Relation between XL algorithm and Gröbner Bases Algorithms

Makoto Sugita[1], Mitsuru Kawazoe[2], and Hideki Imai[3]

[1] IT Security Center, Information-technology Promotion Agency, Japan
2-28-8 Honkomagome, Bunkyo-ku Tokyo, 113-6591, Japan
m-sugita@ipa.go.jp
[2] Department of Mathematics and Information Sciences,
College of Integrated Arts and Sciences,
Osaka Prefecture University,
1-1 Gakuen-cho Sakai Osaka 599-8531 Japan
kawazoe@mi.cias.osakafu-u.ac.jp
[3] Institute of Industrial Science, University of Tokyo
4-6-1 Kamaba, Meguro-ku Tokyo, 153-8505, Japan
imai@iis.u-tokyo.ac.jp

Abstract. We clarify a relation between the XL algorithm and Gröbner bases algorithms. The XL algorithm was proposed to be a more efficient algorithm to solve a system of equations with a special assumption without trying to calculate a whole Gröbner basis. But in our result, it is shown that the XL algorithm is also a Gröbner bases algorithm which can be represented as a redundant version of a Gröbner bases algorithm F_4 under the assumption in XL.

keywords : Algebraic attacks, Gröbner bases algorithm, F_4, XL algorithm

1 Introduction

In this paper we examine a relation between the XL algorithm used in an algebraic attack and Gröbner bases algorithms as a problem of solving systems of multivariate polynomial equations. Algebraic attack is one of the most efficient algorithm for public key cryptosystems, block ciphers and stream ciphers. Algebraic attack was first applied to Matsumoto-Imai Public Key Scheme in [3] by Jacques Patarin and a similar attack was also applied in [4]. In [5], the XL algorithm was first introduced and applied to HFE which is an improved version of Matsumoto-Imai Public Key Scheme. This attack was improved in [7] and the complexity of the XL algorithm was evaluated in detail in [6]. Algebraic attack was also applied to block ciphers in [8], where the complexity for attacking AES and Serpent was evaluated. Moreover, the algebraic attack was applied to stream cipher in [9], [10], [11] and improved in [12].

F_4 and F_5 algorithm were introduced by Jean-Charles Faugère in [13] and [14], respectively. These algorithms are the fastest previously known. Using these

algorithms, 80-bit HFE were first cryptanalyzed in [15], whereas the XL algorithm was not applicable to 80-bit HFE.

In [6], it is noted that the most efficient variant of this algorithm which we are aware of is due to Jean-Charles Faugère, and its complexity in the case of $m = n$ quadratic equations is:

- If K is big, the complexity is proved to be $\mathcal{O}(2^{3n})$ and is $\mathcal{O}(2^{2.7n})$ in practice.
- When $K = \mathrm{GF}(2)$, the complexity is about $\mathcal{O}(2^{2n})$ (which is worse than the $\mathcal{O}(2^{2n})$ complexity of exhaustive search).

XL algorithm was proposed as a technique which can be viewed as a combination of bounded degree Gröbner bases and linearization. The basic idea of this technique is to generate from each polynomial equation a large number of higher degree variants by multiplying it with all the possible monomials of some bounded degree, and then to linearize the expanded system. In [6], the time complexity of the XL technique was analyzed and it was proposed that they had provided a strong theoretical and practical evidence that the expected running time of this technique is:

- Polynomial when the number m of (random) equations is at least ϵn^2, and this for all $\epsilon > 0$
- Subexponential if m exceeds n even by a small number.

which is much better than that of F_4 algorithms.

In this report, we clarify relations between the XL algorithm and Gröbner base algorithms. In our result, the XL algorithm is proved to be also a Gröbner bases algorithm which can be expressed as a redundant version of a Gröbner bases algorithm F_4. The XL algorithm was proposed to be a more efficient algorithm to solve a system of equations with a special assumption without trying to calculate a whole Gröbner basis. But our result implies that the XL algorithm is not so efficient as it was expected. Moreover, we also treat XSL and show that the "T' method" in XSL can be interpreted in terms of Buchberger's algorithm by using "Toy Example" in [8].

The rest of this report is as follows. In Section 2, we recall the description of XL algorithm. In Section 3 and Section 4, we give an overview of the theory of Gröbner bases and F_4 algorithm, respectively. In Section 5, we clarify a relation between the XL algorithm and the F_4 algorithm. In Section 6, we conclude this report and in Appendix, we consider a relation between XSL and Buchberger's algorithm.

2 Description of XL algorithm

Here we recall the description of XL algorithm introduced in [6].

Let k be a finite field, and let $\mathcal{A}$ be a system of multivariate equations $l_j = 0 (1 \leq j \leq m)$ where each l_j is the multivariate polynomial $f_j(x_1, \cdots, x_n) - b_j$ for $f_j \in k[\mathbf{x}] := k[x_1, \ldots, x_n]$ and $b_j \in k$. We assume that a system of equation $\mathcal{A}$ has a unique solution $(x_1, \ldots, x_n) = (a_1, \ldots, a_n) \in k^n$.

We say that the equations of the form $\prod_{j=1}^{r} x_{i_j} * l_j = 0$ are of type $\mathbf{x}^r l$, and we call $\mathbf{x}^r l$ the set of all these equations. We also denote by $\mathbf{x}^r$ the set of all monomials of degree exactly r, $\prod_{j=1}^{r} x_{i_j}$.

