

**Principles of Communication Systems - Part II**  
**Prof. Aditya K. Jagannatham**  
**Department of Electrical Engineering**  
**Indian Institute of Technology, Kanpur**

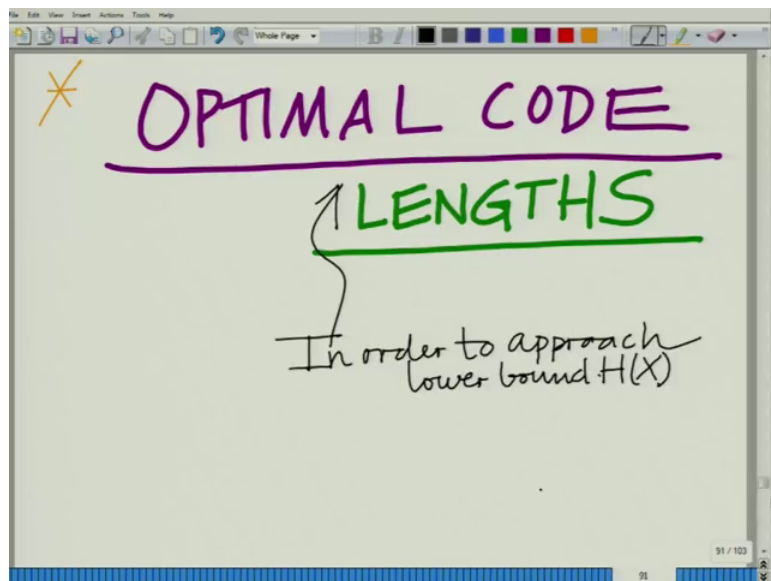
**Lecture - 45**

**Optimal Code Length, Constrained Optimization, Morse Code Example**

Hello, welcome to another module in this massive open online course. So, we are looking at the average code length for code and we were shown the fundamental bound, where in the average code length is lower bounded by the entropy of the source, the average code length. So, the entropy of the source is the lowest possible average code length of any prefix free code that can be designed for a given source.

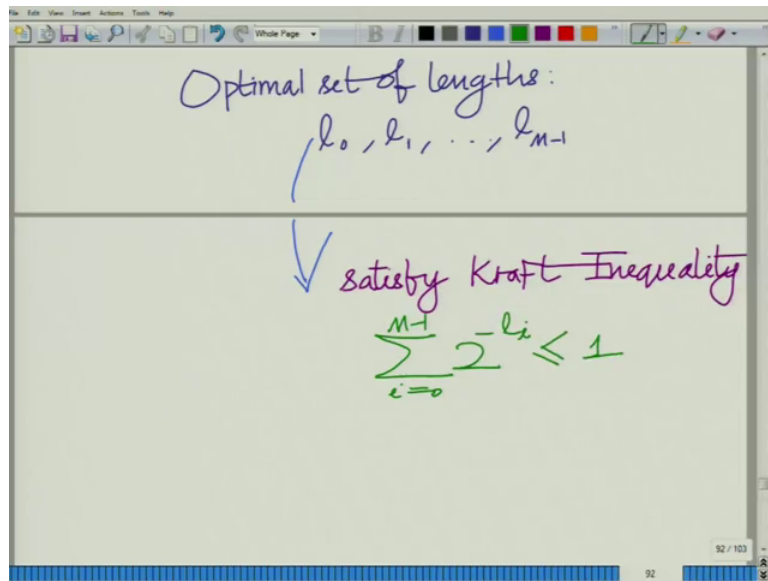
Now let us look at what are the optimal codeword lengths which can be used to approach this minimum possible average code length all right.

(Refer Slide Time: 00:55)



So, in this module what we would like to look at is would like to address the question of, what are the optimal if I were to design such an codeword code with minimum average code length minimum average, minimum average code length, what are the optimal code lengths? All right, and the idea is basically in order to approach, in order to over approach lower bound which is of course, we have seen is given by the entropy  $H(X)$  of the source.

(Refer Slide Time: 01:54)



So, we want to find the optimal set of lengths, optimal set of codeword lengths  $l_0, l_1, \dots, l_{m-1}$  of course, now since we are designing the prefix free code remember this has to satisfy the Kraft inequality. These codeword lengths have to satisfy, these codeword lengths have to satisfy Kraft inequality. That is summation  $i$  equal to 0 to  $m-1$   $2^{-l_i}$  has to be less than or equal to 1. So, we can formulate this to find the optimal codeword lengths we can formulate this as an optimization problem, right? We want to find the optimal value of any function we can formulate as an optimal. Here now optimization the we would like to optimize the codeword length, the in particular we would like to minimize this average code length all right.

