Private Read Update Write (PRUW) with Storage Constrained Databases

Sajani Vithana   Sennur Ulukus
Department of Electrical and Computer Engineering
University of Maryland, College Park, MD 20742
spallego@umd.edu   *ulukus@umd.edu*

***Abstract*—We investigate the problem of private read update write (PRUW) in relation to federated submodel learning (FSL) with storage constrained databases. In PRUW, a user privately reads a submodel from a system of N databases containing M submodels, updates it locally, and writes the update back to the databases without revealing the submodel index or the value of the update. The databases considered in this problem are only allowed to store a given amount of information specified by an arbitrary storage constraint. We provide a storage mechanism that determines the contents of each database prior to the application of the PRUW scheme, such that the total communication cost is minimized. We show that the proposed storage scheme achieves a lower total cost compared to what is achieved by using *coded storage* or *divided storage* to meet the given storage constraint.**

I. Introduction

Federated submodel learning (FSL) [1]–[7] is a form of federated learning (FL) [8]–[12] where the model is divided into multiple submodels so that each user is able to download and update only the specific submodel(s) that can be trained by the user's local data. FSL consists of two phases in communication, namely, the reading phase in which the users download the required submodel, and the writing phase in which the users upload the generated update. Original FL which requires the users to download and update the entire model is inefficient compared to FSL in cases where the users do not have the types of data suitable to train the entire model. Although FSL is efficient in terms of communication cost and processing power of local users, it introduces an important issue with respect to user privacy. The submodel that a given user updates may leak information on the type of data the user has. Moreover, the values of the updates uploaded by a user may leak information about the local data of the user, as in FL [13]–[22]. Consequently, in order to guarantee the privacy of a given user, the index of the updating submodel as well as the values of the updates must be kept private from databases. The combined process of privately reading, updating and writing is known as private read update write (PRUW).

Existing works [1], [2], [4]–[7] provide PRUW schemes with different notions of privacy [23], [24]. PRUW with information-theoretic privacy is equivalent to the problem of private information retrieval (PIR) in the reading phase, see e.g., [25]–[53]. The PRUW scheme that is based on information-theoretic privacy, with the lowest known total cost (reading+writing) so far, is presented in [6] and [5]. This scheme requires N databases that store ML bits where M is the number of submodels and L is the size of a submodel. However, in practice, the available databases may not have the capacity to store ML bits since the model sizes can be large in general. Thus, the existing schemes cannot be directly defined on databases with given storage limitations. In this work, we investigate efficient storage mechanisms that can be used for PRUW on databases with given storage constraints.

The works most closely related to ours are [4]–[6], [27]–[30]. FSL with coded/uncoded storage is studied in [4]–[6] under general privacy and security constraints. For storage constrained PIR, [27] presents a PIR scheme with *coded storage*, while [28], [29] provide uncoded schemes with *divided storage*. A scheme that utilizes *both coded and divided* storage in order to minimize the storage cost is presented in [30].

In this paper, we investigate the problem of PRUW with homogeneous storage constrained databases, where the common storage capacity of each database is specified as a fraction μ of ML. PRUW with storage constrained databases consists of two main stages: 1) determining the contents of each database, 2) employing a PRUW scheme. In this work, we focus on the first phase of determining the storage of each database such that the optimized version of the PRUW scheme in [6] can be applied in the next stage while satisfying the given storage constraint. The storage mechanism we propose takes the given storage constraint μ as an input and determines the contents of each database prior to the read/write process, such that the total communication cost (reading+writing) is minimized.

A given storage constraint can be met by *dividing* the submodels according to [28], [29] and storing subsets of them in each database, or by utilizing *coded* storage [6]. In this work, we provide a *hybrid* storage mechanism that combines both coding and dividing. We show that the hybrid mechanism achieves a lower total cost compared to each of the two individual mechanisms for any given storage constraint μ. We also provide lower bounds on the achievable total costs of each individual storage mechanism, and illustrate how the proposed hybrid scheme achieves a lower total cost.

II. Problem Formulation

We consider a FL model consisting of M independent submodels, each containing L bits, stored in a system of N, $N \geq 4$, non-colluding databases. Each database has a storage

This work was supported by ARO Grant W911NF2010142, and NSF Grants CCF 17-13977 and ECCS 18-07348.

capacity of μML bits, where $\mu \in \left[\frac{1}{N-3}, 1\right]$. Each database must satisfy $H(S_n) \leq \mu ML$, where S_n is the content of database n, $n \in \{1, \dots, N\}$. At any given time instance, a single user reads, updates and writes a single submodel of interest, while keeping the submodel index and the value of the update private from all databases and other users.

Privacy of the submodel index: No information on the index of the submodel being updated θ is allowed to leak to any of the databases, i.e., for each n,

$$I(\theta^{[t]}; Q_n^{[t]}, U_n^{[t]} | Q_n^{[1:t-1]}, S_n^{[1:t-1]}, U_n^{[1:t-1]}) = 0, \tag{1}$$

where $Q_n^{[t]}$ and $U_n^{[t]}$ are the query and update sent by the user to database n at time t in the reading and writing phases.

Privacy of the value of the update: No information on the value of the update is allowed to leak to any of the databases, i.e., for each n,

$$I(\Delta_\theta^{[t]}; U_n^{[t]} | Q_n^{[1:t]}, S_n^{[1:t-1]}, U_n^{[1:t-1]}) = 0, \tag{2}$$

where Δ_θ is the update generated by the user.

