DOI:
10.1109/TIFS.2020.2972166

Document Version
Peer reviewed version

Link to publication record in King's Research Portal

Citation for published version (APA):
Aliasgari, M., Simeone, O., & Kliewer, J. (2020). Private and Secure Distributed Matrix Multiplication with Flexible Communication Load. *IEEE Transactions on Information Forensics and Security*, *15*, 2722-2734. Article 8985291. Advance online publication. https://doi.org/10.1109/TIFS.2020.2972166

Private and Secure Distributed Matrix Multiplication with Flexible Communication Load

Malihe Aliasgari*, **Osvaldo Simeone**† and **Jörg Kliewer***
* Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, U.S.A.
† Department of Engineering, King's College London, Department of Engineering, London, U.K.

Abstract—**Large matrix multiplications are central to large-scale machine learning applications. These operations are often carried out on a distributed computing platform with a master server and multiple workers in the cloud operating in parallel. For such distributed platforms, it has been recently shown that coding over the input data matrices can reduce the computational delay, yielding a trade-off between recovery threshold, i.e., the number of workers required to recover the matrix product, and communication load, i.e., the total amount of data to be downloaded from the workers. In this paper, in addition to exact recovery requirements, we impose security and privacy constraints on the data matrices, and study the recovery threshold as a function of the communication load. We first assume that both matrices contain private information and that workers can collude to eavesdrop on the content of these data matrices. For this problem, we introduce a novel class of secure codes, referred to as secure generalized PolyDot (SGPD) codes, that generalize state-of-the-art non-secure codes for matrix multiplication. SGPD codes allow a flexible trade-off between recovery threshold and communication load for a fixed maximum number of colluding workers while providing perfect secrecy for the two data matrices. We then study a connection between secure matrix multiplication and private information retrieval. We specifically assume that one of the data matrices is taken from a public set known to all the workers. In this setup, the identity of the matrix of interest should be kept private from the workers. For this model, we present a variant of generalized PolyDot codes that can guarantee both secrecy of one matrix and privacy for the identity of the other matrix for the case of no colluding servers.**

Index Terms—**Coded distributed computation, distributed learning, secret sharing, information theoretic security, private information retrieval.**

I. Introduction

A. Motivation and Problem Definition

At the core of many signal processing and machine learning applications are tensor operations, most notably large matrix multiplications [2]. In the presence of practically sized data sets, such operations are typically carried out using distributed computing platforms with a master server and multiple workers that can operate in parallel over distinct parts of the data set. The master server plays the role of the parameter server, distributing data to the workers and periodically reconciling their internal state [3]. Workers are commercial off-the-shelf servers that are characterized by possible temporary failures and delays [4].

Straggling workers can affect the computation latency by orders of magnitude, e.g., [5], [6]. While current distributed computing platforms conventionally handle straggling servers by means of replication of computing tasks [7], recent work has shown that encoding the input data can help reduce the computation latency. More generally, coding is able to control the trade-off between computational delay and communication load between workers and master server [8]–[17]. Furthermore, stochastic coding can help keeping both input and output data secure from the workers, assuming that the latter are honest, i.e., carrying out the prescribed protocol, but curious [18]–[25]. This paper contributes to this line of work by investigating the trade-off between computational delay and communication load as a function of the privacy level.

As illustrated in Figs. 1 and 2, we focus on the basic problem of computing a matrix multiplication $\mathbf{C} = \mathbf{AB}$ in a distributed computing system of P workers that can process each only a fraction $1/m$ and $1/n$ of matrices $\mathbf{A}$ and $\mathbf{B}$, respectively. In the first setup under study, illustrated in Fig. 1, both matrices $\mathbf{A}$ and $\mathbf{B}$ are to be kept private from the workers. Here, three performance criteria are of interest:

- the recovery threshold P_R, that is, the number of workers that need to complete their task before the master server can recover the product $\mathbf{C}$;
- the communication load C_L between workers and master server, i.e., the amount of information to be downloaded from the workers;
- the maximum number P_C of colluding servers that ensures perfect secrecy for both data matrices $\mathbf{A}$ and $\mathbf{B}$.

In the second setup of interest shown in Fig. 2, only matrix $\mathbf{A}$ is private, while matrix $\mathbf{B}$ is selected from a public data set $\mathcal{B}$. In this case, apart from the security constraint on $\mathbf{A}$, we only impose a privacy constraint on the identity of the specific matrix $\mathbf{B} \in \mathcal{B}$ of interest. As a motivation for this second setup, consider a recommender system based on collaborative filtering [26]. In this case, recommendations are based on the product of two matrices, one describing the profile of a user, or a group of users, and one representing features of the items of interest, such as movies, music or TV shows. The users' profile matrix can be modelled by the private matrix

This work was supported in part by the European Research Council (ERC) under the European Union Horizon 2020 research and innovative programme (grant agreement No 725731) and by U.S. NSF grants CNS-1526547, CCF-1525629.

Part of this paper was presented at the IEEE International Symposium on Information Theory (ISIT), 2019 [1].

M. Aliasgari and J. Kliewer are with Helen and John C. Hartmann Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, New Jersey, USA (email: ma839@njit.edu; jkliewer@njit.edu).

O. Simeone is with King's Centre for Learning & Information Processing (kclip), Centre for Telecommunication Research (CTR), the Department of Engineering, King's College London, London, UK (email: osvaldo.simeone@kcl.ac.uk).

$\mathbf{A}$, hence ensuring the privacy of users' data; while the items' data matrix for each category is represented by one of the matrices in the public data set $\mathcal{B} = \{\mathbf{B}^{(k)}\}_{k=1}^{L}$. This latter assumption captures the constraint that users may want to keep the confidential types of items they are interested in. For this problem, the criteria of interest are still P_R and P_C, and we simplify the problem by setting $P_C = 1$. This paper focuses on the design of coding and computing techniques for both problems.

B. Related Work

In order to put our contribution in perspective, we briefly review prior related work. Consider first solutions that provide no security guarantees, i.e., $P_C = 0$, for the problem in Fig. 1. As a direct extension of [8], a first approach is to use product codes that apply separately the maximum distance separable (MDS) codes to encode the two matrices [27]. The recovery threshold of this scheme is improved by [9], which introduces *polynomial codes*. The construction in [9] is proved to be optimal under the assumption that *minimal communication* is allowed between workers and master server. In [15], MatDot codes are introduced, resulting in a lower recovery threshold at the expense of a larger communication load. The construction in [13] bridges the gap between polynomial and MatDot codes and presents PolyDot codes, yielding a trade-off between recovery threshold and communication load. An extension of this scheme, termed Generalized PolyDot (GPD) codes improves on the recovery threshold of PolyDot codes [14], which is independently obtained also by the construction in [28]. In [14], GPD codes are used to design a unified coded computing strategy for the training of deep neural networks.

Much less work has been done in the literature for the case in which security constraints are factored in, i.e., where $P_C \neq 0$, for the problem of Fig. 1. In [19], Lagrange coding is presented that achieves the minimum recovery threshold for multilinear functions by generalizing MatDot codes. In [18], [25], coded schemes have been used to develop multi-party computation techniques to calculate arbitrary polynomials of massive matrices, preserving the security of the data matrices. In [20], [21], [23] a reduction of the communication load is obtained by extending polynomial codes. While these works focus on either minimizing recovery threshold or communication load, the *trade-off* between these two fundamental quantities has not been addressed in the open literature to the best of our knowledge. A new class of secure distributed matrix multiplication and its capacity is studied in [29].

In the second part of this work, we study a connection between secure matrix multiplication and private information retrieval (PIR), as illustrated in Fig. 2. The PIR problem was introduced in [30] and has been widely studied in recent years, e.g., in [31]–[40]. In [38] and [39] the PIR setup was investigated for the problem of distributed matrix multiplication illustrated in Fig. 2 that imposes PIR guarantees for the index of matrix $\mathbf{B}$ within a public library. In [38], a coding strategy is proposed that combines the PIR scheme for non-colluding servers (i.e., with $P_C = 1$) [30] with polynomial codes [9]. In [39], the authors introduce a related approach for this problem, and show that it outperforms the scheme proposed in [38] in terms of upload and download cost. The code design in [39] focuses on the minimization of the communication load, and does not explore the trade-off between this metric and the recovery threshold.

C. Main Contribution

In this paper we first present a novel class of secure computation codes, referred to as secure GPD (SGPD) codes, for the setup in Fig. 1, SGPD codes generalize GPD codes to operate at a flexible communication load level. This yields a new achievable trade-off between recovery threshold P_R and communication load C_L as a function of a prescribed number of colluding workers P_C. In the process, we also introduce a novel perspective on distributed computing codes based on the signal processing concepts of convolution and z-transform. SGPD codes were first introduced in the conference version of this paper [1], which did not contain complete proofs and provided only limited illustrations and examples. Then, SGPD codes are modified to offer a solution, introduced here for the first time, for the scenario in Fig. 2. This is done through concatenation with the PIR code in [38], which ensures both secrecy of the input matrix $\mathbf{A}$ and privacy of the identity for the desired matrix in the library $\mathcal{B}$ if $P_C = 1$. The resulting codes are referred to as private and secure GPD (PSGPD) codes. They generalize the approach in [39], enabling a trade-off between (upload) communication load and recovery threshold. We finally illustrate the benefits of the proposed codes, which offer a flexible trade-off between communication load and recovery threshold, by analyzing the overall completion time due to both computation and communication.

D. Organization

The rest of the paper is organized as follows. In Section II, we present the system models for secure matrix multiplication (Fig. 1 in Section II-C) and for private and secure matrix multiplication (Fig. 2 in Section II-D), respectively. In Section III we propose an intuitive interpretation of the GPD code introduced in [15]. Using z-transforms, Section IV proposes a novel extension of GPD codes by imposing a security constraint on the data matrices and deriving the resulting trade-off between recovery threshold P_R and communication load C_L. In this section, we also study overall completion latency encompassing both computation and communication latencies for SGPD codes. In Section V, we address the setup in Fig. 2, again with respect to the trade-off between P_R and C_L and to the overall completion latency. The paper is concluded in Section VI.

II. Problem Statement

A. Notation

Throughout the paper, we denote a matrix with upper boldface letters (e.g., $\mathbf{X}$), and lower boldface letters indicate a vector or a sequence of matrices (e.g., $\mathbf{x}$). Furthermore, a math calligraphic font refers to a set (e.g., $\mathcal{X}$). A set $\mathbb{F}$ represents the Galois field with cardinality $|\mathbb{F}|$. We denote by $\mathbb{N}$ the set of all non-zero positive integers, and for some $a, b \in \mathbb{N}$, $a \leq b$,

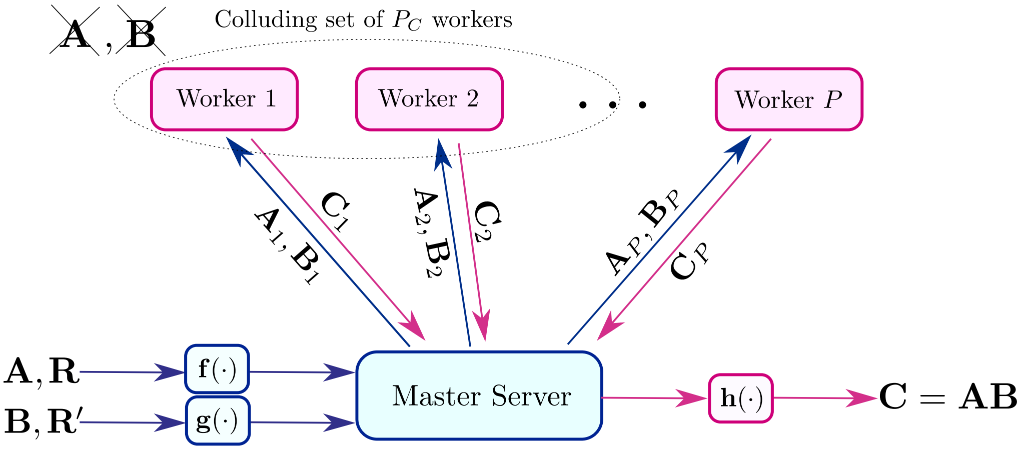
$\cancel{\mathbf{A}}, \cancel{\mathbf{B}}$
Colluding set of P_C workers
Worker 1
Worker 2
Worker P
$\mathbf{A}_1, \mathbf{B}_1$
$\mathbf{C}_1$
$\mathbf{A}_2, \mathbf{B}_2$
$\mathbf{C}_2$
$\mathbf{A}_P, \mathbf{B}_P$
$\mathbf{C}_P$
$\mathbf{A}, \mathbf{R}$
$\mathbf{B}, \mathbf{R}'$
$\mathbf{f}(\cdot)$
$\mathbf{g}(\cdot)$
Master Server
$\mathbf{h}(\cdot)$
$\mathbf{C} = \mathbf{AB}$

Fig. 1: Secure matrix multiplication: the master server encodes both input matrices $\mathbf{A}$ and $\mathbf{B}$, to be kept secure from the workers, and both random matrices $\mathbf{R}$ and $\mathbf{R}'$, respectively, to define the computational tasks of the slave servers or workers. The workers may fail or straggle, and they are honest but curious, with colluding subsets of workers of size at most P_C. The master server must be able to decode the product $\mathbf{C} = \mathbf{AB}$ from the output of a subset of P_R servers, which defines the recovery threshold.

