Secure and Efficient General Matrix Multiplication On Cloud Using Homomorphic Encryption

1st Yang Gao
University of Central Florida
yang.gao@ucf.edu

2nd Gang Quan
Florida International University
gaquan@fiu.edu

3rd Soamar Homsi
Air Force Research Laboratory
soamar.homsi@us.af.mil

4th Wujie Wen
Lehigh University
wuw219@lehigh.edu

5th Liqiang Wang
University of Central Florida
liqiang.wang@ucf.edu

Abstract—Despite the enormous technical and financial advantages of cloud computing, security and privacy have always been the primary concerns for adopting cloud computing facilities, especially for government agencies and commercial sectors with high-security requirements. Homomorphic Encryption (HE) has recently emerged as an effective tool in ensuring privacy and security for sensitive applications by allowing computing on encrypted data. One major obstacle to employing HE-based computation, however, is its excessive computational cost, which can be orders of magnitude higher than its counterpart based on the plaintext. In this paper, we study the problem of how to reduce the HE-based computational cost for general Matrix Multiplication (MM), *i.e.*, a fundamental building block for numerous practical applications, by taking advantage of the Single Instruction Multiple Data (SIMD) operations supported by HE schemes. Specifically, we develop a novel element-wise algorithm for general matrix multiplication, based on which we propose two <u>HE</u>-based <u>G</u>eneral <u>M</u>atrix <u>M</u>ultiplication (HEGMM) algorithms to reduce the HE computation cost. Our experimental results show that our algorithms can significantly outperform the state-of-the-art approaches of HE-based matrix multiplication.

Index Terms—Homomorphic Encryption, privacy protection, Matrix Multiplication

I. INTRODUCTION

Cloud computing has become an attractive solution for industry and individuals due to its flexibility, scalability, reliability, sustainability, and affordability [3], [57]. Despite the tremendous technical and business advantages of cloud computing, security has been one of the primary concerns for cloud users, especially for those with high-security requirements [15], [41]. Even though cloud platforms allow their users to have full control over security settings and policies, public cloud infrastructures are commonly shared among different users and applications, making the applications vulnerable to malicious attacks.

Homomorphic Encryption (HE) [73], [74], [77] has emerged as an effective tool to address the security and privacy concerns associated with outsourcing data and computation to untrusted third parties, such as public cloud service providers. HE maintains data secrecy while in transit and during processing and assures that the decrypted results are identical to the outcome when the same operations are applied to the data in plaintext. HE has raised growing interest from researchers and practitioners of many security- and privacy-sensitive cloud applications in various domains such as health care, finance, and government agencies. One of the grand challenges, however, is how to deal with the tremendous computational cost for HE computations, which can be orders of magnitude higher than that in the plaintext space [52]. Unless HE computation cost can be effectively reduced, it would be infeasible to apply HE schemes in practical cloud applications.

In this paper, we study the problem of how to reduce HE computation cost for general matrix multiplications (MM) by taking advantage of the single instruction multiple data (SIMD) scheme for HE operations [67]. The SIMD scheme enables multiple data values to be packed into one ciphertext, and one single HE operation can be performed on all data elements in the ciphertext simultaneously. Accordingly, we develop a novel approach for HE-based MM operations, focusing on source matrices of arbitrary shapes. Specifically, we make the following contributions.

1) We present a novel element-wise method for MM. This method is general and can be applied to source matrices of arbitrary shapes with a significant performance improvement;
2) We develop two HE MM algorithms, with the second one improving the first one significantly. Our HE MM algorithms pack matrix elements judiciously in encrypted message "slots" and perform pertinent operations by taking advantage of the SIMD structure in HE schemes to reduce the number of primitive HE operations, such as HE multiplications, rotations, and additions, which are computationally expensive, and therefore can significantly reduce the computational cost;
3) We perform a rigorous analysis for the logical correctness of the algorithms and their complexities;
4) We implement our algorithms using a Python HE library, Pyfhel [58]. Extensive experimental results show that our proposed algorithms can significantly outperform the state-of-the-art approaches.

II. BACKGROUND AND RELATED WORK

In this section, we briefly introduce the relevant background of HE and discuss the related work.

A. Homomorphic Encryption (HE)

Homomorphic encryption (e.g. BGV [77], BFV [75], [76], and CKKS [78]) enables computations to be performed based on encrypted data, with results still in encrypted form. As such, HE represents a promising tool to greatly enhance data privacy and security, especially when outsourcing computations to the public cloud. In the meantime, HE can be extremely computationally intensive [32], and improving its computation efficiency is key to making this technology practical for real applications.

When performing HE matrix multiplication on the cloud, source matrices are first encrypted by clients and transferred to the cloud, and the results are transferred back to clients for decryption. Encrypting each individual element of a matrix into one cyphertext can lead to excessive encryption, decryption, and communication costs, in addition to a large number of HE operations. Table I shows our profiling results on encryption/decryption latency, message size, and computational costs with different HE operations (More detailed experimental settings are discussed in Sec. IV-A).

To this end, Gentry and Halevi [67] proposed an efficient key generation technique that enables SIMD operations in HE. By encrypting multiple data items into one ciphertext, one single operation can be applied to all encrypted elements in the same ciphertext simultaneously, and thus, space and computing resources can be used more efficiently.

As an example, BFV [75], [76] can support a number of primitive HE operations such as HE-Add (Addition), HE-Mult (Multiplication), HE-CMult (Constant Multiplication), and HE-Rot (Rotation). Given ciphertexts $ct_x = Enc(x_0, x_1, ..., x_n)$, $ct_y = Enc(y_0, y_1, ..., y_n)$ and a plaintext $pt = (p_0, p_1, ..., p_n)$, we have

- HE-Add: $ct_x + ct_y = Enc(x_0 + y_0, x_1 + y_1, ..., x_n + y_n)$
- HE-Mult: $ct_x \times ct_y = Enc(x_0 \times y_0, x_1 \times y_1, ..., x_n \times y_n)$
- HE-CMult: $ct_x \times pt = Enc(x_0 \times p_0, x_1 \times p_1, ..., x_n \times p_n)$
- HE-Rot: $Rot(ct_x, i) = Enc(x_i, x_{i+1}, .., x_n, x_0, .., x_{i-1})$

The HE operations are computationally intensive and can consume excessive computational time. In addition, HE operations also introduce *noises* when performed on encrypted data [35], which must be well under control for the results to be decrypted successfully. Several HE operations, especially HE-Mult, can be extremely time-consuming (approximately $600\times$ higher than its counterpart as shown in Table I) and introduce much larger noise [34]. Therefore, reducing the number of HE operations (especially the HE-Mult operations) becomes critical in designing practical applications, such as matrix multiplication, under the HE framework.

B. Related Work

There are numerous research efforts on improving the computational efficiency of MM (e.g. [48], [50], [46], [49], [50], [47], [51]). However, none of them can be readily adapted to optimize the computation efficiency of MM in the context of HE computation.

A naive method for HE MM is to encrypt each row/column in each matrix and then compute it using the traditional MM method. For the HE MM of $\mathcal{A}_{m \times l} \times \mathcal{B}_{l \times n}$, this would result in excessive storage requirements and computation times: $m \times n$ encrypted messages and totally $m \times l \times n$ HE-Mult operations. Another simple and intuitive approach (e.g. [70]) is to transform the MM problem into the matrix-vector multiplication problem and then adopt the SIMD scheme [66], [67] to perform the calculation. However, this requires $m + n$ ciphertexts and $m \times n$ homomorphic multiplication operations, which are still very costly.

Duong et al. [1] and Mishra et al. [2] presented approaches to pack the source matrix into a single polynomial, and then perform HE MM based on secure computation of inner product of encrypted vectors. It works for one single HE MM with well-defined dimensions but becomes problematic when multiple successive HE MMs are required in a cloud center.

Jiang et al. [61] proposed an intriguing HE MM approach for *square matrix* with $O(d)$ computational complexity. They expanded their HE MM algorithm to handle rectangle MM ($\mathcal{A}_{l \times d} \times \mathcal{B}_{d \times d}$) with $l \leq d$ and $d \mod l = 0$. Source matrices can be enlarged to suit shape requirements for MM with variable shapes, although this may increase processing time and resource utilization. Huang et al. [59] advocated using blocking to better handle rectangular MM with source matrices as *block* matrices with square matrices. This method is appealing for big matrices that cannot fit in one ciphertext. However, it is limited to square or two source rectangular matrices with integer multiple columns and rows.

Rathee et al. [68] proposed to encrypt source matrices into the two-dimensional *hypercube* structure [66] and then transform the MM problem to a series of matrix-vector multiplication problems. Huang et al. [60] extended this approach to make it applicable to general MM. As shown in section III-C, we have developed a more effective algorithm with higher computational efficiency.

III. APPROACHES

When performing HE matrix multiplication in the SIMD manner, we need to make sure that two operands are aligned and located at the same location, *i.e.*, the same slot in the two encoded ciphertexts. Rearranging individual slots in an encrypted message can be costly. Therefore, a key to the success of reducing the computational complexity of the HE MM is how to perform the MM using element-by-element additions and multiplication operations. In what follows, we first introduce a novel algorithm to calculate MM with arbitrary dimensions using element-wise additions and multiplications. We then discuss in more detail how we implement the HE MM algorithm on packed ciphertexts in the SIMD manner and its enhanced version.

A. The Matrix Multiplication Method using Element-Wise Computations

Consider an MM problem, $\mathcal{C}_{m \times n} = \mathcal{A}_{m \times l} \times \mathcal{B}_{l \times n}$, where m, l, and $n \in \mathbb{Z}^+$. Our goal is to develop an algorithm such that $\mathcal{C}_{m \times n} = \sum_i \mathcal{A}_i \odot \mathcal{B}_i$, where $\mathcal{A}_i, \mathcal{B}_i$ are certain transformations of $\mathcal{A}_{m \times l}, \mathcal{B}_{l \times n}$, respectively, and $\odot$ represents the element-wise multiplication. For ease of our presentation, we define four matrix transformation operators as follows:

$$
\begin{aligned}
\sigma(\mathcal{A})_{i,j} &= \mathcal{A}_{i,[i+j]_l}, & 0 \le i < m, 0 \le j < l \quad (1) \\
\tau(\mathcal{B})_{i,j} &= \mathcal{B}_{[i+j]_l,j}, & 0 \le i < l, 0 \le j < n \quad (2) \\
\epsilon^k_{m \times n}(\mathcal{A})_{i,j} &= \mathcal{A}_{i,[j+k]_l}, & 0 \le i < m, 0 \le j < n \quad (3) \\
\omega^k_{m \times n}(\mathcal{B})_{i,j} &= \mathcal{B}_{[i+k]_l,j}, & 0 \le i < m, 0 \le j < n \quad (4)
\end{aligned}
$$

where $[x]_y$ denotes $x \bmod y$.

The two transformation operators, σ and τ, are similar to those introduced in [61], but *more general and applicable to an arbitrary shape matrix* instead of a square matrix alone. Essentially, a σ transformation rotates each row of a matrix horizontally by its corresponding row index (for example, each element in the 2_{nd} row is cyclically rotated 2 positions to the left), and a τ transformation rotates a column by its corresponding column index. Figures 1(a) and 1(b) illustrate examples of σ and τ transformations.

We define two *new* transformation operators, $\epsilon^k_{m \times n}$ and $\omega^k_{m \times n}$, with respect to a matrix of arbitrary shape. Given $\mathcal{C}_{m \times n} = \mathcal{A}_{m \times l} \times \mathcal{B}_{l \times n}$, operator $\epsilon^k_{m \times n}(\mathcal{A})$ generates a matrix with size of $m \times n$ from $\mathcal{A}_{m \times l}$ (duplicating or cropping columns when necessary), by shifting matrix $\mathcal{A}_{m \times l}$ to the left for k columns. Similarly, $\omega^k_{m \times n}(\mathcal{B})$ generates a matrix with size of $m \times n$ from $\mathcal{B}_{l \times n}$ (duplicating or cropping rows when necessary), by shifting matrix $\mathcal{B}_{l \times n}$ upward for k rows. Figures 1(c) and 1(d) illustrate the transformation operators $\epsilon^0_{3 \times 5}(\mathcal{A})$, $\epsilon^1_{3 \times 5}(\mathcal{A})$, $\omega^0_{3 \times 5}(\mathcal{B})$, and $\omega^1_{3 \times 5}(\mathcal{B})$, respectively.
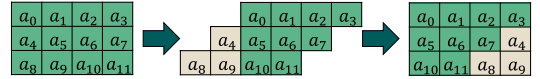
With the operators defined above, we can perform a general MM using the element-wise operations as follows:

$$
\mathcal{A}_{m \times l} \times \mathcal{B}_{l \times n} = \sum_{k=0}^{l-1} (\epsilon^k_{m \times n} \circ \sigma(\mathcal{A})) \odot (\omega^k_{m \times n} \circ \tau(\mathcal{B})), \quad (5)
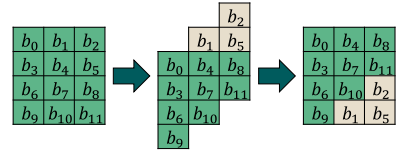$$

where $\circ$ represents the composition operation. Note that the multiplication (*i.e.,* $\odot$) in Equation (5) is element-wise and applied to the entire operands. Figure 2 shows an example of how an MM can be conducted based on Equation (5). Given two source matrices, i.e., $\mathcal{A}_{5 \times 3} \times \mathcal{B}_{3 \times 4}$, with $m = 5, l = 3$, and $n = 4$, σ and τ transformations are conducted on $\mathcal{A}$ and $\mathcal{B}$, respectively. Then three iterations of ϵ and ω transformations

are performed to obtain three partial products, which are accumulated to get the final product. We have the following proof sketch to show that the above method indeed produces the correct MM results for arbitrary matrices.
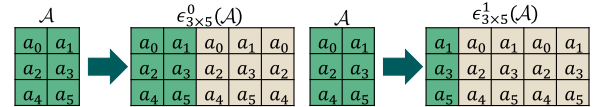
$$
\begin{aligned}
& \sum_{k=0}^{l-1} (\epsilon^k_{m \times n} \circ \sigma(\mathcal{A}))_{i,j} \cdot (\omega^k_{m \times n} \circ \tau(\mathcal{B}))_{i,j} \\
&= \sum_{k=0}^{l-1} \sigma(\mathcal{A})_{i,[j+k]_l} \cdot \tau(\mathcal{B})_{[i+k]_l,j} \\
&= \sum_{k=0}^{l-1} \mathcal{A}_{i,[i+j+k]_l} \cdot \mathcal{B}_{[i+j+k]_l,j} \\
&= \sum_{k=0}^{l-1} \mathcal{A}_{i,k} \cdot \mathcal{B}_{k,j} = (\mathcal{A} \cdot \mathcal{B})_{i,j}
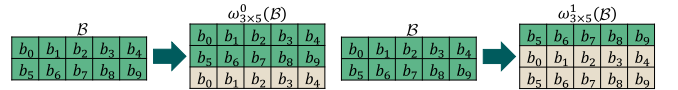\end{aligned}
\quad (6)
$$

(a) σ operator: rotating i_{th} row left by i slots.

