HIERARCHIES OF GENERALIZED KOLMOGOROV COMPLEXITIES AND NONENUMERABLE UNIVERSAL MEASURES COMPUTABLE IN THE LIMIT

JÜRGEN SCHMIDHUBER*

IDSIA, Galleria 2, 6928 Manno (Lugano), Switzerland

ABSTRACT

The traditional theory of Kolmogorov complexity and algorithmic probability focuses on monotone Turing machines with one-way write-only output tape. This naturally leads to the universal enumerable Solomonoff-Levin measure. Here we introduce more general, nonenumerable but cumulatively enumerable measures (CEMs) derived from Turing machines with lexicographically nondecreasing output and random input, and even more general approximable measures and distributions computable in the limit. We obtain a natural hierarchy of generalizations of algorithmic probability and Kolmogorov complexity, suggesting that the "true" information content of some (possibly infinite) bitstring x is the size of the shortest nonhalting program that converges to x and nothing but x on a Turing machine that can edit its previous outputs. Among other things we show that there are objects computable in the limit yet more random than Chaitin's "number of wisdom" Omega, that any approximable measure of x is small for any x lacking a short description, that there is no universal approximable distribution, that there is a universal CEM, and that any nonenumerable CEM of x is small for any x lacking a short enumerating program. We briefly mention consequences for universes sampled from such priors.

Keywords: computability in the limit, generalized algorithmic probability, generalized Kolmogorov complexity hierarchy, halting probability Omega, cumulatively enumerable measures, computable universes.

1. Introduction and Outline

We extend the theory of algorithmic probability [40, 25, 13, 31, 41, 45, 28, 29, 16, 14, 17, 37, 1, 30, 39, 9, 42, 24, 23] by studying nonenumerable priors computable in the limit [19, 33, 15]. This leads to a hierarchy of generalizations of the concepts of universal prior and Kolmogorov complexity. For instance, many objects with high traditional Kolmogorov complexity have low generalized Kolmogorov complexity; many objects with low traditional algorithmic probability have high generalized algorithmic probability.

---

*juergen@idsia.ch - http://www.idsia.ch/~juergen

To set the stage for our main results, Section 2 (Preliminaries) introduces universal Turing Machines (TMs) more general than those considered in previous related work: unlike traditional TMs, *General* TMs or GTMs may edit their previous outputs (compare Burgin's inductive TMs [6]), and *Enumerable Output Machines* (EOMs) may do this provided the output does not decrease lexicographically. In the spirit of *computability in the limit* [19, 33, 15] we will define: a formally describable bitstring x has a finite, never halting GTM program that computes x such that each output bit is revised at most finitely many times; that is, each finite prefix of x eventually *stabilizes* (Defs. 2.1-2.5); describable *functions* can be implemented by such programs (Def. 2.10); weakly decidable *problems* have solutions computable by never halting programs whose output is wrong for at most finitely many steps (Def. 2.11). This constructive notion of formal describability is less restrictive than the traditional notion of computability [43], mainly because we do not insist on the existence of a halting program that computes an upper bound of the convergence time of the n-th bit of x. Formal describability thus pushes constructivism [5, 2] to the extreme, barely avoiding the nonconstructivism embodied by even less restrictive concepts of describability — compare Δ_n^0-describability [34]. Theorem 2.1 will generalize the weakly decidable halting problem by demonstrating that it is not weakly decidable whether a finite string is a description of a describable object (there is a related insight for analytic TMs with real-valued inputs by Hotz, Vierke and Schieffer [21]).

Outline of main results. Section 3 generalizes the traditional concept of Kolmogorov complexity or algorithmic information [25, 40, 13] of finite x (the length of the shortest halting program computing x) to the case of objects describable by nonhalting programs on EOMs and GTMs (Defs. 3.2-3.3). It is shown that the generalization for EOMs is describable, but the one for GTMs is not (Theorem 3.1). Certain objects are much more compactly encodable on EOMs than on traditional *monotone* TMs, and Theorem 3.3 shows that there are also objects with short GTM descriptions yet incompressible on EOMs and therefore "more random" than Chaitin's Ω [14, 10, 39, 9, 42], the halting probability of a TM with random input, which is incompressible only on monotone TMs. This leads to a natural TM type-specific Kolmogorov complexity hierarchy expressed by Inequality (12).

Section 4 introduces the nondescribable *convergence probability* of a GTM (Def. 4.14) as well as very general formally describable (semi)measures that are computable in the limit. Unfortunately, Theorem 4.2 (proof by M. Hutter) shows that there is no universal describable measure that dominates all others, in the sense that it assigns higher probability to any bitstring x, save for a constant factor independent of x. In our search for universal measures we introduce *cumulatively enumerable* measures (CEMs, Def. 4.5), where the cumulative probability of all strings lexicographically greater than a given string y is EOM-computable or *computably enumerable (c.e.)*. In general the CEMs are not c.e., but they are computable in the limit, and they dominate the traditional c.e. priors studied in classic work by Solomonoff, Levin and others [40, 45, 29, 16, 17, 37, 41, 14, 30]. Theorem 4.1 shows that there is indeed a universal nonenumerable CEM. It is also shown that

this CEM assigns to x the probability that a universal EOM whose input bits are chosen randomly produces an output starting with x (Corollary 4.3 and Lemma 4.2).

Section 5 establishes relationships between generalized Kolmogorov complexity and generalized algorithmic probability, extending previous work on c.e. semimeasures by Levin, Gács, and others [45, 29, 16, 14, 17, 30]. For instance, Theorem 5.3 shows that the universal CEM assigns a probability to each c.e. object proportional to $\frac{1}{2}$ raised to the power of the length of its minimal EOM-based description, times a small corrective factor. For GTMs there is no analogue statement, but at least we can show that objects with approximable probabilities yet without very short descriptions on GTMs are necessarily very unlikely *a priori* (Theorems 5.4 and 5.5). Additional suspected elegant links between generalized Kolmogorov complexity and probability are expressed in form of Conjectures 5.1-5.3.

2. Preliminaries

2.1. Notation

Much but not all of the notation used here is similar or identical to the one used in the standard textbook on Kolmogorov complexity by Li and Vitányi [30].

Since sentences over any finite alphabet are encodable as bitstrings, without loss of generality we focus on the binary alphabet $B = \{0, 1\}$. λ denotes the empty string, B^* the set of finite sequences over B, B^∞ the set of infinite sequences over B, $B^\sharp = B^* \cup B^\infty$. x, y, z, z^1, z^2 stand for strings in $B^\sharp$. If $x \in B^*$ then xy is the concatenation of x and y (e.g., if $x = 10000$ and $y = 1111$ then $xy = 100001111$). Let us order $B^\sharp$ lexicographically: if x precedes y alphabetically (like in the example above) then we write $x \prec y$ or $y \succ x$; if x may also equal y then we write $x \preceq y$ or $y \succeq x$ (e.g., $\lambda \prec 001 \prec 010 \prec 1 \prec 1111...$). The context will make clear where we also identify $x \in B^*$ with a unique nonnegative integer $1x$ (e.g., string 0100 is represented by integer 10100 in the dyadic system or $20 = 1*2^4 + 0*2^3 + 1*2^2 + 0*2^1 + 0*2^0$ in the decimal system). Indices $i, j, m, m_0, m_1, n, n_0, t, t_0$ range over the positive integers, constants c, c_0, c_1 over the positive reals, f, g denote functions mapping integers to integers, *log* the logarithm with basis 2, $lg(r) = max_k\{integer\ k\ :\ 2^k \leq r\}$ for real $r > 0$. For $x \in B^* \backslash \{\lambda\}$, $0.x$ stands for the real number with dyadic expansion x (note that $0.x0111.... = 0.x1 = 0.x10 = 0.x100...$ for $x \in B^*$, although $x0111.... \neq x1 \neq x10 \neq x100...$). For $x \in B^*$, $l(x)$ denotes the number of bits in x, where $l(x) = \infty$ for $x \in B^\infty$; $l(\lambda) = 0$. x_n is the prefix of x consisting of the first n bits, if $l(x) \geq n$, and x otherwise ($x_0 := \lambda$). For those $x \in B^*$ that contain at least one 0-bit, x' denotes the lexicographically smallest $y \succ x$ satisfying $l(y) \leq l(x)$ (x' is undefined for x of the form $111...1$). We write $f(n) = O(g(n))$ if there exists c, n_0 such that $f(n) \leq cg(n)$ for all $n > n_0$.

For notational simplicity we will use the $\sum$ sign also to indicate summation over uncountably many strings in $B^\sharp$, rather than using traditional measure notation and $\int$ signs. The reader should not feel uncomfortable with this notational liberty —

density-like nonzero sums over uncountably many bitstrings, each with individual measure zero, will not play any critical role in the proofs.

2.2. Turing Machines: Monotone TMs (MTMs), General TMs (GTMs), Enumerable Output Machines (EOMs)

The standard model of theoretical computer science is the Turing Machine (TM) [43]. It allows for emulating any known computer. Most TMs considered in previous work are *monotone,* that is, once they print out a bit they cannot revise it later. We will point out that such TMs are in a certain sense less expressive than other TMs that may edit their previous outputs — certain objects without any short nonhalting programs on traditional TMs have very short programs on other TMs. To deal with the differences between monotone computability, enumerability, and computability in the limit, we consider three types of TMs.

