This is an electronic reprint of the original article.
This reprint may differ from the original in pagination and typographic detail.

Li, Jie; Hollanti, Camilla

Improved Private and Secure Distributed Matrix Multiplication

Published in:
2021 IEEE International Symposium on Information Theory, ISIT 2021 - Proceedings

DOI:
[10.1109/ISIT45174.2021.9517825](https://doi.org/10.1109/ISIT45174.2021.9517825)

Published: 12/07/2021

Document Version
Peer reviewed version

Please cite the original version:
Li, J., & Hollanti, C. (2021). Improved Private and Secure Distributed Matrix Multiplication. In *2021 IEEE International Symposium on Information Theory, ISIT 2021 - Proceedings* (pp. 1859-1864). (IEEE International Symposium on Information Theory - Proceedings; Vol. 2021-July). IEEE.
<https://doi.org/10.1109/ISIT45174.2021.9517825>

This material is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of the repository collections is not permitted, except that material may be duplicated by you for your research use or educational purposes in electronic or print form. You must obtain permission for any other use. Electronic or print copies may not be offered, whether for sale or otherwise to anyone who is not an authorised user.

Improved Private and Secure Distributed Matrix Multiplication

Jie Li and Camilla Hollanti

Department of Mathematics and Systems Analysis, Aalto University
FI-00076 Aalto, Finland

E-mails: jieli873@gmail.com, camilla.hollanti@aalto.fi

Abstract—Consider the problem of a user having a private matrix A and N non-colluding servers sharing a library of L ($L > 1$) matrices $B^{(0)}, B^{(1)}, \dots, B^{(L-1)}$, for which the user wishes to compute $AB^{(\theta)}$ for some $\theta \in [0, L)$ without revealing any information of the matrix A to the servers, and keeping the index θ private to the servers. This problem is known as private and secure distributed matrix multiplication (PSDMM) and is supposed to have wide application potential. However, studies of PSDMM are still scarce in the literature. In this paper, we propose a new efficient private and secure distributed matrix multiplication coding scheme, which has a better performance than state-of-the-art schemes in that it achieves a smaller recovery threshold and download cost as well as providing a more flexible tradeoff between the upload and download costs.

I. INTRODUCTION

Matrix multiplication is one of the key operations in many science and engineering fields, such as machine learning and cloud computing. Carrying out the computation on distributed servers are desirable for improving efficiency and reducing the user's computation load, as the user can divide the computation at hand into several sub-tasks to be carried out by the helper servers. By scaling out computations across many distributed servers, the computation latency can be affected by orders of magnitude due to the presence of stragglers, see *e.g.*, [1]. Recent work has shown that coding technique can help reduce the computation latency [2]–[6].

As the computations are scaling out across many distributed servers, besides stragglers, security is also a concern as the servers might be curious about the matrix contents. A typical assumption is that any X of them may collude to deduce information about the matrices possessed by the user [7]. This raises the problem of *secure distributed matrix multiplication* (SDMM), which has recently received a lot of attention from an information-theoretic perspective [7]–[14].

In a variant of this problem, the user privacy should also be taken into account. Consider the scenario that the user has a private matrix A and there are N non-colluding servers sharing a library of L matrices $B^{(0)}, B^{(1)}, \dots, B^{(L-1)}$, for which the user wishes to compute $AB^{(\theta)}$ for some $\theta \in [0, L)$ without revealing any information of the matrix A to the servers, and the servers should not identify the index θ of the desired matrix $B^{(\theta)}$. This problem is termed *private and secure distributed matrix multiplication* (PSDMM), which is

supposed to have wide application potential, including e-Health [15], smart systems [16], private infection tracking [17], AI recommendation service [18], etc. PSDMM together with the problem of SDMM is also quite relevant to the problem of private information retrieval (PIR) [19]–[33]. Indeed, a connection between a form of SDMM and a form of PIR was drawn in [8], where an upper bound of the capacity of SDMM is characterized by that of PIR. Later in [29], a scheme that was initially used to study the problem of PIR was also applied to another PSDMM model that differs from the one in this paper. Further in [34], the problem of SDMM is converted into a PIR problem. The PSDMM code in [35] is also based on PIR. Besides, PSDMM is also relevant to the problem of private computation [36]. Especially, when the private matrix A is a row vector while the matrices $B^{(i)}$ stored across the servers are column vectors, the problem of PSDMM can be viewed as the problem of private computation.

Generally, three performance metrics are of particular interest for PSDMM and more generally for any matrix multiplication schemes:

- The upload cost: the amount of data transmitted from the user to the servers to assign the sub-tasks;
- The download cost: the amount of data to be downloaded from the servers;
- The recovery threshold R_c : the number of servers that need to complete their task before the user can recover the desired matrix product.

Previous schemes aim at minimizing the above three metrics. Usually, the amount of data to be downloaded from an individual server is fixed after the user assigns the sub-tasks to the servers. Thus in most cases, including all the schemes involved in this paper, minimizing the download cost is equivalent to minimizing the recovery threshold R_c .

