How Much Can F5 really Do?

Jintai Ding, Zheng Zhang, and Joshua Deaton

University of Cincinnati, OH, USA
jintai.ding@gmail.com
zhang2zh@mail.uc.edu,
jdeaton1995@gmail.com

Abstract. Our purpose is to compare how much the F5 algorithm can gain in efficiency compared to the F4 algorithm. This can be achieve as the F5 algorithm uses the concept of signatures to foresee potential useless computation which the F4 algorithm might make represented by zero rows in the reduction of a large matrix. We experimentally show that this is a modest increase in efficiency for the parameters we tested.

1 Introduction

In Algebraic Geometry and its applications which includes problems in Robotics, Cryptography, etc., the fundamental problem is (efficiently) solving a system of m polynomial equations in n variables

$$f_1(x_1, x_2, \dots, x_n) = 0 \;\; f_2(x_1, x_2, \dots, x_n) = 0 \;\; \cdots \;\; f_m(x_1, x_2, \dots, x_n) = 0$$

where for each $i \in \{1, 2, \dots, m\}$ we have $f_i \in \mathbb{F}[x_1, x_2, \dots, x_n]$, the polynomial ring over a field $\mathbb{F}$ which throughout this paper we will assume to be finite.

There are cases in which efficient methods exist. In the case that each polynomial f_i is linear, meaning that it has no terms of degree greater than one, then the problem reduces to the techniques of linear algebra and may be solved by Gaussian elimination. More advanced techniques may be used if the linear polynomials have special form, like if they are sparse (which many coefficients being zero) in which case the Wiedermann algorithm would lead to greater efficiency. In the case that the system is univariate i.e. $n = 1$, then the best method is to first find the greatest common divisor of the polynomials

$$f(x) = gcd(f_1(x), f_2(x), \dots, f_m(x))$$

using the Extended Euclidean Algorithm. Then Berlekamp's Algorithm can be used to find a solution if one exists [1].

However, in the case that both the system is multivariate i.e. $n > 1$ and nonlinear, then finding a solution to the system is an NP-complete problem even when the system is only quadratic and $|\mathbb{F}| = 2$ [20]. The problem is believed to exponential for random system an average as well. The best current methods are the XL (eXtended Linearization) family originally by [6] and the F4 [16] and F5 [17] families by Faugere (really, F4/F5 are the start of solving the system with other algorithms needed later, but they are by for the most computationally intense part of finding the solution).

Both families of algorithms reduce the problem of finding a solution to that of solving a large, sparse matrix. While in overdetermined system XL often has the advantage [30], in other cases F4 and F5 are superior and are often the benchmark in describing the complexity of solving the system. F5 was made to be an improvement over F4 be reducing unnecessary computation done in F4, and in this paper we present some experimental results on how big of an improvement that this can be.

Both F4 and F5 (and their variants) are algorithms to designed to compute what is called a Groebner basis for the system

$$\mathscr{F} = (f_1(x_1, x_2, \ldots, x_n), f_2(x_1, x_2, \ldots, x_n), \cdots, f_m(x_1, x_2, \ldots, x_n)).$$

To describe what a Groebner basis, we first need to define some terms. We will follow the terminology of the excellent textbook *Ideals, Varieties, and Algorithms* by Cox et al[7] .

Definition 1. *A subset $I \subset \mathbb{F}[x_1, x_2, \ldots, x_n]$ is an ideal of $\mathbb{F}[x_1, x_2, \ldots, x_n]$ if*

1. $0 \in I$.
2. $f, g \in I \implies f + g \in I$.
3. $f \in I$ *and* $h \in \mathbb{F}[x_1, x_2, \ldots, x_n] \implies hf \in I$.

The Hilbert Basis Theorem shows, as $\mathbb{F}$ is a field and thus a Noetherian ring, an ideal I of $\mathbb{F}[x_1, x_2, \ldots, x_n]$ is finitely generated by a finite set of elements of elements

$$I = \langle f_1, f_2, \ldots, f_s \rangle := \left\{ \sum_{i=1}^{s} h_i f_i \;\middle|\; h_i \in \mathbb{F}[x_1, x_2, \ldots, x_n] \right\}.$$

Definition 2. *We define the variety of a basis $\{f_1, f_2, \ldots, f_s\} \subset \mathbb{F}[x_1, x_2, \ldots, x_n]$ to be the set*

$$V(f_1, f_2, \ldots, f_s) = \{a \in \mathbb{F}^n \mid f_i(a) = 0, 1 \le i \le s\}.$$

It is easy to show that if $\{f_1, f_2, \ldots, f_s\}$ and $\{g_1, g_2, \ldots, g_r\}$ are two bases for the ideal I, then

$$V(f_1, f_2, \ldots, f_s) = V(g_1, g_2, \ldots, g_r).$$

This changes the problem of trying to solve a polynomial system to finding a basis for its ideal from which it is easier to find solutions. We now define a order on the monomials of $\mathbb{F}[x_1, x_2, \ldots, x_n]$

Definition 3. *A monomial order $>$ on the set of monomials in $\mathbb{F}[x_1, x_2, \ldots, x_n]$ is a well ordered, total ordering such that*

$$\alpha > \beta \implies \alpha\gamma > \beta\gamma.$$

for all monomials α, β, γ.

