SoK: Gröbner Basis Algorithms for Arithmetization Oriented Ciphers

Jan Ferdinand Sauer and Alan Szepieniec

AS Discrete Mathematics GmbH, Switzerland
{ferdinand,alan}@asdm.gmbh

Abstract. Many new ciphers target a concise algebraic description for efficient evaluation in a proof system or a multi-party computation. This new target for optimization introduces algebraic vulnerabilities, particularly involving Gröbner basis analysis. Unfortunately, the literature on Gröbner bases tends to be either purely mathematical, or focused on small fields. In this paper, we survey the most important algorithms and present them in an intuitive way. The discussion of their complexities enables researchers to assess the security of concrete arithmetization-oriented ciphers. Aside from streamlining the security analysis, this paper helps newcomers enter the field.

Keywords: Algebraic Cryptanalysis · Gröbner Basis · Arithmetization Oriented Cipher.

1 Introduction

A range of emerging applications build on cryptographic protocols that employ *arithmetization.* This process first characterizes a computation as a sequence of basic finite field operations, and then performs these operations in the context of a cryptographic protocol that preserves some target security property, such as privacy, soundness, *etc.* Some examples of protocols that arithmetize are zero-knowledge proof systems [16, 56], succinct-verifier proof systems [27], and multi-party protocols [38] – all of which have numerous already-deployed applications both in a blockchain context and elsewhere.

A recurring computational task in these applications consists of evaluating a keyed or unkeyed symmetric cryptographic primitive. However, traditional block ciphers and hash functions are optimized for hardware and software implementations, not for arithmetic simulation by a cryptographic protocol. The demand for high performance in these applications has brought about two notable changes to the symmetric cryptography landscape:

1. *New designs for symmetric primitives optimized for arithmetic protocols.* Recent years have seen several new proposals and design strategies with this selling point: LowMC [6], MiMC [5], GMiMC [4], Jarvis and Friday [10], Vision and Rescue [7,81], Starkad and Poseidon [58]. These so-called *arithmetization-oriented ciphers (AOCs)* target a concise description in terms of operations over a large finite field.

2. *Renewed interest in the field of algebraic attacks on symmetric primitives.* As a result of their concise description in terms of finite field operations, AOCs are vulnerable to a collection of attacks that exploit the low degree or sparsity of polynomial representations of the cipher under attack [3,18,25,66]. Traditional ciphers mitigate this vulnerability by including operations not native to the working field in their circuit, resulting in a huge description in terms of polynomials. As a result, most of traditional symmetric cryptanalysis focuses on statistical attacks such as differential and linear cryptanalysis. For AOCs, these types of attacks are generally less relevant as a consequence of their large working field.

One of the most powerful general-purpose algebraic attack methodologies revolves around the computation of a Gröbner basis from a set of polynomials. This Gröbner basis computation is one step in a polynomial system solving problem where the polynomials are provided by the cipher's description, and the polynomials' common solution contains the sought-after secret key or hash preimage. The complexity of polynomial system solving is therefore a vitally important question in the context of analyzing the security of an AOC or determining the correct parameters for a target security level.

Unfortunately, polynomial system solving and Gröbner bases remain poorly understood subjects, particularly in the context of AOCs. There are several reasons for this poor understanding. Firstly, until recently, there was no demand for AOCs and thus no reason for cryptanalysts to study their security. Secondly, the field is not very developed, in the sense that contributions come from rather few researchers with diverse backgrounds, hampering collaboration. Thus thirdly, and most importantly, the literature on the topic is scattered across many sources, often contradictory, and in many cases out of date.

This paper aims to systematize existing knowledge in the context of Gröbner basis attacks on AOCs. No scientific novelty is claimed; our contribution is merely the presentation of existing knowledge in an intuitive and self-consistent way. In doing so, we hope to make the field more accessible to researchers and eventually improve the understanding of the concrete security that various AOCs offer.

The structure and scope of this paper is best illustrated in relation to the general attack pipeline, which consists of several steps.

1. *Polynomial modeling.* The cipher is described in terms of low-degree multivariate polynomials, such that any solution common to all polynomials results in a win for the attacker.
2. *Gröbner basis computation in degree-refining term order.* A Gröbner basis algorithm computes a Gröbner basis for the ideal spanned by this list of polynomials, relative to a term order refining the degree, typically *degrevlex*.
3. *Term order change.* The Gröbner basis with respect to *degrevlex* is transformed into a Gröbner basis for the same polynomial ideal but with respect to *lex*, an alternative term order.
4. *Solution readout.* Using a mixture of the Euclidean algorithm, univariate polynomial factorization, and back-substitution of roots, an attacker extracts a common solution to the system of polynomials.

Step 1, modeling the cipher and attack, is out of scope for this paper. We assume an attacker already knows which polynomials they want to find a solution to. Indeed, the concise polynomial formulation of AOCs is their selling point.

In Section 2, we cover the basic definitions of variety, ideal, term order, and Gröbner basis. We show here how solution readout works for Gröbner bases in *lex* order of zero-dimensional ideals, and how a naïve transformation of Buchberger's criterion gives rise to a correct (albeit slow) Gröbner basis algorithm.

In Section 3, we cover the F_4 and F_5 algorithms, which constitute the state of the art. We also cover the XL family of algorithms, which is popular in the context of cryptanalysis and small fields. While these algorithms are often described as Gröbner-like due to their various similarities, the framing is quite different.

In Section 4, we cover FGLM and sketch the Gröbner walk, the two most important term order change algorithms. It is important to consider the term order change as a separate step in the attack because the Gröbner basis computation does not always dominate. Indeed, it is possible to construct AOCs where the Gröbner basis comes for free, and where the cipher's security comes from the difficulty of computing this term order change.

We close in Section 5 with open questions and a discussion, and suggest future work.

2 Gröbner Bases

After modeling a cryptographic primitive as a system of polynomial equations, the objective is to find a solution to that system of equations, i.e., some vector $\boldsymbol{a}$ that simultaneously satisfies all multivariate polynomial equations. The set of all these values is called the *variety.*

Definition 1 (Variety[1]). *The variety of* $\mathcal{F} \subseteq \mathbb{F}[\mathbf{x}]$ *is defined as*

$$V(\mathcal{F}) = \left\{ (a_0, \ldots, a_{n-1}) \in \overline{\mathbb{F}}^n \,\middle|\, \forall f \in \mathcal{F} : f(\boldsymbol{a}) = 0 \right\}.$$

Note that varieties are defined over the algebraic closure $\overline{\mathbb{F}}$ of $\mathbb{F}$. For this paper, solutions not lying in the base field $\mathbb{F}$ are not of interest; only elements of $\mathbb{F}$ are valid inputs (keys, messages, *etc.*) and outputs (ciphertexts, hashes, *etc.*). Therefore, special care must be taken to eliminate parasitical extension field solutions, for instance by adjoining the field equations.

Identifying each equation with its polynomial allows us to operate on polynomials rather than on polynomial equations. Which operations on lists of polynomials can bring us closer to finding a solution? Linear algebra techniques suffice when the polynomials are linear. For higher degree polynomials, Gröbner basis algorithms provide a standard methodology. We motivate this notion before providing the definition.

[1] We only consider *affine* varieties in this work. For projective varieties, corresponding to homogeneous ideals, we refer to the literature on algebraic geometry, like [60].

A natural requirement for allowable operations is that they neither create nor destroy solutions. Adding and subtracting the polynomial equations to derive new ones satisfies this requirement, as does multiplying them with scalar weights. In fact, the weights need not be scalars but can be drawn from the same ring of polynomials over which the original polynomials themselves are defined. This observation motivates the notion of *polynomial ideals*, a generalization of vector spaces that allows polynomial coefficients instead of just scalar coefficients.

Definition 2 (Polynomial[2] ideal). *Let* $\mathcal{F} = \{f_0, \ldots, f_{s-1}\} \subseteq \mathbb{F}[\mathbf{x}]$ *be a set of polynomials. The ideal* I *spanned by* $\mathcal{F}$ *is defined as*

$$I = \langle \mathcal{F} \rangle = \langle f_0, \ldots, f_{s-1} \rangle = \left\{ \sum_{i=0}^{s-1} q_i f_i \;\middle|\; q_i \in \mathbb{F}[\mathbf{x}] \right\}.$$

The set of cosets of I in $\mathbb{F}[\mathbf{x}]$, together with polynomial addition and multiplication, forms a ring. We can therefore consider any polynomial f equivalent to some polynomial $g \equiv f \bmod I$ whenever $f - g \in I$. Intuitively, a useful property of a basis for I is for it to allow the computation of a canonical representative for each equivalence class, as this would admit an equivalence test. This property is useful for finding solutions because the equivalence class of all polynomials that encode information about the solution is $0 \bmod I$. The canonical representatives therefore distinguish useful polynomials from not so useful ones.

This observation is related to divisibility of univariate polynomials. If $f(x)$ is a univariate polynomial and $\langle f(x) \rangle$ its ideal, then the computation of canonical representative of a polynomial $g(x)$ modulo $\langle f(x) \rangle$ follows straightforwardly from polynomial long division: the canonical representative of $g(x)$ is its remainder after division by $f(x)$. Moreover, if $a \in \mathbb{F}$ is a root of $f(x)$, then $(x - a) \mid f(x)$ and one might even write $f(x) \in \langle x - a \rangle$ or $f(x) \equiv 0 \bmod x - a$. If the roots of $f(x)$ and of $g(x)$ are disjoint sets, then $f(x) \not\equiv 0 \bmod g(x)$ and $g(x) \not\equiv 0 \bmod f(x)$.

Unfortunately, polynomial division does not naturally generalize to multivariate polynomials. Univariate polynomial division relies on the natural order of univariate monomials,[3] namely comparing degrees. That is, $x^a \prec x^b$ if and only if $a < b$. Division of univariate monomial x^a by x^b results in x^{a-b}, which is an element of $\mathbb{F}[x]$ if $a \geqslant b$. The generalization to multivariate monomials is straightforward: $\mathbf{x}^{\boldsymbol{\alpha}}$ divided by $\mathbf{x}^{\boldsymbol{\beta}}$ results in $\mathbf{x}^{\boldsymbol{\alpha}-\boldsymbol{\beta}}$, an element of $\mathbb{F}[\mathbf{x}]$ if $\alpha_i \geqslant \beta_i$ for all i. However, generalizing polynomial division is a little more subtle. In the univariate case, the monomials with the highest power of the variable determine the next step in the division process. In the multivariate case, "highest power" is no inherent property: How do the degrees of x^5yz^2 and $x^2y^2z^4$ relate? In order to proceed, we must therefore fix the monomial order.

[2] In this paper, "ideal" always means "polynomial ideal."

[3] The terminology of "term" and "monomial" is not consistent in the literature. We understand a term as a coefficient times a monomial, like, for example, [26,36,37,39]. The meaning is reversed for others [33,46,47]. Some works use "power product" [20,23]. A comprehensive summary of this paper's notation can be found in Appendix A.

2.1 Monomial Orders

For multivariate division to be well defined, a *monomial order* $\prec$ must respect multiplication with monomials, i.e., evaluating operator $\prec$ must give the same result when both operands are multiplied by any monomial $m \in \mathcal{M}$. In other words, $\forall m \in \mathcal{M} : \mathbf{x}^{\boldsymbol{\alpha}} \prec \mathbf{x}^{\boldsymbol{\beta}} \Leftrightarrow m\mathbf{x}^{\boldsymbol{\alpha}} \prec m\mathbf{x}^{\boldsymbol{\beta}}$. Furthermore, $\prec$ must be a well-order, i.e., (1) the monomial $1 = \mathbf{x}^{(0,\dots,0)}$ is the smallest element, and (2) any two monomials are comparable, i.e., $\prec$ is a total order. There are infinitely many orders, but common ones have descriptive identifiers.

Lex In *lexicographical* order, $\mathbf{x}^{\boldsymbol{\alpha}} \prec_{\text{lex}} \mathbf{x}^{\boldsymbol{\beta}}$ if and only if the first nonzero entry of $\boldsymbol{\alpha} - \boldsymbol{\beta}$ is negative. In other words, $\mathbf{x}^{\boldsymbol{\alpha}} \prec_{\text{lex}} \mathbf{x}^{\boldsymbol{\beta}}$ if and only if there exists $0 \leqslant i < n$ such that $\alpha_i < \beta_i$ and $\alpha_j = \beta_j$ for all $0 \leqslant j < i$.

Example 1. In $\mathbb{F}[x, y, z]$ and *lex* order, we have $xy^2 \prec xy^2z \prec x^2z^2 \prec x^2yz \prec x^3$.

Deglex For *degree lexicographical* or *graded lexicographical* order, the monomial's total degrees are compared first, with ties broken by *lex*. In other words, $\mathbf{x}^{\boldsymbol{\alpha}} \prec_{\text{deglex}} \mathbf{x}^{\boldsymbol{\beta}}$ if and only if $\sum_i \alpha_i < \sum_i \beta_i$ or $\sum_i \alpha_i = \sum_i \beta_i$ and $\mathbf{x}^{\boldsymbol{\alpha}} \prec_{\text{lex}} \mathbf{x}^{\boldsymbol{\beta}}$.

Example 2 (continued). In *deglex* order, $xy^2 \prec x^3 \prec xy^2z \prec x^2z^2 \prec x^2yz$.

Degrevlex The *degree reverse lexicographical* or *graded reverse lexicographical* order (abbreviated *grevlex*) also first considers the monomial's total degree. Ties are broken using *invlex*, which is *lex* with inversely labeled variables. The outcome of the comparison is reversed, such that *invlex*-smaller monomials of the same degree are considered *degrevlex*-greater. In other words, $\mathbf{x}^{\boldsymbol{\alpha}} \prec_{\text{degrevlex}} \mathbf{x}^{\boldsymbol{\beta}}$ if and only if either $\sum_i \alpha_i < \sum_i \beta_i$ or $\sum_i \alpha_i = \sum_i \beta_i$ and $\mathbf{x}^{\boldsymbol{\alpha}} \succ_{\text{invlex}} \mathbf{x}^{\boldsymbol{\beta}}$.

Example 3 (continued). In *degrevlex* order, $xy^2 \prec x^3 \prec x^2z^2 \prec xy^2z \prec x^2yz$.

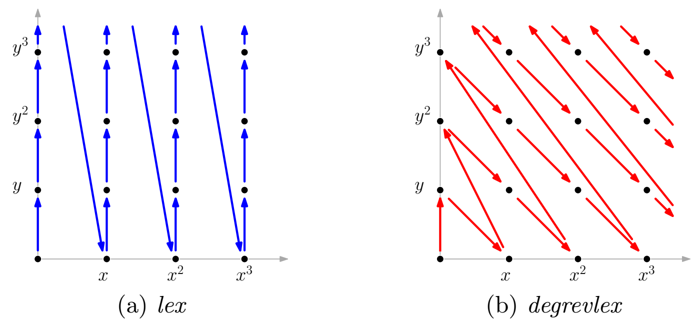


Fig. 1. Two monomial orders over $\mathbb{F}[x, y]$.

Definition 3 (Degree refining order). *A monomial order $\prec$ is* degree refining *if for any $\mathbf{x}^{\boldsymbol{\alpha}}, \mathbf{x}^{\boldsymbol{\beta}}$ with $\sum_i \alpha_i < \sum_i \beta_i$, we have $\mathbf{x}^{\boldsymbol{\alpha}} \prec \mathbf{x}^{\boldsymbol{\beta}}$.*

For example, both *deglex* and *degrevlex* are degree refining orders.

Weight orders The most general way to define a monomial order is using a list of *weight vectors* $\boldsymbol{\omega}_i \in \mathbb{R}^n$, which can be arranged into a *weight matrix* M_ω:

$$M_\omega = \begin{pmatrix} — \boldsymbol{\omega}_0 — \\ — \boldsymbol{\omega}_1 — \\ \vdots \\ — \boldsymbol{\omega}_m — \end{pmatrix}$$

Then, $\mathbf{x}^{\boldsymbol{\alpha}} \prec_{M_\omega} \mathbf{x}^{\boldsymbol{\beta}}$ if the first non-zero entry of vector $M_\omega \cdot \boldsymbol{\alpha} - M_\omega \cdot \boldsymbol{\beta}$ is negative. In other words, first the weight vector $\boldsymbol{\omega}_0$ is used to try and establish the relation of $\mathbf{x}^{\boldsymbol{\alpha}}$ and $\mathbf{x}^{\boldsymbol{\beta}}$, and in case of a tie, subsequent weight vectors are used for comparison.

Example 4. The matrices

$$M_{\text{lex}} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix}, \qquad M_{\text{degrevlex}} = \begin{pmatrix} 1 & \cdots & \cdots & 1 \\ 0 & \cdots & 0 & -1 \\ \vdots & \iddots & \iddots & 0 \\ \vdots & \iddots & \iddots & \vdots \\ 0 & -1 & 0 \cdots & 0 \end{pmatrix}$$

correspond to *lex* and *degrevlex* order, respectively. Note that a monomial order defined by M_ω where $\boldsymbol{\omega}_0 = c \cdot \mathbf{1},\ c \in \mathbb{R}\backslash\{0\}$, is a degree refining order.

Any monomial order can be expressed using weight vectors, but not all matrices define a monomial order. A sufficient condition for a matrix M_ω to define a monomial order is $M_\omega \in \mathbb{R}^{n\times n}$ with only non-negative entries and the rank of M_ω be full. The example $M_{\text{degrevlex}}$ above shows that this is not a necessary condition. Without further discussion thereof, $\ker(M) \cap \mathbb{Z}^n = \{0\}$ is such a necessary condition. An extended treatment of weight orders can be found in [37, Ch. 2, §4], [36, Ch. 1, §2], and [59, Sec. 1.2].

Leading term, coefficient, and monomial Fixing any monomial order $\prec$ gives rise to *leading*[4] *terms* of polynomials. For $f = \sum_{\boldsymbol{\alpha}} c_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$, the leading term $\mathsf{lt}(f)$ is defined as the $\prec$-maximal non-zero summand of f, i.e., $\mathsf{lt}(f) = \max_\prec\{c_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}} \mid c_{\boldsymbol{\alpha}} \neq 0\}$. The *leading coefficient* lc and *leading monomial* lm of f are coefficient and monomial of lt, respectively, i.e., $\mathsf{lt}(f) = \mathsf{lc}(f) \cdot \mathsf{lm}(f)$. All of f's terms smaller than $\mathsf{lt}(f)$, i.e., $f - \mathsf{lt}(f)$, comprise the *tail* of f. Extension of lt, lc, and lm to sets of polynomials are defined, e.g., $\mathsf{lt}(G) = \bigcup_{g\in G} \mathsf{lt}(g)$.

[4] Alternative terminology is *head* or *initial* term, coefficient, or monomial.

2.2 Definition of Gröbner Bases

A *Gröbner basis* of a polynomial Ideal $I \subseteq \mathbb{F}[\mathbf{x}]$ is a finite set of polynomials $G = \{g_0, \dots, g_{t-1}\}$ such that $\langle g_0, \dots, g_{t-1} \rangle = I$ and such that G has "nice" computational properties. For example, ideal membership $f \in I$ can be decided easily with a Gröbner basis; determining inclusion or equivalence of ideals is straightforward (a direct consequence); and projecting I into a ring with fewer variables is trivial for certain monomial orders. Perhaps most relevant for cryptanalysis is computing solutions of the underlying polynomial equations $g_i = 0$. This is straightforward for Gröbner bases in *lex* order, as we shall see in Section 2.3.

Leading up to the definitions of Gröbner basis, the concepts of polynomial reduction and (fully reduced) remainders are useful. In essence, reducing one polynomial by a (set of) polynomials is a generalization of long division for univariate polynomials. Results of such reductions are not generally unique. However, uniqueness of remainders after reduction is guaranteed if the set of divisors is a Gröbner basis – another "nice" property. In the following, let f be a polynomial, $G = \{g_0, \dots, g_{t-1}\}$ a finite set of polynomials, and $\prec$ a monomial order.

Definition 4 (Polynomial reduction). *We say G* reduces *f* to *r if there are polynomials $r, q_0, \dots, q_{t-1} \in \mathbb{F}[\mathbf{x}]$ such that $f = \sum_i q_i g_i + r$ and $\mathsf{lm}(r) \prec \mathsf{lm}(f)$. Polynomial r is called the* remainder *of the reduction of f by G.*

Note 1. Polynomials q_i can be efficiently computed using multivariate polynomial division. Such a division algorithm is reproduced in Appendix B.

Definition 5 (Fully reduced). *We say r is* fully reduced *with respect to G if no further reduction of r by G is possible. In other words, no leading monomial of any element in G divides any monomial of r: $\forall l \in \mathsf{lm}(G), \forall m \in \mathcal{M}(r) : l \nmid m$.*

We write $f \xrightarrow[G]{} r$ to denote full reduction *of f by G resulting in r, i.e., such that r is fully reduced with respect to G.*

Example 5. Fix monomial order *lex*. Let $f = x^2y^2 + y^2z^2 - 2y^2z$ and $G = \{x^2y - 2yz, y^2 - z^2, xz^2\}$. For $q_0 = y, q_1 = z^2, q_2 = 0$, we have $f = \sum_i q_i g_i + z^4$. Since no element of $\mathsf{lm}(G) = \{x^2y, y^2, xz^2\}$ divides z^4, the remainder z^4 is fully reduced with respect to G. We can write $f \xrightarrow[G]{} z^4$.

