When Less Is More: Consequence-Finding in a Weak Theory of Arithmetic

ZACHARY KINCAID*, Princeton University, United States
NICOLAS KOH*, Princeton University, United States
SHAOWEI ZHU*, Princeton University, United States

This paper presents a theory of non-linear integer/real arithmetic and algorithms for reasoning about this theory. The theory can be conceived of as an extension of linear integer/real arithmetic with a weakly-axiomatized multiplication symbol, which retains many of the desirable algorithmic properties of linear arithmetic. In particular, we show that the *conjunctive* fragment of the theory can be effectively manipulated (analogously to the usual operations on convex polyhedra, the conjunctive fragment of linear arithmetic). As a result, we can solve the following consequence-finding problem: *given a ground formula F, find the strongest conjunctive formula that is entailed by F*. As an application of consequence-finding, we give a loop invariant generation algorithm that is monotone with respect to the theory and (in a sense) complete. Experiments show that the invariants generated from the consequences are effective for proving safety properties of programs that require non-linear reasoning.

CCS Concepts: • **Theory of computation** → **Automated reasoning**; *Program analysis*; *Invariants*; • **Mathematics of computing** → *Gröbner bases and other special bases.*

Additional Key Words and Phrases: Decision procedures, theory of arithmetic, convex polyhedra, polynomial ideals, program analysis, nonlinear invariant generation

ACM Reference Format:
Zachary Kincaid, Nicolas Koh, and Shaowei Zhu. 2023. When Less Is More: Consequence-Finding in a Weak Theory of Arithmetic. *Proc. ACM Program. Lang.* 7, POPL, Article 44 (January 2023), 33 pages. https://doi.org/10.1145/3571237

44

1 INTRODUCTION

The theory of linear integer/real arithmetic possesses characteristics that make it useful across a range of applications. Foremost, it is a decidable theory, with practical decision procedures based on (integer) linear programming. Beyond decidability, the conjunctive fragments of linear real and integer arithmetic (corresponding to convex polyhedra and the integer points within them, respectively) can be manipulated effectively. This fact has been leveraged, for instance, for invariant generation [Colón et al. 2003; Cousot and Halbwachs 1978], termination analysis [Podelski and Rybalchenko 2004], optimization [Bjørner et al. 2015; Li et al. 2014; Sebastiani and Tomasi 2012], and program transformation [Feautrier 1996; Lengauer 1993].

Including multiplication in the language of arithmetic has a dramatic effect on the resulting theory. Non-linear (integer) arithmetic is not even recursively axiomatizable, let alone decidable. As a result,

*The authors are listed in alphabetical order.

Authors' addresses: Zachary Kincaid, Princeton University, Princeton, NJ, United States, zkincaid@cs.princeton.edu; Nicolas Koh, Princeton University, Princeton, NJ, United States, ckoh@cs.princeton.edu; Shaowei Zhu, Princeton University, Princeton, NJ, United States, shaoweiz@cs.princeton.edu.

2475-1421/2023/1-ART44
https://doi.org/10.1145/3571237

solvers for non-linear arithmetic rely on heuristic reasoning techniques [Borralleras et al. 2019, 2009; Fuhs et al. 2007; Jovanović 2017; Kremer et al. 2016], and algorithms for manipulating the conjunctive fragment (e.g., [Bagnara et al. 2005; Kincaid et al. 2017]) are imprecise. The use of such heuristics precludes clients of non-linear solvers and abstract domains from satisfying desirable properties—for instance, there are no non-trivial complete ranking function synthesizers or monotone invariant generation schemes for non-linear integer arithmetic. And while heuristics are often effective in practice, they can also be unpredictable. For instance, Hawblitzel et al. [2014] reports "we found Z3's theory of nonlinear arithmetic to be slow and unstable; small code changes often caused unpredictable verification failures," which prompted the authors to develop information-hiding techniques to avoid triggering non-linear heuristics.

This paper develops the theory of *linear integer/real rings* (**LIRR**)—commutative rings extended with an order relation and an "integer" predicate that obey certain axioms from the theory of *linear* integer/real arithmetic. While all axiomatizable theories of non-linear arithmetic are incomplete, **LIRR** is weak by design (relative to say Peano arithmetic), trading power for tractable automated reasoning:

- **LIRR** is decidable. Furthermore, **LIRR** does not lose any of the reasoning power of linear integer/real arithmetic (in a sense made precise in Theorems 1 and 8).
- The conjunctive fragment of **LIRR** can be manipulated effectively, analogously to convex polyhedra. This enables some clients of linear arithmetic (for instance, recurrence-based loop invariant generation—see Section 5) to be "lifted" to non-linear arithmetic.
- **LIRR** is axiomatized by Horn clauses. This implies existence of *minimal models* [Van Emden and Kowalski 1976], which simplifies consequence-finding. It also ensures that the theory is convex [Tinelli 2003] (and stably infinite), so **LIRR** can be combined with other theories via the Nelson-Oppen protocol [Nelson and Oppen 1979].

The key technical contribution of this paper is a suite of algorithms for manipulating *algebraic cones*—a set of polynomials that can be represented as the sum of a polynomial ideal and a polyhedral cone. Algebraic cones can be seen as a representation of a system of polynomial equalities and inequalities (analogous to the constraint representation of a convex polyhedron). Similarly, congruence constraints can be represented by an *algebraic lattice*—a sum of a polynomial ideal and a point lattice. Algebraic cones and lattices are the basis of our decision procedure for the problem of testing satisfiability of a ground formula modulo **LIRR**, wherein they serve as a representation of a Herbrand model. We show that algebraic cones can be effectively manipulated like convex polyhedra: membership and emptiness are decidable, and algebraic cones are closed under intersection, sum, projection, inverse homomorphism, and cutting plane closure with respect to an algebraic lattice (analogous to computing the convex hull of the integer points within a convex polyhedron).

Algorithms for algebraic cones arise as a marriage of techniques between polynomial ideals (based on Gröbner bases) and convex polyhedra. Such combinations have been investigated in prior work (e.g., [Bagnara et al. 2005; Kincaid et al. 2017; Tiwari 2005]), in the context of incomplete heuristics for reasoning about real or integer arithmetic. This paper investigates the strength of this combination through the lens of a first-order theory of arithmetic. The critical finding is that these methods enable complete consequence-finding (modulo **LIRR**). We present an algorithm that, given a ground formula F, computes the set of all polynomials p such that F entails that p is non-negative, modulo **LIRR** (analogous to computing the convex hull of F, modulo linear arithmetic). Such consequence-finding has a wide range of applications in program analysis [Reps et al. 2004]. As a case study, we give one application to non-linear invariant generation, and show that the technique has good practical performance on top of theoretical guarantees.

The paper is organized as follows. Section 2 presents background on logic, commutative algebra, and polyhedral theory. The theory of linear/integer rings is presented in two steps. Section 3 presents the theory of *linear real rings*, **LRR**, which is essentially the theory of linear integer/real rings excluding the integer predicate and its associated axioms. The full theory **LIRR** is given in Section 4. An invariant generation algorithm that demonstrates the use of consequence-finding is given in Section 5. Section 6 evaluates the decision procedure for **LIRR** and the invariant generation algorithm experimentally. The **LIRR** decision procedure is not empirically competitive with state-of-the-art heuristic solvers. On the other hand, the experimental results for the invariant generation procedure are positive, establishing the value of consequence-finding modulo **LIRR**. Related work is discussed in Section 7. Proofs for all statements in this paper can be found in Kincaid et al. [2022b].

2 BACKGROUND

2.1 First-Order Logic

A **signature** $\sigma = (F, R, ar)$ consists of a set of function symbols F and a set of relation symbols R that are mutually disjoint, and a function $ar : (F \cup R) \to \mathbb{Z}^{\geq 0}$ mapping each symbol to its arity. For any set of symbols X (presumed disjoint from F and R), we use $\sigma(X)$ to denote the extension of σ with the constant symbols X. A σ-structure $\mathfrak{A}$ consists of a set $U^{\mathfrak{A}}$ (the *universe* of $\mathfrak{A}$) along with an interpretation $f^{\mathfrak{A}} : (U^{\mathfrak{A}})^{ar(f)} \to U^{\mathfrak{A}}$ of each function symbol $f \in F$ and an interpretation $r^{\mathfrak{A}} \subseteq (U^{\mathfrak{A}})^{ar(r)}$ of each relation symbol $r \in R$. The set of σ-terms and σ-formulas are defined in the usual way. A formula is said to be a **sentence** if it has no free variables, and **ground** if it has neither free nor bound variables. A σ-structure $\mathfrak{A}$ is a **model** of a set of sentences T if for all $F \in T$, $\mathfrak{A}$ satisfies F (written $\mathfrak{A} \models F$). A σ**-theory** T is a set of sentences closed under entailment (for any sentence F, if F is satisfied by every model of T, then F belongs to T). For any σ-structure $\mathfrak{A}$, $Th(\mathfrak{A})$ denotes the σ-theory consisting of all sentences F such that $\mathfrak{A} \models F$. If T is a σ-theory and F is a $\sigma(X)$-formula, we say that F is **satisfiable modulo** T if there exists a $\sigma(X)$-structure $\mathfrak{A}$ that satisfies F along with each formula in T. If F and G are $\sigma(X)$-formulas, we say that F **entails** G **modulo** T (written $F \models_T G$) if every $\sigma(X)$-structure that satisfies F and each sentence in T also satisfies G.

2.2 Commutative Algebra

This section recalls some basic facts about commutative algebra (see [Cox et al. 2015]). Let σ_{or} be the signature of ordered rings, consisting of a binary addition (+) and multiplication ($\cdot$) operators, the constants 0 and 1, equality, and a binary relation $\leq$.

The **commutative ring axioms**, **CR**, are as follows:

$$\begin{array}{lr}
\forall x, y, z.\, x + (y + z) = (x + y) + z \text{ and } \forall x, y, z.\, x \cdot (y \cdot z) = (x \cdot y) \cdot z & \text{(Associativity)} \\
\forall x, y.\, x + y = y + x \text{ and } \forall x, y.\, x \cdot y = y \cdot x & \text{(Commutativity)} \\
\forall x.\, x + 0 = x \text{ and } \forall x.\, x \cdot 1 = x & \text{(Identity)} \\
\forall x, y, z.\, x \cdot (y + z) = (x \cdot y) + (x \cdot z) & \text{(Distributivity)} \\
\forall x, \exists y.\, x + y = 0 & \text{(Additive inverse)}
\end{array}$$

A model of these axioms is called a **commutative ring**. Examples of commutative rings include the integers $\mathbb{Z}$, the rationals $\mathbb{Q}$, and the reals $\mathbb{R}$. For any commutative ring R and finite set of variables X, let $R[X]$ denote the set of polynomials over X with coefficients in R; this too forms a commutative ring.

Modules are a generalization of linear spaces in which the scalars form a ring rather than a field. If R is a commutative ring, an R-**module** is a commutative group $(M, 0, +)$ equipped with a *scalar multiplication* operation $\cdot : R \times M \to M$ satisfying the usual axioms of linear spaces ($a \cdot (m + n) = a \cdot m + a \cdot n$, $(a + b) \cdot m = a \cdot m + b \cdot m$, $(a \cdot b) \cdot m = a \cdot (b \cdot m)$, and $1 \cdot m = m$). For instance,

R is itself an R-module where scalar multiplication is the usual ring multiplication; $R[X]$ is both an R-module and an $R[X]$-module.

Let R be a commutative ring and N be an R-module. For $L, M \subseteq N$ and $S \subseteq R$ we use $L + M$ to denote the set of sums of elements in L and M, and $S\langle M\rangle$ to denote the set of weighted sums of elements of M with coefficients in S:

$$L + M \triangleq \{\ell + m : \ell \in L, m \in M\}$$
$$S\langle M\rangle \triangleq \left\{s_1 m_1 + \cdots + s_n m_n : n \in \mathbb{Z}^{\geq 0}, s_1, \dots, s_n \in S, m_1, \dots, m_n \in M\right\}$$

For instance, if V is a linear space over $\mathbb{Q}$ and $G \subseteq V$ then $\mathbb{Q}\langle G\rangle$ is the *span* of G—the smallest linear subspace of V containing G. Note that $S\langle M\rangle$ always contains zero (since we may take $n = 0$). We omit braces for finite sets, and write $S\langle m_1, \dots, m_k\rangle$ for $S\langle\{m_1, \dots, m_k\}\rangle$.

Let R be a commutative ring. An **ideal** $I \subseteq R$ is a sub-module of R (considered as an R-module); i.e., a set that (1) contains zero, (2) is closed under addition, and (3) is closed under multiplication by arbitrary elements of R. An ideal I defines a congruence relation $\equiv_I$, where $p \equiv_I q$ if and only if $p - q \in I$. (The notation $p - q$ abbreviates $p + (-q)$, where $-q$ is the unique additive inverse of q.) One may think of an ideal as a set of elements that are congruent to zero with respect to *some* congruence relation, with the closure conditions of ideals corresponding to the idea that the sum of two zero-elements is zero, and the product of a zero-element with anything is again zero. We use R/I to denote the **quotient ring** in which the elements are sets of the form $r + I$ for some $r \in R$ (that is, equivalence classes of $\equiv_I$), and sum and product are defined as $(r + I) + (s + I) = (r + s) + I$ and $(r + I) \cdot (s + I) = (r \cdot s) + I$. Note that any subset $P \subseteq R$ generates an ideal (the *smallest* with respect to inclusion order containing P), which is exactly $R\langle P\rangle$. When R is clear from context, we will write $\langle P\rangle$ for the ideal $R\langle P\rangle$ generated by the set P.

Fix a set of variables X. We use $[X]$ to denote the set of monomials over X. A **monomial ordering** $\preceq$ is a total order on $[X]$ such that (1) $1 \preceq m$ for all m, and (2) for any $m \preceq n$ and any monomial v, we have $mv \preceq nv$. Applications often require fixing a monomial order, but the choice of *which* is irrelevant. A reasonable default is *degree reverse lexicographic order*: first compare monomials by their total degree, then break using a reversed lexicographic order. (Assuming a fixed monomial ordering) the **leading monomial** $\text{LM}(p)$ of a polynomial $p = a_1 m_1 + \cdots + a_n m_k \in \mathbb{Q}[X]$, $a_1, \dots, a_n \neq 0$, is the greatest monomial among $m_1, \dots, m_k$. The leading monomial of the zero polynomial is undefined.

Fix a monomial ordering $\preceq$. Let p be a non-zero polynomial in $\mathbb{Q}[X]$. Then p can be written as $p = am + q$ where $m = \text{LM}(p)$, a is the coefficient of m in p, and $q = p - am$, and interpret p as a rewrite rule $m \to -\frac{1}{a}q$. Intuitively, if p is in some ideal I, then $p \equiv_I 0$, and $m \equiv_I -\frac{1}{a}q$. For instance, the polynomial $\frac{1}{2}x^2 - y$ can be interpreted as a rewrite rule $x^2 \to 2y$, and using this rule we may rewrite $x^3 + x^2 \to 2xy + x^2 \to 2xy + 2y$. Observe that if $I = \left\langle \frac{1}{2}x^2 - y\right\rangle$, $x^3 + x^2 \equiv_I 2xy + 2y$. Applying a rewrite rule to a polynomial replaces one of its monomials with a smaller polynomial (in the sense that the polynomial is zero or its leading monomial is smaller than the one it replaced). A set of polynomials G is a **Gröbner basis** (with respect to $\preceq$) if its associated rewrite system is confluent. Assuming that G is a Gröbner basis, we use $\mathbf{red}_G : \mathbb{Q}[X] \to \mathbb{Q}[X]$ to denote the function that maps any polynomial to its normal form under the rewrite system G. Equivalently, $\mathbf{red}_G(p)$ is the unique polynomial q such that (1) $p - q \in \langle G\rangle$, and (2) no monomial in q is divisible by $\text{LM}(g)$ for any $g \in G$. We note some important properties of $\mathbf{red}_G$:

- (Membership) For all $p \in \mathbb{Q}[X]$, $\mathbf{red}_G(p) = 0$ if and only if $p \in \langle G\rangle$
- (Linearity) If $a_1, \dots, a_n \in \mathbb{Q}$ and $p_1, \dots, p_n \in \mathbb{Q}[X]$, then $\mathbf{red}_G(\sum_{i=1}^n a_i p_i) = \sum_{i=1}^n a_i \mathbf{red}_G(p_i)$
- (Ordering) For all $p \in \mathbb{Q}[X]$, $p = q_1 g_1 + \cdots + q_n g_n + \mathbf{red}_G(p)$ for some $q_1, \dots, q_n \in \mathbb{Q}[X]$, $g_1, \dots, g_n \in G$, and $\text{LM}(q_i g_i) \preceq \text{LM}(p)$ for all i.

- (Representation independence) If G_1 and G_2 are Gröbner bases with respect to the same monomial ordering and $\langle G_1 \rangle = \langle G_2 \rangle$, then $\mathbf{red}_{G_1}(p) = \mathbf{red}_{G_2}(p)$ for all p.

For any finite set of polynomials P and monomial ordering $\preceq$, we may compute a Gröbner basis for the ideal generated by P (e.g., using Buchberger's algorithm [Buchberger 1976]). We use $\textit{GröbnerBasis}_{\preceq}(P)$ to denote this basis.

2.3 Polyhedral Theory

This section recalls some basic facts about polyhedral theory (see for example [Schrijver 1999]). In the following, we use **linear space** to refer to a linear space over the field $\mathbb{Q}$. We use $\mathbb{Q}^n$ to denote the linear space of rational vectors of length n. Note that for any set of variables X, $\mathbb{Q}[X]$ is an (infinite-dimensional) linear space. We use $\mathbb{Q}[X]^1$ to denote the ($|X|$+1-dimensional) linear space of polynomials of degree at most one (i.e., polynomials of the form $a_1x_1 + \ldots a_nx_n + b$, with $a_1, \ldots, a_n, b \in \mathbb{Q}$ and $x_1, \ldots, x_n \in X$).

Let V be a linear space. A set $C \subseteq V$ is called **convex** if for every $u, v \in C$, the line segment between u and v is contained in C (i.e., for every λ in the interval $[0, 1]$, we have $\lambda u + (1 - \lambda)v \in C$). For a set $G \subseteq V$, we use $conv(G)$ to denote the **convex hull** of G—the smallest convex set that contains G:

$$conv(G) \triangleq \left\{ \lambda_1 g_1 + \cdots + \lambda_n g_n : \lambda_1, \ldots, \lambda_n \in \mathbb{Q}^{\geq 0}, g_1, \ldots, g_n \in G, \sum_{i=1}^{n} \lambda_i = 1 \right\}$$

where $\mathbb{Q}^{\geq 0}$ denotes the set of non-negative rational numbers. A set $C \subseteq V$ is called a (convex) **cone** if it contains zero and is closed under addition and multiplication by non-negative scalars. For a set $G \subseteq V$, the **conical hull** of G is the smallest convex cone containing G; it is precisely $\mathbb{Q}^{\geq 0}\langle G \rangle$.

Let V be a linear space and $C \subseteq V$ be a cone. We say that $v \in C$ is an **additive unit** if both v and $-v$ belong to C, and denote the set of additive units as $\mathcal{U}(C)$ ($\mathcal{U}(C)$ is also known as the *lineality space* of C—the largest linear space contained in C). We say that a C is **salient** if $\mathcal{U}(C) = 0$. We say that C is **finitely generated** (or polyhedral) if $C = \mathbb{Q}^{\geq 0}\langle G \rangle$ for some finite set G.

Let X be a finite set of variables. The set of functions $\mathbb{Q}^X$ mapping variables to rationals is a linear space (of dimension $|X|$). A **polyhedron** in $\mathbb{Q}^X$ is of the form $P = \left\{ x \in \mathbb{Q}^X : p_1(x) \geq 0, \ldots, p_n(x) \geq 0 \right\}$ for some linear polynomials $p_1, \ldots, p_n \in \mathbb{Q}[X]^1$. We say that inequality $p(x) \geq 0$ is **valid** for a polyhedron P if it is satisfied by every point in P. The set of valid inequalities of a polyhedron $P = \left\{ x \in \mathbb{Q}^X : p_1(x) \geq 0, \ldots, p_n(x) \geq 0 \right\}$ forms a cone; by Farkas' lemma [Schrijver 1999, Cor 7.ld] this cone is precisely $\mathbb{Q}^{\geq 0}\langle p_1, \ldots, p_n, 1 \rangle$. The **integer hull** P_I of a polyhedron P is defined to be $P_I \triangleq conv\left(P \cap \mathbb{Z}^X\right)$, the convex hull of the integer points of P. The integer hull of a polyhedron P in $\mathbb{Q}^X$ is itself a polyhedron. A *dual view* of the integer hull of a polyhedron is the *cutting plane closure* of its valid inequalities [Chvátal 1973], and this can be computed from the constraints of P by the *iterated Gomory-Chvátal closure* process [Schrijver 1999, Ch. 23]. For an example illustrating the intuition behind cutting planes, suppose that we know that $2x - 1 \geq 0$, and that x takes integer values—then we must have $x - \frac{1}{2} \geq 0$, and thus $\left\lfloor x - \frac{1}{2} \right\rfloor = x + \left\lfloor -\frac{1}{2} \right\rfloor = x - 1 \geq 0$. That is, we may "shift" the halfspace to eliminate some real solutions while keeping all integer points. A set $C \subseteq \mathbb{Q}[X]^1$ is *closed under cutting planes* iff for any $n, m \in \mathbb{Z}$ with $n > 0$ and any $p \in \mathbb{Z}[X]$ such that $np + m \in C$, we have $p + \lfloor m/n \rfloor \in C$. We use $cp(C)$ to denote the cutting plane closure of a cone C—the smallest cone that contains C and which is closed under cutting planes. Assuming that C is finitely generated, $cp(C)$ can be computed by any algorithm for computing the integer hull of convex polyhedra.