The usual ordering given are the lexicographical ordering, the graded lexicographical ordering, and the reverse lexicographical ordering whose definitions can be found in [7]. Given a monomial ordering $>$ and a polynomial $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$, we may define the leading monomial of f $LM(f)$ to be the largest monomial of f with a nonzero coefficient, the leading coefficient $LC(f)$ to be the coefficient of $LM(f)$, and the leading term $LT(f) := LC(F)LM(f)$. For a set of polynomials F, we define $LT(F) := \{LT(f) \mid f \in F\}$.

1.1 Buchberger's Algorithm

We now may define a Groebner basis:

Definition 4. *For a monomial ordering $>$ in $\mathbb{F}[x_1, x_2, \ldots, x_n]$, a finite basis $\{f_1, f_2, \ldots, f_s\}$ of an ideal I is a Groebner basis of I is*

$$\langle LT(f_1), LT(f_2), \ldots, LT(f_s)\rangle = \langle LT(I)\rangle.$$

An examination of both the linear case with Gaussian elimination and the univariate case with finding the greatest common divisor finds that finding a Groebner basis is what we try to do.

Groebner bases were first defined by Buchberger his in 1965 thesis [3] in which he also gave an algorithm, now called Buchberger's Algorithm, to compute a Groebner basis given a set of polynomials

$$\mathscr{F} = (f_1(x_1, x_2, \ldots, x_n), f_2(x_1, x_2, \ldots, x_n), \cdots, f_m(x_1, x_2, \ldots, x_n)).$$

The core idea behind Buchberger's Algorithm is the S-polynomial:

Definition 5. *Let $f_1, f_2 \in \mathbb{F}[x_1, x_2, \ldots, x_n]$. We define the S-polynomial for the pair $\{f_1, f_2\}$ as*

$$S(f_1, f_2) = \frac{t}{LT(f_1)} f_1 - \frac{t}{LT(f_2)} f_2$$

where $t = lcm(LM(f_1), LM(f_2))$.

Notice that the leading monomial of $S(f, g)$ will be different than both that of f_1 and f_2. Buchberger proved a condition for a basis to be a Groebner basis based on the set of all S-polynomials of the generators.

Theorem 1. *(Buchberher's Criterion) [7] Let $F = \{f_1, f_2, \ldots, f_m\}$ be a basis for an ideal $I \subset \mathbb{F}[x_1, x_2, \ldots, x_n]$. Then F is a Groebner basis if and only if for each $i \neq j$ the normal form of $S(f_i, f_j)$ modulo F is zero.*

Here, the normal form of a polynomial is an extension of the division algorithm given by Algorithm 1.

Buchberger's strategy is to select one of the S-polynomials $S(f_1, f_2)$ of the generators $\{f_1, f_2, \ldots, f_m\}$, create its normal form r, and if that normal form is nonzero it is added to the basis. This process will eventually terminate due to the Hilbert Basis theorem. See Algorithm 2 for the pseudo-code. For many years, improvements to the Buchberger algorithm were focused on the finding a strategy for selecting the S-polynomial such that it does not reduce to zero (which is worthless computation) and leads to a Groebner basis as soon as possible [21]. An improvement to which selects subset of S-polynomials to reduce simultaneously is the F4 algorithm to which we now turn.

Algorithm 1 NormalForm(g,F)

```
Input: A polynomial $g \in \mathbb{F}[x_1, x_2, \ldots, x_n]$, a finite subset $F \subset \mathbb{F}[x_1, x_2, \ldots, x_n]$
Output: The normal form of $g$ modulo $F$
h := g
while there exists $f \in F$ such that $LM(f) \mid LM(h)$ do
    choose $f \in F$ such that $LM(f) \mid LM(h)$
    $h := h - \frac{LT(h)}{LT(f)} f$
end while
return $h$
```

Algorithm 2 Buchberger(F)

```
Input: A finite subset $F \subset \mathbb{F}[x_1, x_2, \ldots, x_n]$
Output: A Groebner basis $G$ for the ideal generated by $F$
$G := F$
$B := \{\{g_1, g_2\} \mid g_1, g_2 \in G, g_1 \neq g_2\}$
while $B \neq \emptyset$ do
    $B^* := \{g_1, g_2\} \subset B$
    $B := B \setminus B^*$
    $r :=$ NormalForm($S(g_1, g_2), G$)
    if $r \neq 0$ then
        $B := B \cup \{\{g, r\} \mid g \in G\}$
        $G := G \cup \{r\}$
    end if
end while
return $G$
```

2 Faugere's F4 Algorithm

2.1 F4 algorithm

F4 is a family of algorithms, originally proposed by Faugere in 1999 [16], which are an improvement of Buchberger algorithm which utilizes symbolic precomputation and sparse linear algebra techniques to more efficiently compute a Groebner basis. They are especially efficient using the reverse graded lexicographical ordering, which then can be turned into another ordering with another algorithm whose cost will be small in comparison to F4. The essential idea is to reduce several S-polynomials simultaneously by turning the reduction process into what amounts to row reduction of a large matrix and extracting non-zero rows of said matrix.

