Sparsity and Privacy in Secret Sharing: A Fundamental Trade-Off

Rawad Bitar, *Member, IEEE*, Maximilian Egger, *Graduate Student Member, IEEE*, Antonia Wachter-Zeh, *Senior Member, IEEE*, and Marvin Xhemrishi, *Graduate Student Member, IEEE*

Abstract— **This work investigates the design of sparse secret sharing schemes that encode a sparse private matrix into sparse shares. This investigation is motivated by distributed computing, where the multiplication of sparse and private matrices is moved from a computationally weak main node to untrusted worker machines. Classical secret-sharing schemes produce dense shares. However, sparsity can help speed up the computation. We show that, for matrices with i.i.d. entries, sparsity in the shares comes at a fundamental cost of weaker privacy. We derive a fundamental tradeoff between sparsity and privacy and construct optimal sparse secret sharing schemes that produce shares that leak the minimum amount of information for a desired sparsity of the shares. We apply our schemes to distributed sparse and private matrix multiplication schemes with no colluding workers while tolerating stragglers. For the setting of two non-communicating clusters of workers, we design a sparse one-time pad so that no private information is leaked to a cluster of untrusted and colluding workers, and the shares with bounded but non-zero leakage are assigned to a cluster of partially trusted workers. We conclude by discussing the necessity of using permutations for matrices with correlated entries.**

Index Terms— **Sparse private matrix multiplication, straggler tolerance, optimal leakage, information-theoretic privacy.**

I. INTRODUCTION

DISTRIBUTED computing became a ubiquitous requirement with the emergence of machine learning applications where an enormous amount of data is processed and, due to limitations of the computation power, forwarded to computing nodes [4]. We consider a main node/worker setting, where a central entity called *main node* owns a large amount of private data and needs to run intensive computations on it. The intensive computation is split into multiple tasks of potentially smaller complexity that can be assigned to external computation nodes called *workers* to be run in parallel.

Offloading computations to external nodes faces several challenges, such as leaking private information from sensitive data, which could be prohibited by data-protection laws [5],

and increased overall latency due to straggler behavior, i.e., slow or unresponsive nodes [6].

We restrict our attention to distributed matrix multiplication as being a computationally exhaustive part of several machine learning algorithms, such as support vector machines, principal component analysis, and gradient descent algorithms [7], [8]. A vast literature on the topic shows that coding-theoretic techniques can be used to mitigate the effect of stragglers and to ensure strong information-theoretic privacy of the data, e.g., [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], and [29] and the survey in [30].

However, those techniques require "mixing" the private data with random matrices (for privacy) to generate the tasks sent to the workers. Hence, all structure of the input matrices, particularly sparsity, is lost. Sparse matrices, i.e., matrices with a relatively large number of zero entries, appear naturally in some machine learning tasks [31]. The structure of sparse matrices is important and can be leveraged by deploying efficient storage and computations strategies [32], [33].

In this work, we introduce the problem of encoding sparse private matrices into matrices that are still sparse and, when offloaded to workers, maintain information-theoretical privacy guarantees as will be formally defined in the sequel. We explain next the closest related works and highlight our contributions.

A. Related Work

Straggler mitigation in non-private distributed matrix multiplication is extensively studied in the literature, e.g., [9], [10], [11], [12], [13], [14], [15], and [16]. The idea is to encode the input matrices into new ones, called *tasks*, such that stragglers can be mitigated when assigned to the workers. Sparsity-preserving encoding techniques in distributed matrix multiplication have recently received attention in the scientific community. The authors of [34] propose encoding the input matrices using an LT code [35] with a carefully tailored Soliton distribution to generate sparse matrices (tasks). In [36], the authors design encoding strategies for many settings, including distributing the multiplication of a sparse matrix with a dense vector. Fractional repetition codes are used to preserve a plausible level of sparsity in the tasks and mitigate stragglers. Moreover, an extra layer of non-sparse coded matrices is added to increase the straggler mitigation capability of the proposed codes. In [37], the same authors improve their previous scheme [36] by introducing a sparser encoding strategy of the last layer. In [38], the authors propose

sparse encoding strategies that account for better numerical stability properties (compared to other existing schemes) and for heterogeneous task allocations (proportional to the workers storage capabilities). In [39], the authors propose to use a random Khatri-Rao product to construct a distributed (non-private) matrix-matrix multiplication that is numerically stable. The work [40] modifies the construction of [39] to obtain sparse encoding strategies with similar numerical stability when the input matrices are sparse. The works in [41] and [42] consider the case where a dense input matrix is sparsified to speed up the overall computation.

In terms of privacy, several encoding strategies that mitigate stragglers and guarantee perfect information-theoretic privacy have been proposed. The literature distinguishes between one-sided privacy, where the main node wants to offload the multiplication of two matrices, one of which has to remain private, e.g., [22], [26], [27], and [28], and double-sided-privacy where both input matrices must remain private, e.g., [17], [19], [20], [21], [24], [25], [29], [43], and [44]. However, none of the works on private distributed matrix multiplication considered encoding strategies that generate tasks that inherit the sparsity of the input matrices. The reason is that, as we shall show next, perfect information-theoretic privacy requires encoding strategies that destroy the sparsity of the input matrices. Very recently, [45] studied the tradeoff between privacy and storage rate in distributed storage.

B. Contributions

We introduce the problem of constructing codes for private distributed matrix multiplication that mitigate stragglers and inherit sparsity from the private input matrices. We start by showing that insisting on perfect information-theoretic privacy does not allow leveraging the beneficial properties of sparse matrices. Therefore, we open the door to allowing sparsity and privacy to co-exist by relaxing the privacy guarantees. We prove the existence of a fundamental trade-off between sparsity and information-theoretic privacy. We optimize this trade-off to design optimal codes for private distributed matrix multiplication with sparsity guarantees.

C. Organization

We set the notations in Section II and formulate the general problem in Section III. In Section IV, we describe the considered system model and our main results. In Section V, we construct a novel sparse one-time padding strategy and optimize it for given requirements. Section VI generalizes into a sparse secret-sharing scheme with an arbitrary number of shares. In Section VII, we explain how to apply the introduced schemes to sparse and private matrix multiplication with stragglers. We discuss practical solutions for sparse matrices with correlated entries in Section VIII and conclude the paper in section IX.

II. NOTATION

We use uppercase and lowercase bold letters to denote matrices and vectors, e.g., $\mathbf{X}$ and $\boldsymbol{x}$, respectively. By $\mathbf{X}_{\{i,j\}}$,

we denote the (i, j)-th entry of a matrix $\mathbf{X}$. We use letters in uppercase *typewriter* font for random variables, e.g., $\mathtt{Y}$. The random variables representing a matrix $\mathbf{X}$ and its (i, j)-th entry $\mathbf{X}_{\{i,j\}}$ are denoted by $\mathtt{X}$ and $\mathtt{X}_{\{i,j\}}$, respectively. A finite field of cardinality q is denoted by $\mathbb{F}_q$ and we use $\mathbb{F}_q^*$ to define its multiplicative group, i.e., $\mathbb{F}_q^* \triangleq \mathbb{F}_q \setminus \{0\}$. Sets are denoted by letters in calligraphic font, e.g., $\mathcal{X}$. For a positive integer b, we define the set $[b] \triangleq \{1, 2, \ldots, b\}$. Given b random variables $\mathtt{Y}_1, \ldots, \mathtt{Y}_b$ and a set $\mathcal{I} \subseteq [b]$, we denote by $\{\mathtt{Y}_i\}_{i \in \mathcal{I}}$ the set of random variables indexed by $\mathcal{I}$, i.e., $\{\mathtt{Y}_i\}_{i \in \mathcal{I}} \triangleq \{\mathtt{Y}_i | i \in \mathcal{I}\}$. For two random variables $\mathtt{X}$ and $\mathtt{Y}$, we denote $\mathrm{Pr}(\mathtt{X} = x)$ and $\mathrm{Pr}(\mathtt{X} = x | \mathtt{Y} = y)$ by $\mathrm{Pr}_{\mathtt{X}}(x)$ and $\mathrm{Pr}_{\mathtt{X}|\mathtt{Y}}(x|y)$, respectively. The probability mass function (PMF) of a random variable $\mathtt{X} \in \mathbb{F}_q$ is denoted by $\mathrm{P}_{\mathtt{X}} = [p_1, p_2, \ldots, p_q]$, i.e., $\mathrm{Pr}_{\mathtt{X}}(i) = p_i$ for all $i \in \mathbb{F}_q$. Throughout the paper, we will use $\mathrm{H}_q(\mathtt{X})$ and $\mathrm{H}_q([p_1, p_2, \ldots, p_q])$ interchangeably to denote the q-ary entropy of the random variable $\mathtt{X} \in \mathbb{F}_q$. The q-ary mutual information between two random variables $\mathtt{X}$ and $\mathtt{Y}$ is denoted by $\mathrm{I}_q(\mathtt{X}; \mathtt{Y})$. We will use $\mathrm{D}_{\mathrm{KL}}(\mathrm{P}_{\mathtt{X}} \| \mathrm{P}_{\mathtt{Y}})$ to denote the Kullback-Leibler (KL) divergence between the PMFs of $\mathtt{X}$ and $\mathtt{Y}$.

III. PROBLEM FORMULATION

We introduce the problem of encoding private and sparse matrices into sparse matrices such that observing a collection of the output matrices reveals little to no information about the input matrices. In other words, the problem can be understood as constructing secret sharing schemes whose output is a collection of sparse matrices.

Formally, let $\mathbf{A}_1, \ldots, \mathbf{A}_k$ be k private matrices of the same dimensions drawn from a finite alphabet $\mathcal{A}$.

Definition 1 (Secret Sharing): An (n, t, z) secret sharing (SecShare) scheme with $t = k + z$ is a randomized encoding SecShare : $\mathcal{A}^k \times \mathcal{A}^z \to \mathcal{A}^n$ *that outputs n matrices, called shares. Let $\mathtt{A}_1, \ldots, \mathtt{A}_k$ and $\widetilde{\mathtt{A}}_1, \ldots, \widetilde{\mathtt{A}}_n$ be random variables whose realizations are the private matrices and the output shares, respectively. A secret sharing scheme satisfies the following conditions:*

1) Reconstruction: *any t shares suffice to reconstruct the input matrices, i.e., $\mathrm{H}_q(\mathtt{A}_1, \ldots, \mathtt{A}_k | \widetilde{\mathtt{A}}_{i_1}, \ldots, \widetilde{\mathtt{A}}_{i_t}) = 0$.*
2) Perfect privacy: *any z out of the n shares reveal no information about the private matrices, i.e., $\mathrm{I}_q(\mathtt{A}_1, \ldots, \mathtt{A}_k; \widetilde{\mathtt{A}}_{i_1}, \ldots, \widetilde{\mathtt{A}}_{i_z}) = 0$.*

Constructions of (n, t, z) secret sharing schemes generate z random matrices $\mathbf{R}_1, \ldots, \mathbf{R}_z$ drawn independently and uniformly at random from $\mathcal{A}$. The scheme then computes the polynomial $\mathbf{F}(x) = \sum_{i=0}^{k-1} \mathbf{A}_{i+1} x^i + \sum_{i=0}^{z-1} \mathbf{R}_{i+1} x^{k+i}$. Let $\alpha_1, \ldots, \alpha_n$ be distinct non-zero numbers. Each share $\widetilde{\mathbf{A}}_i$ is the evaluation of $\mathbf{F}(x)$ at α_i, i.e., $\widetilde{\mathbf{A}}_i = \mathbf{F}(\alpha_i)$.

The problem with secret sharing schemes is that the shares are dense, i.e., have a relatively small number of zero entries, irrespective of the structure of the input matrices. This is due to drawing the matrices $\mathbf{R}_i$ independently and uniformly at random from $\mathcal{A}$, which is crucial for the perfect privacy to hold. Let $s_{\mathbf{A}_i}$ be the sparsity[1] of the input matrices $\mathbf{A}_i$.

---

[1]The sparsity could be understood in several ways, e.g., the ratio of the number of zero entries to the total number of entries, and will be defined formally when we explain the system model.

Secret sharing schemes output shares with sparsity equal $1/|\mathcal{A}|$ irrespective of the values of the $s_{\mathbf{A}_i}$'s.

We first investigate the existence of sparse secret sharing schemes (as defined next), which are of independent interest.

Definition 2 (Sparse Secret Sharing): A sparse secret sharing scheme with parameters $(n, t, z; \varepsilon, s_1, \ldots, s_n)$ is a randomized encoding SparseSecShare : $\mathcal{A}^k \times \mathcal{A}^z \to \mathcal{A}^n$ that takes as input $k = t - z$ private and sparse matrices with sparsity $s_{\mathbf{A}_i}$, $i = 1, \ldots, k$, and outputs n sparse matrices called shares that satisfy the reconstruction property and the following:

1) Privacy: *any z out of the n shares leak a small amount of information about the private matrices, i.e.,*

$$I_q(\mathbb{A}_1, \ldots, \mathbb{A}_k; \widetilde{\mathbb{A}}_{i_1}, \ldots, \widetilde{\mathbb{A}}_{i_z}) \leq \varepsilon.$$

2) Sparsity: *Each share $\widetilde{\mathbf{A}}_i$ has a desired sparsity $s_{\widetilde{\mathbf{A}}_i} = s_i$.*

The observant reader notices immediately that the privacy guarantee of sparse secret sharing schemes is relaxed from the perfect information-theoretic privacy to weak information-theoretic privacy that allows what we call an ε leakage. We shall show that there exists a tension between ε and $s_{\widetilde{\mathbf{A}}_i}$, $i = 1, \ldots, n$, and that if $\varepsilon = 0$ is desired, then $s_{\widetilde{\mathbf{A}}_i} \leq 1/|\mathcal{A}|$ which is satisfied by regular secret sharing schemes. We focus on constructing sparse secret sharing schemes with $z = 1$ and $k = 1$, i.e., schemes with shares of the form $\widetilde{\mathbf{A}}_i = \mathbf{A} + \mathbf{R}\alpha_i$.

With sparse secret-sharing schemes, we define the problem of constructing straggler-tolerant sparse and private matrix multiplication schemes.

