G. J. Chaitin

Algorithmic Information Theory

Abstract: This paper reviews algorithmic information theory, which is an attempt to apply information-theoretic and probabilistic ideas to recursive function theory. Typical concerns in this approach are, for example, the number of bits of information required to specify an algorithm, or the probability that a program whose bits are chosen by coin flipping produces a given output. During the past few years the definitions of algorithmic information theory have been reformulated. The basic features of the new formalism are presented here and certain results of R. M. Solovay are reported.

Historical introduction

To our knowledge, the first publication of the ideas of algorithmic information theory was the description of R. J. Solomonoff's ideas given in 1962 by M. L. Minsky in his paper, "Problems of formulation for artificial intelligence" [1]:

"Consider a slightly different form of inductive inference problem. Suppose that we are given a very long 'data' sequence of symbols; the problem is to make a prediction about the future of the sequence. This is a problem familiar in discussion concerning 'inductive probability.' The problem is refreshed a little, perhaps, by introducing the modern notion of universal computer and its associated language of instruction formulas. An instruction sequence will be considered acceptable if it causes the computer to produce a sequence, perhaps infinite, that *begins* with the given finite 'data' sequence. Each acceptable instruction sequence thus makes a prediction, and Occam's razor would choose the simplest such sequence and advocate its prediction. (More generally, one could weight the different predictions by weights associated with the simplicities of the instructions.) If the simplicity function is just the length of the instruction, we are then trying to find a minimal description, i.e., an optimally efficient encoding of the data sequence.

"Such an induction method could be of interest only if one could show some significant invariance with respect to choice of defining universal machine. There is no such invariance for a fixed pair of data strings. For one could design a machine which would yield the entire first string with a very small input, and the second string only for some very complex input. On the brighter side, one can see that in a sense the induced structure on the space of data strings has some invariance in an 'in the large' or 'almost everywhere' sense. Given two different universal machines, the induced structures cannot be desperately different. We appeal to the 'translation theorem' whereby an arbitrary instruction formula for one machine may be converted into an equivalent instruction formula for the other machine by the addition of a constant prefix text. This text instructs the second machine to simulate the behavior of the first machine in operating on the remainder of the input text. Then for data strings much larger than this translation text (and its inverse) the choice between the two machines cannot greatly affect the induced structure. It would be interesting to see if these intuitive notions could be profitably formalized.

"Even if this theory can be worked out, it is likely that it will present overwhelming computational difficulties in application. The recognition problem for minimal descriptions is, in general, unsolvable, and a practical induction machine will have to use heuristic methods. [In this connection it would be interesting to write a program to play R. Abbott's inductive card game [2].]"

Algorithmic information theory originated in the independent work of Solomonoff (see [1, 3–6]), of A. N. Kolmogorov and P. Martin-Löf (see [7–14]), and of G. J. Chaitin (see [15–26]). Whereas Solomonoff weighted together all the programs for a given result into a probability measure, Kolmogorov and Chaitin concentrated their attention on the size of the smallest program. Recently it has been realized by Chaitin and independently by L. A. Levin that if programs are stipulated to be self-delimiting, these two differing approaches become essentially equivalent. This paper attempts to cast into a unified scheme the recent work in this area by Chaitin [23, 24] and by R. M. Solovay [27, 28]. The reader may also find it interesting to examine the parallel efforts of Levin (see [29–35]). There has been a sub-

stantial amount of other work in this general area, often involving variants of the definitions deemed more suitable for particular applications (see, e.g., [36–47]).

Algorithmic information theory of finite computations [23]

• *Definitions*

Let us start by considering a class of Turing machines with the following characteristics. Each Turing machine has three tapes: a program tape, a work tape, and an output tape. There is a scanning head on each of the three tapes. The program tape is read-only and each of its squares contains a 0 or a 1. It may be shifted in only one direction. The work tape may be shifted in either direction and may be read and erased, and each of its squares contains a blank, a 0, or a 1. The work tape is initially blank. The output tape may be shifted in only one direction. Its squares are initially blank, may have a 0, a 1, or a comma written on them, and cannot be rewritten. Each Turing machine of this type has a finite number n of states, and is defined by an $n \times 3$ table, which gives the action to be performed and the next state as a function of the current state and the contents of the square of the work tape that is currently being scanned. The first state in this table is by convention the initial state. There are eleven possible actions: halt, shift work tape left/right, write blank/0/1 on work tape, read square of program tape currently being scanned and copy onto square of work tape currently being scanned and then shift program tape, write 0/1/comma on output tape and then shift output tape, and consult oracle. The oracle is included for the purpose of defining relative concepts. It enables the Turing machine to choose between two possible state transitions, depending on whether or not the binary string currently being scanned on the work tape is in a certain set, which for now we shall take to be the null set.

From each Turing machine M of this type we define a *probability* P, an *entropy* H, and a *complexity* I. $P(s)$ is the probability that M eventually halts with the string s written on its output tape if each square of the program tape is filled with a 0 or a 1 by a separate toss of an unbiased coin. By "string" we shall always mean a finite binary string. From the probability $P(s)$ we obtain the entropy $H(s)$ by taking the negative base-two logarithm, i.e., $H(s)$ is $-\log_2 P(s)$. A string p is said to be a *program* if when it is written on M's program tape and M starts computing scanning the first bit of p, then M eventually halts after reading all of p and without reading any other squares of the tape. A program p is said to be a *minimal program* if no other program makes M produce the same output and has a smaller size. And finally the complexity $I(s)$ is defined to be the least n such that for some contents of its program tape M eventually halts with s written on the output tape after reading precisely n squares of the program tape; i.e., $I(s)$ is the size of a minimal program for s. To summarize, P is the probability that M calculates s given a random program, H is $-\log_2 P$, and I is the minimum number of bits required to specify an algorithm for M to calculate s.

