MutantXL

Jintai Ding, Johannes Buchmann, Mohamed Saied Emam Mohamed,
Wael Said Abd Elmageed Mohamed and Ralf-Philipp Weinmann

Abstract. We show how the concept of mutants can be used to speed up the XL algorithm for solving systems of multivariate equations over finite fields significantly.

Keywords. Mutant, MutantXL algorithm, XL algorithm.

1. Introduction

The intractability of solving large systems of multivariate polynomial equations over finite fields is the security basis for many cryptosystems such as the Matsumoto-Imai scheme [8], HFE [11], C^*_{-+} and HM [12], and the schemes by T.Moh [9], and Ding [4, 5]. Therefore, algorithms for solving such systems are important tools for cryptanalysis. In recent years, several algorithms such as XL [2], F4 [6], and F5[7] have been proposed that outperform the standard Buchberger [1] algorithm in certain cases. In 2006, Ding [3] discovered the mutant concept, which characterizes the degeneration of a polynomial system, and he suggested a new strategy to use this new concept to improve various algorithms. In this paper, we study the concept of mutants and explore the potential of their application to improve polynomial solving algorithms further. We explain such an improvement in the case of the XL algorithm and we present examples that demonstrate the improvements.

The paper is organized as follows. In Section 2 we present the concept of a mutant. In Section 3 we explain MutantXL algorithm. Section 4 contains the experimental results.

2. Mutants

In this section we introduce mutants and explain their importance for solving systems of multivariate polynomial equations. Throughout the paper we let F be a finite field and we let q be its cardinality. We consider the ring

$$R = F[x_1, \ldots, x_n]/(x_1^q - x_1, ..., x_n^q - x_n)$$

of functions over R in in the n variables $x_1, \ldots, x_n$. Here $x_i^q - x_i = 0$, $1 \leq i \leq n$ are the so-called field equations. In R, each element is uniquely expressed as a polynomial where each x_i has degree less than q. The degree of this polynomial is called the degree of the corresponding function. For the sake of convenience, we will identify functions in R with their representing polynomials.

Let P be a finite set of polynomials in R. Many algorithms such as XL, F4, and F5 for solving the system

$$p(\vec{x}) = 0, \quad p \in P, \tag{2.1}$$

where

$$\vec{x} = (x_1, \ldots, x_n),$$

use the following strategy. They find additional polynomials of not so large degree in the ideal generated by the elements of P by multiplying them by monomials. They linearize the system (2.1) by replacing the monomials with new variables and apply Gaussian elimination.

In many experiments with those algorithms, we have observed that during Gaussian elimination certain polynomials of degree lower than what they normally should be appear, which could be used beneficially in the algorithms. If those polynomials are univariate, then we know how to use them, namely for substitution as described in [10]. But if they are not, they are just treated like any other polynomial in the algorithm. We call those polynomials *mutants* and show that they deserve special treatment. We will now give a mathematical definition of mutants.

Let g be polynomial in the ideal generated by the elements of P. Naturally, it can be written as

$$g = \sum_{p \in P} g_p p \tag{2.2}$$

where $g_p \in R$, $p \in P$. The *level* of this representation is defined to be $\max\{\deg g_p p | p \in P\}$. Note that this level depends on P. The *level* of the polynomial g is defined to be the minimum level of all of its representations. The polynomial g is called a *mutant* with respect to P if its degree is less than its level. Note here that we will never have mutants if the elements in P are homogenous.

Next, we explain the meaning of mutants. When a mutant is written as a linear combination (2.2), then one of the polynomials $g_p p$ has a degree exceeding the degree of the mutant. This means that a mutant of degree d cannot be found as a linear combination of polynomials of the form mp where m is a monomial, $p \in P$ and the degree of mp is at most d. However, such mutants could help in solving the system (2.1) if we can find them efficiently. In the next section we explain MutantXL, a modification of the XL algorithm that uses mutants. In Section 4 we will present examples that show that MutantXL could indeed superior over XL.

3. The MutantXL algorithm

We explain the MutantXL algorithm and how this algorithm is different from XL. We use the notation of the previous section. So P is a finite set of polynomials in R and we consider the system (2.1) of multivariate equations. To simplify our explanation we assume that the system (2.1) is quadratic and has a unique solution.

As XL the MutantXL algorithm uses *linearization* of polynomials. It is obtained as follows. Monomials are replaced by variables which are ordered according to the graded lexicographical ordering. With this linearization the multivariate system (2.1) becomes a linear system. Conversely, each linear system can be viewed as a multivariate system.