Definition 3 (Sparse and Private Matrix Multiplication): A straggler-tolerant sparse and private matrix multiplication scheme with parameters $(N, \sigma, z, \varepsilon_1, \varepsilon_2, s_1, s_2)$ takes as input two private and sparse matrices $\mathbf{A}$ and $\mathbf{B}$ and offloads their multiplication to N available worker nodes. The matrices are encoded each into $m \geq N$ shares, denoted by $\widetilde{\mathbf{A}}_1, \ldots, \widetilde{\mathbf{A}}_m, \widetilde{\mathbf{B}}_1, \ldots, \widetilde{\mathbf{B}}_m$. The shares are grouped into pairs called tasks and denoted by $\mathbf{T}_{(i,j)} = (\widetilde{\mathbf{A}}_i, \widetilde{\mathbf{B}}_j)$, for $i, j \in [m]$, which are then assigned to the N workers. Denote by $\mathcal{W}_\ell$ the set of all tasks assigned to worker ℓ, $\ell \in [N]$, i.e., $\mathcal{W}_\ell \triangleq \{\mathbf{T}_{(i,j)} \mid \mathbf{T}_{(i,j)} \text{ assigned to worker } \ell\}$. Each worker computes sequentially the multiplication of all $(\widetilde{\mathbf{A}}_i, \widetilde{\mathbf{B}}_j) \in \mathcal{W}_\ell$ and sends them back to the main node until either it finishes all its tasks or it is asked to stop computing. Let $\mathcal{C}_\ell$ be the collection of multiplications sent by worker ℓ to the main node. The scheme satisfies the following conditions:

1) Straggler tolerance: *The matrix $\mathbf{C} = \mathbf{AB}$ can be recovered from the collection of any $N - \sigma$ sets $\mathcal{C}_\ell$, i.e., with a slight abuse of notation[2] we can write for any distinct $i_1, \ldots, i_{N-\sigma} \in \{1, \ldots, N\}$*

$$H_q(\mathbb{C} \mid \mathcal{C}_{i_1}, \ldots, \mathcal{C}_{i_{N-\sigma}}) = 0.$$

2) Privacy: *Any z workers can obtain a small amount of information about the input matrices, i.e.,*

$$I_q(\mathbb{A}; \mathcal{W}_{i_1}, \ldots, \mathcal{W}_{i_z}) \leq \varepsilon_1, \quad I_q(\mathbb{B}; \mathcal{W}_{i_1}, \ldots, \mathcal{W}_{i_z}) \leq \varepsilon_2,$$

where $0 \leq \varepsilon_1 \leq H_q(\mathbb{A})$ and $0 \leq \varepsilon_2 \leq H_q(\mathbb{B})$.

3) Sparsity: *The average sparsity of the tasks in $\mathcal{W}_\ell$ corresponding to matrices $\mathbf{A}$ and $\mathbf{B}$ are respectively equal to $s_1 \leq s_{\mathbf{A}}$ and $s_2 \leq s_{\mathbf{B}}$.*

IV. SYSTEM MODEL AND MAIN RESULTS

A. Sparsity and Private Matrices

In this work, we restrict our attention to private matrices whose entries are independent and identically distributed.

Definition 4 (Sparsity Level of a Matrix): The sparsity level $S(\mathbb{X})$ of a matrix $\mathbb{X}$ with entries independently and identically distributed is equal to the probability of the (i, j)-th entry $\mathbb{X}_{\{i,j\}}$ of $\mathbb{X}$ being equal to 0, i.e.,

$$S(\mathbb{X}) = \Pr_{\mathbb{X}_{\{i,j\}}}(0).$$

The private matrices of the main node are assumed to belong to a finite field $\mathbf{A} \in \mathbb{F}_q^{m \times n}$, $\mathbf{B} \in \mathbb{F}_q^{n \times \ell}$ and have sparsity levels $S(\mathbf{A}) = s_{\mathbf{A}}$ and $S(\mathbf{B}) = s_{\mathbf{B}}$, where[3] $s_{\mathbf{A}}, s_{\mathbf{B}} > q^{-1}$. The PMF of the entries of $\mathbf{A}$ (and $\mathbf{B}$) can be expressed as follows

$$\Pr_{\mathbb{A}_{\{i,j\}}}(a) = \begin{cases} s_{\mathbf{A}}, & a = 0 \\ \dfrac{1 - s_{\mathbf{A}}}{q - 1}, & a \in \mathbb{F}_q^* \end{cases}. \tag{1}$$

Though limiting, the assumption of the entries being i.i.d. serves as a stepping stone toward understanding the trade-off between privacy and sparsity. The goal is for the main node to offload the computation $\mathbf{C} = \mathbf{AB}$ to N workers.

B. Collusions and Stragglers

The workers are assumed to be *honest-but-curious* and will follow the protocol dictated by the main node. We say that up to z workers *collude* if any collection of up to z workers collaborate to infer information about the private matrices from the tasks they have obtained. For the particular case of $z = 1$, we say that the workers do not collude.

Workers are assigned multiple computational tasks that they compute and send their results sequentially to the main node. A *full straggler* is a worker that does not return any computation to the main node. A *partial straggler* is a worker that returns some of its computations to the main node while most of the other workers have finished all their computations.

C. Information Leakage

Let the random variables $\mathbb{A}$ and $\mathbb{B}$ represent the private matrices $\mathbf{A}$ and $\mathbf{B}$, respectively, and $\mathcal{W}_\ell$, $\ell \in [N]$, denote the collection of random variables representing the tasks assigned to worker ℓ. We say that the tasks assigned to worker ℓ leak $\varepsilon_{\mathbf{A}} \triangleq I_q(\mathbb{A}; \mathcal{W}_\ell)$ amount of information about the matrix $\mathbf{A}$. We define the relative leakage by $\bar{\varepsilon}_{\mathbf{A}} \triangleq I_q(\mathbb{A}; \mathcal{W}_\ell)/H_q(\mathbb{A})$, where $0 \leq \bar{\varepsilon}_{\mathbf{A}} \leq 1$. Perfect information-theoretic privacy corresponds to $\varepsilon_{\mathbf{A}} = \bar{\varepsilon}_{\mathbf{A}} = 0$ and no privacy corresponds to $\varepsilon_{\mathbf{A}} = H_q(\mathbb{A})$ or $\bar{\varepsilon}_{\mathbf{A}} = 1$. The same can be defined for $\mathbf{B}$.

Remark 1 (Operational Meaning): While the operational meaning of perfect information-theoretic privacy is clear: an eavesdropper with unbounded computational power cannot do better than guessing; a non-zero leakage has more ambiguity

---

[2]For clarity of presentation, the set $\mathcal{C}_\ell$ represents also a collection of random variables.

[3]For the special case $s_{\mathbf{A}}, s_{\mathbf{B}} \leq q^{-1}$, the shares of classical secret sharing schemes have larger sparsity q^{-1} and are, thus, enough.

in its operational meaning. We are interested in understanding the fundamental theoretical trade-off between information-theoretic privacy and sparsity. In practice, a leakage of $\varepsilon > 0$ means that any eavesdropper can learn some partial information about the private data.

D. Main Results

We study sparse secret sharing schemes for $z = 1$ and matrices with i.i.d entries. We start by showing in Lemma 1 that a fundamental trade-off exists between sparsity and privacy. Namely, given a private matrix $\mathbf{A}$ with sparsity $s_{\mathbf{A}} \geq q^{-1}$, a sparse secret sharing satisfies $s_{\widetilde{\mathbf{A}}_i} > q^{-1}$ if and only if $\varepsilon > 0$.

Lemma 1: Let $\mathbb{A}$ represent a sparse matrix that has i.i.d entries. For any secret sharing scheme with $z = 1$, perfect information-theoretic privacy can be achieved if and only if the padding matrix $\mathbb{R}$ is generated independently of $\mathbb{A}$ with its entries being i.i.d. uniformly distributed.

Proof: The proof is shown in Appendix A. □

Therefore, to provide sparsity guarantees of the encoded matrices, we have to deviate from perfect information-theoretic privacy. We start by constructing a sparse one-time pad, cf. Construction 1, that takes as input a private sparse matrix $\mathbf{A}$ and outputs two sparse shares $\mathbf{R}$ and $\mathbf{A}+\mathbf{R}$ each of which leaks a pre-specified amount of information about $\mathbf{A}$. We show the existence of a fundamental tradeoff between the achievable sparsity of the shares and the privacy guarantees. Further, given a sparse matrix $\mathbf{A}$ and fixing the same desired sparsity of the shares, Construction 1 provides shares with the provably smallest leakage, cf. Lemma 2 and Theorem 1. We focus on shares with the same sparsity levels since by Lemma 7 we know that for a desired average sparsity, constructing shares with the same sparsity levels minimizes leakage.

We remark in Lemma 3 the existence of sparse one-time pads where one share leaks no information about $\mathbf{A}$ and characterize the sparsity-privacy tradeoff in such a case. The implication of the Lemma 3 is of particular importance in the setting in which the workers can be grouped into two non-communicating clusters. One cluster consists of fully untrusted workers, while the workers of the other cluster are partially trusted. We apply the result of Lemma 3 to construct straggler-tolerant sparse and private matrix multiplication schemes that can tolerate collusions. In particular, we guarantee perfect information-theoretic privacy against all the untrusted workers and a desired leakage for any z partially trusted workers, cf. Theorem 3.

In Construction 2, we construct sparse secret sharing schemes with $t = 2, z = 1$ and $n \geq 2$. We observe that the sparsity-privacy tradeoff extends to include the number of shares, i.e., for a fixed desired sparsity level, increasing the number of shares decreases the achievable privacy guarantees. Theorem 2 shows that for the same desirable sparsity of the shares, Construction 2 provides the provably smallest leakage possible.

We provide a discussion on the effect of correlation between the entries of the matrix $\mathbf{A}$ (and $\mathbf{B}$) on the constructed schemes and show how to break the correlation with random permutations, cf. Section VIII.

V. SPARSE ONE-TIME PAD

To build intuition, in this section, we construct a sparse one-time pad, i.e., a secret sharing scheme with parameters $n = 2, t = 2, z = 1$, and we require $q^{-1} < s_1, s_2 \leq s$ (cf. Definition 2). The goal is to understand how a sparse secret sharing scheme can be constructed and to quantify the optimal leakage ε that can be obtained for fixed sparsity levels s_1, s_2.

A one-time pad [46] takes as input the matrix $\mathbf{A} \in \mathbb{F}_q^{\mathsf{m} \times \mathsf{n}}$, generates a matrix $\mathbf{R} \in \mathbb{F}_q^{\mathsf{m} \times \mathsf{n}}$ whose entries are generated i.i.d uniformly at random from $\mathbb{F}_q$ and outputs $\mathbf{R}$ and $\mathbf{A} + \mathbf{R}$ as shares. The key ingredient to obtaining a *sparse* one-time pad is to generate the entries of $\mathbf{R}$ *dependently* on $\mathbf{A}$. We consider generating $\mathbf{R}$ by using a conditional PMF as in Construction 1.

Construction 1 (Generation of the Padding Matrix $\mathbf{R}$): To construct a sparse one-time pad, the entries of the padding matrix $\mathbf{R}$ are generated dependently on the entries of $\mathbf{A}$ as[4]

$$\Pr_{\mathbb{R}_{\{i,j\}}|\mathbb{A}_{\{i,j\}}}(r|0) = \begin{cases} p_1, & r = 0 \\ p_1^{\text{inv}}, & r \neq 0, \end{cases}, \tag{2}$$

$$\Pr_{\mathbb{R}_{\{i,j\}}|\mathbb{A}_{\{i,j\}}}(r|a) = \begin{cases} p_2, & r = 0 \\ p_3, & r = -a \\ p_{2,3}^{\text{inv}}, & r \notin \{0, -a\}, \end{cases}, \tag{3}$$

where $p_1^{\text{inv}} \triangleq \frac{1-p_1}{q-1}$ and $p_{2,3}^{\text{inv}} \triangleq \frac{1-p_2-p_3}{q-2}$.

The motivation behind this choice of PMF stems from the entries of $\mathbf{A}$ being i.i.d with uniform distribution over the non-zero values of $\mathbb{F}_q$. Hence, the entries of $\mathbf{R}$ are also chosen i.i.d. with a high probability for the zero values and a uniform distribution over the non-zero values, as explained next. Intuitively, the PMFs presented in Eqs. (2) and (3) ensure sparsity and privacy as follows. The event occurring with probability p_1 ensures that a zero entry in the matrix $\mathbf{A}$ is inherited in both $\mathbf{A} + \mathbf{R}$ and $\mathbf{R}$. The events with probabilities p_2 and p_3 create a zero entry in $\mathbf{R}$ and $\mathbf{A} + \mathbf{R}$, respectively, where there was a non-zero entry in $\mathbf{A}$. The events with probabilities p_1^{inv} and $p_{2,3}^{\text{inv}}$ create non-zero entries in both $\mathbf{R}$ and $\mathbf{A} + \mathbf{R}$ irrespective of the value of the corresponding entry of $\mathbf{A}$. We do not distinguish cases between different values of the non-zero entries of $\mathbf{A}$ since we assume they are uniformly distributed over $\mathbb{F}_q \setminus \{0\}$. We numerically validate our construction by observing that a sparse one-time pad with desired sparsity levels attains the minimum leakage possible when considering matrices $\mathbf{R}$ as in Eqs. (2) and (3).

A. Analysis of Sparsity and Leakage

Proposition 1 shows the effect of p_1, p_2 and p_3 on the sparsity of $\mathbf{R}$ and $\mathbf{A} + \mathbf{R}$. The increase of p_1 increases both sparsity levels $s_{\mathbf{R}}$ and $s_{\mathbf{A}+\mathbf{R}}$. On the other hand, the increase of p_2 and p_3 increases only $s_{\mathbf{R}}$ and $s_{\mathbf{A}+\mathbf{R}}$, respectively.

Proposition 1: Given an input matrix $\mathbf{A}$ with sparsity level $s_{\mathbf{A}}$ whose entries are i.i.d, and if a padding matrix $\mathbf{R}$ is constructed using the conditional PMFs as in Eqs. (2) and (3), then the sparsity levels $s_{\mathbf{R}}$ and $s_{\mathbf{A}+\mathbf{R}}$ are

$$s_{\mathbf{R}} = p_1 s_{\mathbf{A}} + p_2(1 - s_{\mathbf{A}}) \leq 1, \tag{4}$$

---

[4]Similarly, one can define a PMF that generates a padding matrix $\mathbf{S}$ depending on $\mathbf{B}$.

$$s_{\mathbf{A+R}} = p_1 s_{\mathbf{A}} + p_3(1 - s_{\mathbf{A}}) \leq 1. \qquad (5)$$

Proof: The proof follows by calculating the PMF of $\mathtt{R}$ and $\mathtt{A + R}$ and applying the definition of sparsity. $\square$

Note that from Construction 1 and Proposition 1, we have that $s_{\mathbf{R}} + s_{\mathbf{A+R}} \leq 1 + s_{\mathbf{A}}$. At first glance, it may seem that increasing p_1, p_2, and p_3 is beneficial. However, increasing those parameters will also increase the leakage of $\mathbf{R}$ and $\mathbf{A+R}$. Observe that choosing $p_1 = p_2 = 1$ and $p_3 = 0$ results in the highest sparsity possible of $\mathbf{R}$ equal to one and allow $\mathbf{A + R}$ to have the same sparsity level of $\mathbf{A}$. However, $\mathbf{R}$ is the all-zero matrix. Despite $\mathbf{R}$ leaking no information about $\mathbf{A}$, observing $\mathbf{A+R = A}$ reveals all the entries of $\mathbf{A}$. To understand this tension between privacy and sparsity, we quantify the minimum leakage that can be obtained from both $\mathbf{R}$ and $\mathbf{A+R}$ for fixed $s_{\mathbf{R}}$ and $s_{\mathbf{A+R}}$.