It is important to note that blanks are not allowed on the program tape, which is imagined to be entirely filled with 0's and 1's. Thus programs are not followed by endmarker blanks. This forces them to be self-delimiting; a program must indicate within itself what size it has. Thus no program can be a prefix of another one, and the programs for M form what is known as a prefix-free set or an instantaneous code. This has two very important effects: It enables a natural probability distribution to be defined on the set of programs, and it makes it possible for programs to be built up from subroutines by concatenation. Both of these desirable features are lost if blanks are used as program endmarkers. This occurs because there is no natural probability distribution on programs with endmarkers; one, of course, makes all programs of the same size equiprobable, but it is also necessary to specify in some arbitrary manner the probability of each particular size. Moreover, if two subroutines with blanks as endmarkers are concatenated, it is necessary to include additional information indicating where the first one ends and the second begins.

Here is an example of a specific Turing machine M of the above type. M counts the number n of 0's up to the first 1 it encounters on its program tape, then transcribes the next n bits of the program tape onto the output tape, and finally halts. So M outputs s iff it finds length(s) 0's followed by a 1 followed by s on its program tape. Thus $P(s) = \exp_2(-2 \text{ length}(s) - 1)$, $H(s) = 2 \text{ length}(s) + 1$, and $I(s) = 2 \text{ length}(s) + 1$. Here $\exp_2(x)$ is the base-two exponential function 2^x. Clearly this is a very special-purpose computer which embodies a very limited class of algorithms and yields uninteresting functions P, H, and I.

On the other hand it is easy to see that there are "general-purpose" Turing machines that maximize P and minimize H and I; in fact, consider those universal Turing machines which will simulate an arbitrary Turing machine if a suitable prefix indicating the machine to simulate is added to its programs. Such Turing machines yield essentially the same P, H, and I. We therefore pick, somewhat arbitrarily, a particular one of these, U, and the definitive definition of P, H, and I is given in terms of it. The universal Turing machine U works as follows. If U finds i 0's followed by a 1 on its program tape, it simulates the computation that the ith Turing machine of the above type performs upon reading the remainder of the program tape. By the ith Turing ma-

chine we mean the one that comes ith in a list of all possible defining tables in which the tables are ordered by size (i.e., number of states) and lexicographically among those of the same size. With this choice of Turing machine, P, H, and I can be dignified with the following titles: $P(s)$ is the *algorithmic probability* of s, $H(s)$ is the *algorithmic entropy* of s, and $I(s)$ is the *algorithmic information* of s. Following Solomonoff [3], $P(s)$ and $H(s)$ may also be called the a priori probability and entropy of s. $I(s)$ may also be termed the descriptive, program-size, or information-theoretic complexity of s. And since P is maximal and H and I are minimal, the above choice of special-purpose Turing machine shows that $P(s) \geq \exp_2(-2 \text{ length}(s) - O(1))$, $H(s) \leq 2 \text{ length}(s) + O(1)$, and $I(s) \leq 2 \text{ length}(s) + O(1)$.

We have defined $P(s)$, $H(s)$, and $I(s)$ for individual strings s. It is also convenient to consider computations which produce finite sequences of strings. These are separated by commas on the output tape. One thus defines the joint probability $P(s_1, \cdots, s_n)$, the joint entropy $H(s_1, \cdots, s_n)$, and the joint complexity $I(s_1, \cdots, s_n)$ of an n-tuple $s_1, \cdots, s_n$. Finally one defines the conditional probability $P(t_1, \cdots, t_m | s_1, \cdots, s_n)$ of the m-tuple $t_1, \cdots, t_m$ given the n-tuple $s_1, \cdots, s_n$ to be the quotient of the joint probability of the n-tuple and the m-tuple divided by the joint probability of the n-tuple. In particular $P(t|s)$ is defined to be $P(s,t)/P(s)$. And of course the conditional entropy is defined to be the negative base-two logarithm of the conditional probability. Thus by definition $H(s,t) = H(s) + H(t|s)$. Finally, in order to extend the above definitions to tuples whose members may either be strings or natural numbers, we identify the natural number n with its binary representation.

• *Basic relationships*

We now review some basic properties of these concepts. The relation

$$H(s,t) = H(t,s) + O(1)$$

states that the probability of computing the pair s, t is essentially the same as the probability of computing the pair t, s. This is true because there is a prefix that converts any program for one of these pairs into a program for the other one. The inequality

$$H(s) \leq H(s,t) + O(1)$$

states that the probability of computing s is not less than the probability of computing the pair s, t. This is true because a program for s can be obtained from any program for the pair s, t by adding a fixed prefix to it. The inequality

$$H(s,t) \leq H(s) + H(t) + O(1)$$

states that the probability of computing the pair s, t is not less than the product of the probabilities of computing s and t, and follows from the fact that programs are self-delimiting and can be concatenated. The inequality

$$O(1) \leq H(t|s) \leq H(t) + O(1)$$

is merely a restatement of the previous two properties. However, in view of the direct relationship between conditional entropy and relative complexity indicated below, this inequality also states that being told something by an oracle cannot make it more difficult to obtain t. The relationship between entropy and complexity is

$$H(s) = I(s) + O(1),$$

i.e., the probability of computing s is essentially the same as $1/\exp_2(\text{the size of a minimal program for } s)$. This implies that a significant fraction of the probability of computing s is contributed by its minimal programs, and that there are few minimal or near-minimal programs for a given result. The relationship between conditional entropy and relative complexity is

$$H(t|s) = I_s(t) + O(1).$$

Here $I_s(t)$ denotes the complexity of t relative to a set having a single element which is a minimal program for s. In other words,

$$I(s,t) = I(s) + I_s(t) + O(1).$$

This relation states that one obtains what is essentially a minimal program for the pair s, t by concatenating the following two subroutines:

- a minimal program for s
- a minimal program for calculating t using an oracle for the set consisting of a minimal program for s.

