Notes on Algorithmic Information Theory

Michael Weiss

November 10, 2018

1 Introduction

These notes originated as a reworking of some of the material in Chaitin's book [4], with some extra odds-and-ends and some simplifications. I probably wrote them around 1990. The interested reader should also consult the article by Ming Li and Paul M. B. Vitányi [10]. Li and Vitányi have also written a textbook [11] on the subject. There are also various survey articles; consult the bibliographies of the above references.

Li and Vitányi's article contains a treatment of the genesis of algorithmic information theory; if you want to delve into the history, that's one place to look. Another useful reference is the review by Peter Gács of Chaitin's book [6]. These notes do not attempt to give the history of the subject (I plead lack of expertise); my few brief remarks in this vein are based on Li and Vitányi's article, and Gács' review.

Three aspects of the theory strike me particularly. First, one can define an entropy function for finite bit strings, $H(x)$, which shares many of the formal properties of the entropy functions of physics and communication theory. For example, there is a probability distribution P such that

$H(x) = -\log P(x) + O(1)$. (The $O(1)$ notation is defined at the end of this introduction.) Next, one can give a precise definition for the concept "random infinite bit string". In fact, one can give rather different looking definitions which turn out be equivalent; the equivalence seems "deep". Finally, we have an analog of the halting problem: loosely speaking, what is the probability that a randomly chosen Turing machine halts? The binary expansion of this probability (denoted Ω by Chaitin) is random.

Section 2 defines the algorithmic entropy function H and proves simple properties about it. Section 3 gives the Coding Lemma, a key technical result that stems directly from analogous results in classical (i.e., non-algorithmic) information theory. Section 4 gives various definitions of the notion "random infinite bit string", and proves their equivalence. Ω is also defined and shown to be random. Finally, Section 5 develops the analogy between the algorithmic H and the classical H.

Note on the $O(1)$ notation: an equation of the form $f(\ldots) = g(\ldots) + O(1)$ means that difference of the two sides is bounded in absolute value, independently of any free variables occurring in the expressions $f(\ldots)$ and $g(\ldots)$. Formally:

$$(\exists c)(\forall \ldots)(|f(\ldots) - g(\ldots)| \leq c)$$

Likewise, $f(\ldots) \leq g(\ldots) + O(1)$ means:

$$(\exists c)(\forall \ldots)(f(\ldots) \leq g(\ldots) + c)$$

2 Definition of H

First, a little bit of motivation, and some historical remarks (based entirely on Li and Vitányi [10], and the review by Gács [6]). The sequence of twenty bits 11111111111111111111 looks less random than 10010110011001010001. For finite sequences one cannot expect any sharp division between the "random" and "non-random" sequences, but for infinite binary sequences one might hope to define a precise notion of "random".

Probability theory, curiously enough, gets along quite well without a definition of a random sequence. One talks for example of a uniformly distributed random variable between 0 and 1, but what is really needed is a measure (Lebesgue measure) on the interval [0,1]. Kolmogorov in the 1930's showed how classical probability theory could be put on a firm axiomatic basis with this approach [8].

In the 1920's, R. von Mises offered a definition for the notion of a random infinite sequence. His definition did not prove entirely satisfactory, nor did later refinements by A. Wald and A. Church. One might speculate that most mathematicians in the 1940's and '50's regarded this approach as a dead end, especially in light of the success of the measure-theoretic approach to probability theory. The following decade proved otherwise.

In the early 1960's, Solomonoff [13], Kolmogorov [7], and Chaitin [1, 2] all proposed a definition of the "algorithmic information content" or "algorithmic entropy" of a finite bit string: the length of the shortest description of the string under some fixed recursive coding scheme, given say by a Turing machine T. If T is taken to be a universal Turing machine, then the algorithmic information content is invariant (does not depend on the choice of T, up to an additive constant). The term "Kolmogorov complexity", $K(x)$, is often used in the literature for this concept.

Note that this makes precise our vague intuitions about the strings 11111111111111111111 and 10010110011001010001, since we can obviously give a short description of the first string, but essentially have to quote the second. A slightly more subtle example: a sequence of 1000000 1's is less "random", in a sense, than a sequence of 394172 1's. Indeed, to specify a very large number with a very short description, the number has to have some regularity that we can exploit. Li and Vitányi remark that Archimedes' "The Sand-Reckoner" was the first paper to make use of these ideas.

As for infinite random bit strings, Per Martin-Löf [12] finally arrived at a satisfactory definition in 1966. We will look at this in more detail in Section 4.

The analogy with classical entropy played a seminal role in the early history of algorithmic information theory. From Boltzman comes the formula $H = -\log P$; Gibbs refined and extended the theory of entropy. Shannon transplanted these concepts to communication (or coding) theory. To transplant them again to algorithmic information theory, we need a probability distribution on the set of all finite bit strings. The appropriate distribution is known as the Solomonoff-Levin distribution. This distribution does not mesh perfectly with the original definition of algorithmic information $K(x)$. Pursuing this line of thought, Levin [9] was lead to revise the definition of algorithmic information (further results were obtained by Gács [5]). The new definition uses only so-called *self-delimiting* codes, already an important concept in Shannon's theory. Completing the circle of ideas, Levin showed the equivalence of various definitions of "random". Chaitin [3] obtained similar results.

Notation: let

$$\begin{aligned}
\mathbb{N} &= \text{natural numbers} \\
\mathbb{B} &= \text{finite bit strings} \\
&= \{\Lambda, 0, 1, 00, 01, 10, 11, 000, \ldots\} \\
\mathbb{B}^\infty &= \text{infinite bit strings}
\end{aligned}$$

If $s, t \in \mathbb{B}$, let

$$\begin{aligned}
|s| &= \text{the length of } s \\
\mu(s) &= 2^{-|s|} \\
st &= \text{the concatenation of } s \text{ and } t
\end{aligned}$$

Finally, for $n \in \mathbb{N}$, let $\bar{n}$ be a string of n 1's.

Note the canonical one-one correspondences:

$$\begin{array}{lcl}
\mathbb{N} & \leftrightarrow & \mathbb{B} \text{ (lexicographic order)} \\
\mathbb{B}^\infty & \leftrightarrow & [0,1] \text{ up to measure } 0 \\
\mathbb{B} & \leftrightarrow & \text{closed intervals in } \mathbb{B}^\infty \\
 & \leftrightarrow & \text{closed dyadic intervals in } [0,1]
\end{array}$$

We use the correspondence between $\mathbb{N}$ and $\mathbb{B}$ to carry across the usual notions of recursion theory to $\mathbb{B}$. For example, we have partial recursive functions from $\mathbb{B}$ to $\mathbb{B}$, recursively enumerable subsets of $\mathbb{B}$, etc. As for the correspondence between $\mathbb{B}$ and closed intervals in $[0,1]$, note that $\mu(s) =$ measure of the corresponding interval.

Let $\psi : \mathbb{B} \to \mathbb{B}$ be partial recursive. We can regard ψ as an encoding of elements of $\mathbb{B}$, so define

$$H_\psi(x) = \min\{|s| : \psi(s) = x\}$$

Set $H_\psi(x) = \infty$ if there is no s for which $\psi(s) = x$. ($H_\psi(x)$ is not generally computable, but that needn't bother us unless we're intuitionists.) Obviously, H_ψ depends strongly on ψ.

Definition: Let $\{\phi_n\}$ be an effective enumeration of the partial recursive functions from $\mathbb{B}$ to $\mathbb{B}$. We say ψ is *universal* if and only if there is a recursive $\rho : \mathbb{N} \times \mathbb{B} \to \mathbb{B}$ such that

(a) For all $n \in \mathbb{N}$ and $s \in \mathbb{B}$,

$$\psi(\rho(n, s)) \equiv \phi_n(s)$$

(Here, '$\equiv$' means that one side converges if and only if the other side converges, in which case the two sides are equal.)

(b) For each $n \in \mathbb{N}$ there is a constant c_n such that for all s,

$$|\rho(n, s)| \le c_n + |s|$$

Proposition: If ψ is universal, and ϕ is any partial recursive function from $\mathbb{B}$ to $\mathbb{B}$, then

$$H_\psi(x) \leq H_\phi(x) + O(1)$$

(Our $O(1)$ convention means here that the constant implicit in $O(1)$ is independent of x, but depends of course on ψ and ϕ.)

Corollary: If ψ and ψ' are universal, then

$$H_\psi(x) = H_{\psi'}(x) + O(1)$$

Proposition: There exists a universal ψ.

