MAC TECHNICAL MEMORANDUM 43

# SUPER-EXPONENTIAL COMPLEXITY OF PRESBURGER ARITHMETIC

Michael J. Fischer

Michael O. Rabin

February 1974

This research was supported in part by the National Science Foundation under research grant GJ-34671 and in part by the Artificial Intelligence Laboratory, an M.I.T. research program supported by the Advanced Research Projects Agency, Department of Defense which was monitored by ONR Contract No. N00014-70-A-0362-0003.

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

PROJECT MAC

CAMBRIDGE MASSACHUSETTS 02139

*This empty page was substituted for a blank page in the original document.*

# SUPER-EXPONENTIAL COMPLEXITY OF PRESBURGER ARITHMETIC[†]

Michael J. Fischer
Massachusetts Institute of Technology, Cambridge, Massachusetts

and

Michael O. Rabin
Hebrew University, Jerusalem, Israel

## ABSTRACT

Lower bounds are established on the computational complexity of the decision problem and on the inherent lengths of proofs for two classical decidable theories of logic: the first order theory of the real numbers under addition, and Presburger arithmetic -- the first order theory of addition on the natural numbers. There is a fixed constant $c > 0$ such that for every (non-deterministic) decision procedure for determining the truth of sentences of real addition and for all sufficiently large $n$, there is a sentence of length $n$ for which the decision procedure runs for more than $2^{cn}$ steps. In the case of Presburger arithmetic, the corresponding bound is $2^{2^{cn}}$. These bounds apply also to the minimal lengths of proofs for any complete axiomatization in which the axioms are easily recognized.

---

[†]This research was supported in part by the National Science Foundation under research grant GJ-34671 to M.I.T. Project MAC, and in part by the Artificial Intelligence Laboratory, an M.I.T. research program sponsored by the Advanced Research Projects Agency, Department of Defense, under Office of Naval Research contract number N00014-70-A-0362-0003. The preparation of the manuscript was supported by the Hebrew University and the University of Toronto.

## 1. Introduction and Main Theorems.

We present some results obtained in the Fall of 1972 on the computational complexity of the decision problem for certain theories of addition. In particular we prove the following results.

Let L be the set of formulas of the first-order functional (predicate) calculus written using just + and =. Thus, for example, $\sim[x + y = y + z] \vee x + x = x$ is a formula of L, and $\forall x \exists y[x + y = y]$ is a sentence of L. Even though this is not essential, we shall sometimes permit the use of the individual constants 0 and 1 in writing formulas of L. We assume a finite alphabet for expressing formulas of L, so a variable in general is not a single atomic symbol but is encoded by a sequence of basic symbols.

Let $\eta = \langle N, + \rangle$ be the structure consisting of the set $N = \{0, 1, 2, \ldots\}$ of natural numbers with the operation + of addition. Let $Th(\eta)$ be the first-order theory of $\eta$, i.e. the set of all sentences of L which are true in $\eta$. For example, $\forall x \forall y[x + y = y + x]$ is in $Th(\eta)$. Presburger has shown that $Th(\eta)$ is decidable [2]. For brevity's sake, we shall call $Th(\eta)$ *Presburger arithmetic* and denote it by PA.

Theorem 1. There exists a constant $c > 0$ such that for every decision procedure (algorithm) AL for PA, there exists an integer $n_0$ so that for every $n > n_0$ there exists a sentence F of L of length n for which AL requires more than $2^{2^{cn}}$ computational steps to decide whether $F \in PA$.

The previous theorem applies also in the case of non-deterministic algorithms. This implies that not only algorithms require a super-exponential number of computational steps, but also proofs of true statements concerning

addition of natural numbers are super-exponentially long. Let AX be a system of axioms in the language L (or in an extension of L) such that a sentence $F \in L$ is provable from AX ($AX \vdash F$) if and only if $F \in PA$. Let AX satisfy the condition that to decide for a sentence F whether $F \in AX$, i.e. whether F is an axiom, requires a number of computational steps which is polynomial in the length $|F|$ of F.

Theorem 2. There exists a constant $c > 0$ so that for every axiomatization AX of Presburger arithmetic with the above properties there exists an integer $n_0$ so that for every $n > n_0$ there exists a sentence $F \in PA$ such that the shortest proof of F from the axioms AX is longer than $2^{2^{cn}}$. By the length of a proof we mean the number of its symbols.

With slight modifications, Theorem 2 holds for any (consistent) system AX of axioms in a language M in which the notion of integer and the operation + on integers are definable by appropriate formulas so that under this interpretation, all the sentences of PA are provable from AX. The ordinary axioms ZF for set theory have this property.

The result concerning super-exponential length of proof applies, in this more general case, to the sentences of M which are encodings of sentences of PA under the interpretation, i.e. to sentences which express elementary properties of addition of natural numbers.

The previous results necessarily involve a cut-point $n_0(AL)$ or $n_0(AX)$ at which the super-exponential length of computation or proofs sets in. It is significant that a close examination of our proofs reveals that $n_0(AL) = O(|AL|)$ and $n_0(AX) = O(|AX|)$. Thus computations and proofs become very long quite early in the game.

The theory PA of addition of natural numbers is one of the simplest most basic imaginable mathematical theories. Unlike the theory of addition and multiplication of natural numbers, PA is decidable. Yet any decision procedure for PA is inherently difficult.

Let us now consider the structure $\mathcal{R} = \langle R, + \rangle$ of all *real* numbers R with addition. The theory $Th(\mathcal{R})$ (in the same language L) is also decidable. In fact, to find a decision procedure for $Th(\mathcal{R})$ is even simpler than a procedure for PA; this is mainly because $\mathcal{R}$ is a divisible group without torsion. Yet the following holds.

