Private Polynomial Computation from Lagrange Encoding

Netanel Raviv[⋆] and **David A. Karpuk**[†]

[⋆]Department of Electrical Engineering, California Institute of Technology, Pasadena, CA 91125, USA.
[†]Departamento de Matemáticas, Universidad de los Andes, Bogotá, Colombia.

Abstract—**Private computation is a generalization of private information retrieval, in which a user is able to compute a function on a distributed dataset without revealing the identity of that function to the servers. In this paper it is shown that Lagrange encoding, a powerful technique for encoding Reed-Solomon codes, enables private computation in many cases of interest. In particular, we present a scheme that enables private computation of polynomials of any degree on Lagrange encoded data, while being robust to Byzantine and straggling servers, and to servers colluding to attempt to deduce the identities of the functions to be evaluated. Moreover, incorporating ideas from the well-known Shamir secret sharing scheme allows the data itself to be concealed from the servers as well. Our results extend private computation to high degree polynomials and to data-privacy, and reveal a tight connection between private computation and coded computation.**

I. Introduction

A. Private Information Retrieval

Private Information Retrieval (PIR) refers to the process of downloading a file from a database, without revealing to the database which file is being downloaded. PIR was originally introduced in the seminal work [4] from the perspective of Computer Science, where the goal is to construct PIR scheme with minimal communication cost. Much work has been done on PIR from this point of view, both under information-theoretic constraints [30] and computational ones [19], of which we also briefly mention [10, 29].

Recently, due to rising interest in large scale distributed storage systems and emerging quantum attacks on computationally secure protocols, PIR has experienced a flurry of research from the Information Theory community. In the Information-Theoretic formulation of the PIR problem, the primary performance metric of a PIR scheme is the download rate, which refers to the amount of downloaded bits per one information bit. The capacity of PIR, given a specific system setup, is the maximum possible download rate. Commonly, one assumes that files of length K are distributed over N different servers using an $[N, K]$ *Maximum Distance Separable* (MDS) storage code, and every T out of these N servers can collude to try to deduce the identity of the file being downloaded. The capacity of several variations on the PIR problem is now known, for example for $K = T = 1$ by [23], for $K = 1$ and $T \geq 1$ by [24], and for $K \geq 1$ and $T = 1$ by [2]. Additional interesting and practically relevant variants on PIR appear

when one accounts for unreliability of the servers, whether they be unresponsive, malicious, or curious as to the contents of the data they are storing. Variants along these lines have been studied and capacity expressions have been derived in [3, 11, 24].

Many open questions about PIR remain, for example the PIR capacity is unknown when $K, T > 1$, that is, when a non-trivial storage code is used and we have non-trivial server collusion; less is known when $K, T > 1$ and we try to account for unreliable servers. Given the difficulty of proving optimality in such generality, one is often content with achieving the more modest goal of constructing PIR schemes with large download rate and ignoring questions of capacity. Algebraic constructions of PIR schemes for coded data with colluding servers appeared in [8, 26], which were then generalized in [27] to account for unresponsive and Byzantine servers. Additional work along these lines has been done in [9, 13] for non-MDS storage codes. These schemes generally fall under the umbrella of "one-shot schemes", a term recently christened in [6], where it was also shown how to lift such schemes to obtain schemes which achieve capacity in certain cases. Reed-Solomon and Reed-Muller codes have been especially useful in such constructions, because of their compatibility with the star product between two linear codes (see Section II-C).

B. Private Computation

Private Computation (PC) is a generalization of PIR wherein the user wants to not just privately download a file from the database, but privately compute an arbitrary function of it. In PC, privacy refers to hiding the identity of the function to be computed. One recovers the PIR problem by specifying to functions given by coordinate projections. The principal performance metric for PC protocols is the *PC-rate* (or simply, rate), which is the ratio between the number of desired function evaluations and the total number of function evaluations downloaded.

For functions which are linear combinations of the files, PC was studied in [16, 25] for uncoded databases, and in [17, 18] for coded databases. The case of non-linear functions, and especially polynomial functions of degree larger than one, was studied by the second author of the current work in [12], where a PC scheme was constructed for polynomial functions on systematically coded databases.

C. Coded Computation

The term *Coded Computation* (CC) broadly refers to a family of techniques in which redundancy is added to datasets, in order to alleviate various issues that arise in distributed computations. Motivated by recent applications in machine learning, studies in Coded Computation have focused, e.g., on accelerating distributed tasks, combating malicious interventions, providing various forms of privacy, alleviating the communication load in iterative algorithms, and more.

The study of this topic has been highly prolific in recent years, and works on the topic have mostly been task-specific. Typical tasks of interest include matrix multiplication [7, 31], accelerating gradient descent algorithms [15, 20, 28], communication reduction [14], and data shuffling [1, 5]. Yet, coded computing of general polynomials has been addressed only recently in [32], which is tightly connected to our results. In [32] it was shown that coding the data by using the well-known Lagrange polynomials can amend issues of resiliency, security, and privacy in many tasks of interest.

D. Current Contributions

We present a Private Computation scheme for the evaluation of degree G polynomials on K data vectors $\mathbf{x}_k \in \mathbb{F}_q^M$, encoded using an $[N, K+E]$ Reed-Solomon code, which hides the identity of the polynomials to be computed from any T colluding servers, and hides the contents of the data vectors $\mathbf{x}_k$ from any E colluding servers. The scheme is robust against any P stragglers and any A adversaries, and has PC rate

$$R = \frac{\frac{N - (G(K+E-1) + T + P + 2A)}{N}}{\frac{K}{G(K+E-1)+1}}, \tag{1}$$

which clearly requires that $N > (G(K+E-1)+T+P+2A)$. We provide a non-trivial example which illustrates the scheme construction. Lastly, we propose an alternative scheme in the case that $P = A = E = 0$, which employs systematic data encoding and achieves a rate of

$$R = \frac{\min\{N - (G(K-1)+T), K\}}{N}.$$

This second scheme construction first appeared in [12]. We provide some basic analysis as to the relative performance of the two schemes.

The scheme construction borrows ideas from Coded Computation, especially those of [32]. Data storage is realized by evaluating interpolating polynomials, which allows the evaluation of polynomials on encoded data to be viewed as the evaluation of a single variable polynomial. This single-variable polynomial is the composition of two polynomial functions of known degree, and its degree can therefore be calculated explicitly and serves as crucial knowledge for the scheme construction and rate calculation.

Our techniques are applicable in many real-world scenarios. Beyond direct applications in distributed computations over finite fields, such as large scale matrix multiplication, one can also apply our techniques in real-number scenarios with minor additional effort, by quantization and embedding of real values into a large enough finite field. Such quantization techniques were employed in machine learning scenarios, e.g., in distributed linear regression in [32], and in distributed logistic regression in [22].

E. Comparison with Previous Work

The scheme construction presented here also generalizes some ideas from [27] to deal with non-linear functions and data privacy. In the case of $E = 0$ and $G = 1$ the current scheme achieves a rate of

$$R = \frac{N - (K + T + P + 2A - 1)}{N}. \tag{2}$$

If the functions to be computed are all distinct coordinate projections, then we reduce to the case of the PIR problem, and the rate of (2) matches that of [27]. If one further assumes that $P = A = 0$, one achieves a rate of

$$R = \frac{N - (K + T - 1)}{N} \tag{3}$$

which is that of [8]. The rate (3) is also the asymptotic rate (as the number of files $M \to \infty$) of [18] when $T = 1$, which studies linear computations of coded databases. In all cases where $E = 0$ and the PIR or PC capacity C is known, the current scheme achieves the asymptotic capacity as the number of files grows, that is, $R = \lim_{M \to \infty}$ C.

The capacity for the case $E > 0$ is subtler. The case of $K = 1$, $G = 1$, $P = A = 0$ is that of the recent work [11], which studies the PIR problem under the constraints of T-function privacy and E-data privacy, for storage systems in which every server stores an amount of data which is comparable to the entire dataset. In [11], the authors show that

$$\lim_{M \to \infty} \mathsf{C} = \begin{cases} \frac{N - (E+T)}{N} & \text{if } N > E + T \\ 0 & \text{if } N \le E + T \end{cases}$$

where C is the PIR capacity of this setting, and use a technique they deem *Cross Subspace Alignment* to construct an explicit scheme which achieves the above rate. The rate R we achieve in this scenario is

$$R = \frac{N - (E+T)}{N} \cdot \frac{1}{E+1}$$

whenever $N > E + T$, which is strictly worse than that of [11] when $E > 0$. This essentially stems from the fact that it is not clear how to align the noise terms arising from the function-privacy randomness and data-privacy randomness as in [11] for non-linear functions. Thus our PC scheme leaves room for improvement in the case of non-trivial data privacy.

II. PRELIMINARIES

A. Notation

Throughout the current paper, we adopt the following notation. For a given integer N, define $[N]$ to be the set $\{1, 2, \ldots, N\}$ of the first N natural numbers. We use lowercase letters $a, b, c, \ldots$ for scalars or indexing variables, and uppercase letters $A, B, C, \ldots$ for system parameters, such as the number of servers, dimension of the storage code, etc. Boldface lowercase letters $\mathbf{a}, \mathbf{b}, \mathbf{c}, \ldots$ are reserved for vectors,

and boldface uppercase letters $\mathbf{A}, \mathbf{B}, \mathbf{C}, \ldots$ for matrices. Calligraphic letters $\mathcal{A}, \mathcal{B}, \mathcal{C}, \ldots$ are used to denote sets, including linear codes. The symbol $\mathbb{F}_q$ is used to denote the finite field of cardinality q. Lowercase Greek letters such as $\phi, \psi, \rho, \ldots$ will generally be used to denote functions, and $\phi \circ \psi$ denotes the composition of ϕ and ψ, that is, $(\phi \circ \psi)(x) = \phi(\psi(x))$.

An $[N, K, D]_q$ code $\mathcal{C}$ is a linear code over $\mathbb{F}_q$ with length N, dimension K, and minimum distance D. We will often omit the D or q if they are unimportant or clear from context. Recall that $\mathcal{C}$ is MDS (Maximum Distance Separable) if $D = N - K + 1$.