Proof: Let $\rho(n,s) = \bar{n}0s$. We define ψ as follows: ψ strips off the prefix $\bar{n}0$ from its input, sets s equal to what's left, then simulates the computation of $\phi_n(s)$. QED

$H_\psi(x)$ with ψ universal was the original definition of algorithmic entropy. For many applications this is adequate, but to get the cleanest theory, one should make a different choice. This calls for some discussion.

First, note that any partial recursive function ψ from $\mathbb{B}$ to $\mathbb{B}$ can be regarded as a programming language. If $\psi(x) = y$, we think of x as a program that runs for a while and eventually produces y as output. (For now, we consider only programs that take no input.) For familiar programming languages, such as Pascal or Lisp, the source code would have to be expressed as a bit string using a coding scheme like ASCII. It is a common observation that the standard programming languages are "universal", in the sense that any algorithm can in principle be coded in any of them.

Let 'L' stand for the reader's favorite universal programming language. We won't be too specific about L, since no theorems are based on it; we talk about L just to provide motivation[1]. Since L is a "universal language", you might think that the partial recursive function $\mathrm{L} : \mathbb{B} \to \mathbb{B}$ would be

[1]Chaitin's book [4] uses Lisp. My treatment derives from his approach.

universal. It does satisfy condition (a), but not (b). The function $\rho(n, s)$ should be an L program representing the computation $\phi_n(s)$. (Remember, the program operates on empty input.) In general, it will be necessary to represent s inside the program, and since each symbol of L takes 7 bits, we get instead of (b):

(b′) For each $n \in \mathbb{N}$ there is a constant c_n such that for all s,

$$|\rho(n, s)| \leq c_n + 7|s|$$

This implies that for any partial recursive ϕ, $H_{\mathrm{L}}(x) \leq 7H_\phi(x) + O(1)$.

In exchange for the factor of 7, H_{L} has three very nice properties, if L is at all "reasonable" (further discussion below):

1. H_{L} is subadditive, i.e.,

   $$H_{\mathrm{L}}(x, y) \leq H_{\mathrm{L}}(x) + H_{\mathrm{L}}(y) + O(1)$$

   (We'll define $H_\phi(x, y)$ for partial recursive $\phi : \mathbb{B} \to \mathbb{B}$ below.)

2. $P_{\mathrm{L}}(x)$, the probability that a randomly chosen L program outputs x, can be meaningfully defined.

3. $H_{\mathrm{L}}(x)$ satisfies the following *quotation bound*:

   $$H_{\mathrm{L}}(x) \leq 7|x| + O(1)$$

   We'll see reason for the name "quotation bound" below.

Now for the promised definitions:

Definition: Choose some fixed recursive pairing function (i.e., bijection) $\langle\rangle : \mathbb{B} \times \mathbb{B} \to \mathbb{B}$. For any partial recursive ϕ, define

$$H_\phi(x, y) = H_\phi(\langle x, y\rangle) = \min\{|s| : \phi(s) = \langle x, y\rangle\}$$

Definition: For any partial recursive ϕ, define

$$P_\phi(x) = \sum\{\mu(s) : \phi(s) = x\}$$

$$P_\phi(x, y) = P_\phi(\langle x, y\rangle) = \sum\{\mu(s) : \phi(s) = \langle x, y\rangle\}$$

and define

$$\Omega_\phi = \sum\{\mu(s) : \phi(s) \text{ converges}\}$$

Note that the sums may not converge: $\Omega_\phi = \infty$ is possible. Two facts are immediate:

$$P_\phi(x) \geq 2^{-H_\phi(x)}$$

(The convention on when $H_\phi(x) = \infty$ fits in nicely here.) Also,

$$\Omega_\phi = \sum\{P_\phi(x) : x \in \mathbb{B}\}$$

Properties 1 and 2 of H_{L} are important if we want H to mirror the classical H from information theory. Property 1 basically says that L has subroutines in some form. If s is a program for x, and t is a program for y, then one should be able (if L is "reasonable") to create a program of length $\leq |s| + |t| + O(1)$ that computes both x and y, and hence $\langle x, y\rangle$; this is subadditivity.

Property 2 will follow if the set of L programs is prefix-free, i.e., you can't add symbols to the end of a complete program to form a longer program. Prefix-freeness implies that $\Omega_{\mathrm{L}} \leq 1$. Many languages have an explicit end-marker (e.g., Pascal's **end.**, with the period); Lisp achieves prefix-freeness by balancing parentheses. One can always supplement a programming language with an end-of-file symbol, and demand that every program end with it—but also, its first occurrence ends the program. Of course, you have to have an otherwise unused symbol in your alphabet for this to work. In any case, we'll regard prefix-freeness as a "reasonability" requirement on L.

Property 3 falls out of the ability to quote (i.e., represent) a bit string x in a L program of length $7|x| + O(1)$—hence the name "quotation bound".

Again, "reasonable" languages should certainly have a facility for quoting arbitrary bitstrings.

If we replace L with a universal ψ (satisfying condition (b)), then what happens to these three properties?

1. Subadditivity is lost.
2. Convergence of Ω_ψ is lost.
3. The quotation bound holds, without the factor of 7:

$$H_\psi(x) \leq |x| + O(1)$$

The proofs of these facts are instructive.

Proposition: If ψ is universal, then $H_\psi(x) \leq |x| + O(1)$.

Proof: Let $id : \mathbb{B} \to \mathbb{B}$ be the identity function. Then $H_\psi(x) \leq H_{id}(x) + O(1) = |x| + O(1)$. QED

Proposition: If ψ is universal, then $\Omega_\psi = \infty$.

Proof: $\Omega_\psi = \sum\{P_\psi(x) : x \in \mathbb{B}\} \geq \sum\{2^{-H_\psi(x)} : x \in \mathbb{B}\}$, but since $H_\psi(x) \leq |x| + O(1)$, this last sum diverges. QED

Lemma: If $\beta : \mathbb{B} \times \mathbb{B} \to \mathbb{B}$ is injective, then for any $c \in \mathbb{N}$ there exist $x, y \in \mathbb{B}$ such that

$$|\beta(x,y)| > |x| + |y| + c$$

Proof: This is a simple counting argument. Let n be a natural number. The number of ordered pairs $(x, y) \in \mathbb{B} \times \mathbb{B}$ for which $|x| + |y| = n$ is easily seen to be $(n+1)2^n$. The number of bit strings of length $\leq n+c$ is $2^{n+c+1}-1$. For n large enough, the first set has more elements than the second, so a one-one β cannot map the first set into the second. QED

Proposition: If ψ is universal, then for any constant K, there exist $x, y \in \mathbb{B}$ such that

$$H_\psi(x, y) > H_\psi(x) + H_\psi(y) + K$$

In other words, the inequality $H_\psi(x, y) \leq H_\psi(x) + H_\psi(y) + O(1)$ is false.

Proof: Define $\beta(x, y)$ to be the shortest s such that $\psi(s) = \langle x, y\rangle$.[2] Then $|\beta(x, y)| = H_\psi(x, y)$. So for any c, there are x and y such that $H_\psi(x, y) > |x| + |y| + c$. But $H_\psi(x) + H_\psi(y) \leq |x| + |y| + O(1)$ by the quotation bound, so given K we can choose c such that $H_\psi(x) + H_\psi(y) + K \leq |x| + |y| + c$, for all x and y. Hence there are x and y such that $H_\psi(x) + H_\psi(y) + K < H_\psi(x, y)$. QED

Giving up the quotation bound (property 3) in exchange for the properties 1 and 2 turns out to be a good trade. The prefix-free property of L is clearly what is needed to produce an $\Omega \leq 1$; we'll see in a moment that it also is sufficient to give subadditivity.

Definition: A function $\psi : \mathbb{B} \to \mathbb{B}$ is *prefix-free* if and only if the domain of ψ is prefix-free. $\mathcal{P}$ is the class of prefix-free partial recursive functions.

Proposition: If $\psi \in \mathcal{P}$, then $\Omega_\psi \leq 1$.

Definition: A function $\psi \in \mathcal{P}$ is *$\mathcal{P}$-universal* if and only if there is a recursive $\rho : \mathbb{N} \times \mathbb{B} \to \mathbb{B}$ such that

(a) For all $n \in \mathbb{N}$ and $s \in \mathbb{B}$, whenever $\phi_n \in \mathcal{P}$,

$$\psi(\rho(n, s)) \equiv \phi_n(s)$$

(b) For each $n \in \mathbb{N}$ there is a constant c_n such that for all s,

$$|\rho(n, s)| \leq c_n + |s|$$

[2]More precisely, the first in lexicographic order, to avoid ambiguity in case two equally short bit strings produce $\langle x, y\rangle$.

