Mathematics 9
Course assignment
Information theory, source coding and channel coding theorem

Team no. 11

Ahmed salah sheweita 27
Mohammed Ahmed Ali Helal 152
Mohammed wagdy nomeir 184
Marwan nabil mohammed 195
Hisham gaber ahmed 229

Contents

- EECS II lecture notes OCW MIT
- elements of information theory by Thomas and cover
- a first course in probability by Sheldon ross

Introduction - motivation

Probability theory and probabilistic models of random processes lend themselves to major applications in many fields of scientific research, from statistics and finance to artificial intelligence and machine learning to physics thermo dynamics and quantum physics, Of particular interest to the student of communication systems are the ideas from a branch of mathematics related to probability theory: information theory, their usefulness becomes evident for example in the case of a digital signal source, emitting symbols associated with probabilities.

Probability theory provides a basis for the formal treatment of a range of topics and concepts belonging to information theory, The entry point to such a treatment is a definition of the abstract concept of Information, which utilizes the probability concept encountered in the study of probability theory, thus any further employment of the information concept in other abstractions or applications, which is indeed the case for the ideas of information theory discussed here, implies a strong relation between such ideas and the ones from probability theory.

With this in mind, and with the widespread usage of information theory methods and concepts in modern computer and communication systems, we can consider information theory as a branch of applied mathematics. Motivations for such work on information theory include the economic transmission and storage of digital information, both motives have driven great figures in this field to create and develop methods aimed at reducing the resources needed to transmit or store a given amount of data or information 'source coding', another motive is increasing the integrity of

Information transmission over noisy channels

‘Channel coding’, consequently, various methods of data compression and error detection have been devised.

Information theory was developed by Claude E. Shannon in the 1940s,in 1948 he published a paper” a mathematical theory of communication” considered the inception of the theory, Shannon borrowed ideas developed by Ralph Hartley who considered information a measurable quantity and the notion of entropy from thermodynamics which was developed by L. Boltzmann and J. W. Gibbs.

Information and entropy

2.1 definition of information

“Information is the resolution of uncertainty” ~Claude E. Shannon

The definition of information most relevant to a theoretical study is a measurable quantity which is related to the probability of an outcome of a random experiment,

For example: a source transmitting a sequence of symbols, each transmission is a random experiment, the Outcome (S_i) of which corresponds to a particular probability $P(S_i)$, there is a Finite set of possible outcomes (or symbols) each with a specific probability of occurrence, thus we have a complete probability model of the transmission experiment, we define the information

$$I(S = s_i) = \log_2\left(\frac{1}{p_S(s_i)}\right)$$

obtained from an experiment as:

A unit of measurement for information is the “Bit” an acronym for (binary information unit) not to be confused with the binary digit, -although we also use this quantity to specify the minimum number of binary digits required to transmit a symbol.

From the definition it is evident that the more likely a symbol is to occur, the less information it gives us and subsequently the less resources it should take to be transmitted (ideally).

A curious consequence of the axioms of probability on our definition of information is the fact that information of independent outcomes is additive, to put it in a different way” if the sequence of symbols transmitted by a source consists of independent and identically distributed symbols, then the information obtained from the whole sequence is the sum of the information obtained from each symbol alone”, this can be easily proved using the properties of logarithms and probabilities of independent events
(if the events A,B,C are mutually independent then P(ABC) =P(A)* P(B)* P(C)).

2.2 axioms of information theory

Consider an event E that can occur when an experiment is performed. How surprised would we be to hear that E does, in fact, occur? It seems reasonable to suppose that the amount of surprise engendered by the information that E has occurred should depend on the probability of E. For instance, if the experiment consists of rolling a pair of dice, then we would not be too surprised to hear that E has occurred when E represents the event that the sum of the dice is even (and thus has probability 1/2), whereas we would certainly be more surprised to hear that E has occurred when E is the event that the sum of the dice is 12 (and thus has probability 1/36).

In this section, we attempt to quantify the concept of surprise. To begin, let us agree to suppose that the surprise one feels upon learning that an event E has occurred depends only on the probability of E, and let us denote by S(p) the surprise evoked by the occurrence of an event having probability p .

We determine the functional form of S(p) by first agreeing on a set of reasonable conditions that S(p) should satisfy and then proving that these axioms require that S(p) have a specified form.