Let V be a linear space. We say that a subset $L \subseteq V$ is a **point lattice** if there exists some $v_1, \ldots, v_n \in V$ such that $L = \mathbb{Z}\langle v_1, \ldots, v_n \rangle$. We call a set of generators $\{v_1, \ldots, v_n\}$ for a point lattice

L a **basis** for L if it is linearly independent. A basis for a point lattice can be computed as the Hermite normal form of its generators in polynomial time [Schrijver 1999, Ch. 4].

3 LINEAR REAL RINGS

In this section, we develop the theory of *linear real rings*, **LRR**. Linear real rings are a common extension of the theory of commutative rings and the positive fragment of the theory of linear real arithmetic. Section 3.1 presents the axioms of the theory and introduces *regular* and *algebraic* cones. Regular cones correspond to models of the theory, and algebraic cones correspond to effective structures; cones that are both regular and algebraic correspond to a class of effective models of **LRR**. Section 3.2 presents a decision procedure for satisfiability of ground formulas modulo **LRR**. Section 3.3 gives a procedure that discovers all implied inequalities of a ground formula modulo **LRR**, represented as a (regular) algebraic cone.

3.1 Linear Real Rings

Define **LRR** to be the σ_{or}-theory axiomatized by the axioms of commutative rings **CR** as well as the following theorems of **LRA**:

$$\begin{array}{lr}
\forall x.\, x \le x & \text{(Reflexivity)} \\
\forall x, y, z.\, x \le y \land y \le z \Rightarrow x \le z & \text{(Transitivity)} \\
\forall x, y, z.\, x \le y \land y \le x \Rightarrow x = y & \text{(Antisymmetry)} \\
\forall x, y, z.\, (x \le y \Rightarrow x + z \le y + z) & \text{(Compatibility)} \\
0 \le 1 \land 0 \ne 1 & \text{(Non-triviality)} \\
\text{for all } n \in \mathbb{Z}^{\ge 1}, \exists x.\, \underbrace{x + \cdots + x}_{n \text{ times}} = 1 & \text{(Divisibility)} \\
\text{for all } n \in \mathbb{Z}^{\ge 0}, \forall x.\, 0 \le \underbrace{x + \cdots + x}_{n \text{ times}} \Rightarrow 0 \le x & \text{(Perforation-freeness)}
\end{array}$$

Note that divisibility and perforation-freeness are axiom schemata, with one axiom for each natural number ($\exists x.\, x = 1$, $\exists x.\, x + x = 1$, $\exists x.\, x + x + x = 1$, and so on). Equivalently, **LRR** is the theory consisting of all σ_{or}-sentences that hold in all structures $\mathfrak{A}$ where $(U^{\mathfrak{A}}, 0^{\mathfrak{A}}, 1^{\mathfrak{A}}, +^{\mathfrak{A}}, \cdot^{\mathfrak{A}})$ forms a commutative ring and where $(U^{\mathfrak{A}}, 0^{\mathfrak{A}}, +^{\mathfrak{A}}, \le^{\mathfrak{A}})$ forms an unperforated partially ordered divisible group. We regard the real numbers $\mathbb{R}$ as the "standard" model of **LRR**; other models include the rationals, the complex numbers (where complex numbers with the same imaginary part are ordered by their real part) and, as we shall see shortly, more exotic interpretations. The signature of ordered rings should be regarded as a "minimal" signature for **LRR**, but models of **LRR** can be lifted to richer languages. In the following, we will make use of an extended language that includes atomic formulas $p \le q$ and $p = q$ where p and q are polynomials with rational coefficients; any such atom can be translated into the language of ordered rings by scaling and re-arranging terms (for example, the formula $0 \le \frac{1}{2}x - y$ can be translated to the $\sigma_{or}(\{x, y\})$ formula $(1 + 1) \cdot y \le x$).

The axioms that **LRR** adds onto commutative rings are a subset of the axioms of **LRA**. Naturally we might ask if the subset is "enough." Notably, totality of the order (an axiom of linear real arithmetic) is independent of **LRR** (e.g., $\le$ is non-total for the complex numbers). Nevertheless, **LRR** is at least as strong as **LRA**, in the sense that satisfiability modulo **LRA** can be reduced to satisfiability modulo **LRR** (noting that every formula in the language of **LRA** is equisatisfiable to a negation-free formula modulo **LRA**).

Theorem 1. Let F be a ground negation-free formula in the language of **LRA**. Then F is satisfiable modulo **LRA** iff it is satisfiable modulo **LRR**.

Proof. The $\Rightarrow$ direction is trivial, since the reals are a model of **LRR**. For the $\Leftarrow$ direction, we show the contrapositive: suppose that F is unsatisfiable modulo **LRA**, and show that it is unsatisfiable modulo **LRR**.

First observe that for any **LRR**-model $\mathfrak{A}$, and any terms c_1, d_1, c_2, d_2 such that $\mathfrak{A} \models c_1 \leq d_1$ and $\mathfrak{A} \models c_2 \leq d_2$, we have $\mathfrak{A} \models c_1 + c_2 \leq d_1 + d_2$. Since $\mathfrak{A} \models c_1 \leq d_1$, we have $\mathfrak{A} \models c_1 + c_2 \leq d_1 + c_2$ by compatibility; since $\mathfrak{A} \models c_2 \leq d_2$, we have $F \models d_1 + c_2 \leq d_1 + d_2$ by compatibility and commutativity. Finally, by transitivity we have $\mathfrak{A} \models c_1 + c_2 \leq d_1 + d_2$.

Without loss of generality, we may suppose that F is conjunction of inequalities. For convenience, we first assume that all inequalities are non-strict. Thus F can be written in the form $A\vec{x} \leq \vec{b}$. Since this system is unsatisfiable modulo **LRA**, then by Farkas' lemma there is some $\vec{y} \geq 0$ such that $\vec{y}^T A = 0$ and $\vec{y}^T \vec{b} < 0$. Without loss of generality, we may suppose that $\vec{y}$ is an integer vector. It follows from the observation above that $F \models_{\mathbf{LRR}} 0 \leq b$. Since $\leq$ is unperforated, we have $F \models_{\mathbf{LRR}} 0 \leq -1$, and by non-triviality and antisymmetry we have $F \models_{\mathbf{LRR}}$ *false*; i.e., F is unsatisfiable modulo **LRR**.

Now consider the case of disequalities (and thus formulas with strict inequalities, treating $p < q$ as an abbreviation for $p \leq q \wedge p \neq q$). In this case, we may suppose without loss of generality that F is conjunction of inequalities and disequalities, which can be written in the form $A\vec{x} \leq \vec{b} \wedge \bigwedge_{i=1}^{n} \vec{c}_i^T \vec{x} \neq d_i$ (with at least one disequality, or else we fall into the case above). Since F is unsatisfiable, we must have $A\vec{x} \leq \vec{b} \models_{\mathbf{LRA}} \bigvee_{i=1}^{n} \vec{c}_i^T \vec{x} = d_i$. Since **LRA** is a convex theory, we must have $A\vec{x} \leq \vec{b} \models_{\mathbf{LRA}} \vec{c}_i^T \vec{x} = d_i$ for some i. We may then argue as above that $A\vec{x} \leq \vec{b} \models_{\mathbf{LRR}} \vec{c}_i^T \vec{x} \leq d_i$ and $A\vec{x} \leq \vec{b} \models_{\mathbf{LRR}} d_i \leq \vec{c}_i^T \vec{x}$, and so by antisymmetry $A\vec{x} \leq \vec{b} \models_{\mathbf{LRR}} \vec{c}_i^T \vec{x} = d_i$, and thus F is unsatisfiable modulo **LRR**. □

In the remainder of this section, we develop a model theory of **LRR**, based on *regular cones*. For any set of variables X, we say that a set $C \subseteq \mathbb{Q}[X]$ is a **regular cone** if it is a cone (closed under addition and multiplication by non-negative rationals), $1 \in C$, and $\mathcal{U}(C)$ forms an ideal in $\mathbb{Q}[X]$. We say that C is **consistent** if $C \neq \mathbb{Q}[X]$; in the case that C is regular, C is consistent iff $-1 \notin C$. Consistent cones and regular cones are both closed under intersection, the latter is because ideals are closed under intersection.

Let X be a set of symbols, and let $\mathfrak{A}$ be a $\sigma_{or}(X)$-structure satisfying the axioms of **LRR**. Define $C(\mathfrak{A}) \triangleq \{p \in \mathbb{Q}[X] : \mathfrak{A} \models 0 \leq p\}$ to be its non-negative consequences. Naturally, $C(\mathfrak{A})$ forms a consistent regular cone. We now show that, conversely, any consistent regular cone can be associated with a model of **LRR**.

Let X be a set of variables, let $C \subseteq \mathbb{Q}[X]$ be a regular cone. Define a $\sigma_{or}(X)$-structure $\mathfrak{M}(C)$ where

- The universe and function symbols $0, 1+, \cdot$ are interpreted as in the quotient ring $R \triangleq \mathbb{Q}[X]/I$, where I is the ideal of additive units $\mathcal{U}(C)$. Thus, elements of the universe are sets of polynomials with rational coefficients of the form $p + I$, where $p \in \mathbb{Q}[X]$.
- Each constant symbol $x \in X$ is interpreted as $x + I$.
- $\leq$ is interpreted as the relation $\{(p + I, q + I) : q - p \in C\}$.

Observe that $\mathfrak{M}(C) \models 0 \leq q$ iff $q \in C$, and $\mathfrak{M}(C) \models 0 = q$ iff $q \in \mathcal{U}(C)$.

Lemma 1. Let X be a set of variables, and let $C \subseteq \mathbb{Q}[X]$ be a consistent regular cone. Then $\mathfrak{M}(C)$ is a model of **LRR**.

Proof. Clearly $R = \mathbb{Q}[X]/\mathcal{U}(C)$ is a commutative ring and satisfies the divisibility axioms. Since C is a cone, it follows that $\leq$ is reflexive, transitive, compatible with addition, and perforation-free; furthermore since C is regular we have that $\leq$ is antisymmetric. Non-triviality follows from the fact that C is consistent. □

While regular cones give us a "standard form" in which **LRR** models can be represented, they cannot be manipulated effectively. For this purpose, we introduce *algebraic cones*, which are (not necessarily regular) cones that admit a finite representation.

We say that a cone $C \subseteq \mathbb{Q}[X]$ is **algebraic** if there is an ideal I and a finitely-generated cone D such that $C = I + D$. An algebraic cone can be represented as a pair (Z, P) (with $Z, P \subseteq \mathbb{Q}[X]$) where $Z = \{z_1, \dots, z_m\}$ ("zeros") is a basis for an ideal and $P = \{p_1, \dots, p_n\}$ ("positives") is a basis for a cone; the algebraic cone represented by (Z, P) is denoted by

$$alg.cone_X(Z, P) \triangleq \langle Z \rangle + \mathbb{Q}^{\geq 0}\langle P \rangle = \left\{ \sum_{i=1}^{m} q_i z_i + \sum_{j=1}^{n} \lambda_j p_j : q_1, \dots, q_m \in \mathbb{Q}[X], \lambda_1, \dots, \lambda_n \in \mathbb{Q}^{\geq 0} \right\}$$

We will omit the X subscript when it is clear from context. Say that the pair (Z, P) is **oriented** (with respect to a monomial ordering $\preceq$) if:

(1) Z is a Gröbner basis for $\langle Z \rangle$ (with respect to $\preceq$), and
(2) Each $p_i \in P$ is reduced with respect to Z (i.e., $\mathbf{red}_Z(p_i) = p_i$ for all i).

The following shows that the problem of checking membership in an algebraic cone can be reduced to checking membership in a finitely-generated cone (which can be checked in polytime using linear programming). It comes in two parts: (1) checking membership assuming an *oriented* representation (Lemma 2) and (2) computing an oriented representation (Lemma 3).

Lemma 2 (Membership)**.** Let X be a set of variables, and $Z, P \subseteq \mathbb{Q}[X]$ be finite sets of polynomials such that (Z, P) is oriented. For any polynomial $q \in \mathbb{Q}[X]$, we have $q \in alg.cone(Z, P)$ iff $\mathbf{red}_Z(q) \in \mathbb{Q}^{\geq 0}\langle P \rangle$.

Proof. ($\Leftarrow$) If $\mathbf{red}_Z(q) \in \mathbb{Q}^{\geq 0}\langle P \rangle$ then since $(q - \mathbf{red}_Z(q)) \in \langle Z \rangle$ we have $q = (q - \mathbf{red}_Z(q)) + \mathbf{red}_Z(q) \in \langle Z \rangle + \mathbb{Q}^{\geq 0}\langle P \rangle = alg.cone(Z, P)$.

($\Rightarrow$) Suppose $q \in alg.cone(Z, P)$. Then we have $q = z + p$ for some $z \in \langle Z \rangle$ and $p \in \mathbb{Q}^{\geq 0}\langle P \rangle$. Since $z \in \langle Z \rangle$, we have $\mathbf{red}_Z(z) = 0$, and since p is a (non-negative) linear combination of polynomials that are reduced w.r.t. Z, we have $\mathbf{red}_Z(p) = p$. It follows that $\mathbf{red}_Z(q) = \mathbf{red}_Z(z) + \mathbf{red}_Z(p) = p$, and so $\mathbf{red}_Z(q) \in \mathbb{Q}^{\geq 0}\langle P \rangle$. □

For any set of variables X, finite sets of polynomials $Z, P \subseteq \mathbb{Q}[X]$, and monomial ordering $\preceq$, define $orient_{\preceq}(Z, P) \triangleq (G, \{\mathbf{red}_G(p) : p \in P, \mathbf{red}_G(p) \neq 0\})$ where $G = GröbnerBasis_{\preceq}(Z)$ is a Gröbner basis for $\langle Z \rangle$ with respect to the order $\preceq$.

Lemma 3 (Orientation)**.** Let X be a set of variables, $Z, P \subseteq \mathbb{Q}[X]$ be finite sets of polynomials, and $\preceq$ be a monomial ordering. Then $orient_{\preceq}(Z, P)$ is oriented with respect to $\preceq$ and $alg.cone(Z, P) = alg.cone(orient_{\preceq}(Z, P))$.

Proof. Let $G = GröbnerBasis_{\preceq}(Z)$ and $P' = \{\mathbf{red}_G(p) : p \in P, \mathbf{red}_G(p) \neq 0\}$. Since $\mathbf{red}_G$ is idempotent, $orient_{\preceq}(G, P')$ is oriented w.r.t. $\preceq$. Clearly, $\langle Z \rangle = \langle G \rangle$. Since algebraic cones are closed under addition and multiplication by non-negative rationals, it is sufficient to prove that $P \subseteq \langle G \rangle + \mathbb{Q}^{\geq 0}\langle P' \rangle$ and $P' \subseteq \langle Z \rangle + \mathbb{Q}^{\geq 0}\langle P \rangle$.

- $P \subseteq \langle G \rangle + \mathbb{Q}^{\geq 0}\langle P' \rangle$: Suppose $p \in P$. Since $p - \mathbf{red}_G(p) \in \langle G \rangle$ and $\mathbf{red}_G(p) \in \mathbb{Q}^{\geq 0}\langle P' \rangle$ (since it either belongs to P' or it is zero), we have $p = (p - \mathbf{red}_G(p)) + \mathbf{red}_G(p) \in \langle G \rangle + \mathbb{Q}^{\geq 0}\langle P' \rangle$.
- $P' \subseteq \langle Z \rangle + \mathbb{Q}^{\geq 0}\langle P \rangle$: Suppose $p' \in P$. Then $p' = \mathbf{red}_G(p)$ for some $p \in P$. It follows that $p' - p \in \langle G \rangle = \langle Z \rangle$, and so $p' = (p' - p) + p \in \langle Z \rangle + \mathbb{Q}^{\geq 0}\langle P \rangle$. □

As a consequence of decidability of membership in algebraic cones, we have a model checking procedure for models associated to algebraic cones. Given a ground formula F and generators Z, P for an algebraic cone, checking $\mathfrak{M}(alg.cone(Z, P)) \models F$ is decidable.

Function *regularize*(Z, P)
  Input : Finite sets of polynomials Z and P
  Output : Oriented pair (Z', P') such that $\mathbb{Q}^{\geq 0}\langle P' \rangle$ is salient and $alg.cone(Z', P')$ is the least regular cone that contains $alg.cone(Z, P)$
  $P \leftarrow P \cup \{1\}$;
  while *there is some non-zero* $t \in \mathcal{U}(\mathbb{Q}^{\geq 0}\langle P \rangle)$ **do**
    /* *Sampling from* $\mathcal{U}(\mathbb{Q}^{\geq 0}\langle P \rangle)$ *can be implemented by (e.g.) linear programming* */
    $(Z, P) \leftarrow orient_{\leq}(Z \cup \{t\}, P)$;
  return (Z, P)

Algorithm 1: Saturation

3.2 Satisfiability Modulo LRR

This section presents a decision procedure for testing satisfiability of ground $\sigma_{or}(X)$-formulas modulo the theory **LRR**. As usual, it is sufficient to develop a *theory solver*, which can test satisfiability of the conjunctive fragment; formulas with disjunctions can be accommodated using DPLL($\mathcal{T}$) [Ganzinger et al. 2004].

Without loss of generality, a ground conjunctive $\sigma_{or}(X)$-formula F can be written in the form

$$F = \left(\bigwedge_{p \in P} 0 \leq p \right) \wedge \left(\bigwedge_{q \in Q} \neg(0 \leq q) \right) \wedge \left(\bigwedge_{r \in R} \neg(0 = r) \right)$$

where P, Q, and R are finite sets of polynomials (noting the equivalences $x \leq y \equiv 0 \leq y - x$ and $x = y \equiv 0 \leq x - y \wedge 0 \leq y - x$). In the following, we will first show that it is possible to compute a finite representation of the *least regular cone* C that contains all of the non-negative polynomials P (Theorem 3), and then show that F is satisfiable if and only if C is consistent and $\mathfrak{M}(C) \models F$ (Theorem 4). Since C is algebraic and computable from F, and checking that C is consistent and that $\mathfrak{M}(C) \models F$ is decidable, this yields a sound and complete procedure for checking satisfiability of ground $\sigma_{or}(X)$-formulas modulo **LRR**.

We first show that we can compute the sum of two algebraic cones by combining their bases for the zeros and the positives, before proving the correctness of Algorithm 1.

Theorem 2 (Sum). Let $Z_1, P_1, Z_2, P_2 \subseteq \mathbb{Q}[X]$ be finite sets of polynomials. Then

$$alg.cone(Z_1, P_1) + alg.cone(Z_2, P_2) = alg.cone(Z_1 \cup Z_2, P_1 \cup P_2)\ .$$

Theorem 3. Let X be a finite set of variables. For any finite sets of polynomials $Z, P \subseteq \mathbb{Q}[X]$, *regularize*$(Z, P)$ (Algorithm 1) returns a pair (Z', P') such that $\mathbb{Q}^{\geq 0}\langle P' \rangle$ is salient and $alg.cone(Z', P')$ is the least regular cone that contains $alg.cone(Z, P)$.

