

Ciphertext-Ciphertext Matrix Multiplication: Fast for Large Matrices

Jai Hyun Park

jaihyunp@gmail.com



FHE.org – June 5, 2025

Summary

- Fast ciphertext-ciphertext matrix multiplication (CCMM)
 - 85.2 s for CCMM of 4096×4096 matrices in a single thread CPU
 - How? Reduce CCMM to plaintext matrix multiplications
- Fast ciphertext matrix transpose (CMT)
 - 0.76 s for CMT of a 2048×2048 matrices in a single thread CPU
- Lightweight CCMM and CMT algorithms with smaller key sizes



Matrix Multiplication

- Matrix multiplication is central in high-performance computing
 - highly optimized libraries for basic linear algebra subprograms (BLAS)
 - Can be 10-30x faster than a naïve implementation for large matrices



Matrix Multiplication

- Matrix multiplication is central in high-performance computing
 - highly optimized libraries for basic linear algebra subprograms (BLAS)
 - Can be 10–30x faster than a naïve implementation for large matrices

What about matrix multiplication on encrypted data?

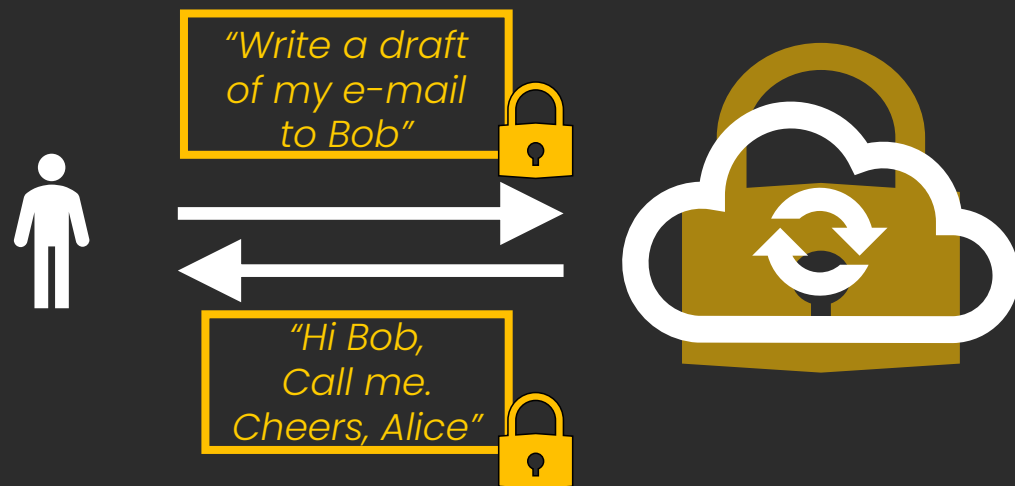


Matrix Multiplication

- Matrix multiplication is central in high-performance computing
 - highly optimized libraries for basic linear algebra subprograms (BLAS)
 - Can be 10-30x faster than a naïve implementation for large matrices

What about matrix multiplication on encrypted data?

Privacy-preserving machine learning as a service

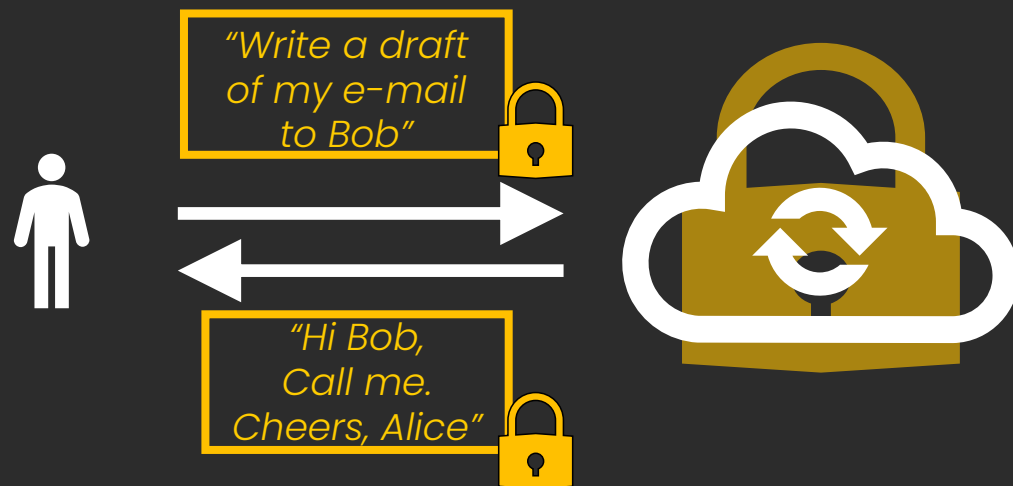


Matrix Multiplication

- Matrix multiplication is central in high-performance computing
 - highly optimized libraries for basic linear algebra subprograms (BLAS)
 - Can be 10-30x faster than a naïve implementation for large matrices

What about matrix multiplication on encrypted data?

Privacy-preserving machine learning as a service



- PPMM: plaintext-plaintext matrix multiplication
- PCMM: plaintext-ciphertext matrix multiplication
- CCMM: ciphertext-ciphertext matrix multiplication
- PCMMs and CCMMs with diverse dimensions
 - e.g., PCMM of dimension 128 ~ 16384 for GPT-3.5



PPMM vs. PCMM vs. CCMM



PPMM vs. PCMM vs. CCMM

- PPMM – BLAS libraries
 - highly optimized open libraries
 - Can be 30x faster than a naïve implementation

For matrix dimension 2^{12} :

PPMM
(OpenBLAS)

1.47 seconds



PPMM vs. PCMM vs. CCMM

- PPMM – BLAS libraries
 - highly optimized open libraries
 - Can be 30x faster than a naïve implementation
- PCMM – BCHPS'24
 - Reduction from PCMM to PPMM
 - Optimizations with shared-α, truncation, and others

For matrix dimension 2^{12} :

PPMM
(OpenBLAS)

1.47 seconds

PCMM
(BCHPS'24)

17.1 seconds



PPMM vs. PCMM vs. CCMM

- PPMM – BLAS libraries
 - highly optimized open libraries
 - Can be 30x faster than a naïve implementation
- PCMM – BCHPS'24
 - Reduction from PCMM to PPMM
 - Optimizations with shared-α, truncation, and others
- CCMM – JKLS'18
 - cubic bit complexity
 - 0.6 seconds for matrix dimension 64

For matrix dimension 2^{12} :

PPMM
(OpenBLAS)

1.47 seconds

PCMM
(BCHPS'24)

17.1 seconds



PPMM vs. PCMM vs. CCMM

- PPMM – BLAS libraries
 - highly optimized open libraries
 - Can be 30x faster than a naïve implementation
- PCMM – BCHPS'24
 - Reduction from PCMM to PPMM
 - Optimizations with shared-α, truncation, and others
- CCMM – JKLS'18
 - cubic bit complexity
 - 0.6 seconds for matrix dimension 64

For matrix dimension 2^{12} :

PPMM
(OpenBLAS)

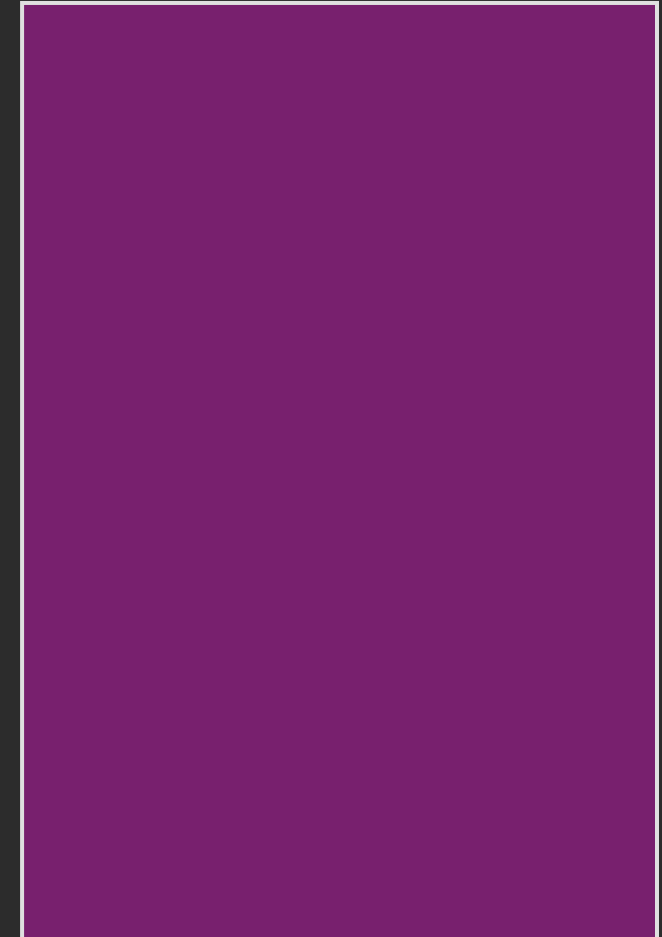
1.47 seconds

PCMM
(BCHPS'24)

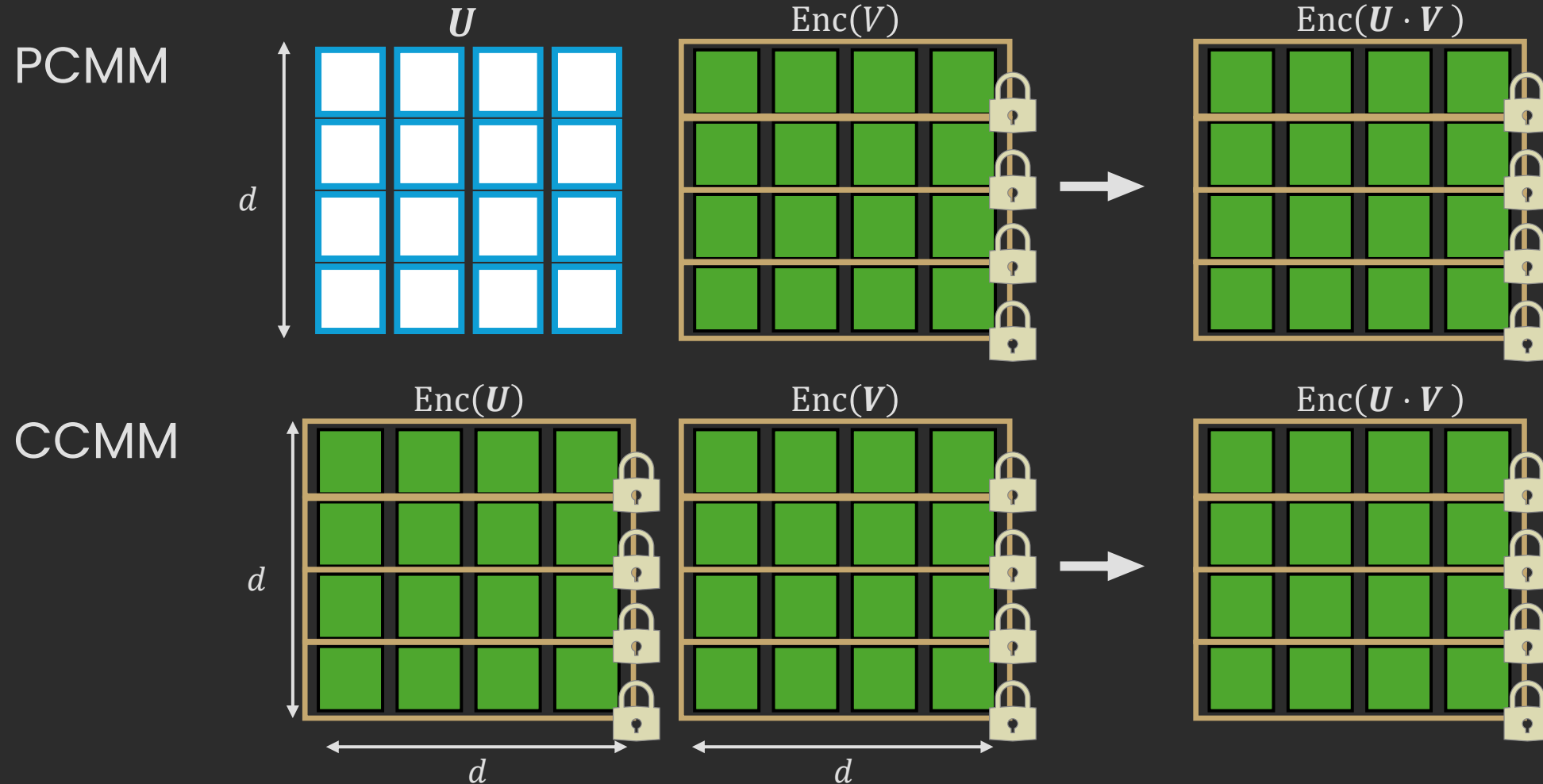
17.1 seconds

CCMM
(JKLS'18, estimated)

> 19 hours



PCMM and CCMM



PCMM/CCMM with CKKS

- CKKS
 - Plaintext: vector of real numbers
 - Native operations: // add, // mult, and rotate.



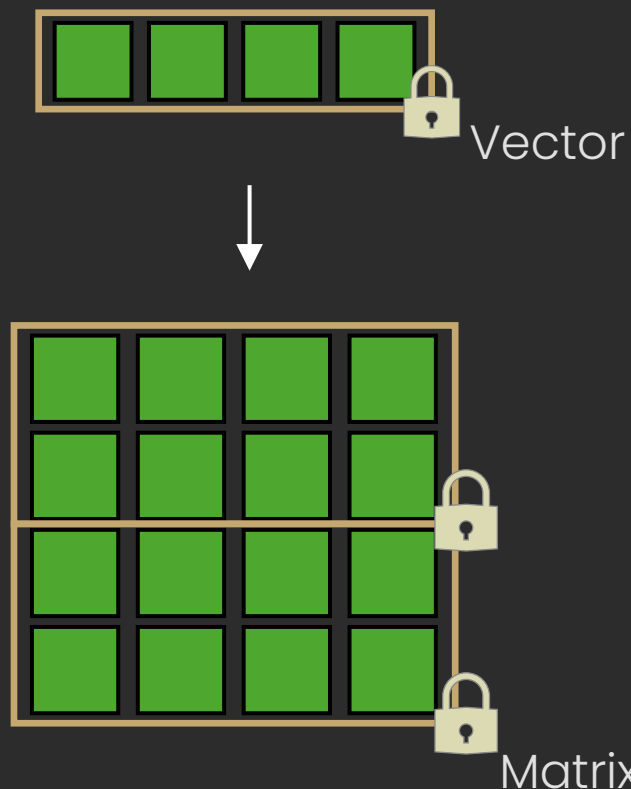
PCMM/CCMM with CKKS

- CKKS
 - Plaintext: vector of real numbers
 - Native operations: // add, // mult, and rotate.



