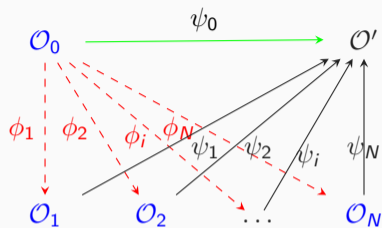


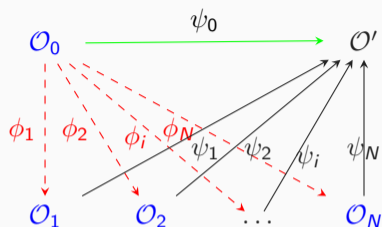
Optimization 2: Reduce signature size by using seeds



$\mathcal{P}(\mathcal{O}_0, \dots, \mathcal{O}_N, \phi_1, \dots, \phi_N)$	$\mathcal{V}(\mathcal{O}_0, \dots, \mathcal{O}_N)$
$\text{com} \leftarrow \mathcal{O}', \mathcal{O}'', \dots, \mathcal{O}^{(r)}$	$\xrightarrow{\text{com}}$
$\text{ch} \leftarrow_R \{0, N-1\}^r$	
$\text{resp} \leftarrow \psi_{\text{ch}_1}, \psi_{\text{ch}_2}, \dots, \psi_{\text{ch}_r}$	$\xrightarrow{\text{resp}}$
	$\mathcal{O}^{(i)} \stackrel{?}{=} \psi_{\text{ch}_i}(\mathcal{O}_{\text{ch}_i})$

- ▶ **The map ψ_0 is chosen at random** \Rightarrow one can include **only seed** in signature
 - ψ_0 can be reconstructed from the seed
- ▶ **Problem:** This works only for $\text{ch} = 0$, and probability of choosing challenge 0 is $1/N$
 - \Rightarrow not a big benefit in general

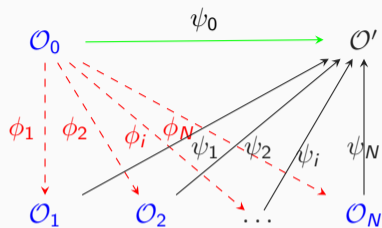
Optimization 2: Reduce signature size by using seeds



$\mathcal{P}(\mathcal{O}_0, \dots, \mathcal{O}_N, \phi_1, \dots, \phi_N)$	$\mathcal{V}(\mathcal{O}_0, \dots, \mathcal{O}_N)$
$\text{com} \leftarrow \mathcal{O}', \mathcal{O}'', \dots, \mathcal{O}^{(r)}$	$\xrightarrow{\text{com}}$
$\text{ch} \leftarrow_R \{0, N-1\}^r$	
$\text{resp} \leftarrow \psi_{\text{ch}_1}, \psi_{\text{ch}_2}, \dots, \psi_{\text{ch}_r}$	$\xrightarrow{\text{resp}}$
	$\mathcal{O}^{(i)} \stackrel{?}{=} \psi_{\text{ch}_i}(\mathcal{O}_{\text{ch}_i})$

- ▶ **The map ψ_0 is chosen at random** \Rightarrow one can include **only seed** in signature
 - ψ_0 can be reconstructed from the seed
- ▶ **Problem:** This works only for $\text{ch} = 0$, and probability of choosing challenge 0 is $1/N$
 - \Rightarrow not a big benefit in general
 - \Rightarrow **signature is not of constant size**

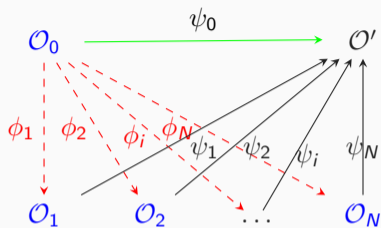
Optimization 2: Reduce signature size by using seeds



$\mathcal{P}(\mathcal{O}_0, \dots, \mathcal{O}_N, \phi_1, \dots, \phi_N)$	$\mathcal{V}(\mathcal{O}_0, \dots, \mathcal{O}_N)$
$\text{com} \leftarrow \mathcal{O}', \mathcal{O}'', \dots, \mathcal{O}^{(r)}$	$\xrightarrow{\text{com}}$
$\text{ch} \leftarrow_R \{0, N-1\}^r$	
$\text{resp} \leftarrow \psi_{\text{ch}_1}, \psi_{\text{ch}_2}, \dots, \psi_{\text{ch}_r}$	$\xrightarrow{\text{resp}}$
	$\mathcal{O}^{(i)} \stackrel{?}{=} \psi_{\text{ch}_i}(\mathcal{O}_{\text{ch}_i})$

- ▶ **The map ψ_0 is chosen at random** \Rightarrow one can include **only seed** in signature
 - ψ_0 can be reconstructed from the seed
- ▶ **Problem:** This works only for $\text{ch} = 0$, and probability of choosing challenge 0 is $1/N$
 - \Rightarrow not a big benefit in general
 - \Rightarrow **signature is not of constant size**
- ▶ **Idea:** Always have a fixed number M of 0 challenges

