Optimal Download Cost of Private Information Retrieval for Arbitrary Message Length

Hua Sun, *Student Member, IEEE*, and Syed Ali Jafar, *Fellow, IEEE*

***Abstract*— A private information retrieval (PIR) scheme is a mechanism that allows a user to retrieve any one out of K messages from N non-communicating replicated databases, each of which stores all K messages, without revealing anything (in the information theoretic sense) about the identity of the desired message index to any individual database. If the size of each message is L bits and the total download required by a PIR scheme from all N databases is D bits, then D is called the download cost and the ratio L/D is called an achievable rate. For fixed $K, N \in \mathbb{N}$, the capacity of PIR, denoted by C, is the supremum of achievable rates over all PIR schemes and over all message sizes, and was recently shown to be $C = (1+1/N+1/N^2+\cdots+1/N^{K-1})^{-1}$. In this paper, for arbitrary K and N, we explore the minimum download cost D_L across all PIR schemes (not restricted to linear schemes) for arbitrary message lengths L under arbitrary choices of alphabet (not restricted to finite fields) for the message and download symbols. If the same M-ary alphabet is used for the message and download symbols, then we show that the optimal download cost in M-ary symbols is $D_L = \lceil \frac{L}{C} \rceil$. If the message symbols are in M-ary alphabet and the downloaded symbols are in M'-ary alphabet, then we show that the optimal download cost in M'-ary symbols, $D_L \in \{\lceil (L'/C) \rceil, \lceil (L'/C \rceil - 1, \lceil (L'/C) \rceil - 2\}$, where $L' = \lceil L \log_{M'} M \rceil$, i.e., the optimal download cost is characterized to within two symbols.**

***Index Terms*— Private information retrieval, download cost, capacity, finite message length.**

I. Introduction

In THE private information retrieval (PIR) problem [1], [2], we have K messages, stored at N distributed and non-communicating databases. A PIR scheme allows a user to retrieve any one of the K messages, while revealing no information to any individual database (even if the database has unbounded computation power) about the retrieved message index. Typical quality measures of PIR schemes include communication complexity [1]–[7], computational overhead [8]–[10], storage overhead [11]–[19], upload cost, download cost [11], [12], [14], [18], [20], and rate [14], [18], [20]. In this work we will focus on download cost and rate. If the size of each message is L bits and the total download required by a PIR scheme from all N databases is D bits, then D is called the download cost and the ratio L/D is called an achievable rate. The capacity of PIR, denoted by C, is defined to be the supremum of achievable rates over all PIR schemes and over all message sizes. It was shown recently in [20] that[1]

$$C = \left(1 + \frac{1}{N} + \frac{1}{N^2} + \cdots + \frac{1}{N^{K-1}}\right)^{-1} \tag{1}$$

The reciprocal of capacity, $1/C$, similarly represents the infimum of download cost per message bit over all PIR schemes and over all message sizes. Fundamental information theoretic measures such as these are essentially asymptotic in character, involving limits as message lengths $L \to \infty$. Remarkably, [20] shows that these asymptotically optimal values are also achieved exactly when the message length parameter L is any integer multiple of N^K. However, since in practice the message length parameter L can be arbitrary, an important question that remains open is to determine optimal download cost and rate values for arbitrary fixed values of L, in particular when L is not an integer multiple of N^K. In this work, we explore the minimum download cost across all PIR schemes (not restricted to linear schemes) for arbitrary message lengths under arbitrary choices of alphabet (not restricted to finite fields) for the message and download symbols. If the same M-ary alphabet is used for the message and download symbols, then we show that the optimal download cost in M-ary symbols is $D_L = \lceil \frac{L}{C} \rceil$. If the message symbols are in M-ary alphabet and the downloaded symbols are in M'-ary alphabet, then we show that the optimal download cost in M'-ary symbols, $D_L \in \left\{\left\lceil \frac{L'}{C} \right\rceil, \left\lceil \frac{L'}{C} \right\rceil - 1, \left\lceil \frac{L'}{C} \right\rceil - 2\right\}$, where $L' = \lceil L \log_{M'} M \rceil$. Correspondingly, the maximum achievable rate is automatically characterized in every case as L/D_L.

Notation: $\mathbb{N}$ *is the set of natural numbers. For integers* $Z_1, Z_2, Z_1 \leq Z_2$, *we use the compact notation* $[Z_1 : Z_2] = \{Z_1, Z_1 + 1, \cdots, Z_2\}$. *Similarly,* $A_{[Z_1:Z_2]} \triangleq \{A_{Z_1}, A_{Z_1+1}, \cdots, A_{Z_2}\}$ *for any variable* A. *The notation* $X \sim Y$ *is used to indicate that* X *and* Y *are identically distributed. The notation* $|A|$ *is used to denote the cardinality of a set when* A *is a set, and the length of a tuple when* A *is a tuple. For sets* S_1, S_2, *we define* S_1/S_2 *as the set of elements that are in* S_1 *and not in* S_2. *For a permutation function* $\lambda(\cdot)$ *applied to some* l*-tuple* $U = (U(1), U(2), \cdots, U(l))$, *we will allow some abuse of notation to write* $\lambda(U) = (U(\lambda(1)), U(\lambda(2)), \cdots, U(\lambda(l)))$.

[1] We will use the symbol C to represent the expression in (1) throughout this paper.

Manuscript received March 30, 2017; revised July 1, 2017; accepted July 2, 2017. Date of publication July 11, 2017; date of current version August 29, 2017. This work was supported in part by NSF under Grant CCF-1617504 and Grant CNS-1731384, and in part by ONR under Grant N00014-16-1-26. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Aris Gkoulalas Divanis. *(Corresponding author: Hua Sun.)*

The authors are with the Center of Pervasive Communications and Computing, Department of Electrical Engineering and Computer Science, University of California at Irvine, Irvine, CA 92697 USA (e-mail: huas2@uci.edu; syed@uci.edu).

Digital Object Identifier 10.1109/TIFS.2017.2725225

II. Problem Statement

There are K messages $W_1, \cdots, W_K$, each of which is an arbitrary string of length L comprised of M-ary symbols.

$$W_k = \Big(W_k(1), W_k(2), \cdots, W_k(L)\Big) \in [0 : M-1]^L \quad \forall k \in [1 : K] \tag{2}$$

Note that there are M^L possible distinct realizations of each message. Note that message realizations are arbitrary, i.e., we do not enforce any statistical assumption on the message symbols.

There are N databases. Each database stores all the messages $W_1, \cdots, W_K$.

Depending upon the desired message index $\theta \in [1 : K]$, the user follows one of K strategies. These strategies are specified in terms of KN random queries, $Q_n^{[\theta]}$, $\forall n \in [1 : N], \forall \theta \in [1 : K]$ that are privately generated by the user a-priori, i.e., without any knowledge of the message realizations. In order to retrieve W_θ, the user sends the query $Q_n^{[\theta]}$ to the n-th database, $\forall n \in [1 : N]$.

Upon receiving $Q_n^{[\theta]}$, the n-th database returns an answering string $A_n^{[\theta]}$, which is a function of $Q_n^{[\theta]}$ and the data stored (i.e., messages $W_1, \cdots, W_K$). The answering string $A_n^{[\theta]}$ is comprised of M'-ary symbols, $A_n^{[\theta]} \in [0 : M'-1]^{|A_n^{[\theta]}|}$.

From all the information that is now available to the user $(A_{[1:N]}^{[\theta]}, Q_{[1:N]}^{[\theta]})$, he must be able to correctly decode the desired message W_θ. That is, the following correctness constraint must be satisfied.

$$\text{[Correctness] } W_\theta \text{ is a deterministic function of } A_{[1:N]}^{[\theta]}, Q_{[1:N]}^{[\theta]}. \tag{3}$$

To protect the user's privacy, the query presented to each database must be identically distributed regardless of the desired message index.

$$\text{[Privacy] } Q_n^{[\theta]} \sim Q_n^{[\theta']}, \quad \forall \theta, \theta' \in [1 : K], n \in [1 : N]. \tag{4}$$

Note that the databases do not collude so that the privacy constraint (4) is specified with respect to each individual database.

The download cost, D, for a PIR scheme is the maximum value (across all random realizations of queries) of the total number of M'-ary symbols downloaded by the user from all the databases.

$$D = \max \sum_{n=1}^{N} |A_n^{[\theta]}| \tag{5}$$

Our goal is to characterize the optimal (minimum over all PIR schemes) download cost D_L, for *arbitrary* fixed message size L. The optimality is across *all* PIR schemes, i.e., including non-linear PIR schemes.

III. Results

A. Optimal Download Cost for Matching Alphabet ($M = M'$)

Consider the setting where the messages and downloads are comprised of symbols from the same alphabet, i.e., $M = M' \in \mathbb{N}/\{1\}$. Our main result for this setting appears in the following theorem.

Theorem 1: For PIR with $N \in \mathbb{N}$ databases, each storing all $K \in \mathbb{N}$ messages, each message comprised of $L \in \mathbb{N}$ symbols from M-ary alphabet, $M \in \mathbb{N}/\{1\}$, where the downloads are comprised of symbols from the same M-ary alphabet, the optimal download cost is $D_L = \left\lceil \frac{L}{C} \right\rceil$ M-ary symbols.

The proof of converse (i.e., the impossibility claim) of Theorem 1 follows from the capacity result of [20] and appears in Section IV. The achievability is proved, first for the case $L = N^{K-1}$ in Section V, and then for arbitrary L in Section VI.

Based on Theorem 1, the following observations are in order.

