On Coded Caching for Two Users with Overlapping Demand Sets

Chih-Hua Chang, Chih-Chun Wang *Senior Member, IEEE*, Borja Peleato *Senior Member, IEEE*
Purdue University, West Lafayette, IN 47907, USA
Email:{chang377,chihw,bpeleato}@purdue.edu

***Abstract*—Coded caching is a technique for reducing congestion in communication networks by prefetching content during idle periods and exploiting multicasting opportunities during periods of heavy traffic. Most of the existing research in this area has focused on minimizing the worst case (i.e., peak) rate in a broadcast link with multiple identically distributed user requests. However, modern content delivery networks are investing very heavily in profiling their users and predicting their preferences.**

The minimal achievable rate of a coded caching scheme with heterogeneous user profiles is still unknown in general. This paper presents the first steps towards solving that problem by analyzing the case of two users with distinct but overlapping demand sets. Specifically, it provides a complete characterization of the uniform-average-rate capacity when the sets overlap in just one file and shows that such capacity can be achieved with selfish and uncoded prefetching. Then, it characterizes the same capacity under selfish and uncoded prefetching when the demand sets overlap in two or more files. The paper also provides explicit prefetching schemes that achieve those capacities. All our results allow for arbitrary (and not necessarily identical) users' cache sizes and number of files in each demand set.

I. Introduction

The increasing demand of video streaming data has led to a significant challenge in content distribution over communication networks. Coded caching has recently received great success in reducing the peak transmission rate in some networks by exploiting multicasting opportunities in the underlying broadcast channels. A coded caching scheme has two phases: placement and delivery. In the placement phase, the users can access the files at the server to fill their cache memories during the off-peak hours. In the delivery phase, the users announce their requests and the server, with full knowledge of the users' cache contents, transmits the information required to satisfy such requests for all the users.

Caching has been extensively studied in content distribution networks with different objectives such as access latency and transmission rate [1], [2]. Traditional "uncoded caching" schemes focus on caching the content most likely to be requested and, when the content absent from the users' caches is requested, they deliver it in plain form, uncoded. The most commonly used placement algorithms for uncoded caching are *least frequently used* (LFU) and *least recently used* (LRU), where the former retains the most frequently accessed subfiles in the past and the latter keeps the most recently used subfiles.

This work was supported in parts by NSF under Grant CCF-1422997, Grant ECCS-1407604, Grant CCF-1618475, and Grant CCF-1816013.

Coded caching, initially proposed in [3], can reduce the *worst-case* delivery time by a factor of $(\frac{1}{1+KM/FN})$ respect to traditional uncoded caching schemes, where N is the number of files, K is the number of users, F is the individual file size, and M is the individual cache size normalized by F. Existing works have characterized the coded caching capacity for some special N and K values [3]–[6] and derived order-optimal capacity expressions for general N and K [3], [7]–[10]. The worst-case setting is analytically appealing but it is not throughput optimal in practice with a time sequence of requests. Some recent results have focused on the *average-rate capacity* [8], [11]–[14], evaluating the transmission rate over a distribution of demands. The order-optimal average-rate capacity results of *user-independent file popularity*, i.e., all users having the same file popularity profile has been investigated in [8], [9], [11]–[14]. Nevertheless, the assumption of user-independent file popularity in these works is not compatible with traditional uncoded caching schemes, since the number of users with similar preferences can be small and each user often operates in a distributed manner with its own file popularity and prediction algorithm.

The rate improvement of coded caching primarily comes from the broadcasting nature and the simultaneous users' demands. However, the latter will not occur frequently and will require synchronization among the users. When the demands of users are served sequentially, coded caching increases the system complexity without providing any rate reduction. Clearly, the system should apply coded caching when users' demands are synchronized, and employ traditional uncoded caching when they come sparsely in time. One way to cover both scenarios is to enforce the so-called *uncoded prefetching* condition, which ensures that cached content is always of an uncoded form so that the content is useful to both the coded and uncoded delivery. It is obvious that such flexibility is at the expense of degrading the overall capacity since uncoded prefetching is a subclass of general prefetching schemes.

Motivated by the above discussion on limitations of user-independent file popularity and flexibility of uncoded prefetching, this work studies the average-rate capacity of coded caching with selfish and uncoded prefetching. Specifically, we assume that each user k is associated with a file demand set (FDS) Θ_k and requests each of those files with probability $\frac{1}{|\Theta_k|}$ (see Section II for details). Users will only cache segments from files in their demand set, hence the term *selfish* prefetching. The FDS setting reflects user-dependent file popularity

978-1-7281-5089-5/20/$31.00 ©2020 IEEE

by considering different users k_1 and k_2 with $\Theta_{k_1} \neq \Theta_{k_2}$. Average-rate capacity results for disjoint FDS ($\Theta_{k_1} \cap \Theta_{k_2} = \emptyset$) and dominant FDS ($\Theta_{k_1} \subset \Theta_{k_2}$) were derived in [15], [16]. This work studies the case of two users ($K = 2$) and arbitrary number of files N, where the FDSs Θ_1 of user 1 and Θ_2 of user 2 overlap in α common files.

