REPORT -3

WIRELESS COMMUNICATIONS

Prof. Dr. Said E. El-Khamy
Communication & Electronics

Asmaa Gamal Abdel-Halem Mabrouk Nagy
أسماء جمال عبد الحليم مبروك ناجي
15010473

Different Techniques of Error Correction Codes, Selection diversity Proofs with their Graph sketching using computer, & Rake Receivers

Table of contents:

FIRSTLY:

DIFFERENT TECHNIQUES OF ERROR CORRECTION CODES TO COUNTER FOR BURST ERRORS

Introduction to Errors

Before moving on to the main topic we will first understand what are errors in computer network. Errors are basically some fault, problem or issue in a message. In computer networks messages are the collection of information that needs to be transferred in order to pass information from one source to another. If the message is not the same as it was while transferring then this situation is an error. In simple words we can say that when any message at sender side does not match the message received from receiver, then that is an error. These errors need to be corrected to allow smooth transfer of information from sender to receiver. So there are various types of error correcting codes and methods in computer network which help in correcting the errors. We will study each of them in detail. So let's get started.

Error Correcting Codes

Error correcting codes are a sequence of numbers that are produced or generated by some algorithms for detecting and correcting the errors that occurs while transfer of information. As we have already seen above errors so these codes detect and find the exact position at which error has been occurred and then try to correct them to allow proper transfer of information.
There are 2 types of error correcting codes in computer networks:

1.1. Block codes

1.2. Convolutional codes

1. Block codes :

code information in blocks as the name suggests. In block codes information bits are followed by parity bits. They are memoryless.
Block coding is a method used in digital electronics to encode data into a specific format. The purpose of block coding is to add redundant information to the data, which can be used to detect and correct errors that may occur during transmission or storage. Block coding is often used in conjunction with error correction codes (ECCs) to provide a more robust way of transmitting and storing data.

There are several types of block codes, including:

1. Hamming Codes: Hamming codes are a type of block code that can detect and correct single-bit errors. They are commonly used in digital systems to ensure the accuracy of transmitted data.

2. Reed-Solomon Codes: Reed-Solomon codes are a type of block code that can correct multiple-bit errors. They are commonly used in storage systems, such as CD-ROMs and DVDs, to ensure the integrity of stored data.
3. BCH Codes: BCH codes are a type of block code that can correct a specific number of errors. They are commonly used in digital communication systems to ensure the accuracy of transmitted data.
4. Block coding can provide many benefits in digital electronics, including improved reliability, increased data accuracy, and greater efficiency in the transmission and storage of data. However, block coding also has some disadvantages, including increased complexity and increased overhead in terms of processing time and memory usage.

In summary, block coding is a method used in digital electronics to encode data into a specific format, adding redundant information to the data to detect and correct errors that may occur during transmission or storage. There are several types of block codes, including Hamming codes, Reed-Solomon codes, and BCH codes, and they offer many benefits in terms of data accuracy and reliability, but also come with some disadvantages, such as increased complexity and overhead.

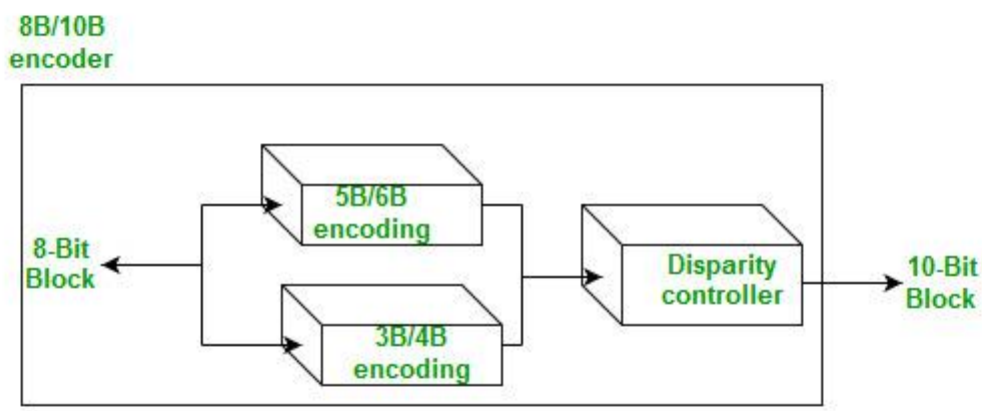
Conversion of Digital Data to Digital Signal involves three techniques:

1. Line Coding
2. Block Coding
3. Scrambling

Out of which Line coding is always needed, block coding and scrambling may or may not be needed. **Block coding** helps in error detection and re-transmission of the signal. It is normally referred to as mB/nB coding as it replaces each m-bit data group with an n-bit data group (where n>m). Thus, its adds extra bits (redundancy bits) which helps in synchronization at receiver's and sender's end and also providing some kind of error detecting capability. It normally involves three steps: division, substitution, and combination. In the division step,a sequence of bits is divided into groups of m-bits. In the substitution step, we substitute an m-bit group for an n-bit group. Finally, the n-bit groups are combined together to form a stream which has more bits than the original bits. Examples of mB/nB coding: **4B/5B (four binary/five binary) –** This coding scheme is used in combination with NRZ-I. The problem with NRZ-I was that it has a synchronization problem for long sequences of zeros. So, to overcome it we substitute the bit stream from 4-bit to 5-bit data group **before encoding it with NRZ-I**. So that it does not have a long stream of zeros. The block-coded stream does not have more than three consecutive zeros (see encoding table).

