COMPUTATIONAL COMPLEXITY OF DECISION PROCEDURES FOR POLYNOMIALS[†]

LEONARD ADLEMAN and KENNETH MANDERS

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

(Extended Abstract)

In this article we present results of our research on the computational complexity of decision procedures for polynomials. The work combines techniques and results from number theory and computational complexity. A number of connections are established between the fields, allowing results in one to be applied in the other. The work suggests numerous research problems of considerable interest.

In section I we present theorems which give, among other things, nontrivial lower and upper complexity bounds for decision procedures concerning the existence of bounded natural number solutions to polynomials. The proofs (section III) incorporate essentially all the number theoretic techniques involved in the recent (1970) solution to Hilbert's 10th problem: Does there exist an effective procedure for deciding whether a diophantine equation (i.e. a multivariable polynomial with integer coefficients) has a solution in the integers?

Section II contains motivational material, open problems and additional results.

I(a) Computational Complexity of Polynomial Decision Problems

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 iff $a_1,\ldots,a_n \in \omega$ and $P(a_1,\ldots,a_n) = 0$. E will be a fixed binary encoding of all polynomials with integer coefficients (E will be specified in section II). Where the context permits we will write P instead of E(P). For any $x \in \omega$, $|x|$ will denote the length of x in binary. Let g be a unary function, then:

$$S_g = \{P(x_1,\ldots,x_n) \mid (\exists a_1,\ldots,a_n \in \omega)[P(a_1,\ldots,a_n) = 0] \text{ and } a_i \leq g(E(P)),\ i = 1,2,\ldots,n\}$$

C_g will denote the problem of accepting from input E(P) those $P \in S_g$. For any functions f and g we say C_g infinitely often requires f steps on a nondeterministic multitape Turing machine iff for any nondeterministic multitape Turing machine M which accepts C_g, there is a rational c, and there are infinitely many $P \in S_g$ such that the number of steps taken by M on input E(P) is greater than cf(E(P)).

Theorem 1. For all $c \in \omega$, $c \neq 0$:

a) Let $g(x) = c$, then C_g is NP-complete.

b) Let $g(x) = |x|^c$, then C_g is NP-complete.

c) Let $g(x) = 2^{|x|^c}$, then C_g is NP-complete.

Theorem 2. For all $c \in \omega$, $c > 1$, let $g(x) = 2^{2^{2^{|x|^{5\cdot c}}}}$. Then:

(i) C_g infinitely often requires $|P|^c$ steps on a nondeterministic multitape Turing machine.

(ii) There is a one-tape nondeterministic Turing machine with running time within $2^{2^{|x|^{5\cdot c+1}}}$ which accepts S_g.

Theorem 3. Let $g(x) = 2^{2^{2^{2^{5|x|}}}}$. Then

(i) C_g infinitely often requires $2^{|P|}$ steps on a nondeterministic multitape Turing machine.

(ii) There is a one-tape nondeterministic Turing machine with running time within $2^{2^{2^{5|x|+1}}}$ which accepts S_g.

Definition. A function f: ω → ω is a super running time iff:

(i) f is total.

(ii) f is the running time of some deterministic Turing machine.

(iii) f is monotonic increasing.

(iv) $(\overset{\infty}{\forall} y)[f(y) > 2^y]$

(v) There is a total deterministic running time function Φ such that

$$f(x) \leq 2^{2^{\Phi(x)}} \leq 2f(x) .$$

The following is a technical theorem, the main content of which is expressed in its corollary Theorem 5.

Theorem 4. For all super running times Φ_i, let $g(x) = 2^{2^{2^{\Phi_i(m(x))^5}}}$. Then:

(i) C_g infinitely often requires $\Phi_i(m(P)/4)$ steps on a nondeterministic multitape Turing machine.

(ii) There is a one-tape nondeterministic Turing machine with running time within $2^{2^{\Phi_i(m(x))^5+1}}$ which accepts S_g where m is a function dependent on i such that: $(\overset{\infty}{\forall} x \in \omega)[2^{\sqrt{\log(x)/c}} \geq m(x) \geq 2^{\sqrt{\log(x)/c+1}}]$ for some constant c independent of i (i.e. m is neither arbitrarily large nor arbitrarily small).

Theorem 5.[1] (Polynomial Compression Theorem) For all recursive functions f, there is a recursive function h such that

A) $(\forall x)[h(x) > f(x)]$ and

B) If $g(x) = 2^{2^{2^{h(x^2)^5}}}$ then:

(i) C_g infinitely often requires h(p) steps on a nondeterministic multitape Turing machine.

(ii) There is a one-tape nondeterministic Turing machine with running time within $2^{2^{2^{h(x^2)^5+1}}}$ which accepts S_g.

The proofs in section III actually support somewhat stronger versions of Theorems 2, 3, 4 and 5: The class of polynomial considered can be restricted to

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

[1] There are many variations of Theorems 4 and 5 just as there are many variations of the abstract compression theorem. The versions presented here were selected for clarity rather than generality. In particular, similar results hold with much weaker notions of super running time, or when only deterministic Turing machines are considered.

those with fewer than 114 variables.

Other than the results of Meyer, Stockmeyer [11, 18], Rabin, Fisher, Strassen and Solovay concerning regular expressions and logical theories, we know of no other concrete problems with provably non-trivial lower bounds. In addition the Polynomial Compression Theorem yields problems of arbitrary difficulty. Thus we finally have a concrete example for the analogous result in abstract complexity theory ([2]).

The Polynomial Compression Theorem is of distinct importance to computational number theory: It shows that there can be no way to dramatically improve the technique of exhaustive search as a general method for finding bounded solutions to polynomials.

II(a)

Given a polynomial $P(x_1 \ldots x_n)$ and a function f, we say a set $A \subseteq \omega$ is defined by P within bound f if and only if

$$x \in A \Leftrightarrow (\exists y_2 \ldots y_n \leq f(x))[P(x,y_2 \ldots y_n) = 0] .$$

The proof of Theorem 4 in section III indicates an effective procedure for taking an algorithm ϕ recognizing a set A and producing a polynomial $P(x_1 \ldots x_{100})$ and an f such that A is defined by P within bound f. The bounding function f can be seen to be a function of the running time of the algorithm ϕ. Conversely, given a polynomial P and a bounding function f which in the above sense define a set A, then it is straightforward to construct an algorithm ϕ which recognizes A and whose running time is a function of f. Thus good bounds on the size of solutions of polynomials found in number theory are convertible into fast algorithms and vice-versa. For example, consider the prime numbers. Polynomials defining the primes are of obvious interest to number theorists. Unfortunately, current defining polynomials are cumbersome in the sense that they require solutions of magnitude approximately $2^{2^{2^{2^{|x|^c}}}}$ for some constant $c \in \omega$ [8]. By converting the remarkably fast algorithms of Miller [12] or Pratt [14], we are able to produce the following improvement:

Theorem 6. There is a polynomial $p(x_1 \ldots x_{100})$ and a $c \in \omega$ such that:

$$x \in \text{Primes} \leftrightarrow \exists y_2 \ldots y_{100} \leq 2^{2^{2^{|x|^c}}} \quad [p(x,y_2 \ldots y_{100}) = 0] .$$

Open Problem 1. Does there exist a polynomial $p(x_1,\ldots,x_n)$ and a $c \in \omega$ such that:

$$x \in \text{Primes} \leftrightarrow (\exists y_2 \ldots y_n \leq 2^{|x|^c})[P(x,y_2,\ldots,y_n) = 0] \ ?$$

Open Problem 2. Does there exist a polynomial in fewer than 11 variables which defines the primes within some bound?