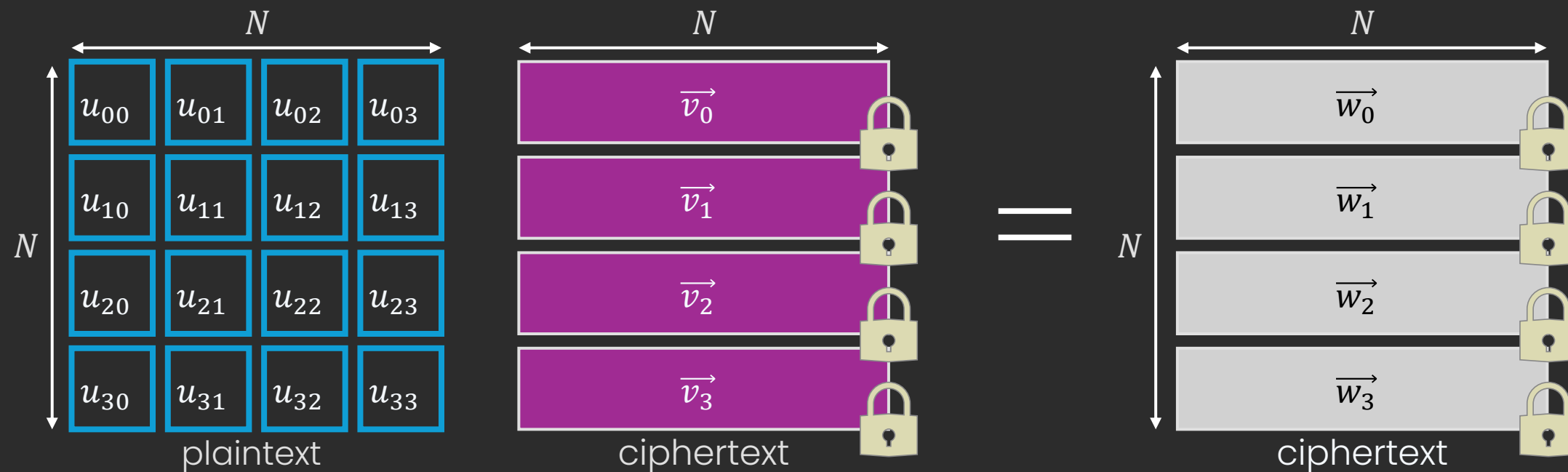
PCMM/CCMM with CKKS

- CKKS
 - Plaintext: vector of real numbers
 - Native operations: // add, // mult, and rotate.
- With the native operations, PCMM requires lots of rotates.
 - For example, [JKLS18] has a cubic bit complexity, but is orders of magnitude slower than PPMM.



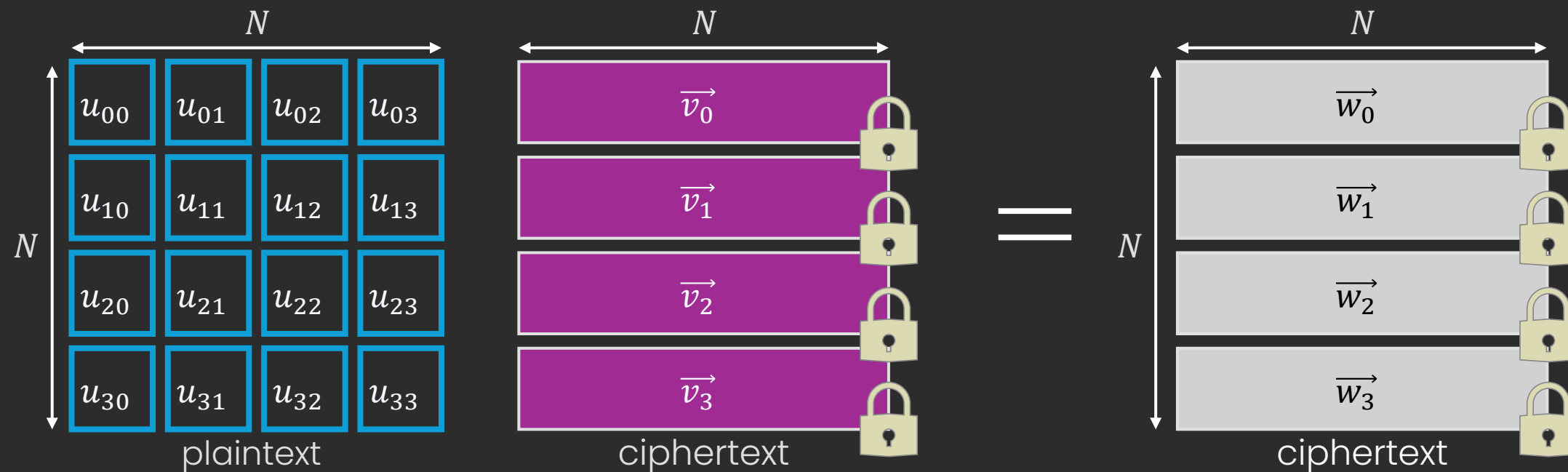
Why BLAS?

- PCMM when the matrix dimension $d = N$



Why BLAS?

- PCMM when the matrix dimension $d = N$

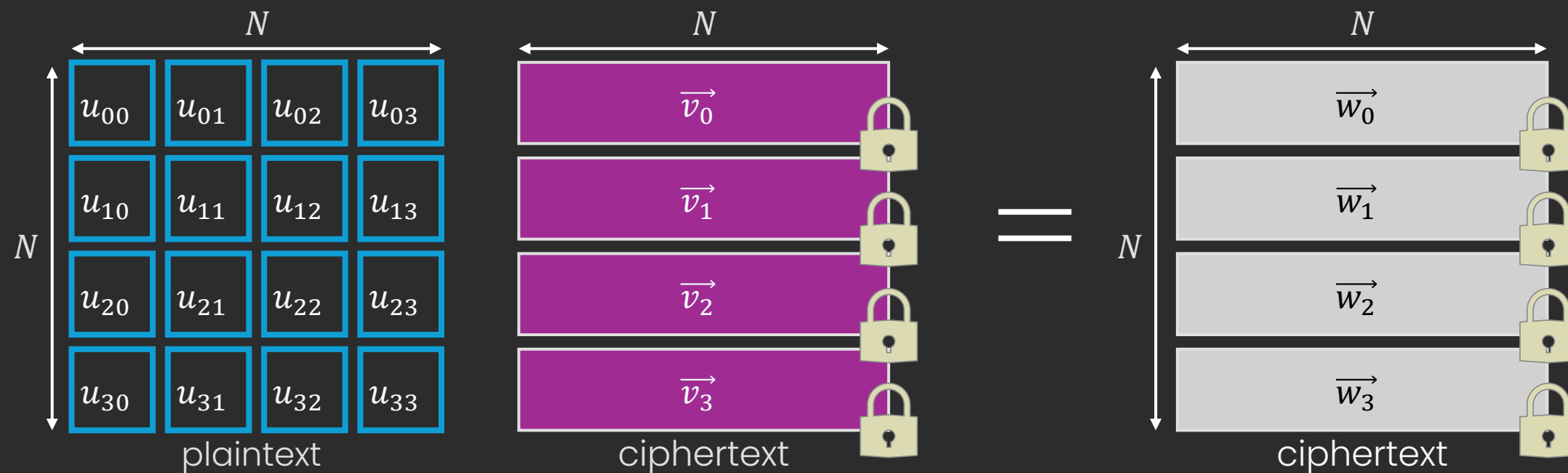


- Linear algebra: $\vec{w}_j = u_{j0}\vec{v}_0 + u_{j1}\vec{v}_1 + u_{j2}\vec{v}_2 + u_{j3}\vec{v}_3 = \sum_i u_{ji}\vec{v}_i$
- Linear HE : $Enc(\vec{w}_j) = \sum_i u_{ji}Enc(\vec{v}_i)$



Why BLAS?

- PCMM when the matrix dimension $d = N$



- Linear algebra: $\vec{w}_j = u_{j0}\vec{v}_0 + u_{j1}\vec{v}_1 + u_{j2}\vec{v}_2 + u_{j3}\vec{v}_3 = \sum_i u_{ji}\vec{v}_i$
- Linear HE : $Enc(\vec{w}_j) = \sum_i u_{ji}Enc(\vec{v}_i)$

> 2500 seconds for $N = 2^{13}$



How to utilize BLAS?

Q. How to utilize PPMM BLAS libraries?



How to utilize BLAS?

Q. How to utilize PPMM BLAS libraries?

A. Reduction from PCMM/CCMM to PPMM



How to utilize BLAS?

Q. How to utilize PPMM BLAS libraries?

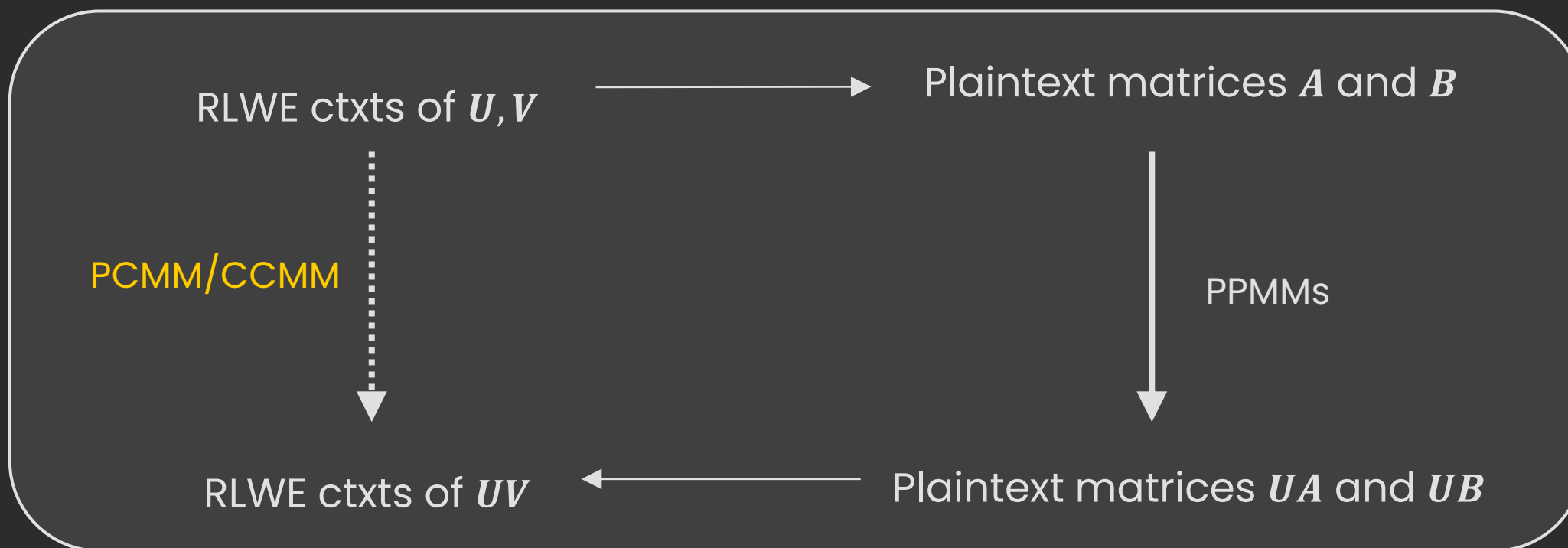
A. Reduction from PCMM/CCMM to PPMM



How to utilize BLAS?

Q. How to utilize PPMM BLAS libraries?

A. Reduction from PCMM/CCMM to PPMM



RLWE-based Encryption of Matrices

- In the ring $\mathbb{Z}_q[X]/(X^N + 1)$, an RLWE ciphertext $(a, b = -a \cdot s + m)$ is:



RLWE-based Encryption of Matrices

- In the ring $\mathbb{Z}_Q[X]/(X^N + 1)$, an RLWE ciphertext $(a, b = -a \cdot s + m)$ is:

$$[a_0 \ a_1 \ \cdots \ a_{N-1}] \begin{bmatrix} s_0 & s_1 & \cdots & s_{N-1} \\ -s_{N-1} & s_0 & \cdots & s_{N-2} \\ \vdots & \vdots & \ddots & \vdots \\ -s_1 & -s_2 & \cdots & s_0 \end{bmatrix} + [b_0 \ b_1 \ \cdots \ b_{N-1}] = [m_0 \ m_1 \ \cdots \ m_{N-1}]$$

✓ a_i, b_i, s_i, m_i are coeffs of a, b, s, m



RLWE-based Encryption of Matrices

- In the ring $\mathbb{Z}_Q[X]/(X^N + 1)$, an RLWE ciphertext $(a, b = -a \cdot s + m)$ is:

$$\begin{bmatrix} a_0 & a_1 & \cdots & a_{N-1} \end{bmatrix} \cdot \begin{bmatrix} s_0 & s_1 & \cdots & s_{N-1} \\ -s_{N-1} & s_0 & \cdots & s_{N-2} \\ \vdots & \vdots & \ddots & \vdots \\ -s_1 & -s_2 & \cdots & s_0 \end{bmatrix} + \begin{bmatrix} b_0 & b_1 & \cdots & b_{N-1} \end{bmatrix} = \begin{bmatrix} m_0 & m_1 & \cdots & m_{N-1} \end{bmatrix}$$

✓ a_i, b_i, s_i, m_i are coeffs of a, b, s, m

Toep(s)



RLWE-based Encryption of Matrices

- In the ring $\mathbb{Z}_Q[X]/(X^N + 1)$, an RLWE ciphertext $(a, b = -a \cdot s + m)$ is:

$$\begin{bmatrix} a_0 & a_1 & \cdots & a_{N-1} \end{bmatrix} \begin{bmatrix} s_0 & s_1 & \cdots & s_{N-1} \\ -s_{N-1} & s_0 & \cdots & s_{N-2} \\ \vdots & \vdots & \ddots & \vdots \\ -s_1 & -s_2 & \cdots & s_0 \end{bmatrix} + \begin{bmatrix} b_0 & b_1 & \cdots & b_{N-1} \end{bmatrix} = \begin{bmatrix} m_0 & m_1 & \cdots & m_{N-1} \end{bmatrix}$$

✓ a_i, b_i, s_i, m_i are coeffs of a, b, s, m

Toep(s)

- N RLWE ciphertexts are:

$$\begin{bmatrix} \text{---} & a_1^t & \text{---} \\ \text{---} & a_2^t & \text{---} \\ & \vdots & \\ \text{---} & a_N^t & \text{---} \end{bmatrix} \begin{bmatrix} s_0 & s_1 & \cdots & s_{N-1} \\ -s_{N-1} & s_0 & \cdots & s_{N-2} \\ \vdots & \vdots & \ddots & \vdots \\ -s_1 & -s_2 & \cdots & s_0 \end{bmatrix} + \begin{bmatrix} \text{---} & b_1^t & \text{---} \\ \text{---} & b_2^t & \text{---} \\ & \vdots & \\ \text{---} & b_N^t & \text{---} \end{bmatrix} = \begin{bmatrix} \text{---} & m_1^t & \text{---} \\ \text{---} & m_2^t & \text{---} \\ & \vdots & \\ \text{---} & m_N^t & \text{---} \end{bmatrix}$$



RLWE-based Encryption of Matrices

- In the ring $\mathbb{Z}_Q[X]/(X^N + 1)$, an RLWE ciphertext $(a, b = -a \cdot s + m)$ is:

$$\begin{bmatrix} a_0 & a_1 & \cdots & a_{N-1} \end{bmatrix} \begin{bmatrix} s_0 & s_1 & \cdots & s_{N-1} \\ -s_{N-1} & s_0 & \cdots & s_{N-2} \\ \vdots & \vdots & \ddots & \vdots \\ -s_1 & -s_2 & \cdots & s_0 \end{bmatrix} + \begin{bmatrix} b_0 & b_1 & \cdots & b_{N-1} \end{bmatrix} = \begin{bmatrix} m_0 & m_1 & \cdots & m_{N-1} \end{bmatrix}$$

$\checkmark a_i, b_i, s_i, m_i$ are coeffs of a, b, s, m

Toep(s)

- N RLWE ciphertexts are:

$$\begin{bmatrix} \text{---} & a_1^t & \text{---} \\ \text{---} & a_2^t & \text{---} \\ & \mathbf{A} & \\ \text{---} & a_N^t & \text{---} \end{bmatrix} \begin{bmatrix} s_0 & s_1 & \cdots & s_{N-1} \\ -s_{N-1} & s_0 & \cdots & s_{N-2} \\ \vdots & \vdots & \ddots & \vdots \\ -s_1 & -s_2 & \cdots & s_0 \end{bmatrix} + \begin{bmatrix} \text{---} & b_1^t & \text{---} \\ \text{---} & b_2^t & \text{---} \\ & \mathbf{B} & \\ \text{---} & b_N^t & \text{---} \end{bmatrix} = \begin{bmatrix} \text{---} & m_1^t & \text{---} \\ \text{---} & m_2^t & \text{---} \\ & \mathbf{M} & \\ \text{---} & m_N^t & \text{---} \end{bmatrix}$$



$\text{PCMM} \leq \text{PPMMs}$ (BCHPS'24, LZ'22)

$$\begin{matrix} \text{Matrix } A & \text{Matrix } \text{Toep}(s) & + & \text{Matrix } B & = & \text{Matrix } V \end{matrix}$$

Matrix V is labeled with dimensions N by N .



