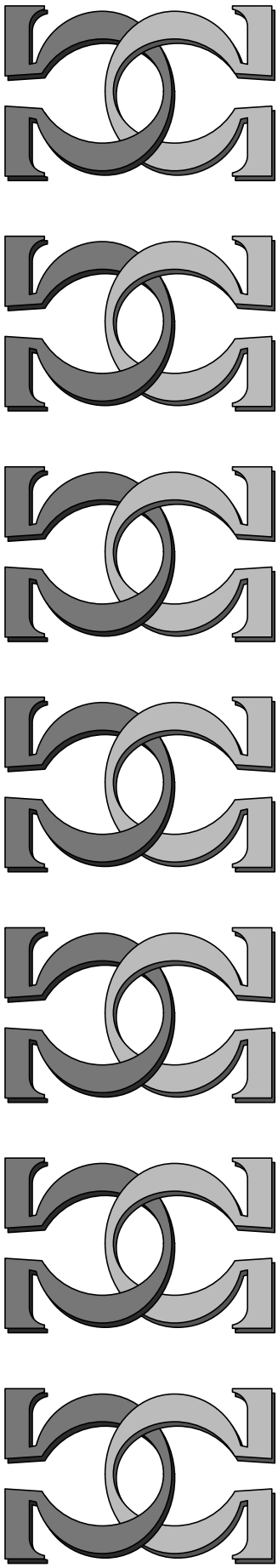

CDMTCS
Research
Report
Series

Solving Problems with Finite Test Sets

Cristian S. Calude
Auckland University, New Zealand

Helmut Jürgensen
University of Western Ontario, Canada

Shane Legg
Auckland University, New Zealand

CDMTCS-112
September 1999

Centre for Discrete Mathematics and
Theoretical Computer Science

Solving Problems with Finite Test Sets*

Cristian S. Calude,[†] Helmut Jürgensen,[‡] Shane Legg[§]

Abstract

Every finite and every co-finite set of non-negative integers is decidable. This is true and it is not, depending on whether the set is given constructively. A similar constraint is applicable in language theory and many other fields. The constraint is usually understood and, hence, omitted.

The phenomenon of a set being finite, but possibly undecidable, is, of course, a consequence of allowing non-constructive arguments in proofs. In this note we discuss a few ramifications of this fact. We start out with showing that every number theoretic statement that can be expressed in first-order logic can be reduced to a finite set, to be called a test set. Thus, if one knew the test set, one could determine the truth of the statement. The crucial point is, of course, that we may not able to know what the finite test set is. Using problems in the class Π_1 of the arithmetic hierarchy as an example, we establish that the bound on the size of the test set is Turing-complete and that it is upper-bounded by the busy-beaver function.

This re-enforces the fact that there is a vast difference between finiteness and constructive finiteness. In the context of the present re-opened discussion about the notion of computability – possibly extending its realm through new computational models derived from physics – the constraint of constructivity of the model itself may add another twist.

1 Introduction

In the early days of decidability theory and also of theoretical computer science it was not uncommon to find statements like *every finite and every co-finite set of non-negative integers is decidable* in the research literature and in textbooks, and to find "proofs" of this using the argument that a decision algorithm could use table look-up; moreover, such statements themselves would be used in proofs of the decidability of other problems via reduction to finite or co-finite sets.[1] Of course every finite or co-finite set is decidable, but only – as is well-known – if it is given constructively. Similar constraints are applicable in

*The research reported in this paper was partially supported by Auckland University, Research Grant A18/XXXXX/62090/3414050, and by the Natural Sciences and Engineering Council of Canada, Grant OGP0000243.

[†]Department of Computer Science, The University of Auckland, Private Bag 92019, Auckland, New Zealand; e-mail: cristian@cs.auckland.ac.nz.

[‡]Department of Computer Science, The University of Western Ontario, London, Ontario, Canada N6A 5B7, and Institut für Informatik, Universiät Potsdam, Am Neuen Palais 10, D-14469, Potsdam, Germany; e-mail: helmut@uwo.ca.

[§]Department of Mathematics, The University of Auckland, Private Bag 92019, Auckland, New Zealand.

