Information Theory. Synopsis of Lecture

Damian Niwiński, University of Warsaw
Wojciech Jaworski, University of Warsaw

December 10, 2018

Contents

1 Entropy

Je n'ai fait celle-ci plus longue que parce que je n'ai pas eu le loisir de la faire plus courte.

I have made this [letter] longer, because I have not had the time to make it shorter.

Blaise Pascal, *Lettres provinciales*, 1657

Prologue: A precious bit of information

Consider the following puzzle[1]. 100 prisoners on an island wear hats: red or blue. The process of getting these hats was completely random. No one knows his own hat, although everybody sees the hats of others. In order to save their life, *all* prisoners should correctly guess the color of their hats. No communication is possible, and all answers should be given *at once*.

The task seems hopeless, but fortunately, help is coming from the sky. An avionette pilot, who sees everyone, will pass the necessary information to the prisoners through the light signals (red or blue, say). The prisoners expected such a possibility, and could fix some strategy un advance. But time is pressing, so the question is crucial:

How many signals are needed?

As you may guess, the answer is: *just one*. Please explain why[2].

Exercise. If no help comes, the prisoners still have $\frac{1}{2}$ chance to survive, provided they can secretly fix a common strategy, e.g., *let's assume the xor is 0* (say). Show that this is also a *lower bound*, i.e., no strategy can give a guarantee better than $\frac{1}{2}$.

1.1 *Notation*, what is it?

An experiment: guess a *word* which somebody has thought of. Should it work as well with a *number*?

Note that integers written in a positional system are "densely packed", unlike words of natural language. That is, all strings over $\{0, 1, \ldots, 9\}$ denote some numbers (up to leading 0's), while only few strings over $\{a, b, \ldots, z\}$ are (meaningful) words. This last property is in fact fortunate: if we hear someone inexactly, ot if someone makes a typo in e-mail, we can still recover the correct word.

Everyday life examples: writing the amount on cheque both by digits and by words, or spelling a flight number by phone.

Information theory tries to reconcile two antagonistic objectives:

- to make the message as short as possible,
- to prevent errors while the message is sent by an uncertain channel.

Is there any message that we could not make shorter? We are warned by Berry's paradox:

Let n be the smallest integer that cannot be described in English with less than 1000 signs.

[1]It is inspired by *Mathematical Puzzles* by Peter Winkler [4], although the author of these notes has not checked if this problem is in the book.

[2]Coding colors by bits 0 and 1, it is enough to know the *xor* of all colors.

(Thus we have described it.)

The concept of notation should be understood properly. Notation is not a part of an object, but it is given "from outside" to a set of objects, in order to distinguish between them.

Definition 1. *Any 1:1 function $\alpha : S \to \Sigma^*$, where Σ is a finite alphabet, is a* notation for *S.*

How efficient a notation can be? Assume that Σ has at least $r \geq 2$ elements. (The case of $|\Sigma| = 1$ is uninteresting.) The number of strings over Σ shorter than some $n \geq 1$ is

$$1 + r + r^2 + \ldots + r^{n-1} = \frac{r^n - 1}{r - 1} < r^n. \tag{1}$$

Therefore, if α is a notation for a finite set S with $k \geq r^n$ elements, there must be $s \in S$, such that $|\alpha(s)| \geq n$. If, given k, we choose n, such that $r^n \leq k < r^{n+1}$, we may conclude that $|\alpha(s)| \geq \lfloor \log_r k \rfloor$. We thus proved the following.

Proposition 1. *If $\alpha : S \to \Sigma^*$ is notation for a finite set S, with $|S| \geq 1$ and $|\Sigma| \geq 2$ then, for some $s \in S$, $|\alpha(s)| \geq \lfloor \log_r |S| \rfloor$.* □

For the natural numbers, we have another interesting property.

Proposition 2. *If $\alpha : \mathbb{N} \to \Sigma^*$ is notation for natural numbers with $|\Sigma| = r \geq 2$ then, for infinitely many k's, $|\alpha(k)| > \log_r k$.*

Proof. For $n \geq |\alpha(0)| + 1$, let

$$k_n = \min\{k \in \mathbb{N} : |\alpha(k)| \geq n\}.$$

Then clearly $k_n > 0$, and for $i = 0, 1, \ldots, k_n - 1$, the string $\alpha(i)$ is shorter than n, hence by (1) $k_n < r^n$, and consequently

$$\begin{aligned} \log_r k_n &< n \\ &\leq |\alpha(k_n)| \end{aligned}$$

(the last inequality follows from the definition of k_n). Clearly, the set of all such k_n's is infinite, which implies the claim. □

Remark. The estimation of Proposition 2 can be viewed as tight. For example, a natural binary notation

n	0	1	2	3	4	5	6	7	...
$\alpha(n)$	ε	0	1	00	01	10	11	000	...

(given by the formula $\alpha(n) = \{0,1\}^{-1}\text{bin}(n+1)$) satisfies $|\alpha(n)| \leq \lceil \log_2 n \rceil$, for each $n \geq 2$. The intuition behind the numbers k in Proposition 2, satisfying $|\alpha(k)| > \log_r k$, is that they are *hard* for the notation α, as they require long description. In Section 3, where we study a specific notation sending a number n to the shortest program generating it, these numbers will be called *Kolmogorov random*.

As an application, we can see an "information-theoretical" proof of

Proposition 3 (Euclid). *There are infinitely many prime numbers.*

Proof. Suppose to the contrary, that there are only $p_1, \dots, p_M$. This would induce a notation $\alpha : \mathbb{N} \to \{0, 1, \#\}$, for $n = p_1^{\beta_1} \dots p_M^{\beta_M}$,

$$\alpha(n) = bin(\beta_1)\#bin(\beta_2)\# \dots \#bin(\beta_M),$$

where $bin(\beta)$ is the usual binary notation for β ($|bin(\beta)| \leq 1 + \log_2 \beta$). Since $2^{\beta_i} \leq p_i^{\beta_i} \leq n$, we have $\beta_i \leq \log_2 n$, for all i. Consequently

$$|\alpha(n)| \leq M(2 + \log_2 \log_2 n)$$

for all $n > 0$, which clearly contradicts that $|\alpha(n)| \geq \log_3 n$, for infinitely many n's. □

1.2 Codes

Any mapping $\varphi : S \to \Sigma^*$ can be naturally extended to the morphism $\hat{\varphi} : S^* \to \Sigma^*$,

$$\hat{\varphi}(s_1 \dots s_\ell) = \varphi(s_1) \dots \varphi(s_\ell)$$

Definition 2. *A notation $\varphi : S \to \Sigma^*$ for a finite non-empty set S is a* code *if the mapping $\hat{\varphi}$ is 1:1. A code is* prefix–free *if moreover $\neg\varphi(s) \leq \varphi(s')$, for $s \neq s'$. (The notation $x \leq y$ applied to strings means that x is a prefix of y.)*

A prefix-free code is sometimes called *instantaneous*, which is related to the property that a code word $\hat{\varphi}(w)$ can be decoded on-line; more specifically, we have the following property.

Fact 1. *A code φ is prefix-free iff, for any $v, w \in S^*$, $\hat{\varphi}(v) \leq \hat{\varphi}(w)$ implies $v \leq w$.*

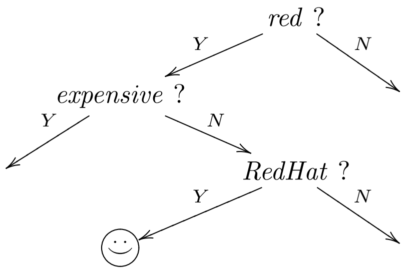
Note that the property of being an (instantaneous) code depends only on the set $\hat{\varphi}(S)$. Notice that $\varepsilon \notin \hat{\varphi}(S)$ (why ?). Any prefix-free set is a code, the set $\{aa, baa, ba\}$ is example of a non-instantaneous code, while $\{a, ab, ba\}$ is not a code at all.

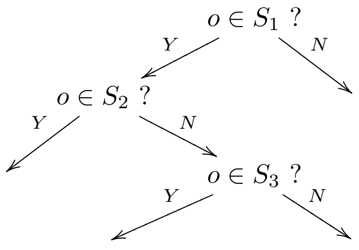
In the sequel we will usually omit "hat" and identify $\hat{\varphi}$ with φ.

We are generally interested in making $|\varphi(w)|$ short, for the sake of storing data, as well as communication. Clearly, in order to encode a set S of m elements with an alphabet Σ of r letters (with $m, r \geq 2$, say), it is enough to use strings of length $\lceil \log_r m \rceil$, so that $|\varphi(w)| \leq |w| \cdot \lceil \log_r m \rceil$, for any $w \in S^*$. However, in order to make the coding more efficient, it is useful to use shorter strings for those elements of S which occur more frequently.

1.2.1 Relation to strategies in 20 question game

There is an analogy between efficient codes and strategies in a so-called *20 question game*. In this game one person invents an object o (presumably, from some large set S), and the remaining players try to guess it by asking questions (normally, up to 20), the answers to which can be only *yes* or *no*. So the questions are generally of the form $o \in S'$?, where $S' \subseteq S$.

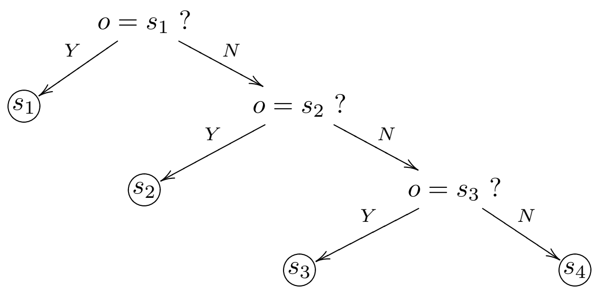




Clearly, $\lceil \log_2 |S| \rceil$ questions suffice to identify any object in S. Can we do better?

In general of course not, since a tree with 2^k leaves must have depth at least k. However, if some objects are more *probable* than others, we can improve the *expected* number of questions. (Besides, this feature makes the real game interesting.)

Suppose the elements of a set $S = \{s_1, s_2, s_3, s_4\}$ are given with probabilities $p(s_1) = \frac{1}{2}$, $p(s_2) = \frac{1}{4}$, $p(s_3) = p(s_4) = \frac{1}{8}$. Then the strategy



guarantees the expected number of questions

$$1 \cdot \frac{1}{2} + 2 \cdot \frac{1}{4} + 3 \cdot \left(\frac{1}{8} + \frac{1}{8}\right) = \frac{7}{4}$$

which is less than $\lceil \log_2 4 \rceil = 2$.

In general, any binary tree with leaves labeled by elements of a finite set S represents some strategy for the game over S (if we neglect the 20 threshold). All questions can be reconstructed bottom-up from the leaves, so we need not bother about them. Identifying directions *left* and *right* with 0 and 1, respectively, we have a mapping $S \to \{0,1\}^*$, which sends each s to the corresponding leaf. In the example above, this would be

$$s_1 \mapsto 0,\ s_2 \mapsto 10,\ s_3 \mapsto 110,\ s_4 \mapsto 111.$$

Clearly, this mapping is an instantaneous code, in which the maximal (expected) length of a code word equals the maximal (expected) number of questions.

Question. Find a strategy in the 20 question game that minimizes the average number of questions.

(We will see an answer in Section 1.3.3.)

The situation can be extended to the case of $|\Sigma| = r \geq 2$. We do not develop a corresponding game, but will often explore the correspondence between instantaneous codes and r-ary trees.

Generally, a *tree over a set* X (or X-tree, for short) is any non-empty set $T \subseteq X^*$ closed under prefix relation (denoted $\leq$). In this context, an element w of T is a *node* of level $|w|$, ε is the *root*, $\leq$-maximal nodes are *leaves*, a node wv (with $w, v \in X^*$) is *below* w, and wx (with $x \in X$) is an immediate *successor* (or child) of w. A *subtree* of T induced by $w \in T$ is $T_w = \{v : wv \in T\}$.

Now, any instantaneous code $\varphi : S \to \Sigma^*$ induces a tree over Σ, $T_\varphi = \{w : \text{for some } s,\ w \leq \varphi(s)\}$. Conversely, any tree $T \subseteq \Sigma^*$ with $|S|$ leaves induces an instantaneous code (and in fact many $|S|!$ functions, depending on permutation of S).

1.2.2 Kraft–McMillan inequality

Our goal is to optimize the length of an instantaneous code, or more generally of any code. We will now see a lower bound for this task. We have already noted in Proposition 1 that, for any notation α for a finite set S, at least one $\alpha(s)$ must be long, therefore $\frac{1}{r^{|\alpha(s)|}}$ must be small. If a notation is a code, we have a somewhat stronger inequality expressed in terms of all fractions $\frac{1}{r^{|\alpha(s)|}}$.

Given a code $\varphi : S \to \Sigma^*$, let $|\varphi| : S \to \mathbb{N}$ denote the *length function*, given by $|\varphi|(s) = |\varphi(s)|$.

Theorem 1 (Kraft's inequality)**.** *Let $2 \leq |S| < \infty$ and $|\Sigma| = r$. A function $\ell : S \to \mathbb{N}$ is the length function, i.e., $\ell = |\varphi|$, for some instantaneous code $\varphi : S \to \Sigma^*$, if and only if*

$$\sum_{s \in S} \frac{1}{r^{\ell(s)}} \quad \leq \quad 1. \tag{2}$$

Proof. ($\Rightarrow$) If all words $\varphi(s)$ have the same length k then, considering that φ is 1:1, we clearly have

$$\sum_{s \in S} \frac{1}{r^{|\varphi(s)|}} \leq \frac{r^k}{r^k} = 1. \ (*)$$

More generally, let k be the maximal length of all $\varphi(s)$'s. For any s with $|\varphi(s)| = i$, let

$$P_s = \{\varphi(s)v : v \in \Sigma^{k-i}\}$$

(in other words, this is the set of nodes of level k below $\varphi(s)$ in the full Σ-tree). Clearly

$$\sum_{w \in P_s} \frac{1}{r^{|w|}} = \frac{r^{k-i}}{r^k} = \frac{1}{r^i}$$

and the sets P_s, $P_{s'}$ are disjoint for $s \neq s'$. Hence again

$$\sum_{s \in S} \frac{1}{r^{|\varphi(s)|}} = \sum_{s \in S} \sum_{w \in P_s} \frac{1}{r^{|w|}} \leq \frac{r^k}{r^k} = 1.$$

($\Leftarrow$) Without loss of generality, we can assume that $S = \{1, \ldots, m\}$, and $\ell(1) \leq \ldots \leq \ell(m)$. For $i = 0, 1, \ldots, m-1$, we inductively define $\varphi(i+1)$ to be the first *lexicographically* element w of $\Sigma^{\ell(i+1)}$ which is not comparable to any of $\varphi(1), \ldots, \varphi(i)$ w.r.t. the prefix ordering $\leq$. It remains to show that there is always such w. Fix $i < m$ and, like in the previous case, let P_j (for $j = 1, \ldots, i$) be the set of nodes of level $\ell(i+1)$ below $\varphi(j)$, we have $|P_j| = r^{\ell(i+1)-\ell(j)}$. We need to verify that

$$r^{\ell(i+1)-\ell(1)} + r^{\ell(i+1)-\ell(2)} + \ldots + r^{\ell(i+1)-\ell(i)} < r^{\ell(i+1)}$$

which amounts to

$$\frac{1}{r^{\ell(1)}} + \frac{1}{r^{\ell(2)}} + \ldots + \frac{1}{r^{\ell(i)}} < 1.$$

This follows directly from the hypothesis; we may assume that the inequality is strict since $i < m$. □

If a code is not instantaneous, the Kraft inequality still holds, but the argument is more subtle.

Theorem 2 (McMillan's inequality)**.** *For any code $\varphi : S \to \Sigma^*$, there is an instantaneous code φ' with $|\varphi| = |\varphi'|$.*

Proof. The case of $|S| = 1$ is trivial, and if $|S| \geq 2$ then $r = |\Sigma| \geq 2$ as well. It is then enough to show that φ satisfies the Kraft inequality. Let $K = \sum_{s \in S} \frac{1}{r^{|\varphi(s)|}}$. Suppose to the contrary that $K > 1$. Let $\mathit{Min} = \min\{|\varphi(s)| : s \in S\}$, $\mathit{Max} = \max\{|\varphi(s)| : s \in S\}$. Consider

$$K^n = \left(\sum_{s \in S} \frac{1}{r^{|\varphi(s)|}} \right)^n = \sum_{i = \mathit{Min} \cdot n}^{\mathit{Max} \cdot n} \frac{N_{n,i}}{r^i},$$

where $N_{n,i}$ is the number of sequences $q_1, \ldots, q_n \in S^n$, such that $i = |\varphi(q_1)| + \ldots + |\varphi(q_n)| = |\varphi(q_1 \ldots q_n)|$. Since φ is a code, at most one such sequence can be encoded by a word in Σ^i, hence

$$\frac{N_{n,i}}{r^i} \leq 1.$$

This follows

$$K^n \leq (\mathit{Max} - \mathit{Min}) \cdot n + 1$$

which clearly fails for sufficiently large n. The contradiction proves that $K \leq 1$. □

1.3 Entropy

Let us come back to *Question* stated in Section 1.2.1. In view of the inequalities established above, we can reformulate it as follows. Let $S = \{s_1, \ldots, s_m\}$ be a set with $m \geq 2$, and let $p : S \to [0,1]$ be a probability distribution on S; we abbreviate $p(s_i) = p_i$. Let $r \geq 2$.

Question. Among the tuples of natural numbers $\ell_1, \ldots, \ell_m$, which satisfy the inequality $\sum_i \frac{1}{r^{\ell_i}} \leq 1$, find the one that minimizes $\sum_i p_i \, \ell_i$.

We need first to recall some concepts from the calculus.

1.3.1 Properties of convex functions

Definition 3. *A function $f : [a,b] \to \mathbb{R}$ is* convex *(on $[a,b]$) if $\forall x_1, x_2 \in [a,b]$, $\forall \lambda \in [0,1]$,*

$$\lambda f(x_1) + (1-\lambda) f(x_2) \quad \geq \quad f(\lambda x_1 + (1-\lambda) x_2). \tag{3}$$

It is strictly convex *if the inequality is strict, except for $\lambda \in \{0,1\}$ and $x_1 = x_2$.*

Geometrically, it means that any chord linking two points of the function graph lies (strictly) above the graph.

Lemma 1. *If f is continuous on $[a,b]$ and has a second derivative on (a,b) with $f'' \geq 0$ ($f'' > 0$) then it is convex (strictly convex).*

Proof. Assume $f'' \geq 0$. Then by the Mean value theorem, f' is weakly increasing on (a,b) (for $a < t_1 < t_2 < b$, $f'(t_2) - f'(t_1) = f''(\tilde{t})(t_2 - t_1) \geq 0$).
Let $x_\lambda = \lambda x_1 + (1-\lambda) x_2$. Rearranging our formula a bit, we have to show

$$\lambda(f(x_\lambda) - f(x_1)) \overset{?}{\leq} (1-\lambda)(f(x_2) - f(x_\lambda)).$$

Using the Mean value theorem, this time for f, it reduces to

$$\begin{aligned} \lambda f'(\tilde{x_1})(x_\lambda - x_1) \quad &\overset{?}{\leq} \quad (1-\lambda) f'(\tilde{x_2})(x_2 - x_\lambda) \\ \lambda(1-\lambda) f'(\tilde{x_1})(x_2 - x_1) \quad &\overset{?}{\leq} \quad \lambda(1-\lambda) f'(\tilde{x_2})(x_2 - x_1), \end{aligned}$$

which holds since f' is weakly increasing. If $f'' > 0$ the argument is similar. □

In this course, unless stated otherwise, we consider only *finite* probabilistic spaces. If we say that X is a *random variable* on S, we tacitly assume that S is given with *probability* mapping $p : S \to [0,1]$ (i.e., $\sum_{s \in S} p(s) = 1$), and $X : S \to \mathbb{R}$.
Recall that the *expected value* of X is

$$EX = \sum_{s \in S} p(s) \cdot X(s).$$

If $S = \{s_1, \ldots, s_m\}$, we adopt the notation $p(s_i) = p_i$, $X(s) = x_i$. In this writing $EX = p_1 x_1 + \ldots + p_m x_m$.
Note that EX does not depend on those x_i's for which $p_i = 0$. We say that X is *constant* if there are no $x_i \neq x_j$ with $p_i, p_j > 0$.

Theorem 3 (Jensen's inequality)**.** *If $f : [a,b] \to \mathbb{R}$ is a convex function then, for any random variable $X : S \to [a,b]$,*

$$Ef(X) \quad \geq f(EX). \tag{4}$$

If moreover f is strictly convex then the inequality is strict unless X is constant.

Proof. By induction on $|S|$. The case of $|S| = 1$ is trivial, and if $|S| = 2$, the inequality amounts to

$$p_1 f(x_1) + p_2 f(x_2) \quad \geq (>) \quad f(p_1 x_1 + p_2 x_2)$$

which is just the definition of (strict) convexity. (Note that X is constant iff $p_1 \in \{0,1\}$ or $x_1 = x_2$.)

Let $S = \{s_1, \ldots, s_m\}$, and suppose the claim holds for any random variables over S', $|S'| \leq m-1$. Without loss of generality we may assume that $p_m < 1$. Let $p_i' = \frac{p_i}{1-p_m}$, for $i = 1, \ldots, m-1$. We have

$$\begin{aligned}
\sum_{i=1}^{m} p_i\, f(x_i) &= p_m f(x_m) + (1-p_m) \sum_{i=1}^{m-1} p_i'\, f(x_i) \\
&\geq p_m f(x_m) + (1-p_m) f\left(\sum_{i=1}^{m-1} p_i'\, x_i \right) \\
&\geq f\left(p_m x_m + (1-p_m) \sum_{i=1}^{m-1} p_i'\, x_i \right) \\
&= f\left(\sum_{i=1}^{m} p_i x_i \right).
\end{aligned}$$

Note that we have used the induction hypothesis twice: for the random variable given by probabilities $p_1', \ldots, p_{m-1}'$ and values $x_1, \ldots, x_{m-1}$, and for the random variable given by probabilities $p_m, 1-p_m$, and values x_m and $\sum_{i=1}^{m-1} p_i' x_i$, respectively.

Now suppose f is strictly convex and in the above the equalities hold. Then the first auxiliary random variable is constant, i.e., $x_i = C$, for all $i = 1, \ldots, m-1$, unless $p_i' = p_i = 0$. Since the second auxiliary random variable must also be constant, we have, whenever $p_m > 0$, $x_m = \sum_{i=1}^{m-1} p_i' x_i = C$, as well. □

Convention We let

$$0 \log_r 0 = 0 \log_r \frac{1}{0} = 0. \tag{5}$$

This is justified by the fact that $\lim_{x \to 0} x \log_r x = \lim_{x \to 0} -x \log_r \frac{1}{x} = \lim_{y \to \infty} -\frac{\log_r y}{y} = 0$.

From Lemma 1, we deduce that, if $r > 1$ then the function $x \log_r x$ is strictly convex on $[0, \infty)$ (i.e., on any $[0, M]$, $M > 0$). Indeed,

$$(x \log_r x)'' = \left(\log_r x + x \cdot \frac{1}{x} \cdot \log_r e \right)' = \frac{1}{x} \cdot \log_r e > 0.$$

1.3.2 Gibbs' inequality

The following theorem, due to J.W.Gibbs[3], is extremely useful and in these notes is usually referred to as *Golden Lemma*.

[3]Josiah Willard Gibbs (1839-1903) was one of the fathers of statistical mechanics, and actually coined this name.

Theorem 4 (Golden Lemma). *Suppose* $1 = \sum_{i=1}^q x_i \geq \sum_{i=1}^q y_i$*, where* $x_i \geq 0$ *and* $y_i > 0$*, for* $i = 1, \ldots, q$*, and let* $r > 1$*. Then*

$$\sum_{i=1}^q x_i \cdot \log_r \frac{1}{y_i} \quad \geq \quad \sum_{i=1}^q x_i \cdot \log_r \frac{1}{x_i},$$

and the equality holds only if $x_i = y_i$*, for* $i = 1, \ldots, q$*.*

Proof. Let us first assume that $\sum_{i=1}^q y_i = 1$. We have

$$\mathit{Left} - \mathit{Right} = \sum_{i=1}^q x_i \cdot \log_r \frac{x_i}{y_i} = \sum_{i=1}^q y_i \cdot \left(\frac{x_i}{y_i}\right) \cdot \log_r \frac{x_i}{y_i}$$

Applying Jensen's inequality to function $x \log_r x$ (on $[0, \infty)$), we get

$$\sum_{i=1}^q y_i \cdot \left(\frac{x_i}{y_i}\right) \cdot \log_r \frac{x_i}{y_i} \geq \log_r \sum_{i=1}^q y_i \cdot \left(\frac{x_i}{y_i}\right) = 0.$$

Here we consider the random variable which takes the value $\left(\frac{x_i}{y_i}\right)$ with probability y_i. As the function $x \log_r x$ is even strictly convex on $[0, \infty)$ (c.f. page 9), the equality implies that this random variable is constant. Remembering that $y_i > 0$, and $\sum_{i=1}^q x_i = \sum_{i=1}^q y_i$, we then have $x_i = y_i$, for $i = 1, \ldots, q$.

