Presburger Arithmetic with Bounded Quantifier Alternation

C. R. Reddy and D. W. Loveland
Duke University
Durham, N.C. 27706

Presburger arithmetic has enjoyed a revival of interest in the last few years. The well-known decision procedure of Presburger [6] for a first-order theory of integer addition was improved by Cooper [2] who applied it within a program verifier. Subsequent attention is being given subdomains of Presburger arithmetic (PA) for the purpose of program verification (e.g. Shostak [7] and Suzuki and Jefferson [8] who concern themselves with extending universal PA by the addition of limited classes of funtions). On a less pragmatic level, attention has been given to the complexity of the decision problem for truth in PA. Fischer and Rabin [4] give a lower bound by establishing that every nondeterministic Turing machine that decides PA requires at least $2^{2^{cn}}$ time to decide the truth of a PA formula of length n, for almost all n. Oppen [5] showns that the Cooper version of the Presburger algorithm (the Cooper-Presburger algorithm) yields a deterministic time bound of

$$2^{2^{2^{cn \lg n}}}$$

for deciding the truth of PA formulas of length n, for sufficiently large n and a suitable $c > 0$. Ferrante and Rackoff [3] show that the space requirements for a suitable deterministic decision procedure for PA is one exponent lower than Oppen's time bound. Given our present knowledge of various complexity trade-offs these upper bounds appear sharp.

This paper concerns both the complexity aspects of PA and the pragmatics of improving algorithms for dealing with restricted subcases of PA for uses such as program verification. We improve the Cooper-Presburger decision procedure and show that the improved version permits us to obtain time and space upper bounds for PA classes restricted to a bounded number of alternations of quantifiers. The improvement is one exponent less than the upper bounds for the decision problem for full PA. The pragmatists not interested in complexity bounds can read the results as substantiation of the intuitive feeling that the improvement to the Cooper-Presburger algorithm is a real, rather than ineffectual, improvement. (It can be easily shown that the bounds obtained here are not achievable using the Cooper-Presburger procedure).

The restriction of permitting only a fixed number of quantifiers is a natural one. Mathematicians have always found that statements become significantly more difficult to understand with each alternation of quantifiers. (Formal hierarchies such as the arithmetic hierarchy support this.) Undoubtedly as a consequence of this observed increase in complexity, we rarely, if ever, deal in formulas with more than a few alternations of quantifiers, while we are relatively casual about the introduction of quantifiers of the same kind. Our results reflect this: the restricted classes limit the number of alternations but not the total number of quantifiers.

Presburger Arithmetic

We take Presburger Arithmetic (PA) to be the first-order theory of linear inequalities over the integers. Specifically, the language of PA is a first-order language over the non-logical alphabet 0,1,+,-,≤, using the following atomic symbols:

b,-b; one binary string b for each non-negative integer	constant symbols,
xb; one binary string b for each non-negative integer	variables,
+	function symbol,
≤	predicate symbol.

Terms have the form $a_1xb_1 + \ldots + a_nxb_n + a_{n+1}$, for $n \geq 0$, where the a_i are constant symbols and xb_i are variables. (We do not formalize the limited concatenation function used here.) If an a_i is of the form -b, we optionally employ parentheses. Typical terms are 0 and (-1)x0 + (101x11 + (-1011)). (Had we employed decimal notation instead of binary notation, our results would change in very minor ways only.)

Atomic formulas have the form $t_1 \leq t_2$ where t_1 and t_2 are terms. We also include the boolean values TRUE and FALSE as atomic formulas. The set L_{PA} of formulas over this language is defined inductively from these atomic formulas in the usual

This research was supported by National Science Foundation Grant DCR75-00666

way, using connectives $\sim, \wedge, \vee$ and quantifiers $\forall$ and $\exists$. (Of course, connectives $\supset$ and $\equiv$ can be added as primitives; we prefer to consider them as definable functions.)

The intended interpretation of the formula is as suggested by the notation, with the domain of interpretation the set of all integers. We denote by PA the set of true formulas in the set L_{PA} of formulas defined above.