Proof. Let Z_i, P_i, and t_i denote the values of Z, P, and t after i iterations of the loop in Algorithm 1. We first observe an invariant of the algorithm: for all i, we have

$$\begin{aligned} alg.cone(Z_{i+1}, P_{i+1}) &= alg.cone(orient_{\leq}(Z_i \cup \{t\}, P_i)) && \text{Definition} \\ &= alg.cone(Z_i \cup \{t\}, P_i) && \text{Lemma 3} \\ &= alg.cone(Z_i, P_i) + \langle t \rangle && \text{Theorem 2} \end{aligned}$$

We first prove that the algorithm terminates. For iteration $i + 1$, we have $\langle Z_{i+1} \rangle = \langle Z_i \rangle + \langle t_i \rangle$. Since t_i is a conic combination of polynomials in P_i which are reduced with respect to Z_i, we have $\mathbf{red}_{Z_i}(t_i) = t_i \neq 0$ and so $t_i \notin \langle Z_i \rangle$. Hence we have $\langle Z_{i+1} \rangle \supsetneq \langle Z_i \rangle$. For a contradiction, suppose that the algorithm does not terminate. Then we have an infinite strictly ascending chain of ideals

Table 1. Execution of Algorithm 1 on input $Q = \{x^2 - xy, xy - x^2, x^2y - z, w - xy^2, z - w, w^3\}$.

Iteration	Z	P	Additive unit
0	$\emptyset$	$\{1\} \cup Q$	$x^2 - xy$
1	$\{xy - x^2\}$	$\{1, x^3 - z, w - x^3, z - w, w^3\}$	$x^3 - z$
2	$\{xy - x^2, x^3 - z, yz - xz\}$	$\{1, w - z, z - w, w^3\}$	$w - z$
3	$\{xy - x^2, x^3 - z, yz - xz, w - z\}$	$\{1, z^3\}$	–

$\langle Z_0 \rangle \subsetneq \langle Z_1 \rangle \subsetneq \dots$ in $\mathbb{Q}[X]$, which contradicts the fact that $\mathbb{Q}[X]$ is a Noetherian ring [Cox et al. 2015, Ch. 2 §5].

Suppose that the loop terminates after n iterations—we must show that $\mathit{alg.cone}(\mathit{regularize}(Z_n, P_n))$ is the least regular cone that contains $\mathit{alg.cone}(Z, P)$. Since $Z_0 = Z$ and $P_0 = P \cup \{1\}$, and we have $\mathit{alg.cone}(Z_i, P_i) \subseteq \mathit{alg.cone}(Z_{i+1}, P_{i+1})$ for all i, we have that $\mathit{alg.cone}(\mathit{regularize}(Z_n, P_n))$ must contain $\mathit{alg.cone}(Z, P)$ and 1. By the termination condition, we have that $\mathbb{Q}^{\geq 0}\langle P_n \rangle$ is salient, and therefore $\mathit{alg.cone}(Z_n, P_n)$ is regular. It remains to show that it is the *least* such regular cone. Suppose that there is another regular cone C with $\mathit{alg.cone}(Z, P) \subseteq C$. We show that for all iterations i, $\mathit{alg.cone}(Z_i, P_i) \subseteq C$ by induction on i. Initially this is true since $Z_0 = Z$ and $P_0 = Q \cup \{1\}$, and C contains $\mathit{alg.cone}(Z, P)$ (by assumption) and 1 (since C is regular). For the inductive step, we suppose that $\mathit{alg.cone}(Z_i, P_i) \subseteq C$ and prove that $\mathit{alg.cone}(Z_{i+1}, P_{i+1}) = \mathit{alg.cone}(Z_i, P_i) + \langle t_i \rangle \subseteq C$. Since C is closed under addition and $\mathit{alg.cone}(Z_i, P_i) \subseteq C$ by the inductive hypothesis, it is sufficient to show that $\langle t_i \rangle \subseteq C$. Since $t_i \in \mathcal{U}(\mathbb{Q}^{\geq 0}\langle P_i \rangle)$ and $\mathbb{Q}^{\geq 0}\langle P_i \rangle \subseteq C$, we must have $t_i \in \mathcal{U}(C)$. Since C is regular, $\mathcal{U}(C)$ is an ideal, and so $\langle t_i \rangle \subseteq \mathcal{U}(C) \subseteq C$. □

Example 3.1. Table 1 illustrates Algorithm 1 on the set of polynomials

$$Q = \{x^2 - xy, xy - x^2, x^2y - z, w - xy^2, z - w, w^3\} \ .$$

Each row i gives the set of zero polynomials Z and set of positive polynomials P at the beginning of iteration i, along with the selected additive unit t. Intuitively, each round selects an additive unit from the positives, removes it from the positives and adds it to the zeros. The algorithm terminates at iteration 3: the positive cone is salient, and so there is no additive unit to select. ⌟

The following theorem is the basis of our decision procedure for **LRR**: it shows that to test satisfiability of F, we only need to check whether the least cone C that agrees with all positive atoms of F is consistent and other atoms do not contradict the consequences of C.

Theorem 4. Let F be the ground conjunctive formula

$$F = \left(\bigwedge_{p \in P} 0 \leq p\right) \land \left(\bigwedge_{q \in Q} \neg(0 \leq q)\right) \land \left(\bigwedge_{r \in R} \neg(0 = r)\right).$$

Let C be the least regular cone that contains P. Then F is satisfiable modulo **LRR** iff C is consistent and $\mathfrak{M}(C) \models F$.

Proof. If C is consistent and $\mathfrak{M}(C) \models F$, then F is satisfiable modulo **LRR** ($\mathfrak{M}(C)$ is a model of F (by Lemma 1) of **LRR**). We thus only need to prove the other direction.

Suppose that F is satisfiable modulo **LRR**. Then there is some model $\mathfrak{A}$ of **LRR** with $\mathfrak{A} \models F$. We have that $C \subseteq C(\mathfrak{A})$, since $C(\mathfrak{A})$ is a regular cone that contains P (since $\mathfrak{A} \models F$), and C is the least such cone. It follows that C is consistent, since if $-1 \in C$ then $-1 \in C(\mathfrak{A})$, which is not possible because $C(\mathfrak{A})$ is consistent. It remains to show that $\mathfrak{M}(C) \models F$. Clearly $\mathfrak{M}(C) \models 0 \leq p$ for all $p \in P$, since $P \subseteq C$. For any $q \in Q$ we have $\mathfrak{A} \models \neg(0 \leq q)$, since $\mathfrak{A} \models F$. Thus $q \notin C(\mathfrak{A})$ and so $q \notin C$

since $C \subseteq C(\mathfrak{A})$. Hence $\mathfrak{M}(C) \models \neg(0 \leq q)$. Similarly, we have that $\mathfrak{M}(C) \models \neg(0 = r)$ for all $r \in R$. Combining the above, we have $\mathfrak{M}(C) \models F$. □

Decision Procedure for **LRR**. Summarizing, we have the following decision procedure for satisfiability of ground conjunctive $\sigma_{or}(X)$-formulas modulo **LRR**. Let F be of the form in Theorem 4. First compute a representation (Z', P') of the least regular cone containing P using Algorithm 1. If $\mathbf{red}_{Z'}(1) = 0$, then F is unsatisfiable ($alg.cone(Z', P')$ is inconsistent). Otherwise, check whether $\mathfrak{M}(alg.cone(Z', P')) \models F$ by testing whether there is some $q \in Q$ with $\mathbf{red}_{Z'}(q) \in \mathbb{Q}^{\geq 0}\langle P' \rangle$ (Lemma 2), or some $r \in R$ with $\mathbf{red}_{Z'}(r) = 0$; if such a q or r exists, then F is unsatisfiable (Theorem 4), otherwise, $\mathfrak{M}(alg.cone(Z', P'))$ satisfies F.

3.3 Consequence-finding modulo LRR

This section describes an algorithm that computes all polynomial inequalities that are entailed by a ground formula modulo **LRR**. Let X be a set of symbols and let F be a $\sigma_{or}(X)$-formula. Define the **non-negative cone** $\mathbf{C}_X(F)$ of F as follows:

$$\mathbf{C}_X(F) \triangleq \{p \in \mathbb{Q}[X] : F \models_{\mathbf{LRR}} 0 \leq p\} \ .$$

It is easy to verify that for any formula F, $\mathbf{C}_X(F)$ is a regular cone. We will omit the X subscript when it can be understood from the context.

A simple "eager" strategy for computing non-negative cones operates as follows. Suppose that F is a ground $\sigma_{or}(Y)$-formula and that $X \subseteq Y$. First, we place F in disjunctive normal form; i.e., we compute a formula that is equivalent to F and takes the form $\bigvee_{i=1}^{n} G_i$ where each G_i is a conjunctive formula. Observe that

$$\mathbf{C}_X(F) = \mathbf{C}_X\left(\bigvee_{i=1}^{n} G_i\right) = \bigcap_{i=1}^{n} \mathbf{C}_X(G_i) = \bigcap_{i=1}^{n} (\mathbf{C}_Y(G_i) \cap \mathbb{Q}[X])$$

since a disjunctive formula entails that a polynomial is non-negative exactly when each disjunct does. Thus, the problem of computing $\mathbf{C}_X(F)$ can be reduced to three sub-problems: (1) computing the (regular and algebraic) non-negative cone of a *conjunctive* formula, (2) projection of an algebraic cone onto a subset of symbols (i.e., the intersection of an algebraic cone with $\mathbb{Q}[X]$), and the intersection of algebraic cones. In the following, we show how to solve each sub-problem, and then present a "lazy" variant of the consequence-finding procedure that avoids (explicit) DNF computation.

3.3.1 Non-Negative Cones of Conjunctive Formulas. This section addresses the following problem: given a ground $\sigma_{or}(Y)$-formula F, compute a representation of the cone $\mathbf{C}_Y(F)$. In fact, the solution to this problem is immediate: $\mathbf{C}_Y(F)$ coincides with the least regular cone that contains the non-negative atoms of F, which can be computed by the *regularize* procedure (Algorithm 1):

Lemma 4. Let Y be a set of symbols and let

$$F = \left(\bigwedge_{p \in P} 0 \leq p\right) \land \left(\bigwedge_{q \in Q} \neg(0 \leq q)\right) \land \left(\bigwedge_{r \in R} \neg(0 = r)\right)$$

be a ground $\sigma_{or}(Y)$-formula. Then $alg.cone_Y(regularize(\emptyset, P)) = \mathbf{C}_Y(F)$.

Proof. Let $C = alg.cone_Y(regularize(\emptyset, P))$. By Theorem 3, C is the least regular cone that contains P. Since $\mathbf{C}_Y(F)$ is a regular cone that contains P, we have $C \subseteq \mathbf{C}_Y(F)$. It remains only to show that $\mathbf{C}_Y(F) \subseteq C$. If C is inconsistent (i.e., $C = \mathbb{Q}[Y]$), then $\mathbf{C}_Y(F) \subseteq C$ is immediate. Otherwise, $\mathfrak{M}(C) \models F$ by Theorem 4. For any $q \in \mathbf{C}_Y(F)$, we have $F \models_{\mathbf{LRR}} 0 \leq q$ and so $\mathfrak{M}(C) \models 0 \leq q$ and therefore $q \in C$. □

3.3.2 Projection of Algebraic Cones. This section addresses the following problem: given finite sets $Z, P \subseteq \mathbb{Q}[Y]$ and a subset $X \subseteq Y$, compute $Z', P' \subseteq \mathbb{Q}[X]$ such that $\mathit{alg.cone}(Z', P') = \mathit{alg.cone}(Z, P) \cap \mathbb{Q}[X]$. First, we review standard methods for solving this problem for polynomial ideals and finitely-generated cones (that is, the case when P is empty, and the case where Z is empty). The algorithm for algebraic cones is a combination of the two.

Any monomial m over variables Y can be regarded as the product of two monomials $m_X m_{\overline{X}}$ where m_X is a monomial over X and $m_{\overline{X}}$ is a monomial over $Y \setminus X$. For any monomial ordering $\preceq$, we may define an *elimination ordering* $\preceq_X$ where $m_X m_{\overline{X}} \preceq_X n_X n_{\overline{X}}$ iff $m_{\overline{X}} \prec n_{\overline{X}}$ or $m_{\overline{X}} = n_{\overline{X}}$ and $m_X \preceq n_X$. The classical algorithm for ideal projection computes a basis for $\mathbb{Q}[Y]\langle Z\rangle \cap \mathbb{Q}[X]$ by computing a Gröbner basis G for Z with respect to the order $\preceq_X$, and then taking $G \cap \mathbb{Q}[X]$ [Cox et al. 2015, Ch. 3]. By the *ordering* and *membership* properties of Gröbner bases, if $p \in \mathbb{Q}[Y]\langle Z\rangle = \mathbb{Q}[Y]\langle G\rangle$, then $p = q_1 g_1 + \cdots + q_n g_n$ for some $q_1, \ldots, q_n \in \mathbb{Q}[Y]$ and $g_1, \ldots, g_n \in G$ with $\textsc{Lm}(q_i g_i) \preceq_X \textsc{Lm}(p)$ for all i. Supposing that p is also in $\mathbb{Q}[X]$, each q_i and g_i must also be in $\mathbb{Q}[X]$.

We now turn to the case of finitely-generated cones. Since P is a finite collection of polynomials, there is a finite set of monomials that appear in any polynomial in P, which we call M. Then we can see $\mathbb{Q}^{\geq 0}\langle P\rangle \cap \mathbb{Q}[X] = \mathbb{Q}^{\geq 0}\langle P\rangle \cap \mathbb{Q}\langle M \cap [X]\rangle$; and so our problem is to compute the intersection of a cone and a linear space—this is the *dual view* of the problem solved by polyhedral projection, for which there are several known algorithms. For the sake of completeness, we will describe how to apply Fourier-Motzkin elimination, which is one such algorithm. Suppose that we wish to compute $\mathbb{Q}^{\geq 0}\langle P\rangle \cap \mathbb{Q}\langle N\rangle$ for some set of monomials N. Fourier-Motzkin elimination proceeds by eliminating one dimension, a monomial $m \in M \setminus N$, at a time. First, we may normalize P so that every polynomial in P takes the form $p + am$, where m does not appear in p, and a is either 0, 1, or -1 (by multiplying each polynomial with an appropriate non-negative scalar). Then, take $Q = \{p : p + 0m \in P\} \cup \{p + q : p + m \in P, q - m \in P\}$. $\mathbb{Q}^{\geq 0}\langle Q\rangle$ is precisely $\mathbb{Q}^{\geq 0}\langle P\rangle \cap \mathbb{Q}\langle M \setminus m\rangle$, since any non-negative combination of elements of P that results in a coefficient of 0 for m is also a non-negative combination of elements of Q. Repeating this process for each monomial in $M \setminus N$, we get a finite set of polynomials that we denote $\overline{\mathit{project}}_N(P)$, with $\mathbb{Q}^{\geq 0}\left\langle \overline{\mathit{project}}_N(P)\right\rangle = \mathbb{Q}^{\geq 0}\langle P\rangle \cap \mathbb{Q}\langle N\rangle$.

Finally, we put the two pieces together, by observing that we can project an algebraic cone $\mathit{alg.cone}(Z, P)$ by separately projecting the ideal $\langle Z\rangle$ and cone $\mathbb{Q}^{\geq 0}\langle P\rangle$, *provided that* (Z, P) *is oriented with respect to the elimination ordering*. That is, we define

$$\mathit{project}_X(Z, P) \triangleq \left(G \cap \mathbb{Q}[X], \overline{\mathit{project}}_{[X]}(\{\mathbf{red}_G(p) : p \in P\})\right)$$

where G is a Gröbner basis for $\langle Z\rangle$ w.r.t. the elimination order $\preceq_X$.

Theorem 5 (Projection)**.** Let $Z, P \subseteq \mathbb{Q}[Y]$ be finite sets of polynomials, and let $X \subseteq Y$. Then

$$\mathit{alg.cone}_X(\mathit{project}_X(Z, P)) = \mathit{alg.cone}_Y(Z, P) \cap \mathbb{Q}[X] .$$

Proof. Let G be a Gröbner basis for $\mathbb{Q}[Y]\langle Z\rangle$ w.r.t. the elimination order $\preceq_X$, let $Z' = G \cap \mathbb{Q}[X]$, and let $P' = \overline{\mathit{project}}_{[X]}(\mathbf{red}_G(P))$, where $\mathbf{red}_G(P)$ denotes the set $\{\mathbf{red}_G(p) : p \in P\}$. By Lemma 3, we have $\mathit{alg.cone}_Y(Z, P) = \mathit{alg.cone}_Y(G, \mathbf{red}_G(P))$. We only need to show that $\mathit{alg.cone}_X(Z', P') = \mathit{alg.cone}_Y(G, \mathbf{red}_G(P)) \cap \mathbb{Q}[X]$. We know that $\mathit{alg.cone}_X(Z', P') \subseteq \mathit{alg.cone}_Y(G, \mathbf{red}_G(P)) \cap \mathbb{Q}[X]$ since $Z' = G \cap \mathbb{Q}[X]$ is a basis for $\mathbb{Q}[Y]\langle G\rangle \cap \mathbb{Q}[X]$ and $\mathbb{Q}^{\geq 0}\langle P'\rangle = \mathbb{Q}^{\geq 0}\left\langle \overline{\mathit{project}}_{[X]}(\mathbf{red}_G(P))\right\rangle = \mathbb{Q}^{\geq 0}\langle \mathbf{red}_G(P)\rangle \cap \mathbb{Q}[X]$. Thus we only need to show the other direction.

Suppose that $q \in \mathit{alg.cone}_Y(G, \mathbf{red}_G(P)) \cap \mathbb{Q}[X]$. Since $q \in \mathbb{Q}[X]$ and $\textsc{Lm}(\mathbf{red}_G(q)) \preceq_X \textsc{Lm}(q)$, we have $\mathbf{red}_G(q) \in \mathbb{Q}[X]$. Since $q \in \mathit{alg.cone}_Y(G, \mathbf{red}_G(P))$, we have $\mathbf{red}_G(q) \in \mathbb{Q}^{\geq 0}\langle \mathbf{red}_G(P)\rangle$ by the membership lemma (Lemma 2). It follows that $\mathbf{red}_G(q) \in \mathbf{red}_G(P) \cap \mathbb{Q}[X] = \mathbb{Q}^{\geq 0}\langle P'\rangle$. Since $q, \mathbf{red}_G(q) \in \mathbb{Q}[X]$ we have $q - \mathbf{red}_G(q) \in \mathbb{Q}[X]$ and thus $q - \mathbf{red}_G(q) \in \mathbb{Q}[Y]\langle G\rangle \cap \mathbb{Q}[X] = \mathbb{Q}[X]\langle Z'\rangle$. Thus we have $q = (q - \mathbf{red}_G(q)) + \mathbf{red}_G(q) \in \mathbb{Q}[X]\langle Z'\rangle + \mathbb{Q}^{\geq 0}\langle P'\rangle = \mathit{alg.cone}_X(Z', P')$. □

3.3.3 Intersection of Algebraic Cones. This section addresses the following problem: given finite sets $Z_1, P_1, Z_2, P_2 \subseteq \mathbb{Q}[X]$, compute $Z, P \subseteq \mathbb{Q}[X]$ such that $\mathit{alg.cone}(Z, P) = \mathit{alg.cone}(Z_1, P_1) \cap \mathit{alg.cone}(Z_2, P_2)$. The essential idea is to reduce the problem to a projection problem—essentially the same idea as the standard algorithm for ideal intersection [Cox et al. 2015, Ch. 4 §3] and the constraint-based algorithm for polyhedral join [Benoy et al. 2005].[1]

The essential idea is to introduce a parameter t that does not belong to X, and to "tag" each element of a cone $C_1 = \mathit{alg.cone}(Z_1, P_1)$ by multiplying by t, and to tag elements of $C_2 = \mathit{alg.cone}(Z_2, P_2)$ by multiplying by $1 - t$. If p is a polynomial that belongs to C_1 and C_2, then $tp + (1 - t)p = p$ belongs to their "tagged sum." This yields the following definition:

$$\mathit{intersect}(Z_1, P_1, Z_2, P_2) \triangleq \mathit{project}_X(tZ_1 \cup (1 - t)Z_2, tP_1 \cup (1 - t)P_2)$$

where the notation $pQ \triangleq \{pq : q \in Q\}$ (for a polynomial p and set of polynomials Q) denotes the "tagging" operation.

