TRANSACTIONS OF THE
AMERICAN MATHEMATICAL SOCIETY
Volume 284, Number 1, July 1984

COMPLEXITY OF SUBCASES OF PRESBURGER ARITHMETIC

BY

BRUNO SCARPELLINI

ABSTRACT. We consider formula subclasses of Presburger arithmetic which have a simple structure in one sense or the other and investigate their computational complexity. We also prove some results on the lower bounds of lengths of formulas which are related to questions on quantifier elimination.

1. Introduction. The table on pp. 3, 4 in the monograph [3] of J. Ferrante and C. W. Rachkoff shows that most of the known decidable theories are so complex that they may as well be considered as undecidable. It thus makes sense to investigate subclasses of formulas of such theories which distinguish themselves e.g. by a simple structure or otherwise. The hope is to find a formula class which is decidable in deterministic polynomial time and is of interest because interesting statements can be expressed by formulas of this class. Here we investigate Presburger arithmetic from this point of view. We thereby work with Presburger arithmetic which contains a function symbol ϕ_2, representing multiplication by 2, i.e. $\phi_2(x) = 2x$. The presence of ϕ_2 has no influence on complexity, as far as the theory as a whole is concerned but can make a difference for certain subclasses. Presburger arithmetic is a decidable theory of high complexity, as has been shown by M. J. Fischer and M. O. Rabin in [5]. Subclasses of high complexity are obtained by considering formulas of bounded quantifier alternation; these have been investigated by M. Fürer in [4].

In order to describe some formula classes considered here let M be the set of all quantifierfree formulas. We then fix a list $Q_1,\ldots,Q_s$ of quantifiers and consider classes of formulas of the form:

(1) $(Q_1x_1)\cdots(Q_sx_s)(Ey_1)\cdots(Ey_t)L$, $L \in M$,
(2) $(Q_1x_1)\cdots(Q_sx_s)L$, $L \in M$,
(3) $(Ey_1)\cdots(Ey_t)L$, $L \in M$,
(4) $(Ey_1)\cdots(Ey_k)L$, $L \in M$ *and* k *fixed.*

It turns out that only class (4) is possibly decidable in deterministic polynomial time. Class (3) is NP-complete and for suitable choice of the quantifiers $Q_1,\ldots,Q_s$ class (2) is NP-hard while class (1) admits an exponential lower bound.

In addition to questions of this kind we also consider problems about lengths of formulas of the sort treated by L. Hodes and E. Specker in [8], with the difference

Received by the editors February 25, 1982 and, in revised form, August 5, 1983.
1980 *Mathematics Subject Classification.* Primary 68C01, 68C25; Secondary 03B25, 03F20.
Key words and phrases. Presburger arithmetic, complexity of decidable theories, complexity of formulas, lower bounds for lengths of formulas, NP-hard problems, quantifier elimination.

0002-9947/84 $1.00 + $.25 per page

that here we obtain lower bounds while in [8] upper bounds are derived. For a precise formulation of the problem and the results we refer to §5. The methods used here are not restricted to Presburger arithmetic but apply to other theories of sufficiently high complexity.

2. Preliminaries. In the sequel we consider two versions of Presburger arithmetic. The first contains besides the logical symbols the two place predicate constant $\leqslant$ ("smaller than or equal"), the function symbol + (addition) and the constants 0, 1. The second version contains also the function symbol ϕ_2 denoting multiplication by 2 (i.e. $\phi_2(x) = 2x$). We denote by Th any of these two theories, and when necessary we state explicitly whether Th contains ϕ_2 or not. Equality $=$ and $<$ ("strictly smaller") are of course expressible by $\leqslant$, + and are used as abbreviations only. The standard model for Th is the set N of natural numbers, with $\leqslant$, +, 0, 1 interpreted in the obvious way. A term t without variables has a well-defined natural number as value, denoted by $[t]$. For n a positive integer, bin(n) denotes its binary expansion. Since the language of Presburger arithmetic has to be generated from finitely many symbols we take the expression $x_{\mathrm{bin}(n)}$ as the nth individual variable and abbreviate it by x_n.

3. A class of functions. In order to formulate our results on lengths of formulas it is necessary to introduce a class of functions which behave like polynomials but which are situated between polynomials and the exponential functions. We put $\exp(x) = \exp_1(x) = 2^x$, $\exp_{k+1}(x) = \exp(\exp_k(x))$. We define inductively $\log_1(x) = \log(x)$, $\log_{k+1}(x) = \log(\log_k(x))$, where $\log(x)$ is the logarithm of x with respect to base 2; in order to avoid pathologies we put $\log(1) = 1$.

DEFINITION 1. (1) F_k is the class of functions defined as follows: $f \in F_k$ iff there are integers $p > 1$, $a, b > 0$ such that $f(x) = \exp_k(a(\log_k(x))^p + b)$, (2) $F = \bigcup_k F_k$.

The classes F_k have simple closure and asymptotic properties which due to their elementary character will be stated without proof.

LEMMA 1. (1) $\lim_n f(n)n^{-s} = \infty$ *for all* $s > 0$ *and* $f \in F_1$,

(2) $\lim_n f(n)g(n)^{-1} = 0$ *for all* $f \in F_k$, $g \in F_{k+1}$,

(3) *for all* $f_1, f_2 \in F_k$ *there is a* $g \in F_k$ *with* $f_1(f_2(n))$, $f_1(n)f_2(n)$ *and* $f_1(n) + f_2(n) \leqslant g(n)$ *for all* n,

(4) *the same as* (3) *but with* F *in place of* F_k,

(5) *for every* $f \in F_k$, $a > 0$, $0 < p < 1$ *we have* $\lim_n f(n) \exp(-a \cdot n^p) = 0$. □

In order to get an insight into the growth strength of functions $f \in F_k$ we put $f_1(n) = f(n)$, $f_{m+1}(n) = f_m(f(n))$. One then easily shows that asymptotically $f_n(n) \geqslant \exp_{k+1}(n)$ holds.

4. Some lower bound results. (A) In this section we prove two lower bound results, both concerning subclasses of formulas from Presburger arithmetic. The first of these results is proved because of its independent interest, the second result plays a basic role in the next section.

(B) A formula is called existential if it has the form $(Ey_1) \cdots (Ey_t)A$ with A quantifierfree; the prefix $(Ey_1) \cdots (Ey_t)$ is allowed to be empty. A formula is called

universal if it has the form $(\forall y_1)\cdots(\forall y_t)A$ with A quantifierfree and the prefix possibly empty. The first result is

THEOREM 1. *There is a list $Q_1,\ldots,Q_s$ of quantifiers and an $\varepsilon > 0$ with the property:* (*) *there is no $\gamma 2^{\varepsilon n(\log n)^{-1}}$-bounded nondeterministic Turing machine which accepts precisely the set S of all true closed formulas of the form $(Q_1x_1)\cdots(Q_sx_s)A$, where A is an existential formula.* □

The proof uses a trivialized form of the Fischer-Rabin construction [**5**] and is based on that version of this construction which is described in [**7**]. It is convenient to split the construction into a series of steps (S1)–(S7).

(S1) According to well known time hierarchy theorems [**14**], speed up theorems and the relationship between one tape and multitape machines there is a nondeterministic, 2^{5^n}-time bounded one tape Turing machine T with the property:

(P) there is no nondeterministic, $\gamma 2^n$-time bounded one tape Turing machine T^* which accepts precisely the same binary strings as T.

Henceforth such a machine T is given in a fixed way. Without loss of generality we may assume that the alphabet A of T consists of the three symbols 0, 1, B (blanc). The formulas of Presburger arithmetic are encoded in a familiar way by 0, 1-strings (see e.g. [7] for such an encoding); if G is any such formula we denote its encoding by $\mu(G)$. The encoding is chosen so as to satisfy $|\mu(G)| \leqslant k$ length (G) for suitable k.

(S2) With T at hand we proceed to the announced construction. The idea is simply to replace the predicates $M_n(x, y, z)$ in [**5**] by their "linearized" version, whose inductive definition is as follows:

(a) $M_1(x, y, z)$ is $(x = 0 \wedge z = 0) \vee (x = 1 \wedge y = z)$.

