Title: Introduction to Algorithmic Information Theory and Tutorial

Date: Apr 11, 2018 11:30 AM

URL: http://pirsa.org/18040109

Abstract: The progression of theories suggested for our world, from ego- to geo- to helio-centric models to universe and multiverse theories and beyond, shows one tendency: The size of the described worlds increases, with humans being expelled from their center to ever more remote and random locations. If pushed too far, a potential theory of everything (TOE) is actually more a theories of nothing (TON). Indeed such theories have already been developed. I show that including observer localization into such theories is necessary and sufficient to avoid this problem. I develop a quantitative recipe to identify TOEs and distinguish them from TONs and theories in-between. This precisely shows what the problem is with some recently suggested universal TOEs.

Introduction to Kolmogorov Complexity

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

ANU

Contents

- Summary of Shannon Entropy
- Prefix Codes and Kraft Inequality
- (Universal) Prefix/Monotone Turing Machines
- Sharpened Church-Turing Theses
- Kolmogorov Complexity
- Computability Issues
- Relation to Shannon Entropy

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 $:\Longleftrightarrow$ $\exists\, z(\neq \epsilon)$ such that $xz = y$.

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

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

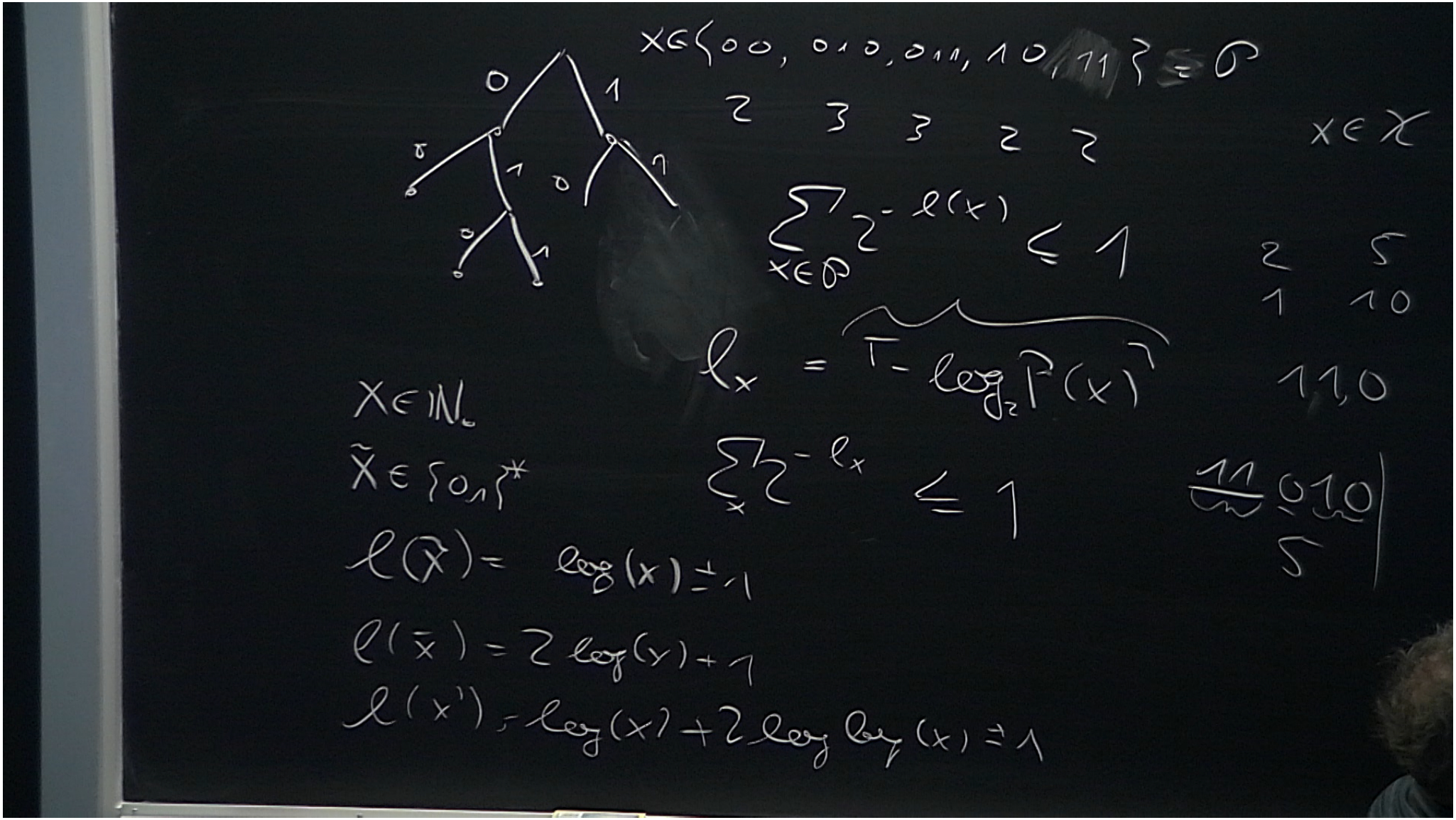
Kraft Inequality

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.

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 $f(x) \leq g(x) + O(1)$


$X \in \{00, 010, 011, 10, 11\} = \mathcal{C}$
2 3 3 2 2
$x \in X$
$\sum_{x \in \mathcal{C}} 2^{-\ell(x)} \leq 1$
2 5
1 10
$\ell_x = \lceil -\log_2 P(x) \rceil$
11,0
$X \in \mathbb{N}$
$\tilde{X} \in \{0,1\}^*$
$\sum_x 2^{-\ell_x} \leq 1$
11 010
5
$\ell(\tilde{x}) = \log(x) + 1$
$\ell(\bar{x}) = 2\log(x) + 1$
$\ell(x') = \log(x) + 2\log\log(x) + 1$

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))

