New Directions in Multivariate Public Key Cryptography

Shuhong Gao
Joint with Ray Heindl

Clemson University

The 4th International Workshop
on Finite Fields and Applications
Beijing University, May 28-30, 2010.

Public Key Cryptography

Diffie and Hellman (1976) proposed the concept of *public key cryptosystems* with which users can communicate without sharing any prior secret key.

Public Key Cryptography

Diffie and Hellman (1976) proposed the concept of *public key cryptosystems* with which users can communicate without sharing any prior secret key.

- **One-way function**: easy to evaluate, but hard to invert (on average).
- **Trapdoor one-way function**: easy to evaluate, but hard to invert without knowledge of some "trapdoor".

Public Key Cryptography

Diffie and Hellman (1976) proposed the concept of *public key cryptosystems* with which users can communicate without sharing any prior secret key.

- **One-way function**: easy to evaluate, but hard to invert (on average).
- **Trapdoor one-way function**: easy to evaluate, but hard to invert without knowledge of some "trapdoor".

A public key cryptosystem consists of a family of trapdoor one-way functions.

Examples

Candidate one-way function: Polynomials are easy to evaluate but NP-hard to invert, however, its average complexity is not known.

Examples

Candidate one-way function: Polynomials are easy to evaluate but NP-hard to invert, however, its average complexity is not known.

Univariate: Given $\alpha \in \mathbb{F}_q$, and a sparse polynomial

$$f(x) = \sum_{i=1}^{t} f_i x^{d_i} \in \mathbb{F}_q[x]$$

where q is large (say $q = 2^{5000}$), d_i's are as big as q, and t is small (say $t = 500$). Then it is easy to compute $f(\alpha)$, but it is NP-complete to decide, for a given $\beta \in \mathbb{F}_q$ and a sparse $f \in \mathbb{F}_q[x]$, whether there exists $\alpha \in \mathbb{F}_q$ such that $f(\alpha) = \beta$.

Examples

Candidate trapdoor one-way functions:

- RSA cryptosystem encryption: $f(x) = x^e \pmod{n}$ where n is a product two primes and $\gcd(e, \phi(n)) = 1$.
  "Trapdoor": the factorization of n, or an integer d such that $ed \equiv 1 \pmod{\phi(n)}$.

Examples

Candidate trapdoor one-way functions:

- RSA cryptosystem encryption: $f(x) = x^e (\mathrm{mod}\, n)$ where n is a product two primes and $\gcd(e, \phi(n)) = 1$.
  "Trapdoor": the factorization of n, or an integer d such that $ed \equiv 1 (\mathrm{mod}\, \phi(n))$.
- Discrete log based cryptosystems: G is a cyclic group generated by α of order n. Pick a random integer $k \in \{1, 2, \ldots, n-1\}$ as a "trapdoor" and let $\beta = \alpha^k$ be public. The encryption function is then

$$f(x) = (x \oplus \beta^r, \alpha^r)$$

where $r \in \{1, 2, \ldots, n-1\}$is random for each x.

Public key cryptosystems

- **Practical cryptosystems**: RSA cryptosystem based on integer factorization and cryptosystems based on discrete log in finite fields and elliptic curves.

Public key cryptosystems

- **Practical cryptosystems**: RSA cryptosystem based on integer factorization and cryptosystems based on discrete log in finite fields and elliptic curves.
- **Main Problem**: Quantum computers, if built, can factor integers and solve discrete logs in polynomial time (Shor 1997).

Public key cryptosystems

- **Practical cryptosystems**: RSA cryptosystem based on integer factorization and cryptosystems based on discrete log in finite fields and elliptic curves.
- **Main Problem**: Quantum computers, if built, can factor integers and solve discrete logs in polynomial time (Shor 1997).
- **Possible alternatives**:
  - Lattice-based cryptography
  - Code-based cryptography
  - Hash-based cryptography
  - Multivariate cryptography

Multivariate Public Key Cryptography

NP-complete problem: Given a system of quadratic polynomials $F = (f_1, \ldots, f_m) \in \mathbb{F}_q[X_1, \ldots, X_n]^m$ and a point $y = (y_1, \ldots, y_m) \in \mathbb{F}_q^m$, decide whether there exists a point $x = (x_1, \ldots, x_n) \in \mathbb{F}_q^n$ such that

$$F(x) = y.$$

To build a multivariate cryptosystem, we hope to disguise a "nice polynomial system" as an arbitrary quadratic polynomial system.

MPKC: Structure

- Let $k = \mathbb{F}_q$ be a finite field with q elements.
- Public Key: $\bar{F} : k^n \longrightarrow k^m$ given by

$$\bar{F}(x_1, \ldots, x_n) = \begin{pmatrix} \bar{f}_1(x_1, \ldots, x_n) \\ \vdots \\ \bar{f}_m(x_1, \ldots, x_n) \end{pmatrix}^T$$

where $\bar{f}_i \in k[x_1, \ldots, x_n]$ are quadratic.

MPKC: Encryption/Decryption

Encryption:

$$(x_1, \ldots, x_n) \rightarrow \boxed{\bar{F}} \rightarrow (y_1, \ldots, y_m)$$

Decryption:

$$(y_1, \ldots, y_m) \rightarrow \boxed{\text{“}\bar{F}^{-1}\text{”}} \rightarrow (x_1, \ldots, x_n)$$

MPKC: Public Key Construction

We build the trapdoor one-way function $\bar{F}$ as

$$\bar{F} = L_1 \circ F \circ L_2,$$

where F is a multivariate map that can be easily inverted, and L_1, L_2 are invertible affine transformations, which are secret.

Evaluation:

$$(x_1, \ldots, x_n) \to \boxed{\bar{F}} \to (y_1, \ldots, y_m)$$

Inversion:

$$(y_1, \ldots, y_m) \to \boxed{L_1^{-1}} \to \boxed{F^{-1}} \to \boxed{L_2^{-1}} \to (x_1, \ldots, x_n)$$

MPKC: Security

$$\underline{\text{Public}} \qquad \underline{\text{Private}}$$
$$\bar{F} \quad = \quad L_1 \circ F \circ L_2$$

Security is dependent on the known difficulty of

- Solving quadratic multivariate systems over finite fields.
- Factoring multivariate maps.

MPKC: Efficiency

- Encryption: polynomial evaluation
- Decryption: depends on the system
- public Key: coefficients of quadratic systems

Goal: Construct secure MPKC

- Difficult problem: Invertibility, security, and efficiency are interrelated, and a good system must satisfy all three requirements at the same time.
- Some past success, but mostly with signature schemes.

Existing Cryptosystems

- Matsumoto-Imai (1988): generalizes of RSA using monomials
- Hidden Field Equations (Patarin 1996)
- Oil-Vinegar (Patarin 1997)
- Triangular (Fell and Diffie 1985, Shamir 1993, Moh 1999, Yang and Chen 2004)
- Other systems: Rainbow (Ding and Schmidt 2005), MFE (Wang et al. 2006), ℓ-IC (Ding et al. 2007)

Jintai Ding, Jason Gower, and Deiter Schmidt, *Multivariate Public Key Cryptosystems*, Springer (2006).

Triangular Systems

Central map:

$$F(x_1, \ldots, x_n) = \begin{pmatrix} x_1 \\ x_2 + g_2(x_1) \\ \vdots \\ x_{n-1} + g_{n-1}(x_1, x_2, \ldots, x_{n-2}) \\ x_n + g_n(x_1, x_2, \ldots, x_{n-2}, x_{n-1}) \end{pmatrix}^T$$

Triangular Systems

Central map:

$$F(x_1, \ldots, x_n) = \begin{pmatrix} x_1 \\ x_2 + g_2(x_1) \\ \vdots \\ x_{n-1} + g_{n-1}(x_1, x_2, \ldots, x_{n-2}) \\ x_n + g_n(x_1, x_2, \ldots, x_{n-2}, x_{n-1}) \end{pmatrix}^T$$

Inversion: iteratively solve for each component.

Triangular Systems

Central map:

$$F(x_1, \ldots, x_n) = \begin{pmatrix} x_1 \\ x_2 + g_2(x_1) \\ \vdots \\ x_{n-1} + g_{n-1}(x_1, x_2, \ldots, x_{n-2}) \\ x_n + g_n(x_1, x_2, \ldots, x_{n-2}, x_{n-1}) \end{pmatrix}^T$$

Inversion: iteratively solve for each component.

Weakness: triangular structure (first polynomial is **linear**, next few are too simple).

Oil-Vinegar Systems (Patarin 1997)

Consider two sets of variables $\{\check{x}_1, \ldots, \check{x}_v\}$ and $\{x_1, \ldots, x_o\}$. An **oil-vinegar polynomial** has the form

$$\sum a_{ij}\check{x}_i\check{x}_j + \sum b_{ij}\check{x}_i x_j + \sum c_i\check{x}_i + \sum d_i x_i + e.$$

Oil-Vinegar Systems (Patarin 1997)

Consider two sets of variables $\{\check{x}_1, \ldots, \check{x}_v\}$ and $\{x_1, \ldots, x_o\}$. An **oil-vinegar polynomial** has the form

$$\sum a_{ij}\check{x}_i\check{x}_j + \sum b_{ij}\check{x}_i x_j + \sum c_i\check{x}_i + \sum d_i x_i + e.$$

Given a system of o oil-vinegar polynomials, $F = (f_1, \ldots, f_o)$, if we substitute field values for $\check{x}_1, \ldots, \check{x}_v$, the result is an $o \times o$ linear system, which is linear in the oil variables $x_1, \ldots, x_o$.