(b) $M_{n+1}(x, y, z)$ is

$$(Eu)(Ev)(M_n(u, y, v)) \wedge ((x = u + u \wedge z = v + v) \vee (x = u + u + 1 \wedge z = v + v + y)).$$

The semantical properties of M_n which follow by induction on n are

(c) $M_n(x, y, z)$ holds iff $x < 2^n \wedge xy = z$. Evidently, only $\wedge$, $\vee$, E are used in the construction of M_n. Next, let P be a three place predicate variable. Let Λ_n be the product of all primes $p < 2^n$. By proceeding as in [7] we construct a formula $F(x, y, z/P)$ whose semantical property is described by

(d) $F(x, y, z/M_n)$ holds iff $x, y, z < \Lambda_n \wedge xy = z$.

(S3) Next we associate with each word $X \in \{0,1\}^*$ a formula $H_X(x)$ with the property:

(e) for a constant term t, $H_X(t)$ is true iff bin($[t]$) = $1X$ (i.e. 1 followed by X).

The definition of H_X is by induction with respect to the length of X and is such that only atomic formulas, $\wedge$ and E are used in its construction. The formulas M_n and H_X satisfy certain inequalities:

(f) length(M_n) $\leqslant a(n + 1)$ and length(H_X) $\leqslant b(|X| + 1)$ for suitable constants a, b.

This is achieved as in [5] by observing a certain economy of variables. That is, a certain sufficiently large stock $y_1,\dots,y_t$ is given once and for all and it causes no difficulties to construct M_n and H_X in such a way that only variables from the stock $y_1,\dots,y_t$ are used. We assume that the set $y_1,\dots,y_t$ is disjoint from the set $x_1,\dots,x_s$ of variables to be used in the next section.

(S4) By proceeding in the same way as in [7] we end up with:

(1) a set $x_1,\dots,x_s$ of variables,

(2) a string $Q_1,\dots,Q_s$ of quantifiers,

(3) two natural numbers δ and τ related to the alphabet of T and the prime number theorem respectively,

(4) a quantifierfree formula $F(P,Q)$ built up from atomic formulas, the three place predicate variable P, the one place predicate variable Q and the individual variables $x_1,\dots,x_s$,

(5) a basic lemma, namely

LEMMA 2. *Let X be a $0,1$-string; let $G(X)$ be the formula $(Q_1x_1)\cdots(Q_sx_s)F(M_{\delta n+\tau}, H_X)$ where $n = |X|$. Then* (1) *the formula $G(X)$ is true iff T has an accepting computation on input X of length $\leqslant 2^{5n}$,*

(2) $\mathrm{length}(G(X)) \leqslant \lambda|X|$ *for a suitable constant* λ,

(3) *there is a deterministic polynomial machine which transforms the input X into $G(X)$.* □

(S5) In order to prove Theorem 1 we have to transform the formulas $G(X)$ into prenex normal form in an appropriate way.

Immediately relevant for the proof of Theorem 1 is

LEMMA 3. *There are formulas $U_n(x, y, z)$, $V_X(x)$ whose properties are:*

(1) *$U_n(x, y, z)$ is equivalent to $\neg M_n(x, y, z)$,*

(2) *$V_X(x)$ is equivalent to $\neg H_X(x)$,*

(3) $\mathrm{length}(U_n) \leqslant a(n+1)$,

(4) $\mathrm{length}(V_X) \leqslant b|X|$ *(for suitable a, b),*

(5) *only $\wedge$, $\vee$, E are used in the construction of U_n, V_X,*

(6) *there are polynomial machines, which, given 1^n and X as input, compute U_n and V_X respectively.*

PROOF. For the construction of U_n we need the auxiliary formula $P_n(x)$, defined by: (a) $P_1(x)$ is $x = 1 + 1$, (b) $P_{n+1}(x)$ is $(Ey)(P_n(y) \wedge y + y = x)$. We then evidently have: (c) $P_n(x)$ holds iff $x = 2^n$. It is clear that a finite set of variables is sufficient for the construction of all formulas P_n and that a constant c exists with $\mathrm{length}(P_n) \leqslant c(n+1)$. Now $\neg M_n(x, y, z)$ holds iff $x \geqslant 2^n \vee (x < 2^n \wedge xy \neq z)$. Thus we can take for U_n the formula

$$(Et)\big(P_n(t) \wedge t \leqslant x\big) \vee (Et)\big(M_n(x, y, t) \wedge (t + 1 \leqslant z \vee z + 1 \leqslant t)\big).$$

It goes without saying that U_n satisfies (1), (3), (5), (6) of the lemma. For $V_X(x)$ we may take the formula

$$(Ey)\big(H_X(y) \wedge (x + 1 \leqslant y \vee y \leqslant x + 1)\big). \quad \square$$

Notation. In connection with Lemma 4 we use provisionally the following notation: if B is $\neg M_n(x, y, z)$ or $\neg H_X(x)$ then B' is $U_n(x, y, z)$ or $V_X(x)$ respectively.

(S6) Lemma 3 enables us to introduce a suitable prenex normal form of $G(X)$. By definition $G(X)$ is the formula $(Q_1x_1)\cdots(Q_sx_s)F(M_{\delta n+\tau}, H_X)$, with $n = |X|$, and with F as in step (S4). Now there is a negationless propositional formula $E(X_1,\ldots,X_a, Y_1,\ldots,Y_b, Z_1,\ldots,Z_c)$, formulas $A_1,\ldots,A_a$ of the form $M_{\delta n+\tau}(x, y, z)$ or $H_X(x)$, formulas $B_1,\ldots,B_b$ of the form $\neg M_{\delta n+\tau}(x, y, z)$ or $\neg H_X(x)$ $(x, y, z \in \{x_1,\ldots,x_s\})$, and formulas $C_1,\ldots,C_c$ which are atomic formulas or negations of atomic formulas such that $E(A_1,\ldots,A_a, B_1,\ldots,B_b, C_1,\ldots,C_c)$ is equivalent to $F(M_{\delta n+\tau}, H_X)$. By Lemma 3 and our notation, $E(A_1,\ldots,A_a, B'_1,\ldots,B'_b, C_1,\ldots,C_c)$ too is equivalent to $F(M_{\delta n+\tau}, H_X)$. Due to the positivity of E and by Lemma 3 we may pass to the prenex normal form by moving all existential quantifiers to the left. The result is a formula $(Ey_1)\cdots(Ey_t)L_X(x_1,\ldots,x_s, y_1,\ldots,y_t)$ with L_X and $y_1,\ldots,y_t$ depending on X, which is still equivalent to $F(M_{\delta n+\tau}, H_X)$. Let $\tilde{G}(X)$ be the formula $(Q_1x_1)\cdots(Q_sx_s)(Ey_1)\cdots(Ey_t)L_X(x_1,\ldots,x_s, y_1,\ldots,y_t)$. The formula $\tilde{G}(X)$ is equivalent to $G(X)$ and it is clear that there is a polynomial machine which transforms the input X into $\tilde{G}(X)$. However, an easy bookkeeping shows that the passage from $G(X)$ to $\tilde{G}(X)$ causes an increase in length by a factor $\log|X|$; this is a consequence of the fact that x_n is an abbreviation for $x_{\mathrm{bin}(n)}$. To sum up, $\tilde{G}(X)$ satisfies (1), (3) of Lemma 2 and in place of (2) the condition

$$(2^*) \qquad \text{length}(\tilde{G}(X)) \leqslant L_0|X|\log|X| \quad \text{for suitable } L_0.$$

From this we infer

$$(2^{**}) \qquad |\mu(\tilde{G}(X))| \leqslant L|X|\log|X| \quad \text{for suitable } L.$$

(S7) We now reproduce the diagonal argument in [7] as it stands. Let $\varepsilon > 0$ satisfy $\varepsilon L < 1$, with L as in (2**) in (S6). Let T' be a $\gamma 2^{\varepsilon n(\log n)^{-1}}$-time bounded nondeterministic machine which accepts precisely those binary strings which have the form $\mu((Q_1x_1)\cdots(Q_sx_s)D)$, where $(Q_1x_1)\cdots(Q_sx_s)D$ is a closed true formula, and where D is an existential formula. Finally, let M_0 be a polynomial machine which transforms an input $X \in \{0,1\}^*$ into $\mu(\tilde{G}(X))$. Let T^* be a new machine which acts as follows: it transforms an input X into $\mu(\tilde{G}(X))$ with the aid of M_0 and then acts upon $\mu(\tilde{G}(X))$ according to the instructions of T'. One then easily verifies:

(a) T^* is $\lambda 2^n$-time bounded for a suitable constant λ,

(b) T^* accepts X iff $\tilde{G}(X)$ holds, that is iff T accepts X.

