Part I

Basics of coding theory and linear codes

CODING and CRYPTOGRAPHY

IV054

IV054

Modern **Coding theory** is a very beautiful and often very surprising mathematical theory that is very much applied and broadly used for transmission of digital information, without which modern telecommunication would be practically impossible.

IV054

Modern **Coding theory** is a very beautiful and often very surprising mathematical theory that is very much applied and broadly used for transmission of digital information, without which modern telecommunication would be practically impossible. . Mostly everyone is daily using outcomes of modern coding and decoding.

Modern **Cryptography** is rich on clever use of beautiful and often much surprising concepts and methods that allows to use outcomes of modern classical and also surprisingly quantum tools, to make transmission of information so safe that even very powerful eavesdropper has next to zero chance to read transmitted information that not intended to him.

CODING and CRYPTOGRAPHY

IV054

Modern **Coding theory** is a very beautiful and often very surprising mathematical theory that is very much applied and broadly used for transmission of digital information, without which modern telecommunication would be practically impossible. . Mostly everyone is daily using outcomes of modern coding and decoding.

Modern **Cryptography** is rich on clever use of beautiful and often much surprising concepts and methods that allows to use outcomes of modern classical and also surprisingly quantum tools, to make transmission of information so safe that even very powerful eavesdropper has next to zero chance to read transmitted information that not intended to him.

In spite of the fact that both coding and cryptography areas have already many very efficient systems using only very small memories, new and new applications require to develop again and again new, faster, and less memory demanding systems for both coding and cryptography.

IV054-CRYPTO team;

Teaching stuff

- Prof. Jozef Gruska DrSc - lecturer
- RNDr. Matej Pivoluska PhD - tutorials end CRYPTO team member
- RNDr Lukáš Boháč - head of CRYPTO-team
- Mgr. Libor Cáha PhD, member of CRYPTO-team
- Bc. Henrieta Micheľová, member of CRYPTO-team
- Bc. Roman Oravec, member of CRYPTO-team

IV054-CRYPTO team;

Teaching stuff

- Prof. Jozef Gruska DrSc - lecturer
- RNDr. Matej Pivoluska PhD - tutorials end CRYPTO team member
- RNDr Lukáš Boháč - head of CRYPTO-team
- Mgr. Libor Cáha PhD, member of CRYPTO-team
- Bc. Henrieta Micheľová, member of CRYPTO-team
- Bc. Roman Oravec, member of CRYPTO-team

Teaching loads: lecture - 2 hours, tutorial 2 hours - non obligatory

IV054-CRYPTO team;

Teaching stuff

- Prof. Jozef Gruska DrSc - lecturer
- RNDr. Matej Pivoluska PhD - tutorials end CRYPTO team member
- RNDr Lukáš Boháč - head of CRYPTO-team
- Mgr. Libor Cáha PhD, member of CRYPTO-team
- Bc. Henrieta Micheľová, member of CRYPTO-team
- Bc. Roman Oravec, member of CRYPTO-team

Teaching loads: lecture - 2 hours, tutorial 2 hours - non obligatory
Languages: lecture - English, tutorials 1 in English and 1 in Czech-Slovak

IV054-CRYPTO team;

Teaching stuff

- Prof. Jozef Gruska DrSc - lecturer
- RNDr. Matej Pivoluska PhD - tutorials end CRYPTO team member
- RNDr Lukáš Boháč - head of CRYPTO-team
- Mgr. Libor Cáha PhD, member of CRYPTO-team
- Bc. Henrieta Micheľová, member of CRYPTO-team
- Bc. Roman Oravec, member of CRYPTO-team

Teaching loads: lecture - 2 hours, tutorial 2 hours - non obligatory
Languages: lecture - English, tutorials 1 in English and 1 in Czech-Slovak

Prerequisites: Basics of discrete mathematics and linear algebra See: "Appendix" in http://www.fi.muni.cz/usr/gruska/crypto21,

IV054 - Homeworks and exams

Homeworks 5-6 sets of homeworks of 6-8 exercises designed and evaluated by our CRYPTO-team created mainly from some of best former IV054-students

IV054 - Homeworks and exams

Homeworks 5-6 sets of homeworks of 6-8 exercises designed and evaluated by our CRYPTO-team created mainly from some of best former IV054-students

Termination of the course - Exams or zapocty

IV054 - Homeworks and exams

Homeworks 5-6 sets of homeworks of 6-8 exercises designed and evaluated by our CRYPTO-team created mainly from some of best former IV054-students

Termination of the course - Exams or zapocty

Each student will get 5 questions. Number of questions a student has to respond will depend on the number of points received for homeworks. Each student will get automatically A in case (s)he received number of points from exercises $>=$ 85% of MAX - maximal number of points any student got from exercises.

Automatically a student gets B, with an easy way to get A, in case the number of points (s)he received is in interval (75,85)% of MAX.......

IV054 - Homeworks and exams

Homeworks 5-6 sets of homeworks of 6-8 exercises designed and evaluated by our CRYPTO-team created mainly from some of best former IV054-students

Termination of the course - Exams or zapocty

Each student will get 5 questions. Number of questions a student has to respond will depend on the number of points received for homeworks. Each student will get automatically A in case (s)he received number of points from exercises $>=$ 85% of MAX - maximal number of points any student got from exercises.

Automatically a student gets B, with an easy way to get A, in case the number of points (s)he received is in interval (75,85)% of MAX.......

Teaching materials

IV054 - Homeworks and exams

Homeworks 5-6 sets of homeworks of 6-8 exercises designed and evaluated by our CRYPTO-team created mainly from some of best former IV054-students

Termination of the course - Exams or zapocty

Each student will get 5 questions. Number of questions a student has to respond will depend on the number of points received for homeworks. Each student will get automatically A in case (s)he received number of points from exercises $>=$ 85% of MAX - maximal number of points any student got from exercises.

Automatically a student gets B, with an easy way to get A, in case the number of points (s)he received is in interval (75,85)% of MAX.......

Teaching materials

- Detailed slides of all lectures. (Each chapter will consists of a (i) short prologue, (ii) basic materials and an (iii) Appendix -for much demanding students.

IV054 - Homeworks and exams

Homeworks 5-6 sets of homeworks of 6-8 exercises designed and evaluated by our CRYPTO-team created mainly from some of best former IV054-students

Termination of the course - Exams or zapocty

Each student will get 5 questions. Number of questions a student has to respond will depend on the number of points received for homeworks. Each student will get automatically A in case (s)he received number of points from exercises $>=$ 85% of MAX - maximal number of points any student got from exercises.

Automatically a student gets B, with an easy way to get A, in case the number of points (s)he received is in interval (75,85)% of MAX.......

Teaching materials

- Detailed slides of all lectures. (Each chapter will consists of a (i) short prologue, (ii) basic materials and an (iii) Appendix -for much demanding students.
- Appendix of fundamental discrete math and linear algebra - 45 pages
- Two lecture notes of solved examples (at least 100 in each one) and short (2-3) pages overviews for all chapters.
- Posted solutions of homeworks

IV054 - goals

Goals

1. To learn beautiful and powerful **basics** of the coding theory and of the classical as well as quantum modern cryptography and steganography-watermarking needed for all informaticians; in almost all areas of informatics and for transmission and storing information.

Goals

1. To learn beautiful and powerful **basics** of the coding theory and of the classical as well as quantum modern cryptography and steganography-watermarking needed for all informaticians; in almost all areas of informatics and for transmission and storing information.

2. To verify, for ambitious students, their capability to work hard to be successful in very competitive informatics+mathematics environments.

BIBLIOGRAPHY

- J. Gruska: Foundation of computing. Thomson International Computer Press, 1997
- V. Pless: Introduction to the theory of error correcting codes, John Willey, 1998
- A. De Vos: Reversible Computing, Viley, VCH Verlg, 2010, 249 p.
- W. Trape, L. Washington: Introduction to cryptography with coding theory
- D.R. Stinson: Cryptography: Theory and practice, CRC Press, 1995
- A. Salomaa: Public-key cryptography, Springer, 1990
- B. Schneier: Applied cryptography, John Willey and Son, 1996
- J. Hoffsten, J. Peper, J. Silveman: An introuction to Mathematial cryptography (elypti curves), Springer, 2008
- I. J. Cox: Digital Watermarking and Steganography, Morgan Kufman eries in Multimedia Information and Systems, 2008
- J. Gruska: Quantum computing, McGraw Hill, 1999,430 pages
- M. A. Nielsen, I. I. Chuang: Quantum computtion and Quantum Information Cambridge University Press, 2000, 673 p.
- D. Kahn: The codebreker. Two tory of secret wriing, Mcmilan, 1996 (An alternative and informative hitory of cryptography.)

CONTENS

CONTENS

Beautiful and much applied **coding theory** - efficiency and miniaturization of modern information transition systems depends much on the quality and efficiency of the underlying encoding and decoding systems.

CONTENS

Beautiful and much applied **coding theory** - efficiency and miniaturization of modern information transition systems depends much on the quality and efficiency of the underlying encoding and decoding systems.

Basic encryption systems and decryption methods of the classical, secret and public key cryptography. Blocks and streams encryption and decryption systems and methods.

CONTENS

Beautiful and much applied **coding theory** - efficiency and miniaturization of modern information transition systems depends much on the quality and efficiency of the underlying encoding and decoding systems.

Basic encryption systems and decryption methods of the classical, secret and public key cryptography. Blocks and streams encryption and decryption systems and methods.

Digital signatures. Authentication protocols, privacy preservation and secret sharing methods. Basics and applications of such **primitives as cryptographical hash functions, pseudorandomness, and elliptic curves.**

CONTENS

Beautiful and much applied **coding theory** - efficiency and miniaturization of modern information transition systems depends much on the quality and efficiency of the underlying encoding and decoding systems.

Basic encryption systems and decryption methods of the classical, secret and public key cryptography. Blocks and streams encryption and decryption systems and methods.

Digital signatures. Authentication protocols, privacy preservation and secret sharing methods. Basics and applications of such **primitives as cryptographical hash functions, pseudorandomness, and elliptic curves.**

Fundamental crypto protocols and zero-knowledge protocols as well as probabilistic proofs as some highlights of the fundamentals of informatics.

CONTENS

Beautiful and much applied **coding theory** - efficiency and miniaturization of modern information transition systems depends much on the quality and efficiency of the underlying encoding and decoding systems.

Basic encryption systems and decryption methods of the classical, secret and public key cryptography. Blocks and streams encryption and decryption systems and methods.

Digital signatures. Authentication protocols, privacy preservation and secret sharing methods. Basics and applications of such **primitives as cryptographical hash functions, pseudorandomness, and elliptic curves.**

Fundamental crypto protocols and zero-knowledge protocols as well as probabilistic proofs as some highlights of the fundamentals of informatics.

Steganography and watermarkinga as key information hiding and discovery methods - for a huge variety of applications.

CONTENS

Beautiful and much applied **coding theory** - efficiency and miniaturization of modern information transition systems depends much on the quality and efficiency of the underlying encoding and decoding systems.

Basic encryption systems and decryption methods of the classical, secret and public key cryptography. Blocks and streams encryption and decryption systems and methods.

Digital signatures. Authentication protocols, privacy preservation and secret sharing methods. Basics and applications of such **primitives as cryptographical hash functions, pseudorandomness, and elliptic curves.**

Fundamental crypto protocols and zero-knowledge protocols as well as probabilistic proofs as some highlights of the fundamentals of informatics.

Steganography and watermarkinga as key information hiding and discovery methods - for a huge variety of applications.

Fundamentals of quantum information transmission and Cryptography. Surprising and even shocking practical applications of quantum information transmission and cryptography. Top current cryptosystem for applications.
Comment: Concerning both lectures and homeworks the overall requirement for students will be significantly smaller than in previous years.

Codes basics and linear codes

Basics of coding theory
and
an introduction to linear codes

PROLOGUE - I.

ROSETTA SPACECRAFT

ROSETTA SPACECRAFT

- In 1993 in Europe Rosetta spacecraft project started.

ROSETTA SPACECRAFT

- In 1993 in Europe Rosetta spacecraft project started.
- In 2004 Rosetta spacecraft was launched.

ROSETTA SPACECRAFT

- In 1993 in Europe Rosetta spacecraft project started.
- In 2004 Rosetta spacecraft was launched.
- In August 2015 Rosetta spacecraft got on the orbit of the comet 67P ((4.3×4.11 of its size) one of 4000 known comets of the solar systems) and sent to earth a lot of photos of 67P.

ROSETTA SPACECRAFT

- In 1993 in Europe Rosetta spacecraft project started.
- In 2004 Rosetta spacecraft was launched.
- In August 2015 Rosetta spacecraft got on the orbit of the comet 67P ((4.3×4.11 of its size) one of 4000 known comets of the solar systems) and sent to earth a lot of photos of 67P.
- In spite of the fact that the comet 67P is 720 millions of kilometers from the earth

ROSETTA SPACECRAFT

- In 1993 in Europe Rosetta spacecraft project started.
- In 2004 Rosetta spacecraft was launched.
- In August 2015 Rosetta spacecraft got on the orbit of the comet 67P ((4.3×4.11 of its size) one of 4000 known comets of the solar systems) and sent to earth a lot of photos of 67P.
- In spite of the fact that the comet 67P is 720 millions of kilometers from the earth and there is a lot of noise for signals on the way

ROSETTA SPACECRAFT

- In 1993 in Europe Rosetta spacecraft project started.
- In 2004 Rosetta spacecraft was launched.
- In August 2015 Rosetta spacecraft got on the orbit of the comet 67P ((4.3×4.11 of its size) one of 4000 known comets of the solar systems) and sent to earth a lot of photos of 67P.
- In spite of the fact that the comet 67P is 720 millions of kilometers from the earth and there is a lot of noise for signals on the way encoding of photos arrived in such a form that they could be decoded to get excellent photos of the comet.

ROSETTA SPACECRAFT

- In 1993 in Europe Rosetta spacecraft project started.
- In 2004 Rosetta spacecraft was launched.
- In August 2015 Rosetta spacecraft got on the orbit of the comet 67P ((4.3×4.11 of its size) one of 4000 known comets of the solar systems) and sent to earth a lot of photos of 67P.
- In spite of the fact that the comet 67P is 720 millions of kilometers from the earth and there is a lot of noise for signals on the way encoding of photos arrived in such a form that they could be decoded to get excellent photos of the comet.
- All that was, to the large extent, due to the enormously high level coding theory already had in 1993.

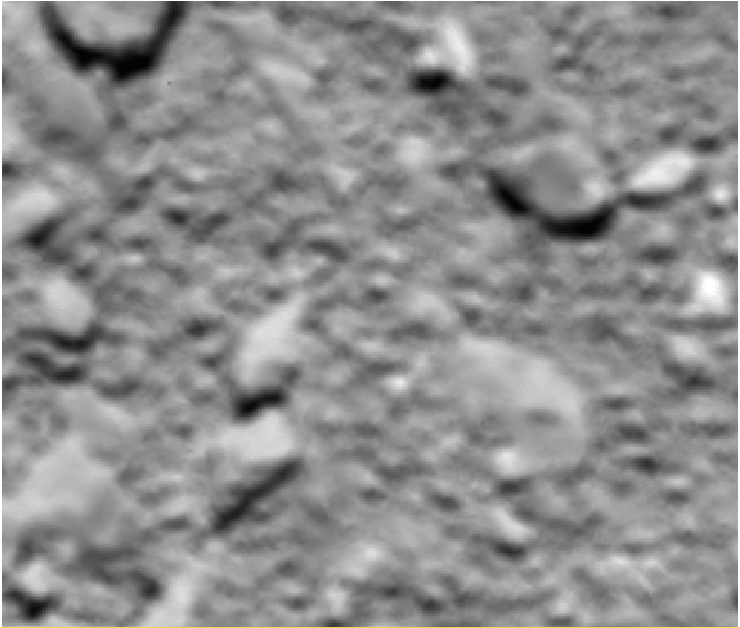
ROSETTA SPACECRAFT

- In 1993 in Europe Rosetta spacecraft project started.
- In 2004 Rosetta spacecraft was launched.
- In August 2015 Rosetta spacecraft got on the orbit of the comet 67P ((4.3×4.11 of its size) one of 4000 known comets of the solar systems) and sent to earth a lot of photos of 67P.
- In spite of the fact that the comet 67P is 720 millions of kilometers from the earth and there is a lot of noise for signals on the way encoding of photos arrived in such a form that they could be decoded to get excellent photos of the comet.
- All that was, to the large extent, due to the enormously high level coding theory already had in 1993.
- Since that time coding theory has made another enormous progress that has allowed, among other things, almost perfect mobile communication and transmission of music in time and space.

ROSETTA spacecraft

ROSETTA LANDING - VIEW from 21 km -29.9.2016

ROSETTA LANDING - VIEW from 51 m -29.9.2016



CHAPTER 1: BASICS of CODING THEORY

ABSTRACT

Coding theory - theory of error correcting codes - is one of the most interesting and applied part of informatics.

ABSTRACT

Coding theory - theory of error correcting codes - is one of the most interesting and applied part of informatics.

Goals of coding theory are to develop systems and methods that allow to detect/correct errors caused when information is transmitted through **noisy channels**.

ABSTRACT

Coding theory - theory of error correcting codes - is one of the most interesting and applied part of informatics.

Goals of coding theory are to develop systems and methods that allow to detect/correct errors caused when information is transmitted through **noisy channels**.

All real communication systems that work with digitally represented data, as CD players, TV, fax machines, internet, satellites, mobiles, require to use error correcting codes because all real channels are, to some extent, noisy – due to various interference/destruction caused by the environment

ABSTRACT

Coding theory - theory of error correcting codes - is one of the most interesting and applied part of informatics.

Goals of coding theory are to develop systems and methods that allow to detect/correct errors caused when information is transmitted through **noisy channels**.

All real communication systems that work with digitally represented data, as CD players, TV, fax machines, internet, satellites, mobiles, require to use error correcting codes because all real channels are, to some extent, noisy – due to various interference/destruction caused by the environment

- Coding theory problems are therefore among the very basic and most frequent

PROLOGUE - II.

INFORMATION

INFORMATION

is often an important and very valuable commodity.

INFORMATION

is often an important and very valuable commodity.

This lecture is about how to protect or even hide information

INFORMATION

is often an important and very valuable commodity.

This lecture is about how to protect or even hide information

against noise or even unintended user,

INFORMATION

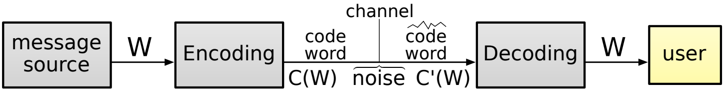
is often an important and very valuable commodity.

This lecture is about how to protect or even hide information

against noise or even unintended user,

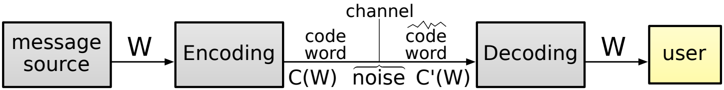
using mainly classical, but also quantum tools.

Error-correcting codes are used to correct messages when they are (erroneously) transmitted through noisy channels.



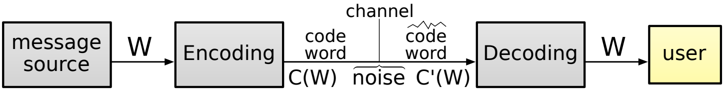
CODING THEORY - BASIC CONCEPTS

Error-correcting codes are used to correct messages when they are (erroneously) transmitted through noisy channels.



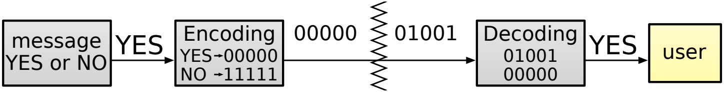
Error correcting framework

Error-correcting codes are used to correct messages when they are (erroneously) transmitted through noisy channels.

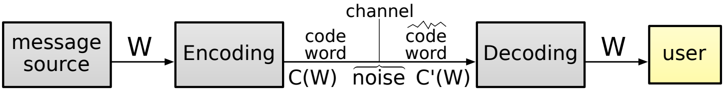


Error correcting framework

Example

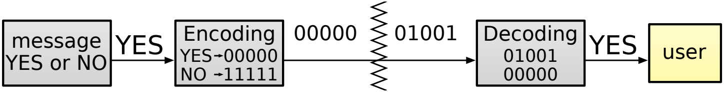


Error-correcting codes are used to correct messages when they are (erroneously) transmitted through noisy channels.



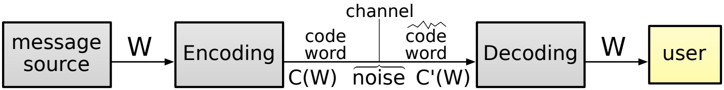
Error correcting framework

Example



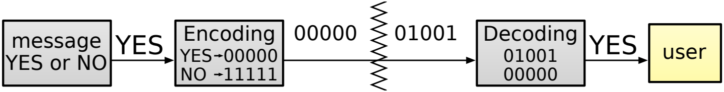
A code C over an alphabet Σ is a nonempty subset of Σ^* ($C \subseteq \Sigma^*$).