The Cooper-Presburger algorithm takes two relations definable in PA as primitive (in addition to the basic $\leq$ relation) and we do likewise. The relations are "k divides t" and "k does not divide t", written $k|t$ and $k\not|t$ respectively, where k is a positive integer (technically, the binary representation for the integer) and t is a term. We define $k|t$ as $\exists x(t=kx)$ where $t_1=t_2$ denotes $(t_1\leq t_2) \wedge (t_2\leq t_1)$. $k\not|t$ is defined as $\sim(k|t)$. We also adopt the shorthand t_1-t_2 for $t_1+(-t_2)$. The use of the defined relations = and < in stating the input formula is also possible without affecting our results; we choose to exclude their use for simplicity.

Other formulations of Peano arithmetic are common. Often the domain of the interpretation is the non-negative integers; this is achievable here by adding $0\leq x$ to the input formula and forbidding the negation function. Usually the term ax, where a is the binary representation of an integer k and x is a variable, is not permitted; rather the equivlent term x+x+...+x (k times) is used. This alternative is significantly different because ax has length approximately log k (the log function is of base two throughout) whereas kx denotes a length k formula. However, it is possible to express kx by an O(log k) formula in a language disallowing terms ax. We illustrate the idea by a formula that expresses y=11x:

$$\exists w(y=w+w+x \wedge \exists y(w=y+y+x \wedge y=x+x))$$

(It is important that variable reuse occur so that only a fixed number of variables are needed for all kx formulas; see Fischer and Rabin [4].) Thus our results apply to formulations of PA excluding terms of the form ax.

The Modified Presburger Algorithm

As previously stated, the algorithm we give here is an improvement of Cooper's version of the Presburger algorithm, itself a definite improvement over the original formulation. The algorithm decides the truth of formulas in PA presented in prenex normal form (all quantifiers leftmost) by the process of the elimination of quantifiers, proceeding from the innermost quantifier outward. By this we mean that an equivalent formula is found that does not contain the quantified variable being eliminated. When all quantifiers are eliminated, the formula is variable-free and can be checked rapidly for truth. Thus we need give only the process to eliminate each quantifier.

We assume that the formula is of the form $\exists xF(x)$ or $\forall xF(x)$ with F(x) quantifier-free, and further assume that all "like terms" are collected in atomic formulas, e.g. replace $x-4\leq 2x-2$ by $-2\leq x$. The letters a,b,c and d represent terms not containing x; α,δ and ε are positive integers and j is a non-negative integer. The expression $(b+j)/\alpha x$ indicates that term b+j replaces αx at every occurrence of αx in the formula; however, if the number of occurrences of x in a term is not a multiple of α, both sides of the relation are multiplied by α and the replacement is made.

We now give the modified Cooper-Presburger algorithm.

Step 1. If the given formula is $\forall xF(x)$, replace $\forall xF(x)$ by $\sim\exists x\sim F(x)$.

Step 2. Move all negation symbols that are to the right of every quantifier inward as far as possible (using de Morgan's laws) and adjust atoms so as to have only the following forms as atoms (in particular, replace $\sim(t_1\leq t_2)$ by $t_2+1\leq t_1$):

(i) $\alpha x\leq a$;

(ii) $b\leq\alpha x$;

(iii) $\delta|(\alpha x+c)$;

(iv) $\varepsilon\not|(\alpha x+d)$.

The resultant formula is $\exists xF'(x)$.

Step 3. We define $F_{-\infty}(x)$ as the formula F(x) with all type (i) atoms of Step 2 replaced by TRUE and all type (ii) atoms of Step 2 replace by FALSE. Replace $\exists xF'(x)$ by

(1) $$\exists jF_{-\infty}(j) \vee \bigvee_b \exists j[F'((b+j)/\alpha x) \wedge 0\leq j \wedge \alpha|(b+j)]$$

Step 4. Let σ be the least common multiple of all δ and ε in atoms $\delta|t$ and $\varepsilon\not|t$ containing the variable j. Replace the formula obtained at Step 3 by

(2) $$\bigvee_{j=0}^{\sigma-1} F'_{-\infty}(j) \vee \bigvee_b \bigvee_{j=0}^{\sigma-1} [F'((b+j)/\alpha x) \wedge \alpha|(b+j)].$$

End of the procedure.

In lines (1) and (2) above the disjunction $\bigvee_b$ is taken over all terms b in atoms of type (ii) of Step 2.