This however contradicts property (P) of machine T stated in (S1), whence Theorem 1 is proved.

(C) We now come to the second of the mentioned results which is more technical in nature but which is indispensable for the applications in the next section. In order to state it we need some simple syntactical preparations which cannot be avoided since we have to consider a large number of Turing machines over the fixed alphabet $A = \{a_1, a_2, a_3\}$.

(1) We start with the alphabet $A = \{a_1, a_2, a_3\}$, where a_1, a_2, a_3 represent in that order 0, 1 and B respectively. In addition we have three symbols L, N, R (left, neutral, right) which describe the motion of the scanning eye.

(2) Next we fix an integer $n \geqslant 1$ which plays the role of a parameter in that only Turing machine computations of length $\leqslant 2^n$ will be considered. We introduce 2^n states $q_0,\dots,q_{\tau_n}$, $\tau_n = 2^n - 1$. We define a mapping σ which maps all these symbols into $\{0,1\}^*$ as follows: $\sigma(a_j) = 10^j10^{4-j}1$, $\sigma(L) = 10111$, $\sigma(N) = 11011$, $\sigma(R) = 11101$, $\sigma(q_j) = 0^{d_j}\mathrm{bin}(j)$, where $d_j = n - |\mathrm{bin}(j)|$.

(3) A Turing machine T over the alphabet a_1, a_2, a_3 with states among $q_0,\dots,q_{\tau_n}$ is given by the list $\delta_1,\dots,\delta_N$ of its five tuples and will in fact be identified with the list. A five tuple is thereby as usual an expression $\delta = q_ia_jq_sa_td$, $d \in \{L, N, R\}$. With σ we associate the binary string $\sigma(\delta) = \sigma(q_i)\sigma(a_j)\sigma(q_s)\sigma(a_t)\sigma(d)$. We then codify the Turing machines $T = \delta_1,\dots,\delta_N$ by the binary string $\sigma(T) = \sigma(\delta_1)\cdots\sigma(\delta_N)$. If the parameter n is given then it is evident that a binary string ξ can be written in at most one way in the form $\sigma(\delta_1)\cdots\sigma(\delta_N)$, and if so, then $\delta_1,\dots,\delta_N$ can effectively be recovered. We denote by S_n the set of all nondeterministic Turing machines T over a_1, a_2, a_3 with states among $q_0,\dots,q_{\tau_n}$ whose binary encoding $\sigma(T)$ has length $\leqslant 2^{n-4}$.

(4) Next we have to consider the set of all Boolean functions from $\{0,1\}^n$ into $\{0,1\}$; we denote this set by B_n. We encode each $f \in B_n$ by a binary string ζ_f of length $(n+1)2^n$ as follows. Let $\lambda_1,\dots,\lambda_{\tau_n}$ be the list of all binary strings of length n, ordered lexicographically. As ζ_f we take the string $\lambda_0f(\lambda_0)\lambda_1f(\lambda_1)\cdots\lambda_{\tau_n}f(\lambda_{\tau_n})$. Since binary strings of length $(n+1)2^n$ are ordered lexicographically we may speak in an obvious way about the lexicographically smallest or first Boolean function having a certain property.

(5) With every Turing machine $T \in S_n$ we associate a Boolean function $f_T \in B_n$ as follows. For $\lambda \in \{0,1\}^n$ we put $f_T(\lambda) = 1$ iff there is an accepting computation of T on input λ of length $\leqslant 2^n$. Now there is a trivial, but crucial observation. There are at most $2^{2^{n-4}}$ machines in S_n, but 2^{2^n} functions in B_n. Thus there is at least one function $f \in B_n$ which is not of the form f_T, $T \in S_n$. Among these we take the lexicographically smallest one and denote it by f_0. We call f_0 the first Boolean function different from all f_T, $T \in S_n$.

(6) If $2^n \leqslant x < 2^{n+1}$ then $\mathrm{bin}(x) = 1\zeta$ for some $\zeta \in \{0,1\}^n$; we put $\zeta = x^*$. We now can state the second lower bound result.

THEOREM 2. *There is a list $Q_1,\dots,Q_s$ of quantifiers, a constant C and for every integer $n \geqslant 1$ a formula $E_n(x)$ from Presburger arithmetic such that*:

(a) *$E_n(x)$ is true iff $2^n \leqslant x < 2^{n+1}$ and if $f_0(x^*) = 1$, where f_0 is the first Boolean function different from all functions f_T, $f_T \in S_n$,*

(b) *$E_n(x)$ has the form*

$$(Q_1x_1)\cdots(Q_sx_s)(Ey_1)\cdots(Ey_{t_n})G_n(x, x_1,\dots,x_s, y_1,\dots,y_{t_n})$$

where G_n is quantifierfree,

(c) $\mathrm{length}(E_n(x)) \leqslant Cn\log n$,

(d) *there is a polynomial machine, which, given 1^n as input, computes $E_n(x)$.* □

A full proof of Theorem 2, which would amount to a meticulous formalization of the concepts introduced in (1)–(6) is not within the scope of this paper. However, a

closer look quickly shows that the only building blocks needed in order to formalize (1)–(6) are, as in the proof of Theorem 1: (1) the formulas $M_n(x, y, z)$ and $F(x, y, z/P)$ introduced in (S2)(B), (2) two sufficiently large constants δ, τ related to the prime number theorem and the alphabet a_1, a_2, a_3. As in [7] we construct the concatenation predicate $\mathrm{Con}_n(x, y, z)$ with the aid of $F(x, y, z/M_{\delta n+\tau})$, which enables us to concatenate sufficiently large binary strings. We then construct formulas $B(y/P)$, $D(x, y/P)$, $T(y/P)$, $L(x, y/P)$, $W(x, y, z/P)$ with x, y, z individual variables and P a three place predicate variable which have the properties listed below. Let $B_n(y)$, $D_n(x, y)$, $T_n(y)$, $L_n(x, y)$ and $W_n(x, y, z)$ be short for $B(y/M_{\delta n+\tau}), \ldots, W(x, y, z/M_{\delta n+\tau})$ respectively. The properties in question are:

(1) $B_n(y)$ holds iff y is (the encoding of) a Boolean function,

(2) $D_n(x, y)$ holds iff $2^n \leqslant x < 2^{n+1}$ and if y is a Boolean function whose value at x^* is 1,

(3) $T_n(y)$ holds iff y is (the encoding of) a Turing machine in S_n,

(4) $L_n(x, y)$ holds iff x, y are Boolean functions and if x is lexicographically smaller than y,

(5) $W_n(x, y, z)$ holds iff $2^n \leqslant x < 2^{n+1}$, if y is a Turing machine and if z is (the encoding of) an accepting computation of y on input x of time length $\leqslant 2^n$.