Error-correcting codes are used to correct messages when they are (erroneously) transmitted through noisy channels.



Error correcting framework

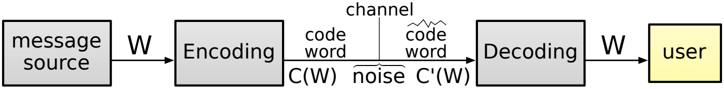
Example



A code C over an alphabet Σ is a nonempty subset of $\Sigma^*(C \subseteq \Sigma^*)$.
A q-nary code is a code over an alphabet of q-symbols.

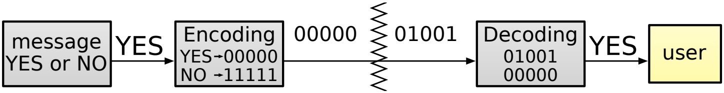
CODING THEORY - BASIC CONCEPTS

Error-correcting codes are used to correct messages when they are (erroneously) transmitted through noisy channels.



Error correcting framework

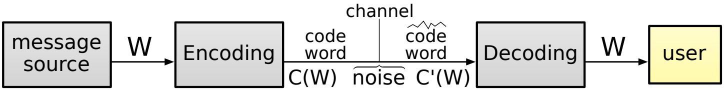
Example



A code C over an alphabet Σ is a nonempty subset of $\Sigma^*(C \subseteq \Sigma^*)$.
A q-nary code is a code over an alphabet of q-symbols.
A binary code is a code over the alphabet $\{0, 1\}$.

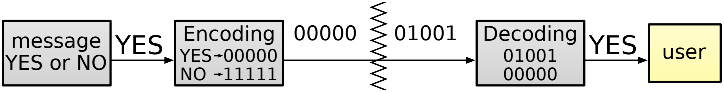
CODING THEORY - BASIC CONCEPTS

Error-correcting codes are used to correct messages when they are (erroneously) transmitted through noisy channels.



Error correcting framework

Example



A code C over an alphabet Σ is a nonempty subset of Σ^* ($C \subseteq \Sigma^*$).

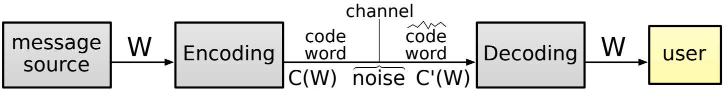
A q-nary code is a code over an alphabet of q-symbols.

A binary code is a code over the alphabet $\{0, 1\}$.

Examples of codes $C1 = \{00, 01, 10, 11\}$ $C2 = \{000, 010, 101, 100\}$

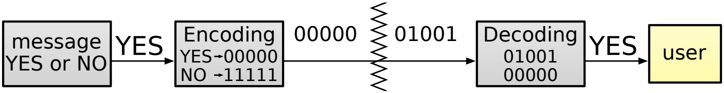
CODING THEORY - BASIC CONCEPTS

Error-correcting codes are used to correct messages when they are (erroneously) transmitted through noisy channels.



Error correcting framework

Example



A code C over an alphabet Σ is a nonempty subset of Σ^* ($C \subseteq \Sigma^*$).

A q-nary code is a code over an alphabet of q-symbols.

A binary code is a code over the alphabet $\{0, 1\}$.

Examples of codes $C1 = \{00, 01, 10, 11\}$ $C2 = \{000, 010, 101, 100\}$

$C3 = \{00000, 01101, 10111, 11011\}$

CHANNELS

CHANNELS

is any physical medium in which information is stored or through which information is transmitted.

CHANNELS

is any physical medium in which information is stored or through which information is transmitted.
(Telephone lines, optical fibres and also the atmosphere are examples of channels.)

CHANNELS

is any physical medium in which information is stored or through which information is transmitted.
(Telephone lines, optical fibres and also the atmosphere are examples of channels.)

NOISE

CHANNELS

is any physical medium in which information is stored or through which information is transmitted.
(Telephone lines, optical fibres and also the atmosphere are examples of channels.)

NOISE

may be caused by sunspots, lighting, meteor showers, random radio disturbances, poor typing, poor hearing, ...

CHANNELS

is any physical medium in which information is stored or through which information is transmitted.
(Telephone lines, optical fibres and also the atmosphere are examples of channels.)

NOISE

may be caused by sunspots, lighting, meteor showers, random radio disturbances, poor typing, poor hearing, ...

TRANSMISSION GOALS

CHANNELS

is any physical medium in which information is stored or through which information is transmitted.
(Telephone lines, optical fibres and also the atmosphere are examples of channels.)

NOISE

may be caused by sunspots, lighting, meteor showers, random radio disturbances, poor typing, poor hearing, ...

TRANSMISSION GOALS

1. Encoding of information should be very fast.

CHANNELS

is any physical medium in which information is stored or through which information is transmitted.
(Telephone lines, optical fibres and also the atmosphere are examples of channels.)

NOISE

may be caused by sunspots, lighting, meteor showers, random radio disturbances, poor typing, poor hearing, ...

TRANSMISSION GOALS

1. Encoding of information should be very fast.
2. Very similar messages should be encoded very differently.

CHANNELS

is any physical medium in which information is stored or through which information is transmitted.
(Telephone lines, optical fibres and also the atmosphere are examples of channels.)

NOISE

may be caused by sunspots, lighting, meteor showers, random radio disturbances, poor typing, poor hearing, ...

TRANSMISSION GOALS

1. Encoding of information should be very fast.
2. Very similar messages should be encoded very differently.
3. Transmission of encoded messages should be very easy.

CHANNELS

is any physical medium in which information is stored or through which information is transmitted.
(Telephone lines, optical fibres and also the atmosphere are examples of channels.)

NOISE

may be caused by sunspots, lighting, meteor showers, random radio disturbances, poor typing, poor hearing, ...

TRANSMISSION GOALS

1. Encoding of information should be very fast.
2. Very similar messages should be encoded very differently.
3. Transmission of encoded messages should be very easy.
4. Decoding of received messages should be very easy.

CHANNELS

is any physical medium in which information is stored or through which information is transmitted.
(Telephone lines, optical fibres and also the atmosphere are examples of channels.)

NOISE

may be caused by sunspots, lighting, meteor showers, random radio disturbances, poor typing, poor hearing, ...

TRANSMISSION GOALS

1. Encoding of information should be very fast.
2. Very similar messages should be encoded very differently.
3. Transmission of encoded messages should be very easy.
4. Decoding of received messages should be very easy.
5. Corection of errors introduced in the channel should be reasonably easy.

CHANNELS

is any physical medium in which information is stored or through which information is transmitted.
(Telephone lines, optical fibres and also the atmosphere are examples of channels.)

NOISE

may be caused by sunspots, lighting, meteor showers, random radio disturbances, poor typing, poor hearing, ...

TRANSMISSION GOALS

1. Encoding of information should be very fast.
2. Very similar messages should be encoded very differently.
3. Transmission of encoded messages should be very easy.
4. Decoding of received messages should be very easy.
5. Corection of errors introduced in the channel should be reasonably easy.
6. As large as possible amount of information should be transferred reliably per a time unit.

CHANNELS

is any physical medium in which information is stored or through which information is transmitted.
(Telephone lines, optical fibres and also the atmosphere are examples of channels.)

NOISE

may be caused by sunspots, lighting, meteor showers, random radio disturbances, poor typing, poor hearing, ...

TRANSMISSION GOALS

1. Encoding of information should be very fast.
2. Very similar messages should be encoded very differently.
3. Transmission of encoded messages should be very easy.
4. Decoding of received messages should be very easy.
5. Corection of errors introduced in the channel should be reasonably easy.
6. As large as possible amount of information should be transferred reliably per a time unit.

BASIC METHOD OF FIGHTING ERRORS: REDUNDANCY!!!

CHANNELS

is any physical medium in which information is stored or through which information is transmitted.
(Telephone lines, optical fibres and also the atmosphere are examples of channels.)

NOISE

may be caused by sunspots, lighting, meteor showers, random radio disturbances, poor typing, poor hearing, ...

TRANSMISSION GOALS

1. Encoding of information should be very fast.
2. Very similar messages should be encoded very differently.
3. Transmission of encoded messages should be very easy.
4. Decoding of received messages should be very easy.
5. Corection of errors introduced in the channel should be reasonably easy.
6. As large as possible amount of information should be transferred reliably per a time unit.

BASIC METHOD OF FIGHTING ERRORS: REDUNDANCY!!!

Example: 0 is encoded as 00000 and 1 is encoded as 11111.

CHANNELS - MAIN TYPES

CHANNELS - MAIN TYPES

Discrete channels and continuous channels are main types of channels.

CHANNELS - MAIN TYPES

Discrete channels and continuous channels are main types of channels.

With an example of continuous channels we will deal in Chapter 2. Main model of the noise in discrete channels is:

CHANNELS - MAIN TYPES

Discrete channels and continuous channels are main types of channels.

With an example of continuous channels we will deal in Chapter 2. Main model of the noise in discrete channels is:

- **Shannon stochastic (probabilistic) noise model: Probability $\mathbf{Pr}(y|x)$, for any output y and input x) is given that output is y in case input is x,**

CHANNELS - MAIN TYPES

Discrete channels and continuous channels are main types of channels.

With an example of continuous channels we will deal in Chapter 2. Main model of the noise in discrete channels is:

- **Shannon stochastic (probabilistic) noise model: Probability Pr**$(y|x)$**, for any output** y **and input** x**) is given that output is** y **in case input is** x**,, and in additionthe probability of too many errors is low.**

CHANNELS - MAIN TYPES

Discrete channels and continuous channels are main types of channels.

With an example of continuous channels we will deal in Chapter 2. Main model of the noise in discrete channels is:

- **Shannon stochastic (probabilistic) noise model: Probability Pr($y|x$), for any output y and input x) is given that output is y in case input is x,, and in additionthe probability of too many errors is low.**

DISCRETE CHANNELS - MATHEMATICAL VIEWS

DISCRETE CHANNELS - MATHEMATICAL VIEWS

Formally, a discrete Shannon stochastic channel is described by a triple $C = (\Sigma, \Omega, p)$, where

- Σ is an input alphabet
- Ω is an output alphabet
- Pr is a probability distribution on $\Sigma \times \Omega$ and for each $i \in \Sigma, o \in \Omega$, $Pr(i, o)$ **is the probability that the output of the channel is o if the input is i.**

DISCRETE CHANNELS - MATHEMATICAL VIEWS

Formally, a discrete Shannon stochastic channel is described by a triple $C = (\Sigma, \Omega, p)$, where

- Σ is an input alphabet
- Ω is an output alphabet
- Pr is a probability distribution on $\Sigma \times \Omega$ and for each $i \in \Sigma, o \in \Omega$, $Pr(i, o)$ **is the probability that the output of the channel is o if the input is i.**

IMPORTANT CHANNELS

DISCRETE CHANNELS - MATHEMATICAL VIEWS

Formally, a discrete Shannon stochastic channel is described by a triple $C = (\Sigma, \Omega, p)$, where

- Σ is an input alphabet
- Ω is an output alphabet
- Pr is a probability distribution on $\Sigma \times \Omega$ and for each $i \in \Sigma, o \in \Omega$, $Pr(i, o)$ **is the probability that the output of the channel is o if the input is i.**

IMPORTANT CHANNELS

- **Binary symmetric channel maps, with fixed probability p_0, each binary input into the opposite one. Hence, $\mathbf{Pr}(0,1) = \mathbf{Pr}(1,0) = p_0$ and $\mathbf{Pr}(0,0) = \mathbf{Pr}(1,1) = 1 - p_0$.**

DISCRETE CHANNELS - MATHEMATICAL VIEWS

Formally, a discrete Shannon stochastic channel is described by a triple $C = (\Sigma, \Omega, p)$, where

- Σ is an input alphabet
- Ω is an output alphabet
- Pr is a probability distribution on $\Sigma \times \Omega$ and for each $i \in \Sigma, o \in \Omega$, $Pr(i, o)$ **is the probability that the output of the channel is o if the input is i.**

IMPORTANT CHANNELS

- **Binary symmetric channel maps, with fixed probability p_0, each binary input into the opposite one. Hence, $\mathbf{Pr}(0,1) = \mathbf{Pr}(1,0) = p_0$ and $\mathbf{Pr}(0,0) = \mathbf{Pr}(1,1) = 1 - p_0$.**
- **Binary erasure channel maps, with fixed probability p_0, binary inputs into $\{0, 1, e\}$, where e is so called the erasure symbol, and $\mathbf{Pr}(0,0) = \mathbf{Pr}(1,1) = p_0$, $\mathbf{Pr}(0,e) = \mathbf{Pr}(1,e) = 1 - p_0$.**

DISCRETE CHANNELS - MATHEMATICAL VIEWS

Formally, a discrete Shannon stochastic channel is described by a triple $C = (\Sigma, \Omega, p)$, where

- Σ is an input alphabet
- Ω is an output alphabet
- Pr is a probability distribution on $\Sigma \times \Omega$ and for each $i \in \Sigma, o \in \Omega$, $Pr(i, o)$ **is the probability that the output of the channel is o if the input is i.**

IMPORTANT CHANNELS

- **Binary symmetric channel maps, with fixed probability p_0, each binary input into the opposite one. Hence, $\mathbf{Pr}(0,1) = \mathbf{Pr}(1,0) = p_0$ and $\mathbf{Pr}(0,0) = \mathbf{Pr}(1,1) = 1 - p_0$.**
- **Binary erasure channel maps, with fixed probability p_0, binary inputs into $\{0, 1, e\}$, where e is so called the erasure symbol, and $\mathbf{Pr}(0,0) = \mathbf{Pr}(1,1) = p_0$, $\mathbf{Pr}(0,e) = \mathbf{Pr}(1,e) = 1 - p_0$.**
- **White noise Gaussian channel that models errors in the deep space.**

BASIC CHANNEL CODING PROBLEMS

Summary: The task of a communication channel coding is to encode the information to be sent over the channel in such a way that even in the presence of some channel noise, several (or a specific number of) errors can be detected and/or corrected.

Summary: The task of a communication channel coding is to encode the information to be sent over the channel in such a way that even in the presence of some channel noise, several (or a specific number of) errors can be detected and/or corrected. There are two basic coding

methods

> **BEC** (Bawkwarda) Err or Cerection Coding allows the receiver only to detect errors. If an error is detected, then the sender is requested to re transmit the message.]

Summary: The task of a communication channel coding is to encode the information to be sent over the channel in such a way that even in the presence of some channel noise, several (or a specific number of) errors can be detected and/or corrected. There are two basic coding

methods

BEC (Bawkwarda) Err or Cerection Coding allows the receiver only to detect errors. If an error is detected, then the sender is requested to re transmit the message.]

DEC (Forward Error Cerection) Coding] allows the receiver to correct a certain amount of errors.

IMPORTANCE of ERROR-CORRECTING CODES for CRYPTOGRAPHY

In a good cryptosystem a change of a single bit of the cryptotext should change so many bits of the plaintext obtained from the cryptotext that the plaintext gets incomprehensible.

IMPORTANCE of ERROR-CORRECTING CODES for CRYPTOGRAPHY

In a good cryptosystem a change of a single bit of the cryptotext should change so many bits of the plaintext obtained from the cryptotext that the plaintext gets incomprehensible.

Methods to detect and correct errors when cryptotexts are transmitted are therefore much needed.

IMPORTANCE of ERROR-CORRECTING CODES for CRYPTOGRAPHY

In a good cryptosystem a change of a single bit of the cryptotext should change so many bits of the plaintext obtained from the cryptotext that the plaintext gets incomprehensible.

Methods to detect and correct errors when cryptotexts are transmitted are therefore much needed.

Also many non-cryptography applications require error-correcting codes. For example, mobiles, CD-players,. . .

WHY WE NEED TO KEEP IMPROVING ERROR-CORRECTING CODES

When error correcting capabilities of some code are improved - that is a better code is found - this has the following impacts:

WHY WE NEED TO KEEP IMPROVING ERROR-CORRECTING CODES

When error correcting capabilities of some code are improved - that is a better code is found - this has the following impacts:

- For the same quality of the received information, it is possible to achieve that the transmission system operates in more severe conditions;

WHY WE NEED TO KEEP IMPROVING ERROR-CORRECTING CODES

When error correcting capabilities of some code are improved - that is a better code is found - this has the following impacts:

- For the same quality of the received information, it is possible to achieve that the transmission system operates in more severe conditions;
- For example;
    1. It is possible to reduce the size of antennas or solar panels and the weight of batteries;

WHY WE NEED TO KEEP IMPROVING ERROR-CORRECTING CODES

When error correcting capabilities of some code are improved - that is a better code is found - this has the following impacts:

- For the same quality of the received information, it is possible to achieve that the transmission system operates in more severe conditions;
- For example;
  1. It is possible to reduce the size of antennas or solar panels and the weight of batteries;
  2. In the space travel systems such savings can be measured in hundred of thousands of dollars;

WHY WE NEED TO KEEP IMPROVING ERROR-CORRECTING CODES

When error correcting capabilities of some code are improved - that is a better code is found - this has the following impacts:

- For the same quality of the received information, it is possible to achieve that the transmission system operates in more severe conditions;
- For example;
  1. It is possible to reduce the size of antennas or solar panels and the weight of batteries;
  2. In the space travel systems such savings can be measured in hundred of thousands of dollars;
  3. In mobile telephone systems, improving the code enables the operators to increase the potential number of users in each cell.

WHY WE NEED TO KEEP IMPROVING ERROR-CORRECTING CODES

When error correcting capabilities of some code are improved - that is a better code is found - this has the following impacts:

- For the same quality of the received information, it is possible to achieve that the transmission system operates in more severe conditions;
- For example;
  1. It is possible to reduce the size of antennas or solar panels and the weight of batteries;
  2. In the space travel systems such savings can be measured in hundred of thousands of dollars;
  3. In mobile telephone systems, improving the code enables the operators to increase the potential number of users in each cell.
- Another field of applications of error-correcting codes is that of mass memories: computer hard drives, CD-Rooms, DVDs and so on.

REDUNDANCY - BASIC IDEA of ERROR CORRECTION

REDUNDANCY - BASIC IDEA of ERROR CORRECTION

Details of the techniques used to protect information against noise in practice are sometimes rather complicated, but basic principles are mostly easily understood.

REDUNDANCY - BASIC IDEA of ERROR CORRECTION

Details of the techniques used to protect information against noise in practice are sometimes rather complicated, but basic principles are mostly easily understood.

The key idea is that in order to protect a message against a noise, we should encode the message by adding some redundant informationto the message.

REDUNDANCY - BASIC IDEA of ERROR CORRECTION

Details of the techniques used to protect information against noise in practice are sometimes rather complicated, but basic principles are mostly easily understood.

The key idea is that in order to protect a message against a noise, we should encode the message by adding some redundant informationto the message.

This should be done in such a way that even if the message is corrupted by a noise, there will be enough redundancy in the encoded message to recover, or to decode the message completely.

MAJORITY VOTING DECODING - BASIC IDEA

The basic idea of so called majority voting decoding/principle or of maximal likelihood decoding/principle, when a code C is used, is

MAJORITY VOTING DECODING - BASIC IDEA

The basic idea of so called majority voting decoding/principle or of maximal likelihood decoding/principle, when a code C is used, is

to decode a received message w'

by a codeword w that is the closest codeword to w'

MAJORITY VOTING DECODING - BASIC IDEA

The basic idea of so called majority voting decoding/principle or of maximal likelihood decoding/principle, when a code C is used, is

to decode a received message w'

by a codeword w that is the closest codeword to w'

in the whole set of the codewords of the given code C.

EXAMPLE

EXAMPLE

In case: (a) the encoding

$$0 \to 000 \qquad 1 \to 111,$$

is used,

EXAMPLE

In case: (a) the encoding

$$0 \to 000 \qquad 1 \to 111,$$

is used,

(b) the probability of the bit error is $p < \frac{1}{2}$ and,

EXAMPLE

In case: (a) the encoding

$$0 \to 000 \quad 1 \to 111,$$

is used,

(b) the probability of the bit error is $p < \frac{1}{2}$ and,

(c) the following majority voting decoding

$$000, 001, 010, 100 \to 000 \quad \text{and} \quad 111, 110, 101, 011 \to 111$$

is used,

EXAMPLE

In case: (a) the encoding

$$0 \to 000 \quad 1 \to 111,$$

is used,

(b) the probability of the bit error is $p < \frac{1}{2}$ and,

(c) the following majority voting decoding

$$000, 001, 010, 100 \to 000 \quad \text{and} \quad 111, 110, 101, 011 \to 111$$

is used,

then the probability of an erroneous decoding (for the case of 2 or 3 errors) is

$$3p^2(1-p) + p^3 = 3p^2 - 2p^3 < p$$

EXAMPLE: Coding of a path avoiding an enemy territory

Story Alice and Bob share an identical map (Fig. 1) shown in Fig.1. Only Alice knows the route through which Bob can reach her avoiding the enemy (graded) territory. Alice wants to send Bob the information about the safe route he should take.

EXAMPLE: Coding of a path avoiding an enemy territory

Story Alice and Bob share an identical map (Fig. 1) shown in Fig.1. Only Alice knows the route through which Bob can reach her avoiding the enemy (graded) territory. Alice wants to send Bob the information about the safe route he should take.

