Information Theory and Statistics
Lecture 7: Prefix-free Kolmogorov complexity

Łukasz Dębowski
ldebowsk@ipipan.waw.pl

Ph. D. Programme 2013/2014

Project co-financed by the European Union
within the framework of the European Social Fund

Towards algorithmic information theory

- In Shannon's information theory, the amount of information carried by a random variable depends on the ascribed probability distribution.
- Andrey Kolmogorov (1903–1987), the founding father of modern probability theory, remarked that it is extremely hard to imagine a reasonable probability distribution for utterances in natural language but we can still estimate their information content.
- For that reason Kolmogorov proposed to define the information content of a particular string in a purely algorithmic way.
- A similar approach has been proposed a year earlier by Ray Solomonoff (1926–2009), who sought for optimal inductive inference.

Project co-financed by the European Union
within the framework of the European Social Fund
KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI
PhD STUDIES
UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Kolmogorov complexity—analogue of Shannon entropy

- The fundamental object of algorithmic information theory is Kolmogorov complexity of a string, an analogue of Shannon entropy.
- It is defined as the length of the shortest program for a Turing machine such that the machine prints out the string and halts.
- The Turing machine itself is defined as a deterministic finite state automaton which moves along one or more infinite tapes filled with symbols from a fixed finite alphabet and which may read and write individual symbols.
- The concrete value of Kolmogorov complexity depends on the used Turing machine but many properties of Kolmogorov complexity are universal.

Project co-financed by the European Union
within the framework of the European Social Fund

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

PhD STUDIES

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Prefix-free Kolmogorov complexity

- A particular definition of Kolmogorov complexity, called prefix-free Kolmogorov complexity, is convenient to discuss links between Kolmogorov complexity and entropy.
- The fundamental idea is to force that the accepted programs form a prefix-free set.
- Here we will use the construction by Gregory Chaitin (1947–), introduced in 1975.

Project co-financed by the European Union
within the framework of the European Social Fund

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

PhD STUDIES

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Prefix-free Turing machine

We will consider a machine that has two tapes:

1. a bidirectional tape $(\mathbf{X_i})_{i\in\mathbb{Z}}$ filled with symbols $\mathbf{0}$, $\mathbf{1}$, and $\mathbf{B}$,
2. a unidirectional tape $(\mathbf{Y_k})_{k\in\mathbb{N}}$ filled with symbols $\mathbf{0}$ and $\mathbf{1}$.

The machine head moves in both directions along tape $(\mathbf{X_i})_{i\in\mathbb{Z}}$ and only in the direction of growing $\mathbf{k}$ along tape $(\mathbf{Y_k})_{k\in\mathbb{N}}$.
Tape $(\mathbf{X_i})_{i\in\mathbb{Z}}$ is both read and written, tape $(\mathbf{Y_k})_{k\in\mathbb{N}}$ is read only.

Project co-financed by the European Union
within the framework of the European Social Fund
KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI
PhD STUDIES
UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Formal definition

Formally, a Turing machine is defined as a **6**-tuple $\mathbf{T} = (\mathbf{Q}, \mathbf{s}, \mathbf{h}, \mathbf{\Gamma}, \mathbf{B}, \delta)$, where

1. $\mathbf{Q}$ is a finite, nonempty set of states,
2. $\mathbf{s} \in \mathbf{Q}$ is the start state,
3. $\mathbf{h} \in \mathbf{Q}$ is the halt state,
4. $\mathbf{\Gamma} = \{\mathbf{0}, \mathbf{1}\}$ is a finite, nonempty set of symbols,
5. $\mathbf{B} \in \mathbf{\Gamma}$ is the blank symbol,
6. $\delta : \mathbf{Q} \setminus \{\mathbf{h}\} \times \mathbf{\Gamma} \times \mathbf{\Gamma} \cup \{\mathbf{B}\} \to \mathbf{Q} \times \{\mathbf{0}, \mathbf{R}\} \times \mathbf{\Gamma} \cup \{\mathbf{B}\} \times \{\mathbf{L}, \mathbf{0}, \mathbf{R}\}$ is a function called a transition function.

