JOURNAL OF COMPUTER AND SYSTEM SCIENCES 16, 323–332 (1978)

A $2^{2^{2^{pn}}}$ Upper Bound on the Complexity of Presburger Arithmetic*

DEREK C. OPPEN

Artificial Intelligence Laboratory, Computer Science Department, Stanford University, Stanford, California 94305

Received February 6, 1976; revised September 26, 1977

The decision problem for the theory of integers under addition, or "Presburger Arithmetic," is proved to be elementary recursive in the sense of Kalmar. More precisely, it is proved that a quantifier elimination decision procedure for this theory due to Cooper determines, for any n, the truth of any sentence of length n within deterministic time $2^{2^{2^{pn}}}$ for some constant $p > 1$. This upper bound is approximately one exponential higher than the best known lower bound on nondeterministic time. Since it seems to cost one exponential to simulate a nondeterministic algorithm with a deterministic one, it may not be possible to significantly improve either bound.

INTRODUCTION

In recent years, considerable research has been done on the complexity of decidable theories in logic. The results tend to take two forms. In lower bound worst case results, a theory is proved to have some inherent lower bound on its complexity such that every decision procedure for the theory must exceed this lower bound on some set of input formulas. In upper bound worst case results, a theory is shown to admit a dicision procedure which satisfies some bound on its complexity.

We shall consider one well-known theory, the so-called "Presburger Arithmetic" or theory of integers under addition. This theory has long been known to be decidable for truth [6]. We will analyze a decision procedure for this theory due to Cooper [2] and prove that there is a superexponential upper bound on the size of formula produced by the algorithm when all variables have been eliminated. Thus there is a superexponential bound on the time required to decide the truth of any formula, and the decision problem is elementary recursive in the sense of Kalmar.

More precisely, the superexponential bound on deterministic time required for a sentence of length n is proved to be $2^{2^{2^{pn}}}$ for some constant $p > 1$. This upper bound is approximately one exponential higher than the best known lower bound on nondeterministic time [4]. Since it seems to cost one exponential to simulate a nondeterministic

* An earlier version of part of this paper was presented at the Fifth Annual ACM Conference on the Theory of Computing, May 1973, and appears in the author's Ph. D. Thesis (University of Toronto, 1975). The research was supported by the National Research Council of Canada, the Department of Computer Science, University of Toronto, and the Department of Computer Science, Stanford University, under ARPA Contract DAHC 15-73-C-0435.

0022-0000/78/0163–0323$02.00/0

algorithm with a deterministic one, it may not be possible to significantly improve either bound.

This result contrasts with the complexity results for two related theories. Meyer [5] has shown that the weak monadic second order theory of two successors is not elementary recursive. On the other hand, the set of satisfiable sentences in quantifier-free Presburger arithmetic is NP-complete (see Section 4).

1. Presburger Arithmetic

We will first define $\mathscr{L}^+$, the language of Presburger arithmetic. The symbols of $\mathscr{L}^+$ are (,), $\wedge$, $\vee$, $\neg$, $\forall$, $\exists$, $=$, $<$, $+$, $-$, 0, 1, $x, y, z,\ldots$. $x, y, z,\ldots$ are called *variables* and may be subscripted. An *expression* is a finite sequence of symbols of $\mathscr{L}^+$.

Define a *term* as follows:

1. Variables, 0, and 1 are terms.
2. If t_1 and t_2 are terms, then so are $(t_1 + t_2)$ and $-t_1$.
3. These are the only terms.

An *atomic formula* or atom is an expression of one of the forms $(t_1 < t_2)$ or $(t_1 = t_2)$ where t_1 and t_2 are terms.

A *formula* is defined as follows:

1. An atom is a formula.
2. If A and B are formulas and x is a variable, then $\exists xA$, $\forall xA$, $(A \vee B)$, $(A \wedge B)$, $(A \supset B)$, $(A \equiv B)$, and $\neg A$ are all formulas.
3. These are the only formulas.

A *sentence* is a formula which has no free variables. From now on, we will adopt the usual conventions for parentheses.

