IN MEMORY OF MARTIN DAVIS

WESLEY CALVERT, VALENTINA HARIZANOV,
EUGENIO G. OMODEO, ALBERTO POLICRITI,
AND ALEXANDRA SHLAPENTOKH

In 1950, Martin David Davis found the culture in his graduate program at Princeton deeply alienating, and he was ready to be out of there. So he solved an open problem of Kleene (establishing the backbone for a major branch of modern computability theory), wrote down his initial steps toward the eventually successful solution of a Hilbert problem, and graduated.

FIGURE 1. Martin in the late 1940s. (Courtesy of his son Harold Davis)

Davis's solution to the problem of Kleene became the hyperarithmetical hierarchy, which we will explain in Section 1. His work on Hilbert's Tenth Problem included what would later be called his "daring hypothesis" [33]: a conjecture, later verified, that the computably enumerable sets were exactly the Diophantine sets, as we will descuss in Section 2.

In spite of the economic restrictions which his family, as immigrants who arrived shortly before the Great Depression, had to face, he received a high-quality education. He arrived at City College of New York as a freshman in 1944 and soon became interested in the foundations of real analysis and in logic. He approached Post, who introduced him to the writings by Church and Kleene on algorithmic unsolvability and to Hilbert's Tenth Problem, which would soon become Davis's "lifelong obsession". When Davis had to choose where to undertake graduate studies, Post advised him to go to Princeton, where, as Davis later expressed it, the "culture clash" between his Jewish working class background and the "genteel Princeton atmosphere" made him eager to conclude quickly: in fact, he got his Ph.D. in just 2 years, under the guidance of Church, in 1950.
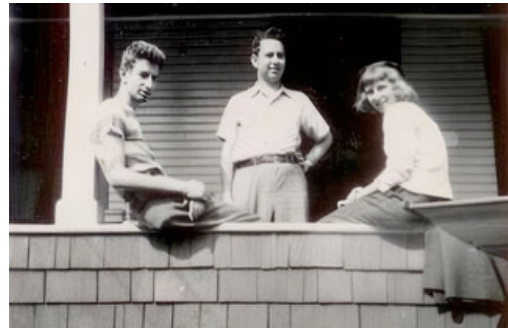

FIGURE 2. Martin, middle, with his friends Jacob T. and Judith Schwartz. (Courtesy of Diana Robinson Schwartz)

Davis's first position was at the University of Illinois at Urbana-Champaign, but "the Korean war and the hot breath of the draft" led him to leave that job for the Control Systems Laboratory. He later moved to the Institute for Advanced Study, the University of California at Davis, the Ohio State University, the Rensselaer Polytechnic Institute, Yeshiva University, and New York University. Certain summer projects funded by military and civilian research agencies enabled him to make crucial achievements ("It was in the summer of 1959 that Hilary and I really hit the jackpot," he says, to describe the original, raw discovery of what would become known as the celebrated Davis–Putnam–Robinson theorem). Over the course of his career, Davis supervised a total of 25 Ph.D. students, including scholars now known for work in mathematics, computer science, and philosophy.

Figure 3. Martin ready to give an answer, late 1990s. (Courtesy of Domenico Cantone)

Davis's expository books have become classics and have been translated into various languages: *Computability and Unsolvability*; *A First Course in Functional Analysis*, *Applied Nonstandard Analysis*; *Computability, Complexity, and Languages: Fundamentals of Theoretical Computer Science* (with Ron Sigal and Elaine J. Weyuker); and *The Universal Computer: The Road from Leibniz to Turing*. His 1993 *Lecture Notes in Logic* are a true jewel.

His many honors and awards include the Steele Prize, the Chauvenet Prize (with Reuben Hersch), Fellowship of the AAAS, a Guggenheim Foundation Fellowship, the Herbrand Award of the International Conference on Automated Deduction; and the Pioneering Achievement Award from the ACM SIG on Design Automation.

Martin Davis left two rather comprehensive autobiographic accounts [15, 18] and the long interviews [28, 29]. For this reason, the present note will primarily focus on his scientific achievements.

1. Computability

Computably Enumerable Sets and Universal Turing Machines. Modern computability theory started in 1936 with Turing's seminal paper on computable numbers with an application to the Entscheidungsproblem, a decision problem of Hilbert and Ackermann for which Turing provided a negative solution. Turing introduced what today we would call a Turing machine (he called it an a-machine), which is essentially an abstraction of a computer. Turing's and other formalisms for an intuitive concept of an effectively calculable function, developed by Gödel, Kleene, Church, Post and others, had profound significance for the emerging science of computing.

One of the main concepts in computability theory (also called recursion theory) is that of a computable function and computable relation. A function $f : \mathbb{N}^n \to \mathbb{N}$ is *computable* if there is a Turing machine that on every input $a_1, \ldots, a_n$ halts and outputs its value $f(a_1, \ldots, a_n)$. Addition and multiplication on natural numbers are computable functions (operations). A set of natural numbers is *computable* if its characteristic function is computable. For example, the set of prime numbers is computable. All finite sets are computable. Clearly, the complement of a computable set is computable. Computable n-ary relations are defined similarly. It can be shown that there is a computable bijection $h : \mathbb{N}^2 \to \mathbb{N}$, thus allowing algorithmic coding of pairs and, more generally, finite tuples of natural numbers by natural numbers. Decidable problems are encoded by computable relations.

A function $f : D \to \mathbb{N}$, where $D \subseteq \mathbb{N}^n$, is *partial computable* if there is a Turing machine that on every input in the domain D of f halts and outputs its value, while on every input in $\mathbb{N}^n$ that is not in the domain of f it does not halt, thus computing forever. Clearly, computable functions are partial computable functions that are total. Partial computable functions coincide with partial recursive functions defined by Kleene, starting with some basic functions and applying the operations of composition, primitive recursion, and unbounded search.