Here we will follow the thesis of Cabarcas [4] in describing roughly the basic version of F4. We note that this is slightly different than the notation used by Faugere originally, and Cabarcas more closely follows the form of Buchberger's algorithm. Our treatment will not be complete, and the reader should consult either the thesis or the original F4 paper for details. Suppose we want to construct a Groebner basis G for a set of polynomial $F = (f_1, \ldots, f_s)$ for some fixed monomial order >. We perform the following steps whose pseudo-code can be seen in Algorithm 3

Algorithm 3 BasicF4(F)

```
Input: A finite subset $F \subset \mathbb{F}[x_1, x_2, \ldots, x_n]$
Output: A Groebner basis $G$ for the ideal generated by $F$
$G := F$
$B := \{\{g_1, g_2\} \mid g_1, g_2 \in G, g_1 \neq g_2\}$
while $B \neq \emptyset$ do
    Select $B^* \subset B$, $B^* \neq \emptyset$
    $B := B \setminus B^*$
    $L := \left\{ \frac{\text{lcm}(\text{LM}(g_i),\text{LM}(g_j))}{\text{LT}(g_i)} \cdot g_i \middle| \{g_i, g_j\} \in B^* \right\}$
    $H :=$ BasicSymbPreProc$(L, G)$
    $\tilde{H} :=$ rref$(H)$
    $\tilde{H}^+ := \{h \in \tilde{H} \mid LM(h) \notin LM(H)\}$
    $G := G \cup \tilde{H}^+$
    $B := B \cup \{\{h, g\} \mid h \in \tilde{H}^+, g \in G, h \neq g\}$
end while
return $G$
```

Initialization First let us set $G = F$, and then construct the index set

$$B := \{\{g_i, g_j\} \mid 1 \leq i < j \leq s\}$$

which records all pair in the ideal for which we do not know if the S-polynomial will reduce to zero. We will edit this set until it is empty and then G will be a Groebner basis for the ideal of F.

Select Through some selection strategy, select a nonempty subset $B^* \subset B$ for which we will simultaneously reduce the S-polynomials. Remove these from B by redefining $B := B \setminus B'$. An example of a selection strategy would be the *normal strategy* for F4 which would be to select all the critical pairs with a minimal degree where the degree of a pair (f, g) is defined to be

$$\deg(\text{lcm}(\text{LM}(f), \text{LM}(g))).$$

This was originally found to be the best strategy [16]. If the selection strategy is to only select one pair, we reduce a single S-polynomial and thus recover the Buchberger algorithm.

Symbolic Precomputation We wish to consider the set H of all polynomials which might help us reduce the S-polynomials in B^*. First, we define the set

$$L := \left\{ \frac{\text{lcm}(\text{LM}(g_i), \text{LM}(g_j))}{\text{LT}(g_i)} \cdot g_i \middle| \{g_i, g_j\} \in B^* \right\}$$

which we notice is effectively the "halves" of the S-polynomials. Then we perform BasicSymbPreProc(L,G) in which we selected polynomials still in the ideal of G (and thus in the ideal of F) which will help reduce H.

Reduce H as a matrix We then view H as the Macaulay matrix whose rows represent the polynomials in H and columns the monomials in those polynomials (in

Algorithm 4 BasicSymbPreProc(L, G)

```
Input: Finite subsets $L, G \subset \mathbb{F}[x_1, x_2, \dots, x_n]$
$H := L$
$done := LM(H)$
while $M(H) \neq done$ do
    Select $t$ in $M(H) \setminus done$
    $done := done \cup \{t\}$
    if there exist $g \in G$ s.t. $LM(g) \mid t$ then
        choose $g \in G$ s.t $LM(g) \mid t$
        $H := H \cup \left\{\frac{t}{LM(g)} g\right\}$
    end if
end while
return $H$
```

descending order with respect to >). We perform linear algebra techniques to create the row echelon form $\tilde{H}$ of H.
Update B Finally for this iteration of the loop we append to G each $h \in \tilde{H}$ which has a new leading monomial. we then selected another subset of B to repeat the process.

That this does terminate and produce a Groebner basis can be found in the original F4 paper, and there are also improvements to this basic form of F4 as well such as reducing the basis at each step, discarding pairs which are known to reduce to zero automatically, and choosing the element to add in the reprocessing computation in a clever way [16].

3 Faugere's F5 algorithm

3.1 The F5 algorithm

Inspired by the idea from the paper by Moller et al. which states the relationship between the zero reduction and syzygies [23], Faugere proposed the F5 algorithm to avoid the redundant computations when computing a Grobner basis. In particular, the idea behind the F5 algorithm is to detect the useless reduction even before they occur. According to Faugere's paper [17], F_5 has been claimed to be extremely successful in Groebner basis computation. In 2002 Faugere claimed that the previously intractable "cyclic 10 module p" problem was successfully solved by an improved version of F5 algorithm in 16 hours on a 1GHz PIII. However, the implementation of the F5 algorithm that was used in this computation has never been published. Later in August 2004, the same problem was solved by a new optimized implementation of the F4 algorithm (released in May 2004) in 17.6 days on a 750MHz Sunfire v880. Moreover, as an application to cryptography, the HFE system of 80 quadratic equations in 80 variables over the binary field was first solved by Faugere's F5/2 algorithm in May 2002 in 52.2 hours on a 1GHz Alpha EV68. As a comparison, the new optimize F4 in Magma V2.11 solves the system in 22.1 hours on a 750MHz Sunfire v880 (using 14.4GB of memory)[27]. Thus, the F5 algorithm is not always faster than the

F4 algorithm. Faugere stated that the F5 algorithm is expected to reach its maximal efficiency in underdetermined systems.

There are many theoretical deficiencies in Faugere's original F5 paper. First, the proof of the termination was left unclear. Steger partially filled the details [28], but his proof is still under some hypothesis. Later, S. Pan et al. proved the termination of the F5B algorithm, an equivalent version of F5 in Buchberger's style [24]. The termination of the original F5 was proved by V. Galkin [19]. Second, Faugere only partially proved the correctness of the F5 algorithm, and the proof was based on extra assumption. Third, there are some errors in the pseudocode for an implementation the F5 algorithm.

