Extended Algorithm for Solving Underdefined Multivariate Quadratic Equations

*Hiroyuki Miura[1] Yasufumi Hashimoto[2] Tsuyoshi Takagi[1]

Kyushu University[1]

University of the Ryukyus[2]

June 7, 2013

Contents

Background (1/2)

It is known that solving MQ-Problem is NP-hard [Garey et al., 1979].

MQ-Problem

Let $f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})$ be quadratic polynomials of $\boldsymbol{x} = {}^t(x_1, x_2, \ldots, x_n)$ over a finite field k.

$$\begin{cases} f_1(x_1, ..., x_n) = \sum_{1 \le i \le j \le n} a_{1,i,j} x_i x_j + \sum_{1 \le i \le n} b_{1,i} x_i + c_1 \\ f_2(x_1, ..., x_n) = \sum_{1 \le i \le j \le n} a_{2,i,j} x_i x_j + \sum_{1 \le i \le n} b_{2,i} x_i + c_2 \\ \vdots \\ f_m(x_1, ..., x_n) = \sum_{1 \le i \le j \le n} a_{m,i,j} x_i x_j + \sum_{1 \le i \le n} b_{m,i} x_i + c_m, \end{cases}$$

where $a_{l,i,j}, b_{l,i}, c_l \in k$ and $l = 1, ..., m$.
MQ-Problem is to find a tuple $(a_1, \ldots, a_n) \in k^n$ such that $f_i(a_1, ..., a_n) = 0$ for all $i = 1, \ldots, m$.

Background (2/2)

The security of Multivariate Public Key Cryptosystems (MPKCs) depends on the hardness of MQ-Problem.

Some examples of MPKCs :

- Matsumoto Imai Cryptosystem [Matsumoto et al., Eurocrypt'88]
- Hidden Field Equation [Patarin, Eurocrypt'96]
- Unbalanced Oil and Vinegar [Kipnis et al., Eurocrypt'99]
- Rainbow [Ding et al., ACNS'05]

Classification of MQ-Problem

Let n be the number of unknowns and m be the number of equations.
We call "Overdefined" when $n \ll m$, and "Underdefined" when $n \gg m$.

Previous works & Main Result

Table: Algorithms solving "Underdefined" MQ-Problem

	Characteristic	Applicable range	Complexity
Kipnis et al. [Eurocrypt'99]	2	$n \geq m(m+1)$	(Poly.)
	odd		(Exp.)
Courtois et al. [PKC'02]	2	$n \geq m(m+1)$	(Poly.)
	odd	$n \geq 2^{\frac{m}{7}} m(m+1)$	(Poly.)
		$n \geq 2^{\frac{m}{7}} (m+1)$	(Exp.)
Thomae et al. [PKC'12]	any	$n > m$	(Exp.)
Proposed	2	$n \geq m(m+3)/2$	(Poly.)
	odd		(Exp.)

(n : the number of unknowns, m : the number of equations)

Comparison (char k is 2)