In general, the remainder r after full reduction of f is not unique. For example, choosing $q_0 = 0, q_1 = x^2 + z^2 - 2z, q_2 = x$ above, we have full reduction $f \xrightarrow[G]{} z^4 - 2z^3$. However, reducing a polynomial by a Gröbner basis guarantees uniqueness of the remainder. In fact, this is one way to define Gröbner bases.

Definition 6 (Gröbner basis by remainder). *G is a Gröbner basis for I if and only if $\langle G \rangle = I$ and the remainder after full reduction by G of any $f \in \mathbb{F}[\mathbf{x}]$ is unique.*

An equivalent albeit less intuitive definition pertains to the leading monomials of G and of elements of $\langle G \rangle = I$.

Definition 7 (Gröbner basis by leading monomials). *G is a Gröbner basis for I if and only if $\langle G\rangle = I$ and $\langle \mathsf{lm}(G)\rangle = \langle \mathsf{lm}(I)\rangle$.*

In other words, the ideal generated by the leading monomials of G is equal to the ideal generated by the leading monomials of all elements of I.

Theorem 1. *Definitions 6 and 7 are equivalent.*

Proof. "7 $\Rightarrow$ 6":

Fully reducing $f \in \mathbb{F}[\mathbf{x}]$ by G gives r such that $f = g + r$, $g = \sum_{i=0}^{t-1} q_i g_i$, and $\mathsf{lm}(r) \prec \min_i(\mathsf{lm}(g_i))$. Suppose there is another full reduction by G, i.e., $f = g' + r'$ for $g' = \sum_{i=0}^{t-1} q_i' g_i$, resulting in different remainder $r' \neq r$. Then, the difference of the remainders $r - r' = g' - g$ is an element of I. Consequently, the leading monomial of $r - r'$ is in $\langle \mathsf{lm}(I)\rangle$.

Since $\langle \mathsf{lm}(I)\rangle$ is generated by monomials, $\mathsf{lm}(r - r') \in \langle \mathsf{lm}(I)\rangle$ implies divisibility of $\mathsf{lm}(r - r')$ by one of the basis elements of $\langle \mathsf{lm}(I)\rangle$. Since $\langle \mathsf{lm}(I)\rangle = \langle \mathsf{lm}(g_0), \ldots, \mathsf{lm}(g_{t-1})\rangle$, this basis element is some $\mathsf{lm}(g_i)$, i.e., $\mathsf{lm}(g_i) \mid \mathsf{lm}(r - r')$ and thus $\mathsf{lm}(g_i) \preceq \mathsf{lm}(r - r')$ for some g_i. This contradicts $\mathsf{lm}(r - r') \preceq \max(\mathsf{lm}(r), \mathsf{lm}(r')) \prec min_i(\mathsf{lm}(g_i))$, proving $r = r'$ and thus uniqueness of the remainder.

"6 $\Rightarrow$ 7":

Since $\langle \mathsf{lm}(G)\rangle$ and $\langle \mathsf{lm}(I)\rangle$ are both generated by monomials, showing their equality in terms of polynomial membership can be reduced to equality in terms of monomial membership, i.e.,

$$\forall m \in \mathcal{M} : m \in \langle \mathsf{lm}(G)\rangle \Leftrightarrow m \in \langle \mathsf{lm}(I)\rangle. \tag{1}$$

To see that this reduction is valid, take polynomial $f = \sum_{\boldsymbol{\alpha}} c_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}} \in \mathbb{F}[\mathbf{x}]$. Then, f is in $\langle \mathsf{lm}(I)\rangle$ if and only if every $\mathbf{x}^{\boldsymbol{\alpha}}$ is in $\langle \mathsf{lm}(I)\rangle$. Likewise, f is in $\langle \mathsf{lm}(G)\rangle$ if and only if every $\mathbf{x}^{\boldsymbol{\alpha}}$ is in $\langle \mathsf{lm}(G)\rangle$. Therefore, it suffices to show that full reduction by G leaving unique remainders implies Equation (1).

Let f be a nonzero monomial, i.e., $\mathsf{lm}(f) = f$, and $f \xrightarrow[G]{} r$. Distinguish 3 cases:

1. $f \prec \mathsf{lm}(r)$.
   This case contradicts r being the result of a reduction.
2. $f = \mathsf{lm}(r)$.
   Then, $f = r \notin \langle G\rangle$ because no reduction took place. Suppose $f \in \langle \mathsf{lm}(G)\rangle$. Then $f = \sum q_i \mathsf{lm}(g_i)$ for some q_i, and since f is a monomial, only one term in this sum remains, meaning $f = q_i \mathsf{lm}(g_i)$ for some i. This contradicts no reduction haven taken place. Thus $f \notin \langle \mathsf{lm}(G)\rangle$. Suppose $f \in \langle \mathsf{lm}(\langle G\rangle)\rangle$ so that $f = \mathsf{lm}(\sum q_i g_i)$ for some q_i. By "undropping" trailing terms, $f = \sum q_i g_i - r'$ with $\mathsf{lm}(r') \prec f$. If $r' \neq 0$, it is a second remainder. If $r' = 0$ then $f \in \langle G\rangle$. Both are contradictions, so $f \notin \langle \mathsf{lm}(I)\rangle$.
3. $f \succ \mathsf{lm}(r)$.
   Then $f - r \in \langle G\rangle$ and $f = \mathsf{lm}(f) \in \langle \mathsf{lm}(I)\rangle$. Also $f = \sum q_i g_i + r$ so some $\mathsf{lm}(g_i) | f$ and $f \in \langle \mathsf{lm}(G)\rangle$.

Regardless of the case, $f \in \langle \mathsf{lm}(G) \rangle \Leftrightarrow f \in \langle \mathsf{lm}(I) \rangle$, completing the proof. □

Definition 8 (Reduced Gröbner basis). *G is a* reduced *Gröbner basis for I if and only if*

(1) G is a Gröbner basis for I,
(2) all $g \in G$ are monic, and
(3) all $g \in G$ have no monomial in $\langle G \setminus \{g\} \rangle$.

Proposition 1. *The reduced Gröbner basis for a non-trivial ideal with respect to a fixed monomial ordering exists and is unique.*

Proof. For a full proof, we refer to [37, Ch.2 §7 Thm. 5].

From neither Definition 6 nor 7 it is straightforward to even check whether a set of polynomials is a Gröbner basis, and they certainly do not readily suggest how to compute one. From Definition 7 it becomes clear that leading monomials are of importance. Rephrasing the definition, a set of polynomials G is a Gröbner basis if for any element f in the ideal, the leading monomial of f is divisible by the leading monomial of some element of G. This suggests that, at the very minimum, combining any two $g_i, g_j \in G$ in a way that their respective leading terms cancel should always result in a polynomial g_k whose leading monomial $\mathsf{lm}(g_k)$ is already present in G's set of leading monomials $\mathsf{lm}(G)$. This (roughly sketched) necessary condition turns out to be a sufficient one, as we will see in Theorem 2. The intuition of canceling leading terms is captured by S-polynomials.

Definition 9 (lcm of monomials). *Let $\mathbf{x}^{\boldsymbol{\alpha}}, \mathbf{x}^{\boldsymbol{\beta}} \in \mathcal{M}$ be monomials in n variables. Set $\boldsymbol{\gamma} = (\gamma_0, \ldots, \gamma_{n-1})$ with $\gamma_i = \max\{\alpha_i, \beta_i\}$. Define* $\operatorname{lcm}(\mathbf{x}^{\boldsymbol{\alpha}}, \mathbf{x}^{\boldsymbol{\beta}}) = \mathbf{x}^{\boldsymbol{\gamma}}$.

Definition 10 (S-Polynomial). *The* S-Polynomial of $f, g \in \mathbb{F}[\mathbf{x}]$ *is*

$$S(f,g) = \frac{u}{\mathsf{lt}(f)} f - \frac{u}{\mathsf{lt}(g)} g\,, \quad u = \operatorname{lcm}(\mathsf{lm}(f), \mathsf{lm}(g)).$$

Example 6. Let $\mathbb{F}[\mathbf{x}] = \mathbb{Q}[x, y, z]$, the monomial order be *deglex*, $f = 3x^2y + 6xyz$, and $g = 2yz + 2y$ Then,

$$\begin{aligned} S(f,g) &= \frac{x^2yz}{3x^2y} \cdot f - \frac{x^2yz}{2yz} \cdot g \\ &= \frac{z}{3} \cdot f - \frac{x^2}{2} \cdot g \\ &= 2xyz^2 - x^2y. \end{aligned}$$

Canceling the leading terms of two polynomials by way of computing their S-polynomial does not necessarily result in a smaller leading term: In Example 6, the leading monomials of $S(f,g)$ is different to the leading monomials of f and g, i.e., the leading terms of f and g *did* get canceled. However, the leading monomial of the S-polynomial is greater in the specified order, i.e., $\mathsf{lm}(S(f,g)) = xyz^2 \succ_{\text{deglex}} x^2y = \max\{\mathsf{lm}(f), \mathsf{lm}(g)\}$.

By construction, the S-polynomial of $f_i, f_j \in \mathcal{F}$ lies in the ideal $\langle \mathcal{F} \rangle$. Thus, if $\mathcal{F}$ is a Gröbner basis, full reduction of $S(f_i, f_j)$ for any f_i, f_j will leave the unique remainder 0, as per Definition 6. Conversely, if the same full reduction results in $r \neq 0$, then $\mathcal{F}$ is not a Gröbner basis. In fact, we just articulated what is known as *Buchberger's criterion* for testing if a set of polynomials is a Gröbner basis. It turns out that Gröbner bases can be defined in terms of this criterion. The resulting notion is equivalent to Definitions 6 and 7 but requires only a *finite* number of checks to determine whether a set of polynomials $\mathcal{F}$ is a Gröbner basis for $\langle \mathcal{F} \rangle$.

Theorem 2 (Buchberger's criterion). *G is a Gröbner basis for $\langle G \rangle$ if and only if $S(g_i, g_j) \xrightarrow[G]{} 0$ for all* critical pairs $(g_i, g_j) \in G \times G$.

Proof. For a full proof, we refer to [20,23,37].

Given that $S(f_i, f_j) \in \langle \mathcal{F} \rangle$, remainder r of $S(f_i, f_j)$ after full reduction by $\mathcal{F}$ is also an element of $\langle \mathcal{F} \rangle$. Thus, adding r to $\mathcal{F}$ does not change the ideal $\langle \mathcal{F} \rangle$, but intuitively makes $\mathcal{F}$ be "closer" to a Gröbner basis. This is the basic idea of *Buchberger's algorithm* [20,23], shown in Algorithm 1.

Intuitively, the algorithm's correctness follows directly from Buchberger's criterion. Showing termination is a little trickier. One essential ingredient is Hilbert's basis theorem [62], which implies that a Gröbner basis is always finite [37, Ch. 2, §5, Corr. 6]. Additionally, one needs to show that including a non-zero remainder of reducing an S-polynomial by a preliminary Gröbner basis brings us "closer" to a Gröbner basis. For proofs of correctness and termination, we refer to [20,23,37].

Algorithm 1: BUCHBERGER'S ALGORITHM

```
   Input: $\mathcal{F} = \{f_0, \ldots, f_{s-1}\} \subseteq \mathbb{F}[\mathbf{x}]$
   Output: a Gröbner basis $G$ for $\langle f_0, \ldots, f_{s-1} \rangle$
1  $G = \varnothing$
2  $G' = F$
3  while $G \neq G'$ do
4      $G = G'$
5      foreach $(g_i, g_j)$ in $G \times G$ do                  // critical pair
6          $r =$ (any) full reduction of $S(g_i, g_j)$ by $G$     // $S(g_i, g_j) \xrightarrow[G]{} r$
7          if $r \neq 0$ then
8              $G' = G' \cup \{r\}$
9  return $G$
```

The complexity of Algorithm 1 is quite difficult to estimate since many choices – the monomial order, the order of selecting pairs (g_i, g_j), and others –

can influence execution dramatically. A lot of work has been put into finding upper and lower complexity bounds for Gröbner basis computations, independent of the algorithm. The general measure of this complexity is by the highest total degree $d_{\max}$ among the polynomials in the Gröbner basis G for some ideal I, i.e., $d_{\max} = \max\{\deg(g) \mid g \in G\}$. Bounds exist for different input parameters, like the number s of input polynomials $\{f_0, \ldots, f_{s-1}\}$, their total degrees $d_i = \deg(f_i)$ with maximum $d = \max\{d_i\}$, the degrees' arithmetic mean D, or the dimension of the ideal $\dim(I)$. For the special case of underdetermined systems, i.e., $s \leqslant n$, $d_{\max}$ is not greater than the *Bézout bound*, defined as the product of all d_i, i.e., $d_{\max} \leqslant \prod_i d_i$ [69, Thm. 1]. More generally, $d_{\max}$ is doubly exponential in the number of variables, i.e., $d_{\max} \leqslant 2\left(\frac{1}{2}d^2 + d\right)^{2^{n-1}}$ [43], and this bound is tight in the worst-case [72]. A more recent bound is $d_{\max} \leqslant d^{n^{\Theta(1)} 2^{\Theta(\dim(I))}}$ [73], depending also on the dimension of I. For the special case of a zero-dimensional ideal, an even sharper bounds exists, with $d_{\max}$ polynomial in $\max\{S, D^n\}$, where S is the size of the input polynomials in dense representation and D the arithmetic mean $\frac{\sum d_i}{s}$ [61]. A lower bound is $d_{\max} \geqslant d^{2^{n/2}}$ for sufficiently large n and d [84].

While these results paint a bleak picture for the worst case, Gröbner bases for many polynomial systems arising from "real-world" scenarios *can* be computed. Algorithms for this problem have improved dramatically over the last decades. Some of the most important ones are surveyed in Section 3.

2.3 Computing $V(I)$ from G_{lex}

Equipped with the tools to compute Gröbner bases, we can come back to what we originally set out to do: breaking a cryptographic primitive by efficiently computing solutions to the system of polynomials $\mathcal{F}$ that model it, i.e., elements of the variety $V(\langle\mathcal{F}\rangle)$. Gröbner bases in *lex* order make it especially easy to achieve this. In this section, we provide intuition and an algorithm for the process of finding the variety's elements. While applicable in greater generality, for this paper we are only interested in the case of $\mathbb{F}$ being a finite field. For the theoretical background, we refer to [37, Ch. 3, §1], which is treating *elimination ideals*, the *elimination theorem*, and the *extension theorem*, and uses these tools to prove the algorithm's correctness.

One more piece of terminology regarding a system's number of solutions $|V(\mathcal{F})|$ frequently occurs in the context of Gröbner basis theory.

Definition 11 (Zero-dimensional ideal). *An ideal I is* zero-dimensional *if and only if its variety $V(I)$ has finitely many elements.*

Proposition 2. *Let $I \subseteq \mathbb{F}_q[\mathbf{x}]$ be some ideal and $\mathcal{E}_q^n = \langle x_i^q - x_i \mid 0 \leqslant i < n\rangle$ the ideal generated by all field equations. Then, $I \cup \mathcal{E}_q^n$ is zero-dimensional.*

Proof. Since $V(I \cup \mathcal{E}_q^n) = V(I) \cap \mathbb{F}_q^n$, we have $\dim(I \cup \mathcal{E}_q^n) = 0$. □

A reduced *lex* ordered Gröbner basis G_{lex} of a zero-dimensional ideal has exactly one polynomial g that is univariate in the *lex*-greatest variable x_{n-1} [36, Ch. 2, §3]. A root a_{n-1} of g can be the rightmost coordinate of an element $(a_0, \dots, a_{n-1}) \in V(I)$, and all such rightmost coordinates are roots of g. That is, the roots of g are a superset of the partial solutions to the system of equations defining I. Substituting variable x_{n-1} in G_{lex} by a_{n-1} corresponds to a projection of I into $\mathbb{F}[x_0, \dots, x_{n-2}]$, a polynomial ring with one variable less. This substitution $G_{\text{lex}}(x_{n-1} = a_{n-1})$ results in a new Gröbner basis, also in *lex* order, for $I \cap \mathbb{F}[x_0, \dots, x_{n-2}]$. Thus, above steps can be applied recursively until all variables have been eliminated. An element of $V(I)$ is then given by the univariate roots a_i used for projection at each level of recursion. Backtracking to iterate over all roots generates all of $V(I)$. A more formal description is given in Algorithm 2, and an example execution trace in Example 7 and Figure 2.

Algorithm 2: Variety from G_{lex}

```
   Input: Gröbner basis $G_{\text{lex}}$ of zero-dimensional ideal in lex order,
          number of variables $n$
   Output: Variety $V(\langle G_{\text{lex}} \rangle)$
1  if $n = 0$ then                    // only constants in $G_{\text{lex}}$: recursion ends
2      return $\{(\ )\}$                   // empty tuple, start of full solution
3  $V = \varnothing$
4  gcd_poly = gcd of all $g_i \in G$ univariate in $x_{n-1}$
5  partial_solutions = roots(gcd_poly)      // intersection of roots of the $g_i$
6  foreach root in partial_solutions do
7      $G_{\text{proj}} = G_{\text{lex}}(x_{n-1} = \text{root})$             // project into $\mathbb{F}[x_0, \dots, x_{n-2}]$
8      rest_of_solution = Variety from $G_{\text{lex}}(G_{\text{proj}}, n-1)$
9      $V = V \cup \{(s_0, \dots, s_{n-2}, \text{root}) \mid \boldsymbol{s} \in \text{rest\_of\_solution}\}$
10 return $V$
```

In Line 5 of Algorithm 2, univariate root finding is used as a black-box subroutine. Efficient algorithms for this well-studied problem are, for example, Berlekamp's algorithm [17], the Cantor-Zassenhaus algorithm [32], the Kaltofen-Shoup algorithm [67], and Kedlaya-Umans improvements thereof [68]. The Kedlaya-Umans algorithm is the fastest currently known algorithm for finding univariate roots. It has complexity of $d^{1.5} + d \log q$ operations in $\mathbb{F}_q$ for polynomials of degree d [54, p. 405]. An excellent introduction to the topic can be found in [54, Ch. 14].

In Line 4, parasitical solutions not lying in $\mathbb{F}_q$ but in algebraic closure $\overline{\mathbb{F}}_q$ can be discarded by also including field equation $x_{n-1}^q - x_{n-1}$ in the gcd.

Example 7. Let $\mathbb{F}[\mathbf{x}] = \mathbb{R}[x, y, z]$ and

$$f_0 = x - y, \quad f_1 = xyz, \quad f_2 = x^2 + y^2 + z^2 - 1,$$

the zeros of which respectively describe the "diagonal" plane $x = y$, the union of the three planes separating the octants, and the unit sphere. Set $I = \langle f_0, f_1, f_2 \rangle$.

The reduced Gröbner basis for I in *lex* order is given by $G_{\text{lex}} = \{g_0, g_1, g_2\}$ with

$$g_0 = x - y, \quad g_1 = y^2 - 0.5z^2 - 0.5, \quad g_2 = z^3 - z.$$

We have $g_0 = f_0$ describing the same plane as above. The zeroes of g_1 describe an elliptic cylinder along the x-axis, and the zeroes of g_2 are three planes coinciding with or orthogonal to the x-y-plane. Running Algorithm 2 on G_{lex} results in the execution trace given in Figure 2. Reading out the four solutions, we have $V(I) = \{\pm(0,0,1),\ \pm(1/\sqrt{2}, 1/\sqrt{2}, 0)\}$.

$$\begin{tikzcd}
 & \left\{\begin{matrix} y^2 \\ x-y \end{matrix}\right\} \arrow[r, "y=0"] & \{x\} \arrow[r, "x=0"] & \varnothing \\
 & & \left\{x+\frac{1}{\sqrt{2}}\right\} \arrow[r, "x=\frac{-1}{\sqrt{2}}"] & \varnothing \\
\left\{\begin{matrix} z^3-z \\ y^2+\frac{1}{2}z^2-\frac{1}{2} \\ x-y \end{matrix}\right\} \arrow[uur, "z=-1"] \arrow[r, "z=0"] \arrow[ddr, "z=1"] & \left\{\begin{matrix} y^2-\frac{1}{2} \\ x-y \end{matrix}\right\} \arrow[ur, "y=\frac{-1}{\sqrt{2}}"] \arrow[dr, "y=\frac{1}{\sqrt{2}}"] & & \\
 & & \left\{x-\frac{1}{\sqrt{2}}\right\} \arrow[r, "x=\frac{1}{\sqrt{2}}"] & \varnothing \\
 & \left\{\begin{matrix} y^2 \\ x-y \end{matrix}\right\} \arrow[r, "y=0"] & \{x\} \arrow[r, "x=0"] & \varnothing
\end{tikzcd}$$

Fig. 2. Example execution trace of Algorithm 2 for Example 7.

Note 2. Given a set of univariate polynomials $f_i(x)$, the Euclidean algorithm computes their polynomial gcd, containing exactly all shared factors. The roots of this gcd are those roots shared by all the f_i. In the multivariate case, the gcd of some $f_i(\mathbf{x})$ is not defined since $\mathbb{F}[x_0, \ldots, x_{n-1}]$ is not generally a Euclidean domain for $n > 2$. The variety V of the ideal spanned by the $f_i(\mathbf{x})$, efficiently computable given a Gröbner basis, contains all $\boldsymbol{a}$ for which all f_i evaluate to 0, the property of a shared root. Gröbner basis algorithms are thus a multivariate generalization of the Euclidean algorithm.