We let $\mathcal{P}_{M,G}$ denote the set of all polynomials with coefficients in $\mathbb{F}_q$ in M variables with total degree at most G, that is,

$$\mathcal{P}_{M,G} = \{\phi \in \mathbb{F}_q[X_1, \ldots, X_M] : \deg(\phi) \leq G\}.$$

Note that $\mathcal{P}_{M,G}$ is a finite-dimensional vector space over $\mathbb{F}_q$ and therefore supports a uniform distribution. The parameter M will be of less interest to us in general and hence we will often write $\mathcal{P}_G$ for $\mathcal{P}_{M,G}$.

B. Reed-Solomon Codes

Given parameters N, K, and $q \geq N$, we construct the $[N, K]$ *Reed-Solomon* code over $\mathbb{F}_q$ as follows. Let $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_N)$ be N distinct elements of $\mathbb{F}_q$, and let

$$\mathbb{F}_q[z]^{<K} = \{\phi \in \mathbb{F}_q[z] : \deg(\phi) \leq K - 1\}$$

We define the Reed-Solomon code $\mathcal{RS}_K(\boldsymbol{\alpha})$ associated with this data to be the image of the evaluation map

$$ev : \mathbb{F}_q[z]^{<K} \to \mathbb{F}_q^N, \quad ev(\phi) = (\phi(\alpha_1), \ldots, \phi(\alpha_N))$$

It is a classical result that $\mathcal{RS}_K(\boldsymbol{\alpha})$ is an MDS code.

We will be particularly interested in the encoding of Reed-Solomon codes. If one chooses a basis $\phi_1, \ldots, \phi_K$ of the vector space $\mathbb{F}_q[z]^{<K}$, then a generator matrix (and therefore an encoding map) is given by

$$\mathbf{G} = \begin{pmatrix} \phi_1(\alpha_1) & \phi_1(\alpha_2) & \cdots & \phi_1(\alpha_N) \\ \phi_2(\alpha_1) & \phi_2(\alpha_2) & \cdots & \phi_2(\alpha_N) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_K(\alpha_1) & \phi_K(\alpha_2) & \cdots & \phi_K(\alpha_N) \end{pmatrix}.$$

The following two choices of generator matrices will be the most important for our purposes.

1) Canonical Encoding: The basis $1, z, \ldots, z^{K-1}$ of $\mathbb{F}_q[X]^{<K}$ gives rise to the Vandermonde generator matrix

$$\mathbf{G} = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_N \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{K-1} & \alpha_2^{K-1} & \cdots & \alpha_N^{K-1} \end{pmatrix}$$

Another way to view this encoding map is to map a message $\mathbf{a} = (a_1, \ldots, a_K) \in \mathbb{F}_q^{K-1}$ to the polynomial $\phi_{\mathbf{a}} = \sum_{i=0}^{K-1} a_{i+1} z^i$. The codeword is then $(\phi_{\mathbf{a}}(\alpha_1), \ldots, \phi_{\mathbf{a}}(\alpha_N)) = \mathbf{a}\mathbf{G}$.

2) Lagrange Encoding: [32] In this case we choose a basis of $\mathbb{F}_q[z]^{<K}$ to consist of interpolating polynomials. More specifically, choose some $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_K)$ consisting of distinct $\beta_k \in \mathbb{F}_q$. If $\mathbf{a} = (a_1, \ldots, a_K) \in \mathbb{F}_q^K$ is a message vector, define $u_{\mathbf{a}, \boldsymbol{\beta}} \in \mathbb{F}_q[z]^{<K}$ by the property $u_{\mathbf{a}, \boldsymbol{\beta}}(\beta_k) = a_k$ for all $k = 1, \ldots, K$. The Lagrange Interpolation formula guarantees the existence of $u_{\mathbf{a}, \boldsymbol{\beta}}$, and the uniqueness is a consequence of the degree restriction.

A generator matrix corresponding to this encoding map is given by

$$\mathbf{G} = \mathbf{G}(\boldsymbol{\alpha}, \boldsymbol{\beta}) \triangleq$$
$$\begin{pmatrix} \prod_{j \in [K] \setminus \{1\}} \frac{\beta_j - \alpha_1}{\beta_j - \beta_1} & \cdots & \prod_{j \in [K] \setminus \{1\}} \frac{\beta_j - \alpha_N}{\beta_j - \beta_1} \\ \prod_{j \in [K] \setminus \{2\}} \frac{\beta_j - \alpha_1}{\beta_j - \beta_2} & \cdots & \prod_{j \in [K] \setminus \{2\}} \frac{\beta_j - \alpha_N}{\beta_j - \beta_2} \\ \vdots & \ddots & \vdots \\ \prod_{j \in [K] \setminus \{K\}} \frac{\beta_j - \alpha_1}{\beta_j - \beta_K} & \cdots & \prod_{j \in [K] \setminus \{K\}} \frac{\beta_j - \alpha_N}{\beta_j - \beta_K} \end{pmatrix}, \quad (4)$$

where we omit the notation $\boldsymbol{\alpha}, \boldsymbol{\beta}$ when clear from context. If one defines $\beta_k = \alpha_k$ for $k = 1, \ldots, K$, then the above matrix $\mathbf{G}$ is in systematic form. The fact that $\operatorname{Im} \mathbf{G}(\boldsymbol{\alpha}, \boldsymbol{\beta}) \triangleq \{\mathbf{a}\mathbf{G}(\boldsymbol{\alpha}, \boldsymbol{\beta}) | \mathbf{a} \in \mathbb{F}_q^K\}$ equals $\mathcal{RS}_K(\boldsymbol{\alpha})$ for every $\boldsymbol{\beta}$ might not be readily seen, but is easily proved in the following lemma.

Lemma 1. *For every set $\boldsymbol{\alpha} \subseteq \mathbb{F}_q$ of size N, and every set $\boldsymbol{\beta} \subseteq \mathbb{F}_q$ of size K, we have that $\operatorname{Im} \mathbf{G}(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \mathcal{RS}_K(\boldsymbol{\alpha})$.*

Proof. Let $\mathbf{G} = \mathbf{G}(\boldsymbol{\alpha}, \boldsymbol{\beta})$. First, observe that $\mathbf{a}\mathbf{G} = ev(u_{\mathbf{a}, \boldsymbol{\beta}})$ for every $\mathbf{a} \in \mathbb{F}_q^K$. This readily implies the inclusion $\operatorname{Im} \mathbf{G} \subseteq \mathcal{RS}_K(\boldsymbol{\alpha})$, since $\mathcal{RS}_K(\boldsymbol{\alpha})$ includes all evaluations at $\boldsymbol{\alpha}$ of all univariate polynomials of degree at most $K - 1$, one of which is $u_{\mathbf{a}, \boldsymbol{\beta}}$. To prove the converse inclusion, let $ev(f) \in \mathcal{RS}_K(\boldsymbol{\alpha})$, and define $\mathbf{y} = (f(\beta_1), \ldots, f(\beta_K))$. Since the polynomial $u_{\mathbf{y}, \boldsymbol{\beta}}$ agrees with the polynomial f on $\{\beta_i\}_{i=1}^K$, and since the degrees of both f and $u_{\mathbf{y}, \boldsymbol{\beta}}$ are at most $K - 1$, if follows that $f = u_{\mathbf{y}, \boldsymbol{\beta}}$, which implies that $ev(u_{\mathbf{y}, \boldsymbol{\beta}}) = ev(f)$, and hence $ev(f) \in \operatorname{Im} \mathbf{G}$. $\square$

C. Star Products

Let $\mathbf{x} = (x_1, \ldots, x_N), \mathbf{y} = (y_1, \ldots, y_N) \in \mathbb{F}_q^N$. Their *star product* is defined to be

$$\mathbf{x} \star \mathbf{y} = (x_1 y_1, \ldots, x_N y_N) \in \mathbb{F}_q^N$$

Now let $\mathcal{C}$ and $\mathcal{D}$ be length N linear codes over $\mathbb{F}_q$. Their *star product* $\mathcal{C} \star \mathcal{D}$ is another linear code of length N, defined to be

$$\mathcal{C} \star \mathcal{D} = \operatorname{span}_{\mathbb{F}_q} \{\mathbf{c} \star \mathbf{d} : \mathbf{c} \in \mathcal{C}, \ \mathbf{d} \in \mathcal{D}\}.$$

Similarly, if $G \geq 1$ is any positive integer, we define

$$\mathcal{C}^{\star G} = \underbrace{\mathcal{C} \star \cdots \star \mathcal{C}}_{G \text{ times}} = \operatorname{span}_{\mathbb{F}_q} \{\mathbf{c}_1 \star \cdots \star \mathbf{c}_G : \mathbf{c}_g \in \mathcal{C}\}.$$

Reed-Solomon codes are especially well-behaved with respect to star products, since the star product of two evaluation vectors is the evaluation vector of the product of the two functions. The following proposition essentially appears in

[8] and [12], but we include a proof here for the sake of completeness.

Proposition 2. *Let $\mathcal{RS}_K(\boldsymbol{\alpha})$ and $\mathcal{RS}_T(\boldsymbol{\alpha})$ be Reed-Solomon codes of length N with the same evaluation vector $\boldsymbol{\alpha}$. Then:*

(i) $\mathcal{RS}_K(\boldsymbol{\alpha}) \star \mathcal{RS}_T(\boldsymbol{\alpha}) = \mathcal{RS}_{\min\{K+T-1,N\}}(\boldsymbol{\alpha})$, *and*

(ii) $\mathcal{RS}_K(\boldsymbol{\alpha})^{\star G} = \mathcal{RS}_{\min\{G(K-1)+1,N\}}(\boldsymbol{\alpha})$.