(b) τ operator: rotating j_{th} column upward by j slots.

(c) $\epsilon^0_{3 \times 5}(\mathcal{A}_{3 \times 2})$ and $\epsilon^1_{3 \times 5}(\mathcal{A}_{3 \times 2})$ with $\mathcal{C}_{3 \times 5} = \mathcal{A}_{3 \times 2} \times \mathcal{B}_{2 \times 5}$.

(d) $\omega^0_{3 \times 5}(\mathcal{B}_{2 \times 5})$ and $\omega^1_{3 \times 5}(\mathcal{B}_{2 \times 5})$ with $\mathcal{C}_{3 \times 5} = \mathcal{A}_{3 \times 2} \times \mathcal{B}_{2 \times 5}$.

Fig. 1. The illustration of σ, τ, ϵ, and ω transformation operators

B. The HE-based General Matrix Multiplication (HEGMM)

With the element-wise matrix multiplication method introduced above, we are now ready to present our approach for HE matrix multiplication in the SIMD manner. As mentioned before,
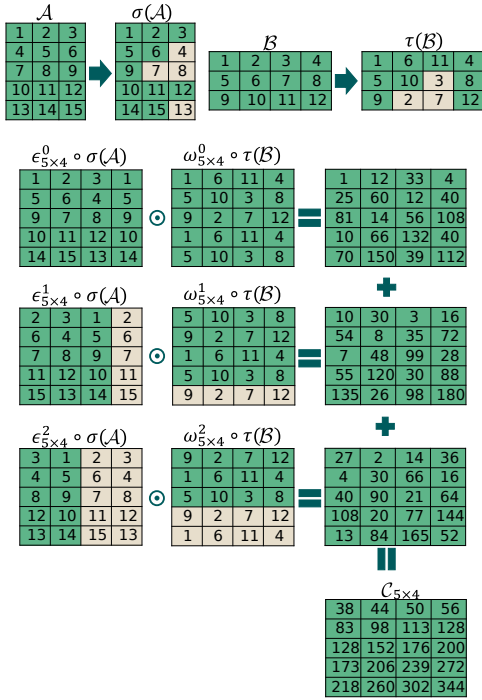
Fig. 2. An illustration example of the element-wise MM for $\mathcal{A}_{5\times3} \times \mathcal{B}_{3\times4}$ with $m = 5, l = 3$, and $n = 4$, σ and τ transformations are first conducted on $\mathcal{A}$ and $\mathcal{B}$, respectively. Then three iterations of ϵ and ω transformations are performed to obtain three partial products, which are accumulated to get the final product.

it is critical to minimize the number of HE operations (such as those, especially the HE-Mult, as shown in Table I) and thus reduce the computational cost. In this subsection, we first introduce how transformations, such as $\sigma, \tau, \epsilon_{m\times n}^k$, and $\omega_{m\times n}^k$, are performed using the primitive HE operations. We then present our first algorithm for HE MM based on the element-wise matrix multiplication strategy presented above.

1) Linear Transformation: To perform MM under the HE framework, two-dimensional matrices need to be flattened (either with column-major or row-major order) into one-dimensional ciphertexts, and all operations are performed on the ciphertexts. Therefore, a critical challenge to implement the computational strategy in Equation (5) is how to efficiently conduct $\sigma, \tau, \epsilon_{m\times n}^k$, and $\omega_{m\times n}^k$, $k = \{0, ..., (l-1)\}$ transformation operations. Note that an arbitrary linear transformation over a vector $\boldsymbol{m}$, i.e., $L : \mathcal{R}_x \rightarrow \mathcal{R}_y$, can be expressed as $L : \boldsymbol{m} \rightarrow U \cdot \boldsymbol{m}$, where $\mathbf{U} \in \mathcal{R}_{y\times x}$ is the transformation matrix. As shown by Halevi and Shoup [66], matrix-vector multiplications can be calculated using the combination of rotation and element-wise multiplication operations. Specifically, for $0 \le z < x$, let the z-th *diagonal vector* of $\mathbf{U}$ be

$$
\boldsymbol{u}_z = \begin{cases} \underbrace{(U_{0,z}, U_{1,z+1}, ..., U_{x-z-1,x-1}, 0, ..., 0)}_{|x|} & \text{if } z \ge 0 \\ \underbrace{(0, ..., 0, U_{z,0}, U_{z+1,1}, ..., U_{y-1,y-z-1})}_{|x|} & \text{if } z < 0 \end{cases}
$$

where x and y are the matrix dimensions and z is the index of diagonal vector.

Then we have

$$
\mathbf{U} \cdot \boldsymbol{m} = \sum_{-y \le z < x} (\boldsymbol{u}_z \odot HE\text{-}Rot(\boldsymbol{m}; z)), \qquad (7)
$$

where $\odot$ denotes the component-wise multiplication.

According to Equation (7), we can construct the transformations defined in Equations (1)-(4) with flattened matrix $\tilde{\mathcal{A}}$ and $\tilde{\mathcal{B}}$ such that

$$
\begin{aligned}
\sigma(\tilde{\mathcal{A}}) &= \mathbf{U}^\sigma \cdot \tilde{\mathcal{A}}, & (8) \\
\tau(\tilde{\mathcal{B}}) &= \mathbf{U}^\tau \cdot \tilde{\mathcal{B}}, & (9) \\
\epsilon_{m\times n}^k(\tilde{\mathcal{A}}) &= \mathbf{U}^{\epsilon_{m\times n}^k} \cdot \tilde{\mathcal{A}}, & (10) \\
\omega_{m\times n}^k(\tilde{\mathcal{B}}) &= \mathbf{U}^{\omega_{m\times n}^k} \cdot \tilde{\mathcal{B}}. & (11)
\end{aligned}
$$

Let $\tilde{\mathcal{A}}$ and $\tilde{\mathcal{B}}$ be source matrices flattened in the *column-major* order. By generalizing the location change patterns for the operators, we can define $\mathbf{U}^\sigma$, $\mathbf{U}^\tau$, $\mathbf{U}^{\epsilon_{m\times n}^k}$ and $\mathbf{U}^{\omega_{m\times n}^k}$ as follows:

$$
\mathbf{U}_{i+j\cdot m,h}^\sigma = \begin{cases} 1 & \text{if } h = i + [i+j]_l \cdot m, \\ 0 & \text{otherwise}; \end{cases} \qquad (12)
$$

$$
\mathbf{U}_{i+j\cdot l,h}^\tau = \begin{cases} 1 & \text{if } h = [i+j]_l + j \cdot l, \\ 0 & \text{otherwise}; \end{cases} \qquad (13)
$$

$$
\mathbf{U}_{i,j}^{\epsilon_{m\times n}^k} = \begin{cases} 1 & \text{if } j = [k \cdot m + i]_{m\cdot l} \\ 0 & \text{otherwise} \end{cases} \qquad (14)
$$

$$
\mathbf{U}_{i,j}^{\omega_{m\times n}^k} = \begin{cases} 1 & \text{if } j = [k + [i]_m]_l + \lfloor i/m \rfloor \cdot l \\ 0 & \text{otherwise}; \end{cases} \qquad (15)
$$

For the sake of clarity, scopes of i, j and h in Equation (12)-(15) are listed below.

	i	j	h
$\mathbf{U}^\sigma$	$[0, m)$	$[0, l)$	$[0, ml)$
$\mathbf{U}^\tau$	$[0, l)$	$[0, n)$	$[0, nl)$
$\mathbf{U}^{\epsilon_{m\times n}^k}$	$[0, mn)$	$[0, ml)$	N/A
$\mathbf{U}^{\omega_{m\times n}^k}$	$[0, mn)$	$[0, nl)$	N/A

When $\tilde{\mathcal{A}}$ and $\tilde{\mathcal{B}}$ are matrices flattened in the *row-major* order, similar transformation matrices can be constructed. We omit it due to page limit.

Note that, from equation (7), the $\sigma, \tau, \epsilon_{m\times n}^k$, and $\omega_{m\times n}^k$ operations can be realized using a sequence of HE-Rot, HE-CMult, and HE-Add operations. Figure 3 shows examples of transformations $\epsilon_{5\times3}^1(\mathcal{A})$ and $\omega_{3\times5}^1(\mathcal{B})$ as well as the associated matrices $\mathbf{U}^{\epsilon_{5\times3}^1}$ and $\mathbf{U}^{\omega_{3\times5}^1}$ for matrix $\mathcal{A}_{5\times3}$ and $\mathcal{B}_{3\times5}$, respectively. In the meantime, equation (7) clearly shows that the computational cost depends heavily on how many diagonal vectors (i.e., $\boldsymbol{u}_z$ in equation (7)) in the corresponding transformation matrices, i.e., $\mathbf{U}^\sigma$, $\mathbf{U}^\tau$, $\mathbf{U}^{\epsilon_{m\times n}^k}$, and $\mathbf{U}^{\omega_{m\times n}^k}$, are non-zeros. The more the non-zero diagonal vectors are, the higher the computation costs become. To this end, we have the following theorems that reveal important properties related to non-zero diagonal vectors in these transformation matrices.

Theorem III.1. *Let $\sigma(\mathcal{A}) = U^\sigma \mathcal{A}$ for $\mathcal{A}$ with a dimension of $m \times l$. There are at most $2 \cdot \min(m, l) - 1$ non-zero diagonals in U^σ no matter whether the matrix is flattened with a column-major or row-major order.*

Theorem III.2. *Let $\tau(\mathcal{B}) = U^\tau \mathcal{B}$ for $\mathcal{B}$ with a dimension of $l \times n$. There are at most $2 \cdot \min(n, l) - 1$ non-zero diagonals in U^τ no matter if the matrix is flattened with a column-major or row-major order.*

Theorem III.3. *Let $\epsilon_{m \times n}^k(\mathcal{A}) = U^{\epsilon_{m \times n}^k} \mathcal{A}$ be the linear transformation $\epsilon_{m \times n} : \mathcal{R}_{m \times l} \to \mathcal{R}_{m \times n}$ with matrix $\mathcal{A}$ having a dimension of $m \times l$. There are at most $\lfloor \frac{n}{l} \rfloor + 1$ non-zero diagonal vectors in $U^{\epsilon_{m \times n}^k}$ when the matrix is flattened with the **column-major** order; There are at most $(\lfloor \frac{n}{l} \rfloor + 2) \cdot m$ non-zero diagonal vectors in $U^{\epsilon_{m \times n}^k}$ when matrix $\mathcal{A}$ is flattened with the **row-major** order. Specifically, when $n = l$, there are no more than 2 non-zero diagonals in $U^{\epsilon_{m \times n}^k}$, no matter if the matrix is flattened in column-major or row-major order.*

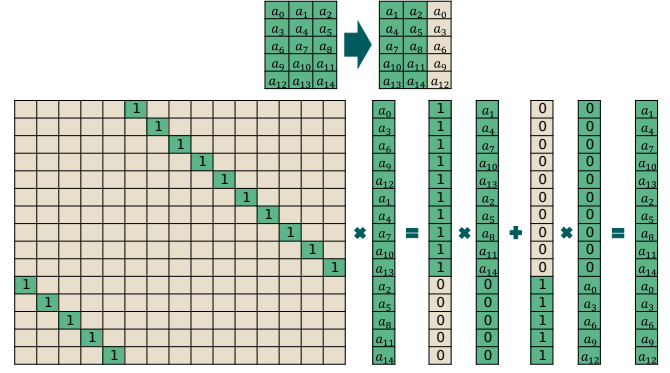**Theorem III.4.** *Let $\omega_{m \times n}^k(\mathcal{B}) = U^{\omega_{m \times n}^k} \mathcal{B}$ be the linear transformation $\omega_{m \times n} : \mathcal{R}_{l \times n} \to \mathcal{R}_{m \times n}$ with matrix $\mathcal{B}$ having a dimension of $l \times n$. There are at most $(\lfloor \frac{m}{l} \rfloor + 2) \cdot n$ non-zero diagonal vectors in $U^{\omega_{m \times n}^k}$ when the matrix is flattened with **column-major** order; There are at most $\lfloor \frac{m}{l} \rfloor + 1$ non-zero diagonal vectors in $U^{\omega_{m \times n}^k}$ when matrix $\mathcal{B}$ is flattened with **row-major** order. Specifically, when $m = l$, there are no more than 2 non-zero diagonals in $U^{\omega_{m \times n}^k}$, no matter if the matrix is flattened in column-major or row-major order.*

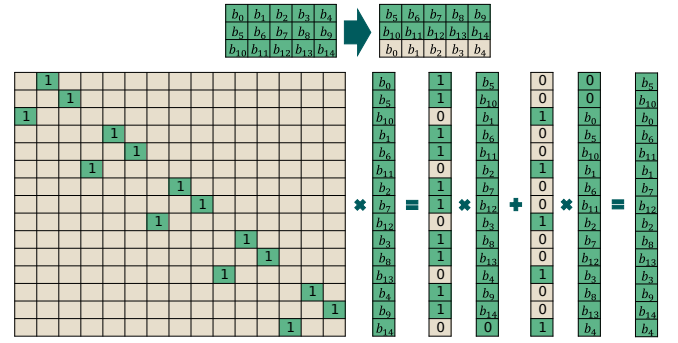The proofs for Theorems III.1-III.4 can be found in the appendix A.

According to Theorems III.1 and III.2, the numbers of non-zero diagonal vectors in U^σ and U^τ depend solely on the dimensions of corresponding matrices and are independent of how the matrices are flattened. However, as shown in Theorems III.3 and III.4, the numbers of non-zero diagonal vectors in $U^{\epsilon_{m \times n}^k}$ and $U^{\omega_{m \times n}^k}$ depend on not only the dimensions of matrices but also the way they are flattened.