The set of such machines is denoted $\mathcal{S}$.

Project co-financed by the European Union
within the framework of the European Social Fund

From formal description to machine operation

The machine operation is as follows:

- Initially, the machine is in state $\mathbf{s}$ and reads $\mathbf{Y_1}$ and $\mathbf{X_1}$.
- Subsequently, the machine shifts in discrete steps along the tape in the way prescribed by the transition function:

  If the machine is in state $\mathbf{a}$ and reads symbols $\mathbf{y}$ and $\mathbf{x}$ then, for $\delta(\mathbf{a}, \mathbf{y}, \mathbf{x}) = (\mathbf{a'}, \mathbf{M_Y}, \mathbf{x'}, \mathbf{M_X})$, the machine:

  - moves along tape $(\mathbf{Y_i})_{\mathbf{i}\in\mathbb{Z}}$ one symbol to the right or does not move if $\mathbf{M_Y = R}$ or $\mathbf{M_Y = 0}$,
  - writes symbol $\mathbf{x'}$ on tape $(\mathbf{X_i})_{\mathbf{i}\in\mathbb{Z}}$,
  - moves along tape $(\mathbf{X_i})_{\mathbf{i}\in\mathbb{Z}}$ one symbol to the left, to the right or does not move if $\mathbf{M_X = L}$, $\mathbf{M_X = R}$, or $\mathbf{M_X = 0}$,
  - assumes state $\mathbf{a'}$ in the next step.
- This procedure takes place until the machine reaches the halt state $\mathbf{h}$. Then the computation stops.

Project co-financed by the European Union
within the framework of the European Social Fund

Halting on an input

We say that machine $\mathbf{S} \in \mathcal{S}$ halts on input $\mathbf{(p, q)} \in \Gamma^* \times (\Gamma^* \cup \Gamma^{\mathbb{N}})$ and returns $\mathbf{w} \in \Gamma^*$ if:

(A) The head in the start state reads symbols $\mathbf{Y_1}$ and $\mathbf{X_1}$, and the initial state of the tapes is $\mathbf{Y_1^{|p|} = p}$ and $\mathbf{X_1^{|q|+1} = qB}$ or $\mathbf{X_1^{\infty} = q}$ if $\mathbf{q}$ is an infinite sequence. Besides we put $\mathbf{X_i = 0}$ for $\mathbf{i < 1}$ and $\mathbf{i > |q| + 1}$ if $\mathbf{q}$ is finite.

(B) The head in the halt state reads symbols $\mathbf{Y_{|p|}}$ and $\mathbf{X_j}$, and the final state of the tape $\mathbf{(X_i)_{i \in \mathbb{Z}}}$ is $\mathbf{X_j^{|w|+j} = wB}$.