The following are examples of sentences in $\mathscr{L}^+$:

$$1 < 1,$$

$$\forall y \exists x \forall z[x + y = z].$$

We define the *standard interpretation* $\mathscr{I}^+$ for $\mathscr{L}^+$ as follows:

1. The domain from which the variables take their values is the set of positive and negative integers.
2. $=, <, +, -$, 0, 1 all take their natural interpretations.

A sentence is either true or false under $\mathscr{I}^+$. Let $\mathscr{T}^+$ be the set of all sentences of $\mathscr{L}^+$ which are true under $\mathscr{I}^+$. For instance, $\forall x[x + 1 = 1 + x]$ is in $\mathscr{T}^+$. $\mathscr{T}^+$ is commonly called the theory of integers under addition or *Presburger arithmetic*. Presburger [6] proved that $\mathscr{T}^+$ is decidable, that is, that $\mathscr{T}^+$ is recursive.

For convenience we will permit the use of certain other symbols in writing formulas in $\mathscr{L}^+$. 2 will stand for $(1 + 1)$, 3 for $((1 + 1) + 1)$, and, in general, k will stand for $1 + \cdots + 1$ (1 repeated k times). $0, 1, 2, \ldots$ are called *numerals*. The term $t_1 + (-t_2)$ will be written as $t_1 - t_2$ and $t + \cdots + t$ (t repeated k times) as kt. We will also use the symbols $\leqslant$, $\simeq$, $\geqslant$, $>$, $|$, and $\nmid$. ($|$ denotes divisibility by a constant, $\nmid$ indivisibility by a constant.) These are all definable in terms of $<$ and $+$ alone. For instance, $x < y \equiv x + 1 \leqslant y$, $k \mid t \equiv \exists x[x + \cdots + x = t]$ where x does not appear in t and is repeated k times, and $k \nmid t \equiv \neg k \mid t$.

2. The Decision Procedure for Truth

A *decision procedure* for truth for Presburger arithmetic is an algorithm which, given a sentence in $\mathscr{L}^+$, decides whether that sentence is true or false under $\mathscr{I}^+$, that is, whether or not it is in $\mathscr{T}^+$. The decision procedure we will now describe is due to Cooper [2] and involves *quantifier elimination*, a general method we now describe. Suppose we have an algorithm P which, given a formula of $\mathscr{L}^+$ of the form $\exists x F(x)$ where $F(x)$ is quantifier free, returns a formula F' of $\mathscr{L}^+$ which is equivalent to $\exists x F(x)$ but which contains no quantifiers. (In other words, P "eliminates" a quantifier.) This suggests the following decision procedure: given any sentence A of $\mathscr{L}^+$, first replace any expression $\forall x$ by the equivalent expression $\neg\exists x\neg$. Apply P to the innermost quantified subformulas of A, replacing subformulas of the form $\exists x F(x)$ by equivalent subformulas without quantifiers. Then apply P to the next level of quantified subformulas. Continue until all quantifiers have been eliminated. The resulting formula contains no variables and therefore evaluates immediately to true or false. We now describe the algorithm for Presburger arithmetic.

Consider a formula of the form $\exists x F(x)$ where $F(x)$ is quantifier free but need not otherwise be in any special form. Assume like terms have been collected. We wish to construct an equivalent formula not containing x. The following algorithm for doing so is due to Cooper [2].

Step 1. Eliminate logical negations by first driving them in as far as possible (using de Morgan's laws, etc.) and then replacing literals consisting of a negated atom by an equivalent unnegated atom. (For instance, replace $\neg(x \neq a)$ by $x = a$.) Replace atomic formulas containing symbols other than $<$, $|$, or $\nmid$ by equivalent formulas containing only $<$ as relation symbol. (For instance, replace $x = a$ by $x < a + 1 \wedge a - 1 < x$.)

