INFORMATION THEORY & CODING. EC602

Stream: ECE

Subject Name: INFORMATION THEORY & CODING

Subject Code: EC602

Contact hour: 2L+2T

Total contact hour- 40

Credits: 3

Prepared by: Dr. Arun Kumar Mondal, Dr. Avali Banerjee

Course Objective:
This course provides a basic understanding of the fundamental theories and laws of information theory and coding theory and the construction of both source codes and error-detection-correction codes and application in digital communication systems

Course Outcome

The course outcomes are to enable the students to:

CO.EC602.1	Understand the concepts of information, mutual information and entropy and various source coding techniques for a reliable digital communication system
CO.EC602.2	Analyze the need for error control techniques in a digital communication system channel models, channel capacity and channel coding techniques.
CO.EC602.3	Apply linear algebra, concept of Galois field, conjugate roots, minimal polynomial in channel coding techniques for error control.
CO.EC602.4	Generate different error control codes like linear block codes, cyclic codes, BCH codes, and perform error detection and correction.
CO.EC602.5	Design the circuit for different error control coding techniques.

Module No. 1

Source Coding

During the late 1920s, Harry Nyquist and Ralph Hartley developed a handful of fundamental ideas related to the transmission of information, particularly in the context of the telegraph as a communications system. At the time, these concepts were powerful breakthroughs individually, but they were not part of a comprehensive theory. In the 1940s, Claude Shannon developed the concept of channel capacity, based in part on the ideas of Nyquist and Hartley, and then formulated a complete theory of information and its transmission.

Uncertainty

Uncertainty is a situation which involves imperfect or unknown information. It applies to predictions of future events, to physical measurements that are already made, or to the unknown. Uncertainty arises in partially observable and/or stochastic environments, as well as due to ignorance, indolence, or both.[1] It arises in any number of fields, including insurance, philosophy, physics, statistics, economics, finance, psychology, sociology, engineering, metrology, meteorology, ecology and information science.

For example, if it is unknown whether or not it will rain tomorrow, then there is a state of uncertainty. If probabilities are applied to the possible outcomes using weather forecasts or even just a calibrated probability assessment, the uncertainty has been quantified. Suppose it is quantified as a 90% chance of sunshine. If there is a major, costly, outdoor event planned for tomorrow then there is a risk since there is a 10% chance of rain, and rain would be undesirable. Furthermore, if this is a business event and $100,000 would be lost if it rains, then the risk has been quantified (a 10% chance of losing $100,000). These situations can be made even more realistic by quantifying light rain vs. heavy rain, the cost of delays vs. outright cancellation, etc.[*citation needed*]

Some may represent the risk in this example as the "expected opportunity loss" (EOL) or the chance of the loss multiplied by the amount of the loss (10% × $100,000 = $10,000). That is useful if the organizer of the event is "risk neutral", which most people are not. Most would be willing to pay a premium to avoid the loss. An insurance company, for example, would compute an EOL as a minimum for any insurance coverage, then add onto that other operating costs and profit. Since many people are willing to buy insurance for many reasons, then clearly the EOL alone is not the perceived value of avoiding the risk.

Quantitative uses of the terms uncertainty and risk are fairly consistent from fields such as probability theory, actuarial science, and information theory. Some also create new terms without substantially changing the definitions of uncertainty or risk. For example, surprisal is a variation on uncertainty sometimes used in information theory. But outside of the more mathematical uses of the term, usage may vary widely. In cognitive psychology, uncertainty can be real, or just a matter of perception, such as expectations, threats, etc.

Information

Information is any entity or form that provides the answer to a question of some kind or resolves uncertainty. It is thus related to data and knowledge, as data represents values attributed to parameters, and knowledge signifies understanding of real things or abstract concepts. [1] As it regards data, the information's existence is not necessarily coupled to an observer (it exists beyond an event horizon, for example), while in the case of knowledge, the information requires a cognitive observer.

Information is conveyed either as the content of a message or through direct or indirect observation. That which is perceived can be construed as a message in its own right, and in that sense, information is always conveyed as the content of a message.

Information can be encoded into various forms for transmission and interpretation (for example, information may be encoded into a sequence of signs, or transmitted via a signal). It can also be encrypted for safe storage and communication.

Information reduces uncertainty. The uncertainty of an event is measured by its probability of occurrence and is inversely proportional to that. The more uncertain an event, the more information is required to resolve uncertainty of that event. The bit is a typical unit of information, but other units such as the nat may be used. For example, the information encoded in one "fair" coin flip is $\log_2(2/1) = 1$ bit, and in two fair coin flips is $\log_2(4/1) = 2$ bits.

The concept that *information is the message* has different meanings in different contexts. [2] Thus the concept of information becomes closely related to notions of constraint, communication, control, data, form, education, knowledge, meaning, understanding, mental stimuli, pattern, perception, representation, and entropy.

In information theory, *information* is taken as an ordered sequence of symbols from an alphabet, say an input alphabet χ, and an output alphabet Υ. Information processing consists of an input-output function that maps any input sequence from χ into an output sequence from Υ. The mapping may be probabilistic or deterministic. It may have memory or be memoryless. [3]

Information theory studies the quantification, storage, and communication of information. It was originally proposed by Claude E. Shannon in 1948 to find fundamental limits on signal processing and communication operations such as data compression, in a landmark paper entitled "A Mathematical Theory of Communication". Applications of fundamental topics of information theory include lossless data compression (e.g. ZIP files), lossy data compression (e.g. MP3s and JPEGs), and channel coding (e.g. for digital subscriber line (DSL)). Its impact has been crucial to the success of the Voyager missions to deep space, the invention

of the compact disc, the feasibility of mobile phones, the development of the Internet, the study of linguistics and of human perception, the understanding of black holes, and numerous other fields.

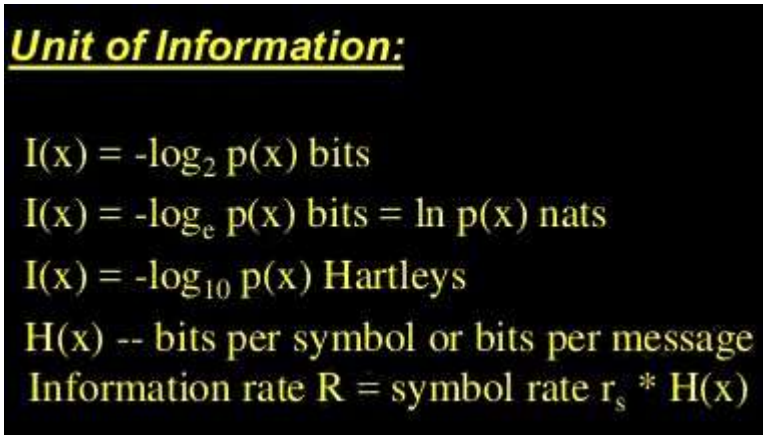
A key measure in information theory is "entropy". Entropy quantifies the amount of uncertainty involved in the value of a random variable or the outcome of a random process. For example, identifying the outcome of a fair coin flip (with two equally likely outcomes) provides less information (lower entropy) than specifying the outcome from a roll of a die (with six equally likely outcomes). Some other important measures in information theory are mutual information, channel capacity, error exponents, and relative entropy.

The field is at the intersection of mathematics, statistics, computer science, physics, neurobiology, information engineering, and electrical engineering. The theory has also found applications in other areas, including statistical inference, natural language processing, cryptography, neurobiology,[1] human vision,[2] the evolution[3] and function[4] of molecular codes (bioinformatics), model selection in statistics,[5] thermal physics,[6] quantum computing, linguistics, plagiarism detection,[7] pattern recognition, and anomaly detection.[8] Important sub-fields of information theory include source coding, channel coding, algorithmic complexity theory, algorithmic information theory, information-theoretic security, and measures of information.

Information theory is based on probability theory and statistics. Information theory often concerns itself with measures of information of the distributions associated with random variables. Important quantities of information are entropy, a measure of information in a single random variable, and mutual information, a measure of information in common between two random variables. The former quantity is a property of the probability distribution of a random variable and gives a limit on the rate at which

data generated by independent samples with the given distribution can be reliably compressed. The latter is a property of the joint distribution of two random variables, and is the maximum rate of reliable communication across a noisy channel in the limit of long block lengths, when the channel statistics are determined by the joint distribution.

The choice of logarithmic base in the following formulae determines the unit of information entropy that is used. A common unit of information is the bit, based on the binary logarithm. Other units include the nat, which is based on the natural logarithm, and the decimal digit, which is based on the common logarithm. In what follows, an expression of the form $p \log p$ is considered by convention to be equal to zero whenever $p = 0$. This is justified because $\lim_{p \to 0+} p \log p = 0$ for any logarithmic base.



INFORMATION THEORY & CODING. EC602

Entropy of an information source

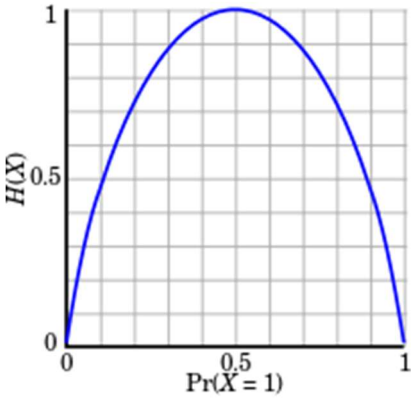
Based on the probability mass function of each source symbol to be communicated, the Shannon entropy H, in units of bits (per symbol), is given by

$$H = -\sum_{i} p_i \log_2(p_i)$$

where p_i is the probability of occurrence of the i-th possible value of the source symbol. This equation gives the entropy in the units of "bits" (per symbol) because it uses a logarithm of base 2, and this base-2 measure of entropy has sometimes been called the "shannon" in his honor. Entropy is also commonly computed using the natural logarithm (base e, where e is Euler's number), which produces a measurement of entropy in "nats" per symbol and sometimes simplifies the analysis by avoiding the need to include extra constants in the formulas. Other bases are also possible, but less commonly used. For example, a logarithm of base 2^8 = 256 will produce a measurement in bytes per symbol, and a logarithm of base 10 will produce a measurement in decimal digits (or hartleys) per symbol.

Intuitively, the entropy H_X of a discrete random variable X is a measure of the amount of *uncertainty* associated with the value of X when only its distribution is known.

The entropy of a source that emits a sequence of N symbols that are independent and identically distributed (iid) is $N \cdot H$ bits (per message of N symbols). If the source data symbols are identically distributed but not independent, the entropy of a message of length N will be less than $N \cdot H$.



The entropy of a Bernoulli trial as a function of success probability, often called the *binary entropy function*, $H_b(p)$. The entropy is maximized at 1 bit per trial when the two possible outcomes are equally probable, as in an unbiased coin toss.

If one transmits 1000 bits (0s and 1s), and the value of each of these bits is known to the receiver (has a specific value with certainty) ahead of transmission, it is clear that no information is transmitted. If, however, each bit is independently equally likely to be 0 or 1, 1000 shannons of information (more often called bits) have been transmitted. Between these two extremes, information can be quantified as follows. If $\mathbb{X}$ is the set of all messages $\{x_1, \ldots, x_n\}$ that X could be, and $p(x)$ is the probability of some

$$H(X) = \mathbb{E}_X[I(x)] = -\sum_{x \in \mathbb{X}} p(x) \log p(x).$$

,

then the entropy, H, of X is defined:[9]

(Here, $I(x)$ is the self-information, which is the entropy contribution of an individual message, and $\mathbb{E}_X$ is the expected value.) A property of entropy is that it is maximized when all the messages in the message space are equiprobable $p(x) = 1/n$; i.e., most unpredictable, in which case $H(X) = \log n$.

The special case of information entropy for a random variable with two outcomes is the *binary entropy function*, usually taken to the logarithmic base 2, thus having the shannon (Sh) as unit:

$$\mathbf{H_b(P) = -p\log_2 p - (1-p)\log_2 p.}$$

Joint entropy

The *joint entropy* of two discrete random variables X and Y is merely the entropy of their pairing: (X, Y). This implies that if X and Y are independent, then their joint entropy is the sum of their individual entropies.

For example, if (X, Y) represents the position of a chess piece — X the row and Y the column, then the joint entropy of the row of the piece and the column of the piece will be the entropy of the position of the piece.

$$H(X, Y) = \mathbb{E}_{X,Y}[-\log p(x, y)] = -\sum_{x,y} p(x, y) \log p(x, y)$$

Despite similar notation, joint entropy should not be confused with *cross entropy*.

Conditional entropy (equivocation)

The *conditional entropy* or *conditional uncertainty* of X given random variable Y (also called the *equivocation* of X about Y) is the average conditional entropy over Y:[10]

$$H(X|Y) = \mathbb{E}_Y[H(X|y)] = -\sum_{y \in Y} p(y) \sum_{x \in X} p(x|y) \log p(x|y) = -\sum_{x,y} p(x, y) \log p(x|y).$$

Because entropy can be conditioned on a random variable or on that random variable being a certain value, care should be taken not to confuse these two definitions of conditional entropy, the former of which is in more common use. A basic property of this form of conditional entropy is that:

$$H(X|Y) = H(X, Y) - H(Y).$$

Mutual information (transinformation)

Mutual information measures the amount of information that can be obtained about one random variable by observing another. It is important in communication where it can be used to maximize the amount of information shared between sent and received signals. The mutual information of *X* relative to *Y* is given by:

$$I(X;Y) = \mathbb{E}_{X,Y}[SI(x,y)] = \sum_{x,y} p(x,y) \log \frac{p(x,y)}{p(x)\,p(y)}$$

where SI (*S*pecific mutual *I*nformation) is the pointwise mutual information.

A basic property of the mutual information is that $I(X;Y) = H(X) - H(X|Y).$

That is, knowing *Y*, we can save an average of *I*(*X*; *Y*) bits in encoding *X* compared to not knowing *Y*.

Mutual information is symmetric:

$$I(X;Y) = I(Y;X) = H(X) + H(Y) - H(X,Y).$$

Mutual information can be expressed as the average Kullback–Leibler divergence (information gain) between the posterior probability distribution of *X* given the value of *Y* and the prior distribution on *X*:

$$I(X;Y) = \mathbb{E}_{p(y)}\left[D_{\mathrm{KL}}(p(X|Y=y)\|p(X))\right].$$

In other words, this is a measure of how much, on the average, the probability distribution on *X* will change if we are given the value of *Y*. This is often recalculated as the divergence from the product of the marginal distributions to the actual joint distribution:

$$I(X;Y) = D_{\mathrm{KL}}(p(X,Y)\|p(X)p(Y)).$$

Mutual information is closely related to the log-likelihood ratio test in the context of contingency tables and the multinomial distribution and to Pearson's χ^2 test: mutual information can be considered a statistic for assessing independence between a pair of variables, and has a well-specified asymptotic distribution.

Source coding theorem

In information theory, **Shannon's source coding theorem** (or **noiseless coding theorem**) establishes the limits to possible data compression, and the operational meaning of the Shannon entropy.

The **source coding theorem** shows that (in the limit, as the length of a stream of independent and identically-distributed random variable (i.i.d.) data tends to infinity) it is impossible to compress the data such that the code rate (average number of bits per symbol) is less than the Shannon entropy of the source, without it being

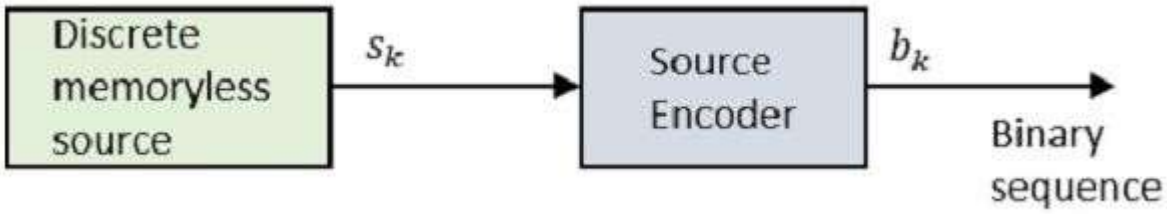
virtually certain that information will be lost. However it is possible to get the code rate arbitrarily close to the Shannon entropy, with negligible probability of loss.