Security of submodels: No information on the submodels is allowed to leak to any of the databases, i.e., for each n,

$$I(\mathbf{W}_{1:M}^{[t]}; S_n^{[t]}) = 0, \tag{3}$$

where $\mathbf{W}_k^{[t]}$ is the kth submodel at time t.

Correctness in the reading phase: The user should be able to correctly decode the required submodel from the answers received in the reading phase, i.e.,

$$H(\mathbf{W}_\theta^{[t-1]} | Q_{1:N}^{[t]}, A_{1:N}^{[t]}) = 0, \tag{4}$$

where $A_n^{[t]}$ is the answer from database n at time t.

Correctness in the writing phase: Each parameter i, $i \in \{1, \dots, L\}$ of $\mathbf{W}_\theta$ must be correctly updated at time t as,

$$\mathbf{W}_{\theta,i}^{[t]} = \mathbf{W}_{\theta,i}^{[t-1]} + \Delta_{\theta,i}. \tag{5}$$

The reading and writing costs are defined as $C_R = \frac{\mathcal{D}}{L}$ and $C_W = \frac{\mathcal{U}}{L}$, respectively, where $\mathcal{D}$ is the total number of bits downloaded in the reading phase and $\mathcal{U}$ is the total number of bits uploaded in the writing phase. The total cost C_T is the sum of the reading and writing costs, i.e., $C_T = C_R + C_W$.

III. Main Result

In this section, we present the achievable total cost of the proposed storage mechanism. Let $C_T(\mu)$ be the total cost corresponding to the storage constraint μ.

Theorem 1 *For any given N and $\mu \in \left[\frac{1}{N-3}, 1\right]$, the achievable total cost of the proposed scheme is given by the boundary of the lower convex hull of $(\mu, C_T(\mu))$ pairs given by,*

$$\left(\mu = \frac{r}{NK_r}, C_T(\mu) = \frac{4r}{r - K_r - 1}\right), \quad r = 4, \dots, N,$$
$$K_r = 1, \dots, r-3, \ \textit{s.t.}\ (r - K_r - 1) \mod 2 = 0. \tag{6}$$

Remark 1 *The two main constrained storage mechanisms (divided storage and coded storage) are subsets of the proposed hybrid storage mechanism. Divided storage corresponds to cases where r is even in $\{4, \dots, N\}$ and $K_r = 1$, while coded storage corresponds to $r = N$ and $K_r \in \{1, \dots, N-3\}$, with $(N - K_r - 1) \mod 2 = 0$. Since the achievable total cost of the proposed scheme is characterized by the boundary of the lower convex hull of all achievable points, the total cost of the proposed scheme is less than or equal to that of the two main storage mechanisms for each μ.*

IV. Proposed Storage Mechanism

In this section, we present the proposed storage mechanism that specifies the specific storage in each database according to the given storage constraint μ, prior to the application of the PRUW scheme. First, we present the following lemma, which is a crucial component of the proposed scheme.

Lemma 1 *Let $(\mu_1, C_T(\mu_1))$ and $(\mu_2, C_T(\mu_2))$ be two pairs of storage constraints and corresponding achievable total costs. Then, the pair $(\mu, C_T(\mu))$ is also achievable for any $\gamma \in [0,1]$ where,*

$$\mu = \gamma\mu_1 + (1-\gamma)\mu_2 \tag{7}$$
$$C_T(\mu) = \gamma C_T(\mu_1) + (1-\gamma) C_T(\mu_2) \tag{8}$$

Proof: Since $(\mu_1, C_T(\mu_1))$ and $(\mu_2, C_T(\mu_2))$ are achievable, let S_1 and S_2 be the schemes that produce the achievable pairs $(\mu_1, C_T(\mu_1))$ and $(\mu_2, C_T(\mu_2))$, respectively. A new scheme can be generated by applying S_1 on a γ fraction of bits of all submodels and S_2 on the rest of the bits. The storage of this scheme is, $\gamma ML\mu_1 + (1-\gamma)ML\mu_2 = \mu ML$ bits. The corresponding total cost of the combined scheme is

$$C_T = \frac{\gamma L C_T(\mu_1) + (1-\gamma) L C_T(\mu_2)}{L} = \gamma C_T(\mu_1) + (1-\gamma) C_T(\mu_2), \tag{9}$$

completing the proof. ■

Next, we need to obtain the optimized version of the PRUW scheme in [6] in order to determine the contents of each database, on which the scheme is applied.

A. Optimized PRUW Scheme

The contents of a single subpacket in database n, $n \in \{1, \dots, r\}$ of the scheme in [6] with a subpacketization of y and $x+1$ noise terms with no dropouts is given by

$$\mathbf{S}_n = \begin{bmatrix} \sum_{i=1}^{K} \frac{1}{f_{1,i} - \alpha_n} \begin{bmatrix} \mathbf{W}_{1,1}^{[i]} \\ \vdots \\ \mathbf{W}_{M,1}^{[i]} \end{bmatrix} + \sum_{j=0}^{x} \alpha_n^j \mathbf{I}_{1,j} \\ \vdots \\ \sum_{i=1}^{K} \frac{1}{f_{y,i} - \alpha_n} \begin{bmatrix} \mathbf{W}_{1,y}^{[i]} \\ \vdots \\ \mathbf{W}_{M,y}^{[i]} \end{bmatrix} + \sum_{j=0}^{x} \alpha_n^j \mathbf{I}_{y,j} \end{bmatrix}, \tag{10}$$