Note 3. The number of elements in a *lex* Gröbner basis G for ideal I generally tends to be (much) larger than the number of polynomials f_i originally defining I, even though Example 7 does not show this phenomenon.

Note 4. A Gröbner basis G_{lex} for ideal I in *lex* order contains a lot more information than strictly necessary for, say, multivariate polynomial division having unique remainders. In particular, G_{lex} allows to easily compute a Gröbner basis for the intersection $I \cap \mathbb{F}[x_0, \ldots, x_\ell]$ for $\ell < n$, simply by dropping all elements of G_{lex} containing variables greater than x_ℓ. Computing a Gröbner basis for such an intersection using a degree refining order, like *degrevlex*, is not straightforward. This intuitively – albeit not rigorously – explains why computing a Gröbner basis in *lex* order is generally much more difficult than computing a Gröbner basis in, say, *degrevlex* order [15].

3 Computing Gröbner Bases

In the previous section, we presented a generic way to compute a Gröbner basis, namely Buchberger's algorithm. Directly using this algorithm is usually too inefficient to be of any practical use. This is because Buchberger's algorithm performs a lot of unnecessary reductions, i.e., reductions where the remainder is zero. Improvements to Buchberger's algorithm exist, mostly in the form of criteria predicting when a critical pair reduces to zero. This allows the algorithm to discard such a pair before the costly reduction step is being performed [21,55,57,79].

The two Gröbner basis algorithms F_4 and F_5 have structural similarities to Buchberger's algorithm, but use a number of additional concepts. Since they constitute the state of the art, especially for attacking AOCs, we present them in detail in Sections 3.1 and 3.2. Furthermore, the algorithms *eXtended Linearization* (XL) and *Mutant XL*, both of which directly compute elements of an ideal's variety, are explained in Section 3.3. Other approaches, for example *Slimgb* [19], *MMM* [71], or *M4GB* [65], as well as concepts like *dynamic* Gröbner basis algorithms [28,30,80], are not discussed in this document.

3.1 F$_4$

Buchberger's algorithm considers every combination of two elements from the working Gröbner basis, computes this critical pair's S-polynomial, and reduces that S-polynomial by the working basis to check whether it reduces to zero. For every reduction, polynomial division by the working basis is started "from scratch," potentially performing the same computation more than once. For example, if we first reduce some f to 0, any polynomial multiple g of f will also reduce to 0 and we know this if g has been reduced to f. The Gröbner basis algorithm F$_4$ due to Faugère leverages such prior reductions by *batching*. More concretely, the coefficients of multiple S-Polynomials[5] are put into a specially crafted matrix, the echelon row-reduced form of which yields their fully reduced remainders. This results in an overall faster Gröbner basis algorithm because (1) multiple S-polynomials are being reduced at the same time, enabling the re-use of intermediary results, and (2) existing fast linear algebra techniques and implementations can be used. We motivate the inner workings of F$_4$ by introducing the *Macaulay matrix* of a set of polynomials, which is a more general concept than the matrices used in F$_4$.

[5] Actually of the "halves" comprising the S-Polynomials, see below.

Definition 12 (Macaulay matrix [70, Notation 3]). *Given a set of polynomials $\mathcal{F} \subseteq \mathbb{F}[\mathbf{x}]$ and monomial order $\prec$, the* Macaulay matrix $M_d(\mathcal{F})$ *of degree d is a matrix with coefficients in $\mathbb{F}$. Let $\mathbf{x}_d$ denote the $\prec$-greatest monomial of total degree d. The $\binom{n+d}{n}$ columns are labeled by all monomials $\mathbf{x}^{\boldsymbol{\alpha}}$ with $\mathbf{x}^{\boldsymbol{\alpha}} \preceq \mathbf{x}_d$ and sorted, from left to right, in $\prec$-descending order. The rows are labeled by all monomial multiples $\mathbf{x}^{\boldsymbol{\beta}} f_i$, $f_i \in \mathcal{F}$, such that $\mathsf{lt}(\mathbf{x}^{\boldsymbol{\beta}} f_i) \preceq \mathbf{x}_d$. Entry $m_{k,l}$ of $M_d(\mathcal{F})$ is the coefficient of monomial $\mathbf{x}^{\boldsymbol{\alpha}}$ in polynomial $\mathbf{x}^{\boldsymbol{\beta}} f_i$, where $\mathbf{x}^{\boldsymbol{\alpha}}$ is the label of column l and $\mathbf{x}^{\boldsymbol{\beta}} f_i$ is the label of row k.*

Example 8. For *degrevlex* order, the Macaulay matrix of degree 2 of bivariate polynomials $\mathcal{F} = \{f_0, f_1\} = \{x^2 + y^2, x + 2y\} \subseteq \mathbb{F}[x, y]$ is

$$M_2(\mathcal{F}) = \begin{array}{c} \\ f_0 \\ f_1 \\ yf_1 \\ xf_1 \end{array} \!\!\begin{array}{c} \begin{array}{cccccc} x^2 & xy & y^2 & x & y & 1 \end{array} \\ \left(\begin{array}{cccccc} 1 & & 1 & & & \\ & & & 1 & 2 & \\ & 1 & 2 & & & \\ 1 & 2 & & & & \end{array}\right) \end{array} .$$

Performing a row operation on a Macaulay matrix such that a non-zero coefficient gets canceled, corresponds to one step in the polynomial reduction algorithm. Thus, computing the row-reduced echelon form of a Macaulay matrix corresponds to full polynomial reduction of all polynomials *by* all polynomials. If the Macaulay matrix $M_d(\mathcal{F})$ is of high enough degree d, its row-reduced echelon form gives rise to a Gröbner basis[6] for $\mathcal{F}$ [69, Sec. III B]. However, the combination of two problems make this approach intractable in practice: (1) Not knowing beforehand which degree d is high enough, and (2) the roughly cubic complexity of echelon row-reducing a matrix.

For these reasons, the F$_4$ algorithm does not reduce the Macaulay matrix. Instead, a matrix M containing *just* enough information for full polynomial reduction of a subset of the critical pairs is constructed at each step. Intuitively, M is a less verbose version of a Macaulay matrix, where rows and columns not contributing to the polynomial reduction are not included. This omission decreases computation time and memory requirements. The relevant rows are those corresponding to polynomials that *might* at some point during the polynomial division process cancel *some* monomial. Also, M does not contain any zero-columns, like the column with label "1" in Example 8. A formal description of which polynomials are included is given in subroutine "Symbolic Preprocessing" in Algorithm 3.

The matrix M and the insight that computing its row-reduced echelon form corresponds to full polynomial reduction of all its polynomials are the reasons F$_4$ is so much more efficient than Buchberger's algorithm. The question of which polynomials to include in M gives rise to the *selection strategy*. Many selection strategies exist [46, Sec. 2.5], really making F$_4$ a *family* of algorithms. The fastest strategy in practice, thus dubbed the *normal* strategy, is to pick those pairs $\{i, j\}$

[6] This neatly illustrates how Gröbner bases generalize solving a system of linear equations to solving a system of polynomial equations.

Algorithm 3: SYMBOLIC PREPROCESSING

```
Input: S-Polynomial “halves” $L \subseteq \mathbb{F}[\mathbf{x}]$, working Gröbner basis $G$
Output: $M$
1 processed_monoms = $\mathsf{lm}(L)$
2 while processed_monoms $\neq \mathcal{M}(L)$ do
3     $\mathbf{x}^{\boldsymbol{\beta}} = \max_{\prec}\{\mathcal{M}(L) \setminus$ processed_monoms$\}$
4     processed_monoms = processed_monoms $\cup\{\mathbf{x}^{\boldsymbol{\beta}}\}$
5     if $\exists g \in G$ such that $\mathsf{lm}(g) \mid \mathbf{x}^{\boldsymbol{\beta}}$ then
6         $L = L \cup \left\{\frac{\mathbf{x}^{\boldsymbol{\beta}}}{\mathsf{lm}(g)} \cdot g\right\}$
7 $M$ = matrix with columns labeled by $\mathcal{M}(L)$ in $\prec$-decreasing order, rows are
    respective coefficients of elements of $L$
8 return $M$
```

with lowest total degree of $\operatorname{lcm}(\mathsf{lm}(f_i), \mathsf{lm}(f_j))$. Intuitively, this strategy keeps the degree of the working polynomials as small as possible at all times.

After M is computed and echelon row-reduced, all those polynomials with leading monomial not already in the preliminary Gröbner basis G are added to it. The list of critical pairs is then updated using these new elements. Thus, Buchberger's criterion (Theorem 2) works as the termination criterion once again. The complete algorithm is given in Algorithm 4. For a proof of its correctness and termination, we refer to [46, Thm. 2.2] or [37, Ch. 10, §3].

Algorithm 4: F_4

```
Input: $\mathcal{F} = \{f_0, \ldots, f_{s-1}\} \subseteq \mathbb{F}[\mathbf{x}]$
Output: a Gröbner basis $G$ for $\langle f_0, \ldots, f_{s-1}\rangle$
1  $G = \mathcal{F}$
2  $t = s$
3  $B = \{\{i, j\} \mid 0 \leqslant i < j < s\}$ // indices of crit. pairs not considered yet
4  while $B \neq \varnothing$ do
5      Select non-empty $B' \subseteq B$        // e.g., according to normal strategy
6      $B = B \setminus B'$
7      $L = \left\{\frac{\operatorname{lcm}(\mathsf{lm}(f_i),\mathsf{lm}(f_j))}{\mathsf{lt}(f_i)} \cdot f_i \,\middle|\, \{i, j\} \in B'\right\}$        // S-polynomial "halves"
8      $M$ = SYMBOLIC PREPROCESSING$(L, G)$
9      $N$ = row-reduced echelon form of $M$
10     $N^+ = \{n \in \operatorname{rows}(N) \mid \mathsf{lm}(n) \notin \langle \mathsf{lm}(\operatorname{rows}(M))\rangle\}$ // only new lead monoms
11     foreach $r$ in rows$(N^+)$ do
12         $f_t$ = polynomial form of $r$
13         $G = G \cup \{f_t\}$
14         $B = B \cup \{\{i, t\} \mid 0 \leqslant i < t\}$
15         $t = t + 1$
16 return $G$
```

F_4 as given in Algorithm 4 can be heavily improved upon. For example, results of subroutine "Symbolic Preprocessing" can be reused between iterations. Furthermore, there are a number of criteria to determine whether the S-polynomial of a critical pair will reduce to zero, including acclaimed Gebauer-Möller installation [21, 22, 29, 55, 63]. Originally applied to Buchberger's algorithm, they can be made use of in F_4 as well. Apart from discarding unnecessary critical pairs, the order in which pairs are selected plays a crucial role for computation speed. For an in-depth review of such improvements, we refer to [46, Sec. 2.4] and [37, Ch. 10, §3].

3.2 F_5

Both Buchberger's algorithm and F_4 compute, in some form, the S-polynomial of a critical pair and reduce that S-polynomial by the current preliminary Gröbner basis. Any nonzero remainder is added to the working Gröbner basis. However, a reduction to zero does not produce new information, but does waste time. In practice, roughly 90% of F_4's execution time is spent performing such useless computations[7] [47]. Establishing ahead of time whether a reduction is necessary thus greatly improves overall performance.

Using a combination of *syzygies*, which are kernel elements of map $\phi_{\mathcal{F}}(\boldsymbol{\omega}) = \sum_i \omega_i f_i$ for input set $\mathcal{F}$, and *signatures*, the F_5 criterion allows the F_5 algorithm to skip a lot of useless reductions. In fact, if the input system is a *regular sequence*, no useless reduction is performed at all.

Definition 13 (Regular sequence of polynomials [14, Def. 2]). *A list of polynomials $f_0, \ldots, f_{s-1} \in \mathbb{F}[\mathbf{x}]$ is a* regular sequence *if, for all $0 \leqslant i < s$, polynomial f_i^h is not a zero-divisor in quotient ring*

$$\mathbb{F}[\mathbf{x}] \big/ \langle f_0^h, \ldots, f_{i-1}^h \rangle \,,$$

where f_i^h is the homogeneous part of f_i of highest degree.

Intuitively, a regular sequence is a generalization of linear independence, where no f_i can be expressed through a polynomially weighted sum $\sum_{j=0}^{i-1} q_j f_j$, i.e., f_i does not lie in the ideal spanned by the f_j with $j < i$.

Some further concepts are necessary before the F_5 algorithm can be properly introduced. For this, we follow [37, Ch. 10, §4] as opposed to the original publication [47], allowing to present F_5 in a way structurally similar to both Buchberger's algorithm and F_4. If the set of generator polynomials $\mathcal{F}$ for ideal I is clear from context, we set $\phi = \phi_{\mathcal{F}}$.

Any element g of the ideal defined by $\langle \mathcal{F} \rangle$, including elements of a Gröbner basis for I, is a polynomially weighted sum of the generator polynomials $f_i \in \mathcal{F}$. That is, $g = \sum_i \omega_i f_i$ for some $\boldsymbol{\omega} \in \mathbb{F}[\mathbf{x}]^s$, defining the function $\phi_{\mathcal{F}}(\boldsymbol{\omega}) = \sum_i \omega_i f_i$.

[7] The parameters this fraction depends on are not clear, but the sheer magnitude already motivates looking for improvements.

This sum is implicitly computed in both Buchberger's algorithm and F_4, since the main interest lies with the resulting polynomial g. F_5 computes and uses these weights explicitly. These *vectors of origin*[8] succinctly keep track of how the algorithm arrived at, for example, a Gröbner basis element, and enable the algorithm to avoid redundant deductions. A vector of origin $\boldsymbol{\omega}$ is an element of the s-dimensional free module on $\mathbb{F}[\mathbf{x}]$, i.e., $\boldsymbol{\omega} \in \mathbb{F}[\mathbf{x}]^s$. Observe that the image of module $\mathbb{F}[\mathbf{x}]^s$ under $\phi_{\mathcal{F}}$ is exactly the ideal spanned by $\mathcal{F}$, i.e., $\phi_{\mathcal{F}}(\mathbb{F}[\mathbf{x}]^s) = \langle \mathcal{F} \rangle$.

Monomial order $\prec$ can be extended to order $\preccurlyeq$ on vectors of origin. The most common extension[9] is *position-over-term* ordering $\preccurlyeq_{\text{pot}}$, where the indices of the vectors' highest-index non-zero entry are compared first, with ties broken by comparing the entries leading monomials according to $\prec$.

Definition 14 ($\preccurlyeq_{\text{pot}}$). *For order $\preccurlyeq$, extending $\prec$ to vectors of polynomials, we have $\boldsymbol{g} \preccurlyeq \boldsymbol{h}$ if there exists index i such that for all $j > i$, we have $g_j = h_j = 0$ and either $g_i = 0 \neq h_i$ or*[10] *$\mathsf{lm}(g_i) \prec \mathsf{lm}(h_i)$.*

Example 9. Extending $\prec_{\text{lex}}$ to $\preccurlyeq_{\text{pot}}$, we have the following ordering.

$$\begin{pmatrix} x^2 \\ 0 \\ 0 \end{pmatrix} \preccurlyeq \begin{pmatrix} x \\ x+y^4 \\ 0 \end{pmatrix} \preccurlyeq \begin{pmatrix} y \\ x^2 \\ 0 \end{pmatrix} \equiv \begin{pmatrix} x^5y^4+1 \\ -3x^2+xy^3 \\ 0 \end{pmatrix} \preccurlyeq \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

Definition 15 (Signature). *The* signature $\mathfrak{s}$ *of vector of polynomials $\boldsymbol{g}$ is $\mathfrak{s}(\boldsymbol{g}) = \mathsf{lm}(g_i)\boldsymbol{e}_i$, where the highest-index non-zero entry g_i of $\boldsymbol{g}$ is at position i.*

Example 10 (continued). The signatures of the vectors used in Example 9 are as follows.

$$\begin{pmatrix} x^2 \\ 0 \\ 0 \end{pmatrix} \preccurlyeq \begin{pmatrix} 0 \\ x \\ 0 \end{pmatrix} \preccurlyeq \begin{pmatrix} 0 \\ x^2 \\ 0 \end{pmatrix} \equiv \begin{pmatrix} 0 \\ x^2 \\ 0 \end{pmatrix} \preccurlyeq \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

Intuitively, the signature is a vector's minimal amount of information needed for correct sorting according to $\preccurlyeq$.

The F_5 criterion considers only signatures of vectors of origin, not the full vectors or their evaluation under ϕ. A good part of F_5's efficiency can be attributed to the fact that operations on and comparisons of signatures can be performed very efficiently. Since signatures are the product of some monomial and a unit vector, divisibility of monomials can be extended to signatures in a straightforward manner.

[8] We introduce this terminology to reduce confusion with *signature* vectors.

[9] Another common option is *term-over-position* ordering $\preccurlyeq_{\text{top}}$, where the leading monomials of the highest-indexed non-zero entries are compared first, using their indices as tie breaker. We refer to [36, Ch. 5, §2] for a more detailed discussion. For the rest of this paper, we set $\preccurlyeq$ to $\preccurlyeq_{\text{pot}}$ if not stated differently.

[10] Both [36] and the original publication of F_5 [47] use $g_i \neq 0 = h_i$ as the first condition. In this paper, we use the more intuitive definition of [37, Ch. 10, §4].

Definition 16 (Signature divisibility). *For vectors of origin $\boldsymbol{g}$ and $\boldsymbol{h}$, we say* $\mathfrak{s}(\boldsymbol{g}) = \mathbf{x}^{\boldsymbol{\alpha}}\boldsymbol{e}_i$ divides $\mathfrak{s}(\boldsymbol{h}) = \mathbf{x}^{\boldsymbol{\beta}}\boldsymbol{e}_j$, *and write* $\mathfrak{s}(\boldsymbol{g}) \mid \mathfrak{s}(\boldsymbol{h})$, *if* $i = j$ *and* $\mathbf{x}^{\boldsymbol{\alpha}} \mid \mathbf{x}^{\boldsymbol{\beta}}$.

Vectors of origin that are kernel elements of ϕ are called *syzygy vectors* and are of special interest for F_5.

Definition 17 (Syzygy vector). *A vector of origin $\boldsymbol{s} \in \mathbb{F}[\mathbf{x}]^s$ is called a* syzygy vector *if it maps to* 0 *under ϕ, i.e., $\phi(\boldsymbol{s}) = 0$.*

It is easy to see that the set of syzygy vectors is a submodule of $\mathbb{F}[\mathbf{x}]^s$. Since the zero polynomial does not contribute any information to the Gröbner basis computation, it is helpful to identify syzygy vectors early. For ideal $I = \langle f_0, \dots, f_{s-1} \rangle$, some syzygy vectors are trivial to state.

Definition 18 (Koszul syzygy). *For any $0 \leqslant i < j < s$, the vector $f_i\boldsymbol{e}_j - f_j\boldsymbol{e}_i$ is a syzygy vector. It is called a* Koszul syzygy, *or* principal syzygy.

If $\mathcal{F}$ is a regular sequence, the set of Koszul syzygies is a basis for the entire submodule of syzygies [44, Corr. 7.1]

F_5 performs an analog to polynomial reduction by a working Gröbner basis, but on vectors of origin and with an "early abort." More concretely, when reducing $\phi(\boldsymbol{f})$ by a set of polynomials $\phi(\boldsymbol{G}) = \{\phi(\boldsymbol{g}_i) \mid \boldsymbol{g}_i \in \boldsymbol{G}\}$, a step of the polynomial division algorithm is only performed if the signature $\mathfrak{s}(\boldsymbol{f})$ does not change with regards to $\preccurlyeq$.

Definition 19 ($\mathfrak{s}$-reduction). *We say that $\boldsymbol{G} \subseteq \mathbb{F}[\mathbf{x}]^s$* $\mathfrak{s}$-reduces *$\boldsymbol{f} \in \mathbb{F}[\mathbf{x}]^s$ to $\boldsymbol{h} \in \mathbb{F}[\mathbf{x}]^s$ if there exist $\boldsymbol{g}_i \in \boldsymbol{G}$, $\mathbf{x}^{\boldsymbol{\alpha}} \in \mathbb{F}[\mathbf{x}]$, and $c \in \mathbb{F}$, such that*

(1) $\boldsymbol{h} = \boldsymbol{f} - c\mathbf{x}^{\boldsymbol{\alpha}}\boldsymbol{g}_i$
(2) $\mathsf{lt}(\phi(c\mathbf{x}^{\boldsymbol{\alpha}}\boldsymbol{g}_i))$ *is a term in $\phi(\boldsymbol{f})$, and either*
(3a) $\mathfrak{s}(\boldsymbol{f}) \succcurlyeq \mathfrak{s}(\mathbf{x}^{\boldsymbol{\alpha}}\boldsymbol{g}_i)$ *or*
(3b) $\mathfrak{s}(\boldsymbol{f}) = \mathfrak{s}(\mathbf{x}^{\boldsymbol{\alpha}}\boldsymbol{g}_i)$.