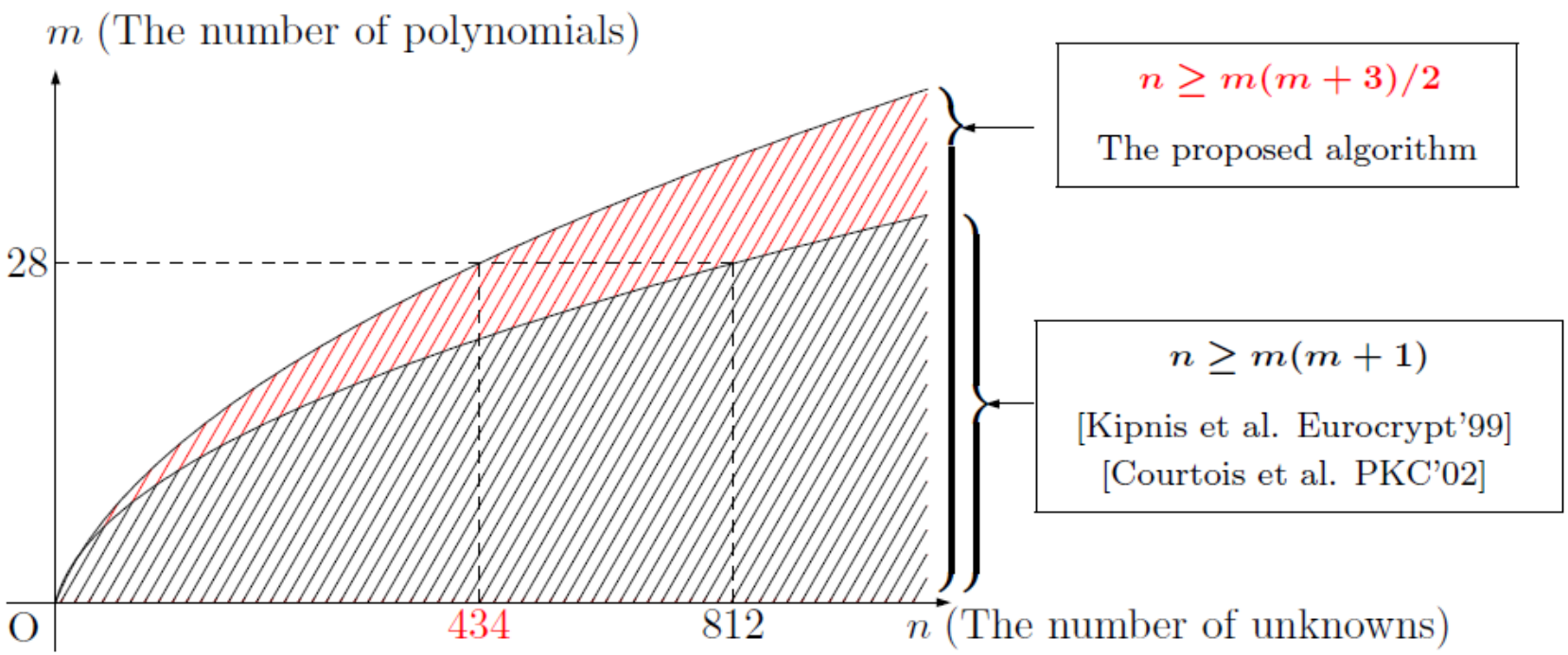


Figure: Applicable range of proposed algorithm and other known algorithms

Comparison (char k is odd)