with random noise vectors $\mathbf{I}_{i,j}$ (for security of submodels [24]) and distinct constants $f_{i,j}, \alpha_n$. In the reading phase, the user sends queries $\mathbf{Q}_{n,\ell}$, $\ell \in \{1,\dots,K\}$ to retrieve each of $\mathbf{W}_{[\theta],1}^{[1]},\dots,\mathbf{W}_{[\theta],y}^{[K]}$, where θ is the required submodel index,

$$\mathbf{Q}_{n,\ell} = \begin{bmatrix} \frac{\prod_{i=1,i\neq\ell}^{K}(f_{1,i}-\alpha_n)}{\prod_{i=1,i\neq\ell}^{K}(f_{1,i}-f_{1,\ell})}\mathbf{e}_M(\theta)+\prod_{i=1}^{K}(f_{1,i}-\alpha_n)\mathbf{Z}_{1,\ell} \\ \vdots \\ \frac{\prod_{i=1,i\neq\ell}^{K}(f_{y,i}-\alpha_n)}{\prod_{i=1,i\neq\ell}^{K}(f_{y,i}-f_{y,\ell})}\mathbf{e}_M(\theta)+\prod_{i=1}^{K}(f_{y,i}-\alpha_n)\mathbf{Z}_{y,\ell} \end{bmatrix} \tag{11}$$

where $\mathbf{Z}_{i,j}$ are random noise vectors. The databases send the answers $A_{n,\ell}$, $\ell \in \{1,\dots,K\}$ given by,

$$A_{n,\ell} = \mathbf{S}_n^T \mathbf{Q}_{n,\ell} = \sum_{i=1}^{y} \frac{1}{f_{i,\ell}-\alpha_n}\mathbf{W}_{\theta,i}^{[\ell]} + \sum_{j=0}^{K+x} \alpha_n^j \tilde{\mathbf{I}}_j, \tag{12}$$

where $\tilde{\mathbf{I}}_i$ are combinations of random noise terms.[1] Using the answers of all r databases, $\{\mathbf{W}_{\theta,1}^{[\ell]},\dots,\mathbf{W}_{\theta,y}^{[\ell]}\}$ can be obtained for each ℓ if $r = y+x+K+1$. The resulting reading cost is

$$C_R = \frac{Kr}{Ky} = \frac{r}{r-x-K-1}. \tag{13}$$

In the writing phase, the user sends K bits to each of the r databases, which are linear combinations of y update bits,

$$U_{n,\ell} = \sum_{j=1}^{y} \prod_{i=1,i\neq j}^{y} (f_{i,\ell}-\alpha_n)\tilde{\Delta}_{\theta,j}^{[\ell]} + \prod_{i=1}^{y}(f_{i,\ell}-\alpha_n)\hat{z}_\ell, \tag{14}$$

where $\tilde{\mathbf{\Delta}}_{\theta,j}^{[\ell]} = \frac{\prod_{i=1,i\neq\ell}^{K}(f_{j,i}-f_{j,\ell})}{\prod_{i=1,i\neq j}^{y}(f_{i,\ell}-f_{j,\ell})}\mathbf{\Delta}_{\theta,j}^{[\ell]}$ for $j \in \{1,\dots,y\}$. Once database n receives the update bits, it calculates the incremental update with the aid of the two matrices given by,

$$\Omega_{n,\ell} = \text{diag}\left(\frac{\prod_{r\in\mathcal{F}}(\alpha_r-\alpha_n)}{\prod_{r\in\mathcal{F}}(\alpha_r-f_{1,\ell})}\mathbf{1}_M,\dots,\frac{\prod_{r\in\mathcal{F}}(\alpha_r-\alpha_n)}{\prod_{r\in\mathcal{F}}(\alpha_r-f_{y,\ell})}\mathbf{1}_M\right) \tag{15}$$

$$\tilde{\mathbf{Q}}_{n,\ell} = \text{diag}\left(\frac{1}{\prod_{i=1}^{K}(f_{1,i}-\alpha_n)}\mathbf{1}_M,\dots,\frac{1}{\prod_{i=1}^{K}(f_{y,i}-\alpha_n)}\mathbf{1}_M\right) \times \mathbf{Q}_{n,\ell}, \tag{16}$$

where $\Omega_{n,\ell}$ is the null shaper in [6] with $|\mathcal{F}| = x-y$ and $\tilde{\mathbf{Q}}_{n,\ell}$ is the scaled query vector. $\mathbf{1}_M$ is the vector of all ones of size $1 \times M$. The incremental update is calculated as,

$$\bar{U}_{n,\ell} = \Omega_{n,\ell} \times U_{n,\ell} \times \tilde{\mathbf{Q}}_{n,\ell} \tag{17}$$

$$= \begin{bmatrix} \frac{1}{f_{1,\ell}-\alpha_n}\mathbf{\Delta}_{\theta,1}^{[\ell]}\mathbf{e}_M(\theta) + \mathbf{P}_{\alpha_n}^{[1]}(x) \\ \vdots \\ \frac{1}{f_{y,\ell}-\alpha_n}\mathbf{\Delta}_{\theta,y}^{[\ell]}\mathbf{e}_M(\theta) + \mathbf{P}_{\alpha_n}^{[y]}(x) \end{bmatrix}, \tag{18}$$

where $P_{\alpha_n}^{[j]}(i)$ is a polynomial of α_n of degree i, indexed by j. Then, the submodels are updated by $\mathbf{S}_n(t) = \mathbf{S}_n(t-1) + \sum_{\ell=1}^{K}\bar{U}_{n,\ell}$. The resulting writing and total costs are,

$$C_W = \frac{K(r-(x-y))}{Ky} = \frac{2r-2x-K-1}{r-x-K-1}, \tag{19}$$

$$C_T = C_R + C_W = \frac{3r-2x-K-1}{r-x-K-1}. \tag{20}$$

Note that the total cost is an increasing function of x since $\frac{dC_T}{dx} = \frac{r+K+1}{(r-x-K-1)^2} > 0$. Since $x \geq y$ must be satisfied by x in order to write to y parameters using a single bit, the optimum value of x that minimizes the total cost is,

$$x = \begin{cases} y = \frac{r-K-1}{2}, & \text{if } r-K-1 \text{ is even,} \\ y+1 = \frac{r-K}{2}, & \text{if } r-K-1 \text{ is odd.} \end{cases} \tag{21}$$

