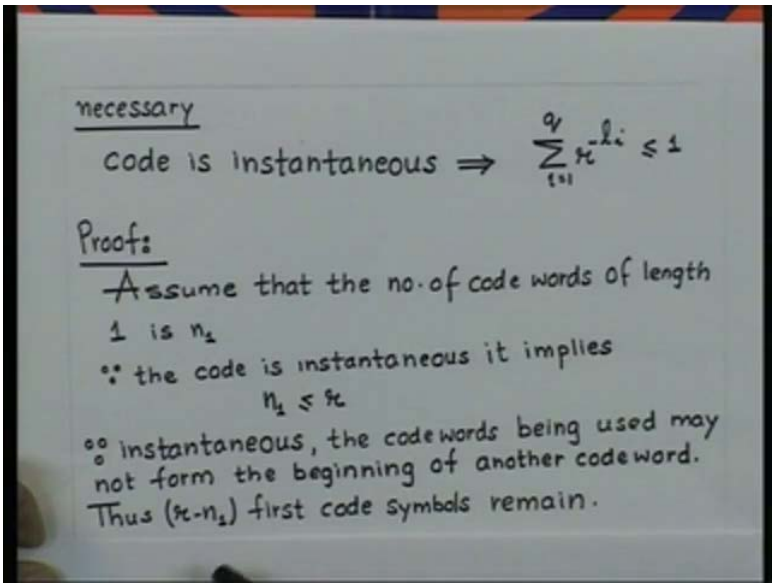
Information Theory And Coding
Prof S. N. Merchant
Electrical Engineering
Indian Institute of Technology, Bombay

Lecture - 9
Kraft-Mcmillan Equality and Compact Codes

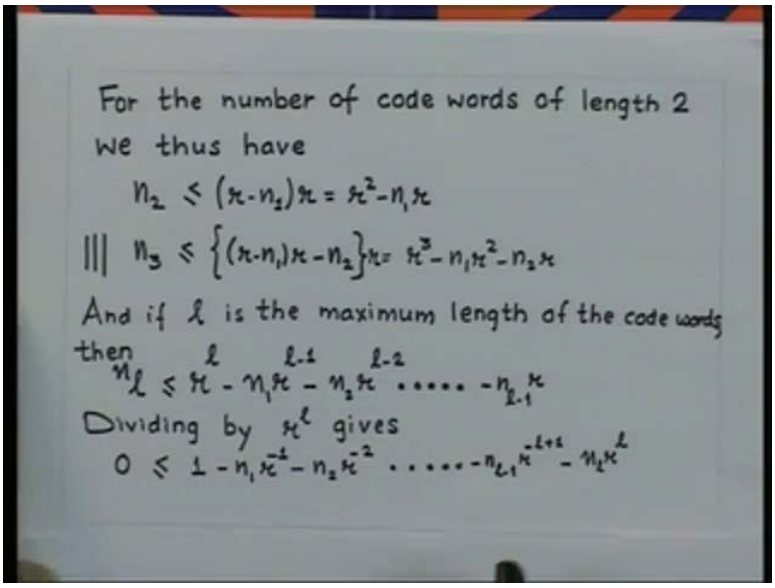
We have looked at the proof of a sufficient part of Kraft inequality. Now, we will look at the proof for the necessary part of the Kraft inequality.

(Refer Slide Time: 00:59)



The necessary part of the Kraft inequality says the, if the code is instantaneous, then it implies that summation of r minus l i for i equal to 1 to q is necessarily less than or equal to 1. Let us try to prove this relationship. The proof goes as follows. Let us assume that the number of code words of length 1 is n 1. So, because the code is instantaneous it implies that n 1 should be less than equal to r. Here, r is the size of the code alphabet. Now, because the code is instantaneous, the code words being used may not form the beginning of another codeword. Thus we have r minus n 1 first code symbols. Once I have chosen n 1 code words of length 1, the remaining r minus n 1 which can be used as a prefix for the code words.

