Introduction to Kolmogorov Complexity

Marcus Hutter
Canberra, ACT, 0200, Australia
http://www.hutter1.net/

ANU

Abstract

In this talk I will give a brief introduction to the field of algorithmic information theory (AIT), its underlying philosophy, and the most important concepts. AIT arises by mixing information theory and computation theory to obtain an objective and absolute notion of information in an individual object, and in so doing gives rise to an objective and robust notion of randomness of individual objects. This is in contrast to classical information theory that is based on random variables and communication, and has no bearing on information and randomness of individual objects. Next week I'll present some recent applications.

Contents

Summary of Shannon Entropy

Let $X, Y \in \mathcal{X}$ be discrete random variable with distribution $P(X,Y)$.

Definition 1 (Definition of Shannon entropy)

$$\text{Entropy}(X) \equiv H(X) := -\sum_{x\in\mathcal{X}} P(x)\log P(x)$$
$$\text{Entropy}(X|Y) \equiv H(X|Y) := -\sum_{y\in\mathcal{Y}} P(y)\sum_{x\in\mathcal{X}} P(x|y)\log P(x|y)$$

Theorem 2 (Properties of Shannon entropy)

- Upper bound: $H(X) \leq \log|\mathcal{X}| = n$ for $\mathcal{X} = \{0,1\}^n$
- Extra information: $H(X|Y) \leq H(X) \leq H(X,Y)$
- Subadditivity: $H(X,Y) \leq H(X)+H(Y)$
- Symmetry: $H(X|Y)+H(Y) = H(X,Y) = H(Y,X)$
- Information non-increase: $H(f(X)) \leq H(X)$ for any f

Relations for Kolmogorov Complexity will formally look very similar.

Prefix Sets & Codes

String x is (proper) prefix of y $\quad :\Longleftrightarrow \quad$ $\exists\, z(\neq \epsilon)$ such that $xz = y$.

Set $\mathcal{P}$ is prefix-free or a prefix code $\quad :\Longleftrightarrow \quad$ no element is a proper prefix of another.

Example: A self-delimiting code (e.g. $\mathcal{P} = \{0, 10, 11\}$) is prefix-free.

Kraft Inequality

Theorem 3 (Kraft Inequality)
For a binary prefix code $\mathcal{P}$ we have $\sum_{x\in\mathcal{P}} 2^{-\ell(x)} \leq 1$.
Conversely, let $\ell_1, \ell_2, ...$ be a countable sequence of natural numbers such that Kraft's inequality $\sum_k 2^{-\ell_k} \leq 1$ is satisfied. Then there exists a prefix code $\mathcal{P}$ with these lengths of its binary code.

Proof of the Kraft-Inequality

Proof $\Rightarrow$: Assign to each $x \in \mathcal{P}$ the interval $\Gamma_x := [0.x, 0.x + 2^{-\ell(x)})$.

Length of interval Γ_x is $2^{-\ell(x)}$.

Intervals are disjoint, since $\mathcal{P}$ is prefix free, hence

$$\sum_{x \in \mathcal{P}} 2^{-\ell(x)} = \sum_{x \in \mathcal{P}} \text{Length}(\Gamma_x) \leq \text{Length}([0,1]) = 1$$

Proof idea $\Leftarrow$:

Choose l_1, l_2, ... in increasing order.

Successively chop off intervals of lengths 2^{-l_1}, 2^{-l_2}, ...

from left to right from $[0,1)$ and

define left interval boundary as code.

Identify Numbers and Binary (Prefix) Strings

$x \in \mathbb{N}_0$	0	1	2	3	4	5	6	7	$\cdots$
$x \in \{0,1\}^*$	ϵ	0	1	00	01	10	11	000	$\cdots$
$\ell(x)$	0	1	1	2	2	2	2	3	$\cdots$
$\bar{x} = 1^{\ell(x)}0x$	0	100	101	11000	11001	11010	11011	1110000	$\cdots$
$x' = \overline{\ell(x)}x$	0	100 0	100 1	101 00	101 01	101 10	101 11	11000 000	$\cdots$