Step 2. Let δ' be the least common multiple of the coefficients of x. Multiply both sides of all atoms containing x by appropriate constants so that the coefficients of all the x's are δ'. For instance, if in a particular atom the coefficient of x is k (recall that like terms have been collected), then multiply every constant and the coefficient of every variable in that atom by δ'/k. (By definition, δ' is a multiple of k so δ'/k is an integer.) The coefficient of x in the new atom is δ' and the new atom is equivalent to the original atom. Replace $\exists x F(\delta' x)$ by $\exists x[F(x) \wedge \delta' \mid x]$. Then, each of the atoms of the new $F(x)$ either does not contain x or is of one of the forms:

A. $x < a_i$;
B. $b_i < x$;
C. $\delta_i \mid x + c_i$;
D. $\epsilon_i \nmid x + d_i$;

where a_i, b_i, c_i, and d_i are expressions without x and δ_i and ϵ_i are positive integers.

Step 3. Let δ be the least common multiple of the δ_i and ϵ_i. Let $F_{-\infty}(x)$ be $F(x)$ with "true" substituted for all atoms of type A, and "false" for all atoms of type B. Let $F_{\infty}(x)$ be $F(x)$ with "false" substituted for all atoms of type A and "true" for all atoms of type B. Depending on whether the number of atoms of type A exceeds the number of atoms of type B or not, replace $\exists x F(x)$ by

$$F^{-\infty} \equiv \bigvee_{j=1}^{\delta} F_{-\infty}(j) \vee \bigvee_{j=1}^{\delta} \bigvee_{b_i} F(b_i + j)$$

or

$$F^{\infty} \equiv \bigvee_{j=1}^{\delta} F_{\infty}(-j) \vee \bigvee_{j=1}^{\delta} \bigvee_{a_i} F(a_i - j),$$

respectively. Simplify by collecting like terms.

To illustrate Cooper's algorithm, we will use it to eliminate x from the formula

$$\exists x F(x) \equiv \exists x[2x < 2y + 3z + 5w \wedge 3y + 5z + 2w < 3x \wedge 5x < 5y + 2z + 3w].$$

Step 1 does not change the formula.

In Step 2, $\delta' = 30$, the least common multiple of the coefficients of x. We now multiply both sides of each of the conjuncts of $F(x)$ by, respectively, 15, 10, and 6 so that all the coefficients of x are $\delta' = 30$. This gives us

$$\exists x F'(x) \equiv \exists x[30x < 30y + 45z + 75w \wedge 30y + 50z + 20w < 30x \wedge 30x < 30y + 12z + 18w].$$

We then replace $\exists x F'(x)$ by

$$\exists x F''(x) \equiv \exists x[x < 30y + 45z + 75w \wedge 30y + 50z + 20w < x \wedge x < 30y + 12z + 18w \wedge 30 \mid x].$$

In Step 3, $\delta = 30$. $F_{-\infty}(x) \equiv$ ["true" $\wedge$ "false" $\wedge$ "true" $\wedge$ $30 \mid x$] $\equiv$ "false." So, we replace $\exists x F(x)$ by

$$\begin{aligned} F^{-\infty} &\equiv \bigvee_{j=1}^{30} (\text{“false”}) \vee \bigvee_{j=1}^{30} [30y + 50z + 20w + j < 30y + 45z + 75w \\ &\qquad \wedge 30y + 50z + 20w < 30y + 50z + 20w + j \\ &\qquad \wedge 30y + 50z + 20w + j < 30y + 12z + 18w \\ &\qquad \wedge 30 \mid 30y + 50z + 20w + j] \\ &\equiv \bigvee_{j=1}^{30} [j < -5z + 55w \wedge j < -38z - 2w \wedge 30 \mid 30y + 50z + 20w + j]. \end{aligned}$$

This is a formula that is equivalent to the original one, but which does not contain x.

To justify Cooper's algorithm, we must prove that $F^{-\infty}$ and F^{∞} are both equivalent to the original formula. We will consider only $F^{-\infty}$. Notice first that the new formula $\exists x F(x)$ produced by Steps 1 and 2 is equivalent to the original We therefore wish to show that the new $\exists x F(x) \equiv \bigvee_{j=1}^{\delta} F_{-\infty}(j) \vee \bigvee_{j=1}^{\delta} \bigvee_{b_i} F(b_i + j)$. The following proof appears in [2].

