Polynomial XL: A Variant of the XL Algorithm Using Macaulay Matrices over Polynomial Rings

Hiroki Furue and Momonari Kudo

Department of Mathematical Informatics,
Graduate School of Information Science and Technology,
The University of Tokyo

Abstract

Solving a system of m multivariate quadratic equations in n variables (the $\mathcal{MQ}$ problem) is one of the main challenges of algebraic cryptanalysis. The XL algorithm (XL for short) is a major approach for solving the $\mathcal{MQ}$ problem with linearization over a coefficient field. Furthermore, the hybrid approach with XL (h-XL) is a variant of XL guessing some variables beforehand. In this paper, we present a variant of h-XL, which we call the *polynomial XL (PXL)*. In PXL, the whole n variables are divided into k variables to be fixed and the remaining $n-k$ variables as "main variables", and we generate the Macaulay matrix with respect to the $n-k$ main variables over a polynomial ring of the k variables. By eliminating some columns of the Macaulay matrix over the polynomial ring before guessing k variables, the amount of manipulations required for each guessed value can be reduced. Our complexity analysis indicates that PXL is efficient on the system with $n \approx m$. For example, on systems over $\mathbb{F}_{2^8}$ with $n = m = 80$, the number of manipulations required by the hybrid approaches with XL and Wiedemann XL and PXL is estimated as 2^{252}, 2^{234}, and 2^{220}, respectively.

1 Introduction

Solving a system over a finite field is one of the most major problems in the field of computer science. Especially, the problem of solving a quadratic system (the $\mathcal{MQ}$ problem) is used to construct various cryptographic systems. Multivariate public key cryptography (MPKC), which is based on the

difficulty of solving the $\mathcal{MQ}$ problem, is expected to be secure against quantum computer attacks from the NP-completeness of the $\mathcal{MQ}$ problem [18]. Indeed, multivariate signature schemes such as Rainbow [13] and GeMSS [9] have been selected in the third round of the NIST post-quantum cryptography standardization project [23]. The security of MPKCs strongly depends on how efficiently the $\mathcal{MQ}$ problem can be solved.

In this research, given a quadratic polynomial system $\mathcal{F} = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))$ in n variables $\mathbf{x} = (x_1, \ldots, x_n)$ over a finite field $\mathbb{F}_q$ of q elements, we aim to find a solution to $\mathcal{F}(\mathbf{x}) = \mathbf{0} \in \mathbb{F}_q^m$. Furthermore, throughout the rest of this paper, we deal with only the case of $n \leq m$ (*overdetermined*). Note that algorithms solving the overdetermined $\mathcal{MQ}$ problem can be easily applied to the case of $n > m$, since, after $n - m$ variables are randomly specified, the resulting system will have a solution on average.

Among various methods for solving algebraic systems (e.g., resultant based method [11, Chapter 3] and Wu's method [29]), Gröbnor basis methods are main categories solving the $\mathcal{MQ}$ problem. Buchberger's algorithm [8] is a first major method of computing a Gröbnor basis of the ideal spanned by a given system. In 1999, Faugère proposed an efficient variant of Buchberger's algorithm which is called F4 algorithm [15], and he published the even more efficient F5 algorithm in 2002 [16]. F4 and F5 are todays major algorithms for obtaining a Gröbnor basis, and solutions of a system can be easily derived from a Gröbnor basis ([12, Chapter 3]).

Another strategy to solve the $\mathcal{MQ}$ problem is linearization, and the most basic linearization-based algorithm is the *XL algorithm* proposed by Courtois et al. [10]. The XL algorithm is an extension of Relinearization algorithm [19], and its idea is very simple; linearizing the given system by regarding each monomial as one variable. To obtain a solution, we need to make the number of independent equations close to the total number of monomials. For this, we multiply every polynomial of the given system by every monomial with degree smaller than or equal to a certain value. We then regard the obtained system as a linear system and generate a matrix each entry of which corresponds to a coefficient in an equation of the system (this matrix is called *Macaulay matrix*). If sufficient number of equations are prepared, then a univariate equation is obtained by performing Gaussian elimination on the Macaulay matrix. We then solve the obtained univariate equation and repeat such processes with respect to the remaining variables. Note that XL is considered to be a redundant variant of F4 algorithm (see [1, 2] for details). Furthermore, Yang et al. [27] analyzed a variant of the XL algorithm called *Wiedemann XL (WXL)*, which adopts Wiedemann's algorithm [24] instead of the Gaussian elimination in the XL

framework. WXL provides another complexity estimate which is used to estimate the security of various MPKCs such as Rainbow [13].

The hybrid approach [5, 26] (first proposed as FXL, in which the "F" stands for "fix") is proposed as an approach applying an $\mathcal{MQ}$ solver such as F4, F5, or XL efficiently, which mixes exhaustive search and an $\mathcal{MQ}$ solver. In the hybrid approach, k variables are fixed, and the remaining system in $n-k$ variables are solved by an $\mathcal{MQ}$ solver. These processes are iterated until a solution is found. In the case of $n \approx m$, the hybrid approach may be effective, since the gain obtained by working on systems with less variables may overcome the loss due to the exhaustive search on the fixed variables. In this paper, we call the hybrid approach with XL (resp. WXL) *h-XL* (resp. *h-WXL*).

Our contributions

In this paper, we propose a new variant of the XL algorithm, which we call the *polynomial XL* (PXL). PXL is constructed by improving h-XL. For the $\mathcal{MQ}$ system $\mathcal{F}$ of m equations in n variables $\mathbf{x} = (x_1, \ldots, x_n)$ over $\mathbb{F}_q$, PXL first sets the number k of guessed variables as in the hybrid approach. We then regard the given system as a system in $n-k$ variables $x_{k+1}, \ldots, x_n$, whose coefficients belong to the polynomial ring $\mathbb{F}_q[x_1, \ldots, x_k]$. We multiply every polynomial of $\mathcal{F}$ by every monomial in the $n-k$ variables with degree smaller than or equal to a certain degree and generates the Macaulay matrix over $\mathbb{F}_q[x_1, \ldots, x_k]$. The main idea of PXL is to partly perform Gaussian elimination on the matrix over the polynomial ring before fixing the k variables and complete the elimination after fixing. By doing so, we can reduce the amount of manipulations for each guessed value compared with h-XL, since the size of the un-eliminated part of the matrix after the partial Gaussian elimination is much smaller than that of the original one. This enables us to solve the system more efficiently for some parameters.

We here briefly explain why such an elimination on the Macaulay matrix over the polynomial ring is possible (see Section 3 for details). If the given system in n variables is seen as the system in $n-k$ variables over $\mathbb{F}_q[x_1, \ldots, x_k]$, then the coefficients of quadratic terms in the $n-k$ variables can be seen as elements of $\mathbb{F}_q$ since the given system is originally quadratic. For the same reason, in every polynomial obtained by multiplying monomials to polynomials of $\mathcal{F}$, the coefficients of the highest degree part do not include the k variables $x_1, \ldots, x_k$. This indicates that some submatrices of the Macaulay matrix over $\mathbb{F}_q[x_1, \ldots, x_k]$ can be seen as matrices over the finite field $\mathbb{F}_q$. By utilizing this structure, we can partly perform Gaussian

elimination on the Macaulay matrix before fixing the k variables.

In this paper, after describing the proposed algorithm, we also discuss the estimation of the time and space complexities and compare them with those of h-XL and h-WXL. Comparing the time complexities, we show that the proposed algorithm is more efficient in the case of $n \approx m$. For example, on the system over $\mathbb{F}_{2^8}$ with $n = m = 80$, the number of manipulations in $\mathbb{F}_q$ required by h-WXL and PXL is estimated as 2^{234} and 2^{220}, respectively. On the other hand, in terms of the space complexity, the proposed algorithm is not well compared to h-WXL since the sparsity of the Macaulay matrix is not maintained through the proposed algorithm. Therefore, the relationship between PXL and h-WXL can be seen as a trade-off between time and memory.

Finally, we discuss the relationship between the proposed algorithm and XFL [10, 25] proposed as a variant of FXL. In XFL, the k variables to be fixed are chosen firstly, and monomials including only the remaining $n - k$ variables are eliminated before fixing. XFL is similar to PXL in the point that the equations are partly eliminated before fixing. However, PXL eliminates some monomials not only over the field but also over the polynomial ring of the k variables. Therefore, PXL can be regarded as the extension of XFL.

Organizations

The rest of this paper is organized as follows: Section 2 reviews the XL algorithm and the hybrid approach. Section 3 is devoted to describing the proposed algorithm PXL. We estimate the time complexity and compare it with those of h-XL and h-WXL in Section 4. Finally, Section 5 is devoted to the conclusion, where we summarize the key points and suggest possible future works.

2 Preliminaries

In this section, we recall the definition of the XL algorithm [10], and discuss its solving degree and its complexity. (The correctness of XL will be discribed in Appendix B.) We also explain the hybrid approach, wich is a way of combining an exhaustive search with an $\mathcal{MQ}$ solver such as XL.