[1]We refrain from giving references, because pointing to past mistakes is not the aim of this paper. However, the interested reader is likely to find such statements by just perusing a few older books.

language theory and many other fields. The constraint is, of course, usually understood and, hence, omitted. The phenomenon of a set being finite, but possibly undecidable, is, of course, a consequence of allowing non-constructive arguments in proofs. In this note we discuss a few ramifications of this fact. We start out with showing that every number theoretic statement that can be expressed in first-order logic can be reduced to a finite set, to be called a test set. Thus, if one knew the test set, one could determine the truth of the statement. This rather simple result models what is sometimes referred to as experimental mathematics: Simply stated, if the statement is true we don't need to do anything and if it is false we find the smallest counter-example by computer. We then show how several classical problems fall into this category. The crucial point is, of course, that we may not be able to know what the finite test set is. Using problems in the class Π_1 of the arithmetic hierarchy as an example, we establish that the bound on the size of the test set is Turing-complete and that it is upper-bounded by the busy-beaver function.

This re-enforces the fact that there is a vast difference between finiteness and constructive finiteness. In the context of the present re-opened discussion about the notion of computability – possibly extending its realm through new computational models derived from physics – the constraint of constructivity of the model itself may add another twist.

Let $\mathbb{N}$ denote the set of positive integers, let $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$, and, for $k \in \mathbb{N}$, consider a k-ary predicate P on $\mathbb{N}$, that is, a mapping of $\mathbb{N}^k$ into the set $\mathbb{B} = \{0, 1\}$ of truth values. Consider the formula

$$f = Q_1 n_1 \, Q_2 n_2 \, \ldots \, Q_k n_k \, P(n_1, n_2, \ldots, n_k)$$

where $Q_1, Q_2, \ldots, Q_k \in \{\forall, \exists\}$ are quantifier symbols. In analogy to the arithmetic classes, we say that f is in the class $\hat{\Pi}_s$ or $\hat{\Sigma}_s$ if the quantifier prefix of f starts with $\forall$ or $\exists$, respectively, and contains $s - 1$ alternations of quantifier symbols. When P is computable, then f is in Π_s or Σ_s, respectively.[2] It is sufficient to consider only such formulæ f in which no two consecutive quantifier symbols are the same; in the sequel we make this assumption without special mention. With f as above, one has $s = k$.

As usual in logic, we write $P(n_1, \ldots, n_k)$ instead of $P(n_1, \ldots, n_k) = 1$ when $n_1, \ldots, n_k$ are elements of $\mathbb{N}$. Thus, $\neg P(n_1, \ldots, n_k)$ if and only if $P(n_1, \ldots, n_k) = 0$. Moreover, since we consider variable symbols only in the domain $\mathbb{N}$, if f is any formula in first-order logic, we write f *is true* instead of f *is true in* $\mathbb{N}$.

Let Γ_s be one of the classes $\hat{\Pi}_s$, $\hat{\Sigma}_s$, Π_s, and Σ_s. We refer to the task of proving or refuting a first-order logic formula as a *problem* and especially, to problems expressed by formulæ in Γ_s as *Γ_s-problems.*

We say that a problem is being *solved* if the corresponding formula is proved or disproved to be true, that is, if the truth value of the formula is determined. A problem is said to be *finitely solvable* if it can be solved by examining finitely many cases.[3]

For example, consider the predicate

$$P(n) = \begin{cases} 1, & \text{if } n \text{ is even or } n = 1 \text{ or } n \text{ is a prime,} \\ 0, & \text{otherwise,} \end{cases}$$

[2] See [32] for general background on arithmetic classes.

[3] A rigorous definition of this notion is given in Section 3 below.

that is, $P(n) = 0$ if and only if n is an odd number greater than 1 which is not a prime. Then the problem expressed by the formula $\forall n\, P(n)$ is finitely solvable;[4] indeed, it is sufficient to check all n up to and including 9.

In this paper, we mainly consider $\hat{\Pi}_1$-problems and Π_1-problems. For example, *Goldbach's conjecture* is a Π_1-problem. It states that *every even* $n \in \mathbb{N}$ *is the sum of two primes.*[5] To express this in the terminology as introduced, let $P_{\mathrm{G}} : \mathbb{N} \to \mathbb{B}$ be such that

$$P_{\mathrm{G}}(n) = \begin{cases} 1, & \text{if } n \text{ is odd or } n \text{ is the sum of two primes,} \\ 0, & \text{otherwise.} \end{cases}$$

Thus, $f_{\mathrm{G}} = \forall n\, P_{\mathrm{G}}(n)$ is true if and only if Goldbach's conjecture is true.