Proof. Let $f, g \in \mathbb{F}_q[z]$, where $\deg(f) < K$ and $\deg(g) < T$. Then $\deg(fg) < K+T-1$, and hence $\mathcal{RS}_K(\boldsymbol{\alpha}) \star \mathcal{RS}_T(\boldsymbol{\alpha}) \subseteq \mathcal{RS}_{\min\{K+T-1,N\}}(\boldsymbol{\alpha})$. Conversely, every polynomial of degree less than $K+T-1$ is a linear combination of polynomials of the form fg, where $\deg(f) < K$ and $\deg(g) < T$. This suffices to prove part (i). Part (ii) is proved using part (i) and an easy induction argument on G. $\square$

D. The Shamir Secret Sharing Scheme

The perfect privacy guarantees in the sequel can be seen as a special case of the Shamir secret sharing scheme [21]. By now a classic result, the Shamir secret sharing scheme allows N parties to share L secrets, such that sets of at most X parties cannot infer anything about the secrets, and sets of at least $X+L$ parties can reconstruct all L secrets. The Shamir scheme relies on linear encoding of the following form. Let $\mathbf{s}_1, \ldots, \mathbf{s}_L$ be the secrets, seen as column vectors over a finite field. An external trusted party generates X random column vectors $\mathbf{t}_1, \ldots, \mathbf{t}_X$ of the same length as the secrets, performs linear encoding

$$[\mathbf{s}_1, \ldots, \mathbf{s}_L, \mathbf{t}_1, \ldots, \mathbf{t}_X] \cdot \begin{pmatrix} \mathbf{A} \\ \mathbf{B} \end{pmatrix} = (\mathbf{y}_1, \ldots, \mathbf{y}_N),$$

where $\mathbf{A} \in \mathbb{F}_q^{L \times N}$ and $\mathbf{B} \in \mathbb{F}_q^{X \times N}$, and distributes the *shares* $\mathbf{y}_1, \ldots, \mathbf{y}_N$ to the parties. The following lemma is well-known, and will be most useful in the sequel.

Lemma 3. *If $\mathbf{B}$ is an MDS matrix (i.e., if every $X \times X$ submatrix of it is invertible), then*

$$I(\mathbf{s}_1, \ldots, \mathbf{s}_L; \mathbf{y}_{n_1}, \ldots, \mathbf{y}_{n_X}) = 0$$

for every subset $\{n_1, \ldots, n_X\} \subseteq [N]$.

E. Private Computation of Coded Data

We consider the problem of Private Computation on distributed storage systems of the following type; this follows a standard setup in the PIR literature, see [2, 8, 26]. Let $\mathbf{x}_1, \ldots, \mathbf{x}_K \in \mathbb{F}_q^{M \times 1}$, and let $\mathcal{C}$ be an $[N, K]$ code over $\mathbb{F}_q$ with generator matrix $\mathbf{G} \in \mathbb{F}_q^{K \times N}$. Define vectors $\mathbf{y}_n \in \mathbb{F}_q^{M \times 1}$ for $n = 1, \ldots, N$ by

$$[\mathbf{y}_1 \; \cdots \; \mathbf{y}_N] = \mathbf{X} \cdot \mathbf{G}, \text{ where}$$
$$\mathbf{X} \triangleq [\mathbf{x}_1 \; \cdots \; \mathbf{x}_K]. \tag{5}$$

The vector $\mathbf{y}_n$ is stored on server n. We refer to $\mathcal{C}$ as the *storage code*.

Given the above setup, a user wishes to compute $\phi_b(\mathbf{x}_k)$ for some functions $\phi_1, \ldots, \phi_B$, for all $k = 1, \ldots, K$. We assume that the functions ϕ_b all belong to some (necessarily finite-dimensional) vector space $\mathcal{S}$ of functions $\mathbb{F}_q^{M \times 1} \to \mathbb{F}_q$. To accomplish this goal, the user sends S *queries*

$\rho_n^{(1)}, \ldots, \rho_n^{(S)} \in \mathcal{S}$ to the n'th server, who responds with the *answers* $\rho_n^{(1)}(\mathbf{y}_n), \ldots, \rho_n^{(S)}(\mathbf{y}_n)$. From all NS answers, the user must be able to decode the desired function evaluations:

$$H(\{\phi_b(\mathbf{x}_k)\}|\{\rho_n^{(s)}(\mathbf{y}_n)\}) = 0.$$

It is useful to think of the above as happening over S *rounds* or *iterations*, so that during the s'th round the user queries the servers with the functions $\rho_n^{(s)}$ and obtains the answers $\rho_n^{(s)}(\mathbf{y}_n)$. Similarly, it is useful to think of the parameter B as analogous to the block length of a file in traditional PIR. We view the parameters B and S as free for the user to adjust to maximize their download rate. Here the terms $\phi_b(\mathbf{x}_k)$ are random variables in the sense that the contents of the database are, to the user, unknown and therefore best treated as random. The terms $\rho_n^{(s)}(\mathbf{y}_n)$ are random variables in the sense that the queries $\rho_n^{(s)}$ are sampled according to some distribution employed by the user to preserve privacy. We forego making this precise for the sake of readability.

Our primary function space of interest is $\mathcal{S} = \mathcal{P}_G = \mathcal{P}_{M,G}$, the space of polynomial functions of total degree at most G from $\mathbb{F}_q^{M \times 1}$ to $\mathbb{F}_q$. Thus we in general have $\phi_b, \rho_n^{(s)} \in \mathcal{P}_G$, for every $b \in [B]$, $n \in [N]$, and $s \in [S]$.

For any T-subset $\mathcal{T} = \{n_1, \ldots, n_T\}$ of $[N]$, we let $\rho_{\mathcal{T}}$ be the joint distribution of all $\rho_n^{(s)}$ for all $n \in \mathcal{T}$ and all $s = 1, \ldots, S$. PC scheme has T-*function-privacy* if

$$I(\phi_1, \ldots, \phi_B; \rho_{\mathcal{T}}) = 0$$
$$\text{for all } T\text{-subsets } \{n_1, \ldots, n_T\} \text{ of } [N].$$

That is, a PC scheme has T-function-privacy if the identities of the functions $\phi_1, \ldots, \phi_B$ to be computed remain private even after any T of the servers collude to attempt to deduce the identities of the ϕ_b.

A PC scheme has E-*data-privacy* if

$$I(\mathbf{x}_1, \ldots, \mathbf{x}_K; \mathbf{y}_{n_1}, \ldots, \mathbf{y}_{n_E}) = 0$$
$$\text{for all } E\text{-subsets } \{n_1, \ldots, n_E\} \text{ of } [N].$$

That is, the servers in the distributed storage system remain oblivious to the contents of the uncoded data, even if E of them collude to attempt to deduce the identities of the $\mathbf{x}_k$.

A PC scheme is *robust against P stragglers* or *unresponsive servers* if the user is still able to decode the values $\phi_b(\mathbf{x}_k)$ even if, during any round of the scheme, up to P servers respond with an erasure symbol ? instead of the true answer $\rho_n^{(s)}(\mathbf{y}_n)$. Similarly, a PC scheme is *robust against A adversaries* or *Byzantine servers* if the user is still able to decode the values $\phi_b(\mathbf{x}_k)$ even if, during any round of the scheme, up to A servers respond with an arbitrary element of $\mathbb{F}_q$ instead of the true answer $\rho_n^{(s)}(\mathbf{y}_n)$. From a coding-theoretic perspective, having P stragglers and A adversaries simply means that during the s'th round, the user receives the total response vector

$$(\rho_1^{(s)}(\mathbf{y}_1), \ldots, \rho_N^{(s)}(\mathbf{y}_N)) + \boldsymbol{\epsilon}^{(s)}$$

where $\boldsymbol{\epsilon}^{(s)}$ is a vector containing at most P erasure symbols ? and at most A non-zero elements of $\mathbb{F}_q$. Here the erasure symbol is understood to be absorbing with respect to addition:

q	field size
K	number of uncoded data vectors $\mathbf{x}_k$
M	length of vectors $\mathbf{x}_k$
G	degree of polynomial functions to be evaluated on the $\mathbf{x}_k$
N	number of servers
$\mathcal{C}$	$[N, K]$ storage code
T	number of colluding function-curious servers
E	number of colluding data-curious servers
A	number of adversaries / Byzantine servers
P	number of stragglers / unresponsive servers
S	number of rounds
B	number of functions to be evaluated
R	download rate of PC scheme

TABLE I
IMPORTANT PARAMETERS USED IN PRIVATE COMPUTATION SCHEMES.

$x + ? = ?$ for all $x \in \mathbb{F}_q$. See [3, 24, 27] for more on PIR from systems with stragglers and adversaries.

Given a PC scheme, our principal metric of efficiency will be the *download rate*, also referred to as the *PC rate* or simply *rate*, which is defined to be

$$R = \frac{KB}{NS}. \tag{6}$$

That is, the rate R is the number of desired function evaluations $\phi_b(\mathbf{x}_k)$ the user obtains, divided by the total number of function evaluations $\rho_n^{(s)}(\mathbf{y}_n)$ downloaded. From a strict Information-Theoretic point of view, it would be more correct to define the rate to be

$$R = \frac{H\left(\{\phi_b(\mathbf{x}_k)\}\right)}{\sum_{n=1}^{N}\sum_{s=1}^{S} H\left(\rho_n^{(s)}(\mathbf{y}_n)\right)}, \tag{7}$$

which better accounts for potential dependencies between the variables $\phi_b(\mathbf{x}_k)$. If all $\phi_b(\mathbf{x}_k)$ are independent for all b, k, and furthermore $H(\phi_b(\mathbf{x}_k)) = H(\rho_n^{(s)}(\mathbf{y}_n))$ for all b, k, n, s, then the two expressions in (6) and (7) for the rate coincide. This is indeed the case given reasonable independence conditions on the data vectors and the functions to be evaluated, but for the sake of compactness and readability we will ignore these subtleties and use (6) as our definition of rate.

Much of the current literature on PIR concerns itself with establishing the *capacity* C of a given PIR setup. The capacity of a PIR problem is defined to be the supremum of all possible PIR rates. As the current work is only concerned with explicit scheme constructions and not with establishing the capacity of any PC setup, we will mostly ignore this notion in the sequel.

We aggregate all of the important parameters of our system in Table I.

F. Private Information Retrieval as Private Computation and a Remark on Upload Cost

The problem of Private Information Retrieval is a special case of the problem of Private Computation described above. Indeed, in PIR one typically supposes that the data matrix $\mathbf{X}$ is composed of several row blocks $\mathbf{x}^i \in \mathbb{F}_q^{B \times K}$, which are the files of the system. Setting the functions ϕ_b to be the B coordinate projections corresponding to the B coordinates of

some $\mathbf{x}^i$, we see that computing $\phi_b(\mathbf{x}_k)$ for all $b = 1, \ldots, B$ and all $k = 1, \ldots, K$ is equivalent to retrieving each row of $\mathbf{x}^i$, that is, downloading the file $\mathbf{x}^i$. Since coordinate projections have degree $G = 1$, any PC scheme for privately computing polynomials of arbitrary degree G specifies to a PIR scheme when we set $G = 1$ and choose the ϕ_b to all be distinct coordinate projections.