(Refer Slide Time: 03:39).

The image shows a handwritten mathematical formulation of a constrained optimization problem on a whiteboard. At the top, the title "Constrained Optimization" is written in purple and underlined. Below it, the text "problem for minimizing average code length." is written in purple. The objective function is given as  $\min \sum_{i=0}^{M-1} p_i l_i$ , with a purple arrow pointing to it labeled "objective". The constraint is given as  $\text{s.t. } \sum_{i=0}^{M-1} 2^{-l_i} \leq 1$ , with a purple arrow pointing to it labeled "constraint". There is a small green mark  $l_i \geq 0$  at the top center. The whiteboard interface includes a toolbar at the top and a status bar at the bottom showing "92 / 103".

So, we can formulate this is an object optimization problem, subject to this constrain there is a constrain remember a such prefix the lengths of any such prefix free code design have to satisfy the craft inequality.

So, the craft inequality access the constraint all right. So, therefore, we will we will have a constrained optimization problem for minimizing the average code length all right. So, we can formulate this as a constrained. So, that is the key word here that is, this is the constrained optimization problem. And what is the optimization problem? The optimization problem is specifically minimize the average code length which is summation  $i$  equal to 0 to  $m$  minus 1  $p_i l_i$  subject to the constraint which is the craft inequality,  $i$  equal to 0 to  $m$  minus 1  $2^{-l_i} \leq 1$ . This is known as the objective of the optimization problem, this is known as the, this is known as the objective, this is known as the constraint.

(Refer Slide Time: 05:09).

We can use Lagrange multiplier framework to solve problem above

$$F = \sum_{i=0}^{M-1} p_i l_i + \lambda \left( \sum_{i=0}^{M-1} 2^{-i} - 1 \right)$$

Lagrange multiplier

$\frac{\partial F}{\partial p_i}$  ← Differentiate wrt to each  $p_i$  and set equal to 0.

And now what we can do is this is a constraint optimization problem. So, we can use the Lagrange, we can use the Lagrange multiplier, we can use the Lagrange multiplier framework, the Lagrange multiplier framework. There also known as the KKT framework the Karush–Kuhn–Tucker framework to solve a constraint optimization problem which basically use as a Lagrange multiplier.

So, basically we can use a Lagrange multiplier. Let us put it this way we can say rather than Lagrange multiplier, you can use a Lagrange multiplier framework to solve the optimization problem above. And what is the Lagrange multiplier framework? We form a Lagrangian with the objective the must be familiar with this from a basic course on mathematics or even an advance course on optimization. So, I found the Lagrangian which comprises of the objective summation  $p_i l_i$  plus lambda times the constraint summation  $i$  equal to 0 to  $m$  minus 1  $2^{-i} - 1$ . Now this lambda this is known as the Lagrange multiplier, the Lagrange multiplier, the Lagrange lambda is known as the this is known as the Lagrange multiplier. Now what I am going to do I am going to differentiate with respect to each  $p_i$  ok.

So, what we are going to employ is from the KKT framework the Karush–Kuhn–Tucker framework, what we do is differentiate this way of the Lagrangian which comprises of the objective plus lambda which is the Lagrange multiplier times the constraint. Remember this is your constraint. You have to have one Lagrange's Lagrange multiplier

for each constraint, we have only one constraint. So, we have a single Lagrange multiplier and now we are differentiating with respect to each  $p_i$  and set equal to 0. And differently with respect to each  $p_i$  and set equal to 0. Now when you differentiate with respect to  $p_i$  of course, you can say summation  $p_i l_i$  if you differentiate with respect to  $p_i$  you get  $l_i$  plus.

(Refer Slide Time: 08:17)

$$p_i + \lambda 2^{-l_i} (-1) \ln 2 = 0$$

$$\Rightarrow 2^{-l_i} = \frac{p_i}{\lambda \ln 2}$$

$$\Rightarrow l_i = \log_2 \frac{\lambda \ln 2}{p_i}$$

Now, when you differentiate lambda summation  $2$  to the power of minus  $l_i$  that fix  $2$  to the power of minus  $l_i$  of course, if you differentiate any other  $2$  to the power of minus  $l_j$  with respect to  $l_i$  that derivative is 0. Similarly, if you differentiate any other  $p_j l_j$  with respect to  $l_i$  the differentiate derivative is zero.

