DIOPHANTINE COMPLEXITY[†]

Leonard Adleman and Kenneth Manders

Computer Science Division
Department of Electrical Engineering and Computer Sciences
and the Electronics Research Laboratory
University of California at Berkeley
Berkeley, California 94720

and

Group in Logic and Methodology
University of California at Berkeley
Berkeley, California 94720

(Abstract)

I. Introduction

In the 1930's Gödel together with Church, Kleene, and Turing established a relationship between computation and elementary number theory. Using techniques developed by Robinson, Putnam, Davis[4] and Matijasevic[7] in their celebrated solution to Hilbert's 10th problem, we began in [2] a detailed analysis to determine what consequences this relationship might have for computational complexity. We found that there were consequences not only for computational complexity (nontrivial lower bounds on decision procedures for polynomials, the polynomial compression theorem) but also for number theory (new polynomial definitions of primality and exponentiation) and logic (a syntactical characterization of the elementary-computable functions).

In this paper, making further use of techniques of Matijasevic[7,9] and Robinson[8], we make fundamental improvements in the proofs used in [2] and produce exponential improvements in all results found there. We also introduce a new simple type of computational device which provides a direct interface between number theory and computational theory. We conjecture that this device is a fully adequate model for non-deterministic computation. If our conjecture is true, then machines which use only addition and multiplication are as fast as those with the full repertoire of Turing machine operations. As a consequence number theoretic techniques would be directly applicable to computational complexity problems. We give several theorems in support of the conjecture, among them a normal form theorem for nondeterministic Turing machines. We also describe a regular set which is computable in polynomial time on our new machine if and only if our conjecture is true. If the conjecture is false, then this regular set isolates in a very precise way the difference between the operations of addition and multiplication and the operations of a Turing machine.

II. Nondeterministic Diophantine Machines

We begin by introducing a new type of computational device -- the nondeterministic diophantine machine (NDDM). It is both number-theoretically and machine-theoretically convenient and will serve as an interface between the theories. It also relieves us of the need for number-theoretic notation and conventions in contexts where they would be inconvenient. Finally it helps illustrate an important conceptual point: in comparing number theory and computational complexity we are really just comparing the relative "power" of the operations of addition and multiplication with those of Turing machines. Use of such a convenient formalism has been made possible by the development of techniques in number theory for the solution of Hilbert's 10th problem.

Given a multivariable polynomial $p(x_1,\dots,x_n)$ with integer coefficients, the corresponding NDDM is the nondeterministic Turing machine (NDTM) with the following algorithm:

"On input $a \in \omega$, guess $a_2,\dots,a_n \in \omega$.
If $p(a,a_2,\dots,a_n) = 0$, accept a."

For example, if $p(x_1,x_2) = x_1 - x_2^2$, then the corresponding NDDM has the algorithm

"On input $a \in \omega$, guess $a_2 \in \omega$.
If $a - a_2^2 = 0$, accept a."

It is easy to see that this NDDM accepts exactly the set of perfect squares. It is natural to ask what class of numerical relations on ω are accepted by NDDM's. The answer follows from Matijasevic's solution of Hilbert's 10th problem:

Theorem. For every numerical relation A on ω:
A is accepted by a NDDM $\Leftrightarrow$ A is accepted by a NDTM .

Thus in one sense, no computational power is lost in restricting NDTM's to those with the special NDDM algorithmic form. This provides initial justification for the use of NDDM's as a computational formalism, analogous to the proofs of the equivalence of recursive functions, Turing machines, etc. A focal point of our research has been to determine how much speed, if any, has been lost by restriction of NDTM's to NDDM's.

Our best answer is provided below in the Fundamental Theorem. First we give some definitions.

Definition. For all $A \subseteq \omega$, for all $\Phi\colon \omega \to \omega$ A is accepted on a NDDM within time Φ if and only if there is a polynomial $p(x_1,x_2,\dots,x_n)$ such that for all x

$$x \in A \Leftrightarrow (\exists |x_2|,\dots,|x_n| \le \Phi(x))[P(x_1,x_2,\dots,x_n) = 0]$$

where $|x|$ denotes the length of the number x.

This measurement of time differs slightly from that normally associated with NDTM's. Essentially, it ignores the cost of the multiplications and additions

[†] Research supported by National Science Foundation Grant DCR72-03725-A02.

that the NDDM makes, and only counts time when the machine is making its guesses. The complete Turing machine calculation on input x would take $(\Phi(x))^2$ or less steps. We choose this definition because it is convenient number-theoretically. In fact, the Fundamental Theorem (first form) as presented below holds (with a more cumbersome proof) even if time on NDDM's is measured exactly as on NDTM's.

Definition. For all functions $\Phi: \omega \to \omega$

(I) Φ is a super running time of type I if and only if there is a polynomial p with range in ω such that either:

(i) for all x, $\Phi(x) = p(|x|)$ or

(ii) for all x, $\Phi(x) = 2^{p(|x|)}$

(II) Φ is a super running time of type II if and only if there is a total-deterministic running time function $\hat{\Phi}$ such that for all x

$$2^{5\hat{\Phi}^2(x)-2} \le 4\Phi(x) \le 2^{2^{\hat{\Phi}(x)-1}}$$

(III) Φ is a super running time if and only if it is a super running time of type I or type II.

Fundamental Theorem (first form). For all $A \subseteq \omega$, for all super running times Φ:

(a) A is accepted on a NDDM within time Φ $\Rightarrow$ A is accepted on a NDTM within time Φ^2.

(b) A is accepted on a NDTM within time Φ $\Rightarrow$ A is accepted on a NDDM within time $2^{10\Phi^2}$

Fundamental Theorem (second form). For all $A \subseteq \omega$, for all super running times Φ:

(a) There exists a polynomial $P(x,x_2,\ldots,x_n)$ such that:

$$x \in A \Leftrightarrow (\exists |x_2|,\ldots,|x_n| \le \Phi(x))[P(x,x_2,\ldots,x_n) = 0]$$

$\Rightarrow$ A is accepted on a NDTM within time Φ^2.