(Refer Slide Time: 02:36)

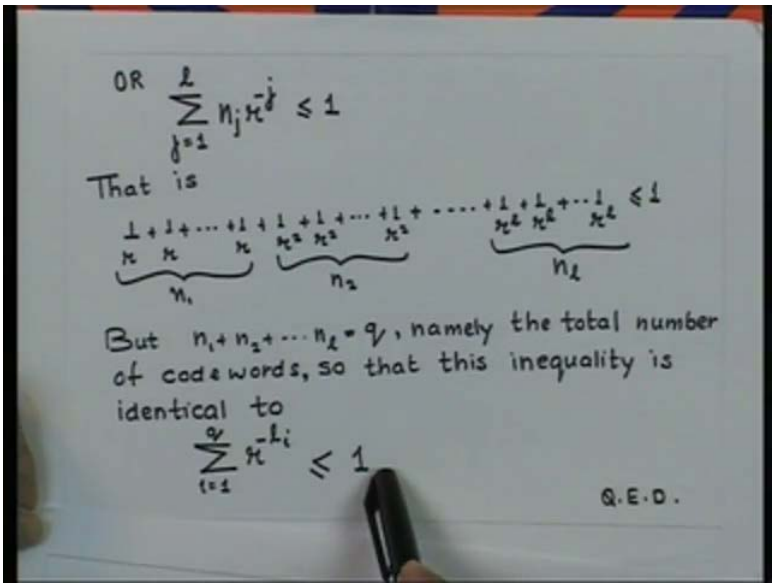


Similarly, for the number of code words of length 2, we have n 2 should be less than equal to r minus n 1 multiplied by r minus n 1 are the prefixes left out. This gets multiplied by the size of the code alphabet which is r, this has this is the number of code words which I can form of length 2. Now, since the code is instantaneous if I assume that then the number of code words which I have length 2, that is n 2 should be less than this quantity. If you simplify this quantity it turns out to be this.

Similarly, I can show that code word of length 3 that is n 3 should be less than equal to this expression. On the right-hand side here this again is the number of prefixes of length 2 which are left out. That gets multiplied by the size of the code alphabet. This total number gives me the number of code words of length 3 which I can form. Since, the code is instantaneous the actual number of the code words of length t3 which we have, we call n 3 should be less than this.

Now, if we assume that l is the maximum length of the code words. Then n l is the number of code words which we have of this maximum length l based on this derivation which we had done earlier. We can show that n l is less than this expression on the right-hand side. Here, if you take this expression and divide by r l we get this expression. It is after taking this quantity and dividing it by r l on the right-hand side that we get this quantity.

(Refer Slide Time: 04:54)



This quantity we can write as summation n j r minus j less than or equal to, where n j denotes the number of code words of length j. Here, j can range from 1 the maximum length which we have in the code. That is l. If you write this out we will get 1 by r plus 1 by r. The number of terms corresponding to 1 by r will be n 1 number of terms corresponding to 1 by r 2. This will be n 2 because this corresponds to code word of length 2. This corresponds to code word of length 1. Similarly, this corresponds to the number of code words of length r, l. So, this expression is equivalent to this expression which I have.

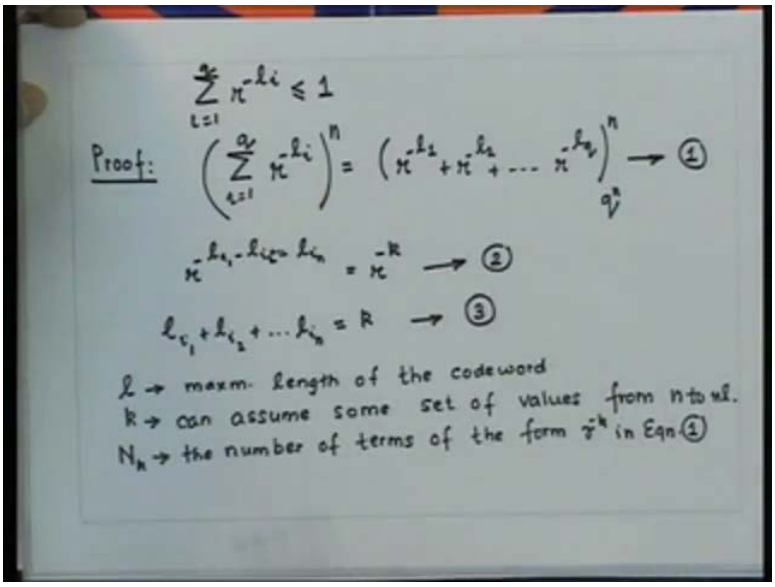
Now summation of n 1 plus n 2 plus n l should be equal to q. That is the size of the source alphabet which we have, so n 1 plus n 2 plus n l is equal to q. That is the total number of code word. That is this inequality which I have written here. It is identical to writing r minus l i less than equal to 1. This is a proof for the necessity part of Krafts inequality. We have seen that instantaneous code is a subclass of a uniquely decodable code.