PCMM \leq PPMMs (BCHPS'24, LZ'22)

$$U \left[A \text{Toep}(s) + B \right] = U V$$

The diagram illustrates the PCMM operation. On the left, a blue matrix U is multiplied by the sum of two purple matrices, A and B , each multiplied by a Toep(s) operation. The result is a blue matrix U multiplied by a green matrix V . The dimensions of matrices A , B , and V are indicated as N .



PCMM \leq PPMMs (BCHPS'24, LZ'22)

$$\begin{array}{c}
 \begin{array}{c} \text{Grid of } U \end{array} \left[\begin{array}{c} \text{Grid of } A \\ \text{Toep}(s) \end{array} + \begin{array}{c} \text{Grid of } B \end{array} \right] = \begin{array}{c} \text{Grid of } U \end{array} \begin{array}{c} \text{Grid of } V \end{array} \\
 \begin{array}{c} \text{Grid of } UA \\ \text{Toep}(s) \end{array} + \begin{array}{c} \text{Grid of } UB \end{array} = \begin{array}{c} \text{Grid of } UV \end{array}
 \end{array}$$

Diagram illustrating the relationship between PCMM and PPMMs. The top equation shows a grid of U multiplied by a block matrix (consisting of a grid of A and a Toeplitz matrix $\text{Toep}(s)$ plus a grid of B) equals a grid of U multiplied by a grid of V . The bottom equation shows a grid of UA multiplied by a Toeplitz matrix $\text{Toep}(s)$ plus a grid of UB equals a grid of UV . Dimensions N are indicated for the V and UV grids.



PCMM \leq PPMMs (BCHPS'24, LZ'22)

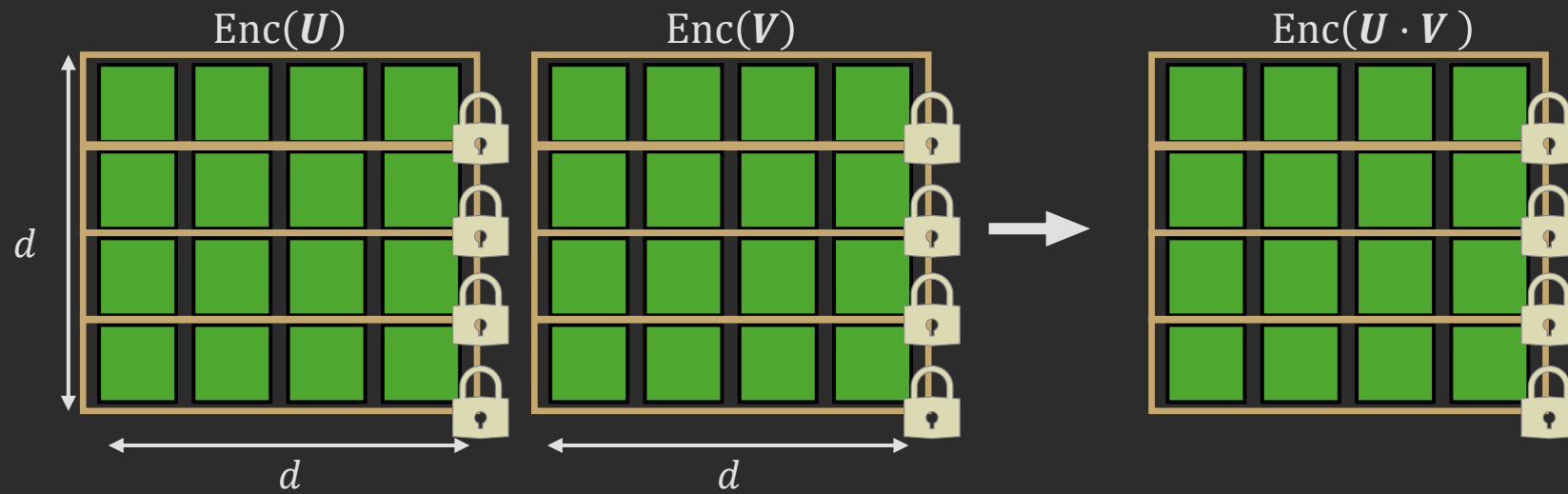
$$\begin{array}{c}
 \begin{array}{c} \text{Blue grid} \\ U \end{array} \left[\begin{array}{c} \text{Purple grid} \\ A \end{array} \begin{array}{c} \text{White box} \\ \text{Toep}(s) \end{array} + \begin{array}{c} \text{Purple grid} \\ B \end{array} \right] = \begin{array}{c} \text{Blue grid} \\ U \end{array} \begin{array}{c} \text{Green grid} \\ V \end{array} \\
 \begin{array}{c} \text{Purple grid} \\ UA \end{array} \begin{array}{c} \text{White box} \\ \text{Toep}(s) \end{array} + \begin{array}{c} \text{Purple grid} \\ UB \end{array} = \begin{array}{c} \text{Green grid} \\ UV \end{array}
 \end{array}$$

Diagram illustrating the reduction of PCMM to PPMMs. The top equation shows a blue grid U multiplied by a sum of a purple grid A and a white box $\text{Toep}(s)$, plus a purple grid B , resulting in a blue grid U multiplied by a green grid V . The bottom equation shows a purple grid UA multiplied by a white box $\text{Toep}(s)$, plus a purple grid UB , resulting in a green grid UV . Dimensions N are indicated for the grids.

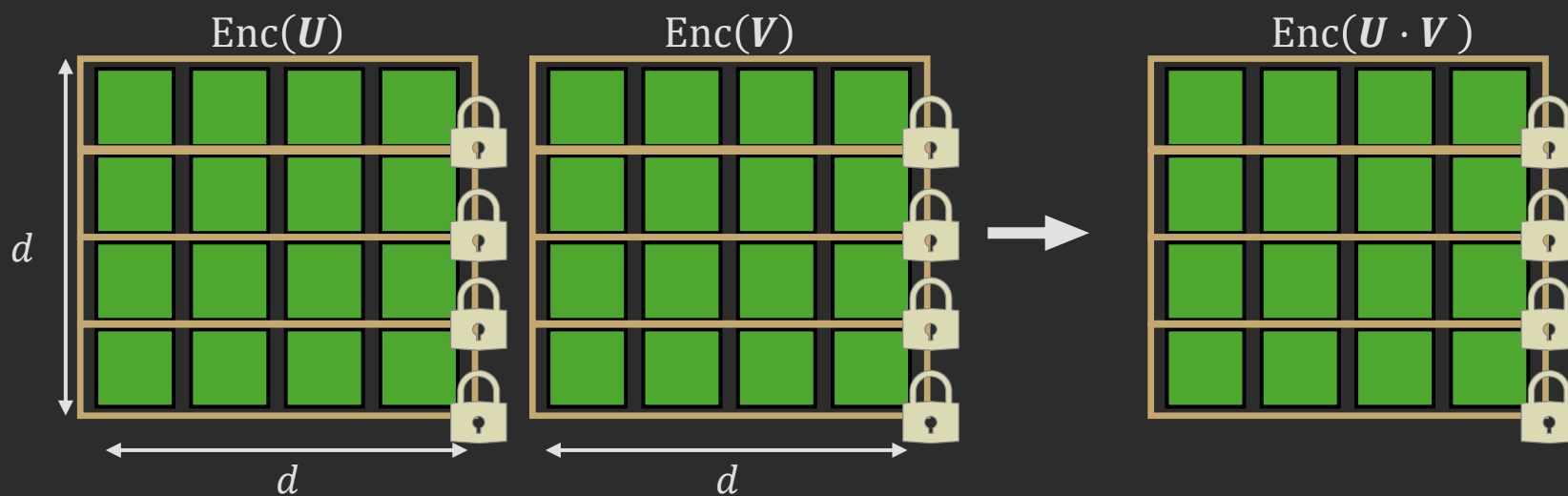
- ❖ $N \times N \times N$ PCMM \leq two $N \times N \times N$ PPMMs modulo Q
- ❖ We use fast **PPMM BLAS libraries** for $N \times N \times N$ PCMM
- ❖ For PCMMs with other dimensions, see BCHPS'24



CCMM for Large Matrices



CCMM for Large Matrices



- CCMM with RLWE-based (fully) homomorphic encryption schemes
 - Compatibility with the other machine learning tasks
 - High efficiency



CCMM?

$$\begin{bmatrix} U & V \end{bmatrix} = \left[\begin{bmatrix} A \\ \text{Toep}(s) \end{bmatrix} + B \right] \left[\begin{bmatrix} A' \\ \text{Toep}(s) \end{bmatrix} + B' \right]$$



CCMM?

$$\begin{aligned}
 \begin{bmatrix} U & V \end{bmatrix} &= \left[\begin{bmatrix} A \\ \text{Toep}(s) \end{bmatrix} + \begin{bmatrix} B \end{bmatrix} \right] \left[\begin{bmatrix} A' \\ \text{Toep}(s) \end{bmatrix} + \begin{bmatrix} B' \end{bmatrix} \right] \\
 &= \begin{bmatrix} A \\ \text{Toep}(s) \end{bmatrix} \begin{bmatrix} A' \\ \text{Toep}(s) \end{bmatrix} + \begin{bmatrix} A \\ \text{Toep}(s) \end{bmatrix} \begin{bmatrix} B' \end{bmatrix} + \begin{bmatrix} B \\ \text{Toep}(s) \end{bmatrix} \begin{bmatrix} A' \\ \text{Toep}(s) \end{bmatrix} + \begin{bmatrix} B \\ \text{Toep}(s) \end{bmatrix} \begin{bmatrix} B' \end{bmatrix}
 \end{aligned}$$



CCMM?

$$\begin{aligned}
 \begin{bmatrix} U & V \end{bmatrix} &= \left[\begin{bmatrix} A \\ \text{Toep}(s) \end{bmatrix} + B \right] \left[\begin{bmatrix} A' \\ \text{Toep}(s) \end{bmatrix} + B' \right] \\
 &= \begin{bmatrix} A \\ \text{Toep}(s) \end{bmatrix} \begin{bmatrix} A' \\ \text{Toep}(s) \end{bmatrix} + \begin{bmatrix} A \\ \text{Toep}(s) \end{bmatrix} B' + B \begin{bmatrix} A' \\ \text{Toep}(s) \end{bmatrix} + B B'
 \end{aligned}$$

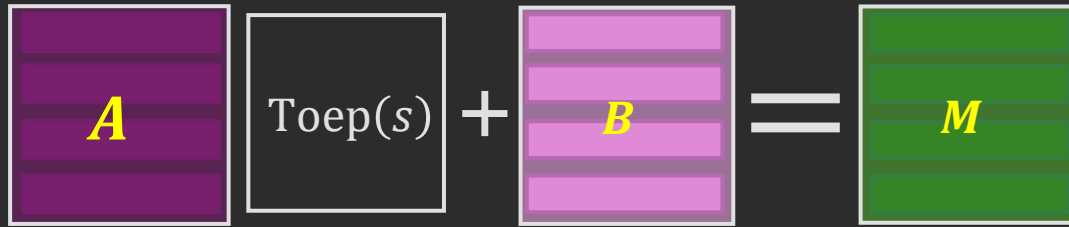


CCMM?

$$\begin{aligned}
 \begin{bmatrix} U \\ V \end{bmatrix} &= \left[\begin{bmatrix} A \\ \text{Toep}(s) \end{bmatrix} + B \right] \left[\begin{bmatrix} A' \\ \text{Toep}(s) \end{bmatrix} + B' \right] \\
 &= \begin{bmatrix} A \\ \text{Toep}(s) \end{bmatrix} \begin{bmatrix} A' \\ \text{Toep}(s) \end{bmatrix} + \begin{bmatrix} A \\ \text{Toep}(s) \end{bmatrix} B' + B \begin{bmatrix} A' \\ \text{Toep}(s) \end{bmatrix} + B B' \\
 &\xrightarrow{?} \begin{bmatrix} A \\ A'' \end{bmatrix} \text{Toep}(s) \text{Toep}(s) + \begin{bmatrix} A \\ B'' \end{bmatrix} \text{Toep}(s) + B \begin{bmatrix} A' \\ \text{Toep}(s) \end{bmatrix} + B B'
 \end{aligned}$$



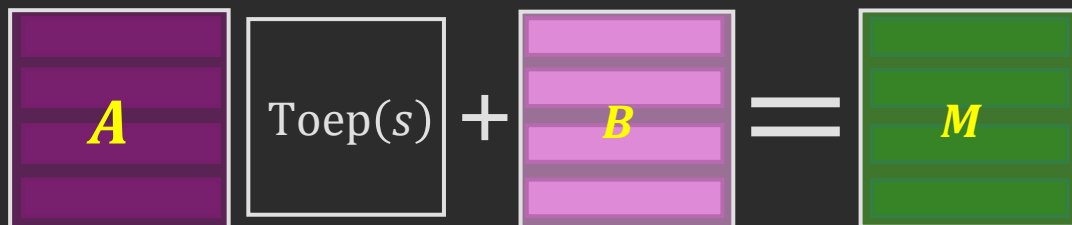
A.Toep(s) vs. Toep(s).A



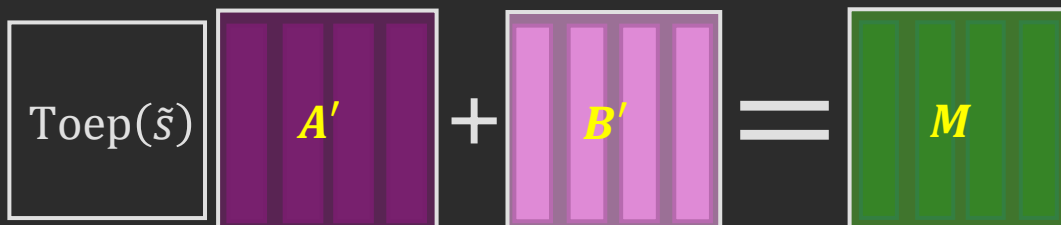
Encrypting each row: $a_i s + b_i = m_i$



A.Toep(s) vs. Toep(s).A



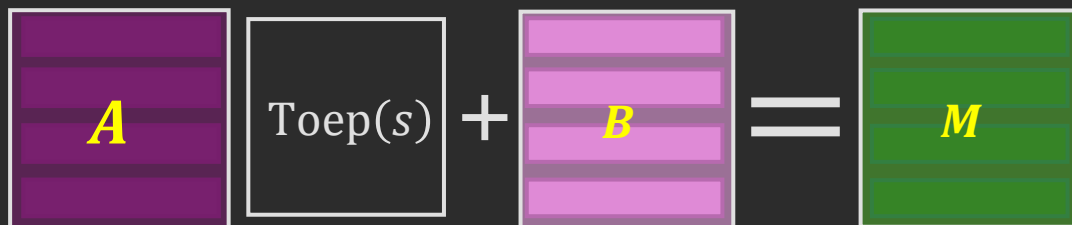
Encrypting each row: $a_i s + b_i = m_i$



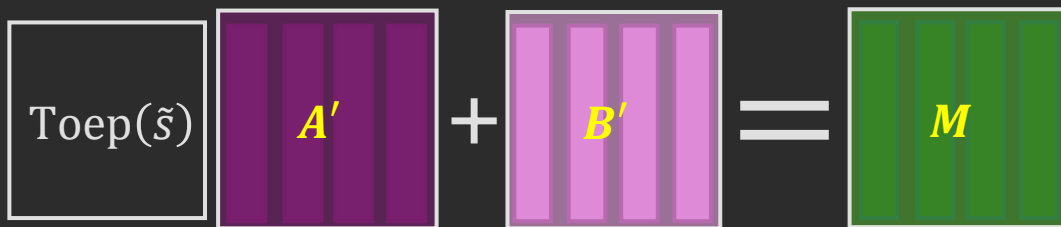
Encrypting each column: $a_j s + b_j = m'_j$



