PANU RAATIKAINEN

ON INTERPRETING CHAITIN'S INCOMPLETENESS THEOREM

ABSTRACT. The aim of this paper is to comprehensively question the validity of the standard way of interpreting Chaitin's famous incompleteness theorem, which says that for every formalized theory of arithmetic there is a finite constant c such that the theory in question cannot prove any particular number to have Kolmogorov complexity larger than c. The received interpretation of theorem claims that the limiting constant is determined by the complexity of the theory itself, which is assumed to be good measure of the strength of the theory.

I exhibit certain strong counterexamples and establish conclusively that the received view is false. Moreover, I show that the limiting constants provided by the theorem do not in any way reflect the power of formalized theories, but that the values of these constants are actually determined by the chosen coding of Turing machines, and are thus quite accidental.

KEY WORDS: algorithmic information theory, incompleteness, Kolmogorov complexity

1. INTRODUCTION

Gregory Chaitin's information-theoretic incompleteness result, or shortly, Chaitin's Theorem, is arguably one of the most famous and most popularized results of logic in, perhaps, the last three decades.[1] It says, roughly, that for every formal system **F**, there is a finite constant c such that **F** cannot prove any true statements of the form $K(w) > c$, even though there are infinitely many w for which this is true. Here $K(w)$ is the algorithmic complexity, or the Kolmogorov complexity, of w, defined below.

Chaitin's Theorem has indeed attracted lots of attention. Martin Davis, a distinguished logician and pioneer of recursive function theory, calls it "a dramatic extension of Gödel's incompleteness result" (1978, p. 265). Davis writes:

> To fully understand the devasting import of this result it is important to realize that there exist rules of proof (presumably sound) for proving statements of the form $I(w) > n$ which include all methods of proof available in ordinary mathematics. . . we are forced to conclude that there is some definite number k_0, such that it is in principle impossible, by ordinary mathematical methods, to prove that any string of bits has complexity greater than k_0. This is a remarkable limitation of mathematics as we know it (1978, p. 266).

Chaitin's result has also generated plenty of discussion about its meaning and philosophical relevance. Chaitin's own interpretation of the result has

Journal of Philosophical Logic **27:** 569–586, 1998.

become almost universally accepted – at least if one judges from the published literature. According to this received view, the theorem shows that in a formalized theory one cannot prove an object to be more complex than the complexity of the theory itself. It is assumed that the algorithmic complexity of the axioms gives a good measure of the power, or information content, of the theory. The constant c is assumed to depend on the complexity of axioms. The finite bound given by the constant c is hence thought to reflect the power, or information content, of the theory.

The aim of this paper is to comprehensively question the validity of this interpretation. Previously, Michiel van Lambalgen (1989) put forward arguments to the same effect. But although his discussion certainly shows that there must be something wrong with the received interpretation, it does not really tell what. Furthermore, it leaves various questions concerning the matter open. And the received view keeps on occurring in the literature, which may mean that van Lambalgen's criticism has not been felt to be clear and conclusive enough. My purpose here is to strengthen the criticism by exhibiting stronger counterexamples and show conclusively that the received view is false. In addition, I answer certain questions concerning the crucial limiting constant of the theorem left open in the existing literature. I also attempt to thoroughly analyze the true nature of the phenomena in question, and try to understand what has misdirected people to the received interpretation.

2. SOME RECURSION THEORETIC PREREQUISITES

I shall first review some basic notions and results of recursion theory that are used below (for details, see e.g. Rogers (1967), Odifreddi (1988)). I assume for simplicity that we have a bijective coding of Turing machines onto $\mathbf{N}$, and enumerate them as $T_0,\ T_1,\ T_2,\ T_3, \ldots$; I use the Rogers' notation to denote the corresponding partial recursive functions by $\varphi_0, \varphi_1, \varphi_2, \varphi_3, \ldots$. I shall also write $T_m(n)\downarrow$, if the Turing machine with code m halts, when given n as an input. When considering only computations without input, I write $T_m\downarrow$ for $T_m(0)\downarrow$. If the output of the computation that halts is y, I abbreviate this by $T_m(n)\downarrow y$, or $T_m\downarrow y$ for $T_m(0)\downarrow y$. If the machine does not halt, I abbreviate this by $T_m(n)\uparrow$, or $T_m\uparrow$ for $T_m(0)\uparrow$. I write $\varphi_n \simeq \varphi_m$ if the respective values of φ_n and φ_m are either both undefined, or both defined and with the same value.

A central result of recursive function theory, due to Kleene (1938), that I shall utilize repeatedly, is the following:

THE FIXED-POINT THEOREM 2.1. *Given a recursive function f, there is an e such that e and $f(e)$ compute the same function, i.e.* $\varphi_e \simeq \varphi_{f(e)}$.

A crucial fact on which all the following considerations are based is the existence of *Universal Turing machine* T_U, a machine that, when given a code number e of any Turing machine and any input n, computes the very same computation that T_e, i.e. $T_U(e, n) \simeq T_e(n)$. To simplify the following considerations, I assume that one can simply concatenate the two inputs of T_U to a single input, having the length $e + n$. As I need this fact in sequel, let me emphasize this: the *cost of simulating* a Turing machine T_e with input n, i.e. $T_e(n)$, by this Universal Turing machine, is simply e, the code of the Turing machine to be simulated.