TENNESSEAN

Three ways to encode the safe route (by steps North, West, South, Eat) from Bob to Alice are:

EXAMPLE: Coding of a path avoiding an enemy territory

Story Alice and Bob share an identical map (Fig. 1) shown in Fig.1. Only Alice knows the route through which Bob can reach her avoiding the enemy (graded) territory. Alice wants to send Bob the information about the safe route he should take.

TENNESSEAN

Three ways to encode the safe route (by steps North, West, South, Eat) from Bob to Alice are:

1. $C1 = \{N = 00, W = 01, S = 11, E = 10\}$

In such a case any error in the code word

000001000001011111010100000000010100

would be a disaster.

EXAMPLE: Coding of a path avoiding an enemy territory

Story Alice and Bob share an identical map (Fig. 1) shown in Fig.1. Only Alice knows the route through which Bob can reach her avoiding the enemy (graded) territory. Alice wants to send Bob the information about the safe route he should take.

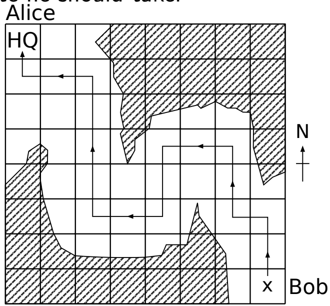


Fig. 1

TENNESSEAN

Three ways to encode the safe route (by steps North, West, South, Eat) from Bob to Alice are:

1. $C1 = \{N = 00, W = 01, S = 11, E = 10\}$

In such a case any error in the code word

0000010000010111110101000000000010100

would be a disaster.

2. $C2 = \{000, 011, 101, 110\}$

EXAMPLE: Coding of a path avoiding an enemy territory

Story Alice and Bob share an identical map (Fig. 1) shown in Fig.1. Only Alice knows the route through which Bob can reach her avoiding the enemy (graded) territory. Alice wants to send Bob the information about the safe route he should take.

TENNESSEAN

Three ways to encode the safe route (by steps North, West, South, Eat) from Bob to Alice are:

1. $C1 = \{N = 00, W = 01, S = 11, E = 10\}$

In such a case any error in the code word

000001000001011111010100000000010100

would be a disaster.

2. $C2 = \{000, 011, 101, 110\}$

A single error in encoding each of symbols N, W, S, E can be detected.

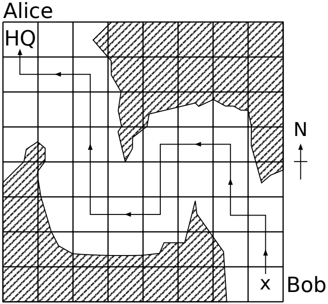


Fig. 1

EXAMPLE: Coding of a path avoiding an enemy territory

Story Alice and Bob share an identical map (Fig. 1) shown in Fig.1. Only Alice knows the route through which Bob can reach her avoiding the enemy (graded) territory. Alice wants to send Bob the information about the safe route he should take.

TENNESSEAN

Three ways to encode the safe route (by steps North, West, South, Eat) from Bob to Alice are:

1. $C1 = \{N = 00, W = 01, S = 11, E = 10\}$

In such a case any error in the code word

000001000001011111010100000000010100

would be a disaster.

2. $C2 = \{000, 011, 101, 110\}$

A single error in encoding each of symbols N, W, S, E can be detected.

3. $C3 = \{00000, 01101, 10110, 11011\}$

A single error in decoding each of symbols N, W, S, E can be corrected.

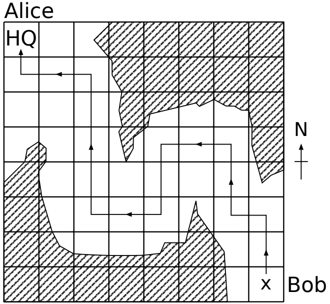


Fig. 1

BASIC TERMINOLOGY

BASIC TERMINOLOGY

Datawords - words of a message

BASIC TERMINOLOGY

Datawords - words of a message
Codewords - words of some code.

BASIC TERMINOLOGY

Datawords - words of a message
Codewords - words of some code.
Block code - a code with all codewords of the same length.

BASIC TERMINOLOGY

Datawords - words of a message
Codewords - words of some code.
Block code - a code with all codewords of the same length.

Basic assumptions about channels

BASIC TERMINOLOGY

Datawords - words of a message
Codewords - words of some code.
Block code - a code with all codewords of the same length.

Basic assumptions about channels

1. **Code length preservation.** Each output word of a channel it should have the same length as the corresponding input codeword.

BASIC TERMINOLOGY

Datawords - words of a message
Codewords - words of some code.
Block code - a code with all codewords of the same length.

Basic assumptions about channels

1. **Code length preservation.** Each output word of a channel it should have the same length as the corresponding input codeword.
2. **Independence of errors.** The probability of any one symbol being affected by an error in transmissions is the same.

BASIC TERMINOLOGY

Datawords - words of a message
Codewords - words of some code.
Block code - a code with all codewords of the same length.

Basic assumptions about channels

1. **Code length preservation.** Each output word of a channel it should have the same length as the corresponding input codeword.
2. **Independence of errors.** The probability of any one symbol being affected by an error in transmissions is the same.

Basic strategy for decoding

BASIC TERMINOLOGY

Datawords - words of a message
Codewords - words of some code.
Block code - a code with all codewords of the same length.

Basic assumptions about channels

1. **Code length preservation.** Each output word of a channel it should have the same length as the corresponding input codeword.
2. **Independence of errors.** The probability of any one symbol being affected by an error in transmissions is the same.

Basic strategy for decoding

For decoding we use the so-called **maximal likelihood principle**, or nearest neighbor decoding strategy, or **majority voting decoding strategy** which says that

BASIC TERMINOLOGY

Datawords - words of a message
Codewords - words of some code.
Block code - a code with all codewords of the same length.

Basic assumptions about channels

1. **Code length preservation.** Each output word of a channel it should have the same length as the corresponding input codeword.
2. **Independence of errors.** The probability of any one symbol being affected by an error in transmissions is the same.

Basic strategy for decoding

For decoding we use the so-called **maximal likelihood principle**, or nearest neighbor decoding strategy, or **majority voting decoding strategy** which says that

the receiver should decode a received word w'

as

the codeword w that is the closest one to w'.

HAMMING DISTANCE

The intuitive concept of "**closeness**" of two words is well formalized through **Hamming distance $h(x, y)$ of words x, y**.

HAMMING DISTANCE

The intuitive concept of “**closeness**“ of two words is well formalized through **Hamming distance** $h(x, y)$ **of words** x**,** y. For two words x, y

h(x, y) = the number of symbols in which the words x and y differ.

HAMMING DISTANCE

The intuitive concept of "**closeness**" of two words is well formalized through **Hamming distance** $h(x, y)$ **of words** x**,** y. For two words x, y

h(x, y) = the number of symbols in which the words x and y differ.

Example: $h(10101, 01100) =$

HAMMING DISTANCE

The intuitive concept of "**closeness**" of two words is well formalized through **Hamming distance** $h(x, y)$ **of words** x**,** y. For two words x, y

h(x, y) = the number of symbols in which the words x and y differ.

Example: $h(10101, 01100) = 3,$

HAMMING DISTANCE

The intuitive concept of "**closeness**" of two words is well formalized through **Hamming distance $h(x, y)$ of words x, y**. For two words x, y

h(x, y) = the number of symbols in which the words x and y differ.

Example: $h(10101, 01100) = 3$, $h(fourth, eighth) =$

HAMMING DISTANCE

The intuitive concept of "**closeness**" of two words is well formalized through **Hamming distance** $h(x, y)$ **of words** x**,** y. For two words x, y

h(x, y) = the number of symbols in which the words x and y differ.

Example: $h(10101, 01100) = 3$, $h(fourth, eighth) = 4$

HAMMING DISTANCE

The intuitive concept of "**closeness**" of two words is well formalized through **Hamming distance $h(x, y)$ of words x, y**. For two words x, y

h(x, y) = the number of symbols in which the words x and y differ.

Example: $h(10101, 01100) = 3,$ $h(fourth, eighth) = 4$

Properties of th Hamming distance

HAMMING DISTANCE

The intuitive concept of "**closeness**" of two words is well formalized through **Hamming distance $h(x, y)$ of words x, y**. For two words x, y

h(x, y) = the number of symbols in which the words x and y differ.

Example: $h(10101, 01100) = 3$, $h(fourth, eighth) = 4$

Properties of th Hamming distance

1. $h(x, y) = 0 \Leftrightarrow x = y$

HAMMING DISTANCE

The intuitive concept of "**closeness**" of two words is well formalized through **Hamming distance** $h(x, y)$ **of words** x, y. For two words x, y

h(x, y) = the number of symbols in which the words x and y differ.

Example: $h(10101, 01100) = 3$, $h(fourth, eighth) = 4$

Properties of th Hamming distance

1. $h(x, y) = 0 \Leftrightarrow x = y$
2. $h(x, y) = h(y, x)$

HAMMING DISTANCE

The intuitive concept of "**closeness**" of two words is well formalized through **Hamming distance** $h(x, y)$ **of words** x, y. For two words x, y

h(x, y) = the number of symbols in which the words x and y differ.

Example: $h(10101, 01100) = 3$, $h(fourth, eighth) = 4$

Properties of th Hamming distance

1. $h(x, y) = 0 \Leftrightarrow x = y$
2. $h(x, y) = h(y, x)$
3. $h(x, z) \leq h(x, y) + h(y, z)$ triangle inequality

HAMMING DISTANCE

The intuitive concept of "**closeness**" of two words is well formalized through **Hamming distance** $h(x, y)$ **of words** x**,** y. For two words x, y

h(x, y) = the number of symbols in which the words x and y differ.

Example: $h(10101, 01100) = 3$, $h(fourth, eighth) = 4$

Properties of th Hamming distance

1. $h(x, y) = 0 \Leftrightarrow x = y$
2. $h(x, y) = h(y, x)$
3. $h(x, z) \leq h(x, y) + h(y, z)$ triangle inequality

An important parameter of codes C is their minimal distance.

HAMMING DISTANCE

The intuitive concept of "**closeness**" of two words is well formalized through **Hamming distance $h(x,y)$ of words x, y**. For two words x, y

h(x, y) = the number of symbols in which the words x and y differ.

Example: $h(10101, 01100) = 3$, $h(fourth, eighth) = 4$

Properties of th Hamming distance

1. $h(x,y) = 0 \Leftrightarrow x = y$
2. $h(x,y) = h(y,x)$
3. $h(x,z) \leq h(x,y) + h(y,z)$ triangle inequality

An important parameter of codes C is their minimal distance.

$$h(C) = min\{h(x,y) \mid x, y \in C, x \neq y\},$$

HAMMING DISTANCE

The intuitive concept of "**closeness**" of two words is well formalized through **Hamming distance** $h(x, y)$ **of words** x**,** y. For two words x, y

h(x, y) = the number of symbols in which the words x and y differ.

Example: $h(10101, 01100) = 3,$ $h(fourth, eighth) = 4$

Properties of th Hamming distance

1. $h(x, y) = 0 \Leftrightarrow x = y$
2. $h(x, y) = h(y, x)$
3. $h(x, z) \leq h(x, y) + h(y, z)$ triangle inequality

An important parameter of codes C is their minimal distance.

$$h(C) = min\{h(x, y) \,|\, x, y \in C, x \neq y\},$$

Therefore, $h(C)$ is the smallest number of errors that can change one codeword into another.

HAMMING DISTANCE

The intuitive concept of "**closeness**" of two words is well formalized through **Hamming distance** $h(x, y)$ **of words** x**,** y. For two words x, y

h(x, y) = the number of symbols in which the words x and y differ.

Example: $h(10101, 01100) = 3,$ $h(fourth, eighth) = 4$

Properties of th Hamming distance

1. $h(x, y) = 0 \Leftrightarrow x = y$
2. $h(x, y) = h(y, x)$
3. $h(x, z) \leq h(x, y) + h(y, z)$ triangle inequality

An important parameter of codes C is their minimal distance.

$$h(C) = min\{h(x, y) \mid x, y \in C, x \neq y\},$$

Therefore, $h(C)$ is the smallest number of errors that can change one codeword into another.

Basic error correcting theorem

HAMMING DISTANCE

The intuitive concept of "**closeness**" of two words is well formalized through **Hamming distance $h(x, y)$ of words x, y**. For two words x, y

h(x, y) = the number of symbols in which the words x and y differ.

Example: $h(10101, 01100) = 3,$ $h(fourth, eighth) = 4$

Properties of th Hamming distance

1. $h(x, y) = 0 \Leftrightarrow x = y$
2. $h(x, y) = h(y, x)$
3. $h(x, z) \leq h(x, y) + h(y, z)$ triangle inequality

An important parameter of codes C is their minimal distance.

$$h(C) = min\{h(x, y) \,|\, x, y \in C, x \neq y\},$$

Therefore, $h(C)$ is the smallest number of errors that can change one codeword into another.

Basic error correcting theorem

1. A code C can detect up to s errors if $h(C) \geq s + 1$.

HAMMING DISTANCE

The intuitive concept of “**closeness**“ of two words is well formalized through **Hamming distance $h(x, y)$ of words x, y**. For two words x, y

h(x, y) = the number of symbols in which the words x and y differ.

Example: $h(10101, 01100) = 3$, $h(fourth, eighth) = 4$

Properties of th Hamming distance

1. $h(x, y) = 0 \Leftrightarrow x = y$
2. $h(x, y) = h(y, x)$
3. $h(x, z) \leq h(x, y) + h(y, z)$ triangle inequality

An important parameter of codes C is their minimal distance.

$$h(C) = min\{h(x, y) \mid x, y \in C, x \neq y\},$$

Therefore, $h(C)$ is the smallest number of errors that can change one codeword into another.

Basic error correcting theorem

1. A code C can detect up to s errors if $h(C) \geq s + 1$.
2. A code C can correct up to t errors if $h(C) \geq 2t + 1$.

HAMMING DISTANCE

The intuitive concept of "**closeness**" of two words is well formalized through **Hamming distance $h(x, y)$ of words x, y**. For two words x, y

h(x, y) = the number of symbols in which the words x and y differ.

Example: $h(10101, 01100) = 3$, $h(fourth, eighth) = 4$

Properties of th Hamming distance

1. $h(x, y) = 0 \Leftrightarrow x = y$
2. $h(x, y) = h(y, x)$
3. $h(x, z) \leq h(x, y) + h(y, z)$ triangle inequality

An important parameter of codes C is their minimal distance.

$$h(C) = min\{h(x, y) \,|\, x, y \in C, x \neq y\},$$

Therefore, $h(C)$ is the smallest number of errors that can change one codeword into another.

Basic error correcting theorem

1. A code C can detect up to s errors if $h(C) \geq s + 1$.
2. A code C can correct up to t errors if $h(C) \geq 2t + 1$.

Proof (1) Trivial.

HAMMING DISTANCE

The intuitive concept of "**closeness**" of two words is well formalized through **Hamming distance $h(x, y)$ of words x, y**. For two words x, y

h(x, y) = the number of symbols in which the words x and y differ.

Example: $h(10101, 01100) = 3$, $h(fourth, eighth) = 4$

Properties of th Hamming distance

1. $h(x, y) = 0 \Leftrightarrow x = y$
2. $h(x, y) = h(y, x)$
3. $h(x, z) \leq h(x, y) + h(y, z)$ triangle inequality

An important parameter of codes C is their minimal distance.

$$h(C) = min\{h(x, y) \,|\, x, y \in C, x \neq y\},$$

Therefore, $h(C)$ is the smallest number of errors that can change one codeword into another.

Basic error correcting theorem

1. A code C can detect up to s errors if $h(C) \geq s + 1$.
2. A code C can correct up to t errors if $h(C) \geq 2t + 1$.

Proof (1) Trivial. (2) Suppose $h(C) \geq 2t + 1$. Let a codeword x is transmitted and a word y is received such that $h(x, y) \leq t$.

HAMMING DISTANCE

The intuitive concept of "**closeness**" of two words is well formalized through **Hamming distance $h(x, y)$ of words x, y**. For two words x, y

h(x, y) = the number of symbols in which the words x and y differ.

Example: $h(10101, 01100) = 3$, $h(fourth, eighth) = 4$

Properties of th Hamming distance

1. $h(x, y) = 0 \Leftrightarrow x = y$
2. $h(x, y) = h(y, x)$
3. $h(x, z) \leq h(x, y) + h(y, z)$ triangle inequality

An important parameter of codes C is their minimal distance.

$$h(C) = min\{h(x, y) \,|\, x, y \in C, x \neq y\},$$

Therefore, $h(C)$ is the smallest number of errors that can change one codeword into another.

Basic error correcting theorem

1. A code C can detect up to s errors if $h(C) \geq s + 1$.
2. A code C can correct up to t errors if $h(C) \geq 2t + 1$.

Proof (1) Trivial. (2) Suppose $h(C) \geq 2t + 1$. Let a codeword x is transmitted and a word y is received such that $h(x, y) \leq t$. If $x' \neq x$ is any codeword, then $h(y, x') \geq t + 1$ because otherwise $h(y, x') < t + 1$ and therefore $h(x, x') \leq h(x, y) + h(y, x') < 2t + 1$

HAMMING DISTANCE

The intuitive concept of “**closeness**“ of two words is well formalized through **Hamming distance $h(x, y)$ of words x, y**. For two words x, y

h(x, y) = the number of symbols in which the words x and y differ.

Example: $h(10101, 01100) = 3,$ $h(fourth, eighth) = 4$

Properties of th Hamming distance

1. $h(x, y) = 0 \Leftrightarrow x = y$
2. $h(x, y) = h(y, x)$
3. $h(x, z) \leq h(x, y) + h(y, z)$ triangle inequality

An important parameter of codes C is their minimal distance.

$$h(C) = min\{h(x, y) \,|\, x, y \in C, x \neq y\},$$

Therefore, $h(C)$ is the smallest number of errors that can change one codeword into another.

Basic error correcting theorem

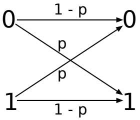
1. A code C can detect up to s errors if $h(C) \geq s + 1$.
2. A code C can correct up to t errors if $h(C) \geq 2t + 1$.

Proof (1) Trivial. (2) Suppose $h(C) \geq 2t + 1$. Let a codeword x is transmitted and a word y is received such that $h(x, y) \leq t$. If $x' \neq x$ is any codeword, then $h(y, x') \geq t + 1$ because otherwise $h(y, x') < t + 1$ and therefore $h(x, x') \leq h(x, y) + h(y, x') < 2t + 1$ what contradicts the assumption $h(C) \geq 2t + 1$.

BINARY SYMMETRIC CHANNEL

BINARY SYMMETRIC CHANNEL

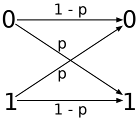
Consider a transition of binary symbols such that each symbol has probability of error $p < \frac{1}{2}$.



Binary symmetric channel

BINARY SYMMETRIC CHANNEL

Consider a transition of binary symbols such that each symbol has probability of error $p < \frac{1}{2}$.

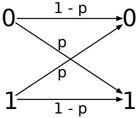


Binary symmetric channel

If n symbols are transmitted, then the probability of t errors is

BINARY SYMMETRIC CHANNEL

Consider a transition of binary symbols such that each symbol has probability of error $p < \frac{1}{2}$.



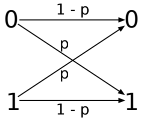
Binary symmetric channel

If n symbols are transmitted, then the probability of t errors is

$$p^t(1-p)^{n-t}\binom{n}{t}$$

BINARY SYMMETRIC CHANNEL

Consider a transition of binary symbols such that each symbol has probability of error $p < \frac{1}{2}$.



Binary symmetric channel

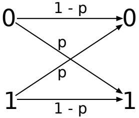
If n symbols are transmitted, then the probability of t errors is

$$p^t(1-p)^{n-t}\binom{n}{t}$$

In the case of binary symmetric channels, the "nearest neighbour decoding strategy" is also "maximum likelihood decoding strategy".

BINARY SYMMETRIC CHANNEL

Consider a transition of binary symbols such that each symbol has probability of error $p < \frac{1}{2}$.



Binary symmetric channel

If n symbols are transmitted, then the probability of t errors is

$$p^t(1-p)^{n-t}\binom{n}{t}$$

In the case of binary symmetric channels, the "nearest neighbour decoding strategy" is also "maximum likelihood decoding strategy".

SURPRISING POWER of PARITY BITS

SURPRISING POWER of PARITY BITS

Example Let all 2^{11} of binary words of length 11 be codewords

SURPRISING POWER of PARITY BITS

Example Let all 2^{11} of binary words of length 11 be codewords and let the probability of a bit error be $p = 10^{-8}$.

SURPRISING POWER of PARITY BITS

Example Let all 2^{11} of binary words of length 11 be codewords and let the probability of a bit error be $p = 10^{-8}$.
Let bits be transmitted at the rate 10^7 bits per second.