Let $\mathbb{N}$ be the set of positive integers and $D \in \mathbb{N}$. We consider all the polynomials $\prod_j x_{i_j} l_j$ of total degree $\leq D$ and let $\mathcal{I}_D$ be the linear space spanned by these polynomials. Then, $\mathcal{I}_D$ is the linear space spanned by all the $x^r l$ with $0 \leq r \leq D-1$ and total degree $\leq D$. Moreover $\mathcal{I}_D \subset \mathcal{I}$, where $\mathcal{I}$ is the ideal generated by l_i's.

XL algorithm is given as an algorithm which solves systems of quadratic equations having the unique solution in k^n. It is described as follows [6]:

Definition 1. *XL algorithm is described as follows.*

1. **multiply:** *Generate all the products* $\prod_{j=1}^{r} x_{i_j} * l_i \in \mathcal{I}_D$ *with* $r \leq D-2$ *and total degree* $\leq D$.
2. **Linearize:** *Consider each monomial in the* x_i *of degree* $\leq D$ *as a new variable and perform Gaussian elimination on the equations obtained in 1. The ordering on the monomials must be such that all the terms containing one variable (say* x_1*) are eliminated last.*
3. **Solve:** *Assume that step 2 yields at least one univariate equation in the powers of* x_1*. Solve this equation over the finite fields (e.g., with Berlekamp's algorithm).*
4. **Repeat:** *Simplify the equations and repeat the process to find the values of the other variables.*

In the original definition of XL in [6], only quadratic equations are treated. There is no reason to restrict our interest to a system of quadratic equations. But to deal a system of equations including a non-quadratic equation, we have to replace $D-2$ to $D-1$ in the above description. In our discussion hereafter, we change the value $D-2$ to $D-1$ in order to work in general case. Note that this change does not contradict to the original XL setting when a system of equations consists of quadratic equations.

3 Gröbner basis and Gröbner bases algorithm

In this section, we recall the basic notion of Gröbner basis and Buchberger's algorithm which calculates a Gröbner basis.

3.1 Basic notations and definitions[2]

Let $k[\mathbf{x}] = k[x_1, \ldots, x_n]$ be a polynomial ring with variables $x_1, \ldots, x_n$ over a field k.

A *monomial* in $x_1, \ldots, x_n$ is a product of the form $x_1^{\alpha_1} \cdots x_n^{\alpha_n}$ for nonnegative integers $\alpha_1, \ldots, \alpha_n$. For the simplicity of the notation, we write $\mathbf{x}^\alpha = x_1^{\alpha_1} \cdots x_n^{\alpha_n}$ for an n-tuple $\alpha = (\alpha_1, \ldots, \alpha_n)$. For a monomial $\mathbf{x}^\alpha = x_1^{\alpha_1} \cdots x_n^{\alpha_n}$, $|\alpha| := \sum_{i=1}^{n} \alpha_i$ is called the *total degree* of this monomial. In the following, the set

of all monomials in variables $x_1, \ldots, x_n$ is denoted by $M(x_1, \cdots, x_n)$, or simply by M. In the theory of Gröbner bases, we need to consider a *monomial ordering* (cf. [2]). One of such ordering is the *lexicographic order* defined as follows:

Definition 2. *For $\alpha = (\alpha_1, \ldots, \alpha_n), \beta = (\beta_1, \ldots, \beta_n) \in \mathbb{Z}^n_{\geq 0}$, We say $\mathbf{x}^\alpha >_{lex} \mathbf{x}^\beta$ if the left-most nonzero entry of the vector $\alpha - \beta \in \mathbb{Z}^n$ is positive.*

Another one is the *graded lexicographic order*:

Definition 3. *For $\alpha, \beta \in \mathbb{Z}^n_{\geq 0}$, We say $\mathbf{x}^\alpha >_{glex} \mathbf{x}^\beta$ if $|\alpha| = \sum_{i=1}^n \alpha_i > |\beta| = \sum_{i=1}^n \beta_i$, or $|\alpha| = |\beta|$ and $\alpha >_{lex} \beta$.*

There are many monomial orderings. We choose one of such orderings on T and write it as $<$.

A nonzero polynomial f in $k[\mathbf{x}]$ is written as $f = \sum_\alpha c_\alpha \mathbf{x}^\alpha$ where each c_α is a nonzero element of k. Here $c_\alpha \mathbf{x}^\alpha$ is called a *term* of f. For $f = \sum_\alpha c_\alpha \mathbf{x}^\alpha \in k[\mathbf{x}]$, we use the following notations:

$T(f) = \{c(\alpha_1, \cdots, \alpha_n) x_1^{\alpha_1} \cdots x_n^{\alpha_n} \,|\, c(\alpha_1, \cdots, \alpha_n) \neq 0\}$: the set of terms of f

$M(f) = \{x_1^{\alpha_1} \cdots x_n^{\alpha_n} \,|\, c(\alpha_1, \cdots, \alpha_n) \neq 0\}$: the set of monomials of f

Then for $f = \sum_\alpha c_\alpha \mathbf{x}^\alpha$, we define the *total degree* $\deg(f)$, the *multidegree* $\mathrm{multdeg}(f)$, the *leading monomial* $\mathrm{LM}(f)$, the *leading coefficient* $\mathrm{LC}(f)$ and the *leading term* $\mathrm{LT}(f)$ of f with respect to $<$ as follows: $\deg(f) := \max\{|\alpha| = \sum_{i=1}^n \alpha_i \,|\, c_\alpha \neq 0\}$, $\mathrm{multdeg}(f) := \max\{\alpha \in \mathbb{Z}^n_{\geq 0} | c_\alpha \neq 0\}$, $\mathrm{LM}(f) := \max(M(f))$, $\mathrm{LC}(f) :=$ the coefficient of $\mathrm{LT}(f)$, $\mathrm{LT}(f) := \max(T(f)) = \mathrm{LC}(f) \cdot \mathrm{LM}(f)$ And for a finite subset F of $k[\mathbf{x}]$, we define $\mathrm{LT}(F) = \{\mathrm{LT}(f) | f \in F\}$, $\mathrm{LM}(F) = \{\mathrm{LM}(f) | f \in F\}$ and $\mathrm{M}(F) = \{\mathrm{M}(f) | f \in F\}$.