We assume throughout that S(p) is defined for all $0 < p \leq 1$, but is not defined for events having $p = 0$. Our first condition is just a statement of the intuitive fact that there is no surprise in hearing that an event which is sure to occur has indeed occurred.

Axiom 1: $S(1) = 0$

Our second condition states that the more unlikely an event is to occur, the greater is the surprise evoked by its occurrence.

Axiom 2:

S(p) is a strictly decreasing function of p; that is,

if $p < q$; then $S(p) > S(q)$.

The third condition is a mathematical statement of the fact that we would intuitively expect a small change in p to correspond to a small change in S(p).

Axiom 3:

S(p) is a continuous function of p.

To motivate the final condition, consider two independent events E and F having respective probabilities $P(E) = p$ and $P(F) = q$.

Since $P(EF) = pq$, the surprise evoked by the information that both E and F have occurred is $S(pq)$.

Now, suppose that we are told first that E has occurred and then, afterward, that F has also occurred. Since $S(p)$ is the surprise evoked by the occurrence of E, it follows that $S(pq)$ $S(p)$ represents the additional surprise evoked when we are informed that F has also occurred. However, because F is independent of E, the knowledge that E occurred does not change the probability of F; hence, the additional surprise should just be $S(q)$. This reasoning suggests the final condition.

Axiom 4

$S(pq) = S(p) + S(q)$ *where* $0 < p \leq 1, \quad 0 < q \leq 1$

from the previous axioms we can define s(.) as:

$s(p) = c \, log_a(1/p)$

and we usually put $c = 1$ and $a = 2$.

A unit of measurement for information in this case is the Bit an acronym for (binary information unit) not to be confused with the binary digit we can rewrite the equation above in a more common notation as

$$I(X = x_i) = \log_2 (1/P_X (x_i)).$$

2.3 Entropy

It is the expected value of information obtained from learning the outcome of a random experiment with a known probability model.

An example of such an experiment is a digital signal source emitting a single symbol at a time, the symbol belongs to a finite set of symbols all having associated probabilities.

The formal expression for obtaining this quantity is:

$$H(S) = \sum_{i=1}^{N} p_S(s_i) I(S = s_i)$$

$$= \sum_{i=1}^{N} p_S(s_i) \log_2 \left(\frac{1}{p_S(s_i)} \right)$$

Where N: the number of symbols in a finite set of symbols available to the source.

The use of a probabilistic notion is clear, the notion of expectation (mean),
here it is the weighted sum of the information from each possible symbol.

Significance of Entropy

The idea of entropy becomes important when dealing with efficient binary encoding of symbols.

Binary encoding is the process of assigning binary codes to each symbol in order to be transmitted over a digital communication channel. Efficient here means exploiting the redundancy in the source alphabet (set of possible symbols with varying probabilities) to produce a varying length encoding for each symbol that depends on its associated probability, with the result of reducing the overall binary digits required to transmit a message (a sequence of symbols with specific length).

The degree of optimization of the message length has a lower limit equal to the entropy of the source, various encoding schemes have been devised in an effort to approach that boundary, to relate entropy to encoding efficiency we can look at it this way:

Entropy, measured in bits, tells us the average amount of information required to resolve the uncertainty about an outcome of an experiment, equivalently, it is average number of binary digits required to transmit a single symbol, if a fewer binary digits are transmitted for each symbol on average, there would be uncertainty about that symbol meaning (undetermined).

2.4 Joint entropy

In the previous discussion we viewed the entropy of a single random variable, we now need to extend the definition for multi-random variables which is known as "joint entropy'.

For 2 discrete random variables the joint entropy H(X, Y) is defined by

$$H(X,Y) = \Sigma_{x\in S_X}\Sigma_{y\in S_Y} p_{X,Y}(x,y)\log_2(\frac{1}{p_{X,Y}(x,y)})$$

Another important quantity is conditional entropy of a random variable given another random variable which is defined by

$$\begin{aligned} H(Y|X) &= \Sigma_x p(x) H(Y|X=x) \\ &= -\Sigma_x p(x)\Sigma_y p(y|x)\log_2 p(y|x) \\ &= -\Sigma_x\Sigma_y p(x,y)\log_2 p(y|x) \end{aligned}$$

The final equation in this section relates all the previous results together this equation is called the chain rule of entropy

$$\begin{aligned} H(X,Y) &= H(X) + H(Y|X) \\ &= H(Y) + H(X|Y) \end{aligned}$$

2.5 relative entropy and mutual information

The entropy of a random variable is a measure of the uncertainty of the random variable; it is a measure of the amount of information required on the average to describe the random variable. In this section we introduce two related concepts: relative entropy and mutual information.