Monotone TMs (MTMs). Most current theory of description size and inductive inference is based on MTMs (compare [30, p. 276 ff]) with several tapes, each tape being a finite chain of adjacent squares with a scanning head initially pointing to the leftmost square. There is one output tape and at least two work tapes (sufficient to efficiently compute everything traditionally regarded as computable). The MTM has a finite number of internal states, one of them being the initial state. MTM behavior is specified by a lookup table mapping current state and contents of the squares above work tape scanning heads to a new state and an instruction to be executed next. There are instructions for shifting work tape scanning heads one square left or right (appending new squares when necessary), and for writing 0 or 1 on squares above work tape scanning heads. The only input-related instruction requests an input bit determined by an external process and copies it onto the square above the first work tape scanning head (no extra input tape). There may or may not be a halt instruction to terminate a computation. MTMs are called *monotone* because they have a one-way write-only output tape — they cannot edit their previous output, because the only ouput instructions are: append new square at the right end of the output tape and fill it with 0/1.

General TMs (GTMs). GTMs embody the concept of computability in the limit. They are like MTMs but have additional output instructions to edit their previous output. Our motivation for introducing GTMs is that certain bitstrings are very compactly describable on nonhalting GTMs but not on MTMs, as will be seen later. This has consequences for definitions of individual describability and probability distributions on describable things. The additional instructions are: (a) shift output scanning head right/left (but not out of bounds); (b) delete square at the right end of the output tape (if it is not the initial square or above the scanning head); (c) write 1 or 0 on square above output scanning head. Compare Burgin's *inductive* TMs and super-recursive algorithms [6, 8].

Example 2.1 (Pseudorandom universe based on halting problem) Why consider GTMs at all? Because in a certain sense they represent the non-plus-ultra of constructive computability. For instance, consider a programmer of virtual realities. Which are his fundamental limits; which universes could he possibly generate?

He could write a finite program that computes a never-ending sequence x of finite bitstrings $x^1, x^2, \ldots$ representing the history of some discrete universe, where x^k represents the state at discrete time step k, and x^1 the "Big Bang" [35]. "Noise" will have to be simulated by invoking a finite pseudorandom generator subroutine PRG. Suppose its n-th output bit $PRG(n)$ is 1 if the n-th program of a list of all possible programs halts, and 0 otherwise. There is no halting algorithm computing $PRG(n)$ for all n, otherwise the halting problem would be solvable, which it is not [43]. Hence in general there is no computer that outputs x *and only* x without ever editing some previously computed history. That is, if we want to study the set of all programmable universes [36] then we should not limit ourselves to MTMs but consider GTMs as well.

Note, however, that the output of a GTM might not stabilize in the sense that certain output bits might flip from 0 to 1 and back forever.

Enumerable Output Machines (EOMs). EOMs embody the important concept of computable enumerability. Like GTMs, EOMs can edit their previous output, but not such that it decreases lexicographically. The expressive power of EOMs lies in between those of MTMs and GTMs, with interesting universality properties whose analogues do not hold for GTMs. EOMs are like MTMs, except that the only permitted output instruction sequences are: (a) shift output tape scanning head left/right unless this leads out of bounds; (b) replace bitstring starting above the output scanning head by the string to the right of the scanning head of the second work tape, readjusting output tape size accordingly, but only if this lexicographically increases the contents of the output tape. The necessary test can be hardwired into the finite TM transition table. A significant fraction of this paper concerns EOMs with randomly chosen input bits.

Self-Delimiting Programs. Sequences of input bits requested by halting TMs are traditionally called *self-delimiting programs* because they convey all information about their own length [29, 16, 14]. Here we will use this term also for complete finite input sequences requested by *non*halting MTMs, EOMs, and GTMs. Consider three possible cases: (1) the input sequence is a halting self-delimiting program (traditional case), (2) the TM at some point ceases requesting new input bits but does not halt — then we have a nonhalting self-delimiting program, which may produce either finite or infinite output, (3) the TM never ceases requesting new input bits, which suggests the notion of infinite programs. In any case no program can be prefix of another one.

Example 2.2 (Illustration of TM Hierarchy) There are short self-delimiting MTM programs for the infinite dyadic expansion of π, but not for Ω, the halting probability of an MTM whose input bits are obtained by tossing an unbiased coin whenever it requests a new bit [14, 10, 39, 9, 42]. On the other hand, there are short self-delimiting EOM programs for Ω, but not for certain bitstrings that have short self-delimiting GTM programs, to be exhibited by Theorem 3.3.

Most traditional computability theory focuses on properties of halting programs. Given an MTM or EOM or GTM T with halt instruction and $p, x \in B^*$, we write

$$T(p) = x \tag{1}$$

for "p computes x on T and halts". Much of this paper, however, deals with programs that never halt, and with TMs that do not need halt instructions.

Definition 2.1 (Convergence) *Let $p \in B^\sharp$ denote the input string or program read by TM T. Let $T_t(p)$ denote T's finite output string after t instructions. We say that p and p's output* stabilize *and* converge *towards $x \in B^\sharp$ iff for each n satisfying $0 \leq n \leq l(x)$ there exists a postive integer t_n such that for all $t \geq t_n$: $T_t(p)_n = x_n$ and $l(T_t(p)) \leq l(x)$. Then we write*

$$T(p) \rightsquigarrow x. \tag{2}$$

Although each beginning or prefix of x eventually becomes stable during the possibly infinite computation, there need not be a halting program that computes an upper bound of stabilization time, given any p and prefix size. Compare the concept of computability *in the limit* [19, 33, 15] and [20, 32].

Definition 2.2 (TM-Specific Individual Describability) *Given a TM T, an $x \in B^\sharp$ is* T-describable *or* T-computable *iff there is a finite $p \in B^*$ such that $T(p) \rightsquigarrow x$.*

According to this definition, objects with infinite shortest descriptions on T are not T-describable, reflecting the fact that we will never ever be able to produce a complete T-based description of such an object.

Definition 2.3 (Universal TMs) *Let C denote a set of TMs. C has a universal element if there is a TM $U^C \in C$ such that for each $T \in C$ there exists a constant string $p_T \in B^*$ (the* compiler*) such that for all possible programs p, if $T(p) \rightsquigarrow x$ then $U^C(p_T p) \rightsquigarrow x$.*

Definition 2.4 (M, E, G) *Let M denote the set of MTMs, E denote the set of EOMs, G denote the set of GTMs.*

M, E, G all have universal elements, according to the fundamental *compiler theorem* (for instance, a fixed compiler can translate arbitrary LISP programs into equivalent FORTRAN programs).

Definition 2.5 (Individual Describability) *Let C denote a set of TMs with universal element U^C. Some $x \in B^\sharp$ is* C-describable *or* C-computable *if it is U^C-describable. E-describable strings are called* computably enumerable *(c.e.). G-describable strings are called* formally describable *or simply* describable.

Definition 2.6 (Always converging TMs) *TM T always converges if for all of its possible programs $p \in B^\sharp$ there is an $x \in B^\sharp$ such that $T(p) \rightsquigarrow x$.*

For example, MTMs and EOMs converge always. GTMs do not.

Definition 2.7 (Approximability) *Let $0.x$ denote a real number, $x \in B^\sharp \backslash \{\lambda\}$. $0.x$ is* approximable *by TM T if there is a $p \in B^*$ such that for each real $\epsilon > 0$ there exists a t_0 such that*

$$| \ 0.x - 0.T_t(p) \ | < \epsilon$$

for all times $t \geq t_0$. $0.x$ is approximable *if there is at least one GTM T as above.*

Lemma 2.1 *If $0.x$ is approximable, then x is describable, and vice versa.*

Henceforth we will exchangeably use the expressions *approximable, describable, computable in the limit.*

2.4. Formally Describable Functions

Much of the traditional theory of computable functions focuses on halting programs that map subsets of B^* to subsets of B^*. The output of a program that does not halt is usually regarded as undefined, which is occasionally expressed by notation such as $T(p) = \infty$. In this paper, however, we will not lump together all the possible outputs of nonhalting programs onto a single symbol "undefined." Instead we will consider mappings from subsets of B^* to subsets of $B^\sharp$, sometimes from $B^\sharp$ to $B^\sharp$.

Definition 2.8 (Encoding B^*) *Encode $x \in B^*$ as a self-delimiting input $p(x)$ for an appropriate TM, using*

$$l(p(x)) = l(x) + 2log \ l(x) + O(1) \tag{3}$$

bits as follows: write $l(x)$ in binary notation, insert a "0" after every "0" and a "1" after every "1," append "01" to indicate the end of the description of the size of the following string, then append x.

For instance, $x = 01101$ gets encoded as $p(x) = 1100110101101$.

Definition 2.9 (Recursive Functions) *A function $h : D_1 \subset B^* \to D_2 \subset B^*$ is recursive if there is a TM T using the encoding 2.8 such that for all $x \in D_1$: $T(p(x)) = h(x)$.*

Definition 2.10 (Describable Functions) *Let T denote a TM using the encoding of Def. 2.8. A function $h : D_1 \subset B^* \to D_2 \subset B^\sharp$ is T-describable if for all $x \in D_1 : T(p(x)) \rightsquigarrow h(x)$. Let C denote a set of TMs using encoding 2.8, with universal element U^C. h is C-describable or C-computable if it is U^C-computable. If the T above is universal among the GTMs with such input encoding (see Def. 2.3) then h is* describable.