I next define the concept of *acceptable indices*, due to Rogers (1958, 1967). The "usual" initial coding φ_e is called the standard coding. A *system of indices* is any family ψ of maps ψ^n from ω onto the set of n-ary partial recursive functions. Usually, one can drop the mention of the number of variables.

DEFINITION 2.2. A system of indices ϕ is called *acceptable* if there are total recursive functions f and g such that $\phi_e \simeq \varphi_{f(e)}$ and $\varphi_e \simeq \phi_{g(e)}$.

A system is thus acceptable if it is possible to go effectively from the standard coding to the system, and *vice versa*. Rogers has shown that a system of indices is acceptable iff it satisfies both enumeration and parametrization, and that every acceptable system of indices satisfies the Fixed-Point Theorem. (Cf. Rogers (1967), Odifreddi (1988).)

Hence, one can say that acceptable systems of indices provide the same structure theory for recursive functions as the standard one. Thus, from the point of view of computatiblity, it does not really make any difference which acceptable system of indices one uses.

3. CHAITIN'S INCOMPLETENESS THEOREM

I shall next give the definition of Kolmogorov complexity, or algorithmic complexity, and present Chaitin's Theorem.

DEFINITION 3.1. The *algorithmic complexity* of number x, $K(x)$, is defined as follows: $K(x) = \mu e(\varphi_e(0) \simeq x)$; i.e. it is the least e s.t. $\varphi_e(0) \simeq x$.

There are actually many variant definitions in the literature, allowing inputs, confining to self-delimiting programs etc. But since this paper deals

solely with Chaitin's incompleteness theorem, such niceties of algorithmic information theory need not worry us. The simple definition above is sufficient for this purpose.

Note that $\varphi_x(0) \simeq y$, considered as a relation, is r.e., or Σ_1^0. The algorithmic complexity $K(x) = y$ is defined as $\varphi_y(0) \simeq x \,\&\, \forall z < y(\neg\varphi_z(0) \simeq x)$ and hence $K(x) = y$, as a relation, is Σ_2^0.

In what follows, it is assumed that the formal system **F** is recursively axiomatizable and that it is "sufficiently rich" (contains Robinson-arithmetic **Q**, or **Q** can be interpreted in it). Furthermore, it is assumed that **F** is sound, i.e. its theorems (at least those of the form "$K(x) > y$") are true. More formally, this can be expressed by the following reflection principle: $Prov_{\mathbf{F}}(\ulcorner K(w) > c \urcorner) \Rightarrow K(w) > c$.

If φ is a formula in $L(\mathbf{F})$, the Gödel number of φ is denoted by $\ulcorner\varphi\urcorner$. The arithmetized proof predicate of **F**, saying that x is the Gödel number of a proof of a formula with Gödel number y, is abbreviated by $Prf_{\mathbf{F}}(x, y)$. The provability predicate $Prov_{\mathbf{F}}(y)$ is then defined as $\exists x Prf_{\mathbf{F}}(x, y)$.

I am now ready to state Chaitin's incompleteness theorem and discuss different ways of proving it:

CHAITIN'S THEOREM 3.2. *For every* (*sufficiently rich*) *formalized theory* **F** *there is a constant* c *such that* **F** *does not prove* $K(w) > c$ *for any* w.

Sketch of proof. Let T_m be a Turing machine that operates, informally speaking, as follows:

$$T_m = \text{"Find the least } x \text{ s.t. } Prf_{\mathbf{F}}(x, \ulcorner K(w) > c \urcorner) \text{ for some } w\text{. Print } w\text{."}$$

Let us then pick some number c such that $c > m$. The only crucial thing is that c is a large number that can nevertheless be produced by a simple subroutine that does not much enlarge the size of the machine T_m.

One can show that T_m does not halt (and, hence, that **F** does not prove $K(w) > c$, for any w). For assume $T_m{\downarrow}$. Now $T_m{\downarrow}w \Rightarrow K(w) > c$, because of the soundness assumption. On the other hand, $T_m{\downarrow}w \Rightarrow K(w) \leq m$, by the definition of $K(x)$, and it was assumed that $m < c$; hence $K(w) < c$. *Contradiction.*

Hence $T_m{\uparrow}$, and **F** does not prove $K(w) > c$, for any w.

3.1. *Discussion: The Idea of the Proof*

Like Gödel's proof, this proof makes a positive use of a paradox. To quote Chaitin:

The proof of this closely resembles G. G. Berry's paradox of 'the first natural number which cannot be named in less than a billion words' [P. R.: But this sentence names the very number in 14 words!]. . . The version of Berry's paradox that will do the trick is 'that object having the shortest proof that its algorithmic information content is greater than a billion bits'. . . (1982, p. 949)

In another context, Chaitin explains his theorem as follows:

The result. . . is the following computer program: 'Find a series of binary digits that can be proved to be of a complexity greater than the number of bits of this program.' The program tests all possible proofs in the formal system in order of their size until it encounters the first one proving that a specific binary sequence is of a complexity greater than the number of bits in the program. Then it prints the series it has found and halts. . . . The program supposedly calculates a number that no program of its size should be able to calculate. . . . The absurdity of this conclusion merely demonstrates that the program will never find the number it is designed to look for. . . (1975, p. 52)

The idea of my proof-sketch above follows the proofs given by Davis (1978) and Boolos (in (Boolos and Jeffrey, 1989)) quite closely. Strictly speaking, however, there is an unsatisfactory gap in these proofs. Namely, it is assumed that one can just pick a number c which satisfies our need, but this is left as a principle of faith. I shall next discuss two ways of completing this proof.

