Uncoded Placement with Linear Sub-Messages for Private Information Retrieval from Storage Constrained Databases

Nicholas Woolsey, *Student Member, IEEE,* Rong-Rong Chen, *Member, IEEE,* and Mingyue Ji, *Member, IEEE*

***Abstract*—We propose capacity-achieving schemes for private information retrieval (PIR) from uncoded databases (DBs) with both homogeneous and heterogeneous storage constraints. In the PIR setting, a user queries a set of DBs to privately download a message, where privacy implies that no one DB can infer which message the user desires. In general, a PIR scheme is comprised of storage placement and delivery designs. Previous works have derived the capacity, or infimum download cost, of PIR with uncoded storage placement and sufficient conditions of storage placement to meet capacity. However, the currently proposed storage placement designs require splitting each message into an exponential number of sub-messages with respect to the number of DBs. In this work, when DBs have the same storage constraint, we propose two simple storage placement designs that satisfy the capacity conditions. Then, for more general heterogeneous storage constraints, we translate the storage placement design process into a "filling problem". We design an iterative algorithm to solve the filling problem where, in each iteration, messages are partitioned into sub-messages and stored at subsets of DBs. All of our proposed storage placement designs require a number of sub-messages per message at most equal to the number of DBs.**

***Index Terms*—Private information retrieval (PIR), storage-constrained databases, heterogeneous storage sizes**

I. Introduction

The private information retrieval (PIR) problem was originally introduced by Chor *et al.* [3], [4] and the fundamental-theoretic limits of PIR were recently characterized in [5]. In the PIR problem, a user privately downloads one of K messages from a set of N non-colluding databases (DBs). Privacy implies that no DB can infer which of the K messages the user is downloading. To achieve privacy the user generates strategic queries to the DBs such that sub-messages from all K messages are requested. To gauge the performance of the PIR scheme, the rate, R, is defined as the ratio of desired bits (or size of each message), L, to the total number of downloaded bits, D. In the traditional setting of full storage PIR (FS-PIR), each DB has access to all K messages and the capacity, or maximum achievable rate, of PIR is $\left(1+\frac{1}{N}+\frac{1}{N^2}+\cdots+\frac{1}{N^{K-1}}\right)^{-1}$ [5]. Multiple achievable schemes have been developed which achieve FS-PIR capacity by exploiting downloaded undesired sub-messages for coding opportunities [5]–[7].

This paper includes and expands content from our conference papers [1], [2].

The authors are with the Department of Electrical and Computer Engineering at University of Utah, Salt Lake City, UT 84112, USA. (e-mail: nicholas.woolsey@utah.edu, rchen@ece.utah.edu and mingyue.ji@utah.edu).

More recently, the problem of homogeneous storage constrained PIR (SC-PIR) was proposed such that each DB can only store $\mu_0 KL$ bits where $\frac{1}{N} \leq \mu_0 \leq 1$ [8]. Define $t=\mu_0 N$, the capacity of homogeneous SC-PIR was shown to be $\left(1+\frac{1}{t}+\frac{1}{t^2}+\cdots+\frac{1}{t^{K-1}}\right)^{-1}$ for $t=1,2.\ldots,N$ [9], [10]. Different from FS-PIR, there is an additional design aspect to SC-PIR which is the storage placement must be strategically designed. For example, the original homogeneous SC-PIR scheme met capacity [8] by using the storage placement scheme of the classical shared link coded caching problem [11]. One of the limitations of this scheme is the storage placement requires that each message is split into $O(\exp N)$ sub-messages. Hence, the proposed PIR scheme of [8] can be impractical for a large number of databases. This achievable scheme was generalized to the decentralized storage placement in [9]. In addition, linear coded storage placement at the databases has been analyzed in [12] and [13]. Furthermore, Tian *et al.* [14] used Shannon theoretic approach to analyze the SC-PIR problem for the canonical case of $K=2$ and $N=2$ and proposed the optimal linear scheme. More interestingly, they also showed that non-linear scheme can use less storage than the optimal linear scheme.

The SC-PIR problem was generalized by Banawan *et al.* in [15] to study the case where DBs have heterogeneous storage requirements. In this setting, the storage capacity of the N databases are defined by a vector[1] $\boldsymbol{\mu} \in \mathbb{R}_+^N$, such that DB n can only store up to $\mu[n]KL$ bits and $0 \leq \mu[n] \leq 1$. Surprisingly, the authors in [15] showed that the capacity of heterogeneous SC-PIR is the same as homogeneous SC-PIR where $t = \sum_{n=1}^{N} \mu[n]$. Furthermore, the authors translated the storage placement problem into a linear program (LP). A relaxed version of the LP demonstrated that, to achieve capacity, sub-libraries (sub-message sets) should be stored at t DBs (or $\lfloor t \rfloor$ and $\lceil t \rceil$ DBs for non-integer t) which was also shown in [1] for the homogeneous case. The authors of [15] also showed the existence of a solution to the LP for general N. However, an explicit placement solution was only derived for $N=3$ DBs. For general N, the LP has $O(\exp N)$ variables, representing the potential sub-messages. Hence, this scheme has high complexity for large N.

In this paper, we propose capacity-achieving SC-PIR storage placement schemes with at most N sub-messages per message. The schemes operate under general homogeneous and

[1] $\mathbb{R}_+^N$ denotes the set of non-negative real-valued vectors in N-dimensional space

heterogeneous DB storage requirements when t is an integer. Inspired by previous works [8], [15], we formally introduce a design framework which utilizes previously developed FS-PIR schemes for delivery and design new storage placement schemes. For homogeneous SC-PIR, we abandon the idea of using classical shared link coded caching approaches and propose two novel combinatorial schemes. Based on the sufficient conditions to achieve capacity in heterogeneous SC-PIR problem, we translate the storage placement to a *filling problem (FP)*. Instead of deriving an explicit LP solution, we propose an iterative algorithm which places a sub-library at t DBs in each iteration. Finally, we propose a method to convert a non-integer t storage placement problem into two integer t storage placement problems that can be solved using the proposed placement schemes for integer t.

Our main contributions in this paper are as follows:

1) We propose two capacity achieving storage placement schemes for homogeneous SC-PIR which require at most N sub-messages per message when t is an integer.
2) We propose a capacity achieving iterative algorithm to solve the heterogeneous SC-PIR storage placement problem with integer t that converges in N iterations and yields at most N sub-messages per message.
3) We develop a memory sharing design to expand general integer t SC-PIR placement solutions to the non-integer t scenario with general heterogeneous storage requirements.

The remainder of this paper is organized as follows. In Section II, we describe the problem formulation of SC-PIR. In Section III, we present a design architecture for SC-PIR storage placement schemes. The homogeneous SC-PIR storage designs are presented in Section IV. In Section V, we use the sufficient conditions of SC-PIR capacity to translate the heterogeneous SC-PIR storage placement problem into an equivalent filling problem. In Section VI, we develop an iterative solution to the filling problem and analyze its convergence. In Section VII, we expand our designs for non-integer t. In Section VIII we discuss this work and future directions. Concluding remarks are given in Section IX.

Notation Convention: We use $|\cdot|$ to represent the cardinality of a set or the length of a vector. Also $[n] := 1, 2, \ldots, n$ and $[n_1 : n_2] = n_1, n_1 + 1, \ldots, n_2$. A bold symbol such as $\boldsymbol{a}$ indicates a vector and $a[i]$ denotes the i-th element of $\boldsymbol{a}$. $\mathbb{R}_+^n$ is the set of non-negative reals in n-dimensional space and $\mathbb{Z}^+$ is the set of all positive integers.

II. Problem Formulation and Related Work

There are K independent messages, $W_1, \ldots, W_K$, each of size L bits. In particular,

$$H(W_1, \ldots, W_K) = H(W_1) + \cdots + H(W_K) \tag{1}$$

$$H(W_1) = \cdots = H(W_K) = L. \tag{2}$$

The messages are collectively stored in an uncoded fashion among N non-colluding DBs, labeled as DB1, DB2, ...,DBN. The storage capacity of the DBs are defined by a vector $\boldsymbol{\mu} \in \mathbb{R}_+^N$ where, for all $n \in [N]$, DB n has the storage capacity of $\mu[n]KL$ bits and $0 < \mu[n] \leq 1$. Furthermore, for all $n \in [N]$, define Z_n as the storage contents of DB n such that

$$\forall n \in [N], \;\; H(Z_n) \leq \mu[n]KL. \tag{3}$$

Also, we define $t \triangleq \sum_{n=1}^{N} \mu[n]$ as the number of times each bit of the messages is stored among the DBs. To design an achievable PIR scheme we assume $t \geq 1$ so that each bit of the messages can be stored at least once across the DBs. A user makes a request W_k and sends a query $Q_n^{[k]}$, which is independent of the messages, to each DB $n \in [N]$,

$$\forall k \in [K], \;\; I(W_1, \ldots, W_K; Q_1^{[k]}, \ldots, Q_N^{[k]}) = 0. \tag{4}$$

Each DB $n \in [N]$ sends an answer $A_n^{[k]}$ such that

$$\forall k \in [K], \;\; \forall n \in [N], \;\; H(A_n^{[k]} | Z_n, Q_n^{[k]}) = 0. \tag{5}$$

Furthermore, given the answers from all the databases, the user must be able to recover the requested message and therefore,[2]

$$H(W_k | A_1^{[k]}, \ldots, A_n^{[k]}, Q_1^{[k]}, \ldots, Q_n^{[k]}) = 0. \tag{6}$$

The user generates queries in a manner to ensure privacy such that no DB can infer which message the user desires, *i.e.* for all $n \in [N]$

$$I(k; Q_n^{[k]}, A_n^{[k]}, W_1, \ldots, W_K, Z_1, \ldots, Z_N) = 0. \tag{7}$$

Let D be the total number of downloaded bits

$$D = \sum_{n=1}^{N} H\left(A_n^{[k]}\right). \tag{8}$$

Given $\boldsymbol{\mu}$, we say that a pair (D, L) is achievable if there exists a SC-PIR scheme with rate

$$R \triangleq \frac{L}{D} \tag{9}$$

that satisfies (5)-(7). The SC-PIR capacity is defined as

$$C(\boldsymbol{\mu}) = \sup\{R : (D, L) \text{ is achievable}\}. \tag{10}$$

A. Capacity of SC-PIR

Define $t \triangleq \sum_{n=1}^{N} \mu[n]$ as the cumulative storage of the DBs normalized by KL. The following was shown in [10] for homogeneous and [15] for heterogeneous storage requirements. The capacity of SC-PIR for integer t is

$$C(\boldsymbol{\mu}) = \left(1 + \frac{1}{t} + \frac{1}{t^2} + \cdots + \frac{1}{t^{K-1}}\right)^{-1}, \quad \text{for } t \in [N]. \tag{11}$$

Moreover, assuming storage at the DBs is uncoded, the capacity for non-integer t is

$$C(\boldsymbol{\mu}) = \left((\lceil t \rceil - t) C_{\lfloor t \rfloor}^{-1} + (t - \lfloor t \rfloor) C_{\lceil t \rceil}^{-1}\right)^{-1}, \quad \text{for } t \notin \mathbb{Z}^+ \tag{12}$$

where $1 < t < N$ and $C_{\lfloor t \rfloor}^{-1}$ and $C_{\lceil t \rceil}^{-1}$ are the inverse of SC-PIR capacity with cumulative storage of $\lfloor t \rfloor$ and $\lceil t \rceil$.

[2]In this work, we explore zero-error PIR schemes.

III. SC-PIR General Design Architecture

In this section, we introduce the General Design Architecture (GDA) for SC-PIR which is used throughout this paper. While similar ideas have been previously developed for the design of several SC-PIR schemes [8], [15], an explicit formulation of the GDA for SC-PIR, as described below, will help facilitate a streamlined design framework from an arbitrary FS-PIR design to a derived SC-PIR design. In the GDA, we split the SC-PIR problem into placement and delivery phases. In the placement phase, the storage contents of the DBs is defined. Then, in the delivery phase, FS-PIR schemes are used to privately download sub-messages of the user requested message. The achievable rate of this approach is established.

Definition 1: The **General Design Architecture (GDA)** is the method used to create an SC-PIR scheme with the following placement and delivery phases. It applies to both homogeneous and heterogeneous storage constraints.