Define $\mathsf{L_R} \triangleq I_q(\mathtt{R}; \mathtt{A})$ and $\mathsf{L_{A+R}} \triangleq I_q(\mathtt{A + R}; \mathtt{A})$ to be the leakage of $\mathbf{R}$ and $\mathbf{A + R}$, respectively. Minimizing the leakage amounts to minimizing those quantities. Since the entries of $\mathbf{A}$ are assumed to be i.i.d. and the entries of $\mathbf{R}$ are also i.i.d, we can write $\mathsf{L_R} = mn\mathsf{L}_1 \triangleq mnI_q(\mathtt{R}_{\{i,j\}}; \mathtt{A}_{\{i,j\}})$ and $\mathsf{L_{A+R}} = mn\mathsf{L}_2 \triangleq mnI_q(\mathtt{A}_{\{i,j\}} + \mathtt{R}_{\{i,j\}}; \mathtt{A}_{\{i,j\}})$. It then suffices to minimize L_1 and L_2. Our goal is to minimize the total leakage of a sparse one-time pad defined as $\mathsf{L_{tot}} \triangleq \mathsf{L}_1 + \mathsf{L}_2$. Minimizing the leakage is done over all possible choices of conditional PMFs $P_{\mathtt{R}_{\{i,j\}}|\mathtt{A}_{\{i,j\}}}$, i.e., over the set of all possible values of $\mathcal{P} \triangleq \{p_{r|a} : r, a \in \mathbb{F}_q\}$. Therefore, minimizing the leakage is equivalent to the following minimization problem[5]

$$\begin{aligned}
\mathsf{L_{opt}} &= \min_{\mathcal{P}} \mathsf{L_{tot}} = \min_{\mathcal{P}} \mathsf{L}_1 + \mathsf{L}_2 \\
&= \min_{\mathcal{P}} I_q(\mathtt{R}_{\{i,j\}}; \mathtt{A}_{\{i,j\}}) + I_q(\mathtt{A}_{\{i,j\}} + \mathtt{R}_{\{i,j\}}; \mathtt{A}_{\{i,j\}}) \\
&= \min_{\mathcal{P}} D_{KL}\left(P_{\mathtt{A,R}} \| P_{\mathtt{A}} P_{\mathtt{R}}\right) + D_{KL}\left(P_{\mathtt{A,A+R}} \| P_{\mathtt{A}} P_{\mathtt{A+R}}\right) \\
&= \min_{\mathcal{P}} \sum_{a,b \in \mathbf{F}_q} P_{\mathtt{A}}(a) \\
&\quad \times \left(p_{b-a|a} \log \frac{p_{b-a|a}}{P_{\mathtt{A+R}}(b)} + p_{b|a} \log \frac{p_{b|a}}{P_{\mathtt{R}}(b)}\right).
\end{aligned}$$

This is a constrained optimization problem whose constraints follow from the requirement of having a valid PMF and the desired sparsities $s_{\mathbf{R}}$ and $s_{\mathbf{A+R}}$, and can be written as

$$\forall a \in \mathbb{F}_q : p_{0|a} + \sum_{r \in \mathbb{F}_q^\star} p_{r|a} - 1 = 0,$$

$$p_{0|0} \cdot s_{\mathbf{A}} + \sum_{a \in \mathbb{F}_q^\star} p_{0|a} \cdot P_{\mathtt{A}}(a) - s_{\mathbf{R}} = 0,$$

$$p_{0|0} \cdot s_{\mathbf{A}} + \sum_{a \in \mathbb{F}_q^\star} p_{-a|a} \cdot P_{\mathtt{A}}(a) - s_{\mathbf{A+R}} = 0.$$

Optimizing over $\mathcal{P}$ means that the optimization problem should be performed over q^2 variables. From numerical simulations, we observed that when the entries of $\mathbf{A}$ are distributed as in in Eq. (1), then the optimization boils down to optimize only over the three variables p_1, p_2 and p_3 shown in Construction 1, i.e., the probabilities that do not contribute

---

[5]For ease of presentation, we omit the subscripts $\{i, j\}$ in the second to last equality.

into the sparsity levels $s_{\mathbf{R}}$, $s_{\mathbf{A+R}}$ can be uniformly distributed. Then, the optimization problem becomes as shown next.

Lemma 2: If the entries of the private matrix $\mathbf{A}$ are distributed as in Eq. (1) and the desired sparsity levels $s_{\mathbf{R}}$, $s_{\mathbf{A+R}}$, of the padding and padded matrices, are fixed, then the optimal element-wise total leakage is given by optimizing $\min_{p_1, p_2, p_3} \mathsf{L}(p_1, p_1^{inv}, p_2, p_3, p_{2,3}^{inv})$, where

$$\begin{aligned}
&\mathsf{L}(p_1, p_1^{inv}, p_2, p_3, p_{2,3}^{inv}) \\
&= s\left[p_1\left(\log \frac{p_1}{s_{\mathbf{A+R}}} + \log \frac{p_1}{s_{\mathbf{R}}}\right)\right. \\
&\quad \left. + (q-1)p_1^{inv}\left(\log \frac{p_1^{inv}}{s_{\mathbf{A+R}}^{inv}} + \log \frac{p_1^{inv}}{s_{\mathbf{R}}^{inv}}\right)\right] + (1-s) \\
&\quad \cdot \left[p_2\left(\log \frac{p_2}{s_{\mathbf{A+R}}^{inv}} + \log \frac{p_2}{s_{\mathbf{R}}}\right) + p_3\left(\log \frac{p_3}{s_{\mathbf{A+R}}} + \log \frac{p_3}{s_{\mathbf{R}}^{inv}}\right)\right. \\
&\quad \left. + (q-2)p_{2,3}^{inv}\left(\log \frac{p_{2,3}^{inv}}{s_{\mathbf{A+R}}^{inv}} + \log \frac{p_{2,3}^{inv}}{s_{\mathbf{R}}^{inv}}\right)\right],
\end{aligned}$$

for $s_{\mathbf{R}}^{inv} \triangleq (1 - s_{\mathbf{R}})/(q - 1)$, $s_{\mathbf{A+R}}^{inv} \triangleq (1 - s_{\mathbf{A+R}})/(q - 1)$ with the following constraints

$$\begin{aligned}
c_1(p_1, p_1^{inv}) &\triangleq p_1 + (q-1)p_1^{inv} - 1 &&= 0, &&(6) \\
c_2(p_2, p_3, p_{2,3}^{inv}) &\triangleq p_2 + p_3 + (q-2)p_{2,3}^{inv} - 1 &&= 0, &&(7) \\
c_3(p_1, p_2) &\triangleq p_1 s + p_2(1-s) - s_{\mathbf{R}} &&= 0, &&(8) \\
c_4(p_1, p_3) &\triangleq p_1 s + p_3(1-s) - s_{\mathbf{A+R}} &&= 0. &&(9)
\end{aligned}$$

Proof: The proof follows by using the definition of KL-divergence and is omitted for brevity. $\square$

For analytical tractability, we further simplify the problem by fixing a desired average sparsity level $s_{avg} \triangleq \frac{s_{\mathbf{R}} + s_{\mathbf{A+R}}}{2}$ instead of fixing the values of $s_{\mathbf{R}}$ and $s_{\mathbf{A+R}}$ separately. We show in Lemma 7 (Appendix C) that for a fixed s_{avg}, the minimum total leakage is obtained for $s_{\mathbf{R}} = s_{\mathbf{A+R}}$. Therefore, in the sequel, we will consider minimizing the total leakage for a desired average sparsity s_{avg}, where the sparsity levels of the matrices are $s_{\mathbf{R}} = s_{\mathbf{A+R}} = s_{avg}$.

B. Optimal Sparse One-Time Pad

We show in Theorem 1 that the optimization problem in Lemma 2 is a convex optimization problem that admits only one solution. Hence, the result is twofold: *(i)* given a desired s_{avg}, the minimum leakage possible is characterized; and *(ii)* by constructing the matrix $\mathbf{R}$ as in Construction 1 with parameters $p_1^\star$, $p_2^\star$ and $p_3^\star$ as in Theorem 1, the desired sparsity and the minimum leakage possible are attained.

Theorem 1: Given a desired sparsity level s_{avg}, the optimal PMF of the form given in Eqs. (2) and (3) that minimizes the leakage is obtained by picking $p_1 = p_1^\star$ as the root of a polynomial $c_3 p_1^3 + c_2 p_1^2 + c_1 p_1 + c_0$ that satisfies $\max\{2s_{avg} - 1 + s, 0\} \leq 2p_1 s \leq 2\min\{s, s_{avg}\}$. The coefficients of the polynomial are

$$\begin{aligned}
c_3 &\triangleq s^2(4 + \bar{q}), \\
c_2 &\triangleq 4s(1 - s - 2s_{avg}) - \bar{q}s(2s_{avg} + s), \\
c_1 &\triangleq (1 - s - 2s_{avg})^2 + \bar{q}(2ss_{avg} + s_{avg}^2),
\end{aligned}$$

$$\mathsf{c}_0 \triangleq -\bar{q}s_{avg}^2,$$

where $\bar{q} \triangleq (q-2)^2/(q-1)$. The parameters p_1^{inv}, $p_2^\star$, $p_3^\star$ can be obtained by using $p_1^\star$ in Eqs. (6) to (9).

Proof: The proof is shown in Appendix B. □

Note that a unique solution exists for $p_1^\star$, $p_2^\star$, $p_3^\star$ since we minimize a convex function over a convex feasibility set [47].

C. Semi-Perfect One-Time Pad

Lemma 1 closes the door on obtaining a perfectly secure (with zero leakage) one-time pad with sparse shares. More precisely, given a sparse matrix $\mathbf{A}$, if $\mathbf{R}$ is generated as in Construction 1, then $L_1 + L_2 = 0$ occurs if and only if $s_{\mathbf{R}} = s_{\mathbf{A}+\mathbf{R}} = q^{-1}$. If the field size q is very large, then the shares cannot be considered sparse. However, Lemma 1 does not rule out the case where $L_1 + L_2 > 0$ and $L_1 \cdot L_2 = 0$, i.e., one of the shares achieves a zero leakage. However, constructing a semi-perfect one-time pad scheme can be useful in distributed matrix multiplications where part of the workers are fully untrusted. In contrast, the other part can be partially trusted, cf. Section VII-B. We show that it is possible to construct shares $\mathbf{R}$ and $\mathbf{A}+\mathbf{R}$ where the latter achieves perfect privacy, allowing for a sparse semi-perfect one-time pad. This can be achieved using Construction 1 with $p_1 = p_3 = p$ and $p_2 = \frac{1-p}{q-1}$, for any $0 < p < 1$ as shown in Lemma 3.

Lemma 3: Given a private sparse matrix $\mathbf{A}$ with sparsity level $s_{\mathbf{A}} = s > q^{-1}$ and if a padding matrix $\mathbf{R}$ is generated dependently on $\mathbf{A}$ as in Construction 1 for $p_1 = p_3 = p$ and $p_2 = \frac{1-p}{q-1}$ for $0 < p < 1$, then the sparsity levels of the padding and the padded matrix $s_{\mathbf{R}}$ and $s_{\mathbf{A}+\mathbf{R}}$ and the leakage L_1 increase with p, while no private information about $\mathbf{A}$ is leaked through the padded matrix, i.e., $L_2 = 0$.

Proof: The sparsity levels of the shares are computed by plugging $p_1 = p_3 = p$ and $p_2 = \frac{1-p}{q-1}$ into Eqs. (2) and (3) to obtain

$$s_{\mathbf{R}} = p\frac{(sq-1)}{q-1} + \frac{(1-s)}{q-1}, \quad \text{and} \quad s_{\mathbf{A}+\mathbf{R}} = p.$$

Clearly, $s_{\mathbf{A}+\mathbf{R}}$ increases with p and $s_{\mathbf{R}}$ increases with p for the case where $s > q^{-1}$.

From [48, Theorem 2.7.4], we conclude that the leakage $L_1 = I_q(\mathbb{R}_{\{i,j\}}; \mathbb{A}_{\{i,j\}})$ is a convex function in p since the PMF of the input matrix is fixed and the conditional PMF of $\mathbb{R}_{\{i,j\}}|\mathbb{A}_{\{i,j\}}$ can be written as a convex mixture of two conditional distributions as follows

$$\Pr_{\mathbb{R}_{\{i,j\}}|\mathbb{A}_{\{i,j\}}}(r|a) = p\mathbb{1}_{r=-a} + (1-p)\frac{1}{q-1}\mathbb{1}_{r\neq-a},$$

$\forall r, q \in \mathbb{F}_q$, where $\mathbb{1}_{condition}$ is the indicator function that returns 1 if the condition is true and 0 otherwise (in this case the indicator functions are conditional distributions).

The leakage $L_2 = I_q(\mathbb{A}_{\{i,j\}} + \mathbb{R}_{\{i,j\}}; \mathbb{A}_{\{i,j\}})$ is shown to be equal to zero by writing down the definition of mutual information and noting that the entropy is independent of the alphabet and only depends on the PMF. □

Lemma 3 introduces a trade-off between the sparsity levels of the shares $s_{\mathbf{A}}$, $s_{\mathbf{A}+\mathbf{R}}$ and the leakage L_1, which we will optimize in the context of sparse and private matrix multiplication schemes that can tolerate stragglers (Section VII-B).

VI. SPARSE SECRET SHARING

In this section, we construct sparse secret sharing schemes for parameters $n \geq 3$, $t = 2$ and $z = t - 1 = 1$, i.e., a generalization of the well-known Shamir secret sharing schemes [49] for $t = 2$. Given an input matrix $\mathbf{A}$, the challenge is to design a random matrix $\mathbf{R}$ such that n evaluations of the encoding polynomial

$$f(x) = \mathbf{A} + x\mathbf{R},$$

give n shares that satisfy the desired sparsity and privacy guarantees. Our approach is to design the matrix $\mathbf{R}$ based on the entries of $\mathbf{A}$ and the evaluation points $\alpha_1, \ldots, \alpha_n$ used to generate the shares as shown next.

Construction 2 (Conditional Distribution of the Random Matrix for Sparse Polynomial Encoding): Given a set of n distinct non-zero evaluation points α_i for $i \in [n]$, the entries of the matrix $\mathbf{R}$ are drawn according to the following conditional distribution.

$$\Pr_{R_{\{i,j\}}|\mathbb{A}_{\{i,j\}}}(r|0) = \begin{cases} p_1, & r = 0 \\ \dfrac{1-p_1}{q-1}, & r \neq 0, \end{cases} \quad (10)$$

$$\Pr_{R_{\{i,j\}}|\mathbb{A}_{\{i,j\}}}(r|a) = \begin{cases} p_s, & r \in \left\{-\dfrac{a}{\alpha_i}\right\}_{i \in [n]} \\ \dfrac{1-np_s}{q-n}, & r \notin \left\{-\dfrac{a}{\alpha_i}\right\}_{i \in [n]}, \end{cases} \quad (11)$$

where $r \in \mathbb{F}_q$, $a \in \mathbb{F}_q^*$, n is the number of shares and $\frac{1}{\alpha_i}$ is the multiplicative inverse of α_i in $\mathbb{F}_q$.