2.1 Notation

We first fix the notations that are used in the rest of this section. Let $R[x] = R[x_1, \ldots, x_n]$ denote the polynomial ring with n variables over a commutative ring R with unity. For a subset $S \subset R[x]$, the ideal of $R[x]$ generated by S is denoted by $\langle S \rangle$. In particular, when S is a finite set $\{f_1, \ldots, f_m\}$, we denote it by $\langle f_1, \ldots, f_m \rangle$. Let $[x]$ denote the set of all monomials in $R[x]$, say

$$[x] := \{x_1^{\alpha_1} \cdots x_n^{\alpha_n} : (\alpha_1, \ldots, \alpha_n) \in (\mathbb{Z}_{\geq 0})^n\}.$$

For each $d \geq 0$, we also denote by T_d (resp. $T_{\leq d}$) the set of all monomials in $R[x]$ of degree d (resp. $\leq d$), namely we set

$$T_d = \left\{ x_1^{\alpha_1} \cdots x_n^{\alpha_n} \in [x] : \sum_{k=1}^{n} \alpha_k = d \right\}, \tag{2.1}$$

$$T_{\leq d} = \left\{ x_1^{\alpha_1} \cdots x_n^{\alpha_n} \in [x] : \sum_{k=1}^{n} \alpha_k \leq d \right\}. \tag{2.2}$$

For a subset $F \subset R[x]$, we set $[x] \cdot F := \{t \cdot f : t \in [x],\ f \in F\}$. A finite subset of $[x] \cdot F$ is called a *shift* of F. For a polynomial $f \in R[x]$ and a monomial $t \in [x]$, we denote by $\mathrm{coeff}(f, t)$ the coefficient of t in f. The *supporting set* $\mathrm{supp}(f)$ of f is defined as the set of monomials whose coefficients in f are not zero, say

$$\mathrm{supp}(f) := \{t \in [x] : \mathrm{coeff}(f, t) \neq 0\}.$$

In the following, we fix a monomial order $\succ$ on $[x]$, and we order elements of any subset of $[x]$ by $\succ$. For a non-zero polynomial $f \in R[x] \setminus \{0\}$, we denote by $\mathrm{LT}(f)$, $\mathrm{LM}(f)$, $\mathrm{LC}(f)$ and $\mathrm{mltdeg}(f)$ the leading term, the leading monomial, the leading coefficient and the multi-degree of f with respect to $\succ$, respectively. Note that $\mathrm{LT}(f) = \mathrm{LC}(f) \cdot \mathrm{LM}(f)$. For a subset $F \subset R[x]$, we set $\mathrm{LT}(F) := \{\mathrm{LT}(f) : f \in F\}$ and $\mathrm{LM}(F) := \{\mathrm{LM}(f) : f \in F\}$. Recall that a finite subset $G \subset R[x]$ is called a *Gröbner basis* for an ideal $I \subset R[x]$ with respect to $\succ$ if it generates I and if $\langle \mathrm{LT}(G) \rangle = \langle \mathrm{LT}(I) \rangle$. It is well-known that every non-zero ideal of $R[x]$ has a Gröbner basis.

2.2 Macaulay matrices

For a finite subset $F = \{f_1, \ldots, f_m\} \subset R[x]$ of ordered polynomials and a finite subset $T = \{t_1, \ldots, t_\ell\} \subset [x]$ with $t_1 \succ \cdots \succ t_\ell$, we define the

Macaulay matrix of F with respect to T by an $(m \times \ell)$-matrix over R whose (i,j)-entry is the coefficient of t_j in f_i, and denote it by $\mathrm{Mac}(F,T)$, say

$$\mathrm{Mac}(F,T) := \begin{array}{c} \\ f_1 \\ f_2 \\ \vdots \\ f_m \end{array} \begin{array}{c} \begin{array}{cccc} t_1 & t_2 & \cdots & t_\ell \end{array} \\ \begin{pmatrix} \mathrm{coeff}(f_1,t_1) & \mathrm{coeff}(f_1,t_2) & \cdots & \mathrm{coeff}(f_1,t_\ell) \\ \mathrm{coeff}(f_2,t_1) & \mathrm{coeff}(f_2,t_2) & \cdots & \mathrm{coeff}(f_2,t_\ell) \\ \vdots & \vdots & & \vdots \\ \mathrm{coeff}(f_m,t_1) & \mathrm{coeff}(f_m,t_2) & \cdots & \mathrm{coeff}(f_m,t_\ell) \end{pmatrix} \end{array}.$$

Conversely, for an $(m \times \ell)$-matrix A over R and T given as above, a unique list F' of polynomials in $R[x]$ such that $\mathrm{Mac}(F',T) = A$ is denoted by $\mathrm{Mac}^{-1}(A,T)$, namely, for

$$A = \begin{array}{c} \\ 1 \\ 2 \\ \vdots \\ m \end{array} \begin{array}{c} \begin{array}{cccc} t_1 & t_2 & \cdots & t_\ell \end{array} \\ \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,\ell} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,\ell} \\ \vdots & \vdots & & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,\ell} \end{pmatrix} \end{array}$$

with $a_{i,j} \in R$, we set $g_i := \sum_{j=1}^{m} a_{i,j} t_j$ for $1 \le i \le m$, and $\mathrm{Mac}^{-1}(A,T) := \{g_1, \ldots, g_m\}$.

Example 1. Consider the following three quadratic polynomials (over $R = \mathbb{Z}$) in two variables x_1 and x_2:

$$\begin{aligned} f_1 &= 5x_1^2 + 6x_1x_2 + 4x_1 + 5x_2 + 3, \\ f_2 &= 4x_1^2 + 5x_1x_2 + 3x_2^2 + 6x_1 + 2x_2 + 2, \\ f_3 &= 2x_1^2 + 4x_1x_2 + 2x_2^2 + 6x_1 + x_2 + 2. \end{aligned}$$

Put $F := \{f_1, f_2, f_3\}$, $S := T_1 \cdot F = \{x_i f_j : 1 \le i \le 2,\ 1 \le j \le 3\}$, where T_1 is the set of monomials in x_1 and x_2 of degree one (cf. (2.1)). We order elements of S as follows: $S = \{x_1f_1, x_1f_2, x_1f_3, x_2f_1, x_2f_2, x_2f_3\}$. Let $\succ_{glex}$ be the graded lex order on $[x]$ with $x_1 \succ x_2$, that is, for $x^\alpha, x^\beta \in [x]$ with $\alpha, \beta \in (\mathbb{Z}_{\ge 0})^2$, we say $x^\alpha \succ_{glex} x^\beta$ if $|\alpha| > |\beta|$, or $|\alpha| = |\beta|$ and x^α is greater than x^β with respect to the lexicographical order with $x_1 \succ x_2$. When we order elements of $T_{\le 3}$ by $\succ_{glex}$, the Macaulay matrix $\mathrm{Mac}(S, T_{\le 3})$ of S with

respect to $T_{\leq 3}$ is given as follows:

$$\begin{array}{c} \\ x_1f_1 \\ x_1f_2 \\ x_1f_3 \\ x_2f_1 \\ x_2f_2 \\ x_2f_3 \end{array} \begin{array}{c} \begin{array}{cccccccccc} x_1^3 & x_1^2x_2 & x_1x_2^2 & x_2^3 & x_1^2 & x_1x_2 & x_2^2 & x_1 & x_2 & 1 \end{array} \\ \begin{pmatrix} 5 & 6 & 0 & 0 & 4 & 5 & 0 & 3 & 0 & 0 \\ 4 & 5 & 3 & 0 & 6 & 2 & 0 & 2 & 0 & 0 \\ 2 & 4 & 2 & 0 & 6 & 1 & 0 & 2 & 0 & 0 \\ 0 & 5 & 6 & 0 & 0 & 4 & 5 & 0 & 3 & 0 \\ 0 & 4 & 5 & 3 & 0 & 6 & 2 & 0 & 2 & 0 \\ 0 & 2 & 4 & 2 & 0 & 6 & 1 & 0 & 2 & 0 \end{pmatrix} \end{array}.$$

Here, we present an algorithm that is used in the XL algorithm as a subroutine. For a given shift S of a finite set F of polynomials, this algorithm computes another set of generators for the ideal $\langle F \rangle$, by constructing a Macaulay matrix.

Algorithm 2 (Basis conversion via Macaulay matrices)**.** The following sequence of procedures corresponds to the **Linearize** step of the XL algorithm (Algorithm 3 below):

Input: A shift S of F, that is, a finite subset of $[x] \cdot F$.

Output: A set G of generators for the ideal $\langle F \rangle$.

(1) Taking $T \subset [x]$ to be such that $\bigcup_{f \in S} \mathrm{supp}(f) \subset T$ (in fact, it suffices to put $T := \bigcup_{f \in S} \mathrm{supp}(f)$), construct the Macaulay matrix $\mathrm{Mac}(S, T)$ of S with respect to T.

(2) Compute the reduced row echelon form A of $\mathrm{Mac}(S, T)$.

(3) Obtain a set G of polynomials from A, where $G := \mathrm{Mac}^{-1}(A, T)$, and output it.

In general, Algorithm 2 outputs not necessarily a Gröbner basis of the ideal generated by the input polynomials, but Theorem 13 in Appendix shows that for sufficiently large shifts, the output is a Gröbner basis.

2.3 XL Algorithm

This subsection briefly reviews *the XL algorithm* (which stands for eXtended Linearizations, or for multiplication and linearization), which is proposed in [10] by Courtois et al. to solve a system of multivariate polynomials (over finite fields). We describe the XL algorithm with Macaulay matrices defined

in the previous subsection. The notations are the same as in the previous subsections, unless otherwise noted.

In Algorithm 3 below, we write down the XL algorithm in a form that can work for a multivariate system (of arbitrary degree) over a finite field K. Note that the original paper [10] treats only the case where the input polynomials are all quadratic, but clearly their idea is applicable to a general multivariate system of higher degree. For simplicity, we here assume that the set of zeros over the algebraic closure $\overline{K}$ of the input system is finite.

Algorithm 3 (XL algorithm, [10], Section 3, Definition 1)**.**

Input: A finite subset $F = \{f_1, \ldots, f_m\} \subset K[x]$, and a degree bound D.

Output: A zero of the ideal $\langle F \rangle$.