The **source coding theorem for symbol codes** places an upper and a lower bound on the minimal possible expected length of codewords as a function of the entropy of the input word (which is viewed as a random variable) and of the size of the target alphabet.

The Code produced by a discrete memoryless source, has to be efficiently represented, which is an important problem in communications. For this to happen, there are code words, which represent these source codes.

For example, in telegraphy, we use Morse code, in which the alphabets are denoted by **Marks** and **Spaces**. If the letter **E** is considered, which is mostly used, it is denoted by "**.**" Whereas the letter **Q** which is rarely used, is denoted by "**--.-**"

Let us take a look at the block diagram.



Where $\mathbf{S_k}$ is the output of the discrete memoryless source and $\mathbf{b_k}$ is the output of the source encoder which is represented by **0s** and **1s**.

The encoded sequence is such that it is conveniently decoded at the receiver.

Let us assume that the source has an alphabet with **k** different symbols and that the $\mathbf{k^{th}}$ symbol $\mathbf{S_k}$ occurs with the probability $\mathbf{P_k}$, where **k = 0, 1…k-1**.

Let the binary code word assigned to symbol $\mathbf{S_k}$, by the encoder having length $\mathbf{l_k}$, measured in bits.

Hence, we define the average code word length L of the source encoder as

$L\overline{}=\sum k=0k-1pklk$

L represents the average number of bits per source symbol

If $Lmin=minimumpossiblevalueofL\overline{}$ Then

coding efficiency can be defined as

$\eta=LminL\overline{}$

With $L\overline{}\geq Lmin$ we

will have $\eta\leq 1$

However, the source encoder is considered efficient when η=1

For this, the value *Lmin* has to be determined.

Let us refer to the definition, "*Given a discrete memoryless source of entropy H(δ)*

, the average code-word length ***L*** *for any source encoding is bounded as* $L \geq H(\delta)$

."

In simpler words, the code word (example: Morse code for the word QUEUE is -.- ..- . ..- .) is always greater than or equal to the source code (QUEUE in example). Which means, the symbols in the code word are greater than or equal to the alphabets in the source code.

Hence with $Lmin=H(\delta)$
, the efficiency of the source encoder in terms of Entropy $H(\delta)$

may be written as $\eta=H(\delta)L$

This source coding theorem is called as **noiseless coding theorem** as it establishes an error-free encoding. It is also called as **Shannon's first theorem**.

Source coding is a mapping from (a sequence of) symbols from an information source to a sequence of alphabet symbols (usually bits) such that the source symbols can be exactly recovered from the binary bits (lossless source coding) or recovered within some distortion (lossy source coding). This is the concept behind data compression.

Source coding theorem

In information theory, the **source coding theorem** informally states that

N i.i.d. random variables each with entropy $H(X)$ can be compressed into more than $N H(X)$ bits with negligible risk of information loss, as $N \to \infty$; but conversely, if they are compressed into fewer than $N H(X)$ bits it is virtually certain that information will be lost.

Source coding theorem for symbol codes

Let Σ_1, Σ_2 denote two finite alphabets and let Σ_1^* and Σ_2^* denote the set of all finite words from those alphabets (respectively).

Suppose that X is a random variable taking values in Σ_1 and let f be a uniquely decodable code from Σ_1^* to Σ_2^* where $|\Sigma_2| = a$. Let S denote the random variable given by the length of codeword $f(X)$.

If f is optimal in the sense that it has the minimal expected word length for X, then (Shannon 1948):

$$\frac{H(X)}{\log_2 a} \leq \mathbb{E}S < \frac{H(X)}{\log_2 a} + 1$$

Proof: Source coding theorem

Given X is an i.i.d. source, its time series X_1, ..., X_n is i.i.d. with entropy $H(X)$ in the discrete-valued case and differential entropy in the continuous-valued case. The Source coding theorem states that for any $\varepsilon > 0$, i.e. for any rate $H(X) + \varepsilon$ larger than the entropy of the source, there is large enough n and an encoder that takes n i.i.d. repetition of the source, $X^{1:n}$, and maps it to $n(H(X) + \varepsilon)$ binary bits such that the source symbols $X^{1:n}$ are recoverable from the binary bits with probability at least $1 - \varepsilon$.

Proof of Achievability. Fix some $\varepsilon > 0$, and let

$$p(x_1, \ldots, x_n) = \Pr\left[X_1 = x_1, \cdots, X_n = x_n\right].$$

The typical set, $A\varepsilon n$, is defined as follows:

$$A_n^\varepsilon = \left\{(x_1, \cdots, x_n) \ : \ \left|-\frac{1}{n}\log p(x_1, \cdots, x_n) - H_n(X)\right| < \varepsilon\right\}.$$

The Asymptotic Equipartition Property (AEP) shows that for large enough n, the probability that a sequence generated by the source lies in the typical set, $A\varepsilon\ n$, as defined approaches one. In particular, for sufficiently large n,

$P((X_1, X_2, \cdots, X_n) \in A_n^\varepsilon)$ can be made arbitrarily close to 1, and specifically, greater than $1 - \varepsilon$ (See AEP for a proof).

The definition of typical sets implies that those sequences that lie in the typical set satisfy:

$2^{-n(H(X)+\varepsilon)} \leq p\left(x_1, \cdots, x_n\right) \leq 2^{-n(H(X)-\varepsilon)}$ Note that:

$(X_1, X_2, \cdots X_n)$ The probability of a sequence being drawn from $A\varepsilon$ n is greater than $1 - \varepsilon$.

$|A_n^\varepsilon| \leq 2^{n(H(X)+\varepsilon)}$, which follows from the left hand side (lower bound) for $p(x_1, x_2, \cdots x_n)$.

$|A_n^\varepsilon| \geq (1-\varepsilon)2^{n(H(X)-\varepsilon)}$, which follows from upper bound for $p(x_1, x_2, \cdots x_n)$

and the lower bound on the total probability of the whole set $A\varepsilon n$. Since $|A_n^\varepsilon| \leq 2^{n(H(X)+\varepsilon)}, n.(H(X)+\varepsilon)$ bits are enough to point to any string in this set.

The encoding algorithm: The encoder checks if the input sequence lies within the typical set; if yes, it outputs the index of the input sequence within the typical set; if not, the encoder outputs an arbitrary $n(H(X) + \varepsilon)$ digit number. As long as the input sequence lies within the typical set (with probability at least $1 - \varepsilon$), the encoder doesn't make any error. So, the probability of error of the encoder is bounded above by ε.

Proof of Converse. The converse is proved by showing that any set of size smaller than $A\varepsilon\ n$ (in the sense of exponent) would cover a set of probability bounded away from 1.

Proof: Source coding theorem for symbol codes

For $1 \leq i \leq n$ let s_i denote the word length of each possible x_i. Define $q_i = a^{-s_i}/C$, where C is chosen so that $q_1 + ... + q_n = 1$. Then

$$\begin{aligned}
H(X) &= -\sum_{i=1}^{n} p_i \log_2 p_i \\
&\leq -\sum_{i=1}^{n} p_i \log_2 q_i \\
&= -\sum_{i=1}^{n} p_i \log_2 a^{-s_i} + \sum_{i=1}^{n} p_i \log_2 C \\
&= -\sum_{i=1}^{n} p_i \log_2 a^{-s_i} + \log_2 C \\
&\leq -\sum_{i=1}^{n} -s_i p_i \log_2 a \\
&\leq \mathbb{E}S \log_2 a
\end{aligned}$$

where the second line follows from Gibbs' inequality and the fifth line follows from Kraft's inequality:

$$C = \sum_{i=1}^{n} a^{-s_i} \leq 1$$

so log $C \leq 0$.

For the second inequality we may set

$$s_i = \lceil -\log_a p_i \rceil$$

so that

$$-\log_a p_i \leq s_i < -\log_a p_i + 1$$

and so

$$a^{-s_i} \leq p_i$$

and

$$\sum a^{-s_i} \leq \sum p_i = 1$$

and so by Kraft's inequality there exists a prefix-free code having those word lengths. Thus the minimal S satisfies

$$\begin{aligned}
\mathbb{E}S &= \sum p_i s_i \\
&< \sum p_i \left(-\log_a p_i + 1\right) \\
&= \sum -p_i \frac{\log_2 p_i}{\log_2 a} + 1 \\
&= \frac{H(X)}{\log_2 a} + 1
\end{aligned}$$

Extension to non-stationary independent sources

Fixed Rate lossless source coding for discrete time non-stationary independent sources

Define typical set $A\varepsilon n$ as:

$$A_n^\varepsilon = \left\{ x_1^n \ : \ \left| -\frac{1}{n} \log p\left(X_1, \cdots, X_n\right) - \overline{H_n}(X) \right| < \varepsilon \right\}.$$

Then, for given $\delta > 0$, for n large enough, $\Pr(A\varepsilon n) > 1 - \delta$. Now we just encode the sequences in the typical set, and usual methods in source coding show that the cardinality of this set is smaller than $2^{n(\overline{H_n}(X)+\varepsilon)}$.

Thus, on an average, $H_n(X) + \varepsilon$ bits suffice for encoding with probability greater than $1 - \delta$, where ε and δ can be made arbitrarily small, by making n larger.

INFORMATION THEORY & CODING. EC602

Source coding technique

In signal processing, **data compression**, **source coding**, or **bit-rate reduction** involves encoding information using fewer bits than the original representation. Compression can be either lossy or lossless. Lossless compression reduces bits by identifying and eliminating statistical redundancy. No information is lost in lossless compression. Lossy compression reduces bits by removing unnecessary or less important information.

The process of reducing the size of a data file is often referred to as data compression. In the context of data transmission, it is called source coding; encoding done at the source of the data before it is stored or transmitted.[4] Source coding should not be confused with channel coding, for error detection and correction or line coding, the means for mapping data onto a signal.

Compression is useful because it reduces resources required to store and transmit data. Computational resources are consumed in the compression process and, usually, in the reversal of the process (decompression). Data compression is subject to a space–time complexity trade-off. For instance, a compression scheme for video may require expensive hardware for the video to be decompressed fast enough to be viewed as it is being decompressed, and the option to decompress the video in full before watching it may be inconvenient or require additional storage. The design of data compression schemes involves trade-offs among various factors, including the degree of compression, the amount of distortion introduced (when using lossy data compression), and the computational resources required to compress and decompress the data.

Entropy encoding

In information theory an **entropy encoding** is a lossless data compression scheme that is independent of the specific characteristics of the medium.

One of the main types of entropy coding creates and assigns a unique prefix-free code to each unique symbol that occurs in the input. These entropy encoders then compress data by replacing each fixed-length input symbol with the corresponding variable-length prefix-free output codeword. The length of each codeword is approximately proportional to the negative logarithm of the probability. Therefore, the most common symbols use the shortest codes.

According to Shannon's source coding theorem, the optimal code length for a symbol is $-\log_b P$, where b is the number of symbols used to make output codes and P is the probability of the input symbol.

Two of the most common entropy encoding techniques are Huffman coding and arithmetic coding.[1] If the approximate entropy characteristics of a data stream are known in advance (especially for signal compression), a simpler static code may be useful. These static codes include universal codes (such as Elias gamma coding or Fibonacci coding) and Golomb codes (such as unary coding or Rice coding).

Since 2014, data compressors have started using the Asymmetric Numeral Systems family of entropy coding techniques, which allows combination of the compression ratio of arithmetic coding with a processing cost similar to Huffman coding.

Huffman Coding

Huffman coding is based on the frequency of occurance of a data item (pixel in images). The principle is to use a lower number of bits to encode the data that occurs more frequently. Codes are stored in a *Code Book* which may be constructed for each image or a set of images. In all cases the code book plus encoded data must be transmitted to enable decoding.

The Huffman algorithm is now briefly summarised:

- A bottom-up approach

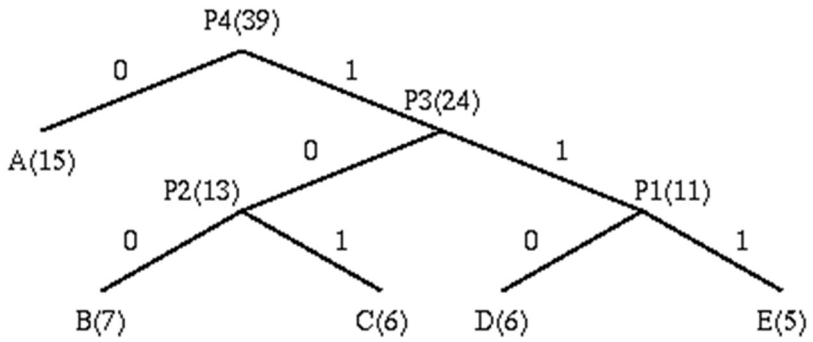
1. Initialization: Put all nodes in an OPEN list, keep it sorted at all times (e.g., ABCDE).

2. Repeat until the OPEN list has only one node left:

(a) From OPEN pick two nodes having the lowest frequencies/probabilities, create a parent node of them.

(b) Assign the sum of the children's frequencies/probabilities to the parent node and insert it into OPEN.

(c) Assign code 0, 1 to the two branches of the tree, and delete the children from OPEN.



Symbol	Count	log(1/p)	Code	Subtotal (# of bits)
A	15	1.38	0	15
B	7	2.48	100	21
C	6	2.70	101	18
D	6	2.70	110	18
E	5	2.96	111	15
			TOTAL (# of bits):	87

The following points are worth noting about the above algorithm:

- Decoding for the above two algorithms is trivial as long as the coding table (the statistics) is sent before the data. (There is a bit overhead for sending this, negligible if the data file is big.)
- **Unique Prefix Property**: no code is a prefix to any other code (all symbols are at the leaf nodes) -> great for decoder, unambiguous.

- If prior statistics are available and accurate, then Huffman coding is very good.

In the above example:

Number of bits needed for Huffman Coding is: 87 / 39 = 2.23

The Shannon-Fano Coding

This is a basic information theoretic algorithm. A simple example will be used to illustrate the algorithm:

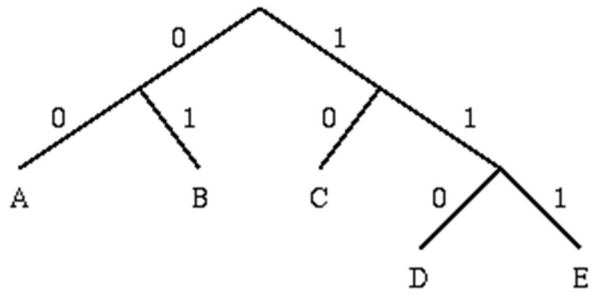
Symbol	A	B	C	D	E
Count	15	7	6	6	5

Encoding for the Shannon-Fano Algorithm:

- A top-down approach

1. Sort symbols according to their frequencies/probabilities, e.g., ABCDE.

2. Recursively divide into two parts, each with approx. same number of counts.



Symbol	Count	log(1/p)	Code	Subtotal (# of bits)
A	15	1.38	00	30
B	7	2.48	01	14
C	6	2.70	10	12
D	6	2.70	110	18
E	5	2.96	111	15

TOTAL (# of bits): 89

Module No. 2

INFORMATION THEORY & CODING. EC602

Channel Capacity and Coding

Channel Models

In a digital channel model, the transmitted message is modelled as a digital signal at a certain protocol layer. Underlying protocol layers, such as the physical layer transmission technique, is replaced by a simplified model. The model may reflect channel performance measures such as bit rate, bit errors, latency/delay, delay jitter, etc. Examples of digital channel models are:

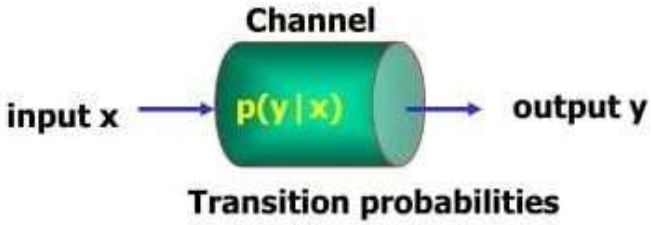
Communication Channel

A (discrete) channel is a system consisting of an input alphabet X and output alphabet Y and a probability transition matrix $p(y|x)$ that expresses the probability of observing the output symbol y given that we send the symbol x

Examples of channels:

CDs, CD – ROMs, DVDs, phones,

Ethernet, Video cassettes etc.



