On Locally Minimal Nullstellensatz Proofs

Leonardo de Moura and Grant Olney Passmore
{leonardo@microsoft.com, g.passmore@ed.ac.uk}
Microsoft Research and LFCS, University of Edinburgh

Abstract

Hilbert's weak Nullstellensatz guarantees the existence of algebraic proof objects certifying the unsatisfiability of systems of polynomial equations not satisfiable over any algebraically closed field. Such proof objects take the form of ideal membership identities and can be found algorithmically using Gröbner bases and cofactor-based linear algebra techniques. However, these proof objects may contain redundant information: a proper subset of the equational assumptions used in these proofs may be sufficient to derive the unsatisfiability of the original polynomial system. For using Nullstellensatz techniques in SMT-based decision methods, a *minimal* proof object is often desired. With this in mind, we introduce a notion of *locally* minimal Nullstellensatz proofs and give ideal-theoretic algorithms for their construction.

1 Introduction

Modern Satisfiability Modulo Theories (SMT) solvers have application in the verification of software and hardware artifacts and are seeing increasing use in areas as diverse as planning and formalised mathematics. At a high-level, an SMT solver consists of an orchestrated combination of a DPLL based SAT solver and a number of satellite "theory" solvers (T-solvers) which implement decision methods for decidable elementary theories such as linear integer and real arithmetic, bit-vector arithmetic, and the theory of uninterpreted functions with equality. The effectiveness of an SMT decision loop depends crucially upon the ability of its T-solvers to identify "small" inconsistent components of formulas [3, 4]. Thus when one develops a new T-solver, the investigation of techniques for finding such "small" inconsistent subformulas is an important concern.

Many verification problems, such as those arising from hybrid systems, embedded and physical systems, and numerical algorithms, require deciding the satisfiability of non-linear arithmetical formulas over the real numbers. By Tarski [9], it is well known that the full elementary theory of polynomial real arithmetic is decidable, but classical (quantifier elimination) approaches to this problem are prohibitively expensive for formulas found in real applications. Recently, a number of new (semi-) decision

procedures for the quantifier-free fragment of this theory have been proposed [10, 7, 6]. All of them use a Gröbner bases procedure as a subroutine.

The work described in this paper can be seen as a contribution to the development of effective T-solvers for non-linear polynomial arithmetic over both the real and complex numbers. In particular, we consider the problem of finding "small" proof objects certifying the unsatisfiability of systems of polynomial equations over any algebraically closed field. We consider this problem within the context of Gröbner basis calculations.

We start by defining algebraic notions of proof minimality and redundancy, and two proof minimization transformations: *cofactor-subsumption* and *basis-subsumption*. Then, we describe a simple algorithm for extracting proof objects from a Gröbner bases procedure. Our algorithm is optimal for the linear case, that is, it produces only non-redundant proof objects. Finally, we show that a restricted form of cofactor subsumption can be efficiently implemented and used to reduce the amount of redundancy in our proof objects.

2 Background

Given $\{p_1, \ldots, p_k\}$, a finite subset of $\mathbb{Q}[\vec{x}]$, the polynomial ideal $\mathcal{I}(\{p_1, \ldots, p_k\})$ is the set of polynomials $\{\sum_{i=1}^{k} p_i q_i \mid q_i \in \mathbb{Q}[\vec{x}]\}$. Hilbert's Weak Nullstellensatz states that any set of polynomial equations $\{p_1 \simeq 0, \ldots, p_k \simeq 0\}$ is unsatisfiable over $\mathbb{C}^n$ iff $\mathcal{I}(p_1, \ldots, p_k) = \mathbb{Q}[\vec{x}]$. Therefore, if $\varphi = \bigwedge_{i=1}^{k} p_i \simeq 0$, then $\langle \mathbb{C}, +, -, *, 0, 1 \rangle \models \neg\exists\vec{x}(\varphi(\vec{x}))$ iff $\exists q_1, \ldots, q_k \in \mathbb{Q}[\vec{x}]$ s.t. $\sum_{i=1}^{k} p_i q_i = 1$. An element $x_1^{i_1} \ldots x_n^{i_n}$ in $\mathbb{Q}[x_1, \ldots, x_n]$ is called a *power-product* (or *term*), and an element $c x_1^{i_1} \ldots x_n^{i_n}$ with $c \in \mathbb{Q}$ and $x_1^{i_1} \ldots x_n^{i_n}$ a power-product is called a *monomial*. We say a monomial is *monic* if $c = 1$. (This terminology is not universally agreed upon.) We use $\mathbb{M}$ to denote the set of all power-products in $\mathbb{Q}[x_1, \ldots, x_n]$. From hereafter, we use p, q and r to denote polynomials, m to denote power-products and monic monomials, c to denote coefficients, and cm to denote monomials. We say a power-product $x_1^{i_1} \ldots x_n^{i_n}$ contains x_k if $i_k > 0$. Given two power-products $m_1 = x_1^{i_1} \ldots x_n^{i_n}$ and $m_2 = x_1^{j_1} \ldots x_n^{j_n}$, $m_1 m_2$ denotes the power-product $x_1^{i_1+j_1} \ldots x_n^{i_n+j_n}$, if $i_k \geq j_k$ for $k \in \{1, \ldots, n\}$, then $\frac{m_1}{m_2}$ denotes the power-product $x_1^{i_1-j_1} \ldots x_n^{i_n-j_n}$, and the *least common multiple* $\mathsf{lcm}(m_1, m_2)$ of m_1 and m_2 is the power product $x_1^{max(i_1,j_1)} \ldots x_n^{max(i_n,j_n)}$. We say a polynomial p *contains the power-product* m if p contains the monomial cm for some coefficient $c \neq 0$. Given a polynomial $p = c_1 m_1 + \ldots + c_n m_n$ and a monomial cm, we use cmp to denote the polynomial $(c_1 c)m_1 m + \ldots + (c_n c)m_n m$. Similarly, given a polynomal $p = c_1 m_1 + \ldots + c_n m_n$ and a polynomal q, we use pq to denote the polynomal $c_1 m_1 q + \ldots + c_n m_n q$. In the work that follows, all polynomials are assumed to be in a sum-of-monomials normal form (e.g., a polynomial will never contain two distinct monomials formed from the same power-product).