Up to now, there are not many constructions for PSDMM codes. The PSDMM code in [35] based on PIR can provide a flexible tradeoff between the upload cost and download cost, but requires a high sub-packetization degree and cannot mitigate stragglers. In addition, the minimum value of the upload cost of PSDMM has also been derived in [35]. However, achieving the minimum upload cost with this scheme requires a huge download cost. In [18], PSDMM codes based on the polynomial codes in [3] were presented. In [12], Aliasgari *et al.* proposed a new PSDMM code based on the entangled

polynomial codes in [6], which generalizes the one in [18], and allows for a flexible tradeoff between the upload cost and the download cost (or equivalently recovery threshold). Very recently, Yu *et al.* [13] studied the problem of PSDMM based on Lagrange coded computing [37] and bilinear complexity [38]. The detailed performance of the above codes will be illustrated in Section IV together with the new code proposed in this work. To the best of our knowledge, except for the minimum value of the upload cost derived in [35], no other fundamental limits for the PSDMM problem have been reported in the literature.

In this paper, we propose a new efficient PSDMM code and show that optimizing the performance is equivalent to optimizing the *degree table*, which was first introduced in [7]. An efficient degree table is also presented. The proposed PSDMM code is explicit and has, in certain aspects, better performance than previous ones. More specifically,

- The new PSDMM code outperforms the ones in [12] and [18] in that it has a smaller recovery threshold as well as download cost under the same upload cost, and it provides a more flexible tradeoff between the upload and download costs than the one in [18].
- The new PSDMM code also has a smaller recovery threshold and download cost when compared with the ones in [13] and [35] for some regions. In addition, it can tolerate stragglers and does not require sub-packetization, which superior to the one in [35].

The detailed comparisons will be given in Section IV.

The rest of this paper is organized as follows. Section II introduces the problem settings. Section III presents a motivating example and the new PSDMM code construction. Section IV compares the new PSDMM code to previous ones. Finally, Section V draws the concluding remarks.

II. PROBLEM SETTINGS

Let $[i, j]$ denote the set $\{i, i+1, \dots, j-1\}$ for any two integers $i < j$. Throughout this paper, we assume that the N servers share the library of L matrices $B^{(i)}$, $i \in [0, L)$ in a replicated form, *i.e.*, every server stores all the L matrices. We also assume that the private matrix A possessed by the user is a $t \times s$ matrix over a finite field $\mathbf{F}_q$, and the matrices $B^{(i)}$, $i \in [0, L)$ stored in each server are $s \times r$ matrices over $\mathbf{F}_q$, where $L > 1$ and t, s , and r are positive integers.

Typically, a PSDMM scheme contains three phases:

- Encoding phase: The user encodes A to obtain $\tilde{A}_i$ for $i \in [0, N)$.
- Query, communication and computation phase: The user sends $\tilde{A}_i$ and a query $q_i^{(\theta)}$ (which is usually independent of the matrix $\tilde{A}_i$) to server i , who then first encodes the library into $\tilde{B}_i$ based on the received query, and then computes $\tilde{A}_i \tilde{B}_i$ and returns the result to the user, where $i \in [0, N)$.
- Decoding phase: From the results returned by the fastest R_c servers, the user can then decode the desired matrix product $AB^{(\theta)}$.

We say that an encoding scheme for PSDMM is *private* if the index θ is private to any individual server, *i.e.*,

$$I(\theta; q_i^{(\theta)}, \tilde{A}_i, B^{(0)}, B^{(1)}, \dots, B^{(L-1)}) = 0,$$

for any $i \in [0, N)$ and $\theta \in [0, L)$, where $I(Y; Z)$ denotes the mutual information between Y and Z . A scheme is called *secure* if no information is leaked about the matrix A to any server, *i.e.*,

$$I(\tilde{A}_i; A) = 0,$$

for any $i \in [0, N)$.

In addition, we wish to minimize the recovery threshold R_c as well as the communication cost, which is comprised of *upload cost* and *download cost*. The upload cost is defined as $\sum_{i=0}^{N-1} H(\tilde{A}_i)$. The download cost is defined as the number of elements in $\mathbf{F}_q$ that the user downloaded from the fastest R_c servers $j_0, \dots, j_{R_c-1}$, maximized over $\{j_0, \dots, j_{R_c-1}\} \subseteq [0, N)$, *i.e.*,

$$\max_{\{j_0, \dots, j_{R_c-1}\} \subseteq [0, N)} \sum_{i=j_0}^{j_{R_c-1}} H(\tilde{A}_i \tilde{B}_i),$$

where $H(Y)$ denotes the entropy of Y .

III. PRIVATE AND SECURE DISTRIBUTED MATRIX MULTIPLICATION

In this section, we propose a new PSDMM code. The construction is largely similar to the one in [12], but the encoding phase of the new PSDMM scheme is more efficient. This is enabled by considering the problem of designing efficient encoding polynomials in the encoding phase and the server computation phase, in the view of designing a *degree table*, which was previously introduced for the problem of SDMM [7]. A better degree table is found, yielding better encoding. This leads to a smaller recovery threshold and download cost. Before presenting the general construction, we first give a motivating example.

