

# **MXL2: Solving Polynomials Equations over GF(2) Using an Improved Mutant Strategy**

Jintai Ding  
University of Cincinnati  
Johannes Buchmann  
Mohamed Saied Emam Mohamed  
Wael Said Abd Elmageed Mohamed  
TU Darmstadt

# Outline

- Motivation
- MQ problem
- MutantXL Algorithm
- MXL2 Algorithm
- Practical Results
- Conclusions

# Motivation

- Factoring and discrete logarithm: insecure under the assumption that quantum computer with enough Qbits exist
- Multivariate based cryptosystems: potential to resist the quantum computer attacks
- Breaking a good cipher should require “as much work as solving a system of simultaneous equations in a large number of unknowns”

Solving Multivariate system efficiently

# The MQ Problem

Given finite set  $P$  of polynomials in  $X = (x_1, \dots, x_n)$   
over finite field  $F$

Find  $v \in F^n$ ,  $p(v) = 0 \forall p \in P$

MQ is NP-complete

Fraenkel, Yesha (1977), Goubin, Patarin (1997)

# Solving Algorithms and Strategies

- Linearization based algorithms
  - Relinearization
  - XL algorithm
  - XL Variants
  - MutantXL
- Gröbner based algorithms
  - Buchberger's algorithm
  - F4 Algorithm
  - F5 algorithm

# XL Algorithm

Solve  $P(v) = 0$ , degree 2

*Initialize:*  $D = 2$

*Solve:* Apply Gauss  $\rightarrow$  row echelon form  
If system solved, return solution, terminate

*Enlarge:* Increment  $D$  by 1  
Multiply all  $p$  in  $P$  by all monomials such  
that degree is  $D$ , include in  $P$ , go to Solve

# Ding's Idea

“In the process of space enlargement, we should treat elements of different degrees differently.”

In particular, the elements of lower degree should play a more important role than the ones of higher degree.

# Mutants

$$I = \langle P \rangle$$

$$\forall g \in I, \quad g = \sum_{p \in P} g_p p, \quad g_p \in R$$

Level of this representation = maximum degree of  $g_p p$ .

Level of  $g$  = minimum level of all representations.

If degree  $g <$  Level of  $g$ ,  $g$  is Mutant with respect to  $P$

**These mutants are not used in XL**

# Mutant XL

*Initialize:*  $D = 2, ED = 2, M = \phi$  and  $U = \phi$ .

*Gauss:*  $P \rightarrow \tilde{P}, P = \tilde{P}$

*ExtractUnivariates:*  $U \leftarrow \{p \in P : p \text{ is univariate}\}$ .

*Solve:* If  $U \neq \phi$  then solve and substitute. If  $P = \phi$  return the solution and terminate, else  $D = ED = \max\{\deg(p) : p \in P\}$  and go to Gauss.

*ExtractMutants:*  $M \leftarrow \{p \in P : \deg(p) < ED\}$ .

*MultiplyMutants:* If  $M \neq \phi$ , then multiply all lower degree mutants, add the new polynomials to  $P, ED = k + 1 (k = \min\{\deg(p) : p \in M\})$  and go back to Gauss.

*Enlarge:* Multiply all higher degree polynomials,  $D = D + 1, ED = D$ , and go back to Gauss.

# MutantXL toy example

HFE system of 4 equations in 4 variables over  $\mathbb{F}_2$

$$p_1 : x_1x_2 + x_2x_3 + x_2x_4 + x_3x_4 + x_1 + x_3 + 1 = 0$$

$$p_2 : x_1x_2 + x_1x_3 + x_1x_4 + x_3x_4 + x_2 + x_3 + 1 = 0$$

$$p_3 : x_1x_2 + x_1x_3 + x_2x_3 + x_3x_4 + x_1 + x_4 + 1 = 0$$

$$p_4 : \quad \quad x_1x_3 + x_1x_4 + x_2x_3 + x_2x_4 + 1 = 0$$

# MutantXL toy example

Gauss:

$$\tilde{p}_1 : x_1 x_2 + x_2 x_3 + x_2 x_4 + x_3 x_4 + x_1 + x_3 + 1 = 0$$

$$\tilde{p}_2 : x_1 x_3 + x_1 x_4 + x_2 x_3 + x_2 x_4 + x_1 + x_2 = 0$$

$$\tilde{p}_3 : x_1 x_4 + x_2 x_3 + x_1 + x_2 + x_3 + x_4 = 0$$

$$\tilde{p}_4 : x_1 + x_2 + 1 = 0$$

$\tilde{p}_4$  is mutant

# MutantXL toy example

Multiply  $\tilde{p}_4$  by all monomials  $x_1, \dots, x_4$  to get

$$p_5 : \quad x_1 x_2 = 0$$

$$p_6 : x_1 x_3 + x_2 x_3 + x_3 = 0$$

$$p_7 : x_1 x_4 + x_2 x_4 + x_4 = 0$$

# MutantXL toy example

Gauss:

$$\tilde{p}_5 : x_2 x_3 + x_2 x_4 + x_3 x_4 + x_1 + x_3 + 1 = 0$$

$$\tilde{p}_6 : \quad \quad \quad x_3 x_4 + x_1 + x_3 + x_4 + 1 = 0$$

$$\tilde{p}_7 : \quad \quad \quad \quad \quad \quad x_3 + x_4 + 1 = 0$$

# MutantXL toy example

Multiply  $\tilde{p}_7$  by all monomials  $\mathbf{x}_1, \dots, \mathbf{x}_4$ :