- **Placement**: First, define a fractional placement vector $\boldsymbol{\alpha} = (\alpha[1], \alpha[2], \cdots, \alpha[F])$ where $0 < \alpha[f] \leq 1$ for any $f \in [F]$ and $\sum_{f=1}^{F} \alpha[f] = 1$. Given $\boldsymbol{\alpha}$, each message W_k from the library is split into F non-overlapping sub-messages $W_{k,1}, \cdots W_{k,F}$ such that $W_k = \bigcup_{f=1}^{F} W_{k,f}$ where the length of each sub-message $W_{k,f}$ is $\alpha[f]L$. This leads to a set of F sub-libraries $\{\mathcal{M}_1, \cdots, \mathcal{M}_F\}$ where each sub-library $\mathcal{M}_f = \bigcup_{k=1}^{K} W_{k,f}$ includes a fraction $\alpha[f]$ from every file W_k in the library. We then place each of these sub-libraries into a sub-group of databases. The resulting placement scheme is specified by F sub-groups of databases $\mathcal{N}_1 \ldots \mathcal{N}_F$, where each sub-group $\mathcal{N}_f$ consists of all the databases that store sub-library $\mathcal{M}_f$. Note that, depending on its storage constraint, for each $n \in [N]$, DB n can store multiple sub-libraries and its storage is given by $Z_n = \bigcup_{\{f \in [F] \,:\, n \in \mathcal{N}_f\}} \mathcal{M}_f$. Furthermore, since DB n should not be assigned more sub-libraries then its storage constraint $\mu[n]$ allows, the placement scheme must satisfy
$$\sum_{\{f \in [F] \,:\, n \in \mathcal{N}_f\}} \alpha[f] \leq \mu[n]. \tag{13}$$
- **Delivery**: Given that a user requests file W_θ, $\theta \in [K]$, then, for each $f \in [F]$, the user privately downloads sub-message $W_{\theta,f}$ from the sub-group of DBs $\mathcal{N}_f$ using a query defined by any given FS-PIR scheme. ♢

In Appendix A, we prove that the GDA is private. The rate of an SC-PIR scheme defined by the GDA is given in the following proposition.

Proposition 1: The rate of a SC-PIR scheme defined by the GDA is
$$R = \left(\frac{\alpha[1]}{R_1} + \frac{\alpha[2]}{R_2} + \cdots + \frac{\alpha[F]}{R_F} \right)^{-1}, \tag{14}$$
where R_f is the rate of the FS-PIR scheme used to privately download the requested sub-message $W_{\theta,f}$ from the sub-group of DBs $\mathcal{N}_f$. □

Proof: We count the number of downloaded bits. For all $f \in [F]$, $R_f = \frac{\alpha_f L}{D_f}$ where D_f is the number of downloaded bits necessary to privately download $W_{\theta,f}$ of size $\alpha_f L$ bits from the databases in $\mathcal{N}_f$. Therefore, the total number of bits required to privately download the entirety of W_θ is
$$D = D_1 + D_2 + \cdots + D_F = L \left(\frac{\alpha[1]}{R_1} + \frac{\alpha[2]}{R_2} + \cdots + \frac{\alpha[F]}{R_F} \right).$$
Since $R = \frac{L}{D}$, we obtain (14). ■

The placement phase of the GDA requires that each message is split into F sub-messages. Then, in the delivery phase, the employed FS-PIR scheme may require the sub-messages to be split further. The amount of further splitting is defined by the specific FS-PIR scheme (see [5]–[7]). If the FS-PIR scheme to privately download $W_{\theta,f}$ requires the sub-messages $\{W_{k,f} : k \in [K]\}$ to be further split into S_f sub-messages, then the total number of sub-messages per message, S, resulting from the GDA is
$$S = \sum_{f=1}^{F} S_f. \tag{15}$$

Remark 1: This work focuses the design of the placement phase and reducing F, the resulting number of sub-messages per message from the placement phase. Then for delivery, the topic of reducing the sub-messages of FS-PIR has been studied in other works [6], [7]. Our presented SC-PIR designs use FS-PIR schemes found in [5]–[7] for delivery.

IV. Capacity-Achieving Homogeneous SC-PIR Placement Schemes

In this section, we present our new capacity achieving designs, termed Homogeneous Placement 1 (HoP1) and Homogeneous Placement 2 (HoP2). In the homogeneous setting, we assume all DBs have the same storage capacity such that $\mu_0 = \mu[1] = \cdots = \mu[N]$ and $t = \mu_0 N$. The state-of-the-art capacity-achieving homogeneous SC-PIR scheme requires $F = \binom{N}{t}$ sub-messages per message for placement [8] as it uses a placement design from coded caching [11]. Surprisingly, in this work we show that coded caching placement is not necessary to achieve SC-PIR capacity, and we significantly reduce the number of sub-messages with our new designs. The use of HoP1 and HoP2 require $F = \frac{N}{t}$ and $F = N$ sub-messages per message, respectively. HoP1 has a fewer number of sub-messages but requires that $t \in [N]$ and $\frac{N}{t} \in \mathbb{Z}^+$, while HoP2 operates for any $t \in [N]$. Our results are summarized in the following theorem.

Theorem 1: Assume that $t \in \mathbb{Z}^+$ and all DBs have the same storage capacity. Using the GDA with either HoP1 or HoP2 for placement and a capacity achieving FS-PIR scheme for delivery, the following rate is achievable
$$R = \left(1 + \frac{1}{t} + \frac{1}{t^2} + \cdots + \frac{1}{t^{K-1}} \right)^{-1}, \tag{16}$$
which achieves the capacity of SC-PIR. □

We prove Theorem 1 by presenting HoP1 and HoP2 and deriving their rate in Sections IV-A through IV-D. Next, we start with a motivating example and then present our new placement designs.

A. A Homogeneous SC-PIR Example

In this section, we provide an example of a SC-PIR scheme using HoP1. Consider $N = 4$ DBs labeled as DB1 through DB4 and each DB has the storage capacity to store $\mu_0 = \frac{1}{2}$ of the message library containing $K = 3$ messages, denoted by A, B and C. Moreover, $t = \mu_0 N = 2$.

Placement: We split each message into $F = 2$ disjoint, equal-size sub-messages such that $A = \{A_1, A_2\}$, $B = \{B_1, B_2\}$, and $C = \{C_1, C_2\}$. Then, we define two sub-libraries $\mathcal{M}_1 = \{A_1, B_1, C_1\}$ and $\mathcal{M}_2 = \{A_2, B_2, C_2\}$. The sub-library $\mathcal{M}_1$ is stored at DB1 and DB2 and sub-library $\mathcal{M}_2$ is stored at DB3 and DB4. This corresponds to sub-groups of databases $\mathcal{N}_1 = \{1, 2\}$ and $\mathcal{N}_2 = \{3, 4\}$. The storage at each database is given by

$$Z_1 = Z_2 = \mathcal{M}_1 = \{A_1, B_1, C_1\}, \tag{17}$$
$$Z_3 = Z_4 = \mathcal{M}_2 = \{A_2, B_2, C_2\}. \tag{18}$$

Each DB stores exactly half the library and meets its storage capacity.

Delivery: Assuming the user requests message A, we use the FS-PIR scheme of [5] twice: first, to privately download A_1 from DB1 and DB2 and second, to privately download A_2 from DB3 and DB4. To execute the scheme of [5], each sub-message from the placement is further split into 8 disjoint, equal-size sub-messages. For example, $A_1 = \{A_1^1 \dots A_1^8\}$ and $B_2 = \{B_2^1 \dots B_2^8\}$ and the other sub-messages are labeled similarly. The queries to the DBs defined by the FS-PIR scheme of [5] are shown in Table I.

TABLE I
STORAGE CONSTRAINED PIR, $N = 4$, $K = 3$, $\mu_0 = \frac{1}{2}$

DB1	DB2	DB3	DB4
$A_1^5 \quad B_1^8 \quad C_1^6$	$A_1^1 \quad B_1^3 \quad C_1^1$	$A_2^5 \quad B_2^7 \quad C_2^4$	$A_2^2 \quad B_2^6 \quad C_2^2$
$A_1^6 + B_1^3$ $A_1^7 + C_1^1$ $B_1^6 + C_1^5$	$A_1^3 + B_1^8$ $A_1^8 + C_1^6$ $B_1^7 + C_1^3$	$A_2^1 + B_2^6$ $A_2^6 + C_2^2$ $B_2^3 + C_2^6$	$A_2^7 + B_2^7$ $A_2^8 + C_2^4$ $B_2^8 + C_2^7$
$A_1^2 + B_1^7 + C_1^3$	$A_1^4 + B_1^6 + C_1^5$	$A_2^3 + B_2^8 + C_2^7$	$A_2^4 + B_2^3 + C_2^6$

The query to each database maintains privacy since the query is symmetric such that for each sub-message of A that is requested, a sub-message from B and C are also requested. All coded pairs of sub-messages from the 3 messages are requested an equal number of times. Ultimately, the user can decode all sub-messages of message A, because downloaded sub-messages of B and C can be used for decoding (see Table I). We refer the reader to [5] for more details on delivery. In total, there are 28 downloaded sub-messages of size $\frac{L}{16}$ bits which are either coded or uncoded. Therefore, the total number of downloaded bits is $D = \frac{28L}{16} = \frac{7L}{4}$ and the rate is $R = \frac{L}{D} = \frac{16}{28} = \frac{4}{7}$, which achieves capacity defined in (11).

B. General SC-PIR Scheme when $\frac{N}{t} \in \mathbb{Z}^+$

Definition 2: **Homogeneous Placement 1 (HoP1)** Assume a homogeneous storage constraint of $\mu_0 = \frac{t}{N} = \mu[1] = \cdots = \mu[N]$. For the case when both t and $\frac{N}{t}$ are positive integers, we propose HoP1 following the GDA described in Section III. For HoP1, we choose $F = \frac{N}{t}$ and a uniform fractional placement vector $\boldsymbol{\alpha} = \left[\frac{t}{N}, \dots, \frac{t}{N}\right]$. The sub-libraries $\mathcal{M}_1, \dots, \mathcal{M}_F$ are defined according to the GDA. We place each sub-library $\mathcal{M}_f$ in t consecutive databases. This leads to non-overlapping sub-groups of DBs $\mathcal{N}_1, \dots, \mathcal{N}_F$, where $\mathcal{N}_f = \{(f-1)t + 1, \cdots, ft\}$ for each $f \in [F]$. Note that in HoP1, each DB belongs to only one sub-group of DBs, storing exactly one sub-library, and hence meeting the storage constraint of $\mu_0 = \frac{t}{N}$.

Achievable Rate: By using HoP1 with the GDA, SC-PIR capacity is achieved by using a capacity-achieving FS-PIR scheme for delivery. In this case, for all $f \in [F]$, the rate to download $W_{\theta,f}$ from the DBs of $\mathcal{N}_f$ is

$$\begin{aligned} R_f &= \left(1 + \frac{1}{|\mathcal{N}_f|} + \frac{1}{|\mathcal{N}_f|^2} + \cdots + \frac{1}{|\mathcal{N}_f|^{K-1}}\right)^{-1} \\ &= \left(1 + \frac{1}{t} + \frac{1}{t^2} + \cdots + \frac{1}{t^{K-1}}\right)^{-1} \end{aligned} \tag{19}$$

where $|\mathcal{N}_f| = t$ is the number of DBs in sub-group $\mathcal{N}_f$. By (14) of Proposition 1, the rate of the SC-PIR scheme is simply $R = R_f$ which is the capacity of SC-PIR with uncoded storage.

C. A Homogeneous SC-PIR Example when $\frac{N}{t} \notin \mathbb{Z}^+$

In the following, we present a SC-PIR example using HoP2 which allows $\frac{N}{t} \notin \mathbb{Z}^+$ and requires $F = N$ number of sub-messages per message for the placement phase. Consider $N = 5$ DBs, labeled DB1 through DB5, each with the storage capacity to store a $\mu_0 = \frac{3}{5}$ fraction of the $K = 2$ message library containing messages A and B. Moreover, $t = \mu_0 N = 3$.

Placement: Messages A and B are split into $F = 5$ disjoint, equal-size sub-messages labeled as $\{A_1, \dots, A_5\}$ and $\{B_1, \dots, B_5\}$, respectively. Then, for each $f \in [F]$, define the sub-library $\mathcal{M}_f = \{A_f, B_f\}$. A "cyclic" placement is used such that sub-libraries $\mathcal{M}_1$, $\mathcal{M}_2$, $\mathcal{M}_3$, $\mathcal{M}_4$ and $\mathcal{M}_5$ are placed into sub-groups of DBs according to $\mathcal{N}_1 = \{4, 5, 1\}$, $\mathcal{N}_2 = \{5, 1, 2\}$, $\mathcal{N}_3 = \{1, 2, 3\}$, $\mathcal{N}_4 = \{2, 3, 4\}$ and $\mathcal{N}_5 = \{3, 4, 5\}$, respectively. As a result, the storage at DB n is[3]

$$Z_n = \bigcup_{f \in \{[0:2] \oplus_N n\}} \mathcal{M}_f. \tag{20}$$

For instance, DB2 stores sub-libraries in $Z_2 = \{\mathcal{M}_2, \mathcal{M}_3, \mathcal{M}_4\}$ and DB5 stores sub-libraries in $Z_5 = \{\mathcal{M}_5, \mathcal{M}_1, \mathcal{M}_2\}$.