Compare functions in the *arithmetic hierarchy* [34] and the concept of Δ_n^0 describability, e.g., [30, p. 46-47].

2.5. Weak Decidability and Convergence Problem

Traditionally, decidability of some problem class implies there is a halting algorithm that prints out the answer, given a problem from the class. We now relax the notion of decidability by allowing for infinite computations on EOMs or GTMs whose answers converge after finite yet possibly unpredictable time. Essentially, an

answer needs to be correct for almost all the time, and may be incorrect for at most finitely many initial time steps (compare the concept of *computability in the limit* [20, 19, 33, 15] and super-recursive algorithms [6, 8]).

Definition 2.11 (Weak decidability) *Consider a* characteristic function $h : D_1 \subset B^* \to B$: $h(x) = 1$ *if x satisfies a certain property, and* $h(x) = 0$ *otherwise. The problem of deciding whether or not some* $x \in D_1$ *satisfies that property is* weakly decidable *if $h(x)$ is describable (compare Def. 2.10).*

Example 2.3 Is a given string $p \in B^*$ a halting program for a given MTM? The problem is not decidable in the traditional sense (no halting algorithm solves the general halting problem [43]), but weakly decidable and even E-decidable, by a trivial algorithm: print "0" on first output square; simulate the MTM on work tapes and apply it to p, once it halts after having read no more than $l(p)$ bits print "1" on first output square.

Example 2.4 It is weakly decidable whether a finite bitstring p is a program for a given TM. Algorithm: print "0"; feed p bitwise into the internally simulated TM whenever it requests a new input bit; once the TM has requested $l(p)$ bits, print "1"; if it requests an additional bit, print "0". After finite time the output will stabilize forever.

Example 2.5 It is weakly decidable whether number theory is consistent (contrast this with Gödel's famous negative result [18]).

Theorem 2.1 (Convergence Problem) *Given a GTM, it is not weakly decidable whether a finite bitstring is a converging program, or whether some of the output bits will fluctuate forever.*

    Proof. (Based on the standard diagonalization trick [11, 18, 43]; compare a related result for analytic TMs [12, 21]). Let us write $T(x) \downarrow$ if there is a $z \in B^\sharp$ such that $T(x) \rightsquigarrow z$. Let us write $T(x) \updownarrow$ if T's output fluctuates forever in response to x (e.g., by flipping from 1 to zero and back forever). Let $A_1, A_2, \ldots$ be an effective enumeration of all GTMs. Uniquely encode all pairs of finite strings (x, y) in $B^* \times B^*$ as finite strings $code(x, y) \in B^*$. Suppose there were a GTM U such that (*****): for all $x, y \in B^* : U(code(x, y)) \rightsquigarrow 1$ if $A_x(y) \downarrow$, and $U(code(x, y)) \rightsquigarrow 0$ otherwise. Then one could construct a GTM T with $T(x) \rightsquigarrow 1$ if $U(code(x, x)) \rightsquigarrow 0$, and $T(x) \updownarrow$ otherwise. Let y be the index of $T = A_y$, then $A_y(y) \downarrow$ if $U(code(y, y)) \rightsquigarrow 0$, otherwise $A_y(y) \updownarrow$. By (*****), however, $U(code(y, y)) \rightsquigarrow 1$ if $A_y(y) \downarrow$, and $U(code(y, y)) \rightsquigarrow 0$ if $A_y(y) \updownarrow$. Contradiction. $\square$

3. Complexity of Constructive Descriptions

    The remaining sections of this paper contain its main contributions, embedded in the context of earlier work.

    Traditionally, the Kolmogorov complexity [25, 40, 13] or algorithmic complexity or algorithmic information of $x \in B^*$ is the length of the shortest halting program computing x:

Definition 3.1 (Kolmogorov Complexity K) *Fix a universal MTM or EOM*

or GTM U with halt instruction, and define

$$K(x) = \min_p \{l(p) : U(p) = x\}. \tag{4}$$

We will now extend this in novel ways to nonhalting EOMs and GTMs.

3.1. Generalized Kolmogorov Complexity for EOMs and GTMs

Definition 3.2 (Generalized K_T) *Given any TM T, define*

$$K_T(x) = \min_p \{l(p) : T(p) \rightsquigarrow x\}$$

Compare Schnorr's less general "process complexity" for MTMs [37, 44].

Proposition 3.1 (K^M, K^E, K^G based on Invariance Theorem) *Consider Def. 2.4. Let C denote a set of TMs with universal TM U^C ($T \in C$). We drop the index T, writing*

$$K^C(x) = K_{U^C}(x) \leq K_T(x) + O(1).$$

This is justified by an appropriate *Invariance Theorem* [25, 40, 13]: there is a positive constant c such that $K_{U^C}(x) \leq K_T(x) + c$ for all x, since the size of the compiler that translates arbitrary programs for T into equivalent programs for U^C does not depend on x.

Definition 3.3 (Km_T, Km^M, Km^E, Km^G) *Given TM T and $x \in B^*$, define*

$$Km_T(x) = \min_p \{l(p) : T(p) \rightsquigarrow xy, y \in B^\sharp\}. \tag{5}$$

Consider Def. 2.4. If C denotes a set of TMs with universal TM U^C, then define $Km^C(x) = Km_{U^C}(x)$.

Km^C is a generalization of Schnorr's [37] and Levin's [28] complexity measure Km^M for MTMs.

Describability of $K(x)$ and $K^E(x)$. $K(x)$ is not computable by a halting program [25, 40, 13], but obviously G-computable or describable; the z with $0.z = \frac{1}{K(x)}$ is even c.e. Even $K^E(x)$ is describable for finite x, using the following algorithm:

> Run all EOM programs in *"dovetail style"* such that the n-th step of the i-th program is executed in the $n+i$-th phase ($i = 1, 2, \ldots$); whenever a program outputs x, place it (or its prefix read so far) in a tentative list L of x-computing programs or program prefixes; whenever an element of L produces output $\succ x$, delete it from L; whenever an element of L requests an additional input bit, update L accordingly. After every change of L replace the current estimate of $K^E(x)$ by the length of the shortest element of L. This estimate will eventually stabilize forever.

Theorem 3.1 $K^G(x)$ *is not describable.*

Proof. Identify finite bitstrings with the integers they represent. If $K^G(x)$ were describable then also

$$h(x) = max_y\{K^G(y) : 1 \leq y \leq g(x)\}, \tag{6}$$

where g is any fixed recursive function, and also

$$f(x) = min_y\{y : K^G(y) = h(x)\}. \tag{7}$$

Since the number of descriptions p with $l(p) < n - O(1)$ cannot exceed $2^{n-O(1)}$, but the number of strings x with $l(x) = n$ equals 2^n, most x cannot be compressed by more than $O(1)$ bits; that is, $K^G(x) \geq log\ x - O(1)$ for most x. From (7) we therefore obtain $K^G(f(x)) > log\ g(x) - O(1)$ for large enough x, because $f(x)$ picks out one of the incompressible $y \leq g(x)$. However, obviously we also would have $K^G(f(x)) \leq l(x) + 2log\ l(x) + O(1)$, using the encoding of Def. 2.8. Contradiction for quickly growing g with low complexity, such as $g(x) = 2^{2^x}$. $\qquad\square$

3.2. Expressiveness of EOMs and GTMs

Since GTMs may occasionally rewrite parts of their output, they are computationally more expressive than MTMs in the sense that they permit much more compact descriptions of certain objects — compare also [7]. For instance, $K(x) - K^G(x)$ is unbounded, as will be shown next. This will later have consequences for predictions, given certain observations.

Theorem 3.2 $K(x) - K^G(x)$ *is unbounded.*

Proof. Define

$$h'(x) = max_y\{K(y) : 1 \leq y \leq g(x)\}; \quad f'(x) = min_y\{y : K(y) = h'(x)\}, \tag{8}$$

where g is recursive. Then $K^G(f'(x)) = O(l(x) + K(g))$ (where $K(g)$ is the size of the minimal halting description of function g), but $K(f'(x)) > log\ g(x) - O(1)$ for sufficiently large x — compare the proof of Theorem 3.1. Therefore $K(f'(x)) - K^G(f'(x)) \geq O(log\ g(x))$ for infinitely many x and quickly growing g with low complexity. $\qquad\square$

3.2.1. EOMs dominate MTMs

Similarly, some x are compactly describable on EOMs but not on MTMs. To see this, consider Chaitin's Ω, the halting probability of an MTM whose input bits are obtained by tossing an unbiased coin whenever it requests a new bit [14]. Ω is c.e. (dovetail over all programs p and sum up the contributions $2^{-l(p)}$ of the halting p), but there is no recursive upper bound on the number of instructions required to compute Ω_n, given n. This implies $K(\Omega_n) = n + O(1)$ [14] and also $K^M(\Omega_n) = n + O(1)$. It is easy to see, however, that on nonhalting EOMs there are much more compact descriptions:

$$K^E(\Omega_n) \leq O(K(n)) \leq O(log\ n); \tag{9}$$

10

that is, there is no upper bound of

$$K^M(\Omega_n) - K^E(\Omega_n).\tag{10}$$

3.2.2. GTMs dominate EOMs — novel objects less regular than Ω

We will now show that there are describable strings that are constructively computable in the limit and have a short GTM description yet are "even more random" than Chaitin's Omegas, in the sense that even on EOMs they do not have any compact descriptions — contrast this with Kurtz's and Kucera's nonconstructive randomness in the arithmetic hierarchy [27, 26].