- $\mathcal{P} = \{\bar{x} : x \in \{0,1\}^*\}$ is prefix code with $\ell(\bar{x}) = 2\ell(x)+1 \sim 2\log x$
- $\mathcal{P} = \{x' : x \in \{0,1\}^*\}$ forms an asymptotically shorter prefix code with $\ell(x') = \ell(x) + 2\ell(\ell(x)) + 1 \;\sim\; \log x + 2\log\log x$
- Allows to pair strings x and y (and z) by $\langle x,y\rangle := x'y$ (and $\langle x,y,z\rangle := x'y'z$). Uniquely decodable, since x' and y' are prefix.
- Since ' serves as a separator we also write $f(x,y)$ instead of $f(x'y)$
- Notation: $f(x) \stackrel{+}{<} g(x)$ means $f(x) \leq g(x) + O(1)$

Turing Machines & Effective Enumeration

- Turing machine (TM) = (mathematical model for an) idealized computer.
- Instruction i: If symbol on tape under head is 0/1, write 0/1/- and move head left/right/not and goto instruction j.
- $\{$partial recursive functions $\} \equiv \{$functions computable with a TM$\}$.
- A set of objects $S = \{o_1, o_2, o_3, ...\}$ can be (effectively) enumerated $:\Longleftrightarrow \exists$ TM machine mapping i to $\langle o_i \rangle$,
  where $\langle\rangle$ is some (often omitted) default coding of elements in S.

Sharpened Church-Turing Theses

TMs and p.r. functions are important due to ...

Thesis 4 (Church-Turing) The class of algorithmically computable numerical functions (in the intuitive sense) coincides with the class of Turing computable = partial recursive functions.

Thesis 5 (Short compiler) Given two natural Turing-equivalent formal systems $F1$ and $F2$, then there always exists a single short program on $F2$ which is capable of interpreting all $F1$-programs.

Lisp, Forth, C, Universal TM, ... have mutually short interpreters.

Prefix Turing Machine

For technical reasons we need the following variants of a Turing machine

Definition 6 (Prefix Turing machine T (pTM))

- one unidirectional read-only input tape,
- one unidirectional write-only output tape,
- some bidirectional work tapes, initially filled with zeros.
- all tapes are binary (no blank symbol!),
- T halts on input p with output $x :\Longleftrightarrow T(p)=x$
  $:\Longleftrightarrow$ p is to the left of the input head
  and x is to the left of the output head after T halts.
- $\{p : T(p)=x\}$ forms a prefix code.
- We call such codes p self-delimiting programs.

Monotone Turing Machine

For technical reasons we need the following variants of a Turing machine

Definition 7 (Monotone Turing machine T (mTM))

- one unidirectional read-only input tape,
- one unidirectional write-only output tape,
- some bidirectional work tapes, initially filled with zeros.
- all tapes are binary (no blank symbol!),
- T outputs/computes a string starting with x (or a sequence ω) on input $p :\Longleftrightarrow T(p) = x*$ (or $T(p) = \omega$) $:\Longleftrightarrow$ p is to the left of the input head when the last bit of x is output.
- T may continue operation and need not to halt.
- For given x, $\{p : T(p) = x*\}$ forms a prefix code.
- We call such codes p minimal programs.

Universal Prefix/Monotone Turing Machine

$\langle T \rangle :=$ some canonical binary coding of (table of rules) of TM T

$\Rightarrow$ set of TMs $\{T_1, T_2, ...\}$ can be effectively enumerated $\Rightarrow \exists U ...$

Theorem 8 (Universal prefix/monotone Turing machine U)
which simulates (any) pTM/mTM T_i with input $y‘q$ if fed with input $y‘i‘q$, i.e.

$$U(y‘i‘q) = T_i(y‘q) \, \forall i, q$$

For $p \neq y‘i‘q$, $U(p)$ does not halt. y is side information.

Theorem 9 (Halting Problem. That's the price we pay for $\exists U$)
There is no TM T: $\quad T(i‘p) = 1 \Longleftrightarrow T_i(p)$ does not halt.

Formalization of Simplicity/Complexity

- Intuition: A string is simple if it can be described in a few words, like "the string of one million ones",
- and is complex if there is no such short description, like for a random string whose shortest description is specifying it bit by bit.
- Effective descriptions or codes $\Rightarrow$ Turing machines as decoders.
- p is description/code of x on pTM $T :\Longleftrightarrow T(p)=x$.
- Length of shortest description: $K_T(x) := \min_p\{\ell(p) : T(p)=x\}$.
- This complexity measure depends on T :-(

Universality/Minimality of K_U

Is there a TM which leads to shortest codes among all TMs for all x?