Memoryless:

- output only on input
- input and output alphabet finite

Types of Channels – Symmetric channel

Every row of the channel matrix contains same set of numbers p_1 to p_J.

Every column of the channel matrix contains same set of numbers q_1 to q_J.

0.2	0.2	0.3	0.3
0.3	0.3	0.2	0.2

0.2	0.3	0.5
0.3	0.5	0.2
0.5	0.2	0.3

Types of Channels – Symmetric channel

Binary symmetric channel – special case.

0 and 1 transmitted and received.

ϵ is probability of error.

$$\begin{matrix} 1-\epsilon & \epsilon \\ \epsilon & 1-\epsilon \end{matrix} \qquad 0 \leq \epsilon \leq 1$$

H(Y/X) is independent of input distribution and solely determined by channel matrix

$$H(Y/X) = -\sum_{i=1}^{m} \sum_{j=1}^{n} p(x_i, y_j) \log p(y_j / x_i)$$

Types of Channels – Symmetric channel

- $H(Y/X) = -\sum_{i=1}^{m} \sum_{j=1}^{n} p(x_i,y_j) \log p(y_j / x_i)$
- $H(Y/X) = -\Sigma_i p(x_i) \ \Sigma_j p(y_j / x_i) \log p(y_j / x_i)$
- (Check with given matrix.)

$p(x_1)\{ (1-\epsilon) \log(1-\epsilon)+ \epsilon \log\epsilon\} + p(x_2)\{ (1-\epsilon)\log(1-\epsilon)+ \epsilon\log \epsilon\}$

- Generalizing—
- $H(Y/X) = \Sigma_j p(y_j / x_i) \log p(y_j / x_i)$

- H(Y/X) is independent of input distribution and solely determined by channel matrix

$$x_3 \begin{bmatrix} 0.2 & 0.5 & 0.3 \end{bmatrix}$$

Types of Channels – Lossless channel

- Output uniquely specifies the input.
- H(X/Y) = 0 Noise less channel
- Matrix has one and only one non-zero element in each column.
- Channel with error probability 0 as well as 1 are noiseless channels.
- P(Y/X) =

½	½	0	0	0	0
0	0	3/5	3/10	1/10	0
0	0	0	0	0	1

Types of Channels – Lossless channel

- Output uniquely specifies the input.
- H(X/Y) = 0 Noise less channel
- Matrix has one and only one non-zero element in each column.
- Channel with error probability 0 as well as 1 are noiseless channels.
- P(Y/X) =

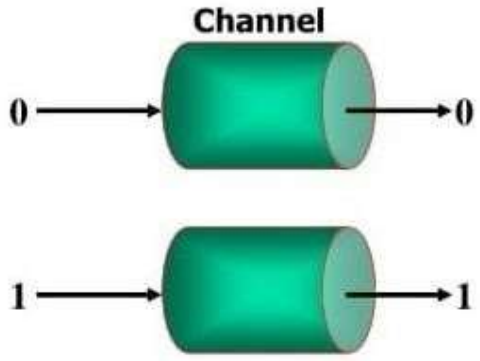
Types of Channels – Deterministic channel

- Input uniquely specifies the output.
- H(Y/X) = 0
- Matrix has one and only one non-zero element in each row.
- Also Sum in row should be 1.
- Elements are either 0 or 1
- P(Y/X) =

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Noiseless binary channel

Noiseless binary channel



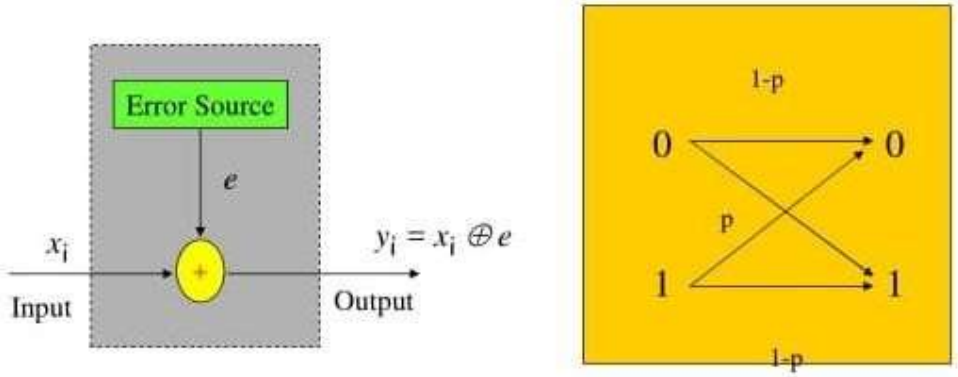
Transition Matrix

p(y | x) =

	0	1
0	1	0
1	0	1

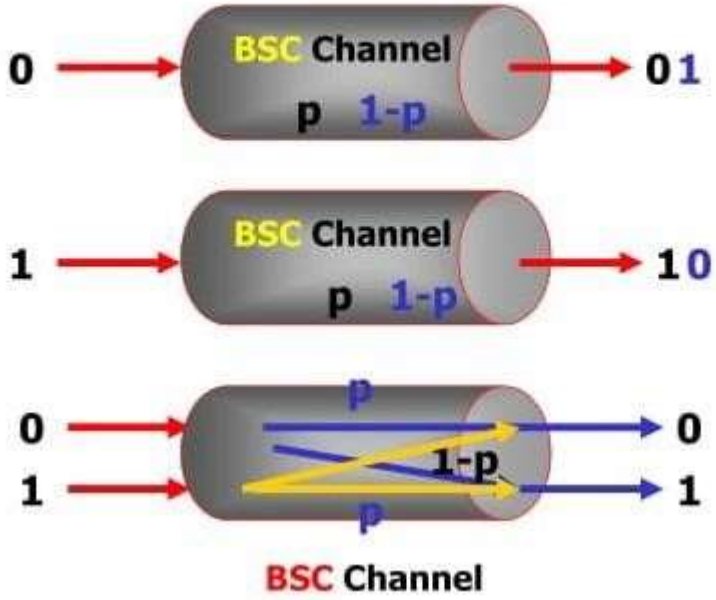
Binary Symmetric Channel (BSC)

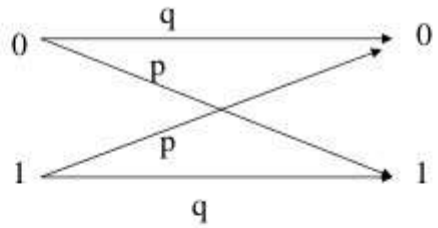
(Noisy channel)



Binary Symmetric Channel (BSC)

(Noisy channel)





p + q = 1

- q is probability of correct reception .
- p is probability of wrong reception.
- We have to find C, channel capacity.
- p(0) = p(1) = 0.5
- As p(X) is given, assume above is p(Y/X).
- Find p(X,Y) and p(Y).
- Find C = {H(Y) – H(Y/X)}max
- p(X,Y) = p/2 q/2
  q/2 p/2
- **C = 1+ p log p +q log q**

DES
– BI

One of the

One of the
Symmetric

$P_{11} = p_{22}$

$P_{12} = p_{21}$

Cha

nnel Capacity

In information theory, the Shannon–Hartley theorem tells the maximum rate at which information can be transmitted over a communications channel of a specified bandwidth in the presence of noise. It is an application of the noisy-channel coding theorem to the archetypal case of a continuous-time analog communications channel subject to Gaussian noise. The theorem establishes Shannon's channel capacity for such a communication link, a bound on the maximum amount of error-free information per time unit that can be transmitted with a specified bandwidth in the presence of the noise interference, assuming that the signal power is bounded, and that the Gaussian noise process is characterized by a known power or power spectral density. The law is named after Claude Shannon and Ralph Hartley.

Statement of the theorem

The Shannon–Hartley theorem states the channel capacity *C*, meaning the theoretical tightest upper bound on the information rate of data that can be communicated at an arbitrarily low error rate using an average received signal power *S* through an analog communication channel subject to additive white Gaussian noise of power *N*:

$$C = B \log_2 \left(1 + \frac{S}{N}\right)$$

where

C is the channel capacity in bits per second, a theoretical upper bound on the net bit rate (information rate, sometimes denoted *I*) excluding error-correction codes;
B is the bandwidth of the channel in hertz (passband bandwidth in case of a bandpass signal);

S is the average received signal power over the bandwidth (in case of a carrier-modulated passband transmission, often denoted *C*), measured in watts (or volts squared);
N is the average power of the noise and interference over the bandwidth, measured in watts (or volts squared); and
S/N is the signal-to-noise ratio (SNR) or the carrier-to-noise ratio (CNR) of the communication signal to the noise and interference at the receiver (expressed as a linear power ratio, not as logarithmic decibels).

Nyquist rate

Main article: Nyquist rate

In 1927, Nyquist determined that the number of independent pulses that could be put through a telegraph channel per unit time is limited to twice the bandwidth of the channel. In symbols, $f_p \leq 2B$

where f_p is the pulse frequency (in pulses per second) and B is the bandwidth (in hertz). The quantity $2B$ later came to be called the *Nyquist rate*, and transmitting at the limiting pulse rate of $2B$ pulses per second as *signalling at the Nyquist rate*. Nyquist published his results in 1928 as part of his paper "Certain topics in Telegraph Transmission Theory."

Hartley's law

During 1928, Hartley formulated a way to quantify information and its line rate (also known as data signalling rate *R* bits per second).[1] This method, later known as Hartley's law, became an important precursor for Shannon's more sophisticated notion of channel capacity.

Hartley argued that the maximum number of distinguishable pulse levels that can be transmitted and received reliably over a communications channel is limited by the dynamic range of the signal amplitude and the precision with which the receiver can distinguish amplitude levels. Specifically, if the amplitude of the transmitted signal is restricted to the range of $[-A \ldots +A]$ volts, and the precision of the receiver is $\pm\Delta V$ volts, then the maximum number of distinct pulses *M* is given by

$$M = 1 + \frac{A}{\Delta V}.$$

By taking information per pulse in bit/pulse to be the base-2-logarithm of the number of distinct messages *M* that could be sent, Hartley[2] constructed a measure of the line rate *R* as:

$$R = f_p \log_2(M),$$

where f_p is the pulse rate, also known as the symbol rate, in symbols/second or baud.
Hartley then combined the above quantification with Nyquist's observation that the number of independent pulses that could be put through a channel of bandwidth *B* hertz was $2B$ pulses per second, to arrive at his quantitative measure for achievable line rate.

Hartley's law is sometimes quoted as just a proportionality between the analog bandwidth, *B*, in Hertz and what today is called the digital bandwidth, *R*, in bit/s.[3] Other times it is quoted in this more quantitative form, as an achievable line rate of *R* bits per second:[4]

$$R \leq 2B\log_2(M).$$

Hartley did not work out exactly how the number *M* should depend on the noise statistics of the channel, or how the communication could be made reliable even when individual symbol pulses could not be reliably distinguished to *M* levels; with Gaussian noise statistics, system designers had to choose a very conservative value of *M* to achieve a low error rate.

The concept of an error-free capacity awaited Claude Shannon, who built on Hartley's observations about a logarithmic measure of information and Nyquist's observations about the effect of bandwidth limitations.

Hartley's rate result can be viewed as the capacity of an errorless *M*-ary channel of 2*B* symbols per second. Some authors refer to it as a capacity. But such an errorless channel is an idealization, and if M is chosen small enough to make the noisy channel nearly errorless, the result is necessarily less than the Shannon capacity of the noisy channel of bandwidth *B*, which is the Hartley–Shannon result that followed later.

Noisy channel coding theorem and capacity

Main article: Noisy-channel coding theorem

Claude Shannon's development of information theory during World War II provided the next big step in understanding how much information could be reliably communicated through noisy channels. Building on Hartley's foundation, Shannon's noisy channel coding theorem (1948) describes the maximum possible efficiency of error-correcting methods versus levels of noise interference and data corruption.[5][6] The proof of the theorem shows that a randomly constructed error-correcting code is essentially as good as the best possible code; the theorem is proved through the statistics of such random codes.

Shannon's theorem shows how to compute a channel capacity from a statistical description of a channel, and establishes that given a noisy channel with capacity C and information transmitted at a line rate *R*, then if

$$R < C$$

there exists a coding technique which allows the probability of error at the receiver to be made arbitrarily small. This means that theoretically, it is possible to transmit information nearly without error up to nearly a limit of C bits per second.

The converse is also important. If

$$R > C$$

the probability of error at the receiver increases without bound as the rate is increased. So no useful information can be transmitted beyond the channel capacity. The theorem does not address the rare situation in which rate and capacity are equal.

The Shannon–Hartley theorem establishes what that channel capacity is for a finite-bandwidth continuoustime channel subject to Gaussian noise. It connects Hartley's result with Shannon's channel

capacity theorem in a form that is equivalent to specifying the *M* in Hartley's line rate formula in terms of a signalto-noise ratio, but achieving reliability through error-correction coding rather than through reliably distinguishable pulse levels.

If there were such a thing as a noise-free analog channel, one could transmit unlimited amounts of errorfree data over it per unit of time (Note: An infinite-bandwidth analog channel can't transmit unlimited amounts of error-free data, without infinite signal power). Real channels, however, are subject to limitations imposed by both finite bandwidth and nonzero noise.

Bandwidth and noise affect the rate at which information can be transmitted over an analog channel. Bandwidth limitations alone do not impose a cap on the maximum information rate because it is still possible for the signal to take on an indefinitely large number of different voltage levels on each symbol pulse, with each slightly different level being assigned a different meaning or bit sequence. Taking into account both noise and bandwidth limitations, however, there is a limit to the amount of information that can be transferred by a signal of a bounded power, even when sophisticated multi-level encoding techniques are used.

In the channel considered by the Shannon–Hartley theorem, noise and signal are combined by addition. That is, the receiver measures a signal that is equal to the sum of the signal encoding the desired information and a continuous random variable that represents the noise. This addition creates uncertainty as to the original signal's value. If the receiver has some information about the random process that generates the noise, one can in principle recover the information in the original signal by considering all possible states of the noise process. In the case of the Shannon–Hartley theorem, the noise is assumed to be generated by a Gaussian process with a known variance. Since the variance of a Gaussian process is equivalent to its power, it is conventional to call this variance the noise power.

Such a channel is called the Additive White Gaussian Noise channel, because Gaussian noise is added to the signal; "white" means equal amounts of noise at all frequencies within the channel bandwidth. Such noise can arise both from random sources of energy and also from coding and measurement error at the sender and receiver respectively. Since sums of independent Gaussian random variables are themselves Gaussian random variables, this conveniently simplifies analysis, if one assumes that such error sources are also Gaussian and independent.

Comparison of Shannon's capacity to Hartley's law

Comparing the channel capacity to the information rate from Hartley's law, we can find the effective number of distinguishable levels *M*:[7]

$$2B\log_2(M) = B\log_2\left(1+\frac{S}{N}\right)$$

$$M = \sqrt{1+\frac{S}{N}}.$$

The square root effectively converts the power ratio back to a voltage ratio, so the number of levels is approximately proportional to the ratio of signal RMS amplitude to noise standard deviation.