The relative entropy is a measure of the distance between two distributions.
The relative entropy D(p||q) is a measure of the inefficiency of assuming that the distribution is q when the true distribution is p. For example, if we knew the true distribution p of the random variable, we could construct a code with average description length H(p). If, instead, we used the code for a distribution q, we would need H(p) + D(p||q) bits on the average to describe the random variable.

Definition The relative entropy or KullbackLeibler distance between two probability mass functions p(x) and q(x) is defined as

$$D(p||q) = \Sigma_{x \in S_X} p(x) \log_2 \frac{p(x)}{q(x)}$$

However, it is not a true distance between distributions since it is not symmetric i.e. D(p||q) ≠D(q||p) in general.

Nonetheless, it is often useful to think of relative entropy as a distance between distributions.

We now introduce mutual information, which is a measure of the amount of information that one random variable contains about another random variable.
It is the reduction in the uncertainty of one random variable due to the knowledge of the other.

Definition Consider two random variables X and Y with a joint probability mass function p(x, y) and marginal probability mass functions p(x) and p(y). The mutual information I (X; Y) is the relative entropy between the joint distribution and the product distribution p(x)p(y):

$$I(X;Y) = \Sigma_x \Sigma_y p(x,y) \log_2 \frac{p(x,y)}{p(x)p(y)}$$

2.6 relationship between entropy and mutual information

We can rewrite the definition of mutual information I (X; Y) as

$$\begin{aligned} I(X;Y) &= \Sigma_x \Sigma_y p(x,y) \log_2(p(x,y)/p(x)p(y)) \\ &= \Sigma_{x,y} p(x,y) \log_2 p(x|y)/p(x) \\ &= H(X) - H(X|Y) \end{aligned}$$

Thus, the mutual information I (X; Y) is the reduction in the uncertainty of X due to the knowledge of Y. By symmetry, it also follows that

$$I(X;Y) = H(Y) - H(Y|X)$$

Thus, X says as much about Y as Y says about X,
since H(X, Y) = H(X) +H(Y |X), we can rewrite the above equation as

Finally, we note that

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

$$I(X;X) = H(X) - H(X|X) = H(X)$$

Thus, the mutual information of a random variable with itself is the entropy of the random variable. This is the reason that entropy is sometimes referred to as self-information.

Applications of information theory in communication systems

3.1 Data compression

In the last decade we have been witnessing a transformation some call it a revolution in the way we communicate, and the process is still under way. This transformation includes the Internet, the development of mobile communications, etc...
Data compression is one of the enabling Technologies for each of these aspects of the multimedia revolution. The advent of digital TV would not be possible without compression. Data compression, which for a long time was the domain of a relatively small group of engineers and scientists, is now ubiquitous.

Compression Techniques
When we speak of a compression technique or compression algorithm, we are actually referring to two algorithms. There is the compression algorithm that takes an input and generates a

representation that requires fewer bits, and there is a reconstruction algorithm that operates on the compressed representation to generate the reconstruction.
Based on the requirements of reconstruction, data compression schemes can be divided into two broad classes: lossless compression schemes and lossy compression schemes, which generally provide much higher compression than lossless compression.

Lossless Compression

Lossless compression techniques, as their name implies, involve no loss of information. If data have been lossless compressed, the original data can be recovered exactly from the compressed data. Lossless compression is generally used for applications that cannot tolerate any difference between the original and reconstructed data. Text compression is an important area for lossless compression. It is very important that the reconstruction is identical to the original text, as very small differences can result in statements with very different meanings.
If data of any kind are to be processed or “enhanced” later to yield more information, it is important that the integrity be preserved.

Lossy Compression

Lossy compression techniques involve some loss of information, and data that have been compressed using lossy techniques generally cannot be recovered or reconstructed exactly.
In return for accepting this distortion in the reconstruction, we can generally obtain much higher compression ratios than is possible with lossless compression.

In many applications, this lack of exact reconstruction is not a problem. For example, when storing or transmitting speech, the exact value of each sample of speech is not necessary. Depending on the quality required of the reconstructed speech, varying amounts of loss of information about the value of each sample can be tolerated. If the quality of the reconstructed speech is to be similar to that heard on the telephone, a significant loss of information can be tolerated. However, if the reconstructed speech needs to be of the quality heard on a compact disc, the amount of information loss that can be tolerated is much lower.