Here, as for the Cooper-Presburger procedure, there is a dual form

(3) $$\bigvee_{j=0}^{\sigma-1} F'(-j) \vee \bigvee_a \bigvee_{j=0}^{\sigma-1} [F'((a-j)/\alpha x) \wedge \alpha|(a-j)]$$

where $F_\infty(x)$ is as $F_{-\infty}(x)$ with TRUE and FALSE interchanged. In implementation (3) should be used in place of (2) whenever the atoms of type (i) are

fewer in number than the atoms of type (ii). For simplicity of argument we will assume we eliminate the quantified variable x via (2) only; it is easily seen that a selection of (2) or (3) does not effect our results.

The distinction between the algorithm presented in Cooper [2] and the algorithm presented here is merely that we choose to normalize the coefficients of x in F(x) only after an atom $b \le \alpha x$ has been selected to generate the replacement b+j for αx throughout F(x). By such a postponement we do not need to find the least common multiple ρ of all the coefficients of x in F(x), an event of explosive potential on the magnitude of ρ if there exist many atoms of type $b \le \alpha x$ (and $\alpha x \le a$) all with relatively prime coefficients of x. We need only demand that $\alpha \mid (b+j)$ rather than that $\rho \mid (b+j)$. This trick gains one nothing if the different α's subsequently are all multiplied together as apparently occurs at Step 4. However, in processing a sequence of eliminations of $\exists$'s only, the quantifiers can be imported inside the disjunctions created by preceding quantifier elimination, thus making it unnecessary to multiply all the coefficients together simply because they occur also as δ's in $\delta \mid t$ atoms somewhere in the processed formula. Clearly, the need to eliminate a $\forall$ quantifier following a string of type $\exists\exists\ldots\exists$ upsets this since the negative symbol imported after conversion to $\sim\exists\sim$ leaves the $\exists$ facing a conjunction of formulas. Thus the alternation of quantifiers is where we take our lumps, with strong consequence to the size of our constants, an undesirable situation as we shall see.

Since the modification we make simply alters the timing of the normalization of coefficients of x in effect, the justification of our modification closely follows the justification for the Cooper-Presburger algorithm given in Cooper [2], so is not given here. However, the consequence of this modification is to provide results not otherwise obtainable. We consider these results now.

Bounds on Constants Size

Let $L_{PA}(m)$, $m \ge 0$, denote the set of all closed formulas of L_{PA} in prenex normal form with no more than m alternations of quantifiers. Let PA(m) $m \ge 0$, be the set of all formulas of $L_{PA}(m)$ true under the intended interpretation.

Our goal is to obtain space and time bounds on the (worst-case) computation effort to determine of formulas of $L_{PA}(m)$ if they are in PA(m). The first step in this direction is to obtain an upper bound on the magnitude of the coefficients of variables and constant subterms that can occur in a formula after all quantifier elimination has occurred. We then follow the method of Ferrante and Rackoff [3] and use these bounds to determine bounds on variable substitution which leads to the desired space bound.

If for integers j and k we have $|j| \le k$ we will say that j is limited by k.

In the theorem that follows, the m=0 case is of little interest since better results are known; see e.g. Borosh and Treybig [1].

Theorem If a formula of length n in PA(m), $m \ge 0$, is processed by the modified Cooper-Presburger algorithm, then in the resultant formula

a) the largest coefficient is limited by

$$2^n(2^n)^{2^n}$$

b) the largest least common multiple σ (from Step 4) is limited by

$$2^{2^{cn^{m+2}}} \quad ; \text{ and}$$

c) the largest integer encountered is limited by

$$2^{2^{cn^{m+3}}}$$

for some c>0 and all n>4.

Proof. Processing a formula by the algorithm given earlier means iterative elimination of quantifiers $Q_1x_1, \ldots, Q_px_p$, in this order, from $Q_px_p \ldots Q_1x_1F(x_1,\ldots,x_p)$, leaving a variable-free formula. Here each Q_i is either $\exists$ or $\forall$. We obtain bounds on the appropriate constants at each iteration and then obtain our desired result. Let γ_k denote the magnitude of the largest coefficient of any variable in the formula after the elimination of k quantifiers. Coefficients are altered as a result of Step 3 only; they are altered by the substitution of b+j for αx, with b and α determined by the term $b \le \alpha x$. To execute the substitution, the terms of the receiving inequality or divisibility relation are multiplied by α(or 1), the substitution for αx made, and like elements collected. Because receiving relations have only one occurrence of a βx term, at most one new occurrence of each variable is introduced upon substitution, so at most two like terms are combined upon collecting terms. Thus,

$$\gamma_{k+1} \le 2\gamma_k^2 .$$

This holds regardless of alternations of quantifiers. From Step 3 we observe that the δ and ε constants also are bounded by γ_k after k quantifiers have been eliminated.