$[a, b] \triangleq \{a, a+1, \ldots, b\}$. For any real number r, $\lceil r \rceil$ represents the largest integer nearest to r. he function $H(\cdot)$ represents the entropy of its argument, and $I(X;Y)$ denotes the mutual information of the random variables X and Y.

B. System Model

As illustrated in Figs. 1 and 2, we consider a distributed computing system with a master server and P slave servers or workers. The master server is interested in computing securely the matrix product $\mathbf{C} = \mathbf{AB}$ of two data matrices $\mathbf{A}$ and $\mathbf{B}$ with dimensions $T \times S$ and $S \times D$, respectively. The matrices have i.i.d. uniformly distributed entries from a sufficient large finite field $\mathbb{F}$, with $|\mathbb{F}| > P$. More precisely, we will consider two scenarios. In the first, both matrices $\mathbf{A}$ and $\mathbf{B}$ are available at the master server and contain confidential data that should be kept secure from the workers (see Fig. 1). In the second, only matrix $\mathbf{A}$ contains confidential information, and there are L public matrices in the set $\mathcal{B} = \{\mathbf{B}^{(r)}\}_{r=1}^{L}$ from which the master node wishes to compute the product $\mathbf{C}^{(\kappa)} = \mathbf{AB}^{(\kappa)}$ for some κth index $\kappa \in [1, L]$. The index must be kept private against the workers (see Fig. 2). In the following, we first describe the system model for the setup in Fig. 1, referred to as *secure matrix multiplication*, followed by the setup for the model in Fig. 2, referred to as *private and secure matrix multiplication*.

C. Secure Matrix Multiplication

For the scenario in Fig. 1 workers receive information on matrices $\mathbf{A} \in \mathbb{F}^{T \times S}$ and $\mathbf{B} \in \mathbb{F}^{S \times D}$ from the master server; they process this information and they respond to the master server, which finally recovers the product $\mathbf{C} = \mathbf{AB}$ with minimal computational effort. Due to communication and complexity constraints, each worker can receive only TS/m and SD/n symbols, respectively, for some integers m and n. The workers are honest but curious. Accordingly, we impose the secrecy constraint that, even if up to $P_C < P$ workers collude, the workers cannot obtain any information about both matrices $\mathbf{A}$ and $\mathbf{B}$ based on the data received from the master server.

To keep the data secure and to leverage possible computational redundancy at the workers (namely, if $P/m > 1$ and/or $P/n > 1$), the master server sends encoded versions of the input matrices to the workers due to the above mentioned communication and complexity constraints. Specifically, it produces the encoded matrices $\mathbf{A}_p = \mathbf{f}_p(\mathbf{A}, \mathbf{R})$, where $\mathbf{R}$ is a random matrix of dimension $T' \times S'$, for some integers T' and S' to be defined below, via the function

$$\mathbf{f}_p : \mathbb{F}^{T \times S} \times \mathbb{F}^{T' \times S'} \rightarrow \mathbb{F}^{T/t \times S/s}, \tag{1}$$

for some integers t and s such that $m = st$. The resulting TS/m entries in the output of function $\mathbf{f}_p$ are then sent to worker p, with $p \in [1, P]$. Likewise, the master server computes the encoded matrices $\mathbf{B}_p = \mathbf{g}_p(\mathbf{B}, \mathbf{R}')$, where $\mathbf{R}'$ is a random matrix of dimension $S' \times D'$, for some integers S' and D' to be defined below, using the function

$$\mathbf{g}_p : \mathbb{F}^{S \times D} \times \mathbb{F}^{S' \times D'} \rightarrow \mathbb{F}^{S/s \times D/d}, \tag{2}$$

for some integers s and d such that $n = sd$. The resulting SD/n entries in $\mathbf{B}_p$ are then sent to worker p. The random matrices $\mathbf{R}$ and $\mathbf{R}'$ consists of i.i.d. uniformly distributed entries from a field $\mathbb{F}$. The security constraint imposes the condition

$$I(\mathbf{A}_{\mathcal{P}}, \mathbf{B}_{\mathcal{P}}; \mathbf{A}, \mathbf{B}) = 0, \tag{3}$$

for all subsets of $\mathcal{P} \subset [1, P]$ of P_C workers, where the random matrices $\mathbf{R}$ and $\mathbf{R}'$ serve as random keys in order to meet the security constraint (3) [41].

Each worker p computes the product $\mathbf{C}_p = \mathbf{A}_p \mathbf{B}_p$ of the encoded sub-matrices $\mathbf{A}_p$ and $\mathbf{B}_p$. The master server collects a subset of $P_R \leq P$ outputs from the workers as defined by the subset $\{\mathbf{C}_p\}_{p \in \mathcal{P}_R}$ with $|\mathcal{P}_R| = P_R$. It then applies a decoding function as $\mathbf{h}\left(\{\mathbf{C}_p\}_{p \in \mathcal{P}_R}\right)$,

$$\mathbf{h} : \underbrace{\mathbb{F}^{T/t \times D/d} \times \cdots \times \mathbb{F}^{T/t \times D/d}}_{P_R \text{ times}} \rightarrow \mathbb{F}^{T \times D}. \tag{4}$$

Note that *correct decoding* translates into the condition

$$H(\mathbf{AB} | \{\mathbf{C}_p\}_{p \in \mathcal{P}_R}) = 0. \tag{5}$$

A coding and decoding strategy that satisfies condition (3) and (5) is said to be *feasible*.

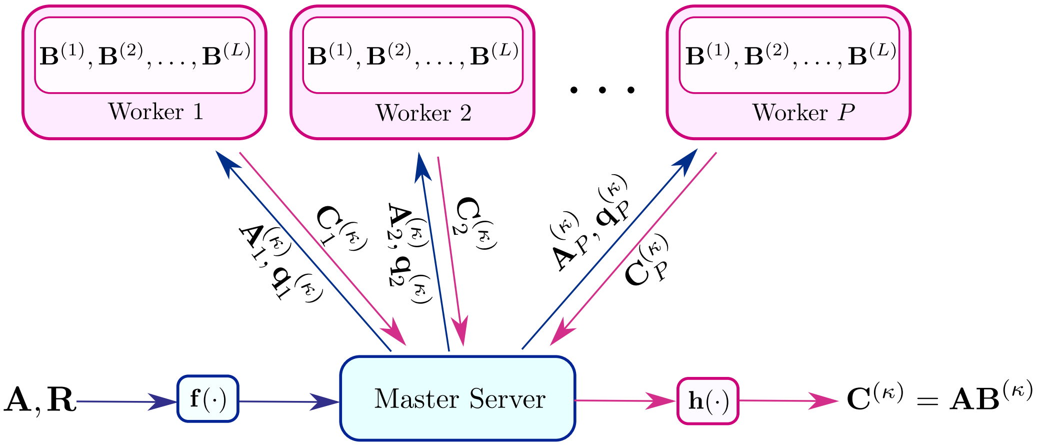
$\mathbf{B}^{(1)}, \mathbf{B}^{(2)}, \dots, \mathbf{B}^{(L)}$
Worker 1
$\mathbf{B}^{(1)}, \mathbf{B}^{(2)}, \dots, \mathbf{B}^{(L)}$
Worker 2
...
$\mathbf{B}^{(1)}, \mathbf{B}^{(2)}, \dots, \mathbf{B}^{(L)}$
Worker P
$\mathbf{A}_1^{(\kappa)}, \mathbf{q}_1^{(\kappa)}$
$\mathbf{C}_1^{(\kappa)}$
$\mathbf{A}_2^{(\kappa)}, \mathbf{q}_2^{(\kappa)}$
$\mathbf{C}_2^{(\kappa)}$
$\mathbf{A}_P^{(\kappa)}, \mathbf{q}_P^{(\kappa)}$
$\mathbf{C}_P^{(\kappa)}$
$\mathbf{A}, \mathbf{R}$
$\mathbf{f}(\cdot)$
Master Server
$\mathbf{h}(\cdot)$
$\mathbf{C}^{(\kappa)} = \mathbf{A}\mathbf{B}^{(\kappa)}$

Fig. 2: Private and secure matrix multiplication: the master server encodes the input matrix $\mathbf{A}$, to be kept secret from the workers, and generates the encoded matrix $\mathbf{A}_p^{(\kappa)}$ for each worker p. It also sends a query $\mathbf{q}_p^{(\kappa)}$ as a function of the index $\kappa \in [1, L]$, to be kept private from workers, of the desired product $\mathbf{C}^{(\kappa)} = \mathbf{A}\mathbf{B}^{(\kappa)}$, with matrices $\{\mathbf{B}^{(r)}\}_{r=1}^{L}$ available at all workers. The non-colluding workers may fail or straggle, and they are honest but curious. The master server must be able to decode the product $\mathbf{C}^{(\kappa)}$ from the output of a subset of P_R servers, which defines the recovery threshold.

For given parameters m and n the performance of a coding and decoding scheme is measured by the triple (P_C, P_R, C_L), where C_L is defined as

$$C_L = \sum_{p \in \mathcal{P}_R} |\mathbf{C}_p|; \tag{6}$$

$|\mathbf{C}_p|$ is the dimension of the product matrix $\mathbf{C}_p$ computed by worker p. Note that condition (5) requires the inequality $\min\{P_R/m, P_R/n\} \geq 1$ or $P_R \geq P_{R,\min} \triangleq \max\{m, n\}$, which is hence a lower bound for the minimum recovery threshold. Furthermore, the communication load is lower bounded by $C_L \geq C_{L,\min} \triangleq TD$, which is the size of the product $\mathbf{C} = \mathbf{A}\mathbf{B}$.

D. Private and Secure Matrix Multiplication

In this subsection, we discuss the private and secure matrix multiplication problem illustrated in Fig. 2. In this setup, the master server wishes to compute the product $\mathbf{C}^{(\kappa)} = \mathbf{A}\mathbf{B}^{(\kappa)}$ of a confidential input matrix $\mathbf{A}$ with a matrix $\mathbf{B}^{(\kappa)}$ from a set of public matrices $\{\mathbf{B}^{(1)}, \dots, \mathbf{B}^{(L)}\}$, while keeping the index κ of the matrix $\mathbf{B}^{(\kappa)}$ of interest private from the workers.

