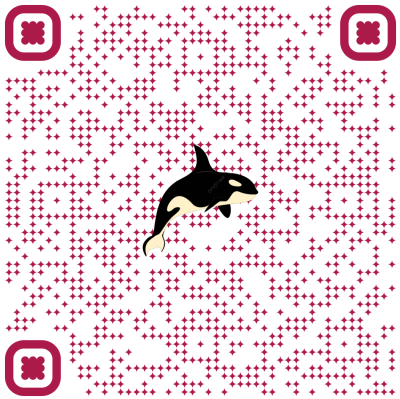
Quantifier Elimination Over the Integers

Rui-Juan Jing [1], Yuzhuo Lei [2], Christopher F. S. Maligec [2], Marc Moreno Maza [2], **Chirantan Mukherjee [2]**

[1]School of Mathematical Sciences, Jiangsu University,
[2]Ontario Research Center for Computer Algebra, University of Western Ontario

April 7, 2025



Content

Contents

What is QE?

Input

Consider a formula in prenex normal form,

$$F = Q_1 x_1 \ \ldots \ Q_m x_m \ \phi(x_1, \ldots, x_m, y_1, \ldots, y_n)$$

where,

1. $Q_1, \ldots Q_m$ is a sequence of quantifiers (existential $\exists$ or universal $\forall$),
2. $x_1, \ldots, x_m$ are bound variables,
3. $y_1, \ldots, y_n$ are free variables and,
4. $\phi(x_1, \ldots, x_m, y_1, \ldots, y_n)$ is a quantifier-free formula.

Output

A set $\mathscr{D}(y_1, \ldots, y_n)$ consisting of all tuples $(y_1, \ldots, y_n) \in \mathbb{Z}^n$ that make F true.

Bound variable

Definition

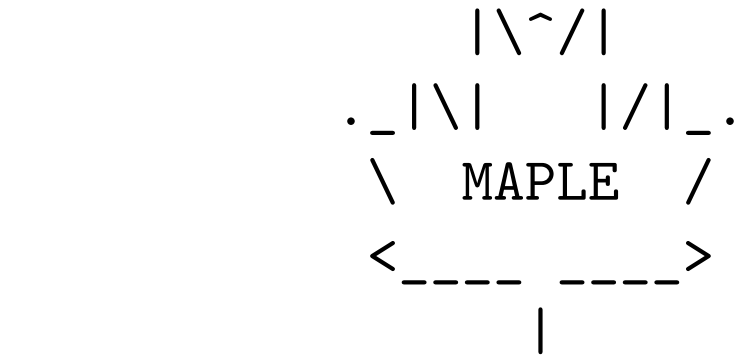
1. An upper bound for a variable x is a predicate of the form $x \leq a$ or $x < a$.
2. A lower bound for a variable x is a predicate of the form $a \leq x$ or $a < x$.

Example

$$\exists x \Big((x < 13 \vee 15 < x) \wedge x \leq y \Big)$$

1. There are two upper bounds for x:
   1. $x < 13$
   2. $x \leq y$
2. There is one lower bound for x:
   1. $15 < x$

Maple Worksheet



Applications

- Optimizing Compilers:
  - Array Dependence Analysis (Delinearization Problem)
  - Polyhedral frameworks (LLVM's Polly [GrGrLe12], GCC's Graphite).
- Program Verification:
  - Stanford Pascal Verifier [Lu79], Microsoft Spec#.
  - CompCert
- Theorem Proving:
  - SAT/SMT Solvers (Microsoft's Z3 [MoBj08], CVC5 [Ba22])
  - Proof Assistants (Coq [BeCa04], Isabelle [NiPaWe02], HOL Light [Ha96], Lean [Mo15]).

Software Implementations

- ISL (Integer Set Library) [Ve24]
- TaPAS (Talence Presburger Arithmetic Suite) [ChGuUn15]
- Yices [Du14]
- Princess (Scala Theorem Prover) [Rü08]
- Our Software

Why is QE hard?

Example (Hilbert's Tenth Problem [Hi02])

Is there a general algorithm for,
`Input:` polynomial $p(x_1, \ldots, x_n)$ with coefficients in $\mathbb{Z}$
`Output:` $\exists (a_1, \ldots, a_n) \in \mathbb{Z}^n$ such that $p(a_1, \ldots, a_n) = 0$

No! [Ma70, DaPuRo61]

Definition

Peano Arithmetic [Pe89] is a first order language $\mathscr{L}(0, s, +, *)$.