This similarity in form between Shannon's capacity and Hartley's law should not be interpreted to mean that *M* pulse levels can be literally sent without any confusion. More levels are needed to allow for redundant coding and error correction, but the net data rate that can be approached with coding is equivalent to using that *M* in Hartley's law.

Alternative forms

Frequency-dependent (colored noise) case

In the simple version above, the signal and noise are fully uncorrelated, in which case $S + N$ is the total power of the received signal and noise together. A generalization of the above equation for the case where the additive noise is not white (or that the S/N is not constant with frequency over the bandwidth) is obtained by treating the channel as many narrow, independent

Gaussian channels in parallel:

$$C = \int_0^B \log_2\left(1 + \frac{S(f)}{N(f)}\right) df$$

where

C is the channel capacity in bits per second;
B is the bandwidth of the channel in Hz;
S(f) is the signal power spectrum
N(f) is the noise power spectrum f
is frequency in Hz.

Note: the theorem only applies to Gaussian stationary process noise. This formula's way of introducing frequency-dependent noise cannot describe all continuous-time noise processes. For example, consider a noise process consisting of adding a random wave whose amplitude is 1 or −1 at any point in time, and a channel that adds such a wave to the source signal. Such a wave's frequency components are highly dependent. Though such a noise may have a high power, it is fairly easy to transmit a continuous signal with much less power than one would need if the underlying noise was a sum of independent noises in each frequency band.

Approximations

For large or small and constant signal-to-noise ratios, the capacity formula can be approximated:

- If S/N >> 1, then

$$C \approx 0.332 \cdot B \cdot \mathrm{SNR\ (in\ dB)}$$

where

$$\text{SNR (in dB)} = 10 \log_{10} \frac{S}{N}.$$

- Similarly, if S/N << 1, then

$$C \approx 1.44 \cdot B \cdot \frac{S}{N}.$$

In this low-SNR approximation, capacity is independent of bandwidth if the noise is white, of spectral density N_0 watts per hertz, in which case the total noise power is B.N_0.

$$C \approx 1.44 \cdot \frac{S}{N_0}$$

INFORMATION THEORY & CODING. EC602

MODULE 3

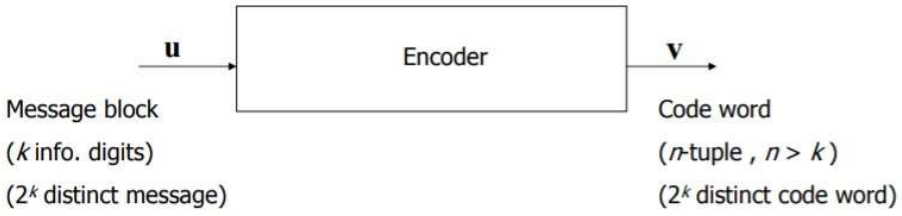
Linear Block Codes

Introduction to Linear Block Codes

This set of 2^k code words is called a block code.

For a block code to be useful, there should be a one-to-one correspondence between a message u and its code word v.

A desirable structure for a block code to possess is the linearity. With this structure, the encoding complexity will be greatly reduced.



Block code: k message bits encoded to n code bits i.e., each of 2^k messages encoded into a unique n-bit codeword via a linear transformation.

Property: Sum of any two codewords is also a codeword necessary and sufficient for code to be linear.

(n,k) code has rate k/n.

Sometime written as (n,k,d), where d is the minimum Hamming Distance of the code.

Generator Matrix of Linear Block Code

INFORMATION THEORY & CODING. EC602

Linear transformation:

C=D.G

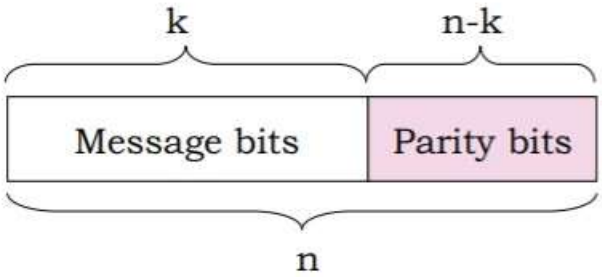
C is an n-element row vector containing the codeword
D is a k-element row vector containing the message
G is the kxn generator matrix

Each codeword bit is a specified linear combination of message bits.

Each codeword is a linear combination of rows of G.

(n,k) Systematic Linear Block Codes

• Split data into k-bit blocks
• Add (n-k) parity bits to each block using (n-k) linear equations, making each block n bits long



• Every linear code can be represented by an equivalent systematic form --- ordering is not significant, direct inclusion of k message bits in n-bit codeword is.

Since an (n, k) linear code C is a k-dimensional subspace of the vector space Vn of all the binary ntuple, it is possible to find k linearly independent code word, g_0 , g_1 ,…, g_{k-1} in C

$$\mathbf{v} = u_0\mathbf{g}_0 + u_1\mathbf{g}_1 + \cdots + u_{k-1}\mathbf{g}_{k-1} \qquad (3.1)$$

where $u_i = 0$ or 1 for $0 \leq i < k$

Let us arrange these k linearly independent code words as the rows of a k × n matrix as follows:

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ . \\ . \\ . \\ . \\ \mathbf{g}_{k-1} \end{bmatrix} = \begin{bmatrix} g_{00} & g_{01} & g_{02} & \cdot & \cdot & g_{0,n-1} \\ g_{10} & g_{11} & g_{12} & \cdot & \cdot & g_{1,n-1} \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ g_{k-1,0} & g_{k-1,1} & g_{k-1,2} & \cdot & \cdot & g_{k-1,n-1} \end{bmatrix} \qquad (3.2)$$

If $u = (u_0, u_1, \ldots, u_{k-1})$ is the message to be encoded, the corresponding code word can be given as follows:

$v = u.G$

$$= (u_0, u_1, \ldots, u_{k-1}) \bullet \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ . \\ . \\ . \\ \mathbf{g}_{k-1} \end{bmatrix} \qquad (3.3)$$

$$= u_0\mathbf{g}_0 + u_1\mathbf{g}_1 + \cdots + u_{k-1}\mathbf{g}_{k-1}$$

• Corresponds to using invertible transformations on rows and permutations on columns of G to get

• G = [I | A] --- identity matrix in the first k columns

A linear systematic (n, k) code is completely specified by a k × n matrix G of the following form

INFORMATION THEORY & CODING. EC602

$$G_{kxn} = \left[\; I_{k\times k} \;\middle|\; A_{k\times(n-k)} \;\right]$$

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \mathbf{g}_2 \\ . \\ . \\ . \\ \mathbf{g}_{k-1} \end{bmatrix} = \left[\begin{array}{cccc|ccccccc} p_{00} & p_{01} & . & . & p_{0,n-k-1} & 1 & 0 & 0 & . & . & 0 \\ p_{10} & p_{11} & . & . & p_{1,n-k-1} & 0 & 1 & 0 & . & . & 0 \\ p_{20} & p_{21} & . & . & p_{2,n-k-1} & 0 & 0 & 1 & . & . & 0 \\ & & & & & & & & & & \\ & & & & & & & & & & \\ & & & & & & & & & & \\ p_{k-1,0} & p_{k-1,1} & . & . & p_{k-1,n-k-1} & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right] \qquad (3.4)$$

P matrix — $k \times k$ identity matrix

where $p_{ii} = 0$ or 1

For any k × n matrix G with k linearly independent rows, there exists an (n-k) ×n matrix H with n-k linearly independent rows such that any vector in the row space of G is orthogonal to the rows of H and any vector that is orthogonal to the rows of H is in the row space of G.

An n-tuple v is a code word in the code generated by G if and only if $v \bullet H^T = 0$

This matrix H is called a parity-check matrix of the code

The 2n-k linear combinations of the rows of matrix H form an (n, n – k) linear code C_d

This code is the null space of the (n, k) linear code C generated by matrix G

C_d is called the dual code of C

If the generator matrix of an (n,k) linear code is in the systematic form of (3.4), the parity-check matrix may take the following form

$$\mathbf{H} = \left[\mathbf{I}_{n-k} \quad \mathbf{P}^T\right]$$

$$= \begin{bmatrix} 1 & 0 & 0 & . & . & . & 0 & p_{00} & p_{10} & . & . & . & p_{k-1,0} \\ 0 & 1 & 0 & . & . & . & 0 & p_{01} & p_{11} & . & . & . & p_{k-1,1} \\ 0 & 0 & 1 & . & . & . & 0 & p_{02} & p_{12} & . & . & . & p_{k-1,2} \\ . & & & & & & & & & & & & \\ . & & & & & & & & & & & & \\ . & & & & & & & & & & & & \\ 0 & 0 & 0 & . & . & . & 1 & p_{0,n-k-1} & p_{1,n-k-1} & . & . & . & p_{k-1,n-k-1} \end{bmatrix}$$

Let h_j be the j_{th} row of H

$$\mathbf{g}_i \cdot \mathbf{h}_j = p_{ij} + p_{ij} = 0$$

for $0 \leq i < k$ and $0 \leq j < n - k$

This implies that $G \bullet H^T = 0$.= + = 0

Let $u = (u_0, u_1, \ldots, u_{k-1})$ be the message to be encoded

In systematic form the corresponding code word would be

$v = (v_0, v_1, \ldots, v_{n-k-1}, u_0, u_1, \ldots, u_{k-1})$

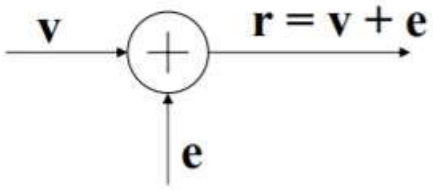
Using the fact that $v \bullet H_T = 0$, we obtain

$$v_j + u_0 p_{0j} + u_1 p_{1j} + \cdots + u_{k-1} p_{k-1,j} = 0$$

Syndrome and Error Detection

Let $v = (v_0, v_1, \ldots, v_{n-1})$ be a code word that was transmitted over a noisy channel

Let $r = (r_0, r_1, \ldots, r_{n-1})$ be the received vector at the output of the channel



$e = r + v = (e_0, e_1, \ldots, e_{n-1})$ is an n-tuple

$e_i = 1$ for $r_i \neq v_i$

$e_i = 0$ for ri = vi

The n-tuple e is called the error vector (or error pattern)

Upon receiving r, the decoder must first determine whether r contains transmission errors

If the presence of errors is detected, the decoder will take actions to locate the errors, Correct errors (FEC) , Request for a retransmission of v (ARQ)

When r is received, the decoder computes the following (n – k)-tuple :

$s = r \cdot H^T$

s = 0 if and only if r is a code word and receiver accepts r as the transmitted code word $s \neq 0$ if and only if r is not a code word and the presence of errors has been detected

When the error pattern e is identical to a nonzero code word (i.e., r contain errors but $s = r \cdot H^T = 0$), error patterns of this kind are called undetectable error patterns

Since there are $2^k – 1$ nonzero code words, there are $2^k – 1$ undetectable error patterns

Since r is the vector sum of v and e, it follows from (3.10) that s
$= r \cdot H^T = (v + e) \cdot H^T = v \cdot H^T + e \cdot H^T$

however, $v \cdot H^T = 0$

consequently, we obtain the following relation between the syndrome and the error pattern :
$s = e \cdot H^T$

The Minimum Distance of a Block Code

Let v and w be two n-tuple, the Hamming distance between v and w, denoted d(v,w), is defined as the number of places where they differ
For example, the Hamming distance between v = (1 0 0 1 0 1 1) and w = (0 1 0 0 0 1 1) is 3

Error Detection

A linear block code can detect all error patterns of dmin − 1 or fewer errors.

If e 6= 0 is a codeword, then no errors are detected

There are 2k−1 undetectable error patterns, but there are 2n − 1 possible nonzero error patterns.

The number of detectable error patterns is $2^n – 1 – (2^k – 1) = 2^n – 2^k$

Usually, $2^k – 1$ is a small fraction of $2^n – 2^k$.

INFORMATION THEORY & CODING. EC602

Error Correction

A linear block code can correct all error patterns of t or fewer errors, where $t = \lfloor d_{min} - 1 \rfloor / 2$ and $\lfloor x \rfloor$ is the largest integer $\leq x$.

A code is usually capable of correcting many error patterns of t + 1 or more errors, but not all of them. In fact, up to 2^{n-k} error patters may be corrected, equal to the number of syndromes.

The Standard Array

1. Write out all 2^k codewords in a row starting with $c_0 = 0$.

2. From the remaining $2^n - 2^k$ n-tuples, select an error pattern e_2 of weight 1 and place it under c_0. Under each codeword put $c_i + e_2$

3. Select a minimum weight error patter e_3 from the remaining unused n-tuples and place it under c0 = 0. Under each codeword put $c_i + e_3$.

4. Repeat 3) until all n-tuples have been used.

Perfect Codes

The packing radius is the radius of the largest sphere that can be drawn around every codeword in n−space such that no two spheres intersect. ¤ The value of this radius is b(dmin − 1)/2c.

The covering radius of a code is the radius of the smallest sphere that can be drawn about every codeword such that every point in n−space is included.

A perfect code is one whose packing and covering radii are equal.

A perfect code satisfies the Hamming bound with equality.
A quasi-perfect code is one for which the covering radius equals the packing radius plus one.

Hamming Codes

• Hamming codes correct single errors with the minimum number of parity bits: $n = 2^{n-k} - 1$
• (7,4,3)
• (15,11,3)
• (2m –1,2m -1-m,3)

INFORMATION THEORY & CODING. EC602

INFORMATION THEORY & CODING. EC602

MODULE 4

Cyclic Codes

Introduction

Binary cyclic codes form a subclass of linear block codes. Easier to encode and decode

Definition A (n, k) linear block code C is called a cyclic code if

1. The sum of any two codewords in the code is also a codeword. (Linear)

Example: Ci + Cj = Ck

2. Any cyclic shift of a codeword in the code is also a codeword. (Cyclic)

Example: If $C = [C_0 \ C_1 \ ..... \ C_{n-1}]$ is a codeword,

$C^{(1)} = [C_{n-1} \ C_0 \ ..... \ C_{n-3} \ C_{n-2}]$

$C^{(2)} = [C_{n-2} \ C_{n-1} \ ..... \ C_{n-4} \ C_{n-3}]$.

.

$C^{(n-1)} = [C_1 \ C_2 \ ..... \ C_{n-1} \ C_0]$

are also codewords.

We can represent the code word $C=[C_0 \ C_1 \ \dots \ C_{n-1}]$ by a code polynomial

$C(X) = C_0 + C_1 X + ... + C_{n-1} X^{n-1}$

The coefficients $C_i = \{0,1\}$ and each power of X in the polynomial C(X) represents a one-bit shift in time. Hence, multiplication of the polynomial C(X) by X may be viewed as a shift to the right.