3.2. *The Fixed-Point Construction*

The self-referential character of the crucial Turing machine already suggests that we can, effectively, find a suitable number by applying Kleene's Fixed-Point Theorem. This is indeed one way to complete the proof.

Given a number c, one can effectively find (a code number of) the Turing machine T_m that operates as defined above. Let us denote by f the recursive function assigning m to c, i.e. $f\colon c \to m$. Now by the Fixed-Point Theorem one can effectively find an e such that $T_e \simeq T_{f(e)}$. By definition, $T_{f(e)}$ = "Find the least x s.t. $Prf_{\mathbf{F}}(x, \ulcorner K(w) > e \urcorner)$ for some w. Print w."

As $T_e \simeq T_{f(e)}$, T_e = "Find the least x s.t. $Prf_{\mathbf{F}}(x, \ulcorner K(w) > e \urcorner)$ for some w. Print w."

One has thus effectively found an e that does the trick.

3.3. *Chaitin's Construction*

Chaitin himself has used a different strategy to get his limiting constant. Chaitin has published various proofs of his result, all somewhat different. What I give now is, I think, a fair summary of the basic idea of his proofs. Obviously, I have omitted or simplified many fine details.

First, one defines $K_T(n)$, the algorithmic complexity of a number (string) n relative to a specific Turing machine T, to be the size of the shortest program (input) that, when given to T, produces n. Obviously, this may, in many cases, leave some or all numbers without determinate complexity. Next, Chaitin considers a Universal Turing machine T_U that can simulate any specific Turing machine. The "absolute" algorithmic complexity is then defined to be the algorithmic complexity relative to such a T_U (this, of course, coincides with our above definition).

We recall that the *cost of simulating* $T_e(n)$, by our Universal machine T_U, is simply e. This implies the crucial lemma of Chaitin's proof:

LEMMA 3.3. *For any Turing machine T and for any n, $K(n) \leq K_T(n) + c$, where the constant c is simply the code number of T.*

One next defines a specific purpose Turing machine T as follows:

The machine takes as its input ordered pairs $(\ulcorner A \urcorner, k)$ where A is a conjunction of the axioms of the formalized theory in question, and k is a number. (I assume again that we can concatenate $\ulcorner A \urcorner$ and k to a single input, having the size $\ulcorner A \urcorner + k$.) Let T operate, informally speaking, as follows:

T = "Find the least x s.t. $Prf_A(x, \ulcorner K(w) > c \urcorner)$ for some w, where $c = \ulcorner A \urcorner + 2k$. Print w and halt."

It can be assumed that the axioms A are fixed in the following.

One then checks what the code number of this Turing machine T is. Let us denote its code by n. If T halts, the complexity of w relative to T, $K_T(w)$, is by the defition of relative complexity $\leq (\ulcorner A \urcorner + k)$. Note that n, as the code of T, is the cost of simulating T by T_U. Hence the "absolute" complexity $K(w) \leq (\ulcorner A \urcorner + k) + n$, by the above lemma.

One next gives n as the latter member of input for T, i.e. let $k = n$. Hence if T halts, it has found a proof (from A) that $K(w) > c$, where $c = \ulcorner A \urcorner + 2n$. By the soundness assumption, this is indeed the case. Thus, on one hand, $K(w) > \ulcorner A \urcorner + 2n$. But on the other hand, it was seen above that $K(w) \leq (\ulcorner A \urcorner + k) + n$, and because k is now equal to n, $K(w) \leq (\ulcorner A \urcorner + 2n$. *Contradiction.* Hence T cannot halt, and there cannot exist a proof (from A) that $K(w) > c$, where $c = \ulcorner A \urcorner + 2n$, for any w.

3.4. *Discussion: Comparing the Two Methods*

Although Chaitin's way to his result is not the most direct one, one must admit that it is in certain ways very ingenious. It manages to avoid any direct self-reference, and hence one does not need to use the Fixed-Point Theorem or some related advanced result of recursive function theory, that

some may find difficult to understand. Instead, only the quite generally understandable ideas of a Universal Turing machine and the cost of simulating a specific purpose Turing machine are needed. On the other hand, for a logician with good knowledge of recursive function theory, this route may appear unnecessarily complicated. As a conclusion, it might be said that it is a matter of personal taste which way of finding a suitable limiting constant one prefers.

4. THE RECEIVED INTERPRETATION OF CHAITIN'S THEOREM

As was mentioned, Chaitin's Theorem has generated quite a lot of discussion concerning its meaning and philosophical relevance. The interpretation favoured by Chaitin himself has become the received view on the matter, and has been faithfully repeated in the literature (see references in note 1).