Theorem 3. There exists a constant $d > 0$ so that for the theory $Th(\mathcal{R})$ of addition of real numbers, the statement of Theorem 1 holds with the lower bound $2^{dn}$.

Similarly for the length of proofs of sentences in $Th(\mathcal{R})$.

Theorem 4. There exists a constant $d > 0$ so that for every axiomatization AX for $Th(\mathcal{R})$ the statement of Theorem 2 holds with the lower bound $2^{dn}$.

Corollary 5. The theory of addition and multiplication of reals (Tarski's Algebra [3]) is exponentially complex in the sense of Theorems 3 and 4.

(This result was obtained independently by V. Strassen.)

Ferrante and Rackoff [1] describe decision procedures for $Th(\mathcal{R})$ and PA which run in deterministic space $2^{cn}$ and $2^{2^{dn}}$ (and hence in deterministic time $O(2^{2^{cn}})$ and $O(2^{2^{2^{dn}}})$), respectively, for certain constants c and d. Any substantial improvement in our lower bounds would settle some general open automata-theoretic questions on the relation between time and space. For example, a lower bound of time $2^{n^2}$ for the decision problem for $Th(\mathcal{R})$

would give an example of a problem doable in space $S(n) = 2^{cn}$ but not in time bounded by a polynomial in S.

Variations of the methods employed in the proofs of Theorems 1-4 lead to complexity results for the (decidable) theories of multiplication of natural numbers, finite Abelian groups, and other classes of Abelian groups. Some of these results are stated in Section 7 and will be presented in full in a subsequent paper.

The fact that decision and proof procedures for such simple theories are exponentially complex is of significance to the program of theorem proving by machine on the one hand, and to the more general issue of what is knowable in mathematics on the other hand.

## 2. Algorithms.

Since we intend to prove results concerning the complexity of algorithms, we must say what notion of algorithm we use. Actually our methods of proof and our results are strong enough to apply to any reasonable class of algorithms or computing machines. However, for the sake of definiteness, we shall assume throughout this paper that our algorithms are the programs for Turing machines on the alphabet {0, 1}.

We proceed to give an informal description of these algorithms. The machine-tape is assumed to be one-way infinite extending to the right from an initial left-most square. At any given time during the progress of a computation, all but a finite number of the tape's squares contain 0. An *instruction* has the form: "i: If 0 then print $X_0$, move $M_0$, go to one of $i_1, i_2, \ldots$ ; if 1 then print $X_1$, move $M_1$, go to one of $j_1, j_2, \ldots$ ." Here $i, i_1, i_2, \ldots, j_1, j_2, \ldots$ are natural numbers, the so-called *instruction*

*numbers*; $X_0$ and $X_1$ are either 0 or 1; and $M_0$ and $M_1$ are either R or L (for "move right" and "move left", respectively).

The possibility of going to one of several alternative instructions embodies the non-deterministic character of our algorithms. Another type of instruction is: "i: *Stop*." Instructions are abbreviated by dropping the verbal parts. Thus, "3: 0, 1, L, 72, 5; 1, 1, R, 15, 3." is an example of an instruction. A program AL is a sequence $I_1, \ldots, I_n$ of instructions. For definiteness' sake we assume that the instruction number of $I_i$ is i and that $I_n$ is the instruction "n: *Stop*." Furthermore AL is assumed to be coded in the binary alphabet {0, 1} in such a way that "*Stop*" also serves as an end-word indicating the end of the binary word AL.

Let $x \in \{0, 1\}^*$ be an input word. To describe the possible computations by the algorithm AL on x, we assume that x is placed in the leftmost positions of the machine's tape and the scanning head is positioned in the leftmost square of the tape. The computation starts with the first instruction $I_1$. A *halting computation* on x is a sequence $C = (I_{i_1}, \ldots, I_{i_m})$ of instructions of AL so that $i_1 = 1$ and $i_m = n$. At each step $1 \leq p \leq m$, the motion of the scanning head, the printing on the scanned square, and the transfer to the next instruction $I_{i_{p+1}}$, are according to the current instruction $I_{i_p}$. The *length* $\ell(C)$ of C is, by definition, m.

It is clear that a truly non-deterministic program may have several possible computations on a given input x.

## 3. Method for Complexity Proofs.

Having settled on a definite notion of algorithm, we shall describe a general method for establishing lower bounds for theories of addition which are formalized in L. We do not develop our methods of proof in their fullest generality but rather utilize the fact that we deal with natural or real numbers to present the proofs in a more readily understandable and concrete form. The refinements and generalizations which are needed for other theories of addition will be introduced in a subsequent paper.

Theorem 6. Let $f(n)$ be one of the two functions $2^n$ or $2^{2^n}$. Assume for a complete theory T that there exists a polynomial $p(n)$ and a constant $d > 0$ so that for every program AL and binary word x, there exists a sentence $F_{AL,x}$ with the following properties.

(a) $F_{AL,x} \in T$ if and only if some halting computation C of AL on x satisfies $\ell(C) \leq f(|x|)$.

(b) $|F_{AL,x}| \leq d\cdot(|AL| + |x|)$.

(c) $\sim F_{AL,x}$ is Turing machine calculable from AL and x in time less than $p(|AL| + |x|)$.

(We recall that all our objects such as F, AL, etc., are binary words, and that $|w|$ denotes the length of w.)

Under these conditions, there exists a constant $c > 0$ so that for every decision algorithm AL for T there exists a number $n_0 = n_0(AL)$ so that for every $n > n_0$ there exists a sentence $\sigma \in T$ such that $|\sigma| = n$ and every computation by AL for deciding $\sigma$ takes more than $f(cn)$ steps. Furthermore $n_0(AL) = O(|AL|)$.

