Elements of Information Theory

for

EE511

by

A. DAHIMENE

Institut de Génie Electrique et Electronique

Université M'Hamed Bougara,

Boumerdes

Content

Elements of Information theory

1. Introduction

In this set of course notes, we are going to have a brief overview of the theory of information. The information is defined quantitatively and not qualitatively. This means that we will measure how much information is transferred in a communication system to an end user and not what he will do with this information. As was stated in previous courses, randomness is an important aspect of communication theory. If the destination of the message knows a priori the message, there is no point in performing the operation of transmission. To try to have an intuitive notion of the concept of information, let us consider the following example.

During summer, we receive the following weather forecast: "tomorrow will be sunny". It is evident that this message does not give us much information since it tells us about a highly probable event. However, if the message is: "There will be a thunderstorm tomorrow", then it contains a lot of information. This is due to the fact that the described event is quite rare in summer. From this example, we can conclude that the more uncertain event is the one that contains the highest amount of information.

2. Definition

Consider a discrete random variable X taking values from a finite set $\{x_1, x_2, \ldots, x_M\}$[1]. The probabilities of the events $\{X = x_i\}$ are $P[x_i] = p_i$. The information provided by the occurrence of the "*symbol*" x_i is defined by the following function $I[x_i]$ of the probability p_i satisfying the following axioms:

1. $I[x_i] \geq 0$

[1] We will call the random variable and the set of values by the same name X.

2. If the two events $\{X = x_i\}$ and $\{X = x_j\}$ are independent, the information provided by occurrence of the pair of symbols x_i and x_j is: $I[x_i x_j] = I[x_i] + I[x_j]$

3. The function $I[x_i]$ is a continuous function of the probability p_i.

Let us call this function f. Then, from $P[x_i x_j] = P[x_i]P[x_j] = p_i p_j$, we can write:

$f(p_i p_j) = f(p_i) + f(p_j)$. The only continuous function of its argument having the above property is the logarithm function. So, we can write: $I[x_i] = k \log_a p_i$. The constant k must be negative because the probability has a value that is less than one. The base of the logarithm depends on the unit selected for measuring the information. In this course, we are going to measure it in bits (**bi**nary uni**t**) and we adopt as a definition:

$$I[x_i] = -\log_2 p_i \tag{1}$$

If we use Neperian logarithms, information will be measured in nats (**na**tural uni**t**) and if we use base 10 logarithms, the information is measured in Hartley's. Hartley was the first engineer who proposed the use of logarithms to measure information. However, Hartley used the log of the cardinal of X to define the average information provided by the realization of one symbol of X.

Since we have decided to use "bits" as units, the base of the logarithm is two and does not need to be indicated. Before we go on, we have to give a more precise definition of information sources.

3. Information source

An information source can be considered as a generator of a random process. The nature of the source depends on the generated stochastic process. If the process takes values from a discrete set, the source is discrete. If the process takes values from a continuum, the source is an analog source. Discrete sources generate sequences of values from a discrete set called "Alphabet". In general, we consider finite alphabets. An analog source can generate a sequence of numbers (discrete time stochastic process) or a continuous time process. In this course, we are going to consider first discrete source with finite alphabets.

Consider a finite alphabet $X = \{x_1, x_2, \ldots, x_M\}$. An element of X is called a "*symbol*". A sequence of symbols is called a "*message*". For example, a message consisting of a sequence of n symbols from X is $\xi_1 \xi_2 \ldots \xi_n$ where ξ_k can take values x_{ki} from X.

Stationary source:

If the process generated by the information source is stationary, the source is stationary. A discrete stationary source is such that: $\forall n_0 \in \mathbb{Z}$

$\Pr\{\xi_{k1} = x_{i1}, \xi_{k2} = x_{i2}, \ldots, \xi_{kn} = x_{in}\} = \Pr\{\xi_{k1+n_0} = x_{i1}, \xi_{k2+n_0} = x_{i2}, \ldots, \xi_{kn+n_0} = x_{in}\}$ where the values x_{ij} are taken from the alphabet *X*.

Discrete Memoryless Source (DMS):

If the symbols constituting the messages produced by the source are independent, the source is memoryless. A discrete memoryless source is completely characterized by the set of probabilities of the different symbols corresponding to the alphabet. For a DMS, the probability of a message is simply the product of the probabilities of the symbols composing the message.

Consider now an arbitrary DMS. It generates messages from a given alphabet *X* composed of *M* different symbols, each one with a probability of occurrence $\Pr[x_i] = p_i$. If a symbol x_i has a probability $p_i = 1$, then there is no information gained by the occurrence of the symbol x_i since $I[x_i] = -\log 1 = 0$ [2]. This is due to the fact that there is no uncertainty about the fact that the source will produce the symbol. So, we can say that ***the information gained by the occurrence of a symbol is equal to the uncertainty that we had before the realization of the event*** $\{X = x_i\}$[3]**.**

4. Entropy

Let us consider a very long message generated by a DMS: $u = x_{k1}x_{k2}\ldots x_{kn}$ where the symbols x_{kj} are taken from the alphabet $X = \{x_1, x_2, \ldots, x_M\}$. The message is constituted by a sequence of *n* symbols where *n* is very large. So, it contains n_1 times the symbol x_1, n_2 times the symbol x_2, up to n_M times the symbol x_M. It is evident that $n_1 + n_2 + \cdots + n_M = n$. Given that the source is memoryless, the information provided by the message is the sum of the amount of information of the individual symbols.

---

[2] It is understood that the log is taken with base 2.

[3] As stated before, we denote the random variable and the alphabet by the same letter.

$$I\left[x_{k1}x_{k2}\ldots x_{kn}\right]=n_1 I\left[x_1\right]+n_2 I\left[x_2\right]+\cdots+n_M I\left[x_M\right]$$
$$=-\left[n_1\log p_1+n_2\log p_2+\cdots+n_M\log p_M\right]$$

We can factorize n in the above expression:

$$I\left[x_{k1}x_{k2}\ldots x_{kn}\right]=n\left[-\frac{n_1}{n}\log p_1-\frac{n_2}{n}\log p_2-\cdots-\frac{n_M}{n}\log p_M\right]$$

The length n of the message being very long, the ratios $\frac{n_k}{n}$ are approximately equal to the symbol probabilities p_k. We can write:

$$I\left[x_{k1}x_{k2}\ldots x_{kn}\right]\approx n\left[-\sum_{k=1}^{M}p_k\log p_k\right]$$

The expression between brackets indicates an average information per symbol generated by the source. This average is a global characterization of the source. This quantity is called "***Entropy***" and is denoted $H\left(X\right)$.

$$H\left(X\right)=-\sum_{k=1}^{M}p_k\log p_k \qquad (2)$$

The entropy is measured in bits/symbol.

Example: Consider a binary source with the alphabet $X=\left\{x_1,x_2\right\}$ with probabilities $p_1=p$ and $p_2=1-p$. Its entropy is given by:

$$H(X)=-p\log p-(1-p)\log(1-p)$$

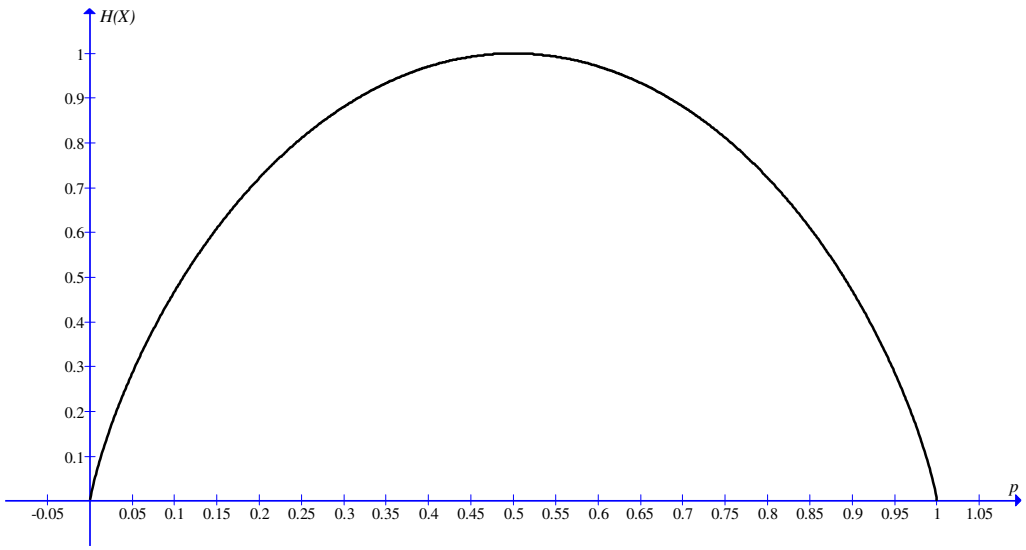


Figure 1 Entropy of a binary source

From Figure 1, we observe that the entropy is maximal for $p=\frac{1}{2}$, i.e. when the two symbols are equiprobable. For this value of probability, the average uncertainty is 1 bit/symbol. Equiprobable symbols from a binary source produce 1 bit of information at every occurrence of a symbol. The average uncertainty is zero if one of the two symbols is impossible (the other one is certain). It is clear that such source will always produce the same known symbol.