SURPRISING POWER of PARITY BITS

Example Let all 2^{11} of binary words of length 11 be codewords
and let the probability of a bit error be $p = 10^{-8}$.
Let bits be transmitted at the rate 10^7 bits per second.
The probability that a word is transmitted incorrectly is approximately

$$11p(1-p)^{10} \approx \frac{11}{10^8}.$$

SURPRISING POWER of PARITY BITS

Example Let all 2^{11} of binary words of length 11 be codewords and let the probability of a bit error be $p = 10^{-8}$.
Let bits be transmitted at the rate 10^7 bits per second.
The probability that a word is transmitted incorrectly is approximately

$$11p(1-p)^{10} \approx \frac{11}{10^8}.$$

Therefore $\frac{11}{10^8} \cdot \frac{10^7}{11} = 0.1$ of words per second are transmitted incorrectly.

SURPRISING POWER of PARITY BITS

Example Let all 2^{11} of binary words of length 11 be codewords
and let the probability of a bit error be $p = 10^{-8}$.
Let bits be transmitted at the rate 10^7 bits per second.
The probability that a word is transmitted incorrectly is approximately

$$11p(1-p)^{10} \approx \frac{11}{10^8}.$$

Therefore $\frac{11}{10^8} \cdot \frac{10^7}{11} = 0.1$ of words per second are transmitted incorrectly.
Therefore, one wrong word is transmitted every 10 seconds, 360 erroneous words every hour and 8640 words every day without being detected!

SURPRISING POWER of PARITY BITS

Example Let all 2^{11} of binary words of length 11 be codewords and let the probability of a bit error be $p = 10^{-8}$.
Let bits be transmitted at the rate 10^7 bits per second.
The probability that a word is transmitted incorrectly is approximately

$$11p(1-p)^{10} \approx \frac{11}{10^8}.$$

Therefore $\frac{11}{10^8} \cdot \frac{10^7}{11} = 0.1$ of words per second are transmitted incorrectly.
Therefore, one wrong word is transmitted every 10 seconds, 360 erroneous words every hour and 8640 words every day without being detected!
Let now one parity bit be added.

SURPRISING POWER of PARITY BITS

Example Let all 2^{11} of binary words of length 11 be codewords
and let the probability of a bit error be $p = 10^{-8}$.
Let bits be transmitted at the rate 10^7 bits per second.
The probability that a word is transmitted incorrectly is approximately

$$11p(1-p)^{10} \approx \frac{11}{10^8}.$$

Therefore $\frac{11}{10^8} \cdot \frac{10^7}{11} = 0.1$ of words per second are transmitted incorrectly.
Therefore, one wrong word is transmitted every 10 seconds, 360 erroneous words every hour and 8640 words every day without being detected!
Let now one parity bit be added.
Any single error can be detected!!!

SURPRISING POWER of PARITY BITS

Example Let all 2^{11} of binary words of length 11 be codewords
and let the probability of a bit error be $p = 10^{-8}$.
Let bits be transmitted at the rate 10^7 bits per second.
The probability that a word is transmitted incorrectly is approximately

$$11p(1-p)^{10} \approx \tfrac{11}{10^8}.$$

Therefore $\frac{11}{10^8} \cdot \frac{10^7}{11} = 0.1$ of words per second are transmitted incorrectly.
Therefore, one wrong word is transmitted every 10 seconds, 360 erroneous words every hour and 8640 words every day without being detected!
Let now one parity bit be added.
Any single error can be detected!!!
The probability of at least two errors is:

$$1-(1-p)^{12} - 12(1-p)^{11}p \approx \binom{12}{2}(1-p)^{10}p^2 \approx \tfrac{66}{10^{16}}$$

SURPRISING POWER of PARITY BITS

Example Let all 2^{11} of binary words of length 11 be codewords
and let the probability of a bit error be $p = 10^{-8}$.
Let bits be transmitted at the rate 10^7 bits per second.
The probability that a word is transmitted incorrectly is approximately

$$11p(1-p)^{10} \approx \tfrac{11}{10^8}.$$

Therefore $\frac{11}{10^8} \cdot \frac{10^7}{11} = 0.1$ of words per second are transmitted incorrectly.
Therefore, one wrong word is transmitted every 10 seconds, 360 erroneous words every hour and 8640 words every day without being detected!
Let now one parity bit be added.
Any single error can be detected!!!
The probability of at least two errors is:

$$1-(1-p)^{12} - 12(1-p)^{11}p \approx \binom{12}{2}(1-p)^{10}p^2 \approx \tfrac{66}{10^{16}}$$

Therefore, approximately $\frac{66}{10^{16}} \cdot \frac{10^7}{12} \approx 5.5 \cdot 10^{-9}$ words per second are transmitted with an undetectable error.

SURPRISING POWER of PARITY BITS

Example Let all 2^{11} of binary words of length 11 be codewords
and let the probability of a bit error be $p = 10^{-8}$.
Let bits be transmitted at the rate 10^7 bits per second.
The probability that a word is transmitted incorrectly is approximately

$$11p(1-p)^{10} \approx \frac{11}{10^8}.$$

Therefore $\frac{11}{10^8} \cdot \frac{10^7}{11} = 0.1$ of words per second are transmitted incorrectly.
Therefore, one wrong word is transmitted every 10 seconds, 360 erroneous words every hour and 8640 words every day without being detected!
Let now one parity bit be added.
Any single error can be detected!!!
The probability of at least two errors is:

$$1-(1-p)^{12} - 12(1-p)^{11}p \approx \binom{12}{2}(1-p)^{10}p^2 \approx \frac{66}{10^{16}}$$

Therefore, approximately $\frac{66}{10^{16}} \cdot \frac{10^7}{12} \approx 5.5 \cdot 10^{-9}$ words per second are transmitted with an undetectable error.
Corollary One undetected error occurs only once every 2000 days! ($2000 \approx \frac{10^9}{5.5 \times 86400}$).

NOTATIONS and EXAMPLES

Notation: An (n, M, d)-code C is a code such that

NOTATIONS and EXAMPLES

Notation: An (n, M, d)-code C is a code such that

- n - is the length of codewords.
- M - is the number of codewords.
- d - is the minimum distance of two codewords in C.

NOTATIONS and EXAMPLES

Notation: An (n, M, d)-code C is a code such that

- n - is the length of codewords.
- M - is the number of codewords.
- d - is the minimum distance of two codewords in C.

Example:
$C1 = \{00, 01, 10, 11\}$ is a (2,4,1)-code.

NOTATIONS and EXAMPLES

Notation: An (n, M, d)-code C is a code such that

- n - is the length of codewords.
- M - is the number of codewords.
- d - is the minimum distance of two codewords in C.

Example:
$C1 = \{00, 01, 10, 11\}$ is a (2,4,1)-code.
$C2 = \{000, 011, 101, 110\}$ is a (3,4,2)-code.

NOTATIONS and EXAMPLES

Notation: An (n, M, d)-code C is a code such that

- n - is the length of codewords.
- M - is the number of codewords.
- d - is the minimum distance of two codewords in C.

Example:

$C1 = \{00, 01, 10, 11\}$ is a (2,4,1)-code.
$C2 = \{000, 011, 101, 110\}$ is a (3,4,2)-code.
$C3 = \{00000, 01101, 10110, 11011\}$ is a (5,4,3)-code.

NOTATIONS and EXAMPLES

Notation: An (n, M, d)-code C is a code such that

- n - is the length of codewords.
- M - is the number of codewords.
- d - is the minimum distance of two codewords in C.

Example:
$C1 = \{00, 01, 10, 11\}$ is a (2,4,1)-code.
$C2 = \{000, 011, 101, 110\}$ is a (3,4,2)-code.
$C3 = \{00000, 01101, 10110, 11011\}$ is a (5,4,3)-code.

Comment: A good (n, M, d)-code has small n, large M and also large d.

EXAMPLES from DEEP SPACE TRAVELS

EXAMPLES from DEEP SPACE TRAVELS

Examples (Transmission of photographs from the deep space)

EXAMPLES from DEEP SPACE TRAVELS

Examples (Transmission of photographs from the deep space)

- In 1965-69 **Mariner 4-5** probes took the first photographs of another planet - 22 photos. Each photo was divided into 200×200 elementary squares - pixels. Each pixel was assigned 6 bits representing 64 levels of brightness. and so called Hadamard code was used.

EXAMPLES from DEEP SPACE TRAVELS

Examples (Transmission of photographs from the deep space)

- In 1965-69 **Mariner 4-5** probes took the first photographs of another planet - 22 photos. Each photo was divided into 200×200 elementary squares - pixels. Each pixel was assigned 6 bits representing 64 levels of brightness. and so called Hadamard code was used.

  Transmission rate was 8.3 bits per second.

EXAMPLES from DEEP SPACE TRAVELS

Examples (Transmission of photographs from the deep space)

- In 1965-69 **Mariner 4-5** probes took the first photographs of another planet - 22 photos. Each photo was divided into 200×200 elementary squares - pixels. Each pixel was assigned 6 bits representing 64 levels of brightness. and so called Hadamard code was used.

  Transmission rate was 8.3 bits per second.

- In 1970-72 **Mariners 6-8** took such photographs that each picture was broken into 700×832 squares. So called Reed-Muller (32,64,16) code was used.

EXAMPLES from DEEP SPACE TRAVELS

Examples (Transmission of photographs from the deep space)

- In 1965-69 **Mariner 4-5** probes took the first photographs of another planet - 22 photos. Each photo was divided into 200×200 elementary squares - pixels. Each pixel was assigned 6 bits representing 64 levels of brightness. and so called Hadamard code was used.

  Transmission rate was 8.3 bits per second.

- In 1970-72 **Mariners 6-8** took such photographs that each picture was broken into 700×832 squares. So called Reed-Muller (32,64,16) code was used.

  Transmission rate was 16200 bits per second. (Much better quality pictures could be received)

HADAMARD CODE

In Mariner 5, 6-bit pixels were encoded using 32-bit long Hadamard code that could correct up to 7 errors.

HADAMARD CODE

In Mariner 5, 6-bit pixels were encoded using 32-bit long Hadamard code that could correct up to 7 errors.

Hadamard code has 64 codewords. 32 of them are represented by the 32×32 matrix $H = \{h_{IJ}\}$, where $0 \leq i, j \leq 31$ and

$$h_{ij} = (-1)^{a_0 b_0 + a_1 b_1 + \ldots + a_4 b_4}$$

HADAMARD CODE

In Mariner 5, 6-bit pixels were encoded using 32-bit long Hadamard code that could correct up to 7 errors.

Hadamard code has 64 codewords. 32 of them are represented by the 32 $\times$ 32 matrix $H = \{h_{IJ}\}$, where $0 \leq i, j \leq 31$ and

$$h_{ij} = (-1)^{a_0 b_0 + a_1 b_1 + \ldots + a_4 b_4}$$

where i and j have binary representations

$$i = a_4 a_3 a_2 a_1 a_0, \; j = b_4 b_3 b_2 b_1 b_0$$

HADAMARD CODE

In Mariner 5, 6-bit pixels were encoded using 32-bit long Hadamard code that could correct up to 7 errors.

Hadamard code has 64 codewords. 32 of them are represented by the 32 $\times$ 32 matrix $H = \{h_{IJ}\}$, where $0 \leq i, j \leq 31$ and

$$h_{ij} = (-1)^{a_0 b_0 + a_1 b_1 + \ldots + a_4 b_4}$$

where i and j have binary representations

$$i = a_4 a_3 a_2 a_1 a_0, \; j = b_4 b_3 b_2 b_1 b_0$$

The remaining 32 codewords are represented by the matrix $-H$.

HADAMARD CODE

In Mariner 5, 6-bit pixels were encoded using 32-bit long Hadamard code that could correct up to 7 errors.

Hadamard code has 64 codewords. 32 of them are represented by the 32×32 matrix $H = \{h_{IJ}\}$, where $0 \leq i, j \leq 31$ and

$$h_{ij} = (-1)^{a_0 b_0 + a_1 b_1 + \ldots + a_4 b_4}$$

where i and j have binary representations

$$i = a_4 a_3 a_2 a_1 a_0,\ j = b_4 b_3 b_2 b_1 b_0$$

The remaining 32 codewords are represented by the matrix $-H$.
Decoding was quite simple.

CODES RATES

For q-nary (n, M, d)-code C we define the code rate, or information rate, R_C, by

$$R_C = \frac{lg_Q M}{n}.$$

CODES RATES

For q-nary (n, M, d)-code C we define the code rate, or information rate, R_C, by

$$R_C = \frac{lg_Q M}{n}.$$

The code rate represents the ratio of the number of needed input data symbols to the number of transmitted code symbols.

CODES RATES

For q-nary (n, M, d)-code C we define the code rate, or information rate, R_C, by

$$R_C = \frac{lg_Q M}{n}.$$

The code rate represents the ratio of the number of needed input data symbols to the number of transmitted code symbols.

If a q-nary code has code rate R, then we say that it transmits R q-symbols per a channel use - or R is a number of bits per a channel use (box) - in the case of binary alphabet.

CODES RATES

For q-nary (n, M, d)-code C we define the code rate, or information rate, R_C, by

$$R_C = \frac{lg_Q M}{n}.$$

The code rate represents the ratio of the number of needed input data symbols to the number of transmitted code symbols.

If a q-nary code has code rate R, then we say that it transmits R q-symbols per a channel use - or R is a number of bits per a channel use (box) - in the case of binary alphabet.

Code rate (6/32 for Hadamard code), is an important parameter for real implementations, because it shows what fraction of the communication bandwidth is being used to transmit actual data.

The ISBN-code I.

Each book till 1.1.2007 had Iinternational Sstandard BOOK Number which was a 10-digit codeword produced by the publisher with the following structure:

The ISBN-code I.

Each book till 1.1.2007 had Iinternational Sstandard BOOK Number which was a 10-digit codeword produced by the publisher with the following structure:

l	p	m	w	$= x_1 \ldots x_{10}$
language	publisher	number	weighted check sum	
0	07	709503	0	

such that

The ISBN-code I.

Each book till 1.1.2007 had Iinternational Sstandard BOOK Number which was a 10-digit codeword produced by the publisher with the following structure:

l	p	m	w	$= x_1 \dots x_{10}$
language	publisher	number	weighted check sum	
0	07	709503	0	

such that $\sum_{i=1}^{10}(11-i)x_i \equiv 0 \ (mod 11)$

The publisher has to put $x_{10} = X$ if x_{10} is to be 10.

The ISBN-code I.

Each book till 1.1.2007 had Iinternational Sstandard BOOK Number which was a 10-digit codeword produced by the publisher with the following structure:

l	p	m	w	$= x_1 \dots x_{10}$
language	publisher	number	weighted check sum	
0	07	709503	0	

such that $\sum_{i=1}^{10}(11-i)x_i \equiv 0\ (mod 11)$

The publisher has to put $x_{10} = X$ if x_{10} is to be 10.
The ISBN code was designed to detect: (a) any single error (b) any double error created by a transposition

EQUIVALENCE of CODES

EQUIVALENCE of CODES

Definition Two q-ary codes are called equivalent if one can be obtained from the other by a combination of operations of the following type:

EQUIVALENCE of CODES

Definition Two q-ary codes are called equivalent if one can be obtained from the other by a combination of operations of the following type:

(a) a permutation of the positions of the code.

EQUIVALENCE of CODES

Definition Two q-ary codes are called equivalent if one can be obtained from the other by a combination of operations of the following type:

(a a permutation of the positions of the code.

(b a permutation of symbols appearing in a fixed position.

EQUIVALENCE of CODES

Definition Two q-ary codes are called equivalent if one can be obtained from the other by a combination of operations of the following type:

(a) a permutation of the positions of the code.

(b) a permutation of symbols appearing in a fixed position.

Question: Let a code be displayed as an M $\times$ n matrix. To what correspond operations (a) and (b)?

EQUIVALENCE of CODES

Definition Two q-ary codes are called equivalent if one can be obtained from the other by a combination of operations of the following type:

- (a) a permutation of the positions of the code.
- (b) a permutation of symbols appearing in a fixed position.

Question: Let a code be displayed as an $M \times n$ matrix. To what correspond operations (a) and (b)?

Claim: Distances between codewords are unchanged by operations (a), (b).

Consequently, equivalent codes have the same parameters (n,M,d) (and correct the same number of errors).

EQUIVALENCE of CODES

Definition Two q-ary codes are called equivalent if one can be obtained from the other by a combination of operations of the following type:

- (a a permutation of the positions of the code.
- (b a permutation of symbols appearing in a fixed position.

Question: Let a code be displayed as an M $\times$ n matrix. To what correspond operations (a) and (b)?

Claim: Distances between codewords are unchanged by operations (a), (b). Consequently, equivalent codes have the same parameters (n,M,d) (and correct the same number of errors).

Examples of equivalent codes

$$(1) \begin{Bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \end{Bmatrix} \begin{Bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \end{Bmatrix}$$

EQUIVALENCE of CODES

Definition Two q-ary codes are called equivalent if one can be obtained from the other by a combination of operations of the following type:

- (a a permutation of the positions of the code.
- (b a permutation of symbols appearing in a fixed position.

Question: Let a code be displayed as an M $\times$ n matrix. To what correspond operations (a) and (b)?

Claim: Distances between codewords are unchanged by operations (a), (b).

Consequently, equivalent codes have the same parameters (n,M,d) (and correct the same number of errors).

Examples of equivalent codes

$$(1)\ \begin{Bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \end{Bmatrix} \begin{Bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \end{Bmatrix}$$

$$(2)\ \begin{Bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 2 & 2 & 2 \end{Bmatrix} \begin{Bmatrix} 0 & 1 & 2 \\ 1 & 2 & 0 \\ 2 & 0 & 1 \end{Bmatrix}$$

EQUIVALENCE of CODES

Definition Two q-ary codes are called equivalent if one can be obtained from the other by a combination of operations of the following type:

- (a) a permutation of the positions of the code.
- (b) a permutation of symbols appearing in a fixed position.

Question: Let a code be displayed as an M $\times$ n matrix. To what correspond operations (a) and (b)?

Claim: Distances between codewords are unchanged by operations (a), (b). Consequently, equivalent codes have the same parameters (n,M,d) (and correct the same number of errors).

Examples of equivalent codes

$$(1)\ \begin{Bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 \end{Bmatrix} \begin{Bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \end{Bmatrix}$$

$$(2)\ \begin{Bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 2 & 2 & 2 \end{Bmatrix} \begin{Bmatrix} 0 & 1 & 2 \\ 1 & 2 & 0 \\ 2 & 0 & 1 \end{Bmatrix}$$

Lemma Any q-ary (n, M, d)-code over an alphabet $\{0, 1, \ldots, q-1\}$ is equivalent to an (n, M, d)-code which contains the all-zero codeword $00\ldots0$.

Proof Trivial.

THE MAIN CODING THEORY PROBLEM

A good (n, M, d)-code should have a small n, large M and large d.

THE MAIN CODING THEORY PROBLEM

A good (n, M, d)-code should have a small n, large M and large d.

The main coding theory problem is to optimize one of the parameters n, M, d for given values of the other two.

THE MAIN CODING THEORY PROBLEM

A good (n, M, d)-code should have a small n, large M and large d.

The main coding theory problem is to optimize one of the parameters n, M, d for given values of the other two.

Notation: $A_q(n, d)$ is the largest M such that there is an q-nary (n, M, d)-code.

THE MAIN CODING THEORY PROBLEM

A good (n, M, d)-code should have a small n, large M and large d.

The main coding theory problem is to optimize one of the parameters n, M, d for given values of the other two.

Notation: $A_q(n, d)$ is the largest M such that there is an q-nary (n, M, d)-code.

Theorem

(a) $A_q(n, 1) = q^n$;

(b) $A_q(n, n) = q$.

THE MAIN CODING THEORY PROBLEM

A good (n, M, d)-code should have a small n, large M and large d.

The main coding theory problem is to optimize one of the parameters n, M, d for given values of the other two.

Notation: $A_q(n, d)$ is the largest M such that there is an q-nary (n, M, d)-code.

Theorem

(a) $A_q(n, 1) = q^n$;

(b) $A_q(n, n) = q$.

Proof

THE MAIN CODING THEORY PROBLEM

A good (n, M, d)-code should have a small n, large M and large d.

The main coding theory problem is to optimize one of the parameters n, M, d for given values of the other two.

Notation: $A_q(n, d)$ is the largest M such that there is an q-nary (n, M, d)-code.

Theorem

(a $A_q(n, 1) = q^n$;

(b $A_q(n, n) = q$.

Proof

(a First claim is obvious;

THE MAIN CODING THEORY PROBLEM

A good (n, M, d)-code should have a small n, large M and large d.

The main coding theory problem is to optimize one of the parameters n, M, d for given values of the other two.

Notation: $A_q(n, d)$ is the largest M such that there is an q-nary (n, M, d)-code.

Theorem

(a) $A_q(n, 1) = q^n$;

(b) $A_q(n, n) = q$.

Proof

(a) First claim is obvious;

(b) Let C be an q-nary (n, M, n)-code.

THE MAIN CODING THEORY PROBLEM

A good (n, M, d)-code should have a small n, large M and large d.

The main coding theory problem is to optimize one of the parameters n, M, d for given values of the other two.