Similarly, *Riemann's hypothesis* is a Π_1 problem.[6] Consider the complex function

$$\zeta(s) = \frac{1}{1 - 2^{1-s}} \cdot \sum_{n=1}^{\infty} \frac{(-1)^{n-1}}{n^s},$$

where $s = \sigma + \mathrm{i}\, t$, $\sigma, t \in \mathbb{R}$, $\sigma > 0$, and $s \neq 1$. Riemann conjectured that all zeroes $s_0 = \sigma_0 + \mathrm{i}\, t_0$ of ζ satisfy $\sigma_0 = \frac{1}{2}$ and are simple [30].

By a result of [14], Riemann's hypothesis can be expressed in terms of the function $\delta_{\mathrm{R}} : \mathbb{N} \to \mathbb{R}$ defined by

$$\delta_{\mathrm{R}}(k) = \prod_{n<k} \prod_{j \leq n} \eta_{\mathrm{R}}(j),$$

where

$$\eta_{\mathrm{R}}(j) = \begin{cases} p, & \text{if } j = p^r \text{ for some prime } p \text{ and some } r \in \mathbb{N}, \\ 1, & \text{otherwise.} \end{cases}$$

Riemann's hypothesis is equivalent with the assertion that

$$\left(\sum_{k \leq \delta_{\mathrm{R}}(n)} \frac{1}{k} - \frac{n^2}{2} \right)^2 < 36n^3,$$

for all $n \in \mathbb{N}$, see [14].[7] Hence, let

$$P_{\mathrm{R}}(n) = \begin{cases} 1, & \text{if } \left(\sum_{k \leq \delta_{\mathrm{R}}(n)} \frac{1}{k} - \frac{n^2}{2} \right)^2 < 36n^3, \\ 0, & \text{otherwise.} \end{cases}$$

Thus, $f_{\mathrm{R}} = \forall n\, P_{\mathrm{R}}(n)$ is true if and only if the Riemann hypothesis is true. Clearly, P_{R} is decidable. Therefore, Riemann's hypothesis is a Π_1-problem.

As in the case of the Goldbach conjecture, also for the Riemann hypothesis huge computations have been performed to search for a counter-example – or to increase the confidence [3], [4], [5], [6], [25].

---

[4] This example is based on a folklore joke on induction proofs: To prove that *all odd natural numbers greater than* 2 *are primes* one proceeds as follows: 3 is a prime; 5 is a prime; 7 is a prime; 9 is a measuring error; 11 is prime; 13 is a prime; this is enough evidence.

[5] The conjecture was stated in 1742 by Goldbach in a letter to Euler [17]. According to [22], in 1980 the Goldbach conjecture was known to be true for all $n \leq 10^8$; in [35] of December 1994, it is claimed that no counter-example exists up to $2 \cdot 10^{10}$. Hardy states that the Goldbach problem is "probably as difficult as any of the unsolved problems in mathematics" [19]. See also [26] and [34].

[6] The problem is first proposed in [30]; see also [31].

[7] For another proof see [23], pp. 117–122.

Of course, not every mathematical statement is a Π_1-problem. For instance, the conjecture stating the existence of infinitely many twin primes, that is, consecutive odd primes such as 857 and 859, is not a Π_1-problem. With

$$P_{\mathrm{T}}(n,m) = \begin{cases} 1, & m > n \text{ and } m \text{ and } m+2 \text{ are primes,} \\ 0, & \text{otherwise,} \end{cases}$$

this conjecture can be stated as

$$f_{\mathrm{T}} = \forall n \exists m \, P_{\mathrm{T}}(n,m).$$

The formula f_{T} is in the class Π_2. Bennett claims that most mathematical conjectures can be settled indirectly by proving *stronger* Π_1-problems, see [2]. For the twin-prime conjecture such a stronger Π_1-problem is obtained as follows. Consider the predicate

$$P'_{\mathrm{T}}(n) = \begin{cases} 1, & \text{if there is } m \text{ with } 10^{n-1} \leq m \leq 10^n, \, m \text{ and } m+2 \text{ primes,} \\ 0 & \text{otherwise.} \end{cases}$$

Let $f'_{\mathrm{T}} = \forall n P'_{\mathrm{T}}(n)$. Thus, f'_{T} gives rise to a Π_1-problem and, if f'_{T} is true, then also f_{T} is true.

In this paper we discuss the fact – surprising (only) at first thought – that every $\hat{\Pi}_s$-problem and every $\hat{\Sigma}_s$-problem has a finite test set. Of course, there cannot be a constructive proof of this statement. Moreover, already for $s = 1$ the size of the test sets behaves as badly as the busy beaver.