The efficiency of F_5 mainly comes from the new criteria in the algorithm, the F_5 criterion and the rewritten criterion. These two powerful criteria can detect and thus reject a lot of useless critical pairs. Faugere's idea is to associate each polynomial a signature. By analyzing of the signature, we can track fundamental structural information of polynomials. So F_5 is the ancestor of many signature-based algorithm to compute Groebner basis.

In this section, we will describe the nature of the F5 algorithm, then state the criteria (the syzygy criterion and the rewritten criterion). We will provide two examples to show how these two criteria work and how they are related to the zero rows in the F4 algorithm. More detail about the F5 algorithm and proof of these two criteria can be found here [17]. We will borrow the definitions and terminology from [29].

3.2 Polynomials and signatures

Let $R = k[x_1, \cdots, x_n]$, consider the vector space R^s. The standard basis of R^s are denoted by $\mathbf{e}_1, \cdots, \mathbf{e}_s$. The monomial ordering can be extended to module R^s in the following fashion.

Definition 6. *We define the module term ordering $\prec$ on R^s as follows: $x^\alpha \mathbf{e}_i \prec x^\beta \mathbf{e}_j$ if and only if*
1. $i > j$ or
2. $i = j$ and $x^\alpha < x^\beta$. □

Remark 1. Let $<$ be the lex order and let $\prec$ be the corresponding module term ordering on R^s. We can define the leading term, leading monomial, and leading coefficient of a vector $\mathbf{g} \in R^s$ in a natural way. For example: Let $\mathbf{g} = (3x^2 + y)\mathbf{e}_1 + y^3\mathbf{e}_2$. The leading term of $\mathbf{g}$ is $\mathrm{LT}(\mathbf{g}) = 3x^2\mathbf{e}_1$, the leading monomial of $\mathbf{g}$ is $\mathrm{LM}(\mathbf{g}) = x^2\mathbf{e}_1$ and the leading coefficient of $\mathbf{g}$ is $\mathrm{LC}(\mathbf{g}) = 3$.

Let F be a set of polynomials $\{f_1, \cdots, f_s\} \subset R$, and $I =< f_1, \cdots, f_s >$, then for any polynomial $f \in I$, f has the form $f = \sum_{i=1}^{s} a_i f_i$. We define a homomorphism $\phi_F : R^s \to I$ in the way that

$$\phi_F(a_1, \cdots, a_s) = \sum_{i=1}^{s} a_i f_i.$$

Definition 7. *Let* $\mathbf{g} = (g_1, \cdots, g_s) \in R^s$*. If* $g = \phi_F(g)$*, the signature of* g*, denoted* sig(g)*, is* LM$(\mathbf{g})$ *the leading monomial of* $\mathbf{g}$. □

The relationship between a polynomial and its signature is fixed by the labeled polynomial.

Definition 8. *Let* $f \in I = \langle f_1, \cdots, f_s\rangle$*, and suppose that* f *has a signature* sig$(f) = x^\alpha \mathbf{e}_i$*, then we define a labeled polynomial* $r \in R^s \times R$ *of* f *to be* $r_f = (x^\alpha \mathbf{e}_i, f)$*. We call* sig$(r_f) = x^\alpha \mathbf{e}_i$ *the signature part of* r_f*, and* poly$(r_f) = f$ *the polynomial part of* r_f*.*

Remark 2. We further define the leading term, leading monomial, and leading coefficient of r_f: $\mathrm{LT}(r_f) = \mathrm{LT}(f)$, $\mathrm{LM}(r_f) = \mathrm{LM}(f)$, and $\mathrm{LC}(r_f) = \mathrm{LC}(f)$.

Remark 3. Let $r_f = (x^\alpha \mathbf{e}_i, f)$ and $r_g = (x^\beta \mathbf{e}_j, g)$ be two labeled polynomials. Let cx^γ be a term. The operations on labeled polynomial can be defined as follows:
1. $cx^\gamma r_f = (x^{\alpha+\gamma}\mathbf{e}_i, cx^\gamma f)$.
2. $r_f + r_g = (\max\{x^\alpha \mathbf{e}_i, x^\beta \mathbf{e}_j\}, f + g)$

Definition 9. *Let* r_f *and* r_g *be two labeled polynomials. Suppose* u, v *are terms in* R *such that* $u\mathrm{LT}(r_f) = v\mathrm{LT}(r_g) = \mathrm{lcm}(\mathrm{LM}(r_f), \mathrm{LM}(r_g))$*, we call* $[r_f, r_g] = (u, r_f, v, r_g)$ *a critical pair of* r_f *and* r_g*. The s-polynomial of a critical pair* $[r_f, r_g] = (u, r_f, v, r_g)$ *is defined as* $ur_f - vr_g$*.*

3.3 The F5 criterion

The F_5 criterion, also known as the syzygy criterion, detects useless computation arisen from the syzygies.

Definition 10. *A syzygy with respect to the polynomials* $\{f_1, \cdots, f_s\}$ *is an element* $\mathbf{a} = (a_1, \cdots, a_s) \in R^s$ *such that*

$$\sum_{i=1}^{s} a_i f_i = 0$$

□

Definition 11. *Let* $r_f = (x^\alpha \mathbf{e}_i, f)$ *be a labeled polynomial,* cx^γ *be a nonzero term in* R*. Let* B *be a set of label polynomials. The labeled polynomial* $cx^\gamma r_f = (x^{\alpha+\gamma}\mathbf{e}_i, cx^\gamma f)$ *is said to be NOT normalized by* B *if there exists a labeled polynomial* $r_g = (x^\beta \mathbf{e}_j, g) \in B$ *such that:*
1. $\mathrm{LM}(g) | x^{\alpha+\gamma}$ *and*
2. $i < j$ □

We are now ready to state the $F5$ criterion.