TABLE II
STORAGE CONSTRAINED PIR, $N = 5$, $K = 2$, $\mu = \frac{3}{5}$

DB1 $(1, 2, 3)$	DB2 $(2, 3, 4)$	DB3 $(3, 4, 5)$	DB4 $(4, 5, 1)$	DB5 $(5, 1, 2)$
$A_1^3 \quad B_1^2$	$A_2^3 \quad B_2^2$	$A_3^3 \quad B_3^3$	$A_4^2 \quad B_4^3$	$A_5^2 \quad B_5^1$
$A_2^1 + B_2^2$ $A_3^2 + B_3^3$	$A_3^3 + B_3^3$ $A_4^1 + B_4^3$	$A_4^3 + B_4^3$ $A_5^3 + B_5^1$	$A_5^1 + B_5^1$ $A_1^1 + B_1^2$	$A_1^2 + B_1^2$ $A_2^2 + B_2^2$

[3]To generally apply the cyclic placement approach, we impose the following notation: $a \oplus_N b = (a + b - 1 \mod N) + 1$ and $[a_1 : a_2] \oplus_N b = \{a' \oplus_N b : a' \in [a_1 : a_2]\}$.

Delivery: The delivery phase repeats the FS-PIR scheme of [7] 5 times. To execute the scheme of [7], each sub-message from the placement is further split into 3 disjoint, equal-size sub-messages. For example, $A_1 = \{A_1^1 \dots A_1^3\}$ and $B_2 = \{B_2^1 \dots B_2^3\}$. The queries of a user that privately downloads message A are shown in Table II. The top row of the table contains database labels and the 3-tuple below each database label defines the subscripts of the sub-messages that are locally available to that database. The remaining three rows of the table show the queries of the user.

For instance, to obtain sub-messages of A_1, $\{A_1^j, j \in [3]\}$, the user applies the FS-PIR scheme to DB1, DB4, and DB5. The user obtains A_1^3 from DB1 and can decode A_1^1 from DB4's transmission of $A_1^1 + B_1^2$ because the user receives B_1^2 from DB1. Similarly, the user decodes A_1^2 from DB5's transmission of $A_1^2 + B_1^2$. These transmissions are highlighted in red in Table II. To ensure privacy, the queries are symmetric and no sub-message is requested more than once from any one database. The user downloads 20 sub-messages of size $\frac{L}{15}$ bits therefore $D = 20 \cdot \frac{L}{15} = \frac{4L}{3}$ and $R = \frac{3}{4}$ which meets capacity defined in (11).

D. A General SC-PIR Scheme for arbitrary $t \in [N]$

Definition 3: **Homogeneous Placement 2 (HoP2)** Assume a homogeneous storage constraint of $\mu_0 = \frac{t}{N} = \mu[1] = \cdots = \mu[N]$. For the case when t is an integer, but $\frac{N}{t}$ can be a non-integer, we propose HoP2 following the GDA described in Section III. For HoP2, we choose $F = N$ and a uniform fractional placement vector $\boldsymbol{\alpha} = \left[\frac{1}{N}, \dots, \frac{1}{N}\right]$. The sub-libraries $\mathcal{M}_1, \dots, \mathcal{M}_F$ are defined according to the GDA. We place each sub-library in t DBs in a "cyclic" fashion. This leads to overlapping sub-groups of DBs $\mathcal{N}_1, \dots, \mathcal{N}_F$, where $\mathcal{N}_f = [-(t-1) : 0] \oplus_N f$ for each $f \in [F]$. Note that in HoP2, each DB belongs to exactly t sub-group of DBs, storing t sub-libraries, and hence meeting the storage constraint of $t \cdot \frac{1}{N} = \mu_0$. ◊

Achievable Rate: By using HoP2 with the GDA, SC-PIR capacity is achieved by using a capacity-achieving FS-PIR scheme for delivery. Similar to HoP1, every sub-library is stored at t DBs and based on FS-PIR capacity, a requested sub-message, $W_{\theta,f}$, can be downloaded with rate $R_f = \left(1 + \frac{1}{t} + \frac{1}{t^2} + \cdots + \frac{1}{t^{K-1}}\right)^{-1}$. Therefore, the overall SC-PIR rate is $R = R_f$.

E. Comparison to the State-of-the-Art

The state-of-the-art homogeneous SC-PIR design proposed in [8] achieves the capacity defined in (11) for integer t. The placement strategy of [8] is to place a sub-library at every unique set of t DBs similar to the achievable coded caching scheme of [11]. In other words, the placement requires each message to be split into $F = \binom{N}{t}$ sub-messages, making it impractical for a large number of DBs. Moreover, the state-of-the-art SC-PIR design uses the FS-PIR scheme of [5] for delivery which requires each sub-message be further split into t^K sub-messages. The total number of sub-messages per message is then $S = \binom{N}{t} t^K$. Since the sub-messages are equal-size, the minimum message size is $L = \binom{N}{t} t^K$ bits where each sub-message would be 1 bit.

We have demonstrated the coded caching placement is unnecessary, since our placement designs require each message is split into at most $F = N$ sub-messages. Using the FS-PIR scheme of [5] for delivery and HoP2 for placement, there are $S = Nt^K$ sub-messages per message. Furthermore, [6] derives the minimum message size of FS-PIR to achieve capacity.[4] Using this FS-PIR scheme yields $S = \binom{N}{t}(t-1)$ with the state-of-the-art SC-PIR placement and $S = N(t-1)$ with HoP2. In either case, by using HoP2 we observe an exponential reduction in sub-messages compared to the state-of-the-art SC-PIR scheme. The use of HoP1 or HoP2 in conjunction with the FS-PIR scheme of [6] allows for a quadratic number of sub-messages.

V. Translating Heterogeneous SC-PIR to a Filling Problem

When $t \in \mathbb{Z}^+$, capacity can be achieved for SC-PIR with general heterogeneous requirements by placing sub-libraries at sub-groups of exactly t DBs. In this way, the placement design translates to an equivalent filling problem (FP). While we have developed FP solutions to the homogeneous case, it becomes more challenging to solve the FP for the heterogeneous setting. Intuitively, one way to solve the FP is to iteratively define sub-libraries and place each of them at t DBs. However, we will show that this method is impractical as it can get "stuck" and reach a point where there are less than t "unfilled" DBs, at which point the capacity cannot be achieved. This motivates us to look for a set of conditions to help avoid such "invalid" placement and guide the design of an iterative placement algorithm. In the remainder of this section, we start by stating sufficient conditions to achieve heterogeneous SC-PIR capacity. Then, we present an example of an FP solution to achieve capacity in an equivalent heterogeneous SC-PIR storage placement problem. Next, we formally define the FP and define a set of feasible conditions which guarantees a FP solution.

A. Sufficient Conditions to Achieve Capacity for SC-PIR

We first provide sufficient conditions for a storage placement scheme to achieve the SC-PIR capacity in Lemma 1. These conditions were first presented in [1, Theorem 4] for the homogeneous case and then in [15, Lemmas 3,4] for the more general heterogeneous case. For completeness, we also provide a different proof to Lemma 1 that is shorter than the one given in [15].

Lemma 1: For a network with general heterogeneous storage requirements, consider the GDA with a capacity achieving FS-PIR scheme for delivery. The resulting SC-PIR scheme is capacity-achieving if the storage placement satisfies one of the following two conditions:

(a) $t \in \mathbb{Z}^+$ and $\forall f \in [F]$, the sub-library $\mathcal{M}_f$ is stored at t DBs, $|\mathcal{N}_f| = t$,

[4]The FS-PIR scheme of [6] uses an asymmetric design such that the possible queries result in varying rates. However, the expected rate meets FS-PIR capacity.

(b) $t \notin \mathbb{Z}^+$ and $\forall f \in [F]$, the sub-library $\mathcal{M}_f$ is stored at $\lfloor t \rfloor$ or $\lceil t \rceil$ DBs, $|\mathcal{N}_f| \in \{\lfloor t \rfloor, \lceil t \rceil\}$ such that

$$\sum_{\{f\in[F]:|\mathcal{N}_f|=\lfloor t \rfloor\}} \alpha[f] = \lceil t \rceil - t, \tag{21}$$

$$\sum_{\{f\in[F]:|\mathcal{N}_f|=\lceil t \rceil\}} \alpha[f] = t - \lfloor t \rfloor. \tag{22}$$

Proof: From [5], the rate of a capacity achieving FS-PIR scheme to privately download one of K messages from x DBs is $R_{\rm FS}(x) = \left(1 + \frac{1}{x} + \cdots + \frac{1}{x^{K-1}}\right)^{-1}$. For $t \in \mathbb{Z}^+$, it follows from Proposition 1 that the rate of the SC-PIR scheme is

$$\begin{aligned} R &= \left(\frac{\alpha[1]}{R_{\rm FS}(t)} + \cdots + \frac{\alpha[F]}{R_{\rm FS}(t)}\right)^{-1} = R_{\rm FS}(t) \\ &= \left(1 + \frac{1}{t} + \cdots + \frac{1}{t^{K-1}}\right)^{-1} \end{aligned} \tag{23}$$

which is the capacity of SC-PIR defined in (11). For $t \notin \mathbb{Z}^+$, it follows from Proposition 1 that

$$\begin{aligned} R &= \Big(\frac{1}{R_{\rm FS}(\lfloor t \rfloor)} \sum_{f:|\mathcal{N}_f|=\lfloor t \rfloor} \alpha[f] + \frac{1}{R_{\rm FS}(\lceil t \rceil)} \sum_{f:|\mathcal{N}_f|=\lceil t \rceil} \alpha[f]\Big)^{-1} \\ &= \Big(\frac{\lceil t \rceil - t}{R_{\rm FS}(\lfloor t \rfloor)} + \frac{t - \lfloor t \rfloor}{R_{\rm FS}(\lceil t \rceil)}\Big)^{-1}. \end{aligned} \tag{24}$$

Moreover, $R^{-1} = (\lceil t \rceil - t)R_{\rm FS}^{-1}(\lfloor t \rfloor) + (t - \lfloor t \rfloor)R_{\rm FS}^{-1}(\lceil t \rceil)$ is a linear interpolation of the points $\big(\lfloor t \rfloor, R_{\rm FS}^{-1}(\lfloor t \rfloor)\big)$ and $\big(\lceil t \rceil, R_{\rm FS}^{-1}(\lceil t \rceil)\big)$ which was shown to meet the capacity of SC-PIR with uncoded storage defined in (12). ■

B. A Heterogeneous SC-PIR Example

We use the Heterogeneous Placement Algorithm (HePA) presented later in Section VI to iteratively fill the DB storage where each iteration fills some contents of t DBs. Let $N = 8$ and the storage constraints of the DBs are

$$\boldsymbol{\mu} = [0.9,\ 0.65,\ 0.4,\ 0.3,\ 0.25,\ 0.2,\ 0.2,\ 0.1]. \tag{25}$$

For example, by this notation, DB3 has a storage capacity of $\frac{4}{10}KL$ bits. By summing the elements of $\boldsymbol{\mu}$, we obtain $t = 3$. By Lemma 1, using the GDA, capacity can be achieved if each sub-library, $\mathcal{M}_f$, is stored at exactly $t = 3$ DBs. This translates to a "filling problem" (FP) where our goal is to iteratively fill the storage of the DBs and in each iteration we fill some storage in exactly $t = 3$ DBs. In the first iteration, we define a sub-library, $\mathcal{M}_1$, which contains $\mu[1]L = \frac{1}{10}L$ arbitrary bits from each of the K messages and assign $\mathcal{M}_1$ to the DB subset $\mathcal{N}_1 = \{1, 2, 8\}$. Notice that $\mathcal{M}_1$ contains $\mu[8]KL$ bits and there is no remaining storage at DB8 after this iteration. Later in V-D, we will show that this iteration is moving towards a solution. Next, we aim to fill the storage contents of DB7 and let $\mathcal{M}_2$ contain $\frac{1}{5}L$ arbitrarily unpicked bits (i.e., bits are not in $\mathcal{M}_1$) from each of the K messages. Then, $\mathcal{M}_2$ is stored at the DB subset $\mathcal{N}_2 = \{1, 2, 7\}$. In general, $\mathcal{N}_f$ includes the DB with the least remaining non-zero storage and the $t - 1$ DBs with the most remaining storage. In each iteration, we aim to fill the DB with the least remaining non-zero storage. However, sometimes an iteration only yields a partial fill in order to converge towards a solution. For example, in the 5-th iteration, DB2 has the smallest remaining non-zero storage and is only partially filled. The design of each iteration is discussed in more detail in Section VI. This process is continued until the storage of all DBs are filled.