1. Peano Arithmetic is undecidable [Ch36, Tu36] and incomplete [Go31].
2. QE of semi-algebraic expressions over $\mathbb{R}$ is decidable [TaMc51].
3. QE of semi-algebraic expressions over real closed fields by cylindrical algebraic decompostion is also decidable [Co75].

Contents

Setup

$$F = Q_1 x_1 \ \ldots \ Q_m x_m \ \phi(x_1, \ldots, x_m, y_1, \ldots, y_n)$$

Assume F is in disjunctive normal form

$$\phi(x_1, \ldots, x_m, y_1, \ldots, y_n) \ = \ \bigvee_i \bigwedge_j \ \Phi_{ij}(x_1, \ldots, x_m, y_1, \ldots, y_n),$$

where,

1. each $\Phi_{ij}(x_1, \ldots, x_m, y_1, \ldots, y_n)$ is an atomic formula (or an atom),
2. thus a formula free of quantifiers and connectives.

Remark

We can assume that each atom is either

1. a non-strict inequality $\ell(x_1, \ldots, x_m, y_1, \ldots, y_n) \leq 0$ or,
2. a divisibility relation $k \mid \ell(x_1, \ldots, x_m, y_1, \ldots, y_n)$.

Presburger Arithmetic

Definition

Presburger arithmetic (PA) [Pr29] is a weaker version of Peano's arithmetic $\mathscr{L}(0, s, +)$.

1. PA is decidable and complete [Pr29].
2. PA originally applied to only $\mathbb{N}$, but it was later extended to work over $\mathbb{Z}$ by Robinson [Ro49].
3. Cooper [Co72] improved the QE procedure for eliminating the existential quantifiers ($\exists$) from formulas.

Complexity Estimates

1. Let n be the length of a statement in a PA.
2. Fischer & Rabin [FiRa74]
   1. They established a lower bound of $2^{2^{\Omega(n)}}$
   2. This implies that any decision algorithm requires at least double exponential time, i.e., at least $2^{2^{c \cdot n}}$ for some constant $c > 0$.
3. Oppen [Op78]
   1. For formulas with a fixed number of quantifier alternations, provided an upper bound of $2^{2^{2^{O(n)}}}$

Contents

General Idea for QE

$$F = Q_1 x_1 \ \ldots \ Q_m x_m \ \phi(x_1, \ldots, x_m, y_1, \ldots, y_n)$$

When $m = 0$

F itself is a quantifier-free formula, then it suffices to determine the tuples of integer values $(y_1, \ldots, y_n)$ for which $\phi(y_1, \ldots, y_n)$ is true.

When $m > 0$

1. We can view F as $Q_1 x_1 F'$, where F' is a formula and x_1 is free.
2. By induction assume that F' has been converted to a quantifier-free formula. Now two cases arise,
   1. either Q_1 is the existential quantifier,
   2. or Q_1 is the universal quantifier and we use De Morgan's law, $\forall x_1 \ F'$ with $\neg(\exists x_1 \ \neg(F'))$.

Cooper's Algorithm [Co72]

1. Each atom of the input formula, say $\Psi(x)$, is one of the following four types: $A_{\mathbf{y}} < ax,\ ax < A_{\mathbf{y}},\ d \mid (ax + A_{\mathbf{y}}),\ \text{ and }\ \neg(d \mid (ax + A_{\mathbf{y}}))$, where $a, d \in \mathbb{Z}$ and $A_y \in \mathbb{Z}[y_1, \ldots, y_n]$ is a linear polynomial.
2. Set $x' = \ell\, x, \quad$ with $\quad \ell = \operatorname{lcm}(\text{coefficients of } x)$.
3. There are two possibilities,
   1. either infinitely many (arbitrarily small) integers k satisfy $\Psi(x')$, or
   2. there is a least integer k satisfying $\Psi(x')$.
4. Replace each lower bound inequality in $\Psi(x')$ by false and each upper bound by true, denoting the result as $\Psi_{-\infty}(x')$. Then,

$$\exists x'\, \Psi(x') \iff \bigvee_{i=1}^{\delta} \Psi_{-\infty}(i) \vee \bigvee_{i=1}^{n} \bigvee_{j=1}^{\delta} \Psi(b_i + j),$$

where $b_1, \ldots, b_n$ are the lower bounds in $\Psi(x')$ and δ is the lcm of all divisibility moduli.