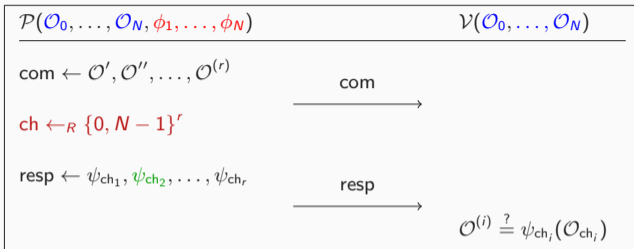
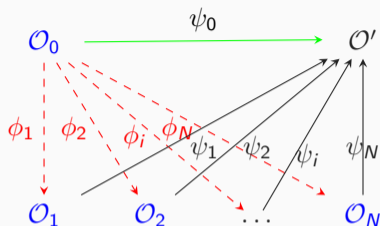
Optimization 2: Reduce signature size by using seeds



$\mathcal{P}(\mathcal{O}_0, \dots, \mathcal{O}_N, \phi_1, \dots, \phi_N)$	$\mathcal{V}(\mathcal{O}_0, \dots, \mathcal{O}_N)$
$\text{com} \leftarrow \mathcal{O}', \mathcal{O}'', \dots, \mathcal{O}^{(r)}$	$\xrightarrow{\text{com}}$
$\text{ch} \leftarrow_R \{0, N-1\}^r$	
$\text{resp} \leftarrow \psi_{\text{ch}_1}, \psi_{\text{ch}_2}, \dots, \psi_{\text{ch}_r}$	$\xrightarrow{\text{resp}}$
	$\mathcal{O}^{(i)} \stackrel{?}{=} \psi_{\text{ch}_i}(\mathcal{O}_{\text{ch}_i})$

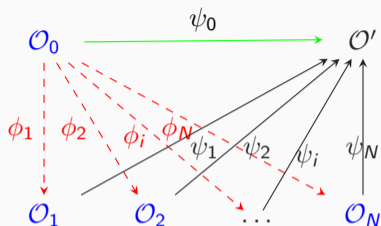
- ▶ **The map ψ_0 is chosen at random** \Rightarrow one can include **only seed** in signature
 - ψ_0 can be reconstructed from the seed
- ▶ **Problem:** This works only for $\text{ch} = 0$, and probability of choosing challenge 0 is $1/N$
 - \Rightarrow not a big benefit in general
 - \Rightarrow **signature is not of constant size**
- ▶ **Idea:** Always have a fixed number M of 0 challenges
 - **We need a special hash function** that always produces fixed weight outputs

Optimization 2: Reduce signature size by using seeds



- ▶ **The map ψ_0 is chosen at random** \Rightarrow one can include **only seed** in signature
 - ψ_0 can be reconstructed from the seed
- ▶ **Problem:** This works only for $\text{ch} = 0$, and probability of choosing challenge 0 is $1/N$
 - \Rightarrow not a big benefit in general
 - \Rightarrow **signature is not of constant size**
- ▶ **Idea:** Always have a fixed number M of 0 challenges
 - **We need a special hash function** that always produces fixed weight outputs
 - Always necessary to find the best trade-off

Optimization 2: Reduce signature size by using seeds



$\mathcal{P}(\mathcal{O}_0, \dots, \mathcal{O}_N, \phi_1, \dots, \phi_N)$	$\mathcal{V}(\mathcal{O}_0, \dots, \mathcal{O}_N)$
$\text{com} \leftarrow \mathcal{O}', \mathcal{O}'', \dots, \mathcal{O}^{(r)}$	$\xrightarrow{\text{com}}$
$\text{ch} \leftarrow_R \{0, N-1\}^r$	
$\text{resp} \leftarrow \psi_{\text{ch}_1}, \psi_{\text{ch}_2}, \dots, \psi_{\text{ch}_r}$	$\xrightarrow{\text{resp}}$
	$\mathcal{O}^{(i)} \stackrel{?}{=} \psi_{\text{ch}_i}(\mathcal{O}_{\text{ch}_i})$

- ▶ **The map ψ_0 is chosen at random** \Rightarrow one can include **only seed** in signature
 - ψ_0 can be reconstructed from the seed
- ▶ **Problem:** This works only for $\text{ch} = 0$, and probability of choosing challenge 0 is $1/N$
 - \Rightarrow not a big benefit in general
 - \Rightarrow **signature is not of constant size**
- ▶ **Idea:** Always have a fixed number M of 0 challenges
 - **We need a special hash function** that always produces fixed weight outputs
 - Always necessary to find the best trade-off

Generic hard equivalence problem $\text{EQ}(\mathcal{O}_0, \mathcal{O}_1)$:

Given \mathcal{O}_0 and \mathcal{O}_1 , find (if any) an isomorphism ϕ s.t. $\mathcal{O}_1 = \phi(\mathcal{O}_0)$

Generic hard equivalence problem $\text{EQ}(\mathcal{O}_0, \mathcal{O}_1)$:

Given \mathcal{O}_0 and \mathcal{O}_1 , find (if any) an isomorphism ϕ s.t. $\mathcal{O}_1 = \phi(\mathcal{O}_0)$

Generic hard equivalence problem $\text{EQ}(\mathcal{O}_0, \mathcal{O}_1)$:

Given \mathcal{O}_0 and \mathcal{O}_1 , find (if any) an isomorphism ϕ s.t. $\mathcal{O}_1 = \phi(\mathcal{O}_0)$

- ▶ **Isomorphism of polynomials** - Patarin's signature, 1998

Generic hard equivalence problem $\text{EQ}(\mathcal{O}_0, \mathcal{O}_1)$:

Given \mathcal{O}_0 and \mathcal{O}_1 , find (if any) an isomorphism ϕ s.t. $\mathcal{O}_1 = \phi(\mathcal{O}_0)$

- ▶ **Isomorphism of polynomials** - Patarin's signature, 1998
- ▶ **Quasigroup isotopy** - Identification scheme, [Denes 2001]

Generic hard equivalence problem $\text{EQ}(\mathcal{O}_0, \mathcal{O}_1)$:

Given \mathcal{O}_0 and \mathcal{O}_1 , find (if any) an isomorphism ϕ s.t. $\mathcal{O}_1 = \phi(\mathcal{O}_0)$

- ▶ **Isomorphism of polynomials** - Patarin's signature, 1998
- ▶ **Quasigroup isotopy** - Identification scheme, [Denes 2001]
- ▶ **Isogeny on elliptic curves** - SeaSign 2018, SqiSign 2020, [De Feo et al.]

Generic hard equivalence problem $\text{EQ}(\mathcal{O}_0, \mathcal{O}_1)$:

Given \mathcal{O}_0 and \mathcal{O}_1 , find (if any) an isomorphism ϕ s.t. $\mathcal{O}_1 = \phi(\mathcal{O}_0)$

- ▶ **Isomorphism of polynomials** - Patarin's signature, 1998
- ▶ **Quasigroup isotopy** - Identification scheme, [Denes 2001]
- ▶ **Isogeny on elliptic curves** - SeaSign 2018, SqiSign 2020, [De Feo et al.]
- ▶ **Code equivalence** - LESS - [Biasse et al. 2020], LESS-FM - [Barenghi et al. 2021]

Generic hard equivalence problem $\text{EQ}(\mathcal{O}_0, \mathcal{O}_1)$:

Given \mathcal{O}_0 and \mathcal{O}_1 , find (if any) an isomorphism ϕ s.t. $\mathcal{O}_1 = \phi(\mathcal{O}_0)$

- ▶ **Isomorphism of polynomials** - Patarin's signature, 1998
- ▶ **Quasigroup isotopy** - Identification scheme, [Denes 2001]
- ▶ **Isogeny on elliptic curves** - SeaSign 2018, SqiSign 2020, [De Feo et al.]
- ▶ **Code equivalence** - LESS - [Biasse et al. 2020], LESS-FM - [Barenghi et al. 2021]
- ▶ **Alternate trilinear form equivalence** - [Tang et al. 2022]

Generic hard equivalence problem $\text{EQ}(\mathcal{O}_0, \mathcal{O}_1)$:

Given \mathcal{O}_0 and \mathcal{O}_1 , find (if any) an isomorphism ϕ s.t. $\mathcal{O}_1 = \phi(\mathcal{O}_0)$

- ▶ **Isomorphism of polynomials** - Patarin's signature, 1998
- ▶ **Quasigroup isotopy** - Identification scheme, [Denes 2001]
- ▶ **Isogeny on elliptic curves** - SeaSign 2018, SqiSign 2020, [De Feo et al.]
- ▶ **Code equivalence** - LESS - [Biasse et al. 2020], LESS-FM - [Barenghi et al. 2021]
- ▶ **Alternate trilinear form equivalence** - [Tang et al. 2022]
- ▶ **Lattice isomorphism** - [Ducas–van Woerden 2022]

Generic hard equivalence problem $\text{EQ}(\mathcal{O}_0, \mathcal{O}_1)$:

Given \mathcal{O}_0 and \mathcal{O}_1 , find (if any) an isomorphism ϕ s.t. $\mathcal{O}_1 = \phi(\mathcal{O}_0)$

- ▶ **Isomorphism of polynomials** - Patarin's signature, 1998
- ▶ **Quasigroup isotopy** - Identification scheme, [Denes 2001]
- ▶ **Isogeny on elliptic curves** - SeaSign 2018, SqiSign 2020, [De Feo et al.]
- ▶ **Code equivalence** - LESS - [Biasse et al. 2020], LESS-FM - [Barenghi et al. 2021]
- ▶ **Alternate trilinear form equivalence** - [Tang et al. 2022]
- ▶ **Lattice isomorphism** - [Ducas–van Woerden 2022]
- ▶ **Matrix code equivalence** - [Reijnders–Samardjiska–Trimoska 2022]

Generic hard equivalence problem $\text{EQ}(\mathcal{O}_0, \mathcal{O}_1)$:

Given \mathcal{O}_0 and \mathcal{O}_1 , find (if any) an isomorphism ϕ s.t. $\mathcal{O}_1 = \phi(\mathcal{O}_0)$

- ▶ **Isomorphism of polynomials** - Patarin's signature, 1998
- ▶ **Quasigroup isotopy** - Identification scheme, [Denes 2001]
- ▶ **Isogeny on elliptic curves** - SeaSign 2018, SqiSign 2020, [De Feo et al.]
- ▶ **Code equivalence** - LESS - [Biasse et al. 2020], LESS-FM - [Barenghi et al. 2021]
- ▶ **Alternate trilinear form equivalence** - [Tang et al. 2022]
- ▶ **Lattice isomorphism** - [Ducas–van Woerden 2022]
- ▶ **Matrix code equivalence** - [Reijnders–Samardjiska–Trimoska 2022]
- ▶ ...