If $\gamma_0 \le 2^n$ then we see that

$$\gamma_n \le 2^n(2^n)^{2^n}$$

Two inequalities of form $\alpha x \le a$ or of form $b \le \alpha x$ are coefficient distinct iff there is at least one variable with different coefficients in the two inequalities. Thus two inequalities not coefficient distinct differ only in the additive constant. Let d_j^i denote the number of coefficient distinct inequalities that exist after i alternations of eliminated quantifiers and j eliminations of quantifiers after the last alternation. Let $q=d_0^0$, the intial number of coefficient distinct

inequalities. It is the number of coefficient distinct inequalities rather than the total number of inequalities that determines the growth rate of σ and consequently the growth rate of the additive constants in the linear terms. Let ℓ denote the maximum number of quantifiers between alternations. Between alternations of quantifiers we effectively have a string of existential quantifiers (the negations between universal quantifiers cancel out) and no negation symbol is imported to alter the the disjunctions introduced by successive elimination. This allows the $\exists x$ to be imported inside all the disjunctions, so that each $\exists x$ has as its scope a formula containing no more inequalities than the formula that exists immediately following the previous alternation. We now determine the d_i^0, $1 \le i \le \ell$.

$$d_0^0 = q, \quad \text{by definition;}$$

then

$$d_1^0 \le q^2.$$

This follows because Step 3 generates at most q disjunctions of form $\exists jG$, where G contains at most q inequalities. Thus q^2 coefficient distinct inequalities might exist at this stage. Step 4 generates no new coefficient distinct inequalities.

Repeating this argument, we get

$$d_\ell^0 \le q^{\ell+1}.$$

When an alternation occurs a negation symbol appears that must be moved inward. Since the $\exists$ then cannot be imported, we must view the string to the right of the $\exists$ as a single entity and begin as if at the beginning but with a larger formula. Thus,

$$d_0^1 \le q^{\ell+1},$$

$$d_1^1 \le (d_1^1)^2$$

and

$$d_\ell^1 \le (d_1^1)^{\ell+1} = q^{(\ell+1)^2} = d_1^2.$$

Repeating this, we finally obtain

$$d_m^\ell \le (d_1^m)^{\ell+1} = q^{(\ell+1)^{m+1}}.$$

We now want a bound for σ at each quantifier elimination step. Our modification of the Presburger algorithm pays off here because each δ and ε is bounded by γ_k. We can bound the number of distinct δ's and ε's by the suitable d_j^i. It suffices to bound σ uniformly, so in the following computation we bound q by n, γ_0 by 2^n, and the number of quantifier removals also by n, where n is the input string length.

$$\text{Max } \sigma \le (\text{max. coef. size})^{\text{no. of distinct coef.ineq.}}$$

$$\le [\gamma_n]^{d_\ell^m} \le [2^n (2^n)^{2^n}]^{n^{(\ell+1)^{m+1}}}$$

$$\le 2^{2^{cn^{m+2}}}, \text{ for some } c > 0 \text{ and all } n \ge 2.$$

To estimate the largest integer encountered in the formula after k quantifiers have been removed we must estimate the additive constant and the coefficients. During a substitution an existing additive constant can be multiplied by a coefficient and added to the constant from the substitution term. If S_k bounds the additive constants after k eliminations, then

$$S_{k+1} \le 2\gamma_k S_k + \sigma_{k+1}$$

bounds the additive constants after k+1 eliminations. If also $\gamma_k \le S_k$ then S_k bounds the additive constants and the coefficients after k elimination and thus bounds all integers encountered. We note that

$$S_k = 2^{2^{ckn^{m+2}}}$$

satisfies the recurrence equation above, specifically

$$2^{2^{c(k+1)n^{m+2}}} \le 2[2^k (2^n)^{2^k}] 2^{2^{ckn^{m+2}}} + 2^{2^{cn^{m+2}}}$$

for all $n > 4$, for some $c > 0$.