Theorem 3.3 *For all n there are $z \in B^*$ with*

$$K^E(z) > n - O(1), \quad yet \quad K^G(z) \leq O(log\ n).$$

That is, $K^E(z) - K^G(z)$ is unbounded.

    Proof. For $x \in B^*\backslash\{\lambda\}$ and universal EOM T define

$$\Xi(x) = \sum_{y \in B^\sharp : 0.y > 0.x} \quad \sum_{p:T(p)\rightsquigarrow y} 2^{-l(p)}.\tag{11}$$

The reader should not worry about the sum over uncountably many objects, because the dyadic expansion of $\Xi(x)$ is EOM-computable or c.e. The algorithm works as follows:

        Algorithm A: Initialize the real-valued variable V by 0, run all possible programs of EOM T dovetail style such that the n-th step of the i-th program is executed in the $n + i$-th phase; whenever the output of a program prefix q starts with some y satisfying $0.y > 0.x$ for the first time, set $V := V + 2^{-l(q)}$; henceforth ignore continuations of q.

V approximates $\Xi(x)$ from below in c.e. fashion — infinite p are not worrisome as T must only read a finite prefix of p to observe $0.y > 0.x$ if the latter holds indeed. We will now show that knowledge of $\Xi(x)_n$, the first n bits of $\Xi(x)$, allows for constructing a bitstring z with $K^E(z) \geq n - O(1)$ when x has low complexity.

    Suppose we know $\Xi(x)_n$. Once algorithm A above yields $V > \Xi(x)_n$ we know that no programs p with $l(p) < n$ will contribute any more to V. Choose the shortest z satisfying $0.z = (0.y_{min} - 0.x)/2$, where y_{min} is the lexicographically smallest y previously computed by algorithm A such that $0.y > 0.x$. Then z cannot be among the strings T-describable with fewer than n bits. Using the Invariance Theorem (compare Proposition 3.1) we obtain $K^E(z) \geq n - O(1)$.

    While prefixes of Ω are greatly compressible on EOMs, z is not. On the other hand, z is compactly G-describable: $K^G(z) \leq K(x) + K(n) + O(1)$. For instance, choosing a low-complexity x, we have $K^G(z) \leq O(K(n)) \leq O(log\ n)$.   □

    The above construction shows that the novel objects are "just as computable in the limit" as Ω, despite being more random. An interesting topic of future research

may be to identify the *most* random describable object, if there is one, which we doubt.

The discussion above also reveals a natural complexity hierarchy. Ignoring additive constants, we have

$$K^G(x) \leq K^E(x) \leq K^M(x), \tag{12}$$

where for each "$\leq$" relation above there are x which allow for replacing "$\leq$" by "$<$."

3.3. Which is the "True" Information Content of x?

Traditionally, the algorithmic information conveyed by x is identified with $K(x)$, the size of the shortest halting program of x. This makes a lot of sense, because we often want to limit ourselves to programs that at some point allow us to be sure that their output is complete. In the light of the results above, however, one might argue that the "true" information content of x is actually embodied by the smaller $K^G(x)$, the size of the smallest nonhalting GTM program for x. $K^G(x)$ is not computable in the limit (Theorem 3.1), while $K(x)$ is. K^E is between K and K^G in a certain sense — it is "closer" to K^G than K while being "just as computable in the limit" as K.

3.4. Relation to Conditional Complexity

For $x \in B^*$, the nonapproximable $K^G(x)$ can also be rewritten as $K^G(x) = K(x \mid \Omega)$ (Peter Gács, personal communication, 2001). What about the approximable and thus more accessible $K^E(x)$? It may be of interest to identify and examine the y in $K^E(x) = K(x \mid y)$.

4. Measures and Probability Distributions

We will now show how the Kolmogorov complexity hierarchy introduced above translates into an algorithmic prior hierarchy.

Suppose x represents the history of our universe up until now. What is its most likely continuation $y \in B^\sharp$? Bayes' theorem yields

$$P(xy \mid x) = \frac{P(x \mid xy)P(xy)}{\sum_{z \in B^\sharp} P(xz)} = \frac{P(xy)}{N(x)} \propto P(xy) \tag{13}$$

where $P(z^2 \mid z^1)$ is the probability of z^2, given knowledge of z^1, and

$$N(x) = \sum_{z \in B^\sharp} P(xz) \tag{14}$$

is a normalizing factor. The most likely continuation y is determined by $P(xy)$, the *prior probability* of xy. Now what are the formally describable ways of assigning prior probabilities to computable universes? In what follows we will first consider traditional describable *semimeasures* on B^*, then nontraditional probability distributions on $B^\sharp$.

4.1. Dominant and Universal (Semi)Measures

Traditionally, the algorithmic probability of $x \in B^*$ is defined as the probability of producing x by a halting program for a universal TM whose input bits are selected randomly. Since some of the possible input sequences cause nonhalting computations, the individual probabilities do not add up to 1. This and related issues inspired work on *semimeasures* [40, 45, 41, 17, 30] as opposed to classical *measures* considered in traditional statistics.

The next three definitions concerning semimeasures on B^* are almost but not quite identical to those of discrete semimeasures [30, p. 245 ff] and continuous semimeasures [30, p. 272 ff] based on the work of Levin and Zvonkin [45].

Definition 4.1 (Semimeasures) *A (binary) semimeasure μ is a function $B^* \rightarrow [0, 1]$ that satisfies:*

$$\mu(\lambda) = 1; \quad \mu(x) \geq 0; \quad \mu(x) = \mu(x0) + \mu(x1) + \bar{\mu}(x), \tag{15}$$

where $\bar{\mu}$ is a function $B^* \rightarrow [0, 1]$ satisfying $0 \leq \bar{\mu}(x) \leq \mu(x)$.

A notational difference to the approach of Levin [45] (who writes $\mu(x) \leq \mu(x0) + \mu(x1)$) is the explicit introduction of $\bar{\mu}$. Compare the introduction of an undefined element u by Li and Vitanyi [30, p. 281]. Note that $\sum_{x \in B^*} \bar{\mu}(x) \leq 1$. Later we will discuss the interesting case $\bar{\mu}(x) = P(x)$, the a priori probability of x.

Definition 4.2 (Dominant Semimeasures) *A semimeasure μ_0 dominates another semimeasure μ if there is a constant c_μ such that for all $x \in B^*$*

$$\mu_0(x) > c_\mu \mu(x). \tag{16}$$

Definition 4.3 (Universal Semimeasures) *Let $\mathcal{M}$ be a set of semimeasures on B^*. A semimeasure $\mu_0 \in \mathcal{M}$ is universal if it dominates all $\mu \in \mathcal{M}$.*

In what follows, we will introduce novel, nonenumerable but describable semimeasures dominating the c.e. ones considered in previous work [45][30, p. 245 ff, p.272 ff].

4.2. A Novel Universal Cumulatively Enumerable Measure (CEM)

The traditional universal enumerable measure [40, 45, 29, 16, 17, 41, 14, 30] studied in the theory of optimal inductive inference [40, 41, 24, 23] reflects properties of a universal MTM with random input. In what follows we will simply replace the MTM by a more expressive EOM and obtain a nonenumerable measure that (1) dominates the traditional measure, (2) is "just as computable" in the limit as the traditional one, (3) is universal in its class. It will turn out though that this approach does not generalize to the case of GTMs. Some new definitions are in order.

Definition 4.4 (Cumulative measure $C\mu$) *For semimeasure μ on B^* define the cumulative measure $C\mu$:*

$$C\mu(x) := \sum_{y \succeq x: \; l(y) = l(x)} \mu(y) + \sum_{y \succ x: \; l(y) < l(x)} \bar{\mu}(y). \tag{17}$$

13

Note that we could replace "$l(x)$" by "$l(x) + c$" in the definition above. Recall that x' denotes the smallest $y \succ x$ with $l(y) \leq l(x)$ (x' may be undefined). We have

$$\mu(x) = C\mu(x) \ if \ x = 11...1; \ \ else \ \ \mu(x) = C\mu(x) - C\mu(x'). \tag{18}$$

Definition 4.5 (CEMs) *Semimeasure μ is a CEM if $C\mu(x)$ is c.e. for all $x \in B^*$.*

Then $\mu(x)$ is the difference of two finite c.e. values, according to (18).

Theorem 4.1 *There is a universal CEM.*

   Proof.   We first show that one can enumerate the CEMs, then construct a universal CEM from the enumeration. Check out the differences to Levin's proofs that there is a universal discrete semimeasure and a universal c.e. semimeasure [45, 28], and Li and Vitányi's presentation of the latter [30, p. 273 ff], attributed to J. Tyszkiewicz.

   Without loss of generality, consider only EOMs without halt instruction and with fixed input encoding of B^* according to Def. 2.8. Such EOMs are c.e., and correspond to an effective enumeration of all c.e. functions from B^* to $B^\sharp$. Let EOM_i denote the i-th EOM in the list, and let $EOM_i(x, n)$ denote its output after n instructions when applied to $x \in B^*$. The following procedure filters out those EOM_i that already represent CEMs, and transforms the others into representations of CEMs, such that we obtain a way of generating all and only CEMs.