Equivalence problems for MEDS and ALTEQ

- ▶ MEDS is based on the following equivalence problem.
- ▶ **Matrix code** - a subspace of $\mathcal{M}_{m \times n}(\mathbb{F}_q)$ of dimension k endowed with **rank metric**.

- ▶ MEDS is based on the following equivalence problem.
- ▶ **Matrix code** - a subspace of $\mathcal{M}_{m \times n}(\mathbb{F}_q)$ of dimension k endowed with **rank metric**.

Matrix Code Equivalence (MCE) problem [Berger,2003]

$\text{MCE}(k, n, m, q, \mathcal{C}, \mathcal{D})$:

Input: Two k -dimensional matrix codes $\mathcal{C}, \mathcal{D} \subset \mathcal{M}_{m,n}(q)$

Question: Find – if any – $\mathbf{A} \in \text{GL}_m(q), \mathbf{B} \in \text{GL}_n(q)$ s.t. for all $\mathbf{C} \in \mathcal{C}$, it holds that

$$\mathbf{ACB} \in \mathcal{D}$$

- ▶ ALTEQ is based on the following equivalence problem.
- ▶ **Alternating trilinear form** - a map $\phi : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \rightarrow \mathbb{F}_q$ that
 - (1) is linear in each argument, and
 - (2) evaluates to 0 whenever two arguments are the same.

- ▶ ALTEQ is based on the following equivalence problem.
- ▶ **Alternating trilinear form** - a map $\phi : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \rightarrow \mathbb{F}_q$ that
 - (1) is linear in each argument, and
 - (2) evaluates to 0 whenever two arguments are the same.

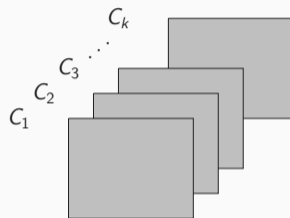
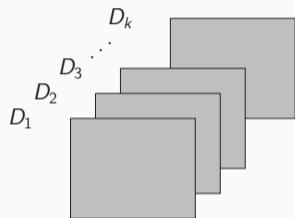
Alternating Trilinear Form Equivalence (ATFE) [Grochow-Qiao-Tang, 2021]

ALTEQ(n, q, ϕ, ψ):

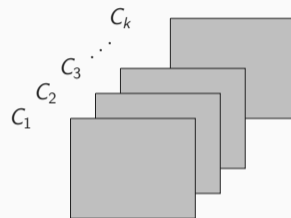
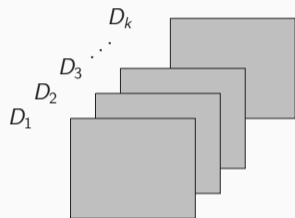
Input: Two alternating trilinear forms $\phi, \psi : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \rightarrow \mathbb{F}_q$.

Question: Find – if any – $\mathbf{A} \in \text{GL}_n(q)$ s.t. for any $u, v, w \in \mathbb{F}_q^n$, $\phi(u, v, w) = \psi(\mathbf{A}^t(u), \mathbf{A}^t(v), \mathbf{A}^t(w))$.

Matrix codes:



Matrix codes:

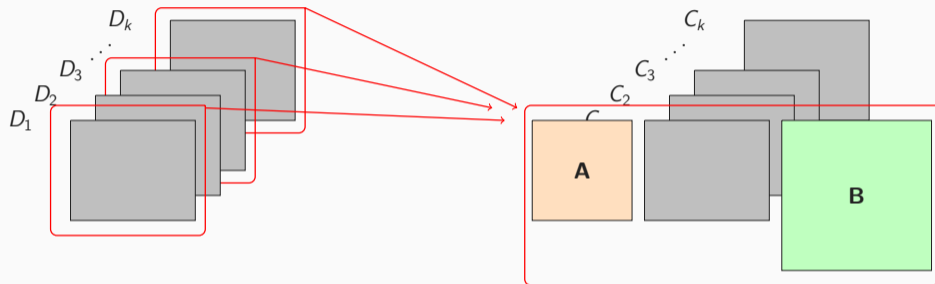


MCE:

- ▶ matrix codes of rectangular matrices

MCE and ATFE look very similar!

Matrix codes:



MCE:

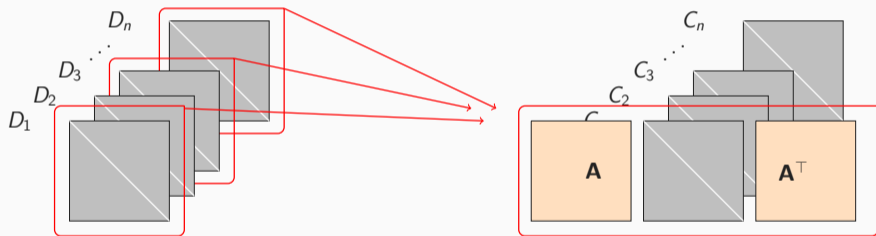
- ▶ matrix codes of rectangular matrices
- ▶ isometry (**A**, **B**)

MCE and ATFE look very similar!