Proposition: If ψ is $\mathcal{P}$-universal, then for any $\phi \in \mathcal{P}$

$$H_\psi(x) \leq H_\phi(x) + O(1)$$

Corollary: If ψ and ψ' are $\mathcal{P}$-universal, then

$$H_\psi(x) = H_{\psi'}(x) + O(1)$$

Corollary: $H_\psi(x)$ is never ∞ for $\mathcal{P}$-universal ψ.

Proof: Let $|x| = n$. Let id_n be the identity function on strings of length n, and undefined on all other strings—so $id_n \in \mathcal{P}$. Then $H_\psi(x) \leq H_{id_n}(x) + c_n = |x| + c_n$, where c_n depends on id_n (and hence on n). QED

We'll eventually show the stronger result: if $|x| = n$, then $H_\psi(x) \leq n + H(\bar{n}) + O(1)$.

Our next order of business is the construction of a $\mathcal{P}$-universal partial recursive function.

Note first that if we have a "program" for a partial recursive ϕ, we can use it to define $\tilde{\phi} \in \mathcal{P}$. Simply dovetail all the computations $\phi(x)$ for all $x \in \mathbb{B}$. From time to time we add a new x to the domain of $\tilde{\phi}$, as follows: if a computation $\phi(x)$ converges, and we don't already have any prefix or extension of x in the domain of $\tilde{\phi}$, then add x to the domain of $\tilde{\phi}$. Of course, we define $\tilde{\phi}(x) = \phi(x)$ whenever $\tilde{\phi}(x)$ is defined. (In particular, if ϕ is already in $\mathcal{P}$, then $\phi = \tilde{\phi}$.) Warning: in general, $\tilde{\phi}$ is not independent of the choice of program for ϕ. However, this won't matter for our application.

Let ψ be the universal function constructed above. Recall that $\psi(\bar{n}0s)$ simulates the computation $\phi_n(s)$, where ϕ_n is a fixed effective enumeration of the partial recursive functions from $\mathbb{B}$ to $\mathbb{B}$. Pick a specific program for ψ, and define $\tilde{\psi}$ as above. Then $\tilde{\psi}$ is $\mathcal{P}$-universal—for if ϕ_n is in $\mathcal{P}$, then it is straightforward to show that for any s, $\tilde{\psi}(\bar{n}0s) \equiv \phi_n(s)$.

We now choose some fixed $\mathcal{P}$-universal function, denote it by Ψ, and let

$$\begin{array}{lcl} H(x) & = & H_\Psi(x) \\ P(x) & = & P_\Psi(x) \\ H(x,y) & = & H_\Psi(x,y) \\ P(x,y) & = & P_\Psi(x,y) \\ \Omega & = & \Omega_\Psi \end{array}$$

We've already observed that $\Omega_\psi \leq 1$ for any $\psi \in \mathcal{P}$, so $\Omega \leq 1$. Indeed, that was the whole motivation for going to prefix-free functions. But we also get subadditivity.

Proposition: $H(x,y) \leq H(x) + H(y) + O(1)$.

Proof: We will construct a $\phi \in \mathcal{P}$ such that whenever $\Psi(s) = x$ and $\Psi(t) = y$, $\phi(st) = \langle x, y \rangle$. The first step for ϕ is to split st into s and t. But since $\Psi \in \mathcal{P}$, s can be characterized as the unique prefix of st for which $\Psi(s)$ converges. So ϕ can find s by dovetailing. Next, ϕ computes $\Psi(s) = x$ and $\Psi(t) = y$. Finally, ϕ constructs $\langle x, y \rangle$. It is easy to check that ϕ is prefix-free. So $H(x,y) \leq H_\phi(x,y) + O(1) = H(x) + H(y) + O(1)$. QED

This proof is a good illustration of the general technique for proving inequalities of the form $H(\ldots) \leq (\ldots) + O(1)$: first construct a $\phi \in \mathcal{P}$ such that $H_\phi(\ldots) = (\ldots)$, then appeal to the basic inequality $H(x) \leq H_\phi(x) + O(1)$. (In fact, it's enough if we can construct a $\phi \in \mathcal{P}$ such that $H_\phi(\ldots) \leq (\ldots) + O(1)$.) The same method easily yields the following results:

Proposition: $H(x) \leq H(x,y) + O(1)$.

Proposition: $H(x, \overline{|x|}) \leq H(x) + O(1)$.

To avoid awkward expressions like $\overline{|x|}$, we adopt the convention that if $n \in \mathbb{N}$ is used in a context where an element of $\mathbb{B}$ is required, then $\bar{n}$ is meant. Thus, combining both of these propositions, $H(x, |x|) = H(x) + O(1)$.

Proposition: If $\eta : \mathbb{B} \to \mathbb{B}$ is partial recursive, then $H(\eta(x)) \leq H(x) + O(1)$.

(We treat the inequality as vacuously true for x for which $\eta(x)$ is undefined.)

In other words, a partial recursive function can add only a bounded amount of information to a bit string. (Note that $x \mapsto \langle x, |x| \rangle$ is partial recursive, so this generalizes the previous proposition.)

3 The Coding Lemma

In this section we will see how to construct a $\psi \in \mathcal{P}$ satisfying a list of requests of the form $H_\psi(x_n) \leq b_n$, assuming certain conditions are met. We call this the Coding Lemma, since in essence we are trying to find short codes for the list of elements x_n. Kraft in 1949 proved a version of this lemma for a finite list of requests.

If we can construct such a ψ, then by the remarks concluding the previous section, we have $H(x_n) \leq b_n + O(1)$, where the constant implied by $O(1)$ is independent of n.

If $H_\psi(x_n) \leq b_n$, then

$$2^{-b_n} \leq 2^{-H_\psi(x_n)} \leq P_\psi(x_n)$$

and so

$$\sum 2^{-b_n} \leq \Omega_\psi \leq 1$$

a necessary condition known as the *Kraft inequality.* The coding lemma basically says the Kraft inequality is also sufficient. We have to be a little careful, because we want ψ to be partial recursive.

Note that if η is a partial recursive function from $\mathbb{N}$ to some set S, then there is a partial recursive function ζ with the same range but which is defined on either all of $\mathbb{N}$, or else an initial segment of $\mathbb{N}$. Just dovetail all the computations $\{\eta(i) : i \in \mathbb{N}\}$, and let $\zeta(n)$ be the result of the n-th computation to converge. By an *r.e. sequence* in S, we mean the range of a partial recursive function from $\mathbb{N}$ to S. When we write $\{s_n\}$ for such a sequence, we assume implicitly that S is the range of ζ as just described, and $s_n = \zeta(n)$. One should imagine a machine spitting out the elements $s_0, s_1, \ldots$, where one is never sure that any more s_n will appear in the machine's output.

Coding Lemma: Let $\{(x_n, b_n)\}$ be an r.e. sequence in $\mathbb{B} \times \mathbb{N}$. Suppose

$$\sum 2^{-b_n} \leq 1$$

Then there is a $\psi \in \mathcal{P}$ such that:

(a) For all n,

$$H_\psi(x_n) \leq b_n$$

(b) For all $x \in \mathbb{B}$,

$$P_\psi(x) = \sum \{2^{-b_n} : x_n = x\}$$

(c) For all n there exists m such that

$$x_n = x_m \text{ and } H_\psi(x_n) = b_m$$

That is, for all n,

$$H_\psi(x_n) = \min\{b_m : x_m = x_n\}$$

(All we'll use is part (a).)