Now suppose $\sum_{i=1}^q y_i < 1$. Let $y_{q+1} = 1 - \sum_{i=1}^q y_i$, and $x_{q+1} = 0$. Then, by the previous case we have

$$\sum_{i=1}^q x_i \cdot \log_r \frac{1}{y_i} = \sum_{i=1}^{q+1} x_i \cdot \log_r \frac{1}{y_i} \geq \sum_{i=1}^{q+1} x_i \cdot \log_r \frac{1}{x_i} = \sum_{i=1}^q x_i \cdot \log_r \frac{1}{x_i}.$$

Note that the equality may not hold in this case, as it would imply $x_i = y_i$, for $i = 1, \ldots, q+1$, which contradicts the choice of $y_{q+1} \neq x_{q+1}$. $\square$

The Golden Lemma gives us a partial answer to *Question* stated at the beginning of this section (page 8). Indeed, if the numbers $\ell_1, \ldots, \ell_q$ satisfy Kraft's inequality $\sum_{i=1}^q \frac{1}{r^{\ell_i}} \leq 1$ then letting $x_i = p_i$ and $y_i = \frac{1}{r^{\ell_i}}$, we obtain

$$\sum_{i=1}^q p_i \, \ell_i \quad \geq \quad \sum_{i=1}^q p_i \cdot \log_r \frac{1}{p_i}, \tag{6}$$

and the equality holds only if $p_i = \frac{1}{r^{\ell_i}}$, for $i = 1, \ldots, q$. Hence, if it happens that probabilities are integer powers of $\frac{1}{r}$ then the choice $\ell_i = \log_r p_i$ minimizes $\sum_{i=1}^q p_i \ell_i$.

This is in particular the case of the strategy in our example in section 1.2.1, where $r = 2$ and the probabilities were $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}$. The Golden Lemma implies that this strategy is indeed optimal. If the probabilities are not powers of $\frac{1}{r}$, the right-hand side of the inequality (6) still makes sense, and it leads to the central concept of Information Theory.

1.3.3 Shannon's entropy

Definition 4 (Shannon's entropy). [4] *The* entropy *of a (finite) probabilistic space* S *(with parameter* $r > 1$*) is*

$$H_r(S) \quad = \quad \sum_{s \in S} p(s) \cdot \log_r \frac{1}{p(s)} \tag{7}$$

$$\quad = \quad -\sum_{s \in S} p(s) \cdot \log_r p(s). \tag{8}$$

[4] Claude Shannon (1916-2001) was the inventor of Information theory and one of the fathers of digital computers.

(In the above formula, we use our convention (5).) Traditionally, we abbreviate $H = H_2$.

Remark. We can present $H_r(S)$ as the expected value of a random variable LogP_r defined on S by[5]

$$\mathrm{LogP}_r(s) \quad = \quad \begin{cases} \log_r \frac{1}{p(s)} & \text{if} \quad p(s) \neq 0 \\ 0 & \text{if} \quad p(s) = 0. \end{cases} \tag{9}$$

We have, from definition,

$$H_r(S) \quad = \quad E\,\mathrm{LogP}_r. \tag{10}$$

From general perspective, the use of the function log in the definition of entropy can be seen in context of the so-called *Weber-Fechner law* of cognitive science, stating that the human *perception* (P) of the growth of a physical *stimuli* (S), is proportional to the *relative* growth of the stimuli rather than to its absolute growth,

$$\partial P \quad \approx \quad \frac{\partial S}{S}$$

which, after integration, gives us

$$P \quad \approx \quad \log S.$$

This has been observed in perception of weight, brightness, sound (both intensity and height), and even one's economic status. If we view probability as the measure of frequency, and hence its inverse $\frac{1}{p(s)}$ as the measure of seldomness – or maybe *strangeness* – then the function $\log \frac{1}{p(s)}$ occurring in the equation (7) can be read as our "perception of strangeness".

What values entropy can take, depending on the function p ? From definition we readily have $H_r(S) \geq 0$, and this value is indeed attained if the whole probability is concentrated in one point. On the other hand, we have

Fact 2.

$$H_r(S) \quad \leq \quad \log_r |S| \tag{11}$$

and the equality holds if and only if $p(s) = \frac{1}{|S|}$, for all $s \in S$.

Proof. Indeed, taking in the Golden Lemma $x_i = p(s_i)$ and $y_i = \frac{1}{|S|}$, we obtain

$$\sum_{s \in S} p(s) \cdot \log_r \frac{1}{p(s)} \leq \sum_{s \in S} p(s) \cdot \log_r |S| = \log_r |S|,$$

with the equality for $p(s) = \frac{1}{|S|}$, as desired. □

1.4 Minimal code length

As we have seen, if all probabilities are powers of $\frac{1}{2}$ then the entropy equals to the (average) length of an optimal code. We will see that it is always a lower bound.

[5]The case of $p(s) = 0$ is distinguished for technical reasons, since the expected value is well-defined for random variables with values in $\mathbb{R}$.

Definition 5 (Minimal code length)**.** *Let S be a finite set equipped with a probability distribution p. For a code φ over a set S, let*

$$L(\varphi) \quad = \quad \sum_{s \in S} p(s) \cdot |\varphi(s)|.$$

Given an integer $r \geq 2$, let $L_r(S)$ be the minimum of all $L(\varphi)$'s, where φ ranges over all codes $\varphi : S \to \Sigma^*$, with $|\Sigma| = r$.

Remark. Let us verify that the *minimum* in the definition of $L_r(S)$ indeed exists (not only infimum), i.e., $L_r(S) = L(\varphi)$, for some code $\varphi : S \to \Sigma^*$. If $|S| \leq r$ then obviously $1 = L_r(S) = L(\varphi)$, for some one letter code. Suppose $|S| > r$, we will show that it is enough to consider a finite family of codes. Let $K = L(\psi)$, for some code ψ. Clearly, it is enough to consider codes φ satisfying $L(\varphi) \leq K$. For any such code, and any $s \in S$, we have $|\varphi(s)| \cdot p(s) \leq K$, which gives us a common upper bound on $\varphi(s)$ for those s, for which $p(s) > 0$, i.e.,

$$\max\{|\varphi(s)| : s \in S \text{ and } p(s) > 0\} \quad \leq \quad M, \tag{12}$$

for some M. If there are no s with $p(s) = 0$ then we are done, since obviously only finitely many codes satisfy (12). Suppose however that there is a non-empty subset $S \supseteq Z = \{s : p(s) = 0\}$. Then *a priori* a code may assign arbitrary long words to $s \in Z$. Note however that, in this case, any code satisfies

$$\sum_{s \in S - Z} \frac{1}{r^{|\varphi(s)|}} \quad < \quad 1$$

which implies

$$\min_{\varphi} \left(1 - \sum_{s \in S - Z} \frac{1}{r^{|\varphi(s)|}} \right) > 0,$$

where φ ranges over those codes satisfying (12). Denoting this last value by δ, and choosing Y such that $\sum_{s \in Z} \frac{1}{r^Y} \leq \delta$, we may in Definition 5 above restrict our attention to finitely many codes φ, namely those satisfying (12), and additionally $|\varphi(s)| \leq Y$, for $s \in Z$. $\square$

Note that, because of the McMillan Theorem (page 7), the value of $L_r(S)$ would not change if φ have ranged over instantaneous codes.

Theorem 5. *Let S be a finite probabilistic space and r an integer, with $r \geq 2$. Then*

$$H_r(S) \leq L_r(S) \leq H_r(S) + 1. \tag{13}$$

Moreover, the equality $H_r(S) = L_r(S)$ holds if and only if $|S| \geq 2$ and all probabilities $p(s)$ are integer powers of $\frac{1}{r}$, and the equality $L_r(S) = H_r(S) + 1$ holds if and only if $p(s) = 1$, for some $s \in S$ (equivalently, $H_r(S) = 0$).

Proof. If $|S| = 1$ then obviously $H_r(S) = 0$, and $L_r(S) = 1$. Let us assume $|S| \geq 2$.

For the first inequality, it is enough to show that

$$H_r(S) \quad \leq \quad L(\varphi)$$

holds for any code $\varphi : S \to \Sigma^*$, with $|\Sigma| = r$. We obtain this readily taking in the Golden Lemma $x_i = p(s_i)$ and $y_i = \frac{1}{r^{|\varphi(s)|}}$.

Now, if the equality $H_r(S) = L_r(S)$ holds then we have also $H_r(S) = L(\varphi)$, for some code φ. Again from Golden Lemma, we obtain $p(s) = \frac{1}{r^{|\varphi(s)|}}$, for all $s \in S$.

On the other hand, if each probability $p(s)$ is of the form $\frac{1}{r^{\ell(s)}}$, then by the Kraft inequality, there exists a code φ with $|\varphi(s)| = \ell(s)$, and for this code $L(\varphi) = H_r(S)$. Hence $L_r(S) \leq H_r(S)$, but by the previous inequality, the equality must hold.

For the second inequality of (13), we will construct a code $\varphi : S \to \Sigma^*$ (with $|\Sigma| = r$), satisfying

$$L(\varphi) \quad \leq \quad H_r(S) + 1.$$

(known as *Shannon-Fano coding*). We only construct an appropriate length function ℓ; the existence of a desired code will follow from Kraft's inequality (Theorem 1). We let

$$\ell(s) \quad = \quad \left\lceil \log_r \frac{1}{p(s)} \right\rceil$$

for those $s \in S$ for which $p(s) > 0$. Then

$$\sum_{s:p(s)>0} \frac{1}{r^{\ell(s)}} \leq \sum_{p(s)>0} p(s) = \sum_{s \in S} p(s) = 1.$$

We consider several cases. If $(\forall s \in S)\, p(s) > 0$, then ℓ is defined on the whole S, and the above coincides with the Kraft inequality. But as $\ell(s) < \log_r \frac{1}{p(s)} + 1$, we obtain

$$\sum_{s \in S} p(s) \cdot \ell(s) < \sum_{s \in S} p(s) \cdot \left(\log_r \frac{1}{p(s)} + 1 \right) = H_r(S) + 1.$$

Now suppose that $p(s)$ may be 0, for some s. If

$$\sum_{p(s)>0} \frac{1}{r^{\ell(s)}} \quad < \quad 1,$$

then we can readily extend the definition of ℓ to all s, such that the Kraft inequality $\sum_{s \in S} \frac{1}{r^{\ell(s)}} \leq 1$ is satisfied. Again, there is a code with length ℓ, satisfying $\ell(s) < \log_r \frac{1}{p(s)} + 1$, whenever $p(s) > 0$, and hence

$$\sum_{s \in S} p(s) \cdot \ell(s) < \sum_{s \in S} p(s) \cdot \left(\log_r \frac{1}{p(s)} + 1 \right) = H_r(S) + 1.$$

(Remember our convention that $0 \cdot \log \frac{1}{0} = 0$.)

Finally, suppose that

$$\sum_{p(s)>0} \frac{1}{r^{\ell(s)}} \quad = \quad 1.$$

We choose s' with $p(s') > 0$, and let

$$\begin{aligned} \ell'(s') &= \ell(s') + 1 \\ \ell'(s) &= \ell(s), \text{ for } s \neq s'. \end{aligned}$$

Now again we can extend ℓ' to all s in such a way that the Kraft inequality holds. In order to evaluate the average length of this code, let us first observe that our assumptions yield that $\ell(s) = \log_r \frac{1}{p(s)}$, whenever $p(s) > 0$. (Indeed, we have $\frac{1}{r^{\ell(s)}} \leq p(s)$ by definition of ℓ, and $1 = \sum_{p(s)>0} \frac{1}{r^{\ell(s)}} = \sum_{p(s)>0} p(s)$, hence $p(s) = \frac{1}{r^{\ell(s)}}$, whenever $p(s) > 0$.) Then the code with length ℓ' satisfies

$$\sum_{s \in S} p(s) \cdot \ell'(s) = \sum_{p(s)>0} p(s) \cdot \ell'(s) = p(s') + \sum_{p(s)>0} p(s) \cdot \ell(s) = p(s') + H_r(S).$$

Hence we get $L_r(S) \leq H_r(S) + 1$ and the inequality is strict unless we cannot find s' with $0 < p(s') < 1$. □

Remark. The construction of a code φ in the proof above is not very efficient (it requires knowledge of the $\log_r \frac{1}{p(s)}$); moreover it needs not produce an optimal code. A well-known algorithm of the *Huffman coding* produces an optimal code for binary alphabet (see, e.g., Wikipiedia https://en.wikipedia.org/wiki/Huffman_coding). By Theorem 5, the average length of the Huffman code satisfies the inequality $H_2(\varphi) < H_2(S) + 1$. $\square$

The above theorem may appear somewhat pessimistic, as it implies that in most cases the inequality is strict: $H_r(S) < L_r(S)$, i.e., our coding is "imperfect". Note that probabilities usually are not chosen by us, but rather come from Nature.

However, it turns out that, even with a *fixed* S and p we can, in a sense, bring the average code length closer and closer to $H_r(S)$. This is achieved by some relaxation of the concept of a code.

Example Let $S = \{s_1, s_2\}$ with $p(s_1) = \frac{3}{4}$, $p(s_2) = \frac{1}{4}$. Then clearly $L_2(S) = 1$. However, $H_2(S) < 1$, since the probabilities are not the powers of $\frac{1}{2}$.

This means that we are unable to make the encoding of a message $\alpha \in S^*$ shorter than α itself, even on average. Now, consider the following mapping:

$s_1 s_1 \mapsto 0$	$s_1 s_2 \mapsto 10$
$s_2 s_1 \mapsto 110$	$s_2 s_2 \mapsto 111$

Of course, this is not a code of S, but apparently we could use this mapping to encode sequences over S of even length. Indeed, it *is* a code for the set S^2. Consider $S^2 = S \times S$ as the product (probabilistic) space with

$$p(s_i, s_j) \quad = \quad p(s_i) \cdot p(s_j).$$

Then the average length of our encoding of the *two*-symbols blocks is

$$\left(\frac{3}{4}\right)^2 \cdot 1 + \frac{3}{4} \cdot \frac{1}{4} \cdot (2+3) + \left(\frac{1}{4}\right)^2 \cdot 3 = \frac{9}{16} + \frac{15}{16} + \frac{3}{16} = \frac{27}{16} < 2.$$

As the reader may expect, if we proceed in this vein for $n = 2, 3, \ldots$, we can obtain more and more efficient encoding. But can we overcome the entropy bound, i.e., to get

$$\frac{L_r(S^n)}{n} \quad \overset{?}{<} \quad H_r(S)$$

for some n ?

We will see that this is *not* the case, but the Shannon First Theorem (next lecture) will tell us that the entropy bound can be approached arbitrarily close, as $n \to \infty$.

1.4.1 Entropy of the product space

We first compute the entropy $H(S^n)$ of S^n viewed as the product space with $p(q_1, \ldots, q_n) = p(q_1) \cdot \ldots \cdot p(q_n)$. This could be done by a tedious elementary calculation, but it will be more instructive to deduce the formula from general properties of random variables.

Recall that the expected value of a random variable $X : S \to \mathbb{R}$ (over a finite probabilistic space S) can be presented in two ways, readily equivalent to each other:

$$EX \quad = \quad \sum_{s \in S} p(s) \cdot X(s) \tag{14}$$

$$= \quad \sum_{t \in \mathbb{R}} t \cdot p(X = t). \tag{15}$$

In the last equation we assume that the sum of arbitrarily many[6] 0's is 0, and

$$p(X = t) \quad = \quad \sum_{\{s: X(s) = t\}} p(s). \tag{16}$$

Notational convention. We often need to express the probability that some property $\psi(X)$ holds for a random variable X. Therefore, we extend the notation of (16) to

$$p(\psi(X)) \quad = \quad \sum_{\{s: \psi(X(s)) \textit{ holds true}\}} p(s). \tag{17}$$

The last expression is often abbreviated to $\sum_{\psi(X(s))} p(s)$.

We recall a basic fact from Probability Theory, which follows from the first presentation of the expected value (14).

Linearity of expectation. If X and Y are arbitrary random variables (defined on the same probabilistic space) then, for any $\alpha, \beta \in \mathbb{R}$,

$$E(\alpha X + \beta Y) \quad = \quad \alpha EX + \beta EY. \tag{18}$$

Now consider two probabilistic spaces S and Q. (According to the tradition, if confusion does not arise, we use the same letter p for the probability functions on all spaces.)
Let $S \times Q$ be the product space with the probability given by

$$p(s, q) \quad = \quad p(s) \cdot p(q).$$

Given random variables $X : S \to \mathbb{R}$ and $Y : Q \to \mathbb{R}$, we define the random variables $\hat{X}$, $\hat{Y}$, over $S \times Q$, by

$$\begin{aligned} \hat{X}(s, q) &= X(s) \\ \hat{Y}(s, q) &= Y(q). \end{aligned}$$

Note that

$$p(\hat{X} = t) = \sum_{\hat{X}(s,q)=t} p(s, q) = \sum_{X(s)=t} \sum_{q \in Q} p(s) \cdot p(q) = \sum_{X(s)=t} p(s) \; = \; P(X = t).$$

Similarly, $p(\hat{Y} = t) = p(Y = t)$.
Therefore, $E\hat{X} = EX$ and $E\hat{Y} = EY$ and, by linearity of expectation,

$$E(\hat{X} + \hat{Y}) = E\hat{X} + E\hat{Y} = EX + EY. \tag{19}$$

Now, let $\mathrm{LogP}_r^{(S)}$ be the random variable defined by (9) for the space S, and let $\mathrm{LogP}_r^{(Q)}$ and $\mathrm{LogP}_r^{(S \times Q)}$ be defined similarly for the spaces Q and $S \times Q$, respectively. For any $(s, q) \in S \times Q$, such that $p(s), p(q) > 0$, we have

$$\mathrm{LogP}_r^{(S\times Q)}(s, q) = \log_r \frac{1}{p(s)} \cdot \frac{1}{p(q)} = \log_r \frac{1}{p(s)} + \log_r \frac{1}{p(q)} = \widehat{\mathrm{LogP}_r^{(S)}}(s, q) + \widehat{\mathrm{LogP}_r^{(Q)}}(s, q). \tag{20}$$

The random variables $\mathrm{LogP}_r^{(S\times Q)}$ and $\widehat{\mathrm{LogP}_r^{(S)}} + \widehat{\mathrm{LogP}_r^{(Q)}}$ may differ for those (s, q), for which $p(s) \cdot p(q) = 0$, but (20) combined with (19) and (10) is enough to conclude

$$H_r(S \times Q) = E\,\mathrm{LogP}_r^{(S\times Q)} = E\,\widehat{\mathrm{LogP}_r^{(S)}} + E\,\widehat{\mathrm{LogP}_r^{(Q)}} = H_r(S) + H_r(Q). \tag{21}$$

[6]This is only a notational convention and *not* an extension of the concept of sum. That is, while writing $\sum_{t\in\mathbb{R}} \tau(t)$, we assume that the summation runs over those t, for which $\tau(t) \neq 0$.

Consequently,

$$H_r S^n \quad = \quad n \cdot H_r S. \tag{22}$$

1.4.2 Shannon's coding theorem

We are ready to state Shannon's coding theorem, sometimes also called Shannon's First Theorem.

Theorem 6 (Shannon's coding theorem)**.** *For any finite probabilistic space S and $r \geq 2$,*

$$\lim_{n\to\infty} \frac{L_r(S^n)}{n} = H_r(S).$$

Proof. We have from Theorem 5

$$H_r(S^n) \leq L_r(S^n) \leq H_r(S^n) + 1,$$

but since, by (22), $H_r(S^n) = n \cdot H_r(S)$, we have further

$$H_r(S) \leq \frac{L_r(S^n)}{n} \leq H_r(S) + \frac{1}{n},$$

which yields the claim. □

1.5 Conditional entropy

Entropy of random variable. We often consider a random variable (over a finite domain) that takes values in some abstract set $\mathcal{T}$, e.g., a set of words, rather than in real numbers.

We define the *entropy of a random variable* $X : S \to \mathcal{T}$, by

$$H_r(X) \quad = \quad \sum_{t\in\mathcal{T}} p(X = t) \cdot \log_r \frac{1}{p(X = t)} \tag{23}$$

Similarly, as in equation (10), we can present $H_r(X)$ as the expected value of a (real-valued) random variable $\mathrm{LogPX}_r : S \to \mathbb{R}$, defined by

$$\mathrm{LogPX}_r(s) \quad = \quad \begin{cases} \log_r \frac{1}{p(X=X(s))} & \text{if} \quad p(s) > 0 \\ 0 & \text{if} \quad p(s) = 0. \end{cases} \tag{24}$$

Note that $\mathrm{LogPX}_r(s)$ is well-defined because $p(X = X(s)) \geq p(s)$. We claim that

$$H_r(X) \quad = \quad E\,\mathrm{LogPX}_r. \tag{25}$$

Indeed,

$$\begin{aligned} \sum_{t\in\mathcal{T}} p(X = t) \cdot \log_r \frac{1}{p(X = t)} \quad &= \quad \sum_{t\in\mathcal{T}} \sum_{X(s)=t} p(s) \cdot \log_r \frac{1}{p(X = t)} \\ &= \quad \sum_{s\in S} p(s) \cdot \log_r \frac{1}{p(X = X(s))}, \end{aligned}$$

which yields (25) by the definition of LogPX_r and equation (14). (In case of $p(s) = 0$, we can replace $\log_r \frac{1}{p(X=t)}$ by 0 without affecting the result.)

Notational conventions. If the actual random variables are known from the context, we often abbreviate the event $X = a$ by just a; so we may write, e.g., $p(x|y)$ instead of $p(X = x|Y = y)$, $p(x \wedge y)$ instead of $p((X = x) \wedge (Y = y))$, etc. Please note that this should not be confused with the notation $p(s)$ used for a probability distribution over a probabilistic space. If, for instance, $X : S \to \mathcal{T}$ is a random variable over a probabilistic space S and $s \in S$ then $p(s)$ means probability of the event s, whereas $p(t)$ abbreviates $p(T = t) = \sum_{X(s)=t} p(s)$.

Conditional entropy. Let $A : S \to \mathcal{A}$, $B : S \to \mathcal{B}$, be two random variables. For $b \in \mathcal{B}$ with $p(b) > 0$, let

$$H_r(A|b) \quad = \quad \sum_{a \in \mathcal{A}} p(a|b) \cdot \log_r \frac{1}{p(a|b)}.$$

If $p(b) = 0$, we let, by convention, $H_r(A|b) = 0$. Now let

$$H_r(A|B) \quad = \quad \sum_{b \in \mathcal{B}} p(b) H_r(A|b). \tag{26}$$

Note that if A and B are independent then in the above formula $p(a|b) = a$, and hence $H_r(A|B) = H_r(A)$. On the other hand, $H_r(A|A) = 0$; more generally, if $\varphi : \mathcal{A} \to \mathcal{B}$ is any function then

$$H_r(\varphi(A)|A) \quad = \quad 0. \tag{27}$$

Indeed, if $p(A = a) > 0$ then $p(\varphi(A) = \varphi(a)|A = a) = 1$, and hence $\log_r \frac{1}{p(\varphi(A)=\varphi(a)|A=a)} = 0$.

We will see more properties of the the conditional entropy in the sequel.

Joint entropy. We also consider the couple (A, B) as a random variable $(A, B) : S \to \mathcal{A} \times \mathcal{B}$,

$$(A, B)(s) \quad = \quad (A(s), B(s)) \, .$$

Note that the probability that this variable takes value (a, b) is $p((A, B) = (a, b)) = p((A = a) \wedge (B = b))$, which we abbreviate by $p(a \wedge b)$. This probability is, in general, different from $p(a) \cdot p(b)$. In the case if, for all $a \in \mathcal{A}, b \in \mathcal{B}$,

$$p(a \wedge b) \quad = \quad p(a) \cdot p(b),$$

(i.e., the events $A = a$ and $B = b$ are independent), the variables A and B are called *independent*.

Now $H_r(A, B)$ is well defined by

$$H_r(A, B) \quad = \quad \sum_{a \in \mathcal{A}, b \in \mathcal{B}} p(a \wedge b) \cdot \log_r \frac{1}{p(a \wedge b)}.$$

We first observe that if A and B are independent then

$$H_r(A, B) \quad = \quad H_r(A) + H_r(B). \tag{28}$$

This can be inferred from the equality (for $p(a \wedge b) > 0$)

$$\log_r \frac{1}{p(a \wedge b)} \quad = \quad \log_r \frac{1}{p(a)} + \log_r \frac{1}{p(b)}, \tag{29}$$

either by a straightforward calculation, or by exploring the characterization (25). Indeed, (29) yields $\mathrm{LogP(A,B)}_r = \mathrm{LogPA}_r + \mathrm{LogPB}_r$ (if $p(s) = 0$, the equality holds trivially). Hence (29) follows by linearity of expectation (18).