3.2 Error free data

In information theory and coding theory with applications in computer science and telecommunication, error detection and correction or error control are techniques that enable reliable delivery of digital data over unreliable communication channels. Many communication channels are subject to channel noise, and thus errors may be introduced during transmission from the source to a receiver. Error detection techniques allow detecting such errors, while error correction enables reconstruction of the original data in many cases.

Parity Checks

A parity check is the process that ensures accurate data transmission between nodes during communication. A parity bit is appended to the original data bits to create an even or odd bit number; the number of bits with value one. The source then transmits this data via a link, and bits are checked and verified at the destination. Data is considered accurate if the number of bits (even or odd) matches the number transmitted from the source.

Parity checking, which was created to eliminate data communication errors, is a simple method of network data verification and has an easy and understandable working mechanism.

As an example, if the original data is 1010001, there are three 1s. When even parity checking is used, a parity bit with value 1 is added to the data's left side to make the number of 1s even; transmitted data becomes 11010001. However, if odd parity checking is used, then parity bit value is zero; 01010001.

If the original data contains an even number of 1s (1101001), then parity bit of value 1 is added to the data's left side to make the number of 1s odd, if odd parity checking is used and data transmitted becomes 11101001. In case data is transmitted incorrectly, the parity bit value becomes incorrect; thus, indicating error has occurred during transmission.

3.3 Data encryption

Encryption is the conversion of electronic data into another form, called cipher text, which cannot be easily understood by anyone except authorized parties.

The primary purpose of encryption is to protect the confidentiality of digital data stored on computer systems or transmitted via the Internet or other computer networks. Modern encryption algorithms play a vital role in the security assurance of IT systems and communications as they can provide not only confidentiality, but also the following key elements of security:

- Authentication: the origin of a message can be verified.
- Integrity: proof that the contents of a message have not been changed since it was sent.
- Non-repudiation: the sender of a message cannot deny sending the message.

Data, often referred to as plaintext, is encrypted using an encryption algorithm and an encryption key. This process generates cipher text that can only be viewed in its original form if decrypted with the correct key. Decryption is simply the inverse of encryption, following the same steps but reversing the order in which the keys are applied.

Source coding theorem

4.1 Motivation of source codes "lossless compression"

What is Source Coding?
The process by which information symbols are mapped to alphabetical symbols. The mapping is generally performed in sequences or groups of information and alphabetical symbols. Also, it must be performed in such a manner that it guarantees the exact recovery of the information symbol back from the alphabetical symbols otherwise it will destroy the basic theme of the source coding

Why it is important?
There are several reasons for using compression:

- Shorter messages take less time to transmit and so the complete message arrives more quickly at the recipient. This is good for both the sender and recipient since it frees up their network capacity for other purposes and reduces their network charges. For high-volume senders of data (such as Google, say), the impact of sending half as many bytes is economically significant.
- Using network resources sparingly is good for all the users who must share the internal resources (packet queues and links) of the network. Fewer resources per message means more messages can be accommodated within the network's resource constraints.

- Over error-prone links with non-negligible bit error rates, compressing messages before they are channel-coded using error-correcting codes can help improve throughput because all the redundancy in the message can be designed in to improve error resilience, after removing any other redundancies in the original message. It is better to design in redundancy with the explicit goal of correcting bit errors, rather than rely on whatever sub-optimal redundancies happen to exist in the original message.

- Examples for source coding applications:
  gzip, compress, winzip, ...
  Mobile voice, audio, and video transmission
  Internet voice, audio, and video transmission
  Digital television
  MP3 and portable video players (iPod, ...)
  Digital Versatile Discs (DVDs) and Blu-Ray Discs

4.2 How much compression?

Ideally we'd like to design our compression algorithm to produce as few bits as possible: just enough bits to represent the information in the message, but no more. How do we measure the information content of a message? Claude Shannon proposed that we define information as a mathematical quantity expressing the probability of occurrence of a particular sequence of symbols as contrasted with that of alternative sequences.
Let pi be the probability that the I_{th} choice occurs. Then the amount of information received when learning of choice I is

$$\textit{Information from } I_{th} \textit{ choice} = \log_2(1/Pi) \textit{ bits} \quad (1)$$