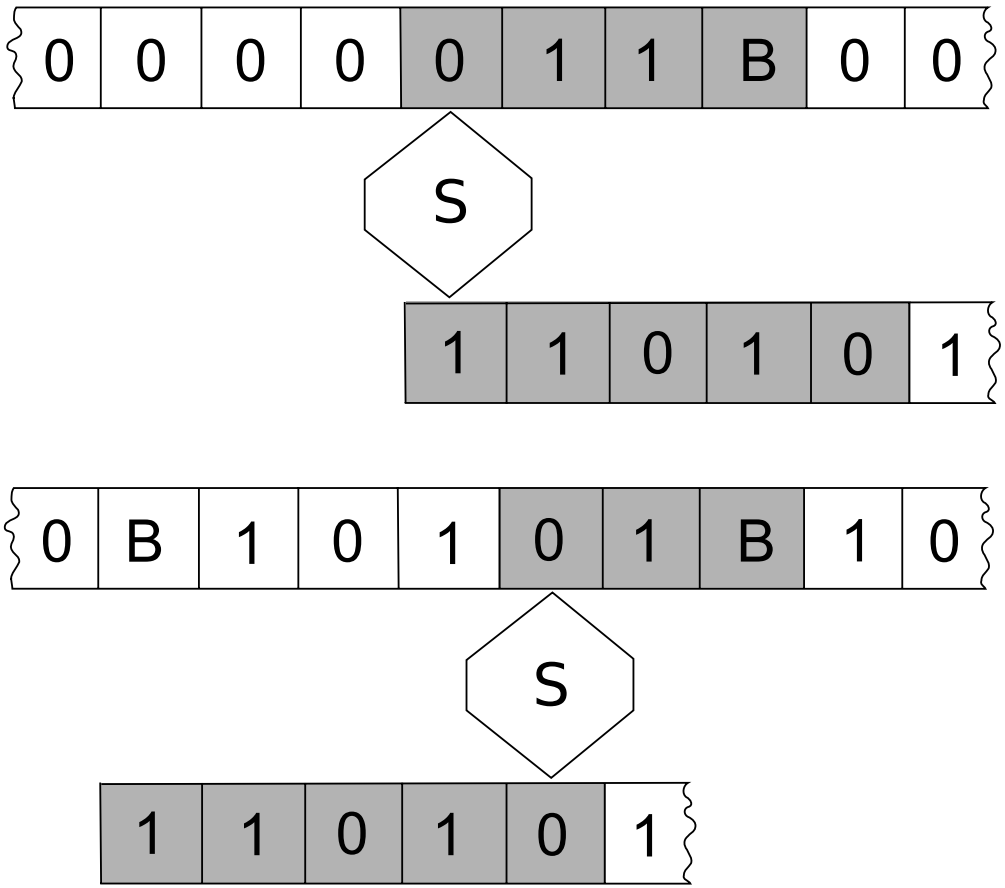
Assuming that condition (A) is satisfied, we write

$$\mathbf{S(p|q)} = \begin{cases} \mathbf{w}, & \text{if condition (B) is satisfied,} \\ \mathbf{\infty}, & \text{if condition (B) is not satisfied for any } \mathbf{w} \in \Gamma^*. \end{cases}$$

It can be easily seen that for a given $\mathbf{q}$ the set of strings $\mathbf{p}$ such that machine $\mathbf{S}$ halts on input $\mathbf{(p, q)}$ is prefix-free. Such strings are called *self-delimiting programs*.

Project co-financed by the European Union
within the framework of the European Social Fund

Prefix-free Turing machine (II)



The initial and the final state of machine **S** for $\mathbf{S(11010|011) = 01}$.

Project co-financed by the European Union
within the framework of the European Social Fund

Prefix-free complexity

Definition (prefix-free complexity)

Prefix-free conditional Kolmogorov complexity $\mathbf{K_S(w|q)}$ of a string $\mathbf{w} \in \mathbf{\Gamma^*}$ given string $\mathbf{q} \in \mathbf{\Gamma^*}$ and machine $\mathbf{S} \in \boldsymbol{\mathcal{S}}$ is defined as

$$\mathbf{K_S(w|q)} := \min_{\mathbf{p} \in \mathbf{\Gamma^*}} \{|\mathbf{p}| : \mathbf{S(p|q)} = \mathbf{w}\} .$$

Project co-financed by the European Union
within the framework of the European Social Fund

Universal machines

Definition (universal machine)

Machine $\mathbf{V} \in \boldsymbol{\mathcal{S}}$ is called *universal* if for each machine $\mathbf{S} \in \boldsymbol{\mathcal{S}}$ there exists a string $\mathbf{u} \in \boldsymbol{\Gamma}^*$ such that

$$\mathbf{V(up|q) = S(p|q)}$$

for all $\mathbf{p} \in \boldsymbol{\Gamma}^*$ and $\mathbf{q} \in \boldsymbol{\Gamma}^* \cup \boldsymbol{\Gamma}^{\mathbb{N}}$.

Universal machines exist. The proof is simple but tedious.