A. A motivating example

Assume that $L = 2$, t, s, r are even, and the user wishes to compute $AB^{(0)}$. Divide the matrices A and $B^{(i)}$ into block matrices

$$A = \begin{pmatrix} A_{0,0} & A_{0,1} \\ A_{1,0} & A_{1,1} \end{pmatrix}, \quad B^{(i)} = \begin{pmatrix} B_{0,0}^{(i)} & B_{0,1}^{(i)} \\ B_{1,0}^{(i)} & B_{1,1}^{(i)} \end{pmatrix},$$

where $A_{k,j} \in \mathbf{F}_q^{\frac{t}{2} \times \frac{s}{2}}$ and $B_{j',k'}^{(i)} \in \mathbf{F}_q^{\frac{s}{2} \times \frac{r}{2}}$. Then

$$AB^{(0)} = \begin{pmatrix} C_{0,0}^{(0)} & C_{0,1}^{(0)} \\ C_{1,0}^{(0)} & C_{1,1}^{(0)} \end{pmatrix}$$

where

$$\begin{cases} C_{0,0}^{(0)} = A_{0,0}B_{0,0}^{(0)} + A_{0,1}B_{1,0}^{(0)}, \\ C_{0,1}^{(0)} = A_{0,0}B_{0,1}^{(0)} + A_{0,1}B_{1,1}^{(0)}, \\ C_{1,0}^{(0)} = A_{1,0}B_{0,0}^{(0)} + A_{1,1}B_{1,0}^{(0)}, \\ C_{1,1}^{(0)} = A_{1,0}B_{0,1}^{(0)} + A_{1,1}B_{1,1}^{(0)}. \end{cases} \quad (1)$$

Let Z be a random matrix over $\mathbf{F}_q^{\frac{t}{2} \times \frac{s}{2}}$. Then, define a polynomial

$$f(x) = A_{0,0}x^{\alpha_{0,0}} + A_{0,1}x^{\alpha_{0,1}} + A_{1,0}x^{\alpha_{1,0}} + A_{1,1}x^{\alpha_{1,1}} + Zx^\gamma, \quad (2)$$

where $\alpha_{k,j}, \gamma$ are some integers to be specified later.

Let a_1 and $a_{0,0}, \dots, a_{0,N-1}$ be $N+1$ pairwise distinct elements randomly chosen from $\mathbf{F}_q$. For every $i \in [0, N)$, the user first evaluates $f(x)$ at $a_{0,i}$, then sends $f(a_{0,i})$ and the query

$$q_i^{(0)} = (a_{0,i}, a_1) \quad (3)$$

to server i . Upon receiving the query $q_i^{(0)}$, server i encodes the library into a matrix as $g(a_{0,i})$ where $g(x)$ is defined as

$$g(x) = \sum_{j=0}^1 \sum_{k=0}^1 B_{j,k}^{(0)} x^{\beta_{j,k}} + \sum_{j=0}^1 \sum_{k=0}^1 B_{j,k}^{(1)} a_1^{\beta_{j,k}}. \quad (4)$$

After encoding the library, server i computes $f(a_i)g(a_i)$ and then returns the result to the user.

Let $h(x) = f(x)g(x)$, where the exact expression is given in (5) in the next page. Then the results returned to the user by the servers are exactly the evaluations of $h(x)$ at some evaluation points.

The user wishes to obtain the data in (1) (related to the useful terms in $h(x)$) from any R_c out of the N evaluations of $h(x)$, which can be fulfilled if

- (i) For $k \in [0, 2)$ and $k' \in [0, 2)$,

$$\alpha_{k,0} + \beta_{0,k'} = \alpha_{k,1} + \beta_{1,k'}.$$

- (ii) For $U = \{\alpha_{k,0} + \beta_{0,k'} | 0 \leq k < 2, 0 \leq k' < 2\}$ and

$$\begin{aligned} I = & \{\alpha_{k,j} + \beta_{j',k'} | 0 \leq k, k' < 2, 0 \leq j \neq j' < 2\} \\ & \cup \{\gamma + \beta_{j',k'} | 0 \leq j', k' < 2\} \\ & \cup \{\alpha_{k,j} | 0 \leq k, j < 2\} \cup \{\gamma\}, \end{aligned}$$

$$|U| = 4 \text{ and } U \cap I = \emptyset.$$

- (iii) $R_c = \deg(h(x)) + 1$.

The task can be finished because

- (i) guarantees that each $C_{k,k'}^{(0)}$ appears in $h(x)$,
- (ii) guarantees that $C_{k,k'}^{(0)}, k, k' = 0, 1$ are coefficients of different terms of $h(x)$ with different degrees, which are different from the degrees of the interference terms of $h(x)$, i.e., each $C_{k,k'}^{(0)}$ is the coefficient of a unique term in $h(x)$,
- (iii) guarantees the decodability from Lagrange interpolation.
- By (2) and (ii), we can get $\gamma \neq \alpha_{k,j}$ for $k, j = 0, 1$, thus one easily obtains $I(\{f(a_i)\}; A) = 0$ for any $i \in [0, N)$ and $a_i \in \mathbf{F}_q$, then security is fulfilled. While the privacy condition is met by the definition of the query vector in Eq. (3) for the desired index, the detailed proof is similar to [18].

