Fully Private Grouped Matrix Multiplication

Lev Tauz and Lara Dolecek
Department of Electrical and Computer Engineering, University of California, Los Angeles, USA
levtauz@ucla.edu and dolecek@ee.ucla.edu

***Abstract*—In this paper, we consider the novel concept of batch size privacy in distributed coded matrix multiplication which adds the constraint that workers cannot learn the number of matrix products being calculated. Batch size privacy helps hide the activity of the master and prevents workers from discriminating users based on usage patterns. As a primary example, we focus on the model of fully private grouped matrix multiplication where a master wants to compute a group of matrix products between two matrix libraries that can be accessed by all workers while ensuring that any number of prescribed colluding workers learn nothing about which matrix products the master desires, nor the number of matrix products. We present an achievable scheme using a variant of Cross-Subspace Alignment (CSA) codes that offers flexibility in communication and computation costs with good straggler resilience.**

I. Introduction and System Model

With the arrival of Big Data, many modern data driven applications are being outsourced to a distributed system of many workers in order to achieve scalability. However, outsourcing the computations and data storage across workers comes with additional concerns such as the presence of *stragglers* (i.e., workers that fail or are slow to respond) who hamper the speed of the system or the privacy concerns about storing confidential data across many workers. Coded computation is a field of research that tackles these issues utilizing techniques from channel coding for a variety of system models [1]–[3]. In this work, we consider the fundamental operation of large matrix multiplication which is an important component for machine learning and data analysis.

The literature on coded matrix multiplication with privacy concerns has tackled a variety of different models. For example, *secure and private matrix multiplication* (SPMM) [1] tasks a system to compute the product of a private matrix $\mathbf{A}$ with a specific matrix $\mathbf{B}_j$, $1 \leq j \leq k$, among a library of matrices $\{\mathbf{B}_1, \ldots, \mathbf{B}_k\}$ which are stored at the workers while ensuring the workers learn nothing about the matrix $\mathbf{A}$ and the index j. Another example is *fully private matrix multiplication* (FPMM) [2] where the workers store two libraries of matrices and the master tasks the workers to privately calculate the product of two desired matrices from the shared libraries while being oblivious to the indices of the desired matrices. A further variation is *secure batch matrix multiplication* (SBMM) [3] where the master tasks the system to calculate the product of multiple matrix pairs without the workers learning anything about the matrices.

In this work, we propose a brand new privacy consideration known as *batch size privacy* where the master wants to hide the number of batches, i.e. the number of matrix products, it wishes to compute. If we look at the previous models, the master did not care whether the workers knew how many matrix products the master requested. This can be problematic, for example, in SPMM if the user wishes to calculate all the matrix products between $\mathbf{A}$ and $\{\mathbf{B}_1, \ldots, \mathbf{B}_k\}$ in a short time and knowing the batch size reveals which matrices were requested. Even schemes for SBMM that consider multiple matrix products do not consider batch size privacy and, often, use the fact that the workers know the batch size to provide further optimization. From a practical perspective, knowing the batch size can reveal to the workers the type of entity that is asking for this request. For example, small IOT devices would generally request small batches due to their targeted workload while large analytics engines would request larger batch sizes. Being able to discriminate entities based on their usage pattern can be problematic for users who wish to hide their activity. Thus, there is significant merit in studying batch size privacy and providing a novel coding scheme to address it.

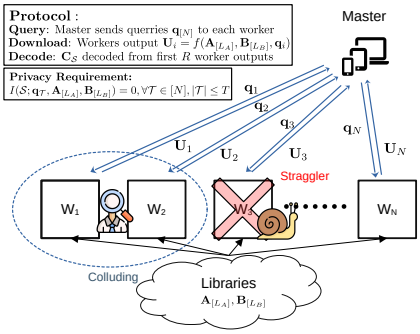

Fig. 1: System model of FPGMM.