Data Sequence	Encoded Sequence	Data Sequence	Encoded Sequence
0000	11110	1000	10010
0001	01001	1001	10011
0010	10100	1010	10110
0011	10101	1011	10111
0100	01010	1100	11010
0101	01011	1101	11011
0110	01110	1110	11100
0111	01111	1111	11101

At the receiver, the NRZ-I encoded digital signal is first decoded into a stream of bits and then decoded again to remove the redundancy bits. **Drawback –** Though 4B/5B encoding solves the problem of synchronization, it increases the signal rate of NRZ-L. Moreover, it does not solve the DC component problem of NRZ-L. **8B/10B (eight binary/ten binary) –** This encoding is similar to 4B/5B encoding except that a group of 8 bits of data is now substituted by a 10-bit code and it provides greater error detection capability than 4B/5B. It is actually a combination of 5B/6B and 3B/4B encoding. The most five significant bits of a 10-bit block is fed into the 5B/6B encoder; the least 3 significant bits is fed into a 3B/4B encoder. The split is done to simplify the mapping table.



A group of 8 bits can have 2^8 different combinations while a group of 10 bits can have 2^10 different combinations. This means that there are 2^10-2^8=768 redundant groups that are not used for 8B/10B encoding and can be used for error detection and disparity check. Thus, this technique is better than 4B/5B because of better error-checking capability and better synchronization. **Reference-** Data Communications and Networking By Behrouz A.Forouzan(Book)

Advantages of Block Coding in Digital Electronics:

1. Improved Reliability: Block coding can improve the reliability of digital systems by adding redundant information to the data, which can be used to detect and correct errors that may occur during transmission or storage.
2. Increased Data Accuracy: Block coding can help ensure the accuracy of transmitted or stored data by detecting and correcting errors that may occur.
3. Greater Efficiency: Block coding can increase the efficiency of digital systems by reducing the number of errors that occur during transmission or storage, reducing the need for retransmission or storage.
4. Robustness: Block coding can provide a more robust way of transmitting and storing data by detecting and correcting errors that may occur, reducing the impact of errors on the overall system.

Disadvantages of Block Coding in Digital Electronics:

1. Increased Complexity: Block coding can increase the complexity of digital systems, making them more difficult to design, implement, and maintain.
2. Increased Overhead: Block coding can increase the overhead of digital systems in terms of processing time and memory usage, reducing overall system performance.
3. Increased Cost: Block coding can increase the cost of digital systems, as it requires additional hardware or software to implement the error correction and detection mechanisms.

Convolutional codes:

codes convolve information bit sequence. In convolution codes, information bits are spread along the sequence. Unlike block codes convolutional codes have memory. They use small codewords in comparison to block codes, both achieving the same quality.
These two are the two types of error correcting codes, now we will discuss about some error correcting codes and their working and how they correct and detect an error. In general we will study 4 basic error correcting codes in computer network.
They are:

1. Hamming codes
2. Binary Convolution Code
3. Reed - Solomon Code
4. Low-Density Parity Check Code

   We will cover it in detail.

2.Low-Density Parity Check Code

It is a linear block code which has the capability of correcting errors in blocks of large sizes. It is constructed using a sparse tanner graph. A low-density parity-check code is a code specified by a parity-check matrix.

MESSAGE ENCODING:

As we have seen above it is specified by a parity check matrix, which contains a major amount of 0's and very less amount of 1's. The parity check matrix has rows and columns which represent equations and bits in code symbols respectively. In this there are basically two parameters as n,i,j where n is block size, i is fixed number of 1's in each column and j is fixed number of 1's in each row.

MESSAGE DECODING:

There are two methods or processes through which message can be decoded.
When parity check bits are very small, we have to use decoders that help in doing parity checks. In the process, if we came across any bit that contains more than fixed number of parity equations that are not satisfied then we will simply reverse that particular bit. When we get new values, parity equations are recalculated. This whole process keep on running continuously until we get all parity equations which are satisfied.

The above process is easy to understand and implement but the problem arises when parity check bits are large. In that case we will be using probabilistic algorithms method. We will apply these algorithms on low density parity check graphs. As we have already seen above, LDPS is constructed using a sparse tanner graph, same graph is used here. The graph has two parts one which contains code symbols and other contains parity equations. Now if a code symbol is present then there is a line which connects two parts together. Message is decoded by passing these messages along these lines. Message is transmitted to check nodes from there it will come back to message nodes and in this whole process parity values are evaluated. This method is not easy to implement but it provided better result as compared to above method.

3.Polar codes:

Polar codes break the wheel somewhat in the field of channel coding with an unorthodox approach that resembles some of the operations more conventionally seen in the standard communication chains between the baseband and radio front ends.

One of the biggest surprises in 5G standardization so far has been the acceptance of polar codes as an official channel coding technology. Such decisions are of course complex ones that are often as much about political persuasion as technology virtue. Regardless, the feat achieved by this relatively nascent technology is remarkable. It was only a short time ago that the popular belief was that turbo codes would never be eclipsed. So, what makes polar codes different and how do they work?

How polar codes work

Polar codes are a channel coding technology and all channel coding technology works in basically a quite similar way. Communication links are susceptible to errors due to random noise, interference, device impairments, etc. that corrupt the original data stream at the receiving end. Channel coding basically employs a set of algorithmic operations on the original data stream at the transmitter, and another set of operations on the received data stream at the receiver to correct these errors. In channel coding terminology, the entirety of these operations at the transmitter and receiver are respectively denoted as encoding and decoding operations. The focus of channel coding research may be stated quite simply: develop high performance channel codes that mitigate the effect of the errors in a communication link (bit-error-rate is the common performance measure used here). However, the real challenge here is doing this in a manner of sufficiently low complexity that allows practical implementation into the silicon technology of the day. The complexity of a code determines everything, e.g. how much power it consumes, how much memory it needs, how much computation power it requires, and how much latency it incurs, all of which at the end of the day determine whether a code is good for any particular use case.