We provide a concrete exponent assignment for this example in Table I (named degree table¹ in [7]). From the given

TABLE I
AN ASSIGNMENT FOR EXPONENTS OF $f(x)$ AND $g(x)$, WHERE $\spadesuit, \heartsuit, \clubsuit, \diamond$ ARE USED TO HIGHLIGHT THE USEFUL TERMS OF $h(x)$ IN (5).

+	$\beta_{0,0} = 4$	$\beta_{0,1} = 9$	$\beta_{1,0} = 3$	$\beta_{1,1} = 8$
$\alpha_{0,0} = 1$	5 $\spadesuit$	10 $\heartsuit$	4	9
$\alpha_{0,1} = 2$	6	11	5 $\spadesuit$	10 $\heartsuit$
$\alpha_{1,0} = 3$	7 $\clubsuit$	12 $\diamond$	6	11
$\alpha_{1,1} = 4$	8	13	7 $\clubsuit$	12 $\diamond$
$\gamma = 0$	4	9	3	8

assignment, we see that $\deg(h(x)) = 13$ and thus $R_c = 14$.

B. General construction

In the following, we propose a general construction for PSDMM. Let m, p , and n be positive integers such that $m \mid t$, $p \mid s$, $n \mid r$, then we divide the matrices A and $B^{(i)}$ ($i \in [0, L)$) into block matrices as

$$A = (A_{k,j})_{0 \leq k < m, 0 \leq j < p}, \quad B^{(i)} = (B_{j',k'}^{(i)})_{0 \leq j' < p, 0 \leq k' < n},$$

where $A_{k,j} \in \mathbf{F}_q^{\frac{t}{m} \times \frac{s}{p}}$ and $B_{j',k'}^{(i)} \in \mathbf{F}_q^{\frac{s}{p} \times \frac{r}{n}}$.

Then

$$AB^{(i)} = C^{(i)} = (C_{k,k'}^{(i)})_{0 \leq k < m, 0 \leq k' < n}$$

where

$$C_{k,k'}^{(i)} = \sum_{j=0}^{p-1} A_{k,j} B_{j,k'}^{(i)}, \quad k \in [0, m), \quad k' \in [0, n).$$

Let Z be a random matrix over $\mathbf{F}_q^{\frac{t}{m} \times \frac{s}{p}}$, and define a polynomial

$$f(x) = \sum_{k=0}^{m-1} \sum_{j=0}^{p-1} A_{k,j} x^{\alpha_{k,j}} + Zx^\gamma, \quad (6)$$

where $\alpha_{k,j}, \gamma$ are some integers to be specified later.

Suppose the user wishes to compute $AB^{(\theta)}$ for some $\theta \in [0, L)$, then let $a_0, \dots, a_{\theta-1}, a_{\theta+1}, \dots, a_{L-1}$ and $a_{\theta,0}, \dots, a_{\theta,N-1}$ be $N+L-1$ pairwise distinct elements randomly chosen from $\mathbf{F}_q$. For every $i \in [0, N)$, the user first evaluates $f(x)$ at $a_{\theta,i}$, then sends $f(a_{\theta,i})$ and the query

$$q_i^{(\theta)} = (a_0, \dots, a_{\theta-1}, a_{\theta,i}, a_{\theta+1}, \dots, a_{L-1}) \quad (7)$$

to server i .

Let $g_t(x) = \sum_{j=0}^{p-1} \sum_{k=0}^{n-1} B_{j,k}^{(t)} x^{\beta_{j,k}}$, where $\beta_{j,k}, j \in [0, p), k \in [0, n)$ are some integers to be specified later. Let $q_i^{(\theta)}[t]$ be the t -th element in the vector $q_i^{(\theta)}$. Then, upon receiving the

¹In [7], the variables in the first column and the first row of the degree table denote the exponents of the encoding polynomial for the left side matrix A and the right side matrix B . Although there are L right side matrices in the PSDMM problem, but the exponents of the encoding polynomials for the L matrices are the same in our scheme according to (4), thus the PSDMM problem can also be studied in the view of optimizing the degree table.

$$\begin{aligned}
h(x) = & \underbrace{(A_{0,0}B_{0,0}^{(0)}x^{\alpha_{0,0}+\beta_{0,0}} + A_{0,1}B_{1,0}^{(0)}x^{\alpha_{0,1}+\beta_{1,0}} + (A_{0,0}B_{0,1}^{(0)}x^{\alpha_{0,0}+\beta_{0,1}} + A_{0,1}B_{1,1}^{(0)}x^{\alpha_{0,1}+\beta_{1,1}}))}_{\text{useful terms}} \\
& + \underbrace{(A_{1,0}B_{0,0}^{(0)}x^{\alpha_{1,0}+\beta_{0,0}} + A_{1,1}B_{1,0}^{(0)}x^{\alpha_{1,1}+\beta_{1,0}} + (A_{1,0}B_{0,1}^{(0)}x^{\alpha_{1,0}+\beta_{0,1}} + A_{1,1}B_{1,1}^{(0)}x^{\alpha_{1,1}+\beta_{1,1}}))}_{\text{useful terms}} \\
& + \underbrace{\sum_{k=0}^1 \sum_{k'=0}^1 (A_{k,0}B_{1,k'}^{(0)}x^{\alpha_{k,0}+\beta_{1,k'}} + A_{k,1}B_{0,k'}^{(0)}x^{\alpha_{k,1}+\beta_{0,k'}}) + \sum_{j'=0}^1 \sum_{k'=0}^1 ZB_{j',k'}^{(0)}x^{\gamma+\beta_{j',k'}}}_{\text{interference terms}} \\
& + \underbrace{(A_{0,0}x^{\alpha_{0,0}} + A_{0,1}x^{\alpha_{0,1}} + A_{1,0}x^{\alpha_{1,0}} + A_{1,1}x^{\alpha_{1,1}} + Zx^\gamma) \left(B_{0,0}^{(1)}a_1^{\beta_{0,0}} + B_{0,1}^{(1)}a_1^{\beta_{0,1}} + B_{1,0}^{(1)}a_1^{\beta_{1,0}} + B_{1,1}^{(1)}a_1^{\beta_{1,1}} \right)}_{\text{interference terms}} \quad (5)
\end{aligned}$$