$$p_8 : x_1 x_3 + x_1 x_4 + x_1 = 0$$

$$p_9 : x_2 x_3 + x_2 x_4 + x_2 = 0$$

$$p_{10} : \quad \quad \quad x_3 x_4 = 0$$

# MutantXL toy example

Elimination:

$$\tilde{p}_8 : x_2 + x_4 = 0$$

$$\tilde{p}_9 : 0 = 0$$

$$\tilde{p}_{10} : x_4 + 1 = 0$$

MutantXL solves this system with  $D = 2$ .

**XL solves with  $D = 3$  with more equations.**

# Experiments

Which examples do we consider?

HFE examples using the code made by Segers  
(Sizes: 30,35,40,45,50,55)

<http://www.win.tue.nl/~henkvt/images/ReportSegersGB2-11-04.pdf>

HFE example 25 by hotaru distribution.

<http://cvs.sourceforge.jp/cgi-bin/viewcvs.cgi/hotaru/>

Random dense MQ systems generated by Courtois  
(sizes: 5,...,30)

<http://www.cryptosystem.net/aes/toyciphers.html>

# MutantXL Problems

MutantXL algorithm has two problems:

- 1) The number of lower degree mutants is very large.  
Many reductions to zero.
- 2) There are insufficient number of mutants. MutantXL algorithm behaves like XL algorithm.

# MutantXL Problems

MutantXL algorithm has two problems:

- 1) The number of lower degree mutants is very large.  
Many reductions to zero.

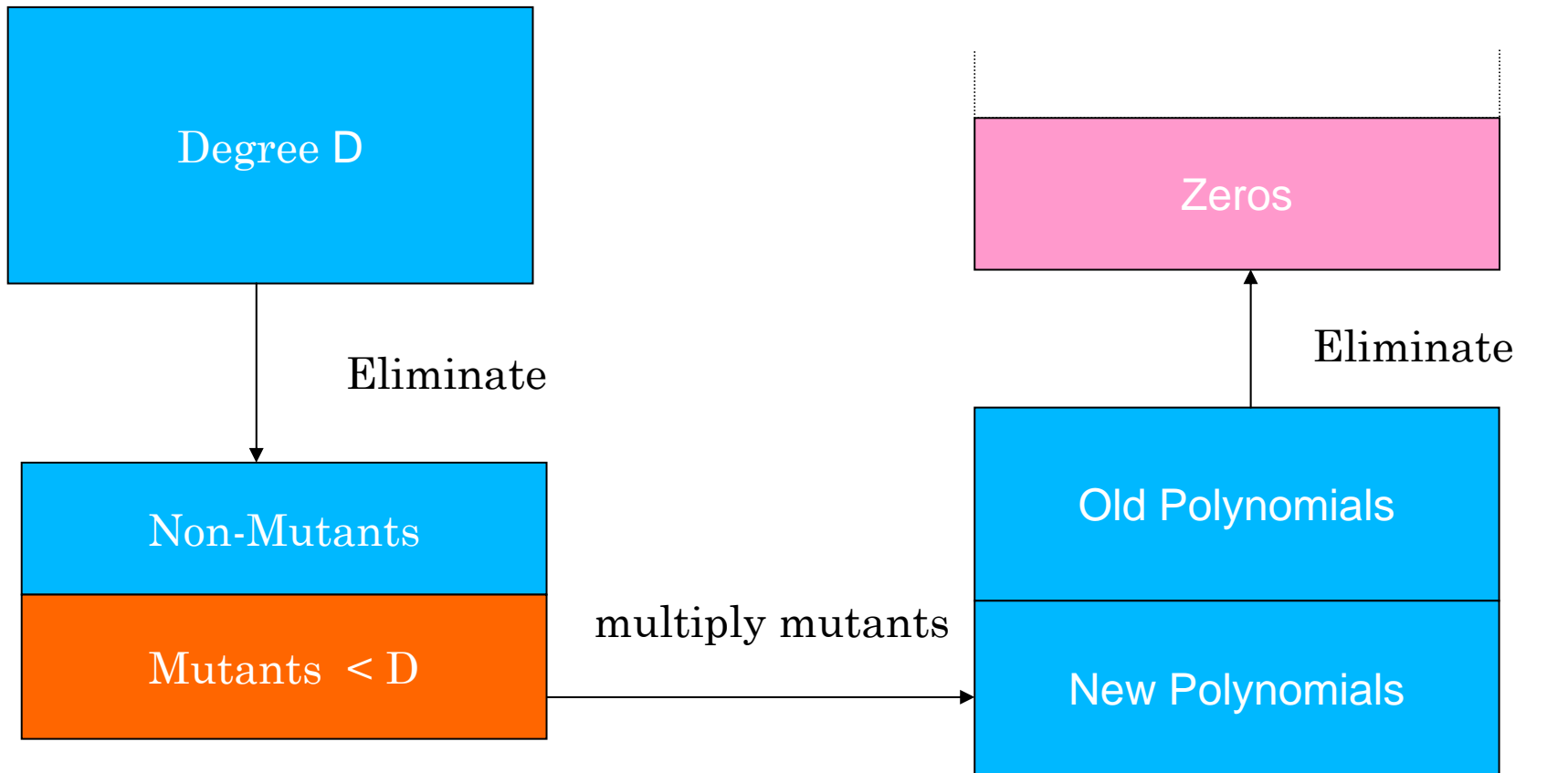
“Multiply only a necessary number of lower degree mutants.”