If Condition (3a) holds, the $\mathfrak{s}$-reduction is regular. *Otherwise, if Condition (3b) holds, the $\mathfrak{s}$-reduction is* singular. *Vector $\boldsymbol{f}$ is* fully $\mathfrak{s}$-reduced *with respect to $\boldsymbol{G}$ if no $\mathfrak{s}$-reduction of $\boldsymbol{f}$ by $\boldsymbol{G}$ is possible.*

Observe that Condition (2) corresponds to one step in the multivariate polynomial division algorithm. The Conditions (3a) and (3b) define the "early abort." Full regular $\mathfrak{s}$-reduction means that $\phi(\boldsymbol{f})$ is reduced as much as possible without changing signature $\mathfrak{s}(\boldsymbol{f})$. A corresponding algorithm can be found in Appendix C.

Definition 20 ($\mathfrak{s}$-reduction to zero). *We say that $\boldsymbol{f}$* $\mathfrak{s}$-reduces to zero by $\boldsymbol{G}$ *if there exists some syzygy vector $\boldsymbol{s}$ such that $\boldsymbol{f}$ can be $\mathfrak{s}$-reduced to $\boldsymbol{s}$ by $\boldsymbol{G}$.*

Note that $\boldsymbol{f}$ $\mathfrak{s}$-reducing to zero by $\boldsymbol{G}$ implies $\phi(\boldsymbol{f}) \xrightarrow[\phi(\boldsymbol{G})]{} 0$, where $\phi(\boldsymbol{G}) = \{\phi(\boldsymbol{g}_i) \mid \boldsymbol{g}_i \in \boldsymbol{G}\}$, but not necessarily that full $\mathfrak{s}$-reduction of $\boldsymbol{f}$ by $\boldsymbol{G}$ is $\mathbf{0} \in \mathbb{F}[\mathbf{x}]^s$ or $\phi(\boldsymbol{f}) = 0 \in \mathbb{F}[\mathbf{x}]$.

Definition 21 (Signature Gröbner basis). *A set of vectors of origin $\boldsymbol{G} = \{\boldsymbol{g}_0, \dots, \boldsymbol{g}_{t-1}\} \subseteq \mathbb{F}[\mathbf{x}]^s$ with monic $\phi(\boldsymbol{g}_i)$ is called* signature Gröbner basis *for ideal I generated by $\mathcal{F} = \{f_0, \dots, f_{s-1}\}$ if all vectors of origin $\boldsymbol{f} \in \mathbb{F}[\mathbf{x}]^s$ $\mathfrak{s}$-reduce to zero by $\boldsymbol{G}$.*

Similarly, $\boldsymbol{G}$ is a signature Gröbner basis below signature $\mathbf{x}^{\boldsymbol{\alpha}}\boldsymbol{e}_i$ *if all vectors of origin $\boldsymbol{f} \in \mathbb{F}[\mathbf{x}]^s$ with signature $\mathfrak{s}(\boldsymbol{f}) \preccurlyeq \mathbf{x}^{\boldsymbol{\alpha}}\boldsymbol{e}_i$ $\mathfrak{s}$-reduce to zero by $\boldsymbol{G}$.*

Observe that by the definition of ϕ, we have $\phi(\mathbb{F}[\mathbf{x}]^s) = \langle\mathcal{F}\rangle = I$, intuiting the requirement that *all* vectors of origin $\boldsymbol{f} \in \mathbb{F}[\mathbf{x}]^s$ need to reduce to zero by $\boldsymbol{G}$. Like the original Definitions 6 and 7 for Gröbner bases, Definition 21 is not practical to compute a signature Gröbner basis. This is remedied by generalizing Buchberger's criterion to vectors of origin. Thus, Buchberger's criterion is used again to ensure correctness. The central concept of Buchberger's criterion are S-polynomials, and their generalization to vectors of origin leads to *S-vectors*.

Definition 22 (S-vector). *The* S-vector *of $\boldsymbol{f}$ and $\boldsymbol{g}$ with monic $\phi(\boldsymbol{f})$, $\phi(\boldsymbol{g})$ is*

$$S(\boldsymbol{f}, \boldsymbol{g}) = \frac{u}{\mathsf{lm}(\phi(\boldsymbol{f}))}\boldsymbol{f} - \frac{u}{\mathsf{lm}(\phi(\boldsymbol{g}))}\boldsymbol{g}, \quad u = \operatorname{lcm}(\mathsf{lm}(\phi(\boldsymbol{f}), \phi(\boldsymbol{g}))) \ .$$

If $\mathfrak{s}(\boldsymbol{f}) \neq \mathfrak{s}(\boldsymbol{g})$, the S-vector $S(\boldsymbol{f}, \boldsymbol{g})$ is called regular*. Otherwise, the S-vector is called* singular*.*

Observe that ϕ is a homomorphism from the set of S-vectors of $\mathbb{F}[\mathbf{x}]^s$ to the set of S-polynomials of $\mathbb{F}[\mathbf{x}]$:

$$\phi(S(\boldsymbol{f}, \boldsymbol{g})) = S(\phi(\boldsymbol{f}), \phi(\boldsymbol{g})). \tag{2}$$

Theorem 3 (Vectorized Buchberger's criterion). *Let $\boldsymbol{G} = \{\boldsymbol{g}_0, \dots, \boldsymbol{g}_{t-1}\}$, $\boldsymbol{g}_i \in \mathbb{F}[\mathbf{x}]^s$ with monic $\phi(\boldsymbol{g}_i)$, and $I = \langle f_0, \dots, f_{s-1}\rangle$ an ideal in $\mathbb{F}[\mathbf{x}]$. Then, $\boldsymbol{G}$ is a signature Gröbner basis for I if and only if*

(1) S-vector $S(\boldsymbol{g}_i, \boldsymbol{g}_j)$ $\mathfrak{s}$-reduces to zero by $\boldsymbol{G}$ for any $0 \leqslant i, j < t$, and
(2) unit vector $\boldsymbol{e}_i$ $\mathfrak{s}$-reduces to zero by $\boldsymbol{G}$ for any $0 \leqslant i < s$.

Similarly, $\boldsymbol{G}$ is a signature Gröbner basis below $\mathbf{x}^{\boldsymbol{\alpha}}\boldsymbol{e}_i$ for I if and only if above conditions hold for all S-vectors and unit vectors with signature $\preccurlyeq \mathbf{x}^{\boldsymbol{\alpha}}\boldsymbol{e}_i$.

Proof (idea). Using the homomorphic property of ϕ from Equation (2), Theorem 3 can be reduced to the original Buchberger criterion of Theorem 2.

Note that condition (2) ensures that $\boldsymbol{G}$ is a signature Gröbner basis for I, not for some ideal $J \subsetneq I$ strictly contained in I. Signature Gröbner bases imply *actual* Gröbner bases in the following manner.

Proposition 3. *Let $\boldsymbol{G} = \{\boldsymbol{g}_0, \dots, \boldsymbol{g}_{t-1}\}$ be a signature Gröbner basis for $I = \langle\mathcal{F}\rangle$. Then, $\phi(\boldsymbol{G}) = \{\phi(\boldsymbol{g}_i) \mid \boldsymbol{g}_i \in \boldsymbol{G}\}$ is a Gröbner basis for I.*

Proof. By the definition of signature Gröbner basis, all elements of $\mathbb{F}[\mathbf{x}]^s$ $\mathfrak{s}$-reduce to zero by $\boldsymbol{G}$. This includes all S-vectors $S(\boldsymbol{g}_i, \boldsymbol{g}_j)$ for any $\boldsymbol{g}_i, \boldsymbol{g}_j \in \boldsymbol{G}$. Since Equation (2) holds, we have $S(\phi(\boldsymbol{g}_i), \phi(\boldsymbol{g}_j)) \xrightarrow[\phi(\boldsymbol{G})]{} 0$. Thus, Buchberger's criterion applies to $\phi(\boldsymbol{G})$. □

Above definitions give the tools to state the two propositions comprising the F_5 criterion. The first proposition discards vectors of origin whose corresponding polynomial are a strict multiple of an already identified Gröbner basis element.

Proposition 4. *Let $\boldsymbol{g}$ and $\boldsymbol{h}$ be vectors of origin with $\mathfrak{s}(\boldsymbol{g}) = \mathfrak{s}(\boldsymbol{h})$. Let $\boldsymbol{G}$ be a signature Gröbner basis below this signature. If $\boldsymbol{g}$ and $\boldsymbol{h}$ are fully regular $\mathfrak{s}$-reduced by $\boldsymbol{G}$, then $\phi(\boldsymbol{g}) = \phi(\boldsymbol{h})$.*

Proof. Assume that $\phi(\boldsymbol{g}) \neq \phi(\boldsymbol{h})$. Then, without loss of generality, $\mathbf{0} \neq \mathfrak{s}(\boldsymbol{g} - \boldsymbol{h}) \preccurlyeq \mathfrak{s}(\boldsymbol{g})$. Thus, $\boldsymbol{g} - \boldsymbol{h}$ $\mathfrak{s}$-reduces to zero by $\boldsymbol{G}$. Let $\boldsymbol{m}$ be the $\preccurlyeq$-leading term of $\boldsymbol{g} - \boldsymbol{h}$. Without loss of generality, $\boldsymbol{m}$ appears in $\boldsymbol{g}$. This implies that $\boldsymbol{g}$ is not regular $\mathfrak{s}$-reduced by $\boldsymbol{G}$, contradicting the assumption. □

The second proposition in the F_5 criterion uses syzygy vectors to detect unnecessary reductions ahead of time. In an earlier attempt to leverage syzygies for faster Gröbner basis computation, the overhead of computing a basis for the syzygy submodule outweighed the resulting savings [79]. F_5 improves on this by building the syzygy submodule basis and the signature Gröbner basis at the same time.

Proposition 5. *Let $\boldsymbol{G} = \{\boldsymbol{g}_0, \ldots, \boldsymbol{g}_{t-1}\}$ be a signature Gröbner basis below signature $\mathbf{x}^{\boldsymbol{\alpha}}\boldsymbol{e}_i$ for ideal I. Let $\boldsymbol{h} = S(\boldsymbol{g}_i, \boldsymbol{g}_j)$ with signature $\mathfrak{s}(\boldsymbol{h})$ at most $\mathbf{x}^{\boldsymbol{\alpha}}\boldsymbol{e}_i$. If there exists syzygy vector $\boldsymbol{s}$ such that $\mathfrak{s}(\boldsymbol{s}) \mid \mathfrak{s}(\boldsymbol{h})$, then $\boldsymbol{h}$ $\mathfrak{s}$-reduces to zero by $\boldsymbol{G}$.*

Proof. By the definition of signature division, there exists $\mathbf{x}^{\boldsymbol{\gamma}}$ such that $\mathfrak{s}(\mathbf{x}^{\boldsymbol{\gamma}}\boldsymbol{s}) = \mathfrak{s}(\boldsymbol{h})$. Thus, $\boldsymbol{h}' = \boldsymbol{h} - \mathbf{x}^{\boldsymbol{\gamma}}\boldsymbol{s} \preccurlyeq \mathbf{x}^{\boldsymbol{\alpha}}\boldsymbol{e}_i$. Consequently, $\boldsymbol{h}'$ $\mathfrak{s}$-reduces to zero by $\boldsymbol{G}$, i.e., to some syzygy vector $\boldsymbol{s}'$. Thus, $\boldsymbol{h}$ $\mathfrak{s}$-reduces to $\boldsymbol{s}''$ by $\boldsymbol{G}$, where $\boldsymbol{s}''$ is the $\mathfrak{s}$-reduction of $\boldsymbol{s}' + \mathbf{x}^{\boldsymbol{\gamma}}\boldsymbol{s}$ by $\boldsymbol{G}$. Since the set of syzygies form a module, $\boldsymbol{s}''$ is a syzygy vector. It follows that $\boldsymbol{h}$ $\mathfrak{s}$-reduces to zero by $\boldsymbol{G}$. □

Definition 23 (F_5 criterion). *For signature Gröbner basis $\boldsymbol{G}$ for $I = \langle \mathcal{F} \rangle$ below signature $\mathbf{x}^{\boldsymbol{\alpha}}\boldsymbol{e}_i$, a set of syzygy vectors $\boldsymbol{S}$, and vector of origin $\boldsymbol{f}$ fully regular $\mathfrak{s}$-reduced by $\boldsymbol{G}$, the* F_5 criterion *is true if and only if*

(1) there is no $\boldsymbol{g}_i \in \boldsymbol{G}$ with $\mathfrak{s}(\boldsymbol{g}_i) = \mathfrak{s}(\boldsymbol{f})$, and
(2) there is no $\boldsymbol{s} \in \boldsymbol{S}$ with $\mathfrak{s}(\boldsymbol{s}) \mid \mathfrak{s}(\boldsymbol{f})$.

It is possible to combine the criterion's two checks using *rewriting*, for which we refer to [45]. With the F_5 criterion at its heart, F_5 is given in Algorithm 5. We present F_5 more in line with later improvements and generalizations, like RB [44, Alg. 3] and GVW [52, 53], highlighting the structural similarities to Buchberger's algorithm and F_4. F_5 is in fact a family of algorithms, since several selection strategies for Line 5 exist. Selecting the $\preccurlyeq$-smallest vector, i.e., the vector with smallest signature, is the most efficient strategy in practice. This way, F_5 first computes a Gröbner basis for ideal $\langle f_0 \rangle$, then for $\langle f_0, f_1 \rangle$, then $\langle f_0, f_1, f_2 \rangle$, and so on.

The F_4 and F_5 families of algorithms are not disjoint. Concretely, the selection strategy in Line 5 can be altered such that multiple vectors of origin are

Algorithm 5: F_5

```
Input: $\mathcal{F} = \{f_0, \ldots, f_{s-1}\} \subseteq \mathbb{F}[\mathbf{x}]$
Output: a Gröbner basis $G$ for $\langle f_0, \ldots, f_{s-1} \rangle$
$\boldsymbol{G} = \varnothing$   // preliminary signature Gröbner basis below some signature
$\boldsymbol{P} = \{\boldsymbol{e}_0, \ldots, \boldsymbol{e}_{s-1}\}$   // ensures $\boldsymbol{G}$ is not basis for strict sub-ideal
$\boldsymbol{S} = \{f_i\boldsymbol{e}_j - f_j\boldsymbol{e}_i \mid 0 \leqslant i < j < s\}$   // Koszul syzygies
while $\boldsymbol{P} \neq \varnothing$ do
    $\boldsymbol{g}$ = some element of $\boldsymbol{P}$   // e.g., $\preccurlyeq$-smallest
    $\boldsymbol{P} = \boldsymbol{P} \setminus \{\boldsymbol{g}\}$
    if F5-Criterion($\boldsymbol{g}, \boldsymbol{G}, \boldsymbol{S}$) then
        $\boldsymbol{h}$ = regular $\mathfrak{s}$-reduction of $\boldsymbol{g}$ by $\boldsymbol{G}$
        if $\phi_{\mathcal{F}}(\boldsymbol{h}) = 0$ then
            $\boldsymbol{S} = \boldsymbol{S} \cup \{\boldsymbol{h}\}$
        else
            $\boldsymbol{h} = \frac{1}{\mathsf{lc}(\phi_{\mathcal{F}}(\boldsymbol{h}))}\boldsymbol{h}$   // normalize
            $\boldsymbol{P} = \boldsymbol{P} \cup \{S(\boldsymbol{k}, \boldsymbol{h}) \mid \boldsymbol{k} \in \boldsymbol{G} \text{ and } \mathfrak{s}(\boldsymbol{k}) \neq \mathfrak{s}(\boldsymbol{h})\}$
            $\boldsymbol{G} = \boldsymbol{G} \cup \{\boldsymbol{h}\}$
return $\phi_{\mathcal{F}}(\boldsymbol{G})$
```

chosen. This allows to incorporate techniques from F_4, in particular simultaneous reduction of polynomials. These ideas are expanded upon in, for example, MatrixF$_5$ [11, Sec. 1.5.2] and $\mathrm{F}_{4/5}$ [1].

Intuitively, the correctness of F_5 follows from Theorem 3 and Propositions 4 and 5, i.e., the vectorized Buchberger criterion and the fact that the F_5 criterion discards only unnecessary critical pairs. Because the original publication's proof of termination for general systems of polynomials was flawed, many algorithms slightly altering F_5 to guarantee termination have appeared. An excellent survey, putting these variations as well as generalizations and extensions of F_5 in relation using unified notation and terminology, is due to Eder & Faugère [44]. For the full proof of termination of F_5 as presented here, we refer to [44, Sec. 10].

Complexity Even though there is a compact description of F_5, it is quite difficult to estimate its complexity directly. The algorithm terminates when set $\boldsymbol{P}$ is empty. Since $\boldsymbol{P}$ is growing and shrinking depending on how often the F_5 criterion does or does not apply, the point of termination is inherently difficult to predict. After all, if some method could easily predict applicability of the F_5 criterion, it would automatically give rise to an even faster Gröbner basis algorithm. However, since F_5 does eventually terminate, we know that at some point, set $\boldsymbol{P}$ will not grow anymore. In a degree refining monomial order, the total degree of the highest-degree $\operatorname{lcm}(\phi(\boldsymbol{k}), \phi(\boldsymbol{h}))$ for which $\mathfrak{s}(\boldsymbol{k}) \neq \mathfrak{s}(\boldsymbol{h})$ appearing in Line 13 at any point during F_5's execution coincides with the *degree of regularity*,[11] also known as the *Hilbert regularity*, of $\mathcal{F}$ [12]. For a formal treatment of this notion, we refer to [37, Ch. 9, §3].

The degree of regularity gives an upper bound for the total degree of polynomials appearing during F_5's execution. Consequently, it is used to estimate the complexity of F_5. Namely, for a polynomial ring with n variables, running F_5 on a system of polynomials $f_0, \ldots, f_{s-1}$ whose degree of regularity is d_{reg}, the complexity of F_5 is in $O\big(\binom{n+d_{\mathrm{reg}}}{n}^{\omega}\big)$, where ω is the linear algebra constant [12, Thm. 7].

Intuitively, this complexity comes from the regular $\mathfrak{s}$-reductions in Line 8. These reductions are most costly when the degrees of $\phi(\boldsymbol{g})$ or of the polynomials in $\phi(\boldsymbol{G})$ are the largest, i.e., are equal to the degree of regularity. For the sake of estimating the complexity, imagine keeping the underlying working Gröbner basis $\phi(\boldsymbol{G})$ as a triangulated Macaulay matrix of width m. The regular $\mathfrak{s}$-reduction can then be performed with complexity in $O(m^{\omega})$ using linear algebra techniques. The width m of the Macaulay matrix corresponds to the number of monomials in n variables up to degree d_{reg}, i.e., $m = \binom{n+d_{\mathrm{reg}}}{n}$. For a more detailed treatment, we refer to [12, 13].

In the same manner, the degree of regularity allows estimating the complexity of F_4. Perhaps surprisingly, the complexities of F_4 and F_5 are asymptotically the same, but experiments show that the terms obscured by big-O notation are quite different between the two [41].

Unfortunately, we don't know if it is generally possible to compute the degree of regularity faster than running F_5. However, for the class of regular polynomial systems, a shortcut exists. The degree of regularity d_{reg} of a regular system adheres to the *Macaulay bound* [12], i.e.,

$$d_{\mathrm{reg}}(\mathcal{F}) = 1 + \sum_{i=0}^{s-1} \deg(f_i) - 1. \tag{3}$$

If the system is overdetermined, as can be the case for some cryptographic applications, it is inherently irregular. Here, the notion of *semi-regularity* generalizes regularity. Briefly summarized, the degree of regularity of a semi-regular sequence is the index of the first non-positive coefficient in the projective Hilbert series of $(f_0^h, \ldots, f_{s-1}^h)$. For a more detailed treatment, we refer to [14, Sec. 2.3].

Two approaches exist for estimating the degree of regularity of the polynomial system arising from an AOC. The first is to assume, argue, or ideally prove that the primitive's system of polynomials is (semi-)regular. In this case, the degree of (semi-)regularity can easily be computed from the Hilbert series of the ideal defined by the primitive's polynomial system. The second is using F_5 to find the degree of regularity for several round-reduced variants of the primitive, then extrapolating to the full-round primitive. It is an open question whether the actual degree of regularity of the full-round primitive generally corresponds to the extrapolated value.

[11] Two different notions for the degree of regularity exist. The one not explained here is most commonly used to lower bound the complexity of Gröbner basis computations [40, 42]. It is not clear how the two concepts generally relate [31, Sec. 4]. In fact, the definitions in [42] and [40] don't even coincide.

3.3 eXtended Linearization, with Mutants and Wiedemann

As outlined in Section 1, Gröbner bases are only of incidental interest when algebraically attacking an AOC. The actual objective is finding an element of the variety corresponding to the secret key, hash preimage, etc. The *eXtended Linearization* (XL) [35] family of algorithms, including its various extensions and improvements [24,26,39,74–77], strives to find this element directly. This changes the general attack pipeline given in Section 1: after polynomially modeling the primitive, the intermediary steps of computing a Gröbner basis, changing the term order, and reading out the solution are condensed into one step, namely running XL or one of its extensions to directly arrive at the solution.