(b) A is accepted on a NDTM within time Φ $\Rightarrow$ there is a polynomial $p(x,x_2,\ldots,x_n)$ such that:

$$x \in A \Leftrightarrow (\exists |x_2|,\ldots,|x_n| \le 2^{10\Phi^2(x)})[P(x,x_2,\ldots,x_n) = 0]$$

This result represents a fundamental improvement over similarly motivated results in [2]. The proof techniques available at the time (especially the crucial Bounded Quantifier Theorem) could not be improved to yield the present result. This result is obtained by a new and different approach to diophantine description of Turing machine computations incorporating recent ideas of Matijasevic.[9] The new techniques yield two kinds of theorems indicating that Turing machines are hardly (if at all) more powerful (i.e. faster) than NDDM's: The Fundamental Theorem and the Exact Normal Form Theorem for NDTM's in Section IV below. This is important because it suggests that direct and practical application of number-theoretic results and methods to computational theory is possible.

III. Consequences of the Fundamental Theorem

The Fundamental Theorem can be seen as a "bridge" between number theory and computational complexity. Below we present some consequences of the theorem.

We will consider the complexity of a number of computational problems. In each, the inputs will be polynomials some of which will be accepted and others rejected. The most famous example of such a problem is Hilbert's tenth -- where we are to accept those polynomials with solutions and reject all others. Matijasevic[7] has recently shown that this problem is unsolvable. We will show that certain variants of this problem, while solvable, are intractable (i.e. not solvable in polynomial time).

One reason why Hilbert's tenth problem is important and why solutions to polynomials are so heavily studied in number theory is that so many properties of numbers can be decided in terms of such solutions. For example,

for all a, a is a perfect square $\Leftrightarrow (\exists b)[a-b^2 = 0]$

for all a, a is composite $\Leftrightarrow (\exists b,c)[(b+2)(c+2) - a = 0]$

In these examples, as is often the case, we actually know something about the size of the solutions if they exist. For example it is trivial to see that

for all a, a is a perfect square
$\Leftrightarrow (\exists |b| \le |a|)[a-b^2 = 0]$

for all a, a is composite
$\Leftrightarrow (\exists |b|,|c| \le |a|)[(b+2)(c+2) - a = 0]$

or, giving bounds in terms of sizes of polynomials,

for all a, a is a perfect square
$\Leftrightarrow (\exists |b| \le |a-x^2|)[a-b^2 = 0]$

for all a, a is composite
$\Leftrightarrow (\exists |b|,|c| \le |(x_1+2)(x_2+2)-a|)[(b+2)(c+2) - a = 0]$

From this it is clear that if there was a polynomial running time algorithm which would recognize those polynomials with solutions in numbers whose size does not exceed the size of the polynomial, many number theoretic problems would also be computable in polynomial time. In [6] we showed that such an algorithm is unlikely. Below, we show that similarly motivated problems are in fact intractable.

Notation and Definitions. Let $P(x_1,\ldots,x_n)$ be a polynomial with integer coefficients and variables $x_1,\ldots,x_n$ ranging over natural numbers (only). For all $a_1,\ldots,a_n \in \omega$, let $P(a_1,\ldots,a_n)$ be the value of P evaluated at $a_1,\ldots,a_n$. $\langle a_1,\ldots,a_n\rangle$ is said to be a solution of P if and only if $a_1,\ldots,a_n \in \omega$ and $P(a_1,\ldots,a_n) = 0$.

For any functions f and g we say a set S infinitely often requires f steps on a multitape nondeterministic Turing machine if and only if for any nondeterministic multitape Turing machine M which accepts S, there is a positive rational c, and there are infinitely many $x \in S$ such that the number of steps taken by M on input x is greater than $cf(x)$.

Theorem 3.1. For all $c \in \omega$, $c > 1$,

if

$$S_c = \{P(x_1,\ldots,x_n) | (\exists |x_1|,\ldots,|x_n| \le 2^{|p|^{2c}}) [P(x_1,\ldots,x_n) = 0]\}$$

then

(a) S_c infinitely often requires $|p|^c$ steps on a multitape nondeterministic Turing machine.

(b) There exists a one-tape nondeterministic Turing machine which accepts S_c within $2^{(2|p|)^{2c}}$ steps.

Theorem 3.2. If

$$S = \{P(x_1,\ldots,x_n) | (\exists |x_1|,\ldots,|x_n| \le 2^{2^{2|p|}}) [P(x_1,\ldots,x_n) = 0]$$

then

(a) S infinitely often requires $2^{|p|}$ steps on a multitape nondeterministic Turing machine.

(b) There exists a one-tape nondeterministic

Turing machine which accepts S within $2^{2^{2|p|+1}}$ steps.

In keeping with the intuition that searching polynomials for solutions of greater and greater size requires more and more time[†] we have:

Theorem 3.3. For all super running times Φ if

$$S = \{P(x_1,\dots,x_n) \mid (\exists |x_1|,\dots,|x_n| \leq 2^{10\Phi^2(m(p))}) \\ [P(x_1,\dots,x_n)=0]\}$$

then

(a) S infinitely often requires $\Phi(m(p)/4)$ on a multitape nondeterministic Turing machine.

(b) There exists a one-tape nondeterministic Turing machine which accepts S within $2^{20\Phi^2(m(p))}$ steps

where m is a function independent of Φ such that for some $c \in \omega$, $p/c \leq m(p) \leq p$ (i.e. $m(p)$ is roughly p).