Let us now consider a DMS with *M* symbols. Intuitively, we can state that the source with such alphabet that produces the highest amount of information is the one for which all symbols are equiprobable. It is the case for *M* = 2. For the general case, we can solve the following optimization problem:

Find the set $\{p_1, p_2, \ldots, p_M\}$ that maximizes $H(X) = -\sum_{k=1}^{M} p_k \log p_k$ with the constraint $\sum_{k=1}^{M} p_k = 1$. This is a typical problem of constrained optimization. We use the Lagrange multiplier concept shown below:

Find x_i, $i = 1, \ldots M$ such that $f(x_1, \ldots, x_M)$ maximum along with $g(x_1, \ldots, x_M) = 0$. We build an augmented objective function $J(x_1, \ldots, x_M) = f(x_1, \ldots, x_M) + \lambda g(x_1, \ldots, x_M)$ and we optimize the augmented function. The value of the variable λ is obtained by replacing the solution inside the constraint. In our case, the function is:

$J = -\sum_{k=1}^{M} p_k \log p_k + \lambda\left[\left(\sum_{k=1}^{M} p_k\right) - 1\right]$ and $\frac{\partial J}{\partial p_k} = 0$ provides $\ln p_k = -(1 + \lambda \ln 2)$. So, the all the *M* numbers p_k are equal. This implies that $p_k = \frac{1}{M}$. The entropy of the alphabet is maximized when the symbols are equiprobable (we have the highest uncertainty in the choice of a symbol from the source).

Entropy of two alphabets

Consider the following two alphabets $X = \{x_1, x_2, \ldots, x_M\}$ with probabilities $p_1, p_2, \ldots, p_M$ and $Y = \{y_1, y_2, \ldots, y_N\}$ with probabilities $q_1, q_2, \ldots, q_N$. The two random variables are independent. We want to compute the average uncertainty of the source generating symbols consisting of pairs $x_i y_j$. The joint entropy is:

$$H(X,Y) = H(X) + H(Y) \tag{3}$$

Proof: start with $H(X,Y) = -\sum_{k=1}^{M}\sum_{j=1}^{N} \Pr\left[x_k y_j\right] \log \Pr\left[x_k y_j\right]$ and use the fact that $\Pr\left[x_k y_j\right] = p_k q_j$ for independent alphabets.

For the general case, the joint entropy is given by:

$$H(X,Y) = H(X) + H(Y \mid X) = H(Y) + H(X \mid Y) \tag{4}$$

The conditional entropies are given by:

$$H(X \mid Y) = -\sum_{i=1}^{M}\sum_{j=1}^{N} \Pr\left(x_i, y_j\right) \log\left(\Pr\left(x_i \mid y_j\right)\right) \tag{5}$$

$$H(Y \mid X) = -\sum_{i=1}^{M}\sum_{j=1}^{N} \Pr\left(x_i, y_j\right) \log\left(\Pr\left(y_j \mid x_i\right)\right) \tag{6}$$

5. Extension of a source

If we consider messages with *n* symbols produced by a source having an alphabet *X*, we can assimilate these messages to a source having an alphabet $X \times X \times \cdots \times X = X^n$. If the source is DMS, the successive symbols are independent and we can generalize the previous result (equation(3)):

$$H(X^n) = nH(X) \tag{7}$$

The result (7) is not valid if the source has memory. In this case, the uncertainty on the occurrence of a given symbol is lessened by our knowledge of the previous symbols that have already happened. For example, we can compute the entropy of the first symbol of the message:

$$H(X^1) = -\sum_{k=1}^{M} p_k \log p_k$$

If we consider now a length 2 message $x_{1i}x_{2j}$, $i, j = 1, \ldots, M$. The second symbol probability is now conditioned by the first one: $\Pr\left[x_{2j} \mid x_{1i}\right]$ and its information is $-\log \Pr\left[x_{2j} \mid x_{1i}\right]$. So, we can define the entropy of the second symbol given the first one as the following average:

$$H\left(X^2 \mid X^1\right) = -\sum_{i=1}^{M}\sum_{j=1}^{M} \Pr\left[x_{1i}, x_{2j}\right] \log \Pr\left[x_{2j} \mid x_{1i}\right]$$

Now, for messages with *n* symbols, the entropy of the n^{th} symbol given the previous $n-1$ is:

$$H\left(X^n \mid X^1X^2\ldots X^{n-1}\right) = -\sum_{i_1=1}^{M}\ldots\sum_{i_n}^{M}\Pr\left[x_{1i_1}x_{2u_2}\ldots x_{ni_n}\right]\log\Pr\left[x_{ni_n} \mid x_{1i_1}x_{2u_2}\ldots x_{n-1i_{n-1}}\right]$$

If the source is stationary, we can define the limit:

$$H_\infty(X) = \lim_{n\to\infty} H\left(X^n \mid X^1X^2\cdots X^{n-1}\right) \tag{8}$$

The entropy defined by (8) is the accepted definition of the entropy of a source having memory. In some references, we can also find the following definition[4]:

$$H'_\infty(X) = \lim_{n\to\infty}\left[\frac{1}{n}H\left(X^n\right)\right] \tag{9}$$

This is the average entropy of a symbol contained in a very long message.

When the source is stationary, the two entropies are equal. For sources that have memory, the following inequality is always satisfied:

$$H_\infty(X) \le H(X^1) \tag{10}$$

This means that the relationship that exists between symbols in a message reduces the information (If we can predict the next symbol in a message, we don't need to transmit it. For such sources, we can define the notion of "redundancy":

$$R = 1 - \frac{H_\infty(X)}{H(X^1)} \tag{11}$$

We can use the concept of entropy to predict result of some random experiment. Consider a random experiment described by the set of outcomes $X = \{x_1,\ldots,x_M\}$. If it is hard to perform the experiment, we can use other related experiments $A_1,\ldots,A_k$, each giving partial information on *X*. It is important to know how many of these experiments must be made in order to have complete knowledge of *X*. It is clear that this number must be the smallest integer *k* satisfying

$$H(X) \le H\left(A_1, A_2,\ldots, A_k\right) \tag{12}$$

Example[5]:

A sultan received 12 bags filled with gold coins representing taxes from the 12 provinces of kingdom. Each coin is supposed to weight 100 g. However, the sultan has been told that the coins from the bag sent by one of his emirs weighted 5 g less. The Sultan had at his disposal a scale that could only determine whether the weights of the two plates are equal or if one is

---

[4] Cover, T. M. and J. A. Thomas, *Elements of Information Theory*, 2nd Ed., John Wiley, 2006.
[5] Guiasu, S. and R. Theodorescu, *Incertitude et Information*, les presses de l'université Laval, Québec, 1971.

heavier than the other. If we take one coin out of each bag, what is the minimum number of weighting required to discover the dishonest emir? We can solve this problem by using information theory.

The experiment *X* of discovering the bad coin directly from the twelve coins is represented by a set with twelve outcomes. If we don't have any other information, choosing a coin at random has a probability $1/12$ of being correct (The twelve coins are equiprobable). So, we have $H(X) = \log 12$.