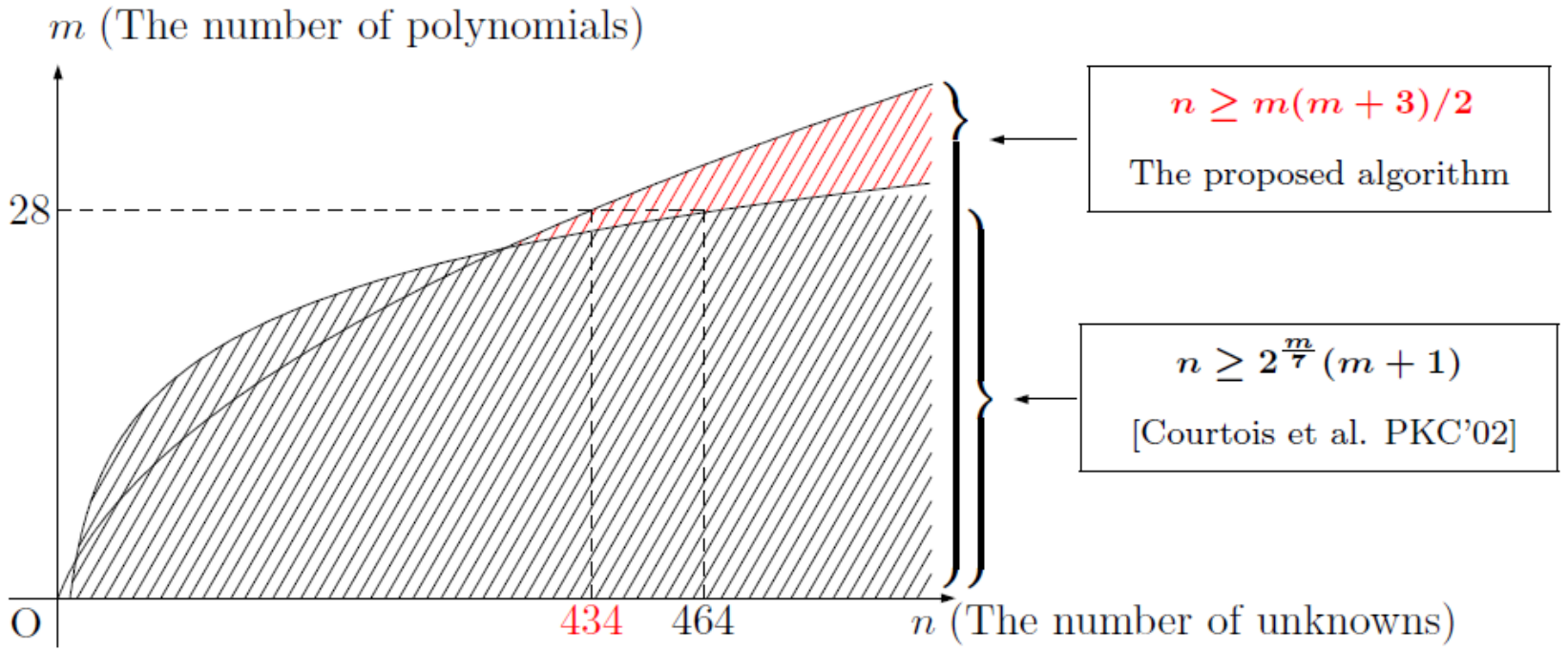


Figure: Applicable range of proposed algorithm and other known algorithms

Proposed Algorithm - Ideas(1/5)

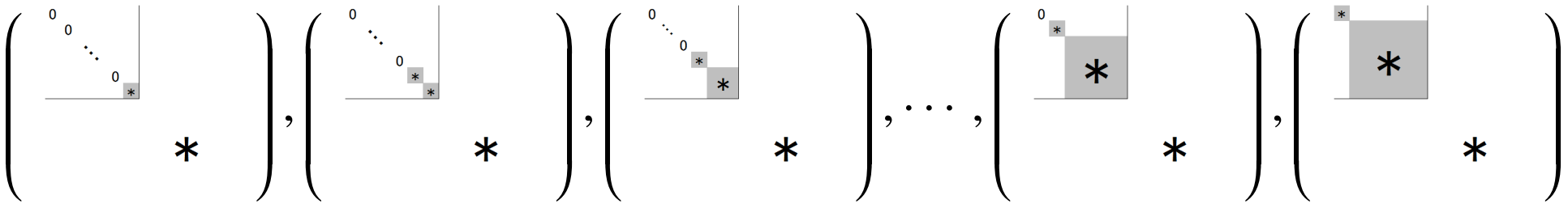
For $i = 1, \ldots, m$ the polynomials $f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})$ are denoted by

$$f_i(x_1, x_2, \ldots, x_n) = {}^t\boldsymbol{x} F_i \boldsymbol{x} + (\text{linear.})$$

where $F_1, \ldots, F_m$ are $n \times n$ matrices over k.

We can change the upper left $m \times m$ part of $F_1, F_2, \ldots, F_m$ to the following form using linear transformations of their coefficient matrices.

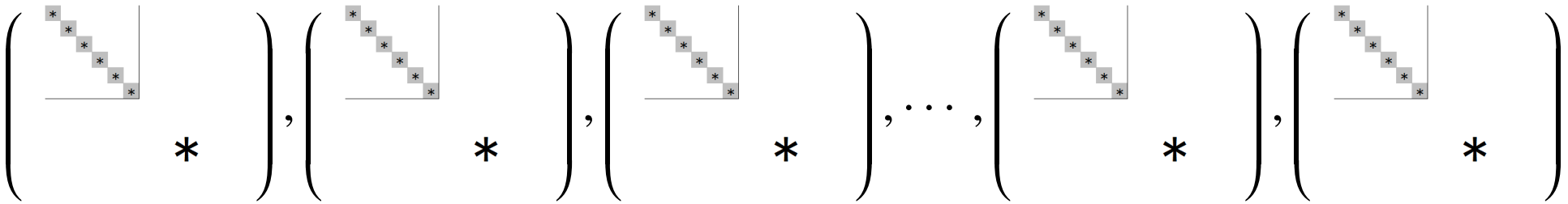
$F_1, F_2, F_3, \cdots, F_{m-1}, F_m \mapsto$



Proposed Algorithm - Ideas(2/5)