So, you were picking that particular  $i$  when you differentiating with respect to  $l_i$  plus lambda times derivative of  $2$  to the power of minus  $l_i$  is  $2$  to the power of minus  $l_i$  into derivative of minus  $l_i$  which is  $1$  times log that is  $\log 2$  to the base  $e$  or that is basically  $1$  into  $\log$  natural  $2$ . That is basically the natural logarithm of  $2$  and this must be equal to  $0$ . I am sorry, when you differentiate  $p_i l_i$  this is simply  $p_i$ , differentiate  $p_i l_i$  with respect to  $l_i$  is equal to  $p_i$ . So, this is equal to  $0$  which implies that of course, if you differentiate here lambda into minus  $1$  minus lambda with respect to  $l_i$  that derivative is  $0$ . Because lambda is Lagrange multiplier differentiated with respect to  $l_i$  that derivative is  $0$ . So, we have this equation over here now we have solve this to obtain  $p_i$ .

So, what we have is basically 2 to the power of minus  $l_i$  if you look at that that is equal to well or 2 to the power of minus  $l_i$  is equal to  $p_i$  divided by  $\lambda \ln 2$ . Which implies  $l_i$  equals log to the base 2 of  $\lambda \ln 2$  divided by  $p_i$ . That is easy to show. So, this is the optimal average length.

(Refer Slide Time: 10:40)

The image shows a whiteboard with handwritten mathematical derivations. At the top, it states  $2^{-l_i} = \frac{p_i}{\lambda \ln 2}$ . Below this, the equation  $l_i = \log_2 \left( \frac{\lambda \ln 2}{p_i} \right)$  is boxed in orange. A red arrow points from the boxed equation to the text "Optimal codeword length." Another red arrow points from the boxed equation to the text "Still have to find  $\lambda$ ." Below this, it says "To find  $\lambda$  use constraint" followed by the equation  $\sum_{i=0}^{n-1} 2^{-l_i} = 1$ . The whiteboard interface includes a menu bar at the top with "File", "Edit", "View", "Insert", "Actions", "Tools", and "Help". A toolbar with various drawing tools is visible below the menu bar. The bottom right corner of the whiteboard shows "94 / 103".

Now, solving this Lagrangian setting the derivative differentiating with respect to  $p_i$  setting equal to 0, we have obtained the optimal, optimal codeword length. What average length optimal, but of course, we have to find Lagrange multiplier, we still have to find the Lagrange multiplier  $\lambda$ , still have to find  $\lambda$  and we find  $\lambda$  from the constraint.

(Refer Slide Time: 11:33)

Still have to find  $\lambda$ .

To find  $\lambda$  use constraint

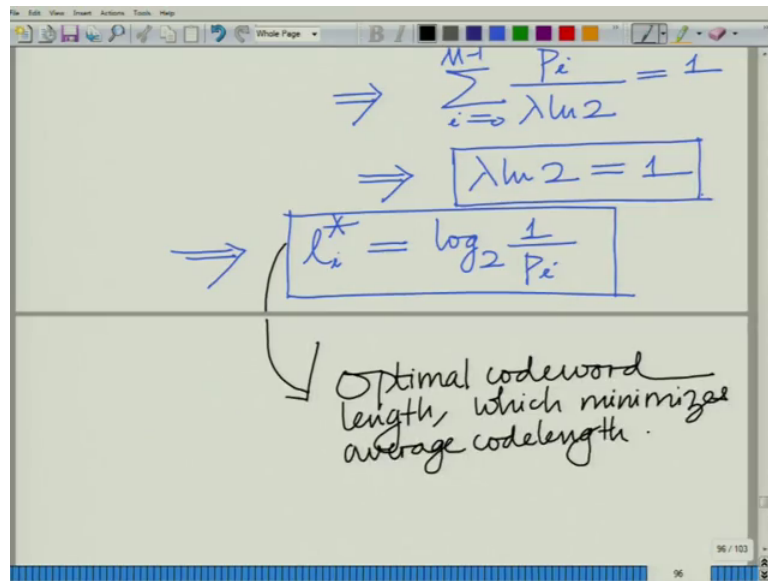
$$\sum_{i=0}^{M-1} 2^{-l_i} = 1$$

substitute  $l_i$  from  $\#$

$$\sum_{i=0}^{M-1} 2^{-\log_2 \frac{\lambda \ln 2}{p_i}} = 1$$
$$\Rightarrow \sum_{i=0}^{M-1} \frac{p_i}{\lambda \ln 2} = 1$$
$$\Rightarrow \boxed{\lambda \ln 2 = 1}$$