Suppose first that the right-hand side is true. If one of the disjuncts $F(b_i + j)$ is true, then certainly $\exists x F(x)$ is true. Suppose then that $F_{-\infty}(j)$ is true for some j such that $1 \leqslant j \leqslant \delta$. Then, by definition of $F_{-\infty}$, $F_{-\infty}(j - k\delta)$ is true for any positive integer k. In particular, if k is sufficiently large, $F_{-\infty}(j - k\delta) \equiv F(j - k\delta)$ and so again $\exists x F(x)$ is true.

Conversely, suppose $\exists x F(x)$ is true. Let a be such that $F(a)$ is true. Suppose first that a is of the form $b_i + j$ for some b_i and some j where $1 \leqslant j \leqslant \delta$. Then $F(b_i + j)$ is true and the right-hand side is true. Suppose then that a is not of this form and consider $F(a - \delta)$. Suppose $F(a - \delta)$ is false. Then, some atom in F must change from true to false as a changes to $a - \delta$. This can only happen for relations of the form $b_i < x$ (recall that we have eliminated negations), and then only if $a - \delta \leqslant b_i < a$. But then $a = b_i + j$ for some j between 1 and δ. Contradiction. So $F(a - \delta)$ must be true. This argument may be repeated until we reach some $a - k\delta$ such that either $a - k\delta$ is of the form $b_i + j$ or $a - k\delta$ is so small that $F(a - k\delta) \equiv F_{-\infty}(a - k\delta)$. In either case, $F^{-\infty}$ is true.

3. Analysis of Cooper's Algorithm

We are now ready to analyze the algorithm, by first investigating the growth in the size of the formula produced by successive applications of the algorithm. The basic idea in the following analysis is to relate the growth in the number of atoms and the size of the constants appearing in these atoms essentially to the number of distinct coefficients that may appear.

Consider the formula $Q_m x_m Q_{m-1} x_{m-1} \cdots Q_2 x_2 Q_1 x_1 F(x_1, x_2, \ldots, x_m)$ where $Q_1, Q_2, \ldots, Q_m$ are each either $\exists$ or $\forall$ and $F(x_1, x_2, \ldots, x_m)$ is a quantifier-free formula in our language. We apply the algorithm m times to successively eliminate $x_1, x_2, \ldots, x_m$.

Let c_k be the number of distinct positive integers δ_i and ϵ_i appearing in atoms of the form $\delta_i \mid t$ or $\epsilon_i \nmid t$, t a term, plus the number of distinct coefficients of variables in the formula $F_k \equiv Q_m x_m Q_{m-1} x_{m-1} \cdots Q_{k+1} x_{k+1} F_k'(x_{k+1}, \ldots, x_m)$ produced after the kth iteration of the algorithm. Similarly, let s_k be the largest absolute value of the integer constants (including all coefficients) and a_k the total number of atoms in F_k. In particular, let the values of c_0, s_0, and a_0 be c, s, and a, respectively, their values before the algorithm is first applied.

Theorem 1.

$$c_1 \leqslant c^4,$$

$$s_1 \leqslant s^{4c},$$

$$a_1 \leqslant a^4 s^{2c}.$$

Proof. Let a', a'', and a''' be the number of atoms after Steps 1, 2, and 3, respectively of the algorithm, assuming there are a atoms before the algorithm is executed. Similarly define c', c'', c''' and s', s'', s'''.

