part is particular to source channel coding part is particular to the channel. And he showed that by dividing this into two blocks, one source coding and the other channel coding for most of the simple channels. We will consider by dividing this processing block into 2 parts does not lose any optimality.

(Refer Slide Time: 06:38)



Meaning by, you can achieve the maximum rate at which we can transmit the source to the destination. A given source to the destination through a given channel will not be reduced if you try to break it break the processing block in to two parts and try to design these two blocks individually in optimum way. So, if we do optimum source coding and optimum channel coding here that will also be the optimum processing block optimum transmitter together. So, that is the very significant result, this applies to not all types of communication applications, but it applies to the channels we will consider. So, we will discuss in this class mostly source coding and we will also discuss channel coding in sometime.

So, let us consider source coding. So, what is the job of source coding? We have a source here which generates some data which needs to be compressed to the maximum possible extent. Let us take an example to see, to develop the intuition behind what is to be done in this block. So, let us say we have horse races. Every day that there is say, one horse's race held among say eight horses. So, there are eight horses 1 2 1 to eight number of first or second horses and so on. So, there are 8 horses and everyday there is a race between them and the probability the first horse wins the race is half the second horse. So, this is the strongest horse probability, that the second horse wins is say one fourth. Probability that the third horse wins is let us say, one eighth and then the next horse has probability one sixteenth.

So, these are the probability of wining then the fifth horse winning probability sixty-four. This also has sixth horse also has probability one sixty-fourth, this also one sixty-fourth this also one sixty-fourth. So, these are the probabilities with which in the individual horses win a race. And everyday there is a race and at the end of say few years and needs to transmit the information of which race was won by which horse to another place. Let us say, over a year we have 365 horse races and we have 364 or 365 number of results to be transmitted. Now, since there are eight horses; obviously, one can transmit each result by transmitting three bits through a channel.

So, it is sufficient if we can afford to transmit three bits to convey the information convey the result of one race. But can we do better? Is it possible to transmit at lower a rate per horse race; Can we transmit lower than 3 bits to still transmit the same amount of information? So, the answer is yes, it is possible. How? Let us say we assign the following. That whenever we want to transmit the result 1, that is whenever the first horse wins and we want to transmit the number 1, we transmit instead of 3 bits we just transmit 1 bit and we transmit say 0.

And whenever the second horse wins we transmit 1 0. Similarly, for third horse when the third horse wins we transmit 1 1 0. When the fourth horse wins we transmit 1 1 1 0. When the fifth horse transmits wins we transmit 1 1 1 1 0 0 six bits. If we are, if we were wondering how we decide the number of bits to be transmitted, it is nothing but log of 1 by the probability. There 1 by this probability is 64 to log base 2 of 64 is the number of bits I am assigning to this. So, there are six bits here there are 4 bits here because log sixteen base 2 is 4 log eight base 3 is the log eight base 2 is 3 and So on. So, this also we want to assign 6 bits. So, we will transmit 0 1 for this, 1 1 1 0 for this 1 1 1 1 1.

So, these are my; these are the bits we will transmit for the corresponding results. So, if the first horse wins I will transmit 0. If the second horse wins I will transmit 1 0 and So on. So, these are the width sequences that we will transmit corresponding to the horse number if that horse wins. Now, what is the number of bits we will need on average? Now, let us say we want to transmit 365 of number results to the destination. On average how many bits will we need to transmit? So, we will transmit the result of the first race using this coding, meaning by if the fifth horse wins won the first race, we will simply transmit this fifth bit sequence and then we will take the second horse race result. And if that was won by the third horse, we will transmit these 3 bits next and so on.

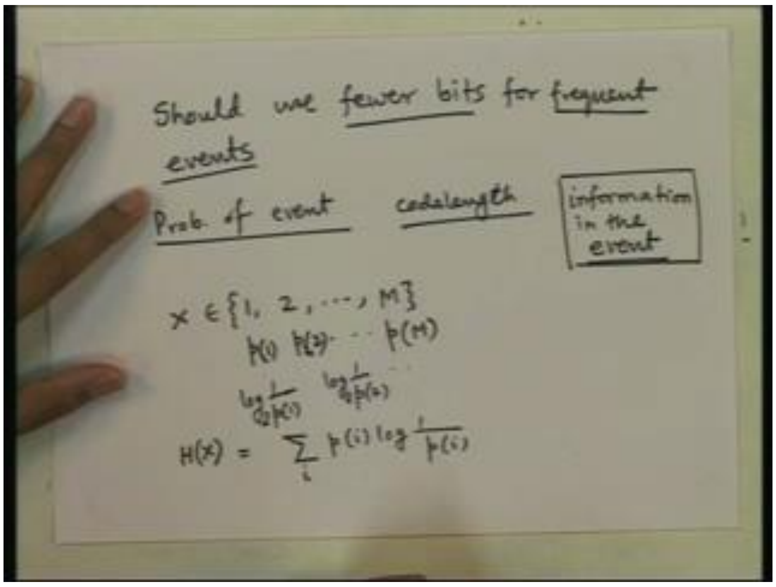
So, we will transmit 365 number of bit sequences depending on the results of those 365 horse races. And if we do that what is the average number of bits we will require to transmit to convey the 365 horse races results? So, the results will be half because the first horse has the probability half of winning half of the times the first horse will win. So, per race we will need half the times I will need only 1 bit. So, with probability half I will need 1 bit with probability; one-fourth we will need 2 bits and so on. With probability eighth we will need 3 bits. So, we will add all these. So, this is the average of average length of the bits sequences we have assigned to each horse. So, the average bits sequence length that we will need to transmit to convey 1 horse race result is this quantity. This is the average of all these bit sequences. The weighted average because there is a probability associated with each string. This string will be used with probability one-fourth. So, we have to multiply by one-fourth to the length of this code.