The following quotations from Chaitin's publications are, I think, quite presentative:

> . . .I would like to measure the power of a set of axioms and rules of inference. I would like to be able to say that if one has ten pounds of axioms and a twenty-pound theorem, then that theorem cannot be derived from those axioms. (1982, p. 942)

> Since complexity has been defined as a measure of randomness, this theorem implies that in a formal system no number can be proved to be random unless the complexity of the number is less than that of the system itself. (1975, p. 52)

> . . . 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 employded in the demonstration (1977, p. 353). . . . it is reasonable to measure the power of formal axiomatic theories in information-theoretic terms . . . (1977, p. 357)

> It is not the number of bits in the program itself that is the limiting factor but the number of bits in the formal system as a whole. Hidden in the program are the axioms and rules of the inference that determine the behaviour of the system and provide the algorithm for testing proofs. . . . The size of the entire program therefore exeeds the complexity of the formal system by a fixed number of bits c. (The actual value of c depends on the machine language employed.) The theorem proved by the paradox can therefore be stated as follows: In a formal system of complexity n it is impossible to prove that a particular series of binary digits is of a complexity greater than $n + c$, where c is a constant that is independent of the particular system employed. (1975, p. 52)

4.1. *A Summary of the Received Interpretation*

I shall now try to recapitulate and specify the main content of this interpretation. First, following van Lambalgen, I shall call the *minimal* c such that for all w, the formal system **F** does not prove $K(w) > c$, the *characteristic*

constant of the formal system **F**, and denote it by $c_{\mathbf{F}}$. I think that it is useful to analyze the received interpretation as consisting of two separate parts:

(1) One can reasonably measure the power, or the information content, of formal axiomatic theories by the algorithmic complexity of axioms (or, as it is sometimes suggested, axioms and rules of inference taken together).
(2) The limiting constant $c_{\mathbf{F}}$ (the "characteristic constant" of **F**) depends only on the complexity of the axioms of **F**.

These together imply the familiar suggestion that the limiting constant of a theory somehow reflects the power of the theory.

It should be noted that in part (2) there occurs some degree of vagueness: sometimes it is said that it is the length of the axioms that is relevant, sometimes the complexity of axioms, sometimes the size or complexity of axioms and rules of inference taken together. However, my critical discussion below applies to any of these alternatives.

4.2. *Critique of the Received Intepretation*

As mentioned above, Michiel van Lambalgen (1989) has criticized this received view. He says that this interpretation is "at present only scantly supported by facts". According to him, "[w]hat matters is not so much the information content of the formal system S as a whole, but rather that of its intersection S' with the set of statements of the form '$K(w) > m$'". He then refers to a result of Kreisel and Levy (1968), and argues that there are an infinite number of even stronger number theories S_n, which lie between **PA** and **ZF**, and have the same characteristic constant c. Van Lambalgen emphasizes that "we do not even know whether $c_{\mathbf{ZF}} > c_{\mathbf{PA}}$!... and, worse, we even have no idea how to establish results of this sort."

I agree with the basic points of this criticism, but I shall show that much more can be said about this issue. I think that when one reaches the end of this paper it will be clear that not only is the received interpretation "at present only scantly supported by facts", but that it is completely at odds with facts.[2] First, I give a couple of strong counterexamples to the received view, which use admittedly manipulated and *ad hoc*, but still acceptable (in Rogers' sense, see above 2.2), coding systems. These serve as preparation to the more general considerations that follow. My discussion shows also that the last question raised by van Lambalgen, i.e. whether $c_{\mathbf{ZF}} > c_{\mathbf{PA}}$ and how to establish such results, is not actually well defined.

5. HOW TO MAKE THE CHARACTERISTIC CONSTANT ZERO

As an amusing application of the Fixed-Point Theorem, I now show how to collapse the characteristic constant to zero, in any formalized theory.

THEOREM 5.1. *For any formal system* **F**, *it is possible to define an acceptable system of indices in which the characteristic constant* $c_{\mathbf{F}} = 0$.

Proof. Let us fix the formal system **F** containing elementary arithmetic for which we want to prove the theorem. Assume that some initial bijective coding π of Turing machines onto natural numbers is given: they can thus be enumerated as $T_0, T_1, T_2, \ldots$.

Let us next define a code-switching function π^n (relative to given n) as follows:

$$\pi^n(x) = \begin{cases} 0, & \text{if } x = n, \\ x+1, & \text{if } x < n, \\ x, & \text{if } x > n. \end{cases}$$

Obviously, given the initial coding π and the parameter n, one can effectively obtain the new coding determined by π^n. Call the algorithmic complexity relative to this new coding K^n, i.e. $K^n(x) = \mu z(\exists y[\pi^n(y) = z \wedge T_y{\downarrow}x])$.

Algorithmic complexity is arithmetical, and one can construct an arithmetic formula that defines it (for the initial coding). Given this formula, one can then effectively find a formula that defines K^n for given n, simply by formalizing the above definition of π^n. Also, given the Gödel number of the former formula, one can effectively find the Gödel number of the latter formula.

One can then also effectively find the Turing machine T_m (whose code is m in the initial coding) that searches for the least x such that for some number p, $Prf_{\mathbf{F}}(x, \ulcorner K^n(\mathbf{p}) > \mathbf{0} \urcorner)$, and when it finds it (if such a x exists) prints the p for which x proves this.