2 Notation and Basic Notions

In this section we briefly review some basic notions and introduce some notation. Let X be a non-trivial alphabet, that is, a non-empty, finite set with at least 2 elements. Then X^* is the set of all *words* over X. A *(formal) language over* X is a subset of X^*.

We assume that the reader is familiar with the theory of computable functions on integers or strings (see [32, 8]). If U is a universal Turing machine which maps strings over X to non-negative integers, and π is a program for U then $U(\pi)$ denotes the result of applying U to π and an empty input tape. In particular, we write $U(\pi) = \infty$ when U does not halt on π.

3 Finite Solvability

For $s \in \mathbb{N}$, let $\hat{\Gamma}_s$ denote any of $\hat{\Pi}_s$ and $\hat{\Sigma}_s$, and let Γ_s denote any of Π_s and Σ_s.

Definition 3.1 *Let*

$$f = Q_1 n_1 \, Q_2 n_2 \, \ldots Q_s n_s \, P(n_1, n_2, \ldots, n_s),$$

with $s \in \mathbb{N}$, where $Q_1, Q_2, \ldots, Q_s$ are alternating quantifier symbols.

1. *A* test set *for f is set $T \subseteq \mathbb{N}^s$ such that f is true in $\mathbb{N}^s$ if and only if it is true in T.*

2. *The* problem *of f is* finitely solvable *if there is a finite test set for f.*

Theorem 3.1 *Let $s \in \mathbb{N}$. Every $f \in \hat{\Gamma}_s$ is finitely solvable.*

Proof. Let

$$f = Q_1 n_1 \, Q_2 n_2 \, \ldots Q_s n_s \, P(n_1, n_2, \ldots, n_s),$$

with $s \in \mathbb{N}$, where $Q_1, Q_2, \ldots, Q_s$ are alternating quantifier symbols. We determine a sequence $N_1, N_2, \ldots, N_s$ of finite sets with $N_i \subseteq \mathbb{N}^i$ such that the problem posed by f can be solved by checking all s-tuples $(n_1, n_2, \ldots, n_s) \in N_s$.

We define the sets N_i by induction on i. For this purpose, let

$$f_i(m_1, \ldots, m_{i-1}) = Q_i n_i \, \ldots Q_s n_s \, P(m_1, \ldots, m_{i-1}, n_i, \ldots, n_s),$$

where $m_1, \ldots, m_{i-1} \in \mathbb{N}$. In particular, $f_1() = f$ and $f_{s+1}(m_1, \ldots, m_s) = P(m_1, \ldots, m_s)$.

For $i = 1$, if $Q_1 = \forall$, let

$$\nu_1 = 1 \quad \text{if } f = f_1(\,) \text{ is true,}$$

and

$$\nu_1 = \min\{m_1 \mid m_1 \in \mathbb{N}, \neg f_2(m_1)\} \quad \text{otherwise;}$$

if $Q_1 = \exists$, let

$$\nu_1 = 1 \quad \text{if } f = f_1(\,) \text{ is not true,}$$

and

$$\nu_1 = \min\{m_1 \mid m_1 \in \mathbb{N}, f_2(m_1)\} \quad \text{otherwise.}$$

Let $N_1 = \{(m_1) \mid m_1 \in \mathbb{N}, m_1 \leq \nu_1\}$.

Now, suppose N_{i-1} has been defined and $i \leq s$. For each $(m_1, \ldots, m_{i-1}) \in N_{i-1}$, define $\nu_i(m_1, \ldots, m_{i-1}) \in \mathbb{N}_0$ as follows. If $Q_i = \forall$, let

$$\nu_i(m_1, \ldots, m_{i-1}) = 1 \quad \text{if } f_i(m_1, \ldots, m_{i-1}) \text{ is true,}$$

and

$$\nu_i(m_1, \ldots, m_{i-1}) = \min\{m_i \mid m_i \in \mathbb{N}, \neg f_{i+1}(m_1, \ldots, m_{i-1}, m_i)\} \quad \text{otherwise;}$$

if $Q_i = \exists$, let

$$\nu_i(m_1, \ldots, m_{i-1}) = 1 \quad \text{if } f_i(m_1, \ldots, m_{i-1}) \text{ is not true,}$$

and

$$\nu_i(m_1, \ldots, m_{i-1}) = \min\{m_i \mid m_i \in \mathbb{N}, f_{i+1}(m_1, \ldots, m_{i-1}, m_i)\} \quad \text{otherwise.}$$

Let

$$N_i = \{(m_1, \ldots, m_i) \mid (m_1, \ldots, m_{i-1}) \in N_{i-1}, m_i \in \mathbb{N}, m_i \leq \nu(m_1, \ldots, m_{i-1})\}.$$

We now prove,[8] by induction on i, that each set $T_i = N_i \times \mathbb{N}^{s-i}$ is a test set for f. Then, in particular, N_s is a finite test set for f.