Proof: (Sketch). Strategy: define s_n inductively such that $|s_n| = b_n$ and $\{s_0, \ldots, s_n\}$ is prefix-free, and set $\psi(s_n) = x_n$. We chose s_n to be the lexicographically first element of the set of *available* bit strings of the right length, where initially all bit strings are available, but after $\{s_0, \ldots, s_{n-1}\}$ have been chosen, a bit string s is available if and only if $\{s_0, \ldots, s_{n-1}, s\}$

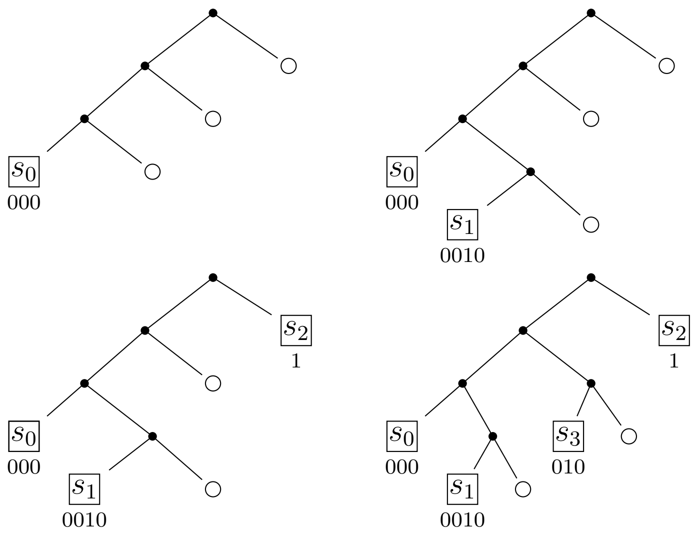

Figure 1: The Coding Lemma
This diagram shows successive stages for the sequence of requests $(b_0, b_1, b_2, b_3) = (3, 4, 1, 3)$. The open circles are the roots r_i and the boxes are the allocated strings s_j. The initial request for a string of length 3 is met with $s_0 = 000$, the first available string. The roots are then 001, 01, and 1. The request $b_1 = 4$ results in $s_1 = 0010$, with roots now 0011, 01, and 1. Etc.

is prefix-free, i.e., s is not a prefix or extension of any s_i, $i = 0, \ldots, n-1$. (See fig.1.)

Picturing $\mathbb{B}$ as the nodes in an infinite binary tree, the set of available bit strings at any stage consists of a disjoint union of a finite number of subtrees—this is easy to prove by induction. Let $\{r_1, \ldots, r_k\}$ be the roots of these subtrees at stage n, ordered left to right (i.e., not lexicographically, but in the obvious ordering in which, for example, 000 is to the left of 10. This is the same as lexicographic order *after* we pad out all the r_i with 0's on the right to make them the same length.)

By induction one can prove that the following two facts hold at every stage n:

$$|r_1| > \cdots > |r_k|$$

i.e.,

$$\mu(r_1) < \cdots < \mu(r_k)$$

and

$$\sum_{i=0}^{n} 2^{-b_i} + \sum_{i=1}^{k} \mu(r_i) = 1$$

By the Kraft inequality, $2^{-b_{n+1}} \leq 1 - \sum_{i=0}^{n} 2^{-b_i} = \sum_{i=1}^{k} \mu(r_i)$. But since $\mu(r_i)$ is $2^{-|r_i|}$ and $\mu(r_1) < \ldots < \mu(r_k)$, it follows that $2^{-b_{n+1}} \leq \mu(r_k)$. (Think of the binary expansion of $\sum_{i=1}^{k} \mu(r_i)$.) Thus $b_{n+1} \geq |r_k|$ and there will always be an available bit string of the desired length. QED

We can regard this as the solution to a storage allocation problem. One might say that our allocation policy does not fragment storage: if a request can't be satisfied, it's because there isn't enough total storage left.

4 Randomness

The goal of this section is to define the notion of a random infinite bit string, and to show that Ω is random.

We will consider four definitions of "random"[3]. Two are probabilistic, two information theoretic (i.e., depend on the function H). The equivalence of the two kinds of definitions seems "deep", and is another goal of this

[3]I am not exactly sure about all the twists and turns in the histories of these definitions. I have mentioned Martin-Löf's name in the notes, since all my (secondary) sources agree that he first introduced the "constructive measure" definition of random. The story for the other definitions is apparently more complicated.

section. We will use the term "random" for the probabilistic definitions, and "patternless" for the information theoretic definitions.

The probabilistic definitions start from a simple idea. Suppose $\{A_n\}$ is a sequence of subsets of $\mathbb{B}^\infty$ such that $\lim \mu(A_n) = 0$. Then the probability that a given $r \in \mathbb{B}^\infty$ belongs to all A_n is 0, and so a test for randomness is that r fails to belong to at least one of the A_n. No infinite bit string can pass all such tests, but if we allow only "constructively given" sequences $\{A_n\}$, then it turns out that almost all elements of $\mathbb{B}^\infty$ are random.

The information theoretic definitions start from an equally simple idea. If r is random, then initial segments of r should be "patternless", i.e., should not have short descriptions. In other words, if r_n is the first n bits of r, then $H(r_n)$ should not be significantly less than n for large n.

We need some preliminaries on $\mathbb{B}^\infty$ to make these ideas precise. Note that elements of $\mathbb{B}$ can be identified with subsets of $\mathbb{B}^\infty$ in the obvious way; also, if we picture $\mathbb{B}^\infty$ as an infinite binary tree, then elements of $\mathbb{B}$ correspond to subtrees.

There is an obvious measure μ on $\mathbb{B}^\infty$ that makes $\mathbb{B}^\infty$ into a measure space, essentially the same at the unit interval $[0,1]$. This μ is an extension of the μ already defined on $\mathbb{B}$: if x is in $\mathbb{B}$, and X is the corresponding subset of $\mathbb{B}^\infty$, then $\mu(x) = \mu(X)$. (Generally, we won't even make any distinction between x and X, writing x (or X) for both concepts.)

The extension property just mentioned is important for the theorems of this section. Recall that we restricted our attention to $\mathcal{P}$ at least partly because $\sum\{\mu(s) : s \in \mathbb{B}\}$ does not converge. Let's define

$$\mu'(s) = 3^{-|s|-1}$$

then $\sum\{\mu'(s) : s \in \mathbb{B}\} = 1$. However, μ on $\mathbb{B}^\infty$ is not an extension of μ' on $\mathbb{B}$, and so working with μ' instead of μ does not lead to nice theorems.

Notation: r will denote an infinite bit string throughout, and r_n denotes the first n bits of r.

Definition: A *test* is a sequence $\{A_n\}$ of subsets of $\mathbb{B}^\infty$ such that $\lim \mu(A_n) = 0$. An infinite bit string r *passes the test* $\{A_n\}$ if and only if for some n, $r \notin A_n$. Also, r *passes the test infinitely often* if and only if $r \notin A_n$ for infinitely many n, and *passes the test almost always* if $r \notin A_n$ for almost all n.

Proposition: For any $r \in \mathbb{B}^\infty$ there is a test that r does not pass.

So we need to define a subclass of the class of all tests if we are to have any r which pass all of them. Since $\mu(\bigcap A_n) = 0$ for any test $\{A_n\}$, if the subclass is countable, then almost all $r \in \mathbb{B}^\infty$ will pass all tests in the subclass. The class we'll finally adopt is the class of "constructively given" tests.

Let $f : \mathbb{N} \to \mathbb{B}$ be recursive. Let $A_n = f(n)$. We certainly want such a sequence $\{A_n\}$ to count as "constructively given". However, the class of tests of the form $\{f(n)\}$, $f : \mathbb{N} \to \mathbb{B}$ recursive, is too narrow to lead to a satisfactory definition of "random". Say $(b_0 b_1 \ldots)$ passes all such tests. It follows that $(b_0 b_0 b_1 b_1 \ldots)$ also passes this class of tests—but we would never consider such a "doubled" bit string to be random.

Definition: Let $f : \mathbb{N} \times \mathbb{N} \to \mathbb{B}$ be partial recursive. Let

$$A_n = \bigcup_k f(n, k)$$

(If $f(n, k)$ does not converge for some n and k, then we treat that term as representing the empty set.) We say that the sequence $\{A_n\}$ is *constructively given*.

Definition: A *constructive test* is a constructively given sequence $\{A_n\}$ such that, for all n, $\mu(A_n) \leq 2^{-n}$. We say r is *random* if it passes all constructive tests.

The above definition of random is due to Martin-Löf.

Proposition: Almost all $r \in \mathbb{B}^\infty$ are random.

By looking at subsequences of a constructive test, we can change the rate of convergence of $\lim \mu(A_n)$. The notion of a regulator of convergence makes this precise:

Definition: A *regulator* is a monotonically decreasing recursive function $g : \mathbb{N} \to \mathbb{Q}^+$ (positive rationals) such that $\lim g(n) = 0$. A *constructive test with regulator* g is a constructively given sequence $\{A_n\}$ such that for all n, $\mu(A_n) \leq g(n)$.