• *Algorithmic randomness*

Consider an arbitrary string s of length n. From the fact that $H(n) + H(s|n) = H(n,s) = H(s) + O(1)$, it is easy to show that $H(s) \leq n + H(n) + O(1)$, and that less than $\exp_2(n - k + O(1))$ of the s of length n satisfy $H(s) < n + H(n) - k$. It follows that for most s of length n, $H(s)$ is approximately equal to $n + H(n)$. These are the most complex strings of length n, the ones which are most difficult to specify, the ones with highest entropy, and they are said to be the *algorithmically random* strings of length n. Thus a typical string s of length n will have $H(s)$ close to $n + H(n)$, whereas if s has pattern or can be distinguished in some fashion, then it can be compressed or coded into a program that is considerably smaller. That $H(s)$ is usually $n + H(n)$ can be thought of as follows: In order to specify a typical string s of length n, it is necessary first to specify its size n, which requires $H(n)$ bits, and it is necessary then to specify each of the n bits in s, which requires n more bits and brings the total

to $n + H(n)$. In probabilistic terms this can be stated as follows: the sum of the probabilities of all the strings of length n is essentially equal to $P(n)$, and most strings s of length n have probability $P(s)$ essentially equal to $P(n)/2^n$. On the other hand, one of the strings of length n that is least random and that has most pattern is the string consisting entirely of 0's. It is easy to see that this string has entropy $H(n) + O(1)$ and probability essentially equal to $P(n)$, which is another way of saying that almost all the information in it is in its length. Here is an example in the middle: If p is a minimal program of size n, then it is easy to see that $H(p) = n + O(1)$ and $P(p)$ is essentially 2^{-n}. Finally it should be pointed out that since $H(s) = H(n) + H(s|n) + O(1)$ if s is of length n, the above definition of randomness is equivalent to saying that the most random strings of length n have $H(s|n)$ close to n, while the least random ones have $H(s|n)$ close to 0.

Later we shall show that even though most strings are algorithmically random, i.e., have nearly as much entropy as possible, an inherent limitation of formal axiomatic theories is that a lower bound n on the entropy of a specific string can be established only if n is less than the entropy of the axioms of the formal theory. In other words, it is possible to prove that a specific object is of complexity greater than n only if n is less than the complexity of the axioms being employed in the demonstration. These statements may be considered to be an information-theoretic version of Gödel's famous incompleteness theorem.

Now let us turn from finite random strings to infinite ones, or equivalently, by invoking the correspondence between a real number and its dyadic expansion, to random reals. Consider an infinite string X obtained by flipping an unbiased coin, or equivalently a real x uniformly distributed in the unit interval. From the preceding considerations and the Borel-Cantelli lemma it is easy to see that with probability one there is a c such that $H(X_n) > n - c$ for all n, where X_n denotes the first n bits of X, that is, the first n bits of the dyadic expansion of x. We take this property to be our definition of an algorithmically random infinite string X or real x.

Algorithmic randomness is a clear-cut property for infinite strings, but in the case of finite strings it is a matter of degree. If a cutoff were to be chosen, however, it would be well to place it at about the point at which $H(s)$ is equal to length(s). Then an infinite random string could be defined to be one for which all initial segments are finite random strings, within a certain tolerance.

Now consider the real number Ω defined as the halting probability of the universal Turing machine U that we used to define P, H, and I; i.e., Ω is the probability that U eventually halts if each square of its program tape is filled with a 0 or a 1 by a separate toss of an unbiased coin. Then it is not difficult to see that Ω is in fact an algorithmically random real, because if one were given the first n bits of the dyadic expansion of Ω, then one could use this to tell whether each program for U of size less than n ever halts or not. In other words, when written in binary the probability of halting Ω is a random or incompressible infinite string. Thus the basic theorem of recursive function theory that the halting problem is unsolvable corresponds in algorithmic information theory to the theorem that the probability of halting is algorithmically random if the program is chosen by coin flipping.

This concludes our review of the most basic facts regarding the probability, entropy, and complexity of finite objects, namely strings and tuples of strings. Before presenting some of Solovay's remarkable results regarding these concepts, and in particular regarding Ω, we would like to review the most important facts which are known regarding the probability, entropy, and complexity of infinite objects, namely recursively enumerable sets of strings.

Algorithmic information theory of infinite computations [24]

In order to define the probability, entropy, and complexity of r.e. (recursively enumerable) sets of strings it is necessary to consider unending computations performed on our standard universal Turing machine U. A computation is said to produce an r.e. set of strings A if all the members of A and only members of A are eventually written on the output tape, each followed by a comma. It is important that U not be required to halt if A is finite. The members of the set A may be written in arbitrary order, and duplications are ignored. A technical point: If there are only finitely many strings written on the output tape, and the last one is infinite or is not followed by a comma, then it is considered to be an "unfinished" string and is also ignored. Note that since computations may be endless, it is now possible for a semi-infinite portion of the program tape to be read.