Given two monic monomials p_1 and p_2 of the form $\underline{m_1} + q_1$ and $\underline{m_2} + q_2$, let $\tau_{1,2}$ be the $\mathsf{lcm}(m_1, m_2)$, then we use $\mathsf{spol}(p_1, p_2)$ to denote the polynomial $(\frac{\tau_{1,2}}{m_1})q_1 -$

$(\frac{\tau_{1,2}}{m_2})q_2$. Given a set of polynomials S, it is easy to see that if $\{p_1, p_2\} \subseteq \mathcal{I}(S)$, then $\mathsf{spol}(p_1, p_2) \in \mathcal{I}(S)$.

An order relation $\prec$ on the set $\mathbb{M}$ is *admissible* if $m_1 \prec m_2$ implies that $m_1 m \prec m_2 m$, for all m_1, m_2 and m in $\mathbb{M}$. A *monomial order* is a total order on $\mathbb{M}$ which is admissible and a well ordering. Given two polynomials p_1 and p_2, we say $p_1 \prec p_2$ if there is a monomial cm in p_2 such that for all monomials $c_i m_i$ in p_1, $m_i \prec m$.

2.1 Abstract Gröbner Basis

Given a monomial order $\prec$, the key idea in Buchberger's algorithm is to use a polynomial $cm + q$, where $q \prec m$, as a rewrite rule $cm \rightarrow -q$. For clarity, we will write polynomials used as rewrite rules in a form in which the head monomial has been underlined. For instance, when using $\underline{cm} + q$ as a rewrite rule we will mean $cm \rightarrow -q$. We say a polynomial used as a rewrite rule $\underline{cm} + q$ is *monic* if $c = 1$. To simplify the presentation that follows, we will assume all polynomials used as rewrite rules are monic. The monic polynomial $p = \underline{m} + q$ induces a *reduction relation* $\mapsto_p$ on polynomials. It is defined as $q_1 + c_1 m_1 m \mapsto_p q_1 - c_1 m_1 q$ for arbitrary polynomials q_1 and monomials $c_1 m_1$. Given a set of monic polynomials $G = \{p_1, \ldots, p_k\}$, the reduction relation induced by G is defined as: $\mapsto_G = \bigcup_{i=1}^{k} \mapsto_{p_i}$.

Definition 1 (Gröbner bases)**.** *A finite set of monic polynomials G is a Gröbner basis of the ideal $\mathcal{I}(F)$ iff $\mathcal{I}(G) = \mathcal{I}(F)$ and $\mapsto_G$ is confluent.*

The inference rules in Figure 1 work on pairs of sets of polynomials (S, G). In all rules, the coefficients c and c_1 are assumed to be non-zero. We use $(S_1, G_1) \vdash (S_2, G_2)$ to indicate that (S_1, G_1) can be transformed to (S_2, G_2) by applying one of the inference rules in Figure 1. The proofs of all theorems in this section are included in [5].