Since each Turing machine is a finite list of instructions, Turing machines can be algorithmically enumerated without repetitions as:

$$M_0, M_1, M_2, \ldots .$$

A Turing machine on a given input may halt and output its value, or it may compute forever. For each Turing machine M_e, we denote the n-ary partial function it computes by $\varphi_e^{(n)}$ and its domain by $W_e^{(n)}$. Hence

$$\varphi_0^{(n)}, \varphi_1^{(n)}, \varphi_2^{(n)}, \ldots$$

is a computable enumeration of all n-ary partial computable functions. For $n = 1$, we omit the superscript. Moreover, there is a binary partial computable function ψ such that $\psi(e, x) = \varphi_e(x)$.

The above enumeration gives rise to a universal Turing machine, which can simulate any Turing machine on any input and leads to the idea of stored-program computer. Davis has written on the universal Turing machine in a number of papers starting in 1956. His lecture in 2012 titled "Universality is ubiquitous" is available at https://www.youtube.com/watch?v=ZVTgtODXONc

In a written version published in [16], Davis wrote: "Turing's concept of 'universal machine' will be discussed as an abstraction, as embodied in physical devices, as present in nature, and in connection with the artificial intelligence project".

In addition to his large body of expository work on universal Turing machines, Davis was also an early technical contributor to the subject. Turing had constructed a universal machine, but had not dealt with universal Turing machines as a class of objects. John McCarthy and Claude Shannon posed the problem of giving a definition of universal Turing machines, which would deal, for instance, with the simplicity of the encoding by which the universal machine simulates arbitrary machines. Davis solved this problem in [5], as we explain next.

Definition 1. Let S be a set. We say that S is *computably enumerable* (also called *recursively enumerable*) if and only if S is empty or the range of a computable function.

It is not hard to see that a set is computably enumerable if and only if it is the domain W_e of some partial computable (equivalently, partial recursive) function φ_e. If W_e is nonempty, then W_e can be computably enumerated by the procedure that simultaneously runs

$$M_e(0), M_e(1), \ldots, M_e(k), \ldots$$

and enumerates those k for which $M_e(k)$ halts, as soon as the halting occurs. Here, simultaneously means that at each step we add a new input and also run all activated inputs for an additional computational step. The converse, that every computably enumerable set is some W_e, is also true.

Since we can computably enumerate the Turing machines, we can also computably enumerate the computably enumerable sets by

$$W_0, W_1, W_2, \ldots.$$

Clearly, every computable set is computably enumerable since a decision algorithm can be transformed into an enumeration algorithm. Computable sets are exactly computably enumerable sets that also have computably enumerable complements.

Definition 2. [5]

(1) We say that a set S is *c.e. complete* if and only if it is computably enumerable and for any computably enumerable set W, there is a computable function $\sigma : \mathbb{N} \to \mathbb{N}$ with $x \in W$ if and only if $\sigma(x) \in S$.

(2) We define δ_M to be the set of all initial configurations of M from which M will eventually halt.

(3) We say that M is *universal* if and only if δ_M is a computably enumerable set that is complete.

Davis proved that a machine which is universal in this sense does, indeed, simulate all Turing machines. However, having universal machines defined as a class, rather than simply observed as a phenomenon, opened the door to lines of thinking that involve quantification over all universal Turing machines, such as the foundational work of Solomonoff [39] and Kolmogorov [30] on information theory (Kolmogorov complexity), and the theory of algorithmic randomness arising, in part, from it [27].

The diagonal *halting set H* consists of all inputs e on which the Turing machine with index e halts. That is,

$$H = \{e\colon M_e \text{ halts on input } e\}.$$

It can be shown that the set H is computably enumerable. It is not computable since its complement is not computably enumerable. If the complement $\overline{H}$ were computably enumerable, then for some e_0, we would have $\overline{H} = W_{e_0}$. Then

$$e_0 \in H \Leftrightarrow e_0 \in W_{e_0} \Leftrightarrow e_0 \in \overline{H},$$

which is a contradiction.

It can also be shown that a set A is computably enumerable if and only there is a computable binary relation R such that for every a,

$$a \in A \Leftrightarrow (\exists x) R(a, x).$$

We can relativize all of these notions using Turing's notion of an *oracle* machine. A machine with an oracle for a set S is a Turing machine which carries out its computation with the additional resource of read-only access to the characteristic function of the set S. In this way, even if a set U is not computable, it may be computable with an oracle for another set — for instance, its complement. The halting set relative to S (also called the *jump* of S and denoted S') is defined exactly as before, but replacing the machines with machines with oracle S.

The Hyperarithmetical Hierarchy. In his 1950 Ph.D. thesis at Princeton, Martin addressed a problem posed by Kleene. It was already known that every formula of classical predicate logic is equivalent to a formula consisting of a block of quantifiers ("for all" and "there exists"), followed by a quantifier-free formula. Kleene noted that the optimal form of such

an equivalent formula corresponded to the degree of unsolvability of satisfying that formula. For instance, Turing's halting problem is equivalent to the problem of satisfying a particular sentence with a single existential quantifier, but not to the satisfaction of any quantifier-free formula.

This gives rise to a hierarchy of formulas — and, equivalently, of decision problems, according to the number of quantifiers, and whether those quantifiers are "for all" or "there exists." Since there is no computational difference between determining the existence of a single element and determining the existence of a finite tuple of elements, we consider only alternations of quantifiers.

Definition 3 (The Arithmetical Hierarchy)**.** Let $S \subseteq \mathbb{N}^m$.