The definitions of the probability $P(A)$, the entropy $H(A)$, and the complexity $I(A)$ of an r.e. set of strings A may now be given. $P(A)$ is the probability that U produces the output set A if each square of its program tape is filled with a 0 or a 1 by a separate toss of an unbiased coin. $H(A)$ is the negative base-two logarithm of $P(A)$. And $I(A)$ is the size in bits of a minimal program that produces the output set A, i.e., $I(A)$ is the least n such that there is a program tape contents that makes U undertake a computation in the course of which it reads precisely n squares of the program tape and produces the set of strings A. In order to define the joint and conditional probability and entropy we need a mechanism for encoding two r.e. sets A and B into a single set A *join*

B. To obtain A *join* B one prefixes each string in A with a 0 and each string in B with a 1 and takes the union of the two resulting sets. Enumerating A *join* B is equivalent to simultaneously enumerating A and B. So the joint probability $P(A, B)$ is $P(A \textit{ join } B)$, the joint entropy $H(A,B)$ is $H(A \textit{ join } B)$, and the joint complexity $I(A,B)$ is $I(A \textit{ join } B)$. These definitions can obviously be extended to more than two r.e. sets, but it is unnecessary to do so here. Lastly, the conditional probability $P(B|A)$ of B given A is the quotient of $P(A,B)$ divided by $P(A)$, and the conditional entropy $H(B|A)$ is the negative base-two logarithm of $P(B|A)$. Thus by definition $H(A,B) = H(A) + H(B|A)$.

As before, one obtains the following basic inequalities:

$$H(A,B) = H(B,A) + O(1),$$

$$H(A) \leq H(A,B) + O(1),$$

$$H(A, B) = H(B,A) + O(1),$$

$$O(1) \leq H(B|A) \leq H(B) + O(1),$$

$$I(A,B) \leq I(A) + I(B) + O(1).$$

In order to demonstrate the third and the fifth of these relations one imagines two unending computations to be occurring simultaneously. Then one interleaves the bits of the two programs in the order in which they are read. Putting a fixed size prefix in front of this, one obtains a single program for performing both computations simultaneously whose size is $O(1)$ plus the sum of the sizes of the original programs.

So far things look much as they did for individual strings. But the relationship between entropy and complexity turns out to be more complicated for r.e. sets than it was in the case of individual strings. Obviously the entropy $H(A)$ is always less than or equal to the complexity $I(A)$, because of the probability contributed by each minimal program for A:

$$H(A) \leq I(A).$$

But how about bounds on $I(A)$ in terms of $H(A)$? First of all, it is easy to see that if A is a singleton set whose only member is the string s, then $H(A) = H(s) + O(1)$ and $I(A) = I(s) + O(1)$. Thus the theory of the algorithmic information of individual strings is contained in the theory of the algorithmic information of r.e. sets as the special case of sets having a single element:

For singleton A, $I(A) = H(A) + O(1)$.

There is also a close but not an exact relationship between H and I in the case of sets consisting of initial segments of the set of natural numbers (recall we identify the natural number n with its binary representation). Let us use the adjective "initial" for any set consisting of all natural numbers less than a given one:

For initial A, $I(A) = H(A) + O(\log H(A))$.

Moreover, it is possible to show that there are infinitely many initial sets A for which $I(A) > H(A) + O(\log H(A))$. This is the greatest known discrepancy between I and H for r.e. sets. It is demonstrated by showing that occasionally the number of initial sets A with $H(A) < n$ is appreciably greater than the number of initial sets A with $I(A) < n$. On the other hand, with the aid of a crucial game-theoretic lemma of D. A. Martin, Solovay [28] has shown that

$$I(A) \leq 3\, H(A) + O(\log H(A)).$$

These are the best results currently known regarding the relationship between the entropy and the complexity of an r.e. set; clearly much remains to be done. Furthermore, what is the relationship between the conditional entropy and the relative complexity of r.e. sets? And how many minimal or near-minimal programs for an r.e. set are there?

We would now like to mention some other results concerning these concepts. Solovay has shown that:

There are $\exp_2(n - H(n) + O(1))$ singleton sets A with $H(A) < n$,

There are $\exp_2(n - H(n) + O(1))$ singleton sets A with $I(A) < n$.

We have extended Solovay's result as follows:

There are $\exp_2(n - H'(n) + O(1))$ finite sets A with $H(A) < n$,

There are $\exp_2(n - H(L_n) + O(\log H(L_n)))$ sets A with $I(A) < n$,

There are $\exp_2(n - H'(L_n) + O(\log H'(L_n)))$ sets A with $H(A) < n$.

Here L_n is the set of natural numbers less than n, and H' is the entropy relative to the halting problem; if U is provided with an oracle for the halting problem instead of one for the null set, then the probability, entropy, and complexity measures one obtains are P', H', and I' instead of P, H, and I. Two final results:

$$I'(A, \text{the complement of } A) \leq H(A) + O(1);$$

the probability that the complement of an r.e. set has cardinality n is essentially equal to the probability that a set r.e. in the halting problem has cardinality n.

More advanced results [27]

The previous sections outline the basic features of the new formalism for algorithmic information theory obtained by stipulating that programs be self-delimiting instead of having endmarker blanks. Error terms in the basic relations which were logarithmic in the previous approach [9] are now of the order of unity.

In the previous approach the complexity of n is usually $\log_2 n + O(1)$, there is an information-theoretic char-

acterization of recursive infinite strings [25, 26], and much is known about complexity oscillations in random infinite strings [14]. The corresponding properties in the new approach have been elucidated by Solovay in an unpublished paper [27]. We present some of his results here. For related work by Solovay, see the publications [28, 48, 49].

• *Recursive bounds on $H(n)$*

Following [23, p. 337], let us consider recursive upper and lower bounds on $H(n)$. Let f be an unbounded recursive function, and consider the series $\Sigma \exp_2(-f(n))$ summed over all natural numbers n. If this infinite series converges, then $H(n) < f(n) + O(1)$ for all n. And if it diverges, then the inequalities $H(n) > f(n)$ and $H(n) < f(n)$ each hold for infinitely many n. Thus, for example, for any $\epsilon > 0$, $H(n) < \log n + \log\log n + (1+\epsilon)\,\log\log\log n + O(1)$ for all n, and $H(n) > \log n + \log\log n + \log\log\log n$ for infinitely many n, where all logarithms are base two. See [50] for the results on convergence used to prove this.