Theorem 1. *$(S_1, G_1) \vdash (S_2, G_2)$ implies $\mathcal{I}(S_1 \cup G_1)) = \mathcal{I}(S_2 \cup G_2))$.*

Definition 2 (Procedure)**.** *A Gröbner basis procedure $\mathfrak{G}$ is a program that accepts a set of polynomials $\{p_1, \ldots, p_k\}$, a monomial order $\prec$, and uses the rules in Figure 1 to generate a (finite or infinite) sequence $(S_1 = \{p_1, \ldots, p_k\},\ G_1 = \emptyset) \vdash (S_2, G_2) \vdash (S_3, G_3) \vdash \ldots$. This sequence is called a* run *of $\mathfrak{G}$.*

Given a set of monic polynomials G, the set of S-polynomials $\mathsf{SP}(G)$ is defined as the set $\{\mathsf{spol}(p_1, p_2) \mid p_1, p_2 \in G\}$.

Definition 3 (Correct Procedure)**.** *A Gröbner basis procedure $\mathfrak{G}$ is said to be correct iff it produces only finite runs $(S_1,\ G_1 = \emptyset) \vdash \ldots \vdash (S_n = \emptyset,\ G_n)$, and $\mathsf{SP}(G_n) \subseteq (S_1 \cup S_2 \cup \ldots \cup S_{n-1})$.*

Theorem 2. *Let $\mathfrak{G}$ be a correct Gröbner basis procedure, then for any run $(S_1,\ G_1 = \emptyset) \vdash \ldots \vdash (S_n = \emptyset,\ G_n)$, G_n is a Gröbner basis for $\mathcal{I}(S_1)$.*

$$\textsf{Orient} \quad \frac{S \cup \{\underline{cm} + q\}, G}{S, G \cup \{\underline{m} + (\frac{1}{c})q\}}$$

$$\textsf{Superpose} \quad \frac{S, G \cup \{p_1, p_2\}}{S \cup \{\textsf{spol}(p_1, p_2)\}, G \cup \{p_1, p_2\}}$$

$$\textsf{Delete} \quad \frac{S \cup \{0\},\ G}{S, G}$$

$$\textsf{Simplify-S} \quad \frac{S \cup \{c_1 m_1 m_2 + q_1\}, G \cup \{\underline{m_2} + q_2\}}{S \cup \{q_1 - c_1 m_1 q_2\}, G \cup \{\underline{m_2} + q_2\}}$$

$$\textsf{Simplify-H} \quad \frac{S, G \cup \{\underline{m_1 m_2} + q_1,\ \underline{m_2} + q_2\}}{S \cup \{q_1 - m_1 q_2\}, G \cup \{\underline{m_2} + q_2\}} \quad \textbf{if } m_1 \neq 1$$

$$\textsf{Simplify-T} \quad \frac{S, G \cup \{\underline{m} + c_1 m_1 m_2 + q_1,\ \underline{m_2} + q_2\}}{S, G \cup \{\underline{m} - c_1 m_1 q_2 + q_1,\ \underline{m_2} + q_2\}}$$

Figure 1: Inference rules.

Definition 4 (Eager Simplification)**.** *Given a Gröbner basis procedure* $\mathfrak{G}$*, we say* $\mathfrak{G}$ *implements eager simplification iff* $\mathfrak{G}$ *only applies* Orient *to* $p \in S_i$ *when* Simplify-S *cannot be applied to* p*.*

Proposition 3. *Given a Gröbner basis procedure* $\mathfrak{G}$ *using eager simplification, then for any run* $(S_1, G_1) \vdash (S_2, G_2) \vdash \ldots$*, for all* $j \geq 1$*, there is no* $\underline{m_1} + q_1$ *and* $\underline{m_2} + q_2$ *in* G_j *such that* $m_1 = m_2$ *and* $q_1 \neq q_2$*. Moreover, in this case, the condition* $m_1 \neq 1$ *in the rule* Simplify-H *is only restricting self simplifications.*

Definition 5 (Fairness)**.** *A Gröbner basis procedure* $\mathfrak{G}$ *is said to be fair iff for any run* $(S_1, G_1) \vdash (S_2, G_2) \vdash \ldots$

$$\textsf{SP}(\bigcup_{i \geq 1} \bigcap_{j \geq i} G_j) \subseteq \bigcup_{i \geq 1} S_i.$$

Theorem 4. *If a Gröbner basis procedure* $\mathfrak{G}$ *implements eager simplification, is fair, and* Superpose *is applied at most once for any pair of polynomials in* $\bigcup_{i \geq 1} G_i$*, then* $\mathfrak{G}$ *is correct.*

3 Algebraic Notions of Proof Minimality

Let $\mathbb{B} = \{p_1, \ldots, p_k\}$ be a finite subset of $\mathbb{Q}[\vec{x}]$. As the considered Nullstellensatz proofs take the form of ideal membership certificates, we first build much of the algebraic machinery that follows in terms of *general* ideal membership certificates (e.g., those of the form $p \in \mathcal{I}(\mathbb{B})$ for arbitrary $p \in \mathbb{Q}[\vec{x}]$) and then later specialise the results to the case of Nullstellensatz proofs (e.g., those of the form $1 \in \mathcal{I}(\mathbb{B})$). We use the word "proof" to mean exclusively "Nullstellensatz proof" and "certificate" to mean "arbitrary ideal membership certificate," the latter of which could be a proof.