Finally, we determine S_n, a bound on all integers encountered after the elimination of all quantifiers. Since S_n is S_k for k=n, we have

$$S_n \le 2^{2^{cn^{m+3}}}$$

Space and Time Bounds

In order to get the desired space bound and the related time bound for the determination of PA(m) we use the bound of the previous section to determine an upper bound on the range of the quantified variables, so that the unbounded quantifiers can be replaced by bounded quantifiers. To determine if a formula is in PA(m) it then is sufficient to sequentially test (at worst) all possible variable substitutions, in each instance checking a variable-free formula. The space required is the space needed to write out the longest formula, a function of the size of the integers replacing the variables. The time needed to execute this enumeration is then bounded in the traditional way.

We prove two theorems needed for our result. We let Qx denote either $\exists x$ or $\forall x$, and let $Qx \le p$

denote either $\exists x \le p$ or $\forall x \le p$, where $\exists x \le p$ indicates "there exists an x limited by p" and $\forall x \le p$ denotes "for all x limited by p". The two theorems are variations of theorems of Ferrante and Rackoff [3].

Theorem. If $QxF(x, x_1,\ldots,x_k)$ is a formula of length n whose universal or existential closure is in $L_{PA}(m)$ and if integers $n_1,\ldots,n_k$ are bounded by w, then $QxF(x,n_1,\ldots,n_k)$ is true iff

$$(Qx \le (k\cdot w+1)2^{2^{(c+1)n^{m+3}}})F(x,n_1,\ldots,n_k)$$

is true for some c>0 and all n>4.

Proof. We make use of the bounds established in the previous section; in fact, the constant c may be taken to be the value obtained there. We assume first that the quantifier Qx is $\exists x$.

If $\exists xF(x,n_1,\ldots,n_k)$ is false then $(\exists x \le p)F(x,n_1,\ldots,n_k)$ is false for any p>0 we choose.

If $\exists xF(x,n_1,\ldots,n_k)$ is true then we must establish that $(\exists x \le p)F(x,n_1,\ldots,n_k)$ is true for the value p stated in the theorem.

Let F' be the formula obtained by removal of the quantifiers of F using the quantifier elimination procedure given earlier. Using earlier notation, we note that the largest coefficient of F' is limited by γ_k, in turn bounded by γ_n. The largest constant in F' is bounded by S_n. Because of the truth-preservation property of the quantifier-elimination procedure, we know for all x that $F(x,n_1,\ldots n_k)$ iff $F'(x,n_1,\ldots,n_k)$, so we may seek the bound for $(\exists x \le p)F'(x,n_1,\ldots,n_k)$. But $\exists xF'(x,n_1,\ldots,n_k)$ implies by the validity of the quantifier elimination process that either $F'_{-\infty}(j,n_1,\ldots,n_k)$ or $F'((b+j)/\alpha, n_1,\ldots,n_k)$ with b and α determined by some atom $b \le \alpha x$ in F' and $0 \le j \le \sigma$, where σ is the l.c.m. of certain δ's and ε's in F'. (Recall that one purpose of variable j is to make $(b+j)/\alpha$ an integer.) Since $\alpha \ge 1$ we bound $(b+j)/\alpha$ by b+j. The term b is of the form $\sum_{j=1}^{k} a_i x_i + e$ with coefficients a_i limited by γ_n and constant e limited by S_n. Thus, using results from the previous theorem we have

$$\max(b+j) \le k\cdot\gamma_n \cdot w+S_n+\sigma_{max}$$

$$\le k\cdot 2^n(2^n)^{2^n}\cdot w+2^{2^{cn^{m+3}}+2^{cn^{m+2}}}$$

$$\le (kw+1)2^{2^{(c+1)n^{m+3}}} = p.$$

For this limit p we know there is an x limited by p such that $F'(x,n_1,\ldots,n_k)$ is true if $\exists xF'(x,n_1,\ldots,n_k)$ unless the latter holds because $F'_{-\infty}(j,n_1,\ldots,n_k)$ holds. But the truth of $F'_{-\infty}(j,n_1,\ldots,n_k)$ means that $F'(x,n_1,\ldots,n_k)$ is true for any x smaller than all values a/α from atoms $\alpha x \le a$ and all b/α from atoms $b \le \alpha x$ such that appropriate divisibility and non-divisibility conditions are satisfied. Such an x is limited by the same bound as computed above because the a terms have the same bounds as the b terms. That is,

$$|\min(a-j)| \le k\cdot\gamma_n\cdot w+S_n+\sigma_{max} = \text{bound}[\max(b+j)].$$