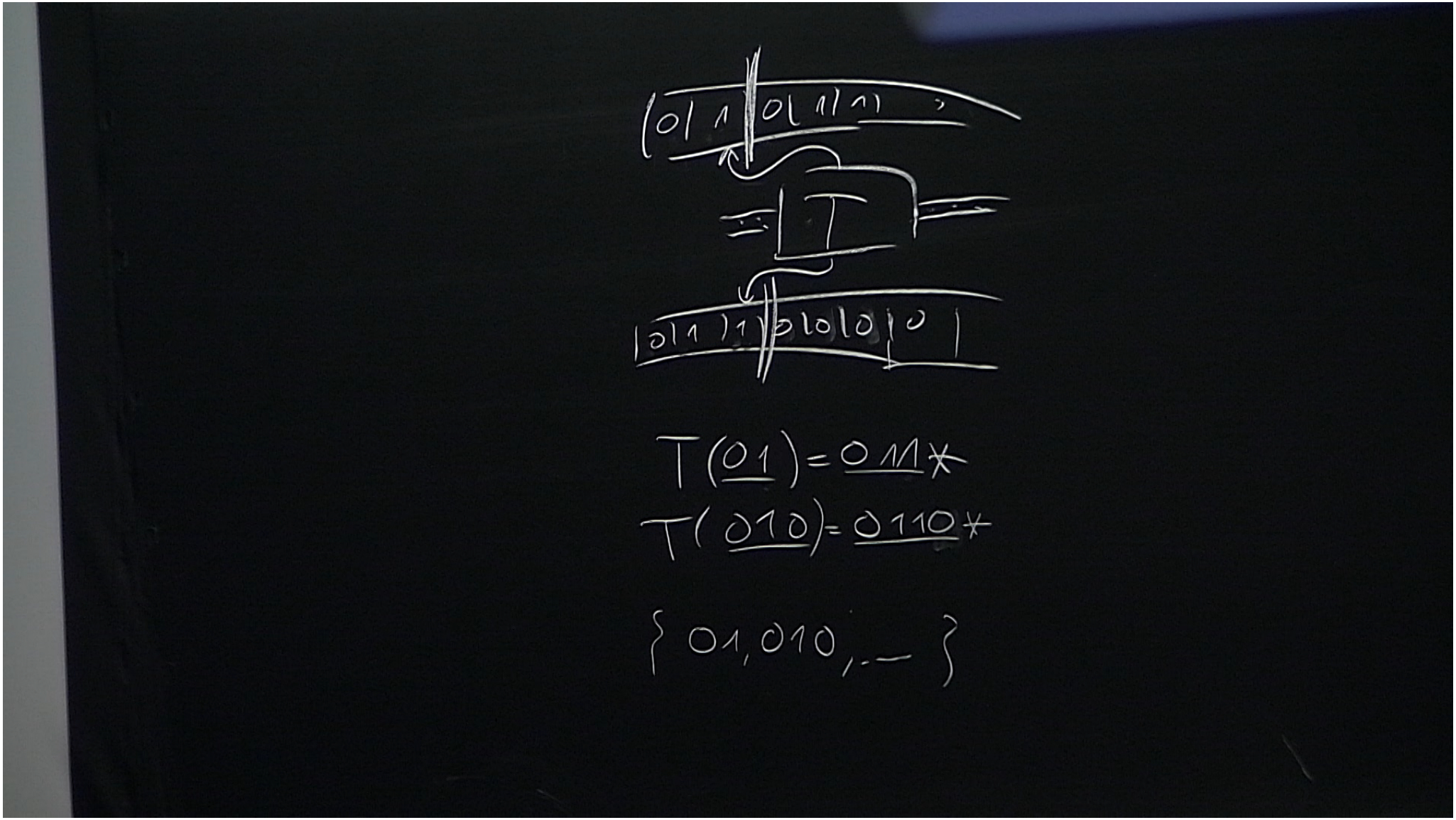
- one unidirectional read-only input tape,
- one unidirectional write-only output tape,
- some bidirectional work tapes, initially filled with zeros.
- all tapes are binary (no blank symbol!),
- T halts on input p with output $x :\Longleftrightarrow T(p) = x$
  $:\Longleftrightarrow$ p is to the left 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.

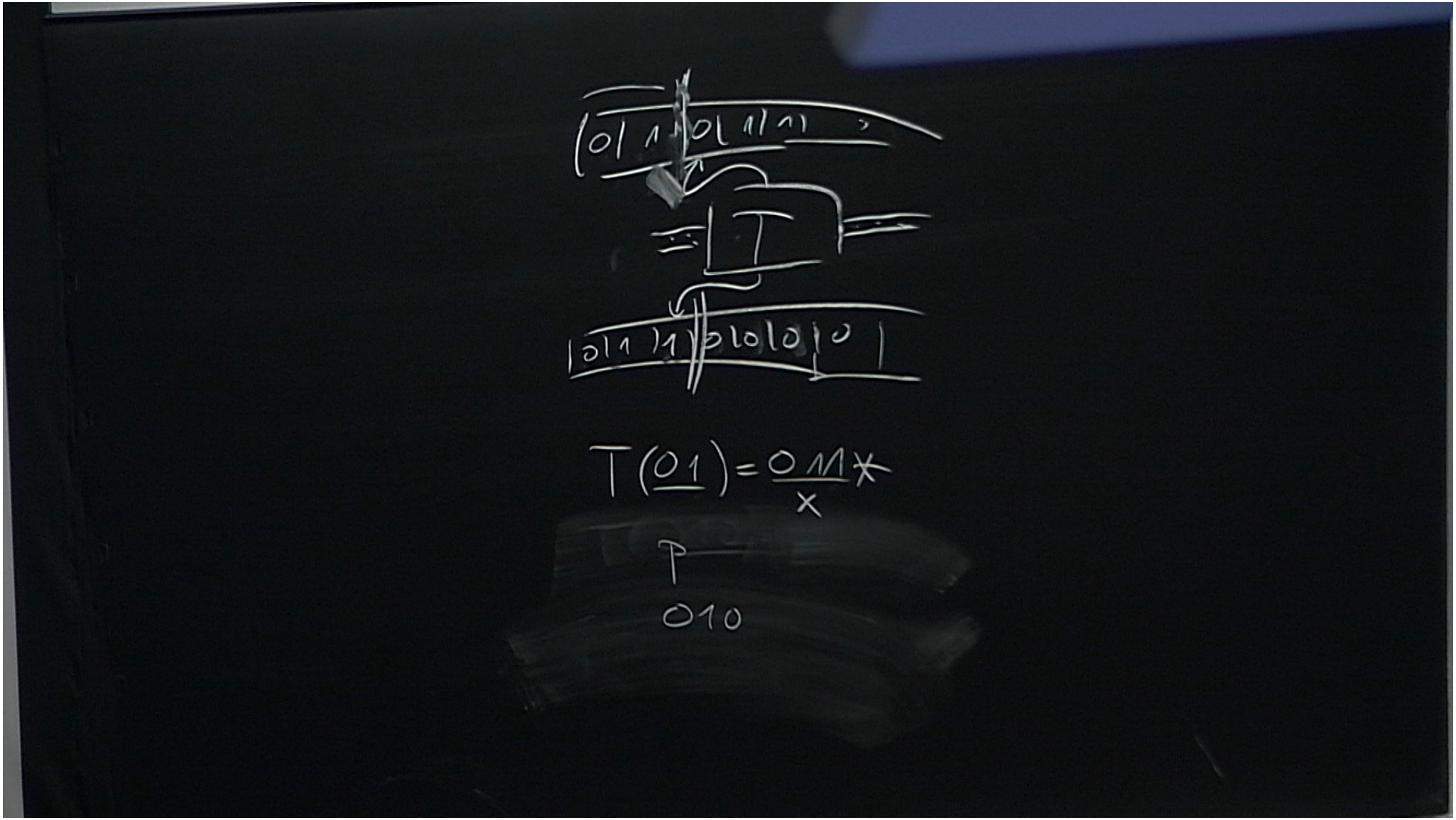
Monotone Turing Machine

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.


T(01) = 011*
T(010) = 0110*
{01, 010, ...}


T
T(01)=0 11*
x
T
010

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.

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

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: $T(i\text{‘}p) = 1 \Longleftrightarrow T_i(p)$ 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 $c_{U'U''}$.

(Conditional) Prefix Kolmogorov Complexity

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

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

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

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

e.g. $K(x,y) = K(\langle x,y\rangle) = K(x\text{`}y)$.

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

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! ■

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

Theorem 18 (Probability coding / MDL)