Similar to the secure model in Fig. 1, we consider a distributed computing system with a master server and P honest but curious workers. The master server contains a confidential data matrix $\mathbf{A}$ with dimension $T \times S$. Each worker has access to the library $\mathcal{B}$, which consists of L distinct matrices $\{\mathbf{B}^{(1)}, \dots, \mathbf{B}^{(L)}\}$, each with dimension $S \times D$. As above, all matrices contain data symbols chosen uniformly i.i.d. from a sufficient large finite field $\mathbb{F}$, with $|\mathbb{F}| > P$. The master server is interested in computing the matrix product $\mathbf{C}^{(\kappa)} = \mathbf{A}\mathbf{B}^{(\kappa)}$ of the data matrix $\mathbf{A}$ and of a matrix $\mathbf{B}^{(\kappa)}$ for some index $\kappa \in [1, L]$. This should be done while keeping the data matrix $\mathbf{A}$ secret against the workers in the same sense as in the scenario of Fig. 1, while also ensuring that the index κ is kept secret from the workers.

To do so, as in the PIR problem [33], [34], the master server generates P query vectors $\mathbf{q}_1^{(\kappa)}, \dots, \mathbf{q}_P^{(\kappa)} \in \mathbb{F}^L$, for some $L > 1$ as a function of the desired index κ and sends each worker $p \in [1, P]$, the query vector $\mathbf{q}_p^{(\kappa)}$. We assume that the workers do not collude, i.e., we set $P_C = 1$. Extensions to any $P_C > 1$ are possible and are left for future work. We note that, when the input matrix $\mathbf{A}$ is an identity matrix, the setup reduces to a PIR problem.

To keep the data matrix $\mathbf{A}$ secure against workers, the master server sends each worker $p \in [1, P]$ an encoded version $\mathbf{A}_p^{(\kappa)} = \mathbf{f}_p(\kappa, \mathbf{A}, \mathbf{R}) \in \mathbb{F}^{T/t \times S/s}$ which is a function of index κ, and through it, of the query $\mathbf{q}_p^{(\kappa)}$, of the data matrix $\mathbf{A}$ and of a random matrix $\mathbf{R}$, for some integers t and s such that $m = ts$.

Upon receiving $(\mathbf{q}_p^{(\kappa)}, \mathbf{A}_p^{(\kappa)})$, each worker p uses the query $\mathbf{q}_p^{(\kappa)}$ to derive an $S/s \times D/d$ matrix $\mathbf{B}_p^{(\kappa)} = \mathbf{g}_p(\mathbf{q}_p^{(\kappa)}, \mathcal{B}) \in \mathbb{F}^{S/s \times D/d}$ from the library $\mathcal{B}$ by using an encoding function

$$\mathbf{g}_p : \mathbb{F}^L \times \underbrace{\mathbb{F}^{S \times D} \times \cdots \times \mathbb{F}^{S \times D}}_{L \text{ times}} \to \mathbb{F}^{S/s \times D/d}, \tag{7}$$

for some integers s and d such that $n = sd$. We emphasize that, unlike the setup considered in Fig. 1, the *content* of the desired matrix $\mathbf{B}^{(\kappa)}$ is not secure against workers, since the library $\mathcal{B}$ is public. Each worker p then computes the product $\mathbf{C}_p^{(\kappa)} = \mathbf{A}_p^{(\kappa)}\mathbf{B}_p^{(\kappa)}$ and sends it to the master server. The master server collects a subset $\{\mathbf{C}_p^{(\kappa)}\}_{p \in \mathcal{P}_R}$ of $P_R \leq P$ outputs from the workers with $|\mathcal{P}_R| = P_R$. It then applies a decoding function $\mathbf{h}(\{\mathbf{C}_p^{(\kappa)}\}_{p \in \mathcal{P}_R})$, as in (4), in order to retrieve the desired product $\mathbf{C}^{(\kappa)} = \mathbf{A}\mathbf{B}^{(\kappa)}$.

To guarantee the *secrecy of input matrix* $\mathbf{A}$, in a manner similar to (3), we have the constraint

$$I(\mathbf{A}_p^{(\kappa)}, \mathbf{B}_p^{(\kappa)}, \mathbf{q}_p^{(\kappa)}, \mathcal{B}; \mathbf{A}) = 0, \tag{8}$$

for all $p \in [1, P]$. Following the PIR formulation on [38], in order to ensure the *privacy of index* κ, for some value of κ the information available at each worker should be statistically indistinguishable from that available for any other value $\kappa' \neq \kappa$. Mathematically, for all $\kappa, \kappa' \in [1, L]$ with $\kappa' \neq \kappa$ and for all workers $p \in [1, P]$, we have the condition

$$(\mathbf{q}_p^{(\kappa)}, \mathbf{A}_p^{(\kappa)}, \mathbf{C}_p^{(\kappa)}, \mathcal{B}) \sim (\mathbf{q}_p^{(\kappa')}, \mathbf{A}_p^{(\kappa')}, \mathbf{C}_p^{(\kappa')}, \mathcal{B}), \tag{9}$$

that is, the joint distribution of variables $(\mathbf{q}_p^{(\kappa')}, \mathbf{A}_p^{(\kappa')}, \mathbf{C}_p^{(\kappa')}, \mathcal{B})$ should be the same for any pair

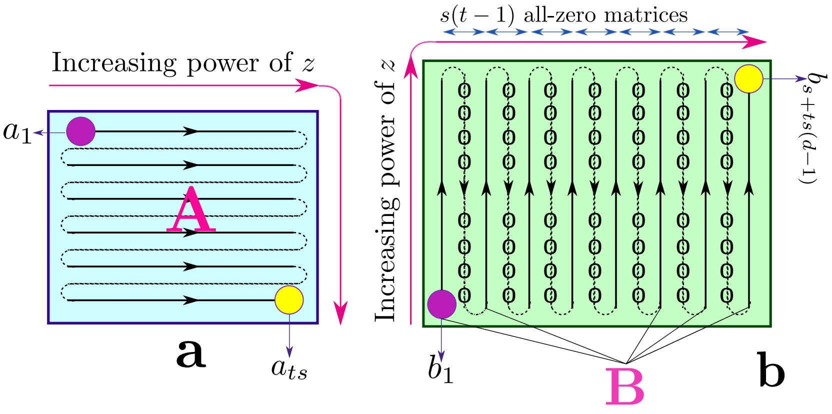

Fig. 3: Construction of the time sequences $\mathbf{a}$ and $\mathbf{b}$ used to define the generalized PolyDot (GPD) code. The zero dashed lines in $\mathbf{b}$ indicates all-zero block sequences. Each solid arrows in $\mathbf{a}$ and $\mathbf{b}$ shows a distinct row of $\mathbf{A}$ and a column of $\mathbf{B}$, respectively.

of index values $\kappa' \neq \kappa$. Finally, the *correct decoding* requirement is defined as in (5), that is

$$H(\mathbf{AB}^{(\kappa)}|\{\mathbf{C}_p^{(\kappa)}\}_{p\in\mathcal{P}_R}) = 0. \tag{10}$$

A coding and decoding strategy that satisfies conditions (8), (9), and (10) is said to be *feasible*. For given parameters m and n the performance is measured by the pair (P_R, C_L), with $P_C = 1$, where C_L is the communication load defined in (6).

III. Background: Generalized PolyDot Code without Security Constraint

In this section, we consider the system model shown in Fig. 1 and review the GPD construction first proposed in [15] and later improved in [14], [28] for the special case of no secrecy constrains, i.e., $P_C = 0$. In the process, we propose a novel intuitive interpretation of GPD encoding and decoding based on the distributed computation of samples from convolutions via z-transforms.

We start by recalling that the GPD coding scheme achieves the best currently known trade-off between recovery threshold P_R and communication load C_L for $P_C = 0$, i.e., under no security constraint. The entangled polynomial codes of [28] have the same properties in terms of (P_R, P_C). The GPD codes for $P_C = 0$ also achieve the optimal recovery threshold among all linear coding strategies in the cases of $t = 1$ or $d = 1$, also they minimize the recovery threshold for the minimum communication load $C_{L,\min}$ [9], [28].

The GPD code splits the data matrices $\mathbf{A}$ and $\mathbf{B}$ both horizontally and vertically as

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{1,1} & \dots & \mathbf{A}_{1,s} \\ \vdots & \ddots & \vdots \\ \mathbf{A}_{t,1} & \dots & \mathbf{A}_{t,s} \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} \mathbf{B}_{1,1} & \dots & \mathbf{B}_{1,d} \\ \vdots & \ddots & \vdots \\ \mathbf{B}_{s,1} & \dots & \mathbf{B}_{s,d} \end{bmatrix}. \tag{11}$$

The parameters s, t, and d can be set arbitrarily under the constraints $m = ts$ and $n = sd$. Note that polynomial codes set $s = 1$, while MatDot codes have $t = d = 1$ [13]. All sub-matrices $\mathbf{A}_{i,j}$ and $\mathbf{B}_{k,l}$ have dimensions $T/t \times S/s$ and $S/s \times D/d$, respectively. The GPD code computes each block (i, j) of the product $\mathbf{C} = \mathbf{AB}$, namely $\mathbf{C}_{i,j} = \sum_{k=1}^{s} \mathbf{A}_{i,k}\mathbf{B}_{k,j}$, for $i \in [1, t]$ and $j \in [1, d]$, in a distributed fashion. This is done by means of polynomial encoding and polynomial interpolation. As we review next, the computation of block $\mathbf{C}_{i,j}$ can be interpreted as the evaluation of the middle sample of the convolution $\mathbf{c}_{i,j} = \mathbf{a}_i * \mathbf{b}_j$ between the block sequences $\mathbf{a}_i = [\mathbf{A}_{i,1}, \dots, \mathbf{A}_{i,s}]$ and $\mathbf{b}_j = [\mathbf{B}_{s,j}, \dots, \mathbf{B}_{1,j}]$. In fact, the sth sample of the block sequence $\mathbf{c}_{i,j}$ equals $\mathbf{C}_{i,j}$, i.e., $[\mathbf{c}_{i,j}]_s = \mathbf{C}_{i,j}$. The computation is carried out distributively in the frequency domain by using z-transforms with different workers being assigned distinct samples in the frequency domain.

To elaborate, define the block sequence $\mathbf{a}$ obtained by concatenating the block sequences $\mathbf{a}_i$ as $\mathbf{a} = \{\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_t\}$. Pictorially, a sequence $\mathbf{a}$ is obtained from the matrix $\mathbf{A}$ by reading the blocks in the left-to-right top-to-bottom order, as seen in Fig. 3. We also introduce the longer time block sequence $\mathbf{b}$ as

$$\mathbf{b} = \{\mathbf{b}_1, \mathbf{0}, \mathbf{b}_2, \mathbf{0}, \dots, \mathbf{b}_d\}, \tag{12}$$

with $\mathbf{0}$ being a block sequence of $s(t^* - 1)$ all-zero block matrices with dimensions $S/s \times D/d$. The sequence $\mathbf{b}$ can be obtained from the matrix $\mathbf{B}$ by following the bottom-to-top left-to-right order shown in Fig. 3 and by adding the all-zero block sequences between any two columns of the matrix $\mathbf{B}$.