To find lambda, We use the constraint that is summation  $i$  equal to 0 to  $m$  minus 1  $2$  to the power of minus  $l_i$  is equal to 1. Remember this is the constraint which is satisfied with equality, if we have a positive Lagrange multiplier. And now you substitute this value of  $l_i$  which is summation  $i$  equal to 0 to  $m$  minus 1  $2$  to the power of minus of course,  $l_i$  I substitute from let us call this equation has hash. So, substitute. So, in this substitute  $l_i$  from equation hash. So, that will be  $2$  to the power of minus  $\log$  to the base 2  $\lambda \ln 2$  divided by  $p_i$  is equal to 1. Now  $2$  to the power of minus  $\log$  to the base 2  $\lambda \ln 2$  divided  $p_i$  this is nothing but summation  $i$  equal to 0 to  $m$  minus 1 this quantity is  $p_i$  divided by  $\lambda \ln 2$ , which is equal to 1 now we know summation  $p_i$  is equal to 1 this implies simply that  $\lambda \ln 2$  is equal to 1 ok.

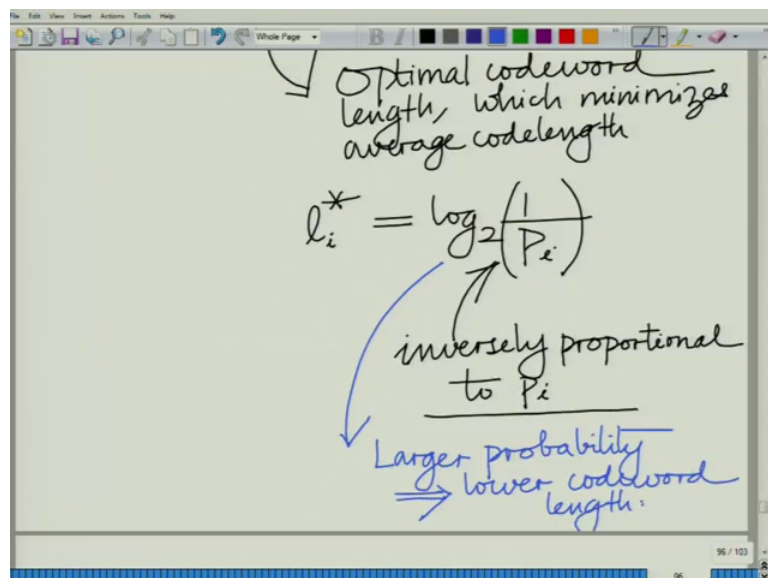
(Refer Slide Time: 13:29)



A screenshot of a digital whiteboard showing a handwritten derivation. At the top, the equation  $\sum_{i=1}^{M-1} \frac{P_i}{\lambda \ln 2} = 1$  is written. Below it, the constraint  $\lambda \ln 2 = 1$  is boxed. Then, the optimal codeword length  $l_i^* = \log_2 \frac{1}{P_i}$  is boxed. A checkmark is placed to the left of the boxed equation. Below the box, a note reads: "Optimal codeword length, which minimizes average code length."

So, that is the equality that we have using the constraint and therefore, this finally, substituting the value of this lambda  $\ln 2$  the optimal length which we can call as  $l_i^*$  equals  $\log$  to the base 2  $\lambda \ln 2$  which is 1 divided by  $P_i$ . So, optimal codeword length and this is interesting, the optimal codeword length what we have derived is the optimal codeword length that is, optimal codeword length which minimizes, which minimizes the average code length.