Remarkably, there exists a Turing machine (the universal one) which "nearly" has this property:

Theorem 10 (Universality/Minimality of K_U)
$K_U(x) \leq K_T(x) + c_{TU}$,
where $c_{TU} \stackrel{+}{<} K_U(T) < \infty$ is independent of x

Pair of UTMs U' and U'': $|K_{U'}(x) - K_{U''}(x)| \leq c_{U'U''}$.

Thesis 5 holds $\Longleftrightarrow$ $c_{U'U''}$ small for natural UTMs U' and U''.

Henceforth we write $O(1)$ for terms like $c_{U'U''}$.

Proof of Universality of K_U

Proof idea: If p is the shortest description of x under $T = T_i$, then i‘p is a description of x under U.

Formal proof:

Let p be shortest description of x under T, i.e. $\ell(p) = K_T(x)$.

$\exists i : T = T_i$

$\Rightarrow U(i\text{‘}p) = x$

$\Rightarrow K_U(x) \leq \ell(i\text{‘}p) = \ell(p) + c_{TU}$ with $c_{TU} := \ell(i\text{‘})$.

Refined proof:

$p := \arg\min_p\{\ell(p) : T(p) = x\}$ = shortest description of x under T

$r := \arg\min_p\{\ell(p) : U(p) = \langle T\rangle\}$ = shortest description of T under U

$q :=$ decode r and simulate T on p.

$\Rightarrow U(qrp) = T(p) = x \Rightarrow$

$K_U(x) \leq \ell(qrp) \stackrel{+}{=} \ell(p) + \ell(r) = K_T(x) + K_U(\langle T\rangle)$.

(Conditional) Prefix Kolmogorov Complexity

Definition 11 ((conditional) prefix Kolmogorov complexity)
= shortest program p, for which reference U outputs x (given y):

$$K(x) := \min_p\{\ell(p) : U(p) = x\},$$

$$K(x|y) := \min_p\{\ell(p) : U(y‘p) = x\}$$

For (non-string) objects: $K(\text{object}) := K(\langle\text{object}\rangle)$,

e.g. $K(x,y) = K(\langle x,y\rangle) = K(x‘y)$.

Upper Bound on K

Theorem 12 (Upper Bound on K)

$$K(x) \stackrel{+}{<} \ell(x) + 2\log\ell(x), \qquad K(n) \stackrel{+}{<} \log n + 2\log\log n$$

Proof:

There exists a TM T_{i_0} with $i_0 = O(1)$ and $T_{i_0}(\epsilon\text{`}x\text{`}) = x$,

then $U(\epsilon\text{`}i_0\text{`}x\text{`}) = x$,

hence $K(x) \leq \ell(\epsilon\text{`}i_0\text{`}x\text{`}) \stackrel{+}{=} \ell(x\text{`}) \stackrel{+}{<} \ell(x) + 2\log\ell(x)$. ■

Lower Bound on K / Kraft Inequality

Theorem 13 (lower bound for most n, Kraft inequality)

$$\sum_{x\in\{0,1\}^*} 2^{-K(x)} \leq 1, \quad K(x) \geq l(x) \quad \text{for 'most'} \quad x$$

$$K(n) \to \infty \quad \text{for} \quad n \to \infty.$$

This is just Kraft's inequality which implies a lower bound on K valid for 'most' n.

'most' means that there are only $o(N)$ exceptions for $n \in \{1, ..., N\}$.

Extra Information & Subadditivity

Theorem 14 (Extra Information)

$$K(x|y) \stackrel{+}{<} K(x) \stackrel{+}{<} K(x,y)$$

Providing side information y can never increase code length,

Requiring extra information y can never decrease code length.

Proof: Similarly to Theorem 12

Theorem 15 (Subadditivity)

$$K(xy) \stackrel{+}{<} K(x,y) \stackrel{+}{<} K(x)+K(y|x) \stackrel{+}{<} K(x)+K(y)$$

Coding x and y separately never helps.

Proof: Similarly to Theorem 14

Symmetry of Information

Theorem 16 (Symmetry of Information)

$$K(x|y,K(y))+K(y) \stackrel{+}{=} K(x,y) \stackrel{+}{=} K(y,x) \stackrel{+}{=} K(y|x,K(x))+K(x)$$

Is the analogue of the logarithm of the multiplication rule for conditional probabilities (see later).

Proof: $\geq = \leq$ similarly to Theorem 15.

For $\leq = \geq$, deep result: see [LV08, Th.3.9.1]. ■

Proof Sketch of $K(y|x)+K(x)\leq K(x,y)+O(\log)$

all $+O(\log)$ terms will be suppressed and ignored. Counting argument:

(1) Assume $K(y|x) > K(x,y) - K(x)$.