Corollary 3.4 (Diophantine Compession Theorem). For all recursive functions f, there exists a recursive function h such that:

If

$$S = \{P(x_1,\dots,x_n) \mid (\exists |x_1|,\dots,|x_n| \leq 2^{10h^2(m(p))}) \\ [P(x_1,\dots,x_n)=0]\}$$

then

(a) S infinitely often requires $h(m(p)/4) > f(p)$ steps on a multitape nondeterministic Turing machine.

(b) There exists a one-tape nondeterministic Turing machine which accepts S within $2^{20h^2(m(p))}$ steps.

In fact Theorems 3.1-3.4 can be strengthened so that the class of polynomials considered can be restricted to those with a fixed finite number of variables.[2] The Diophantine Compression Theorem is a concrete example of the analogous result in abstract complexity theory.

The Fundamental Theorem can also be used to improve the bound on the size of solutions of polynomials representing the primes.[2] The proof of this fact relies on Pratt's fast nondeterministic algorithm for primality[10] and appears to have no natural number theoretic counterpart.

Theorem 3.5. There exists a polynomial $p(x_1,\dots,x_{400})$ such that

$$x \in \text{Primes} \Leftrightarrow (\exists |x_2|,\dots,|x_n| \leq 2^{10|x|^6}) \\ [P(x,x_2,\dots,x_n)=0]$$

IV. Polynomial Time Computations on a NDDM

We now study in detail the class D of numerical relations accepted by NDDM's in time polynomial in the size of the input. Comparison of D to NP turns out to be analogous to the well-known comparison of P to NP. We define the concepts of D-reducibility and D-completeness (analogous to p-reducibility and NP-completeness) and obtain very striking D-complete problems which are (surprisingly) in P. Using any one of these, we obtain a normal form for NDTM's as NDDM's with severely restricted deterministic computational ability. These new normal forms are fully equivalent to NDTM's, not only as regards computational strength, but also in the time required for computations. The simplicity of the D-complete problems suggests that $NP = D$.

Definition.

(i) For all $n \in \omega$, D^n is the set of all numerical relations R definable by a formula of the form:

$$\langle x_1,x_2,\dots,x_m\rangle \in R \Leftrightarrow \exists y_1,\dots,y_n \leq 2^{q(|(x_1+x_2+\cdots+x_m)|)} \\ [P(x_1,x_2,\dots,x_m,y_1,\dots,y_n)=0]$$

where q and p are polynomials.

(ii) $D = \bigcup_{i\in\omega} D^i$

(iii) For any m-ary numerical relation R and any ℓ-ary numerical relation S: R is D-reducible to S (notation: $R \leq_D S$) if and only if R is definable by a formula of the form:

$$\langle x_1,\dots,x_m\rangle \in R \\ \Leftrightarrow \exists y_1,\dots,y_\ell,y_{\ell+1},\dots,y_n \leq 2^{q(|x_1+\cdots+x_m|)} \\ [P(x_1,\dots,x_m,y_1,\dots,y_n)=0 \;\&\; \langle y_1,\dots,y_\ell\rangle \in S]$$

where q and p are polynomials.

(iv) For any numerical relation R in NP: R is D-complete[†] if and only if every other numerical relation in NP is D-reducible to R.

Note that the definition of D agrees with the machine-theoretic one given above, and that $D \subseteq NP$. All the notions in this section have natural definitions in machine theoretic and number theoretic terms. Thus $R \subseteq W$ is D-reducible to $S \subseteq W$ if and only if there is a polynomial $P(x_1,\dots,x_n)$ such that the NDTM M with the following algorithm:

"On input x, guess $x_2,\dots,x_n$.
If $x_2 \in S$ and $P(x,x_2,\dots,x_n) = 0$, accept x."

accepts R in polynomial time. Clearly $R \leq_D S$ and $S \in D$ implies $R \in D$.

Definition.

(i) R_0 is the regular set $(00+10)^*$ of binary numbers with all even digits zero.

(ii) Nocarry is the binary relation $\{\langle \ell_1,\ell_2\rangle$: ℓ_1, $\ell_2 \in \omega$ and for all i if the i^{th} digit of ℓ_1 in binary is 1 then the i^{th} digit of ℓ_2 in binary is $0\}$.

(iii) Rev is the deterministic even linear context free language generated by $S \rightarrow 1S1|0S0|0c0|1c1$.

(iv) RevNoc is the deterministic even linear context free language generated by $S \rightarrow 1S0|0S1|0S0|0c1|1c0|0c0|$.

Note that all these relations (sets) can be decided in deterministic polynomial time, R_0 by a deterministic finite state machine with only 3 states, Rev and RevNoc by deterministic pushdown automaton.

Theorem 4.1.

(a) R_0 is D-complete.

(b) $R_0 \in D \Leftrightarrow D = NP$.

The ultimate simplicity of R_0, and the Fundamental Theorem together with results in [1], induces our

[†] This is not strictly true. There are gaps in the size distribution of solutions to polynomials just as there are gaps between running times of Turing machines.

[†] In [6] we refer to this as NP(D)-complete.

Conjecture. D = NP

In considering whether $R_0 \in D$, we find:

Theorem 4.2. Let R be a regular set defined by a regular expression formed by "+" and "•" from $(0+1)^* \cup \{\alpha^*: \alpha \in (0+1)^*\}$. (These are exactly the regular expressions without nontrivial occurrences of "+" within the scope of "*".) Then $R \in D$.

Theorem 4.3. The following are D-equivalent (mutually D-reducible) and hence D-complete.

(a) R_0

(b) Nocarry

(c) Rev and RevNoc (together)

Definition. A Nondeterministic Diophantine R_0-Machine (NDDR) is a NDDM with the added capability of asking whether the first number it guesses is in R_0 (and getting either a "yes" answer or no answer at all). Specifically, given a polynomial $p(x_1,\ldots,x_n)$, the corresponding NDDR is the NDTM with algorithm

"On input a, guess $a_2,\ldots,a_n$ and if $p(a,a_2,\ldots,a_n) = 0$ and $a_2 \in R_0$, then accept a."