We first show that the average-rate capacity when Θ_1 and Θ_2 share a single common file ($\alpha = 1$) can be achieved by selfish and uncoded prefetching. Then we find the average-rate capacity of selfish and uncoded prefetching for an arbitrary number of common files. From a practical perspective, this paper answers the following question:

> Suppose there are two users with significantly different file preferences (as is common in practice). However, there are α files that are simultaneously desired by both users, where α can be any number. What is the best coded caching in this scenario?

II. Problem Formulation

We consider a coded caching system with one server and $K = 2$ users. The central server has access to N files $W_1, \ldots, W_N$, with the same file size of F bits. Each W_n is independently and uniformly distributed over $\{0,1\}^F$. The cache content of user k is denoted Z_k and has size $M_k F$ bits for $k \in \{1,2\}$. Without loss of generality, we assume $M_k \in [0, N]$. For any positive integer x, we define $[x] \triangleq \{1, \ldots, x\}$.

The operation of the system contains two phases, the *placement phase* and the *delivery phase*. In the placement phase, each user k populates its cache content by

$$Z_k = \phi_k(W_1, \ldots, W_N), \ \forall k \in \{1,2\} \tag{1}$$

where ϕ_k is the caching function of user k. In the delivery phase, the two users send a demand request $\vec{d} \triangleq (d_1, d_2) \in [N]^2$ to the server, i.e., user k demands file W_{d_k}. We denote the probability mass function (pmf) of the random request d_k by $p_{d_k}^{[k]}$ for $k \in \{1,2\}$. In this work, we assume that the demands of user-1 and user-2 are *statistically independent*, that is, the joint pmf of the two users' demand $\vec{d} \in [N]^2$ is $p_{\vec{d}} = p_{d_1}^{[1]} p_{d_2}^{[2]}$.

After receiving the demand index vector $\vec{d}$, the server broadcasts an encoded signal

$$X_{\vec{d}} = \psi(\vec{d}, W_1, \ldots, W_N) \tag{2}$$

of $R_{\vec{d}} F$ bits with encoding function ψ using an error-free link to all K users. Each user k then uses $X_{\vec{d}}$ and his/her cache content Z_k to decode the requested file

$$\hat{W}_{d_k} = \mu_k(\vec{d}, X_{\vec{d}}, Z_k), \tag{3}$$

where μ_k is the decoding function of user k. A coded caching scheme is completely specified by K caching functions $\{\phi_k\}_{k=1}^K$, one encoding function ψ, and K decoding functions $\{\mu_k\}_{k=1}^K$.

Definition 1. *The* file demand set *(FDS) of user k is defined as $\Theta_k \triangleq \{n \in [N] : p_n^{[k]} > 0\}$, which is the set of files that user k desires with a strictly positive probability.*

Definition 2. *A coded caching scheme is* zero-error feasible *if $\hat{W}_{d_k} = W_{d_k}$ for all $k \in [K]$, all $d_k \in \Theta_k$ in the FDS, and all $(W_1, \cdots, W_N) \in \{0,1\}^{NF}$.*

Throughout this manuscript, we consider exclusively zero-error feasible schemes.

Definition 3. *A coded caching scheme uses* uncoded prefetching *if the cache content of user k is*

$$Z_k = (w_1, \ldots, w_N) = \phi_k(W_1, \ldots, W_N), \ \forall k \in [K] \tag{4}$$

where each w_i is an uncoded subfile of W_i, $i \in [N]$. That is, each user k stores uncoded fractions of the N files.

Definition 4. *A coded caching scheme uses* selfish prefetching *if all K caching functions ϕ_k in* (1) *can be replaced by*

$$Z_k = \phi_k(\{W_n : n \in \Theta_k\}), \ \forall k \in [K]. \tag{5}$$

Namely, each user k only stores the files that he/she is interested in, thus the name selfish. *In contrast, the more general design using* (1) *is referred to as an* unselfish *scheme.*

Definition 5. *The average-rate of a coded caching scheme is defined as*

$$\bar{R} = \sum_{\forall \vec{d}, d_k \in \Theta_k} p_{\vec{d}} R_{\vec{d}}. \tag{6}$$

The uniform-average-rate of a scheme is defined as

$$\tilde{R} = \frac{1}{\prod_{k=1}^K |\Theta_k|} \sum_{\forall \vec{d}, d_k \in \Theta_k} R_{\vec{d}}. \tag{7}$$

Furthermore, we will use $\tilde{R}_{\mathrm{su}}$ to denote the uniform-average-rate of a selfish and uncoded prefetching scheme.