Solovay has obtained the following results regarding recursive upper bounds on H, i.e., recursive h such that $H(n) < h(n)$ for all n. First he shows that there is a recursive upper bound on H which is almost correct infinitely often, i.e., $|H(n) - h(n)| < c$ for infinitely many values of n. In fact, the lim sup of the fraction of values of i less than n such that $h(i)|H(i) - h(i)| < c$ is greater than 0. However, he also shows that the values of n for which $|H(n) - h(n)| < c$ must in a certain sense be quite sparse. In fact, he establishes that if h is any recursive upper bound on H then there cannot exist a tolerance c and a recursive function f such that there are always at least n different natural numbers i less than $f(n)$ at which $h(i)$ is within c of $H(i)$. It follows that the lim inf of the fraction of values of i less than n such that $|H(i) - h(i)| < c$ is zero.

The basic idea behind his construction of h is to choose f so that $\Sigma \exp_2(-f(n))$ converges "as slowly" as possible. As a byproduct he obtains a recursive convergent series of rational numbers Σa_n such that if Σb_n is any recursive convergent series of rational numbers, then lim sup a_n/b_n is greater than zero.

• *Nonrecursive infinite strings with simple initial segments*

At the high-order end of the complexity scale for infinite strings are the random strings, and the recursive strings are at the low order end. Is anything else there? More formally, let X be an infinite binary string, and let X_n be the first n bits of X. If X is recursive, then we have $H(X_n) = H(n) + O(1)$. What about the converse, i.e., what can be said about X given only that $H(X_n) = H(n) + O(1)$? Obviously $H(X_n) = H(n, X_n) + O(1) = H(n) + H(X_n|n) + O(1)$. So $H(X_n) = H(n) + O(1)$ iff $H(X_n|n) = O(1)$. Then using a relativized version of the proof in [37, pp. 525–526], one can show that X is recursive in the halting problem. Moreover, by using a priority argument Solovay is actually able to construct a nonrecursive X that satisfies $H(X_n) = H(n) + O(1)$.

• *Equivalent definitions of an algorithmically random real*

Pick a recursive enumeration $O_0, O_1, O_2, \cdots$ of all open intervals with rational endpoints. A sequence of open sets $U_0, U_1, U_2, \cdots$ is said to be *simultaneously r.e.* if there is a recursive function h such that U_n is the union of those O_i whose index i is of the form $h(n,j)$, for some natural number j. Consider a real number x in the unit interval. We say that x has the Solovay randomness property if the following holds. Let $U_0, U_1, U_2, \cdots$ be any simultaneously r.e. sequence of open sets such that the sum of the usual Lebesgue measure of the U_n converges. Then x is in only finitely many of the U_n. We say that x has the Chaitin randomness property if there is a c such that $H(X_n) > n - c$ for all n, where X_n is the string consisting of the first n bits of the dyadic expansion of x. Solovay has shown that these randomness properties are equivalent to each other, and that they are also equivalent to Martin-Löf's definition [10] of randomness.

• *The entropy of initial segments of algorithmically random and of Ω-like reals*

Consider a random real x. By the definition of randomness, $H(X_n) > n + O(1)$. On the other hand, for any infinite string X, random or not, we have $H(X_n) \leq n + H(n) + O(1)$. Solovay shows that the above bounds are each sometimes sharp. More precisely, consider a random X and a recursive function f such that $\Sigma \exp_2(-f(n))$ diverges (e.g., $f(n)$ = integer part of $\log_2 n$). Then there are infinitely many natural numbers n such that $H(X_n) \geq n + f(n)$. And consider an unbounded monotone increasing recursive function g (e.g., $g(n)$ = integer part of $\log\log n$). There are infinitely many natural numbers n such that it is simultaneously the case that $H(X_n) \leq n + g(n)$ and $H(n) \geq f(n)$.

Solovay has obtained much more precise results along these lines about Ω and a class of reals which he calls "Ω-like." A real number is said to be an *r.e. real* if the set of all rational numbers less than it is an r.e. subset of the rational numbers. Roughly speaking, an r.e. real x is Ω-like if for any r.e. real y one can get in an effective manner a good approximation to y from any good approximation to x, and the quality of the approximation to y is at most $O(1)$ binary digits worse than the quality of the approximation to x. The formal definition of Ω-like is as follows. The real x is said to *dominate* the real y if

there is a partial recursive function f and a constant c with the property that if q is any rational number that is less than x, then $f(q)$ is defined and is a rational number that is less than y and satisfies the inequality $c|x-q| \geq |y-f(q)|$. And a real number is said to be Ω-*like* if it is an r.e. real that dominates all r.e. reals. Solovay proves that Ω is in fact Ω-like, and that if x and y are Ω-like, then $H(X_n) = H(Y_n) + O(1)$, where X_n and Y_n are the first n bits in the dyadic expansions of x and y. It is an immediate corollary that if x is Ω-like then $H(X_n) = H(\Omega_n) + O(1)$, and that all Ω-like reals are algorithmically random. Moreover Solovay shows that the algorithmic probability $P(s)$ of any string s is always an Ω-like real.

In order to state Solovay's results contrasting the behavior of $H(\Omega_n)$ with that of $H(X_n)$ for a typical real number x, it is necessary to define two extremely slowly growing monotone functions α and α'. $\alpha(n) = \min H(j)$ $(j \geq n)$, and α' is defined in the same manner as α except that H is replaced by H', the algorithmic entropy relative to the halting problem. It can be shown (see [29, pp. 90–91]) that α goes to infinity, but more slowly than any monotone partial recursive function does. More precisely, if f is an unbounded nondecreasing partial recursive function, then $\alpha(n)$ is less than $f(n)$ for almost all n for which $f(n)$ is defined. Similarly α' goes to infinity, but more slowly than any monotone partial function recursive in α does. More precisely, if f is an unbounded nondecreasing partial function recursive in the halting problem, then $\alpha'(n)$ is less than $f(n)$ for almost all n for which $f(n)$ is defined. In particular, $\alpha'(n)$ is less than $\alpha(\alpha(n))$ for almost all n.

