Introduction

Coding Technology

Illés Horváth

2025/09/10

General information

Lecturer: Illés Horváth

E-mail: horvath.illes.antal@gmail.com

Homepage of the course:

https://webspn.hit.bme.hu/~illes/ct/

Office hours: Wednesdays 14:00-15:00, building I, room IB115.

Lectures:

- Wednesdays 10:15-11:45, room QBF09
- Fridays 10:15-11:45 on odd weeks, room QBF09

Practical classes (problem solving):

- Fridays 10:15-11:45 on even weeks, room QBF09, 70% attendance required

Requirements

Final mark is based on a midterm test (50 points) and an exam (50 points).

For the midterm test, 20 points is required to pass. The midterm test can be retaken once for free, and once for an extra fee.

Midterm test: 7 Nov. 2025, 10:15, QBF09 (lecture).
Retake: 20 Nov. 2025, 18:00, room TBA.
Re-retake: 16 Dec. 2025, 12:00, room TBA.

For the exam, 20 points is required to pass.

Maximal total score is 50+50=100. Marks based on the total score are as follows:

- 0–39: 1 (fail)
- 40–54: 2 (pass)
- 55–69: 3 (satisfactory)
- 70–84: 4 (good)
- 85–100: 5 (excellent)

Introduction

Data is an important resource of modern societies. It is an important challenge to store and transmit data in ways that are ...

- reliable $\rightarrow$ Error Control,
- efficient $\rightarrow$ Data Compression,
- private $\rightarrow$ Cryptography.

Coding Theory is the study of codes and their properties for the above applications; it is a very practical scientific field.

Information Theory is more theoretical; it deals with the general study of the quantification and properties of information. That said, Coding Theory and Information Theory are not exclusive.

Introduction

During the Coding Technology codes, our main focus will be on various types of codes:

- Error correction codes adding redundancy to the messages that allow detection and/or correction of errors, allowing reliable communication over noisy channels. Also known as Error Control, or Channel Coding.
- Data compression codes identify patterns and eliminate redundancies in data. Also known as Source Coding.
- Cryptography protocols that turn readable information into unintelligible text, allowing secure private communication over public channels.

In addition to that, we will also discuss the following:

- necessary mathematical background (basic probability, linear algebra, combinatorics. . .)
- architectural considerations (how to implement various methods on modern computer architectures).

Introduction

During the Coding Technology codes, our main focus will be on various types of codes:

- **Error correction codes** adding redundancy to the messages that allow detection and/or correction of errors, allowing reliable communication over noisy channels. Also known as Error Control, or Channel Coding.
- Data compression codes identify patterns and eliminate redundancies in data. Also known as Source Coding.
- Cryptography protocols that turn readable information into unintelligible text, allowing secure private communication over public channels.

In addition to that, we will also discuss the following:

- necessary mathematical background (basic probability, linear algebra, combinatorics. . .)
- architectural considerations (how to implement various methods on modern computer architectures).

General setup

General data transmission scheme (without any coding):

Problem: the channel is noisy, it is possible that the message arrives with some errors. How can we make sure that the receiver gets the original message?

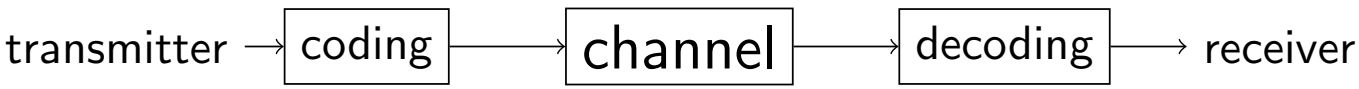
One possible approach is the Automatic Repeat Request (ARQ): the transmitter keeps resending the same message until the receiver acknowledges receiving it without errors.

Properties:

- the receiver must be able to detect if there are errors;
- needs a back channel;
- potentially high latency;
- potentially reduced channel capacity.