A.Toep(s) vs. Toep(s).A



Encrypting each row: $a_i s + b_i = m_i$



Encrypting each column: $a_j s + b_j = m'_j$

- ✓ N RLWE ciphertexts to encrypt
 $N \times N$ matrix M
 - Row: $A \cdot \text{Toep}(s) + B = M$
 - Column: $\text{Toep}(\tilde{s}) \cdot A' + B' = M$



Ciphertext Matrix Transpose (CMT)

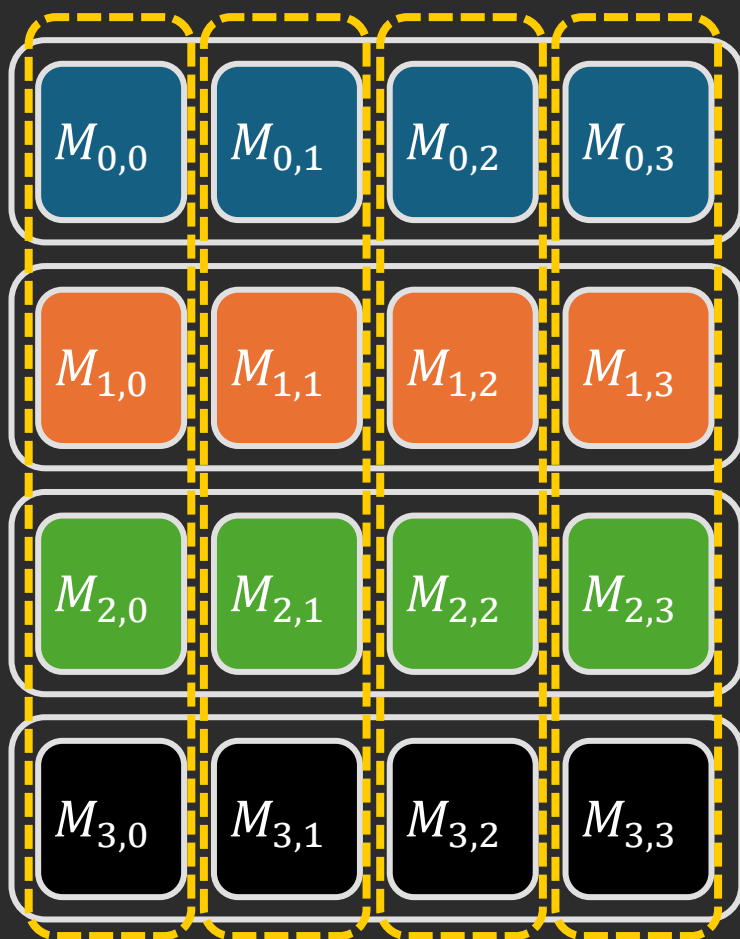
N ciphertexts

CMT:

$$m_i(X) = \sum_{j \in [N]} M_{i,j} X^j$$



Ciphertext Matrix Transpose (CMT)



N ciphertexts

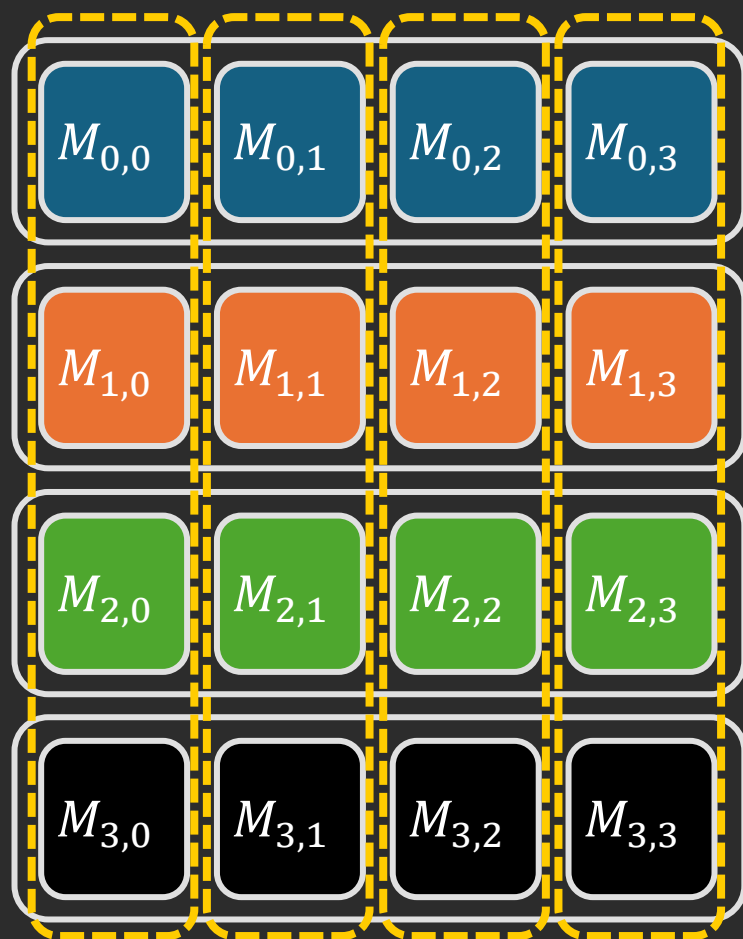
CMT: $m_i(X) = \sum_{j \in [N]} M_{i,j} X^j$

N ciphertexts

$m'_j(X) = \sum_{i \in [N]} M_{i,j} X^i$



Ciphertext Matrix Transpose (CMT)



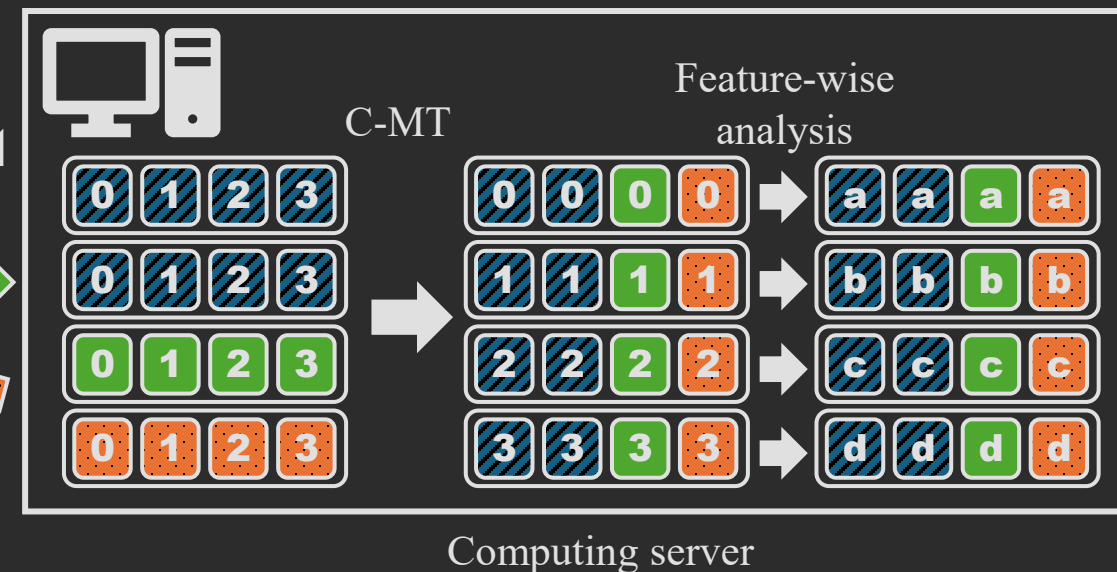
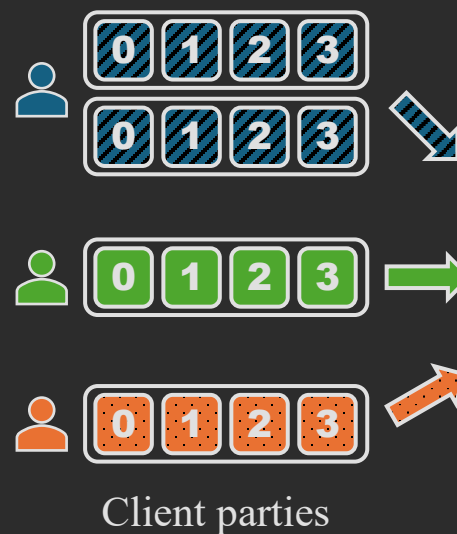
N ciphertexts

CMT: $m_i(X) = \sum_{j \in [N]} M_{i,j} X^j$

N ciphertexts

$m'_j(X) = \sum_{i \in [N]} M_{i,j} X^i$

Application



CMT with N keyswitchings

N ciphertexts

$$m_i(X) = \sum_{j \in [N]} M_{i,j} X^j$$



CMT

$$m'_j(X) = \sum_{i \in [N]} M_{i,j} X^i$$

N ciphertexts



CMT with N keyswitchings

Trace

$$\forall i, j, \quad M_{i,j} = N^{-1} \cdot \sum_{\sigma \in \text{Aut}} \sigma(m_i(X) \cdot X^{-j})$$

N ciphertexts

$$m_i(X) = \sum_{j \in [N]} M_{i,j} X^j$$

CMT

$$m'_j(X) = \sum_{i \in [N]} M_{i,j} X^i$$

N ciphertexts



CMT with N keyswitchings

Trace

$$\forall i, j, \quad M_{i,j} = N^{-1} \cdot \sum_{\sigma \in \text{Aut}} \sigma(m_i(X) \cdot X^{-j})$$

$$\{m_i\}_i$$

 N^2 keyswitchings

$$\{M_{i,j}\}_{i,j}$$

 N^2 add

$$\{m'_j\}_j$$

 N ciphertexts

$$m_i(X) = \sum_{j \in [N]} M_{i,j} X^j$$

CMT

$$m'_j(X) = \sum_{i \in [N]} M_{i,j} X^i$$

 N ciphertexts

CMT with N keyswitchings

Trace

$$\forall i, j, \quad M_{i,j} = N^{-1} \cdot \sum_{\sigma \in \text{Aut}} \sigma(m_i(X) \cdot X^{-j})$$

$$\forall j, \quad m'_j(X) = N^{-1} \cdot \sum_{i \in [N]} \sum_{\sigma \in \text{Aut}} \sigma(X^{-j} \cdot m_i(X)) \cdot X^i$$

N ciphertexts

$$m_i(X) = \sum_{j \in [N]} M_{i,j} X^j$$

CMT

$$m'_j(X) = \sum_{i \in [N]} M_{i,j} X^i$$

N ciphertexts



CMT with N keyswitchings

Trace

$$\forall i, j, \quad M_{i,j} = N^{-1} \cdot \sum_{\sigma \in \text{Aut}} \sigma(m_i(X) \cdot X^{-j})$$

$$\begin{aligned} \forall j, \quad m'_j(X) &= N^{-1} \cdot \sum_{i \in [N]} \sum_{\sigma \in \text{Aut}} \sigma(X^{-j} \cdot m_i(X)) \cdot X^i \\ &= N^{-1} \cdot \sum_{\sigma \in \text{Aut}} \sum_{i \in [N]} \sigma(X^{-j}) \cdot \sigma(m_i(X)) \cdot X^i \end{aligned}$$

 N ciphertexts

$$m_i(X) = \sum_{j \in [N]} M_{i,j} X^j$$



CMT

$$m'_j(X) = \sum_{i \in [N]} M_{i,j} X^i$$

 N ciphertexts

CMT with N keyswitchings

Trace

$$\forall i, j, \quad M_{i,j} = N^{-1} \cdot \sum_{\sigma \in \text{Aut}} \sigma(m_i(X) \cdot X^{-j})$$

$$\forall j, \quad m'_j(X) = N^{-1} \cdot \sum_{i \in [N]} \sum_{\sigma \in \text{Aut}} \sigma(X^{-j} \cdot m_i(X)) \cdot X^i$$

$$= N^{-1} \cdot \sum_{\sigma \in \text{Aut}} \sum_{i \in [N]} \sigma(X^{-j}) \cdot \sigma(m_i(X)) \cdot X^i$$

$$= N^{-1} \cdot \sum_{\sigma \in \text{Aut}} \sigma(X^{-j}) \cdot \sigma\left(\sum_{i \in [N]} m_i(X) \cdot \sigma^{-1}(X^i)\right)$$

 N ciphertexts

$$m_i(X) = \sum_{j \in [N]} M_{i,j} X^j$$



CMT

$$m'_j(X) = \sum_{i \in [N]} M_{i,j} X^i$$

 N ciphertexts

CMT with N keyswitchings

Trace

$$\forall i, j, \quad M_{i,j} = N^{-1} \cdot \sum_{\sigma \in \text{Aut}} \sigma(m_i(X) \cdot X^{-j})$$

Sharing automorphisms

$$\forall j, \quad m'_j(X) = N^{-1} \cdot \sum_{i \in [N]} \sum_{\sigma \in \text{Aut}} \sigma(X^{-j} \cdot m_i(X)) \cdot X^i$$

$$= N^{-1} \cdot \sum_{\sigma \in \text{Aut}} \sum_{i \in [N]} \sigma(X^{-j}) \cdot \sigma(m_i(X)) \cdot X^i$$

$$= N^{-1} \cdot \sum_{\sigma \in \text{Aut}} \sigma(X^{-j}) \cdot \underbrace{\sigma \left(\sum_{i \in [N]} m_i(X) \cdot \sigma^{-1}(X^i) \right)}_{\widetilde{m}_\sigma(X)}$$

 N ciphertexts

$$m_i(X) = \sum_{j \in [N]} M_{i,j} X^j$$

CMT

$$m'_j(X) = \sum_{i \in [N]} M_{i,j} X^i$$

 N ciphertexts

CMT with N keyswitchings

Trace

$$\forall i, j, \quad M_{i,j} = N^{-1} \cdot \sum_{\sigma \in \text{Aut}} \sigma(m_i(X) \cdot X^{-j})$$

Sharing automorphisms

$$\begin{aligned} \forall j, \quad m'_j(X) &= N^{-1} \cdot \sum_{i \in [N]} \sum_{\sigma \in \text{Aut}} \sigma(X^{-j} \cdot m_i(X)) \cdot X^i \\ &= N^{-1} \cdot \sum_{\sigma \in \text{Aut}} \sum_{i \in [N]} \sigma(X^{-j}) \cdot \sigma(m_i(X)) \cdot X^i \\ &= N^{-1} \cdot \sum_{\sigma \in \text{Aut}} \sigma(X^{-j}) \cdot \sigma \left(\sum_{i \in [N]} m_i(X) \cdot \sigma^{-1}(X^i) \right) \end{aligned}$$

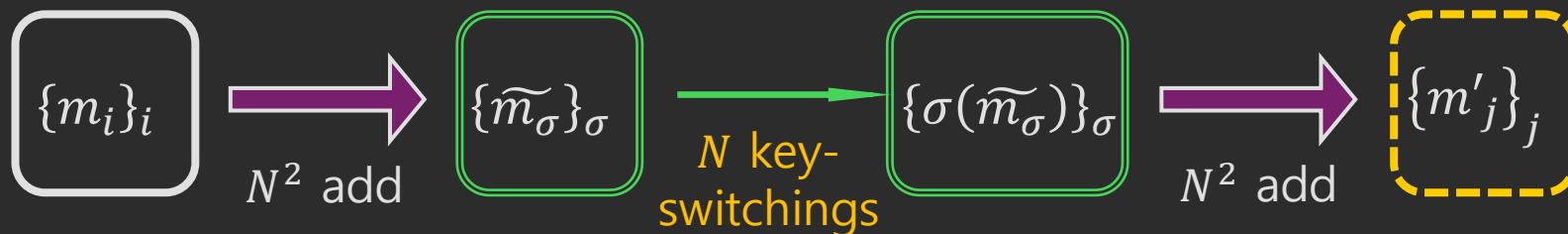
$\widetilde{m}_\sigma(X)$

 N ciphertexts

$$m_i(X) = \sum_{j \in [N]} M_{i,j} X^j$$

CMT

$$m'_j(X) = \sum_{i \in [N]} M_{i,j} X^i$$

 N ciphertexts

Tweak Algorithm

- ✓ $\widetilde{m}_\sigma(X) = \sum_i \sigma^{-1}(X^i) \cdot m_i$
- ✓ $m'_j(X) = \sum_\sigma \sigma(X^{-j}) \cdot \sigma(\widetilde{m}_\sigma)$