Theorem 4.4 (Exact Normal Form Theorem for NDTM's). For every numerical relation A on ω:

A is accepted in polynomial time on a NDTM

⇔ A is accepted in polynomial time on a NDDR .

Of course any relation D-equivalent to R_0 will give such a normal form for NDTM's. We find Theorem 4.4 particularly significant because it localizes very precisely the "nonarithmetical component" of the computational power of Turing machines if NP ≠ D.

Definition. For any class of relations K: K^c is the class of complements of the members of K.

Theorem 4.5. $D^c \subseteq D \Rightarrow NP^c = NP$

Theorem 4.6. $(D^8)^c \subseteq D \Rightarrow D = NP$

The last theorem follows from the fact that the complement of R_0 (i.e. the set of sequences with an even digit equal to 1) is in D^8, by an application of Theorem 4.1. The preceding theorem then follows. (In [6] using different methods, we have obtained an improvement of Theorem 4.5 to $(D^2)^c \subseteq NP \Rightarrow NP^c = NP$.) Where we stand on the relationship of D to D^c is indicated by

Theorem 4.7. $(D^1)^c \subseteq D$

This is because, by Sturm's Theorem, we can localize the roots of $p(a,x) = 0$ in a number of intervals $(n,n+1)$ bounded by the degree of $p(a,x)$ in x.

To further illustrate the close relationship of D and NP consider the Kleene type hierarchy formed from D,[2] where Σ_n^D is characterized as sets definable with n-1 alternations of blocks of polynomial bounded quantifiers (the first one of which is ∃) followed by a diophantine predicate (e.g. $\Sigma_1^D = D$). The following theorem reduces an important question about the Meyer-Stockmeyer Hierarchy[11] to the D-hierarchy.

Theorem 4.8

(i) A set S is in the Meyer-Stockmeyer Hierarchy if and only if it is in the D-hierarchy.

(ii) The Meyer-Stockmeyer Hierarchy collapses if and only if the D-hierarchy collapses.

V. Outline of Proof of the Fundamental Theorem

Part (a) of the Fundamental Theorem follows trivially from the definition of NDDM's and their running times. The proof of part (b) relies on the work of Matijasevic[7,9] and Robinson[8]. The techniques used differ from those used in [2] in that some highly number-theoretic methods (Gödel encoding lemma, Bounded Quantifier Theorem) are absent here and have been replaced by more computational methods; this leads to improved bounds.

In the proof we wish to express that a NDTM M halts on an input x in t steps. Instead of asserting the existence of the usual sequence of ID's, we will assert the existence of several binary sequences which combined have the same information. In particular, if M has z instructions (quintuples) $\sigma_1,\ldots,\sigma_z$ we will assert the existence of I_0, I_1, $I_ƀ$, $I_{\sigma_1},\ldots,I_{\sigma_z}$. Intuitively, I_0, I_1 and $I_ƀ$ each have t segments of length t where the i^{th} segment records how M's tape looks after i-1 instructions have been executed: I_1 will have a 1 in segment $m \le t$ at position $n \le t$ if and only if there is a 1 on the n^{th} tape square after M has executed m-1 instructions in the computation on input x being encoded. I_0 and $I_ƀ$ will give analogous information about zeros and blanks.

For each $j \le z$, I_{σ_j} will also have t segments of length t and will have a 1 in segment $m \le t$ at position $n \le t$ if and only if the m^{th} instruction executed by M in the computation on input x was σ_j and the tape head was positioned at the n^{th} tape square before execution of that instruction.

For simplification of the proof, we will assume that M has one semi-infinite tape, one accepting state, etc. The proof follows:

Lemma I. For all nondeterministic Turing machines M there exists a system of conditions S such that for all x, t; $t \ge 1$:

(i) M halts on input x in exactly t steps ⇒

(a) There exists $T,I_0,I_1,I_ƀ,I_{\sigma_1},\ldots,I_{\sigma_z}$ $z_0,z_ƀ \le 2t^2$ satisfying the system S.

(b) There exists $I_0,I_1,I_ƀ,I_{\sigma_1},\ldots,I_{\sigma_z}$, $z_0,z_ƀ \le 2t^2$ such that $t,I_0,I_1,I_ƀ,I_{\sigma_1},\ldots,I_{\sigma_z},z_0,z_ƀ$ satisfies the system S.

(c) For all $T,I_0,I_1,I_ƀ,I_{\sigma_1},\ldots,I_{\sigma_z},z_0,z_ƀ$: $T,I_0,I_1,I_ƀ,I_{\sigma_1},\ldots,I_{\sigma_z},z_0,z_ƀ$ satisfies the system S ⇒ $T \ge t$.

(ii) There exists $T,I_0,I_1,I_ƀ,I_{\sigma_1},\ldots,I_{\sigma_z},z_0,z_ƀ$ satsifying the system S ⇒ M halts on input x

where S is:

1) I_0, I_1, $I_ƀ$ are disjoint (where a, b are disjoint if and only if for all i, if the i^{th} digit of a in binary is 1 then the i^{th} digit of b in binary is 0).

2) $I_0 + I_1 + I_ƀ = 2^{T^2} - 1$.

3) $I_1 - x \equiv 0 \bmod (2^T)$

4) $I_{\sigma_1} \equiv 1 \bmod (2^T)$ & $I_{\sigma_2} \equiv I_{\sigma_3} \equiv \cdots \equiv I_{\sigma_z} \equiv 0 \bmod (2^T)$ or $I_{\sigma_2} \equiv 1 \bmod (2^T)$ & $I_{\sigma_1} \equiv I_{\sigma_3} \equiv \cdots \equiv I_{\sigma_z} \equiv 0 \bmod (2^T)$ (where with no loss of generality we assume σ_1 and σ_2 are the only legal 1st instructions (quintuples) M can execute (i.e. $\sigma_1 = \langle q_0 0 \cdots \rangle$ and $\sigma_2 = \langle q_2 1 \cdots \rangle$) and $\sigma_3 \cdots \sigma_z$ are the remaining instructions of M).