In the frequency domain, the z-transforms of sequences $\mathbf{a}$ and $\mathbf{b}$ are obtained as

$$\mathbf{F_a}(z) = \sum_{r=0}^{ts-1} [\mathbf{a}]_{r+1} z^r = \sum_{i=1}^{t}\sum_{j=1}^{s} \mathbf{A}_{i,j} z^{s(i-1)+j-1}, \tag{13}$$

$$\mathbf{F_b}(z) = \sum_{r=0}^{s-1+ts(d-1)} [\mathbf{b}]_{r+1} z^r = \sum_{k=1}^{s}\sum_{l=1}^{d} \mathbf{B}_{k,l} z^{s-k+ts(l-1)}, \tag{14}$$

respectively. The master server evaluates the polynomials $\mathbf{F_a}(z)$ and $\mathbf{F_b}(z)$ in P non-zero distinct points $z_1, \dots, z_P \in \mathbb{F}$ and sends the corresponding linearly encoded matrices $\mathbf{A}_p = \mathbf{F_a}(z_p)$ and $\mathbf{B}_p = \mathbf{F_b}(z_p)$ to server p. The encoding functions are hence given by the polynomial evaluations (13) and (14), for $z_1, \dots, z_p$. Server p computes the multiplication $\mathbf{F_a}(z_p)\mathbf{F_b}(z_p)$ and sends it to the master server. The master

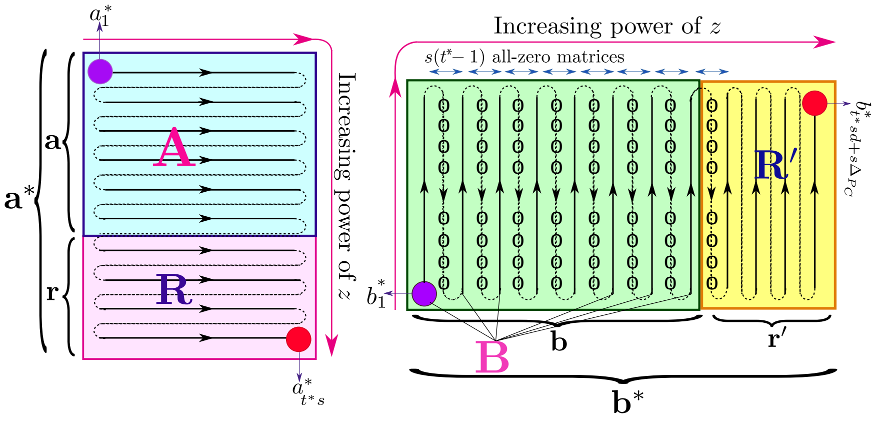

Fig. 4: Construction of the time block sequences $\mathbf{a}^* = [\mathbf{a}, \mathbf{r}]$ and $\mathbf{b}^* = [\mathbf{b}, \mathbf{r}']$ in (20) and (21) used to define the SGPD code for the case $s < t$. The zero dashed lines in $\mathbf{b}$ and $\mathbf{r}'$ indicate all-zero block sequences.

server computes the inverse z-transform for the received products $\{\mathbf{A}_p\mathbf{B}_p\}_{p\in\mathcal{P}_R} = \{\mathbf{F}_\mathbf{a}(z_p)\mathbf{F}_\mathbf{b}(z_p)\}_{p\in\mathcal{P}_R}$, obtaining the convolution $\mathbf{a} * \mathbf{b}$.

From the convolution $\mathbf{a} * \mathbf{b}$ we can see that the master server is able to compute all the desired blocks $\mathbf{C}_{i,j}$ by reading the middle samples of the convolutions $\mathbf{c}_{i,j} = \mathbf{a}_i * \mathbf{b}_j$ from samples of the sequence $\mathbf{c} = \mathbf{a} * \mathbf{b}$ in the order $[\mathbf{c}]_{s-1} = \mathbf{C}_{1,1}, [\mathbf{c}]_{2s-1} = \mathbf{C}_{2,1}, \ldots, [\mathbf{c}]_{ts-1} = \mathbf{C}_{t,1}, [\mathbf{c}]_{s-1+t^*s} = \mathbf{C}_{1,2}, \ldots, [\mathbf{c}]_{ts-1+t^*s} = \mathbf{C}_{t,2}, \ldots$. Note that, in particular, the zero block subsequences added to sequence $\mathbf{b}$ ensure that no interference from the other convolutions, $\mathbf{c}_{i',j'}$ affects the middle (sth) sample of a convolution $\mathbf{c}_{i,j}$ with $i' \neq i$ and $j' \neq j$.

To carry out the inverse transform, the master server needs to collect as many values $\mathbf{F}_\mathbf{a}(z_p)\mathbf{F}_\mathbf{b}(z_p)$ as there are samples of the sequence $\mathbf{a} * \mathbf{b}$, yielding the recovery threshold

$$P_R = tsd + s - 1. \tag{15}$$

Equivalently, in terms of the underlying polynomial interpretation, the master server needs to collect a number of evaluations of the polynomial $\mathbf{F}_\mathbf{a}(z)\mathbf{F}_\mathbf{b}(z)$ equal to the degree of $\mathbf{F}_a(z)\mathbf{F}_b(z)$ plus one. This computation is of complexity order $\mathcal{O}(TDP_R(\log(P_R))^2)$ [13]. Furthermore, the communication load is given as

$$C_L = P_R\frac{TD}{td}, \tag{16}$$

where $TD/(td)$ is the size of each matrix $\mathbf{F}_\mathbf{a}(z)\mathbf{F}_\mathbf{b}(z)$.

IV. Secure PolyDot Code

In this section, we propose a novel extension of the GPD code that is able to ensure the secrecy constraint for any $P_C < P$. We also derive the corresponding achievable set of triples (P_C, P_R, C_L). As we will discuss, the projection of this set onto the plane defined by the condition $P_C = 0$ includes the set of pairs (P_R, C_L) in (15) and (16) obtained by the GPD code [14]. The proposed secure GPD (SGPD) code augments matrices $\mathbf{A}$ and $\mathbf{B}$ by adding P_C random block matrices to the input matrices $\mathbf{A}$ and $\mathbf{B}$, in a manner similar to prior works [18]–[21], [23], yielding augmented matrices $\mathbf{A}^*$ and $\mathbf{B}^*$. As we will see, a direct application of the GPD codes to these matrices is suboptimal.

In contrast, we propose a novel way to construct sequences $\mathbf{a}^*$ and $\mathbf{b}^*$ from matrices $\mathbf{A}^*$ and $\mathbf{B}^*$ that enables the definition of a more efficient code by means of the z-transform approach discussed in the previous section. To this end, we follow the design criterion of decreasing the recovery threshold P_R for a given communication load C_L. Based on the discussion in the previous section, this goal can be realized by decreasing the length of the sequence $\mathbf{c}^* = \mathbf{a}^* * \mathbf{b}^*$, which can in turn be ensured by reducing the length of the sequence $\mathbf{b}^*$ for a given length of the sequence $\mathbf{a}^*$. We accomplish this objective by *(i)* adaptively appending rows *or* columns with random elements to matrix $\mathbf{A}$, and, correspondingly columns *or* rows to $\mathbf{B}$, which can reduce the recovery threshold; and *(ii)* modifying the zero padding procedure (see Fig. 3) for the construction of sequence $\mathbf{b}^*$. In order to account for point *(i)*, we consider separately the two cases $s < t$ and $s \geq t$.

A. Secure Generalized PolyDot Code: The $s < t$ Case

As illustrated in Fig. 4, when $s < t$, we augment the input matrices $\mathbf{A}$ and $\mathbf{B}$ by adding

$$\Delta_{P_C} \triangleq \left\lceil \frac{P_C}{s} \right\rceil, \tag{17}$$

random row and column blocks to matrices $\mathbf{A}$ and $\mathbf{B}$, respectively. Accordingly, the $t^* \times s$ augmented block matrix $\mathbf{A}^*$ with $t^* = t + \Delta_{P_C}$ is obtained as

$$\mathbf{A}^* = \begin{bmatrix} \mathbf{A} \\ \mathbf{R} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{1,1} & \ldots & \mathbf{A}_{1,s} \\ \vdots & \ddots & \vdots \\ \mathbf{A}_{t,1} & \ldots & \mathbf{A}_{t,s} \\ \mathbf{R}_{1,1} & \ldots & \mathbf{R}_{1,s} \\ \vdots & \ddots & \vdots \\ \mathbf{R}_{\Delta_{P_C},1} & \ldots & \mathbf{R}_{\Delta_{P_C},s} \end{bmatrix}, \tag{18}$$

while the $s \times d^*$ augmented matrix $\mathbf{B}^* = [\mathbf{B}\ \mathbf{R}']$ with $d^* = d + \Delta_{P_C}$ is obtained as

$$\mathbf{B}^* = \begin{bmatrix} \mathbf{B}_{1,1} & \ldots & \mathbf{B}_{1,d} & \mathbf{R}'_{s,1} & \ldots & \mathbf{R}'_{s,\Delta_{P_C}} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{B}_{s,1} & \ldots & \mathbf{B}_{s,d} & \mathbf{R}'_{1,1} & \ldots & \mathbf{R}'_{1,\Delta_{P_C}} \end{bmatrix}. \tag{19}$$

In (18) and (19), if s divides P_C, all block matrices $\mathbf{R}_{i,j} \in \mathbb{F}^{\frac{T}{t} \times \frac{S}{s}}$ and $\mathbf{R}'_{i,j} \in \mathbb{F}^{\frac{S}{s} \times \frac{D}{d}}$ are generated with i.i.d. uniform random elements in $\mathbb{F}$. Otherwise, if $\Delta_{P_C} - P_C/s > 0$, the last $s\Delta_{P_C} - P_C$ matrices in (18), with right-to-left ordering in the last row of $\mathbf{R}_{i,j}$, and in (19) with top-to-bottom ordering in the last column of $\mathbf{R}'_{i,j}$, are all-zero block matrices.

As illustrated in Fig. 4, in the SGPD scheme, the block sequence $\mathbf{a}^*$ is defined in the same way as in the conventional GPD, yielding

$$\mathbf{a}^* = \{\mathbf{a}_1, \ldots, \mathbf{a}_t, \mathbf{r}_1, \ldots, \mathbf{r}_{\Delta_{P_C}}\}, \tag{20}$$

where $\mathbf{r}_i$ is the ith row of the block matrix $\mathbf{R}$, $i \in [1, \Delta_{P_C}]$. We also define the time block sequence $\mathbf{b}^* = \{\mathbf{b}, \mathbf{r}'\}$ as

$$\mathbf{b}^* = \{\mathbf{b}_1, \mathbf{0}, \mathbf{b}_2, \mathbf{0}, \ldots, \mathbf{b}_d, \mathbf{0}, \mathbf{r}'_1, \mathbf{r}'_2, \ldots, \mathbf{r}'_{\Delta_{P_C}}\}, \tag{21}$$

where $\mathbf{0}$ is block sequences of $s(t^*-1)$ all-zero block matrices, respectively, with dimensions $S/s \times D/d$, while $\mathbf{r}'_j$ is the jth column of the random matrix $\mathbf{R}'$. The key novel idea of this construction is that no zero matrices are introduced between the columns of matrix $\mathbf{R}'$. As shown in Theorem 1 below, this construction allows the master server to recover all the desired submatrices $\mathbf{C}_{i,j}$ for $i \in [1, t]$ and $j \in [1, d]$ from the middle samples of the convolutions $\mathbf{c}_{i,j} = \mathbf{a}_i * \mathbf{b}_j$ (see Fig. 5 for an illustration).