Channel coding methods broadly fall under two classes: block codes and convolutional codes. Block codes work on a block of data/bits with fixed size and apply the manipulation to this block at the transmitter and receiver. Reed-Solomon codes, commonly used on the hard disks of computers, are one example of this type of code. Convolutional codes, on the other hand, work on streams of data with more arbitrary numbers of data/bits. These codes apply a sliding window method that provides a substantial decoding benefit. Simple Viterbi codes are an example of this type of code.

As you might suspect, considerable productive research has been done over the years into the concatenation of block and convolutional codes to combine the benefits of both. For example, the RSV code, which was the best performing code until turbo codes, was a hybrid of the Reed-Solomon code, which is a block code, with a Viterbi convolutional code.

So how do turbo codes work? Turbo coding is an example of a concatenated code technique. The performance gains compared to other such codes are shockingly huge. They were the first set of codes to approach the Shannon capacity limit with a relatively moderate level of complexity. Turbo codes combine two convolutional type encoders, two serial decoders and an interleaver. Turbo codes get their name from the novel feedback loop they employ that, conceptually, *at least* resembles the same mechanism by which turbo exhaust systems work in cars. The real innovation in turbo codes lies in the smarts surrounding how soft information is used. Prior systems required hard knowledge about the bits (e.g. 0's or 1's) being received. However, turbo codes only require a probabilistic measure of each bit to be decoded correctly. This basically allows a lot more data to be successfully conveyed over turbo coded channels.

How are polar codes different?

Polar codes break the wheel somewhat in the field of channel coding. Polar codes operate on blocks of symbols/bits and are therefore technically members of the block code family. The construction of these codes follow a very unorthodox approach compared to more traditional approaches like turbo codes. In many ways, the methods resemble some of the operations that are mostly used in standard communication chains between the baseband and radio ends, primarily the family of Fast Fourier Transform (FFT) like techniques that anybody with a communications or signal processing background will be quite familiar with.

The transformation involves two key operations called channel combining and channel splitting. The channel combining process reminds me of the bit manipulations and sub-block mapping in OFDM waveform configuration, as is used in conventional radio modulation schemes. That is, at the encoder, channel combining initially assigns/maps carefully curated combinations of bits/symbols to specific channels (e.g. as in sub-blocks in an OFDM symbol).

The channel splitting that follows performs an implicit transformation operation (analogous to frequency domain to time domain as performed by an IFFT operation), translating these bit/symbol combinations into decoder-ready, time domain vectors. The decoding operation at the receiver, in symmetry with the encoding, tries to estimate these time domain bit streams by using a conventional successive-cancellation decoding technique, analogous to spectral domain estimation.

The real magic in polar codes lies in the clever bit manipulations and mappings to the channels at the encoder. The original technique, in combination with channel splitting, and successive-cancellation decoding, is shown to essentially convert a block of bits, and their associated channels between the encoder and decoder, into a *polarized* bit stream at the receiver. That is a received bit and its associated channel ends up being either a “good channel” or “bad channel” pole/category. It has been mathematically proven that as the size of the bit block increases, the received bit stream *polarizes* in a way that the number of “good channels” approaches Shannon capacity. This phenomenon is what gives polar codes their name and makes them the first and only explicitly proven capacity-achieving practical channel codes ever conceived.

The astonishing characteristic of this coding technique is that the implementation complexity, especially the decoding part, is considerably less when compared to existing turbo codes. Moreover, polar codes and their emerging array of variants and hybrids are shown to out-perform turbo codes in terms of error-correction performance in a wide range of use-cases. The better error-correction performance along with lower implementation and operation complexities make polar codes a genuine step up on turbo code technology.

What is next for this technology?

Polar codes have so far replaced turbo codes in 5G eMBB (enhanced Mobile Broadband) control channels and on the physical broadcast channel. While these channels typically only operate at low data rates, the reduced complexity of polar codes has made them a more attractive choice than turbo codes. Two more uses are still to be addressed in 5G: URLLC (ultra-reliable low latency communications) and MMTC (massive machine-type communications). These use cases, particularly URLLC, have comparable data rate requirements to control channels. The excellent error-correction capability of polar codes certainly makes them a strong contender in this race in progress.

Looking to the future, for polar codes to be crowned as the universal coding method, the main challenge is to break the data rate barrier, while retaining error-correction performance as well as the low-complexity advantages. Data channels typically operate in much higher data rates than control channels. 5G peak mobile broadband data rates are expected to be around 20Gbps. Beyond-5G systems are expected to operate at Terabit/s data rates. Today's most advanced polar code implementations currently deliver only around 5Gbps. There is however a solid theoretical framework underlying polar codes that will be built on and should not prevent them from full ascension over time.

4. TURBO CODES:

Turbo codes form the basis of mobile communications in 3G and 4G networks. Invented in 1991 by Claude Berrou, and published in 1993 with Alain Glavieux and Punya Thitimajshima, they have now become a reference point in the field of information and communication technologies. As Télécom Bretagne, birthplace of these "error-correcting codes", prepares to host the 9th international symposium on turbo codes, let's take a closer look at how these codes work and the important role they play in our daily lives.