Open Problem 3. Does there exist a polynomial $p(x_1,\ldots,x_n)$ and a $c \in \omega$ such that:

$$<\ell,m,k> \ \in \{<x,y,z> | x = \tbinom{y}{z}\} \Leftrightarrow (\exists y_4 \ldots y_n \leq 2^{|\ell|^c}) \\ [P(\ell,m,k,y_4,\ldots,y_n) = 0] \ ?$$

Open Problem 3 is of particular interest since a positive answer would improve the lower bounds in this paper by one level of exponentiation. Our new results show that such a definition is possible for exponentiation:

Theorem 7. There is a polynomial $p(x_1,\ldots,x_6)$ and a $c \in \omega$ such that:

$$<\ell,m,k> \ \in \{<x,y,z> | x = y^z\} \Leftrightarrow (\exists y_3,y_4,y_5 \leq 2^{|\ell|^c}) \\ [P(\ell,m,k,y_4,y_5,y_6) = 0] .$$

It has long been known (Gödel, 1931) that exponentiation was first-order definable in $\text{Th}(<\omega,+,\cdot>)$, and more recently it was proved (Matijesevic, 1970) that it was definable by an existential predicate. Theorem 7 gives an optimal bounded definition.

II(b)

Our results have consequences in which might be termed "the bounded 1st order theory of the natural numbers with plus and times": Select a class of functions F. Then the formulas of the theory are just those of the usual 1st order theory except that all quantifiers are bounded by some function $f \in F$ applied to a free variable. In particular let $D = \{f | f(x) = 2^{p(|x|)}$ for some polynomial p with positive coefficients$\}$. Then define D^1 to be the set of all numerical relations S such that S is definable by a predicate with exactly one existentially quantified variable, bounded by a function in D applied to a free variable of the predicate.

Example. Let S_2 be the set of perfect squares. Then $S_2 \in D^1$ since $x \in S_2 \Leftrightarrow (\exists y \leq 2^{|x|})[y \cdot y = x]$.

Example. For all n let $S_n = \{x | (\exists y)[y^n = x]\}$. Then $S_n \in D^1$. For all m, n let $S_{m,n} = \{x | x = m \bmod(n)\}$. Then $S_{m,n} \in D^1$.

Define D^2 to be the set of all numerical relations S such that S is definable by a predicate with exactly two existentially quantified variables, each bounded by a function in D applied to a free variable.

Example. Let C be the set of composite numbers, then $C \in D^2$ since $x \in C \Leftrightarrow (\exists y_1,y_2 \leq 2^{|x|})[(y_1+2)(y_2+2) - x = 0]$.

Define D^i analogously.[2] Let $\mathcal{D} = \bigcup_{i \in \omega} D^i$. We can now define a Kleene type hierarchy in an obvious way letting $\Sigma_1^{\mathcal{D}} = \mathcal{D}$, $\Pi_1^{\mathcal{D}} = \{S | \text{complement of } S \text{ is in } \mathcal{D}\}$, etc. Let $\mathbb{D} = \bigcup_{i \in \omega} \Sigma_i^{\mathcal{D}}$.

Definitions. PTIME is the set of numerical relations A: A is decidable in deterministic polynomial time. NPTIME is the set of numerical relations A: A is accepted in nondeterministic polynomial time. For all polynomials q and for all $i \in \omega$

$$R_{i,q} = \{p | p \text{ has } i \text{ variables and } \exists a_1,\ldots,a_i \leq 2^{2(|p|)}: \\ p(a_1,\ldots,a_i) = 0\} .$$

Proposition 1. For all q, $R_{1,q} \in$ PTIME.

This follows by any of various methods, for example by a bisection technique based on Sturm's Theorem. (Note that we are identifying polynomials with their Gödel numbers.)

Theorem 8. $D^1 \subseteq$ PTIME.

Proof. Let $A \in D^1$ and $A \subseteq \omega$ (if A is a set of n-tuples the proof is similar). Then for some polynomial q:

$$x \in A \Leftrightarrow (\exists y \leq 2^{q(|x|)})[H(x,y)] ,$$

where H(x,y) is a quantifier-free predicate in the language of $<\omega,+,\cdot>$. It is not difficult (see section III) to transform H into a polynomial p such that

$$x \in A \Leftrightarrow (\exists y \leq 2^{q(|x|)})[p(x,y) = 0] .$$

From this it follows that there is a q' such that:

$$x \in A \Leftrightarrow (\exists y \leq 2^{q'(|p|)})[p(x,y) = 0] .$$

That is, $x \in A \Leftrightarrow P(x,y) \in R_{1,q'}$. Now the theorem follows from Proposition 1.

[2] By Theorem 7, exponentiation is in D^3.

We find this result particularly interesting since it allows us to conclude on purely syntactical grounds that a set is decidable in deterministic polynomial time.

Open Problem 4. $D^2 \subseteq$ PTIME?

This could be proved by showing that Proposition 1 can be extended to $R_{2,q}$. Since clearly $D^2 \subseteq$ NPTIME, a negative answer implies PTIME $\neq$ NPTIME, though the converse does not necessarily hold. A positive answer implies that the primes are in PTIME (without assuming the Extended Riemann Hypothesis [12]).

Open Problem 5. $\mathcal{D} \subseteq$ PTIME?

Open Problem 6. $R \subseteq \mathcal{D}$, where R is the set of regular sets? Many intuitive "arithmetic" sets are in $\mathcal{D}$ but showing that syntactically defined ones are is difficult. Note that Theorem 7 can be used to define a substring predicate.

Open Problem 7. PTIME $\subseteq \mathcal{D}$?

It follows from the syntactical characterization of NP given in [18] and the fact that $\mathcal{D} \subseteq$ NPTIME that this question is equivalent to NPTIME = $\mathcal{D}$?

Open Problem 8. Give a computational characterization of $\mathcal{D}$. We hope that a set with such a natural definition has a natural computational analogue.

Open Problem 9. Is $\mathcal{D}$ closed under complement?

This has consequences relevant to the open problems in section II(a).

Open Problem 10. Find a set $A \in \mathcal{D}$ such that for all $B \subseteq \omega$, if $B \in \mathcal{D}$, then B is polynomial reducible to A (i.e. show that A is $\mathcal{D}$-complete).

Open Problem 11. Are there an i and a q such that $R_{i,q}$ is NP-complete? Theorem 2 shows that $\bigcup_{i\in\omega} R_{i,q}$ is NP-complete.

Let $F = \{f \mid \text{for some } n \in \omega,\ f(x) = \underbrace{2^{2^{\cdot^{\cdot^{\cdot^{2^x}}}}}}_{n}\}$. Then define F^i, $\mathcal{F}$, $\Sigma_i^{\mathcal{F}}$, $\mathbb{F}$ as with D above. The following new theorem on hierarchies follows from our results:

Theorem 9. $\mathcal{F} = \mathbb{F} = \varepsilon^2$ where ε^2 is the third level of the Grzegorczyk hierarchy [7] (i.e., the elementary computable functions).

It is worth noting that this result is proved using our new results in section III and is entirely independent of the results of Matijesevic (though it relies heavily on the work of Robinson, Putnam and Davis done by the early 60's). This also gives an example of a collapsing computational hierarchy.

Open Problem 12. Compare the hierarchies indicated here with others -- in particular the Grzegorczyk [7], Meyer-Stockmeyer [18] and those found in [3].