$\tilde{R}$ can be viewed as a first-order approximation to the average-rate $\bar{R}$ that replaces the joint distribution $p_{\vec{d}}$ with a uniform distribution over the FDS $\prod_{k=1}^K \Theta_k$. In [6], an exact characterization of $\bar{R}$ has been provided for the 2-user/2-file setting ($N = K = 2$), which involves a detailed discussion of $N^K = 4$ dimensional request rates on the underlying values of (M_1, M_2) and $p_{\vec{d}}$. Instead of focusing on the exact $\bar{R}$, in this work we focus on the simplified, more tractable quantity $\tilde{R}$, but allow the N value to be ≥ 2.

III. Main Results

Consider a system with $K = 2$ users and arbitrary N files, where users 1 and 2 have FDS Θ_1 and Θ_2, respectively, with $|\Theta_1| = N_1$ and $|\Theta_2| = N_2$. Without loss of generality, we assume $N_1 \leq N_2 \leq N$ such that there are four possible relationships between Θ_1 and Θ_2:

$$\begin{cases} \Theta_1 = \Theta_2 & \text{Identical} \\ \Theta_1 \cap \Theta_2 = \emptyset & \text{Disjoint} \\ \Theta_1 \subset \Theta_2 & \text{Dominant} \\ \Theta_1 \cap \Theta_2 \neq \{\emptyset, \Theta_1\} & \text{Overlapped} \end{cases} \tag{8}$$

The optimal coded caching schemes for $K = 2$ users and arbitrary N files for the identical and disjoint (Θ_1, Θ_2) cases were

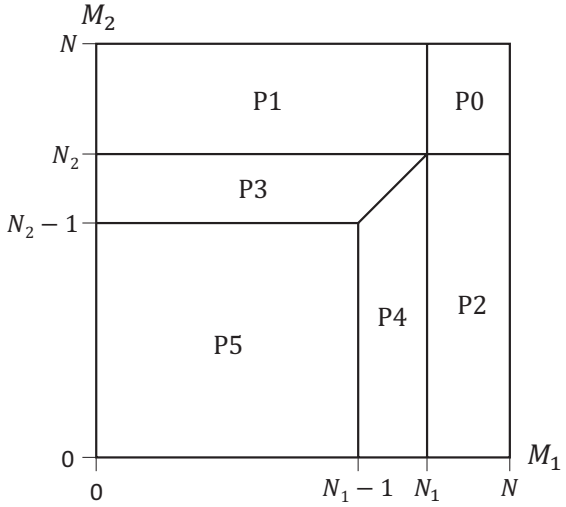

Fig. 1. The minimum average rate $\tilde{R}$ of coded caching for $|\Theta_1| = N_1$, $|\Theta_2| = N_2$, and $\alpha = 1$. For any (M_1, M_2) inside each subregion, the rate $\tilde{R}$ is characterized by the corresponding equation marked in that region.

solved in [15]. The dominant cases with $(|\Theta_1| = 1, |\Theta_2| = N)$ and with $(|\Theta_1| = 2, |\Theta_3| = 3)$ were also solved in [15] with some further discussion for large (Θ_1, Θ_2) values.

This work focuses on the case of overlapped Θ_1 and Θ_2. We refer to files in the intersection as *common* and their number is denoted by $\alpha \triangleq |\Theta_1 \cap \Theta_2|$. All other files (not common) will be labeled *unique*, since they belong to a single FDS. Section III-A describes the minimum average rate $\tilde{R}$ of coded caching when Θ_1 and Θ_2 overlap in a single file ($\alpha = 1$). Then Section III-B presents the minimum average rate $\tilde{R}_{\text{su}}$ with *selfish and uncoded prefetching for arbitrary overlap between* Θ_1 *and* Θ_2 *(*$\alpha \geq 1$*).*

A. Minimum average rate for $\alpha = 1$

Proposition 1 ($|\Theta_1| = N_1$, $|\Theta_2| = N_2$, $\alpha = 1$)**.** *Consider* $K = 2$ *users and* $N = N_1 + N_2 - 1$ *files with file demand sets* $\Theta_1 = \{1, \ldots, N_1\}$, $\Theta_2 = \{N_1, \ldots, N_1 + N_2 - 1\}$ *such that* $\Theta_1 \cap \Theta_2 = \{N_1\}$ *and* $\alpha = 1$*. The uniform-average rate* $\tilde{R}$ *is tightly characterized by:*

$$\tilde{R} \geq 0 \tag{P0}$$

$$\tilde{R} \geq 1 - M_1/N_1 \tag{P1}$$

$$\tilde{R} \geq 1 - M_2/N_2 \tag{P2}$$

$$\tilde{R} \geq \left(2 - \frac{1}{N_1}\right) - \frac{M_1}{N_1} - \frac{N_1 - 1}{N_1 N_2} M_2 \tag{P3}$$

$$\tilde{R} \geq \left(2 - \frac{1}{N_2}\right) - \frac{M_2}{N_2} - \frac{N_2 - 1}{N_1 N_2} M_1 \tag{P4}$$

$$\tilde{R} \geq \left(2 - \frac{1}{N_1 N_2}\right) - \frac{M_1}{N_1} - \frac{M_2}{N_2} \tag{P5}$$