(2) $(x,y) \in A := \{\langle u,z\rangle : K(u,z) \leq k\}, \quad k := K(x,y) = O(\log)$

(3) $y \in A_x := \{z : K(x,z) \leq k\}$

(4) Use index of y in A_x to describe y: $K(y|x) \leq \log|A_x|$

(5) $\log|A_x| > K(x,y) - K(x) =: l = O(\log)$ by (1) and (4)

(6) $x \in U := \{u : \log|A_u| > l\}$ by (5)

(7) $\{\langle u,z\rangle : u \in U, z \in A_u\} \subseteq A$

(8) $\log|A| \leq k$ by (2), since at most 2^k codes of length $\leq k$

(9) $2^l|U| < \min\{|A_u| : u \in U\}|U| \leq |A| \leq 2^k$ by (6),(7),(8), resp.

(10) $K(x) \leq \log|U| < k - l = K(x)$ by (6) and (9). Contradiction! ■

Information Non-Increase

Theorem 17 (Information Non-Increase)

$$K(f(x)) \stackrel{+}{<} K(x)+K(f) \quad \text{for recursive} \quad f:\{0,1\}^*\to\{0,1\}^*$$

Definition: The Kolmogorov complexity $K(f)$ of a function f is defined as the length of the shortest self-delimiting program on a prefix TM computing this function.

Interpretation: Transforming x does not increase its information content.

Hence: Switching from one coding scheme to another by means of a recursive bijection leaves K unchanged within additive $O(1)$ terms.

Proof: Similarly to Theorem 15

Coding Relative to Probability Distribution, Minimal Description Length (MDL) Bound

Theorem 18 (Probability coding / MDL)

$$K(x) \stackrel{+}{<} -\log P(x) + K(P)$$
$$\text{if} \quad P : \{0,1\}^* \to [0,1] \text{ is enumerable and } \sum_x P(x) \leq 1$$

This is at the heart of the MDL principle [Ris89], which approximates $K(x)$ by $-\log P(x) + K(P)$.

Proof of MDL Bound

Proof for computable P:

Idea: Use the Shannon-Fano code based on probability distribution P.

Let $s_x := \lceil -\log_2 P(x) \rceil \in \mathbb{N}$

$\Rightarrow \sum_x 2^{-s_x} \leq \sum_x P(x) \leq 1$.

$\Rightarrow$: $\exists$ prefix code p for x with $\ell(p) = s_x$ (by Kraft inequality)

Since the proof of Kraft inequality is (can be made) constructive and P is computable, there exists an effective prefix code in the sense, that $\exists$ TM T : $\forall x \exists p : T(p) = x$ and $\ell(p) = s_x$.

$\Rightarrow K(x) \overset{+}{<} K_T(x) + K(T) \leq s_x + K(T) \overset{+}{<} -\log P(x) + K(P)$