5) $I_{\sigma_z} + I_{\sigma_{z-1}} \neq 0$ (where with no loss of generality we assume M has exactly 2 halt instructions $\sigma_z = \langle q_h 00Rq_h \rangle$ and $\sigma_{z-1} = \langle q_h 11Rq_h \rangle$).

6) $I_{\sigma_1}, I_{\sigma_2}, \ldots, I_{\sigma_z}$ are disjoint.

7) $$2^{T-1} \sum_{j\in L_q} I_{\sigma_j} + 2^{T+1} \sum_{j\in R_q} I_{\sigma_j} - \sum_{j\in S_q} I_{\sigma_j} - C_q = 0$$

(one such clause for each state q of M) where:

$L_q = \{j \mid \sigma_j \text{ moves left to } q\}$

$R_q = \{j \mid \sigma_j \text{ moves right to } q\}$

$S_q = \{j \mid \sigma_j \text{ is executed from state } q\}$

$$C_q = \begin{cases} 1 & \text{if } q = q_0 \\ 0 & \text{if } q \neq q_0 \end{cases}$$

8) $\sum_{j\in R_1} I_{\sigma_j}$, I_0 & $I_{\text{ƀ}}$ are disjoint

$\sum_{j\in R_0} I_{\sigma_j}$, I_1 & $I_{\text{ƀ}}$ are disjoint

$\sum_{j\in R_{\text{ƀ}}} I_{\sigma_j}$, I_0 & I_1 are disjoint

where:

$R_1 = \{j \mid \sigma_j$ is an instruction reading a 1 (i.e. $\sigma_j = \langle\cdot 1 \cdots\rangle)\}$

$R_0 = \{j \mid \sigma_j$ is an instruction reading a 0$\}$

$R_{\text{ƀ}} = \{j \mid \sigma_j$ is an instruction reading a ƀ$\}$

9) $z_0 + z_{\text{ƀ}} + x = 2^T - 1$ &

$$(I_1 - x) - 2^T\Big(I_1 + \sum_{j\in S_{01}} I_{\sigma_j} + \sum_{j\in S_{\text{ƀ}1}} I_{\sigma_j} - \sum_{j\in S_{10}} I_{\sigma_j} - \sum_{j\in S_{1\text{ƀ}}} I_{\sigma_j}\Big) \equiv 0 \bmod (2^{T^2}) \quad \&$$

$$(I_0 - z_0) - 2^T\Big(I_0 + \sum_{j\in S_{10}} I_{\sigma_j} + \sum_{j\in S_{\text{ƀ}0}} I_{\sigma_j} + \sum_{j\in S_{0\text{ƀ}}} I_{\sigma_j} - \sum_{j\in S_{01}} I_{\sigma_j}\Big) \equiv 0 \bmod (2^{T^2}) \quad \&$$

$$(I_{\text{ƀ}} - z_{\text{ƀ}}) - 2^T\Big(I_{\text{ƀ}} + \sum_{j\in S_{0\text{ƀ}}} I_{\sigma_j} + \sum_{j\in S_{1\text{ƀ}}} I_{\sigma_j} - \sum_{j\in S_{\text{ƀ}0}} I_{\sigma_j} - \sum_{j\in S_{\text{ƀ}1}} I_{\sigma_j}\Big) \equiv 0 \bmod (2^{T^2})$$

where $S_{01} = \{j \mid \sigma_j$ reads a 0 and writes a 1$\}$, S_{10}, $S_{\text{ƀ}1}$, $S_{1\text{ƀ}}$, $S_{0\text{ƀ}}$ and $S_{\text{ƀ}0}$ are defined analogously.

Proof of Lemma I. Conditions 1) and 2) combined assert that I_0, I_1, and $I_{\text{ƀ}}$ together describe a tape with exactly one symbol (0, 1, or ƀ) in each square. Condition 3) asserts that the tape starts with x written on it. (With no loss of generality we assume that M is designed in such a way that 3) also asserts that all symbols after x on the initial tape are ƀ.) Condition 4) asserts that M starts in its start state and executes its first instruction while reading square 1. Further, other instruction sequences do not contradict this. Condition 5) asserts that M halts. Conditions 6), 7) and 4) allow for an inductive argument which shows:

(i) In any segment $m \leq T$, the I_σ's together assert that exactly one instruction was executed by M.

(ii) The instructions executed in successive segments follow legally "state to state".

(iii) The position of the head on the m^th instruction follows from the position of the head on the m-1^st instruction together with the type of instruction the m-1^st was (i.e. moves right or moves left).

Condition 8) together with conditions 1) and 2) assert that the tape *T* which I_0, I_1 and $I_{\text{ƀ}}$ describe is consistent with the instruction sequences in the sense that if I_{σ_j} asserts that a 1 was read on instruction m over tape square n then *T* has a 1 in the m^th segment n^th position. Similarly for 0 and ƀ. Condition 9) asserts that the string of instructions *I* described by $I_{\sigma_1}, \ldots, I_{\sigma_z}$ and the tape *T* described by I_1, I_0 and $I_{\text{ƀ}}$ have been reconciled. That is, segment m+1 of *T* follows from segment m by the m^th instruction of *I*.