Example 3.2. Consider the regular cones

$$\begin{array}{rclcl} C_1 & = & \mathbf{C}\left(x = 1 \wedge y \leq 1\right) & = & \mathit{alg.cone}(\{x - 1\}, \{1, 1 - y\}) \\ C_2 & = & \mathbf{C}\left(y = 2 \wedge 2 \leq x^2\right) & = & \mathit{alg.cone}(\{y - 2\}, \left\{1, x^2 - 2\right\}) \end{array}$$

To intersect C_1 and C_2, form the "tagged sum" (Z, P) where

$$\begin{array}{rclcl} Z & = & \{t(x - 1), (1 - t)(y - 2)\} & = & \{tx - t, -ty + 2t + y - 2\} \\ P & = & \left\{t, t(1 - y), (1 - t), (1 - t)(x^2 - 2)\right\} & = & \left\{t, t - y, -t + 1, -tx^2 + 2t + x^2 - 2\right\} \end{array}$$

Then project (Z, P) onto the variables $\{x, y\}$, in two steps. First orient (Z, P) w.r.t $\preceq_{\{x,y\}}$

$$G = \mathit{Gr\ddot{o}bnerBasis}_{\preceq_{\{x,y\}}}(Z) = \{tx - t, -ty + 2t + y - 2, xy - 2x - y + 2\}$$

$$\mathbf{red}_G(P) = \left\{t, -t - y + 2, -t + 1, t + x^2 - 2\right\}$$

and then intersect G with $\mathbb{Q}[x, y]$ and project the monomial t out of $\mathbf{red}_G(P)$ using Fourier-Motzkin elimination. The resulting cone is

$$C_1 \cap C_2 = \mathit{alg.cone}(\{xy - 2x - y + 2\}, \left\{1, -y + 2, x^2 - y, x^2 - 1\right\}),$$

which is the non-negative cone of the formula $(x - 1) \cdot (y - 2) = 0 \wedge y \leq 2 \wedge y \leq x^2 \wedge 1 \leq x^2$. ⌟

To prove correctness of this construction, we need the following technical lemma relating cones to their "tagged" counterparts:

Lemma 5. Let X be a set of variables and $t \notin X$. Let $Z, P \subseteq \mathbb{Q}[X]$ be finite sets of polynomials, and let $f \in \mathbb{Q}[t]$ be a polynomial in t. Then for all $a \in \mathbb{Q}$ such that $f(a) \geq 0$, and for all $q \in \mathit{alg.cone}_{X,t}(fZ, fP)$, we have $q[t \mapsto a] \in \mathit{alg.cone}_X(Z, P)$ (where $q[t \mapsto a]$ denotes substitution of all occurrences of t in q with a).

Proof. Let $Z = \{z_1, \dots, z_m\}$ and $P = \{p_1, \dots, p_n\}$. For any $q \in \mathit{alg.cone}_{X,t}(fZ, fP)$ we can write

$$q = \sum_{i=1}^{m} g_i f z_i + \sum_{j=1}^{n} \lambda_j f p_j \quad (\forall i.\, q_i \in \mathbb{Q}[X, t], \lambda_i \in \mathbb{Q}^{\geq 0})$$

$$q[t \mapsto a] = \sum_{i=1}^{m} (g_i f(a))\, z_i + \sum_{j=1}^{n} \left(\lambda_j f(a)\right) p_j \in \mathit{alg.cone}(Z, P)$$

since each $g_i f(a)$ is a polynomial in X and each $\lambda_j f(a)$ is a non-negative rational. □

[1] Recalling that cones are dual to polyhedra, cone intersection corresponds to polyhedral join, and cone sum to polyhedral meet.

Theorem 6 (Intersection)**.** Let $Z_1, P_1, Z_2, P_2 \subseteq \mathbb{Q}[X]$ be finite sets of polynomials over some set of variables X. Then

$$\mathit{alg.cone}(\mathit{intersect}(Z_1, P_1, Z_2, P_2)) = \mathit{alg.cone}(Z_1, P_1) \cap \mathit{alg.cone}(Z_2, P_2)\ .$$

Proof. We prove each side of the equation is included in the other:

- $\subseteq$: Let $q \in \mathit{alg.cone}(\mathit{intersect}(Z_1, P_1, Z_2, P_2))$. Since $\mathit{intersect}(Z_1, P_1, Z_2, P_2) = \mathit{project}_X(tZ_1 \cup (1-t)Z_2, tP_1 \cup (1-t)P_2)$, we have $q \in \mathit{alg.cone}(tZ_1 \cup (1-t)Z_2, tP_1 \cup (1-t)P_2) \cap \mathbb{Q}[X]$ by Theorem 5. Then q can be written as $q_1 + q_2$ for some $q_1 \in \mathit{alg.cone}(tZ_1, tP_1)$ and $q_2 \in \mathit{alg.cone}((1-t)Z_2, (1-t)P_2)$. Then we have
$$\begin{aligned} q &= q[t \mapsto 0] && q \in \mathbb{Q}[X] \\ &= q_1[t \mapsto 0] + q_2[t \mapsto 0] && q = q_1 + q_2, \text{linearity of substitution} \\ &= q_2[t \mapsto 0] && t \text{ divides } q_1 \\ &\in \mathit{alg.cone}(Z_2, P_2) && \text{Lemma 5} \end{aligned}$$
Symmetrically, we have $q = q[t \mapsto 1] = q_1[t \mapsto 1] \in \mathit{alg.cone}(Z_1, P_1)$, and so q belongs to the intersection $\mathit{alg.cone}(Z_1, P_1) \cap \mathit{alg.cone}(Z_2, P_2)$.
- $\supseteq$: Let $q \in \mathit{alg.cone}(Z_1, P_1) \cap \mathit{alg.cone}(Z_2, P_2)$. We have $tq \in \mathit{alg.cone}(tZ_1, tP_1)$ and $(1-t)q \in \mathit{alg.cone}((1-t)Z_2, (1-t)P_2)$, and therefore
$$\begin{aligned} q = tq + (1-t)q &\in \mathit{alg.cone}_{X,t}(tZ_1, tP_1) + \mathit{alg.cone}_{X,t}((1-t)Z_2, (1-t)P_2) \\ &= \mathit{alg.cone}_{X,t}(tZ_1 \cup (1-t)Z_2, tP_1 \cup (1-t)P_2) && \text{Theorem 2} \end{aligned}$$
Since q also belongs to $\mathbb{Q}[X]$, we have $q \in \mathit{alg.cone}_{X,t}(tZ_1 \cup (1-t)Z_2, tP_1 \cup (1-t)P_2) \cap \mathbb{Q}[X] = \mathit{alg.cone}(\mathit{intersect}(Z_1, P_1, Z_2, P_2))$. □

3.3.4 Lazy Consequence-Finding. We conclude with a consequence-finding algorithm that avoids the (explicit) computation of disjunctive normal incurred by the eager strategy presented at the beginning of this section. Algorithm 2 operates by iteratively selecting a cube from the DNF of F, computing its non-negative cone (Lemma 4), and adding blocking clauses so that the same cube would not be selected again in future iterations.

Function $\mathit{consequence}(F, X)$
  Input : Ground $\sigma_{or}(Y)$-formula F and set of symbols $X \subseteq Y$
  Output : Oriented pair (Z, P) with $\mathit{alg.cone}_X(Z, P) = \mathbf{C}_X(F)$.
  $G \leftarrow F$;
  $(Z, P) \leftarrow (\{1\}, \emptyset)$; /* *Invariant:* $\mathbf{C}_X(F) \subseteq \mathit{alg.cone}(Z, P)$ */
  while G *is satisfiable* **do**
    /* $\mathit{alg.cone}(Z', P') = \mathbf{C}_Y(G_i)$ *for some cube* G_i *of the DNF of F (Lemma 4)* */
    $(Z', P') \leftarrow \mathit{get\text{-}model}(G)$;
    $(Z', P') \leftarrow \mathit{project}(Z', P', X)$;
    $(Z, P) \leftarrow \mathit{intersect}(Z, P, Z', P')$;
    /* *Block any model* $\mathfrak{A}$ *with* $\mathbf{C}_X(\mathfrak{A}) \subseteq \mathit{alg.cone}(Z, P)$ */
    $G \leftarrow G \wedge \neg \left((\bigwedge_{z \in Z} z = 0) \wedge \left(\bigwedge_{p \in P} 0 \leq p\right)\right)$
  return (Z, P)

Algorithm 2: Lazy consequence finding

Theorem 7. Given a ground $\sigma_{or}(Y)$-formula F and $X \subseteq Y$, $\mathit{consequence}(F, X)$ returns an oriented pair (Z, P) such that $\mathit{alg.cone}_X(Z, P) = \mathbf{C}_X(F)$.

Proof. Say that a regular cone C is a *cube cone* of formula H if $C = \mathbf{C}_X(H)$ for some cube in the DNF of H, and C is consistent. Let *cube.cone*(H) denote the (finite) set of cube cones of H.

We first prove termination. Let G_i denote the formula G on the ith iteration of the loop, and let B_i denote the ith blocking clause, so for each i we have $G_{i+1} = G_i \land \neg B_i$. By Theorem 4 and Lemma 4, we have that

$$\mathit{cube.cones}(G_{i+1}) = \mathit{cube.cones}(G_i \land \neg B_i) = \{Q \in \mathit{cube.cones}(G_i) : \mathfrak{M}(Q) \models \neg B_i\} \ .$$

Since the model returned by *get-model* is always in *cube.cone*(G_i) (Theorem 4 and Lemma 4) and that model always satisfies B_i, we have a strictly descending sequence $\mathit{cube.cones}(G_0) \supsetneq \mathit{cube.cones}(G_1) \supsetneq \mathit{cube.cones}(G_2) \supsetneq \dots$. Since *cube.cones*($G_0$) is a finite set, this sequence must have finite length and therefore the algorithm terminates.

Next we show that $\mathit{alg.cone}_X(Z, P) = \mathbf{C}_X(F)$, where (Z, P) is the pair returned by the algorithm. Suppose that the while loop exits after N iterations. We show that $\mathit{alg.cone}_X(Z, P) = \mathbf{C}_X(F)$ that by proving each side of the equation is included in the other:

- $\subseteq$: Since the while loop exists after N iterations, we have that $G_N = F \land \neg B_1 \land \dots \land \neg B_N$ is unsatisfiable modulo **LRR**. Since $B_1 \models_{\mathbf{LRR}} B_2 \models_{\mathbf{LRR}} \dots \models_{\mathbf{LRR}} B_N$ (by Theorem 6), we have $F \models_{\mathbf{LRR}} B_N$, and therefore $\mathit{alg.cone}_X(Z, P) = \mathbf{C}_X(B_N) \subseteq \mathbf{C}_X(F)$.
- $\supseteq$: By Theorems 5 and 6, $\mathit{alg.cone}_X(Z, P) = \mathbb{Q}[X] \cap \bigcap_{i=1}^N C_i$ where each $C_i \in \mathit{cube.cone}(F)$, thus $\mathit{alg.cone}_X(Z, P) \supseteq \mathbb{Q}[X] \cup \bigcap_{C \in \mathit{cube.cone}(F)} C = \mathbf{C}_X(F)$. □

4 INTEGER ARITHMETIC

Let σ_{or}^Z be the signature of ordered rings extended with an additional unary relation symbol Int. Define the theory of *linear integer real rings* **LIRR** to be the σ_{or}^Z-theory axiomatized by the axioms of **LRR** along with the following:

$$\begin{array}{ll} \mathit{Int}(1) & \\ \forall x, y.\, \mathit{Int}(x) \land \mathit{Int}(y) \Rightarrow \mathit{Int}(x + y) & \text{(Int closure +)} \\ \forall x, y.\, \mathit{Int}(x) \land x + y = 0 \Rightarrow \mathit{Int}(y) & \text{(Int closure -)} \\ \text{for all } n \in \mathbb{Z}^{\geq 1} \text{ and } m \in \mathbb{Z}, \forall x.\, \mathit{Int}(x) \land 0 \leq nx + m \Rightarrow 0 \leq x + \left\lfloor \frac{m}{n} \right\rfloor & \text{(Cutting plane)} \end{array}$$

The cutting plane axiom is an axiom schema, with one axiom for each choice of positive integer n and integer m. Intuitively, Int identifies a set of elements as "integers", and an inequality $0 \leq nx + m$ can be strengthened to $0 \leq x + \left\lfloor \frac{m}{n} \right\rfloor$ whenever x is an "integer", as is the case for integers in $\mathbb{R}$. **LIRR** is thus the theory consisting of all σ_{or}^Z sentences that hold in all structures $\mathfrak{A}$ where $(U^{\mathfrak{A}}, 0^{\mathfrak{A}}, 1^{\mathfrak{A}}, +^{\mathfrak{A}}, \cdot^{\mathfrak{A}}, \leq^{\mathfrak{A}})$ is a model of **LRR**, $\mathit{Int}^{\mathfrak{A}}$ is an additive subgroup of $(U^{\mathfrak{A}}, 0^{\mathfrak{A}}, +^{\mathfrak{A}})$ that contains $1^{\mathfrak{A}}$ (identifying the "integers"), and "integers" behave like integers with regards to $\leq$. We regard $\mathbb{R}$ as the standard model of **LIRR**, with Int identifying the subset of integers; $\mathit{Th}^Z(\mathbb{R})$ refers to the theory consisting of all σ_{or}^Z-sentences satisfied by $\mathbb{R}$. We can extend σ_{or}^Z-structures to interpret $\mathit{Int}(p)$ for $p \in \mathbb{Q}[X]$ by translating $\mathit{Int}(p)$ to the formula $\exists x.\, \mathit{Int}(x) \land p = x$, where x is a fresh symbol and $p = x$ is translated into a σ_{or}-formula as described in Section 3. For example, the formula $\mathit{Int}(\frac{1}{2}x - y)$ is translated to $\exists z.\, \mathit{Int}(z) \land (1 + 1) \cdot z + (1 + 1) \cdot y = x$.

An induction axiom is conspicuously absent from **LIRR**. Nevertheless, the axiomatization is sufficient for positive linear formulas.

Theorem 8. Let F be a ground $\sigma_{or}^Z(X)$-formula that is free of negation and multiplication. Then F is satisfiable modulo **LIRR** iff F is satisfiable modulo $\mathit{Th}^Z(\mathbb{R})$.

Proof. The $\Leftarrow$ direction is trivial, since any model of $\mathit{Th}^Z(\mathbb{R})$ is a model of **LIRR**.

For the $\Rightarrow$ direction, we prove that if F is unsatisfiable modulo $\mathit{Th}^Z(\mathbb{R})$, then it is unsatisfiable modulo **LIRR**. We may assume without loss of generality that F takes the form $\bigwedge_{p \in P} 0 \leq p \land \bigwedge_{s \in S} \mathit{Int}(s)$. We may also assume that $S \subseteq X$: if $F = G \land \mathit{Int}(t)$ for some non-constant term t, then

F is equisatisfiable with $G \land Int(y) \land y \leq t \land t \leq y$ for a fresh constant y (modulo $Th^Z(\mathbb{R})$ and also modulo **LIRR**). Furthermore, we may assume that $S = X$ (i.e., all symbols are integers), since if some symbol is not constrained to be an integer, it can be projected by Fourier-Motzkin elimination, resulting in an equisatisfiable formula (again modulo both theories).

Suppose that F is unsatisfiable modulo $Th^Z(\mathbb{R})$—i.e., the polyhedron defined by $\bigwedge_{p \in P} 0 \leq p$ has no integer points. Then there is a cutting-plane proof of $0 \leq -1$ from $\bigwedge_{p \in P} 0 \leq p$ [Chvátal 1973; Schrijver 1980]. Since each inference step of a cutting-plane proof is valid modulo **LIRR**, F is unsatisfiable modulo **LIRR**. □

We now extend models of **LRR** to **LIRR**. Let X be a set of variables. For sets $C, L \subseteq \mathbb{Q}[X]$, say that C is **closed under cutting planes with respect to** L if for any $n, m \in \mathbb{Z}$, $n > 0$, $p \in L$, if $np + m \in C$, then $p + \left\lfloor \frac{m}{n} \right\rfloor \in C$. Define the **cutting plane closure of** C **with respect to** L, denoted $cp_L(C)$, to be the least cone that contains C and is closed under cutting planes with respect to L. If C is a regular cone, define $\mathfrak{M}(C, L)$ to be the extension of $\mathfrak{M}(C)$ that interprets Int as $\{p + \mathcal{U}(C) : p \in L\}$. Observe that when $\mathcal{U}(C) \subseteq L$, $\mathfrak{M}(C, L) \models Int(p)$ iff $p \in L$, for all $p \in \mathbb{Q}[X]$.

Say that the pair (C, L) is **regular** if C is a regular cone, L is an additive subgroup of $\mathbb{Q}[X]$ that contains 1 and $\mathcal{U}(C)$, and C is closed under cutting planes with respect to L. (L is an additive subgroup if $0 \in L$ and $x - y \in L$ whenever $x, y \in L$.) If (C_1, L_1) and (C_2, L_2) are regular, we consider (C_1, L_1) to be smaller than (C_2, L_2) if $C_1 \subseteq C_2$ and $L_1 \subseteq L_2$.

Let X be a set of symbols, and let $\mathfrak{A}$ be a σ_{or}^Z structure satisfying the axioms of **LIRR**. Define $\mathcal{L}(\mathfrak{A}) \triangleq \{p \in \mathbb{Q}[X] : \mathfrak{A} \models Int(p)\}$ to be the set that $\mathfrak{A}$ identifies as "integers". Naturally, $(C(\mathfrak{A}), \mathcal{L}(\mathfrak{A}))$ is regular. For the converse, we have the following.

Lemma 6. Let X be a set of variables and $C, L \subseteq \mathbb{Q}[X]$. If (C, L) is regular and C is consistent, $\mathfrak{M}(C, L)$ is a model of **LIRR**.

Analogously with regular and algebraic cones for **LRR**, regular (C, L) give us a "standard form" in which **LIRR** models can be represented, but they lack a finite representation and cannot be manipulated effectively—we will define *algebraic* (C, L) for this purpose.

We begin with *algebraic lattices*, which serve as a finite representation (certain) additive subgroups of $\mathbb{Q}[X]$. Call a set $L \subseteq \mathbb{Q}[X]$ an **algebraic lattice** if $L = I + L_0$ for an ideal I and a point lattice L_0. For sets $Z, B \subseteq \mathbb{Q}[X]$, define $alg.lattice_X(Z, B) = \mathbb{Q}[X]\langle Z \rangle + \mathbb{Z}\langle B \rangle$. If (Z, B) is oriented (with respect to $\preceq$) and B is linearly independent, say that (Z, B) is **independently oriented** (with respect with $\preceq$). Assuming Z is a Gröbner basis, an independently oriented representation of $alg.lattice(Z, B)$ can be computed by

$$indep.orient_Z(B) \triangleq int.basis(\{\mathbf{red}_Z(b) : b \in B\})$$

where *int.basis* denotes a function that computes a basis for a point lattice (this can be done in polytime using a Hermite Normal Form computation [Schrijver 1999, Ch. 4]).

Lemma 7 (Orientation)**.** Let X be a set of variables, $Z, B \subseteq \mathbb{Q}[X]$ be finite sets of polynomials such that Z is a Gröbner basis. Then $\left(Z, indep.orient_Z(B)\right)$ is independently oriented, and $alg.lattice(Z, B) = alg.lattice(Z, indep.orient_Z(B))$.

Note that membership in a point lattice can be checked in polytime (to check $t \in \mathbb{Z}\langle b_1, \ldots, b_n \rangle$, first solve $a_1 b_1 + \cdots + a_n b_n = t$ and then check whether each $a_1, \ldots, a_n$ in the solution is an integer). Thus, by the following lemma we have that checking membership in an algebraic lattice is decidable.

Lemma 8 (Membership)**.** Let X be a set of variables, and $Z, B \subseteq \mathbb{Q}[X]$ be finite sets of polynomials such that (Z, B) is independently oriented. For any polynomial $p \in \mathbb{Q}[X]$, we have $p \in alg.lattice(Z, B)$ iff $\mathbf{red}_Z(p) \in \mathbb{Z}\langle B \rangle$.

Lastly, we develop a notion of algebraic cone/lattice pairs, which will serve as effective σ_{or}^{Z} structures. For $C, L \subseteq \mathbb{Q}[X]$, say that (C, L) is **algebraic** if there exists finite sets $Z, P, B \subseteq \mathbb{Q}[X]$ such that $C = \mathit{alg.cone}_X(Z, P)$ and $L = \mathit{alg.lattice}_X(Z, B)$. We denote this by $(C, L) = \mathit{alg.cone.lattice}_X(Z, P, B)$, and drop the subscript X when the context is clear. Say that (Z, P, B) is **oriented** (with respect with $\preceq$) if (Z, P) is oriented and (Z, B) is independently oriented (both with respect with $\preceq$). If an algebraic (C, L) is also regular, it can be represented by an oriented triple (Z, P, B) such that $\mathbb{Q}^{\geq 0}\langle P\rangle$ is salient: first compute an oriented (Z, P) with $\mathbb{Q}^{\geq 0}\langle P\rangle$ salient such that $C = \mathit{alg.cone}(Z, P)$ (Theorem 3), and then B such that (Z, B) is oriented and $L = \mathit{alg.lattice}(Z, B)$ (Lemma 7). Under this condition, we have $\mathcal{U}(\mathit{alg.cone}(Z, P)) = \langle Z\rangle \subseteq \mathit{alg.lattice}(Z, B)$, and thus $\mathfrak{M}(\mathit{alg.cone.lattice}(Z, P, B)) \models \mathit{Int}(p)$ iff $p \in \mathit{alg.lattice}(Z, B)$. Since membership in an algebraic cone and lattice are both decidable, it follows that checking whether $\mathfrak{M}(\mathit{alg.cone.lattice}(Z, P, B)) \models F$ for a ground formula F is decidable, provided that (Z, P, B) is oriented and P is salient.