2) The HEGMM Algorithm: HEGMM is a straightforward implementation of Equation (5). We first conduct σ and τ transformations (lines 2-3) on the source matrices of $\mathcal{A}$ and $\mathcal{B}$. We then go through a loop (lines 5-9) that apply $\epsilon_{m \times n}^k$ and $\tau_{m \times n}^k$ transformations and element-wise multiplication and addition to calculate and accumulate the partial product. The final result can be obtained by decrypting the sum of the product (line 11).

The computational complexity of Algorithm 1 mainly comes from the required HE operations associated with the $\sigma, \tau, \epsilon_{m \times n}^k$, and $\omega_{m \times n}^k$ operations. Assuming $\mathcal{A}$ and $\mathcal{B}$ are encrypted, from Theorem III.1 and Theorem III.2, we know that there are $2 \min(m, l) - 1$ (resp. $2 \min(n, l) - 1$) non-diagonals for the σ (resp. τ) operation. Therefore, according to equation (7), the σ (resp. τ) operation requires $2 \min(m, l) - 1$ (resp. $2 \min(n, l) - 1$) HE-CMult, HR-Rot, and HR-Add

(a) The process of linear transformation for $\epsilon_{5 \times 3}^1$ transformation on matrix $\mathcal{A}_{5 \times 3}$. u_5 is the vector with 10 1's and 5 0's while u_{-10} with 10 0's and 5 1's. Therefore, according to Equation 7, it rotates flattened $\tilde{\mathcal{A}}$ 5 slots and times u_5. Then it rotates flattened $\tilde{\mathcal{A}}$ 10 slots reversely and times u_{-10}. Finally, add all above partial products together.

(b) The process of linear transformation for $\omega_{3 \times 5}^1$ transformation on matrix $\mathcal{B}_{3 \times 5}$. u_1 is the vector with ten 1's and five 0's while u_{-2} with 10 0's and 5 1's. Therefore, according to Equation 7, it rotates flattened $\tilde{\mathcal{B}}$ 1 slots and times u_1. Then it rotates flattened $\tilde{\mathcal{A}}$ 2 slots reversely and times u_{-2}. Finally, add all above partial products together.

Fig. 3. The permutation matrices $U^{\epsilon_{5 \times 3}^1}$ and $U^{\omega_{3 \times 5}^1}$ and linear transformations of $\epsilon_{5 \times 3}^1(\mathcal{A}_{5 \times 3})$ and $\omega_{3 \times 5}^1(\mathcal{B}_{5 \times 3})$.

Algorithm 1: HEGMM: HE-based General Matrix Multiplication

Input: matrix $\mathcal{A}_{m \times l}$ and matrix $\mathcal{B}_{l \times n}$
Output: $\mathcal{C}_{m \times n}$

1 **[Step1]**
2 $ct.\mathcal{A}^{(0)} \leftarrow \sigma(\mathcal{A})$
3 $ct.\mathcal{B}^{(0)} \leftarrow \tau(\mathcal{B})$
4 **[Step2]**
5 **for** $k = 0$ *to* $l - 1$ **do**
6 $\quad ct.\mathcal{A}^{(k)} \leftarrow \epsilon_{m \times n}^k(ct.\mathcal{A}^{(0)})$
7 $\quad ct.\mathcal{B}^{(k)} \leftarrow \omega_{m \times n}^k(ct.\mathcal{B}^{(0)})$
8 $\quad ct.\mathcal{C} \leftarrow ct.\mathcal{C} + ct.\mathcal{A}^{(k)} \odot ct.\mathcal{B}^{(k)}$
9 **end**
10 **[Step3]**
11 $\mathcal{C}_{m \times n} \leftarrow ct.\mathcal{C}$
12 **return** $\mathcal{C}_{m \times n}$

---

operations. These computational costs have nothing to do

with how the matrices are flattened (e.g., in column-major or in row-major order), and they also become trivial if they are performed on $\mathcal{A}$ and $\mathcal{B}$ in plaintext. However, the ϵ and ω operations, which must be performed multiple times in the cloud, require HE operations depending on not only the dimensions of matrices but also the way they are flattened. As a result, the computational complexities can be dramatically different under different scenarios, as shown in Theorems III.3 and III.4. In the next sub-section, we show how we can take advantage of this fact to reduce the computational cost effectively.

C. The Enhanced HEGMM Algorithm

In this section, we introduce a more elaborated approach for HEGMM that can be more computationally efficient. The fundamental principle we rely on to develop this algorithm is presented in Theorem III.3 and III.4. For the HE matrix multiplication of $\mathcal{A}_{m \times l} \times \mathcal{B}_{l \times n}$, the proposed new algorithm can significantly improve the computation efficiency when $m = \min\{m, l, n\}$ and/or $n = \min\{m, l, n\}$. If not, we can always resort to Algorithm 1 to find the solution. Therefore, in what follows, we first discuss the new approach based on two cases: (i) $m = \min\{m, l, n\}$; and (ii) $n = \min\{m, l, n\}$. We then present the algorithm and related discussions in detail.

1) $m = \min\{m, l, n\}$: For the HE MM of $\mathcal{A}_{m \times l} \times \mathcal{B}_{l \times n}$, Algorithm 1 needs to perform l iterations, with each iteration including one ϵ transformation, one ω transformation, one HE-Add, and one HE-Mult operation. Assuming the matrix is flattened with the ***column-major*** order, according to Theorem III.3 and III.4, one ϵ transformation and one ω transformation would result in no more than $((\lfloor \frac{n}{l} \rfloor + 1) + 2n)$ non-zero diagonals in corresponding transformation matrices, with each non-zero diagonal requiring one HE-Add, one HE-Rot, and one HE-CMult operations. However, if we can expand matrix $\mathcal{A}_{m \times l}$ to $\bar{\mathcal{A}}_{l \times l}$, then the number of non-zero diagonals becomes no more than $((\lfloor \frac{n}{l} \rfloor + 1) + 2)$ instead. Since $n \geq 1$ and

$$((\lfloor \frac{n}{l} \rfloor + 1) + 2n) \geq ((\lfloor \frac{n}{l} \rfloor + 1) + 2),$$

the number of non-zero diagonals and, thus, the computational cost can be dramatically reduced.

Note that, if we assume the matrix is flattened with the ***row-major*** order, one ϵ transformation and one ω transformation would result in no more than $((\lfloor \frac{n}{l} \rfloor + 2)m + 1)$ non-zero diagonals in corresponding transformation matrices. When expanding matrix $\mathcal{A}_{m \times l}$ to $\bar{\mathcal{A}}_{l \times l}$, the total number of non-zero diagonals in the corresponding transformation matrices becomes $((\lfloor \frac{n}{l} \rfloor + 2)l + 1)$, according to Theorems III.3 and III.4. It becomes obvious that using the ***column-major*** order is a better choice than using the ***row-major*** order in this case, since when $m > 1$, we have

$$((\lfloor \frac{n}{l} \rfloor + 2)m + 1) \geq ((\lfloor \frac{n}{l} \rfloor + 1) + 2).$$

The question now becomes how to expand $\mathcal{A}_{m \times l}$ to $\bar{\mathcal{A}}_{l \times l}$ and maintain the logical correctness of the MM result. One intuitive approach is to expand $\mathcal{A}_{m \times l}$ by filling zeroes to the
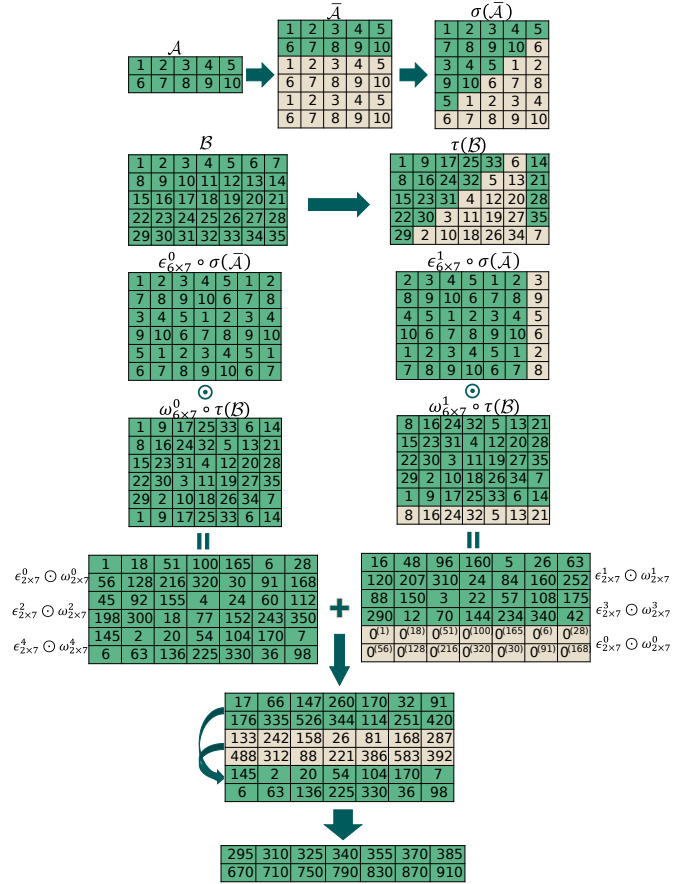

Fig. 4. An illustration example of the Enhanced HEGMM Algorithm for multiplying two matrices $\mathcal{A}_{2 \times 5}$ and $\mathcal{B}_{5 \times 7}$, with $m = 2$, $l = 5$ and $n = 7$. $\bar{\mathcal{A}}$ is the matrix by duplicating $\mathcal{A}$ 3 times, i.e., $t = \lceil 5/2 \rceil = 3$ and $\mathcal{B}_{5 \times 7}$ remains unchanged. The partial products are accumulated to obtain the final product. Note $\epsilon^0_{2 \times 7}(\sigma(\mathcal{A})) \odot \omega^0_{2 \times 7}(\tau(\mathcal{B}))$ is generated twice, and the duplicated partial products should be excluded from the final results.

newly added elements, i.e., the elements in rows from row m to $(l - m - 1)$. The final product, as a sub matrix, can be extracted from the product of $\bar{\mathcal{A}}_{l \times l} \times \mathcal{B}_{l \times n}$ easily. Note that expanding the matrix dimension does not increase the computational complexity in the SIMD scheme as long as the result matrix can fit in one message.

Rather than simply filling zeroes, we can expand $\mathcal{A}_{m \times l}$ by duplicating the rows of $\mathcal{A}_{m \times l}$ repeatedly. This helps to reduce the number of iterations (lines 5-9) in Algorithm 1, thanks to our observations that are formulated in the following theorem.

Theorem III.5. *Let $\mathcal{A}_{m \times l}$ and $\mathcal{B}_{l \times n}$ with $m < l$, and let $\bar{A}$ be matrix expanded with $t = \lceil \frac{l}{m} \rceil$ copies of $\mathcal{A}$ vertically, i.e., $\bar{\mathcal{A}} = \{\bar{A}_0; \bar{A}_1; ...; \bar{A}_{(t-1)}\}^T$ with $\bar{A}_0 = \bar{A}_1 = ... = \bar{A}_{(t-1)} = \mathcal{A}_{m \times l}$. Then*

- $\epsilon^k_{tm \times n}(\sigma(\bar{\mathcal{A}})) \odot \omega^k_{tm \times n}(\tau(\mathcal{B}))$ *contains t items of $\epsilon^p_{m \times n}(\sigma(\mathcal{A})) \odot \omega^p_{m \times n}(\tau(\mathcal{B}))$, with $p \in \{[k]_l, [k + m]_l, ..., [k + (t-1)m]_l\}$.*
- $\epsilon^k_{tm \times n}(\sigma(\bar{\mathcal{A}})) \odot \omega^k_{tm \times n}(\tau(\mathcal{B}))$, $k = 0, 1, ..., (m - 1)$ *contains all items of $\epsilon^p_{m \times n}(\sigma(\mathcal{A})) \odot \omega^p_{m \times n}(\tau(\mathcal{B}))$, with $p \in \{0, 1, ..., (l - 1)\}$.*

According to Theorem III.5, after expanding $\mathcal{A}_{m \times l}$ with t copies of $\mathcal{A}_{m \times l}$ vertically to form $\bar{\mathcal{A}}_{tm \times l}$, each iteration of Algorithm 1 can now produce t partial products $\epsilon^p_{m \times n}(\mathcal{A}) \odot \omega^p_{m \times n}(\mathcal{B})$. As a result, the required HE computations can be greatly reduced, which can be better illustrated using the example in Figure 4.

Figure 4 shows two source matrices $\mathcal{A}_{2 \times 5}$ and $\mathcal{B}_{5 \times 7}$, with $m = 2$, $l = 5$ and $n = 7$. $\bar{\mathcal{A}}$ is the matrix by duplicating $\mathcal{A}$ three times, i.e., $t = \lceil 5/2 \rceil = 3$. Note that, each $\epsilon_{6 \times 7}(\sigma(\bar{\mathcal{A}})) \odot \omega_{6 \times 7}(\tau(\mathcal{B}))$ contains three copies of $\epsilon_{2 \times 7}(\sigma(\mathcal{A})) \odot \omega_{2 \times 7}(\tau(\mathcal{B}))$, as shown in the figure: $\epsilon^0_{6 \times 7}(\sigma(\bar{\mathcal{A}})) \odot \omega^0_{6 \times 7}(\tau(\mathcal{B}))$ contains $\epsilon^0_{2 \times 7}(\sigma(\mathcal{A})) \odot \omega^0_{2 \times 7}(\tau(\mathcal{B}))$, $\epsilon^2_{2 \times 7}(\sigma(\mathcal{A})) \odot \omega^2_{2 \times 7}(\tau(\mathcal{B}))$, and $\epsilon^4_{2 \times 7}(\sigma(\mathcal{A}) \odot \omega^4_{2 \times 7}(\tau(\mathcal{B}))$. We then need to add all the partial products together to get the final result.

As such, by duplicating $\mathcal{A}_{m \times l}$ into $\bar{\mathcal{A}}_{tm \times l}$, we can reduce not only the HE operations associated with the ϵ and ω operations but also the HE-Mult operations (i.e., at most m HE-Mult operations according to Theorem III.5) for partial production calculations, which is highly costly. Even though extra HE rotations are needed to extract the partial results, the computation cost is much smaller than that of HE-Mult as shown in Table I. It is worth mentioning that, while one $\epsilon^k_{m \times n}(\bar{\mathcal{A}}) \odot \omega^k_{m \times n}(\mathcal{B})$ helps to produce multiple copies of $\epsilon^p_{m \times n}(\mathcal{A}) \odot \omega^p_{m \times n}(\mathcal{B})$, as shown in Figure 4, some of them may be produced repeatedly. These redundant copies should be identified, which can be easily identified according to Theorem III.5, and excluded from the final results.