Proof. There exists a number $c > 0$ and an $m_0$ so that for $m \geq m_0$ we have

$$p(2m) + f(c\cdot(2dm + 1)) \leq f(m). \tag{1}$$

Namely, let $c < 1/(2d)$ and recall that $p(n)$ is a polynomial, whereas $f(n)$ is $2^n$ or $2^{2^n}$.

Let AL be a (non-deterministic) decision algorithm for T. We construct a new algorithm $AL_0$ as follows. We do not care how $AL_0$ behaves on an input word x which is not a program. If x is a program, then $AL_0$ starts by constructing the sentence $F = \sim F_{x,x}$. The program $AL_0$ then switches to AL which works on the input F. If AL stops on F and determines that $F \in T$, then $AL_0$ halts; in all other cases $AL_0$ does not halt. Thus, for a program x as input, $AL_0$ halts if and only if the program x does *not* halt on the input x in fewer than $f(|x|)$ steps. Note that by possibly padding $AL_0$ with irrelevant instructions, we may assume that $m_0 \leq |AL_0| \leq |AL| + k$, where k is independent of AL.

Denote the binary word $AL_0$ by z and let $\sigma$ be the sentence $\sim F_{z,z}$. $F_{z,z}$ cannot be true, for if it were true, then $\sim F_{z,z}$ would be false and $AL_0$ would not halt on z, whereas the truth of $F_{z,z}$ implies that z (= $AL_0$) does halt on the input z (even in at most $f(|z|)$ steps), a contradiction.

Thus, $\sigma$ is true and hence $AL_0$ (= z) halts on z. The truth of $\sigma$ also implies that every halting computation of $AL_0$ on z is longer than $f(|z|)$.

Let $m = |z|$. By (b), we have

$$n = |\sigma| \leq 2dm + 1. \tag{2}$$

Let t be the least number of steps that AL takes, by some halting computation,

to decide $\sigma$. By the definition of $AL_0$ and the fact that fewer than $p(2m)$ steps are required to find $\sigma = \sim F_{z,z}$ from z (this follows from (c) and $|z| = m$), there is a halting computation of the program $AL_0$ on z requiring fewer than $p(2m) + t$ steps. By the truth of $\sigma$,

$$p(2m) + t > f(m).$$

Using (1) and (2),

$$t > f(c\cdot(2dm + 1)) \geq f(cn).$$

Take $n_0$ to be $n = |\sigma|$. Then $n_0 \leq 2dm + 1 \leq 2d(|AL| + k) + 1$, so $n_0 = O(|AL|)$. The fact that the result holds for AL and every $n > n_0$ (with possibly a smaller constant c) is obtained by first padding $AL_0$ by irrelevant instructions, and then padding the resulting $\sigma$ by prefixing a quantifier $\exists x_j$ of an appropriate length, where $|\exists x_j| = 1 + |j|$. The details are left to the reader. □

For utilizing Theorem 6 we need a method for constructing sentences $F_{AL,w}$ with the properties (a) - (c). One such method is provided by:

Theorem 7. Let $\mathcal{A} = \langle A, + \rangle$ be an additive structure such that $N \subseteq A$, and on N the operation + is ordinary addition. Let f(n) again be one of the functions $2^n$ or $2^{2^n}$. Assume that $T = Th(\mathcal{A})$ is a theory of addition (formalized in the language L) for which there exists $c > 0$ such that for every n and for every binary word w, $|w| = n$, there exist formulas $I_n(y)$, $J_n(y)$, $S_n(x, y)$ and $H_w(x)$ with the following properties.

(i) $|S_n(x, y)| \leq cn$, $|I_n(y)| \leq cn$, $|J_n(y)| \leq cn$, and $|H_w(x)| \leq cn$.

(ii) $I_n(b)$ is true in $\mathcal{A}$ for $b \in A$ if and only if $b \in N$ and $b < f(n)^2$.
$J_n(b)$ is true exactly for $b = f(n)$.

(iii) $S_n$ codes all binary sequences of length $f(n)^2$. Namely, for every binary sequence $\beta \in \{0, 1\}^*$, $|\beta| = f(n)^2$, there exists an $a \in A$ so that for $i \in N$, $0 \leq i < f(n)^2$, $S_n(a, i)$ is true in $\mathcal{A}$ if $\beta(i) = 1$, and $S_n(a, i)$ is false if $\beta(i) = 0$, where for any sequence $\beta$, $\beta(i)$ denotes the $i+1^{st}$ element of $\beta$, $0 \leq i < |\beta|$.

(iv) $H_w(x)$ is true for $a \in A$ if and only if the first $f(n)$ symbols of the sequence coded by $a$ in the sense of (iii) has the form $wO^p$, $p = f(n) - |w|$.

(v) $S_n(x, y)$, $I_n(y)$, $J_n(y)$ and $H_w(x)$ are Turing machine calculable from n and w in a polynomial number of steps.

From such formulas $S_n$, $I_n$, $J_n$ and $H_w$, a formula $F_{AL,w}$ with the properties (a) - (c) can be constructed, so that T satisfies the conclusion of Theorem 6.

Proof. We shall describe, by use of sequences of length $f(n)^2$, all possible halting computations of length at most $f(n)$ of a program AL on an input w. Let $C = (I_{i_1}, \ldots, I_{i_m})$ be such a computation. Assume that AL has k instructions; by our notational conventions every computation starts with the first instruction $I_1$ and the last instruction $I_k$ of AL is: "k: *Stop*." Thus in C, $i_1 = 1$ and $i_m = k$.

Let us adopt the convention that after the stop instruction, the scanning head, the (stop) instruction, and the tape contents stay stationary and unchanged at all subsequent time instants. Since $m \leq f(n)$, the scanning head never moves beyond $f(n)$ squares from the initial left-most square of the tape. We assume also that the Turing machine never attempts to shift its head left off the beginning of the tape.