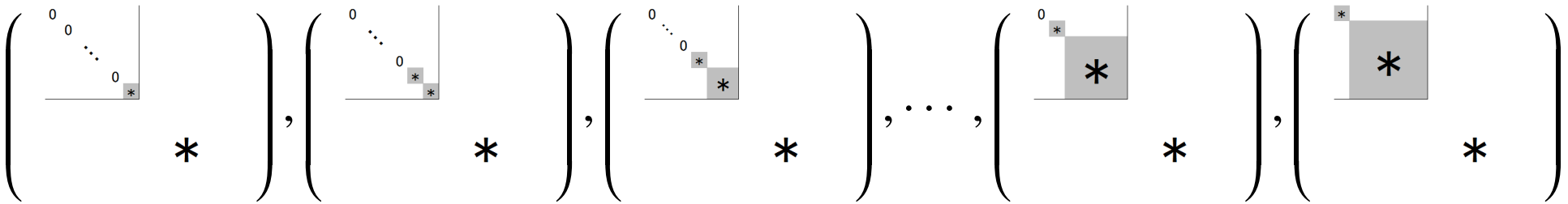
Algorithm by Kipnis et al. [Eurocrypt'99]

$F_1, F_2, F_3, \cdots, F_{m-1}, F_m \mapsto$



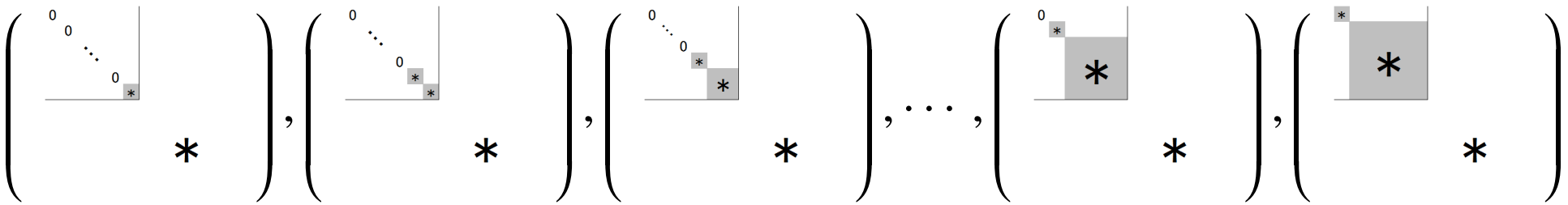
Proposed Algorithm

$F_1, F_2, F_3, \cdots, F_{m-1}, F_m \mapsto$



Proposed Algorithm - Ideas(3/5)

$$F_1, F_2, F_3, \cdots, F_{m-1}, F_m \mapsto$$



These matrices mean the following quadratic equations.

$$\begin{cases} x_m^2 & & + \sum_{1\le i\le m} x_i L_{1,i}(x_{m+1},\ldots,x_n) + Q_{1,2}(x_{m+1},\ldots,x_n) = 0 \\ x_{m-1}^2 & +Q_{2,1}(x_m) & + \sum_{1\le i\le m} x_i L_{2,i}(x_{m+1},\ldots,x_n) + Q_{2,2}(x_{m+1},\ldots,x_n) = 0 \\ x_{m-2}^2 & +Q_{3,1}(x_{m-1},x_m) & + \sum_{1\le i\le m} x_i L_{3,i}(x_{m+1},\ldots,x_n) + Q_{3,2}(x_{m+1},\ldots,x_n) = 0 \\ & & \vdots \\ x_2^2 & +Q_{m-1,1}(x_3,\ldots,x_m) & + \sum_{1\le i\le m} x_i L_{m-1,i}(x_{m+1},\ldots,x_n) + Q_{m-1,2}(x_{m+1},\ldots,x_n) = 0 \\ x_1^2 & +Q_{m,1}(x_2,\ldots\ldots,x_m) & + \sum_{1\le i\le m} x_i L_{m,i}(x_{m+1},\ldots,x_n) + Q_{m,2}(x_{m+1},\ldots,x_n) = 0. \end{cases}$$

where L's are linear polynomials and Q's are quadratic polynomials.