Contents

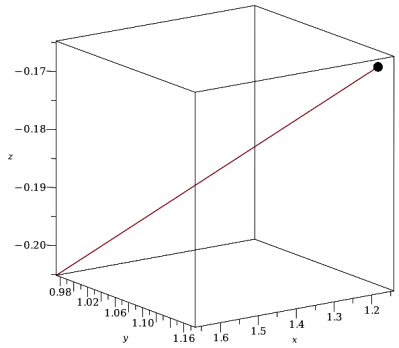
$\mathbb{Z}$Polyhedron

A $\mathbb{Z}$Polyhedron is a subset of integer points of a polyhedral set.

Example

`Input:`

$$\begin{cases} 7x + 12y + 31z &= 17, \\ 3x + 5y + 14z &= 7, \\ x + y &\geq 0, \\ y - x &\leq 0. \end{cases}$$



`Output:`

$$\begin{cases} x &= -13z - 1, \\ y &= 5z + 2, \\ z &\leq -1. \end{cases} \quad \text{and} \quad \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} -13 \\ 5 \\ 1 \end{pmatrix} \mathbf{t} + \begin{pmatrix} -1 \\ 2 \\ 0 \end{pmatrix}.$$

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \in \mathbb{Z}\text{Polyhedron}\left(\begin{pmatrix} -13 \\ 5 \\ 1 \end{pmatrix}, \begin{pmatrix} -1 \\ 2 \\ 0 \end{pmatrix} \right).$$

$$F = Q_1 x_1 \ \ldots \ Q_m x_m \ \phi(x_1, \ldots, x_m, y_1, \ldots, y_n)$$

Remark

We can re-arrange the quantifier-free part as,

$$\phi(x_1, \ldots, x_m, y_1, \ldots, y_n) = \bigvee_i Z_i(x_1, \ldots, x_m, y_1, \ldots, y_n),$$

where each Z_i is a predicate of the form

$$\begin{pmatrix} x_1 \\ \vdots \\ x_m \\ y_1 \\ \vdots \\ y_m \end{pmatrix} \in \mathbb{Z}Polyhedron(P_i, L_i),$$

for some polyhedra P_i and integer lattices L_i.

Only one variable x to eliminate (m = 1) in F

1. Let $f_1, \ldots, f_s, g_1, \ldots, g_r \in \mathbb{Z}[x, \mathbf{y}]$ be linear polynomials and $k_1, \ldots, k_r$ be positive integers.
2. Consider the formula:

$$F(\mathbf{y}) : (\exists x \in \mathbb{Z}) \begin{cases} f_1 & \leq & 0 \\ \vdots & \vdots & \vdots \\ f_s & \leq & 0 \end{cases} \wedge \boxed{\begin{array}{rcll} g_1 & \equiv & 0 & \mod k_1 \\ \vdots & \vdots & \vdots & \\ g_r & \equiv & 0 & \mod k_r \end{array}} \tag{1}$$

3. We developed a generalized version the Chinese Remaindering theorem for multivariate parametric systems of linear congruences.
4. Substitute the solution into the system on linear inequalities.
5. Our goal is to determine $\mathscr{D}(\mathbf{y})$, the set of the tuples of integer values $(y_1, \ldots y_n)$ for which $F(\mathbf{y})$ is true. We call $\mathscr{D}(\mathbf{y})$ the integer projection of $F(\mathbf{y})$.