4.1 Satisfiability Modulo LIRR

This section presents a decision procedure for testing satisfiability of ground conjunctive $\sigma_{or}^{Z}(X)$-formulas modulo **LIRR**. Without loss of generality, a ground conjunctive formula F can be written in the form

$$F = \left(\bigwedge_{p \in P} 0 \leq p\right) \land \left(\bigwedge_{q \in Q} \neg(0 \leq q)\right) \land \left(\bigwedge_{r \in R} \neg(0 = r)\right) \land \left(\bigwedge_{s \in S} \mathit{Int}(s)\right) \land \left(\bigwedge_{t \in T} \neg \mathit{Int}(t)\right),$$

where P, Q, R, S, T are all finite sets of polynomials. In the following, we show that it is possible to compute a finite representation of the least regular (C, L) such that C contains P and L contains S (Theorem 10). Then we show that F is satisfiable modulo **LIRR** if and only if C is consistent and $\mathfrak{M}(C, L) \models F$, and moreover that $C = \mathbf{C}(F)$ (Theorem 11). Since checking consistency of C and $\mathfrak{M}(C, L) \models F$ is decidable, this yields a decision procedure for ground **LIRR**-formulas, as well as an algorithm for computing the non-negative cone of a ground formula.

4.1.1 Cutting Plane Closure. This section addresses the following problem: given finite sets $Z, P, B \subseteq \mathbb{Q}[X]$, compute Z', P' such that $\mathit{alg.cone}(Z', P')$ contains $\mathit{alg.cone}(Z, P)$ and is closed under cutting planes with respect to the algebraic lattice $\mathit{alg.lattice}(Z, B)$. This is the key operation needed to compute the least regular pair (C, L) containing a given pair, which will be addressed in Section 4.1.2. Our strategy for computing cutting plane closure is to reduce it to the problem of computing the integer hull of a polyhedron.

First, we show that the cutting plane closure of a cone $\mathit{alg.cone}(Z, P)$ with respect to the algebraic lattice $\mathit{alg.lattice}(Z, B)$ coincides with the cutting plane closure of $\mathit{alg.cone}(Z, P)$ with respect to the point lattice $\mathbb{Z}\langle B\rangle$:

Lemma 9. Let $C, L \subseteq \mathbb{Q}[X]$ be such that C is a cone. Then $cp_{\mathcal{U}(C)+L}(C) = cp_L(C)$.

Proof. Let $L' = \mathcal{U}(C) + L$. Since $L \subseteq L'$, $cp_{L'}(C) \supseteq cp_L(C)$. For $cp_{L'}(C) \subseteq cp_L(C)$, first observe that $cp_L(C)$ is a cone that contains C, so it suffices to show that it is closed under cutting planes with respect to L'.

Let $np + m \in cp_L(C)$, where $n, m \in \mathbb{Z}$, $n > 0$, and $p \in L'$. Then $p = z + v$ for some $z \in \mathcal{U}(C)$ and $v \in L$. Since $np + m \in cp_L(C)$ and $nz \in \mathcal{U}(C) \subseteq \mathcal{U}(cp_L(C))$, we have $(np + m) - nz \in cp_L(C)$ and so

$$nv + m = nv + m + nz - nz = n(z + v) + m - nz = (np + m) - nz \in cp_L(C)\,.$$

Since $cp_L(C)$ is closed under cutting planes with respect to L and $v \in L$, $v + \left\lfloor \frac{m}{n} \right\rfloor \in cp_L(C)$. Then $p + \left\lfloor \frac{m}{n} \right\rfloor = z + v + \left\lfloor \frac{m}{n} \right\rfloor \in cp_L(C)$. Thus, $cp_{L'}(C) \subseteq cp_L(C)$. □

Next, we show that computing the cutting plane closure of an algebraic cone with respect to a point lattice can be reduced to computing the cutting plane closure of the cone of inequalities defining a (finite-dimensional) polyhedron.

Lemma 10. Let $B = \{b_1, \dots, b_k\} \subseteq \mathbb{Q}[X]$, and let $C \subseteq \mathbb{Q}[X]$ be a cone. Let $Y = \{y_1, \dots, y_k\}$ be a set of symbols disjoint from X and define a linear map $f : \mathbb{Q}[Y]^1 \to \mathbb{Q}[X]$ by

$$f(a_0 + a_1 y_1 + \cdots + a_k y_k) = a_0 + a_1 b_1 + \cdots + a_k b_k.$$

Then $cp_{\mathbb{Z}\langle B\rangle}(C) = C + f(cp_{\mathbb{Z}\langle Y\rangle}(f^{-1}(C)))$.

Proof. ($\subseteq$) Let $D = f(cp_{\mathbb{Z}\langle Y\rangle}(f^{-1}(C)))$. Since $cp_{\mathbb{Z}\langle B\rangle}(C)$ is the *least* cone that contains C and is closed under cutting planes with respect to $\mathbb{Z}\langle B\rangle$, it is sufficient to prove the following:

(1) $C + D$ is a cone: it is the sum of two cones and is thus a cone.
(2) $C + D$ contains C: since C is non-empty we have $0 \in D$, so $C \subseteq C + D$.
(3) $C + D$ is closed under cutting planes with respect to $\mathbb{Z}\langle B\rangle$. Suppose $np + m \in C + D$, with $n, m \in \mathbb{Z}$, $n > 0$, and $p \in \mathbb{Z}\langle B\rangle$; we must show that $p + \lfloor \frac{m}{n} \rfloor \in C + D$.
Without loss of generality, we may suppose $np + m \in D$—the argument is as follows. Since $np + m \in C + D$, we have $np + m = g + h$ for some $g \in C$ and $h \in D$. Since $np + m \in \mathbb{Z}\langle B\rangle + \mathbb{Z} = f(\mathbb{Q}[Y]^1)$, we have $g = (np + m) - h \in f(\mathbb{Q}[Y]^1)$ since the image of f is a subspace of $\mathbb{Q}[X]$ containing both $(np + m)$ and h. So $g \in C \cap f(\mathbb{Q}[Y]^1)$, and thus $g \in f(f^{-1}(C))$. Since $cp_{\mathbb{Z}\langle Y\rangle}(\cdot)$ is extensive (i.e., $S \subseteq cp_{\mathbb{Z}\langle Y\rangle}(S)$ for all S) and f is linear, we have $np + m = g + h \in D$.
We now prove that $p + \lfloor \frac{m}{n} \rfloor \in C + D$. Since $p \in \mathbb{Z}\langle B\rangle$, there exists $p' \in \mathbb{Z}\langle Y\rangle$ such that $f(p') = p$. Then $f(np' + m) = np + m \in D$. Since f is a linear map, $np' + m + g \in cp_{\mathbb{Z}\langle Y\rangle}(f^{-1}(C))$ for some $g \in \ker(f) \triangleq f^{-1}(0)$. This is a subspace, so $-g \in \ker(f)$. Since $0 \in C$, $\ker(f) = f^{-1}(0) \subseteq f^{-1}(C) \subseteq cp_{\mathbb{Z}\langle Y\rangle}(f^{-1}(C))$. So $-g \in cp_{\mathbb{Z}\langle Y\rangle}(f^{-1}(C))$, and since $cp_{\mathbb{Z}\langle Y\rangle}(f^{-1}(C))$ is a cone, $np' + m = (np' + m + g) - g \in cp_{\mathbb{Z}\langle Y\rangle}(f^{-1}(C))$. Since $p' \in \mathbb{Z}\langle Y\rangle$, $p' + \lfloor \frac{m}{n} \rfloor \in cp_{\mathbb{Z}\langle Y\rangle}(f^{-1}(C))$. So $p + \lfloor \frac{m}{n} \rfloor = f(p' + \lfloor \frac{m}{n} \rfloor) \in C + D$.

($\supseteq$) First note that $C \subseteq cp_{\mathbb{Z}\langle B\rangle}(C)$, and since $cp_{\mathbb{Z}\langle B\rangle}(C)$ is closed under addition, it suffices to show that $f(cp_{\mathbb{Z}\langle Y\rangle}(f^{-1}(C))) \subseteq cp_{\mathbb{Z}\langle B\rangle}(C)$. In turn, it suffices to show $cp_{\mathbb{Z}\langle Y\rangle}(f^{-1}(C)) \subseteq f^{-1}(cp_{\mathbb{Z}\langle B\rangle}(C))$. Let $D = f^{-1}(cp_{\mathbb{Z}\langle B\rangle}(C))$. As before, it suffices to show that D is a cone, $f^{-1}(C) \subseteq D$ and D is closed under cutting planes with respect to $\mathbb{Z}\langle Y\rangle$.

It is easy to verify that D is a cone, and since $cp_{\mathbb{Z}\langle B\rangle}(\cdot)$ is extensive, $f^{-1}(C) \subseteq D$. Suppose that $np + m \in f^{-1}(cp_{\mathbb{Z}\langle B\rangle}(C))$, with $p \in \mathbb{Z}\langle Y\rangle$, $n, m \in \mathbb{Z}$, and $n > 0$; we must show that $p + \lfloor \frac{m}{n} \rfloor \in D$. By linearity of f, $nf(p) + m = f(np + m) \in cp_{\mathbb{Z}\langle B\rangle}(C)$. Since $f(p) \in \mathbb{Z}\langle B\rangle$, and $cp_{\mathbb{Z}\langle B\rangle}(C)$ is closed under cutting planes with respect to $\mathbb{Z}\langle B\rangle$, $f(p) + \lfloor \frac{m}{n} \rfloor \in cp_{\mathbb{Z}\langle B\rangle}(C)$. Thus, $f(p + \lfloor \frac{m}{n} \rfloor) \in cp_{\mathbb{Z}\langle B\rangle}(C)$, and $p + \lfloor \frac{m}{n} \rfloor \in D$. □

It remains to show that the reduction in Lemma 10 is effective. The only non-trivial operation involved in the reduction that we have not already seen is computing the inverse image of an algebraic cone $C \subseteq \mathbb{Q}[X]$ under a linear map with finite-dimensional domain $f : \mathbb{Q}[Y]^1 \to \mathbb{Q}[X]$. This can be accomplished by extending the linear map to a ring homomorphism $\hat{f} : \mathbb{Q}[Y] \to \mathbb{Q}[X]$, taking the inverse image of C under $\hat{f}$, and intersecting it with $\mathbb{Q}[Y]^1$.

For disjoint finite sets of variables X and Y, a ring homomorphism $f : \mathbb{Q}[Y] \to \mathbb{Q}[X]$, and finite $Z, P \subseteq \mathbb{Q}[X]$, define

$$\textit{inverse-hom}(Z, P, f, Y) \triangleq \textit{project}_Y(\{y - f(y) : y \in Y\} \cup Z, P).$$

Theorem 9 (Inverse image)**.** Let $Z, P \subseteq \mathbb{Q}[X]$ be finite sets, Y be a finite set of variables distinct from X, and $f : \mathbb{Q}[Y] \to \mathbb{Q}[X]$ be a ring homomorphism. Then

$$\mathit{alg.cone}_Y(\mathit{inverse\text{-}hom}(Z, P, f, Y)) = f^{-1}(\mathit{alg.cone}_X(Z, P)).$$

Note that $\mathbb{Q}[Y]^1$ is an algebraic (in fact, polyhedral) cone in $\mathbb{Q}[Y]$, so we can intersect an algebraic cone with $\mathbb{Q}[Y]^1$ using the procedure of Section 3.3.3. Precisely, let $Y \subseteq X$ be sets of variables. Define

$$\mathit{intersect\text{-}subspace}(Z, P, Y) = \mathit{second}(\mathit{intersect}(Z, P, \emptyset, P_{\mathbb{Q}[Y]^1})),$$

where $P_{\mathbb{Q}[Y]^1} = Y \cup -Y \cup \{1, -1\}$, and *second* returns the second component of the pair. Note that the first component is always $\emptyset$ (or equivalently, $\{0\}$) since it is a Gröbner basis for the ideal $\langle Z \rangle \cap \langle \emptyset \rangle = \{0\}$.

Let $\overline{\mathit{cut}}(P, Y)$ be a procedure that computes the cutting plane closure of the cone generated by $P \subseteq \mathbb{Q}[Y]^1$, with respect to $\mathbb{Z}\langle Y \rangle$. That is, $\mathbb{Q}^{\geq 0}\left\langle \overline{\mathit{cut}}(P, Y) \right\rangle = \mathit{cp}_{\mathbb{Z}\langle Y \rangle}(\mathbb{Q}^{\geq 0}\langle P \rangle)$. This may be done by e.g., the iterated Gomory-Chvátal closure. [2]

We now arrive at an effective implementation of the reduction in Lemma 10. Define an operation $\mathit{cut}(Z, P, B)$ by:

$$\mathit{cut}(Z, P, B) \triangleq \mathit{substitute}(\hat{f}, \overline{\mathit{cut}}(\mathit{intersect\text{-}subspace}(\mathit{inverse\text{-}hom}(Z, P, \hat{f}, Y), Y))),$$

where $Y = \{y_1, \ldots, y_n\}$ is a set of fresh variables corresponding to $B = \{v_1, \ldots, v_n\}$ generating a point lattice, $\hat{f} : \mathbb{Q}[Y] \to \mathbb{Q}[X]$ is the ring homomorphism defined by $\hat{f}(y_i) = v_i$, and $\mathit{substitute}(\hat{f}, P')$ applies $\hat{f}$ to each polynomial in P'. By linearity of $\hat{f}$, $\mathit{substitute}(\hat{f}, P')$ computes the image of the polyhedral cone generated by P'.

Lemma 11. Let $Z, P, B \subseteq \mathbb{Q}[X]$ be finite subsets. Then we have

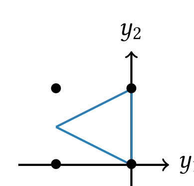
$$\mathit{cp}_{\mathbb{Z}\langle B \rangle}(\mathit{alg.cone}(Z, P)) = \mathit{alg.cone}(Z, P \cup \mathit{cut}(Z, P, B)).$$

Example 4.1. We illustrate $\mathit{cut}(Z, P, B)$ with $Z = \emptyset$, $P = \left\{x_1 - 2x_2 + 1, x_1 + 2x_2, -x_1, x_2^2\right\}$ and $B = \{2x_1, 2x_2\}$.

Let $f : \mathbb{Q}[y_1, y_2]^1 \to \mathbb{Q}[x_1, x_2]$ be the function mapping $f(y_1) = 2x_1$ and $f(y_2) = 2x_2$, and let $\hat{f}$ denote its extension to a ring homomorphism $\mathbb{Q}[y_1, y_2] \to \mathbb{Q}[x_1, x_2]$. First, we compute $\mathit{inverse\text{-}hom}(Z, P, \hat{f}, \{y_1, y_2\})$, yielding the pair $(\emptyset, P')$ with $P' = \left\{\frac{1}{2}y_1 - y_2 + 1, \frac{1}{2}y_1 + y_2, -\frac{1}{2}y_1, \frac{1}{4}y_2^2\right\}$ (representing the algebraic cone $\hat{f}^{-1}(\mathit{alg.cone}(Z, P))$ which, in this case, happens to be polyhedral). Next, we compute generators for the polyhedral cone $f^{-1}(\mathit{alg.cone}(Z, P)) = \hat{f}^{-1}(\mathit{alg.cone}(Z, P)) \cap \mathbb{Q}[y_1, y_2]^1$ via $\mathit{intersect\text{-}subspace}(\emptyset, P', \{y_1, y_2\})$, yielding $P'' = \left\{\frac{1}{2}y_1 - y_2 + 1, \frac{1}{2}y_1 + y_2, -\frac{1}{2}y_1\right\}$.

Next, we compute generators for $\mathit{cp}(\mathbb{Q}^{\geq 0}\langle P'' \rangle)$. Consider the polyhedron Q that is defined by P'':

$$Q = \left\{(y_1, y_2) \in \mathbb{Q}^2 : 0 \leq \frac{1}{2}y_1 - y_2 + 1, 0 \leq \frac{1}{2}y_1 + y_2, 0 \leq -\frac{1}{2}y_1\right\}$$



This polyhedron has vertices $(0, 0)$, $(0, 1)$ and $(-1, \frac{1}{2})$, so its integer hull is defined by $y_1 = 0$, $0 \leq y_2$, and $y_2 \leq 1$. Hence, the cutting plane closure is $\overline{\mathit{cut}}(P'') = \{y_1, -y_1, y_2, -y_2 + 1\}$.

[2]When $1 \in P$, this can also be done by any algorithm that computes integer hulls. Precisely, $\mathit{cp}_{\mathbb{Z}\langle Y \rangle}(\mathbb{Q}^{\geq 0}\langle P \rangle) = \mathit{cp}(\mathbb{Q}^{\geq 0}\langle P \rangle)$, where the latter is the cutting plane closure of the (cone of) valid inequalities for $Q = \left\{x \in \mathbb{Q}^Y : p(x) \geq 0 \text{ for all } p \in P\right\}$, which is also the cone of inequalities defining Q_I. The discrepancy arises because $1 \geq 0$ is always a valid inequality, and is contained in the latter; $\mathit{cp}_{\mathbb{Z}\langle Y \rangle}(\cdot)$ generalizes cutting plane closure to cones that may not contain 1.

Function $rcp(Z, P, B)$
  Input : Finite sets of polynomials Z, P, B
  Output: (Z', P', B') such that (Z', P', B') is oriented, $\mathbb{Q}^{\geq 0}\langle P'\rangle$ is salient, and $\mathit{alg.cone.lattice}(Z', P', B')$ is the least regular (C, L) satisfying $\mathit{alg.cone}(Z, P) \subseteq C$ and $\mathit{alg.lattice}(Z, B) \subseteq L$.
  $(Z', P') \leftarrow \mathit{regularize}(Z, P)$;
  $B' \leftarrow \mathit{indep.orient}_{Z'}(B \cup \{1\})$;
  repeat
    $(Z, P) \leftarrow (Z', P')$;
    $Q \leftarrow \mathit{cut}(Z', P', B')$;
    $(Z', P') \leftarrow \mathit{regularize}(Z', P' \cup Q)$;
    $B' \leftarrow \mathit{indep.orient}_{Z'}(B')$;
  until $\mathit{alg.cone}(Z', P') = \mathit{alg.cone}(Z, P)$;
  return (Z', P', B');

Algorithm 3: Regular cutting plane closure of a cone

Finally, applying *substitute* gives $\mathit{cut}(Z, P, B) = \{2x_1, -2x_1, 2x_2, -2x_2 + 1\}$. By Lemma 11,

$$\begin{aligned} cp_{\mathbb{Z}\langle B\rangle}(\mathit{alg.cone}(Z, P)) &= \mathit{alg.cone}(Z, P) + \mathbb{Q}^{\geq 0}\langle\{2x_1, -2x_1, 2x_2, -2x_2 + 1\}\rangle \\ &= \mathbb{Q}\langle\{x_1\}\rangle + \mathbb{Q}^{\geq 0}\left\langle x_2^2, x_2, -2x_2 + 1\right\rangle . \end{aligned}$$
⌟

4.1.2 Regular Cutting Plane Closure. This section addresses the following problem: given finite sets $Z, P, B \subseteq \mathbb{Q}[X]$, compute Z', P', B' such that $\mathit{alg.cone.lattice}(Z', P', B')$ is the least regular pair containing $\mathit{alg.cone.lattice}(Z, P, B)$ (component-wise). Regularity of (Z, P, B) requires that (1) $\mathit{alg.cone}(Z, P)$ is a regular cone and (2) $\mathit{alg.cone}(Z, P)$ is closed under cutting planes with respect to $\mathit{alg.lattice}(Z, P)$. The *regularize* and *rcp* procedures achieve (1) and (2) respectively, so the essential problem is to achieve both conditions simultaneously. Algorithm 3 accomplishes this by iterating *regularize* and *rcp* until a fixed point is reached.

Theorem 10 (Regular cutting plane closure)**.** Let X be a finite set of variables, and $Z, P, B \subseteq \mathbb{Q}[X]$ be finite sets. Then $rcp(Z, P, B)$ terminates, and $\mathit{alg.cone.lattice}(rcp(Z, P, B))$ is the least regular (C, L) satisfying $\mathit{alg.cone}(Z, P) \subseteq C$ and $\mathit{alg.lattice}(Z, B) \subseteq L$.