Let $H_n(w)$ be the formula

$$B_n(w) \wedge (\forall y)\big(T_n(y) \rightarrow (Ex)\big((D_n(x, w) \wedge \neg(Ez)W_n(x, y, z)) \vee (\neg D_n(x, w) \wedge (Ez)W_n(x, y, z))\big)\big).$$

Obviously $H_n(w)$ expresses that w is a Boolean function which differs from all Boolean functions f_T, $T \in S_n$. The first Boolean function which is different from all Boolean functions f_T, $T \in S_n$ is then described by the formula

$$2^n \leqslant x < 2^{n+1} \wedge (Ew)\big(H_n(w) \wedge D_n(x, w) \wedge (\forall w')\big(L_n(w', w) \rightarrow \neg H_n(w')\big)\big),$$

to be denoted by $E_n'(x)$. Thus $E_n'(x)$ satisfies (a), (b) of Theorem 2 and in addition: (*) length $(E_n'(x)) \leqslant c_0 n$ for suitable c_0. We now proceed with $E_n'(x)$ in the same way as with the formula $G(X)$ in (S5) and (S6)(B) of this section, making thereby use of Lemma 4. We then end up with a formula $E_n(x)$ which now indeed satisfies (a)–(d) of Theorem 2. The construction of the formulas B_n, D_n, T_n, L_n and W_n by means of the concatenation predicate $\mathrm{Con}_n(x, y, z)$ is essentially routine, but there are one or two points which require some attention. We will briefly discuss a typical such point in the appendix. Theorem 2 is more technical than Theorem 1 and therefore less convincing. It would be possible to derive Theorem 1 from Theorem 2, but in a somewhat involved way which will not be reproduced here. The full force of Theorem 2 will become apparent in the next section where it is an indispensable tool for the proof of Theorem 3.

5. On lengths of formulas. For any natural number $p > 0$ let py be short for $y + \cdots + y$ p-times; let x/p denote the formula $(Ey)(x = py)$ ("x is divisible by p"). As is well known [**2**, **9**] one can transform every Presburger formula F into a logically equivalent formula F' which is a negationless Boolean combination of atomic formulas and of formulas of the form t/p (t a term). By moving all quantifier

leftwards one obtains an existential formula G which is logically equivalent to F. However, even if F is universal, G will in general be considerably longer than F. This situation suggests the following quite natural problem: (*) given a universal formula F, find a lower bound to the length of existential formulas G which are logically equivalent to F. Problem (*) arises of course also in connection with other types of formulas. It is in fact with formulas of the form $(\forall x)(Ey_1)\cdots(Ey_t)L$, L quantifierfree, with which our next result is concerned (Theorem 3). An answer to question (*) is then provided for by the corollary to Theorem 3.

THEOREM 3. *Let F be the function class of Definition* 1. *There is an integer $s > 0$ with the following property. There is no function $f \in F$ which satisfies the condition*: (C) *for every formula $(\forall x)L$ with at most s free variables and with L existential, there is a logically equivalent existential formula G such that* $\text{length}(G) \leqslant f(\text{length}((\forall x)L))$. □

The proof of Theorem 3 is based on a fact which will be proved in the last section, namely

FACT 1. The set of all closed existential formulas is NP-complete.

PROOF OF THEOREM 3. We proceed in steps (S1)-(S3).

(S1) To start with, let us assume that a function $h_0 \in F$ exists which has the property (C) of the theorem. Without loss of generality we may assume that the free variables of the formula G which is associated with $(\forall x)L$ according to condition (C) occur among the free variables of $(\forall x)L$. Next let $Q_1,\ldots,Q_s$ be the fixed list of quantifiers which appears in Theorem 2. By applying condition (C) at most s times we infer from the properties of the function class F that there is a function $h_1 \in F$ such that: (C*) for every formulas $(Q_1x_1)\cdots(Q_sx_s)L$ with at most one free variable and with L existential, there is an equivalent existential formula G with

$$\text{length}(G) \leqslant h_1(\text{length}((Q_1x_1)\cdots(Q_sx_s)L)).$$

Again we may assume that if G contains free variables at all then it is the unique free variable in $(Q_1x_1)\cdots(Q_sx_s)L$. Thus by Theorem 2 and condition (C*) there is for every integer $n > 0$ an existential formula $E_n^*(x)$ which is logically equivalent to $E_n(x)$ and such that: (a) $\text{length}(E_n^*(x)) \leqslant h_1(Cn\log n)$.

(S2) Next we use Fact 1. According to this fact there is a nondeterministic polynomial machine M_0 which accepts precisely the closed true existential formulas. We now associate with every existential formula $F(x)$, having only x free, a nondeterministic machine $M(F(x))$ which acts as follows. Given an input $\zeta \in \{0,1\}^*$, $M(F(x))$ first constructs the formula $(Ey)(H_\zeta(y) \wedge F(y))$, where $H_\zeta(y)$ is the formula introduced in step (S3) of §3. Afterwards, $M(F(x))$ transforms this formula into a logically equivalent, closed existential formula F_ζ^* by moving successively all (necessarily existential) quantifiers leftwards. $M(F(x))$ then acts upon F_ζ^* according to the instructions of M_0. Thus $M(F(x))$ accepts ζ iff $(Ey)(H_\zeta(y) \wedge F(y))$ is true. It is only a matter of routine to show that the mapping $F(x) \to M(F(x))$ can be chosen so as to satisfy: (b) there is a polynomial p such that $|\sigma(M(F(x)))| \leqslant p(\text{length}(F(x)))$ (with σ the mapping introduced in (3), (C) of §4), (c) there is a polynomial $q(u,v)$ such that if $M(F(x))$ accepts ζ, then it does so in time $\leqslant q(\text{length}(F(x)), |\zeta|)$.

(S3) We now look at the machine $M(E_n^*(x))$. By combining (a) in (S1) with (b) in (S2) we obtain: (d) $|\sigma(M(E_n^*(x)))| \leqslant p(h_1(Cn \log n))$. From (a) in (S1) and (c) in (S2) on the other hand we obtain: (e) if $M(E_n^*(x))$ accepts a binary string of length n then it does so in nondeterministic time $\leqslant q(h_1(Cn \log n), n)$. Now p, q are polynomial while h_1 belongs to F. From Lemma 1 we infer that there is a function $h \in F$ such that $p(h_1(Cn \log n))$ and $q(h_1(Cn \log n), n)$ both are $\leqslant h(n)$ for all n. Now let n be so large that $h(n) < 2^{n-4}$ (Lemma 1). The machine $M(E_n^*(x))$ then belongs to the set S_n (introduced in (3), (C) of §4), and if it accepts a binary string of length n then it does so in nondeterministic time $< 2^n$. On the other hand, $M(E_n^*(x))$ accepts a string $\zeta \in \{0,1\}^*$ of length n iff $(Ey)(H_\zeta(y) \wedge E_n^*(y))$ is true, that is iff $(Ey)(H_\zeta(y) \wedge E_n(y))$ is true. By Theorem 2, $(Ey)(H_\zeta(y) \wedge E_n(y))$ is true iff $f_0(\zeta) = 1$, where f_0 is the first Boolean function different from all functions f_T, $T \in S_n$. Now let $f^* \in B_n$ be the function with the property: $f^*(\zeta) = 1$ iff $M(E_n^*(x))$ accepts ζ. By the above remarks f^* belongs to the set f_T, $T \in S_n$, and thus $f^* \neq f_0$. From the just mentioned properties of $M(E_n^*(x))$ on the other hand we infer that $f^*(\zeta) = f_0(\zeta)$ holds for all ζ with $|\zeta| = n$, whence $f^* = f_0$ follows, leading to a contradiction. □

COROLLARY. *Let the integer s be as in Theorem* 3. *There is no function $h \in F$ with the following property*: (C′) *for every universal formula G with at most $s + 1$ free variables there is a logically equivalent existential formula L such that* $\text{length}(L) \leqslant h(\text{length}(G))$.

PROOF. Assume that a function h_0 with property (C′) exists. Then there is also a function $f \in F$ with the property: (∗) for every existential formula G' with at most $s + 1$ free variables there is a logically equivalent universal formula L' with length $(L') \leqslant f(\text{length}(G'))$. Without loss of generality we may assume that the free variables of L in (C′) and L' in (∗) occur among the free variables of G and G' respectively. Now consider a formula H of the form $(\forall x)Q$ with at most s free variables and with Q existential. From (∗) and (C′) it follows that there is an existential formula Q' with at most s free variables which is equivalent to H and which satisfies

$$\text{length}(Q') \leqslant h_0(\text{length}(H) + f(\text{length}(H))).$$

By Lemma 1, this contradicts Theorem 3 whence the corollary follows. □

Theorem 3 and its corollary are in the spirit of [**8**], but point in the converse direction. The main open question is how to rise the lower bound, which is presumably exponential. On the other hand we have not fully exploited the technical possibilities contained in the proofs of Theorems 2, 3. By the introduction of alternating machines one should be able to extend Theorem 3 to other classes of formulas.