(1) We say that S is Σ_1^0 if and only if there is a computable set $T \subseteq \mathbb{N}^{m+1}$ such that $\bar{a} \in S$ if and only if $\exists x\,[(x, \bar{a}) \in T]$.
(2) We say that S is Π_1^0 if and only if there is a computable set $T \subseteq \mathbb{N}^{m+1}$ such that $\bar{a} \in S$ if and only if $\forall x\,[(x, \bar{a}) \in T]$.
(3) We say that S is Σ_{n+1}^0 if and only if there is a Π_n^0 set $T \subseteq \mathbb{N}^{m+1}$ such that $\bar{a} \in S$ if and only if $\exists x\,[(x, \bar{a}) \in T]$.
(4) We say that S is Π_{n+1}^0 if and only if there is a Σ_n^0 set $T \subseteq \mathbb{N}^{m+1}$ such that $\bar{a} \in S$ if and only if $\forall x\,[(x, \bar{a}) \in T]$.

The Σ_1^0 sets are exactly the computably enumerable sets. The Σ_{n+1}^0 sets are exactly those computably enumerable relative to the n-times iterated jump of the empty set. There are certainly sets of natural numbers that are not Σ_n^0 or Π_n^0 for any n, and Kleene asked whether the hierarchy could be continued to transfinite levels.

Davis [3] carried out this generalization by iterating the jump starting at an arbitrary set, and also describing a uniform join of infinitely many jumps. In the following definition, ω denotes the least transfinite ordinal, the order type of the natural numbers.

Definition 4. Let $K_0 = \emptyset$.

(1) Let $K_{\alpha+1} = K_\alpha'$.
(2) Let $K_{\omega n}$ be the set defined by $2^{x_1} 3^{x_2} \in K_{\omega n}$ if and only if $x_1 \in K_{\omega(n-1)+x_2}$.

Davis proved that this hierarchy is proper and that it extends the finite-level Kleene hierarchy to all ordinals less than ω^2. After Davis's work, the major pieces still missing were extension to larger ordinals and the fact that the choice of representative sets at limit levels is not unique. It was five years later that

Spector showed that, up to Turing degree, the definition is robust [40]. This transfinite hierarchy is now at the core of modern computable mathematics [1].

Hypercomputation, Neural Networks, and Unconventional Computation. A major effort of the later part of Davis's career was devoted to defining the bounds of computation. In response to a community of scholars who applied relativistic and quantum theories to propose computing devices more powerful than a Turing machine (an idea they called "hypercomputation"), Davis responded with unbridled skepticism.

Davis's key paper [13] on this subject considered a key proposal for a hypercomputer, a certain kind of neural network. The proposal described a certain model with parameters. If those parameters range over rational numbers, the machine could determine membership in the computable sets, as expected; however, if the parameters were allowed to range over arbitrary reals, it could determine membership in arbitrary subsets of $\mathbb{N}$. Davis pointed out that the ability of the model to determine membership in arbitary sets followed immediately from Turing's observation that any countable set was computable relative to some oracle, still via a Turing machine. In this way, Davis showed that neural networks reflect computation within the bounds of the Church–Turing Thesis — that is, equivalent to a computation that can be done by a Turing machine.

Many of the hypercomputation models rely on some access to certain full-precision real numbers, to which, Davis pointed out, no scientific observation could give us access.

In a paper entitled, "Why there is no such discipline as hypercomputation" (published as an introduction to a special issue of *Applied Mathematics and Computation* devoted to papers on exactly this discipline), Davis argued that if there were a hypercomputer, we would be unable to verify its performance, since we could only see finitely many outputs. Moreover, any real computer is subject not merely to the limitation of a Turing machine — that is, the limitation that only a finite time and a finite span of memory can be used — but to a much more strict limitation of a constant bound on these quantities, depending only on the machine, and not on the algorithm or the data. While investigation of algorithms in the Turing machine context has important meaning, both in theory and in practice, they must finally be executed on finite state automata, a much more limiting device.

4

On a personal note VH. The first time I met Martin Davis at a conference, I was very impressed by his kindness, modesty, sense of humor, and friendliness. Later, I invited him and Virginia to visit me at George Washington University and give a math colloquium talk. It was in November 2007 and his lecture was titled "Unsolvability and undecidability in the Diophantine realm," an ordinary title compared to his 2020 MSRI talk "Here there be monsters." During the GW talk to a packed room, Davis covered many years of work, progress, and struggles on Hilbert's Tenth Problem. When a person in the audience asked him how he managed to persist for so many years working on one problem, he replied that his obsession with this problem "was a disease."

Because of the advances of computer technology, Martin Davis's 2020 talk, "Here there be monsters," was possible when MSRI semester program on Decidability, Definability and Computability in Number Theory had to be moved online. His talk was an opening one for the program and is posted on https://www.msri.org/seminars/25120.

FIGURE 4. Martin and his wife Virginia as grandparents, holding Katie Rose. (Courtesy of Harold Davis)

2. HILBERT'S TENTH PROBLEM

A brief history. Martin Davis was one of the four people who collectively solved Hilbert's Tenth Problem. The other three were Hilary Putnam, Julia Robinson and Yuri Matiyasevich.

The history of Hilbert's Tenth Problem starts in 1900 when, during an International Congress of Mathematicians in Paris, David Hilbert presented a list of 23 problems that had a great influence on mathematics in the twentieth century and continue to influence the subject in the twenty-first. The tenth problem on the list asked to devise a process that determines whether any given Diophantine (polynomial) equation with integer coefficients has a solution in the integers.