3.1 Algebraic Notions of Redundancy

Definition 6 (Basis redundancy)**.** *We say* $\mathbb{B}$ *is* p-non-redundant *iff* $p \in \mathcal{I}(\mathbb{B})$ *and* $\forall B \subset \mathbb{B}\ (p \notin \mathcal{I}(B))$*. Similarly, we say* $\mathbb{B}$ *is* p-redundant *iff* $p \in \mathcal{I}(\mathbb{B})$ *and* $\exists B \subset \mathbb{B}\ (p \in \mathcal{I}(B))$*.*

Definition 7 (Membership set)**.** *We define* $Mem(p, p_1, \ldots p_k) \subseteq \mathbb{Q}[\vec{x}]^k$ *to be the collection of (flat) ideal membership certificates showing* $p \in \mathcal{I}(p_1, \ldots, p_k)$ *as follows:*

$$Mem(p, p_1, \ldots, p_k) = \left\{ \langle q_1, \ldots, q_k \rangle \mid \sum_{i=1}^{k} p_i q_i = p \right\}.$$

When no confusion can arise, we will write $Mem(p, \mathbb{B})$ *in place of* $Mem(p, p_1, \ldots p_k)$*. Given* $\alpha \in Mem(p, \mathbb{B})$*, coordinate* $\alpha(i)$ *is known as the* i*th cofactor (of* p *w.r.t.* $\mathbb{B}$*) in* α*.*

Definition 8 (Proof set)**.** *We define* $\Pr(p_1, \ldots p_k)$ *to be the collection of (flat) Nullstellensatz proofs of the complex unsatisfiability of* $\{p_1 \simeq 0, \ldots, p_k \simeq 0\}$ [1] *over* $\mathbb{C}^n$*. That is,* $\Pr(p_1, \ldots p_k) = Mem(1, p_1, \ldots p_k)$*. When no confusion can arise, we will write* $\Pr(\mathbb{B})$ *in place of* $\Pr(p_1, \ldots p_k)$*.*

It is natural to identify the collection of hypotheses used in a certificate $\alpha \in Mem(p, \mathbb{B})$ with those members of $\mathbb{B}$ whose corresponding cofactors in α are non-zero.

Definition 9 (Basis of hypotheses)**.** *Given* $\alpha \in Mem(p, \mathbb{B})$*, we define* $Hyp(\mathbb{B}, \alpha)$ *to be the collection of* $\mathbb{B}$*-hypotheses used in* α *as follows:*

$$Hyp(\mathbb{B}, \alpha) = \{p_i \in \mathbb{B} \mid \alpha(i) \neq 0 \mid 1 \leq i \leq k\}.$$

Definition 10 (Non-redundant certificate)**.** *We say a membership certificate* $\alpha \in Mem(p, \mathbb{B})$ *is* non-redundant *iff the collection of* $\mathbb{B}$*-hypotheses used in* $\alpha, Hyp(\mathbb{B}, \alpha)$*, is* p*-non-redundant.*

[1] The interested reader may note the connection between $\Pr(p_1, \ldots, p_k)$ and the first syzygy module of $\langle p_1, \ldots, p_k \rangle$. In particular, $Syz(p_1, \ldots, p_k) = Mem(0, p_1, \ldots, p_k)$ while $\Pr(p_1, \ldots, p_k) = Mem(1, p_1, \ldots, p_k)$.

Observe that $\alpha \in Mem(p, \mathbb{B})$ (resp. $\alpha \in \Pr(\mathbb{B})$) is non-redundant iff $\neg\exists \alpha' \in Mem(p, \mathbb{B})$ (resp. $\alpha' \in \Pr(\mathbb{B})$) s.t. $Hyp(\mathbb{B}, \alpha') \subset Hyp(\mathbb{B}, \alpha)$. Thus if $\alpha \in \Pr(\mathbb{B})$ is a non-redundant proof, then no strict subset of the hypotheses used in the proof is sufficient to show the unsatisfiability of the system $\mathbb{B}$ over $\mathbb{C}^n$. However, this is an essentially *local* notion, dependent on the context of the current proof. In particular, the non-redundancy of a proof α does not in general mean that there is no smaller subset $B \subset \mathbb{B}$ s.t. $|B| < |Hyp(\mathbb{B}, \alpha)|$ that is itself unsatisfiable over $\mathbb{C}^n$. This can be seen with the following simple example.