Similarly to Construction 1, the intuition behind Construction 2, is to pick probabilities that contribute to having zero-valued entries in the shares while reducing the dependence between the shares and $\mathbf{A}$ to reduce the leakage. The parameter p_s tailors the probability of artificially having zero-valued entries in a single share, while p_1 controls the probability of inheriting a 0 in every share. Construction 2, is restricted to generating n shares that have the same sparsity levels. The motivation of this restriction is that for $n = 2$, the total leakage is minimized when all shares have the same sparsity, cf. Lemma 7. Therefore, we conjecture that this holds for $n \geq 3$, i.e., if more than 2 shares are generated per input, then the optimal leakage is obtained when all shares have the same sparsity levels.

Lemma 4 quantifies the sparsity level of shares constructed as in Construction 2, Lemma 5 presents the optimization problem and Theorem 2 provides the values of p_1 and p_s that minimize the total leakage.

Lemma 4: Following Construction 2, the shares have the following level of sparsity:

$$s_d = p_1 s_{\mathbf{A}} + p_s(1 - s_{\mathbf{A}}) \leq \frac{1 + ns_{\mathbf{A}} - s_{\mathbf{A}}}{n}.$$

Proof: The proof follows by calculating the PMF of the shares and applying the definition of sparsity. The bound is because (10) and (11) must form a valid PMF, and hence, $p_s \leq \frac{1}{n}$. □

We define $z(x, y) \triangleq x \log(\frac{x}{y})$ for ease of notation in the remainder of the paper.

Lemma 5: Let the entries of the private matrix $\mathbf{A}$ be distributed as in Eq. (1). When considering shares of the form given in Construction 2 with fixed desired sparsity level s_d, the optimal element-wise total leakage is given by $\min_{p_1, p_s} \mathrm{L}(p_1, p_1^{inv}, p_s, p_s^{inv})$, where

$$
\begin{aligned}
&\mathrm{L}(p_1, p_1^{inv}, p_s, p_s^{inv}) \\
&= s\big[z(p_1, s_d) + (q-1)z(p_1^{inv}, s_d^{inv})\big] + (1-s) \\
&\quad \times \big[z(p_s, s_d) + (n-1)z(p_s, s_d^{inv}) + (q-n)z(p_s^{inv}, s_d^{inv})\big].
\end{aligned}
\tag{12}
$$

Thereby, $s_d^{inv} = \frac{1 - s_d}{q - 1}$ is the likelihood of a share's entry being non-zero. The constraints of the optimization problem are

$$
c_1(p_1, p_1^{inv}) \triangleq p_1 + (q-1)p_1^{inv} - 1 = 0, \tag{13}
$$
$$
c_2(p_s, p_s^{inv}) \triangleq np_s + (q-n)p_s^{inv} - 1 = 0, \tag{14}
$$
$$
c_3(p_1, p_s) \triangleq p_1 s + p_s(1-s) - s_d = 0. \tag{15}
$$

Further, p_1, p_1^{inv}, p_s and $p_s^{inv} \triangleq (1 - np_s)/(q - n)$ are bounded between zero and 1.

Proof: For each share's fixed desired sparsity level s_d, minimizing the total leakage for *any sparse secret sharing scheme* amounts to solving the following optimization problem.

$$
\begin{aligned}
\mathrm{L}_{\mathrm{opt}} &= \min_{\mathcal{P}} \sum_{i \in [n]} \mathrm{I}_q\big(\mathtt{A}_{\{i,j\}} + \alpha_i \mathtt{R}_{\{i,j\}}; \mathtt{A}_{\{i,j\}}\big) \\
&= \min_{\mathcal{P}} \sum_{i \in [n]} \sum_{a,y \in \mathbb{F}_q} \mathrm{P}_{\mathtt{A}}(a)\left(p_{(y-a)\alpha_i^{-1}|a} \log \frac{p_{(y-a)\alpha_i^{-1}|a}}{\mathrm{P}_{\mathtt{A}+\alpha_i \mathtt{R}}(y)}\right).
\end{aligned}
\tag{16}
$$

Again, this is a constrained optimization problem with the following constraints

$$
\forall b \in \mathbb{F}_q : p_{0|b} + \sum_{x \in \mathbb{F}_q^\star} p_{x|b} - 1 = 0,
$$

$$
\forall i \in [n] : p_{0|0} \cdot \mathrm{P}_{\mathtt{A}}(0) + \sum_{b \in \mathbb{F}_q^\star} p_{-b\alpha_j^{-1}|b} \cdot \mathrm{P}_{\mathtt{A}}(b) - s_d = 0.
$$

When restricting our attention to shares of the form given in Construction 2, the optimization problem transforms into the one given in the statement of the Lemma. □

The solution to the optimization problem in Lemma 5 is given in the following Theorem 2.

Theorem 2: Given a desired sparsity level s_d, the value $p_s^\star$ that minimizes the leakage of the shares as in Construction 2 is the real root of the polynomial $\sum_{j=0}^{n+1} \mathsf{b}_j p_s^j$ with coefficients

$$
\mathsf{b}_{n+1} = -1 - (-n)^n \tilde{q}
$$
$$
\mathsf{b}_n = \big(\tilde{s}(-n)^n - n(-n)^{n-1}\big)\tilde{q} - \bar{s}
$$
$$
\mathsf{b}_k = \left(\tilde{s}\binom{n}{k}(-n)^k - \binom{n}{k-1}(-n)^{k-1}\right)\tilde{q}, \forall k \in [n-1]
$$
$$
\mathsf{b}_0 = \tilde{q}\tilde{s},
$$

where $\tilde{s} = s_d/(1-s)$, $\bar{s} = (s - s_d)/(1-s)$ and $\tilde{q} = (q-1)/(q-n)^n$. The root $p_s^\star$ must satisfy $0 \le p_s^\star(1-s) \le \min\left\{s_d, \frac{1}{n}\right\}$ and $p_1^\star$ can be computed as
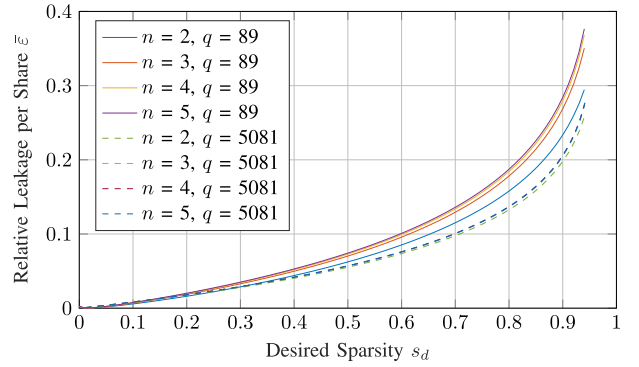
$$
p_1^\star = \frac{s_d - p_s^\star(1-s)}{s}.
$$

Fig. 1. The relative leakage of $g_i(\alpha_j)$ versus its desired sparsity for different number of shares n, $s = 0.95$ and $q \in \{89, 5081\}$.

As for Theorem 1, a unique solution exists for $p_1^\star, p_s^\star$ due to the convex properties of the optimization.

Proof: The proof is provided in Appendix D. □

Fig. 1 depicts the relative leakage $\bar{\varepsilon}$ per share as a function of s_d and the number of shares n. We consider the finite field $\mathbb{F}_q$ for $q = 89$ and $q = 5081$. As n grows,[6] each share leaks more about the private input matrix for a fixed s_d. For large values of q, the loss of privacy incurred by increasing n becomes negligible. In addition, for fixed n and fixed s_d, increasing q decreases the relative leakage per share $\bar{\varepsilon}$. For example, for $q = 89$ and $s_d = 0.9$, $\bar{\varepsilon}$ is equal to 0.234 for $n = 2$ and increases to 0.284 when increasing n to 5. However, for $q = 5081$ and $s_d = 0.9$, $\bar{\varepsilon}$ increases from 0.199 to 0.207 when increasing n from 2 to 5. Notice that for fixed n and s_d, increasing q decreases $\bar{\varepsilon}$, cf. Fig. 1.

We show next how to apply the designed sparse secret sharing schemes to distributed matrix multiplication.

VII. SPARSE AND PRIVATE MATRIX MULTIPLICATION SCHEMES

In this section, we focus on designing straggler tolerant sparse and private matrix multiplication schemes.

We construct schemes with parameters $(N, \sigma = N - 3, \varepsilon_1^\star, \varepsilon_2^\star, s_{d_1}, s_{d_2})$, see Definition 3, where $\varepsilon_1^\star$ and $\varepsilon_2^\star$ are the optimal leakages obtained for the desired sparsity levels s_{d_1} and s_{d_2}. We then tackle the setting in which the matrix $\mathbf{B}$ is public and the workers can be grouped into two clusters: fully non-trusted workers to whom no information about $\mathbf{A}$ should be leaked; and partly trusted workers to whom a small positive leakage about $\mathbf{A}$ can be tolerated. In this setting, we construct a scheme that satisfies the perfect privacy of $\mathbf{A}$ for one cluster and minimizes the leakage for the other. The constructed scheme tolerates a desired number of full and partial stragglers.

A. General Setting

The scheme follows from an immediate application of the scheme constructed in Section VI. The input matrices are encoded into shares using our sparse polynomial encoding introduced in Construction 2 with the parameters derived in

---

[6]This statement holds for $n \ll q$. If n approaches q, the constraints imposed by the optimization problem in Eq. (16) lead to non-sparse secret sharing.
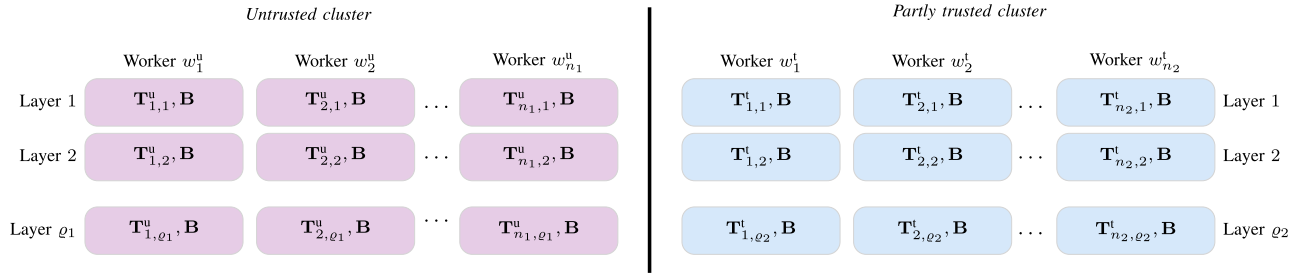
Fig. 2. The two non-communicating clusters, namely the untrusted and partly trusted, are illustrated on the left and right-hand side, respectively. The workers of the untrusted cluster $w_1^u, w_2^u, \ldots, w_{n_1}^u$ get each ϱ_1 tasks (created as in Section VII-B2)) that leak nothing about the input matrix. On the other hand, each worker of the partly trusted cluster $w_1^t, w_2^t, \ldots, w_{n_2}^t$ gets ϱ_2 tasks that leak some information about the input matrix. Every worker has to multiply the designated matrices with the public matrix $\mathbf{B}$ and then send each computation back to the main node.

Theorem 2. Namely, the main node picks a set of distinct non-zero evaluation points $\{\alpha_j\}_{j=1}^N$ and creates a polynomial pair $f(x), g(x)$ as

$$f(x) = \mathbf{A} + x\mathbf{R} \quad \text{and} \quad g(x) = \mathbf{B} + x\mathbf{S}, \quad (17)$$

where $\mathbf{R}$ and $\mathbf{S}$ are generated depending on $\mathbf{A}$ and $\mathbf{B}$, respectively. Each worker j is then assigned $f(\alpha_j)$ and $g(\alpha_j)$ and its task is to return the multiplication $f(\alpha_j) \cdot g(\alpha_j)$ to the main node. Define the degree-2 polynomial $h(x)$ as

$$h(x) = f(x) \cdot g(x) = \mathbf{AB} + x(\mathbf{RB} + \mathbf{AS}) + x^2\mathbf{RS}. \quad (18)$$

Then, the main node obtains the desired computation $\mathbf{C} = \mathbf{AB} = h(0)$ from the responses of any three workers. This task assignment strategy allows for *full* straggler tolerance, i.e., $\mathcal{C}_j = \emptyset$ if worker j is a straggler or $\mathcal{C}_j = \{h(\alpha_j)\}$, otherwise.

a) Reduction of computation load per worker: Each worker gets assigned a computation as large as the multiplication $\mathbf{AB}$. To reduce the computation load in a system of N workers and simultaneously tolerate at least σ stragglers, the input matrices $\mathbf{A}$ and $\mathbf{B}$ can be divided into m parts each such that $\sigma < m$, $m|N$ and $\frac{N}{m} = 3 + \sigma$. The division of the matrices is column-wise for $\mathbf{A}$ and row-wise for $\mathbf{B}$ so that the multiplication $\mathbf{C} = \mathbf{AB}$ is the sum of $\mathbf{A}_i\mathbf{B}_i$ for $i = 1, \ldots, m$. The main node creates m polynomial pairs $(f_i(x), g_i(x))$ to encode the $\mathbf{A}_i, \mathbf{B}_i$'s and assigns $\sigma + 3$ distinct evaluations of each polynomial pair $i \in [m]$ to a group of $\sigma + 3$ workers indexed by $\mathcal{Z}_i, i \in [m]$, where $|\mathcal{Z}_i| = \sigma + 3$. That is, worker j gets $m(\sigma+3)/N$ pairs of sparse sub-matrices $\{(f_i(\alpha_j), g_i(\alpha_j))\}_{i \in m : j \in \mathcal{Z}_i}$. Consequently, each worker computes a $(\sigma + 3)/N$-fraction of the entire multiplication. The groups $\mathcal{Z}_i, i \in [m]$ can, for example, be created by a cyclic repetition scheme similar to [36], [50], [51], which ensures that 3 evaluations of each polynomial $h_i(x)$ are guaranteed to be returned despite having any σ full stragglers. In the best case, σm stragglers can be tolerated, i.e., σ workers from each group. The leakage is smaller as the padding matrices used for each encoding are independent and each worker observes a smaller matrix.

b) Computation cost: Our sparse and private scheme introduced above sacrifices efficient polynomial design for matrix multiplication as considered, e.g., in [15], [52] and their privacy-preserving extensions [17], [53]. We compare the computation cost of our scheme to secure MatDot codes [53] for different numbers of workers and varying straggler tolerance

depending on the sparsity s_d of the shares. Therefore, assume that $\mathbf{A}$ is split into m blocks $\mathbf{A}_1, \ldots \mathbf{A}_m$, each of size $\mathbb{F}_q^{\mathsf{m} \times \frac{\mathsf{n}}{m}}$, and $\mathbf{B}$ split into $\mathbf{B}_1, \ldots, \mathbf{B}_m \in \mathbb{F}_q^{\frac{\mathsf{n}}{m} \times \ell}$.