If we are to rephrase Hilbert's question in modern terms, we could say that he asked for an algorithm (or a computer program) taking as its input coefficients of a polynomial equation in several variables over $\mathbb{Z}$ and generating a "yes" or "no" answer to the question about existence of the roots of this polynomial over $\mathbb{Z}$.

At the time Hilbert formulated his question a formal notion of an algorithm, let alone computer program, did not yet exist. He asked for a process terminating in a finite number of steps, and this was later interpreted to mean an algorithm. The theorem proved by Davis, Putnam, Robinson and Matiyasevich showed that such an algorithm does not exist.

Lagrange's four-squares theorem from 1772 establishes that every natural number can be expressed as the sum of squares of four integers. Hence, the algorithmic solvability of a Diophantine equation in the integers is equivalent to the algorithmic solvability of a Diophantine equation in the natural numbers.

The first step towards the solution of the problem was made by Davis in 1949 when he showed that any computably enumerable of natural numbers has the following form:

$$\{a \mid \exists y \forall k \leqslant y \exists x_1, \ldots, x_n : p(a, k, y, x_1, \ldots, x_n) = 0\},$$

where $p(\ldots)$ is a polynomial with coefficients in $\mathbb{Z}$ and all variables range over $\mathbb{Z}$.

It is not hard to see that a set of natural numbers defined using existential quantifiers and a polynomial equation is computably enumerable. More precisely, let $p(t, x_1, \ldots, x_n)$ be a polynomial in $n + 1$ variables and consider the following set S of natural numbers:

$$\{a \in \mathbb{N} \mid \exists x_1 \in \mathbb{N} \ldots \exists x_n \in \mathbb{N} : p(a, x_1, \ldots, x_n) = 0\}.$$

We can enumerate all $(n + 1)$-tuples of natural numbers and plug them into the polynomial p. Each time the result is 0, we add the first coordinate of the

$(n + 1)$-tuple to S eventually listing every element of S. The polynomial p is called a *Diophantine definition* of S and sets defined using existential quantifiers and polynomial equations are called *Diophantine sets*.

In 1953, Davis [4] established that the collection of Diophantine sets is closed under unions and intersections, but not under complements.

Observe that Davis's formula defining all computably enumerable sets is very similar to the characterization of Diophantine sets (which clearly defines at least some computably enumerable sets). In 1949, Davis conjectured that every computably enumerable set is definable by an existential polynomial formula. It took 20 years for this conjecture to be proven.

Davis and Putnam proved that, under an additional hypothesis, the bounded universal quantifier can be eliminated, in favor of an "exponential Diophantine equation," that is, an equation where variables are allowed to appear in the exponents. The additional hypothesis was at the time a conjecture, but is now a theorem, and concerns the lengths of arithmetic progressions of primes. Robinson eliminated the need for the conjecture (Figure 5 and [32]). Finally in 1969, using Fibonacci numbers, Matiyasevich showed that exponential Diophantine equations can be replaced by polynomial equations.
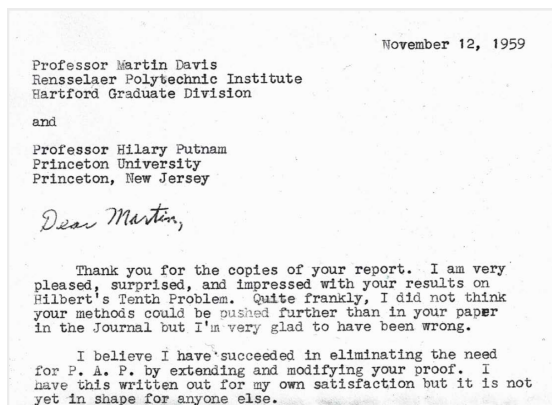

FIGURE 5. Julia Robinson announces the possibility of getting rid of the hypothesis made by Davis and Putnam. (Courtesy of Harold Davis)

It is not hard to see that the theorem proved by Davis, Putnam, Robinson and Matiyasevich implies a negative answer to Hilbert's question. Indeed, assume that an algorithm requested by Hilbert exists. Let the set S defined above be a computably enumerable set that is not computable. Then we can determine whether an $a \in \mathbb{N}$ is in S by determining whether the polynomial $p(a, x_1, \ldots, x_n)$ has roots in $\mathbb{N}$. Hence, if the algorithm for solving Diophantine equations exists, then there is an algorithm to determine membership in S. This contradicts our assumption that S is not computable. Therefore, the algorithm requested by Hilbert does not exist.

In the Preface to 1982 Dover edition of his book *Computability and Unsolvability*, Davis wrote: "One of the great pleasures of my life came in February 1970, when I learned of the work of Yuri Matiyasevich which completed the proof of the crucial conjecture and thereby showed that Hilbert's Tenth Problem is recursively unsolvable."
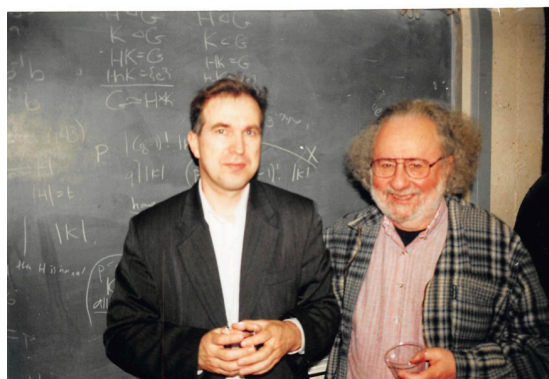

FIGURE 6. Yuri Matiyasevich with Martin. (Courtesy of Harold Davis)