Project co-financed by the European Union
within the framework of the European Social Fund

Invariance theorem

Theorem (invariance theorem)

For any two universal machines $\mathbf{V}$ *and* $\mathbf{V'}$ *there exists a constant* $\mathbf{c}$ *such that*

$$|\mathbf{K_V(w|q)} - \mathbf{K_{V'}(w|q)}| \leq \mathbf{c}$$

for any $\mathbf{w} \in \mathbf{\Gamma}^*$ *and* $\mathbf{q} \in \mathbf{\Gamma}^* \cup \mathbf{\Gamma}^{\mathbb{N}}$.

Proof

We have $\mathbf{V(up|q) = V'(p|q)}$ and $\mathbf{V'(u'p|q) = V(p)}$ for certain strings $\mathbf{u}$ and $\mathbf{u'}$. Hence $\mathbf{K_V(w|q) \leq K_{V'}(w|q) + |u|}$ and $\mathbf{K_{V'}(w|q) \leq K_V(w|q) + |u'|}$.

Project co-financed by the European Union
within the framework of the European Social Fund
KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI
PhD STUDIES
UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Prefix-free complexity

Definition (prefix-free complexity II)

Let $\mathbf{V} \in \boldsymbol{\mathcal{S}}$ be a certain universal machine. We put

$$\mathbf{K(w|q) := K_V(w|q).}$$

This quantity is called *prefix-free conditional Kolmogorov complexity*.

Project co-financed by the European Union
within the framework of the European Social Fund

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

PhD STUDIES

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Unconditional complexity

Unconditional complexities are defined as

$$\mathbf{K_S(w) := K_S(w|\lambda)},$$
$$\mathbf{K(w) := K(w|\lambda)},$$

where $\boldsymbol{\lambda}$ is the empty string.

Similarly, we use $\mathbf{V(p) := V(p|\lambda)}$ for any other function $\mathbf{V}$.

Project co-financed by the European Union
within the framework of the European Social Fund

Towards complexity of arbitrary objects

- We want to discuss of objects such as numbers or tuples of numbers and sequences.
- We will define the complexity of such object as the complexity of the corresponding sequence, where the sequence is given by a fixed encoding of the object.

Project co-financed by the European Union
within the framework of the European Social Fund

Complexities of objects defined

- Let $\mathbb{A}$ be a set of discrete objects, such as $\mathbb{A} = \mathbb{Q}^*$, and let $\phi : \mathbb{A} \to \Gamma^*$ be some bijection.
- Additionally, we put $\phi(\infty) := \infty$, where ∞ denotes the indefinite value.
- Let $\mathbb{B}$ be a set of not necessarily discrete objects, such as $\mathbb{B} = \mathbb{R}^{\mathbb{N}}$, and let $\psi : \mathbb{B} \to \Gamma^* \cup \Gamma^{\mathbb{N}}$ be some bijection.
- Additionally, we put $\psi(\infty) := \infty$.
- For a machine $S \in \mathcal{S}$, $a \in \mathbb{A}$, and $b \in \mathbb{B}$, we define

$$S(a|b) := S(\phi(a)|\psi(b)).$$

- The Kolmogorov complexity of objects will be defined as

$$K_S(a|b) := K_S(\phi(a)|\psi(b)),$$
$$K(a|b) := K(\phi(a)|\psi(b)).$$

Project co-financed by the European Union
within the framework of the European Social Fund

Computable discrete functions

The Kolmogorov complexity of a discrete partial function $\mathbf{f} : \mathbb{B} \to \mathbb{A} \cup \{\infty\}$ is defined as

$$\mathbf{K(f)} := \min_{\mathbf{p} \in \Gamma^*} \{|\mathbf{p}| : \forall_{\mathbf{u} \in \mathbb{B}} \mathbf{V(p|u)} = \mathbf{f(u)}\} .$$

Definition (computable discrete function)

A function $\mathbf{f} : \mathbb{B} \to \mathbb{A} \cup \{\infty\}$ is called *computable* if $\mathbf{K(f)} < \infty$.