Proposition: If r is random, then for any regulator g and any constructive test with regulator g, r passes the test infinitely often.

Proposition: If r is not random, then for any regulator g there exists a constructive test with regulator g such that r fails the test (i.e., r belongs to all the sets in the sequence.)

Both propositions are proved by looking at subsequences of a test. Notice the asymmetry: the second proposition is not the contrapositive of the first.

Definition: A *finite-measure test* is a constructively given sequence $\{A_n\}$ such that $\sum \mu(A_n) < \infty$. We say r is *strongly random* if it passes all finite-measure tests almost always.

Strong randomness implies randomness because the strong definition subjects the candidate r to a wider class of tests, and r has to pass each test in a more stringent sense.

The motivation for the strong definition comes from the first Borel-Cantelli lemma: if $\sum \mu(A_n) < \infty$, then the probability that r belongs to infinitely many of the A_n is 0. Thus a random r should pass a finite-measure test almost always. An instructive "counterexample": let $\{A_n\}$ be the elements of $\mathbb{B}$ in lexicographic order. Then $\lim \mu(A_n) = 0$, clearly, but every $r \in \mathbb{B}^\infty$ belongs to infinitely many of the A_n.

Theorem: Randomness implies strong randomness.

Proof: Suppose r is not strongly random, so for some finite-measure test $\{A_n\}$, r belongs to infinitely many of the A_n. Let B_k be the set of all r that belong to at least k of the A_n. It is straightforward to show that $\{B_k\}$ is constructively given. Each element of B_k contributes at least k times to the sum $\sum \mu(A_n)$, so $\mu(B_k) \leq (\sum \mu(A_n))/k$. Thus if we choose a regulator g such that $(\sum \mu(A_n))/k \leq g(k)$, then $\{B_k\}$ is a constructive test with a regulator g. But r belongs to all the B_k, so r fails the test $\{B_k\}$. QED.

If you use some recursive betting scheme to bet on successive bits of a random r, you'd expect to win about half the time. This result holds; see for example Theorem R7 in Chaitin [4] for a precise formulation and proof.

Next we turn to the information theoretic definitions of random. Recall r_n denotes the first n bits of r, and r is an infinite bit string.

Definition: We say r is *weakly patternless* if and only if $\liminf H(r_n) - n > -\infty$, i.e.,

$$(\exists k)(\forall n) \quad H(r_n) > n - k$$

We say r is *patternless* if and only if $\lim H(r_n) - n = \infty$, i.e.,

$$(\forall k)(\exists N_k)(\forall n > N_k) \quad H(r_n) > n + k$$

Obviously patternless implies weakly patternless. We complete the cycle of implications by showing that strongly random implies patternless, and weakly patternless implies random.

The intuition behind the first of these implications is pretty simple: for any $x \in \mathbb{B}$, $\mu(x) = 2^{-|x|}$ and $P(x) \geq 2^{-H(x)}$. We know that $\sum P(x)$ converges, so if we look just at x's for which $H(x) \approx |x|$, then $\mu(x)$ will converge too. This is enough to apply the finite-measure criterion. (Note that this argument does not work if we replace μ with μ'.)

Theorem: Strongly random implies patternless.

Proof: Pick an arbitrary k. Let

$$A = \{x \in \mathbb{B} : H(x) \leq |x| + k\}$$

For any $x \in A$,

$$P(x) \geq 2^{-H(x)} \geq 2^{-|x|-k} = 2^{-k}\mu(x)$$

so (remembering that $x \in \mathbb{B}$ is identified with $x \subseteq \mathbb{B}^\infty$)

$$\sum\{\mu(x) : x \in A\} \leq 2^k \sum\{P(x) : x \in A\} \leq 2^k$$

Now let $A_n = \{x \in A : |x| = n\}$. It's straightforward to show that $\{A_n\}$ is constructively given, and clearly $\sum \mu(A_n) = \sum\{\mu(x) : x \in A\}$. So $\{A_n\}$ is a finite-measure test. Therefore if r is strongly random, $r \notin A_n$ for almost n, i.e., $H(r_n) > n + k$ for almost n. Since k was arbitrary, we are done. QED

Our next goal is to prove weakly patternless implies random. We will need a lemma for this.

Lemma: If $\{A_n\}$ is a constructively given sequence, then each A_n is the disjoint union of elements of $\mathbb{B}$:

$$A_n = \bigsqcup_i A_{n,i}, \quad A_{n,i} \in \mathbb{B}$$

Moreover, the map $(n, i) \mapsto A_{n,i}$ can be chosen partial recursive.

Proof: The $f(n, i)$ in the definition of "constructively given" has the required properties except for disjointness. The "$\eta - \zeta$" dovetailing argument given just before the proof of the Coding Lemma allows us to assume that for any n, the domain of $f(n, i)$ is either $\mathbb{N}$ or else an initial segment of $\mathbb{N}$. So define

$$B_{n,0} = f(n, 0)$$

$$B_{n,i} = f(n, i) - \bigcup_{j=0}^{i-1} f(n, j) \quad (i > 0)$$

$A_n = \bigsqcup_i B_{n,i}$, clearly, and each $B_{n,i}$ is the disjoint union of a finite number of elements of $\mathbb{B}$, so QED.

Theorem: Weakly patternless implies random.

Proof: This is an application of the Coding Lemma: if r is not random, then we construct a code which prevents r from being weakly patternless.

Suppose r is not random. So r belongs to A_n for all n, where $\{A_n\}$ is a constructive test regulated by g, g being a regulator we can choose at will. Each A_n is a disjoint union of elements of $\mathbb{B}$—say $A_n = \bigsqcup_i A_{n,i}$. The condition $r \in A_n$ then says that for each n, one of the $A_{n,i}$ is an initial segment of r.

We want to give "short codes" to the $A_{n,i}$. Specifically, we will construct a $\psi \in \mathcal{P}$ such that $H_\psi(A_{n,i}) \leq |A_{n,i}| - n$, and hence $H(A_{n,i}) \leq |A_{n,i}| - n + c$ for some constant c. It is then clear that $\liminf H(r_m) - m = -\infty$, i.e.,

$$(\forall k)(\exists m) \quad H(r_m) \leq m - k$$

The Coding Lemma says we can construct such a ψ provided

$$\sum_{n,i} 2^{-|A_{n,i}|+n} = \sum_n \left(2^n \sum_i \mu(A_{n,i}) \right) \leq 1$$

Because A_n is the disjoint union of the $A_{n,i}$, this last sum is

$$\sum_n 2^n \mu(A_n) \leq \sum_n 2^n g(n)$$

Choosing $g(n) = 1/2^{n^2+1}$ completes the proof. QED

Hence all four notions of "random" are equivalent.

We conclude this section with the proof that Ω is random. There are two ingredients to this. First, Ω can be computed as a limit from below, just by dovetailing all the $\Psi(s)$ computations and seeing which converge. Second, as noted at the end of Section 2, a partial recursive function can add at most a bounded amount of information to a bit string:

Proposition: If $\eta : \mathbb{B} \to \mathbb{B}$ is partial recursive, then $H(\eta(x)) \leq H(x) + O(1)$.

Theorem: Ω is random.

Proof: The set of $\{s \in \mathbb{B} : \Psi(s) \text{ converges}\}$ is recursively enumerable; say $\{s_k\}$ is an enumeration of it. Let $\omega_m = \sum_{k=0}^{m} \mu(s_k)$. Then $\lim \omega_m = \Omega$.

Let Ω_n denote the first n bits of Ω. We will define a recursive function η such that $H(\eta(\Omega_n)) > n$. Since $H(\eta(\Omega_n)) \leq H(\Omega_n) + O(1)$, it will then follow that $H(\Omega_n) > n - O(1)$, i.e., Ω is weakly patternless.

Given Ω_n, the computation of $\eta(\Omega_n)$ proceeds as follows: compute ω_m for larger and larger m until we find an ω_m whose first n bits are the same as Ω_n.[4] This means that all the subsequent estimates $\omega_{m+1}, \omega_{m+2}, \ldots$, have the same first n bits as ω_m. So $s_{m+1}, s_{m+2}, \ldots$, all have length greater than n. In other words, if $\Psi(s)$ converges and $|s| \leq n$, then s belongs to the set $\{s_0, \ldots, s_m\}$. Pick an x not in the set $\{\Psi(s_0), \ldots, \Psi(s_m)\}$, so $H(x) > n$. Set $\eta(\Omega_n) = x$. QED

It can be shown that the binary expansion of Ω is not the characteristic function of a recursively enumerable set. In other words, there is no Turing machine which from time to time spits out statements of the form "bit n of Ω is a 1", spitting out only true statements and eventually listing all 1 bits.