The progress of the computation C on the input w will be described by stringing together f(n) instantaneous descriptions of the computation in the following manner. Let $W_j$ be the first (left-most) f(n) symbols of the tape at time j, $1 \leq j \leq f(n)$. Then the string $W_1W_2\ldots W_{f(n)} = W \in \{0, 1\}^*$ codes all the relevant information concerning the tape contents during the computation C. We have $|W| = f(n)^2$. Also, $W_m = W_{m+1} = \ldots$ .

To trace the motion of the scanning head and the sequence of instructions during the computation C, we define $U_j \in \{0, 1, \ldots, k\}^*$ to be $0^{p_j}\, i_j\, 0^{q_j}$ where $p_j + q_j + 1 = f(n)$ and $p_j$ is the distance at time j of the scanning head from the start square, $1 \leq j \leq f(n)$. Recall that $i_j$ is the instruction number of the $j^{th}$ instruction executed in C. Also $i_m = i_{m+1} = \ldots = k$, the stop instruction. Put $U = U_1U_2\ldots U_{f(n)}$. We have $|U| = f(n)^2$.

The fact that the pair (W, U), where $W \in \{0, 1\}^*$, $U \in \{0, 1, \ldots, k\}^*$, $|W| = |U| = f(n)^2$, describes a halting computation of AL on w, is equivalent to a number of statements which say, roughly, that the first f(n) symbols are the initial configuration; that the transformation from a block of f(n) symbols to the next block is by an instruction of AL; and that U contains k (the number of the halting instruction). More precisely, (W, U) codes a halting computation of length at most f(n) of AL on w, where $|w| = n$, if and only if:

($\alpha$) $W(0)\ldots W(f(n)-1) = w0^p$, $p = f(n) - |w|$.

($\beta$) $U(0)\ldots U(f(n)-1) = 10^{f(n)-1}$.

($\gamma$) If $U(i) = 0$ and $i + f(n) < f(n)^2$, then $W(i + f(n)) = W(i)$.

($\delta$) If $U(i) = q$, $i + f(n) + 1 < f(n)^2$, $0 < q < k$, $W(i) = 0$, and $I_q$ is, say, "q: 0, 1, R, $k_1$, ... , $k_t$; 1, ...", then $W(i + f(n)) = 1$, $U(i + f(n) + 1) = k_1$ or $U(i + f(n) + 1) = k_2$ or etc. Similarly for other instruction and tape-symbol combinations.

($\varepsilon$) If, for $f(n) < i < f(n)^2$, $U(i) \neq 0$, then exactly one of $U(i - f(n)) \neq 0$, or $U(i - f(n) - 1) \neq 0$, or $U(i - f(n) + 1) \neq 0$ holds. Also, if $U(i) \neq 0$, then $U(i \pm 1) = U(i \pm 2) = 0$.

($\zeta$) $U(i) = k$ for some $i$, $0 \leq i < f(n)^2$. If $U(i) = k$ and $i + f(n) < f(n)^2$, then $U(i + f(n)) = k$ and $W(i + f(n)) = W(i)$.

From the assumption that (W, U) satisfies ($\alpha$) - ($\zeta$), it can be proved by induction on $1 \leq j < f(n)$ that $(W_{j+1}, U_{j+1})$ is an instantaneous description which follows from $(W_j, U_j)$ by an application of the instruction $I_{i_j}$ whose number appears in $U_j$. Also, $(W_{f(n)}, U_{f(n)})$ is a halting instantaneous description.

Thus, the existence of a pair (W, U), $W \in \{0, 1\}^*$, $U \in \{0, 1, \ldots, k\}^*$, $|W| = |U| = f(n)^2$, which satisfies ($\alpha$) - ($\zeta$) is a necessary and sufficient condition for the existence of a halting computation C on w with $\ell(C) \leq f(n)$.

Conditions (i) - (v) provide means for making statements about arbitrary (0, 1) sequences of length $f(n)^2$, about integers $0 \leq i < f(n)^2$, and about the integer f(n), all by use of formulas of L of size O(n). Also, the ordinary ordering $\leq$ on N restricted to integers of size less than $f(n)^2$ can be expressed by the length O(n) formula

$$x \leq_n y \leftrightarrow \exists z[I_n(x) \wedge I_n(y) \wedge I_n(z) \wedge x + z = y].$$

Hence, the existence of (W, U) satisfying ($\alpha$) - ($\zeta$) can be expressed by a sentence $F_{AL,w} = F$ with the desired properties (a) - (c). Namely, express 0, 1, ... , k in binary notation by words of equal length $p = |k|$. Then, via $S_n(x, y)$, a single element $a \in A$ exists which codes W, and elements $a_1,\ldots,a_p \in A$ code U. The sentence F will start with quantifiers and relativization:

$$F = \exists x\, \exists x_1 \ldots \exists x_p \;\; \forall y \;\; \forall z \; [I_n(y) \wedge J_n(z) \rightarrow E_\alpha \wedge E_\beta \wedge E_\gamma \wedge E_\delta \wedge E_\varepsilon \wedge E_\zeta].$$

x codes the sequence W and $x_1,\ldots,x_p$ together code the sequence U. The clauses $E_\alpha$ ... $E_\zeta$ express the corresponding conditions ($\alpha$) - ($\zeta$). Thus, for example, $E_\alpha$ is $H_w(x)$; $E_\beta$ is $H_{u_1}(x_1) \wedge H_{u_2}(x_2) \wedge \ldots \wedge H_{u_p}(x_p)$, where $u_1 = 10^{n-1}$ and $u_j = 0^n$, $2 \le j \le p$; and $E_\gamma$ is

$$[\sim S_n(x_1, y) \wedge \ldots \wedge \sim S_n(x_p, y) \wedge I_n(y + z) \rightarrow [S_n(x, y + z) \leftrightarrow S_n(x, y)]].$$