Definition 12. *(F5 criterion) Let* $[r_f, r_g] = (u, r_f, v, r_g)$ *be a critical pair and* B *be a set of labeled polynomials. If either* ur_f *or* vr_g *is not normalized by* B*, then the critical pair* $[r_f, r_g]$ *meets the F5 criterion.*

3.4 Example 1

Consider $f_1 = x_2^2 + x_1, f_2 = x_2x_3 + x_2$ in $\mathbb{F}_2[x_1, x_2, x_3]$ with graded lex order. The s-polynomial of f_1, f_2 is $S(f_1, f_2) = x_3f_1 + x_2f_2 = x_1x_3 + x_2^2$. Set $f_3 = S(f_1, f_2) = x_1x_3 + x_2^2$. Now let us consider $f_4 = S(f_2, f_3) = x_1f_2 + x_2f_3 = x_1x_2 + x_2^3 = x_2f_1$. So $f_4 = S(f_2, f_3)$ is a useless computation. In the F4 algorithm, the s-polynomial $S(f_2, f_3)$ will produce a zero row reduction.

$$M_1 = \left(\begin{array}{cccc} & x_1x_2x_3 & x_2^3 & x_1x_2 \\ x_2f_1 & 0 & 1 & 1 \\ x_1f_2 & 1 & 0 & 1 \\ x_2f_3 & 1 & 1 & 0 \end{array}\right)$$

Apply row reduction on M_1, we have that

$$M_1' = \left(\begin{array}{cccc} & x_1x_2x_3 & x_2^3 & x_1x_2 \\ x_2f_1 & 1 & 0 & 1 \\ x_1f_2 & 0 & 1 & 1 \\ x_2f_3 & 0 & 0 & 0 \end{array}\right)$$

Now let us check if this example satisfies any of these two criteria. Let $r_{f_1} = (\mathbf{e}_1, f_1)$ and $r_{f_2} = (\mathbf{e}_2, f_2)$ be the labeled polynomials of f_1, f_2 respectively. The s-polynomial of r_{f_1}, r_{f_2} is $r_{f_3} = S(r_{f_1}, r_{f_2}) = (x_3\mathbf{e}_1, x_1x_3 + x_2^2)$. Consider the critical pair of $[r_{f_2}, r_{f_3}] = (x_1, r_{f_2}, x_2, r_{f_3})$. $x_2r_{f_3} = (x_2x_3\mathbf{e}_1, x_1x_2x_3 + x_2^3)$ is not normalized by $\{r_{f_2}\}$. Hence, the critical pair $[r_{f_2}, r_{f_3}]$ meets the F5 criteria.

3.5 The rewritten criterion

The rewritten criterion aims to detect duplicated reductions of polynomials.

Definition 13. *Let r_f be a labeled polynomial, and cx^γ be a nonzero term in R. Let B be a set of labeled polynomials. The labeled polynomial $cx^\gamma r_f = (x^{\alpha+\gamma}, cx^\gamma f)$ is said to be rewritable by B if there exists a labeled polynomial $r_g = (x^\beta\mathbf{e}_i, g) \in B$ such that:*
1. $x^\beta\mathbf{e}_i | x^{\alpha+\gamma}\mathbf{e}_i$ and
2. r_g is generated later than r_f. □

The rewritten criterion is defined as follows.

Definition 14. *Let $[r_f, r_g] = (u, r_f, v, r_g)$ be a critical pair and B be a set of labeled polynomials. If either ur_f or vr_g is rewritable by B, then the critical pair $[r_f, r_g]$ meets the rewritten criterion.*

3.6 Example 2

Consider the polynomials $f_1 = x_1^2 + x_2^2, f_2 = x_1^2 + x_3 \in \mathbb{F}_2[x_1, x_2, x_3]$ with graded lex order. The s-polynomial of f_1, f_2 is $S(f_1, f_2) = f_1 + f_2 = x_2^2 + x_3$. Set $f_3 = S(f_1, f_2) = x_2^2 + x_3$. If we consider the s-polynomial of f_2, f_3, we have that $S(f_2, f_3) = x_2^2f_2 + x_1^2f_3 = x_2^2x_3 + x_1^2x_3 = x_3f_1$. So reduction on the s-polynomial of f_2 and f_3 will be a duplicated work if $x_3 f_1$ has been reduced. In the F4 algorithm, reductions on both $x_3 f_1$ and the s-polynomial of f_2 and f_3 will produce zero row reduction.

$$M_2 = \left(\begin{array}{cccc} & x_1^2x_2^2 & x_1^2x_3 & x_2^2x_3 \\ x_3f_1 & 0 & 1 & 1 \\ x_2^2f_2 & 1 & 0 & 1 \\ x_1^2f_3 & 1 & 1 & 0 \end{array}\right)$$

Apply row reduction on M_2, we have that

$$M_2' = \left(\begin{array}{cccc} & x_1^2x_2^2 & x_1^2x_3 & x_2^2x_3 \\ x_3f_1 & 1 & 0 & 1 \\ x_2^2f_2 & 0 & 1 & 1 \\ x_1^2f_3 & 0 & 0 & 0 \end{array}\right)$$

Now let $r_{f_1} = (\mathbf{e}_1, f_1)$ and $r_{f_2} = (\mathbf{e}_2, f_2)$ be the labeled polynomials corresponding to f_1, f_2 respectively. The s-polynomial r_{f_3} of r_{f_1}, r_{f_2} is $r_{f_3} = (\mathbf{e}_1, x_2^2 + x_3)$. Consider the critical pair $[r_{f_1}, r_{f_3}] = (x_2^2, r_{f_1}, x_1^2, r_{f_3})$, $x_1^2 r_{f_1}$ is rewritable by $\{r_{f_3}\}$ since $\mathbf{e}_1$ divides $x_2^2\mathbf{e}_1$ and r_{f_3} is generated later than r_{f_1}.