- 2) There are insufficient number of mutants. MutantXL algorithm behaves like XL algorithm.

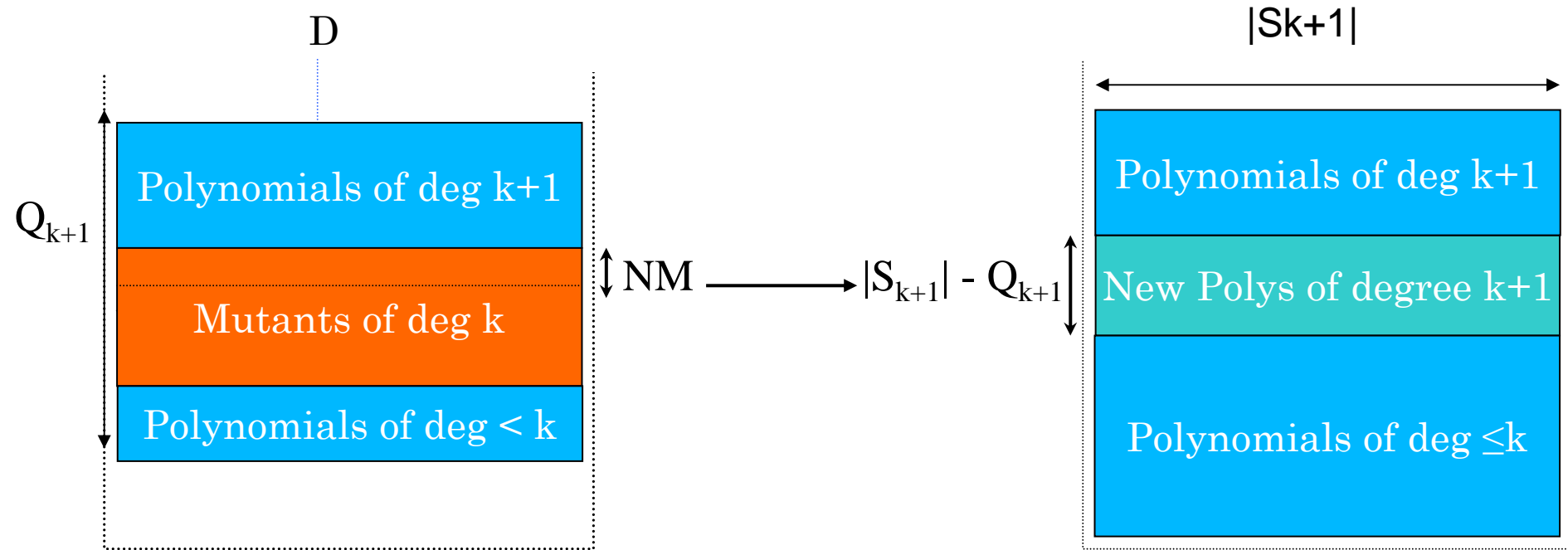
“Enlarge the system partially.”

# Improvements to MutantXL

Large number of mutants.