[8] We decided to include this rather straight-forward proof as it was only by this proof that we discovered some subtle traps in the construction of the test sets.

Consider $i = 1$. Suppose first that $Q_1 = \forall$. The set $N_1$is $\{(1)\}$ and, clearly, the set T_1 is a test set[9] for f. When f is false the set N_1 consist of all positive integers up to the first counter-example for the first variable of P. Hence, again, T_1 is a test set for f. On the other hand, suppose that $Q_1 = \exists$. Then $N_1 = \{(1)\}$ when f is false. Clearly T_1 is a test set[10] for f. When f is true the set N_1 consists of all positive integers up to the first witness for the first variable of P. Again T_1 is a test set for f.

Now consider $i > 1$ and assume that T_{i-1} is a test set for f. First suppose that $Q_i = \forall$. Consider $(m_1, \ldots, m_{i-1}) \in N_{i-1}$. If $f_i(m_1, \ldots, m_{i-1})$ is true then $\nu_i(m_1, \ldots, m_{i-1}) = 1$. As T_{i-1} is a test set for f, to test whether f is true on $\{(m_1, \ldots, m_{i-1})\} \times \mathbb{N}^{s-i+1}$ it suffices to test on $\{(m_1, \ldots, m_{i_1}, 1)\} \times N^{s-i}$, and $(m_1, \ldots, m_{i-1}, 1) \in N_i$. If $f_i(m_1, \ldots, m_{i-1})$ is false then N_i contains all the i-tuples $(m_1, \ldots, m_{i-1}, m_i)$ with m_i ranging from 1 to the smallest counter-example. Hence, as T_{i-1} is a test set for f so is T_i.

Now suppose that $Q_i = \exists$. If $f_i(m_1, \ldots, m_{i-1})$ is false then $\nu_i(m_1, \ldots, m_{i-1}) = 1$. As T_{i-1} is a test set for f, to test whether f is true on $\{(m_1, \ldots, m_{i-1})\} \times \mathbb{N}^{s-i+1}$ it suffices to test on $\{(m_1, \ldots, m_{i_1}, 1)\} \times N^{s-i}$, and $(m_1, \ldots, m_{i-1}, 1) \in N_i$. If $f_i(m_1, \ldots, m_{i-1})$ is true then N_i contains all the i-tuples $(m_1, \ldots, m_{i-1}, m_i)$ with m_i ranging from 1 to the smallest witness. Hence, as T_{i-1} is a test set for f so is T_i. □

The proof of Theorem 3.1 is non-constructive and this remains so even when P is decidable. Thus, from this proof we do not learn anything about the number of cases one needs to check in order to prove or disprove the truth of f. It is clear from the theories of arithmetic classes and degrees of unsolvability that, in general, finite test sets cannot be *constructed* for this type of problems even when the predicate is computable. We try to shed some light, from a different perspective, on some of the reasons why this cannot be done.

The proof of Theorem 3.1 highlights a typical pitfall in proofs in computability theory when the reasoning of classical logic is used. The proof and the statement proved are computationally meaningless as neither helps with *actually solving the* $\hat{\Gamma}_s$*-problem.* The "construction" of the sets N_i in the proof disguises the fact that none of these finite sets may be computable. See, for example, the formula f_G expressing Goldbach's conjecture.

The statement of Theorem 3.1 has some similarity with the *Test Set Theorem* in formal language theory. This theorem, originally known as Ehrenfeucht's conjecture, can be stated as follows: *Let X and Y be alphabets, and let $L \subseteq X^*$. There exists a finite subset F of L, a* test set, *such that, for any two morphisms f, g from X^* to Y^*, $f(u) = g(u)$ for all $u \in L$ whenever $f(u) = g(u)$ for all $u \in F$.* This was proved independently in [1] and [18].[11] In [7] and also [11] it is pointed out that the existence of the test sets is not constructive.[12] In the statement of the Test Set Theorem for languages, the order of the quantifiers, that is, $\forall L \exists F \forall f \forall g$, is very important. The modified order $\forall L \forall f \forall g \exists F$ results in a far simpler statement, for which a proof can be given using the same ideas as in the proof of Theorem 3.1.