Oil-Vinegar Systems (Patarin 1997)

Consider two sets of variables $\{\check{x}_1, \ldots, \check{x}_v\}$ and $\{x_1, \ldots, x_o\}$. An **oil-vinegar polynomial** has the form

$$\sum a_{ij}\check{x}_i\check{x}_j + \sum b_{ij}\check{x}_i x_j + \sum c_i\check{x}_i + \sum d_i x_i + e.$$

Given a system of o oil-vinegar polynomials, $F = (f_1, \ldots, f_o)$, if we substitute field values for $\check{x}_1, \ldots, \check{x}_v$, the result is an $o \times o$ linear system, which is linear in the oil variables $x_1, \ldots, x_o$.

Signature generation: Given a document $(y_1, \ldots, y_o)$, choose $(\check{x}_1, \ldots, \check{x}_v) \in k^v$ at random, and solve the resulting linear system for $x_1, \ldots, x_o$. The signature is then $(\check{x}_1, \ldots, \check{x}_v, x_1, \ldots, x_o)$.

New Framework for MPKCs

We propose a new framework for constructing central maps that combines ideas from Triangular and Oil-Vinegar Systems.

New Framework for MPKCs

We propose a new framework for constructing central maps that combines ideas from Triangular and Oil-Vinegar Systems.

Benefits:

- Combine Triangular and Oil-Vinegar systems to build *encryption* schemes.

New Framework for MPKCs

We propose a new framework for constructing central maps that combines ideas from Triangular and Oil-Vinegar Systems.

Benefits:

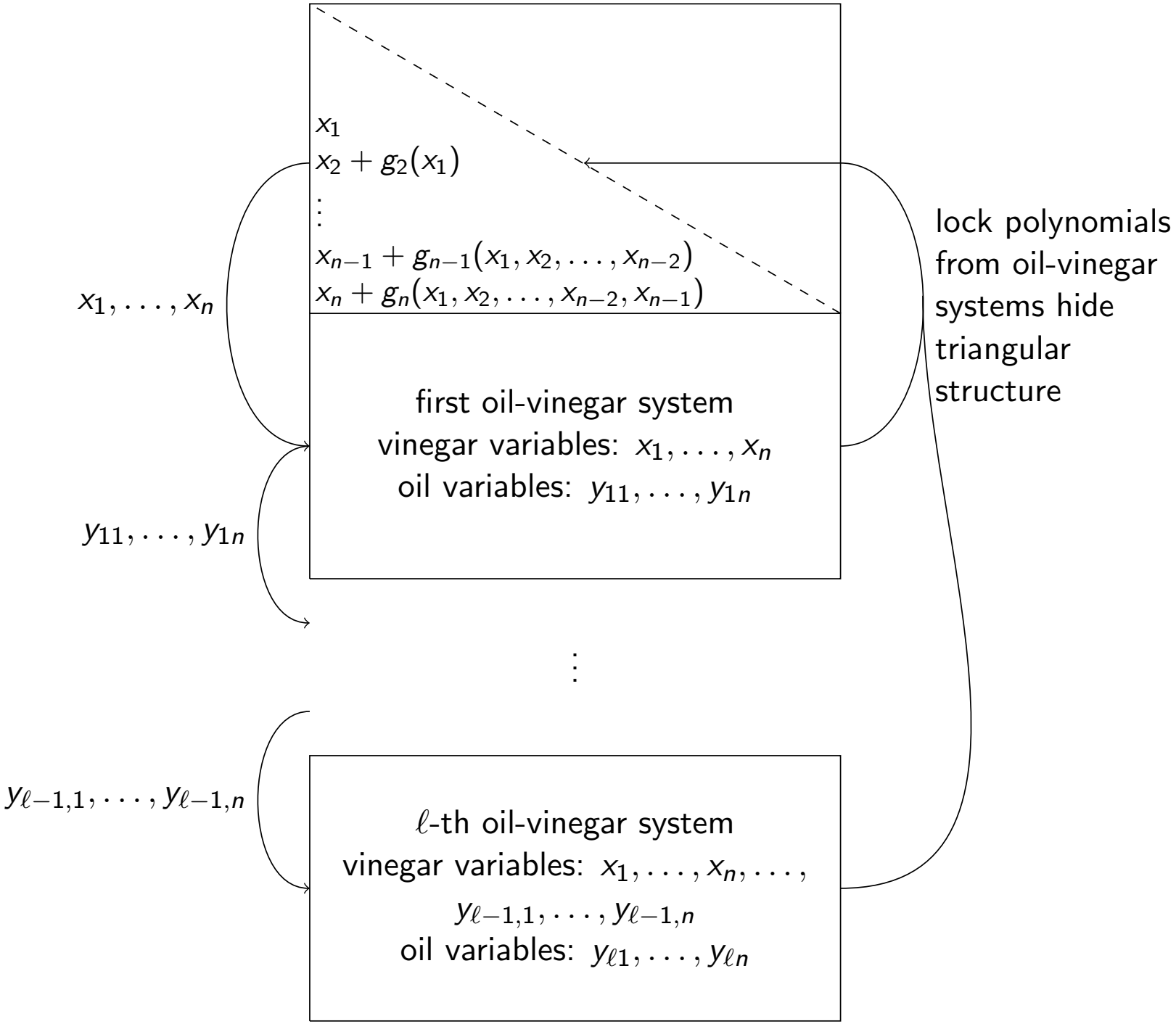
- Combine Triangular and Oil-Vinegar systems to build *encryption* schemes.
- Achieve invertibility more generally, then address the issues of security and efficiency.

New Framework for MPKCs

We propose a new framework for constructing central maps that combines ideas from Triangular and Oil-Vinegar Systems.

Benefits:

- Combine Triangular and Oil-Vinegar systems to build *encryption* schemes.
- Achieve invertibility more generally, then address the issues of security and efficiency.
- Introduce flexibility in construction.


x_1
$x_2 + g_2(x_1)$
$x_{n-1} + g_{n-1}(x_1, x_2, \ldots, x_{n-2})$
$x_n + g_n(x_1, x_2, \ldots, x_{n-2}, x_{n-1})$
$x_1, \ldots, x_n$
first oil-vinegar system
vinegar variables: $x_1, \ldots, x_n$
oil variables: $y_{11}, \ldots, y_{1n}$
$y_{11}, \ldots, y_{1n}$
$y_{\ell-1,1}, \ldots, y_{\ell-1,n}$
ℓ-th oil-vinegar system
vinegar variables: $x_1, \ldots, x_n, \ldots,$
$y_{\ell-1,1}, \ldots, y_{\ell-1,n}$
oil variables: $y_{\ell 1}, \ldots, y_{\ell n}$
lock polynomials from oil-vinegar systems hide triangular structure

Example: MFE Cryptosystem (Wang et al. 2006)

It is based on the following identity. Let

$$A(X) = \det \begin{pmatrix} X_1 & X_3 \\ X_4 & X_2 \end{pmatrix}, \quad A(Y) = \det \begin{pmatrix} Y_1 & Y_3 \\ Y_4 & Y_2 \end{pmatrix},$$

and

$$\begin{pmatrix} X_1 & X_3 \\ X_4 & X_2 \end{pmatrix} \begin{pmatrix} Y_1 & Y_3 \\ Y_4 & Y_2 \end{pmatrix} = \begin{pmatrix} f_1 & f_3 \\ f_4 & f_2 \end{pmatrix}.$$

Taking determinants on both sides gives the identity

$$A(X)B(Y) = f_1 f_2 - f_3 f_4.$$

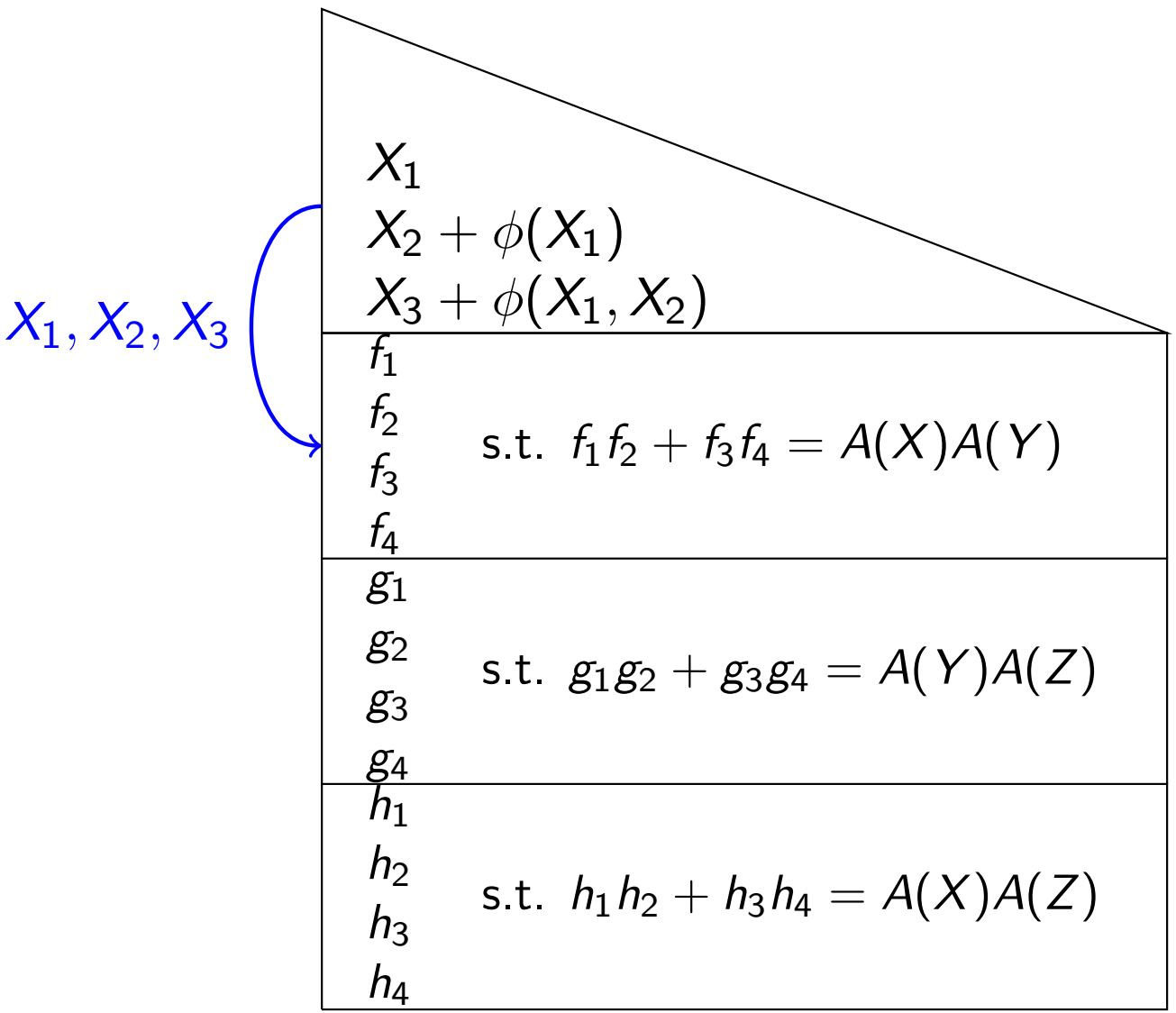
Example: MFE Cryptosystem (Wang et al. 2006)