1) Given the message size and alphabet constraints, since the minimum download cost corresponds to the maximum rate, Theorem 1 equivalently characterizes the optimal rate for arbitrary message size in the matching alphabet case, as $L/\lceil \frac{L}{C} \rceil$.
2) Reference [11] shows that when $K \geq 2$ and $N \geq L+1$, then the optimal download is $D_L = L + 1$. This result can be recovered as a special case of Theorem 1 by noting that when $K \geq 2$ and $N \geq L + 1$,
$$D_L = \left\lceil \frac{L}{C} \right\rceil \tag{6}$$
$$= L + \left\lceil L\left(\frac{1}{N} + \frac{1}{N^2} + \cdots + \frac{1}{N^{K-1}}\right)\right\rceil \tag{7}$$
$$= L + 1 \tag{8}$$
where (8) follows because $0 < \frac{1}{N} + \frac{1}{N^2} + \cdots + \frac{1}{N^{K-1}} < \frac{1}{N-1} \leq \frac{1}{L}$. Theorem 1 completes the picture by characterizing the optimal download cost for all N, K, L.
3) Reference [20] presents a PIR scheme which achieves a rate equal to capacity C if $L = nN^K$ where $n \in \mathbb{N}$ is any positive integer, so that the corresponding download is $D = \frac{L}{C}$. This result can be recovered as a special case of Theorem 1 by noting that when $L = nN^K$, then $\frac{L}{C} = nN^K(1+1/N+\cdots+1/N^{K-1}) = nN(1+N+\cdots+N^{K-1})$ is a positive integer so that $D_L = \left\lceil \frac{L}{C} \right\rceil = \frac{L}{C}$.
4) A naive extension of the PIR scheme of [20] to the setting when L is not an integer multiple of N^K, is obtained by padding zeros to each message so that the message lengths are rounded up to the closest integer multiple of N^K. The gap between the download cost of the naive scheme and the optimal download cost in Theorem 1 can be unbounded. For an example, if $L = N^{K-1}$, then the download cost of the naive scheme is $D = N^K/C$, while the optimal download cost is $D_L = \left\lceil \frac{L}{C} \right\rceil = N^{K-1}/C$.
5) In the absence of any constraints on message lengths, we know from [20] that the maximum achievable rate across all PIR schemes is the capacity C. For constrained message length L, Theorem 1 shows that the maximum achievable rate is $L/D_L = L/\left\lceil \frac{L}{C} \right\rceil$ which is in general less than C. The message length $L = N^{K-1}$ is particularly significant in light of Theorem 1, because this is the shortest message length for which the achieved rate equals the capacity C. This is seen as follows.

In order to achieve the capacity, the download cost must be $D = \frac{L}{C} = D_L$ which must be a positive integer value. But if $L < N^{K-1}$, then

$$D = \frac{L}{C} = L\left(1 + \frac{1}{N} + \cdots + \frac{1}{N^{K-1}}\right) \tag{9}$$

$$= L\left(\frac{1 + N + \cdots + N^{K-1}}{N^{K-1}}\right) \notin \mathbb{N} \tag{10}$$

because N^{K-1} and $1 + N + \cdots + N^{K-1}$ are co-prime. This is verified, e.g., through Bezout's identity,

$$N^{K-1}(N) + (1 + N + \cdots + N^{K-1})(1 - N) = 1 \tag{11}$$

B. Optimal Download Cost for Mismatched Alphabet ($M \neq M'$)

Now consider PIR schemes with mismatched alphabet, i.e., the messages are represented in M-ary alphabet, and the downloaded symbols are in M'-ary alphabet, $M' \neq M$. For this setting the optimal download cost to within 2 symbols is characterized in the following theorem.

Theorem 2: For PIR with $N \in \mathbb{N}$ databases, each storing all $K \in \mathbb{N}$ messages, each message comprised of $L \in \mathbb{N}$ symbols from M-ary alphabet, $M \in \mathbb{N}/\{1\}$, where the downloads are comprised of symbols from M'-ary alphabet, $M' \in \mathbb{N}/\{1\}$, $M' \neq M$, the optimal download cost $D_L \in \left\{\left\lceil \frac{L'}{C} \right\rceil, \left\lceil \frac{L'}{C} \right\rceil - 1, \left\lceil \frac{L'}{C} \right\rceil - 2\right\}$, where $L' = \lceil L \log_{M'} M \rceil$.

The proof of Theorem 2 appears in Section VII.

The following observations place Theorem 2 in perspective.

1) The proof of Theorem 2 presented in Section VII shows that the download cost $\left\lceil \frac{\lceil L \log_{M'} M \rceil}{C} \right\rceil$ is always achievable, and the download cost for any PIR scheme cannot be less than $\lceil \frac{L \log_{M'} M}{C} \rceil$. Therefore, in particular, when $\left\lceil \frac{\lceil L \log_{M'} M \rceil}{C} \right\rceil = \lceil \frac{L \log_{M'} M}{C} \rceil$, the *exact* optimal download cost is $D_L = \lceil \frac{L \log_{M'} M}{C} \rceil$.
2) It is easy to create examples where mismatched alphabet leads to less efficient PIR schemes than possible with matched alphabets. However, this is not always the case. The following examples show how mismatched alphabet can in some cases be beneficial in terms of rate relative to matched alphabet. Consider $N = 2, K = 2, L = 3, M = 9$. Here $C = 2/3$. The highest rate achievable with matched alphabet ($M' = M$) is $\frac{L}{\lceil L/C \rceil} = 3/5 < C$ whereas the rate achieved with the mismatched alphabet $M' = 3 < M$, is $\frac{L \log_{M'} M}{\lceil L'/C \rceil} = 2/3 = C$. Similarly one can construct examples with $M' > M$ where mismatched alphabet produces a higher rate than the best possible with matched alphabet, e.g., $N = 2, K = 2, L = 3, M = 4$ where the best rate with matched alphabet is again $3/5 < C$, but the mismatched alphabet $M' = 8$ achieves rate $2/3 = C$.

IV. Proof of Theorem 1: Converse

The converse for Theorem 1 is the impossibility claim, i.e., that no PIR scheme with matched alphabet ($M = M'$) can achieve a download cost smaller than $D_L = \lceil \frac{L}{C} \rceil$. This is proved as follows.

The message realizations are arbitrary, as is the choice of the desired message index $\theta \in [1 : K]$. By arbitrary, what is meant is that all realizations are possible. Therefore the PIR scheme must work for every possible realization of message symbols and θ. Any PIR scheme that works for arbitrary realizations, will also work if they are uniformly randomly generated. Therefore, for the converse argument let us assume uniform distributions on the realizations of message symbols, and on θ. The advantage of assigning a distribution to these arbitrary quantities is that we are able to use the information theoretic formulation of the PIR problem as in [20], and the upper bounds on rate that are derived in [20] are also applicable in our current setting. In particular, C is still an upper bound on the achievable rate of a PIR scheme with arbitrary message realizations and θ and arbitrary message length L. Since capacity is an upper bound on the rate of all PIR schemes, $C \geq L/D_L$, so that $D_L \geq \frac{L}{C}$, and because $D_L \in \mathbb{N}$, we must have $D_L \geq \lceil \frac{L}{C} \rceil$.

V. Proof of Theorem 1: Achievability for $L = N^{K-1}$

In [20], it is shown that the capacity (and the corresponding optimal download cost) of PIR is achievable when $L = N^K$ bits. Here we present a more efficient PIR scheme to show that a smaller message size, $L = N^{K-1}$ bits (M-ary symbols in general), is sufficient to achieve a rate equal to C (and the optimal download cost) when the alphabets are matched, i.e., $M = M'$. This PIR scheme is significant because (as noted in Observation 5, Section III-A) $L = N^{K-1}$ is the smallest message size needed to achieve capacity, and also because it is the key ingredient that will allow us to subsequently expand the achievability proof to arbitrary L in Section VI. Note that since the $N = 1$ case is trivial (optimal to download all messages), we will consider only $N \geq 2$ in this section.

The PIR scheme that we present here is closely related to the capacity achieving PIR scheme presented in [20]. For both schemes the queries are comprised only of sums of symbols from various messages. Since our new scheme considers M-ary alphabet, the "sums" are interpreted as modulo-M sums. In both schemes no symbol appears more than once in the query for any particular database. The difference between the two schemes lies in the requirement of symmetry across databases. Recall that the PIR scheme of [20] is based on the iterative application of three steps corresponding to symmetry across databases, symmetry across messages within the query to each database, and exploiting side information. The key to reducing the message size from $L = N^K$ to $L = N^{K-1}$ is to eliminate the requirement of symmetry across databases. Therefore, the new PIR scheme for $L = N^{K-1}$, formalized in the **Q-Gen** Algorithm in Section V-D, is based on the iterative application of the following two steps.

(1) *Enforcing Message Symmetry within the Queries to Each Database*: The goal is to make the queries to a database symmetric with respect to messages.

For instance if the query to database 1 includes l instances of sums of symbols from messages W_1, W_2, W_3, then it must include l instances of sums of symbols from each of the $\binom{K}{3}$ combinations of 3 messages. Message symmetry is defined formally in Section V-B. The procedure is formalized in the **M-Sym** Algorithm, presented in Section V-C. All the queries that do not involve desired message symbols ($\mathcal{I}$ terms in the **Q-Gen** Algorithm) are introduced only through the **M-Sym** algorithm.

(2) *Exploiting Side Information*: The goal of this step is to exploit queries from other databases that were added to enforce message symmetry (and do not contain desired message symbols), as side information to construct new queries which are sums of symbols from desired message and the side information available from other databases. This step is formalized in the **Exploit-SI** Algorithm, presented in Section V-C. Except for an initialization step, all the queries involving desired message symbols ($\mathcal{M}$ terms in the **Q-Gen** Algorithm) are introduced only through the **Exploit-SI** algorithm.

Let us start with a few simple examples for small K, N values to illustrate the key ideas.

A. Examples

1) $K = 2$ Messages, $N = 2$ Databases, $L = N^{K-1} = 2$ Symbols per Message: Let $[a_1, a_2]$ represent a random permutation of $L = 2$ symbols from W_1. Similarly, let $[b_1, b_2]$ represent an independent random permutation of $L = 2$ symbols from W_2. The key to the privacy of the scheme is that these random permutations are generated privately by the user and are unknown to the databases.

Suppose the desired message is W_1, i.e., $\theta = 1$. The PIR scheme always starts by requesting the first desired symbol (in this case, a_1) from the first database (DB1). Applying Step (1), we achieve message symmetry by including b_1 from DB1. Next we apply Step (2) to exploit the side information available at DB1, i.e., b_1, in order to retrieve a new desired symbol a_2 from the second database (DB2) by mixing it with b_1. At this point the query to each database is symmetric, and there is no side information that remains unexploited. Thus the construction is complete.

DB1	DB2
a_1	

$\xrightarrow{(1)}$

DB1	DB2
a_1, b_1	

$\xrightarrow{(2)}$

DB1	DB2
a_1, b_1	$a_2 + b_1$

Similarly, the queries for $\theta = 2$ are constructed as follows.

DB1	DB2
b_1	

$\xrightarrow{(1)}$

DB1	DB2
a_1, b_1	

$\xrightarrow{(2)}$

DB1	DB2
a_1, b_1	$a_1 + b_2$

Note that the application of Step (1) only introduces new terms that do not involve symbols from the desired message, whereas the application of Step (2) only introduces new terms that involve symbols from the desired message.