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

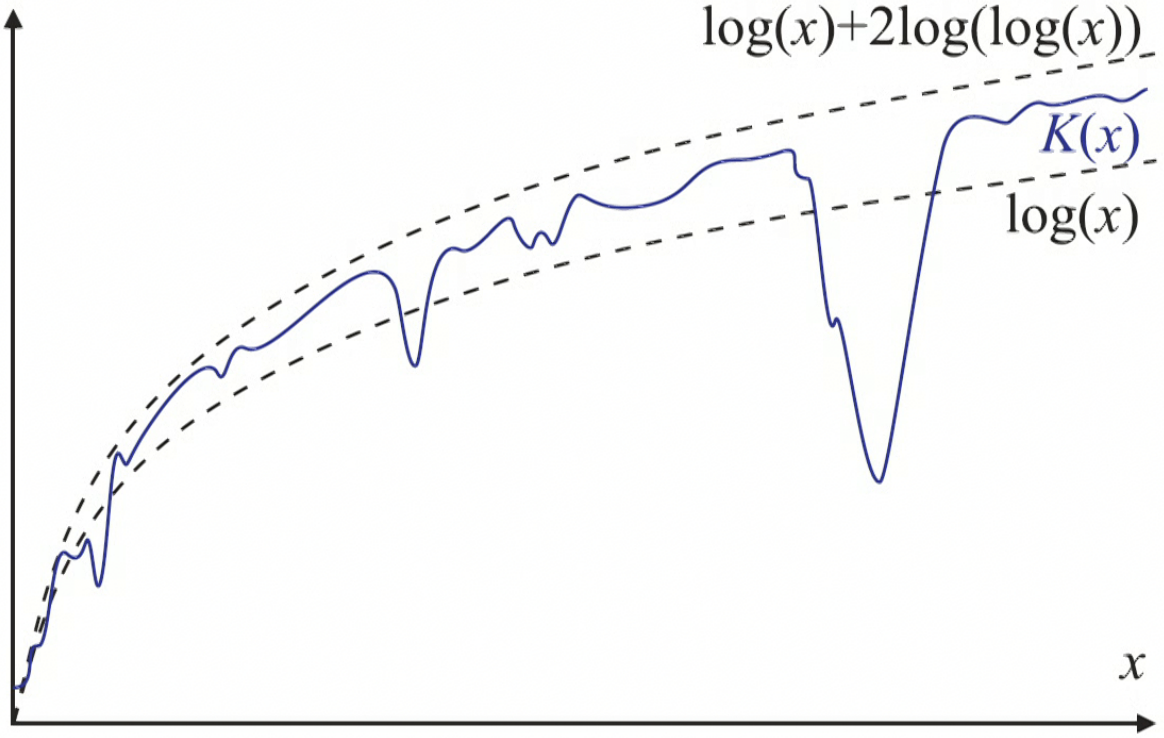
This is at the heart of the MDL principle [Ris89], which approximates $K(x)$ by $-\log P(x) + K(P)$.

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:

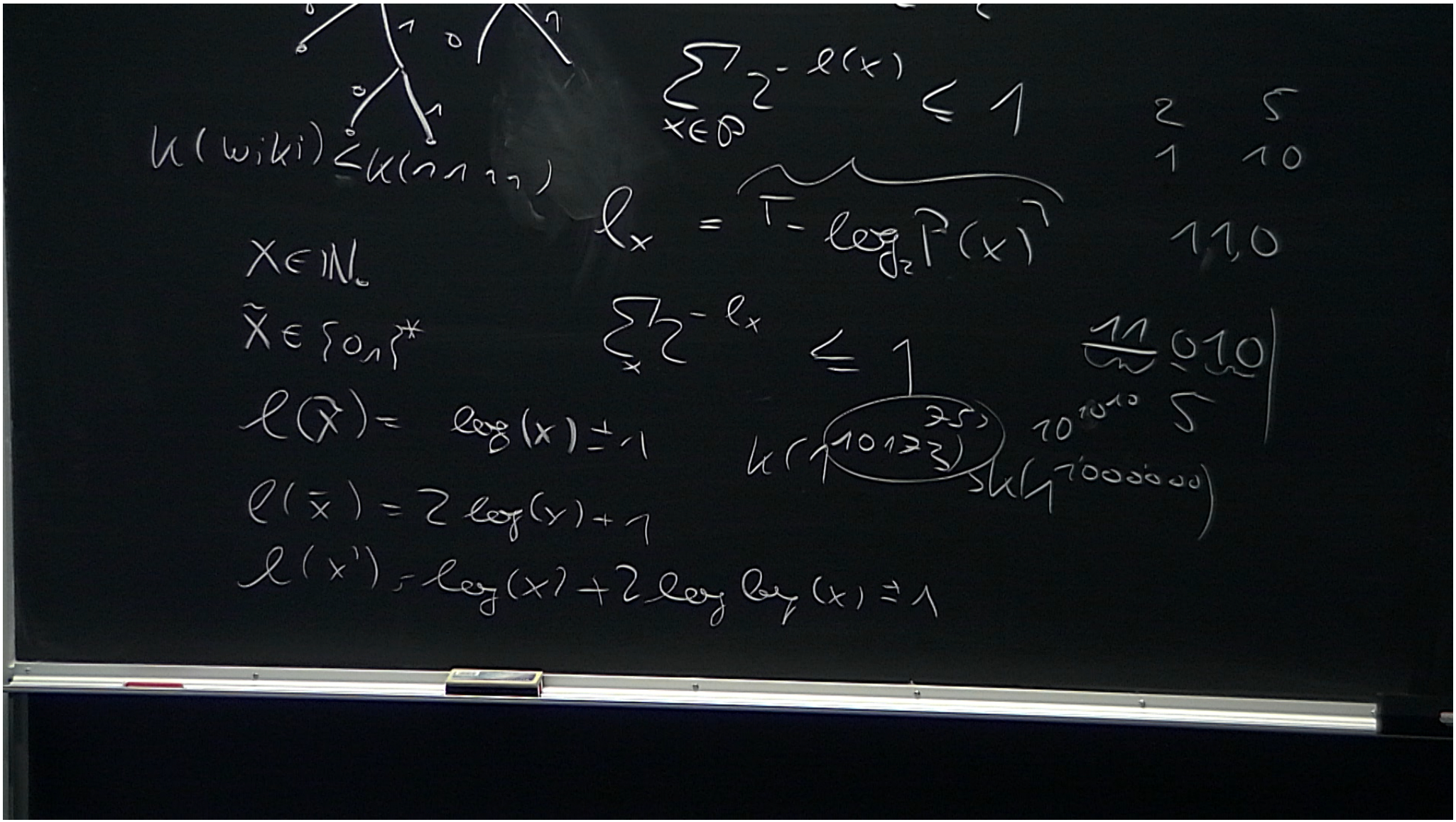
Theorem 21 (Monotone Kolmogorov Complexity Km)

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

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 $Km(\omega) < \infty$.


$\sum_{x \in \mathcal{D}} 2^{-\ell(x)} \leq 1$
$k(\text{wiki}) \leq k(\text{wiki wiki})$
$\ell_x = \lceil -\log_2 P(x) \rceil$
$x \in \mathbb{N}_0$
$\tilde{x} \in \{0,1\}^*$
$\sum_x 2^{-\ell_x} \leq 1$
$\ell(\tilde{x}) = \log(x) + 1$
$\ell(\bar{x}) = 2\log(x) + 1$
$\ell(x') = \log(x) + 2\log\log(x) + 1$
$k(10^{10^{10}})$
$k(10^{100000000})$

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 $f(x) = \frac{T_1(x)}{T_2(x)}$,