Error-correcting codes

Another approach: using error-correcting codes. These codes add some redundant information to the message that allows recovery of the original message even if there are some errors:



Properties:

- no back channel required;
- reduced channel capacity;
- no extra latency;
- the message can be recovered in case of errors (up to a certain number of errors, depending on the coding – tradeoff with reduction in channel capacity!)

It is also important how easy or hard coding and decoding is (computational complexity, memory requirements etc.)

Repeater code

Example. Bob is going to the market. His wife tells him what to buy.

Bob is not sure he heard it correctly, so his wife repeats it. What can happen?

What if Bob hears 'broccoli' once, and 'zucchini' for the second time? He can tell there was an error, but he cannot tell which is the original message.

If Bob hears 'broccoli' both times, then he can be sure he received the message correctly – unless he heard it wrong both times!

Sending the original message repeated is called the Repeater Code.

The $2\times$ repeater code can detect 1 error, but cannot correct 1 error.

Error detection and correction

What if the message is repeated $3\times$?

As long as there are at most 2 errors, Bob can detect that there were errors.

As long as there is at most 1 error, Bob can detect it – and even recover the original message! (How?)

We say that the $3\times$ repeater code can detect 2 errors and correct 1 error.

What about the $n\times$ repeater code? It can detect $n-1$ errors and correct $\lfloor \frac{n-1}{2} \rfloor$ errors.

The repeater code is the most simple error-correcting code, but not very efficient. We aim to study much more efficient error-correcting codes. (However, the repeater code will often be the most simple special case of more complicated error-correcting codes.)

Binary operations

We are going to work with binary numbers (bits): GF(2)=$\{0, 1\}$.
The standard operations are as follows:

$+$	0	1
0	0	1
1	1	0

$\times$	0	1
0	0	0
1	0	1

A binary vector is an ordered list of 0's and 1's:

$$v = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

n elements $\rightarrow$ n-dimensional vector.

Scalar: a single bit.

Vector operations

Addition is element-by-element:

$$\begin{array}{rr} & \begin{bmatrix} 0 & 1 & 0 & 1 & 1 \end{bmatrix} \\ + & \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \end{bmatrix} \\ \hline = & \begin{bmatrix} 1 & 1 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

Only defined if the sizes are identical.

Properties are the same as for usual addition of numbers:

- commutativity: $u + v = v + u$
- associativity: $(u + v) + w = u + (v + w)$

Scalar-vector addition $a + v$ – not defined!

Scalar-vector multiplication $a \cdot v$: each element of v is multiplied by a.

Weight and Hamming-distance

The weight of a binary vector is equal the number of 1's it contains:

$$w([0\,1\,0\,1\,1\,1\,0\,1\,0\,]) = 5.$$

The Hamming-distance of two binary vectors is equal to the number of positions in which they differ:

$$d([0\,1\,0\,1\,1\,1\,0], [1\,1\,0\,1\,1\,0\,0]) = 2.$$

Equivalently, the Hamming-distance can also be written as

$$d(u, v) = w(u - v).$$

Transpose, inner product

- The transpose of a vector v is a column vector v^T:

$$\begin{bmatrix} 1 & 1 & 0 \end{bmatrix}^T = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}$$

- Transposing twice returns the original vector: $(v^T)^T = v$.
- The inner product (dot product/scalar product) of u and v is

$$u \cdot v^T = \sum_{i=1}^{n} u_i v_i = u_1 v_1 + \ldots + u_n v_n$$

- Only defined if the vectors are of the same dimension.
- The result of the inner product is a scalar.

Linear combinations, linear independence

- The linear combination of $v_1, \ldots, v_k$ is the vector

$$\sum_{i=1}^{k} c_i v_i = c_1 v_1 + \ldots + c_k v_k$$

  for some $c_1, \ldots, c_k$ scalars.

- $v_1, \ldots, v_k$ are linearly independent if

$$\sum_{i=1}^{k} c_i v_i = 0 \quad \text{(all 0 vector!)}$$

  only if $c_1 = \ldots = c_k = 0$.

Vector space, linear subspaces

The vector space $\{0,1\}^n$ contains all binary vectors of length n. It is a finite vector space with size 2^n.