The ideal in $k[\mathbf{x}]$ generated by a subset F is denoted by $\langle F \rangle$. We also denote by $\langle I_1, \ldots, I_n \rangle$ the minimal ideal containing ideals $I_1, \ldots, I_n$.

3.2 Definition of Gröbner basis

In the theory of Gröbner bases, we need a notion of a division of a polynomial by a finite set of polynomials.

Definition 4. *Fix an order for $M(\mathbf{x}) = M(x_1, \ldots, x_n)$ and let $F = (f_1, \ldots, f_m)$ be an ordered set of polynomials in $k[\mathbf{x}]$. Then any $f \in k[\mathbf{x}]$ can be written as $f = a_1 f_1 + \cdots + a_m f_m + r$ where $a_i, r \in k[\mathbf{x}]$ such that no non-zero term of r is divisible by any of $\mathrm{LM}(f_1), \ldots \mathrm{LM}(f_m)$. We call this r a remainder of f on division by F and write it as $\overline{f}^F$.*

We should note that $\overline{f}^F$ is not unique. Now we define a *Gröbner basis*:

Definition 5. *Let M be the set of all monomial of $k[\mathbf{x}]$ and $>$ a fixed order of M. A finite subset $G = \{g_1, \ldots, g_m\}$ of an ideal $\mathcal{I}$ is called a Gröbner basis if*

$$\langle \mathrm{LT}(g_1), \ldots, \mathrm{LT}(g_m) \rangle = \langle \mathrm{LT}(\mathcal{I}) \rangle.$$

For a given ideal $\mathcal{I}$, its Gröbner basis is not unique. But the *reduced Gröbner basis*, which is defined as follows, is uniquely determined.

Definition 6. *A Gröbner basis $G = \{f_1, \dots f_m\}$ of an ideal $\mathcal{I}$ is called reduced Gröbner basis if for all i,* $\mathrm{LC}(f_i) = 1$ *and* $\mathrm{LM}(f_i)$ *is not divisible by any element of* $\mathrm{LM}(\mathrm{G}\backslash\mathrm{f_i})$.

3.3 Buchberger's algorithm

An algorithm which calculates a Gröbner basis is called a *Gröbner bases algorithm*. The Buchberger's algorithm is one of them. It is based on the Buchberger's criterion for when a basis of an ideal is a Gröbner basis. We begin with the following definition:

Definition 7. *Let $f, g \in k[\mathbf{x}]$ be nonzero polynomials. The S-polynomial of f and g is the combination*

$$S(f,g) := \mathrm{LC}(g)\frac{\mathrm{lcm}(\mathrm{LM}(f), \mathrm{LM}(g))}{\mathrm{LM}(f)} f - \mathrm{LC}(f)\frac{\mathrm{lcm}(\mathrm{LM}(f), \mathrm{LM}(g))}{\mathrm{LM}(g)} g.$$

The Buchberger's criterion is as follows:

Theorem 1. *A basis $G = \{g_1, \dots g_m\}$ of an ideal $\mathcal{I}$ in $k[\mathbf{x}]$ is a Gröbner basis if and only if for all pairs $i \neq j$, $\overline{S(g_i, g_j)}^{G} = 0$.*

As a result of Theorem 1, we have the following algorithm which calculates a Gröbner basis of a given ideal in $k[\mathbf{x}]$, called the *Buchberger's algorithm*:

Buchberger's algorithm
Input: an ordered set $F = (f_1, \dots, f_m)$ in $k[\mathbf{x}]$
Output: a Gröbner basis $G = (g_1, \dots, g_s)$ for $I = \langle f_1, \dots, f_m \rangle$ with $F \subset G$
$G := F$
Repeat
  $H := G$
  For each pair (p, q), $p \neq q$ in H,
    If $S := \overline{S(p,q)}^{H} \neq 0$, Then $G := G \cup \{S\}$
Until H:=G

We remark that the reduced Gröbner basis is calculated by finitely many steps from a Gröbner basis.

4 F_4 algorithm [13]

In this section, we recall a Gröbner bases algorithm F_4 given by Faugère [13]. The F_4 algorithm uses linear algebra in an efficient way. So we begin with a matrix representation of a given list of polynomials. In the following, by convention, for an $s \times m$ matrix A, the jth element of the ith row of A is denoted by A_{ij} and the ith row of a matrix A is denoted by $\mathrm{row}(A, i)$. For a matrix A or a vector $\mathbf{v}$, the transpose of each is denoted by ${}^tA$, ${}^t\mathbf{v}$ respectively.