$\Downarrow$

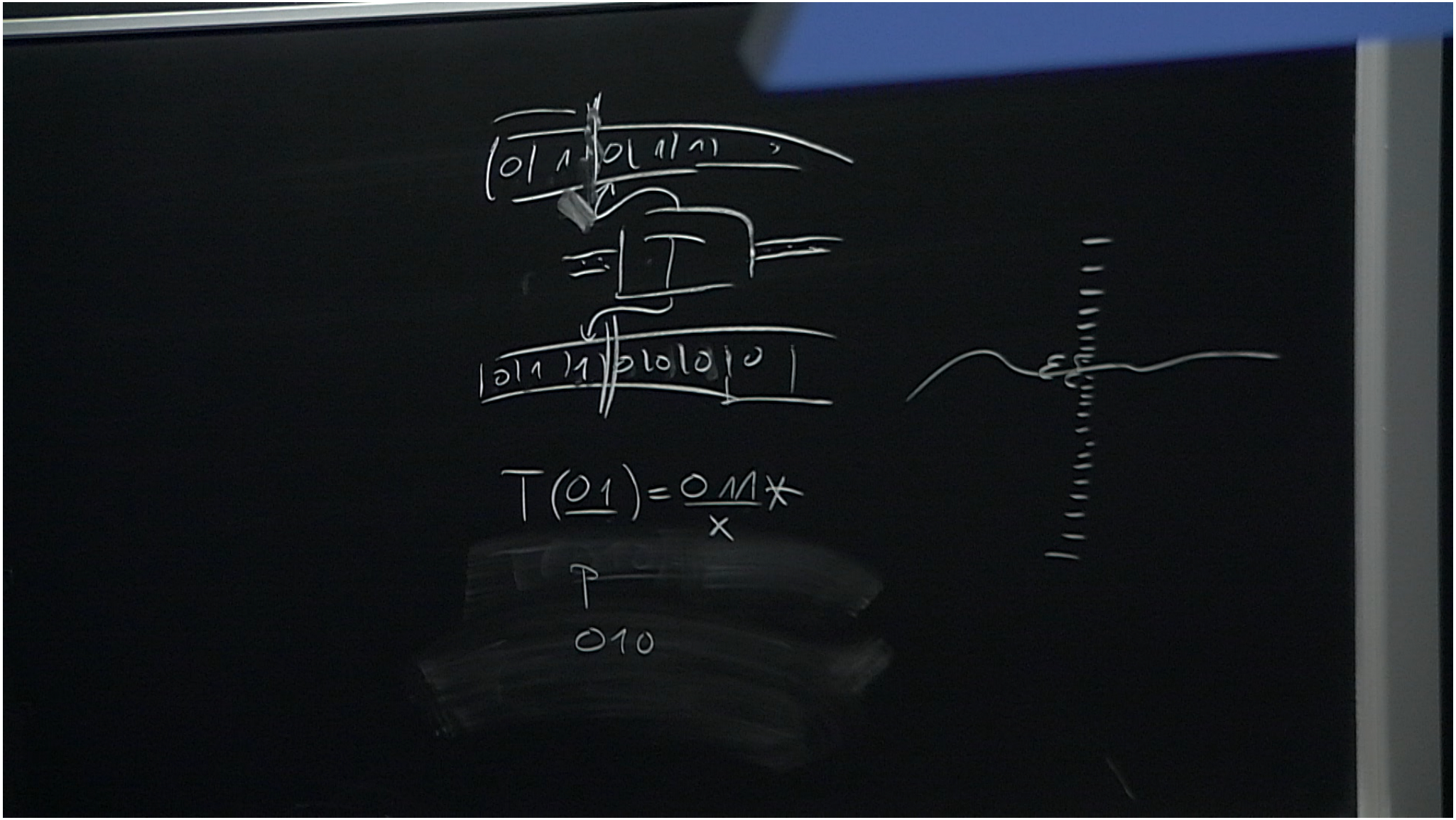
f is estimable or computable *iff* $\exists$ recursive $\phi(\cdot,\cdot)$ $\forall$ $\varepsilon > 0$:
$|\phi(x, \lfloor \frac{1}{\varepsilon} \rfloor) - f(x)| < \varepsilon$ $\forall x$.

$\Downarrow$

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
$\lim_{t\to\infty} \phi(x,t) = f(x)$.


T
T(01)=01M*
T
010

(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

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.

Presented Applications of AIT

- Philosophy: problem of induction
- Machine learning: time-series forecasting
- Artificial intelligence: foundations [COMP4620/COMP8620]
- Probability theory: choice of priors
- Information theory: individual randomness/information
- Data mining: clustering, measuring similarity
- Bioinformatics: phylogeny tree reconstruction
- Linguistics: language tree reconstruction

2 UNIVERSAL A PRIORI PROBABILITY

- The Universal a Priori Probability M
- Relations between Complexities
- Fundamental Universality Property of M

The Universal a Priori Probability M

Solomonoff defined the universal probability distribution $M(x)$ as the probability that the output of a universal monotone Turing machine starts with x when provided withfair coin flips on the input tape.

Definition 2.1 (Solomonoff distribution) Formally,

$$M(x) := \sum_{p\,:\,U(p)=x*} 2^{-\ell(p)}$$

The sum is over minimal programs p for which U outputs a string starting with x.

Since the shortest programs p dominate the sum, $M(x)$ is roughly $2^{-Km(x)}$. More precisely ...

Relations between Complexities

Theorem 2.2 (Relations between Complexities)

$KM := -\log M$, Km, and K are ordered in the following way:

$$0 \leq K(x|\ell(x)) \stackrel{+}{<} KM(x) \leq Km(x) \leq K(x) \stackrel{+}{<} \ell(x) + 2\log\ell(x)$$

Proof sketch:

The second inequality follows from the fact that,
given n and Kraft's inequality $\sum_{x\in\mathcal{X}^n} M(x) \leq 1$,
there exists for $x \in \mathcal{X}^n$ a Shannon-Fano code of length $-\log M(x)$,
which is effective since M is enumerable.

Now use the MDL bound conditioned to n.

The other inequalities are obvious from the definitions. ■

3 UNIVERSAL SEQUENCE PREDICTION

- Solomonoff, Occam, Epicurus
- Prediction
- Simple Deterministic Bound
- Solomonoff's Major Result
- Implications of Solomonoff's Result
- Universal Inductive Inference
- More Stuff / Critique / Problems

Prediction

How does all this affect prediction?

If $M(x)$ correctly describes our (subjective) prior belief in x, then

$$M(y|x) := M(xy)/M(x)$$

must be our posterior belief in y.

From the symmetry of algorithmic information $K(x,y) \stackrel{+}{=} K(y|x,K(x)) + K(x)$, and assuming $K(x,y) \approx K(xy)$, and approximating $K(y|x,K(x)) \approx K(y|x)$, $M(x) \approx 2^{-K(x)}$, and $M(xy) \approx 2^{-K(xy)}$ we get:

$$M(y|x) \approx 2^{-K(y|x)}$$

This tells us that M predicts y with high probability iff y has an easy explanation, given x (Occam & Epicurus).

Simple Deterministic Bound

Sequence prediction algorithms try to predict the continuation $x_t \in \{0,1\}$ of a given sequence $x_1...x_{t-1}$. Simple deterministic bound:

$$\sum_{t=1}^{\infty} |1-M(x_t|x_{<t})| \overset{a}{\leq} -\sum_{t=1}^{\infty} \ln M(x_t|x_{<t}) \overset{b}{=} -\ln M(x_{1:\infty}) \overset{c}{\leq} \ln 2 \cdot Km(x_{1:\infty})$$

(a) use $|1-a| \leq -\ln a$ for $0 \leq a \leq 1$.

(b) exchange sum with logarithm and eliminate product by chain rule.

(c) used Theorem 2.2.

If $x_{1:\infty}$ is a computable sequence, then $Km(x_{1:\infty})$ is finite, which implies $M(x_t|x_{<t}) \to 1 \quad (\sum_{t=1}^{\infty} |1-a_t| < \infty \Rightarrow a_t \to 1)$.

$\Rightarrow$ if environment is a computable sequence (digits of π or Expert or ...), after having seen the first few digits, M correctly predicts the next digit with high probability, i.e. it recognizes the structure of the sequence.

Simple Deterministic Bound

Sequence prediction algorithms try to predict the continuation $x_t \in \{0,1\}$ of a given sequence $x_1...x_{t-1}$. Simple deterministic bound:

$$\sum_{t=1}^{\infty} |1-M(x_t|x_{<t})| \overset{a}{\leq} -\sum_{t=1}^{\infty} \ln M(x_t|x_{<t}) \overset{b}{=} -\ln M(x_{1:\infty}) \overset{c}{\leq} \ln 2 \cdot Km(x_{1:\infty})$$

(a) use $|1-a| \leq -\ln a$ for $0 \leq a \leq 1$.

(b) exchange sum with logarithm and eliminate product by chain rule.

(c) used Theorem 2.2.

If $x_{1:\infty}$ is a computable sequence, then $Km(x_{1:\infty})$ is finite, which implies $M(x_t|x_{<t}) \to 1 \quad (\sum_{t=1}^{\infty} |1-a_t| < \infty \Rightarrow a_t \to 1)$.