6. Simple formula classes which are NP-hard. (A) In this section we describe two formula classes of simple structure which are NP-hard. The first is concerned with formulas having a fixed number of quantifiers, the second with formulas of quantifier depth two. These results show that for decidable theories with high complexity NP-hardness already occurs at a very low level.

THEOREM 4. *Let the function symbol ϕ_2 occur among the symbols of Presburger arithmetic. There are quantifiers $Q_1,\dots,Q_s$ such that the set of closed true formulas of the form $(Q_1x_1)\cdots(Q_sx_s)L$, L quantifierfree, is NP-hard.*

PROOF. (S1) We define formulas $M_n(x, y, z)$ as follows:

(a) $M_1(x, y, z)$ is $(x = 0 \wedge z = 0)$,

(b) $M_{n+1}(x, y, z)$ is $M_n(x, y, z) \vee (x = n \wedge ny = z)$, with n and ny short for $1 + \cdots + 1$ and $y + \cdots + y$ n-times respectively. A simple estimate shows that there is a C such that:

(c) length$(M_n) \leqslant Cn^2$. The semantical properties of M_n are:

(d) $M_n(x, y, z)$ holds iff $x < n \wedge xy = z$. Next let Λ_n be the product of all primes $< n$. Let P be a three place predicate variable. As in [**5**] one constructs a formula $F(x, y, z/P)$ with the property:

(e) $F(x, y, z/M_n)$ holds iff $x, y, z < \Lambda_n \wedge xy = z$. Since $\Lambda_n > 2^{kn}$ for some $k > 0$ there is an integer $\delta > 0$ such that $\Lambda_{\delta n} > 2^n$. We thus have:

(f) if $x, y, z \leqslant 2^n$ then $F(x, y, z/M_{\delta n})$ holds iff $xy = z$.

(S2) We note that by Lemma 3 there is a polynomial machine which associates with every integer a, coded in binary, a constant term t_a such that bin$([t_a]) = a$. We now recall a result of Adleman and Manders [**1**]. Let E be the set of triples (a, b, c) of integers > 0 coded in binary, such that the diophantine equation $ax^2 + by = c$ has solution in the set of positive integers. The result then says: $(*)$ E is NP-complete. We now associate with every triple (a, b, c) of integers, coded in binary, a closed formula G^*_{abc}. Let $u, v \cdots \leqslant p$ be short for $u \leqslant p \wedge v \leqslant p \wedge \cdots$; let $F(x, y, z/n)$ be short for $F(x, y, z/M_{\delta n})$. The formula G^*_{abc} then is

$$(Ex, y, u, v, w)\,(x, y, t_a, t_b \leqslant t_c \wedge F(x, t_a, u/|c|) \wedge F(x, u, v/|c|) \wedge F(y, t_b, w/|c|) \wedge v + w = t_c).$$

From the definition of G^*_{abc} and from property (f) of $F(x, y, z/n)$ we infer $(**)$ G^*_{abc} is true iff $(a, b, c) \in E$.

(S3) As a last step we transform G^*_{abc} into its prenex normal form G_{abc}. We then find quantifiers $Q_1,\dots,Q_s$, variables $x_1,\dots,x_s$ not depending on a, b, c and a quantifierfree formula $L_{abc}(x_1,\dots,x_s)$ such that G_{abc} is

$$(Q_1x_1)\cdots(Q_sx_s)L_{abc}(x_1,\dots,x_s).$$

Since the mapping $(a, b, c) \to G_{abc}$ is clearly polynomial, the theorem follows. □

It is not known if Theorem 4 is true for Presburger arithmetic without ϕ_2. As a substitute in this case we have Theorem 5 below which is based on the concept of modular formula.

Let $x \equiv z(n)$ be short for $(Ey)(x = z + ny \vee z = x + ny)$. In order to state Theorem 5 we need

DEFINITION 2. A closed formula F is called modular if there is a negationless propositional formula $H(X_1,\dots,X_s)$, positive integers $n_1,\dots,n_s$ and constant terms $t_1,\dots,t_s$ such that F is $(Ex)H(x \equiv t_1(n_1),\dots,x \equiv t_s(n_s))$.

THEOREM 5. *The set of all true modular formulas is NP-complete.*

PROOF. (S1) For every propositional formula $F(X_1,\dots,X_s)$ we find a negationless propositional formula $P_F(X_1,\dots,X_s, Y_1,\dots,Y_s)$ such that $F(X_1,\dots,X_s)$ is equivalent to $P_F(X_1,\dots,X_s,\neg X_1,\dots,X_s)$. With $F(X_1,\dots,X_s)$ we associate a modular formula as follows. Let $p_1, p_2,\dots$ be the primes in increasing order, let $G_F(x)$ be the formula

$$\bigwedge_{j=1}^{s}\left(x \equiv 0(p_j) \vee x \equiv 1(p_j)\right)$$

$$\wedge P_F(x \equiv 1(p_1),\dots,x \equiv 1(p_s), x \equiv 0(p_1),\dots,x \equiv 0(p_s)).$$

The modular formula associated with $F(X_1,\dots,X_s)$ is $(Ex)G_F(x)$. It is clear that there is a polynomial machine which transforms $F(X_1,\dots,X_s)$ into $(Ex)G_F(x)$.

(S2) The theorem is proved if we can show that $F(X_1,\dots,X_s)$ is satisfiable iff $(Ex)G_F(x)$ is true. Let $F(X_1,\dots,X_s)$ be satisfiable. We then find values $e_1,\dots,e_s \in \{0,1\}$ such that $F(e_1,\dots,e_s)$ and hence $P_F(e_1,\dots,e_s,\bar{e}_1,\dots,\bar{e}_s)$ are true, where $\bar{e}_j = 1 - e_j$, $j = 1,\dots,s$. By the Chinese remainder theorem there is a natural number d such that $d \equiv e_j(p_j), j = 1,\dots,s$. Thus the truth value of $d \equiv 1(p_j)$ is e_j, that of $d \equiv 0(p_j)$ is $\bar{e}_j$. From this, the truth of $G_F(d)$ and hence of $(Ex)G_F(x)$ immediately follows. Conversely, if $G_F(d)$ is true for some d, then there are $e_1,\dots,e_s \in \{0,1\}$ such that $d \equiv e_j(p_j)$ holds from $j = 1,\dots,s$. The truth values of $d \equiv 1(p_j)$ and $d \equiv 0(p_j)$ are then e_j and $\bar{e}_j$ respectively. Thus $P_F(e_1,\dots,e_s,\bar{e}_1,\dots,\bar{e}_s)$ is true and hence $F(X_1,\dots,X_s)$ satisfiable. □

The class of modular formulas seems to be the structurally simplest formula class in Presburger arithmetic without ϕ_2 which exhibits NP-hardness. We have in fact only proved NP-hardness of this class; that it is in NP will be an easy consequence of the results in the next section.

7. The complexity of existential formulas. (A) In this section we investigate the complexity of the set E of closed existential formulas. That E is NP-hard follows immediately from Theorem 5. In order to prove $E \in$ NP we need a result on linear inequalities due to M. Sieveking and J. v. z. Gathen [**12, 13**]. Let $A = (a_{ij})$, $i = 1,\dots,m$, $j = 1,\dots,n$, be an integer matrix and $b = (b_i)$, $i = 1,\dots,m$, an integer vector. Put $c = \max(|a_{ij}|, |b_i|)$, $i = 1,\dots,m, j = 1,\dots,n$; put $r = (n+1)n^{n(n+1)}c^{n^2}$. The result in question then says: (*) if the system $Ax \leqslant b$ (or $Ax \geqslant b$) has a positive integer solution $x = (x_1,\dots,x_n)$, $x_j \geqslant 0$, then it has a positive integer solution $x' = (x'_1,\dots,x'_n)$, $x'_j \geqslant 0$, with $x'_j \leqslant r, j = 1,\dots,n$.