Any finite list of polynomials in $k[\mathbf{x}]$ is represented by a pair of a matrix and an ordered subset of T as follows. For a given finite list $L = (f_1, \ldots, f_s)$, let $M_L = [t_1, \ldots, t_m]$ be an ordered set of monomials of all polynomials in L. Note that as a set, $M_L = M(L)$. Put A_{ij} as the coefficient of t_j in f_i for $i = 1, \ldots, s, j = 1, \ldots, m$ and construct an $s \times m$ matrix $A = (A_{ij})$. Then L is represented by (A, M_L) as

$$^tL = A\, ^tM_L.$$

The matrix A in the above is called *the coefficient matrix* of L with respect to M_L and denoted by $A^{(L,M_L)}$. Conversely for any pair (A, X) of an $s \times m$ matrix A and an ordered subset X of M with m elements, we define

$$\mathrm{Rows}(A, X) := X\, ^t\widehat{A}$$

where $\widehat{A}$ is the matrix obtained from A by removing rows whose all elements are zero. Note that $\mathrm{Rows}(A, X) = \{\mathrm{row}(A, i)\, ^tX \mid \mathrm{row}(A, i) \neq \mathbf{0}\}$ as sets.

By using the above matrix representation, we define *the row echelon basis* of a given finite subset of $k[\mathbf{x}]$: For the coefficient matrix $A = A^{(L,M_L)}$ of L with respect to M_L, let $\tilde{A}$ be the row echelon form of A obtained by using elementary row operations in a standard linear algebra[4]. Then we say that $\tilde{L} = \mathrm{Rows}(\tilde{A}, M_L)$ is the *row echelon basis* of L. When $L = \mathit{Sort}(\{M(f) | f \in F\}, <)$ for a given finite subset F of $k[\mathbf{x}]$ and an ordering $<$, $\tilde{F} := \tilde{L}$ is called *the row echelon basis* of F with respect to $<$. When we take the reduced row echelon form of L, we say $\tilde{L}$ *the reduced row echelon basis* of L (In [13], this is called the row echelon basis).

Note that, by the definition, the ideal generated by $\mathrm{Rows}(\tilde{A}, M_L)$ is the same ideal generated by L, that is,

$$\langle \mathrm{Rows}(\tilde{A}, L) \rangle = \langle L \rangle.$$

Example 1. The following example illustrates the relation between $L = M_<(F)$, $A^{(L,M_L)}$ and the row echelon basis of F.

Let $f_1 = x^3 - x^2y - 3y^2 + x$, $f_2 = x^3 + y^2 + x$, $f_3 = x^3 + x^2y + 3y^2 + x$ and $L = (f_1, f_2, f_3)$. When we choose the graded lexicographic order, $M_L = (x^3, x^2y, y^2, x)$.

$$\begin{cases} f_1 = x^3 - x^2y - 3y^2 + x \\ f_2 = x^3 + y^2 + x \\ f_3 = x^3 + x^2y + 3y^2 + x \end{cases} \longleftrightarrow \begin{pmatrix} 1 & -1 & -1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 3 & 1 \end{pmatrix}$$

$$\downarrow \text{Gaussian elimination}$$

$$\begin{cases} g_1 = x^3 - x^2y - 3y^2 + x \\ g_2 = x^2 + 2y^2 \\ g_3 = 0 \end{cases} \longleftrightarrow \begin{pmatrix} 1 & -1 & -1 & 1 \\ 0 & 1 & 2 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Hence we have the row echelon basis $\tilde{L} = (g_1, g_2, g_3)$. If we take the reduced row echelon form of the coefficient matrix of L, we have the reduced row echelon basis $(x^3 - y^2 + x, x^2 + 2y^2)$ of L.

[4] This procedure is so-called the Gaussian elimination.

The following proposition, which directly comes from elementary properties of row echelon forms, shows the importance of row echelon bases.

Proposition 1. *[13, Corollary 2.1] Let F be a finite subset of $k[\mathbf{x}]$, $<$ an ordering and $\tilde{F}$ the row echelon basis of F with respect to $<$. We define*

$$\tilde{F}^+ = \{g \in \tilde{F} | \mathrm{LM}(g) \notin \mathrm{LM}(F)\}.$$

For all subset F_- of F such that size(F_-) = size$(\mathrm{LM}(F))$ and $\mathrm{LM}(F_-) = \mathrm{LM}(F)$, then $G = \tilde{F}^+ \cup F_-$ is a triangular basis of the k-module V_F generated by F. For all $f \in V_F$ there exist $(\lambda_j)_j$ elements of k and $(g_j)_j$ elements of G such that $f = \sum_j \lambda_j g_j$, $\mathrm{LM}(g_1) = \mathrm{LM}(f)$ and $\mathrm{LM}(g_j) > \mathrm{LM}(g_{j+1})$.

To describe the F_4 algorithm, we need the following definition.

Definition 8. *(1) A critical pair of two polynomials (f_i, f_j) is an element of $M^2 \times k[\mathbf{x}] \times M \times k[\mathbf{x}]$, $Pair(f_i, f_j) := (lcm_{ij}, t_i, f_i, t_j, f_j)$ such that*

$$lcm(Pair(f_i, f_j)) = lcm_{ij} = \mathrm{LM}(t_i f_i) = \mathrm{LM}(t_j f_j) = lcm(\mathrm{LM}(f_i), \mathrm{LM}(f_j)).$$

(2) For a critical pair $p_{ij} = Pair(f_i, f_j)$, lcm_{ij} is called the degree of p_{ij} and denoted by $\deg(p_{ij})$. We define the two projections $Left(p_{ij}) := (t_i, f_i)$ and $Right(p_{ij}) = (t_j, f_j)$. For a set P of critical pairs, we write $Left(P) = \bigcup_{p_{ij} \in P} Left(p_{ij})$ and $Right(P) = \bigcup_{p_{ij} \in P} Right(p_{ij})$.
(3) For $f \in k[\mathbf{x}]$ and a finite set G in $k[\mathbf{x}]$, we say that f is top reducible modulo G if there exist $g \in G$ and $q, r \in k[\mathbf{x}]$ such that $f = qg + r$ and $\mathrm{LM}(r) < \mathrm{LM}(f)$.