$\Rightarrow$ if environment is a computable sequence (digits of π or Expert or ...), after having seen the first few digits, M correctly predicts the next digit with high probability, i.e. it recognizes the structure of the sequence.

Simple Deterministic Bound

Sequence prediction algorithms try to predict the continuation $x_t \in \{0,1\}$ of a given sequence $x_1...x_{t-1}$. Simple deterministic bound:

$$\sum_{t=1}^{\infty} |1-M(x_t|x_{<t})| \overset{a}{\leq} -\sum_{t=1}^{\infty} \ln M(x_t|x_{<t}) \overset{b}{=} -\ln M(x_{1:\infty}) \overset{c}{\leq} \ln 2 \cdot Km(x_{1:\infty})$$

(a) use $|1-a| \leq -\ln a$ for $0 \leq a \leq 1$.

(b) exchange sum with logarithm and eliminate product by chain rule.

(c) used Theorem 2.2.

If $x_{1:\infty}$ is a computable sequence, then $Km(x_{1:\infty})$ is finite, which implies $M(x_t|x_{<t}) \to 1 \quad (\sum_{t=1}^{\infty} |1-a_t| < \infty \Rightarrow a_t \to 1)$.

$\Rightarrow$ if environment is a computable sequence (digits of π or Expert or ...), after having seen the first few digits, M correctly predicts the next digit with high probability, i.e. it recognizes the structure of the sequence.

More Stuff / Critique / Problems

- Other results: M convergence rapidly also on stochastic sequences; solves the zero-prior & old evidence & new theories problems; can confirm universal hypotheses; is reparametrization invariant; predicts better than all other predictors.
- Prior knowledge y can be incorporated by using "subjective" prior $w^U_{\nu|y} = 2^{-K(\nu|y)}$ or by prefixing observation x by y.
- Additive/multiplicative constant fudges and U-dependence is often (but not always) harmless.
- Incomputability: K and M can serve as "gold standards" which practitioners should aim at, but have to be (crudely) approximated in practice (MDL [Ris89], MML [Wal05], LZW [LZ76], CTW [WSTT95], NCD [CV05]).

4 MARTIN-LÖF RANDOMNESS

- When is a Sequence Random? If it is incompressible!
- Motivation: For a fair coin 00000000 is as likely as 01100101, but we "feel" that 00000000 is less random than 01100101.
- Martin-Löf randomness captures the important concept of randomness of **individual** sequences.
- Martin-Löf random sequences pass all effective randomness tests.

5 THE MINIMUM DESCRIPTION LENGTH PRINCIPLE

- MDL as Approximation of Solomonoff's M
- The Minimum Description Length Principle

The Minimum Description Length Principle

Identification of probabilistic model "best" describing data:

Probabilistic model(=hypothesis) H_ν with $\nu \in \mathcal{M}$ and data D.

Most probable model is $\nu^{\text{MDL}} = \arg\max_{\nu\in\mathcal{M}} p(H_\nu|D)$.

Bayes' rule: $p(H_\nu|D) = p(D|H_\nu)\cdot p(H_\nu)/p(D)$.

Occam's razor: $p(H_\nu) = 2^{-Kw(\nu)}$.

By definition: $p(D|H_\nu) = \nu(x)$, $D = x$ =data-seq., $p(D)$ =const.

Take logarithm:

Definition 5.1 (MDL) $\nu^{\text{MDL}} = \arg\min_{\nu\in\mathcal{M}}\{K\nu(x) + Kw(\nu)\}$

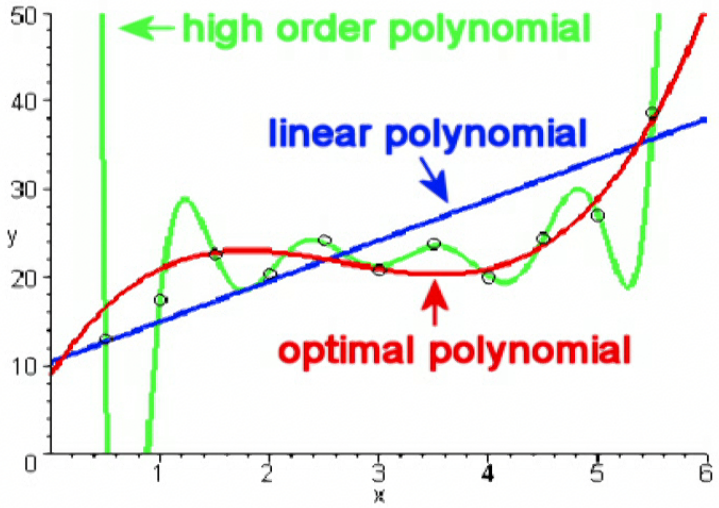
$K\nu(x) := -\log\nu(x)$ = length of Shannon-Fano code of x given H_ν.

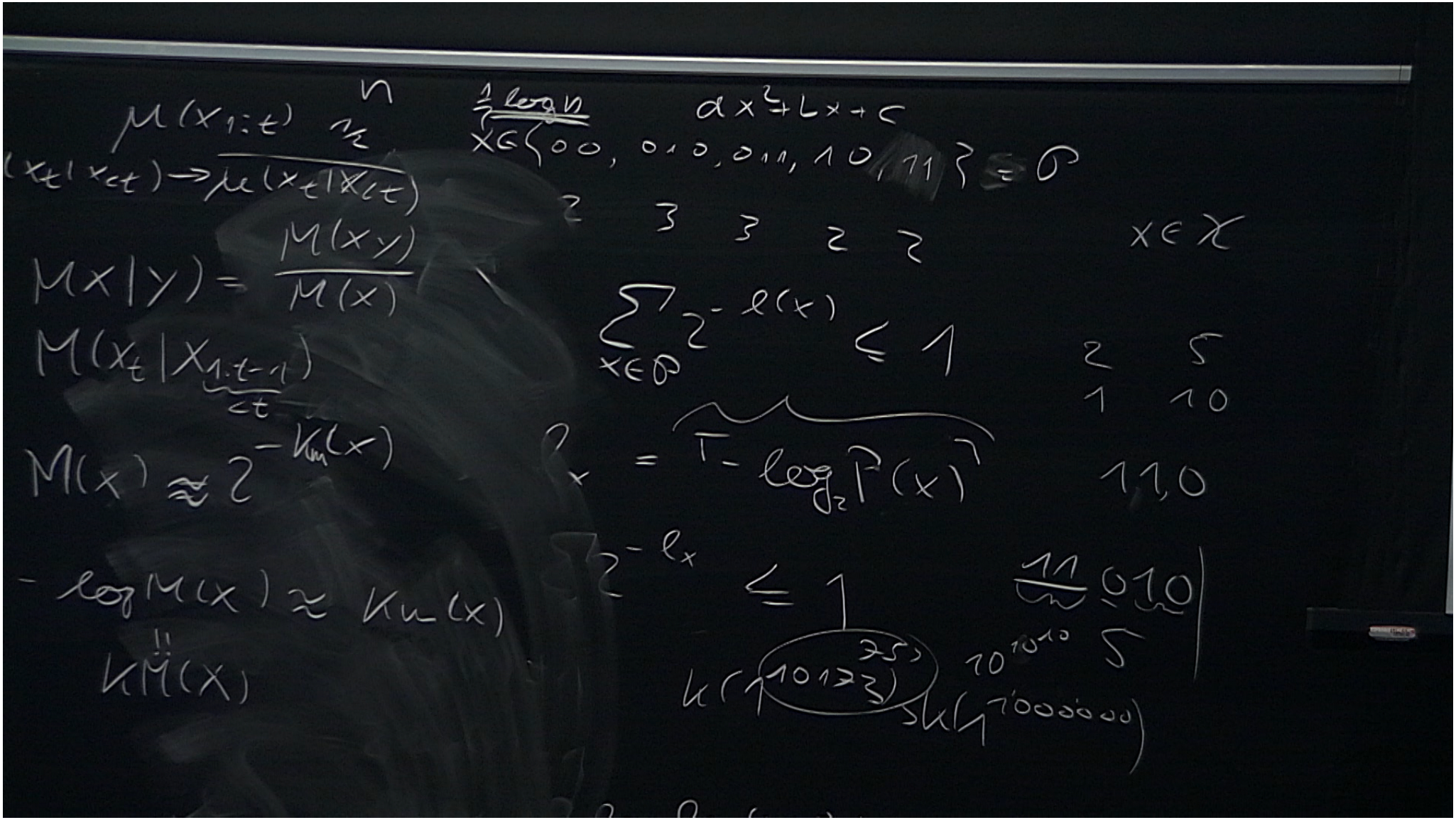
$Kw(\nu)$ = length of model H_ν.