To study batch size privacy, we consider a variation of FPMM which we refer to as *fully private grouped matrix multiplication* (FPGMM) where a master can request multiple matrix products, i.e., a group of products, in a single request among two libraries of matrices stored at the workers. Since the data is stored at the workers, the master wishes to preserve the privacy of which matrix product are requested and how many were requested. Formally, assume a distributed system with one master and N workers. All workers store two libraries of matrices $\mathbf{A}_{[L_A]} = \{\mathbf{A}_i \in \mathbb{F}_q^{\alpha \times \alpha}, \forall i \in [L_A]\}$ and $\mathbf{B}_{[L_B]} = \{\mathbf{B}_i \in \mathbb{F}_q^{\alpha \times \alpha}, \forall i \in [L_B]\}$ where $\mathbb{F}_q$ is a finite field of size q and L_A, L_B are non-negative integers.[1] Given a set $\mathcal{S} \subseteq [L_A] \times [L_B]$, the master wants to obtain the matrix products $\mathbf{C}_\mathcal{S} \triangleq \{\mathbf{A}_i \mathbf{B}_j : (i,j) \in \mathcal{S}\}$. We assume that $\mathcal{S}$ is equally likely to be any non-empty subset of $[L_A] \times [L_B]$ and is chosen independently of the data stored in the two matrix libraries. The master does not want the workers to learn anything about $\mathcal{S}$ including its cardinality.

The FPGMM scheme contains the following phases: 1) Encoding Phase: The master designs queries $\mathbf{q}_i, i \in [N]$ based on $\mathcal{S}$; 2) Query and Computation: The master sends query $\mathbf{q}_i, i \in [N]$ to worker i. Worker i then uses $\mathbf{q}_i$ to encode the data libraries using a function $f(\mathbf{A}_{[L_A]}, \mathbf{B}_{[L_B]}, \mathbf{q}_i) = \mathbf{U}_i$ and outputs $\mathbf{U}_i$; 3) Reconstruction: The master downloads $\mathbf{U}_i$ from the servers. Some workers may be stragglers and fail to respond. The master attempts to reconstruct $\mathbf{C}_\mathcal{S}$ from the responding servers. Additionally, FPGMM requires that $\mathcal{S}$ is kept private from up to T colluding workers. This privacy requirement includes both the elements and cardinality of $\mathcal{S}$. The model is summarized in Fig. 1.

For FPGMM, the important metrics are the following:

- *Recovery Threshold R*: The minimum number of worker outputs needed in order to reconstruct $\mathbf{C}_\mathcal{S}$.
- *Normalized Computational Complexity (NCC)*: The average order of the number of arithmetic operations required to compute the function f at each worker, normalized by $|\mathcal{S}|\alpha^3$.

[1]We note that our schemes are applicable for non-square matrices and that we focus on square matrices only for notational convenience.

- *Normalized Download Cost (NDC)*: The total number of symbols retrieved by the master normalized by the number of symbols in $\mathbf{C}_{\mathcal{S}}$.

II. Main Result and Example

The main result is demonstrated in the following theorem.

Theorem 1. *Assume a distributed system with N workers, a computation list $\mathcal{S}$, and T colluding workers. For any positive integers m, n, r such that $m|\alpha$, $n|\alpha$, $r|mn$, and $|\mathbb{F}_q| \geq |\mathcal{S}|mn+N$, there exists a privacy preserving scheme for up to T colluding workers that achieves the following system metrics:*

$$\textit{Recovery Threshold: } R = (\frac{r+1}{r})|\mathcal{S}|mn + 2T - 1,$$

$$\textit{NDC: } \frac{R}{|\mathcal{S}|mn} = \frac{r+1}{r} + \frac{2T-1}{|\mathcal{S}|mn}, \quad \textit{NCC: } \mathcal{O}\left(\frac{r}{|\mathcal{S}|mn}\right) \tag{1}$$

Additionally, r is the number of groups and provides no information about $\mathcal{S}$.

The full proof of Theorem 1 can be found in the full version [4]. Now, we shall show an illustrative example to highlight the key components of our scheme. Assume that $L_A = L_B = 2$ and that $\mathcal{S} = \{(1,1),(1,2)\}$. As such, we want to retrieve $\mathbf{C}_{\mathcal{S}} = \{\mathbf{A}_1\mathbf{B}_1, \mathbf{A}_1\mathbf{B}_2\}$. Additionally, let $T = 1$ to protect privacy against 1 curious worker.