What role do turbo codes play in correction codes?

In the digital communications sector, there are several error-correcting codes, with varying levels of complexity. Typically, repeating the same message several times in binary code is a relatively safe bet, yet it is extremely costly in terms of bandwidth and energy consumption.

Turbo codes are a much more developed way of integrating information redundancy. They are based on the transmission of the initial message in three copies. The first copy is the raw, non-encoded information. The second is modified by encoding each bit of information using an algorithm shared by the coder and decoder. Finally, another version of the message is also encoded, but after modification (specifically, a permutation). In this third case, it is no longer the original message that is encoded and then sent, but rather a transformed version. These three versions are then decoded and compared in order to find the original message.

Where are turbo codes used?

In addition to being used to encode all our data in 3G and 4G networks, turbo codes are also used in many different fields. NASA uses them for its communication with space probes which have been built since 2003. The space community, which has to contend with many constraints on communication processes, is particularly fond of these codes, as ESA also uses them for many of its probes. But more generally, turbo codes represent a safe and efficient encoding technique in most communication technologies.

How have turbo codes become so successful?

In 1948, American engineer and mathematician Claude Shannon proposed a theorem stating that codes always exist that are capable of minimizing channel-related transmission errors, up to a certain level of disturbance. In other words, Shannon asserted that, despite the noise in a channel, the transmitter will always be able to transmit an item of information to the receiver, almost error-free, when using efficient codes.

The turbo codes developed by Claude Berrou in 1991 meet these requirements, and are close to the theoretical limit for information transmitted with an error rate close to zero. Therefore, they represent highly efficient error-correcting codes. His experimental results, which validated Shannon's theory, earned Claude Berrou the Marconi Prize in 2005 – the highest scientific distinction in the field of communication sciences. His research earned him a permanent membership position in the French Academy of Sciences.

5. Fountain code:

In coding theory, **fountain codes** (also known as **rateless erasure codes**) are a class of erasure codes with the property that a potentially limitless sequence of encoding symbols can be generated from a given set of source symbols such that the original source symbols can ideally be recovered from any subset of the encoding symbols of size equal to or only slightly larger than the number of source symbols. The term *fountain* or *rateless* refers to the fact that these codes do not exhibit a fixed code rate.

A fountain code is optimal if the original *k* source symbols can be recovered from any *k* successfully received encoding symbols (i.e., excluding those that were erased). Fountain codes are known that have efficient encoding and decoding algorithms and that allow the recovery of the original *k* source symbols from any *k'* of the encoding symbols with high probability, where *k'* is just slightly larger than *k*.

LT codes were the first practical realization of fountain codes. Raptor codes and online codes were subsequently introduced, and achieve linear time encoding and decoding complexity through a pre-coding stage of the input symbols.

Applications

Fountain codes are flexibly applicable at a fixed code rate, or where a fixed code rate cannot be determined a priori, and where efficient encoding and decoding of large amounts of data is required.

One example is that of a data carousel, where some large file is continuously broadcast to a set of receivers. Using a fixed-rate erasure code, a receiver missing a source symbol (due to a transmission error) faces the coupon collector's problem: it must successfully receive an encoding symbol which it does not already have. This problem becomes much more apparent when using a traditional short-length erasure code, as the file must be split into several blocks, each being separately encoded: the receiver must now collect the required number of missing encoding symbols for *each* block. Using a fountain code, it suffices for a receiver to retrieve *any* subset of encoding symbols of size slightly larger than the set of source symbols. (In practice, the broadcast is typically scheduled for a fixed period of time by an operator based on characteristics of the network and receivers and desired delivery reliability, and thus the fountain code is used at a code rate that is determined dynamically at the time when the file is scheduled to be broadcast.)

Another application is that of hybrid ARQ in reliable multicast scenarios: parity information that is requested by a receiver can potentially be useful for *all* receivers in the multicast group.

In standards

Raptor codes are the most efficient fountain codes at this time,[2] having very efficient linear time encoding and decoding algorithms, and requiring only a small constant number of XOR operations per generated symbol for both encoding and decoding.[3] IETF RFC 5053 specifies in detail a systematic Raptor code, which has been adopted into multiple standards beyond the IETF, such as within the 3GPP MBMS standard for broadcast file delivery and streaming services, the DVB-H IPDC standard for delivering IP services over DVB networks, and DVB-IPTV for delivering commercial TV services over an IP network. This code can be used with up to 8,192 source symbols in a source block, and a total of up to 65,536 encoded symbols generated for a source block. This code has an average relative reception overhead of 0.2% when applied to source blocks with 1,000 source symbols, and has a relative reception overhead of less than 2% with probability 99.9999%.[4] The relative reception overhead is defined as the extra encoding data required beyond the length of the source data to recover the original source data, measured as a percentage of the size of the source data. For example, if the relative reception overhead is 0.2%, then this means that source data of size 1 megabyte can be recovered from 1.002 megabytes of encoding data.