In this section, we outline XL and its two most important improvements *Wiedemann XL* [76, Ch. 5.3] and *Mutant XL* [26,39]. Additionally, we point out similarities between Gröbner basis algorithm F_4 and the MXL algorithms [2].

eXtended Linearization The two core techniques of XL are, to no surprise, *extending* and *linearization*. Extending a system of polynomials $\mathcal{F}$ means multiplying its elements by all monomials $m \in \mathcal{M}$ up to a certain degree d: $\{m \cdot f \mid m \in \mathcal{M}, \deg(m) \leqslant d, f \in \mathcal{F}\}$. Linearizing a system of polynomials corresponds to building its Macaulay matrix as described in Definition 12. This amounts to dropping all algebraic relations between the monomials by interpreting the polynomial's coefficients as vectors over the underlying field. Techniques from linear algebra then allow to efficiently solve the ("polynomial") system. The cost of this approach is an explosion in the number of solutions for the linear system which are not solutions for the polynomial system, i.e., the introduction of parasitical solutions.

The XL algorithm combats the potentially prohibitive memory requirements of this explosion, as well as the prohibitive time needed to sift through all solutions to identify non-parasitical ones, by simplifying the polynomial system once a partial solution is identified. This, in turn, introduces the disadvantage that the algorithm might fail because a specific partial solution cannot be extended to a full solution.

The XL algorithm is given in Algorithm 6. Its inputs are the polynomial system $\mathcal{F}$ and additional user-defined integer D, which essentially defines the degree up to which the Macaulay matrices are built. XL iterates the steps *extension*, *linearization*, *solution finding*, and *system simplification*. The partial solution coming from the univariate polynomial f_{uni} is recorded in v to successively build an element of the variety. If the current partial solution cannot be extended, and assuming the primitive under attack was correctly modeled, the partial solution of a previous iteration was an incorrect choice. In this case, XL terminates with an error. Re-running it with different randomness might lead to a different result.

The monomial order of Line 5 can be *lex* order, but doesn't have to be. In particular, any monomial order such that variable x_i only appears in the right-most columns of the Macaulay matrix is sufficient, although generally, not all such monomial orders will lead to the same performance of XL. Interestingly

Algorithm 6: XL

```
   Input: $\mathcal{F} = \{f_0, \dots, f_{s-1}\} \subseteq \mathbb{F}[\mathbf{x}]$, integer $D$
   Output: an element $v$ of the variety $V(\mathcal{F})$
1  $v = ()$
2  $d_{\max} = \max(\{\deg(f) \mid f \in \mathcal{F}\})$
3  for $0 \leqslant i < n$ do
4      $\mathcal{F}_{\text{ext}} = \{m \cdot f \mid m \in \mathcal{M}, \deg(m) \leqslant D - d_{\max}, f \in \mathcal{F}\}$    // extend
5      $\mathcal{F}_{\text{mat}}$ = matrix form of $\mathcal{F}_{\text{ext}}$ in $x_i$-eliminating order    // linearize
6      $\mathcal{F}_{\text{ech}}$ = row reduced echelon form of $\mathcal{F}_{\text{mat}}$
7      $\mathcal{F}_{\text{new}}$ = polynomials of $\mathcal{F}_{\text{ech}}$
8      if $\exists f_{\text{uni}} \in \mathcal{F}_{\text{new}}$ with $f_{\text{uni}}$ univariate in $x_i$ then
9          $r_i$ = a root of $f_{\text{uni}}$   // solve, e.g., using Berlekamp's algorithm
10         $v = v || r_i$    // record partial solution
11         $\mathcal{F} = \{f(x_i = r_i) \mid f \in \mathcal{F}\}$    // simplify system
12     else
13         return error    // partial solution cannot be extended
14 return $v$
```

enough, the monomial order does not have to be the same across iterations, in contrast to the previously discussed (Gröbner basis) algorithms.

The complexity of XL is not trivial to estimate. The bulk of the work is computing the echelon row reduced form of the Macaulay matrices. XL's *operating degree* corresponds to the highest degree d for which a Macaulay matrix was constructed during the algorithm's execution. However, both the operating degree as well the Macaulay matrices' sizes are difficult to know before executing XL.

(Parallel) Wiedemann XL Solving the linear system of Line 6 with sparse linear algebra techniques due to Wiedemann [83] gives rise to WXL [76, Ch. 5.3]. An explanation of Wiedemann's algorithm is beyond the scope of this document. The main improvement of WXL over XL is the lower linear algebra constant for matrix multiplication of 2 plus an additional term dependent on the sparsity of the matrices, which is generally not known beforehand. Again, estimating the sizes of the involved Macaulay matrices is non-trivial, and will be expanded upon below.

WXL is parallelizable, and the resulting algorithm consequently named PWXL [76, 77]. The existence of WXL highlights the importance of using the linear algebra constant implied by sparse techniques, while PWXL necessitates to take parallelization into account when conservatively estimating the complexity of a Gröbner basis computation.

Mutant XL A different improvement of XL is the introduction of so-called *mutants*, leading to *Mutant XL* [26, 39]. Mutants are polynomials used to keep the linear systems as small as possible.

Intuitively, for polynomial system $\mathcal{F} = \{f_0, \dots, f_{s-1}\}$, a mutant is a polynomial $f \in \langle \mathcal{F} \rangle$ resulting in a degree drop. More formally, any expression of $f \in \langle \mathcal{F} \rangle$

as $f = \sum_i h_i f_i$ yields a list of polynomials $(h_0, \ldots, h_{s-1})$ denoted *representation*[12] of f. The *level* of a representation is the highest total degree of any $h_i f_i$, and the *level of* f is the smallest level of all its representations.

Definition 24 (Mutant). *A polynomial $f \in \langle \mathcal{F} \rangle$ that has a level strictly greater than its total degree* $\deg(f)$ *is a* mutant.

An algorithmic description of a simplified MXL is given in Algorithm 7. It essentially performs the following steps: First, the working polynomials are multiplied by all monomials up to certain degree. Linearizing the polynomials results in a linear system. These steps amount to the construction of a Macaulay matrix of some degree d, as before. The linear system is triangularized and any mutants, which can be identified using d_{mut}, are added to the set of working polynomials. When a univariate polynomial is computed, one of its roots is recorded as part of the solution, and the system of polynomials is simplified accordingly, as before. Once the partial solution has been fully extended, the algorithm terminates, yielding the element of the variety.

Relationship between F_4 and MXL Interestingly, the MXL algorithms are essentially equivalent to F_4 with the normal selection strategy [2, 9], albeit with a different time-memory trade-off. Consequently, estimating the complexity of MXL is similarly difficult.

The relation between F_5's degree of regularity and MXL's operating degree, especially in the context of AOCs, is unclear. There are certain systems of polynomials for which the operating degree is dramatically higher [40]. For other systems, experiments indicate that the operating degree is at most 2 higher than the degree of regularity. Using yet again different parameters, a more mathematical approach describes the asymptotic difference to be at most 1 [85]. When estimating the complexity of Gröbner basis computations conservatively, using the degree of regularity seems like a good choice.

4 Term Order Change Algorithms

Out of all monomial orders, computing a Gröbner basis in *degrevlex* order is usually the fastest [15]. However, extracting the variety of an ideal is efficient only with a *lex*-ordered Gröbner basis. The runtime difference between computing bases for the two orders is usually so large that it is faster to first compute a *degrevlex*-ordered basis, then perform a *term order change*[13] to get a *lex*-ordered basis. The two most important term order change algorithms are *FGLM* [48] and the *Gröbner Walk* [33], introduced in Sections 4.1 and 4.3, respectively.

[12] Using the terminology introduced in the previous Section 3.2, a representation corresponds to a vector of origin.

[13] Introducing a slight inconsistency, we refrain from using "*monomial* order change," favoring "*term* order change," which is more commonly found in the literature.

Algorithm 7: Simplified MXL

```
Input: $\mathcal{F} = \{f_0, \ldots, f_{s-1}\} \subseteq \mathbb{F}[\mathbf{x}]$ with $\langle \mathcal{F} \rangle$ zero-dimensional
Output: an element $v$ of the variety $V(\mathcal{F})$
$d_{\max} = \max\{\deg(f) \mid f \in \mathcal{F}\}$
$d_{\mathrm{mut}} = \min\{\deg(f) \mid f \in \mathcal{F}\}$                 // used for identifying mutants
mutants $= \varnothing$
$v = (\epsilon, \ldots, \epsilon)$          // variety element with $\epsilon$ as placeholder. $|v| = n$
while True do
    $\mathcal{F}_{\mathrm{mat}}$ = Macaulay matrix of $\mathcal{F}$ in degrevlex order of degree $d_{\max}$
    $\mathcal{F}_{\mathrm{ech}}$ = row reduced echelon form of $\mathcal{F}_{\mathrm{mat}}$                // echelonize
    $\mathcal{F}_{\mathrm{new}}$ = polynomials of $\mathcal{F}_{\mathrm{ech}}$
    if $\exists f_{\mathrm{uni}} \in \mathcal{F}_{\mathrm{new}}$ with $f_{\mathrm{uni}}$ univariate in some $x_i$ then
        $r_i$ = a root of $f_{\mathrm{uni}}$   // solve, e.g., using Berlekamp's algorithm
        $v_i = r_i$                                  // record partial solution
        $\mathcal{F}_{\mathrm{new}} = \{f(x_i = r_i) \mid f \in \mathcal{F}_{\mathrm{new}}\}$              // simplify system
        if $\nexists \epsilon \in v$ then
            return $v$                                      // solution found
        else
            $d_{\max} = \max\{\deg(f) \mid f \in \mathcal{F}_{\mathrm{new}}\}$
            $d_{\mathrm{mut}} = \min\{\deg(f) \mid f \in \mathcal{F}_{\mathrm{new}}\}$
    else
        mutants = mutants $\cup \{f \in \mathcal{F}_{\mathrm{new}} \mid \deg(f) < d_{\mathrm{mut}}, \mathsf{lm}(f) \notin \mathsf{lm}(\mathcal{F})\}$
        if mutants $\neq \varnothing$ then
            $d_{\min} = \min(\{\deg(f) \mid f \in \text{mutants}\})$         // multiply mutants
            $\text{mutants}_{\mathrm{sel}} = \{f \in \text{mutants} \mid \deg(f) = d_{\min}\}$
            mutants = mutants $\setminus$ $\text{mutants}_{\mathrm{sel}}$
            $\text{mutants}_{\mathrm{ext}} = \{m \cdot f \mid m \in \mathcal{M}, \deg(m) = 1, f \in \text{mutants}_{\mathrm{sel}}\}$
            $\mathcal{F}_{\mathrm{new}} = \mathcal{F}_{\mathrm{new}} \cup \text{mutants}_{\mathrm{ext}}$
            $d_{\mathrm{mut}} = d_{\min} + 1$
        else
            $\mathcal{F}_{\mathrm{sel}} = \{f \in \mathcal{F}_{\mathrm{new}} \mid \deg(f) = d_{\max}\}$                  // extend
            $\mathcal{F}_{\mathrm{ext}} = \{m \cdot f \mid m \in \mathcal{M}, \deg(m) = 1, f \in \mathcal{F}_{\mathrm{sel}}\}$
            $\mathcal{F}_{\mathrm{new}} = \mathcal{F}_{\mathrm{new}} \cup \mathcal{F}_{\mathrm{ext}}$
            $d_{\max} = d_{\max} + 1$
            $d_{\mathrm{mut}} = d_{\max}$
    $\mathcal{F} = \mathcal{F}_{\mathrm{new}}$
```

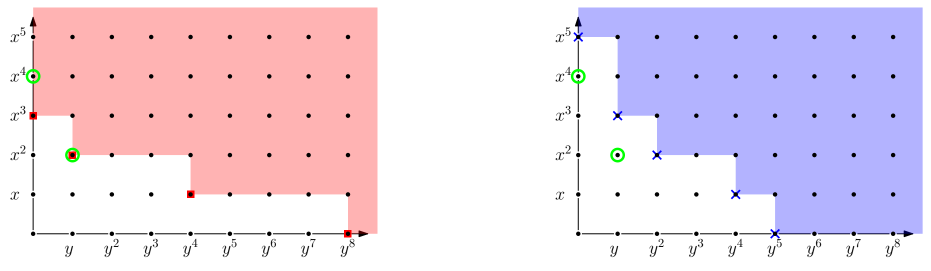
4.1 FGLM

Gröbner bases for the same ideal but with respect to different monomial orders can differ drastically. For a (not so drastic) example, the following two Gröbner bases G_{lex} and G_{invlex} generate the same ideal I but share no polynomial.

$$
\begin{aligned}
G_{\text{lex}} = \{\, g_0 &= x^3 - x^2 - y^6 + y^5, & G_{\text{invlex}} = \{\, g_0 &= y^5 - x^2y - x^4 + 2x^3 - x^2,\\
g_1 &= x^2y + y^7 - y^6, & g_1 &= xy^4 + x^2y + x^4 - x^3,\\
g_2 &= xy^4 + y^6, & g_2 &= x^2y^2 - x^4 + x^3,\\
g_3 &= y^8 \,\} & g_3 &= x^3y,\\
& & g_4 &= x^5 - x^4 \,\}
\end{aligned}
$$

Consequently, full reduction of, for example, $f = x^5 + x^2y^2 - y^6$ by G_{lex} yields remainder $r_{\text{lex}} = x^2 + 2y^7 - y^5$, which is quite different from $r_{\text{invlex}} = -x^2y + x^4$, the remainder of f after full reduction by G_{invlex}.

The monomials that are not reducible by a Gröbner basis fall into a *staircase*, defined by the leading monomials of the basis. An example is given in Figure 3, visualizing the staircases defined by G_{lex} and G_{invlex}. Monomials in the shaded area are reducible by the respective Gröbner basis. All monomials comprising the remainder after full reduction lie inside the staircase.



(a) G_{lex} (b) G_{invlex}

Fig. 3. Staircases of G_{lex} and G_{invlex}. Monomials of r_{invlex} are encircled (○).

While r_{invlex} is not further reducible by G_{invlex}, the monomials of r_{invlex} *can* be reduced by G_{lex}, and vice versa. In Figures 3a and 3b, the monomials of r_{invlex} are encircled, showing that they fall outside the staircase defined by G_{lex}. Full reduction of terms $-x^2y$ and x^4 from r_{invlex} by G_{lex} gives polynomials $(y^7 - y^6)$ and $(x^2 + y^7 + y^6 - y^5)$, respectively. The sum of these two polynomials is equal to r_{lex}. The example, a summary of which can be found in Figure 4, shows two things: (1) Reducing a polynomial is the same as reducing its monomials, then summing, and (2) given a *mapping* between all staircase monomials of one Gröbner basis G_{old} and their reduced forms by another Gröbner basis G_{new}, it is easy to "convert" polynomials reduced by G_{old} to polynomials reduced by

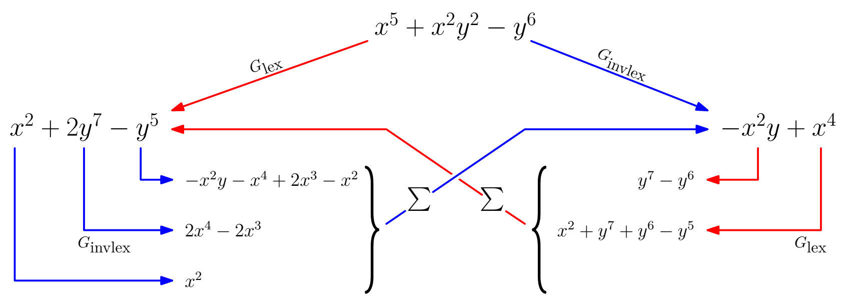


Fig. 4. Monomials and their full reductions, allowing conversion of remainders.

G_{new}, *without changing their residue class.* This mapping is the central tool in the FGLM term order change algorithm.

The remaining question is how to find a mapping between staircase monomials of G_{old} to their G_{new}-reduced forms when G_{new} is not known. For this task, FGLM uses linear algebra in the quotient ring $\mathbb{F}[\mathbf{x}]/I$ taken as an $\mathbb{F}$-vector space. The monomials in the staircase given by G_{old} are a basis for this vector space, since any monomial outside the staircase can be reduced to a polynomial whose monomials fall inside the staircase. FGLM iterates over all monomials in $\prec_{\text{new}}$-increasing order and reduces them by G_{old}, thereby building a mapping like in above example. The mapping can be taken as a dictionary: The monomials are the keys, the (polynomial) remainders of the monomials after reduction by G_{old} are the values. Crucially, at every step the current remainder is checked for a *linear dependency* on the values already in the dictionary. Finding such a linear relation is straightforward, for example with the Gauß'ian algorithm. If a linear relation exists, it means that the current remainder is in the ideal I, since full reduction by G_{old} results in 0. Applying the found linear relation to the corresponding keys then gives an element of the new Gröbner basis G_{new}. Any monomials divisible by the leading term of the new basis element need not be considered when iterating all monomials, since they fall outside the staircase of G_{new} and thus cannot be a basis element of G_{new}.

Because FGLM considers all monomials not divisible by some leading monomial of G_{new}, the algorithm does not terminate if there are infinitely many such monomials. Thus, the $\mathbb{F}$-vector space $\mathbb{F}[\mathbf{x}]/I$ has to be of finite dimension, meaning that ideal I has to be zero-dimensional.

In summary, FGLM iterates over monomials until it hits the border of the staircase defined by G_{new}, which can be efficiently detected through linear dependencies between the monomials' reduced forms. The monomials are iterated over in $\prec_{\text{new}}$-ascending order, determining the order G_{new} is a Gröbner basis for. The full algorithm is given in Algorithm 8.

Example 11. We present here a walk-through of Algorithm 8. The inputs are *lex* as the new monomial order $\prec_{\text{new}}$, Gröbner basis G_{invlex} (on the facing page) as

Algorithm 8: FGLM

```
   Input: $\prec_{\text{new}}$, $G_{\text{old}}$, $\prec_{\text{old}}$
   Output: $G_{\text{new}}$
1  dict = $\varnothing$
2  $G_{\text{new}} = \varnothing$
3  next_monoms = $\{1\}$
4  while next_monoms $\neq \varnothing$ do
5      monom = $\min_{\prec_{\text{new}}}\{\text{next\_monoms}\}$
6      next_monoms = next_monoms $\setminus \{\text{monom}\}$
7      if $\nexists g \in G_{\text{new}}$ such that $\mathsf{lm}_{\prec_{\text{new}}}(g) \mid \text{monom}$ then
        //still within staircase
8          reduced_monom = full reduction of monom by $G_{\text{old}}$ // requires $\prec_{\text{old}}$
9          if reduced_monom + $\sum_{\nu\in\text{dict}} \omega_\nu \cdot \text{value}(\nu) = 0$ for some $\omega_\nu \in \mathbb{F}$ then
10             $G_{\text{new}} = G_{\text{new}} \cup \{\text{monom} + \sum_{\nu\in\text{dict}} \omega_\nu \cdot \nu\}$
11         else
12             dict = dict $\cup$ {monom : reduced_monom}
13             next_monoms = next_monoms $\cup \{x_i \cdot \text{monom} \mid i \in \{0,\ldots,n-1\}\}$
14 return $G_{\text{new}}$
```

G_{old}, and *invlex* as the corresponding monomial order $\prec_{\text{old}}$. The evolution of variables are listed in Table 1.

In iteration 1 of the main loop (Line 4), we have `monom` $= 1$. Since $G_{\text{new}} = \varnothing$, the outer if-clause (Line 7) evaluates to true. Reducing 1 by G_{old} results in 1, and since `dict` is empty, no ω_ν exist such that the second if-clause (Line 9) could evaluate to true. Thus, after the first iteration, we have `dict` $= \{1 : 1\}$ and `next_monoms` $= \{y, x\}$.

In iteration 2, `monom` is set to the $\prec_{\text{new}}$-smallest monomial in `next_monoms`, which is y. Both if-clauses evaluate as before and no reduction of `monom` takes place, resulting in `dict` $= \{1 : 1, y : y\}$ and `next_monoms` $= \{y^2, x, xy\}$ after iteration 2. Iterations 3 through 5 are executed similarly.

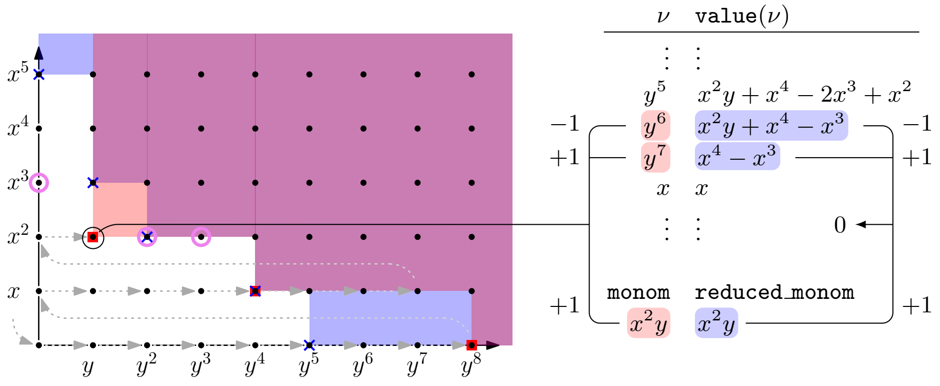
In iteration 6, `monom` $= y^5$ is reducible by G_{old}. The two if-clauses still evaluate to true and false respectively, resulting in new dictionary entry $\{y^5 : x^2y + x^4 - 2x^3 + x^2\}$. Variable `next_monoms` now equals $\{y^6, x, xy, xy^2, xy^3, xy^4, xy^5\}$. Iterations 7 and 8 behave similarly, with dictionary entries $\{y^6 : x^2y + x^4 - x^3\}$ and $\{y^7 : x^4 - x^3\}$ being added respectively.