Input: $(X_1, X_2, X_3, Y_1, Y_2, Y_3, Z_1, Z_2, Z_3) \in k^9$

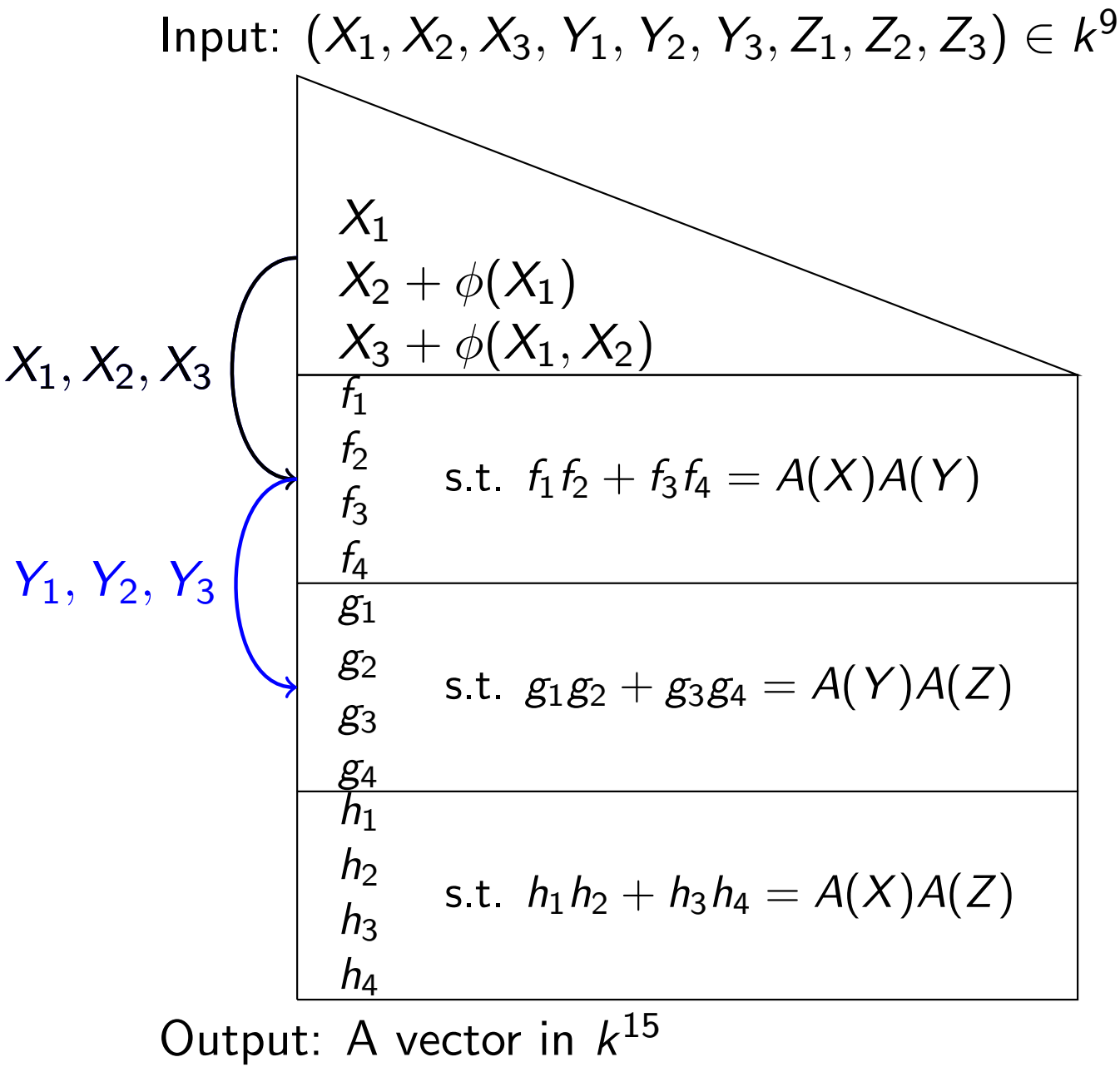
X_1 $X_2 + \phi(X_1)$ $X_3 + \phi(X_1, X_2)$	
f_1 f_2 f_3 f_4	s.t. $f_1 f_2 + f_3 f_4 = A(X)A(Y)$
g_1 g_2 g_3 g_4	s.t. $g_1 g_2 + g_3 g_4 = A(Y)A(Z)$
h_1 h_2 h_3 h_4	s.t. $h_1 h_2 + h_3 h_4 = A(X)A(Z)$

Output: A vector in k^{15}

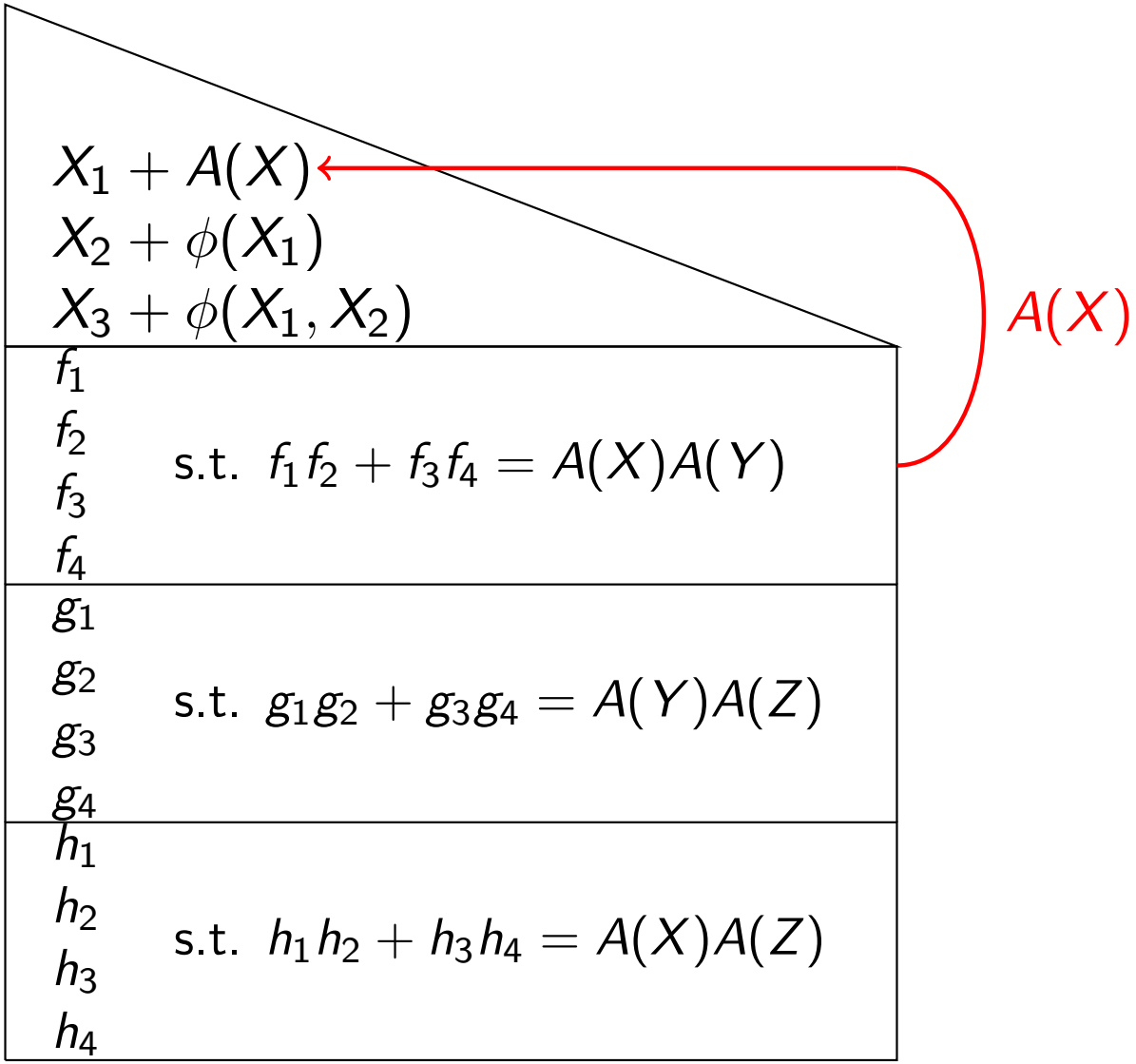
Example: MFE Cryptosystem (Wang et al. 2006)