Project co-financed by the European Union
within the framework of the European Social Fund

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

PhD STUDIES

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Semi-computable real functions

Definition (lower semicomputable real function)

A function $\mathbf{f} : \mathbb{B} \to \mathbb{R}$ is called *lower semicomputable* if there is a computable function $\mathbf{A} : \mathbb{B} \times \mathbb{N} \to \mathbb{Q}$ which satisfies $\mathbf{A(x, k+1)} \geq \mathbf{A(x, k)}$ and

$$\forall_{x \in \mathbb{B}} \lim_{k \to \infty} \mathbf{A(x, k)} = \mathbf{f(x)}.$$

Definition (upper semicomputable real function)

A function $\mathbf{f} : \mathbb{B} \to \mathbb{R}$ is called *upper semicomputable* if there is a computable function $\mathbf{A} : \mathbb{B} \times \mathbb{N} \to \mathbb{Q}$ which satisfies $\mathbf{A(x, k+1)} \leq \mathbf{A(x, k)}$ and

$$\forall_{x \in \mathbb{B}} \lim_{k \to \infty} \mathbf{A(x, k)} = \mathbf{f(x)}.$$

Project co-financed by the European Union
within the framework of the European Social Fund

Computable real functions

Definition (computable real function)

A function $\mathbf{f} : \mathbb{B} \to \mathbb{R}$ is called *computable* if there is a computable function $\mathbf{A} : \mathbb{B} \times \mathbb{N} \to \mathbb{Q}$ which satisfies

$$\forall_{x \in \mathbb{B}} \, |f(x) - A(x, k)| < 1/k.$$

We put

$$K(f) := \min_{A} K(A).$$

Project co-financed by the European Union
within the framework of the European Social Fund

Further notations

We will write $\mathbf{p(x)} \overset{+}{<} \mathbf{q(x)}$ and $\mathbf{p(x)} \overset{+}{>} \mathbf{q(x)}$ if there exists a constant $\mathbf{c}$ such that $\mathbf{p(x)} \leq \mathbf{q(x)} + \mathbf{c}$ and $\mathbf{p(x)} \geq \mathbf{q(x)} + \mathbf{c}$ holds respectively for all $\mathbf{x}$. We will write $\mathbf{p(x)} \overset{+}{=} \mathbf{q(x)}$ when we have both $\mathbf{p(x)} \overset{+}{<} \mathbf{q(x)}$ and $\mathbf{p(x)} \overset{+}{>} \mathbf{q(x)}$.

Project co-financed by the European Union
within the framework of the European Social Fund

A bound for Kolmogorov complexity

Theorem

$$\mathbf{K(w|q)} \stackrel{+}{<} \mathbf{|w| + 2\log|w|}\,.$$

Proof

There is a prefix-free machine **S** which satisfies $\mathbf{S(bw|q) = w}$, where **b** is a prefix-free binary code for $\mathbf{|w|}$. The length of this code can be made as small as $\mathbf{2\log|w|}$. The machine works as follows. First it reads the binary codeword **b** from the unidirectional tape. Then it copies string **w** to the bidirectional tape. After copying the last symbol of **w** the machine halts because it knows its length from reading codeword **b**. Hence we have the desired bound.

Project co-financed by the European Union
within the framework of the European Social Fund

Further inequalities (I)

Theorem

$$\mathbf{K(w|u)} \stackrel{+}{<} \mathbf{K(w)} \stackrel{+}{<} \mathbf{K(\langle u, w\rangle)}.$$

Proof

$\mathbf{K(w|u)} \stackrel{+}{<} \mathbf{K(w)}$ because a certain program that computes $\mathbf{w}$ given $\mathbf{u}$ has form "ignore $\mathbf{u}$ and execute the shortest program that computes $\mathbf{w}$".

$\mathbf{K(w)} \stackrel{+}{<} \mathbf{K(\langle u, w\rangle)}$ because a certain program that computes $\mathbf{w}$ has form "execute the shortest program that computes $\langle \mathbf{u}, \mathbf{w}\rangle$ and compute $\mathbf{w}$ from $\langle \mathbf{u}, \mathbf{w}\rangle$".