(Refer Slide Time: 14:29)



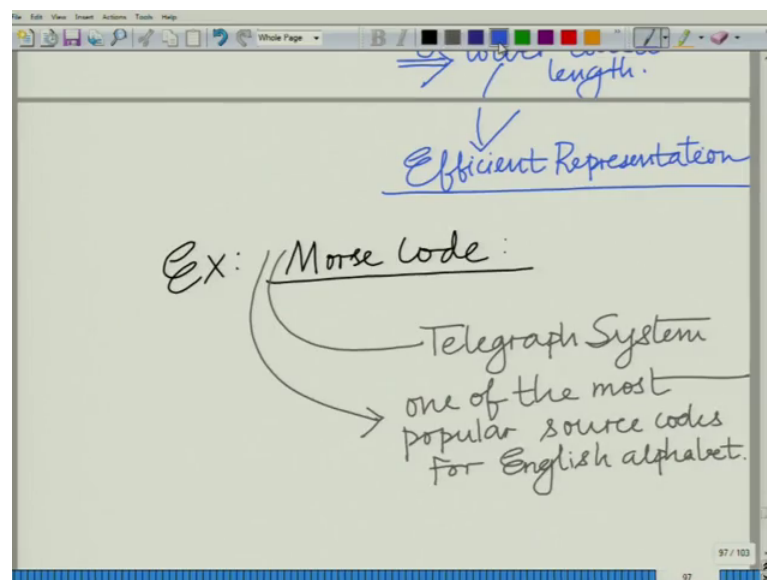
A screenshot of a digital whiteboard showing a handwritten explanation. At the top, a note reads: "Optimal codeword length, which minimizes average code length" with a checkmark. Below it, the equation  $l_i^* = \log_2 \left( \frac{1}{P_i} \right)$  is written. A blue arrow points from the text "inversely proportional to  $P_i$ " to the  $P_i$  in the denominator of the equation. Another blue arrow points from the text "Larger probability  $\Rightarrow$  lower codeword length" to the entire equation.



And observe that  $l_i$  is equal to  $\log_2 \frac{1}{p_i}$  this is inversely proportional, look at this. So, this is inversely proportional to  $p_i$  which means the larger the probability that is symbol which have a larger probability have to be represented with lower and lower codeword lengths. Because they occur more frequently an efficient representation would be to represent them using code words which have the, which have lower lengths.

So, the largest probability symbol naturally has to have the lowest code length, codeword length for efficiency. And that is what there is a optimal length that is, what there is the optimal lengths that we have derived using this paradigm that is the Lagrange multiplier using the Lagrange multiplier the lagrangian that is KKT frame work. Using by solving this constraint optimization problem, the optimal lengths that we have derived codeword lengths that we have derived reflect this very fact. So, this implies this implies larger probability, So this says that larger probability implies lower, larger probability implies lower code. This is for efficient representation, this for in towards in efficient. And let me aimed by giving your simple example.

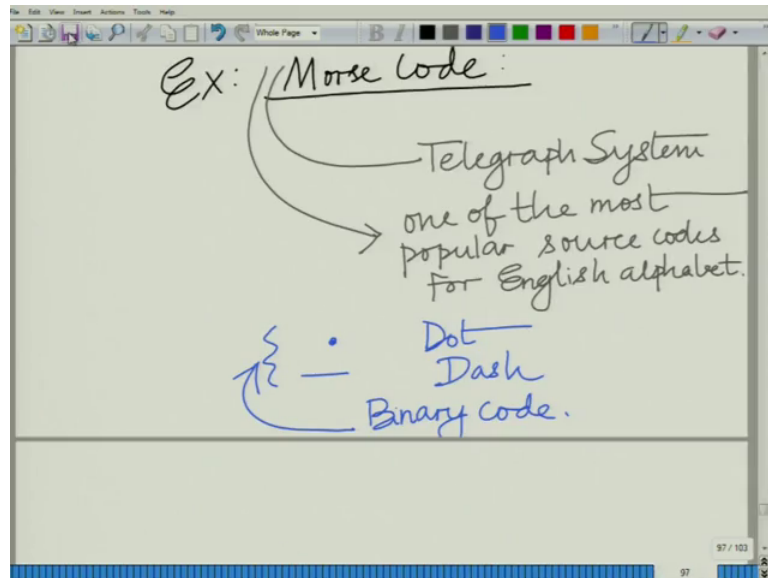
(Refer Slide Time: 16:19)