II(c) A Polynomial Model of Computation

Let $\{P_i\}_{i\in\omega}$ be the enumeration of polynomials by size induced by our Gödel numbering E. We can interpret $\{P_i\}_{i\in\omega}$ as an enumeration of programs for the partial recursive functions as follows:

With $P_i(x_1 \ldots x_k)$ identify the following algorithm:

"on input a go to stage 0

Stage n: check whether $p_i(a, a_2, \ldots, a_k) = 0$ for any combination of $a_2, \ldots, a_k \leq n$.

If any solutions are found, output the least a_2 occurring in any of them and halt.

Otherwise go to stage n+1."

It can be shown that under this interpretation $\{P_i\}_{i\in\omega}$ is an acceptable Gödel numbering of the partial recursive functions in the sense of Rogers [16]. Further, stepcounting functions for $\{P_i\}_{i\in\omega}$ are given by the following algorithm:

Q_i = "on input a go to stage 0

Stage n: check whether $p_i(a, a_2 \ldots a_k) = 0$ for any combination of $a_2, \ldots, a_k \leq n$.

If yes output n and halt.

Otherwise go to stage n+1."

Thus Q_i computes f_i where $f_i(a)$ = the least z such that:

$$(\exists a_2 \ldots a_k \leq z)[P(a, a_2 \ldots a_k) = 0] .$$

By the methods in the proof of Theorem 4, there exists a polynomial $\text{Ᵽ}_i(x_1, \ldots, x_\nu)$ effective in the procedure Q_i, such that for any a, n:

$$f_i(a) = n \Leftrightarrow (\exists a_3, \ldots, a_\nu \in \omega)[\text{Ᵽ}_i(a, n, a_3 \ldots a_\nu) = 0] .$$

It can be shown that $\{\text{Ᵽ}_i\}_{i\in\omega}$ is a sequence of stepcounting functions for $\{P_i\}_{i\in\omega}$ satisfying the Blum axioms [2]. These two facts have many startling consequences. Because the Gödel numbering $\{P_i\}_{i\in\omega}$ is acceptable, the recursion theorem holds and yields a fixed point theorem for polynomials. For any $P_i(x_1 \ldots x_n)$ let $S_{p_i} = \{a \in \omega \mid p(a, 1, x_3 \ldots x_n) \text{ has a solution}\}$. Then

Theorem 10. For any recursive f there exists an index $i \in \omega$:

$$S_{p_i} = S_{p_{f(i)}} .$$

For example: Let f be the function computed by the following algorithm which performs a "syntactical" manipulation on polynomials:

"On input $p(x_1 \ldots x_n)$, output

$500(p(x_1^2, x_2^4, \ldots, x_{n-1}^{2^{n-1}}, 6x_n^{2^n})) + 7$"

then for some i, $S_{p_i} = S_{p_{f(i)}}$

Since $\{\text{Ᵽ}_i\}_{i\in\omega}$ is a sequence of stepcounting functions, the speed-up theorem (this was first observed by David [5]), the compression theorem, the gap theorem, and so on, all hold for polynomials. As all of these theorems give information above the size of solutions of polynomials, they have number-theoretic interest. It also appears doubtful that direct number-theoretic proofs could be obtained.

II(d)

The Fields Medal winning work by Alan Baker provides a good example of how problems involving bounded solutions to polynomials arise naturally in mathematics. In a recent paper [1] he settles a famous conjecture due to Gauss. The result can be put in the following form:

There is a polynomial P (which can be given explicitly) such that the following are equivalent:

(i) There are exactly 10 (rather than 9) complex quadratic fields of class number one.

(ii) P has a solution in which all variables are less than $2^{2^{2^{2^P}}}$.

(iii) $P \in S_g$ where $g(x) = 2^{2^{2^{2^x}}}$.

III. Outline of Proofs

III(a) Preliminaries

Theorems 1a, 2, 3, 4 and 5 hold for any reasonable encoding of polynomials. However, Theorems 1b and 1c require an encoding which leaves C_g in NP. This can be accomplished by any natural notation for polynomials provided we write powers as products, for instance $x_1^3x_2^5$ as "$x_1x_1x_1x_2x_2x_2x_2x_2$". With this encoding, the simple technique: "guess a bounded solution set, evaluate the input polynomial on this set, accept iff the value is 0" works in nondeterministic polynomial time. However, if an encoding which represents powers in exponential notation would be chosen, then there would be no obvious way to show C_g is in NP. For example the simple technique on inputs like "exp(x,500) = 0" (for $x_1^{500} = 0$) can cause numbers on the order of 500^{500} to be written.

Throughout the proofs, $\{\phi_i\}_{i\in\omega}$ will denote a fixed enumeration of nondeterministic semi-infinite Turing machines. When the context permits, we will use ϕ to denote any of: the i^{th} Turing machines; the function computed by the i^{th} Turing machine; the algorithm used by the i^{th} Turing machine.

For any $i \in \omega$, Φ_i will denote the running time of ϕ_i. If ϕ_i halts on input x, we write "$\phi_i(x)\downarrow$", otherwise "$\phi_i(x)\uparrow$".

The proofs of Theorems 2, 3 and 5 closely follow the proof of Theorem 4. Thus we will outline only the proofs of Theorems 1 and 4. First we note several simple facts:

Lemma 1.

(i) $\forall x_1\ldots x_n$: $[(P_1(x_1\ldots x_n) = 0 \text{ and } P_2(x_1\ldots x_n) = 0)$
$\Leftrightarrow P_1^2(x_1\ldots x_n) + P_2^2(x_1\ldots x_n) = 0]$

(ii) $\forall x_1\ldots x_n$: $[((p_1(x_1\ldots x_n) = 0 \text{ or } p_2(x_1\ldots x_n) = 0)$
$\Leftrightarrow P_1(x_1\ldots x_n) + P_2(x_1\ldots x_n) = 0]$

(iii) z is the remainder when x is divided by y (notation: $z = \text{Rem}(x,y)) \Leftrightarrow \exists w_1,w_2[yw_1+z = x$ and $z+1+w_2 = y]$. If the parameter z is a constant known to be less than y, for example $z = 0$, then of course the second clause can be omitted.

(iv) $\forall w_1\ldots w_n \in \omega$, $\forall z_1\ldots z_n \in \omega$: If the w_i's are relatively prime in pairs, then

$$w_1|z_1 \,\&\, w_2|z_2 \,\&\, \cdots \,\&\, w_n|z_n$$
$$\Leftrightarrow w_1w_2w_3\ldots w_n|w_2w_3\ldots w_nz_1 + w_1w_3\ldots w_nz_2 + w_1\ldots w_{n-1}z_n$$

(v) $\forall a,b,p,q \in \omega$:

$$a|b \,\&\, p \geq q \Leftrightarrow \exists z \in \omega\text{: } b - ax - a(b+1)(q-p) = 0$$