Example: C= [1101] can be represented by

$C(X) = 1 + X + X^3$

$C^{(i)}(X)$ is recognized as the code polynomial of the code word $[C_{n-i} \ \dots \ C_{n-1} \ C_0 \ C_1 \ \dots \ C_{n-i-1}]$ obtained by applying i cyclic shifts to the code word $[C_0 \ C_1 \ \dots C_{n-1}]$.
It can be shown that $C^{(i)}(X)$ is the remainder resulting from dividing $X^iC(X)$ by $X^n + 1$. That is,

INFORMATION THEORY & CODING. EC602

$X^iC(X) = q(X)(X^n + 1) + C^{(i)}(X)$

where $q(X) = C^{n-1} + C^{n-i+1} X + ... + C_{n-1}X^{i-1}$

Generator Polynomial

If g(X) is a polynomial of degree (n - k) and is a factor of X^n+1, then g(X) generates an (n, k) cyclic code in which the code polynomial C(X) for a data vector M = [m_0 m_1 m_2 . . . m_{k-1}] is generated by C(X) = M(X) g(X)

where $C(X) = C_0 + C_1X + C_2X^2 + ... + C_{n-1} X^{n-1}$
$M(X) = m_0 + mX + m_2X^2 + ... + m_{k-1} X^{k-1}$
$g(X) = g_0 + g_1 X + g_2X_2 + ... + g_{n-k} X_{n-k}$
g(X) is the generating polynomial

Systematic cyclic code generation

Let the message polynomial be defined by

$$M(X) = m_0 + mX + m_2X^2 + ... + m_{k-1} X^{k-1}$$

and $B(X) = b_0 + b_1 X + b_2X^2 + ... + b_{n-k-1} X^{n-k-1}$

We want the code polynomial to be in the form

$C(X) = B(X) + X^{n-k} M(X)$

Hence, $A(X)g(X) = B(X) + X^{n-k} M(X)$

Equivalently, we may write

$$\frac{X^{n-k}M(X)}{g(X)} = A(X) + \frac{B(X)}{g(X)}$$

This equation states that the polynomial B(X) is the remainder left over after dividing $X^{n-k}M(X)$ by g(x).

Parity-check polynomial

INFORMATION THEORY & CODING. EC602

An (n,k) cyclic code is uniquely specified by its generator polynomial g(X) of order (n-k). Such a code is also uniquely specified by another polynomial of degree k, which is called the parity-check polynomial, defined by

$h(X) = 1 + h^1 X + h_2 X^2 + ... + h_{k-1} X^{k-1} + X^k$

In linear block code, we have $GH^T = 0$. Now, we have $g(X)h(X) \bmod(X^n + 1) = 0$ and we state that

$g(X)h(X) = (X^n + 1)$

Syndrome

Let the received word be $[r_0 \quad r_1 \quad ..... \quad r_{n-1}]$ and

$R(X) = r_0 + r_1X \quad ..... \quad r_{n-1}X_{n-1}$

Now, $R(X) = q(X)g(X) + S(X)$

where S(X) is called syndrome polynomial because its coefficients make up the syndrome S.

Systematic Encoding of Cyclic Codes

To encode a k-bit message $[u_0 \quad u_1 \cdot \cdot \cdot u_{k-1}]$ construct the message polynomial

$u(X) = u_0 + u_1X + \cdot \cdot \cdot + u_{k-1}X^{k-1}$

Given a generator polynomial g(X) of an (n, k) cyclic code, the corresponding codeword is u(X)g(X).

This is not a systematic encoding.

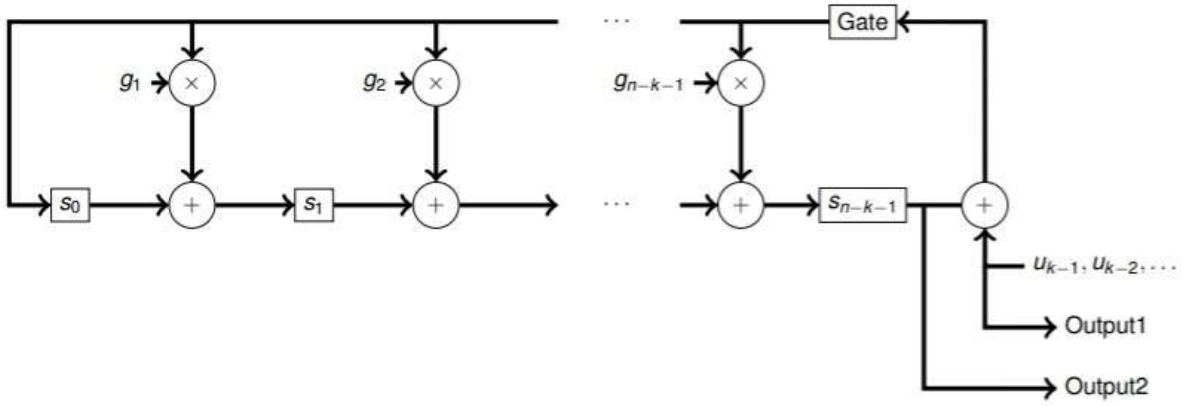
A systematic encoding of the message can be obtained as follows

Divide $X^{n-k}u(X)$ by g(X) to obtain remainder b(X)

The code polynomial is given by $b(X) + X^{n-k}u(X)$

Systematic Encoding Circuit for Cyclic Codes

Let $g(X) = 1 + g_1X + g_2X^2 + \cdot \cdot \cdot + g_{n-k-1}X^{n-k-1} + X^{n-k}$



Turn on the gate. Shift the message bits $u_{k-1}, u_{k-2}, \ldots, u_0$ into the circuit and channel simultaneously. Only Output1 is fed to the channel.

Turn off the gate and shift the contents of the shift register into the channel. Only Output2 is fed to the channel.

Syndrome Computation

Errors are detected when the received vector is not a codeword

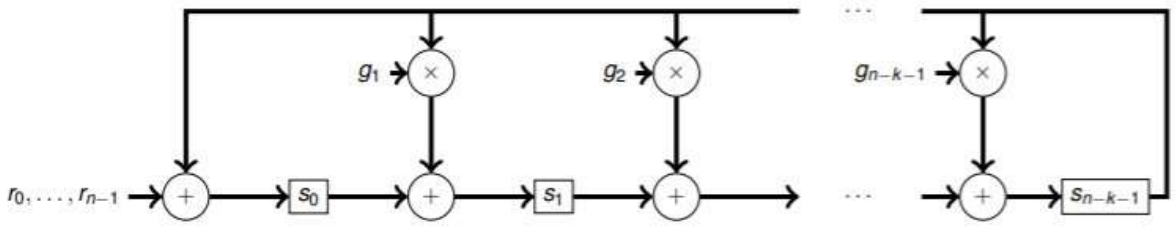
For linear block codes, r is a codeword $\Longleftrightarrow rH^T = 0$

$s = rH^T$ is called the syndrome vector

For cyclic codes, the received polynomial r(X) is a code polynomial $\Longleftrightarrow$ r(X) mod g(X) = 0

s(X) = r(X) mod g(X) is called the syndrome polynomial

The following circuit computes the syndrome polynomial



Golay Codes

The condition for a code to be perfect is that its n, M and d values satisfy the sphere - packing bound

$$M \leq \frac{q^n}{\sum_{k=0}^{t} \binom{n}{k} (q-1)^k}$$

with d = 2t + 1. Golay found three other possible integer triples (n, M, d) that do not correspond to the parameters of a Hamming or trivial perfect code. They are (23, 2^{12}, 7) and (90, 2^{78}, 5) for q = 2 and (11, 3^{6}, 5) for q = 3.

MODULE 5 BCH CODES:

SET:

The sample statement that a set is a collection of elements. A set with in objects is written as $\{a_1, a_2, a_3, a_4, \ldots, a_n\}$ the objects a_1, a_2, a_3 are called as the set elements. If the number of elements are finite then they are called as finite set. Eg. A={1,2,3,5}
If the number of elements are infinite then they are called as infinite set

Eg. A={±1, ±2, ±3, ±5}

GROUP

A group is constructed from a set by defining a group operation * between the elements such that the following conditions are satisfied

- The group is closed under the operation * that for any two elements a and b within the set then the element c= a*b is also an element of the set.
- The operation * is associative, so that given the elements a*(b*c) = (a*b)*c

- An identity element is present in the set. a * I = I * a = a
- For every element there exists a unique inverse a' n the set such that a*a' = a'*a = I

if the group has the property that the two elements of it

$$a*b = b*a$$

then the commutative group or multiplicative group

Additive Group:

Construct an additive group of integers modulo -5 over the set {0,1,2,3,4} We need to get the remainders when we obtain the integers pair wise.

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

Multiplicative Group:

Multiplicative group can be constructed using the modulo m multiplication, it is the remainder of the product of the two integers.

X	1	2	3	4
1	1	2	3	4
2	2	4	1	3
3	3	1	4	2
4	4	3	2	1

Field

A group is extended to a field. A field is an algebraic system consisting of a set, an identity element for each operation, two operations and their respective inverse operations.
A example field, F = (A, O1, O2, I1, I2)

A is set of

INFORMATION THEORY & CODING. EC602

O1 is the operation of addition, the inverse operation is subtraction
O2 is the operation of multiplication
I1 is the identity element zero (0)
I2 is the identity element one (1)

Design a modulo 7 prime field with multiplication and addition as field operators

+	0	1	2	3	4	5	6	X	1	2	3	4	5	6
0	0	1	2	3	4	5	6	1	1	2	3	4	5	6
1	1	2	3	4	5	6	0	2	2	4	6	1	3	5
2	2	3	4	5	6	0	1	3	3	6	2	5	1	4
3	3	4	5	6	0	1	2	4	4	1	5	2	6	3
4	4	5	6	0	1	2	3	5	5	3	1	6	4	2
5	5	6	0	1	2	3	4	6	6	5	4	3	2	1
6	6	0	1	2	3	4	5							

Galois Field

GF(p) for any prime, p, this Galois Field has p elements which are the residue classes of integers modulo p m

GF(p) for any prime, p, and m greater than zero, this Galois Field m has p elements which is a Field of polynomials over GF(p) modulo an irreducible polynomial of degree m.

GF(q) for q = p for prime, p, and m greater than zero, this Galois Field has q elements of the vector space of dimension m over GF(p).

If p is a prime number, then it is also possible to define a field with pm elements for any m . These fields are named for the great French algebraist Evari ste Galois . They have many applications in coding theory. The fields, denoted GF(pm), are comprised of the polynomials of degree m − 1 over the field p
. These polynomials are expressed as am − 1 xm−1 + ... + a1 x1+a0x0 where the coefficients a I take on values in the set {0, 1, ..., p−1}.
When employed in coding applications p is commonly 2 and thus the coefficients {a0 , ..., am−1} are taken from the binary digits {0,1} . In coding applications, for m ≤ 32, it is common to represent an entire polynomial in GF(2m) as a single integer value in which individual bits of the integer repressent the coefficients of the polynomial. The least significant bit of the integer represents the a0 coefficient.

Galois Field(2) GF(2)

The two elements are nearly always called 0 and 1, being the additive and multiplicative identities, respectively.

The field's addition operation is given by the table below, which corresponds to the logical XOR operation.

+	**0**	**1**
0	0	1
1	1	0

The field's multiplication operation corresponds to the logical AND operation.

×	**0**	**1**
0	0	0
1	0	1

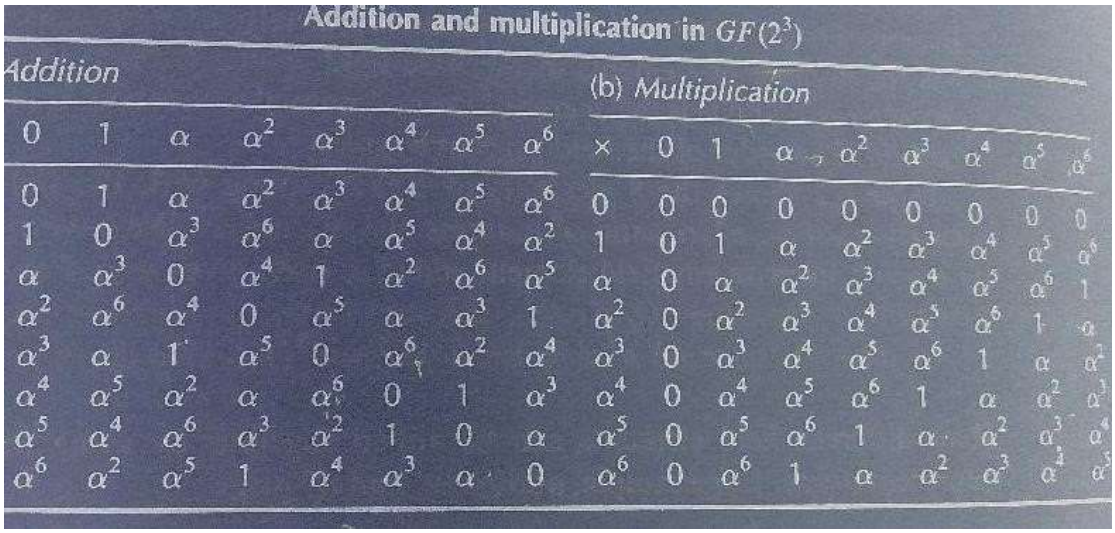
GF(2^m)

The elements of GF(2^m) are binary polynomials, i.e. polynomials whose coefficients are either 0 or 1. There are 2^m such polynomials in the field and the degree of each polynomial is no more than *m*-1. Therefore the elements can be represented as *m*-bit strings. Each bit in the bit string corresponding to the coefficient in the polynomial at the same position. For example, $GF(2^3)$ contains 8 element {0, 1, α, α+1, α^2, α^2+1, α^2+α, α^2+α+1}. α+1 is actually $0\alpha^2+1\alpha+1$, so it can be represented as a bit string 011. Similarly, $\alpha^2+\alpha = 1\alpha^2+1\alpha+0$, so it can be represented as 110.

Field elements of GF (2^3)
0 1 α α^2 α_3
= α+1 α_4 =
α_2+ α α^5 =
α^2+α+1
α^6 = α^2+1

Addition and multiplication field of Galois field,

INFORMATION THEORY & CODING. EC602



Addition and multiplication in $GF(2^3)$

Addition

	0	1	α	α^2	α^3	α^4	α^5	α^6
0	0	1	α	α^2	α^3	α^4	α^5	α^6
1	1	0	α^3	α^6	α	α^5	α^4	α^2
α	α	α^3	0	α^4	1	α^2	α^6	α^5
α^2	α^2	α^6	α^4	0	α^5	α	α^3	1
α^3	α^3	α	1	α^5	0	α^6	α^2	α^4
α^4	α^4	α^5	α^2	α	α^6	0	1	α^3
α^5	α^5	α^4	α^6	α^3	α^2	1	0	α
α^6	α^6	α^2	α^5	1	α^4	α^3	α	0

(b) Multiplication

$\times$	0	1	α	α^2	α^3	α^4	α^5	α^6
0	0	0	0	0	0	0	0	0
1	0	1	α	α^2	α^3	α^4	α^5	α^6
α	0	α	α^2	α^3	α^4	α^5	α^6	1
α^2	0	α^2	α^3	α^4	α^5	α^6	1	α
α^3	0	α^3	α^4	α^5	α^6	1	α	α^2
α^4	0	α^4	α^5	α^6	1	α	α^2	α^3
α^5	0	α^5	α^6	1	α	α^2	α^3	α^4
α^6	0	α^6	1	α	α^2	α^3	α^4	α^5

Irreducible and primitive polynomials

A polynomial is said to be irreducible if it cannot be factored into nontrivial polynomials over the same field.

Eg. In the finite field GF(2): is irreducible is not irreducible, since

$x^2 + x(x+1) = x^2 + 2x + 1 \equiv x^2 + 1 \pmod 2$.

$x^2 + 1$

The number of irreducible polynomials of degree $n$over GF(2) is

	irreducible polynomials
1	$1 + x$, x
2	$1 + x + x^2$
3	$1 + x + x^3$, $1 + x^2 + x^3$
4	$1 + x + x^4$, $1 + x + x^2 + x^3 + x^4$, $1 + x^3 + x^4$
5	$1 + x^2 + x^5$, $1 + x + x^2 + x^3 + x^5$, $1 + x^3 + x^5$, $1 + x + x^3 + x^4 + x^5$, $1 + x^2 + x^3 + x^4 + x^5$, $1 + x + x^2 + x^4 + x^5$

A primitive polynomial is a polynomial that generates all elements of an extension field from a base field. Primitive polynomials are also irreducible polynomials. For any prime or prime power q and any positive integer n, there exists a primitive polynomial of degree $n$over GF(q). A primitive polynomial is a special case of irreducible polynomial.