What this implies that the sufficient part of the Kraft inequality should be also applicable to the uniquely decodable code. What I mean by that is that if this quantity is satisfied then I should be able to synthesize a uniquely decodable code. The next question is that, are there some restrictions on the length if the code. Is it uniquely decodable? We will

find that the necessary part of the Kraft inequality also applies to uniquely decodable code.

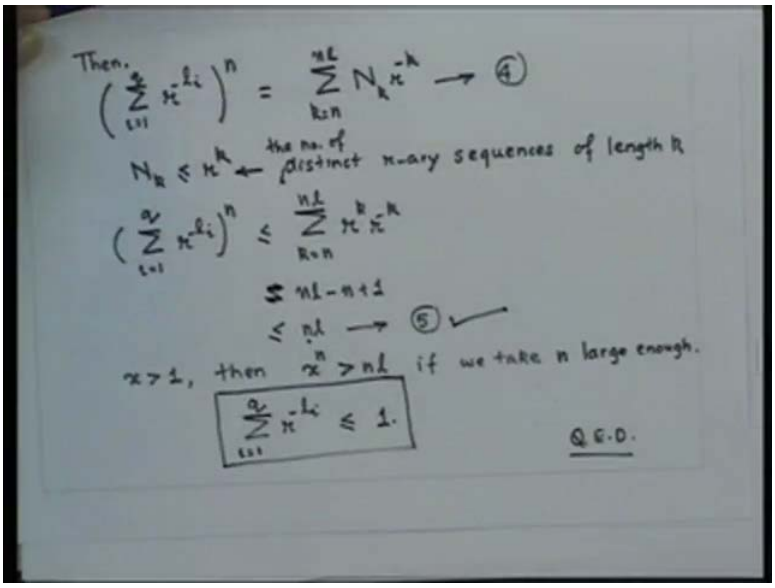
What I mean by that is that, if the code is uniquely decodable than like in the case of instantaneous code this inequality has to be satisfied. Let us try to prove this inequality for uniquely decodable code. Now, this inequality says that if a code is uniquely decodable then this is satisfied. This was first discovered by Mc Millan. What we are going to discuss is Mc Millan inequality. If you assume that a code is uniquely decodable

(Refer Slide Time: 08:11)



What I have to prove is this, is it true. The proof follows, as follows. Let me consider a quantity raise to n. Now, this is equivalent to writing r minus when I expand this equation. We will have 2 n terms each of the forms r minus l i 1 minus l i 2 minus. So when I expand this I will have q raise to n terms. Each term will be of the form given here. This will be equal to r raise to, let me call it r raise to minus k. Here by definition l i 1 plus l i 2 plus l i n is equal to, let me assume that l is the maximum length of the codeword in that uniquely decodable code. Now, k can assume some sets, some set of values from n to n l. Let us define n k as the number of terms of the form r raise to minus k in this equation 1. Then I can write this expression.

(Refer Slide Time: 11:45)