[4]To be sure this is possible, we must represent Ω in non-terminating form. That is, we choose the 0111... form over the 1000... form. In fact, since Ω is random, this choice doesn't arise, but we don't know that yet. We represent the ω_m in terminating form, however.

5 *H* Revisited

The purpose of this section is to prove three facts about H:

$$H(x) = -\log P(x) + O(1)$$

(all logs are to base 2)

$$H(x, y) = H(x) + H(y/x) + O(1)$$

$$H(x) \leq |x| + H(|x|) + O(1)$$

We've already observed the inequality $P(x) \geq 2^{-H(x)}$; taking logs, this becomes $H(x) \geq -\log P(x)$. The reverse inequality (apart from an additive constant) shows that it is not possible for $P(x)$ to be large (i.e., close to 1) by having a great many long codes for x. In other words, $\sum\{2^{-|s|} : \Psi(s) = x\}$ has the same order of magnitude as its largest term, $2^{-H(x)}$.

If we could construct a $\psi \in \mathcal{P}$ such that $H_\psi(x) \leq -\log P(x)$, then by the usual argument the desired inequality for H would follow. In fact, we can construct something just as good: a $\psi \in \mathcal{P}$ for which $H_\psi(x) \leq -\log P(x) + 2$.

The proof is an application of the Coding Lemma. By dovetailing, we can effectively compute $P(x)$ as a limit from below (analogous to what we did for Ω at the end of the last section), and hence we can effectively compute $-\log P(x)$ as a limit from above. This enables us to construct an r.e. sequence of requests $H_\psi(x_m) \leq b_m$ which includes every $x \in \mathbb{B}$, and which forces the desired inequality for $H_\psi(x)$.

Theorem: $H(x) = -\log P(x) + O(1)$

Proof: Let $\{s_i\}$ be an effective enumeration of all s for which $\Psi(s)$ converges. As we compute $\Psi(s_0), \Psi(s_1), \ldots$, we maintain, for a growing list of x's, a table of lower bounds for $P(x)$ and upper bounds for $-\log P(x)$.

More specifically, the entry for a given x contains a positive lower bound $L(x) \le P(x)$ and a sequence of upper bounds $a_1 > \ldots > a_k \ge -\log P(x)$. The a_j's are all natural numbers. There is an entry for every x for which we know $P(x)$ is positive, i.e., for which we've found an s such that $\Psi(s) = x$. The following facts are maintained as we construct the table:

$$\begin{array}{l} 0 < L(x) \le P(x) \\ a_1 > \ldots > a_k \\ 2^{-a_k} \le L(x) < 2^{-(a_k-1)} \\ a_k \ge -\log L(x) > a_k - 1 \end{array}$$

Of course, the last line of inequalities is just a restatement of the previous line. Here is how the table is constructed. Initially it is empty. When we compute $\Psi(s_i) = x$, if there is already an entry for x, then we update $L(x)$ by adding $\mu(s_i)$ to it. If necessary, we append a new element to the list of a_j's to maintain the inequality $a_k \ge -\log L(x) > a_k - 1$, but if $\lceil -\log L(x) \rceil$ remains the same, then we don't. If there is no entry for x, then we make one with $L(x) = \mu(s_i)$ and $a_1 = \lceil -\log L(x) \rceil$.

Suppose we dovetailed all the a_j sequences for the different x's into a single sequence of requests $\{H_\psi(x_m) \le b_m\}$, would the Kraft inequality be satisfied?[5] Not quite: we have, for any x,

$$\sum_{j=1}^{k} 2^{-a_i} \le 2^{-a_k}(1 + \frac{1}{2} + \frac{1}{4} + \ldots) \le 2P(x)$$

so we'd have $\sum 2^{-b_m} \le 2\Omega \le 2$. We compensate by entering requests of the form $H_\psi(x) \le a_j + 1$ (i.e., the sequence $\{b_m\}$ is obtained by dovetailing the sequences $\{a_1 + 1, \ldots, a_k + 1\}$.)

We now have $H_\psi(x) \le a_k + 1$. What does this tell us about $-\log P(x)$? Well, observe that while the $L(x)$ may keep growing indefinitely, tending to

[5]The requests $\{H_\psi(x) \le a_j\}$ appear on this dovetailed list in the order the a's are added to the table.

$P(x)$ in the limit, the sequence $a_1 > \ldots > a_k$ must stabilize. If a_k is the last element ever added to the sequence for x, then $-\log P(x) > a_k - 1$, i.e., $H_\psi(x) \leq a_k + 1 < -\log P(x) + 2$. QED

As a corollary, we can give an upper bound for the number of codes for x of length up to $H(x) + k$, for any k.

Our next topic is $H(x/y)$, or relative information. Intuitively, $H(x/y)$ should be the information needed to obtain x given y, so one is inclined to define $H_\psi(x/y) = \min\{|s| : \psi^y(s) = x\}$, where $\psi^y(s)$ means y is available to ψ as an "oracle". More precisely, we want to consider partial recursive functions $\psi : \mathbb{B} \times \mathbb{B} \to \mathbb{B}$. We'll look for s such that $\psi(s, y) = x$.

For more motivation, let's look at the programming language L again. In Section 1, we considered only L programs without input. Suppose a L program s takes input y and produces output x: $\mathrm{L}(s, y) = x$. We imagine L is endowed with suitable functions for manipulating bit strings. We allow arbitrary bit strings as input, but the program is an ASCII encoding of a syntactically valid sequence of symbols of L.

If $\mathrm{L}(s, y) = x$, it's natural to try to obtain a L program that produces x without any input. This is easy: just quote y inside the program. However, quoting y involves encoding it into ASCII, so the length of the resulting program will be about $|s| + 7|y|$.

In classical recursion theory, $\phi_{f(y)}(s)$ and $\phi(s, y)$ are practically the same thing—a recursive function of one variable whose index depends recursively on a parameter is practically the same as a recursive function of two variables. Because of the factor of 7, the two concepts are significantly different in our context.

We will accordingly be interested in partial recursive functions $\psi : \mathbb{B} \times \mathbb{B} \to \mathbb{B}$, where we want prefix-freeness on the first argument but impose no restrictions on the second argument.

Definition: A function $\psi : \mathbb{B} \times \mathbb{B} \to \mathbb{B}$ belongs to $\mathcal{P}$ if and only if it is partial recursive, and for each $y \in \mathbb{B}$, the function

$$s \mapsto \psi(s, y)$$

is prefix-free.

Notation: $\psi(\cdot, y)$ denotes the function $s \mapsto \psi(s, y)$.

Definition: Suppose $\{\phi_n\}$ is an effective enumeration of the partial recursive functions from $\mathbb{B} \times \mathbb{B}$ to $\mathbb{B}$. A function $\psi : \mathbb{B} \times \mathbb{B} \to \mathbb{B}$ is $\mathcal{P}$-*universal* if and only if ψ is in $\mathcal{P}$, and if there is a recursive function $\rho : \mathbb{N} \times \mathbb{B} \to \mathbb{B}$ such that

(a) For all $n \in \mathbb{N}$ and $y \in \mathbb{B}$, if $\phi_n(\cdot, y) \in \mathcal{P}$ (i.e., one-variable $\mathcal{P}$), then for all $s \in \mathbb{B}$,
$$\psi(\rho(n, s), y) \equiv \phi_n(s, y)$$

(b) For each $n \in \mathbb{N}$ there is a constant c_n such that for all s,
$$|\rho(n, s)| \leq c_n + |s|$$

The definition of $\mathcal{P}$-universal is perhaps a little stronger than expected: instead of requiring $\psi(\rho(n, s), y) \equiv \phi_n(s, y)$ only when $\phi_n \in \mathcal{P}$, we require it for all values of y such that $\phi_n(\cdot, y) \in \mathcal{P}$. (By definition, ϕ_n is in $\mathcal{P}$ if and only if $\phi_n(\cdot, y)$ is in $\mathcal{P}$ for all y.) However, the construction of a $\mathcal{P}$-universal function of one variable parametrizes without trouble to yield a $\mathcal{P}$-universal function of two variables.