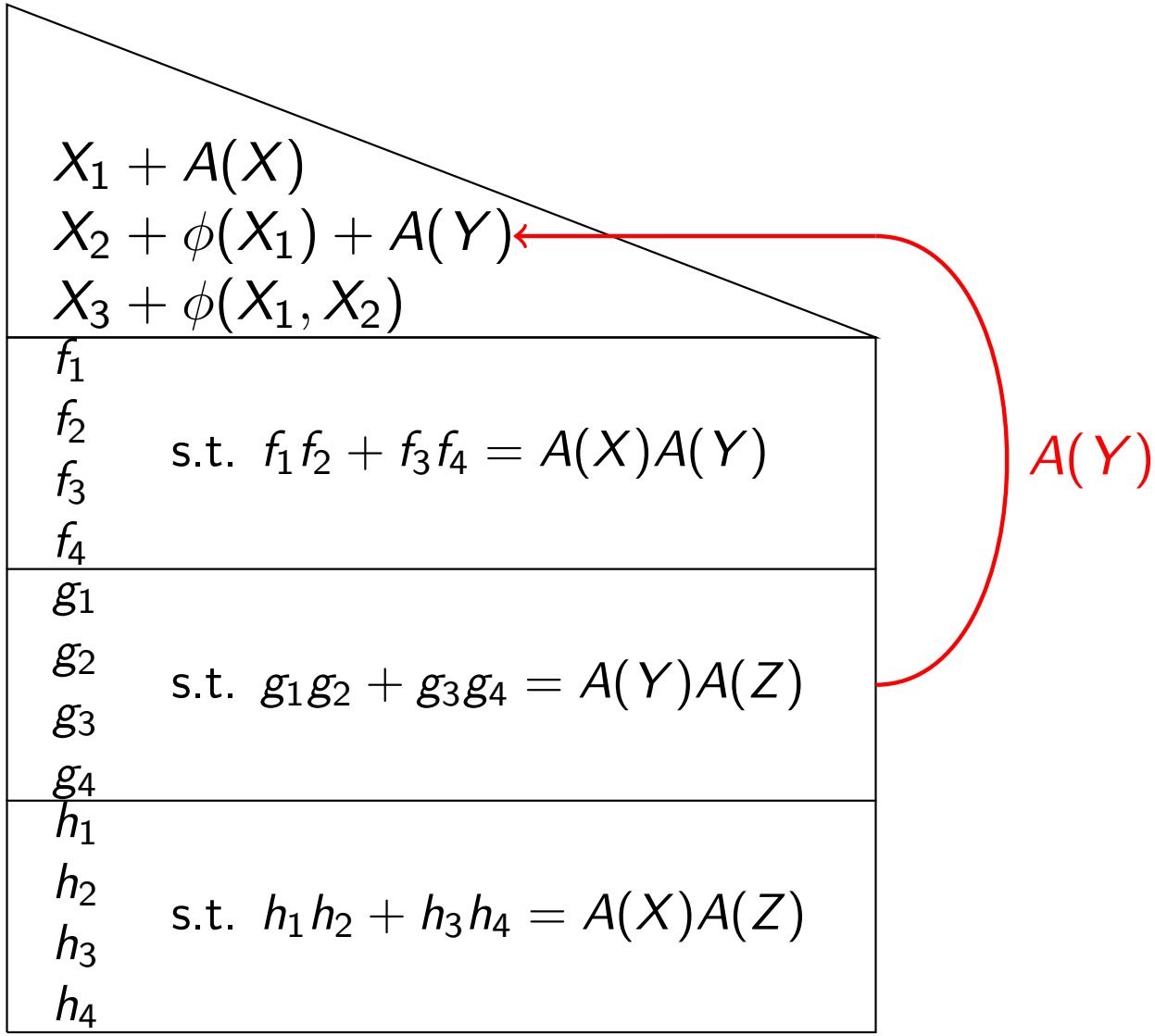
Example: MFE Cryptosystem (Wang et al. 2006)



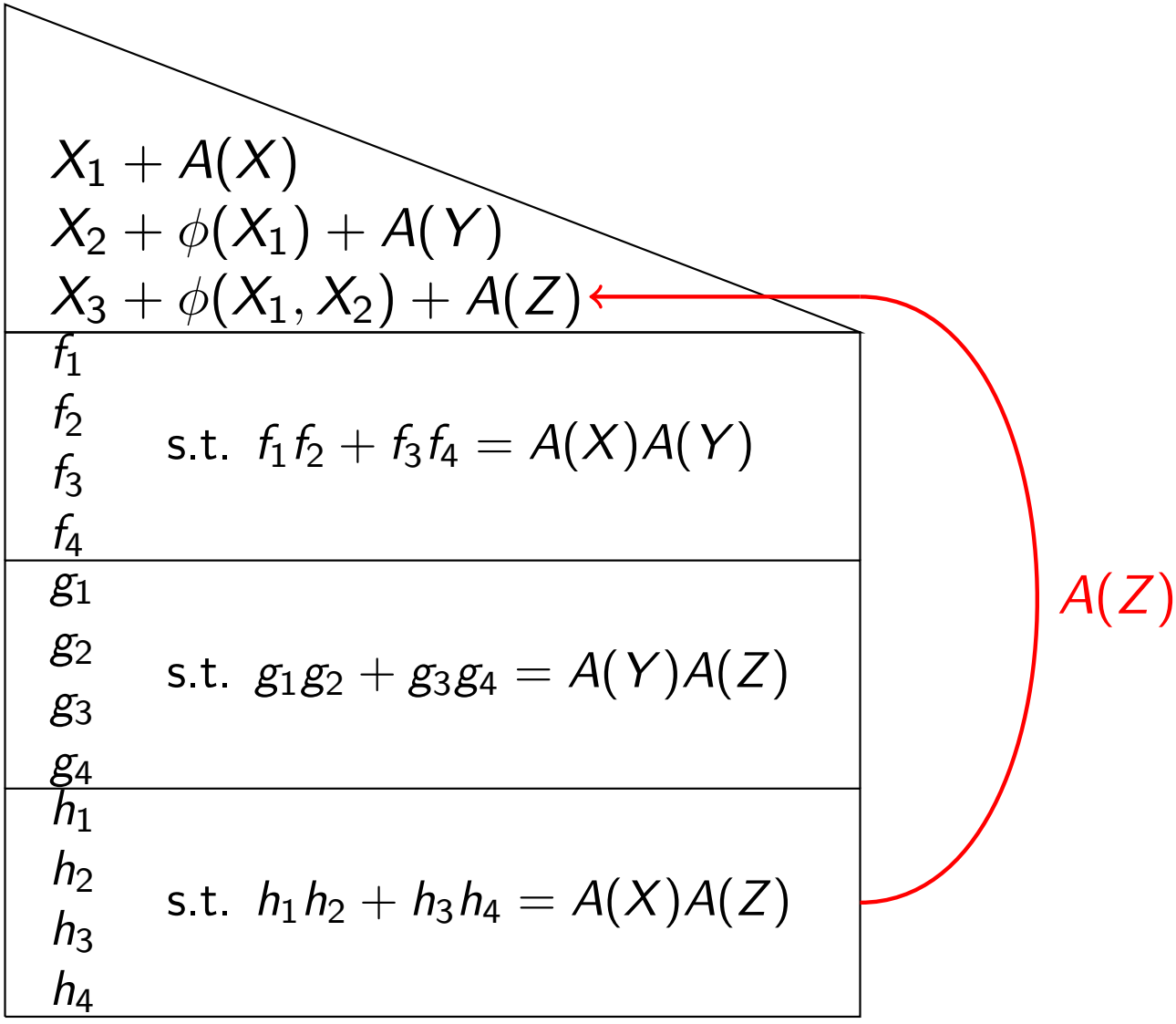
Example: MFE Cryptosystem (Wang et al. 2006)



Example: MFE Cryptosystem (Wang et al. 2006)



Example: MFE Cryptosystem (Wang et al. 2006)



Example: MFE Cryptosystem (Wang et al. 2006)

Input: $(X_1, X_2, X_3, Y_1, Y_2, Y_3, Z_1, Z_2, Z_3) \in k^9$

	$X_1 + A(X)$ $X_2 + \phi(X_1) + A(Y)$ $X_3 + \phi(X_1, X_2) + A(Z)$
f_1 f_2 f_3 f_4	s.t. $f_1 f_2 + f_3 f_4 = A(X)A(Y)$
g_1 g_2 g_3 g_4	s.t. $g_1 g_2 + g_3 g_4 = A(Y)A(Z)$
h_1 h_2 h_3 h_4	s.t. $h_1 h_2 + h_3 h_4 = A(X)A(Z)$