To determine whether a polynomial is primitive or not :
Let $p(x) = x^4 + x^3 + x^2 + x + 1$
Determine $2^m-1 = 15$

If p(x) divides $x^{15} + 1$ then it is irreducible. But to state that it is primitive we need to determine whether it divides into $x^{14} + 1$, $x^{13}+1$ ….or x^5+1. if it divides into any of these polynomial then it is not a primitive polynomial.

Here in this example p(x) is not a primitive polynomial.

Example: Determine that the following polynomial are primitive or not.

a) $x^2 + x + 1$
b) $x^3 + x + 1$

Computations with Polynomials

Mathematcal computations over polynomials whose coefficients are from the binary field GF(2). Let us consider a polynomial f(X) with variable X and with coefficients from GF(2) is of the following form:

$$f(X) = f_0 + f_1 X + f_2 X^2 + \cdots + f_n X^n,$$

$$\text{where } f_i = 0 \text{ or } 1 \text{ for } 0 \leq i \leq n.$$

The degree of a polynomial is the largest power of X with a nonzero coefficient. For the polynomial above, if f n = l ,f(X) is a polynomial of degree n; if f n = 0, ,f(X) is a polynomial of degree less than n. The degree of f(X) = f0 is zero.

Addition :

Let us consider two polynomials x+1 and x^2+1. Addition of these will be :

x+1
$\underline{x^2+1}$ $\underline{x^2+}$
$\underline{x}$

Division

Divide $f(X) = 1 + X + X^x + X^s + X^6$ by $g(X) = 1 + X + X^3$?

Sol:

$$
\begin{array}{r}
X^3 + X^2 \text{ (quotient)} \\
\hline
X^3 + X + 1 \;)\; X^6 + X^5 + X^4 \qquad\quad + X + 1 \\
\underline{X^6 \qquad + X^4 + X^3} \\
X^5 \qquad + X^3 \qquad + X + 1 \\
\underline{X^5 \qquad + X^3 + X^2} \\
X^2 + X + 1 \text{ (remainder).}
\end{array}
$$

We can easily verify that

$$X^6 + X^5 + X^4 + X + 1 = (X^3 + X^2)(X^3 + X + 1) + X^2 + X + 1.$$

BCH codes

The Bose, Chaudhuri, and Hocquenghem (BCH) codes form a large class of powerful random errorcorrecting cyclic codes. This class of codes is a remarkable generalization of the Hamming code for multiple-error correction.

BCH codes are a class of linear cyclic codes with an arbitrary set of linear block codes as its roots. For a cyclic code any codeword polynomial has its generator polynomial as a factor; so the roots of the code's generator polynomial g(x) are also the roots of codewords. BCH codes are constructed using the roots of g(x) in extended Galois field; binary primitive BCH codes – which are multiple random error correcting in nature – form an important sub class. A t error correcting binary BCH code can be generated by the following parameters

Block length $n = 2^m - 1$
No. of parity check bits: $n - K \leq mt$ Minimum
distance: $d_{min} \geq 2t + 1$

Procedure :

A t error correcting cyclic code wth generator polynomial g(x) is a binary BCH code if and only if g(x) is the least degree polynomial over GF(2) that has roots $\beta, \beta^2, \beta^3, \ldots, \beta^{2t}$ where β is the element of g(x).

$g(x) = LCM [m_1(x), m_2(x), m_3(x), \ldots, m_{2t}(x)]$ where $m_1(x)$, $m_2(x)$, … are the minimal polynomial of the $\alpha, \alpha^2, \alpha^3$.

INFORMATION THEORY & CODING. EC602

Decoding of BCH codes The decoding of BCH codes involves the following steps:

(i) Form the syndrome polynomial

(ii) s(x) = s0+ s1x + s2x2+ ·· ·+sn-K-1x n-K-1 with the set { s0,s1,s2.... s-K-1} being the values of r(x) at α, α2,.... α2t.

(iii) If s(x) is zero, r(x) itself is a codeword; else proceed as follows. (ii)With the syndromes obtained in step 1 above, form the error-locator polynomial σ(x) using any of algorithms like Peterson Gorenstein Zierler algorithm

(iv) obtain the roots of σ(x) and their respective inverses which indicate he rror locations.

(v) Complement the bits in the positions indicated by the error locations to obtain the decoded codeword. It may be noted here that alternately the syndrome polynomial can be obtained by dividing r(x) by g(x) and evaluating the remainder at α, α^2,.... α^{2t}. This is same as the syndrome.

Sample Questions

1. What are primitive polynomials? Verify whether $p(x) = x^4+x+1$ is a primitive polynomial

2. Verify whether the codes are cyclic or not {0000, 0110, 1100, 0011, 1001}

3. Determine the conjugates of $\square^3$ in GF(2^3) and GF(2^4).

4. The generator polynomial of a cyclic code is a factor of

   a) X^n+1 b) $X^{(n+1)}+1$ c) $X^{(n+2)}+1$ d) none of these

5. Explain BCH codes and its syndrome detection technique. Design a triple error correcting BCH code over GF(2^4).

INFORMATION THEORY & CODING. EC602

MODULE 6

Convolutional codes

A **Convolutional code** is a type of error-correcting code that generates parity symbols via the sliding application of a boolean polynomial function to a data stream. The sliding application represents the 'convolution' of the encoder over the data, which gives rise to the term 'convolutional coding.' The sliding nature of the convolutional codes facilitates trellis decoding using a time-invariant trellis. Time invariant trellis decoding allows convolutional codes to be maximum-likelihood soft-decision decoded with reasonable complexity.

The ability to perform economical maximum likelihood soft decision decoding is one of the major benefits of convolutional codes. This is in contrast to classic block codes, which are generally represented by a timevariant trellis and therefore are typically hard-decision decoded.

Convolutional codes are often characterized by the base code rate and the depth (or memory) of the encoder [n,k,K].

Convolutional codes are commonly specified by three parameters; (n,k,m).

n = number of output bits k = number of

input bits m = number of memory registers

The base code rate is typically given as n/k, where n is the input data rate and k is the output symbol rate.

The depth is often called the "constraint length" 'K', where the output is a function of the current input as well as the previous K-1 inputs.

The depth may also be given as the number of memory elements 'v' in the polynomial or the maximum possible number of states of the encoder (typically 2^v). Constraint Length, K = k (m-1)

Convolutional codes are often described as continuous. However, it may also be said that convolutional codes have arbitrary block length, rather than being continuous, since most real-world convolutional encoding is performed on blocks of data. Convolutional encoded block codes typically employ termination. The arbitrary block length of convolutional codes can also be contrasted to classic block codes, which generally have fixed block lengths that are determined by algebraic properties.

The code rate of a convolutional code is commonly modified via symbol puncturing. For example, a convolutional code with a 'mother' code rate n/k=1/2 may be punctured to a higher rate of, for example, 7/8 simply by not transmitting a portion of code symbols. The performance of a punctured convolutional code generally scales well with the amount of parity transmitted. The ability to perform economical soft decision

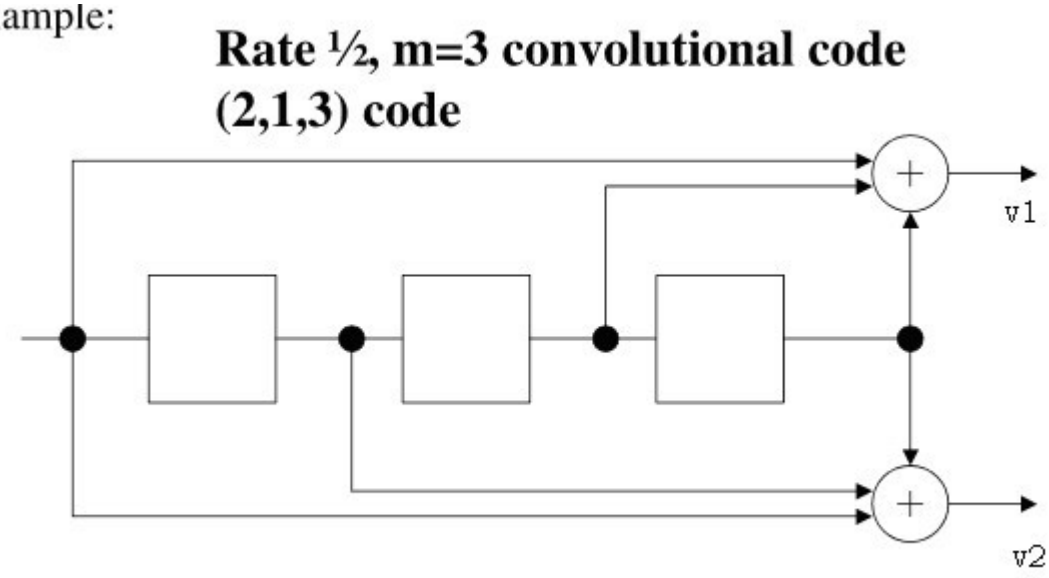
decoding on convolutional codes, as well as the block length and code rate flexibility of convolutional codes, makes them very popular for digital communications

GENERATION OF CONVOLUTION CODE:

A convolutional code is generated by passing the information sequence to be transmitted through a linear finite-state shift register. In general, the shift register consists of K(k-bit) stages and n linear algebraic function generators

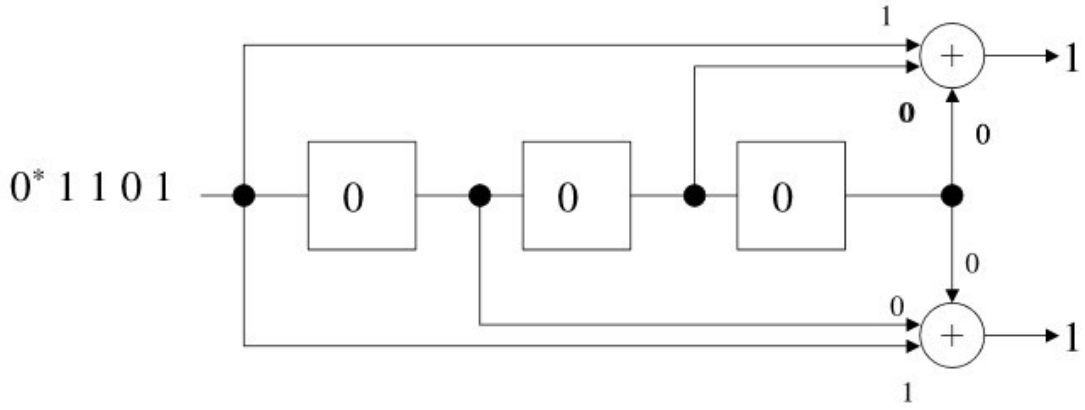
Let us consider a binary convolutional encoder with constraint length K=3, k=1, and n =2. The generators are: g1 =[1011], g2 =[1101].

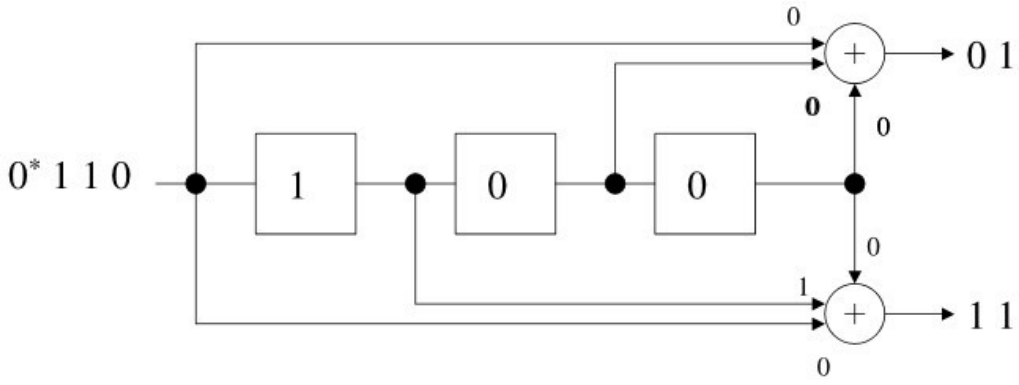
Example:

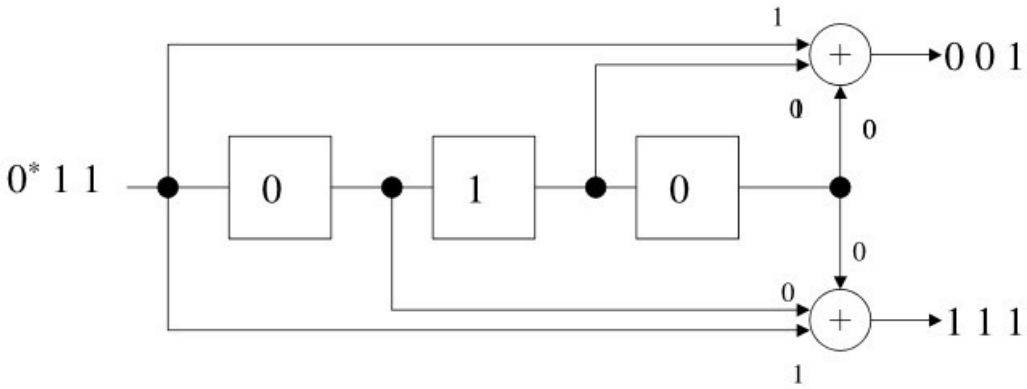


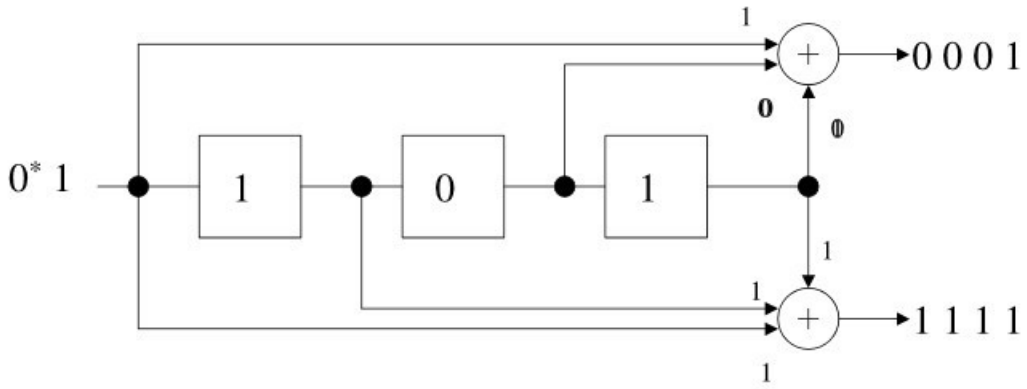
Let us consider the binary sequence be 101 10
One bit enters the circuit at a time and the encoded output are considered as v1, v2