Theorem 1. *For a given security level $P_C < P$, the proposed SGPD code achieves the recovery threshold P_R*

$$\begin{cases} tsd + s - 1, & \textit{if } P_C = 0, \\ t^*s(d+1) + s\Delta_{P_C} - 1, & \textit{if } P_C \geq 1 \textit{ and } \Delta_{P_C} = \frac{P_C}{s}, \\ t^*s(d+1) - s\Delta_{P_C} + 2P_C - 1, & \textit{if } P_C \geq 1 \textit{ and } \Delta_{P_C} > \frac{P_C}{s}, \end{cases} \tag{22}$$

and the communication load (16)*, where $t^* = t + \Delta_{P_C}$ and $d^* = d + \Delta_{P_C}$ for any integer values $t, s,$ and d such that $s < t$, $m = ts$, and $n = sd$.*

Proof. The z-transform of sequences $\mathbf{a}^*$ and $\mathbf{b}^*$ are given respectively as

$$\begin{aligned} \mathbf{F}_{\mathbf{a}^*}(z) = & \underbrace{\sum_{i=1}^{t} \sum_{j=1}^{s} \mathbf{A}^*_{i,j} z^{s(i-1)+(j-1)}}_{\triangleq \mathbf{F}_1(z)} \\ & + \underbrace{\sum_{i=t+1}^{t^*} \sum_{j=1}^{s} \mathbf{A}^*_{i,j} z^{s(i-1)+j-1}}_{\triangleq \mathbf{F}_2(z)}, \end{aligned} \tag{23}$$

$$\begin{aligned} \mathbf{F}_{\mathbf{b}^*}(z) = & \underbrace{\sum_{k=1}^{s} \sum_{l=1}^{d} \mathbf{B}^*_{k,l} z^{s-k+t^*s(l-1)}}_{\triangleq \mathbf{F}_3(z)} \\ & + \underbrace{\sum_{k=1}^{s} \sum_{l=d+1}^{d^*} \mathbf{B}^*_{k,l} z^{t^*sd+s(l-d)-k}}_{\triangleq \mathbf{F}_4(z)}. \end{aligned} \tag{24}$$

The master server evaluates $\mathbf{F}_{\mathbf{a}^*}(z)$ and $\mathbf{F}_{\mathbf{b}^*}(z)$ at P non-zero distinct points $z_1, \ldots, z_P \in \mathbb{F}$, which define the encoding functions, and sends both matrices $\mathbf{A}_p = \mathbf{F}_{\mathbf{a}^*}(z_p)$ and $\mathbf{B}_p = \mathbf{F}_{\mathbf{b}^*}(z_p)$ to worker p. Worker p performs the multiplication $\mathbf{F}_{\mathbf{a}^*}(z_p)\mathbf{F}_{\mathbf{b}^*}(z_p)$, and sends the results back to the master server. To reconstruct all blocks $\mathbf{C}_{i,j}$ of matrix $\mathbf{C} = \mathbf{AB}$, the master server carries out a polynomial interpolation, or equivalently, it computes the inverse z-transform, upon receiving a number of multiplication results equal to at least the length of the sequence $\mathbf{c}^* = \mathbf{a}^* * \mathbf{b}^*$. As we detail next, the (i, l) block $\mathbf{C}_{i,l} = \sum_{r=1}^{s} \mathbf{A}_{i,r}\mathbf{B}_{r,l}$, for all $i \in [1, t]$ and $l \in [1, d]$, of matrix $\mathbf{C} = \mathbf{AB}$ can be seen equal to the $(si-1+(l-1)t^*s)$th sample of the convolution $\mathbf{c}^* = \mathbf{a}^* * \mathbf{b}^*$. An illustration can be found in Fig. 5.

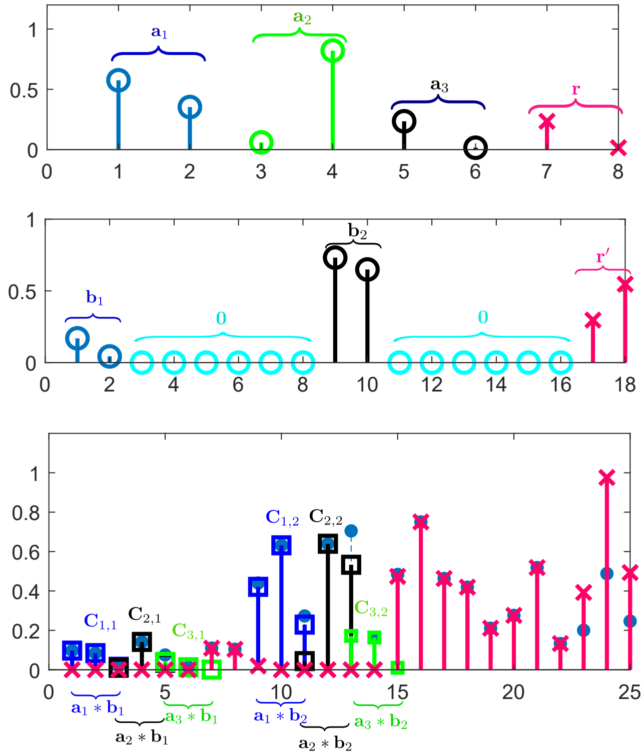

Fig. 5: Outcome of the communication $\mathbf{C}_{i,j} = \mathbf{a}_i * \mathbf{b}_j$ for $t = 3, s = 2, d = 2$, and $P_C = 2$. Dashed blue stems with filled markers represent the convolution $\mathbf{c}^*$. Individual convolutions $\mathbf{c}_{i,j}$ are shown in different colors with square markers. Contributions from one or both random matrices are shown as red crosses. The desired submatrices $\mathbf{C}_{i,j}$ are seen to equal the corresponding samples from the sequence $\mathbf{c}^*$, associated with the center points of the individual convolutions.

To see this, we first note that, by the properties of GPD codes, matrix $\mathbf{C}_{i,l}$ is the coefficient of the monomial $z^{si-1+(l-1)t^*s}$ in $\mathbf{F}_1(z)\mathbf{F}_3(z)$. Note that this holds since the polynomial $\mathbf{F}_1(z)$ and $\mathbf{F}_3(z)$ are defined as GPD codes. We now need to show that no other contribution to this term arises from the products $\mathbf{F}_1(z)\mathbf{F}_4(z)$, $\mathbf{F}_2(z)\mathbf{F}_3(z)$, and $\mathbf{F}_2(z)\mathbf{F}_4(z)$. The terms in the product $\mathbf{F}_1(z)\mathbf{F}_4(z)$ have exponents $(t^*sd + s(i-1) + s(l-d) - 1)$, for $i \in [1, t]$ and $l \in [d+1, d^*]$, which do not include the desired values $(si-1+(l-1)t^*s)$ for $i \in [1, t]$ and $l \in [1, d]$. A similar discussion applies to the product $\mathbf{F}_2(z)\mathbf{F}_3(z)$, whose exponents are $(s(i + t^*l - t^*) - 1)$, for $i \in [t+1, t^*]$ and $l \in [1, d]$, and $\mathbf{F}_2(z)\mathbf{F}_4(z)$, whose exponents are $(t^*sd + s(i-1) + s(l-d) - 1)$, for $i \in [t+1, t^*]$ and $l \in [d+1, d^*]$.

In order to recover the convolution $\mathbf{c}^*$, the master server

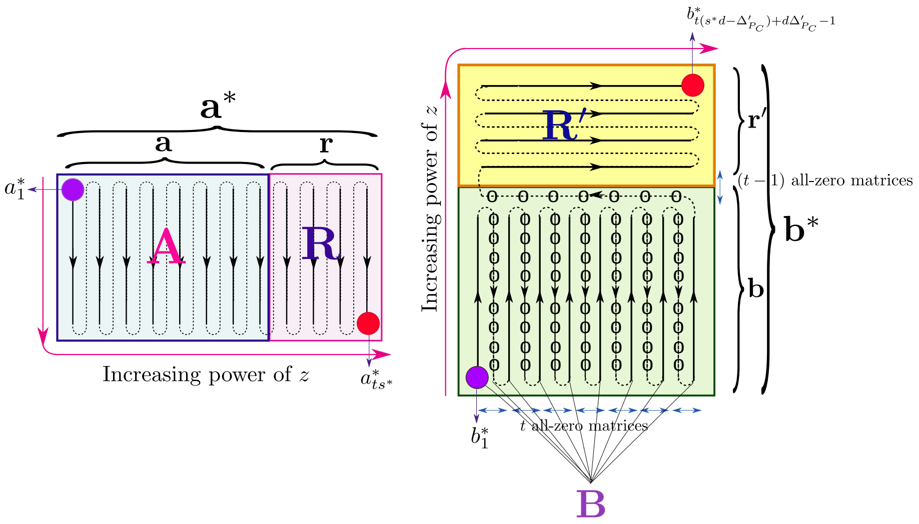

Fig. 6: Construction of the time block sequences $\mathbf{a}^*$ and $\mathbf{b}^*$ in (31) and (32) used to define the secure generalized PolyDot (SGPD) code for the case $s \geq t$. The solid line and the zero dashed lines in $\mathbf{b}^*$ indicate columns of $\mathbf{B}$ and all-zero block sequences, respectively.

needs to collect a number of values of the product $\mathbf{F_a}(z)\mathbf{F_b}(z)$ equal to the length of the sequence $\mathbf{c}^*$, which can be computed as the degree $\deg\left(\mathbf{F_a}(z)\mathbf{F_b}(z)\right)+1$, where $\deg(\mathbf{F_a}(z)\mathbf{F_b}(z))$ is

$$\begin{cases} t^*s(d+1)+s\Delta_{P_C}-1, & \text{if } \Delta_{P_C}=\frac{P_C}{s}, \\ dst^*-s\Delta_{P_C}+2P_C+t-2, & \text{if } \Delta_{P_C}>\frac{P_C}{s}. \end{cases} \tag{25}$$

For $P_C \geq 1$ this implies the recovery threshold P_R in (22). The communication load C_L in (16) follows from the fact that there are $TD/(td)$ entries in $\mathbf{F}_{\mathbf{a}^*}(z_p)\mathbf{F}_{\mathbf{b}^*}(z_p)$, for all $p \in [1, P_R]$.

The security constraint (3) can be proved in a manner similar to [20] by the following steps:

$$\begin{aligned}
&I(\mathbf{A},\mathbf{B};\mathbf{A}_\mathcal{P},\mathbf{B}_\mathcal{P})\\
=&H(\mathbf{A}_\mathcal{P},\mathbf{B}_\mathcal{P})-H(\mathbf{A}_\mathcal{P},\mathbf{B}_\mathcal{P}|\mathbf{A},\mathbf{B})\\
\overset{(a)}{=}&H(\mathbf{A}_\mathcal{P},\mathbf{B}_\mathcal{P})-H(\mathbf{A}_\mathcal{P},\mathbf{B}_\mathcal{P}|\mathbf{A},\mathbf{B})\\
&+H(\mathbf{A}_\mathcal{P},\mathbf{B}_\mathcal{P}|\mathbf{A},\mathbf{B},\mathbf{R}_1,\dots,\mathbf{R}_{P_C},\mathbf{R}'_1,\dots,\mathbf{R}'_{P_C})\\
=&H(\mathbf{A}_\mathcal{P},\mathbf{B}_\mathcal{P})-I(\mathbf{A}_\mathcal{P},\mathbf{B}_\mathcal{P};\mathbf{R}_1,\dots,\mathbf{R}_{P_C},\mathbf{R}'_1,\dots,\mathbf{R}'_{P_C}|\mathbf{A},\mathbf{B})\\
=&H(\mathbf{A}_\mathcal{P},\mathbf{B}_\mathcal{P})-H(\mathbf{R}_1,\dots,\mathbf{R}_{P_C},\mathbf{R}'_1,\dots,\mathbf{R}'_{P_C}|\mathbf{A},\mathbf{B})\\
&+H(\mathbf{R}_1,\dots,\mathbf{R}_{P_C},\mathbf{R}'_1,\dots,\mathbf{R}'_{P_C}|\mathbf{A},\mathbf{B},\mathbf{A}_\mathcal{P},\mathbf{B}_\mathcal{P})\\
\overset{(b)}{=}&H(\mathbf{A}_\mathcal{P},\mathbf{B}_\mathcal{P})-H(\mathbf{R}_1,\dots,\mathbf{R}_{P_C},\mathbf{R}'_1,\dots,\mathbf{R}'_{P_C})\\
\overset{(c)}{\leq}&H(\mathbf{A}_\mathcal{P})+H(\mathbf{B}_\mathcal{P})-\sum_{p=1}^{P_C}H(\mathbf{R}_p)-\sum_{p=1}^{P_C}H(\mathbf{R}'_p)\\
\overset{(d)}{=}&H(\mathbf{A}_\mathcal{P})+H(\mathbf{B}_\mathcal{P})-P_C\frac{TS}{m}\log|\mathbb{F}|-P_C\frac{SD}{n}\log|\mathbb{F}|\\
\overset{(e)}{\leq}&\sum_{p=1}^{P_C}H(\mathbf{A}_p)+\sum_{p=1}^{P_C}H(\mathbf{B}_p)-P_C\frac{TS}{m}\log|\mathbb{F}|-P_C\frac{SD}{n}\log|\mathbb{F}|\\
\overset{(f)}{=}&P_C\frac{TS}{m}\log|\mathbb{F}|+P_C\frac{SD}{n}\log|\mathbb{F}|-P_C\frac{TS}{m}\log|\mathbb{F}|\\
&-P_C\frac{SD}{n}\log|\mathbb{F}|\\
=&0,
\end{aligned} \tag{26}$$

where (a) follows from the definition of encoding functions, since $\mathbf{A}_\mathcal{P}$ is a deterministic function of $\mathbf{A}$ and $\mathbf{R}_p$, and $\mathbf{B}_\mathcal{P}$ is a deterministic function of $\mathbf{B}$ and $\mathbf{R}'_p$, respectively, for all $p \in [1, P_C]$; (b) follows from (23) and (24), since from P_R polynomial evaluations $\mathbf{A}_\mathcal{P}$ and $\mathbf{B}_\mathcal{P}$ in (23) and (24) we can recover $2P_C$ unknowns when the coefficients $\mathbf{A}_{i,j}$ and $\mathbf{B}_{k,l}$ are known, given that we have $P_R \geq 2P_C$; (c) and (d) follows since $\mathbf{R}_p$ and $\mathbf{R}'_p$ are independent uniformly distributed entries; (e) follows by upper bounding the joint entropy using the sum of individual entropies; and (f) follows from an argument similar to (d). Hence, the proposed scheme is information-theoretically secure. □