One example of our scheme goes as follows. The master specifies to the workers to partition the $\mathbf{B}_{[L_B]}$ matrices as follows:

$$\mathbf{B}_j = \begin{bmatrix}\mathbf{B}_{i,1} & \mathbf{B}_{i,2}\end{bmatrix}, \forall j \in [2].$$

Now, to calculate $\{\mathbf{A}_1\mathbf{B}_1, \mathbf{A}_1\mathbf{B}_2\}$, a sufficient condition is to calculate $\{\mathbf{A}_1\mathbf{B}_{1,b}\}_{b=1}^{2} \cup \{\mathbf{A}_1\mathbf{B}_{2,b}\}_{b=1}^{2}$

Let $f_{1,1}, f_{1,2}, f_{2,1}$, and $f_{2,2}$ be distinct elements from $\mathbb{F}_q$. The master groups up the computations into two groups $\{\mathbf{A}_1\mathbf{B}_{1,1}, \mathbf{A}_1\mathbf{B}_{2,1}\}$ and $\{\mathbf{A}_1\mathbf{B}_{1,2}, \mathbf{A}_1\mathbf{B}_{2,2}\}$. Note that the grouping is arbitrary but the number of groups is carefully chosen. If the number of groups was instead 4, then the workers can easily determine that $|\mathcal{S}|= 2$ due to knowledge of the partitioning parameters. This limits the straightforward applicability of CSA codes used for SBMM [3] to be used for FPGMM because they are designed for grouping computations based on the size of the batches, i.e. dependent on $|\mathcal{S}|$. We address this issue by grouping computations based on the partitioning parameters which are chosen independently of $\mathcal{S}$.

Consider the following encoding functions

$$a_{i,k}(x) = \omega_k(x) \times \left(z_{i,k}^a + \begin{cases} \frac{1}{x-f_{1,1}} + \frac{1}{x-f_{2,1}} & i=1, k=1, \\ \frac{1}{x-f_{1,2}} + \frac{1}{x-f_{2,2}} & i=1, k=2, \\ 0 & i=2, \end{cases} \right) \tag{2}$$

$$b_{j,l,k}(x) = z_{j,l,k}^b + \begin{cases} \frac{1}{x-f_{j,k}} & l=k, \\ 0 & else, \end{cases} \tag{3}$$

for $i,j,l,k \in [2]$ where $\omega_k(x) = (x-f_{1,k})(x-f_{2,k})$ and $z_{i,k}^a$, $z_{j,l,k}^b$ are random noise terms that are independently and uniformly chosen from $\mathbb{F}_q$. The master assigns each worker $g \in [N]$ a distinct element x_g from $\mathbb{F}_q \setminus \{f_{1,1}, f_{1,2}, f_{2,1}, f_{2,2}\}$. Thus, the query $\mathbf{q}_g, g \in [N]$ that the master sends to worker g contains the evaluations of the encoding functions $\{a_{i,k}(x_g)\}_{i,k\in[2]}$ and $\{b_{j,l,k}(x_g)\}_{j,l,k\in[2]}$, the partitioning parameters, and the number of groups. Note that each encoding function contains a uniformly random variable. By Shamir's well-known secret sharing scheme [5], each worker cannot gain any information about the coefficients in the encoding functions and, thus, cannot learn anything about $\mathcal{S}$. Hence, the scheme is $T = 1$ private.

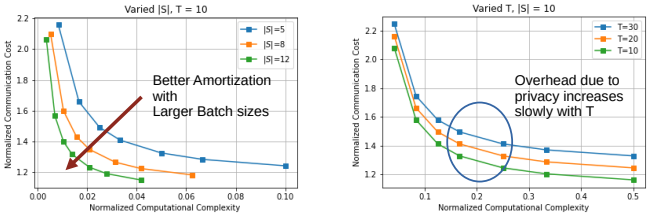

Fig. 2: Plots of NCC and NDC for fixed T (left) and fixed $|\mathcal{S}|$ (right).