Lemma 6: With N workers and at most σ stragglers, our scheme outperforms secure MatDot codes [15] with $z = 1$ for a sparsity value s_d of the shares fulfilling

$$s_d \geq 1 - \sqrt{\frac{2N}{(\sigma + 3)(N - \sigma - 1)}}.$$

Proof: Using a secure MatDot code with $z = 1$, we require at least $2m + 1$ evaluations to interpolate the multiplication of the two polynomials. Given a straggler tolerance of σ, we require that $N - \sigma \geq 2m + 1$, and hence $m \leq (N - \sigma - 1)/2$. Consequently, each worker computes at least $\mathsf{m} \times \frac{2\mathsf{n}}{N-\sigma-1} \times \ell$ multiplications. Using our scheme with a straggler tolerance of σ, the task can be divided into at most $m \leq \frac{N}{\sigma+3}$ polynomials to reduce the compute complexity per worker. Considering a probabilistic output sparsity guarantee s_d of the shares, then each worker computes on average at least $\mathsf{m}(1 - s_d)^2 \times \frac{\mathsf{n}(\sigma+3)}{N} \times \ell$. Neglecting the requirements that $m|\mathsf{n}$ and $\sigma + 3|N$, we can find that for sparsity levels given in the statement of the lemma our sparsity-preserving strategy is more computationally efficient. $\square$

We plot the break-even points for different values of N and σ in Fig. 3. To gain over the naive construction ignoring sparsity, the most sparsity is required for cases in which the number of tolerated stragglers is roughly in the middle between 0 and N. When σ is small, the negative impact of providing straggler tolerance by repetition through our scheme is limited. When σ is large, even polynomial constructions like MatDot cannot split the matrices into small chunks. Hence, the overhead of our scheme is limited, thereby facilitating the positive effect of preserving the sparsity. In secure polynomial codes [17], the matrices are split into outer products, i.e., $\mathbf{A}$ is split into m blocks $\mathbf{A}_1, \ldots \mathbf{A}_m$, each of size $\mathbb{F}_q^{\frac{\mathsf{m}}{m} \times \mathsf{n}}$, and $\mathbf{B}$ split into $\mathbf{B}_1, \ldots, \mathbf{B}_m \in \mathbb{F}_q^{\mathsf{n} \times \frac{\ell}{m}}$, which represents the other extreme compared to MatDot codes [15]. The break-even point for sparsity is $s_d \geq 1 - \sqrt{\frac{N}{(\sigma+3)(\sqrt{N-\sigma}-1)^2}}$, hence slightly worse than for MatDot codes. PolyDot codes, also proposed in [15], interpolate between those results.
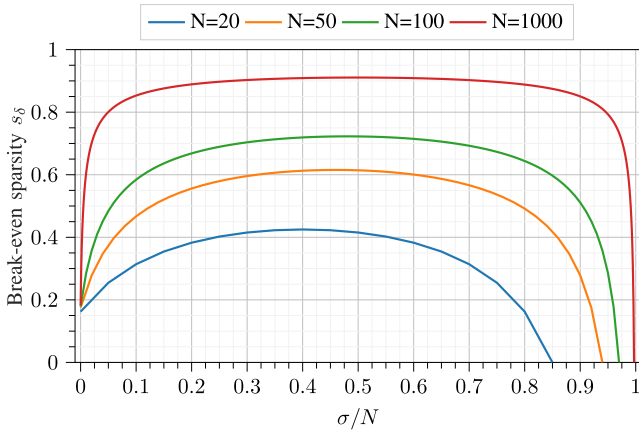
Fig. 3. Sparsity values above which our scheme is beneficial over polynomial codes in terms of compute complexity for different values of N and σ.

B. Setting of Clustered Workers

In this setting, we assume that the workers can be grouped into two non-communicating clusters with the properties explained next. We introduce a one-sided private and sparse matrix multiplication scheme, where the matrix $\mathbf{B}$ is publicly available. The main advantage of such schemes is tolerating collusions, i.e., allowing $z > 1$.

1) The Clustering Model: The N workers, where $N = n_1 + n_2$, are divided into two clusters of workers with the following properties:

- **Untrusted cluster:** The cluster consists of n_1 workers, w_i^{u}, for $i = 1, \ldots, n_1$, which are fully untrusted and can all collude. No information leakage about $\mathbf{A}$ should be tolerated, i.e., perfect privacy is required.
- **Partly trusted cluster:** This cluster is composed of n_2 workers w_i^{t}, for $i = 1, \ldots, n_2$, that are partly trusted by the main node. A small amount of information about $\mathbf{A}$ can be leaked to any z, $1 \le z < n_2$, colluding workers.

The grouping of workers into two *non-communicating* clusters is practically relevant since the untrusted cluster can represent an externally hired computational node pool offered by a single provider. In contrast, the partly trusted cluster might represent a locally available computing pool, that can be partly trusted. We combine the sparse one-time pad of Lemma 3 with the cyclic shift task assignment [36], [51] to obtain the desired privacy, sparsity, and straggler tolerance.

2) Task Distribution and Straggler Mitigation: Given the matrix $\mathbf{A} \in \mathbb{F}_q^{m \times n}$, the main node generates a padding matrix $\mathbf{R} \in \mathbb{F}_q^{m \times n}$ following Construction 1 and setting $p_1 = p_3 \triangleq p$ and $p_2 = \frac{1-p}{q-1}$. The main node splits the padded matrix $\mathbf{A} + \mathbf{R}$ row-wise into n_1 sub-matrices and splits the padding matrix $\mathbf{R}$ into n_2 sub-matrices as follows

$$\mathbf{A} + \mathbf{R} = \begin{pmatrix} (\mathbf{A} + \mathbf{R})_1 \\ (\mathbf{A} + \mathbf{R})_2 \\ \vdots \\ (\mathbf{A} + \mathbf{R})_{n_1} \end{pmatrix} \quad \text{and} \quad \mathbf{R} = \begin{pmatrix} \mathbf{R}_1 \\ \mathbf{R}_2 \\ \vdots \\ \mathbf{R}_{n_2} \end{pmatrix},$$

where $(\mathbf{A} + \mathbf{R})_i \in \mathbb{F}_q^{\frac{m}{n_1} \times n}$ for $i \in [n_1]$ and $\mathbf{R}_i \in \mathbb{F}_q^{\frac{n}{n_2} \times \ell}$ for $i \in [n_2]$. Then, the main node sends to every worker w_i^{u} of

the untrusted cluster a sub-matrix $\mathbf{T}_{i,1}^{\mathrm{u}} \triangleq (\mathbf{A} + \mathbf{R})_i$ for $i \in [n_1]$ and to every worker w_a^{t} of the partly trusted cluster the sub-matrices $\mathbf{T}_{a,1}^{\mathrm{t}} \triangleq \mathbf{R}_a$ for $a \in [n_2]$. The collection of the matrices $\mathbf{T}_{i,1}^{\mathrm{u}}$ and $\mathbf{T}_{a,1}^{\mathrm{t}}$ is called the first *layer* of computation tasks. The workers compute $\mathbf{T}_{i,1}\mathbf{B}$ and return the result to the main node.

Assigning one layer only is not resilient to stragglers. To tolerate $\varrho_1 - 1$ stragglers in the untrusted cluster and $\varrho_2 - 1$ in the partly cluster, the main node assigns more computation layers to the workers following a cyclic shifting strategy:

$$\mathbf{T}_{(i \mod n_1)+1, j}^{\mathrm{u}} = \mathbf{T}_{i, j-1}^{\mathrm{u}}, \quad i \in [n_1], j \in [\varrho_1] \setminus \{1\},$$
$$\mathbf{T}_{(a \mod n_2)+1, b}^{\mathrm{t}} = \mathbf{T}_{a, b-1}^{\mathrm{t}}, \quad a \in [n_2], b \in [\varrho_2] \setminus \{1\}.$$

The system is illustrated in Fig. 2. Every worker of the untrusted cluster w_i^{u}, $i \in [n_1]$, sequentially multiplies and sends $\mathbf{T}_{i,j}^{\mathrm{u}}\mathbf{B}$ to the main node for $j = 1, \ldots, \varrho_1$. The same applies to the workers of the partly trusted cluster.

The privacy, sparsity, and straggler tolerance guarantees of this task assignment strategy are stated in Theorem 3.

Theorem 3: The cyclic shift assignment strategy explained above provides the following guarantees:

- Privacy guarantees*: In the* untrusted cluster, *perfect privacy of* $\mathbf{A}$ *is guaranteed even if all the workers collude, i.e.,* $I_q(\{\mathcal{T}_{i,j}^u\}_{i \in [n_1], j \in [\varrho_1]}; A) = 0$. *In the* partly trusted cluster, *if up to* z *workers collude, then with* $L_1 = I_q(R_{\{i,j\}}; A_{\{i,j\}})$, *the leakage about* $\mathbf{A}$ *is given by*

$$I_q(\{\mathcal{T}_{i,j}^t\}_{i \in \mathcal{Z}, j \in [\varrho_2]}; A) = \min\left\{\frac{\varrho_2 z}{n_2}, 1\right\} \cdot m \cdot n \cdot L_1. \tag{19}$$

- Sparsity: *For a desired maximum leakage* ε, *i.e.,* $I_q(\{\mathcal{T}_{i,j}^t\}_{i \in \mathcal{Z}, j \in [\varrho_2]}; A) \le \varepsilon$, *the maximum sparsity levels allowed for the matrices* $\mathbf{A} + \mathbf{R}$ *and* $\mathbf{R}$ *are given by*

$$s_{\mathbf{A}+\mathbf{R}} = p^\star, \quad \text{and} \quad s_{\mathbf{R}} = p^\star \frac{sq-1}{q-1} + \frac{1-s}{q-1},$$

where

$$p^\star = \max_{I_q(\{\mathcal{T}_{i,j}^t\}_{i \in \mathcal{Z}, j \in [\varrho_2]}; A) \le \varepsilon} p. \tag{20}$$

- Straggler tolerance: *The main node is able to reconstruct the desired computation after receiving any* $K^u \triangleq \frac{-\varrho_1^2 + \varrho_1(2n_1-1)}{2} + 1$ *and any* $K^t \triangleq \frac{-\varrho_2^2 + \varrho_2(2n_2-1)}{2} + 1$ *responses from the workers of the untrusted and partly trusted clusters, respectively. In other words, partial stragglers and up to* $\varrho_1 - 1$ *and* $\varrho_2 - 1$ *stragglers can be tolerated from the respective clusters.*

Proof: The proof by construction is omitted for brevity. □

We plot in Fig. 4 the value of p^*, representing the sparsity of the shares, versus z, the number of colluding workers of the partly trusted cluster for a varying choice of maximum relative leakage $\bar{\varepsilon}$. As reflected from the theory, decreasing $\bar{\varepsilon}$ decreases p^*, meaning that the sparsity levels $s_{\mathbf{R}}, s_{\mathbf{A}+\mathbf{R}}$ should have lower values to guarantee higher privacy constraints. In addition, for a fixed relative leakage $\bar{\varepsilon}$, as z increases,
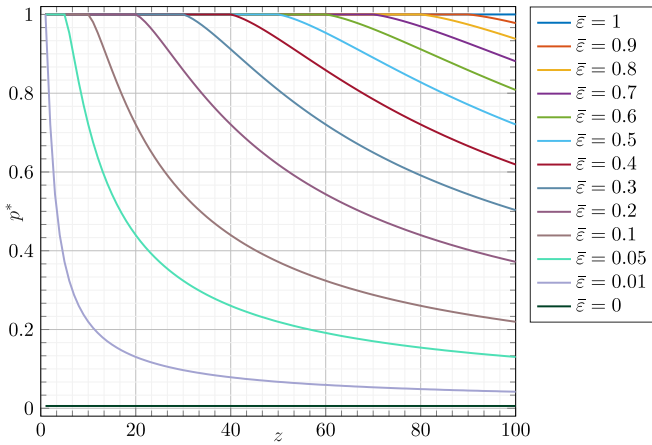
Fig. 4. The value of $p^\star$ as a function of z, the number of colluding workers of the partly trusted cluster, for different privacy constraints reflected by choice of $\bar{\varepsilon}$. We take $n_2 = 100$, $\varrho_2 = 1$, $s_\mathbf{A} = 0.93$ and consider the extension finite field $\mathbb{F}_{256}$. The value of $p^\star$ controls the sparsity of the shares, i.e., $s_{\mathbf{A}+\mathbf{R}} = p^\star$ and $s_\mathbf{R} \approx 0.93 p^*$, as in Eq. (20). The case $\bar{\varepsilon} = 0$ corresponds to perfect privacy. It requires $p^\star = q^{-1}$ for all $z \in [n_2]$, whereas $\bar{\varepsilon} = 1$ corresponds to the non-private case, hence allowing the choice $p^\star = 1$ for all $z \in [n_2]$. For $0 < \bar{\varepsilon} < 1$, the increase of z and/or the decrease of $\bar{\varepsilon}$ reduces the achievable $s_{\mathbf{A}+\mathbf{R}}$ and $s_\mathbf{R}$.

$p^\star$ (and the allowed sparsity $s_\mathbf{R}, s_{\mathbf{A}+\mathbf{R}}$) decreases since more workers collude and can have access to a bigger portion of resemblance to the private matrix $\mathbf{A}$, which should be made more private.

VIII. MATRICES WITH CORRELATED ENTRIES

In this section, we discuss the use of shuffling to break the potential correlation between the entries of sparse matrices and mention further research directions.

In certain applications, e.g., the matrix entries might be correlated when the sparse matrices represent a face [54]. However, our theory and the proposed schemes assume the entries of $\mathbf{A}$ (and $\mathbf{B}$) to be independent (see Definition 4). Sparsity-preserving pre-processing techniques can be applied to break the correlation between the entries of sparse matrices. In this section, we discuss the effect of applying random permutations to entries of $\mathbf{A}$ and $\mathbf{B}$. We will focus on cases where $\mathbf{A}$ and $\mathbf{B}$ must remain private. The ideas generalize to cases in which $\mathbf{B}$ is public.

Shuffling a matrix is equivalent to multiplying it with permutation matrices from the left and the right. Those permutation matrices are private to the main node and are unknown to the workers. Considering input matrices $\mathbf{A} \in \mathbb{F}_q^{m \times n}$ and $\mathbf{B} \in \mathbb{F}_q^{n \times \ell}$, the main node picks three permutation matrices $\mathbf{P}_1 \in \mathbb{F}_2^{m \times m}$, $\mathbf{P}_2 \in \mathbb{F}_2^{n \times n}$ and $\mathbf{P}_3 \in \mathbb{F}_2^{\ell \times \ell}$. Each column of $\mathbf{P}_i$ is generated by sampling a random unit vector without replacement from the respective field. The main node then conducts the following pre-processing operations

$$\mathbf{A}' = \mathbf{P}_1 \cdot \mathbf{A} \cdot \mathbf{P}_2 \text{ and } \mathbf{B}' = \mathbf{P}_2^\mathsf{T} \cdot \mathbf{B} \cdot \mathbf{P}_3.$$

Note that permutation matrices are orthogonal, i.e., the inverse equals their transpose. The matrices $\mathbf{A}', \mathbf{B}'$ can then be encoded as in Section VII-A.