Two inequalities (s = 2) and no congruences (r = 0)

1. Replace $\begin{cases} f_1 & \leq & 0 \\ f_2 & \leq & 0 \end{cases}$ with $\begin{cases} A_{\mathbf{y}} - a\,x, \leq & 0 \\ -B_{\mathbf{y}} + b\,x \leq & 0 \end{cases}$, where $A_{\mathbf{y}}, B_{\mathbf{y}} \in \mathbb{Z}[\mathbf{y}]$ are linear, and $a, b \in \mathbb{Z}$ are non-zero.
2. Focus on $a > 0, b > 0$. (The other cases are easy!)
3. Formula 1 simplifies to:

$$F(\mathbf{y}) : (\exists x \in \mathbb{Z})\ (A_{\mathbf{y}} \leq a\,x)\ \wedge\ (b\,x \leq B_{\mathbf{y}}), \tag{2}$$

4. We use an idea suggested in Williams [Wi76] as well as Pugh's lemma [Pu91, Pu92].

Integer projection: first theorem

Theorem

Let, $\ell = lcm(a, b), b' = \ell/a$ *and* $a' = \ell/b$. *For* $0 \leq k < b$ *define,*

$$E_k := \{\mathbf{y} \mid rem(B_{\mathbf{y}}, b) = k\}.$$

Then the following conditions are equivalent,

1. $F(\mathbf{y}) := (\exists x \in \mathbb{Z})(A_{\mathbf{y}} \leq ax) \wedge (bx \leq B_{\mathbf{y}})$,
2. $\bigvee_{k=0}^{k=b-1}(\mathbf{y} \in E_k) \wedge (a'k \leq a'B_{\mathbf{y}} - b'A_{\mathbf{y}})$

Remark

- The number of atoms in the above formula grows doubly exponentially with the number of variables being eliminated.
- We proved that this formula is equivalent to the condition computed by Cooper's algorithm
- In practice, the latter has a (much) larger number of atoms.

Pugh's dark shadow

Recall that we are eliminating the integer variable x from:

$$F(\mathbf{y}) := (\exists x \in \mathbb{Z})(A_{\mathbf{y}} \leq ax) \wedge (bx \leq B_{\mathbf{y}})$$

Pugh has found a sufficient but not necessary condition to reduce the number of cases.

Lemma (Dark Shadow [Pu91, Pu92])

If we have,

$$aB_{\mathbf{y}} - bA_{\mathbf{y}} \geq (a-1)(b-1), \tag{3}$$

then $F(\mathbf{y})$ holds.

Integer projection: second theorem

We combine the above first theorem with Pugh's lemma.

Theorem

Define $\kappa(a,b) := \lceil \frac{(a-1)(b-1)}{a'} \rceil$. *Then, Formula (1) is equivalent to:*

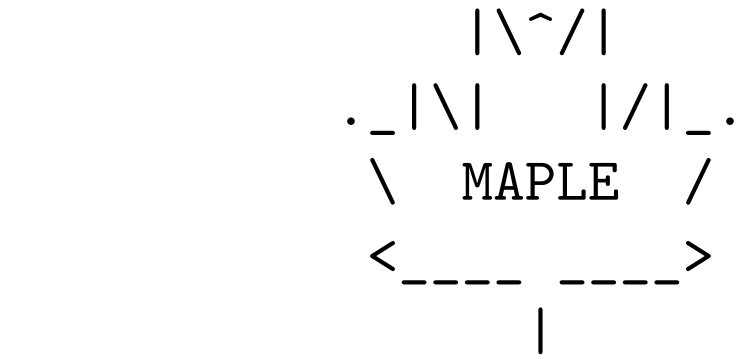
$$((a-1)(b-1) \leq aB_{\mathbf{y}} - bA_{\mathbf{y}}) \bigvee \bigvee_{k=\kappa(a,b)}^{k=b-1} (\mathbf{y} \in E_k) \wedge (a'k \leq a'B_{\mathbf{y}} - b'A_{\mathbf{y}}) \,. \tag{4}$$

Remark

1. the second theorem reduces significantly the number of "cuts".
2. To take a concrete example, say with $a = 7$ and $b = 11$,
   1. with the first theorem alone k ranges from 0 to 10,
   2. with Pugh's lemma, k ranges from 8 to 10.