Remark 1. *When $P_C \geq 1$ a direct application of the GPD construction in Fig. 3 would yield the larger recovery threshold*

$$P_R=\begin{cases} t^*sd^*+s-1, & \textit{if } \Delta_{P_C}=\frac{P_C}{s}, \\ dst^*+s-1-2(s\Delta_{P_C}-P_C), & \textit{if } \Delta_{P_C}>\frac{P_C}{s}. \end{cases} \tag{27}$$

B. Secure Generalized PolyDot Code: The $s \geq t$ Case

As illustrated in Fig. 6, when $s \geq t$, we instead augment input matrices $\mathbf{A}$ and $\mathbf{B}$ by adding

$$\Delta'_{P_C} \triangleq \left\lceil \frac{P_C}{\min\{t,d\}} \right\rceil \tag{28}$$

column and row blocks to matrices $\mathbf{A}$ and $\mathbf{B}$. This can be seen to yield a smaller recovery threshold. Accordingly, the $t \times s^*$ augmented block matrix $\mathbf{A}^* = [\mathbf{A}\ \mathbf{R}]$ with $s^* = s + \Delta'_{P_C}$ is obtained as

$$\mathbf{A}^*=\begin{bmatrix} \mathbf{A}_{1,1} & \dots & \mathbf{A}_{1,s} & \mathbf{R}_{1,1} & \dots & \mathbf{R}_{1,\Delta'_{P_C}} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{t,1} & \dots & \mathbf{A}_{t,s} & \mathbf{R}_{t,1} & \dots & \mathbf{R}_{t,\Delta'_{P_C}} \end{bmatrix}, \tag{29}$$

while the $s^* \times d$ augmented block matrix $\mathbf{B}^*$ is defined as

$$\mathbf{B}^* = \left[\begin{array}{c} \mathbf{R}' \\ \mathbf{B} \end{array}\right] = \left[\begin{array}{ccc} \mathbf{R}'_{\Delta'_{P_C},1} & \cdots & \mathbf{R}'_{\Delta'_{P_C},d} \\ \vdots & \ddots & \vdots \\ \mathbf{R}'_{1,1} & \cdots & \mathbf{R}'_{1,d} \\ \mathbf{B}_{1,1} & \cdots & \mathbf{B}_{1,d} \\ \vdots & \ddots & \vdots \\ \mathbf{B}_{s,1} & \cdots & \mathbf{B}_{s,d} \end{array}\right]. \tag{30}$$

As for (29) and (30), if $\Delta'_{P_C} - P_C/\min\{t, d\} > 0$, the last $s\Delta'_{P_C} - P_C$ block matrices in (29), with bottom-to-top right-to-left ordering in $\mathbf{R}$, and in (30) with right-to-left top-to-bottom ordering in $\mathbf{R}'$, are all-zero block matrices. The construction of sequences $\mathbf{a}^*$ and $\mathbf{b}^*$ is analogous to the GPD in the non-secure case. In particular, as seen in Fig. 6, the time block sequence $\mathbf{a}^*$ is

$$\mathbf{a}^* = \{\mathbf{a}_1, \mathbf{r}_1, \mathbf{a}_2, \mathbf{r}_2, \ldots, \mathbf{a}_t, \mathbf{r}_t\}, \tag{31}$$

whereas the block sequence $\mathbf{b}^*$ is defined as

$$\mathbf{b}^* = \{\mathbf{b}_1, \mathbf{0}, \mathbf{b}_2, \ldots, \mathbf{0}, \mathbf{b}_d, \hat{\mathbf{0}}, \mathbf{r}'_{\Delta'_{P_C}}, \ldots, \mathbf{r}'_1\}. \tag{32}$$

Here, $\mathbf{0}$ and $\hat{\mathbf{0}}$ are a block sequence of t and $t-1$ all-zero block matrices with dimensions $S/s \times D/d$, respectively, while $\mathbf{r}'_i$ is the ith row of the random matrix $\mathbf{R}'$.

Theorem 2. *For a given security level $P_C < P$, the proposed SGPD code achieves the recovery threshold*

$$P_R = t(s^*d - \Delta'_{P_C}) + ts + 2P_C - 1 \tag{33}$$

and the communication load (16)*, where $s^* = s + \Delta'_{P_C}$ for any integer values $t, s,$ and d such that $s \geq t$, $m = ts$, and $n = sd$.*

Proof. We define the z-transform of sequences $\mathbf{a}^*$ and $\mathbf{b}^*$ respectively as

$$\mathbf{F}_{\mathbf{a}^*}(z) = \sum_{i=1}^{t}\sum_{j=1}^{s} \mathbf{A}^*_{i,j} z^{i-1+t(j-1)} + \sum_{i=1}^{t}\sum_{j=s+1}^{s^*} \mathbf{A}^*_{i,j} z^{i-1+t(j-1)}, \tag{34}$$

$$\mathbf{F}_{\mathbf{b}^*}(z) = \sum_{k=1+\Delta'_{P_C}}^{s^*}\sum_{l=1}^{d} \mathbf{B}^*_{k,l} z^{(s^*-k)t+ts^*(l-1)} + \sum_{k=1}^{\Delta'_{P_C}}\sum_{l=1}^{d} \mathbf{B}^*_{k,l} z^{t(s^*d-\Delta'_{P_C})+d(\Delta'_{P_C}-k)+l-1}. \tag{35}$$

The (i, l) block $\mathbf{C}_{i,l} = \sum_{r=1}^{s} \mathbf{A}_{i,r}\mathbf{B}_{r,l}$, for all $i \in [1, t]$ and $l \in [1, d]$, of matrix $\mathbf{C} = \mathbf{AB}$ can be seen equal to the $(i - 1 + t(s^*l - 1))$th sample of the convolution $\mathbf{c}^* = \mathbf{a}^* * \mathbf{b}^*$. The rest of the proof follows in a manner akin to Theorem 1. □

Remark 2. *The computational complexity of SGPD codes for both workers and master server can be summarized as follows. Each worker is assigned to compute the multiplication $\mathbf{C}_p = \mathbf{A}_p\mathbf{B}_p$, requiring $TSD/(tsd)$ multiplications. For the master server, encoding matrices $\mathbf{A}_p$ and $\mathbf{B}_p$ at each worker amounts to evaluating z-transforms $\mathbf{F}_{\mathbf{a}^*}(z)$ and $\mathbf{F}_{\mathbf{b}^*}(z)$ at a random point z_p. This requires multiplying z_p by $(ts + P_C)$ and $(sd + P_C)$ submatrices, each of dimension $T/t \times S/s$ and $S/s \times D/d$, respectively. This requires $P_C(TS/(ts) + SD/(sd)) + TS + SD$ multiplications. Overall, the master server needs to carry out $PP_C(TS/(ts) + SD/(sd)) + P(TS + SD)$ multiplications. For decoding, the master server interpolates a polynomial degree $P_R - 1$ for each element in $\mathbf{C}$. Using a polynomial interpolation algorithm, the decoding complexity amounts to $(P_R - 1)(\log(P_R - 1))^2 TD/(td)$ multiplications [42].*

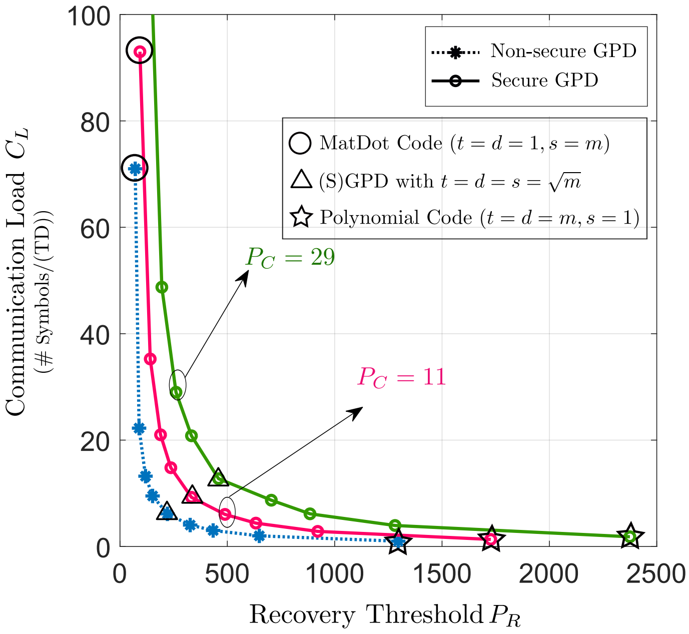

Fig. 7: Communication load C_L versus recovery threshold P_R for both non-secure generalized PolyDot (GPD) and secure generalized PolyDot (SGPD) codes ($m = n = 36$ and $P = 3000$ workers).

Example 1. *We now provide some numerical results of the proposed SGPD scheme. We set $P = 3000$ workers and parameters $m = n = 36$. The trade-off between communication load C_L and recovery threshold P_R for both non-secure conventional GPD codes ($P_C = 0$) and proposed SGPD code with colluding workers $P_C = 11$ and $P_C = 29$ is illustrated in Fig. 7. The figure quantifies the loss in terms of achievable pairs (P_R, C_L) that is caused by the security constraint.*

C. Trading Off Computation and Communication Latencies

In this subsection, we elaborate on the importance of enabling a flexible trade-off between communication load and recovery threshold by analyzing the overall completion time for the matrix multiplication task at hand. The completion delay is the sum of latencies due to computation and communication.