Proof. Assume that (Z, P, B) is oriented. Let Z_i, P_i, B_i, Q_i be Z', P', B', Q at the end of the ith iteration respectively. Define $C_i \triangleq \mathit{alg.cone}(Z_i, P_i)$ and $L_i \triangleq \mathit{alg.lattice}(Z_i, B_i)$. When $i = 0$, these values are the values just before entry to the loop. For any cone $C \subseteq \mathbb{Q}[X]$, let $R(C)$ denote the least regular cone containing C.

Define $D_{i+1} \triangleq cp_{\mathbb{Z}\langle B_i\rangle}(C_i)$. By Lemma 11, $D_{i+1} = \mathit{alg.cone}(Z_i, P_i \cup Q_i)$. By Theorem 3, $C_{i+1} = R(D_{i+1}) = R(cp_{\mathbb{Z}\langle B_i\rangle}(C_i))$. First note the following properties.

(P1) $\mathcal{U}(C_i) = \mathcal{U}(\mathit{alg.cone}(Z_i, P_i)) = \langle Z_i\rangle$, because the output (Z_i, P_i) of *regularize* is oriented and $\mathbb{Q}^{\geq 0}\langle P_i\rangle$ is salient (Theorem 3).
(P2) $(C_i)_i$ is an increasing sequence of regular cones by subset ordering, because $C_{i+1} = R(cp_{\mathbb{Z}\langle B_i\rangle}(C_i))$.
(P3) For all i, $1 \in C_i$, $\mathit{alg.cone}(Z, P) \subseteq C_i$, $1 \in L_i$, and $\mathit{alg.lattice}(Z, B) \subseteq L_i$: By Theorem 3 and (P2), $1 \in C_i$ and $\mathit{alg.cone}(Z, P) \subseteq C_i$ for all i. By Lemma 7, $1 \in L_0$ and $L \subseteq L_0$. By (P1) and (P2), $\langle Z_i\rangle = \mathcal{U}(C_i) \subseteq \mathcal{U}(C_{i+1}) = \langle Z_{i+1}\rangle$. Then $\mathit{alg.lattice}(Z_i, B_i) \subseteq \mathit{alg.lattice}(Z_{i+1}, B_i) = \mathit{alg.lattice}(Z_{i+1}, B_{i+1})$. Hence, $1 \in L_i$ and $\mathit{alg.lattice}(Z, B) \subseteq L_i$ for all i.

We now prove termination. By (P2), $(\mathcal{U}(C_i))_i$ is an ascending chain of ideals. Since $\mathbb{Q}[X]$ is Noetherian, $\mathcal{U}(C_n) = \mathcal{U}(C_{n+1})$ for some $n \geq 0$. Since $C_i \subseteq D_{i+1} \subseteq C_{i+1}$ for all i, $\mathcal{U}(C_n) \subseteq \mathcal{U}(D_{n+1}) \subseteq \mathcal{U}(C_{n+1}) =$

$\mathcal{U}(C_n)$, so $\mathcal{U}(D_{n+1}) = \mathcal{U}(C_n)$ is an ideal. Since $C_i \subseteq D_{i+1}$ and $1 \in C_i$ for all i (P3), $1 \in D_{n+1}$. So D_{n+1} is a regular cone. Since C_{n+1} is the least regular cone containing D_{n+1}, we have $C_{n+1} = D_{n+1}$.

We show that C_{n+1} is closed under cutting planes with respect to $\mathbb{Z}\langle B_{n+1}\rangle$, so that $C_{n+2} = D_{n+1} = C_{n+1}$ and the algorithm terminates. By (P1), $\langle Z_n\rangle = \mathcal{U}(C_n) = \mathcal{U}(C_{n+1}) = \langle Z_{n+1}\rangle$. Note that Z_n and Z_{n+1} are Gröbner bases with respect to the same monomial ordering, so by the representation independence property of Gröbner bases, $\mathbf{red}_{Z_n}(b) = \mathbf{red}_{Z_{n+1}}(b)$ for all $b \in \mathbb{Q}[X]$. Also, note that (Z_i, B_i) is independently oriented, by Lemma 7 and noting that the output of *regularize* is oriented. So $\mathbf{red}_{Z_i}(b) = b$ for all $b \in B_i$.

$$\begin{aligned}\mathbb{Z}\langle B_{n+1}\rangle &= \mathbb{Z}\left\langle \mathit{int.basis}\left(\left\{\mathbf{red}_{Z_{n+1}}(b) : b \in B_n\right\}\right)\right\rangle \\ &= \mathbb{Z}\left\langle \left\{\mathbf{red}_{Z_{n+1}}(b) : b \in B_n\right\}\right\rangle = \mathbb{Z}\left\langle \left\{\mathbf{red}_{Z_n}(b) : b \in B_n\right\}\right\rangle = \mathbb{Z}\langle B_n\rangle\end{aligned}$$

Hence,

$$cp_{\mathbb{Z}\langle B_{n+1}\rangle}(C_{n+1}) = cp_{\mathbb{Z}\langle B_n\rangle}(C_{n+1}) = cp_{\mathbb{Z}\langle B_n\rangle}(D_{n+1}) = cp_{\mathbb{Z}\langle B_n\rangle}(cp_{\mathbb{Z}\langle B_n\rangle}(C_n)) = cp_{\mathbb{Z}\langle B_n\rangle}(C_n) = C_{n+1},$$

where the last equality follows from the definition of D and noting $C_{n+1} = D_{n+1}$. Then $C_{n+2} = R(cp_{\mathbb{Z}\langle B_{n+1}\rangle}(C_{n+1})) = R(C_{n+1}) = C_{n+1}$, and the algorithm terminates.

Now let N be the last iteration of the algorithm. We show that $\mathit{alg.cone.lattice}(Z_N, P_N, B_N) = (C_N, L_N)$ is the least regular (C, L) satisfying $\mathit{alg.cone}(Z, P) \subseteq C$ and $\mathit{alg.lattice}(Z, B) \subseteq L$.

We first show that $\mathit{alg.cone.lattice}(Z_N, P_N, B_N)$ is regular. Since $N \geq 1$, $\mathit{alg.cone}(Z_N, P_N) = C_N = R(cp_{\mathbb{Z}\langle B_{N-1}\rangle}(C_{N-1}))$ is regular. By (P1), $\mathcal{U}(\mathit{alg.cone}(Z_N, P_N)) = \langle Z_N\rangle \subseteq \mathit{alg.lattice}(Z_N, B_N) = L_N$. By (P3), $1 \in L_N$. By the proof of termination, $cp_{\mathbb{Z}\langle B_N\rangle}(C_N) = C_N$. By (P1) and Lemma 9, C_N is closed under cutting planes with respect to L_N. Thus, $\mathit{alg.cone.lattice}(Z_N, P_N, B_N)$ is regular.

Finally, note that (1) $\mathit{alg.cone}(Z, P) \subseteq C_N$ and (2) $\mathit{alg.lattice}(Z, B) \subseteq L_N$ by (P3). We may show that (C_N, L_N) is the least regular pair satisfying (1) and (2) by supposing that (C', L') is regular and satisfies (1) and (2), and proving $C_i \subseteq C'$ and $L_i \subseteq L'$ by induction on i. □

4.1.3 Satisfiability and Consequence-Finding modulo **LIRR**. Since we have a procedure for computing the regular cutting plane closure of an algebraic cone with respect to a point lattice, the following theorem yields both a decision procedure for **LIRR** and an algorithm for consequence-finding modulo **LIRR**:

Theorem 11. Let F be the ground conjunctive formula

$$F = \left(\bigwedge_{p \in P} 0 \leq p\right) \wedge \left(\bigwedge_{q \in Q} \neg(0 \leq q)\right) \wedge \left(\bigwedge_{r \in R} \neg(0 = r)\right) \wedge \left(\bigwedge_{s \in S} \mathit{Int}(s)\right) \wedge \left(\bigwedge_{t \in T} \neg\mathit{Int}(t)\right).$$

Let (C, L) be the least regular pair such that $P \subseteq C$ and $S \subseteq L$. Then we have the following:

(1) F is satisfiable iff C is consistent and $\mathfrak{M}(C, L) \models F$.
(2) $C = \mathbf{C}(F)$.

Summarizing, we have the following decision procedure for satisfiability of conjunctive $\sigma^Z_{or}(X)$-formulas modulo **LIRR**. Let F be a ground conjunctive formula of the form in Theorem 11. First compute $(Z', P', B') = \mathit{rcp}(\emptyset, P, B)$. If $\mathbf{red}_{Z'}(1) = 0$, then F is unsatisfiable ($\mathit{alg.cone}(Z', P')$ is inconsistent). Otherwise, we check whether $\mathfrak{A} = \mathfrak{M}(\mathit{alg.cone.lattice}(Z', P', B'))$ satisfies F by testing whether there is some $q \in Q$ with $\mathbf{red}_{Z'}(q) \in \mathbb{Q}^{\geq 0}\langle P'\rangle$ (Lemma 2), or some $r \in R$ with $\mathbf{red}_{Z'}(r) = 0$, or some $t \in T$ with $\mathbf{red}_{Z'}(r) \in \mathbb{Z}\langle B'\rangle$ (Lemma 8); if such a q, r, or t exists, then F is unsatisfiable (Theorem 4), otherwise, $\mathfrak{A}$ satisfies F.

Consequence-finding modulo **LIRR** operates in the same way as Algorithm 2. The only difference is that *get-model* returns a triple (Z', P', B') instead of pair (Z', P')—but the point lattice basis B' is simply ignored by the rest of the algorithm.

5 INVARIANT GENERATION MODULO LIRR

This section describes an application of consequence-finding modulo **LIRR** to the problem of computing loop invariants. The method is based on extracting recurrence relations (of a particular form) from the body of a loop, and using their closed forms to summarize loop dynamics. The key property of our loop-invariant generation procedure is that it is *monotone*: if more information about the program's behavior is given to the procedure, then it may only compute invariants that are more precise (modulo **LIRR**). Monotonicity is achieved due to the fact that it is possible to find and manipulate the set of *all* implied inequalities of a formula modulo **LIRR**. Section 5.1 provides background on summarization and the loop summarization procedure presented in Section 5.2.

5.1 Program Summarization

Loop summarization is the problem of over-approximating the reflexive transitive closure of a transition formula (a notion that will be made precise in the following). One might think of a transition formula as a logical summary of the action of a program through one iteration of the body of some loop, and an approximation of its reflexive transitive closure as a summary for (an unbounded number of iterations of) the loop. Using the algebraic program analysis framework, a loop summarization procedure can be "lifted" to a whole-program analysis [Farzan and Kincaid 2015; Kincaid et al. 2021; Zhu and Kincaid 2021], allowing us to focus our attention on this relatively simple logical sub-problem.

Fix a finite set of symbols X (corresponding to the variables in a program of interest) and a set of "primed copies" $X' = \{x' : x \in X\}$. Define a **transition formula** F to be an existential $\sigma_{or}^{Z}(X \cup X')$-formula. A transition formula represents a relation over program states, where the unprimed variables correspond to the pre-state and the primed variables correspond to the post-state. For any pair of transition formulas $F[X, X']$ and $G[X, X']$, define their sequential composition as $F \circ G \triangleq \exists X''.F[X' \mapsto X''] \land G[X \mapsto X'']$ where $X'' \triangleq \{x'' : x \in X\}$ is a set of variable symbols disjoint from X and X' (representing the intermediate state between a computation of F and a computation of G). Define the t-fold composition of a transition formula F as $F^t \triangleq F \circ \ldots \circ F$ (t times). The problem of over-approximating reflexive transitive closure is, *given a transition formula F, find a transition formula $F^\star$ such that $F^t \models_{\mathbf{LIRR}} F^\star$ for all $t \in \mathbb{Z}^{\geq 0}$.*

Our approach to this problem is based on the one in [Ancourt et al. 2010; Farzan and Kincaid 2015]. Given a transition formula F, this approach computes an over-approximation of its reflexive transitive closure $F^\star$ in two steps. In the first step, it extracts a system of recurrences $F \models_{\mathbf{LRA}} \bigwedge_{i=1}^{n} r_i' \leq r_i + a_i$ where each r_i denotes a linear term over X, each r_i' denotes a corresponding linear term over X' and each a_i is a rational number (for instance $(x' + y') \leq (x + y) + 2$, indicating that the sum of x and y increases by at most two). In the second step, it computes their closed forms $F^\star \triangleq \exists t.t \geq 0 \land \bigwedge_{i=1}^{n} r_i' \leq r_i + ta_i$. In the following, we generalize this strategy to the non-linear setting by considering recurrences where a_i is not a rational number but rather an *invariant* of the loop—a polynomial in X that does not change under the action of F.

5.2 A Loop Summarization Operator

This section describes the class of recurrences that we wish to compute (Section 5.2.1), a method to compute them (Section 5.2.2), and then defines a loop summarization operator and shows that it is monotone and (in a sense) complete modulo **LIRR** (Section 5.2.3).

5.2.1 Invariant-Bounded Differences. Lift the mapping from symbols in X to their primed counterpart in X' into a ring homomorphism of polynomial terms $(-)' : \mathbb{Q}[X] \to \mathbb{Q}[X']$ (so, e.g., $(xy + 3z)'$ denotes the polynomial $(x'y' + 3z')$). Define the space of **linear invariant functionals** $LinInv(F)$

of a formula F to be those linear terms over X that are invariant under the action of F:

$$\mathit{LinInv}(F) \triangleq \{k \in \mathbb{Q}\langle X\rangle : F \models_{\mathbf{LIRR}} k' = k\} \ .$$

Define the space of **invariant polynomials** $\mathit{Inv}(F)$ to be the subring of $\mathbb{Q}[X]$ generated by $\mathit{LinInv}(F)$. Clearly, for any $p \in \mathit{Inv}(F)$ we have $F \models_{\mathbf{LIRR}} p' = p$ (but the reverse does not hold). Finally, we define the class of recurrences of interest, the **invariant-bounded differences** of F, to be

$$\mathit{diff}(F) \triangleq \{(r, a) \in \mathbb{Q}\langle X\rangle \times \mathit{Inv}(F) : F \models_{\mathbf{LIRR}} r' \leq r + a\} \ .$$

Note that for any $(r, a) \in \mathit{diff}(F)$ and any $t \in \mathbb{Z}^{\geq 0}$, we have $F^t \models_{\mathbf{LIRR}} r' \leq r + ta$.

Example 5.1. Consider the program

```
while(*) { if (*) { w = w + 1; } else { w = w + 2; }
           x = x + z ; y = y + z ; z = (x - y)(x - y); }
```

where $*$ denotes non-deterministic choice. The transition formula for the loop body is

$$F = (w' = w + 1 \lor w' = w + 2) \land x' = x + z \land y' = y + z \land z' = z + (x' - y')^2.$$

Notice that the linear polynomial $x - y$ is an invariant of the loop; $\mathit{LinInv}(F)$ is the linear space spanned by $(x - y)$. The polynomial $(x - y)^2$ is thus an invariant polynomial in the subring $\mathit{Inv}(F)$. The invariant-bounded differences $\mathit{diff}(F)$ includes (but is not limited to) $(-w, -1)$ (w increases by at least 1), $(w, 2)$ (w increases by at most 2), as well as $\left(z, (x - y)^2\right)$ and $\left(-z, -(x - y)^2\right)$ (z is always set to $z + (x - y)^2$ at each iteration). ⌟

Example 5.2. Consider the program **while** `(*) { w = -w; }`. The transition formula F of the loop body is $w' = -w$. The polynomial w^2 is an invariant polynomial of the loop: $F \models_{\mathbf{LIRR}} (w')^2 = w^2$. However, F has no *linear* invariants, so $\mathit{LinInv}(F) = \{0\}$ and $\mathit{Inv}(F) = \{0\}$. ⌟

5.2.2 Computing Invariant-Bounded Differences. We proceed in three steps, showing how to represent and compute $\mathit{LinInv}(F)$, $\mathit{Inv}(F)$, and finally $\mathit{diff}(F)$.

Since $\mathit{LinInv}(F)$ is a linear space it can be represented by a basis, which we compute as follows. Let $D \triangleq \{d_x : x \in X\}$ denote a set of variables distinct from those in X, X'. Define a ring homomorphism $\delta : \mathbb{Q}[D] \to \mathbb{Q}[X \cup X']$ by $\delta(d_x) \triangleq x - x'$ and a second ring homomorphism $\mathit{pre} : \mathbb{Q}[D] \to \mathbb{Q}[X]$ by $\mathit{pre}(d_x) = x$. Then $\mathit{LinInv}(F) = \mathit{pre}(\delta^{-1}(\mathcal{U}(\mathbf{C}\,(F))) \cap \mathbb{Q}\langle D\rangle)$: $F \models_{\mathbf{LIRR}} k' = k$ iff $F \models_{\mathbf{LIRR}} k' - k = 0$ iff $k' - k$ is a unit in the nonnegative cone of F; since k is linear, $k' - k = \delta(d)$ for some linear term $d \in \mathbb{Q}\langle D\rangle$. A basis for $\mathit{LinInv}(F)$ can be computed using the primitives we have developed in the preceding (consequence-finding, cone intersection, and inverse image).

Let $\{a_1, \ldots, a_n\}$ be a basis for $\mathit{LinInv}(F)$. The subring $\mathit{Inv}(F)$ of $\mathbb{Q}[X]$ can be represented as the elements of the polynomial ring $\mathbb{Q}[K]$, where $K = \{k_1, \ldots, k_n\}$ is a set of fresh variables, one for each basis element of $\mathit{LinInv}(F)$. Let inv denote the (injective, since $\{a_1, \ldots, a_n\}$ is linearly independent) ring homomorphism $\mathbb{Q}[K] \to \mathbb{Q}[X]$ that sends k_i to a_i for each i. Then the image of inv is precisely $\mathit{Inv}(F)$ (i.e., $\mathbb{Q}[K]$ and $\mathit{Inv}(F)$ are isomorphic).

Finally, we show how to represent and compute $\mathit{diff}(F)$. Each element of $\mathit{diff}(F)$ is a pair (r, a) consisting of a linear term $r \in \mathbb{Q}\langle X\rangle$ and a polynomial $a \in \mathit{Inv}(F)$. As a technical convenience, we can represent such a pair as a polynomial in $\mathbb{Q}\langle D\rangle + \mathbb{Q}[K]$. Since D and K are disjoint, there exist (unique) linear maps $\pi_D : \mathbb{Q}\langle D\rangle + \mathbb{Q}[K] \to \mathbb{Q}\langle D\rangle$ and $\pi_K : \mathbb{Q}\langle D\rangle + \mathbb{Q}[K] \to \mathbb{Q}[K]$ such that for all $p \in \mathbb{Q}\langle D\rangle + \mathbb{Q}[K]$ we have $p = \pi_D(p) + \pi_K(p)$. Then we may define a bijection $\mathit{rep} : \mathbb{Q}\langle D\rangle + \mathbb{Q}[K] \to \mathbb{Q}\langle X\rangle \times \mathit{Inv}(F)$ by $\mathit{rep}(p) = (\mathit{pre}(\pi_D(p)), \mathit{inv}(\pi_K(p)))$. Define $\widetilde{\mathit{diff}}(F)$ to be the inverse image of $\mathit{diff}(F)$ under rep (i.e., an exact representation of $\mathit{diff}(F)$ in the space $\mathbb{Q}\langle D\rangle + \mathbb{Q}[K]$).

Thus, it suffices to show how to compute $\widetilde{\mathit{diff}}(F)$. Define a ring homomorphism $\delta_{\mathit{inv}} : \mathbb{Q}[D, K] \to \mathbb{Q}[X \cup X']$ by $\delta_{\mathit{inv}}(d_x) = x - x'$ and $\delta_{\mathit{inv}}(k_i) = a_i$ (i.e., the common extension of δ and inv). Then we see that $\widetilde{\mathit{diff}}(F) \triangleq \delta_{\mathit{inv}}^{-1}(\mathbf{C}\,(F)) \cap (\mathbb{Q}\langle D\rangle + \mathbb{Q}[K])$; that is, the invariant-bounded differences of F

Function $lin(Z, P, D, K)$

  Input : Finite sets of polynomials $Z, P \subseteq \mathbb{Q}[D, K]$ over disjoint variables D and K

  Output : A pair (V, R) with $\mathbb{Q}[K]\langle V\rangle + \mathbb{Q}^{\geq 0}\langle R\rangle = alg.cone(Z, P) \cap (\mathbb{Q}[K] + \mathbb{Q}\langle D\rangle)$

  $(Z, P) \leftarrow orient_{\leq_K}(Z, P)$;

  if $\mathbf{red}_Z(1) = 0$ **then**

    return $(\{1\}, D \cup -D)$; /* $alg.cone(Z, P) = \mathbb{Q}[D, K]$, so return $\mathbb{Q}[K] + \mathbb{Q}\langle D\rangle$ */

  $R \leftarrow \overline{project}_{D \cup [K]}(P)$; /* Project onto the set of variables D and monomials over K */

  $V \leftarrow \emptyset$;

  /* Add polynomials in $Z \cap \mathbb{Q}[K]$ to V; if a polynomial is linear in d, add it (and its negation) to R */

  foreach $z \in Z$ **do**

    if $\textsc{Lm}(z) \in \mathbb{Q}[K]$ **then** $V \leftarrow V \cup \{z\}$;

    else if $\textsc{Lm}(z) \in \mathbb{Q}\langle D\rangle$ **then** $R \leftarrow R \cup \{z, -z\}$;

  return (V, R)

Algorithm 4: Linear restriction

correspond exactly to the members of $\delta_{inv}^{-1}(\mathbf{C}\,(F))$ that are linear in the D variables. We illustrate this with an example.