In Private Information Retrieval, one typically justifies ignoring the upload cost of a PIR scheme by assuming that the files $\mathbf{x}^i$ have entries in some field extension $\mathbb{K}$ of $\mathbb{F}_q$, while the storage code $\mathcal{C}$ and queries $\rho_n^{(s)}$ all remain defined over $\mathbb{F}_q$. Provided that $[\mathbb{K} : \mathbb{F}_q] \gg 0$, the upload cost is dominated by the download cost, which justifies using the download rate as the sole performance metric. Generally, one can ignore the extension field $\mathbb{K}$ as all important operations and analysis occur over the base field $\mathbb{F}_q$.

Similar considerations allow one to ignore the upload cost in Private Computation. One can suppose that the data matrix is defined over some large extension $\mathbb{K}$ of $\mathbb{F}_q$, and that the function space $\mathcal{P}_G$ consists of polynomials with coefficients in $\mathbb{F}_q$ itself. Communicating an arbitrary polynomial in $\mathcal{P}_G$ costs $\sum_{i=0}^{G} \binom{M}{i}$ elements of $\mathbb{F}_q$, and therefore the user's total upload cost will be $NS \sum_{i=0}^{G} \binom{M}{i}$. However, the total download cost is easily seen to be $NS \cdot [\mathbb{K} : \mathbb{F}_q]$ many elements of $\mathbb{F}_q$, and therefore if $\mathbb{K}$ satisfies $[\mathbb{K} : \mathbb{F}_q] \gg \sum_{i=0}^{G} \binom{M}{i}$, one can ignore the upload cost. We note that the upload cost increases very quickly in G, and therefore in practice this inequality may be satisfied only for small G.

Finally, we note that a simple relaxation of the privacy requirement can drastically reduce the upload cost. Notice that the expression $NS \sum_{i=0}^{G} \binom{M}{i}$ for upload cost stems from the expression $NS \dim_{\mathbb{F}_q}(\mathcal{P}_G)$, where $\dim_{\mathbb{F}_q}(\mathcal{P}_G)$ is the dimension of the set $\mathcal{P}_G$ as a subspace over $\mathbb{F}_q$. More generally, in order to communicate a function in a subspace $\mathcal{P} \subseteq \mathcal{P}_G$, where $\mathcal{P}$ is known to all, one only has to communicate $\dim_{\mathbb{F}_q}(\mathcal{P})$ field elements. Hence, one can determine any such subspace $\mathcal{P}$, either a priori or by communicating a sparse basis of it. Then, one can apply an identical scheme in which $\phi_b \in \mathcal{P}$ for every $b \in [B]$. The resulting privacy, however, will be restricted to $\mathcal{P}$, that is, the definition of T-function privacy will be reduced to

$$I(\phi_1, \ldots, \phi_B | \forall i \in [B], \phi_i \in \mathcal{P}; \rho_{\mathcal{T}} | \forall n_i \in \mathcal{T}, \rho_{n_i} \in \mathcal{T}) = 0$$

for all T-subsets $\{n_1, \ldots, n_T\}$ of $[N]$.

III. A GENERAL CONSTRUCTION

In this section we present the main Private Computation scheme of the paper which applies for $N > (G(K + E - 1) + T + P + 2A)$. With the system parameters as in Table I, the PC scheme has rate

$$R = \frac{\frac{N - (G(K + E - 1) + T + P + 2A)}{N}}{\frac{K}{G(K + E - 1) + 1}}.$$

In what follows, we describe the data encoding procedure, then present the scheme in broad strokes. In the third subsection,

we present the scheme in detail, with an emphasis on the presentation of the first two rounds of the scheme for the sake of clarity. We show that the scheme has the stated rate, has E-data privacy, has T-function privacy, and is robust against any P stragglers and any A adversaries.

A. Data Encoding

Let us first describe the data encoding procedure, which follows the Lagrange encoding of Reed-Solomon codes as in Section II-B2. Let $\mathbf{x}_k \in \mathbb{F}_q^M$ for $k = 1, \ldots, K$ be the K data vectors, and let $\mathbf{t}_1, \ldots, \mathbf{t}_E \in \mathbb{F}_q^M$ be E i.i.d. uniform random vectors. Define $\mathbf{X} = [\mathbf{x}_1 \cdots \mathbf{x}_K \; \mathbf{t}_1 \cdots \mathbf{t}_E]$. Let $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_{K+E}) \in \mathbb{F}_q^{K+E}$ consist of $K + E$ distinct elements of $\mathbb{F}_q$. Define the interpolation polynomial $u_{\mathbf{X}}(z) = u_{\mathbf{X}, \boldsymbol{\beta}}(z)$ by the property

$$u_{\mathbf{X}}(\beta_k) = \mathbf{x}_k \qquad \text{for } k = 1, \ldots, K$$
$$u_{\mathbf{X}}(\beta_{K+e}) = \mathbf{t}_e \qquad \text{for } e = 1, \ldots, E.$$

By basic facts about polynomial interpolation, we have $\deg(u_{\mathbf{X}}(z)) \leq K + E - 1$. Note that $u_{\mathbf{X}}(z)$ is more accurately described as a vector of polynomials of length M, that is, $u_{\mathbf{X}}(z) \in \mathbb{F}_q[z]^M$. However, we continue to refer to it as a polynomial for simplicity. We now choose an evaluation vector $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_N) \in \mathbb{F}_q^N$ where the α_n are all distinct and non-zero, and set

$$\mathbf{y}_n = u_{\mathbf{X}}(\alpha_n)$$

which is then stored on server $n = 1, \ldots, N$. If we further have $\{\beta_k\}_{k \in [K]} \cap \{\alpha_n\}_{n \in [N]} = \varnothing$, then it is straightforward to show using Lemma 3 that we have E-data privacy. See [32, Section IV.C] for more details.

B. Basic Scheme Outline

To begin, define $N' \triangleq N - (P + 2A)$, and let

$$H \triangleq N' - (G(K + E - 1) + T), \quad L \triangleq G(K + E - 1) + 1.$$

Define B and S to be positive integers satisfying the equation

$$BL = HS.$$

The exact choice of B and S is not of crucial importance, so one can choose them to be minimal or simply set $S = L$ and $B = H$ for simplicity, which are the minimal solutions whenever $\gcd(L, H) = 1$.

The scheme will evaluate $\phi_b(\mathbf{x}_k)$ for B functions $\phi_b \in \mathcal{P}_G$. This will be accomplished by downloading all coefficients of $\gamma_b(z) \triangleq \phi_b(u_{\mathbf{X}}(z))$, and then evaluating the $\gamma_b(z)$ on the β_k. We have $\deg(\gamma_b(z)) \leq G(K+E-1)$, and thus L is the number of coefficients we need to download to completely determine a single $\gamma_b(z)$. As the individual coefficients of $\gamma_b(z)$ will also play a role in the scheme construction, we define $\gamma_{b\ell}$ by

$$\gamma_b(z) = \sum_{\ell=0}^{L-1} \gamma_{b\ell} z^{\ell}$$

The parameter H is defined so that the user will download H unique coefficients $\gamma_{b\ell}$ during each round of the scheme.

We define some $\zeta_b^{(s)}(z) \in \mathbb{F}_q[z, z^{-1}]$ for $b = 1, \ldots, B$ and $s = 1, \ldots, S$, whose precise nature will be made clear in the scheme construction. We also choose[1] $\psi_t^{(s)} \in \mathcal{P}_G$ to be i.i.d. uniform random elements, which are chosen anew during each round. Now set

$$\rho^{(s)} = \sum_{b=1}^{B} \zeta_b^{(s)}(z)\phi_b + z^H \sum_{t=1}^{T} z^{t-1}\psi_t^{(s)}$$

Given our evaluation points $\alpha_n \in \mathbb{F}_q$, we construct *query functions* $\rho_n^{(s)}$, for $n = 1, \ldots, N$, the n'th of which is equal to $\rho^{(s)}$ evaluated at $z = \alpha_n$. That is,

$$\rho_n^{(s)} \triangleq \sum_{b=1}^{B} \zeta_b^{(s)}(\alpha_n)\phi_b + \alpha_n^H \sum_{t=1}^{T} \alpha_n^{t-1}\psi_t^{(s)} \in \mathcal{P}_G. \quad (8)$$

Note that the coefficients $\zeta_b^{(s)}(\alpha_n)$ and α_n^{H+t-1} in (8) are independent of the functions ϕ_b and $\psi_t^{(s)}$, and hence can be computed a priori. Here $\rho_n^{(s)}$ is transmitted to the n'th server during the s'th round, who responds with $\rho_n^{(s)}(\mathbf{y}_n)$. We define the s'th *response polynomial* $r^{(s)}(z) \in \mathbb{F}_q[z]$ to be

$$r^{(s)}(z) \triangleq \rho^{(s)}(u_{\mathbf{X}}(z))$$
$$= \sum_{b=1}^{B} \zeta_b^{(s)}(z)\phi_b(u_{\mathbf{X}}(z)) + z^H \sum_{t=1}^{T} z^{t-1}\psi_t^{(s)}(u_{\mathbf{X}}(z))$$
$$= \sum_{b=1}^{B} \zeta_b^{(s)}(z)\gamma_b(z) + z^H \delta^{(s)}(z)$$

where we define $\delta^{(s)}(z) \triangleq \sum_{t=1}^{T} z^{t-1}\psi_t^{(s)}(u_{\mathbf{X}}(z))$. Note that

$$z^H \delta^{(s)}(z) \in \text{span}\{z^H, \ldots, z^{H+G(K+E-1)+T-1}\}, \quad (9)$$

and thus the evaluation vector of $z^H \delta^{(s)}(z)$, which contains only randomness, lives in a subspace of $\mathbb{F}_q^N$ of dimension $G(K + E - 1) + T = N' - H$.