THEOREM 6. (A) *Let* Th *contain* ϕ_2. *Then there is a* $\lambda > 0$ *with the property*: *if* $F = (Ex_1)\cdots(Ex_k)L(x_1,\dots,x_k)$ (L *quantifierfree*) *is a closed true formula, if* $n =$ length(F), *then there are integers* $0 \leqslant d_j \leqslant (k+1)k^{k(k+1)}2^{\lambda nk^2}$, $j = 1,\dots,k$, *such that* $L(d_1,\dots,d_k)$ *is true.*

(B) *Let* Th *not contain* ϕ_2. *Then there is a* $\lambda > 0$ *with the property*: *if* $F = (Ex_1)\cdots(Ex_k)L(x_1,\dots,x_k)$ (L *quantifierfree*) *is a closed true formula, if* $n =$ length(F), *then there are integers* $0 \leqslant d_j \leqslant (k+1)k^{k(k+1)}(\lambda n)^{k^2}$, $j = 1,\dots,k$, *such that* $L(d_1,\dots,d_k)$ *is true.*

PROOF. (A) Let Th contain ϕ_2. Since $\neg a \leqslant b$ is equivalent to $b + 1 \leqslant a$ we may assume that $L(x_1,\dots,x_k)$ is negationaless. Next, by rearranging terms we associate

with every atomic formula $t \leqslant q$ occuring in F an equivalent linear inequality $a_1x_1 + \cdots + a_kx_k \leqslant b$, with $a = (a_1,\ldots,a_k)$ an integer vector and b an integer. It is not difficult to show that there is a $\lambda > 0$, independent of F, such that: (I) $\max(|a_1|,\ldots,|a_k|, |b|) \leqslant 2^{\lambda n}$. Since F is true, there are integers $0 \leqslant d'_j$, $j = 1,\ldots,k$ such that $L(d'_1,\ldots,d'_k)$ is true. Now let $L'(x_1,\ldots,x_k) = D_1 \vee \cdots \vee D_m$ be the disjunctive normal form of $L(x_1,\ldots,x_k)$, with D_j conjunctions of atomic formulas. Since $L(d'_1,\ldots,d'_k)$ is true, there is a D_j, say $t_1 \leqslant q_1 \wedge \cdots \wedge t_N \leqslant q_N$, which is true under the assignment of $d'_1,\ldots,d'_k$ to $x_1,\ldots,x_k$. By our remark above we can replace D_j by an equivalent system of inequalities: (II) $a_1x \leqslant b_1,\ldots,a_Nx \leqslant b_N$, with $a_j = (a_{j1},\ldots,a_{jk})$ integer vectors, b_j integers and with $x = (x_1,\ldots,x_k)$. Thus $d'_1,\ldots,d'_k$ is a solution of system (II). By (I) we have the inequalities: (III) $\max(|a_{ij}|, |b_i|) \leqslant 2^{\lambda n}$, $j = 1,\ldots,k$, $i = 1,\ldots,N$. Thus, by the result (*) of Sieveking-v.z. Gathen there is an integer solution $d_1,\ldots,d_k$ of (II) which satisfies: (IV) $0 \leqslant d_j \leqslant (k+1)k^{k(k+1)}2^{\lambda nk^2}$, $j = 1,\ldots,k$. Since $d_1,\ldots,d_k$ is a solution of (II), D_j is true under the assignment of $d_1,\ldots,d_k$ to $x_1,\ldots,x_k$; thus $L'(d_1,\ldots,d_k)$ and finally $L(d_1,\ldots,d_k)$ are true.

(B) If Th does not contain ϕ_2, inequality (I) in (A) is replaced by: (I*) $\max(|a_{ij}|, |b_i|) \leqslant \lambda n$. Apart from this, the proof is verbally the same. □

REMARK. As has been pointed out by the referee, Theorem 6 is hinted at in [**11**].

COROLLARY 1. *The set E of closed true existential formulas is in NP.*

PROOF. Put $c = \max(\lambda, 1)$. There is a nondeterministic polynomial machine T, which, given a closed formula $F = (Ex_1)\cdots(Ex_k)L(x_1,\ldots,x_k)$ as input (L quantifierfree), "guesses" the binary representation $\text{bin}(d_1),\ldots,\text{bin}(d_k)$ of a positive integer vector $d_1,\ldots,d_k$ such that $|\text{bin}(d_j)| \leqslant 3c(n+1)^3$ (where $n = \text{length}(F)$), and test whether $L(d_1,\ldots,d_k)$ is true or not; it accepts F if $L(d_1,\ldots,d_k)$ is true. By Theorem 6(A), T accepts E. □

COROLLARY 2. *Let* Th *not contain* ϕ_2. *Let* E_k *be the set of closed, true existential formulas* $F = (Ex_1)\cdots(Ex_k)L(x_1,\ldots,x_k)$, L *quantifierfree. Then there is a deterministic polynomial machine which accepts* E_k.

PROOF. Let $[\log x]$ be the smallest integer $\geqslant \log x$. There is a deterministic polynomial machine T_k, which, given a formula $F = (Ex_1)\cdots(Ex_k)L(x_1,\ldots,x_k)$ (L quantifierfree) as input, runs through all positive integer vectors $d_1,\ldots,d_k$ which satisfy $|\text{bin}(d_j)| \leqslant (k+1)^3 + k^2([\log\lambda] + [\log n])$ (where $n = \text{length}(F)$) in order to find one for which $L(d_1,\ldots,d_k)$ is true. If it finds one then it accepts F, otherwise it refutes F. By Theorem 6(B), T_k accepts E_k. □

(B) It remains to discuss the class E_k of Corollary 2 for Th containing ϕ_2. To this end we need a few notions from the theory of convex sets. If $P_0,\ldots,P_N$ are points in Euclidian m-space R^m then $[P_0,\ldots,P_N]$ denotes their convex hull. We call $P_0,\ldots,P_N$ convex independent if $P_j \notin [P_0,\ldots,P_{j-1}, P_{j+1},\ldots,P_N]$, $j = 0,\ldots,N$. The convex set $[P_0,\ldots,P_N]$ is k-dimensional if it is contained in a k-dimensional hyperplane but in no hyperplane of lower dimension. A k-dimensional simplex is a k-dimensional convex set of the form $[P_0,\ldots,P_k]$. A point $P \in R^m$ is called a lattice point if it has

integer coordinates. Below we need

LEMMA 4. *Let* $P_0,\dots,P_N \in R^k$ *be convex independent points and* $[P_0,\dots,P_N]$ *a d-dimensional convex set. Let* $S_1,\dots,S_M$ *be the list of all d-dimensional simplexes which can be formed with points from the list* $P_0,\dots,P_N$. *Then* $[P_0,\dots,P_N] = \bigcup_j S_j$, $1 \leqslant j \leqslant M$. □

The proof is by an elementary induction with respect to N; for lack of space we have to omit it. In what follows, rational numbers γ are coded by ordered pairs $(\pm \mathrm{bin}(a), \mathrm{bin}(b))$, with a, b positive integers ($b > 0$) such that $\gamma = \pm ab^{-1}$. Points $P \in R^k$ are coded as ordered k-tuples of rationals, convex sets $[P_0,\dots,P_N]$ as sequence of points $P_0,\dots,P_N$, and likewise with related concepts. It is then clear what is meant by saying that a Turing machine T is given a simplex S as input. Free use will be made of the fact, proved in [**12**], that all familiar questions on linear systems of diophantine equations such as existence and uniqueness of solutions can be decided by deterministic polynomial machines. Our main result depends on a hypothesis whose status will be discussed after the proof of Theorem 7.