- ▶ An alternating trilinear form is $\phi : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \rightarrow \mathbb{F}_q$.
- ▶ We can record ϕ as an $n \times n \times n$ 3-way array $\mathbf{C} = [c_{i,j,k}]$, where $c_{i,j,k} = \phi(e_i, e_j, e_k)$.
 - Note that $c_{i,j,k} = -c_{j,i,k} = -c_{k,j,i} = -c_{i,k,j} = c_{j,k,i} = c_{k,i,j}$.
- ▶ A 3-way array \mathbf{C} can also be represented as a matrix tuple (C_1, \dots, C_n) , $C_i \in \mathcal{M}_n(q)$.

MCE and ATFE look very similar!

- ▶ An alternating trilinear form is $\phi : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \rightarrow \mathbb{F}_q$.
- ▶ We can record ϕ as an $n \times n \times n$ 3-way array $\mathbf{C} = [c_{i,j,k}]$, where $c_{i,j,k} = \phi(e_i, e_j, e_k)$.
 - Note that $c_{i,j,k} = -c_{j,i,k} = -c_{k,j,i} = -c_{i,k,j} = c_{j,k,i} = c_{k,i,j}$.
- ▶ A 3-way array \mathbf{C} can also be represented as a matrix tuple (C_1, \dots, C_n) , $C_i \in \mathcal{M}_n(q)$.



ATFE:

- ▶ matrix codes with “symmetries in the three directions”.
- ▶ isometry $(\mathbf{A}, \mathbf{A}^\top)$ and \mathbf{A} on the third direction too

- ▶ The objects in MCE and ATFE are both 3-way arrays.
 - A 2-way array, $[c_{i,j}]$, is a matrix.
 - A 3-way array, $[c_{i,j,k}]$, is sometimes called a 3-tensor.
 - The 3-way arrays from ATFE are subject to certain structural constraints.

MCE and ATFE are polynomial-time equivalent

- ▶ The objects in MCE and ATFE are both 3-way arrays.
 - A 2-way array, $[c_{i,j}]$, is a matrix.
 - A 3-way array, $[c_{i,j,k}]$, is sometimes called a 3-tensor.
 - The 3-way arrays from ATFE are subject to certain structural constraints.
- ▶ The isomorphisms in MCE and ATFE are both invertible matrices.
 - $L, R \in \text{GL}_n(q)$ sends $C \in \mathcal{M}_n(q)$ to $L^t C R$.
 - $L, R, T = (t_{i,j}) \in \text{GL}_n(q)$ sends $(C_1, \dots, C_n) \in \mathcal{M}_n(q)^n$ to $(L^t C'_1 R, \dots, L^t C'_n R)$, where $C'_i = \sum_j t_{i,j} C_j$.
 - The isomorphism in ATFE imposes that $L = R = T$.

MCE and ATFE are polynomial-time equivalent

- ▶ The objects in MCE and ATFE are both 3-way arrays.
 - A 2-way array, $[c_{i,j}]$, is a matrix.
 - A 3-way array, $[c_{i,j,k}]$, is sometimes called a 3-tensor.
 - The 3-way arrays from ATFE are subject to certain structural constraints.
- ▶ The isomorphisms in MCE and ATFE are both invertible matrices.
 - $L, R \in \text{GL}_n(q)$ sends $C \in \mathcal{M}_n(q)$ to $L^t C R$.
 - $L, R, T = (t_{i,j}) \in \text{GL}_n(q)$ sends $(C_1, \dots, C_n) \in \mathcal{M}_n(q)^n$ to $(L^t C'_1 R, \dots, L^t C'_n R)$, where $C'_i = \sum_j t_{i,j} C_j$.
 - The isomorphism in ATFE imposes that $L = R = T$.

Theorem ([Grochow-Qiao-Tang, 2023])

MCE and ATFE are polynomial-time equivalent.

- ▶ Relations between isomorphism problems for some algebraic structures are studied in [Reijnders–Samardjiska–Trimoska, Grochow–Qiao–Tang, D’Alconzo, Couvreur–Debris–Alazard–Gaborit. . .]

A complexity class for isomorphism problems of algebraic structures

- ▶ Relations between isomorphism problems for some algebraic structures are studied in [Reijnders–Samardjiska–Trimoska, Grochow–Qiao–Tang, D’Alconzo, Couvreur–Debris–Alazard–Gaborit. . .]
- ▶ The complexity class **TI** was defined in [Grochow–Qiao], consisting of problems polynomial-time reducible to MCE.
 - MCE was called 3-Tensor Isomorphism in [Grochow–Qiao].
 - In analogy with the complexity class GI for Graph Isomorphism.
- ▶ MCE and ATFE are **TI**-complete.

A complexity class for isomorphism problems of algebraic structures

- ▶ Relations between isomorphism problems for some algebraic structures are studied in [Reijnders–Samardjiska–Trimoska, Grochow–Qiao–Tang, D’Alconzo, Couvreur–Debris–Alazard–Gaborit. . .]
- ▶ The complexity class **TI** was defined in [Grochow–Qiao], consisting of problems polynomial-time reducible to MCE.
 - MCE was called 3-Tensor Isomorphism in [Grochow–Qiao].
 - In analogy with the complexity class GI for Graph Isomorphism.
- ▶ MCE and ATFE are **TI**-complete.
- ▶ **TI**-complete problems include isomorphism problems for tensors, finite groups, (associative and Lie) algebras, (systems of) polynomials. . .

- ▶ TI-complete problems appear in computational group theory, multivariate cryptography, and quantum information.
 - Experiences from these areas suggest that TI-complete problems are difficult to solve in practice.