Other ramifications. The *atomic diagram* of a structure $\mathscr{A}$ is the set of all atomic and negations of atomic sentences allowing additional constants for elements of the domain, which are true in $\mathscr{A}$. A structure is *computable* if the characteristic function of its atomic diagram is computable. The standard model of arithmetic, $\mathscr{N} = (\mathbb{N}, +, \cdot, S, 0)$, the natural numbers with addition, multiplication, successor function, and zero, is a computable structure. Gödel established that all computable relations are definable in $\mathscr{N}$. For any computable relation there are two natural defining formulas: one with a block of existential quantifiers followed by a formula with only bounded quantifiers, $\forall x < y$ and $\exists x < y$, and the other one with a block of universal quantifiers followed by a formula with only bounded quantifiers. A block of existential (universal) quantifiers can be replaced by a single existential (universal) quantifier by coding tuples of natural numbers by a single natural number. It follows from the proof of Hilbert's Tenth Problem that bounded quantifiers can be eliminated

from the above formulas, so a computable set is definable in $\mathcal{N}$ both by an existential and a universal formula.

"Positive aspects of a negative solution". This was a part of the title of the 1974 paper by Davis, Matiyasevich and Robinson [21]. Perhaps this title was a response to some opinions in the mathematical community that the negative answer to Hilbert's question about polynomial equation meant that the subject matter was closed. Nothing could have been further from the truth.

Among other things, the authors of the 1974 article explained that in a manner of speaking the negative solution was inevitable in part because a big part of Mathematics can be encoded into polynomial equations, e.g., Riemann's Hypothesis. More precisely, Riemann Hypothesis is true if and only if a certain polynomial equation with known coefficients has no integer solutions! Such polynomials exist for many other famous problems: Goldbach's conjecture, consistency of ZFC, etc. Thus existence of an algorithm to solve polynomial equations would resolve unreasonably many open questions in mathematics.

Perhaps the most positive consequence of Davis–Putnam–Robinson–Matiyasevich theorem is that it, together with definability results of Robinson, became a foundation of a new field: definability and decidability in number theory. This field seeks to understand what is definable and decidable in the language of rings (i.e., the language of polynomial equations) and its various dialects over rings and fields of interest to number theory. From its inception this field was situated on the boundary of several mathematical areas: number theory, algebraic geometry, model theory and computability theory and led to some interesting interactions between these fields.

The question of $\mathbb{Q}$. Perhaps the most important question in this new area is the analog of Hilbert's Tenth Problem for $\mathbb{Q}$. More precisely, the question is whether there exists an algorithm that can determine whether an arbitrary polynomial equation in several variables with integer coefficients has solutions in $\mathbb{Q}$. One can show that a positive answer to Hilbert's question for $\mathbb{Z}$ implies a positive answer to the question over $\mathbb{Q}$. However, the reverse implication is not clear.

One way to show that there is no algorithm to determine whether polynomial equations have solutions over $\mathbb{Q}$ is to construct a Diophantine definition of $\mathbb{Z}$ over $\mathbb{Q}$. However, there are conjectures by Mazur and others implying that such a definition does not exist. The question concerning (non)existence of this Diophantine definition is another major problem in the area.

On a personal note AS. I first spoke to Martin (on the phone) in the Spring of 1983 when I was deciding on a graduate school. Martin encouraged me to come to NYU. While there, I was lucky enough to take his class on Hilbert's Tenth Problem. Martin developed a different method for showing that exponential equations were Diophantine (polynomial) using the Pell equation in place of Fibonacci numbers. His method and its generalizations to norm equations served me well in many a paper.

I continued to be in touch with Martin until his death seeking his advice on many issues. He was mathematically engaged until the very end. I believe his last talk took place online. It was the opening talk for the MSRI semester on Definability, Decidability and Computability in Number Theory mentioned above.

3. Automated Reasoning

Martin began to program computers in 1951, when he was recruited in a group that developed programs for an ORDVAC machine, in support of the military during the Korean War [1]. He was assigned the task of writing, in absolute binary machine language, the prototype of a system by which ORDVAC was to navigate 100 airplanes in real time. After this tumultuous training on concrete programming,

---

[1]Martin then entered the organization, led by Frederick Seitz, named Control Systems Laboratory. Thanks to this move and, subsequently, enjoying a two-year ONR grant at the IAS in Princeton, he managed to avoid being inducted into the army. Concerning ORDVAC and JOHHNIAC, cf. [17, p. 162].

FIGURE 8. Participants at a meeting at the American Institute of Mathematics on extensions of Hilbert's Tenth Problem, 2005.

that lasted roughly one year, Martin was confident enough on his skills with computers that he managed to receive funding for a project on implementing Presburger's decision algorithm for integer arithmetic without multiplication. Martin's implementation of that algorithm took place on a JOHNNIAC computer available at the Institute for Advanced Study in Princeton in the summer of 1954. The prover could only ascertain very simple statements [6, 12], yet its accomplishment marked a milestone in computational logic. The preface of [38] says that it produced "what appears to be the first computer generated mathematical proof," and this accomplishment qualifies Martin as a trailblazer of the field today known as 'automated reasoning.' For some twenty-five years,[2] Martin continued to contribute to that field.