TABLE II

A COMPARISON OF KEY PARAMETERS BETWEEN THE PROPOSED PSDMM CODE AND SOME PREVIOUS CODES. $R(p, m, n)$ DENOTES THE MINIMUM NUMBER OF ELEMENT-WISE MULTIPLICATIONS REQUIRED TO COMPLETE THE MULTIPLICATION OF AN $m \times p$ MATRIX AND A $p \times n$ MATRIX [6].

	Upload cost	Download cost	Recovery threshold R_c	References
Chang-Tandon's code	$R_c \frac{ts}{m}$	$tr \frac{m+1}{m} \left(1 + \frac{m+1}{R_c} + \dots + \left(\frac{m+1}{R_c} \right)^{L-1} \right)$	$\geq m+1$	[35]
Kim-Lee's code	$N \frac{ts}{m}$	$R_c \frac{tr}{mn}$	$(m+1)(n+1)$	[18]
Aliasgari <i>et al.</i> 's PSGPD code	$N \frac{ts}{mp}$	$R_c \frac{tr}{mn}$	$pmn + pn + pm - p + 2$	[12]
Yu <i>et al.</i> 's PSDMM code	$N \frac{ts}{mp}$	$R_c \frac{tr}{mn}$	$2R(p, m, n) + 1$	[13]
The new PSDMM code	$N \frac{ts}{mp}$	$R_c \frac{tr}{mn}$	$pmn + pm + n$	Theorem 1

query $q_i^{(\theta)}$, server i first encodes the library into the following matrix

$$\sum_{t=0}^{L-1} g_t(q_i^{(\theta)}[t]) = \sum_{j=0}^{p-1} \sum_{k=0}^{n-1} B_{j,k}^{(\theta)} a_{\theta,i}^{\beta_{j,k}} + \sum_{t=0, t \neq \theta}^{L-1} \sum_{j=0}^{p-1} \sum_{k=0}^{n-1} B_{j,k}^{(t)} a_t^{\beta_{j,k}}.$$

For convenience, let

$$\begin{aligned}
g(x) &= g_\theta(x) + \sum_{t=0, t \neq \theta}^{L-1} g_t(q_i^{(\theta)}[t]) \\
&= \sum_{j=0}^{p-1} \sum_{k=0}^{n-1} B_{j,k}^{(\theta)} x^{\beta_{j,k}} + \underbrace{\sum_{t=0, t \neq \theta}^{L-1} \sum_{j=0}^{p-1} \sum_{k=0}^{n-1} B_{j,k}^{(t)} a_t^{\beta_{j,k}}}_{\text{constant}}.
\end{aligned}$$

Now it is obvious that $g(a_{\theta,i}) = \sum_{t=0}^{L-1} g_t(q_i^{(\theta)}[t])$.

After encoding the library, server i computes $f(a_{\theta,i})g(a_{\theta,i})$ and then sends the result back to the user. Let $h(x) = f(x)g(x)$, i.e.,

$$\begin{aligned}
h(x) &= \sum_{k=0}^{m-1} \sum_{j=0}^{p-1} \sum_{j'=0}^{p-1} \sum_{k'=0}^{n-1} A_{k,j} B_{j',k'}^{(\theta)} x^{\alpha_{k,j}+\beta_{j',k'}} \\
&+ \sum_{j'=0}^{p-1} \sum_{k'=0}^{n-1} ZB_{j',k'}^{(\theta)} x^{\gamma+\beta_{j',k'}} \\
&+ \left(\sum_{k=0}^{m-1} \sum_{j=0}^{p-1} A_{k,j} x^{\alpha_{k,j}} + Zx^\gamma \right) \left(\sum_{t=0, t \neq \theta}^{L-1} \sum_{j=0}^{p-1} \sum_{k=0}^{n-1} B_{j,k}^{(t)} a_t^{\beta_{j,k}} \right), \quad (8)
\end{aligned}$$

then the results returned from the servers are exactly the evaluations of $h(x)$ at some evaluation points, thus we can derive the following result.

