On Demand-Private Hotplug Caching Systems

Yinbin Ma and Daniela Tuninetti
University of Illinois Chicago, Chicago, IL 60607, USA
Email:{yma52, danielat}@uic.edu

***Abstract*—Maddah-Ali and Niesen (MAN) showed that coded caching can effectively mitigate peak-time network load by leveraging local caches at the end users. In the MAN model, a server stores multiple files, populates the local caches, and serves single-file requests from cache-aided users via an error-free shared link. The delivered multicast messages in the MAN scheme to serve the users' requests require all users to stay active and expose demand information. To preserve demand privacy and accommodate for offline users, the hotplug caching model with demand privacy against colluding users was introduced and an achievable schemes based on the "privacy keys" idea was proposed at ISIT 2023. This ISIT 2024 paper proposes three new achievable schemes for this model which use Maximum Distance Separable codes coupled with the "virtual users" idea. These new schemes turn out to be exactly optimal under certain conditions, and otherwise optimal to within a constant factor.**

***Index Terms*—Coded caching with offline users; Achievablity; Demand privacy; Colluding users; Optimality for small memory size; Multiplicative constant gap.**

I. Introduction

Coded caching is a technique that aims to tradeoff network load from a server for storage space at the end users. In [1], Maddah-Ali and Niesen first defined a shared-link network model with multiple cache-aided users (referred to as MAN), and characterized its fundamental limits via a cut-set converse. MAN uses a combinatorial uncoded placement strategy and network coding for the delivery. Compared to uncoded delivery, the MAN scheme achieves a great load reduction, thanks to a global coded cache gain, and it is order optimality to within a constant factor.

Wan *et al.* derived a converse bound under the constraint of uncoded cache placement and proved that the MAN scheme achieves it when the system has more files than users [2]. The Yu *et al.*'s scheme (referred to as YMA) improved the MAN scheme and was shown to meet the Wan *et al.*'s converse under the constraint of uncoded placement for all system parameters [3]. In terms of order optimality, Yu *et al.* proved that the YMA scheme is optimal to within a factor of 2 [4].

In the MAN setting, users request a single file from the server, i.e., *single-file retrieval* (SFR). In [5], Wan *et al.* extended the demands so as to include *scalar linear function retrieval* (SLFR), that is, a linear function of the files; they showed that a YMA-type scheme is optimal under the constraint of uncoded placement for the expanded demand space.

The MAN setting has limitation. In partcular, i) the MAN scheme halts when some users are offline, i.e., who fail to submit their demand; and ii) decoding the delivered multicast messages requires each user to know all demands. For the first issue, [1], [6] introduced a *decentralized* coded caching scheme, where the cache placement happens in a decentralized manner. In contrast to those works, in [7] we introduced the *hotplug model*, where the server can accommodate a fixed number of offline users. Compared to the decentralized scheme, the proposed hotplug schemes have better load performance and are exactly optimal for certain system parameters, and otherwise optimal to within a constant factor.

For the second issue, [8] introduced the *Virtual Users* (VU) idea, whereby the server pretends to serve a lot more users than those actually present in the system; the resulting scheme was shown to be order optimal in [9]. The Authors of [10] introduced the *Privacy Keys* (PK) idea, which essentially uses a one-time pad to mask a demand and stores PKs in the caches to enable successful decoding; the resulting scheme is order optimal for both SFR and SLFR demands under a privacy constraint against colluding users. In [11], the SLFR hotplug model with demand privacy against colluding users is considered; a scheme that combines Maximum Distance Separable (MDS) with the PK idea is shown to achieve lower subpacketization and better load in the small cache regime than trivial baseline schemes. To the best of our knowledge, no order optimality results are known for the hotplug model with demand privacy; this paper aims to fill this gap.

Main Contributions: Inspired by [12], we derive three new schemes for the hotplug model with demand privacy against colluding users based on the combination of VU and MDS codes. Two schemes are shown to be optimal in the small or large memory regime, respectively. We also show a constant multiplicative gap for all memory regimes.

Paper Outline: The rest of paper is organized as follows. Section II states the problem formulation. Section III summarizes related prior works. Section IV summarizes our main results. Section V gives a specific example. Section VI conclude this paper. Some proofs can be found in Appendix.

II. Problem Settings

A. Notation Convention

We adopt the following notation convention. Calligraphic symbols denote sets, bold lowercase symbols vectors, bold uppercase symbols matrices, and sans-serif symbols system parameters. $|\cdot|$ is the cardinality of a set or the length of a vector. For integers a and b, $\binom{a}{b}$ is the binomial coefficient, or 0 if $a \geq b \geq 0$ does not hold. For an integer b, we let $[b] := \{1, \ldots, b\}$. For sets $\mathcal{S}$ and $\mathcal{Q}$, we let $\mathcal{S} \setminus \mathcal{Q} := \{k : k \in \mathcal{S}, k \notin \mathcal{Q}\}$. For a collection $\{Z_1, \ldots Z_n\}$ and a index set

$\mathcal{S} \subseteq [n]$, we let $Z_{\mathcal{S}} := \{Z_i : i \in \mathcal{S}\}$. For a set $\mathcal{G}$ and an integer t, we let $\Omega^t_{\mathcal{G}} := \{\mathcal{T} \subseteq \mathcal{G} : |\mathcal{T}| = t\}$.