where we used Theorem 10. ■

General Proof Ideas

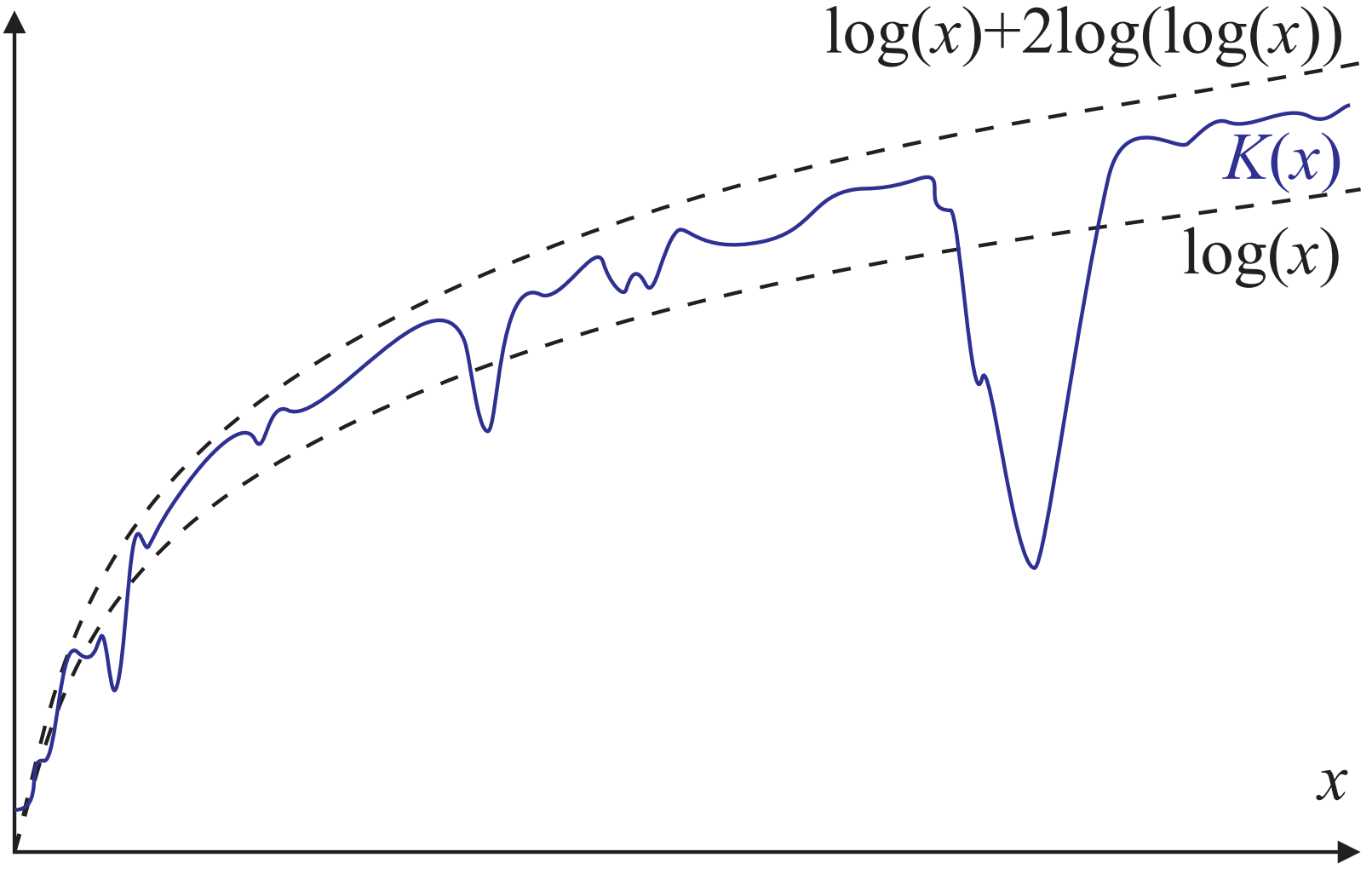
- All upper bounds on $K(z)$ are easily proven by devising **some** (effective) code for z of the length of the right-hand side of the inequality and by noting that $K(z)$ is the length of the shortest code among all possible effective codes.

- Lower bounds are usually proven by counting arguments (Easy for Thm.13 by using Thm.3 and hard for Thm.16)

- The number of short codes is limited.
  More precisely: The number of prefix codes of length $\leq \ell$ is bounded by 2^ℓ.

Remarks on Theorems 12-18

All (in)equalities remain valid if K is (further) conditioned under some z, i.e. $K(...) \rightsquigarrow K(...|z)$ and $K(...|y) \rightsquigarrow K(...|y,z)$.



Relation to Shannon Entropy

Let $X, Y \in \mathcal{X}$ be discrete random variable with distribution $P(X,Y)$.

Definition 19 (Definition of Shannon entropy)

$$\text{Entropy}(X) \equiv H(X) := -\sum_{x\in\mathcal{X}} P(x)\log P(x)$$
$$\text{Entropy}(X|Y) \equiv H(X|Y) := -\sum_{y\in\mathcal{Y}} P(y)\sum_{x\in\mathcal{X}} P(x|y)\log P(x|y)$$

Theorem 20 (Properties of Shannon entropy)

- Upper bound: $H(X) \leq \log|\mathcal{X}| = n$ for $\mathcal{X} = \{0,1\}^n$
- Extra information: $H(X|Y) \leq H(X) \leq H(X,Y)$
- Subadditivity: $H(X,Y) \leq H(X)+H(Y)$
- Symmetry: $H(X|Y)+H(Y) = H(X,Y) = H(Y,X)$
- Information non-increase: $H(f(X)) \leq H(X)$ for any f

Relations for H are essentially expected versions of relations for K.