A linear subspace is a set of vectors that is closed under linear combinations.

Given a set of vectors $v_1, \ldots, v_k$, the set of all linear combinations of $v_1, \ldots, v_k$ is a linear subspace called the span of $v_1, \ldots, v_k$. The size of the subspace is equal to 2^k where $k \leq n$ is the maximal number of linearly independent vectors among $v_1, \ldots, v_k$. We call k the dimension of the spanned linear subspace.

(Geometry analogy: in the regular 3D space, 1-dimensional subspaces are lines going through the origin, and 2-dimensional subspaces are planes going through the origin.)

Orthogonality

u and v are orthogonal if $uv^T = 0$. Example.

$$u = \begin{bmatrix} 1 & 0 & 0 & 1 \end{bmatrix} \quad v = \begin{bmatrix} 1 & 1 & 0 & 1 \end{bmatrix}$$
$$u^T v = 1 \cdot 1 + 0 \cdot 1 + 0 \cdot 0 + 1 \cdot 1 = 0.$$

If the non-zero $v_1, \ldots, v_k$ are pairwise orthogonal, then they are linearly independent.

For a given set of vectors $v_1, \ldots, v_k$, all vectors orthogonal to $v_1, \ldots, v_k$ form a linear subspace called the orthogonal complement.

The dimensions of the span and orthogonal complement of $v_1, \ldots, v_k$ always add up to the dimension of the entire space.

(Geometry analogy: in the regular 3D space, the orthogonal complement of a line going through the origin is the plane orthogonal to the line and going through the origin.)

Binary matrices

A binary matrix is a 2D array of 0's and 1's, e.g.,

$$A = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

The size of the matrix is #rows $\times$ #columns, e.g. 2×3.

Element (i, j) is in row i, column j, e.g., $A_{13} = 1$.

A $1 \times n$ matrix is a (row) vector.

An $n \times 1$ matrix is a column vector.

Matrix addition and transpose

Matrix addition is element-by-element (same as for vectors); only defined if the sizes are identical.

Its properties are also the same:

- Commutativity: $A + B = B + A$
- Associativity: $(A + B) + C = A + (B + C)$

The transpose of an $m \times n$ matrix A is the $n \times m$ matrix A^T, defined such that

$$A_{ij}^T = A_{ji}, \forall i = 1, \ldots, n, \quad j = 1, \ldots, m$$

Example.

$$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}^T = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 1 & 1 \end{bmatrix}$$

From the definition: $(A^T)^T = A$.

Matrix-matrix multiplication

- The product of $m \times n$ matrix A and $n \times p$ matrix B is the $m \times p$ matrix D such that

$$D_{ij} = \sum_{\ell=1}^{n} A_{i\ell} B_{\ell j}$$

- Basically inner products between the rows of A and the columns of B.
- The number of columns in A and the number of rows in B have to match

Properties:

- Associativity: $(AB)C = A(BC)$
- Transpose: $(AB)^T = B^T A^T$
- but NOT commutative: $AB \neq BA$ in general!

Identity matrix

Identity matrix: square matrix with 1's in the top left to bottom right diagonal, 0's everywhere else.

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

For any square matrix A (of the same size),

$$AI = IA = A.$$

Basic probability

A *probability space* is an $(\Omega, \mathbb{P}, \mathcal{F})$ triple, where

- Ω is the set of outcomes,
- $\mathcal{F}$ is the set of events, which are subsets of Ω, and
- $\mathbb{P}$ is the probability function, defined on $\mathcal{F}$.