Similarly, this string will be used only one sixty-fourth times; that means, with probability one sixty-fourth. So, we have to multiply these lengths 6 to 164 and then with that way we take the weighted average. So, this length turns out to be 2 bits instead of 3 bits. So, if we use same length code for. So, these are the codes for the horses. If we length say if we use fixed length code for each horse we will need 3 bits, because they are 8 horses so we will need 3 bits. So, we will assign 0 0 0 to this 0 0 1 to this and 0 1 0 to next and so on. So, fixed length code we will require 3 bits whereas, with variable length code we can use we can transmit with only 2 bits. So, it suggests that whichever horse wins more frequently, that horse should be assigned a shorter bit sequence as the code.

So, to transmit the result corresponding to that horse, we should transmit less number of bits compared to the horse race which have smaller probability of winning. So, the frequent events should be communicated with less number of bits to save the bits that is the fundamental principle of source coding. So, that also is very intuitive because we want to minimize the number of bits we transmit. So, for doing that; obviously, the events that we need to transmit more frequently, if we assign less number of bits to those events intentionally then we will be transmitting on average less number of bits. Because the most frequent events are assigned less number of bits.

And only the events which are very rare they are assigned longer bit sequences. Because, they do not matter much because we will not be transmitting those bit sequences very often. So, we will be transmitting on average less number of bits. So, that is what is applied, that is principle is applied in this when we develop this code. So, the summary of this is that we should use fewer bits for frequent events.

(Refer Slide Time: 18:37)



Should we fewer bits for frequent events

Prob. of event   codelength   information in the event

$x \in \{1, 2, \cdots, M\}$

$p(1) \; p(2) \cdots p(M)$

$\log_2 \frac{1}{p(1)} \quad \log_2 \frac{1}{p(2)}$

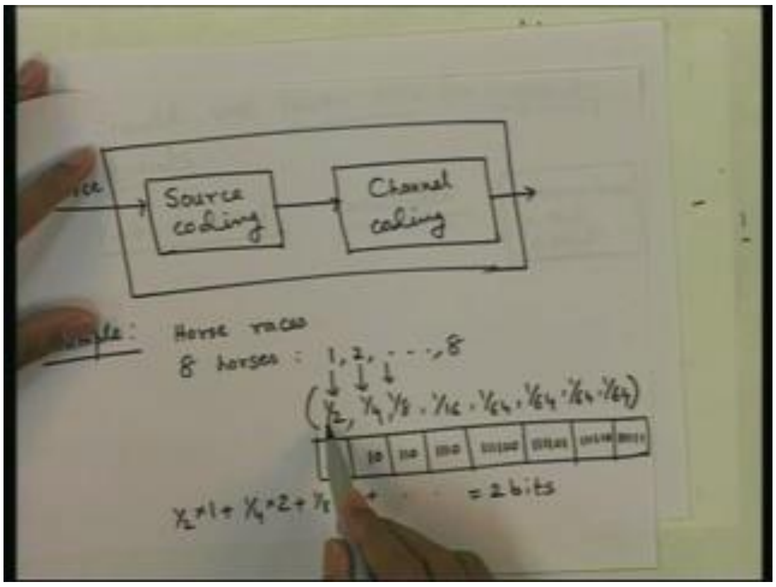
$H(x) = \sum_i p(i) \log \frac{1}{p(i)}$

That means the smaller code length. Why is that it also suggest that if we talk about information in an event that is, how much information a particular event contains? The frequent events contain less information because those events are accepted any way. So, they need not be communicated they have less information. So, for example, it is like saying that, in the summer it is it is like telling that tomorrow will be a sunny day. There is very little information in that statement, because people expect that there will be a sunny day tomorrow anyway. So, if you say that tomorrow is going to be sunny day it is a does not have much information. On the other hand in the summer if you tell that tomorrow there will be rain it will rain tomorrow. Then that will contain a lot of information because it is not so, frequent event it is a rare event in the summer.

So, it has lot of information. So, the frequent events have less information and rare events have more information. So, to transmit a rare event, you should transmit more number of bits. Because, that contains more information and as a result it is it is expected that you should use more number of bits to transmit that event. And things all match because we also see that by doing that we will minimize the number of bits on average we will transmit. So, there is a relation between how many bits you need to assign to an event when designing a source code with the amount of information that event contains. The event if the probability of the event, so there are 3 things probability of the event then, code length for that event and then, the information in that event.

So, they are all related if the probability of the event is small; that means, it is a rare event. The code length should be higher because we do not mind transmitting more bits to a rare event. Because that will be transmitted rarely anyway and another way to look at it is that that event is rare. So, it has more information. So, we should transmit more bits. So, what is the way to quantize the information in an event? Can we say that this event has so much information and other event has so much information; can you quantize it? So, one way to quantize is, as we saw in the example the probability, the event of the first horse winning is very highly probable. So, we assign less number of bits and the number of bits we assign in this case is the log of 1 by the probability and that is defined to be the information contained in that event.