B. SFR Hotplug Model

An $(\mathsf{A}, \mathsf{K}, \mathsf{N})$ SFR (single-file retrieval) hotplug model consists of N files stored at a central server, and K cache-aided users connected to the server via an error-free shared link. File i, $i \in [\mathsf{N}]$, is denoted by F_i, contains B i.i.d uniformly random symbols over a finite field $\mathbb{F}_{\mathsf{q}}$, where q is a prime-power number. User k, $k \in [\mathsf{K}]$, has a cache, denoted by Z_k, which can store MB symbols. The server is aware that only $\mathsf{A} \leq \mathsf{K}$ users are active, but the identity of the active users remain unknown until they sent their SFR demands.

In the *placement phase*, the server generates a random variable τ_k for every user k, and fills the cache of user k based on the files and τ_k, i.e.,

$$H(Z_k \mid F_{[\mathsf{N}]}, \tau_k) = 0, \quad \forall k \in [\mathsf{K}]. \tag{1}$$

The cache size is MB symbols for $\mathsf{M} \in [0, \mathsf{N}]$.

In the *delivery phase*, only A users are active, indexed by the set $\mathcal{I} \in \Omega^{\mathsf{A}}_{[\mathsf{K}]}$. Active user $k \in \mathcal{I}$ requests file $d_k \in [\mathsf{N}]$, and the collection of all active users' demands is denoted by $d_{\mathcal{I}}$. The server serves the demands of the active users by sending a message X which satisfies

$$H(X \mid F_{[\mathsf{N}]}, \mathcal{I}, d_{\mathcal{I}}, \tau_{[\mathsf{K}]}) = 0, \tag{2}$$

which contains RB symbols, where $\mathsf{R} \in [0, \mathsf{N}]$ is the load.

Each active user decodes its desired file based on the delivery message and its own cache, i.e.,

$$H(F_{d_k} \mid X, Z_k, d_k) = 0, \quad \forall k \in \mathcal{I}. \tag{3}$$

In addition, for any set $\mathcal{B} \subseteq \mathcal{I}$ of colluding active users, any active user in $\mathcal{B}$ must not obtain any information about the demands of other active users, i.e.,

$$I(d_{\mathcal{I}\setminus\mathcal{B}}; d_{\mathcal{B}}, Z_{\mathcal{B}}, X \mid F_{[\mathsf{N}]}, \mathcal{I}) = 0, \quad \forall \mathcal{B} \subseteq \mathcal{I}. \tag{4}$$

The optimal load with demand privacy is defined as

$$\mathsf{R}^\star_{\mathrm{p}}(\mathsf{M}) = \limsup_{\mathsf{B}\to\infty} \min_{X, Z_1, \ldots Z_{\mathsf{K}}} \max_{\mathcal{I}, d_{\mathcal{I}}} \{\mathsf{R} : \text{conditions in (1), (2), (3), and (4) are satisfied}\}, \tag{5}$$

If $\mathsf{R}^\star$ is the minimal load without the demand privacy constraint in (4), then trivially $\mathsf{R}^\star \leq \mathsf{R}^\star_{\mathrm{p}}$.

III. Prior Results and Baseline Schemes

A. YMA scheme with SLFR demands

In the SLFR model, demands are linear combinations of files (rather than a single files). The classical MAN coded caching model is obtained for $\mathsf{A} = \mathsf{K}$ and without the privacy constraint (4). For this classical setting, the YMA scheme is optimal when the placement is uncoded [3], [5], for both SFR and SLFR demands. YMA for SLFR demands is a key ingredient for the schemes we shall introduce later.

Placement Phase: Fix $t \in [0 : \mathsf{K}]$. Partition each file into $\binom{\mathsf{K}}{t}$ equal-size subfiles as

$$F_i = (F_{i,\mathcal{W}} : \mathcal{W} \in \Omega^t_{[\mathsf{K}]}), \quad \forall i \in [\mathsf{N}]. \tag{6}$$

The cache content of user k is

$$Z_k = (F_{i,\mathcal{W}} : i \in [\mathsf{N}], \mathcal{W} \in \Omega^t_{[\mathsf{K}]}, k \in \mathcal{W}), \quad \forall k \in [\mathsf{K}]. \tag{7}$$

The memory size is thus

$$\mathsf{M}^{(\mathrm{YMA})}_t := \mathsf{N}\frac{\binom{\mathsf{K}-1}{t-1}}{\binom{\mathsf{K}}{t}} = \mathsf{N}\frac{t}{\mathsf{K}}. \tag{8}$$