Example 1. *Let the system* Γ *of polynomial equations be defined as follows:*

$$\Gamma = \{x^2y^2 - 1 \simeq 0,\ x^2y \simeq 0,\ xy \simeq 0,\ x + 1 \simeq 0,\ y + 1 \simeq 0\}.$$

Let $B = \{x^2y^2 - 1, x^2y, xy, x + 1, y + 1\}$ *be the basis of polynomials corresponding to* Γ*. Observe that* $\Pr(B) \neq \emptyset$*. Among others, it contains the following two proofs:*

$$\alpha = \langle -1, y, 0, 0, 0\rangle \text{ corresponding to } 1 = (-1)(x^2y^2 - 1) + y(x^2y), \text{ and}$$
$$\beta = \langle 0, 0, 1, -y, 1\rangle \text{ corresponding to } 1 = xy + -y(x + 1) + y + 1.$$

Then, we have, $Hyp(B, \alpha) = \{x^2y^2 - 1, x^2y\}$, $Hyp(B, \beta) = \{xy, x + 1, y + 1\}$. *Observe that both* $Hyp(B, \alpha)$ *and* $Hyp(B, \beta)$ *are non-redundant and* $|Hyp(B, \alpha)| < |Hyp(B, \beta)|$.

Thus, non-redundancy of a proof does not mean it is a proof that uses the *globally* least number of hypotheses, but rather that it is in some sense *locally* minimal: If one begins with a non-redundant proof and drops any used hypothesis, then no proof of unsatisfiability for the resulting system will exist. This is made precise with the following lemma.

Lemma 1. *Let* $\alpha \in \Pr(\mathbb{B})$ *be a non-redundant proof. Then, every* $B \subset Hyp(\mathbb{B}, \alpha)$ *is satisfiable over* $\mathbb{C}^n$.

We now wish to address the following fundamental problem: Given a certificate $\alpha \in Mem(p, \mathbb{B})$, can α be feasibly transformed into a non-redundant certificate? With feasibility in mind, we look only for transformations which arise by a combination of (i) dropping used hypotheses and (ii) modifying non-zero cofactors. In particular, all transformations $\alpha \mapsto \alpha'$ are s.t. $Hyp(\mathbb{B}, \alpha') \subset Hyp(\mathbb{B}, \alpha)$. In devising such techniques, one needs to refer to individual hypotheses contributing to the redundancy.

Definition 11. *Given a certificate* $\alpha \in Mem(p, \mathbb{B})$ *and a* j *s.t.* $1 \leq j \leq k$*, we say* α *is* j*-redundant iff* $\alpha(j) \neq 0$ *and* $Mem(p, Hyp(\mathbb{B}, \alpha) \setminus \{p_j\}) \neq \emptyset$.

3.2 Redundancy in the Linear Case

Before discussing the elimination of redundancy in the general non-linear setting, it is instructive to examine the linear case. If $\mathbb{B}$ is a system of linear polynomials, then

the calculation of a Gröbner basis for $\mathbb{B}$ degenerates into Gaussian elimination. By adopting the strategy of *eager simplification*, one can guarantee that for every proof $\alpha \in Mem(p, \mathbb{B})$, α is not j-redundant.

Theorem 5. *If $\mathfrak{G}$ is a fair Gröbner basis procedure implementing* eager simplification*, $p \in \mathcal{I}(\mathbb{B})$, p and $\mathbb{B}$ are linear, then forall $\alpha \in Mem(p, \mathbb{B})$, α is non-redundant.*[2]

Thus the simple process of excluding all p_i s.t. $\alpha(i) = 0$ from contributing to a certificate, as is done by the use of $Hyp(\mathbb{B}, \alpha)$ in our definition of redundancy, is sufficient to eliminate all redundant linear certificates when an eagerly simplifying Gröbner basis procedure is used. If eager simplification is not used, however, this property may fail to hold. In [2], we describe an example where a redundant certificate is produced by a non eagerly simplifying Gröbner basis procedure.

3.3 Redundancy in the General Case

We now return to proof redundancy in the context of the general non-linear case. The following concepts form the basis for our proof minimization transformations.