The relationship of $\tilde{R}$ *versus* (M_1, M_2) *is illustrated in Fig. 1.*

The proof of Proposition 1 is relegated in [17]. Note that the $\tilde{R}$ described by (P0) to (P5) is the optimal uniform-average-rate for general coded caching (including unselfish and coded prefetching designs). Subsection III-B will describe a coded caching scheme with selfish and uncoded prefetching which achieves this bound for $\alpha = 1$. Therefore the $\tilde{R}$ in Prop. 1 is also the optimal uniform-average-rate for selfish and uncoded prefetching designs when $\alpha = 1$.

B. Minimum average rate with selfish and uncoded prefetching for $\alpha \geq 1$

Selfish and uncoded prefetching is not optimal in general, since it imposes unnecessary restrictions on the content of the caches. However, there are some cases in which it can proved that selfish and/or uncoded prefetching is optimal. One such example is Prop. 1 above and another example is provided in [15] for dominant $\Theta_1 = \{1\}$ and $\Theta_2 = \{1, \ldots, N\}$, $N \geq 2$. Furthermore, as discussed in Section I the simplicity of selfish and uncoded prefetching is a significant advantage when files, users, or links change dynamically.

This section provides an exact characterization of how the caches should be populated for a general M_1, M_2, N_1, N_2, and α (number of common files) in the case of $K = 2$ users with selfish and uncoded prefetching.

Lemma 1. *Without loss of generality, the optimal selfish and uncoded prefetching schemes for* $K = 2$ *users satisfy the following rules:*

1) If a user is devoting a portion C *of its cache to prefetch* γ *common files, this capacity should be evenly distributed among the files, i.e., a portion* $\frac{C}{\gamma}$ *of its cache to each of the common files. The same holds for unique (non-common) files.*

2) If both users are to cache a part of the same file, their cached portions should overlap as little as possible.

Proof. The Lemma can be straightforwardly proven by using symmetry and multicasting gain. See [17] for details. □

Since we are assuming selfish and uncoded prefetching, each user will have to devote part of its cache to store common files and the rest to store its own unique files. Let M_1^c and M_2^c denote the amount of cache capacity devoted to common files by users 1 and 2, respectively, and M_1^u and M_2^u that devoted to unique files. Assuming that no capacity goes unused, we have $M_i^c + M_i^u = M_i$ for $i = 1, 2$. The average rate with selfish and uncoded prefetching is given by

$$\tilde{R}_{\text{su}} = \sum_{j=1}^{5} p_j R_j, \tag{9}$$

where p_j and R_j, $j = 1, \ldots, 5$ correspond to the probability and transmission rate for different types of requests. Their expressions are shown in Table I, with $f_{ci} = \frac{M_i^c}{\alpha}$, $f_{ui} = \frac{M_i^u}{N_i - \alpha}$ respectively representing the fraction of each common and unique file being cached by user $i = 1, 2$.

The following two propositions optimize the selfish and uncoded prefetching to minimize the uniform-average rate $\tilde{R}_{\text{su}}$ in a system with a single server and $K = 2$ end users with arbitrary cache capacities and file demand sets. When there are multiple such schemes yielding the same $\tilde{R}_{\text{su}}$, our results provide a full characterization of the optimal solution set of

TABLE I
PROBABILITY MASS AND TRANSMISSION RATE FOR DIFFERENT TYPES OF USERS' REQUESTS.

Request d_1	Request d_2	Probability $p_{\vec{d}}$	Rate
unique	unique	$p_1 = \frac{(N_1-\alpha)(N_2-\alpha)}{N_1 N_2}$	$R_1 = (1-f_{u1})+(1-f_{u2})$
unique	common	$p_2 = \frac{(N_1-\alpha)\alpha}{N_1 N_2}$	$R_2 = (1-f_{c2})+(1-f_{u1})$
common	unique	$p_3 = \frac{\alpha(N_2-\alpha)}{N_1 N_2}$	$R_3 = (1-f_{c1})+(1-f_{u2})$
Same common file		$p_4 = \frac{\alpha}{N_1 N_2}$	$R_4 = 1-\min(f_{c1}, f_{c2})$
Different common files		$p_5 = \frac{\alpha(\alpha-1)}{N_1 N_2}$	$R_5 = 2-f_{c1}-f_{c2}-\min(f_{c1}, f_{c2}, 1-f_{c1}, 1-f_{c2})$

(M_1^c, M_2^c). Prop. 2 focuses on the case $\alpha = 1$ and Prop. 3 on $\alpha \geq 2$. In all cases, it is enough to specify how each user should split its cache between common and unique files, since Lemma 1 can then be used to determine a suitable set of file segments for each user to cache. In order to simplify the equations, we will denote the number of unique files in each demand set as $\beta_1 = N_1 - \alpha$ and $\beta_2 = N_2 - \alpha$.