Step 1. The elimination of logical negations does not affect c', s', nor a' since we do not require the formula to be in any special form. Elimination of relation symbols other than $|$, $\nmid$, or $<$ may double the number of atoms and may increase by one the largest absolute value of those integer constants that do not appear as coefficients of variables. As an illustration of this, note that $x = a$ is replaced by $x < a + 1 \wedge a - 1 < x$. The number of atoms with relation symbols $|$ or $\nmid$ remains at most a:

$$a' \leqslant 2a; \qquad s' \leqslant s + 1; \qquad c' \leqslant c.$$

Step 2. Replacing x by $\delta' x$ may affect the value of s'. The worst case occurs for an atom containing both the term x (with coefficient 1) and the term s'. An example of this is the atom $x < \cdots + s'$. The constant term s' grows to $\delta' s'$, where δ' is the least common multiple of the coefficients of x. Since there are at most c distinct coefficients of x, each of them at most s, $\delta' \leqslant s^c$. Hence $s'' \leqslant s^c s' \leqslant (s + 1)^{c+1}$.

The value of c'' may also be altered. There are at most $c - 1$ variables other than x with coefficients different from any particular coefficient of x and there are are at most c distinct coefficients of x. Hence, c' may grow to at most $c(c - 1)$ plus 1 for the unity coefficient of x after Step 2 plus 1 for the constant δ' occurring in $\delta' \mid x$. As an example of this, consider the effect of Step 2 on the formula $\exists x(2x < 2y + 3z + 5w \wedge 3x < 3y + 5z + 2w \wedge 5x < 5y + 2z + 3w)$; in this case, $c' = 3$ while $c'' = 8$. Thus, $c'' \leqslant c(c - 1) + 2 \leqslant c^2$ for $c > 1$.

Finally, Step 2 increases the number of atoms by 1:

$$a'' \leqslant 2a + 1; \qquad s'' \leqslant (s + 1)^{c+1}; \qquad c'' \leqslant c^2.$$

Step 3. We consider first a'''. The number of atoms in $\bigvee_{j=1}^{\delta} F_{-\infty}(j)$ is at most $\delta(a + 1)$ since all atoms with the relation $<$ simplify to "true" or "false" and there are at most $a + 1$ atoms of the form $\delta_i \mid x + d_i$ or $\epsilon_i \nmid x + e_i$. Now, the number of terms b_i is at most a (in spite of Steps 1 and 2) and there are at most $2a + 1$ atoms in $F(b_i + j)$. Hence the number of atoms in $\bigvee_{j=1}^{\delta} \bigvee_{b_i} F(b_i + j)$ is bounded by $\delta(a)(2a + 1)$ and the number of atoms a''' in $F^{-\infty}$ is at most $\delta(2a^2 + 2a + 1) \leqslant \delta a^4$, for $a > 1$.

We must now find a bound on δ. Each constant δ_i or ϵ_i appearing in atoms of the form $\delta_i \mid x + d_i$ or $\epsilon_i \nmid x + e_i$ is the product of two integers α and β where $\alpha \leqslant s$ and $\beta \mid \delta'$. This follows from Step 2. There are at most c such distinct α so the least common multiple δ of all the δ_i and ϵ_i is at most $s^c\delta'$. Hence, $\delta \leqslant s^{2c}$ and $a''' \leqslant a^4 s^{2c}$.

Simplifying by collecting like terms may affect both s''' and c'''. The largest absolute value of the constants may now be at most $2s'' + 2^{2c} \leqslant 2(s + 1)^{c+1} + s^{2c} \leqslant 3(s + 1)^{2c}$. A similar argument to that given for Step 2 gives a bound of c^4 for c''':

$$a''' \leqslant a^4 s^{2c}; \qquad s''' \leqslant 3(s + 1)^{2c}; \qquad c''' \leqslant c^4.$$

Sufficient for our purposes are the inequalities:

$$a_1 \leqslant a^4 s^{2c}; \qquad s_1 \leqslant s^{4c}; \qquad c_1 \leqslant c^4$$

valid for $s, c > 2$.

THEOREM 2. *If $s, c > 2$ then*

$$c_k \leqslant c^{4^k},$$
$$s_k \leqslant s^{(4c)^{4^k}},$$
$$a_k \leqslant a^{4^k} s^{(4c)^{4^k}}.$$

Proof. The proof follows from the previous theorem by induction on k and the fact that we do not require formulas to be in any special form.

We now prove our main result.