More information is received when learning of an unlikely choice (small $\boldsymbol{Pi}$) than learning of a likely choice (large pi). This jibes with our intuition about compression developed in so we'll use fewer bits when encoding such symbols. Similarly, infrequently occurring symbols have a lower pi and thus convey more information, so we'll use more bits when encoding such symbols. This exactly matches our goal of matching the size of the transmitted data to the information content of the message.
We can use equation (1) to compute the information content when learning of a choice by computing the weighted average of the information received for each particular choice:

Information content in a choice $= \sum_{i=1}^{N} \boldsymbol{Pi} * \mathbf{log_2(1/Pi)}$ **(2)**

This quantity is referred to as the information **entropy** or **Shannon's entropy** and is a **lower bound on the amount of information which must be sent**, on the average, when transmitting data about a particular choice.
What happens if we violate this lower bound, i.e., we send fewer bits on the average than called for by equation (22.3)? In this case the receiver will not have sufficient information and there will be some remaining ambiguity – exactly what ambiguity depends on the encoding, but in order to construct a code of fewer than the required number of bits, some of the choices must have been mapped into the same encoding. Thus, when the recipient receives one of the overloaded encodings, it doesn't have enough information to tell which of the choices actually occurred.

4.3 Fixed length vs. variable length encoding schemes

The easiest way to encode a message comprised from a set of symbols is to encode each symbol (regardless of its probability) with a fixed length code, i.e. for the case of the English language there would be 27 symbols (26 letters + space) requiring a 5 bit code for each symbol, so to transmit a 1000 symbol message we need 5000 bits transmitted.

With the knowledge of the probability weights of the English letters (which can depend on context also), we can calculate the entropy (the lower limit) of this kind of source, in fact Shannon calculated the entropy of English letters, a recent estimate puts the value between 1 and 1.5 bits of information per symbol, consequently, the minimum number of binary digits to encode a 1000 letter message goes down to 1500 bits, the way we can approach this lower limit is by using a variable length encoding scheme that can exploit the fact that more probable symbols convey much less information than unlikely symbols, thus requiring less space and resources(binary digits).

An example of such scheme is the Huffman coding.

4.4 Huffman coding

A development upon the fixed encoding scheme was devised by an MIT graduate in 1951, the Huffman code by David Huffman, it is a simple yet powerful algorithm that generates an optimal encoding provided a set of iid symbols, in its simplest form, the encoding is done per symbol, for more refined results (shorter message lengths) encoding pairs or triples of symbols can be chosen.

It's important to note that the Huffman technique is not adaptive, meaning it wouldn't be optimal if the symbol probabilities change from a message to the next.

A description of the algorithm in its simplest form (per symbol encoding) follows:

-a prerequisite is the mapping of all symbols available to the source to their probabilities, it's assumed implicitly that this mapping is static (doesn't change from a message to the next).

-1: choose two symbols with least probabilities.

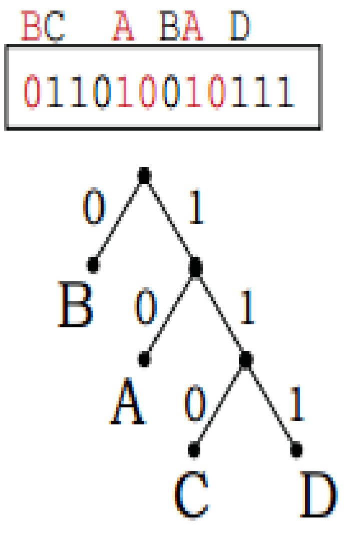
-2: construct a tree diagram with those two symbols as leafs.

-3: merge both symbols and add their probabilities, this new symbol is the node linking the two in the diagram.

-4: repeat the previous steps until all symbols are included on the tree.

-5: assign 0's to symbols on the leafs of the tree except one of the last two leafs, then assign 1's to branches.

choice$_i$	p_i	*encoding*
"A"	1/3	10
"B"	1/2	0
"C"	1/12	110
"D"	1/12	111



Huffman Decoding Tree

Limitations to Huffman coding

But in reality, a complete probability model of the source may not be known, also this model might change with time and a static encoding won't be much of a use in such case,
another useful property which can't be exploited by the Huffman technique , is the dependence of symbols upon previously transmitted symbols (context) ,this can in theory reduce the entropy of the source significantly, as the symbols become more predictable from their context, i.e. the case with English words and letters which are mostly used together like 'q' and 'u'.