In the late 1950s, the seminal report [22] on computational methods for the propositional calculus arose from his collaboration with the distinguished analytic philosopher Hilary Putnam. With him Martin enjoyed discussing "all day long about everything under the sun" for various consecutive summers. The Davis-Putnam-Logemann-Loveland procedure [20] (DPLL for short), still fundamental in today's architectures of fast Boolean satisfiability solvers, was rooted in that collaboration. Between 1958 and 1960, simultaneously with the Davis-Putnam and DPLL projects, three major projects (led by Gilmore, Dunham-Fridshal-Sward, and Wang,

respectively—see [31]) were developing propositional provers. It was Davis and Putnam who set up the overall organization that, after them, would become standard in the automation of quantification theory. They adopted the Conjunctive Normal Form (CNF, a propositional conjunction of disjunctions of affirmed or denied logical variables) in pursuing an unsatisfiability test and embedded their rules for propositional logic into the enhanced proof framework. Martin and Hilary Putnam viewed a tester able to establish whether or not a given CNF formula is truth-functionally satisfiable as a key component in a general-purpose procedure for quantification theory.[3] This general procedure can then be applied to obtain proofs (by contradiction) in virtually any mathematical domain. It is surprising that, to these days, the DPLL-procedure constitutes a kernel of any efficient CNF-tester.

Any propositional formula can be brought to an equisatisfiable CNF formula in linear time (cf. [41]), hence CNF-satisfiability would become paradigmatic of the whole collection of NP-hard problems.[4] Martin never took sides on the "P vs NP" problem: he argued that we have no reliable intuition of what an algorithm of, say, complexity n^{27} can do. He discussed his views on the problem at more length in [18].

Martin and Hilary Putnam had made it clear that their method would outperform competitors of the time: by exploiting it in a 30-minute *hand computation*, they in fact validated a claim that a theorem-proving program developed by Paul C. Gilmore had been unable to validate with a 21-minute run on an IBM 704 machine [23]. The improvement was not due to a better handling of quantifiers, but due to an improvement in the propositional part [36]. Later, the promise of these hand computations was realized in computer implementations. Davis and Putnam's proof procedure for finitely axiomatized theories was implemented by Logemann and Loveland at New York University. They found and removed a bottleneck in the propositional-level component of the procedure. Later an implementation in LISP at Bell Labs gave further incremental improvements.[5]

---

[2]Still much later, around 1990, Martin would again look at the problem of automatic proof discovery. In the paper [19] he coauthored, that presents first-order predicate calculus under a very unusual light, he points out: "The above very elementary examples only hint at the kinds of proof procedures which our free variable formulation should make possible. But there is reason to believe that they may turn out to be of interest".

[3]Let us recall that in quantification theory, unlike propositional logic, the problem of validity is semi-decidable (that is, computably enumerable) without being algorithmically solvable. That is to say: while a systematic search will sooner or later unearth the proof of a theorem, rejecting an unprovable conjecture may turn out impossible.

[4]Cf. https://www.claymath.org/wp-content/uploads/2022/06/pvsnp.pdf.

[5]The mentioned implementation at NYU improved the 30 minute hand computation time to 2 minutes (cf. [20]).

The Bell Labs implementation of Martin's method [7, 2, 42] was named *Linked Conjunct.* The operating principle required that each logical variable in an unsatisfiable CNF formula would be paired with the same variable with the opposite sign in another conjunct. Not long after, there would be a proliferation of new proof search methods arising from John Alan Robinson's influential resolution principle[6]. It turns out, however, that many of these refinements can be naturally explained from the standpoint of Linked Conjunct [34].

FIGURE 9. Alberto Policriti, Martin Davis, Eugenio Omodeo, and Franco Parlamento in Trieste in 2009.

The original expectations about stand-alone theorem provers have been somewhat retargeted, over the years, to proof-checking systems that range from highly interactive reasoning assistants to mere proof-script verifiers. Martin also had a role in this change of perspective, as witnessed by his papers [26] and [11]. The former of these, jointly authored by him and his lifelong friend and colleague "Jack" (namely Jacob T. Schwartz, cf. [24, 25, 14]), addressed the issue of *metamathematical extensibility* in a full-blown program- and proof-verification technology. Which mechanisms can ensure long-term reliable use of a proof checker which undergoes augmentations with new symbols, schemes of notation, and extended rules of inference?

As for [11], it stemmed from Martin's experimentation with Richard Weyhrauch's FOL proof checker developed at John McCarthy's Artificial Intelligence Laboratory.[7] Martin recounts in [15]: "I found it neat to be able to sit at a keyboard and actually develop a complete formal proof, but I was irritated by the need to pass through many painstaking tiny steps to justify inferences that were quite obvious", and then

adds: "Using the LISP source code for the linked-conjunct theorem prover..., a Stanford undergraduate successfully implemented an 'obvious' facility as an add-on to FOL".

On a personal note EO. In 1975 Martin offered a summer course on computability in the pretty Italian town of Perugia. The dozen students in his class were initially amazed at the discrepancy between Martin as an unpretentious, easygoing person and his reputation as a distinguished scholar. Admiration quickly prevailed over astonishment when Martin began his lectures: for the entire one-month duration of his course, concepts remained clear, precise, and accessible. Even when he reached his cherished advanced topic: Hilbert's Tenth Problem.

I owe to having been a student in Martin's class in Perugia the fact that I could complete my academic formation at NYU, where Martin was my advisor for a Master's and next for a Ph.D. degree in Computer Science. A stream of Italian students and researchers (three had been my classmates in Perugia; Alberto Policriti and others belong to a successive generation) would, like me, reach Martin overseas in the following decade. This testifies to the influence that Martin's crystal-clear lectures, and the subtlety with which he addressed issues of philosophical relevance and depth, used to exert on his audience — in Italy much as elsewhere.

On a personal note AP. On the evening of a beautiful day of the fall of 1990 I was invited by Martin at his place in the upper-west side in New York, to a "party for two Yuri's". The two Yuri's were Yuri Gurevich and Yuri Matiyasevich, both in town and hosted by Martin. For Yuri Matiyasevich that was the first visit to the United States.