Bilinear Product: Given two vectors $\mathbf{a} \in \mathbb{F}_{\mathsf{q}}^{\mathsf{N}}$ and $\mathbf{b} \in \mathbb{F}_{\mathsf{q}}^{\binom{\mathsf{K}}{t}}$ We denote the "bilinear product of the subfiles" as

$$\mathrm{T}(\mathbf{a}, \mathbf{b}) := \sum_{n\in[\mathsf{N}]} \sum_{j\in[\binom{\mathsf{K}}{t}]} a_n F_{n,\mathcal{W}_j} b_j \in \mathbb{F}_{\mathsf{q}}^{\mathsf{B}/\binom{\mathsf{K}}{t}}. \tag{9}$$

Note that we can express subfile $F_{i,\mathcal{W}_j}$ as $F_{i,\mathcal{W}_j} = \mathrm{T}(\mathbf{e}_i, \mathbf{e}_j), i \in [\mathsf{N}], j \in [\binom{\mathsf{K}}{t}]$, where $\mathbf{e}_i$ is a one-hot vector with 1 in position i; with a slight abuse of notation, we also write $F_{i,\mathcal{W}} = \mathrm{T}(\mathbf{e}_i, \mathbf{e}_{\mathcal{W}})$, for $i \in [\mathsf{N}]$, $\mathcal{W} \in \Omega^t_{[\mathsf{K}]}$.

SLFR Demands: Each user $k \in [\mathsf{K}]$ demands the linear combination of files denoted by

$$B_k = \sum_{n\in[\mathsf{N}]} d_{k,n} F_n, \forall k \in [\mathsf{K}]. \tag{10}$$

where $d_{k,n} \in \mathbb{F}_{\mathsf{q}}$. We denote the demand vector for user k as $\mathbf{d}_k = [d_{k,1}, \ldots, d_{k,\mathsf{N}}]$. Given the file partitioning in (7), we can partition (10) as $B_k = (B_{k,\mathcal{W}} : \mathcal{W} \in \Omega^t_{[\mathsf{K}]})$ where *block* $B_{k,\mathcal{W}}$ is defined as

$$B_{k,\mathcal{W}} := \sum_{n\in[\mathsf{N}]} d_{k,n} F_{n,\mathcal{W}} = \mathrm{T}(\mathbf{d}_k, \mathbf{e}_{\mathcal{W}}), \ \forall k \in [\mathsf{K}]. \tag{11}$$

Delivery Phase: Given the demand matrix $\mathbf{D} = [\mathbf{d}_1; \ldots; \mathbf{d}_{\mathsf{K}}]$, the server constructs the multicast messages

$$X_{\mathcal{S}} = \sum_{k\in\mathcal{S}} \alpha_{k,\mathcal{S}\setminus\{k\}} B_{k,\mathcal{S}\setminus\{k\}} \tag{12a}$$

$$= \sum_{k\in\mathcal{S}} \alpha_{k,\mathcal{S}\setminus\{k\}} \mathrm{T}(\mathbf{d}_k, \mathcal{S}\setminus\{k\}), \ \forall \mathcal{S} \in \Omega^{t+1}_{[\mathsf{K}]}, \tag{12b}$$

where the *encoding coefficients* $\alpha_{k,\mathcal{S}\setminus\{k\}} \in \{+1, -1\}$ are chosen as shown in [5], [13]. The server selects a *leader set* $\mathcal{L} \subseteq [\mathsf{K}]$ such that $\mathsf{rank}(\mathbf{D}[\mathcal{L}]) = \mathsf{rank}(\mathbf{D})$, and sends

$$X = (\mathcal{L}, \mathbf{D}, X_{\mathcal{S}} : \mathcal{S} \in \Omega^{t+1}_{[\mathsf{K}]}, \mathcal{S} \cap \mathcal{L} \neq \emptyset). \tag{13}$$

As shown in [5], [13], the delivery signal in (13) allows all users to successfully decode their requested SLFR. The key observation is that it is possible to choose the encoding coefficients in (12) in such a way that the non-leader users, indexed by $[\mathsf{K}] \setminus \mathcal{L}$, can locally reconstruct the not-sent multicast messages $X_{\mathcal{A}}$, for all $\mathcal{A} \in \Omega^{t+1}_{[\mathsf{K}]\setminus\mathcal{L}}$, from the delivery signal in (13).

B. Extension of YMA to hotplug without privacy

When $\mathsf{A} < \mathsf{K}$, the server can assign to the offline users the demand of the first leader user. As a consequence, for the worst case of $|\mathcal{L}| = \min(\mathsf{A}, \mathsf{N})$, the memory size is unchanged while the load is given by the following theorem.