(Refer Slide Time: 22:52)

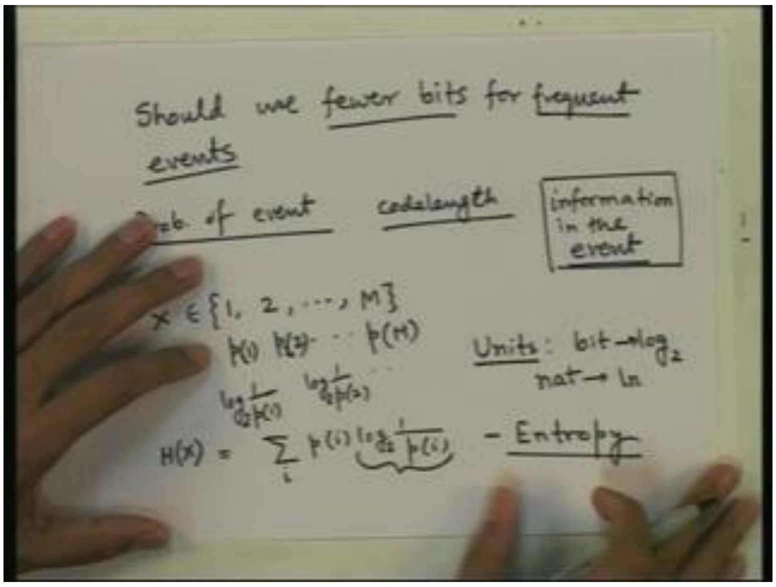


So, if we have a random variable x, which say takes values from 1 2 2 let us say M, with probabilities p 1 p 2, let us say p 1 p 2 and p M. Then then the then the information contained in the event 1 is log 1 by p 1 this has information log 1 by p 2 and so on. So, we will also use it is quite intuitive that, we should probably use the number of bits in code length same as the amount information it has. So, we have to the take logarithm base 2, as we have taken in the example and then construct a code having the code length so much for 2, so much for 1 etcetera. So, this is the information contained in the event 1 this is the information contained in the event 2 and so on.

So, what is the average information contained in this random variable x? It is called entropy of the event, denoted by H of x in entropy of the random variable x and it is defined to be set up all i is summation p i log 1 by p i. So, as we have as we computed the average length of the code before for example, we similarly, take the average of these information, weighted average, weighted with respect to the probabilities. So, as one can see; as one can guess, from the example we have considered that this is also going to be the average code length. If we consider if we construct a code in such a way that the I th event I th point is assigned a code length log 2 log base 2 1 by p i. That the average length is going to be this.

So, because with pi probability the I th symbol is going to appear is to be generated by the source. So, and for that symbol we will use a code length of this length code of this length. So, with pi probability this length will be used. So, summation of that the product will be the average code length. So, now the question is, Is there such a code that is the question? And we will consider code construction techniques later in this course will not going to the details to we will just outline the results that are available now in this class. So, the summary is that information contained in the I th event is this much. This this is the quantity, this is the information in bits. So, if we take log base 2 the unit of information that is taken is bits.

(Refer Slide Time: 26:51)



Should we fewer bits for frequent events

Prob. of event   codelength $\boxed{\text{information in the event}}$

$x \in \{1, 2, \cdots, M\}$

$p(1)\ p(2) \cdots p(M)$

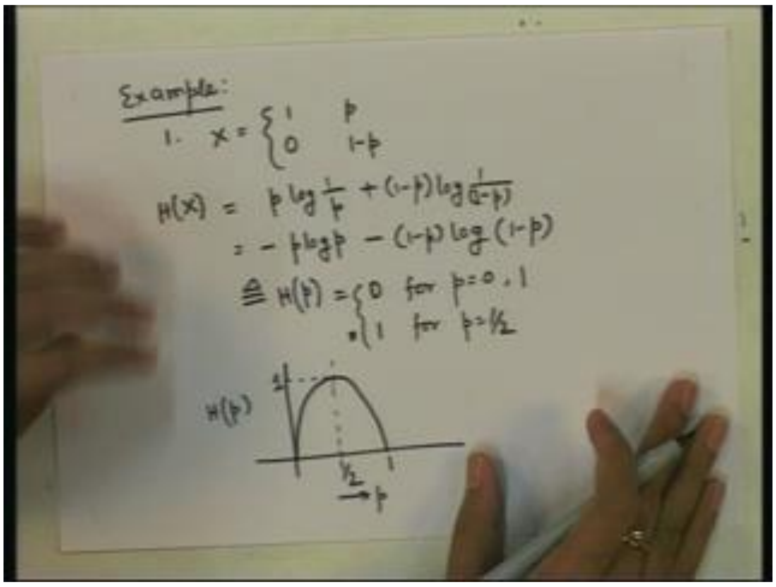
$\log_2 \frac{1}{p(1)}\ \log_2 \frac{1}{p(2)} \cdots$

Units: bit → $\log_2$
nat → ln

$H(X) = \sum_i p(i) \underbrace{\log_2 \frac{1}{p(i)}}$ - Entropy

So, the unit this is the binary unit is bit, if log base 2 is taken. And you can take any base actually, but the 2 units are very commonly used in information theory. One is bit the other is Nat. This is natural unit when you take natural logarithm that is log base E that is also called Ln. So, when we take Ln the unit is called nat. So, these same quantity, but you take Ln. So, much information is there in the I th event. So, much Nat of information is there in the I th event. So, this is the entropy. Let us take an example.