III(b) Outline of Proof of Theorem 1

Let S = $\{\phi|\phi$ is a satisfiable propositional formula in conjunctive form with at most three variables per clause$\}$. By Cook [4] it suffices to show (since C_g is clearly in NP) that $S \leq_m^p S_g$ (i.e., there is a recursive function f such that f runs in polynomial time and $\phi \in S \Leftrightarrow f(\phi) \in S_g$). Let $\phi = C_1 \,\&\, C_2 \,\&\, \cdots \,\&\, C_n$ where the C_i's are disjunctive clauses involving propositional variables denoted by X_i^1, X_i^2, X_i^3 (where for $i \neq j$ it can hold that X_i^n and X_j^m denote the same variables). Let ψ be a sentence in the first-order language of $<\omega,+,\cdot>$ such that $\psi = \exists x_1^1x_1^2x_1^3x_2^1x_2^2\cdots x_n^3[[[x_1^1 = 0 \vee x_1^1-1 = 0] \,\&\, [x_1^2 = 0 \vee x_1^2-1 = 0] \,\&\, \cdots \,\&\, [x_n^3 = 0 \vee x_n^3-1 = 0]] \,\&\, [P_{1112} \,\&\, P_{1113} \,\&\, P_{1123} \,\&\, P_{1211} \,\&\, \cdots \,\&\, P_{nn23}] \,\&\, [P_1 \,\&\, P_2 \,\&\, \cdots \cdots \,\&\, P_n]]$ where

$$P_{ijmn} = \begin{cases} x_i^m - x_j^n = 0 & \text{if } X_i^m \text{ denotes the same propositional variable as } X_j^n \\ 0 = 0 & \text{otherwise} \end{cases}$$

and P_i is $(x_i^1-(0)1)(x_i^2-(0)1)(x_i^3-(0)1) = 0$ iff C_i is $((\neg)X_i^1$ or $(\neg)X_i^2$ or $(\neg)X_i^3)$. Now, using (i) and (ii) of Lemma 1, we can rewrite ψ as $(\exists x_1^1x_1^2\ldots x_n^3)[P_\phi = 0]$. It is clear that $P_\phi \in S_g \Leftrightarrow \phi \in S$ and that our transformation can be done in polynomial time.

III(c) Proof of Theorem 4

Theorem 4 is derived from the Compression Theorem of abstract complexity theory. Different forms of this theorem yield different results; the form we use for Theorem 4 is a variant of the version due to Sieferas, Fisher, Meyer [17].

Lemma 2 (Abstract Compression Theorem). Let Φ_i be a super running time. There is an infinite set $A \subseteq \omega$ such that:

(i) $(\forall k \in \omega)$ ϕ_k (nondeterministically) accepts a finite variant of A
$\Rightarrow \exists c \in \omega \overset{\infty}{\exists} x \in \omega$: $\Phi_k(x) \geq c\cdot\Phi_i(\frac{x}{4})$ & $\phi_k(x)\downarrow$

(ii) $(\exists j \in \omega)$ ϕ_j (nondeterministically) accepts A &
$\overset{\infty}{\forall} x \in \omega$: $\Phi_j(x) = \Phi_i(x)$

Whenever functions Φ_i and ϕ_j are related as in Lemma 2, we will say that ϕ_j is compressed for Φ_i.

The crucial step in the proof of Theorem 4 is the demonstration that for any machine ϕ_j and any input x, we can in polynomial time obtain a small polynomial P_{jx} which has small solutions if and only if $\phi_j(x)$ halts. Now if ϕ_j accepts a set A and is compressed for Φ_i, and ϕ is a nondeterministic algorithm accepting any set of polynomials containing exactly P_{jx} for which $\phi_j(x)$ halts, then from ϕ we can derive a new algorithm which accepts A and has running time approximately Φ. It follows that any such ϕ must take approximately Φ_i steps. The conversion of algorithms to polynomials is described by a lemma, which is proved in the next section.

Lemma 3 (Translation Lemma). There is an effective procedure h such that:

(i) $\forall i,j,x \in \omega$: h(i,j,x) is a polynomial with integer coefficients and 114 variables.

(ii) If Φ_i is a deterministic total running-time function and Φ_j is a nondeterministic running time function and

$$\overset{\infty}{\forall} x \in \omega[\phi_j(x)\downarrow \Rightarrow \Phi_j(x) \leq 2^{2^{\Phi_i(x)}} \leq 2\Phi_j(x)]$$

and $\overset{\infty}{\forall} x \in \omega$: $\Phi_i(x) \geq \log\log x$

then

(a) $\overset{\infty}{\forall} x \in \omega$, $\phi_j(x)\downarrow \Rightarrow \exists z_1,\ldots,z_{114} \leq 2^{2^{2^{(\Phi_j(x))^5}}}$:
$h(i,j,x)(z_1,\ldots,z_{114}) = 0$

(b) $\overset{\infty}{\forall} x \in \omega$, $\phi_j(x)\uparrow \Rightarrow \forall z_1,\ldots,z_{114} \leq 2^{2^{2^{2^{2^{2^{\Phi_i(y)}}}}}}$:
$h(i,j,x)(z_1,\ldots,z_{114}) \neq 0$

(iii) $\forall i,j \in \omega$: $\overset{\infty}{\forall} x \in \omega$:

$$c(\log x)^2 \leq |h(i,j,x)| \leq (c+1)(\log x)^2$$

where c is a constant independent of i, j and x.

(iv) $\forall i,j,x \in \omega$: $H(i,j,x) \leq d|h(i,j,x)|$, where H(i,j,x) is the running time of h on input <i,j,x> and d is a constant independent of i, j and x.

Proof of Theorem 4 from the Translation Lemma: Let Φ_i be a super running time with respect to Φ_k for some $k \in \omega$. Let ϕ_j be a compressed algorithm for Φ_i. We apply the translation lemma and consider

$$\{h(k,j,x)\text{: } x \in \omega\} .$$

We will first define the function m referred to in the statement of Theorem 4. It follows from Lemma 3(iii) that:

$$\overset{\infty}{\forall} x \in \omega: \quad 2^{c\cdot\log^2 x} \leq h(k,j,x) \leq 2^{(c+1)\log^2 x}$$

where $h(k,j,x)$ is the Gödel number of the output of h; setting $p = h(k,j,x)$, we obtain:

$$\overset{\infty}{\forall} x \in \omega: \quad x \leq 2^{\sqrt{\frac{\log p}{c}}} \quad ; \quad x \geq 2^{\sqrt{\frac{\log p}{c+1}}} \ .$$

Now we define m by:

$$\overset{\infty}{\forall} y \in \omega: m(y) = \begin{cases} x \text{ if } y = h(k,j,x) \text{ for some } x \\ \text{any } z \text{ such that } 2^{\sqrt{\frac{\log y}{c+1}}} \leq z \leq 2^{\sqrt{\frac{\log y}{c}}} , \\ \quad \text{otherwise.} \end{cases}$$

Thus we obtain the bounds on m given in Theorem 4:

$$\overset{\infty}{\forall} y \in \omega: \quad 2^{\sqrt{\frac{\log y}{c+1}}} \leq m(y) \leq 2^{\sqrt{\frac{\log y}{c}}} \ .$$

Now we can define g as:

$$\forall p \in \omega: \quad g(p) = 2^{2^{2^{[\Phi_i(m(p))]^5}}}$$

and S_g as

S_g = {p: p is the Gödel number of a polynomial in at most 114 variables with a solution bounded by g(p)}

and we show that

$$\overset{\infty}{\forall} x \in \omega: \quad \phi_j(x)\downarrow \Leftrightarrow h(k,j,x) \in S_g \ . \qquad (1)$$

By (ii) of the Abstract Compression Theorem

$$\overset{\infty}{\forall} x \in \omega: \quad \phi_j(x)\downarrow \Rightarrow \Phi_j(x) = \Phi_i(x)$$

and hence by (ii) of the Transformation Lemma

(a) $\overset{\infty}{\forall} x \in \omega$: $\phi_j(x)\downarrow \Rightarrow h(k,j,x)$ has a solution

$\leq 2^{2^{2^{(\Phi_i(x))^5}}}$ and by the definition of m

$\Rightarrow h(k,j,x) \in S_g$.