During the s'th round, the user observes the vector

$$ev(r^{(s)}(z)) + \boldsymbol{\epsilon}^{(s)}$$

where $\boldsymbol{\epsilon}^{(s)}$ is a vector which contains at most P erasure symbols, coming from the P stragglers, and at most A arbitrary elements of $\mathbb{F}_q$, coming from the A adversaries. During each round, this will allow the user to decode the coefficients of $1, z, \ldots, z^{H-1}$ in $r^{(s)}(z)$, which will come from the terms in $\sum_{b=1}^{B} \zeta_b^{(s)}(z)\gamma_b(z)$. The general expression for $r^{(s)}(z)$ will be of the form

$$r^{(s)}(z) = \underbrace{a_{-C}^{(s)}z^{-C} + \cdots + a_{-1}^{(s)}z^{-1}}_{a_i^{(s)} \text{ known from previous rounds}}$$
$$+ \underbrace{a_0^{(s)} + \cdots + a_{H-1}^{(s)}z^{H-1}}_{a_i^{(s)} \text{ decoded during } s\text{'th round}} + \underbrace{z^H \delta^{(s)}(z)}_{\text{noise}}$$

---

[1]As mentioned in Subsection II-F, one can restrict the attention to some subspace $\mathcal{P} \subseteq \mathcal{P}_G$ to reduce upload costs, in which case we must have $\phi_b \in \mathcal{P}$ for every $b \in [B]$ and $\psi_t^{(s)} \in \mathcal{P}$ for every s and t. The resulting query in (8) will be in $\mathcal{P}$ as a linear combination. The guaranteed privacy will be restricted to $\mathcal{P}$, i.e., an attacker will not be able to learn anything about the functions $\phi_1, \ldots, \phi_B$ other than them being in $\mathcal{P}$.

for some C which can depend on the round index s. The assumption that $\alpha_n \neq 0$ for all n guarantees that $ev(r^{(s)}(z))$ is a well-defined element of $\mathbb{F}_q^N$. Strictly speaking, $r^{(s)}(z)$ is not a polynomial as it will contain negative powers of z, but the distinction is not relevant to the scheme construction, and so we refer to it as a polynomial for simplicity.

During the s'th round, the decoding process roughly proceeds as follows. The user subtracts off the evaluation vectors of $a_{-C}^{(s)} z^{-C}, \ldots, a_{-1}^{(s)} z^{-1}$ from what they receive, and are left with an element of a Reed-Solomon code with parameters $[N, N', P+2A+1]$. This allows them to correct the P erasures and A errors in the vector $\epsilon^{(s)}$. After this decoding, they may obtain the elements $a_0^{(s)}, \ldots, a_{H-1}^{(s)}$ as the coefficients of the evaluation vectors of $1, z, \ldots, z^{H-1}$, which are linearly independent from the evaluation vector of $z^H \delta^{(s)}$.

The scheme construction will guarantee that the sets $A^{(s)} = \{a_i^{(s)} \mid i = 0, \ldots, H-1\}$ of coefficients decoded during round s each consist of H unique coefficients of the polynomials $\gamma_b(z)$. That is, $A^{(s)} \cap A^{(t)} = \varnothing$ for $s \neq t$, and since each has size H, the user will have decoded $HS = BL$ unique coefficients of the B polynomials $\gamma_b(z)$ at the end of the scheme. Since this is all of the coefficients of these polynomials, the user can reconstruct all $\gamma_b(z)$ entirely, and therefore compute

$$\phi_b(\mathbf{x}_k) = \phi_b(u_{\mathbf{X}}(\beta_k)) = \gamma_b(\beta_k)$$

for all $b = 1, \ldots, B$ and all $k = 1, \ldots, K$. The PC rate is therefore

$$R = \frac{KB}{NS} = \frac{\dfrac{N - (G(K+E-1)+T+P+2A)}{N}}{\dfrac{K}{G(K+E-1)+1}}.$$

as claimed.

C. Construction of $\zeta_b^{(s)}(z)$ and Decoding

It remains to define the functions $\zeta_b^{(s)}(z)$ and describe the decoding process. To begin, define the matrix

$$\zeta(z) = \left(\zeta_b^{(s)}(z) \right)_{\substack{1 \leq s \leq S \\ 1 \leq b \leq B}} \in \mathbb{F}_q[z, z^{-1}]^{S \times B}$$

and let $\zeta^{(s)}(z)$ be the s'th row of $\zeta(z)$. We start by defining integers Q_1 and U_1 by using Euclidean division to write

$$H = Q_1 L + U_1, \quad Q_1 = \lfloor H/L \rfloor \quad \text{and} \quad 0 \leq U_1 < L.$$

For the sake of presentation, we first define $\zeta^{(1)}(z)$ and $\zeta^{(2)}(z)$ and demonstrate the decoding process during the first two rounds of the scheme. We then give a recursive definition of $\zeta_b^{(s)}(z)$ which works for every $s = 2, \ldots, S$.

1) Round $s = 1$: Define $\zeta^{(1)}(z)$ to be

$$\zeta^{(1)}(z) = \begin{pmatrix} 1 & z^L & \cdots & z^{(Q_1-1)L} & z^{Q_1 L} & 0 & \cdots & 0 \end{pmatrix}$$

or equivalently,

$$\zeta_b^{(1)}(z) = \begin{cases} z^{(b-1)L} & 1 \leq b \leq Q_1 + 1 \\ 0 & Q_1 + 2 \leq b \leq B \end{cases}$$

The response polynomial $r^{(1)}(z)$ is then of the form

$$r^{(1)}(z) = \gamma_1(z) + z^L \gamma_2(z) + \cdots + z^{(Q_1-1)L} \gamma_{Q_1}(z) + z^{Q_1 L} \gamma_{Q_1+1}(z) + z^H \delta^{(1)}(z)$$

First, note that

$$\begin{aligned} \deg(z^{Q_1 L} \gamma_{Q_1+1}(z)) &\leq Q_1 L + G(K+E-1) \\ &\leq H + G(K+E-1) \leq N' - 1, \end{aligned}$$

that alongside (9) implies that the evaluation vector of $r^{(1)}(z)$ lives in $\mathcal{RS}_{N'}(\alpha)$, which is an MDS code with parameters $[N, N', P+2A+1]$ and can thus correct P erasures and A errors contained in the vector $\epsilon^{(s)}$. After the erasure/error correction, the user is left with the evaluation vector of $r^{(1)}(z)$, which determines $r^{(1)}(z)$ completely. Since $\deg(\gamma_b(z)) \leq L$ for all b, the coefficients of $1, z, \ldots, z^{H-1}$ of the above allow the user to decode all of the coefficients of $\gamma_1(z), \ldots, \gamma_{Q_1}(z)$, and the first U_1 coefficients of $\gamma_{Q_1+1}(z)$. That is, $A^{(1)} = \{\gamma_{10}, \ldots, \gamma_{Q_1+1, U_1-1}\}$.

2) Round $s = 2$: Let Q_2 be the largest integer such that $-U_1 + (Q_2 - Q_1)L \leq H$, and define $\zeta^{(2)}(z)$ by

$$\zeta_b^{(2)}(z) = \begin{cases} 0 & 1 \leq b \leq Q_1 \\ z^{-U_1 + (b-(Q_1+1))L} & Q_1 + 1 \leq b \leq Q_2 + 1 \\ 0 & Q_2 + 2 \leq b \leq B. \end{cases}$$

The response polynomial $r^{(2)}(z)$ is then of the form

$$\begin{aligned} r^{(2)}(z) = \ & z^{-U_1} \gamma_{Q_1+1}(z) + z^{-U_1+L} \gamma_{Q_1+2}(z) + \cdots \\ & + z^{-U_1+(Q_2-Q_1)L} \gamma_{Q_2+1}(z) + z^H \delta^{(2)}(z) \\ = \ & \underbrace{\gamma_{Q_1+1,0} z^{-U_1} + \cdots + \gamma_{Q_1+1, U_1-1} z^{-1}}_{\text{coefficients known from round } s = 1} + \\ & \underbrace{\gamma_{Q_1+1, U_1} + \cdots + \gamma_{Q_2+1, 0} z^{-U_1 + (Q_2-Q_1)L} + \cdots + \gamma_{Q_2+1, U_2-1} z^{-U_1+(Q_2-Q_1)L + U_2 - 1}}_{H \text{ coefficients decoded during round } s = 2} \\ & + \underbrace{z^H \eta^{(2)}(z)}_{\text{noise}}, \end{aligned}$$

where U_2 is chosen so that $-U_1 + (Q_2 - Q_1)L + U_2 - 1 = H - 1$, that is, so that the first U_2 coefficients of $\gamma_{Q_2+1}(z)$ are downloaded during round 2. Here $\eta^{(2)}(z)$ is a polynomial which incorporates the noise polynomial $\delta^{(2)}(z)$ and the monomials of $\gamma_{Q_2+1}(z)$ whose coefficients are *not* downloaded during the s'th round. One sees easily that

$$\begin{aligned} \deg\left(z^{-U_1+(Q_2-Q_1)L} \gamma_{Q_2+1}(z) \right) &\leq H + G(K+E-1) \\ &\leq N' - 1 \end{aligned}$$

from which we see that $\deg(z^H \eta^{(2)}(z)) \leq N' - 1$ as well.

The coefficients $\gamma_{Q_1+1,0}, \ldots, \gamma_{Q_1+1, U_1-1}$ are exactly the first U_1 coefficients of $\gamma_{Q_1}(z)$, and are known to the user from round $s = 1$. The user can therefore subtract the vectors $\gamma_{Q_1+1,0} ev(z^{-U_1}), \ldots, \gamma_{Q_1+1, U_1-1} ev(z^{-1})$ from the response vector in round 2. After this subtraction, the user is left with the vector

$$ev\left(\begin{array}{c}\gamma_{Q_1+1,U_1}+\cdots+\\ \gamma_{Q_2+1,U_2-1}z^{-U_1+(Q_2-Q_1)L+U_2-1}+\\ z^H\eta^{(2)}(z)\end{array}\right)+\epsilon^{(2)}$$

$$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad}_{\in\mathcal{RS}_{N'}(\boldsymbol{\alpha})}$$

where $\epsilon^{(2)}$ contains P erasure symbols and A non-zero elements of $\mathbb{F}_q$. Because $\mathcal{RS}_{N'}(\boldsymbol{\alpha})$ is an MDS code with parameters $[N,N',P+2A+1]$, the user can correct these erasures and errors. After this erasure/error correction, the user further decodes the coefficients $A^{(2)}=\{\gamma_{Q_1+1,U_1},\ldots,\gamma_{Q_2+1,U_2-1}\}$ as the coefficients of the evaluation vectors of $1,z,\ldots,z^{H-1}$.