In iteration 9, we have `monom` $= y^8$. Reduction of `monom` by G_{old} results in `reduced_monom` $= 0$. This means that `monom` is an element of the ideal $\langle G_{\text{old}}\rangle$. Clearly, there is a linear dependency between `reduced_monom` and the `dict`-values, namely $\omega_\nu = 0$ for all $\nu \in$ `dict`. The second if-clause thus evaluates to true for the first time, and after iteration 9, we have $G_{\text{new}} = \{y^8\}$. Variable `next_monoms` now equals $\{xy^i \mid 0 \leqslant i \leqslant 7\}$. Iterations 10 through 13 again simply add entries to the dictionary and expand the set `next_monoms`.

In iteration 14, we have `monom` $= xy^4$ with `reduced_monom` $= -x^2y - x^4 + x^3$. Setting $\omega_{y^6} = 1$, there is a linear dependency between `reduced_monom`

Table 1. Variables in Example 11 before execution of the main loop. The `dict` can be inferred from columns `monom` and `reduced_monom`.

iteration	`monom`	`reduced_monom`	`next_monoms`	G_{new}
0	–	–	$\{1\}$	$\varnothing$
1	1	1	$\{y, x\}$	$\varnothing$
2	y	y	$\{y^2, x, xy\}$	$\varnothing$
3	y^2	y^2	$\{y^3, x, xy, xy^2\}$	$\varnothing$
4	y^3	y^3	$\{y^4, x, xy, xy^2, xy^3\}$	$\varnothing$
5	y^4	y^4	$\{y^5, x, xy, xy^2, xy^3, xy^4\}$	$\varnothing$
6	y^5	$x^2y + x^4 - 2x^3 + x^2$	$\{y^6, x, xy, xy^2, xy^3, xy^4, xy^5\}$	$\varnothing$
7	y^6	$x^2y + x^4 - x^3$	$\{y^7, x, xy, xy^2, xy^3, xy^4, xy^5, xy^6\}$	$\varnothing$
8	y^7	$x^4 - x^3$	$\{y^8, x, xy, xy^2, xy^3, xy^4, xy^5, xy^6, xy^7\}$	$\varnothing$
9	y^8	0	$\{x, xy, xy^2, xy^3, xy^4, xy^5, xy^6, xy^7\}$	$\{y^8\}$
10	x	x	$\{xy, xy^2, xy^3, xy^4, xy^5, xy^6, xy^7, x^2\}$	$\{y^8\}$
11	xy	xy	$\{xy^2, xy^3, xy^4, xy^5, xy^6, xy^7, x^2, x^2y\}$	$\{y^8\}$
12	xy^2	xy^2	$\{xy^3, xy^4, xy^5, xy^6, xy^7, x^2, x^2y, x^2y^2\}$	$\{y^8\}$
13	xy^3	xy^3	$\{xy^4, xy^5, xy^6, xy^7, x^2, x^2y, x^2y^2, x^2y^3\}$	$\{y^8\}$
14	xy^4	$-x^2y - x^4 + x^3$	$\{xy^5, xy^6, xy^7, x^2, x^2y, x^2y^2, x^2y^3\}$	$\{y^8, xy^4 + y^6\}$
15	xy^5	–	$\{xy^6, xy^7, x^2, x^2y, x^2y^2, x^2y^3\}$	$\{y^8, xy^4 + y^6\}$
16	xy^6	–	$\{xy^7, x^2, x^2y, x^2y^2, x^2y^3\}$	$\{y^8, xy^4 + y^6\}$
17	xy^7	–	$\{x^2, x^2y, x^2y^2, x^2y^3\}$	$\{y^8, xy^4 + y^6\}$
18	x^2	x^2	$\{x^2y, x^2y^2, x^2y^3, x^3\}$	$\{y^8, xy^4 + y^6\}$
19	x^2y	x^2y	$\{x^2y^2, x^2y^3, x^3\}$	$\{y^8, xy^4 + y^6, x^2y + y^5 - y^6\}$
20	x^2y^2	–	$\{x^2y^3, x^3\}$	$\{y^8, xy^4 + y^6, x^2y + y^5 - y^6\}$
21	x^2y^3	–	$\{x^3\}$	$\{y^8, xy^4 + y^6, x^2y + y^5 - y^6\}$
22	x^3	x^3	$\varnothing$	$\{y^8, xy^4 + y^6, x^2y + y^5 - y^6, x^3 - x^2 - y^6 + y^5\}$


x^5
x^4
x^3
x^2
x
y y^2 y^3 y^4 y^5 y^6 y^7 y^8
ν value(ν)
y^5 $x^2y + x^4 - 2x^3 + x^2$
-1 y^6 $x^2y + x^4 - x^3$ -1
$+1$ y^7 $x^4 - x^3$ $+1$
x x
0
monom reduced_monom
$+1$ x^2y x^2y $+1$

(a) Visualization of variables. (b) Linear relation and ω_ν's.

Fig. 5. Visualization of iteration 19 of Example 11. Right-hand Figure 5b highlights the linear relation between dictionary values and `reduced_monom`, resulting in a new basis element composed of `monom` and corresponding dictionary keys. In left-hand Figure 5a, the staircase of basis G_{old} is blue, partial staircase of G_{new} is red. Their elements' leading monomials are marked (✕ and 🟥 respectively). Arrows indicate, in order, the monomials `monom` of previous iterations. Elements of `next_monoms` are encircled (○).

and the values of `dict`, namely `reduced_monom` $+ \, \omega_{y^6}$`value`$(y^6) = 0$. Thus, the second if-clause evaluates to true and `monom` $+ \sum_{\nu \in \texttt{dict}} \omega_\nu \cdot \nu = xy^4 + y^6$ is added to G_{new} as a new basis element. Variable `next_monoms` is now $\{xy^5, xy^6, xy^7, x^2, x^2y, x^2y^2, x^2y^3\}$.

The leading monomial with respect to $\prec_{\text{new}}$ of the newly added basis element $\mathsf{lm}_{\prec_{\text{new}}}(xy^4 + y^6) = xy^4$ divides the three $\prec_{\text{new}}$-smallest elements of `next_monoms`. Thus, the outer if-clause evaluates to true in iterations 15 through 17. We already know these monomials to be in the ideal, and they are removed from `next_monoms` without further processing. Iteration 18 removes x^2 from `next_monoms`, creates dictionary entry $\{x^2 : x^2\}$, and adds x^3 to `next_monoms`. (The monomial x^2y, also computed in Line 13, is already an element of set `next_monoms`.)

Iteration 19 is visualized in Figure 5. Setting $\omega_{y^6} = -1$ and $\omega_{y^7} = 1$ results in a linear dependency, and the inner if-clause evaluates to true. Element $x^2y + y^5 - y^6$ is added to G_{new}. Consequently, in iterations 20 and 21, the two $\prec_{\text{new}}$-smallest elements of `next_monoms`, i.e., x^2y^2 and x^2y^3, are discarded.

In iteration 22, with `monom` $= x^3$, G_{new} is extended by $x^3 - x^2 + y^5 - y^6$. This results in empty `next_monoms` at the start of iteration 23. Variable G_{new} now equals G_{lex} on page 28. It is returned, and the algorithm terminates.

Complexity Compared to the complexity of Gröbner basis computations, the complexity of FGLM is rather well understood. Essentially, in the outermost loop (Line 4), all monomials not reducible by G_{new}, i.e., all monomials in the staircase of G_{new}, are enumerated. These monomials correspond precisely to a basis B of the $\mathbb{F}$-vector space $\mathbb{F}[\mathbf{x}]/I$. Then, $D = \dim_{\mathbb{F}}(\mathbb{F}[\mathbf{x}]/I)$ is the *volume* of the staircase. Since $I = \langle G_{\text{old}} \rangle = \langle G_{\text{new}} \rangle$, volume D is independent of the Gröbner basis. As a consequence, D can be easily deduced from any Gröbner basis' leading monomials, in particular $\mathsf{lm}(G_{\text{old}})$.

Any fully reduced polynomial has a maximum of D terms. During the execution of FGLM, only fully reduced polynomials are stored in the dictionary. Representation of one such polynomial is possible with a vector of dimension D, by listing all its coefficients. As every monomial in the staircase results in one entry in the dictionary, the dictionary's values can be stored as one matrix of dimension at most $D \times D$. Keeping this matrix in triangular form allows the linear dependency check of Line 9 to be performed with complexity in $O(D^2)$. Since not only the monomials in the staircase but also those at the staircase's border are enumerated during the execution of FGLM, up to nD many such linear dependency checks are performed. The total complexity is thus in $O(nD^3)$. For a full proof of this sketch, we refer to [48, Sec. 5].

4.2 Sparse FGLM

Sparse FGLM [49] employs sparse linear algebra techniques to improve on FGLM in terms of complexity. Sparse FGLM is a randomized algorithm, possibly failing but detectably so. Furthermore, Sparse FGLM inherently converts a given Gröbner basis to *lex* order, i.e., does not take the target order as an input.[14] Here, we only outline the steps of Sparse FGLM for recovering the univariate polynomial of the Gröbner basis in *lex* order. After relabeling variables such that this polynomial is in the variable for the key, the next steps are trivial. A parallel version of Sparse FGLM using Coppersmith's Block-Wiedemann algorithm [34] is presented in [64].

Given a Gröbner basis G for ideal I over polynomial ring $\mathbb{F}[\mathbf{x}]$, let B denote any canonical basis of $\mathbb{F}$-vector space $\mathbb{F}[\mathbf{x}]/I$, and D the dimension of the space. The first step in Sparse FGLM is identifying multiplication matrix $T \in \mathbb{F}^{D \times D}$ such that for all monomials $m \in \mathbb{F}[\mathbf{x}]/I$, we have $T\boldsymbol{m} = \boldsymbol{x_0 m}$, where $\boldsymbol{m}$ and $\boldsymbol{x_0 m}$ are the vectorizations of m and $x_0 m$ according to B, respectively. In particular, $\boldsymbol{x_0 m}$ is the vector representing $x_0 m$ after reduction by G. In the next step, Wiedemann's sparse linear algebra solver finds a kernel vector $\boldsymbol{k}$ of T. A nonzero kernel element $\boldsymbol{k}$ of T corresponds to the weights of the FGLM linear relation, according to B. Thus, $\boldsymbol{k}$ gives rise to the univariate polynomial $k \in I$ in the *lex*-ordered Gröbner basis.

The complexity of Sparse FGLM boils down to that of Wiedemann's algorithm, i.e., $O(DN_0)$, where N_0 is the number of non-zero entries of T [54, Ch. 12.3]. In general, N_0 is greater than D, equality holding only for certain degenerated Gröbner bases, which allow to directly read off the interesting part of the solution. Sparsity of T can be estimated in the following way. Let t denote the number of monomials m such that $x_0 m$ can be reduced by G, but m cannot. Let further $|G|$

[14] Two more algorithms are presented in [49], both generalizing Sparse FGLM in some manner. All three algorithms can be seen as special cases of FGLM.

denote the number of non-zero non-leading coefficients of all elements of G. Then, T consists of t columns with at most $|G|$ non-zero entries, and each remaining column is some unit vector. In summary, $N_0 \leqslant t|G| + (D - t)$. For an asymptotic analysis of T's sparsity, we refer to [49, Sec. 6.4].

4.3 The Gröbner Walk

The *Gröbner Walk* [33] is a term order change algorithm conceptually very different from FGLM. The Walk is of particular interest because it does not impose any restrictions on the ideal's dimension but terminates even for positive-dimensional ideals. However, the ideals for the applications considered in this paper are always of dimension zero. Additionally, and in contrast to FGLM, the complexity of the Gröbner Walk is not well understood. Experimental comparisons indicate that it rarely outperforms FGLM. For these reasons, the Walk is only sketched. A comprehensive introduction is given in [36, Ch. 8, §§4&5], whose example we use. Leading up to the Gröbner Walk, we first introduce the concepts of *marked* Gröbner bases, *Gröbner cones*, and *Gröbner fans* [78].

A useful concept for the remainder of this section is a polynomial's *leading form*, a generalization of the leading term. A leading form is most easily described using monomial orders given by weight vectors, as introduced in Section 2.1. Given a monomial order $\prec_{\boldsymbol{\omega}}$ by weight matrix $\left(\begin{smallmatrix}\boldsymbol{\omega}\\ M_{\boldsymbol{\omega}}\end{smallmatrix}\right)$ where $\boldsymbol{\omega} \in \mathbb{R}^n_{\geqslant 0}$[15] and $M_{\boldsymbol{\omega}} \in \mathbb{R}^{(n-1)\times n}$, the leading form of polynomial $f = \sum_{\boldsymbol{\alpha}} c_{\boldsymbol{\alpha}} \mathbf{x}^{\boldsymbol{\alpha}}$ is the sum of all $\boldsymbol{\omega}$-maximal terms in f, i.e., those terms for which $\boldsymbol{\omega} \cdot \boldsymbol{\alpha}$ is maximal among all of f's terms with non-zero coefficients. For example, given $f = x^2yz - xy^2z^2 + 3xyz \in \mathbb{F}[x, y, z]$ and $\boldsymbol{\omega} = (2, 1, 1)$, we have $\mathsf{lf}(f) = x^2yz - xy^2z^2$.

A polynomial f in conjunction with its leading form $\mathsf{lf}(f)$ is called a *marked polynomial*. For ease of notation, the leading form is commonly underlined: $f = \underline{x^2yz - xy^2z^2} + 3xyz$. As for leading terms, monomials, and coefficients, the extension to sets is defined: $\mathsf{lf}(\mathcal{F}) = \bigcup_{f\in\mathcal{F}} \mathsf{lf}(f)$.

The (unique) reduced Gröbner basis for some monomial order $\prec_{\boldsymbol{\omega}}$ is a *marked* Gröbner basis if all its elements are marked with respect to $\boldsymbol{\omega}$. Given a fixed ideal I, the marked Gröbner bases for different monomial orderings $\prec_{\boldsymbol{\omega}}$ and $\prec_{\boldsymbol{\tau}}$ can be the same. For example, the marked Gröbner basis of $\langle x + y\rangle$ with respect to both *lex* and *degrevlex* is $\{\underline{x} + y\}$, whereas for *invlex*, it is $\{x + \underline{y}\}$. Note that the (unmarked) reduced Gröbner basis of $\langle x + y\rangle$ is independent of the monomial order since none of the polynomial's terms are distinguished.

Two monomial orderings are considered equivalent for some ideal I if their marked Gröbner bases coincide:

$$\prec_{\boldsymbol{\omega}} \sim \prec_{\boldsymbol{\tau}} \quad \Leftrightarrow \quad \mathsf{lf}_{\boldsymbol{\omega}}(I) = \mathsf{lf}_{\boldsymbol{\tau}}(I) \tag{4}$$

where $\mathsf{lf}_{\boldsymbol{\gamma}}(I)$ is the set of leading forms with respect to $\boldsymbol{\gamma}$ of the reduced Gröbner basis of I with respect to $\prec_{\boldsymbol{\gamma}}$. The equivalence class of $\prec_{\boldsymbol{\omega}}$ under $\sim$ is called *Gröbner cone* $C_{\boldsymbol{\omega}}(I)$. A Gröbner cone is a convex polyhedron [50, Prop. 2.6].

[15] For a more general treatment of weight vectors in $\mathbb{R}^n$ as opposed to $\mathbb{R}^n_{\geqslant 0}$, see [50, 51].

The Gröbner cones partition $\mathbb{R}^n_{\geqslant 0}$. Any ideal has only finitely many Gröbner cones [78, Lemma 2.6].

Given Gröbner basis $G_{\boldsymbol{\omega}} = \{f_0, \dots, f_{s-1}\}$ for I with respect to $\prec_{\boldsymbol{\omega}}$, we can also describe the cone $C_{\boldsymbol{\omega}}(I)$ as follows. Let $f_i = c_{\boldsymbol{\alpha}_{i,\max}} \mathbf{x}^{\boldsymbol{\alpha}_{i,\max}} + \sum_j c_{\boldsymbol{\alpha}_{i,j}} \mathbf{x}^{\boldsymbol{\alpha}_{i,j}}$ where all coefficients $c_{\boldsymbol{\alpha}_{i,\max}}$ and $c_{\boldsymbol{\alpha}_{i,j}}$ are non-zero and $\mathbf{x}^{\boldsymbol{\alpha}_{i,\max}} \succ_{\boldsymbol{\omega}} \mathbf{x}^{\boldsymbol{\alpha}_{i,j}}$ for all j, i.e., $\mathbf{x}^{\boldsymbol{\alpha}_{i,\max}}$ is the leading monomial of f_i. In other words, we have $\boldsymbol{\alpha}_{i,\max} \cdot \boldsymbol{\omega} \geqslant \boldsymbol{\alpha}_{i,j} \cdot \boldsymbol{\omega}$ for all j. By setting $\boldsymbol{v}_{i,j} = (\boldsymbol{\alpha}_{i,\max} - \boldsymbol{\alpha}_{i,j})$, we get a set of vectors $\{\boldsymbol{v}_{i,j}\} \subseteq \mathbb{R}^n$ describing cone $C_{\boldsymbol{\omega}}(I)$:

$$\mathbb{R}^n_{\geqslant 0} \ni \boldsymbol{\tau} \in C_{\boldsymbol{\omega}}(I) \Leftrightarrow \forall i, j : \boldsymbol{v}_{i,j} \cdot \boldsymbol{\tau} \geqslant 0. \tag{5}$$

In particular, each $\boldsymbol{v}_{i,j}$ corresponds to one face of the cone: if $\boldsymbol{v}_{i,j} \cdot \boldsymbol{\tau} < 0$ for some $\boldsymbol{\tau}$, then $\mathbf{x}^{\boldsymbol{\alpha}_{i,\max}} \prec_{\boldsymbol{\tau}} \mathbf{x}^{\boldsymbol{\alpha}_{i,j}}$, meaning $\mathsf{lf}_{\boldsymbol{\tau}}(f_i) \neq \mathsf{lf}_{\boldsymbol{\omega}}(f_i)$, i.e., $\boldsymbol{\tau}$ is not in the same cone as $\boldsymbol{\omega}$.

Definition 25 (Gröbner Fan [50]). *The collection of all Gröbner cones* $\{C_{\boldsymbol{\omega}}(I)\}_{\boldsymbol{\omega} \in \mathbb{R}^n_{\leqslant 0}}$ *of ideal* $I \subseteq \mathbb{F}[x_0, \dots, x_{n-1}]$ *is the* Gröbner fan *of* I*.*[16]

Example 12 ([36, Ch. 8, §4]). The Gröbner fan of an ideal I in (up to) 3 variables can be neatly visualized. Let $G = \{x^2 - y, xz - y^2 + yz\} \subseteq \mathbb{F}[x, y, z]$. The intersection of all (seven) Gröbner cones in the Gröbner fan for G with the hyperplane $|\boldsymbol{\gamma}| = 1$ are shown in Figure 6. The single ray marked $\boldsymbol{\omega}$ corresponds to weight vector $(7, 7, 2)$, intersecting the plane at $(7/16, 7/16, 1/8)$.

The Gröbner basis associated with $\boldsymbol{\omega}$ coincides with the Gröbner basis for *degrevlex*, namely $\{x^2 - y, y^2 - xz - yz\}$. However, the marked Gröbner bases differ: for $\boldsymbol{\omega}$, we have $\{\underline{x^2} - y, \underline{y^2} - xz - yz\}$, but for *degrevlex*, the most dominant weight vector being $(1, 1, 1)$, the marked Gröbner basis is $\{\underline{x^2} - y, \underline{y^2 - xz - yz}\}$.

To increase intuition about the Walk, it is worth highlighting some correspondences between weight orders, leading forms, and a Gröbner cone's faces – see also Table 2 for a brief summary. In particular, given $\prec_{\boldsymbol{\omega}}$ by weight matrix $\binom{\boldsymbol{\omega}}{M_{\boldsymbol{\omega}}}$, note that if a polynomial f's leading form coincides with its leading term, i.e., if $\mathsf{lf}_{\boldsymbol{\omega}}(f) = \mathsf{lt}_{\boldsymbol{\omega}}(f)$, then weight vector $\boldsymbol{\omega}$ suffices to identify the leading term of f. However, if f has more than one $\boldsymbol{\omega}$-maximal term, weight vectors of $M_{\boldsymbol{\omega}}$ are needed to "break ties" for correct monomial sorting, and we have $\mathsf{lf}_{\boldsymbol{\omega}}(f) \neq \mathsf{lt}_{\boldsymbol{\omega}}(f)$. This (lack of) necessity for breaking ties, i.e., whether a single weight vector suffices to identify the leading monomial, directly corresponds to a Gröbner cone's interior and faces. Infinitely many monomial orderings exist for which some $\boldsymbol{\omega}$ is the most significant weight vector. Regardless, if $\boldsymbol{\omega}$ is in the interior of its cone $C_{\boldsymbol{\omega}}(I)$ – for example as in Figure 6 – the reduced Gröbner basis $G_{\boldsymbol{\omega}}$ is the same for all these monomial orderings. However, for a vector $\boldsymbol{\tau}$ on a cone's face – like $(1/3, 1/3, 1/3)$ in Figure 6 – there are M_0, M_1 such that the Gröbner bases for monomial orders implied by $\binom{\boldsymbol{\tau}}{M_0}$ and $\binom{\boldsymbol{\tau}}{M_1}$ differ. The additional weight vectors in M_0 and M_1 define how to break ties between $\boldsymbol{\tau}$-maximal monomials. The Walk uses such "neighboring" monomial orders to convert between arbitrary monomial orders.