Names: Two-part MDL or MAP or MML ($\exists$ slight/major differences)

Application: Regression / Polynomial Fitting

- Data $D = \{(x_1, y_1), ..., (x_n, y_n)\}$
- Fit polynomial $f_d(x) := a_0 + a_1 x + a_2 x^2 + ... + a_d x^d$ of degree d through points D
- Measure of error: $SQ(a_0...a_d) = \sum_{i=1}^{n} (y_i - f_d(x_i))^2$
- Given d, minimize $SQ(a_{0:d})$ w.r.t. parameters $a_0...a_d$.
- This classical approach does not tell us how to choose d? ($d \geq n-1$ gives perfect fit)




$M(x_{1:t}) \overset{n}{\to} \infty$
$\frac{1}{n}\log n$
ax^2+bx+c
$x \in \{00, 010, 011, 10, 11\} = \mathcal{D}$
$x \in \mathcal{X}$
$M(x|y) = \frac{M(xy)}{M(y)}$
$\sum_{x \in \mathcal{D}} 2^{-\ell(x)} \leq 1$
$M(x_t|x_{1:t-1})$
$M(x) \approx 2^{-K_M(x)}$
$\ell_x = \lceil -\log_2 P(x) \rceil$
$-\log M(x) \approx K_M(x)$
$K(M)(x)$
$2^{-\ell_x} \leq 1$
$K(10123)$
$10^{10^{10}}$
5
$K(10000000)$

Conditional Kolmogorov Complexity

Question: When is object=string x similar to object=string y?

Universal solution: x similar y $\Leftrightarrow$ x can be easily (re)constructed from y $\Leftrightarrow$ Kolmogorov complexity $K(x|y) := \min\{\ell(p) : U(p,y) = x\}$ is small

Examples:

1) x is very similar to itself ($K(x|x) \stackrel{+}{=} 0$)
2) A processed x is similar to x ($K(f(x)|x) \stackrel{+}{=} 0$ if $K(f) = O(1)$).
   e.g. doubling, reverting, inverting, encrypting, partially deleting x.
3) A random string is with high probability not similar to any other string ($K(\text{random}|y) = \text{length}(\text{random})$).

The problem with $K(x|y)$ as similarity=distance measure is that it is neither symmetric nor normalized nor computable.

The Universal Similarity Metric

- Symmetrization and normalization leads to a/the universal metric d:

$$0 \leq d(x,y) := \frac{\max\{K(x|y), K(y|x)\}}{\max\{K(x), K(y)\}} \leq 1$$

- Every effective similarity between x and y is detected by d
- Use $K(x|y) \approx K(xy) - K(y)$ (coding T) and $K(x) \equiv K_U(x) \approx K_T(x)$ $\Longrightarrow$ computable approximation: Normalized compression distance:

$$d(x,y) \approx \frac{K_T(xy) - \min\{K_T(x), K_T(y)\}}{\max\{K_T(x), K_T(y)\}} \lesssim 1$$

- For T choose Lempel-Ziv or gzip or bzip(2) (de)compressor in the applications below.
- Theory: Lempel-Ziv compresses asymptotically better than any probabilistic finite state automaton predictor/compressor.

Tree-Based Clustering [CV'05]

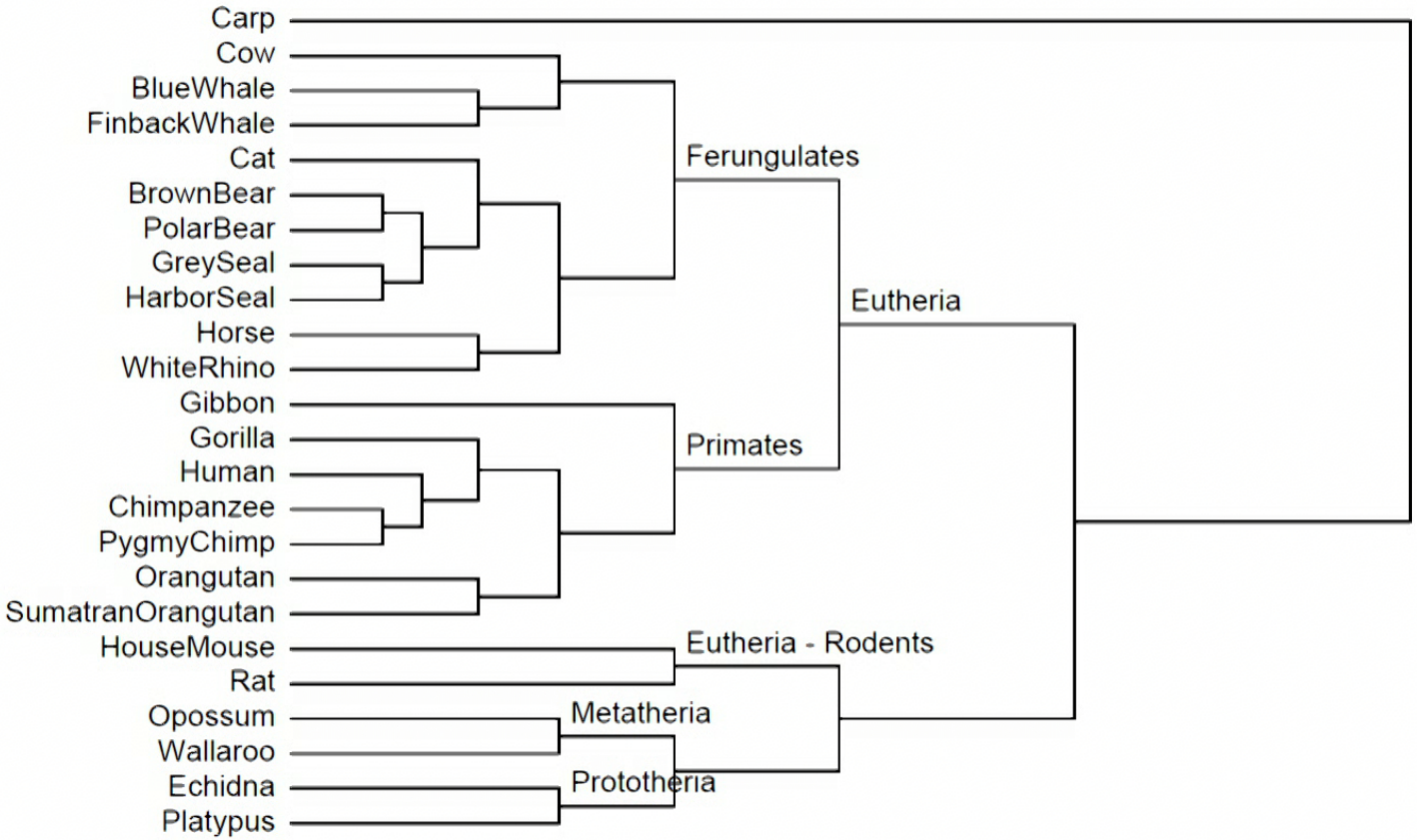
- If many objects $x_1, ..., x_n$ need to be compared, determine the Similarity matrix: $M_{ij} = d(x_i, x_j)$ for $1 \leq i, j \leq n$
- Now cluster similar objects.
- There are various clustering techniques.
- Tree-based clustering: Create a tree connecting similar objects,
- e.g. quartet method (for clustering)
- Applications: Phylogeny of 24 Mammal mtDNA,
  50 Language Tree (based on declaration of human rights),
  composers of music, authors of novels, SARS virus, fungi,
  optical characters, galaxies, ... [Cilibrasi&Vitanyi'05]