Notation: We will use Ψ to denote a fixed $\mathcal{P}$-universal function of two variables, as well as of one variable. Without loss of generality, we may assume
$$\Psi(s) \equiv \Psi(s, \Lambda)$$
i.e., the one-variable Ψ is just the two-variable Ψ with empty input.

Definition: For any $x, y \in \mathbb{B}$, $\phi : \mathbb{B} \times \mathbb{B} \to \mathbb{B}$ partial recursive,

$$H_\phi(x?y) = \min\{|s| : \phi(s, y) = x\}$$

Definition: For any $x \in \mathbb{B}$,

$$x^* = (\text{the first } s \text{ such that } \Psi(s) = x)$$

Thus $H(x) = |x^*|$. We let

$$H_\phi(x/y) = H_\phi(x?y^*)$$

As usual,

$$H(x?y) = H_\Psi(x?y)$$

$$H(x/y) = H_\Psi(x/y)$$

$H(x?y)$ seems the more natural definition, and so we work with it to see how far we can get.

Proposition: For any $\phi \in \mathcal{P}$,

$$H(x?y) \leq H_\phi(x?y) + O(1)$$

$$H(x/y) \leq H_\phi(x/y) + O(1)$$

The next proposition is intuitively satisfying, though we won't need it for anything.

Proposition: $H(x/y) \leq H(x?y) + O(1)$, and $H(x?y) \leq H(x) + O(1)$.

The important thing about relative information is the additivity formula. One half of this is easy.

Proposition: $H(x, y) \leq H(x?y) + H(y) + O(1)$.

Proof: This is a straightforward modification of the argument in Section 2 that proved that $H(x, y) \leq H(x) + H(y)$.

Proposition: $H(x, y) \leq H(x/y) + H(y) + O(1)$.

Proof: Ditto.

These propositions do not seem "deep" (they don't rely on the Coding Lemma, for example) but they quickly yield a form of "quotation bound" for H. Recall from the end of Section 2 that $H(x) \leq H(x, |x|) + O(1)$. (In fact, $H(x) = H(x, |x|) + O(1)$, but we won't need that.)

Proposition: $H(x) \leq |x| + H(|x|) + O(1)$.

Proof: $H(x) \leq H(x, |x|) + O(1) \leq H(x?|x|) + H(|x|) + O(1)$. The "$id_n$" argument from Section 2 shows that $H(x?|x|) \leq |x| + O(1)$. QED

We can sharpen this. Let Υ be a fixed universal function (not $\mathcal{P}$-universal, but universal) from $\mathbb{B}$ to $\mathbb{B}$. Let $J(x) = \min\{|s| : \Upsilon(s) = x\}$. We observed in Section 2 that $J(x) \leq |x| + O(1)$, since we don't have to worry about prefix-freeness. It is also easy to see that for any $n \in \mathbb{N}$, $J(n) \leq \log n + O(1)$.

Proposition: $H(x) \leq J(x) + H(J(x)) + O(1)$.

Proof: Exercise for the reader.

Now we can iterate: $H(x) \leq J(x) + H(J(x)) + O(1) \leq J(x) + J(J(x)) + H(J(J(x))) + O(1) \leq \ldots$. Since $J(n) \leq \log n + O(1)$, it does not take long for this series to terminate.

It is also interesting to imagine explicit prefix-free schemes for "quoting" a bit string. That is, we want a total recursive function $\eta : \mathbb{B} \to \mathbb{B}$ (the quoting function) and a partial function $\zeta \in \mathcal{P}$ (the unquoting function) such that $\zeta(\eta(x)) = x$ for all x. We then have $H(x) \leq |\eta(x)| + O(1)$ for all x. Choosing $\eta(x) = n0x$, where $n = |x|$, gives $H(x) \leq |x| + |x| + O(1) = 2|x| + O(1)$. Recall that n is n 1's in a row. Choosing $\eta(x) = m0n'x$, where

n' is $|x|$ written in binary, and $m = |n'|$, gives $H(x) \leq |x| + 2\log|x| + O(1)$. Repeating this idea gives $H(x) \leq |x| + \log|x| + 2\log\log|x| + O(1)$. The general pattern is clear in all these inequalities: to describe x, we will usually have to describe the length of x as well as the actual bits in x.

Our last major theorem is the additivity of information: $H(x, y) = H(x/y) + H(y) + O(1)$. The proof uses the Coding Lemma. We need, in fact, a parametrized version of the Coding Lemma.

Extended Coding Lemma: Let $\{(x_m, b_m, t_m)\}$ be an r.e. sequence in $\mathbb{B} \times \mathbb{N} \times \mathbb{B}$. For any t, let $\{(x_n(t), b_n(t), t)\}$ be the subsequence of all (x_m, b_m, t_m) such that $t_m = t$.

There exists a $\psi \in \mathcal{P}$ such that for each t, if

$$\sum 2^{-b_n(t)} \leq 1$$

then the following all hold:

(a) For all n,

$$H_\psi(x_n(t)?t) \leq b_n(t)$$

(b) For all $x \in \mathbb{B}$,

$$P_\psi(x?t) = \sum\{2^{-b_n(t)} : x_n(t) = x\}$$

(c) For all n there exists m such that

$$x_n(t) = x_m(t) \text{ and } H_\psi(x_n(t)?t) = b_m(t)$$

That is, for all n,

$$H_\psi(x_n(t)?t) = \min\{b_m(t) : x_m(t) = x_n(t)\}$$

As before, all we'll use is part (a).

Proof: Simply carry out the construction of the Coding Lemma simultaneously for each t. QED

Let us indicate how we are going to apply the Extended Coding Lemma. The combined sequence $\{(x_m, b_m, t_m)\}$ must be r.e. The Kraft inequalities are applied to subsequences obtained by picking a particular value of t and looking only at (x_m, b_m, t_m) with $t_m = t$. Suppose the Kraft inequality holds for a particular t, and we pick a particular x and look at the subsequence (x_m, b_m, t_m) for which $x_m = x$ and $t_m = t$. Denote this subsequence by $\{(x, b_n(x,t), t)\}$. If $b_n(x,t)$ converges down to some function $f(x,t)$ of x and t, then we will have constructed a ψ for which $H_\psi(x?t) \le f(x,t)$.

We already have $H(x,y) \le H(x?y) + H(y) + O(1)$. Let us try to prove the reverse inequality; this will give insight into the rationale for the definition of $H(x/y)$. $H(x,y) \ge H(x?y) + H(y) + O(1)$ can be rewritten $H(x?y) \le H(x,y) - H(y) + O(1)$. It is natural to try to enforce this inequality with the Extended Coding Lemma, which means that we need an r.e. sequence of bounds converging down to $H(x,y) - H(y)$. Unfortunately, $H(y)$ in general cannot be computed as a limit from below, so this approach runs into a blind alley. In fact, the inequality $H(x,y) \ge H(x?y) + H(y) + O(1)$ is false; we will show this after we prove the inequality for $H(x/y)$.

Consider the analogous inequality for $H(x/y)$:

$$H(x/y) = H(x?y^*) \le H(x,y) - H(y) + O(1)$$

or substituting t for y^*, so $\Psi(t) = y$ and $H(y) = |t|$:

$$H(x?t) \le H(x, \Psi(t)) - |t| + O(1)$$

We can compute the right hand side of this inequality as an r.e. limit from above, so we can apply the Extended Coding Lemma.

We will need one last lemma. Notation: $f(x) \simeq g(x)$ means the ratio $f(x)/g(x)$ is bounded away from both 0 and ∞.

Lemma: $P(y) \simeq \sum_x P(x,y)$.

Proof: The trick is to look at the compositions

$$s \overset{\Psi}{\mapsto} \langle x, y\rangle \mapsto y$$

and

$$s \overset{\Psi}{\mapsto} y \mapsto \langle y, y\rangle$$

We leave details to the reader. QED

Theorem: $H(x,y) = H(x/y) + H(y) + O(1)$.

Proof: The inequality we are trying to enforce is

$$H_\psi(x?t) \leq H(x, \Psi(t)) - |t| + C$$

for some constant C independent of x and t. We want this to hold for all *minimal* t, that is, for all t such that $\Psi(t)$ converges, and $H(\Psi(t)) = |t|$ (so there is no shorter program for computing $\Psi(t)$). If we have an s for which $\Psi(s) = \langle x, \Psi(t)\rangle$, then $|s|$ is an upper bound for $H(x, \Psi(t))$. So we dovetail all computations $\Psi(\cdot)$, and whenever we find

$$\Psi(s) = \langle x, \Psi(t)\rangle$$

we add

$$(x, |s| - |t| + C, t)$$

to the list $\{(x_m, b_m, t_m)\}$. C here is a constant we will pick so as to make the Kraft inequalities hold.