We can now state Solovay's results. Consider a real number x uniformly distributed in the unit interval. With probability one there is a c such that $H(X_n) > n + H(n) - c$ holds for infinitely many n. And with probability one, $H(X_n) > n + \alpha(n) + O(\log\alpha(n))$. Whereas if x is Ω-like, then the following occurs: $H(X_n) < n + H(n) - \alpha(n) + O(\log\alpha(n))$, and for infinitely many n we have $H(X_n) < n + \alpha'(n) + O(\log\alpha'(n))$. This shows that the complexity of initial segments of the dyadic expansions of Ω-like reals is atypical. It is an open question whether $H(\Omega_n) - n$ tends to infinity; Solovay suspects that it does.

Algorithmic information theory and metamathematics

There is something paradoxical about being able to prove that a specific finite string is random; this is perhaps captured in the following antinomies from the writings of M. Gardner [51] and B. Russell [52]. In reading them one should interpret "dull," "uninteresting," and "indefinable" to mean "random," and "interesting" and "definable" to mean "nonrandom."

"[Natural] numbers can of course be interesting in a variety of ways. The number 30 was interesting to George Moore when he wrote his famous tribute to 'the woman of 30,' the age at which he believed a married woman was most fascinating. To a number theorist 30 is more likely to be exciting because it is the largest integer such that all smaller integers with which it has no common divisor are prime numbers. . . . The question arises: Are there any uninteresting numbers? We can prove that there are none by the following simple steps. If there are dull numbers, we can then divide all numbers into two sets—interesting and dull. In the set of dull numbers there will be only one number that is the smallest. Since it is the smallest uninteresting number it becomes, *ipso facto*, an interesting number. [Hence there are no dull numbers!]" [51].

"Among transfinite ordinals some can be defined, while others cannot; for the total number of possible definitions is $\aleph_0$, while the number of transfinite ordinals exceeds $\aleph_0$. Hence there must be indefinable ordinals, and among these there must be a least. But this is defined as 'the least indefinable ordinal,' which is a contradiction" [52].

Here is our incompleteness theorem for formal axiomatic theories whose arithmetic consequences are true. The setup is as follows: The axioms are a finite string, the rules of inference are an algorithm for enumerating the theorems given the axioms, and we fix the rules of inference and vary the axioms. Within such a formal theory a specific string cannot be proven to be of entropy more than $O(1)$ greater than the entropy of the axioms of the theory. Conversely, there are formal theories whose axioms have entropy $n + O(1)$ in which it is possible to establish all true propositions of the form "H(specific string) $\geq n$."

Proof Consider the enumeration of the theorems of the formal axiomatic theory in order of the size of their proofs. For each natural number k, let s^* be the string in the theorem of the form "$H(s) \geq n$" with n greater than H(axioms) $+ k$ which appears first in this enumeration. On the one hand, if all theorems are true, then $H(s^*) > H$(axioms) $+ k$. On the other hand, the above prescription for calculating s^* shows that $H(s^*) \leq H$(axioms) $+ H(k) + O(1)$. It follows that $k < H(k) + O(1)$. However, this inequality is false for all $k \geq k^*$, where k^* depends only on the rules of inference. The apparent contradiction is avoided only if s^* does not exist for $k = k^*$, i.e., only if it is impossible to prove in the formal theory that a specific string has H greater than H(axioms) $+ k^*$.

Proof of Converse The set T of all true propositions of the form "$H(s) < k$" is r.e. Choose a fixed enumeration of T without repetitions, and for each natural number n

let s^* be the string in the last proposition of the form "$H(s) < n$" in the enumeration. It is not difficult to see that $H(s^*,n) = n + O(1)$. Let p be a minimal program for the pair s^*, n. Then p is the desired axiom, for $H(p) = n + O(1)$ and to obtain all true propositions of the form "$H(s) \geq n$" from p one enumerates T until all s with $H(s) < n$ have been discovered. All other s have $H(s) \geq n$.

We developed this information-theoretic approach to metamathematics before being in possession of the notion of self-delimiting programs (see [20–22] and also [53]); the technical details are somewhat different when programs have blanks as endmarkers. The conclusion to be drawn from all this is that even though most strings are random, we will never be able to explicitly exhibit a string of reasonable size which demonstrably possesses this property. A less pessimistic conclusion to be drawn is that it is reasonable to measure the power of formal axiomatic theories in information-theoretic terms. The fact that in some sense one gets out of a formal theory no more than one puts in should not be taken too seriously: a formal theory is at its best when a great many apparently independent theorems are shown to be closely interrelated by reducing them to a handful of axioms. In this sense a formal axiomatic theory is valuable for the same reason as a scientific theory; in both cases information is being compressed, and one is also concerned with the tradeoff between the degree of compression and the length of proofs of interesting theorems or the time required to compute predictions.

Algorithmic information theory and biology

Above we have pointed out a number of open problems. In our opinion, however, the most important challenge is to see if the ideas of algorithmic information theory can contribute in some form or manner to theoretical mathematical biology in the style of von Neumann [54], in which genetic information is considered to be an extremely large and complicated program for constructing organisms. We alluded briefly to this in a previous paper [21], and discussed it at greater length in a publication [19] of somewhat limited access.

Von Neumann wished to isolate the basic conceptual problems of biology from the detailed physics and biochemistry of life as we know it. The gist of his message is that it should be possible to formulate mathematically and to answer in a quite general setting such fundamental questions as "How is self-reproduction possible?", "What is an organism?", "What is its degree of organization?", and "How probable is evolution?". He achieved this for the first question; he showed that exact self-reproduction of universal Turing machines is possible in a particular deterministic model universe.