(Refer Slide Time: 27:56)



Example:

1. $X = \begin{cases} 1 & p \\ 0 & 1-p \end{cases}$

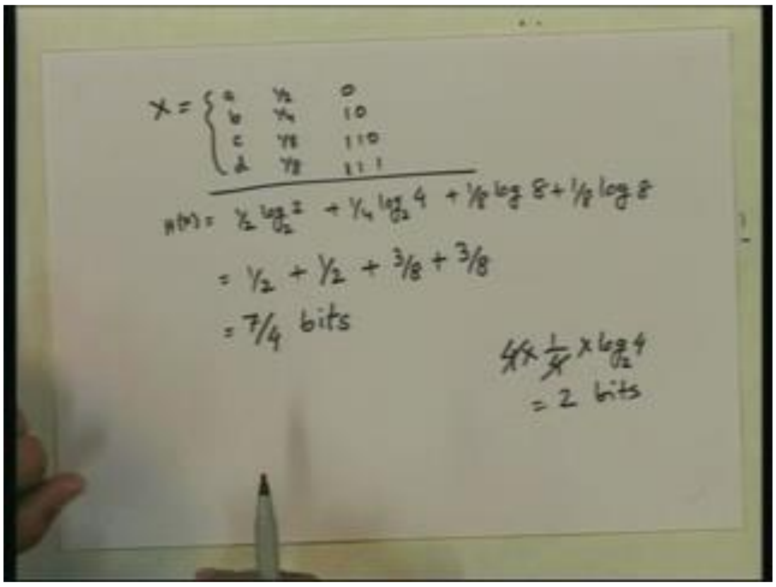
$$H(X) = p \log \frac{1}{p} + (1-p) \log \frac{1}{(1-p)}$$
$$= -p \log p - (1-p) \log (1-p)$$
$$\triangleq H(p) = \begin{cases} 0 & \text{for } p = 0, 1 \\ 1 & \text{for } p = 1/2 \end{cases}$$

The simplest example is of course, the binary random variable. Suppose, the random variable x takes value 1 with probability p and 0 with probability 1 minus p. Then, what is the entropy of this random variable? It is p is a probability of 1 times log of 1 by p plus probability of 0 is 1 minus p time’s log of 1 minus p. That is same as minus p log p minus 1 minus p log 1 minus p and this is also denoted by the notation H p. So, this is the probability of entropy of the random variable and this is the binary entropy function. It is a function of this real quantity p. And this is a very important function plays a very important role in many context and what is the value of these for say p is equal to 0? it is 1 for p is equal to zero. If you have p is equal to 0 you have this quantity 0, assume that 0 log 0 is 0 than 0 and this is this is 1 log 1 that is also 0.

So, this is 0 for p equal to 0 and also for p equal to 1. But it is 1 for P equal to half. If P is half then this is half log half, half log base to half is one. So, it is it is log base 2 half is minus one. So, this is half and this is also similarly, half so, half plus half is 1. So, it has the one can check that this has the maximum value of 1 bit; that means, 1 binary random variable can contain at most bit of information, that make sense. Because a bit can contain at most 1 bit of information. According to definition, the definition has taken care of that, that 1 bit cannot can contain more than 1 bit of information. It can contain less than 1 bit of information. Because, suppose a source always generates once, then do those bits contain any information? It does not, they do not contain any information because it is known once you know once bit you know that the source can generates only once.

So, whether you see the values or you do not see you know the result anyway you know the outcome. So, the output does not contain any information. So, every bit need not contain an information and need not contain the full 1 bit of information. Maximum information it can contain is 1 bit. So, how does this function vary with p? The, if you plot the graph of H p, it will be a graph like this is a symmetric about half. This is H p; this is 1 and this is the maximum entropy it can have that is 1 bit. And as one can see when the probability p is half, it has the maximum information and as it is away more and more away from half it contains less information. So, this is binary entropy function.

(Refer Slide Time: 32:38)



$$X = \begin{cases} a & 1/2 & 0 \\ b & 1/4 & 10 \\ c & 1/8 & 110 \\ d & 1/8 & 111 \end{cases}$$

$$H(X) = \tfrac{1}{2}\log_2 2 + \tfrac{1}{4}\log_2 4 + \tfrac{1}{8}\log_2 8 + \tfrac{1}{8}\log_2 8$$

$$= \tfrac{1}{2} + \tfrac{1}{2} + \tfrac{3}{8} + \tfrac{3}{8}$$

$$= \tfrac{7}{4} \text{ bits}$$

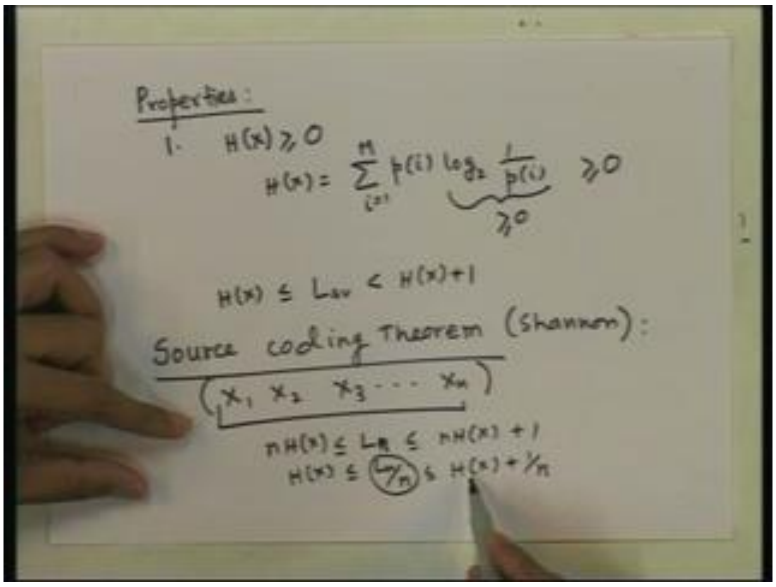
$$\cancel{4} \times \tfrac{1}{\cancel{4}} \times \log_2 4 = 2 \text{ bits}$$