For this reason, an adaptive variable length encoding scheme is needed to exploit such redundancy in sources.

4.5 Lempel-Ziv-welch compression technique

A lossless compression of symbol sequences which is adaptive to the context of the symbols, i.e. has memory, is the technique developed by ziv and Lempel, and improved by welch , this method is widely used to compress text, images(gif ,png ,tiff),generic files(zip).

A great advantage is that the source probability model neither has to be known or constant over time.

The core idea of it is that what is actually transmitted through the channel is a fixed-length code representing dictionary addresses (=array indices), both the receiving and the sending systems have a data structure (array/dictionary) that is dynamically built as the message is sent, starting from a core collection of symbols, both dictionaries are built simultaneously from symbol sequences "strings" within the message, those sequences are then referenced wholly by indices instead of referencing each constituting symbol alone, this can be of great usefulness in the case of English text for example, where many words are repeated over and over throughout the message and can be considered redundancies (don't add information).

The algorithm for the emitter and receiver of the message (equivalently the compressor/decompressor)
ensures that the dictionaries on both sides are mirrored (identical).

Channel coding theorem

Shannon's source-coding theorem

Also called the main theorem. In words:

You can compress N independent, identically distributed random variables, each with entropy H, down to NH bits with negligible loss of information (as N tends to infinity)

If you compress them into fewer than NH bits you will dramatically lose information

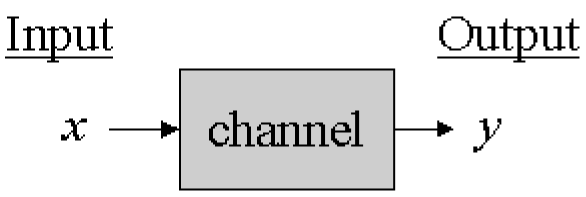
The theorem:

Let X be an ensemble with H(X) = H bits. Given $\varepsilon > 0$ and $0 < \delta < 1$, there exists a positive integer N_o such that, for $N > N_0$,

$$\left| \frac{1}{N} H\delta(X^N) - H \right| < \varepsilon$$

From the previous equation, the number of bits that we need to specify outcomes x with vanishingly small error probability ε does not exceed $H + \delta$, and if we accept a vanishingly small error, the number of bits we need to specify x drops from $NH_0(X)$ to $N\,(H + \varepsilon)$.

Information channels



H(X): information in input ensemble X

I(X; Y): is what we know about X given Y

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) \\ &= \sum_{xy} P(xy) \log \frac{P(xy)}{P(x)P(y)} \\ &= H(Y) - H(Y|X) \end{aligned}$$

X and Y form a joint ensemble, I(X; Y) is the average mutual information between x and y

Channel coding theorem

Definition: Channel capacity

The information capacity of a channel is: $C = \max [I(X; Y)]$

$I(X; Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$

The Theorem:

There is a nonnegative channel capacity C associated with each discrete memoryless channel with the following property: For any symbol rate $R < C$, and any error rate $E > 0$, there is a protocol that achieves a rate $>= R$ and a probability of error $<= E$.

- If the entropy of our symbol stream is equal to or less than the channel capacity, then there exists a coding technique that enables transmission over the channel with arbitrarily small error.
- Can transmit information at a rate $H(X) <= C$.

Shannon’s theorem tells us the asymptotically maximum rate

- It does not tell us the code that we must use to obtain this rate
- Achieving a high rate may require a prohibitively long code

Error-correction codes

Error-correcting codes allow us to detect and correct errors in symbol streams

It is used in all signal communications (digital phones, etc.)

Used in quantum computing to improve effects of decoherence

Many techniques and algorithms

Block codes

Hamming codes

BCH codes

Reed-Solomon codes

Turbo codes

Hamming codes

An example: Construct a code that corrects a single error

We add m check bits to our message

Can encode at most (2m - 1) error positions

The unused (all zeros) check means no error

Errors can occur in the message bits and/or in the check bits

If n is the length of the original message then 2m - 1 >= (n + m)

Examples:

If $n = 11$, $m = 4$: $2^4 - 1 = 15 >= (n + m) = 15$

If $n = 1013$, $m = 10$: $2^{10} - 1 = 1023 >= (n + m) = 1023$