The operation *A* of weighting the coins has three possible outcomes: equilibrium, right coin heavier, left coin heavier. Here also, we must assume that the three outcomes are equiprobable. So, $H(A) = \log 3$. We must perform *k* time the operation *A*. Each repetition is independent on the preceding one, so $H(A, A, \ldots, A) = kH(A)$. Using the inequality(12), we obtain: $k \geq \frac{\log 12}{\log 3}$. The nearest integer value is: $k = 3$. So, we must perform at least three weightings in order to determine which bag contains the counterfeit coin. Of course, every weighting operation should eliminate the maximum of uncertainty. We start the operation by selecting one coin out of each bag. We divide this set of twelve coins into three groups: two of them are going to be compared and the third is what remains. In order to obtain the maximum of information, the probability to find the group that contains the counterfeit coin should be the same (maximum entropy). So, the three groups should contain the same number of coins. So, we divide the set of 12 coins into three sets containing each 4 coins. We select any two groups at random and we compare them using the scale. As a result of this experiment, we identify the group that contains the bad coin. If the scale remains in equilibrium, it is the remaining group that contains the coin, otherwise, the bad coin belongs to the lighter group identified by the scale. Once the group identified, we are also going to divide it into three subgroups. However, 4 cannot be divided by 3. So, the group containing the bad coin is going to be divided into three groups containing respectively 1, 1 and 2 coins. The second experiment (weighting) will be done between the two single coin groups. If the counterfeit coin is one of the two, it will be identified at that time. Otherwise, we will have to wait until the next experiment.

6. Coding a source (binary alphabet)

When we want to transmit messages generated by a source, we usually have to encode the different sequences of symbols generated to symbols taken from another alphabet (more suitable for transmission). In this part of the course, we are going to see only the binary case.

Equal length coding

In this part, we are going to assign the same number of binary digits (say n) to a given number of source symbols. Let us start with an example. Assume we are given a DMS with an alphabet $X = \{x_1, x_2, x_3, x_4, x_5, x_6\}$. If we encode single symbols using binary digits, we need at least 3 binary digits since $\log[\text{Card}(X)] = 2.58$. However, $2^3 = 8$, so, there will be two binary combinations that will not be used. We could have the following assignment:

x_1 000
x_2 001
x_3 010
x_4 011
x_5 100
x_6 101

We see that the 110 and 111 will not be used. We can represent the encoding process by a binary tree: The coding tree. An upward going branch is labeled "0" and a downward going one is labeled "1". The source symbols are placed at nodes and the code represents the label of the path from the root of the tree to the node representing the symbol.

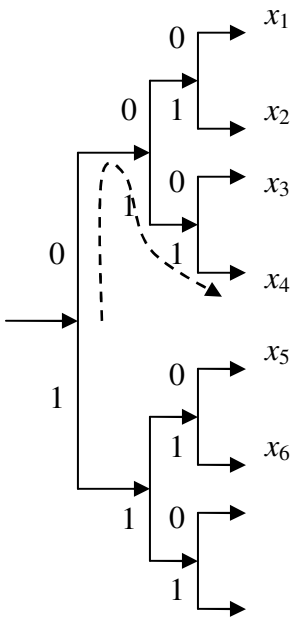


Figure 2 Coding tree

Figure 2 represents the encoding of the above source. The path shown with dotted line leading from the root to the symbol x_4 is labeled 0 1 1, which the code assigned to that symbol. The last terminal nodes are not assigned. We can have more efficient coding by using extensions of the source. If we use the second extension, we obtain an alphabet with $6^2 = 36$ symbols. In this case, $\log_2(36) = 5.1699$. It means that we should use 6 binary digits for each possible pair of symbols. In this case, there is no gain. However, if we go to the third extension, we obtain an alphabet with $6^3 = 216$ symbols. We have $\log_2(216) = 7.7548875$. So, 8 binary digits are enough to encode a sequence of three symbols. So, without using the probability distribution of the alphabet, we have obtained a gain since we use $\frac{8}{3} = 2.66$ binary digit per symbol. We can show that at the limit (infinite extension), the number of binary digit needed to encode the source X is $\log_2[\text{Card}(X)]$.

Variable length coding

We can have more efficient coding if we use variable length codes. One such code can be obtained from the previous source and the coding tree shown in Figure 2.

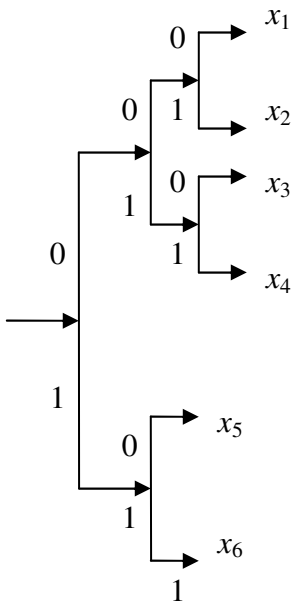


Figure 3 Variable length coding tree

The tree shown in Figure 3 has different path length leading to the source symbols. The assigned binary codes are now:

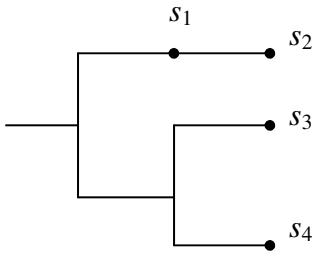
x_1 000
x_2 001
x_3 010
x_4 011
x_5 10
x_6 11

We can see that two codewords have a length of two while the other four have a length of three. It means that the average length for this code is shorter. The decoding of a given sequence of symbols is straightforward. We just have to follow the encoding tree all the way to a node labelled by a symbol. For example, consider the sequence $x_1x_5x_4$, its code is: 00010011. Starting from the root of the tree represented by Figure 3, the sequence 000 leads us directly to the node labelled x_1. We restart from the root, now the sequence 10 takes us to x_5, etc.

When we use variable length coding there are some properties that a code must satisfy. It must first satisfy the **unique decoding property**. Consider a source with four symbols:

s_1 0
s_2 10
s_3 00
s_4 11

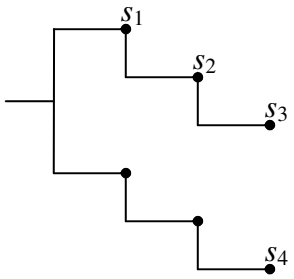
The sequence $s_1s_3s_4$ is encoded as: 00011. However, this sequence corresponds also to $s_1s_1s_1s_4$. That means that it is impossible to decode the transmitted sequence.



The above coding tree shows that the code assigned to s_1 is the beginning (prefix) of the one assigned to s_2. Another desirable property is that the code should be **instantaneous**. It means that the symbol is decoded as soon as its corresponding code is finished being scanned. The code described by the coding tree of Figure 3 is instantaneous. The following code is not instantaneous but it is uniquely decodable.

s_1 0
s_2 01
s_3 011
s_4 111

Consider the binary sequence 0111111. We have to wait until we read the last binary digit before we start decoding. The above sequence corresponds to the symbols $s_1s_4s_4$. If we add one more 1 at the end, the binary sequence becomes 01111111. It corresponds to $s_2s_4s_4$.



Its coding tree is shown above. We see that here also we have codewords that are the prefix of other codewords. The symbols are assigned to intermediate nodes of the coding tree. A code is instantaneous if the symbols are assigned to terminal nodes (leaves). It means that no codeword is a prefix of another codeword. It is highly desirable for a code to be instantaneous. One necessary and sufficient condition for the existence of an instantaneous code having code lengths (number of binary digits) l_i assigned to symbols s_i, i = 1, …, M is provided by Kraft's inequality.

Kraft's inequality:

$$\sum_{i=1}^{M} 2^{-l_i} \le 1 \tag{13}$$

Before we proceed, we have to define the average length of a code:

Average length of a code:

Given a source with alphabet $X = \{x_1, x_2, \ldots, x_M\}$ with probabilities $\{p_1, p_2, \ldots, p_M\}$, if each x_i symbol is encoded with l_i binary digits, i = 1, 2, …, M, the average length of the source is:

$$\bar{l} = \sum_{i=1}^{M} p_i l_i \tag{14}$$

Source coding theorem

Given a source with alphabet $X = \{x_1, x_2, \ldots, x_M\}$ with probabilities $\{p_1, p_2, \ldots, p_M\}$, there exists an instantaneous code with an average length $\bar{l}$ such that:

$$H(X) \le \bar{l} \le H(X)+1 \tag{15}$$

Proof: let us choose a code with codeword lengths such that $I[x_i] \le l_i \le I[x_i]+1$ for $i=1,\ldots,M$. We multiply each one of the inequalities by p_i. We obtain: $p_i I[x_i] \le p_i l_i \le p_i I[x_i] + p_i$ for $i=1,\ldots,M$. Adding the M inequalities, we obtain the inequalities (15). Now, because $I[x_i] \le l_i$, we obtain $p_i \ge 2^{-l_i}$ and here again we add the M inequalities. The result is:

$1 \ge \sum_{i=1}^{M} 2^{-l_i}$ This is nothing but Kraft's inequality. It means that there exists an instantaneous code with code lengths as given above.

(Q.e.d.)