Fix a minimal t. We wish to bound

$$\sum\{2^{-|s|+|t|-C} : (\exists x)\Psi(s) = \langle x, \Psi(t)\rangle\}$$

This equals

$$2^{|t|-C}\sum\{\mu(s) : (\exists x)\Psi(s) = \langle x, \Psi(t)\rangle\} = 2^{|t|-C}\sum_x P(x, \Psi(t))$$

By the lemma above,

$$\sum_x P(x, \Psi(t)) \leq (\text{constant}) P(\Psi(t)) \leq (\text{constant}) 2^{-H(\Psi(t))} = (\text{constant}) 2^{-|t|}$$

since t is minimal. The constants in the above inequalities are not the same, but they are all independent of t. So the sum we wish to bound is less than or equal to some constant times 2^{-C}, and we can choose C to make the Kraft inequalities all hold.

Thus for any minimal t, $H_\psi(x?t) \leq H(x, \Psi(t)) - |t| + C$, and so setting $y = \Psi(t)$, we have $H_\psi(x/y) \leq H(x, y) - H(y) + C$. It follows in the usual fashion that $H(x/y) \leq H(x, y) - H(y) + O(1)$, which combined with previous results proves the theorem. QED

We will show finally that $H(x, y) \neq H(x?y) + H(y) + O(1)$, i.e., $H(x?y) + H(y) - H(x, y)$ is unbounded. This result is due to Gács, who in fact proved something stronger (but we only give the weaker result).

The proof consists of a series of lemmas, some of independent interest. We've seen that $H(x)$ can be effectively computed as a limit "from above". The next lemma shows that $H(x)$ is nonetheless far from being recursive.

Lemma: There does not exists a partial recursive $f : \mathbb{B} \to \mathbb{N}$ with an infinite domain such that $f(x) = H(x)$ whenever $f(x)$ converges.

Proof: Suppose f has all the stated properties. Then the range of f must be unbounded, since for any n, there are only finitely many x such that $H(x) = n$. A simple dovetailing argument shows that given any recursive $g : \mathbb{N} \to \mathbb{N}$, there is a recursive $\eta : \mathbb{N} \to \mathbb{B}$ such that for all n,

$$g(n) \leq f(\eta(n)) = H(\eta(n))$$

But $H(\eta(n)) \leq H(n) + O(1)$, for any recursive η. We can now choose g to obtain a contradiction. For example, since $H(x) \leq 2|x| + O(1)$, if $g(n) = 3n$, then the inequality $g(n) \leq H(\eta(n)) \leq H(n) + O(1) \leq 2n + O(1)$ is violated for large enough n. QED

The next lemma reveals another aspect of how hard it is to compute $H(x)$ from x.

Lemma: $H(H(x)?x)$ is unbounded.

Proof: Suppose there is a constant c such that $H(H(x)?x) \leq c$ for all x. We will use this to obtain an f contradicting the previous lemma.

Consider the finite family of partial recursive functions $\{\Psi(s,\cdot) : |s| \leq c\}$. The hypothesis about c says that for any x, one of these functions applied to x yields $H(x)$. The pigeon-hole principle says that one of these functions agrees with $H(x)$ infinitely often. Unfortunately, this does not yield a proof, since the f we are trying to construct must agree with $H(x)$ whenever it converges.

We can weed out some "false values" by computing $H(x)$ "from above". That is, given x, we dovetail all the computations $\Psi(s,x)$ for $|s| \leq c$ along with all the computations $\Psi(s)$ for any s. Whenever we find a k which is an output of the first group of computations, and which we know to be not less than $H(x)$ from the second group of computations, we mark it as a candidate for the true value of $H(x)$.

In other words, we compute the set:

$$K_x = \{k : H(k?x) \leq c \text{ and } H(x) \leq k\}$$

From the discussion above, it is clear that K_x is finite (in fact of cardinality at most $2^{c+1}-1$), and that $H(x) \in K_x$—in fact, $\min K_x = H(x)$.

If we knew how many elements K_x had, we would have an effective means for computing $\min K_x$ (and hence $H(x)$): enumerate elements of K_x until we have them all, and then take the minimum. While we don't know the size of K_x, we have a finite number of guesses (1 to $2^{c+1}-1$), one of which is guaranteed to be correct.

Let $m = \limsup \operatorname{card}(K_x)$, so $\operatorname{card}(K_x) = m$ for infinitely many x, but card$(K_x) \leq m$ for all but a finite number of x. (This use of the lim sup is a form of the pigeon-hole principle.) Define

$$f'(x) = \begin{cases} \min K_x \text{ if } \mathrm{card}(K_x) = m \\ \text{diverges otherwise} \end{cases}$$

It is easy to verify that f' is partial recursive with an infinite domain, and that f' agrees with H on this domain except for a finite number of points. Let f agree with f' except at these points, where f diverges. Then f is a partial recursive function with infinite domain which agrees with H on its domain, contradicting the previous lemma. QED

The function f in this proof is not defined in an intuitionistically acceptable fashion, since m is not effectively computable, nor are the finite number of exceptional points where f' computes the wrong value. Classically speaking, we've shown that a certain partial recursive function exists, but have not constructed it explicitly. For further discussion, see Rogers.

It is now a short step to showing that $H(x?y) + H(y) - H(x, y)$ is not bounded.

Lemma: $H(x, H(x)) = H(x) + O(1)$.

Proof: We know already that $H(x) \leq H(x, H(x)) + O(1)$. Let $\psi(s) = \langle \Psi(s), |s| \rangle$. Since $\Psi(x^*) = x$ and $|x^*| = H(x)$, it follows that $\psi(x^*) = \langle x, H(x) \rangle$, so $H_\psi(x, H(x)) \leq H(x)$, which implies the desired result. QED

Corollary: $H(x?y) + H(y) - H(x, y)$ is not bounded.

Proof: If it were, then we'd have $H(x, H(x)) = H(H(x)?x) + H(x) + O(1)$. But we know already that $H(x, H(x)) = H(x) + O(1)$, so this would imply $H(H(x)?x) = O(1)$, which we have shown to be false. QED

Note by a similar argument that $H(H(x)/x)$ is bounded. Given the definition of $H(x/y)$, this is hardly surprising.

References

[1] Gregory J. Chaitin. On the length of programs for computing finite binary sequences. *Journal of the Association for Computing Machines*, 13:547–569, 1966.

[2] Gregory J. Chaitin. On the length of programs for computing finite binary sequences: Statistical considerations. *Journal of the Association for Computing Machines*, 16:145–159, 1969.

[3] Gregory J. Chaitin. A theory of program size formally identical to information theory. *Journal of the Association for Computing Machines*, 22:329–340, 1975.

[4] Gregory J. Chaitin. *Algorithmic Information Theory.* Cambridge University Press, 1987.

[5] Peter Gács. On the symmetry of algorithmic information. *Soviet Math. Dokl.*, 15:1477–1480, 1974.

[6] Peter Gács. Review of Chaitin's *Algorithmic Information Theory. Journal of Symbolic Logic*, 54(2):624–627, 1989.

[7] A. N. Kolmogorov. Three approaches to the quantitative definition of information. *Problems in Information Transmission*, 1(1):1–7, 1965.

[8] A. N. Kolmogorov. *Grundbegriffe der Wahrscheinlichkeitrechung.* Berlin, 1933.

[9] Leonid A. Levin. Laws of information conservation (non-growth) and aspects of the foundation of probability theory. *Problems in Information Transmission*, 10:206–210, 1974.

[10] Ming Li and Paul M. B. Vitányi. Kolmogorov complexity and its applications. In J. van Leeuwen, editor, *Handbook of Theoretical Computer*

Science, volume A, chapter 4, pages 187–254. Elsevier Science/MIT Press, 1990.

[11] Ming Li and Paul M. B. Vitányi. *An introduction to Kolmogorov Complexity and Its Applications*. Addison Wesley, Reading, MA., to appear.

[12] Per Martin-Löf. The definition of random sequences. *Information and Control*, 9:602–619, 1966.

[13] R. J. Solomonoff. A formal theory of inductive inference. *Information and Control*, 7:1–22 and 224–254, 1964.