Theorem III.1 (Extension of YMA to SLFR hotplug [7, Theorem 1])**.** *For an $(\mathsf{A}, \mathsf{K}, \mathsf{N})$ SLFR hotplug coded caching system without privacy, let $t \in [0, \mathsf{K}]$, the lower convex envelope of the following points is achievable*

$$(\mathsf{M}_t, \mathsf{R}_t)^{(\text{YMA+})} = \left(\frac{t}{\mathsf{K}}, \frac{\binom{\mathsf{K}}{t+1} - \binom{\mathsf{K}-\min(\mathsf{A},\mathsf{N})}{t+1}}{\binom{\mathsf{K}}{t}} \right). \tag{14}$$

Works [3], [6] describe a decentralized placement for the classical model where each symbol of all files is stored in each user's cache with a probability $\mu := \mathsf{M}/\mathsf{N}$. Similarly to the YMA+ scheme in Theorem III.1, we can extend the decentralized scheme from the classical model to the hotplug model by filling in "fake" demands for the offline users. As a result, the following load is achievable for the SLFR hotplug without privacy

$$\mathsf{R}^{(\text{decen+})}(\mathsf{M}) = \frac{1-\mu}{\mu} \left(1 - (1-\mu)^{\min(\mathsf{A},\mathsf{N})}\right). \tag{15}$$

For the SLFR hotplug without privacy we have [7, Theorem 2]

$$\mathsf{R}^\star(\mathsf{M}) \le \mathsf{R}^{(\text{YMA+})}(\mathsf{M}) \le \mathsf{R}^{(\text{decen+})}(\mathsf{M}) \le 2\mathsf{R}^\star(\mathsf{M}). \tag{16}$$

C. Privacy-Keys scheme with SFR demands

In [10], a scheme which preserves demand privacy against colluding users by using *Privacy Keys* (PK) was proposed for the classical setting of $\mathsf{A} = \mathsf{K}$.

Placement Phase: Files are split is as in (6). We shall continue to use the notation of bilinear product in (9). For each user k, $k \in [\mathsf{K}]$, the server picks i.i.d uniformly at random the *key vector* $\mathbf{p}_k \in \mathbb{F}_\mathsf{q}^\mathsf{N}$, such that $\sum_{i=1}^\mathsf{N} p_{k,i} = \mathsf{q} - 1$. The server populates the cache of user k as

$$Z_k = \left\{ \mathrm{T}(\mathbf{e}_i, \mathbf{e}_{\{k\}\cup\mathcal{Q}}) : i \in [\mathsf{N}], \mathcal{Q} \in \Omega^{t-1}_{[\mathsf{K}]\setminus\{k\}} \right\} \tag{17a}$$

$$\bigcup \left\{ \mathrm{T}(\mathbf{p}_k, \mathbf{e}_\mathcal{W}) : \mathcal{W} \in \Omega^t_{[\mathsf{K}]\setminus\{k\}} \right\}, \ \forall k \in [\mathsf{K}], \tag{17b}$$

where (17a) is the uncoded part of the cache as in (7), while in (17b) are the PKs of the form $\mathrm{T}(\mathbf{p}_k, \mathbf{e}_\mathcal{W})$ for all sets $\mathcal{W}$ that do not include k. The needed memory size is

$$\mathsf{M}_t^{(\text{PK})} := \mathsf{N} \frac{\binom{\mathsf{K}-1}{t-1}}{\binom{\mathsf{K}}{t}} + \frac{\binom{\mathsf{K}-1}{t}}{\binom{\mathsf{K}}{t}} = 1 + \frac{t}{\mathsf{K}}(\mathsf{N}-1) \ge 1, \tag{18}$$

where $\mathsf{M}_t^{(\text{YMA})} = \mathsf{N}\frac{t}{\mathsf{K}} \le \mathsf{M}_t^{(\text{PK})}$ as users also store PKs.

Delivery Phase: For every active user $k \in \mathcal{I}$, user k requests a single file F_{d_k} where $d_k \in [\mathsf{N}]$, alternatively it can be written as $\mathbf{e}_{d_k}$ in the SLFR fashion. We denote the *query vector* for user k as

$$\mathbf{q}_k := \mathbf{p}_k + \mathbf{e}_{d_k} \in \mathbb{F}_\mathsf{q}^\mathsf{N}, \ \forall k \in [\mathsf{K}], \tag{19}$$

which can be thought of as a one-time pad of the demand vector $\mathbf{e}_{d_k}$ by the key vector $\mathbf{p}_k$. Note that $\sum_{i=1}^\mathsf{N} q_{k,i} = \sum_{i=1}^\mathsf{N} p_{k,i} + 1 = 0$. The collection of all query vectors gives the query matrix $\mathbf{Q} = [\mathbf{q}_1; \ldots; \mathbf{q}_\mathsf{K}] \in \mathbb{F}_\mathsf{q}^{\mathsf{K}\times\mathsf{N}}$. For those offline users, similar to YMA+, the server fills in their demands from active users demands, thus $\mathsf{rank}(\mathbf{Q}) = \min(\mathsf{K}, \mathsf{N}-1)$. The server constructs the multicast messages

$$X_\mathcal{S} = \sum_{k\in\mathcal{S}} \alpha_{k,\mathcal{S}\setminus\{k\}} \, \mathrm{T}(\mathbf{q}_k, \mathbf{e}_{\mathcal{S}\setminus\{k\}}), \ \forall \mathcal{S} \in \Omega^{t+1}_{[\mathsf{K}]}, \tag{20}$$

where $\alpha_{k,\mathcal{S}\setminus\{k\}} \in \{+1, -1\}$. The server selects a *leader set* $\mathcal{L} \subseteq [\mathsf{K}]$ such that $\mathsf{rank}(\mathbf{Q}[\mathcal{L}]) = \mathsf{rank}(\mathbf{Q})$, and sends

$$X = (\mathcal{L}, \mathbf{Q}, X_\mathcal{S} : \mathcal{S} \in \Omega^{t+1}_{[\mathsf{K}]}, \mathcal{S} \cap \mathcal{L} \neq \emptyset). \tag{21}$$

It can be shown that the signal X and the cached content guarantee successful recovery of the demanded file while preserving privacy against colluding users [10].