(1) **Multiply:** Construct a shift $I_{\leq D}$ of F by computing all the products $t \cdot f_i$ with $t \in [x]$ and $\deg(tf_i) \leq D$, say

$$I_{\leq D} := \bigcup_{i=1}^{m} \{t \cdot f_i : t \in [x],\ \deg(tf_i) \leq D\}.$$

(2) **Linearize:** For $S := I_{\leq D}$ above as an input, execute Algorithm 2 for some elimination monomial order such that all the terms containing one variable (say x_n) are eliminated last. In Step (1) of Algorithm 2, take T to be $\{t \in [x] : \deg(t) \leq D\}$. Let G be the output of Algorithm 2.

(3) **Solve:** If G contains a univarite polynomial in $K[x_n]$, compute its root by e.g., Berlekamp's algorithm [4].

(4) **Repeat:** Substituting a root into x_n, simplify the equations, and then repeat the process to find the values of the other variables.

It will be proved in Corollary 16 (in Appendix B) that Algorithm 2 in Step (2) outputs a Gröbner basis of $\langle F \rangle$ with respect to an elimination order, for D larger than *Dube's degree bound* $\mathrm{Dube}_{n+1,d}$, where $d := \max\{\deg(f_i) : 1 \leq i \leq m\}$. This implies that for such D, Algorithm 3 surely finds a zero of $\langle F \rangle$, see Proposition 18.

One considerable variant of Algorithm 3 is: In the **Linearize** step, compute a Göbner basis G_0 for $\langle F \rangle$ with respect to an arbitrary monomial order, by executing Algorithm 2 for sufficiently large D. Then, apply the FGLM basis conversion algorithm [17] to compute a Gröbner basis G with respect to the required elimination monomial order.

2.4 Complexity

As it was noted in the previous subsection, Dubé's degree bound given in Appendix B is a bound of the degree at which **Linearize**-step of XL outputs a Gröbner basis. However, in practical, XL can find a solution at degree *much* smaller than the Dubé's degree bound: Computing a Gröbner basis enables us to enumerate *all* zeros of a given (zero-dimensional) multivariate system, but it suffices for the motivation of XL to find *at least one* root.

In this subsection, we first consider the practical bound of the degree at which XL outputs a solution, under some assumptions. After that, by using this bound, the complexities of XL and its variant Wiedeman XL are estimated. In the following, we suppose that systems to be solved are given as quadratic equations over the finite field $\mathbb{F}_q$ with q elements.

We denote by $\langle I_{\leq d} \rangle_{\mathbb{F}_q}$ the $\mathbb{F}_q$-linear space generated by the set $I_{\leq d}$. To obtain the dimension of $\langle I_{\leq d} \rangle_{\mathbb{F}_q}$, we consider the linear dependency of multiples of monomials in $x_1, \ldots, x_n$ and f_i, f_j. When we write

$$f_i(\mathbf{x}) \;=\; \sum_{k \leq \ell} a_{k,\ell}^{(i)} x_k x_\ell + \sum_k b_k^{(i)} x_k + c^{(i)},$$

then we have

$$\begin{aligned} & \sum_{k \leq \ell} a_{k,\ell}^{(i)} (x_k x_\ell f_j) + \sum_k b_k^{(i)} (x_k f_j) + c^{(i)} f_j \\ = \; & \sum_{k \leq \ell} a_{k,\ell}^{(j)} (x_k x_\ell f_i) + \sum_k b_k^{(j)} (x_k f_i) + c^{(j)} f_i. \end{aligned} \tag{2.3}$$

This means that a set of polynomials $\{t \cdot f_\ell \mid t \in T_{\leq 2},\, \ell \in \{i, j\}\}$ is linearly dependent over $\mathbb{F}_q$. Furthermore, equations obtained by multiplying monomials in $x_1, \ldots, x_n$ to the both sides of (2.3) indicate the linear dependencies at degree larger than 2. The following proposition enables us to compute the dimension of $\langle I_{\leq d} \rangle_{\mathbb{F}_q}$ assuming no other source of dependencies than the above:

Proposition 4 ([25], Proposition 1)**.** *If all dependencies of $\langle I_{\leq d} \rangle_{\mathbb{F}_q}$ result from the dependency of $\{t \cdot f_\ell \mid t \in T_{\leq 2},\, \ell \in \{i, j\}\}$, then*

$$|T_{\leq d}| - \dim_{\mathbb{F}_q}(\langle I_{\leq d} \rangle_{\mathbb{F}_q}) = \mathrm{coeff}\left((1-t)^{m-n-1}(1+t)^m, t^d\right), \tag{2.4}$$

for all $d < \min\{q, D_{reg}\}$. Here D_{reg} is the degree of regularity for XL given by

$$D_{reg} = \min\left\{d \mid \mathrm{coeff}\left((1-t)^{m-n-1}(1+t)^m, t^d\right) \leq 0\right\}. \tag{2.5}$$

Here, we consider the rank of the Macaulay matrix required to obtain a univarite equation. Constructing the Macaulay matrix $\mathrm{Mac}(I_{\leq D}, T_{\leq D})$, we suppose to use *elimination orders* such that $x_1^D, x_1^{D-1}, \ldots, x_1, 1$ are listed at the end. In this case, if $\mathrm{rank}(\mathrm{Mac}(I_{\leq D}, T_{\leq D})) \geq |T_{\leq D}| - D$, then the last nonzero row vector of the matrix in the row echelon form yields a univarite equation of x_1. Therefore, from Proposition 4, the minimum D required for the reliable termination of XL is given by

$$D = \min \left\{ d \mid \mathrm{coeff}\left((1-t)^{m-n-1} (1+t)^m, t^d \right) \leq d \right\}, \tag{2.6}$$

in the case where q is sufficiently large. We call this degree D the *solving degree of XL*. Note that if n and m are determined, then the solving degree can be computed. One can easily confirm that the solving degree is much smaller than the Dubé's degree bound (e.g., the solving degree of XL on systems with $n = 10$ and $m = 11$ is 12, whereas the Dubé's degree bound on the same system is approximately 10^{76}).

We here consider the time complexity of XL. The **Linearize** step is clearly dominant in terms of the time complexity. In the **Linearize** step, Gaussian elimination is performed on a matrix with $m \cdot \binom{n+D-2}{D-2}$ rows and $\binom{n+D}{D}$ columns, where D is the solving degree of XL. Here, following [22], we use a practical assumption that, if we pick rows at random under the constraint that we have enough equations at each degree $d \leq D$, then usually we have a linearly independent set. From this assumption, the complexity of XL is roughly estimated as that of Gaussian elimination on a matrix with $\binom{n+D}{D}$ rows and columns, and it is given by

$$O\left(\binom{n+D}{D}^{\omega} \right), \tag{2.7}$$

where $2 \leq \omega < 3$ is the constant in the complexity of matrix multiplication.

Wiedemann XL (WXL) We explain a variant of XL using Wiedemann's algorithm [24] instead of Gaussian elimination, which was first analyzed in [27]. Wiedemann's algorithm generally solves sparse linear systems more efficiently than Gaussian elimination. According to [13], the complexity of WXL is estimated by

$$O\left(\binom{n-k}{2} \cdot \binom{n-k+D}{D}^2 \right), \tag{2.8}$$

where D is the solving degree of XL. (We remove the constant part from the complexity in [13], since we focus on asymptotic complexity.) Note that WXL consumes less memory than the plain XL, since it can deal with the Macaulay matrix as a sparse matrix.

2.5 Hybrid Approach

In this subsection, we describe the hybrid approach [5, 26]. This approach is constructed combining an exhaustive search with an $\mathcal{MQ}$ solver, such as F4, F5, or XL.

In this approach, given the $\mathcal{MQ}$ system of m equations in n variables, k $(0 \leq k \leq n)$ variables are randomly guessed before an $\mathcal{MQ}$ solver is applied to the system in the remaining $n-k$ variables; this is repeated until a solution is obtained. For the case of applying the hybrid approach to the plain XL, the following step is added into Algorithm 3

(0) **Fix:** Fix the k variables $x_1, \ldots, x_k$.

The complexities of the hybrid approaches using the plain XL and WXL as $\mathcal{MQ}$ solvers are estimated as

$$O \quad \left(q^k \cdot \binom{n-k+D}{D}^{\omega} \right), \tag{2.9}$$

$$O \quad \left(q^k \cdot \binom{n-k}{2} \cdot \binom{n-k+D}{D}^2 \right), \tag{2.10}$$

respectively. The number k of guessed variables is generally chosen such that the above complexity takes the minimum value.

3 Main Algorithm

In this section, we propose a new variant of the XL algorithm solving the $\mathcal{MQ}$ problem of m equations in n variables over $\mathbb{F}_q$ where $n \leq m$. We first discuss Macaulay matrices over polynomial rings, and second describe the outline of our proposed algorithm "polynomial XL (PXL)". After that, the details of the most technical step will be described in Subsection 3.3, and the termination and solving degree will be discussed in Subsection 3.4.

3.1 Macaulay Matrices over Polynomial Rings

Given an $\mathcal{MQ}$ system $\mathcal{F} = \{f_1(\mathbf{x}), \ldots, f_m(\mathbf{x})\}$ in n variables $\mathbf{x} = (x_1, \ldots, x_n)$ over $\mathbb{F}_q$, we discuss Macaulay matrices over polynomial rings. For a positive integer $k \in \{1, \ldots, n\}$, we divide $x_1, \ldots, x_n$ into k variables $x_1, \ldots, x_k$ and the remaining $n-k$ variables $x_{k+1}, \ldots, x_n$ as "main variables". Then, $f_1, \ldots, f_m$ can be regarded as elements of the polynomial ring in the $n-k$ main variables over the polynomial ring in the k variables, that is $(\mathbb{F}_q[x_1, \ldots, x_k])[x_{k+1}, \ldots, x_n]$.