2) $n = \min\{m, l, n\}$: We can employ the same analysis flow as above. There are two major differences compared with the case of $m = \min\{m, l, n\}$. (*i*) We duplicate matrix $\mathcal{B}$ *horizontally* to expand $\mathcal{B}$ instead of $\mathcal{A}$; (*ii*) The ***row-major*** order is a better option than the ***column-major*** order in this case.

When $n = \min\{m, l, n\}$, if the matrix is flattened with the ***row-major*** order, according to Theorem III.3 and III.4, one ϵ transformation and one ω transformation would result in no more than $(2m + \lfloor \frac{m}{l} \rfloor + 1)$ non-zero diagonals in corresponding transformation matrices. When expanding $\mathcal{B}_{l \times n}$ to $\bar{\mathcal{B}}_{l \times l}$, the total number of non-zero diagonals in corresponding transformation matrices is reduced to $(2 + \lfloor \frac{m}{l} \rfloor + 1)$ instead. However, if the ***column-major order*** is used, the total number of non-zero diagonals after expanding is $(1 + (2 + \lfloor \frac{m}{l} \rfloor) \cdot n)$, which makes the ***row-major*** order a better option to flatten the matrices.

Similarly, when we expand $\mathcal{B}$ by duplicating $\mathcal{B}_{l \times n}$, we can generate multiple partial products, i.e., $\epsilon^p_{m \times n}(\mathcal{A}) \odot \omega^p_{m \times n}(\mathcal{B})$, using one HE-Mult operation, as supported by the following theorem. The proof is quite similar to that for Theorem III.5 and thus omitted due to page limit.

Theorem III.6. *Let $\mathcal{A}_{m \times l}$ and $\mathcal{B}_{l \times n}$ with $n < l$, and let $\bar{\mathcal{B}}$ be matrix expanded with $t = \lceil \frac{l}{n} \rceil$ copies of $\mathcal{B}$ horizontally, i.e., $\bar{\mathcal{B}} = \{\mathcal{B}; \mathcal{B}; ...; \mathcal{B}\}$. Then*

- $\epsilon^k_{m \times tn}(\sigma(\mathcal{A})) \odot \omega^k_{m \times tn}(\tau(\bar{\mathcal{B}}))$ *contains t items of $\epsilon^p_{m \times n}(\sigma(\mathcal{A})) \odot \omega^p_{m \times n}(\tau(\mathcal{B}))$, with $p = [k]_l, [k +$*

$n]_l, ..., [k + (t-1)n]_l$;
- $\epsilon^k_{m \times tn}(\sigma(\mathcal{A})) \odot \omega^k_{m \times tn}(\tau(\bar{\mathcal{B}})), \ k = 0, 1, ..., (n-1)$ *contains all items of $\epsilon^p_{m \times n}(\sigma(\mathcal{A})) \odot \omega^p_{m \times n}(\tau(\mathcal{B}))$, with $p = 0, 1, ..., (l-1)$.*
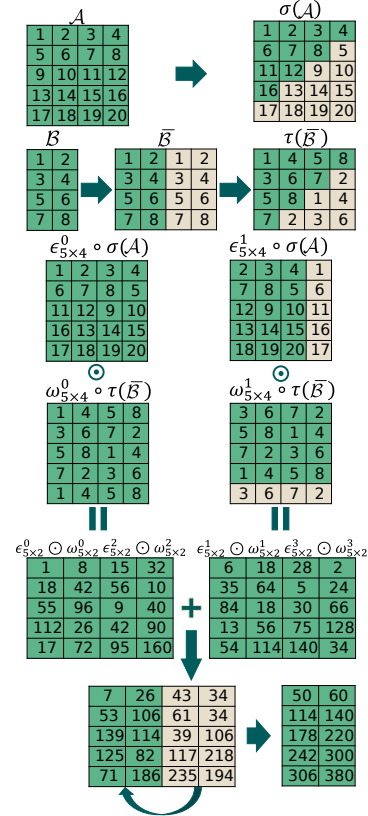

Fig. 5. This is an illustrative example of the enhanced HE MM algorithm for multiplying two matrices $\mathcal{A}_{5 \times 4}$ and $\mathcal{B}_{4 \times 2}$, with $m = 5$, $l = 4$ and $n = 2$. $\bar{\mathcal{B}}$ is the matrix by duplicating $\mathcal{B}$ horizontally for two times, i.e., $t = \lceil 4/2 \rceil = 2$ and $\mathcal{A}_{5 \times 4}$ remains unchanged. The partial products are accumulated to obtain the final product.

Figure 5 shows an illustrative example of HE MM with two source matrices $\mathcal{A}_{5 \times 4}$ and $\mathcal{B}_{4 \times 2}$, with $m = 5$, $l = 4$ and $n = 2$. $\bar{\mathcal{B}}$ is the matrix by duplicating $\mathcal{B}$ horizontally for two times, i.e., $t = \lceil 4/2 \rceil = 2$. Note that, each $\epsilon_{5 \times 4}(\sigma(\mathcal{A})) \odot \omega_{5 \times 4}(\tau(\bar{\mathcal{B}}))$ using one HE-Mult operation can produce two copies of $\epsilon_{5 \times 2}(\sigma(\mathcal{A})) \odot \omega_{5 \times 2}(\tau(\mathcal{B}))$, as shown in the figure: $\epsilon^0_{5 \times 4}(\sigma(\mathcal{A})) \odot \omega^0_{5 \times 4}(\tau(\bar{\mathcal{B}}))$ contains $\epsilon^0_{5 \times 2}(\sigma(\mathcal{A})) \odot \omega^0_{5 \times 2}(\tau(\mathcal{B}))$ and $\epsilon^2_{5 \times 2}(\sigma(\mathcal{A})) \odot \omega^2_{5 \times 2}(\tau(\mathcal{B}))$. $\epsilon^1_{5 \times 4}(\sigma(\mathcal{A})) \odot \omega^1_{5 \times 4}(\sigma(\bar{\mathcal{B}}))$ contains $\epsilon^1_{5 \times 2}(\sigma(\mathcal{A})) \odot \omega^1_{5 \times 2}(\tau(\mathcal{B}))$ and $\epsilon^3_{5 \times 2}(\sigma(\mathcal{A})) \odot \omega^3_{5 \times 2}(\tau(\mathcal{B}))$. We then need to add all the partial products together to get the final result. As such, we only need to perform at most n HE-Mult operations according to Theorem III.6 to obtain all the partial products. Redundant copies may also be generated during this process, which should be identified according to Theorem III.6 and excluded from the final results.

The overall algorithm for the enhanced HE-based General

Algorithm 2: HEGMM-Enhanced

Input: matrix $\mathcal{A}_{m \times l}$, $\mathcal{B}_{l \times n}$
Output: matrix $\mathcal{C}_{m \times n} = \mathcal{A}_{m \times l} \times \mathcal{B}_{l \times n}$

1   $p \leftarrow \min(m, l, n)$
2   $t \leftarrow \lceil l/p \rceil$
    // Determine M and N by shape
3   **if** $p = m$ **then**
4      $\bar{\mathcal{A}}_{M \times l} \leftarrow [\underbrace{\mathcal{A}; \mathcal{A}; \mathcal{A}; ...; \mathcal{A}}_{t}]^T$
5      $\bar{\mathcal{B}}_{l \times N} \leftarrow \mathcal{B}$
6      $M = t \times m, N = n$
7   **else if** $p = n$ **then**
8      $\bar{\mathcal{A}}_{M \times l} \leftarrow \mathcal{A}$
9      $\bar{\mathcal{B}}_{l \times N} \leftarrow [\underbrace{\mathcal{B}; \mathcal{B}; \mathcal{B}; ...; \mathcal{B}}_{t}]$
10     $M = m, N = t \times n$
11   **else**
12     $\bar{\mathcal{A}}_{M \times l} \leftarrow \mathcal{A}$
13     $\bar{\mathcal{B}}_{l \times N} \leftarrow \mathcal{B}$
14     $M = m, N = n$
15   **end**
    // Prepossessing on client
16   $ct.s\bar{\mathcal{A}} \leftarrow \epsilon^0_{M \times \max(l,N)}(\sigma(\bar{\mathcal{A}}))$
17   $ct.t\bar{\mathcal{B}} \leftarrow \omega^0_{\max(l,M) \times N}(\tau(\bar{\mathcal{B}}))$
    // Multiplication on Cloud
18   $ct.\mathcal{C}_{m \times n} \leftarrow ct.s\bar{\mathcal{A}} \odot ct.t\bar{\mathcal{B}}$
19   **for** $k = 0, 1, ..., (p-1)$ **do**
20     $ct.\bar{\mathcal{A}}^{(k)} \leftarrow \epsilon^k_{M \times N}(ct.s\bar{\mathcal{A}})$
21     $ct.\bar{\mathcal{B}}^{(k)} \leftarrow \omega^k_{M \times N}(ct.t\bar{\mathcal{B}})$
22     $ct.\mathcal{C}_{temp} \leftarrow ct.\bar{\mathcal{A}}^{(k)} \odot ct.\bar{\mathcal{B}}^{(k)}$
    // $ct.\mathcal{C}_{temp}$ contains of t items of
    $\epsilon^{k+i \cdot p}_{m \times n}(\sigma(\mathcal{A})) \odot \omega^{k+i \cdot p}_{m \times n}(\tau(\mathcal{B}))(0 \leq i < t)$.
23   **for** $i = 0, 1, ..., (t-1)$ **do**
24       $j = [k + i \cdot p]_l$
25   **if** $\epsilon^j_{m \times n}(\sigma(\mathcal{A})) \odot \omega^j_{m \times n}(\tau(\mathcal{B})) \in ct.\mathcal{C}_{temp}$ has *not been accumulated in* $ct.\mathcal{C}_{m \times n}$ **then**
26         $ct.\mathcal{C}_{m \times n} \leftarrow$
         $ct.\mathcal{C}_{m \times n} + \epsilon^j_{m \times n}(\sigma(\mathcal{A})) \odot \omega^j_{m \times n}(\tau(\mathcal{B}))$
27   **end**
28   **end**
    // Return encrypted result to client
29   **return** $ct.\mathcal{C}_{m \times n}$

MM, named *HEGMM-En*, is presented in Algorithm 2. Note that, when $m < l$ and $n < l$, we can choose to duplicate either $\mathcal{A}$ or $\mathcal{B}$. In Algorithm 2, we choose the smaller of m and n and expand either $\mathcal{A}$ or $\mathcal{B}$ accordingly (lines 3 - 10). When $l = \min\{m, l, n\}$, we make no change of $\mathcal{A}$ and $\mathcal{B}$ (lines 11 - 15). After initializing several relevant variables (lines 16 - 18), Algorithm 2 goes through a loop to compute and accumulate the partial products (lines 19-28). To be more specific, we first conduct ϵ and ω transformations based on the expanded matrix ($\mathcal{A}$ or $\mathcal{B}$) (lines 20 - 21), which are combined

together into $\mathcal{C}_{temp}$ using the element-wise HE multiplication (line 22). The algorithm then extracts the possible t copies of $\epsilon_{m \times n}(\sigma(\mathcal{A})) \odot \omega_{m \times n}(\tau(\mathcal{B}))$ from $\mathcal{C}_{temp}$ and accumulates them to $\mathcal{C}_{m \times n}$, according to Theorems III.5 and III.6, and the redundant copies are excluded from the $\mathcal{C}_{m \times n}$.

Note that, compared with Algorithm 1, Algorithm 2 only needs to perform $p = \min\{m, l, n\}$ loops (line 19) instead of l. We assume that the proper order is adopted when flattening the matrix: When $p = \min\{m, l, n\} = m$, $\mathcal{A}$ is expanded and the column-major order is adopted to flatten matrices; When $p = \min\{m, l, n\} = n$, $\mathcal{B}$ is expanded and the row-major order is adopted to flatten matrices; When $p = \min\{m, l, n\} = l$, neither $\mathcal{A}$ and $\mathcal{B}$ is expanded, and either major order can be adopted to flatten matrices.

IV. EXPERIMENTS

In this section, we evaluate the performance of the two algorithms developed in this paper, i.e., HEGMM and HEGMM-Enhanced, and compare them with the state-of-the-art schemes for HE-based matrix multiplication.

A. Experimental platform

We implemented HEGMM and HEGMM-Enhanced using a Python HE library, named Pyfhel [58] with BFV scheme [76], [75]. We set the HE scheme based on the RLWE (Ring Learning With Errors) [33] assumption over the cyclotomic ring $R_q = \mathbb{Z}_q[X]/(X^N + 1)$ with $N = 2^{12}$. Thus each ciphertext can hold up to $N = 2^{12}$ slots for plaintext values, the largest square matrix that can be accommodated in one ciphertext is thus 64×64.

In our experiments, we studied the following approaches.

- **E2DM-S**, which is presented in [61] on square matrix multiplication. For a general MM $\mathcal{A}_{m \times l} \times \mathcal{B}_{l \times n}$, we can transform $\mathcal{A}_{m \times l}$ and $\mathcal{B}_{l \times n}$ to two square matrices, $\mathcal{A}'_{d \times d}$ and $\mathcal{B}'_{d \times d}$ with $d = \max\{m, l, n\}$ and use this algorithm to calculate the result;
- **E2DM-R**, which is presented in [61] on rectangular matrix multiplication $\mathcal{A}_{r \times d} \times \mathcal{B}_{d \times d}$. For a general MM $\mathcal{A}_{m \times l} \times \mathcal{B}_{l \times n}$, we can expand $\mathcal{A}_{m \times l}$ and/or $\mathcal{B}_{l \times n}$ accordingly and use this algorithm to calculate the result;
- **Huang-MM**, which is introduced in [60] and implemented with Pyfhel [58].
- **HEGMM**, which is shown in Algorithm 1.
- **HEGMM-En**, which is shown in Algorithm 2.

We randomly generated 2000 pairs of matrices, with column and row numbers evenly distributed with $[1, 64]$, as the test cases. Note that, even though Huang *et al.* [60], HEGMM and HEGMM-Enhanced can handle MM with column or row numbers exceeding 64, as long as the total element is no more than 2^{12}, we limited the largest dimension size to 64 so that **E2DM-S** and **E2DM-R** can always apply. We assume that σ and τ transformations of E2DM and HEGMM are performed on plaintext, and to be fair, we assume the portion of Huang *et al.*'s algorithm, i.e., extracting diagonal vector from matrix, is also performed on plaintext. All experiments were conducted

on a server with Intel Xeon Silver 4114 with 10 cores at 2.2GHz.