4 Complexity of F4 and F5

4.1 Degree of Regularity

For the remainder of this paper in judging the efficiency of F4 versus that of F5, let us assume that we are working over a finite field $\mathbb{F}_q$ of size q, which is a power of a prime number. $F4$ works over different fields than these, but the finite fields are what is largely used in cryptography which is our focus here. The complexity of F4 will be determined the largest size of the matrix H involved and the linear algebra cost in working with that matrix. The size of the matrix will be determined by what is called the degree of regularity which is the degree at which the first non-trivial relation from the original polynomials $f_1, \cdots, f_m$ occurs. The trivial relations are $f_i^h f_j^h - f_j^h f_i^h = 0$ and $f_i^q - f_i = 0$. All others are nontrivial. We will denote this by D_{reg}. As F4 will have to deal with polynomial of degree D_{reg} [9], the number of rows and columns of our matrix will be roughly the number of monomials of degree equal to or less then D_{reg} which is $\binom{n+D_{reg}}{D_{reg}}$. We note that the degree of regularity for F4 and F5 are the same D_{reg} [30], the matrix that F5 and F5 inspired algorithms will be working with is essentially the same size as F4 but having fewer rows due to the use of signatures. Thus, we may measure the gain in efficient that F5 can bring by seeing how many rows reduce to zero in F4.

The complexity for F4/F5 will be approximately $\binom{n+D_{reg}}{D_{reg}}^{\omega}$ where $2 \le \omega \le 3$ is the complexity exponent of matrix multiplication. ω is likely to be about $\log_2(7) \approx 2.8$ though may be as low as 2.3727 [30]. We note that there has been work on improving the linear algebra cost involved in Groebner basis calculations due to the special shape of the matrices involved such as the GBLA library [2].

4.2 Random Quadratic Systems

Most of multivariate cryptography, which is a serious contender for the new post quantum cryptosystems [5] required given Shor's algorithm [26] and recent work on quantum computing, is based on systems of m polynomial equations in n variables over a finite field $\mathbb{F}_q$. Though higher degree systems exists like the work of Ding and Yang for hashing [12], the best current schemes like the NIST post-quantum competition finalist Rainbow [11] rely on quadratic systems as they are the most efficient [10]. The finding of pre-images for the systems can provide a method of creating an encryption scheme or a signature scheme depending on the parameters. Most schemes are designed so that their public key is seemingly a random system. Thus, solving a random quadratic multivariate polynomial system lies in the heart of cryptanalysis of multivariate public key cryptography. Though some MPKC system contains linear terms and constant terms in their public keys, it is the quadratic part that guarantees the security. Yeh *et al.* [30] give D_{reg}, and thus the main parameter in determining complexity of F4/F5, for random quadratic systems over small fields ($D_{reg} \leq q$) as

$$D_{reg} = \min\left\{d : [t^d]\frac{(1-t^q)^n(1-t^2)^m}{(1-t)^n(1-t^{2q})^m} < 0\right\}$$

and for larger fields ($D_{reg} < q$)

$$D_{reg} = \min\left\{d : [t^d](1-t)^{m-n}(1+t)^m < 0\right\}.$$

where the notation $[t^d]g(t)$ means the coefficient of t^d in the series expansion of $g(t)$.

4.3 The HFE Encryption Scheme

For certain polynomial systems with specific (even if hidden) structure like HFE and its variants, the D_{reg} may be much smaller than the D_{reg} for a random system, one such example is the HFE cryptosystem and its variants like HFEv- [8, 9, 13, 14, 22]. The HFE encryption scheme was proposed by Pataran [25]. Let k be a finite field of q elements, and K be a degree n extension of k. The k-vector isomorphism $\phi : k^n \to K$ allows we to go back and forth between the fields k^n and its extension K. It follows that the Frobenius isomorphism $X \mapsto X^{q^i}$ is a linear map. Patarin's idea is to use another Frobenius isomorphism $X \mapsto X^{q^j}$ so that the central map remains quadratic in the form of $X \mapsto \sum_{i,j} X^{q^i+q^j}$. The central map $\mathscr{F} : K \to K$ of the HFE encryption scheme is a univariate polynomial in the form of

$$\mathscr{F}(X) = \sum_{i,j}^{q^i+q^j \leq D} \alpha_{i,j} X^{q^i+q^j} + \sum_{i}^{q^i \leq D} \beta_i X^{q^i} + \gamma$$

in which the coefficients $\alpha_{i,j}$, β_i and γ are randomly chosen from the extension field K and D is the upper bound of the degree of $\mathscr{F}$. The public key of the HFE encryption scheme is the composition:

$$\mathscr{P} = \mathscr{S} \circ \phi^{-1} \circ \mathscr{F} \circ \phi \circ \mathscr{T}.$$