Example 5.3. We continue Example 5.1. The nonnegative cone of F is $\mathbf{C}\,(F) = alg.cone(Z, P)$, where Z and P are:

$$\begin{aligned} Z &= \left\{(w - w' + 1)(w - w' + 2), x - x' + z, y - y' + z, z - z' + (x - y)^2\right\} \\ P &= \{-w + w' - 1, w - w' + 2\} \end{aligned}$$

Applying δ^{-1} to the units of this cone gives $\mathbb{Q}[D]\left\langle (d_w + 1)(d_w + 2), d_x - d_y\right\rangle$ (to see why the inverse image contains $d_x - d_y$, observe that $\delta(d_x - d_y) = (x - x') - (y - y') = (x - x' + z) - (y - y' + z) \in \langle Z\rangle$). Intersecting this with $\mathbb{Q}\langle D\rangle$ yields $\mathbb{Q}\left\langle d_x - d_y\right\rangle$. Hence, $LinInv(F) = \mathbb{Q}\langle x - y\rangle$.

The subring $Inv(F)$ generated by these linear invariants is represented by $\mathbb{Q}[k_1]$ (with $inv(k_1) = x - y$). We have the following corresponding elements in $diff(F)$ and $\widetilde{diff}(F)$:

$$(-w, -1) \sim -d_w - 1,\ (w, 2) \sim d_w + 2,\ \left(z, (x - y)^2\right) \sim d_z + k_1^2, \text{ and } \left(-z, -(x - y)^2\right) \sim -d_z - k_1^2$$

Notice how $-d_w - 1$ and $d_w + 2$ correspond to P, and how $d_z + k_1^2$ and $-d_z - k_1^2$ correspond to the last polynomial $(z - z') + (x - y)^2 \in Z$. ⌟

We now continue with showing how to compute $\widetilde{diff}(F)$. Since we already saw how to compute inverse images of algebraic cones, it remains only to show that we can compute the intersection of an algebraic cone over $\mathbb{Q}[D, K]$ with $\mathbb{Q}\langle D\rangle + \mathbb{Q}[K]$; i.e., the set of polynomials in cone that are linear in the set of variables D, but may contain arbitrary monomials in K. This is *nearly* solved by the intersection algorithm in Section 3.3.3, since $\mathbb{Q}\langle D\rangle + \mathbb{Q}[K]$ is the sum of a finitely-generated cone $\mathbb{Q}\langle D\rangle = \mathbb{Q}^{\geq 0}\langle D \cup -D\rangle$ and an ideal $\mathbb{Q}[K]$; however, $\mathbb{Q}\langle D\rangle + \mathbb{Q}[K]$ is not an algebraic cone because $\mathbb{Q}[K]$ is not an ideal *in* $\mathbb{Q}[D, K]$. However, the essential process behind cone intersection carries over, which yields Algorithm 4.

Lemma 12. Let D, K be disjoint finite sets of variables and $Z, P \subseteq \mathbb{Q}[D, K]$ be finite sets of polynomials over D and K. Let $(V, R) = lin(Z, P, D, K)$. Then

$$\mathbb{Q}[K]\langle V\rangle + \mathbb{Q}^{\geq 0}\langle R\rangle = alg.cone(Z, P) \cap (\mathbb{Q}[K] + \mathbb{Q}\langle D\rangle)$$

5.2.3 Loop Summarization. The core logic of our loop summarization operator appears in Algorithm 5, which computes an over-approximation $exp(F, t)$ of the t-fold composition of F (symbolic

Function $exp(F, t)$
  Input : Transition formula F, symbol t
  Output: Over-approximation of the t-fold composition of F
  $F \leftarrow$ Skolemize F;
  $\{a_1, \ldots, a_n\} \leftarrow$ basis for $LinInv(F)$;
  $(Z, P) \leftarrow consequence(F, X \cup X')$;
  $D = \{d_x : x \in X\} \leftarrow$ set of $|X|$ fresh variables;
  $K = \{k_1, \ldots, k_n\} \leftarrow$ set of n fresh variables;
  Let $\delta_{inv} : \mathbb{Q}[K \cup D] \to \mathbb{Q}[X \cup X']$ be the homomorphism mapping $d_x \mapsto x$ and $k_i \mapsto a_i$;
  $(Z', P') \leftarrow inverse\text{-}hom(Z, P, \delta_{inv}, D \cup K)$;
  $(V, R) \leftarrow lin(Z', P', D, K)$;
  Define cf to be the map that sends $p \mapsto \delta(\pi_D(p)) + t \cdot inv(\pi_K(p))$;
  return $(\bigwedge_{v \in V} 0 = cf(v)) \land (\bigwedge_{r \in R} 0 \leq cf(r))$

Algorithm 5: Over-approximation of t-fold composition of F

in t) using the methodology described in the last two sections. The loop summarization operator is defined as:

$$F^\star \triangleq \exists t. Int(t) \land t \geq 0 \land exp(F, t) .$$

Example 5.4. Returning to Example 5.3, the invariant-bounded differences computed by lin are $V = \emptyset$ and $R = \{-d_w - 1, d_w + 2, \pm(d_z + k_1^2), \pm(d_x - d_y)\}$. Hence, Algorithm 5 computes

$$exp(F, t) = -(w - w') - t \geq 0 \land w - w' + 2t \geq 0 \land z - z' + t(x - y)^2 = 0 \land (x - x') - (y - y') = 0.$$

That is, the loop may be summarized as having the effect

$$w + t \leq w' \leq w + 2t \land z' = z + t(x - y)^2 \land x - y = x' - y' .$$

Thus, our algorithm computes exact dynamics for w and z, but loses information about the behavior of x and y. ⌟

Theorem 12 (Soundness). Let F be a transition formula. Then $F^t \models_{\mathbf{LIRR}} F^\star$ for all $t \in \mathbb{Z}^{\geq 0}$.

We conclude with a statement of the completeness and monotonicity properties of Algorithm 5.

Lemma 13 (Completeness). Given any transition formula F, we have $exp(F, t) \models_{\mathbf{LIRR}} r' \leq r + ta$ for all $(r, a) \in diff(F)$.

Theorem 13 (Monotonicity). If $F \models_{\mathbf{LIRR}} G$, then $F^\star \models_{\mathbf{LIRR}} G^\star$.

6 EXPERIMENTAL EVALUATION

We consider two experimental questions raised by the relatively weak strength of **LRR**/**LIRR**.

(1) (Section 6.2): Are the theories **LRR**/**LIRR** strong enough to prove unsatisfiability of formulas that are unsatisfiable modulo $Th(\mathbb{R})/Th(\mathbb{Z})$? How does the performance of the decision procedures for **LRR**/**LIRR** compare with state-of-the-art SMT solvers?
(2) (Section 6.3): Is the theory **LIRR** strong enough to enable client applications that rely on consequence-finding, such as the invariant generation algorithm presented in Section 5? How does **LIRR**-supported invariant generation compare with state-of-the-art automated program verification tools?

6.1 Experimental Setup

Implementation. We implemented an SMT solver for **LRR** and **LIRR**, which we call Chilon[3]. Our implementation relies on Z3 (as a SAT solver) [de Moura and Bjørner 2008] Apron/NewPolka [Jeannet and Miné 2009] for polyhedral operations, FLINT [Hart et al. 2021] for lattice operations, and Normaliz [Bruns et al. 2021] for Hilbert basis computation (a sub-procedure of iterated Gomory-Chvátal closure). Based on Chilon, we implemented the consequence-finding procedure as well as the approximate transitive closure operator for invariant generation. We have also integrated the invariant generation procedure into a static analysis framework that facilitates evaluation on software verification benchmarks, which is used as a proxy to evaluate the quality of the generated invariants.

Environment. We ran all experiments on an Oracle VirtualBox virtual machine with Lubuntu 22.04 LTS (Linux kernel version 5.15 LTS), with a two-core Intel Core i5-5575R CPU @ 2.80 GHz and 4 GB of RAM. All tools were run with the benchexec [Wendler and Beyer 2022] tool under a time limit of 2 minutes on all benchmarks. Reported times are total aggregate, measured in seconds and averaged across 5 runs.

SMT Benchmarks and Solvers. SMT tasks are taken from the quantifier-free nonlinear real arithmetic and nonlinear integer arithmetic (QF_NRA and QF_NIA) divisions of SMT-COMP 2021 [Barbosa et al. 2022] For each division, we randomly draw the same number of tasks from each directory, resulting in about 100 tasks. We compare the performance of Chilon on these tasks against other solvers for standard theories of arithmetic, including Z3 4.8.13, MathSAT 5.6.5, CVC4 1.8, and Yices 2.6.4.

Program Verification Benchmarks and Tools. The program verification tasks are the safe, integer-only [4] benchmarks from the c/ReachSafety-Loops subcategory of SV-COMP 2022 [Beyer 2022]. Tasks from the nla-digbench and nla-digbench-scaling directories constitute the nonlinear benchmark suite, while all other tasks form the linear suite. We compare against CRA (or Compositional Recurrence Analysis [Farzan and Kincaid 2015]), another invariant generation tool based on analyzing recurrence relations; VeriAbs 1.4.2, the winner for the ReachSafety category of SV-COMP; and Ultimate Automizer 0.2.2, which performed best on the nla-digbench suite.

6.2 How Does Chilon Perform on SMT Tasks?

Table 2 records the results of running the five solvers on nonlinear SMT benchmarks. Since the reals are a model of **LRR**, if a formula is unsatisfiable modulo **LRR** then it is unsatisfiable modulo **NRA**, but the converse does not hold i.e., using a **LRR**-solver on **NRA** tasks can give false SAT results, but not false UNSAT results). The same holds for **LIRR** and **NIA**.

Chilon does not appear to be competitive for either QF_NRA or QF_NIA. It proves UNSAT for 6 out of 40 and 14 out of 41 tasks in the QF_NRA and QF_NIA suites, respectively, lower than other SMT solvers we compared against. Chilon reports false positives on 23 out of 40 *UNSAT* tasks in QF_NRA and 15 out of 41 *UNSAT* tasks in QF_NIA. It performs poorly on *crafted* tasks, in which the unsatisfiability proof often requires reasoning about the interaction between multiplication and the order relation, which is not axiomatized by our theories. It performs particularly well on verification tasks (e.g., the hycomp suite in QF_NRA, and the LassoRanker and UltimateLassoRanker suites in QF_NIA), but all other solvers we tested also performed well on these tasks. Chilon is competitive with other solvers in terms of running time. We note that there is substantial room for improving

[3]The famous ancient Greek proverb "less is more", is attributed to Chilon of Sparta, one of the Seven Sages of Greece.
[4]That is, the error location is unreachable, and all variables are integer-typed.

the performance of Chilon; in particular, it does not tightly integrate the theory solver with the underlying SAT solver as do most modern SMT solvers.

The experimental results suggest that the theories of linear integer / real rings is not generically suitable as an alternative to the theory of integers / reals for applications that only require a SAT or UNSAT answer.

Table 2. Comparison of Chilon with other SMT solvers on SMT-COMP benchmarks. The "#P" column denotes the number of proved tasks, and the "#E" column denotes the number of tasks on which a solver times out / runs out of memory. The max standard deviation in runtime across all benchmarks is Chilon: 25.58, Z3: 18.71, MathSat: 2.03, CVC4: 6.10, Yices: 4.29.

			Chilon			Z3			MathSAT			CVC4			Yices		
suite	label	#task	#P	#E	time	#P	#E	time	#P	#E	time	#P	#E	time	#P	#E	time
QF_NRA	SAT	38	-	10/0	1215.9	36	2/0	380.6	15	21/0	2603.4	9	28/0	3405.7	29	9/0	1105.3
	UNSAT	40	6	11/0	1368.9	30	9/1	1193.4	29	10/0	1296.6	31	9/0	1194.1	32	8/0	1070.4
	?	27	-	14/2	1789.1	-	12/0	1495.3	-	18/0	2224.6	-	12/0	1827.8	-	13/0	1643.7
QF_NIA	SAT	29	-	13/0	1595.9	26	3/0	535.9	21	8/0	1065.9	17	7/0	905.2	19	10/0	1211.5
	UNSAT	41	14	11/0	1347.2	36	5/0	662.2	37	4/0	593.0	36	1/0	248.9	36	5/0	617.1
	?	28	-	16/1	2241.1	-	11/0	1361.1	-	12/0	1548.6	-	12/0	1467.1	-	12/0	1632.8

6.3 How Does Chilon-inv Perform on Program Verification Tasks?

In this subsection, we evaluate the strength of consequence-finding modulo **LIRR** via the invariant generation scheme described in Section 5, implemented in the tool Chilon-inv. We compare with two similar recurrence-based invariant generation techniques, both of which rely on consequence-finding: CRA-lin [Farzan and Kincaid 2015] uses a complete consequence-finding procedure modulo linear arithmetic; and CRA [Kincaid et al. 2017] uses a heuristic consequence-finding procedure modulo non-linear arithmetic. We also evaluate a second configuration of Chilon-inv, which uses a refinement algorithm from [Cyphert et al. 2019] to improve analysis precision; this is guaranteed because the analysis that Chilon-inv implements is monotone.

Table 3 compares the performance of invariant generation using Chilon with other methods that utilize consequence-finding. Results show that Chilon-inv indeed performs strictly better than CRA-lin on both suites (particularly on the `nonlinear` suite). This is expected, since Chilon-inv's consequence-finding algorithm is more powerful than that of CRA-lin, and Chilon-inv considers a larger class of recurrences. Chilon-inv outperforms CRA on the `nonlinear` suite, but CRA dominates the `linear` suite. This can be attributed to CRA's control-flow refinement techniques that are not implemented in the base analysis in Chilon-inv (with refinement, Chilon-inv matches the performance of CRA on the `linear` suite—see Table 4).

Table 3. Comparison of recurrence-based invariant generation schemes. Timeouts are reported in parentheses. The max standard deviation in runtime across all benchmarks is Chilon: 1.89, CRA-lin: 0.08, CRA: 0.63.

		Chilon-inv		CRA-lin		CRA	
	#tasks	#correct	time	#correct	time	#correct	time
`linear`	178	117	955.9 (6)	111	**739.0** (2)	**140**	1195.3 (5)
`nonlinear`	290	**232**	**2259.5** (15)	27	2617.8 (19)	90	12539.4 (91)

Table 4. Comparison of Chilon (with and without refinement) against leading SV-COMP competitors. Timeouts are reported in parentheses. The max standard deviation in runtime across all benchmarks is Chilon: 1.89, Chilon-inv + Refine: 0.11, UAutomizer: 9.08, VeriAbs: 0.00.

		Chilon-inv		Chilon-inv + Refine		UAutomizer		VeriAbs	
	#tasks	#correct	time	#correct	time	#correct	time	#correct	time
`linear`	178	117	**955.9** (6)	140	1289.0 (8)	122	8865.4 (56)	**143**	6520.3 (34)
`nonlinear`	290	232	**2259.5** (15)	**233**	2355.0 (16)	183	16569.2 (107)	172	17342.5 (118)

Table 4 provides context by comparing with state-of-the-art program verification tools. VeriAbs is a portfolio verifier that employs a variety of techniques, such as bounded model checking, k-induction, and loop summarization [Darke et al. 2021]. Notably, VeriAbs can summarize a numeric loops by accelerating linear recurrences involving only constants or variables that are unmodified within the loop [Darke et al. 2015]; the technique presented in Section 5.1 generalizes this scheme. Ultimate Automizer implements a counterexample-guided abstraction refinement algorithm following the trace abstraction paradigm [Heizmann et al. 2009]. Chilon-inv is competitive on the `linear` suite (particularly with refinement enabled) and outperforms the other tools on the `nonlinear` suite.

7 RELATED WORK

Real Arithmetic. The decidability of the theory of real closed fields is a classical result due to Tarski [1949] and Seidenberg [1954]. Practical (complete) algorithms for this theory are based on cylindrical algebraic decomposition (CAD) [Collins 1975; Jovanović and de Moura 2013; Kremer and Ábrahám 2020]. Due to the high computational complexity of decision procedures for the reals, a number of (incomplete) heuristic techniques have been devised [Tiwari 2005; Tiwari and Lincoln 2014; Zankl and Middeldorp 2010]. Similar to our approach, Tiwari [2005]'s method combines techniques from Gröbner bases and linear programming; however, the result of the combination is a semi-decision procedure for the existential theory of the reals, whereas we achieve a decision procedure for a weaker theory.

The δ*-complete* decision procedure for the existential theory of the reals presented in [Gao et al. 2012] is similar in spirit to our work, in that the method gives up completeness in the classical sense while retaining a weaker version of it. Rather than using a standard model of the reals and relaxing the definition of satisfaction (each constraint is "within δ" of being satisfied), we take the approach of using classical first-order logic, but admit non-standard models.

Positivestellensätze are a class of theorems that characterize sets of positive polynomial consequences of a system of inequalities over the reals (or a real closed field) [Krivine 1964]. Lemma 4 might be thought of as an analogue of a Positivestellensatz for **LRR**. Putinar's Positivestellensatz [Putinar 1993] bears particular resemblance to our results: it asserts that the entailment $\bigwedge_{z \in Z} z = 0 \land \bigwedge_{p \in P} p \geq 0 \models_{\mathbf{NRA}} q \geq 0$ holds exactly when q is in the *quadratic module* generated by Z and P (provided that certain technical restrictions implying compactness are satisfied). The quadratic module generated by Z and P is $\langle Z \rangle + \Sigma^2[X]\langle P \cup \{1\}\rangle$ (where $\Sigma^2[X]$ is the set of sum-of-squares polynomials over X), mirroring the structure of algebraic cones, but with sum-of-squares polynomials in place of non-negative rationals. Every quadratic module over $\mathbb{R}[X]$ is also regular algebraic cone (following from the fact that its additive units form an ideal [Marshall 2008, Prop 2.1.2]) (but not vice versa).

Integer Arithmetic. Unlike the case of the reals, the theory consisting of σ_{or}-sentences that hold over the integers is undecidable (in fact, not even recursively axiomatizable). However, a number of effective heuristics have been proposed, including combining real relaxation with branch-and-bound [Jovanović 2017; Kremer et al. 2016], bit-blasting [Fuhs et al. 2007], and linearization [Borralleras et al. 2019, 2009]. Linearization shares the idea of using linear arithmetic to reason about non-linear formulas. However, our approaches are quite different: Borralleras et al. [2019, 2009] essentially restrict the domain of constant symbols to finite ranges (making the approach sound but incomplete for satisfiability), whereas our approach is sound but incomplete for validity.

Non-Linear Invariant Generation. There are several abstract domains that are capable of representing conjunction of polynomial inequalities [Bagnara et al. 2005; Colón 2004; Gulavani and Gulwani 2008; Kincaid et al. 2017]. Such domains incorporate "best effort" techniques for reasoning about non-linear arithmetic. Notably, Kincaid et al. [2017] and Bagnara et al. [2005] combine techniques from commutative algebra and polyhedral theory. Our approach differs in that we designed *complete* inference, albeit modulo a weak theory of arithmetic.

There is a line of work on *complete* algorithms for finding invariant polynomial equations of (restricted) loops [Hrushovski et al. 2018; Humenberger et al. 2018; Kovács 2008; Rodríguez-Carbonell and Kapur 2004]. Another approach is to reduce polynomial invariant generation to linear invariant generation by introducing new dimensions, which provides completeness up to a degree-bound [de Oliveira et al. 2016; Müller-Olm and Seidl 2004]. Chatterjee et al. [2020] obtains a completeness (up to technical parameters) result for template-based generation of invariant polynomial inequalities, based on a "bounded" version of Putinar's Positivestellensatz. Lemma 13 is a kind of completeness result, which is relative to a class of recurrences, rather than an invariant "shape."