Monotone Kolmogorov Complexity Km

A variant of K is the monotone complexity $Km(x)$ defined as the shortest program on a monotone TM computing a string starting with x:

Theorem 21 (Monotone Kolmogorov Complexity Km)

$$Km(x) := \min_p\{\ell(p) : U(p) = x*\}$$

has the following properties:

- $Km(x) \stackrel{+}{<} \ell(x)$,
- $Km(xy) \geq Km(x) \in \mathbb{N}_0$,
- $Km(x) \stackrel{+}{<} -\log \mu(x) + K(\mu)$ if μ comp. measure (defined later).

It is natural to call an infinite sequence ω computable if $Km(\omega) < \infty$.

Computable Functions $f : \mathbb{N} \to \mathbb{R}$

f is (finitely) computable or recursive *iff* there are Turing machines $T_{1/2}$ with output interpreted as natural numbers and $f(x) = \frac{T_1(x)}{T_2(x)}$,

$$\Downarrow$$

f is estimable or computable *iff* $\exists$ recursive $\phi(\cdot,\cdot)$ $\forall$ $\varepsilon > 0$: $|\phi(x, \lfloor \frac{1}{\varepsilon} \rfloor) - f(x)| < \varepsilon$ $\forall x$.

$$\Downarrow$$

f is lower semicomputable or enumerable *iff* $\phi(\cdot,\cdot)$ is recursive and $\lim_{t\to\infty} \phi(x,t) = f(x)$ and $\phi(x,t) \leq \phi(x,t+1)$.

$$\Downarrow$$

f is approximable or limit-computable *iff* $\phi(\cdot,\cdot)$ is recursive and $\lim_{t\to\infty} \phi(x,t) = f(x)$.

(Non)Computability of K and Km complexity

Theorem 22 ((Non)computability of K and Km Complexity)
The prefix complexity $K : \{0,1\}^* \to \mathbb{N}$ and the monotone complexity $Km : \{0,1\}^* \to \mathbb{N}$ are co-enumerable, but not finitely computable.

Proof: Assume K is computable.

$\Rightarrow f(m) := \min\{n : K(n) \geq m\}$ exists by Theorem 13 and is computable (and unbounded).

$K(f(m)) \geq m$ by definition of f.

$K(f(m)) \leq K(m) + K(f) \stackrel{+}{<} 2\log m$ by Theorem 17 and 12.

$\Rightarrow m \leq \log m + c$ for some c, but this is false for sufficiently large m.

Co-enumerability of K as exercise. ■

Kolmogorov Complexity vs Shannon Entropy

Shannon Entropy H:

+ computable
+ relations in Thm.20 are exact
– only about expected information
– requires true sampling distribution

Kolmogorov Complexity K:

+ information of individual strings
+ no sampling distribution required
+ captures all effective regularities
– incomputable
– additive slack in most relations
– depends on choice of UTM U

Summary

- A quantitative theory of information has been developed.
- All enumerable objects are coded=identified as strings.
- Codes need to be prefix free, satisfying Kraft's inequality.
- Augment Church-Turing thesis with the short compiler assumption.
- Kolmogorov complexity quantifies information,

  and is **the** complexity measure.
- Major drawback: K is only semicomputable.

Applications of AIT

- Philosophy: problem of induction
- Machine learning: time-series forecasting
- Artificial intelligence: foundations
- Probability theory: choice of priors
- Information theory: individual randomness/information
- Data mining: clustering, measuring similarity
- Bioinformatics: phylogeny tree reconstruction
- Linguistics: language tree reconstruction

- Computer science: string matching, complexity/formal-language/automata theory
- Math: ∞ primes, quantitative Goedel incompleteness
- Physics: Boltzmann entropy, Maxwell daemon, reversible computing
- Operations research: universal search
- Others: Music, cognitive psychology, OCR

Literature

[LV08] M. Li and P. M. B. Vitányi. *An Introduction to Kolmogorov Complexity and its Applications*. Springer, Berlin, 3rd edition, 2008.

[MH07] M. Hutter. *Algorithmic Information Theory*. Scholarpedia, 2:3 (2007) 2519 [an extended encyclopedic entry]

[MH07] M. Hutter. *Kolmogorov Complexity*. Scholarpedia, 3:1 (2008) 2573 [an extended encyclopedic entry]

[Hut04] M. Hutter. *Universal Artificial Intelligence: Sequential Decisions based on Algorithmic Probability*. Springer, Berlin, 2005.

[Cal02] C. S. Calude. *Information and Randomness: An Algorithmic Perspective*. Springer, Berlin, 2nd edition, 2002.