Tweak Algorithm

- ✓ $\widetilde{m}_\sigma(X) = \sum_i \sigma^{-1}(X^i) \cdot m_i$
- ✓ $m'_j(X) = \sum_\sigma \sigma(X^{-j}) \cdot \sigma(\widetilde{m}_\sigma)$

$$\text{Tweak}(\{m_i\}_{i \in [n]}) \mapsto \left\{ \sum_{i \in [n]} X^{\frac{2N}{n}ij} \cdot m_i \right\}_{j \in [n]}$$

N is the ring degree of RLWE



Tweak Algorithm

- ✓ $\widetilde{m}_\sigma(X) = \sum_i \sigma^{-1}(X^i) \cdot m_i$
- ✓ $m'_j(X) = \sum_\sigma \sigma(X^{-j}) \cdot \sigma(\widetilde{m}_\sigma)$

$$\text{Tweak}(\{m_i\}_{i \in [n]}) \mapsto \left\{ \sum_{i \in [n]} X^{\frac{2N}{n}ij} \cdot m_i \right\}_{j \in [n]}$$

N is the ring degree of RLWE

- $\text{Tweak}(\{m_i\}_{i \in [n]})$ can be done with
 - $\text{Tweak}(\{m_{2i}\}_{i \in [n/2]})$
 - $\text{Tweak}(\{m_{2i+1}\}_{i \in [n/2]})$
 - n ring additions



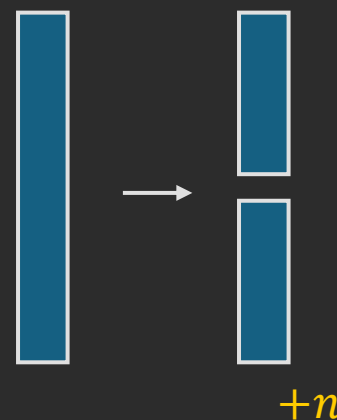
Tweak Algorithm

- ✓ $\widetilde{m}_\sigma(X) = \sum_i \sigma^{-1}(X^i) \cdot m_i$
- ✓ $m'_j(X) = \sum_\sigma \sigma(X^{-j}) \cdot \sigma(\widetilde{m}_\sigma)$

$$\text{Tweak}(\{m_i\}_{i \in [n]}) \mapsto \left\{ \sum_{i \in [n]} X^{\frac{2N}{n}ij} \cdot m_i \right\}_{j \in [n]}$$

N is the ring degree of RLWE

- $\text{Tweak}(\{m_i\}_{i \in [n]})$ can be done with
 - $\text{Tweak}(\{m_{2i}\}_{i \in [n/2]})$
 - $\text{Tweak}(\{m_{2i+1}\}_{i \in [n/2]})$
 - n ring additions



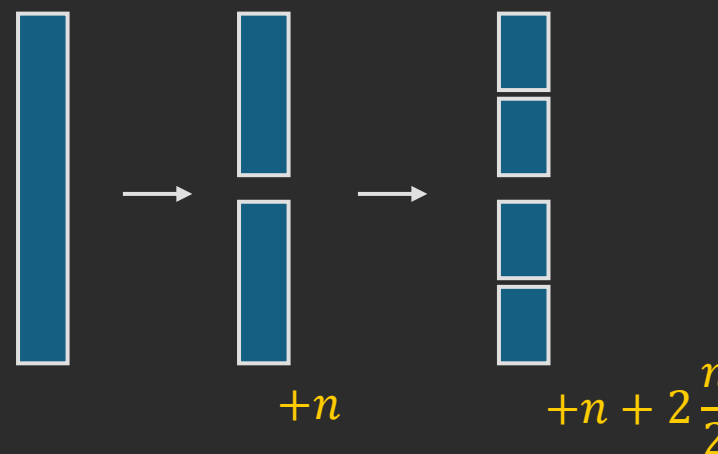
Tweak Algorithm

- ✓ $\widetilde{m}_\sigma(X) = \sum_i \sigma^{-1}(X^i) \cdot m_i$
- ✓ $m'_j(X) = \sum_\sigma \sigma(X^{-j}) \cdot \sigma(\widetilde{m}_\sigma)$

$$\text{Tweak}(\{m_i\}_{i \in [n]}) \mapsto \left\{ \sum_{i \in [n]} X^{\frac{2N}{n}ij} \cdot m_i \right\}_{j \in [n]}$$

N is the ring degree of RLWE

- $\text{Tweak}(\{m_i\}_{i \in [n]})$ can be done with
 - $\text{Tweak}(\{m_{2i}\}_{i \in [n/2]})$
 - $\text{Tweak}(\{m_{2i+1}\}_{i \in [n/2]})$
 - n ring additions



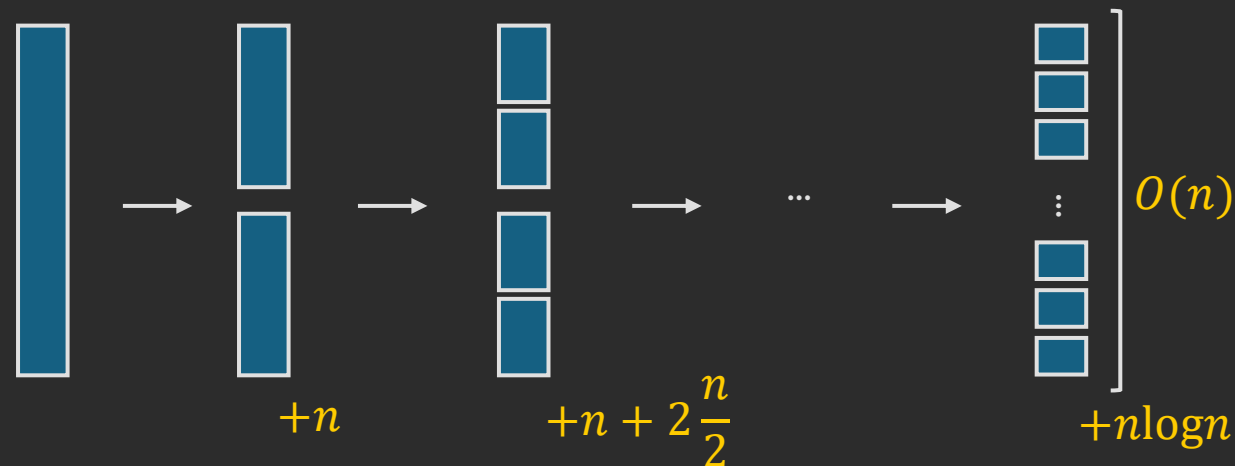
Tweak Algorithm

- ✓ $\widetilde{m}_\sigma(X) = \sum_i \sigma^{-1}(X^i) \cdot m_i$
- ✓ $m'_j(X) = \sum_\sigma \sigma(X^{-j}) \cdot \sigma(\widetilde{m}_\sigma)$

$$\text{Tweak}(\{m_i\}_{i \in [n]}) \mapsto \left\{ \sum_{i \in [n]} X^{\frac{2N}{n}ij} \cdot m_i \right\}_{j \in [n]}$$

N is the ring degree of RLWE

- $\text{Tweak}(\{m_i\}_{i \in [n]})$ can be done with
 - $\text{Tweak}(\{m_{2i}\}_{i \in [n/2]})$
 - $\text{Tweak}(\{m_{2i+1}\}_{i \in [n/2]})$
 - n ring additions



Tweak Algorithm

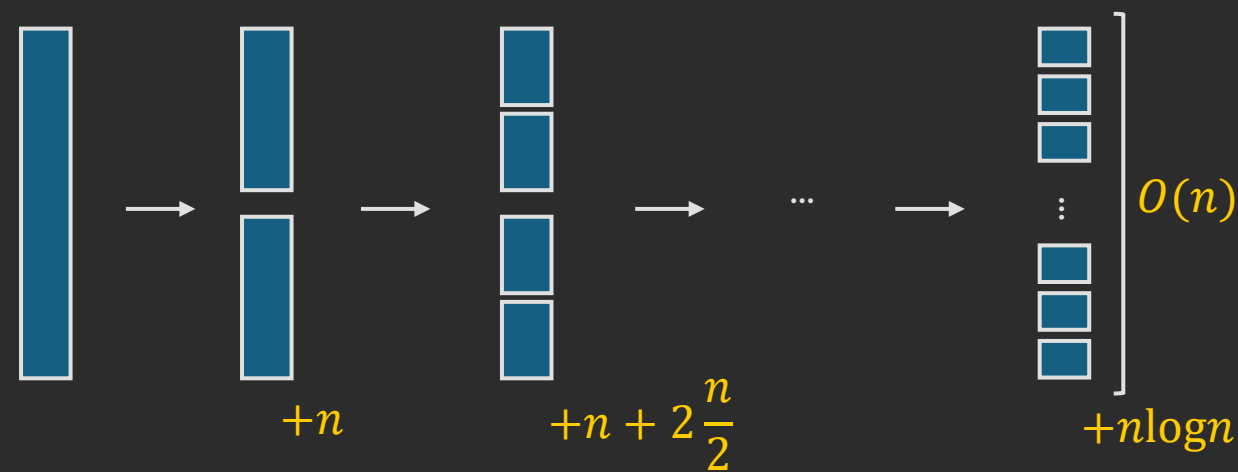
- ✓ $\widetilde{m}_\sigma(X) = \sum_i \sigma^{-1}(X^i) \cdot m_i$
- ✓ $m'_j(X) = \sum_\sigma \sigma(X^{-j}) \cdot \sigma(\widetilde{m}_\sigma)$

$$\text{Tweak}(\{m_i\}_{i \in [n]}) \mapsto \left\{ \sum_{i \in [n]} X^{\frac{2N}{n}ij} \cdot m_i \right\}_{j \in [n]}$$

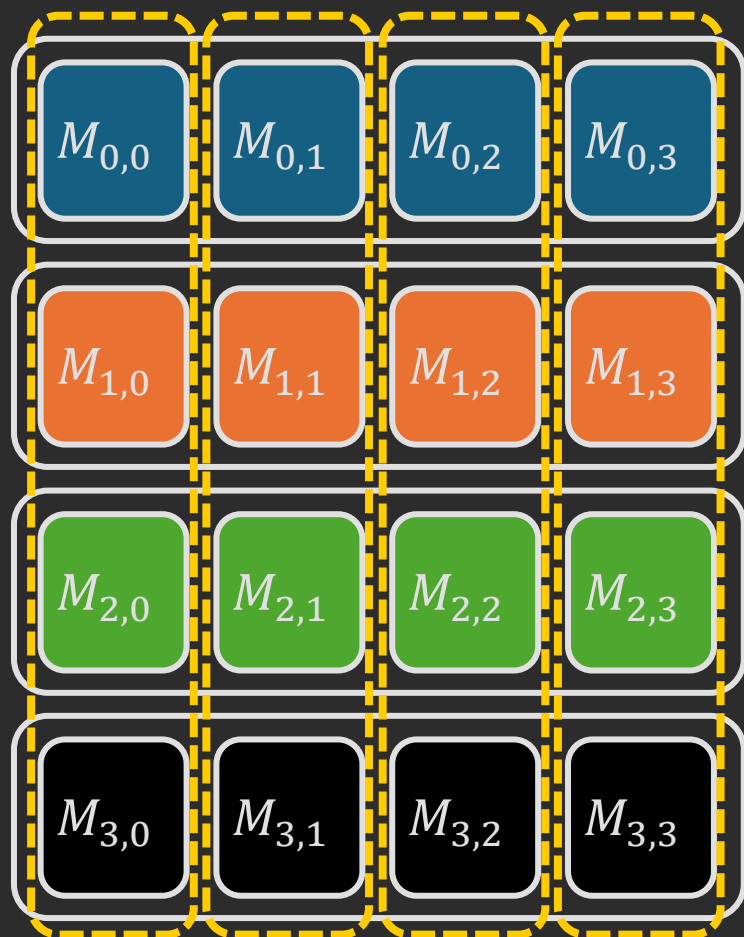
N is the ring degree of RLWE

- $\text{Tweak}(\{m_i\}_{i \in [n]})$ can be done with
 - $\text{Tweak}(\{m_{2i}\}_{i \in [n/2]})$
 - $\text{Tweak}(\{m_{2i+1}\}_{i \in [n/2]})$
 - n ring additions