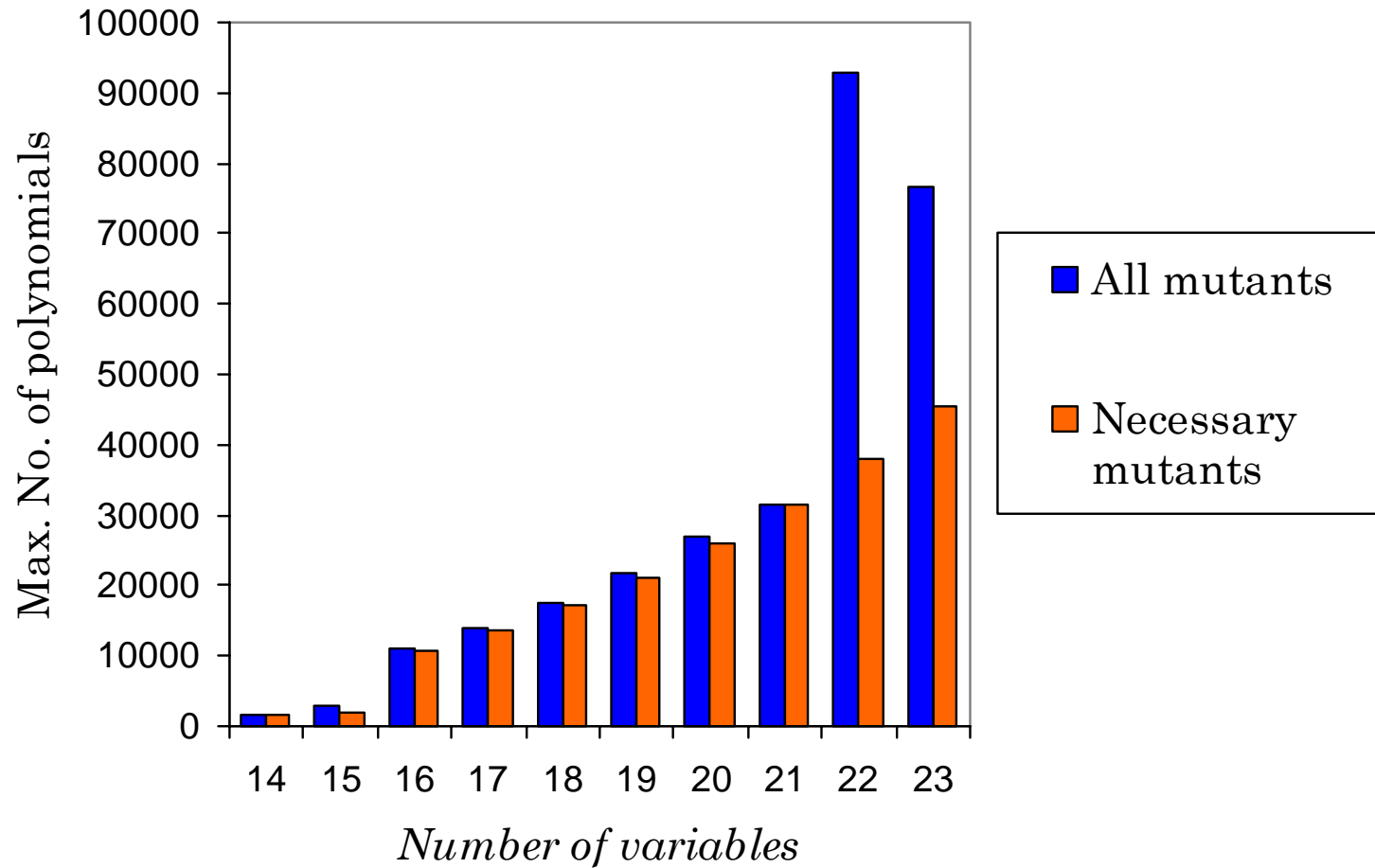
# Improvements to MutantXL



$$S_{k+1} = \{m \in R : \deg(m) \leq k + 1\} \quad |S_{k+1}| = \sum_{l=1}^{k+1} \binom{n}{l}, 1 \leq k + 1 \leq n$$

$$NM = \left\lceil (|S_{k+1}| - Q_{k+1}) / n \right\rceil$$

# Improvements to MutantXL



# Improvements to MutantXL

$$X = \{x_1, x_2, \dots, x_n\}, x_1 < x_2 < \dots < x_n$$

$$R = GF(2)[x_1, \dots, x_n] / (x_1^2 - x_1, \dots, x_n^2 - x_n)$$

Definition 1:

Leading Variable:  $LV(p) = x, p \in P \subset R$  and  $x \in X$   
 $x$  is the smallest variable in the leading term of  $p$ .

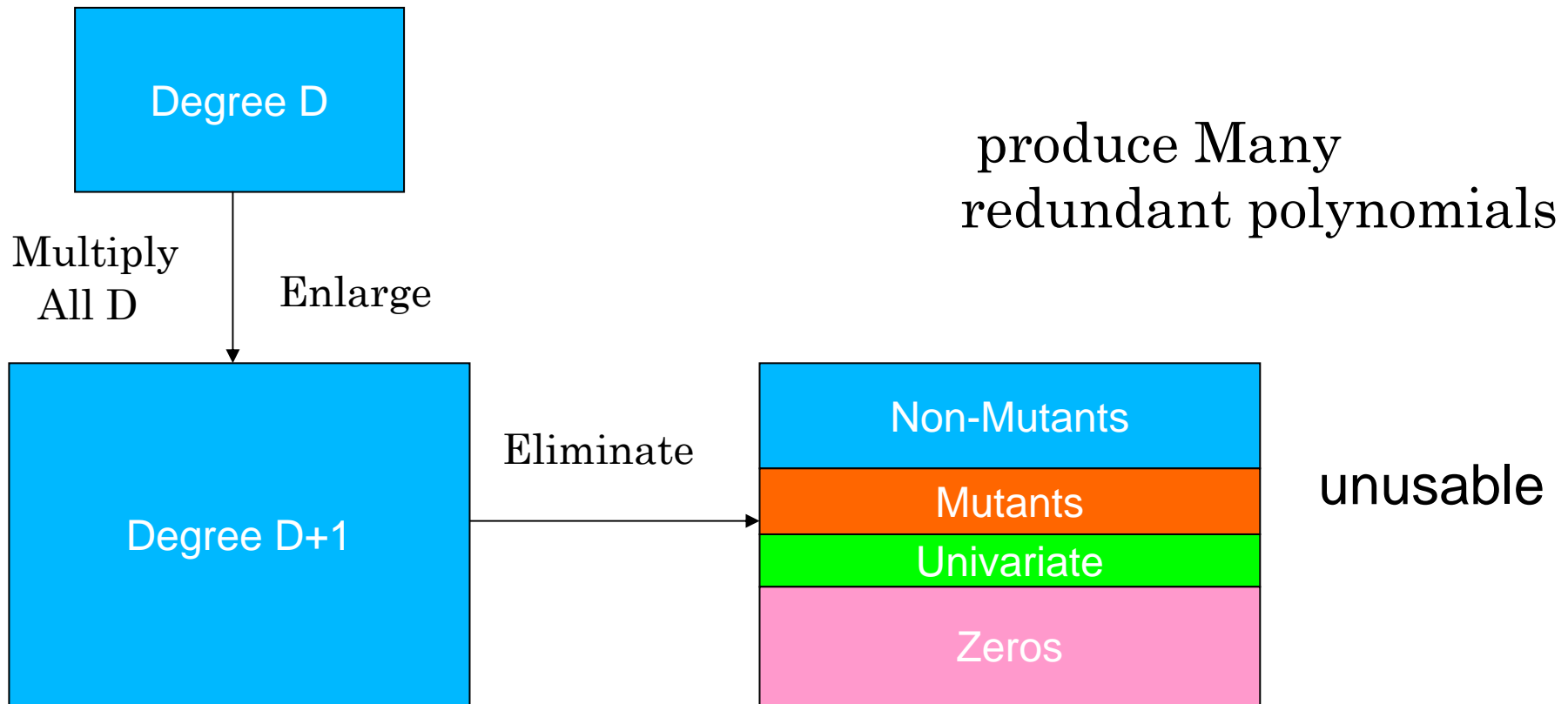
Definition 2:

$x$  Partition is  $P^x \subset P$  such that

$$P^x = \{p \in P : LV(p) = x\}$$

# Improvements to MutantXL

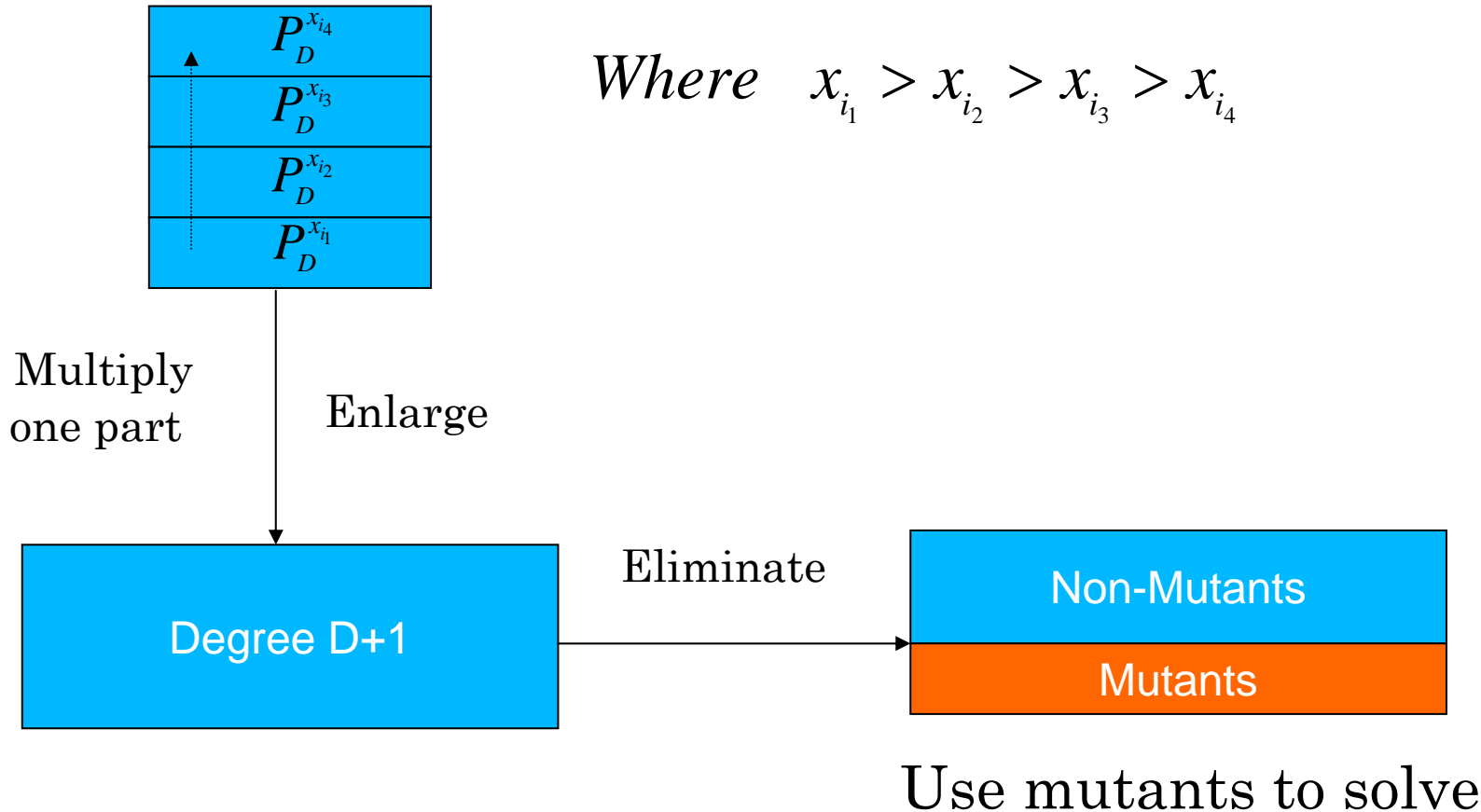
Insufficient number of mutants.