The resulting storage placement of the HePA is shown in Fig. 1. There are $F = 7$ sub-libraries and each is stored at exactly 3 DBs. A vector $\boldsymbol{\alpha} \in \mathbb{R}_+^F$ defines the fraction of the library that is stored (or filled) in each iteration. For example, $\alpha[1] = 0.1$ and $\alpha[2] = 0.2$ correspond to the first two iterations described above. All elements of $\boldsymbol{\alpha}$ are shown in the table of Fig. 1. The corresponding DBs that store a sub-library in iteration f are highlighted by red arrows in the table. By using a capacity achieving FS-PIR scheme for delivery, the placement achieves SC-PIR capacity. This is because the HePA is designed to place a sub-library at a sub-group of t DBs in each iteration and satisfy the sufficient conditions of Lemma 1 to achieve SC-PIR capacity.

The HePA has the advantage of linear complexity. We will show in Section VI-B that the HePA always converges in N iterations and requires at most $F = N$ sub-messages per message when t is an integer. The complexity is significantly reduced from the placement strategy of the state-of-the-art placement of [15], which aims to solve a linear program (LP) with $\binom{N}{t}$ variables representing the size of the sub-libraries at all possible sets of t DBs.

Fig. 1 contains two additional parameters, t' and e, which are discussed in greater detail later in Section VI-B. Moreover, t' is the sum of the cumulative remaining storage of all DBs and e is the number of DBs that each has a remaining storage that is equal to $\frac{t'KL}{t}$ bits. These parameters are significant when deriving the convergence of the HePA.

C. The Filling Problem

Definition 4: **$(\boldsymbol{m}, \tau)$-Filling Problem (FP)** Assume $\mathcal{B}$ is a basis containing the set of all $\{0, 1\}$-vectors of length N, each of which contains exactly τ 1s. Given a vector $\boldsymbol{m} \in \mathbb{R}_+^N$, representing a storage vector of N DBs, find a set of non-negative scalars $\{\alpha_{\boldsymbol{b}} \in \mathbb{R}_+ : \boldsymbol{b} \in \mathcal{B}\}$ such that

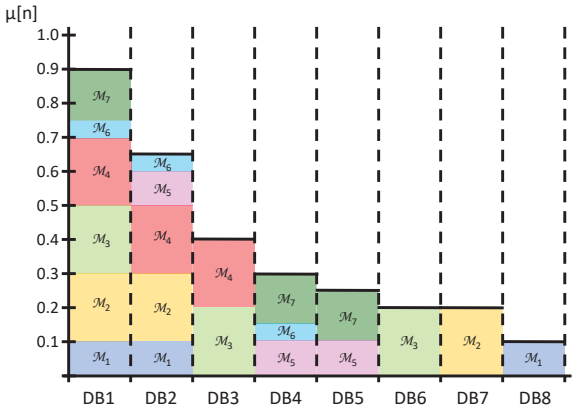
$$\sum_{\boldsymbol{b}\in\mathcal{B}} \alpha_{\boldsymbol{b}} \boldsymbol{b} = \boldsymbol{m}. \tag{26}$$

◊

For the heterogeneous SC-PIR problem with $t \in \mathbb{Z}^+$, the capacity achieving storage placement solution is equivalent to the $(\boldsymbol{\mu}, t)$-FP.

D. Existence of the $(\boldsymbol{m}, \tau)$-FP Solution

To design an iterative algorithm to solve the general FP, we aim to find a set of necessary and sufficient conditions that ensures each iteration allows a valid FP solution based on the remaining storage. In other words, since each iteration will yield a new FP, we are interested in conditions that show a solution to the $(\boldsymbol{m}, \tau)$-FP exists. This will guide the algorithm design such that we avoid "invalid" placements that would prevent convergence. For example, consider an SC-PIR system with $\boldsymbol{\mu} = [0.7, 0.7, 0.6]$ and $t = 2$. If in the first iteration, we



f	$\alpha[f]$	DB1	DB2	DB3	DB4	DB5	DB6	DB7	DB8	t'	e
1	0.1	0.9	0.65	0.4	0.3	0.25	0.2	0.2	0.1	3.0	0
2	0.2	0.8	0.55	0.4	0.3	0.25	0.2	0.2	0	2.7	0
3	0.2	0.6	0.35	0.4	0.3	0.25	0.2	0		2.1	0
4	0.2	0.4	0.35	0.2	0.3	0.25	0			1.5	0
5	0.1	0.2	0.15	0	0.3	0.25				0.9	0
6	0.05	0.2	0.05		0.2	0.15				0.6	1
7	0.15	0.15	0		0.15	0.15				0.45	3
-	-	0			0	0				0	-

Fig. 1. A solution to the filling problem using the Heterogeneous Placement Algorithm (HePA) when $t = 3$ and $\boldsymbol{\mu} = [0.9, 0.65, 0.4, 0.3, 0.25, 0.2, 0.2, 0.1]$. (left) A bar graph depicting the storage requirements of the DBs and the storage placement solution. (right) A table representing the remaining storage of the DBs for each iteration. The red arrows highlight which DBs are assigned a sub-message subset in each iteration. Note that the sub-libraries $\mathcal{M}_f$ have different sizes $\alpha[f]$ and each sub-library $\mathcal{M}_f$ is placed into a sub-group of $t = 3$ DBs in $\mathcal{N}_f$. Here, we have $\mathcal{N}_1 = \{1, 2, 8\}$, $\mathcal{N}_2 = \{1, 2, 7\}$, $\mathcal{N}_3 = \{1, 3, 6\}$, $\mathcal{N}_4 = \{1, 2, 3\}$, $\mathcal{N}_5 = \{2, 4, 5\}$, $\mathcal{N}_6 = \{1, 2, 4\}$, $\mathcal{N}_7 = \{1, 4, 5\}$.

place 0.4 of the message library at DB1 and DB3 then the remaining storage at the DBs is $\boldsymbol{m} = [0.3, 0.7, 0.2]$. However, an $(\boldsymbol{m}, t)$-FP solution does not exist since $m[1] + m[3] < m[2]$. It is impossible to fill the remaining storage two DBs at a time and completely fill DB2. Our goal is to design an algorithm that never reaches a situation like this. The algorithm will execute an iteration only if a FP solution exists after that iteration. This requires knowledge of necessary and sufficient conditions for a $(\boldsymbol{m}, \tau)$-FP solution to exist.

Theorem 2: Given $\boldsymbol{m} \in \mathbb{R}_+^N$ and $\tau \in \mathbb{Z}^+$ an $(\boldsymbol{m}, \tau)$-FP solution exists *if and only if*

$$m[n] \leq \frac{\sum_{i=1}^{N} m[i]}{\tau}, \quad \forall n \in [N]. \tag{27}$$

□

The proof of Theorem 2 is given in Appendix B.

Remark 2: First, we note that Theorem 2 guarantees the existence of a $(\boldsymbol{\mu}, t)$-FP solution for any general SC-PIR storage requirement $\boldsymbol{\mu}$ with integer t because $\mu[n] \leq 1 = \frac{\sum_{i=1}^{N} \mu[i]}{t}$ for all $n \in [N]$ and thus (27) is met. This establishes the existence of a heterogeneous capacity achieving SC-PIR scheme using a different approach from that of the previously known proof in [15, Lemma 5]. Second, while Theorem 2 itself does not provide an explicit algorithm to solve the storage placement problem for SC-PIR, it provides the theoretical basis needed to develop such an algorithm, e.g., the HePA in Section VI. Condition (27) is critical to ensure that each iteration of the HePA algorithm permits a new FP solution under the updated storage constraints. Third, condition (27) is also used to develop a storage sharing solution for $t \notin \mathbb{Z}^+$, as described in Section VII and Appendix D.

VI. Heterogeneous Placement Algorithm

In this section, we present the Heterogeneous Placement Algorithm (HePA) that achieves SC-PIR capacity for DBs with general heterogeneous storage requirements and integer t. Moreover, we will prove that HePA always converges, requires at most N iterations, and each message is split into at most N sub-messages. The results are summarized in the following theorem.

Theorem 3: Assume that $t \in \mathbb{Z}^+$ and DBs have general heterogeneous storage requirements. By using the GDA with HePA for placement and a capacity achieving FS-PIR scheme for delivery, the following rate is achievable

$$R = \left(1 + \frac{1}{t} + \frac{1}{t^2} + \cdots + \frac{1}{t^{K-1}}\right)^{-1} \tag{28}$$

which achieves the capacity of SC-PIR. □

Theorem 3 is proved in the following sections by presenting the scheme and proving its correctness and convergence characteristics.

A. Heterogeneous Placement Algorithm to Achieve SC-PIR Capacity

Heterogeneous Placement Algorithm (HePA). The HePA works under a heterogeneous storage constraint. As opposed to the homogeneous designs HoP1 and HoP2 where α, $\{\mathcal{M}_1, \cdots, \mathcal{M}_F\}$, and $\{\mathcal{N}_1, \cdots, \mathcal{N}_F\}$ are all specified at once, due to the heterogeneous storage, HePA adopts an iterative procedure to find these sequentially in at most N iterations. In each iteration f, HePA aims to fill the storage of the DB with the smallest remaining (non-zero) storage. Specifically, it determines $\alpha[f], \mathcal{M}_f$, and then places $\mathcal{M}_f$ at the t DBs in $\mathcal{N}_f$ that includes the DB with the smallest remaining storage and the $t-1$ DBs with the largest remaining storage. After at most N iterations, all $\{\mathcal{M}_1, \cdots, \mathcal{M}_F\}$ and $\{\mathcal{N}_1, \cdots, \mathcal{N}_F\}$ are specified and the heterogeneous storage constraint is met at each DB.

The HePA is outlined in Algorithm 1. Lines 5 through 18 define the f-th iteration, which we summarize as follows with reference to specific lines of Algorithm 1. By line 5, let N' be the number of DBs with non-zero remaining storage and $\boldsymbol{m} \in \mathbb{R}_+^N$ be the remaining storage of each DB normalized by

KL after the first $f-1$ iterations. Line 6 defines t' as the cumulative remaining storage of the DB network

$$t' = \sum_{n=1}^{N} m[n]. \tag{29}$$

By line 7, we introduce a length N vector $\boldsymbol{\ell}$ as a permutation of $[N]$ to order the DBs according to their remaining storage from largest to smallest. In other words, given N' and $\boldsymbol{\ell}$, we have

$$m[\ell[1]] \geq m[\ell[2]] \geq \cdots \geq m[\ell[N']] > 0, \tag{30}$$

$$m[\ell[N'+1]] = m[\ell[N'+2]] = \cdots = m[\ell[N]] = 0. \tag{31}$$

Then, by line 8, define the DB sub-group $\mathcal{N}_f$ that includes the DB with the smallest remaining (non-zero) storage and the $t-1$ DBs with the largest remaining storage

$$\mathcal{N}_f \;=\; \{\ell[1], \cdots, \ell[t-1], \ell[N']\}. \tag{32}$$

We need to define a sub-library, $\mathcal{M}_f$, of size $\alpha[f]KL$ bits to be stored at the DBs of $\mathcal{N}_f$. Ideally, to completely fill DB $\ell[N']$, we should set $\alpha[f] = m[\ell[N']]$. However, to ensure that $\boldsymbol{m}$ meets the conditions of (27) after the f-th iteration (i.e. after $\boldsymbol{m}$ is updated in lines 16-18), we do the following. When $N' \geq t+1$, from line 10, we let

$$\alpha[f] = \min\left(\frac{t'}{t} - m[\ell[t]],\;\; m[\ell[N']]\right). \tag{33}$$

When $N' = t$, the t DBs with non-zero remaining storage make up $\mathcal{N}_f$. In this case, all of the remaining storage of these t DBs are equal (can be shown using Theorem 2), and by line 12, we let $\alpha[f] = m[\ell[1]]$. In line 14, we define a sub-library, $\mathcal{M}_f$, containing $\alpha[f]L$ unstored bits from each message. In line 15, we store $\mathcal{M}_f$ at the DBs of $\mathcal{N}_f$. Then, in lines 16-18, we adjust $\boldsymbol{m}$ accordingly to reflect the remaining storage at each DB after the f-th iteration.

The HePA always converges to yield a SC-PIR capacity-achieving placement and requires at most N iterations to converge. Note that, the iterations of the HePA only operate when $N' \geq t$, because it will never reach a point where $N' < t$, unless $N' = 0$ and all DBs have been completely filled. In Appendix C, we demonstrate each iteration of the HePA fills a non-zero amount of storage and after each iteration the remaining storage satisfies the conditions of (27) such that a $(\boldsymbol{m}, t)$-FP solution exists. Convergence is characterized in the following section.