❖ The cost of $\text{Tweak}(\{m_i\}_{i \in [n]})$ is $Nn \log n$



CMT Algorithm

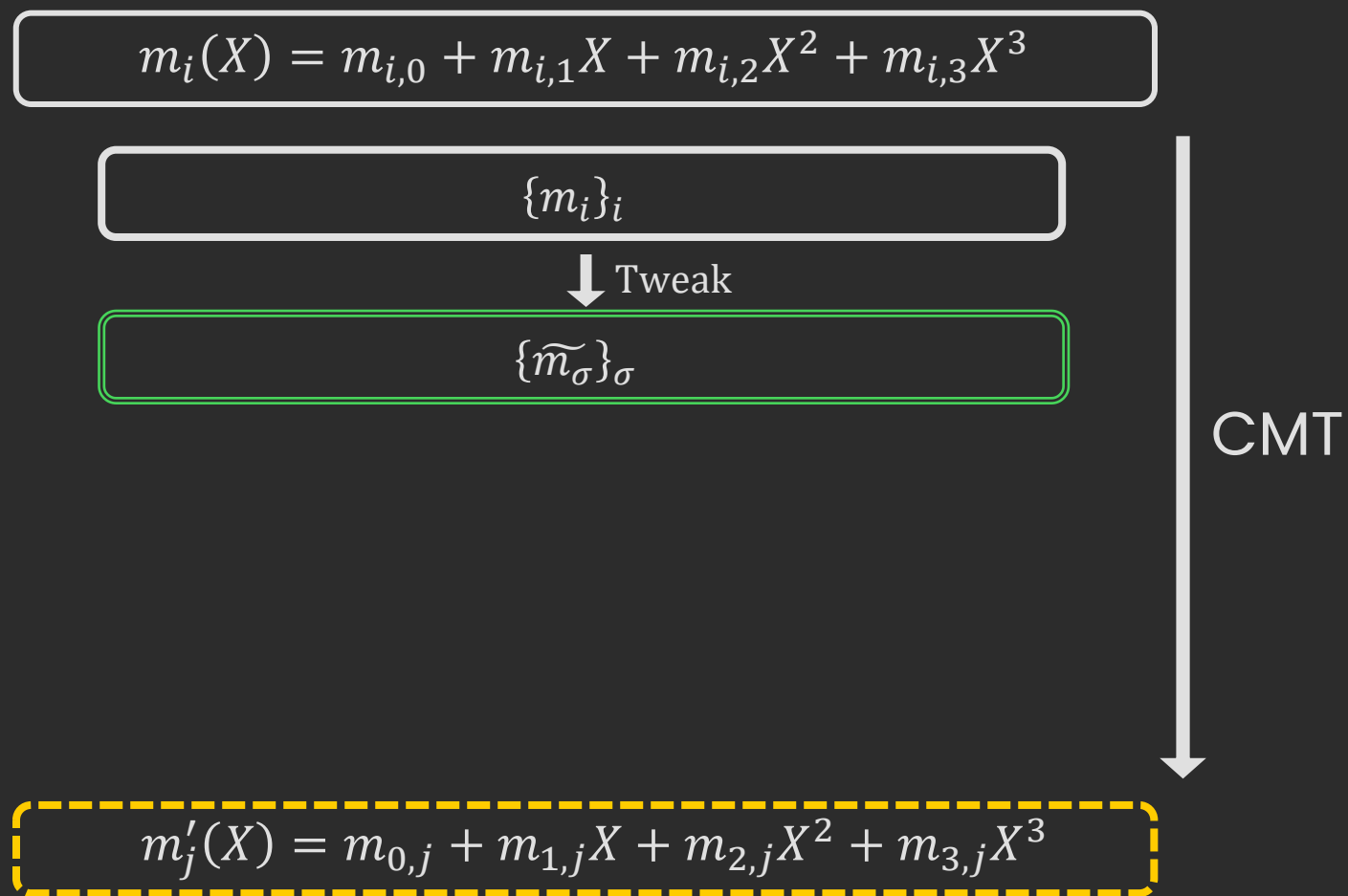
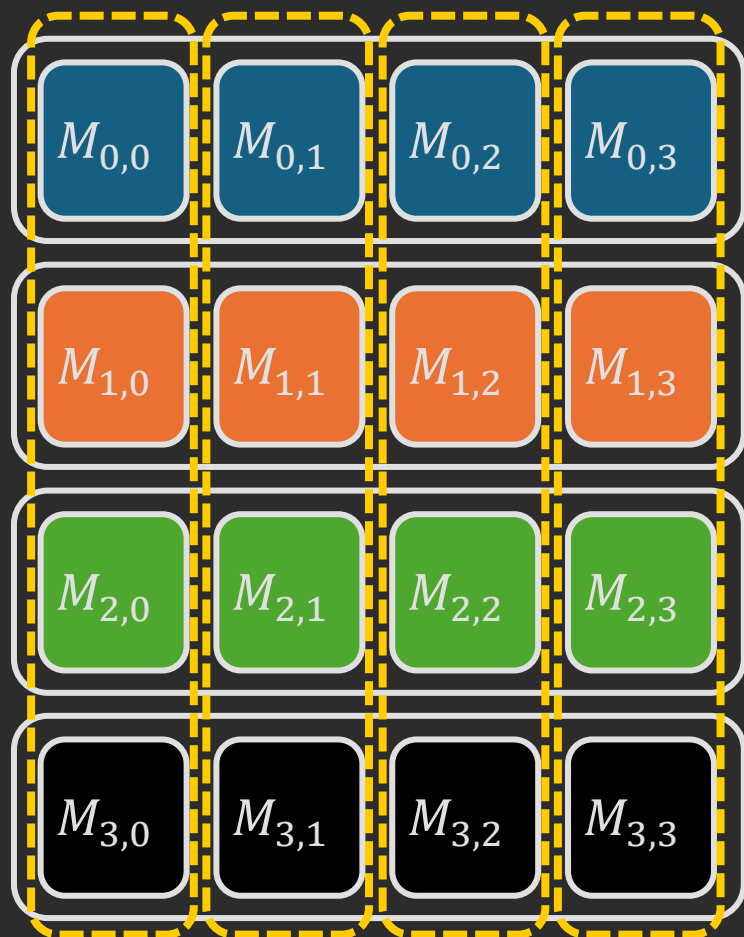


$$m_i(X) = m_{i,0} + m_{i,1}X + m_{i,2}X^2 + m_{i,3}X^3$$

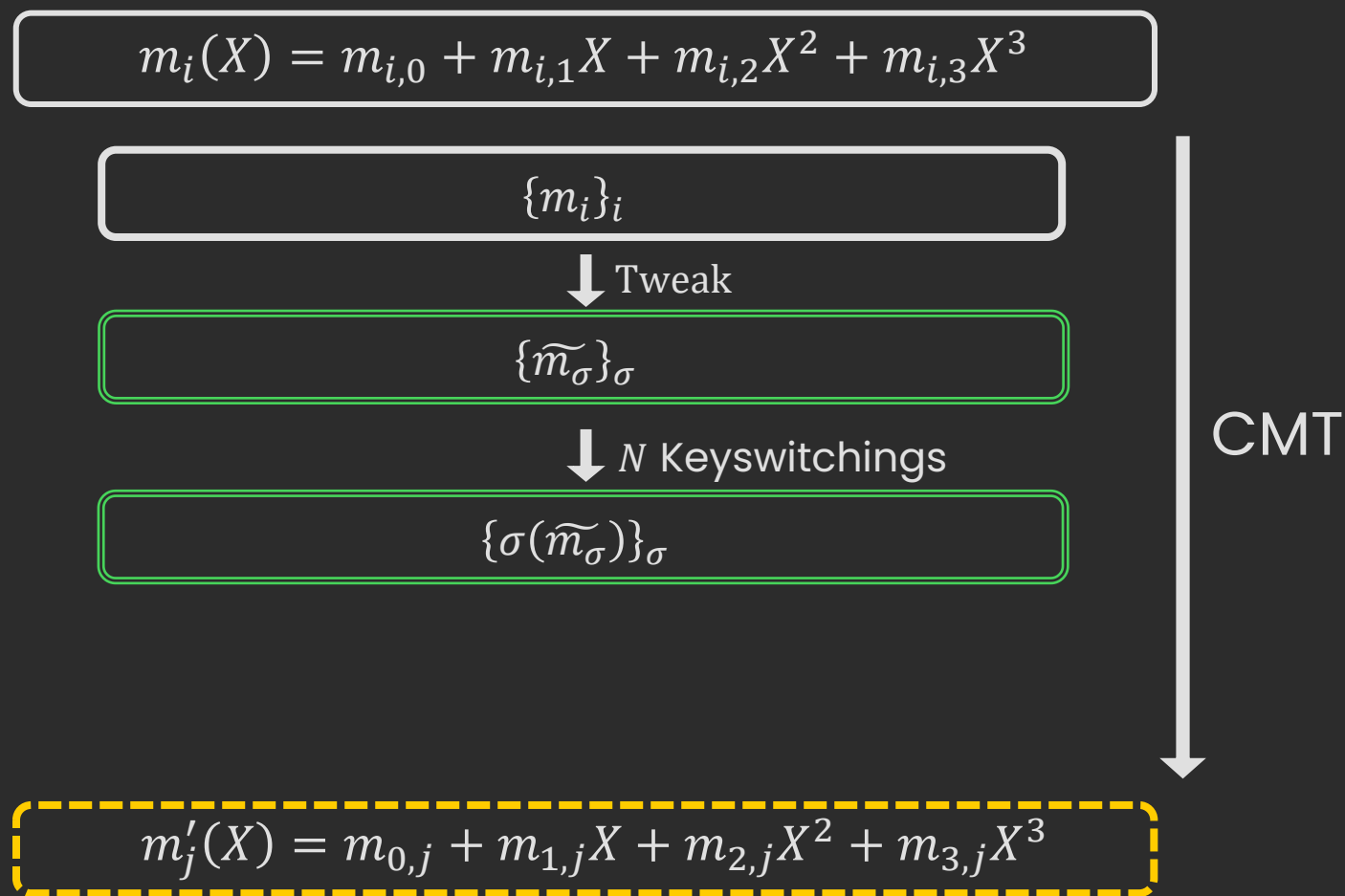
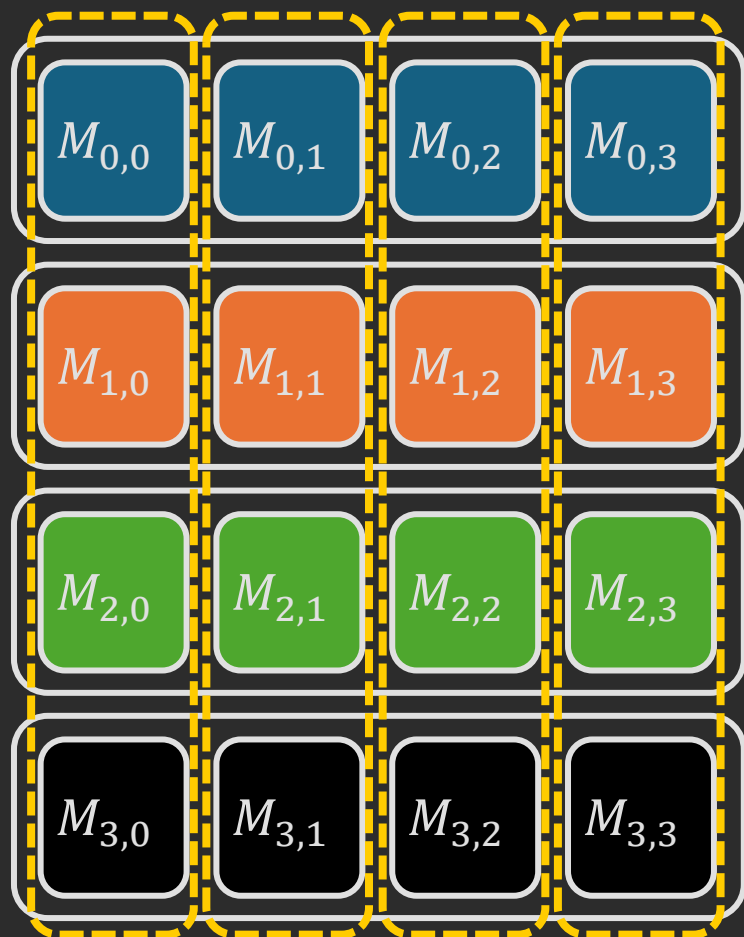
$$m'_j(X) = m_{0,j} + m_{1,j}X + m_{2,j}X^2 + m_{3,j}X^3$$



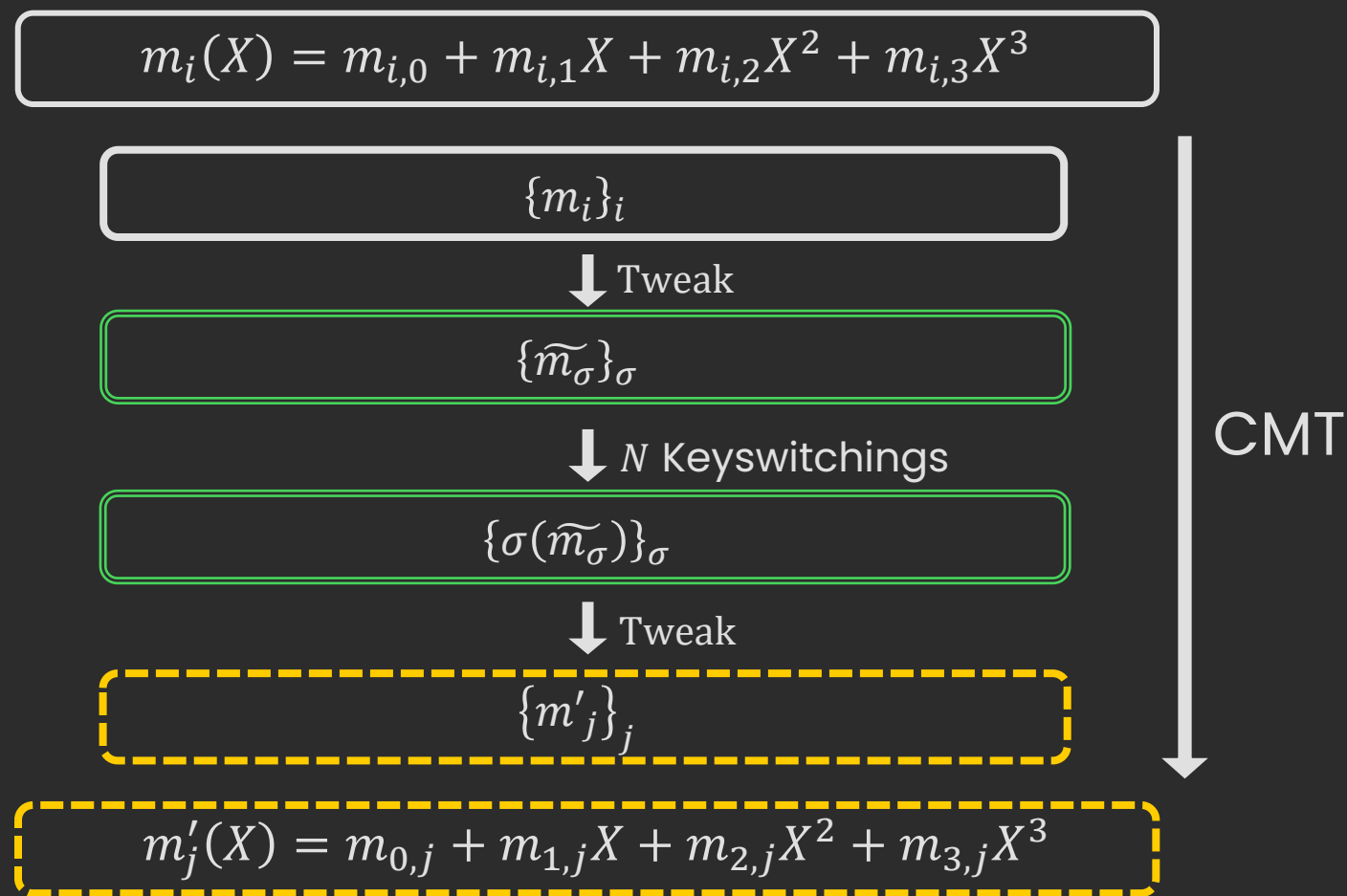
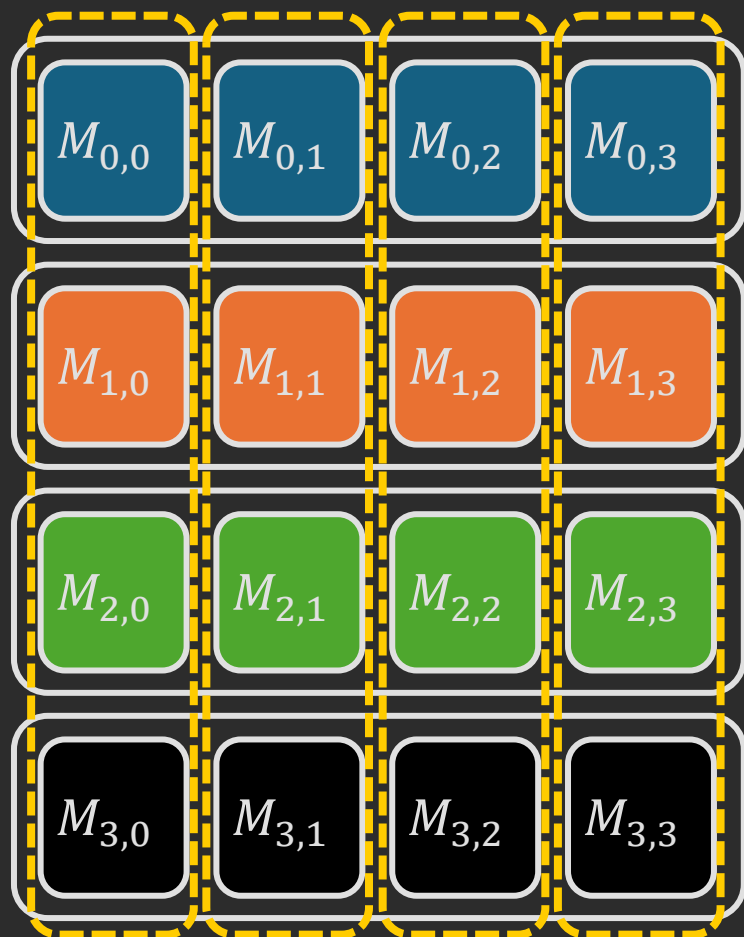
CMT Algorithm