Proposition 2. *When $\alpha = 1$, the average rate is minimized by the following placement scheme: cache unique files first. If there is space left after having cached all of them, then fill it with common file segments. More generally, the optimal set of cache distributions (M_1^c, M_2^c) is:*

$$\max(0, M_1-\beta_1) \leq M_1^c \leq \max(M_1-\beta_1, M_2-\beta_2) \quad (10)$$
$$\max(0, M_2-\beta_2) \leq M_2^c \leq \max(M_1-\beta_1, M_2-\beta_2). \quad (11)$$

Proof. See proof for Prop 3. □

The average rate $\tilde{R}_{\text{su}}$ resulting from the placement scheme in Prop. 2 matches that in Prop. 1 and Fig. 1, as expected.

Proposition 3. *When $\alpha \geq 2$, the average rate is minimized by the following placement scheme:*

1) *Cache common files in the smallest among M_1 and M_2, up to a maximum of half from each common file. Then have the other user cache an identical (but disjoint) fraction of common files.*
2) *If any cache capacity remains after 1, it should be devoted to caching unique files.*
3) *If a cache is still not full after 1 and 2, a user should cache additional segments from common files.*

The general equations characterizing the optimal set of cache distributions (M_1^c, M_2^c) for $\alpha \geq 2$ are somewhat tedious, see [17] for details, but the optimal set can be easily found from Fig. 2 and 3 for any given $\alpha, M_1, M_2, N_1, N_2$.

Proof. The average rate in Eq. (9) and Table I is piecewise linear in M_1^c and M_2^c, with four linear regions corresponding to which of the following is minimum: $M_1^c, M_2^c, \alpha - M_1^c, \alpha - M_2^c$, as shown in Fig. 2. The problem that we aim to solve is

$$\underset{M_i^c, M_i^u, i\in\{1,2\}}{\text{Minimize}} \quad \tilde{R}_{\text{su}} \quad (12)$$
$$\text{subject to} \quad M_i^c + M_i^u = M_i, \quad i \in \{1,2\} \quad (13)$$
$$0 \leq M_i^c \leq \alpha, \quad i \in \{1,2\} \quad (14)$$
$$0 \leq M_i^u \leq \beta_i = N_i - \alpha, \quad i \in \{1,2\} \quad (15)$$

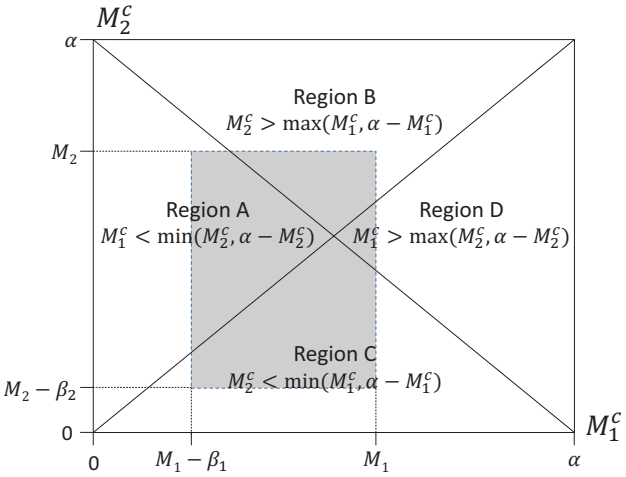

Fig. 2. Average rate is piecewise linear in the fractions of overlapping files being cached by each user. The feasible region is shaded.

Eq. (13) allows us to express $M_i^u = M_i - M_i^c$ for $i = 1, 2$. The feasible region of joint (13), (14), and (15) is thus $\max(0, M_i - \beta_i) \leq M_i^c \leq \min(M_i, \alpha)$ for $i = 1, 2$ as shaded in Fig. 2. The gradient of $\tilde{R}_{\text{su}}$ with respect to M_1^c and M_2^c is given by

- Region A: $\nabla \tilde{R}_{\text{su}}(M_1^c, M_2^c) = \frac{1}{N_1 N_2}(1-\alpha, 1)$
- Region B: $\nabla \tilde{R}_{\text{su}}(M_1^c, M_2^c) = \frac{1}{N_1 N_2}(0, \alpha)$
- Region C: $\nabla \tilde{R}_{\text{su}}(M_1^c, M_2^c) = \frac{1}{N_1 N_2}(1, 1-\alpha)$
- Region D: $\nabla \tilde{R}_{\text{su}}(M_1^c, M_2^c) = \frac{1}{N_1 N_2}(\alpha, 0)$

These gradients are illustrated in Fig. 3.