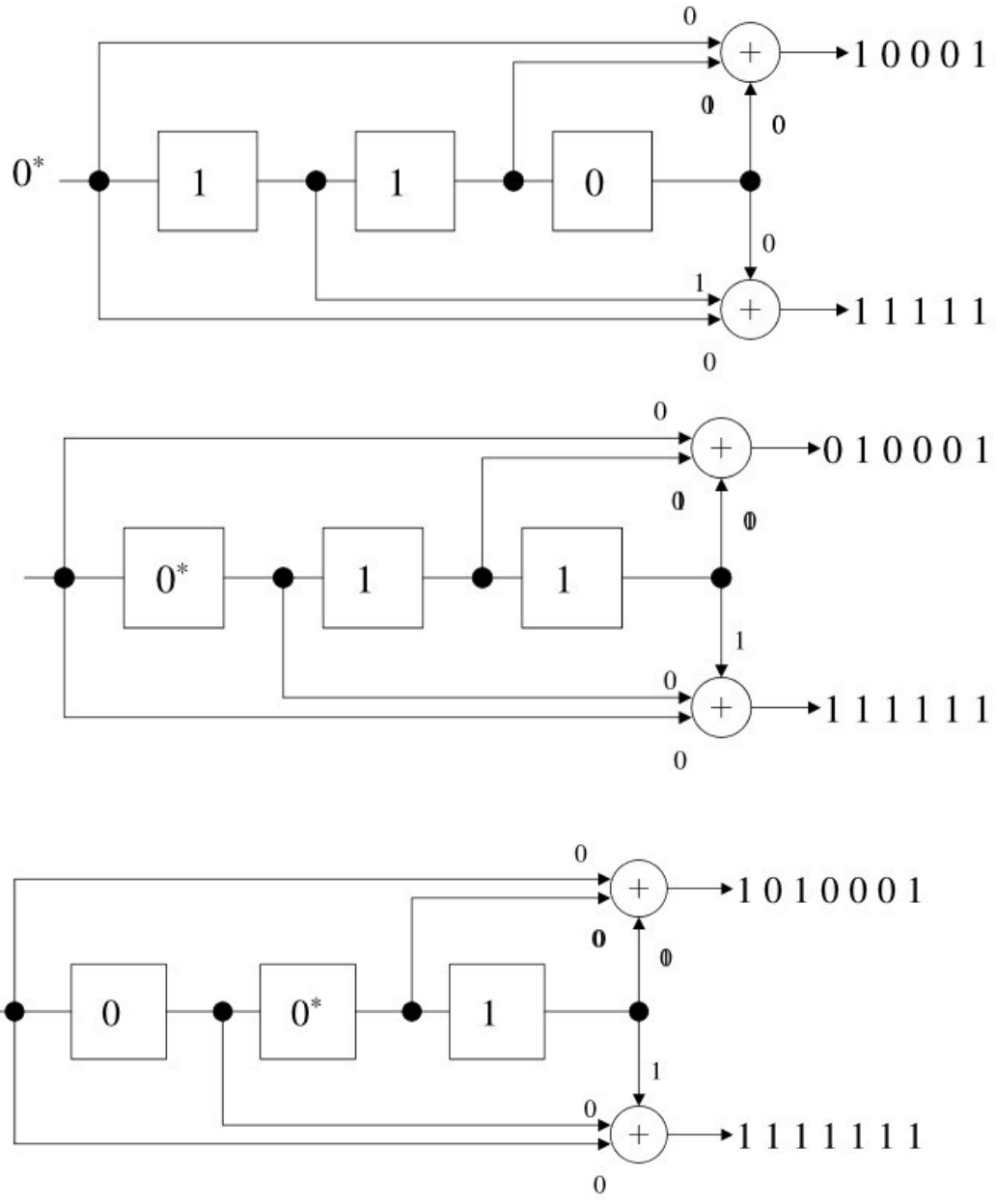
Input: 10110


1
1
0
0
0* 1 1 0 1
0
0
0
0
0
1
1

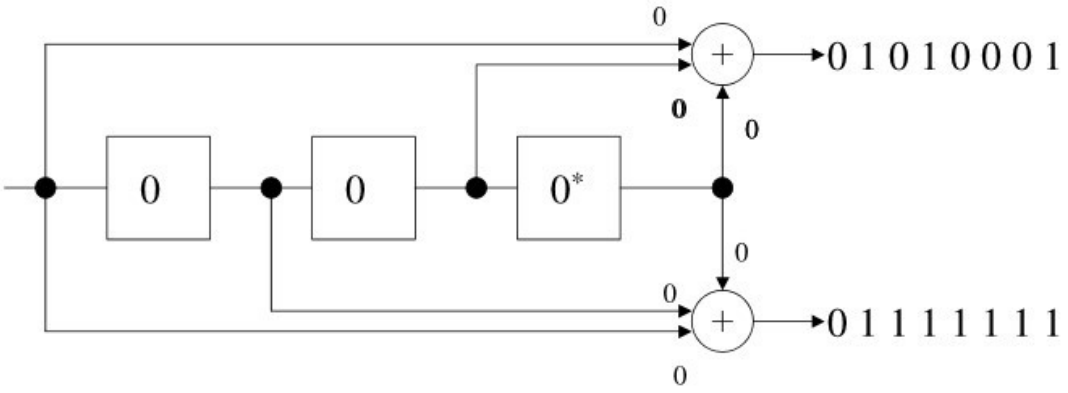

0
0 1
0
0
0* 1 1 0
1
0
0
0
1
1 1
0


1
0 0 1
0
0
0* 1 1
0
1
0
0
0
1 1 1
1


1
0 0 0 1
0
0
0* 1
1
0
1
1
1
1 1 1 1
1


0
1 0 0 0 1
0
0
0*
1
1
0
0
1
1 1 1 1 1
0
0
0 1 0 0 0 1
0
0
0*
1
1
1
0
1 1 1 1 1 1
0
0
1 0 1 0 0 0 1
0
0
0
0*
1
1
0
1 1 1 1 1 1 1
0
0

INFORMATION THEORY & CODING. EC602



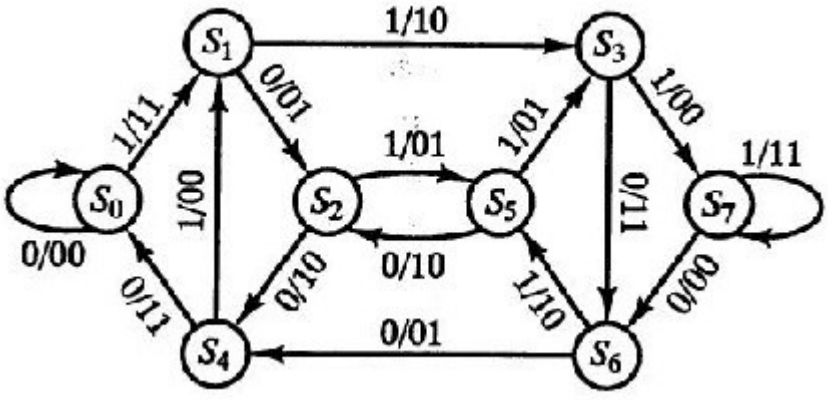
Input: 10110
Output: 11, 01, 01, 01, 11, 01, 11, 00

There are three alternative methods that are often used to describe a convolutional code:

Tree diagram
Trellis diagram
State disgram

State Diagram:

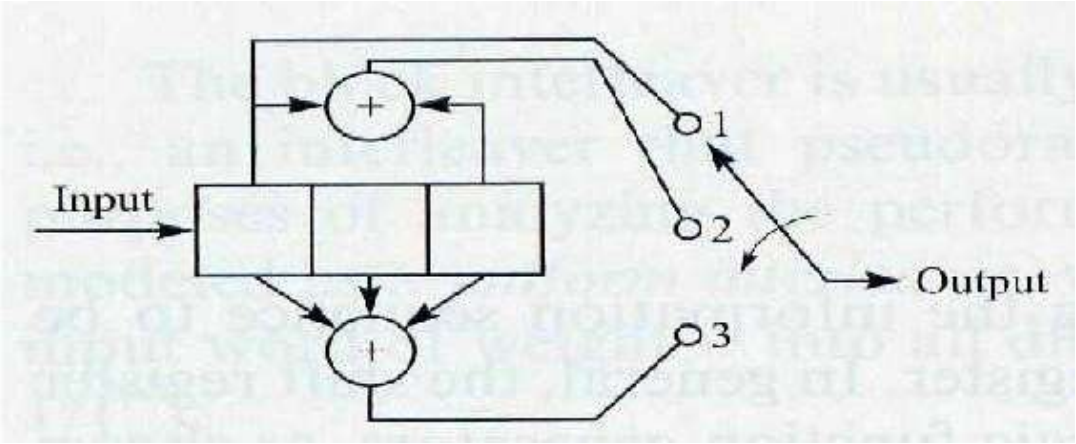
The encoder can be viewed as a finite state machine, for which we can draw a state diagram with transition labels
For the above circuit diagram the state diagram is as below



Where S0, S1, S2, S3, S4, S5, S6, S7 are the possible state of the shift registers.

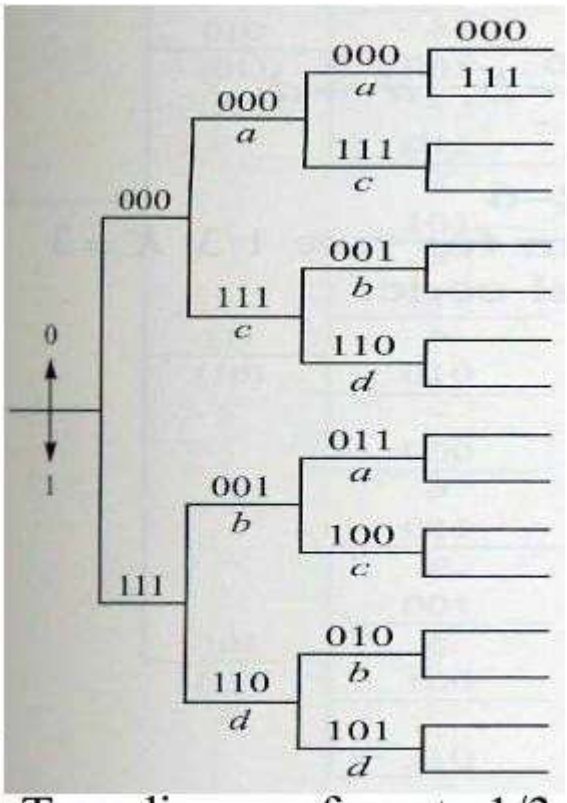
TRELLIS DIAGRAM

The tree diagram for the convolto encoder below



Tree Diagram

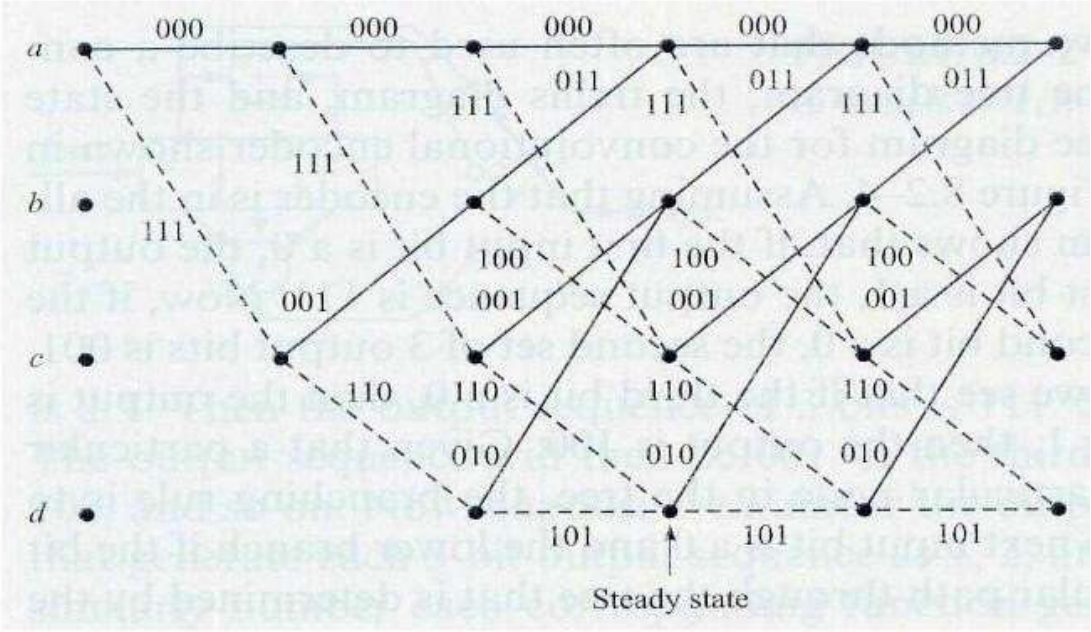
Note that the tree diagram in the right repeats itself after the third stage. This is consistent with the fact that the constraint length K=3. The output sequence at each stage is determined by the input bit and the two previous input bits. In other words, we may sat that the 3-bit output sequence for each input bit is determined by the input bit and the four possible states of the shift register, denoted as a=00, b=01, c=10, and d=11.



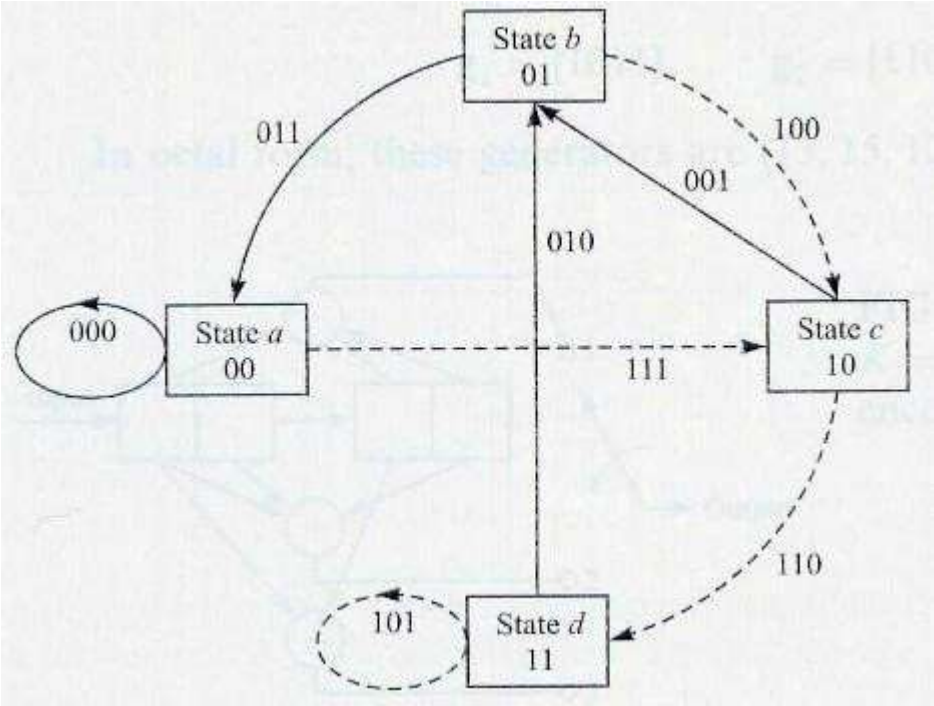
Tree diagram for rate 1/3, K=3 convolutional code.

Tree diagram for rate 1/3, K=3 convolutional code.

Trellis diagram

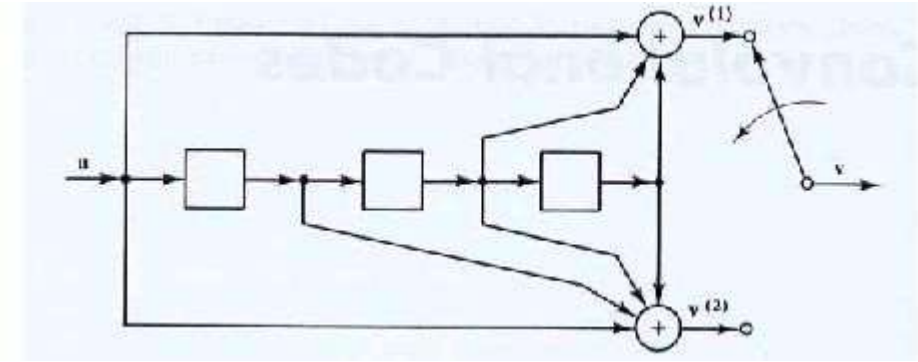


State Diagram



Encoding of Convolutional Codes

Example: A (2, 1, 3) binary convolutional codes:



the encoder consists of an m= 3-stage shift register together with n=2 modulo-2 adders and a multiplexer for serializing the encoder outputs. The mod-2 adders can be implemented as EXCLUSIVE-OR gates. Since mod-2 addition is a linear operation, the encoder is a linear feedforward shift register.