The random permutation breaks most correlations between the entries of the input matrices. The colluding workers will
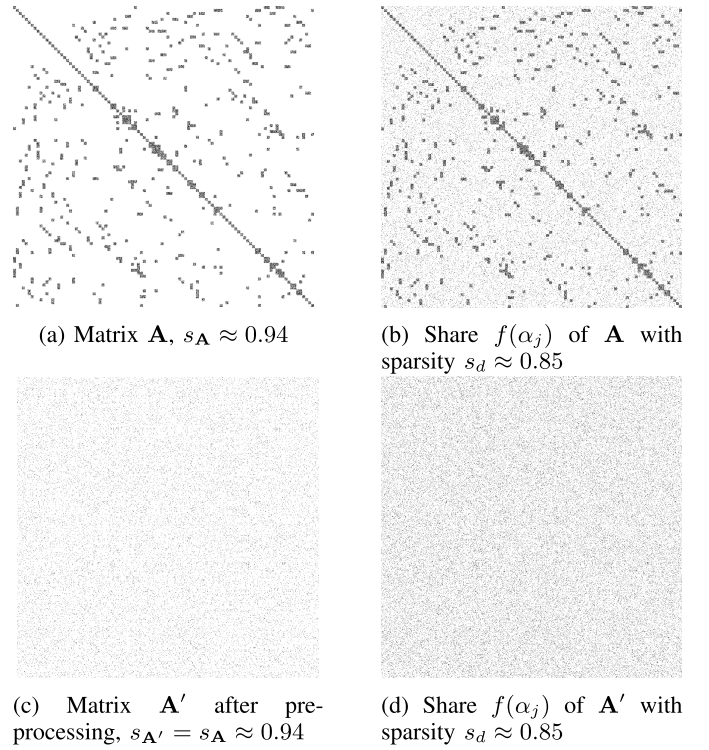

(a) Matrix $\mathbf{A}$, $s_\mathbf{A} \approx 0.94$

(b) Share $f(\alpha_j)$ of $\mathbf{A}$ with sparsity $s_d \approx 0.85$

(c) Matrix $\mathbf{A}'$ after pre-processing, $s_{\mathbf{A}'} = s_\mathbf{A} \approx 0.94$

(d) Share $f(\alpha_j)$ of $\mathbf{A}'$ with sparsity $s_d \approx 0.85$

Fig. 5. A depiction of the impact of random permutations on the correlated entries of $\mathbf{A}$. The matrix with correlated entries is shown in Fig. 5a. Naively applying our sparse secret sharing scheme randomizes the values of the entries of $\mathbf{A}$ but does not break the correlation between the entries of the shares, cf. Fig. 5b. The effect of applying the random permutation is shown in Figs. 5c and 5d where the correlation between the entries is graphically broken.

need to revert the permutation to leverage the statistical correlation between the entries of the private matrices. However, information-theoretic privacy guarantees cannot be given since the entries of $\mathbf{A}'$ and $\mathbf{B}'$ might still have some correlation.

In Fig. 5, we depict a random permutation of an input matrix $\mathbf{A}_i \in \mathbb{F}_{89}^{816 \times 816}$ with sparsity $s_\mathbf{A} \approx 0.94$ that follows a distribution similar to the one in Eq. (1), but with correlated entries. After pre-processing is applied, the correlations are graphically broken (see Fig. 5a and Fig. 5c). The process from Fig. 5a to Fig. 5b represents the encoding phase as in Eq. (17) without shuffling. The correlation between the entries of the shares is shown graphically and can be leveraged by the colluding workers after the input matrix is shuffled and encoded (Fig. 5c and Fig. 5d), then graphically, the correlation between the entries is broken.

This discussion shows that additional care must be taken when encoding matrices with correlated entries. Future research can be conducted to construct sparse secret-sharing schemes that break the correlation between the entries. Alternatively, privacy guarantees of sparsity-preserving pre-processing techniques can be analyzed.

IX. CONCLUSION

In this work, we opened the door to the coexistence of sparsity and privacy in secret sharing and distributed computing. We focused on the no-collusion regime, i.e., $z = 1$, and assumed that the entries of the private matrices are independent and identically distributed. We showed a fundamental tradeoff

between privacy and sparsity in the shares. Hence, insisting on perfect information-theoretic privacy does not allow for sparsity in the shares. We constructed sparse and private secret sharing schemes that take as input the desired sparsity and output shares with the provably lowest leakage possible. Using that, we constructed sparse and private matrix multiplication schemes for different settings of interest, some of which also include collusions. We discussed the correlation effect between the entries on the introduced methods. Generalizing our results for $z > 1$ remains an interesting open problem. Similarly, it is interesting to analyze the effect of lifting the assumption of independent entries and to construct schemes tailored for this setting.

APPENDIX

A. Proof of Lemma 1

Proof: We focus only on the element-wise leakage since A has i.i.d. entries. The following analysis holds for any entry $\mathrm{A}_{\{i,j\}}$ and $\mathrm{R}_{\{i,j\}}$ of $\mathbf{A}$ and $\mathbf{R}$. Hence, we abuse notation and drop the i, j index, i.e., we use A for $\mathrm{A}_{\{i,j\}}$ and R for $\mathrm{R}_{\{i,j\}}$. We do the analysis first for secret sharing schemes with $z = t = 1$ and generalize to secret sharing schemes with $z = 1$ and $k \geq 1$ afterwards. For $z = t = 1$, let $\mathrm{S}_i \triangleq \mathrm{A} + \alpha_i \mathrm{R}$ be the corresponding entry of share $i \in [N]$. It holds that

$$\Pr_{\mathrm{S}_i}(s) = \sum_{a=0}^{q-1} \Pr_{\mathrm{A}}(a) \Pr_{\mathrm{R}|\mathrm{A}}(\alpha_i^{-1}(s-a)|a), \forall s \in \mathbb{F}_q. \quad (21)$$

Then the entropy $\mathrm{H}_q(\mathrm{S}_i)$ can be written as [7]

$$
\begin{aligned}
\mathrm{H}_q(\mathrm{S}_i) &\overset{(a)}{=} -\sum_{s_j \in \mathbb{F}_q} \Pr_{\mathrm{S}_i}(s_j) \log_q \left(\Pr_{\mathrm{S}_i}(s_j) \right) \\
&\overset{(b)}{=} -\sum_{s_j \in \mathbb{F}_q} \left(\sum_{a \in \mathbb{F}_q} \Pr_{\mathrm{A}}(a) \Pr_{\mathrm{R}|\mathrm{A}}(\alpha_i^{-1}(s_j-a)|a) \right) \\
&\qquad \times \log_q \left(\frac{\sum_{a \in \mathbb{F}_q} \Pr_{\mathrm{A}}(a) \Pr_{\mathrm{R}|\mathrm{A}}(\alpha_i^{-1}(s_j-a)|a)}{\sum_{a \in \mathbb{F}_q} \Pr_{\mathrm{A}}(a)} \right) \\
&\overset{(c)}{\geq} -\sum_{s_j \in \mathbb{F}_q} \sum_{a \in \mathbb{F}_q} \Pr_{\mathrm{A}}(a) \Pr_{\mathrm{R}|\mathrm{A}}(\alpha_i^{-1}(s_j-a)|a) \\
&\qquad \times \log_q \left(\Pr_{\mathrm{R}|\mathrm{A}}(\alpha_i^{-1}(s_j-a)|a) \right) \\
&\overset{(d)}{=} \sum_{a \in \mathbb{F}_q} \Pr_{\mathrm{A}}(a) \mathrm{H}_q(\mathrm{R}|\mathrm{A}=a) \\
&= \mathrm{H}_q(\mathrm{R}|\mathrm{A}),
\end{aligned}
$$

where (a) is the definition of q−ary entropy, (b) is obtained by using Eq. (21) and dividing the argument of the logarithm function by $\sum_{t \in \mathbb{F}_q} \Pr_{\mathrm{A}}(t) = 1$, (c) follows from the log-sum inequality and (d) is obtained by grouping the summation terms in (c).

The log-sum inequality implies that (c) is satisfied with equality if and only if $\Pr_{\mathrm{R}|\mathrm{A}}(\alpha_i^{-1}(s_j-a)|a)$ is constant for every $a \in \mathbb{F}_q$, which has to hold for all $s_j \in \mathbb{F}_q$. That is, there

---

exist q disjoint sets (one for each $s_j \in \mathbb{F}_q$) $\left\{ \Pr_{\mathrm{R}|\mathrm{A}}(\alpha_i^{-1}(s_j - a)|a) \right\}_{a \in \mathbb{F}_q}$ whose entries must take the same value.

Given two shares S_1 and S_2 with distinct evaluation points α_1 and α_2, we have two of such sets. For a fixed $s_1 \in \mathbb{F}_q$ (in the equation for S_1) and for every $s_2 \in \mathbb{F}_q$, there exists a tuple $(s_1, s_2, \bar{a})$ with $\bar{a} \in \mathbb{F}_q$ such that $\alpha_1^{-1}(s_1 - \bar{a}) = \alpha_2^{-1}(s_2 - \bar{a})$. In particular, the value of $\bar{a}$ can be calculated as

$$
\begin{aligned}
\alpha_1^{-1}(s_1 - \bar{a}) &= \alpha_2^{-1}(s_2 - \bar{a}) \\
\underbrace{\alpha_1^{-1}\alpha_2}_{\triangleq c \neq 0} s_1 + \bar{a}(1 - \alpha_1^{-1}\alpha_2) &= s_2 \\
\bar{a} &= (s_2 - cs_1)/(1 - c).
\end{aligned}
$$

This particular choice $\bar{a}$ shows that

$$\left\{ \Pr_{\mathrm{R}|\mathrm{A}}(\alpha_1^{-1}(s_1 - a)|a) \right\}_{a \in \mathbb{F}_q} \cap \left\{ \Pr_{\mathrm{R}|\mathrm{A}}(\alpha_2^{-1}(s_2 - a)|a) \right\}_{a \in \mathbb{F}_q} \neq \emptyset.$$

Hence the elements of both sets must be equal, i.e.,

$$\Pr_{\mathrm{R}|\mathrm{A}}(\alpha_1^{-1}(s_1 - a)|a) = \Pr_{\mathrm{R}|\mathrm{A}}(\alpha_2^{-1}(s_2 - a)|a) \quad \forall a \in \mathbb{F}_q,$$

to fulfill (c) with equality. Moving forward, by fixing s_1 one can find such intersections for all $s_2 \in \mathbb{F}_q$. Since the sum over s_2 in (c) covers constraints on the whole space of possible probability values $\Pr_{\mathrm{R}|\mathrm{A}}(r|a)$, we can link the constraints of the q disjoint sets $\left\{ \Pr_{\mathrm{R}|\mathrm{A}}(\alpha_2^{-1}(s_2 - a)|a) \right\}_{a \in \mathbb{F}_q}$ for all s_2 through the set $\left\{ \Pr_{\mathrm{R}|\mathrm{A}}(\alpha_1^{-1}(s_1 - a)|a) \right\}_{a \in \mathbb{F}_q}$ for any fixed s_1. Consequently, all entries in the set $\left\{ \Pr_{\mathrm{R}|\mathrm{A}}(\alpha_1^{-1}(s - a)|a) \right\}_{a,s \in \mathbb{F}_q} = \left\{ \Pr_{\mathrm{R}|\mathrm{A}}(r|a) \right\}_{a,r \in \mathbb{F}_q}$ have to be equal to fulfill (c) with equality. The same steps can be repeated for more than two shares. We can further write the leakage as

$$
\begin{aligned}
\mathrm{I}_q(\mathrm{A}; \mathrm{S}_i) &= \mathrm{H}_q(\mathrm{A} + \alpha_i \mathrm{R}) - \mathrm{H}_q(\mathrm{A} + \alpha_i \mathrm{R}|\mathrm{A}) \\
&= \mathrm{H}_q(\mathrm{S}_i) - \mathrm{H}_q(\mathrm{R}|\mathrm{A}) \geq 0,
\end{aligned}
$$

where equality is only given if (c) is fulfilled with equality, and hence R is independent of A, i.e., $\Pr_{\mathrm{R}|\mathrm{A}}(r|a) = \Pr_{\mathrm{R}}(r)$, and is uniform.

The proof above can be generalized to schemes with $z = 1$ and $k \geq 1$, i.e., shares of the form $\mathrm{S}_i \triangleq \mathrm{A}_1 + \alpha_i \mathrm{A}_2 + \alpha_i^2 \mathrm{A}_3 + \cdots + \alpha_i^{k-1} \mathrm{A}_k + \alpha_i^k \mathrm{R}$, where A_i denotes the random variable describing the i-th split of matrix A. Let M_i be the random variable representing the linear combination of secrets of share i, i.e., $\mathrm{M}_i \triangleq \mathrm{A}_1 + \alpha_i \mathrm{A}_2 + \alpha_i^2 \mathrm{A}_3 + \cdots + \alpha_i^{k-1} \mathrm{A}_k$, and m_i its realization. By replacing, in (21), A with M_i, a by m_i, and α_i^{-1} by α_i^{-k}, we obtain for the entropy of each share the lower bound $\mathrm{H}_q(\mathrm{S}_i) \geq \mathrm{H}_q(\mathrm{R}|\mathrm{M}_i)$. The subsequent arguments proving that the inequality (c) only holds when R is uniformly distributed are still valid. Hence, it remains to prove that $\mathrm{I}_q(\{\mathrm{A}_i\}_{i \in k}; \mathrm{S}_i) = 0$ if and only if R is uniform and hence independent of $\{\mathrm{A}_i\}_{i \in k}$. We have

$$
\begin{aligned}
\mathrm{I}_q(\{\mathrm{A}_i\}_{i \in k}; \mathrm{S}_i) &= \mathrm{H}_q(\mathrm{S}_i) - \mathrm{H}_q(\mathrm{S}_i|\{\mathrm{A}_i\}_{i \in k}) \\
&\overset{(a)}{=} \mathrm{H}_q(\mathrm{S}_i) - \mathrm{H}_q(\mathrm{R}|\{\mathrm{A}_i\}_{i \in k}, \mathrm{M}_i) \\
&\overset{(b)}{\geq} \mathrm{H}_q(\mathrm{R}|\mathrm{M}_i) - \mathrm{H}_q(\mathrm{R}|\{\mathrm{A}_i\}_{i \in k}, \mathrm{M}_i) \\
&\overset{(c)}{\geq} \mathrm{H}_q(\mathrm{R}|\mathrm{M}_i) - \mathrm{H}_q(\mathrm{R}|\mathrm{M}_i) = 0,
\end{aligned}
$$

---

[7] We consider that $0 \log(0) = 0$ for ease of notation.

where (a) holds since $\mathbb{M}_i$ is a deterministic function of $\{\mathbb{A}_i\}_{i \in k}$, (b) holds because $\mathrm{H}_q(\mathbb{S}_i) \geq \mathrm{H}_q(\mathbb{R}|\mathbb{M}_i)$, and (c) holds since conditioning does not increase entropy. Due to the non-negativity of mutual information, equality is given if and only if the log-sum inequality is fulfilled with equality, and hence when $\mathbb{R}$ is uniform and independent of $\{\mathbb{A}_i\}_{i \in k}$. $\square$