The reader can supply the details of the construction of the remaining expressions $E_\delta$, $E_\varepsilon$ and $E_\zeta$ and verify that, altogether, the $F_{AL,w}$ thus formed satisfies (a) - (c) of Theorem 6. □

4. Proof of Theorem 3 (Real Addition).

We start by showing that for the theory Th($\mathcal{R}$) of real addition, there exist formulas $S_n(x, y)$, $I_n(y)$, etc. as postulated in Theorem 7 with $f(n) = 2^n$, thereby proving Theorem 3. Several of the results in this section will play

later on a role in the proof for PA.

Let F(x, y) be any formula and consider the conjunction

$$G = F(x_1, y_1) \wedge F(x_2, y_2) \wedge F(x_3, y_3).$$

It is readily seen that $G \leftrightarrow G_1$ where

$$G_1 = \forall x\ \forall y\ [((x = x_1 \wedge y = y_1) \vee (x = x_2 \wedge y = y_2) \vee (x = x_3 \wedge y = y_3)) \rightarrow F(x, y)].$$

Note that $|G| = 3 \cdot |F(x, y)|$, whereas $|G_1| = |F(x, y)| + c$, where c is independent of F(x, y). A similar rewriting exists for formulas F with more than two variables and for conjunctions of more than three instances of F. The above device is a special case of a more general theorem due to M. Fischer and A. Meyer. It was discovered independently by several people including V. Strassen.

Theorem 8. There exists a constant $c > 0$ so that for every n there is a formula $M_n(x, y, z)$ of L such that for real numbers A, B, C,

$$M_n(A, B, C) \textit{ is true} \leftrightarrow A \in N \wedge A < 2^{2^n} \wedge AB = C.$$

Also, $|M_n(x, y, z)| \leq c(n+1)$ and $M_n(x, y, z)$ is Turing machine computable from n in time polynomial in n.

Proof. The construction of $M_n(x, y, z)$ will be inductive on n. For $n = 0$ we have $2^{2^0} = 2$ and we define $M_0(x, y, z)$ as

$$[x = 0 \wedge z = 0] \vee [x = 1 \wedge z = y].$$

From $M_k$ we get $M_{k+1}$ by observing that $x \in N$ and $x < 2^{2^{k+1}}$ if and only if

there exist $x_1, x_2, x_3, x_4 \in N$ all less than $2^{2^k}$ so that $x = x_1x_2 + x_3 + x_4$. For this decomposition we have

$$z = xy = x_1(x_2y) + x_3y + x_4y.$$

Hence, $M_{k+1}(x, y, z)$ is equivalent to

$$\exists u_1u_2\ldots u_5x_1\ldots x_4[\ M_k(x_1, x_2, u_1) \wedge M_k(x_2, y, u_2) \wedge M_k(x_1, u_2, u_3)$$
$$\wedge M_k(x_3, y, u_4) \wedge M_k(x_4, y, u_5) \wedge x = u_1 + x_3 + x_4$$
$$\wedge z = u_3 + u_4 + u_5\ ].$$

(Strictly speaking, a triple sum such as $u_1 + x_3 + x_4$ should be written as a chain of sums of two variables, but we shall not do it here.) Now, $|M_{k+1}| \geq 5|M_k|$, which will not do. However, by using the device preceding the theorem, the five occurrences of $M_k$ can be replaced by a single occurrence to yield $M_{k+1}$. Thus, $|M_{k+1}(x, y, z)| \leq |M_k(x, y, z)| + c$ for an appropriate $c > 0$. Hence, $|M_n(x, y, z)| \leq c(n+1)$. (We assume c is chosen large enough so $c \geq |M_0(x, y, z)|$.)

Actually, for the above bound to hold, it is necessary to show that the number of distinct variable names in $M_n$ does not grow with n, for to encode one of v variables requires (on the average) a string of length O(log v). In fact, 15 different variable names are sufficient to express $M_n$. This is because the new variables introduced in constructing $M_{k+1}$ from $M_k$ need only be distinct from each other and from the *free* variables of $M_k$; however no difficulty arises if they coincide with variables *bound* inside $M_k$. A closer look at the construction of $M_{k+1}$ shows that 12 new variables are introduced, which must be distinct from the three free variables of $M_k$, giving a total of 15 distinct names needed. □

Corollary 9. The formula $M_n(x, 0, 0)$ is true for a real number x if and only if $x \in N$ and $x < 2^{2^n}$.

The natural numbers $x < 2^{2^n}$ code all binary sequences of length $2^n$. Namely, write x in binary notation

$$x = x(0) + x(1)\cdot 2 + \ldots + x(2^n-1)\cdot 2^{2^n-1}.$$

We use the function $2^i$ to obtain element x(i) of x.

Theorem 10. There exists a formula $Pow_n(x, y, z)$ such that for integers a, b, c for which $0 \le a, b^a, c < 2^{2^n}$, $Pow_n(a, b, c)$ is true if and only if $b^a = c$. Also, $|Pow_n(x, y, z)| \le d(n+1)$ for an appropriate $d > 0$ and all n.