Suppose we are given a sentence of length n encoding $Q_m x_m Q_{m-1} x_{m-1} \cdots Q_1 x_1$ $F(x_1, x_2, \ldots, x_m)$ and wish to find an upper bound on the space required in producing the quantifier-free formula F_m. We can assume $m \leqslant n$, $c \leqslant n$, $a \leqslant n$, $s \leqslant n$. For each k, the space required to store F_k is bounded by the product of the number of atoms a_k in F_k, the maximum number $m+1$ of constants per atom, the maximum amount of space s_k required to store each constant, and some constant q (included for the various arithmetic and logical operators, etc.). That is, the space required to store F_k satisfies:

$$\text{space} \leqslant q \cdot n^{4^n} \cdot n^{(4n)^{4^n}} \cdot (n+1) \cdot n^{(4n)^{4^n}} \leqslant 2^{2^{2^{pn}}}$$

for some constant $p > 1$.

An upper bound on the deterministic time required to test the validity of a sentence in Presburger arithmetic will be dominated by, say, the square of the time required to write out the largest F_k. Thus, the above space bound is also a bound on deterministic time.

We have been assuming that the original formula was of the form $Q_m x_m \cdots Q_1 x_1$ $F(x_1, \ldots, x_m)$, where $F(x_1, \ldots, x_m)$ is quantifier free. However, it is easily verified that an arbitrary formula can be put into this form in linear time and further that the space required by the new formula is bounded by, say, twice the space required by the original. A small modification of the algorithm described here avoids putting the formula in this form and this modification should probably be used in practice since it avoids increasing the scope of the quantifiers. The results proved here hold regardless.

The bounds proved here also hold if we permit other logical connectives such as $\equiv$ and $\supset$. If we eliminate these connectives at the very beginning, the number of occurrences of atoms may of course exponentiate, but none of the other relevant quantities change. That is, if the original formula is of length n, the formula obtained from it by eliminating $\equiv$ and $\supset$ in the usual fashion satisfies $a \leqslant 2^n$, $m \leqslant n$, $c \leqslant n$, $s \leqslant n$. The only change this makes to the expression bounding the space needed to store F_k is in the second term, which now becomes $(2^n)^{4^n}$, but which is still dominated by $2^{2^{2^{pn}}}$.

The upper bound can also be easily shown to hold for the theory of natural numbers under addition, the theory actually considered by Fischer and Rabin [4].

Finally, Rabin [7] has shown that a numeral of length n, in binary notation, is definable in the theory considered here by a formula of length $O(n)$ and hence our upper bound also holds for the theory whose language includes all the numerals rather than just the constants 0 and 1.

4. Observations and Conclusions

The analysis of the previous section has shown that there is a decision procedure for Presburger arithmetic and a constant $p > 0$ such that for all positive integers n and all sentences of $\mathscr{L}^+$ of length n, the decision procedure requires at most $2^{2^{2^{pn}}}$ computational steps to determine the truth of the sentence. This bound also applies to space as well, but Ferrante and Rackoff [3] have shown that one can reduce the upper bound on space by one exponential.

Fischer and Rabin [4] have proved that there exists a constant $c > 0$ such that for every decision procedure for the Presburger arithmetic, there exists an integer n_0 such that for every $n > n_0$ there exists a sentence of $\mathscr{L}^+$ of length n for which the decision procedure requires more than $2^{2^{cn}}$ computational steps to decide the truth of the sentence. Their lower bound also applies to the length of proof required and thus on the space required.

There is a one exponential difference between the lower time bound and upper time bound. Fischer and Rabin's lower bound applies to nondeterministic procedures as well as deterministic ones while our upper bound applies to a deterministic procedure. Since it seems to cost one exponential to simulate a nondeterminstic algorithm by a deterministic one, it will probably be difficult to significantly improve either bound. Any substantial improvement would settle some open questions on the relation between time and space.

Two important points should be made as to why we are able to prove a superexponential bound for Presburger arithmetic.