A more advanced Raptor code with greater flexibility and improved reception overhead, called RaptorQ, has been specified in IETF RFC 6330. The specified RaptorQ code can be used with up to 56,403 source symbols in a source block, and a total of up to 16,777,216 encoded symbols generated for a source block. This code is able to recover a source block from any set of encoded symbols equal to the number of source symbols in the source block with high probability, and in rare cases from slightly more than the number of source symbols in the source block. The RaptorQ code is an integral part of the ROUTE instantiation specified in ATSC A-331 (ATSC 3.0)

For data storage

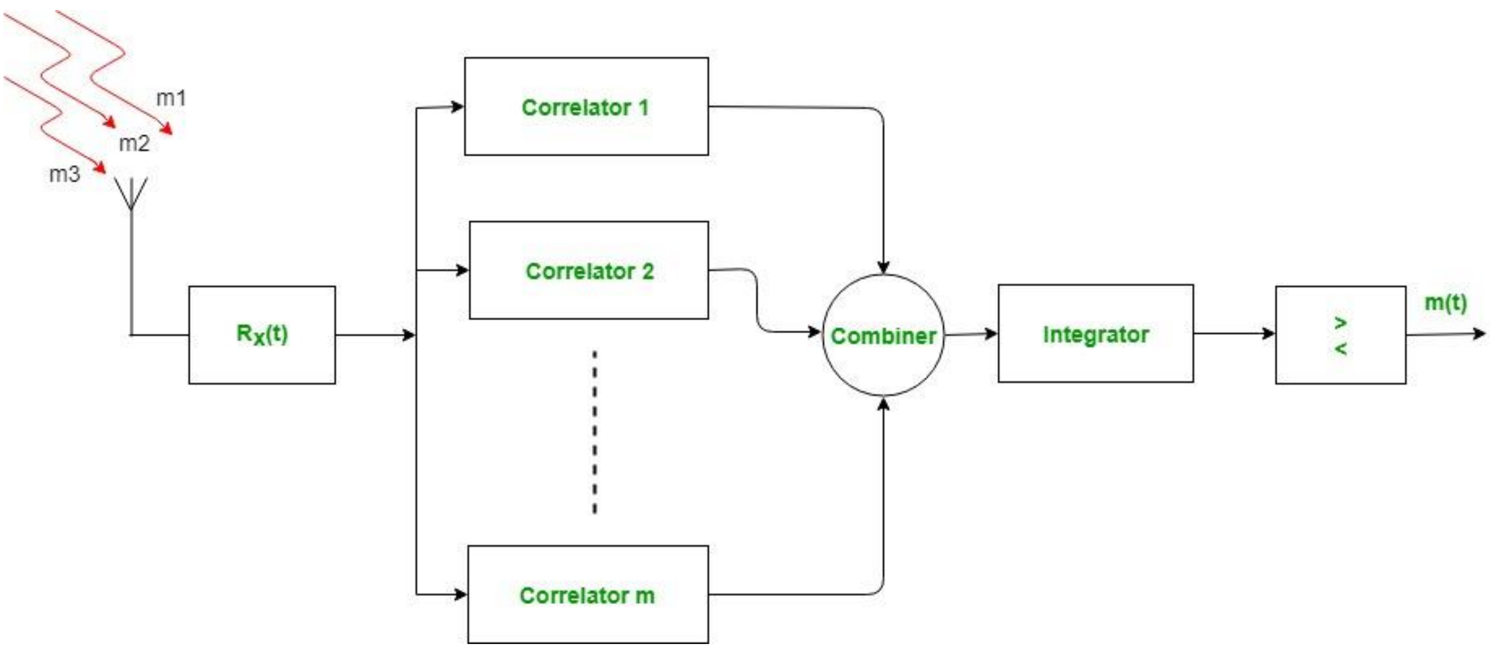
Erasure codes are used in data storage applications due to massive savings on the number of storage units for a given level of redundancy and reliability. The requirements of erasure code design for data storage, particularly for distributed storage applications, might be quite different relative to communication or data streaming scenarios. One of the requirements of coding for data storage systems is the systematic form, i.e., the original message symbols are part of the coded symbols. Systematic form enables reading off the message symbols without decoding from a storage unit. In addition, since the bandwidth and communication load between storage nodes can be a bottleneck, codes that allow minimum communication are very beneficial particularly when a node fails and a system reconstruction is needed to achieve the initial level of redundancy. In that respect, fountain codes are expected to allow efficient repair process in case of a failure: When a single encoded symbol is lost, it should not require too much communication and computation among other encoded symbols in order to resurrect the lost symbol. In fact, repair latency might sometimes be more important than storage space savings. Repairable fountain codes are projected to address fountain code design objectives for storage systems. A detailed survey about fountain codes and their applications can be found at.

A different approach to distributed storage using fountain codes has been proposed in Liquid Cloud Storage. Liquid Cloud Storage is based on using a large erasure code such as the RaptorQ code specified in IETF RFC 6330 (which provides significantly better data protection than other systems), using a background repair process (which significantly reduces the repair bandwidth requirements compared to other systems), and using a stream data organization (which allows fast access to data even when not all encoded symbols are available).

SECONDLY:

RAKE RECEIVER

The dictionary meaning of rake is *to gather or collect together something* and actually is a garden tool to collect leaves. But in the terms of computer network, it is for the purpose of collecting signals from multiple paths arriving at the receiver end and used specially in CDMA cellular systems. A Rake Receiver is a radio receiver which is designed for the purpose to counter the effects of multipath fading. Due to reflections from multiple obstacles in the environment, the radio channel can consist of multiple copies of the transmitted signal having different amplitude, phases or delays. A rake receiver can resolve this issue and combine them. For this purpose, several sub-receivers are used which are known as "fingers". The idea of a basic rake receiver was first proposed by **Price** and **Green**.



When the transmitter transmits the signal then it travels through the environment which consists of various obstacles and the transmitted signal is reflected by them and is received by the rake receiver from multiple paths. Rake receiver then feeds them to different fingers (correlators). The delays in each received signal are compensated and are feeded to the Combiner, Integrator and Comparator which combines them suitably with different appropriate time delays.