When $\alpha = 1$, the gradients point away from the origin, so the average rate is minimized when M_1^c and M_2^c are as small as possible. This proves Prop. 2.

When $\alpha = 2$, the gradients point away from the lower left diagonal, so the average rate is minimized when $M_1^c = M_2^c$ and they are less than $\frac{\alpha}{2}$, or as small as possible if that is not feasible (i.e., if $\max(M_1 - N_1 + \alpha, M_2 - N_2 + \alpha) > \frac{\alpha}{2}$).

When $\alpha > 2$, the gradients point away from the center for all four regions and the minimum average rate is achieved by $M_1^c = M_2^c = \min(\frac{\alpha}{2}, M_1, M_2)$, when feasible.

The three caching schemes proposed in Props. 2-3 fulfill those conditions. □

Corollary 1. *If we assume $\min(M_1, M_2) \geq 1$ and $\max(M_1 - \beta_1, M_2 - \beta_2) \leq 1$ as shown in Fig. 3, the optimal set for $\alpha = 2$*

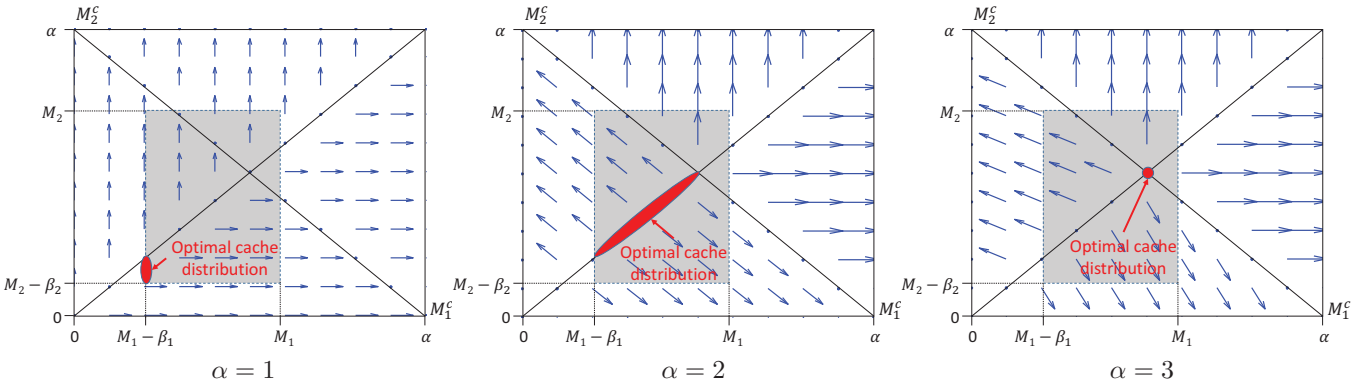

Fig. 3. Gradient fields of $\tilde{R}_{\text{su}}$ respect to M_1^c, M_2^c, for different values of α. The optimal (M_1^c, M_2^c) which minimize the average rate within the feasible region (shaded) is also shown.

is given by:

$$\max(0, \beta) \leq M_1^c = M_2^c \leq \min\left(\frac{\alpha}{2}, M_1, M_2\right). \tag{16}$$

Under those same assumptions, the optimal (unique) solution when $\alpha > 2$ is $M_1^c = M_2^c = \frac{\alpha}{2}$.

The uniform-average capacity of a coded caching scheme with selfish and uncoded prefetching when $\alpha \geq 2$ can be found by evaluating Eq. (9) with the policy in Prop. 3. Specifically, when $M_2 \geq M_1$, $\tilde{R}_{\text{su}}$ is tightly characterized by

$$\tilde{R}_{\text{su}} \geq 0 \tag{Q0}$$

$$\tilde{R}_{\text{su}} \geq 1 - \frac{M_1}{N_1} \tag{Q1}$$

$$\tilde{R}_{\text{su}} \geq 2 - \frac{M_1}{N_1} - \frac{M_2}{N_2} - \frac{\alpha(N_2 - M_2)}{N_1 N_2} \tag{Q2}$$

$$\tilde{R}_{\text{su}} \geq 2 - \frac{M_1}{N_1} - \frac{M_2}{N_2} - \frac{\alpha^2}{2N_1 N_2} \tag{Q3}$$

$$\tilde{R}_{\text{su}} \geq 2 - \frac{M_1}{N_1} - \frac{M_2}{N_2} - \frac{N_2 - M_2 + M_1(\alpha - 1)}{N_1 N_2} \tag{Q4}$$

$$\tilde{R}_{\text{su}} \geq 2 - \frac{M_1}{N_1} - \frac{M_2}{N_2} - \frac{M_2(\alpha - 2) + \alpha}{N_1 N_2} \tag{Q5}$$

for the regions illustrated in Fig. 4. The equations for the remaining regions (i.e., $M_1 > M_2$) can be found by symmetry, exchanging the roles of users 1 and 2.