Let us take another example and compute the entropy of the of a random variable. Let us consider a random variable x, which takes 4 values a b c and d with probabilities half, one-fourth, one-eighth and one-eighth. So, what is the entropy? It is this probability half times log 1 by half that is 2 base 2 plus one-fourth log 4 plus one-eighth log eight plus one-eight log eight. And this is half log 2 base 2 is 1 plus this is 2. So, half plus this is 3. So, 3 by eight plus 3 by eight 3 by eight is 3 by 4 and this is 1. So, 1 plus 3 by 4 that is seven by 4 bits. So, this random variable contains 7 by 4 bits of information. What is the maximum amount of information, such a random variable could have which takes 4 values? A random variable which takes 4 values can have maximum amount of information 2 bits as one can expect right.

When will that happen? When the probability is the same. All if all the events have the same probability 1 by 4 1 by 4 1 by 4 1 by 4 1 by 4 then, you would have 1 by 4 log 4 plus 1 by 4 log plus 1 by 4 log 4. And that is 4 times 1 by 4 log 4. So, you will have 4 times 1 by 4 times log 4 which is 2. So, 2 bits is maximum information x can have. So, one can see that, any random variable will have any random variable taking M number of values will have maximum amount of information if the probability distribution is uniform over the values. So, that means, all the events all the outcomes have the same probability. Then the then that random variable contains the maximum amount of information. Okay, and if they are not equally likely then what is the way we should construct codes. So, that if we want to represent a by some bits; b by some other bits; c by some other bits and d by some other bits. Then, if we want to minimize the number of bits on average used on average then what is the way what is length of code words we should assign.

The length of code words should be same as the information contained in the corresponding events corresponding outcomes and that is log 1 by p. So, this should have 1 bit; this should have 2 bits; that should have 3 bits; this should have also 3 bits. So, one such code is 0 then 1 0 1 1 0 1 1 1. This may look very simple and similar to the previous example, but it is not necessarily so simple in general for arbitrary probabilities. Okay, one can see that, in this case also the number of bits used on average it is same as 7 by 4 bits. Because this is 1 times 1 bit; half times is 1 plus one-fourth times 2; one-eighth ten times 3; one-eight times 3, here are the all of them it is basically this. Now, we have introduced we have introduced we have introduced the notion of entropy of a random variable, but let us now see some properties of that function.

(Refer Slide Time: 37:03)



Properties:

1. $H(x) \geq 0$

$$H(x) = \sum_{i=1}^{M} p(i) \underbrace{\log_2 \frac{1}{p(i)}}_{\geq 0} \geq 0$$

$$H(x) \leq L_{av} < H(x)+1$$

Source coding Theorem (Shannon):

$$(\underbrace{x_1 \; x_2 \; x_3 \cdots x_n})$$

$$nH(x) \leq L_n \leq nH(x)+1$$

$$H(x) \leq \boxed{L_n/n} \leq H(x)+1/n$$

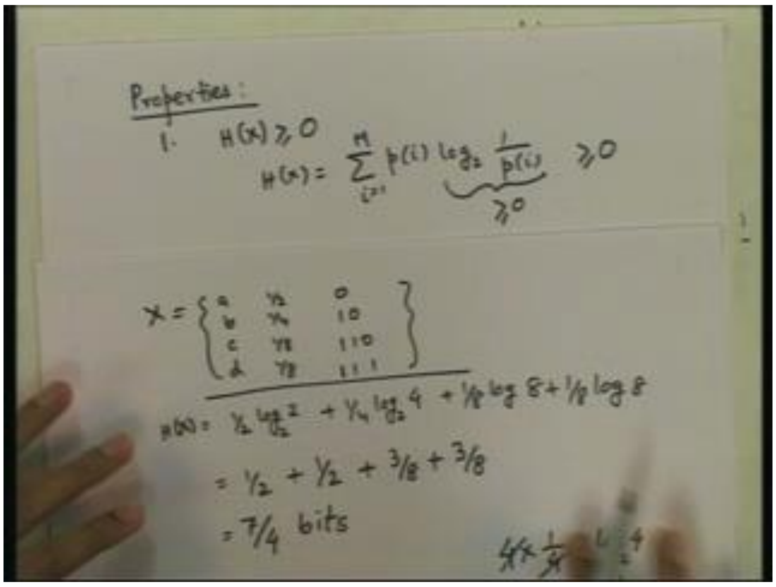
First of all H of x though it is written as a function of the random variable x it only depends on the probability distribution of x. It does not depend on what values x takes Say, x takes 4 values, with first one having probability half; second one having probability one-fourth and third one having probability one-eighth and fourth one having probability one-eighth. Then it does not matter what you call those values. It depends only on those probabilities, for examples, these probabilities, Half; one-fourth; one-eighth; one-eighth. The entropy only depends on this it does not depend on these values.