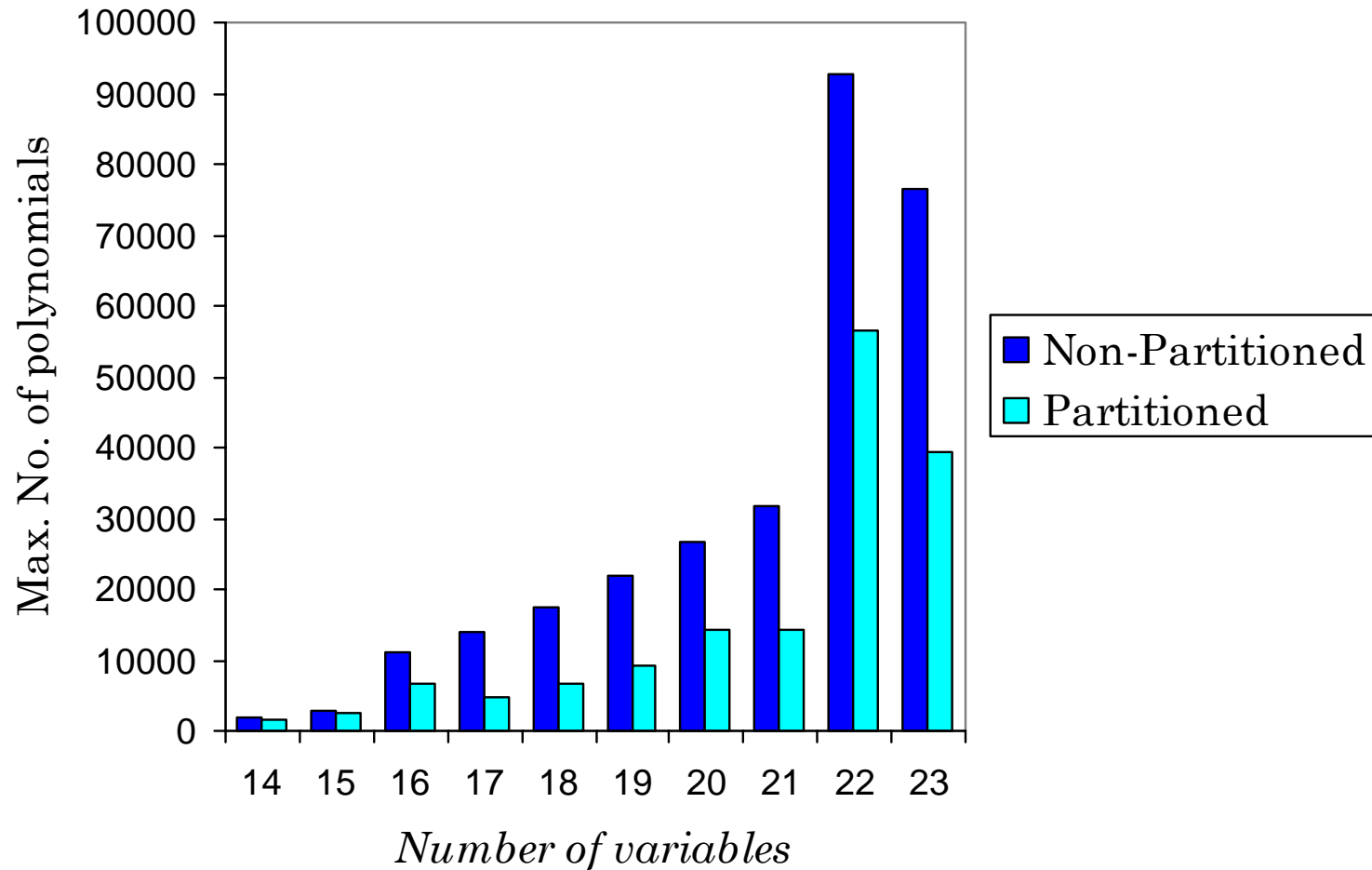
# Improvements to MutantXL

Partitioned enlargement technique

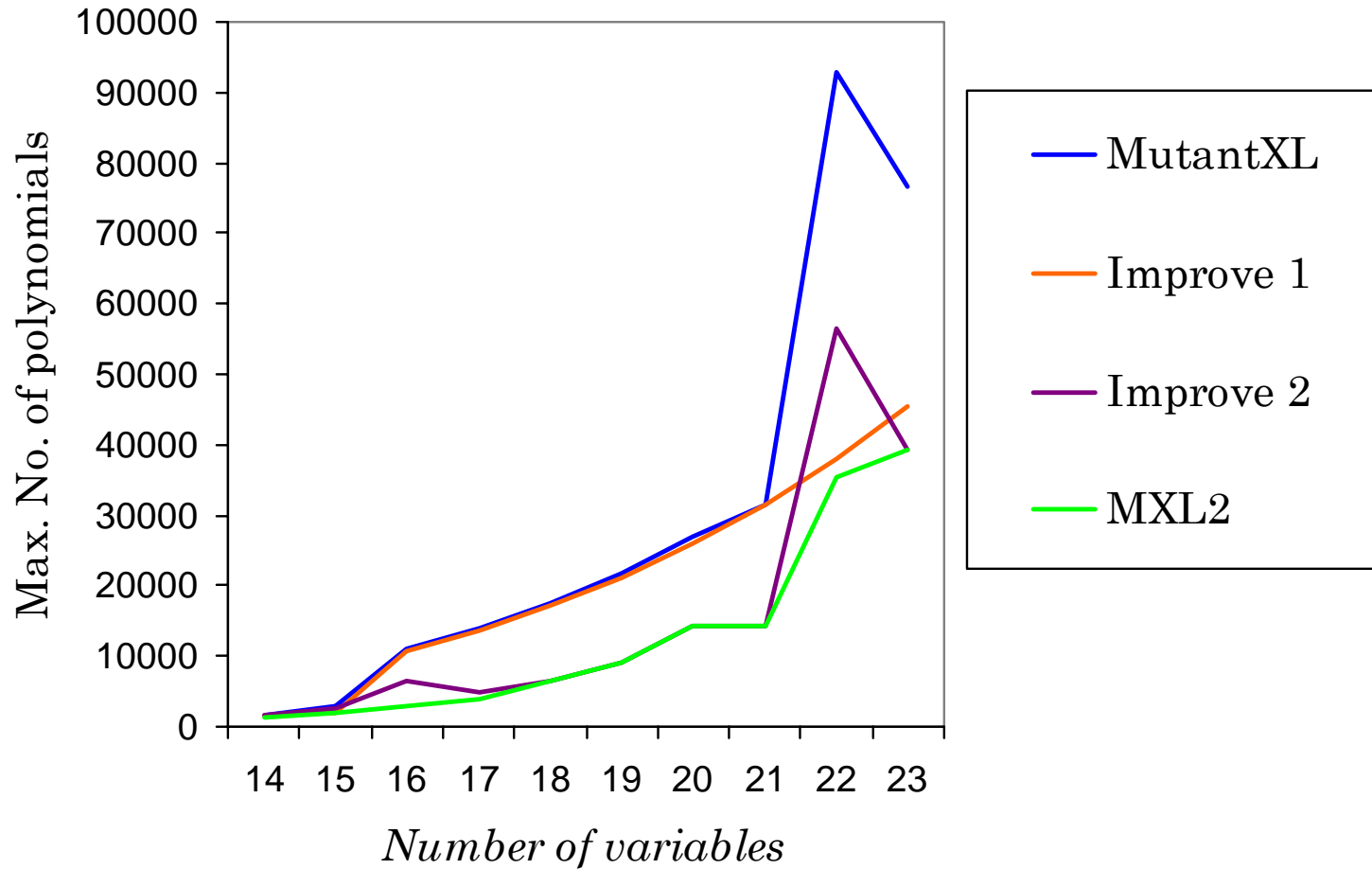
Where  $x_{i_1} > x_{i_2} > x_{i_3} > x_{i_4}$



# Improvements to MutantXL



# Improvements to MutantXL



# MXL2 algorithm

*Initialize:*  $D = 2, ED = 2, M = \phi$  and  $U = \phi$ .

*Gauss:*  $P \rightarrow \tilde{P}, P = \tilde{P}$

*ExtractUnivariates:*  $U \leftarrow \{p \in P : p \text{ is univariate}\}$ .

*Solve:* If  $U \neq \phi$  then solve and substitute. If  $P = \phi$  return the solution and terminate, else  $D = ED = \max\{\deg(p) : p \in P\}$  and go to Gauss.

*ExtractMutants:*  $M \leftarrow \{p \in P : \deg(p) < ED\}$ .

*MultiplyMutants:* If  $M \neq \phi$ , then multiply a **necessary number** lower degree mutants, add the new polynomials to  $P, ED = k + 1$  ( $k = \min\{\deg(p) : p \in M\}$ ) and go back to Gauss.