To this end, following a well-established model [43], [11], we assume that computation at each worker p requires a random time T_p^{comp}, measured in some specified unit of time, that is modeled as a shifted exponential distribution with cumulative distribution function (cdf)

$$F^{\text{comp}}(T^{\text{comp}}) = 1 - \exp\left(-\frac{\mu TSD}{tsd}(T^{\text{comp}} - T^{\text{comp}}_{\min})\right), \tag{36}$$

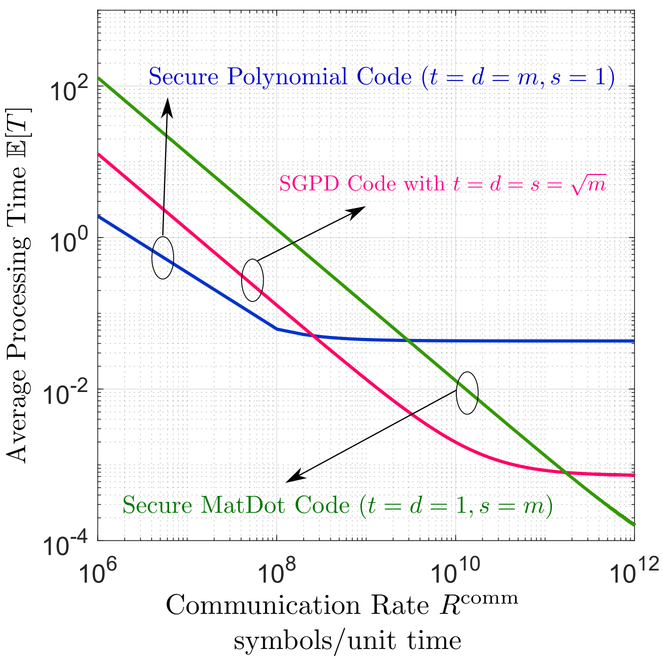

Fig. 8: Average completion time $\mathbb{E}[T]$ versus communication rate R^{comm} for secure generalized PolyDot (SGPD) codes with $P = 3000$, $P_C = 29$, $T = S = D = 1008$, $\mu = 0.5 \times 10^{-4}$, and $T^{\text{comp}} = 1$, and $m = n = 36$: *(i)* $t = d = 36$, $s = 1$ (SGPD code), *(ii)* $t = s = d = 6$, and *(iii)* $t = d = 1$, $s = 36$ (secure MatDot code).

for $T \geq T^{\text{comp}}_{\min}$ and $F^{\text{comp}}(T) = 0$ otherwise. According to (36), the parameter $T^{\text{comp}}_{\min}$ represents the minimum processing time, and $1/\mu$ represents the average excess computing time, with respect to $T^{\text{comp}}_{\min}$, per multiplication (recall Remark 2). Assuming independent computing times, for a given recovery threshold P_R, the computation time T^{comp} is hence given as the P_Rth-order statistic, i.e., the P_Rth smallest variable, among the i.i.d. variables $(T^{\text{comp}}_1, \ldots, T^{\text{comp}}_P)$. Its expectation is given by [44]

$$\mathbb{E}[T^{\text{comp}}] = \frac{tsd}{\mu TSD} \sum_{i=1}^{P_R} \frac{1}{P - P_R + i} = \frac{tsd}{\mu TSD}(H_P - H_{P-P_R}), \tag{37}$$

where H_P is the generalized harmonic number defined as $H_P = \sum_{i=1}^{P} 1/i$.

Suppose now that the workers communicate with the master server are a link with an overall download rate R^{comm} (symbols per unit time). The communication latency is hence given as

$$T^{\text{comm}} = P_R \frac{TD}{tdR^{\text{comm}}}, \tag{38}$$

since the workers need to return $P_R TD/(td)$ symbols to the master server. Overall, the average completion time is given as

$$\mathbb{E}[T] = T^{\text{comp}}_{\min} + \frac{tsd}{\mu TSD}(H_P - H_{P-P_R}) + P_R \frac{TD}{tdR^{\text{comm}}}. \tag{39}$$

Example 2. *Let consider $P = 3000$ workers and parameters $m = n = 36$. We assume that $P_C = 29$, $T = S = D = 1008$, $\mu = 0.5 \times 10^{-4}$, and $T^{comm}_{\min} = 1$. We compare the performance of the following SGPD codes:* (i) *$t = d = 36$ and $s = 1$ (secure Polynomial code);* (ii) *$t = s = d = 6$;* (iii) *$t = d = 1$ and $s = 36$ (secure MatDot code). The values of C_L and P_R for these codes are shown in Fig. 7. The average completion time* (39) *is plotted versus the communication rate R^{comm} in Fig. 8. The figure shows that the optimal choice of the latency-minimizing SGPD code along the curve in Fig. 7 depends on the system's operating point: For small communication rates, it is preferable to reduce the communication load C_L, and hence secure Polynomial codes are the best choice; while for large communication rate, it is optimal to choose codes with an increasingly large value of the communication load C_L.*

V. Secure and Private Generalized PolyDot Code

In this section, we study the setup shown in Fig. 2. We propose a variant of the private and secure GPD code introduced in [38] that we refer to as private and secure GPD (PSGPD) code. Note that in [38] a private coded matrix multiplication scheme is proposed only for Polynomial codes with $s = 1$ in (11). We derive the corresponding achievable set of pairs (P_R, C_L) as defined in Section II under the condition $P_C = 1$, i.e., the workers do not collude.

Theorem 3. *For a given security level $P_C = 1$, there is an achievable PSGPD codes with the recovery threshold*

$$P_R = \begin{cases} s(t+1)d, & \textit{if } s < t, \\ ts(d+1) - t + 1, & \textit{if } s \geq t, \end{cases} \tag{40}$$

and the communication load (16)*, for any integer values t, s, and d such that $m = ts$, and $n = sd$.*

Proof. The proof is presented in Appendix A. □

Remark 3. *The computational complexity of PSGPD codes for both workers and master server is summarized as follows. In PSGPD codes, each worker has two duties, namely encoding the library $\mathcal{B}$ and computing the multiplication $\mathbf{C}_p^{(\kappa)} = \mathbf{A}_p^{(\kappa)}\mathbf{B}_p^{(\kappa)}$. Encoding the library, i.e., computing the matrix $\mathbf{B}_p^{(\kappa)}$ in* (44)*, requires to evaluate $\mathbf{F}_{\mathbf{B}^{(r)}}(z)$, $r \in [1, L]$ at query vector $\mathbf{q}_p^{(\kappa)}$. Hence, the former task requires LSD multiplications, while the latter entails $TSD/(tsd)$ multiplications. In total, each worker carries out $LSD + TSD/(tsd)$ multiplications. The master server encodes matrix $\mathbf{A}_p^{(\kappa)}$ with $(1 + ts)TS/(ts)$ multiplications. In total, for all P workers, the master server needs $P(1 + ts)TS/(ts)$ multiplications. The computation complexity of the decoding complexity of the master server is the same as for SGPD codes, namely $\mathcal{O}((P_R - 1)(\log(P_r - 1))^2 TD/(td)))$.*

Example 3. *Let us consider $P = 3000$ workers and parameters $m = n = 36$. We assume that $P_C = 1$ in order to compare the performance of proposed SGPD and PSGPD codes. Note that both recovery threshold and communication load of the PSGPD code do not depend on the number of public matrices $|\mathcal{B}| = L$ in the library. The trade-off between communication load C_L and recovery threshold P_R is illustrated in Fig. 9 for both codes. The figure shows that, for a fixed value of P_R, the resulting achievable value of the communication load C_L is smaller for PSGPD than for SGPD codes. This suggests that the privacy requirement on the index κ imposed by PSGPD is less demanding than the security constraint on matrix* $\mathbf{B}$ *under which SGPD codes operate.*

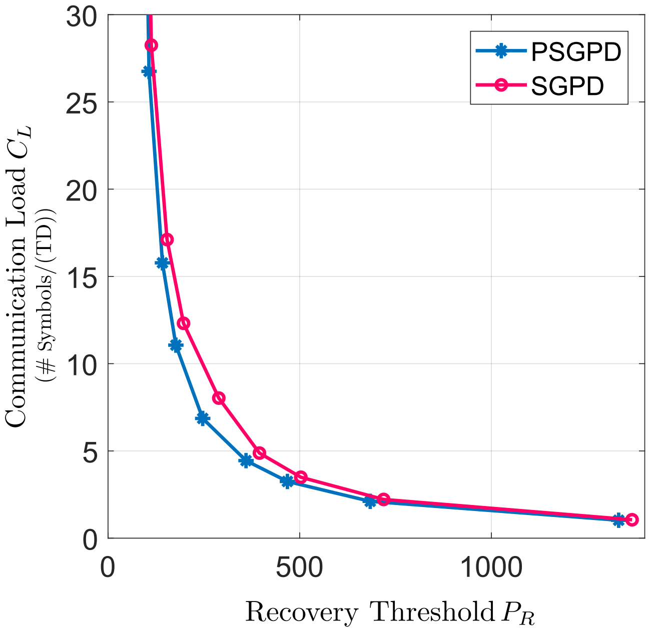

Fig. 9: Communication load C_L versus recovery threshold P_R for secure generalized PolyDot (SGPD) codes with $P_C = 1$ and private and secure generalized PolyDot (PSGPD) codes ($m = n = 36$ and $P = 3000$ workers).

Remark 4. *As for SGPD codes, the overall average completion time of PSGPD codes can be derived following the same steps as described in Section IV-C.*

VI. Concluding Remarks

In this work, we have considered the problem of secure and private distributed matrix multiplication on $\mathbf{C} = \mathbf{AB}$ in terms of design of computational codes for two settings. In the first setting, the two matrices $\mathbf{A}$ and $\mathbf{B}$ contain confidential data and must be kept secure from the workers; and in the second setting , matrix $\mathbf{A}$ is confidential, while matrix $\mathbf{B}$ is selected in a private manner from a library of public matrices. For both problems, this work presents the best currently known trade-off between communication load and recovery threshold. This is done by presenting two code constructions that generalize the state-of-the-art GPD codes [13]–[15], in combination with PIR based codes [38].

Among important items for future research, we mention the extension of the proposed PSGPD construction to $P_C > 1$. Here, we note that one can design an achievable PSGPD scheme for any arbitrary privacy level by trivially concatenating a robust PIR scheme for arbitrary colluding workers and private databases [33] and the proposed SGPD code. However, this approach would require multiplying the data matrix $\mathbf{A}$ with all L public matrices in the set $\mathcal{B} = \{\mathbf{B}^{(r)}\}_{r=1}^{L}$ for each worker $p \in [1, P]$, implying a significantly increased computation load. Future work will focus on PSGPD schemes for any number of colluding workers that provides a smaller computational complexity at the workers. Finally, the establishment of a converse bound and the consideration of non-perfect communication channels between workers and master server [45] are open problems.

Appendix A
Proof of Theorem 3

We start by discussing the $s < t$ case, as done in Section IV. The polynomial encoding function for the input matrix $\mathbf{A}$, is obtained is defined as in (23) for $P_C = 1$, that is

$$\mathbf{F}_{\mathbf{A}}(z) = \sum_{i=1}^{t}\sum_{j=1}^{s} \mathbf{A}_{i,j} z^{s(i-1)+(j-1)} + \mathbf{R} z^{st}, \tag{41}$$

where we recall that $\mathbf{R}$ is an $T/t \times S/s$ random matrix with i.i.d. uniform random elements in $\mathbb{F}$. The encoded matrices are given as $\mathbf{A}_p^{(\kappa)} = \mathbf{F}_{\mathbf{A}}(z_{\kappa,p})$ for values $z_{\kappa,p}$ to be discussed below. For the desired index κ, the master server also computes the query vector $\mathbf{q}_p^{(\kappa)}$ for all $p \in [1, P]$. This is obtained as

$$\mathbf{q}_p^{(\kappa)} = [z_1, \dots, z_{\kappa-1}, z_{\kappa,p}, z_{\kappa+1}, \dots, z_L], \tag{42}$$

where all points $\{z_i\}_{i \neq \kappa}$ are selected uniformly i.i.d. from $\mathbb{F}$ but are identical for all p. The points $\{z_{\kappa,p}\}_{p=1}^{P}$ are selected i.i.d. as distinct elements from $\mathbb{F}$ (recall that we have $|\mathbb{F}| > P$). We note that, as in the PIR scheme [38], the query vector (42) does not leak any information on index κ in the sense defined by condition (9). The master server evaluates $\mathbf{F}_{\mathbf{A}}(z)$ in (41) at the distinct random point $z_{\kappa,p}$, to produce the encoded matrices $\mathbf{A}_p^{(\kappa)} = \mathbf{F}_{\mathbf{A}}(z_{\kappa,p})$, and then sends $\mathbf{A}_p^{(\kappa)}$ along with the query vector $\mathbf{q}_p^{(\kappa)}$ to worker $p \in [1, P]$.

Each worker p, after receiving the query vectors $\mathbf{q}_p^{(\kappa)}$, encodes the library $\mathcal{B}$ into a matrix $\mathbf{B}_p^{(\kappa)}$ as follows. Define the polynomial encoding function for each matrix $\mathbf{B}^{(r)}$, $r \in [1, L]$, in the library $\mathcal{B}$ as in (24) for $P_C = 0$, i.e.,

$$\mathbf{F}_{\mathbf{B}^{(r)}}(z) = \sum_{k=1}^{s}\sum_{l=1}^{d} \mathbf{B}_{k,l}^{(r)} z^{s-k+(l-1)s(t+1)}. \tag{43}$$

Each worker p computes the encoded matrices as

$$\begin{aligned} \mathbf{B}_p^{(\kappa)} &\triangleq \sum_{r \in [1,L]} \mathbf{F}_{\mathbf{B}^{(r)}}([\mathbf{q}_p^{(\kappa)}]_r) \\ &= \mathbf{F}_{\mathbf{B}^{(\kappa)}}(z_{\kappa,p}) + \sum_{r \in [1,L] \setminus \kappa} \mathbf{F}_{\mathbf{B}^{(r)}}(z_r), \end{aligned} \tag{44}$$

where $[\mathbf{q}_p^{(\kappa)}]_r$ denotes the rth element of the query vector $\mathbf{q}_p^{(\kappa)}$.