Proof. Construct, by induction on k, a sequence $E_k(x, y, z, u, v, w)$ of formulas with the property that for integers a, b, c for which $0 \le a < 2^{2^k}$, $0 \le b^a$, $c < 2^{2^n}$ and real numbers A, B, C, $E_k(a, b, c, A, B, C)$ is true in $\langle R, + \rangle$ if and only if $A \in N$, $A < 2^{2^n}$, $b^a = c$, and $AB = C$. Thus, $E_k$ has $M_n$ built into it since

$$E_k(0, 1, 1, A, B, C) \leftrightarrow M_n(A, B, C).$$

The case k = 0 is given by

$$[(x = 0 \wedge z = 1) \vee (x = 1 \wedge z = y)] \wedge M_n(u, v, w).$$

To obtain $E_{k+1}(x, y, z, u, v, w)$ from $E_k$, we again use the decomposition $x = x_1x_2 + x_3 + x_4$ of every integer $0 \le x < 2^{2^{k+1}}$ in terms of integers $0 \le x_1, x_2, x_3, x_4 < 2^{2^k}$. Then we have

$$y^x = (y^{x_1})^{x_2} \cdot y^{x_3} \cdot y^{x_4}.$$

Now, $y^{x_1}$ is expressed by a $z_1$ such that $E_k(x_1, y, z_1, 0, 0, 0)$; then $(y^{x_1})^{x_2}$ is a $z_2$ such that $E_k(x_2, z_1, z_2, 0, 0, 0)$ etc. Whenever we have to write a product such as $x_1x_2$ or $(y^{x_1})^{x_2} \cdot y^{x_3}$, we use the formula $E_k(0, 1, 1, u, v, w)$. In this way we can write the formula $E_{k+1}(x, y, z, u, v, w)$. Using the usual device of contracting a conjunction of instances of $E_k$ into one occurrence, we see that $|E_{k+1}| \leq |E_k| + d$ for some $d > 0$, and hence $|E_n| \leq d(n+1) + c(n+1)$, where $c(n+1)$ is the bound on the length of $M_n$. As before, only a bounded number of variable names are needed.

Recalling the definition of $E_k(x, y, z, u, v, w)$, we see that

$$\mathrm{Pow}_n(x, y, z) \leftrightarrow E_n(x, y, z, 0, 0, 0)$$

has the desired properties. □

Theorem 11. There exists a formula $S_n(x, y)$ of L which for $x, y \in R$ is true in $\langle R, + \rangle$ if and only if x and y are integers, $x < 2^{2^{2n}}$ and $y < 2^{2n}$, and the $y+1^{st}$ digit $x(y)$ of x, counting from the low-order end of the binary representation of x, is 1. The formula $S_n(x, y)$ satisfies the conditions of Theorem 7 for $f(n) = 2^n$.

Proof. That x and y are integers in the appropriate ranges is easily expressible by formulas of size $O(n)$. Recall that for the integers which satisfy $M_{2n}(x, 0, 0)$, i.e. $0 \leq x < 2^{2^{2n}}$, the ordering $\leq$ is expressible by a formula of length $O(n)$.

Now $x(y) = 1$ if and only if there exists an integer $z$, $2^y \le z < 2^{y+1}$ so that $x \ge z$ and $2^{y+1}$ divides $x - z$. This fact is easily expressible by a formula $S_n(x, y)$ of L using $Pow_{2n}$ and $M_{2n}$.

That formulas $I_n(y)$ and $J_n(y)$ with the properties listed in Theorem 7 exist is immediate. Thus to finish the proof of Theorem 3 we need the following.

Theorem 12. For every binary word $w$, $|w| = n$, there exists a formula $H_w(x)$ of L which is true in $\langle R, + \rangle$ for an integer $0 \le x < 2^{2^{2n}}$ if and only if $x(0)\ldots x(2^n-1) = w0^p$, $p = 2^n - n$. The formula $H_w(x)$ satisfies the conditions of Theorem 7.

Proof. Define for binary words $u$, by induction on $|u|$, formulas $K_u(z)$ as follows.

$$K_0(z) \leftrightarrow z = 0,$$

$$K_1(z) \leftrightarrow z = 1,$$

$$K_{u0}(z) \leftrightarrow \exists y[\, K_u(y) \wedge z = y + y \,],$$

$$K_{u1}(z) \leftrightarrow \exists y[\, K_u(y) \wedge z = y + y + 1 \,].$$

Clearly, if $K_w(z)$ is true, then, considered as a sequence, $z$ satisfies $w(i) = z(i)$ for $0 \le i < |w|$, $z(i) = 0$ for $i \ge |w|$. Using this $K_w(z)$ and the formulas $S_n(x, y)$ and $J_n(y)$, we can write the formula $H_w(x)$ by formally expressing the statement that for $z$ such that $K_w(z)$, $x(i) = z(i)$, $0 \le i < 2^n$. □

Thus we have proved, for Th($\mathcal{R}$), the existence of formulas $S_n(x, y)$, $I_n(y)$, $J_n(y)$, and $H_w(x)$ which satisfy the conditions of Theorem 7 for $f(n) = 2^n$. This completes the proof of Theorem 3.

5. Proof of Theorem 4 (Lengths of Proofs for Real Addition).

We now show that for Th($\mathcal{R}$) proofs are also exponentially long. This is an easy consequence of Theorem 3.

Let AX be a consistent system of axioms which is complete for Th($\mathcal{R}$), i.e. every sentence $F \in$ Th($\mathcal{R}$) is provable from AX ($AX \vdash F$). Furthermore, there exists an algorithm B which decides in polynomial time $p(|G|)$ for a sentence G of L whether $G \in AX$.

Let c be the constant of Theorem 3. For every polynomial q(x), there exists a constant $0 < d$ so that from a certain point on, $q(2^{dn}) < 2^{cn}$.