Hypothesis A_k. There is a deterministic polynomial machine T_k which, given a k-dimensional simplex $S = [P_0,\dots,P_k] \subseteq R^k$ as input, acts as follows:

(1) if S contains no lattice points it transforms S into ϕ,

(2) if S contains lattice points it transforms S into a lattice point $P_S \in S$. □

There is a seemingly stronger

Hypothesis A'_k. There is a deterministic polynomial machine T'_k, which, given a d-dimensional simplex $S = [P_0,\dots,P_d] \subseteq R^k$, $(d \leqslant k)$, as input, acts according to (1), (2) in hypothesis A_k. □

It can easily be shown that A_k implies A'_k; we use them interchangeably.

THEOREM 7. *Let* Th *contain* ϕ_2; *let Hypothesis* A_k *be true. Then there is a deterministic polynomial Turing machine* T_k^* *which, given a closed formula* $F = (Ex_1)\cdots(Ex_k)L(x_1,\dots,x_k)$ (*L quantifierfree*) *as input, acts as follows*:

(1) *if F is false then* T_k^* *transforms F into* ϕ,

(2) *if F is true then* T_k^* *transforms F into the binary representation* $(\mathrm{bin}(d_1),\dots,\mathrm{bin}(d_k))$ *of a positive integer vector* $(d_1,\dots,d_k)$ *for which* $L(d_1,\dots,d_k)$ *is true.*

PROOF. We proceed in three steps:

(a) description of an algorithm,

(b) verification that it accomplishes (1), (2) of the theorem,

(c) a brief discussion of its polynomial time character.

(S1) Let $F = (Ex_1)\cdots(Ex_k)L(x_1,\dots,x_k)$, L quantifierfree be given; we may assume that L is negationless. We put $n = \mathrm{length}(F)$ and $C_{kn} = k(k+1)^{k(k+1)}2^{\lambda nk^2}$ with λ as in Theorem 6 and denote by $L^*(x_1,\dots,x_k)$ the formula $\bigwedge_{j=1}^{k} 0 \leqslant x_j \leqslant C_{kn} \wedge L(x_1,\dots,x_k)$. By Theorem 6 the formula F is true iff the formula $F^* = (Ex_1)\cdots(Ex_k)L^*(x_1,\dots,x_k)$ is true. Next let $t_1 \leqslant q_1,\dots,t_N \leqslant q_N$ be the list of all atomic formulas which occur in $L(x_1,\dots,x_k)$. We associate with every formula $t_j \leqslant q_j$ the corresponding equivalent inequality $a_j x \leqslant b_j$, with $a_j = (a_{j1},\dots,a_{jk})$ an integer

vector, b_j an integer and $x = (x_1,\dots,x_k)$. As in the proof of Theorem 6 we have: (I) $|a_{js}|, |b_j| \leqslant 2^{\lambda n}$. We add to the list $a_j x \leqslant b_j, j = 1,\dots,N$ the $2k$ inequalities $0 \leqslant x_j, x_j \leqslant C_{kn}, j = 1,\dots,k$. These inequalities give rise to a sequence of hyperplanes $a_j x = b_j, j = 1,\dots,N, x_j = 0, x_j = C_{kn}, j = 1,\dots,k$, denoted symbolically by $H_1,\dots,H_{N+2k}$. We now consider all possible k-tuples $(H_{i_1},\dots,H_{i_k}), 1 \leqslant i_1 < \cdots < i_k \leqslant N + 2k$ which can be formed with hyperplanes from this list. Let $C_1,\dots,C_a$ be the list of all such k-tuples. Since $N \leqslant n$, necessarily $a \leqslant \binom{n+2k}{k}$. If we intersect all hyperplanes belonging to one and the same k-tuple the intersection can: (1) consist of exactly one point, (2) be empty or contain more than one point. By cancelling all k-tuples from the list $C_1,\dots,C_a$ for which (2) holds, we get the list $C_1',\dots,C_b'$ of k-tuples which satisfy (1). Each C_j' determines the unique intersection point P_j of all hyperplanes in C_j'; we thus obtain the list $P_1,\dots,P_b$ of all such intersection points. From this list we omit superfluous occurences; thus if $P_j = P_k$ for $j < k$ we omit P_k. We also omit those points which do not lie in the cube $0 \leqslant x_j \leqslant C_{kn}, j = 1,\dots,k$. We thus obtain a new list: (II) $P_1',\dots,P_c'$. As next step we construct all simplexes of dimension $d \leqslant k$ whose corners belong to the list (II). To this end we take all $d+1$-tuples $(P_{j_0}', P_{j_1}',\dots,P_{j_d}'), j_0 < j_1 < \cdots < j_d, d \leqslant k$. By retaining those $d+1$-tuples which do describe a d-dimensional simplex and by cancelling all others we obtain the list $S_1,\dots,S_e$ of all simplexes of dimension $d \leqslant k$ whose corners belong to the list (II). Evidently $e \leqslant \sum_{d=0}^{k} \binom{c}{d+1}$. Since $c \leqslant \binom{n+2k}{k}$, there is a τ depending only on k such that $e \leqslant n^\tau$. The last step of the algorithm uses the machine T_k' in A_k'. With the aid of T_k' we cancel those simplexes from the list $S_1,\dots,S_e$ which do not contain lattice points. As a result we obtain the new list: (III) $S_1',\dots,S_g'$ $(g \leqslant n^\tau)$. By using T_k' a second time we determine for every $j \leqslant g$ a necessarily positive lattice point $P_j'' \in S_j'$, thus giving rise to the list of lattice points: (IV) $P_1'',\dots,P_g''$. This terminates the algorithm.

(S2) We now come to (b) of our proof. Let $\zeta_j = (\zeta_{j1},\dots,\zeta_{jk})$ be the coordinate vector of the lattice point P_j'' in (IV); by construction $0 \leqslant \zeta_{js} \leqslant C_{kn}$. We claim: (V) if F^* is true then $L^*(\zeta_{j1},\dots,\zeta_{jk})$ is true for some $j \leqslant g$. In order to verify (V), let $\xi = (\xi_1,\dots,\xi_k)$ be an integer vector such that $L^*(\xi_1,\dots,\xi_k)$ is true. Let $L'(x_1,\dots,x_k)$ be the disjunctive normal form of $L^*(x_1,\dots,x_k)$. Since $L(x_1,\dots,x_k)$ is negationless, $L'(x_1,\dots,x_k)$ is a disjunction of conjunctions of the form $\bigwedge_{j=1}^{k} 0 \leqslant x_j \leqslant C_{kn} \wedge t \leqslant q$. Since $L^*(\xi_1,\dots,\xi_k)$ is true, there is such a conjunction, say

$$D = \bigwedge_{j=1}^{k} 0 \leqslant x_j \leqslant C_{kn} \wedge t_1 \leqslant q_1 \wedge \cdots \wedge t_m \leqslant q_m$$

which is true under the assignment of $\xi_1,\dots,\xi_k$ to $x_1,\dots,x_k$. Let $a_j x \leqslant b_j$ be the equivalent linear inequality associated with $t_j \leqslant q_j, j = 1,\dots,m$. Thus $\xi_1,\dots,\xi_k$ is an integer solution of the system of inequalities: (VI) $0 \leqslant x_j \leqslant C_{kn}, j = 1,\dots,k$, $a_j x \leqslant b_j, j = 1,\dots,m$. Now the set S of points $\in R^k$ which satisfies (VI) is a d-dimensional convex set for some $d \leqslant k$. Let $Q_1,\dots,Q_h$ be the extremal points of S; as is well known S is their convex hull $[Q_1,\dots,Q_h]$. On the other hand, each extremal point Q_i is the unique intersection point of k distinct hyperplanes from the list $x_j = 0, x_j = C_{kn}, j = 1,\dots,k, a_j x = b, j = 1,\dots,m$; furthermore every Q_i lies in