- ▶ **TI**-complete problems appear in computational group theory, multivariate cryptography, and quantum information.
 - Experiences from these areas suggest that **TI**-complete problems are difficult to solve in practice.
- ▶ Isomorphism problems for cubic forms and quadratic polynomial systems, as studied since 1996 [Patarin], are **TI**-complete.
 - Results from the study of polynomial isomorphism are valuable for MCE and ATFE.

- ▶ **TI**-complete problems appear in computational group theory, multivariate cryptography, and quantum information.
 - Experiences from these areas suggest that **TI**-complete problems are difficult to solve in practice.
- ▶ Isomorphism problems for cubic forms and quadratic polynomial systems, as studied since 1996 [Patarin], are **TI**-complete.
 - Results from the study of polynomial isomorphism are valuable for MCE and ATFE.
- ▶ Linear code monomial equivalence and graph isomorphism are in **TI** [Couvreur–Debris–Alazard–Gaborit, Grochow–Qiao].
 - Linear code monomial equivalence supports LESS.

Why use MCE and ATFE in post-quantum cryptography?

- ▶ A natural development of Shor's quantum algorithms for integer factorisation and discrete logarithm is the hidden subgroup problem framework.
- ▶ MCE and ATFE can be cast in this framework for general linear groups.

Why use MCE and ATFE in post-quantum cryptography?

- ▶ A natural development of Shor's quantum algorithms for integer factorisation and discrete logarithm is the hidden subgroup problem framework.
- ▶ MCE and ATFE can be cast in this framework for general linear groups.
- ▶ A strong negative evidence for the “standard technique” to work in this setting [Hallgren-Moore-Rötteler-Russell-Sen, 2010].

[Moore-Russell-Vazirani] . . . the strongest such insights we have about the limits of quantum algorithms.

Cryptanalysis for MCE and ATFE

- ▶ Consider 3-way arrays of size $n \times n \times n$ over \mathbb{F}_q under the action of (L, R, T) or $(T, T, T) \in \text{GL}_n(q) \times \text{GL}_n(q) \times \text{GL}_n(q)$.
- ▶ Brute-force algorithm: $q^{n^2} \cdot \text{poly}(n, \log q)$.
 - After fixing T , to recover L and R can be done in time $\text{poly}(n, \log q)$.

- ▶ Consider 3-way arrays of size $n \times n \times n$ over \mathbb{F}_q under the action of (L, R, T) or $(T, T, T) \in \text{GL}_n(q) \times \text{GL}_n(q) \times \text{GL}_n(q)$.
- ▶ Brute-force algorithm: $q^{n^2} \cdot \text{poly}(n, \log q)$.
 - After fixing T , to recover L and R can be done in time $\text{poly}(n, \log q)$.
- ▶ We will introduce three approaches.
 - Direct Gröbner basis attack.
 - Hybrid Gröbner basis: $q^n \cdot \text{poly}(n, \log q)$.
 - Utilising low-rank points (via birthday paradox and invariants).

- ▶ Let $C = [c_{i,j,k}]$ and $D = [d_{i,j,k}]$ be two $n \times n \times n$ 3-way arrays over \mathbb{F}_q .
- ▶ We view C as a matrix tuple (C_1, \dots, C_n) , $C_i \in \mathcal{M}_n(q)$.

- ▶ Let $C = [c_{i,j,k}]$ and $D = [d_{i,j,k}]$ be two $n \times n \times n$ 3-way arrays over \mathbb{F}_q .
- ▶ We view C as a matrix tuple (C_1, \dots, C_n) , $C_i \in \mathcal{M}_n(q)$.
- ▶ Recall that $L, R, T = (t_{i,j}) \in \text{GL}_n(q)$ sends $(C_1, \dots, C_n) \in \mathcal{M}_n(q)^n$ to $(L^t C'_1 R, \dots, L^t C'_n R)$, where $C'_i = \sum_j t_{i,j} C_j$.

- ▶ Let $C = [c_{i,j,k}]$ and $D = [d_{i,j,k}]$ be two $n \times n \times n$ 3-way arrays over \mathbb{F}_q .
- ▶ We view C as a matrix tuple (C_1, \dots, C_n) , $C_i \in \mathcal{M}_n(q)$.
- ▶ Recall that $L, R, T = (t_{i,j}) \in \text{GL}_n(q)$ sends $(C_1, \dots, C_n) \in \mathcal{M}_n(q)^n$ to $(L^t C'_1 R, \dots, L^t C'_n R)$, where $C'_i = \sum_j t_{i,j} C_j$.
- ▶ Viewing the entries of L , R and T as variables, the question is whether $(L^t C'_1 R, \dots, L^t C'_n R) = (D_1, \dots, D_n)$.
 - This amounts to n^3 cubic polynomials in $3n^2$ variables.

Cubic modelling $(L^t C'_1 R, \dots, L^t C'_n R) = (D_1, \dots, D_n)$ where $C'_i = \sum_j t_{ij} C_j$.

- ▶ This gives rise to n^3 cubic polynomials in $3n^2$ variables for MCE.
- ▶ And $\binom{n}{3}$ cubic polynomials in n^2 variables for ATFE.

Cubic modelling $(L^t C'_1 R, \dots, L^t C'_n R) = (D_1, \dots, D_n)$ where $C'_i = \sum_j t_{i,j} C_j$.