Proposed Algorithm - Ideas(4/5)

We obtain m^2 linear polynomials.

$$
\begin{array}{ccccccc}
L_{1,1} & L_{1,2} & \cdots & L_{1,m-2} & L_{1,m-1} & L_{1,m} \\
L_{2,1} & L_{2,2} & \cdots & L_{2,m-2} & L_{2,m-1} & L_{2,m} \\
L_{3,1} & L_{3,2} & \cdots & L_{3,m-2} & L_{3,m-1} & L_{3,m} \\
\vdots & \vdots & & \vdots & \vdots & \vdots \\
L_{m-1,1} & L_{m-1,2} & \cdots & L_{m-1,m-2} & L_{m-1,m-1} & L_{m-1,m} \\
L_{m,1} & L_{m,2} & \cdots & L_{m,m-2} & L_{m,m-1} & L_{m,m}
\end{array}
$$

Proposed Algorithm - Ideas(4/5)

We obtain m^2 linear polynomials.

$$
\begin{array}{ccccccc}
L_{1,1} & L_{1,2} & \cdots & L_{1,m-2} & L_{1,m-1} & L_{1,m} \\
L_{2,1} & L_{2,2} & \cdots & L_{2,m-2} & L_{2,m-1} & L_{2,m} \\
L_{3,1} & L_{3,2} & \cdots & L_{3,m-2} & L_{3,m-1} & L_{3,m} \\
\vdots & \vdots & & \vdots & \vdots & \vdots \\
L_{m-1,1} & L_{m-1,2} & \cdots & L_{m-1,m-2} & L_{m-1,m-1} & L_{m-1,m} \\
L_{m,1} & L_{m,2} & \cdots & L_{m,m-2} & L_{m,m-1} & L_{m,m}
\end{array}
$$

Solve linear equations of $x_{m+1}, \ldots, x_n$ so that the shaded part are zero, and substitute the solutions $x_{m+1}, \ldots, x_n$ into the previous equations.

Proposed Algorithm - Ideas(5/5)

Finally, we obtain the following quadratic equations.

$$\begin{cases} x_m^2 - \lambda_1 = 0 \\ x_{m-1}^2 + Q(x_m) - \lambda_2 = 0 \\ x_{m-2}^2 + Q(x_{m-1}, x_m) - \lambda_3 = 0 \\ \qquad\qquad \vdots \\ x_2^2 + Q(x_3, \ldots, x_m) - \lambda_{m-1} = 0 \\ x_1^2 + Q(x_2, \ldots, x_m) - \lambda_m = 0. \end{cases}$$

where $\lambda_1, \ldots, \lambda_m \in k$.

Quadratic equations of this form are easy to solve!

Proposed Algorithm - Toy Example(1/6)

Let $k = \mathrm{GF}(7), n = 5, m = 2$.

$$(*)\begin{cases} f_1(x_1,\ldots,x_5) = 4x_1^2 + 5x_1x_2 + 5x_1x_3 + 3x_1x_4 + 4x_1x_5 + x_2^2 + 4x_2x_3 \\ \qquad\qquad +x_2x_5 + x_3^2 + 4x_3x_4 + x_3x_5 + 4x_4^2 + 3x_4x_5 = 0 \\ f_2(x_1,\ldots,x_5) = 2x_1^2 + 4x_1x_2 + 6x_1x_3 + 3x_1x_4 + 6x_1x_5 + 3x_2^2 + 6x_2x_3 \\ \qquad\qquad +3x_2x_4 + 5x_2x_5 + x_3^2 + 2x_3x_4 + 2x_3x_5 + 4x_5^2 = 0 \end{cases}$$

Then, coefficient matrices is as follows.

$$F_1 = \begin{pmatrix} 4 & 5 & 5 & 3 & 4 \\ 0 & 1 & 4 & 0 & 1 \\ 0 & 0 & 1 & 4 & 1 \\ 0 & 0 & 0 & 4 & 3 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}, F_2 = \begin{pmatrix} 2 & 4 & 6 & 3 & 6 \\ 0 & 3 & 6 & 3 & 5 \\ 0 & 0 & 1 & 2 & 2 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 \end{pmatrix}.$$

Proposed Algorithm - Toy Example(2/6)

$$F_1 = \begin{pmatrix} 4 & 5 & 5 & 3 & 4 \\ 0 & 1 & 4 & 0 & 1 \\ 0 & 0 & 1 & 4 & 1 \\ 0 & 0 & 0 & 4 & 3 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}, F_2 = \begin{pmatrix} 2 & 4 & 6 & 3 & 6 \\ 0 & 3 & 6 & 3 & 5 \\ 0 & 0 & 1 & 2 & 2 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 4 \end{pmatrix}.$$