If you call it a b c d or 1 2 3 4 or horse, elephant, camel, cat it does not matter. It only depends on these probability values.

So, truly speaking this is this entropy is only a function of the probability mass function of x, not on the x as it is. But it depends only on the probability distribution of x and mass function of x. The first property of H x is greater than equal to 0. It is always positive, it is nonnegative. The proof is very simple, because H x is summation over i, i equal to 1 to M p i log 1 by p i. Now, each of these quantities is greater than equal to 0.

It is strictly greater than 0, for this is not equal to 1. So, these are greater than equal to 0 and these are all greater than equal to 0, this whole thing is greater than equal to 0. Okay, now when we do source coding, we will see when we discuss source coding techniques we will see that, instead of doing the coding in individual symbols like we have done here, we have found the a code for this random variable.

(Refer Slide Time: 39:41)



Properties:

1. $H(x) \geqslant 0$

$$H(x) = \sum_{i=1}^{M} p(i) \underbrace{\log_2 \frac{1}{p(i)}}_{\geqslant 0} \geqslant 0$$

$$x = \left\{ \begin{matrix} a & 1/2 & 0 \\ b & 1/4 & 10 \\ c & 1/8 & 110 \\ d & 1/8 & 111 \end{matrix} \right\}$$

$$H(x) = \tfrac{1}{2}\log_2 2 + \tfrac{1}{4}\log_2 4 + \tfrac{1}{8}\log_2 8 + \tfrac{1}{8}\log_2 8$$

$$= \tfrac{1}{2} + \tfrac{1}{2} + \tfrac{3}{8} + \tfrac{3}{8}$$

$$= \tfrac{7}{4} \text{ bits}$$

[illegible]

In general, we may need to combine many symbols together and then do coding. And if you allow that then what there is nice characterization how much one can compress a source. So, the answer is that part symbol you will need about H x number of bits as we have seen in the examples. Also, in the case of the particular examples we have considered the number of bits that we needed was exactly equal to the entropy of the source. But in general it is not possible to encode exactly at the entropy number of bits.

But it is possible to encode the source with average number of bits which is greater than equal to H x.

So, the average number of bits is the average code length of any source code. We can have a good code which will have average length greater than equal to H x, but which is also less than H x plus one. One can show that for any good source code for any source code if the average length will be greater than equal to H x. And, there is also the best code possible will always have length less than average length less than H x plus one. So, that means H x is a very crucial number for source coding, that you can do coding. You can do compression maximum up to H x and it is also possible to compress very up to very near H x between H x and H x plus 1.

And, now if you combine many sources it is it can be checked that one can go as near H x as possible as one wants for compressing the source, for representing the source. So, and that is the famous source coding theorem by Shannon. Which says that, if we have a random variable x, if H x is the entropy of the entropy of x then, we can we can represent x by H x number of bits on average if we do block wise for coding. How is it possible? Suppose, now if we if we the source is not only generating 1 random variable, it is generating the random variable again and again that is it is generating a random process.
Let us suppose that the source generates a I I D random process. Meaning by, every it is generating many symbols one after another and every symbol is having the same distribution. It is just a copy of x and they are all independent and identically distributed.

So, they have the same distribution of x and they are all independent. So, let us call them x 1 x 2 x 3 and so on. So, it is a sequence of random variables the source is generating each one is have the same distribution and they are independent. Then we can combine, say n number of such a random variables I I D random variables and then do coding together. That is, consider all possible values this entropy takes and the corresponding probabilities and generate a code for this whole sequence of random variables.

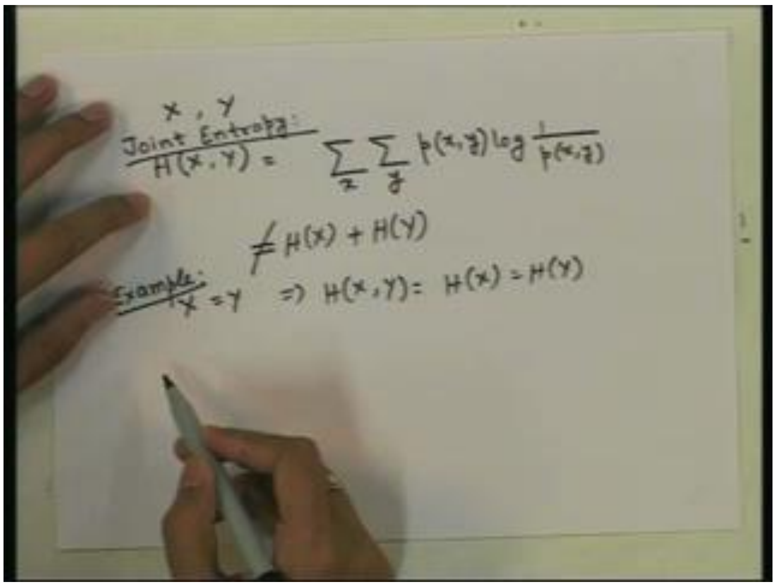
Then, what is the entropy of this? we have to analyze; we can we will see in the next class. The total entropy this these n random variables contain is the n times each random variables entropy because they are independent. It also makes sense because they are independent. So, this does not have any common information with x 2. So, the total amount of information all the random variables contain is the sum of the information's each of them contain.