- ▶ This gives rise to n^3 cubic polynomials in $3n^2$ variables for MCE.
- ▶ And $\binom{n}{3}$ cubic polynomials in n^2 variables for ATFE.

Quadratic inverse modelling For ATFE, let $T' = [t'_{i,j}]$. Then set

$(T^t C_1 T, \dots, T^t C_n T) = (D'_1, \dots, D'_n)$ where $D'_i = \sum_j t'_{i,j} D_j$, and $TT' = I_n$.

- ▶ This is by [Bouillaguet-Faugère-Fouque-Perret, 2010].
- ▶ $n \cdot \binom{n}{2} + n^2$ quadratic polynomials in $2n^2$ variables.

Cubic modelling $(L^t C'_1 R, \dots, L^t C'_n R) = (D_1, \dots, D_n)$ where $C'_i = \sum_j t_{i,j} C_j$.

- ▶ This gives rise to n^3 cubic polynomials in $3n^2$ variables for MCE.
- ▶ And $\binom{n}{3}$ cubic polynomials in n^2 variables for ATFE.

Quadratic inverse modelling For ATFE, let $T' = [t'_{i,j}]$. Then set

$(T^t C_1 T, \dots, T^t C_n T) = (D'_1, \dots, D'_n)$ where $D'_i = \sum_j t'_{i,j} D_j$, and $TT' = I_n$.

- ▶ This is by [Bouillaguet-Faugère-Fouque-Perret, 2010].
- ▶ $n \cdot \binom{n}{2} + n^2$ quadratic polynomials in $2n^2$ variables.

Quadratic dual modelling Use the dual space of \mathcal{D} to express that $L^t C_i R \in \mathcal{D}$.

- ▶ This is by [Chou-Niederhagen-Persichetti-Randrianarisoa-Reijnders-Samardjiska-Trimoska].
- ▶ This gives rise to $n \cdot (n^2 - n)$ homogeneous quadratic polynomials in $2n^2$ variables for MCE.
- ▶ And $n \cdot \left(\binom{n}{2} - n\right)$ quadratic polynomials in n^2 variables for ATFE.
- ▶ **Note** that some syzygies arise, complicating the analysis [MEDS spec].

- ▶ We set up $n \times n$ variable matrices L and R for MCE (or T and T' for ATFE).
- ▶ In [Faugère-Perret, 2006], it was discovered that Gröbner basis runs in polynomial time, provided that one (or two) rows of L are known.

- ▶ We set up $n \times n$ variable matrices L and R for MCE (or T and T' for ATFE).
- ▶ In [Faugère-Perret, 2006], it was discovered that Gröbner basis runs in polynomial time, provided that one (or two) rows of L are known.
- ▶ For ATFE, knowing one row of T is enough, leading to a $q^n \cdot \text{poly}(n, \log q)$ -time algorithm.
- ▶ For MCE, knowing two rows of L is enough, leading to an $q^{2n} \cdot \text{poly}(n, \log q)$ -time algorithm.

- ▶ We set up $n \times n$ variable matrices L and R for MCE (or T and T' for ATFE).
- ▶ In [Faugère-Perret, 2006], it was discovered that Gröbner basis runs in polynomial time, provided that one (or two) rows of L are known.
- ▶ For ATFE, knowing one row of T is enough, leading to a $q^n \cdot \text{poly}(n, \log q)$ -time algorithm.
- ▶ For MCE, knowing two rows of L is enough, leading to an $q^{2n} \cdot \text{poly}(n, \log q)$ -time algorithm.
- ▶ Further observations from [Beullens, 2023]:
 - Knowing one row of T up to scalar is enough.
 - For low-rank points, the kernel information can be incorporated.

- ▶ Let $\phi : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \rightarrow \mathbb{F}_q$ be an alternating trilinear form.
- ▶ For $u \in \mathbb{F}_q^n$, let $\phi_u : \mathbb{F}_q^n \times \mathbb{F}_q^n \rightarrow \mathbb{F}_q$ by $\phi_u(v, w) = \phi(u, v, w)$.
- ▶ An isomorphism invariant for u : $r = \text{Rank}(\phi_u)$.

- ▶ Let $\phi : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \rightarrow \mathbb{F}_q$ be an alternating trilinear form.
- ▶ For $u \in \mathbb{F}_q^n$, let $\phi_u : \mathbb{F}_q^n \times \mathbb{F}_q^n \rightarrow \mathbb{F}_q$ by $\phi_u(v, w) = \phi(u, v, w)$.
- ▶ An isomorphism invariant for u : $r = \text{Rank}(\phi_u)$.
- ▶ Algorithms based on birthday paradox and hybrid Gröbner basis [Bouillaguet-Fouque-Véber, 2013; Beullens, 2023].
 - Suppose there exist $\approx q^k$ -many rank- r points for a random ϕ .
 - (1) Sample $q^{k/2}$ -many rank- r points for ϕ and ψ , respectively.
 - (2) For every pair, use hybrid Gröbner basis to find a “matched” pair.
 - Algorithm cost: $O(q^{k/2} \cdot \text{samp-cost} + q^k \cdot \text{gb-cost})$.