In general case we have the following.

Theorem 7.

$$H_r(A,B) \quad \leq \quad H_r(A) + H_r(B). \tag{30}$$

Moreover, the equality holds if and only if A and B are independent.

Proof. We rewrite the right-hand side a bit, in order to apply Golden Lemma. We use the obvious equalities $p(a) = \sum_{b\in\mathcal{B}} p(a \wedge b)$, and $p(b) = \sum_{a\in\mathcal{A}} p(a\wedge b)$.

$$\begin{aligned} H_r(A) + H_r(B) &= \sum_{a\in\mathcal{A}} p(a)\log_r \frac{1}{p(a)} + \sum_{b\in\mathcal{B}} p(b)\log_r \frac{1}{p(b)} \\ &= \sum_{a\in\mathcal{A}}\sum_{b\in\mathcal{B}} p(a\wedge b)\log_r \frac{1}{p(a)} + \sum_{b\in\mathcal{B}}\sum_{a\in\mathcal{A}} p(a\wedge b)\log_r \frac{1}{p(b)} \\ &= \sum_{a\in\mathcal{A}, b\in\mathcal{B}} p(a\wedge b)\log_r \frac{1}{p(a)p(b)} \end{aligned}$$

Note that the last expression is well defined, because if $p(a) = 0$ or $p(b) = 0$ then $p(a \wedge b) = 0$, as well.
Let us momentarily denote

$$\mathcal{A}^+ = \{a \in \mathcal{A} : p(a) > 0\}, \;\; \mathcal{B}^+ = \{b \in \mathcal{B} : p(b) > 0\}.$$

Clearly, the above equation will not change if we restrict the sum to $\mathcal{A}^+ \times \mathcal{B}^+$, i.e.,

$$H_r(A) + H_r(B) \quad = \sum_{(a,b)\in\mathcal{A}^+\times\mathcal{B}^+} p(a\wedge b)\log_r \frac{1}{p(a)p(b)}.$$

On the other hand, $H_r(A,B)$ will be also unaffected if we restrict the sum in similar way, i.e.,

$$H_r(A,B) = \sum_{a\in\mathcal{A}, b\in\mathcal{B}} p(a\wedge b)\cdot\log_r \frac{1}{p(a\wedge b)} = \sum_{a\in\mathcal{A}^+, b\in\mathcal{B}^+} p(a\wedge b)\cdot\log_r \frac{1}{p(a\wedge b)}.$$

Therefore, by applying the Golden Lemma (Theorem 4) to $x = p(a \wedge b)$, $y = p(a)\cdot p(b)$, where (a, b) ranges over $\mathcal{A}^+ \times \mathcal{B}^+$, we obtain

$$\begin{aligned} H_r(A,B) &= \sum_{(a,b)\in\mathcal{A}^+\times\mathcal{B}^+} p(a\wedge b)\log_r \frac{1}{p(a\wedge b)} \\ &\leq \sum_{(a,b)\in\mathcal{A}^+\times\mathcal{B}^+} p(a\wedge b)\log_r \frac{1}{p(a)p(b)} \\ &= H_r(A) + H_r(B). \end{aligned}$$

Moreover, the equality holds only if $p(a \wedge b) = p(a) \cdot p(b)$, for all $(a, b) \in \mathcal{A}^+ \times \mathcal{B}^+$, and consequently, for all $a \in \mathcal{A}$, $b \in \mathcal{B}$, i.e., if A and B are independent. On the other hand, we have already seen that independence of A and B implies this equality. □

1.6 Mutual information

Definition 6 (information)**.** *The value*

$$I_r(A;B) \quad = \quad H_r(A) + H_r(B) - H_r(A,B). \tag{31}$$

is called mutual information *of variables A and B.*

Remark. The above concepts and properties have some interpretation in terms of *20 questions game* (page 5). Suppose an object to be identified is actually a couple (a, b), where a and b are values of random variables A and B, respectively. Now, if A and B are independent, we can do nothing better than identify a and b separately. Thus our series of questions splits into "questions about a" and "questions about b", which is reflected by the equality $H_r(A, B) = H_r(A) + H_r(B)$. However, if A and B are dependent, we can take advantage of mutual information and decrease the number of questions.

To increase readability, since now on we will omit subscript r, writing H, I, ..., instead of H_r, I_r, ... Unless stated otherwise, all our results apply to any $r > 1$. Without loss of generality, the reader may assume $r = 2$.

Remark From the transformations used in the proof of the theorem above, we easily deduce

$$I(A; B) \quad = \quad \sum_{a\in\mathcal{A},b\in\mathcal{B}} p(a \wedge b) \left(\log \frac{1}{p(a)p(b)} - \log \frac{1}{p(a \wedge b)} \right). \tag{32}$$

Hence $I(A; B)$ can be viewed as a measure of the distance between the actual distribution of the joint variable $(A; B)$ and its distribution if A and B were independent.

Note that the above sum is non-negative, although some summands $\left(\log \frac{1}{p(a)p(b)} - \log \frac{1}{p(a\wedge b)}\right)$ can be negative.

The following property generalizes the equality $H(A, B) = H(A) + H(B)$ to the case of arbitrary (possibly dependent) variables.

Fact 3 (Chain rule)**.**

$$H(A, B) \quad = \quad H(A|B) + H(B). \tag{33}$$

Proof. Let $\mathcal{B}^+ = \{b : p(b) > 0\}$. We calculate:

$$\begin{aligned}
H(A, B) &= \sum_{a\in\mathcal{A},b\in\mathcal{B}} p(a \wedge b) \cdot \log \frac{1}{p(a \wedge b)} \\
&= \sum_{a\in\mathcal{A}} \sum_{b\in\mathcal{B}^+} p(a|b)p(b) \cdot \log \frac{1}{p(a|b)p(b)} \\
&= \sum_{a\in\mathcal{A}} \sum_{b\in\mathcal{B}^+} p(a|b)p(b) \cdot \left(\log \frac{1}{p(a|b)} + \log \frac{1}{p(b)} \right) \\
&= \sum_{b\in\mathcal{B}^+} p(b) \cdot \sum_{a\in\mathcal{A}} p(a|b) \cdot \log \frac{1}{p(a|b)} + \sum_{b\in\mathcal{B}^+} p(b) \log \frac{1}{p(b)} \cdot \underbrace{\sum_{a\in\mathcal{A}} p(a|b)}_{1} \\
&= H(A|B) + H(B).
\end{aligned}$$

□

we immediately get the following.

Corollary 1. *For random variables A and B,*

$$H(A|B) \quad \leq \quad H(A). \tag{34}$$

Moreover, the equality holds iff A and B are independent.

The above can be interpreted that the entropy of A may only decrease if we additionally know B. Note however, that this inequality holds *on average*, and may not be true for a particular value of B.

Exercise Show an example where $H(A|b) > H(A)$.

Applying the chain rule, we get alternative formulas for information:

$$\begin{aligned} I(A;B) &= H(A) - H(A|B) && (35)\\ &= H(B) - H(B|A). && (36)\end{aligned}$$

This also implies that $I(A;B) \leq \min\{H(A), H(B)\}$.

The Chain rule generalizes easily to the case of $n \geq 2$ variables $A_1, A_2, \ldots, A_n$.

$$\begin{aligned} H(A_1,\ldots,A_n) &= H(A_1|A_2,\ldots,A_n) + H(A_2,\ldots,A_n)\\ &= H(A_1|A_2,\ldots,A_n) + H(A_2|A_3,\ldots,A_n) + H(A_3,\ldots,A_n)\\ &= \sum_{i=1}^{n} H(A_i|A_{i+1},\ldots,A_n) && (37)\end{aligned}$$

if we adopt convention $H(A|\emptyset) = H(A)$. From equation (37), we obtain a generalization of Theorem 7.

Corollary 2.

$$H(A_1,\ldots,A_n) \leq H(A_1) + \ldots + H(A_n) \qquad (38)$$

and the equality holds if and only if $A_1,\ldots,A_n$ are independent.

Proof. Indeed, we have $H(A_i|A_{i+1},\ldots,A_n) \leq H(A_i)$, for $i = 1,\ldots,n$, and the equality holds whenever A_i is independent from $(A_{i+1},\ldots,A_n)$ (for $i < n$). Hence, the equality holds in (38) iff each A_i is independent from $(A_{i+1},\ldots,A_n)$, for $i = 1,\ldots,n-1$, i.e., all A_i's are independent. □

A more subtle generalization follows from relativization.

Fact 4 (Conditional chain rule)**.**

$$H(A,B|C) = H(A|B,C) + H(B|C). \qquad (39)$$

Proof. We use the fact that, whenever $p(a \wedge b|c) > 0$,

$$p(a \wedge b|c) = \frac{p(a \wedge b \wedge c)}{p(c)} = \frac{p(a \wedge b \wedge c)}{p(b \wedge c)} \cdot \frac{p(b \wedge c)}{p(c)} = p(a|b \wedge c) \cdot p(b|c).$$

Then we have

$$\begin{aligned} H(A,B|c) &= \sum_{a\in\mathcal{A},b\in\mathcal{B}} p(a \wedge b|c) \cdot \log \frac{1}{p(a \wedge b|c)}\\ &= \sum_{a,b} p(a|b \wedge c) \cdot p(b|c) \cdot \left(\log \frac{1}{p(a|b \wedge c)} + \log \frac{1}{p(b|c)}\right)\\ &= \sum_b p(b|c) \cdot \sum_a p(a|b \wedge c) \cdot \log \frac{1}{p(a|b \wedge c)} + \sum_b p(b|c) \cdot \log \frac{1}{p(b|c)} \cdot \underbrace{\sum_a p(a|b \wedge c)}_{1}.\end{aligned}$$

To make sure that the respective conditional probabilities are defined, we may assume that b ranges over those values for which $p(b \wedge c) > 0$. (The equations still hold, since the other summands disappear.)

By taking the average over $p(c)$, we further have

$$\begin{aligned} H(A,B|C) &= \sum_{c\in\mathcal{C}} p(c)\cdot H(A,B|c) \\ &= \sum_c p(c)\cdot\sum_b p(b|c)\cdot\sum_a p(a|b\wedge c)\cdot\log\frac{1}{p(a|b\wedge c)} + \sum_c p(c)\cdot\sum_b p(b|c)\cdot\log\frac{1}{p(b|c)} \\ &= \underbrace{\sum_{b,c} p(b\wedge c)\cdot\sum_a p(a|b\wedge c)\cdot\log\frac{1}{p(a|b\wedge c)}}_{H(A|B,C)} + \underbrace{\sum_c p(c)\cdot\sum_b p(b|c)\cdot\log\frac{1}{p(b|c)}}_{H(B|C)}, \end{aligned}$$

as required. □

We leave to the reader to show that

$$H(A,B|C) \quad\leq\quad H(A|C)+H(B|C) \tag{40}$$

and the equality holds if and only if A and B are *conditionally independent given* C, i.e.,

$$p(A=a\wedge B=b|C=c) \quad=\quad p(A=a|C=c)\cdot p(B=b|C=c).$$

The proof can go along the same lines as the proof of Theorem 7.

Conditional information We let the *mutual information of* A *and* B *given* C be defined by

$$\begin{aligned} I(A;B|C) &= H(A|C)+H(B|C) - \underbrace{H(A,B|C)}_{H(A|B,C)+H(B|C)} & (41) \\ &= H(A|C) - H(A|B,C). & (42) \end{aligned}$$

Finally, let *mutual information of* A*,* B*, and* C be defined by

$$R(A;B;C) \quad=\quad I(A;B) - I(A;B|C). \tag{43}$$

Let us see that this definition is indeed symmetric, i.e., does not depend on the particular ordering of A,B,C:

$$\begin{aligned} I(A;C)-I(A;C|B) &= H(A)-H(A|C)-(H(A|B)-H(A|B,C)) \\ &= \underbrace{H(A)-H(A|B)}_{I(A;B)} - \underbrace{(H(A|C)-H(A|B,C))}_{I(A;B|C)}. \end{aligned}$$

Note however, that in contrast to $I(A;B)$ and $I(A;B|C)$, $R(A;B;C)$ can be *negative.*

Example Let A and B be independent random variables with values in $\{0,1\}$, and let

$$C = A\oplus B.$$

Then $I(A;B)=0$, while

$$I(A;B|C) = H(A|C) - \underbrace{H(A|B,C)}_{0}$$

and we can easily make sure that $H(A|C)>0$ (cf. the example on page 23 below).

The set of equations relating the quantities $H(X), H(Y), H(Z)$, $H(X,Y), H(X,Y|Z)$, $I(X;Y), I(X;Y|Z)$, $R((X;Y;Z)$, $\ldots$, can be pictorially represented by the so-called *Venn diagram.* (See, e.g., Wikipedia https://en.wikipedia.org/wiki/Information_diagram; note however that this is only a helpful representation without *extra* meaning.)

1.7 Application: Perfect secrecy

A *cryptosystem* is a triple of random variables:

- M with values in a finite set $\mathcal{M}$ (messages),
- K with values in a finite set $\mathcal{K}$ (keys),
- C with values in a finite set $\mathcal{C}$ (cipher-texts).

Moreover, there must be a function $Dec : \mathcal{C} \times \mathcal{K} \to \mathcal{M}$, such that

$$M = Dec(C, K)$$

(unique decodability).

Note that we do not require that C be a function of M and K, since the encoding need not, in general, be functional. It can, for example, use random bits (like in the Elgamal cryptosystem, see, e.g., [3]).

A cryptosystem is *perfectly secret* if $I(C; M) = 0$.

Example: One time pad Here $\mathcal{M} = \mathcal{K} = \mathcal{C} = \{0,1\}^n$, for some $n \in \mathbb{N}$, and

$$C = M \oplus K$$

where $\oplus$ is the component-wise *xor* (e.g., $101101 \oplus 110110 = 011011$). Hence $Dec(v, w) = v \oplus w$, as well. Moreover we assume that K has uniform distribution over $\{0,1\}^n$, i.e., $p(K = v) = \frac{1}{2^n}$, for $v \in \{0,1\}^n$, and that K and M are independent.

In order to show perfect secrecy, it is enough to prove that M and C are independent (see Theorem 7 and Definition 6), and to this end, it is enough to show

$$p(C = w | M = u) = p(C = w). \tag{44}$$

We have

$$\begin{aligned} p(C = w) &= \sum_{u \oplus v = w} p(M = u \wedge K = v) \\ &= \sum_{u} p(M = u) \cdot \frac{1}{2^n} \\ &= \frac{1}{2^n}. \end{aligned}$$

On the other hand, we have

$$\begin{aligned} p(C = w | M = u) &= \frac{p(C = w \wedge M = u)}{p(M = u)} && (45) \\ &= \frac{p(K = u \oplus w \wedge M = u)}{p(M = u)} && (46) \\ &= \frac{p(K = u \oplus w) \cdot p(M = u)}{p(M = u)} && (47) \\ &= \frac{1}{2^n}. && (48) \end{aligned}$$

To infer (46) from (45), we used the fact that, in One time pad, the values of M and C determine the value of K; hence we have the equivalences

$$C = w \wedge M = u \iff K = u \oplus w \wedge C = w \wedge M = u \iff K = u \oplus w \wedge M = u.$$

This proves (44).

Exercise. Show that the independence of M and K is really necessary to achieve perfect secrecy of One time pad.

Remark. Note that the distribution of M is not, in general, uniform, and consequently C and K may be dependent. In the example below (with $n = 1$), we have $p(K = 1|C = 0) = p(K = 0|C = 1) = \frac{2}{3}$, and therefore it is more likely that $K = 1 - i$ rather than i, whenever $C = i$.

0	0	1	1	1	1	M
0	1	0	0	1	1	K
0	1	1	1	0	0	C=M+K

This is precisely why the system must be "one time". After one use of a key, the adversary gets already some information about it. Consequently, if a key is reused, the cryptologist has a non-zero chance to decode the message. This indeed happened in history; see the literature[7] on the American *VENONA* project (1943–1980), which succeeded in reconstructing a number of KGB messages, because Soviets reused some portion of keys in the One time pad.

Theorem 8 (Shannon's Pessimistic Theorem)**.** *Any perfectly secret cryptosystem satisfies*

$$H(K) \quad \geq \quad H(M).$$

Consequently (c.f. Theorem 5)

$$L_r(K) \geq H_r(K) \geq H_r(M) \geq L_r(M) - 1.$$

Roughly speaking, to guarantee perfect secrecy, the keys must be (almost) as long as messages, which is highly impractical.

Proof. We have

$$H(M) \quad = \quad H(M|C,K) + \underbrace{I(M;C)}_{H(M)-H(M|C)} \quad + \quad \underbrace{I(M;K|C)}_{H(M|C)-H(M|K,C)} \quad .$$

But $H(M|C;K) = 0$, since $M = Dec(C,K)$ is a function of (C,K), and $I(M;C) = 0$, by assumption, hence

$$H(M) \quad = \quad I(M;K|C).$$

By symmetry, we have

$$H(K) \quad = \quad H(K|M,C) + I(K;C) + \underbrace{I(K;M|C)}_{H(M)},$$

which gives the desired inequality. $\square$

As another application of the quantitative concept of information, we observe a property which at first sight may appear a bit surprising. Let A and B be random variables; we may think that A represents some experimental data, and B our knowledge about them. Can we increase the information about A by processing B (say, by analysis, computation, etc.)? It turns out that we cannot.

Lemma 2. *Suppose A and C are conditionally independent, given B (see page 21). Then*

$$I(A;C) \quad \leq \quad I(A;B).$$

[7] E.g., the very informative Dirk Rijmenants' website http://users.telenet.be/d.rijmenants/en/onetimepad.htm.

Proof. First note the following *chain rule for information*:

$$\underbrace{I(A;(B,C))}_{H(A)-H(A|B,C)} \quad = \quad \underbrace{I(A;C)}_{H(A)-H(A|C)} \quad + \quad \underbrace{I(A;B|C)}_{H(A|C)-H(A|B,C)} \quad .$$

By symmetry, and from the conditional independence of A and C

$$I(A;(B,C)) \quad = \quad I(A;B) + \underbrace{I(A;C|B)}_{0},$$

which yields the desired inequality. □

Note that the equality holds iff, additionally, A and B are conditionally independent given C.

Corollary 3. *If f is a function then*

$$I(A;f(B)) \quad \leq \quad I(A;B). \tag{49}$$

Proof. Follows from the Lemma, since

$$I(A;f(B)|B) = \underbrace{H(f(B)|B)}_{0} - \underbrace{H(f(B)|A,B)}_{0} = 0.$$

□

2 Information channels

2.1 Channels

Definition 7 (channel)**.** *A* communication channel Γ *is given by*

- *a finite set $\mathcal{A}$ of* input *objects,*
- *a finite set $\mathcal{B}$ of* output *objects,*
- *a mapping $\mathcal{A} \times \mathcal{B} \to [0,1]$, sending (a,b) to $P(a \to b)$, such that, for all $a \in \mathcal{A}$,*

$$\sum_{b \in \mathcal{B}} P(a \to b) \quad = \quad 1.$$

Random variables A and B with values in $\mathcal{A}$ and $\mathcal{B}$, respectively, form an input-output *pair for the channel Γ if, for all $a \in \mathcal{A}, b \in \mathcal{B}$,*

$$p(B=b|A=a) \quad = \quad P(a \to b).$$

We visualize it by

$$A \to \boxed{\Gamma} \to B.$$

Note that if A and B form an *input-output* pair then

$$p(A=a \wedge B=b) \quad = \quad P(a \to b) \cdot p(A=a).$$

Hence, the distribution of (A, B) forming an input-output pair is uniquely determined by A (for fixed Γ). In particular, a suitable B exists and its distribution is determined by

$$p(B = b) \quad = \quad \sum_{a \in \mathcal{A}} P(a \to b) \cdot p(A = a). \tag{50}$$

Knowing this, the reader may easily calculate $H(A, B)$, $H(B|A)$, $I(A; B)$, etc. (depending on Γ and A).

We define the *capacity* of the channel Γ by

$$C_\Gamma \quad = \quad \max_A I(A; B), \tag{51}$$

where, for concreteness, $I = I_2$. Here A ranges over all random variables with values in $\mathcal{A}$, and (A, B) forms an input-output pair for Γ. The maximum exists because $I(A; B)$ is a continuous mapping from the compact set $\{p \in [0, 1]^{\mathcal{A}} : \sum_{a \in \mathcal{A}} p(a) = 1\}$ to $\mathbb{R}$, which moreover is bounded since $I(A; B) \leq H(A) \leq \log |\mathcal{A}|$.

If $\mathcal{A} = \{a_1, \ldots, a_m\}$, $\mathcal{B} = \{b_1, \ldots, b_n\}$, then the channel can be represented by a matrix

$$\begin{pmatrix} P_{11} & \ldots & P_{1n} \\ \ldots & \ldots & \ldots \\ P_{m1} & \ldots & P_{mn}, \end{pmatrix}$$

where $P_{ij} = P(a_i \to b_j)$.

The formula for distribution of B in matrix notation is

$$(p(a_1), \ldots, p(a_m)) \cdot \begin{pmatrix} P_{11} & \ldots & P_{1n} \\ \ldots & \ldots & \ldots \\ P_{m1} & \ldots & P_{mn}, \end{pmatrix} \quad = \quad (p(b_1), \ldots, p(b_n)). \tag{52}$$

Examples

We can present a channel as a bipartite graph from $\mathcal{A}$ to $\mathcal{B}$, with an arrow $a \to b$ labeled by $P(a \to b)$ (if $P(a \to b) = 0$, the arrow is not represented).

Faithful (noiseless) channel Let $\mathcal{A} = \mathcal{B} = \{0, 1\}$.

$$\begin{tikzcd} 0 \arrow[r] & 0 \\ 1 \arrow[r] & 1 \end{tikzcd}$$

The matrix representation of this channel is

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

Since A is always a function of B, we have $I(A; B) = H(A)$, and hence the capacity is

$$C_\Gamma = \max_A I(A; B) = \max_A H(A) = \log_2 |\mathcal{A}| = 1.$$

Inverse faithful channel

$$\begin{tikzcd} 0 \arrow[dr] & 0 \\ 1 \arrow[ur] & 1 \end{tikzcd}$$

matrix representation: $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$, capacity: $C_\Gamma = 1$.

Noisy channel without overlap Here $\mathcal{A} = \{0,1\}$, $\mathcal{B} = \{0,1,2,3\}$.

$$\begin{tikzcd}
 & 0 \\
0 \arrow[ur, "\frac{1}{2}"] \arrow[r, "\frac{1}{2}"'] & 1 \\
1 \arrow[r, "\frac{1}{3}"] \arrow[dr, "\frac{2}{3}"'] & 2 \\
 & 3
\end{tikzcd}$$

The matrix representation is

$$\begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & \frac{1}{3} & \frac{2}{3} \end{pmatrix}$$

Here again A is a function of B, hence $I(A;B) = H(A) - H(A|B) = H(A)$, and therefore $C_\Gamma = 1$.

Noisy typewriter Here[8] we assume $\mathcal{A} = \mathcal{B} = \{a, b, \dots, z\}$ (26 letters, say), and

$$p(\alpha \to \alpha) = p(\alpha \to \mathit{next}(\alpha)) = \frac{1}{2}$$

where $\mathit{next}(a) = b$, $\mathit{next}(b) = c$, $\dots$, $\mathit{next}(y) = z$, $\mathit{next}(z) = a$.

We leave to the reader to draw graphical and matrix representation.

To compute the capacity, first observe that, for any α,

$$H(B|\alpha) = p(\alpha|\alpha) \cdot \log \frac{1}{p(\alpha|\alpha)} + p(\mathit{next}(\alpha)|\alpha) \cdot \log \frac{1}{p(\mathit{next}(\alpha)|\alpha)} = (\frac{1}{2} + \frac{1}{2}) \cdot \log_2 2 = 1.$$

Hence

$$C_\Gamma = \max_A I(A;B) = \max_A H(B) - \underbrace{H(B|A)}_{1} = \log 26 - 1 = \log 13$$

(the maximum is achieved for A with uniform distribution).

The reader may have already grasped that capacity is a desired value, like information, and unlike entropy. What are the channels with the minimal possible capacity, i.e., $C_\Gamma = 0$?

Bad channels Clearly $C_\Gamma = 0$ whenever $I(A;B) = 0$ for all input-output pairs, i.e., all such pairs are independent. This requires that $p(B = b|A = a) = p(B = b)$, for all $a \in \mathcal{A}$, $b \in \mathcal{B}$ (unless $p(A = a) = 0$), hence for a fixed b, all values $p(B = b|A = a)$ (i.e., all values in a column in the matrix representation) must be equal.