[9] In fact, the empty set would be a test set for f. However, if one uses this idea, that is sets ν_1 to 0 rather than 1 – and similarly for ν_i in general – then the 'construction' seems to break down.

[10] Again the empty set could have been used were it not for problems with the subsequent steps of the 'construction'.

[11] Explanations of the proofs are given in [27] and [33]. For further information see [13].

[12] Under special assumptions on L like regularity, test sets can be effectively constructed [20], [21]; see also [13].

In the sequel, for $f \in \hat{\Gamma}_s$, let $N(f) = N_s$ with N_s as in the proof of Theorem 3.1. In particular, when $s = 1$, then $N(f)$ is the set $\{(n_1) \mid n_1 \in \mathbb{N}, n_1 \leq \nu_1\}$. For this case, we define $\nu(f) = \nu_1$.

4 Π_1-Problems

In this section, we analyze the case of Π_1-problems in greater detail. Let X be an arbitrary but fixed alphabet. We use X as the alphabet for programs of universal Turing machines. We also fix a computable bijective function $\langle , \rangle : X^* \times \mathbb{N}_0 \to X^*$. Consider $f = \forall n\, P(n)$ where P is a computable predicate on $\mathbb{N}$. We assume that P is given as a program for an arbitrary, but fixed universal Turing machine U. Thus P is given as a word $\pi_P \in X^*$ such that $U(\langle \pi_P, n \rangle) = P(n)$ for all $n \in \mathbb{N}$. One can, therefore, consider ν as a partial function of X^* into $\mathbb{N}_0$, that is, $\nu(\pi_P) = \nu(f)$ with f as above. We first determine an upper bound on $\nu(f)$ for $f \in \Pi_1$.

The *busy beaver function* $\sigma : \mathbb{N} \to \mathbb{N}$ ([29]; see also [15], [16], Chapter 39) is defined as follows:

$$\sigma(n) = \max\{U(x) \mid x \text{ is a program of length } n \text{ for } U \text{ and } U(x) \text{ halts on } x\}.$$

Let P be a computable unary predicate on $\mathbb{N}$, let $f = \forall n\, P(n)$, hence $f \in \Pi_1$. Consider a program p_f for U such that

$$U(p_f) = \min\{n \mid \neg P(n)\},$$

if f is not true, and such that U runs forever on p_f if f is true. Such a program always exists because the program, which tries $P(1)$, $P(2)$, ... and halts with the first n such that $\neg P(n)$, has the required properties. Let $m_f = |p_f|$. If f is not true then U halts on p_f with output $\nu(f)$. Hence $\nu(f) \leq \sigma(m_f)$. If f is true then $\nu(f) = 0$. This proves the following statement.

Proposition 4.1 *For every $f \in \Pi_1$, $\nu(f) \leq \sigma(m_f)$.*

By Theorem 4.1, to solve the problem of f one only needs to check the truth value of $P(n)$ for all n not exceeding $\sigma(m_f)$. This could be very useful if σ were computable. However, σ grows faster than any computable function. Hence, the bound $\nu(f) \leq \sigma(m_f)$ does not help in the actual solution of the problem of f. In fact, no computable bound exists! Here is the argument. For any $\pi \in X^*$, define the predicate P_π on $\mathbb{N}$ by

$$P_\pi(n) = \begin{cases} 1, & U(\pi) \text{ does not halt within } n \text{ steps,} \\ 0, & \text{otherwise.} \end{cases}$$

Clearly, the predicate is computable. Let $f_\pi = \forall n\, P_\pi(n)$. Then f_π is true if and only if $U(\pi)$ does not halt.

Assume now that there is a program to compute an upper bound of $\nu(f)$ for any $f \in \Pi_1$; this program takes, as input, a program ρ computing the predicate P^ρ and computes as output an integer $\nu'(\rho)$ such that $\nu(f^\rho) \leq \nu'(\rho)$, where $f^\rho = \forall n\, P^\rho(n)$. We show that this assumption implies the existence of an algorithm deciding the halting problem for Turing machines. Indeed, consider $\pi \in X^*$. To decide whether $U(\pi)$ halts, first compute a program p_π computing P_π. Next compute $\nu'(p_\pi)$. As $f_\pi = f^{p_\pi}$, one has $\nu(f_\pi) \leq \nu'(p_\pi)$. Hence, to determine whether f_π is true, it is sufficient to determine whether $P_\pi(n)$ for all $n \leq \nu'(p_\pi)$. If so, then $U(\pi)$ halts; otherwise it doesn't.