Project co-financed by the European Union
within the framework of the European Social Fund
KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI
PhD STUDIES
UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Further inequalities (II)

Theorem

$$\mathbf{K(uw)} \overset{+}{<} \mathbf{K(\langle u, w\rangle)} \overset{+}{<} \mathbf{K(u) + K(w|u)} \overset{+}{<} \mathbf{K(u) + K(w)}.$$

Proof

$\mathbf{K(uw)} \overset{+}{<} \mathbf{K(\langle u, w\rangle)}$ because a certain program that computes $\mathbf{uw}$ has form "execute the shortest program that computes $\langle \mathbf{u}, \mathbf{w}\rangle$ and compute $\mathbf{uw}$ from $\langle \mathbf{u}, \mathbf{w}\rangle$".

Similarly, $\mathbf{K(\langle u, w\rangle)} \overset{+}{<} \mathbf{K(u) + K(w|u)}$ because a certain program that computes $\langle \mathbf{u}, \mathbf{w}\rangle$ has form "execute the shortest program that computes $\mathbf{u}$ and the shortest program that computes $\mathbf{w}$ given $\mathbf{u}$ and from that compute $\langle \mathbf{u}, \mathbf{w}\rangle$".

The last inequality follows from $\mathbf{K(w|u)} \overset{+}{<} \mathbf{K(w)}$.

Project co-financed by the European Union
within the framework of the European Social Fund

Further inequalities (III)

Theorem

$$\mathbf{K(f(w)) \overset{+}{<} K(w) + K(f).}$$

Proof

The inequality follows from the fact that a certain program that computes $\mathbf{f(w)}$ has form "execute the shortest program that computes $\mathbf{w}$ and to the result apply the program that computes $\mathbf{f(w)}$ given $\mathbf{w}$".

Project co-financed by the European Union
within the framework of the European Social Fund

Incompressible strings

A string x_1^n will be called c-*incompressible* if $K(x_1^n) \geq n - c$.
There are infinitely many incompressible strings.

Theorem

There are at least $2^n - 2^{n-c} + 1$ *distinct* c*-incompressible strings of length* n.

Proof

There are at most $2^{n-c} - 1$ programs of length smaller than $n - c$ and there are 2^n strings of length n. Subtracting the latter from the former, we obtain the desired bound.

In particular, there is at least one 0-incompressible string of length n and at least a half of strings of length n is 1-incompressible.

Project co-financed by the European Union
within the framework of the European Social Fund
KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI
PhD STUDIES
UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Halting probability

Lemma

Consider halting probability

$$\Omega = \sum_{\mathbf{p}:\, \mathbf{V}(\mathbf{p}) \neq \infty} 2^{-|\mathbf{p}|}.$$

Let $\Omega_1^{\mathbf{n}}$ *be the first* $\mathbf{n}$ *digits of* Ω *and let* $\mathbf{p}$ *be a string of a length smaller than* $\mathbf{n}$. *Given* $\Omega_1^{\mathbf{n}}$ *we may decide whether machine* $\mathbf{V}$ *stops on input* $\mathbf{p}$.

Project co-financed by the European Union
within the framework of the European Social Fund

Proof of the lemma

We have $\mathbf{0.\Omega_1^n \leq \Omega < 0.\Omega_1^n + 2^{-n}}$. Let us simulate the computation of machine $\mathbf{V}$ on all inputs shorter than $\mathbf{n}$. Namely, in the $\mathbf{i}$-th step we execute the $\mathbf{j}$-th step of computations for all $\mathbf{k}$-th inputs which satisfy $\mathbf{j + k = i}$. In the beginning of the simulations, we set the approximation of $\mathbf{\Omega}$ as $\mathbf{\Omega' := 0}$. When $\mathbf{V}$ halts for a certain input $\mathbf{p}$, we improve the approximation by setting $\mathbf{\Omega' := \Omega' + 2^{-|p|}}$. At a certain instant, $\mathbf{\Omega'}$ becomes equal or greater than $\mathbf{0.\Omega_1^n}$. Then it becomes clear that machine $\mathbf{V}$ will not halt on any other input shorter than $\mathbf{n}$ and we may decide on the halting problem.