B. Computational time evaluations

To better understand the performance of the five different algorithms listed above, we categorize the test cases into 5 groups: (1) $m = \min\{m, l, n\}$; (2) $l = \min\{m, l, n\}$; (3) $n = \min\{m, l, n\}$; (4) $l \bmod m = 0$; (5) $m = l = n$ (the square matrix). Note that cases in (4) and (5) are the most favorable ones for **E2DM-R** and **E2DM-S**, respectively. For test cases in each group, execution times were collected for the five different approaches. We use the better ones by **E2DM-S** and **E2DM-R** as the performance that can achieved by **E2DM**. We then calculate the speedups that can be achieved using **HEGMM**, **HEGMM-En** over **E2DM** and **Huang** *et al.*.

The average, median, and maximum speedup for each group, as well as the overall results, are listed in Tables II and III.

As in Table II, **HEGMM** outperforms **Huang** *el al.* in all groups, with a speedup of 1.93 on average and the maximum of over 4.96. Compared with **E2DM**, **HEGMM** can achieve better performance in all cases other than if the matrices are square or when $m = min(m, l, n)$. As shown in Table II, **HEGMM** can achieve a speedup of 3.3 on average with the maximum of over 154.12 over the best of **E2DM**. When source matrices are square, **HEGMM** is equivalent to **E2DM** with slight overhead for taking care of generality of matrices. When $m = min(m, l, n)$, the time complexity of **E2DM-R** is $\mathcal{O}(m)$ while **HEGMM** is $\mathcal{O}(l)$. Therefore, **E2DM-R** can potentially achieve better performance, especially when $m << l$.

The enhanced algorithm, i.e., **HEGMM-En**, can significantly outperform the rest of the approaches for arbitrary HE MM, as shown in Table III. This is because **HEGMM-En** can reduce HE-Mult operations significantly by properly duplicating the source matrices. Specifically, **HEGMM-En** can achieve an average speedup of 4.13 with the maximum of 132.42 over best of **E2DM**, and an average speedup of 4.50 with the maximum of 23.68 over the **Huang** *et al.*. For square matrices, **HEGMM-En** is equivalent to **E2DM** and requires slightly more time than due to the overhead for taking care of the generality of matrices.

We also use Figure 6 to compare the performance of these approaches from a different perspective. Specifically, Figure 6 shows the number of test cases that can achieve speedups between $(0, 1]$, $[1, 2]$, and $(2, +\infty)$ by **HEGMM**, **HEGMM-En** and **Huang** *et al.* over the best results by **E2DM-S** and **E2DM-R**. In a total of 2000 test cases, there were 1324 cases that **HEGMM** outperform both **E2DM-S** and **E2DM-R**, while it is 1805 for **HEGMM-En**, which indicates that **HEGMM-En** performs significantly better than **HEGMM**. For **Huang** *et al.*, only 610 samples outperform **E2DM**. Overall, the experimental findings indicate that the algorithms **HEGMM** and **HEGMM-EN** exhibit a significant performance superiority compared to current methodologies in 66.2% and 90.2% of the samples, respectively.
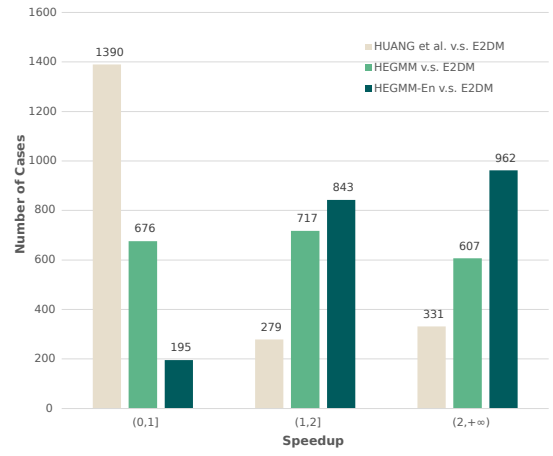

Fig. 6. The statistics of the speedups for the algorithms HEGMM, HEGMM-En, E2DM [61], and Huang *et al.* [60].
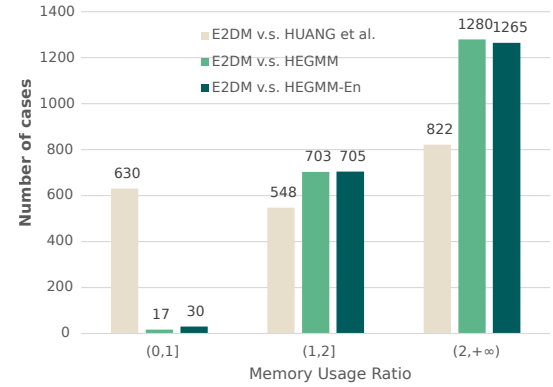

Fig. 7. The statistics of memory usage ratio for the algorithms HEGMM, HEGMM-En, E2DM [61], and Huang *et al.* [60].

C. Memory evaluations

HE computations may demand not only excessive computation time but also memory usage as well. We are therefore interested in studying the memory usage of these approaches. We collected the memory usage for each algorithm during its runtime for our test cases with results normalized against the memory usage by **E2DM** and presented in Figure 7, where a total of 2000 experimental sets were conducted. In comparison to **E2DM**, both **HEGMM** and **HEGMM-En** tend to consume less memory. As shown in Figure 7, less than 17 (resp. 30) out of the total 2000 test results show that **E2DM** consumes less memory than **HEGMM** (resp. **HEGMM-EN**). In contrast, 630 test cases using **Huang** *et al.* have higher memory usage compared to **E2DM**. Overall, the experimental results clearly demonstrate the advantage of memory usage efficiency of **HEGMM** and **HEGMM-En** over the existing approaches.

D. Evaluation of large matrix multiplication

Our test cases above are limited to the maximum matrix dimension of 64x64, the largest one that can fit into one ciphertext in our setting. When matrix sizes exceed this limit, we

TABLE II
The performance comparison of HEGMM, E2DM [61] and Huang *et al.* [60] in different scenarios.

	$m = min(m,l,n)$		$l = min(m,l,n)$		$n = min(m,l,n)$		$l \bmod m = 0$		$square$		overall	
	s-up1[†]	s-up2	s-up1	s-up2	s-up1	s-up2	s-up1	s-up2	s-up1	s-up2	s-up1	s-up2
Average	0.90	2.21	10.74	1.99	1.83	2.29	1.49	2.12	1.00	1.67	3.30	1.93
Median	0.58	2.21	3.61	1.95	1.20	2.35	0.52	2.10	1.00	1.72	1.04	2.04
Max	39.50	3.25	154.12	4.96	136.82	3.28	39.50	3.07	1.02	2.36	154.12	4.96

[†]*s-up1* is the speedup achieved by **HEGMM** over the best of **E2DM** [61]; *s-up2* is the speedup achieved by **HEGMM** over **Huang *et al.* [60]**;

TABLE III
The performance comparison of HEGMM-En, E2DM [61] and Huang *et al.* [60] in different scenarios.

	$m = min(m,l,n)$		$l = min(m,l,n)$		$n = min(m,l,n)$		$l \bmod m = 0$		$square$		overall	
	s-up3[‡]	s-up4	s-up3	s-up4	s-up3	s-up4	s-up3	s-up4	s-up3	s-up4	s-up3	s-up4
Average	1.69	6.60	10.32	2.01	4.06	6.55	2.74	7.28	0.99	1.66	4.13	4.50
Median	1.13	4.88	3.70	1.97	2.56	4.83	1.37	6.23	1.00	1.72	1.38	2.48
Max	33.28	23.29	132.42	4.26	113.31	23.68	33.28	21.75	1.01	2.36	132.42	23.68

[‡]*s-up3* is the speedup achieved by **HEGMM-En** over the best of **E2DM** [61]; *s-up4* is the speedup achieved by **HEGMM-En** over **Huang *et al.* [60]**;

can resort to the traditional *blocking* algorithm, i.e., by dividing a large matrix into a series of smaller blocks, to perform the MM calculation. We want to study the performance of our proposed approaches when incorporated into MM blocking algorithms for $\mathcal{A}_{100 \times 100} \times \mathcal{B}_{100 \times 100}$.

Partitioning large source matrices properly based on different MM algorithms is an interesting problem but beyond the scope of this paper. In our experiments, we hire two intuitive partition methods: **P1:** partitioning the matrix 100×100 to four equal-size square matrices of 50×50; **P2:** partitioning the matrix 100×100 to four sub matrices of 64×64, 64×36, 36×64, and 36×36.

Different HE MM algorithms were employed for blocking MMs. We ran the experiments 10 times, and the average results were collected and shown in Table IV. As expected, for **P1** when all matrices are square, **E2DM-S**, **E2DM-R**, **HEGMM** and **HEGMM-EN** perform quite similarly, while **HEGMM** and **HEGMM-EN** take a little longer due to overhead in dealing with the generality of the matrices. **Huang *et al.*** shows a much slower performance than the others. We believe this is because that **Huang *et al.*** approach requires duplicating diagonals of a source matrix with the complexity of $O(logN)$, with N the size of the matrix. The duplication operation involves expensive *HE-CMult* and *HE-Rot* operations. This is particularly computationally expensive when N is not a power of two. In contrast, the time complexity of same step in **E2DM** and **HEGMM** is $O(2)$ for **P1**.

For **P2**, **HEGMM**, **HEGMM-EN**, and **Huang *et al.*** can perform better because they can take advantage of the irregular shapes of the matrices. In particular, **HEGMM-EN** (resp. **HEGMM**) has a complexity of $O(\min(m,l,n))$ (resp. $O(l)$). In contrast, **E2DM-S** runs much longer because it needs to expand matrices 64×36 and 36×64 to form 64×64 matrix. **E2DM-R** is incapable of processing matrices with such irregular shapes, as it has a tendency to enlarge matrix of 36×64 to 72×72, which is larger than the ciphertext size.

TABLE IV
Time evaluation of the blocking algorithm

Partition	E2DM-S	E2DM-R	Huang *et al.*	HEGMM	HEGMM-EN
P1	39.06s	**39.01s**	74.34s	39.12s	39.15s
P2	29.76s	N/A	37.51s	**26.17s**	26.23s

V. Conclusions

HE has great potential for security and privacy protection when outsourcing data processing to the cloud. However, the excessive computational overhead associated with the HE operations makes it prohibitive for many practical cloud applications. We study how to reduce the HE computational cost for general MM operation, an essential building block in many computational fields. We present two HE MM algorithms, with one improving another, to reduce the computational complexity of MM by taking advantage of the SIMD structure in the HE scheme. We also conduct rigorous analytical studies on the correctness and computational complexity of these two algorithms. Experiment results show that our proposed approach can significantly outperform the existing methods. In our future research, we plan to investigate how to reduce the HE computational cost for sparse matrix multiplication.

REFERENCES

[1] D. H. Duong, P. K. Mishra, and M. Yasuda, "Efficient secure matrix multiplication over lwe-based homomorphic encryption," *Tatra Mountains Mathematical Publications*, vol. 67, no. 1, pp. 69–83, 2017. [Online]. Available: https://doi.org/10.1515/tmmp-2016-0031

[2] P. K. Mishra, D. H. Duong, and M. Yasuda, "Enhancement for secure multiple matrix multiplications over ring-lwe homomorphic encryption," in *Information Security Practice and Experience*, J. K. Liu and P. Samarati, Eds. Springer, 2017, pp. 320–330.

[3] B. Varghese and R. Buyya, "Next generation cloud computing: New trends and research directions," *Future Generation Computer Systems*, vol. 79, pp. 849–861, 2018.

[4] J. H. Cheon, A. Kim, and D. Yhee, "Multi-dimensional packing for heaan for approximate matrix arithmetics," *Cryptology ePrint Archive*, 2018.

[5] M. J. Atallah, K. N. Pantazopoulos, J. R. Rice, and E. E. Spafford, "Secure outsourcing of scientific computations," in *Advances in Computers*. Elsevier, 2002, vol. 54, pp. 215–272.

[6] X. Lei, X. Liao, T. Huang, and F. Heriniaina, "Achieving security, robust cheating resistance, and high-efficiency for outsourcing large matrix multiplication computation to a malicious cloud," *Information sciences*, vol. 280, pp. 205–217, 2014.

[7] S. Fu, Y. Yu, and M. Xu, "A secure algorithm for outsourcing matrix multiplication computation in the cloud," in *Proceedings of the Fifth ACM international workshop on security in cloud computing*, 2017, pp. 27–33.

[8] S. Zhang, C. Tian, H. Zhang, J. Yu, and F. Li, "Practical and secure outsourcing algorithms of matrix operations based on a novel matrix encryption method," *IEEE Access*, vol. 7, pp. 53 823–53 838, 2019.

[9] P. K. Mishra, D. Rathee, D. H. Duong, and M. Yasuda, "Fast secure matrix multiplications over ring-based homomorphic encryption," *Information Security Journal: A Global Perspective*, vol. 30, no. 4, pp. 219–234, 2021.

[10] S. Wang and H. Huang, "Secure outsourced computation of multiple matrix multiplication based on fully homomorphic encryption," *KSII Transactions on Internet and Information Systems (TIIS)*, vol. 13, no. 11, pp. 5616–5630, 2019.

[11] J. H. Cheon and A. Kim, "Homomorphic encryption for approximate matrix arithmetic," *Cryptology ePrint Archive*, 2018.

[12] Y. Tian, M. Al-Rodhaan, B. Song, A. Al-Dhelaan, and T. H. Ma, "Somewhat homomorphic cryptography for matrix multiplication using gpu acceleration," in *2014 International Symposium on Biometrics and Security Technologies (ISBAST)*. IEEE, 2014, pp. 166–170.

[13] E. Hesamifard, H. Takabi, M. Ghasemi, and R. N. Wright, "Privacy-preserving machine learning as a service." *Proc. Priv. Enhancing Technol.*, vol. 2018, no. 3, pp. 123–142, 2018.

[14] R. Hiromasa, M. Abe, and T. Okamoto, "Packing messages and optimizing bootstrapping in gsw-fhe," *IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 99, no. 1, pp. 73–82, 2016.

[15] R. Scale, "State of the cloud report," Tech. Rep, Tech. Rep., 2015.

[16] E. Kushilevitz and R. Ostrovsky, "Replication is not needed: Single database, computationally-private information retrieval," in *Proceedings 38th annual symposium on foundations of computer science*. IEEE, 1997, pp. 364–373.

[17] J. D. C. Benaloh, *Verifiable secret-ballot elections*. Yale University, 1987.