The information sequence u =(u , u , u , …) enters the encoder one bit at a time. 0 1 2 Since the encoder is a linear system, the two encoder output = be obtained as the convolution of the input sequence u with the two encoder "impulse response.he impulse responses are obtained by letting u =(1 0 0 …) and observing the two output sequence. Since the encoder has an m-time unit memory, the impulse responses can last at most m+1 time units, and are written as :

$g^{(1)} = (g_{01} , g_{11} , g_{21} , g_{31} , g_{41})$ $g^{(2)}$

$= (g_{02} , g_{12} , g_{22} , g_{32} , g_{42})$

The encoder of the binary (2, 1, 3) code is

The impulse response g sequences of the code $g^{(1)}$ = (1011)

$g^{(2)}$ = (1111) are called the generator sequence of the code. The encoding equations can now be written as $v^{(1)} = u * g^{(1)}$ $v^{(2)} = u * g^{(2)}$ where * denotes discrete convolution and all operations are mod-2.

The code word v = ($\upsilon_{0(1)}$, $\upsilon_{0(2)}$, $\upsilon_{1(1)}$, $\upsilon_{1(2)}$, $\upsilon_{2(1)}$, $\upsilon_{2(2)}$ …..)

Generator Matrix

If the generator sequence $g^{(1)}$ and $g^{(2)}$ arranged in the matrix and g are interlaced and then where the blank areas are all zeros, the encoding equations can be rewritten in matrix form as v = uG.

$$\mathbf{G} = \begin{bmatrix} g_0^{(1)} g_0^{(2)} & g_1^{(1)} g_1^{(2)} & g_2^{(1)} g_2^{(2)} & \cdots & g_m^{(1)} g_m^{(2)} & & \\ & g_0^{(1)} g_0^{(2)} & g_1^{(1)} g_1^{(2)} & \cdots & g_{m-1}^{(1)} g_{m-1}^{(2)} & g_m^{(1)} g_m^{(2)} & \\ & & g_0^{(1)} g_0^{(2)} & \cdots & g_{m-2}^{(1)} g_{m-2}^{(2)} & g_{m-1}^{(1)} g_{m-1}^{(2)} & g_m^{(1)} g_m^{(2)} \\ & & & \ddots & & & \ddots \end{bmatrix}$$

G is called the generator matrix of the code. Note that each row of G is identical to the preceding row but shifted n = 2 places to right,

If u has finite length L, then G has L rows and 2(m+L) columns, and v has length 2(m + L).

Example:

- If $\mathbf{u}^{(1)} = (1\ 0\ 1)$ and $\mathbf{u}^{(2)} = (1\ 1\ 0)$, then $\mathbf{u} = (1\ 1, 0\ 1, 1\ 0)$ and $\mathbf{v} = \mathbf{uG}$

$$= (1\ 1, 0\ 1, 1\ 0)\begin{bmatrix} 1\ 0\ 1 & 1\ 1\ 1 & & \\ 0\ 1\ 1 & 1\ 0\ 0 & & \\ & 1\ 0\ 1 & 1\ 1\ 1 & \\ & 0\ 1\ 1 & 1\ 0\ 0 & \\ & & 1\ 0\ 1 & 1\ 1\ 1 \\ & & 0\ 1\ 1 & 1\ 0\ 0 \end{bmatrix}$$

$$= (1\ 1\ 0,\ \ 0\ 0\ 0,\ \ 0\ 0\ 1,\ \ 1\ 1\ 1),$$

it agree with our previous calculation using discrete convolution.

Decoding There are several different approaches to decoding of convolutional codes.

These are grouped in two basic categories.

1.Sequential Decoding - Fano algorithm

2. Maximum likely-hood decoding - Viterbi decoding

Both of these methods represent 2 different approaches to the same basic idea behind decoding. The basic idea behind decoding Assume that 3 bits were sent via a rate ½ code. We receive 6 bits. (Ignore flush bits for now.) These six bits may or may not have errors. We know from the encoding process that these bits map uniquely. So a 3 bit sequence will have a unique 6 bit output. But due to errors, we can receive any and all possible combinations of the 6 bits. The permutation of 3 input bits results in eight possible input sequences. Each of these has a unique mapping to a six bit output sequence by the code. These form the set of permissible sequences and the decoder's task is to determine which one was sent.

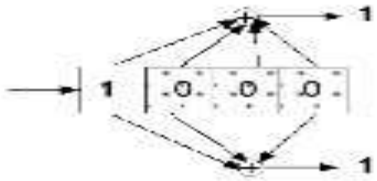
Viterbi decoding is the best known implementation of the maximum likely-hood decoding. Here we narrow the options systematically at each time tick. The principal used to reduce the choices is this. The errors occur infrequently. The probability of error is small. The probability of two errors in a row is much smaller than a single error, that is the errors are distributed randomly.

The Viterbi decoder examines an entire received sequence of a given length. The decoder computes a metric for each path and makes a decision based on this metric. All paths are followed until two paths converge on one node. Then the path with the higher metric is kept and the one with lower metric is discarded. The paths selected are called the survivors. For an N bit sequence, total numbers of possible received sequences are 2 . Of these only 2 valid. The Viterbi algorithm applies the maximum-likelihood principles to limit the comparison to 2 of the power of kL surviving paths instead of checking all paths.

The most common metric used is the Hamming distance metric. This is just the dot product between the received codeword and the allowable codeword.

INFORMATION THEORY & CODING. EC602

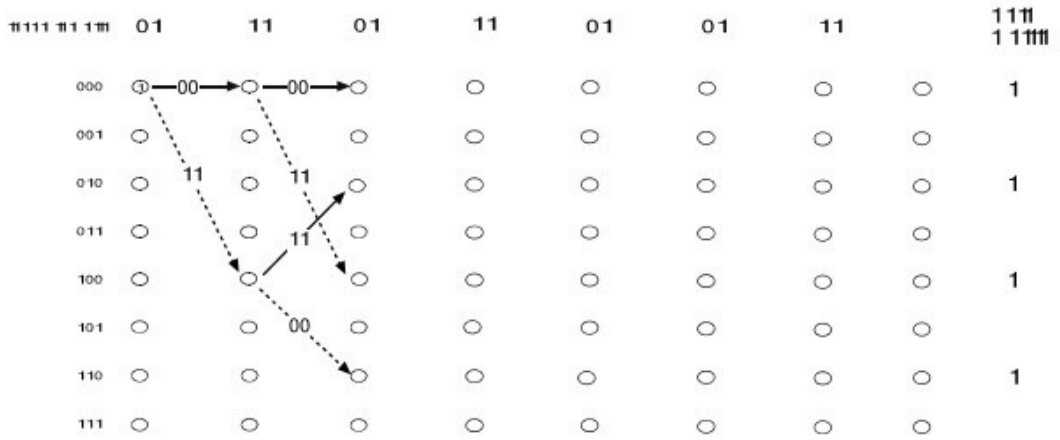
For an encoder given, the decoding process of received sequence using Viterbi decoding.



Let's decode the received sequence 01 11 01 11 01 01 11 using Viterbi decoding.

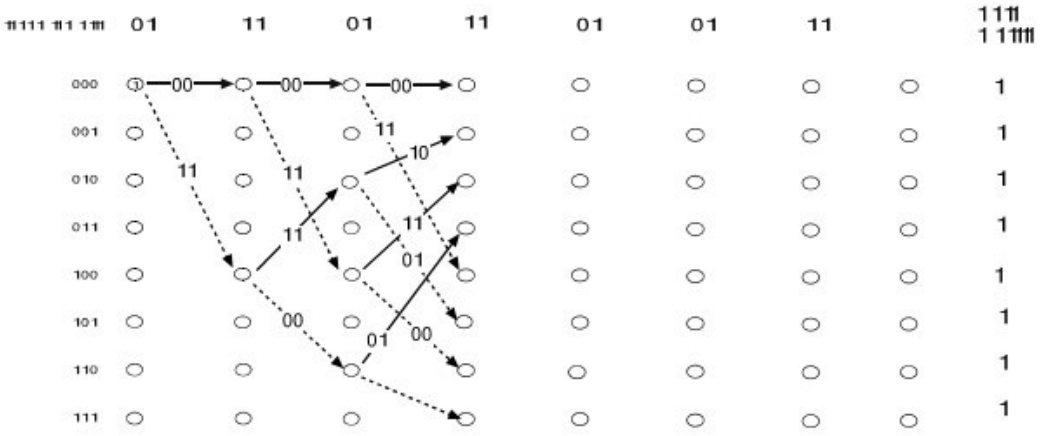
Viterbi Decoding, Step 1

2. At t = 1, the decoder fans out from these two possible states to four states. The branch metrics for these branches are computed by looking at the agreement with the codeword and the incoming bits, which are 11. The new metric is shown on the right of the trellis.



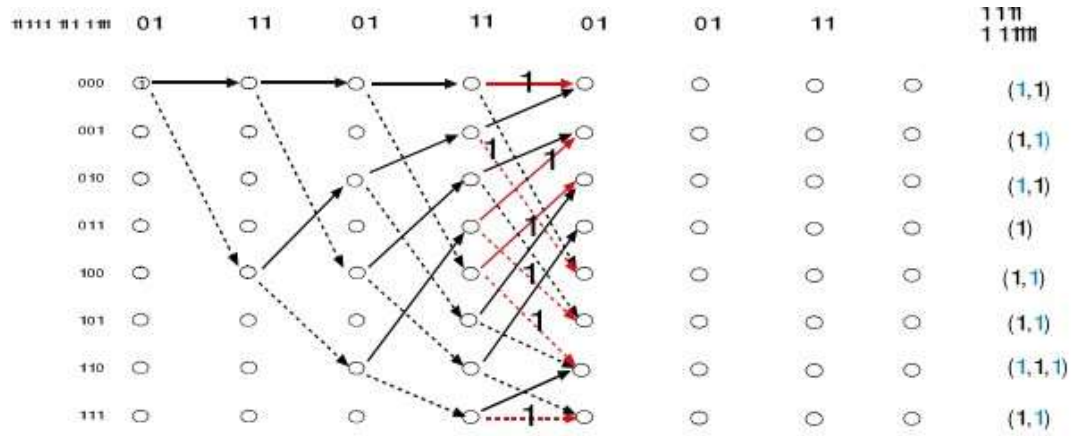
Viterbi Decoding, Step 2

At t = 2, the four states have fanned out to eight to show all possible paths. The path metrics calculated for bits 01 and added to pervious metrics from t = 1.

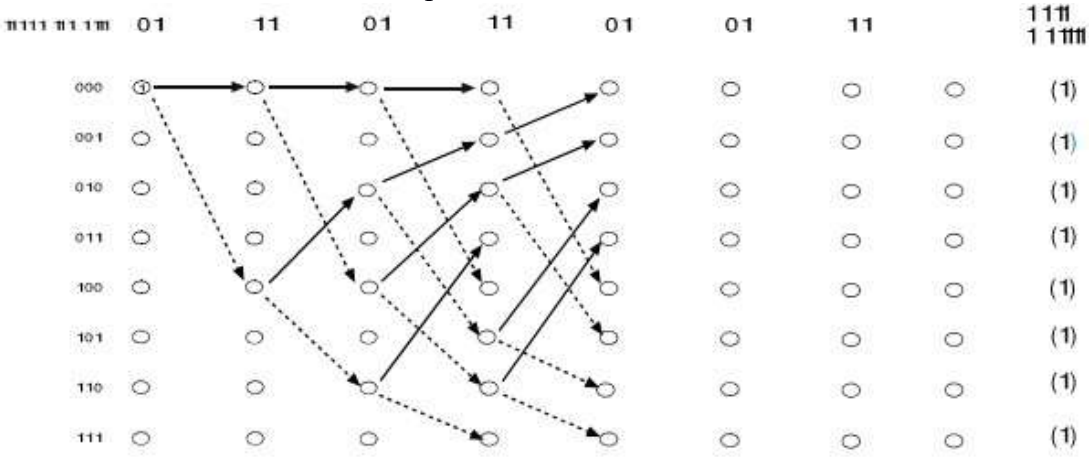


Step 3

At t = 4, the trellis is fully populated. Each node has at least one path coming into it. The metrics are as shown in the figure above. At t = 5, the paths progress forward and now begin to converge on the nodes. Two metrics are given for each of the paths coming into a node. Per the Maximum likelihood principle, at each node we discard the path with the lower metric because it is least likely. This discarding of paths at each node helps to reduce the number of paths that have to be examined and gives the Viterbi method its strength.



Now at each node, we have one or more path converging. The metrics for all paths are given on the right. At each node, we keep only the path with the highest metric and discard all others, marked with an X (shown in red). After discarding the paths with the smaller metric, we have the following paths left. The metric shown is that of the winner path.



Fg: Step 4 after discarding

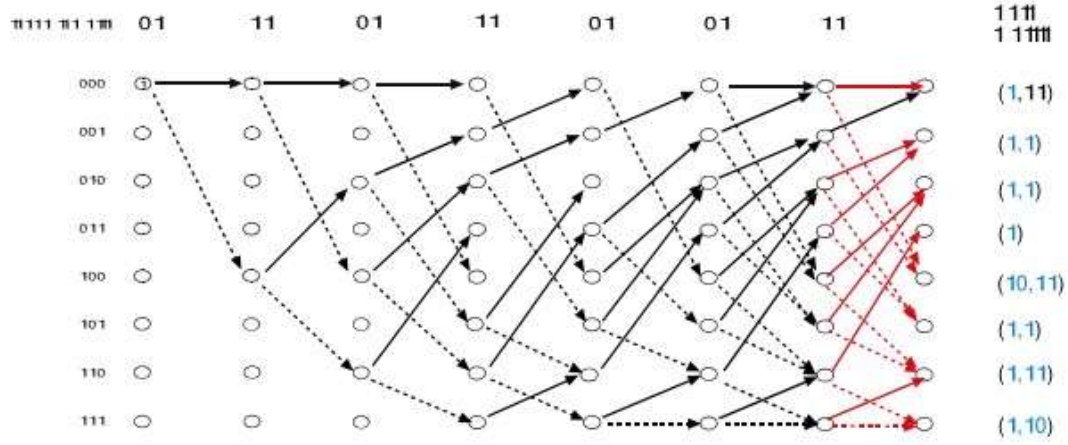


Fig. Complete viterbi diagram

INFORMATION THEORY & CODING. EC602

The trellis is complete. We now look at the path with the highest metric. The path traced by states 000, 100, 010, 101, 110, 011, 001, 000 and corresponding to bits 1011000 is the decoded sequence.

Sample Questions

1. Define constraint length in convolutional codes.
2. Verify whether the codes are cyclic or not {0000, 0110, 1100, 0011, 1001}
3. For a (3,1,2) convolution code with $g^{(1)} = (1\ 0\ 1)$ and $g^{(2)} = (1\ 1\ 1)$ $g^{(3)} = (1\ 0\ 0)$
   a) Determine the code word for message u = (1001)
   b) Give the hardware realization of the encoder.
   c) Give the state diagram for the encoder
   d) Using viterbi decoding technique decode the received code word r = 101010010000

BOOKS:

1. **Information theory, coding and cryptography - Ranjan Bose; TMH.**
2. **Introduction to Error Control Codes - Salvatore Gravano, Oxford**

REFERENCE BOOKS:

1. **Information and Coding - N Abramson; McGraw Hill.**
2. **Introduction to Information Theory - M Mansurpur; McGraw Hill.**
3. **Information Theory - R B Ash; Prentice Hall.**
4. **Error Control Coding - Shu Lin and D J Costello Jr; Prentice Hall.**
5. **Todd K Moon,- Error Correction Coding: Mathematical Methods and Algorithms, John Wiley & Sons**