There is such an enormous difference between dead and organized living matter that it must be possible to give a quantitative structural characterization of this difference, i.e., of degree of organization. One possibility [19] is to characterize an organism as a highly interdependent region, one for which the complexity of the whole is much less than the sum of the complexities of its parts. C. H. Bennett [55] has suggested another approach based on the the notion of "logical depth." A structure is deep "if it is superficially random but subtly redundant, in other words, if almost all its algorithmic probability is contributed by slow-running programs. A string's logical depth should reflect the amount of computational work required to expose its buried redundancy." It is Bennett's thesis that "a priori the most probable explanation of 'organized information' such as the sequence of bases in a naturally occurring DNA molecule is that it is the product of an extremely long evolutionary process." For related work by Bennett, see [56].

This, then, is the fundamental problem of theoretical biology that we hope the ideas of algorithmic information theory may help to solve: to set up a nondeterministic model universe, to formally define what it means for a region of space-time in that universe to be an organism and what is its degree of organization, and to rigorously demonstrate that, starting from simple initial conditions, organisms will appear and evolve in degree of organization in a reasonable amount of time and with high probability.

Acknowledgments

The quotation by M. L. Minsky in the first section is reprinted with the kind permission of the publisher American Mathematical Society from *Mathematical Problems in the Biological Sciences, Proceedings of Symposia in Applied Mathematics XIV*, pp. 42-43, copyright © 1962. We are grateful to R. M. Solovay for permitting us to include several of his unpublished results in the section entitled "More advanced results." The quotation by M. Gardner in the section on algorithmic information theory and metamathematics is reprinted with his kind permission, and the quotation by B. Russell in that section is reprinted with permission of the Johns Hopkins University Press. We are grateful to C. H. Bennett for permitting us to present his notion of logical depth in print for the first time in the section on algorithmic information theory and biology.

References

1. M. L. Minsky, "Problems of Formulation for Artificial Intelligence," *Mathematical Problems in the Biological Sciences, Proceedings of Symposia in Applied Mathematics XIV*, R. E. Bellman, ed., American Mathematical Society, Providence, RI, 1962, p. 35.