Here, we define some notations again over $(\mathbb{F}_q[x_1,\ldots,x_k])[x_{k+1},\ldots,x_n]$ as in Subsection 2.1. For each $b \geq a \geq 0$, we define the sets T_a, $T_{\leq a}$, and $T_{a;b}$ of monomials in $x_{k+1},\ldots,x_n$ as follows:

$$\begin{aligned} T_a &:= \left\{ x_{k+1}^{\alpha_1} \cdots x_n^{\alpha_{n-k}} : \sum_{i=1}^{n-k} \alpha_i = a \right\}, \\ T_{\leq a} &:= \left\{ x_{k+1}^{\alpha_1} \cdots x_n^{\alpha_{n-k}} : \sum_{i=1}^{n-k} \alpha_i \leq a \right\}, \\ T_{a;b} &:= T_a \cup T_{a+1} \cup \cdots \cup T_b. \end{aligned}$$

Similarly, for each $b \geq a \geq 2$, we define the sets I_a, $I_{\leq a}$, and $I_{a;b}$ of polynomials in $(\mathbb{F}_q[x_1,\ldots,x_k])[x_{k+1},\ldots,x_n]$ as follows:

$$\begin{aligned} I_a &:= \bigcup_{i=1}^{m} \{t \cdot f_i : t \in T_{a-2}\}, \\ I_{\leq a} &:= \bigcup_{i=1}^{m} \{t \cdot f_i : t \in T_{\leq a-2}\}, \\ I_{a;b} &:= I_a \cup I_{a+1} \cup \cdots \cup I_b. \end{aligned}$$

For an integer $D \geq 0$, we consider $\mathrm{Mac}(I_{\leq D}, T_{\leq D})$ over the polynomial ring $\mathbb{F}_q[x_1,\ldots,x_k]$ with orders first comparing the degree of monomials and polynomials in both of $I_{\leq D}$ and $T_{\leq D}$. In the following, we call such a Macaulay matrix as the Macaulay matrix of $\mathcal{F}$ at degree D over $\mathbb{F}_q[x_1,\ldots,x_k]$.

Example 5. For the system of equations

$$\begin{aligned} f_1 &= 5x_1^2 + 6x_1x_2 + 4x_1x_3 + x_2x_3 + 5x_3^2 + 4x_1 + 5x_2 + 3, \\ f_2 &= 4x_1^2 + 5x_1x_2 + 4x_1x_3 + 3x_2^2 + 5x_2x_3 + x_3^2 + 6x_1 + 2x_2 + 3x_3 + 2, \\ f_3 &= 2x_1^2 + 4x_1x_2 + 2x_2^2 + 6x_3^2 + 6x_1 + x_2 + 3x_3 + 2, \end{aligned}$$

in $\mathbb{F}_7[x_1,x_2,x_3]$, if we set $k=1$, then the Macaulay matrix of $\mathcal{F}$ at degree 3 over $\mathbb{F}_7[x_1]$ with monomials in the graded lex order and polynomials ordered such as $\{x_2f_1, x_2f_2, x_2f_3, x_3f_1, x_3f_2, x_3f_3, f_1, f_2, f_3\}$ is as follows:

	x_2^3	$x_2^2x_3$	$x_2x_3^2$	x_3^3	x_2^2	x_2x_3	x_3^2	x_2	x_3	1
x_2f_1	0	1	5	0	$6x_1+5$	$4x_1$	0	$5x_1^2+4x_1+3$	0	0
x_2f_2	3	5	1	0	$5x_1+2$	$4x_1+3$	0	$4x_1^2+6x_1+2$	0	0
x_2f_3	2	0	6	0	$4x_1+1$	3	0	$2x_1^2+6x_1+2$	0	0
x_3f_1	0	0	1	5	0	$6x_1+5$	$4x_1$	0	$5x_1^2+4x_1+3$	0
x_3f_2	0	3	5	1	0	$5x_1+2$	$4x_1+3$	0	$4x_1^2+6x_1+2$	0
x_3f_3	0	2	0	6	0	$4x_1+1$	3	0	$2x_1^2+6x_1+2$	0
f_1	0	0	0	0	0	1	5	$6x_1+5$	$4x_1$	$5x_1^2+4x_1+3$
f_2	0	0	0	0	3	5	1	$5x_1+2$	$4x_1+3$	$4x_1^2+6x_1+2$
f_3	0	0	0	0	2	0	6	$4x_1+1$	3	$2x_1^2+6x_1+2$

Let $\mathcal{PM}$ be a Macaulay matrix of $\mathcal{F}$ at degree D over $\mathbb{F}_q[x_1, \ldots, x_k]$ and we discuss the structure of $\mathcal{PM}$ in the following. For two integers d_1 and d_2 ($2 \leq d_1 \leq D$, $0 \leq d_2 \leq D$), we denote by $\mathcal{PM}[I_{d_1}, T_{d_2}]$ the submatrix of $\mathcal{PM}$ corresponding to polynomials of I_{d_1} and monomials of T_{d_2}. Then, $\mathcal{PM}$ is clearly divided by submatrices $\mathcal{PM}[I_{d_1}, T_{d_2}]$ ($2 \leq d_1 \leq D, 0 \leq d_2 \leq D$).

Lemma 6. *For an $\mathcal{MQ}$ system $\mathcal{F}$ and positive integers $k \leq n$ and $D \geq 2$, let $\mathcal{PM}$ be a Macaulay matrix of $\mathcal{F}$ at degree D over $\mathbb{F}_q[x_1, \ldots, x_k]$. Then, for $2 \leq d \leq D$, every $\mathcal{PM}[I_d, T_{d'}]$ with $d' \notin \{d, d-1, d-2\}$ is a zero matrix, and all elements of $\mathcal{PM}[I_d, T_d]$ belong to $\mathbb{F}_q$.*

This lemma holds due to quadraticity of $\mathcal{F}$. The proposed algorithm which will be given in the next subsection utilizes this structure.

3.2 Outline

In this subsection, we describe the outline of the proposed algorithm as a variant of h-XL, which is called polynomial XL (PXL). The main difference between our PXL and h-XL is the following: While h-XL performs Gaussian elimination after substituting actual k values to $x_1, \ldots, x_k$, PXL *partly* performs Gaussian elimination *before* fixing k variables. These manipulations are possible due to the structure of Macaulay matrices over $\mathbb{F}_q[x_1, \ldots, x_k]$ described in Subsection 3.1.

Here, we roughly describe PXL as follows:

Algorithm 7 (Polynomial XL)**.**

Input: An $\mathcal{MQ}$ system $\mathcal{F} = \{f_1(\mathbf{x}), \ldots, f_m(\mathbf{x})\}$, the number k of guessed variables, and a degree D of regularity for XL.

Output: A solution to $\mathcal{F}(\mathbf{x}) = \mathbf{0} \in \mathbb{F}_q^m$.

(1) **Multiply**: Generate all the products $t \cdot f_i$ ($t \in T_{\leq D-2}$)

(2) **Linearize(1)**: Generate a Macaulay matrix over $\mathbb{F}_q[x_1, \ldots, x_k]$ and partly perform Gaussian elimination.

(3) **Fix**: Fix the k variables in the resulting matrix of step 2.

(4) **Linearize(2)**: Perform Gaussian elimination on the resulting matrix of step 3.

(5) **Solve**: Assume that step 4 yields at least one univariate equation. Solve this equation over the finite field $\mathbb{F}_q$.

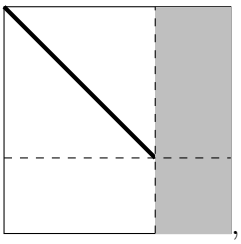
(6) **Repeat**: Simplify the equations and repeat the process to solve the other variables.

Note that the last four steps from **Fix** to **Repeat** are iterated until a solution is found.

Recall from Subsection 2.5 that, after k variables are guessed, the hybrid approach solves the remaining system using F4, F5, or (plain) XL. The **Fix** step also guesses values of k variables as in the hybrid approach. For n variables $x_1, \ldots, x_n$ of the given system, we set the first k variables $x_1, \ldots, x_k$ as guessed variables.

Before the **Fix** step, the **Multiply** and **Linearize(1)** steps are executed regarding the given polynomials as those in $(\mathbb{F}_q[x_1, \ldots, x_k])[x_{k+1}, \ldots, x_n]$. We here recall from Lemma 6 that the Macaulay matrix over $\mathbb{F}_q[x_1, \ldots, x_k]$ constructed in the **Linearize(1)** step has some submatrices with entries in $\mathbb{F}_q$. By utilizing this property, the **Linearize(1)** step repeats to transform such a block into the row echelon form and to eliminate entries of its upper and lower blocks. The detail of this step is explained in Subsection 3.3 below.

After the **Linearize(1)** step, the resulting Macaulay matrix has the following form



with the permutation of rows and columns. (All elements in the white parts are zero, whereas all elements in the gray parts and on the diagonal line are nonzero.) Then, for a sufficiently large D, we can solve the system eliminating only the lower-right submatrix in the above figure, since, by doing so, at least one univariate equation is obtained. Therefore, the **Fix** step substitutes values of k variables to $x_1, \ldots, x_k$ only in the lower-right submatrix, and the **Linearize(2)** step solves the system performing Gaussian elimination over the finite field. We remark that the **Solve** and **Repeat** steps are executed similarly to the plain XL.