3) Round $s=2,\ldots,S$: We generalize the construction of the $\zeta_b^{(2)}(z)$ to the following recursive construction of $\zeta_b^{(s)}$ for all $s=2,\ldots,S$. Having already defined U_{s-1} and Q_{s-1}, we define Q_s to be the maximal integer such that $-U_{s-1}+(Q_s-Q_{s-1})L\le H$, and then set

$$\zeta_b^{(s)}(z)=\begin{cases}0 & 1\le b\le Q_{s-1}\\ z^{-U_{s-1}+(b-(Q_{s-1}+1))L} & Q_{s-1}+1\le b\le Q_s+1\\ 0 & Q_s+2\le b\le B.\end{cases}$$

Lastly, define U_s by the equation $-U_{s-1}+(Q_s-Q_{s-1})L+U_s-1=H-1$. In round $s-1$, the user decoded the first U_{s-1} coefficients of $\gamma_{Q_{s-1}}(z)$. The response polynomial $r^{(s)}(z)$ can therefore be written as

$$r^{(s)}(z)=\underbrace{\gamma_{Q_{s-1},0}z^{-U_{s-1}}+\cdots+\gamma_{Q_{s-1},U_{s-1}-1}z^{-1}}_{\text{coefficients known from round }s-1}$$

$$+\underbrace{\gamma_{Q_{s-1},U_{s-1}}+\cdots+\gamma_{Q_s,0}z^{-U_{s-1}+(Q_s-Q_{s-1})L}+}_{}$$
$$\underbrace{\cdots+\gamma_{Q_s,U_s-1}z^{-U_{s-1}+(Q_s-Q_{s-1})L+U_s-1}+}_{H\text{ coefficients decoded during round }s}$$

$$\underbrace{z^H\eta^{(s)}(z)}_{\text{noise}}.$$

As in rounds $s=1,2$, the user first subtracts the vectors corresponding to negative powers of z, the coefficients of which have been decoded in round $s-1$. As in rounds $s=1,2$, one shows easily that $\deg(z^H\eta^{(s)}(z))\le N'-1$, hence what is left after this subtraction is an element of $\mathcal{RS}_{N'}(\boldsymbol{\alpha})$ plus a vector $\epsilon^{(s)}$ with P erasure symbols and A non-zero elements of $\mathbb{F}_q$. After performing erasure/error correction in this Reed-Solomon code, the user may now decode the set $A^{(s)}=\{\gamma_{Q_{s-1},U_{s-1}},\ldots,\gamma_{Q_s,U_s-1}\}$ as the coefficients of the evaluation vectors of $1,z,\ldots,z^{H-1}$. We see that during the s'th round, the user obtains all coefficients of Q_s-Q_{s-1} distinct polynomials $\gamma_b(z)$, and U_s coefficients of a single additional polynomial.

After S rounds, the user has decoded all of the sets $A^{(s)}$ for $s=1,\ldots,S$. Since they are all clearly disjoint and of size H, the user has obtained $HS=BL$ coefficients of the $\gamma_b(z)$, which suffices to reconstruct all of these polynomials. The scheme construction is therefore complete.

D. Proof of T-function Privacy

Since a function in $\mathcal{P}_G$ can be seen as a vector over $\mathbb{F}_q$, it follows from (8) that the query functions are of the following

form.

$$\begin{bmatrix}\rho_1^{(s)}\\\vdots\\\rho_N^{(s)}\end{bmatrix}^\top=\begin{bmatrix}\phi_1\\\vdots\\\phi_B\\\psi_1^{(s)}\\\vdots\\\psi_T^{(s)}\end{bmatrix}^\top\begin{pmatrix}\zeta_1^{(s)}(\alpha_1)&\cdots&\zeta_1^{(s)}(\alpha_N)\\\vdots&\ddots&\vdots\\\zeta_B^{(s)}(\alpha_1)&\cdots&\zeta_B^{(s)}(\alpha_N)\\\alpha_1^H&\cdots&\alpha_N^H\\\alpha_1^{H+1}&\cdots&\alpha_N^{H+1}\\\vdots&\ddots&\vdots\\\alpha_1^{H+T-1}&\cdots&\alpha_N^{H+T-1}\end{pmatrix}.$$

It is readily verified by Lemma 3 that $I(\phi_1,\ldots,\phi_B;\rho_{n_1}^{(s)},\ldots,\rho_{n_T}^{(s)})=0$ for any T-subset $\mathcal{T}$ of $[N]$, and any single $s=1,\ldots,S$. Since the $\psi_t^{(s)}$ are chosen independently between rounds, the result follows.

IV. EXAMPLE

In this section we illustrate some of the subtleties of the scheme construction through a detailed example for the parameters $N=14$, $K=2$, $G=2$, $T=1$, $P=1$, $A=1$, $E=2$, which achieves a PC rate of

$$R=\frac{N-(G(K+E-1)+T+P+2A)}{N}.$$
$$\frac{K}{G(K+E-1)+1}=\frac{4}{49}.$$

The auxiliary parameters are given by $N'=11$, $H=4$, $L=7$, $B=4$, and $S=7$. The data encoding operates by choosing two i.i.d. uniform random vectors $\mathbf{t}_1,\mathbf{t}_2$ and distinct elements $\beta_1,\ldots,\beta_4\in\mathbb{F}_q$, and defining the interpolating polynomial $u_{\mathbf{X}}(z)$ by the conditions

$$u_{\mathbf{X}}(\beta_1)=\mathbf{x}_1,\ u_{\mathbf{X}}(\beta_2)=\mathbf{x}_2,\ u_{\mathbf{X}}(\beta_3)=\mathbf{t}_1,\ u_{\mathbf{X}}(\beta_4)=\mathbf{t}_2.$$

We have $\deg(u_{\mathbf{X}}(z))\le 3$. Choosing some $\alpha_1,\ldots,\alpha_{14}\in\mathbb{F}_q$ to all be non-zero and distinct from β_1 and β_2, we encode by setting $\mathbf{y}_n=u_{\mathbf{X}}(\alpha_n)$ for all $n=1,\ldots,14$.

The scheme will compute, over $S=7$ rounds, the function values $\phi_b(\mathbf{x}_k)$ for $B=4$ quadratic functions ϕ_b. The polynomials $\gamma_b(z)=\phi_b(u_{\mathbf{X}}(z))$ whose coefficients will be downloaded have degree $\deg(\gamma_b(z))\le 6$, and hence are each completely determined by $L=7$ coefficients.

The query functions $\rho_n^{(s)}$ take the form

$$\rho_n^{(s)}=\sum_{b=1}^B\zeta_b^{(s)}(\alpha_n)\phi_b+\alpha_n^4\psi_1^{(s)}$$

where $\psi_1^{(s)}$ is a uniform random element of $\mathcal{P}_2$. The functions $\zeta_b^{(s)}(z)$ are given by

$$\zeta(z)=\left(\zeta_b^{(s)}(z)\right)_{\substack{1\le s\le 7\\1\le b\le 4}}=\begin{pmatrix}1&0&0&0\\z^{-4}&z^3&0&0\\0&z^{-1}&0&0\\0&z^{-5}&z^2&0\\0&0&z^{-2}&0\\0&0&z^{-6}&z\\0&0&0&z^{-3}\end{pmatrix}$$

from which we can compute the response polynomials $r^{(s)}(z)$ to be

$$r^{(1)}(z) = \gamma_{10} + \gamma_{11}z + \gamma_{12}z^2 + \gamma_{13}z^3 + z^4\eta^{(1)}(z)$$
$$r^{(2)}(z) = \gamma_{10}z^{-4} + \gamma_{11}z^{-3} + \gamma_{12}z^{-2} + \gamma_{13}z^{-1} + \gamma_{14} +$$
$$\gamma_{15}z + \gamma_{16}z^2 + \gamma_{20}z^3 + z^4\eta^{(2)}(z)$$
$$r^{(3)}(z) = \gamma_{20}z^{-1} + \gamma_{21} + \gamma_{22}z + \gamma_{23}z^2 +$$
$$\gamma_{24}z^3 + z^4\eta^{(2)}(z)$$
$$r^{(4)}(z) = \gamma_{20}z^{-5} + \gamma_{21}z^{-4} + \gamma_{22}z^{-3} + \gamma_{23}z^{-2} +$$
$$\gamma_{24}z^{-1} + \gamma_{25} + \gamma_{26}z + \gamma_{30}z^2 +$$
$$\gamma_{31}z^3 + z^4\eta^{(4)}(z)$$
$$r^{(5)}(z) = \gamma_{30}z^{-2} + \gamma_{31}z^{-1} + \gamma_{32} + \gamma_{33}z + \gamma_{34}z^2 +$$
$$\gamma_{35}z^3 + z^4\eta^{(5)}(z)$$
$$r^{(6)}(z) = \gamma_{30}z^{-6} + \gamma_{31}z^{-5} + \gamma_{32}z^{-4} + \gamma_{33}z^{-3} +$$
$$\gamma_{34}z^{-2} + \gamma_{35}z^{-1} + \gamma_{36} + \gamma_{40}z + \gamma_{41}z^2 +$$
$$\gamma_{42}z^3 + z^4\eta^{(6)}(z)$$
$$r^{(7)}(z) = \gamma_{40}z^{-3} + \gamma_{41}z^{-2} + \gamma_{42}z^{-1} + \gamma_{43} + \gamma_{44}z +$$
$$\gamma_{45}z^2 + \gamma_{46}z^3 + z^4\eta^{(7)}(z)$$

In the first round, the polynomial $r^{(1)}(z)$ has degree $\leq N' - 1 = 10$, hence its evaluation vector lives in a Reed-Solomon code with parameters $[14, 11, 4]$, allowing the user to correct the erasure and error from the straggler and adversary. The user then decodes $\gamma_{10}, \ldots, \gamma_{13}$. In all successive rounds, it is clear from the above expressions that all coefficients of negative powers of z are decoded in previous rounds, and can thus be subtracted from the response vector. After this subtraction, what is left is in a Reed-Solomon code with parameters $[14, 11, 4]$, which can correct $P = 1$ erasure and $A = 1$ error. The user performs this erasure/error correction, and then decodes the coefficients of $1, z, z^2, z^3$ in every round.