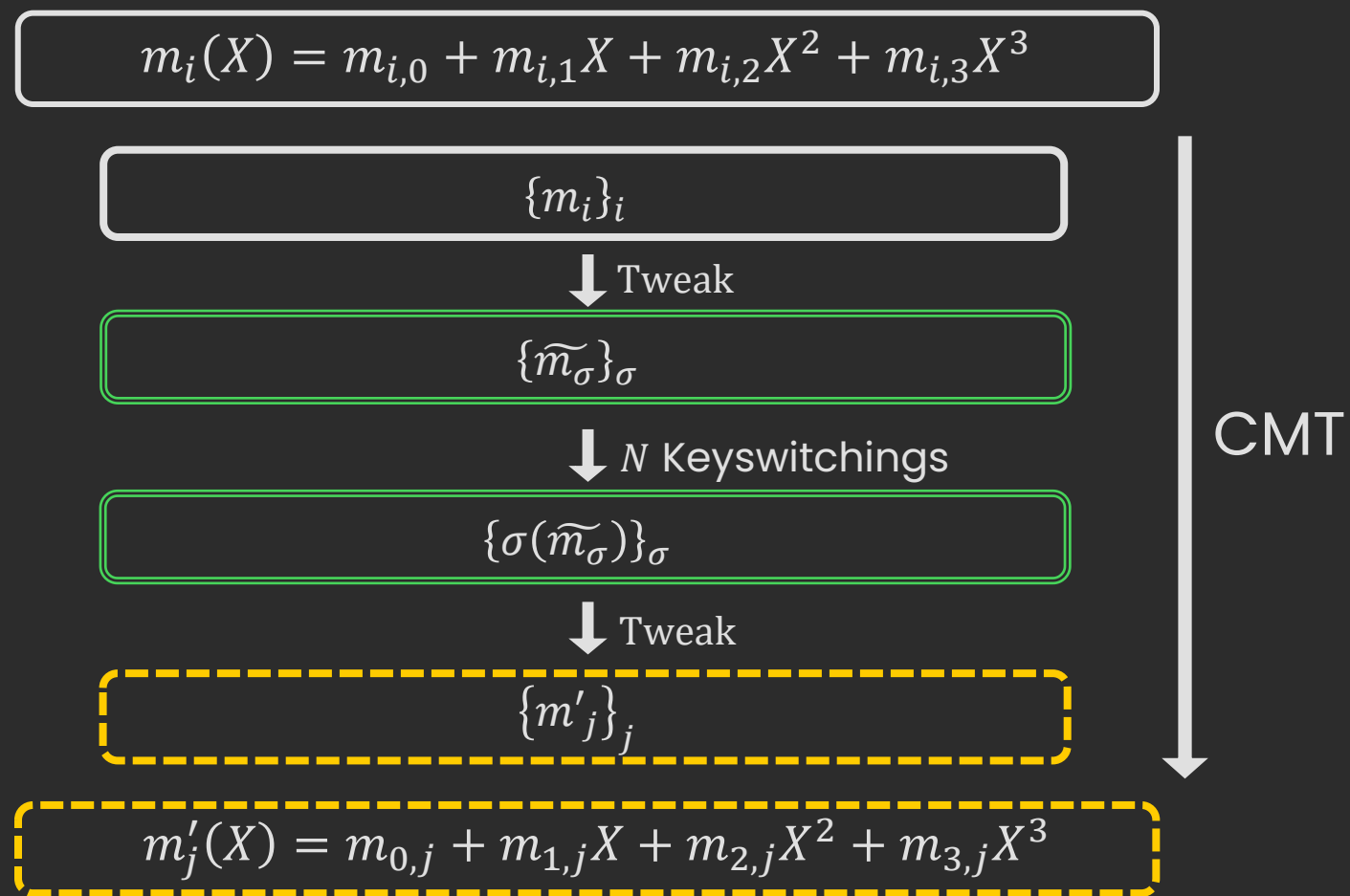
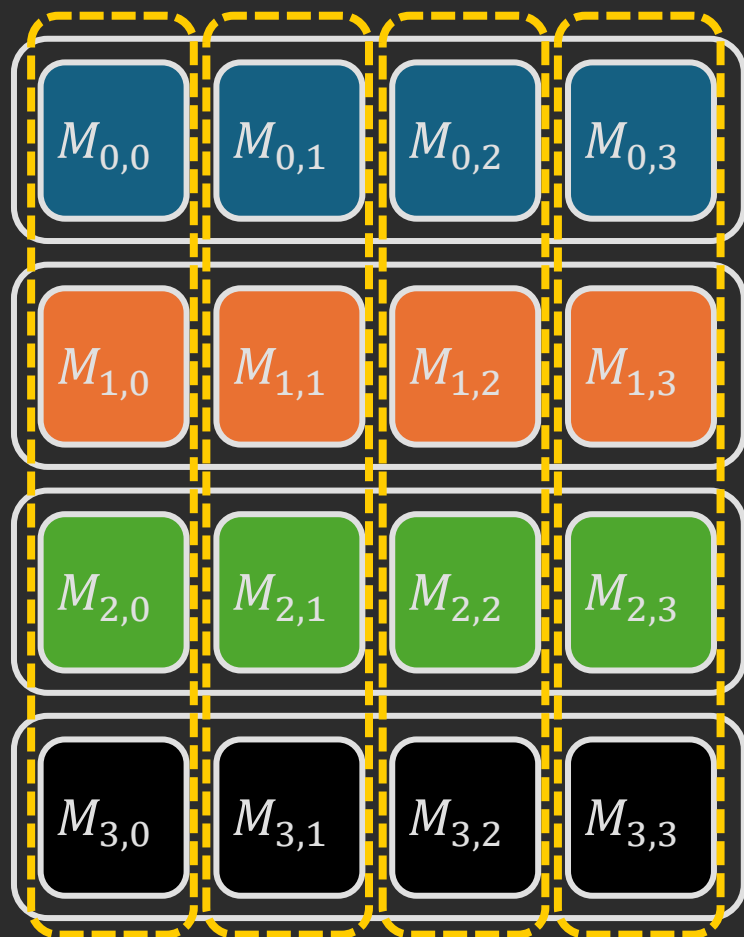
CMT Algorithm



CMT Algorithm



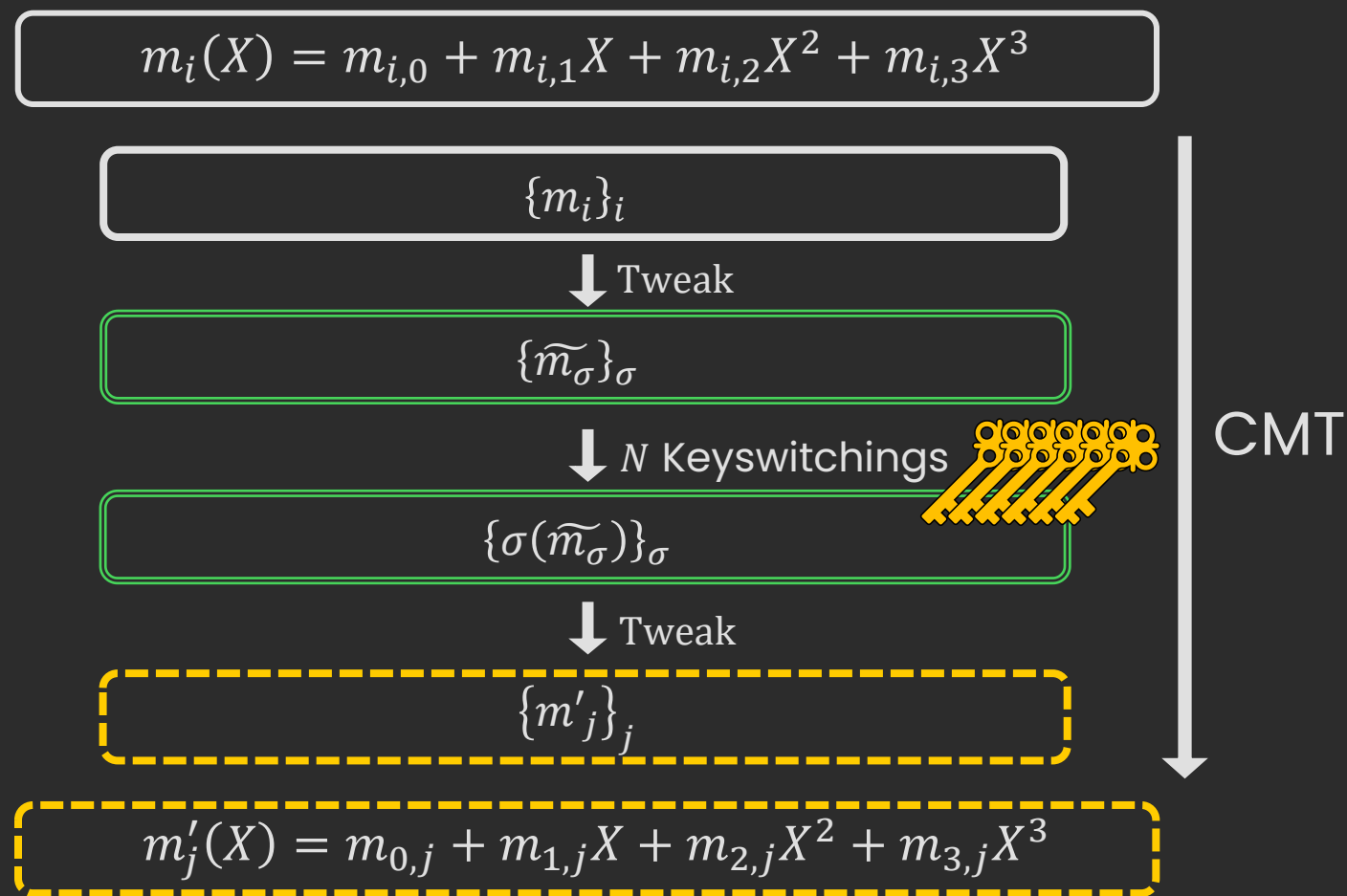
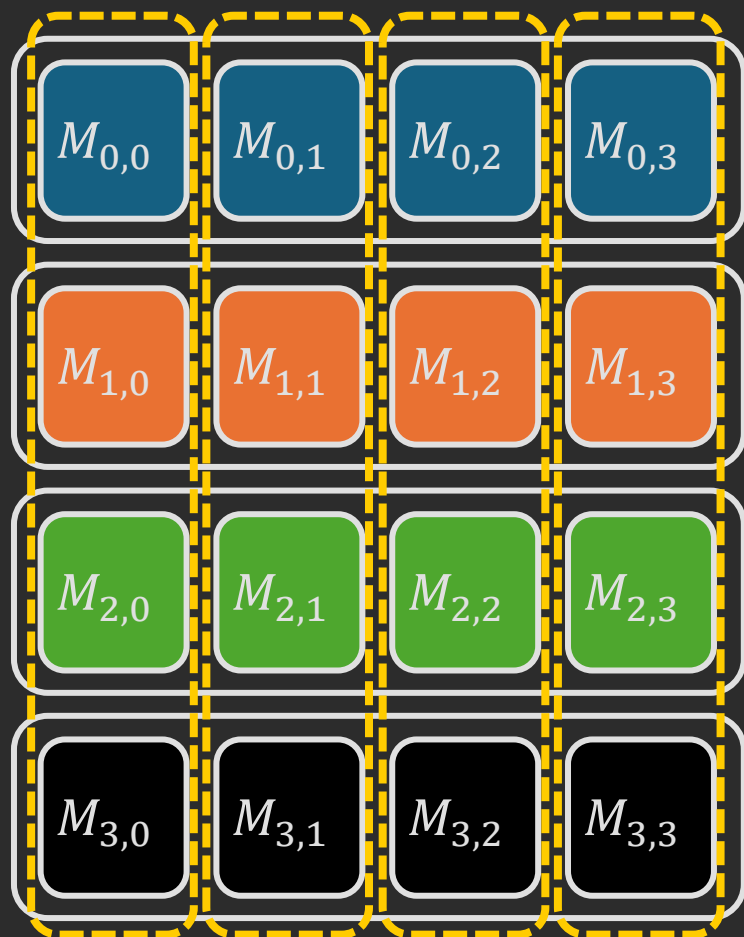
CMT Algorithm