- *Initilization* Use Gaussian elimination to make P linearly independent. Set the set of *root polynomials* to P, the *degree bound* of those root polynomials to their degree, the *total degree bound* D to the minimum degree of the root polynomials, the *old system* to the empty system, and the new system to (2.1).
- *Gauss* Use linearization to transform the *new system* into row echelon form.
- *Solve* If there are univariate polynomials in the *new system*, then determine the values of the corresponding variables. If this solves the system return the solution and terminate. Otherwise, substitute the values for the variables in the *root polynomials* and in the *new system*, apply Gaussian elimination to make the set of root polynomials linearly independent, and go back to *Gauss*.
- *Enlarge system* No univariate polynomials have been found in the previous step. Add polynomials of degree $< L$ in the *new system* which are not in the *old system* to the set of *root polynomials* and set their degree bound to their degree. These polynomials are mutants with respect to the old system. Select the *root polynomials* with the least degree bound, set D to this degree bound plus 1. Multiply each of the new root polynomials p by all monomials of degree $D - \deg p$, include the resulting polynomials in the *new system*, and go back to *Gauss*.

The XL algorithm can be obtained from MutantXL if in the *Enlarge system* step the part in which mutants are found and added to the set of root polynomials is skipped. This implies that the linear systems in XL are subsystems of a linear systems in MutantXL. So MutantXL terminates since XL terminates.

We show that the polynomials found in the *Enlarge system* step are, in fact, mutants with respect to the current set of root polynomials. Let p be such a polynomial and let V be the vector space generated by the root polynomials over F. By construction, the root polynomials of degree $\leq \deg p$ form a basis of the subspace of V of all polynomials of degree $\leq \deg p$ in triangular form. They are not changed in the *Gauss* step. Therefore, p is not in this subspace. So if p is written as a linear combination of the root polynomials, the degree of one summand exceeds $\deg p$. But this shows that p is a mutant.

#Eq	#Var	Max D		Largest matrix Rank		#Mut
		MXL	XL	MXL	XL	
7	7	3	4	63 × 64 62	203 × 99 98	14
10	10	3	4	210 × 176 175	560 × 386 385	10
11	11	4	4	803 × 562 561	737 × 562 561	1
13	13	4	4	1457 × 1093 1092	1196 × 1093 1092	3
15	15	4	5	2340 × 1941 1760	8520 × 4944 4943	270
17	17	4	5	3349 × 3214 2737	14025 × 9402 9401	255
19	19	3	4	2565 × 1160 1159	3628 × 5036 3306	117
25	25	4	>4	14218×15276 13750	?×? ?	1919

Table 1. Performance of MutantXL versus XL

4. Experimental results

In this section, we present experimental results and compare the performance of MutantXL with the performance of XL. We use seven HFE examples from [13] and another HFE system (25 equations in 25 variables) from the Hotaru distribution [14]. The results can be found in Table 1. For each example we present the number of equations and variables of the initial system, the maximum degree bound D used in the algorithm, the size of the largest linear system to which Gauss is applied, and the number of mutants found in the algorithm. Table 1 clearly shows that in six cases MutantXL outperforms XL and in the other two cases the two algorithms do the same.

In Table 2 we also present details for the HFE-25 example. In this example we initially have $D = 2$. Whenever the system is enlarged, we say that MutantXL and XL enter a new round. For each round we present the total degree bound D, the number of mutants found, and the number of root polynomials. In each round of this example, Gauss is only called once and no substitution is made except for the last round in which the system is solved. For each round we present the total degree bound D, the size of the matrix before Gauss and its rank, the number of root polynomials used in this round, and the number of mutants found in this round.

Round	Degree Bound	Matrix Size	Rank	#Sub	#Roots	#Mut
1	2	25×326	25	0	25	0
2	3	650×2626	650	0	25	0
3	4	8150×15276	7825	0	25	50
4	4	9075×15276	9075	0	75	200
5	4	14075×15276	13719	0	275	1669
6	2	14218×15276	13750	25	1944	-

Table 2. Results for HFE-25

Round	#Degree 1	#Degree 2	#Degree 3	#Degree 4
1	0	15	0	0
2	0	0	625	0
3	0	0	50	7125
4	0	0	200	1050
5	20	249	1400	2975
6	5	26	0	0
Total	25	280	2275	11150

Table 3. Mutants in the HFE-25 example

In Table 3 we also present details for the mutants found in the new system after Gauss. The total number of mutants produced by MutantXL for each degree can be found in the last row.

From the experiments above, we can conclude that the MutantXL algorithm can indeed outperform the XL algorithm and can solve multivariate systems at a lower degree than the usual XL algorithm. Since the total degree bound that the XL algorithm needs to go up is typically the bottle neck of this algorithm, this is quite a considerable improvement.

In the future we will study how sparse linear algebra algorithms such as the Wiedemann algorithm can further improve MutantXL.