Having decoded all $\gamma_{b\ell}$ for $b = 1, \ldots, 4$ and $\ell = 0, \ldots, 6$, the user can compute $\phi_b(\mathbf{x}_k) = \gamma_b(\beta_k)$ for $b = 1, \ldots, 4$ and $k = 1, 2$. The user has computed $KB = 8$ function evaluations while downloading $NS = 98$ elements of $\mathbb{F}_q$ total, hence the rate of the scheme is

$$R = \frac{KB}{NS} = \frac{8}{98} = \frac{4}{49}$$

as claimed.

V. IMPROVED PC RATE FOR SYSTEMATIC ENCODING

In this section it is shown that whenever $P = A = E = 0$ and $N \leq 2G(K-1) + T + 1$, a larger PC rate is achievable by systematic encoding. The choice of the specific systematic code is not important, so one can simply assume that the data is encoded with systematic Lagrange encoding (i.e., where $\beta_i = \alpha_i$ for all $i \in [K]$ in (4)). The scheme in this section appeared in [12].

First, let the storage code be $\mathcal{C} = \mathcal{RS}_K(\boldsymbol{\alpha})$. Define $\mathcal{D} = \mathcal{RS}_T(\boldsymbol{\alpha})$, called the *retrieval code* of the scheme, and let $\mathcal{E} \triangleq \mathcal{C}^{\star G} \star \mathcal{D}$. It follows from Lemma 1 and Proposition 2 that $\mathcal{E} = \mathcal{RS}_{G(K-1)+T}(\boldsymbol{\alpha})$. Denote the minimum distance of $\mathcal{E}$ by $D_{\mathcal{E}}$, and let $F \triangleq D_{\mathcal{E}} - 1 = N - G(K-1) - T$. Next, let B and S

be the minimal integers that satisfy $\min\{F, K\}S = KB$ (i.e., $S = \frac{\text{lcm}(K,\min\{F,K\})}{\min\{F,K\}}$ and $B = \frac{\text{lcm}(K,\min\{F,K\})}{K}$), and choose subsets of $[K] -$

$$\mathcal{I}^{(1)} = \mathcal{I}^{(1,1)} \cup \mathcal{I}^{(1,2)} \cup \ldots \cup \mathcal{I}^{(1,B)}$$
$$\mathcal{I}^{(2)} = \mathcal{I}^{(2,1)} \cup \mathcal{I}^{(2,2)} \cup \ldots \cup \mathcal{I}^{(2,B)}$$
$$\vdots$$
$$\mathcal{I}^{(S)} = \mathcal{I}^{(S,1)} \cup \mathcal{I}^{(S,2)} \cup \ldots \cup \mathcal{I}^{(S,B)}, \qquad (10)$$

such that in each row the sets in the union are pairwise disjoint, such that $|\mathcal{I}^{(s)}| = \min\{F, K\}$ for every $s \in [S]$, and such that for every $b \in [B]$ we have that $\cup_{s=1}^S \mathcal{I}^{(s,b)} = [K]$. The process of choosing the sets $\{\mathcal{I}^{(s,b)}\}_{(s,b)\in[S]\times[B]}$ in (10) is very simple, and similar constructions have appeared in [8, 12]. For our current purposes, the choice of these sets is best illustrated by the following example.

Example 4. *Assume that $F = 4$ and $K = 6$, which implies that $S = 3$ and $B = 2$. Consider the following matrix*

$$\begin{pmatrix} 1 & 1 & 1 & 1 & & \\ 2 & 2 & & & 1 & 1 \\ & & 2 & 2 & 2 & 2 \end{pmatrix},$$

which naturally corresponds to the sets

$$\mathcal{I}^{(1,1)} = \{1,2,3,4\} \qquad \mathcal{I}^{(1,2)} = \varnothing$$
$$\mathcal{I}^{(2,1)} = \{5,6\} \qquad \mathcal{I}^{(2,2)} = \{1,2\}$$
$$\mathcal{I}^{(3,1)} = \varnothing \qquad \mathcal{I}^{(3,2)} = \{3,4,5,6\}.$$

One verifies immediately that the sets $\{\mathcal{I}^{(s,b)}\}_{(s,b)\in[S]\times[B]}$ have the stated properties.

To continue with the scheme construction, let Q be the dimension of $\mathcal{P}_G$, and fix a basis $\psi^1, \ldots, \psi^Q$ of $\mathcal{P}_G$. In round $s \in [S]$, the user employs fresh randomness in order to choose codewords $\mathbf{d}^1, \ldots, \mathbf{d}^Q \in \mathcal{D}$, and defines

$$\psi_n = \mathbf{d}^1(n) \cdot \psi^1 + \cdots + \mathbf{d}^Q(n) \cdot \psi^Q \qquad (11)$$

for every $n \in [N]$, where $\mathbf{d}^i(n)$ is the n'th entry of $\mathbf{d}^i$. Then, the user defines the queries

$$\rho_n^{(s)} = \begin{cases} \psi_n + \phi_b & \text{if there exists } b \in [B] \\ & \text{such that } n \in \mathcal{I}^{(s,b)} \\ \psi_n & \text{else} \end{cases}.$$

Intuitively, the set $\mathcal{I}^{(s)} \subseteq [N]$ contains the indices of the $\mathbf{y}_i$'s for which some $\phi_b(\mathbf{y}_i), b \in [B]$ is retrieved during the s'th round. In addition, for every $b \in [B]$, the set $\mathcal{I}^{(s,b)}$ contains the indices of the $\mathbf{y}_i$'s such that $\phi_b(\mathbf{y}_i)$ is retrieved during the s'th round; formally, we have that

$$\mathcal{I}^{(s,b)} \triangleq \{t \in [N] \mid \phi_b(\mathbf{y}_t) \text{ is retrieved in round } s\}.$$

The T-function-privacy relies on the following lemma, analogues of which were fully proved in [12, Theorem 1] and in [8, Theorem 8], among others.

Lemma 5. *If $\mathcal{D}$ is an MDS code then the scheme is T-function-private.*

In turn, the response vector is of the form

$$\boldsymbol{\rho}^{(s)}(\mathbf{Y}) = \underbrace{[\psi_1(\mathbf{y}_1), \ldots, \psi_N(\mathbf{y}_N)]}_{\triangleq \boldsymbol{\psi}(\mathbf{Y})} + \mathbf{v}^{(s)}, \qquad (12)$$

where

$$(\mathbf{v}^{(s)})_n = \begin{cases} \phi_b(\mathbf{x}_n) & \text{if there exists } b \in [B] \\ & \text{such that } n \in \mathcal{I}^{(s,b)} \\ 0 & \text{else} \end{cases},$$

and notice that $\phi(\mathbf{y}_n) = \phi_b(\mathbf{x}_n)$ merely since $\mathcal{I}^{(s)} \subseteq [K]$, and since the encoding is systematic.

Lemma 6. *The codeword* $\boldsymbol{\psi}(Y) \triangleq [\psi_1(\mathbf{y}_1), \ldots, \psi_N(\mathbf{y}_N)]$ *is in* $\mathcal{E}$.

Proof. According to (11), one can readily verify that

$$[\psi_1(\mathbf{y}_1), \ldots, \psi_N(\mathbf{y}_n)] =$$
$$\left[\sum_{j=1}^{Q} \mathbf{d}^j(1) \cdot \psi^j(\mathbf{y}_1), \ldots, \sum_{j=1}^{Q} \mathbf{d}^j(N) \cdot \psi^j(\mathbf{y}_N) \right]$$
$$= \sum_{j=1}^{Q} \left[\mathbf{d}^j(1) \cdot \psi^j(\mathbf{y}_1), \ldots, \mathbf{d}^j(N) \cdot \psi^j(\mathbf{y}_N) \right]$$
$$= \sum_{j=1}^{Q} [\psi^j(\mathbf{y}_1), \ldots, \psi^j(\mathbf{y}_N)] \star \mathbf{d}^j,$$

and hence it remains to prove that $[\psi^j(\mathbf{y}_1), \ldots, \psi^j(\mathbf{y}_N)] \in \mathcal{C}^{\star G}$ for every $j \in [Q]$. To this end, assume that the chosen basis $\psi^1, \ldots, \psi^Q$ consists only of monomials, and let $\psi^j = X_1^{a_1} X_2^{a_2} \cdots X_M^{a_M}$ for some $j \in [Q]$. Denoting $\mathbf{y}_n \triangleq [y_{n,1}, y_{n,2}, \ldots, y_{n,M}]$ for every $n \in [N]$, which implies that $[y_{1,m}, y_{2,m}, \ldots, y_{N,m}] \in \mathcal{C}$ for every $m \in [M]$, implies that

$$[\psi^j(\mathbf{y}_1), \ldots, \psi^j(\mathbf{y}_N)] =$$
$$\left[\prod_{m=1}^{M} y_{1,m}^{a_m}, \prod_{m=1}^{M} y_{2,m}^{a_m}, \ldots, \prod_{m=1}^{M} y_{N,m}^{a_m} \right]$$
$$= [y_{1,1}, y_{2,1}, \ldots, y_{N,1}]^{\star a_1} \star \ldots \star$$
$$[y_{1,M}, y_{2,M}, \ldots, y_{N,M}]^{\star a_M}$$
$$\in \mathcal{C}^{\star a_1} \star \ldots \star \mathcal{C}^{\star a_M} \subseteq \mathcal{C}^{\star G}. \quad \square$$

Since $\boldsymbol{\psi}(\mathbf{Y})$ is a codeword in $\mathcal{E}$, and since the Hamming weight of $\mathbf{v}^{(s)}$ is at most $|\mathcal{I}^{(s)}| \leq F = D_{\mathcal{E}} - 1$, it follows that the vector $\mathbf{v}^{(s)}$ can be extracted from $\boldsymbol{\rho}^{(s)}(\mathbf{Y})$ by an error correction algorithm for $\mathcal{E}$. Therefore, according to the definition of the sets $\{\mathcal{I}^{(s,b)}\}_{(s,b) \in [S] \times [B]}$, it follows that the user can retrieve $\{\phi_b(\mathbf{x}_k)\}_{(k,b) \in [K] \times [B]}$, which concludes the correctness of the scheme, whose PC rate is

$$\frac{KB}{NS} = \frac{\min\{F, K\}}{N}. \qquad (13)$$

Recall that for the parameters $P = A = E = 0$, the PC rate of the general scheme (1) is

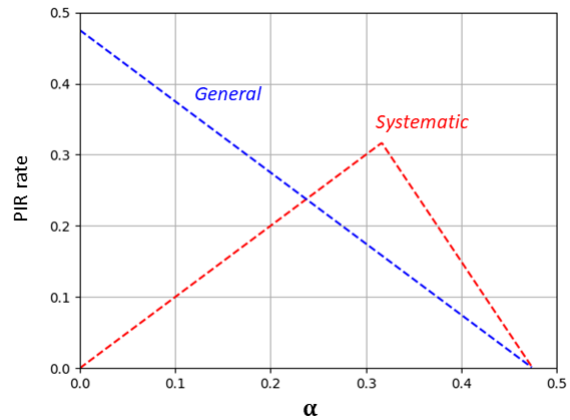$$\frac{N - G(K - 1) - T}{N} \cdot \frac{K}{G(K - 1) + 1}. \qquad (14)$$

Fig. 1. Asymptotic comparison of PC rates for various values of $\alpha = K/N$ between the general scheme (Subsection III) and the systematic one (Subsection V) for $G = 2$ and $\beta = T/N = 0.05$.