Working of Rake Receiver:

1. The received signal from multiple paths arrives at the antenna with different delays and phases.
2. The signal is first passed through a bank of matched filters, each of which corresponds to a specific path.
3. The output of each matched filter is sampled at the symbol rate and the resulting samples are combined.
4. The combining process is done using a technique called maximum ratio combining (MRC) which gives more weight to the signals that have higher signal-to-noise ratio (SNR).
5. The combined signal is then demodulated to obtain the transmitted symbols.
6. The rake receiver also performs channel estimation by estimating the complex gains of each path using a technique called pilot symbols.
7. The channel estimates are used to adjust the weights in the combining process to ensure optimal performance.
8. The rake receiver also uses a technique called diversity combining to reduce the effect of fading by combining the signals from multiple antennas.
9. The combined signal from each antenna is then passed through the matched filter and combined using maximum ratio combining.
10. The rake receiver is able to recover the transmitted signal even in the presence of severe multipath fading and interference.

Some key applications of Rake receiver are:

- **CDMA Systems:** Rake receivers are extensively used in CDMA systems, where they are used to combat the effects of multipath fading.
- **Wireless Networks:** Rake receivers are also used in wireless networks to improve the performance of the system, especially in environments where the signal is weak.
- **Satellite Communications:** In satellite communication systems, Rake receivers are used to detect and extract weak signals that have traveled long distances.
- **Mobile Communications:** Rake receivers are used in mobile communication systems to improve the quality of the received signal by minimizing the effects of multipath fading.
- **High-Speed Data Transmission:** Rake receivers are used in high-speed data transmission systems to reduce errors caused by the transmission of signals over long distances.

Some key advantages of rake receiver are:

- **Multipath mitigation:** The rake receiver mitigates the effect of multipath propagation by combining multiple replicas of the same signal, which have been transmitted over different paths.
- **Diversity gain:** By combining multiple replicas of the same signal, the rake receiver provides diversity gain, which helps to improve the signal-to-noise ratio (SNR) and reduce the bit error rate (BER) of the received signal.

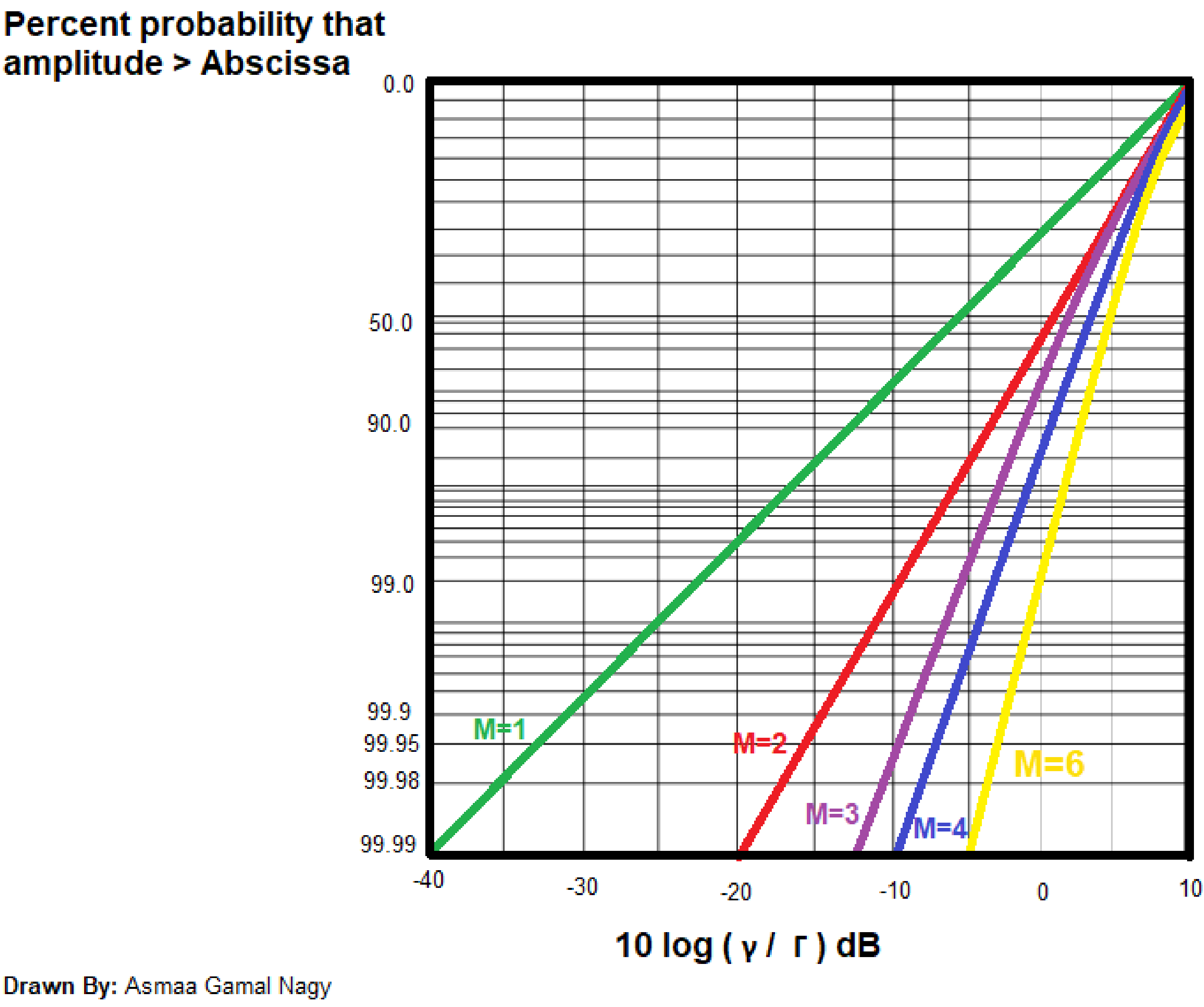
- **Interference rejection:** The rake receiver can also be used to reject interference from other signals that are transmitted over the same frequency band.
- **Simple implementation:** The rake receiver is a simple and effective technique that can be implemented using digital signal processing (DSP) algorithms.
- **Compatibility:** The rake receiver is compatible with different modulation schemes and can be used in various wireless communication systems, such as code division multiple access (CDMA), time division multiple access (TDMA), and frequency division multiple access (FDMA).
- **Improvement in system capacity:** The use of a rake receiver in a wireless communication system can increase the system capacity by allowing the use of more frequency bands or by increasing the number of users that can be supported.