Notation: $A_q(n, d)$ is the largest M such that there is an q-nary (n, M, d)-code.

Theorem

(a) $A_q(n, 1) = q^n$;

(b) $A_q(n, n) = q$.

Proof

(a) First claim is obvious;

(b) Let C be an q-nary (n, M, n)-code. Any two distinct codewords of C have to differ in all n positions.

THE MAIN CODING THEORY PROBLEM

A good (n, M, d)-code should have a small n, large M and large d.

The main coding theory problem is to optimize one of the parameters n, M, d for given values of the other two.

Notation: $A_q(n, d)$ is the largest M such that there is an q-nary (n, M, d)-code.

Theorem

(a $A_q(n, 1) = q^n$;

(b $A_q(n, n) = q$.

Proof

(a First claim is obvious;

(b Let C be an q-nary (n, M, n)-code. Any two distinct codewords of C have to differ in all n positions. Hence symbols in any fixed position of M codewords have to be different.

THE MAIN CODING THEORY PROBLEM

A good (n, M, d)-code should have a small n, large M and large d.

The main coding theory problem is to optimize one of the parameters n, M, d for given values of the other two.

Notation: $A_q(n, d)$ is the largest M such that there is an q-nary (n, M, d)-code.

Theorem

(a $A_q(n, 1) = q^n$;

(b $A_q(n, n) = q$.

Proof

(a First claim is obvious;

(b Let C be an q-nary (n, M, n)-code. Any two distinct codewords of C have to differ in all n positions. Hence symbols in any fixed position of M codewords have to be different. Therefore $\Rightarrow A_q(n, n) \leq q$.

THE MAIN CODING THEORY PROBLEM

A good (n, M, d)-code should have a small n, large M and large d.

The main coding theory problem is to optimize one of the parameters n, M, d for given values of the other two.

Notation: $A_q(n, d)$ is the largest M such that there is an q-nary (n, M, d)-code.

Theorem

(a $A_q(n, 1) = q^n$;

(b $A_q(n, n) = q$.

Proof

(a First claim is obvious;

(b Let C be an q-nary (n, M, n)-code. Any two distinct codewords of C have to differ in all n positions. Hence symbols in any fixed position of M codewords have to be different. Therefore $\Rightarrow A_q(n, n) \leq q$. Since the q-nary repetition code is (n, q, n)-code, we get $A_q(n, n) \geq q$.

A SPHERE and its VOLUME

Notation F_q^n - is a set of all words of length n over the alphabet $\{0, 1, 2, \ldots, q-1\}$

A SPHERE and its VOLUME

Notation F_q^n - is a set of all words of length n over the alphabet $\{0, 1, 2, \ldots, q-1\}$
Definition For any codeword $u \in F_q^n$ and any integer $r \geq 0$ the sphere of radius r and centre u is denoted by

$$S(u, r) = \{v \in F_q^n \mid h(u, v) \leq r\}.$$

A SPHERE and its VOLUME

Notation F_q^n - is a set of all words of length n over the alphabet $\{0, 1, 2, \ldots, q-1\}$
Definition For any codeword $u \in F_q^n$ and any integer $r \geq 0$ the sphere of radius r and centre u is denoted by

$$S(u, r) = \{v \in F_q^n \mid h(u, v) \leq r\}.$$

Theorem A sphere of radius r in F_q^n, $0 \leq r \leq n$ contains

$$\binom{n}{0} + \binom{n}{1}(q-1) + \binom{n}{2}(q-1)^2 + \ldots + \binom{n}{r}(q-1)^r$$

words.

A SPHERE and its VOLUME

Notation F_q^n - is a set of all words of length n over the alphabet $\{0, 1, 2, \ldots, q-1\}$
Definition For any codeword $u \in F_q^n$ and any integer $r \geq 0$ the sphere of radius r and centre u is denoted by

$$S(u, r) = \{v \in F_q^n \mid h(u, v) \leq r\}.$$

Theorem A sphere of radius r in F_q^n, $0 \leq r \leq n$ contains

$$\binom{n}{0} + \binom{n}{1}(q-1) + \binom{n}{2}(q-1)^2 + \ldots + \binom{n}{r}(q-1)^r$$

words.

Proof Let u be a fixed word in F_q^n. The number of words that differ from u in m positions is

$$\binom{n}{m}(q-1)^m.$$

PICTURES of SATURN TAKEN by VOYAGER

PICTURES of SATURN TAKEN by VOYAGER

Pictures of Saturn taken by Voyager, in 1980, had 800×800 pixels with 8 levels of brightness.

Pictures of Saturn taken by Voyager, in 1980, had 800×800 pixels with 8 levels of brightness.

Since pictures were in color, each picture was transmitted three times; each time through different color filter. The full color picture was represented by

$$3 \times 800 \times 800 \times 8 = 13360000 \text{ bits.}$$

Pictures of Saturn taken by Voyager, in 1980, had 800×800 pixels with 8 levels of brightness.

Since pictures were in color, each picture was transmitted three times; each time through different color filter. The full color picture was represented by

$$3 \times 800 \times 800 \times 8 = 13360000 \text{ bits.}$$

To transmit pictures Voyager used the so called Golay code G_{24}.

FRAMEWORK

FRAMEWORK

The goal of coding theory is to develop for a given set of **messages** M,

FRAMEWORK

The goal of coding theory is to develop for a given set of **messages** M,

for example for the set of names of students/participants of this crypto lecture,

FRAMEWORK

The goal of coding theory is to develop for a given set of **messages** M,

for example for the set of names of students/participants of this crypto lecture,

a **code** - a set of **codewords**,

FRAMEWORK

The goal of coding theory is to develop for a given set of **messages** M,

for example for the set of names of students/participants of this crypto lecture,

a **code** - a set of **codewords**,

for example **UČO**s

FRAMEWORK

The goal of coding theory is to develop for a given set of **messages** M,

for example for the set of names of students/participants of this crypto lecture,

a **code** - a set of **codewords**,

for example **UČO**s

and to send through a noisy Chanel UČO of students instead of their names,

FRAMEWORK

The goal of coding theory is to develop for a given set of **messages** M,

for example for the set of names of students/participants of this crypto lecture,

a **code** - a set of **codewords**,

for example **UČO**s

and to send through a noisy Chanel UČO of students instead of their names,
in such a way that what will be received can be used to determine name that had to be transmitted

AN IDEA - I.

Let us assume that UČO of each student can be seen as its encoding.

AN IDEA - I.

Let us assume that UČO of each student can be seen as its encoding.

Is it possible to give to each student in this class an UČO in such a way that the sum of UČos of any two student of this class will be again an UČO of some student of this class?

AN IDEA - I.

Let us assume that UČO of each student can be seen as its encoding.

Is it possible to give to each student in this class an UČO in such a way that the sum of UČos of any two student of this class will be again an UČO of some student of this class?

The answer is NO and the proof of that is almost trivial.

AN IDEA - I.

Let us assume that UČO of each student can be seen as its encoding.

Is it possible to give to each student in this class an UČO in such a way that the sum of UČos of any two student of this class will be again an UČO of some student of this class?

The answer is NO and the proof of that is almost trivial. Is it possible to give to each

student UČO in such a way that bit-wise sums of binary representations of UČos of any two student of this class will be again binary representations of UČos of some students of this class?

AN IDEA - I.

Let us assume that UČO of each student can be seen as its encoding.

Is it possible to give to each student in this class an UČO in such a way that the sum of UČos of any two student of this class will be again an UČO of some student of this class?

The answer is NO and the proof of that is almost trivial. Is it possible to give to each

student UČO in such a way that bit-wise sums of binary representations of UČos of any two student of this class will be again binary representations of UČos of some students of this class?

In general, does it has a sense to look for such codes that some important sum of any two codewords is again a codeword?

LINEAR CODES - I.

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n, q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n, q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

Definition A subset $C \subseteq F_q^n$ is a linear code if

1. $u + v \in C$ for all $u, v \in C$

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n,q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

Definition A subset $C \subseteq F_q^n$ is a linear code if

1. $u + v \in C$ for all $u, v \in C$
   (if $u = (u_1, u_2, \ldots, u_n), v = (v_1, v_2. \ldots, v_n)$ then
   $u + v = (u_1 +_W V_1, u_2 +_W V_2 \ldots, u_n +_W V_n))$

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n,q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

Definition A subset $C \subseteq F_q^n$ is a linear code if

1. $u + v \in C$ for all $u, v \in C$
   (if $u = (u_1, u_2, \ldots, u_n), v = (v_1, v_2 \ldots, v_n)$ then
   $u + v = (u_1 +_W V_1, u_2 +_W V_2 \ldots, u_n +_W V_n))$
2. $au \in C$ for all $u \in C$, and all $a \in GF(q)$

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n,q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

Definition A subset $C \subseteq F_q^n$ is a linear code if

1. $u + v \in C$ for all $u, v \in C$
   (if $u = (u_1, u_2, \ldots, u_n), v = (v_1, v_2 \ldots, v_n)$ then
   $u + v = (u_1 +_W V_1, u_2 +_W V_2 \ldots, u_n +_W V_n)$)
2. $au \in C$ for all $u \in C$, and all $a \in GF(q)$
   if $u = (u_1, u_2, \ldots, u_n)$,, then $au = (au_1, au_2, \ldots, au_n)$)

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n, q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

Definition A subset $C \subseteq F_q^n$ is a linear code if

1. $u + v \in C$ for all $u, v \in C$
   (if $u = (u_1, u_2, \ldots, u_n), v = (v_1, v_2 \ldots, v_n)$ then
   $u + v = (u_1 +_W V_1, u_2 +_W V_2 \ldots, u_n +_W V_n))$
2. $au \in C$ for all $u \in C$, and all $a \in GF(q)$
   if $u = (u_1, u_2, \ldots, u_n)$,, then $au = (au_1, au_2, \ldots, au_n))$

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n, q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

Definition A subset $C \subseteq F_q^n$ is a linear code if

1. $u + v \in C$ for all $u, v \in C$
   (if $u = (u_1, u_2, \ldots, u_n), v = (v_1, v_2 \ldots, v_n)$ then
   $u + v = (u_1 +_W V_1, u_2 +_W V_2 \ldots, u_n +_W V_n))$
2. $au \in C$ for all $u \in C$, and all $a \in GF(q)$
   if $u = (u_1, u_2, \ldots, u_n),$, then $au = (au_1, au_2, \ldots, au_n))$

Lemma A subset $C \subseteq F_q^n$ is a linear code iff one of the following conditions is satisfied

1. C is a subspace of F_q^n.
2. Sum of any two codewords from C is in C (for the case $q = 2$)

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n, q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

Definition A subset $C \subseteq F_q^n$ is a linear code if

1. $u + v \in C$ for all $u, v \in C$
(if $u = (u_1, u_2, \ldots, u_n), v = (v_1, v_2 \ldots, v_n)$ then
$u + v = (u_1 +_W V_1, u_2 +_W V_2 \ldots, u_n +_W V_n))$
2. $au \in C$ for all $u \in C$, and all $a \in GF(q)$
if $u = (u_1, u_2, \ldots, u_n)$,, then $au = (au_1, au_2, \ldots, au_n))$

Lemma A subset $C \subseteq F_q^n$ is a linear code iff one of the following conditions is satisfied

1. C is a subspace of F_q^n.
2. Sum of any two codewords from C is in C (for the case $q = 2$)

If C is a k-dimensional subspace of F_q^n, then C is called $[n, k]$-code.

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n,q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

Definition A subset $C \subseteq F_q^n$ is a linear code if

1. $u + v \in C$ for all $u, v \in C$
   (if $u = (u_1, u_2, \ldots, u_n), v = (v_1, v_2 \ldots, v_n)$ then
   $u + v = (u_1 +_W V_1, u_2 +_W V_2 \ldots, u_n +_W V_n))$
2. $au \in C$ for all $u \in C$, and all $a \in GF(q)$
   if $u = (u_1, u_2, \ldots, u_n),$, then $au = (au_1, au_2, \ldots, au_n))$

Lemma A subset $C \subseteq F_q^n$ is a linear code iff one of the following conditions is satisfied

1. C is a subspace of F_q^n.
2. Sum of any two codewords from C is in C (for the case $q = 2$)

If C is a k-dimensional subspace of F_q^n, then C is called $[n, k]$-code. It has q^k codewords.

LINEAR CODES - I.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n,q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

Definition A subset $C \subseteq F_q^n$ is a linear code if

1. $u + v \in C$ for all $u, v \in C$
   (if $u = (u_1, u_2, \ldots, u_n), v = (v_1, v_2 \ldots, v_n)$ then
   $u + v = (u_1 +_W V_1, u_2 +_W V_2 \ldots, u_n +_W V_n))$
2. $au \in C$ for all $u \in C$, and all $a \in GF(q)$
   if $u = (u_1, u_2, \ldots, u_n)$,, then $au = (au_1, au_2, \ldots, au_n))$

Lemma A subset $C \subseteq F_q^n$ is a linear code iff one of the following conditions is satisfied

1. C is a subspace of F_q^n.
2. Sum of any two codewords from C is in C (for the case $q = 2$)

If C is a k-dimensional subspace of F_q^n, then C is called $[n,k]$-code. It has q^k codewords. If the minimal distance of C is d, then it is said to be the $[n,k,d]$ code.

Linear codes are special sets of words of a fixed length n over an alphabet $\Sigma_q = \{0, .., q-1\}$, where q is a (power of) prime.

In the following two chapters F_q^n (or $V(n,q)$) will be considered as the vector spaces of all n-tuples over the Galois field $GF(q)$ (with the elements $\{0, .., q-1\}$ and with arithmetical operations modulo q.)

Definition A subset $C \subseteq F_q^n$ is a linear code if

1. $u + v \in C$ for all $u, v \in C$
   (if $u = (u_1, u_2, \ldots, u_n), v = (v_1, v_2 \ldots, v_n)$ then
   $u + v = (u_1 +_W V_1, u_2 +_W V_2 \ldots, u_n +_W V_n))$
2. $au \in C$ for all $u \in C$, and all $a \in GF(q)$
   if $u = (u_1, u_2, \ldots, u_n)$,, then $au = (au_1, au_2, \ldots, au_n))$

Lemma A subset $C \subseteq F_q^n$ is a linear code iff one of the following conditions is satisfied

1. C is a subspace of F_q^n.
2. Sum of any two codewords from C is in C (for the case $q = 2$)

If C is a k-dimensional subspace of F_q^n, then C is called $[n, k]$-code. It has q^k codewords. If the minimal distance of C is d, then it is said to be the $[n, k, d]$ code.

LINEAR CODES - II.

If C is a linear $[n, k]$ code, then it has several bases.

If C is a linear $[n, k]$ code, then it has several bases.

A base **B** of C is such a sets of k codewords of C that each codeword of C is a linear combination of the codewords from the base **B**.

If C is a linear $[n, k]$ code, then it has several bases.

A base **B** of C is such a sets of k codewords of C that each codeword of C is a linear combination of the codewords from the base **B**.

Each base **B** of C is usually represented by a (k, n) matrix, $G_{\mathbf{B}}$, so called a **generator matrix of C**, the i-th row of which is the i-th codeword of **B**.

EXERCISE

EXERCISE

Which of the following binary codes are linear?

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES

$C_2 = \{000, 011, 101, 110\}$

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES

$C_2 = \{000, 011, 101, 110\}$ – YES

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES

$C_2 = \{000, 011, 101, 110\}$ – YES

$C_3 = \{00000, 01101, 10110, 11011\}$

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$ – NO

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES

$C_2 = \{000, 011, 101, 110\}$ – YES

$C_3 = \{00000, 01101, 10110, 11011\}$ – YES

$C_5 = \{101, 111, 011\}$ – NO

$C_6 = \{000, 001, 010, 011\}$

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES

$C_2 = \{000, 011, 101, 110\}$ – YES

$C_3 = \{00000, 01101, 10110, 11011\}$ – YES

$C_5 = \{101, 111, 011\}$ – NO

$C_6 = \{000, 001, 010, 011\}$ – YES

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES

$C_2 = \{000, 011, 101, 110\}$ – YES

$C_3 = \{00000, 01101, 10110, 11011\}$ – YES

$C_5 = \{101, 111, 011\}$ – NO

$C_6 = \{000, 001, 010, 011\}$ – YES

$C_7 = \{0000, 1001, 0110, 1110\}$

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES

$C_2 = \{000, 011, 101, 110\}$ – YES

$C_3 = \{00000, 01101, 10110, 11011\}$ – YES

$C_5 = \{101, 111, 011\}$ – NO

$C_6 = \{000, 001, 010, 011\}$ – YES

$C_7 = \{0000, 1001, 0110, 1110\}$ – NO

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$ – NO
$C_6 = \{000, 001, 010, 011\}$ – YES
$C_7 = \{0000, 1001, 0110, 1110\}$ – NO

How to create a linear code?

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$ – NO
$C_6 = \{000, 001, 010, 011\}$ – YES
$C_7 = \{0000, 1001, 0110, 1110\}$ – NO

How to create a linear code?

Notation: If S is a set of vectors of a vector space, then let $\langle S\rangle$ be the set of all linear combinations of vectors from S.

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$ – NO
$C_6 = \{000, 001, 010, 011\}$ – YES
$C_7 = \{0000, 1001, 0110, 1110\}$ – NO

How to create a linear code?

Notation: If S is a set of vectors of a vector space, then let $\langle S\rangle$ be the set of all linear combinations of vectors from S.

Theorem For any subset S of a linear space, $\langle S\rangle$ is a linear space that consists of the following words:

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$ – NO
$C_6 = \{000, 001, 010, 011\}$ – YES
$C_7 = \{0000, 1001, 0110, 1110\}$ – NO

How to create a linear code?

Notation: If S is a set of vectors of a vector space, then let $\langle S\rangle$ be the set of all linear combinations of vectors from S.

Theorem For any subset S of a linear space, $\langle S\rangle$ is a linear space that consists of the following words:

- the zero word,

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$ – NO
$C_6 = \{000, 001, 010, 011\}$ – YES
$C_7 = \{0000, 1001, 0110, 1110\}$ – NO

How to create a linear code?

Notation: If S is a set of vectors of a vector space, then let $\langle S \rangle$ be the set of all linear combinations of vectors from S.

Theorem For any subset S of a linear space, $\langle S \rangle$ is a linear space that consists of the following words:

- the zero word,
- all words in S,

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$ – NO
$C_6 = \{000, 001, 010, 011\}$ – YES
$C_7 = \{0000, 1001, 0110, 1110\}$ – NO

How to create a linear code?

Notation: If S is a set of vectors of a vector space, then let $\langle S \rangle$ be the set of all linear combinations of vectors from S.

Theorem For any subset S of a linear space, $\langle S \rangle$ is a linear space that consists of the following words:

- the zero word,
- all words in S,
- all sums of two or more words in S.

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$ – NO
$C_6 = \{000, 001, 010, 011\}$ – YES
$C_7 = \{0000, 1001, 0110, 1110\}$ – NO

How to create a linear code?

Notation: If S is a set of vectors of a vector space, then let $\langle S\rangle$ be the set of all linear combinations of vectors from S.

Theorem For any subset S of a linear space, $\langle S\rangle$ is a linear space that consists of the following words:

- the zero word,
- all words in S,
- all sums of two or more words in S.

Example

$$S = \{0100, 0011, 1100\}$$
$$\langle S\rangle = \{0000, 0100, 0011, 1100, 0111, 1011, 1000, 1111\}.$$

EXERCISE

Which of the following binary codes are linear?

$C_1 = \{00, 01, 10, 11\}$ – YES
$C_2 = \{000, 011, 101, 110\}$ – YES
$C_3 = \{00000, 01101, 10110, 11011\}$ – YES
$C_5 = \{101, 111, 011\}$ – NO
$C_6 = \{000, 001, 010, 011\}$ – YES
$C_7 = \{0000, 1001, 0110, 1110\}$ – NO

How to create a linear code?

Notation: If S is a set of vectors of a vector space, then let $\langle S \rangle$ be the set of all linear combinations of vectors from S.

Theorem For any subset S of a linear space, $\langle S \rangle$ is a linear space that consists of the following words:

- the zero word,
- all words in S,
- all sums of two or more words in S.

Example

$$S = \{0100, 0011, 1100\}$$
$$\langle S \rangle = \{0000, 0100, 0011, 1100, 0111, 1011, 1000, 1111\}.$$

BASIC PROPERTIES of LINEAR CODES I

BASIC PROPERTIES of LINEAR CODES I

Notation: Let w(x) (weight of x) denote the number of non-zero entries of x.

BASIC PROPERTIES of LINEAR CODES I

Notation: Let w(x) (weight of x) denote the number of non-zero entries of x.

Lemma If $x, y \in F_q^n$, then $h(x, y) = w(x - y)$.

BASIC PROPERTIES of LINEAR CODES I

Notation: Let w(x) (weight of x) denote the number of non-zero entries of x.

Lemma If $x, y \in F_q^n$, then $h(x, y) = w(x - y)$.

Proof $x - y$ has non-zero entries in exactly those positions where x and y differ.

BASIC PROPERTIES of LINEAR CODES I

Notation: Let w(x) (weight of x) denote the number of non-zero entries of x.