Tree-Based Clustering [CV'05]

- If many objects $x_1, ..., x_n$ need to be compared, determine the Similarity matrix: $M_{ij} = d(x_i, x_j)$ for $1 \leq i, j \leq n$
- Now cluster similar objects.
- There are various clustering techniques.
- Tree-based clustering: Create a tree connecting similar objects,
- e.g. quartet method (for clustering)
- Applications: Phylogeny of 24 Mammal mtDNA, 50 Language Tree (based on declaration of human rights), composers of music, authors of novels, SARS virus, fungi, optical characters, galaxies, ... [Cilibrasi&Vitanyi'05]

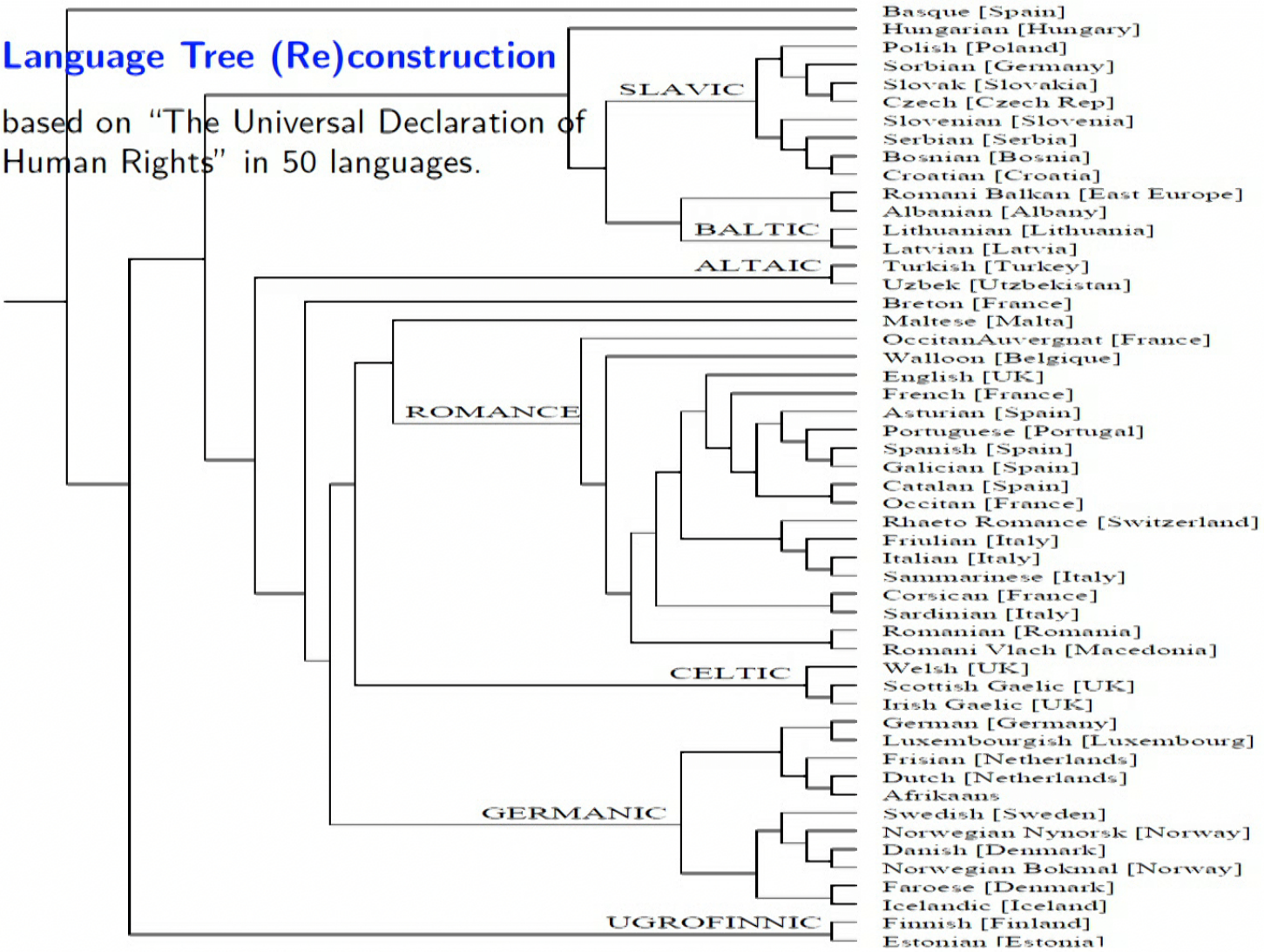
Genomics & Phylogeny: Mammals

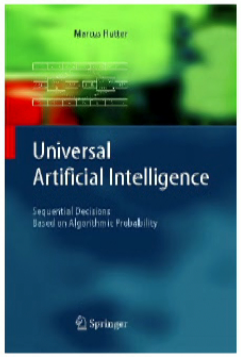
Evolutionary tree built from complete mammalian mtDNA of 24 species:



Language Tree (Re)construction

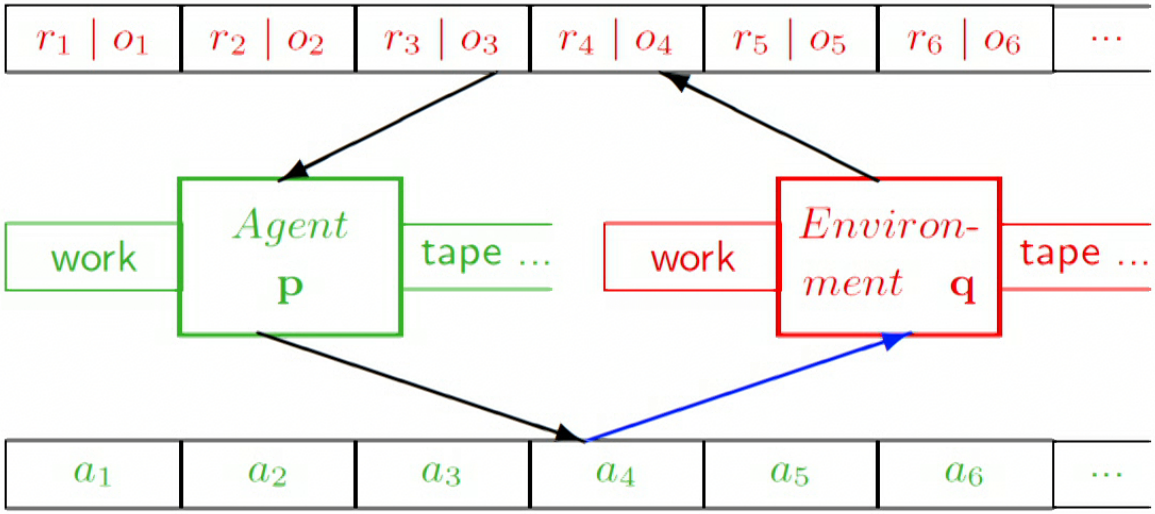
based on "The Universal Declaration of Human Rights" in 50 languages.