The number and (even more) the names of the people invited to the party were — especially for a young Italian Ph.D. student — rather overwhelming, and I was definitely scared when I was greeted amicably by Virginia upon my arrival. However, every tension promptly dissolved when Martin introduced Yuri Matiyasevich to the audience. He started recalling what remained to be proved after his first reduction of Hilbert's Tenth Problem and how, at the time, he conjectured that the last pending issue (a number theoretic hypothesis raised by Julia Robinson around 1950) would have certainly be solved by a ... *clever young Russian* ... in the near future.

Yuri was then introduced to everybody as the (constructive and positive) proof of his conjecture.
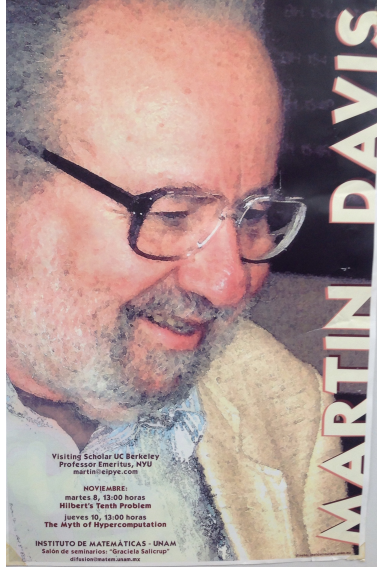

Figure 10. Announcement of a conference by Martin in Mexico. (Courtesy of Laura Elena Morales Guerrero)

4. Conclusions

Recollections of contributions by Martin to computability theory, Hilbert's Tenth Problem, and Automated Reasoning have been scattered all over the preceding text; many more could be cited: e.g., in [9], Martin stretches the algorithmic unsolvability of Hilbert's Tenth Problem into this result: For each proper nonempty subset A of $\mathbb{N} \cup \{\aleph_0\}$, no algorithm can establish, given any polynomial p with integer coefficients, whether the number of distinct positive integer solutions to the equation $p = 0$ belongs to A.

Beyond the proofs of specific theorems, Martin's scientific legacy included a broader contribution in the promotion of formal methods and theoretical computer science. He significantly contributed to the recognition of computability theory as an autonomous branch of mathematics. Martin developed a program in logic, and formed a logic group, first at Yeshiva University (being able to involve in it stars such as Raymond Smullyan) and then at the Bronx campus of the Courant Institute (NYU). In the 1960s, he devoted a good deal of time and energy in preparing an anthology of fundamental articles by Gödel, Turing, Post, Kleene, and Rosser [8]. His involvement with symbolic logic originated in the 1940s from his passionate interest in the foundations of real analysis, which also led him to author a classic on nonstandard analysis in the 1970s and to serve for decades as

the moderator of FOM, an automated e-mail list for discussing foundations of mathematics (see `https://cs.nyu.edu/mailman/listinfo/fom`).

Over the years, Martin lectured in several countries (to cite a few: Italy, Japan, India, England, Russia, and Mexico), and his lectures have—along with his publications—exerted a wide influence. The centennial of Frege's *Begriffsschrift*, Martin reports, "fundamentally changed the direction of my work" [15]: Being invited to place some contemporary trends in a proper historical context, he finds "trying to trace the path from ideas and concepts developed by logicians ... to their embodiment in software and hardware ... endlessly fascinating" [15].

Figure 11. Martin and his wife Virginia in 2019. (Courtesy of Nina and Yuri Matiyasevich)

References

[1] C. J. Ash and J. F. Knight. *Computable Structures and the Hyperarithmetical Hierarchy*. Number 144 in Studies in Logic and the Foundations of Mathematics. Elsevier, 2000.

[2] T. J. Chinlund, M. Davis, P. G. Hinman, and M. Douglas McIlroy. Theorem-proving by matching. Technical report, Bell Telephone Laboratories, Incorporated, Murray Hill, New Jersey, 1964.

[3] M. Davis. *On the theory of recursive unsolvability*. PhD thesis, Princeton University, May 1950.

[4] M. Davis. Arithmetical problems and recursively enumerable predicates. *J. Symbolic Logic*, 18(1):33–41, 1953.

[5] M. Davis. A note on universal Turing machines. In Claude E. Shannon and John McCarthy, editors, *Automata Studies*, volume 34 of *Ann. of Math. Stud.*, pages 167–175. Princeton University Press, 1956.

[6] M. Davis. A program for Presburger's algorithm. In *Summaries of talks presented at the Summer Institute of Symbolic Logic in 1957 at Cornell University*, volume 2, pages 215–223, Princeton, NJ, 1960. Communications Research Division, Institute for Defense Analyses. Reprinted as "A computer program for Presburger's algorithm" in [38, pp. 41–48].

[7] M. Davis. Eliminating the irrelevant from mechanical proofs. In *Proc. Symp. Appl. Math.*, volume 15, pages 15–30, Providence, RI, 1963. AMS. (Reprinted in [38, pp. 315–330]; Russian transl. in *Kiberneticheskiy sbornik. Novaya seriya*, 7, 1970, pp. 160–179).

[8] M. Davis, editor. *The Undecidable — Basic Papers in Undecidable Propositions, Unsolvable Problems and Computable Functions*. Raven Press, New York, 1965. Corrected reprinted edition, Dover Publications, Mineola, NY, 2004.

[9] M. Davis. On the number of solutions of Diophantine equations. *Proc. Amer. Math. Soc.*, 35(2):552–554, 1972.

[10] M. Davis. The mathematics of non-monotonic reasoning. *Artif. Intell.*, 13(1-2):73–80, 1980.

[11] M. Davis. Obvious logical inferences. In *Proceedings of the 7th IJCAI* Volume 1, pages 530–531, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc.