Theorem 4.1 *The upper bound ν is Turing-complete.*

Proof. We already showed that an oracle for ν or an upper bound on ν allows one to decide the halting problem. The conversely follows from Proposition 4.1. □

Corollary 4.1 *There is no constructive proof showing that every $f \in \Pi_1$ has a finite test set.*

With appropriate modifications, a statement similar to Corollary 4.1 can be proved for Σ_1. In fact, for any $s \in \mathbb{N}$ and any Γ_s, there is no constructive proof of the fact that every $f \in \Gamma_s$ has a finite test set.

5 Conclusions

Many true Π_1-problems are undecidable, hence independent with respect to a given sufficiently rich, sound, and computably axiomatizable theory. The analysis above can help us in understanding this phenomenon. Knowing that P is false can be used to get a proof that "P is false": we keep computing $P(n)$, for large enough n until we get an n such that $\neg P(n)$. But this situation is not symmetric: if we know that P is true we might not be able to prove that "P is true", and this case is quite frequent [10]. Indeed, even when we "have" the proof, that is, we have successfully checked that $P(n) \neq 0$, for all $n \leq \nu((\forall n)P(n))$, we might not be able to "realise" that we have achieved the necessary bound.

The correspondence $P \mapsto \nu((\forall n)P(n))$ exists and is perfectly legitimate from a classical point of view, but has no constructive "meaning". To a large extent the mathematical activity can be regarded as a gigantic, collective effort to compute individual instances of the function $\nu((\forall n)P(n))$. This point of view is consistent with Post's description of mathematical creativity [28]: "Every symbolic logic is incomplete and extendible relative to the class of propositions constituting K_0. The conclusion is inescapable that even for such a fixed, well defined body of mathematical propositions, *mathematical thinking is, and must remain, essentially creative.*"[13] It also gives support to the "quasi-empirical" view of mathematics, which sustains that although mathematics and physics are different, it is more a matter of degree than black and white [12, 9]; see also [24].

In essence, the seemingly paradoxical situation arises from the fact that, in classical logic, it may happen that only finite resources are needed for defining a finite object while finite resources will not suffice to determine the same object constructively. The finite "character" of a problem may nevertheless rule out – in a very fundamental way – that its solution can be obtained by finite means.

Acknowledgment

We thank Douglas Bridges, Greg Chaitin and Solomon Marcus for stimulating discussions.

[13] As usual, K_0 means the halting problem in this quote.

References

[1] M. H. Albert, J. Lawrence: A proof of Ehrenfeucht's conjecture. *Theoret. Comput. Sci.* **41** (1985), 121–123.

[2] C. H. Bennett: Chaitin's Omega. In M. Gardner (editor): *Fractal Music, Hypercards, and More ...* 307–319. W. H. Freeman, New York, 1992.

[3] R. P. Brent, J. v. d. Lune, H. J. J. t. Riele, D. T. Winter: On the zeros of the Riemann zeta function in the critical strip I. *Math. Comp.* **33** (1979), 1361–1372.

[4] R. P. Brent, J. v. d. Lune, H. J. J. t. Riele, D. T. Winter: On the zeros of the Riemann zeta function in the critical strip II. *Math. Comp.* **39** (1982), 681–688.

[5] R. P. Brent, J. v. d. Lune, H. J. J. t. Riele, D. T. Winter: On the zeros of the Riemann zeta function in the critical strip III. *Math. Comp.* **41** (1983), 759–767.

[6] R. P. Brent, J. v. d. Lune, H. J. J. t. Riele, D. T. Winter: On the zeros of the Riemann zeta function in the critical strip IV. *Math. Comp.* **46** (1986), 667–681.

[7] C. Calude: Note on Ehrenfeucht's conjecture and Hilbert's basis theorem. *Bull. EATCS* **29** (1986), 18–22.

[8] C. Calude: *Theories of Computational Complexities*. North-Holland, Amsterdam, 1988.

[9] C. S. Calude, G. J. Chaitin: Randomness everywhere. *Nature*, 400, 22 July (1999), 310–311.