Example

Example. We flip a fair coin 3 times. Then

$$\Omega = \{HHH, HHT, HTT, HTH, THH, THT, TTH, TTT\}$$

where H denotes heads, T denotes tails. Example events:

$$A = \text{“the first flip is heads”} = \{HHH, HHT, HTH, HTT\},$$
$$B = \text{“all three flips are the same”} = \{HHH, TTT\}.$$

$$\mathbb{P}(HHH) = \cdots = \mathbb{P}(TTT) = \frac{1}{8},$$

while

$$\mathbb{P}(A) = \frac{4}{8} \quad \text{and} \quad \mathbb{P}(B) = \frac{2}{8}.$$

Properties of the probability function

The probability function always satisfies the following:

$$\mathbb{P}(\emptyset) = 0,$$
$$\mathbb{P}(\Omega) = 1,$$

and if $A_1, A_2, \ldots$ are disjoint events, then

$$\mathbb{P}\Big(\bigcup_{i=1}^{\infty} A_i\Big) = \sum_{i=1}^{\infty} \mathbb{P}(A_i).$$

(This property is called σ-additivity.)

Example

Example. We select a random point from the interval $[0, 1]$ uniformly. Then

$$\Omega = [0, 1],$$

and

$$\begin{aligned} \mathbb{P}(\{0.5\}) &= 0, \\ \mathbb{P}([0.5, 0.7]) &= 0.2, \\ \mathbb{P}([0.5, 0.7] \cup [0.9, 1]) &= 0.3. \end{aligned}$$

Conditional probability

For A and B events, the *conditional probability of A assuming B* is defined as

$$\mathbb{P}(A|B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)}.$$

A and B are *independent* if

$$\mathbb{P}(A \cap B) = \mathbb{P}(A)\mathbb{P}(B),$$

or, equivalently,

$$\mathbb{P}(A|B) = \mathbb{P}(A).$$

Independence is symmetric.

Example

Example. We flip a fair coin three times. What is the conditional probability that the first flip is heads, assuming there are at least 2 heads?

$$A = \text{“the first flip is heads”} = \{HHH, HHT, HTH, HTT\},$$
$$B = \text{“there are at least 2 heads”} = \{HHH, HHT, HTH, THH\},$$

and so

$$A \cap B = \{HHH, HHT, HTH\}.$$

Then

$$\mathbb{P}(A|B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)} = \frac{3/8}{4/8} = \frac{3}{4}.$$

Conditional probability

Theorem (Bayes)

$$\mathbb{P}(B|A) = \frac{\mathbb{P}(A|B)\mathbb{P}(B)}{\mathbb{P}(A)}.$$

$B_1, \ldots, B_k$ are a complete system (set) of events if

$$B_i \cap B_j = \emptyset \text{ for } i \neq j,$$
$$B_1 \cup \cdots \cup B_k = \Omega.$$

Theorem (Total probability)

If $B_1, \ldots, B_k$ is a complete set of events, then

$$\mathbb{P}(A) = \mathbb{P}(A|B_1)\mathbb{P}(B_1) + \cdots + \mathbb{P}(A|B_k)\mathbb{P}(B_k).$$

Example

We have 2 identically-looking dice, one of which is fair (it gives the numbers 1, 2, 3, 4, 5 and 6 with probability $\frac{1}{6} - \frac{1}{6}$ each), but the other one is loaded: 6 has a probability of $\frac{1}{2}$. We pick one of them at random and roll it twice. What is the probability that we roll two sixes? What is the conditional probability of the event that we picked the loaded die, assuming we roll two sixes?

Define the events

$$A = \{\text{we roll two sixes}\},$$
$$B_1 = \{\text{we pick the fair die}\},$$
$$B_2 = \{\text{we pick the loaded die}\}.$$

The information given is the following:

$$\mathbb{P}(B_1) = \mathbb{P}(B_2)$$

because we pick at random,

$$\mathbb{P}(A|B_1) = \left(\frac{1}{6}\right)^2, \qquad \mathbb{P}(A|B_2) = \left(\frac{1}{2}\right)^2.$$

Example

Then we can use total probability to compute

$$\mathbb{P}(A) = \mathbb{P}(A|B_1)\mathbb{P}(B_1) + \mathbb{P}(A|B_2)\mathbb{P}(B_2) = \frac{1}{2} \cdot \frac{1}{36} + \frac{1}{2} \cdot \frac{1}{4} = \frac{10}{72}.$$

Finally, to compute $\mathbb{P}(B_2|A)$ we can use Bayes:

$$\mathbb{P}(B_2|A) = \frac{\mathbb{P}(A|B_2)\mathbb{P}(B_2)}{\mathbb{P}(A)} = \frac{\frac{1}{2} \cdot \frac{1}{4}}{\frac{10}{72}} = \frac{9}{10},$$

so we picked the loaded die with 90% probability.