Step 1

We will make the (1,1)-element of F_1 zero.

$$F_1 \mapsto F_1 - 2F_2 = \begin{pmatrix} 0 & 4 & 0 & 4 & 6 \\ 0 & 2 & 6 & 1 & 5 \\ 0 & 0 & 6 & 0 & 4 \\ 0 & 0 & 0 & 4 & 3 \\ 0 & 0 & 0 & 0 & 6 \end{pmatrix}$$

Step 2 - (i)

We will make the coefficients of x_1x_2 in ${}^t\boldsymbol{x}F_1\boldsymbol{x}$ and ${}^t\boldsymbol{x}F_2\boldsymbol{x}$ zero by using linear transformation $\boldsymbol{x} \mapsto T_2\boldsymbol{x}$. Let

$$T_2 = \begin{pmatrix} 1 & a_{1,2} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & a_{3,2} & 1 & 0 & 0 \\ 0 & a_{4,2} & 0 & 1 & 0 \\ 0 & a_{5,2} & 0 & 0 & 1 \end{pmatrix}.$$

Then, the coefficients of x_1x_2 in ${}^t(T_2\boldsymbol{x})F_1(T_2\boldsymbol{x})$ and ${}^t(T_2\boldsymbol{x})F_2(T_2\boldsymbol{x})$ are

$$4a_{4,2} + 6a_{5,2} + 4 \quad \text{and} \quad 4a_{1,2} + 6a_{3,2} + 3a_{4,2} + 6a_{5,2} + 4.$$

So, we solve

$$\begin{cases} 4a_{4,2} + 6a_{5,2} + 4 = 0 \\ 4a_{1,2} + 6a_{3,2} + 3a_{4,2} + 6a_{5,2} + 4 = 0 \end{cases}$$

and get $(a_{1,2}, a_{3,2}, a_{4,2}, a_{5,2}) = (0, 0, 0, 4)$.

Proposed Algorithm - Toy Example(4/6)

$$F_1 \mapsto {}^tT_2 F_1 T_2 = \begin{pmatrix} 0 & 0 & 0 & 4 & 6 \\ 0 & 6 & 6 & 1 & 1 \\ 0 & 2 & 6 & 0 & 4 \\ 0 & 5 & 0 & 4 & 3 \\ 0 & 3 & 0 & 0 & 6 \end{pmatrix}$$

$$F_2 \mapsto {}^tT_2 F_1 T_2 = \begin{pmatrix} 2 & 0 & 6 & 3 & 6 \\ 0 & 3 & 6 & 3 & 0 \\ 0 & 1 & 1 & 2 & 2 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 4 \end{pmatrix}$$

These matrices mean the following polynomials.

$$6x_2^2 + x_1(4x_4 + 6x_5) + x_2(x_3 + 6x_4 + 4x_5) + 6x_3^2 + 4x_3x_5 + 4x_4^2 + 3x_4x_5 + 6x_5^2$$
$$2x_1^2 + 3x_2^2 + x_1(6x_3 + 3x_4 + 6x_5) + x_2(3x_4 + 2x_5) + x_3^2 + 2x_3x_4 + 2x_3x_5 + 4x_5^2$$

Proposed Algorithm - Toy Example(5/6)

Step 3

$$\begin{cases} 6x_2^2 + x_1(4x_4 + 6x_5) + x_2(x_3 + 6x_4 + 4x_5) + 6x_3^2 + 4x_3x_5 + 4x_4^2 + 3x_4x_5 + 6x_5^2 = 0 \\ 2x_1^2 + 3x_2^2 + x_1(6x_3 + 3x_4 + 6x_5) + x_2(3x_4 + 2x_5) + x_3^2 + 2x_3x_4 + 2x_3x_5 + 4x_5^2 = 0 \end{cases}$$

We solve these linear equations

$$\begin{cases} 4x_4 + 6x_5 = 0 \\ x_3 + 6x_4 + 4x_5 = 0 \\ 6x_3 + 3x_4 + 6x_5 = 0 \end{cases}$$

$$\begin{array}{|cc|} \hline L_{1,1} & L_{1,2} \\ L_{2,1} & L_{2,2} \\ \hline \end{array}$$

and get $(x_3, x_4, x_5) = (1, 6, 3)$. Then, we obtain the following quadratic equations.