Some key disadvantages of rake receiver are:

- **Complex implementation:** The Rake receiver is a complex receiver with a large number of matched filters, which makes it difficult to implement.
- **High power consumption:** The Rake receiver requires a lot of power to operate, which can be a problem in battery-powered devices.
- **Limited performance in deep fades:** The Rake receiver is not very effective in deep fades, where the signal is severely attenuated. In such cases, other techniques such as diversity combining may be more effective.
- **Sensitivity to multipath delay spread:** The Rake receiver is designed to work in the presence of multipath, but it is sensitive to the delay spread of the channel. In channels with large delay spreads, the performance of the Rake receiver may degrade.
- **Limited applicability to narrowband systems:** The Rake receiver is designed to work with wideband systems that have significant multipath propagation. In narrowband systems with little or no multipath, the Rake receiver may not be necessary or effective.

The Rake receiver is an important solution to the problem of multipath fading in wireless communication. It is designed to improve the signal quality and reduce the effects of fading by using multiple copies of the transmitted signal that arrive at the receiver at different times. Although the Rake receiver is more complex, more power-consuming, and more expensive than other types of receivers, it provides better coverage and improved data rates, making it an attractive solution for high-speed wireless communication.

<u>Thirdly:</u> **Sketching using computer for the selection diversity Graph of its SNR probability distribution using the threshold SNR for M branch & the mean SNR on each branch.**



<u>Fourthly:</u> The Handwritten Proofs of:

- ✓ Selection space diversity SNR
- ✓ SNR probability density function PDF
- ✓ Selection SNR Improvement factor.

Proof: Selection diversity Improvement

* Consider: a Fading Channel (Rayleigh = NLOS + Multipath fading only)

* Prove : SNR PDF of 1 branch in Selection diversity follows Chi^2 distribution

Solution

$$S(t) \xrightarrow{\text{input}} \boxed{\text{Channel}} \xrightarrow{\text{output}} r(t)$$

∵ input - output relation:

$$r(t) = \underbrace{\alpha(t)\, e^{-j\theta(t)}}_{\substack{\text{effect of fading} \\ \equiv \text{ Multiplicated noise}}}\; S(t) + \underbrace{n(t)}_{\substack{\text{effect of noise} \\ \equiv \text{ Gaussian noise is added (summed)}}}$$

$$\therefore \text{average } SNR_{SD} = \overline{SNR}_{SD} = \Gamma = \frac{E_b}{N_0}\, \overline{\alpha^2(t)}$$

↳ average magnitude square of fading

$\alpha > 1 \xrightarrow{\therefore} SNR \uparrow$

$\alpha < 1 \xrightarrow{\therefore} SNR \downarrow$

∵ $r(t)$ follows Rayleigh distribution

$$\therefore \boxed{P(r) = \frac{r}{\sigma^2}\, e^{-r^2/2\sigma^2}} \xrightarrow{\text{where}} \begin{aligned} r &= \text{received envelope} \\ &= |r(t)| = \sqrt{S_i^2 + S_q^2} \end{aligned}$$

$$\therefore \text{Power of received envelope} = \boxed{Z = r^2}$$

$$\therefore \boxed{dZ = 2r\,dr}$$

∴ Changing Variables in Probabilities: $r^2 \to Z$

$$P(Z)\,\cancel{dZ} = P(r)\,dr = \frac{\cancel{r}}{\sigma^2}\, e^{-\frac{r^2}{2\sigma^2}} \cdot \frac{\cancel{dZ}}{2r}$$

$$\therefore P(Z) = \frac{1}{2\sigma^2}\, e^{-\frac{Z}{2\sigma^2}} \qquad (\text{where } Z > 0)$$

let: $\Gamma = 2\sigma^2$ (but this Value is a ratio without units)

$$\hookrightarrow SNR = \gamma_i = \frac{\text{Signal Power}}{\text{noise Power}} = \frac{E_{b_i}\, \alpha_i^2}{(N_{0_i})} = \frac{Z}{\sigma^2} \rightarrow \text{Scaler or number or Const in 1 branch} \quad (i)$$