Now we describe the F_4 algorithm [13].
Algorithm F_4

Input: $\begin{cases} F : \text{a finite subset of } k[\mathbf{x}] \\ Sel : \text{a function } \mathrm{List}(Pairs) \to \mathrm{List}(Pairs) \end{cases}$
Output: a finite subset of $k[\mathbf{x}]$.
$G := F$, $\tilde{F}_0^+ := F$ and $d := 0$
$P := \{Pair(f, g) | f, g \in G \text{ with } f \neq g\}$
While $P \neq \phi$ Do
  $d := d + 1$
  $P_d := Sel(P)$
  $P := P \setminus P_d$
  $L_d := \mathrm{Left}(P_d) \cup \mathrm{Right}(P_d)$
  $\tilde{F}_d^+ := Reduction(L_d, G)$
  For $h \in \tilde{F}_d^+$ Do
    $P := P \cup \{Pair(h, g) | g \in G\}$
    $G := G \cup \{h\}$
Return G

Reduction

Input: $\begin{cases} L : \text{a finite subset of } M \times k[\mathbf{x}] \\ G : \text{a finite subset of } k[\mathbf{x}] \end{cases}$

Output: a finite subset of $k[\mathbf{x}]$ (possibly an empty set).
$F :=$ Symbolic Preprocessing(L, G)
$\tilde{F} :=$ Reduction to Row Echelon Basis of F w.r.t. $<$
$\tilde{F}^+ := \{f \in \tilde{F} | \mathrm{LM}(f) \notin \mathrm{LM}(F)\}$
Return $\tilde{F}^+$

Symbolic Preprocessing

Input: $\begin{cases} L : \text{a finite subset of } M \times k[\mathbf{x}] \\ G : \text{a finite subset of } k[\mathbf{x}] \end{cases}$
Output: a finite subset of $k[\mathbf{x}]$
$F := \{t * f | (t, f) \in L\}$
$Done := \mathrm{LM}(F)$
While $M(F) \neq Done$ Do
$Done := Done \cup \{m\}$ where $m \in M(F) \backslash Done$
If m is top reducible modulo G Then
$m = m' * \mathrm{LM}(f)$ for some $f \in G$ and some $m' \in M$
$F := F \cup \{m' * f\}$
Return F

In [13], Faugère proved the following theorem.

Theorem 2. *[13] For a finitely generated ideal $\mathcal{I} = \langle F \rangle$ in $k[\mathbf{x}]$, The algorithm F_4 computes a Gröbner basis G of $\mathcal{I}$ such that $F \subseteq G$ and $\langle G \rangle = \langle F \rangle$.*

If we take the reduced row echelon basis in *Reduction* process, we have the reduced Gröbner basis.

In the F_4 algorithm, a choice of *Sel* is very important. The best function proposed in [13] is to take all the critical pairs with a minimal total degree: For a list P of critical pairs of a given set,

$$Sel(P) := \{p \in P | \deg(lcm(p)) = d\}$$

where $d := min\{\deg(lcm(p)), p \in P\}$.

In [13], the strategy of P derived from the above *Sel* is called the *normal strategy* for F_4. The F_4 algorithm gives a rapid Gröbner bases algorithm.

5 Relation between XL algorithm and F_4 algorithm

In this section, we show our main results.

5.1 Pre-assumption of XL algorithm

Here we consider the assumption on systems of equations treated in XL.

Let $k = \mathbb{F}_q$ be a finite field with q elements and let $\mathcal{A}$ be a system of multivariate equations $l_j = 0 (1 \leq j \leq m)$ where each l_j is the multivariate polynomial $f_j(x_1, \cdots, x_n) - b_j$ whose coefficients are all in k.

Pre-assumption for XL algorithm to work is described implicitly in Introduction of [6] as follows:

Condition 1 *The number of the given equations $\mathcal{A}$ is large enough to determine the values of $x_1, \cdots, x_n$ as $a_1, \cdots, a_n \in k$, respectively.*

In other words, Condition 1 says that $\mathcal{A}$ has the unique solution in k^n. Since equations are defined over a finite field $k = \mathbb{F}_q$, the ideal we have to consider is generated by $l_j \in \mathcal{A}$ and $x_i^q - x_i$'s. Thus, Condition 1 is equivalent to the following condition, in terms of Gröbner basis because of the uniqueness of the reduced Gröbner basis.

Condition 2 *The reduced Gröbner basis of the ideal generated by all equations in $\mathcal{A}$ and $x_i^q - x_i$, $i = 1, 2, \ldots, n$, is $\{x_1 - a_1, \cdots, x_n - a_n\}$.*

Proposition 2. *Let $\mathcal{A}$ be a system of multivariate equations in $k[x_1, \ldots, x_n]$ with $k = \mathbb{F}_q$. Let $\mathcal{I}_A$ the ideal generated by all equation in $\mathcal{A}$ and $x_i^q - x_i$ for $i = 1, 2, \ldots, n$. Then $\mathcal{A}$ can determine the value $(x_1, \cdots, x_n) = (a_1, \cdots, a_n) \in k^n$ if and only if $\{x_1 - a_1, \cdots, x_n - a_n\}$ is the reduced Gröbner basis of $\mathcal{I}_A$.*

Proof. This statement immediately follows from the uniqueness of the reduced Gröbner bases. □

As we show in the next subsection, XL algorithm is also a Gröbner bases algorithm. Therefore, the problem to solve $\mathcal{A}$ defined over $k = \mathbb{F}_q$ under the Condition 1 coincides with the problem to yield the reduced Gröbner bases of the ideal generated by $\mathcal{A}$ and field equation $x_i^q - x_i$ under the Condition 2, which is not a new problem.