To see why this scheme is private, recall that $[a_1, a_2]$ are random permutations of two symbols from W_1 and $[b_1, b_2]$ are random permutations of two symbols from W_2. These permutations are known only to the user, and not to the databases. Therefore, regardless of whether $\theta = 1$ or $\theta = 2$, DB1 is asked for one randomly chosen symbol of each message, and DB2 is asked for a sum of a pair of randomly chosen symbols from each message. Since the permutations are uniform, all possible realizations are equally likely, and privacy is guaranteed. A formal proof of privacy for the general setting appears in Section V-F.

The scheme is correct, because each desired message symbol is either downloaded directly or as a sum with side information terms that are separately downloaded.

Finally, note that the download cost is $D = 3 = \lceil \frac{L}{C} \rceil$, because $C = 2/3$ for this case. The rate achieved is $L/D = 2/3 = C$.

2) $K = 3$ Messages, $N = 2$ Databases, $L = N^{K-1} = 4$ Symbols per Message: Let $[a_1, \cdots, a_4]$ represent a random permutation of 4 M-ary symbols from message W_1. Similarly, $[b_1, \cdots, b_4]$ and $[c_1, \cdots, c_4]$ are random permutations of 4 M-ary symbols each from messages W_2, W_3, respectively. The uniformly random and independent permutations are generated privately by the user. Suppose $\theta = 1$. The query generation algorithm proceeds as follows.

DB1	DB2
a_1	

$\xrightarrow{(1)}$

DB1	DB2
a_1, b_1, c_1	

$\xrightarrow{(2)}$

DB1	DB2
a_1, b_1, c_1	$a_2 + b_1$
	$a_3 + c_1$

$\xrightarrow{(1)}$

DB1	DB2
a_1, b_1, c_1	$a_2 + b_1$
	$a_3 + c_1$
	$b_2 + c_2$

$\xrightarrow{(2)}$

DB1	DB2
a_1, b_1, c_1	$a_2 + b_1$
$a_4 + b_2 + c_2$	$a_3 + c_1$
	$b_2 + c_2$

Again, note that the application of Step (1) only introduces new terms that do not involve symbols from the desired message, whereas the application of Step (2) only introduces new terms that involve symbols from the desired message. The queries generated when $\theta = 2, 3$ are as follows.

$\theta = 2$

DB1	DB2
a_1, b_1, c_1	$a_1 + b_2$
$a_2 + b_4 + c_2$	$b_3 + c_1$
	$a_2 + c_2$

$\theta = 3$

DB1	DB2
a_1, b_1, c_1	$a_1 + c_2$
$a_2 + b_2 + c_4$	$b_1 + c_3$
	$a_2 + b_2$

Correctness is straightforward, privacy is ensured by message symmetry and random permutations, and the rate is $L/D = 4/7$ which matches the capacity C for this case. The download achieved is $D = 4$ symbols which is also optimal.

3) $K = 3$ Messages, $N = 3$ Databases, $L = N^{K-1} = 9$ Symbols per Message: Let $[a_1, \cdots, a_9], [b_1, \cdots, b_9], [c_1, \cdots, c_9]$ be three i.i.d. uniform permutations of symbols from messages W_1, W_2, W_3, respectively. The query generation algorithm for $\theta = 1$

proceeds as follows.

DB1	DB2	DB3
a_1		

$\xrightarrow{(1)}$

DB1	DB2	DB3
a_1, b_1, c_1		

$\cdots$

$\cdots \xrightarrow{(2)}$

DB1	DB2	DB3
a_1, b_1, c_1	$a_2 + b_1$	$a_4 + b_1$
	$a_3 + c_1$	$a_5 + c_1$

$\cdots$

$\cdots \xrightarrow{(1)}$

DB1	DB2	DB3
a_1, b_1, c_1	$a_2 + b_1$	$a_4 + b_1$
	$a_3 + c_1$	$a_5 + c_1$
	$b_2 + c_2$	$b_3 + c_3$

$\cdots$

$\cdots \xrightarrow{(2)}$

DB1	DB2	DB3
a_1, b_1, c_1	$a_2 + b_1$	$a_4 + b_1$
$a_6 + b_2 + c_2$	$a_3 + c_1$	$a_5 + c_1$
$a_7 + b_3 + c_3$	$b_2 + c_2$	$b_3 + c_3$
	$a_8 + b_3 + c_3$	$a_9 + b_2 + c_2$

Again, note that the application of Step (1) only introduces new terms that do not involve symbols from the desired message, whereas the application of Step (2) only introduces new terms that involve symbols from the desired message. The scheme when $\theta = 2, 3$ is as follows.

$\theta = 2$

DB1	DB2	DB3
a_1, b_1, c_1	$a_1 + b_2$	$a_1 + b_4$
$a_2 + b_6 + c_2$	$b_3 + c_1$	$b_5 + c_1$
$a_3 + b_7 + c_3$	$a_2 + c_2$	$a_3 + c_3$
	$a_3 + b_8 + c_3$	$a_2 + b_9 + c_2$

$\theta = 3$

DB1	DB2	DB3
a_1, b_1, c_1	$a_1 + c_2$	$a_1 + c_4$
$a_2 + b_2 + c_6$	$b_1 + c_3$	$b_1 + c_5$
$a_3 + b_3 + c_7$	$a_2 + b_2$	$a_3 + b_3$
	$a_3 + b_3 + c_8$	$a_2 + b_2 + c_9$

Correctness is straightforward, privacy is ensured by message symmetry and random permutations, and the rate is $L/D = 9/13$ which matches the capacity C for this case. The download achieved is $D = 13$ symbols which is also optimal.

Next we go beyond the simple examples to the general N, K setting. Let us start by introducing some new definitions and notation, some of which is needed only to suppress those aspects of the general setting that are notationally cumbersome but otherwise inconsequential.

B. Definitions and Additional Notation

1) U_k: For all $k \in [1 : K]$, define[2] ordered tuples

$$U_k \triangleq [U_k(1), U_k(2), \cdots, U_k(N^{K-1})] \tag{12}$$

[2]The U_k symbols will eventually be mapped to random permutations of message W_k symbols. We use $[U_k(l)]$ instead of, say $[a_l], [b_l]$ as in the examples, because while the latter notation is more clear, it does not generalize to K messages.

2) k-Sums, Types: We use the terminology **k-sum** to denote an expression representing the sum of k distinct variables, each drawn from a *different* U_i tuple, i.e., $U_{i_1}(j_1) + U_{i_2}(j_2) + \cdots + U_{i_k}(j_k)$, where $i_1, i_2, \cdots, i_k \in [1 : K]$ are all *distinct* indices. Furthermore, we will define such a k-sum to be of **type** $\{i_1, i_2, \cdots, i_k\}$, or $i_{[1:k]}$ in our compact notation. If q represents a k-sum, the function $\text{type}(q)$ returns its type. Denote $\mathcal{T}_k$ as the set of all possible types of a k-sum, i.e., all possible choices of k distinct indices in $[1 : K]$. Note that $|\mathcal{T}_k| = \binom{K}{k}$.

The next two items are introduced to facilitate a compact notation. The first of these is a function, $\texttt{new}(\cdot)$, which will allow us to suppress unimportant details about symbol indices.

3) The $\texttt{new}(\cdot)$ Function: For any ordered tuple U, let $\texttt{new}(U)$ be a function that, starting with $U(1)$, returns the "next" element in U each time[3] it is called with the same tuple U as its argument. So, for example, the following sequence of calls to this function: $\texttt{new}(U_2), \texttt{new}(U_1), \texttt{new}(U_1), \texttt{new}(U_1) + \texttt{new}(U_2)$ will produce $U_2(1), U_1(1), U_1(2), U_1(3) + U_2(2)$ as the output.

4) Ordered Access to Elements of a Set: In a similar spirit, for any set A, when accessing its elements (e.g., in an algorithm), we will use the notation $\overrightarrow{A}$ to indicate that the elements of A are to be accessed in some specified order, the details of which are not significant, because all ordering rules will produce (possibly different) optimal PIR schemes. Let us assume by default that the ordering is the natural lexicographic increasing order. For example, $\overrightarrow{[1 : K]}$ refers to increasing order of integers 1 through K. $\overrightarrow{\mathcal{T}_k}$ denotes that the types, i.e., the $\{i_1, i_2, \cdots, i_k\}$ terms in $\mathcal{T}_k$ are individually sorted and then accessed in lexicographic increasing order. For a set Q that is comprised of various k-sums the notation $\overrightarrow{Q}$ represents that the order in which the elements are accessed is, first in increasing order of k, then within the same k in increasing order of type, and then for multiple instances of the same type the elements are accessed in increasing order of the j index of the $U_i(j)$ with the smallest i. Some examples of this notation:

$$\bigcup_{k \in \overrightarrow{[1:2]}} \{U_1(k) + \texttt{new}(U_2)\} = \{U_1(1) + U_2(1), U_1(2) + U_2(2)\}$$

$$\bigcup_{q \in \overrightarrow{Q}} \{q + \texttt{new}(U_1)\} = \{U_1(1) + U_2(4), U_1(2) + U_2(2) + U_3(3), U_1(3) + U_2(3) + U_3(2)\}$$

where $Q = \{U_2(2) + U_3(3), U_2(4), U_2(3) + U_3(2)\}$, so that $\overrightarrow{Q}$ denotes that the terms of Q are accessed in the order $U_2(4), U_2(2) + U_3(3), U_2(3) + U_3(2)$.

5) The Count$(\cdot)$ and Max$(\cdot)$ Functions: $\text{Count}(Q, i_{[1:k]})$ denotes the number of k-sums of type $\{i_1, i_2, \cdots, i_k\}$ that are present in Q

$$\text{Count}(Q, i_{[1:k]}) \triangleq |\{q : \ q \in Q, \text{type}(q) = i_{[1:k]}\}|, \tag{13}$$

$\text{Max}(Q, k)$ denotes the maximum of the number of k-sums of the same type in Q, with the maximization being across all

[3]We will deal with N^{K-1}-tuples and the algorithms will guarantee that the $\texttt{new}(.)$ function is not called more than N^{K-1} times for the same tuple.

types of k-sums,

$$\text{Max}(Q,k) \stackrel{\triangle}{=} \max_{i_{[1:k]}\in\mathcal{T}_k} \text{Count}(Q, i_{[1:k]}) \tag{14}$$

6) Message Symmetry: Message symmetry is defined as the condition that $\forall k \in [1:K]$, Q contains equal number of k-sums for every type $\{i_1, i_2, \cdots, i_k\} \in \mathcal{T}_k$.

$$\text{Count}(Q, i_{[1:k]}) = \text{Count}(Q, i'_{[1:k]}), \quad \forall i_{[1:k]}, i'_{[1:k]} \in \mathcal{T}_k \tag{15}$$

C. Two Subroutines

For the sake of clarity, here we separately present the two procedures needed to implement the message symmetry and side information exploitation steps, which will ultimately be incorporated into the overall query generation algorithm.