Now the trick is to find a machine which would operate like this and have its own initial code number as the parameter n in K^n.

Informally, the desired machine could be described as follows: "Starting from 0, check of every number whether it is (a Gödel number of) a proof of a formula of the form $K^e(\mathbf{p}) > \mathbf{0}$) for some number p. If such a theorem is found, print the particular number p for which this fact is proved. And let e be the (initial) code number of *this program.*"

More formally, this self-reference can be handled as follows:

There is a recursive function $f(x)$ such that given the parameter n in K^n, $f(n) = m$ is the (initial) code number m of the Turing machine as

described above. By the Fixed-Point Theorem, there is an e s.t. T_e and $T_{f(e)}$ compute the same function.

Such a T_e is a machine of the desired sort. Moreover, the proof of the Fixed-Point Theorem not only tells us that such a number e exists; it also explicitly produces it. Hence, one can effectively find such an e.

However, it follows that this program will never halt, and hence, one cannot prove $K^e(\mathbf{p}) > 0$, for any p. For if one could prove such a theorem, the program e would halt, and print a number p for which this was provable, by a proof with the smallest Gödel number. Hence the complexity of p is at most the code of T_e. But in the coding determined by π^n it is by definition 0 (i.e. $\pi^e(e) = 0$), and hence $K^e(\mathbf{p}) = 0$. On the other hand, we have proved that $K^e(\mathbf{p}) > 0$. Assuming soundness, this is true. *Contradiction.* This proves the claim. □

5.1. *Discussion: Consequences*

Note that if one applies this construction to some strong system, say, to full infinistic Zermelo-Fraenkel set theory **ZFC**, and the coding of Turing machines is kept fixed, the characteristic constant of every sound theory from the weakest elementary arithmetic (e.g. Robinson's **Q**) to **ZFC** is the same, namely 0.

This shows clearly that, under some acceptable codings, the characteristic constants of theories having radically different stengths, whether measured by proof-theoretical strength or algorithmic complexity of axioms, may be the same.

6. How to make the characteristic constant arbitrarily large

One can also, on the other hand, easily construct a situation where the "characteristic constant" of a formal system, satisfying this time certain extra constraints, becomes arbitrarily large. This constructions shows clearly that a theory may well prove a number to have a complexity larger than the complexity of the theory. I merely give a sketch of the idea.

6.1. *The Construction*

Let **F** be a formalized theory containing elementary arithmetic. Let $T_{\mathbf{F}}$ be a Turing machine that prints the axioms of **F** and halts. Let us simply stipulate that the code of this machine $T_{\mathbf{F}}$ is 1. Assume further that **F** is strong enough to prove that $T_{\mathbf{F}}$ prints its axioms and halts (this does not apper to be a very strong assumption).

The initial state is denoted by q_1. Consider next the Turing machine: $q_2\,0\,0\,q_2$. Obviously this machine does not even start, and this trivial fact is certainly provable in any formalized theory containing elementary arithmetic. Then fix a coding in the following way: let the code of the above machine be 2, and up to some chosen number n (e.g. $n = 10^{10^{10^{10}}}$) let the machine T_k ($k \leq n$) consist of the above instruction $q_2\,0\,0\,q_2$, repeated $k - 1$ times. Obviously, one can then also prove in **F** that none of these machines prints anything.

Then let the machine T_{n+1} consist of the instruction $q_1\ 0\ 1\ q_2$, i.e. the machine prints 1 and halts. That it does so can also be proved in elementary arithmetic. Hence, we can prove in **F** that $K(1) > n$. Therefore, the charateristic constant $c_{\mathbf{F}}$ of the theory **F** considered must be even larger than this. And of course, the number n can be chosen as large as one wants.

On the other hand, the complexity of the axioms (or, in the alternative approach, axioms and rules of inference) under this coding is 1. Hence, under this coding, **F** can prove a specific string to be of a complexity much greater than the complexity of its axioms.

6.2. *Discussion: Natural and Unnatural Codings*

At this point one may protest that the codings I have used above (in 5.1 and 6.1) are quite bizarre and unnatural. This I grant, of course. But these cases exemplify how much depends on the coding system used, and I think that it is not tenable to claim that some unique coding is natural and has a priviledged status, whereas all the others are artificial. Moreover, one cannot even claim that a particular way of representing the class of computable functions (Turing machines, recursive functions, λ-definability, Post systems, URIM-programs, etc.) is priviledged. Further, even if some "canonical" coding technique and an approach to computability is fixed, any change in the abritrary order in which one codes, say, instructions of Turing machines, or recursive functions, may change radically the values of characteristic constants.

In what follows, I shall show that my case against the received interpretation does not ultimately rest on odd coding systems. The evidence below will show beyond reasonable doubt that the received view is simply and unquestionably false.

7. THE TRUE SOURCE OF THE CHARACTERISTIC CONSTANT

Above considerations at least strongly suggest that the value of the "characteristic constant" of a formalized theory does not reflect the power or

the information content of axioms. What, then, one may ask, is the true source of "characteristic constants"? Fortunately, I can give an exact answer to this.