Construct a non-deterministic algorithm AL for Th($\mathcal{R}$) as follows. Given a sentence F, AL writes down (non-deterministically) a binary sequence P. Then AL checks whether P is a proof of F from AX or a proof of ~F from AX. The computation halts only if one of the two possibilities occurs. Because of the assumptions on AX, this check can be made in a polynomial number of steps $h(|P|)$. Thus the whole computation, if it halts, requires $|P| + h(|P|) = q(|P|)$ steps. If every true sentence F would have a proof P with $|P| < 2^{dn}$ where $n = |F|$, then for every such F there would be some halting computation of length less than $q(2^{dn})$, i.e. also less than $2^{cn}$ for all sufficiently large n, a contradiction.

6. Proof of Theorems 1-2 (Presburger Arithmetic).

The proof for Theorem 1 follows closely along the lines of the proof of Theorem 3 and utilizes our previous results. In particular we note that Theorems 8-10 apply, as they stand and with the same proofs, to PA. Note also that the order $\leq$ on N is definable in PA using +. Throughout this section, let f(n) be $2^{2^n}$.

Theorem 12. There exists a function $g(n) \geq 2^{f(n)^2} = 2^{2^{2^{n+1}}}$ so that for every n there exists a formula $\mathrm{Prod}_n(x, y, z)$ with the following properties. For integers A, B, C,

$$\mathrm{Prod}_n(A, B, C) \text{ is true in } \mathfrak{N} \leftrightarrow A, B, C < g(n) \text{ and } AB = C.$$

There exists a constant $c > 0$ so that $|\mathrm{Prod}_n| \leq c(n+1)$ for all n. The formula $\mathrm{Prod}_n$ is Turing machine constructible from n in time polynomial in n.

Proof. We shall use the Prime Number Theorem which says that the number of primes smaller than m is asymptotically equal to $m/\log_e m$; hence bigger than $m/\log_2 m$ for all sufficiently large m. Thus, for $m = 2^{2^{n+2}}$, the number of primes $p < m$ exceeds $2^{2^{n+2}}/2^{n+2} > 2^{2^{n+1}} = f(n)^2$. Let $g(n) = \prod_{p<m} p$, where p runs over primes, $m = 2^{2^{n+2}}$; then $g(n) \geq 2^{f(n)^2}$ since $2 \leq p$ for all primes.

By use of the formula $M_{n+2}(x, y, z)$, we can write two formulas $\mathrm{Res}_{n+2}(x, y, z)$ and $P_{n+2}(x)$ of length O(n) with the following meanings. Let res(x, y) denote the residue (remainder) of x when divided by y. Then

$$\mathrm{Res}_{n+2}(x, y, z) \leftrightarrow [\, y < 2^{2^{n+2}} \wedge \mathrm{res}(x, y) = z \,];$$

$$P_{n+2}(x) \leftrightarrow [\ x < 2^{2^{n+2}} \text{ and } x \text{ is prime }].$$

The formula $\mathrm{Res}_{n+2}$ is written in L as

$$z < y \wedge \exists q\, \exists w[\ M_{n+2}(y, q, w) \wedge x = w + z\ ].$$

We recall that for any q and w, $M_{n+2}(y, q, w)$ holds if and only if $y < 2^{2^{n+2}}$ and $yq = w$.

The formula $P_{n+2}(x)$ is, simply,

$$M_{n+2}(x, 0, 0) \wedge \forall y\, \forall z[\ M_{n+2}(y, z, x) \rightarrow [\ y = 1 \vee y = x\ ]].$$

By formally saying that $x \geq 1$ is the smallest integer divisible by all primes $p < 2^{2^{n+2}}$, we can write a formula $G_{n+2}(x)$ which is true precisely for $x = g(n)$. Now $\mathrm{Prod}_n(x, y, z)$ is true if and only if

$$x, y, z < g(n) \wedge \forall u[\ u < 2^{2^{n+2}} \rightarrow \mathrm{res}(x,u)\cdot\mathrm{res}(y,u) = \mathrm{res}(z,u)\ ]. \quad (3)$$

Namely, this implies that $xy = z \pmod p$ for all $p < 2^{2^{n+2}}$, which together with $x, y, z < g(n)$ is equivalent to $xy = z$. Now, by use of $G_{n+2}(x)$, $M_{n+2}(x, y, z)$ and $\mathrm{Res}_{n+2}(x, y, z)$, the above relation (3) can be expressed by a formula $\mathrm{Prod}_n$ with the desired properties. □

Exponentiation can be defined just as in the proof of Theorem 10 except that we now use $\mathrm{Prod}_n(x, y, z)$ instead of $M_n(x, y, z)$ to obtain a sequence of formulas $E'_k(x, y, z, u, v, w)$. For integers a, b, c, A, B, C for which $0 \leq a < 2^{2^k}$ and $0 \leq b^a$, $c < g(n)$, $E'_k(a, b, c, A, B, C)$ is true in $\mathfrak{N}$ if and only if $A, B, C < g(n)$, $b^a = c$, and $AB = C$. Also $|E'_n| = O(n)$.

Having now multiplication up to g(n) and exponentiation $2^i$ up to $i < 2^{2^{n+1}}$ expressed by formulas of length O(n), we can code sequences of length $2^{2^{n+1}} = f(n)^2$ in exactly the same manner as in Section 4. This completes the proof of Theorem 1 by again appealing to Theorem 7.

The proof of Theorem 2 now follows exactly the lines of the proof of Theorem 4 given in Section 5.

## 7. Other Results.

The techniques presented in this paper for proving lower bounds on logical theories may be extended in a number of directions to yield several other results. We outline some of them below without proof; they will be presented in full in a subsequent paper.