1) Algorithm (1)*: Achieving Message Symmetry (M-Sym Algorithm):* The algorithm takes as input a set Q comprised of various k-sums, and produces as output a set Q^* comprised of additional terms that need to be included in Q to make it message symmetric, i.e., $Q \cup Q^*$ satisfies message symmetry. For each $k \in [1:K]$, and for each type $i_{[1:k]} \in \mathcal{T}_k$, the algorithm checks if there are $\text{Max}(Q,k)$ instances of that type, and if not, then it generates as many new instances as necessary to bring up the number of instances of that type to $\text{Max}(Q,k)$.

Algorithm (1) M-Sym Algorithm

Input: Q
Output: Q^*
Initialize: $Q^* \leftarrow \emptyset$.
for $k = 1 : K$ **do**
  for each $i_{[1:k]} \in \overrightarrow{\mathcal{T}_k}$ **do**
    if $\text{Count}(Q, i_{[1:k]}) < \text{Max}(Q,k)$ **then**
      for $i = 1 : \text{Max}(Q,k) - \text{Count}(Q, i_{[1:k]})$ **do**

$$Q^* \leftarrow Q^* \cup \{\texttt{new}(U_{i_1}) + \texttt{new}(U_{i_2}) + \cdots + \texttt{new}(U_{i_k})\}$$

      end for (i)
    end if
  end for $(i_{[1:k]})$
end for (k)

Note that $Q \cup Q^*$ satisfies message symmetry because for all types $i_{[1:k]} \in \mathcal{T}_k$, $\text{Count}(Q \cup Q^*, i_{[1:k]}) = \text{Max}(Q \cup Q^*, k) = \text{Max}(Q,k)$.

To illustrate the M-Sym Algorithm, let us revisit the examples presented earlier in Section V-A. In all three examples, when invoking Step (1), we run the M-Sym Algorithm once for each database. For example, consider the third example with $K = 3$ messages, $N = 3$ databases, and desired message index $\theta = 1$. When we invoke Step (1) for the second time, consider DB2. The input to the M-Sym Algorithm is $Q = \{a_2 + b_1, a_3 + c_1\}$, which is not yet symmetric, and the output of the M-Sym Algorithm is $Q^* = \{b_2 + c_2\}$, whose union with Q is now symmetric. The symmetry of the messages is important to prove the privacy of the PIR scheme (see Lemma 3).

2) Algorithm (2)*: Exploiting Side Information (Exploit-SI Algorithm):* Algorithm (2) formalizes the side information exploitation step. This algorithm takes as input N query sets $Q_1, Q_2, \cdots, Q_N$, which are comprised of side information terms, i.e., terms that do not contain any desired message symbols, i.e., $\forall n \in [1:N]$ and $\forall q \in Q_n$, $\theta \notin \text{type}(q)$ and which have not previously been exploited. The algorithm produces N sets $Q'_1, Q'_2, \cdots, Q'_N$ as output. $Q'_n, n \in [1:N]$ is built by combining each element q in $Q_1, \cdots, Q_{n-1}, Q_{n+1}, \cdots, Q_N$ with a "new" variable U_θ (which corresponds to a desired message symbol).

Algorithm (2) Exploit-SI Algorithm

Input: $Q_1, Q_2, \cdots, Q_N$
Output: $Q'_1, Q'_2, \cdots, Q'_N$
Initialize: All output are initialized as null sets.
for $n = 1 : N$ **do**
  for $n' = 1 : N$ **and** $n' \neq n$ **do**
    for each $q \in \overrightarrow{Q_{n'}}$ **do**

$$Q'_n \leftarrow Q'_n \cup \{\texttt{new}(U_\theta) + q\}$$

    end for (q)
  end for (n')
end for (n)

To illustrate the Exploit-SI Algorithm, let us revisit the examples presented earlier in Section V-A. In all three examples, when invoking Step (2), we run the Exploit-SI Algorithm once. For example, consider the third example with $K = 3$ messages, $N = 3$ databases and desired message index $\theta = 1$. When we invoke Step (2) for the second time, the input to the Exploit-SI Algorithm is $Q_1 = \{\emptyset\}$, $Q_2 = \{a_2 + b_1, a_3 + c_1, b_2 + c_2\}$, $Q_3 = \{a_4 + b_1, a_5 + c_1, b_3 + c_3\}$, and the output of the Exploit-SI Algorithm is $Q'_1 = \{a_6 + b_2 + c_2, a_7 + b_3 + c_3\}$, $Q'_2 = \{a_8 + b_3 + c_3\}$, $Q'_3 = \{a_9 + b_2 + c_2\}$, where each side information symbol from other databases is used to retrieve a new desired symbol. The exploitation of side information is important for the efficiency of the PIR scheme such that the rate achieved matches the capacity (see Lemma 2).

D. A Deterministic Query Generation Algorithm (Q-Gen Algorithm)

We now proceed to a query generation algorithm.[4] The algorithm produces N query sets $Q(\text{DB}, \theta)$, for all $\text{DB} \in [1:N]$ as functions of θ. For internal book-keeping in the algorithm, we will partition each query set into K subsets called blocks, such that block $k \in [1:K]$ contains only k-sums. Further we will partition each block into two subsets denoted by $\mathcal{I}$ and $\mathcal{M}$ such that the $\mathcal{M}$ partition contains only those types of k-sums which involve variables from U_θ, and

[4]Note that this is not the final step in the query generation. The output of this deterministic algorithm is in terms of the U_k variables. The final step, to be presented in Section V-E, maps U_k variables to private random permutations of W_k variables, to produce the random queries that are then sent to the databases.

the $\mathcal{I}$ partition contains the remaining k-sums which do not involve the U_θ variables.

As in the simple examples presented earlier, for all $\text{DB} \in [1:N], \theta \in [1:K]$, the query sets $Q(\text{DB}, \theta)$ are built starting only from a single element in $Q(1, \theta)$, which is the first desired message symbol U_θ, and then evolves through iterative application of the M-Sym and Exploit-SI sub-routines. Note that the memory of calls to the $\texttt{new}(\cdot)$ function is assumed to be global, i.e., the memory is not reset when the sub-routines are called. Similarly, θ is assumed to be available to the sub-routines as a global variable.

Algorithm (3) Q-Gen Algorithm

Input: θ

Output: $Q(1, \theta), \cdots, Q(N, \theta)$

Initialize: All query sets are initialized as null sets. Also initialize $\text{Block} \leftarrow 1$;

$$Q(1, \theta, \text{Block}, \mathcal{M}) \leftarrow \{\texttt{new}(U_\theta)\}$$
$$Q(1, \theta, \text{Block}, \mathcal{I}) \leftarrow \textbf{M-Sym}(Q(1, \theta, \text{Block}, \mathcal{M}))$$
$$\forall \text{DB} \in [2:N], \quad Q(\text{DB}, \theta, \text{Block}, \mathcal{M}) \leftarrow \emptyset, \quad Q(\text{DB}, \theta, \text{Block}, \mathcal{I}) \leftarrow \emptyset,$$

for $\text{Block} = 2 : K$ **do**

$$(Q(1, \theta, \text{Block}, \mathcal{M}), \cdots, Q(N, \theta, \text{Block}, \mathcal{M})) \leftarrow \textbf{Exploit-SI}(Q(1, \theta, \text{Block}-1, \mathcal{I}), \cdots, Q(N, \theta, \text{Block}-1, \mathcal{I}))$$

  for $\text{DB} = 1 : N$ **do**

$$Q(\text{DB}, \theta, \text{Block}, \mathcal{I}) \leftarrow \textbf{M-Sym}(Q(\text{DB}, \theta, \text{Block}, \mathcal{M}))$$

  end for (DB)

end for (Block)

for $\text{DB} = 1 : N$ **do**

$$Q(\text{DB}, \theta) \leftarrow \bigcup_{\text{Block} \in [K]} \big(Q(\text{DB}, \theta, \text{Block}, \mathcal{I}) \cup Q(\text{DB}, \theta, \text{Block}, \mathcal{M})\big)$$

end for (DB)

To illustrate the Q-Gen Algorithm, let us revisit the examples presented earlier in Section V-A. Consider the third example with $K = 3$ messages, $N = 3$ databases and desired message index $\theta = 1$. For Block = 1, the queries for DB1 are generated as $Q(1, \theta, \text{Block}, \mathcal{M}) = \{a_1\}$, $Q(1, \theta, \text{Block}, \mathcal{I}) = \{b_1, c_1\}$. For Block = 2 and Block = 3, we first exploit side information to retrieve more desired symbols, and then make the queries to each database symmetric to restore the privacy. We end at Block 3 because all side information is used and the queries to each database satisfy symmetry.

Based on Algorithm (3), we have two immediate observations.

1) Consider the number of instances with type $\{i_1, \cdots, i_{k-1}, \theta\}$ in $Q(\text{DB}, \theta, k, \mathcal{M})$, i.e.,
$$\text{Count}(Q(\text{DB}, \theta, k, \mathcal{M}), \{i_1, \cdots, i_{k-1}, \theta\}).$$
$Q(\text{DB}, \theta, k, \mathcal{M})$ is produced in Step 6 of Algorithm (3) as one of the outputs of the Expoit-SI algorithm. From Step 7 of the Expoit-SI algorithm, we know that the instances with type $\{i_1, \cdots, i_{k-1}, \theta\}$ in $Q(\text{DB}, \theta, k, \mathcal{M})$ are produced by combining a new variable from U_θ with each element of type $\{i_1, \cdots, i_{k-1}\}$ in $Q(\text{DB}', \theta, k-1, \mathcal{I}), \text{DB}' \neq \text{DB}$, i.e.,
$$\text{Count}(Q(\text{DB}, \theta, k, \mathcal{M}), \{i_1, \cdots, i_{k-1}, \theta\}) = \sum_{\text{DB}' \neq \text{DB}} \text{Count}(Q(\text{DB}', \theta, k-1, \mathcal{I}), \{i_1, \cdots, i_{k-1}\}) \quad \forall \text{DB} \in [1:N], \quad \theta \in [1:K], \ k \in [2:K], \ \{i_1, \cdots, i_{k-1}\} \in \mathcal{T}_{k-1} \tag{16}$$
2) From Step 4 and Step 8 of Algorithm (3), we know that $Q(\text{DB}, \theta, k, \mathcal{I}) \cup Q(\text{DB}, \theta, k, \mathcal{M}), \forall k \in [1:K]$ satisfies message symmetry (15).

1) Structure of $Q(DB, \theta)$: Key properties of $Q(\text{DB}, \theta)$ are summarized in the following lemma.

Lemma 1: $Q(\text{DB}, \theta)$ produced by Algorithm (3) satisfies the following properties.

1) $Q(\text{DB}, \theta), \forall \text{DB} \in [1:N], \theta \in [1:K]$ is a union of K disjoint sets (called "blocks"), that are indexed by $k \in [1:K]$. Block k only contains k-sums. For any type $i_{[1:k]} \in \mathcal{T}_k$, block k of $Q(\text{DB}, \theta)$ contains $v(\text{DB}, k)$ instances of type $i_{[1:k]}$, where $v(\text{DB}, k)$ is a function only of DB, k.
2) $\forall i \in [1:K]$, if $U_i(j)$ and $U_i(j')$ appear anywhere in the same $Q(\text{DB}, \theta)$ then $j \neq j'$.
3) Exactly $v(\text{DB}) \triangleq \sum_{k=1}^{K} v(\text{DB}, k)\binom{K-1}{k-1}$ distinct variables for each $U_i, i \in [1:K]$ appear in $Q(\text{DB}, \theta)$.

Proof:

1) Block $k, k \in [1:K]$ of $Q(\text{DB}, \theta)$ is the set $Q(\text{DB}, \theta, k, \mathcal{I}) \cup Q(\text{DB}, \theta, k, \mathcal{M})$, which satisfies message symmetry based on Observation 2. From Step 4 of Algorithm (3), we know that Block 1 only contains 1-sums. From Steps 6 and 8, we know that the type of each instance in Block $k, k \in [2:K]$ contains one more variable than that of any instance in Block $k-1$. Therefore, by induction, Block k only contains k-sums. As each Block k satisfies message symmetry, we have
$$\text{Count}(Q(\text{DB}, \theta, k, \mathcal{M}), \{i_1, \cdots, i_{k-1}, \theta\}) = \text{Max}(Q(\text{DB}, \theta), k) \tag{17}$$
$$\text{Count}(Q(\text{DB}, \theta, k-1, \mathcal{I}), \{i_1, \cdots, i_{k-1}\}) = \text{Max}(Q(\text{DB}, \theta), k-1) \tag{18}$$
and (16) reduces to
$$\text{Max}(Q(\text{DB}, \theta), k) = \sum_{\text{DB}' \neq \text{DB}} \text{Max}(Q(\text{DB}', \theta), k-1) \tag{19}$$
Combined with the fact that $\text{Max}(Q(1, \theta), 1) = 1$, $\text{Max}(Q(\text{DB}, \theta), 1) = 0, \forall \text{DB} \in [2:N]$ (obtained

from Step 4 of Algorithm (3)), we conclude that $\text{Max}(Q(\text{DB},\theta),k)$ depends only on DB and k. Therefore, $v(\text{DB},k)=\text{Max}(Q(\text{DB},\theta),k)$ and $v(\text{DB},k)$ is a function of only DB, k.

2) Fix any database DB. Consider the case where $i=\theta$ first. Note that desired variables only appear in $Q(\text{DB},\theta,\text{Block},\mathcal{M})$. From Step 4 and Step 6 in Algorithm (3), we see that the desired variables, i.e., the U_θ variables appear only through the $\texttt{new}(U_\theta)$ function so that each of them has a distinct index. Next, consider the non-desired variables, $U_i, i\neq\theta$, which either appear in Steps 4 and 8 through the $\texttt{new}(U_k)$ function or appear in Step 6 which in turn come from $Q(\text{DB},\theta,\text{Block}-1,\mathcal{M})$ and each of them was introduced once through the $\texttt{new}(U_k)$ function and used exactly once. Therefore, these U_k variables also have distinct indices within $Q(\text{DB},\theta)$.

3) This claim follows directly from the previous two claims. Note that we have shown that all variables from U_i are distinct, so $v(\text{DB})$ is equal to the number of times that variables in U_i appear in $Q(\text{DB},\theta)$. In the k-th block, $Q(\text{DB},\theta)$ contains $v(\text{DB},k)$ instances of k-sums of each type and there are $\binom{K}{k-1}$ types of k-sums that include i. Therefore, the number of instances of tuple U_i in block k is $v(\text{DB},k)\binom{K-1}{k-1}$. Summing over all K blocks, we have $v(\text{DB})=\sum_{k=1}^{K}v(\text{DB},k)\binom{K-1}{k-1}$.

■

According to Lemma 1 the query sets $Q(\text{DB},\theta)$ are comprised of K blocks, the k-th block contains $v(\text{DB},\theta)$ instances of every possible type of k-sum, and no $U_i(j)$ variable appears more than once in $Q(\text{DB},\theta)$. Therefore, the structure of the query set may be summarized in the following corollary.

Corollary 1: Given DB,θ, for every U_k, $k\in[1:K]$, there exists its permutation $\underline{U_k}$ that depends only on DB,θ,k,

$$\underline{U_k}\triangleq\lambda_{\text{DB},\theta,k}(U_k) \tag{20}$$

such that $Q(\text{DB},\theta)$ can be expressed as

$$Q(\text{DB},\theta)=\bigcup_{k\in\overrightarrow{[1:K]}}\ \bigcup_{i_{[1:k]}\in\overrightarrow{\mathcal{I}_k}}\ \bigcup_{l=1}^{v(\text{DB},k)}\{\texttt{new}(\underline{U}_{i_1})+\dots \dots\texttt{new}(\underline{U}_{i_2})+\cdots+\texttt{new}(\underline{U}_{i_k})\} \tag{21}$$

Remark: As an example, consider the example with $K=3$, $N=3, L=9$ that was presented earlier in Section V-A. Suppose $\text{DB}=2,\theta=3$. The query $Q(\text{DB},\theta)=Q(2,3)$ is reproduced as follows.

$$Q(2,3)=\{a_2+b_2,a_1+c_2,b_1+c_3,a_3+b_3+c_8\},$$

which can be equivalently written in the form in Corollary 1 by setting

$$\lambda_{2,3,1}(U_1)=(a_2,a_1,a_3,a_4,a_5,a_6,a_7,a_8,a_9) \tag{22}$$

$$\lambda_{2,3,2}(U_2)=(b_2,b_1,b_3,b_4,b_5,b_6,b_7,b_8,b_9) \tag{23}$$

$$\lambda_{2,3,3}(U_3)=(c_2,c_3,c_8,c_1,c_4,c_5,c_6,c_7,c_9) \tag{24}$$

Note that here $U_1=[a_1,\cdots,a_9], U_2=[b_1,\cdots,b_9], U_3=[c_1,\cdots,c_9]$.

E. Mapping to Message Symbols to Produce $Q_{DB}^{[\theta]}$

To produce the actual query sent to the databases, we map the $U_k(i)$ variables to message symbols. This mapping is specified by K privately chosen permutations $\gamma_1,\gamma_2,\cdots,\gamma_K$, each of which is uniformly random over all possible $(N^{K-1})!$ permutations over the index set $[1:N^{K-1}]$ and the permutations are independent of each other and of θ. Specifically, $U_k(i)$ is replaced with $W_k(\gamma_k(i))$, $\forall k\in[1:K], i\in[1:N^{K-1}]$. This operator is denoted by Γ . For example, $\Gamma(\{U_1(2),U_3(4)+U_5(6)\})=\{W_1(\gamma_1(2)),W_3(\gamma_3(4))+W_5(\gamma_5(6))\}$. After this random mapping is applied to $Q(\text{DB},\theta)$, we obtain the actual query set $Q_{\text{DB}}^{[\theta]}$ that is sent to database DB.

$$Q_{\text{DB}}^{[\theta]}=\text{“}\Gamma\big(Q(\text{DB},\theta)\big)\text{”} \tag{25}$$

We use the double-quotes notation around a symbol to represent the *query* about its realization. For example, while $W_1(1)$ is the realization of one message symbol, in our notation “$W_1(1)$” only represents the *question*: “what is the value of $W_1(1)$?” $Q_{\text{DB}}^{[\theta]}$ is a (unordered) set and the questions in the set are sent in an order that is independent of θ (say, uniformly random) to the databases.

F. Proof of Correctness, Privacy and Optimality

We prove that the achievable scheme is correct, private and optimal in the following two lemmas.

Lemma 2: The PIR scheme constructed through the **Q-Gen** Algorithm is correct, i.e., it satisfies (3). The message size is $L=N^{K-1}$ and the download cost is optimal, $D=\frac{L}{C}$.

Remark: $\frac{L}{C}$ is an integer, so that $D_L=\lceil\frac{L}{C}\rceil=\frac{L}{C}$.

Proof: Note that all desired message symbols are either retrieved directly with no interference or they appear with interference q that is downloaded separately from another database so it can be subtracted to retrieve the desired symbols. Therefore, all the desired message symbols are retrievable and the correctness constraint (3) is satisfied.

In order to compute the message size and download cost, we proceed as follows. Using (19), we have

$$v(1,1)=1 \tag{26}$$

$$v(\text{DB},1)=0,\quad\forall\text{DB}\in[2:N] \tag{27}$$

$$v(\text{DB},k)=\sum_{\text{DB}'\neq\text{DB}}v(\text{DB}',k-1),\forall k\in[2:K] \tag{28}$$

$$\Rightarrow v(2,k)=\cdots=v(N,k),\forall k\in[2:K] \tag{29}$$

Now we know that the number of instances of each type over each block is the same for databases 2 to N. Next we derive the total number of instances of each type over each block across all databases. For all $k\in[2:K]$,

$$v(1,k)\overset{(28)(29)}{=}(N-1)v(2,k-1) \tag{30}$$

$$v(2,k)\overset{(28)(29)}{=}v(1,k-1)+(N-2)v(2,k-1) \tag{31}$$

$$\Rightarrow\quad v(1,k)+(N-1)v(2,k)\overset{(30)(31)}{=}(N-1)v(2,k-1)+(N-1)\Big(v(1,k-1)+(N-2)v(2,k-1)\Big) \tag{32}$$

$$= (N-1)\Big(v(1,k-1)+(N-1)v(2,k-1)\Big) \tag{33}$$

$$= \cdots \tag{34}$$

$$= (N-1)^{k-1}\Big(v(1,1)+(N-1)v(2,1)\Big) \tag{35}$$

$$\overset{(27)}{=} (N-1)^{k-1} \tag{36}$$

From Lemma 1, we have shown that there are $v(\mathrm{DB}) = \sum_{k=1}^{K} v(\mathrm{DB},k)\binom{K-1}{k-1}$ desired variables in each $Q(\mathrm{DB},\theta)$. Note that desired variables all appear through $\mathtt{new}(U_\theta)$ so that they are distinct across databases. Thus the message size (the total number of desired symbols that are retrieved) is

$$L = \sum_{\mathrm{DB}=1}^{N}\sum_{k=1}^{K} v(\mathrm{DB},k)\binom{K-1}{k-1} \tag{37}$$

$$\overset{(29)}{=} \sum_{k=1}^{K}\Big(v(1,k)+(N-1)v(2,k)\Big)\binom{K-1}{k-1} \tag{38}$$

$$\overset{(36)}{=} \sum_{k=1}^{K}(N-1)^{k-1}\binom{K-1}{k-1} \tag{39}$$

$$= \sum_{k=0}^{K-1}(N-1)^{k}\binom{K-1}{k} = N^{K-1} \tag{40}$$

We next compute the download cost and show that the achieved download cost is optimal, i.e., $D = \frac{L}{C}$. The k-th block of $Q(\mathrm{DB},\theta)$ contains $v(\mathrm{DB},k)$ instances of k-sums of each possible type, and there are $\binom{K}{k}$ possible types of k-sums. Therfore, the cardinality of $Q(\mathrm{DB},\theta)$ is $\sum_{k=1}^{K} v(\mathrm{DB},k)\binom{K}{k}$. Summing over all databases, we have

$$\begin{aligned}
D &= \sum_{\mathrm{DB}=1}^{N}\sum_{k=1}^{K} v(\mathrm{DB},k)\binom{K}{k} \\
&\overset{(29)}{=} \sum_{k=1}^{K}\Big(v(1,k)+(N-1)v(2,k)\Big)\binom{K}{k} \\
&\overset{(36)}{=} \sum_{k=1}^{K}(N-1)^{k-1}\binom{K}{k} \\
&= \sum_{k=1}^{K-1}(N-1)^{k-1}\binom{K}{k} + (N-1)^{K-1} \\
&= \sum_{k=1}^{K-1}(N-1)^{k-1}\left[\binom{K-1}{k-1}+\binom{K-1}{k}\right]+(N-1)^{K-1} \\
&= \sum_{k=1}^{K}(N-1)^{k-1}\binom{K-1}{k-1} + \sum_{k=1}^{K-1}(N-1)^{k-1}\binom{K-1}{k} \\
&\overset{(40)}{=} N^{K-1} + \sum_{k=1}^{K-1}(N-1)^{k-1}\binom{K-1}{k} \\
&= L + \frac{1}{N-1}\sum_{k=1}^{K-1}(N-1)^{k}\binom{K-1}{k} \\
&= L + \frac{1}{N-1}\left[\sum_{k=0}^{K-1}(N-1)^{k}\binom{K-1}{k} - 1\right] \\
&= L + \frac{1}{N-1}(N^{K-1}-1) = L + N^{K-1}\left(\frac{\frac{1}{N}-\frac{1}{N^K}}{1-\frac{1}{N}}\right) \\
&= L\left(\frac{1-\frac{1}{N^K}}{1-\frac{1}{N}}\right) = \frac{L}{C}
\end{aligned}$$

■

Lemma 3: The PIR scheme constructed through the **Q-Gen** Algorithm is private, i.e., it satisfies (4).

Proof: From Corollary 1, we know that $Q(\mathrm{DB},\theta)$ depends on θ only through the permutation functions $\lambda_{\mathrm{DB},\theta,k}(U_k)$, for each $k \in [1:K]$. But, U_k are uniform permutations of message symbols, $U_k = \gamma_k(W_k)$. Because any permutation of a uniform permutation is also uniform,

$$\lambda_{\mathrm{DB},\theta,k}(\gamma_k(W_k))) \sim \gamma_k(W_k). \tag{41}$$

Furthermore, because $\gamma_1, \gamma_2, \cdots, \gamma_j$ are independent,

$$\begin{aligned}
&(\lambda_{\mathrm{DB},\theta,1}(\gamma_1(W_1))), \lambda_{\mathrm{DB},\theta,2}(\gamma_2(W_2))), \cdots, \lambda_{\mathrm{DB},\theta,K}(\gamma_K(W_K)))) \\
&\qquad \sim (\gamma_1(W_1), \gamma_2(W_2), \cdots, \gamma_K(W_K))
\end{aligned}$$

Since $Q(\mathrm{DB},\theta)$ is a function of $(\lambda_{\mathrm{DB},\theta,1}(\gamma_1(W_1))), \lambda_{\mathrm{DB},\theta,2}(\gamma_2(W_2))), \cdots, \lambda_{\mathrm{DB},\theta,K}(\gamma_K(W_K))))$, which is identically distributed for all $\theta \in [1:K]$, $Q(\mathrm{DB},\theta)$ is also identically distributed for all $\theta \in [1:K]$. Thus condition (4) is satisfied and the scheme is private.

■

Remark: The query size of the PIR scheme to database DB is equal to the size of the description for the permutation in $\lambda_{DB,\theta,k}(\gamma_k(W_k))$. Note that we do not attempt to optimize the upload cost in this paper, which is possible, as shown in [20].

Remark: In all PIR schemes presented in this paper, we assume that each database stores all K messages. So the total storage required across all N databases is equal to KN times the message size.

VI. Proof of Theorem 1: Achievability for Arbitrary L

The optimal PIR scheme is a combination (analogous to time sharing arguments in channel capacity studies) of the capacity achieving scheme with message size $L = N^{K-1}$ that was presented in the previous section, and a PIR scheme from [11] (see the remark on replicated storage above Section V of [11]) which is related to blind interference alignment as noted in [21] (see the discussion section of [21]). Since the main objective of [11] is PIR with distributed storage, the scheme that we need is recovered as an implicit special case of [11] (when replication coding is used across the databases). To make the scheme explicit, we restate this result in the following theorem.

Theorem 3 [11]: For PIR with $N \geq 2$ databases, each storing $K \in \mathbb{N}$ messages, each message comprised of $L = N-1$ symbols from M-ary alphabet, $M \in \mathbb{N}/\{1\}$, where the downloads are comprised of symbols from the same M-ary alphabet, the download cost $D = N = L+1$ M-ary symbols is achievable.

While the scheme is implicitly contained in [11], for the sake of completeness we give an explicit proof of Theorem 3 in Section VI-E. We also note that the binary alphabet ($M = 2$) case is considered recently in [17] (see [17, Construction 1]).

A. Examples

To convey the main ideas let us start with some examples for small values of K, N, L. The idea of constructing the optimal achievable scheme is to greedily use the most efficient PIR scheme (the capacity achieving scheme) repeatedly, and when the number of remaining symbols per message is less than required, we turn to the next most efficient scheme (the scheme in Theorem 3), and continue to use the scheme in Theorem 3 with possibly smaller and smaller message sizes until all symbols are considered.

1) $K = 2$ Messages, $N = 2$ Databases, $L = 3$ Symbols per Message: We show that the download cost $D = \lceil \frac{L}{C} \rceil = \lceil 3/(2/3) \rceil = 5$ symbols is achievable. The scheme is as follows. For each message, divide the $L = 3$ message symbols into two groups, where the first group is comprised of 2 symbols and the second group is comprised of 1 symbol. For the first group, we use the capacity achieving scheme with message length $N^{K-1} = 2$ so that the download cost achieved is $2/C = 3$ symbols. For the second group, we use the scheme described in Theorem 3 so that the download cost achieved is $N = 2$ symbols. Adding the two, the overall download cost is $D = 5$ symbols, as desired.

2) $K = 3$ Messages, $N = 3$ Databases, $L = 25$ Symbols per Message: We show that the download cost $D = \lceil \frac{L}{C} \rceil = \lceil 25/(9/13) \rceil = 37$ symbols is achievable. The scheme is as follows. For each message, divide the $L = 25$ symbols into three groups, where the first group is comprised of 18 symbols, the second group is comprised of 6 symbols and the third group is comprised of 1 symbol. For the first group, we further divide the 18 symbols to 2 sub-groups, each of which is comprised of 9 symbols. For each sub-group, we use the capacity achieving scheme with message length $N^{K-1} = 9$ so that the download cost achieved per sub-group is $9/C = 13$ symbols. In total, the download cost for the first group is 26 symbols. Note that the second group only has 6 symbols per message so that we can not use the capacity achieving scheme and we turn to the scheme in Theorem 3. For the second group, we further divide the 6 symbols to 3 sub-groups, each of which is comprised of 2 symbols. For each sub-group, we use the scheme described in Theorem 3 with $N = 3$ databases, so that the download cost per sub-group is $N = 3$ symbols. In total, the download cost for the second group is 9 symbols. Note now that the third group only has 1 symbol per message so that even the scheme for the second group does not apply and we turn to the same class of scheme but with shorter (matching) message length. For the third group, we use the scheme described in Theorem 3 with $N' = 2$ databases (say, the first two databases) and message size $L' = 1$ symbol (matching the size of the third group), so that the download cost achieved is $N' = 2$ symbols. Adding the download cost of the three groups up, the overall download cost is $D = 26 + 9 + 2 = 37$ symbols, as desired.

B. Description of Achievable Scheme for Arbitrary L

We now describe the general achievable scheme for arbitrary L, following the examples presented above. We first fully use the capacity achieving scheme with message size N^{K-1}. To this end, we view each N^{K-1} symbols as a group and proceed until the number of symbols that remain is smaller than N^{K-1},

$$L = G_1 N^{K-1} + L_1 \tag{42}$$

where $G_1 = \lfloor \frac{L}{N^{K-1}} \rfloor$ and $0 \leq L_1 \leq N^{K-1} - 1$. If $L_1 = 0$, we are done. Otherwise, for the L_1 symbols that remain, we fully use the scheme in Theorem 3 with N databases and message size $N - 1$. We view each $N - 1$ symbols as a group and proceed until the number of symbols left is smaller than $N - 1$,

$$L_1 = G_2(N-1) + L_2 \tag{43}$$

where $G_2 = \lfloor \frac{L_1}{N-1} \rfloor$ and $0 \leq L_2 \leq N - 2$. If $L_2 = 0$, we are done. Otherwise, for the $L_2 \geq 1$ symbols that are left, we use the scheme in Theorem 3 with $L_2 + 1$ databases (say, the first $L_2 + 1 \leq N - 1$ databases) and message size L_2. Therefore the message size and the achievable download cost are

$$L = G_1\ N^{K-1} + G_2(N-1) + L_2$$
$$D = \begin{cases} G_1 N^{K-1}/C + G_2 N & \text{if } L_2 = 0 \\ G_1 N^{K-1}/C + G_2 N + L_2 + 1 & \text{otherwise} \end{cases}$$

This completes the description of our achievable scheme.

Remark: Our achievability proofs have used three types of schemes, i.e., the capacity achieving scheme for message size N^{K-1}, the scheme in Theorem 3 with message size $N-1$, and the scheme in Theorem 3 with message size L_2. The upload cost (query size) of our achievable scheme is equal to the sum of the upload cost of all three types of schemes. Note that when we use the same type of scheme several times, we can reuse the same query such that the upload cost does not scale (see [2, Proposition 4.1.1]).

C. Proof That the Scheme Is Correct and Private

Since we construct our PIR scheme as a concatenation of multiple PIR schemes, let us present the following theorem to show that such a concatenation yields a PIR scheme that is correct and private.

Theorem 4: For PIR with $N \in \mathbb{N}$ databases, each storing all $K \in \mathbb{N}$ messages, each message comprised of $L \in \mathbb{N}$ symbols from M-ary alphabet, $M \in \mathbb{N}/\{1\}$, where the downloads are comprised of symbols from the same M-ary alphabet, if there are $J \in \mathbb{N}$ schemes with message length $L_j, j \in [1 : J]$ and download cost $D_j, j \in [1 : J]$, respectively, and the message lengths add up to L, i.e., $\sum_{j=1}^{J} L_j = L$, then there exists a PIR scheme with message length L and download cost $D = \sum_{j=1}^{J} D_j$.

Proof: The scheme is based on dividing the L message symbols to J groups so that the j-th group is comprised of L_j symbols per message. Then we use the given scheme with message length L_j for the j-th group, so that the download cost achieved is D_j symbols. Specifically, the queries for each group are generated independently, given the same desired message index. Combining the download cost for all J groups, we achieve the desired download cost. We are left to prove that this symbol sharing scheme produces a correct and private PIR scheme.

Correctness is easy to see as the scheme for each group is correct. Privacy is proved as follows. Consider any database $n, n \in [1:N]$ and any desired message index $\theta, \theta \in [1:K]$. Denote the query of the scheme for the j-th group as $Q_n^{[\theta]}(j)$. Since the scheme for the j-th group is private, we have that $Q_n^{[\theta]}(j) \sim Q_n^{[\theta']}(j)$, for all $\theta, \theta' \in [1:K]$ and $\forall j \in [1:J]$. Now since for any θ, the queries for each group are generated independently, their joint probability distribution function is the product of the marginal probability distribution functions, i.e.,

$$\begin{aligned}&\Pr(Q_n^{[\theta]}(1), Q_n^{[\theta]}(2), \cdots, Q_n^{[\theta]}(J))\\ &\quad = \Pr(Q_n^{[\theta]}(1)) \times \Pr(Q_n^{[\theta]}(2)) \times \cdots \times \Pr(Q_n^{[\theta]}(J))\\ &\quad = \Pr(Q_n^{[\theta']}(1)) \times \Pr(Q_n^{[\theta']}(2)) \times \cdots \times \Pr(Q_n^{[\theta']}(J))\end{aligned}$$

for all $\theta, \theta' \in [1:K]$. Therefore the overall query for all groups is identically distributed regardless of the index of the desired message θ, and the symbol sharing scheme is private (4). ∎

D. Proof That the Achieved Download Cost $D = \lceil \frac{L}{C} \rceil$

We next show that the achievable download cost in (44) satisfies $D \in [\frac{L}{C}, \frac{L}{C} + 1)$ so that $D = \lceil \frac{L}{C} \rceil$. Note that in the converse proof, we have already shown that for all PIR schemes, $D \geq \frac{L}{C}$ holds. So we only need to prove that $D < \frac{L}{C} + 1$. Here we have two cases.

Case 1: $L_2 = 0$. We have

$$\begin{aligned}D &< \frac{L}{C} + 1 && (44)\\ &\Leftrightarrow G_1 N^{K-1}/C + G_2 N\\ &< (G_1 N^{K-1} + G_2(N-1))/C + 1 && (45)\\ &\Leftrightarrow G_2 N < G_2(N-1)/C + 1 && (46)\end{aligned}$$

When $N = 1$, we have $G_2 = 0$ so that (46) holds. When $N \geq 2$, plugging in $C = \frac{1-1/N}{1-(1/N)^K} = N^{K-1}\left(\frac{N-1}{N^K-1}\right)$, we have

$$\begin{aligned}G_2 N &< G_2\left(\frac{N^K - 1}{N^{K-1}}\right) + 1 && (47)\\ &\Leftrightarrow G_2 < N^{K-1} && (48)\end{aligned}$$

which holds because $G_2 = \lfloor \frac{L_1}{N-1} \rfloor \leq L_1 \leq N^{K-1} - 1 < N^{K-1}$.

Case 2: $L_2 \geq 1$. Note that when $L_2 \geq 1$, we have $N \geq 2$ such that $C = \frac{1-1/N}{1-(1/N)^K}$. As a result,

$$\begin{aligned}D &< \frac{L}{C} + 1 && (49)\\ &\Leftrightarrow G_1 N^{K-1}/C + G_2 N + L_2 + 1\\ &< (G_1 N^{K-1} + G_2(N-1) + L_2)/C + 1 && (50)\\ &\Leftrightarrow G_2 N + L_2 < (G_2(N-1) + L_2)/C && (51)\\ &\Leftrightarrow G_2 N + L_2 < (G_2(N-1) + L_2)\left(\frac{N^K - 1}{(N-1)N^{K-1}}\right)\\ &\Leftrightarrow \frac{G_2}{N^{K-1}} < L_2\left(\frac{N^{K-1} - 1}{(N-1)N^{K-1}}\right) && (52)\\ &\Leftrightarrow G_2(N-1) < L_2(N^{K-1} - 1) && (53)\end{aligned}$$

which is proved as follows

$$\begin{aligned}L_2(N^{K-1} - 1) &\geq N^{K-1} - 1 && (54)\\ &\geq L_1 = G_2(N-1) + L_2 > G_2(N-1) && (55)\end{aligned}$$

Thus the proof is complete.

E. Proof of Theorem 3

We now present the scheme with download cost $D = N$ and message length $L = N - 1$. Consider

$$W_k = \Big(W_k(1), W_k(2), \cdots, W_k(N-1)\Big), \quad \forall k \in [1:K] \quad (56)$$

where each $W_k(i), i \in [1:N-1]$ is an M-ary symbol.

The queries are specified as follows. To retrieve W_θ privately, the user first generates a random vector of length $(N-1)K$, $[h_1(1), \cdots, h_1(N-1), \cdots, h_\theta(1), \cdots, h_\theta(N-1), \cdots, h_K(N-1)]$, where each element is uniformly distributed over $\{0, 1\}$. Then the queries are set as follows.

$$\begin{aligned}Q_1^{[\theta]} &= [h_1(1), \cdots, h_\theta(1), \cdots, h_\theta(N-1), \cdots, h_K(N-1)]\\ Q_2^{[\theta]} &= [h_1(1), \cdots, h_\theta(1) \oplus 1, \cdots, h_\theta(N-1), \cdots, h_K(N-1)]\\ &\cdots\\ Q_N^{[\theta]} &= [h_1(1), \cdots, h_\theta(1), \cdots, h_\theta(N-1) \oplus 1, \cdots, h_K(N-1)]\end{aligned}$$

where $\oplus$ represents the modulo-2 sum. The answer from each database is the modulo-M sum of the scalar product of each message symbol and the corresponding coefficient in the query vector.

$$\begin{aligned}A_1^{[\theta]} &= \sum_{k=1}^{K}\sum_{i=1}^{N-1} h_k(i) W_k(i)\\ A_2^{[\theta]} &= \sum_{k=1}^{K}\sum_{i=1}^{N-1} h_k(i) W_k(i) + (-1)^{h_\theta(1)} W_\theta(1)\\ &\cdots\\ A_N^{[\theta]} &= \sum_{k=1}^{K}\sum_{i=1}^{N-1} h_k(i) W_k(i) + (-1)^{h_\theta(N-1)} W_\theta(N-1)\end{aligned}$$

The user decodes $W_\theta(i), i \in [1:N-1]$ by subtracting $A_1^{[\theta]}$ from $A_{i+1}^{[\theta]}$, with no error. Therefore, the PIR scheme is correct.

Privacy is guaranteed because each query is independent of the desired message index θ. This is because regardless of the desired message index θ, each query $Q_n^{[\theta]}, \forall n$ is individually comprised of elements that are i.i.d. uniform over $\{0, 1\}$.

Each answer is comprised of 1 symbol, so the download cost achieved is $D = N$ symbols. The proof is complete.

VII. Proof of Theorem 2

A. Converse

First let us prove the converse. As in the converse proof of Theorem 1, the PIR capacity [20] provides a general upper bound on rate, and therefore a general lower bound on download cost for any given message length, which holds regardless of the choice of alphabet used to represent the

messages and download symbols. For message length L and download cost D, the rate is $\frac{L\log_2(M)}{D\log_2(M')}$ which cannot exceed capacity. Therefore we automatically have the lower bound on download cost as $D \geq \frac{L\log_2(M)}{C\log_2(M')}$, and because $D \in \mathbb{N}$, we must have

$$D \geq \left\lceil \frac{L\log_2(M)}{C\log_2(M')} \right\rceil \tag{57}$$

B. Achievability

For the proof of achievability, let us construct a simple (sub-optimal) PIR scheme whose download cost is nonetheless guaranteed to be within 2 M'-ary symbols of the lower bound. The scheme is described as follows.