- ▶ Let $\phi : \mathbb{F}_q^n \times \mathbb{F}_q^n \times \mathbb{F}_q^n \rightarrow \mathbb{F}_q$ be an alternating trilinear form.
- ▶ For $u \in \mathbb{F}_q^n$, let $\phi_u : \mathbb{F}_q^n \times \mathbb{F}_q^n \rightarrow \mathbb{F}_q$ by $\phi_u(v, w) = \phi(u, v, w)$.
- ▶ An isomorphism invariant for u : $r = \text{Rank}(\phi_u)$.
- ▶ Algorithms based on birthday paradox and hybrid Gröbner basis [Bouillaguet-Fouque-Véber, 2013; Beullens, 2023].
 - Suppose there exist $\approx q^k$ -many rank- r points for a random ϕ .
 - (1) Sample $q^{k/2}$ -many rank- r points for ϕ and ψ , respectively.
 - (2) For every pair, use hybrid Gröbner basis to find a “matched” pair.
 - Algorithm cost: $O(q^{k/2} \cdot \text{samp-cost} + q^k \cdot \text{gb-cost})$.
- ▶ Sampling step: min-rank or graph-walking [Beullens, 2023]

- ▶ Algorithms based on distinguishing isomorphism invariants with low-rank points [Bouillaguet-Fouque-Véber, 2013; Beullens, 2023].
 - Suppose there exist $\approx q^k$ -many rank- r points for a random ϕ .
 - Suppose there exist distinguishing isomorphism invariants associated with such points.

- ▶ Algorithms based on distinguishing isomorphism invariants with low-rank points [Bouillaguet-Fouque-Véber, 2013; Beullens, 2023].
 - Suppose there exist $\approx q^k$ -many rank- r points for a random ϕ .
 - Suppose there exist distinguishing isomorphism invariants associated with such points.
- (1) Sample $q^{k/2}$ -many rank- r points for ϕ and ψ , respectively.
- (2) For every point, compute the isomorphism invariant.
- (3) By birthday paradox, there exists a pair of points of the same invariant. Use hybrid Gröbner basis to complete.

- ▶ Algorithms based on distinguishing isomorphism invariants with low-rank points [Bouillaguet-Fouque-Véber, 2013; Beullens, 2023].
 - Suppose there exist $\approx q^k$ -many rank- r points for a random ϕ .
 - Suppose there exist distinguishing isomorphism invariants associated with such points.
- (1) Sample $q^{k/2}$ -many rank- r points for ϕ and ψ , respectively.
- (2) For every point, compute the isomorphism invariant.
- (3) By birthday paradox, there exists a pair of points of the same invariant. Use hybrid Gröbner basis to complete.
 - Algorithm cost: $O(q^{k/2} \cdot (\text{samp-cost} + \text{inv-cost}) + \text{gb-cost})$.

- ▶ Algorithms based on distinguishing isomorphism invariants with low-rank points [Bouillaguet-Fouque-Véber, 2013; Beullens, 2023].
 - Suppose there exist $\approx q^k$ -many rank- r points for a random ϕ .
 - Suppose there exist distinguishing isomorphism invariants associated with such points.
 - (1) Sample $q^{k/2}$ -many rank- r points for ϕ and ψ , respectively.
 - (2) For every point, compute the isomorphism invariant.
 - (3) By birthday paradox, there exists a pair of points of the same invariant. Use hybrid Gröbner basis to complete.
 - Algorithm cost: $O(q^{k/2} \cdot (\text{samp-cost} + \text{inv-cost}) + \text{gb-cost})$.
- ▶ Distinguishing isomorphism invariant candidates: ranks of the neighbours of low-rank points, and more [Narayanan-Qiao-Tang].

Parameters and performances of MEDS **and** ALTEQ

Parameters and performance of MEDS

Level	param. set	public key size (KB)	signature size (KB)	key gen (ms)	sign (ms)	verify (ms)
I	MEDS-9923	9.9	9.9	1	272	271
	MEDS-13220	13.2	13	1.3	46.7	46
III	MEDS-41711	41.7	41	5.1	779	762
	MEDS-69497	55.6	54.7	6.7	203.8	200.4

Table: An overview of the parameters and performance of MEDS.

Optimizations:

- ▶ **Standard:** Multiple Public Keys + Fixed-Weight Challenge Strings + Seed tree
- ▶ **New:** Public Key Compression
 - generate public key partially from seed \Rightarrow signature size reduction
 - **Work in progress:** use similar idea during signing

Parameters and performance of ALTEQ

Level	mode	public key size (KB)	signature size (KB)	key gen (ms)	sign (ms)	verify (ms)
I	Balanced	8	16	0.093	0.629	0.496
	ShortSig	512	10	1.902	0.194	0.092
III	Balanced	32	48	0.582	6.986	6.483
	ShortSig	1024	24	5.152	1.705	1.304

Table: An overview of the parameters and performance of ALTEQ.

Optimizations:

- ▶ **Standard:** Multiple Public Keys + Fixed-Weight Challenge Strings (+ Seed tree)
- ▶ **New:** Invertible matrix decomposition
 - Represent an invertible matrix as a product of column matrices for faster signing and verification

- ▶ Digital signature based on equivalence problems: design and optimisations
- ▶ Matrix code equivalence (MCE) and alternating trilinear form equivalence (ATFE)
- ▶ Algorithms for MCE and ATFE
- ▶ MEDS and ALTEQ: parameters and performances

Thank you for listening!