3.3 Details of Linearize(1) Step

In this subsection, we describe the details of the **Linearize(1)** step in the proposed algorithm. As in Subsection 3.1, we let $\mathcal{PM}$ be a Macaulay matrix

of $\mathcal{F}$ at degree D over $\mathbb{F}_q[x_1, \ldots, x_k]$.

The **Linearize(1)** step is mainly performed on each $\mathcal{PM}[I_d, T_{(d-2);d}]$ $(2 \leq d \leq D)$ and iterated from $d = D$ to 2. This step at each degree d is described as follows:

(d)-1. Perform Gaussian elimination on $\mathcal{PM}[I_d, T_d]$.

(d)-2. Perform the same row operations as (d)-1 on $\mathcal{PM}[I_d, T_{(d-2);(d-1)}]$.

(d)-3. Using the *leading coefficients* of the resulting $\mathcal{PM}[I_d, T_d]$, eliminate the corresponding columns of $\mathcal{PM}$. Here, a leading coefficient is the leftmost nonzero entry in each row of a matrix of the row echelon form.

In the following, we show that the **Linearize(1)** step described above works well. We first show that, at the time of executing the (d)-1 step on $\mathcal{PM}[I_d, T_d]$, the submatrix $\mathcal{PM}[I_d, T_{\leq D}]$ does not change from the first state. Since $\mathcal{PM}[I_d, T_{(d+1);D}]$ is a zero matrix, the (d)-3 step of the degree from D to $d+1$ does not affect $\mathcal{PM}[I_d, T_{\leq D}]$. Therefore, from Lemma 6, only $\mathcal{PM}[I_d, T_d]$, $\mathcal{PM}[I_d, T_{d-1}]$, and $\mathcal{PM}[I_d, T_{d-2}]$ among $\mathcal{PM}[I_d, T_{d'}]$ $(0 \leq d' \leq D)$ are not zero matrices, and $\mathcal{PM}[I_d, T_d]$ can be seen as a matrix over the finite field $\mathbb{F}_q$. This fact indicates that the manipulations in the (d)-1 and (d)-2 steps can be performed correctly. Furthermore, the (d)-3 step can be also performed correctly, since the leading coefficients of the resulting $\mathcal{PM}[I_d, T_d]$ are 1. As a result, we have that all the manipulations are practicable. In addition, we remark that these manipulations are clearly regarded as row operations on $\mathcal{PM}$.

After the **Linearize(1)** step, all manipulations are performed on the submatrix obtained by concatenating rows and columns including no leading coefficient of the resulting $\mathcal{PM}[I_d, T_d]$ with $2 \leq d \leq D$. In the following, we call this submatrix the resulting matrix of **Linearize(1)**.

3.4 Termination and Solving Degree

This subsection discusses the termination of PXL and the minimum value D where PXL finds a solution under the same assumption as in Proposition 4, which is called the *solving degree of PXL*.

From the discussion in Subsection 3.3, even if the **Linearize(1)** step is exchanged with the **Fix** step, then the **Linearize(1)** step works well as row operations on $\mathcal{PM}$. Therefore, for any guessed values, the rank of $\mathcal{PM}$ after the **Fix** step in PXL is the same as that of the Macaulay matrix after the **Fix** step in h-XL. This means that the rank of $\mathcal{PM}$ will be sufficiently close

to the number of columns of $\mathcal{PM}$ at sufficiently large D as similar to h-XL. Namely, PXL outputs a solution at sufficiently large D.

In the following, we discuss the solving degree of PXL. In the plain XL, the solving degree is obtained by equality (2.6) to yield a univarite (e.g., x_n) equation including multiples of $D+1$ monomials (e.g., $1, x_n, \ldots, x_n^D$). However, in PXL, some of such monomials may be eliminated in the **Linearize(1)** step. Then, the upper bound of the solving degree of PXL clearly coincides with the degree of regularity for XL in equality (2.5). Hence, we estimate the solving degree of PXL by the degree of regularity for XL. One can confirm that this degree is the same as the degree obtained by the solving degree of XL in most cases. Indeed, for all cases $10 \leq n \leq m \leq 100$, if the solving degree of XL is larger than 2, then it is always equal to the degree of regularity for XL.

Remark 8. This remark states that PXL does not need to use all row vectors of Macaulay matrices practically. As in the estimation in Subsection 2.3, we use the following assumption about the number of rows of Macaulay matrices: If we pick rows at random under the constraint that we have enough equations at each degree $d \leq D$, then usually we have a linearly independent set. From this assumption, PXL randomly chooses approximately $|T_{\leq D}|$ independent row vectors from the Macaulay matrix with $|I_{\leq D}|$ rows and executes the **Linearize(1)** step on the submatrix composed of chosen row vectors.

4 Complexity

In this section, we first estimate the size of the resulting matrix of **Linearize(1)**. After that, we estimate the time complexity of PXL and compare it with those of h-XL and h-WXL. Note that we take the input parameter D of PXL as the degree of regularity for XL.

4.1 Size of Resulting Matrix of Linearize(1)

Let α be the number of columns of the resulting matrix of **Linearize(1)**. For each d with $2 \leq d \leq D$, we define the set $\bar{I}_d$ of polynomials by

$$\bar{I}_d \coloneqq \{\, \text{the degree } d \text{ part of } a \mid a \in I_d \}.$$

If we denote by $\langle \bar{I}_d \rangle_{\mathbb{F}_q}$ the $\mathbb{F}_q$-linear space generated by the set $\bar{I}_d$, then the number of columns eliminated in the step (d)-1 of **Linearize(1)** on

$\mathcal{PM}[I_d, T_d]$ is equal to $\dim_{\mathbb{F}_q}(\langle \bar{I}_d \rangle_{\mathbb{F}_q})$. Therefore, we have

$$\begin{aligned} \alpha &= |T_{\leq D}| - \sum_{d=2}^{D} \dim_{\mathbb{F}_q}(\langle \bar{I}_d \rangle_{\mathbb{F}_q}) \\ &= \sum_{d=2}^{D} \left(|T_d| - \dim_{\mathbb{F}_q}(\langle \bar{I}_d \rangle_{\mathbb{F}_q}) \right) + |T_1| + |T_0|. \end{aligned} \tag{4.1}$$

Using the same assumption as in Proposition 4, the value of (4.1) can be estimated by

$$\sum_{d=0}^{D} \max \left\{ \mathrm{coeff}\left((1-t)^{m-(n-k)} (1+t)^m, t^d \right), 0 \right\}. \tag{4.2}$$

Recall from Remark 8 that PXL randomly chooses approximately $|T_{\leq D}|$ rows from the Macaulay matrix. In the following, we suppose that approximately $|T_d| - \mathrm{coeff}\left((1-t)^{m-(n-k)} (1+t)^m, t^d \right)$ rows are chosen from I_d for $2 \leq d \leq D-1$. By doing so, we can take the resulting matrix of **Linearize(1)** as an approximately $\alpha \times \alpha$ matrix. Furthermore, when $\tilde{I}_d$ denotes the subset of I_d including polynomials corresponding to randomly chosen rows, we have

$$\sum_{d=2}^{D} (|\tilde{I}_d| - |T_d|) \approx \alpha. \tag{4.3}$$

4.2 Time Complexity

In this subsection, we approximately estimate the time complexity of PXL.

Proposition 9 (Time Complexity of PXL)**.** *The time complexity of PXL is approximately estimated by*

$$C_{(d)1} + C_{(d)2} + C_{(d)3} + C_{\mathsf{fix}} + C_{\mathsf{li2}}. \tag{4.4}$$

*Here, $C_{(d)1}$ (resp. $C_{(d)2}$, $C_{(d)3}$) denotes the sum of the number of manipulations in $\mathbb{F}_q$ required for each $(d)-1$ (resp. $(d)-2$, $(d)-3$) with $2 \leq d \leq D$. Furthermore, C_{fix} and C_{li2} denote the number of manipulations in $\mathbb{F}_q$ required for the **fix** and **Linearize(2)** steps, respectively.*

The estimations $C_{(d)1}$, $C_{(d)2}$, $C_{(d)3}$, C_{fix}, and C_{li2} are determined from the number n of all variables, the number k of guessed variables, the degree D of regularity for XL, and the size α of the resulting matrix of **Linearize(1)**. In the following, we will obtain these estimations.

Time Complexity of (d)-1 Recall that the (d)-1 step performs Gaussian elimination on $\mathcal{PM}[\tilde{I}_d, T_d]$. Since the complexity of the (d)-1 step is $\max\{|\tilde{I}_d|, |T_d|\}^\omega$ for each $2 \leq d \leq D$, it follows from equation (4.3) that an upper bound on the sum of the complexity estimation of the (d)-1 step for $2 \leq d \leq D$ is given by

$$\begin{aligned}\sum_{d=2}^{D} \max\{|\tilde{I}_d|, |T_d|\}^\omega &\leq \left(\sum_{d=2}^{D} \max\{|\tilde{I}_d|, |T_d|\}\right)^\omega \\ &\leq (|T_D| + \alpha)^\omega \\ &\leq (2 \cdot |T_D|)^\omega = O\left(\binom{n-k+D}{D}^\omega\right), \end{aligned} \tag{4.5}$$

and thus we set $C_{(d)1}$ as $\binom{n-k+D}{D}^\omega$.