The resulting total costs of the two cases are,

$$C_T = \begin{cases} \frac{4r}{r-K-1}, & \text{if } r-K-1 \text{ is even,} \\ \frac{4r-2}{r-K-2}, & \text{if } r-K-1 \text{ is odd.} \end{cases} \tag{22}$$

Note that since the subpacketization $y \geq 1$, r and K must satisfy,

$$1 \leq K \leq \begin{cases} r-3, & \text{if } r-K-1 \text{ is even,} \\ r-4, & \text{if } r-K-1 \text{ is odd.} \end{cases} \tag{23}$$

B. Proposed Storage Mechanism

For a given N we first find the basic achievable pairs of $(\mu, C_T(\mu))$ as follows. Let $\mu = \frac{r}{NK_r}$ for $r = 4,\dots,N$ and $K_r = 1,\dots,r-3$. For a given μ with a given r and K_r, following steps need to be followed in order to perform PRUW while meeting the storage constraint:

1) Divide the L bits of each submodel into N sections and label them as $\{1,\dots,N\}$.
2) Allocate sections $n : (n-1+r) \bmod N$ to database n for $n \in \{1,\dots,N\}$.[2]
3) Use the storage specified in (10) with $K = K_r$ and x, y given in (21) to encode each of the allocated sections of all submodels. Note that a given coded bit of a given section of each submodel stored across different databases contains the same noise polynomial that only differs in α_n.
4) Use the PRUW scheme described in Section IV-A on each of the subsets of $n : (n-1+r) \bmod N$ databases to read/write to section $(n-1+r) \mod N$ of the required submodel for $n \in \{1,\dots,N\}$.

For each $\mu = \frac{r}{NK_r}$, $r = 4,\dots,N$, $K_r = 1,\dots,r-3$, the above process gives an achievable $(\mu, C_T(\mu))$ pair, where $C_T(\mu)$ is given as follows using (22),

$$C_T(\mu) = \begin{cases} \frac{4r}{r-K_r-1}, & \text{if } r-K_r-1 \text{ is even,} \\ \frac{4r-2}{r-K_r-2}, & \text{if } r-K_r-1 \text{ is odd.} \end{cases} \tag{24}$$

Note that the above two cases, which correspond to the value of $(r-K_r-1) \bmod 2$, are a result of two different schemes. The case with even values of $r-K_r-1$ has a subpacketization that is equal to the degree of noise polynomial in storage, which does not require the null shaper, while the case with odd values of $r-K_r-1$ contains two more noise terms than the subpacketization, which requires the null shaper; see

[1] Note that the terms corresponding to $W_{\theta,i}^{[j\neq\ell]}$ for $i = 1,\dots,y$ and $j = 1,\dots,K$ in the calculation of the answers in (12) are included in the combined noise terms $\tilde{I}_j$ (aligned along the noise subspace).

[2] The indices here follow a cyclic pattern, i.e., if $(n-1+r) \bmod N < n$, $n : (n-1+r) \bmod N$ implies $\{n,\dots,N,1,\dots,(n-1+r) \bmod N\}$.

(21). The scheme corresponding to odd values of $r-K_r-1$ is inefficient compared to the even case due to the additional noise term present in storage. This observation combined with Lemma 1 results in the following lemma.

Lemma 2 *For a given $\mu=\frac{r}{NK_r}$, if r and K_r are such that $r-K_r-1$ is odd, it is more efficient to perform a linear combination of two PRUW schemes with nearest two even $r^{[i]}-K_r^{[i]}-1$, $i=1,2$, instead of performing direct PRUW with the given r and K_r, while satisfying the same storage constraint μ, i.e., with $\mu_1=\frac{r^{[1]}}{NK_r^{[1]}}$ and $\mu_2=\frac{r^{[2]}}{NK_r^{[2]}}$.*

Proof: For a given $\mu=\frac{r}{NK_r}$, the nearest $\mu_1=\frac{r^{[1]}}{NK_r^{[1]}}$ is $\frac{r-1}{NK_r}$, since (23) with K replaced by K_r needs to be satisfied for the PRUW scheme to work. Similarly, $\mu_2=\frac{r+1}{NK_r}$. Let $C_T(\mu)$, $C_T(\mu_1)$ and $C_T(\mu_2)$ be the total costs incurred by the scheme with μ, μ_1 and μ_2, respectively. From (24), we have,

$$C_T(\mu)=\frac{4r-2}{r-K_r-2}>C_T(\mu_1)=\frac{4r-4}{r-K_r-2}>C_T(\mu_2)=\frac{4(r+1)}{r-K_r}. \tag{25}$$

Note that $\mu_1<\mu<\mu_2$. From Lemma 1, there exists some $\gamma\in[0,1]$ that allocates the storage for the two PRUW schemes corresponding to μ_1 and μ_2 that achieves the same storage constraint as μ, and results in a total cost of $\gamma C_T(\mu_1)+(1-\gamma)C_T(\mu_2)$, that satisfies,

$$C_T(\mu_2)<\gamma C_T(\mu_1)+(1-\gamma)C_T(\mu_2)<C_T(\mu_1)<C_T(\mu), \tag{26}$$

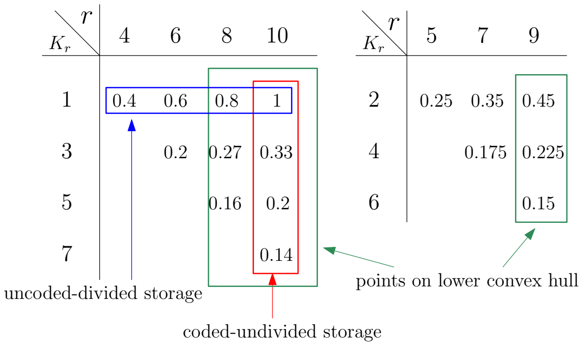
completing the proof. ■

Once the basic $(\mu, C_T(\mu))$ pairs corresponding to $\mu=\frac{r}{NK_r}$ for $r=4,\dots,N$, $K_r=1,\dots,r-3$ with $(r-K_r-1) \mod 2=0$ are obtained, the achievable total cost of the proposed scheme for any μ is characterized by the boundary of the lower convex hull of the above basic $(\mu, C_T(\mu))$ pairs, using Lemma 1, denoted by T_{ach}. Therefore, for a given N and μ, the proposed PRUW storage mechanism is obtained by utilizing the correct linear combination of PRUW schemes that correspond to the nearest two basic $(\mu, C_T(\mu))$ pairs on T_{ach}. The resulting total cost is $T_{ach}(\mu)$.[3]

V. Lower Bounds on Achievable Costs

In this section, we provide lower bounds on the achievable costs derived in Section IV-B. Based on Lemma 2 and (24), for a given K_r, the achievable total cost of the proposed scheme for a given N and $\mu\leq\frac{1}{K_r}$ can be lower bounded as,

$$C_T(\mu,K_r)\geq\frac{4r}{r-K_r-1}=\frac{4N\mu}{N\mu-1-\frac{1}{K_r}}=LB(\mu,K_r), \tag{27}$$



K_r \ r	4	6	8	10
1	0.4	0.6	0.8	1
3		0.2	0.27	0.33
5			0.16	0.2
7				0.14

K_r \ r	5	7	9
2	0.25	0.35	0.45
4		0.175	0.225
6			0.15

Fig. 1: All possible pairs of (r, K_r) and corresponding values of μ for $N=10$.

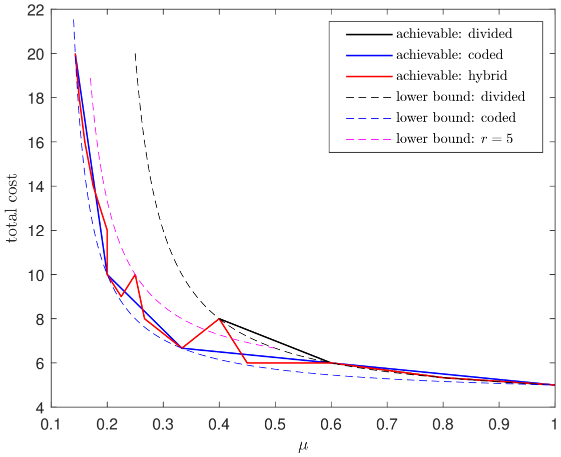


Fig. 2: Achievable total costs and lower bounds of divided, coded and hybrid schemes for $N=10$, before obtaining the convex hull boundary.

since $LB(\mu,K_r)$ is a convex function of μ and the achievable total cost is a piecewise linear function with points corresponding to $\mu=\frac{r}{NK_r}$ with $(r-K_r-1) \mod 2=0$ on $LB(\mu,K_r)$. Similarly, for a given r, the achievable total cost of any $\mu\leq\frac{r}{N}$ is lower bounded as,

$$C_T(\mu,r)\geq\frac{4r}{r-K_r-1}=\frac{4N\mu}{N\mu-1-\frac{N\mu}{r}}=LB(\mu,r). \tag{28}$$

Note that the two storage mechanisms defined by divided storage ($K_r=1$, $r<N$) and coded storage ($K_r>1$, $r=N$) are subsets of the proposed storage mechanism where $K_r\geq1$ and $r\leq N$. The total cost of the divided storage mechanism is lower bounded by $LB_d(\mu)=\frac{4N\mu}{N\mu-2}$, while that of coded storage is lower bounded by $LB_c(\mu)=\frac{4N\mu}{N\mu-1-\mu}$. Clearly, the lower bound of the coded scheme is less than that of the divided storage scheme except at $\mu=1$, where the two bounds are the same. For all other cases ($K_r>1$, $r<N$), the lower bounds $LB(\mu,r)$ and $LB(\mu,K_r)$ satisfy,

$$LB_c(\mu)\leq LB(\mu,K_r), \quad LB(\mu,r)\leq LB_d(\mu), \tag{29}$$

For each $\mu\in\left[\frac{1}{N-3},1\right]$: Even though the coded scheme is better in terms of the lower bounds, the achievable costs

[3] The proposed storage scheme can be directly extended for the case with colluding databases (X-security of submodels T-privacy of queries and V-privacy of updates) by ensuring that the number of noise terms added to storage (10), queries (11) and updates (14) are given by $\max\{X,\lceil\frac{V+r-K_r-2}{2}\rceil\}$, T and V, respectively. All lemmas and arguments presented in this paper are still valid for colluding databases with slightly different expressions involving the extra terms X, T and V.

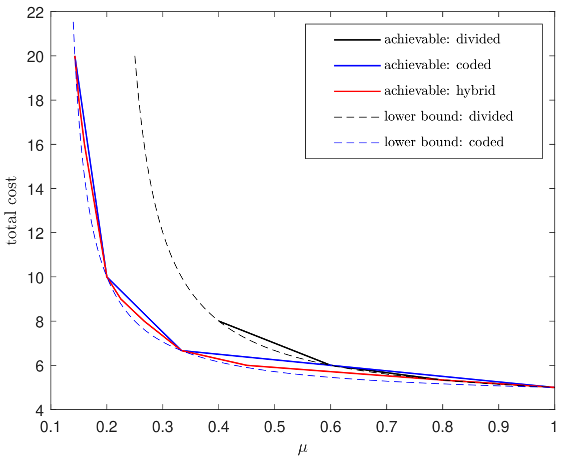
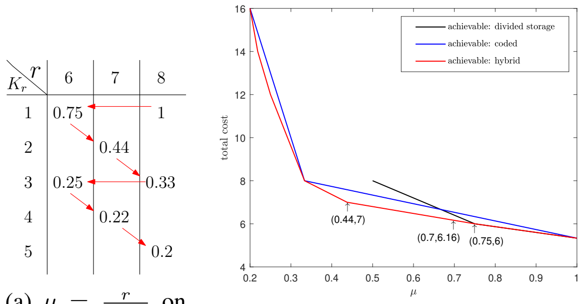


Fig. 3: Lowest achievable costs of coded, divided and hybrid schemes for $N = 10$.

show that the divided storage scheme performs better at larger values of μ, as shown in Fig. 3 for $N = 10$. For the same example with $N = 10$, the proposed hybrid storage mechanism can be applied by first determining the basic achievable $\left(\mu = \frac{r}{NK_r}, C_T(\mu)\right)$ pairs for $r = 4, \dots, N$, $K_r = 1, \dots, r-3$ with $(r - K_r - 1) \bmod 2 = 0$. The pairs of (r, K_r) and the corresponding values of μ are shown in Fig. 1. The resulting pairs of $(\mu, C_T(\mu))$, before finding the convex hull, are shown in Fig. 2. Note that each achievable $(\mu, C_T(\mu))$ pair of the hybrid scheme lies on one of the lower bounds characterized in (27) and (28). For instance, $(0.25, 10)$ on the red curve in Fig. 2 is on the lower bound that corresponds to $r = 5$ in (28) as shown by the pink dotted line. However, since we are interested in finding the lowest possible total costs, from Lemma 1, the lowest possible total cost is characterized by the boundary of the lower convex hull of all achievable points of the hybrid scheme as shown in Fig. 3. Note that the set of basic achievable $\left(\mu = \frac{r}{NK_r}, C_T(\mu)\right)$ pairs on the lower convex hull boundary corresponds to μ's with (r, K_r) pairs with $r = N = 10$, $r = N - 1 = 9$ and $r = N - 2 = 8$ with K_r's that satisfy $(r - K_r - 1) \bmod 2 = 0$, as marked in Fig. 1.

VI. A Specific Example

In this section, we describe how the PRUW process is carried out in an arbitrary setting with given N and μ. Consider an example with $N = 8$ databases and $\mu = 0.7$. The first step is to find the basic achievable $\left(\mu = \frac{r}{NK_r}, C_T(\mu)\right)$ pairs of $N = 8$ that lie on the lower convex hull boundary. Fig. 4(a) shows the (r, K_r) pairs and the corresponding μ's of such pairs. The required storage constraint $\mu = 0.7$ is in between 0.44 and 0.75, which correspond to (r, K_r) pairs $(7, 2)$ and $(6, 1)$, respectively. Therefore, the PRUW scheme for $N = 8$, $\mu = 0.7$ is obtained by the following steps:



K_r \ r	6	7	8
1	0.75		1
2		0.44	
3	0.25		0.33
4		0.22	
5			0.2

(a) $\mu = \frac{r}{NK_r}$ on the boundary.

(b) Achievable total cost at $\mu = 0.7$.

Fig. 4: Example with $N = 8$.

1) γL bits of all submodels are stored according to the proposed storage mechanism corresponding to $(r, K_r) = (7, 2)$, and the rest of the $(1 - \gamma)L$ bits of all submodels are stored according to $(r, K_r) = (6, 1)$. Therefore, γL bits of the required submodel are updated using the scheme corresponding to $(r, K_r) = (7, 2)$, and the rest of the bits are updated by the scheme corresponding to $(6, 1)$. In order to find the value of γ, we equate the total storage of each database to the given constraint, i.e.,
$$\gamma ML \times \frac{7}{8} \times \frac{1}{2} + (1 - \gamma)ML \times \frac{6}{8} = 0.7ML \qquad (30)$$
which gives $\gamma = 0.16$.
2) Let $L_1 = 0.16L$ and $L_2 = 0.84L$. L_1 bits of each submodel is divided into 8 sections and labeled $1, \dots, 8$. Sections $n : (n + 6) \bmod 8$ are allocated to database n for $n \in \{1, \dots, N\}$. Each database uses the storage in (10) with $K = 2$ and $y = x = \frac{r-K_r-1}{2} = 2$ to store each subpacket of all sections allocated to it. Then, the PRUW scheme described in Section IV-A is applied to read/write to the L_1 bits of the required submodel.
3) The same process is carried out on the rest of the L_2 bits with the scheme corresponding to $(6, 1)$.

The total costs incurred by the two schemes are $C_{T_1} = \frac{4r}{r-K_r-1} = \frac{4\times7}{7-2-1} = 7$ and $C_{T_2} = \frac{4r}{r-K_r-1} = \frac{4\times6}{6-1-1} = 6$, respectively. Therefore, the total cost of $N = 8$ and $\mu = 0.7$ is $C_T = \frac{\gamma LC_{T_1} + (1-\gamma)LC_{T_2}}{L} = 6.16$, which is shown in Fig. 4(b).

In both examples with $N = 10$ and $N = 8$, the boundary of the lower convex hull of the achievable $(\mu, C_T(\mu))$ points was determined by simply connecting the points $(\mu = \frac{r}{NK_r}, C_T(\mu))$ with $r = N, N - 1, N - 2$ with all possible values of K_r. In general, we have the following result.

Lemma 3 *For any given N, let $T_{ach}^{[h]}$ be the piecewise linear curve obtained by connecting the achievable points of the hybrid scheme $(\mu = \frac{r}{NK_r}, C_T(\mu))$ corresponding to $r = N, N-1, N-2$, $K_r = 1, \dots, r-3$ and $(r-K_r-1) \bmod 2 = 0$. Let $T_{ach}^{[d]}$ and $T_{ach}^{[c]}$ be the minimum achievable total costs of the divided and coded schemes, respectively. Then,*
$$T_{ach}^{[h]}(\mu) \leq \min\{T_{ach}^{[d]}(\mu), T_{ach}^{[c]}(\mu)\}, \quad \forall \mu \in [\frac{1}{N-3}, 1]. \quad (31)$$

The proof of Lemma 3 is based on the geometric placement of the points $(\mu = \frac{r}{NK_r}, C_T(\mu))$ for $r = N, N-1, N-2$ with $K_r = 1, \dots, r-3$ and $(r - K_r - 1) \bmod 2 = 0$ on the lowest three bounds given in (28) along with the condition (23).

References

[1] C. Niu, F. Wu, S. Tang, L. Hua, R. Jia, C. Lv, Z. Wu, and G. Chen. Billion-scale federated learning on mobile clients: A submodel design with tunable privacy. In *MobiCom*, April 2020.

[2] M. Kim and J. Lee. Information-theoretic privacy in federated submodel learning. Available online at arXiv:2008.07656.

[3] C. Niu, F. Wu, S. Tang, L. Hua, R. Jia, C. Lv, Z. Wu, and G. Chen. Secure federated submodel learning. Available online at arXiv:1911.02254.

[4] Z. Jia and S. A. Jafar. X-secure T-private federated submodel learning. In *IEEE ICC*, June 2021.

[5] S. Vithana and S. Ulukus. Efficient private federated submodel learning. In *IEEE ICC*, May 2022.

[6] Z. Jia and S. A. Jafar. X-secure T-private federated submodel learning with elastic dropout resilience. Available online at arXiv:2010.01059.

[7] C. Naim, R. D'Oliveira, and S. El Rouayheb. Private multi-group aggregation. *IEEE ISIT*, July 2021.

[8] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication efficient learning of deep networks from decentralized data. *AISTATS*, April 2017.

[9] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, and M. Bennis *et al.* Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 14(1-2):1–210, June 2021.

[10] Q. Yang, Y. Liu, T. Chen, and Y. Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology*, 10(2):1–19, January 2019.

[11] T. Li, A. K. Sahu, A. S. Talwalkar, and V. Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37:50–60, May 2020.

[12] A. Hard, K. Rao, R. Mathews, S. Ramaswamy, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage. Federated learning for mobile keyboard prediction. Available at arXiv: 1811.03604.

[13] S. Ulukus, S. Avestimehr, M. Gastpar, S. A. Jafar, R. Tandon, and C. Tian. Private retrieval, computing and learning: Recent progress and future challenges. *IEEE Journal on Selected Areas in Communications*, 40(3):729–748, March 2022.

[14] M. Nasr, R. Shokri, and A. Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *IEEE Symposium on Security and Privacy*, May 2019.

[15] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov. Exploiting unintended feature leakage in collaborative learning. In *IEEE Symposium on Security and Privacy*, May 2019.

[16] J. Geiping, H. Bauermeister, H. Droge, and M. Moeller. Inverting gradients–how easy is it to break privacy in federated learning? Available online at arXiv:2003.14053.

[17] R. C. Geyer, T. Klein, and M. Nabi. Differentially private federated learning: A client level perspective. In *NeurIPS*, December 2017.

[18] S. Asoodeh and F. Calmon. Differentially private federated learning: An information-theoretic perspective. In *ICML-FL*, July 2020.

[19] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang. Learning differentially private recurrent language models. In *ICLR*, May 2018.

[20] K. Bonawitz, V. Ivanov, B. Kreuter, et al. Practical secure aggregation for privacy-preserving machine learning. In *CCS*, October 2017.

[21] Y. Li, T. Chang, and C. Chi. Secure federated averaging algorithm with differential privacy. *IEEE MLSE*, September 2020.

[22] N. Agarwal, A. Suresh, F. Yu, S. Kumar, and H. B. McMahan. cpSGD: Communication-efficient and differentially-private distributed SGD. In *NeurIPS*, December 2018.