Closer reflection shows that the value of $c_{\mathbf{F}}$ is actually determined simply by the smallest (by its code) Turing machine which does not halt, but for which this cannot be proved in **F**. For consider the smallest e for which $\neg\exists x\varphi_e(0) \simeq x$ is true but cannot be proved in given formal system **F**. It follows that it is impossible to prove in **F** that $\varphi_n(0) \simeq m \,\&\, \forall z < n \neg\varphi_z(0) \simeq m$, for any m and any $n > e$, since one cannot in **F** refute the possibility that e is such a z. That is, one cannot prove that any particular number has a complexity greater than e.

Now it is really hard to see why the code of such a Turing machine would reveal anything interesting about the "power" or "information content" of **F**. Note that it may happen (as our above considerations, as well as van Lambalgen's earlier arguments, already show) that theories of enormously different strength (in any reasonable sense) have this constant in common.

Note, by the way, that this observation provides almost trivial proof of Chaitin's Theorem (although somewhat less constructive, in the sense that it does not exhibit any particular constant), given the well-known fact that no formal system can prove all the true sentences of the form $\neg\exists x\varphi_e(0) \simeq x$.

8. Confusing use and mention

In my analysis of the received interpretation, I isolated the claim that it is reasonable to measure the power or information content of a formal system by the complexity of its axioms (or, alternatively, its axioms and rules of inference). I shall next argue that this claim is highly implausible, and is most likely based on confusion between mention and use.

To begin, I take for granted the basic proof-theoretical notion of strength, according to which theory $\mathbf{F}_1$ is at least as strong as $\mathbf{F}_2$, if $\mathbf{F}_1$ proves every theorem that $\mathbf{F}_2$ proves. It is indeed very difficult to consider seriously any notion of power, strength or information content of a theory that violates this natural notion. But the idea of measuring them by the complexity of axioms is clearly in odds with this intuitive picture. A weak theory may very well be more difficult to axiomatize, i.e. be more complex, than a strong theory properly containing it.

As a ridiculously simple example, take as a theory $\mathbf{F}_1$ some very large and (with respect to given coding) enormously complex finite set of equations of the form $\mathbf{n} = \mathbf{n}$. Now according to the received interpretation, this

theory is extremely powerful or informative. But of course, it is in fact very weak and quite useless. It is contained in the very simple and trivial theory $\mathbf{F}_2$ that consist of the single axiom $(\forall x)(x = x)$.

To take examples of theories in actual use, the full impredicative second order arithmetic $\mathbf{Z}_2$ is in fact simpler to present than most of its weaker subtheories used in logic. Also, the axioms of strictly finitistic primitive recursive arithmetic **PRA** appear to be more complex than the axioms of boldly infinistic Zermelo-Fraenkel set theory **ZFC**, but it is indeed hard to give any interesting sense to the claim that **PRA** is more powerful, or has greater information content, than **ZFC**. Simply stated, power, or information content, of a theory is completely independent from the syntactical difficulty of representing it.

I think there is a clear case of confusion here between use and mention, a distinction whose inportance has been emphasized especially by Quine (1940). He has nicely illustrated this distinction by the example that Boston (word used) contains some 800,000 people, but "Boston" (word mentioned) contains six letters. It was a related confusion in statistical information theory that led Carnap and Bar-Hillel to the fundamental distinction between semantic information and syntactic information, i.e. between the information content of a sentence and the probability of its syntactical presentation (see Bar-Hillel (1964)).

In the present context, the relevance of this distinction is exemplified by the fact that in a suitable language the sentence expressing (used) that a particular object has a very large complexity, e.g. "$K(\mathbf{n}) > \mathbf{m}$" (for a large m), may itself have a quite simple (when mentioned) syntactical form.

Now Chaitin's metaphor that "if one has ten pounds of axioms and a twenty-pound theorem, then that theorem cannot be derived from those axioms", if referring to Chaitin's theorem, seems to commit this confusion, i.e. it compares the complexity of axioms as mentioned and the complexity asserted by a theorem when used. Now one may ask what happens if axioms and theorems are compared in the same level. But of course, one can derive from any axiom system, however simple in its syntactic form, theorems having arbitrarily complex syntactical form. Hence, if one compares the complexity of axioms (mentioned) and theorems (mentioned), the claim is trivially false.

9. ON STRENGTHS AND CHARACTERISTIC CONSTANTS OF THEORIES

Let us now return to the examination of characteristic constants. Recall that it was shown in Section 7 that the value of a characteristic constant is actually determined by the smallest (by its code) Turing machine that does

not halt but for which this cannot be proved. Next, I give some further examples that clearly illuminate the problems of the received interpretation.

First, consider some theory **F** with characteristic constant c. One can then form two extensions of **F**, call them $\mathbf{F}_1$ and $\mathbf{F}_2$, such that we add to $\mathbf{F}_1$ the (true) sentence that T_c does not halt, and to $\mathbf{F}_2$ a similar (true) sentence about some other machine T_d $(c < d)$, i.e. that T_d does not halt. But now $c_{\mathbf{F}_2} = c_{\mathbf{F}}$, while $c_{\mathbf{F}_1} > c_{\mathbf{F}}$. But there is no reason to expect that $\mathbf{F}_1$ is in some sense more powerful, or has larger information content, than $\mathbf{F}_2$. Neither can one grant that the algorithmic complexity of the axioms of $\mathbf{F}_1$ is larger than that of $\mathbf{F}_2$.