The above theorem is an existence theorem. It does not give us a way to construct such code. There exist methods for constructing instantaneous codes with an average length that is as short as possible. The basic idea is to build a coding tree with source symbols as terminal nodes. Furthermore, the shortest codes are assigned to the most probable symbols and the longest codes to the least probable symbols.

In our course, we are going to see two methods for assigning binary codes to symbols.

The Shannon-Fano algorithm

The Shannon-Fano algorithm is based on first order the symbols by decreasing probability and then dividing the set of symbols into two almost equiprobable sets. We assign a zero to one set and a one to the other set. We repeat the process for each one of the obtained groups until we are left with a group containing only one symbol.

Example: Consider a DMS $X = \{x_1, x_2, x_3, x_4, x_5, x_6\}$ with probabilities $\{0.4, 0.3, 0.1, 0.1, 0.05, 0.05\}$. We generate the following table:

x_1	0.4	0			
x_2	0.3		0		
x_3	0.1				0
x_4	0.1	1		0	1
x_5	0.05		1		0
x_6	0.05			1	1

The code assignment is:

$$
\begin{array}{ll}
x_1 & 0 \\
x_2 & 10 \\
x_3 & 1100 \\
x_4 & 1101 \\
x_5 & 1110 \\
x_6 & 1111
\end{array}
$$

The average length is $\overline{l} = 2.2$ binary digits/symbol while the entropy is H(X) = 2.1464 bit/symbols. The above assignment is not unique. We can also derive the following assignment:

$$
\begin{array}{ll}
x_1 & 0 \\
x_2 & 10 \\
x_3 & 110 \\
x_4 & 11110 \\
x_5 & 111110 \\
x_6 & 111111
\end{array}
$$

This coding produces the same average length.

The Huffman procedure

The Huffman procedure is an optimum procedure. It produces the shortest possible code (on average). The procedure is very simple to implement. It consists of the repetition of the following two steps:

1. Order the symbols by decreasing value of probability.
2. Group the last two symbols to obtain a new symbol with a probability equal to the sum of probabilities. We obtain a new alphabet with $M - 1$symbols.
3. We repeat 1and 2 until we are left with only one symbol.

The grouping of symbols generates a coding tree. We assign a zero for an up going branch and a one for a down going branch. We simply read the code by following the path from the root to the symbol.

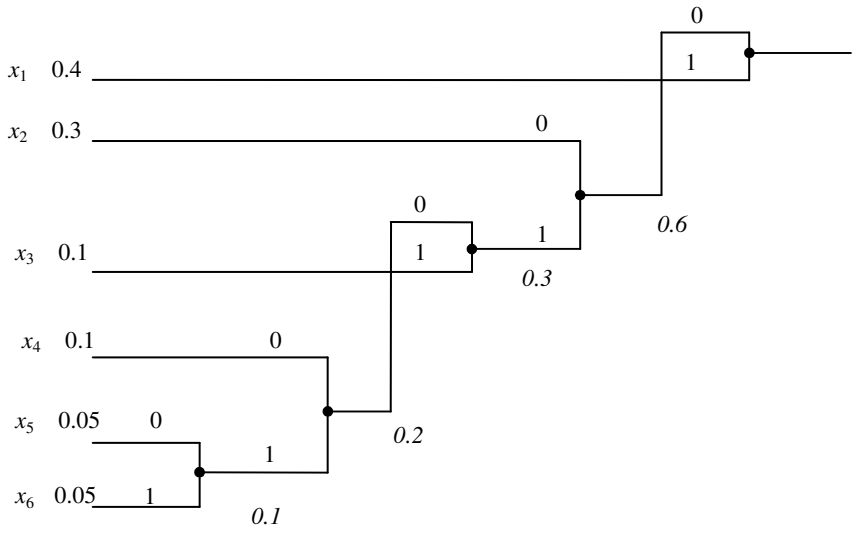


Figure 4 Huffman coding tree

We can see the succession of operations in Figure 4. The sum of probabilities is written in italic. The codewords are read by following the path from the root to the symbols. The code is shown below:

x_1	1
x_2	00
x_3	011
x_4	0100
x_5	01010
x_6	01011

We remark that the length assignment is the same as the second one obtained by the Shannon-Fano algorithm. The average length is the same: $\bar{l} = 2.2$ binary digits/symbol.

We define le code efficiency as being the ratio:

$$\eta = \frac{H(X)}{\bar{l}} \tag{16}$$

As we did with fixed length coding, we can have shorter codewords if we encode extensions of the source. Consider the following source: $X = \{x_1, x_2\}$ with probabilities $\{0.9, 0.1\}$. Its entropy is $H(X) = 0.46899$ bit/symbol. If we use the first extension, the code length is $\bar{l} = 1$ binary digit/symbol. The efficiency is about 47%. For the second extension, the average length for a pair of symbols (using Huffman procedure) is 1.29 giving an average length $\bar{l} = 0.645$ binary digit/symbol. The efficiency becomes about 72%. If we go to the third extension the average length of a codeword for a triplet of symbols is 1.674, giving $\bar{l} = 0.558$ binary digit/symbol. The efficiency becomes about 84%. We see that the

efficiency increases as we encode extensions. In fact, if we apply the source coding theorem for an extension of a DMS, we obtain:

$H(X^n) \le \bar{l}_n \le H(X^n)+1$, where $\bar{l}_n = n\bar{l}$ is the average length of the code of the nth extension. We have seen also that $H(X^n) = nH(X)$, so for an nth extension, the coding theorem gives $H(X) \le \bar{l} \le H(X)+\frac{1}{n}$. The shortest average length is the entropy of the source. We can show that the above result is valid even if the source has memory.

7. Transmission of information and channels

Up to now, we have characterized discrete sources of information. In this part of the course, we are going to study relationships between two different alphabets. Consider the following diagram:

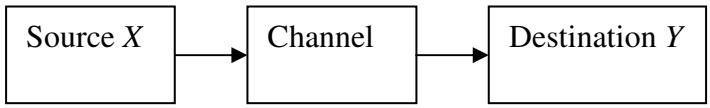
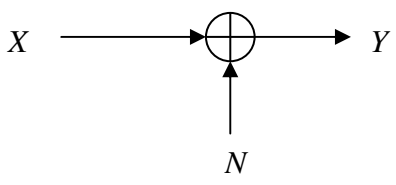


Figure 5 Communication system

The source produces messages taken from the alphabet *X*. The destination observes messages that are sequences from the alphabet *Y*. So, the "channel" is a mapping between sequences of symbols from *X* to sequences from alphabet *Y*. Depending on the source and destination alphabets, we can have different types of channels. If the source and destination alphabets are discrete, we are dealing with a discrete channel. If the input and destination are continuous random variables (processes), the channel is continuous. We can also have mixed type of channels.

Example:

Additive white Gaussian channel:



In the above figure, X is a bandlimited stochastic process, N is a bandlimited white Gaussian noise and Y is the output of the channel.

The discrete memoryless channel

In this part of the course, our concern will be on discrete channels. So, the source has a finite alphabet X of size M and the output of the channel is a finite alphabet Y of size N. We can characterize a discrete channel by conditional probabilities. We transmit a message $x_{i1}x_{i2}\ldots x_{ik}$ and we receive $y_{i1}y_{i2}\ldots y_{ik}$. The messages are of size k. We say that the channel is memoryless if $\Pr\left[y_{i1}y_{i2}\ldots y_{ik} \mid x_{i1}x_{i2}\ldots x_{ik} \right] = \prod_{j=1}^{k} \Pr\left[y_{ij} \mid x_{ij} \right]$. It means that the joint probabilities do not depend on the order of the symbols. We are going to characterize essentially discrete memoryless channels (DMC).

