Achievable Rate of Private Function Retrieval from MDS Coded Databases

Sarah A. Obead, Jörg Kliewer
Helen and John C. Hartmann Department of Electrical and Computer Engineering
New Jersey Institute of Technology
Newark, New Jersey 07102
Email: sao23@njit.edu, jkliewer@njit.edu

***Abstract*—We study the problem of private function retrieval (PFR) in a distributed storage system. In PFR the user wishes to retrieve a linear combination of M messages stored in non-colluding (N, K) MDS coded databases while revealing no information about the coefficients of the intended linear combination to any of the individual databases. We present an achievable scheme for MDS coded PFR with a rate that matches the capacity for coded private information retrieval derived recently, $R = (1 + R_c + R_c^2 + \cdots + R_c^{M-1})^{-1} = \frac{1-R_c}{1-R_c^M}$, where $R_c = \frac{K}{N}$ is the rate of the MDS code.**

I. Introduction

Private retrieval of information from public databases has received significant attention already for several decades from researchers in the computer science community (see, e.g., [1], [2]). While this line of work, commonly known as private information retrieval (PIR), is concerned with downloading individual messages in a private manner from databases, a recently proposed generalization of this problem [3], [4] addresses the private computation of functions of these messages. In accordance with [3] we denote this approach as private function retrieval (PFR) in the following. In PFR a user has access to a given number of databases and intends to compute a function of messages stored in these databases. This function is kept private from the databases, as they may be under the control of an eavesdropper. Both works [3], [4] characterize the fundamental information theoretic communication overhead needed to reliably compute the given function and specify the corresponding capacity and achievable rates as a function of the message size, the number of messages, and the number of databases, respectively. Further, the authors assume that the data is replicated on each database. Surprisingly, the obtained PFR capacity result is equal to the PIR capacity of [5].

However, although repetition coding adds the largest amount of redundancy and thus protects effectively against erasures, it is associated with a large storage cost. A more general way to optimally trade-off the available redundancy (or rate) versus the erasure correcting capability is given by MDS codes. In particular, for an (N, K) MDS code with N code symbols, K information symbols and rate $R_c = K/N$, $N-K$ erasures can be recovered from any K code symbols. Coded PIR has been addressed in two different lines of work. Achievable schemes for MDS coded PIR have been presented in [6]–[8] and the capacity has been established in [9]. On the other hand, in [10] linear codes with k different reconstruction sets for each code symbol have been proposed in form of so called k-server PIR.

This work is supported by NSF grant CNS-1526547.

In this paper we propose coded PFR, which to the best of our knowledge has not been addressed yet in the recent literature, with the notable exception of the parallel work [11]. In [11] a private computation scheme for polynomial functions of the data over T colluding databases is constructed by generalizing the star-product PIR scheme of [8]. Specifically, for systematically coded storage, the scheme proposed in [11] considers functions which are polynomials of fixed degree G, and achieves a rate matching the asymptotic rate of coded PIR $R = 1 - R_c$ for $G = T = 1$. In contrast, our scheme is constructed for MDS coded storage and considers only the computation of linear functions of the messages, but is able to achieve a strictly larger rate than [11] for any finite number of messages. In particular, we provide a characterization of an achievable rate of MDS coded PFR if the user wishes to compute an arbitrary linear combination of M independent equal-sized messages over some finite field $\mathbb{F}_q$, distributed over N non-colluding MDS-coded databases. Surprisingly, our achievable rate matches the capacity for MDS coded PIR in [9]. This demonstrates that, compared to the naïve scheme, where M coded messages are downloaded and linearly combined offline at the user (requiring M-times the coded PIR rate), downloading the result of the computation privately and directly from the databases does not incur any penalty in rate compared to the coded PIR case. Thus, our result strictly generalizes the achievable schemes in [3], [4] which represent special cases of our proposed PFR scheme.

II. Problem Statement

In the following, we use $[1 : X]$ to denote the set $\{1, \ldots, X\}$. Similarly, $X_{1:N} = \{X_1, \ldots, X_N\}$.

A. System Model

In coded PFR, a user wishes to privately retrieve a linear combination of the messages stored in the databases such that the coefficients of the linear combination are kept secret from each individual database. Consider a linear distributed storage system storing M equal-sized messages on N non-colluding databases. The message $W_m, m \in [1 : M]$, is composed from L symbols chosen independently and uniformly at random from the finite field $\mathbb{F}_q$ with

$$H(W_1) = \cdots = H(W_M) = L \log q, \tag{1}$$

$$H(W_1,\ldots, W_M) = H(W_1) + \ldots + H(W_M) = ML \log q. \tag{2}$$

Each message is divided into $\tilde{L}$ segments, each of K symbols, forming a $\tilde{L} \times K$ matrix, where $L = \tilde{L}K$. The messages are stored using an (N, K) MDS code with the generator matrix defined by

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_1 & \mathbf{g}_2 & \dots & \mathbf{g}_N \end{bmatrix}_{K \times N}, \tag{3}$$

with $\mathbf{g}_n,\ n \in [1 : N]$, denoting the n-th column vector of $\mathbf{G}$. The generator matrix produces a code that can tolerate up to $N - K$ erasures by retrieving data from any set $\mathcal{K} \subset [1 : N]$ databases, where $|\mathcal{K}| \geq K$. The encoding process for message W_m is defined as follows:

$$\begin{aligned}\begin{bmatrix}\mathbf{w}_{m,t}\end{bmatrix}_{1\times K} & \begin{bmatrix} \mathbf{g}_1 & \mathbf{g}_2 & \dots & \mathbf{g}_N \end{bmatrix}_{K \times N} \\ &= \begin{bmatrix} \mathbf{g}_1^T \mathbf{w}_{m,t}^T & \dots & \mathbf{g}_N^T \mathbf{w}_{m,t}^T \end{bmatrix}_{1\times N},\end{aligned} \tag{4}$$

where $\mathbf{w}_{m,t},\ \forall m \in [1 : M], \forall t \in [1 : \tilde{L}]$, denotes the K-dimensional vector of symbols of the t-th segment from the message W_m. The resulting N coded symbols for each segment are then distributed over the N databases, and the code rate is given by $R_c = \frac{K}{N}$.

Consequently, the code symbols stored at each database are represented by an $M \times \tilde{L}$ matrix $\mathbf{W}_n\ \forall n \in [1 : N]$, as follows:

$$\begin{aligned}\mathbf{W}_n &= \begin{bmatrix} \mathbf{g}_n^T \mathbf{w}_{1,1}^T & \mathbf{g}_n^T \mathbf{w}_{1,2}^T & \cdots & \mathbf{g}_n^T \mathbf{w}_{1,\tilde{L}}^T \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{g}_n^T \mathbf{w}_{M,1}^T & \mathbf{g}_n^T \mathbf{w}_{M,2}^T & \cdots & \mathbf{g}_n^T \mathbf{w}_{M,\tilde{L}}^T \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{W}_n[1] & \mathbf{W}_n[2] & \dots & \mathbf{W}_n[\tilde{L}] \end{bmatrix}\end{aligned} \tag{5}$$

where we use $\mathbf{W}_n[t],\ t \in [1 : \tilde{L}]$, to denote the t-th column.

In PFR, the linear combination κ the user intends to retrieve is represented as

$$\widetilde{W}_\kappa = \mathbf{v}_\kappa [W_1, \dots, W_M]^T \tag{6}$$

$$= v_\kappa(1) W_1 + \dots + v_\kappa(M) W_M \tag{7}$$

$$= \begin{bmatrix} \mathbf{v}_\kappa \mathbf{W}[1] & \dots & \mathbf{v}_\kappa \mathbf{W}[L] \end{bmatrix}, \tag{8}$$

where $\mathbf{W}[\ell],\ \forall \ell \in [1 : L]$, denotes a column of the ℓ-th symbols of the original messages, $\mathbf{v}_\kappa$ is an M-dimensional non-zero coefficient vector of the linear combination (row vector) indexed by κ, the coefficients $v_\kappa(m),\ \forall m \in [1 : M]$, are chosen from the finite field $\mathbb{F}_q$, and the addition "$+$" is done element-wise over the same field. We assume that the vector $\mathbf{v}_\kappa$ is an element of the set $\mathcal{V}$ that contains all possible distinct M-dimensional vectors defined over $\mathbb{F}_q$ where $\kappa \in [1 : V],\ V = |\mathcal{V}| = \frac{q^M - 1}{q - 1}$.

In order for the user to retrieve the linear combination $\widetilde{W}_\kappa$, while keeping κ secret from each database, it generates N query matrices for the databases $\{Q_1^{[\kappa]}, \dots, Q_N^{[\kappa]}\}$. Since the query matrices are generated by the user without prior knowledge of the realizations of the stored messages, the queries must be independent of the messages,

$$I(Q_1^{[\kappa]}, \dots, Q_N^{[\kappa]}; W_1, \dots, W_M) = 0, \quad \forall \kappa \in [1 : V]. \tag{9}$$

Upon the reception of the query $Q_n^{[\kappa]}$, the n-th database generates an answer string $A_n^{[\kappa]}$ as a deterministic function of the received query and the stored symbols from each message. Hence,

$$H(A_n^{[\kappa]} | Q_n^{[\kappa]}, \mathbf{W}_n) = 0, \quad \forall \kappa \in [1 : V], \forall n \in [1 : N]. \tag{10}$$

To maintain user privacy, the query-answer function must be identically distributed for each possible linear combination $\kappa \in [1 : V]$ from the perspective of each database $n \in [1 : N]$. In other words, the scheme's queries and answers strings must be independent from the desired linear combination index, therefore the following privacy constraint must be satisfied:

$$I(A_n^{[\kappa]}, Q_n^{[\kappa]}, \mathbf{W}_n; \kappa) = 0, \quad \forall \kappa \in [1 : V]. \tag{11}$$

After the user receives all answer strings from each database, the user must be able to reliably decode the desired linear combination message $\widetilde{W}_\kappa$ with a probability of error P_e that goes to zero as the message size L approaches infinity. Following Fano's inequality, this translates to the correctness constraint

$$H(\widetilde{W}_\kappa | A_{1:N}^{[\kappa]}, Q_{1:N}^{[\kappa]}) = o(L), \tag{12}$$

where $o(L)$ represents any function of L, $f(L)$, that satisfies $\lim_{L\to\infty} f(L)/L \to 0$.

The retrieval rate of the coded PFR scheme is characterized by the message length L, the query structure Q, and the query-answer function, and is defined as the ratio between the size of the desired linear combination message and the total number of downloaded symbols in bits as

$$R = \frac{H(\widetilde{W}_\kappa)}{\sum_{n=1}^N H(A_n^{[\kappa]})}. \tag{13}$$

A rate R is said to be achievable if there exist a sequence of coded PFR schemes that satisfy the privacy and correctness constraints of (11), (12) for $P_e \to 0$ as $L \to \infty$.

III. Achievable Rate of MDS Coded PFR

Theorem 1. *For an (N, K) coded distributed storage system with code rate $R_c = \frac{K}{N}$, M messages and a set of V linear combinations defined over the field $\mathbb{F}_q$, a PFR scheme can be constructed which achieves a rate*

$$R = \frac{1 - R_c}{1 - R_c^M} \tag{14}$$

$$= \left(1 + \frac{K}{N} + \frac{K^2}{N^2} + \dots + \frac{K^{M-1}}{N^{M-1}}\right)^{-1}. \tag{15}$$

Remark 1. This achievable rate generalizes the achievable rate of repetition coded PFR [4] which corresponds to the special case of $K = 1$. Also, (14) is only a function of the distributed storage code rate R_c and the number of stored independent messages M, and is universal in the sense that it does not depend on the number of linear combinations V defined over the finite field $\mathbb{F}_q$ nor on the explicit structure of the code.

Remark 2. If we consider each of the V linear combinations of messages in (6) as a new *virtual message* $\widetilde{W}_\kappa$, and then apply the coded PIR scheme of [9], the rate of the scheme is $\frac{1-R_c}{1-R_c^V}$ which is smaller than (14) since $M \leq V$.

Remark 3. If the set of possible linear combination $\mathcal{V}$ is reduced to *only* include linear combinations with the property $\mathbf{v}_{1:M} \in \mathcal{V} : [\mathbf{v}_1^T\ \mathbf{v}_2^T\ \dots\ \mathbf{v}_M^T] = \mathbf{I}_M$ for $V = M$, the achievable rate of (14) is tight. That is because in this setting the problem of coded PFR is reduced to coded PIR where the converse is implied from [9]. Also, we note that (14) is equivalent to the coded PIR capacity [9], which has been observed in [4] for $K = 1$. Thus, downloading linear combinations of messages does not incur additional costs over downloading individual messages.

Remark 4. Eq. (14) is a strictly decreasing function in the number of messages M for fixed R_c. As the number of messages increases $M \to \infty$, the achievable rate approaches $1 - R_c$. Moreover, as $R_c \to 1$ in (15), $R \to \frac{1}{M}$, indicating that to maintain the privacy of the desired linear combination, the user has to download all the messages and perform the computation off-line.

IV. Proof of Theorem 1

A. Query generation

The generation of the queries is shown in Algorithm 1. Let $B \in [1 : V]$ be the block indicator and $R \in [1 : K]$ be the repetition indicator, respectively. Let the v-sum be the combination of v distinct elements out of V elements. Since we have $\binom{V}{v}$ different combinations, we denote each different combination as a *type* of the v-sum. Let the components of these combinations be symbols of the V virtual messages. As mentioned above, we generate the query set for each database in blocks, where a block represents a group of all $\binom{V}{v}$ types of v-sums for all $v \in [1 : V]$, resulting in V blocks in total. To this end, we let the size of the *dependent* virtual messages to be $L = KN^V$ (i.e., $\tilde{L} = N^V$).

For a desired linear combination $\kappa \in [1 : V]$ we use the notation $Q^{[\kappa]}(DB)$ to indicate the query set of the database $DB \in [1 : N]$. This set is composed from VK disjoint subsets $Q^{[\kappa]}_{B,R}(DB)$ generated for each block B and repetition R. We require $K^{V-B}(N-K)^{B-1}$ distinct instances of each type of v-sum for every set $Q^{[\kappa]}_{B,R}(DB)$. Each block and repetition subset is further subdivided into two subsets: the first subset $Q^{[\kappa]}_{B,R}(DB, \mathcal{M})$ consists of the v-sum types with symbols from the desired linear combination, and the second subset $Q^{[\kappa]}_{B,R}(DB, \mathcal{I})$ contains only v-sum types with symbols from undesired linear combinations. The query sets for all databases are generated by Algorithm 1 with the following procedures.

1) Index assignment: In the MDS-coded PIR scheme [9], the user privately applies a random permutation over the coded symbols of each message independently. The goal is to make the coded symbols queried from each database to appear to be chosen randomly and independently from the desired message. However, for the PFR problem the linear function is computed element-wise, thus there is a dependency across the symbols with the same index, which must be maintained under a permutation. To this end, we modify the permutation to be fixed across all messages. Let $\pi(\cdot)$ be a random permutation function over $[1 : \tilde{L}]$. We use the notation $U^{[n]}_\kappa(t)$, where

$$U^{[n]}_\kappa(t) \triangleq \sigma_t \mathbf{v}_\kappa \mathbf{W}_n[\pi(t)] \ \forall n \in [1 : N]^1 \qquad (16)$$

denotes the permuted code symbol associated with the virtual message $\widetilde{W}_\kappa$. The random variable σ is used to indicate the sign assigned to each individual coded symbol, $\sigma_t \in \{+1, -1\}$ [4]. Both σ_t and π are randomly selected privately and uniformly by the user.

[1]For brevity we drop the superscript from $U^{[n]}_\kappa(t)$ whenever this is clear from the context.

2) Block $B = 1$: This this block is described by Steps 3 to 10 of Algorithm 1, where we have $v = 1$ for the v-sum.

Initialization: In the initialization step, the user queries the first database $DB = 1$ for K^{V-1} distinct code symbols from the desired linear combination $U_\kappa(i)$. This is done by calling the function "new(U_κ)" that will select a symbol from message U_κ with a new index i each time it is called (Step 6).

Database symmetry: Database symmetry is obtained via the "For" loop in Step 3, resulting in a total number of NK^{V-1} symbols over all databases.

Message symmetry: In Step 7, to maintain message symmetry, the user asks each database for the same number of distinct symbols of all other linear combinations $U_\theta(i)$, $\theta \in \{1, \ldots, V\} \setminus \{\kappa\}$, resulting in a total number of NVK^{V-1} symbols. As a result, the query sets for each database are symmetric with respect to all linear combination vectors in $[1 : V]$. We associate the symbols of undesired messages in K groups $G \in [1 : K]$ to be exploited as distinct side information for different rounds of the scheme as shown in Step 7.

3) Side-information exploitation: In Steps 11 to 20, we generate the blocks $B \in [2 : V]$ by applying two subroutines "Exploit-SI" and "M-Sym", respectively. We first reuse the subroutine "Exploit-SI" [4] to generate queries for new symbols of the desired linear combination U_κ by combining these symbols with different side information groups from the previous block associated with $N - K$ neighboring databases as shown in Step 13. This is required by our proposed MDS coded scheme to ensure privacy and is in contrast to [4], where the side information of previous blocks from *all* databases is utilized.

Then, the subroutine "M-Sym" [4] is used to generate side information to be exploited in the following blocks. This subroutine select symbols of undesired messages to generate v-sums that enforce symmetry in the block queries. For example in $B = 2$, if we have the queries $U_\kappa(i) + U_2(j)$, and $U_\kappa(l) + U_3(r) \in Q^{[\kappa]}_{2,R}(DB, \mathcal{M})$, this subroutine will generate $U_2(l) + U_3(i)$. As a result, we can show that the symmetry over the linear combinations and databases is maintained. By the end of this step we have in total $N\binom{V}{B}K^{V-B}(N-K)^{B-1}$ queries for each block from all databases.

4) Generation of further query rounds: We require further query rounds to obtain K linear equations for each coded symbol to be able to decode. To this end, we circularly shift the order of the database at each repetition. The shift is done for the initial block, $B = 1$, in Steps 22 to 25. However, for the following blocks we only rotate the indices of desired messages U_κ and combine them with new groups of side information from the neighboring databases from the first round as seen in Steps 26 to 33. This rotation and side information exploitation for $B \in [2 : V]$ is done using the subroutine "Reuse-SI" (omitted in the interest of space).

5) Query set assembly: Finally, in Steps 35 to 37, we assemble each query set from the queries disjoint subsets obtained in the previous blocks and rounds.

Remark 5. Note that the proposed scheme significantly differs from the one presented in [4] in terms of how the side information is exploited due to coding. In particular, we distribute the side information over K rounds such that every database is queried for each message and linear combination only once.

Algorithm 1: Query set generation algorithm

```
Input: $\kappa, K, N, M$, and $V$.
Output: $Q^{[\kappa]}(1), \dots, Q^{[\kappa]}(N)$
1. Initialize: All query sets are initialized as a null set $Q^{[\kappa]}(1), \dots, Q^{[\kappa]}(N) \leftarrow \emptyset$, the block counter $B = 1$, and repetition counter $R = 1$. Let number of neighboring databases $Nb = N - K$
2. Let repetition $R_B = K^{V-B}(N-K)^{B-1} \quad \forall B \in [1:V]$
3. For first database block $DB = 1:N$ do
4.   For side information group $G = 1:K$ do
5.     For repetition group $RG = 1:(R_1/K)$ do
6.         $Q^{[\kappa]}_{1,R}(DB, \mathcal{M}) \leftarrow \{u_\kappa\}, u_\kappa = \text{new}(U_\kappa)$
7.         $Q^{[\kappa]}_{1,R}(DB, \mathcal{I}_G) \leftarrow \{\text{new}(U_1), \dots, \text{new}(U_V)\} \backslash \{u_\kappa\}$
8.     End For (repeat within the same SI group)
9.   End For (repetition for SI groups)
10. End For
11. For block $B = 2:V$ do
12.   For $DB = 1:N$ do
13.       $Q^{[\kappa]}_{B,R}(DB, \mathcal{M}) \leftarrow$ Exploit-SI$\big(Q^{[\kappa]}_{B-1,R}(DB+1, \mathcal{I}_{Nb}) \cup \dots \cup Q^{[\kappa]}_{B-1,R}(DB+Nb, \mathcal{I}_1)\big)$
14.     For side-information group $G = 1:K$ do
15.       For $RG = 1:(R_B/K)$ do
16.           $Q^{[\kappa]}_{B,R}(DB, \mathcal{I}_G) \leftarrow$ M-Sym$(Q^{[\kappa]}_{B,R}(DB, \mathcal{M}))$
17.       End For (repeat within the same SI group)
18.     End For (repeat for SI groups)
19.   End For (repeat for each database)
20. End for (repeat for each block)
21. For query round $R = 2:K$ do
22.   For $DB = 1:N$ do
23.       $Q^{[\kappa]}_{1,R}(DB, \mathcal{M}) \leftarrow Q^{[\kappa]}_{1,R-1}(DB-1, \mathcal{M})$
24.       $Q^{[\kappa]}_{1,R}(DB, \mathcal{I}_G) \leftarrow Q^{[\kappa]}_{1,R-1}(DB-1, \mathcal{I}_G)$
25.   End For (initializing rounds)
26.   For block $B = 2:V$ do
27.     For $DB = 1:N$ do
28.       For side information group $G = 1:K$ do
29.           $Q^{[\kappa]}_{B,R}(DB, \mathcal{I}_G) \leftarrow Q^{[\kappa]}_{B,R-1}(DB-1, \mathcal{I}_G)$
30.           $Q^{[\kappa]}_{B,R}(DB, \mathcal{M}) \leftarrow$ Reuse-SI$\big(Q^{[\kappa]}_{B,R}(DB, \mathcal{I}_G), Q^{[\kappa]}_{B-1,1}(DB+1, \mathcal{I}_{Nb+R-1}) \cup \dots \cup Q^{[\kappa]}_{B-1,1}(DB+Nb, \mathcal{I}_R)\big)$
31.       End For (SI groups)
32.     End For (repeating for each database)
33.   End For (repeating for each block)
34. End For (repeating for each round)
35. For $DB = 1:N$ do
36.     $Q^{[\kappa]}(DB) \leftarrow \bigcup_{B=1}^{V} \bigcup_{R=1}^{K} \big(Q^{[\kappa]}_{B,R}(DB, \mathcal{I}) \cup Q^{[\kappa]}_{B,R}(DB, \mathcal{M})\big)$
37. End For (assembling the query sets)
```

B. Sign assignment and redundancy elimination

We carefully assign an alternating sign $\sigma_t \in [+1, -1]$ to each symbol in the query set, based on the desired linear combination index κ [4]. The intuition behind the sign assignment is to introduce a uniquely solvable linear equation system from the different v-sum types. By obtaining such an equation system in each block, the user can opt from downloading these queries, compute them off-line, and thus reduce the download rate. Based on this insight we can state the following lemma.

Lemma 1 ([4])**.** *For all $\kappa \in [1:V]$, each database $n \in [1:N]$, and based on the side information available from the neighboring databases, there are $\binom{V-M}{v}$ redundant v-sum types out of all possible types $\binom{V}{v}$ in each block $v \in [1:V-M]$ of the query sets.*

Lemma 1 is also applicable when the desired linear function is performed over MDS-coded databases due to the fact that each MDS-coded symbol is itself a linear combination. That is, the MDS code can be seen as an inner code and the desired linear function as an outer "code" with respect to the databases. Hence, the redundancy resulting from the linear dependencies between messages is also present under MDS coding as the messages in Lemma 1 in our case consist of MDS code symbols. Thus, we can extend Lemma 1 to our scheme. We now make the final modification to our PFR query sets. We first directly apply the sign assignment σ_t, then remove the redundant v-sum types from every block $B \in [1:V]$. Finally, we generate the query matrices $Q^{[\kappa]}_{1:N}$ using a one-to-one mapping function f, for which $Q^{[\kappa]}(DB)$ is the preimage.

Proof. The proof of optimality for arbitrary N, K, M, and V follows from the structure of the query and Lemma 1. The achievable rate is given as

$$
\begin{aligned}
R &\overset{(a)}{=} \frac{KN^V}{KN\sum_{v=1}^{V}\left(\binom{V}{v} - \binom{V-M}{v}\right)K^{V-v}(N-K)^{v-1}} \\
&= \frac{N^V\left(\frac{N-K}{N}\right)}{\sum_{v=1}^{V}\left(\binom{V}{v}K^{V-v}(N-K)^{v} - \binom{V-M}{v}K^{V-v}(N-K)^{v}\right)} \\
&\overset{(b)}{=} \frac{N^V\left(\frac{N-K}{N}\right)}{(N^V - K^V) - \sum_{v=1}^{V-M}\binom{V-M}{v}K^{V-v}(N-K)^{v}} \\
&= \frac{N^V\left(\frac{N-K}{N}\right)}{(N^V - K^V) - K^M\sum_{v=1}^{V-M}\binom{V-M}{v}K^{V-M-v}(N-K)^{v}} \\
&= \frac{N^V\left(1 - \frac{K}{N}\right)}{(N^V - K^V) - K^M\left(N^{V-M} - K^{V-M}\right)} \\
&= \frac{N^V\left(1 - \frac{K}{N}\right)}{(N^V - K^V) - K^M N^{V-M} + K^V} \\
&= \frac{N^V\left(1 - \frac{K}{N}\right)}{N^V - K^M N^{V-M}} = \frac{1 - R_c}{1 - R_c^M};
\end{aligned}
$$

where (a) follows from the definition of the PFR rate (13); (b) follows from the fact that the second term of the summation in the denominator is equal to zero for $v > V-M$, consequently we can change the upper bound of the summation; and the first term of the summation follows from the binomial theorem. ∎

C. Correctness (decodability)

To prove correctness, we show that the user can obtain the desired linear combination $\widetilde{W}_\kappa$ from the answers retrieved from N databases. From the query answers $A^{[\kappa]}_{1:N}$, we group the K identical queries from different rounds and databases. Each group will result in K linearly independent equations that can be uniquely solved. We decode, block by block, starting from block one, which we directly decode and obtain $KN\big(\binom{V}{v}-\binom{V-M}{v}\big)K^{V-v}(N-K)^{v-1}$ decoded symbols. Now, using these symbols we regenerate $\binom{V-M}{v}$ redundant symbols according to Lemma 1 and obtain $KN\binom{V}{v}K^{V-v}(N-K)^{v-1}$ symbols in total. Out of these queries there are $KN\big(\binom{V}{v}-\binom{V-1}{v}\big)K^{V-v}(N-K)^{v-1}$ symbols from $\widetilde{W}_\kappa$.

Next, for blocks $B \in [2:V]$ we use the symbols obtained in the previous block $B-1$ to remove the side information associated with the desired linear combination symbols of the current block B. Then, the operations from the first block (decode and retrieve redundancy) are repeated. As a result, we obtain a total number of symbols equal to $KN\sum_{v=1}^{V}\big(\binom{V}{v}-\binom{V-1}{v}\big)K^{V-v}(N-K)^{v-1} = \frac{N}{N-K}\big(N^V - KN^{V-1}\big) = KN^V$ denoting precisely the number of symbols in $\widetilde{W}_\kappa$.

D. Privacy

Privacy is guaranteed by preserving an equal number of requests for any linear combination $\widetilde{W}_\kappa$, where the requests are symmetric from the perspective of the accessed virtual messages. As the MDS code can be seen as an outer code, the privacy arguments in [3], [4] apply here as well. In particular, each database is queried with precisely the same v-sum type components regardless of the desired linear combination index. This symmetry is ensured over all v-sum types as can be seen from Steps 13 and 16 in Algorithm 1 where the same subroutines "M-Sym" and "Exploit-SI" are used for each block and database. Moreover, since each of the queries $\{Q^{[\kappa]}(1),\ldots,Q^{[\kappa]}(N)\}$ generated by Algorithm 1 contains a set of distinct symbols from the desired linear combination, we let the user select a permutation $\pi(t)$ and a sign assignment σ_t uniformly at random acting as a one-time pad for each set of quires. With other words, for any $U_\kappa^{[n]}(t) = \sigma_t \mathbf{v}_\kappa \mathbf{W}_n[\pi(t)]$ there exist $\sigma_t, \pi(t)$ such that $Q^{[\theta]}(DB) \leftrightarrow Q^{[\kappa]}(DB)\ \forall \kappa, \theta \in [1:V]$. Thus, $A_n^{[\kappa]}$ and $Q_n^{[\kappa]}$ are statistically independent of κ and (11) holds.

V. Example

We consider $M=2$ messages stored using a $(3,2)$ MDS code. The user wishes to obtain a linear combination over the binary field (i.e., $\mathbf{v}_\kappa \in \mathbb{F}_2^M$, $V=3$). Therefore, we have the linear combinations $\mathbf{v}_1 = [1\ 0], \mathbf{v}_2 = [0\ 1], \mathbf{v}_3 = [1\ 1]$, and each message must be of length $L = KN^V = 54$ symbols. We simplify the notation by letting $a_t^{[n]} = U_1^{[n]}(t)$, $b_t^{[n]} = U_2^{[n]}(t)$, and $c_t^{[n]} = U_3^{[n]}(t)$ for all $t \in [1:27]$, $n \in [1:N]$. Let the desired linear combination index be $\kappa = 3$.

Query set construction: Algorithm 1 starts with $B=1$ by generating queries for each database and $K^{V-1} = 4$ distinct instances of $c_t^{[n]}$ (i.e., from database 1 query $c_{1:4}^{[1]} \triangleq \{c_1^{[1]},\ldots,c_4^{[1]}\}$). By message symmetry this also applies for $a_t^{[n]}$ and $b_t^{[n]}$ which form two groups of side information sets to be used in the next block with $NK^{V-1} = 12$ symbols in total from each linear combination. Next, one group of side information is queried jointly with a new instant of the desired message. For example, for database 1 and type $b+c$ we have $b_{5:6}^{[1]} - c_{13:14}^{[1]} \triangleq \{b_5^{[1]} - c_{13}^{[1]}, b_6^{[1]} - c_{14}^{[1]}\}$. The remaining blocks and rounds follow from Algorithm 1. After generating the query set for each database, we apply the sign assignment and remove the redundant queries. The answer strings from each query are shown in Table 1. Note that there is no $c_t^{[n]}$ in the first block as they are redundant and can be generated by the user.