[18] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[19] S. Goldwasser and S. Micali, "Probabilistic encryption & how to play mental poker keeping secret all partial information," in *Proceedings of the fourteenth annual ACM symposium on Theory of computing*, 1982, pp. 365–377.

[20] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE transactions on information theory*, vol. 31, no. 4, pp. 469–472, 1985.

[21] J. Benaloh and D. Tuinstra, "Receipt-free secret-ballot elections," in *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, 1994, pp. 544–553.

[22] D. Naccache and J. Stern, "A new public key cryptosystem based on higher residues," in *Proceedings of the 5th ACM Conference on Computer and Communications Security*, 1998, pp. 59–66.

[23] T. Okamoto and S. Uchiyama, "A new public-key cryptosystem as secure as factoring," in *International conference on the theory and applications of cryptographic techniques*. Springer, 1998, pp. 308–318.

[24] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *International conference on the theory and applications of cryptographic techniques*. Springer, 1999, pp. 223–238.

[25] I. Damgård and M. Jurik, "A generalisation, a simplification and some applications of paillier's probabilistic public-key system," in *International workshop on public key cryptography*. Springer, 2001, pp. 119–136.

[26] A. Kawachi, K. Tanaka, and K. Xagawa, "Multi-bit cryptosystems based on lattice problems," in *International Workshop on Public Key Cryptography*. Springer, 2007, pp. 315–329.

[27] S. C. U. M. B. Tackmann, "Constructing confidential channels from authenticated channels—public-key encryption revisited."

[28] D. Boneh, E.-J. Goh, and K. Nissim, "Evaluating 2-dnf formulas on ciphertexts," in *Theory of cryptography conference*. Springer, 2005, pp. 325–341.

[29] T. Sander, A. Young, and M. Yung, "Non-interactive cryptocomputing for nc/sup 1," in *40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039)*. IEEE, 1999, pp. 554–566.

[30] A. López-Alt, E. Tromer, and V. Vaikuntanathan, "On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption," in *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, 2012, pp. 1219–1234.

[31] Z. Brakerski and V. Vaikuntanathan, "Fully homomorphic encryption from ring-lwe and security for key dependent messages," in *Annual cryptology conference*. Springer, 2011, pp. 505–524.

[32] S. Ames, M. Venkitasubramaniam, A. Page, O. Kocabas, and T. Soyata, "Secure health monitoring in the cloud using homomorphic encryption: A branching-program formulation," in *Enabling Real-Time Mobile Cloud Computing through Emerging Technologies*.

[33] V. Lyubashevsky, C. Peikert, and O. Regev, "On ideal lattices and learning with errors over rings," in *29th Intl. Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2010, pp. 1–23.

[34] B. Reagen, W.-S. Choi, Y. Ko, V. T. Lee, H.-H. S. Lee, G.-Y. Wei, and D. Brooks, "Cheetah: Optimizing and accelerating homomorphic encryption for private inference," in *IEEE International Symposium on High-Performance Computer Architecture (HPCA)*.

[35] M. Nocker, D. Drexel, M. Rader, A. Montuoro, and P. Schöttle, "Heman–homomorphically encrypted machine learning with onnx models," *arXiv preprint arXiv:2302.08260*, 2023.

[36] M. Van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, "Fully homomorphic encryption over the integers," in *Annual international conference on the theory and applications of cryptographic techniques*. Springer, 2010, pp. 24–43.

[37] Y. Ishai and A. Paskin, "Evaluating branching programs on encrypted data," in *Theory of Cryptography Conference*. Springer, 2007, pp. 575–594.

[38] A. Acar, H. Aksu, A. S. Uluagac, and M. Conti, "A survey on homomorphic encryption schemes: Theory and implementation," *ACM Computing Surveys (Csur)*, vol. 51, no. 4, pp. 1–35, 2018.

[39] M. Ghobaei-Arani, S. Jabbehdari, and M. A. Pourmina, "An autonomic resource provisioning approach for service-based cloud applications: A hybrid approach," *Future Generation Computer Systems*, vol. 78, pp. 191–210, 2018.

[40] V. R. Pancholi and B. P. Patel, "Enhancement of cloud computing security with secure data storage using aes," *International Journal for Innovative Research in Science and Technology*, vol. 2, no. 9, pp. 18–21, 2016.

[41] V. Rajaraman, "Cloud computing," *Resonance*, vol. 19, no. 3, pp. 242–258, 2014.

[42] B. Power and J. Weinman, "Revenue growth is the primary benefit of the cloud," *IEEE Cloud Computing*, vol. 5, no. 4, pp. 89–94, 2018.

[43] S. Becker, G. Brataas, M. Cecowski, D. Huljenić, S. Lehrig, and I. Stupar, "The cloudscale method for managers," in *Engineering Scalable, Elastic, and Cost-Efficient Cloud Computing Applications*. Springer, 2017, pp. 149–165.

[44] A. Fawzi, M. Balog, A. Huang, T. Hubert, B. Romera-Paredes, M. Barekatain, A. Novikov, F. J. R Ruiz, J. Schrittwieser, G. Swirszcz *et al.*, "Discovering faster matrix multiplication algorithms with reinforcement learning," *Nature*, vol. 610, no. 7930, pp. 47–53, 2022.

[45] F. Liu, J. Tong, J. Mao, R. Bohn, J. Messina, L. Badger, D. Leaf *et al.*, "Nist cloud computing reference architecture," *NIST special publication*, vol. 500, no. 2011, pp. 1–28, 2011.

[46] P. Jiang, C. Hong, and G. Agrawal, "A novel data transformation and execution strategy for accelerating sparse matrix multiplication on gpus," in *Proceedings of the 25th ACM SIGPLAN symposium on principles and practice of parallel programming*, 2020, pp. 376–388.

[47] P. Valero-Lara, I. Martínez-Pérez, S. Mateo, R. Sirvent, V. Beltran, X. Martorell, and J. Labarta, "Variable batched dgemm," in *2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP)*, 2018, pp. 363–367.

[48] I. Masliah, A. Abdelfattah, A. Haidar, S. Tomov, M. Baboulin, J. Falcou, and J. Dongarra, "Algorithms and optimization techniques for high-performance matrix-matrix multiplications of very small matrices," *Parallel Computing*, vol. 81, pp. 1–21, 2019.

[49] W. Liu and B. Vinter, "An efficient gpu general sparse matrix-matrix multiplication for irregular data," in *IEEE 28th international parallel and distributed processing symposium*. IEEE, 2014, pp. 370–381.

[50] Y. Nagasaka, S. Matsuoka, A. Azad, and A. Buluç, "High-performance sparse matrix-matrix products on intel knl and multicore architectures," in *Proceedings of the 47th International Conference on Parallel Processing Companion*, 2018, pp. 1–10.

[51] Z. Zhang, H. Wang, S. Han, and W. J. Dally, "Sparch: Efficient architecture for sparse matrix multiplication," in *2020 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 2020, pp. 261–274.

[52] R. Ran, N. Xu, W. Wang, Q. Gang, J. Yin, and W. Wen, "Cryptogcn: Fast and scalable homomorphically encrypted graph convolutional network inference," *arXiv preprint arXiv:2209.11904*, 2022.

[53] A. Patra, T. Schneider, A. Suresh, and H. Yalame, "{ABY2. 0}: Improved {Mixed-Protocol} secure {Two-Party} computation," in *30th USENIX Security Symposium (USENIX Security 21)*, 2021, pp. 2165–2182.

[54] J. I. Choi, D. Tian, G. Hernandez, C. Patton, B. Mood, T. Shrimpton, K. R. Butler, and P. Traynor, "A hybrid approach to secure function evaluation using sgx," in *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security*, 2019, pp. 100–113.

[55] N. Husted, S. Myers, A. Shelat, and P. Grubbs, "Gpu and cpu parallelization of honest-but-curious secure two-party computation," in *Proceedings of the 29th Annual Computer Security Applications Conference*, 2013, pp. 169–178.

[56] Y. Zhang, A. Steele, and M. Blanton, "Picco: a general-purpose compiler for private distributed computation," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, 2013, pp. 813–826.

[57] T. Vasiljeva, S. Shaikhulina, and K. Kreslins, "Cloud computing: Business perspectives, benefits and challenges for small and medium enterprises (case of latvia)," *Procedia Engineering*, vol. 178, pp. 443–451, 2017.

[58] A. Ibarrondo and A. Viand, "Pyfhel: Python for homomorphic encryption libraries," in *Proceedings of the 9th on Workshop on Encrypted Computing & Applied Homomorphic Cryptography*, 2021, pp. 11–16.

[59] Z. Huang, C. Hong, C. Weng, W.-j. Lu, and H. Qu, "More efficient secure matrix multiplication for unbalanced recommender systems," *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 1, pp. 551–562, 2023.

[60] H. Huang and H. Zong, "Secure matrix multiplication based on fully homomorphic encryption," *Journal of Supercomputing*, pp. 1–22, 2022.

[61] X. Jiang, M. Kim, K. Lauter, and Y. Song, "Secure outsourced matrix computation and application to neural networks," in *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*, 2018, pp. 1209–1222.

[62] V. Gupta, S. Wang, T. Courtade, and K. Ramchandran, "Oversketch: Approximate matrix multiplication for the cloud," in *2018 IEEE International Conference on Big Data (Big Data)*. IEEE, 2018, pp. 298–304.

[63] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of cryptography conference*. Springer, 2006, pp. 265–284.

[64] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy," *Foundations and Trends® in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[65] C. Dwork, "A firm foundation for private data analysis," *Communications of the ACM*, vol. 54, no. 1, pp. 86–95, 2011.

[66] S. Halevi and V. Shoup, "Algorithms in helib," in *Annual Cryptology Conference*. Springer, 2014, pp. 554–571.

[67] N. P. Smart and F. Vercauteren, "Fully homomorphic simd operations," *Designs, codes and cryptography*, vol. 71, no. 1, pp. 57–81, 2014.

[68] D. Rathee, P. K. Mishra, and M. Yasuda, "Faster pca and linear regression through hypercubes in helib," in *Proceedings of the 2018 Workshop on Privacy in the Electronic Society*, 2018, pp. 42–53.

[69] A. C. Yao, "Protocols for secure computations," in *23rd annual symposium on foundations of computer science (sfcs 1982)*. IEEE, 1982, pp. 160–164.

[70] W.-j. Lu, S. Kawasaki, and J. Sakuma, "Using fully homomorphic encryption for statistical analysis of categorical, ordinal and numerical data," *Cryptology ePrint Archive*, 2016.

[71] M. Yasuda, T. Shimoyama, J. Kogure, K. Yokoyama, and T. Koshiba, "New packing method in somewhat homomorphic encryption and its applications," *Security and Communication Networks*, vol. 8, no. 13, pp. 2194–2213, 2015.

[72] M. Naehrig, K. Lauter, and V. Vaikuntanathan, "Can homomorphic encryption be practical?" in *Proceedings of the 3rd ACM workshop on Cloud computing security workshop*, 2011, pp. 113–124.

[73] R. L. Rivest, L. Adleman, M. L. Dertouzos *et al.*, "On data banks and privacy homomorphisms," *Foundations of secure computation*, vol. 4, no. 11, pp. 169–180, 1978.

[74] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the forty-first annual ACM symposium on Theory of computing*, 2009, pp. 169–178.

[75] J. Fan and F. Vercauteren, "Somewhat practical fully homomorphic encryption," *Cryptology ePrint Archive*, 2012.

[76] Z. Brakerski, "Fully homomorphic encryption without modulus switching from classical gapsvp," in *Annual Cryptology Conference*. Springer, 2012, pp. 868–886.

[77] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(leveled) fully homomorphic encryption without bootstrapping," *ACM Transactions on Computation Theory (TOCT)*, vol. 6, no. 3, pp. 1–36, 2014.

[78] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *International conference on the theory and application of cryptology and information security*. Springer, 2017, pp. 409–437.

[79] M. Albrecht, M. Chase, H. Chen, J. Ding, S. Goldwasser, S. Gorbunov, S. Halevi, J. Hoffstein, K. Laine, K. Lauter *et al.*, "Homomorphic encryption standard," in *Protecting Privacy through Homomorphic Encryption*. Springer, 2021, pp. 31–62.

[80] Inferati, "Introduction to the bfv encryption scheme," https://inferati.com/blog/fhe-schemes-bfv, accessed Oct 4, 2022.

[81] Wikipedia contributors, "Single instruction, multiple data — Wikipedia, the free encyclopedia," 2022, [Online; accessed 4-October-2022]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Single_instruction,_multiple_data&oldid=1112117357

[82] C. Gentry and S. Halevi, "Implementing gentry's fully-homomorphic encryption scheme," in *Annual international conference on the theory and applications of cryptographic techniques*. Springer, 2011, pp. 129–148.

[83] N. P. Smart and F. Vercauteren, "Fully homomorphic encryption with relatively small key and ciphertext sizes," in *International Workshop on Public Key Cryptography*. Springer, 2010, pp. 420–443.

[84] S. Halevi and V. Shoup, "Bootstrapping for helib," *Journal of Cryptology*, vol. 34, no. 1, pp. 1–44, 2021.

THE PROOF FOR THE THEOREM III.1∼THEOREM III.4

Theorem. III.1 *Let $\sigma(\mathcal{A}) = \boldsymbol{U}^\sigma \mathcal{A}$ for $\mathcal{A}$ with a dimension of $m \times l$. There are at most $2 \cdot \min(m,l) - 1$ non-zero diagonals in $\boldsymbol{U}^\sigma$ no matter if the matrix is flattened with a column-major or row-major order.*

Proof. When applying σ transformation on matrix $\mathcal{A}_{m \times l}$ in **column-major** order, $\boldsymbol{U}^\sigma$ is formulated in Equation (12). Note that $\boldsymbol{U}^\sigma_{i+j \cdot m,h} = 1$ when $h = i + [i+j]_l \cdot m$ and, for all elements of $\boldsymbol{U}^\sigma_{i+j \cdot m,h}$ that belong to the same diagonal, we have $h - (i + j \cdot m)$ as a constant.