Achievable Rate: For integer t, the HePA meets the sufficient conditions of Lemma 1. Therefore, by using the HePA for placement with the GDA, SC-PIR capacity is achieved by using a capacity-achieving FS-PIR scheme for delivery.

B. Convergence

Surprisingly, we find that the HePA requires at most N iterations to produce a capacity-achieving SC-PIR placement for DBs with heterogeneous storage requirements. Moreover, the number of iterations is equal to the number of sub-messages per message, F, required for the storage placement. The convergence of the HePA is summarized in the following lemma.

Algorithm 1 Heterogeneous Placement Algorithm (HePA)

Input: $\boldsymbol{\mu}$, t, L and $W_1, \ldots, W_K$

1: $\boldsymbol{m} \leftarrow \boldsymbol{\mu}$
2: $f \leftarrow 0$
3: **while** $\boldsymbol{m} > \mathbf{0}$ **do**
4: $\quad f \leftarrow f+1$
5: $\quad N' \leftarrow$ number of non-zero elements in $\boldsymbol{m}$
6: $\quad t' \leftarrow \sum_{n=1}^{N} m[n]$
7: $\quad \boldsymbol{\ell} \leftarrow$ indices that would sort $\boldsymbol{m}$ from largest to smallest
8: $\quad \mathcal{N}_f \leftarrow \{\ell[1], \ldots, \ell[t-1], \ell[N']\}$
9: $\quad$ **if** $N' \geq t+1$ **then**
10: $\quad\quad \alpha[f] \leftarrow \min\left(\frac{t'}{t} - m[\ell[t]], m[\ell[N']]\right)$ ▷ ensures (27) is met after the f-th iteration
11: $\quad$ **else**
12: $\quad\quad \alpha[f] \leftarrow m[\ell[1]]$
13: $\quad$ **end if**
14: $\quad \mathcal{M}_f \leftarrow \alpha[f]L$ unstored bits from each message $\{W_k : k \in [K]\}$
15: $\quad$ Store $\mathcal{M}_f$ at the DBs of $\mathcal{N}_f$
16: $\quad$ **for** $n \in \mathcal{N}_f$ **do**
17: $\quad\quad m[n] \leftarrow m[n] - \alpha[f]$
18: $\quad$ **end for**
19: **end while**
20: $F \leftarrow f$

Lemma 2: The HePA always converges and requires at most N iterations to converge.

Proof: We label the outcome of each iteration of the HePA as either a *complete fill* (CF) or *partial fill* (PF) defined below.

Definition 5: A *complete fill* (CF) refers to an iteration where the remaining storage at the DB with the smallest remaining non-zero storage is completely filled. ♢

Definition 6: A *partial fill* (PF) refers to an iteration where the remaining storage at the DB with the smallest remaining non-zero storage is *not* completely filled. ♢

To obtain an upper bound on the number of iterations to fill the DBs, we count the maximum number of possible PFs and CFs. First, since HePA guarantees the existence of a $(\boldsymbol{m}, t)$-FP solution at each iteration, it follows from (27) that the maximum possible remaining storage at any DB at that iteration is $\frac{t'}{t}$. Our proof involves a variable, e, which counts the number of DBs whose remaining storage equals the maximum value of $\frac{t'}{t}$, defined as

$$e = \sum_{n=1}^{N} \mathbb{1}\left(m[n] = \frac{t'}{t}\right) \tag{34}$$

where $\mathbb{1}(\cdot)$ is the indicator function. In the following, for ease of notation, let $\tilde{\boldsymbol{m}}$ be the ordered remaining storage at the DBs from largest to smallest such that

$$\tilde{m}[n] = m[\ell[n]], \;\; \forall n \in [N]. \tag{35}$$

The following claim discusses the sufficient condition which guarantees a CF for a given iteration.

Claim 1: If a given iteration satisfies $e = t-1$ and $N' \geq t+1$, then this iteration must be a CF, and N' will be reduced by at least 1 after that iteration.

Proof: Using $e = t-1$ and (29), we obtain

$$(t-1)\frac{t'}{t} + \tilde{m}[t] + \cdots + \tilde{m}[N'] = t', \tag{36}$$

which follows that $\tilde{m}[N'] \leq \frac{t'}{t} - \tilde{m}[t]$. By (33), during this iteration, $\alpha[f]KL$ bits are stored at DB $\ell[N']$ where $\alpha[f] = \tilde{m}[N']$. This completes the proof of Claim 1. ∎

Claim 2: If a given iteration satisfies $e \leq t-1$ and $N' \geq t+1$, then e will not decrease after that iteration. Moreover, if the iteration is a PF then e will be increased by at least 1 after that iteration.

Proof: The e DBs with remaining storage equal to $\frac{t'}{t}$ are included in the set of $t-1$ DBs with the largest remaining storage since $\tilde{m}[n] \leq \frac{t'}{t}$ for all $n \in [N]$ based on the conditions of (27). Therefore, after an iteration, the remaining storage of these e DBs are reduced by $\alpha[f]$ and their remaining storage becomes $\frac{t'}{t} - \alpha[f]$. Furthermore, let t'' be the sum of storage after this iteration. Then, $t'' = t' - t\alpha[f]$ and $\frac{t''}{t} = \frac{t'}{t} - \alpha[f]$. Hence, whether the iteration is a PF or CF, e is not decreasing from one iteration to the next.

Next, if the iteration is a PF, then by (33), we obtain $\tilde{m}[N'] > \frac{t'}{t} - \tilde{m}[t]$. Therefore, $\alpha[f] = \frac{t'}{t} - \tilde{m}[t]$. Furthermore, $\frac{t''}{t} = \tilde{m}[t]$. As the remaining storage at DB $\ell[t]$ remains $\tilde{m}[t] = \frac{t''}{t}$ and this DB is not included in the e DBs with $\frac{t'}{t}$ remaining storage,[5] e is increased by at least 1 after this iteration. This completes the proof of Claim 2. ∎

By Claims 1 and 2, we can conclude that at most $t-1$ PFs and $N-t$ CFs are possible during the execution of the HePA as N' is decreased from N to t. Then when $N' = t$, there are t DBs with equal remaining storage and the special case of the HePA which fills the remaining storage of these DBs. As a result, at most $(t-1)+(N-t)+1 = N$ iterations of the HePA are necessary to completely fill the DBs. ∎

Remark 3: It can also be shown that if $N' \geq 2t$ then an iteration of the HePA will result in a CF. In other words, the first $N-2t+1$ iterations are guaranteed to be a CF.

Remark 4: If $\tilde{m}[N'] = \frac{t'}{t} - \tilde{m}[t]$, where $\tilde{\boldsymbol{m}}$ is defined in (35), then an iteration of HePA will result in a CF and e will increase by at least 1. Moreover, if there are multiple DBs with remaining storage equal to $\tilde{m}[t]$, then e will increase by more than 1 if the iteration is a PF. These special cases demonstrate that sometimes a number of iterations strictly less than N may be sufficient to fill the DBs as seen in the example of Section V-B.

C. Comparison to the State-of-the-Art

The state-of-the-art heterogeneous SC-PIR design proposed in [15] achieves the capacity defined in (11) for integer t. However, the placement strategy involves solving a linear program (LP) with $\binom{N}{t}$ variables representing the size of the sub-libraries placed at all possible sets of t DBs. Solving such a LP is generally infeasible for large N. Moreover, the placement solution will have $F = \binom{N}{t}$ sub-messages per message. Although the solution may yield some sub-messages with zero-size, there is no guarantee on the maximum number of non-zero sub-messages, except that it will be less than or equal to $\binom{N}{t}$. Alternatively, for general heterogenous storage requirements with integer t, the HePA produces a capacity-achieving storage solution with at most N sub-messages per message, $F \leq N$. Also, the HePA is feasible for a large number of DBs as only N iterations are required for convergence.

[5]This is because that if this DB is included the e DBs with $\frac{t'}{t}$ remaining storage, then $e \geq t$.

VII. A Capacity Achieving Placement Design for Non-Integer t

In previous sections, we assumed that t is an integer and sub-libraries are always stored at t nodes. In practice, t may not be an integer. In this section, we aim to find a capacity achieving solution to the general heterogeneous SC-PIR problem with a non-integer t. The main challenge is to design a *storage sharing* scheme that satisfies (27) which guarantees the existence of a FP solution. The key idea is to split the storage placement problem into two storage placement sub-problems with integer t. In the following, we first provide a motivating example and then derive the sufficient conditions to meet SC-PIR capacity. A general scheme that meets the sufficient conditions is presented at the end of this section.

A. An Example for $t \notin \mathbb{Z}^+$

Let $N = 5$ with storage requirements defined by

$$\boldsymbol{\mu} = \left[1, \ \frac{3}{5}, \ \frac{2}{5}, \ \frac{1}{5}, \ \frac{1}{5}\right] \tag{37}$$

and $t = \frac{12}{5}$. The HePA cannot operate on these requirements since t is not an integer. Instead we have to satisfy the conditions (21) and (22) of Lemma 1. In other words, for each DB we split the storage into two parts. We define $\boldsymbol{\mu}^{(2)}$ and $\boldsymbol{\mu}^{(3)}$ as the set of storage requirements allotted for the $\left(\boldsymbol{\mu}^{(2)}, 2\right)$ and $\left(\boldsymbol{\mu}^{(3)}, 3\right)$ FPs, respectively, such that $\boldsymbol{\mu}^{(2)} + \boldsymbol{\mu}^{(3)} = \boldsymbol{\mu}$.

From (21) and (22) of Lemma 1, the elements of $\boldsymbol{\mu}^{(2)}$ must sum to $\lfloor t \rfloor (\lceil t \rceil - t) = \frac{6}{5}$ and the elements of $\boldsymbol{\mu}^{(3)}$ must sum to $\lceil t \rceil (t - \lfloor t \rfloor) = \frac{6}{5}$. A naive approach is to simply split the storage of each DB in half:

$$\boldsymbol{\mu}^{(2)} = \boldsymbol{\mu}^{(3)} = \frac{1}{2}\boldsymbol{\mu} = \left[\frac{1}{2}, \ \frac{3}{10}, \ \frac{1}{5}, \ \frac{1}{10}, \ \frac{1}{10}\right]. \tag{38}$$

While (21) and (22) are met, there is an issue with this approach since a $\left(\boldsymbol{\mu}^{(3)}, 3\right)$-FP solution does not exist. This is the case since $\mu^{(1)}[1] = \frac{1}{2} > \frac{1}{3}\sum_{n=1}^{5} \mu^{(3)}[n] = \frac{2}{5}$ and condition (27) of Theorem 2 is not met.

We can use an alternative approach to define the storage sharing in order to meet Theorem 2 by enforcing $\mu^{(2)}[1] \geq \frac{3}{5}$. Then, we find

$$\mu^{(3)}[1] + \mu^{(2)}[1] = \mu[1] = 1 \quad \text{and} \quad \mu^{(3)}[1] \leq 1 - \frac{3}{5} = \frac{2}{5}. \tag{39}$$

To ensure that there are no issues with the storage sharing of the other DBs, we enforce

$$\mu^{(2)}[n] \geq \mu[n] - \frac{\lceil t \rceil (t - \lfloor t \rfloor)}{\lceil t \rceil} = \mu[n] - \frac{2}{5}, \ \text{for } n = 1,2,3,4,5 \tag{40}$$

such that $\mu^{(3)}[n] = \mu[n] - \mu^{(2)}[n] \leq \frac{2}{5}$ which ensures that $\boldsymbol{\mu}^{(3)}$ has a FP solution. Similarly, we enforce

$$\mu^{(3)}[n] \geq \mu[n] - \frac{\lfloor t \rfloor (\lceil t \rceil - t)}{\lfloor t \rfloor} = \mu[n] - \frac{3}{5}, \text{ for } n = 1, 2, 3, 4, 5 \tag{41}$$

so that $\boldsymbol{\mu}^{(2)}$ has a FP solution. Using this approach we can define the data sharing scheme by

$$\begin{aligned} \boldsymbol{\mu}^{(2)} &= \left[\frac{3}{5}, \ \frac{1}{5}, \ 0, \ 0, \ 0\right] + \left[0, \ \frac{2}{15}, \ \frac{2}{15}, \ \frac{1}{15}, \ \frac{1}{15}\right] \\ &= \left[\frac{3}{5}, \ \frac{1}{3}, \ \frac{2}{15}, \ \frac{1}{15}, \ \frac{1}{15}\right] \end{aligned} \tag{42}$$

$$\begin{aligned} \boldsymbol{\mu}^{(3)} &= \left[\frac{2}{5}, \ 0, \ 0, \ 0, \ 0\right] + \left[0, \ \frac{4}{15}, \ \frac{4}{15}, \ \frac{2}{15}, \ \frac{2}{15}\right] \\ &= \left[\frac{2}{5}, \ \frac{4}{15}, \ \frac{4}{15}, \ \frac{2}{15}, \ \frac{2}{15}\right]. \end{aligned} \tag{43}$$

Specifically, on the RHS of the first equation of (42) and (43), the first term ensures Theorem 2 is met. In our general scheme, we label these terms as $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$, respectively. Then, the second term is defined such that both of the resulting vectors sum to $\frac{6}{5}$ and (21) and (22) are met which are also defined on our general scheme. At this point, the HePA can be used to define the placement of the equivalent $\left(\boldsymbol{\mu}^{(2)}, 2\right)$ and $\left(\boldsymbol{\mu}^{(3)}, 3\right)$ FPs.