(b) $\overset{\infty}{\forall} x \in \omega$: $\phi_i(x)\uparrow \Rightarrow h(k,j,x)$ has no solution

$\leq 2^{2^{2^{2^{2^{\Phi_k(x)}}}}}$ so by the first inequality of (v) in the definition of super running times,

$$\overset{\infty}{\forall} x \in \omega: \phi_j(x)\uparrow \Rightarrow h(k,j,x) \text{ has no solution} \leq 2^{2^{2^{2^{\Phi_i(x)}}}}$$

and hence:

$$\Rightarrow h(k,j,x) \text{ has no solution} \leq 2^{2^{2^{(\Phi_i(x))^5}}}$$

$$\Rightarrow h(k,j,x) \notin S_g \ .$$

Because of (1), for any nondeterministic multitape algorithm ϕ with running-time function Φ which accepts S_g, the following is a nondeterministic multitape algorithm which accepts a finite variant of the set dom ϕ_j:

"On input x, find h(k,j,x). Apply ϕ to h(k,j,x). Accept x if ϕ accepts h(k,j,x)."

Clearly this algorithm, on input x such that $\phi_j(x)\downarrow$, will (for all but finitely many x) accept x in $H(x) + \Phi(h(k,j,x))$ steps; by (i) of the Abstract Compression Theorem.

$$\exists c \in \omega\ \overset{\infty}{\exists} x \in \omega: H(x) + \Phi(h(k,j,x)) \geq c\Phi_i(\tfrac{x}{4}) \ \&\ \phi_j(x)\downarrow$$

and thus

$$\exists c \in \omega\ \overset{\infty}{\exists} h(k,j,x) \in S_g: H(x) + \Phi(h(k,j,x)) \geq c\Phi_i(\tfrac{x}{4}) \ .$$

But $x = m(h(k,j,x))$ by definition of m, and $H(x) \leq d|h(k,j,x)|$ by (iv) of the Translation Lemma, so, setting $h(k,j,x) = p$,

$$\exists c \in \omega\ \overset{\infty}{\exists} p \in S_g: \Phi(p) \geq c\Phi_i(\tfrac{m(p)}{4}) - d|p|$$

which yields (i) of Theorem 4 if we disregard the term $-d|p|$ which is of lesser order than $\Phi_i(\frac{m(p)}{4})$ because of the monotonicity of Φ_i, the size of m, and the minimal growth requirement on Φ_i. Part (ii) of Theorem 4 follows from an obvious "guess and check" strategy.

III(d) Proof of Translation Lemma

Preliminary to the proof of Lemma 3, we will prove a simpler version, Lemma 4 below. Throughout the section, we will treat one-tape machines for the sake of clarity. The proofs for multitape machines are completely analogous.

Lemma 4. Let $\{\phi_i\}_{i\in\omega}$ be an enumeration by size of nondeterministic semi-finite Turing machines. There is an effective procedure h such that:

(i) $\forall i,x \in \omega$: $h(i,x)$ is a polynomial with integer coefficients and 58 variables.

(ii) If $\overset{\infty}{\forall} x \in \omega$: $\Phi_i(x) \geq \log\log(x)$, then

(a) $\overset{\infty}{\forall} x \in \omega$: $\phi_i(x)\downarrow \Rightarrow \exists z_1,\ldots,z_{58} \leq 2^{2^{2^{(\Phi_i(x))^5}}}$:
$h(i,x)(z_1,\ldots,z_{58}) = 0$

(b) $\overset{\infty}{\forall} x \in \omega$: $\phi_i(x)\uparrow \Rightarrow \forall z_1,\ldots,z_{58}$:
$h(i,x)(z_1,\ldots,z_{58}) \neq 0$.

(iii) $\forall i \in \omega$

$\overset{\infty}{\forall} x \in \omega$: $c(\log x)^2 \leq |h(i,j,x)| \leq (c+1)(\log^2 x)$

where c is a constant independent of i and x.

(iv) $\forall i,x \in \omega$: $H(i,x) \leq d|h(i,x)|$, where $H(i,x)$ is the running-time of h on input $<i,x>$ and d is a constant independent of i and x.

Proof of Lemma 4. Lemma 4 will follow from a chain of equivalences (1)-(7). For any ϕ_i and any $x,y \in \omega$:

(1) Some computation σ of $\phi_i(x)$ halts within y steps.

$\Leftrightarrow$

(2) There is a finite sequence σ' (corresponding to σ) of instantaneous descriptions $I_0,\ldots,I_y$, each of length $y+1$, such that:

(a) $I_0 = q_0\alpha_1\alpha_2\cdots\alpha_{|x|}\text{ƀƀ}\cdots\text{ƀ}$, where:
α_i: i^{th} digit of x, $i = 1,\ldots,|x|$
ƀ: blank
q_0: number representing initial state of ϕ_i.

(b) $I_y = q_{ACC}\beta_1\beta_2\cdots\beta_y$ where:
β_i: arbitrary symbols of our alphabet; $i = 1,\ldots,y$
q_{ACC}: number representing the accepting state of ϕ_i

(c) I_{j+1} follows directly from I_j by the rules of ϕ_i; $j = 1,2,\ldots,y-1$

$\Leftrightarrow$

(3) $\exists v,u\ \forall k \leq y(y+1)$ [

(a) $[\text{Rem}(u,1+v) = q_0\ \&\ \text{Rem}(u,1+(y^2+1)v) = q_{ACC}\ \&$
$\&\ \text{Rem}(u,1+2v) = \alpha_1\ \&\ \cdots\ \&$
$\&\ \text{Rem}(u,1+(|x|+1)v) = \alpha_{|x|}]\ \&$

(b) $(|x|+1) \leq k \leq (y+1) \Rightarrow \text{Rem}(u,1+kv) = \text{ƀ}$
$\&$ [

(c) [Rem(u,1+kv) = ƀ & Rem(u,1+(k+1)v) = ƀ &
& Rem(u,1+(k+2)v) = ƀ & Rem(u,1+(k+y+2)v) = ƀ]
or
[Rem(u,1+kv) = ƀ & Rem(u,1+(k+1)v) = ƀ &
& Rem(u,1+(k+2)v) = 0 & Rem(u,1+(k+y+2)v) = ƀ]
or
[Rem(u,1+kv) = ƀ & Rem(u,1+(k+1)v) = 0 &
& Rem(u,1+(k+2)v) = ƀ & Rem(u,1+(k+y+2)v) = 0]
or

$\vdots$

[Rem(u,1+kv) = 1 & Rem(u,1+(k+1)v) = 1 &
& Rem(u,1+(k+2)v) = 1 & Rem(u,1+(k+y+2)v) = 1]

(d) or
[Rem(u,1+kv) = ƀ & Rem(u,1+(k+1)v) = q_0 &
& Rem(u,1+(k+2)v) = ƀ & Rem(u,1+(k+y+1)v) = $q_{\dots}$ &
& Rem(u,1+(k+y+2)v) = ƀ & Rem(u,1+(k+y+3)v) = ...]
or
[Rem(u,1+kv) = ƀ & Rem(u,1+(k+1)v) = q_0 &
& Rem(u,1+(k+2)v) = ƀ & Rem(u,1+(k+y+1)v) = ƀ &
& Rem(u,1+(k+y+2)v) = ... &
& Rem(u,1+(k+y+3)v) = $q_{\dots}$]
or
[Rem(u,1+kv) = ƀ & Rem(u,1+(k+1)v) = q_1 &
& Rem(u,1+(k+2)v) = 0 & Rem(u,1+(k+y+1)v) = $q_{\dots}$ &
& Rem(u,1+(k+y+2)v) = ƀ & Rem(u,1+(k+y+3)v) = ...]
or

$\vdots$

or
[Rem(u,1+kv) = 1 & Rem(u,1+(k+1)v) = q_n &
& Rem(u,1+(k+2)v) = 1 & Rem(u,1+(k+y+1)v) = 1 &
& Rem(u,1+(k+y+2)v) = ... &
& Rem(u,1+(k+y+3)v) = $q_{\dots}$]
or
[Rem(u,(k+1)v) = q_{ACC} & Rem(u,(k+y+2)v) = q_{ACC}]

(e) or
[Rem(u,1+kv) = q_0 or Rem(u,1+kv) = q_1 or ... or
or Rem(u,1+kv) = q_n]
or
[Rem(u,1+(k+2)v) = q_0 or Rem(u,1+(k+2)v) = q_1 or
or ... or Rem(u,1+(k+2)v) = q_n]
or
[Rem(k,y+1) = 0 or Rem(k,y+1) = y]]].