5.2 Expression of the XL algorithm as a Gröbner bases algorithm

We use the same notation as in (3.1). For XL algorithm, we can give an F_4-like description whose strategy is given as a trivial strategy, that is, $Sel(P) := P$.

Definition 9. *Let P be a list of pairs of polynomials. For $p = (f, g) \in P$ and $d \in \mathbb{N}$, we define two functions $XLLeft(p, d) = \{(t, f) | t \in M, \deg(t * f) \leq d\}$, and $XLRight(p, d) = \{(t, g) | t \in M, \deg(t * g) \leq d\}$. We write $XLLeft(P, d) = \bigcup_{p \in P} XLLeft(p, d)$ and $XLRight(P, d) = \bigcup_{p \in P} XLRight(p, d)$.*

Now we give an F_4-like description of the XL algorithm.

XL Algorithm

Input: $\begin{cases} F : \text{a finite subset of } k[\mathbf{x}] \\ (\text{a function } Sel \text{ is fixed as } Sel(P) = P \text{ here!}) \end{cases}$
Output: a finite subset of $k[\mathbf{x}]$.
$G := F$, $\tilde{F}_0^+ := F$ and $d := 0$
$P := \{Pair(f, g) | f, g \in G \text{ with } f \neq g\}$
While $P \neq \phi$ Do
  $d := d + 1$
  $P_d := Sel(P)$
  $P := P \setminus P_d$

$L_d := \mathrm{XLLeft}(P_d, d) \cup \mathrm{XLRight}(P_d, d)$
$\tilde{F}_d^+ := \mathit{Reduction}(L_d, G)$
For $h \in \tilde{F}_d^+$ Do
$P := P \cup \{\mathit{Pair}(h, g) | g \in G\}$
$G := G \cup \{h\}$
Return G

Remark 1. In the original description of XL, it seems that the bound D is taken globally at once. However, to implement XL, there seems to be the following four ways to realize the process determining the optimal value of D. Let $\mathcal{A}$ be a system of equations you want to solve. Then each way is described as follows.

1. Begin with $D = 1$. Iterate the whole step of XL for $\mathcal{A}$ until you get the solution by increasing D to $D + 1$.
2. Begin with $D = 1$ and iterate XL for $\mathcal{A}$. If you can not obtain the solution for that D, set $D := D + 1$ and iterate the XL for $\mathcal{A}$ from the first step. Repeat this process until you obtain the solution.
3. Begin with $D = 1$. Iterate the XL until you get the solution by increasing D to $D + 1$, but in each iteration replace $\mathcal{A}$ to a system of equations obtained by XL.
4. Begin with $D = 1$. Iterate XL for $\mathcal{A}$. If you can not obtain the solution for that D, then replace $\mathcal{A}$ to a system of equations obtained by XL for D and iterate the XL for the new $\mathcal{A}$ with $D := D+1$. Repeat this process until you obtain the solution.

In the above description of XL, we take the third one. You may take one of the other three realizations but the rest of our result holds for all of them.

In the above description of the XL algorithm, we keep some redundancy in the description to show the similarity to the F_4 algorithm. Note that the input F must be a set which consists of all equations in a given system of equations $\mathcal{A}$ and all field equations $x_i^q - x_i$ for all variables x_i. Moreover, in the above description, we can omit Symbolic Preprocessing because all the products $\prod_{j=1}^{r} x_{i_j} * l_i \in \mathcal{I}_D$ with $r \leq D - 1$ and total degree $\leq D$ generated in 'multiply' include all the polynomials generated in Symbolic Preprocessing.

The above description enable us to prove the following theorem.

Theorem 3. *The algorithm XL computes a Gröbner basis G in $k[\mathbf{x}]$ such that $F \subseteq G$ and $\langle G \rangle = \langle F \rangle$.*

In the proof of this theorem, the following two lemmas are important.

Lemma 1. *[13] Let G be a finite subset of $k[\mathbf{x}]$, L the image by* mult $: M \times G \ni (m, f) \mapsto m * f \in k[\mathbf{x}]$ *of a finite subset of $M \times G$ and $\tilde{F}^+ = \mathit{Reduction}(L, G)$. Then for all $h \in \tilde{F}^+$, $LM(h) \notin \langle \mathrm{LM}(G) \rangle$.*

Lemma 2. *[13] Let G be a finite subset of $k[\mathbf{x}]$, L the image by* mult *of a finite subset of $M \times G$ and $\tilde{F}^+ = \mathit{Reduction}(L, G)$. Then $\tilde{F}^+$ is a subset of $\langle G \rangle$. Moreover, for all f in the k-module generated by L, $\overline{f}^{G \cup \tilde{F}^+} = 0$.*

Then the proof of Theorem 3 is almost the same as that of Theorem2 of [13].