Let us map the messages from M-ary alphabet to M'-ary alphabet. Each message is comprised of L symbols that are from an M-ary message alphabet, i.e., for each message there are M^L possible distinct realizations. L' symbols from M'-ary alphabet are capable of representing $M'^{L'}$ distinct realizations. To have distinct representations for distinct message realizations, we must have $M'^{L'} \geq M^L$. For this, $L' = \lceil L\log_{M'} M\rceil$ is sufficient.[5] Now the message symbols and the download symbols are from the same M'-ary alphabet, so that we can use the PIR scheme used to establish achievability in Theorem 1 to achieve download cost $D = \lceil \frac{L'}{C}\rceil$, measured in M'-ary download symbols. Next let us prove that even for this simple scheme, the gap to optimality is no more than 2 M'-ary symbols.

Since the $N = 1$ case is trivial (optimal to fully download all messages), let us assume $N \geq 2$. Note that for $N \geq 2$ it is always true that $C \geq 1/2$, i.e., $1/C \leq 2$. Starting with the general upper bound (57),

$$\left\lceil \frac{L'}{C} \right\rceil \geq D_L \geq \left\lceil \frac{L\log_2(M)}{C\log_2(M')} \right\rceil \tag{58}$$

$$= \left\lceil \frac{L\log_{M'}(M)}{C} \right\rceil \tag{59}$$

$$\geq \left\lceil \frac{\lceil L\log_{M'}(M)\rceil - 1}{C} \right\rceil \tag{60}$$

$$= \left\lceil \frac{L'}{C} - \frac{1}{C} \right\rceil \tag{61}$$

$$\geq \left\lceil \frac{L'}{C} - 2 \right\rceil \tag{62}$$

$$= \left\lceil \frac{L'}{C} \right\rceil - 2 \tag{63}$$

VIII. Conclusion

Recent work has characterized the capacity, C (supremum of the ratio of message size over download cost, i.e., L/D) of PIR when the message size $L \to \infty$. In this work, we have shown that for arbitrary fixed message size $L \in \mathbb{N}$, when the messages and downloads are comprised of symbols from the same arbitrary M-ary alphabet, the optimal download cost is $D_L = \lceil \frac{L}{C}\rceil$; and when the messages and downloads are comprised of symbols from different alphabets (messages from M-ary alphabet and downloads from M'-ary alphabet, $M \neq M'$), the optimal download cost (in M'-ary symbols) $D_L \in \left\{\left\lceil \frac{L'}{C}\right\rceil, \left\lceil \frac{L'}{C}\right\rceil - 1, \left\lceil \frac{L'}{C}\right\rceil - 2\right\}$, where $L' = \lceil L\log_{M'} M\rceil$.

An interesting feature of our PIR scheme is that it allows arbitrary M-ary alphabet (not restricted to finite fields). This is because the scheme downloads only direct sums modulo-M of various message symbols. As the next step in this direction the extension to TPIR (PIR with T-privacy) may be of interest. The capacity of TPIR for unconstrained alphabet is characterized in [22], and the capacity achieving scheme presented there relies on finite field operations (multiplications) and existence of MDS codes. PIR schemes based on finite fields can be extended to arbitrary M-ary alphabet by decomposing M into its prime factors and concatenating PIR schemes over the finite fields corresponding to the prime factors. However, the extension may be difficult when field size constraints imposed by arbitrary M-ary alphabet are incompatible with the MDS code requirements.

[5]The sub-optimality of the scheme becomes obvious here because, for example if M' is much larger than M, then we could jointly code all K M-ary messages symbols to only 1 M'-ary message symbol, therefore download cost of 1 M'-ary symbol would be enough, whereas our naive scheme will download at least $1/C$ symbols.

References

[1] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, "Private information retrieval," in *Proc. 36th Annu. Symp. Found. Comput. Sci.*, 1995, pp. 41–50.

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

[3] A. Ambainis, "Upper bound on the communication complexity of private information retrieval," in *Automata, Languages and Programming*. London, U.K.: Springer-Verlag, 1997, pp. 401–407.

[4] A. Beimel, Y. Ishai, and E. Kushilevitz, "General constructions for information-theoretic private information retrieval," *J. Comput. Syst. Sci.*, vol. 71, no. 2, pp. 213–247, 2005.

[5] A. Beimel, Y. Ishai, E. Kushilevitz, and J.-F. Raymond, "Breaking the $O(n^{1/(2k-1)})$ barrier for information-theoretic private information retrieval," in *Proc. 43rd Annu. IEEE Symp. Found. Comput. Sci.*, Nov. 2002, pp. 261–270.

[6] S. Yekhanin, "Locally decodable codes and private information retrieval schemes," Ph.D. dissertation, Dept. Elect. Eng. Comput. Sci., Massachusetts Inst. Technol., Cambridge, MA, USA, 2007.

[7] Z. Dvir and S. Gopi, "2-Server PIR with sub-polynomial communication," in *Proc. 47th Annu. ACM Symp. Theory Comput. (STOC)*, 2015, pp. 577–584.

[8] A. Beimel, Y. Ishai, and T. Malkin, "Reducing the servers computation in private information retrieval: PIR with preprocessing," in *Advances in Cryptology—CRYPTO*. Berlin, Germany: Springer-Verlag, 2000, pp. 55–73.

[9] Y. Gertner, S. Goldwasser, and T. Malkin, "A random server model for private information retrieval," in *Randomization and Approximation Techniques in Computer Science*. Berlin, Germany: Springer-Verlag, 1998, pp. 200–217.

[10] G. Di-Crescenzo, Y. Ishai, and R. Ostrovsky, "Universal service-providers for database private information retrieval," in *Proc. 17th Annu. ACM Symp. Principles Distrib. Comput.*, 1998, pp. 91–100.

[11] N. Shah, K. Rashmi, and K. Ramchandran, "One extra bit of download ensures perfectly private information retrieval," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Mar. 2014, pp. 856–860.

[12] T. H. Chan, S.-W. Ho, and H. Yamamoto, "Private information retrieval for coded storage," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Nov. 2015, pp. 2842–2846.

[13] A. Fazeli, A. Vardy, and E. Yaakobi, "Codes for distributed PIR with low storage overhead," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Nov. 2015, pp. 2852–2856.

[14] R. Tajeddine and S. E. Rouayheb. (Feb. 2016). "Private information retrieval from MDS coded data in distributed storage systems." [Online]. Available: https://arxiv.org/abs/1602.01458

[15] S. Rao and A. Vardy. (May 2016). "Lower bound on the redundancy of PIR codes." [Online]. Available: https://arxiv.org/abs/1605.01869
[16] S. Blackburn and T. Etzion. (Jul. 2016). "PIR array codes with optimal PIR rate." [Online]. Available: https://arxiv.org/abs/1607.00235
[17] T. E. R. Simon Blackburn and M. B. Paterson. (Sep. 2016). "PIR schemes with small download complexity and low storage requirements." [Online]. Available: https://arxiv.org/abs/1609.07027
[18] K. Banawan and S. Ulukus. (Sep. 2016). "The capacity of private information retrieval from coded databases." [Online]. Available: https://arxiv.org/abs/1609.08138
[19] Y. Zhang, X. Wang, H. Wei, and G. Ge. (Sep. 2016). "On private information retrieval array codes." [Online]. Available: https://arxiv.org/abs/1609.09167
[20] H. Sun and S. A. Jafar. (Feb. 2016). "The capacity of private information retrieval." [Online]. Available: https://arxiv.org/abs/1602.09134
[21] H. Sun and S. A. Jafar. (Jan. 2016). "Blind interference alignment for private information retrieval." [Online]. Available: https://arxiv.org/abs/1601.07885
[22] H. Sun and S. A. Jafar. (May 2016). "The capacity of robust private information retrieval with colluding databases." [Online]. Available: https://arxiv.org/abs/1605.00635

Hua Sun (S'12) received the B.E. degree in communications engineering from Beijing University of Posts and Telecommunications, Beijing, China, in 2011, and the M.S. degree in electrical and computer engineering from the University of California at Irvine, Irvine, in 2013, where he is currently pursuing the Ph.D. degree. His research interests include information theory and its applications to communications, networking, privacy, and storage.

Mr. Sun received the IEEE Jack Keil Wolf ISIT Student Paper Award in 2016, an IEEE GLOBECOM Best Paper Award in 2016, and the University of California Irvine CPCC Fellowship for the year 2011–2012.

Syed Ali Jafar (S'99–M'04–SM'09–F'14) received the B.Tech. degree from IIT Delhi, Delhi, India, in 1997, the M.S. degree from Caltech, USA, in 1999, and the Ph.D. degree from Stanford, USA, in 2003, all in electrical engineering.

His industry experience includes positions at Lucent Bell Labs, Qualcomm Inc., and Hughes Software Systems. He is currently a Professor with the Department of Electrical Engineering and Computer Science, University of California at Irvine, Irvine, CA, USA. His research interests include multiuser information theory, wireless communications, and network coding.

Dr. Jafar is a recipient of the New York Academy of Sciences Blavatnik National Laureate in physical sciences and engineering, the NSF CAREER Award, the ONR Young Investigator Award, the UCI Academic Senate Distinguished Mid-Career Faculty Award for Research, the School of Engineering Mid-Career Excellence in Research Award, the School of Engineering Maseeh Outstanding Research Award, the IEEE Information Theory Society Best Paper Award, the IEEE Communications Society Best Tutorial Paper Award, the IEEE Communications Society Heinrich Hertz Award, and three IEEE GLOBECOM Best Paper Awards. His student co-authors received the IEEE Signal Processing Society Young Author Best Paper Award, and the Jack Wolf ISIT Best Student Paper Award. He received the UC Irvine EECS Professor of the Year Award six times, in 2006, 2009, 2011, 2012, 2014, and 2017 from the Engineering Students Council and the Teaching Excellence Award in 2012 from the School of Engineering. He was a University of Canterbury Erskine Fellow in 2010 and an IEEE Communications Society Distinguished Lecturer for 2013–2014. He was recognized as a Thomson Reuters Highly Cited Researcher and included by Sciencewatch among The World's Most Influential Scientific Minds in 2014–2016. He served as an Associate Editor of the IEEE TRANSACTIONS ON COMMUNICATIONS from 2004 to 2009, the IEEE COMMUNICATIONS LETTERS from 2008 to 2009, and the IEEE TRANSACTIONS ON INFORMATION THEORY from 2009 to 2012.