Note: u, v encode the table of ID's in the sense of Gödel's Lemma.

Sections (a) and (b) express that the machine gets input x and accepts.

Section (c) expresses that the tape remains unchanged outside the vicinity of the tape head.

Section (d) expresses the legal state transitions, tape head movements, etc.

Section (e) expresses that we are looking at the square next to a state symbol or the boundary between two instantaneous descriptions; in such cases no specification of machine operation is in order.

Thus (c), (d) and (e) conjoined give an exhaustive listing of all possibilities for triples of symbols encoded by u, v.

The equivalence (2) ⇔ (3) follows from Gödel's Sequence Encoding Lemma. In fact it follows that if we can find u, v in (3) at all, there are $v \leq (n+5)y!$, $u \leq (y^2v)^{y^2}$ which work. Here (n+5) is the size of our I.D. alphabet.

⇔

(4) $\exists v,u,z' \ \forall k \leq y(y+1) \ \exists z \leq u$

$$\hat{P}_{i,x}(y,v,u,z',k,z) = 0$$

where $\hat{P}_{i,x}$ is derived from (3) by application of Lemma 1, after putting 3(a) outside of the universal quantifier ∀k and distributing part 3(b) over the disjunctions 3(c)-3(e). Any solution satisfies $z' < u$. $\hat{P}_{i,x}$ will be referred to again in the proof of the Translation Lemma.

⇔

(5) $\exists v,u,z',r,b[\binom{b}{u} \equiv \hat{P}_{i,x}(y,v,u,z',r,b) \equiv 0 \bmod \binom{r}{y^2+y+1}$

$\& \quad r > y(y+1) + [(y^2+y+1)^{y^2+y+1} \cdot Q^{u+1}] \ ]$

where $Q = Q(u,v,y,z')$ is the polynomial obtained from $\hat{P}_{i,x}$ by replacing each occurrence of k by y(y+1), each occurrence of z by u, and changing the sign of all negative coefficients; and where $\binom{s}{t}$ denotes the binomial coefficient $\frac{s!}{t!(s-t)!}$. The equivalence follows from a recent version of the Bounded Quantifier Theorem (Davis, Putnam, Robinson [6]) due to Matijesevic [15]. In fact it follows that if we can find solutions in (5) at all, we can find solutions satisfying

$$r \leq Q! \ , \quad b \leq r^{y^2+y+1}$$

as well as the bounds on v, u, z' given above.

⇔

(6) $\exists v,u,z',r,b$ [

$\exists z_0^1,\dots,z_{22}^1[P_c(r,y^2+y+1,z_0^1,\dots,z_{15}^1) = 0 \ \&$

$\& \ P_\ell(z_4^1,z_5^1,[(z_1^1+1)(8r+8z_{13}^1z_1^1+8z_0^1+2)],$

$(r+1),z_{16}^1,\dots,z_{22}^1) = 0] \ \&$

$\exists z_0^2,\dots,z_{22}^2[P_c(b,u,z_0^2,\dots,z_{15}^2) = 0 \ \&$

$\& \ P_\ell(z_4^2,z_5^2,[(z_1^2+1)(8b+8z_{13}^2z_1^2+8z_0^2+2)],$

$b+1,z_{16}^2,\dots,z_{22}^2) = 0] \ \&$

$\exists z_1^3,\dots,z_4^3[P_r(r,u,v,y,z^1,z_1^3,\dots,z_4^3) = 0] \ \&$

$\exists z_1^4z_2^4[z_1^1z_1^4 = z_1^2 \ \& \ z_1^1z_2^4 = \hat{P}_{i,x}(y,v,u,z',r,b)]]$

where:

(A) P_ℓ is a polynomial with integer coefficients such that:

(a) $\exists x_1,\dots,x_7 P_\ell(a_1a_2a_3a_4,x_1,\dots,x_7) = 0 \Leftrightarrow$ $\langle a_1,a_2\rangle$ is the a_4+1^{st} pair of solutions to

$$x^2((a_3)^2-1) + 1 = y^2$$

in order of size. (See "Pell's Equation", [13]).

(b) (Julia Robinson [15]) $\langle a_1,a_2\rangle$ is the a_4+1^{st} pair of solutions to $x^2(a_3^2-1) + 1 = y^2$ in order of size ⇔

$\exists x_1 \cdots x_7$

$\leq 3a_3^{16[(a_4+1)a_1+(a_4+1)^2a_1]} \cdot 2^{2(a_4+1)}$:

$P_\ell(a_1a_2a_3a_4,x_1,\dots,x_7) = 0$.

(B) P_c is a polynomial with integer coefficients such that:

(a) $a_3 = \binom{a_1}{a_2} \Rightarrow \exists x_1,\dots,x_6 \leq (2a_1)^{[a_2(a_1+1)]^2}$:

$P_c(a_1,a_2,a_3,x_1,\dots,x_{15}) = 0.$

(b) $\exists x_1 \cdots x_{15}$: $P_c(a_1a_2a_3,x_1,\dots,x_{15}) = 0 \Rightarrow$

(i) $a_3 = \binom{a_1}{a_2}$ or

(ii) $\langle x_4,x_5\rangle$ is <u>not</u> the a_4+1^{st} pair of solutions to

$$x^2[(8a_1+8x_{13}x_1+8a_3+2)^2(x_1+1)^2-1] + 1 = y^2 \ .$$

P_ℓ and P_c are derived from systems found in [10]. From their properties it is clear that they can be used in conjunction to define a binomial coefficient, as is done in (6).

(C) P_r is a polynomial (depending on Q) with integer coefficients such that

(a) $\exists x_1 \cdots x_4\ P_r(a_1,\ldots,a_5,x_1,\ldots,x_4) = 0 \Leftrightarrow$

$$a_1 > a_4(a_4+1) + [(a_4)^2+a_4+1]^{(a_4)^2+a_4+1} \cdot Q(a_2,a_3,a_4,a_5)^{a_2+1}$$

(b) $P_r(a_1,\ldots,a_5,x_1,\ldots,x_4) = 0 \Rightarrow$ $a_2,\ldots,a_5,x_1,\ldots,x_4$ are all less than a_1.

$\Leftrightarrow$

(7) $\exists z_1,\ldots,z_{57}\ P_{i,x}(y,z_1,\ldots,z_{57}) = 0$ where $P_{i,x}$ is derived from the system in (6) by application of Lemma 1(i). Moreover, it follows from the bounds already given that if $P_{i,x}$ has a solution for a given y, then there is a solution such that

$$z_j \le 2^{2^{2^{2^{cy^4|y|}}}}, \quad j = 1,2,\ldots,57$$

where c is independent of y but dependent on i and x.