B. Storage Sharing Sufficient Conditions

Define $\boldsymbol{\mu}^{(\lfloor t \rfloor)}, \boldsymbol{\mu}^{(\lceil t \rceil)} \in \mathbb{R}_+^N$ such that $\boldsymbol{\mu}^{(\lfloor t \rfloor)} + \boldsymbol{\mu}^{(\lceil t \rceil)} = \boldsymbol{\mu}$,

$$\sum_{i=1}^{N} \mu^{(\lfloor t \rfloor)}[i] = \lfloor t \rfloor (\lceil t \rceil - t), \tag{44}$$

$$\sum_{i=1}^{N} \mu^{(\lceil t \rceil)}[i] = \lceil t \rceil (t - \lfloor t \rfloor), \tag{45}$$

and the condition (27) for $\mu^{(\lfloor t \rfloor)}[n]$ and $\mu^{(\lceil t \rceil)}[n]$ is given by

$$\mu^{(\lfloor t \rfloor)}[n] \leq \frac{\sum_{i=1}^{N} \mu^{(\lfloor t \rfloor)}[i]}{\lfloor t \rfloor} = \lceil t \rceil - t, \tag{46}$$

$$\mu^{(\lceil t \rceil)}[n] \leq \frac{\sum_{i=1}^{N} \mu^{(\lceil t \rceil)}[i]}{\lceil t \rceil} = t - \lfloor t \rfloor \tag{47}$$

for all $n \in [N]$. We find that (44) and (45) satisfy (21) and (22) to achieve heterogeneous SC-PIR capacity. Moreover, (46) and (47) guarantee that a solution exists to both the $(\boldsymbol{\mu}^{(\lfloor t \rfloor)}, \lfloor t \rfloor)$-FP and $(\boldsymbol{\mu}^{(\lceil t \rceil)}, \lceil t \rceil)$-FP. Then, split each message W_k into two disjoint sub-messages, $W_k^{(\lfloor t \rfloor)}$ of size $(\lceil t \rceil - t)L$ bits and $W_k^{(\lceil t \rceil)}$ of size $(\lfloor t \rfloor - t)L$ bits which are used to for each FP. These two FPs can then be solved by the HePA.

C. A Storage Sharing Solution

Given $\boldsymbol{\mu}$, the following process will yield a $\boldsymbol{\mu}^{(\lfloor t \rfloor)}$ and $\boldsymbol{\mu}^{(\lceil t \rceil)}$ which meet the above conditions. Define $\boldsymbol{m}_1, \boldsymbol{m}_2 \in \mathbb{R}_+^N$ such that

$$m_1[n] = \left[\mu[n] - (t - \lfloor t \rfloor)\right]^+, \tag{48}$$

$$m_2[n] = \left[\mu[n] - (\lceil t \rceil - t)\right]^+ \tag{49}$$

$\forall n \in [N]$, where $[\cdot]^+$ returns the input if the input is non-negative, or returns 0 otherwise. Let

$$r = \frac{\lfloor t \rfloor (\lceil t \rceil - t) - \sum_{n=1}^{N} m_1[n]}{t - \sum_{n=1}^{N} m_1[n] - \sum_{n=1}^{N} m_2[n]}, \tag{50}$$

$$\boldsymbol{\mu}^{(\lfloor t \rfloor)} = \boldsymbol{m}_1 + (\boldsymbol{\mu} - \boldsymbol{m}_1 - \boldsymbol{m}_2) \cdot r, \tag{51}$$

$$\boldsymbol{\mu}^{(\lceil t \rceil)} = \boldsymbol{m}_2 + (\boldsymbol{\mu} - \boldsymbol{m}_1 - \boldsymbol{m}_2) \cdot (1 - r). \tag{52}$$

The correctness of this scheme for $t \notin \mathbb{Z}^+$ is proved in Appendix D. Note that, the initial memory allocation of $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$ are required to have a FP solution. Then, there are many design choices to define the remaining memory sharing and the proposed solution is not unique. Here, we provide one approach which is to split the remaining memory of each DB with constant fraction r.

VIII. Discussion

Surprisingly, from homogeneous to heterogeneous SC-PIR, there is no loss in rate as shown in [15] and no increase in the number of sub-messages as shown here. The total number of sub-messages is the product of the number of sub-messages necessary for the storage and delivery phases. By using the FS-PIR scheme of [6] for delivery and any of our proposed placement designs, the total number of sub-messages per message is at most $S = N \times (t-1) < N^2$. Amazingly, this implies that SC-PIR, with either homogeneous or heterogeneous DB storage requirements, may be practical for a large number of DBs, N. Furthermore, the number of sub-messages is constant with respect to the number of messages, K.

Another important aspect is the required message size in terms of the number of bits using the HePA for the general heterogeneous SC-PIR problem. The sub-messages have different sizes and $\alpha[f]L$ must be an integer for all $f \in [F]$. In general, the minimum size of L based on the HePA is still $O(N^2)$. However, it appears to be a function of all the distinct values of $\boldsymbol{\mu}$.

It is possible to use the HePA on a set of homogeneous storage requirements. It is interesting to observe how the result compares to HoP1 and HoP2. It can be shown that the HePA will completely fill t DBs with each iteration until the number of remaining DBs is $N' \leq 2t - 1$. This pattern reflects the storage placement of HoP1 for $\frac{N}{t} \in \mathbb{Z}^+$. In fact if $\frac{N}{t} \in \mathbb{Z}^+$, then the HePA will yield the same storage placement as HoP1. Otherwise, once $N' \leq 2t-1$, the HePA will place sub-libraries at the remaining DBs in a cyclic manner mimicking HoP2.

There are several interesting directions for future work. First, it remains an open problem to determine the minimum message size L for a given set of storage requirements. It was shown in [6] that the minimum L of an FS-PIR problem can be reduced significantly from N^{K-1} in [7] to $N-1$. The new FS-PIR scheme [6] and proof techniques therein may be useful to derive the minimum L for the homogeneous and heterogeneous SC-PIR problems. Second, another work [9] has considered random placement among databases where a database stores a bit of a given message with probability μ_0. Interestingly, this placement method was also used in

[16] for the coded caching problem. It will be meaningful to examine alternative random placement strategies for the SC-PIR problem where messages are split into a finite number of sub-messages.

IX. Conclusion

In this work, we proposed capacity-achieving storage placement designs for SC-PIR with both homogeneous and heterogeneous storage requirements. The storage placement designs were developed from scratch and require only a linear number of sub-messages per message. Moreover, we provided necessary and sufficient conditions to achieve SC-PIR capacity. We translated the general heterogeneous SC-PIR placement problem into a filling problem (FP). Then, we proposed an iterative algorithm that solves the FP in at most N iterations. With the appropriate choice of FS-PIR delivery scheme, our designs require at most a number of sub-messages per message that is quadratic relative to the number of DBs. Compared to the state-of-the-art SC-PIR designs we demonstrate an exponential reduction in required message size. Finally, we showed that while our design was first proposed for the case of integer t, it can be extended to the case of non-integer t by splitting the SC-PIR placement problem for the latter case into two equivalent problems of integer t.

Appendix A
Privacy of General Design Architecture

We show the condition of (7) holds when using the GDA. Let

$$Q_n^{[k]} = Q_{n,1}^{[k]}, \ldots, Q_{n,F}^{[k]} \quad \text{and} \quad A_n^{[k]} = A_{n,1}^{[k]}, \ldots, A_{n,F}^{[k]} \tag{53}$$

where $Q_{n,f}^{[k]}$ and $A_{n,f}^{[k]}$ are the query to and response from DB n when the user is privately downloading the sub-message $W_{k,f}$ of the requested message, W_k.[6] Furthermore, let

$$Z_n = Z_{n,1}, \ldots, Z_{n,F} \tag{54}$$

where $Z_{n,f}$ is the storage contents of DB n which intersects the sub-library $\mathcal{M}_f = W_{1,f}, \ldots, W_{K,f}$. Finally, let

$$V_{n,f} = Q_{n,f}^{[k]}, A_{n,f}^{[k]}, W_{1,f}, \ldots, W_{K,f}, Z_{1,f}, \ldots, Z_{N,f}. \tag{55}$$

The LHS of (7) can be re-written as

$$\begin{aligned} &I\left(k; Q_n^{[k]}, A_n^{[k]}, W_1, \ldots, W_K, Z_1, \ldots, Z_N\right) \\ &= I\left(k; V_{n,1}, \ldots, V_{n,F}\right) \\ &= \sum_{f=1}^{F} I\left(k; V_{n,f} | V_{n,1}, \ldots, V_{n,f-1}\right). \end{aligned} \tag{56}$$

By showing each term of (56) is 0, we can prove the condition of (7) holds. The first term

$$\begin{aligned} &I\left(k; V_{n,1}\right) \\ &= I\left(k; Q_{n,1}^{[k]}, A_{n,1}^{[k]}, W_{1,1}, \ldots, W_{K,1}, Z_{1,1}, \ldots, Z_{N,1}\right) \end{aligned} \tag{57}$$

equals 0 because we assume the FS-PIR scheme used in private. Next, we find

$$\begin{aligned} &I\left(k; V_{n,f} | V_{n,1}, \ldots, V_{n,f-1}\right) \\ &\overset{(a)}{=} H\left(V_{n,f} | V_{n,1}, \ldots, V_{n,f-1}\right) \\ &\quad - H\left(V_{n,f} | k, V_{n,1}, \ldots, V_{n,f-1}\right) \qquad (58) \\ &\overset{(b)}{=} H\left(V_{n,f} | V_{n,1}, \ldots, V_{n,f-1}\right) - H\left(V_{n,1}, \ldots, V_{n,f} | k\right) \\ &\quad + H\left(V_{n,1}, \ldots, V_{n,f-1} | k\right) \qquad (59) \\ &\overset{(c)}{=} H\left(V_{n,f} | V_{n,1}, \ldots, V_{n,f-1}\right) - H\left(V_{n,1}, \ldots, V_{n,f}\right) \\ &\quad + H\left(V_{n,1}, \ldots, V_{n,f-1}\right) \qquad (60) \\ &\overset{(d)}{=} 0 \qquad (61) \end{aligned}$$

where (a), (b) and (d) hold from rules of information theory. Moreover, (c) holds because the FS-PIR scheme for generates queries such that the distribution of the query, answers, messages and storage contents are independent of the label of the desired message, k.

Appendix B
Proof of Theorem 2, Existence of FP Solution

Proof: The proof is split into two claims.

Claim 3: If a $(\boldsymbol{m}, \tau)$-FP exists then $m[n] \leq \frac{\sum_{i=1}^{N} m[i]}{\tau}$ for all $n \in [N]$.

We prove Claim 3 by induction. Let $\mathcal{B}$ be the basis that includes all possible $\{0,1\}$-vectors of length N with exactly τ 1s. Define an order to the basis $\mathcal{B}$ such that $\mathcal{B} = \{\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots\}$. If a $(\boldsymbol{m}, \tau)$-FP solution exists then $\boldsymbol{m} = \sum_{j=1}^{|\mathcal{B}|} \alpha[j]\boldsymbol{b}_j$ for some $\boldsymbol{\alpha} \in \mathbb{R}_+^{|\mathcal{B}|}$. Define $\boldsymbol{m}^{(k)} = \sum_{j=1}^{k} \alpha[j]\boldsymbol{b}_j$. Notice that $\boldsymbol{m}^{(1)} = \alpha[1]\boldsymbol{b}_1$ and

$$\max_n m^{(1)}[n] = \alpha[1] = \frac{\sum_{i=1}^{N} m[i]}{\tau}. \tag{62}$$

Now, since $m^{(k+1)} = m^{(k)} + \alpha[k+1]\boldsymbol{b}_{k+1}$ and assuming $\max_n m^{(k)}[n] \leq \frac{\sum_{i=1}^{N} m^{(k)}[i]}{\tau}$, we find

$$\begin{aligned} \max_n m^{(k+1)}[n] &\leq \alpha[k+1] + \max_n m^{(k)}[n] \\ &\leq \alpha[k+1] + \frac{\sum_{i=1}^{N} m^{(k)}[i]}{\tau} \\ &= \frac{\tau\alpha[k+1] + \sum_{i=1}^{N} m^{(k)}[i]}{\tau} \\ &= \frac{\sum_{i=1}^{N} m^{(k+1)}[i]}{\tau}. \end{aligned} \tag{63}$$

Therefore, by summing every term, $\alpha[j]\boldsymbol{b}_j$, we find $m[n] \leq \frac{\sum_{i=1}^{N} m[i]}{\tau}$ for all $n \in [N]$. This completes the proof of Claim 3.

Claim 4: If $m[n] \leq \frac{\sum_{i=1}^{N} m[i]}{\tau}$ for all $n \in [N]$ then a $(\boldsymbol{m}, \tau)$-FP solution exists.

The proof of Claim 4 is as follows. Assume $\boldsymbol{m}$ sums to some positive number $a \in \mathbb{R}_+$. Then, the set representing all possibilities for $\boldsymbol{m}$ is

$$\begin{aligned} \mathcal{L}_a = \Big\{ \boldsymbol{m}' \in \mathbb{R}^N : &\sum_{i=1}^{N} m'[i] = a, \\ &0 \leq m'[n] \leq \frac{a}{\tau}, \ \ \forall n \in [N] \Big\}. \end{aligned} \tag{64}$$

[6] Note that, it is possible that the realization of $Q_{n,f}^{[k]}$ and $A_{n,f}^{[k]}$ are empty because DB n does not have $W_{k,f}$ locally stored.

$\mathcal{L}_a$ is defined by the intersection of $2N$ half-spaces and 1 plane and therefore $\mathcal{L}_a$ is convex. Moreover, $\mathcal{L}_a$ is bounded and closed. Therefore, $\mathcal{L}_a$ is defined by all convex combinations of its *corner points*. We aim to show that the corner of points of $\mathcal{L}_a$ include and only include a constant multiple of all possible $\{0,1\}$-vectors with exactly τ 1s.

A point $\boldsymbol{m}' \in \mathcal{L}_a$ is a corner point if and only if $\boldsymbol{m}'$ sums to a and is a unique point that intersects a subset of the planes $m'[n]=0$ and $m'[n]=\frac{a}{\tau}$ for $n\in[N]$. We consider the intersection of all subsets of these planes. Specifically, consider the intersection of the planes $m'[n]=0$ for $n\in\mathcal{E}$ and $m'[n]=\frac{a}{\tau}$ for $n\in\mathcal{S}$ where $\mathcal{E},\mathcal{S}\subseteq[N]$ and $\mathcal{E}\cap\mathcal{S}=\emptyset$. First, notice that for any pair $\mathcal{E},\mathcal{S}$ such that $|\mathcal{E}|=N-\tau$ and $|\mathcal{S}|=\tau$, a unique point is defined

$$m'[n]=\begin{cases} a/\tau, & n\in\mathcal{S} \\ 0 & n\notin\mathcal{S} \end{cases} \tag{65}$$

that sums to a and is in $\mathcal{L}_a$. In other words, any point $\boldsymbol{m}'$ is a corner point of $\mathcal{L}_a$ if is has exactly τ non-zero elements and these elements are equal to $\frac{a}{\tau}$. We have left to show that these are the only corner points.

Next, notice that if $|\mathcal{S}|>\tau$, then $\sum_{i=1}^N m'[i]>a$ and $\boldsymbol{m}'\notin\mathcal{L}_a$ is not a corner point. Similarly, if $|\mathcal{E}|>N-\tau$, then $\sum_{i=1}^N m'[i]<a$ and $\boldsymbol{m}'\notin\mathcal{L}_a$ is not a corner point. Furthermore, if $|\mathcal{S}|<\tau$, $|\mathcal{E}|<N-\tau$ and $|\mathcal{S}\cup\mathcal{E}|=N-1$, then all but one element of $\boldsymbol{m}'$ is defined, but by assuming $\boldsymbol{m}'\in\mathcal{L}_a$ sums to a, the element can be solved for and we end up with one of the previously defined corner points with exactly τ non-zero elements. Finally, consider the case where $|\mathcal{S}|<\tau$, $|\mathcal{E}|<N-\tau$ and $|\mathcal{S}\cup\mathcal{E}|\leq N-2$, then at least two elements of $\boldsymbol{m}'$ are undefined and the system does not have a unique solution, therefore, it does not yield a corner point.

Hence, the corner points of $\mathcal{L}_a$ exactly include a constant multiple of all possible $\{0,1\}$-vectors with exactly τ 1s, or the basis for the FP. Since any point $\boldsymbol{m}\in\mathcal{L}_a$ is a convex combination of these corner points, there exists a $(\boldsymbol{m},\tau)$-FP solution. This holds for all $a\geq 0$. This completes the proof of Claim 4 and also Theorem 2. ∎

Appendix C
Correctness of the Heterogeneous Placement Algorithm

We demonstrate the HePA always converges towards a solution by first showing that each iteration yields a storage vector that has a FP solution such that it satisfies conditions (27) of Theorem 2. We show this by induction. First, from Remark 2, we know that a $(\boldsymbol{\mu},t)$-FP solution exists for general storage requirements $\boldsymbol{\mu}$ with integer t. This means that (27) is satisfied at the beginning of the first iteration $f=1$. Next, we will show that, if a $(\boldsymbol{m},t)$-FP solution exists at the beginning the f-th iteration (at line 5 of Algorithm 1), then a FP solution exists after the f-th iteration (at line 18 of Algorithm 1). For ease of notation, let $\tilde{\boldsymbol{m}}$ denote the ordered remaining storage at the DBs from largest to smallest such that

$$\tilde{m}[n]=m[\ell[n]],\ \text{for } n\in[N]. \tag{66}$$

Throughout the following we assume the remaining storage before the f-th iteration, $\boldsymbol{m}$, satisfies (27) and equivalently $\tilde{\boldsymbol{m}}$ satisfies (27).

Let $\boldsymbol{m}'$ be the remaining storage at the DBs after the f-th iteration

$$\boldsymbol{m}'=\tilde{\boldsymbol{m}}-\alpha[f]\cdot[\underbrace{1,\dots,1}_{t-1},\underbrace{0,\dots,0}_{N'-t},1,\underbrace{0,\dots,0}_{N-N'}]. \tag{67}$$

Note that, the elements of $\boldsymbol{m}'$ are not necessarily in order. After the f-th iteration, the largest remaining storage at any DB is either $m'[1]=\tilde{m}[1]-\alpha[f]$ or $m'[t]=\tilde{m}[t]$. Using (27) and (67), we find

$$m'[1]=\tilde{m}[1]-\alpha[f]\leq\frac{\sum_{i=1}^N\tilde{m}[i]}{t}-\alpha[f]=\frac{\sum_{i=1}^N m'[i]}{t}. \tag{68}$$

Also, by (33), $\alpha[f]\leq\frac{\sum_{i=1}^N m[i]}{t}-m[t]$, then using (67), we find

$$m'[t]=m[t]\leq\frac{\sum_{i=1}^N\tilde{m}[i]}{t}-\alpha[f]=\frac{\sum_{i=1}^N m'[i]}{t}. \tag{69}$$

Furthermore, for all $n\in[N']$, $\alpha[f]\leq\tilde{m}[N']\leq\tilde{m}[n]$ and $m'[n]\geq\tilde{m}[n]-\alpha[f]\geq 0$. Finally, $m'[n]=\tilde{m}[n]=0$ for all $n\in[N'+1:N]$. Since $0\leq m'[n]\leq\frac{\sum_{i=1}^N m'[i]}{t}$ for all $n\in[N]$ and the conditions of (27) are met, by Theorem 2, a $(\boldsymbol{m}',t)$-FP solution exists.

Next, we show that in the f-th iteration we always have $\alpha[f]>0$, and thus a non-zero sub-library is placed at the DBs of $\mathcal{N}_f$. Since a $(\boldsymbol{m},t)$-FP solution exists at the beginning the f-th iteration, i.e., (27) is met, we must have $\tilde{m}[t]\leq\frac{t'}{t}$. This, combined with $\tilde{m}[N']>0$ and (33), ensure that $\alpha[f]\geq 0$. Moreover, $\alpha[f]=0$ if and only if

$$\tilde{m}[t]=\frac{t'}{t}=\frac{\sum_{i=1}^N\tilde{m}[i]}{t}. \tag{70}$$

However, in this case we find for all $n\in[t]$ that $\tilde{m}[n]=\tilde{m}[t]=\frac{t'}{t}$ since

$$\frac{\sum_{i=1}^N\tilde{m}[i]}{t}\geq\tilde{m}[n]\geq\tilde{m}[t]=\frac{\sum_{i=1}^N\tilde{m}[i]}{t}. \tag{71}$$

This means that $N'=t$ and (33) is not used to define $\alpha[f]$. Instead, we set $\alpha[f]=\tilde{m}[1]>0$ when $N'=t$.

Appendix D
Correctness of Non-Integer t Scheme

In this section, when t is not an integer, we will show that $\boldsymbol{\mu}^{(\lfloor t\rfloor)}$ and $\boldsymbol{\mu}^{(\lceil t\rceil)}$ as defined by (51) and (52), respectively, are non-negative vectors which satisfy the conditions of (44)-(47). In the following, we show (44) is satisfied.

$$\begin{aligned}\sum_{n=1}^N\mu^{(\lfloor t\rfloor)}[n]&=\sum_{n=1}^N m_1[n]+r\Big(t-\sum_{n=1}^N m_1[n]-\sum_{n=1}^N m_2[n]\Big)\\&=\sum_{n=1}^N m_1[n]+\lfloor t\rfloor(\lceil t\rceil-t)-\sum_{n=1}^N m_1[n]\\&=\lfloor t\rfloor(\lceil t\rceil-t).\end{aligned} \tag{72}$$

In the following, we show (45) is satisfied.

$$\begin{aligned}
&\sum_{n=1}^{N}\mu^{(\lceil t\rceil)}[n] \\
&\quad=\sum_{n=1}^{N}m_2[n]+(1-r)\Big(t-\sum_{n=1}^{N}m_1[n]-\sum_{n=1}^{N}m_2[n]\Big) \\
&\quad=\sum_{n=1}^{N}m_2[n]+t-\sum_{n=1}^{N}m_1[n]-\sum_{n=1}^{N}m_2[n] \\
&\qquad-\lfloor t\rfloor(\lceil t\rceil-t)+\sum_{n=1}^{N}m_1[n] \\
&\quad=t-\lfloor t\rfloor(\lceil t\rceil-t)=\lceil t\rceil(t-\lfloor t\rfloor). \qquad (73)
\end{aligned}$$

Next, we use the following lemmas which are proven in the latter part of Appendix D.

Lemma 3: Given the vectors $\boldsymbol{m}_1$ and $\boldsymbol{m}_2$ defined in (48) and (49), respectively, we have

$$m_1[n]+m_2[n]\le\mu[n],\quad \forall n\in[N] \qquad (74)$$

and $m_1[n]+m_2[n]=\mu[n]$, *if and only if* $\mu[n]\in\{0,1\}$.[7] □

Lemma 4: Given r defined in (50), we have $0\le r<1$. □

Given Lemmas 3 and 4, since $m_1[n]\ge 0$ and $m_2[n]\ge 0$ for all $n\in[N]$, then $\boldsymbol{\mu}^{(\lfloor t\rfloor)}$ and $\boldsymbol{\mu}^{(\lceil t\rceil)}$ have only non-negative elements. Moreover, [8]

$$\begin{aligned}
\mu^{(\lfloor t\rfloor)}[n]&<m_1[n]+(\mu[n]-m_1[n]-m_2[n]) \\
&=\mu[n]-\Big[\mu[n]-(\lceil t\rceil-t)\Big]^{+}\le\lceil t\rceil-t \qquad (75)
\end{aligned}$$

for all $n\in[N]$. Hence, (46) is satisfied. Similarly,

$$\begin{aligned}
\mu^{(\lceil t\rceil)}[n]&\le m_2[n]+(\mu[n]-m_1[n]-m_2[n]) \\
&=\mu[n]-\Big[\mu[n]-(t-\lfloor t\rfloor)\Big]^{+}\le t-\lfloor t\rfloor \qquad (76)
\end{aligned}$$

for all $n\in[N]$ and (47) is satisfied. This completes the proof of correctness. The rest of this Appendix D is devoted to proving Lemmas 3 and 4.