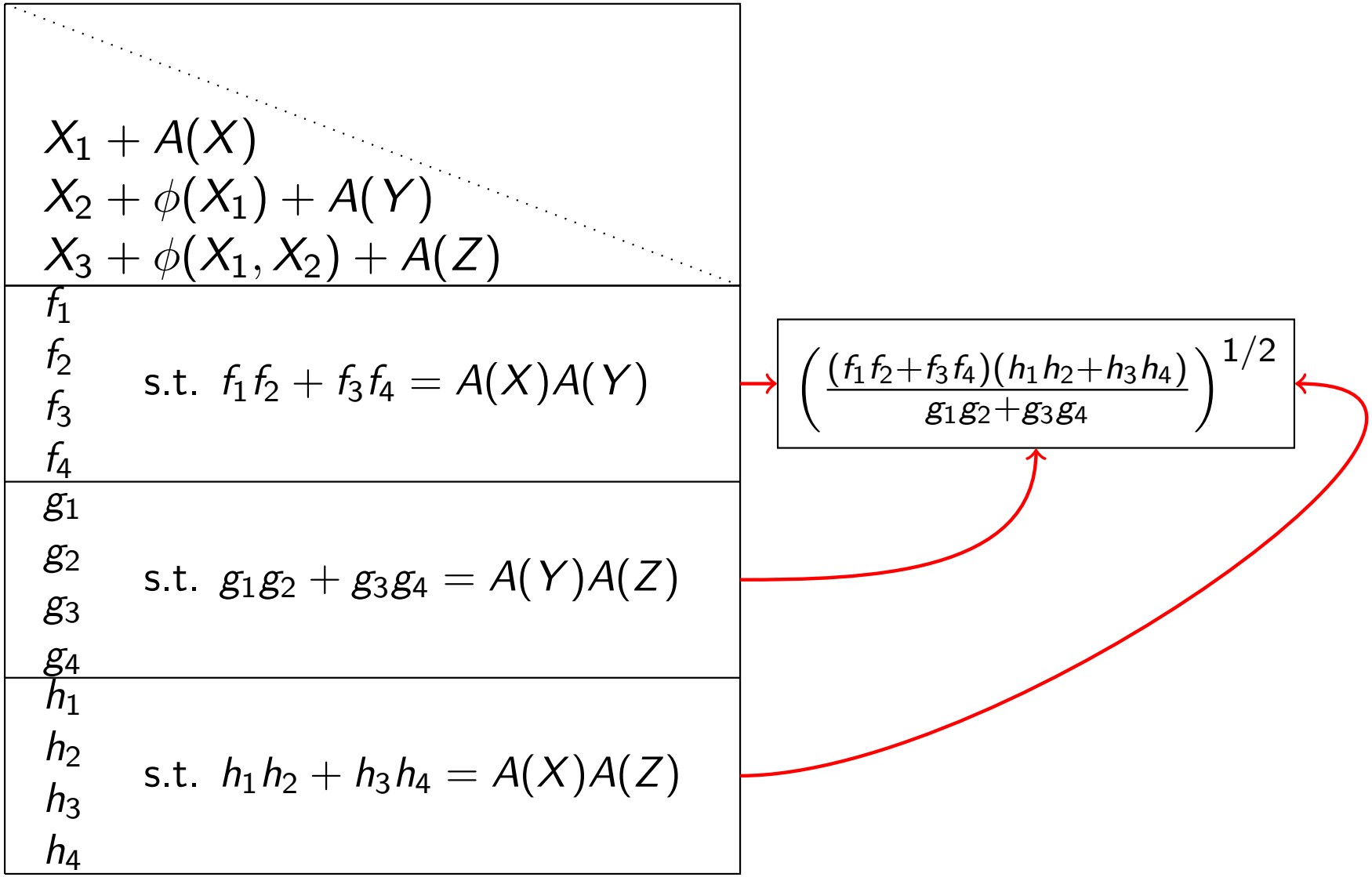
Output: A vector in k^{15}

Decryption

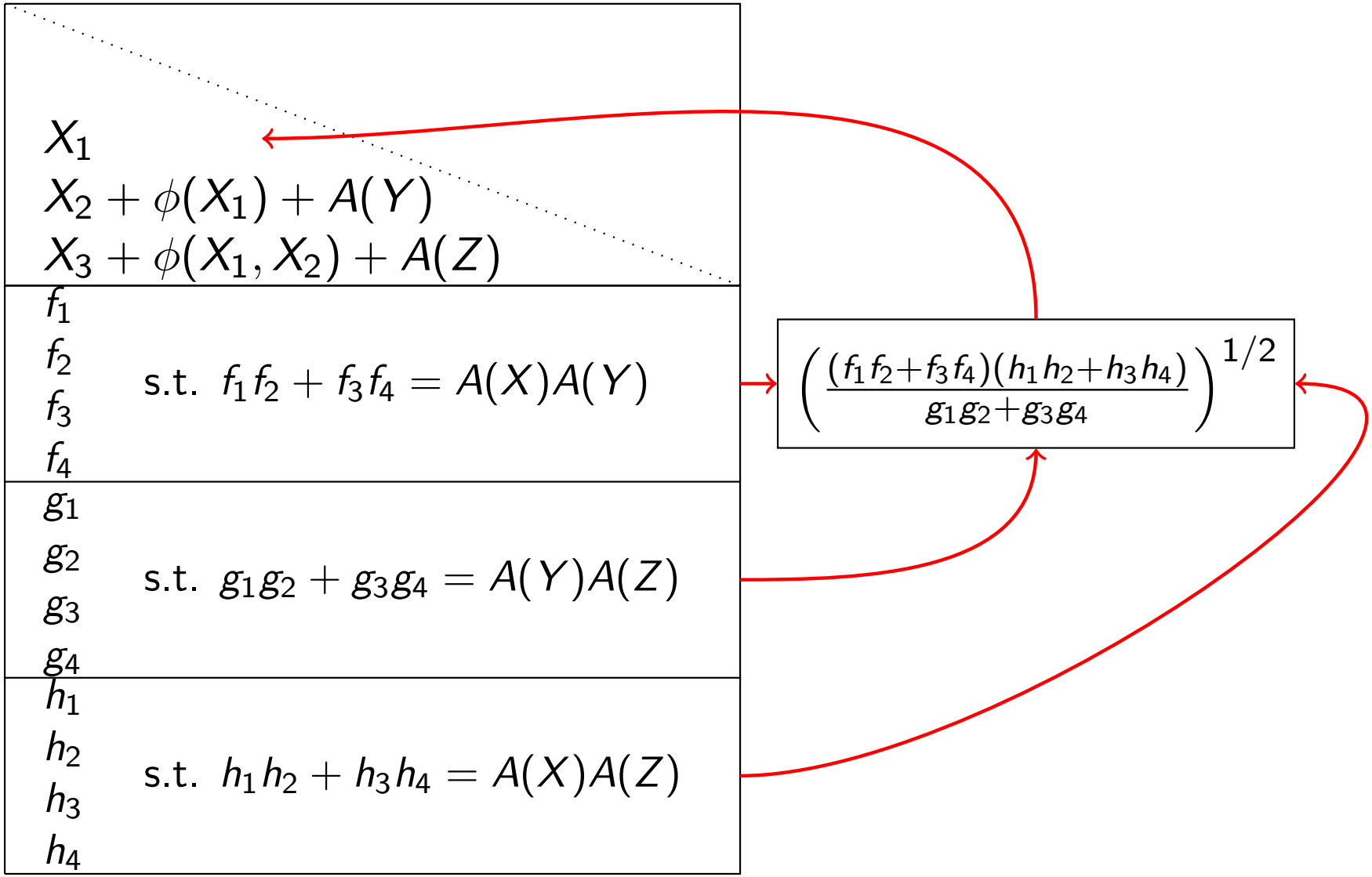
Decryption involves two steps:

1. Restoring the triangular structure and solving for the initial vinegar variables $x_1, \ldots, x_n$.
2. Iteratively solving the oil-vinegar systems for the remaining oil variables.