Now clearly,

$$\phi_i(x)\downarrow \Leftrightarrow \exists \Phi_i(x) \in \omega\colon \phi_i(x) \text{ halts in } \Phi_i(x) \text{ steps .}$$

Thus from (1) $\Leftrightarrow$ (7) we have (i) and (ii) of Lemma 4; $h(i,x) = P_{i,x}$, using the hypothesis that $\Phi_i(x) \ge \log\log x$ so inequalities for almost all y imply the corresponding inequalities (with y replaced by $\Phi_i(x)$) for almost all x. The length of $P_{i,x}$ can be seen to be

$$|h(i,x)| = c(\log(x))^2 + c_i(\log x)$$

where c_i depends on i but not x, and c is independent of i and x. From this (iii) of Lemma 4 follows.

Clearly, $P_{i,x}$ can be obtained from inputs x and i on a multitape machine in a number of steps (up to a constant) equal to its length; thus (iv) of Lemma 4 follows.

Proof of Lemma 3. For any i, j, x, h(i,j,x) will be obtained by a modification of $P_{i,x}$. Motivation for this modification is derived from the following number-theoretic lemma which we state here without proof:

Lemma 5. Let $a,b,x,y \in \omega$, $a > b > 0$. If $\langle a,b,x,y\rangle$ satisfy

$$\left.\begin{aligned} x^2(a^2-1)+1 &= y^2 \\ x &\equiv b \bmod (a-1) \end{aligned}\right\} \qquad (*)$$

then a, b and x are less than y and either

(i) $\langle x,y\rangle$ is the b^{th} solution (in order of size) of

$$x^2(a^2-1)+1 = y^2$$

and $y < (2a)^b$, or

(ii) $y > a^a$, and, for some $k > 0$, $\langle x,y\rangle$ is the $b+k(a-1)^{th}$ solution of $x^2(a^2-1)+1 = y^2$.

In the system of equations S in (6) above, two binomial coefficients are defined, each by a conjunction of an occurrence of P_c and P_ℓ. P_c contains a system of equations of the form (*). It is the function of P_ℓ in each case to ensure that for any solution of S and hence of the conjunction of P_c and P_ℓ, case (i) of Lemma 5 holds rather than case (ii). This in turn ensures that P_c defines a binomial coefficient (see 6B above).

Now consider the system S' of equations obtained from S by omitting both occurrences of "$P_\ell(\cdots) = 0$". It can now no longer be assumed that case (ii) of Lemma 5 cannot occur for the systems of the form (*) in P_c. We will call a solution of S' a solution of type (i) if the solution to the system of the form (*) in each of the P_c's of S' is of type (i); otherwise we will call a solution of S' a solution of type (ii).

From Lemma 5 we note that solutions of (*) of types (i) and (ii) can be distinguished by their size. By proper choice of the parameters a and b occurring in (*) we can ensure that for any solution of (*) of type (ii) the value of the variable y in (*) exceeds the values of all variables occurring in the smallest solutions of type (i).

Applying this to our system S', we would like to choose the parameters in the occurrences of (*) in S' in such a way that the value of the variables in S' corresponding to y in (*) in any type (ii) solution of S' exceed the values of all variables occurring in the smallest type (i) solutions of S'. We can then arrange that such type (ii) solutions will not fall within the bound on solutions allowed in clause ii(a) of Lemma 3; thus there will be no need to use P_ℓ to eliminate such solutions. Since equations "$P_\ell = 0$" have only large solutions, elimination of such equations from S will improve the upper bound on the size of solutions from that obtained in Lemma 4 to the bound required in Lemma 3. The proper choice of parameters will be obtained by a modification of P_c. Let z be an additional variable. A certain term occurring in P_c (which can be observed in the form (6) as the second factor in the third parameter of P_ℓ) will be multiplied by z in all of the several places where it occurs. The effect of this manipulation is to multiply by z the quantity corresponding to "a" in (*).† The resulting polynomial will be $P_c'(z,a_1a_2a_3,x_1,\ldots,x_{15})$. It can be shown that, for any $z \in \omega$, P_c' has a solution of type (i) if and only if P_c does. Moreover, the bound on the size of variables occurring in P_c given above can be replaced by the following bound for P_c':

$$x_1,\ldots,x_{15} \le (2a_1)^{[a_2(a_1+1)]^2} \cdot z^{(a_1+1)} .$$

From Lemma 5 it can be seen that the "proper choice of parameters" can be obtained by making z large. This will make the parameters of P_c' which correspond to "a" in (*) large, which will in turn cause solutions of type (ii) to be large. A large z will be obtained by adding a condition to S': $P_z = 0$, where P_z is a polynomial with integer coefficients such that:

(a) $P_z(z,b,x_1 \cdots x_{12}) = 0 \Rightarrow z \ge 2^{2^{2^{2^b}}}$ and $x_1,\ldots,x_{12} \le z$

(b) $\forall b \in \omega\ \exists z,x_1,\ldots,x_{12}\colon z \le \left(2^{2^{2^{2^b}}}\right)^{2^{2^{2^{2^b}}}}$: $P_z(z,x_1,\ldots,x_{12}) = 0$

In our use of P_z, the parameter b will be replaced by the variable y occurring in $P_{i,x}$ (see Lemma 4): Let the system S" be

†The modification can be precisely specified in terms of formula (1), Section 10, of [10]: In

$$\frac{\psi_{M(X+1)}(N+1)}{\psi_M(N-S+1)\psi_{MX}(S+1)}$$

and associated equations, of P_c, we replace M by zM.

$$P_{i,x}(\bar{y},\bar{z}_1,\ldots,\bar{z}_{57}) = 0 \;\&$$
$$\& \; P_c'(z,r,y^2+y+1,z_0^1,\ldots,z_{15}^1) = 0 \;\&$$
$$\& \; P_c'(z,b,u,z_0^2,\ldots,z_{15}^2) = 0 \;\&$$
$$\& \; P_r(r,u,v,y,z^1,z_1^3,\ldots,z_4^3) = 0 \;\&\; z_1^1 z_1^4 = z_1^2 \;\&$$
$$\& \; z_1^1 z_2^4 = \hat{P}_{j,x}(y,v,u,z',r,b) \;\&\; P_z(z,\bar{y},z_1^5,\ldots,z_{12}^5) = 0$$

It is clear that S" is just the conjunction of

S for <i,x>, with running time variable $\bar{y}$;
S' for <j,x>
$P_z(z,\bar{y},\ldots) = 0$

Let $P_{i,j,x}$ be the polynomial obtained from S" by application of Lemma 1(i). We will show that $P_{i,j,x}$ has the properties required of h(i,j,x) in Lemma 3.