The Agent Model

Most if not all AI problems can be formulated within the agent framework



Formal Definition of Intelligence

- Agent follows policy $\pi : (\mathcal{A}\times\mathcal{O}\times\mathcal{R})^* \rightsquigarrow \mathcal{A}$
- Environment reacts with $\mu : (\mathcal{A}\times\mathcal{O}\times\mathcal{R})^*\times\mathcal{A} \rightsquigarrow \mathcal{O}\times\mathcal{R}$
- Performance of agent π in environment μ
  $=$ expected cumulative reward $= V_\mu^\pi := \mathbb{E}_\mu[\sum_{t=1}^\infty r_t^{\pi\mu}]$
- True environment μ unknown
  $\Rightarrow$ average over wide range of environments
- Ockham+Epicurus: Weigh each environment with its Kolmogorov complexity $K(\mu) := \min_p\{length(p) : U(p) = \mu\}$
- Universal intelligence of agent π is $\Upsilon(\pi) := \sum_\mu 2^{-K(\mu)} V_\mu^\pi$.
- Compare to our informal definition: Intelligence measures an agent's ability to perform well in a wide range of environments.
- AIXI $= \arg\max_\pi \Upsilon(\pi) =$ most intelligent agent.

Computational Issues: Universal Search

- Levin search: Fastest algorithm for inversion and optimization problems.

- Theoretical application:
  Assume somebody found a non-constructive proof of P=NP, then Levin-search is a polynomial time algorithm for every NP (complete) problem.
- Practical applications (J. Schmidhuber)
  Maze, towers of hanoi, robotics, ...
- FastPrg: The asymptotically fastest and shortest algorithm for all well-defined problems.
- AIXItl and ΦMDP: Computable variants of AIXI.
- Human Knowledge Compression Prize: (50'000€)

Formal Definition of Intelligence

- Agent follows policy $\pi : (\mathcal{A}\times\mathcal{O}\times\mathcal{R})^* \rightsquigarrow \mathcal{A}$
- Environment reacts with $\mu : (\mathcal{A}\times\mathcal{O}\times\mathcal{R})^*\times\mathcal{A} \rightsquigarrow \mathcal{O}\times\mathcal{R}$
- Performance of agent π in environment μ
  $=$ expected cumulative reward $= V_\mu^\pi := \mathbb{E}_\mu[\sum_{t=1}^\infty r_t^{\pi\mu}]$
- True environment μ unknown
  $\Rightarrow$ average over wide range of environments
- Ockham+Epicurus: Weigh each environment with its Kolmogorov complexity $K(\mu) := \min_p\{length(p) : U(p) = \mu\}$
- Universal intelligence of agent π is $\Upsilon(\pi) := \sum_\mu 2^{-K(\mu)} V_\mu^\pi$.
- Compare to our informal definition: Intelligence measures an agent's ability to perform well in a wide range of environments.
- AIXI $= \arg\max_\pi \Upsilon(\pi) =$ most intelligent agent.

8 MORE APPLICATIONS OF AIT/KC

- 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

[LV07] M. Li and P. M. B. Vitányi.
Applications of Algorithmic Information Theory.
Scholarpedia, 2:5 (2007) 2658 [an extended encyclopedic entry]

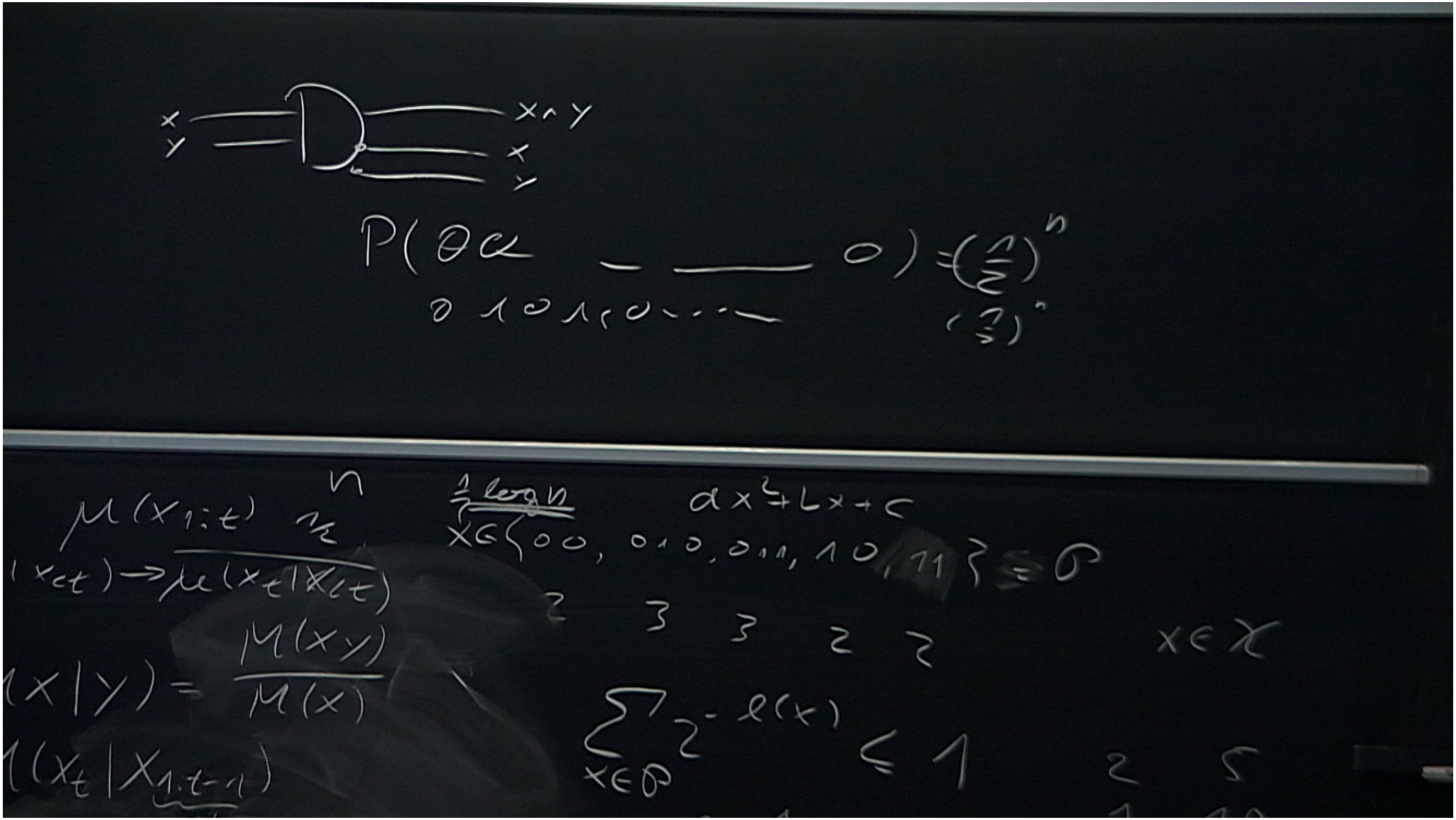
[LV08] M. Li and P. M. B. Vitányi. *An Introduction to Kolmogorov Complexity and its Applications*. Springer, Berlin, 3rd edition, 2008.

[CV05] R. Cilibrasi and P. M. B. Vitányi. *Clustering by compression.* IEEE Trans. Information Theory, 51(4):1523–1545, 2005.

[RH11] S. Rathmanner and M. Hutter. A philosophical treatise of universal induction. *Entropy*, 16(6):1076–1136, 2011.

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

See also Advanced AI course COMP4620/COMP8620 @ ANU


x
y
x∧y
x
y
P(
0
$\left(\frac{1}{2}\right)^n$
$\left(\frac{1}{2}\right)^n$
$\mu(x_{1:t})$
$\frac{1}{2}\log n$
ax^2+bx+c
$x \in \{00, 01, 10, 11\} = \mathcal{X}$
$\mu(x_t|x_{<t}) \to \mu(x_t|x_{<t})$
$\mu(x|y) = \frac{\mu(xy)}{\mu(y)}$
$\mu(x_t|x_{1:t-1})$
$\sum_{x \in \mathcal{X}} 2^{-\ell(x)} \leq 1$