For example, the following channels have this property:

$$\begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix} \qquad \begin{pmatrix} \frac{1}{2} & 0 & \frac{1}{6} & \frac{1}{3} \\ \frac{1}{2} & 0 & \frac{1}{6} & \frac{1}{3} \end{pmatrix} \qquad \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

[8] *Typewriter* had been a mechanical device for writing used between clay tablets and computers, see, e.g., Wikipiedia https://en.wikipedia.org/wiki/Typewriter.

The last example is a particularly dull channel, which always outputs the same value. Note that in this case $H(B)$ is always 0, which means that the entropy may sometimes decrease while sending a message through a channel. However, in most interesting cases it actually increases.

The following example is most important in our further studies.

2.2 Binary symmetric channel (BSC)

Here again $\mathcal{A} = \mathcal{B} = \{0, 1\}$.

$$\begin{tikzcd} 0 \arrow[r, "P"] \arrow[dr, "1-P"] & 0 \\ 1 \arrow[ur, "1-P"'] \arrow[r, "P"'] & 1 \end{tikzcd}$$

Letting $\bar{P} = 1 - P$, the matrix representation is

$$\begin{pmatrix} P & \bar{P} \\ \bar{P} & P \end{pmatrix}$$

Prior to calculating C_Γ, we note the important property.

Fact 5. *If (A, B) forms an input-output pair for a BSC then*

$$H(B) \quad \geq \quad H(A).$$

Moreover, the equality holds only if $P \in \{0, 1\}$ (i.e., the channel is faithful or inverse-faithful), or if $H(A) = 1$ (i.e., the entropy of A achieves the maximal value).

Proof. Let $q = p(A = 0)$. Then $p(A = 1) = \bar{q}$, and we calculate the distribution of B by the formula

$$(q, \bar{q}) \cdot \begin{pmatrix} P & \bar{P} \\ \bar{P} & P \end{pmatrix} = (\underbrace{qP + \bar{q}\bar{P}}_{p(B=0)}, \underbrace{q\bar{P} + \bar{q}P}_{p(B=1)})$$

Let $r = p(B = 0)$. Then

$$\begin{aligned} H(A) &= -q \log q - \bar{q} \log \bar{q} \\ H(B) &= -r \log r - \bar{r} \log \bar{r} \end{aligned}$$

Recall our convention (5) that $0 \log_r 0 = 0 \log_r \frac{1}{0} = 0$, and let h denote the mapping

$$h(x) \quad = \quad x \ln x + (1 - x) \ln(1 - x),$$

defined for $0 \leq x \leq 1$. We easily calculate (for $0 < x < 1$)

$$\begin{aligned} h'(x) &= 1 + \ln x - 1 - \ln(1 - x) \\ h''(x) &= \frac{1}{x} + \frac{1}{1 - x} > 0. \end{aligned}$$

Hence by Lemma 1 (page 8), the function $h(x)$ is strictly convex on $[0, 1]$, and it readily implies that so is the function

$$\log_2 e \cdot h(x) \quad = \quad x \log_2 x + (1 - x) \log_2(1 - x).$$

Taking in the definition of convexity (3) $x_1 = q$, $x_2 = \bar{q}$, and $\lambda = P$ (hence $\lambda x_1 + (1-\lambda)x_2 = r$), and noting that $h(q) = h(\bar{q})$, we obtain that

$$\begin{aligned} q \log q + \bar{q} \log \bar{q} &\geq r \log r + \bar{r} \log \bar{r} \\ \text{i.e., } H(A) &\leq H(B) \end{aligned}$$

and, moreover, the equality holds only if $P \in \{0,1\}$ or if $q = \bar{q}$, which holds iff $H(A) = \log_2 |\{0,1\}| = 1$. $\square$

We are going to calculate C_Γ. It is convenient to use notation

$$H(s) = -s \log_2 s - (1-s) \log_2 (1-s) \tag{53}$$

(justified by the fact that $H(s) = H(X)$, whenever $p(X=0) = s$, $p(X=1) = \bar{s}$). Note that $H(0) = H(1) = 0$, and the maximum of H in $[0,1]$ is $H(\frac{1}{2}) = 1$.

By the definition of conditional entropy, we have

$$\begin{aligned} H(B|A) &= p(A=0) \cdot \left(p(B=0|A=0) \cdot \log \frac{1}{p(B=0|A=0)} + p(B=1|A=0) \cdot \log \frac{1}{p(B=1|A=0)} \right) \\ &\quad + p(A=1) \cdot \left(p(B=0|A=1) \cdot \log \frac{1}{p(B=0|A=1)} + p(B=1|A=1) \cdot \log \frac{1}{p(B=1|A=1)} \right) \\ &= p(A=0) \cdot \left(P \cdot \log \frac{1}{P} + \bar{P} \cdot \log \frac{1}{\bar{P}} \right) + p(A=1) \cdot \left(\bar{P} \cdot \log \frac{1}{\bar{P}} + P \cdot \log \frac{1}{P} \right) \\ &= P \cdot \log \frac{1}{P} + \bar{P} \cdot \log \frac{1}{\bar{P}} \\ &= H(P). \end{aligned}$$

Hence, $H(B|A)$ does not depend on A.

Now, by the calculation of the distribution of B above, we have

$$H(B) = H(qP + \bar{q}\bar{P})$$

which achieves the maximal value $1 = H(\frac{1}{2})$, for $q = \frac{1}{2}$. Hence

$$C_\Gamma = \max_A H(B) - H(B|A) = 1 - H(P). \tag{54}$$

2.3 Decision rules

Suppose we receive a sequence of letters $b_{i_1}, \ldots, b_{i_k}$, transmitted through a channel Γ. Knowing the matrix $(P(a \to b)_{a \in \mathcal{A}, b \in \mathcal{B}})$, can we decode the message?

In some cases the answer is simple. For example, in the inverse faithful channel (page 25), we should just interchange 0 and 1. However, for the noisy typewriter (page 26), no "sure" decoding exists. For instance, an output word *afu*, can result from input *zet*, but also from *aft*, and many others[9] (but not, e.g., from input *abc*).

In general, the objective of the receiver is, given an output letter b, to guess (or "decide") what input symbol a has been sent. This is captured by the concept of a *decision rule*, which can be any mapping $\Delta : \mathcal{B} \to \mathcal{A}$. Clearly the receiver wants to maximize $p(A = \Delta(b)|B = b)$.

The quality of the rule is measured by

$$Pr_C(\Delta, A) =_{def} p(\Delta \circ B = A), \tag{55}$$

[9]The reader is encouraged to find some "meaningful" examples.

where (A, B) forms an input–output pair[10]. We have from definition

$$\begin{aligned} p(\Delta \circ B = A) &= \sum_{a\in\mathcal{A}, b\in\mathcal{B}} p(A = a \wedge B = b \wedge \Delta(b) = a) \\ &= \sum_{b\in\mathcal{B}} p(B = b \wedge A = \Delta(b)). \end{aligned}$$

The last term can be decomposed in two ways using conditional probabilities, whenever defined.

$$p(B = b \wedge A = \Delta(b)) = p(A = \Delta(b)) \cdot \underbrace{p(B = b|A = \Delta(b))}_{P(\Delta(b)\to b)} = p(B = b) \cdot p(A = \Delta(b)|B = b).$$

Similarly, we have

$$\sum_{b:\Delta(b)=a} p(A = a \wedge B = b) = p(A = a \wedge \Delta(B) = a) = p(A = a) \cdot p(\Delta(B) = a|A = a)$$

whenever $p(A) > 0$. This gives us several formulas to compute $Pr_C(\Delta, A)$

$$\begin{aligned} Pr_C(\Delta, A) &= \sum_{b\in\mathcal{B}} p(B = b) \cdot p(A = \Delta(b)|B = b) && (56) \\ &= \sum_{b\in\mathcal{B}} p(A = \Delta(b)) \cdot P(\Delta(b) \to b) && (57) \\ &= \sum_{a\in\mathcal{A}} p(A = a) \cdot p(\Delta(B) = a|A = a), && (58) \end{aligned}$$

all of them useful.

Dually, the *error probability* of the rule Δ is

$$\begin{aligned} Pr_E(\Delta, A) &= 1 - Pr_C(\Delta, A) \\ &= \sum_{a\in\mathcal{A}, b\in\mathcal{B}} p(A = a \wedge B = b \wedge \Delta(b) \neq a). \end{aligned}$$

We can compute it, e.g., by

$$Pr_E(\Delta, A) = \sum_{a\in\mathcal{A}} p(A = a) \cdot p(\Delta \circ B \neq a|A = a) \qquad (59)$$

We refer the reader to https://drive.google.com/file/d/0B3HSaaQuarpMWXFzc194ZTFsTHc/view, Problem 2, for a concrete example.

We are generally interested in rules maximizing $Pr_C(\Delta, A)$, and thus minimizing $Pr_E(\Delta, A)$.

If the distribution of A is known, the above objective is realized by the following.

Ideal observer rule This rule sends $b \in \mathcal{B}$ to $\Delta_o(b) = a$, such that $p(a|b)$ is maximal, where $p(a|b)$ can be calculated (knowing A) by

$$p(a|b) = \frac{p(a \wedge b)}{p(b)} = \frac{P(a \to b) \cdot p(a)}{\sum_{a'\in\mathcal{A}} P(a' \to b) \cdot p(a')}.$$

[10] In this case, the distribution of B is determined by the distribution of A by the equation (50), hence the definition is correct.

(Formally, this definition requires that $p(B=b)>0$; but if $p(B=b)=0$, we can define $\Delta(b)$ arbitrarily, and it will not affect (55).) Clearly, this choice maximizes the right-hand side of (56). Hence, we have

$$Pr_C(\Delta_o, A) \quad \geq \quad Pr_C(\Delta, A),$$

for any rule Δ. Note however, that $Pr_C(\Delta_o, A)$ is in general smaller than 1 (hence $Pr_E(\Delta_o, A) > 0$). In particular if, for some b, the maximal value of $p(a|b)$ is achieved with two different a's then the Ideal observer rule is weak.

Exercise Calculate $Pr_C(\Delta_o, A)$ for the "bad" channels on page 26.

A disadvantage of the ideal observer rule is that it requires some *a priori* knowledge about the message to be sent. If the distribution of A is unknown, a reasonable choice is the following.

Maximal likelihood rule This rule sends $b \in \mathcal{B}$ to $\Delta_{\max}(b) = a$, such that $P(a \to b) = p(b|a)$ is maximal. If A has uniform distribution (i.e., $p(a) = \frac{1}{|\mathcal{A}|}$) then this rule acts as Δ_o, i.e.[11],

$$Pr_C(\Delta_{\max}, A) \quad = \quad Pr_C(\Delta_o, A).$$

Indeed, maximizing $p(a|b)$ given b amounts to maximizing $p(a \wedge b) = p(a|b) \cdot p(b)$, which in the uniform case is $p(a \wedge b) = P(a \to b) \cdot \frac{1}{|\mathcal{A}|}$.

If A is not uniform, the maximal likelihood rule need not be optimal (the reader may easily find an example). However, it is in some sense *globally optimal*. We only sketch the argument informally.

Let $\mathcal{A} = \{a_1, \ldots, a_m\}$, and let $\mathcal{P}$ be the set of all possible probability distributions over $\mathcal{A}$,

$$\mathcal{P} \quad = \quad \{\mathbf{p} : \sum_{a \in \mathcal{A}} \mathbf{p}(a) = 1\}.$$

We identify a random variable A taking values in $\mathcal{A}$ with its probability distribution $\mathbf{p}$ in $\mathcal{P}$; hence $\mathbf{p}(a) = p(A=a)$. Now, using (57), the global value of a rule Δ can be calculated by

$$\begin{aligned}\int_{\mathbf{p} \in \mathcal{P}} Pr_C(\Delta, \mathbf{p})\, d\mathbf{p} \quad &= \quad \int_{\mathbf{p} \in \mathcal{P}} \sum_{b \in \mathcal{B}} \mathbf{p}(\Delta(b)) \cdot P(\Delta(b) \to b)\, d\mathbf{p} \\ &= \quad \sum_{b \in \mathcal{B}} P(\Delta(b) \to b) \cdot \int_{\mathbf{p} \in \mathcal{P}} \mathbf{p}(\Delta(b))\, d\mathbf{p}\end{aligned}$$

But it should be intuitively clear that the value of $\int_{\mathbf{p} \in \mathcal{P}} \mathbf{p}(a)\, d\mathbf{p}$ does not depend on a particular choice of $a \in \mathcal{A}$. (A formal argument refers to the concept of Lebesgues integral. Note however that $\mathbf{p}(a)$ is just a projection of $\mathbf{p}$ on one of its components, and no component is *a priori* privileged.) Thus $\int_{\mathbf{p} \in \mathcal{P}} \mathbf{p}(\Delta(b))\, d\mathbf{p}$ is always the same. Hence, maximization of $\int_{\mathbf{p} \in \mathcal{P}} Pr_C(\Delta, \mathbf{p})\, d\mathbf{p}$ amounts to maximization of $\sum_{b \in \mathcal{B}} P(\Delta(b) \to b)$, and this is achieved with the maximal likelihood rule.

2.4 Multiple use of channel

Recall (Definition 7) that if A and B form an input-output pair for a channel Γ then $p(b|a) = P(a \to b)$. Now suppose that we subsequently send symbols $a_1, a_2, \ldots, a_k$; what is the probability that the output is $b_1, b_2, \ldots, b_k$? One may expect that this is just the product of the $P(a \to b)$'s, we shall see that this is indeed the case if the transmissions are independent.

[11]We have $\Delta_{\max} = \Delta_o$, assuming that both rules make the same choice if there are more a's with the same maximal $P(a \to b)$.

Recall that random variables $X_1, \ldots, X_k$ are independent[12] if

$$p(X_1 = x_1 \wedge \ldots \wedge X_k = x_k) \quad = \quad p(X_1 = x_1) \cdot \ldots \cdot p(X_k = x_k)$$

Extending our notational convention (see page 17), we often abbreviate $p(X_1 = x_1 \wedge \ldots \wedge X_k = x_k)$ by $p(x_1 \ldots x_k)$, etc.

Lemma 3. *If the random variables (A, B) and (A', B') are independent then*

$$p(B = b \wedge B' = b' | A = a \wedge A' = a') \quad = \quad p(B = b | A = a) \cdot p(B' = b' | A' = a'),$$

whenever $p(A = a) > 0$ and $p(A' = a') > 0$.

Proof. Observe first that independence of (A, B) and (A', B') implies independence of A and A'; indeed we have

$$p(a \wedge a') = p\left((a \wedge \bigvee \mathcal{B}) \wedge (a' \wedge \bigvee \mathcal{B}')\right) = \sum_{b,b'} p(a \wedge b) \cdot p(a' \wedge b') = p(a) \cdot p(a').$$

Hence

$$p(b \wedge b' | a \wedge a') = \frac{p(b \wedge b' \wedge a \wedge a')}{p(a \wedge a')} = \frac{p(b \wedge a) \cdot p(b' \wedge a')}{p(a) \cdot p(a')} = p(b|a) \cdot p(b'|a').$$

□

Definition 8 (independence of symbols)**.** *A sequence $(A_1, B_1), \ldots, (A_k, B_k)$ of input-output pairs for a channel Γ has the* independent symbols *property if*

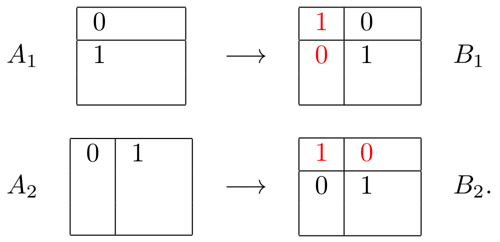
$$p(b_1 \ldots b_k | a_1 \ldots a_k) \quad = \quad p(b_1|a_1) \cdot \ldots \cdot p(b_k|a_k) \tag{60}$$

whenever $p(a_1 \ldots a_k) > 0$.

Corollary 4. *Suppose that $(A_1, B_1), \ldots, (A_k, B_k)$, are independent random variables, such that each (A_i, B_i) forms an input-output pair for a channel Γ. Then the sequence $(A_1, B_1), \ldots, (A_k, B_k)$ has the independent symbols property.*

Proof. Clearly the independence of $(A_1, B_1), \ldots, (A_k, B_k)$ implies that (A_1, B_1) is independent from the random variable $(A_2, \ldots, A_k, B_2, \ldots, B_k)$. Hence, we prove the desired equality by repeated application of the Lemma. □

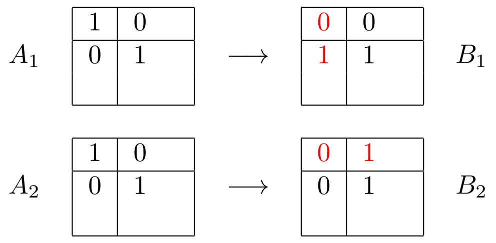
Remark. The mere assumption that the input variables $A_1, \ldots, A_k$, are independent is not enough to satisfy the independence of symbols, as the following example shows. The channel matrix is $\begin{pmatrix} 2/3 & 1/3 \\ 1/3 & 2/3 \end{pmatrix}$.



Here A_1 and A_2 have both distribution $A_i(0) = \frac{1}{3}$, $A_i(1) = \frac{1}{3}$ and are independent, whereas B_1 and B_2 are identical. The red color indicates the places where the transmission error have occurred. We have, in particular, $p(11|00) = p(00|01) = p(00|10) = p(11|11) = 1$, so the independence property clearly fails.

[12] The reader should note that this assumption is stronger than pairwise independence. For example, one can easily construct random variables $X_1, \ldots, X_k$ $(k > 2)$, with values in $\{0, 1\}$, such that each two (even each $k - 1$) of them are independent, but $X_k = \bigoplus_{i=1}^{k-1} X_i$.

To see that the independence of the output variables $B_1, \dots, B_k$, is not sufficient either, it is enough to reverse the above example.



The matrix of the channel is now $\begin{pmatrix} 1/2 & 1/2 \\ 1/5 & 4/5 \end{pmatrix}$.

We have, for example, $p(B_1 = 0 \wedge B_2 = 0 | A_1 = 1 \wedge A_2 = 1) = \frac{1}{5}$, whereas $p(B_1 = 0|A_1 = 1) \cdot p(B_2 = 0|A_2 = 1) = \frac{1}{5} \cdot \frac{1}{5} = \frac{1}{25}$.

The independence assumption in Corollary 4 may appear unrealistic in some applications. Indeed, it can be replaced by somewhat weaker hypotheses.

Definition 9 (memorylessness)**.** *A sequence $(A_1, B_1), \dots, (A_k, B_k)$ of input-output pairs for a channel Γ is* memoryless *if*

$$p(b_k | a_1 \dots a_k, b_1 \dots b_{k-1}) \quad = \quad p(b_k | a_k) \tag{61}$$

whenever $p(a_1 \dots a_k, b_1 \dots b_{k-1}) > 0$.

Definition 10 (absence of feedback)**.** *A sequence $(A_1, B_1), \dots, (A_k, B_k)$ of input-output pairs for a channel Γ has* no feedback *if*

$$p(a_k | a_1 \dots a_{k-1}, b_1 \dots b_{k-1}) \quad = \quad p(a_k | a_1 \dots a_{k-1}) \tag{62}$$

whenever $p(a_1 \dots a_{k-1}, b_1 \dots b_{k-1}) > 0$.

Theorem 9. *A sequence $(A_1, B_1), \dots, (A_k, B_k)$ of input-output pairs for a channel Γ has the independent symbols property if and only if it is memoryless and has no feedback.*[13]

Proof. First, we prove independence of symbols by means induction on k. We show

$$p(a_1 \dots a_k, b_1 \dots b_k) \quad = \quad p(b_1|a_1) \cdot \ldots \cdot p(b_k|a_k) \cdot p(a_1 \dots a_k),$$

whenever the last probability is > 0. The case of $k = 1$ is trivial. To show the induction step, we have, from (61),

$$p(a_1 \dots a_k, b_1 \dots b_k) \quad = \quad p(b_k|a_k) \cdot p(a_1 \dots a_k, b_1 \dots b_{k-1}),$$

whenever $p(a_1 \dots a_k, b_1 \dots b_{k-1}) > 0$, and from (62)

$$p(a_1 \dots a_k, b_1 \dots b_{k-1}) \quad = \quad p(a_1 \dots a_{k-1}, b_1 \dots b_{k-1}) \cdot \frac{p(a_1 \dots a_k)}{p(a_1 \dots a_{k-1})}$$

whenever $p(a_1 \dots a_{k-1}, b_1 \dots b_{k-1}) > 0$. But, by the induction hypothesis,

$$\frac{p(a_1 \dots a_{k-1}, b_1 \dots b_{k-1})}{p(a_1 \dots a_{k-1})} \quad = \quad p(b_1|a_1) \cdot \ldots \cdot p(b_{k-1}|a_{k-1}),$$

[13]Thanks go to Michał Skrzypczak, who contributed the implication *only if* of this theorem (in 2009, as a student of the course).

which gives the claim. If $p(a_1 \dots a_{k-1}, b_1 \dots b_{k-1}) = 0$, we have

$$\underbrace{p(a_1 \dots a_k, b_1 \dots b_k)}_{=0} = \underbrace{p(b_1|a_1) \cdot \dots \cdot p(b_{k-1}|a_{k-1})}_{=0} \cdot p(b_k|a_k) \cdot p(a_1 \dots a_k).$$

On the other hand, if $p(a_1 \dots a_k, b_1 \dots b_{k-1}) = 0$ and $p(a_1 \dots a_{k-1}, b_1 \dots b_{k-1}) > 0$, we have

$$0 = p(a_k|a_1 \dots a_{k-1}, b_1 \dots b_{k-1}) \overset{(62)}{=} p(a_k|a_1 \dots a_{k-1})$$

This contradict the assumption that $p(a_1 \dots a_k) > 0$.

Before we prove that memorylessness and absence of feedback is a consequence of independence of symbols, we show that

$$p(a_1 \dots a_k, b_1 \dots b_{k-1}) \quad = \quad p(b_1|a_1) \cdot \dots \cdot p(b_{k-1}|a_{k-1}) \cdot p(a_1 \dots a_k). \tag{63}$$

Indeed

$$p(a_1 \dots a_k, b_1 \dots b_{k-1}) = \sum_{b_k \in \mathcal{B}} p(a_1 \dots a_k, b_1 \dots b_k) =$$

$$= \sum_{b_k \in \mathcal{B}} p(b_1|a_1) \cdot \dots \cdot p(b_k|a_k) \cdot p(a_1 \dots a_k) = p(b_1|a_1) \cdot \dots \cdot p(b_{k-1}|a_{k-1}) \cdot p(a_1 \dots a_k).$$

Now, we prove memorylessness assuming that $p(a_1 \dots a_k, b_1 \dots b_{k-1}) > 0$ (as a consequence $p(a_1 \dots a_k) > 0$)

$$p(b_k|a_1 \dots a_k, b_1 \dots b_{k-1}) = \frac{p(a_1 \dots a_k, b_1 \dots b_k)}{p(a_1 \dots a_k, b_1 \dots b_{k-1})} \overset{(60)}{\underset{(63)}{=}}$$

$$= \frac{p(b_1|a_1) \cdot \dots \cdot p(b_k|a_k) \cdot p(a_1 \dots a_k)}{p(b_1|a_1) \cdot \dots \cdot p(b_{k-1}|a_{k-1}) \cdot p(a_1 \dots a_k)} = p(b_k|a_k)$$

and the absence of feedback assuming that $p(a_1 \dots a_{k-1}, b_1 \dots b_{k-1}) > 0$ and $p(a_1 \dots a_k) > 0$.

$$p(a_k|a_1 \dots a_{k-1}, b_1 \dots b_{k-1}) = \frac{p(a_1 \dots a_k, b_1 \dots b_{k-1})}{p(a_1 \dots a_{k-1}, b_1 \dots b_{k-1})} \overset{(63)}{\underset{(60)}{=}}$$

$$= \frac{p(b_1|a_1) \cdot \dots \cdot p(b_{k-1}|a_{k-1}) \cdot p(a_1 \dots a_k)}{p(b_1|a_1) \cdot \dots \cdot p(b_{k-1}|a_{k-1}) \cdot p(a_1 \dots a_{k-1})} = p(a_k|a_1 \dots a_{k-1}).$$

If $p(a_1 \dots a_{k-1}, b_1 \dots b_{k-1}) > 0$ and $p(a_1 \dots a_k) = 0$ we have

$$p(a_k|a_1 \dots a_{k-1}, b_1 \dots b_{k-1}) = 0 = p(a_k|a_1 \dots a_{k-1})$$

□

Remark. The independent symbols property is weaker than the independence of input-output pairs considered in Corollary 4, and in fact it forces neither independence of $A_1, \dots, A_k$, nor independence of $B_1, \dots, B_k$. For, it is enough to consider the faithful channel, which obviously satisfies the equation (60), for any sequence of input-output pairs $(A_1, B_1), \dots, (A_k, B_k)$ (in this case $A_i = B_i$).