Definition 12. *Given a certificate $\alpha \in Mem(p, \mathbb{B})$ and a j s.t. $1 \leq j \leq k$, we say α is*

- *j-cofactor-subsumed $\iff$ $\alpha(j) \in \mathcal{I}(B)$ s.t. $B \subseteq (Hyp(\mathbb{B}, \alpha) \setminus \{p_j\})$,*
- *j-basis-subsumed $\iff$ $p_j \in \mathcal{I}(B)$ s.t. $B \subseteq (Hyp(\mathbb{B}, \alpha) \setminus \{p_j\})$,*
- *j-$\star$-subsumed $\iff$ $\alpha(j)p_j \in \mathcal{I}(B)$ s.t. $B \subseteq (Hyp(\mathbb{B}, \alpha) \setminus \{p_j\})$.*

We use $\mathbf{1}_j$ to denote $\langle q_1, \ldots, q_k \rangle \in \mathbb{Q}[\vec{x}]^k$, where $q_j = 1$, and $q_i = 0$ for all $j \neq i$. Let α and β be in $\mathbb{Q}[\vec{x}]^k$, and p in $\mathbb{Q}[\vec{x}]$. Then $\alpha + \beta$ denotes $\langle \alpha(1) + \beta(1), \ldots, \alpha(k) + \beta(k) \rangle$, and $p\alpha$ denotes $\langle p\alpha(1), \ldots, p\alpha(k) \rangle$. First, we focus on cofactor-subsumption. Note that j-cofactor-subsumption is an algebraic generalisation – using the intuition that ideals are an algebraic generalisation of zeroness – of the fact that if a cofactor coordinate $\alpha(j)$ of a certificate is explicitly 0, then its corresponding hypothesis p_j does not contribute to the certificate in an essential way. Let $\alpha \in Mem(p, \mathbb{B})$ and $\beta \in Mem(\alpha(j), \mathbb{B})$ with $Hyp(\mathbb{B}, \beta) \subseteq Hyp(\mathbb{B}, \alpha) \setminus \{p_j\}$. Then, we define the *certificate transformer* $\prod_{j,\beta}(\alpha)$ for j-cofactor-subsumption (w.r.t. $\mathbb{B} = \{p_1, \ldots, p_k\}$) as $\alpha + (-\alpha(j))\mathbf{1}_j + p_j\beta$.

Theorem 6. *Let $\alpha \in Mem(p, \mathbb{B})$ be a j-cofactor-subsumed certificate with $Hyp(\mathbb{B}, \alpha) = B$, and $\beta \in Mem(\alpha(j), \mathbb{B})$ with $Hyp(\mathbb{B}, \beta) \subseteq B \setminus \{p_j\}$. Then, $\prod_{j,\beta}(\alpha) \in Mem(p, \mathbb{B})$, and $Hyp(\mathbb{B}, \prod_{j,\beta}(\alpha)) \subseteq B \setminus \{p_j\}$.*

The proof of Theorem 6 consists of straightforward algebraic manipulation. Similarly, we define the *certificate transformer* $\coprod_{j,\beta}(\alpha)$ for j-basis-subsumption (w.r.t.

[2]The proof of this theorem is included in [2].

$\mathbb{B} = \{p_1, \ldots, p_k\}$) as $\alpha + (-\alpha(j))\mathbf{1}_j + \alpha(j)\beta$. Note that, in this case, $\beta \in Mem(p_j, \mathbb{B})$. Finally, we reveal that j-$\star$-subsumption is actually not needed. This is because $\mathbb{Q}[\vec{x}]$ is an integral domain, and thus a given certificate $\alpha \in Mem(p, \mathbb{B})$ is j-$\star$-subsumed iff it is either j-cofactor-subsumed or j-basis-subsumed.

4 Algorithmics and SMT

We now address the problem of how to build certificates in Gröbner basis procedures based on the inference rules in Figure 1. A *certified polynomial* (w.r.t. $\mathbb{B}$) is a pair (p, α) s.t. $\alpha \in Mem(p, \mathbb{B})$. The basic idea is lift the rules in Figure 1 to certified polynomials. For example, the lifted Simplify-S rule is:

$$\textsf{Simplify-S} \quad \frac{S \cup \{(c_1 m_1 m_2 + q_1,\ \alpha_1)\}, G \cup \{(\underline{m_2} + q_2,\ \alpha_2)\}}{S \cup \{(q_1 - c_1 m_1 q_2,\ \alpha_1 - c_1 m_1 \alpha_2)\}, G \cup \{(\underline{m_2} + q_2,\ \alpha_2)\}}$$

The remaining rules are described in [2].