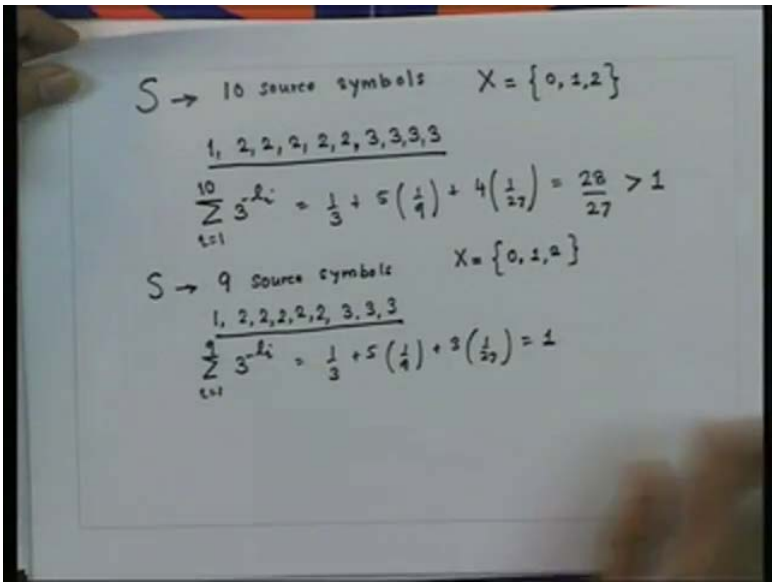


This expression can be written as summation of N k r raise to minus k, where k is equal to n to n l. Now, if you look at the expression equation number 3. We see that N k is also the number of strings of n code words that can be formed so that each string has a length of exactly k code symbols. If the code is uniquely decodable then this N k must be no greater than r raise to k. If the code is uniquely decodable N k has to be less than equal to r k. This r raise to k is the number of distinct r- ary. This is distinct r- ary sequences of length k. This is the number of distinct r- ary sequence of length k. Since our code is uniquely decodable N k has to be less than this quantity. It implies that r minus l i n i is equal one to q. This should be less than equal to r raise to k r raise to minus k is equal to n l. This simplifies to n l minus n plus. In fact, this is equal and this is less than equal to n l equation 5. It is a proof we seek. If k is greater than 1, then x raise to n will be always greater n l if we take n large enough.

Now, in our case equation 5 holds for any integer n. So what it implies is that x should be less than 1. Our case x is nothing but this quantity out here. This implies that r minus l i, i is equal to 1 to q should be less than equal to 1. If the code is uniquely decodable we have proved that this condition has to be satisfied. This condition is valid both for uniquely decodable code and instantaneous code. So, sometime the Kraft inequality is also stated as Mc Millan Kraft inequality. This is because it is true for both instantaneous code and uniquely decodable code. Now, before we go ahead let us look at a couple of

examples to understand the proof which we have provided in the Kraft and Mc Millan inequality.

(Refer Slide Time: 16:35)



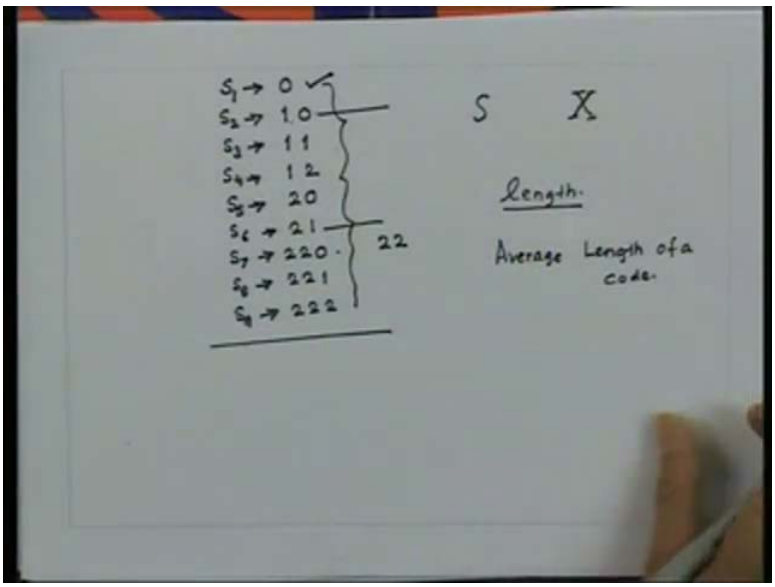
So, let us assume that I have a source consisting of 10 source symbols. Let us assume that this source symbols have to be coded using a code alphabet which is trinary in nature. So it has got 3 symbols 012. I want to code this source into a trinary instantaneous code. I have been also given the code lengths code word lengths for this source symbols. Let me assume that the code word lengths corresponding to this 10 source symbols are 122.