Binary Symmetric Channel

Now we are ready to discuss the main mathematical model of a noisy channel called the Binary Symmetric Channel (BSC).

In a BSC, each bit sent through the channel may be flipped (changed from 0 to 1, or from 1 to 0).

Errors come from physical properties of the channel. We assume the following:

- the channel is memoryless: whether a bit is changed or not is independent from other bits;
- errors are random and symmetric: each bit will be flipped randomly with some probability, and the probability is the same for a 0 or a 1 bit;
- the channel is homogeneous in time, so the probability of a bit flipping is constant in time.

The flipping probability is called bit error probability, and is denoted by p_b. It is a parameter of the channel.

Binary Symmetric Channel

The value of p_b is between 0 and 1.

If $p_b = 0$, we are happy, no errors occur, error correction is not necessary.

What if $p_b = 1$? The receiver can simply flip every bit back.

If $1/2 < p_b < 1$, the receiver can flip every bit, and this is essentially the same as a channel with $0 < p_b < 1/2$.

Binary Symmetric Channel

What if $p_b = 1/2$? Let u denote the sent bit and v denote the received bit. Then

$$P(v = 0|u = 0) = 1/2, \quad P(v = 1|u = 0) = 1/2,$$
$$P(v = 0|u = 1) = 1/2, \quad P(v = 1|u = 1) = 1/2,$$

so, using total probability,

$$P(v = 0) = 1/2 \cdot P(u = 0) + 1/2 \cdot P(u = 1) = 1/2,$$

and

$$P(v = 0) = P(v = 0|u = 0),$$

so the events $P(v = 0)$ and $P(u = 0)$ are independent $\rightarrow$ we get no information about u from knowing v.

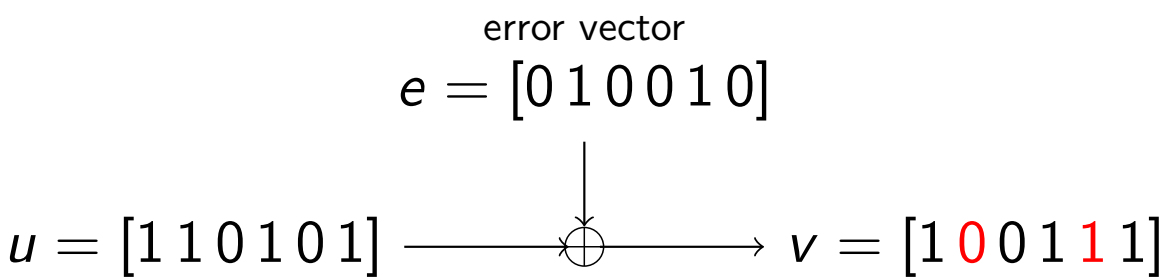
Basically, when $p_b = 1/2$, the output of the channel is independent from the input (white noise). We will generally assume $p_b < 1/2$.

Binary Symmetric Channel

The bit flipping can be mathematically described as an additive error: if an input vector u is sent through the channel, the output vector is

$$v = u \oplus e,$$

where the e error vector contains 1's in positions which are flipped:



Error probability

If the error vector has length n and the bit error probability is P_b, then

$$P(w(e) = i) = \binom{n}{i} P_b^i (1 - P_b)^{n-i} = \binom{n}{i} \left(\frac{P_b}{1 - P_b}\right)^i (1 - P_b)^n.$$

P_b is typically small, and $P(w(e) = i)$ decreases fast in i.

Example. If $n = 3$, $P_b = 0.01$, then

i	0	1	2	3
$P(w(e) = i)$	0.970	0.0294	2.970×10^{-4}	10^{-6}