the cube $0 \leqslant x_j \leqslant C_{kn}$, $j = 1,\ldots,k$. Thus every point Q_i occurs in the list (II) of points $P_1',\ldots,P_c'$ constructed in (S1). Let $S_1^*,\ldots,S_p^*$ be the list of all d-dimensional simplexes which have their corners in the list $Q_1,\ldots,Q_h$. According to Lemma 4, $S = \bigcup S_j^*$, and since $(\xi_1,\ldots,\xi_k) \in S$ we have $(\xi_1,\ldots,\xi_k) \in S_j^*$ for some j. Since S_j^* has its corners in the list $P_1',\ldots,P_c'$ and contains at least one lattice point, it must coincide with one of the simplexes $S_1',\ldots,S_g'$ in the list (III) of (S1). Let S_t' denote the simplex in question. We thus have $S_t' \subseteq S$ and $P_t'' \in S_t'$, with P_t'' from (IV) in (S1). Thus $P_t'' \in S$, that is, $\zeta_t = (\zeta_{t1},\ldots,\zeta_{tk})$ is an integer vector which solves system (VI). Hence the conjunction D becomes true under the assignment $\zeta_{t1},\ldots,\zeta_{tk}$ to $x_1,\ldots,x_k$, that is, $L'(\zeta_{t1},\ldots,\zeta_{tk})$ and thus $L^*(\zeta_{t1},\ldots,\zeta_{tk})$ are true, what proves (V). It is now clear how to check the truth of $F = (Ex_1)\cdots(Ex_k)L(x_1,\ldots,x_k)$ with the aid of the algorithm. One constructs the list (IV) of lattice points $P_1'',\ldots,P_g''$ according to the prescriptions of the algorithm, with $(\zeta_{j1},\ldots,\zeta_{jk})$ the coordinate vector of P_j''. One then looks for a $j \leqslant g$ such that $L(\zeta_{j1},\ldots,\zeta_{jk})$ is true; if there is such a j then F is true, otherwise F is false.

(S3) A detailed verification that there is indeed a deterministic polynomial time machine which transforms the formula $F = (Ex_1)\cdots(Ex_k)L(x_1,\ldots,x_k)$ into the list $P_1'',\ldots,P_g''$ of points is quite lengthy and cannot be given here. Nevertheless, a closer inspection shows that it is routine in view of the already mentioned results of Sieveking-v.z. Gathen [**12**] which says that all relevant questions concerning systems of diophantine equations can be solved in deterministic polynomial time. □

COROLLARY. *There is a deterministic polynomial machine which, given a closed formula* $(Ex)L(x)$ (L *quantifierfree*) *as input, decides whether it is true, and if so computes an integer* $\zeta \geqslant 0$ *such that* $L(\zeta)$ *is true.* □

The main problem which remains is the status of Hypothesis A_k, $k \geqslant 2$. Despite many discussions with experts in lattice point theory the truth of A_k could not be decided; to every proposed solution a counterexample was found. Thus the question whether A_k is true for all k remains open.

Appendix. As stated in §4, (C), the construction of B_n, D_n, T_n, L_n and W_n by means of concatenation is basically routine. Nevertheless there are some subtler points: Turing machines of length $\leqslant 2^{n-4}$ require a binary string of length 2^{n-4}, Boolean functions require a string of length $(n+1)2^n$ and computations of length $\leqslant 2^n$ call for a string of length $a2^{2n} + b2^n$, with a, b constants depending only on the alphabet and the encoding. As an example we discuss the third point; the others are treated similarly. A string of length $a2^{2n} + b2^n$ can be constructed as soon as we have strings of length 2^{2n} and 2^n at disposal. We thus describe as a typical case the construction of a string of length 2^{2n}. For x, y binary strings, let xy be their concatenation, $|x|$ the length of x and x^p short for $x\cdots x$ p-times. We define strings x, y, z by means of the following conditions:

(C1) x has the form $(0^d1)^\lambda$ for some $d, \lambda > 0$,

(C2) y admits a "factorization" $y = y_1\cdots y_\lambda$ with $|y_j| = d+1$, $j = 1,\ldots,\lambda$ and such that:

(a) $y_j = \zeta_j 10^p 1$ with $|\zeta_j| = 2n$,

(b) $\zeta_1 = 0^{2^n}$, $\zeta_\lambda = 1^{2^n}$,
(c) ζ_{j+1} is the lexicographical successor of ζ_j,
(C3) $z = z_1 \cdots z_\lambda$, with $|z_j| = d + 1$ for $j = 1, \ldots, \lambda$ and :
(d) $z_j = 1^{a_j}0^{b_j}1$ for some $a_j, b_j > 0$,
(e) $a_{j+1} = a_j + 1$,
(f) $a_1 = 1$, $b_\lambda = 1$.

Evidently x, y, z are uniquely determined by (C1)–(C3), moreover $\lambda = 2^{2^n}$, $d = 2^{2^n} + 1$ and $a_\lambda = 2^{2^n}$. Thus $|x| = |y| = |z| = 2^{2^n}(2^{2^n} + 2) < 2^{5^n}$. If δ, τ in $F(u, v, w/M_{\delta n+\tau})$ are large enough we can handle concatenation by means of $\mathrm{Con}_n(u, v, w)$ for all u, v, w with $|\mathrm{bin}(u)|, |\mathrm{bin}(v)|, |\mathrm{bin}(w)| \leqslant 2^{5^n}$. It is then routine to describe the strings x, y, z, z_λ and $1^{2^{2^n}}$ by means of $\mathrm{Con}_n(u, v, w)$.

ACKNOWLEDGEMENTS. We are indebted for valuable discussions to E. Specker and V. Strassen and also to the referee who helped us to reorganize the paper.

REFERENCES

1. L. Adleman and K. Manders, *NP-complete decision problems for binary quadratics*, J. Comput. System Sci. **16** (1978), 168–184.

2. D. C. Cooper, *Theorem proving in arithmetic without multiplication*, Machine Intelligence Workshop **7** (1972), 91–100.

3. J. Ferrante and C. W. Rackoff, *The computational complexity of logical theories*, Lecture Notes in Math., vol. 718, Springer-Verlag, Berlin and New York, 1979.

4. M. Fürer, *The complexity of Presburger arithmetic with bounded quantifier alternation*, Theoret. Comput. Sci. **18** (1982), 105–111.

5. M. J. Fischer and M. O. Rabin, *Super-exponential complexity of Presburger arithmetic*, Proc. Sympos. Pure Math., vol. 7, Amer. Math. Soc., Providence, R. I., 1974.

6. A. Häussler, *Polynomial beschränkte nichtdeterministische Turingmaschinen und die Vollständigkeit des aussagenlogischen Erfüllungsproblems*, Lecture Notes in Comput. Sci. **43** (1976), 20–35.

7. J. Heintz, *Untere Schranken für die Komplexität logischer Entscheidungs-probleme*. Lecture Notes in Comput. Sci. **43** (1976), 127–137.

8. L. Hodes and E. Specker, *Lengths of formulas and elimination of quantifiers*, Contributions to Math. Logic (1968), 175–188.

9. D. C. Oppen, *Elementary bounds for Presburger arithmetic*, Proc. 5th ACM Sympos. on Theory of Computing, 1973, pp. 34–37.

10. M. Presburger, *Ueber die Vollständigkeit eines gewissen Systems ganzer Zahlen*, Comptes Rendus, I. Congrès des Math. des Pays Slaves, Warsaw, 1929, pp. 192–201.

11. C. R. Reddy and D. W. Loveland, *Presburger arithmetic with bounded quantifier alternation*, ACM. Symposium on Theory of Computing, 1978.

12. M. Sieveking and J. v. z. Gathen, *Weitere zum Erfüllungsproblem polynomial äquivalente kombinatorische Aufgaben*, Lecture Notes in Comput. Sci. **43** (1976), 49–71.

13. _____, *A bound on solutions of linear integer equalities and inequalities*, Proc. Amer. Math. Soc. **72** (1978), 155–158.

14. J. I. Seiferas, M. J. Fischer and A. R. Meyer, *Refinements of nondeterministic time and space hierarchies*, Proc. Fourteenth Annual IEEE Sympos. on Switching and Automata Theory, 1973, pp. 130–137.

MATHEMATICS INSTITUTE OF THE UNIVERSITY OF BASEL, RHEINSPRUNG 21, BASEL 4051, SWITZERLAND