Lemma If $x, y \in F_q^n$, then $h(x, y) = w(x - y)$.

Proof $x - y$ has non-zero entries in exactly those positions where x and y differ.

Theorem Let C be a linear code and let weight of C, notation $w(C)$, be the smallest of the weights of non-zero codewords of C. Then $h(C) = w(C)$.

BASIC PROPERTIES of LINEAR CODES I

Notation: Let w(x) (weight of x) denote the number of non-zero entries of x.

Lemma If $x, y \in F_q^n$, then $h(x, y) = w(x - y)$.

Proof $x - y$ has non-zero entries in exactly those positions where x and y differ.

Theorem Let C be a linear code and let weight of C, notation $w(C)$, be the smallest of the weights of non-zero codewords of C. Then $h(C) = w(C)$.

Proof There are $x, y \in C$ such that $h(C) = h(x, y)$. Hence $h(C) = w(x - y) \geq w(C)$.

On the other hand, for some $x \in C$

$$w(C) = w(x) = h(x, 0) \geq h(C).$$

BASIC PROPERTIES of LINEAR CODES I

Notation: Let w(x) (weight of x) denote the number of non-zero entries of x.

Lemma If $x, y \in F_q^n$, then $h(x, y) = w(x - y)$.

Proof $x - y$ has non-zero entries in exactly those positions where x and y differ.

Theorem Let C be a linear code and let weight of C, notation $w(C)$, be the smallest of the weights of non-zero codewords of C. Then $h(C) = w(C)$.

Proof There are $x, y \in C$ such that $h(C) = h(x, y)$. Hence $h(C) = w(x - y) \geq w(C)$.

On the other hand, for some $x \in C$

$$w(C) = w(x) = h(x, 0) \geq h(C).$$

Consequence

- If C is a non-linear code with m codewords, then in order to determine $h(C)$ one has to make in general $\binom{m}{2} = \Theta(m^2)$ comparisons in the worst case.

BASIC PROPERTIES of LINEAR CODES I

Notation: Let w(x) (weight of x) denote the number of non-zero entries of x.

Lemma If $x, y \in F_q^n$, then $h(x, y) = w(x - y)$.

Proof $x - y$ has non-zero entries in exactly those positions where x and y differ.

Theorem Let C be a linear code and let weight of C, notation $w(C)$, be the smallest of the weights of non-zero codewords of C. Then $h(C) = w(C)$.

Proof There are $x, y \in C$ such that $h(C) = h(x, y)$. Hence $h(C) = w(x - y) \geq w(C)$.

On the other hand, for some $x \in C$

$$w(C) = w(x) = h(x, 0) \geq h(C).$$

Consequence

- If C is a non-linear code with m codewords, then in order to determine $h(C)$ one has to make in general $\binom{m}{2} = \Theta(m^2)$ comparisons in the worst case.
- **If C is a linear code with m codewords, then in order to determine $h(C), m - 1$ comparisons are enough.**

BASIC PROPERTIES of LINEAR CODES I

Notation: Let w(x) (weight of x) denote the number of non-zero entries of x.

Lemma If $x, y \in F_q^n$, then $h(x, y) = w(x - y)$.

Proof $x - y$ has non-zero entries in exactly those positions where x and y differ.

Theorem Let C be a linear code and let weight of C, notation $w(C)$, be the smallest of the weights of non-zero codewords of C. Then $h(C) = w(C)$.

Proof There are $x, y \in C$ such that $h(C) = h(x, y)$. Hence $h(C) = w(x - y) \geq w(C)$.

On the other hand, for some $x \in C$

$$w(C) = w(x) = h(x, 0) \geq h(C).$$

Consequence

- If C is a non-linear code with m codewords, then in order to determine $h(C)$ one has to make in general $\binom{m}{2} = \Theta(m^2)$ comparisons in the worst case.
- **If C is a linear code with m codewords, then in order to determine $h(C), m - 1$ comparisons are enough.**

BASIC PROPERTIES of LINEAR CODES II

If C is a linear $[n, k]$-code, then it has many basis Γ consisting of k codewords and such that each codeword of C is a linear combination of the codewords from any Γ.

BASIC PROPERTIES of LINEAR CODES II

If C is a linear $[n, k]$-code, then it has many basis Γ consisting of k codewords and such that each codeword of C is a linear combination of the codewords from any Γ.

Example

Code

$$\begin{aligned} C_4 \quad = \quad & \{0000000, 1111111, 1000101, 1100010, \\ & 0110001, 1011000, 0101100, 0010110, \\ & 0001011, 0111010, 0011101, 1001110, \\ & 0100111, 1010011, 1101001, 1110100\} \end{aligned}$$

has, as one of its bases, the set

BASIC PROPERTIES of LINEAR CODES II

If C is a linear $[n, k]$-code, then it has many basis Γ consisting of k codewords and such that each codeword of C is a linear combination of the codewords from any Γ.

Example

Code

$$\begin{aligned} C_4 \quad = \quad & \{0000000, 1111111, 1000101, 1100010, \\ & 0110001, 1011000, 0101100, 0010110, \\ & 0001011, 0111010, 0011101, 1001110, \\ & 0100111, 1010011, 1101001, 1110100\} \end{aligned}$$

has, as one of its bases, the set

$$\{1111111, 1000101, 1100010, 0110001\}.$$

BASIC PROPERTIES of LINEAR CODES II

If C is a linear $[n, k]$-code, then it has many basis Γ consisting of k codewords and such that each codeword of C is a linear combination of the codewords from any Γ.

Example

Code

$$\begin{aligned} C_4 \quad = \quad & \{0000000, 1111111, 1000101, 1100010, \\ & 0110001, 1011000, 0101100, 0010110, \\ & 0001011, 0111010, 0011101, 1001110, \\ & 0100111, 1010011, 1101001, 1110100\} \end{aligned}$$

has, as one of its bases, the set

$$\{1111111, 1000101, 1100010, 0110001\}.$$

How many different bases has a linear code?

BASIC PROPERTIES of LINEAR CODES II

If C is a linear $[n, k]$-code, then it has many basis Γ consisting of k codewords and such that each codeword of C is a linear combination of the codewords from any Γ.

Example

Code

$$\begin{aligned} C_4 \quad = \quad & \{0000000, 1111111, 1000101, 1100010, \\ & 0110001, 1011000, 0101100, 0010110, \\ & 0001011, 0111010, 0011101, 1001110, \\ & 0100111, 1010011, 1101001, 1110100\} \end{aligned}$$

has, as one of its bases, the set

$$\{1111111, 1000101, 1100010, 0110001\}.$$

How many different bases has a linear code?

Theorem A binary linear code of dimension k has

$$\frac{1}{k!} \prod_{i=0}^{k-1} (2^k - 2^i)$$

bases.

EXAMPLE

If a code C has 2^{200} codewords, then there is no way to write down and/or to store all its codewords.

EXAMPLE

If a code C has 2^{200} codewords, then there is no way to write down and/or to store all its codewords.

WHY

EXAMPLE

If a code C has 2^{200} codewords, then there is no way to write down and/or to store all its codewords.

WHY

However, In case we have $[2^{200}, 200]$ linear code C, then to specify/store fully C we need only to store

EXAMPLE

If a code C has 2^{200} codewords, then there is no way to write down and/or to store all its codewords.

WHY

However, In case we have $[2^{200}, 200]$ linear code C, then to specify/store fully C we need only to store

$$200$$

codewords

If a code C has 2^{200} codewords, then there is no way to write down and/or to store all its codewords.

WHY

However, In case we have $[2^{200}, 200]$ linear code C, then to specify/store fully C we need only to store

200

codewords - from one of its basis.

ADVANTAGES and DISADVANTAGES of LINEAR CODES I.

ADVANTAGES and DISADVANTAGES of LINEAR CODES I.

Advantages - are big.

ADVANTAGES and DISADVANTAGES of LINEAR CODES I.

Advantages - are big.

1. Minimal distance $h(C)$ is easy to compute if C is a linear code.

ADVANTAGES and DISADVANTAGES of LINEAR CODES I.

Advantages - are big.

1. Minimal distance $h(C)$ is easy to compute if C is a linear code.
2. Linear codes have simple specifications.

ADVANTAGES and DISADVANTAGES of LINEAR CODES I.

Advantages - are big.

1. Minimal distance $h(C)$ is easy to compute if C is a linear code.
2. Linear codes have simple specifications.

- To specify a non-linear code usually all codewords have to be listed.

ADVANTAGES and DISADVANTAGES of LINEAR CODES I.

Advantages - are big.

1. Minimal distance $h(C)$ is easy to compute if C is a linear code.
2. Linear codes have simple specifications.

- To specify a non-linear code usually all codewords have to be listed.
- To specify a linear $[n, k]$-code it is enough to list k codewords (of a basis).

ADVANTAGES and DISADVANTAGES of LINEAR CODES I.

Advantages - are big.

1. Minimal distance $h(C)$ is easy to compute if C is a linear code.
2. Linear codes have simple specifications.

- To specify a non-linear code usually all codewords have to be listed.
- To specify a linear $[n, k]$-code it is enough to list k codewords (of a basis).

Example One of the generator matrices of the binary code

$$C_2 = \begin{Bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{Bmatrix} \text{ is the matrix } \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}$$

ADVANTAGES and DISADVANTAGES of LINEAR CODES I.

Advantages - are big.

1. Minimal distance $h(C)$ is easy to compute if C is a linear code.
2. Linear codes have simple specifications.

- To specify a non-linear code usually all codewords have to be listed.
- To specify a linear $[n, k]$-code it is enough to list k codewords (of a basis).

Example One of the generator matrices of the binary code

$$C_2 = \begin{Bmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{Bmatrix} \text{ is the matrix } \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}$$

and one of the generator matrices of the code

$$C_4 \text{ is } \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

ADVANTAGES and DISADVANTAGES of LINEAR CODES I.

Advantages - are big.

1. Minimal distance $h(C)$ is easy to compute if C is a linear code.
2. Linear codes have simple specifications.

- To specify a non-linear code usually all codewords have to be listed.
- To specify a linear $[n, k]$-code it is enough to list k codewords (of a basis).

Example One of the generator matrices of the binary code

$$C_2 = \left\{\begin{matrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{matrix}\right\} \text{ is the matrix } \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}$$

and one of the generator matrices of the code

$$C_4 \text{ is } \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

3. There are simple encoding/decoding procedures for linear codes.

ADANTAGES and DISADVANTAGES of LINEAR CODES II.

Disadvantages of linear codes are small:

1. Linear q-codes are not defined unless q is a power of a prime.

Disadvantages of linear codes are small:

1. Linear q-codes are not defined unless q is a power of a prime.
2. The restriction to linear codes might be a restriction to weaker codes than sometimes desired.

EQUIVALENCE of LINEAR CODES I

EQUIVALENCE of LINEAR CODES I

Definition Two linear codes on $GF(q)$ are called equivalent if one can be obtained from another by the following operations:

EQUIVALENCE of LINEAR CODES I

Definition Two linear codes on $GF(q)$ are called equivalent if one can be obtained from another by the following operations:

(a) permutation of the words or positions of the code;

EQUIVALENCE of LINEAR CODES I

Definition Two linear codes on $GF(q)$ are called equivalent if one can be obtained from another by the following operations:

(a permutation of the words or positions of the code;

(b multiplication of symbols appearing in a fixed position by a non-zero scalar.

EQUIVALENCE of LINEAR CODES I

Definition Two linear codes on $GF(q)$ are called equivalent if one can be obtained from another by the following operations:

(a) permutation of the words or positions of the code;

(b) multiplication of symbols appearing in a fixed position by a non-zero scalar.

Theorem Two $k \times n$ matrices generate equivalent linear $[n, k]$-codes over F_q^n if one matrix can be obtained from the other by a sequence of the following operations:

EQUIVALENCE of LINEAR CODES I

Definition Two linear codes on $GF(q)$ are called equivalent if one can be obtained from another by the following operations:

- (a) permutation of the words or positions of the code;
- (b) multiplication of symbols appearing in a fixed position by a non-zero scalar.

Theorem Two $k \times n$ matrices generate equivalent linear $[n, k]$-codes over F_q^n if one matrix can be obtained from the other by a sequence of the following operations:

- (a) permutation of the rows

EQUIVALENCE of LINEAR CODES I

Definition Two linear codes on $GF(q)$ are called equivalent if one can be obtained from another by the following operations:

(a permutation of the words or positions of the code;

(b multiplication of symbols appearing in a fixed position by a non-zero scalar.

Theorem Two $k \times n$ matrices generate equivalent linear $[n, k]$-codes over F_q^n if one matrix can be obtained from the other by a sequence of the following operations:

(a permutation of the rows

(b multiplication of a row by a non-zero scalar

EQUIVALENCE of LINEAR CODES I

Definition Two linear codes on $GF(q)$ are called equivalent if one can be obtained from another by the following operations:

(a) permutation of the words or positions of the code;

(b) multiplication of symbols appearing in a fixed position by a non-zero scalar.

Theorem Two $k \times n$ matrices generate equivalent linear $[n, k]$-codes over F_q^n if one matrix can be obtained from the other by a sequence of the following operations:

(a) permutation of the rows

(b) multiplication of a row by a non-zero scalar

(c) addition of one row to another

EQUIVALENCE of LINEAR CODES I

Definition Two linear codes on $GF(q)$ are called equivalent if one can be obtained from another by the following operations:

(a) permutation of the words or positions of the code;

(b) multiplication of symbols appearing in a fixed position by a non-zero scalar.

Theorem Two $k \times n$ matrices generate equivalent linear $[n, k]$-codes over F_q^n if one matrix can be obtained from the other by a sequence of the following operations:

(a) permutation of the rows

(b) multiplication of a row by a non-zero scalar

(c) addition of one row to another

(d) permutation of columns

EQUIVALENCE of LINEAR CODES I

Definition Two linear codes on $GF(q)$ are called equivalent if one can be obtained from another by the following operations:

(a) permutation of the words or positions of the code;

(b) multiplication of symbols appearing in a fixed position by a non-zero scalar.

Theorem Two $k \times n$ matrices generate equivalent linear $[n, k]$-codes over F_q^n if one matrix can be obtained from the other by a sequence of the following operations:

(a) permutation of the rows

(b) multiplication of a row by a non-zero scalar

(c) addition of one row to another

(d) permutation of columns

(e) multiplication of a column by a non-zero scalar

EQUIVALENCE of LINEAR CODES I

Definition Two linear codes on $GF(q)$ are called equivalent if one can be obtained from another by the following operations:

(a) permutation of the words or positions of the code;

(b) multiplication of symbols appearing in a fixed position by a non-zero scalar.

Theorem Two $k \times n$ matrices generate equivalent linear $[n, k]$-codes over F_q^n if one matrix can be obtained from the other by a sequence of the following operations:

(a) permutation of the rows

(b) multiplication of a row by a non-zero scalar

(c) addition of one row to another

(d) permutation of columns

(e) multiplication of a column by a non-zero scalar

Proof Operations (a) - (c) just replace one basis by another. Last two operations convert a generator matrix to one of an equivalent code.

554,0-1

EQUIVALENCE of LINEAR CODES II

Theorem Let G be a generator matrix of an $[n, k]$-code. Rows of G are then linearly independent .By operations (a) - (e) the matrix G can be transformed into the form: $[I_k|A]$ where I_k is the $k \times k$ identity matrix, and A is a $k \times (n-k)$ matrix.

29

EQUIVALENCE of LINEAR CODES II

Theorem Let G be a generator matrix of an $[n, k]$-code. Rows of G are then linearly independent .By operations (a) - (e) the matrix G can be transformed into the form: $[I_k|A]$ where I_k is the $k \times k$ identity matrix, and A is a $k \times (n-k)$ matrix.

Example

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \rightarrow$$

$$\rightarrow \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix} \rightarrow \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix} \rightarrow$$

ENCODING with LINEAR CODES

ENCODING with LINEAR CODES

is a vector $\times$ matrix multiplication

ENCODING with LINEAR CODES

is a vector $\times$ matrix multiplication

Let C be a linear $[n, k]$-code over F_q^n with a generator $k \times n$ matrix G.

ENCODING with LINEAR CODES

is a vector $\times$ matrix multiplication

Let C be a linear $[n, k]$-code over F_q^n with a generator $k \times n$ matrix G.

Theorem C has q^k codewords.

ENCODING with LINEAR CODES

is a vector $\times$ matrix multiplication

Let C be a linear $[n,k]$-code over F_q^n with a generator $k \times n$ matrix G.

Theorem C has q^k codewords.

Proof Theorem follows from the fact that each codeword of C can be expressed uniquely as a linear combination of the basis codewords/vectors.

ENCODING with LINEAR CODES

is a vector $\times$ matrix multiplication

Let C be a linear $[n, k]$-code over F_q^n with a generator $k \times n$ matrix G.

Theorem C has q^k codewords.

Proof Theorem follows from the fact that each codeword of C can be expressed uniquely as a linear combination of the basis codewords/vectors.

Corollary The code C can be used to encode uniquely q^k messages.
(Let us identify messages with elements of F_q^k.)

ENCODING with LINEAR CODES

is a vector $\times$ matrix multiplication

Let C be a linear $[n,k]$-code over F_q^n with a generator $k \times n$ matrix G.

Theorem C has q^k codewords.

Proof Theorem follows from the fact that each codeword of C can be expressed uniquely as a linear combination of the basis codewords/vectors.

Corollary The code C can be used to encode uniquely q^k messages.
(Let us identify messages with elements of F_q^k.)

Encoding of a message $u = (u_1, \ldots, u_k)$ using the generator matrix G:

ENCODING with LINEAR CODES

is a vector $\times$ matrix multiplication

Let C be a linear $[n, k]$-code over F_q^n with a generator $k \times n$ matrix G.

Theorem C has q^k codewords.

Proof Theorem follows from the fact that each codeword of C can be expressed uniquely as a linear combination of the basis codewords/vectors.

Corollary The code C can be used to encode uniquely q^k messages.
(Let us identify messages with elements of F_q^k.)

Encoding of a message $u = (u_1, \ldots, u_k)$ using the generator matrix G:

$$u \cdot G = \sum_{i=1}^{k} u_i r_i \text{ where } r_1, \ldots, r_k \text{ are rows of } G.$$

ENCODING with LINEAR CODES

is a vector $\times$ matrix multiplication

Let C be a linear $[n,k]$-code over F_q^n with a generator $k \times n$ matrix G.

Theorem C has q^k codewords.

Proof Theorem follows from the fact that each codeword of C can be expressed uniquely as a linear combination of the basis codewords/vectors.

Corollary The code C can be used to encode uniquely q^k messages.
(Let us identify messages with elements of F_q^k.)

Encoding of a message $u = (u_1, \ldots, u_k)$ using the generator matrix G:

$$u \cdot G = \sum_{i=1}^{k} u_i r_i \text{ where } r_1, \ldots, r_k \text{ are rows of } G.$$

Example Let C be a $[7,4]$-code with the generator matrix

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

A message (u_1, u_2, u_3, u_4) is encoded as:???

ENCODING with LINEAR CODES

is a vector $\times$ matrix multiplication

Let C be a linear $[n,k]$-code over F_q^n with a generator $k \times n$ matrix G.

Theorem C has q^k codewords.

Proof Theorem follows from the fact that each codeword of C can be expressed uniquely as a linear combination of the basis codewords/vectors.

Corollary The code C can be used to encode uniquely q^k messages.
(Let us identify messages with elements of F_q^k.)

Encoding of a message $u = (u_1, \ldots, u_k)$ using the generator matrix G:

$$u \cdot G = \sum_{i=1}^{k} u_i r_i \text{ where } r_1, \ldots, r_k \text{ are rows of } G.$$

Example Let C be a $[7,4]$-code with the generator matrix

$$\mathrm{G}= \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

A message (u_1, u_2, u_3, u_4) is encoded as:???
For example:
0 0 0 0 is encoded as? .....

ENCODING with LINEAR CODES

is a vector $\times$ matrix multiplication

Let C be a linear $[n, k]$-code over F_q^n with a generator $k \times n$ matrix G.

Theorem C has q^k codewords.

Proof Theorem follows from the fact that each codeword of C can be expressed uniquely as a linear combination of the basis codewords/vectors.

Corollary The code C can be used to encode uniquely q^k messages.
(Let us identify messages with elements of F_q^k.)

Encoding of a message $u = (u_1, \ldots, u_k)$ using the generator matrix G:

$$u \cdot G = \sum_{i=1}^{k} u_i r_i \text{ where } r_1, \ldots, r_k \text{ are rows of } G.$$

Example Let C be a $[7, 4]$-code with the generator matrix

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

A message (u_1, u_2, u_3, u_4) is encoded as:???
For example:
0 0 0 0 is encoded as? 0000000
1 0 0 0 is encoded as? .....

ENCODING with LINEAR CODES

is a vector $\times$ matrix multiplication

Let C be a linear $[n,k]$-code over F_q^n with a generator $k \times n$ matrix G.

Theorem C has q^k codewords.

Proof Theorem follows from the fact that each codeword of C can be expressed uniquely as a linear combination of the basis codewords/vectors.