[10] C. Calude, H. Jürgensen, M. Zimand: Is independence an exception? *Applied Mathematics and Computation* **66** (1994), 63–76.

[11] C. Calude, D. Vaida: Ehrenfeucht test set theorem and Hilbert basis theorem: A constructive glimpse. In A. Kreczmar, G. Mirkowska (editors): *Mathematical Foundations of Computer Science; Porabka-Kozubnik, Poland; August 28–September 1, 1989. Lecture Notes in Computer Science* **379**, 177–184, Springer-Verlag, Berlin, 1989.

[12] G. J. Chaitin: *The Unknowable.* Springer-Verlag, Singapore, 1999.

[13] C. Choffrut, J. Karhumäki: Combinatorics on words. In G. Rozenberg, A. Salomaa (editors): *Handbook of Formal Language Theory*, Vol. 1, 329–438, Springer-Verlag, Berlin.

[14] M. Davis, Y. V. Matijasevič, J. Robinson: Hilbert's tenth problem. Diophantine equations: Positive aspects of a negative solution. In F. E. Browder (editor): *Mathematical Developments Arising from Hilbert Problems*. 323–378. American Mathematical Society, Providence, RI, 1976.

[15] A. K. Dewdney: A computer trap for the busy beaver, the hardest-working Turing machine. *Scientific American* **251**(8) (1984), 19–23.

[16] A. K. Dewdney: *The New Turing Omnibus*. Computer Science Press, New York, 1993.

[17] L. E. Dickson: *History of the Theory of Numbers*. Carnegie Institute, Washington, 1919, 1920, 1923. 3 volumes.

[18] V. S. Guba: The equivalence of infinite systems of equations in free groups and semigroups. *Mat. Zametki* **40** (1986), 321–324, in Russian.

[19] G. H. Hardy: Goldbach's theorem. *Mat. Tid. B* **1** (1922), 1–16. Reprinted in *Collected Papers of G. H. Hardy*, vol. 1, Oxford University Press, Oxford, 1966, 545–560.

[20] J. Karhumäki, W. Rytter, S. Jarominek: Efficient constructions of test sets for regular and context-free languages. *Theoret. Comput. Sci.* **116** (1993), 305–316.

[21] J. Karhumäki, W. Plandowski, W. Rytter: Polynomial size test sets for context-free languages. *J. Comput. System Sci.* **50** (1995), 11–19.

[22] W. A. Light, T. J. Forres, N. Hammond, S. Roe: A note on the Goldbach conjecture. *BIT* **20** (1980), 525.

[23] Y. V. Matijasevič: *Hilbert's Tenth Problem*. MIT Press, Cambridge, MA, 1993, 117–122.

[24] S. Marcus: Bridging linguistics and computer science, via mathematics. In C. S. Calude (editor): *People and Ideas in Theoretical Computer Science*. 163–176. Springer-Verlag, Singapore, 1998.

[25] A. M. Odlyzko: Tables of zeros of the Riemann zeta function at `http://www.research.att.com/~amo/zeta_tables/index.html`.

[26] C.-T. Pan: *Goldbach Conjecture*. Science Press, Beijing, 1992.

[27] D. Perrin: On the solution of Ehrenfeucht's conjecture. *Bull. EATCS* **27** (1985), 68–70.

[28] E. L. Post: Recursively enumerable sets of positive integers and their decision problems. *Bull. (New Series) Amer. Math. Soc.* **50** (1944), 284–316.

[29] T. Rado: On non-computable numbers. *Bell System Tech. J.* **3** (1962), 977–884.

[30] B. Riemann: Über die Anzahl der Primzahlen unter einer gegebenen Größe. In *Gesammelte mathematische Werke und, wissenchaftlicher Nachlaß*. 177–185. Springer-Verlag, Berlin, 1990.

[31] H. Riesel: *Prime Numbers and Computer Methods for Factorization*. Birkhäuser, Boston, second ed., 1994.

[32] H. Rogers: *Theory of Recursive Functions and Effective Computability*. McGraw-Hill, New York, 1967.

[33] A. Salomaa: The Ehrenfeucht conjecture: A proof for language theorists. *Bull. EATCS* **27** (1985), 71–82.

[34] W. Yuan (editor): *Goldbach Conjecture*. Singapore, 1984. World Scientific.

[35] Names of large numbers & unsolved problems. http://www.smartpages.com/faqs/sci-math-faq/unsolvedproblems/faq.html, December 1994.