DMC's are characterized by a set of conditional probabilities $p_{ij} = \Pr\left[y_j \mid x_i \right]$ which are the probabilities of observing the symbol y_j at the output given that the symbol x_i is presented at the input. These conditional probabilities are grouped inside a matrix P[Y|X] having M rows and N columns and they are called "transition" probabilities.

$$\mathbf{P}\left[Y \mid X \right] = \begin{pmatrix} p_{11} & p_{12} & \cdots & p_{1N} \\ p_{21} & p_{22} & \cdots & p_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ p_{M1} & p_{M2} & \cdots & p_{MN} \end{pmatrix} \tag{17}$$

The above matrix belongs to the class of "positive" matrices (Its elements are all positive). The row sum is equal to one:

$$\sum_{j=1}^{N} p_{ij} = 1 \tag{18}$$

With the above property, the matrix is called a "stochastic" matrix. We can use it to calculate the different probabilities in the communication system. Let us define the following row matrices:

$$\mathbf{P}\left[X \right] = \begin{pmatrix} \Pr(x_1) & \Pr(x_2) & \cdots & \Pr(x_M) \end{pmatrix}$$

$$\mathbf{P}\left[Y \right] = \begin{pmatrix} \Pr(y_1) & \Pr(y_2) & \cdots & \Pr(y_N) \end{pmatrix}$$

We have the following relation between probabilities:

$$\mathbf{P}\left[Y \right] = \mathbf{P}\left[X \right] \mathbf{P}\left[Y \mid X \right] \tag{19}$$

We can also define the following diagonal matrix:

$$\mathbf{P}[X]_{diag} = \begin{pmatrix} \Pr(x_1) & 0 & \cdots & 0 \\ 0 & \Pr(x_2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Pr(x_M) \end{pmatrix}$$

This matrix allows us to compute the different joint probabilities:

$$\mathbf{P}[X,Y] = \begin{pmatrix} \Pr(x_1, y_1) & \Pr(x_1, y_2) & \cdots & \Pr(x_1, y_N) \\ \Pr(x_2, y_1) & \Pr(x_2, y_2) & \cdots & \Pr(x_2, y_N) \\ \vdots & \vdots & \ddots & \vdots \\ \Pr(x_M, y_1) & \Pr(x_M, y_2) & \cdots & \Pr(x_M, y_N) \end{pmatrix}$$

The relation is:

$$\mathbf{P}[X,Y] = \mathbf{P}[X]_{diag}\,\mathbf{P}[Y \mid X] \tag{20}$$

There exists also a graphical representation of a DMC. It is a directed flow graph with input nodes on the left representing the M input symbols and output nodes on the right representing the N output symbols. These nodes are connected through directed branches labelled by the transition probabilities. If a transition probability is zero, its corresponding branch is not represented.

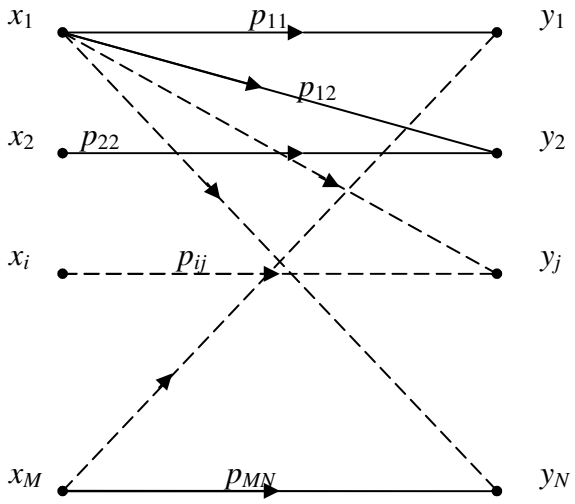


Figure 6 Graph of a DMC

Some special channels

Lossless channel:

The lossless channel is characterized by a channel transition matrix having only one non-zero element in each column. So, each output symbol is connected to only one input symbol. It means that no information is lost in this type of channel.

Deterministic channel:

The deterministic channel is characterized by a channel transition matrix having only one non-zero element in each row. Because the matrix is stochastic, the element of each row must be equal to one. So, when a symbol is presented at the input of the channel, we know exactly which output symbol will appear.

Noiseless channel:

The noiseless channel is both lossless and deterministic. So, a noiseless channel must have the same number of input and output symbols, $M = N$. So, its matrix is an $M \times M$ identity matrix.

Useless channel:

The useless channel is characterized by output symbols that are independent on input ones. So, $p_{ij} = \Pr\left(y_j \mid x_i\right) = \Pr\left(y_j\right)$. The channel matrix has M rows that are identical. Each row contains the N probabilities of the output symbols.

Symmetric channel:

A channel is symmetric if the all the rows of the transition matrix are permutation of each other and all the columns are also permutation of each other. In general, these channels have the same number of inputs and outputs.

Weakly symmetrical channel:

A weakly symmetrical channel has all rows of the transition matrix that are permutation of each other and also the column sums are equal: $\sum_{i=1}^{M} p_{ij} = K = \text{cste}$.

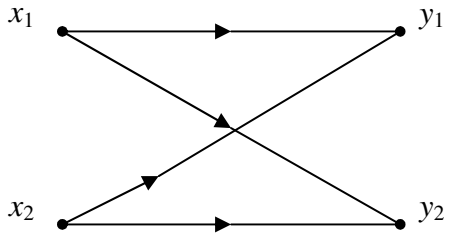
One property of both symmetric and weakly symmetric channels is that if the input symbols are equiprobable, the output symbols are also equiprobable. This is a consequence of equation (19).

$\Pr(y_j) = \sum_{i=1}^{M} \Pr(x_i)\, p_{ij}$ so, if $\Pr(x_i) = \frac{1}{M}$ then $\Pr(y_j) = \frac{1}{M} \sum_{i=1}^{M} p_{ij} = \frac{K}{M} = \frac{1}{N}$. We can see that a weakly symmetric channel must have a column sum satisfying $K = \frac{M}{N}$. This implies also that a symmetric channel transition matrix must have a column sum equal to one. This is a doubly stochastic matrix.

Probability of error of some channels

In this part, we are going to consider channels that have the same number of input and output symbols ($M = N$). In the graph representing the channel, we consider as correct transmission a horizontal line ($i = j$) and an error event is represented by an oblique line ($i \neq j$).

Example: Binary channel:



The transition probability matrix is

$$\begin{pmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{pmatrix}$$

The probabilities of the input are $\Pr(x_1) = \alpha$ and $\Pr(x_2) = 1 - \alpha$. The error events are: We transmit x_1 and we receive y_2 or we transmit x_2 and we receive y_1. So, the probability of error is:

$$\begin{aligned} \Pr[e] &= \Pr(x_1, y_2) + \Pr(x_2, y_1) = \Pr(x_1)\Pr(y_2 \mid x_1) + \Pr(x_2)\Pr(y_1 \mid x_2) \\ &= \alpha p_{12} + (1-\alpha) p_{21} \end{aligned}$$

If the channel is symmetric (Binary Symmetric Channel: BSC), we can write $p_{12} = p_{21} = p$. At that time, $\Pr[e] = p$.

It is easy to show that $\Pr[e] = 0$ for the noiseless channel.

For the general case, the probability of error is:

$$\Pr[e] = \sum_{i=1}^{M} \sum_{\substack{j=1 \\ j \neq i}}^{M} \Pr(x_i, y_j) = \sum_{i=1}^{M} \Pr(x_i) \sum_{\substack{j=1 \\ j \neq i}}^{M} p_{ij}$$

We can also calculate it as:

$$\Pr[e] = 1 - \Pr[c] = 1 - \sum_{i=1}^{M} \Pr(x_i)\, p_{ii}$$

Joint Entropies

Since two alphabets that are related by the channel transition matrix P[*Y*|*X*], we can define many entropies.

Source Entropy:

The source alphabet $X = \{x_1, x_2, \ldots, x_M\}$ is characterized by its entropy:

$$H(X) = -\sum_{i=1}^{M} \Pr(x_i) \log \Pr(x_i)$$

Destination Entropy:

The destination alphabet $Y = \{y_1, y_2, \ldots, y_N\}$ is characterized by its entropy:

$$H(Y) = -\sum_{j=1}^{N} \Pr(y_j) \log \Pr(y_j)$$

Joint Entropy:

$$H(X,Y) = -\sum_{i=1}^{M} \sum_{j=1}^{N} \Pr[x_i y_j] \log \Pr[x_i y_j]$$

This is the average uncertainty on pairs of symbols, one from the source, the other one from the destination.

Equivocation:

The equivocation is the conditional entropy of the input given that the output of the channel has been observed.

$$H(X \mid Y) = -\sum_{i=1}^{M} \sum_{j=1}^{N} \Pr(x_i, y_j) \log\left(\Pr(x_i \mid y_j)\right)$$

The uncertainty that existed before observing the exit from the information channel is represented by $H(X)$. The average uncertainty that remains after observing the exit from the information channel is the equivocation $H(X|Y)$. So, it represents the average information lost in the channel.

Conditional entropy of *Y* given *X*:

Finally, the last entropy that we use to study communication channels is the average uncertainty of the output given our knowledge of the input.

$$H(Y \mid X) = -\sum_{i=1}^{M} \sum_{j=1}^{N} \Pr(x_i, y_j) \log\left(\Pr(y_j \mid x_i)\right)$$

These entropies have already been defined previously by equations(4), (5) and(6).