References

[1] B. Buchberger. *An algorithm for finding a basis for the residue class ring of a zero-dimensional polynomial ring*. Dissertation, University of Innsbruck, 1965.

[2] N. Courtois, A. Klimov, J. Patarin, and A. Shamir. Efficient Algorithms for Solving Overdefined Systems of Multivariate Polynomial Equations. In *Proceedings of International Conference on the Theory and Application of Cryptographic Techniques(EUROCRYPT)*, volume 1807 of *Lecture Notes in Computer Science*, pages 392–407, Bruges, Belgium, May 2000. Springer.

[3] J. Ding. Mutants and its impact on polynomial solving strategies and algorithms. Privately distributed research note, University of Cincinnati and Technical University of Darmstadt, 2006.

[4] J. Ding. A new variant of the matsumoto-imai cryptosystem through perturbation. In *7th International Workshop on Theory and Practice in Public Key Cryptography Public Key Cryptography - PKC*, volume 2947 of *Lecture Notes in Computer Science*, pages 305–318, Singapore, March 2004. Springer.

[5] J. Ding and J. Gower. Inoculating multivariate schemes against differential cryptanalysis. In *9th International Workshop on Theory and Practice in Public Key Cryptography Public Key Cryptography - PKC*, volume 3958 of *Lecture Notes in Computer Science*, pages 290–301. Springer, 2006.

[6] J.-C. Faugére. A new efficient algorithm for computing Gröbner bases (F4). *Pure and Applied Algebra*, 139(1-3):61–88, June 1999.

[7] J.-C. Faugére. A new efficient algorithm for computing Gro"bner bases without reduction to zero (F5). In *Proceedings of the 2002 international symposium on Symbolic and algebraic computation (ISSAC)*, pages 75 – 83, Lille, France, July 2002. ACM.

[8] T. Matsumoto and H. Imai. Public Quadratic Polynomial-Tuples for Efficient Signature-Verification and Message-Encryption. In *Workshop on the Theory and Application of of Cryptographic Techniques Advances in Cryptology- EUROCRYPT*, volume 330 of *Lecture Notes in Computer Science*, pages 419–453, Davos, Switzerland, May 1988. Springer.

[9] T. Moh. A Public Key System With Signature And Master Key Functions. In *Communications in Algebra*, pages 2207 – 2222, 1999.

[10] J. Patarin. Cryptanalysis of the Matsumoto and Imai Public Key Scheme of Eurocrypt'88. In *Proceeding of the International Cryptology Conference*, volume 963 of *Lecture Notes in Computer Science*, pages 248–261. Springer, 1995.

[11] J. Patarin. Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP): two new families of Asymmetric Algorithms. In *Proceeding of International Conference on the Theory and Application of Cryptographic Techniques Advances in Cryptology- Eurocrypt*, volume 1070 of *Lecture Notes in Computer Science*, pages 33–48, Saragossa, Spain, May 1996. Springer.

[12] J. Patarin, L. Goubin, and N. Courtois. C^*_{-+} and HM : Variations Around Two Schemes of T. Matsumoto and H. Imai. In *Proceeding of International Conference on the Theory and Application of Cryptology and Information Security Advances in Cryptology - ASIACRYPT*, volume 1514 of *Lecture Notes in Computer Science*, pages 35 – 50, Beijing, China, October 1998. Springer.

[13] A. Segers. Algebraic Attacks from a Gröbner Basis Perspective. Master's thesis, Department of Mathematics and Computing Science, TECHNISCHE UNIVERSITEIT EINDHOVEN, Eindhoven, October 2004.

[14] M. Shigeo. Hotaru. http://cvs.sourceforge.jp/cgi-bin/viewcvs.cgi/hotaru/hotaru/hfe25-96?view=markup.

Jintai Ding
Department of Mathematical Sciences
University of Cincinnati
Cincinnati OH 45220
USA
e-mail: jintai.ding@uc.edu

Johannes Buchmann
Fachbereich Informatik
Technische Universität Darmstadt
64289 Darmstadt
Germany
e-mail: buchmann@cdc.informatik.tu-darmstadt.de

Mohamed Saied Emam Mohamed
Fachbereich Informatik
Technische Universität Darmstadt
64289 Darmstadt
Germany
e-mail: mohamed@cdc.informatik.tu-darmstadt.de

Wael Said Abd Elmageed Mohamed
Fachbereich Informatik
Technische Universität Darmstadt
64289 Darmstadt
Germany
e-mail: wael@cdc.informatik.tu-darmstadt.de

Ralf-Philipp Weinmann
Fachbereich Informatik
Technische Universität Darmstadt
64289 Darmstadt
Germany
e-mail: weinmann@cdc.informatik.tu-darmstadt.de