Second, consider second order arithmetic $\mathbf{Z}_2$, which is an extremely strong theory, much more than enough to prove any theorem of ordinary mathematics. Let us compare it to extremely weak Robinson arithmetic **Q**. If it happens (this depends on coding, as we have seen) that $c_{\mathbf{Q}} < c_{\mathbf{Z}_2}$, let us add to **Q** all true sentences of the form $\neg\exists x\varphi_e(0) \simeq x$ which are provable in $\mathbf{Z}_2$, for all $e < c_{\mathbf{Z}_2}$. Call this finite extension $\mathbf{Q}^*$. It follows from a result of Kreisel and Levy (1968)[3] that $\mathbf{Q}^*$ cannot possibly come even close to the strength of $\mathbf{Z}_2$. On the other hand, the characteristic constants of $\mathbf{Z}_2$ and $\mathbf{Q}^*$ are the same, and hence, they should, according to the received view, have the same "information content" of "power".

Furthermore, we may add to $\mathbf{Q}^*$ one more true sentence $\neg\exists x\varphi_e(0) \simeq x$ for the first e for which $\mathbf{Z}_2$ does not prove this (i.e. $e = c_{\mathbf{Z}_2}$). Call this theory $\mathbf{Q}^{**}$. Now the characteristic constant of this theory is bigger than that of $\mathbf{Z}_2$. But could one conclude that $\mathbf{Q}^{**}$ is a more "powerful" theory or that it has larger "information content"? No, not in any reasonable sense of the words. Strictly speaking, the proof-theoretical strenghts of $\mathbf{Q}^{**}$ and $\mathbf{Z}_2$ are incomparable, as they both can prove some facts that the other cannot. But in all relevant respects, $\mathbf{Z}_2$ is an enormously more powerful theory than $\mathbf{Q}^{**}$, which can prove hardly any mathematical results, whereas, as was said, all ordinary mathematics can be developed in $\mathbf{Z}_2$.

10. THE COMPLEXITY OF AXIOMS AND THE CHARACTERISTIC CONSTANT

The cases in the two sections above depend at least partly on my (very reasonable, I think) commitment to the proof-theoretical notion of strength. There still remains the question whether one could avoid my criticism by simply stipulatively defining that the information content (or "power") of a formal theory is the algorithmic complexity of its axioms (or axioms and rules of inference). The answer is negative: even this would not save the re-

ceived interpretation; there isn't any connection between the characteristic constant and the algorithmic complexity of the axioms of a theory.

Perharps the simplest consideration that shows this is the following: Given a relatively simple formalized theory, e.g. **Q**, one can construct for it extremely complicated (in the sense of complexity of axioms) *extensions by definitions*. But every such an extension is a *conservative* one, i.e. it proves only the same facts as the original theory (for these notions, see e.g. Shoenfield (1967)), and thus their characteristic constants are the same. Hence the characteristic constant of a theory does not depend at all on the complexity of its axioms (or axioms and rules of inference).

11. DIAGNOSIS OF THE RECEIVED INTERPRETATION

Let us try to understand what are the reasons that have led to the received interpretation. The key to the issue is Chaitin's way to prove his theorem. There, Chaitin uses a specific purpose Turing machine. It is useful to analyze what is really contained in this crucial Turing machine. We can distinguish four different procedures: there are (1) an instruction to find the least x satisfying the following condition; (2) an algorithm for generating, in "size order", all proofs of given theory; (3) an algorithm that examines whether or not a given theorem is of the form $K(w) > c$; and (4) an algorithm that extracts the number w from $\ulcorner K(w) > c \urcorner$, prints w and makes T halt.

These four procedures together make up the crucial Turing machine T. The code number of T, and thus the cost of simulating T, depends on all these "subprograms", how they are put together, and most of all, the coding system used. Note that it depends essentially on both the Gödel numbering of the formalized theory, and then on the coding of Turing machines. Clearly, it is an overstatement to say that the constant obtained by Chaitin's method depends only on the complexity of the axioms and rules of inference, that is, on part (2) alone. (Of course, it is a recursive function of this part, given the rest.)

However, *the fatal mistake* of the received intepretation is to think that Chaitin's proof somehow gives the *minimal* limiting constant, "the characteristic constant", of the theory under consideration. But of course, there is no *a priori* reason to think that the *sufficient* constant effectively obtained by Chaitin's trick is equal to the minimal characteristic constant of the given theory. In fact, one can prove that, in general, it is not.

For assume Chaitin's method does provide us with the minimal such number c. Starting with some simple arithmetic theory (say, Robinson's **Q**), we could find the "smallest" Turing machine, with code c, that does

not halt but for which this cannot be proved. One may then add to **Q** the true sentence $\neg\exists x\varphi_c(0) \simeq x$. Now by applying this procedure repeatedly it would be possible to generate one by one all the Turing machines that do not halt, i.e. the set of Turing machines not halting would be recursively enumerable. As the set of Turing machines that do halt is in fact recursively enumerable, by Post's classical theorem both sets would now be recursive, and we would have an effective method for deciding the halting problem. But the halting problem is, of course, undecidable. Hence Chaitin's method cannot, in general, provide us with the minimal, "characteristic" constant!