Conditional entropy of some channels:

Lossless channel: It has $H(X|Y) = 0$. This is due to the fact that if y_j is known, we know exactly the input x_i that is connected to that output. So, $\Pr[x_i|y_j] = 1$.

Deterministic channel: It has $H(Y|X) = 0$. For this channel, knowledge of the input implies knowledge of the output: $\Pr[y_j|x_i] = p_{ij} = 1$.

Noiseless channel: $H(X|Y) = H(Y|X) = 0$. It is both lossless and deterministic.

Useless channel: Because of independence $H(X|Y) = H(X)$ and $H(Y|X) = H(Y)$.

Mutual Information

We have seen that the equivocation $H(X|Y)$ is a measure of the average information lost in the channel. The average information presented at the input of the channel is measured by $H(X)$. So, the average amount of information that arrives at the output of the channel (information flow) is measured by:

$$I(X;Y) = H(X) - H(X \mid Y) \tag{21}$$

The above quantity is called the "mutual information".

We can express the mutual information using the elementary probabilities:

$$I(X;Y) = \sum_{i=1}^{M}\sum_{j=1}^{N} \Pr(x_i, y_j) \log \frac{\Pr(x_i, y_j)}{\Pr(x_i)\Pr(y_j)} \tag{22}$$

The above relation shows that the mutual information is symmetrical with respect to X and Y.

Properties of $I(X;Y)$:

$$I(X;Y) = I(Y;X) \tag{23}$$

$$I(X;Y) \geq 0 \tag{24}$$

$$I(X;Y) = H(X) - H(X \mid Y) = H(Y) - H(Y \mid X) \tag{25}$$

$$I(X;Y) = H(X) + H(Y) - H(X,Y) \tag{26}$$

Example:

Consider the BSC with transition matrix:

$$\mathbf{P}[Y \mid X] = \begin{pmatrix} 0.9 & 0.1 \\ 0.1 & 0.9 \end{pmatrix}$$

and source probabilities: $\mathbf{P}[X] = (0.5 \quad 0.5)$. Using equation(19), we obtain $\mathbf{P}[Y] = (0.5 \quad 0.5)$ and using equation(20), we can compute the four joint probabilities:

$$\mathbf{P}[X,Y] = \begin{pmatrix} \Pr(x_1, y_1) & \Pr(x_1, y_2) \\ \Pr(x_2, y_1) & \Pr(x_2, y_2) \end{pmatrix} = \begin{pmatrix} 0.45 & 0.05 \\ 0.05 & 0.45 \end{pmatrix}$$

We use now equation(26) to compute the mutual information and we find:

$$I(X;Y) = H(X) + H(Y) - H(X,Y) = 1 + 1 - 1.47 = 0.53 \text{ bit/symbol}$$

We remark that practically half of the information is lost in the channel. However, the probability of error is 0.1. It means that in a long message, nine binary digits out of ten are correct. The problem is that it is impossible for us to know which ones are correct! If the probability of error is 0.5, meaning that one binary digit out of two is correct in a long message, the mutual information will be $I(X;Y) = 0$. There is no flow of information in the channel.

Channel Capacity

The capacity of a channel is a quantity that characterizes the maximum flow of information that can be handled by a channel. So, it is defined as

$$C = \max_{all \Pr(x_i)} I(X;Y) \tag{27}$$

The maximization is taken over all input probability distributions.

Capacity of some channels:

Lossless channel:

We have seen that $H(X|Y) = 0$ for this type of channel. So, $I(X;Y) = H(X)$. It implies that $C = \log M$. (The maximum of $H(X)$ is achieved for a uniform distribution)

Deterministic channel:

For this channel, $H(Y|X) = 0$. So, $I(X;Y) = \mathrm{H}(Y)$. It is always possible to find a distribution of the input that makes the symbols of Y equiprobable. For example, if k_1 symbols of the input generate y_1, k_2 symbols of the input generate y_2, …, then each input of the first group will have a probability equal to $1/k_1 N$, each input of the second group will have a probability equal to $1/k_2 N$, etc. so the capacity of the channel is $C = \log N$.

Noiseless channel:

From the previous results, the capacity of the channel is: $C = \log M$.

Useless channel:

Because of the independence between the two vocabularies, we have $H(X|Y) = H(X)$. So, $I(X;Y) = 0$ and then $C = 0$.

Symmetric and weakly symmetric channel:

For this type of channel, the capacity is: $C = \log N - H(\mathbf{row})$, where $H(\mathbf{row})$ is the entropy computed using the probabilities of one row of the transition matrix.

Proof:

$$I(X;Y) = H(Y) - H(Y \mid X)$$

$$H(Y \mid X) = -\sum_{i=1}^{M}\sum_{j=1}^{N} \Pr(x_i, y_j) \log p_{ij}$$

$$= -\sum_{i=1}^{M} \Pr(x_i) \sum_{j=1}^{N} p_{ij} \log p_{ij}$$

Since all the rows of the transition matrix are permutations of the first row. So, the inner summation is a constant $H(\mathbf{row})$, so:

$$H(Y \mid X) = H(\mathbf{row})$$

So, $I(X;Y) = H(Y) - H(\mathbf{row})$ and we have already seen that a uniform distribution of the input implies a uniform distribution of the output, then:

$$C = \log N - H(\mathbf{row})$$

(Q.e.d.)

Binary Symmetric Channel:

The BSC is a symmetric channel. The above rule applies. The transition matrix is:

$$\begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}$$

The capacity is $C = 1 + p \log p + (1-p)\log(1-p)$.

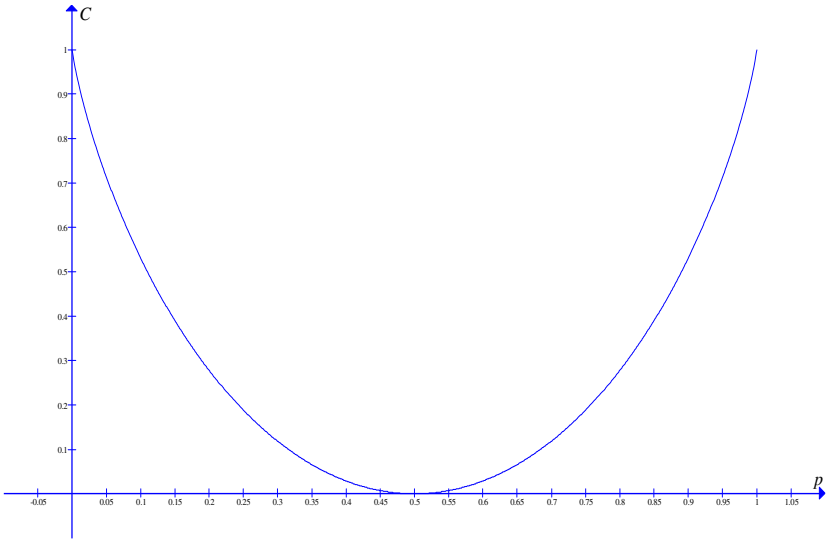


Figure 7 Capacity of a BSC

We see that for the BSC, the capacity is around 1 bit/symbol when the probability of error is small and that it becomes useless when $p = 0.5$. The capacity is symmetrical with respect to $p = 0.5$. If p > 0.5, it means simply that we should invert our decision at the output of the

channel. A typical BSC is a binary communication system using antipodal signaling. The channel probability of error is:

$$p = \frac{1}{2}\operatorname{erfc}\sqrt{\frac{E_b}{N_0}}$$

The calculation of the capacity for the previous types of channel was quite easy. However, it is not always the case. In general, the computation of the capacity of a channel is quite complex and it is achieved by some optimization technique.

The Channel Coding Theorem

The channel capacity is a quantity that characterizes the ability of a channel to transfer correctly information. Although there is a finite probability of error for each symbol transmitted, the following theorem shows that we can correctly transmit long messages. We have already seen this behavior in orthogonal signaling. As the size of the message increases, the probability of message error decreases.

Before we can state the channel coding theorem, we have to define the notion of "channel coding". A code is a mapping from a sequence of symbols of length *k* to another sequence of symbols of length *n*. Usually the two sequences are taken from a binary alphabet. The code rate is the ratio $R_c = k/n$.

Theorem:

Given a source of information with entropy $H(X)$ and a channel with capacity C, there exist a code such that the probability of error when transmitting messages over the channel is arbitrarily small if $H(X) \leq C$. If $H(X) > C$, It is impossible to transmit information reliably over the channel.