in which $\mathscr{S}, \mathscr{T} : k^n \to k^n$ are two invertible affine maps.
In 2002, Faugere experimentally showed the degree of regularity of HFE is much lower than expected for a random system. For some instances of HFE, the systems can even be solved polynomialy. Faugere proved that there is a relatively small upper bound on the degrees of polynomials that occur in the Grobner basis computation [18]. However, no explicit formula to estimate the degree of regularity of HFE or its variant is provided in [18]. Later, Dubios and Gama proposed a mathematical method to compute the degree of regularity on general finite field [15]. A closed formula for HFE- is finally provided by Ding and Kleinjung [9]. They claimed that the degree of regularity of the polynomial system derived from an HFE- system is less than or equal to:

$$\begin{cases} \frac{(q-1)(\lfloor Log_q(D-1)\rfloor+a)}{2} + 2, \text{ if } q \text{ is even and } r+a \text{ is odd,} \\ \frac{(q-1)(\lfloor Log_q(D-1)\rfloor+a+1)}{2} + 2, \text{ otherwise} \end{cases}$$

in which q is the size of base field, and D is the degree of the HFE secret polynomial, $r = \lfloor Log_q(D-1)\rfloor + 1$, and a is the number of removed equations. Moreover, in [13], Ding and Yang further showed that the degree of regularity of HFEv- is bounded by:

$$\begin{cases} \frac{(q-1)(r+v+a-1)}{2} + 2, \text{ if } q \text{ is even and } r+a \text{ is odd,} \\ \frac{(q-1)(r+v+a)}{2} + 2, \text{ otherwise} \end{cases}$$

in which v is the number of vinegar variables in the HFEv- system. If $a = 0$, the formula gives the upper bound of the degree of regularity of HFEv:

$$\begin{cases} \frac{(q-1)(r+v-1)}{2} + 2, \text{ if } q \text{ is even and } r \text{ is odd,} \\ \frac{(q-1)(r+v)}{2} + 2, \text{ otherwise} \end{cases}$$

5 Experimental Results and Analysis

Here we record our experimental results on how many rows reduce to zero in the largest matrix in the F4 algorithm. Note that this is not necessarily the matrix with the largest percentage of zero rows. We performed both the basic and improved version on both a determined ($n = m$) system of random homogeneous quadratic polynomials for the finite field $\mathbb{F}_2$. As the purpose in cryptography would be to find a solution in the base field, we also experiment with the same system appended with the field equations

$$x_1^q = x_1,\ x_2^q = x_2,\ \ldots,\ x_n^q = x_n$$

which also increases the efficiency of finding a Groebner basis. We also experimented with HFE public keys with $D = n$ with field equations attached. We record the number of zero rows in the maximal sized matrix in F4, not necessarily the one with the largest proportion of zero rows.

Examining the three tables finds that the highest proportion of zero rows is approximately 0.617 with many of the matrices being slightly higher than half of zero rows. Sometimes, we are lucky and almost all of the rows are non-zero implying that almost all of them added useful information. This this pattern holds for larger values of n, then the F5 algorithm should not be much better than twice as fast as $F4$. As often the strength of cryptosystems are measured in the hundreds of bits, having no zero row reductions would be a modest increase in efficiency.

n	Max (Rows × Columns)	Zero Rows	Proportion of Zero Rows
2	9×6	3	$1/3 \approx 0.333$
3	24×16	9	$3/8 = 0.375$
4	42×31	11	$11/42 \approx 0.262$
5	74×58	18	$9/37 \approx 0.243$
6	204×113	91	$91/204 \approx 0.446$
7	515×198	318	$318/515 \approx 0.617$
8	938×381	559	$559/938 \approx 0.596$
9	986×536	450	$225/493 \approx 0.456$
10	1477×1446	31	$31/1477 \approx 0.021$
11	2513×2469	44	$44/2513 \approx 0.018$
12	3535×3487	48	$48/3535 \approx 0.014$
13	6119×5201	920	$920/6119 \approx 0.150$

Table 1: Data for Homogeneous System with Field Equations

n	Max (Rows × Columns)	Zero Rows	Proportion of Zero Rows
2	4×3	1	$1/4 = 0.25$
3	12×6	6	$1/2 = 0.5$
4	33×22	13	$13/33 \approx 0.394$
5	118×61	57	$57/118 \approx 0.483$
6	285×166	119	$119/285 \approx 0.418$
7	724×359	366	$183/362 \approx 0.506$
8	2111×934	1179	$1179/2111 \approx 0.559$
9	5257×2341	2920	$2920/5257 \approx 0.555$

Table 2: Data for Homogeneous System without Field Equations

n	Max (Rows × Columns)	Zero Rows	Proportion of Zero Rows
5	70×53	18	$9/35 \approx 0.257$
6	233×105	129	$129/233 \approx 0.554$
7	537×210	329	$329/537 \approx 0.613$
8	804×349	456	$38/67 \approx 0.567$
9	1356×537	821	$821/1356 \approx 0.605$
10	1727×779	948	$948/1727 \approx 0.549$
11	2384×1109	1275	$1275/2384 \approx 0.535$
12	3240×1518	1722	$287/540 \approx 0.531$
13	4434×4315	119	$119/4434 \approx 0.027$
14	6492×6333	159	$53/2164 \approx 0.024$

Table 3: Data for HFE System with $D = n$ and Field Equations

Bibliography

[1] Elwyn R Berlekamp. Factoring polynomials over finite fields. *Bell System Technical Journal*, 46(8):1853–1859, 1967.