If M halts on input x in exactly t steps, choosing $T = t$ and taking $I_{\sigma_1}, \ldots, I_{\sigma_z}$ corresponding to the sequence of ID's of M in the halting computation clearly leads to satisfaction of *S*. Thus (i)(a), (b) hold. By the above discussion, the existence of a solution $T, I_{\sigma_1}, \ldots, I_{\sigma_z}, z_0, z_{\text{ƀ}}$ to *S* implies that M halts on input x within T steps. Hence (ii) and (i)(c) hold. □

In the remainder of this section, a sequence of lemmas will be stated leading to a proof of the Fundamental Theorem; the proofs are omitted here; they are given in [1]. Beyond Lemma II, the development is similar to that in [2].

Matijasevic[9] observed that Kummer proved:

Lemma II. For all m, n, the following are equivalent:

(i) m and n are disjoint.

(ii) $2 \nmid \binom{m+n}{n}$

(iii) There exists $x, y \leq \binom{m+n}{m}$ satisfying:

(I) $x = \binom{m+n}{m}$

(II) $2y + 1 = x$

Let *S'* be the system of equations obtained from *S* by replacing the disjointness conditions 1), 6), and 8) of *S* by conditions of the form indicated in Lemma II(iii). We wish to obtain a system involving only polynomial equations. To do this we must eliminate the binomial coefficientsand and exponentiations 2^T occuring in *S'* in favor of such equations. Appropriate equations were given in [3] and [8], but without reference to the size of solutions which is crucial to our theorem. We begin with those equations and make certain modifications (Lemma VII below) to insure that they have sufficiently small solutions.

Lemma III (Binomial coefficients; cf. Lemma 4.3 of [3]). There exists a system $\mathcal{B}(z,n,k)$ of equations among polynomials involving exponentiation such that for all $z, n, k \in \omega$ with $n \geq k$:

(i) $z = \binom{n}{k}$ ⇒ There exists a solution to $B(z,n,k)$ in natural numbers less than 2^{2n^2+4} with values of exponentiations less than $(2^n+2)^n$

(ii) $\mathcal{B}(z,n,k)$ has any natural-number solution ⇒ $z = \binom{n}{k}$.

Replacing all binomial coefficients in the system *S'* by systems $\mathcal{B}(z,n,k)$, we obtain a system *S"* involving equations among polynomials with exponentiation. Exponentiation will now be eliminated in favor of the function $\psi_A(n)$, $A > 1$:

$\psi_A(n)$ = the n^th solution for y by magnitude of the Pell equation $x^2 - (A^2-1)y^2 = 1$.

This definition is meaningful because the equation always has infinitely many natural-number solutions.[3]

Lemma IV (Exponentiation; cf. lemma on p. 535 of [8]). There exists a system $\mathcal{E}(y,x,n)$ of equations among polynomials involving the function $\psi_A(n)$ such that for all $y, x, n \in \omega$:

(i) $y = x^n \Rightarrow$ There exists a solution to $E(y,x,n)$ in natural numbers less than $xyn\cdot[(8nx)(y+1) + x^2 + 2x]^{2n}$ where the largest ψ-function occurring is $\psi_{(4nx\cdot(y+1))+x^2+2x}(n+1)$.

(ii) $E(y,x,n)$ has any solution in natural numbers $\Rightarrow y = x^n$.

Replacing all exponentiations in the system S'' by systems $E(y,x,n)$, we obtain a system S''' involving equations among polynomials with ψ-functions. The definition of the function $\psi_A(n)$ by a system of equations between polynomials with integer coefficients was the crucial step in Matijasevic's[7] solution to Hilbert's 10th problem.

Lemma V (Strong definability of solutions to Pell equations; cf. Theorem 3.1 of [3]). There exists a system $\hat{P}(a,y,k)$ of equations among polynomials with integer coefficients such that for all $a, y, k \in \omega$, $a > 1$:

(i) $y = \psi_a(k) \Rightarrow$ There exists a solution to $\hat{P}(a,y,k)$ in natural numbers less than $2^{(3a)^{(2k)}}$.

(ii) $\hat{P}(a,y,k)$ has a solution in natural numbers $\Rightarrow y = \psi_a(k)$.

Replacing all ψ-functions in the system S''' by systems $\hat{P}(a,y,k)$, we obtain a system $\hat{S}''''$ involving only equations among polynomials with integer coefficients, and conditions of the form $a \equiv b \bmod c$. The latter can be replaced by equations: $a-b = cx$ (where $a > b$). By Lemma 1 of [2] (namely $p = 0$ & $q = 0 \Leftrightarrow p^2 + q^2 = 0$ for polynomials p and q) all our equations can be combined into a single equation $\hat{P}_M = 0$, where $\hat{P}_M$ is a polynomial with integer coefficients; $\hat{P}_M$ contains the original unknowns

$$x, T, I_0, I_b, I_{\sigma_0}, \ldots, I_{\sigma_z}$$

of the system S, and all those we have accumulated in the process of transforming S to $\hat{P}_M$; in short we will write

$$\hat{P}_M = \hat{P}_M(x,T,\vec{u}) .$$

From the previous lemmas, one now obtains:

Lemma VI (Fundamental Theorem, weak form). For every nondeterministic Turing machine M, and any $x, t \in \omega$, $t > 1$:

(i) M halts on input x in exactly t steps $\Rightarrow$

(a) There exists natural numbers $T, \vec{u}$ less than $2^{2^{2^{10t^2}}}$ such that $\hat{P}_M(x,T,\vec{u}) = 0$.

(b) There exists natural numbers $\vec{u}$ less than $2^{2^{2^{10t^2}}}$ such that $\hat{P}_M(x,t,\vec{u}) = 0$.

(c) For all natural numbers $T, \vec{u}$,

$$\hat{P}_M(x,T,\vec{u}) = 0 \Rightarrow T \geq t .$$

(ii) If $\hat{P}_M$ has a solution (given x) in natural numbers, then M halts on input x.