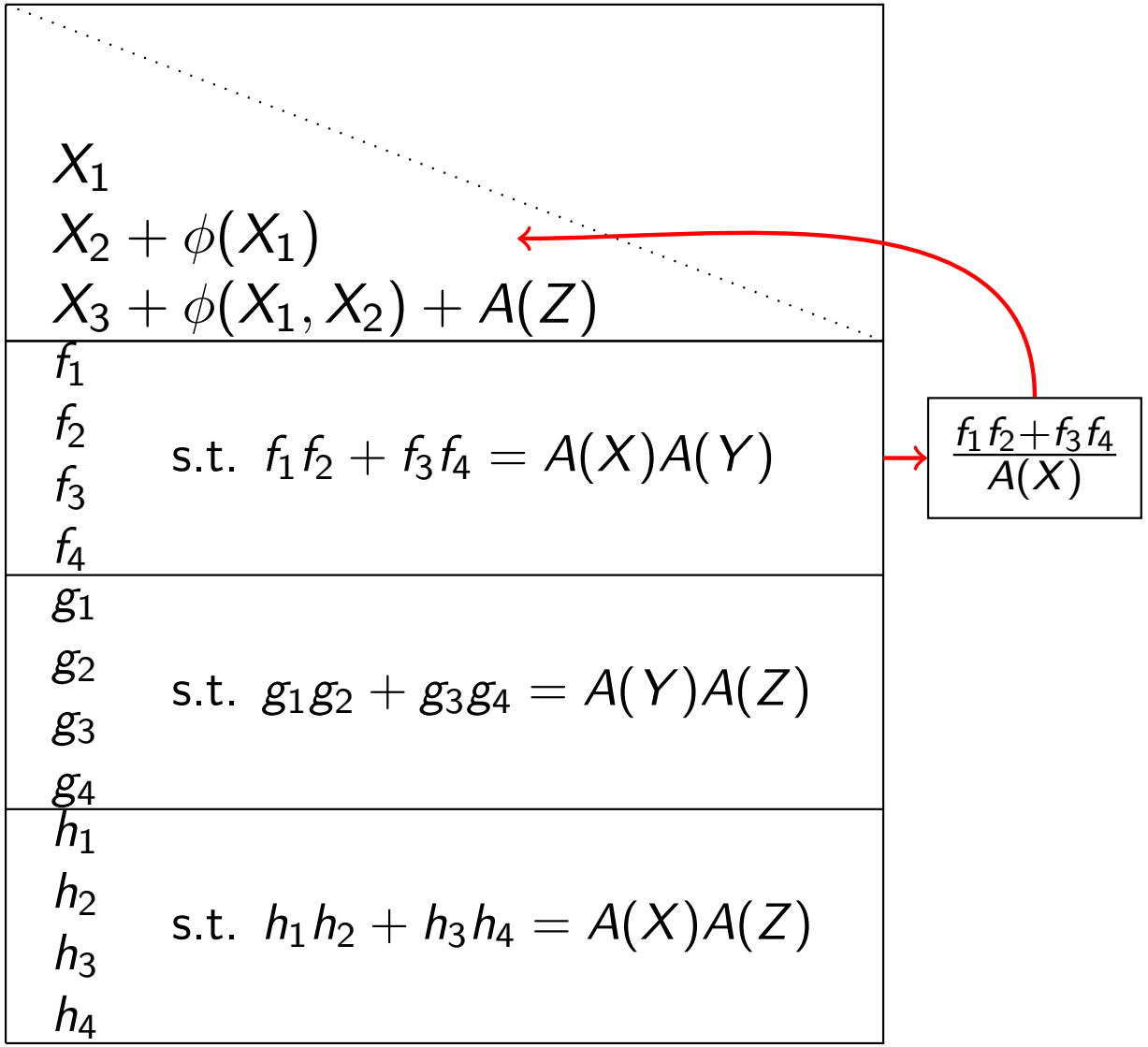
Decryption: Step 1 - Restore triangular structure



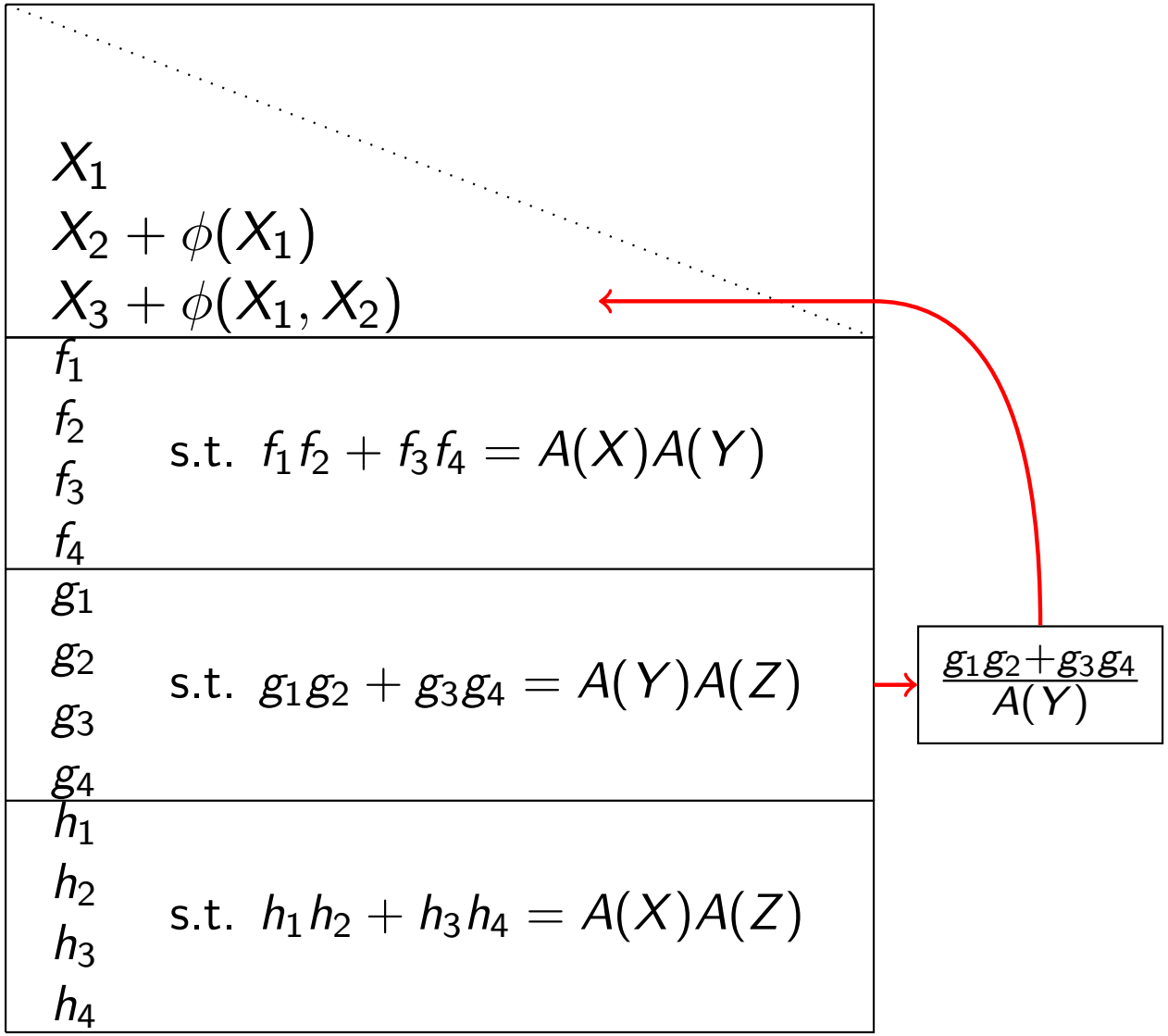
Decryption: Step 1 - Restore triangular structure



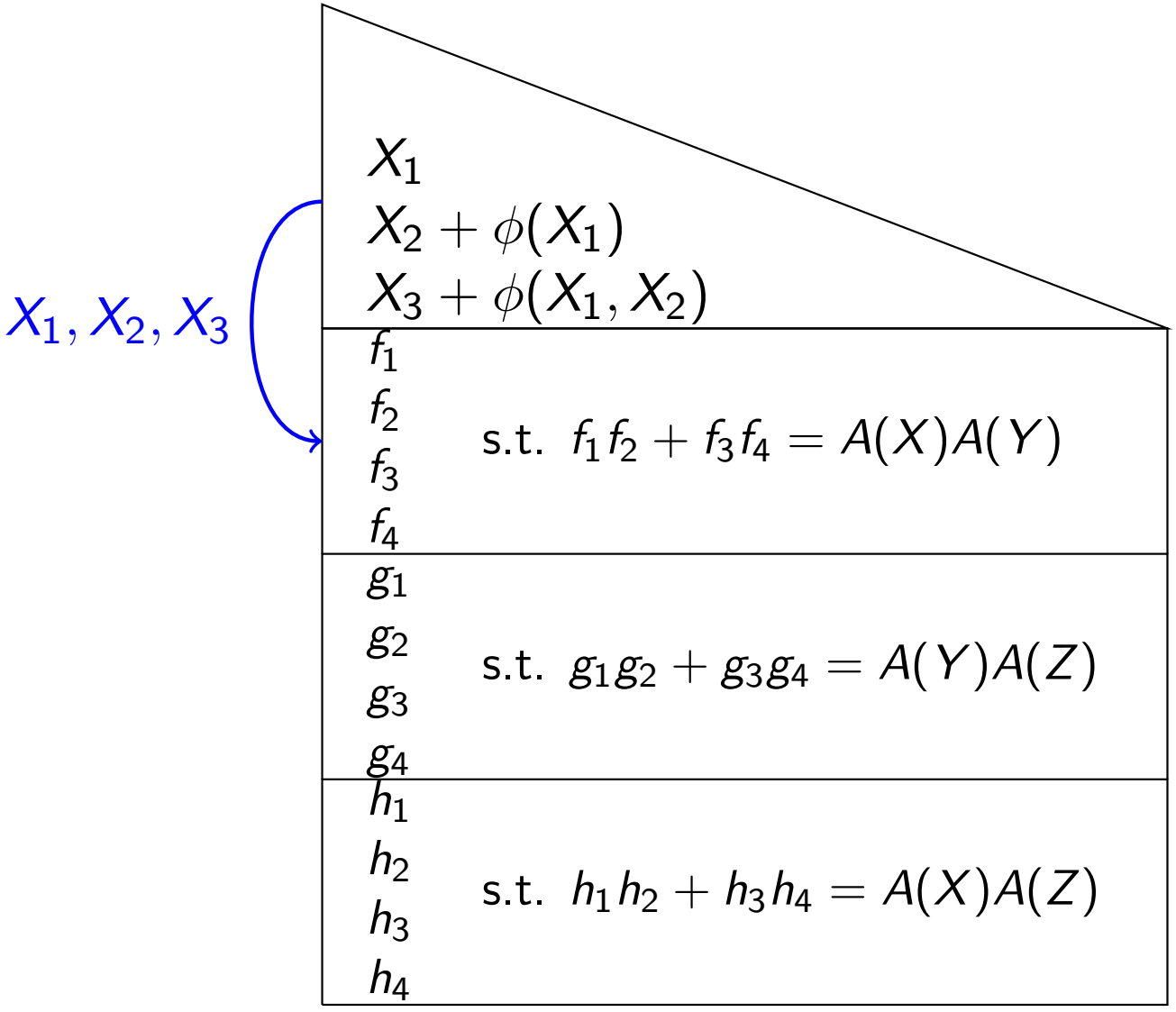
Decryption: Step 1 - Restore triangular structure