Definition 13 (Certified Procedure)**.** *A certified Gröbner basis procedure $\mathfrak{G}$ is a program that accepts a set of polynomials $\{p_1, \ldots, p_k\}$, a monomial order $\prec$, and uses the lifted versions of the rules in Figure 1 to generate a (finite or infinite) sequence $(S_1 = \{(p_1, \mathbf{1}_1), \ldots, (p_k, \mathbf{1}_k)\},\ G_1 = \emptyset) \vdash (S_2, G_2) \vdash (S_3, G_3) \vdash \ldots$*

Note that if $(1, \alpha) \in S_i$ for some i, then α is a proof for the unsatisfiability of $\{p_1 \simeq 0, \ldots, p_k \simeq 0\}$ over $\mathbb{C}^n$.

In the linear case, *zero variables* are used to represent certified polynomials using a single polynomial [1, 8]. The idea is to represent the certified polynomial (p, α) as $p - \alpha(1)z_1 - \ldots - \alpha(k)z_k$, where z_i's are new fresh variables. The new polynomial is still linear because $\alpha(i)$ is always a constant for the linear case. An approach based on zero variables is attractive because a regular procedure can be easily used to obtain certificates. The main idea is to make the zero variables z_i smaller than the variables $\{x_1, \ldots, x_n\}$. This approach cannot be directly applied to the non linear case, because it would require us to make any monomial containing a zero variable z_i smaller than a monomial not containing any zero variable. There is no monomial order with such property, because it violates admissibility. For example, it would require $z_2 x_1 \prec x_1$.

4.1 Structured Certificates

The overhead in a certified Gröbner basis procedure is substantial, since the certificates α can grow in size very quickly. Moreover, it wasteful to compute a certificate for a polynomial that is deleted using the Delete rule. We address this issue using *structured certificates*. Structured certificates are represented using the constructors A (assumption), S (superpose), R (simplify), D (divide).

Definition 14 (Set of Polynomial Structured Certificates)**.** *The set of polynomial structured certificates, $\mathcal{C}$, is defined as the least set s.t.*

Assert: $p \in \mathbb{Q}[\vec{x}] \Longrightarrow \mathsf{A}(p) \in \mathcal{C}$,

Superpose: $\varphi_1, \varphi_2 \in \mathcal{C} \Longrightarrow \mathsf{S}(\varphi_1, \varphi_2) \in \mathcal{C}$,

Simplify: $\varphi_1, \varphi_2 \in \mathcal{C} \ \wedge \ m \in \mathbb{M} \Longrightarrow \mathsf{R}(\varphi_1, \varphi_2, m) \in \mathcal{C}$,

Divide: $\varphi \in \mathcal{C} \Longrightarrow \mathsf{D}(\varphi) \in \mathcal{C}$.

Using structured certificates, the lifted Simplify-S rule is:

$$\textsf{Simplify-S} \quad \frac{S \cup \{(c_1 m_1 m_2 + q_1,\ \varphi_1)\}, G \cup \{(\underline{m_2} + q_2,\ \varphi_2)\}}{S \cup \{(q_1 - c_1 m_1 q_2,\ \mathsf{R}(\varphi_1, \varphi_2, m_1 m_2))\}, G \cup \{(\underline{m_2} + q_2,\ \varphi_2)\}}$$

The remaining rules are described in [2]. The set of hypothesis $hyp(\varphi)$ of a structured certificate φ is defined as: $hyp(\mathsf{A}(p)) = p$, $hyp(\mathsf{S}(\varphi_1, \varphi_2)) = hyp(\mathsf{R}(\varphi_1, \varphi_2, m)) = hyp(\varphi_1) \cup hyp(\varphi_2)$, and $hyp(\mathsf{D}(\varphi)) = hyp(\varphi)$. Given a structured certificate φ (w.r.t. $\mathbb{B}$), it is straightforward to define a function *flat* that maps φ into a (flat) certificate $\mathit{flat}(\varphi)$ [3].

4.2 Restricted cofactor-subsumption and basis-subsumption

We use j-subsumption to denote j-cofactor-subsumption and j-basis-subsumption. We now address the following issue: How to apply j-subsumption effectively in practice? In general, it is too expensive to check whether a certificate α can be j-subsumed or not, because it requires us to answer ideal membership subqueries. That is, given a certificate α, to check whether α can be j-subsumed, we need to compute a Gröbner basis for $Hyp(\mathbb{B}, \alpha) \setminus \{p_j\}$. We overcome this difficulty by approximating the ideal membership subqueries. The idea is to answer these queries using a set of rewrite rules that is not necessarily confluent.