If you want this set of code word lengths and want to design a instantaneous code the first thing we have to do is basically check whether it satisfies the Kraft inequality. To check that, let us do it. Since, we have two 10 source symbols this turns out to be equal to one third plus 51 by 9 plus 41 by 27. This is equal to 28 by 27. This turns out to be greater than 1. What it means is that I cannot design an instantaneous code for this source with this code alphabet where the code word lengths are given by the set. It is not possible. So let us take another example.

Suppose, there was a source S consisting of 9 source symbols. Again my code alphabet consists of 3 symbols 012. I want to design instantaneous code with the code word lengths given. Now, before we start the designing of instantaneous code, it is essential to check whether this set of the code word lengths satisfies the Kraft inequality. Let us do

that. If you substitute the values into this expression we get this to be equal to 1. So, that implies that, it is possible for me to design an instantaneous code with this set of code word lengths. So let us do that.

(Refer Slide Time: 20:13)



I can take the first symbol S1. I will assign 0; to S2 I will assign 10. To S3 I will assign 11, to s4 I assign 12, to S5 20, to S6 is 21, To S7 is 220, To S8 is 221, to s9 is 222. Now, note how the construction of this code illustrates. The coding method used in the proof of the Kraft inequality, we use one prefix of length. Namely zero for the source symbol. S1 this left us with two prefixes of length 1 for the other code words.

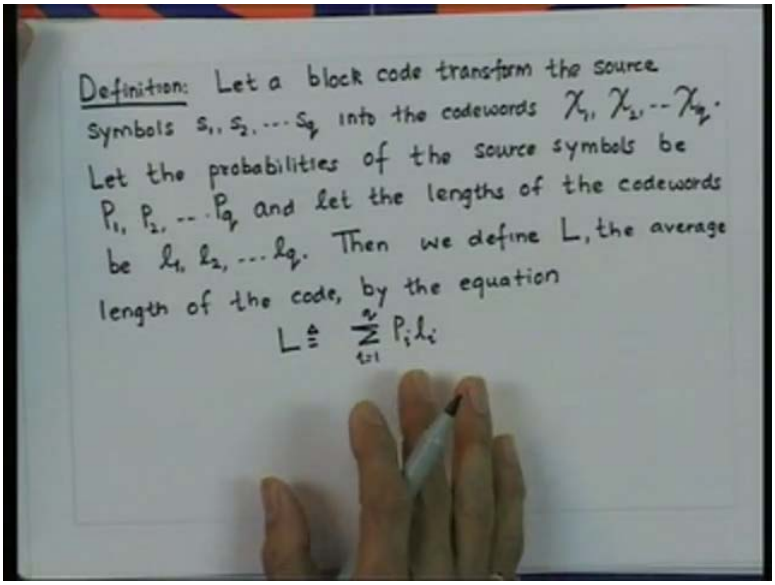
Now, what it means is that, well have 2 times 3. That is 6 permissible code words of length 2. Out of this we have used 5 for the source symbol from S2 to S6. This leaves us with a news prefix 22 for the remaining code words. So with this prefix which is left out and the code size to be 3, we have 3 more code words of code word length 3. Now, exactly we have 3 source symbols left out. So we can assign the remaining code words of length 3, to the remaining source symbol S7, S8 and S9. This helps us to understand the coding method used in the proof of the Kraft inequality.

Now, we have seen how to construct instantaneous code which maps source symbols from a source alphabet to code words. It consisting of code symbols from code alphabet. Now, for a given source S and for a given code alphabet X, we can form many instantaneous code or for that matter many uniquely decodable codes. With so many

abundance of available instantaneous and uniquely decodable code the natural question is how do you select the best code from this lot? To do that, perhaps you have to use some kind of a criterion.