2. M. Gardner, "An Inductive Card Game," *Sci. Amer.* **200,** No. 6, 160 (1959).
3. R. J. Solomonoff, "A Formal Theory of Inductive Inference," *Info. Control* **7,** 1, 224 (1964).
4. D. G. Willis, "Computational Complexity and Probability Constructions," *J. ACM* **17,** 241 (1970).
5. T. M. Cover, "Universal Gambling Schemes and the Complexity Measures of Kolmogorov and Chaitin," *Statistics Department Report 12*, Stanford University, CA, October, 1974.
6. R. J. Solomonoff, "Complexity Based Induction Systems: Comparisons and Convergence Theorems," *Report RR-329*, Rockford Research, Cambridge, MA, August, 1976.
7. A. N. Kolmogorov, "On Tables of Random Numbers," *Sankhyā* **A25,** 369 (1963).
8. A. N. Kolmogorov, "Three Approaches to the Quantitative Definition of Information," *Prob. Info. Transmission* **1,** No. 1, 1 (1965).
9. A. N. Kolmogorov, "Logical Basis for Information Theory and Probability Theory," *IEEE Trans. Info. Theor.* **IT-14,** 662 (1968).
10. P. Martin-Löf, "The Definition of Random Sequences," *Info. Control* **9,** 602 (1966).
11. P. Martin-Löf, "Algorithms and Randomness," *Intl. Stat. Rev.* **37,** 265 (1969).
12. P. Martin-Löf, "The Literature on von Mises' Kollektivs Revisited," *Theoria* **35,** Part 1, 12 (1969).
13. P. Martin-Löf, "On the Notion of Randomness," *Intuitionism and Proof Theory*, A. Kino, J. Myhill, and R. E. Vesley, eds., North-Holland Publishing Co., Amsterdam, 1970, p. 73.
14. P. Martin-Löf, "Complexity Oscillations in Infinite Binary Sequences," *Z. Wahrscheinlichk. verwand. Geb.* **19,** 225 (1971).
15. G. J. Chaitin, "On the Length of Programs for Computing Finite Binary Sequences," *J. ACM* **13,** 547 (1966).
16. G. J. Chaitin, "On the Length of Programs for Computing Finite Binary Sequences: Statistical Considerations," *J. ACM* **16,** 145 (1969).
17. G. J. Chaitin, "On the Simplicity and Speed of Programs for Computing Infinite Sets of Natural Numbers," *J. ACM* **16,** 407 (1969).
18. G. J. Chaitin, "On the Difficulty of Computations," *IEEE Trans. Info. Theor.* **IT-16,** 5 (1970).
19. G. J. Chaitin, "To a Mathematical Definition of 'Life,'" *ACM SICACT News* **4,** 12 (1970).
20. G. J. Chaitin, "Information-theoretic Limitations of Formal Systems," *J. ACM* **21,** 403 (1974).
21. G. J. Chaitin, "Information-theoretic Computational Complexity," *IEEE Trans. Info. Theor.* **IT-20,** 10 (1974).
22. G. J. Chaitin, "Randomness and Mathematical Proof," *Sci. Amer.* **232,** No. 5, 47 (1975). (Also published in the Japanese and Italian editions of *Sci. Amer.*)
23. G. J. Chaitin, "A Theory of Program Size Formally Identical to Information Theory," *J. ACM* **22,** 329 (1975).
24. G. J. Chaitin, "Algorithmic Entropy of Sets," *Comput. & Math. Appls.* **2,** 233 (1976).
25. G. J. Chaitin, "Information-theoretic Characterizations of Recursive Infinite Strings," *Theoret. Comput. Sci.* **2,** 45 (1976).
26. G. J. Chaitin, "Program Size, Oracles, and the Jump Operation," *Osaka J. Math.*, to be published in Vol. 14, No. 1, 1977.
27. R. M. Solovay, "Draft of a paper . . . on Chaitin's work . . . done for the most part during the period of Sept.–Dec. 1974," unpublished manuscript, IBM Thomas J. Watson Research Center, Yorktown Heights, NY, May, 1975.
28. R. M. Solovay, "On Random R. E. Sets," *Proceedings of the Third Latin American Symposium on Mathematical Logic*, Campinas, Brazil, July, 1976. To be published.
29. A. K. Zvonkin and L. A. Levin, "The Complexity of Finite Objects and the Development of the Concepts of Information and Randomness by Means of the Theory of Algorithms," *Russ. Math. Surv.* **25,** No. 6, 83 (1970).
30. L. A. Levin, "On the Notion of a Random Sequence," *Soviet Math. Dokl.* **14,** 1413 (1973).
31. P. Gač, "On the Symmetry of Algorithmic Information," *Soviet Math. Dokl.* **15,** 1477 (1974). "Corrections," *Soviet Math. Dokl.* **15,** No. 6, v (1974).
32. L. A. Levin, "Laws of Information Conservation (Nongrowth) and Aspects of the Foundation of Probability Theory," *Prob. Info. Transmission* **10,** 206 (1974).
33. L. A. Levin, "Uniform Tests of Randomness," *Soviet Math. Dokl.* **17,** 337 (1976).
34. L. A. Levin, "Various Measures of Complexity for Finite Objects (Axiomatic Description)," *Soviet Math. Dokl.* **17,** 522 (1976).
35. L. A. Levin, "On the Principle of Conservation of Information in Intuitionistic Mathematics," *Soviet Math. Dokl.* **17,** 601 (1976).
36. D. E. Knuth, *Seminumerical Algorithms. The Art of Computer Programming, Volume 2*, Addison-Wesley Publishing Co., Inc., Reading, MA, 1969. See Ch. 2, "Random Numbers," p. 1.
37. D. W. Loveland, "A Variant of the Kolmogorov Concept of Complexity," *Info. Control* **15,** 510 (1969).
38. T. L. Fine, *Theories of Probability–An Examination of Foundations*, Academic Press, Inc., New York, 1973. See Ch. V, "Computational Complexity, Random Sequences, and Probability," p. 118.
39. J. T. Schwartz, *On Programming: An Interim Report on the SETL Project. Installment I: Generalities*, Lecture Notes, Courant Institute of Mathematical Sciences, New York University, 1973. See Item 1, "On the Sources of Difficulty in Programming," p. 1, and Item 2, "A Second General Reflection on Programming," p. 12.
40. T. Kamae, "On Kolmogorov's Complexity and Information," *Osaka J. Math.* **10,** 305 (1973).
41. C. P. Schnorr, "Process Complexity and Effective Random Tests," *J. Comput. Syst. Sci.* **7,** 376 (1973).
42. M. E. Hellman, "The Information Theoretic Approach to Cryptography," Information Systems Laboratory, Center for Systems Research, Stanford University, April, 1974.
43. W. L. Gewirtz, "Investigations in the Theory of Descriptive Complexity," *Courant Computer Science Report 5*, Courant Institute of Mathematical Sciences, New York University, October, 1974.
44. R. P. Daley, "Minimal-program Complexity of Pseudo-recursive and Pseudo-random Sequences," *Math. Syst. Theor.* **9,** 83 (1975).
45. R. P. Daley, "Noncomplex Sequences: Characterizations and Examples," *J. Symbol. Logic* **41,** 626 (1976).
46. J. Gruska, "Descriptional Complexity (of Languages)–A Short Survey," *Mathematical Foundations of Computer Science 1976*, A. Mazurkiewicz, ed., *Lecture Notes in Computer Science 45*, Springer-Verlag, Berlin, 1976, p. 65.
47. J. Ziv, "Coding Theorems for Individual Sequences," undated manuscript, Bell Laboratories, Murray Hill, NJ.
48. R. M. Solovay, "A Model of Set-theory in which Every Set of Reals is Lebesgue Measurable," *Ann. Math.* **92,** 1 (1970).
49. R. Solovay and V. Strassen, "A Fast Monte-Carlo Test for Primality," *SIAM J. Comput.* **6,** 84 (1977).
50. G. H. Hardy, *A Course of Pure Mathematics*, Tenth edition, Cambridge University Press, London, 1952. See Section 218, "Logarithmic Tests of Convergence for Series and Integrals," p. 417.
51. M. Gardner, "A Collection of Tantalizing Fallacies of Mathematics," *Sci. Amer.* **198,** No. 1, 92 (1958).
52. B. Russell, "Mathematical Logic as Based on the Theory of Types," *From Frege to Gödel: A Source Book in Mathematical Logic, 1879–1931*, J. van Heijenoort, ed., Harvard University Press, Cambridge, MA, 1967, p. 153; reprinted from *Amer. J. Math.* **30,** 222 (1908).

53. M. Levin, "Mathematical Logic for Computer Scientists," *MIT Project MAC TR-131*, June, 1974, pp. 145, 153.
54. J. von Neumann, *Theory of Self-reproducing Automata*, University of Illinois Press, Urbana, 1966; edited and completed by A. W. Burks.
55. C. H. Bennett, "On the Thermodynamics of Computation," undated manuscript, IBM Thomas J. Watson Research Center, Yorktown Heights, NY.
56. C. H. Bennett, "Logical Reversibility of Computation," *IBM J. Res. Develop.* **17**, 525 (1973).

Received February 2, 1977; revised March 9, 1977

The author is located at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York 10598.