Proviso. In what follows, unless stated otherwise, we always assume that the equation (60) holds, whenever a BSC is used several times.

2.5 Improving reliability

Suppose we use a binary symmetric channel (see page 27) Γ given by the matrix $\begin{pmatrix} P & Q \\ Q & P \end{pmatrix}$, where $P > Q$. In this case $\Delta_{\max}(i) = i$, for $i = 0, 1$, and, for any A,

$$\begin{aligned} Pr_C(\Delta_{\max}, A) &= \sum_{b\in\{0,1\}} p(\Delta_{\max}(b)) \cdot p(\Delta_{\max}(b) \to b) \\ &= p(A = 0) \cdot P + p(A = 1) \cdot P \\ &= P, \end{aligned}$$

hence $Pr_E(\Delta_{\max}, A) = Q$. As it does not depend on A, we simply write $Pr_E(\Delta_{\max}) = Q$.

Can we achieve a better result, using the same channel in a more clever way? A natural solution is to send each bit of the message more than once, say 3 times. As the correct transmission is more likely than the error (since $P > Q$), the receiver should decode the message looking at the majority:

$$\begin{array}{ccccccccccc} 0 & \mapsto & 000 & \to & \boxed{\Gamma} & \to & 000 & 001 & 010 & 100 & \mapsto & 0 \\ 1 & \mapsto & 111 & \to & & \to & 011 & 101 & 110 & 111 & \mapsto & 1 \end{array}$$

Then the whole procedure behaves as a (new) channel

$$\begin{array}{ccccc} 0 & \to & \boxed{\Gamma'} & \to & 0 \\ 1 & \to & & \to & 1 \end{array}$$

What is the matrix of this channel?

Using the Corollary above, we can see that, e.g., the probability $p(0|0)$ that the output is 0 if the input has been 0, amounts to

$$p(000|000) + p(001|000) + p(010|000) + p(100|000) = P^3 + 3P^2Q.$$

Similar calculations made for the remaining $p(i|j)$, easily show that Γ' is again a binary symmetric channel, with the matrix

$$\begin{pmatrix} P^3 + 3P^2Q & Q^3 + 3Q^2P \\ Q^3 + 3Q^2P & P^3 + 3P^2Q \end{pmatrix}$$

Clearly $Q^3 + 3Q^2P < P^3 + 3P^2Q$, hence the error probability of Γ' is

$$Pr_E(\Delta_{\max}) = Q^3 + 3Q^2P.$$

To see that this is indeed less than Q, it is enough to examine the function $Q^3 + 3Q^2(1 - Q) - Q$, which turns out to be negative for $0 < Q < \frac{1}{2}$.

More generally, if the sender sends each bit n times and the receiver decides by majority (for simplicity, suppose that n is odd), we obtain the BSC channel with the matrix

$$\begin{pmatrix} \sum_{i=\lceil \frac{n}{2}\rceil}^{n} \begin{pmatrix} n \\ i \end{pmatrix} P^i \cdot Q^{n-i} & \sum_{i=0}^{\lfloor \frac{n}{2}\rfloor} \begin{pmatrix} n \\ i \end{pmatrix} P^i \cdot Q^{n-i} \\ \sum_{i=0}^{\lfloor \frac{n}{2}\rfloor} \begin{pmatrix} n \\ i \end{pmatrix} P^i \cdot Q^{n-i} & \sum_{i=\lceil \frac{n}{2}\rceil}^{n} \begin{pmatrix} n \\ i \end{pmatrix} P^i \cdot Q^{n-i} \end{pmatrix}$$

Now the probability of error is

$$Pr_E(\Delta_{\max}) = \sum_{i=0}^{\lfloor \frac{n}{2}\rfloor} \begin{pmatrix} n \\ i \end{pmatrix} P^i \cdot Q^{n-i} \le \underbrace{\sum_{i=0}^{\lfloor \frac{n}{2}\rfloor} \begin{pmatrix} n \\ i \end{pmatrix}}_{2^{n-1}} P^{\lfloor \frac{n}{2}\rfloor} \cdot Q^{\lfloor \frac{n}{2}\rfloor}$$

Since $\frac{1}{4} > P \cdot Q$, we have $PQ = \frac{\delta}{4}$, for some $\delta < 1$. Hence

$$Pr_E(\Delta_{\max}) \leq 2^{n-1} \cdot (PQ)^{\lfloor \frac{n}{2} \rfloor} = 2^{n-1} \cdot \frac{\delta^{\lfloor \frac{n}{2} \rfloor}}{2^{2 \cdot \lfloor \frac{n}{2} \rfloor}} = \delta^{\lfloor \frac{n}{2} \rfloor}$$

Therefore $Pr_E(\Delta_{\max}) \to 0$ if $n \to \infty$.

This means that we can make the probability of error arbitrarily small, but it comes at the expense of longer and longer messages. The celebrated Shannon's theorem (which we will learn at the next lecture) shows that, in some sense, this expense is not necessary. To get the intuition that this *may* be possible, observe that our choice of repeating the same symbol has been made for simplicity, but other choices are also possible. For example, while spelling a difficult word (e.g., by phone), one often says the names, e.g., Bravo, Alpha, November, Alpha, Charlie, Hotel (here I have used the *International Radio Operators Alphabet*).

2.6 Hamming distance

For a finite set $\mathcal{A}$ and $n \in \mathbb{N}$, the *Hamming distance* between $u, v \in \mathcal{A}^n$ is defined by

$$d(u,v) \quad = \quad |\{i : u_i \neq v_i\}| \tag{64}$$

It is easy to see that the axioms of the metric space are satisfied:

positivity $d(u,v) = 0 \iff u = v$,

symmetry $d(u,v) = d(v,u)$,

triangle inequality $d(u,w) \leq d(u,v) + d(v,w)$

(the last follows from the fact that $\{i : u_i \neq w_i\} \subseteq \{i : u_i \neq v_i\} \cup \{i : v_i \neq w_i\}$).

Consider a BSC Γ given by a matrix $\begin{pmatrix} P & Q \\ Q & P \end{pmatrix}$ with $P > Q$. The Hamming distance defined above allows for a succinct notation of the conditional probability that the sequence of outputs is $\vec{b} = b_1 \ldots b_k$ if the sequence of inputs is $\vec{a} = a_1 \ldots a_k$. The equation (60) gives us

$$p(b_1 \ldots b_k | a_1 \ldots a_k) \quad = \quad Q^{d(\vec{a},\vec{b})} \cdot P^{k - d(\vec{a},\vec{b})}. \tag{65}$$

2.7 Transmission error

For an input-output pair (A, B), we consider an auxiliary random variable

$$E \quad = \quad A \oplus B,$$

it can be viewed as the error of the transmission by channel. We have

$$p(b|a) \quad = \quad p(E = a \oplus b) \tag{66}$$

Indeed, by definition of BSC

$$p(b|a) = \begin{cases} P & a = b \quad (a \oplus b = 0) \\ Q & a \neq b \quad (a \oplus b = 1) \end{cases}$$

On the other hand,

$$p(E = 0) = p(A = 0) \cdot p(0 \to 0) + p(A = 1) \cdot p(1 \to 1) = P$$

and

$$p(E = 1) = p(A = 0) \cdot p(0 \to 1) + p(A = 1) \cdot p(1 \to q) = Q,$$

so the both sides of (66) coincide, for all a, b.

Now consider a sequence of input-output pairs $(A_1, B_1), \dots, (A_k, B_k)$, satisfying the equation (60). This implies that the random variables $E_1, \dots, E_k$ (where $E_i = A_i \oplus B_i$) are *independent*[14]. Indeed, we have (in what follows, $p(\vec{E} = \vec{e})$, or simply $p(\vec{e})$, abbreviate $p(E_1 = e_1 \wedge \dots \wedge E_k = e_k)$, etc.)

$$p(e_1 \dots e_k) = \sum_{\vec{a}} p(\vec{A} = \vec{a} \wedge \vec{B} = \vec{a} \oplus \vec{e}) = \sum_{\vec{a}} p(\vec{A} = \vec{a}) \cdot p(\vec{B} = \vec{a} \oplus \vec{e} | \vec{A} = \vec{a}),$$

where $\vec{a}$ ranges over those vectors for which $p(\vec{a}) > 0$. But, using (60) and (66), we have

$$\begin{aligned} p(\vec{B} = \vec{a} \oplus \vec{e} | \vec{A} = \vec{a}) &= p(B_1 = a_1 \oplus e_1 | A_1 = a_1) \cdot \ldots \cdot p(B_k = a_k \oplus e_k | A_k = a_k) && (67) \\ &= p(E_1 = e_1) \cdot \ldots \cdot p(E_k = e_k) && (68) \end{aligned}$$

for any $\vec{a}$, hence

$$p(e_1 \dots e_k) \quad = \quad p(e_1) \cdot \ldots \cdot p(e_k)$$

as desired.

2.8 Channel coding

Suppose we dispose of a binary symmetric channel Γ as above ($P > Q$), and wish to send a value of a random variable X with values in $\mathcal{X} = \{x_1, \dots, x_m\}$. In the early lectures we have studied how to efficiently encode the values of X. If the channel is faithful, all we need is to find an optimal encoding $\varphi : \mathcal{X} \to \{0,1\}^*$ and then send the message bit by bit. The average length (time) of transmission will be bounded by $H(X) + 1$ (c.f. the Shannon-Fano coding in the proof of Theorem 5). On the other hand, we can always encode $\mathcal{X}$ using strings of length $\lceil \log m \rceil$, which gives the bound for the worst-case time of the transmission. (The two bounds may be not achievable by the same encoding.)

However, if the channel is insecure, this method would lead to errors. As the example on the page 34 suggests, we should rather use redundant, and hence non-optimal encoding. In what follows, we will struggle for a method which should reconcile two antagonistic objectives:

- keep redundancy as small as possible,
- keep the error probability as small as possible.

We first describe a general scheme of the method.

Transmission algorithm Suppose we are given a random variable X with values in $\mathcal{X} = \{x_1, \dots, x_m\}$.

1. Choose $n \in \mathbb{N}$, and $C \subseteq \{0,1\}^n$ with $|C| = m$.
2. Choose $\varphi : \mathcal{X} \overset{1:1}{\to} C$. Clearly φ is an instantaneous code. Denote $\varphi \circ X$ as $\vec{A}$.
3. Send the string $\vec{A} = a_1 \dots a_n$ by the channel Γ, bit by bit. Let the output received from the channel be $\vec{B} = b_1 \dots b_n$. Assuming that the use of the channel is memoryless and feedback–free, we have (eq. (65))

$$p(b_1 \dots b_n | a_1 \dots a_n) \quad = \quad Q^{d(\vec{a},\vec{b})} \cdot P^{n - d(\vec{a},\vec{b})}.$$

[14] The converse is not true in general, but the independence of $E_1, \dots, E_k$, *and* independence of $(E_1, \dots, E_k)$ from $(A_1, \dots, A_k)$ implies the equation (60).

4. To decode, given $\vec{B} = b_1 \dots b_n$, choose $\Delta(b_1 \dots b_n) = a_1 \dots a_n \in C$ which maximizes $p(b_1 \dots b_n | a_1 \dots a_n)$ (like in the maximal likelihood rule).

   Clearly, this is achieved if $a_1 \dots a_n$ is a code-word in C nearest to $b_1 \dots b_n$ w.r.t the Hamming distance. (We fix some policy of choice if there is more than one word with this property.)

   This Δ is called the *nearest neighbour rule*.

The method described above can be viewed as a new channel (from C to C)

$$C \ni a_1 \dots a_n \to \boxed{\Gamma} \to b_1 \dots b_n \to \Delta(b_1 \dots b_n) \in C$$

with the probability of error

$$Pr_E(\Delta, \vec{A}) \quad = p(\Delta \circ \vec{B} \neq \vec{A}).$$

Our first observation is that the worst case is if (the distribution of) $\vec{A}$ is *uniform*, i.e., $p(\vec{a}) = \frac{1}{m}$, for $\vec{a} \in C$.

Fact 6. *Let $\vec{A}$,$\vec{U}$, be two random variables with values in $C \subseteq \{0,1\}^n$, where $\vec{U}$ is uniform and $\vec{A}$ arbitrary. Then there is a permutation $\sigma : C \overset{1:1}{\to} C$ such that*

$$Pr_E(\Delta, \sigma \circ \vec{A}) \quad \leq \quad Pr_E(\Delta, \vec{U}).$$

First we prove the Lemma:

Lemma 4. *Let $\alpha_1, \dots, \alpha_m$ be a sequence of real numbers and let $p_1, \dots, p_m \in [0,1]$ be such that $p_1 + \dots + p_m = 1$. If $\alpha_1 \leq \dots \leq \alpha_m$ and $p_1 \geq \dots \geq p_m$, then*

$$\sum_{i=1}^{m} p_i \alpha_i \leq \frac{1}{m} \sum_{i=1}^{m} \alpha_i.$$

Proof. We use induction over m. The case $m = 1$ is trivial. During the induction step we consider sequences $\alpha_1 \leq \dots \leq \alpha_m$ and $p_1 \geq \dots \geq p_m$.
We have $p_m \leq \frac{1}{m}$ because the sequence p_i is decreasing. We assume that $p_m = \frac{1}{m} - h$ for some $h \geq 0$. On the other hand, we conclude from the ordering of the sequence α_i that $\alpha_m \geq \frac{1}{m-1} \sum_{i=1}^{m-1} \alpha_i$.
We deduce from inductive assumption that

$$\frac{p_1}{p_1 + \dots + p_{m-1}} \alpha_1 + \dots + \frac{p_{m-1}}{p_1 + \dots + p_{m-1}} \alpha_{m-1} \leq \frac{1}{m-1} \sum_{i=1}^{m-1} \alpha_i.$$

Hence

$$p_1\alpha_1 + \dots + p_{m-1}\alpha_{m-1} + p_m\alpha_m \leq \underbrace{(p_1 + \dots + p_{m-1})}_{1-p_m} \cdot \frac{1}{m-1} \cdot \sum_{i=1}^{m-1} \alpha_i + p_m\alpha_m =$$

$$\left(\frac{m-1}{m} + h\right) \cdot \frac{1}{m-1} \cdot \sum_{i=1}^{m-1} \alpha_i + \left(\frac{1}{m} - h\right) \cdot \alpha_m = \frac{1}{m} \sum_{i=1}^{m} \alpha_i + h \cdot \underbrace{\left(\frac{1}{m-1} \sum_{i=1}^{m-1} \alpha_i - \alpha_m\right)}_{\leq 0} \leq \frac{1}{m} \sum_{i=1}^{m} \alpha_i.$$

□

Now, we prove Fact 6:

Proof. According to the definition

$$Pr_E(\Delta, \vec{A}) = \sum_{\vec{a}\in C} p(\vec{A} = \vec{a})p(\Delta \circ \vec{B} \neq \vec{a}|\vec{A} = \vec{a})$$

We introduce a transmission error $\vec{E} = \vec{A} \oplus \vec{B}$. We have $p(\vec{B} = \vec{b}|\vec{A} = \vec{a}) = p(\vec{E} = \vec{a} \oplus \vec{b})$ and

$$p(\Delta\circ\vec{B} \neq \vec{a}|\vec{A} = \vec{a}) = \sum_{\vec{b}:\Delta(\vec{b})\neq\vec{a}} p(\vec{B} = \vec{b}|\vec{A} = \vec{a}) = \sum_{\vec{b}:\Delta(\vec{b})\neq\vec{a}} p(\vec{E} = \vec{a}\oplus\vec{b}) = \sum_{\vec{e}:\Delta(\vec{a}\otimes\vec{e})\neq\vec{a}} p(\vec{E} = \vec{e}) = p(\Delta(\vec{a}\oplus\vec{E}) \neq \vec{a}).$$

Thus

$$Pr_E(\Delta, \vec{A}) = \sum_{\vec{a}\in C} p(\vec{A} = \vec{a})p(\Delta(\vec{a} \oplus \vec{E}) \neq \vec{a})$$

Similarly, we define $\vec{E}_U$ — a transmission error for our channel with input variable $\vec{U}$ and we obtain

$$Pr_E(\Delta, \vec{U}) = \sum_{\vec{a}\in C} p(\vec{U} = \vec{a})p(\Delta(\vec{a} \oplus \vec{E}_U) \neq \vec{a}) = \frac{1}{|C|}\sum_{\vec{a}\in C} p(\Delta(\vec{a} \oplus \vec{E}_U) \neq \vec{a}) = \frac{1}{|C|}\sum_{\vec{a}\in C} p(\Delta(\vec{a} \oplus \vec{E}) \neq \vec{a}).$$

The last equality holds because $p(\vec{E} = e) = p(\vec{E}_U = e)$ for each e.

Now, we apply Lemma 4. For $C = \{\vec{a}_1, \ldots, \vec{a}_m\}$, we have $\alpha_i = p(\Delta(\vec{a}_i \oplus \vec{E}) \neq \vec{a}_i)$. We may assume without the loss of generality that $\alpha_1 \leq \cdots \leq \alpha_m$. We define $p_i = p(\sigma \circ \vec{A} = \vec{a}_i)$, where σ is a permutation such that $p_1 \geq \cdots \geq p_m$. From the Lemma we obtain

$$\underbrace{\sum_{\vec{a}\in C} p(\sigma \circ \vec{A} = \vec{a})p(\Delta(\vec{a} \oplus \vec{E}) \neq \vec{a})}_{Pr_E(\Delta,\sigma\circ\vec{A})} \leq \underbrace{\frac{1}{m}\sum_{\vec{a}\in C} p(\Delta(\vec{a} \oplus \vec{E}) \neq \vec{a})}_{Pr_E(\Delta,\vec{U})}$$

□

Hence, in order to estimate the efficiency of our method in terms of $Pr_E(\Delta, \vec{A})$, we may assume without loss of generality that $\vec{A}$ is uniform.

In view of the fact just proved, in order to estimate the error probability for arbitrary variable, it is enough to consider $\vec{A}$ with uniform distribution. In this case $Pr_E(\Delta, \vec{A})$ depends only on C, hence we will denote it just by $Pr_E(\Delta, C)$.

Corollary 5.

$$Pr_E(\Delta, C) \leq \frac{1}{|C|}\sum_{\vec{a}\in C} p(\Delta(\vec{a} \oplus \vec{E}) \neq \vec{a})$$

2.9 Transmission rate

For a given source X and code C such that $|X| = r$ and $|C| = m$, the speed of transmission is measured as a ratio between the information generated by source $H_r(X)$ and length of the code C used to transmit it. In general the size of the source alphabet may be different than the size of code. Using k codewords we encode $j = \lfloor k \log_r m\rfloor$ symbols, so in the limit we obtain

$$\frac{H_r(X^j)}{kL(C)} = \frac{H_r(X)\lfloor k \log_r m\rfloor}{kL(C)} \approx \frac{H_r(X) \log_r m}{L(C)}$$

Since we assumed that the probability distribution of source symbols is uniform and all the codewords have the length n we obtain $H_r(X) = 1$ and $L(C) = n$.

Definition 11. *Let $\mathcal{A}$ be an alphabet with $|\mathcal{A}| = r \geq 2$. The* transmission rate *of a code $C \subseteq \mathcal{A}^n$ is*

$$R_r(C) \quad = \quad \frac{\log_r |C|}{n}.$$

If $\mathcal{A} = \{0, 1\}$, we traditionally abbreviate $R_2 = R$.

Note that the two objectives stated on the page 36 mean that we want both $Pr_E(\Delta, C)$ and $R(C)$ to be as small as possible.

Examples We start with a noisy typewriter described on the page 26. Although it does not exactly fit to the setting above, the basic concepts are well illustrated.

Clearly this channel can produce many errors. However, if we have used only each second letter, say $a, c, e, g, \ldots, u, w, y$, then the received message can be always decoded in the correct way.

Can we use this observation to transmit faithfully arbitrary texts?

A simple idea is to encode the letters by pairs, still using only a half of the alphabet, e.g.

a	aa
b	ac
c	cc
d	ce
...	

For example, `hhqtfeabtvjjceefbb` should be read as `greatidea`.

Here, in order to transmit one letter, we need to send two, hence the transmission rate is $\frac{1}{2}$. Can we do better?

If we have an auxiliary symbol, # say, which can be decoded without error, we can encode (somewhat like in the musical notation)

a	a
b	# a
c	c
d	# c
...	

Here the average length of the encoding is $\frac{1}{2} \cdot 1 + \frac{1}{2} \cdot 2 = \frac{3}{2}$, so we can estimate the rate by $\frac{2}{3}$. We can apply this idea without extending the alphabet, by choosing one letter, y say, to play the role of #, and additionally encode y by yy (which decreases the rate slightly).

In the second example we assume a BSC Γ, and explore the improvement we have made on the page 34 to send just two bits 0 and 1. Assume $n = k \cdot \ell$, and let $m = 2^k$. Then every string of bits $a_1 \ldots a_k$ can be encoded by $a_1^\ell \ldots a_k^\ell \in \{0, 1\}^n$, which defines a code of rate $\frac{1}{\ell}$. Refining the analysis of page 35 we can see that, for arbitrary k, we can make $Pr_E(\Delta, C)$ arbitrary small if ℓ is sufficiently large. Roughly speaking, as soon as a BSC is not completely chaotic (i.e., $P \neq Q$), we can transmit arbitrary text with arbitrary small error, but the price to pay is the slowing down the transmission rate almost to 0.

The celebrated Shannon Theorem shows that the situation is much better than that: we can achieve the same thing with the rate close to a *positive* constant, namely the channel capacity C_Γ.

Before stating and proving the theorem, we show a kind of converse (lower bound) result, stating that if no error is permitted, the transmission rate cannot exceed C_Γ. In this fact, Γ can be an arbitrary channel over some input alphabet $\mathcal{A}$, with $|\mathcal{A}| = r \geq 2$ (not necessarily[15] BSC). We assume that $\vec{A}$ is a uniform random

[15]Actually, the reader may notice that for a BSC, the statement is uninteresting, as the zero–error assumption holds only for a faithful channel.

variable with values in some $C \subseteq \mathcal{A}^n$. We let Δ be any decision rule (possibly the best one). We assume, as usual, the independence of symbols equation (60) (c.f. Proviso on page 33).

Fact 7. *If $Pr_E(\Delta, \vec{A}) = 0$ then*

$$R_r(C) \quad \leq \quad \log_r 2 \cdot C_\Gamma.$$

In particular, if $r = 2$, we have

$$R(C) \quad \leq \quad C_\Gamma.$$

Proof. Let $\vec{A} = (A_1, \ldots, A_n)$ and $\vec{B} = (B_1, \ldots, B_n)$ be as in the transmission algorithm. Using the equation (60) we verify by an easy calculation that

$$H(\vec{B}|\vec{A}) \quad = \quad H(B_1|A_1) + \ldots + H(B_n|A_n). \tag{69}$$

We also have (7)

$$H(\vec{B}) \quad \leq \quad H(B_1) + \ldots + H(B_n)$$

Hence

$$\begin{aligned} I(\vec{A}, \vec{B}) &= H(\vec{B}) - H(\vec{B}|\vec{A}) \\ &\leq \sum_{i=1}^{n} H(B_i) - \sum_{i=1}^{n} H(B_i|A_i) \\ &= \sum_{i=1}^{n} \underbrace{(H(B_i) - H(B_i|A_i))}_{I(A_i,B_i)} \\ &\leq n \cdot C_\Gamma \end{aligned}$$

(by definition of C_Γ).
Clearly $I_r(\vec{A}, \vec{B}) = \log_r 2 \cdot I(\vec{A}, \vec{B})$, hence the above implies

$$I_r(\vec{A}, \vec{B}) \quad \leq \quad \log_r 2 \cdot n \cdot C_\Gamma$$

On the other hand, we have

$$\begin{aligned} I_r(\vec{A}, \vec{B}) &= H_r(\vec{A}) - \underbrace{H_r(\vec{A}|\vec{B})}_{0} \\ &= \log_r m \end{aligned}$$

where $m = |C|$. Here $H(\vec{A}|\vec{B})$ vanishes since the assumption $Pr_E(\Delta, \vec{A}) = 0$ implies that $\vec{A}$ is a function of $\vec{B}$, namely $\vec{A} = \Delta(\vec{B})$ (c.f. (27)). Next, $H_r(\vec{A}) = \log_r m$, since $\vec{A}$ is uniform, by assumption.
Hence we have

$$R_r(C) = \frac{\log_r m}{n} \leq \log_r 2 \cdot C_\Gamma$$

as required. □

2.10 Shannon channel coding theorem

We are ready to state the basic result of information theory, due to Claude Shannon (1948). Intuitively it says that message transmission through a noisy channel with arbitrarily small error probability is possible, with the transmission rate arbitrarily close to the channel capacity, provided that the length of code is sufficiently large. We prove the theorem only for BSC, for the general setting see, e.g., [1] (Theorem 8.7.1). The proof below follows [2].