[16] For reasons of simplicity, we omit the concepts of *Gröbner regions* and *restricted* or *extended* Gröbner fans. For details, see [50, 78].

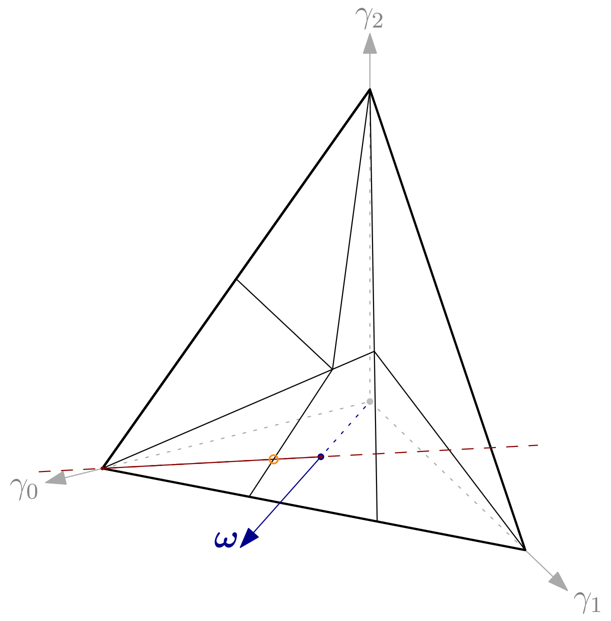


Fig. 6. A slice of the Gröbner fan of G. Vector $\boldsymbol{\omega}$ corresponds to weight $(7,7,2)$. The red line corresponds to the path for the Gröbner Walk from normalized $\boldsymbol{\omega}$ to $(1,0,0)$. The (single) intersection of that path with a cones' face is encircled.

Given a starting monomial order $\prec_{\boldsymbol{\omega}}$ by weight matrix $\left(\begin{smallmatrix}\boldsymbol{\omega}\\ M_{\boldsymbol{\omega}}\end{smallmatrix}\right)$ as before, a marked Gröbner basis $G_{\boldsymbol{\omega}}$ for $\prec_{\boldsymbol{\omega}}$, and a target monomial order $\prec_{\boldsymbol{\tau}}$ induced by $\left(\begin{smallmatrix}\boldsymbol{\tau}\\ M_{\boldsymbol{\tau}}\end{smallmatrix}\right)$, the *Classic Gröbner Walk* [33] converts $G_{\boldsymbol{\omega}}$ into a Gröbner basis $G_{\boldsymbol{\tau}}$ for $\prec_{\boldsymbol{\tau}}$. Since each Gröbner cone and the Gröbner fan are convex, there is a piecewise linear path from $\boldsymbol{\omega}$ to $\boldsymbol{\tau}$ through the Gröbner fan – for example the line segment $(1-\delta)\boldsymbol{\omega}+\delta\boldsymbol{\tau}$ with $0 \leqslant \delta \leqslant 1$. The Gröbner Walk "follows" this[17] path, starting from $\boldsymbol{\omega}$, by (1) crossing into the next neighboring cone in direction $\boldsymbol{\tau}$, and (2) transforming the working Gröbner basis to match that cone. These two steps are iterated until the cone containing the target monomial order is reached, i.e., until the working monomial order is equivalent to the target monomial order. Below, we first discuss the problem associated with step (1), identifying a face of

Table 2. Correspondences between weight vectors, Gröbner cones, and lf & lt.

$\mathsf{lf}_{\boldsymbol{\omega}}(f) = \mathsf{lt}_{\boldsymbol{\omega}}(f)$	vs	$\mathsf{lf}_{\boldsymbol{\omega}}(f) \neq \mathsf{lt}_{\boldsymbol{\omega}}(f)$
one $\boldsymbol{\omega}$-maximal monomial in f	vs	multiple $\boldsymbol{\omega}$-maximal monomials in f
no need to break ties	vs	need to break ties
only need $\boldsymbol{\omega}$ to identify $\mathsf{lt}_{\boldsymbol{\omega}}(f)$	vs	need $\boldsymbol{\omega}$ and $M_{\boldsymbol{\omega}}$ to identify $\mathsf{lt}_{\boldsymbol{\omega}}(f)$
$\boldsymbol{\omega}$ in interior of $C_{\boldsymbol{\omega}}(I)$	vs	$\boldsymbol{\omega}$ on face of $C_{\boldsymbol{\omega}}(I)$
marked basis independent of $M_{\boldsymbol{\omega}}$	vs	marked basis depends on $M_{\boldsymbol{\omega}}$

the cone intersecting with the line segment, and then step (2), the Gröbner basis transformation between two neighboring cones.

As explained above, the starting cone's constraints can be expressed by a set of vectors $\{\boldsymbol{v}_{i,j}\}$ corresponding to the cone's faces. The largest step we can take along the line segment $(1-\delta)\boldsymbol{\omega} + \delta\boldsymbol{\tau}$ while staying in the same cone, i.e., adhering to $\boldsymbol{v}_{i,j} \cdot \boldsymbol{\gamma} \geqslant 0$, is

$$\delta_{\max} = \max(\{\delta \in [0,1] \mid \forall i,j : \boldsymbol{v}_{i,j} \cdot ((1-\delta)\boldsymbol{\omega} + \delta\boldsymbol{\tau}) \geqslant 0\}). \tag{6}$$

We can find $\delta_{\max}$ using Algorithm 9, iterating all faces of the current cone.

Algorithm 9: Gröbner Walk's Step

```
  Input: $\boldsymbol{\omega}$, $\boldsymbol{\tau}$, $\{\boldsymbol{v}_{i,j}\}$
  Output: $\delta_{\max}$
1 $\delta_{\max} = 1$
2 foreach $\boldsymbol{v} \in \{\boldsymbol{v}_{i,j}\}$ do
3     if $\boldsymbol{v} \cdot \boldsymbol{\tau} < 0$ then          // implies leading monomial would change
4         $\delta = \frac{\boldsymbol{v}\cdot\boldsymbol{\omega}}{\boldsymbol{v}\cdot\boldsymbol{\omega} - \boldsymbol{v}\cdot\boldsymbol{\tau}}$                    // biggest step adhering to $\boldsymbol{v}$
5         $\delta_{\max} = \min(\delta, \delta_{\max})$              // smallest of all biggest steps
6 return $\delta_{\max}$
```

Example 13 (continued). Let (normalized) $\boldsymbol{\omega} = (7/16, 7/16, 1/8)$, $G_{\boldsymbol{\omega}} = \{\underline{x^2} - y, \underline{y^2} - xz - yz\} \subseteq \mathbb{F}[x,y,z]$, and $I = \langle G \rangle$, as before. The cone $C_{\boldsymbol{\omega}}(I)$ is defined through $x^2 \succeq_{\boldsymbol{\omega}} y$ from the first polynomial, and $y^2 \succeq_{\boldsymbol{\omega}} xz$ and $y^2 \succeq_{\boldsymbol{\omega}} yz$ from the second. Equivalently:

$$\begin{aligned}
(2,0,0)\cdot\boldsymbol{\omega} \geqslant (0,1,0)\cdot\boldsymbol{\omega} \quad &\Leftrightarrow \quad (2,-1,0)\cdot\boldsymbol{\omega} \geqslant 0 \quad &\Leftrightarrow: \quad \boldsymbol{v}_{0,0}\cdot\boldsymbol{\omega} \geqslant 0\\
(0,2,0)\cdot\boldsymbol{\omega} \geqslant (1,0,1)\cdot\boldsymbol{\omega} \quad &\Leftrightarrow \quad (-1,2,-1)\cdot\boldsymbol{\omega} \geqslant 0 \quad &\Leftrightarrow: \quad \boldsymbol{v}_{1,0}\cdot\boldsymbol{\omega} \geqslant 0\\
(0,2,0)\cdot\boldsymbol{\omega} \geqslant (0,1,1)\cdot\boldsymbol{\omega} \quad &\Leftrightarrow \quad (0,1,-1)\cdot\boldsymbol{\omega} \geqslant 0 \quad &\Leftrightarrow: \quad \boldsymbol{v}_{1,1}\cdot\boldsymbol{\omega} \geqslant 0
\end{aligned}$$

Let the target monomial order be *lex*, with corresponding target weight vector $\boldsymbol{\tau} = (1,0,0)$, coinciding with the γ_0-axis in Figure 6. When executing Algorithm 9, we get $\boldsymbol{v}_{0,0}\cdot\boldsymbol{\tau} = 2$, $\boldsymbol{v}_{1,0}\cdot\boldsymbol{\tau} = -1$, and $\boldsymbol{v}_{1,1}\cdot\boldsymbol{\tau} = 0$, resulting in $\delta_{\max} = \frac{5/16}{5/16+1} = 5/21$.

The next step in the Gröbner Walk is to bring a marked Gröbner basis over to a neighboring cone given a weight vector on the two cones' shared face – for example $(1-\delta_{\max})\boldsymbol{\omega} + \delta_{\max}\boldsymbol{\tau} =: \boldsymbol{\nu}$. For reasons of brevity, we omit all proofs, referring to [33, 36] instead. The transformation requires the Gröbner basis of an intermediate, smaller problem, described next. To find that Gröbner basis, any algorithm, like those described in Section 3, can be used.

[17] The Gröbner Walk is not limited to the described line segment. It works correctly for any path from $\boldsymbol{\omega}$ to $\boldsymbol{\tau}$ not leaving the Gröbner fan. For reasons of simplicity, we do not consider any other paths.

The polynomials G_{inter} defining the intermediate problem are the leading forms of $G_{\boldsymbol{\omega}}$'s elements with respect to $\boldsymbol{\nu}$. In "nice" cases, namely if $\boldsymbol{\nu}$ describes the intersection of the line segment with the cone $C_{\boldsymbol{\omega}}(I)$ on exactly one face, G_{inter} consists only of monomials and one binomial. Let H be the Gröbner basis for $\langle G_{\text{inter}}\rangle$. We can write each element of H as $h_i = \sum_{g\in G_{\boldsymbol{\omega}}} p_{i,g}\mathsf{lf}_{\boldsymbol{\nu}}(g)$ for some polynomials $p_{i,g}$.[18] The same sum where the actual polynomials g replace their respective leading forms, i.e., $\{\sum_{g\in G_{\boldsymbol{\omega}}} p_{i,g}g\}_i$, gives a Gröbner basis $G'_{\boldsymbol{\nu}}$ with respect to $\prec_{\boldsymbol{\nu}}$. Reducing $G'_{\boldsymbol{\nu}}$, i.e., computing the remainder of g with respect to $G'_{\boldsymbol{\nu}} \setminus \{g\}$ for every $g \in G'_{\boldsymbol{\nu}}$, gives the reduced Gröbner basis $G_{\boldsymbol{\nu}}$ for $\langle G_{\boldsymbol{\omega}}\rangle$ with respect to intermediary monomial order $\prec_{\boldsymbol{\nu}}$.

Example 14 (continued). For $\delta_{\max}$ of the previous step, we have $(1-\delta_{\max})\boldsymbol{\omega} + \delta_{\max}\boldsymbol{\tau} = (4/7, 1/3, 2/21) =: \boldsymbol{\nu}$. Combined with the target monomial order *lex*'s weight matrix $M_{\text{lex}} = \left(\begin{smallmatrix}1&0&0\\0&1&0\end{smallmatrix}\right)$ this gives rise to weight order matrix $\left(\begin{smallmatrix}\boldsymbol{\nu}\\M_{\text{lex}}\end{smallmatrix}\right)$, defining monomial order $\prec_{\boldsymbol{\nu}}$. The leading form of $G_{\boldsymbol{\omega}}$ with respect to $\prec_{\boldsymbol{\nu}}$ is $\mathsf{lf}_{\boldsymbol{\nu}}(G_{\boldsymbol{\omega}}) = \{x^2, y^2 - xz\}$. The Gröbner basis of $\mathsf{lf}_{\boldsymbol{\nu}}(G_{\boldsymbol{\omega}})$, also with respect to $\prec_{\boldsymbol{\nu}}$, is $H = \{x^2, y^2 - xz, xy^2, y^4\}$. We can write

$$\begin{aligned}(1,0)\cdot\mathsf{lf}_{\boldsymbol{\nu}}(G_{\boldsymbol{\omega}}) &= x^2,\\ (0,1)\cdot\mathsf{lf}_{\boldsymbol{\nu}}(G_{\boldsymbol{\omega}}) &= y^2 - xz,\\ (z,x)\cdot\mathsf{lf}_{\boldsymbol{\nu}}(G_{\boldsymbol{\omega}}) &= xy^2,\\ (z^2, xz+y^2)\cdot\mathsf{lf}_{\boldsymbol{\nu}}(G_{\boldsymbol{\omega}}) &= y^4,\end{aligned}$$

where, resolving ambiguity caused by the abuse of notation, set $\mathsf{lf}_{\boldsymbol{\nu}}(G_{\boldsymbol{\omega}})$ is interpreted as a vector with elements in the same order as written above. By taking the identified linear combinations of not just the leading form of $G_{\boldsymbol{\omega}}$, but the respective complete polynomials, we get set $G'_{\boldsymbol{\nu}}$ consisting of

$$\begin{aligned}(1,0)\cdot G_{\boldsymbol{\omega}} &= x^2 - y,\\ (0,1)\cdot G_{\boldsymbol{\omega}} &= y^2 - xz - yz,\\ (z,x)\cdot G_{\boldsymbol{\omega}} &= xy^2 - xyz - yz,\\ (z^2, xz+y^2)\cdot G_{\boldsymbol{\omega}} &= y^4 - xyz^2 - y^3z - yz^2.\end{aligned}$$

After reducing each $g \in G'_{\boldsymbol{\nu}}$ by $G'_{\boldsymbol{\nu}} \setminus \{g\}$, we get

$$G_{\boldsymbol{\nu}} = \{x^2 - y, xz - y^2 + yz, xy^2 - y^3 + y^2z - yz, y^4 - 2y^3z + y^2z^2 - yz^2\},$$

which is the reduced Gröbner basis of $\langle G_{\boldsymbol{\omega}}\rangle$ with respect to $\prec_{\boldsymbol{\nu}}$. Furthermore, we have $\prec_{\boldsymbol{\nu}}\sim\prec_{\boldsymbol{\tau}}$, and the Gröbner Walk terminates.

No complexity bound for the Gröbner Walk is known. In part, this is due to one of the Walk's sub-tasks involving the computation of Gröbner bases for ideals not known a-priori, where even *given* an ideal's generating polynomials, estimating the complexity of the Gröbner basis computation is difficult. Additionally, the path

[18] In the terminology of Section 3.2, the $p_{i,g}$ make up the vector of origin for h_i.

through the fan dictates the number of transformations as well as the intermediate problems' polynomials: an intersection with a face of low co-dimension corresponds to polynomials with fewer terms, for which finding a Gröbner basis appears to be easier. Since the shape of the Gröbner fan of an ideal is not known a priori,[19] it is impossible to plot an optimal path.

Several follow-up works improve on the Gröbner Walk as sketched here by identifying better paths. For example, [8] presents methods for heuristically perturbing weight vectors corresponding to starting, intermediate, and target monomial orders within their respective equivalence classes to intersect with the cones' faces more favorably. Deterministic perturbations are discussed in [82]. Additional strategies to dodge intersections of high co-dimensionality include the *Evasive Walk* [8], the *Fractal Walk* [8], and the *Generic Gröbner Walk* [51].

5 Open Questions

We raise some selected questions specific to Gröbner basis attacks on AOCs.

- The degree of regularity is the primary metric when assessing an AOC's resistance against Gröbner basis attacks. For general systems of polynomials, i.e., systems not known to be semi-regular, the only currently available method for computing the degree of regularity is to run F_4 or F_5 on that system. A method to compute – or more closely estimate – this degree without computing a full Gröbner basis would thus increase confidence in a cipher's parameters.
- The polynomial description of some AOCs, e.g., MiMC's, already forms a Gröbner basis, albeit not in *lex* order. The main difficulty in breaking the cipher lies then in extracting the polynomial univariate in the key. This is usually done by performing a term order change, potentially with early abort. The theoretical complexity of changing a term order is poorly understood. Current security arguments rely on the complexity of existing algorithms. While FGLM and Sparse FGLM are well understood, it is unknown whether their complexities are optimal, especially since the Gröbner Walk is reported to outperform FGLM sometimes.
- Generating the polynomial model for the AOC under attack is out of scope for this paper. Regardless, there are interesting open questions associated with that step.
  For example, given a single pair of plaintext & ciphertext – or a single hash digest – the resulting polynomial system is determined, which suffices for the attack strategy described in Section 1. Using multiple pairs of plaintext & ciphertext results in an overdetermined system. Intuitively, this added information should decrease an attack's complexity, implying that finding solutions for the overdetermined system should be easier. However, it is

[19] Knowing the fan implies knowing all reduced Gröbner bases – a term order change becomes superfluous.

unclear whether this is generally the case, or how to exploit the additional information in other ways.
Similarly, intuition suggests that finding a collision for a hash function should be easier than computing a (second) preimage. How to turn this intuition into a polynomial model is unclear.

- In a Gröbner basis attack on an AOC, we are only interested in the polynomial's common solutions that are elements of base field $\mathbb{F}$. One way to ensure the absence of parasitical solutions, i.e., the exclusion of elements from $\overline{\mathbb{F}} \setminus \mathbb{F}$, is to add the field equations to the polynomial system. The solutions of field equation $x^p - x$ for variable x, where $p = |\mathbb{F}|$, are exactly the elements of $\mathbb{F}$. Unfortunately, the field equations are of very high degree, which is detrimental for a Gröbner basis algorithm's runtime if the field size is of cryptographic relevance [31, Sec. 2.1]. By introducing additional variables, we can "decompose" a field equation into multiple equations, each of lower degree. It is unknown whether such decompositions lead to faster Gröbner basis computations, and if so, what the optimal trade-off between the new equations' degrees and the number of new variables is.
- Like any modern symmetric primitive, AOCs are iterated ciphers. Thus, the derived system of polynomials inherently contains sets of structurally similar polynomials. Current Gröbner basis attacks against AOCs do not exploit such similarities because it is not obvious how to do so, if it is possible at all.
- The polynomials processed by F_4 and F_5 generally densify during execution. Modifications allowing application of sparse linear algebra methods would result in faster algorithms both asymptotically and for current problem sizes.
- Given a basis for a polynomial system's syzygy submodule, F_5 performs no reduction to zero. Thus, an AOC's security can be characterized by the "non-triviality" of its syzygy space. The most efficient currently known method of finding a syzygy basis is to run F_5, where it is computed as a byproduct. Some faster method to lower-bound the size of – or directly compute – the syzygy basis and the non-triviality of the involved elements could thus be used to increase confidence in a cipher's parameters.
- For some ciphers, security against Gröbner basis attacks is argued by extrapolating the degree of regularity from round-reduced variants [7, 81]. It is unknown whether this extrapolation accurately predicts the degree of regularity of the full-round primitive.

Acknowledgments

This work was sponsored by the Ethereum Foundation.