Theorem 1: Let $\gamma = 0$, $\alpha_{k,j} = j + kp + 1$, and $\beta_{j,k'} = pm - j + k'(pm + 1)$ for $k \in [0, m)$, $j \in [0, p)$, and $k' \in [0, n)$. The multiplication of A and $B^{(\theta)}$ can be securely computed with upload cost $N \frac{ts}{mp}$ and download cost $R_c \frac{tr}{mn}$, where $R_c = pmn + pm + n$.

Proof: It is easy to check that

(i) For $k \in [0, m)$ and $k' \in [0, n)$,

$$\alpha_{k,0} + \beta_{0,k'} = \dots = \alpha_{k,p-1} + \beta_{p-1,k'}.$$

(ii) $U = \{\alpha_{k,0} + \beta_{0,k'} | 0 \leq k < m, 0 \leq k' < n\}$ is disjoint with

$$\begin{aligned}
I = & \{\alpha_{k,j} + \beta_{j',k'} | 0 \leq k < m, 0 \leq j \neq j' < p, 0 \leq k' < n\} \\
& \cup \{\gamma + \beta_{j',k'} | 0 \leq j' < p, 0 \leq k' < n\} \\
& \cup \{\alpha_{k,j} | 0 \leq k < m, 0 \leq j < p\} \cup \{\gamma\},
\end{aligned}$$

and $|U| = mn$.

Therefore, we can conclude that each $C_{k,k'}$ appears in $h(x)$ by (i), and is the coefficient of a unique term in $h(x)$ by (ii). It is also easy to obtain $R_c = \deg(h(x)) + 1$, which guarantees the decodability. Since $\alpha_{0,0}, \dots, \alpha_{m-1,p-1}, \gamma$ are pairwise distinct, then $I(\{f(a_i)\}; A) = 0$ for any $i \in [0, N)$ and $a_i \in \mathbb{F}_q$, i.e., security is guaranteed. The privacy condition follows from three conditions: 1) the definition of the query in Eq. (7) for the desired index $\theta \in [0, L)$, 2) the matrices $B^{(0)}, B^{(1)}, \dots, B^{(L-1)}$ are independent of θ , and 3) the matrix $\tilde{A}_i$ is random to each server as it is masked by the random

TABLE III
A COMPARISON OF THE NORMALIZED UPLOAD AND DOWNLOAD COSTS BETWEEN THE PROPOSED PSDMM CODE AND CHANG–TANDON’S CODE.

	Normalized upload cost	Normalized download cost	Recovery threshold R_c	References
Chang–Tandon’s code	R_c/m	$\frac{m+1}{m} \left(1 + \frac{m+1}{R_c} + \dots + \left(\frac{m+1}{R_c}\right)^{L-1}\right)$	$\geq m + 1$	[35]
The new PSDMM code	N/mp	R_c/mn	$pmn + pm + n$	Theorem 1

matrix Z according to (6). The detailed and rigorous proof is similar to [18], thus we omit it here.

It is obvious that the upload cost is $\sum_{i=0}^{L-1} H(f(a_i)) = N \frac{ts}{mp}$ and the download cost is $R \frac{tr}{mn}$. This finishes the proof. ■

IV. COMPARISON

Next, we provide comparisons of some key parameters of the proposed PSDMM code and some previous codes. Table II illustrates the first comparison. To this end, note that Chang–Tandon’s code [35] cannot mitigate stragglers and the number of servers can be an arbitrary integer larger than m but should be prefixed. Table III gives a (fairer) comparison of the normalized parameters of the proposed PSDMM code and Chang–Tandon’s code, where the *normalized upload cost* is the ratio of upload cost to $H(A)$ and the *normalized download cost* is the ratio of the download cost to $H(AB^{(i)})$. Figure 1 visualizes the comparison in Table III by assuming $N = R_c$ in the new PSDMM code.

TABLE IV

A COMPARISON OF THE RECOVERY THRESHOLDS BETWEEN THE PSDMM CODE IN [13] AND THE NEW PSDMM CODE FOR SOME SMALL p, m, n .

p	m	n	Upper bound constructions for $R(p, m, n)$	R_c^{BC} [13]	R_c^{New}
2	2	2	7 [39]	15	14
2	2	3	11 [38]	23	19
2	3	3	15 [40]	31	27
3	3	3	23 [41]	47	39
4	4	4	49 [39]	99	84
5	5	5	99 [42]	199	155
6	6	6	160 [43]	321	258
7	7	7	250 [44]	501	399
8	8	8	343 [39]	687	584
9	9	9	520 [44]	1041	819

From Tables II and III, the numerical comparison in Table IV, and Figure 1, we can see that the new PSDMM code has some advantages over the previous constructions:

- The new PSDMM code is more general than Kim–Lee’s code in [18]. Namely, it can provide a more flexible tradeoff between the upload cost and the download cost. In particular, when $p = 1$, the recovery threshold of the new code is one smaller than that of Kim–Lee’s code in [18] under the same upload cost.
- The recovery threshold and the download cost of the new PSDMM code are smaller than those of Aliasgari *et al.*’s PSGPD code in [12] under the same upload cost² and the PSDMM code in [13] for some small parameters.

²In [12], there was a small mistake in the recovery threshold of the PSGPD code. We give the correct value of the recovery threshold in Table II.

Fig. 1. Comparison of the normalized upload and download costs between the new PSDMM code and the one by Chang and Tandon under the parameters $N = R_c = 11$.