Theorem 10 (Shannon Channel Coding Theorem). *Let Γ be a binary symmetric channel (BSC) with a matrix $\begin{pmatrix} P & Q \\ Q & P \end{pmatrix}$, where $P > Q$. Then $\forall \varepsilon, \delta > 0 \ \exists n_0 \quad \forall n \geq n_0 \quad \exists C \subseteq \{0,1\}^n$*

$$C_\Gamma - \varepsilon \leq \quad R(C) \quad \leq C_\Gamma \tag{70}$$

$$Pr_E(\Delta, C) \quad \leq \delta \tag{71}$$

We first informally describe the basic idea. Suppose an input $\vec{A} = a_1 \dots a_n$ is turned into the output $\vec{B} = b_1 \dots b_n$. What is the *expected* distance between $\vec{A}$ and $\vec{B}$? As this distance amounts to the number of transmission errors, and the probability of one error is Q, the Law of Large Numbers tells us that $d(\vec{A}, \vec{B})$ approaches to $Q \cdot n$ if $n \to \infty$. Now, if the decoding fails, i.e., $\Delta(\vec{B}) \neq \vec{A}$, it is useful to distinguish between two possible "reasons" for that:

- $\vec{B}$ is "far" from $\vec{A}$,
- $\vec{B}$ is not that far, but a confusion arises, because some $\vec{A}' \neq \vec{A}$ is at least as good as $\vec{A}$,

where "far" means: exceeding the expected value $Q \cdot n$.

The first type of failure is caused by the channel, but it is corrected by Nature: the Law of Large Numbers guarantees that a big distance between $\vec{A}$ and $\vec{B}$ happens rarely if n is large. The second issue is, to some extent, responsibility of the code designer. Indeed, to prevent confusion, the code-words should be "reasonably far" one from another. Taking the expected distance as the measure of "far", this means that the balls of radius $Q \cdot n$ (in the Hamming metrics) centered in any two code-words should be disjoint. So the question is: how many disjoint balls of radius $Q \cdot n$ can one "pack" in $\{0,1\}^n$? The size of one such ball, as we will see later, can be estimated by $\approx 2^{n \cdot H(Q)}$. This suggests the number of balls of

$$m \approx 2^n : 2^{n \cdot H(Q)} = 2^{n(1-H(Q))} = 2^{n \cdot C_\Gamma},$$

and hence the transmission rate $R(C) \approx C_\Gamma$. The amazing discovery of Shannon is that this bound is really achievable. However, the proof is non-constructive, i.e., it only shows the existence of the desired code.

Lemma 5. *Let Γ be a channel and Δ be a decision rule as stated in the Theorem. Then, for all $\eta, \delta > 0$ there exists n_0 such that for all $n \geq n_0$ and for all $m \leq 2^n$ and for every code $C \subseteq \{0,1\}^n$ such that $|C| = m$ and*

$$Pr_E(\Delta, C) \leq \delta + \frac{1}{m} \sum_{\vec{a} \in C} \sum_{\vec{b} \in C - \{\vec{a}\}} p(d(\vec{b}, \vec{a} \oplus \vec{E}) \leq \rho),$$

where $\rho = n(Q + \eta)$.

To prove the Lemma we use the following

Theorem 11 (Weak Law of Large Numbers). *Let $X_1, X_2, \dots,$ be a sequence of random variables, such that any $X_1, X_2, \dots, X_n$ are independent, and each X_i takes a finite number of real values with the same distribution. Let $\mu = E(X_i)$. Then, for any $\alpha > 0$,*

$$\lim_{n \to \infty} p(|\frac{1}{n} \sum_{i=1}^{n} X_i - \mu| > \alpha) \quad = \quad 0. \tag{72}$$

Proof. By definition of Δ, if, for some $\vec{a} \in C$, $\vec{e} \in \{0,1\}^n$, $d(\vec{a}, \vec{a} \oplus \vec{e}) \le \rho$ and $\forall \vec{b} \in C - \{\vec{a}\}$, $d(\vec{b}, \vec{a} \oplus \vec{e}) > \rho$, then $\Delta(\vec{a} \oplus \vec{e}) = \vec{a}$. Therefore, if $\Delta(\vec{a} \oplus \vec{e}) \neq \vec{a}$ then either $d(\vec{a}, \vec{a} \oplus \vec{e}) > \rho$, or, for some $\vec{b} \in C - \{\vec{a}\}$, $d(\vec{b}, \vec{a} \oplus \vec{e}) \le \rho$. Now we can view the vector $\vec{e}$ as the value of a random variable $\vec{E} = (E_1, \ldots, E_n)$, where $E_i = A_i \oplus B_i$ are as in the page 35. Recall that $E_1, \ldots, E_n$ are independent and have the identical distribution $p(E_i = 0) = P$, $p(E_i = 1) = Q$.

Then the above observation induces the following inequality, for a fixed $\vec{a} \in C$.

$$p(\Delta(\vec{a} \oplus \vec{E}) \neq \vec{a}) \quad \le \quad p(d(\vec{a}, \vec{a} \oplus \vec{E}) > \rho) + \sum_{\vec{b} \in C - \{\vec{a}\}} p(d(\vec{b}, \vec{a} \oplus \vec{E}) \le \rho). \tag{73}$$

Now, we apply the Weak Law of Large Numbers to the sequence $E_1, E_2, \ldots$. Clearly $E(E_i) = 0 \cdot P + 1 \cdot Q = Q$. Hence $p(|\frac{1}{n} \cdot \sum_{i=1}^n E_i - Q| > \eta) \to 0$ if $n \to \infty$. Therefore

$$p(d(\vec{a}, \vec{a} \oplus \vec{E}) > \rho) \; \le \; p(\frac{1}{n} \cdot \sum_{i=1}^n E_i > Q + \eta) \; \le \; p(|\frac{1}{n} \cdot \sum_{i=1}^n E_i - Q| > \eta) \; \le \; \delta, \tag{74}$$

for n sufficiently large.

Now recall that we wish to estimate $Pr_E(\Delta, C)$. Corollary 5 together with (74)gives us

$$\begin{aligned} Pr_E(\Delta, C) \quad & \le \quad \frac{1}{m} \sum_{\vec{a} \in C} p(\Delta(\vec{a} \oplus \vec{E}) \neq \vec{a}) \\ & \le \quad \frac{1}{m} \sum_{\vec{a} \in C} \left(p(d(\vec{a}, \vec{a} \oplus \vec{E}) > \rho) + \sum_{\vec{b} \in C - \{\vec{a}\}} p(d(\vec{b}, \vec{a} \oplus \vec{E}) \le \rho) \right) & (75) \\ & \le \quad \delta + \frac{1}{m} \sum_{\vec{a} \in C} \sum_{\vec{b} \in C - \{\vec{a}\}} p(d(\vec{b}, \vec{a} \oplus \vec{E}) \le \rho), & (76) \end{aligned}$$

if n is sufficiently large. □

Before proceeding further, we will estimate the size of a ball in metric space with Hamming distance of radius $\lambda \cdot n$, where $\lambda \le \frac{1}{2}$. More specifically, we show the following

Lemma 6.

$$\sum_{i \le \lambda \cdot n} \binom{n}{i} \quad \le \quad 2^{n \cdot H(\lambda)}, \tag{77}$$

where H is the function defined by (53).

Proof. Let $\kappa = 1 - \lambda$. Note first that

$$\begin{aligned} \log_2 \lambda^{\lambda n} \cdot \kappa^{\kappa n} \quad & = \quad n \cdot (\lambda \cdot \log_2 \lambda + \kappa \cdot \log_2 \kappa) \\ & = \quad -n \cdot H(\lambda) \end{aligned}$$

Now it is enough to show that, for all $i \le \lambda n$,

$$\lambda^i \kappa^{n-i} \quad \ge \quad \lambda^{\lambda n} \cdot \kappa^{\kappa n}. \tag{78}$$

Indeed, this will give us

$$1 \ge \sum_{i \le \lambda \cdot n} \binom{n}{i} \lambda^i \kappa^{n-i} \ge \sum_{i \le \lambda \cdot n} \binom{n}{i} \lambda^{\lambda n} \cdot \kappa^{\kappa n}$$

and consequently

$$\sum_{i\leq\lambda\cdot n}\binom{n}{i} \leq \frac{1}{\lambda^{\lambda n}\cdot\kappa^{\kappa n}} = 2^{n\cdot H(\lambda)},$$

as required.

If λn is integer, the inequality (78) is obvious (just replace smaller by bigger). Otherwise, we have $\lambda n = \lfloor\lambda n\rfloor + \Delta\lambda$, $\kappa n = \lfloor\kappa n\rfloor + \Delta\kappa$, $\lfloor\lambda n\rfloor + \lfloor\kappa n\rfloor = n-1$, and $\Delta\lambda + \Delta\kappa = 1$. Since $\kappa \geq \lambda$, we have, for $i \leq \lambda n$,

$$\lambda^i\kappa^{n-i} \geq \lambda^{\lfloor\lambda n\rfloor}\cdot\kappa^{\lfloor\kappa n\rfloor+1} = \lambda^{\lfloor\lambda n\rfloor}\cdot\kappa^{\lfloor\kappa n\rfloor}\underbrace{\kappa^{\Delta\lambda+\Delta\kappa}}_{\geq\lambda^{\Delta\lambda}\cdot\kappa^{\Delta\kappa}} \geq \lambda^{\lambda n}\cdot\kappa^{\kappa n}.$$

This completes the proof of Lemma. □

2.11 The probabilistic argument

We come back to the estimation of $Pr_E(\Delta, C)$. Recall that (76) holds for any code C, if only n is sufficiently large. We will now show that, for sufficiently large n, there *exists* a code C satisfying the conditions (70) and (71) of Shannon's theorem; in particular the second term of (76) should be estimated by $\frac{\delta}{2}$.

To this end, rather than searching for a specific code C with the desired property, we will use the *probabilistic method*.

Lemma 7. *Let Γ be a channel and Δ be a decision rule as stated in the Theorem. Then, for all $0 < \eta \leq \frac{1}{2} - Q$ and $m, n \in \mathbb{N}$ such that $m < 2^n$ there exists a code $C \subseteq \{0,1\}^n$ such that $|C| = m$ and*

$$\frac{1}{m}\sum_{\vec{a}\in C}\sum_{\vec{b}\in C-\{\vec{a}\}} p(d(\vec{b}, \vec{a}\oplus\vec{E}) \leq \rho) \leq m\cdot 2^{n(H(Q+\eta)-1)},$$

where $\rho = n(Q+\eta)$.

Proof. Fix $m < 2^n$. Let $\mathcal{C}$ be the set of all sequences $c_1, \ldots, c_m \in \{0,1\}^n$, with $c_i \neq c_j$, for $i \neq j$. Let $N = |\mathcal{C}|$. Clearly

$$N = \binom{2^n}{m}\cdot m!$$

In what follows, we use symbol $\bar{C}$ for a sequence in $\mathcal{C}$, and let $C = \{c_1, \ldots, c_m\}$ be a set of values of $\bar{C}$.

Now the probabilistic argument, due to Claude Shannon, is based on the following simple observation. If, for any δ,

$$\frac{1}{N}\sum_{\bar{C}}\frac{1}{m}\sum_{\vec{a}\in C}\sum_{\vec{b}\in C-\{\vec{a}\}} p(d(\vec{b}, \vec{a}\oplus\vec{E}) \leq \rho) \leq \delta$$

then there exists a code C_0, such that

$$\frac{1}{m}\sum_{\vec{a}\in C_0}\sum_{\vec{b}\in C_0-\{\vec{a}\}} p(d(\vec{b}, \vec{a}\oplus\vec{E}) \leq \rho) \leq \delta.$$

Note that

$$\begin{aligned}\frac{1}{N}\sum_{\bar{C}}\frac{1}{m}\sum_{\vec{a}\in C}\sum_{\vec{b}\in C-\{\vec{a}\}} p(d(\vec{b}, \vec{a}\oplus\vec{E}) \leq \rho) &\leq \frac{1}{N}\sum_{\bar{C}}\frac{1}{m}\sum_{i=1}^{m}\sum_{j\neq i} p(d(c_j, c_i\oplus\vec{E}) \leq \rho) \\ &= \frac{1}{m}\sum_{i=1}^{m}\sum_{j\neq i}\underbrace{\frac{1}{N}\sum_{\bar{C}} p(d(c_j, c_i\oplus\vec{E}) \leq \rho)}_{(*)} \qquad (79)\end{aligned}$$

We will now estimate (*), for a *fixed* pair of indices $i \neq j$.

For $\vec{e} \in \{0,1\}^n$, let $S_\rho(\vec{e})$ be the ball in $\{0,1\}^n$ of radius ρ centered in $\vec{e}$, i.e.,

$$S_\rho(\vec{e}) \quad = \quad \{\vec{b} \in \{0,1\}^n : d(\vec{b},\vec{e}) \leq \rho\}.$$

It is easy to see that

$$d(\vec{b}, \vec{a} \oplus \vec{e}) \leq \rho \quad \Longleftrightarrow \quad \vec{b} \oplus \vec{a} \in S_\rho(\vec{e}).$$

Hence

$$\begin{aligned} \frac{1}{N}\sum_{\bar{C}} p(d(c_j, c_i \oplus \vec{E}) \leq \rho) \quad &= \quad \frac{1}{N}\sum_{\bar{C}} p\left(c_i \oplus c_j \in S_\rho(\vec{E})\right) \\ &= \sum_{\vec{e}\in\{0,1\}^n} p(\vec{E}=\vec{e}) \cdot \underbrace{\frac{1}{N}\sum_{\bar{C}} \chi(c_i \oplus c_j \in S_\rho(\vec{e}))}_{(**)} \end{aligned} \tag{80}$$

where χ is the truth function, i.e.,

$$\chi(\varphi) = \begin{cases} 1 & \text{if } \varphi \text{ holds} \\ 0 & \text{otherwise} \end{cases}$$

We now estimate the value of (**), for a fixed $\vec{e}$. Obviously any vector different from 0^n occurs as a value of $c_i \oplus c_j$, for some sequence $\bar{C} \in \mathcal{C}$, and it is easy to see that each such vector occurs in this role the same number of times, i.e.,

$$|\{\bar{C} : \vec{a} = c_i \oplus c_j\}| = |\{\bar{C} : \vec{b} = c_i \oplus c_j\}| = \frac{N}{2^n - 1}$$

for any $\vec{a}, \vec{b} \in \{0,1\}^n - \{0^n\}$. Hence each $\vec{a} \in S_\rho(\vec{e}) - \{0^n\}$ contributes the value $\frac{N}{2^n-1}$ to the sum $\sum_{\bar{C}} \chi(c_i \oplus c_j \in S_\rho(\vec{e}))$, i.e.,

$$\sum_{\bar{C}} \chi(c_i \oplus c_j \in S_\rho(\vec{e})) \quad = \quad \frac{N}{2^n - 1}|S_\rho(\vec{e}) - \{0^n\}|$$

Therefore we further have

$$\begin{aligned} \sum_{\vec{e}\in\{0,1\}^n} p(\vec{E}=\vec{e}) \cdot \frac{1}{N}\sum_{\bar{C}} \chi(c_i \oplus c_j \in S_\rho(\vec{e})) \quad &= \quad \sum_{\vec{e}\in\{0,1\}^n} p(\vec{E}=\vec{e}) \cdot \frac{1}{2^n-1}|S_\rho(\vec{e}) - \{0^n\}| \\ &= \quad \frac{1}{2^n-1}|S_\rho(\vec{e}) - \{0^n\}| \end{aligned}$$

(since $\sum_{\vec{e}} p(\vec{E} = \vec{e}) = 1$). Now, from (77), we have

$$|S_\rho(\vec{e}) - \{0^n\}| \quad \leq \quad 2^{n \cdot H(Q+\eta)}$$

(recall that $\rho = n(Q + \eta)$). Coming back to (79), we have

$$\begin{aligned} \frac{1}{N}\sum_{\bar{C}} \frac{1}{m}\sum_{\vec{a}\in C}\sum_{\vec{b}\in C-\{\vec{a}\}} p(d(\vec{b}, \vec{a} \oplus \vec{E}) \leq \rho) \quad &\leq \quad \frac{1}{m}\sum_{i=1}^{m}\sum_{j\neq i} \frac{1}{2^n-1} \cdot 2^{n\cdot H(Q+\eta)} \\ &= \quad \frac{1}{m} \cdot m \cdot \underbrace{(m-1) \cdot \frac{1}{2^n-1}}_{\leq \frac{m}{2^n}} \cdot 2^{n\cdot H(Q+\eta)} \\ &\leq \quad m \cdot 2^{n(H(Q+\eta)-1)} \end{aligned} \tag{81}$$

Hence, according to the probabilistic argument there exists a code C_0 that satisfies thesis of the Lemma. $\square$

Now, being equipped with Lemmata, we are prepared for proving the Theorem.

Proof. We know by Lemmata 5 and 7 that forall $\delta > 0$ and $0 < \eta < \frac{1}{2} - Q$ there exists n_0 such that forall $n > n_0$ and $m < 2^n$ there exists a code $C \subseteq \{0,1\}^n$ such that $|C| = m$ and

$$Pr_E(\Delta, C) \quad \le \quad \frac{\delta}{2} + m \cdot 2^{n \cdot H(Q+\eta) - 1}$$

We are given $\varepsilon > 0$ and $\delta > 0$. First, using the continuity of function H, we choose η such that

$$C_\Gamma - \frac{1}{3} \cdot \varepsilon \le 1 - H(Q + \eta)$$

and we obtain n_0. Next, we select

$$n > \max\left(n_0, \frac{3(1 - \log \delta)}{\varepsilon}\right)$$

which is additionally large enough to find k, such that

$$C_\Gamma - \varepsilon \le \frac{k}{n} \le C_\Gamma - \frac{2}{3} \cdot \varepsilon.$$

Then we take $m = 2^k$ and we obtain code $C \subseteq \{0,1\}^n$ such that $|C| = m$, which has the following properties:

$$C_\Gamma - \varepsilon \le \frac{\log_2 m}{n} = R(C) \le C_\Gamma - \frac{2}{3} \cdot \varepsilon$$

$$\begin{aligned}
Pr_E(\Delta, C) \quad &\le \quad \frac{\delta}{2} + m \cdot 2^{n \cdot H(Q+\eta) - 1} \\
&= \quad \frac{\delta}{2} + 2^{n \cdot \left(\frac{\log_2 m}{n} - (1 - H(Q+\eta))\right)} \\
&\le \quad \frac{\delta}{2} + 2^{n \cdot \left(C_\Gamma - \frac{2}{3} \cdot \varepsilon - (C_\Gamma - \frac{1}{3} \cdot \varepsilon)\right)} \\
&\le \quad \frac{\delta}{2} + 2^{-\frac{1}{3} \cdot \varepsilon n} \\
&\le \quad \frac{\delta}{2} + 2^{-\frac{1}{3} \cdot \varepsilon \left(\frac{3(1 - \log \delta)}{\varepsilon}\right)} \\
&= \quad \delta
\end{aligned}$$

□

2.12 Error correcting codes

The Shannon channel coding theorem—neither its statement nor proof—give us no easy method how to construct an optimal code. However, in practice we may wish to trade optimality for efficiency. For example, we may be happy if an algorithm works correctly as long as the number of errors does not exceed a certain threshold. The following concepts will be useful.

Recall the transmission algorithm of section 2.8

$$C \ni a_1 \dots a_n \to \boxed{\Gamma} \to b_1 \dots b_n \to \Delta(b_1 \dots b_n) \in C$$

where $C \subseteq \{0,1\}^n$, and Δ is the nearest neighbour rule. An error occurs on a position i whenever $a_i \neq b_i$. The vector $(a_1 \oplus b_1, \ldots, a_n \oplus b_n)$ identifies all the positions where errors have occurred, which is enough to correct them, since the alphabet is binary. The Hamming distance $d(a_1 \ldots a_n,\, b_1 \ldots b_n)$ amounts to the total number of errors.

We say that the code C *corrects k errors*, if for all $v \in C$, and all $w \in \{0,1\}^n$, if $d(v,w) \leq k$ then $\Delta(w) = v$. Note that in this case the vector $v \oplus w$ identifies the errors.

We say that the code C *detects k errors*, if for all $v \in C$, and all $w \in \{0,1\}^n$, if $0 < d(v,w) \leq k$ then $w \notin C$. That is, if no more than k errors have been made (but at least one error) then the receiver discovers that errors happened, although she may ignore the positions and even the exact number of the errors.

Note that the above are combinatorial properties of a code, not referring to the probabilistic nature of transmission errors. They can be characterized in terms of a geometric parameter of a code, namely the minimum Hamming distance

$$d(C) \quad = \quad \min\{d(v,w) : v, w \in C,\ v \neq w\}.$$

The two properties below follow easily from the definitions and we leave the proofs to the reader.

Lemma 8. *A code C corrects k errors if, and only if, $2k + 1 \leq d(C)$.*

Lemma 9. *A code C detects k errors if, and only if, $k < d(C)$.*

For example, the code $\{0^n, 1^n\}$ corrects $\lfloor \frac{n-1}{2} \rfloor$ errors, whereas the *parity code* consisting of all words $w \in \{0,1\}^n$ with an even number of 1's, detects 1 error but does not correct it[16].

Recall that in the transmission algorithm (section 2.8) we first compute a code C from the values of some actual random variable X; besides, these values are often also texts. Suppose that X takes values in $\{0,1\}^n$. Then the mapping that prolongates w by the "parity check"

$$w \quad \mapsto \quad w \left(\sum_i w_i \bmod 2 \right)$$

transforms X into the parity code in $\{0,1\}^{n+1}$. This idea is often used in practice; the reader my wish to learn the principle behind the Polish identification number PESEL[17]. A more sophisticated use of checking bits was invented by Richard Hamming in 1950, leading to a code that always *corrects* one error. It is sometimes presented in a concrete realization of the *Hamming* (7,4) *code*. We present it here in general form, underlined by an elegant mathematical idea.

Hamming $\left(2^k - 1, 2^k - k - 1\right)$ code

Let us start with some intuitive heuristic. Suppose we extend words in $\{0,1\}^n$ by k checking bits. If we transmit an extended word, a single error may occur on any of the $n+k$ positions, or there may be no error. If this information has to be encoded by a sequence of k checking bits, we have the inequality $n+k+1 \leq 2^k$. For $k \leq 1$ this yields $n = 0$, so let us start with $k = 2$. It turns out that this is indeed possible for n and k satisfying the *equality* $n + k + 1 = 2^k$.

Let $k \geq 2$, and take some word $a_1 \ldots a_n$ with $n = 2^k - k - 1$. It is convenient to place the checking bits not at the end of the word but rather on the positions 2^i, for $i = 0, 1, \ldots, k-1$. (So the original letters are "pushed" accordingly.) For example, for $k = 3$ (hence $n = 4$), we transform a word $a_1 a_2 a_3 a_4$ onto the word $x_1 x_2 \ldots x_7$ as follows

$$\begin{array}{ccccccc} \square & \square & a_1 & \square & a_2 & a_3 & a_4 \\ x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \end{array} \tag{82}$$

[16] The reader may notice that *any odd* number of errors is detected there. But our definition od detecting k errors requires that *any* number of errors *smaller* than k should be also detected, which is not true for parity code if $k \geq 3$.

[17] https://en.wikipedia.org/wiki/PESEL

the checking bits occupying the positions 1,2, and 4. In general, the k checking bits are computed by solving the system of k equations, written with variables $x_1, x_2, \ldots, x_{2^k-1}$, where the variable x_t represents the position t in the transformed word. Note that the indices t can be written with k binary digits[18]

$$t \quad = \quad b_0 + b_1 2 + \ldots + b_{k-1} 2^{k-1}. \tag{83}$$

The system consists of k equations, which we enumerate by $i = 0, 1, \ldots, k-1$. The righthand side of each equation is 0, whereas the lefthand side of the ith equation sums up all those x_t, for which the bit b_i in the binary representation of t is 1.

For example, for $k = 3$, we have the equations

$$\begin{aligned} x_1 + x_3 + x_5 + x_7 &= 0 \\ x_2 + x_3 + x_6 + x_7 &= 0 \\ x_4 + x_5 + x_6 + x_7 &= 0 \end{aligned}$$

Now we compute the checking bits for a word $a_1, \ldots, a_n$ from these equations, where the unknowns are x_{2^i}, for $i = 0, 1, \ldots, k$, and the original bits $a_1, \ldots, a_n$, are substituted into variables corresponding to their new positions (like in (82)). The computation is in Z_2 (i.e., mod2). Note that in each equation there is exactly one unknown (namely x_{2^i} in the ith equation), hence the solution is unique. This completes definition of the Hamming code.

Now suppose that after transmission we receive a word $x'_1\, x'_2, \ldots x'_{n+k}$, and we know that at most one error has occurred. How does it alter the equations?