Contents

Maple worksheet



test	IPD MEMORY(MB)	IPD TIME(s)	NIP MEMORY(MB)	NIP TIME(s)
T1[BoGoWo17]	24.232	0.121	33.811	0.193
T2[BoGoWo17]	57.843	0.281	59.136	0.344
T3[BoGoWo17]	121.978	0.671	189.439	1.256
T4[BoGoWo17]	42.531	0.240	65.162	0.378
T5[BoGoWo17]	22.114	0.110	31.725	0.167
T6[SeLoMe12]	97.739	0.481	64.456	0.333
T7[St23]	671.154	3.506	1066.889	6.608
T8[KöVeWo08]	69.087	0.338	58.668	0.328
T9 [KöVeWo08]	245.156	1.235	979.964	6.462
T17[BoGoWo17]	5.315	0.043	12.771	0.060
T18[CaLiZh22]	39.055	0.200	48.237	0.205
T19[Fe88]	355.466	1.786	1715.958	10.941
T20[Ve24]	25.453	0.154	28.667	0.180
T32[SeLoMe12]	28216.613	156.989	$>$ 10 GB	$>$ 600
T33[SeLoMe12]	70.135	0.351	345.340	1.920
T34[SeLoMe12]	178.657	0.928	366.935	2.487
T35[SeLoMe12]	121.098	0.645	165.582	1.053
T36[SeLoMe12]	1243.682	6.209	798.004	4.822
T44[Ve24]	1.549	0.013	1.550	0.014
T45[Ve24]	1.549	0.014	1.551	0.013
T46[Ve24]	1.551	0.017	1.552	0.013
T47[Ve15]	49.726	0.236	45.779	0.219
T48[Ve15]	53.819	0.260	98.094	0.540
T49[Ve15]	32.190	0.197	27.997	0.153

Table: Maple 2024, Ubuntu 24.04.1 LTS, 16GB RAM and 12th Gen Intel(R) Core(TM) i5-1235U processor

The code can be accessed here.

Contents

Conclusions

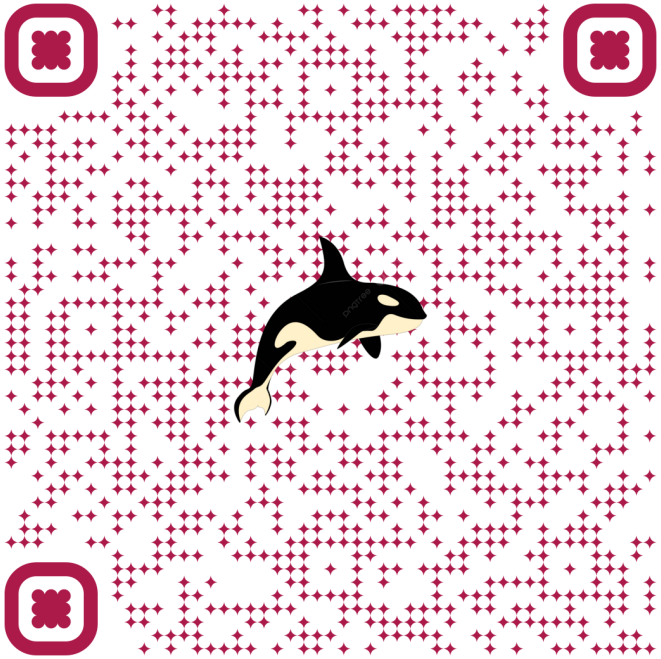
1. We have enabled two modes for QE,
   1. `onequantifieratatime` - eliminate one quantifier at at time
   2. `byblocksofquantifiers` - eliminate blocks of same quantifier in one step

Example

$F := \forall x \forall y \exists z \exists w f(x, y, z, w, \mathbf{p})$

2. We shall continue investigating heuristics to bypass the use of De Morgan's laws when dealing with universal quantifiers,
   1. $\forall \mathbf{x} f(\mathbf{x}) = \mathbf{b}$ is true only when all coefficients are 0
   2. $\forall \mathbf{x} f(\mathbf{x}) \leq \mathbf{b}$ is true only when all coefficients in LHS are 0 and in RHS are ≥ 0
3. We shall explore how we could take advantage of recent developments, such as the results of Haase et al. in [Ha24],
4. We want to explore some non-linear QE problems over the integers, which occur in the field of optimizing compilers, in particular, scheduling problems.

Thank You!



Contents

References I

[KöVeWo08] Matthias Köppe, Sven Verdoolaege, and Kevin M. Woods.
An Implementation of the Barvinok–Woods Integer Projection Algorithm..
Technical Report. Oberlin College, Department of Mathematics, 2008.