[2] Brice Boyer, Christian Eder, Jean-Charles Faugère, Sylvian Lachartre, and Fayssal Martani. Gbla: Gröbner basis linear algebra package. In *Proceedings of the ACM on International Symposium on Symbolic and Algebraic Computation*, pages 135–142, 2016.

[3] Bruno Buchberger. A theoretical basis for the reduction of polynomials to canonical forms. *ACM SIGSAM Bulletin*, 10(3):19–29, 1976.

[4] Daniel Cabarcas. An implementation of faugère's f4 algorithm for computing gröbner bases. 2010.

[5] Matthew Campagna, Lidong Chen, Özgür Dagdelen, Jintai Ding, Jennifer K Fernick, Nicolas Gisin, Donald Hayford, Thomas Jennewein, Norbert Lütkenhaus, Michele Mosca, et al. Quantum safe cryptography and security. *ETSI White Paper*, 8, 2015.

[6] Nicolas Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 392–407. Springer, 2000.

[7] David Cox, John Little, and Donal OShea. *Ideals, varieties, and algorithms: an introduction to computational algebraic geometry and commutative algebra*. Springer Science & Business Media, 2013.

[8] Jintai Ding and Timothy J Hodges. Inverting hfe systems is quasi-polynomial for all fields. In *Annual Cryptology Conference*, pages 724–742. Springer, 2011.

[9] Jintai Ding and Thorsten Kleinjung. Degree of regularity for hfe-. *IACR Cryptology ePrint Archive*, 2011:570, 2011.

[10] Jintai Ding and Albrecht Petzoldt. Current state of multivariate cryptography. *IEEE Security & Privacy*, 15(4):28–36, 2017.

[11] Jintai Ding and Dieter Schmidt. Rainbow, a new multivariable polynomial signature scheme. In *International Conference on Applied Cryptography and Network Security*, pages 164–175. Springer, 2005.

[12] Jintai Ding and Bo-Yin Yang. Multivariates polynomials for hashing. In *International Conference on Information Security and Cryptology*, pages 358–371. Springer, 2007.

[13] Jintai Ding and Bo-Yin Yang. Degree of regularity for hfev and hfev. In *International Workshop on Post-Quantum Cryptography*, pages 52–66. Springer, 2013.

[14] Vivien Dubois and Nicolas Gama. The degree of regularity of hfe systems. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 557–576. Springer, 2010.

[15] Vivien Dubois and Nicolas Gama. The degree of regularity of hfe systems. In *Advances in Cryptology - ASIACRYPT 2010 - 16th International Conference on the Theory and Application of Cryptology and Information Security*, volume 6477 of *Lecture Notes in Computer Science*, pages 557–576. Springer, 2010.

[16] Jean-Charles Faugere. A new efficient algorithm for computing gröbner bases (f4). *Journal of pure and applied algebra*, 139(1-3):61–88, 1999.

[17] Jean Charles Faugère. A new efficient algorithm for computing gröbner bases without reduction to zero (f 5). In *Proceedings of the 2002 international symposium on Symbolic and algebraic computation*, pages 75–83, 2002.

[18] Jean-Charles Faugere and Antoine Joux. Algebraic cryptanalysis of hidden field equation (hfe) cryptosystems using gröbner bases. In *Annual International Cryptology Conference*, pages 44–60. Springer, 2003.

[19] Vasily Galkin. Termination of original f5. *arXiv preprint arXiv:1203.2402*, 2012.

[20] Michael R Gary and David S Johnson. Computers and intractability: A guide to the theory of np-completeness, 1979.

[21] Alessandro Giovini, Teo Mora, Gianfranco Niesi, Lorenzo Robbiano, and Carlo Traverso. "one sugar cube, please" or selection strategies in the buchberger algorithm. In *Proceedings of the 1991 international symposium on Symbolic and algebraic computation*, pages 49–54, 1991.

[22] Xin Jiang, Jintai Ding, and Lei Hu. Kipnis-shamir attack on hfe revisited. In *International Conference on Information Security and Cryptology*, pages 399–411. Springer, 2007.

[23] H Michael Möller, Teo Mora, and Carlo Traverso. Gröbner bases computation using syzygies. In *Papers from the international symposium on Symbolic and algebraic computation*, pages 320–328, 1992.

[24] Senshan Pan, Yupu Hu, and Baocang Wang. The termination of the f5 algorithm revisited. In *Proceedings of the 38th International Symposium on Symbolic and Algebraic Computation*, pages 291–298, 2013.

[25] Jacques Patarin. Hidden fields equations (hfe) and isomorphisms of polynomials (ip): Two new families of asymmetric algorithms. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 33–48. Springer, 1996.

[26] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.

[27] Allan Steel. Allan steel's grobner basis timings page. *http://magma. maths. usyd. edu. au/users/allan/gb/*, 2004.

[28] Till Stegers and Johannes Buchmann. Faugere's f5 algorithm revisited. *IACR Cryptol. ePrint Arch.*, 2006:404, 2006.

[29] Yao Sun and Dingkang Wang. The f5 algorithm in buchberger's style. *Journal of Systems Science and Complexity*, 24(6):1218–1231, 2011.

[30] Jenny Yuan-Chun Yeh, Chen-Mou Cheng, and Bo-Yin Yang. Operating degrees for xl vs. f4/f5 for generic \mathcal{M}q with number of equations linear in that of variables. In *Number Theory and Cryptography*, pages 19–33. Springer, 2013.