First, Cooper's algorithm does not require that the formula be put into disjunctive normal form whenever another quantifier is to be eliminated, unlike the case with the better-known decision procedures for Presburger arithmetic. Putting a formula into disjunctive normal form may cause an exponential growth in the size of the formula (consider $(A_1 \vee B_1) \wedge \cdots \wedge (A_n \vee B_n)$). If the original formula contains much alternation of quantifiers, the formula being produced by the standard decision procedure may grow too large to permit a superexponential upper bound. Thus, the time required by the decision procedure may not have a superexponential upper bound (that is, the time may not be bounded by a fixed composition of exponential functions). Cooper's algorithm, as we have shown, does have a superexponential upper bound.

Second, we have related the growth of the size of the formula at each stage in the execution of the algorithm essentially to the number of *distinct* coefficients of variables that may appear, plus the number of *distinct* integers δ_i and ϵ_i appearing in atoms of the form $\delta_i \mid t$ and $\epsilon_i \nmid t$. This point seems essential to the proof: relating the growth to, say, the number of coefficients or even the number of distinct constants does not appear to lead to an elementary bound.

It is interesting to contrast this result with those for two related theories.

Meyer [5] has shown that there cannot exist an elementary recursive decision procedure for the weak monadic second order theory of two successors.

On the other hand, consider the decision problem for satisfiability for quantifier-free formulas in Presburger arithmetic. Borosh and Treybig [1] show that if a system of linear equations with integer coefficients has a solution in nonnegative integers, then it has a solution with all entries bounded by a small polynomial in the maximum of the absolute values of the minors of the associated augmented matrix. It follows that the set of satisfiable systems of linear equalities over the nonnegative integers is in NP. Every inequality $x \geqslant y$ over the nonnegative integers can be represented by the equality $x = y + z$ where z is a new variable to be satisfied over the nonnegative integers. It follows that the set of satisfiable systems of linear inequalities over the nonnegative integers is in NP. Further, every integer can be represented as the difference of two nonnegative integers, and so, by the same trick of introducing new variables to be satisfied over the nonnegative integers, it follows that the set of satisfiable systems of linear inequalities over the integers is in NP. Consider now the disjunctive normal form of any quantifier-free Presburger formula in which all predicate symbols other than $\geqslant$ have been eliminated. Each disjunct is a conjunction of linear inequalities over the integers; further, the length of each disjunct is linear in the size of the original formula. The original formula is satisfiable if and only if one of these disjuncts is satisfiable. We can nondeterministically guess which atomic formulas in the original formula would make up the satisfiable disjunct. It follows that the set of satisfiable quantifier-free formulas of Presburger arithmetic is NP-complete.

Acknowledgments

The author acknowledges with thanks the helpful conversations he has had with Al Borodin, Al Meyer, Michael Rabin, and Larry Stockmeyer concerning aspects of this work. He particularly thanks Stephen Cook for his supervision of most of the research presented here.

References

1. I. Borosh and L. B. Treybig, Bounds on positive integral solutions of linear diophantine equations, *Proc. Amer. Math. Soc.* (1976), 299–304.
2. D. C. Cooper, Theorem proving in arithmetic without multiplication, *Machine Intelligence* **7** (1972), 91–99.
3. J. Ferrante and C. Rackoff, "A Decision Procedure for the First Order Theory of Real Addition with Order," *MAC Technical Memorandum 33*, Massachusetts Institute of Technology, May 1973.
4. M. Fischer and M. Rabin, Super-exponential complexity of Presburger arithmetic, *in* "Proceedings of the Symposium on the Complexity of Real Computation Processes, April 1973."
5. A. Meyer, "Weak Monadic Second Order Theory of Successor is not Elementary Recursive," manuscript, 1972.

6. M. Presburger, Über die Vollstandigkeit eines gewissen Systems der Arithmetik ganzer Zahlen, in welchen die Addition als einzige Operation hervortritt, *in* "Comptes-Rendus du 1er Congrès des Mathématiciens des Pays Slavs, 1929."
7. M. Rabin, private communication. 1972.