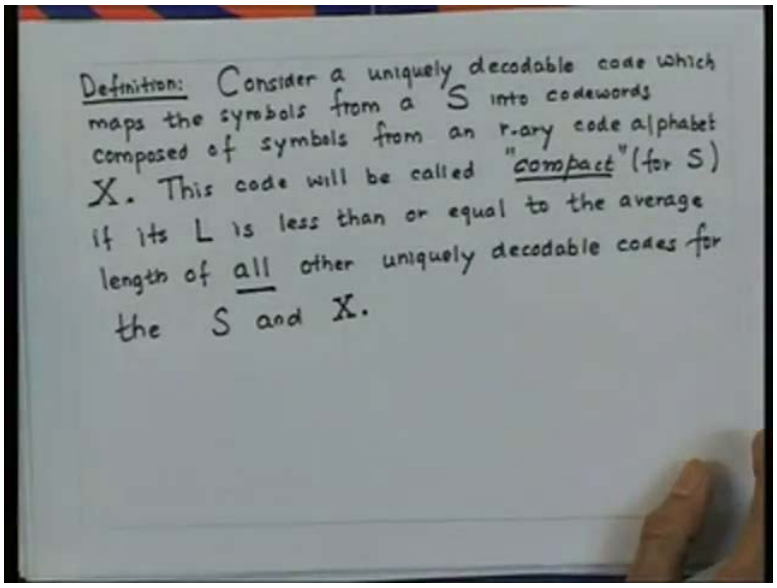
The natural criterion for the selection although by not the only possibility is length. So we can look at the length of the code words of a code to select a particular code. There are no other considerations from the standpoint of mere economy of expression and the resulting economy of communication system. We prefer a code with many short code words to the one with long code words. What it means is that we should define what the length of a code is. Now, a code consists of a code word and since each code word could be of different length we are to define what is known as an average length of a code. So let us define an average length of a code.

(Refer Slide Time: 25:04)



Let a block code transform the source symbols S1, S2 up to Sq into the code word X11, X21 up to Xq. Let the probabilities of the source symbol be P1, P2 up to Pq. Let the lengths of the code words be l1, l2 up to lq. Then we define the average length, which is denoted by l. This is the average length of the code by the equation, by definition. Now, we will not write here average, it is under stood that whenever I write here it means the average length. So, now what the problem reduces is that we should look at only those codes for which the average length of the code is as low as possible. Now, this leads us to another definition of what is known as a compact code. Let us define a compact code.

(Refer Slide Time: 28:16)



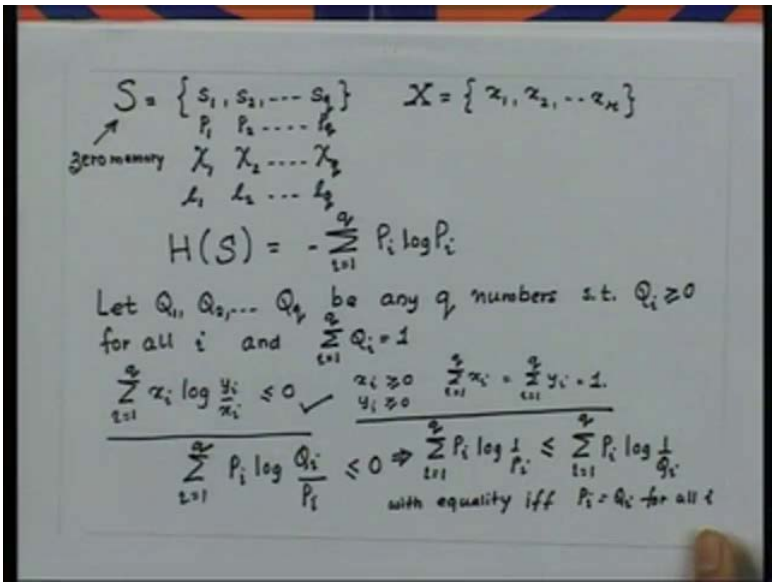
Consider a uniquely decodable code which maps the symbols from a source S into code words composed of symbols from an r-ary code alphabet X. Now, this uniquely decodable code will be called compact. The definition of the compactness is associated with the source S. So this code will be called compact for the source S, if it is average length which is denoted by l is less than or equal to the average length of all the other uniquely decodable codes. For the same source and for the same code alphabet this is a definition of compactness of a code.

Now, the fundamental problem of coding information sources is finding compact codes. What we will do basically, we restrict our search for compact codes to the class of instantaneous code which is a subclass of a uniquely decodable codes. The reason for doing this is that Mc Millans inequality assures us that any set of code word lengths available in a uniquely decodable code is also available in instantaneous codes. This is because of this inequality. We can look ahead for synthesizing of compact instantaneous code.