Consequence-Finding. A key feature of the arithmetic theories introduced in this paper is that they enable complete methods for finding and manipulating the set of consequences of a formula (of a particular form). In the setting of abstract interpretation, this problem is known as *symbolic abstraction* [Reps et al. 2004; Thakur 2014], and is phrased as the problem of computing the (ideally, best) approximation of a formula within some abstract domain. Symbolic abstraction algorithms are known for predicate abstraction [Graf and Saidi 1997], equations [Berdine and Bjørner 2014], affine equations [Reps et al. 2004; Thakur et al. 2015], template constraint domains (intervals, octagons, etc) [Li et al. 2014], and convex polyhedra [Farzan and Kincaid 2015]. Kincaid et al. [2017] gives a symbolic abstraction procedure for the wedge domain (which can express polynomial inequalities); this procedure is "best effort," whereas our consequence-finding algorithm offers completeness guarantees.

It is interesting to note that our lazy consequence-finding algorithm (Algorithm 2) can be seen as an instantiation of Reps et al. [2004]'s symbolic abstraction algorithm. The termination argument of this algorithm relies on the abstract domain satisfying the ascending chain condition, which algebraic cones do *not*; instead, we exploit the fact that we can compute *minimal* models of **LRR** / **LIRR**, and each formula has only finitely many.

DATA AVAILABILITY STATEMENT

The artifact for this work has passed the review by the POPL Artifact Evaluation Committee and is available online [Kincaid et al. 2022a].

ACKNOWLEDGMENTS

This work was supported in part by the NSF under grant number 1942537 and by ONR under grant N00014-19-1-2318. Opinions, findings, conclusions, or recommendations expressed herein are those of the authors and do not necessarily reflect the views of the sponsoring agencies.

REFERENCES

Corinne Ancourt, Fabien Coelho, and François Irigoin. 2010. A Modular Static Analysis Approach to Affine Loop Invariants Detection. *Electron. Notes Theor. Comput. Sci.* 267, 1 (October 2010), 3–16. https://doi.org/10.1016/j.entcs.2010.09.002

Roberto Bagnara, Enric Rodríguez-Carbonell, and Enea Zaffanella. 2005. Generation of Basic Semi-Algebraic Invariants Using Convex Polyhedra. In *Proceedings of the 12th International Conference on Static Analysis* (London, UK) *(SAS'05)*. Springer-Verlag, Berlin, Heidelberg, 19–34. https://doi.org/10.1007/11547662_4

Haniel Barbosa, François Bobot, and Jochen Hoenicke. 2022. SMT-COMP 2021. https://smt-comp.github.io/2021/benchmarks.html.

Florence Benoy, Andy King, and Fred Mesnard. 2005. Computing Convex Hulls with a Linear Solver. *Theory Pract. Log. Program.* 5, 1–2 (Jan 2005), 259–271. https://doi.org/10.1017/S1471068404002261

Josh Berdine and Nikolaj Bjørner. 2014. Computing All Implied Equalities via SMT-Based Partition Refinement. In *Automated Reasoning*, Stéphane Demri, Deepak Kapur, and Christoph Weidenbach (Eds.). Springer International Publishing, Cham, 168–183. https://doi.org/10.1007/978-3-319-08587-6_12

Dirk Beyer. 2022. Progress on Software Verification: SV-COMP 2022. In *Tools and Algorithms for the Construction and Analysis of Systems*, Dana Fisman and Grigore Rosu (Eds.). Springer International Publishing, Cham, 375–402. https://doi.org/10.1007/978-3-030-99527-0_20

Nikolaj Bjørner, Anh-Dung Phan, and Lars Fleckenstein. 2015. νZ - An Optimizing SMT Solver. In *Tools and Algorithms for the Construction and Analysis of Systems*, Christel Baier and Cesare Tinelli (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 194–199. https://doi.org/10.1007/978-3-662-46681-0_14

Cristina Borralleras, Daniel Larraz, Enric Rodríguez-Carbonell, Albert Oliveras, and Albert Rubio. 2019. Incomplete SMT Techniques for Solving Non-Linear Formulas over the Integers. *ACM Trans. Comput. Logic* 20, 4, Article 25 (aug 2019), 36 pages. https://doi.org/10.1145/3340923

Cristina Borralleras, Salvador Lucas, Rafael Navarro-Marset, Enric Rodríguez-Carbonell, and Albert Rubio. 2009. Solving Non-linear Polynomial Arithmetic via SAT Modulo Linear Arithmetic. In *Automated Deduction – CADE-22*, Renate A. Schmidt (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 294–305. https://doi.org/10.1007/978-3-642-02959-2_23

Winfried Bruns, Bogdan Ichim, Christof Söger, and Ulrich von der Ohe. 2021. Normaliz Algorithms for rational cones and affine monoids. Version 3.9.1, https://www.normaliz.uni-osnabrueck.de. https://doi.org/10.1016/j.jalgebra.2010.01.031

Bruno Buchberger. 1976. A Theoretical Basis for the Reduction of Polynomials to Canonical Forms. *SIGSAM Bull.* 10, 3 (August 1976), 19–29. https://doi.org/10.1145/1088216.1088219

Krishnendu Chatterjee, Hongfei Fu, Amir Kafshdar Goharshady, and Ehsan Kafshdar Goharshady. 2020. Polynomial Invariant Generation for Non-Deterministic Recursive Programs *(PLDI 2020)*. https://doi.org/10.1145/3385412.3385969

Václav Chvátal. 1973. Edmonds Polytopes and a Hierarchy of Combinatorial Problems. *Discrete Math.* 4, 4 (apr 1973), 305–337. https://doi.org/10.1016/0012-365X(73)90167-2

George E. Collins. 1975. 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. https://doi.org/10.1007/3-540-07407-4_17

Michael Colón. 2004. Approximating the Algebraic Relational Semantics of Imperative Programs. In *Static Analysis, 11th International Symposium, SAS 2004, Verona, Italy, August 26-28, 2004, Proceedings (Lecture Notes in Computer Science, Vol. 3148)*, Roberto Giacobazzi (Ed.). Springer, 296–311. https://doi.org/10.1007/978-3-540-27864-1_22

Michael A. Colón, Sriram Sankaranarayanan, and Henny B. Sipma. 2003. Linear Invariant Generation Using Non-linear Constraint Solving. In *Computer Aided Verification*, Warren A. Hunt and Fabio Somenzi (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 420–432. https://doi.org/10.1007/978-3-540-45069-6_39

Patrick Cousot and Nicolas Halbwachs. 1978. Automatic Discovery of Linear Restraints among Variables of a Program. In *Proceedings of the 5th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages* (Tucson, Arizona) *(POPL '78)*. Association for Computing Machinery, New York, NY, USA, 84–96. https://doi.org/10.1145/512760.512770

David A. Cox, John Little, and Donal O'Shea. 2015. *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra* (4th ed.). Springer Publishing Company, Incorporated. https://doi.org/10.1007/978-3-319-16721-3

John Cyphert, Jason Breck, Zachary Kincaid, and Thomas Reps. 2019. Refinement of Path Expressions for Static Analysis. *Proc. ACM Program. Lang.* 3, POPL, Article 45 (Jan 2019), 29 pages. https://doi.org/10.1145/3290358

Priyanka Darke, Sakshi Agrawal, and R. Venkatesh. 2021. VeriAbs: A Tool for Scalable Verification by Abstraction (Competition Contribution). In *Proc. TACAS (2) (LNCS 12652)*. Springer, 458–462. https://doi.org/10.1007/978-3-030-72013-1_32

Priyanka Darke, Bharti Chimdyalwar, R. Venkatesh, Ulka Shrotri, and Ravindra Metta. 2015. Over-Approximating Loops to Prove Properties Using Bounded Model Checking. In *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition* (Grenoble, France) *(DATE '15)*. EDA Consortium, San Jose, CA, USA, 1407–1412. https://doi.org/10.7873/DATE.2015.0245

Leonardo de Moura and Nikolaj Bjørner. 2008. Z3: An Efficient SMT Solver. In *Proceedings of the Theory and Practice of Software, 14th International Conference on Tools and Algorithms for the Construction and Analysis of Systems* (Budapest, Hungary) *(TACAS'08/ETAPS'08)*. Springer-Verlag, Berlin, Heidelberg, 337–340. https://doi.org/10.1007/978-3-540-78800-3_24 https://github.com/Z3Prover/z3.

Steven de Oliveira, Saddek Bensalem, and Virgile Prevosto. 2016. Polynomial Invariants by Linear Algebra. In *Automated Technology for Verification and Analysis*, Cyrille Artho, Axel Legay, and Doron Peled (Eds.). Springer International Publishing, Cham, 479–494. https://doi.org/10.1007/978-3-319-46520-3_30

Azadeh Farzan and Zachary Kincaid. 2015. Compositional recurrence analysis. In *2015 Formal Methods in Computer-Aided Design (FMCAD)*. 57–64. https://doi.org/10.1109/FMCAD.2015.7542253

Paul Feautrier. 1996. *Automatic parallelization in the polytope model*. Springer Berlin Heidelberg, Berlin, Heidelberg, 79–103. https://doi.org/10.1007/3-540-61736-1_44

Carsten Fuhs, Jürgen Giesl, Aart Middeldorp, Peter Schneider-Kamp, René Thiemann, and Harald Zankl. 2007. SAT Solving for Termination Analysis with Polynomial Interpretations. In *Theory and Applications of Satisfiability Testing – SAT 2007*, João Marques-Silva and Karem A. Sakallah (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 340–354. https://doi.org/10.1007/978-3-540-72788-0_33

Harald Ganzinger, George Hagen, Robert Nieuwenhuis, Albert Oliveras, and Cesare Tinelli. 2004. DPLL(T): Fast Decision Procedures. In *Computer Aided Verification*, Rajeev Alur and Doron A. Peled (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 175–188. https://doi.org/10.1007/978-3-540-27813-9_14

Sicun Gao, Jeremy Avigad, and Edmund M. Clarke. 2012. δ-Complete Decision Procedures for Satisfiability over the Reals. In *Automated Reasoning*, Bernhard Gramlich, Dale Miller, and Uli Sattler (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 286–300. https://doi.org/10.1007/978-3-642-31365-3_23

Susanne Graf and Hassen Saidi. 1997. Construction of abstract state graphs with PVS. In *Computer Aided Verification*, Orna Grumberg (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 72–83. https://doi.org/10.1007/3-540-63166-6_10

Bhargav S. Gulavani and Sumit Gulwani. 2008. A Numerical Abstract Domain Based on Expression Abstraction and Max Operator with Application in Timing Analysis. In *Proceedings of the 20th International Conference on Computer Aided Verification* (Princeton, NJ, USA) *(CAV '08)*. Springer-Verlag, Berlin, Heidelberg, 370–384. https://doi.org/10.1007/978-3-540-70545-1_35

William Hart, Fredrik Johansson, and Sebastian Pancratz. 2021. FLINT: Fast Library for Number Theory. Version 2.8.0, http://flintlib.org.

Chris Hawblitzel, Jon Howell, Jacob R. Lorch, Arjun Narayan, Bryan Parno, Danfeng Zhang, and Brian Zill. 2014. Ironclad Apps: End-to-End Security via Automated Full-System Verification. In *11th USENIX Symposium on Operating Systems Design and Implementation (OSDI 14)*. USENIX Association, Broomfield, CO, 165–181. https://www.usenix.org/conference/osdi14/technical-sessions/presentation/hawblitzel

Matthias Heizmann, Jochen Hoenicke, and Andreas Podelski. 2009. Refinement of Trace Abstraction. In *Proceedings of the 16th International Symposium on Static Analysis* (Los Angeles, CA) *(SAS '09)*. Springer-Verlag, Berlin, Heidelberg, 69–85. https://doi.org/10.1007/978-3-642-03237-0_7

Ehud Hrushovski, Joël Ouaknine, Amaury Pouly, and James Worrell. 2018. Polynomial Invariants for Affine Programs. In *Logic in Computer Science*. 530–539. https://doi.org/10.1145/3209108.3209142

Andreas Humenberger, Maximilian Jaroschek, and Laura Kovács. 2018. Invariant Generation for Multi-Path Loops with Polynomial Assignments. In *VMCAI*. 226–246. https://doi.org/10.1007/978-3-319-73721-8_11

Bertrand Jeannet and Antoine Miné. 2009. Apron: A Library of Numerical Abstract Domains for Static Analysis. In *Computer Aided Verification*, Ahmed Bouajjani and Oded Maler (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 661–667. https://doi.org/10.1007/978-3-642-02658-4_52 https://github.com/antoinemine/apron.

Dejan Jovanović. 2017. Solving Nonlinear Integer Arithmetic with MCSAT. In *Verification, Model Checking, and Abstract Interpretation*, Ahmed Bouajjani and David Monniaux (Eds.). Springer International Publishing, Cham, 330–346. https://doi.org/10.1007/978-3-319-52234-0_18

Dejan Jovanović and Leonardo de Moura. 2013. Solving Non-Linear Arithmetic. *ACM Commun. Comput. Algebra* 46, 3/4 (Jan 2013), 104–105. https://doi.org/10.1145/2429135.2429155

Zachary Kincaid, John Cyphert, Jason Breck, and Thomas Reps. 2017. Non-Linear Reasoning for Invariant Synthesis. *Proc. ACM Program. Lang.* 2, POPL, Article 54 (dec 2017), 33 pages. https://doi.org/10.1145/3158142

Zachary Kincaid, Nicolas Koh, and Shaowei Zhu. 2022a. *Artifact for article "When Less is More: Consequence Finding in a Weak Theory of Arithmetic"*. https://doi.org/10.5281/zenodo.7321183

Zachary Kincaid, Nicolas Koh, and Shaowei Zhu. 2022b. When Less Is More: Consequence-Finding in a Weak Theory of Arithmetic. https://doi.org/10.48550/ARXIV.2211.04000

Zachary Kincaid, Thomas Reps, and John Cyphert. 2021. Algebraic Program Analysis. In *International Conference on Computer Aided Verification*. Springer, 46–83. https://doi.org/10.1007/978-3-030-81685-8_3

Laura Kovács. 2008. Reasoning Algebraically About P-Solvable Loops. In *Tools and Algorithms for the Construction and Analysis of Systems*, C. R. Ramakrishnan and Jakob Rehof (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 249–264. https://doi.org/10.1007/978-3-540-78800-3_18

Gereon Kremer and Erika Ábrahám. 2020. Fully Incremental Cylindrical Algebraic Decomposition. *J. Symb. Comput.* 100, C (Sep 2020), 11–37. https://doi.org/10.1016/j.jsc.2019.07.018

Gereon Kremer, Florian Corzilius, and Erika Ábrahám. 2016. A Generalised Branch-and-Bound Approach and Its Application in SAT Modulo Nonlinear Integer Arithmetic. In *Computer Algebra in Scientific Computing - 18th International Workshop, CASC 2016, Bucharest, Romania, September 19-23, 2016, Proceedings (Lecture Notes in Computer Science, Vol. 9890)*, Vladimir P. Gerdt, Wolfram Koepf, Werner M. Seiler, and Evgenii V. Vorozhtsov (Eds.). Springer, 315–335. https://doi.org/10.1007/978-3-319-45641-6_21

Jean-Louis Krivine. 1964. Anneaux préordonnés. *Journal d'analyse mathématique* 12 (1964), p. 307–326. https://hal.archives-ouvertes.fr/hal-00165658

Christian Lengauer. 1993. Loop parallelization in the polytope model. In *CONCUR'93*, Eike Best (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 398–416. https://doi.org/10.1007/3-540-57208-2_28

Yi Li, Aws Albarghouthi, Zachary Kincaid, Arie Gurfinkel, and Marsha Chechik. 2014. Symbolic Optimization with SMT Solvers. *SIGPLAN Not.* 49, 1 (Jan 2014), 607–618. https://doi.org/10.1145/2578855.2535857

Murray A. Marshall. 2008. *Positive polynomials and sums of squares.* AMS. https://doi.org/10.1090/surv/146/02

Markus Müller-Olm and Helmut Seidl. 2004. Precise Interprocedural Analysis through Linear Algebra. In *Proceedings of the 31st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages* (Venice, Italy) *(POPL '04)*. Association for Computing Machinery, New York, NY, USA, 330–341. https://doi.org/10.1145/964001.964029

Greg Nelson and Derek C. Oppen. 1979. Simplification by Cooperating Decision Procedures. *ACM Trans. Program. Lang. Syst.* 1, 2 (Oct 1979), 245–257. https://doi.org/10.1145/357073.357079

Andreas Podelski and Andrey Rybalchenko. 2004. A Complete Method for the Synthesis of Linear Ranking Functions. In *Verification, Model Checking, and Abstract Interpretation*, Bernhard Steffen and Giorgio Levi (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 239–251. https://doi.org/10.1007/978-3-540-24622-0_20

Mihai Putinar. 1993. Positive Polynomials on Compact Semi-Algebraic Sets. *Indiana Univ. Math. J.* 42 (1993), 969–984. Issue 3.

Thomas Reps, Mooly Sagiv, and Greta Yorsh. 2004. Symbolic Implementation of the Best Transformer. In *Verification, Model Checking, and Abstract Interpretation*, Bernhard Steffen and Giorgio Levi (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 252–266. https://doi.org/10.1007/978-3-540-24622-0_21

Enric Rodríguez-Carbonell and Deepak Kapur. 2004. Automatic Generation of Polynomial Loop Invariants: Algebraic Foundations. In *Proceedings of the 2004 International Symposium on Symbolic and Algebraic Computation* (Santander, Spain) *(ISSAC '04)*. Association for Computing Machinery, New York, NY, USA, 266–273. https://doi.org/10.1145/1005285.1005324

Alexander Schrijver. 1980. On Cutting Planes. In *Combinatorics 79*, Peter L. Hammer (Ed.). Annals of Discrete Mathematics, Vol. 9. Elsevier, 291–296. https://doi.org/10.1016/S0167-5060(08)70085-2

Alexander Schrijver. 1999. *Theory of linear and integer programming.* Wiley.

Roberto Sebastiani and Silvia Tomasi. 2012. Optimization in SMT with $\mathcal{LA}(\mathbb{Q})$ Cost Functions. In *Automated Reasoning*, Bernhard Gramlich, Dale Miller, and Uli Sattler (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 484–498.

Abraham Seidenberg. 1954. A New Decision Method for Elementary Algebra. *Annals of Mathematics* 60, 2 (1954), 365–374. https://doi.org/10.2307/1969640

Alfred Tarski. 1949. A Decision Method for Elementary Algebra and Geometry. *Journal of Symbolic Logic* 14, 3 (1949), 188–188. https://doi.org/10.2307/2267068

Aditya Thakur, Akash Lal, Junghee Lim, and Thomas Reps. 2015. PostHat and All That: Automating Abstract Interpretation. *Electronic Notes in Theoretical Computer Science* 311 (2015), 15–32. https://doi.org/10.1016/j.entcs.2015.02.003 Fourth Workshop on Tools for Automatic Program Analysis (TAPAS 2013).

Aditya V. Thakur. 2014. *Symbolic Abstraction: Algorithms and Applications.* Ph. D. Dissertation. Comp. Sci. Dept., Univ. of Wisconsin, Madison, WI. Tech. Rep. 1812.

Cesare Tinelli. 2003. Cooperation of Background Reasoners in Theory Reasoning by Residue Sharing. *J. Autom. Reason.* 30, 1 (Jan 2003), 1–31. https://doi.org/10.1023/A:1022587501759

Ashish Tiwari. 2005. An Algebraic Approach for the Unsatisfiability of Nonlinear Constraints. In *Proceedings of the 19th International Conference on Computer Science Logic* (Oxford, UK) *(CSL'05)*. Springer-Verlag, Berlin, Heidelberg, 248–262. https://doi.org/10.1007/11538363_18

Ashish Tiwari and Patrick Lincoln. 2014. A Nonlinear Real Arithmetic Fragment. In *Computer Aided Verification*, Armin Biere and Roderick Bloem (Eds.). Springer International Publishing, Cham, 729–736. https://doi.org/10.1007/978-3-319-08867-9_48

Maarten. H. Van Emden and Robert A. Kowalski. 1976. The Semantics of Predicate Logic as a Programming Language. *J. ACM* 23, 4 (oct 1976), 733–742. https://doi.org/10.1145/321978.321991

Philipp Wendler and Dirk Beyer. 2022. *sosy-lab/benchexec: Release 3.11*. https://doi.org/10.5281/zenodo.6024083

Harald Zankl and Aart Middeldorp. 2010. Satisfiability of Non-linear (Ir)rational Arithmetic. In *Logic for Programming, Artificial Intelligence, and Reasoning*, Edmund M. Clarke and Andrei Voronkov (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 481–500. https://doi.org/10.1007/978-3-642-17511-4_27

Shaowei Zhu and Zachary Kincaid. 2021. Termination analysis without the tears. In *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation*. 1296–1311. https://doi.org/10.1145/3453483.3454110

Received 2022-07-07; accepted 2022-11-07