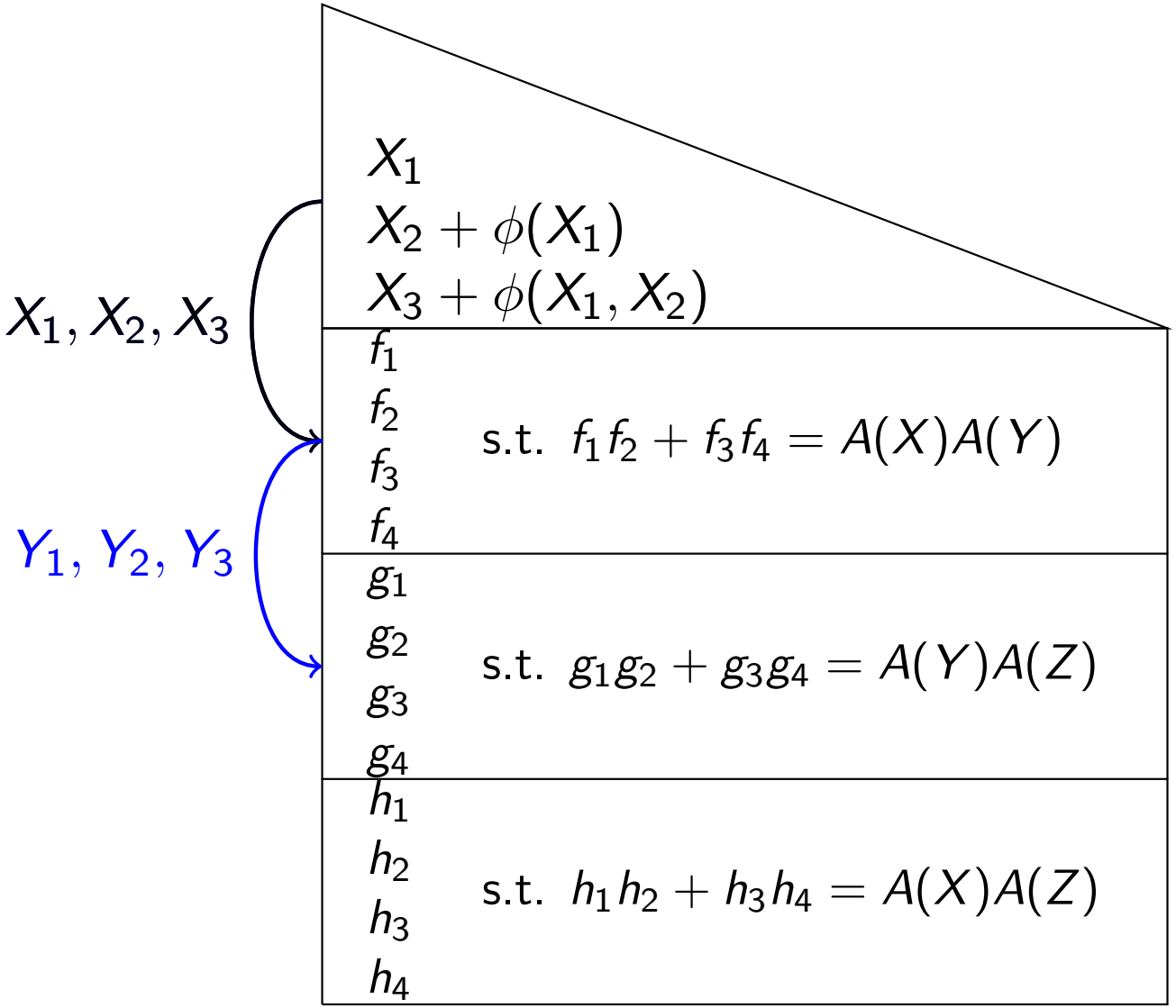
Decryption: Step 1 - Restore triangular structure



Decryption: Step 2 - Solve oil-vinegar systems



Decryption: Step 2 - Solve oil-vinegar systems



Constructing Polynomial Identities

Goal: construct identities of the form

$$A(X)B(Y) = f_1 f_2 + \cdots + f_{m-1} f_m$$

over the polynomial ring $k[X_1, \ldots, X_n, Y_1, \ldots, Y_n]$.

Toolbox:

1. Parameterization
2. Gröbner Basis methods
3. Plücker coordinates
4. Matrix Determinants
5. Grassmann coordinates
6. Graph theory

Polynomial identity in $k[X_1, \ldots, X_8, Y_1, \ldots, Y_8]$

$$A(X)B(Y) = f_1 f_2 + f_3 f_4 + f_5 f_6 + f_7 f_8 + f_9 f_{10},$$

$$\begin{aligned}
\text{where } A(X) &= X_1X_2 + X_3X_4 + X_5X_6 + X_7X_8 \\
B(Y) &= Y_1Y_2 + Y_3Y_4 + Y_5Y_6 + Y_7Y_8 \\
f_1 &= X_4Y_1 + X_8Y_4 + (X_1 + X_4)Y_5 + X_5Y_8 \\
f_2 &= (X_2 + X_3)Y_2 + X_7Y_3 + X_2Y_6 + X_6Y_7 \\
f_3 &= X_8Y_2 + X_4Y_3 + X_5Y_6 + (X_1 + X_4)Y_7 \\
f_4 &= X_7Y_1 + (X_2 + X_3)Y_4 + X_6Y_5 + X_2Y_8 \\
f_5 &= X_2Y_1 + X_6Y_4 + (X_2 + X_3)Y_5 + X_7Y_8 \\
f_6 &= (X_1 + X_4)Y_2 + X_5Y_3 + X_4Y_6 + X_8Y_7 \\
f_7 &= X_6Y_2 + X_2Y_3 + X_7Y_6 + (X_2 + X_3)Y_7 \\
f_8 &= X_5Y_1 + (X_1 + X_4)Y_4 + X_8Y_5 + X_4Y_8 \\
f_9 &= Y_1Y_7 + Y_2Y_8 + Y_3Y_5 + Y_4Y_6 \\
f_{10} &= X_1X_7 + X_2(X_5 + X_8) + X_3X_5 + X_4(X_6 + X_7).
\end{aligned}$$

Polynomial Identity

We introduce the following notation:

$$p_{xy}^{ij} = \begin{vmatrix} x_i & x_j \\ y_i & y_j \end{vmatrix} = x_i y_j - x_j y_i, \quad 1 \le i < j \le 4.$$

Define

$$|\mathbf{xyzw}| = \begin{vmatrix} x_1 & y_1 & z_1 & w_1 \\ x_2 & y_2 & z_2 & w_2 \\ x_3 & y_3 & z_3 & w_3 \\ x_4 & y_4 & z_4 & w_4 \end{vmatrix}.$$

Then

$$|\mathbf{xyzw}| = p_{xy}^{12} p_{zw}^{34} - p_{xy}^{13} p_{zw}^{24} + p_{xy}^{14} p_{zw}^{23} + p_{xy}^{23} p_{zw}^{14} - p_{xy}^{24} p_{zw}^{13} + p_{xy}^{34} p_{zw}^{12}.$$

Polynomial Identity

$$0 = |\mathbf{xyxw}| + |\mathbf{xyxz}| + |\mathbf{xyyw}| + |\mathbf{xyyz}| + \quad (1)$$
$$|\mathbf{zwxw}| + |\mathbf{zwxz}| + |\mathbf{zwyw}| + |\mathbf{zwyz}| \quad (2)$$

Polynomial Identity

$$\begin{aligned} 0 \quad = \quad & |\mathbf{xyxw}| + |\mathbf{xyxz}| + |\mathbf{xyyw}| + |\mathbf{xyyz}| + \qquad (1) \\ & |\mathbf{zwxw}| + |\mathbf{zwxz}| + |\mathbf{zwyw}| + |\mathbf{zwyz}| \qquad (2) \end{aligned}$$

Defining

$$p_{ij} = p_{xw}^{ij} + p_{xz}^{ij} + p_{yw}^{ij} + p_{yz}^{ij}, \quad 1 \le i < j \le 4, \qquad (3)$$

the four determinants of (1) can be **regrouped** as

$$p_{xy}^{12} p_{34} + p_{xy}^{13} p_{24} + p_{xy}^{14} p_{23} + p_{xy}^{23} p_{14} + p_{xy}^{24} p_{13} + p_{xy}^{34} p_{12}.$$

Polynomial Identity

After performing a similar grouping for (2), we get the the identity

$$\begin{aligned} 0 \quad = \quad & (p_{xy}^{12} + p_{zw}^{12})p_{34} + (p_{xy}^{13} + p_{zw}^{13})p_{24} + (p_{xy}^{14} + p_{zw}^{14})p_{23} + \\ & (p_{xy}^{23} + p_{zw}^{23})p_{14} + (p_{xy}^{24} + p_{zw}^{24})p_{13} + (p_{xy}^{34} + p_{zw}^{34})p_{12}. \quad (4) \end{aligned}$$