Project co-financed by the European Union
within the framework of the European Social Fund

Ω is incompressible

Theorem

We have $\mathbf{K}(\Omega_1^{\mathbf{n}}) \overset{+}{>} \mathbf{n}$.

Proof

By the previous lemma, we can enumerate, given $\Omega_1^{\mathbf{n}}$, all programs shorter than $\mathbf{n}$ for which machine $\mathbf{V}$ halts. For any $\mathbf{w}$ which is not computed by these programs we have $\mathbf{K(w)} > \mathbf{n}$. We can construct a computable function ϕ which computes one of these $\mathbf{w}$ given $\Omega_1^{\mathbf{n}}$. Hence $\mathbf{K}(\phi(\Omega_1^{\mathbf{n}})) \geq \mathbf{n}$, which implies $\mathbf{K}(\Omega_1^{\mathbf{n}}) \geq \mathbf{n} - \mathbf{c}$ for a certain $\mathbf{c}$.

Project co-financed by the European Union
within the framework of the European Social Fund
KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI
PhD STUDIES
UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Incomputability of Kolmogorov complexity

Theorem

Kolmogorov complexity $\mathbf{K(\cdot)}$ *is not a computable function.*

Proof

Assume that there exists a program $\mathbf{q}$ which computes $\mathbf{K(w)}$ for any $\mathbf{w}$. Then there exists a program $\mathbf{p}$ which uses $\mathbf{q}$ as a subroutine to print out the shortest string $\mathbf{w}$ such that $\mathbf{K(w)} > |\mathbf{p}|$. Namely, such a program $\mathbf{p}$ inspects strings $\mathbf{w}$ sorted according to their length, computes $\mathbf{K(w)}$ using subroutine $\mathbf{q}$ and checks whether $\mathbf{K(w)} > |\mathbf{p}|$. It is obvious that this inequality will hold for a certain $\mathbf{w}$ because $\mathbf{K(w)}$ is unbounded. But by the definition of Kolmogorov complexity, we have $\mathbf{K(w)} \leq |\mathbf{p}|$ for the same string. Hence our assumption about the existence of program $\mathbf{q}$ was false.

Project co-financed by the European Union
within the framework of the European Social Fund
KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI
PhD STUDIES
UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Towards information-theoretic Gödel theorem

- A similar search for the shortest element appears in the proof of the information-theoretic Gödel theorem, another fundamental result in the algorithmic information theory.
- A formal inference system is a finite collection of axioms and inference rules. The system is called *consistent* if it is not possible to prove both a statement and its negation, whereas the system is called *sound* if only true propositions can be proved. (Thus a sound system is consistent.)
- According to the information-theoretic Gödel theorem, in any sound formal inference system it is not possible to prove incompressibility of any string which is substantially longer than the definition of this formal system.

Project co-financed by the European Union
within the framework of the European Social Fund

Information-theoretic Gödel theorem

Theorem (information-theoretic Gödel theorem)

For a sound formal inference system, there exists a constant $\mathbf{M}$ *such that propositions "*$\mathbf{K(w) > M}$*" are unprovable in the system.*

Proof

Suppose that for any number $\mathbf{M}$ there exists a proof of proposition "$\mathbf{K(w) > M}$". Then we may construct a program of length of $\mathbf{L}$ which searches all proofs of the formal system to find the first proof that a certain string $\mathbf{w}$ has the complexity greater than $\mathbf{M}$ and then prints out that string. Then we have $\mathbf{K(w) \leq L}$. Since $\mathbf{L} \overset{+}{<} \mathbf{2 \log M}$, we obtain contradiction for sufficiently large $\mathbf{M}$.

Project co-financed by the European Union
within the framework of the European Social Fund
KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI
PhD STUDIES
UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY