Private Polynomial Function Computation for Noncolluding Coded Databases

Sarah A. Obead, *Graduate Student Member, IEEE*, Hsuan-Yin Lin, *Senior Member, IEEE*, Eirik Rosnes, *Senior Member, IEEE*, and Jörg Kliewer, *Senior Member, IEEE*

***Abstract*— We consider the problem of private polynomial computation (PPC) from a distributed storage system (DSS). In such setting a user wishes to compute a multivariate polynomial of degree at most g over f variables (or messages) stored in n noncolluding coded databases, i.e., databases storing data encoded with an $[n, k]$ linear storage code, while revealing no information about the desired polynomial evaluation to the databases. For a DSS setup where data is stored using linear storage codes, we derive an outer bound on the PPC rate, which is defined as the ratio of the (minimum) desired amount of information and the total amount of downloaded information, and construct two novel PPC schemes. In the first scheme, we consider Reed-Solomon coded databases with Lagrange encoding, which leverages ideas from recently proposed star-product private information retrieval and Lagrange coded computation. The second scheme considers the special case of coded databases with systematic Lagrange encoding. Both schemes yield improved rates, while asymptotically, as $f \to \infty$, the systematic scheme gives a significantly better computation retrieval rate compared to all known schemes up to some storage code rate that depends on the maximum degree of the candidate polynomials.**

***Index Terms*— Coded computation, information-theoretic privacy, private computation, private information retrieval, Reed-Solomon codes.**

I. Introduction

PRIVATE computation (PC) is a recently proposed generalization of the private information retrieval (PIR) problem. PIR is the problem of obtaining an arbitrary message stored in a public database without revealing the identity of the requested message to the database. The notion of PIR was studied in the computer science community for several decades (see, e.g., [2]–[4]) and has been revisited by information theorists with a focus on reducing the storage overhead while ensuring information-theoretic privacy guarantees and a low download cost (see, e.g., [5]–[12]). PC addresses the computation of functions over the stored messages [13]–[25], also denoted as private function retrieval [14]. In PC, a user intends to compute a function of the messages stored in a number of databases forming a distributed storage system (DSS). This function is to be kept private from the databases, as they may be under the control of an adversary. In this line of research, the main performance metric is the PC rate which is defined as the ratio of the (minimum) desired amount of information and the total amount of downloaded information. Accordingly, the PC capacity is defined as the maximum of all achievable PC rates over all possible PC protocols. In [13], [14], the notion of PC is introduced for noncolluding replicated databases. In these works, the capacity and achievable PC rates for privately computing a given *linear* function, called private linear computation (PLC), were derived as a function of the number of messages and the number of databases, respectively. Interestingly, the obtained PLC capacity is shown to be equal to the PIR capacity of [7]. Recently, the special case of PLC from a single server with side information available at the user is considered in [21], [22]. In these works, the authors derived the capacity of PLC with both coded and uncoded side information under two different privacy conditions on the identities of the messages involved in the desired computation. Finally, a different approach to PLC, private sequential function computation, is introduced in [23] and [24]. The authors consider the case where the user is interested in the computation of a function formed by a specific concatenation and combination of several linear functions while keeping the order of the combination private from a replication-based DSS. For the case of nonlinear function computations, private monomial computation (PMC) for replicated noncolluding databases was addressed in [25] where the PMC capacity for an asymptotically large field size and under a mild technical condition on the size of the base field was derived. The technical condition on the size of the base field can be shown to be satisfied for a sufficiently large base field. The PC capacity for the case where the candidate functions evaluations are the stored messages plus the evaluation of an arbitrary nonlinear function of them was derived in [19].

The extension of PC to *coded DSSs*, where the data is encoded by an $[n, k]$ linear code and then distributed over n storage nodes [26], is addressed in [16]–[18], [20]. In particular, in [17] we proposed a PLC scheme from

Manuscript received August 6, 2021; revised January 31, 2022; accepted March 14, 2022. Date of publication April 14, 2022; date of current version May 17, 2022. This work was supported in part by U.S. NSF under Grant CNS-1526547 and in part by the Research Council of Norway under Grant 240985/F20. An earlier version of this paper was presented at the IEEE International Symposium on Information Theory (ISIT), Paris, France, in July 2019 [1] [DOI: 10.1109/ISIT.2019.8849825]. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Matthieu R. Bloch. *(Corresponding author: Sarah A. Obead.)*

Sarah A. Obead and Jörg Kliewer are with the Helen and John C. Hartmann Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102 USA (e-mail: sao23@njit.edu; jkliewer@njit.edu).

Hsuan-Yin Lin and Eirik Rosnes are with Simula UiB, 5006 Bergen, Norway (e-mail: lin@simula.no; eirikrosnes@simula.no).

Digital Object Identifier 10.1109/TIFS.2022.3166667

maximum distance separable (MDS) coded DSSs. In [18], we settled the PLC capacity and provide a capacity-achieving scheme for a class of linear storage codes considered in [12]. The PLC capacity is shown to match the MDS-coded PIR capacity established in [8], referred to as the MDS-PIR capacity. Moreover, the scheme presented in [18] extends the optimal PIR scheme for coded DSSs in [12] and our PLC scheme from MDS-coded DSSs in [17], strictly generalizing the replication-based PLC schemes of [13], [14]. In [16], private polynomial computation (PPC) over t colluding and systematically coded databases is considered by generalizing the star-product PIR scheme of [10]. An alternative PPC approach from Reed-Solomon (RS) coded databases with Lagrange encoding, was recently proposed in [20]. For low code rates, the scheme improves on the PC rate of [16]. However, for the case of PC of nonlinear functions from noncolluding databases, capacity results for linearly-coded DSSs have not been addressed so far in the open literature to the best of our knowledge.

As a step in this direction, in this work, we consider PPC from noncolluding coded DSSs, propose two novel PPC schemes, and derive an outer bound on the PPC rate over all possible PPC protocols. Our contributions are outlined as follows.

- We adapt the converse proof of [18, Thm. 2] to the coded PPC problem and derive an outer bound on the PPC rate from a DSS encoded with a class of linear storage codes known as MDS-PIR capacity-achieving codes [12] (see Theorem 1).
- In [20], the authors were mainly concerned with constructing PPC schemes with a focus on preserving privacy against colluding DSSs. We, on the other hand, aim our attention at providing PPC solutions that minimize the download cost and we focus on establishing the capacity of the PPC setup. Towards that aim, we propose two new PPC schemes from RS-coded DSSs (one for systematic encoding) by generalizing our previous work on a capacity-achieving PLC scheme in [18] and leveraging ideas from star-product PIR [10] and Lagrange coded computation [27]. Our schemes improve on the rates of the PPC schemes presented in [16], [20] (see Theorems 2 and 3). The systematic scheme is an improved version of the systematic scheme presented in [1].
- To demonstrate the performance of our proposed PPC schemes, numerical results are presented. We show that, compared to the schemes in [16], [20], both proposed PPC schemes yield a larger PC rate, i.e., lower download cost, when the number of messages is small. As the number of messages tends to infinity, the achievable rate of our RS-coded (nonsystematic) PPC scheme approaches the rate of [20] (see Corollary 1), while our systematic scheme outperforms all known schemes up to some storage code rate that depends on the maximum degree of the candidate polynomials (see Remark 4 and Corollary 2).

II. Preliminaries

A. Notation

We denote by $\mathbb{N}$ the set of all positive integers and let $\mathbb{N}_0 \triangleq \{0\} \cup \mathbb{N}$, $[a] \triangleq \{1, 2, \ldots, a\}$, and $[a : b] \triangleq \{a, a+1, \ldots, b\}$ for $a, b \in \mathbb{N}$, $a \leq b$. A random variable is denoted by a capital Roman letter, e.g., X, while its realization is denoted by the corresponding small Roman letter, e.g., x. Vectors are boldfaced, e.g., $\boldsymbol{X}$ denotes a random vector and $\boldsymbol{x}$ denotes a deterministic vector, respectively. The notation $\boldsymbol{X} \sim \boldsymbol{Y}$ is used to indicate that $\boldsymbol{X}$ and $\boldsymbol{Y}$ are identically distributed. Random matrices are represented by bold sans serif letters, e.g., $\mathbf{\mathsf{X}}$, where X represents its realization. In addition, sets are denoted by calligraphic uppercase letters, e.g., $\mathcal{X}$, and $\mathcal{X}^{\mathrm{c}}$ denotes the complement of a set $\mathcal{X}$ in a universe set. We denote a submatrix of X that is restricted in columns by the set $\mathcal{I}$ by $\mathsf{X}|_{\mathcal{I}}$. For a given index set $\mathcal{S}$, we also write $\mathbf{\mathsf{X}}^{\mathcal{S}}$ and $Y_{\mathcal{S}}$ to represent $\{\mathbf{\mathsf{X}}^{(v)} : v \in \mathcal{S}\}$ and $\{Y_j : j \in \mathcal{S}\}$, respectively. Furthermore, some constants and functions are also depicted by Greek letters or a special font, e.g., $\mathtt{X}$. The function $\mathrm{H}(X)$ represents the entropy of X, and $\mathrm{I}(X; Y)$ the mutual information between X and Y. The binomial coefficient of a over b, $a, b \in \mathbb{N}_0$, is denoted by $\binom{a}{b}$ where $\binom{a}{b} = 0$ if $a < b$. The notation $\lfloor \cdot \rfloor$ denotes the floor function.

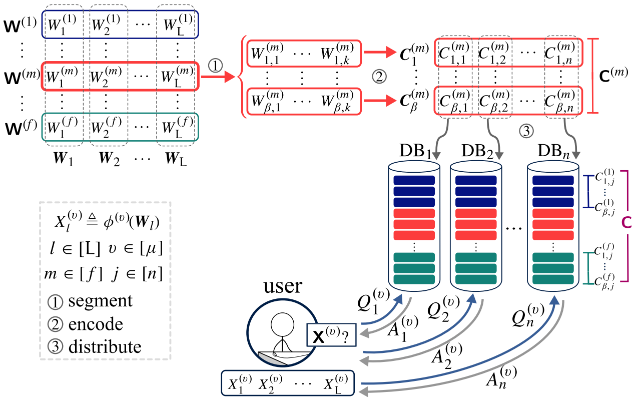

$\mathbf{W}^{(1)}$
$W_1^{(1)}$ $W_2^{(1)}$ $\cdots$ $W_{\mathrm{L}}^{(1)}$
$\mathbf{W}^{(m)}$
$W_1^{(m)}$ $W_2^{(m)}$ $\cdots$ $W_{\mathrm{L}}^{(m)}$
$\mathbf{W}^{(f)}$
$W_1^{(f)}$ $W_2^{(f)}$ $\cdots$ $W_{\mathrm{L}}^{(f)}$
$\boldsymbol{W}_1$ $\boldsymbol{W}_2$ $\cdots$ $\boldsymbol{W}_{\mathrm{L}}$
①
$W_{1,1}^{(m)}$ $\cdots$ $W_{1,k}^{(m)}$
$W_{\beta,1}^{(m)}$ $\cdots$ $W_{\beta,k}^{(m)}$
②
$\boldsymbol{C}_1^{(m)}$
$\boldsymbol{C}_\beta^{(m)}$
$C_{1,1}^{(m)}$ $C_{1,2}^{(m)}$ $\cdots$ $C_{1,n}^{(m)}$
$C_{\beta,1}^{(m)}$ $C_{\beta,2}^{(m)}$ $\cdots$ $C_{\beta,n}^{(m)}$
$\mathbf{C}^{(m)}$
③
DB_1 DB_2 DB_n
$C_{1,j}^{(1)}$
$C_{\beta,j}^{(1)}$
$\mathbf{C}$
$C_{1,j}^{(f)}$
$C_{\beta,j}^{(f)}$
$X_l^{(v)} \triangleq \phi^{(v)}(\boldsymbol{W}_l)$
$l \in [\mathrm{L}]$ $v \in [\mu]$
$m \in [f]$ $j \in [n]$
① segment
② encode
③ distribute
user
$\mathbf{X}^{(v)}$?
$Q_1^{(v)}$ $A_1^{(v)}$
$Q_2^{(v)}$ $A_2^{(v)}$
$Q_n^{(v)}$ $A_n^{(v)}$
$X_1^{(v)}$ $X_2^{(v)}$ $\cdots$ $X_{\mathrm{L}}^{(v)}$

Fig. 1. System model for PPC from an $[n, k]$ coded DSS storing f messages.

We use the customary code parameters $[n, k]$ to denote a code $\mathscr{C}$ over the finite field $\mathbb{F}_q$ of blocklength n and dimension k. A generator matrix of $\mathscr{C}$ is denoted by $\mathsf{G}^{\mathscr{C}}$. A set of coordinates of $\mathscr{C}$, $\mathcal{I} \subseteq [n]$, of size k is said to be an *information set* if and only if $\mathsf{G}^{\mathscr{C}}|_{\mathcal{I}}$ is invertible. $(\cdot)^{\mathsf{T}}$ denotes the transpose operator, while $\operatorname{rank}(\mathsf{V})$ denotes the rank of a matrix V. The function $\chi(\boldsymbol{x})$ denotes the support of a vector $\boldsymbol{x}$, and the linear span of a set of vectors $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_a\}$, $a \in \mathbb{N}$, is denoted by $\operatorname{span}\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_a\}$. Finally, $\mathbb{F}_q[z]$ denotes the set of all univariate polynomials over $\mathbb{F}_q$ in the variable z, and we denote by $\deg(\phi(z))$ the degree of a polynomial $\phi(z) \in \mathbb{F}_q[z]$.

B. Problem Statement and System Model

The PPC problem for coded DSSs is described as follows. We consider a DSS that stores in total f independent messages $\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(f)}$, where each message symbol $W_1^{(m)}, \ldots, W_{\mathrm{L}}^{(m)}$ is chosen independently and uniformly at random from $\mathbb{F}_q$. Thus, $\mathrm{H}(\mathbf{W}^{(m)}) = \mathrm{L}$, $\forall m \in [f]$ (in q-ary units). Let $\mathrm{L} \triangleq \beta k$, for some $\beta, k \in \mathbb{N}$. The DSS stores the f messages encoded using an $[n, k]$ code as follows. Shown in Fig. 1, first, the symbols of each message $\mathbf{W}^{(m)}$, $m \in [f]$, are presented as a $\beta \times k$ matrix, i.e., $\mathbf{W}^{(m)} = \left(W_{i,j}^{(m)}\right)$, $i \in [\beta]$, $j \in [k]$.

Let $\boldsymbol{W}_i^{(m)} = \bigl(W_{i,1}^{(m)}, \dots, W_{i,k}^{(m)}\bigr)$, $i \in [\beta]$, denote a message vector corresponding to the i-th row of $\mathbf{W}^{(m)}$. Second, each $\boldsymbol{W}_i^{(m)}$ is encoded by an $[n,k]$ code $\mathscr{C}$ over $\mathbb{F}_q$ into a length-n codeword $\boldsymbol{C}_i^{(m)} = \bigl(C_{i,1}^{(m)}, \dots, C_{i,n}^{(m)}\bigr)$. The βf generated codewords $\boldsymbol{C}_i^{(m)}$ are then arranged in the array $\mathbf{C} = \bigl((\mathbf{C}^{(1)})^{\mathsf{T}}|\dots|(\mathbf{C}^{(f)})^{\mathsf{T}}\bigr)^{\mathsf{T}}$ of dimensions $\beta f \times n$, where $\mathbf{C}^{(m)} = \bigl((\boldsymbol{C}_1^{(m)})^{\mathsf{T}}|\dots|(\boldsymbol{C}_\beta^{(m)})^{\mathsf{T}}\bigr)^{\mathsf{T}}$. Finally, the code symbols $C_{1,j}^{(m)}, \dots, C_{\beta,j}^{(m)}$, $m \in [f]$, for all f messages are stored on the j-th database, $j \in [n]$.

In PPC from n noncolluding databases, a user wishes to privately compute exactly one function image $X_l^{(\upsilon)} \triangleq \phi^{(\upsilon)}(\boldsymbol{W}_l)$, where $\boldsymbol{W}_l = (W_l^{(1)}, \dots, W_l^{(f)})$, $\forall\, l \in [\mathsf{L}]$, out of μ arbitrary *candidate* polynomials $\phi^{(1)}, \dots, \phi^{(\mu)} \colon \mathbb{F}_q^f \to \mathbb{F}_q$ from the coded DSS. Let $\mathbf{X}^{(\upsilon)} = \bigl(X_1^{(\upsilon)}, \dots, X_{\mathsf{L}}^{(\upsilon)}\bigr)$, where $X_1^{(\upsilon)}, \dots, X_{\mathsf{L}}^{(\upsilon)}$ are independent and identically distributed according to a prototype random variable $X^{(\upsilon)}$ with probability mass function $P_{X^{(\upsilon)}}$. Thus, $\mathsf{H}(\mathbf{X}^{(\upsilon)}) = \mathsf{L}\,\mathsf{H}(X^{(\upsilon)})$, $\forall\, \upsilon \in [\mu]$, $\mathsf{H}(\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(\mu)}) = \mathsf{L}\,\mathsf{H}(X^{(1)}, \dots, X^{(\mu)})$, and we let $\mathsf{H}_{\min} \triangleq \min_{\upsilon \in [\mu]} \mathsf{H}(X^{(\upsilon)})$ and $\mathsf{H}_{\max} \triangleq \max_{\upsilon \in [\mu]} \mathsf{H}(X^{(\upsilon)})$. The user selects an index $\upsilon \in [\mu]$ and wishes to compute the υ-th polynomial while keeping the requested polynomial index υ private from each database. Here, without loss of generality, we also assume that the polynomial candidate set contains its monomial basis, i.e., all monomials required to represent the polynomials in the candidate set as linear combinations of monomials, are included in the candidate set. In order to retrieve the desired polynomial evaluation $\mathbf{X}^{(\upsilon)}$, $\upsilon \in [\mu]$, from the coded DSS, the user sends a query $Q_j^{(\upsilon)}$ to the j-th database for all $j \in [n]$ as illustrated in Fig. 1. The queries are generated by the user without any prior knowledge of the realizations of the candidate polynomials, consequently, they are independent of the candidate polynomials evaluations. In other words, we have

$$\mathsf{I}\Bigl(\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(\mu)}; Q_1^{(\upsilon)}, \dots, Q_n^{(\upsilon)}\Bigr) = 0, \quad \forall\, \upsilon \in [\mu].$$

In response to the received query, database j generates the answer $A_j^{(\upsilon)}$ as a deterministic function of $Q_j^{(\upsilon)}$ and the data stored in the database, and then sends it back to the user. Let $\boldsymbol{C}_j \triangleq \bigl(C_{1,j}^{(1)}, \dots, C_{\beta,j}^{(1)}, C_{1,j}^{(2)}, \dots, C_{\beta,j}^{(f)}\bigr)^{\mathsf{T}}$ denote the f coded chunks that are stored in the j-th database. Thus, $\forall\, \upsilon \in [\mu]$,

$$\mathsf{H}\Bigl(A_j^{(\upsilon)} \Bigm| Q_j^{(\upsilon)}, \boldsymbol{C}_j\Bigr) = 0, \quad \forall\, j \in [n].$$

To guarantee user privacy, in an information-theoretic sense, the query-answer function must be identically distributed for each possible desired polynomial index $\upsilon \in [\mu]$ from the perspective of each database $j \in [n]$. In other words, the scheme's queries and answer strings must be independent from the desired polynomial index, therefore, revealing no information about the identity of the desired polynomial evaluation. Moreover, the user must be able to reliably decode the desired polynomial evaluation $\mathbf{X}^{(\upsilon)}$. Accordingly, we define a PPC protocol for an $[n,k]$ coded DSSs as follows.

Consider a DSS with n noncolluding databases storing f messages using an $[n,k]$ code. The user wishes to retrieve the υ-th polynomial evaluation $\mathbf{X}^{(\upsilon)}$, $\upsilon \in [\mu]$, from the available information $Q_j^{(\upsilon)}$ and $A_j^{(\upsilon)}$, $j \in [n]$. For a PPC protocol, the following conditions must be satisfied $\forall\, \upsilon, \upsilon' \in [\mu]$, $\upsilon \neq \upsilon'$, and $\forall\, j \in [n]$,

$$\text{[Privacy]} \quad (Q_j^{(\upsilon)}, A_j^{(\upsilon)}, \mathbf{X}^{[\mu]}) \sim (Q_j^{(\upsilon')}, A_j^{(\upsilon')}, \mathbf{X}^{[\mu]}), \quad \text{(1a)}$$

$$\text{[Recovery]} \quad \mathsf{H}\bigl(\mathbf{X}^{(\upsilon)} \bigm| A_1^{(\upsilon)}, \dots, A_n^{(\upsilon)}, Q_1^{(\upsilon)}, \dots, Q_n^{(\upsilon)}\bigr) = 0. \quad \text{(1b)}$$

From an information-theoretic perspective, the efficiency of a PPC protocol is measured by the *PPC rate*, which is defined as follows.

Definition 1 (PPC Rate and Capacity for Linearly-Coded DSSs): The exact information-theoretic rate of a PPC scheme, denoted by R*, is defined as the ratio of the minimum desired function size* $\mathsf{L}\,\mathsf{H}_{\min}$ *over the total required download cost, i.e.,* $\mathsf{R} \triangleq \mathsf{L}\,\mathsf{H}_{\min}/\mathsf{D}$*, where* D *is the total required download cost. The PPC* capacity $\mathsf{C}_{\mathsf{PPC}}$ *is the maximum of all achievable PPC rates over all possible PPC protocols for a given* $[n,k]$ *storage code.*

C. Background

A monomial $\boldsymbol{z}^{\boldsymbol{i}}$ in m variables $z_1, \dots, z_m$ with degree g is written as $\boldsymbol{z}^{\boldsymbol{i}} = z_1^{i_1} z_2^{i_2} \cdots z_m^{i_m}$, where $\boldsymbol{i} \triangleq (i_1, \dots, i_m) \in \mathbb{N}_0^m$ is the exponent vector with $\mathrm{wt}(\boldsymbol{i}) \triangleq \sum_{j=1}^m i_j = g$. The set $\{\boldsymbol{z}^{\boldsymbol{i}} : \boldsymbol{i} \in \mathbb{N}_0^m,\ 1 \leq \mathrm{wt}(\boldsymbol{i}) \leq g\}$ of all monomials in m variables of degree at most g has size $\mathsf{M}_g(m) \triangleq \sum_{h=1}^g \binom{h+m-1}{h} = \binom{g+m}{g} - 1$. Moreover, a polynomial $\phi(\boldsymbol{z})$ of degree at most g is represented as $\phi(\boldsymbol{z}) = \sum_{\boldsymbol{i}:\mathrm{wt}(\boldsymbol{i}) \leq g} a_{\boldsymbol{i}} \boldsymbol{z}^{\boldsymbol{i}}$, $a_{\boldsymbol{i}} \in \mathbb{F}_q$. The total number of polynomials in m variables of degree at most g generated with all possible distinct (up to scalar multiplication) $\mathsf{M}_g(m)$-dimensional coefficients vectors defined over $\mathbb{F}_q$ is equal to $\mu_g(m) \triangleq \bigl(q^{\mathsf{M}_g(m)} - 1\bigr)/(q-1)$.

Definition 2 (Star-Product): Let $\mathscr{C}$ *and* $\mathscr{D}$ *be two linear codes of length* n *over* $\mathbb{F}_q$*. The star-product (Hadamard product) of* $\boldsymbol{v} = (v_1, \dots, v_n) \in \mathscr{C}$ *and* $\boldsymbol{u} = (u_1, \dots, u_n) \in \mathscr{D}$ *is defined as* $\boldsymbol{v} \star \boldsymbol{u} = (v_1 u_1, \dots, v_n u_n) \in \mathbb{F}_q^n$*. Further, the star-product of* $\mathscr{C}$ *and* $\mathscr{D}$*, denoted by* $\mathscr{C} \star \mathscr{D}$*, is defined by* $\mathrm{span}\{\boldsymbol{v} \star \boldsymbol{u} : \boldsymbol{v} \in \mathscr{C}, \boldsymbol{u} \in \mathscr{D}\}$ *and the* g*-fold star-product of* $\mathscr{C}$ *with itself is given by* $\mathscr{C}^{\star g} = \mathrm{span}\{\boldsymbol{v}_1 \star \cdots \star \boldsymbol{v}_g : \boldsymbol{v}_i \in \mathscr{C}, i \in [g]\}$.

Definition 3 (RS Code): Let $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_n)$ *be a vector of* n *distinct elements of* $\mathbb{F}_q$*. For* $n \in \mathbb{N}$*,* $k \in [n]$*, and* $q \geq n$*, the* $[n,k]$ *RS code (over* $\mathbb{F}_q$*) is defined as*

$$\mathcal{RS}_k(\boldsymbol{\alpha}) \triangleq \{(\phi(\alpha_1), \dots, \phi(\alpha_n)) \colon \phi \in \mathbb{F}_q[z],\ \deg(\phi) < k\}. \quad (2)$$

It is well-known that RS codes are MDS codes that behave well under the star-product. We state the following proposition that was introduced in [10].

Proposition 1: Let $\mathcal{RS}_k(\boldsymbol{\alpha})$ *be a length-*n *RS code. Then, for* $g \in \mathbb{N}$*, the* g*-fold star-product of* $\mathcal{RS}_k(\boldsymbol{\alpha})$ *with itself is the RS code given by* $\mathcal{RS}_k^{\star g}(\boldsymbol{\alpha}) = \mathcal{RS}_{\min\{g(k-1)+1, n\}}(\boldsymbol{\alpha})$.

Let $\boldsymbol{\gamma} = (\gamma_1, \dots, \gamma_k)$ be a vector of k distinct elements of $\mathbb{F}_q$. For a message vector $\boldsymbol{W} = (W_1, \dots, W_k)$, let $\ell(z) \in \mathbb{F}_q[z]$ be a polynomial of degree at most $k-1$ such that $\ell(\gamma_i) = W_i$

for all $i \in [k]$. Using the Lagrange interpolation formula we present this polynomial as $\ell(z) = \sum_{i\in[k]} W_i \iota_i(z)$, where $\iota_i(z)$ is the Lagrange basis polynomial

$$\iota_i(z) = \prod_{t\in[k]\setminus\{i\}} \frac{z-\gamma_t}{\gamma_i-\gamma_t}.$$

It was shown in [20] that Lagrange encoding is equivalent to the choice of a specific basis for an RS code. Therefore, for encoding we choose the set of Lagrange basis polynomials as the code generating polynomials of (2) [27]. Thus, a generator matrix of $\mathcal{RS}_k(\boldsymbol{\alpha})$ is $\mathsf{G}_{\mathcal{RS}_k}(\boldsymbol{\alpha},\boldsymbol{\gamma}) = (\iota_i(\alpha_j))$, $i \in [k]$, $j \in [n]$. Note that if we choose $\gamma_i = \alpha_i$ for $i \in [k]$, then the generator matrix $\mathsf{G}_{\mathcal{RS}_k}(\boldsymbol{\alpha},\boldsymbol{\gamma})$ becomes systematic.

In [12], a PIR protocol for any linearly-coded DSS that uses an $[n,k]$ code to store f messages, named Protocol 1, is proposed. The PIR rate of Protocol 1 can be derived by finding a *PIR achievable rate matrix* of the underlying storage code $\mathscr{C}$, which is defined as follows.

Definition 4 ([12, Def. 10]): Let $\mathscr{C}$ be an arbitrary $[n,k]$ code. A $\nu \times n$ binary matrix $\Lambda^{\mathrm{PIR}}_{\kappa,\nu}(\mathscr{C})$ is said to be a PIR achievable rate matrix for $\mathscr{C}$ if the following conditions are satisfied.

1) *The Hamming weight of each column of $\Lambda^{\mathrm{PIR}}_{\kappa,\nu}$ is κ, and*
2) *for each matrix row $\boldsymbol{\lambda}_i$, $i \in [\nu]$, $\chi(\boldsymbol{\lambda}_i)$ always contains an information set.*

In other words, each coordinate j of $\mathscr{C}$, $j \in [n]$, appears exactly κ times in $\{\chi(\boldsymbol{\lambda}_i)\}_{i\in[\nu]}$, and every set $\chi(\boldsymbol{\lambda}_i)$ contains an information set.

This gives rise to the following definition.

Definition 5 ([12, Def. 13]): Given an $[n,k]$ code $\mathscr{C}$, if a PIR achievable rate matrix $\Lambda^{\mathrm{PIR}}_{\kappa,\nu}(\mathscr{C})$ with $\kappa/\nu = k/n$ exists, then the code $\mathscr{C}$ is referred to as an MDS-PIR capacity-achieving *code, and the matrix $\Lambda^{\mathrm{PIR}}_{\kappa,\nu}(\mathscr{C})$ is called an* MDS-PIR capacity-achieving *matrix.*

Note that the class of MDS-PIR capacity-achieving codes includes MDS codes, cyclic codes, Reed-Muller codes, and certain classes of distance-optimal local reconstruction codes [12].

III. Converse Bound

In [18], the PLC capacity for a coded DSS using an MDS-PIR capacity-achieving code is shown to be equal to the MDS-PIR capacity. In this section, we derive an outer bound on the PPC rate (Theorem 1 below) by adapting the converse proof of [18, Thm. 2] to the scenario of the linearly-coded PPC problem, where the storage code is MDS-PIR capacity-achieving. We first define an effective rank for the PPC problem as follows.

Definition 6: Let $\mathbf{X}^{[\mu]} = \{\mathbf{X}^{(1)},\dots,\mathbf{X}^{(\mu)}\}$ denote the set of candidate polynomials evaluations where $\mathbf{X}^{(\ell)} = \bigl(X^{(\ell)}_1,\dots,X^{(\ell)}_{\mathrm{L}}\bigr)$, $\ell \in [\mu]$. The effective rank *$r\bigl(\mathbf{X}^{[\mu]}\bigr)$ is defined as*

$$r\bigl(\mathbf{X}^{[\mu]}\bigr) \triangleq \min\Bigl\{s\colon \mathrm{H}\bigl(X^{(\ell_1)}_l,\dots,X^{(\ell_s)}_l\bigr) = \mathrm{H}\bigl(X^{[\mu]}_l\bigr),\\ \{\ell_1,\dots,\ell_s\} \subseteq [\mu],\ s \in [\mu],\ \forall\, l \in [\mathrm{L}]\Bigr\}, \quad (3)$$

and we define the set $\mathcal{L} \triangleq \{\ell_1,\dots,\ell_r\} \subseteq [\mu]$ to be a minimum set that satisfies (3).[1]

Accordingly, an upper bound on the capacity of PPC for a coded DSS where data is encoded and stored using an MDS-PIR capacity-achieving code introduced in Definition 5, is stated as follows.

Theorem 1: Consider a DSS with n noncolluding databases that uses an $[n,k]$ MDS-PIR capacity-achieving code $\mathscr{C}$ to store f messages. Then, the maximum achievable PPC rate over all possible PPC protocols, i.e., the PPC capacity $\mathsf{C}_{\mathrm{PPC}}$, is upper bounded by

$$\mathsf{C}_{\mathrm{PPC}} \leq \frac{\mathsf{H}_{\min}}{\mathsf{H}^{(\mathrm{B})}_{\min} + \sum_{\upsilon=1}^{r-1}\bigl(\frac{k}{n}\bigr)^{\upsilon}\,\mathrm{H}\bigl(X^{(\ell_{\upsilon+1})}\bigm| X^{(\ell_1)},\dots,X^{(\ell_\upsilon)}\bigr)},$$

for any effective rank $r\bigl(\mathbf{X}^{[\mu]}\bigr) = r$, where $\mathsf{H}^{(\mathrm{B})}_{\min} \triangleq \min_{\ell\in\mathcal{L}} \mathrm{H}\bigl(X^{(\ell)}\bigr)$.

Here, we remark that Theorem 1 generalizes [15, Thm. 1], which is a converse bound on the capacity of dependent PIR (DPIR) for noncolluding replicated databases.

Remark 1: Restricting the candidate set to degree $g = 1$ polynomials reduces the PPC problem to a PLC problem where there is a deterministic linear mapping $\mathsf{V}_{\mu\times f}$ between the μ functions evaluations and the f information messages. Thus, the effective rank given in Definition 6 becomes the rank of said mapping, i.e., $r = \mathrm{rank}\bigl(\mathsf{V}_{\mu\times f}\bigr)$. Moreover, the candidate functions evaluations with indices from the set $\mathcal{L} = \{\ell_1,\dots,\ell_r\}$ that satisfies (3) *are independent and identically distributed according to a uniform distribution [18, Lem. 3]. As a result, for $\upsilon \in [r-1]$ we have $\mathrm{H}\bigl(X^{(\ell_{\upsilon+1})}\bigm| X^{\{\ell_1,\dots,\ell_\upsilon\}}\bigr) = \mathrm{H}\bigl(X^{(\ell_{\upsilon+1})}\bigr) = 1$, $\mathsf{H}^{(\mathrm{B})}_{\min} = \mathsf{H}_{\min} = 1$, and the capacity of PLC [18, Thm. 2] follows.*

Accordingly, the proof of Theorem 1 is an extension to our converse proof of PLC in [18] and is presented in Appendix A.

Remark 2: Note that the converse bound of Theorem 1 is generally difficult to compute for a large number of candidate polynomials. However, it is worth mentioning that there are two cases where the computation of the converse bound is straightforward. Namely, the case of the candidate functions being from the linear polynomials *class, following Remark 1, and the case where the set of μ candidate polynomials evaluations includes the f* independent *files, i.e., $\{\mathbf{W}^{(1)},\dots,\mathbf{W}^{(f)}\} \subset \{\mathbf{X}^{(1)},\dots,\mathbf{X}^{(\mu)}\}$. For this case, the rank of the candidate functions set is simply $r = f$ as all the remaining candidate polynomials evaluations are a function of these f files and no other smaller subset captures the value of the joint entropy $\mathrm{H}\bigl(X^{[\mu]}_l\bigr)$ of* (3)*. Since these f files are independent and uniformly distributed, computing the capacity bound reduces to computing only the minimum entropy.*

IV. General PPC Scheme for RS-Coded DSSs

In the following, we build PPC schemes based on Lagrange encoding and our PLC scheme in [18]. Note that a polynomial

[1] There always exists a subset $\{\ell_1,\dots,\ell_s\} \subseteq [\mu]$ that satisfies the joint entropy condition of (3). For the case where the candidate functions result in independent functions evaluations, this set is the set of all function evaluations, i.e., $r = \mu$. Moreover, we naturally assume that $r > 1$, as $\mu > 1$ and $f > 1$. Otherwise, the problem becomes trivial in the sense that there is only one candidate message/computation to retrieve.

can be written as a linear combination of monomials, and therefore PMC is a special case of PPC. Thus, a PPC scheme can be obtained from a PLC scheme by replacing independent messages with a monomial basis. We first discuss the PPC case in general in the following scheme. In RS-coded DSSs, each message vector $\boldsymbol{W}_i^{(m)}$ is encoded by an RS code $\mathcal{RS}_k(\boldsymbol{\alpha})$ with evaluation vector $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)$ over $\mathbb{F}_q$ into a length-n codeword $\boldsymbol{C}_i^{(m)}$ where $\boldsymbol{C}_i^{(m)} = \boldsymbol{W}_i^{(m)} \mathsf{G}_{\mathcal{RS}_k}(\boldsymbol{\alpha}, \boldsymbol{\gamma}) = \bigl(C_{i,1}^{(m)}, \ldots, C_{i,n}^{(m)}\bigr)$ and $C_{i,j}^{(m)} = \ell(\alpha_j)$, $j \in [n]$. Consider an RS-coded DSS with n noncolluding databases storing f messages. The user wishes to retrieve the υ-th polynomial evaluation $\mathbf{X}^{(\upsilon)}$, $\upsilon \in [\mu]$, from the available information from queries $Q_j^{(\upsilon)}$ and answer strings $A_j^{(\upsilon)}$, $j \in [n]$, satisfying conditions (1a) and (1b).

A. Lagrange Coded Computation

Lagrange coded computation [27] is a framework that can be applied to any function computation when the function of interest is a multivariate polynomial of the messages. We extend the application of this framework to PMC and PPC by utilizing the following argument.

Let $\ell_t^{(m)}(z)$ be the Lagrange interpolation polynomial associated with the length-k message segment $\boldsymbol{W}_t^{(m)}$ for some $t \in [\beta]$ and $m \in [f]$. Recall that $\ell_t^{(m)}(z)$ evaluated at γ_j results in an information symbol $W_{t,j}^{(m)}$ and when evaluated at α_j we obtain a code symbol $C_{t,j}^{(m)}$. Let $\boldsymbol{\ell}_t(z) = (\ell_t^{(1)}(z), \ldots, \ell_t^{(f)}(z))$ be a vector of f Lagrange interpolation polynomials associated with the messages $\boldsymbol{W}_t^{(1)}, \ldots, \boldsymbol{W}_t^{(f)}$. Now, given a multivariate polynomial $\phi(\boldsymbol{W}_{t,j})$ of degree at most g, where $\boldsymbol{W}_{t,j} \triangleq \bigl(W_{t,j}^{(1)}, \ldots, W_{t,j}^{(f)}\bigr)^{\mathsf{T}}$, we introduce the composition function $\psi_t(z) = \phi(\boldsymbol{\ell}_t(z))$. Accordingly, evaluating $\psi_t(z)$ at any γ_j, $j \in [k]$, is equal to evaluating the polynomial over the uncoded information symbols, i.e., $\phi(\boldsymbol{W}_{t,j})$ and similarly, evaluating $\psi_t(z)$ at α_j, $j \in [n]$, will result in the evaluation of the polynomial over the coded symbols, i.e., $\phi(\boldsymbol{C}_{t,j})$, where $\boldsymbol{C}_{t,j} \triangleq \bigl(C_{t,j}^{(1)}, \ldots, C_{t,j}^{(f)}\bigr)^{\mathsf{T}}$. Since each Lagrange interpolation polynomial of $\boldsymbol{\ell}_t(z)$ is a polynomial of degree at most $k-1$, it follows that $\deg(\psi_t(z)) \leq g(k-1)$ and we require up to $g(k-1)+1$ coefficients to interpolate and determine the polynomial $\psi_t(z)$.

Note that $\psi_t(z)$ is a linear combination of monomials $z^i \in \mathbb{F}_q[z]$, $i \leq g(k-1)$, and the underlying code $\tilde{\mathscr{C}}$ for $(\psi_t(\alpha_1), \ldots, \psi_t(\alpha_n))$, referred to as the *polynomial decoding code*, is given by the g-fold star-product $\mathcal{RS}_k^{\star g}(\boldsymbol{\alpha})$ of the storage code $\mathcal{RS}_k(\boldsymbol{\alpha})$ according to [20, Lem. 6]. This is due to the fact that the span of $\mathcal{RS}_k^{\star g}(\boldsymbol{\alpha})$ is given by linear combinations of codewords in $\mathcal{RS}_k^{\star g}(\boldsymbol{\alpha})$ where each code symbol represents a monomial. In other words, to construct coded PPC schemes that retrieve polynomials of degree at most g, we require $g(k-1)+1 \leq n$ and $d_{\min}^{\tilde{\mathscr{C}}} \geq n - (g(k-1)+1) + 1$, where $d_{\min}^{\tilde{\mathscr{C}}}$ denotes the minimum distance of $\tilde{\mathscr{C}}$, to be able to decode the computation correctly. It follows from Proposition 1 that $\tilde{\mathscr{C}} = \mathcal{RS}_{\tilde{k}}(\boldsymbol{\alpha})$ with dimension $\tilde{k} = \min\{g(k-1)+1, n\} = g(k-1)+1$ and $d_{\min}^{\tilde{\mathscr{C}}} = n - \tilde{k} + 1 = n - (g(k-1)+1) + 1$.

B. PPC Achievable Rate Matrix

We now extend the notion of a PIR achievable rate matrix for the coded PIR problem in Definition 4 to the coded PPC problem.

Definition 7: Let $\mathscr{C}$ be an arbitrary $[n,k]$ code and denote by $\tilde{\mathscr{C}} = \mathscr{C}^{\star g}$ the $\tilde{k}$-dimensional code generated by the g-fold star-product of $\mathscr{C}$ with itself. A $\nu \times n$ binary matrix $\Lambda_{\kappa,\nu}^{\mathrm{PPC}}$ is called a PPC achievable rate matrix *for $(\mathscr{C}, \tilde{\mathscr{C}})$, if*

1) *$\Lambda_{\kappa,\nu}^{\mathrm{PPC}}$ is a κ-column regular matrix, i.e., its column sums are equal to κ, with $\kappa/\nu = \tilde{k}/n$, and*
2) *for each matrix row $\boldsymbol{\lambda}_i$, $\chi(\boldsymbol{\lambda}_i)$ is always an information set for $\tilde{\mathscr{C}}$, $i \in [\nu]$.*

In [12, Def. 11], two PIR interference matrices are defined from a PIR achievable rate matrix. Similar to the notion of PIR interference matrices, given a PPC achievable rate matrix $\Lambda_{\kappa,\nu}^{\mathrm{PPC}}$, the PPC *interference matrices* $\mathsf{A}_{\kappa \times n}$ and $\mathsf{B}_{(\nu-\kappa)\times n}$, are defined as follows.

Definition 8: For a given $\nu \times n$ PPC achievable rate matrix $\Lambda_{\kappa,\nu}^{\mathrm{PPC}}(\mathscr{C}, \tilde{\mathscr{C}}) = (\lambda_{u,j})$, we define the interference matrices $\mathsf{A}_{\kappa \times n} = (a_{i,j})$ and $\mathsf{B}_{(\nu-\kappa)\times n} = (b_{i,j})$ for the code pair $(\mathscr{C}, \tilde{\mathscr{C}})$ as

$$\begin{aligned} a_{i,j} &\triangleq u \quad \text{if } \lambda_{u,j} = 1, \ \forall\, j \in [n],\ i \in [\kappa],\ u \in [\nu], \\ b_{i,j} &\triangleq u \quad \text{if } \lambda_{u,j} = 0, \ \forall\, j \in [n],\ i \in [\nu-\kappa],\ u \in [\nu]. \end{aligned}$$

For $j \in [n]$, let $\mathcal{A}_j \triangleq \{a_{i,j} : i \in [\kappa]\}$ and $\mathcal{B}_j \triangleq \{b_{i,j} : i \in [\nu-\kappa]\}$. Then, the j-th column of $\mathsf{A}_{\kappa \times n}$ contains the row indices of $\Lambda_{\kappa,\nu}^{\mathrm{PPC}}$ whose entries in the j-th column are equal to 1, while $\mathsf{B}_{(\nu-\kappa)\times n}$ contains the remaining row indices of $\Lambda_{\kappa,\nu}^{\mathrm{PPC}}$. Hence, $\mathcal{B}_j = [\nu] \setminus \mathcal{A}_j$, $\forall\, j \in [n]$.

Note that in Definition 8, for each $j \in [n]$, distinct values of $u \in [\nu]$ should be assigned for all i. Thus, the assignment is not unique in the sense that the order of the entries of each column of A and B can be permuted.

Example 1: Consider a DSS storing messages using a $[4,2]$ RS code $\mathscr{C}$ over $\mathbb{F}_5$ with $\mathsf{G}^{\mathscr{C}} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \end{pmatrix}$ and candidate polynomials of degree at most $g = 2$. We have $\tilde{\mathscr{C}} = \mathscr{C}^{\star 2}$, $\tilde{k} = g(k-1)+1 = 3$, and $\mathsf{G}^{\tilde{\mathscr{C}}} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \\ 0 & 1 & 4 & 4 \end{pmatrix}$. One can verify that

$$\Lambda_{3,4}^{\mathrm{PPC}} = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix}$$

is a valid PPC achievable rate matrix for $(\mathscr{C}, \tilde{\mathscr{C}})$, with $(\kappa, \nu) = (3,4)$, generated using the four information sets of $\tilde{\mathscr{C}}$ and the corresponding interference matrices are given by

$$\mathsf{A}_{3\times 4} = \begin{pmatrix} 1 & 1 & 1 & 2 \\ 2 & 2 & 3 & 3 \\ 3 & 4 & 4 & 4 \end{pmatrix} \quad \text{and} \quad \mathsf{B}_{1\times 4} = \begin{pmatrix} 4 & 3 & 2 & 1 \end{pmatrix}.$$

▽

C. Generic Query Generation

In this subsection, we utilize the query generation algorithm `Q-Gen` that is introduced for PLC from MDS-PIR capacity-achieving coded DSSs in [18] as the basis for our PPC scheme. More specifically, the query generation algorithm `Q-Gen` generates the query of a PIR-like scheme from a linearly-coded DSS with dependent virtual messages representing the evaluations of the μ candidate functions. Accordingly, the PPC scheme requires the length of each message to be $\mathsf{L} = \nu^{\mu} \cdot k$. Before running the main algorithm to generate the query sets, the following index preparation for the coded symbols stored in each database is performed.

1) Index Preparation: Given that the query generation algorithm `Q-Gen` generates a fixed query set structure as a deterministic function of the desired polynomial index, we introduce an index permutation. The goal is to make the symbols queried from each database appear to be chosen randomly and independently from the desired polynomial index. Note that the polynomial is computed separately for the t-th row of all messages, $t \in [\beta]$. Therefore, similar to the coded PLC schemes in [17], [18], we apply a permutation that is fixed across all coded symbols for the t-th row to maintain the dependency across the associated message elements. Let $\pi(\cdot)$ be a random permutation function over $[\beta]$, and let

$$U^{(\upsilon')}_{t,j} \triangleq \phi^{(\upsilon')}(\boldsymbol{C}_{\pi(t),j}), \quad t \in [\beta], \;\; j \in [n], \;\; \upsilon' \in [\mu],$$

denote the t-th permuted symbol associated with the υ'-th virtual message $\mathbf{X}^{(\upsilon')}$ stored in the j-th database, where $\boldsymbol{C}_{t,j} = \big(C^{(1)}_{t,j}, \dots, C^{(f)}_{t,j}\big)^{\mathsf{T}}$. The permutation $\pi(\cdot)$ is randomly selected privately and uniformly by the user.

2) Preliminaries: The query generation procedure is subdivided into μ rounds, where each round τ generates the queries based on the concept of τ*-sums* as defined in the following.

Definition 9 (τ-Sum): For $\tau \in [\mu]$, a sum $U^{(\upsilon_1)}_{i_1,j} + U^{(\upsilon_2)}_{i_2,j} + \cdots + U^{(\upsilon_\tau)}_{i_\tau,j}$, $j \in [n]$, of τ distinct symbols is called a τ-sum for any $(i_1, \dots, i_\tau) \in [\beta]^\tau$, and $\{\upsilon_1, \dots, \upsilon_\tau\} \subseteq [\mu]$ determines the type *of the τ-sum.*

Since we have $\binom{\mu}{\tau}$ different selections of τ distinct elements out of μ elements, a τ-sum can have $\binom{\mu}{\tau}$ different *types*. For a requested polynomial evaluation indexed by $\upsilon \in [\mu]$, a query set $Q^{(\upsilon)}_j$, $j \in [n]$, is composed of μ disjoint subsets of queries, each subset of queries is generated by the operations of each round $\tau \in [\mu]$. In a round we generate the queries for all possible $\binom{\mu}{\tau}$ types of τ-sums. For each round $\tau \in [\mu]$ the corresponding query subset is further subdivided into two subsets $Q^{(\upsilon)}_j(\mathcal{D}; \tau)$ and $Q^{(\upsilon)}_j(\mathcal{U}; \tau)$. The first subset $Q^{(\upsilon)}_j(\mathcal{D}; \tau)$ corresponds to τ-sums with a single symbol from the *desired* polynomial evaluation and $\tau - 1$ symbols from the evaluations of *undesired* polynomials, while the second subset $Q^{(\upsilon)}_j(\mathcal{U}; \tau)$ corresponds to τ-sums with symbols only from the evaluations of undesired polynomials. Here, $\mathcal{D}$ is an indicator for "desired function evaluation," while $\mathcal{U}$ an indicator for "undesired functions evaluations." Note that we require $\kappa^{\mu-(\tau-1)}(\nu - \kappa)^{\tau-1}$ distinct instances of each τ-sum type for every query set $Q^{(\upsilon)}_j$. We utilize these sets to generate the query sets of each round according to the interference matrices $\mathsf{A}_{\kappa \times n}$ and $\mathsf{B}_{(\nu-\kappa)\times n}$.

The queries $Q^{(\upsilon)}_j$ are generated by setting $(\kappa, \nu) = (\tilde{k}, n)$ and invoking the query generation algorithm `Q-Gen` of [18, Sec. IV-A] with the PPC problem parameters as follows:

$$\{Q^{(\upsilon)}_1, \dots, Q^{(\upsilon)}_n\} \leftarrow \texttt{Q-Gen}(\upsilon, \mu, \kappa, \nu, n, \mathsf{A}_{\kappa \times n}, \mathsf{B}_{(\nu-\kappa)\times n}).$$

The total number of queries generated by the algorithm is given by

$$\sum_{j=1}^{n} \big| Q^{(\upsilon)}_j \big| = n \sum_{\tau=1}^{\mu} \binom{\mu}{\tau} \kappa^{\mu-\tau+1} (\nu - \kappa)^{\tau-1}. \tag{4}$$

D. Sign Assignment and Redundancy Elimination

Here, we generalize the coded PLC scheme of [18] in terms of exploiting the dependency between the virtual messages. Let $\mathsf{M}^{\mathrm{c}}_g(f)$ denote the size of the monomial basis of the polynomial candidate set. Then, since any polynomial in the candidate set is a linear function of its monomial basis of size $\mathsf{M}^{\mathrm{c}}_g(f)$, a PPC scheme can be seen as a PLC scheme performed over a set of $\mathsf{M}^{\mathrm{c}}_g(f)$ messages. Hence, the redundancy resulting from the linear dependencies between the virtual messages is also present for PPC and we can extend [18, Lem. 4] and [13, Lem. 1] to this scheme. To exploit the dependency between the virtual messages we adopt a similar sign assignment process to each queried symbol of the virtual monomial messages as detailed in [13, Sec. IV-B]. Using Lagrange interpolation, we will show that it results in a uniquely solvable equation system from the different τ-sum types given the side information available from all other databases. By obtaining such a system of equations in each round $\tau \in [\mu]$ of the protocol, the user can determine some of the answers offline.

Now, consider τ-sum types for $\tau = 1$, where we download individual segments of each virtual message including f independent messages. For this type, the user can determine any polynomial from the f obtained message segments. Based on this insight we can state the following lemma.

Lemma 1: Let $\mu \in [f : \mu_g(f)]$ be the number of candidate polynomials evaluations, including the f independent messages. For each query set, for all $\upsilon \in [\mu]$, each database $j \in [n]$, and based on the queried segments from the f independent messages, any $\binom{\mu-f}{1}$ 1-sum types out of all possible $\binom{\mu}{1}$ types are redundant. On the other hand, for $\tau \in [2 : \mu]$, any $\binom{\mu-\mathsf{M}^{\mathrm{c}}_g(f)}{\tau}$ τ-sum types out of $\binom{\mu}{\tau}$ types are redundant. Thus, the number of nonredundant τ-sum types with $\tau > 1$ is given by $\rho(\mu, \tau) \triangleq \binom{\mu}{\tau} - \binom{\mu-\mathsf{M}^{\mathrm{c}}_g(f)}{\tau}$.

The proof of Lemma 1 is presented in Appendix B. In the following subsection, we show that the recovery and privacy conditions of our proposed PPC scheme are satisfied.

E. Recovery and Privacy

The scheme works as the PLC scheme in [18] by using the code $\tilde{\mathscr{C}}$ instead of the storage code $\mathscr{C}$. This is the case since for *any* polynomial evaluation code $\mathscr{D}$, $\mathscr{D}^{*i} \subseteq \mathscr{D}^{*j}$ for all $i \in [j]$, $j \in \mathbb{N}$, since the all-ones codeword is in $\mathscr{D}$ (see also [20, Lem. 6]). Moreover, since the definition of the PPC achievable rate matrix in Definition 7 is analogous to the corresponding definition of a PIR achievable rate matrix in Definition 4 (by using $\tilde{\mathscr{C}}$ instead of $\mathscr{C}$), it can directly be seen that the arguments in the proof of [12, Thm. 1] (see [12, App. B]) can be applied. Hence, it follows that $\tilde{k}$ distinct evaluations of $\psi_t(z) = \phi(\boldsymbol{\ell}_t(z))$ for each segment t can be recovered. Since $\deg(\psi_t(z)) \leq \tilde{k} - 1$, it follows that the polynomial $\psi_t(z)$ can be reconstructed via polynomial interpolation and then the desired polynomial evaluations can be recovered by evaluating $\psi_t(z)$ at γ_j, $j \in [k]$. This is equal to evaluating the desired polynomial $\phi(\cdot)$ over the uncoded information symbols, i.e., $\phi(\boldsymbol{W}_{t,j})$ due to Lagrange encoding.

As for the privacy of the PPC scheme, using an argumentation similar to the PLC scheme [18, Sec. IV-D], it can be seen that for any desired index $\upsilon \in [\mu]$, the redundant τ-sum types according to Lemma 1 can be fixed, i.e., the same τ-sum types are redundant for all $\upsilon \in [\mu]$, and hence the queries satisfy the privacy condition.

F. Achievable PPC Rate

Since $\tilde{\mathscr{C}}$ is an $[n, \tilde{k}]$ MDS code ($\mathscr{C}$ is an RS code), there always exists a PPC achievable rate matrix $\Lambda^{\mathrm{PPC}}_{\kappa,\nu}$ with $\kappa/\nu = \tilde{k}/n$. Hence, using Lemma 1 we can prove the following theorem.

Theorem 2: Consider a DSS that uses an $[n, k]$ RS code $\mathscr{C}$ to store f messages over n noncolluding databases using Lagrange encoding. Let $\mu \in [f : \mu_g(f)]$ be the number of candidate polynomials evaluations of degree at most g, including the f independent messages. Then, the PPC rate

$$\mathsf{R}_{\mathrm{PPC}} = \begin{cases} \dfrac{1}{f}\,\mathsf{H}_{\min} \quad \textit{if } n \leq g(k-1)+1, \\[2ex] \dfrac{\frac{k}{\tilde{k}}\big(1-\frac{\tilde{k}}{n}\big)\,\mathsf{H}_{\min}}{1-\big(\frac{\tilde{k}}{n}\big)^{\mathsf{M}^{\mathrm{c}}_g(f)} - (\mathsf{M}^{\mathrm{c}}_g(f)-f)\big(1-\frac{\tilde{k}}{n}\big)\big(\frac{\tilde{k}}{n}\big)^{\mu-1}} \\ \qquad\qquad \textit{otherwise} \end{cases} \tag{5}$$

is achievable.

Proof: From (4) and Lemma 1, the achievable PPC rate after removing redundant τ-sums becomes

$$\begin{aligned} \mathsf{R} &\overset{(a)}{=} \frac{k\nu^{\mu}\,\mathsf{H}_{\min}}{n\Big(\binom{\mu}{1}-\binom{\mu-f}{1}\Big)\kappa^{\mu} + n\sum\limits_{\tau=2}^{\mu}\rho(\mu,\tau)\kappa^{\mu-\tau+1}(\nu-\kappa)^{\tau-1}} \\ &= \frac{k\nu^{\mu}\,\mathsf{H}_{\min}}{n\Big[f\kappa^{\mu}+\sum_{\tau=2}^{\mu}\rho(\mu,\tau)\kappa^{\mu-\tau+1}(\nu-\kappa)^{\tau-1}\Big]}, \end{aligned} \tag{6}$$

where (a) follows from the PPC rate in Definition 1, (4), and Lemma 1. Now, if $\nu = \kappa$, or equivalently (from Definition 7) $n = \tilde{k} \overset{(b)}{=} \min\{g(k-1)+1, n\}$, i.e., $n = g(k-1)+1$ (since n cannot be strictly smaller than $g(k-1)+1$ by assumption and (b) is from Proposition 1), then it follows directly from (6) that $\mathsf{R} = k\,\mathsf{H}_{\min}/nf$. Moreover, it can be seen in this case that the proposed scheme reduces to the trivial scheme where the f independent files are downloaded and then the desired polynomial evaluation is performed offline. However, the proposed scheme requires an unnecessarily high redundancy to decode the f files, i.e., $\tilde{k} = n$ instead of $\tilde{k} = k$. As a result, for the case of $n \leq g(k-1)+1$, we opt out of any other achievable scheme and achieve the PPC rate $\mathsf{H}_{\min}/f$ by simply downloading all f files and performing the desired polynomial evaluation offline. Otherwise, i.e., $\nu > \kappa$, or equivalently (from Definition 7), $n > \tilde{k} = \min\{g(k-1)+1, n\}$, i.e., $n > g(k-1)+1$, then from (6) we have

$$\begin{aligned} \mathsf{R} &\overset{(c)}{=} \frac{\frac{k(\nu-\kappa)}{n\kappa}\,\mathsf{H}_{\min}}{\Big[\frac{f(\nu-\kappa)}{\nu}\big(\frac{\kappa}{\nu}\big)^{\mu-1}+\frac{1}{\nu^{\mu}}\sum_{\tau=2}^{\mu}\rho(\mu,\tau)\kappa^{\mu-\tau}(\nu-\kappa)^{\tau}\Big]} \\ &\overset{(d)}{=} \frac{\frac{k(n-\tilde{k})}{n\tilde{k}}\,\mathsf{H}_{\min}}{\Big[f\big(1-\frac{\tilde{k}}{n}\big)\big(\frac{\tilde{k}}{n}\big)^{\mu-1}+\frac{1}{n^{\mu}}\sum_{\tau=2}^{\mu}\rho(\mu,\tau)\tilde{k}^{\mu-\tau}(n-\tilde{k})^{\tau}\Big]} \\ &\overset{(e)}{=} \frac{k}{\tilde{k}}\Big(1-\frac{\tilde{k}}{n}\Big)\,\mathsf{H}_{\min}\Big[f\Big(1-\frac{\tilde{k}}{n}\Big)\Big(\frac{\tilde{k}}{n}\Big)^{\mu-1} \\ &\quad+\frac{1}{n^{\mu}}\Big(\sum\nolimits_{\tau=0}^{\mu}\binom{\mu}{\tau}\tilde{k}^{\mu-\tau}(n-\tilde{k})^{\tau}-\mu\tilde{k}^{\mu-1}(n-\tilde{k})-\tilde{k}^{\mu}\Big) \\ &\quad-\frac{1}{n^{\mu}}\sum\nolimits_{\tau=2}^{\mu}\binom{\mu-\mathsf{M}^{\mathrm{c}}_g(f)}{\tau}\tilde{k}^{\mu-\tau}(n-\tilde{k})^{\tau}\Big]^{-1} \\ &\overset{(f)}{=} \frac{k}{\tilde{k}}\Big(1-\frac{\tilde{k}}{n}\Big)\,\mathsf{H}_{\min}\Big[f\Big(1-\frac{\tilde{k}}{n}\Big)\Big(\frac{\tilde{k}}{n}\Big)^{\mu-1} \\ &\quad+\frac{1}{n^{\mu}}\Big(n^{\mu}-\mu\tilde{k}^{\mu-1}(n-\tilde{k})-\tilde{k}^{\mu}\Big) \\ &\quad-\frac{1}{n^{\mu}}\Big(\sum_{\tau=0}^{\eta}\binom{\eta}{\tau}\tilde{k}^{\mu-\tau}(n-\tilde{k})^{\tau}-\eta\tilde{k}^{\mu-1}(n-\tilde{k})-\tilde{k}^{\mu}\Big)\Big]^{-1} \\ &= \frac{k}{\tilde{k}}\Big(1-\frac{\tilde{k}}{n}\Big)\,\mathsf{H}_{\min}\Big[f\Big(1-\frac{\tilde{k}}{n}\Big)\Big(\frac{\tilde{k}}{n}\Big)^{\mu-1}-\mu\Big(1-\frac{\tilde{k}}{n}\Big)\Big(\frac{\tilde{k}}{n}\Big)^{\mu-1} \\ &\quad+1-\Big(\frac{\tilde{k}}{n}\Big)^{\mu}-\frac{1}{n^{\mu}}\Big(\tilde{k}^{\mu-\eta}\sum\nolimits_{\tau=0}^{\eta}\binom{\eta}{\tau}\tilde{k}^{\eta-\tau}(n-\tilde{k})^{\tau}\Big) \\ &\quad+\eta\Big(1-\frac{\tilde{k}}{n}\Big)\Big(\frac{\tilde{k}}{n}\Big)^{\mu-1}+\Big(\frac{\tilde{k}}{n}\Big)^{\mu}\Big]^{-1} \\ &= \frac{k}{\tilde{k}}\Big(1-\frac{\tilde{k}}{n}\Big)\,\mathsf{H}_{\min}\Big[1+(f-\mu+\eta)\Big(1-\frac{\tilde{k}}{n}\Big)\Big(\frac{\tilde{k}}{n}\Big)^{\mu-1} \\ &\quad-\frac{1}{n^{\mu}}\Big(\tilde{k}^{\mu-\eta}n^{\eta}\Big)\Big]^{-1} \\ &= \frac{\frac{k}{\tilde{k}}\big(1-\frac{\tilde{k}}{n}\big)\,\mathsf{H}_{\min}}{1-\big(\frac{\tilde{k}}{n}\big)^{\mathsf{M}^{\mathrm{c}}_g(f)}-(\mathsf{M}^{\mathrm{c}}_g(f)-f)\big(1-\frac{\tilde{k}}{n}\big)\big(\frac{\tilde{k}}{n}\big)^{\mu-1}}, \end{aligned}$$

where (c) follows since $\nu > \kappa$; (d) holds since we have $\kappa/\nu = \tilde{k}/n$ from Definition 7; (e) follows from expanding the summation over the terms of $\rho(\mu,\tau)$; and (f) follows by defining $\eta \triangleq \mu - \mathsf{M}^{\mathrm{c}}_g(f)$ and the fact that $\binom{m}{n} = 0$ if $m < n$. ■

Corollary 1: Consider a DSS that uses an $[n, k]$ RS code $\mathscr{C}$ to store f messages over n noncolluding databases using Lagrange encoding. Let $\mu \in [f : \mu_g(f)]$ be the number of candidate polynomials evaluations of degree at most g, including the f independent messages. Then, the PPC rate

$$\mathsf{R}_{\mathrm{PPC},\infty} = \frac{k}{n}\left(\frac{\max\{n-g(k-1)-1, 0\}}{g(k-1)+1}\right)\mathsf{H}_{\min} \tag{7}$$

is achievable as $f \to \infty$.

Proof: If $n \leq g(k-1)+1$, then it follows from (5) that the PPC rate approaches zero as $f \to \infty$, which is in accordance with (7). Otherwise, if $n > g(k-1)+1$, the result follows directly from (5) by taking the limit $f \to \infty$ and using the fact that $\tilde{k} \overset{(a)}{=} \min\{g(k-1)+1, n\} = g(k-1)+1 < n$, where (a) follows from Proposition 1. ■

Note that the asymptotic PPC rate in (7) is equal to the rate of the general scheme from [20] when $\mathsf{H}_{\min} = 1$. This difference is due to the simplified rate definition used in [20]. Moreover, our proposed scheme cannot be obtained using the concept of refinement and lifting of so-called one-shot schemes as introduced for PIR in [28], since this concept cannot readily be applied to the function computation case.

Remark 3: Note that in Lemma 1 and Theorem 2 we assume that the set of μ candidate functions includes its monomial basis which at least consists of the f independent files, i.e., $\{\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(f)}\} \subseteq \{\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(\mu)}\}$ and $\mu \geq f$. However, for the PPC problem where this is not the case, one can see that the PPC rate

$$\mathsf{R}_{\mathrm{PPC}} = \begin{cases} \dfrac{1}{f}\,\mathsf{H}_{\min} & \text{if } n \leq g(k-1)+1, \\[2ex] \dfrac{\frac{k}{\tilde{k}}\left(1-\frac{\tilde{k}}{n}\right)\mathsf{H}_{\min}}{1-\left(\frac{\tilde{k}}{n}\right)^{\mu}} & \text{otherwise} \end{cases}$$

is achievable with our general PPC scheme for RS-coded DSSs based on (4). *Moreover, Corollary 1 holds when $\mu \to \infty$.*

V. PPC Scheme for Systematic RS-Encoded DSSs

In this section, we consider the case of RS-coded DSSs with systematic Lagrange encoding and first adapt the concept of the PPC achievable rate matrix from Definition 7.

A. PPC Systematic Achievable Rate Matrix

In contrast to the PPC scheme in Section IV, the basic idea is to utilize the systematic part of the RS code to recover the requested polynomial evaluation directly, i.e., we do not need to interpolate the systematic downloaded symbols to determine the requested polynomial evaluation. Thus, we can further enhance the download rate. However, due to the generic PC query design principles, namely, message symmetry and side information exploitation, we are restricted in how to exploit side information obtained from the systematic nodes. Specifically, for decodability (side information cancellation) to be possible, the side information obtained from the systematic nodes must be utilized in an isolated manner within an information set of the *polynomial decoding code* (see Section IV-A), such that we can reverse the order of the decoding procedure (i.e., unlike our RS-coded PPC scheme, we interpolate first and then cancel the side information). This restriction is further illustrated by a careful construction of a PPC systematic achievable rate matrix (Definition 10 below) and the corresponding interference matrices. Moreover, we modify the general PPC scheme to utilize only the necessary number of nodes, denoted by $\hat{n}$, that guarantee the isolated use of systematic side information. Accordingly, we introduce an achievable rate matrix for the systematic PPC scheme as follows.

Definition 10: Let $\mathscr{C}$ be an arbitrary $[n,k]$ code and denote by $\tilde{\mathscr{C}} = \mathscr{C}^{\star g}$ the $\tilde{k}$-dimensional code generated by the g-fold star-product of $\mathscr{C}$ with itself. Moreover, let[2]

$$\hat{n} \triangleq \begin{cases} n & \text{if } \left\lfloor \frac{n}{\tilde{k}} \right\rfloor = 1 \text{ and } n - \left\lfloor \frac{n}{\tilde{k}} \right\rfloor \tilde{k} < k, \\[1ex] k + \left(\left\lfloor \frac{n}{\tilde{k}} \right\rfloor - 1\right)\tilde{k} & \text{if } \left\lfloor \frac{n}{\tilde{k}} \right\rfloor > 1 \text{ and } n - \left\lfloor \frac{n}{\tilde{k}} \right\rfloor \tilde{k} < k, \\[1ex] k + \left\lfloor \frac{n}{\tilde{k}} \right\rfloor \tilde{k} & \text{if } \left\lfloor \frac{n}{\tilde{k}} \right\rfloor \geq 1 \text{ and } n - \left\lfloor \frac{n}{\tilde{k}} \right\rfloor \tilde{k} \geq k. \end{cases} \tag{8}$$

Then, a $\nu \times \hat{n}$ binary matrix $\Lambda^{\mathrm{S,PPC}}_{\kappa,\nu}$ is called a PPC systematic achievable rate matrix *for $(\mathscr{C}, \tilde{\mathscr{C}})$ if the following conditions are satisfied.*

1) *$\Lambda^{\mathrm{S,PPC}}_{\kappa,\nu}$ is a κ-column regular matrix, and*
2) *there are exactly $\varrho \triangleq \lfloor \hat{n}/\tilde{k} \rfloor \kappa$ rows $\{\boldsymbol{\lambda}_i\}_{i \in [\varrho]}$ and $\nu - \varrho$ rows $\{\boldsymbol{\lambda}_{i+\varrho}\}_{i \in [\nu-\varrho]}$ of $\Lambda^{\mathrm{S,PPC}}_{\kappa,\nu}$ such that $\forall\, i \in [\varrho]$, $\chi(\boldsymbol{\lambda}_i)$ contains an information set for $\tilde{\mathscr{C}}$ and $\forall\, i \in [\nu-\varrho]$, $\chi(\boldsymbol{\lambda}_{i+\varrho}) = [k]$.*

The following lemma shows how to construct a PPC systematic achievable rate matrix with $(\kappa, \nu) = \bigl(k, \hat{n} - \lfloor \hat{n}/\tilde{k} \rfloor (\tilde{k}-k)\bigr)$.

Lemma 2: Let $\mathscr{C}$ be an arbitrary $[n,k]$ code and $\tilde{\mathscr{C}} = \mathscr{C}^{\star g}$. Then, there exists a PPC systematic achievable rate matrix $\Lambda^{\mathrm{S,PPC}}_{\kappa,\nu}$ for $(\mathscr{C}, \tilde{\mathscr{C}})$ with $(\kappa, \nu) = \bigl(k, \hat{n} - \lfloor \hat{n}/\tilde{k} \rfloor (\tilde{k}-k)\bigr)$, where $\tilde{k}$ is the dimension of $\tilde{\mathscr{C}}$.

Proof: Let $\hat{\delta} \triangleq \lfloor \hat{n}/\tilde{k} \rfloor$ and $\Gamma \triangleq \hat{n} - \hat{\delta}\tilde{k}$. From our choices of $\hat{n}$ in (8), one can verify that $\Gamma \leq k$ and Γ is well-defined. Accordingly, construct a matrix $\mathsf{A}_{k \times \hat{n}}$ as in Definition 8 with

$$a_{i,j} = \hat{\delta}k + i, \quad \text{if } j \in [k],\ i \in [\Gamma]. \tag{9}$$

In this way, $k\Gamma$ entries of $\mathsf{A}_{k \times \hat{n}}$ are filled. Next, let $\Bigl\{a_{i_1^{(j)},j}, \ldots, a_{i_{u(j)}^{(j)},j}\Bigr\}$, $j \in [\hat{n}]$, denote the remaining empty entries in column j of $\mathsf{A}_{k \times \hat{n}}$, where $u(j) \leq k$ is the number of empty entries in column j. Hence, the $k\hat{n} - k\Gamma = k(\hat{n} - \Gamma)$ entries

$$\Bigl\{a_{i_1^{(1)},1}, \ldots, a_{i_{u(1)}^{(1)},1}, \ldots, a_{i_1^{(\hat{n})},\hat{n}}, \ldots, a_{i_{u(\hat{n})}^{(\hat{n})},\hat{n}}\Bigr\} \tag{10}$$

are empty. Now, observe that $(\hat{n}-\Gamma)\hat{\delta}^{-1} = \bigl(\hat{n}-(\hat{n}-\hat{\delta}\tilde{k})\bigr)\hat{\delta}^{-1} = \tilde{k} \in \mathbb{N}$. By consecutively assigning $\{1, \ldots, \hat{\delta}k\}$ to the entries of $\mathsf{A}_{k \times \hat{n}}$ in (10) and repeating this process $\tilde{k}$ times, the remaining $\hat{\delta}k \cdot (\hat{n}-\Gamma)/\hat{\delta} = k(\hat{n}-\Gamma)$ empty entries of $\mathsf{A}_{k \times \hat{n}}$ are filled. Note that since values of $[\hat{\delta}k]$ are consecutively assigned, the largest number of empty entries of each column of $\mathsf{A}_{k \times \hat{n}}$ is k, and $\hat{\delta} = \lfloor \hat{n}/\tilde{k} \rfloor \geq 1$, there are no repeated values of $[\hat{\delta}k]$ in any column of $\mathsf{A}_{k \times \hat{n}}$, which implies that condition 1) in Definition 10 is satisfied. From (9) and (10), it can be seen that each $a \in [\hat{\delta}k] = [\varrho]$ occurs in $\tilde{k}$ columns of $\mathsf{A}_{k \times \hat{n}}$ and each $a \in [\hat{\delta}k + 1 : \hat{\delta}k + \Gamma]$ occurs in k columns of $\mathsf{A}_{k \times \hat{n}}$. This implies that condition 2) in Definition 10 is satisfied with $\kappa = k$, $\varrho = \hat{\delta}k$, and $\nu = \Gamma + \hat{\delta}k$, which completes the proof. ■

[2] Note that the first requirement of the final case of (8) is unnecessary as $\lfloor n/\tilde{k} \rfloor \geq 1$ always. However, it is included for symmetry reasons.

Lemma 3: For the PPC systematic achievable rate matrix from Lemma 2, it holds that

$$\nu = \begin{cases} n - \tilde{k} + k & \text{if } \left\lfloor \frac{n}{\tilde{k}} \right\rfloor = 1 \text{ and } n - \left\lfloor \frac{n}{\tilde{k}} \right\rfloor \tilde{k} < k, \\ \left\lfloor \frac{n}{\tilde{k}} \right\rfloor k & \text{if } \left\lfloor \frac{n}{\tilde{k}} \right\rfloor > 1 \text{ and } n - \left\lfloor \frac{n}{\tilde{k}} \right\rfloor \tilde{k} < k, \\ \left\lfloor \frac{n}{\tilde{k}} \right\rfloor k + k & \text{if } \left\lfloor \frac{n}{\tilde{k}} \right\rfloor \geq 1 \text{ and } n - \left\lfloor \frac{n}{\tilde{k}} \right\rfloor \tilde{k} \geq k. \end{cases} \quad (11)$$

Proof: To prove the results, we use Definition 10 and the fact that $\nu = \hat{n} - \lfloor \hat{n}/\tilde{k} \rfloor (\tilde{k} - k)$. Now, if $\lfloor n/\tilde{k} \rfloor = 1$ and $n - \lfloor n/\tilde{k} \rfloor \tilde{k} < k$ (the first case from Definition 10), then it follows directly that $\nu = \hat{n} - \lfloor \hat{n}/\tilde{k} \rfloor (\tilde{k} - k) = n - \lfloor n/\tilde{k} \rfloor (\tilde{k} - k) = n - \tilde{k} + k$. On the other hand, if $\lfloor n/\tilde{k} \rfloor > 1$ and $n - \lfloor n/\tilde{k} \rfloor \tilde{k} < k$ (the second case from Definition 10), then after inserting $\hat{n} = k + \left(\lfloor n/\tilde{k} \rfloor - 1\right)\tilde{k}$ into the expression for ν, $\nu = k\lfloor n/\tilde{k} \rfloor - \lfloor k/\tilde{k} \rfloor (\tilde{k} - k) = k\lfloor n/\tilde{k} \rfloor$, since $\lfloor k/\tilde{k} \rfloor (\tilde{k} - k) = 0$. In a similar manner, the remaining case in (11) can be shown. ■

In the following lemma, we show a lower bound to the fraction κ/ν.

Lemma 4: If a matrix $\Lambda^{\mathrm{S,PPC}}_{\kappa,\nu}(\mathscr{C}, \tilde{\mathscr{C}})$ exists for an $[n, k]$ code $\mathscr{C}$ and the $[n, \tilde{k}]$ code $\tilde{\mathscr{C}}$, then we have $\kappa/\nu \geq k/\left(\hat{n} - \lfloor \hat{n}/\tilde{k} \rfloor (\tilde{k} - k)\right)$.

Proof: Since by definition each row $\boldsymbol{\lambda}_i$ of $\Lambda^{\mathrm{S,PPC}}_{\kappa,\nu}$ contains an information set for $\tilde{\mathscr{C}}$, $i \in [\varrho]$, $\varrho = \lfloor \hat{n}/\tilde{k} \rfloor \kappa$, and each row $\boldsymbol{\lambda}_{i+\varrho} = [k]$, $i \in [\nu - \varrho]$, we have $w_{\mathrm{H}}(\boldsymbol{\lambda}_i) \geq \tilde{k}$, $i \in [\varrho]$, and $w_{\mathrm{H}}\left(\boldsymbol{\lambda}_{i+\varrho}\right) = k$, $i \in [\nu - \varrho]$. Let $\boldsymbol{v}_j$, $j \in [\hat{n}]$, be the j-th column of $\Lambda^{\mathrm{S,PPC}}_{\kappa,\nu}$. If we look at $\Lambda^{\mathrm{S,PPC}}_{\kappa,\nu}$ from both a row-wise and a column-wise point of view, we obtain

$$\begin{aligned} \varrho\tilde{k} + (\nu - \varrho)k &\leq \sum_{i=1}^{\varrho} w_{\mathrm{H}}(\boldsymbol{\lambda}_i) + \sum_{i=1}^{\nu-\varrho} w_{\mathrm{H}}\left(\boldsymbol{\lambda}_{i+\varrho}\right) \\ &= \sum_{j=1}^{\hat{n}} w_{\mathrm{H}}\left(\boldsymbol{v}_j\right) = \kappa\hat{n}. \end{aligned}$$

Thus, we have

$$\varrho\tilde{k} - \varrho k + \nu k = \varrho(\tilde{k} - k) + \nu k \leq \kappa\hat{n},$$

from which the result follows. ■

The systematic PPC scheme requires the length of each message to be $\mathsf{L} = \nu^{\mu} \cdot k$. The queries $Q^{(v)}_j$ are generated by setting $(\kappa, \nu) = (k, \hat{n} - \lfloor \hat{n}/\tilde{k} \rfloor (\tilde{k} - k))$ and invoking the query generation algorithm `Q-Gen` of [18, Sec. IV-A] with the systematic PPC problem parameters as follows:

$$\{Q^{(v)}_1, \ldots, Q^{(v)}_{\hat{n}}\} \leftarrow \texttt{Q-Gen}(v, \mu, \kappa, \nu, \hat{n}, \mathsf{A}_{\kappa\times\hat{n}}, \mathsf{B}_{(\nu-\kappa)\times\hat{n}}).$$

Note that we utilize $\hat{n} \leq n$ databases, including the systematic nodes, in constructing the scheme, while the remaining $n - \hat{n}$ databases are not queried.

B. Sign Assignment and Redundancy Elimination

Since this scheme is a modified version of the general PPC scheme where we utilize the systematic part of the RS code to recover the requested polynomial evaluation directly, the scheme inherently extends the same redundancy and sign assignment arguments stated in Section IV-D. The only difference between the general PPC scheme and the systematic PPC scheme lies within the recovery argument.

C. Recovery and Privacy

The scheme works as the PPC scheme in Section IV, however by mixing between the code $\tilde{\mathscr{C}}$ and the storage code $\mathscr{C}$. Due to this mixture, we require a more complicated decoding process. The key idea of the recovery process of the scheme is illustrated with Example 2 in Section V-E.

Remark 4: The systematic scheme above reduces to the systematic PPC scheme presented in [1] if and only if $n - \tilde{k} \leq k$.[3] *In particular, this happens if and only if the storage code rate $k/n \geq k/(k + g(k-1) + 1)$. Otherwise, $\hat{n}$ is smaller than n and the PPC rate becomes larger than the one for the systematic scheme in [1].*

Remark 4 can be easily verified with the following argument. The two schemes are equivalent *if and only if* $n = \hat{n}$ and $\nu = k + \min\{k, n - \tilde{k}\}$ (see [1, Thm. 2]). Assume that $n - \tilde{k} \leq k$. Then, $1 \leq \lfloor n/\tilde{k} \rfloor \leq \lfloor 1 + k/\tilde{k} \rfloor \leq 2$. If $\lfloor n/\tilde{k} \rfloor = 1$, then it follows directly from (8) and Lemma 3 that $n = \hat{n}$ and $\nu = k + n - \tilde{k} = k + \min\{k, n - \tilde{k}\}$. Otherwise, if $\lfloor n/\tilde{k} \rfloor = 2$, then $k = \tilde{k}$, $3k > n \geq 2k$, and from (8), we have $\hat{n} = k + \tilde{k} = 2k$. Since, by assumption, we have $n - \tilde{k} \leq k$, it follows that $n \leq k + \tilde{k} = 2k$. Combining the two inequalities over n, specifically, $3k > n \geq 2k$ and $n \leq 2k$, we conclude that $n = 2k$ and it holds that $n = \hat{n}$. Now, from Lemma 3, $\nu = 2k = k + \min\{k, n - \tilde{k}\}$, and the equivalence of the two schemes follows. The "only-if" part follows in a similar manner. Finally, the lower bound on the storage code rate follows directly from the condition $n - \tilde{k} \leq k$.

D. Achievable PPC Rate

Using Lemmas 1 and 2, the following theorem follows.

Theorem 3: Consider a DSS that uses an $[n, k]$ RS code $\mathscr{C}$ to store f messages over n noncolluding databases using systematic Lagrange encoding. Let $\mu \in [f : \mu_g(f)]$ be the number of candidate polynomials evaluations of degree at most g, including the f independent messages. Then, the PPC rate

$$\mathsf{R}^{\mathrm{S}}_{\mathrm{PPC}} = \begin{cases} \dfrac{1}{f}\,\mathrm{H}_{\min} \quad \text{if } n \leq g(k-1) + 1, \\ \dfrac{\frac{k}{\hat{n}}\left(\frac{\nu-\kappa}{\kappa}\right)\mathrm{H}_{\min}}{1 - \left(\frac{\kappa}{\nu}\right)^{\mathsf{M}^{\mathrm{c}}_g(f)} - \left(\mathsf{M}^{\mathrm{c}}_g(f) - f\right)\left(1 - \frac{\kappa}{\nu}\right)\left(\frac{\kappa}{\nu}\right)^{\mu-1}} \\ \qquad \text{otherwise,} \end{cases} \quad (12)$$

with $(\kappa, \nu) = (k, \hat{n} - \lfloor \hat{n}/\tilde{k} \rfloor (\tilde{k} - k))$ and $\hat{n}$ as defined in (8), is achievable.

Proof: From (4) and by removing redundant τ-sums from the query sets according to Lemma 1, the achievable PPC rate becomes

$$\begin{aligned} \mathsf{R} &\overset{(a)}{=} \frac{k\nu^{\mu}\,\mathrm{H}_{\min}}{\hat{n}\left(\binom{\mu}{1} - \binom{\mu-f}{1}\right)\kappa^{\mu} + \hat{n}\sum\limits_{\tau=2}^{\mu}\rho(\mu,\tau)\kappa^{\mu-\tau+1}(\nu-\kappa)^{\tau-1}} \\ &= \frac{k\nu^{\mu}\,\mathrm{H}_{\min}}{\hat{n}\kappa\left[f\kappa^{\mu-1} + \sum_{\tau=2}^{\mu}\rho(\mu,\tau)\kappa^{\mu-\tau}(\nu-\kappa)^{\tau-1}\right]}, \end{aligned} \quad (13)$$

where (a) follows from the PPC rate in Definition 1, (4), and Lemma 1.

[3] Note that there is a subtle difference since more τ-sum types for $\tau > 1$ are potential identified as being redundant (depending on the actual candidate set) by using $\mathsf{M}^{\mathrm{c}}_g(f)$ in Lemma 1 than in [1, Lem. 1], which uses $\mathsf{M}_g(f)$.

Now, we first consider the case where $\nu = \kappa$ and show that it is equivalent to $n \leq g(k-1)+1$. Assume that $\nu = \kappa = k$. Then, for the first case of (11) it follows that $\tilde{k} = n$. For the second and third cases of (11), to obtain $\nu = k$, we must have $\lfloor n/\tilde{k} \rfloor = 1$ or $\lfloor n/\tilde{k} \rfloor = 0$, respectively, which violates the condition of the second case and is never true for the third case. Since, by Proposition 1, $\tilde{k} = \min\{g(k-1)+1, n\} = n$, it follows that $n \leq g(k-1)+1$. Conversely, if $n \leq g(k-1)+1$, then $\tilde{k} = \min\{g(k-1)+1, n\} = n$, and it follows from (11) (the first case) that $\nu = \kappa$. Hence, in summary, we have shown that $\nu = \kappa$ is equivalent to $n \leq g(k-1)+1$. As a result, for $n \leq g(k-1)+1$, it follows directly from (13) that $\mathsf{R} = k\,\mathsf{H}_{\min}/\hat{n}f$. Moreover, it can be seen in this case that the proposed systematic PPC scheme reduces to the trivial scheme for which all the f independent files are downloaded and the desired polynomial evaluation is performed offline. However, similar to the general PPC scheme, the proposed systematic PPC scheme requires an unnecessarily high redundancy to decode the f files, i.e., $\tilde{k} = \hat{n}$ instead of $\tilde{k} = k$. As a result, for the case of $n \leq g(k-1)+1$, we again opt out of any other achievable scheme and achieve the PPC rate $\mathsf{H}_{\min}/f$ by simply downloading all f files and performing the desired polynomial evaluation offline.

On the other hand, if $\nu > \kappa$, or equivalently, $n > g(k-1)+1$, then from (13) we have

$$\begin{aligned} \mathsf{R} &\overset{(b)}{=} \frac{\frac{k}{\hat{n}\kappa}\,\mathsf{H}_{\min}}{\frac{f\kappa^{\mu-1}}{\nu^{\mu}} + \frac{1}{\nu^{\mu}(\nu-\kappa)}\sum_{\tau=2}^{\mu}\rho(\mu,\tau)\kappa^{\mu-\tau}(\nu-\kappa)^{\tau}} \\ &= \frac{\frac{k(\nu-\kappa)}{\hat{n}\kappa}\,\mathsf{H}_{\min}}{\frac{f(\nu-\kappa)}{\nu}\left(\frac{\kappa}{\nu}\right)^{\mu-1} + \frac{1}{\nu^{\mu}}\sum_{\tau=2}^{\mu}\rho(\mu,\tau)\kappa^{\mu-\tau}(\nu-\kappa)^{\tau}} \\ &\;\;\vdots \\ &\overset{(c)}{=} \frac{\frac{k}{\hat{n}}\left(\frac{\nu-\kappa}{\kappa}\right)\mathsf{H}_{\min}}{1-\left(\frac{\kappa}{\nu}\right)^{\mathsf{M}_g^{\mathrm{c}}(f)} - \left(\mathsf{M}_g^{\mathrm{c}}(f)-f\right)\left(1-\frac{\kappa}{\nu}\right)\left(\frac{\kappa}{\nu}\right)^{\mu-1}}, \end{aligned}$$

where (b) follows since $\nu > k$ and (c) results from following similar steps as in the proof of the achievable PPC rate of Theorem 2 in Section IV-F. ∎

Corollary 2: Consider a DSS that uses an $[n,k]$ RS code $\mathscr{C}$ to store f messages over n noncolluding databases using systematic Lagrange encoding. Let $\mu \in [f : \mu_g(f)]$ be the number of candidate polynomials evaluations of degree at most g, including the f independent messages. Then, the PPC rate

$$\mathsf{R}^{\mathrm{S}}_{\mathrm{PPC},\infty} = \begin{cases} \frac{1}{n}\left(\max\{n-g(k-1)-1,0\}\right)\mathsf{H}_{\min} \\ \textit{if } \left\lfloor \frac{n}{\tilde{k}} \right\rfloor = 1 \;\; \textit{and} \;\; n - \left\lfloor \frac{n}{\tilde{k}} \right\rfloor \tilde{k} < k, \\ \frac{1}{\hat{n}}\left(\left\lfloor \frac{n}{g(k-1)+1} \right\rfloor k - k\right)\mathsf{H}_{\min} \\ \textit{if } \left\lfloor \frac{n}{\tilde{k}} \right\rfloor > 1 \;\; \textit{and} \\ n - \left\lfloor \frac{n}{g(k-1)+1} \right\rfloor (g(k-1)+1) < k, \\ \frac{1}{\hat{n}}\left(\left\lfloor \frac{n}{g(k-1)+1} \right\rfloor k\right)\mathsf{H}_{\min} \\ \textit{if } \left\lfloor \frac{n}{\tilde{k}} \right\rfloor \geq 1 \;\; \textit{and} \\ n - \left\lfloor \frac{n}{g(k-1)+1} \right\rfloor (g(k-1)+1) \geq k, \end{cases} \tag{14}$$

with $\hat{n}$ as defined in (8)*, is asymptotically achievable for $f \to \infty$.*

Proof: If $n \leq g(k-1)+1$, then it follows from (12) that the PPC rate approaches zero as $f \to \infty$, which is in accordance with (14) (first case, since $\lfloor n/\tilde{k} \rfloor = 1$ and $n - \lfloor n/\tilde{k} \rfloor \tilde{k} = 0 < k$). Otherwise, if $n > g(k-1)+1$, the result follows directly from (12) by taking the limit $f \to \infty$ and using (11) and the fact (see Proposition 1) that $\tilde{k} = \min\{g(k-1)+1, n\} = g(k-1)+1$. ∎

Note that when $n - \tilde{k} \leq k$, the asymptotic PPC rate in (14) is equal to the rate of the systematic scheme from [16, Thm. 3], [20] when $\mathsf{H}_{\min} = 1$. This difference is due to the simplified rate definition used in [16], [20]. However, for the case when $n - \tilde{k} > k$, with the simplified rate definition, i.e., for $\mathsf{H}_{\min} = 1$, the asymptotic PPC rate in (14) is larger compared to the PPC rate of the systematic scheme from [16, Thm. 3], [20]. See also Remark 4.

Remark 5: Similar to Remark 3, in Theorem 3 we assume that the set of μ candidate functions includes its monomial basis which at least consists of the f independent files, i.e., $\{\mathbf{W}^{(1)},\ldots,\mathbf{W}^{(f)}\} \subseteq \{\mathbf{X}^{(1)},\ldots,\mathbf{X}^{(\mu)}\}$ and $\mu \geq f$. However, for the PPC problem where this is not the case, one can see that the PPC rate

$$\mathsf{R}^{\mathrm{S}}_{\mathrm{PPC}} = \begin{cases} \frac{1}{f}\,\mathsf{H}_{\min} & \textit{if } n \leq g(k-1)+1, \\ \frac{\frac{k}{\hat{n}}\left(\frac{\nu-\kappa}{\kappa}\right)\mathsf{H}_{\min}}{1-\left(\frac{\kappa}{\nu}\right)^{\mu}} & \textit{otherwise}, \end{cases}$$

with $(\kappa,\nu) = (k, \hat{n} - \lfloor \hat{n}/\tilde{k} \rfloor(\tilde{k}-k))$ and $\hat{n}$ as defined in (8), is achievable with our PPC scheme for RS-coded DSSs with systematic Lagrange encoding based on (4)*. Moreover, Corollary 2 holds when $\mu \to \infty$.*

We illustrate the key concept of our proposed scheme in Theorem 3 with an example for the special case of PMC.

E. Special Case: PMC Scheme

As the rate of PMC is a decreasing function of the number of candidate monomials, we can increase the PMC rate by limiting ourselves to the set of monomials excluding *parallel* monomials. To this end, we define a parallel monomial as a monomial resulting from raising another monomial to a positive integer power, i.e., to $\{\boldsymbol{W}^{\boldsymbol{i}} : \boldsymbol{i} \in \mathbb{N}_0^f,\ 1 \leq \mathsf{wt}(\boldsymbol{i}) \leq g,\ \boldsymbol{i} \mid p,\ p \in \mathcal{P}_g\}$. Here, $\mathcal{P}_g$ denotes the set of prime numbers less or equal to g and $\boldsymbol{i} = (i_1,\ldots,i_f) \mid p$ means that all nonzero i_j, $j \in [f]$, are divisors of p. For example, for a bivariate monomial over the variables x and y of degree at most $g = 2$ the set of possible monomials is $\{x, y, xy, x^2, y^2\}$. Note that x^2 is a parallel monomial as it can be obtained by raising the monomial x to the power of 2. Thus, x^2 and y^2 are parallel monomials and can be excluded from the set of candidate monomials. Denote by $\mathcal{P} = \{p_1,\ldots,p_{|\mathcal{P}|}\}$ an arbitrary nonempty subset of $\mathcal{P}_g$. By applying the Legendre formula for counting the prime numbers less or equal to g, we obtain the number of nonparallel monomials as

$$\begin{aligned} \widetilde{\mathsf{M}}_g(f) &= \mathsf{M}_g(f) \\ &\quad + \sum_{\substack{\forall\mathcal{P}\subseteq\mathcal{P}_g:\mathcal{P}\neq\emptyset,\\ p_1\cdots p_{|\mathcal{P}|}\leq g}} (-1)^{|\mathcal{P}|}\left[\binom{\left\lfloor \frac{g}{p_1\cdots p_{|\mathcal{P}|}} \right\rfloor + f}{\left\lfloor \frac{g}{p_1\cdots p_{|\mathcal{P}|}} \right\rfloor} - 1\right]. \end{aligned}$$

TABLE I

PMC QUERY SETS FOR $\upsilon = 1$ AFTER SIGN ASSIGNMENT AND REMOVAL OF REDUNDANT QUERIES FOR THE SYSTEMATIC $[4, 2]$ RS-CODED DSS OF EXAMPLE 2, $f = 2$ MESSAGES, AND $\mu = 3$ CANDIDATE MONOMIAL FUNCTIONS. BLUE AND RED SUBSCRIPTS INDICATE SIDE INFORMATION EXPLOITATION IN ROUNDS $\tau = 2$ AND $\tau = 3$, RESPECTIVELY

j	1	2	3	4
$Q_j^{(1)}(\mathcal{D}; 1)$	$x_{1:4,1}$, $x_{9:12,1}$	$x_{5:8,2}$, $x_{9:12,2}$	$x_{1:4,3}$, $x_{5:8,3}$	$x_{1:4,4}$, $x_{5:8,4}$
$Q_j^{(1)}(\mathcal{U}; 1)$	$y_{1:4,1}$, $y_{9:12,1}$	$y_{5:8,2}$, $y_{9:12,2}$	$y_{1:4,3}$, $y_{5:8,3}$	$y_{1:4,4}$, $y_{5:8,4}$
$Q_j^{(1)}(\mathcal{D}; 2)$	$x_{13:14,1} - y_{5:6,1}$ $x_{15:16,1} - z_{5:6,1}$ $x_{21:22,1} - y_{7:8,1}$ $x_{23:24,1} - z_{7:8,1}$	$x_{17:18,2} - y_{1:2,2}$ $x_{19:20,2} - z_{1:2,2}$ $x_{21:22,2} - y_{3:4,2}$ $x_{23:24,2} - z_{3:4,2}$	$x_{13:14,3} - y_{9:10,3}$ $x_{15:16,3} - z_{9:10,3}$ $x_{17:18,3} - y_{11:12,3}$ $x_{19:20,3} - z_{11:12,3}$	$x_{13:14,4} - y_{9:10,4}$ $x_{15:16,4} - z_{9:10,4}$ $x_{17:18,4} - y_{11:12,4}$ $x_{19:20,4} - z_{11:12,4}$
$Q_j^{(1)}(\mathcal{U}; 2)$	$y_{15:16,1} - z_{13:14,1}$ $y_{23:24,1} - z_{21:22,1}$	$y_{19:20,2} - z_{17:18,2}$ $y_{23:24,2} - z_{21:22,2}$	$y_{15:16,3} - z_{13:14,3}$ $y_{19:20,3} - z_{17:18,3}$	$y_{15:16,4} - z_{13:14,4}$ $y_{19:20,4} - z_{17:18,4}$
$Q_j^{(1)}(\mathcal{D}; 3)$	$x_{25,1} - y_{19,1} + z_{17,1}$ $x_{27,1} - y_{20,1} + z_{18,1}$	$x_{26,2} - y_{15,2} + z_{13,2}$ $x_{27,2} - y_{16,2} + z_{14,2}$	$x_{25,3} - y_{23,3} + z_{21,3}$ $x_{26,3} - y_{24,3} + z_{22,3}$	$x_{25,4} - y_{23,4} + z_{21,4}$ $x_{26,4} - y_{24,4} + z_{22,4}$

Example 2: Consider two messages $\mathbf{W}^{(1)}$ and $\mathbf{W}^{(2)}$ that are stored in a noncolluding DSS using a systematic $[4, 2]$ RS code $\mathscr{C}$. Suppose that the user wishes to obtain a monomial function evaluation $\mathbf{X}^{(\upsilon)}$ from the set of nonparallel monomial functions of degree at most $g = 2$. We have $\mu = \mathsf{M}_2^{\mathrm{c}}(2) = \widetilde{\mathsf{M}}_2(2) = 3$, $\upsilon \in [3]$, and the candidate set of monomial functions evaluations is $\{\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \mathbf{W}^{(1)} \star \mathbf{W}^{(2)}\}$, where $\star$ denotes element-wise multiplication. Let the desired monomial function index be $\upsilon = 1$, i.e., the user wishes to obtain the function evaluation $\mathbf{X}^{(1)} = \mathbf{W}^{(1)}$. We have $\tilde{k} = g(k-1)+1 = 3$ and $\hat{n} = n = 4$. It follows that $\nu = \hat{n} - \lfloor \hat{n}/\tilde{k} \rfloor(\tilde{k} - k) = 3$, $\kappa = k = 2$, $\varrho = \lfloor \hat{n}/\tilde{k} \rfloor \kappa = 2$, and

$$\Lambda_{2,3}^{\mathrm{S,PPC}} = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

is a valid PPC systematic achievable rate matrix (see Lemma 2). We further obtain the PC interference matrices $\mathsf{A}_{2\times 4} = \begin{pmatrix} 1 & 2 & 1 & 1 \\ 3 & 3 & 2 & 2 \end{pmatrix}$ and $\mathsf{B}_{1\times 4} = (2 \;\; 1 \;\; 3 \;\; 3)$ from $\Lambda_{2,3}^{\mathrm{S,PPC}}$ using Definition 8.

We simplify the notation by letting $x_{t,j} = C_{t,j}^{(1)}$, $y_{t,j} = C_{t,j}^{(2)}$, and $z_{t,j} = C_{t,j}^{(1)} \cdot C_{t,j}^{(2)}$ for all $t \in [\beta]$, $j \in [4]$, where $\beta = \nu^{\mu} = 27$. Since the desired function evaluation is $\mathbf{X}^{(1)}$, the goal is to privately obtain $x_{t,j}$, $\forall\, t \in [27]$, and successfully decode $\mathbf{X}^{(1)}$. The construction of the query sets is briefly presented in the following steps.[4]

1) Initialization (Round $\tau = 1$): We start with $\tau = 1$ to generate query sets for each database j holding $\kappa^{\mu} = 8$ instances of $x_{t,j}$. By message symmetry this also applies to $y_{t,j}$ and $z_{t,j}$.

2) Following Rounds ($\tau \in [2:3]$): Using the PC interference matrices $\mathsf{A}_{2\times 4}$ and $\mathsf{B}_{1\times 4}$ for the exploitation of side information for the j-th database, $j \in [n]$, we generate the desired query sets $Q_j^{(1)}(\mathcal{D}; \tau)$ by querying a number of new symbols of the desired monomial jointly combined with symbols from other monomials queried in the previous round from database $i \neq j$. Next, the undesired query sets $Q_j^{(1)}(\mathcal{U}; \tau)$ (if $\tau = 2$) are generated by enforcing message symmetry.

In the end, we apply the sign assignment procedure to the query sets for $\upsilon = 1$ and make the final modification to the queries by removing all the 1-sums corresponding to the redundant 1-sum types from the first round (see Lemma 1). This translates to removing the queries for $z_{t,j}$, since they can be generated offline by the user given $x_{t,j}$ and $y_{t,j}$. The resulting query sets are shown in Table I, where $u_{a:b,j} \triangleq \{u_{a,j}, \ldots, u_{b,j}\}$ for $u = x, y, z$, and the side information is highlighted with blue and red for rounds $\tau = 2$ and $\tau = 3$, respectively. The PMC rate $k\nu^{\mu}\,\mathsf{H}_{\min}/\mathsf{D} = (2 \times 3^3 \times \mathsf{H}_{\min})/(2 \times 4 \times 15) = 0.45 \cdot \mathsf{H}_{\min}$ is achievable, where the value of $\mathsf{H}_{\min} = \mathsf{H}(\mathbf{X}^{(3)})$ depends on the underlying field.

Now we show that the $\mathsf{L} = k\beta = 54$ symbols of the desired function evaluation can be reliably decoded. Note that here we assume that the nodes $j \in \{1, 2\}$ are systematic.

3) Initialization Round ($\tau = 1$): The following steps are taken.

(i) Obtain the desired symbols: From the answers retrieved for the query sets $Q_j^{(1)}(\mathcal{D}, 1)$, utilize the information sets $\tilde{\mathcal{I}}_1 = \{1, 3, 4\}$ and $\tilde{\mathcal{I}}_2 = \{2, 3, 4\}$ of $\tilde{\mathscr{C}}$ to decode the symbols of the desired function evaluation $\mathbf{X}^{(1)}$ for $j \in \{1, 2\}$. In other words, from $x_{1:4,1}$, $x_{1:4,3}$, and $x_{1:4,4}$ we use Lagrange interpolation to obtain $x_{1:4,2}$. Similarly, from $x_{5:8,2}$, $x_{5:8,3}$, and $x_{5:8,4}$ we obtain $x_{5:8,1}$. Finally, from the information set $\mathcal{I} = \{1, 2\}$ of $\mathscr{C}$ we readily have $x_{9:12,1}$ and $x_{9:12,2}$. By the end of this round, we obtain $k\nu(\kappa^{\mu-1}) = 24$ symbols from the desired function evaluation $\mathbf{X}^{(1)}$.

(ii) We prepare the side information symbols retrieved in this round to be used in the next round by the following steps.

[4] With some abuse of notation for the sake of simplicity, the generated queries are sets containing their answers.

TABLE II

DECODED AND COMPUTED SYMBOLS FROM THE PMC QUERY SETS FOR $\upsilon = 1$ FROM TABLE I

j	1	2	3	4
$\tilde{Q}_j^{(1)}(\mathcal{D};1)$	$x_{5:8,1}$	$x_{1:4,2}$	$x_{9:12,3}$	$x_{9:12,4}$
$\tilde{Q}_j^{(1)}(\mathcal{U};1)$	$y_{5:8,1},\ z_{5:8,1}$	$y_{1:4,2},\ z_{1:4,2}$	$y_{9:12,3},\ z_{9:12,3}$	$y_{9:12,4},\ z_{9:12,4}$

(a)

j	1	2
$\tilde{Q}_j^{(1)}(\mathcal{D};2)$	$x_{17:18,1},\ x_{19:20,1}$	$x_{13:14,2},\ x_{15:16,2}$
$\tilde{Q}_j^{(1)}(\mathcal{U};2)$	$y_{19:20,1} - z_{17:18,1}$	$y_{15:16,2} - z_{13:14,2}$

(b)

j	1	2
$\tilde{Q}_j^{(1)}(\mathcal{D};3)$	$x_{25,1},\ x_{27,1}$	$x_{26,2},\ x_{27,2}$
$\tilde{Q}_j^{(1)}(\mathcal{U};3)$	$x_{25,1} + y_{23,1} - z_{21,1}$	$x_{26,2} + y_{24,2} - z_{22,2}$

(c)

First, for the answers of the query sets $Q_j^{(1)}(\mathcal{U},1)$, repeat the previous step to decode the undesired symbols $y_{5:8,1}$ and $y_{1:4,2}$. Next, since in this round, due to redundancy elimination, we retrieve symbols of polynomials of degree one, i.e., symbols from the $f = 2$ independent files, we can use Lagrange interpolation with $k = 2$ symbols from the systematic nodes to obtain coded symbols for $j \notin \{1,2\}$. Accordingly, from $x_{9:12,1}$ and $x_{9:12,2}$ we obtain $x_{9:12,3}$ and $x_{9:12,4}$, and similarly for $y_{9:12,3}$ and $y_{9:12,4}$. Finally, using the dependency between x, y, and z and the available symbols, compute $z_{5:8,1}$, $z_{1:4,2}$, $z_{9:12,3}$, and $z_{9:12,4}$. The obtained symbols are shown in Table II(a).

4) Second Round ($\tau = 2$): The decoding procedure is as follows.

(i) Interference cancellation: Utilize the decoded symbols from the set $\tilde{Q}_j^{(1)}(\mathcal{U},1)$ of Table II(a) to cancel the side information, marked in blue in Table I, from the answers of the query sets $Q_j^{(1)}(\mathcal{D},2)$.

(ii) Obtain the desired symbols: Similar to the first round, utilize the information sets $\tilde{\mathcal{I}}_1 = \{1,3,4\}$ and $\tilde{\mathcal{I}}_2 = \{2,3,4\}$ of $\tilde{\mathscr{C}}$ to decode the symbols of the desired function evaluation $\mathbf{X}^{(1)}$ for $j \in \{1,2\}$ shown in $\tilde{Q}_j^{(1)}(\mathcal{D},2)$ of Table II(b). Together with the symbols directly obtained from $j \in \{1,2\}$, by the end of this round, we would have obtained an additional $k\nu(\binom{\mu-1}{\tau-1}\kappa^{\mu-\tau}(\nu-\kappa)^{\tau-1}) = 24$ symbols from the desired function evaluation.

(iii) We prepare the side information τ-sums retrieved in this round to be used in the next round by repeating the previous step to decode the undesired τ-sums $y_{19:20,1} - z_{17:18,1}$ and $y_{15:16,2} - z_{13:14,2}$ of the query sets $\tilde{Q}_j^{(1)}(\mathcal{U},2)$. Note that, unlike in the previous round, we do not have enough symbols to utilize Lagrange interpolation to re-encode the τ-sums $y_{19:20,3} - z_{17:18,3}$ and $y_{19:20,4} - z_{17:18,4}$ as they represent polynomials of degree strictly larger than one.

5) Final Round ($\tau = 3$): The decoding procedure is as follows.

(i) Interference cancellation: Utilize the decoded τ-sums from the set $\tilde{Q}_j^{(1)}(\mathcal{U},2)$ of Table II(b) to cancel the side information, marked in red in Table I, from the query sets $Q_j^{(1)}(\mathcal{D},3)$ for $j \in \{1,2\}$. As a result we obtain the desired symbols of the set $\tilde{Q}_j^{(1)}(\mathcal{D},3)$ shown in Table II(c).

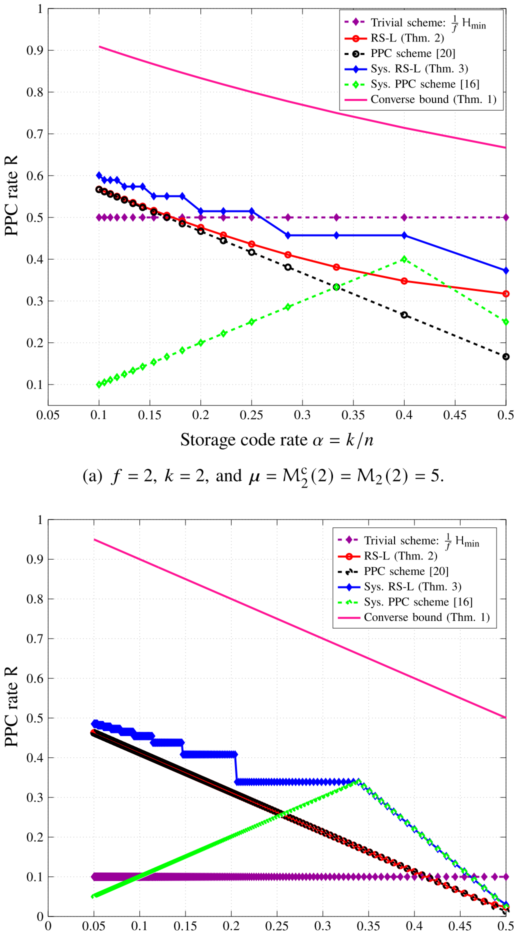
(ii) Generate new symbols: This step is only required when $\hat{n} - \lfloor \hat{n}/\tilde{k} \rfloor \tilde{k} < k$ due to the construction of the interference matrix in the proof of Lemma 2. In particular, the condition is equivalent to $\Gamma < k$. Using the obtained symbols from the previous step, colored in Table II for $\tilde{Q}_j^{(1)}(\mathcal{D},3)$ with blue, along with the side information downloaded in the previous round in $Q_j^{(1)}(\mathcal{U},2)$, generate $\lfloor \hat{n}/\tilde{k} \rfloor \tilde{k} - (n-k) = 1$ new τ-sum with identical indices to the τ-sums retrieved from the nonsystematic nodes. These newly generated symbols are shown in $\tilde{Q}_j^{(1)}(\mathcal{U},3)$.

(iii) Obtain the desired symbols: Here, we reverse the order of operation of the previous rounds where we use Lagrange interpolation first and then cancel the side information. First, utilize the information sets $\tilde{\mathcal{I}}_1 = \{1,3,4\}$ and $\tilde{\mathcal{I}}_2 = \{2,3,4\}$ of $\tilde{\mathcal{C}}$ to decode the τ-sums containing the desired function evaluation for $j \in \{1,2\}$. As a result, we obtain $x_{26,1} + y_{24,1} - z_{22,1}$ and $x_{25,2} + y_{23,2} - z_{21,2}$. Next, cancel the side information from the τ-sums directly obtained from $Q_j^{(1)}(\mathcal{U},2)$ for $j \in \{1,2\}$. Finally, by the end of this round, we would have obtained the final $k\nu(\binom{\mu-1}{\tau-1}\kappa^{\mu-\tau}(\nu-\kappa)^{\tau-1}) = 6$ symbols from the desired function evaluation $\mathbf{X}^{(1)}$.

In summary, the total number of desired function evaluation symbols obtained from this decoding process is $k\nu\sum_{\tau=1}^{\mu}\binom{\mu-1}{\tau-1}\kappa^{\mu-\tau}(\nu-\kappa)^{\tau-1} = k\nu^{\mu} = 54$. $\triangledown$

VI. NUMERICAL RESULTS

In Fig. 2, we compare the PPC rates of Theorems 2 and 3 and those of the schemes from [16], [20] as well as the converse bound from Theorem 1 for various values of the storage code rate $\alpha = k/n$, fixed k, $g = 2$, $f = 2$, $\mu = \mathrm{M}_2^{\mathrm{c}}(2) = \mathrm{M}_2(2) = 5$ for Fig. 2(a), and $f = 10$, $\mu = \mathrm{M}_2^{\mathrm{c}}(10) = \mathrm{M}_2(10) = 65$ for Fig. 2(b). For a small number of files ($f = 2$), the proposed schemes show improved performance for all code rates, while for a relatively large number of files ($f = 10$), the systematic scheme from Theorem 3 shows improved performance up to some code rate (see Remark 4). The converse bound from Theorem 1 shows a relatively large gap for all values of f and storage code rate $\alpha = k/n$. Observe that when neglecting the computational cost at the user, the trivial scheme which downloads all the f files and computes the desired function evaluation offline outperforms all considered PPC schemes when the code rate is above some threshold that depends on both f and g. For $f = 10$ the code rate needs to be close to $1/2$ for the trivial scheme to be the best. Note that the curve for the systematic scheme follows a staircase in which there are $\tilde{k}$ points on each horizontal line of the staircase. This follows directly from the term $\lfloor n/\tilde{k} \rfloor$ in the definition of $\hat{n}$ in (8).



(a) $f = 2$, $k = 2$, and $\mu = \mathsf{M}_2^{\mathrm{c}}(2) = \mathsf{M}_2(2) = 5$.

(b) $f = 10$, $k = 20$, and $\mu = \mathsf{M}_2^{\mathrm{c}}(10) = \mathsf{M}_2(10) = 65$.

Fig. 2. PPC rates as a function of the storage code rate $\alpha = k/n$ for fixed f, k, $g = 2$, and $\mu = \mathsf{M}_2^{\mathrm{c}}(f) = \mathsf{M}_2(f)$. For simplicity, we assume $\mathsf{H}_{\min} = 1$.

VII. Conclusion

For the PPC problem, we have presented two PPC schemes for RS-coded DSSs with Lagrange encoding showing improved computation rates compared to the best known PPC schemes from the literature when the number of messages is small. Asymptotically, as the number of messages tends to infinity, the rate of our RS-coded nonsystematic PPC scheme approaches the rate of the best known nonsystematic PPC scheme. However, for systematically RS-coded DSSs, our scheme significantly outperforms all known PPC schemes up to some specific storage code rate that depends on the maximum degree of the candidate polynomials. Finally, a general converse bound on the PPC rate was derived and compared to the achievable rates of the proposed schemes with some numerical results. The numerical results depicted a gap between the derived converse bound and the achievable rates of the proposed schemes and the best known PPC schemes from literature. Naturally, this gap raises two promising open problems. One is to prove that the converse of Theorem 1 is tight, and the other is to find schemes that exploit the nonlinear dependencies between the candidate functions evaluations, as discussed in [19, Sec. IV] for the uncoded case. Both problems are valuable research directions for future work.