Considering all the non-zero elements in $\boldsymbol{U}^\sigma_{i+j \cdot m,h}$, we have

$$
\begin{aligned}
h - (i + j \cdot m) &= i + [i+j]_l \cdot m - (i + j \cdot m) \\
&= i + (i + j - \left\lfloor \frac{i+j}{l} \right\rfloor \cdot l) \cdot m - (i + j \cdot m) \\
&= (i - \left\lfloor \frac{i+j}{l} \right\rfloor \cdot l) \cdot m.
\end{aligned}
$$

Since $\left\lfloor \frac{i}{l} \right\rfloor + \left\lfloor \frac{j}{l} \right\rfloor \le \left\lfloor \frac{i+j}{l} \right\rfloor \le \left\lfloor \frac{i}{l} \right\rfloor + \left\lfloor \frac{j}{l} \right\rfloor + 1$ and $0 \le j < l$, we have $\left\lfloor \frac{i}{l} \right\rfloor \le \left\lfloor \frac{i+j}{l} \right\rfloor \le \left\lfloor \frac{i}{l} \right\rfloor + 1$.

Now consider two different scenarios: 1) $m < l$; 2) $m \ge l$. When $m < l$, for each $i = \{1, 2, ..., m-1\}$, $h - (i + j \cdot m)$ can at most take two constant values since $\left\lfloor \frac{i}{l} \right\rfloor = 0$ and $0 \le \left\lfloor \frac{i+j}{l} \right\rfloor \le 1$. When $i = 0$, $h - (i + j \cdot m)$ can only be zero since $\left\lfloor \frac{i+j}{l} \right\rfloor = 0$. Therefore, $\boldsymbol{U}^\sigma_{i+j \cdot m,h}$ has at most $2m - 1$ non-zero diagonals under this case.

When $m \ge l$, we have

$$
\begin{aligned}
h - (i + j \cdot m) &= (i - \left\lfloor \frac{i+j}{l} \right\rfloor \cdot l) \cdot m \\
&= (\left\lfloor \frac{i}{l} \right\rfloor \cdot l + p - \left\lfloor \frac{i+j}{l} \right\rfloor \cdot l) \cdot m,
\end{aligned}
$$

with $0 \le p < l$. Since $-1 \le (\left\lfloor \frac{i}{l} \right\rfloor - \left\lfloor \frac{i+j}{l} \right\rfloor) \le 0$, $\boldsymbol{U}^\sigma_{i+j \cdot m,h}$ has at most $2l - 1$ non-zero diagonals under this case.

Therefore, in summary, there are at most $2 \cdot \min(m,l) - 1$ non-zero diagonals in $\boldsymbol{U}^\sigma$ when the matrix is flattened with a column-major. Similar proof can be obtained when the matrix is flattened with the row-major order. $\square$

Theorem. III.2 *Let $\tau(\mathcal{B}) = \boldsymbol{U}^\tau \mathcal{B}$ for $\mathcal{B}$ with a dimension of $l \times n$. There are at most $2 \cdot \min(n,l) - 1$ non-zero diagonals in $\boldsymbol{U}^\tau$ no matter if the matrix is flattened with a column-major or row-major order.*

Proof. When applying τ transformation on matrix $\mathcal{B}_{l \times n}$ in **column-major** order, $\boldsymbol{U}^\tau$ is formulated in Equation (13). Note that $\boldsymbol{U}^\tau_{i+j \cdot l,h} = 1$ when $h = [i+j]_l + j \cdot l$ and, for all elements of $\boldsymbol{U}^\tau_{i+j \cdot l,h}$ that belong to the same diagonal, we have $h - (i + j \cdot l)$ as a constant.

Considering all the non-zero elements in $\boldsymbol{U}^\tau_{i+j \cdot l,h}$, we have

$$
\begin{aligned}
h - (i + j \cdot l) &= [i+j]_l + j \cdot l - (i + j \cdot m) \\
&= i + j - \left\lfloor \frac{i+j}{l} \right\rfloor \cdot l + j \cdot l - (i + j \cdot m) \\
&= j - \left\lfloor \frac{i+j}{l} \right\rfloor \cdot l.
\end{aligned}
$$

Since $\left\lfloor \frac{i}{l} \right\rfloor + \left\lfloor \frac{j}{l} \right\rfloor \le \left\lfloor \frac{i+j}{l} \right\rfloor \le \left\lfloor \frac{i}{l} \right\rfloor + \left\lfloor \frac{j}{l} \right\rfloor + 1$ and $0 \le i < l$, we have $\left\lfloor \frac{j}{l} \right\rfloor \le \left\lfloor \frac{i+j}{l} \right\rfloor \le \left\lfloor \frac{j}{l} \right\rfloor + 1$.

Now consider two different scenarios: 1) $n < l$; 2) $n \ge l$. When $n < l$, for each $j = \{1, 2, ..., n-1\}$, $h - (i + j \cdot l)$ can at most take two constant values since $\left\lfloor \frac{j}{l} \right\rfloor = 0$ and $0 \le \left\lfloor \frac{i+j}{l} \right\rfloor \le 1$. When $i = 0$, $h - (i + j \cdot l)$ can only be zero since $\left\lfloor \frac{i+j}{l} \right\rfloor = 0$. Therefore, $\boldsymbol{U}^\tau_{i+j \cdot l,h}$ has at most $2n - 1$ non-zero diagonals under this case.

When $n \ge l$, we have

$$
\begin{aligned}
h - (i + j \cdot l) &= j - \left\lfloor \frac{i+j}{l} \right\rfloor \cdot l \\
&= \left\lfloor \frac{j}{l} \right\rfloor \cdot l + p - \left\lfloor \frac{i+j}{l} \right\rfloor \cdot l,
\end{aligned}
$$

with $0 \le p < l$. Since $-1 \le (\left\lfloor \frac{j}{l} \right\rfloor - \left\lfloor \frac{i+j}{l} \right\rfloor) \le 0$, $\boldsymbol{U}^\tau_{i+j \cdot l,h}$ has at most $2l - 1$ non-zero diagonals under this case.

Therefore, in summary, there are at most $2 \cdot \min(n,l) - 1$ non-zero diagonals in $\boldsymbol{U}^\tau$ when the matrix is flattened with a column-major. Similar proof can be obtained when the matrix is flattened with the row-major order. $\square$

Theorem. III.3 *Let $\epsilon^k_{m \times n}(\mathcal{A}) = \boldsymbol{U}^{\epsilon^k_{m \times n}} \mathcal{A}$ be the linear transformation $\epsilon_{m \times n} : \mathcal{R}_{m \times l} \to \mathcal{R}_{m \times n}$ with matrix $\mathcal{A}$ having a dimension of $m \times l$. There are at most $\left\lfloor \frac{n}{l} \right\rfloor + 1$ non-zero diagonal vectors in $\boldsymbol{U}^{\epsilon^k_{m \times n}}$ when the matrix is flattened with the **column-major** order; There are at most $(\left\lfloor \frac{n}{l} \right\rfloor + 2) \cdot m$ non-zero diagonal vectors in $\boldsymbol{U}^{\epsilon^k_{m \times n}}$ when matrix $\mathcal{A}$ is flattened with the **row-major** order. Specifically, when $n = l$, there are no more than 2 non-zero diagonals in $\boldsymbol{U}^{\epsilon^k_{m \times n}}$, no matter if the matrix is flattened in column-major or row-major order.*

Proof. When applying ϵ transformation on matrix $\mathcal{A}_{m \times l}$ in **column-major** order, $\boldsymbol{U}^\epsilon$ is formulated in Equation (14). Note that $\boldsymbol{U}^{\epsilon^k_{m \times n}}_{i,j} = 1$ when $j = [k \cdot m + i]_{m \cdot l}$ and, for all elements of $\boldsymbol{U}^{\epsilon^k_{m \times n}}_{i,j}$ that belong to the same diagonal, we have $j - i$ as a constant.

Considering all the non-zero elements in $\boldsymbol{U}^{\epsilon^k_{m \times n}}_{i,j}$, we have

$$
\begin{aligned}
j - i &= [k \cdot m + i]_{m \cdot l} - i \\
&= k \cdot m + i - \left\lfloor \frac{k \cdot m + i}{m \cdot l} \right\rfloor \cdot m \cdot l - i \\
&= k \cdot m - \left\lfloor \frac{k \cdot m + i}{m \cdot l} \right\rfloor \cdot m \cdot l
\end{aligned}
$$

Since $\max(k) = l - 1$ and $\max(i) = m \cdot n - 1$, we have

$$
\begin{aligned}
\max(\frac{k \cdot m + i}{m \cdot l}) &< \frac{l - 1 + n}{l} \\
&\le \left\lfloor \frac{l-1}{l} \right\rfloor + \left\lfloor \frac{n}{l} \right\rfloor + 1 \\
&= \left\lfloor \frac{n}{l} \right\rfloor + 1
\end{aligned}
$$

Therefore, we get $\left\lfloor \frac{k \cdot m + i}{m \cdot l} \right\rfloor \in \{0, 1, ..., \left\lfloor \frac{n}{l} \right\rfloor\}$. Then, $j - i = k \cdot m - \left\lfloor \frac{k \cdot m + i}{m \cdot l} \right\rfloor \cdot m \cdot l$. k, m and l are all constant number for one transformation. The set $\{0, 1, ..., \left\lfloor \frac{n}{l} \right\rfloor\}$ is of size $\left\lfloor \frac{n}{l} \right\rfloor + 1$.

In summary, $\mathbf{U}^{\epsilon^k_{m\times n}}$ has at most $\lfloor\frac{n}{l}\rfloor+1$ constant values when $\mathcal{A}_{m\times l}$ in **column-major**.

Special circumstances is when $n=l$, $\lfloor\frac{n}{l}\rfloor=1$. Therefore, $\lfloor\frac{n}{l}\rfloor+1=2$ and this means $\mathbf{U}^{\epsilon^k_{m\times n}}$ has only **2** non-zero diagonals when $n=l$..

When applying ϵ transformation on matrix $\mathcal{A}_{m\times l}$ in **row-major** order, we can formulate permutation matrix according to formula (15), but apply on $\mathcal{A}_{l\times m}$ instead of $\mathcal{A}_{l\times n}$. Note that $\mathbf{U}^{\epsilon^k_{m\times n}}_{i,j}=1$ when $j=[k+[i]_n]_l+\lfloor i/n\rfloor\cdot l$ and, for all elements of $\mathbf{U}^{\epsilon^k_{m\times n}}_{i,j}$ that belong to the same diagonal, we have $j-i$ as a constant.

Considering all the non-zero elements in $\mathbf{U}^{\epsilon^k_{m\times n}}_{i,j}$, we have

$$\begin{aligned}
j &= k+[i]_n-\left\lfloor\frac{k+[i]_n}{l}\right\rfloor\cdot l+\left\lfloor\frac{i}{n}\right\rfloor\cdot l\\
&= k+[i]_n+\left(\left\lfloor\frac{i}{n}\right\rfloor-\left\lfloor\frac{k+[i]_n}{l}\right\rfloor\right)\cdot l
\end{aligned}$$

Since $i\in[0,mn)$, we split i to m circumstances that $i\in[pn,(p+1)n)$ where $p=\{0,1,2,...,m-1\}$. For for each circumstance that $i\in[pn,(p+1)n)$, we have

$$j=k+i-pn+\left(p-\left\lfloor\frac{k+[i]_n}{l}\right\rfloor\right)\cdot l$$

and

$$j-i=k-pn+\left(p-\left\lfloor\frac{k+[i]_n}{l}\right\rfloor\right)\cdot l$$

Note that we have

$$\left\lfloor\frac{[pn]_n}{l}\right\rfloor\le\left\lfloor\frac{k+[i]_n}{l}\right\rfloor<\left\lfloor\frac{[pn]_n}{l}\right\rfloor+\left\lfloor\frac{n}{l}\right\rfloor+1+1$$

which has $2+\lfloor\frac{n}{l}\rfloor$ constant values. And this means $j-i$, which represents the number of non-zero diagonals in $\mathbf{U}^{\epsilon^k_{m\times n}}$, has $(2+\lfloor\frac{n}{l}\rfloor)\cdot m$ in total when $\mathcal{A}_{m\times l}$ in **row-major** because there are m circumstances.

Special circumstances is when $n=l$, $j-i\in\{0,1\}$. The reason is that, since

$$\left\lfloor\frac{k}{l}\right\rfloor+\left\lfloor\frac{[i]_n}{l}\right\rfloor\le\left\lfloor\frac{k+[i]_n}{l}\right\rfloor\le\left\lfloor\frac{k}{l}\right\rfloor+\left\lfloor\frac{[i]_n}{l}\right\rfloor+1$$

and we also have $k<l$ and $[i]_n<l$, thus

$$0\le\left\lfloor\frac{k+[i]_n}{l}\right\rfloor\le1$$

On the other hand, we have

$$j-i=k-\left\lfloor\frac{k+[i]_n}{l}\right\rfloor\cdot l$$

for each $i\in[pn,(p+1)n)$. $j-i$ has the same constant value in each $i\in[pn,(p+1)n)$ and this means $\mathbf{U}^{\epsilon^k_{m\times n}}$ has only **2** non-zero diagonals when $n=l$. $\quad\square$

Theorem. III.4 *Let $\omega^k_{m\times n}(\mathcal{B})=\mathbf{U}^{\omega^k_{m\times n}}\mathcal{B}$ be the linear transformation $\omega_{m\times n}:\mathcal{R}_{l\times n}\to\mathcal{R}_{m\times n}$ with matrix $\mathcal{B}$ having a dimension of $l\times n$. There are at most $(\lfloor\frac{m}{l}\rfloor+2)\cdot n$ non-zero diagonal vectors in $\mathbf{U}^{\omega^k_{m\times n}}$ when the matrix is flattened with*

column-major order; There are at most $\lfloor\frac{m}{l}\rfloor+1$ non-zero diagonal vectors in $\mathbf{U}^{\omega^k_{m\times n}}$ when matrix $\mathcal{B}$ is flattened with row-major order. Specifically, when $m=l$, there are no more than 2 non-zero diagonals in $\mathbf{U}^{\omega^k_{m\times n}}$, no matter if the matrix is flattened in column-major or row-major order.

Proof. When applying ω transformation on matrix $\mathcal{B}_{l\times n}$ in **column-major** order, $\mathbf{U}^\omega$ is formulated in Equation (15). Note that $\mathbf{U}^{\omega^k_{m\times n}}_{i,j}=1$ when $j=[k+[i]_m]_l+\lfloor i/m\rfloor\cdot l$ and, for all elements of $\mathbf{U}^{\omega^k_{m\times n}}_{i,j}$ that belong to the same diagonal, we have $j-i$ as a constant.