Polynomial Identity

After performing a similar grouping for (2), we get the the identity

$$\begin{aligned} 0 \quad = \quad & (p_{xy}^{12} + p_{zw}^{12})p_{34} + (p_{xy}^{13} + p_{zw}^{13})p_{24} + (p_{xy}^{14} + p_{zw}^{14})p_{23} + \\ & (p_{xy}^{23} + p_{zw}^{23})p_{14} + (p_{xy}^{24} + p_{zw}^{24})p_{13} + (p_{xy}^{34} + p_{zw}^{34})p_{12}. \quad (4) \end{aligned}$$

After a slight modification and a change of variables, (4) becomes

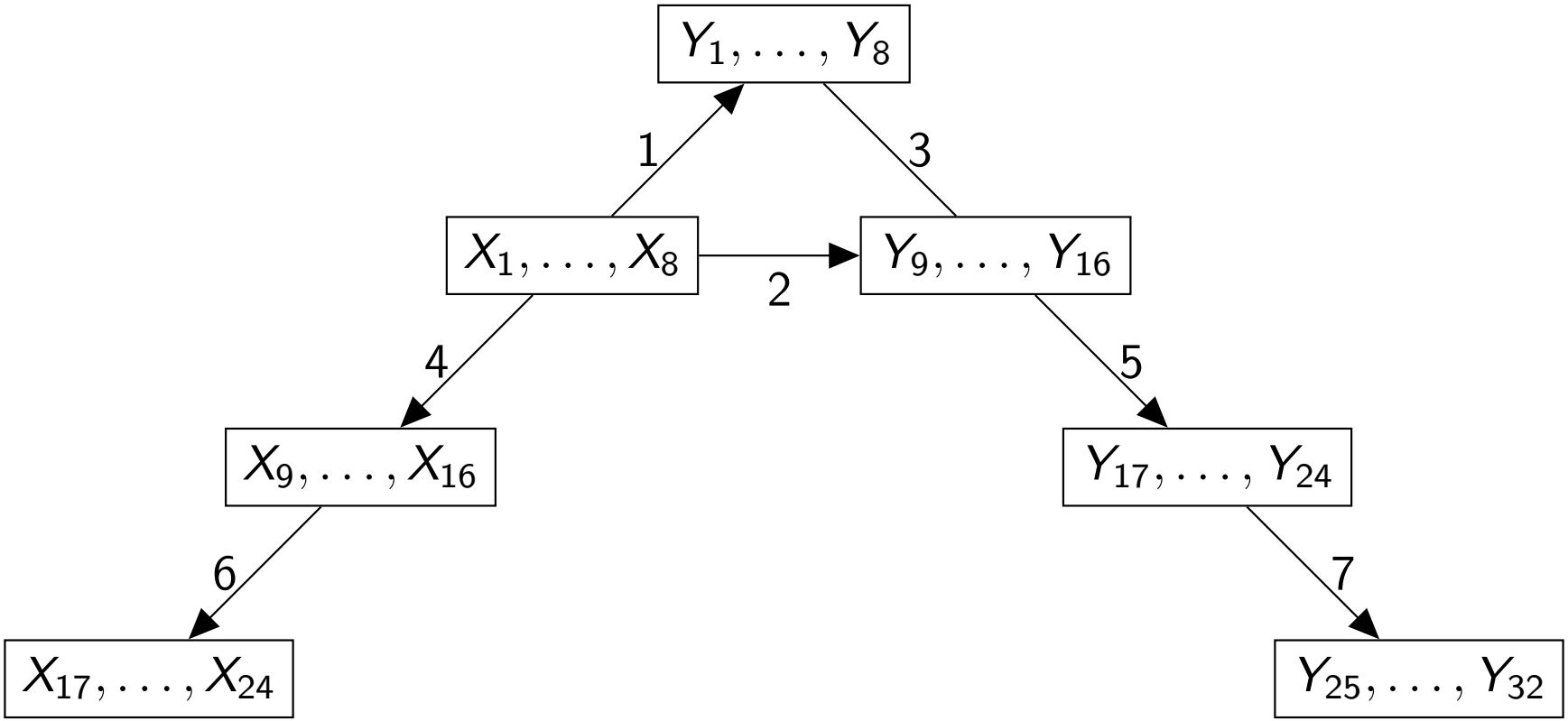
$$0 = A(X)A(Y) + f_1 f_2 + f_3 f_4 + f_5 f_6 + f_7 f_8 + f_9 f_{10}.$$

Our system

- A triangular system with 7 polynomials
- A chain of 7 oil-vinegar systems, each based on $f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8$with f_9 and f_10 attached.
- Lock polynomials based on $A(X)$ and $B(Y)$.

Chain of Oil-Vinegar Systems

The central map has input length 56: $(X_1, \ldots, X_{24}, Y_1, \ldots, Y_{32})$, and output length 74.



Security Analysis

Gröbner basis attacks: F_4 (Faugére 1999), F_5 (Faugére 2002, Barget et al 2005), XL algorithm (Courtois et al 2000), G^2V (Gao, Guan and Volny 2010).

Security Analysis

Gröbner basis attacks: F_4 (Faugére 1999), F_5 (Faugére 2002, Barget et al 2005), XL algorithm (Courtois et al 2000), G^2V (Gao, Guan and Volny 2010).

The total number of operations in k is about

$$O\left(\binom{n+d_{reg}}{n}^{\omega}\right),$$

where ω is the exponent in Gaussian reduction and d_{reg} is the degree of regularity of the ideal formed by the polynomials in the system.

Security Analysis

Attacks based on linear algebra:

- **Minrank attack** (Kipnis and Shamir 1999, Goubin and Courtois 2000). Each quadratic form is associated with a symmetric matrix and the rank of a matrix does not change under linear transforms. This attack explores the fact that some polynomials in the central map may have low rank. The complexity of this attack to our system is

$$q^{\lceil \frac{74}{56} \rceil 8d} = q^{16d}.$$

Security Analysis

Attacks based on linear algebra:

- **Minrank attack** (Kipnis and Shamir 1999, Goubin and Courtois 2000). Each quadratic form is associated with a symmetric matrix and the rank of a matrix does not change under linear transforms. This attack explores the fact that some polynomials in the central map may have low rank. The complexity of this attack to our system is

$$q^{\lceil \frac{74}{56} \rceil 8d} = q^{16d}.$$

- **Dual rank attack** (Yang and Chen 2004). While minrank succeeds when an equation has too few variables, the dual rank attack is effective when a variable appears in too few equations. The complexity of the attack is $(56d)^3 q^{6d}$.

Security Analysis

Attacks based on linear algebra:

- **Separation of oil and vinegar variables attack** (Kipnis and Shamir 1998, Kipnis, Patarin and Goubin 1999). The goal of this attack is to find the transformed oil space, so separate the oil and vinegar variables. The complexity is $20^4 q^{15d}$.

Security Analysis

Attacks based on linear algebra:

- **Separation of oil and vinegar variables attack** (Kipnis and Shamir 1998, Kipnis, Patarin and Goubin 1999). The goal of this attack is to find the transformed oil space, so separate the oil and vinegar variables. The complexity is $20^4 q^{15d}$.
- **Linearization equations attack.** (Patarin 1996, Ding et al 2007). Computations using Magma verify that there are no first order linearization equations. Ding et al broke MFE using second order linearization equations which comes from the associativity of determinant. Our system avoids determinants, so their method does not apply. But there still might be some other second order linearization equations!

Security Analysis

Claimed Security	Input [bits]	Output [bits]	Complexity		Key Size [kBytes]	
			F_5	Rank/UOV	Public	Private
2^{113}	896	1184	2^{114}	2^{113}	245	18
2^{212}	1792	2368	2^{213}	2^{212}	1907	70
2^{114}	1792	2368	2^{114}	2^{209}	490	36

Efficiency

System	Input [bits]	Output [bits]	Encryption	Decryption	
				Central Map	Total
MFE-1	768	960	52ms	2ms	2.7ms
Our System	896	1184	94ms	1.4ms	2.3ms

Open Questions

- Find "nontrivial" solutions to the following equation:

$$AB = CD + EF$$

where $A, B, C, D, E, F \in \mathbb{F}_2[x_1, \ldots, x_n, y_1, \ldots, y_n]$ have degree two.
MFE is based on solution with $n = 4$. Need solutions for $5 \leq n \leq 16$.

Open Questions

- Find "nontrivial" solutions to the following equation:

$$AB = CD + EF$$

where $A, B, C, D, E, F \in \mathbb{F}_2[x_1, \ldots, x_n, y_1, \ldots, y_n]$ have degree two.
MFE is based on solution with $n = 4$. Need solutions for $5 \leq n \leq 16$.

- Can we characterize all quadratic solutions to the equations

$$A(X)A(Y) = CD + EF$$

and

$$A(X)A(Y) = f_1 f_2 + f_3 f_4 + f_5 f_6 + f_7 f_8 + f_9 f_{10}?$$

Open Questions

- Using the tools from algebraic geometry and combinatorics, what other polynomial identities may be constructed to implement cryptosystems in the proposed framework?

Open Questions

- Using the tools from algebraic geometry and combinatorics, what other polynomial identities may be constructed to implement cryptosystems in the proposed framework?
- How to efficiently solve given systems of quadratic equations?
  G, Yinhua Guan, Frank Volny IV and Mingsheng Wang,
  A new algorithm for computing Grobner bases, which is both simpler and faster than F5.

Open Questions

- Using the tools from algebraic geometry and combinatorics, what other polynomial identities may be constructed to implement cryptosystems in the proposed framework?
- How to efficiently solve given systems of quadratic equations?
  G, Yinhua Guan, Frank Volny IV and Mingsheng Wang,
  A new algorithm for computing Grobner bases, which is both simpler and faster than F5.
- **Thank you!**