- The download cost of Chang–Tandon’s PSDMM code in [35] depends on L and is a monotonically increasing function w.r.t. L (see the trends in Figure 1). Besides, Chang–Tandon’s PSDMM code cannot mitigate stragglers and requires a large sub-packetization degree. In contrast, the new PSDMM code does not have such defects. Furthermore, Figure 1 shows that the new PSDMM code has a smaller normalized download cost for some given normalized upload cost compared with Chang–Tandon’s PSDMM code.

V. CONCLUSION

We considered the problem of PSDMM and proposed a scheme based on [12] and the entangled polynomial code in [13]. The performance of the scheme was characterized via a degree table. Alongside, an efficient degree table was given, leading to a new PSDMM scheme that can provide a flexible tradeoff between the upload and download costs, and which has better performance than some previous codes in terms of the recovery threshold and download cost. An interesting future direction is to consider the case where the servers share a library of L matrices in an encoded form, a natural starting point being maximum distance separable (MDS) coded data.

REFERENCES

- [1] G. Joshi, E. Soljanin, and G. Wornell, "Efficient redundancy techniques for latency reduction in cloud systems," *ACM Transactions on Modeling and Performance Evaluation of Computing Systems (TOMPECS)*, vol. 2, no. 2, pp. 1–30, 2017.
- [2] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran, "Speeding up distributed machine learning using codes," *IEEE Transactions on Information Theory*, vol. 64, no. 3, pp. 1514–1529, 2017.
- [3] Q. Yu, M. Maddah-Ali, and S. Avestimehr, "Polynomial codes: an optimal design for high-dimensional coded matrix multiplication," in *Advances in Neural Information Processing Systems*, 2017, pp. 4403–4413.
- [4] S. Dutta, Z. Bai, H. Jeong, T. M. Low, and P. Grover, "A unified coded deep neural network training strategy based on generalized polydot codes," in *2018 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2018, pp. 1585–1589.
- [5] Z. Jia and S. A. Jafar, "Cross subspace alignment codes for coded distributed batch computation," *IEEE Transactions on Information Theory*, vol. 67, no. 5, pp. 2821–2846, 2021.
- [6] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Straggler mitigation in distributed matrix multiplication: Fundamental limits and optimal coding," *IEEE Transactions on Information Theory*, vol. 66, no. 3, pp. 1920–1933, 2020.
- [7] R. G. D'Oliveira, S. El Rouayheb, and D. Karpuk, "GASP codes for secure distributed matrix multiplication," *IEEE Transactions on Information Theory*, vol. 66, no. 7, pp. 4038–4050, 2020.
- [8] W.-T. Chang and R. Tandon, "On the capacity of secure distributed matrix multiplication," in *2018 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2018, pp. 1–6.
- [9] J. Kakar, S. Ebadifar, and A. Sezgin, "On the capacity and straggler-robustness of distributed secure matrix multiplication," *IEEE Access*, vol. 7, pp. 45 783–45 799, 2019.
- [10] H. Yang and J. Lee, "Secure distributed computing with straggling servers using polynomial codes," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 1, pp. 141–150, 2019.
- [11] Z. Jia and S. A. Jafar, "On the capacity of secure distributed matrix multiplication," *arXiv preprint arXiv:1908.06957*, 2019.
- [12] M. Aliasgari, O. Simeone, and J. Kliewer, "Private and secure distributed matrix multiplication with flexible communication load," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 2722–2734, 2020.
- [13] Q. Yu and A. S. Avestimehr, "Entangled polynomial codes for secure, private, and batch distributed matrix multiplication: Breaking the 'cubic' barrier," in *2020 IEEE International Symposium on Information Theory (ISIT)*. IEEE, 2020, pp. 245–250.
- [14] J. Zhu and X. Tang, "Secure batch matrix multiplication from grouping lagrange encoding," *IEEE Communications Letters*, vol. 25, no. 4, pp. 1119–1123, 2021.
- [15] S. Chentharu, K. Ahmed, H. Wang, and F. Whittaker, "Security and privacy-preserving challenges of e-health solutions in cloud computing," *IEEE access*, vol. 7, pp. 74 361–74 382, 2019.
- [16] W. Al Amiri, M. Baza, K. Banawan, M. Mahmoud, W. Alasmay, and K. Akkaya, "Privacy-preserving smart parking system using blockchain and private information retrieval," in *2019 International Conference on Smart Applications, Communications and Networking (SmartNets)*. IEEE, 2019, pp. 1–6.
- [17] A. Bampoulidis, A. Bruni, L. Helming, D. Kales, C. Rechberger, and R. Walch, "Privately connecting mobility to infectious diseases via applied cryptography," *arXiv preprint arXiv:2005.02061*, 2020.
- [18] M. Kim and J. Lee, "Private secure coded computation," *IEEE Communications Letters*, vol. 23, no. 11, pp. 1918–1921, 2019.