Theorem 13. Let $\mathcal{K}$ be any class of additive structures, so if $\mathcal{A} = \langle A, + \rangle \in \mathcal{K}$, then + is a binary associative operation on A. Let $Th(\mathcal{K})$ be the set of sentences of L valid in every structure of $\mathcal{K}$. Assume $\mathcal{K}$ has the property that for every $k \in N$, there is a structure $\mathcal{A}_k = \langle A_k, + \rangle \in \mathcal{K}$ and an element $u \in A_k$ such that the elements u, u+u, u+u+u, ... , k·u are distinct. Then the statement of Theorem 1 holds for $Th(\mathcal{K})$ with the lower bound $2^{dn}$ for some $d > 0$.

Theorem 3 is an immediate corollary of this result, taking $\mathcal{K}$ to be the class of just the one structure $\mathcal{R} = \langle R, + \rangle$. Some other classes to which the result applies are:

(1) The complex numbers under addition.

(2) Finite cyclic groups.

(3) Rings of characteristic p.

(4) Finite Abelian groups.

(5) The natural numbers under multiplication.

The proof of Theorem 13 extends the ideas of Section 4. The element $n \cdot u$ is used as the representation of the integer n, and u itself is selected by existential quantification.

Special properties of certain theories permit us to obtain still larger lower bounds on the decision problem. For example, we get a lower bound of $2^{2^{cn}}$ for (4), the theory of finite Abelian groups. This is obtained by encoding integers up to $2^{2^n}$ by formulas of length O(n) just as in Theorem 13, but instead of representing a sequence by an integer, we let the structure itself encode the sequence. Let $\mathcal{G}$ be a finite Abelian group. Then element s(i) of the sequence s encoded by $\mathcal{G}$ is 1 if and only if $\mathcal{G}$ contains an element x of order $p_i$, where $p_i$ is the $i+1^{st}$ prime. The necessity of using primes as indices instead of integers considerably complicates the analog of Theorem 7.

Another example where we get still larger bounds is (5), the theory of multiplication of the natural numbers (MULT). That MULT is at least as hard as PA is immediate, for the powers of 2 under multiplication are isomorphic to $\mathcal{N}$, and the property of being a power of 2 can be expressed in MULT (assuming we have the constant 2; otherwise we use an arbitrary prime). In fact, the bound can be increased yet another exponential to $2^{2^{2^{cn}}}$ by using the encoding

which associates a sequence s to a positive integer n, where s(i) = 1 if and only if $q_i$ divides n, where $q_i$ is the $i+1^{st}$ prime in some fixed (but arbitrary) ordering of the primes. Again we are forced to use the primes as indices, and again the analog to Theorem 7 is considerably complicated.

Acknowledgement

The authors gratefully acknowledge several helpful ideas and suggestions of A. Meyer, C. Rackoff, R. Solovay, and V. Strassen which lead to and are incorporated in the present paper.

References

1. J. Ferrante and C. Rackoff, "A decision procedure for the first order theory of real addition with order," Project MAC Technical Memorandum 33, M.I.T., Cambridge, Mass. (May 1973), 16 pp.

2. M. Presburger, "Über die Vollständigkeit eines gewissen Systems der Arithmetik ganzer Zahlen, in welchem die Addition als einzige Operation hervortritt", Comptes-rendus du I Congrès des Mathématiciens des Pays Slaves, Warsaw (1930), pp. 92-101, 395.

3. A. Tarski (with the assistance of J.C.C. McKinsey), A Decision Method for Elementary Algebra and Geometry, 2nd ed., University of California Press, Berkeley and Los Angeles, 1951.

| BIBLIOGRAPHIC DATA SHEET | 1. Report Nos. GJ34671 + N00014-70-A-0362-0003 MAC TM-43 | 2. | 3. Recipient's Accession No. |
|---|---|---|---|
| 4. Title and Subtitle<br>Super-Exponential Complexity of Presburger Arithmetic | | | 5. Report Date : Issued February 1974<br>6. |
| 7. Author(s)<br>Michael J. Fischer and Michael O. Rabin | | | 8. Performing Organization Rept. No. MAC TM-43 |
| 9. Performing Organization Name and Address<br>PROJECT MAC; MASSACHUSETTS INSTITUTE OF TECHNOLOGY:<br>545 Technology Square, Cambridge, Massachusetts 02139 | | | 10. Project/Task/Work Unit No.<br>11. Contract/Grant Nos. GJ34671<br>N00014-70-A-0362-0003 |
| 12. Sponsoring Organization Name and Address<br>Office of Naval Research<br>Department of the Navy<br>Information Systems Program<br>Arlington, Va 22217<br>Associate Program Director<br>Office of Computing Activities<br>National Science Foundation<br>Washington, D. C. 20550 | | | 13. Type of Report & Period Covered : Interim Scientific Report<br>14. |

15. Supplementary Notes

16. Abstracts

Lower bounds are established on the computational complexity of the decision problem and on the inherent lengths of proofs for two classical decidable theories of logic: the first order theory of the real numbers under addition, and Presburger arithmetic -- the first order theory of assition on the natural numbers. There is a fixed constant $c > 0$ such that for every (non-deterministic) decision procedure for determining the truth of sentences of real addition and for all sufficiently large n, there is a sentence of length m for which the decision procedure runs for more than $2^{cn}$ steps. In the case of Presburger arithmetic, the corresponding bound is $2^{2^{cn}}$. These bounds apply also to the minimal lengths of proofs for any complete axiomatization in which the axioms are easily recognized.

17. Key Words and Document Analysis. 17a. Descriptors

17b. Identifiers/Open-Ended Terms

17c. COSATI Field/Group

| 18. Availability Statement | 19. Security Class (This Report) | 21. No. of Pages |
|---|---|---|
| Unlimited Distribution<br>Write Project MAC Publications | UNCLASSIFIED | 25 |
| | 20. Security Class (This Page) UNCLASSIFIED | 22. Price |