So, the entropy of all the random variables together will be n times information in each of them the entropy of each of them. So, we will be able to do get a coding scheme, which will require a length L n. Let us say n symbols which lies between n times H x which is the entropy of these random variables and n times H x plus 1. But this average length is for n number of symbols. So, part symbol how many bits are required that is this divided by n. So, that will be bounded by H x and H x plus 1 by n. So, this is the average length that is used part symbols to compress the source; to represent the source and that lies between these two, H x and H x plus 1 by n. And H x plus 1 by n as you increase n this continue becomes smaller and smaller. So, you are approaching H x as n increases.

So, you are taking slightly more number of bits to represent the source and that extra number of bits is decreasing as n tends to infinity it is going to zero. So, we are basically as n tends to infinity you are using only H x number of bits part symbol to represent the source. So, that is basically the source coding theorem. Actually, we will discuss this in more detail when discuss source coding techniques. And we will also prove this that prove source coding theorem this way that by doing block wise source coding and by increasing the block length you can approach the bound H x part symbol.

So, we will need H x number of bits parts symbol to represent a source I I D source. Even for general more general classes of sources not necessary for I I D, entropy rate of such processes can be defined and the source coding theorem holds for more general class of sources than I I D. But we will not going to that details in this course. So, we will just define, what is the joint entropy which is which I mentioned just before as the combined information of some random variables.

(Refer Slide Time: 47:16)



x , Y

Joint Entropy:

$$H(X,Y) = \sum_{x}\sum_{y} p(x,y) \log \frac{1}{p(x,y)}$$

$$\neq H(X) + H(Y)$$

Example: $X = Y \Rightarrow H(X,Y) = H(X) = H(Y)$

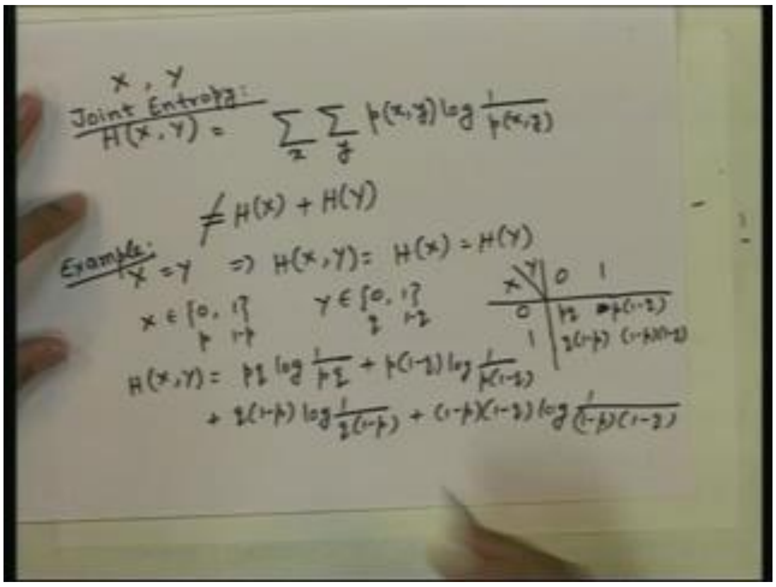
So, if we have to random variables x and y, the joint and entropy H x y. This is the joint entropy H x Y, is defined to be take all x and y. So, these are the values these are like i, i equal to 1 to M you are writing, but x takes many values y take also many values. So, let us take the summation over all possible value of X. This small x denotes a value of capital X. Take all possible values and the summation is to be taken over all possible values of capital X. So, then take p x y, this the joint probability mass function. p x y is the probability that capital X equal to x and capital Y is equal to small y and then log 1 by p x y.

So, this is straight forward because this just like treating x and y together as single random variable. So, this is the probability of this x and y taking these pair of values and so that is all. You are taking summation over all possible pairs of values of this quantity which is the same as the definition for single random variable written in a more general way. So, this is what we mean by joint entropy, that is the combined information x and y have together. This need not be in general H X plus H Y this need not be H X plus H Y. Because, if x and y are dependent then the total information x has and the total information y has. The total information x and y have may not be same as the sum of those two. Because, then may be something common between them x and y may be dependent and as a result they together can may will contain the sum of these two quantities of information.

For example if x and y is same, x and y are same then H X Y is nothing, but H X because y does not contain any extra information then what x contains. So, by taking combined information it will not be more than what H X. So, it is same as H X which is also same as the H Y. So, it need not be same as the sum of the H X plus H Y, but it will be this for independent x and y. If x and y independent then we can expect that y will contain totally disjoint information from x. X and will not x and y will not have any common information and as a result the total information in x and y will be same as the sum of the information x carries and the information y carries.

So, the total information will be sum of the information each of them carries. So, that is the joint entropy. And one can, let us see some other examples. Let us see the see an example where x and y are independent. To see that it will really be equal to H X plus H Y for such an example. So, let us say X takes two values, 0 and it is also it is binary and y also is binary, but they are independent. This has let us say probability p and 1 minus p and this has probability let us say q and 1 minus q. Then what is the joint entropy of x and y?