B. Proof of Theorem 1

Proof: We will use the method of Lagrange multipliers to combine the total leakage as quantified in Lemma 2 with the constraints in Eqs. (6) to (9). The combination, called as the objective function that has to be minimized is quantified as

$$
\begin{aligned}
\mathcal{L} &\triangleq \mathcal{L}(p_1, p_1^{\text{inv}}, p_2, p_3, p_{2,3}^{\text{inv}}, \lambda_1, \lambda_2, \lambda_3, \lambda_4) \\
&= \mathrm{L}(p_1, p_1^{\text{inv}}, p_2, p_3, p_{2,3}^{\text{inv}}) + \lambda_1 c_1(p_1, p_1^{\text{inv}}) + \\
&\quad + \lambda_2 c_2(p_2, p_3, p_{2,3}^{\text{inv}}) + \lambda_3 c_3(p_1, p_2) + \lambda_4 c_4(p_1, p_3).
\end{aligned}
$$

This objective can be minimized by setting the gradient to zero, i.e., $\nabla_{p_1, p_1^{\text{inv}}, p_3, p_2, p_{2,3}^{\text{inv}}, \lambda_1, \lambda_2, \lambda_3, \lambda_4} \mathcal{L} = 0$, which is equivalent to solving a system of nine equations with nine unknowns. Simplifying $\nabla_{p_1, p_1^{\text{inv}}, p_2, p_3, p_{2,3}^{\text{inv}}} \mathcal{L} = 0$ yields

$$
p_1(p_{2,3}^{\text{inv}})^2 = p_1^{\text{inv}} p_2 p_3. \tag{22}
$$

Using the constraints Eqs. (6) to (9), one can express $p_1^{\text{inv}}, p_2, p_3, p_{2,3}^{\text{inv}}$ as polynomials of degree one in p_1, and plugging them into Eq. (31) transfers the problem into finding the root of a polynomial of degree 3 in p_1 such that it satisfies the constraints $s_{\mathbf{A}+\mathbf{R}} - 1 + s \leq p_1 s \leq s_{\mathbf{A}+\mathbf{R}}$ and $s_{\mathbf{R}} - 1 + s \leq p_1 s \leq s_{\mathbf{R}}$. Let $\bar{q} \triangleq \frac{(q-2)^2}{q-1}$, then the polynomial denotes as $\alpha p_1^3 + \beta p_1^2 + \gamma p_1 + \delta = 0$, where $\alpha = s^2(4 + \bar{q})$, $\beta = 4s(1 - s - s_{\mathbf{A}+\mathbf{R}} - s_{\mathbf{R}}) - \bar{q}s(s_{\mathbf{R}} + s_{\mathbf{A}+\mathbf{R}} + s)$, $\gamma = (1 - s - s_{\mathbf{A}+\mathbf{R}} - s_{\mathbf{R}})^2 + \bar{q}(ss_{\mathbf{A}+\mathbf{R}} + ss_{\mathbf{R}} + s_{\mathbf{A}+\mathbf{R}}s_{\mathbf{R}})$ and $\delta = -\bar{q}s_{\mathbf{A}+\mathbf{R}}s_{\mathbf{R}}$. Having p_1, the remaining unknowns $p_1^{\text{inv}}, p_2, p_3$ and $p_{2,3}^{\text{inv}}$ follow directly from (6)-(9). By the mean value theorem and the continuity of $x \log \frac{x}{y}$ on $0 \leq x < \infty$, for some $y > 0$, one can show that the global minimum implies $p_1, p_1^{\text{inv}}, p_2, p_3, p_{2,3}^{\text{inv}} \neq 0$. Hence, the solution is not at any boundary of the optimization problem and is thus captured by the Lagrange multipliers. Since the problem is convex, the solution is a global minima. $\square$

C. Sparsity of Two Shares Minimizing the Leakage

Lemma 7: Given a desired average sparsity level s_{avg}, then the minimum total leakage is obtained for $s_{\mathbf{R}} = s_{\mathbf{A}+\mathbf{R}} = s_{avg}$.

Proof: Given a desired average sparsity level $s_{avg} = \frac{s_{\mathbf{A}} + s_{\mathbf{A}+\mathbf{R}}}{2}$, we define s_δ such that $s_{\mathbf{R}} = s_{avg} - s_\delta$ and $s_{\mathbf{A}+\mathbf{R}} = s_{avg} + s_\delta$ and determine the optimal of s_δ that minimizes the leakage. We utilize the method of Lagrange multipliers to combine the total leakage from Lemma 2, with the constraints in Eqs. (6) to (9). Note that Eqs. (8) and (9) have to be slightly updated by plugging in $s_{\mathbf{R}} = s_{avg} - s_\delta$ and $s_{\mathbf{A}+\mathbf{R}} = s_{avg} + s_\delta$. Thus, the objective function to be minimized is a function of s_δ and can be expressed as

$$
\mathcal{L} \triangleq \mathrm{L}(p_1, p_1^{\text{inv}}, p_2, p_3, p_{2,3}^{\text{inv}}, s_\delta) + \lambda_1 c_1(p_1, p_1^{\text{inv}})
$$

$$
+ \lambda_2 c_2(p_2, p_3, p_{2,3}^{\text{inv}}) + \lambda_3 c_3(p_1, p_2) + \lambda_4 c_4(p_1, p_3).
$$

This objective is minimized by setting the gradient to zero, i.e., $\nabla_{p_1, p_1^{\text{inv}}, p_3, p_2, p_{2,3}^{\text{inv}}, \lambda_1, \lambda_2, \lambda_3, \lambda_4} \mathcal{L} = 0$, which is equivalent to solving a system of eleven equations with eleven unknowns.

This is a convex optimization problem, since the objective function is a sum of convex functions and the constraints are affine in the unknowns. The function $\mathrm{L}(p_1, p_1^{\text{inv}}, p_2, p_3, p_{2,3}^{\text{inv}}, s_\delta)$ is convex in s_δ as shown in the composition theorem [47, Ch. 3.2.4] since $-\log(x)$ is convex and $g(x) = s_{avg} \pm s_\delta$ is convex. Thus, $-\log(s_{avg} \pm s_\delta)$ is convex.

We now state the system of equations given by $\nabla_{p_2, p_3, s_\delta} \mathcal{L} = 0$:

$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial s_\delta} &= s \left[p_1 \left(-\frac{1}{s_{\mathbf{A}+\mathbf{R}}} + \frac{1}{s_{\mathbf{R}}} \right) \right. \\
&\quad \left. + (q-1) p_1^{\text{inv}} \left(\frac{1}{1 - s_{\mathbf{A}+\mathbf{R}}} - \frac{1}{1 - s_{\mathbf{R}}} \right) \right] \\
&\quad + (1-s) \left[p_2 \left(\frac{1}{1 - s_{\mathbf{A}+\mathbf{R}}} + \frac{1}{s_{\mathbf{R}}} \right) - p_3 \left(\frac{1}{s_{\mathbf{A}+\mathbf{R}}} + \frac{1}{1 - s_{\mathbf{R}}} \right) \right. \\
&\quad \left. + (q-2) p_{2,3}^{\text{inv}} \left(\frac{1}{1 - s_{\mathbf{A}+\mathbf{R}}} - \frac{1}{1 - s_{\mathbf{R}}} \right) \right] + \lambda_3 - \lambda_4 = 0
\end{aligned} \tag{23}
$$

$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial p_2} &= (1-s) \left(2 \log(p_2) - \log(s_{\mathbf{R}}) - \log(s_{\mathbf{A}+\mathbf{R}}^{\text{inv}}) + 2 \right) \\
&\quad + \lambda_2 + (1-s) \lambda_3 = 0
\end{aligned} \tag{24}
$$

$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial p_3} &= (1-s) \left(2 \log(p_3) - \log(s_{\mathbf{R}}^{\text{inv}}) - \log(s_{\mathbf{A}+\mathbf{R}}) + 2 \right) \\
&\quad + \lambda_2 + (1-s) \lambda_4 = 0
\end{aligned} \tag{25}
$$

We will first calculate $\lambda_3 - \lambda_4$ from (24) and (25) to be later substituted in (23). By computing (24) − (25) and dividing the result by $(1-s)$, we obtain

$$
\begin{aligned}
&\lambda_3 - \lambda_4 \\
&= 2(\log(p_3) - \log(p_2)) + \log \left(\frac{s_{\mathbf{A}+\mathbf{R}}^{\text{inv}}}{s_{\mathbf{A}+\mathbf{R}}} \right) + \log \left(\frac{s_{\mathbf{R}}}{s_{\mathbf{R}}^{\text{inv}}} \right)
\end{aligned} \tag{26}
$$

$$
\begin{aligned}
&= 2 \log \left(\frac{s_{avg} + s_\delta - s p_1}{1-s} \right) - 2 \log \left(\frac{s_{avg} - s_\delta - s p_1}{1-s} \right) \\
&\quad + \log \left(\frac{1 - s_{avg} - s_\delta}{(q-1)(s_{avg} + s_\delta)} \right) + \log \left(\frac{(q-1)(s_{avg} - s_\delta)}{1 - s_{avg} + s_\delta} \right)
\end{aligned} \tag{27}
$$

$$
= 2 \log \left(\frac{s_{avg} + s_\delta - s p_1}{s_{avg} - s_\delta - s p_1} \right) + \log \left(\frac{(1 - s_{avg} - s_\delta)(s_{avg} - s_\delta)}{(1 - s_{avg} + s_\delta)(s_{avg} + s_\delta)} \right) \tag{28}
$$

where from (26) to (27) we used (8) and (9) to express p_2 and p_3 in terms of p_1, respectively. Simplifying (23) by use of $\nabla_{\lambda_1 \lambda_2, \lambda_3, \lambda_4, \lambda_5} \mathcal{L} = 0$, i.e., the constraints in (6)-(9) yields

$$
\begin{aligned}
&\frac{\partial \mathcal{L}}{\partial s_\delta} - (\lambda_3 - \lambda_4) \\
&= \frac{1}{s_{\mathbf{R}}}(s p_1 + (1-s) p_2) - \frac{1}{s_{\mathbf{A}+\mathbf{R}}}(s p_1 + (1-s) p_3)
\end{aligned}
$$

$$+ \frac{1}{1 - s_{\mathbf{A+R}}} \underbrace{(s(q-1)p_1^{\text{inv}} + (1-s)p_2 + (1-s)(q-2)p_{2,3}^{\text{inv}})}_{(a)}$$

$$- \frac{1}{1 - s_{\mathbf{R}}} \underbrace{((q-1)p_1^{\text{inv}} + (1-s)p_3 + (q-2)(1-s)p_{2,3}^{\text{inv}})}_{(b)}$$

$$= \frac{s_{\mathbf{R}}}{s_{\mathbf{R}}} - \frac{s_{\mathbf{A+R}}}{s_{\mathbf{A+R}}} + \frac{1 - s_{\mathbf{A+R}}}{1 - s_{\mathbf{A+R}}} - \frac{1 - s_{\mathbf{R}}}{1 - s_{\mathbf{R}}} = 0, \qquad (29)$$

where we used that $s_{\mathbf{R}} = sp_1 + (1-s)p_2$ from (8), $s_{\mathbf{A+R}} = sp_1 + (1-s)p_3$ from (9). Further, we obtain $1 - s_{\mathbf{A+R}} = (a)$ by solving (6) for p_1, substituting p_1 in (9) and solving for p_3 and finally inserting in (7), multiplying by $(1-s)$ and solving for $1 - s_{\mathbf{A+R}}$. Lastly, we obtain $1 - s_{\mathbf{R}} = (b)$ by solving (6) for p_1, substituting p_1 in (8) and solving for p_2 and finally inserting in (7), multiplying by $(1-s)$ and solving for $1 - s_{\mathbf{R}}$. Putting (28) and (29) together, we obtain

$$\frac{\partial \mathcal{L}}{\partial s_\delta} = 2 \log \left(\frac{s_{\text{avg}} + s_\delta - sp_1}{s_{\text{avg}} - s_\delta - sp_1} \right)$$

$$+ \log \left(\frac{(1 - s_{\text{avg}} - s_\delta)(s_{\text{avg}} - s_\delta)}{(1 - s_{\text{avg}} + s_\delta)(s_{\text{avg}} + s_\delta)} \right) = 0.$$

Since the optimization is convex, it follows that $s_\delta = 0$ minimizes the objective in Lemma 2. $\square$

D. Proof of Theorem 2

Proof: Being a convex optimization problem with affine constraints, we utilize the method of Lagrange multipliers [55] to combine the leakage in Eq. (12) with the constraints in Eqs. (13) to (15). Then the objective function is

$$\mathcal{L} \triangleq \mathrm{L}(p_1, p_1^{\text{inv}}, p_s, p_s^{\text{inv}}) + \lambda_1 c_1(p_1, p_1^{\text{inv}})$$
$$+ \lambda_2 c_2(p_s, p_s^{\text{inv}}) + \lambda_3 c_3(p_1, p_s), \qquad (30)$$

where $\mathrm{L}(p_1, p_1^{\text{inv}}, p_s, p_s^{\text{inv}})$ is the leakage defined in the optimization problem Lemma 5, which results from Construction 2 (similar to the Lemma 2). Then the minimization problem of Lemma 5 can be solved by setting the gradient of the objective in (30) to zero, i.e., $\nabla_{p_1, p_1^{\text{inv}}, p_s, p_s^{\text{inv}}, \lambda_1, \lambda_2, \lambda_3} \mathcal{L} = 0$, which amounts to solving a system of seven equations with seven unknowns. We utilize a subset of those equations and one can show that given the objective $\mathcal{L}$, we obtain from $\nabla_{p_1, p_1^{\text{inv}}, p_s, p_s^{\text{inv}}} \mathcal{L} = 0$ the relation

$$p_1(p_s^{\text{inv}})^n = p_1^{\text{inv}} p_s^n. \qquad (31)$$

To obtain this result, we use that $\frac{\partial x \log(\frac{x}{y})}{\partial x} = \log(x) - \log(y) + 1$. Then, from (30) and the objective and constraints in Lemma 5, we obtain the following non-linear system of equations for $\nabla_{p_1, p_1^{\text{inv}}, p_s, p_s^{\text{inv}}} \mathcal{L} = 0$:

$$\frac{\partial \mathcal{L}}{\partial p_1} = s \left(\log(p_1) - \log(s_d) + 1 \right) + \lambda_1 + s\lambda_3 = 0,$$

$$\frac{\partial \mathcal{L}}{\partial p_1^{\text{inv}}} = s(q-1)\left(\log(p_1^{\text{inv}}) - \log(s_d^{\text{inv}}) + 1 \right) + (q-1)\lambda_1 = 0,$$

$$\frac{\partial \mathcal{L}}{\partial p_s} = (1-s)\Big(\log(p_s) - \log(s_d) + 1 + (n-1)$$

$$\cdot (\log(p_s) - \log(s_d^{\text{inv}}) + 1)\Big) + n\lambda_2 + (1-s)\lambda_3 = 0,$$

$$\frac{\partial \mathcal{L}}{\partial p_s^{\text{inv}}} = (1-s)(q-n)\left(\log(p_s^{\text{inv}}) - \log(s_d^{\text{inv}}) + 1 \right)$$

$$+ (q-n)\lambda_2 = 0.$$

By scaling, we simplify and reformulate this system of equations and obtain

$$0 = \log(p_1) - \log(s_d) + 1 + \lambda_1' + \lambda_3', \qquad (32)$$

$$0 = -\log(p_1^{\text{inv}}) + \log(s_d^{\text{inv}}) - 1 - \lambda_1', \qquad (33)$$

$$0 = -n\log(p_s) + \log(s_d) + (n-1)$$
$$- n - n\lambda_2' - \lambda_3, \qquad (34)$$

$$0 = n\log(p_s^{\text{inv}}) - n\log(s_d^{\text{inv}}) + n + n\lambda_2', \qquad (35)$$

where $\lambda_1' = \lambda_1/s$, $\lambda_2' = \lambda_2/(1-s)$ and $\lambda_3' = \lambda_3/s$. Summing (32)-(35) leads to $\log\left(p_1 p_s^{\text{inv}^n}\right) - \log\left(p_1^{\text{inv}} p_s^n\right) = 0$. Exponentiating both sides yields the result stated in (31). The condition on the root p_s is a consequence of (14) and (15).