If an error has occurred on position t then precisely those equations will fail where x_t occurs, and these are the equations i, where the $i$the bit of t is 1. For example, if we realize that

$$\begin{aligned} x'_1 + x'_3 + x'_5 + x'_7 &= 0 \\ x'_2 + x'_3 + x'_6 + x'_7 &= 1 \\ x'_4 + x'_5 + x'_6 + x'_7 &= 1, \end{aligned}$$

we know that an error occurred on the position 6. Then the equations reveal t unambiguously. Clearly, if no error has occurred then all equations will be satisfied.

The Hamming code is optimal in the following sense. Note first that if a code $C \subseteq \{0,1\}^m$ corrects t errors then the balls of radius t (w.r.t. the Hamming distance) centered in two distinct code words must be disjoint. Computing the size of a ball in straightforward manner, we obtain the inequality

$$|C| \cdot \left(1 + m + \binom{m}{2} + \ldots + \binom{m}{t} \right) \quad \leq \quad 2^m, \tag{84}$$

this inequality is known as *Hamming's bound*.

For $t = 1$ the inequality boils down to $|C| \cdot (1+m) \leq 2^m$, and the Hamming code (with $|C| = 2^{2^k-k-1}$ and $m = 2^k - 1$) satisfies the *equality* there,

$$2^{2^k-k-1} \cdot \big(1 + (2^k - 1)\big) = 2^{2^k-1}.$$

That is, the balls in question form a partition of $\{0,1\}^m$.

The above observation implies that the inequality in Lemma 8 also turns into equality for the Hamming code (why?), i.e., the minimal distance of two code words is 3. But let us verify it directly. We use the

[18]Don't be confused here. We deliberately count the positions in the word $t = 1, 2, \ldots, 2^k - 1$, but refer to the bits in the binary representation of t by $b_0, b_1, \ldots, b_{k-1}$.

property that the Hamming code is a *linear code*. In general, linearity means that a code $C \subseteq F^m$, for some finite field F, forms a linear subspace of F^m. For a binary code, it just means that C is closed under (component-wise) addition mod2, i.e., if $v, w \in C$ then $v \oplus w \in C$, as well. This property is guaranteed by the fact that the words of the Hamming code constitute all possible solutions of a system of linear equations with the righthand sides being 0. The following characterization takes place.

Lemma 10. *The minimum distance of a linear code $C \subseteq \{0,1\}^n$ equals to the minimum number of 1's in any code word different from 0^n.*

Proof. Note that the vector 0^n belongs to any linear code (as it equals to $w \oplus w$). Now one inequality follows from the fact that the number of 1's in a word w amounts to its distance from 0^n. The other inequality follows from the fact that the distance between v and w amounts to the number of 1's in $v \oplus w$. $\square$

Thus it is enough to verify that, in any Hamming code, there is a word with exactly three 1's. It is obtained by the transformation (82) applied to the word

$$1\underbrace{00\ldots0}_{2^k-2-k}.$$

Indeed, after transformation, $x_3 = 1$, which implies that $x_1 = x_2 = 1$, as well, whereas all other bits equal 0.

3 Złożoność informacyjna Kołmogorowa

3.1 Złożoność informacyjna Kołmogorowa

Odwołajmy się do codziennego doświadczenia. Chyba każdy się zgodzi, że niektóre liczby można zapamiętać łatwiej niż inne i nie zależy to tylko od wielkości liczby. Bez trudu zapamiętamy liczbę $1\underbrace{00\ldots0}_{100}$ czy nawet $100^{\overbrace{100^{\cdots^{100}}}^{100}}$ natomiast zapamietanie 20 "losowych" cyfr sprawi nam kłopot. No, chyba, że to będą np.

31415926535897932384

wtedy, nawet jeśli nie, "trzymamy" tych cyfr w pamięci, możemy je w razie potrzeby odtworzyć, np. korzystając z formuły Leibniza

$$\frac{1}{1} - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \cdots = \frac{\pi}{4}.$$

Nie inaczej ma się sprawa z tekstami. Mamy świadomość, że przy odpowiednim nakładzie pracy i czasu potrafilibyśmy się nauczyć na pamięć wybranej księgi "Pana Tadeusza" (a może nawet całego poematu), natomiast zapamiętanie - powiedzmy - 10 stron "losowych" symboli wydaje się przekraczać możliwości zwykłego człowieka. No chyba, że znowu – znaleźlibyśmy jakiś "klucz", na przykład okazałoby się, że jest to tekst w obcym języku, którego jednak jesteśmy w stanie się nauczyć.

Jakie pojęcia matematyczne kryją się za tym zjawiskiem? W przypadku liczb odpowiedź jest stosunkowo prosta: niektóre liczby całkowite można opisać znacznie krócej niż podając ich rozwinięcia dziesiętne; wystarczy wtedy zapamiętać ów krótszy "opis". W pierwszym przybliżeniu, za złożoność liczby bylibysmy więc skłonni uznać długość jej najkrótszego opisu. Jednak widzieliśmy już na pierwszym wykładzie, że takie postawienie sprawy prowadzi do paradoksu Berry'ego, dlatego też wprowadziliśmy wtedy pojęcie notacji.

A jednak - po lepszym zrozumieniu - idea najktótszego opisu prowadzi do sensownej miary złożoności, którą przedstawimy na tym wykładzie.

Przyjmiemy, że obiekty, które chcemy opisywać, to słowa nad alfabetem $\{0, 1\}$; zarówno liczby naturalne, jak też słowa nad większymi alfabetami można łatwo sprowadzić do tego przypadku.

Ideą Kołmogorowa było, by za złożoność słowa w przyjąć długość najkrótszego programu generującego to słowo, przy wybranym języku programowania, np. języku Pascal. W istocie Kołmogorow nie użył języka Pascal, lecz uniwersalnej maszyny Turinga, jednak - jak wkrótce się przekonamy - wybór ten nie ma większego znaczenia, podobnie jak zresztą wybór innego języka programowania (np. C++ zamiast Pascala).

Zakładamy, że Czytelnik jest zaznajomiony z pojęciem maszyny Turinga[19], choć nie będziemy go używać bardzo intensywnie - jeśli ktoś woli, może nadal myśleć o ulubionym języku programowania. Wszystkie rozważane przez nas maszyny Turinga są deterministyczne i używają binarnego alfabetu wejściowego ($\{0, 1\}$).

Ważną własnością maszyn Turinga jest, że można je kodować za pomocą słów binarnych, jedno z wielu możliwych kodowań opisane jest w wykładzie 1 ze Złożoności obliczeniowej. Nie jest przy tym trudno zagwarantować, by kodowanie było *bezprefiksowe* (tzn. żaden kod nie jest właściwym prefiksem innego).

Zakładając ustalone kodowanie, Turing podał konstrukcję *maszyny uniwersalnej.*

Będziemy używać następującej notacji:

- $M(w) \downarrow$: maszyna M zatrzymuje się dla danych wejściowych w.
- $M(w) \uparrow$: maszyna M zapętla się dla danych wejściowych w.
- $M(w) = v$: $M(w) \downarrow$ i wynikiem obliczenia jest v.

W teorii maszyn Turinga, gdy $M(w) \downarrow$, to zwykle rozróżnia się dwa przypadki: M akceptuje lub odrzuca. W naszych rozważaniach dla prostoty utożsamimy ten ostatni przypadek z $M(w) \uparrow$. Tzn. zakładamy, że jeśli M odrzuca, to wchodzi w nieskończoną pętlę.

Definition 12 (Uniwersalna maszyna Turinga)**.** Uniwersalna maszyna Turinga *jest to dowolna maszyna U o następujących własnościach:*
(1) jeśli na wejściu jest słowo vu, gdzie v jest kodem pewnej maszyny M_v, to U symuluje działanie M_v na u.
W szczególności
(1a) jeśli $M_v(u) \downarrow$, to $U(vu) \downarrow$ i $M_v(u) = U(vu)$,
(1b) jeśli $M_v(u) \uparrow$, to $U(vu) \uparrow$.
(2) Jeśli słowo wejściowe w nie ma prefiksu będącego kodem maszyny, to $U(w) \uparrow$.

Definition 13 (Złożoność Kołmogorowa)**.** Złożonością informacyjną Kołmogorowa *słowa x jest*

$$C_U(x) = \min\{|v| : U(v) = x\}$$

Innymi słowy, $C_U(x)$ jest długością najkrótszego kodu maszyny Turinga wraz z wejściem, $\langle M \rangle y$, takich że $M(y) = x$.

Na pierwszy rzut oka definicja ta istotnie zależy od wyboru kodowania i maszyny uniwersalnej. Istotnie, przyjmując inne kodowanie, moglibyśmy otrzymać inną maszynę uniwersalną U' i w konsekwencji inną wartość $C_{U'}(x)$. Okazuje się jednak, że nie polepszymy w ten sposób złożoności Kołmogorowa więcej niż o stałą (zależną od U i U', ale nie od x.

Fact 8. *Niech M będzie dowolną maszyną Turinga i niech*

$$C_M(x) = \min\{|v| : M(v) = x\}$$

. Wtedy istnieje taka stała c_{UM}, że

$$C_U(x) \leq C_M(x) + c_{UM}$$

dla każdego x.

[19] Zob. Ważniak, Języki, automaty i obliczenia, wykład 12 lub https://www.mimuw.edu.pl/~niwinski/Zlozonosc/2016/complex_blog.pdf, section 2.1.

Proof. Niech $\langle M \rangle$ oznacza kod maszyny M. Wtedy, zgodnie z definicją U, $M(v) = U(\langle M \rangle v)$, dla każdego v, dla którego którakolwiek ze stron jest określona. Mamy więc

$$C_U(x) \leq \min\{|\langle M \rangle| + |v| : M(v) = x\} = C_M(x) + |\langle M \rangle|.$$

Wystarczy więc przyjąć $c_{UM} = |\langle M \rangle|$. □

Corollary 6 (niezmienniczość)**.** *Jeśli U, U' są dwiema maszynami uniwersalnymi (być może dla różnych kodowań), to istnieje stała $c_{UU'}$, że*

$$C_U(x) \leq C_{U'}(x) + c_{UU'}$$

dla każdego x.

Corollary 7 (szacowanie)**.** *Istnieje stała c_U, że*

$$C_U(x) \leq |x| + c_U.$$

Proof. W powyższym fakcie wystarczy przyjąć za M maszynę obliczajacą funkcję identycznościową. □

Wybierając dowolną funkcję $x \mapsto v$, gdzie $U(v) = x$ i $|v| = C_U(x)$, otrzymujemy oczywiście notację. Z Faktu na temat notacji wynika, że dla każdego n istnieje słowo x długości n dla którego $C_U(x) \geq |x|$.

Definition 14 (Słowa losowe)**.** *Słowo x spełniające $C_U(x) \geq |x|$ nazywamy* losowym *(w sensie Kołmogorowa).*

Intuicyjnie, w terminach języka Pascal, powiedzielibyśmy, że są to słowa, dla których nie ma istotnie lepszego programu niż `write ('`x`')`.

Wspomniana powyżej notacja $x \mapsto v$ wydaje się być bardzo sensowną propozycją, ma jednak poważną wadę: przy żadnym wyborze tej funkcji nie jest obliczalna. W równoważnym sformułowaniu, mamy następujące

Theorem 12. *Funkcja $x \mapsto C_U(x)$ nie jest obliczalna.*

Proof. Dowód oparty jest na pomyśle paradoksu Berry'ego: dla każdego n znajdujemy słowo w_n, którego „opis wymaga n symboli" i w ten sposób dochodzimy do sprzeczności.

Przypuśćmy, że powyższa funkcja jest obliczalna. Wówczas obliczalna byłaby również funkcja, która binarną reprezentację liczby n (oznaczmy ją bin (n)) przekształca na pierwsze w porządku wojskowym słowo w, takie że $C_U(w) \geq n$ (oznaczmy je w_n; porządek wojskowy porządkuje wszystkie słowa najpierw według długości a potem leksykograficznie). Niech M będzie maszyną realizującą tę funkcję, tzn. $M(\text{bin } (n)) = w_n$. Wtedy oczywiście $C_M(w_n) \leq |\text{bin } (n)|$. Z drugiej strony, zgodnie z udowodnionym przed chwilą Faktem,

$$C_U(w_n) \leq C_M(w_n) + c_{UM}$$

a założyliśmy, że $n \leq C_U(w_n)$, co dałoby nam

$$n \leq |\text{bin } (n)| + c_{UM}$$

dla wszystkich n, co jest oczywiście niemożliwe. □

3.2 Złożoność bezprefiksowa

Wprowadzimy teraz pewne techniczne wymaganie dla maszyn Turinga, które pociągnie za sobą modyfikację pojęcia informacyjnej złożoności algorytmicznej. Ta nowa definicja jest wygodniejsza przede wszystkim przy określeniu "prawdopodobieństwa stopu" i stałej Chaitina, czym zajmiemy się na następnym wykładzie. Otóż, podobnie jak w teorii kodów, wygodnie jest wykluczyć przypadek, kiedy jedno akceptowalne słowo wejściowe jest prefiksem innego takiego słowa. Można argumentowć, że wprowadzenie odpowiedniego ograniczenia na maszyny Turinga jest zgodne z intencją złożoności Kołmogorowa, ponieważ podobna własność jest spełniona w większości języków programowania, gdzie koniec programu (również programu z danymi) jest ściśle określony.

Zbiór słów $L \subseteq \{0,1\}^*$ jest bezprefiksowy, kiedy żadne słowo w L nie jest prefiksem innego słowa w L. Maszynę Turinga M nazwiemy bezprefiksową, o ile język $L(M) = \{w : M(w)\downarrow\}$ jest bezprefiksowy. Z samej postaci maszyny nie jest łatwo wydedukować, czy jest ona bezprefiksowa, czy nie; w istocie nietrudno jest wykazać, że w ogólności jest to problem nierozstrzygalny. Na pierwszy rzut oka jest to istotna przeszkoda dla stworzenia "bezprefiksowej maszyny uniwersalnej". Okazuje się jednak, że dowolną maszynę Turinga można w pewnym sensie efektywnie "poprawić" do maszyny bezprefiksowej i w ten sposób z listy wszystkich maszyn Turinga otrzymać listę maszyn bezprefiksowych $N_0, N_1, \ldots$, tak że każdy częsciowo-obliczalny język bezprefiksowy L znajdzie się na tej liście jako $L = L(N_i)$, dla pewnego i.

W dalszym ciągu, oprócz częsciowego porządku bycia prefiksem (oznaczanego $\leq$) będziemy na zbiorze $\{0,1\}^*$ rozważać porządek liniowy typu ω. Może to być na przykład tzw. porządek wojskowy, który porządkuje słowa najpierw według długości, a słowa tej samej długości leksykograficznie:

$$u \sqsubseteq w \iff u = w \vee |u| < |w| \vee (|u| = |w| \wedge (\exists x, y, y')\, u = x0y \wedge w = x1y'))$$

Fact 9 (Maszyny bezprefiksowe)**.** *Istnieje algorytm, który dla danej maszyny Turinga M znajduje maszynę M' taką, że*
(i) M' jest bezprefiksowa.
(ii) $L(M') \subseteq L(M)$.
(iii) Jeśli $M(x)\downarrow$ i dla każdego $y \neq x$, prefiksowo porównywalnego z x (tzn. $y < x$ lub $x < y$), zachodzi $M(y)\uparrow$, to

$$M'(x) = M(x).$$

W szczególności, jeśli maszyna M jest bezprefiksowa, to

$$L(M') = L(M).$$

Proof. Mając dane M, nasz algorytm konstruuje maszynę M', której działanie opiszemy nieformalnym programem. (Zgodnie z konwencją ustaloną przed definicją 12, instrukcję REJECT należy rozumieć jako wejście w nieskończoną pętlę.)

1. Input (dla M') $= x = x_1 \ldots x_k$.

2. $A := \varepsilon$ (* A będzie przebiegać kolejne prefiksy x *).

3. Przeglądaj wszystkie słowa w $\{0,1\}^*$ w porządku wojskowym $\varepsilon = w_0, w_1, w_2, \ldots, w_i, \ldots$ i "ruchem zygzakowym" symuluj działanie Mna wejściu Aw_i.

   Dokładniej, symulacja przebiega w fazach: $0, 1, \ldots, i, \ldots$ W i-tej fazie wykonuje się kolejny krok w obliczeniach maszyny M na słowach $Aw_0, Aw_1, \ldots, Aw_{i-1}$ oraz pierwszy krok w obliczeniu M na Aw_i.

   Jeśli w czasie tej symulacji stwierdzisz, że $M(Aw_i)\downarrow$, GOTO 4.

4. if $w_i = \varepsilon$ then
   if $A = x$ then STOP (*accept*);
   Output $= M(Aw_i) = M(x)$
   else REJECT
   else
   if $A = x_1 \ldots x_\ell, \ell < k$
   then $A := x_1 \ldots x_\ell x_{\ell+1}$; GOTO 3
   else (* $w_i > \varepsilon \wedge A = x$ *) REJECT

Bezprefiksowość maszyny M' (warunek (i)) wynika z faktu, że obliczenie dla wejścia x zawiera w sobie obliczenie dla wejścia $x' < x$. Gdyby więc zaszły warunki akceptacji dla wejścia x' (tzn. $w_i = \varepsilon i A = x'$), to w przypadku wejścia x nastąpi wyjście przez (pierwsze) REJECT.

Warunek (ii) jest oczywisty, bo jeśli M' daje Output, to $M'(x) = M(x)$.

Wreszcie, jeśli $M(x) \downarrow$, ale nie zachodzi to dla żadnego właściwego prefiksu ani rozszerzenia x, to ruch zygzakowy gwarantuje wykrycie $M(Aw_i) \downarrow$, dla $A = x$ i $w_i = \varepsilon$, a zatem STOP (*accept*) z warunku (iii). □

Definition 15 (Bezprefiksowa uniwersalna maszyna Turinga)**.** Bezprefiksowa uniwersalna maszyna Turinga *jest to dowolna bezprefiksowa maszyna U, spełniająca warunek (2) definicji maszyny uniwersalnej oraz spełniająca waruki (1a,b), o ile v jest kodem pewnej bezprefiksowejmaszyny Turinga M_v.*

(Sens: U jest uniwersalna dla maszyn bezprefiksowych.)

Maszyny takie istnieją:

Fact 10. *Maszyna U' otrzymana z maszyny uniwersalnej U ("zwykłej") przez konstrukcję z Faktu 9, jest bezprefiksową uniwersalną maszyną Turinga.*

Proof. Jedynym nieoczywistym stwierdzeniem jest, że dla bezprefiksowej maszyny, $M_v(x) \downarrow$ pociąga za sobą $U'(vx) \downarrow$. Mamy $U(vx) \downarrow$; wystarczy więc sprawdzić, że zachodzą założenia warunku (iii). Istotnie, dla $vx < y$ lub $v \leq y < vx$, mamy $U(y) \uparrow$, ponieważ M_v jest z założenia bezprefiksowa i $M_v(z) \uparrow \iff U(vz) \uparrow$. Jeśli natomiast $y < v$, to również $U(y) \uparrow$, ponieważ samo kodowanie jest bezprefiksowe (a więc y nie jest wtedy postaci $\langle M \rangle u$, dla żadnej maszyny M, a w takim przypadku U się zapętla.) □

Zakładamy, że Czytelnik wie już, że problem stopu dla uniwersalnej maszyny Turinga jest nierozstrzygalny. W terminach powyższych definicji oznacza to, że nie istnieje maszyna Turinga obliczająca funkcję charakterystyczną zbioru $L(U)$. (Z definicji taka maszyna zatrzymywałaby się dla każdego wejścia x z wynikiem 1, gdy $U(x) \downarrow$ i 0, gdy $U(x) \uparrow$.) Jak można oczekiwać, podobna własność zachodzi dla maszyny bezprefiksowej.

Fact 11. *Problem stopu dla uniwersalnej maszyny bezprefiksowej jest nierozstrzygalny.*

Proof. Wykażemy, że w przeciwnym razie również problem stopu dla uniwersalnej maszyny Turinga bez wymagania bezprefiskowości byłby rozstrzygalny. Rozważmy funkcję na słowach bitowych

$$\alpha : w_1\, w_2 \ldots w_n \mapsto w_1 0\, w_2 0 \ldots w_n 0\, 01.$$

Zauważmy, że dla dowolnego języka $L \subseteq \{0,1\}^*$, język

$$L^\alpha \quad = \quad \{\alpha(w) : w \in L\}$$

jest bezprefiksowy. Dla dowolnej maszyny Turinga M, łatwo jest skonstruować maszynę M^α rozpoznającą język $L(M)^\alpha$; maszyna M^α jest z definicji bezprefiksowa. Niech U^α będzie taką maszyną dla uniwersalanej maszyny Turinga U w sensie Definicji 12. Niech U' będzie maszyną skonstruowaną dla U w Fakcie 9; jak zauważyliśmy powyżej, U' jest uniwersalną bezprefiksową maszyną Turinga. Zatem, dla dowolnego słowa w,

$$\begin{aligned} U(w) \downarrow \quad &\iff \quad U^\alpha(\alpha(w)) \downarrow \\ &\iff \quad U'(\langle U^\alpha \rangle \alpha(w)). \end{aligned}$$

Zatem rozstrzygalność problemu stopu dla U' implikowałaby rozstrzygalność problemu stopu dla U. □

Definition 16 (Bezprefiksowa złożoność Kołmogorowa)**.** Bezprefiksową złożonością informacyjną Kołmogorowa *słowa x jest*

$$K_U(x) = \min\{|v| : U(v) = x\}$$

gdzie U jest bezprefiksową maszyną uniwersalną.

Kiedy wybór maszyny U nie ma znaczenia, będziemy czasem pisać po prostu $K(x)$.

Dowód Faktu gwarantującego niezmienniczość przechodzi bez zmian, podobnie dowód twierdzenia o nieobliczalności.

Fact 12. *Niech M będzie dowolną prefiksową maszyną Turinga i niech*

$$K_M(x) = \min\{|v| : M(v) = x\}.$$

Wtedy istnieje taka stała c_{UM}, że

$$K_U(x) \leq K_M(x) + c_{UM}$$

dla każdego x.

Podobnie jak poprzednio

Fact 13. *Funkcja $x \mapsto K_U(x)$ nie jest obliczalna.*

Natomiast nie mamy dokładnego odpowiednika Wniosku 7, bo maszyna obliczająca identyczność nie jest bezprefiksowa.

Problem 1. *Wykaż, że istnieją stałe $c_{U,1}, c_{U,2}$, takie że*

$$K_U(x) \quad \leq \quad |x| + c_{U,1} \log |x| + c_{U,2}.$$

Podaj jeszcze lepsze oszacowanie.

3.3 Stała Chaitina

Tak jak w poprzednim wykładzie, ustalamy jakieś bezprefiksowe kodowanie maszyn Turinga oraz bezprefiksową maszynę uniwersalną U.

Definition 17 (Stała Chaitina)**.** Stałą Chaitina *określamy jako sumę szeregu*

$$\Omega = \sum_{U(v)\downarrow} 2^{-|v|}$$

Stałą Chaitina można interpretować jako prawdopodobieństwo, że losowo wybrane dane dla maszyny U spowodują jej zatrzymanie; innymi słowy, że losowo wybrany program (z danymi) się zatrzymuje.

Dokładniej, rozważmy zbiór nieskończonych ciągów zero-jedynkowych, $\{0,1\}^\omega$. Dla słowa $w_1 \dots w_n \in \{0,1\}^*$, określamy

$$p(w_1 \dots w_n \{0,1\}^\omega) = \frac{1}{2^n},$$

w szczególności $p(\{0,1\}^\omega) = 1$. Funkcję p można rozszerzyć na Borelowskie podzbiory $\{0,1\}^\omega$ tak, by stanowiła prawdopodobieństwo. Prawdopodobieństwo to możemy też określić patrząc na ciąg $x \in \{0,1\}^\omega$ jak na wynik nieskończonego procesu Bernoulliego $X_1, X_2, \dots$, gdzie $p(X_i = 0) = p(X_i = 1) = \frac{1}{2}$.

W szczególności Ω stanowi prawdopodobieństwo zdarzenia, że ciąg $x \in \{0,1\}^\omega$ zawiera prefiks v, dla którego $U(v)\downarrow$ (z bezprefiksowości wynika, że jest co najwyżej jeden taki prefiks). Oczywiście konkretna wartość Ω zależy od wyboru kodowania i maszyny uniwersalnej, ale jej istotne własności od tego nie zależą.

Theorem 13 (Własności Ω)**.** *Stała Chaitina ma następujące własności.*
(1) $\Omega \leq 1$.
(2) Istnieje maszyna Turinga T z dodatkową taśmą nieskończoną, na której wypisane są kolejne cyfry binarnego rozwinięcia Ω, która dla danego kodu $\langle M \rangle$ maszyny M odpowiada na pytanie, czy $M(\varepsilon)\downarrow$.
(3) Istnieje stała c taka, że

$$K_U(\omega_1 \dots \omega_n) \geq n - c,$$

gdzie $\omega_1 \dots \omega_n$ oznacza pierwszych n bitów liczby Ω.