(Refer Slide Time: 51:45)



X, Y

Joint Entropy: $H(X,Y) = \sum_x \sum_y p(x,y) \log \frac{1}{p(x,y)}$

$\neq H(X) + H(Y)$

Example: $X = Y \Rightarrow H(X,Y) = H(X) = H(Y)$

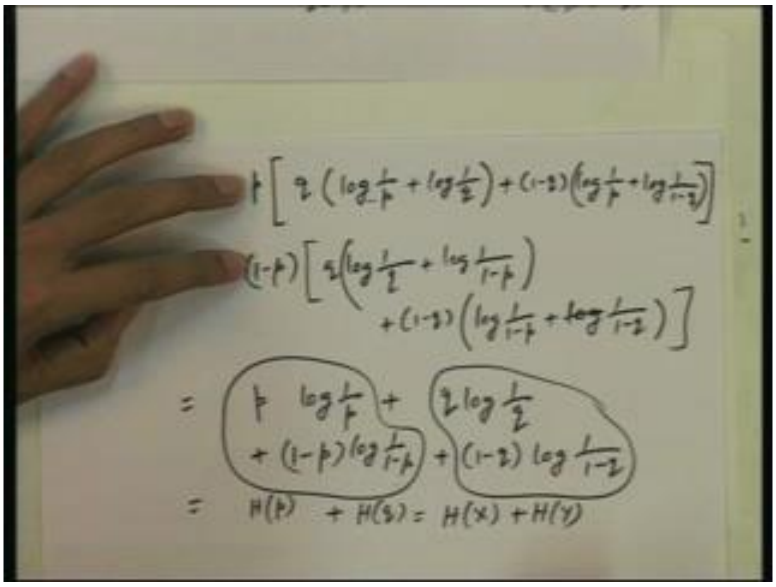
$x \in \{0, 1\}$ with p, $1-p$ $\qquad Y \in \{0, 1\}$ with q, $1-q$

$X \backslash Y$	0	1
0	pq	$p(1-q)$
1	$q(1-p)$	$(1-p)(1-q)$

$$H(X,Y) = pq \log \frac{1}{pq} + p(1-q) \log \frac{1}{p(1-q)} + q(1-p) \log \frac{1}{q(1-p)} + (1-p)(1-q) \log \frac{1}{(1-p)(1-q)}$$

So, X Y this X Y take 0 1 values. 0 1 0 0 has the probability p q. 0 1 has the probability 1 minus this is p 1 minus q. q 1 minus p 1 minus p 1 minus q these are the probabilities. X 0 Y 1 has the probability p times 1 minus q. Similarly, all the other elements. So, what is the joint entropy H X Y? H X Y is this probability p q time’s log 1 by p q plus p 1 minus q time’s log 1 by p 1 minus q plus q 1 minus p log 1 by q 1 minus p plus 1 minus p 1 minus q log 1 by 1 minus p 1 minus q, this is the joint entropy. Now, let us see let us do some manipulation and see if we can prove that this is same as the sum of the 2 entropies. So, let us take p common from these two terms.

(Refer Slide Time: 53:15)



$$p\left[q\left(\log\frac{1}{p}+\log\frac{1}{q}\right)+(1-q)\left(\log\frac{1}{p}+\log\frac{1}{1-q}\right)\right]$$

$$(1-p)\left[q\left(\log\frac{1}{q}+\log\frac{1}{1-p}\right)+(1-q)\left(\log\frac{1}{1-p}+\log\frac{1}{1-q}\right)\right]$$

$$= p\log\frac{1}{p} + q\log\frac{1}{q} + (1-p)\log\frac{1}{1-p} + (1-q)\log\frac{1}{1-q}$$

$$= H(p) + H(q) = H(X) + H(Y)$$

So, this is same as this is same as p times q times log 1 by p q. But log 1 by p q itself can be broken into log 1 by p plus log 1 by q, because multiplication becomes addition outside log. So, log 1 by p plus log 1 by q plus 1 minus q times log 1 b y p plus log 1 by 1 minus q. Similarly, here we can take 1 minus p as common and what do we have here q log 1 by q plus log 1 by 1 minus p plus 1 minus q log 1 by 1 minus p plus 1 log 1 by 1 minus q. So, this now can be written as p log p q log 1 by p let us take log 1 by p term everywhere log 1 by p log 1 by p. So, log 1 by p times p q plus p times 1 minus q. So, p common p log 1 by p in both places and then q plus 1 minus q is 1.

So, it is just p log 1 by p plus take log 1 by q term p q times this plus log 1 minus log 1 by q this is 1 minus p time’s q. So, p time’s q plus 1 minus p time’s q is q. So, it is q. Similarly, log 1 by 1 minus p this is q times 1 minus p and this 1 is 1 minus q times 1 minus p. So, this is 1 minus p log 1 by 1 minus p plus if we take 1 by 1 minus q log 1 by 1 minus q this term and this term. So, p times 1 minus q plus 1 minus p times 1 minus q that is simply 1 minus q. So, 1 minus q log 1 by 1 minus q, but these 2 terms together is simply H of p and these 2 terms together is simply H q. That is this is a H X and this is H Y. So, in this case since x and y are independent we assume they are independent and.

So, we have written the probability distribution or the product distribution here. And then we have derived the joint entropy of these two random variables and we have found that it is same as the some of these two entropies. So, for independent random variables we have seen that joint entropies are same as the sum of the entropies. And, for when there are dependent in particular for an example, we have seen that they when they are the entropy joint entropies not sum as is not same as the sum of the entropies. So, in the next class will continue this some more on information theory.

Thank you.