The weak form of the fundamental theorem (Lemma VI) differs from the desired final version by the size of the bound on the solutions to '$\hat{P}_M = 0$': We will obtain a bound of $2^{2^{10t^2}}$, i.e. an improvement of one level of exponentiation. This will be obtained as follows:

We replace the definition of the ψ-functions (the system $\hat{P}$ of Lemma V) by a weaker set of conditions (in Lemma VII below). These conditions will define the ψ-functions with a smaller bound on the size of solutions, and we will then obtain the improved result as above. The conditions, however, have a defect: If numbers larger than a certain upper bound are used in a solution then unpredictable things occur. Accordingly we are forced to use different equations (the machinery for which we have already set up) to obtain a "large" number to approximate this upper bound and be used to insure that nothing unpredictable occurs. This "large" number will be defined using our ability (Lemma VI) to define running times of Turing machines: to define the running time t of M we consider a deterministic machine $\hat{M}$ which runs in time $\hat{t}$ approximately one level of exponentiation faster than M: $\hat{t} \approx 2^t$. Because of this, we can afford to use the expensive (i.e. requiring large solutions) conditions of Lemmas V and VI to get a polynomial definition of the running-time $\hat{t}$ of $\hat{M}$; the "large" number which we are really after will then be defined in terms of $\hat{t}$ (Lemma IX below).

Lemma VII (Weak definability of solutions to Pell equations). For any $a, y, k, d \in \omega$, $k > 1$, $d > 1$, $a > k > 0$:

(i) $y = \psi_{ad}(k) \Rightarrow$ There exist $x, k \leq (2ad)^k$ satisfying the system P' below.

(ii) If the system P' has natural number solutions, x, k, then

either (a) $y = \psi_{ad}(k)$

or (b) $y \neq \psi_{ad}(k)$ and $x > (ad)^{ad} \geq d^d$.

$$P': \begin{cases} x^2 - (a^2d^2-1)y^2 = 1 \\ y - k = k(ad-1) \quad (\text{i.e. } y \equiv k \bmod ad-1) . \end{cases}$$

For all d of a certain magnitude, Lemma VII provides a way of distinguishing between "correct" solutions of P', where $y = \psi_{ad}(k)$, and "incorrect ones" where $y \neq \psi_{ad}(k)$; namely by the magnitude of the number x. By imposing a bound on the size of x by other unrelated conditions, we will eliminate case (ii)(b), and hence "incorrect" solutions.

Now let S'''' be the system of equations obtained from S''' by replacing all occurrences of ψ-functions by systems P' with a single 'd' common to all. Let $P_M(x,d,T,\vec{u})$ be the polynomial obtained by combining the conditions in the system S'''' (as was done above preceding the statement of Lemma VI).

Lemma VIII. For any nondeterministic Turing machine M and all $x, t, d \in \omega$, $t > 1$:

(i) $d \leq 2^{2^{t^2}}$ and M halts on input x within t steps $\Rightarrow$ There exist natural numbers $T, \vec{u}$ not larger than $2^{2^{10t^2}}$, such that $P_M(x,d,T,\vec{u}) = 0$.

(ii) $d \geq 2^{10t^2}$ and there exist natural-numbers T and $\vec{u}$ not larger than $2^{2^{10t^2}}$ such that $P_M(x,d,T,\vec{u}) = 0 \Rightarrow$ M halts on input x.

Lemma IX. There exists a system $T(t,d)$ of polynomial equations such that for any $t \in \omega$, $t \geq 1$:

(i) There exists a solution to $T(t,d)$ in natural numbers less than $2^{2^{2^{10t^2}}}$.

(ii) For any solution in natural numbers to T, $d \geq 2^{2^{2^t}}$.

Proof of Fundamental Theorem (Final argument). Let the set A be accepted by a NDTM M within super running time Φ. Without loss of generality, we assume that if $x \notin A$, then M never accepts on input x. By definition of super running time, there is a deterministic Turing machine $\hat{M}$ with total running time function $\hat{\Phi}$ such that for all $x \in \omega$:

$$2^{5\hat{\Phi}^2(x)+2} \leq 4\Phi(x) \leq 2^{2^{\hat{\Phi}(x)-1}} . \qquad (*)$$

Let P_A be the polynomial obtained from

$$\hat{P}_{\hat{M}}(x,T,\vec{u}),\ \mathcal{T}(T,d),\ P_M(x,d,T',\vec{u}')$$

by combination as above. Now for all $x \in \omega$

(1) assume $x \in A$; then

(a) $\hat{M}$ halts on input x in time $\hat{\Phi}(x)$. By Lemma VI(1)(b) $\hat{P}_{\hat{M}}$ has a solution with $T = \hat{\Phi}(x)$ and other numbers not exceeding

$$2^{2^{2^{10\hat{\Phi}^2(x)}}} = 2^{2(2^{5\Phi^2(x)})^2} \leq 2^{2^{\Phi^2(x)}} \quad \text{(by (*))};$$

(b) by (a) and Lemma IX(i) $\mathcal{T}$ has a solution with numbers not exceeding

$$2^{2^{2^{10T}}} = 2^{2^{2^{10\hat{\Phi}^2(x)}}} \leq 2^{2^{\Phi^2(x)}}$$

and with d such that

$$d \leq 2^{2^{2^{10\hat{\Phi}^2(x)}}} \leq 2^{2^{\Phi^2(x)}} ;$$

(c) M halts on input x within time $\Phi(x)$ so by (b) and Lemma VIII(i), P_M has a solution in numbers not exceeding $2^{2^{10\Phi^2(x)}}$;

(d) by (a), (b) and (c) P_A has a solution in numbers not exceeding $2^{2^{10\Phi^2(x)}}$;

(2) assume P_A has a solution in numbers not exceeding $2^{2^{10\Phi^2(x)}}$, then

(a) $\hat{P}_{\hat{M}}$ has a solution and by Lemma VI (i)(c), $T \geq \hat{\Phi}(x)$;

(b) $\mathcal{T}$ has a solution and by Lemma VIII (ii) and (a)

$$d \geq 2^{2^{2^{\hat{\Phi}(x)}}} \geq 2^{10\Phi^2(x)} \quad \text{(by (*))} ;$$

(c) by (b) together with Lemma VIII (ii) M halts on input x so $x \in A$.