After receiving $\mathbf{q}_g$, worker g then encodes the matrices using

$$\widehat{\mathbf{A}}_k = \sum_{i=1}^{2} \mathbf{A}_i a_{i,k}(x_g), \widehat{\mathbf{B}}_k = \sum_{j=1}^{2}\sum_{l=1}^{2} \mathbf{B}_{j,l} b_{j,l,k}(x_g) \tag{4}$$

for $k \in [2]$. Now, the worker will calculate $\mathbf{C}(x_g) = \widehat{\mathbf{A}}_1\widehat{\mathbf{B}}_1 + \widehat{\mathbf{A}}_2\widehat{\mathbf{B}}_2$ where $\mathbf{C}(x_g)$ can be simplified into the following form:

$$\frac{\mathbf{A}_1\mathbf{B}_{1,1}}{(x_g - f_{1,1})} + \frac{\mathbf{A}_1\mathbf{B}_{2,1}}{(x_g - f_{2,1})} + \frac{\mathbf{A}_1\mathbf{B}_{1,2}}{(x_g - f_{1,2})} + \frac{\mathbf{A}_1\mathbf{B}_{2,2}}{(x_g - f_{2,2})} + \mathbf{I}(x_g) \tag{5}$$

where $\mathbf{I}(x)$ is a polynomial matrix that contains all the polynomial terms in $\mathbf{C}(x_g)$. Note that the maximum degree of $\mathbf{I}(x)$ is $\max_{k\in[2]}(\deg(\omega_k(x))) + 2T - 2 = 2 + 2*1 - 2 = 2$ since the largest polynomial degree in $\widehat{\mathbf{A}}_k$ and $\widehat{\mathbf{B}}_k$ is $\deg(\omega_k(x)) + T - 1$ and $T-1$, respectively. We highlight the fact that all desired matrices are coefficients to unique rational terms in Eq. (5). We achieved this by encoding each term in a desired matrix product with a unique root in the denominator so that when $\mathbf{C}(x)$ is computed, the desired matrix product remains the only term with the unique root in the denominator. One can think of $\omega(x)$ as a filter where all desired terms are kept with the rational terms and all other terms are aligned into polynomial terms. It is well know that functions of the form in Eq. (5) can be interpolated if the number of evaluations is at least the number of rational and polynomial terms. As such, the master needs only 7 worker outputs since the polynomial terms have 3 coefficients and the rational terms have 4 coefficients. Thus, the recovery threshold is 7. Now, since $C(x) \in \mathbb{F}_q^{\alpha \times \frac{\alpha}{2}}$, the NDC is $\frac{7}{4}$. Additionally, we see that to calculate $C(x)$ the worker had to encode the matrices with complexity $\mathcal{O}\left(L_A\alpha^2 + L_B * 2 * \frac{\alpha^2}{2}\right) = \mathcal{O}\left(6\alpha^2\right)$ and then multiply and add the results with complexity $\mathcal{O}\left(\alpha^3\right)$. Since we assume that α is very large, the NCC is $\mathcal{O}\left(\alpha^3 \times \frac{1}{|\mathcal{S}|\alpha^3}\right) = \mathcal{O}\left(\frac{1}{2}\right)$.

Fig. 2 plots the NCC and NDC for both fixed T and fixed $|\mathcal{S}|$. All points are achievable using our scheme and we can see from the left plot that we get better amortization with larger batch sizes without sacrificing privacy. The right plot demonstrates that the overhead due to batch size privacy is small. As such, we provide an efficient and flexible scheme to solve FPGMM.

References

[1] Z. Jia and S. A. Jafar, "X-secure t-private information retrieval from mds coded storage with byzantine and unresponsive servers," *IEEE Transactions on Information Theory*, vol. 66, pp. 7427–7438, July 2020.

[2] J. Zhu and S. Li, "A systematic approach towards efficient private matrix multiplication," *IEEE Journal on Selected Areas in Information Theory*, vol. 3, pp. 257–274, June 2022.

[3] J. Zhu, Q. Yan, and X. Tang, "Improved constructions for secure multi-party batch matrix multiplication," *IEEE Transactions on Communications*, vol. 69, pp. 7673–7690, Aug. 2021.

[4] L. Tauz and L. Dolecek, "Fully private grouped matrix multiplication with colluding workers," in *IEEE International Symposium on Information Theory (ISIT)*, pp. 2529–2534, 2023.

[5] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, pp. 612–613, Nov. 1979.