Now, this definition of every length is also valid for both 0 memory source and sources with memory including Markov sources. But, for our discussion at the moment we restrict ourselves to zero memory sources. Now, the question that comes to our mind is, is there any kind of relationship between the information measure and this average

length. In order to answer this question, let us examine little more in depth. So let me assume that

(Refer Slide Time: 33:08)



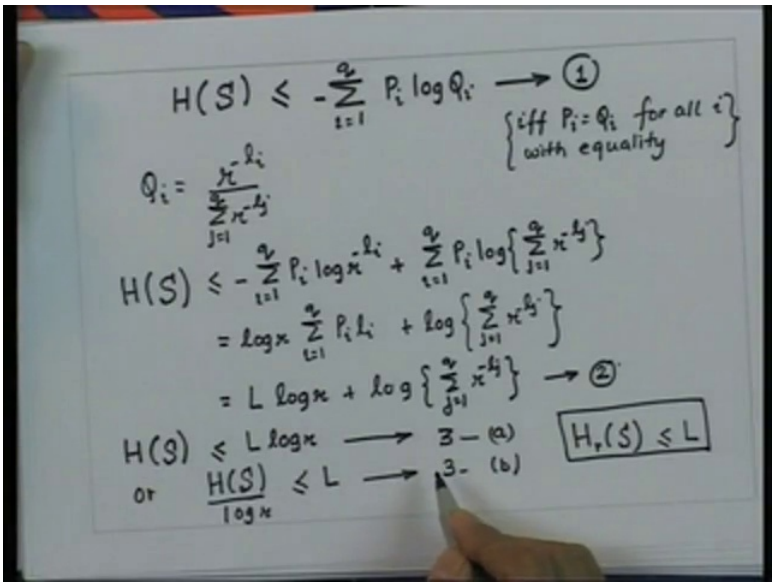
I have a source S with source symbols given as S 1, S 2 up to S q. This is a block code with a code alphabet X 1, X 2 up to X r. We have r-ary code alphabet and for the time being, let us assume that this source is a zero memory source. These source symbols are associated the probabilities P 1, P 2 up to P q. Each of this source symbols are associated the code words, composed of code symbols from this code alphabet X.

So the associated with this code words, we have the lengths for this code words as l1, l2 up to l q. Now, for this zero memory source we know that entropy of the source is given by, let Q 1, Q 2 up to Q q be any Q numbers such that each Q i is greater than equal to 0 for all i. We also assume that the summation of Q i is equal to 1. Now, we have earlier in this course seen an inequality which says that summation of X i log of Y i. Xi is equal to 1 to q is always less than equal to 0, provided X i greater than 0. Y i is greater than 0 and summation over X i and summation over Y i is equal to 1.

So, if these conditions are satisfied than this is a valid inequality. Now, considering P i and Q i in place of X i and Y i respectively. We can write summation of P i log of Q i, P i is less than or equal to 0, which implies that summation of P i log 1 by i is equal to 1 to Q is less than or equal to summation of p I log of 1 by Q i is equal to 1 to Q. Now, this

will be equality. With equality if and only if, P i is equal to Q i for all i, this expression out here is your H S so on the left hand side.

(Refer Slide Time: 38:21)



I can write and as H S is less than equal to summation of P i log of Q i. So this becomes negative. So, with equality if and only if, P i is equal to q i for all i. This condition is with equality. Now, equation 1 is valid for any set of non-negative number S q i which sum to 1. We may therefore, choose q i to be r minus l i over r minus l j. L j is equal to 1, if you choose our non negative q I as given by this expression. We can write H S, this equation number 1 as less than equal to minus summation of P I logo r plus Pi log of substituting this into this, we get this expression.

This expression on simplification will result into the following expression plus log of summation. This is because summation of P i equal to one to q is equal to 1. Therefore, this expression reduces, to this expression and this by definition of the average length is l log r plus log. Now, if we have the requirement that our code is instantaneous then the Kraft Inequality tells us, that this quantity out here should be less than equal to 1.