- [19] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, "Private information retrieval," in *Proceedings of IEEE 36th Annual Foundations of Computer Science*. IEEE, 1995, pp. 41–50.
- [20] H. Sun and S. A. Jafar, "The capacity of private information retrieval," *IEEE Transactions on Information Theory*, vol. 63, no. 7, pp. 4075–4088, 2017.
- [21] —, "The capacity of robust private information retrieval with colluding databases," *IEEE Transactions on Information Theory*, vol. 64, no. 4, pp. 2361–2370, 2017.
- [22] K. Banawan and S. Ulukus, "The capacity of private information retrieval from coded databases," *IEEE Transactions on Information Theory*, vol. 64, no. 3, pp. 1945–1956, 2018.
- [23] R. Tajeddine, O. W. Gnilke, and S. El Rouayheb, "Private information retrieval from MDS coded data in distributed storage systems," *IEEE Transactions on Information Theory*, vol. 64, no. 11, pp. 7081–7093, 2018.
- [24] R. Freij-Hollanti, O. W. Gnilke, C. Hollanti, and D. A. Karpuk, "Private information retrieval from coded databases with colluding servers," *SIAM Journal on Applied Algebra and Geometry*, vol. 1, no. 1, pp. 647–664, 2017.
- [25] R. Freij-Hollanti, O. W. Gnilke, C. Hollanti, A.-L. Horlemann-Trautmann, D. Karpuk, and I. Kubjas, " t -private information retrieval schemes using transitive codes," *IEEE Transactions on Information Theory*, vol. 65, no. 4, pp. 2107–2118, 2018.
- [26] S. Kumar, H.-Y. Lin, E. Rosnes, and A. G. i Amat, "Achieving maximum distance separable private information retrieval capacity with linear codes," *IEEE Transactions on Information Theory*, vol. 65, no. 7, pp. 4243–4273, 2019.
- [27] J. Zhu, Q. Yan, C. Qi, and X. Tang, "A new capacity-achieving private information retrieval scheme with (almost) optimal file length for coded servers," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 1248–1260, 2019.
- [28] R. Zhou, C. Tian, H. Sun, and T. Liu, "Capacity-achieving private information retrieval codes from MDS-coded databases with minimum message size," *IEEE Transactions on Information Theory*, vol. 66, no. 8, pp. 4904–4916, 2020.
- [29] 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, no. 12, pp. 7427–7438, 2020.
- [30] Q. Wang and M. Skoglund, "Symmetric private information retrieval from MDS coded distributed storage with non-colluding and colluding servers," *IEEE Transactions on Information Theory*, vol. 65, no. 8, pp. 5160–5175, 2019.
- [31] R. G. D'Oliveira and S. El Rouayheb, "One-shot pir: Refinement and lifting," *IEEE Transactions on Information Theory*, vol. 66, no. 4, pp. 2443–2455, 2019.
- [32] J. Li, D. Karpuk, and C. Hollanti, "Towards practical private information retrieval from MDS array codes," *IEEE Transactions on Communications*, vol. 68, no. 6, pp. 3415–3425, 2020.
- [33] L. Holzbaur, R. Freij-Hollanti, J. Li, and C. Hollanti, "Towards the capacity of private information retrieval from coded and colluding servers," *arXiv preprint arXiv:1903.12552*, 2019.
- [34] R. G. D'Oliveira, S. E. Rouayheb, D. Heinlein, and D. Karpuk, "Notes on communication and computation in secure distributed matrix multiplication," in *2020 IEEE Conference on Communications and Network Security (CNS)*. IEEE, 2020, pp. 1–6.
- [35] W.-T. Chang and R. Tandon, "On the upload versus download cost for secure and private matrix multiplication," in *2019 IEEE Information Theory Workshop (ITW)*. IEEE, 2019, pp. 1–5.
- [36] H. Sun and S. A. Jafar, "The capacity of private computation," *IEEE Transactions on Information Theory*, vol. 65, no. 6, pp. 3880–3897, 2018.
- [37] Q. Yu, S. Li, N. Raviv, S. M. M. Kalan, M. Soltanolkotabi, and S. A. Avestimehr, "Lagrange coded computing: Optimal design for resiliency, security, and privacy," in *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 2019, pp. 1215–1225.
- [38] M. Bläser, "Fast matrix multiplication," *Theory of Computing*, pp. 1–60, 2013.
- [39] V. Strassen, "Gaussian elimination is not optimal," *numerical mathematics*, vol. 13, no. 4, pp. 354–356, 1969.
- [40] P. Bürgisser, M. Clausen, and M. A. Shokrollahi, *Algebraic complexity theory*. Springer Science & Business Media, 2013, vol. 315.
- [41] J. D. Laderman, "A noncommutative algorithm for multiplying 3×3 matrices using 23 multiplications," *Bulletin of the American Mathematical Society*, vol. 82, no. 1, pp. 126–128, 1976.
- [42] A. Sedoglavic, "A non-commutative algorithm for multiplying 5×5 matrices using 99 multiplications," *arXiv preprint arXiv:1707.06860*, 2017.
- [43] A. V. Smirnov, "The bilinear complexity and practical algorithms for matrix multiplication," *Computational Mathematics and Mathematical Physics*, vol. 53, no. 12, pp. 1781–1795, 2013.
- [44] A. Sedoglavic, "A non-commutative algorithm for multiplying (7×7) matrices using 250 multiplications," *arXiv preprint arXiv:1712.07935*, 2017.