IV. Simulations

This section illustrates the average rate capacity expressions in Fig. 4 for the particular case of $N_1 = N_2 = 128$ and $M_1 = M_2 = M$. It will compare the average rate with the optimal selfish and uncoded prefetching scheme described in Prop. 3 with two others:

- Uncoded transmission (UT): during the delivery phase, transmissions cannot encode multiple file segments into a single message. The server will send to each user whatever file segments it is missing from its cache, uncoded. There is no multicasting gain except when both users demand the same file, hence selfish prefetching is optimal. The minimum uniform-average rate is

$$\tilde{R}_{\text{ut}} \geq \left(2 - \frac{\alpha}{N_x^2}\right)\left(1 - \frac{M}{N_x}\right), \tag{17}$$

  where N_x denotes the value of $N_1 = N_2$.
- Shared caches (SC): users can share the content of their caches with each other. The two caches can therefore be treated as a single memory of size $2M$ that both users can access, without restricting the prefetching to be selfish or uncoded. The minimum uniform-average rate is

$$\tilde{R}_{\text{sc}} \geq \left(2 - \frac{\alpha}{N_x^2}\right)\left(1 - \frac{2M}{2N_x - \alpha}\right). \tag{18}$$

The UT scheme adds constraints to our system and therefore provides an upper bound to the general uniform-average rate capacity, while the SC scheme adds capabilities that our system did not have and therefore provides a lower bound to

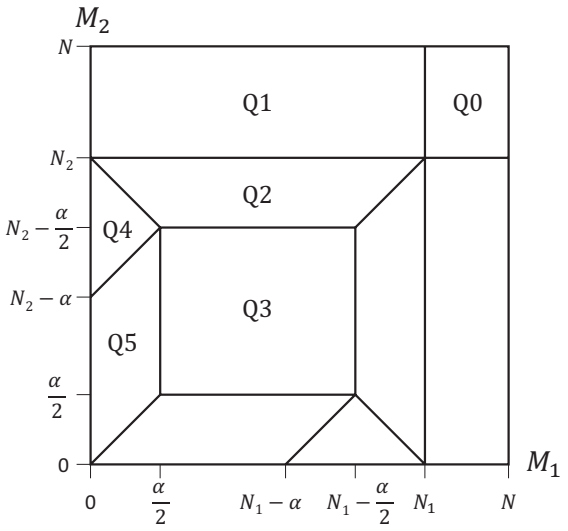

Fig. 4. The minimum average rate $\tilde{R}$ for $|\Theta_1| = N_1$, $|\Theta_2| = N_2$, and $\alpha \geq 2$ with selfish and uncoded prefetching.

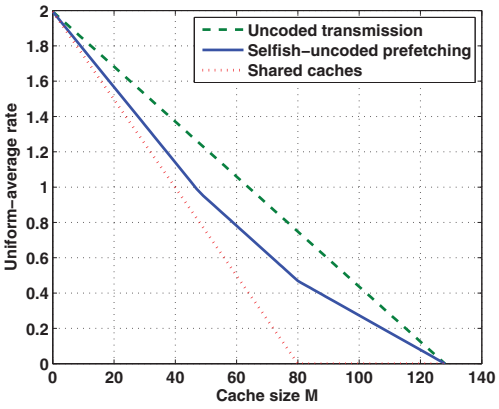

Fig. 5. Uniform-average rate as a function of the cache size M when $N_1 = N_2 = 128$ and $\alpha = 96$.

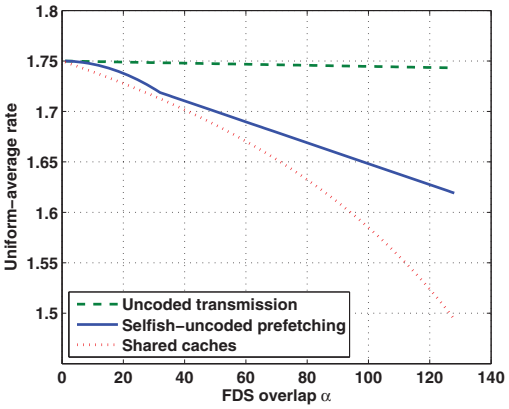

Fig. 6. Uniform-average rate as a function of the FDS overlap α when $N_1 = N_2 = 128$ and $M = 16$.

the general (including unselfish and coded prefetching) coded caching capacity.