> FOR all i DO in dovetail fashion:
>
>> START: let $V\mu_i(x)$ and $V\bar{\mu}_i(x)$ and $VC\mu_i(x)$ denote variable functions on B^*. Set $V\mu_i(\lambda) := V\bar{\mu}_i(\lambda) := VC\mu_i(\lambda) := 1$, and $V\mu_i(x) := V\bar{\mu}_i(x) := VC\mu_i(x) := 0$ for all other $x \in B^*$. Define $VC\mu_i(x') := 0$ for undefined x'. Let z denote a string variable.
>>
>> FOR $n = 1, 2, \ldots$ DO:
>>
>>> **(1)** Lexicographically order and rename all x with $l(x) \leq n$:
>>> $x^1 := \lambda \prec x^2 := 0 \prec x^3 \prec \ldots \prec x^{2^{n+1}-1} := \underbrace{11...1}_{n}$.
>>>
>>> **(2)** FOR $k = 2^{n+1} - 1$ down to 1 DO:
>>>
>>>> **(2.1)** Systematically search for the smallest $m \geq n$ such that $z := EOM_i(x^k, m) \neq \lambda$ AND $0.z \geq VC\mu_i(x^{k+1})$ if $k < 2^{n+1} - 1$; set $VC\mu_i(x^k) := 0.z$.
>>>
>>> **(3)** For all $x \succ \lambda$ satisfying $l(x) \leq n$, set $V\mu_i(x) := VC\mu_i(x) - VC\mu_i(x')$. For all x with $l(x) < n$, set $V\bar{\mu}_i(x) := V\mu_i(x) - V\mu_i(x1) - V\mu_i(x0)$. For all x with $l(x) = n$, set $V\bar{\mu}_i(x) := V\mu_i(x)$.

14

If EOM_i indeed represents a CEM μ_i then each search process in **(2.1)** will terminate, and the $VC\mu_i(x)$ will enumerate the $C\mu_i(x)$ from below, and the $V\mu_i(x)$ and $V\bar{\mu}_i(x)$ will approximate the true $\mu_i(x)$ and $\bar{\mu}_i(x)$, respectively, not necessarily from below though. Otherwise there will be a nonterminating search at some point, leaving $V\mu_i$ from the previous loop as a trivial CEM. Hence we can enumerate all CEMs, and only those. Now define (compare [28]):

$$\mu_0(x) = \sum_{n>0} \alpha_n \mu_n(x), \quad \bar{\mu}_0(x) = \sum_{n>0} \alpha_n \bar{\mu}_n(x), \quad where\ \alpha_n > 0,\ \sum_n \alpha_n = 1,$$

and α_n is a c.e. constant, e.g., $\alpha_n = \frac{6}{\pi n^2}$ or $\alpha_n = \frac{1}{n(n+1)}$ (note a slight difference to Levin's approach which just requests $\sum_n \alpha_n \leq 1$). Then μ_0 dominates every μ_n by Def. 16, and is a semimeasure according to Def. 4.1: $\mu_0(\lambda) = 1$; $\mu_0(x) \geq 0$; and

$$\mu_0(x) = \sum_{n>0} \alpha_n [\mu_n(x0) + \mu_n(x1) + \bar{\mu}_n(x)] = \mu_0(x0) + \mu_0(x1) + \bar{\mu}_0(x). \tag{19}$$

μ_0 also is a CEM by Def. 4.5, because

$$C\mu_0(x) = \sum_{y \succeq x:\ l(x)=l(y)}\ \sum_{n>0} \alpha_n \mu_n(y)\ +\ \sum_{y \succ x:\ l(x)>l(y)}\ \sum_{n>0} \alpha_n \bar{\mu}_n(y) =$$

$$\sum_{n>0} \alpha_n \left(\sum_{y \succeq x:\ l(x)=l(y)} \mu_n(y) + \sum_{y \succ x:\ l(x)>l(y)} \bar{\mu}_n(y) \right) = \sum_{n>0} \alpha_n C\mu_n(x) \tag{20}$$

is c.e., since α_n and $C\mu_n(x)$ are (dovetail over all n). That is, $\mu_0(x)$ is approximable as the difference of two c.e. finite values, according to Equation (18). □

4.3. Approximable and Cumulatively Enumerable Distributions

To deal with infinite x, we will now extend the treatment of semimeasures on B^* to nontraditional probability distributions on $B^\sharp$.

Definition 4.6 (Probabilities) *A probability distribution P on $x \in B^\sharp$ satisfies*

$$P(x) \geq 0; \quad \sum_x P(x) = 1.$$

Definition 4.7 (Semidistributions) *A semidistribution P on $x \in B^\sharp$ satisfies*

$$P(x) \geq 0; \quad \sum_x P(x) \leq 1.$$

Definition 4.8 (Dominant Distributions) *A distribution P_0 dominates another distribution P if there is a constant $c_P > 0$ such that for all $x \in B^\sharp$:*

$$P_0(x) \geq c_P P(x). \tag{21}$$

Definition 4.9 (Universal Distributions) *Let $\mathcal{P}$ be a set of probability distributions on $x \in B^\sharp$. A distribution $P_0 \in \mathcal{P}$ is universal if for all $P \in \mathcal{P}$: P_0 dominates P.*

15

Unfortunately it turns out that the analogue of the universal CEM Theorem 4.1 for EOMs with random input does not hold for GTMs:

Theorem 4.2 *There is no universal approximable semidistribution.*

Proof. The following proof is due to M. Hutter (personal communications by email following a discussion of approximable universes on 2 August 2000 in Munich). It is an extension of a modified[a] proof [30, p. 249 ff] that there is no universal recursive semimeasure.

It suffices to focus on $x \in B^*$. Identify strings with integers, and assume $P(x)$ is a universal approximable semidistribution. We construct an approximable semidistribution $Q(x)$ that is not dominated by $P(x)$, thus contradicting the assumption. Let $P_0(x), P_1(x), \ldots$ be a sequence of recursive functions converging to $P(x)$. We recursively define a sequence $Q_0(x), Q_1(x), \ldots$ converging to $Q(x)$. The basic idea is: each contribution to Q is the sum of n consecutive P probabilities (n increasing). Define $Q_0(x) := 0$; $I_n := \{y : n^2 \leq y < (n+1)^2\}$. Let n be such that $x \in I_n$. Define j_t^n (k_t^n) as the element with smallest P_t (largest Q_{t-1}) probability in this interval, i.e., $j_t^n := \mathrm{minarg}_{x \in I_n} P_t(x)$ $(k_t^n := \mathrm{maxarg}_{x \in I_n} Q_{t-1}(x))$. If $n \cdot P_t(k_t^n)$ is less than twice and $n \cdot P_t(j_t^n)$ is more than half of $Q_{t-1}(k_t^n)$, set $Q_t(x) = Q_{t-1}(x)$. Otherwise set $Q_t(x) = n \cdot P_t(j_t^n)$ for $x = j_t^n$ and $Q_t(x) = 0$ for $x \neq j_t^n$. $Q_t(x)$ is obviously total recursive and non-negative. Since $2n \leq |I_n|$, we have

$$\sum_{x \in I_n} Q_t(x) \leq 2n \cdot P_t(j_t^n) = 2n \cdot \min_{x \in I_n} P_t(x) \leq \sum_{x \in I_n} P_t(x).$$

Summing over n we observe that if P_t is a semidistribution, so is Q_t. From some t_0 on, $P_t(x)$ changes by less than a factor of 2 since $P_t(x)$ converges to $P(x) > 0$. Hence $Q_t(x)$ remains unchanged for $t \geq t_0$ and converges to $Q(x) := Q_\infty(x) = Q_{t_0}(x)$. But $Q(j_{t_0}^n) = Q_{t_0}(j_{t_0}^n) \geq n \cdot P_{t_0}(j_{t_0}^n) \geq \frac{1}{2} n \cdot P(j_{t_0}^n)$, violating our universality assumption $P(x) \geq c \cdot Q(x)$. ☐

Definition 4.10 (Cumulatively Enumerable Distributions – CEDs) *A distribution P on $B^\sharp$ is a CED if $CP(x)$ is c.e. for all $x \in B^*$, where*

$$CP(x) := \sum_{y \in B^\sharp : y \succeq x} P(y) \tag{22}$$

4.4. TM-Induced Distributions and Convergence Probability

Suppose TM T's input bits are obtained by tossing an unbiased coin whenever a new one is requested. Levin's *universal discrete enumerable semimeasure* [28, 14, 16] or *semidistribution m* is limited to B^* and halting programs:

---

[a] As pointed out by M. Hutter (14 Nov. 2000, personal communication) and even earlier by A. Fujiwara (1998, according to P. M. B. Vitányi, personal communication, 21 Nov. 2000), the proof on the bottom of p. 249 of [30] should be modified. For instance, the sum could be taken over $x_{i-1} < x \leq x_i$. The sequence of inequalities $\sum_{x_{i-1} < x \leq x_i} P(x) > x_i P(x_i)$ is then satisfiable by a suitable x_i sequence, since $\liminf_{x \to \infty} \{x P(x)\} = 0$. The basic idea of the proof is correct, though, and quite useful.

Definition 4.11 (m)

$$m(x) = \sum_{p:T(p)=x} 2^{-l(p)};\qquad(23)$$

Note that $\sum_x m(x) < 1$ if T universal. We will now generalize this in obvious but nontraditional ways to $B^\sharp$ and nonhalting programs for MTMs, EOMs, and GTMs.