Definition 15 (j-φ-Independent Polynomial)**.** *Given a certificate φ, a certified polynomial (r, φ') is j-φ-independent iff $hyp(\varphi') \subseteq hyp(\varphi) \setminus \{p_j\}$.*

Let $(S_1, G_1) \vdash \ldots \vdash (S_m, G_m)$ be a run produced by a certified Gröbner basis procedure $\mathfrak{G}$, (p, φ) be some certified polynomial in $\cup_{i=0}^{m}(S_i \cup G_i)$, and $\Delta_{j,\varphi}$ be the set of j-φ-independent polynomials in $\cup_{i=0}^{m} G_i$. Now, suppose we want to check whether $\alpha = \mathit{flat}(\varphi)$ is j-cofactor-subsumed or not. Then, we can simply check whether $\alpha(j)$ rewrites to 0 using an arbitrary subset of $\Delta_{j,\varphi}$. For example, in our prototype, we do not track all polynomials produced in a run. Thus, whenever a certified polynomial (c, φ) (with $c \neq 0$) is included in S_m, we use just the j-φ-independent polynomials in G_m (instead of $\cup_{i=0}^{m} G_i$) to check whether $\mathit{flat}(\varphi)$ can be j-cofactor-subsumed or not.

Example 2. *Let S be a set of polynomials $\{p_1,\ p_2,\ p_3,\ p_4\}$, where:*

$$p_1 = x_1 - x_2, \quad p_2 = x_1 x_3^2 - x_1 x_4^2 + 1, \quad p_3 = x_5 x_4 - x_3, \quad p_4 = x_5 x_3 - x_4$$

[3] The function *flat* is defined in [2].

The set $\{p_1 \simeq 0,\ p_2 \simeq 0,\ p_3 \simeq 0,\ p_4 \simeq 0\}$ *is unsatisfiable over* $\mathbb{C}^5$. *Let* $\mathfrak{G}$ *be a correct Gröbner basis procedure that produces the run* $(S_1 = S, G_1 = \emptyset) \vdash \ldots \vdash (S_m, G_m)$, *where* S_m *contains the certified polynomial* $(1, \varphi)$, *where:*

$$\varphi = \mathsf{R}(\mathsf{S}(p_3, p_4), \mathsf{R}(\mathsf{A}(p_1), \mathsf{R}(\mathsf{A}(p_1), \mathsf{A}(p_2), x_3^2), x_4^2), x_2)$$

The flat certificate $\mathit{flat}(\varphi)$ *associated with* φ *is:*

$$\mathit{flat}(\varphi) = \langle(-x_3^2 + x_4^2),\ 1,\ x_2x_3,\ \ -x_2x_4\rangle.$$

Assume also that some G_i *in the run contains the certified polynomial* $(r, \varphi') = (x_3 - x_4,\ \mathsf{S}(\mathsf{A}(p_3), \mathsf{A}(p_4)))$. *Note that* (r, φ') *is* 1-φ-*independent, and* $-x_3^2 + x_4^2 \mapsto_r 0$. *Thus,* $\mathit{flat}(\varphi)$ *can be* 1-*cofactor-subsumed.*

5 Conclusion

The effectiveness of an SMT solver depends crucially upon the ability of its T-solvers to identify "small" inconsistent set of formulas. Hence, we defined algebraic notions of proof minimality and redundancy for Hilbert's Weak Nullstellensatz, and two useful certificate transformations: *cofactor-subsumption* and *basis-subsumption*. We also described how certificates can be extracted in the framework of abstract Gröbner Basis.

References

[1] G. B. Alan and A. Borning. The cassowary linear arithmetic constraint solving algorithm. *ACM Transactions on Computer Human Interaction*, 1998.

[2] L. de Moura and G. O. Passmore. On locally minimal nullstellensatz proofs. Technical report, Microsoft Research, 2009.

[3] L. de Moura, H. Rueß, and N. Shankar. Justifying equality. In *PDPAR'04*, 2004.

[4] R. Nieuwenhuis and A. Oliveras. Fast Congruence Closure and Extensions. *Inf. Comput.*, 2005(4), 2007.

[5] G. O. Passmore and L. de Moura. Superfluous s-polynomials in strategy-independent gröbner bases. to appear.

[6] G. O. Passmore and P. B. Jackson. Combined decision techniques for the existential theory of the reals. In *Calculemus'09*, 2009.

[7] A. Platzer, J. Quesel, and P. Rümmer. Real world verification. In *CADE-22*, 2009.

[8] H. Rueß and N. Shankar. Solving linear arithmetic constraints. Technical Report SRI-CSL-04-01, SRI International, 2004.

[9] A. Tarski. A decision method for elementary algebra and geometry. Technical report, 2nd edn. University of California Press, Berkeley, 1951.

[10] A. Tiwari. An algebraic approach for the unsatisfiability of nonlinear constraints. In *CSL'05*, volume 3634 of *LNCS*, 2005.