Note

∵ Most of Signal Variance is due to Noise Variance

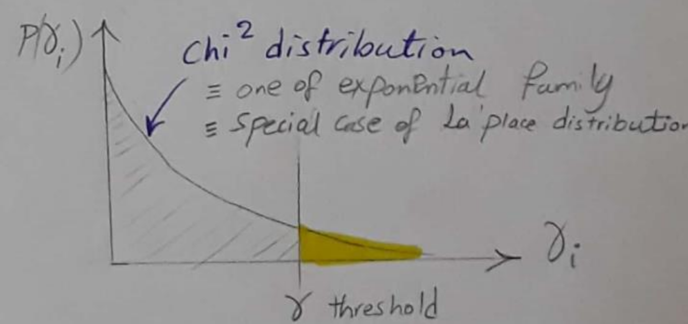
∴ Signal Variance = noise Variance = Noise Power = $N_0 = \sigma^2$

$$\therefore P(z) = \frac{1}{\Gamma} e^{-\frac{z}{\Gamma}}$$ → Chi-square distribution

$$\therefore P\left(\frac{z}{\sigma^2}\right) = \frac{1}{\Gamma} e^{-\frac{1}{\Gamma}\frac{z}{\sigma^2}}, \quad \frac{z}{\sigma^2} \to \gamma_i$$

$$\therefore \boxed{P(\gamma_i) = \frac{1}{\Gamma} e^{-\frac{\gamma_i}{\Gamma}}}$$

[where: → $\gamma_i > 0$
→ SNR @ R_x number (i)]



$$\therefore CDF\Big|_{@1\ antenna} = P(\gamma_i \le \gamma_{threshold})$$

$$= \int_0^{\gamma} P(\gamma_i)\, d\gamma_i = \int_0^{\gamma} \frac{1}{\Gamma} e^{-\frac{\gamma_i}{\Gamma}}\, d\gamma_i$$

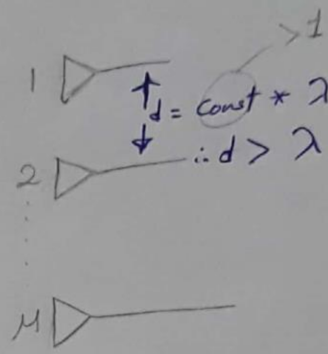
$$= \frac{1}{\Gamma} \cdot \frac{\left[e^{-\gamma_i/\Gamma}\right]_0^{\gamma}}{-1/\Gamma} = +\left[e^{-\gamma_i/\Gamma}\right]_{\gamma}^{0}$$

$$CDF\Big|_{@1\ antenna\ of\ Selection} = 1 - e^{-\gamma/\Gamma}$$ [no diversity Case]

$$\therefore P(\gamma_i > \gamma) = 1 - P(\gamma_i < \gamma) = e^{-\gamma/\Gamma}$$

Proof: For more than 1 antenna the Improvement factor is equal to $\frac{\bar{\gamma}}{\Gamma} = \sum_{k=1}^{M} \frac{1}{k}$ in Selection Space diversity ③

$\boxed{\text{Solution}}$



* for more than one antenna:

γ for indep Paths $d > \lambda$

$\therefore P(\gamma_1, \gamma_2, \dots, \gamma_M \le \gamma) = \prod_{i=1}^{M} P(\gamma_i \le \gamma)$

$= P(\gamma_1 \le \gamma) \cdot P(\gamma_2 \le \gamma) \cdots \cdot P(\gamma_M \le \gamma)$

$\therefore (CDF)^M = (1 - e^{-\gamma/\Gamma})^M = P_M(\gamma) =$ Prob that all antennas signals < threshold

$\therefore P(\gamma_i > \gamma) =$ Prob that anyone of the R_x antennas SNR > threshold

$= 1 - P_M(\gamma) = 1 - (1 - e^{-\gamma/\Gamma})^M$

* to find average SNR in MIMO selection space diversity:

① PDF of MIMO selection = differential of CDF $= P_M(\gamma)$ Small

$= \frac{d}{d\gamma} \underset{\text{capital}}{P_M}(\gamma) = M(1 - e^{-\gamma/\Gamma})^{M-1} * (-)e^{-\gamma/\Gamma} * -\frac{1}{\Gamma}$

$= \frac{M}{\Gamma} e^{-\gamma/\Gamma} (1 - e^{-\gamma/\Gamma})^{M-1}$

② $\therefore \bar{\gamma} = \overline{SNR}_{M\,SD} =$ mean selection SNR with M Branches

$= \int_0^\infty \underbrace{\boxed{\gamma}}_{\gamma_i \text{ of 1 branch}(i) = \Gamma_i = \Gamma} \overset{\text{PDF}}{P_M(\gamma)}\, d\gamma = \int_0^\infty M \frac{\gamma}{\Gamma} \cdot e^{-\gamma/\Gamma} (1 - e^{-\gamma/\Gamma})^{M-1} d\gamma$

$$\boxed{\underbrace{\bar{\gamma}}_{\text{mean/average SNR of M Branches}} = \underbrace{\Gamma}_{\text{average SNR of 1 branch}} \underbrace{\sum_{k=1}^{M} \frac{1}{k}}_{\text{Improvement factor} > 1}}$$

$\therefore \Rightarrow M \uparrow$ avg received Power $\uparrow$ improvement become Smaller $\downarrow$

(4)

∴ average SNR Improvement factor of Selection Diversity $= \boxed{\frac{\bar{\gamma}}{\Gamma}} = \boxed{\sum_{k=1}^{M} \frac{1}{k}}$

∴ $\bar{\gamma} = \boxed{\Gamma \left(1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \cdots + \frac{1}{m} \right)} > 1$

∴ $\bar{\gamma} > \Gamma$ (Because $\sum_{k=1}^{M} \frac{1}{k} > 1$)

#


thank you


BYE