I suspect that the reason for thinking otherwise might be some sort of unintented confusion between necessary and sufficient conditions.

12. CONCLUSIONS

I think that my arguments above show convincingly that the received interpretation of Chaitin's incompleteness theorem is false.

Only by confusing use and mention, or the complexity of syntactical form and semantical content, is it possible to maintain that it is reasonable to measure the power of theories by the algorithmic complexity of their axioms. The value of a real limiting constant, the minimal "characteristic constant", does not have any connection to the complexity of axioms, or the rules of inference. Rather the "quantitative measures" obtained in these considerations depend on accidental features of Gödel numberings and codings of Turing machines used.

Nevertheless, I don't think these conclusions imply that Chaitin's incompleteness theorem is not an interesting result. I myself think it is, only a widely misunderstood one. As such, it is in good company with many other important and famous results.

ACKNOWLEDGMENTS

I am very grateful to M. van Lambalgen, P. Milne, I. Niiniluoto, J. von Plato, G. Sandu, J. Väänänen and the members of the Helsinki Logic Group, for encouragement, inspiration, valuable suggestions, and constructive criticism; I also thank P. Milne for helping me to revise my English. Finally, I thank the Emil Aaltonen Foundation for the financial support that has made this work possible.

NOTES

[1] To give some evidence for the claim: Martin Davis has included Chaitin's Theorem in his well known review article on basic results on computability (Davis, 1978), together with the classical result of Post, Turing, Gödel and Church. George Boolos has is turn added Chaitin's Theorem to the third edition of the respected textbook *Computability and Logic* by Boolos and Jeffrey (1989). See also Delahaye (1989).

Chaitin himself has succesfully popularized his result e.g. in Scientific American (Chaitin, 1975). Chaitin's Theorem has also found its way into many influental popular science books; see e.g. Stewart (1992), Rucker (1987), Ruelle (1991). Two papers by Chaitin are even included in a well known recent anthology on the philosophy of mathematics (Tymoczko (ed.) (1986)).

However, I would like to emphasize that my claim concerning the fame of the result should not, as such, be interpreted to claim anything about its actual relevance for logic or its true philosophical depth.

[2] Actually, the arguments given by van Lambalgen justify a much stronger claim than his quoted expression. I think they are strong enough to refute the received interpretation.

[3] This is, by the way, the same result as the one appealed to by van Lambalgen (see 4.2 above), i.e. Theorem 11 in p. 121 of (Kreisel and Levy, 1968).

REFERENCES

Bar-Hillel, Y. (1964). *Language and Information: Selected Essays on Their Theory and Application*, Addison-Wesley and The Jerusalem Academic Press, Reading, MA and Jerusalem.

Boolos, G. S. and Jeffrey, R. C. (1989). *Computability and Logic*, third edition, Cambridge University Press, Cambridge.

Chaitin, G. J. (1974a). Information-theoretic computational complexity, *IEEE Transactions on Information Theory* **IT-20**, 10–15.

Chaitin, G. J. (1974b). Information-theoretic limitations of formal systems, *J. of the ACM* **21**, 403–424.

Chaitin, G. J. (1975). Randomness and mathematical proof, *Scientific American* **232**(5), 47–52.

Chaitin, G. J. (1977). Algorithmic information theory, *IBM Journal of Research and Development* **21**, 350–359, 496.

Chaitin, G. J. (1982). Gödel's theorem and information, *Internat. J. Theoret. Phys.* **22**, 941–954.

Davis, M. (1978). What is a computation?, in *Mathematics Today* (L. A. Stern, ed.), Springer-Verlag, Berlin, 241–267.

Delahaye, J.-P. (1989). Chaitin's equation: an extension of Gödel's theorem, *Notices of the A.M.S.* **36**(8), 984–7.

Kleene, S. K. (1938). On notations for ordinal numbers, *J. Symbolic Logic* **3**, 150–155.

Kreisel, G. and Levy, A. (1968). Reflection principles and their uses for establishing the complexity of axiomatic systems, *Z. Math. Logik Grundlag. Math.* **14**, 97–142.

van Lambalgen, M. (1989). Algorithmic information theory, *J. Symbolic Logic* **54**, 1389–1400.

Odifreddi, P. (1989). *Classical Recursion Theory*, North-Holland, Amsterdam.

Quine, W. V. (1940). *Mathematical Logic*, W. W. Norton, New York.
Rogers, H. (1958). Gödel numbering of partial recursive functions, *J. Symbolic Logic* **23**, 331–341.
Rogers, H. (1967). *Theory of Recursive Functions and Effective Computability*, McGraw-Hill, New York.
Rucker, R. (1987). *Mind Tools*, Houghton Mifflin, Boston.
Ruelle, D. (1991). *Chance and Chaos*, Princeton University Press, Princeton.
Shoenfield, J. R. (1967). *Mathematical Logic*, Addison-Wesley, Reading, MA.
Stewart, I. (1992). *The Problems of Mathematics*, Oxford University Press, Oxford.
Tymoczko, T. (ed.) (1986). *New Directions in the Philosophy of Mathematics*, Birkhäuser, Boston.

Department of Philosophy,
P.O. Box 24,
FIN-00014 University of Helsinki,
Finland
(E-mail: panu.raatikainen@helsinki.fi)