Time Complexity of (d)-2 In the case of (d)-2, the matrix $\mathcal{PM}[\tilde{I}_d, T_{(d-2);(d-1)}]$ is sparse. Indeed, each row of $\mathcal{PM}[\tilde{I}_d, T_{(d-2);(d-1)}]$ has at most $n-k+1$ nonzero elements. Furthermore, the degree of every element is smaller than or equal to 2. By considering these facts, an upper bound on the sum of the complexity of the (d)-2 step over $\mathbb{F}_q$ for $2 \leq d \leq D$ is

$$\begin{aligned}\sum_{d=2}^{D} \left(\binom{k+2}{2} \cdot (n-k) \cdot |\tilde{I}_d|^2\right) &\leq \binom{k+2}{2} \cdot (n-k) \cdot |\tilde{I}_{\leq D}|^2 \\ &= O\left(k^2 \cdot (n-k) \cdot \binom{n-k+D}{D}^2\right), \end{aligned} \tag{4.6}$$

and thus $C_{(d)2}$ is set to be $k^2 \cdot (n-k) \cdot \binom{n-k+D}{D}^2$.

Time Complexity of (d)-3 Before estimating the time complexity of (d)-3 with $2 \leq d \leq D$, we show the following lemma:

Lemma 10. *At the time of executing the (d)-3 step with $2 \leq d \leq D-1$, the degree of every element of $\mathcal{PM}[\tilde{I}_{(d+1);D}, T_d]$ is lower than or equal to $D-d$.*

Proof. By the induction, we prove that, at the time of the (d)-3 step, the degree of every element of $\mathcal{PM}[\tilde{I}_{(d+1);D}, T_d]$ and $\mathcal{PM}[\tilde{I}_{(d+1);D}, T_{d-1}]$ is lower than or equal to $D-d$ and $D-d+1$, respectively. In the case where $d = D-1$, the above statement clearly holds. In the following, we show that, if the statement holds when $d = d'$ with $3 \leq d' \leq D-1$, then it also holds when $d = d'-1$. Before executing the step (d')-3, $\mathcal{PM}[\tilde{I}_{(d'+1);D}, T_{d'-2}]$ is a zero matrix clearly. Then, the (d')-3 step adds row vectors, which is obtained

by multiplying a polynomial with the degree $D - d'$ to rows corresponding to $\tilde{I}_{d'}$, to rows corresponding to $\tilde{I}_{(d'+1);D}$. Here, the degree of each entry of $\mathcal{PM}[\tilde{I}_{d'}, T_{d'-1}]$ and $\mathcal{PM}[\tilde{I}_{d'}, T_{d'-2}]$ are at most 1 and 2, respectively. Hence, through (d')-3, the degree of each entry of $\mathcal{PM}[\tilde{I}_{(d'+1);D}, T_{d'-2}]$ becomes at most $D-d'+2$ and that of $\mathcal{PM}[\tilde{I}_{(d'+1);D}, T_{d'-1}]$ remains at most $D-d'+1$, Therefore, the statement holds in the case where $d = d' - 1$, as desired. $\square$

Suppose for the efficiency that the (d)-3 step only eliminates elements of rows including no leading coefficients in $\mathcal{PM}[\tilde{I}_{d'}, T_{d'}]$ $(d+1 \leq d' \leq D)$. In this case, the number of the eliminated rows in (d)-3 is at most α from the discussion in Subsection 4.1. Considering this together with Lemma 10, we estimate the complexity of the (d)-3 step as

$$O\left(\binom{k+D-d}{D-d} \cdot k^2 \cdot \alpha \cdot \binom{n-k+d-1}{d}^2\right).$$

Note that the (D)-3 step can be omitted since $\mathcal{PM}[\tilde{I}_{\leq(D-1)}, T_D]$ is a zero matrix. Then, the sum of the complexity of the (d)-3 step for $2 \leq d \leq D-1$ is estimated by

$$\begin{aligned}
&\sum_{d=2}^{D-1} \left(\binom{k+D-d}{D-d} \cdot k^2 \cdot \alpha \cdot \binom{n-k+d-1}{d}^2\right) \\
&\leq k^2 \cdot \alpha \cdot \left(\sum_{d=2}^{D-1} \binom{n-k+d-1}{d}\right) \cdot \left(\sum_{d=2}^{D-1} \left(\binom{k+D-d}{k} \cdot \binom{n-k+d-1}{n-k-1}\right)\right) \\
&\leq O\left(k^2 \cdot \alpha \cdot \binom{n-k+D}{D} \cdot \binom{n+D}{D}\right), \qquad (4.7)
\end{aligned}$$

and thus we set $C_{(d)3}$ as $k^2 \cdot \alpha \cdot \binom{n-k+D}{D} \cdot \binom{n+D}{D}$.

Time Complexity of Fix The size of the resulting matrix of **Linearize(1)** is approximately $\alpha \times \alpha$ due to the discussion in Subsection 4.1, and the degree of every element in the matrix is lower than or equal to D from Lemma 10. Therefore, the time complexity of **Fix** is estimated as that of substituting k values to $x_1, \ldots, x_k$ in α^2 polynomials with degree D in $\mathbb{F}_q[x_1, \ldots, x_k]$. If we use a naive approach, then the complexity of evaluation of a polynomial with degree d in n variables is estimated by $\binom{n+d}{d}$. Therefore, C_{fix} is given by

$$C_{\mathsf{fix}} = q^k \cdot \alpha^2 \cdot \binom{k+D}{D}, \qquad (4.8)$$

since the **Fix** step is iterated for any values of $x_1, \ldots, x_k$.

Time Complexity of Linearize(2) The **Linearize(2)** step performs Gaussian elimination on an $\alpha \times \alpha$ matrix over $\mathbb{F}_q$, and thus we estimate C_{li2} by

$$C_{\mathsf{li2}} = q^k \cdot \alpha^\omega, \tag{4.9}$$

considering q^k times iterations.

Rough Estimations of Time Complexity

Here, we present more compact formula for the time complexity of PXL given by (4.4). Comparing the estimations $C_{(d)2}$ and $C_{(d)3}$, we can easily confirm that the value of $C_{(d)3}$ is larger than that of $C_{(d)2}$. Furthermore, comparing the estimations $C_{(d)1}$ and $C_{(d)3}$, we experimentally confirmed that, for the case where $10 \leq n \leq 100$, $m = n$, $1.5n$, $2n$, and k, D, and α are the values of minimizing the estimation (4.4), the value of $C_{(d)3}$ is always much larger than that of $C_{(d)1}$ (e.g., $C_{(d)1}$ and $C_{(d)3}$ on the case of $n = m = 100$ with $q = 2^8$ is approximately 2^{210} and 2^{259}, respectively). These facts indicates that the complexity of the **Linearize(1)** step is dominated by $C_{(d)3}$ for the practical cases as follows:

$$O\left(k^2 \cdot \alpha \cdot \tbinom{n-k+D}{D} \cdot \tbinom{n+D}{D}\right). \tag{4.10}$$

By using this fact, the time complexity of PXL is roughly estimated by $C_{(d)3} + C_{\mathsf{fix}} + C_{\mathsf{li2}}$, say

$$O\left(k^2 \cdot \alpha \cdot \tbinom{n-k+D}{D} \cdot \tbinom{n+D}{D} + q^k \cdot \left(\alpha^2 \cdot \tbinom{k+D}{D} + \alpha^\omega\right)\right). \tag{4.11}$$

4.3 Comparison

We compare the complexity of our PXL with those of h-XL and h-WXL. We recall that the complexities of h-XL, h-WXL, and PXL are estimated by the estimation (2.9), (2.10), and (4.11), respectively, and, in each of the three approaches, the number k of guessed variables is chosen such that the complexity becomes the smallest value. We will see below, PXL is more efficient than h-XL and h-WXL especially in the case of $n = m$.

Table 1 compares the bit complexities of PXL, h-XL, and h-WXL on the $\mathcal{MQ}$ system of m equations in n variable over $\mathbb{F}_{2^8}$, respectively. Here, n is set to be 20, 40, 60, and 80 and m is set to be equal to n and $\lfloor 1.5n \rfloor$. For example, in the case where $q = 2^8$ and $n = m = 80$, the complexities of h-XL, h-WXL, and PXL are approximately estimated as 2^{252}, 2^{234}, and 2^{220}, respectively. As a result, in the case of $m = n$, Table 1 shows that PXL

Table 1: Comparison of complexities approximated by power of 2 between PXL, h-XL, and h-WXL on the $\mathcal{MQ}$ system with $m = n$ (above) and $m = \lfloor 1.5n \rfloor$ (bellow) over $\mathbb{F}_{2^8}$

$n \quad (m = n)$	20	40	60	80
h-XL	2^{75}	2^{134}	2^{194}	2^{252}
h-WXL	2^{75}	2^{129}	2^{182}	2^{234}
PXL	$\mathbf{2^{62}}$	$\mathbf{2^{117}}$	$\mathbf{2^{169}}$	$\mathbf{2^{220}}$

n $(m \approx 1.5n)$	20	40	60	80
h-XL	2^{47}	2^{79}	2^{115}	2^{146}
h-WXL	2^{49}	2^{78}	2^{110}	2^{137}
PXL	$\mathbf{2^{48}}$	$\mathbf{2^{86}}$	$\mathbf{2^{118}}$	$\mathbf{2^{154}}$

has the less complexity than those of h-XL and h-WXL. On the other hand, PXL is not efficient in highly overdetermined cases. This is because, in such overdetermined cases, k is set to be a very small value for the efficiency. We confirmed that such behavior of PXL can be seen over other finite fields.

Remark 11 (Spece Complexity)**.** It is known that the space complexities of h-XL and h-WXL are approximately estimated by $O\left(\binom{n-k+D}{D}^2\right)$ and $O\left(\binom{n-k}{2} \cdot \binom{n-k+D}{D}\right)$, respectively. The memory space consumed by PXL is upper-bounded by $O\left(\binom{k+D}{D} \cdot \binom{n-k+D}{D}^2\right)$, since the degree of every element of the Macaulay matrix is at most D through PXL from Lemma 10. These estimations cannot be directly compared to each other, since the values of the following two parameters depend on one's choice of an algorithm: The solving degree D and the number k of fixed values.