Corollary The code C can be used to encode uniquely q^k messages.
(Let us identify messages with elements of F_q^k.)

Encoding of a message $u = (u_1, \ldots, u_k)$ using the generator matrix G:

$$u \cdot G = \sum_{i=1}^{k} u_i r_i \text{ where } r_1, \ldots, r_k \text{ are rows of } G.$$

Example Let C be a $[7,4]$-code with the generator matrix

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

A message (u_1, u_2, u_3, u_4) is encoded as:???
For example:
0 0 0 0 is encoded as? 0000000
1 0 0 0 is encoded as? 1000101
1 1 1 0 is encoded as? .....

ENCODING with LINEAR CODES

is a vector $\times$ matrix multiplication

Let C be a linear $[n, k]$-code over F_q^n with a generator $k \times n$ matrix G.

Theorem C has q^k codewords.

Proof Theorem follows from the fact that each codeword of C can be expressed uniquely as a linear combination of the basis codewords/vectors.

Corollary The code C can be used to encode uniquely q^k messages.
(Let us identify messages with elements of F_q^k.)

Encoding of a message $u = (u_1, \ldots, u_k)$ using the generator matrix G:

$$u \cdot G = \sum_{i=1}^{k} u_i r_i \text{ where } r_1, \ldots, r_k \text{ are rows of } G.$$

Example Let C be a $[7, 4]$-code with the generator matrix

$$G= \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

A message (u_1, u_2, u_3, u_4) is encoded as:???
For example:
0 0 0 0 is encoded as? 0000000
1 0 0 0 is encoded as? 1000101
1 1 1 0 is encoded as? 1110100

UNIQUENESS of ENCODING

with linear codes

UNIQUENESS of ENCODING

with linear codes

Theorem If $G = \{w_i\}_{i=1}^k$ is a generator matrix of a binary linear code C of length n and dimension k, then the set of codewords/vectors

$$v = ug$$

ranges over all 2^kn words of length k

Therefore

$$C = \{ug \mid u \in \{0, 1\}^k\}$$

UNIQUENESS of ENCODING

with linear codes

Theorem If $G = \{w_i\}_{i=1}^{k}$ is a generator matrix of a binary linear code C of length n and dimension k, then the set of codewords/vectors

$$v = ug$$

ranges over all 2^kn words of length k

Therefore

$$C = \{ug \mid u \in \{0,1\}^k\}$$

Moreover

$$u_1 G = u_2$$

if and only if

$$u_1 = u_2$$

APPENDIX II.

.

.

APPENDIX II.

EFFICIENT TRANMIION of INFORMATION

Important problems of information theory are how to define formally such concepts as information and how to store or transmit information efficiently.

[1] Notation lg (Ln) [log] will be used for binary, natural and decimal logarithms.

EFFICIENT TRANMIION of INFORMATION

Important problems of information theory are how to define formally such concepts as information and how to store or transmit information efficiently.

Let X be a random variable (source) which takes any value x with probability $p(x)$.

[1] Notation lg (Ln) [log] will be used for binary, natural and decimal logarithms.

EFFICIENT TRANMIION of INFORMATION

Important problems of information theory are how to define formally such concepts as information and how to store or transmit information efficiently.

Let X be a random variable (source) which takes any value x with probability $p(x)$. The entropy of X is defined by

$$S(X) = -\sum_x p(x) lg\ p(x)$$

and it is considered to be the information content of X.

[1]Notation lg (Ln) [log] will be used for binary, natural and decimal logarithms.

EFFICIENT TRANMIION of INFORMATION

Important problems of information theory are how to define formally such concepts as information and how to store or transmit information efficiently.

Let X be a random variable (source) which takes any value x with probability $p(x)$. The entropy of X is defined by

$$S(X) = -\sum_x p(x) lg\ p(x)$$

and it is considered to be the information content of X. rbinary variable X which takes on the value 1 with probability p and the value 0 with probability $1-p$, then the information content of X is:

[1] Notation lg (Ln) [log] will be used for binary, natural and decimal logarithms.

EFFICIENT TRANMIION of INFORMATION

Important problems of information theory are how to define formally such concepts as information and how to store or transmit information efficiently.

Let X be a random variable (source) which takes any value x with probability $p(x)$. The entropy of X is defined by

$$S(X) = -\sum_x p(x) lg\ p(x)$$

and it is considered to be the information content of X. rbinary variable X which takes on the value 1 with probability p and the value 0 with probability $1 - p$, then the information content of X is:

$$S(X) = H(p) = -p\ lg\ p - (1-p) lg(1-p)^1$$

[1]Notation lg (Ln) [log] will be used for binary, natural and decimal logarithms.

EFFICIENT TRANMIION of INFORMATION

Important problems of information theory are how to define formally such concepts as information and how to store or transmit information efficiently.

Let X be a random variable (source) which takes any value x with probability $p(x)$. The entropy of X is defined by

$$S(X) = -\sum_x p(x) lg \; p(x)$$

and it is considered to be the information content of X. rbinary variable X which takes on the value 1 with probability p and the value 0 with probability $1-p$, then the information content of X is:

$$S(X) = H(p) = -p \; lg \; p - (1-p) lg (1-p)^1$$

Problem: What is the minimal number of bits needed to transmit n values of X?

[1]Notation lg (Ln) [log] will be used for binary, natural and decimal logarithms.

EFFICIENT TRANMIION of INFORMATION

Important problems of information theory are how to define formally such concepts as information and how to store or transmit information efficiently.

Let X be a random variable (source) which takes any value x with probability $p(x)$. The entropy of X is defined by

$$S(X) = -\sum_x p(x) lg\ p(x)$$

and it is considered to be the information content of X. rbinary variable X which takes on the value 1 with probability p and the value 0 with probability $1 - p$, then the information content of X is:

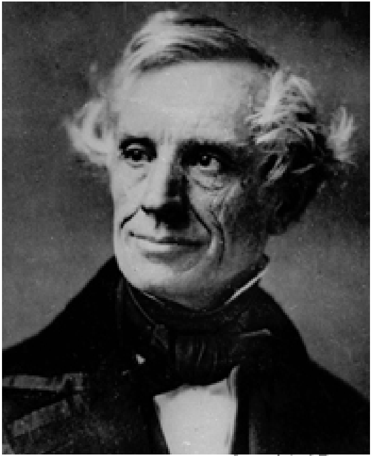
$$S(X) = H(p) = -p\ lg\ p - (1-p) lg(1-p)$$ [1]

Problem: What is the minimal number of bits needed to transmit n values of X?
Basic idea: Encode more (less) probable outputs of X by shorter (longer) binary words.
Example (Moorse code - 1838)

a .-	b -...	c -.-.	d -..	e .	f ..-.	g –.
h	i ..	j .—	k -.-	l .-..	m –	n -.
o —	p .–.	q –.-	r .-.	s ...	t -	u ..-
v ...-	w .–	x -..-	y -.–	z –..		

[1] Notation lg (Ln) [log] will be used for binary, natural and decimal logarithms.

Samuel Moorse



Associated Press

SHANNON's NOISELESS CODING THEOREM

SHANNON's NOISELESS CODING THEOREM

SHANNON's noiseless coding theorem says that in order to transmit n values of X, we need, and it is sufficient, to use $nS(X)$ bits.

SHANNON's NOISELESS CODING THEOREM

SHANNON's noiseless coding theorem says that in order to transmit n values of X, we need, and it is sufficient, to use $nS(X)$ bits.

More exactly, we cannot do better than the bound $nS(X)$ says, and we can reach the bound $nS(X)$ as close as desirable.

SHANNON's NOISELESS CODING THEOREM

SHANNON's noiseless coding theorem says that in order to transmit n values of X, we need, and it is sufficient, to use $nS(X)$ bits.

More exactly, we cannot do better than the bound $nS(X)$ says, and we can reach the bound $nS(X)$ as close as desirable.

Example: Let a source X produce the value 1 with probability $p = \frac{1}{4}$
and the value 0 with probability $1 - p = \frac{3}{4}$

SHANNON's NOISELESS CODING THEOREM

SHANNON's noiseless coding theorem says that in order to transmit n values of X, we need, and it is sufficient, to use $nS(X)$ bits.

More exactly, we cannot do better than the bound $nS(X)$ says, and we can reach the bound $nS(X)$ as close as desirable.

Example: Let a source X produce the value 1 with probability $p = \frac{1}{4}$
and the value 0 with probability $1 - p = \frac{3}{4}$

Assume we want to encode blocks of the outputs of X of length 4.

SHANNON's NOISELESS CODING THEOREM

SHANNON's noiseless coding theorem says that in order to transmit n values of X, we need, and it is sufficient, to use $nS(X)$ bits.

More exactly, we cannot do better than the bound $nS(X)$ says, and we can reach the bound $nS(X)$ as close as desirable.

Example: Let a source X produce the value 1 with probability $p = \frac{1}{4}$ and the value 0 with probability $1 - p = \frac{3}{4}$

Assume we want to encode blocks of the outputs of X of length 4.

By SHANNON's theorem we need $4H(\frac{1}{4}) = 3.245$ bits per blocks (in average)

SHANNON's NOISELESS CODING THEOREM

SHANNON's noiseless coding theorem says that in order to transmit n values of X, we need, and it is sufficient, to use $nS(X)$ bits.

More exactly, we cannot do better than the bound $nS(X)$ says, and we can reach the bound $nS(X)$ as close as desirable.

Example: Let a source X produce the value 1 with probability $p = \frac{1}{4}$ and the value 0 with probability $1 - p = \frac{3}{4}$

Assume we want to encode blocks of the outputs of X of length 4.

By SHANNON's theorem we need $4H(\frac{1}{4}) = 3.245$ bits per blocks (in average)

A simple and practical method known as **Huffman code** requires in this case 3.273 bits per a 4-bit message.

mess.	code	mess.	code	mess.	code	mess.	code
0000	10	0100	010	1000	011	1100	11101
0001	000	0101	11001	1001	11011	1101	111110
0010	001	0110	11010	1010	11100	1110	111101
0011	11000	0111	1111000	1011	111111	1111	1111001

SHANNON's NOISELESS CODING THEOREM

SHANNON's noiseless coding theorem says that in order to transmit n values of X, we need, and it is sufficient, to use $nS(X)$ bits.

More exactly, we cannot do better than the bound $nS(X)$ says, and we can reach the bound $nS(X)$ as close as desirable.

Example: Let a source X produce the value 1 with probability $p = \frac{1}{4}$ and the value 0 with probability $1 - p = \frac{3}{4}$

Assume we want to encode blocks of the outputs of X of length 4.

By SHANNON's theorem we need $4H(\frac{1}{4}) = 3.245$ bits per blocks (in average)

A simple and practical method known as **Huffman code** requires in this case 3.273 bits per a 4-bit message.

mess.	code	mess.	code	mess.	code	mess.	code
0000	10	0100	010	1000	011	1100	11101
0001	000	0101	11001	1001	11011	1101	111110
0010	001	0110	11010	1010	11100	1110	111101
0011	11000	0111	1111000	1011	111111	1111	1111001

Observe that this is a **prefix code** - no codeword is a prefix of another codeword.

DESIGN of HUFFMAN CODE II

Given a sequence of n objects, $x_1, \ldots, x_n$ with probabilities $p_1 \geq \ldots \geq p_n$.

DESIGN of HUFFMAN CODE II

Given a sequence of n objects, $x_1, \ldots, x_n$ with probabilities $p_1 \geq \ldots \geq p_n$.

Stage 1 - shrinking of the sequence.

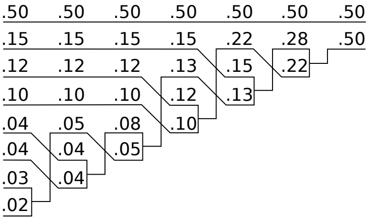
- Replace x_{n-1}, x_n with a new object y_{n-1} with probability $p_{n-1} + p_n$ and rearrange sequence so one has again non-increasing probabilities.

DESIGN of HUFFMAN CODE II

Given a sequence of n objects, $x_1, \dots, x_n$ with probabilities $p_1 \geq \dots \geq p_n$.

Stage 1 - shrinking of the sequence.

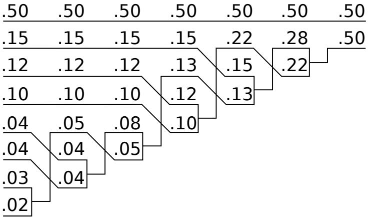
- Replace x_{n-1}, x_n with a new object y_{n-1} with probability $p_{n-1} + p_n$ and rearrange sequence so one has again non-increasing probabilities.
- Keep doing the above step till the sequence shrinks to two objects.



Given a sequence of n objects, $x_1, \ldots, x_n$ with probabilities $p_1 \geq \ldots \geq p_n$.

Stage 1 - shrinking of the sequence.

- Replace x_{n-1}, x_n with a new object y_{n-1} with probability $p_{n-1} + p_n$ and rearrange sequence so one has again non-increasing probabilities.
- Keep doing the above step till the sequence shrinks to two objects.

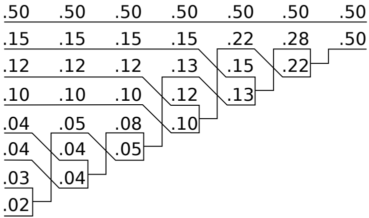


DESIGN of HUFFMAN CODE II

Given a sequence of n objects, $x_1, \ldots, x_n$ with probabilities $p_1 \geq \ldots \geq p_n$.

Stage 1 - shrinking of the sequence.

- Replace x_{n-1}, x_n with a new object y_{n-1} with probability $p_{n-1} + p_n$ and rearrange sequence so one has again non-increasing probabilities.
- Keep doing the above step till the sequence shrinks to two objects.



Stage 2 - extending the code - Apply again and again the following method.

DESIGN of HUFFMAN CODE II

Given a sequence of n objects, $x_1, \ldots, x_n$ with probabilities $p_1 \geq \ldots \geq p_n$.

Stage 1 - shrinking of the sequence.

- Replace x_{n-1}, x_n with a new object y_{n-1} with probability $p_{n-1} + p_n$ and rearrange sequence so one has again non-increasing probabilities.
- Keep doing the above step till the sequence shrinks to two objects.

.50	.50	.50	.50	.50	.50	.50
.15	.15	.15	.15	.22	.28	.50
.12	.12	.12	.13	.15	.22	
.10	.10	.10	.12	.13		
.04	.05	.08	.10			
.04	.04	.05				
.03	.04					
.02						

Stage 2 - extending the code - Apply again and again the following method.

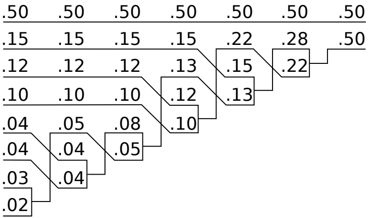
If $C = \{c_1, \ldots, c_r\}$ is a prefix optimal code for a source S_r, then $C' = \{c'_1, \ldots, c'_{r+1}\}$ is an optimal code for S_{r+1}, where

DESIGN of HUFFMAN CODE II

Given a sequence of n objects, $x_1, \ldots, x_n$ with probabilities $p_1 \geq \ldots \geq p_n$.

Stage 1 - shrinking of the sequence.

- Replace x_{n-1}, x_n with a new object y_{n-1} with probability $p_{n-1} + p_n$ and rearrange sequence so one has again non-increasing probabilities.
- Keep doing the above step till the sequence shrinks to two objects.



Stage 2 - extending the code - Apply again and again the following method.

If $C = \{c_1, \ldots, c_r\}$ is a prefix optimal code for a source S_r, then $C' = \{c'_1, \ldots, c'_{r+1}\}$ is an optimal code for S_{r+1}, where

$$\begin{gathered} c'_i = c_i \quad 1 \leq i \leq r-1 \\ c'_r = c_r 1 \\ c'_{r+1} = c_r 0. \end{gathered}$$

DESIGN of HUFFMAN CODE II

DESIGN of HUFFMAN CODE II

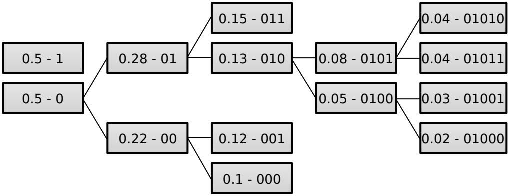
Stage 2 Apply again and again the following method:

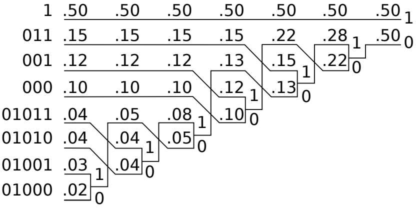
DESIGN of HUFFMAN CODE II

Stage 2 Apply again and again the following method:

If $C = \{c_1, \ldots, c_r\}$ is a prefix optimal code for a source S_r, then $C' = \{c'_1, \ldots, c'_{r+1}\}$ is an optimal code for S_{r+1}, where

$$c'_i = c_i \quad 1 \leq i \leq r-1$$
$$c'_r = c_r 1$$
$$c'_{r+1} = c_r 0.$$





A BIT OF HISTORY I

The subject of error-correcting codes arose originally as a response to practical problems in the reliable communication of digitally encoded information.

A BIT OF HISTORY I

The subject of error-correcting codes arose originally as a response to practical problems in the reliable communication of digitally encoded information.

The discipline was initiated in the paper

Claude Shannon: A mathematical theory of communication, Bell SST.Tech. Journal V27, 1948, 379-423, 623-656

A BIT OF HISTORY I

The subject of error-correcting codes arose originally as a response to practical problems in the reliable communication of digitally encoded information.

The discipline was initiated in the paper

Claude Shannon: A mathematical theory of communication, Bell SST.Tech. Journal V27, 1948, 379-423, 623-656

SHANNON's paper started the scientific discipline **information theory** and **error-correcting codes** are its part.

A BIT OF HISTORY I

The subject of error-correcting codes arose originally as a response to practical problems in the reliable communication of digitally encoded information.

The discipline was initiated in the paper

Claude Shannon: A mathematical theory of communication, Bell SST.Tech. Journal V27, 1948, 379-423, 623-656

SHANNON's paper started the scientific discipline **information theory** and **error-correcting codes** are its part.

Originally, information theory was a part of electrical engineering.

A BIT OF HISTORY I

The subject of error-correcting codes arose originally as a response to practical problems in the reliable communication of digitally encoded information.

The discipline was initiated in the paper

Claude Shannon: A mathematical theory of communication, Bell SST.Tech. Journal V27, 1948, 379-423, 623-656

SHANNON's paper started the scientific discipline **information theory** and **error-correcting codes** are its part.

Originally, information theory was a part of electrical engineering. Nowadays, it is an important part of mathematics and also of informatics.

APPENDIX II- ENTROPY - basics - I.

The concept of **ENTROPY** is one of the most basic and important in modern science, especially in physics, mathematics and information theory.

So called **physical entropy** is a measure of the unavailable energy in a closed thermodynamics system (that is usually considered to be a measure of the system's disorder).

Entropy of an object is a measure of the amount of energy in the object which is unable to do some work.

Entropy is also a measure of the number of possible arrangements of the atoms a system can have.

Information entropy

So called **information** **entropy** is a measure of uncertainty and randomness.

So called **information entropy** is a measure of uncertainty and randomness.

Example If we have a process (a random variable) X producing values 0 and 1, both with probability $\frac{1}{2}$, then we are completely uncertain what will be the next value produced by the process.

So called **information entropy** is a measure of uncertainty and randomness.

Example If we have a process (a random variable) X producing values 0 and 1, both with probability $\frac{1}{2}$, then we are completely uncertain what will be the next value produced by the process.

On the other side, if we have a process (random variable) Y producing value 0 with probability $\frac{1}{4}$ and value 1 with probability $\frac{3}{4}$, then we are more certain that the next value of the process will be 1 than 0.

History Rudolf Clausius coined the term **entropy** in 1865.

SHANNON's VIEW

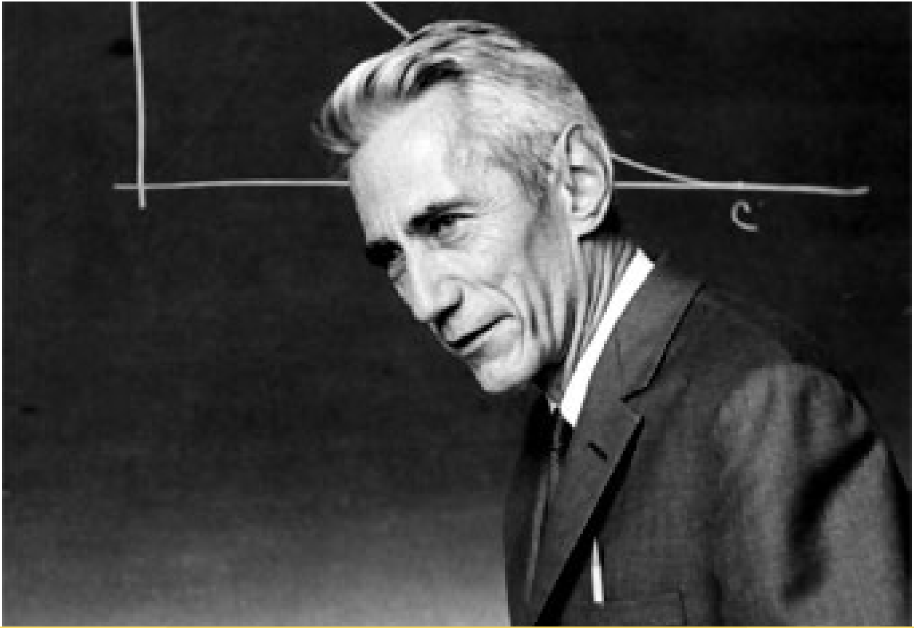
SHANNON's VIEW

In the introduction to his seminal paper "A mathematical theory of communication" Shannon wrote:

SHANNON's VIEW

In the introduction to his seminal paper "A mathematical theory of communication" Shannon wrote:

The fundamental problem of communication is that of reproducing at one point either exactly or approximately a message selected at another point.