Therefore, in order to compare (13) with (14), notice that if $N > 2G(K - 1) + T + 1$, then since $G \geq 1$, it follows that

$$F \geq G(K - 1) + 1 \geq K, \qquad (15)$$

and thus (13) reduces to K/N. Moreover, (15) also implies that

$$\frac{F}{N} \cdot \frac{K}{G(K - 1) + 1} \geq \frac{K}{N},$$

and hence for this parameter regime we have that (14) is superior to (13). Conversely, if $N \leq 2G(K - 1) + T + 1$, then $F \leq G(K - 1) + 1$, and therefore

$$\frac{F}{N} \cdot \frac{K}{G(K - 1) + 1} \leq \frac{K}{N}.$$

Also, since $K \leq G(K - 1) + 1$, we have that

$$\frac{F}{N} \cdot \frac{K}{G(K - 1) + 1} \leq \frac{F}{N},$$

and hence

$$\frac{F}{N} \cdot \frac{K}{G(K - 1) + 1} \leq \frac{\min\{F, K\}}{N},$$

which implies the superiority of the systematic scheme for $N \leq 2G(K - 1) + T + 1$.

To compare (13) and (14) asymptotically, assume that N, K, and T approach infinity, and yet $\alpha \triangleq K/N$ and $\beta \triangleq T/N$ remain constants. The resulting PC rates are $\frac{1 - \beta}{G} - \alpha$ for the general scheme, and $\min\{1 - \alpha G - \beta, \alpha\}$ for the systematic one, and the necessary condition $N \geq G(K - 1) + T + 1$ translates to $\alpha G + \beta \leq 1$. An example for $G = 2$ and $\beta = 0.05$ is given in Figure 1.

VI. DISCUSSION AND OPEN PROBLEMS

In this paper we leveraged a recent notion from coded computing and obtained a robust private computing scheme for computing polynomials of arbitrary degree on encoded data, that can also guarantee privacy of the data. The scheme also

accounts for servers which are stragglers or adversaries, and is robust against servers colluding to attempt to deduce the identity of the functions to be evaluated. Many open questions remain, the most prominent of which is the capacity question, i.e., constructing schemes for this scenario with provably optimal rate. However, as mentioned in Section I, the capacity question of PIR is largely open, even without extending its scope to private computation. Thus, proving optimality is likely to be very difficult, even in restricted cases where some of the system parameters are zero. Another possible research direction is coming up with a proper notion of perfect privacy for *real-valued* computations, which applies directly without embedding into a finite field.

ACKNOWLEDGEMENTS

REFERENCES

[1] M. A. Attia and R. Tandon, "Approximately Optimal Distributed Data Shuffling," *IEEE International Symposium on Information Theory* (ISIT), pp. 721–725, 2018.

[2] K. Banawan and S. Ulukus, "The Capacity of Private Information Retrieval From Coded Databases," in *IEEE Transactions on Information Theory*, vol. 64, no. 3, pp. 1945–1956, 2018.

[3] K. Banawan and S. Ulukus, "The Capacity of Private Information Retrieval from Byzantine and Colluding Databases," in *IEEE Transactions on Information Theory*, vol. 65, no. 2, pp. 1206–1219, 2018.

[4] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan. "Private Information Retrieval," *Journal of the ACM*, vol. 45, no. 6, pp. 965–981, 1998.

[5] J. Chung, K. Lee, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran, "UberShuffle: Communication-efficient Data Shuffling for SGD via Coding Theory," *Advances in Neural Information Processing Systems* (NIPS), 2017.

[6] R. G.L. D'Oliveira, S. El Rouayheb, "One-Shot PIR: Refinement and Lifting," *arxiv:1810.05719*, [cs.IT], 2018.

[7] S. Dutta, V. Cadambe, and P. Grover, "Short-dot: Computing large linear transforms distributedly using coded short dot products," *Advances In Neural Information Processing Systems* (NIPS), pp. 2100–2108, 2016.

[8] R. Freij-Hollanti, O. W. Gnilke, C. Hollanti, and D. Karpuk, "Private Information Retrieval from Coded Databases with Colluding Servers," in *SIAM Journal on Applied Algebra and Geometry*, vol. 1, no. 1, pp. 647–664, Nov. 2017.

[9] R. Freij-Hollanti, O. W. Gnilke, C. Hollanti, A. Horlemann-Trautmann, D. Karpuk and I. Kubjas, "*t*-Private Information Retrieval Schemes Using Transitive Codes," *IEEE Transactions on Information Theory*, to appear.

[10] W. Gasarch. "A Survey on Private Information Retrieval," *Bulletin of the EATCS*, 2004.

[11] Z. Jia, H. Sun, and S. A. Jafar, "Cross Subspace Alignment and the Asymptotic Capacity of X-Secure T-Private Information Retrieval," *arXiv:1808.07457*, [cs.IT], 2018.

[12] D. Karpuk, "Private Computation of Systematically Encoded Data with Colluding Servers," *IEEE International Symposium on Information Theory* (ISIT), pp. 2112–2116, 2018.

[13] S. Kumar, H.-Y. Lin, E. Rosnes, A. Graell i Amat, "Achieving Maximum Distance Separable Private Information Retrieval Capacity With Linear Codes," *arxiv:1712.03898*, 2018.

[14] S. Li, M. A. Maddah-Ali, Q. Yu, A. S. Avestimehr, "A Fundamental Tradeoff Between Computation and Communication in Distributed Computing," *IEEE Transactions on Information Theory*, vol. 64, no. 1, pp. 109–128, 2018.

[15] S. Li, S. M. M. Kalan, A. S. Avestimehr, and M. Soltanolkotabi, "Near-Optimal Straggler Mitigation for Distributed Gradient Methods," *IEEE International Parallel and Distributed Processing Symposium Workshops* (IPDPSW), pp. 857–866, 2018.

[16] M. Mirmohseni, M. A. Maddah-Ali, "Private Function Retrieval," *arXiv:1711.04677*, [cs.IT], 2018.

[17] S. A. Obead and J. Kliewer, "Achievable Rate of Private Function Retrieval from MDS Coded Databases," *IEEE International Symposium on Information Theory* (ISIT), pp. 2117–2121, 2018.

[18] S. A. Obead, H.-Y. Lin, E. Rosnes, J. Kliewer, "Capacity of Private Linear Computation for Coded Databases", *arxiv:1810.04230*, [cs.IT], 2018.

[19] R. Ostrovsky and W. E. Skeith, "A Survey of Single-Database Private Information Retrieval: Techniques and applications," *International Workshop on Public Key Cryptography*, pp. 393–411, 2007.

[20] N. Raviv, I. Tamo, R. Tandon, and A. G. Dimakis, "Gradient Coding from Cyclic MDS Codes and Expander Graphs," *International Conference on Machine Learning* (ICML), 2018.

[21] A. Shamir, "How to Share a Secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[22] J. So, B. Guler, A. S. Avestimehr, and P. Mohassel, "CodedPrivateML: A Fast and Privacy-Preserving Framework for Distributed Machine Learning," *arXiv:1902.00641*, 2019.

[23] H. Sun and S. A. Jafar, "The Capacity of Private Information Retrieval," in *IEEE Transactions on Information Theory*, vol. 63, no. 7, pp. 4075–4088, 2017.

[24] H. Sun and S. A. Jafar, "The Capacity of Robust Private Information Retrieval With Colluding Databases," in *IEEE Transactions on Information Theory*, vol. 64, no. 4, pp. 2361–2370, 2018.

[25] H. Sun and S. A. Jafar, "The Capacity of Private Computation," *arXiv:1710.11098*, 2017.

[26] R. Tajeddine, O. W. Gnilke and S. El Rouayheb, "Private Information Retrieval From MDS Coded Data in

Distributed Storage Systems," in *IEEE Transactions on Information Theory*, vol. 64, no. 11, pp. 7081–7093, 2018.

[27] R. Tajeddine, O. W. Gnilke, D. Karpuk, R. Freij-Hollanti, and C. Hollanti, "Private Information Retrieval from Coded Storage Systems with Colluding, Byzantine, and Unresponsive Servers," *arXiv:1806.08006*, [cs.IT], 2018.

[28] R. Tandon, Q. Lei, A. G. Dimakis, and N. Karampatziakis, "Gradient Coding: Avoiding Stragglers in Distributed Learning," *International Conference on Machine Learning* (ICML), pp. 3368–3376, 2017.

[29] S. Yekhanin. "Private Information Retrieval," *Communications of the ACM*, vol. 53, no. 4, pp. 68–73, 2010.

[30] S. Yekhanin, "Locally Decodable Codes and Private Information Retrieval Schemes," *PhD thesis (MIT)*, 2007.

[31] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Polynomial codes: an Optimal Design for High-Dimensional Coded Matrix Multiplication," *Advances in Neural Information Processing Systems* (NIPS), pp. 4403–4413, 2017.

[32] Q. Yu, N. Raviv, J. So, and A. S. Avestimehr, "Lagrange Coded Computing: Optimal Design for Resiliency, Security and Privacy," *arXiv:1806.00939*, 2018.