[St23] Thomas Sturm.
Algorithmic Quantifier Elimination.
Lecture Notes, 2023.

[Ba94] Alexander I. Barvinok.
A Polynomial Time Algorithm for Counting Integral Points in Polyhedra When the Dimension is Fixed.
Math. Oper. Res. 19(4), 769–779, 1994.

[GrGrLe12] Tobias Grosser, Armin Groesslinger, and Christian Lengauer.
Polly—Performing Polyhedral Optimizations on a Low-Level Intermediate Representation.
In Parallel Processing Letters, Vol. 22. World Scientific, 1250010, 2012.

[Mo15] Leonardo de Moura, Soonho Kong, Jeremy Avigad, Floris van Doorn, and Jakob von Raumer.
The Lean theorem prover (system description).
In Automated Deduction – CADE-25 (Lecture Notes in Computer Science, Vol. 9195). Springer, 378–388, 2015.

[Ha96] John Harrison.
HOL Light: A tutorial introduction.
In Proceedings of the First International Conference on Formal Methods in Computer-Aided Design (FMCAD). Springer, 265–269, 1996.

[NiPaWe02] Tobias Nipkow, Lawrence C. Paulson, and Markus Wenzel.
Isabelle/HOL: A Proof Assistant for Higher-Order Logic.
Lecture Notes in Computer Science, Vol. 2283. Springer Berlin Heidelberg, 2002.

References II

[BeCa04] Yves Bertot and Pierre Castéran.
Interactive theorem proving and program development: Coq'Art.
The calculus of inductive constructions. Vol. 25. Springer Science & Business Media, 2004.

[Ba22] Haniel Barbosa, Clark Barrett, Martin Brain, Gereon Kremer, and Hantao Zhan.
CVC5: A Versatile and Industrial-Strength SMT Solver.
In Tools and Algorithms for the Construction and Analysis of Systems (TACAS) (Lecture Notes in Computer Science, Vol. 13243). Springer, 415–442, 2022.

[MoBj08] Leonardo de Moura and Nikolaj Bjørner.
Z3: An Efficient SMT Solver.
In Tools and Algorithms for the Construction and Analysis of Systems (TACAS) (Lecture Notes in Computer Science, Vol. 4963). Springer, 337–340, 2008.

[Lu79] David C. Luckham, Norihisa Suzuki, Friedrich W. von Henke, Bernd Krieg-Brückner, and Susan S. Owicki.
Stanford Pascal Verifier user manual.
Computer Science Department Report, Stanford University STAN-CS-79-731, 1979.

[Rü08] Philipp Rümmer.
A Constraint Sequent Calculus for First-Order Logic with Linear Integer Arithmetic.
In Logic for Programming, Artificial Intelligence, and Reasoning (LPAR), Vol. 5330. Springer, 274–289, 2008.

[Du14] Bruno Dutertre.
Yices 2.2.
In Computer Aided Verification (CAV 2014), Vol. 8559. Springer, 737–744, 2014.

[ChGuUn15] Supratik Chakraborty, Ashutosh Gupta, and Divyesh Unadkat.
TaPAS: Tools and Algorithms for the Verification of Parameterized Systems.
In Proceedings of the 13th International Symposium on Automated Technology for Verification and Analysis (ATVA). Springer, 352–359, 2015.

References III

[FiRa74] Michael J. Fischer, and Michael O. Rabin.
Super-Exponential Complexity of Presburger Arithmetic.
In Karp, Richard M. (ed.). Complexity of computation. SIAM-AMS Proceedings. Vol. 7. American Mathematical Society. pp. 27–41, 1974.

[Op78] Derek C. Oppen.
A $2^{2^{2^{pn}}}$ upper bound on the complexity of Presburger Arithmetic.
J. Comput. Syst. Sci. 16 (3): 323–332, 1978.

[SeLoMe12] Rachid Seghir, Vincent Loechner, and Benoit Meister.
Integer affine transformations of parametric $\mathbb{Z}$-polytopes and applications to loop nest optimization.
ACM Trans. Archit. Code Optim. 9, 2 (2012), 8:1–8:27, 2012.

[Ve15] Sven Verdoolaege.
Integer set coalescing.
, 2015.