D. Extension of PK to hotplug with privacy

When $\mathsf{A} < \mathsf{K}$, the server can assign to the offline users the demand of the first leader user. As a consequence, the PK memory size is unchanged while the load is given by the following theorem.

Theorem III.2 (Extension of PK to SFR hotplug with privacy [11, Theroem 2])**.** *For an $(\mathsf{A}, \mathsf{K}, \mathsf{N})$ SFR hotplug coded caching system with demand privacy, let $t \in [0, \mathsf{K}]$, the lower convex envelope of $(0, \mathsf{N}), (\mathsf{N}, 0)$, and the following points is achievable*

$$(\mathsf{M}_t, \mathsf{R}_t)^{(\text{PK+})} = \left(1 + \frac{t}{\mathsf{K}}(\mathsf{N}-1), \frac{\binom{\mathsf{K}}{t+1} - \binom{\mathsf{K}-\min(\mathsf{A},\mathsf{N}-1)}{t+1}}{\binom{\mathsf{K}}{t}} \right). \tag{22}$$

E. PK-type schemes for hotplug with privacy

In [14], we combined MDS coded placement and the PK idea to designs schemes; we skip the technical detail due to limited space and refer the readers to [14].

Theorem III.3 (MDS+PK for SFR hotplug with privacy [11, Theroem 3 and 4])**.** *For an $(\mathsf{A}, \mathsf{K}, \mathsf{N})$ hotplug model with privacy, for $t \in [0 : \mathsf{A}]$, the lower convex envelope of $(0, \mathsf{N}), (\mathsf{N}, 0)$, and the following points is achievable*

$$(\mathsf{M}_t, \mathsf{R}_t)^{\text{MDS+PK1}} = \left(\frac{\mathsf{N}\binom{\mathsf{K}-1}{t-1} + \left[\binom{\mathsf{A}}{t} - \binom{\mathsf{K}-1}{t-1}\right]^+}{\binom{\mathsf{A}}{t}}, \frac{\binom{\mathsf{A}}{t+1} - \binom{\mathsf{A}-\min(\mathsf{A},\mathsf{N}-1)}{t+1}}{\binom{\mathsf{A}}{t}} \right), \tag{23}$$

$$(\mathsf{M}, \mathsf{R})^{\text{MDS+PK2}} = \left(\mathsf{N} - \frac{\mathsf{N}-1}{\mathsf{A}}, \frac{1}{\mathsf{A}} \right). \tag{24}$$

F. Virtual-Users Scheme

The Author of [8] introduced the idea of virtual users (VU) to achieve demand privacy in the SFR classical model. In the placement phase, each physical user has associated to it a distinct set of N *virtual-users*; every physical users picks

independently and uniformly at random the cache content from one of its associated virtual users. In the delivery phase, the demands all the KN virtual users are assigned in such a way that each of the N file is demanded exactly K times. As a result, the performance of the VU scheme with this "restricted" demand type is the same as that of the YMA scheme with KN users and N files, i.e., in (14) we replace K with KN. The essence of VU idea is to derive a private scheme for a (K,N) classical model from a non-private scheme for a (KN,N) classical model [15]. In Section IV, we shall derive baseline schemes by leveraging the VU idea.

At this point, no constant gap result for SLFR hotplug with privacy is known, to the best of our knowledge, as opposed to the case without privacy in (16).

IV. Main Results

We first take the YMA and the decentralized schemes for the non-private classical model to obtain a private scheme for the hotplug model by the VU idea. The key observation is that the argument in [12, Theorem 2] extends to the SFR hotplug model: we can derive a scheme for a $(\mathsf{A},\mathsf{K},\mathsf{N})$ SFR hotplug private model from a $(\mathsf{AN},\mathsf{KN},\mathsf{N})$ SFR hotplug NON-private model. Notng that $\min(\mathsf{AN},\mathsf{N}) = \mathsf{N} = \min(\mathsf{KN},\mathsf{N})$, we conclude that the load of the resulting VU-based schemes is unchanged, as the next theorem states.