Note that $\hat{P}_{\hat{M}}$ and $\mathcal{T}$ are used only to generate a number d in the range indicated by Lemma VIII. When Φ is a super running time of type I then such a number can be defined directly from x using simple polynomials and $\hat{P}_{\hat{M}}$ and $\mathcal{T}$ are not needed. □

This completes the argument for the Fundamental Theorem. For the proofs of the consequences of the Fundamental Theorem (Theorems 3.1-3.5) we refer to [2], especially Lemma 3.

To show Theorem 4.1, note that the system S above involves only disjointness conditions and polynomial equations with exponentiation. The exponentiation can be eliminated in favor of any equation between polynomials which allows solutions of size polynomial in the size of the exponentiations eliminated (see [2], Theorem 7 or Lemmas IV and VII above). The disjointness conditions can be eliminated using:

Lemma. For any $m, n \in \omega$: m and n are disjoint iff

$$\begin{aligned} m'm''n'n''[&n' \in R_0 \ \&\ 2m'' \in R_0 \ \&\ n' \in R_0 \ \&\ 2n'' \in R_0 \\ &\&\ m'+m'' = m \ \&\ n'+n'' = n \ \&\ m'+n' \in R_0 \\ &\&\ 2(m''+n'') \in R_0] . \end{aligned}$$

Thus the system of conditions S can be replaced by one entirely in terms of the predicate '$x \in R_0$' and polynomial equations, which can be combined into a single equation as before. By Lemma I, this eventually yields the D-completeness of R_0 and the exact normal form theorem for nondeterministic Turing machines.

VI. Open Problems

1) Let Pr denote the set of prime numbers. By Pratt[10], Pr $\in$ NP. Is Pr $\in$ D?

2) Is Pr D-complete?

3) Let C denote the set of composite numbers. It is easily seen that $C \in D^2$. Prove $C \notin D^1$. Can this be generalized to handle 4)?

4) Prove that for all i, D^i is properly contained in D^{i+1}.

5) What is the relationship of $D \cap D^c$ to P?

6) Under what assumptions short of $P = D$ would $Th(<D,\leq_D>)$ be equivalent to $Th(<P,\leq_P>)$?

7) If $D \neq NP$, are there D-degrees between 0 (the degree of sets in D) and the degree of R_0 (see Ladner[5])?

8) Are the regular sets of star height 1 not covered in Theorem 3.2 all D-equivalent to R_0?

9) NP = D?

10) (A different subdivision of D) For $k \geq 1$, let $D(k)$ be the set of all numerical relations definable by a formula of the form

$$<x_1 \cdots x_m> \in R$$

$$\Leftrightarrow \exists y_1 \cdots y_n \leq 2^{c(|x_1+\cdots+x_m|)^k} : P(x_1 \cdots x_m, y_1 \cdots y_n) = 0$$

where $c > 0$ and P is a polynomial. If $k > \ell$, then $D(\ell) \subseteq D(k)$; $D(1)$ is just the class where the definition can be chosen as

$$\exists y_1 \cdots y_n \leq q(x_1 \cdots x_m): P(x_1 \cdots x_m, y_1 \cdots y_n) = 0 ,$$

p, q polynomials.

The relation '$x = y$' is in $D(2)$. Is it in $D(1)$?

11) Are any of the inclusions $D(\ell) \subseteq D(k)$, for $k > \ell$, strict? An affirmative answer would follow from D = NP, by use of diagonalization over nondeterministic Turing machines running in time n^k. But this question could be independent of 'NP = D'.

References

[1] Adleman, L., "Number Theoretic Aspects of Computational Complexity." Ph.D. Thesis, U.C. Berkeley, 1976.

[2] Adleman, L. and Manders, K., "The Computational Complexity of Decision Procedures for Polynomials," *16th Annual IEEE Symp. on Foundations of Computer Science (1975)*.

[3] Davis, M., "Hilbert's Tenth Problem is Unsolvable," *Amer. Math. Monthly 80* (1973), pp. 233-269.

[4] Davis, M., Putnam, H. and Robinson, J., "The Decision Problem for Exponential Diophantine Equations," *Annals of Math. 74* (1961), pp. 425-436.

[5] Ladner, R., "On the Structure of Polynomial Time Reducibility," *J. ACM 22*, 1 (Jan. 1975).

[6] Manders, K. and Adleman, L., "NP-complete Decision Problems for Quadratic Polynomials," *Proc. 1976 8th Annual ACM Symp. on Theory of Computing*, pp. 23-29.

[7] Matijasevic, Y., "Enumerable Sets Are Diophantine (Russian)," *Dokl. Akad. Nauk SSSR 191* (1970), pp. 279-282.

[8] Matijasevic, Y. and Robinson, J., "Reduction of an Arbitrary Diophantine Equation to One in 13 Unknowns," Acta Arithmetica 27 (1975), pp. 521-553.

[9] Matijasevic, Y., "A New Proof of the Theorem on Exponential Diophantine Representation of Recursively Enumerable Predicates," Zapiski of The Mathematical Institute of Lenningrade Devision of Academy of Sciences USSR, June 1976.

[10] Pratt, V., "Succinct Certificates for Primes," to appear.

[11] Stockmeyer, L. and Meyer, A., "Word Problems Requiring Exponential Time," Proc. 5th Annual ACM Symp. on Theory of Computing (April 1973), pp. 1-9.