Fig. 5 presents the uniform-average rates (normalized by the file size) for the three schemes when $\alpha = 96$ and the cache size per user grows from nothing ($M = 0$) to being able to cache all the files in the FDS ($M = 128$). The rate with our scheme falls between the other two, as expected, but it is interesting to note that for small M it is a lot closer to the SC than to the UT. This tells us that, even if we removed the selfish and uncoded prefetching restrictions, we would not be able to achieve a significant reduction in the rate.

Fig. 6 presents the uniform-average rates (normalized by the file size) for the three schemes when $M = 16$ and the intersection between the file demand sets (FDS) grows from empty ($\alpha = 0$) to a complete overlap ($\alpha = 128$). It can be observed that the UT scheme yields nearly constant rate regardless of the number of common files. This can be explained by the fact that, with $N_x = 128$ files in each demand set, the probability of both users requesting the same file is fairly low, even with a complete overlap. The uniform-average rate of our scheme is again relatively close to the capacity lower bound, specially when the overlap between the users' interests is small.

V. Conclusion

This paper studied the average rate capacity of a coded caching system with two users having arbitrary cache capacities and demanding files from different but overlapping demand sets. It characterized the general capacity when both sets overlap on a single file, and concluded that such minimum average rate can be achieved even if each user can only cache uncoded file segments from its own file demand set. The paper then focused on this scenario of selfish and uncoded prefetching and derived explicit expressions for the minimal achievable uniform-average rate when the file demand sets have arbitrary size and number of overlap. Finally, we provided detailed placement and delivery schemes capable of achieving the derived capacities. Simulation results were used to illustrate our results and bound their gap to an optimal unselfish and coded prefetching scheme.

References

[1] S. Borst, V. Gupta, and A. Walid, "Distributed caching algorithms for content distribution networks," in *Proc. IEEE INFOCOM*, March 2010.
[2] P. Krishnan, D. Raz, and Y. Shavitt, "The cache location problem," *IEEE/ACM Trans. Netw.*, vol. 8, no. 5, pp. 568–582, Oct 2000.
[3] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Trans. Inf. Theory*, vol. 60, no. 5, pp. 2856–2867, May 2014.
[4] C. Tian, "Symmetry, demand types and outer bounds in caching systems," in *Proc. IEEE Int. Symp. Inform. Theory*, Jul. 2016, pp. 825–829.
[5] D. Cao, D. Zhang, P. Chen, N. Liu, W. Kang, and D. Gunduz, "Coded caching with heterogeneous cache sizes and link qualities: The two-user case," in *Proc. IEEE Int. Symp. Inform. Theory*, 2018, pp. 1545–1549.
[6] C. Chang and C. Wang, "Coded caching with full heterogeneity: Exact capacity of the two-user/two-file case," in *Proc. IEEE Int. Symp. Inform. Theory*, July 2019, pp. 6–10.
[7] M. A. Maddah-Ali and U. Niesen, "Decentralized coded caching attains order-optimal memory-rate tradeoff," *IEEE/ACM Trans. Netw.*, vol. 23, no. 4, pp. 1029–1040, Aug 2015.
[8] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "Characterizing the rate-memory tradeoff in cache networks within a factor of 2," *IEEE Trans. Inf. Theory*, vol. 65, no. 1, pp. 647–663, Jan 2019.
[9] T. Luo, V. Aggarwal, and B. Peleato, "Coded caching with distributed storage," *IEEE Trans. Inf. Theory*, pp. 1–1, 2019.
[10] T. Luo and B. Peleato, "The transfer load-i/o trade-off for coded caching," *IEEE Commun. Lett.*, 2018.
[11] U. Niesen and M. A. Maddah-Ali, "Coded caching with nonuniform demands," *IEEE Trans. Inf. Theory*, vol. 63, no. 2, pp. 1146–1158, 2017.
[12] J. Zhang, X. Lin, and X. Wang, "Coded caching under arbitrary popularity distributions," *IEEE Trans. Inf. Theory*, vol. 64, no. 1, pp. 349–366, Jan. 2018.
[13] M. Ji, A. M. Tulino, J. Llorca, and G. Caire, "Order-optimal rate of caching and coded multicasting with random demands," *IEEE Trans. Inf. Theory*, vol. 63, no. 6, pp. 3923–3949, Jun. 2017.
[14] J. Hachem, N. Karamchandani, and S. N. Diggavi, "Coded caching for multi-level popularity and access," *IEEE Trans. Inf. Theory*, vol. 63, no. 5, pp. 3108–3141, May 2017.
[15] C. Chang and C. Wang, "Coded caching with heterogeneous file demand sets — the insufficiency of selfish coded caching," in *Proc. IEEE Int. Symp. Inform. Theory*, July 2019, pp. 1–5.
[16] S. Wang and B. Peleato, "Coded caching with heterogeneous user profiles," in *Proc. IEEE Int. Symp. Inform. Theory*, July 2019.
[17] https://engineering.purdue.edu/~chihw/pub_pdf/ICC2020_FDS_overlapping.pdf.