[23] C. Dwork and A. Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, August 2014.

[24] C. E. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28(4):656–715, October 1949.

[25] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan. Private information retrieval. *Journal of the ACM*, 45(6):965–981, November 1998.

[26] H. Sun and S. A. Jafar. The capacity of private information retrieval. *IEEE Transactions on Information Theory*, 63(7):4075–4088, July 2017.

[27] K. Banawan and S. Ulukus. The capacity of private information retrieval from coded databases. *IEEE Transactions on Information Theory*, 64(3):1945–1956, March 2018.

[28] M. Attia, D. Kumar, and R. Tandon. The capacity of private information retrieval from uncoded storage constrained databases. *IEEE Transactions on Information Theory*, 66(11):6617–6634, November 2020.

[29] N. Woolsey, R. Chen, and M. Ji. Uncoded placement with linear sub-messages for private information retrieval from storage constrained databases. *IEEE Transactions on Communications*, 68(10):6039–6053, October 2020.

[30] K. Banawan, B. Arasli, and S. Ulukus. Improved storage for efficient private information retrieval. In *IEEE ITW*, August 2019.

[31] S. Kumar, H.-Y. Lin, et al. Achieving maximum distance separable private information retrieval capacity with linear codes. *IEEE Transactions on Information Theory*, 65(7):4243–4273, July 2019.

[32] T. Chan, S. Ho, and H. Yamamoto. Private information retrieval for coded storage. In *IEEE ISIT*, June 2015.

[33] A. Fazeli, A. Vardy, and E. Yaakobi. Codes for distributed PIR with low storage overhead. In *IEEE ISIT*, June 2015.

[34] H. Sun and S. A. Jafar. Multiround private information retrieval: Capacity and storage overhead. *IEEE Transactions on Information Theory*, 64(8):5743–5754, August 2018.

[35] 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*, 66(6):3407–3416, June 2020.

[36] Y.-P. Wei, B. Arasli, K. Banawan, and S. Ulukus. The capacity of private information retrieval from decentralized uncoded caching databases. *Information*, 10(12):372–389, December 2019.

[37] C. Tian, H. Sun, and J. Chen. Capacity-achieving private information retrieval codes with optimal message size and upload cost. *IEEE Transactions on Information Theory*, 65(11):7613–7627, November 2019.

[38] I. Samy, M. Attia, R. Tandon, and L. Lazos. Asymmetric leaky private information retrieval. *IEEE Transactions on Information Theory*, 67(8):5352–5369, August 2021.

[39] H. Sun and S. A. Jafar. The capacity of symmetric private information retrieval. *IEEE Transactions on Information Theory*, 65(1):329–322, January 2019.

[40] Z. Wang, K. Banawan, and S. Ulukus. Private set intersection: A multi-message symmetric private information retrieval perspective. *IEEE Transactions on Information Theory*, 68(3):2001–2019, March 2022.

[41] H. Sun and S. A. Jafar. The capacity of robust private information retrieval with colluding databases. *IEEE Transactions on Information Theory*, 64(4):2361–2370, April 2018.

[42] R. Tajeddine, O. W. Gnilke, D. Karpuk, R. Freij-Hollanti, and C. Hollanti. Private information retrieval from coded storage systems with colluding, Byzantine, and unresponsive servers. *IEEE Transactions on Information Theory*, 65(6):3898–3906, June 2019.

[43] S. Kadhe, B. Garcia, A. Heidarzadeh, S. El Rouayheb, and A. Sprintson. Private information retrieval with side information. *IEEE Transactions on Information Theory*, 66(4):2032–2043, April 2020.

[44] K. Banawan and S. Ulukus. Multi-message private information retrieval: Capacity results and near-optimal schemes. *IEEE Transactions on Information Theory*, 64(10):6842–6862, October 2018.

[45] M. J. Siavoshani, S. P. Shariatpanahi, and M. A. Maddah-Ali. Private information retrieval for a multi-message scenario with private side information. *IEEE Trans. on Commun.*, 69(5):3235–3244, May 2021.

[46] Q. Wang, H. Sun, and M. Skoglund. The capacity of private information retrieval with eavesdroppers. *IEEE Transactions on Information Theory*, 65(5):3198–3214, May 2019.

[47] K. Banawan and S. Ulukus. The capacity of private information retrieval from Byzantine and colluding databases. *IEEE Transactions on Information Theory*, 65(2):1206–1219, February 2019.

[48] S. Vithana, K. Banawan, and S. Ulukus. Semantic private information retrieval. *IEEE Transactions on Information Theory*, 68(4):2635–2652, April 2022.

[49] S. Li and M. Gastpar. Single-server multi-message private information retrieval with side information: the general cases. In *IEEE ISIT*, June 2020.

[50] Z. Jia and S. A. Jafar. X-secure T-private information retrieval from MDS coded storage with Byzantine and unresponsive servers. *IEEE Transactions on Information Theory*, 66(12):7427–7438, December 2020.

[51] Z. Jia and S. Jafar. On the asymptotic capacity of X-secure T-private information retrieval with graph-based replicated storage. *IEEE Transactions on Information Theory*, 66(10):6280–6296, October 2020.

[52] J. Xu and Z. Zhang. Building capacity-achieving PIR schemes with optimal sub-packetization over small fields. In *IEEE ISIT*, June 2018.

[53] H. Yang, W. Shin, and J. Lee. Private information retrieval for secure distributed storage systems. *IEEE Transactions on Information Forensics and Security*, 13(12):2953–2964, December 2018.