Definition 4.12 (P_T, KP_T) *Suppose T's input bits are obtained by tossing an unbiased coin whenever a new one is requested.*

$$P_T(x) = \sum_{p:T(p)\rightsquigarrow x} 2^{-l(p)},\quad KP_T(x) = -lgP_T(x)\ for\ P_T(x) > 0,\qquad(24)$$

where $x, p \in B^\sharp$.

Program Continua. According to Def. 4.12, most infinite x have zero probability, but not those with finite programs, such as the dyadic expansion of $0.5\sqrt{2}$. However, a nonvanishing part of the entire unit of probability mass is contributed by continua of mostly incompressible strings, such as those with cumulative probability $2^{-l(q)}$ computed by the following class of uncountably many infinite programs with a common finite prefix q: "repeat forever: read and print next input bit." The corresponding traditional measure-oriented notation for

$$\sum_{x:T(qx)\rightsquigarrow x} 2^{-l(qx)} = 2^{-l(q)}$$

would be

$$\int_{0.q}^{0.q+2^{-l(q)}} dx = 2^{-l(q)}.$$

For notational simplicity, however, we will continue using the $\sum$ sign to indicate summation over uncountable objects, rather than using a measure-oriented notation for probability densities. The reader should not worry about this — the theorems in the remainder of the paper will focus on those $x \in B^\sharp$ with $P(x) > 0$; density-like nonzero sums over uncountably many bitstrings, each with individual measure zero, will not play any critical role in the proofs.

Definition 4.13 (Universal TM-Induced Distributions $P^C; KP^C$) *If C denotes a set of TMs with universal element U^C, then we write*

$$P^C(x) = P_{U^C}(x);\quad KP^C(x) := -lg\ P^C(x)\ for\ P^C(x) > 0.\qquad(25)$$

We have $P^C(x) > 0$ for $D_C \subset B^\sharp$, the subset of C-describable $x \in B^\sharp$. The attribute *universal* is justified, because of the dominance $P_T(x) = O(P^C(x))$, due to the Invariance Theorem (compare Proposition 3.1).

Since all programs of EOMs and MTMs converge, P^E and P^M are proper probability distributions on $B^\sharp$. For instance, $\sum_x P^E(x) = 1$. P^G, however, is just a semidistribution. To obtain a proper probability distribution PN_T, one might think of normalizing by the *convergence probability* Υ:

Definition 4.14 (Convergence Probability) *Given GTM T, define*

$$PN_T(x) = \frac{\sum_{T(p) \leadsto x} 2^{-l(p)}}{\Upsilon^T},$$

where

$$\Upsilon^T = \sum_{p:\exists x:T(p) \leadsto x} 2^{-l(p)}.$$

Υ^T **not describable.** Uniquely encode each TM T as a finite bitstring, and identify M, E, G with the corresponding sets of bitstrings. While the function $f^M : M \to B^\sharp : f(T) = \Omega^T$ is c.e., the function $f^G : G \to B^\sharp : f(T) = \Upsilon^T$ is not even describable, essentially due to Theorem 2.1.

Even $P^E(x)$ and $P^M(x)$ are generally not describable for $x \in B^\sharp$, in the sense that there is no GTM T that takes as an input a finite description (or program) of any M-describable or E-describable $x \in B^\sharp$ and converges towards $P^M(x)$ or $P^E(x)$. This is because in general it is not even weakly decidable (Def. 2.11) whether two programs compute the same output. If we know that one of the program outputs is finite, however, then the conditions of weak decidability are fulfilled. Hence certain TM-induced distributions on B^* are describable, as will be seen next.

Definition 4.15 (TM-Induced Cumulative Distributions) *If C denotes a set of TMs with universal element U^C, then we write (compare Def. 4.10):*

$$CP^C(x) = CP_{U^C}(x). \tag{26}$$

Lemma 4.1 *For $x \in B^*$, $CP^E(x)$ is c.e.*

Proof. The following algorithm computes $CP^E(x)$ (compare proof of Theorem 3.3):

> Initialize the real-valued variable V by 0, run all possible programs of EOM T dovetail style; whenever the output of a program prefix q starts with some $y \succeq x$ for the first time, set $V := V + 2^{-l(q)}$; henceforth ignore continuations of q.

In this way V enumerates $CP^E(x)$. Infinite p are not problematic as only a finite prefix of p must be read to establish $y \succeq x$ if the latter indeed holds. $\square$

Similarly, facts of the form $y \succ x \in B^*$ can be discovered after finite time.

Corollary 4.1 *For $x \in B^*$, $P^E(x)$ is approximable or describable as the difference of two c.e. values:*

$$P^E(x) = \sum_{y \succeq x} P^E(y) - \sum_{y \succ x} P^E(y), \tag{27}$$

Now we will make the connection to the previous subsection on semimeasures on B^*.

4.5. Universal TM-Induced Measures

Definition 4.16 (P-Induced Measure μP) *Given a distribution P on $B^{\sharp}$, define a measure μP on B^* as follows:*

$$\mu P(x) = \sum_{z \in B^{\sharp}} P(xz). \tag{28}$$

Note that $\overline{\mu P}(x) = P(x)$ (compare Def. 4.1):

$$\mu P(\lambda) = 1; \quad \mu P(x) = P(x) + \mu P(x0) + \mu P(x1). \tag{29}$$

For those $x \in B^*$ without 0-bit we have $\mu P(x) = CP(x)$, for the others

$$\mu P(x) = CP(x) - CP(x'). \tag{30}$$

Definition 4.17 (TM-Induced Semimeasures $\mu_T, \mu^M, \mu^E, \mu^G$) *Given some TM T, for $x \in B^*$ define $\mu_T(x) = \mu P_T(x)$. Again we deviate a bit from Levin's B^*-oriented path [45] (survey: [30, p. 245 ff, p. 272 ff]) and extend μ_T to $x \in B^{\infty}$, where we define $\mu_T(x) = \bar{\mu}_T(x) = P_T(x)$. If C denotes a set of TMs with universal element U^C, then we write*

$$\mu^C(x) = \mu_{U^C}(x); \quad K\mu^C(x) := -lg \ \mu^C(x) \ for \ \mu^C(x) > 0. \tag{31}$$

We observe that μ^C is universal among all T-induced semimeasures, $T \in C$. Note that

$$\mu^C(x) = \mu^C(x0) + \mu^C(x1) + P^C(x) \ for \ x \in B^*; \quad \mu^C(x) = P^C(x) \ for \ x \in B^{\infty}. \tag{32}$$

It will be obvious from the context when we deal with the restriction of μ^C to B^*.

Corollary 4.2 *For $x \in B^*$, $\mu^E(x)$ is a CEM and approximable as the difference of two c.e. values: $\mu^E(x) = CP^E(x)$ for x without any 0-bit, otherwise*

$$\mu^E(x) = CP^E(x) - CP^E(x'). \tag{33}$$

4.6. Universal CEM vs EOM with Random Input

Corollary 4.3 and Lemma 4.2 below imply that μ^E and μ_0 are essentially the same thing: randomly selecting the inputs of a universal EOM yields output prefixes whose probabilities are determined by the universal CEM.

Corollary 4.3 *Let μ_0 denote the universal CEM of Theorem 4.1. For $x \in B^*$,*

$$\mu^E(x) = O(\mu_0(x)).$$

Lemma 4.2 *For $x \in B^*$,*

$$\mu_0(x) = O(\mu^E(x)).$$

19

Proof. In the enumeration of EOMs in the proof of Theorem 4.1, let EOM_0 be an EOM representing μ_0. We build an EOM T such that $\mu_T(x) = \mu_0(x)$. The rest follows from the Invariance Theorem (compare Proposition 3.1).

T applies EOM_0 to all $x \in B^*$ in dovetail fashion, and simultaneously simply reads randomly selected input bits forever. At a given time, let string variable z denote T's input string read so far. Starting at the right end of the unit interval $[0,1)$, as the $V\bar{\mu}_0(x)$ are being updated by the algorithm of Theorem 4.1, T keeps updating a chain of finitely many, variable, disjoint, consecutive, adjacent, half-open intervals $VI(x)$ of size $V\bar{\mu}_0(x)$ in alphabetic order on x, such that $VI(y)$ is to the right of $VI(x)$ if $y \succ x$. After every variable update and each increase of z, T replaces its output by the x of the $VI(x)$ with $0.z \in VI(x)$. Since neither z nor the $VC\mu_0(x)$ in the algorithm of Theorem 4.1 can decrease (that is, all interval boundaries can only shift left), T's output cannot either, and therefore is indeed EOM-computable. Obviously the following holds:

$$C\mu P_T(x) = CP_T(x) = C\mu_0(x)$$

and

$$\mu P_T(x) = \sum_{z \in B^\sharp} P_T(xz) = \mu_0(x).$$

$\square$

Summary. The traditional universal c.e. measure [40, 45, 29, 16, 17, 41, 14, 30] derives from universal MTMs with random input. What is the nature of our novel generalization here? We simply replace the MTMs by EOMs. As shown above, this leads to universal cumulatively enumerable measures. In general these are not c.e. any more, but they are "just as computable" in the limit as the c.e. ones — we gain power and generality without leaving the constructive realm and without giving up the concept of universality. The even more dominant approximable measures, however, lack universality.