References

1. Albrecht, M., Perry, J.: F4/5. arXiv preprint arXiv:1006.4933 (2010)
2. Albrecht, M.R., Cid, C., Faugère, J.C., Perret, L.: On the relation between the MXL family of algorithms and Gröbner basis algorithms. Journal of Symbolic Computation **47**(8), 926–941 (2012)

3. Albrecht, M.R., Cid, C., Grassi, L., Khovratovich, D., Lüftenegger, R., Rechberger, C., Schofnegger, M.: Algebraic cryptanalysis of STARK-friendly designs: Application to MARVELlous and MiMC. In: Asiacrypt. pp. 371–397. Springer (2019)
4. Albrecht, M.R., Grassi, L., Perrin, L., Ramacher, S., Rechberger, C., Rotaru, D., Roy, A., Schofnegger, M.: Feistel structures for MPC, and more. In: ESORICS. pp. 151–171. Springer (2019)
5. Albrecht, M.R., Grassi, L., Rechberger, C., Roy, A., Tiessen, T.: MiMC: Efficient encryption and cryptographic hashing with minimal multiplicative complexity. In: Asiacrypt. pp. 191–219. Springer (2016)
6. Albrecht, M.R., Rechberger, C., Schneider, T., Tiessen, T., Zohner, M.: Ciphers for MPC and FHE. In: EuroCrypt. pp. 430–454. Springer (2015)
7. Aly, A., Ashur, T., Ben-Sasson, E., Dhooghe, S., Szepieniec, A.: Design of symmetric-key primitives for advanced cryptographic protocols. IACR ToSC **2020**(3), 1–45 (2020)
8. Amrhein, B., Gloor, O., Küchlin, W.: On the walk. Theoretical Computer Science **187**(1-2), 179–202 (1997)
9. Ars, G., Faugère, J.C., Imai, H., Kawazoe, M., Sugita, M.: Comparison between XL and Gröbner basis algorithms. In: Asiacrypt. pp. 338–353. Springer (2004)
10. Ashur, T., Dhooghe, S.: MARVELlous: a STARK-friendly family of cryptographic primitives. Cryptology ePrint Archive **2018**, 1098 (2018)
11. Bardet, M.: Étude des systèmes algébriques surdéterminés. Applications aux codes correcteurs et à la cryptographie. Ph.D. thesis, Université Pierre et Marie Curie-Paris VI (2004)
12. Bardet, M., Faugère, J.C., Salvy, B.: On the complexity of Gröbner basis computation of semi-regular overdetermined algebraic equations. In: International Conference on Polynomial System Solving. pp. 71–74 (2004)
13. Bardet, M., Faugère, J.C., Salvy, B.: On the complexity of the F_5 Gröbner basis algorithm. Journal of Symbolic Computation **70**, 49–70 (2015)
14. Bardet, M., Faugère, J.C., Salvy, B., Yang, B.Y.: Asymptotic behaviour of the degree of regularity of semi-regular polynomial systems. In: Proceedings of MEGA. vol. 5 (2005)
15. Bayer, D., Stillman, M.: A theorem on refining division orders by the reverse lexicographic order. Duke Mathematical Journal **55**(2), 321–328 (1987)
16. Ben-Sasson, E., Bentov, I., Horesh, Y., Riabzev, M.: Scalable, transparent, and post-quantum secure computational integrity. Cryptology ePrint Archive **2018**, 46 (2018)
17. Berlekamp, E.R.: Factoring polynomials over finite fields. Bell System Technical Journal **46**(8), 1853–1859 (1967)
18. Beyne, T., Canteaut, A., Dinur, I., Eichlseder, M., Leander, G., Leurent, G., Naya-Plasencia, M., Perrin, L., Sasaki, Y., Todo, Y., Wiemer, F.: Out of oddity – new cryptanalytic techniques against symmetric primitives optimized for integrity proof systems. In: CRYPTO. LNCS, vol. 12172, pp. 299–328. Springer (2020)
19. Brickenstein, M.: Slimgb: Gröbner bases with slim polynomials (2005)
20. Buchberger, B.: Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal. PhD thesis, Universität Innsbruck (1965)
21. Buchberger, B.: A criterion for detecting unnecessary reductions in the construction of Gröbner-bases. In: EUROSAM. pp. 3–21. Springer (1979)
22. Buchberger, B.: Gröbner bases: An algorithmic method in polynomial ideal theory. Multidimensional Systems Theory (1985)

23. Buchberger, B.: Bruno Buchberger's PhD thesis 1965: An algorithm for finding the basis elements of the residue class ring of a zero dimensional polynomial ideal. Journal of Symbolic Computation **41**(3-4), 475–511 (2006)
24. Buchmann, J., Cabarcas, D., Ding, J., Mohamed, M.S.E.: Flexible partial enlargement to accelerate Gröbner basis computation over $\mathbb{F}_2$. In: AFRICACRYPT. pp. 69–81. Springer (2010)
25. Buchmann, J., Pyshkin, A., Weinmann, R.P.: Block ciphers sensitive to Gröbner basis attacks. In: CT-RSA. pp. 313–331. Springer (2006)
26. Buchmann, J.A., Ding, J., Mohamed, M.S.E., Mohamed, W.S.A.E.: MutantXL: Solving multivariate polynomial equations for cryptanalysis. In: Dagstuhl Seminar Proceedings. Schloss Dagstuhl-Leibniz-Zentrum für Informatik (2009)
27. Bünz, B., Bootle, J., Boneh, D., Poelstra, A., Wuille, P., Maxwell, G.: Bulletproofs: Short proofs for confidential transactions and more. In: IEEE Symposium on Security and Privacy. pp. 315–334. IEEE (2018)
28. Caboara, M.: A dynamic algorithm for Gröbner basis computation. In: ISSAC. pp. 275–283 (1993)
29. Caboara, M., Kreuzer, M., Robbiano, L.: Minimal sets of critical pairs. In: Mathematical Software, pp. 390–404. World Scientific (2002)
30. Caboara, M., Perry, J.: Reducing the size and number of linear programs in a dynamic Gröbner basis algorithm. Applicable Algebra in Engineering, Communication and Computing **25**(1-2), 99–117 (2014)
31. Caminata, A., Gorla, E.: Solving multivariate polynomial systems and an invariant from commutative algebra. Cryptology ePrint Archive (2017)
32. Cantor, D.G., Zassenhaus, H.: A new algorithm for factoring polynomials over finite fields. Mathematics of Computation pp. 587–592 (1981)
33. Collart, S., Kalkbrener, M., Mall, D.: Converting bases with the Gröbner walk. Journal of Symbolic Computation **24**(3-4), 465–469 (1997)
34. Coppersmith, D.: Solving homogeneous linear equations over gf(2) via block Wiedemann algorithm. Mathematics of Computation **62**(205), 333–350 (1994)
35. Courtois, N., Klimov, A., Patarin, J., Shamir, A.: Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In: EuroCrypt. pp. 392–407. Springer (2000)
36. Cox, D.A., Little, J., O'Shea, D.: Using Algebraic Geometry, vol. 185. Springer Science & Business Media (2006)
37. Cox, D.A., Little, J., O'Shea, D.: Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra. Springer Science & Business Media (2013)
38. Damgård, I., Pastro, V., Smart, N., Zakarias, S.: Multiparty computation from somewhat homomorphic encryption. In: CRYPTO. pp. 643–662. Springer (2012)
39. Ding, J., Buchmann, J., Mohamed, M.S.E., Mohamed, W.S.A.E., Weinmann, R.P.: MutantXL (2008)
40. Ding, J., Schmidt, D.: Solving degree and degree of regularity for polynomial systems over a finite fields. In: Number Theory and Cryptography, pp. 34–49. Springer (2013)
41. Ding, J., Zhang, Z., Deaton, J.: How much can f5 really do? Cryptology ePrint Archive **2021**, 51 (2021)
42. Dubois, V., Gama, N.: The degree of regularity of HFE systems. In: Asiacrypt. pp. 557–576. Springer (2010)
43. Dubé, T.W.: The structure of polynomial ideals and Gröbner bases. SIAM Journal on Computing **19**(4), 750–773 (1990)

44. Eder, C., Faugère, J.C.: A survey on signature-based algorithms for computing Gröbner bases. Journal of Symbolic Computation **80**, 719–784 (2017)
45. Eder, C., Roune, B.H.: Signature rewriting in Gröbner basis computation. In: ISSAC. pp. 331–338 (2013)
46. Faugère, J.C.: A new efficient algorithm for computing Gröbner bases (F_4). Journal of Pure and Applied Algebra **139**(1-3), 61–88 (1999)
47. Faugère, J.C.: A new efficient algorithm for computing Gröbner bases without reduction to zero (F_5). In: ISSAC. pp. 75–83 (2002)
48. Faugère, J.C., Gianni, P., Lazard, D., Mora, T.: Efficient computation of zero-dimensional Gröbner bases by change of ordering. Journal of Symbolic Computation **16**(4), 329–344 (1993)
49. Faugère, J.C., Mou, C.: Sparse FGLM algorithms. Journal of Symbolic Computation **80**, 538–569 (2017)
50. Fukuda, K., Jensen, A., Thomas, R.: Computing Gröbner fans. Mathematics of Computation **76**(260), 2189–2212 (2007)
51. Fukuda, K., Jensen, A.N., Lauritzen, N., Thomas, R.: The generic Gröbner walk. Journal of Symbolic Computation **42**(3), 298–312 (2007)
52. Gao, S., Volny IV, F., Wang, M.: A new algorithm for computing Gröbner bases. Cryptology ePrint Archive (2010)
53. Gao, S., Volny IV, F., Wang, M.: A new algorithm for computing Gröbner bases (rev. 2013) (2013)
54. von zur Gathen, J., Gerhard, J.: Modern Computer Algebra. Cambridge university press (2013)
55. Gebauer, R., Möller, H.M.: On an installation of Buchberger's algorithm. Journal of Symbolic Computation **6**(2-3), 275–286 (1988)
56. Gennaro, R., Gentry, C., Parno, B., Raykova, M.: Quadratic span programs and succinct NIZKs without PCPs. In: EuroCrypt. pp. 626–645. Springer (2013)
57. Giovini, A., Mora, T., Niesi, G., Robbiano, L., Traverso, C.: "One sugar cube, please" or selection strategies in the Buchberger algorithm. In: ISSAC. pp. 49–54 (1991)
58. Grassi, L., Khovratovich, D., Rechberger, C., Roy, A., Schofnegger, M.: Poseidon: A new hash function for zero-knowledge proof systems. In: USENIX Security. USENIX Association (2020)
59. Greuel, G.M., Pfister, G.: A Singular Introduction to Commutative Algebra. Springer Science & Business Media (2012)
60. Hartshorne, R.: Algebraic Geometry, vol. 52. Springer Science & Business Media (2013)
61. Hashemi, A., Lazard, D.: Sharper complexity bounds for zero-dimensional Gröbner bases and polynomial system solving. International Journal of Algebra and Computation **21**(05), 703–713 (2011)
62. Hilbert, D.: Über die Theorie der algebraischen Formen. Mathematische Annalen **36**(4), 473–534 (1890)
63. Hong, H., Perry, J.: Are Buchberger's criteria necessary for the chain condition? Journal of Symbolic Computation **42**(7), 717–732 (2007)
64. Hyun, S.G., Neiger, V., Rahkooy, H., Schost, É.: Block-Krylov techniques in the context of sparse-FGLM algorithms. Journal of Symbolic Computation **98**, 163–191 (2020)
65. Ito, T., Shinohara, N., Uchiyama, S.: An efficient F_4-style based algorithm to solve MQ problems. In: International Workshop on Security. pp. 37–52. Springer (2019)
66. Jakobsen, T., Knudsen, L.R.: The interpolation attack on block ciphers. In: FSE. pp. 28–40. Springer (1997)

67. Kaltofen, E., Shoup, V.: Subquadratic-time factoring of polynomials over finite fields. Mathematics of Computation **67**(223), 1179–1197 (1998)
68. Kedlaya, K.S., Umans, C.: Fast modular composition in any characteristic. In: FOCS. pp. 146–155. IEEE (2008)
69. Lazard, D.: Gröbner bases, Gaussian elimination and resolution of systems of algebraic equations. In: EUROCAL. pp. 146–156. Springer (1983)
70. Macaulay, F.S.: Some formulæ in elimination. Proceedings of the London Mathematical Society **1**(1), 3–27 (1902)
71. Marinari, M.G., Moeller, H.M., Mora, T.: Gröbner bases of ideals defined by functionals with an application to ideals of projective points. Applicable Algebra in Engineering, Communication and Computing **4**(2), 103–145 (1993)
72. Mayr, E.W., Meyer, A.R.: The complexity of the word problems for commutative semigroups and polynomial ideals. Advances in Mathematics **46**(3), 305–329 (1982)
73. Mayr, E.W., Ritscher, S.: Dimension-dependent bounds for Gröbner bases of polynomial ideals. Journal of Symbolic Computation **49**, 78–94 (2013)
74. Mohamed, M.S.E., Cabarcas, D., Ding, J., Buchmann, J., Bulygin, S.: MXL_3: An efficient algorithm for computing Gröbner bases of zero-dimensional ideals. In: ICISC. pp. 87–100. Springer (2009)
75. Mohamed, M.S.E., Mohamed, W.S.A.E., Ding, J., Buchmann, J.: MXL2: Solving polynomial equations over GF(2) using an improved Mutant strategy. In: International Workshop on Post-Quantum Cryptography. pp. 203–215. Springer (2008)
76. Mohamed, W.S.A.: Improvements for the XL Algorithm with Applications to Algebraic Cryptanalysis. Ph.D. thesis, Technische Universität Darmstadt, Darmstadt (2011)
77. Mohamed, W.S.A., Ding, J., Kleinjung, T., Bulygin, S., Buchmann, J.: PWXL: A parallel Wiedemann-XL algorithm for solving polynomial equations over GF(2). In: Conference on Symbolic Computation and Cryptography. p. 89. Citeseer (2010)
78. Mora, T., Robbiano, L.: The Gröbner fan of an ideal. Journal of Symbolic Computation **6**(2-3), 183–208 (1988)
79. Möller, H.M., Mora, T., Traverso, C.: Gröbner bases computation using syzygies. In: ISSAC. pp. 320–328 (1992)
80. Perry, J.: Exploring the dynamic Buchberger algorithm. In: ISSAC. pp. 365–372 (2017)
81. Szepieniec, A., Ashur, T., Dhooghe, S.: Rescue-Prime: a standard specification (SoK). Cryptology ePrint Archive, Report 2020/1143 (2020)
82. Tran, Q.N.: A fast algorithm for Gröbner basis conversion and its applications. Journal of Symbolic Computation **30**(4), 451–467 (2000)
83. Wiedemann, D.: Solving sparse linear equations over finite fields. IEEE Transactions on Information Theory **32**(1), 54–62 (1986)
84. Yap, C.K.: A new lower bound construction for commutative Thue systems with applications. Journal of Symbolic Computation **12**(1), 1–27 (1991)
85. Yeh, J.Y.C., Cheng, C.M., Yang, B.Y.: Operating degrees for XL vs. F_4/F_5 for generic $\mathcal{MQ}$ with number of equations linear in that of variables. In: Number Theory and Cryptography, pp. 19–33. Springer (2013)

A Notation

In Table 3, we briefly summarize the notation used in this document.

Table 3: Notation.

symbol	expansion	explanation
$\mathbb{F}$		field, typically of prime order
$\overline{\mathbb{F}}$		algebraic closure of $\mathbb{F}$
$\mathbb{F}[\mathbf{x}]$	$\mathbb{F}[x_0, \ldots, x_{n-1}]$	multivariate polynomial ring
x_i		variable
$\mathbf{x}$	$(x_0, \ldots, x_{n-1})$	vector of variables
$\boldsymbol{\alpha}$	$(\alpha_0, \ldots, \alpha_{n-1})$	exponent vector
$\mathbf{x}^{\boldsymbol{\alpha}}$	$(x_0^{\alpha_0}, \ldots, x_{n-1}^{\alpha_{n-1}})$	monomial
$\deg(\mathbf{x}^{\boldsymbol{\alpha}})$	$\sum_i \alpha_i$	total degree of $\mathbf{x}^{\boldsymbol{\alpha}}$
$c_{\boldsymbol{\alpha}}\mathbf{x}^{\boldsymbol{\alpha}}$		term
f	$\sum_{\boldsymbol{\alpha}} c_{\boldsymbol{\alpha}}\mathbf{x}^{\boldsymbol{\alpha}}$	polynomial
$f \xrightarrow[G]{} r$		full reduction of f by G
$\mathcal{M}$		set of all monomials
$\mathcal{M}(f)$	$\{\mathbf{x}^{\boldsymbol{\alpha}} \mid c_{\boldsymbol{\alpha}} \neq 0\}$	monomials occurring in f
$\mathcal{M}(G)$	$\bigcup_{g \in G} \mathcal{M}(g)$	extension of $\mathcal{M}(f)$ to sets
$\mathcal{T}$		set of all terms
$\prec$		monomial order on $\mathcal{M}$
$\preccurlyeq$		extension of $\prec$ to $\mathbb{F}[\mathbf{x}]^s$
$\mathsf{lc}(f)$		leading coefficient of f
$\mathsf{lm}(f)$	$\max_{\prec}\{\mathbf{x}^{\boldsymbol{\alpha}} \in \mathcal{M}(f) \mid c_{\boldsymbol{\alpha}} \neq 0\}$	leading monomial of f
$\mathsf{lt}(f)$	$\mathsf{lc}(f) \cdot \mathsf{lm}(f)$	leading term of f
$\mathsf{lf}(f)$		leading form of f
$\mathsf{lc}(G)$	$\bigcup_{g \in G} \mathsf{lc}(g)$	extension of lc to sets
$\mathsf{lm}(G)$	$\bigcup_{g \in G} \mathsf{lm}(g)$	extension of lm to sets
$\mathsf{lt}(G)$	$\bigcup_{g \in G} \mathsf{lt}(g)$	extension of lt to sets
$\mathsf{lf}(G)$	$\bigcup_{g \in G} \mathsf{lf}(g)$	extension of lf to sets
I		polynomial ideal
$\langle f_0, \ldots, f_{s-1}\rangle$	$\{\sum_i p_i f_i \mid p_i \in \mathbb{F}[\mathbf{x}]\}$	ideal spanned by the f_i
$f = 0$		polynomial equation of f
$V(I)$	$\{\boldsymbol{a} \in \overline{\mathbb{F}}^n \mid \forall f \in I : f(\boldsymbol{a}) = 0\}$	affine variety of ideal I
$\boldsymbol{e}_i$	$(0, \ldots, 0, 1, 0, \ldots, 0)$	i-th unit vector
$\phi_{\mathcal{F}}(\boldsymbol{g})$	$\sum_i g_i f_i$ for $f_i \in \mathcal{F}$	map from $\mathbb{F}[\mathbf{x}]^s$ to $\mathbb{F}[\mathbf{x}]$
$\mathfrak{s}(\boldsymbol{g})$	$\mathsf{lm}_{\preccurlyeq}(\boldsymbol{g})$	signature vector of $\boldsymbol{g} \in \mathbb{F}[\mathbf{x}]^s$

B Multivariate Polynomial Division

We restate the multivariate division algorithm of [37, Ch. 2, §3] in Algorithm 10.

Algorithm 10: MULTIVARIATE POLYNOMIAL DIVISION

```
   Input: polynomials $f, g_0, \ldots, g_{t-1} \in \mathbb{F}[\mathbf{x}]$
   Output: polynomials $q_0, \ldots, q_{t-1}, r \in \mathbb{F}[\mathbf{x}]$ such that $f = \sum_{i=0}^{t-1} q_i g_i + r$
1  $q_0 = \ldots = q_{t-1} = r = 0$
2  $p = f$
3  while $p \neq 0$ do
4      $i = 0$
5      division_occured = False
6      while $i < t$ and not division_occured do
7          if $\mathsf{lt}(g_i) \mid \mathsf{lt}(p)$ then
8              $q_i = q_i + \frac{\mathsf{lt}(p)}{\mathsf{lt}(g_i)}$
9              $p = p - \left(\frac{\mathsf{lt}(p)}{\mathsf{lt}(g_i)}\right) g_i$
10             division_occured = True
11         else
12             $i = i + 1$
13     if not division_occured then
14         $r = r + \mathsf{lt}(p)$
15         $p = p - \mathsf{lt}(p)$
16 return $q_0, \ldots, q_{t-1}, r$
```

C Regular Signature Reduction

In Algorithm 11, we describe regular $\mathfrak{s}$-reduction of a vector of origin $\boldsymbol{f}$ by a set of vectors of origin $\boldsymbol{g}_0, \dots, \boldsymbol{g}_{t-1}$, as given in [37, Ch. 10, §4]. The F_5 algorithm of Section 3.2 uses regular $\mathfrak{s}$-reduction as a subroutine. Note the structural similarities to multivariate polynomial division of Algorithm 10.

By changing the second condition in Line 8 to require equality of signatures, the $\mathfrak{s}$-reduction becomes singular as opposed to regular. "Plain" $\mathfrak{s}$-reduction, being neither regular nor singular, can be achieved by modifying the same condition to only disallow all signatures $\succcurlyeq$-greater than $\mathfrak{s}(\boldsymbol{r})$, i.e., allowing both regular and singular $\mathfrak{s}$-reduction steps.

Algorithm 11: REGULAR $\mathfrak{s}$-REDUCTION

```
   Input: vectors of origin $\boldsymbol{f}, \boldsymbol{g}_0, \dots, \boldsymbol{g}_{t-1} \in \mathbb{F}[\mathbf{x}]^s$
   Output: vector of origin $\boldsymbol{r} \in \mathbb{F}[\mathbf{x}]^s$
1  $\boldsymbol{r} = \boldsymbol{f}$
2  poly_r = 0
3  while $\phi(\boldsymbol{r}) \neq$ poly_r do
4      $m = \mathsf{lt}(\phi(\boldsymbol{r}) -$ poly_r$)$
5      $i = 0$
6      reduction_occured = False
7      while $i < t$ and not reduction_occured do
8          if $\mathsf{lt}(\phi(\boldsymbol{g}_i)) \mid m$ and $\mathfrak{s}\big(\frac{m}{\mathsf{lt}(\phi(\boldsymbol{g}_i))}\boldsymbol{g}_i\big) \preccurlyeq \mathfrak{s}(\boldsymbol{r})$ then
9              $\boldsymbol{r} = \boldsymbol{r} - \frac{m}{\mathsf{lt}(\phi(\boldsymbol{g}_i))}\boldsymbol{g}_i$
10             reduction_occured = True
11         else
12             $i = i + 1$
13     if not reduction_occured then
14         poly_r = poly_r + m
15 return $\boldsymbol{r}$
```