[12] M. Davis. The early history of Automated Deduction. In John Alan Robinson and Andrei Voronkov, editors, *Handbook of Automated Reasoning*, pages 3–15, vol. 1. Elsevier and MIT Press, 2001.

[13] M. Davis. The myth of hypercomputation. In Christof Teuscher, editor, *Alan Turing: Life and Legacy of a Great Thinker*, pages 195–211. Springer Berlin Heidelberg, 2004.

[14] M. Davis. Jack Schwartz meets Karl Marx. In Davis and Schonberg [25], pages 23–37.

[15] M. Davis. My life as a logician. In Omodeo and Policriti [35], pages 1–33.

[16] M. Davis. Universality is ubiquitous. In Alisa Bokulich and Juliet Floyd, editors, *Philosophical Explorations of the Legacy of Alan Turing – Turing 100*, volume 234 of *Boston Studies in the Philosophy and History of Science*, pages 153–158. Springer Verlag, 2017.

[17] M. Davis. *The Universal Computer: The Road from Leibniz to Turing*. CRC Press, third edition, 2018.

[18] M. Davis. Seventy years of computer science. In A. Blass, P. Cégielski, N. Dershowitz, M. Droste, and B. Finkbeiner, editors, *Fields of Logic and Computation III - Essays Dedicated to Yuri Gurevich on the Occasion of His 80th Birthday*, volume 12180 of *Lecture Notes in Computer Science*, pages 105–117. Springer, 2020.

[19] M. Davis and R. Fechter. A free variable version of the first-order predicate calculus. *J. Logic Comput.*, 1(4):431–451, 1991.

[20] M. Davis, G. Logemann, and D. W. Loveland. A machine program for theorem-proving. *Commun. ACM*, 5(7):394–397, 1962.

[21] M. Davis, Yu. Matijasevič, and J. Robinson. Hilbert's tenth problem. Diophantine equations: positive aspects of a negative solution. In *Mathematical Developments Arising From Hilbert Problems*, volume 28 of *Proc. Sympos. Pure Math.*, pages 323–378, Providence, RI, 1976. AMS. Reprinted in [37, pp. 269–378].

[22] M. Davis and H. Putnam. Feasible computational methods in the propositional calculus. Technical report, Rensselaer Polytechnic Institute, Research Division, Troy, New York, October 1958. Reprinted in [35, pp. 373–408].

[23] M. Davis and H. Putnam. A computing procedure for quantification theory. *J. ACM*, 7(3):201–215, 1960.

[24] M. Davis and E. Schonberg. Jacob Theodore Schwartz 1930–2009. *Biographical Memoirs of the National Academy of Sciences*, 2011. 19 pages.

[25] M. Davis and E. Schonberg, editors. *From Linear Operators to Computational Biology: Essays in Memory of Jacob T. Schwartz*. Springer, 2013.

[26] M. Davis and J. T. Schwartz. Metamathematical extensibility for theorem verifiers and proof-checkers. *Comput. Math. Appl.*, 5:217–230, 1979.

[27] R. G. Downey and D. R. Hirschfeldt. *Algorithmic Randomness and Complexity*. Theory and Applications of Computability. Springer, 2010.

[28] A. Jackson. Interview with Martin Davis. *Notices Amer. Math. Soc.*, 55(5):560–571, 2008.

[29] A. Jackson. Life as a logician. *Communications of the ACM*, 2021. `https://cacm.acm.org/news/255113-life-as-a-logician/fulltext`.

[30] A. N. Kolmogorov. Three approaches to the definition of the concept 'quantity of information'. *Probl. Peredachi Inf.*, 1:3–11, 1965.

[31] D. Loveland, S. Ashish, and B. Selman. DPLL: The core of modern satisfiability solvers. In Omodeo and Policriti [35], pages 315–335.

[32] J. Robinson M. Davis, H. Putnam. The decision problem for exponential diophantine equations. *Ann. of Math.*, 74(2):425–436, 1961.

[33] Yu. V. Matiyasevich. *Hilbert's tenth problem*. The MIT Press, Cambridge (MA) and London, 1993. xxii+264 pp.

[34] E. G. Omodeo. The Linked Conjunct method for automatic deduction and related search techniques. *Comput. Math. Appl.*, 8(3):185–203, 1982.

[35] E. G. Omodeo and A. Policriti, editors. *Martin Davis on Computability, Computational Logic, and Mathematical Foundations*, volume 10 of *Outstanding Contributions to Logic*. Springer, 2016.

[36] D. Prawitz, H. Prawitz, and N. Voghera. A mechanical proof procedure and its realization in an electronic computer. *J. ACM*, 7(2):102–128, 1960.

[37] J. Robinson. *The collected works of Julia Robinson*, volume 6 of *Collected Works*. AMS, Providence, RI, 1996. ISBN 0-8218-0575-4. With an introduction by Constance Reid. Edited and with a foreword by Solomon Feferman. xliv+338 pp.

[38] J. Siekmann and G. Wrightson, editors. *Automation of Reasoning 1: Classical Papers on Computational Logic 1957-1966*. Springer, Berlin, Heidelberg, 1983.

[39] R. J. Solomonoff. A formal theory of inductive inference. *Information and control*, 7:1–22, 1964.

[40] C. Spector. Recursive well-orderings. *The Journal of Symbolic Logic*, 20:151–163, 1955.

[41] G. S. Tseitin. On the complexity of derivation in propositional calculus. In A. O. Slisenko, editor, *Studies in Constructive Mathematics and Mathematical Logic, Part 2*, pages 115–125. Consultants Bureau, New York, 1970.

[42] D. L. Yarmush. The Linked Conjunct and other algorithms for mechanical theorem-proving. Technical Report IMM 412, Courant Institute of Mathematical Sciences, New York University, July 1976.