Theorem IV.1 (Application of VU to SFR hotplug with privacy)**.** *For an $(\mathsf{A},\mathsf{K},\mathsf{N})$ SFR hotplug model with privacy, let $t \in [0:\mathsf{KN}]$, the lower convex envelope of the following points is achievable*

$$(\mathsf{M}_t,\mathsf{R}_t)^{(\text{YMA\&VU})} = \left(\mathsf{N}\frac{t}{\mathsf{KN}}, \frac{\binom{\mathsf{KN}}{t+1} - \binom{\mathsf{KN}-\mathsf{N}}{t+1}}{\binom{\mathsf{KN}}{t}}\right). \tag{25}$$

In addition, the following is achievable

$$\mathsf{R}^{(\text{decen\&VU})}(\mathsf{M}) = \frac{1-\mu}{\mu}\Big(1-(1-\mu)^{\mathsf{N}}\Big), \quad \mu := \mathsf{M}/\mathsf{N}. \tag{26}$$

The proof of Theorem IV.1 is a special case of the general derivation in [14, Appendix].

Theorem IV.2 (New schemes for SFR hotplug with privacy)**.** *For an $(\mathsf{A},\mathsf{K},\mathsf{N})$ SFR hotplug system with privacy, the following points are achievable*

$$(\mathsf{M},\mathsf{R})^{\text{New1}} = \left(\frac{1}{L}, \mathsf{N}\left(1-\frac{1}{L}\right)\right), \; L := \mathsf{A}(\mathsf{N}-1)+1, \tag{27}$$

$$(\mathsf{M},\mathsf{R})^{\text{New2}} = \left(\frac{1}{\mathsf{A}}, \mathsf{N}-\frac{\mathsf{N}+1}{2\mathsf{A}}\right), \tag{28}$$

$$(\mathsf{M},\mathsf{R})^{\text{New3}} = \left(\mathsf{N}\left(1-\frac{1}{\mathsf{AN}}\right), \frac{1}{\mathsf{AN}}\right). \tag{29}$$

Remark 1. Inspired by [12], all tradeoffs in Theorem IV.2 are derived from an immediate application of the general result proved in Appendix that a scheme for an $(\mathsf{A},\mathsf{K},\mathsf{N})$ SFR hotplug model with privacy can be obtained from scheme for a $(\mathsf{NA},\mathsf{NK},\mathsf{N})$ SFR hotplug system without privacy.

The first new point in (27) extends the optimal classical VU scheme in [15] to the hotplug model.

The point in (29) is exactly optimal as it meets the non-private cut-set bound $\mathsf{R}^\star(\mathsf{M}) \geq 1-\mathsf{M}/\mathsf{N}$. We provide the achievability proof of (27) in Appendix Both the second point in (28) and the third point in (29) are [14, Theorem IV.1 (for $t=1$)] and [14, Theorem IV.3] used with parameters $(\mathsf{AN},\mathsf{KN},\mathsf{N})$. For sake of space, we omit the proof details, which can be found in Appendix.

Note that all points in Theorem IV.2 are function of A and N, and not of K. For the cases of hotplug for which exact optimality is known, the optimal performance only depends on A, and not on K. □

Remark 2 (Extension to SLFR demands)**.** Theorem IV.2 can be extended to SLFR hotplug model seamlessly. In term of VU schemes, the necessary step is to create virtual users who have distinct demands. In SLFR hotplug, there are $\widetilde{\mathsf{N}} := (\mathsf{q}^{\mathsf{N}}-1)/(\mathsf{q}-1)$ non-zero distinct lienar combination demands, which depends on q. The first new tradeoff point requires an MDS code, whose dimension is $\mathsf{K}\widetilde{\mathsf{N}} \times \mathsf{A}\widetilde{\mathsf{N}}$; to guarantee such MDS code exists, we need a large q; an interesting open question is find the minimum of q necessary to implement the SLFR hotplug scheme. □

Remark 3 (Converse bound for hotplug with privacy)**.** A trivial converse bound for the hotplug system with privacy can be obtained by considering the converse bound in [10, Theorem 3] (for demand demand privacy with colluding users in the classical setting) by replacing K with A. The reasoning is that we cannot do better than knowing at the time of placement which A users will be active, and derive the optimal performance of these active users. The lower bound from [10, Theorem 3] is thus as follows

$$\mathsf{R}_{\text{p}}^\star(\mathsf{M}) \geq \max_{\ell\in[\mathsf{N}]}\left\{\ell + \frac{(\mathsf{N}-\ell)\min\{\ell+1,\mathsf{A}\}}{(\mathsf{N}-\ell)+\min\{\ell+1,\mathsf{A}\}} - \ell\mathsf{M}\right\}, \tag{30}$$

which is known to be good in the low memory regime [10], i.e., note $\mathsf{R}_{\text{p}}^\star(0) \geq \mathsf{N}$. In later sections, we shall combine the bound in (30) (with privacy) together with [4, Theorem 2 and Theorem 4] (without privacy) as a converse bound for our plots, and for the gap result next. □

Theorem IV.3 (Optimality for SFR hotplug with privacy)**.** *For an $(\mathsf{A},\mathsf{K},\mathsf{N})$ SFR hotplug system with demand privacy,*

1) *when $\mathsf{A} \geq \mathsf{N}$, we have $\mathsf{R}^{(\text{YMA\&VU})} \leq 2.00884\ \mathsf{R}_{\text{p}}^\star$;*
2) *when $\mathsf{A} < \mathsf{N}$, we have $\mathsf{R}^{(\text{PK+})} \geq 5.4656\ \mathsf{R}_{\text{p}}^\star$;*
3) *when $\mathsf{M} \leq \frac{1}{\mathsf{A}(\mathsf{N}-1)+1}$ or $\mathsf{M} \geq \mathsf{N}\left(1-\frac{1}{\mathsf{AN}}\right)$, the points from Theorem IV.2 are optimal.*

The proof of Theorem IV.3 is provided in [14, Appendix].