[Ve24] S. Verdoolaege.
Integer Set Library: Manual Version: isl-0.2.
, 2024.

[Fe88] P. Feautrier.
Parametric integer programming.
RAIRO. Recherche opérationnelle 22, 3 (1988), 243 – 268, 1988.

[CaLiZh22] Shaowei Cai, Bohan Li, and Xindi Zhang.
Local Search for SMT on Linear Integer Arithmetic.
Verification - 34th International Conference, CAV 2022, Haifa, Israel, August 7-10, 2022, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 13372), Sharon Shoham and Yakir Vizel (Eds.)., Springer, 227–248, 2022.

References IV

[BoGoWo17] Tristram Bogart, John Goodrick, and Kevin Woods.
Parametric Presburger arithmetic: logic, combinatorics, and quasi-polynomial behavior.
Discrete Analysis, 2017.

[Pu91] William Pugh.
The Omega test: a fast and practical integer programming algorithm for dependence analysis.
In Proceedings of the 1991 ACM/IEEE conference on Supercomputing. 4–13, 1991.

[Pu92] William Pugh.
A Practical Algorithm for Exact Array Dependence Analysis.
Commun. ACM 35, 8 (1992), 102–114, 1992.

[WiHo16] H.P. Williams and J.N. Hooker.
Integer programming as projection.
Discrete Optimization 22 (2016), 291–311, 2016.

[Wi76] H Paul Williams.
Fourier-Motzkin elimination extension to integer programming problems.
Journal of combinatorial theory, series A 21, 1 (1976), 118–123, 1976.

[Ha24] Christoph Haase, Shankara Narayanan Krishna, Khushraj Madnani, Om Swostik Mishra, and Georg Zetzsche.
An Efficient Quantifier Elimination Procedure for Presburger Arithmetic.
In 51st International Colloquium on Automata, Languages, and Programming, ICALP 2024, July 8-12, 2024, Tallinn, Estonia (LIPIcs, Vol. 297), Karl Bringmann, Martin Grohe, Gabriele Puppis, and Ola Svensson (Eds.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 142:1–142:17, 2024.

[Co72] David C Cooper.
Theorem proving in arithmetic without multiplication.
Machine intelligence 7, 91-99 (1972), 300, 1972.

References V

[Ro49] Julia Robinson.
Definability and Decision Problems in Arithmetic.
Journal of Symbolic Logic 14, 2 (1949), 98–114, 1949.

[Hi02] David Hilbert.
Mathematical Problems.
Bull. Amer. Math. Soc. 8, 10 (1902), 437–479, 1902.

[Pe89] Giuseppe Peano.
Arithmetices principia, nova methodo exposita.
Fratres Bocca, Turin, Italy, 1889.

[Pr29] Mojżesz Presburger.
Über die Vollständigkeit eines gewissen Systems der Arithmetik ganzer Zahlen.
In Proceedings of the First Congress of Mathematicians of the Slavic Countries. 92–101. Presented at the First Congress of Mathematicians of the Slavic Countries, 1929.

[Ma70] Yuri V. Matiyasevich.
Enumerable sets are Diophantine.
Doklady Akademii Nauk SSSR 191 (1970), 279–282, 1970.

[DaPuRo61] Martin Davis, Hilary Putnam, and Julia Robinson.
The decision problem for exponential Diophantine equations.
Journal of Symbolic Logic 26, 1 (1961), 12–33, 1961.

[Go31] Kurt Gödel.
Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I.
Monatshefte für Mathematik und Physik 38, 1 (1931), 173–198, 1931.

References VI

[Ch36] Alonzo Church.
An Unsolvable Problem of Elementary Number Theory.
American Journal of Mathematics 58, 2 (1936), 345–363, 1936.

[Tu36] Alan M. Turing.
On Computable Numbers, with an Application to the Entscheidungsproblem.
Proceedings of the London Mathematical Society s2-42, 1 (1936), 230–265, 1936.

[TaMc51] Alfred Tarski and J. C. C. McKinsey.
A Decision Method for Elementary Algebra and Geometry.
University of California Press, Berkeley, 1951.

[Co75] George E. Collins.
Quantifier elimination for real closed fields by cylindrical algebraic decompostion.
In Automata Theory and Formal Languages, H. Brakhage (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 134–183, 1975.