Considering all the non-zero elements in $\mathbf{U}^{\omega^k_{m\times n}}_{i,j}$, we have

$$\begin{aligned}
j &= k+[i]_m-\left\lfloor\frac{k+[i]_m}{l}\right\rfloor\cdot l+\left\lfloor\frac{i}{m}\right\rfloor\cdot l\\
&= k+[i]_m+\left(\left\lfloor\frac{i}{m}\right\rfloor-\left\lfloor\frac{k+[i]_m}{l}\right\rfloor\right)\cdot l
\end{aligned}$$

Since $i\in[0,mn)$, we split i to n circumstances that $i\in[pm,(p+1)m)$ where $p=\{0,1,2,...,n-1\}$. For each circumstance that $i\in[pm,(p+1)m)$, we have

$$j=k+i-pm+\left(p-\left\lfloor\frac{k+[i]_m}{l}\right\rfloor\right)\cdot l$$

and

$$j-i=k-pm+\left(p-\left\lfloor\frac{k+[i]_m}{l}\right\rfloor\right)\cdot l$$

Note that we have

$$\left\lfloor\frac{[pm]_m}{l}\right\rfloor\le\left\lfloor\frac{k+[i]_m}{l}\right\rfloor<\left\lfloor\frac{[pm]_m}{l}\right\rfloor+\left\lfloor\frac{m}{l}\right\rfloor+1+1$$

which has $2+\lfloor\frac{m}{l}\rfloor$ constant values. And this means $j-i$, which represents the number of non-zero diagonals in $\mathbf{U}^{\omega^k_{m\times n}}$, has $(2+\lfloor\frac{m}{l}\rfloor)\cdot n$ in total when $\mathcal{B}_{m\times l}$ in **row-major** because there are n circumstances.

Special circumstances is when $m=l$, $j-i\in\{0,1\}$. The reason is that, since

$$\left\lfloor\frac{k}{l}\right\rfloor+\left\lfloor\frac{[i]_l}{l}\right\rfloor\le\left\lfloor\frac{k+[i]_l}{l}\right\rfloor\le\left\lfloor\frac{k}{l}\right\rfloor+\left\lfloor\frac{[i]_l}{l}\right\rfloor+1$$

and we also have $k<l$ and $[i]_l<l$, thus

$$0\le\left\lfloor\frac{k+[i]_l}{l}\right\rfloor\le1$$

On the other hand, we have

$$j-i=k-\left\lfloor\frac{k+[i]_l}{l}\right\rfloor\cdot l$$

for each $i\in[pm,(p+1)m)$. $j-i$ has the same constant value in each $i\in[pm,(p+1)m)$ and this means $\mathbf{U}^{\omega^k_{m\times n}}$ has only **2** non-zero diagonals when $m=l$.

When applying ω transformation on matrix $\mathcal{B}_{l\times n}$ in **row-major** order, we can formulate permutation matrix according to formula (14), but apply on $\mathcal{B}_{n\times l}$ instead of $\mathcal{B}_{m\times l}$. Note that $\mathbf{U}^{\omega^k_{m\times n}}_{i,j}=1$ when $j=[k\cdot n+i]_{n\cdot l}$ and, for all elements of

$\mathbf{U}_{i,j}^{\omega_{m \times n}^k}$ that belong to the same diagonal, we have $j - i$ as a constant.

Considering all the non-zero elements in $\mathbf{U}_{i,j}^{\omega_{m \times n}^k}$, we have

$$
\begin{aligned}
j - i &= [k \cdot n + i]_{n \cdot l} - i \\
&= k \cdot n + i - \left\lfloor \frac{k \cdot n + i}{n \cdot l} \right\rfloor \cdot n \cdot l - i \\
&= k \cdot n - \left\lfloor \frac{k \cdot n + i}{n \cdot l} \right\rfloor \cdot n \cdot l
\end{aligned}
$$

Since $\max(k) = l - 1$ and $\max(i) = m \cdot n - 1$, we have

$$
\begin{aligned}
\max(\frac{k \cdot n + i}{n \cdot l}) &< \frac{l - 1 + m}{l} \\
&\leq \left\lfloor \frac{l-1}{l} \right\rfloor + \left\lfloor \frac{m}{l} \right\rfloor + 1 \\
&= \left\lfloor \frac{m}{l} \right\rfloor + 1
\end{aligned}
$$

Therefore, we get $\left\lfloor \frac{k \cdot n + i}{n \cdot l} \right\rfloor \in \{0, 1, ..., \left\lfloor \frac{m}{l} \right\rfloor\}$. Then, $j - i = k \cdot n - \left\lfloor \frac{k \cdot n + i}{n \cdot l} \right\rfloor \cdot n \cdot l$. Here, k, n and l are all constant number for one transformation. The set $\{0, 1, ..., \left\lfloor \frac{m}{l} \right\rfloor\}$ is of size $\left\lfloor \frac{m}{l} \right\rfloor + 1$. In summary, $\mathbf{U}^{\omega_{m \times n}^k}$ has at most $\left\lfloor \frac{m}{l} \right\rfloor + 1$ constant values when $\mathcal{B}_{m \times l}$ in **row-major**.

Special circumstances is when $m = l$, $\left\lfloor \frac{m}{l} \right\rfloor = 1$. Therefore, $\left\lfloor \frac{m}{l} \right\rfloor + 1 = 2$ and this means $\mathbf{U}^{\omega_{m \times n}^k}$ has only **2** non-zero diagonals when $m = l$.. $\qquad\square$

Theorem. III.5 *Let $\mathcal{A}_{m \times l}$ and $\mathcal{B}_{l \times n}$ with $m < l$, and let $\bar{A}$ be matrix expanded with $t = \lceil \frac{l}{m} \rceil$ copies of $\mathcal{A}$ vertically, i.e., $\bar{A} = \{\bar{A}_0; \bar{A}_1; ...; \bar{A}_{(t-1)}\}^T$ with $\bar{A}_0 = \bar{A}_1 = ... = \bar{A}_{(t-1)} = \mathcal{A}_{m \times l}$. Then*

- *$\epsilon_{tm \times n}^k(\sigma(\bar{A})) \odot \omega_{tm \times n}^k(\tau(\mathcal{B}))$ contains t items of $\epsilon_{m \times n}^p(\sigma(\mathcal{A})) \odot \omega_{m \times n}^p(\tau(\mathcal{B}))$, with $p \in \{[k]_l, [k + m]_l, ..., [k + (t-1)m]_l\}$.*
- *$\epsilon_{tm \times n}^k(\sigma(\bar{A})) \odot \omega_{tm \times n}^k(\tau(\mathcal{B}))$, $k = 0, 1, ..., (m-1)$ contains all items of $\epsilon_{m \times n}^p(\sigma(\mathcal{A})) \odot \omega_{m \times n}^p(\tau(\mathcal{B}))$, with $p \in \{0, 1, ..., (l-1)\}$.*

Proof. Consider a sub matrix of $(\epsilon_{tm \times n}^k \circ \sigma(\bar{A}))$ with dimension of $m \times n$, i.e., $(\epsilon_{tm \times n}^k \circ \sigma(\bar{A}))_{hm+i,j}$, where $0 \leq i < m, 0 \leq j < n$. h is a constant with $0 \leq h < t$. Based on equation (1) and (3), we have

$$
\begin{aligned}
(\epsilon_{tm \times n}^k \circ \sigma(\bar{A}))_{hm+i,j} &= \sigma(\bar{A})_{hm+i,[j+k]_l} \\
&= \bar{A}_{hm+i,[hm+i+j+k]_l} \\
&= \mathcal{A}_{i,[hm+i+j+k]_l} \qquad (16)
\end{aligned}
$$

On the other hand, let $p = [k + hm]_l$, for $0 \leq i < m, 0 \leq j < n$, we have

$$
\begin{aligned}
(\epsilon_{m \times n}^p \circ \sigma(\mathcal{A}))_{i,j} &= \sigma(\mathcal{A})_{i,[j+p]_l} \\
&= \mathcal{A}_{i,[i+j+k+hm]_l}. \qquad (17)
\end{aligned}
$$

Similarly, consider the sub matrix of $(\omega_{tm \times n}^k \circ \tau(\mathcal{B}))$ with dimension of $m \times n$, i.e., $(\omega_{tm \times n}^k \circ \tau(\mathcal{B}))_{hm+i,j}$, with $0 \leq i < m, 0 \leq j < n$. Based on equation (2) and (4), we have

$$
\begin{aligned}
(\omega_{tm \times n}^k \circ \tau(\mathcal{B}))_{hm+i,j} &= \tau(\mathcal{B})_{[hm+i+k]_l,j} \\
&= \mathcal{B}_{[hm+i+j+k]_l,j} \qquad (18)
\end{aligned}
$$

If we let $p = [k + hm]_l$, for $0 \leq i < m, 0 \leq j < n$, and $0 \leq h < t$, we have

$$
\begin{aligned}
\omega_{m \times n}^p \circ \tau(\mathcal{B})_{i,j} &= \tau(\mathcal{B})_{[i+p]_l,j} \\
&= \mathcal{B}_{[i+k+hm+j]_l,j} \qquad (19)
\end{aligned}
$$

Since $0 \leq h < t$, there are total t sub matrices in $\epsilon_{tm \times n}^k(\sigma(\bar{A}))$ and $\omega_{tm \times n}^k(\tau(\mathcal{B}))$, the conclusion for the first part of the theorem follows naturally from equation (16) to (19).

To prove the second part of the theorem, we only need to note that since $t = \lceil \frac{l}{m} \rceil$, we have $tm \geq l$. Therefore, for any $p \in \{0, 1, ..., (l-1)\}$, we must be able to find at least one set of k and h, with $0 \leq k < m$, $0 \leq h < t$, and $p = [k + hm]_l$. Together with equation (16) to (19), we thus prove the theorem.

$\qquad\square$

Theorem. III.6 *Let $\mathcal{A}_{m \times l}$ and $\mathcal{B}_{l \times n}$ with $n < l$, and let $\bar{\mathcal{B}}$ be matrix expanded with $t = \lceil \frac{l}{n} \rceil$ copies of $\mathcal{B}$ horizontally, i.e., $\bar{\mathcal{B}} = \{\mathcal{B}; \mathcal{B}; ...; \mathcal{B}\}$. Then*

- *$\epsilon_{m \times tn}^k(\sigma(\mathcal{A})) \odot \omega_{m \times tn}^k(\tau(\bar{\mathcal{B}}))$ contains t items of $\epsilon_{m \times n}^p(\sigma(\mathcal{A})) \odot \omega_{m \times n}^p(\tau(\mathcal{B}))$, with $p = [k]_l, [k + n]_l, ..., [k + (t-1)n]_l$;*
- *$\epsilon_{m \times tn}^k(\sigma(\mathcal{A})) \odot \omega_{m \times tn}^k(\tau(\bar{\mathcal{B}})$, $k = 0, 1, ..., (n-1)$ contains all items of $\epsilon_{m \times n}^p(\sigma(\mathcal{A})) \odot \omega_{m \times n}^p(\tau(\mathcal{B}))$, with $p = 0, 1, ..., (l-1)$.*

Proof. Consider a sub matrix of $(\epsilon_{m \times tn}^k \circ \sigma(\mathcal{A}))$ with dimension of $m \times n$, i.e., $(\epsilon_{m \times tn}^k \circ \sigma(\mathcal{A}))_{i,hn+j}$, where $0 \leq i < m, 0 \leq j < n$. h is a constant with $0 \leq h < t$. Based on equation (1) and (3), we have

$$
\begin{aligned}
(\epsilon_{m \times tn}^k \circ \sigma(\mathcal{A}))_{i,hn+j} &= \sigma(\mathcal{A})_{i,[hn+j+k]_l} \\
&= \mathcal{A}_{i,[i+hn+j+k]_l} \qquad (20)
\end{aligned}
$$

On the other hand, let $p = [k + hn]_l$, for $0 \leq i < m, 0 \leq j < n$, we have

$$
\begin{aligned}
(\epsilon_{m \times n}^p \circ \sigma(\mathcal{A}))_{i,j} &= \sigma(\mathcal{A})_{i,[j+p]_l} \\
&= \mathcal{A}_{i,[i+j+k+hn]_l}. \qquad (21)
\end{aligned}
$$

Similarly, consider the sub matrix of $(\omega_{m \times tn}^k \circ \tau(\bar{\mathcal{B}}))$ with dimension of $m \times n$, i.e., $(\omega_{m \times tn}^k \circ \tau(\bar{\mathcal{B}}))_{i,hn+j}$, with $0 \leq i < m, 0 \leq j < n$. Based on equation (2) and (4), we have

$$
\begin{aligned}
(\omega_{m \times tn}^k \circ \tau(\bar{\mathcal{B}}))_{i,hn+j} &= \tau(\bar{\mathcal{B}})_{[i+k]_l,hn+j} \\
&= \bar{\mathcal{B}}_{[hn+i+j+k]_l,hn+j} \\
&= \mathcal{B}_{[hn+i+j+k]_l,j} \qquad (22)
\end{aligned}
$$

If we let $p = [k + hn]_l$, for $0 \leq i < m, 0 \leq j < n$, and $0 \leq h < t$, we have

$$
\begin{aligned}
\omega_{m \times n}^p \circ \tau(\mathcal{B})_{i,j} &= \tau(\mathcal{B})_{[i+p]_l,j} \\
&= \mathcal{B}_{[i+k+hn+j]_l,j} \qquad (23)
\end{aligned}
$$

Since $0 \leq h < t$, there are total t sub matrices in $\epsilon_{m \times tn}^k(\sigma(\mathcal{A}))$ and $\omega_{m \times tn}^k(\tau(\mathcal{B}))$, the conclusion for the first part of the theorem follows naturally from equation (20) to (23).

To prove the second part of the theorem, we only need to note that since $t = \lceil \frac{l}{n} \rceil$, we have $tn \geq l$. Therefore, for any $p \in \{0, 1, ..., (l-1)\}$, we must be able to find at least one set of k and h, with $0 \leq k < m$, $0 \leq h < t$, and $p = [k + hn]_l$. Together with equation (20) to (23), we thus prove the theorem.

$\square$

APPENDIX B
MEANING OF SYMBOLIZE

TABLE V
MEANING OF SYMBOLIZE

Symbolize	Meaning
$\mathcal{A}$	left matrix for matrix multiplicaiton
$\mathcal{B}$	right matrix for matrix multiplicaiton
m	the number of row of matrix $\mathcal{A}$
l	the number of column of matrix $\mathcal{A}$
	the number of row of matrix $\mathcal{B}$
n	the number of column of matrix $\mathcal{B}$
σ	The transformation that permute each row
τ	The transformation that permute each column
ϵ	The transformation that permute mutiple columns
ω	The transformation that permute mutiple rows
ct	the prefix of ciphertext
$\mathbf{U}$	permutation matrix
$\odot$	elemenwise multiplication