6 Conclusion

We clarified relations between XL attack and Gröbner base algorithms. The XL algorithm can be represented as a redundant version of F_4. In [6], it is stated that XL does not try to calculate a Gröbner basis and therefore it should be more efficient. But it is not true because XL algorithm calculates a Gröbner basis for a system of equations with the assumption given in [6].

7 Appendix: Relation between XSL and Groöbner bases algorithm

Instead of the general technique XL form [6], Courtois and Pieprzyk [8] proposed a custom-made algorithm that takes the advantage of the specific structure of equations and of their sparsity, which is called XSL attack. In [8], it is stated that the security of AES and Serpent probably does not grow exponentially with the number of rounds, it seems that it could even be polynomial in the number of rounds of the cipher and it seems also to break Rijndael 256-bits and Serpent for key length 192 and 256 bits.

Here we show XSL is interpreted in terms of Gröbner bases algorithm by using the Toy Example for "T' method" in [8]. For the details of the description of "T' method", see [8].

In [8], "Toy Example" is given as follows. Here is a system in which T' is defined with respect to x_1:

$$\begin{cases} x_3x_2 = x_1x_3 + x_2 \\ x_3x_4 = x_1x_4 + x_1x_5 + x_5 \\ x_3x_5 = x_1x_5 + x_4 + 1 \\ x_2x_4 = x_1x_3 + x_1x_5 + 1 \\ x_2x_5 = x_1x_3 + x_1x_2 + x_3 + x_4 \\ x_4x_5 = x_1x_2 + x_1x_5 + x_2 + 1 \\ 0 \quad = x_1x_3 + x_1x_4 + x_1 + x_5 \\ 1 \quad = x_1x_4 + x_1x_5 + x_1 + x_5 \end{cases} \tag{1}$$

Let $>_1$ be a graded reverse lexicographic order which satisfies $x_5 >_1 x_4 >_1 x_3 >_1 x_2 >_1 x_1$. From the definition of a boolean function, $x_i^2 + x_i = 0$ are satisfied for any $i = 1, 2, 3, 4, 5$. In other words, the above system (1) contains the equations $x_i^2 + x_i = 0$, $i = 1, 2, 3, 4, 5$, implicitly.

Rewriting the system by using the order $>_1$, we obtain the following.

$$\begin{cases} x_2x_3 + x_1x_3 + x_2 = 0 \\ x_3x_4 + x_1x_5 + x_1x_4 + x_5 = 0 \\ x_3x_5 + x_1x_5 + x_4 + 1 = 0 \\ x_2x_4 + x_1x_5 + x_1x_3 + x_2 + 1 = 0 \\ x_4x_5 + x_1x_5 + x_1x_2 + x_2 + 1 = 0 \\ x_1x_5 + x_1x_3 + x_5 + x_1 = 0 \\ x_1x_5 + x_1x_4 + x_5 + x_1 + 1 = 0 \end{cases} \tag{2}$$

Put $f = x_1x_5 + x_1x_3 + x_5 + x_1$ and $g = x_1x_5 + x_1x_4 + x_5 + x_1 + 1$. Then multiplying the two equations $f = 0$ and $g = 0$ by x_1, we obtain the two new equations,

$$\begin{cases} x_1x_3 + x_1x_4 + x_1 + x_1x_5 = 0 \\ x_1x_4 = 0 \end{cases} \tag{3}$$

However, the calculation to obtain the first equation in (3) is interpreted as follows: *Calculate the S-polynomial of* f *and* $x_1^2+x_1$ *and then take* $\overline{S(f, x_1^2 + x_1)}^{\{x_1^2+x_1\}}$. That is essentially a part of Buchberger's algorithm. In fact, $S(f, x_1^2+x_1)$ reduces to $x_1x_3 + x_1x_4 + x_1 + x_1x_5$ modulo $x_1^2 + x_1$. In the same way, the calculation to obtain the second equation in (3) is interpreted as follows: *Calculate the S-polynomial of* g *and* $x_1^2 + x_1$ *and then take* $\overline{S(g, x_1^2 + x_1)}^{\{x_1^2+x_1\}}$.

A discussion when we consider other graded reverse lexicographic orders is the same. Here is the same system in which T' is defined with respect to x_2:

$$\begin{cases} x_1x_3 = x_3x_2 + x_2 \\ x_1x_4 = x_3x_2 + x_2 + x_1 + x_5 \\ x_1x_5 = x_2x_4 + x_3x_2 + x_2 + 1 \\ x_3x_5 = x_2x_4 + x_3x_2 + x_2 + 1 + x_4 + 1 \\ x_3x_4 = x_2x_4 + x_1 + 1 \\ x_4x_5 = x_1x_2 + x_2x_4 + x_3x_2 \\ 0 = x_1x_2 + x_2x_5 + x_3x_2 + x_2 + x_3 + x_4 \\ 0 = x_2x_4 \end{cases} \tag{4}$$

Let $>_2$ be a graded reverse lexicographic order which satisfies $x_5 >_2 x_4 >_2 x_3 >_2 x_1 >_2 x_2$. Note that from the definition of a boolean function, $x_i^2 + x_i = 0$ are satisfied for any $i = 1, 2, 3, 4, 5$.