A. Proof of Lemma 3

We first prove (74). In the following, according to the value of $\mu[n]$, we have four cases.

1) If $\mu[n]\le t-\lfloor t\rfloor$ and $\mu[n]\le\lceil t\rceil-t$, then $m_1[n]=m_2[n]=0$ and
$$m_1[n]+m_2[n]=0\le\mu[n]. \qquad (77)$$
2) If $\mu[n]>t-\lfloor t\rfloor$ and $\mu[n]\le\lceil t\rceil-t$, then $m_1[n]=\mu[n]-(t-\lfloor t\rfloor)$, $m_2[n]=0$, and
$$m_1[n]+m_2[n]=\mu[n]-(t-\lfloor t\rfloor)<\mu[n]. \qquad (78)$$
3) If $\mu[n]\le t-\lfloor t\rfloor$ and $\mu[n]>\lceil t\rceil-t$, then $m_1[n]=0$, $m_2[n]=\mu[n]-(\lceil t\rceil-t)$ and
$$m_1[n]+m_2[n]=\mu[n]-(\lceil t\rceil-t)<\mu[n]. \qquad (79)$$
4) If $\mu[n]>t-\lfloor t\rfloor$ and $\mu[n]>\lceil t\rceil-t$, then
$$m_1[n]+m_2[n]\overset{(a)}{=}2\mu[n]-1\overset{(b)}{\le}\mu[n], \qquad (80)$$
where (a) is because $(t-\lfloor t\rfloor)+(\lceil t\rceil-t)=1$ and (b) is because $\mu[n]\le 1$.

By observing cases 1 through 4, $m_1[n]+m_2[n]=\mu[n]$ if $\mu[n]=0$, as shown in case 1, or if $\mu[n]=1$ as shown in case 4, and otherwise $m_1[n]+m_2[n]<\mu[n]$. Therefore, $m_1[n]+m_2[n]=\mu[n]$ *if and only if* $\mu[n]\in\{0,1\}$. This completes the proof of Lemma 3.

B. Proof of Lemma 4

First, we show that the denominator of (50) is strictly positive. By Lemma 3, $m_1[n]+m_2[n]\le\mu[n]$ for all $n\in[N]$, therefore

$$\begin{aligned}
&t-\sum_{n=1}^{N}m_1[n]-\sum_{n=1}^{N}m_2[n] \\
&\qquad=\sum_{n=1}^{N}(\mu[n]-m_1[n]-m_2[n])\ge 0. \qquad (81)
\end{aligned}$$

Furthermore, equality holds in (81) *if and only if* $\mu[n]=m_1[n]+m_2[n]$ for all $n\in[N]$. By Lemma 3, we obtain $\mu[n]\in\{0,1\}$ for all $n\in[N]$, which means that in this case t is an integer (violating our assumption of non-integer t). Hence, we conclude that the denominator of (50) is strictly positive.

Next, the numerator of (50) is strictly less than the denominator of (50), which is shown as follows. First, we can see that

$$\mu[n]\Big(\lceil t\rceil-t\Big)\overset{(a)}{\le}\mu[n]-\Big[\mu[n]-(\lceil t\rceil-t)\Big]^{+}=\mu[n]-m_2[n], \qquad (82)$$

where (a) is because $\mu[n]\le 1$ and $\lceil t\rceil-t<1$. Hence, we obtain

$$\begin{aligned}
\lfloor t\rfloor(\lceil t\rceil-t)<t(\lceil t\rceil-t)&=\sum_{n=1}^{N}\mu[n](\lceil t\rceil-t) \\
&\le\sum_{n=1}^{N}(\mu[n]-m_2[n])=t-\sum_{n=1}^{N}m_2[n] \qquad (83)
\end{aligned}$$

which implies the numerator of (50) is strictly less than the denominator of (50).

Finally, we need to show that the numerator of (50) is non-negative. Let $v\in\mathbb{Z}^{+}$ be the number of storage requirements which are greater than or equal to $t-\lfloor t\rfloor$,

$$v=\sum_{n=1}^{N}\mathbb{1}\left(\mu[n]\ge t-\lfloor t\rfloor\right). \qquad (84)$$

Given (84), we establish two upper bounds on $\sum_{n=1}^{N}m_1[n]$. The first is given by

$$\sum_{n=1}^{N}m_1[n]\le v(\lceil t\rceil-t). \qquad (85)$$

[7]Note that, when $\mu[n]\in\{0,1\}$ for all $n\in[N]$, t is an integer, which is not the scenario of interest in this section.

[8]It can be shown for $a,b\in\mathbb{R}$ that $a-[a-b]^{+}\le b$ by considering the cases of $a\le b$ and $a>b$.

This holds because for any n such that $\mu[n] \geq t - \lfloor t \rfloor$, we have

$$m_1[n] = \mu[n] - (t - \lfloor t \rfloor) \leq 1 - (t - \lfloor t \rfloor) = \lceil t \rceil - t \quad (86)$$

and there are v such n's. For any other n such that $\mu[n] < t - \lfloor t \rfloor$, we have $m_1[n] = 0$, which does not contribute to the summation of (85). The second upper bound is given by

$$\sum_{n=1}^{N} m_1[n] \leq t - v(t - \lfloor t \rfloor). \quad (87)$$

This holds because the cumulative storage of these v DBs is $v(t-\lfloor t \rfloor)+\sum_{n=1}^{N} m_1[n]$ and cannot exceed t. It can be shown that when $v < t$, (85) is a tighter bound, and when $v > t$, (87) is a tighter bound.[9] Then by finding the integer v in each region which gives the largest upper bound, we find

$$\sum_{n=1}^{N} m_1[n] \leq \lfloor t \rfloor (\lceil t \rceil - t), \text{ for } v < t, \quad (88)$$

$$\sum_{n=1}^{N} m_1[n] \leq t - \lceil t \rceil (t - \lfloor t \rfloor), \text{ for } v > t. \quad (89)$$

Since $\lfloor t \rfloor (\lceil t \rceil - t) = t - \lceil t \rceil (t - \lfloor t \rfloor)$, for general v, we conclude that $\sum_{n=1}^{N} m_1[n] \leq \lfloor t \rfloor (\lceil t \rceil - t)$ and the numerator of (50) is non-negative. Therefore, we have shown that $0 \leq r < 1$ and this completes the proof of Lemma 4.

ACKNOWLEDGMENT

Work supported through the National Science Foundation grants CCF-1817154 and SpecEES-1824558 and the Idaho National Laboratory (INL) Laboratory Directed Research and Development (LDRD) Program under DOE Idaho Operations Office Contract DE-AC07-05ID14517.

REFERENCES

[1] N. Woolsey, R. Chen, and M. Ji, "A new design of private information retrieval for storage constrained databases," in *2019 IEEE International Symposium on Information Theory (ISIT)*, July 2019, pp. 1052–1056.

[2] N. Woolsey, R. Chen, and M. Ji, "An optimal iterative placement algorithm for pir from heterogeneous storage-constrained databases," in *GLOBECOM 2019-2019 IEEE Global Communications Conference*. IEEE, 2019, pp. 1–6.

[3] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, "Private information retrieval," in *Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on*. IEEE, 1995, pp. 41–50.

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

[5] H. Sun and S. A. Jafar, "The capacity of private information retrieval," *IEEE Transactions on Information Theory*, vol. 63, no. 7, pp. 4075–4088, 2017.

[6] Chao Tian, Hua Sun, and Jun Chen, "Capacity-achieving private information retrieval codes with optimal message size and upload cost," *IEEE Transactions on Information Theory*, vol. 65, no. 11, pp. 7613–7627, 2019.

[7] H. Sun and S. A. Jafar, "Optimal download cost of private information retrieval for arbitrary message length," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 12, pp. 2920–2932, 2017.

[8] R. Tandon, M. Abdul-Wahid, F. Almoualem, and D. Kumar, "PIR from storage constrained databases-coded caching meets PIR," in *2018 IEEE International Conference on Communications (ICC)*. IEEE, 2018, pp. 1–7.

[9] Y.-P. Wei, B. Arasli, K. Banawan, and S. Ulukus, "The capacity of private information retrieval from decentralized uncoded caching databases," *arXiv preprint arXiv:1811.11160*, 2018.

[10] M. A. Attia, D. Kumar, and R. Tandon, "The capacity of private information retrieval from uncoded storage constrained databases," *arXiv preprint arXiv:1805.04104*, 2018.

[11] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *Information Theory, IEEE Transactions on*, vol. 60, no. 5, pp. 2856–2867, 2014.

[12] K. Banawan and S. Ulukus, "The capacity of private information retrieval from coded databases," *IEEE Transactions on Information Theory*, vol. 64, no. 3, pp. 1945–1956, March 2018.

[13] K. Banawan, B. Arasli, and S. Ulukus, "Improved storage for efficient private information retrieval," *arXiv preprint arXiv:1908.11366*, 2019.

[14] C. Tian, H. Sun, and J. Chen, "A shannon-theoretic approach to the storage-retrieval tradeoff in pir systems," in *2018 IEEE International Symposium on Information Theory (ISIT)*, June 2018, pp. 1904–1908.

[15] K. Banawan, B. Arasli, Y.-P. Wei, and S. Ulukus, "The capacity of private information retrieval from heterogeneous uncoded caching databases," *IEEE Transactions on Information Theory*, 2020.

[16] M. A. Maddah-Ali and U. Niesen, "Decentralized coded caching attains order-optimal memory-rate tradeoff," *Networking, IEEE/ACM Transactions on*, vol. 23, no. 4, pp. 1029–1040, Aug 2015.

[9]Note that, v is an integer and t is assumed to be a non-integer, therefore the case of $v = t$ is not considered.

Nicholas Woolsey (S'17) is a Ph.D. student in the Department of Electrical and Computer Engineering at University of Utah. His research interests include combinatoric designs and algorithms for resource allocation, coding and efficient communications in distributed computing, private and caching networks. From 2014 to 2017, he was an Electrical Engineer at Northrop Grumman Corporation (NGC), Ogden, UT developing test and evaluation methods, modernization solutions and signal processing algorithms for the sustainment of aging aircraft and ground communication systems. While at NGC, he received the "Outside the Box" Grant to investigate the design of a modern receiver that interfaces aging technology and the 2016 Brent Scowcroft Team Award for performing exceptional systems engineering work. He received a B.S. degree in Biomedical Engineering from University of Connecticut in 2012 and M.Eng. degree in Bioengineering from University of Maryland, College Park in 2015 with a focus on signal processing, imaging and optics.

Rong-Rong Chen received the B.S. degree in Applied Mathematics from Tsinghua University, P.R. China in 1993, and the M.S. degree in Mathematics and the Ph.D. degree in Electrical and Computer Engineering from the University of Illinois at Urbana-Champaign in 1995, and 2003, respectively. She was an Assistant Professor at the University of Utah from 2003-2011 and has been an Associate Professor from 2011 to present. Her main research interests are in the area of communication systems and networks, with current emphasis on distributed computing, machine learning, caching networks, statistical signal processing, image reconstructions, and channel coding. She was the recipient of the M. E. Van Valkenburg Graduate Research Award for excellence in doctoral research in the ECE department at the University of Illinois at Urbana-Champaign in 2003. She was a recipient of the prestigious National Science Foundation Faculty Early Career Development (CAREER) award in 2006. She was rated among the Top 15% Instructors of College of Engineering at University of Utah in 2018 and 2017. She has served as an associate editor for IEEE Transactions on Signal Processing and as a guest editor of IEEE Journal on Selected Topics in Signal Processing. She has served on the technical program committees of leading international conferences in wireless communication and networks.

Mingyue Ji (S'09-M'15) received the B.E. in Communication Engineering from Beijing University of Posts and Telecommunications (China), in 2006, the M.Sc. degrees in Electrical Engineering from Royal Institute of Technology (Sweden) and from University of California, Santa Cruz, in 2008 and 2010, respectively, and the PhD from the Ming Hsieh Department of Electrical Engineering at University of Southern California in 2015. He subsequently was a Staff II System Design Scientist with Broadcom Corporation (Broadcom Limited) in 2015-2016. He is now an Assistant Professor of Electrical and Computer Engineering Department and an Adjunct Assistant Professor of School of Computing at the University of Utah. He received the IEEE Communications Society Leonard G. Abraham Prize for the best IEEE JSAC paper in 2019, the best paper award in IEEE ICC 2015 conference, the best student paper award in IEEE European Wireless 2010 Conference and USC Annenberg Fellowship from 2010 to 2014. He is interested the broad area of information theory, coding theory, concentration of measure and statistics with the applications of caching networks, wireless communications, distributed computing and storage, security and privacy and (statistical) signal processing.