❖ $\tilde{O}(N^2)$ operations



CMT Algorithm

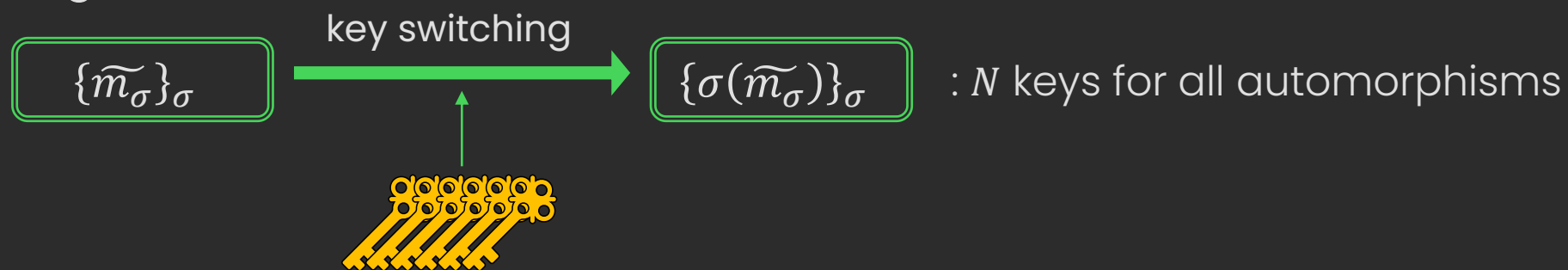


❖ $\tilde{O}(N^2)$ operations



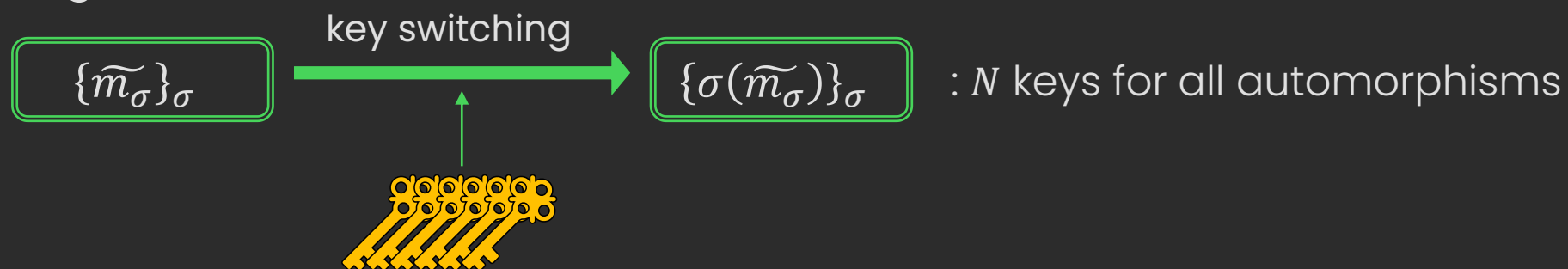
Lightweight CMT Algorithm

Basic algorithm

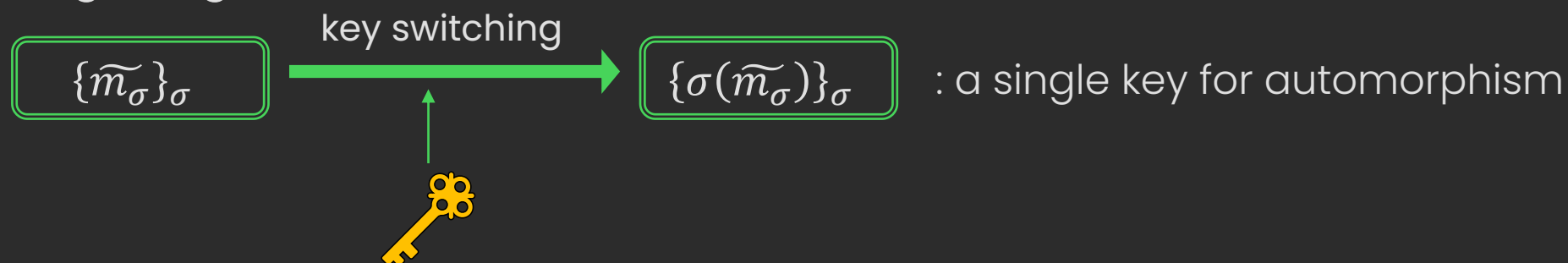


Lightweight CMT Algorithm

Basic algorithm

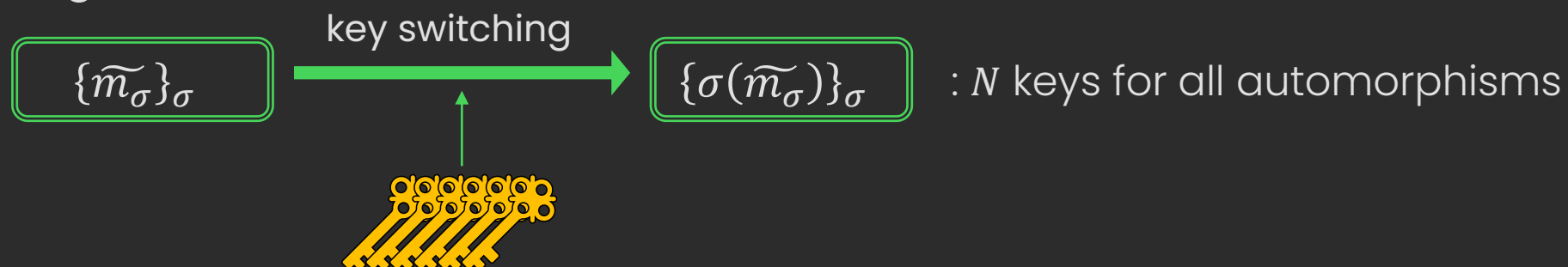


Lightweight algorithm

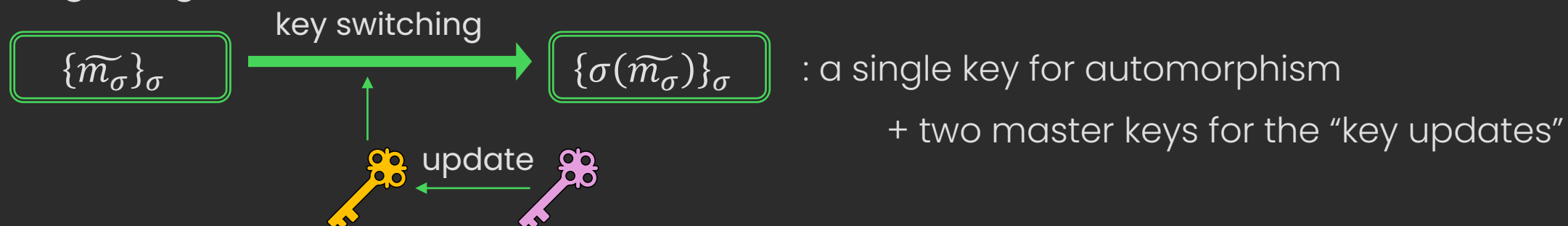


Lightweight CMT Algorithm

Basic algorithm



Lightweight algorithm



Reduction from CCMM to PPMM (1/2)

$$\begin{bmatrix} \text{Blue} & \text{Purple} \end{bmatrix} = \begin{bmatrix} \text{Blue} & \text{Toep}(s) & + & \text{Light Blue} \end{bmatrix} \begin{bmatrix} \text{Purple} & \text{Toep}(s) & + & \text{Light Purple} \end{bmatrix}$$



Reduction from CCMM to PPMM (1/2)

$$\begin{array}{c}
 \begin{bmatrix} \text{Blue} & \text{Purple} \end{bmatrix} = \begin{bmatrix} \text{Blue} & \text{Toep}(s) & + & \text{Light Blue} \end{bmatrix} \begin{bmatrix} \text{Purple} & \text{Toep}(s) & + & \text{Light Purple} \end{bmatrix} \\
 \xrightarrow{\text{CMT}} \begin{bmatrix} \text{Toep}(\tilde{s}) & \text{Blue} & + & \text{Light Blue} \end{bmatrix} \begin{bmatrix} \text{Purple} & \text{Toep}(s) & + & \text{Light Purple} \end{bmatrix}
 \end{array}$$

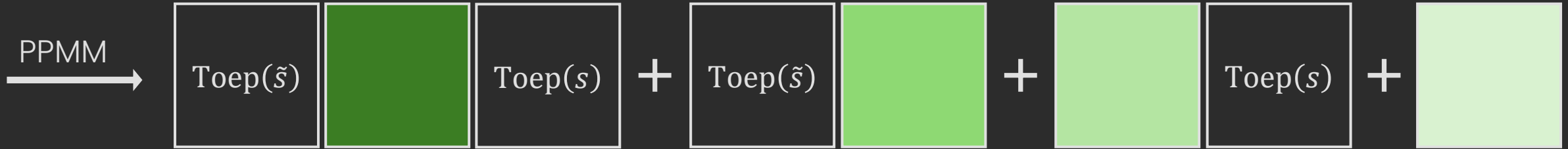


Reduction from CCMM to PPMM (1/2)

$$\begin{aligned}
 & \begin{bmatrix} \text{blue} & \text{purple} \end{bmatrix} = \begin{bmatrix} \text{blue} & \text{Toep}(s) & + & \text{light blue} \end{bmatrix} \begin{bmatrix} \text{purple} & \text{Toep}(s) & + & \text{pink} \end{bmatrix} \\
 & \xrightarrow{\text{CMT}} \begin{bmatrix} \text{Toep}(\tilde{s}) & \text{blue} & + & \text{light blue} \end{bmatrix} \begin{bmatrix} \text{purple} & \text{Toep}(s) & + & \text{pink} \end{bmatrix} \\
 & = \begin{bmatrix} \text{Toep}(\tilde{s}) & \text{blue} & \text{purple} & \text{Toep}(s) & + & \text{Toep}(\tilde{s}) & \text{blue} & \text{pink} \\ & \text{light blue} & \text{purple} & \text{Toep}(s) & + & & \text{light blue} & \text{pink} \end{bmatrix}
 \end{aligned}$$



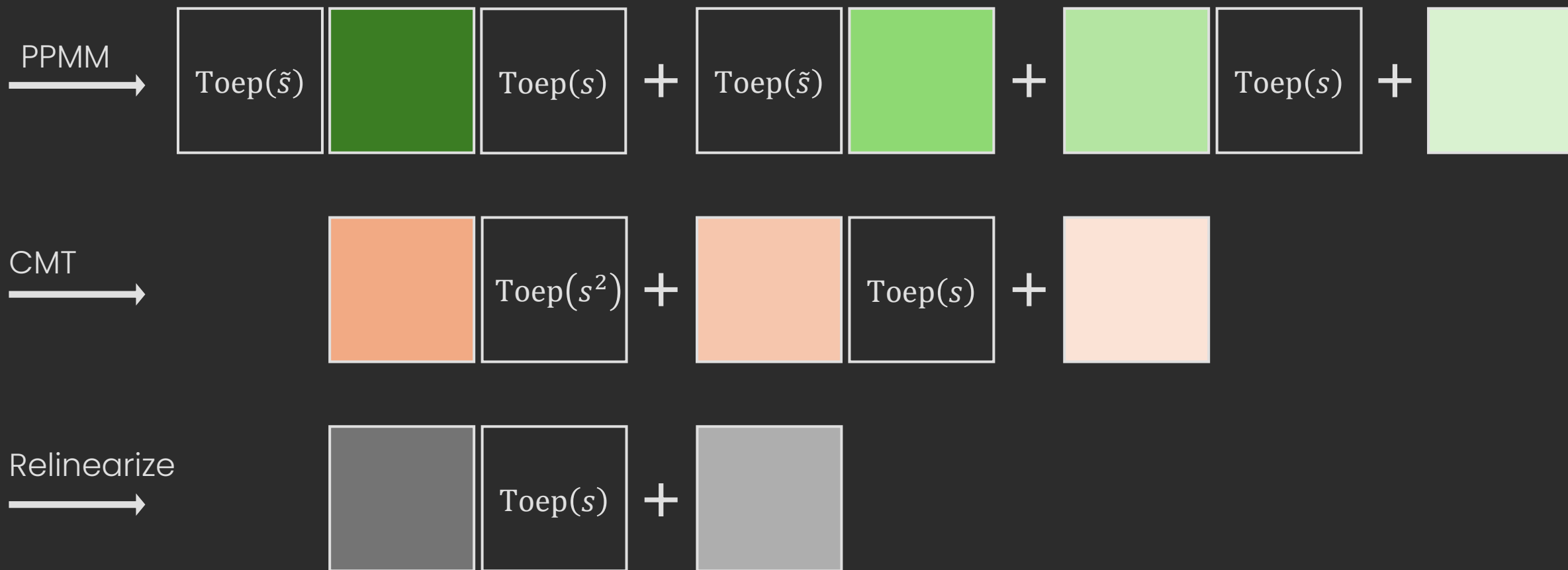
Reduction from CCMM to PPMM (2/2)



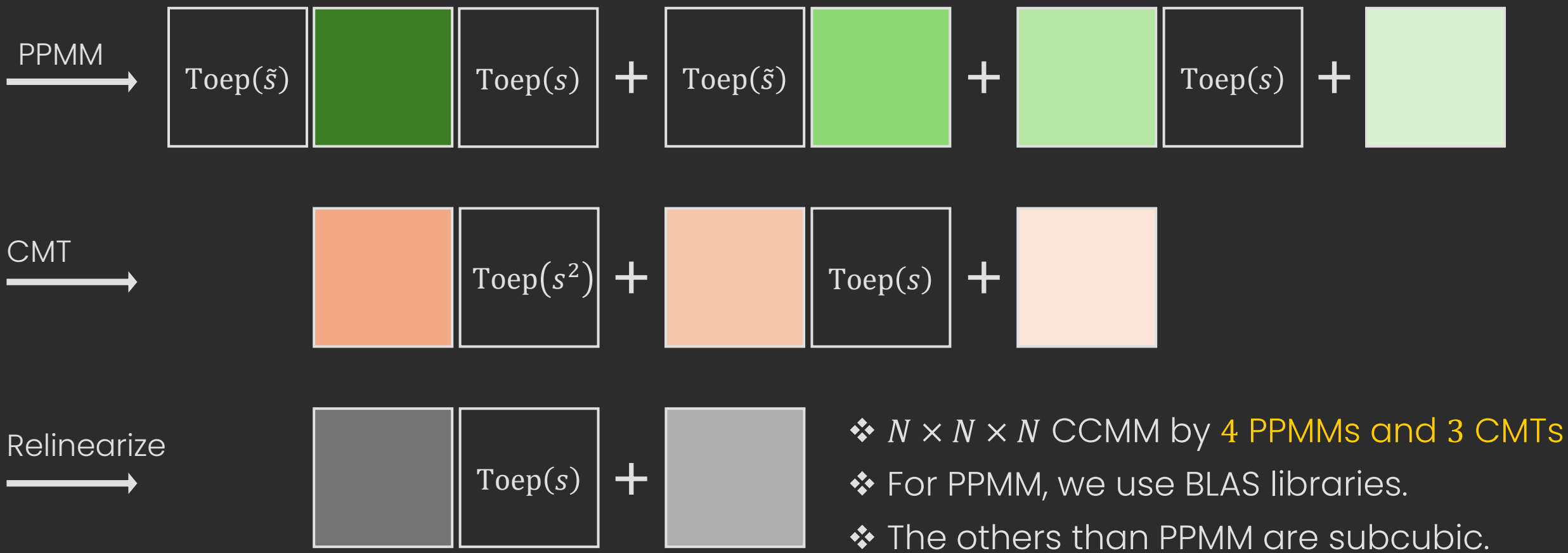
Reduction from CCMM to PPMM (2/2)



Reduction from CCMM to PPMM (2/2)



Reduction from CCMM to PPMM (2/2)



Experimental Results on CCMM

Algorithm	Matrix Dimension	$(\log N, \log Q)$	CMTs	PPMMs	Relin. & Resc.	Total (s)	Prec. (bit)	Key size (MB)
Basic	4096	$(12, 36 + 28)$	25.5	57.1	2.58	85.2	18.7	436
Basic	8192	$(13, 38 + 28)$	104	481	11.8	596	18.5	1960
Lightweight	8192	$(13, 38 + 28)$	186	474	11.8	672	18.5	1.57

All experiments are measured on Intel® Xeon® Gold 6242 CPU at 2.80GHz with a single-thread

All parameters are 128-bit secure

HEaaN library for HE, FLINT library (based on OpenBLAS) for PPMM



Experimental Results on CMT

Algorithm	Matrix Dimension	$(\log N, \log Q)$	Latency (s)	Prec. (bit)	Key size (MB)
Basic	2048	(11, 26)	0.764	10.7	27.3
Basic	4096	(12, 28)	3.04	16.3	134
Lightweight	4096	(12, 28)	4.92	14.2	0.246

All experiments are measured on Intel® Xeon® Gold 6242 CPU at 2.80GHz with a single-thread

All parameters are 128-bit secure

HEaaN library for HE



Follow-up Works

- BCHPS'25. "Encrypted Linear Algebra with BLAS"
arxiv/2503.16080
 - CC-MM / PC-MM / CC-Mv / PC-Mv with preprocessing using GSW
 - Flexible dimensional CC-MM and PC-MM
- Gentry. "Reducing Encrypted Matrix Multiplication to Plaintext Matrix Multiplication"
Presented at FHE.org conference 2025
 - C-MT using multi-variate polynomials
 - No published paper or experimental results available yet



Wrapping up!

- Fast CCMM
 - Leverage efficiency of BLAS libraries



Wrapping up!

- **Fast CCMM**
 - Leverage efficiency of BLAS libraries
- **Fast CMT**
 - Useful beyond being as a tool for CCMM



Wrapping up!

- **Fast CCMM**
 - Leverage efficiency of BLAS libraries
- **Fast CMT**
 - Useful beyond being as a tool for CCMM
- **Lightweight algorithms**
 - CCMM with keys less than 2 MB



Wrapping up!

- **Fast CCMM**
 - Leverage efficiency of BLAS libraries
- **Fast CMT**
 - Useful beyond being as a tool for CCMM
- **Lightweight algorithms**
 - CCMM with keys less than 2 MB

For matrix dimension 2^{12} :

PPMM
(OpenBLAS)

1.47 seconds

PCMM
(BCHPS'24)

17.1 seconds

CCMM
(this work)

85.2 seconds



Wrapping up!

- Fast CCMM
 - Leverage efficiency of BLAS libraries
- Fast CMT
 - Useful beyond being as a tool for CCMM
- Lightweight algorithms
 - CCMM with keys less than 2 MB

eprint : 2025/448

Thank you!

For matrix dimension 2^{12} :

PPMM
(OpenBLAS)

1.47 seconds

PCMM
(BCHPS'24)

17.1 seconds

CCMM
(this work)

85.2 seconds



References

[Park'25] J. H. Park. "Ciphertext-Ciphertext Matrix Multiplication: Fast for Large Matrices." Eurocrypt 2025

[BCHPS'24] Y. Bae, J. H. Cheon, G. Hanrot, J. H. Park, D. Stehlé. *"Plaintext-Ciphertext Matrix Multiplication and FHE Bootstrapping: Fast and Fused."* Crypto 2024

[BCHPS'25] Y. Bae, J. H. Cheon, G. Hanrot, J. H. Park, D. Stehlé. *"Encrypted Linear Algebra with BLAS."* arxiv, 2025

[JKLS'18] X. Jiang, M. Kim, K. Lauter, Y. Song. *"Secure Outsourced Matrix Computation and Application to Neural Networks."* CCS 2018

[LZ'22] J. Liu, L. F. Zhang. *"Privacy-preserving and publicly verifiable matrix multiplication."* IEEE Trans. on Services Computing, 2022