Then we use (31) and $\nabla_{\lambda_1, \lambda_2, \lambda_3} \mathcal{L} = 0$, where the latter system of equations is linear and equivalent to (13)-(15). We solve Eqs. (13) to (15) for $p_1, p_1^{\text{inv}}, p_s^{\text{inv}}$ as a function of p_s to obtain

$$(q-1)\frac{s_d - (1-s)p_s}{s - s_d + (1-s)p_s} = \left((q-n)\frac{p_s}{1 - np_s}\right)^n. \qquad (36)$$

We reformulate (36) to get the following polynomial in p_s with degree $n+1$:

$$-p_s^{n+1} - s_2 p_s^n + cs_1 \sum_{k=1}^n \binom{n}{k}(-np_s)^k - p_s c \sum_{k=0}^n (-np_s)^k.$$

Sorting the coefficients by powers of p_s results in the root finding problem given in Theorem 2. The results for p_1, p_1^{inv} and p_s^{inv} can be obtained from $\nabla_{\lambda_1, \lambda_2, \lambda_3} \mathcal{L} = 0$, i.e., from the constraints stated in Eqs. (13) to (15). The condition on the root p_s is a direct consequence of Eqs. (14) and (15). As for Theorem 1, by the continuity of $x \log(\frac{x}{y})$, $y > 0$, on $0 \leq x < \infty$ and the mean value theorem, it can be shown that the global solution lies in the interior of the convex feasibility set; and hence, is captured by the Lagrange multipliers. $\square$

REFERENCES

[1] M. Xhemrishi, R. Bitar, and A. Wachter-Zeh, "Distributed matrix-vector multiplication with sparsity and privacy guarantees," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2022, pp. 1028–1033.

[2] M. Xhemrishi, M. Egger, and R. Bitar, "Efficient private storage of sparse machine learning data," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Nov. 2022, pp. 214–219.

[3] M. Egger, M. Xhemrishi, A. Wachter-Zeh, and R. Bitar, "Sparse and private distributed matrix multiplication with straggler tolerance," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2023, pp. 2535–2540.

[4] A. D. Kshemkalyani and M. Singhal, *Distributed Computing: Principles, Algorithms, and Systems*, 1st ed. Cambridge, U.K.: Cambridge Univ. Press, 2008.

[5] *European Parliament and Council of the European Union*, document Regulation (EU) 2016/679 of the European Parliament and of the Council, 2006.

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

[7] G. A. Seber and A. J. Lee, *Linear Regression Analysis*, vol. 329. Hoboken, NJ, USA: Wiley, 2012.

[8] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Process. Lett.*, vol. 9, no. 3, pp. 293–300, 1999.

[9] A. Severinson, A. G. I. Amat, and E. Rosnes, "Block-diagonal and LT codes for distributed computing with straggling servers," *IEEE Trans. Commun.*, vol. 67, no. 3, pp. 1739–1753, Mar. 2019.

[10] A. Mallick, M. Chaudhari, and G. Joshi, "Rateless codes for near-perfect load balancing in distributed matrix-vector multiplication," 2018, *arXiv:1804.10331*.

[11] A. B. Das, A. Ramamoorthy, and N. Vaswani, "Efficient and robust distributed matrix computations via convolutional coding," *IEEE Trans. Inf. Theory*, vol. 67, no. 9, pp. 6266–6282, Sep. 2021.

[12] B. Hasircioglu, J. Gomez-Vilardebo, and D. Gündüz, "Bivariate polynomial coding for efficient distributed matrix multiplication," *IEEE J. Sel. Areas Inf. Theory*, vol. 2, no. 3, pp. 814–829, Sep. 2021.

[13] A. Frigård, S. Kumar, E. Rosnes, and A. G. Amat, "Low-latency distributed inference at the network edge using rateless codes (invited paper)," in *Proc. 17th Int. Symp. Wireless Commun. Syst. (ISWCS)*, Sep. 2021, pp. 1–6.

[14] A. K. Pradhan, A. Heidarzadeh, and K. R. Narayanan, "Factored LT and factored raptor codes for large-scale distributed matrix multiplication," *IEEE J. Sel. Area Inf. Theory*, vol. 2, no. 3, pp. 893–906, Sep. 2021.

[15] S. Dutta, M. Fahim, F. Haddadpour, H. Jeong, V. Cadambe, and P. Grover, "On the optimal recovery threshold of coded matrix multiplication," *IEEE Trans. Inf. Theory*, vol. 66, no. 1, pp. 278–301, Jan. 2020.

[16] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran, "Speeding up distributed machine learning using codes," *IEEE Trans. Inf. Theory*, vol. 64, no. 3, pp. 1514–1529, Mar. 2018.

[17] H. Yang and J. Lee, "Secure distributed computing with straggling servers using polynomial codes," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 1, pp. 141–150, Jun. 2018.

[18] M. Kim, H. Yang, and J. Lee, "Private coded matrix multiplication," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 1434–1443, 2020.

[19] R. Bitar, M. Xhemrishi, and A. Wachter-Zeh, "Adaptive private distributed matrix multiplication," *IEEE Trans. Inf. Theory*, vol. 68, no. 4, pp. 2653–2673, Apr. 2022.

[20] J. Kakar, S. Ebadifar, and A. Sezgin, "Rate-efficiency and straggler-robustness through partition in distributed two-sided secure matrix computation," 2018, *arXiv:1810.13006*.

[21] R. G. L. D'Oliveira, S. El Rouayheb, and D. Karpuk, "GASP codes for secure distributed matrix multiplication," *IEEE Trans. Inf. Theory*, vol. 66, no. 7, pp. 4038–4050, Jul. 2020.

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

[23] N. Raviv and D. A. Karpuk, "Private polynomial computation from Lagrange encoding," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 553–563, 2020.

[24] M. Aliasgari, O. Simeone, and J. Kliewer, "Private and secure distributed matrix multiplication with flexible communication load," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 2722–2734, 2020.

[25] O. Makkonen and C. Hollanti, "General framework for linear secure distributed matrix multiplication with Byzantine servers," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Nov. 2022, pp. 143–148.

[26] R. Bitar, Y. Xing, Y. Keshtkarjahromi, V. Dasari, S. El Rouayheb, and H. Seferoglu, "Private and rateless adaptive coded matrix-vector multiplication," *EURASIP J. Wireless Commun. Netw.*, vol. 2021, no. 1, pp. 1–25, Dec. 2021.

[27] R. Bitar, P. Parag, and S. El Rouayheb, "Minimizing latency for secure distributed computing," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2017, pp. 2900–2904.

[28] R. Schlegel, S. Kumar, E. Rosnes, and A. Graell i Amat, "Privacy-preserving coded mobile edge computing for low-latency distributed inference," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 3, pp. 788–799, Mar. 2022.

[29] B. Hasircioglu, J. Gómez-Vilardebó, and D. Gündüz, "Bivariate polynomial codes for secure distributed matrix multiplication," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 3, pp. 955–967, Mar. 2022.

[30] S. Ulukus, S. Avestimehr, M. Gastpar, S. A. Jafar, R. Tandon, and C. Tian, "Private retrieval, computing, and learning: Recent progress and future challenges," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 3, pp. 729–748, Mar. 2022.

[31] J. Madarkar, P. Sharma, and R. P. Singh, "Sparse representation for face recognition: A review paper," *IET Image Process.*, vol. 15, no. 9, pp. 1825–1844, Jul. 2021.

[32] P. Zardoshti, F. Khunjush, and H. Sarbazi-Azad, "Adaptive sparse matrix representation for efficient matrix-vector multiplication," *J. Supercomput.*, vol. 72, pp. 3366–3386, Sep. 2016.

[33] N. Bell and M. Garland, "Implementing sparse matrix-vector multiplication on throughput-oriented processors," in *Proc. Conf. High Perform. Comput. Netw., Storage Anal.*, Nov. 2009, pp. 1–11.

[34] S. Wang, J. Liu, and N. Shroff, "Coded sparse matrix multiplication," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 5152–5160.

[35] M. Luby, "LT codes," in *Proc. 43rd Annu. IEEE Symp. Found. Comput. Sci.*, Nov. 2002, pp. 271–280.

[36] A. B. Das and A. Ramamoorthy, "Coded sparse matrix computation schemes that leverage partial stragglers," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2021, pp. 1570–1575.

[37] A. B. Das and A. Ramamoorthy, "A unified treatment of partial stragglers and sparse matrices in coded matrix computation," *IEEE J. Sel. Area Inf. Theory*, vol. 3, no. 2, pp. 241–256, Jun. 2022.

[38] A. B. Das, A. Ramamoorthy, D. J. Love, and C. G. Brinton, "Distributed matrix computations with low-weight encodings," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2023, pp. 1979–1984.

[39] A. M. Subramaniam, A. Heidarzadeh, and K. R. Narayanan, "Random Khatri–Rao-product codes for numerically-stable distributed matrix multiplication," in *Proc. 57th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Sep. 2019, pp. 253–259.

[40] R. Ji, A. Heidarzadeh, and K. R. Narayanan, "Sparse random Khatri–Rao product codes for distributed matrix multiplication," in *Proc. IEEE Inf. Theory Workshop*, Nov. 2022, pp. 416–421.

[41] M. Fahim and V. R. Cadambe, "Lagrange coded computing with sparsity constraints," in *Proc. 57th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, 2019, pp. 284–289.

[42] G. Suh, K. Lee, and C. Suh, "Matrix sparsification for coded matrix multiplication," in *Proc. 55th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Oct. 2017, pp. 1271–1278.

[43] W.-T. Chang and R. Tandon, "On the capacity of secure distributed matrix multiplication," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2018, pp. 1–6.

[44] Q. Yu and A. S. Avestimehr, "Entangled polynomial codes for secure, private, and batch distributed matrix multiplication: Breaking the 'Cubic' barrier," 2020, *arXiv:2001.05101*.

[45] R. A. Chou, J. Kliewer, R. Bitar, and A. Wachter-Zeh, "Secure distributed storage: Optimal trade-off between storage rate and privacy leakage," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2023, pp. 1324–1329.

[46] C. E. Shannon, "Communication theory of secrecy systems," *Bell Syst. Tech. J.*, vol. 28, no. 4, pp. 656–715, Oct. 1949.

[47] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[48] T. Cover and J. Thomas, *Elements of Information Theory*. Hoboken, NJ, USA: Wiley, 2012.

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

[50] R. Tandon, Q. Lei, A. Dimakis, and N. Karampatziakis, "Gradient coding: Avoiding stragglers in distributed learning," in *Proc. 34th Int. Conf. Mach. Learn.*, Aug. 2017, pp. 3368–3376.

[51] S. El Rouayheb and K. Ramchandran, "Fractional repetition codes for repair in distributed storage systems," in *Proc. 48th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Sep. 2010, pp. 1510–1517.

[52] Q. Yu, M. Maddah-Ali, and S. Avestimehr, "Polynomial codes: An optimal design for high-dimensional coded matrix multiplication," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–7.

[53] H. H. López, G. L. Matthews, and D. Valvo, "Secure MatDot cod-es: A secure, distributed matrix multiplication scheme," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Nov. 2022, pp. 149–154.

[54] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 210–227, Feb. 2009.

[55] R. T. Rockafellar, "Lagrange multipliers and optimality," *SIAM Rev.*, vol. 35, no. 2, pp. 183–238, Jun. 1993.

Rawad Bitar (Member, IEEE) received the Diploma degree in computer and communication engineering from the Faculty of Engineering, Lebanese University, Roumieh, Lebanon, in 2013, the M.S. degree from the Doctoral School, Lebanese University, Tripoli, Lebanon, in 2014, and the Ph.D. degree in electrical engineering from Rutgers University, New Brunswick, NJ, USA, in 2020. He is currently pursuing the Habilitation degree with the Technical University of Munich. He is a Research Group Leader with the Technical University of Munich. His research interests include information theory and coding theory, with a focus on coding for information-theoretically secure distributed systems with application to machine learning and coding for DNA-based data storage.

Antonia Wachter-Zeh (Senior Member, IEEE) received the M.Sc. degree in communications technology from Ulm University, Germany, in 2009, and the Ph.D. degree from Ulm University and Universite de Rennes I, Rennes, France, in 2013. She is an Associate Professor with the School of Computation, Information and Technology, Technical University of Munich (TUM), Munich, Germany. From 2013 to 2016, she was a Post-Doctoral Researcher with the Technion—Israel Institute of Technology, Haifa, Israel. From 2016 to 2020, she was a tenure track Assistant Professor with TUM. She was a recipient of the DFG Heinz Maier-Leibnitz-Preis and the ERC Starting Grant. Her research interests are coding theory; cryptography and information theory and their application to storage, communications, privacy, and security; and machine learning. She is an Associate Editor of IEEE TRANSACTIONS ON INFORMATION THEORY.

Maximilian Egger (Graduate Student Member, IEEE) received the B.Eng. degree (Hons.) in electrical engineering from the University of Applied Sciences Augsburg in 2020, and the M.Sc. degree (Hons.) in electrical engineering and information technology from the Technical University of Munich in 2022. He is currently pursuing the Ph.D. degree with the Institute for Communications Engineering. He pursued a dual bachelor's study accompanied by engineering positions in hardware and software development departments with Hilti AG. In 2019, he received a three-years scholarship from the German Academic Scholarship Foundation. In 2020, he received the VDE Award Bavaria in recognition of his bachelor's thesis. His research interests are in the field of large-scale decentralized computing and federated learning.

Marvin Xhemrishi (Graduate Student Member, IEEE) was born in Tirana, Albania, in 1996. He received the bachelor's degree in telecommunications engineering from the Polytechnic University of Tirana, Albania, in 2017, and the M.Sc. degree (Hons.) in communications engineering from the Technical University of Munich, Munich, Germany, in 2019, where he is currently pursuing the Doctoral degree with the Technical University of Munich in the Associate Professorship for Coding and Cryptography. His research interests lie in the broad area of coding theory and information theory, with a focus on coding theory for private distributed computing, federated learning, and physical-unclonable functions.