Appendix A
Proof of Theorem 1

In this appendix, we prove the converse bound on the PPC rate presented in Theorem 1. As previously mentioned, the proof follows similarly to the converse proof of [18, Thm. 2]. Denote the set of all queries by $\mathcal{Q} \triangleq \bigl\{Q_j^{(v)} : v \in [\mu], j \in [n]\bigr\}$. It can be shown that for both problems of coded PLC and PPC that use an MDS-PIR capacity-achieving storage code,

$$\begin{aligned}&\mathsf{H}\bigl(A_{[n]}^{(v)} \bigm| \mathbf{X}^{\mathcal{V}}, \mathcal{Q}\bigr)\\&\quad\geq \frac{k}{n}\,\mathsf{H}\bigl(\mathbf{X}^{(v')} \bigm| \mathbf{X}^{\mathcal{V}}\bigr) + \frac{k}{n}\,\mathsf{H}\bigl(A_{[n]}^{(v')} \bigm| \mathbf{X}^{\mathcal{V}}, \mathbf{X}^{(v')}, \mathcal{Q}\bigr), \qquad (15)\end{aligned}$$

where $\mathcal{V} \subseteq [\mu]$ is arbitrary, $v \in \mathcal{V}$, and $v' \in [\mu] \setminus \mathcal{V}$.[5]

Next, since there are in total μ function evaluations, by Definition 6 we can recursively use (15) $r-1$ times with $\mathcal{L} = \{\ell_1, \ldots, \ell_r\} \subseteq [\mu]$ to obtain

$$\begin{aligned}\mathsf{H}\bigl(A_{[n]}^{(\ell_1)} \bigm| \mathbf{X}^{(\ell_1)}, \mathcal{Q}\bigr) &\geq \sum_{v=1}^{r-1}\Bigl(\frac{k}{n}\Bigr)^{v}\,\mathsf{H}\bigl(\mathbf{X}^{(\ell_{v+1})} \bigm| \mathbf{X}^{\{\ell_1,\ldots,\ell_v\}}\bigr)\\&\quad+\Bigl(\frac{k}{n}\Bigr)^{r-1}\,\mathsf{H}\bigl(A_{[n]}^{(\ell_r)} \bigm| \mathbf{X}^{\{\ell_1,\ldots,\ell_r\}}, \mathcal{Q}\bigr)\\&\geq \sum_{v=1}^{r-1}\Bigl(\frac{k}{n}\Bigr)^{v}\,\mathsf{H}\bigl(\mathbf{X}^{(\ell_{v+1})} \bigm| \mathbf{X}^{\{\ell_1,\ldots,\ell_v\}}\bigr), \qquad (16)\end{aligned}$$

where (16) follows from the nonnegativity of entropy. Note that in [15], the authors claim that the general converse for the DPIR problem strongly depends on the chosen permutation of the indices of the candidate functions. Here, we also make a similar observation and assume that the order of indices $\{\ell_1, \ldots, \ell_r\}$ is the permutation that maximizes the summation term of (16) and consider that $\mathbf{X}^{(\ell_1)}$ is the polynomial evaluation with the minimum entropy, i.e., $\mathsf{H}\bigl(\mathbf{X}^{(\ell_1)}\bigr) = \mathsf{L}\,\mathsf{H}_{\min}^{(\mathrm{B})}$. Now,

$$\begin{aligned}&\mathsf{L}\,\mathsf{H}(X^{(\ell_1)})\\&= \mathsf{H}\bigl(\mathbf{X}^{(\ell_1)}\bigr) \overset{(a)}{=} \mathsf{H}\bigl(\mathbf{X}^{(\ell_1)} \bigm| \mathcal{Q}\bigr) - \underbrace{\mathsf{H}\bigl(\mathbf{X}^{(\ell_1)} \bigm| A_{[n]}^{(\ell_1)}, \mathcal{Q}\bigr)}_{=0}\\&= \mathsf{I}\bigl(\mathbf{X}^{(\ell_1)}; A_{[n]}^{(\ell_1)} \bigm| \mathcal{Q}\bigr) = \mathsf{H}\bigl(A_{[n]}^{(\ell_1)} \bigm| \mathcal{Q}\bigr) - \mathsf{H}\bigl(A_{[n]}^{(\ell_1)} \bigm| \mathbf{X}^{(\ell_1)}, \mathcal{Q}\bigr)\\&\leq \mathsf{H}\bigl(A_{[n]}^{(\ell_1)} \bigm| \mathcal{Q}\bigr) - \sum_{v=1}^{r-1}\Bigl(\frac{k}{n}\Bigr)^{v}\,\mathsf{H}\bigl(\mathbf{X}^{(\ell_{v+1})} \bigm| \mathbf{X}^{(\ell_1)}, \ldots, \mathbf{X}^{(\ell_v)}\bigr),\end{aligned} \qquad (17)$$

where (a) holds since any message is independent of the queries $\mathcal{Q}$, and knowing the answers $A_{[n]}^{(\ell_1)}$ and the queries $\mathcal{Q}$, one can determine $\mathbf{X}^{(\ell_1)}$, and (17) follows directly from (16).

Finally, the converse proof is completed by showing that

$$\begin{aligned}\mathsf{R} &= \frac{\mathsf{L}\,\mathsf{H}_{\min}}{\sum_{j=1}^{n}\mathsf{H}\bigl(A_j^{(\ell_1)}\bigr)} \overset{(a)}{\leq} \frac{\mathsf{L}\,\mathsf{H}_{\min}}{\mathsf{H}\bigl(A_{[n]}^{(\ell_1)}\bigr)} \overset{(b)}{\leq} \frac{\mathsf{L}\,\mathsf{H}_{\min}}{\mathsf{H}\bigl(A_{[n]}^{(\ell_1)} \bigm| \mathcal{Q}\bigr)}\\&\leq \frac{\mathsf{H}_{\min}}{\mathsf{H}_{\min}^{(\mathrm{B})} + \sum_{v=1}^{r-1}\bigl(\frac{k}{n}\bigr)^{v}\,\mathsf{H}\bigl(X^{(\ell_{v+1})} \bigm| X^{(\ell_1)}, \ldots, X^{(\ell_v)}\bigr)}, \qquad (18)\end{aligned}$$

where (a) holds because of the chain rule of entropy, (b) is due to the fact that conditioning reduces entropy, and we apply (17) to obtain (18).

[5] Similar derivations can be found in, e.g., [8], [18], [29], [30].

APPENDIX B
PROOF OF LEMMA 1

The proof of Lemma 1 relies on two arguments as follows.

(i) For the first round $\tau = 1$, we can directly eliminate redundant 1-sum types based on both the linear and the nonlinear dependencies between the μ candidate polynomial functions evaluations and the f independent messages. As a result, we have a total of $\mu - f$ redundant 1-sum types regardless of the desired polynomial evaluation.

(ii) For $\tau > 1$, we can represent the PPC problem as an allied PLC problem over the monomial basis of the polynomial candidate set. Let $\{\ell_1, \ldots, \ell_s\} \subseteq [\mu]$ be the set of indices that correspond to the monomial basis, where, for simplicity, $s \triangleq \mathsf{M}_g^{\mathrm{c}}(f)$. Then, $X_l^{(\ell_1)}, \ldots, X_l^{(\ell_s)}$ satisfy $\mathsf{H}\bigl(X_l^{(\ell_1)}, \ldots, X_l^{(\ell_s)}\bigr) = \mathsf{H}\bigl(X_l^{[\mu]}\bigr)$, $\forall l \in [\mathsf{L}]$. Without loss of generality, we can order the candidate polynomial functions by monomials first and then according to their degree, i.e., $(X_l^{(1)}, \ldots, X_l^{(s)}) = (X_l^{(\ell_1)}, \ldots, X_l^{(\ell_s)})$, $\forall l \in [\mathsf{L}]$. Accordingly, the candidate functions evaluations are represented in terms of the monomial basis evaluations with a deterministic linear mapping $\hat{\mathsf{V}}_{\mu \times \mathsf{M}_g^{\mathrm{c}}(f)}$ of size $\mu \times \mathsf{M}_g^{\mathrm{c}}(f)$, for all $l \in [\mathsf{L}]$, as $\bigl(X_l^{(1)}, \ldots, X_l^{(\mu)}\bigr)^{\mathsf{T}} = \hat{\mathsf{V}}_{\mu \times s}\bigl(X_l^{(\ell_1)}, \ldots, X_l^{(\ell_s)}\bigr)^{\mathsf{T}}$. Moreover, we have $(\hat{\boldsymbol{v}}_1^{\mathsf{T}}, \ldots, \hat{\boldsymbol{v}}_{\mathsf{M}_g^{\mathrm{c}}(f)}^{\mathsf{T}})^{\mathsf{T}} = \mathsf{I}_{\mathsf{M}_g^{\mathrm{c}}(f)}$, where $\mathsf{I}_{\mathsf{M}_g^{\mathrm{c}}(f)}$ is the $\mathsf{M}_g^{\mathrm{c}}(f) \times \mathsf{M}_g^{\mathrm{c}}(f)$ identity matrix and $\hat{\boldsymbol{v}}_i$ is the i-th row vector of the polynomial coefficient matrix $\hat{\mathsf{V}}_{\mu \times \mathsf{M}_g^{\mathrm{c}}(f)}$. With this mapping, one can show that for a desired polynomial indexed by $\upsilon = 1$, the types of τ-sums corresponding to undesired queries, i.e., τ-sums that do not involve any symbols from the desired function evaluation $\mathbf{U}^{(1)}$ can be divided into two groups as follows.

- Group 1: $\binom{\mu-1}{\tau} - \binom{\mu-\mathsf{M}_g^{\mathrm{c}}(f)}{\tau}$ τ-sum types for which the corresponding τ-sums involve at least one element from the set $\{\mathbf{U}^{(2)}, \mathbf{U}^{(3)} \ldots, \mathbf{U}^{(\mathsf{M}_g^{\mathrm{c}}(f))}\}$.
- Group 2: $\binom{\mu-\mathsf{M}_g^{\mathrm{c}}(f)}{\tau}$ τ-sum types for which the corresponding τ-sums do not involve any element from the set $\{\mathbf{U}^{(2)}, \mathbf{U}^{(3)}, \ldots, \mathbf{U}^{(\mathsf{M}_g^{\mathrm{c}}(f))}\}$,

such that the symbols of the queries corresponding to Group 2 are functions of the symbols of the queries corresponding to Group 1 when the symbols of the desired function evaluation are known. Thus, a number of $\binom{\mu-\mathsf{M}_g^{\mathrm{c}}(f)}{\tau}$ query types in Group 2 are redundant and can be removed from the query set. Accordingly, with the above mapping to an allied PLC problem, we have presented the main component needed to prove the second argument. Then, the result follows directly from [18, Lem. 4] and can be seen as a direct application of the proof of [13, Lem. 1, Sec. V-B] (see [18, App. C] for more details).

REFERENCES

[1] S. A. Obead, H.-Y. Lin, E. Rosnes, and J. Kliewer, “Private polynomial computation for noncolluding coded databases,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Paris, France, Jul. 2019, pp. 1677–1681.

[2] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, “Private information retrieval,” *J. ACM*, vol. 45, no. 6, pp. 965–982, Nov. 1998.

[3] W. Gasarch, “A survey on private information retrieval,” *Bull. Eur. Assoc. Theor. Comput. Sci. (EATCS)*, no. 82, pp. 72–107, Feb. 2004.

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

[5] N. B. Shah, K. V. Rashmi, and K. Ramchandran, “One extra bit of download ensures perfectly private information retrieval,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Honolulu, HI, USA, Jun./Jul. 2014, pp. 856–860.

[6] T. H. Chan, S.-W. Ho, and H. Yamamoto, “Private information retrieval for coded storage,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Hong Kong, Jun. 2015, pp. 2842–2846.

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

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

[9] R. Tajeddine, O. W. Gnilke, and S. El Rouayheb, “Private information retrieval from MDS coded data in distributed storage systems,” *IEEE Trans. Inf. Theory*, vol. 64, no. 11, pp. 7081–7093, Nov. 2018.

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

[11] R. Freij-Hollanti, O. W. Gnilke, C. Hollanti, A.-L. Horlemann-Trautmann, D. Karpuk, and I. Kubjas, “*t*-private information retrieval schemes using transitive codes,” *IEEE Trans. Inf. Theory*, vol. 65, no. 4, pp. 2107–2118, Apr. 2019.

[12] S. Kumar, H.-Y. Lin, E. Rosnes, and A. Graell i Amat, “Achieving maximum distance separable private information retrieval capacity with linear codes,” *IEEE Trans. Inf. Theory*, vol. 65, no. 7, pp. 4243–4273, Jul. 2019.

[13] H. Sun and S. A. Jafar, “The capacity of private computation,” *IEEE Trans. Inf. Theory*, vol. 65, no. 6, pp. 3880–3897, Jun. 2019.

[14] M. Mirmohseni and M. A. Maddah-Ali, “Private function retrieval,” in *Proc. Iran Workshop Commun. Inf. Theory (IWCIT)*, Tehran, Iran, Apr. 2018, pp. 1–6.

[15] Z. Chen, Z. Wang, and S. A. Jafar, “The asymptotic capacity of private search,” *IEEE Trans. Inf. Theory*, vol. 66, no. 8, pp. 4709–4721, Aug. 2020.

[16] D. Karpuk, “Private computation of systematically encoded data with colluding servers,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Vail, CO, USA, Jun. 2018, pp. 2112–2116.

[17] S. A. Obead and J. Kliewer, “Achievable rate of private function retrieval from MDS coded databases,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Vail, CO, USA, Jun. 2018, pp. 2117–2121.

[18] S. A. Obead, H.-Y. Lin, E. Rosnes, and J. Kliewer, “Private linear computation for noncolluding coded databases,” *IEEE J. Sel. Areas Commun.*, vol. 40, no. 3, pp. 847–861, Mar. 2022.

[19] S. A. Obead, H.-Y. Lin, E. Rosnes, and J. Kliewer, “On the capacity of private nonlinear computation for replicated databases,” in *Proc. IEEE Inf. Theory Workshop (ITW)*, Visby, Sweden, Aug. 2019, pp. 1–5.

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

[21] A. Heidarzadeh and A. Sprintson, “Private computation with side information: The single-server case,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Paris, France, Jul. 2019, pp. 1657–1661.

[22] A. Heidarzadeh and A. Sprintson, “Private computation with individual and joint privacy,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Los Angeles, CA, USA, Jun. 2020, pp. 1112–1117.

[23] B. Tahmasebi and M. A. Maddah-Ali, “Private sequential function computation,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Paris, France, Jul. 2019, pp. 1667–1671.

[24] B. Tahmasebi and M. A. Maddah-Ali, “Private function computation,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Los Angeles, CA, USA, Jun. 2020, pp. 1118–1123.

[25] Y. Yakimenka, H.-Y. Lin, and E. Rosnes, “On the capacity of private monomial computation,” in *Proc. Int. Zurich Seminar Inf. Commun. (IZS)*, Zürich, Switzerland, Feb. 2020, pp. 31–35.

[26] A. G. Dimakis, K. Ramchandran, Y. Wu, and C. Suh, “A survey on network codes for distributed storage,” *Proc. IEEE*, vol. 99, no. 3, pp. 476–489, Mar. 2011.

[27] Q. Yu, S. Li, N. Raviv, S. M. M. Kalan, M. Soltanolkotabi, and A. S. Avestimehr, “Lagrange coded computing: Optimal design for resiliency, security, and privacy,” in *Proc. 22nd Int. Conf. Artif. Intell. Statist. (AISTATS)*, Okinawa, Japan, Apr. 2019, pp. 1215–1225.

[28] R. G. L. D’Oliveira and S. El Rouayheb, “One-shot PIR: Refinement and lifting,” *IEEE Trans. Inf. Theory*, vol. 66, no. 4, pp. 2443–2455, Apr. 2020.

[29] H.-Y. Lin, S. Kumar, E. Rosnes, and A. Graell i Amat, “On the fundamental limit of private information retrieval for coded distributed storage,” Aug. 2018, *arXiv:1808.09018*.

[30] J. Xu and Z. Zhang, “On sub-packetization and access number of capacity-achieving PIR schemes for MDS coded non-colluding servers,” *Sci. China Inf. Sci.*, vol. 61, no. 10, Oct. 2018, Art. no. 100306.