$$\begin{cases} 6x_2^2 + 4 = 0 \\ 2x_1^2 + (3x_2^2 + 3x_2) + 6 = 0 \end{cases}$$

Proposed Algorithm - Toy Example(5/6)

Step 3

$$\begin{cases} 6x_2^2 + x_1(4x_4 + 6x_5) + x_2(x_3 + 6x_4 + 4x_5) + 6x_3^2 + 4x_3x_5 + 4x_4^2 + 3x_4x_5 + 6x_5^2 = 0 \\ 2x_1^2 + 3x_2^2 + x_1(6x_3 + 3x_4 + 6x_5) + x_2(3x_4 + 2x_5) + x_3^2 + 2x_3x_4 + 2x_3x_5 + 4x_5^2 = 0 \end{cases}$$

We solve these linear equations

$$\begin{cases} 4x_4 + 6x_5 = 0 \\ x_3 + 6x_4 + 4x_5 = 0 \\ 6x_3 + 3x_4 + 6x_5 = 0 \end{cases}$$

$$\begin{array}{|cc|} \hline L_{1,1} & L_{1,2} \\ L_{2,1} & L_{2,2} \\ \hline \end{array}$$

and get $(x_3, x_4, x_5) = (1, 6, 3)$. Then, we obtain the following quadratic equations.

$$\begin{cases} 6x_2^2 + 4 = 0 \\ 2x_1^2 + (3x_2^2 + 3x_2) + 6 = 0 \end{cases}$$

Proposed Algorithm - Toy Example(6/6)

We solve these quadratic equations

$$\begin{cases} 6x_2^2 + 4 = 0 \\ \\ 2x_1^2 + (3x_2^2 + 3x_2) + 6 = 0 \end{cases}$$

and get $(x_1, x_2) = (3, 2)$.

This is a solution of ${}^t\boldsymbol{x}({}^tT_2F_1T_2)\boldsymbol{x} = 0$ and ${}^t\boldsymbol{x}({}^tT_2F_2T_2)\boldsymbol{x} = 0$,
so a solution of $(*)$ is

$$\begin{aligned} \begin{pmatrix} x_1 & x_2 & x_3 & x_4 & x_5 \end{pmatrix} &= \begin{pmatrix} 3 & 2 & 1 & 6 & 3 \end{pmatrix}{}^tT_2 \\ &= \begin{pmatrix} 3 & 2 & 1 & 6 & 4 \end{pmatrix} \end{aligned}$$

Proposed Algorithm - Analysis (1/2)

We can show facts about the proposed algorithm.

Theorem

The proposed algorithm works when $n \geq m(m+3)/2$.

This condition depends on the number of unknowns and equations of linear equations.

Proposed Algorithm - Analysis (2/2)

Theorem

Let $k = \mathrm{GF}(q)$. The complexity of the proposed algorithm is

$$\begin{cases} O(n^w m(\log q)^2) & (\text{char } k \text{ is } 2) \\ O(2^m n^w m(\log q)^2) & (\text{char } k \text{ is odd}) \end{cases}$$

where $2 \leq w \leq 3$ is the exponent of the Gaussian elimination.

The complexity of this algorithm depends on the probability of existence of square roots over finite field k.
Success probability of this algorithm is about 2^{-m} when char k is odd.

Experimental Results

Table: Computer specification

OS	CPU	RAM	Software
Windows 7 (64bit)	Intel Core i3 (1.33GHz)	4.00 GB	Magma V2.17-9

Table: Performance

Field	n	m	Time / a try	Success probability
$\mathrm{GF}(2^8)$	16	4	8.76 (ms)	99.99 %
	84	11	506.83 (ms)	100 %
	504	28	78710 (ms)	100 %
$\mathrm{GF}(7)$	16	4	3.99 (ms)	11.83 %
	84	11	259.28 (ms)	0.22 %

Conclusion & Future works

Conclusion

- We proposed an algorithm which can solve the MQ-Problem when $n \geq m(m+3)/2$.
  $\rightarrow$ We can make the range of solvable MQ-Problems wider!

- We implemented this algorithm with Magma and checked this algorithm works with high probability.

Future works

- To make the applicable range wider.
- To apply the proposed algorithm to the algorithm by Thomae et al. [PKC'12].

That's all.

Thank you for your attention.