After encoding the library, each worker p computes the matrix product $\mathbf{C}_p^{(\kappa)} = \mathbf{A}_p^{(\kappa)} \mathbf{B}_p^{(\kappa)}$ and then sends $\mathbf{C}_p^{(\kappa)}$ back to the master server. We note that both polynomials $\mathbf{F}_{\mathbf{A}}(z)$ and $\mathbf{F}_{\mathbf{B}^{(\kappa)}}(z)$, assigned to the input matrix $\mathbf{A}$ and the desired matrix $\mathbf{B}^{(\kappa)}$, are evaluated at the same random points $z_{\kappa,1}, \dots, z_{\kappa,P}$ for workers $1, \dots, P$, respectively. Since each undesired matrix is evaluated at an identical random point for all workers the second term in (44), i.e., $\sum_{r \in [1,L] \setminus \kappa} \mathbf{F}_{\mathbf{B}^{(r)}}(z_r)$, can be considered as a constant term.

To reconstruct all blocks $\mathbf{C}_{i,l}^{(\kappa)}$ of the product matrix $\mathbf{C}^{(\kappa)} = \mathbf{AB}^{(\kappa)}$, the master server carries out polynomial interpolation, upon receiving a number of multiplication results equal to at least $\deg(\mathbf{F}_{\mathbf{A}}(z)\mathbf{G}_{\mathbf{B}^{(\kappa)}}(z)) + 1$, which is $s(t+1)d$, for the case $s < t$.

Similarly, for the $s \geq t$ case, the polynomial encoding function for the input matrix $\mathbf{A}$ as in (34) for $P_C = 1$, that is,

$$\mathbf{F}_{\mathbf{A}}(z) = \sum_{i=1}^{t}\sum_{j=1}^{s} \mathbf{A}_{i,j} z^{i-1+t(j-1)} + \mathbf{R} z^{ts}, \tag{45}$$

and the encoding function for matrices $\mathbf{B}^{(r)}$ is given as in (35) for $P_C = 0$, that is

$$\mathbf{F}_{\mathbf{B}^{(r)}}(z) = \sum_{k=1}^{s}\sum_{l=1}^{d} \mathbf{B}_{k,l}^{(r)} z^{(s-k)t+ts(l-1)}. \tag{46}$$

The encoded matrices $\mathbf{A}_p^{(\kappa)}$ and $\mathbf{B}_p^{(\kappa)}$ are defined as above, and so are the query vectors $\mathbf{q}_p^{(\kappa)}$ for all $p \in [1, P]$.

The security of the data matrix $\mathbf{A}$ against non-colluding workers is guaranteed by appending the random matrix $\mathbf{R}$ to the input matrix $\mathbf{A}$ in (41) in the same way as described in Section IV. The details for both cases $s < t$ and $s \geq t$ are given in the proofs of Theorems 1 and 2, respectively, for the case of $P_C = 1$. The privacy condition of (9) follows by definition of the query vectors (42) for the desired index $\kappa \in [1, L]$, as proved in [38]. Finally, the recovery threshold and the communication load follow in a manner analogous to Theorems 1 and 2.

References

[1] M. Aliasgari, O. Simeone, and J. Kliewer, "Distributed and private coded matrix computation with flexible communication load," in *Proc. IEEE Intern. Symp. Inform. Theory (ISIT)*, Jul. 2019, pp. 1092–1096.

[2] M. Janzamin, H. Sedghi, and A. Anandkumar, "Beating the perils of non-convexity: Guaranteed training of neural networks using tensor methods," *arXiv preprint, arXiv:1506.08473*, 2015.

[3] M. Li, D. G. Andersen, J. W. Park, A. J. Smola, A. Ahmed, V. Josifovski, J. Long, E. J. Shekita, and B.-Y. Su, "Scaling distributed machine learning with the parameter server." in *Proc. of the 11th USENIX Conference on Operating Systems Design and Implementation, OSDI*, vol. 14, Oct. 2014, pp. 583–598.

[4] J. Dean and L. A. Barroso, "The tail at scale," *Communications of the ACM*, vol. 56, no. 2, pp. 74–80, Feb. 2013.

[5] 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. 12:1–12:30, Apr. 2017.

[6] D. Wang, G. Joshi, and G. Wornell, "Using straggler replication to reduce latency in large-scale parallel computing," *ACM SIGMETRICS Performance Evaluation Review*, vol. 43, no. 3, pp. 7–11, Dec. 2015.

[7] K.-H. Huang and J. A. Abraham, "Algorithm-based fault tolerance for matrix operations," *IEEE Trans. on Computers*, vol. 100, no. 6, pp. 518–528, Jun. 1984.

[8] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran, "Speeding up distributed machine learning using codes," *IEEE Trans. on Inform. Theory*, vol. 64, no. 3, pp. 1514–1529, Aug. 2017.

[9] Q. Yu, M. Maddah-Ali, and S. Avestimehr, "Polynomial codes: an optimal design for high-dimensional coded matrix multiplication," in *Proc. Advances in Neural Inform. Processing Systems*, Dec. 2017, pp. 4403–4413.

[10] S. Li, M. A. Maddah-Ali, Q. Yu, and A. S. Avestimehr, "A fundamental tradeoff between computation and communication in distributed computing," *IEEE Trans. on Inform. Theory*, vol. 64, no. 1, pp. 109–128, Sep. 2017.

[11] M. Aliasgari, J. Kliewer, and O. Simeone, "Coded computation against processing delays for virtualized cloud-based channel decoding," *IEEE Trans. on Commun.*, vol. 67, no. 1, pp. 28–38, Jan. 2019.

[12] ——, "Coded computation against straggling decoders for network function virtualization," in *Proc. IEEE Intern. Symp. Inform. Theory (ISIT)*, Jun. 2018, pp. 711–715.

[13] S. Dutta, M. Fahim, F. Haddadpour, H. Jeong, V. Cadambe, and P. Grover, "On the optimal recovery threshold of coded matrix multiplication," *arXiv preprint, arXiv:1801.10292*, 2018.

[14] 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 for matrix multiplication," *arXiv preprint, arXiv:1811.10751*, 2018.

[15] M. Fahim, H. Jeong, F. Haddadpour, S. Dutta, V. Cadambe, and P. Grover, "On the optimal recovery threshold of coded matrix multiplication," in *Proc. 55th Allerton Conf. Commun., Control, Comput., IL, USA*, Oct. 2017, pp. 1264–1270.

[16] M. Fahim and V. R. Cadambe, "Numerically stable polynomially coded computing," *arXiv preprint, arXiv:1903.08326*, 2019.

[17] A. M. Subramaniam, A. Heidarzadeh, and K. R. Narayanan, "Random khatri-rao-product codes for numerically-stable distributed matrix multiplication," *arXiv preprint, arXiv:1907.05965*, 2019.

[18] H. A. Nodehi and M. A. Maddah-Ali, "Limited-sharing multi-party computation for massive matrix operations," in *Proc. IEEE Intern. Symp. on Inform. Theory (ISIT)*, Jun. 2018, pp. 1231–1235.

[19] Q. Yu, N. Raviv, J. So, and A. S. Avestimehr, "Lagrange coded computing: Optimal design for resiliency, security and privacy," *arXiv preprint, arXiv:1806.00939*, 2018.

[20] W.-T. Chang and R. Tandon, "On the capacity of secure distributed matrix multiplication," *arXiv preprint, arXiv:1806.00469*, 2018.

[21] 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, Apr. 2019.

[22] H. Yang and J. Lee, "Secure distributed computing with straggling servers using polynomial codes," *IEEE Trans. on Inform. Forensics and Secur.*, vol. 14, no. 1, pp. 141–150, Jan. 2019.

[23] R. G. D'Oliveira, S. E. Rouayheb, and D. Karpuk, "GASP codes for secure distributed matrix multiplication," *arXiv preprint, arXiv:1812.09962*, 2018.

[24] A. B. Das, A. Ramamoorthy, and N. Vaswani, "Random convolutional coding for robust and straggler resilient distributed matrix computation," *arXiv preprint, arXiv:1907.08064*, 2019.

[25] H. A. Nodehi and M. A. Maddah-Ali, "Secure coded multi-party computation for massive matrix operations," *arXiv preprint, arXiv:1908.04255*, 2019.

[26] F. Ricci, L. Rokach, and B. Shapira, *Introduction to recommender systems handbook*. Springer, 2011.

[27] K. Lee, C. Suh, and K. Ramchandran, "High-dimensional coded matrix multiplication," in *Proc. IEEE Intern. Symp. Inform. Theory (ISIT)*, Jun. 2017, pp. 2418–2422.

[28] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Straggler mitigation in distributed matrix multiplication: Fundamental limits and optimal coding," *arXiv preprint, arXiv:1801.07487*, 2018.

[29] Z. Jia and S. A. Jafar, "On the capacity of secure distributed matrix multiplication," *arXiv preprint, arXiv:1908.06957*, 2019.

[30] 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.

[31] W. Gasarch, "A survey on private information retrieval," *Bulletin of the EATCS*, vol. 82, no. 113, pp. 72–107, Feb. 2004.

[32] S. Yekhanin, "Private information retrieval," *Commun. ACM*, vol. 53, no. 4, pp. 68–73, Apr. 2010.

[33] H. Sun and S. A. Jafar, "The capacity of private information retrieval," *IEEE Trans. on Inform. Theory*, vol. 63, no. 7, pp. 4075–4088, Jul. 2017.

[34] K. Banawan and S. Ulukus, "The capacity of private information retrieval from coded databases," *IEEE Trans. on Inform. Theory*, vol. 64, no. 3, pp. 1945–1956, Mar. 2018.

[35] R. Freij-Hollanti, O. W. Gnilke, C. Hollanti, and D. A. Karpuk, "Private information retrieval from coded databases with colluding servers," *SIAM J. Appl. Algebra Geom.*, vol. 1, no. 1, pp. 647–664, Nov. 2017.

[36] F. Kazemi, E. Karimi, A. Heidarzadeh, and A. Sprintson, "Single-server single-message online private information retrieval with side information," in *Proc. IEEE Intern. Symp. Inform. Theory (ISIT)*, Jul. 2019, pp. 350–354.

[37] ——, "Private information retrieval with private coded side information: The multi-server case," *arXiv preprint, arXiv:1906.11278*, 2019.

[38] M. Kim and J. Lee, "Private secure coded computation," *arXiv preprint, arXiv:1902.00167*, 2019.

[39] W.-T. Chang and R. Tandon, "On the upload versus download cost for secure and private matrix multiplication," *arXiv preprint, arXiv:1906.10684*, 2019.

[40] B. Tahmasebi and M. A. Maddah-Ali, "Private sequential function computation," *arXiv preprint, arXiv:1908.01204*, 2019.

[41] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.

[42] H.-T. Kung, *Fast evaluation and interpolation*. Carnegie Mellon University, Tech. Rep., 2009.

[43] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran, "Speeding up distributed machine learning using codes," *IEEE Trans. on Inform. Theory*, vol. 64, no. 3, pp. 1514–1529, Mar. 2017.

[44] S. M. Ross, *Introduction to Probability Models*. Academic Press, 2014.

[45] S. Ha, J. Zhang, O. Simeone, and J. Kang, "Coded federated computing in wireless networks with straggling devices and imperfect CSI," in *Proc. IEEE Intern. Symp. Inform. Theory (ISIT)*, Jul. 2019, pp. 2649–2653.