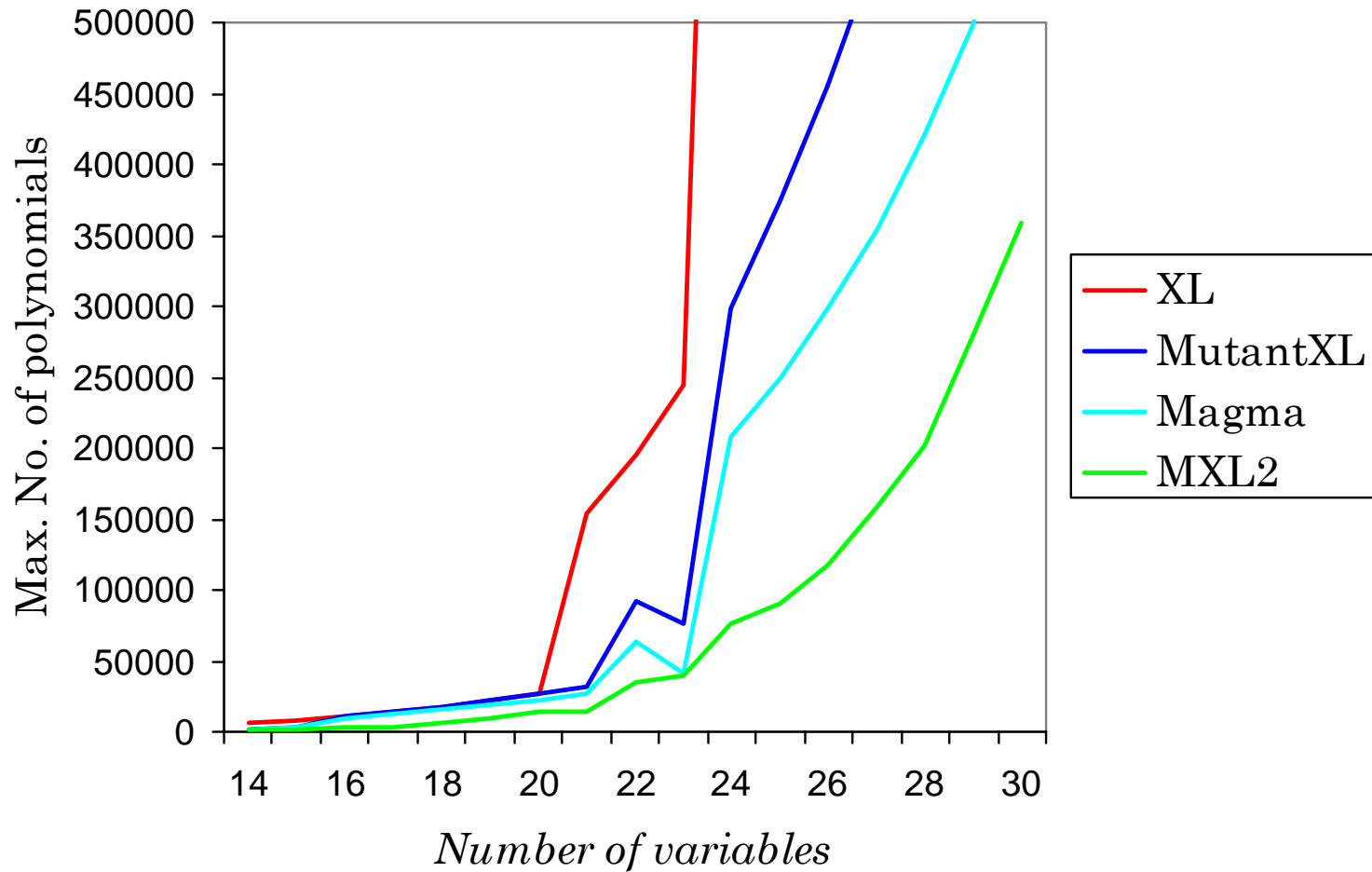
*Enlarge:* Multiply all higher degree polynomials, **partially, if this is the first partition**  $D = D + 1, ED = D$ , and go back to Gauss.

# Practical Results

Maximum matrix for random systems

#eq = #var	XL	MutantXL	Magma	MXL2
14	6475 x 3473	1771 x 1471	1538 x 1336	<b>1191 x 1185</b>
15	8520 x 4944	2786 x 2941	2639 x 1535	<b>1946 x 1758</b>
16	11016 x 6885	11016 x 6885	9993 x 4034	<b>2840 x 2861</b>
17	14025 x 9402	14025 x 9402	12382 x 5784	<b>3740 x 4184</b>
18	17613 x 12616	17613 x 12616	15187 x 8120	<b>6508 x 7043</b>
19	21850 x 16664	21850 x 16664	18441 x 11041	<b>9185 x 11212</b>
20	26810 x 21700	26810 x 21700	22441 x 14979	<b>14302 x 12384</b>
21	153405 x 82160	31641 x 27896	26860 x 19756	<b>14365 x 20945</b>
22	194579 x 110056	92831 x 35443	63621 x 21855	<b>35463 x 25342</b>
23	244145 x 145499	76558 x 44552	41866 x 29010	<b>39263 x 36343</b>
24	No sol. Obtained	298477x190051	207150x78637	<b>75825 x 69708</b>

# Practical Results



# Practical Results

Maximum matrix for HFE systems

#eq = #var	HUD	Magma	MutantXL	MXL2
25	96	12495×15276	14219×15276	<b>11926×15276</b>
30	64	23832×31931	26922×31931	<b>19174×31931</b>
35	48	27644×59536	31255×59536	<b>30030×59536</b>
40	33	45210×102091	49620×102091	<b>46693×102091</b>
45	24	43575×164221	57734×164221	<b>45480×164221</b>
50	40	75012×251176	85025×251176	<b>67826×251176</b>
55	48	104068×368831	119515×368831	<b>60116×368831</b>

# MXL2 Implementation

- All algorithms are implemented by C/C++.
- For the finite field arithmetic operations, the package NTL is used.
- For matrix representations and operations, M4RI is used.
- For parsing polynomial, Flexlexer is used.
- The Standard Template Library (STL) is employed for building data structures such as polynomials, terms and variables.

# Conclusion

- We presented an improvement to the MutantXL algorithm.
- The size of the matrix constructed by MXL2 is smaller than that of the matrix constructed by MutantXL.
- MXL2 outperforms F4 in solving random systems with respect to the memory resources.
- For HFE, in most cases MXL2 using less memory resources than F4.

Thanks