Case 1: $\phi_j(x)\downarrow$, $\phi_i(x)\downarrow$. By Lemma 4, there is a solution of $P_{i,x} = 0$, with running time variable $\bar{y}$; by property (b) of P_z given above, there is a solution of $P(z,\bar{y},\ldots) = 0$ with z within the bounds given. By Lemma 4 and the comments made in describing P_c' there is a solution of type (i) of S' for <j,x> with z as occurring in the solution of $P(z,\bar{y},\ldots) = 0$. Thus there is a solution of S" and $P_{i,j,x}$. We now show that the size of variables in this solution is within the bound specified in Lemma 3. It is immediate from the bound given on the size of solutions of P_c' that this bound must exceed z (as z occurs in it) and hence all variables in P_z are within this bound. On the other hand, by Lemma 4, the variables of S (for <i,x>) can be bounded by

$$2^{2^{2^{2^{(\Phi_i(x))^5}}}}$$

and $\bar{y}$ can be taken as $\Phi_i(x)$. Thus by the bound on z we find

$$z < 2^{\left(2^{2^{2^{2^{\Phi_i(x)}}}}\right)^2} < 2^{\left(2^{2^{2\Phi_j(x)}}\right)^2} < 2^{2^{2^{4\Phi_j(x)}}} .$$

Using this bound we can calculate the bound on the solutions of both occurrences of P_c' as

$$2^{2^{2^{(\Phi_j(x))^5}}}$$

for almost all x. (It should be noted that many of the inequalities used in the derivation of our bounds are invalid for x such that $\Phi_j(x)$ is a very small number. It is at this point that we use the assumption that Φ_i is an ultimately growing function, i.e. $\Phi_i(x) \underset{aax}{\geq}$ log log(x), to infer that the inequalities are valid for almost all x.) Finally, using the inequality between $\Phi_i(x)$ and $\Phi_j(x)$, we infer that the bound on solutions of P_c' dominates that on solutions of S for <i,x> for almost all x. This completes the proof of (ii)(a) of Lemma 3.

Case 2: $\phi_j(x)\uparrow$, $\phi_i(x)\downarrow$. (Note that ϕ_i is total.) Examining S" again, we find solutions to S for <i,x> and $P_z(z,\bar{y},\ldots) = 0$ as before. However the subsystem S' of S" could have no solution of type (i), since by the properties of P_c' and P_c and the equivalences (1)-(6) in the proof of Lemma 4, it would follow that u, v in S' for <j,x> encoded a halting computation of $\phi_j(x)$ contradicting the assumptions. Thus any solutions of S' for <j,x> are of type (ii), and the corresponding a's (compare Lemma 5) have been made multiples of z. But because $P_{i,x}(\bar{y},\ldots) = 0$ and $P_z(z,\bar{y},\ldots) = 0$,

$$z > 2^{2^{2^{2^{2^{\Phi_i(x)}}}}} .$$

Thus by 5(ii), the solution of at least one of the P_c''s in S" must contain variables with values exceeding

$$\left(2^{2^{2^{2^{2^{\Phi_i(x)}}}}}\right)^{2^{2^{2^{2^{2^{\Phi_i(x)}}}}}} > 2^{2^{2^{2^{2^{2^{\Phi_i(x)}}}}}} > 2^{2^{2^{2^{\Phi_j(x)}}}}$$
$$> 2^{2^{2^{(\Phi_j(x))^5}}}$$

for almost all x, which completes the proof of ii(b) of Lemma 3.

It is easily seen by the arguments given in the proof of Lemma 4 that the polynomial $P_{i,j,x}$ also satisfies the other clauses of Lemma 4; the number of variables in the system discussed is 114.

It should be noted that the polynomials $P_{i,x}$ and $P_{i,j,x}$ described above were chosen, for reasons of simplicity, and could be replaced by more carefully constructed systems with about half as many variables.

Open Problem 13. How many variables are required for polynomials expressing Turing machine operation produced by a uniform procedure h with properties as in Lemmas 3 and 4?

By Matijesevic and Robinson [10], it follows that 13 variables would be sufficient if we would disregard the bounds in 3(ii) and 4(ii). Our conjecture: a 15-20 variables result could be obtained by careful use of the techniques in this paper.

As mentioned earlier, different uses of the techniques of this paper produce different results. In particular, if, instead of encoding the operation of ϕ_j on input x as done above, we would encode the operation of a universal Turing machine simulating the operation of ϕ_j on unary inputs only, we obtain theorems with complexity bounds similar to those of Theorems 2, 3, 4 and 5, but which apply to classes of polynomials with a fixed degree as well as a fixed number of variables.

A different improvement is obtained when, as in Theorems 2 and 3 above, the machine ϕ_j to be encoded has elementary running time, e.g. $\Phi_j(x) \leq 2^{|x|^c}$ for some $c \in \omega$. Then the occurrence of $P_{i,x}$ in the system S" and in $P_{i,j,x}$ can be dropped in favor of a direct definition or even (as in the example) explicit reference to $\Phi_j(x)$. This approximately halves the number of variables in $P_{i,j,x}$. Thus Theorems 2 and 3 hold for polynomials with fewer variables than Theorem 4.

Conclusion

It is becoming increasingly supportable that "computable in polynomial time" rather than just "computable" is the formal analogue of the philosophical concept "decidable". Therefore it is important to have methods of classifying intractable problems (i.e., problems which are computable but not in polynomial time) and to apply these methods to natural problems. Unfortunately, the fundamental question in this area, NP $\overset{?}{=}$ P, has shown great resistance to resolution and has discouraged some people from research on this topic. We hope this paper will help show that interesting results can be obtained despite this barrier.

Acknowledgment

We wish to thank Julia Robinson, whose exposition of diophantine definability and the solution of Hilbert's 10th problem during a seminar at Berkeley has been of incalculable value to us.

References

[1] Baker, A., Linear Forms in the Logarithms of Algebraic Numbers, *Mathematika* *13* (1966), p. 205.

[2] Blum, M., A Machine Independent Theory of Complexity of Recursive Functions, *J.ACM* *14* (1967), pp. 322-336.

[3] Bennett, , Ph.D. thesis, University of California, Berkeley.

[4] Cook, S.A., The Complexity of Theorem Proving-Procedures, *Conf. Rec. 3rd ACM Symp. Theory of Computing* (1971), pp. 151-158.

[5] Davis, M.D., Speedup Theorems and Diophantine Equations, *Computational Complexity Symp.*, Courant Institute (1971).

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

[7] Grzegorczyk, A., Some Classes of Recursive Functions, *Ruzprawy Matematyczne* *4*, Instytut Matematyczny Polskiej Akademie Nauk, Warsaw.

[8] Jones, J.P., Sato, D., Wado, H., Wiens, D., Diophantine Representation of the Set of Prime Numbers, *Amer. Math. Monthly*, to appear.

[9] Matijesevic, Y., Anumerable Sets Are Diophantine (Russian), *Dokl. Akad. Nauk SSSR* *191* (1970), pp. 279-282.

[10] Matijesevic, Y., Robinson, J., Reduction of an Arbitrary Diophantine Equation to One in 13 Unknowns, to appear.

[11] Meyer, A.R. and Stockmeyer, L.J., The Equivalence Problem for Regular Expressions with Squaring Requires Exponential Space, *13th Annual IEEE Symp. on Switching and Automata Theory* (Oct. 1972), pp. 125-129.

[12] Miller, G.L., Riemann's Hypothesis and Tests for Primality, *7th ACM Symp. on Theory of Computing* (1975), p. 234.

[13] Niven, I., Zuckerman, H., *An Introduction to the Theory of Numbers*, John Wiley and Sons, Inc. (1972).

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

[15] Robinson, J., Seminar on Hilbert's 10th Problem, Berkeley (1975).

[16] Rogers, H. Jr., *Recursive Functions and Effective Computability*, McGraw-Hill, New York (1967).

[17] Seiferas, J.I., Fischer, M.J., Meyer, A.R., Refinements of the Nondeterministic Time and Space Hierarchies, *14th Annual IEEE Symp. on Switching and Automata Theory* (Oct. 1973), pp. 130-136.

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