TABLE I
THE QUERY RESPONSE FOR PFR FROM $(3,2)$ MDS CODED DATABASES, $M=2$, $V=3$, AND $\kappa=3$

(R,B)	DB1	DB2	DB3
$(1,1)$	$a_{1:4}^{[1]},\ b_{1:4}^{[1]}$	$a_{5:8}^{[2]},\ b_{5:8}^{[2]}$	$a_{9:12}^{[3]},\ b_{9:12}^{[3]}$
$(1,2)$	$b_{5:6}^{[1]} - c_{13:14}^{[1]}$ $a_{5:6}^{[1]} - c_{15:16}^{[1]}$	$b_{9:10}^{[2]} - c_{17:18}^{[2]}$ $a_{9:10}^{[2]} - c_{19:20}^{[2]}$	$b_{1:2}^{[3]} - c_{21:22}^{[3]}$ $a_{1:2}^{[3]} - c_{23:24}^{[3]}$
	$a_{13:14}^{[1]} - b_{15:16}^{[1]}$	$a_{17:18}^{[2]} - b_{19:20}^{[2]}$	$a_{21:22}^{[3]} - b_{23:24}^{[3]}$
$(1,3)$	$a_{17}^{[1]} - b_{19}^{[1]} + c_{25}^{[1]}$	$a_{21}^{[2]} - b_{23}^{[2]} + c_{26}^{[2]}$	$a_{13}^{[3]} - b_{15}^{[3]} + c_{27}^{[3]}$
$(2,1)$	$a_{9:12}^{[1]},\ b_{9:12}^{[1]}$	$a_{1:4}^{[2]},\ b_{1:4}^{[2]}$	$a_{5:8}^{[3]},\ b_{5:8}^{[3]}$
$(2,2)$	$b_{7:8}^{[1]} - c_{21:22}^{[1]}$ $a_{7:8}^{[1]} - c_{23:24}^{[1]}$	$b_{11:12}^{[2]} - c_{13:14}^{[2]}$ $a_{11:12}^{[2]} - c_{15:16}^{[2]}$	$b_{3:4}^{[3]} - c_{17:18}^{[3]}$ $a_{3:4}^{[3]} - c_{19:20}^{[3]}$
	$a_{21:22}^{[1]} - b_{23:24}^{[1]}$	$a_{13:14}^{[2]} - b_{15:16}^{[2]})$	$a_{17:18}^{[3]} - b_{19:20}^{[3]}$
$(2,3)$	$a_{18}^{[1]} - b_{20}^{[1]} + c_{27}^{[1]}$	$a_{22}^{[2]} - b_{24}^{[2]} + c_{25}^{[2]}$	$a_{14}^{[3]} - b_{16}^{[3]} + c_{26}^{[3]}$

Decoding: To decode we start with Block 1, and we obtain $a_1^{[1]}$ from database 1 and $a_1^{[2]}$ from database 2. Thus, by the MDS code properties we can decode and obtain the segment associated with a_1; similarly for all other queries in this block. Now for Block 2, we first remove the side information from the types containing symbols of $c_t^{[n]}$. For example, for $b_5^{[1]} - c_{13}^{[1]}$ from database 1, we have the segment associated with b_5 from the previous block. As a result, we obtain $c_{13}^{[1]}$. Similarly we obtain $c_{13}^{[2]}$ from database 2, and the segment associated with c_{13} can be recovered. Finally, after repeating this process for the rest of the coded segments of c_t we obtain a PFR rate of $R = 0.6$.

References

[1] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan, "Private information retrieval," *J. ACM*, vol. 45, no. 6, pp. 965–981, Nov. 1998.

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

[3] M. Mirmohseni and M. A. Maddah-Ali, "Private function retrieval," Nov. 2017. [Online]. Available: https://arxiv.org/abs/1711.04677

[4] H. Sun and S. A. Jafar, "The capacity of private computation," Nov. 2017. [Online]. Available: https://arxiv.org/abs/1710.11098

[5] ——, "The capacity of private information retrieval," *IEEE Trans. Inf. Theory*, vol. 63, no. 7, pp. 4075–4088, July 2017.

[6] T. H. Chan, S.-W. Ho, and H. Yamamoto, "Private information retrieval for coded storage," in *Proc. IEEE Int. Sympos. on Inf. Theory*. IEEE, June 2015, pp. 2842–2846.

[7] R. Tajeddine and S. E. Rouayheb, "Private information retrieval from MDS coded data in distributed storage systems," in *Proc. IEEE Int. Sympos. on Inf. Theory*, July 2016, pp. 1411–1415.

[8] 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, 2017.

[9] K. Banawan and S. Ulukus, "Private information retrieval from coded databases," in *Proc. IEEE Int. Conf. on Commun.*, May 2017, pp. 1–6.

[10] A. Fazeli, A. Vardy, and E. Yaakobi, "Codes for distributed PIR with low storage overhead," in *Proc. IEEE Int. Sympos. on Inf. Theory*, June 2015, pp. 2852–2856.

[11] D. Karpuk, "Private computation of systematically encoded data with colluding servers," Jan. 2018. [Online]. Available: https://arxiv.org/abs/1801.02194