5. Probability vs Descriptive Complexity

The size of some computable object's minimal description is closely related to the object's probability. This is illustrated by Levin's *Coding Theorem* [29] for the universal discrete enumerable semimeasure m based on halting programs (see Def. 4.11); compare independent work by Chaitin [14] who also gives credit to N. Pippenger:

Theorem 5.1 (Coding Theorem)

$$For\ x \in B^*,\ -log\ m(x) \leq K(x) \leq -log\ m(x) + O(1) \tag{34}$$

In this special case, the contributions of the shortest programs dominate the probabilities of objects computable in the traditional sense. As shown by Gács [17] for the case of MTMs, however, contrary to Levin's [28] conjecture, $\mu^M(x) \neq O(2^{-Km^M(x)})$; but a slightly worse bound does hold:

20

Theorem 5.2

$$K\mu^M(x) - 1 \le Km^M(x) \le K\mu^M(x) + Km^M(K\mu^M(x)) + O(1). \qquad (35)$$

The term -1 on the left-hand side stems from the definition of $lg(x) \le log(x)$. We will now consider the case of probability distributions that dominate m, and semimeasures that dominate μ^M, starting with the case of c.e. objects.

5.1. Novel Theorems for EOMs and GTMs

Theorem 5.3 *For $x \in B^\sharp$ with $P^E(x) > 0$,*

$$KP^E(x) - 1 \le K^E(x) \le KP^E(x) + K^E(KP^E(x)) + O(1). \qquad (36)$$

Using $K^E(y) \le log\ y + 2log\ log\ y + O(1)$ for y interpreted as an integer — compare Def. 2.8 — this yields

$$2^{-K^E(x)} < P^E(x) \le O(2^{-K^E(x)})(K^E(x))^2. \qquad (37)$$

That is, objects that are hard to describe (in the sense that they have only long enumerating descriptions) have low probability.

    Proof. The left-hand inequality follows by definition. To show the right-hand side, one can build an EOM T that computes $x \in B^\sharp$ using not more than $KP^E(x) + K_T(KP^E(x)) + O(1)$ input bits in a way inspired by Huffman-Coding [22]. The claim then follows from the Invariance Theorem. The trick is to arrange T's computation such that T's output converges yet never needs to decrease lexicographically. T works as follows:

    (A) Emulate U^E to construct a real c.e. number $0.s$ encoded as a self-delimiting input program r, simultaneously run all (possibly forever running) programs on U^E dovetail style; whenever the output of a prefix q of any running program starts with some $x \in B^*$ for the first time, set variable $V(x) := V(x) + 2^{-l(q)}$ (if no program has ever created output starting with x then first create $V(x)$ initialized by 0); whenever the output of some extension q' of q (obtained by possibly reading additional input bits: $q' = q$ if none are read) lexicographically increases such that it does not equal x any more, set $V(x) := V(x) - 2^{-l(q')}$.

    (B) Simultaneously, starting at the right end of the unit interval $[0, 1)$, as the $V(x)$ are being updated, keep updating a chain of disjoint, consecutive, adjacent, half-open (at the right end) intervals $IV(x) = [LV(x), RV(x))$ of size $V(x) = RV(x) - LV(x)$ in alphabetic order on x, such that the right end of the $IV(x)$ of the largest x coincides with the right end of $[0, 1)$, and $IV(y)$ is to the right of $IV(x)$ if $y \succ x$. After every variable update and each change of s, replace the output of T by the x of the $IV(x)$ with $0.s \in IV(x)$.

This will never violate the EOM constraints: the c.e. s cannot shrink, and since EOM outputs cannot decrease lexicographically, the interval boundaries $RV(x)$ and $LV(x)$ cannot grow (their negations are c.e., compare Lemma 4.1), hence T's output cannot decrease.

For $x \in B^*$ the $IV(x)$ converge towards an interval $I(x)$ of size $P^E(x)$. For $x \in B^\infty$ with $P^E(x) > 0$, we have: for any $\epsilon > 0$ there is a time t_0 such that for all time steps $t > t_0$ in T's computation, an interval $I_\epsilon(x)$ of size $P^E(x) - \epsilon$ will be completely covered by certain $IV(y)$ satisfying $x \succ y$ and $0.x - 0.y < \epsilon$. So for $\epsilon \to 0$ the $I_\epsilon(x)$ also converge towards an interval $I(x)$ of size $P^E(x)$. Hence T will output larger and larger y approximating x from below, provided $0.s \in I(x)$.

Since any interval of size c within $[0,1)$ contains a number $0.z$ with $l(z) = -lg\, c$, in both cases there is a number $0.s$ (encodable by some r satisfying $r \leq l(s) + K_T(l(s)) + O(1)$)) with $l(s) = -lgP^E(x) + O(1)$, such that $T(r) \rightsquigarrow x$, and therefore $K_T(x) \leq l(s) + K_T(l(s)) + O(1)$. $\qquad\square$

Less symmetric statements can also be derived in very similar fashion:

Theorem 5.4 *Let TM T induce approximable $CP_T(x)$ for all $x \in B^*$ (compare Defs. 4.10 and 4.12; an EOM would be a special case). Then for $x \in B^\sharp$, $P_T(x) > 0$:*

$$K^G(x) \leq KP_T(x) + K^G(KP_T(x)) + O(1). \tag{38}$$

Proof. Modify the proof of Theorem 5.3 for approximable as opposed to c.e. interval boundaries and approximable $0.s$. $\qquad\square$

A similar proof, but without the complication for the case $x \in B^\infty$, yields:

Theorem 5.5 *Let μ denote an approximable semimeasure on $x \in B^*$; that is, $\mu(x)$ is describable. Then for $\mu(x) > 0$:*

$$Km^G(x) \leq K\mu(x) + Km^G(K\mu(x)) + O(1); \tag{39}$$

$$K^G(x) \leq K\bar{\mu}(x) + K^G(K\bar{\mu}(x)) + O(1). \tag{40}$$

As a consequence,

$$\frac{\mu(x)}{K\mu(x)log^2 K\mu(x)} \leq O(2^{-Km^G(x)}); \quad \frac{\bar{\mu}(x)}{K\bar{\mu}(x)log^2 K\bar{\mu}(x)} \leq O(2^{-K^G(x)}). \tag{41}$$

Proof. Initialize variables $V_\lambda := 1$ and $IV_\lambda := [0,1)$. Dovetailing over all $x \succ \lambda$, approximate the GTM-computable $\bar{\mu}(x) = \mu(x) - \mu(x0) - \mu(x1)$ in variables V_x initialized by zero, and create a chain of adjacent intervals IV_x analogously to the proof of Theorem 5.3.

The IV_x converge against intervals I_x of size $\bar{\mu}(x)$. Hence x is GTM-encodable by any program r producing an output s with $0.s \in I_x$: after every update, replace the GTM's output by the x of the IV_x with $0.s \in IV_x$. Similarly, if $0.s$ is in the union of adjacent intervals I_y of strings y starting with x, then the GTM's output will converge towards some string starting with x. The rest follows in a way similar to the one described in the final paragraph of the proof of Theorem 5.3. $\qquad\square$

Using the basic ideas in the proofs of Theorem 5.3 and 5.5 in conjunction with Corollary 4.3 and Lemma 4.2, one can also obtain statements such as:

22

Theorem 5.6 *Let μ_0 denote the universal CEM from Theorem 4.1. For $x \in B^*$,*

$$K\mu_0(x) - O(1) \le Km^E(x) \le K\mu_0(x) + Km^E(K\mu_0(x)) + O(1) \qquad (42)$$

While P^E dominates P^M and P^G dominates P^E, the reverse statements are not true. In fact, given the results from Sections 3.2 and 5, one can now make claims such as the following ones:

Corollary 5.1 *The following functions are unbounded:*

$$\frac{\mu^E(x)}{\mu^M(x)}; \quad \frac{P^E(x)}{P^M(x)}; \quad \frac{P^G(x)}{P^E(x)}.$$

Proof. For the cases μ^E and P^E, apply Theorems 5.2, 5.6 and the unboundedness of (10). For the case P^G, apply Theorems 3.3 and 5.3. □

5.2. Tighter Bounds?

Is it possible to get rid of the small correction terms such as $K^E(KP^E(x)) \le O(log(-logP^E(x)))$ in Theorem 5.3? Note that the construction in the proof shows that $K^E(x)$ is actually bounded by $K^E(s)$, the complexity of the c.e. number $0.s \in I(x)$ with minimal $K_T(s)$. The facts $\sum_x P^M(x) = 1$, $\sum_x P^E(x) = 1$, $\sum_x P^G(x) < 1$, as well as intuition and wishful thinking inspired by Shannon-Fano Theorem [38] and Coding Theorem 5.1 suggest there might indeed be tighter bounds:

Conjecture 5.1 *For $x \in B^\sharp$ with $P^M(x) > 0$: $P^M(x) = O(2^{-K^M(x)})$.*
Conjecture 5.2 *For $x \in B^\sharp$ with $P^E(x) > 0$: $P^E(x) = O(2^{-K^E(x)})$.*
Conjecture 5.3 *For $x \in B^\sharp$ with $P^G(x) > 0$: $P^G(x) = O(2^{-K^G(x)})$.*

Gács has shown though that analogue conjectures for semimeasures such as μ^M on B^* (as opposed to distributions on $B^\sharp$) are false [17].