Here we rewrite the systems using the order $>_2$.

$$\begin{cases} x_1x_3 + x_2x_3 + x_2 = 0 \\ x_1x_4 + x_2x_3 + x_5 + x_1 + x_2 = 0 \\ x_1x_5 + x_2x_4 + x_2x_3 + x_2 + 1 = 0 \\ x_3x_5 + x_2x_4 + x_2x_3 + x_4 + x_2 = 0 \\ x_3x_4 + x_2x_4 + x_1 + 1 = 0 \\ x_4x_5 + x_2x_4 + x_2x_3 + x_1x_2 = 0 \\ x_2x_5 + x_2x_3 + x_1x_2 + x_2 + x_3 + x_4 = 0 \\ x_2x_4 = 0 \end{cases} \tag{5}$$

In the T' method, applying the Gaussian elimination to the system consists of all equations in (5) and (3), we can find the following equations which can be multiplied by x_2.

$$\begin{cases} x_1x_2 + x_2x_5 + x_3x_2 + x_2 + x_3 + x_4 = 0 \\ x_2x_4 = 0 \\ x_2x_4 + x_3x_2 + x_5 + x_2 + 1 = 0 \\ x_3x_2 + x_2 + x_1 + x_5 = 0 \end{cases} \tag{6}$$

The above calculation essentially try to find equations whose terms of degree 2 are all in the form x_2x_i, $i = 1, 3, 4, 5$. However, the same equations can be easily obtained by sorting with the order $>_2$ and the Gaussian elimination. So the above process trying to find a new equation can be multiplied by x_2 is interpreted in terms of Gröbner bases algorithms as follows: *To get new equations which can be multiplied by x_2, sort terms in equations with the order $>_2$ and then execute the Gaussian elimination.*

Then multiplying each equation in (6) by x_2, in other words, calculating a remainder of a division of S-polynomials of each equation and $x_2^2+x_2$ by $x_2^2+x_2$, we obtain the following four equations.

$$\begin{cases} x_2x_5 + x_2x_4 + x_1x_2 + x_2 = 0 \\ x_2x_4 = 0 \\ x_2x_5 + x_4x_2 + x_3x_2 = 0 \\ x_2x_5 + x_2x_3 + x_1x_2 + x_2 = 0 \end{cases} \tag{7}$$

As it is pointed out in [8], we can not obtain a new equation from the second equation in (6) because it is invariant under multiplication by x_2. We can explain the reason in terms of S-polynomials: *This is because* $\overline{S(x_2x_4, x_2^2 + x_2)}^{\{x_2^2+x_2\}} = x_2x_4$.

In "T' method" for the above "Toy Example", iterating the above process, we can obtain the system of equations of the maximum rank. So the above argument shows that "T' method" in XSL can be interpreted in terms of Buchberger's algorithm.

References

1. T.Becker, V.Weispfenning, "Gröbner Bases, a Computational Approach to Commutative Algebra", Graduate Texts in Mathematics, Springer, Berlin, 1993.
2. D.Cox, J. Little, and D. O'Shea, "Using Algebraic Geometry," Springer-Verlag, New York, 1998.
3. Jacques Patarin "Cryptanalysis of the Matsumoto and Imai Public Key Scheme of Eurocrypt'88," Crypto'95, Springer, LNCS 963, pp. 248-261, 1995.
4. Aviad Kipnis, Jacques Patarin, Louis Goubin, "Unbalanced Oil and Vinegar Signature Schemes," Eurocrypt 1999, Springer-Verlag, pp. 216-222.
5. Aviad Kipnis, Adi Shamir: "Cryptanalysis of the HFE Public Key Cryptosystem," Proceedings of Crypto'99, Springer-Verlag.
6. Adi Shamir, Jacques Patarin, Nicolas Courtois, Alexander Klimov, "Efficient Algorithms for solving Overdefined Systems of Multivariate Polynomial Equations," Eurocrypt'2000, LNCS 1807, Springer, pp. 392-407.
7. Nicolas Courtois "The security of Hidden Field Equations (HFE)," Cryptographers' Track RSA Conference 2001, San Francisco 8-12 April 2001, LNCS 2020, Springer, pp. 266-281.
8. Nicolas Courtois and Josef Pieprzyk, "Cryptanalysis of Block Ciphers with Overdefined Systems of Equations," Asiacrypt 2002, LNCS 2501, Springer.
9. Nicolas Courtois, "Higher Order Correlation Attacks, XL algorithm and Cryptanalysis of Toyocrypt," ICISC 2002, LNCS 2587, Springer.

10. Nicolas Courtois and Willi Meier "Algebraic Attacks on Stream Ciphers with Linear Feedback," Eurocrypt 2003, Warsaw, Poland, LNCS 2656, pp. 345-359, Springer.
11. Nicolas Courtois "Fast Algebraic Attacks on Stream Ciphers with Linear Feedback", Crypto 2003, LNCS 2729, Springer.
12. Frederik Armknecht, Matthias Krause, "Algebraic Attacks on Combiners with Memory," Crypto 2003, LNCS 2729, pp. 162-176, Springer.
13. Jean-Charles Faugère, "A new efficient algorithm for computing Gröbner bases (F_4)," Journal of Pure and Applied Algebra 139 (1999) pp. 61-88.
14. Jean-Charles Faugère, "A new efficient algorithm for computing Gröbner basis without reduction to 0 F_5," In T. Mora, editor, Proceeding of ISSAC, pages 75-83, ACM Press, July 2002.
15. Jean-Charles Faugère and A. Joux, "Algebraic Cryptanalysis of Hidden Field Equation (HFE) Cryptsystems Using Gröbner bases," Crypto 2003, LNCS 2729, pp. 44-60, Springer.
16. M. Sugita and H. Imai, "Relations between Algebraic Attacks and Gröbner Base Algorithms", In The 2004 Symposium on Cryptography and Information Security, Japan – SCIS 2004, Jan.27–Feb.30, 2004.