The proof of the theorem is quite involved and cannot be done at the level of our course. However, we can have a short demonstration for the BSC. The following part is just provided for reading and can be skipped. However, it is quite instructive. It is taken from Carlson[6].

Coding for a BSC:

Let us introduce the binary entropy function: $h(x) = -x\log x - (1-x)\log(1-x)$. The capacity of a BSC with probability of error p ($< \frac{1}{2}$) is $C = 1 - h(p)$ in bits/symbol. If the rate

---

[6] Carlson, A., B., Crilly, B.,P. and Rutledge, J., C., *Communication Systems* : *an Introduction to Signal and Noise in Electrical Communication*, 4th ed., Mc Graw Hill, 2002.

of binary symbols through the channel is r, the capacity of the channel can be expressed as $C_t = rC$ in bits/s. Consider the following diagram:

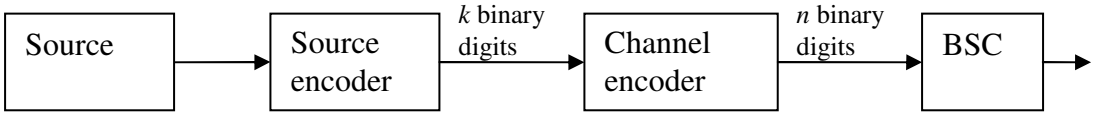


Figure 8 Binary Communication System

The source encoder is assumed to be optimal with an efficiency of 100%. So, the binary digits at its output are assumed to be equiprobable. The channel encoder maps every sequence of k binary digits to sequences of n binary digits. The code rate is $R_c = \frac{k}{n}$. Since the outputs of the source encoder are assumed equiprobable, the information for a message of size k is k bit/message. So, the information of every binary digit entering the channel is $\frac{k}{n} = R_c$. This information should be smaller than the capacity of the channel.

$$\frac{k}{n} \le C$$

$$k = n(C - \varepsilon) \qquad 0 \le \varepsilon \le C$$

Every k sequence at the input of the channel encoder corresponds to one out of $M = 2^k = 2^{n(C-\varepsilon)}$ messages.

At the output of the encoder, we have sequences of n binary digits. It corresponds to an n dimensional vector space. The number of valid codewords is only $M = 2^k$. In this space, we define a distance called the "Hamming distance". The distance between two n dimensional vectors is the number of positions in which they differ. The channel encoder puts one (out of M) vector V at the input of the BSC and we receive a different vector V' at the output of the BSC.

At the receiver, we decide that the observed vector is correct if it falls at a distance smaller than a threshold d. Otherwise, we declare that we have an error. The code is selected at random. It can be a good code (see the coding theory part of the course) or it can assign codewords that are very close to each other. So, we have two possibilities for error. Either the noise makes the Hamming distance $d(V,V') = l$ larger than d (We call this a noise error), or another codeword is assigned at a distance less than d. So, we can say that the probability of error is the sum of two probabilities:

$$\Pr[error] = \Pr[noise\ error] + \Pr[code\ error]$$

The probability to have l noise induced errors in a message of n symbols is clearly a binomial with probability p for each symbol. So, $\Pr[noise\ error] = \Pr[l > d]$. We can use the Chebyshev bound for this. The mean and the variance of the variable l are:

$$E[l] = np \qquad\qquad \sigma_l^2 = np(1-p)$$

So: $\Pr[noise\ error] = \Pr[l > d] \le \left(\frac{\sigma_l}{d - E[l]} \right)^2$

Let use select $d = n\alpha$ with $p < \alpha < \frac{1}{2}$. Then:

$$\Pr[noise\ error] \le \frac{p(1-p)}{n(\alpha - p)^2}$$

The above quantity decreases with increasing n.

To determine and bound the probability of a code error, we should determine, when we select a code at random, what is the probability to have a valid codeword at a Hamming distance less than d. Since we deal with sequences of n binary digits, there exist 2^n possible vectors. We assume that there are m valid codeword inside the "Hamming sphere" of radius d [7] centered on the transmitted vector V. Once we have selected V, there remain $M-1$ possible valid codewords. The probability to select one inside the sphere is then $m/2^n = m2^{-n}$. So, the probability of a code error is:

$$\Pr[code\ error] = (M-1)m2^{-n} < mM\,2^{-n} = m2^{n(C-\varepsilon)}2^{-n}$$

Replacing $C = 1 - h(p)$, we obtain: $\Pr[code\ error] < m2^{-n(h(p)-\varepsilon)}$. The number of vectors inside the sphere is:

$$m = \binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{d} = \sum_{i=0}^{d} \binom{n}{i}$$

The largest term in the above sum of $d+1$ terms is the last one because $d = n\alpha < n/2$. So:

$$m \le (d+1)\binom{n}{d} = (d+1)\frac{n!}{d!(n-d)!}$$

We use Stirling approximation for the factorials of the large numbers n, $(n-d)$ and d.

---

[7] A Hamming sphere of radius d is the set of points situated at a Hamming distance ≤ d from its center.

$$k! \approx \sqrt{2\pi k}\, k^k e^{-k} \qquad k \gg 1$$

After simplification, we finally obtain:

$$m \le \frac{n\alpha + 1}{\sqrt{2\pi n\alpha(1-\alpha)}} 2^{nh(\alpha)}$$

Replacing in the probability of a code error, we obtain:

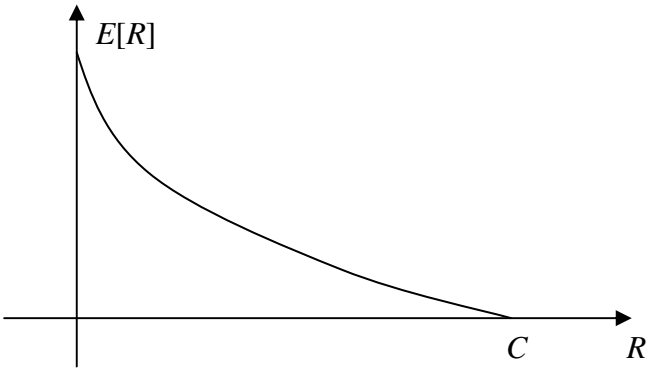
$$\Pr[code\ error] < \frac{n\alpha + 1}{\sqrt{2\pi n\alpha(1-\alpha)}} 2^{-n(\varepsilon + h(p) - h(\alpha))}$$

So, as $n \to \infty$, as long as $\varepsilon > h(\alpha) - h(p)$, the above probability will converge to zero. We have selected $p < \alpha < 1/2$, so ε can be a small positive number. We have seen that the probability of error (due to noise or bad choice of code) will decrease with large *n*. So, this proves that as long as the data rate can be handled by the channel, the probability of error will decrease.

A more general result for the BSC is the following bound [8]:

$$\Pr[error] \le e^{-nE[R]} \tag{28}$$

where $R = R_c H(X)$ and $E[.]$ is a convex up monotone decreasing positive function of its argument.



.

Figure 9 *E*[*R*] vs *R*

[8] Gallager, R. G., *Information Theory and Reliable Communication*, New York, Wiley,1968.

8. Continuous information sources

We have seen that information sources produce stochastic processes. We have studied the discrete case. Let us now consider the continuous case.

Instead of considering a discrete alphabet, we consider a continuous random variable X described by a pdf $f_X(x)$. If we apply the previous definition of entropy for this random variable, we have to go to a limit operation. The probability for the random variable to be in an interval of width Δx around a value x is $f_X(x)\Delta x$. We assume that the random variable has a range space extending from a to b, we will have: (we divide the interval into small intervals of width Δx)

$$H(X) = -\lim_{\Delta x \to 0} \sum f_X(x)\Delta x \log\left(f_X(x)\Delta x\right) = -\lim_{\Delta x \to 0}\left[\sum f_X(x)\Delta x \log\left(\Delta x\right) + \sum f_X(x)\Delta x \log\left(f_X(x)\right)\right]$$ [9]

The first term in the above summation is infinite. This result is quite logical since we need an infinite number of digits to represent an irrational number! So, we cannot simply extend the previous definitions to the continuous case. For a continuous random variable, we are going to define "relative" entropy which is just the limit of the second term:

$$H(X) = -\int_{-\infty}^{+\infty} f_X(x) \log f_X(x) dx \tag{29}$$

Using the above definition, we can have a negative value for the entropy. This entropy will be used to compare between random variables.

Example:

Consider a uniform random variable X.

$$f_X(x) = \begin{cases} 2 & 0 \le x \le \dfrac{1}{2} \\ 0 & elsewhere \end{cases}$$

Equation (29) provides $H(X) = -1$.