On the other hand, focusing on the sparsity/density of matrices, we predict that PXL is not efficient compared with h-WXL in terms of the space complexity: Through the elimination process of Macaulay matrices, WXL can deal with a Macaulay matrix as a sparse matrix due to Wiedemann's algorithm, whereas PXL holds some dense submatrices. Considering this together with the time complexities for practical parameters (cf. Subsection 4.3), we conclude that the relationship between PXL and h-WXL would be a trade-off between time and memory.

Remark 12 (**Fix** over Even Characteristic)**.** In the binary field case, Bouil-

laguet et al. [7] proposed a faster evaluation algorithm using Gray code. In this algorithm, a polynomial with degree d in n variables is evaluated for all values of n variables with $O(2^n \cdot d)$ operations using $O(n^d)$ bits of memory after an initialization phase of negligible complexity. In the case of $\mathbb{F}_{2^r}$, by regarding each polynomial in k variables over $\mathbb{F}_{2^r}$ as r polynomials in $k \cdot r$ variables over $\mathbb{F}_2$ (i.e., if $\theta_1, \ldots, \theta_r$ are basis of $\mathbb{F}_{2^r}$ over $\mathbb{F}_2$, for each variable x_i over $\mathbb{F}_{2^r}$, we set r variables $x_1^{(i)}, \ldots, x_r^{(i)}$ over $\mathbb{F}_2$ satisfying $x_i = \sum_{j=1}^{r} x_j^{(i)} \theta_j$), we can apply this approach to PXL. Then, the time complexity of the **Fix** step over $\mathbb{F}_{2^r}$ is given by $O\left(q^k \cdot \alpha^2 \cdot r \cdot D\right)$, whereas the memory space consumed by this step is estimated by $O\left(\alpha^2 \cdot r \cdot k^D\right)$. This indicates that PXL with Bouillaguet et al.'s algorithm will be better in terms of time complexity but worse in terms of space complexity.

5 Conclusion

We presented a new variant of XL, which is a major approach for solving the $\mathcal{MQ}$ problem. Our proposed polynomial XL (PXL) eliminates the linearized monomials in polynomial rings to solve the system efficiently, and this paper estimates its complexities.

Given a system of m multivariate quadratic equations in n variables, the proposed algorithm first regards the given system as the system in $n - k$ variables $x_{k+1}, \ldots, x_n$, whose coefficients belong to the polynomial ring $\mathbb{F}_q[x_1, \ldots, x_k]$. We then generate a Macaulay matrix over $\mathbb{F}_q[x_1, \ldots, x_k]$ and partly perform Gaussian elimination. Finally, random values are substituted to the k variables, and the remaining part of the matrix is transformed into the row echelon form. This construction reduces the amount of manipulations for each guessed value compared to h-XL. This paper presents an asymptotic estimation of the time complexity, which shows that the proposed algorithm solves the system faster in the case of $n \approx m$ than both h-XL and h-WXL. On the other hand, PXL is less efficient than h-WXL with respect to the space complexity for the following two reasons: Macaulay matrices become denser through PXL and the resulting matrix of **Linearize(1)** has elements with degree higher than or equal to 2.

This paper only discusses solving multivariate quadratic systems. However, as in the plain XL, the proposed algorithm can be also generalized to higher degree cases. Therefore, one considerable future work is to analyze the behavior of PXL on higher degree systems. Furthermore, to implement PXL efficiently is also an important problem. In our experiments given in Appendix A below, we implemented PXL over Magma, but this can be more

optimized by using an alternative programming language, e.g., C. Therefore, to provide such an optimized code for PXL is a challenging task.

Acknowledgments

This work was supported by JST CREST Grant Number JPMJCR2113, Japan, JSPS KAKENHI Grant Number JP21J20391, Japan, and JSPS Grant-in-Aid for Young Scientists 20K14301, Japan.

References

[1] Albrecht, M.-R., Cid, C., Faugère, J.-C., Perret, L.: On the relation between the MXL family of algorithms and Gröbner basis algorithms. J. Symb. Comput., **47**(8), pp. 926–941 (2012)

[2] Ars, G., Faugère, J.-C., Imai, H., Kawazoe, M., Sugita, M.: Comparison between XL and Gröbner basis algorithms. In: ASIACRYPT 2004, LNCS, vol. 3329, pp. 338–353. Springer (2004)

[3] Aschenbrenner, M., Leykin, A.: Degree Bounds for Gröbner Bases in Algebras of Solvable Type. Journal of Pure and Applied Algebra, **213**, pp. 1578–1605 (2009)

[4] Berlekamp, E. R.: Factoring polynomials over finite fields. Bell System Technical Journal. 46: pp. 1853–1859 (1967)

[5] Bettale, L., Faugère, J.-C., Perret, L.: Hybrid approach for solving multivariate systems over finite fields. Journal of Mathematical Cryptology **3**, pp. 177–197 (2009)

[6] Bosma, W., Cannon, J., Playoust, C.: The Magma algebra system. I. The user language. Journal of Symbolic Computation **24**(3-4), pp. 235–265 (1997)

[7] Bouillaguet, C., Chen, H.-C., Cheng, C.-M., Chou, T., Niederhagen, R., Shamir, A., Yang, B.-Y.: Fast exhaustive search for polynomial systems in F2. In: CHES 2010, LNCS, vol. 6225, pp. 203–218. Springer (2010)

[8] Buchberger, B.: Ein algorithmus zum auffinden der basiselemente des restklassenringes nach einem nulldimensionalen polynomideal. PhD thesis, Universität Innsbruck (1965)

[9] Casanova, A., Faugère, J.-C., Macario-Rat, G., Patarin, J., Perret, L., Ryckeghem, J.: GeMSS signature schemes proposal for NIST PQC project (round 3 version). `https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions` (2020)

[10] Courtois, N., Klimov, A., Patarin, J., Shamir, A.: Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In: EUROCRYPT 2000, LNCS, vol. 1807, pp. 392–407. Springer (2000)

[11] Cox, D.-A., Little, J., O'Shea, D.: Using Algebraic Geomatry. Second edition, Springer (2005)

[12] Cox, D.-A., Little, J., O'Shea, D.: Ideals, Varieties, and Algorithms. Fourth edition, Springer (2015)

[13] Ding, J., Chen, M.-S., Kannwischer, M., Patarin, J., Petzoldt, A., Schmidt, D., Yang, B.-Y.: Rainbow signature schemes proposal for NIST PQC project (round 3 version). `https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions` (2020)

[14] Dubé, T.-W.: The Structure of Polynomial Ideals and Gröbner Bases. SIAM Journal on Computing, **19** (4), pp. 750–773 (1990)

[15] Faugère, J.-C.: A new efficient algorithm for computing Gröbner bases (F4). Journal of Pure and Applied Algebra **139**(1-3), pp. 61–88 (1999)

[16] Faugère, J.-C.: A new efficient algorithm for computing Gröbner bases without reduction to zero (F5). In: ISSAC 2002, pp. 75–83. ACM (2002)

[17] 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), pp. 329–344 (1993) (DOI) doi:10.1006/jsco.1993.1051.

[18] Garey, M.-R., Johnson, D.-S.: Computers and intractability: a guide to the theory of NP-completeness. W. H. Freeman (1979)

[19] Kipnis, A., Shamir, A.: Cryptanalysis of the HFE public key cryptosystem by relinearization. In: CRYPTO 1999, LNCS, vol. 1666, pp. 19–30. Springer (1999)

[20] Kudo, M., Harashita, S., Senda, H.: Automorphism groups of superspecial curves of genus 4 over $\mathbb{F}_{11}$. Journal of Pure and Applied Algebra, **224**(9) (2020) (DOI) 10.1016/j.jpaa.2020.106339.

[21] Maletzky, A.: Formalization of Dubé's Degree Bounds for Gröbner Bases in Isabelle/HOL. In: CICM 2019, LNCS, to appear. Springer (2019)

[22] Wael Said Abdelmageed Mohamed: Improvements for the XL Algorithm with Applications to Algebraic Cryptanalysis. PhD thesis, TU Darmstadt (2011)

[23] NIST: Post-quantum cryptography CSRC. `https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization`

[24] Wiedemann, D. H.: Solving sparse linear equations over finite fields. IEEE Trans. Inf. Theor. **32**(1), pp. 54–62 (1986)

[25] Yang, B.-Y., Chen, J.-M.: All in the XL family: theory and practice. In: ICISC 2004, LNCS, vol. 3506, pp. 67–86. Springer (2004)

[26] Yang, B.-Y., Chen, J.-M., Courtois, N.: On asymptotic security estimates in XL and Gröbner bases-related algebraic cryptanalysis. In: ICICS 2004, LNCS, vol. 3269, pp. 401–413. Springer (2004)

[27] Yang, B.-Y., Chen, O.C.-H., Bernstein, D.J., Chen, J.-M.: Analysis of QUAD. In: FSE 2007, LNCS, vol. 4593, pp. 290–308. Springer (2007)

[28] Wiesinger-Widi, M.: Gröbner Bases and Generalized Sylvester Matrices. PhD thesis, Johannes Kepler University Linz (2015)

[29] Wu, W.-T.: Basic Principles of Mechanical Theorem Proving in Elementary Geometries. J. Autom. Reason. **2**(3), pp. 221–252 (1986)

A Experimental Results

Herein, we implemented the proposed algorithm in the Magma algebra system (V2.24-82) [6], where this implementation is not optimized. In this appendix, we mainly discuss the implementation of the **Multiply** and **Linearize(1)** steps (in particular, the last four steps from **Fix** to **Repeat** can be more optimized).