For example let us look at a Morse code one of the earliest and most popular code source codes for the English alphabet used in the of course, everyone must be familiar with the Morse code which is used in the telegraph and one of the most popular source code for English alphabet. And this is nothing but a source code, correct? One of the most

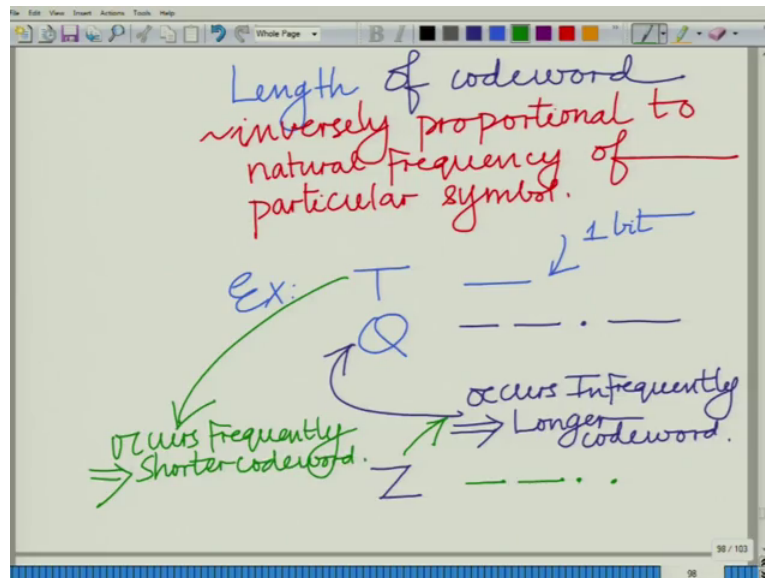
popular, and if you look at the Morse code instead of zeros and ones see of 2 kinds of symbols, you can think of them as zeros and ones you have a dot and you have a dash rather than use. So, you have a dot.

(Refer Slide Time: 17:32)



So, you have a 2 bit. So, you have a binary code. So, you have a binary code. So, you have dots and dashes. So, Morse code can be thought of binary code were each binary code corresponded to the English alphabet and of course, the numbers all right. So, each alphabet and each number is represented by using a series of sequence of dots and dashes and you will see that in the Morse code the length of a codeword is roughly,

(Refer Slide Time: 18:14)



Of course the Morse code was derived more than hundred years ago before information theory was in practice, length of the codeword is roughly, let say approximately, one of the reasons Morse code is very efficient is inversely proportional to the frequency, correct? To the natural frequency of particular symbol. For example, if you can look at the symbol T this is represented using dash. You can think of this as a one bit because T occurs rather frequently in the English alphabets. Several alphabets use T for instants. In fact, the word the uses as T, so many words. So, the word T occurs rather frequently and therefore, it has a very compact representation, a single dash.

Now, some of the alphabets which are the rather in frequently for instead the most common one that you can think of is the alphabet Q if you look at Q therefore, has a longer representation that is dash, dash, dot, dash. So, this is occurs in frequently implies longer codeword. In fact, if you look at z that is also which also occurs infrequently that is dash, dash, dot, dot. Again occurs in frequently.

Therefore, it has a longer codeword this is occurs, T occurs frequently implies shorter codeword. So, that is the basic idea. So, even a code like the Morse code, which was derived, which was of course, established which was designed hundreds of years ago for and which. In fact, on the which In fact, from the back bone of the telegraphy system was in fact based on this principle much before of course, the principles of information theory was formalized that longer, which if we think about is also intuitive that is more

frequently occurring symbols have to be represented by code words having smaller lengths, right? And the less frequently that is the rather infrequent symbols can be represented using a code words which have possibly longer lengths. And that is how one designs an efficient code. And that same is verified by the optimization framework that we derived solution to the optimization problem for minimizing the average codeword length, where we are shown that the lowest average codeword length is achieved using codeword length  $l_i$  which is in fact, which is a log which is given as the log to the base 2  $1/p_i$ .

Of course, we have shown that using a constrained optimization framework all, right? Using the KKT framework formulating the Lagrangian using the Lagrangian multiplier, the Lagrange multiplier solving this, right? Solving the Lagrangian to obtain the optimal codeword lengths  $l_i$  which is given by  $\log_2 1/p_i$  all right. So, will stop here and in the subsequent models we look at schemes to design efficient codes in order to achieve this lower bound all right.

Thank you very much.