We have seen that in the discrete case, a uniform distribution maximizes the entropy of a source. In the continuous case, we can also use optimization theory in order to solve the same problem.

Theorem 1:

The random variable X with finite variance σ^2 that has the largest relative entropy is the Gaussian random variable. Its relative entropy is $H(X) = \dfrac{1}{2}\log\left(2\pi e\sigma^2\right)$.

---

[9] The logarithm is always taken base two unless otherwise stated.

Proof:

The problem that we have to solve is a typical problem of calculus of variation with isoperimetric constraints.

In the calculus of variation, we want to find the function *y* that maximizes (or minimizes) the functional:

$G(y) = \int_a^b L(y, y', x)\,dx$ with constraints $\int_a^b g_1(y, y', x)\,dx = K_1$ and $\int_a^b g_2(y, y', x)\,dx = K_2$.

y is a function of the variable *x* and *y'* is its derivative with respect to *x*. In our case, the functional to be maximized is the relative entropy:

$H(X) = -\int_{-\infty}^{+\infty} f_X(x) \log f_X(x)dx$ along with: $\int_{-\infty}^{+\infty} f_X(x)dx = 1$ and $\int_{-\infty}^{+\infty} (x-m)^2 f_X(x)dx = \sigma^2$.

To solve this problem, we use the augmented function $F = L + \lambda_1 g_1 + \lambda_2 g_2$ and solve the Euler-Lagrange equation:

$$\frac{d}{dx}\left[\frac{\partial F}{\partial y'}\right] - \frac{\partial F}{\partial y} = 0 \tag{30}$$

In our problem, we have:

$y = f_X(x)$, $L(y, y', x) = -y \ln y$ [10], $g_1(y, y', x) = y$ and $g_2(y, y'; x) = (x-m)y$

Since *y'* does not appear explicitly, $\frac{\partial F}{\partial y'} = 0$. Equation (30) becomes:

$-\ln y - 1 + \lambda_1 + \lambda_2 (x-m)^2 = 0$ giving $y = \exp(\lambda_1 - 1)\exp\left(\lambda_2 (x-m)^2\right)$.

Substituting the above into the constraints provides:

$$y = f_X(x) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left[-\frac{(x-m)^2}{2\sigma^2}\right]$$

(Q.e.d)

Replacing in the expression of the relative entropy gives:

$$H(X) = \frac{1}{2}\log\left(2\pi e\sigma^2\right)$$

Continuous Channel

A continuous channel is a mapping between two random variables *X* and *Y*. We know that two continuous random variables are described by several density functions:

---

[10] Since the neperian log and the log base 2 are proportional, it is easier to use the neperian one in the optimization process.

The joint density function $f_{XY}(x,y)$ and two conditional ones: $f_{X|Y}(x\,|\,y)$ and $f_{Y|X}(y\,|\,x)$.

We can then define several entropies:

$$H(X\,|\,Y) = -\iint f_{XY}(x,y)\log f_{X|Y}(x\,|\,y)\,dxdy \tag{31}$$

$$H(Y\,|\,X) = -\iint f_{XY}(x,y)\log f_{Y|X}(y\,|\,x)\,dxdy \tag{32}$$

$$H(X,Y) = -\iint f_{XY}(x,y)\log f_{XY}(x,y)\,dxdy \tag{33}$$

Using the above definition, we can define the "mutual information" between the input and the output of the continuous channel.

$$I(X;Y) = H(X) - H(X\,|\,Y) \tag{34}$$

Even though the relative entropy can be negative, the mutual information is always positive. In the continuous case, we have the same relations between entropies as in the discrete case. So, the mutual information can be expressed also as:

$$I(X;Y) = H(Y) - H(Y\,|\,X) \tag{35}$$

and

$$I(X;Y) = H(X) + H(Y) - H(X,Y) \tag{36}$$

The units of information for continuous random variables are: bits/sample if the log is base 2.

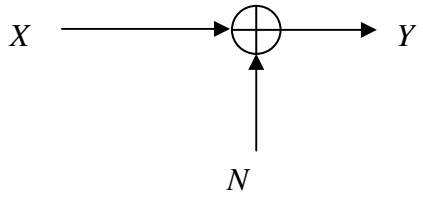
Capacity of continuous channel

As for the discrete channel, we define the capacity as being the maximum mutual information that can be transmitted by the channel. The maximization is taken over all possible distributions at the input of the channel.

$$C = \max_{all\ f_X(x)} I(X;Y) \tag{37}$$

Example: Capacity of an additive Gaussian channel.

Consider the following channel:



The random variable Y is the sum of the random variable X and the zero mean Gaussian noise N with variance σ_N^2. The random variable X is assumed to have finite variance σ_X^2. X and N are independent. The conditional pdf of Y given X is the following Gaussian:

$$f_{Y|X}(y \mid x) = \frac{1}{\sqrt{2\pi}\sigma_N} \exp\left(-\frac{(y-x)^2}{2\sigma_N^2} \right)$$

It is easy to show that $H(Y \mid X) = \frac{1}{2}\log\left(2\pi e\sigma_N^2\right)$. So, $I(X;Y) = H(Y) - \frac{1}{2}\log\left(2\pi e\sigma_N^2\right)$. We see that the mutual information is maximized when $H(Y)$ is maximized. Using theorem 1, $H(Y)$ is maximum when Y is Gaussian. Y will be Gaussian if and only if X is Gaussian. At that time, $H(Y) = \frac{1}{2}\log\left(2\pi e\sigma_Y^2\right)$. Since X and N are independent, $\sigma_Y^2 = \sigma_X^2 + \sigma_N^2$. So, the capacity of the additive Gaussian channel is:

$$C = \frac{1}{2}\log\left(1 + \frac{\sigma_X^2}{\sigma_N^2}\right) \text{ bits/sample} \tag{38}$$

C is the maximum rate of information over the channel. This means that if we want to communicate at C, the input signal should have a Gaussian distribution. We remark also that the capacity depends on the signal to noise ratio: $\frac{\sigma_X^2}{\sigma_N^2} = \frac{S}{N}$. S is the signal power σ_X^2 and N is the noise power σ_N^2.

Capacity of a bandlimited additive Gaussian Channel

If the signal samples X and N come from bandlimited processes, the sample rate should be $r = 2B$, where B is the bandwidth. At this time, if we want to express the capacity in bits/s, we have:

$$C = B\log\left(1 + \frac{S}{N}\right) \text{ bits/s} \tag{39}$$

Example:

If the signal to noise ratio is 30 dB and the bandwidth is 10 kHz, the capacity of the channel is C =99672 bits/s.

Transmission of discrete symbols over a bandlimited Gaussian channel

Let us consider a problem that we have already considered in the chapter of waveform communication. We want to transmit discrete symbols over an additive Gaussian channel. The symbol duration is T and the alphabet contains $M = 2^N$ symbols. The noise is white over the bandwidth B with a p.s.d. of $N_0/2$. As in the previous chapter, we assume that the average

energy of the symbols is *E*. This means that the average signal power is: $S = \frac{E}{T}$. The noise power is $N_0 B$. So, the capacity of the additive bandlimited white Gaussian channel becomes:

$$C = B \log\left(1 + \frac{E}{N_0 BT}\right)$$

Multiplying and dividing *C* by $\frac{E}{N_0 T}$ provides: $C = \frac{E}{N_0 T} \log\left(1 + \frac{E}{N_0 BT}\right)^{\frac{N_0 BT}{E}}$.

If we use an infinite number of dimensions for the signal space (see the orthogonal signaling), the product *BT* will become infinite. At that time, the capacity becomes:

$$C_\infty = \frac{E}{N_0 T} \log e$$

We have used the limit: $e = \lim_{x \to \infty}\left(1 + \frac{1}{x}\right)^x$.

If we transmit M equiprobable symbols, the entropy of the source will be: $H(X) = \log M = N$ in bits/symbols. So, the bit rate in bits/s will be $R_X = \frac{N}{T}$. Replacing *T* in the above expression gives: $\frac{C_\infty}{R_X} = \frac{E}{NN_0} \log e = \frac{E_b}{N_0} \log e$, where E_b is the average energy per binary digit. So, we can see that if we apply Shannon's coding theorem, we can transmit the information with an arbitrarily small probability of error if we use a large number of dimensions and if R_X is smaller than C_∞. So, we obtain:

$$\frac{E_b}{N_0} \geq \frac{1}{\log e} = \ln 2$$

The above is the same result that we have obtained using signal space concepts!