Figure 1 compares the execution time of the **Multiply** and **Linearize(1)** steps of our implementation and the number of manipulations estimated by (4.10) on the system with $n = m$ over $\mathbb{F}_{2^4}$. Here, the number k of fixed variables is chosen so as to minimize the value of (4.11). As a result, Figure 1

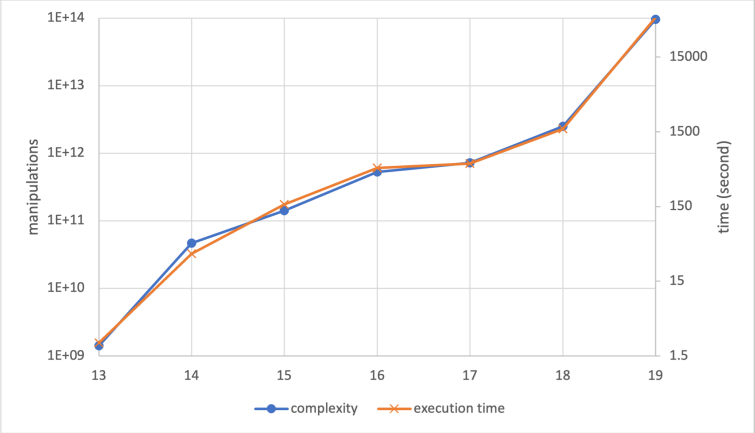


Fig. 1: Comparison between the estimation of complexity by (4.10) and the execution time of the **Multiply** and **Linearize(1)** steps on an $\mathcal{MQ}$ system with $n = m$ over $\mathbb{F}_{2^4}$

shows that the execution time and our estimation (4.10) from $n = 13$ to $n = 19$ have almost the same behavior, which indicates that the estimation of (4.10) is reliable.

B Correctness of the XL algorithm

We shall prove the correctness of XL (Algorithm 3): For sufficiently large D, XL definitely computes one root of the input system. To prove the correctness, we first discuss when a Gröbner basis is outputted by Algorithm 2, which is used as a main sub-routine of XL (see Step (2) of Algorithm 3).

Theorem 13 ([28], Theorem 2.3.3)**.** *Let $F = \{f_1, \ldots, f_m\} \subset K[x]$ be a set of ordered polynomials, and H a Gröbner basis of the ideal $\langle F \rangle \subset K[x] = K[x_1, \ldots, x_n]$. Let S be a finite subset of $[x] \cdot F$ such that for all $h \in H$, there exist $q_1, \ldots, q_m \in K[x]$ with $h = \sum_{i=1}^{m} q_i f_i$ and $\mathrm{supp}(q_i) \cdot \{f_i\} \subset S$ for all $1 \leq i \leq m$. Then, the output G of Algorithm 2 is a Gröbner basis of $\langle F \rangle$.*

Proof. Put $H = \{h_1, \ldots, h_\ell\}$ with $h_i \in K[x]$ for $1 \leq i \leq \ell$. By our assumption, for each $1 \leq i \leq \ell$, there exist $q_{ij} \in K[x]$ with $1 \leq j \leq m$ such that $h_i = \sum_{j=1}^{m} q_{ij} f_j$ and $\mathrm{supp}(q_{ij}) \cdot \{f_j\} \subset S$ for all $1 \leq j \leq m$. Thus we have

$$\bigcup_{i=1}^{\ell} \bigcup_{j=1}^{m} \mathrm{supp}(q_{ij}) \cdot \{f_j\} = \bigcup_{i=1}^{\ell} \bigcup_{j=1}^{m} \{t \cdot f_j : t \in \mathrm{supp}(q_{ij})\} \subset S,$$

and hence

$$h_i = \sum_{j=1}^{m} \sum_{t \in \mathrm{supp}(q_{ij})} \mathrm{coeff}(q_{ij}, t) \cdot t \cdot f_j \in \sum_{g \in S} K \cdot g,$$

where $\sum_{g \in S} K \cdot g$ denotes the set of all K-linear combinations of finite elements in S. Regarding the Macaulay matrix of $\{h_i\}$ with respect to T as a row vector in $K^{\#T}$, we have that it belongs to the linear space generated by row vectors of $\mathrm{Mac}(S, T)$. Thus, putting $G = \{g_1, \ldots, g_{\ell'}\}$, we can write $h_i = \sum_{k=1}^{\ell'} a_{i,k} g_k$ for some $a_{i,k} \in K$. It follows from the definition of a reduced row echelon form that $\mathrm{LT}(g) \neq \mathrm{LT}(g')$ for $g, g' \in G$ with $g \neq g'$. This implies that for each $1 \leq i \leq \ell$, there exists k such that $\mathrm{LT}(h_i) = a_{i,k}\mathrm{LM}(g_k)$. Therefore $\mathrm{LM}(H) \subset \mathrm{LM}(G)$, by which we have $\langle \mathrm{LT}(I) \rangle = \langle \mathrm{LT}(H) \rangle \subset \langle \mathrm{LT}(G) \rangle$. From the construction of G, we also have $\langle \mathrm{LT}(G) \rangle \subset \langle \mathrm{LT}(I) \rangle$, and thus $\langle \mathrm{LT}(I) \rangle = \langle \mathrm{LT}(G) \rangle$. □

Here, we define the Dubé's degree bound of polynomials in a shift inputted for Algorithm 2; for positive integers n and d, the function $\mathrm{Dube}_{n,d}(j)$ with $0 < j \leq n-1$ is defined recursively as

$$\begin{aligned} \mathrm{Dube}_{n,d}(n-1) &= 2d, \\ \mathrm{Dube}_{n,d}(n-2) &= d^2 + 2d, \\ \mathrm{Dube}_{n,d}(j) &= 2 + \tbinom{\mathrm{Dube}_{n,d}(j+1)}{2} + \sum_{i=j+3}^{n-1} \tbinom{\mathrm{Dube}_{n,d}(i)}{i-j+1}. \end{aligned}$$

Theorem 14 ([21], Section 5, Theorem 5 and cf. [14], Theorem 8.2)**.** *Let $F \subset K[x]$ be a finite set of polynomials, and put $d := \max\{\deg(f) : f \in F\}$. Then, for every monomial order $\succ$, there exists the reduced Gröbner basis H of $\langle F \rangle$ with respect to $\succ$ such that every $h \in H$ satisfies $\deg(h) \leq \mathrm{Dube}_{n,d}$.*

Corollary 15 ([21], Section 5, Corollary 2 and cf. [3], Corollary 5.4)**.** *Let $F = \{f_1, \ldots, f_m\} \subset K[x]$ be a finite set of polynomials, and put $d := \max\{\deg(f_i) : 1 \leq i \leq m\}$. Then, for every monomial order $\succ$, there exists a Gröbner basis H of $\langle F \rangle$ with respect to $\succ$ such that:*

- *For every $h \in H$, there exist $q_1, \ldots, q_m \in K[x]$ with $h = \sum_{i=1}^{m} q_i f_i$, and $\deg(q_i f_i) \leq \mathrm{Dube}_{n+1,d}$ for all $1 \leq i \leq m$.*

Corollary 16. *With the same notation as in Theorem 15, let S be a finite subset of $[x] \cdot F$ such that*

$$\bigcup_{i=1}^{m} \{t \cdot f_i : t \in \mathrm{supp}(q_i),\ \deg(t \cdot f_i) \leq \mathrm{Dube}_{n+1,d}\} \subset S.$$

Then, the output G of Algorithm 2 is a Gröbner basis of $\langle F \rangle$.

For example, it suffices to take S to be

$$S_1 := \bigcup_{i=1}^{m} \{t \cdot f_i : t \in [x],\ \deg(t) \leq \mathrm{Dube}_{n+1,d} - \deg(f_i)\}.$$

Proof. Let $h \in H$, and $q_1, \ldots, q_m \in K[x]$ with $h = \sum_{i=1}^{m} q_i f_i$, and $\deg(q_i f_i) \leq \mathrm{Dube}_{n+1,d}$ for all $1 \leq i \leq m$. By Theorem 13, it suffices to show that $\mathrm{supp}(q_i) \cdot \{f_i\} \subset S$ for all $1 \leq i \leq m$. Let $t \cdot f_i$ be an arbitrary element in $\mathrm{supp}(q_i) \cdot \{f_i\}$, where $t \in \mathrm{supp}(q_i)$. It follows from the choice of H that we have $\deg(t \cdot f_i) \leq \deg(q_i \cdot f_i) \leq \mathrm{Dube}_{n+1,d}$, so that $t \cdot f_i \in S$ as desired. □

Example 17. When input polynomials are all quadratic, say $\deg(f_i) = 2$ for all $1 \leq i \leq m$, it suffices for computing a Gröbner basis by Algorithm 2 to take S to be

$$S_1 = \{t \in [x] : \deg(t) \leq D - 2\} \cdot F,$$

where we set $D = \mathrm{Dube}_{n+1,2}$.

Proposition 18 (Correctness of the XL algorithm)**.** *If the ideal $\langle F \rangle$ is zero-dimensional, i.e., $V(F)$ is finite, then the XL algorithm (Algorithm 3) outputs a root of the input system for sufficiently large D.*

Proof. Put $d := \max\{\deg(f) : f \in F\}$, and choose D so that $D \geq \mathrm{Dube}_{n+1,d}$. By Corollary 16, the **Linearize** step computes a Gröbner basis with respect to an elimination monomial order such that all the terms containing one variable (say x_n) are eliminated last. Since $\langle F \rangle$ is zero-dimensional, it follows from [20, Lemma 2.3.2] that G contains a univarite polynomial in $K[x_n] \setminus \{0\}$. □