V. Example of hotplug with privacy

In this section, we illustrate by way of example how the new SFR hotplug model with privacy. The memory-load tradeoff is reported in Fig. 1 for $(\mathsf{A},\mathsf{K},\mathsf{N}) = (3,6,3)$. As shown in Fig. 1, the first and third points in Theorem IV.3 are optimal. As a comparison, we provide other schemes which also

preserve demand privacy: MDS+PK is Theorem III.3; PK+ is Theorem IV.2; YMA&VU is Theorem IV.1; the converse is as stated in Remark 3. We choose this case as an example as it is the smallest example where we can clearly and simply show all the new aspects of the proposed schemes. More examples can be found at [14].

1) The first point in Theorem IV.2: Placement Phase: Partition each file into 7 subfiles as

$$F_n = (F_{n,\ell} : \ell \in [7]) = (\mathrm{T}(\mathbf{e}_n, \mathbf{e}_\ell) : \ell \in [7]), \ \forall n \in [3]. \tag{31}$$

Consider the MDS generator matrix $\mathbf{G} = [\mathbf{g}_j : j \in [18]]$ of rate $7/18$, i.e., any 7 vectors $\mathbf{g}$'s are linear independent. Sample the scalar $p_j \in [3]$ i.i.d. uniformly at random for each user $j \in [6]$. The placement is,

$$Z_j = \{p_j\} \cup \{\sum_{n=1}^{3} \mathrm{T}(\mathbf{e}_n, \mathbf{g}_{3(j-1)+p_j})\}, \ \forall j \in [6]. \tag{32}$$

Delivery Phase: Let $\Psi : [3]^3 \to [3]^3$ denote the *cyclic shift operator*, such that $\Psi(1,2,3) = (3,1,2)$. Let us denote a vector $\mathbb{I}_3 := (1,2,3)$, and define the subtraction under modular 3 as $\ominus$, for example $1 \ominus 1 = 0, 1 \ominus 2 = 2$. Consider an active user set $\mathcal{I} = \{j_1, j_2, j_3\} \in \Omega_6^3$ and their demands $d_{\mathcal{I}} \in \{d_{j_1}, d_{j_2}, d_{j_3}\}$, the server creates an vector of length 3 for each user $j \in \mathcal{I}$ as

$$\widetilde{\mathbf{d}}_j := \Psi^{p_j \ominus d_j}(\mathbb{I}_3), \tag{33}$$

where the operator Ψ^i denotes the i-times cyclic shift operator, which is the operator Ψ applied i times. We define the extended active user set $\widetilde{\mathcal{I}}$ of size 9 as

$$\widetilde{\mathcal{I}} := ((j-1)\mathsf{N} + \ell : j \in \mathcal{I}, \ell \in [3]), \tag{34}$$

and

$$p_{\mathcal{I}} \ominus d_{\mathcal{I}} := (p_{j_1} \ominus d_{j_1}, p_{j_2} \ominus d_{j_2}, p_{j_3} \ominus d_{j_3}). \tag{35}$$

Therefore, the server sends $(\widetilde{\mathcal{I}}, p_{\mathcal{I}} \ominus d_{\mathcal{I}})$, and

$$X = \left\{\mathrm{T}(\mathbf{e}_n, \mathbf{g}_j) : j \in \widetilde{\mathcal{I}}, \ell \in [3], n \in [3] \setminus \{\tilde{d}_{j,\ell}\}\right\},$$

where $\tilde{d}_{j,\ell}$ is the ℓ-th element of $\widetilde{\mathbf{d}}_j$.

Decoding: User $j \in [\mathcal{I}]$, who knows $\{\mathrm{T}(\mathbf{e}_n, \mathbf{g}_{3(j-1)+p_j}) : n \in [3] \setminus \{d_j\}\}$ from X, can "unlocks" $\mathrm{T}(\mathbf{e}_{d_j}, \mathbf{g}_{3(j-1)+p_j})$ from its cache. Next, for every $i \in \mathcal{I}$ and $\ell \in [3]$ such that $\tilde{d}_{i,\ell} \neq d_j$, the bilinear combination $\mathrm{T}(\mathbf{e}_{d_j}, \mathbf{g}_{3(i-1)+\ell})$ is contained in X, in total 6. Since user j knows 7 independent coded subfiles and the MDS rate is $7/18$, it can restore F_{d_j}.

Performance: The point $(\mathsf{M}, \mathsf{R}) = (1/7, 18/7)$ is achievable.