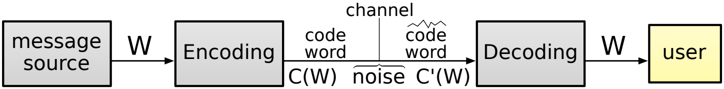


APPENDIX III. SOFT DECODING

APPENDIX

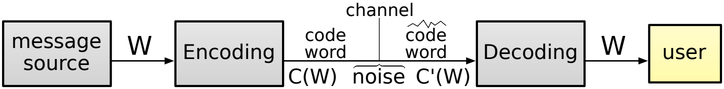
HARD VERSUS SOFT DECODING I

At the beginning of this chapter the process **encoding-channel transmission-decoding** was illustrated as follows:



HARD VERSUS SOFT DECODING I

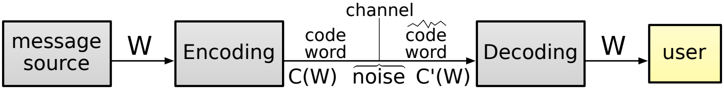
At the beginning of this chapter the process **encoding-channel transmission-decoding** was illustrated as follows:



In that process a binary message is at first encoded into a binary codeword, then transmitted through a noisy channel, and, finally, the decoder receives, for decoding, a potentially erroneous binary message and makes an error correction.

HARD VERSUS SOFT DECODING I

At the beginning of this chapter the process **encoding-channel transmission-decoding** was illustrated as follows:

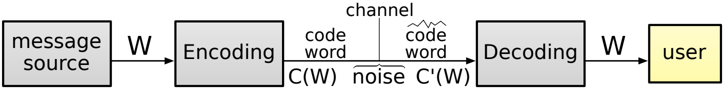


In that process a binary message is at first encoded into a binary codeword, then transmitted through a noisy channel, and, finally, the decoder receives, for decoding, a potentially erroneous binary message and makes an error correction.

This is a simplified view of the whole process.

HARD VERSUS SOFT DECODING I

At the beginning of this chapter the process **encoding-channel transmission-decoding** was illustrated as follows:



In that process a binary message is at first encoded into a binary codeword, then transmitted through a noisy channel, and, finally, the decoder receives, for decoding, a potentially erroneous binary message and makes an error correction.

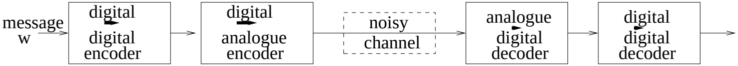
This is a simplified view of the whole process. **In practice the whole process looks quite differently.**

HARD versus SOFT DECODING II

Here is a more realistic view of the whole encoding-transmission-decoding process:

HARD versus SOFT DECODING II

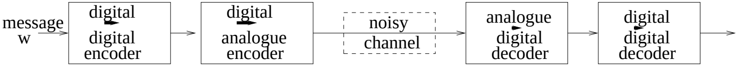
Here is a more realistic view of the whole encoding-transmission-decoding process:



that is

HARD versus SOFT DECODING II

Here is a more realistic view of the whole encoding-transmission-decoding process:

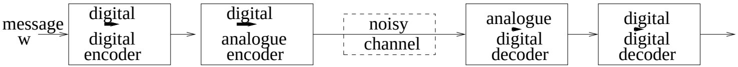


that is

- a binary message is at first transferred to a binary codeword;

HARD versus SOFT DECODING II

Here is a more realistic view of the whole encoding-transmission-decoding process:

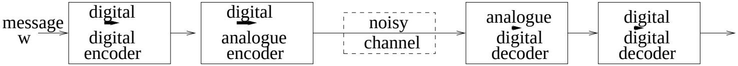


that is

- a binary message is at first transferred to a binary codeword;
- the binary codeword is then transferred to an analogue signal;

HARD versus SOFT DECODING II

Here is a more realistic view of the whole encoding-transmission-decoding process:

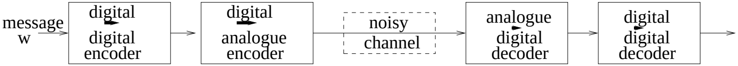


that is

- a binary message is at first transferred to a binary codeword;
- the binary codeword is then transferred to an analogue signal;
- the analogue signal is then transmitted through a noisy channel

HARD versus SOFT DECODING II

Here is a more realistic view of the whole encoding-transmission-decoding process:

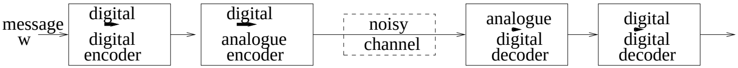


that is

- a binary message is at first transferred to a binary codeword;
- the binary codeword is then transferred to an analogue signal;
- the analogue signal is then transmitted through a noisy channel
- the received analogous signal is then transferred to a binary form that can be used for decoding and, finally

HARD versus SOFT DECODING II

Here is a more realistic view of the whole encoding-transmission-decoding process:



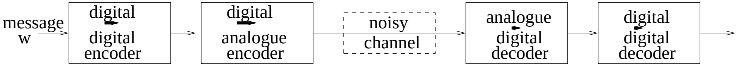
that is

- a binary message is at first transferred to a binary codeword;
- the binary codeword is then transferred to an analogue signal;
- the analogue signal is then transmitted through a noisy channel
- the received analogous signal is then transferred to a binary form that can be used for decoding and, finally
- decoding takes place.

In case the analogous noisy signal is transferred before decoding to the binary signal we talk about a **hard decoding**;

HARD versus SOFT DECODING II

Here is a more realistic view of the whole encoding-transmission-decoding process:



that is

- a binary message is at first transferred to a binary codeword;
- the binary codeword is then transferred to an analogue signal;
- the analogue signal is then transmitted through a noisy channel
- the received analogous signal is then transferred to a binary form that can be used for decoding and, finally
- decoding takes place.

In case the analogous noisy signal is transferred before decoding to the binary signal we talk about a **hard decoding**;
In case the output of analogous-digital decoding is a pair (p_b, b) where p_b is the probability that the output is the bit b (or a weight of such a binary output (often given by a number from an interval $(-V_{max}, V_{max})$), we talk about a **soft decoding**.

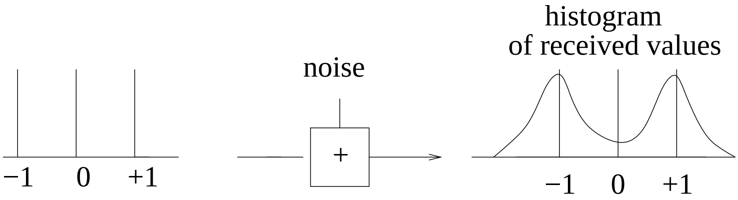
HARD versus SOFT DECODING - III.

In order to deal with such a more general model of transmission and soft decoding, it is common to use, instead of the binary symbols 0 and 1 so-called **antipodal binary symbols** $+1$ and -1 that are represented electronically by voltage $+1$ and -1.

HARD versus SOFT DECODING - III.

In order to deal with such a more general model of transmission and soft decoding, it is common to use, instead of the binary symbols 0 and 1 so-called **antipodal binary symbols** $+1$ and -1 that are represented electronically by voltage $+1$ and -1.

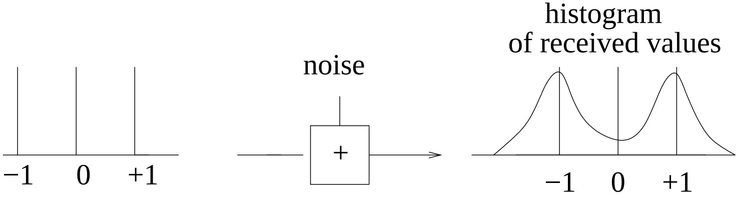
A transmission channel with analogue antipodal signals can then be depicted as follows.



HARD versus SOFT DECODING - III.

In order to deal with such a more general model of transmission and soft decoding, it is common to use, instead of the binary symbols 0 and 1 so-called **antipodal binary symbols** $+1$ and -1 that are represented electronically by voltage $+1$ and -1.

A transmission channel with analogue antipodal signals can then be depicted as follows.



A very important case in practise, especially for space communication, is so-called additive white Gaussian noise (AWN) and the channel with such a noise is called Gaussian channel.

HARD versus SOFT DECODING - COMMENTS

When the signal received by the decoder comes from a devise capable of producing estimations of an analogue nature on the binary transmitted data the error correction capability of the decoder can greatly be improved.

HARD versus SOFT DECODING - COMMENTS

When the signal received by the decoder comes from a devise capable of producing estimations of an analogue nature on the binary transmitted data the error correction capability of the decoder can greatly be improved.

Since the decoder has in such a case an information about the reliability of data received, decoding on the basis of finding the codeword with minimal Hamming distance does not have to be optimal and the optimal decoding may depend on the type of noise involved.

HARD versus SOFT DECODING - COMMENTS

When the signal received by the decoder comes from a devise capable of producing estimations of an analogue nature on the binary transmitted data the error correction capability of the decoder can greatly be improved.

Since the decoder has in such a case an information about the reliability of data received, decoding on the basis of finding the codeword with minimal Hamming distance does not have to be optimal and the optimal decoding may depend on the type of noise involved.

For example, in an important practical case of the Gaussian white noise one search at the minimal likelihood decoding for a codeword with minimal Euclidean distance.

BASIC FAMILIES of CODES

Two basic families of codes are

BASIC FAMILIES of CODES

Two basic families of codes are

Block codes called also as **algebraic codes** that are appropriate to encode blocks of date of the same length and independent one from the other.

BASIC FAMILIES of CODES

Two basic families of codes are

Block codes called also as **algebraic codes** that are appropriate to encode blocks of date of the same length and independent one from the other. Their encoders have often a huge number of internal states and decoding algorithms are based on techniques specific for each code.

BASIC FAMILIES of CODES

Two basic families of codes are

Block codes called also as **algebraic codes** that are appropriate to encode blocks of date of the same length and independent one from the other. Their encoders have often a huge number of internal states and decoding algorithms are based on techniques specific for each code.

Stream codes called also as **convolution codes** that are used to protect continuous flows of data.

BASIC FAMILIES of CODES

Two basic families of codes are

Block codes called also as **algebraic codes** that are appropriate to encode blocks of date of the same length and independent one from the other. Their encoders have often a huge number of internal states and decoding algorithms are based on techniques specific for each code.

Stream codes called also as **convolution codes** that are used to protect continuous flows of data.Their encoders often have only small number of internal states and then decoders can use a complete representation of states using so called *trellises*, iterative approaches via several simple decoders and an exchange of information of probabilistic nature.

BASIC FAMILIES of CODES

Two basic families of codes are

Block codes called also as **algebraic codes** that are appropriate to encode blocks of date of the same length and independent one from the other. Their encoders have often a huge number of internal states and decoding algorithms are based on techniques specific for each code.

Stream codes called also as **convolution codes** that are used to protect continuous flows of data.Their encoders often have only small number of internal states and then decoders can use a complete representation of states using so called *trellises*, iterative approaches via several simple decoders and an exchange of information of probabilistic nature.

Hard decoding is used mainly for block codes and soft one for stream codes. However, distinctions between these two families of codes are tending to blur.

STORY of MORSE TELEGRAPH - I.

STORY of MORSE TELEGRAPH - I.

- In 1825 William Sturgeon discovered electromagnet and showed that using electricity one can make to ring a ring that was far away.

STORY of MORSE TELEGRAPH - I.

- In 1825 William Sturgeon discovered electromagnet and showed that using electricity one can make to ring a ring that was far away.
- The first telegraph designed Charles Whether Stone and demonstrated it at the distance 2.4 km.

STORY of MORSE TELEGRAPH - I.

- In 1825 William Sturgeon discovered electromagnet and showed that using electricity one can make to ring a ring that was far away.
- The first telegraph designed Charles Whether Stone and demonstrated it at the distance 2.4 km.
- Samuel Morse made a significant improvement by designing a telegraph that could not only send information, but using a magnet at other end it could also write the transmitted symbol on a paper.

STORY of MORSE TELEGRAPH - I.

- In 1825 William Sturgeon discovered electromagnet and showed that using electricity one can make to ring a ring that was far away.
- The first telegraph designed Charles Whether Stone and demonstrated it at the distance 2.4 km.
- Samuel Morse made a significant improvement by designing a telegraph that could not only send information, but using a magnet at other end it could also write the transmitted symbol on a paper.
- Morse was a portrait painter whose hobby were electrical machines.

STORY of MORSE TELEGRAPH - I.

- In 1825 William Sturgeon discovered electromagnet and showed that using electricity one can make to ring a ring that was far away.
- The first telegraph designed Charles Whether Stone and demonstrated it at the distance 2.4 km.
- Samuel Morse made a significant improvement by designing a telegraph that could not only send information, but using a magnet at other end it could also write the transmitted symbol on a paper.
- Morse was a portrait painter whose hobby were electrical machines.
- Morse and his assistant Alfredvail invented "Morse alphabet" around 1842.

STORY of MORSE TELEGRAPH - I.

- In 1825 William Sturgeon discovered electromagnet and showed that using electricity one can make to ring a ring that was far away.
- The first telegraph designed Charles Whether Stone and demonstrated it at the distance 2.4 km.
- Samuel Morse made a significant improvement by designing a telegraph that could not only send information, but using a magnet at other end it could also write the transmitted symbol on a paper.
- Morse was a portrait painter whose hobby were electrical machines.
- Morse and his assistant Alfredvail invented "Morse alphabet" around 1842.
- After US Congress approved 30,000 $ on 3.3.1943 for building a telegraph connection between Washington and Baltimore, the line was built fast, and already on 24.5.1943 the first telegraph message was sent: "What Hath God Wrought" - "Čo Boh sent".

STORY of MORSE TELEGRAPH - I.

- In 1825 William Sturgeon discovered electromagnet and showed that using electricity one can make to ring a ring that was far away.
- The first telegraph designed Charles Whether Stone and demonstrated it at the distance 2.4 km.
- Samuel Morse made a significant improvement by designing a telegraph that could not only send information, but using a magnet at other end it could also write the transmitted symbol on a paper.
- Morse was a portrait painter whose hobby were electrical machines.
- Morse and his assistant Alfredvail invented "Morse alphabet" around 1842.
- After US Congress approved 30,000 $ on 3.3.1943 for building a telegraph connection between Washington and Baltimore, the line was built fast, and already on 24.5.1943 the first telegraph message was sent: "What Hath God Wrought" - "Čo Boh sent".
- The era of Morse telegraph ended on 26.1.2006 when the main telegraph company in US, Western Union, announced cancelation of all telegraph services.

STORY of MORSE TELEGRAPH - II.

STORY of MORSE TELEGRAPH - II.

In his telegraphs Moor se used the following two-character audio alphabet

- TIT or dot — a short tone lasting four hundredths of second;
- TAT or dash — a long tone lasting twelve hundredths of second.

In his telegraphs Moor se used the following two-character audio alphabet

- TIT or dot — a short tone lasting four hundredths of second;
- TAT or dash — a long tone lasting twelve hundredths of second.

Morse could called these tones as 0 and 1

STORY of MORSE TELEGRAPH - II.

In his telegraphs Moor se used the following two-character audio alphabet

- TIT or dot — a short tone lasting four hundredths of second;
- TAT or dash — a long tone lasting twelve hundredths of second.

Morse could called these tones as 0 and 1

The binary elements 0 and 1 were first called bits by J. W. Tickle in 1943.

The ISBN-code I

Each book till 1.1.2007 had Iinternational Sstandard BOKO Number which was a 10-digit codeword produced by the publisher with the following structure:

The ISBN-code I

Each book till 1.1.2007 had Iinternational Sstandard BOKO Number which was a 10-digit codeword produced by the publisher with the following structure:

l	p	m	w	$= x_1 \dots x_{10}$
language	publisher	number	weighted check sum	
0	07	709503		

such that

The ISBN-code I

Each book till 1.1.2007 had Iinternational Sstandard BOKO Number which was a 10-digit codeword produced by the publisher with the following structure:

l	p	m	w	$= x_1 \ldots x_{10}$
language	publisher	number	weighted check sum	
0	07	709503		

such that $\sum_{i=1}^{10}(11-i)x_i \equiv 0 \ (mod 11)$

The publisher has to put $x_{10} = X$ if x_{10} is to be 10.

The ISBN-code I

Each book till 1.1.2007 had Iinternational Sstandard BOKO Number which was a 10-digit codeword produced by the publisher with the following structure:

l	p	m	w	$= x_1 \dots x_{10}$
language	publisher	number	weighted check sum	
0	07	709503		

such that $\sum_{i=1}^{10}(11-i)x_i \equiv 0 \ (mod 11)$

The publisher has to put $x_{10} = X$ if x_{10} is to be 10.
The ISBN code was designed to detect: (a) any single error (b) any double error created by a transposition

The ISBN-code I

Each book till 1.1.2007 had Iinternational Sstandard BOKO Number which was a 10-digit codeword produced by the publisher with the following structure:

l	p	m	w	$= x_1 \ldots x_{10}$
language	publisher	number	weighted check sum	
0	07	709503		

such that $\sum_{i=1}^{10}(11-i)x_i \equiv 0 \ (mod 11)$

The publisher has to put $x_{10} = X$ if x_{10} is to be 10.
The ISBN code was designed to detect: (a) any single error (b) any double error created by a transposition

The ISBN-code I

Each book till 1.1.2007 had Iinternational Sstandard BOKO Number which was a 10-digit codeword produced by the publisher with the following structure:

l	p	m	w	$= x_1 \dots x_{10}$
language	publisher	number	weighted check sum	
0	07	709503		

such that $\sum_{i=1}^{10}(11-i)x_i \equiv 0\ (mod 11)$

The publisher has to put $x_{10} = X$ if x_{10} is to be 10.
The ISBN code was designed to detect: (a) any single error (b) any double error created by a transposition Let $X = x_1 \dots x_{10}$ be a correct code and let

$$Y = x_1 \dots x_{j-1} y_j x_{j+1} \dots x_{10} \text{ with } y_j = x_j + a, a \neq 0$$

The ISBN-code I

Each book till 1.1.2007 had Iinternational Sstandard BOKO Number which was a 10-digit codeword produced by the publisher with the following structure:

l	p	m	w	$= x_1 \dots x_{10}$
language	publisher	number	weighted check sum	
0	07	709503		

such that $\quad \sum_{i=1}^{10}(11-i)x_i \equiv 0 \ (mod 11)$

The publisher has to put $x_{10} = X$ if x_{10} is to be 10.
The ISBN code was designed to detect: (a) any single error (b) any double error created by a transposition Let $X = x_1 \dots x_{10}$ be a correct code and let

$$Y = x_1 \dots x_{j-1} y_j x_{j+1} \dots x_{10} \text{ with } y_j = x_j + a, a \neq 0$$

In such a case:

$$\sum_{i=1}^{10}(11-i)y_i = \sum_{i=1}^{10}(11-i)x_i + (11-j)a \neq 0 \ (mod 11)$$

The ISBN-code II

The ISBN-code II

Transposition detection

The ISBN-code II

Transposition detection

Let x_j and x_k be exchanged.

$\sum_{i=1}^{10}(11-i)y_i = \sum_{i=1}^{10}(11-i)x_i + (k-j)x_j + (j-k)x_k = (k-j)(x_j - x_k) \neq 0 \ (mod 11)$

The ISBN-code II

Transposition detection

Let x_j and x_k be exchanged.

$$\sum_{i=1}^{10}(11-i)y_i = \sum_{i=1}^{10}(11-i)x_i + (k-j)x_j + (j-k)x_k = (k-j)(x_j - x_k) \neq 0 \ (mod 11)$$

if $k \neq j$ and $x_j \neq x_k$.

New ISBN code

New ISBN code

Starting 1.1.2007 instead of 10-digit ISBN code a 13-digit ISBN code is being used.

New ISBN code

Starting 1.1.2007 instead of 10-digit ISBN code a 13-digit ISBN code is being used.

New ISBN number can be obtained from the old one by preceding the old code with three digits 978.

New ISBN code

Starting 1.1.2007 instead of 10-digit ISBN code a 13-digit ISBN code is being used.

New ISBN number can be obtained from the old one by preceding the old code with three digits 978.

For details about 13-digit ISBN see

`https://en.wikipedia.org/wiki/International_Standard_Book_Number`