5.3. Between EOMs and GTMs?

The dominance of P^G over P^E comes at the expense of occasionally "unreasonable," nonconverging outputs. Are there classes of always converging TMs more expressive than EOMs? Consider a TM called a PEOM whose inputs are pairs of finite bitstrings $x, y \in B^*$ (code them using $2log\ l(x) + 2log\ l(y) + l(xy) + O(1)$ bits). The PEOM uses dovetailing to run all self-delimiting programs on the y-th EOM of an enumeration of all EOMs, to approximate the probability $PEOM(y, x)$ (again encoded as a string) that the EOM's output starts with x. $PEOM(y, x)$ is approximable (we may apply Theorem 5.5) but not necessarily c.e. On the other hand, it is easy to see that PEOMs can compute all c.e. strings describable on EOMs. In this sense PEOMs are more expressive than EOMs, yet never diverge like GTMs. EOMs can encode some c.e. strings slightly more compactly, however, due to the PEOM's possibly unnecessarily bit-consuming input encoding. An interesting topic of future research may be to establish a partially ordered expressiveness hierarchy among classes of always converging TMs, and to characterize its top, if there is

23

one, which we doubt. Candidates to consider may include TMs that approximate certain recursive or c.e. functions of c.e. strings.

6. Consequences for Physics

Is the entire past and future history of our universe describable by a finite sequence of bits, just like a movie stored on a compact disc, or a never ending evolution of a virtual reality determined by a finite algorithm, such as in Example 2.1? Contrary to a widely spread misunderstanding, quantum physics, quantum computation (e.g., [3]) and Heisenberg's uncertainty principle do not rule this out [35]. In absence of contrarian evidence we might assume our universe is formally describable indeed, or at least sampled from a formally describable distribution — if this assumption is false, then our world will forever remain beyond formal understanding.

As obvious from equation (13), the future of some observer evolving within such a universe depends on this prior distribution. More or less general notions of TM-based describability put forward above lead to more or less dominant priors such as P^G on formally describable universes, P^E and μ^E on enumerable universes, P^M and μ^M and recursive priors on monotonically computable universes. Generally speaking, the theorems above show that any future corresponding to a history without any short description (given the appropriate TM type) is necessarily unlikely. To a certain extent, this justifies "Occam's razor" (e.g., [4]) which expresses the ancient preference of simple solutions over complex ones. A more detailed analysis can be found elsewhere [36].

Acknowledgments

References

1. Y. M. Barzdin. Algorithmic information theory. In *Encyclopaedia of Mathematics*, volume 1, pages 140–142. Reidel, Kluwer Academic Publishers, 1988.

2. M. Beeson. *Foundations of Constructive Mathematics*. Springer-Verlag, Heidelberg, 1985.

3. C. H. Bennett and D. P. DiVicenzo. Quantum information and computation. *Nature*, 404(6775):256–259, 2000.

4. A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Occam's razor. *Information Processing Letters*, 24:377–380, 1987.

5. L. E. J. Brouwer. Over de Grondslagen der Wiskunde. Dissertation, Doctoral Thesis, University of Amsterdam, 1907.

6. M. S. Burgin. Inductive Turing machines. *Notices of the Academy of Sciences of the USSR (translated from Russian)*, 270(6):1289–1293, 1991.

7. M. S. Burgin. Theory of super-recursive algorithms as a source of a new paradigm

for computer simulation. In *Proceedings of the Business and Industry Simulation Symposium*, pages 70–75. Washington, 2000.

8. M. S. Burgin and Y. M. Borodyanskii. Infinite processes and super-recursive algorithms. *Notices of the Academy of Sciences of the USSR (translated from Russian)*, 321(5):800–803, 1991.

9. C. S. Calude. Chaitin Ω numbers, Solovay machines and Gödel incompleteness. *Theoretical Computer Science*, 2001. In press.

10. C. S. Calude, P. H. Hertling, and B. Khoussainov. Recursively enumerable reals and Chaitin Ω numbers. In M. Morvan et al., editor, *Lecture Notes Comp. Sci. 1373, 15th annual symposium on theoretical aspects of computer science, STACS 98*, pages 596–606. Springer, Berlin, 1998.

11. G. Cantor. Über eine Eigenschaft des Inbegriffes aller reellen algebraischen Zahlen. *Crelle's Journal für Mathematik*, 77:258–263, 1874.

12. T. Chadzelek and G. Hotz. Analytic machines. *Theoretical Computer Science*, 219:151–167, 1999.

13. G.J. Chaitin. On the length of programs for computing finite binary sequences: statistical considerations. *Journal of the ACM*, 16:145–159, 1969. Submitted 1965.

14. G.J. Chaitin. *Algorithmic Information Theory*. Cambridge University Press, Cambridge, 1987.

15. R. V. Freyvald. Functions and functionals computable in the limit. *Transactions of Latvijas Vlasts Univ. Zinatn. Raksti*, 210:6–19, 1977.

16. P. Gács. On the symmetry of algorithmic information. *Soviet Math. Dokl.*, 15:1477–1480, 1974.

17. P. Gács. On the relation between descriptional complexity and algorithmic probability. *Theoretical Computer Science*, 22:71–93, 1983.

18. K. Gödel. Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I. *Monatshefte für Mathematik und Physik*, 38:173–198, 1931.

19. E. M. Gold. Limiting recursion. *Journal of Symbolic Logic*, 30(1):28–46, 1965.

20. A. Gregorczyk. On the definitions of computable real continuous functions. *Fundamenta Mathematicae*, 44:61–71, 1957.

21. G. Hotz, G. Vierke, and B. Schieffer. Analytic machines. Technical Report TR95-025, Electronic Colloquium on Computational Complexity, 1995. http://www.eccc.uni-trier.de/eccc/.

22. D. A. Huffman. A method for construction of minimum-redundancy codes. *Proceedings IRE*, 40:1098–1101, 1952.

23. M. Hutter. General loss bounds for universal sequence prediction. In *Proceedings of the 18th International Conference on Machine Learning (ICML-2001)*. Springer, 2001. TR IDSIA-03-01, IDSIA, Switzerland, Jan 2001, cs.AI/0101019.

24. M. Hutter. New error bounds for Solomonoff prediction. *Journal of Computer and System Science*, 62(4):653–667, 2001. http://xxx.lanl.gov/abs/cs.AI/9912008.

25. A.N. Kolmogorov. Three approaches to the quantitative definition of information. *Problems of Information Transmission*, 1:1–11, 1965.

26. A. Kucera. On relative randomness. *Ann. Pure Appl. Logic*, 63(1):61–67, 1993.

27. S. A. Kurtz. On the random oracle hypothesis. *Inform. and Control*, 57:40–47, 1983.

28. L. A. Levin. On the notion of a random sequence. *Soviet Math. Dokl.*, 14(5):1413–

1416, 1973.

29. L. A. Levin. Laws of information (nongrowth) and aspects of the foundation of probability theory. *Problems of Information Transmission*, 10(3):206–210, 1974.

30. M. Li and P. M. B. Vitányi. *An Introduction to Kolmogorov Complexity and its Applications (2nd edition)*. Springer, 1997.

31. P. Martin-Löf. The definition of random sequences. *Information and Control*, 9:602–619, 1966.

32. A. Mostowski. On computable sequences. *Fundamenta Mathematicae*, 44:37–51, 1957.

33. H. Putnam. Trial and error predicates and the solution to a problem of Mostowski. *Journal of Symbolic Logic*, 30(1):49–57, 1965.

34. H. Rogers, Jr. *Theory of Recursive Functions and Effective Computability*. McGraw-Hill, New York, 1967.

35. J. Schmidhuber. A computer scientist's view of life, the universe, and everything. In C. Freksa, M. Jantzen, and R. Valk, editors, *Foundations of Computer Science: Potential - Theory - Cognition*, volume 1337, pages 201–208. Lecture Notes in Computer Science, Springer, Berlin, 1997. Submitted 1996.

36. J. Schmidhuber. Algorithmic theories of everything. Technical Report IDSIA-20-00, quant-ph/0011122, IDSIA, Manno (Lugano), Switzerland, 2000.

37. C. P. Schnorr. Process complexity and effective random tests. *Journal of Computer Systems Science*, 7:376–388, 1973.

38. C. E. Shannon. A mathematical theory of communication (parts I and II). *Bell System Technical Journal*, XXVII:379–423, 1948.

39. T. Slaman. Randomness and recursive enumerability. Technical report, Univ. of California, Berkeley, 1999. Preprint, http://www.math.berkeley.edu/~slaman.

40. R.J. Solomonoff. A formal theory of inductive inference. Part I. *Information and Control*, 7:1–22, 1964.

41. R.J. Solomonoff. Complexity-based induction systems. *IEEE Transactions on Information Theory*, IT-24(5):422–432, 1978.

42. R. M. Solovay. A version of Ω for which ZFC can not predict a single bit. In C. S. Calude and G. Păun, editors, *Finite Versus Infinite. Contributions to an Eternal Dilemma*, pages 323–334. Springer, London, 2000.

43. A. M. Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society, Series 2*, 41:230–267, 1936.

44. V. V. V'yugin. Non-stochastic infinite and finite sequences. *Theoretical Computer Science*, 207(2):363–382, 1998.

45. A. K. Zvonkin and L. A. Levin. The complexity of finite objects and the algorithmic concepts of information and randomness. *Russian Math. Surveys*, 25(6):83–124, 1970.