Punkt (2) oznacza, że "znając" stałą Chaitina potrafilibyśmy rozstrzygać problem stopu, natomiast (3) mówi nam, że z dokładnością do stałej, Ω jest niekompresowalna.

Proof. Ad 1. Ponieważ zbiór

$$L(U) = \{w : U(w) \downarrow\}$$

jest bezprefiksowy, każdy skończony podzbiór $\mathcal{S} \subseteq L(U)$, tworzy kod bezprefiksowy, a zatem z nierówności Krafta spełnia nierówność $\sum_{x \in \mathcal{S}} 2^{-|x|} \leq 1$, co po przejściu do supremum daje żądaną nierówność.

Ad 2. Zanim opiszemy konstrukcję maszyny T, zróbmy pewne obserwacje na temat liczby Ω. Znanym problemem w dowodach własności liczb rzeczywistych jest, że a priori liczba może mieć dwie różne reprezentacje (w szczególności binarne). Działoby się tak, gdyby liczba Ω była dwójkowo wymierna, tzn.
(a) $\Omega = 0.\omega_1\omega_2 \dots \omega_k 0111 \dots$
(b) $\Omega = 0.\omega_1\omega_2 \dots \omega_k 1000 \dots$
Jakkolwiek w przyszłości wykluczymy taką możliwość, w tej chwili musimy jeszcze wziąć ją pod uwagę. Otóż bez zmniejszenia ogólności możemy założyć, że Ω dana jest w postaci (a). Istotnie, gdybyśmy mieli maszynę T dla tego przypadku, to łatwo moglibyśmy ją zmodyfikować do maszyny T', która radziłaby sobie z przypadkiem (b). Maszyna T' działałaby tak samo jak maszyna T, z tym że począwszy od $k+1$-szej cyfry Ω, „widziałaby na odwrót", tzn. 0 traktowałaby jak 1 a 1 jak 0.

Jeśli wybierzemy wariant (a), lub jeśli Ω nie jest dwójkowo wymierna, to dla każdego n, $0.\omega_1\omega_2 \dots \omega_n < \Omega$. A zatem istnieje skończony podzbiór $\mathcal{S} \subseteq L(U)$, taki że liczba wyznaczona przez pierwszych n cyfr Ω spełnia

$$0.\omega_1\omega_2 \dots \omega_n \quad < \quad \sum_{x \in \mathcal{S}} 2^{-|x|}.$$

Oczywiście, jeśli S jest takim zbiorem, to dla każdego $y \in L(U) - S$, $|y| > n$.

Opiszemy teraz działanie maszyny T. Jak zwykle w takich przypadkach, opiszemy algorytm, pozostawiając Czytelnikowi jego formalizację w języku maszyn Turinga. Jeśli na wejściu jest słowo w, $|w| = n$, maszyna T symuluje działanie U na w, a równolegle przegląda kolejne słowa z $v \in \{0,1\}^*$, powiedzmy w porządku wojskowym: $v_0, v_1, v_2, \dots$ (poczynając od $v_0 = \varepsilon$) i symuluje działanie U na v_i ruchem zygzakowym, podobnie jak w algorytmie z dowodu Faktu 9.

W trakcie swojego obliczenia, maszyna T utrzymuje zmienną, powiedzmy $\mathcal{S}'$, której aktualną wartością jest (skończony) zbiór tych słów v, dla których już udało się stwierdzić, że $U(v) \downarrow$.

W skończonym czasie jeden z dwóch przypadków ma miejsce.

(i) T stwierdza, że $U(w) \downarrow$; wtedy daje odpowiedź TAK.

(ii) T stwierdza, że

$$0.\omega_1\omega_2 \dots \omega_n < \sum_{v \in \mathcal{S}'} 2^{-|v|},$$

ale $w \notin \mathcal{S}'$; wtedy daje odpowiedź NIE.

Poprawność algorytmu wynika z obserwacji dokonanej powyżej.

Zauważmy, że w tej chwili możemy już wykluczyć możliwość, że Ω jest liczbą dwójkowo wymierną. Istotnie, Czytelnik pamięta zapewne doskonale, że problem stopu jest nierozstrzygalny, tzn. nie istnieje maszyna bez dodatkowej taśmy, realizująca postulat z warunku (2). Gdyby jednak Ω była dwójkowo wymierna, to opisaną wyżej konstrukcję maszyny T można przeprowadzić bez reprezentowania liczby Ω; zamiast pobierać bity liczby Ω z dodatkowej nieskończonej taśmy, maszyna T mogłaby je sobie łatwo obliczyć. Podobny argument pokazuje znacznie więcej: Ω nie jest liczba wymierną ani algebraiczną, ani w ogole „obliczalną" (zobacz Ćwiczenia w Ważniaku).

Ad 3. Opiszemy działanie pewnej maszyny R. Na słowie wejściowym x, R najpierw symuluje działanie maszyny uniwersalnej U na słowie x. Dalszy opis prowadzimy przy założeniu, że obliczenie się zakończyło z wynikiem $U(x)$ i co więcej

$$U(x) = \omega_1\omega_2\ldots\omega_n,$$

stanowi pierwsze n cyfr rozwinięcia binarnego Ω, dla pewnego n. Niech

$$\Omega_n = \omega_1\omega_2\ldots\omega_n.$$

Oczywiście, dla wielu x nie będzie to prawdą; wtedy maszyna R zgodnie z naszym opisem będzie wykonywać jakieś działania, których wynik nas nie interesuje. Ważne jest jednak, że dla pewnego x istotnie zajdzie $U(x) = \Omega_n$ (z własności maszyny uniwersalnej). Dalsze rozumowanie opiera się na następującej idei: jeśli maszyna U dla wejścia x wygenerowała Ω_n, to maszyna R z tego samego wejścia potrafi wygenerować słowo o złożoności Kołmogorowa $\geq n$; w konsekwencji x musi mieć długość co najmniej n, z dokładnością do stałej.

Powróćmy do opisu działania maszyny R. Po hipotetycznym wygenerowaniu słowa Ω_n, maszyna R, podobnie jak maszyna T w dowodzie punktu (2), ruchem zygzakowym przegląda kolejne słowa y i symuluje działanie U na y, gromadząc w zmiennej $\mathcal{S}'$ te słowa y, dla których obliczenie już się zakończyło. Dodatkowo, dla każdego $y \in \mathcal{S}'$, R zapamiętuje $U(y)$. Pamiętamy, że wykluczyliśmy już możliwość podwójnej reprezentacji Ω. Dlatego też, po pewnym skończonym czasie R stwierdzi, że

$$\sum_{y\in\mathcal{S}'} 2^{-|y|} \geq \Omega_n. \tag{85}$$

Niech v będzie pierwszym w porządku wojskowym słowem takim, że $v \neq U(y)$, dla każdego $y \in \mathcal{S}'$. Zauważmy, że $K_U(v) \geq n$ (z nierówności 85 i definicji Ω). Wtedy wreszcie nasza maszyna R zatrzymuje się z wynikiem $R(x) = v$.

Sprawdźmy dla porządku, że maszyna R jest bezprefiksowa. Istotnie, obliczenie R na wejściu x zakończy się tylko pod warunkiem, że zakończy się jego pierwsza faza, a mianowicie symulacja U na x. Nie ma przy tym znaczenia, czy U wygeneruje ciąg Ω_n, czy coś innego. Ważne, że jeśli $U(x)\downarrow$, to dla każdego x' będącego właściwym prefiksem lub sufiksem x, zachodzi $U(x')\uparrow$; to gwarantuje bezprefiksowość R.

A zatem, zgodnie z Faktem 12, istnieje stała c_{UR}, że

$$K_U(v) \leq K_R(v) + c_{UR}.$$

Ale $K_R(v) \leq |x|$ (skoro R wygenerowała v z wejścia x). To daje nam

$$n \leq K_U(v) \leq K_R(v) + c_{UR} \leq |x| + c_{UR}.$$

Co więcej, nierówność ta zachodzi dla każdego x, takiego że $U(x) = \Omega_n$. A zatem

$$n \leq K_U(\Omega_n) + c_{UR}$$

dla każdego n, tak więc $c = c_{UR}$ może być żądaną stałą. □

3.4 Związek z entropią Shannona

Jeśli stałą Chaitina zinterpretujemy jako prawdopodobieństwo, że bezprefiksowa maszyna uniwersalna U się zatrzymuje, to dla $y \in \{0,1\}^*$,

$$p_U(y) = \sum_{v:U(v)=y} 2^{-|v|}$$

stanowi prawdopodobieństwo zdarzenia, że maszyna U zatrzymuje się z wynikiem y.

Zauważmy, że p_U nie stanowi miary prawdopodobieństwa na $\{0,1\}^*$, w szczególności

$$\sum_{y\in\{0,1\}^*} p_U(y) = \Omega, \text{ a nie } 1.$$

Ale już

$$p(y) = \frac{p_U(y)}{\Omega}$$

wyznacza prawdopodobieństwo na $\{0,1\}^*$.

Jak pamiętamy z wykładu 3, dla skończonej przestrzeni probabilistycznej S, optymalne kodowanie $\varphi : S \to \{0,1\}^*$ było osiągnięte wtedy, gdy

$$|\varphi(y)| \approx -\log_2(p(y))$$

Dokładniej, równość była osiągnięta dla prawdopodobieństw będących potęgami $\frac{1}{2}$, a w ogólności mamy zbieżność asymptotyczną.

Otóż podobny związek możemy wskazać dla bezprefiksowej złożoności Kołmogorowa, która w pewnym sensie wyznacza optymalne kodowanie słów w $\{0,1\}^*$, przy określonym wyżej prawdopodobieństwie p.

Mówiąc nieformalnie, mamy

$$K(y) \approx -\log_2 p(y).$$

Dokładniej, pokażemy następujący

Theorem 14 (Entropia Kołmogorowa)**.** *Istnieje stała c, że dla dowolnego $y \in \{0,1\}^*$,*

$$K(y) - c \leq -\log_2 p(y) \leq K(y) + c.$$

Proof. Oczywiście, wystarczy jeśli pokażemy

$$K(y) - c \leq -\log_2 p_U(y) \leq K(y) + c.$$

Mamy $K(y) = |x|$, dla pewnego x, takiego, że $U(x) = y$, a zatem

$$\frac{1}{2^{|x|}} \leq p_U(y),$$

skąd

$$-\log_2 p_U(y) \leq |x| = K(y).$$

Pozostaje dowieść, że

$$K(y) \leq -\log p_U(y) + c$$

dla pewnej stałej c. Wobec Faktu o niezmienniczości (Fact 12), wystarczy w tym celu skonstruować bezprefiksową maszynę T taką, że dla każdego y istnieje w_y takie, że $T(w_y) = y$ oraz

$$|w_y| \quad \leq \quad -\log p_U(y) + c, \tag{86}$$

gdzie c nie zależy od y.

Zanim zdefiniujemy maszynę T wprowadzimy pojęcie pomocnicze.

Przedziałem binarnym jest z definicji przedział postaci

$$\left[\underbrace{a_1 \cdot \frac{1}{2} + a_2 \cdot \frac{1}{2^2} + \ldots + a_k \cdot \frac{1}{2^k}}_{L}, L + \frac{1}{2^k}\right), \tag{87}$$

gdzie $a_1, \ldots, a_k \in \{0,1\}$; $a_k = 1$. Założenie $a_k = 1$ służy temu, by L jednoznacznie wyznaczało przedział binarny. Na zbiorze przedziałów zdefiniujemy funkcję L, która dla danego przedziału zwraca lewy koniec największego przedziału binarnego w nim zawartego. Gdyby było więcej przedziałów o tej samej długości, L zwraca ten, którego początek jest położony najbardziej na lewo.

Opiszemy teraz działanie maszyny T uruchomionej z wejściem p. Ustawmy wszystkie słowa $z \in \{0,1\}^*$ w porządku wojskowym $z_0, z_1, \ldots$. Używając ruchu zygzakowatego, T symuluje maszynę U na danych $z_0, z_1, \ldots$. Niech $Z_{t,y}$ będzie zbiorem z takich, że $U(z) = y$ oraz $U(z)\downarrow$ w co najwyżej t krokach maszyny T. Podczas obliczeń pamiętamy wielkość

$$\varphi(t,y) = \sum_{z \in Z_{t,y}} \frac{1}{2^{|z|}}.$$

Możemy ją zapamiętać, bo dla każdego t mamy $\varphi(t,y) \neq 0$ tylko dla skończenie wielu y. Oczywiście $\varphi(t,y)$ rośnie wraz z t oraz

$$\lim_{t\to\infty} \varphi(t,y) = p_U(y). \tag{88}$$

Definiujemy funkcję pomocniczą

$$\psi(t,y) = \max\{\frac{1}{2^k} : \frac{1}{2^k} \leq \varphi(t,y)\}.$$

Zauważmy, że funkcja ψ aproksymuje funkcję φ z dokładnością $\frac{1}{2}$, tzn. $\psi(t,y) > \frac{1}{2}\varphi(t,y)$.

Ilekroć w jakiejś chwili t funkcja $\psi(t,y)$ zwiększa swoją wartość odkładamy na odcinku $[0,1]$ przedział $I_{t,y}$ długości $\frac{1}{2}\psi(t,y)$. Dla konkretności zakładamy, że jest to przedział postaci $[\ell, r)$, tzn. zawiera lewy koniec, a nie zawiera prawego. Przedziały odkładamy sąsiadująco, zaczynając od 0. Ponieważ dla każdego a

$$\sum_{\frac{1}{2^k} \leq a} \frac{1}{2^k} \leq 2a,$$

suma długości wszystkich przedziałów odłożonych dla danego y nie przekroczy $p_U(y)$, a suma długości wszystkich przedziałów nie przekroczy 1. Po odłożeniu $I_{t,y}$, znajdujemy dla tego przedziału wartość funkcji $L(I_{t,y})$ (por. formułę (87) powyżej). Jeżeli $L(I_{t,y}) = p$ to maszyna T kończy pracę zwracając y.

Maszyna T jest bezprefiksowa, gdyż zbiór początków parami rozłącznych przedziałów binarnych jest bezprefiksowy.

Jak widać, dla każdego y istnieje p takie, że maszyna skończy pracę i $T(p) = y$. Takich p może być wiele. Pozostaje wskazać wśród nich słowo w_y spełniające warunek (86).

Skoro funkcja $\varphi(t,y)$ zbliża się od dołu do $p_U(y)$ (por. (88)), istnieje t, dla którego $\varphi(t,y) \geq \frac{1}{2}p_U(y)$, a zatem $\psi(t,y) \geq \frac{1}{4}p_U(y)$ i w pewnym kroku obliczenia zostanie odłożony przedział $I_{t,y}$ długości co najmniej $\frac{1}{8}p_U(y)$. $L(I_{t,y})$ będzie naszym szukanym w_y.

Ponieważ (co za chwilę wykażemy) dla przedziału długości $\frac{1}{2^k}$ długość najdłuższego przedziału binarnego w nim zawartego wynosi przynajmniej $\frac{1}{2^{k+2}}$ mamy

$$\frac{1}{2^{|w_y|}} \geq \frac{1}{4}|I_{t,y}| \geq \frac{1}{32}p_U(y)$$

i stąd

$$|w_y| \leq -\log p_U(y) + 5,$$

co czyni zadość warunkowi (86).

Oszacujemy teraz długość przedziału binarnego. Niech A będzie lewym końcem odcinka o długości $\frac{1}{2^k}$. Do odcinka należy liczba postaci $\frac{n}{2^k}$, gdzie $n \in \mathbb{N}$. Przedział binarny wyznaczamy w następujący sposób:

- Jeśli $A = \frac{n}{2^k}$, to $L = \frac{n}{2^k}$ lub $L = \frac{n}{2^k} + \frac{1}{2^{k+1}}$.
- Jeśli $A + \frac{1}{2^{k+1}} \leq \frac{n}{2^k}$, to $L = \frac{n}{2^k} - \frac{1}{2^{k+1}}$.
- Jeśli $A + \frac{1}{2^{k+1}} > \frac{n}{2^k}$, to $L = \frac{n}{2^k} + \frac{1}{2^{k+2}}$ lub $L = \frac{n}{2^k} - \frac{1}{2^{k+2}}$.

□

3.5 Losowość

Losowość jest brakiem regularności. Jednocześnie brak regularności oznacza niekompresowalność. W statystyce losowość próbki określa się za pomocą testów (np. test serii). Testy są to procedury, które dla zadanej próbki i istotności decydują, czy próbka jest losowa. Wykażemy, że niekompresowalność implikuje spełnianie wszystkich efektywnych testów losowości.

Mamy zbiór próbek $S = \{0,1\}^*$ i miarę probabilistyczną p na S. Chcemy podzielić S na elementy, które posiadają pewną wyróżniającą je cechę oraz takie, które jej nie posiadają. Intuicyjnie $x \in S$ jest losowy, gdy nie posiada żadnych wyróżniających go cech.

Przez $L(x) = 2^{-2|x|-1}$ oznaczymy miarę jednostajną na S. Niech $L_n(x) = L(x \big| |x| = n)$, zachodzi $L_n(x) = 2^{-n}$, gdy $|x| = n$ i $L_n(x) = 0$ wpp.

Przykładowo, załóżmy, że $p = L$ i niech badaną cechą ciągu będzie prefiks złożony z samych zer. Oznaczmy przez V_m zbiór wszystkich takich ciągów, które mają prefiks 0^m. Mamy $V_0 \supseteq V_1 \supseteq V_2$ oraz

$$\sum_{x \in V_m} p(x \big| |x| = n) = \frac{|\{x \in \{0,1\}^n : x \in V_m\}|}{2^n} = \frac{2^{n-m}}{2^n} = \frac{1}{2^m}.$$

Jak widać wraz ze wzrostem m cecha wyróżniająca staje się coraz silniejsza, a jednocześnie maleje prawdopodobieństwo wylosowania obiektu mającego taką cechę.

Prawdopodobieństwo wylosowania ciągu x zaczynającego się od m zer wynosi $\frac{1}{2^m}$, zatem z *istotnością* $\frac{1}{2^m}$ możemy odrzucić hipotezę, że x został wylosowany. W ogólności

Definition 18. *Niech $V_0 \supseteq V_1 \supseteq V_2$. V_m jest* regionem krytycznym *z istotnością $\frac{1}{2^m}$ jeśli dla każdego n*

$$\sum_{x \in V_m} p(x \big| |x| = n) \leq \frac{1}{2^m}.$$

Jeśli $x \in V_m$ to z istotnością $\frac{1}{2^m}$ nie jest losowy.

Niech $\delta : S \to \mathbb{N}$ będzie taka, że $V_m = \{x : \delta(x) \geq m\}$.

Definition 19. *$\delta : S \to \mathbb{N}$ jest* p-testem, *gdy dla każdego m*

$$\sum_{x : \delta(x) \geq m} p(x \big| |x| = n) \leq \frac{1}{2^m}.$$

Definition 20. *Test δ nazwiemy* efektywnym *jeśli zbiór*

$$V = \{(m, x) : \delta(x) \geq m\}$$

jest rekurencyjnie przeliczalny, tzn. jeśli istnieje maszyna M_δ generująca (skończony lub nie) ciąg $(m,x) \in V$ i każdy element V jest przez nią generowany.

Definition 21. Uniwersalny test Martina-Löfa ze względu na rozkład p (uniwerslany p-test) *jest to test $\delta_0(\cdot|p)$ taki, że dla każdego efektywnego p-testu δ istnieje stała c taka, że dla każdego $x \in S$*

$$\delta_0(x|p) \geq \delta(x) - c.$$

Niech $U_m = \{x : \delta_0(x|p) \geq m\}$ i $V_m = \{x : \delta(x) \geq m\}$ wtedy

$$V_{m+c} \subseteq U_m.$$

Theorem 15. *Istnieje uniwersalny p-test.*

Proof. Na początek skonstruujemy maszynę T, która wylicza wartości testu, a potem udowodnimy, że jest on uniwersalny.

Bierzemy maszynę uniwersalną U. T symuluje $U(k)$ współbieżnie dla wszystkich $k \in \mathbb{N}$. Jeśli k nie jest kodem maszyny Turinga, $U(k)$ pętli się. W przeciwnym przypadku, jeśli generowany przez $U(k)$ wynik nie jest ciągiem par to T kończy symulację $U(k)$. Jeżeli $U(k)$ wypisze w trakcie symulacji na taśmie parę (m, x) to T sprawdza, czy wygenerowany dotychczas przez $U(k)$ zbiór par spełnia warunki regionów krytycznych, jeśli nie to T kończy symulację $U(k)$. W przeciwnym wypadku T wypisuje na wyjściu $(m-k, x), (m-k-1, x), \ldots, (0, x)$.

Niech δ_k będzie testem wyliczanym przez $U(k)$. Wtedy

$$\delta_0(x|p) = \max\{\delta_k(x) - k : k \geq 1\}.$$

Zatem $\delta_0(\cdot|p)$ spełnia warunek bycia testem uniwersalnym.

Pozostaje jeszcze oszacować wielkość regionów krytycznych.

$$\sum_{x:\delta_0(x|p)\geq m} p(x\big||x|=n) \leq \sum_{k=1}^{\infty} \sum_{x:\delta_k(x)\geq m+k} p(x\big||x|=n) \leq \sum_{k=1}^{\infty} \frac{1}{2^{m+k}} = \frac{1}{2^m}.$$

□

Definition 22. *Niech U będzie uniwersalną maszyną Turinga, a $[\cdot,\cdot]$ funkcją pary.* Warunkowa złożoność Kołmogorowa *jest to*

$$C_U(x|y) = \min\{|z| : U([y, z]) = x\}.$$

Warunkowa złożoność Kołmogorowa z dokładnością do stałej jest niezależna od wyboru maszyny uniwersalnej.

Theorem 16. *Funkcja*

$$\delta_0(x|p) = |x| - C(x\big||x|) - 1$$

jest uniwersalnym testem dla rozkładu jednostajnego L.

Proof. Najpierw zauważmy, że zbiór $\{(m, x) : \delta_0(x|p) \geq m\}$ jest rekurencyjnie przeliczalny. Generuje go bowiem maszyna T, która symuluje współbieżnie maszynę uniwersalną U na wszystkich parach $y, z \in \mathbb{N}$. Gdy $U(z) = x$, to jeśli $|x| \neq y$ ignorujemy wynik. Wpp. mamy $C(x\big||x|) \leq |z|$. Zatem $\delta_0(x|p) \geq |x| - |z| - 1$, czyli T może wypisać na wyjściu $(|x| - |z| - 1, x)$.

Sprawdzamy warunek regionów krytycznych. Ustalamy $|x|$. Liczba x-ów takich, że $C(x\big||x|) \leq |x| - m - 1$ nie może być większa niż liczba programów długości co najwyżej $|x| - m - 1$, więc

$$|\{x : |x| - C(x\big||x|) - 1 \geq m\}| \leq 2^{|x|-m-1}$$

Pokażemy, że dla każdego testu δ istnieje stała c taka, że $\delta_0(x|p) \geq \delta(x)$. Ustalmy uniwersalną maszynę U, niech $\langle M_\delta \rangle$ będzie kodem maszyny generującej δ. Udowodnimy, że $C(x\big||x|) \leq |x| - \delta(x) + 2|\langle M_\delta \rangle|$. Wtedy

$$\delta_0(x, p) = |x| - C(x\big||x|) - 1 \geq \delta(x) - (\underbrace{2|\langle M_\delta \rangle| + 1}_{c}).$$

Ustalmy x. Niech

$$A = \{z : \delta(z) \geq \delta(x), |z| = |x|\}.$$

Zauważmy, że $x \in A$ oraz z własności regionów krytycznych $\delta(x) \leq |x|$ i $|A| \leq 2^{|x|-\delta(x)}$. Mając $|x|$ i $\delta(x)$ możemy wyliczyć przy pomocy M_δ elementy A. Niech s będzie numerem x w tym wyliczeniu, $|\mathrm{bin}(s)| \leq |x| - \delta(x)$. Konstuujemy ciąg

$$t = \underbrace{000\ldots\ldots\ldots0}_{|x|-\delta(x)-|\mathrm{bin}(s)|}\mathrm{bin}(s).$$

Dostajemy maszynę, która znając wcześniej $|x|$ na podstawie t oraz $\langle M_\delta \rangle$ wylicza $\delta(x) = |x| - |t|$ a następnie x. $\square$

References

[1] Thomas M. Cover, and Joy A. Thomas. *Elements of Information Theory*. John Wiley & Sons, 1991, second edition: 2006.

[2] Gareth A. Jones and J. Mary Jones. *Information and Coding Theory*, Springer 2000.

[3] John Talbot and Dominic Welsh. *Complexity and Cryptography*, Cambridge University Press, 2006.

[4] Peter Winkler. *Mathematical Puzzles: A Connoisseur's Collection*. A K Peters, 2004.

[5] Ming Li and Paul Vitanyi. *An Introduction to Kolmogorov Complexity and Its Applications*, Springer, 2008.