Finally, we observe that if Qx is $\forall x$ the same bound for the quantifier holds because if $\forall xF(x,n_1,\ldots,n_k)$ is false then we can show by the above argument that there exists an x_0 limited by p such that $\sim F(x_0,n_1,\ldots,n_k)$. ∎

Theorem. If P is a formula $Q_1x_1\ldots Q_rx_rF(x_1,\ldots,x_r)$ of length n in $L_{PA}(m)$ where F is quantifier-free then P is true (i.e. in PA(m))

iff

$$(Q_1x_1 \le 2^{2^{(c+1)n^{m+3}}})(Q_2x_2 \le 2^{(c+2)n^{m+3}})\ldots$$

$$(Q_rx_r \le 2^{2^{(c+r)n^{m+3}}})F(x_1,\ldots,x_r)$$

is true for some c>0 and all n>4.

Proof. By the previous theorem,

$$|x_1| \le 2^{2^{(c+1)n^{m+3}}}.$$

We now determine a bound for x_{j+1} assuming that for all $j \le i$,

$$|x_j| \le 2^{2^{(c+j)n^{m+3}}}.$$

We then have

$$|x_{i+1}| \le (i\cdot w+1)2^{2^{(c+1)n^{m+3}}}$$

$$\le (i\cdot 2^{2^{(c+i)n^{m+3}}+1})2^{2^{(c+1)n^{m+3}}}$$

$$\le 2^{2^{(c+(i+1))n^{m+3}}} \quad \text{for } n>4,$$

for some c>0 (indeed the constant of the previous theorems). The theorem follows.

The algorithm to determine of a formula in $L_{PA}(m)$ if it is in PA(m) is simply to try all possible instantiations of integers for variables up to the limit known for a minimal solution if such exists. Each proposed instantiation is quite easily evaluated. The total space required is clearly less than double the formula length when instantiated because the checking computations are local events. Because the truth evaluation of each instantiated linear inequality, divisibility and non-divisibility is a small polynomial in the length of the atom, the non-deterministic time agrees with the space bound except for an adjustment in the coefficient of the exponent. The time a deterministic machine needs to run such a computation has a bound of $f \cdot 2^{\text{max space}}$ if f is the maximum time between memory alterations. These facts, plus the fact that at most n variables can appear in the formula and each integer needs space the log of its value, yields the theorem we seek.

Theorem. The membership in PA(m) (i.e. truth) of a formula in $L_{PA}(m)$ of length n can be ascertained by a deterministic machine within space

$$2^{dn^{m+4}}$$

and within time

$$2^{2^{en^{m+4}}}$$

for appropriate d>0 and e>0 for n>4.

References

1. Borosh I. and Treybig, L. B. Bounds on positive integral solutions of linear diophantine equations. Proc. AMS 55, March, 1976.

2. Cooper, D. C. Theorem-proving in arithmetic without multiplication. Machine Intell. 7, J. Wiley, 1972.

3. Ferrante, J. and Rackoff, C. A decision procedure for the first order theory of real addition with order. SIAM J. Comp., March, 1975.

4. Fischer, M. and Rabin, M. O. Super-exponential complexity of Presburger arithmetic. Project MAC. Tech. Memo 43, MIT, Cambridge, 1974.

5. Oppen, D. C. Elementary bounds for Presburger arithmetic. 5th SIGACT, May, 1973.

6. Presburger, M. Uber die Vollstandigkeit eines gewissen Systems der Arithmetic ganzer Zahlen, in welchem die Addition als einzige Operation hervortritt. Compte-Rendus dei Congres des Math. des pays slavs, Warsaw, 1929.

7. Shostak, R. An efficient decision algorithm for arithmetic with function symbols. Talk at Workshop on Auto. Deduction, Aug. 1977.

8. Suzuki, N. and Jefferson, D. Verification decidability of Presburger array programs. CMU Comp. Sci. Report, June, 1977.