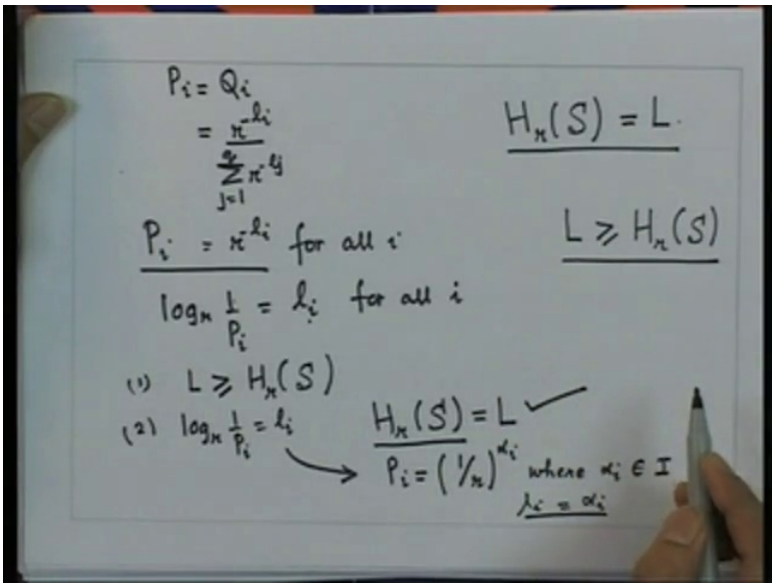
Now, if this quantity is less than equal to 1 than the logarithm of the quantity will be less than or equal to 1. This will always be less than equal to 0. In that case, we can write this as a H S is less than equal to l log r. We can say this two, and this is expression 3 a. H S by log r is less than equal to l. H S is measured in bits l i' s the average number of r-ary symbols. We used to encode S now, if you measure the entropy in r-ary units then

equation 3 can be written as H r S. This equation out here, when I write H S in terms of r-ary units less than equal to, now it should be noted that equation 3 constitutes a milestone in our study of information theory. This equation is the first instant which demonstrates the connection between our definition for information measure and a quantity in this case l. This does not depend upon that definition. So this equation provides some kind of a justification for the definition of our information measure which with we had started earlier in the course.

Now, on the surface equation 3, nearly presents us with a bound on the average length of a code. What it says is that, every length of an instantaneous code which is given by l has to be always greater than or equal to the entropy of the source measured in r-ary units. Now, in certain cases however, it is possible to show much more from the simple arguments leading to this equation.

So, let us carefully examine the conditions for equality in this expression out here. We have derived now; the inequality was introduced at 2 points. First, at this point that is, equation number 1 and the other was at this point. So, in this, if we assume that the code is instantaneous then we assume one more thing. That is, the summation j minus j equal to 1 r minus r j l j is equal to 1. Then this will be an equality. This may not be dropped out from this expression, so in this equation a necessary condition for the equality rather than inequality is this term out here. It should be equal to 1 and then retracing our steps to this equation we see that necessary.

(Refer Slide Time: 46:49)



And certain sufficient condition for equality is P i should be equal to Q i for all i. What this implies is that, equal to Q, and this is equal to 1. So if i have P i satisfying this relationship for all, i then I will get the equality that is H r S is equal to l. This also means that log of r 1 by P i is equal to l I for all i. Now, summarizing 5 instantaneous codes and a zero memory source, we will always have l greater than equal to H r S. Furthermore, l can achieve this lower bound of H r S. If and only if, the code word lengths l i equal to log to the base r of 1 by P i for all i. So it means that if this condition is satisfied for all i then I will achieve the minimum length and that minimum length is the entropy of the source measured in ra R V unit.

Now, this condition is an equivalent to saying that, P i must be of the form 1 by r alpha i, where alpha i is positive integer. Now, with this result we have derived an additional condition for the word lengths of a compact code. What it says, P i of this form, where alpha i are the integers then not only will be satisfied, but we have also derived a code word lengths l i of a compact code.

The l i is nothing but equal to alpha i and having obtained the code word lengths the construction of a compact code follows the procedure of the example discussed earlier in today's class. We designed a code for a source consisting of 9 symbols and a code alphabet consisting of trinary symbols. Now, let us try to answer some of the coding questions raised earlier in the light of what we have studied today. An important

relationship which says that l should be always greater than equal to H r S. So in the light of what we have derived today, let us answer some of the coding questions which we had raised earlier. We will try to do this with the aid of some examples and this we will have a look at, in the next class.