2) The second and third points in Theorem IV.2: These points are attained from non-private HT1 and HT3 Schemes, respectively, in [14, Theorem IV.1 and IV.3] with parameters $(\mathsf{A}, \mathsf{K}, \mathsf{N}) = (9, 18, 3)$.

Placement Phase: Each real user $j \in [6]$, together with 2 extra virtual users forms a group. The server populates the cache content for all users including virtual ones, then a real user uniformly chooses the content of one cache from its assigned group independently and uniformly at random.

Delivery Phase: After the real users have sent their demands, the server transmits to the extended set of users as shown in (33) and (34).

Performance: The point $(\mathsf{M}, \mathsf{R}) = (1/3, 14/6)$ and $(\mathsf{M}, \mathsf{R}) = (24/9, 1/9)$ are achievable respectively.

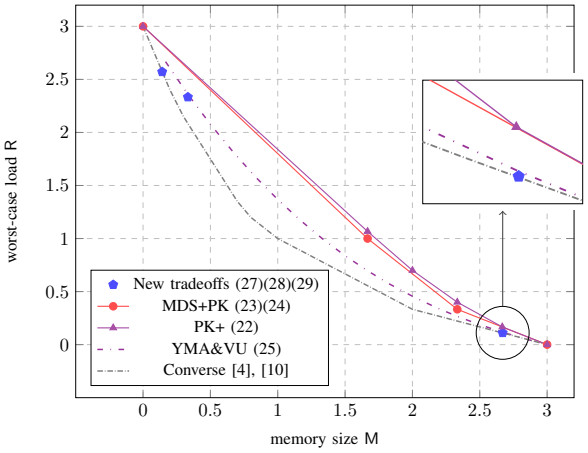

Fig. 1: Memory-load tradeoffs for the hotplug model with privacy when $(\mathsf{A}, \mathsf{K}, \mathsf{N}) = (3, 6, 3)$.

VI. Conclusion

We investigated the hotplug model under the constraint of demand privacy against colluding users. Three new points were shown to be achievable based on the virtual user idea, two of which are proved to be optimal when the memory is either small or large. We also proved other scheme that allowed us to derive an order optimality result for all system parameters.

This work was funded in parts by NSF under awards 1910309 and 2312229.

REFERENCES

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

[2] K. Wan, D. Tuninetti, and P. Piantanida, "An index coding approach to caching with uncoded cache placement," *IEEE Transactions on Information Theory*, vol. 66, no. 3, pp. 1318–1332, 2020.

[3] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "The exact rate-memory tradeoff for caching with uncoded prefetching," *IEEE Transactions on Information Theory*, vol. 64, no. 2, pp. 1281–1296, 2017.

[4] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Characterizing the rate-memory tradeoff in cache networks within a factor of 2," *IEEE Transactions on Information Theory*, vol. 65, no. 1, pp. 647–663, 2018.

[5] K. Wan, H. Sun, M. Ji, D. Tuninetti, and G. Caire, "On the optimal load-memory tradeoff of cache-aided scalar linear function retrieval," *IEEE Transactions on Information Theory*, vol. 67, no. 6, pp. 4001–4018, 2021.

[6] M. A. Maddah-Ali and U. Niesen, "Decentralized coded caching attains order-optimal memory-rate tradeoff," *IEEE/ACM Transactions On Networking*, vol. 23, no. 4, pp. 1029–1040, 2014.

[7] Y. Ma and D. Tuninetti, "On coded caching systems with offline users," in *2022 IEEE International Symposium on Information Theory (ISIT)*, pp. 1133–1138, IEEE, 2022.

[8] S. Kamath, "Demand private coded caching," *arXiv preprint arXiv:1909.03324*, 2019.

[9] K. Wan and G. Caire, "On coded caching with private demands," *IEEE Transactions on Information Theory*, vol. 67, no. 1, pp. 358–372, 2020.

[10] Q. Yan and D. Tuninetti, "Fundamental limits of caching for demand privacy against colluding users," *IEEE Journal on Selected Areas in Information Theory*, vol. 2, no. 1, pp. 192–207, 2021.

[11] Y. Ma and D. Tuninetti, "Demand privacy in hotplug caching systems," in *2023 IEEE International Symposium on Information Theory (ISIT)*, pp. 424–429, IEEE, 2023.

[12] K. K. Namboodiri and B. S. Rajan, "Optimal demand private coded caching for users with small buffers," in *2021 IEEE International Symposium on Information Theory (ISIT)*, pp. 706–711, IEEE, 2021.

[13] Y. Ma and D. Tuninetti, "A general coded caching scheme for scalar linear function retrieval," *IEEE Journal on Selected Areas in Information Theory*, vol. 3, no. 2, pp. 321–336, 2022.

[14] Y. Ma and D. Tuninetti, "On coded caching systems with offline users, with and without demand privacy against colluding users," *arXiv preprint arXiv:2401.06894*, 2024.

[15] S. Kamath, J. Ravi, and B. K. Dey, "Demand-private coded caching and the exact trade-off for n= k= 2," in *2020 National Conference on Communications (NCC)*, pp. 1–6, IEEE, 2020.