The Exact Load-Memory Tradeoff of Multi-Access Coded Caching With Combinatorial Topology

Federico Brunero, Petros Elia

The Exact Load-Memory Tradeoff of Multi-Access Coded Caching With Combinatorial Topology

Federico Brunero and Petros Elia
Communication Systems Department, EURECOM, Sophia Antipolis, France
Email: {brunero, elia}@eurecom.fr

***Abstract*—Recently, Muralidhar *et al.* proposed a novel multi-access system model where each user is connected to multiple caches in a manner that follows the well-known combinatorial topology of combination networks. For such multi-access topology, the same authors proposed an achievable scheme, which stands out for the unprecedented coding gains even with very modest cache resources. In this paper, we identify the fundamental limits of such multi-access setting with exceptional potential, providing an information-theoretic converse which establishes, together with the inner bound by Muralidhar *et al.*, the exact optimal performance under uncoded prefetching.**

***Index Terms*—Coded caching, combinatorial topology, index coding, information-theoretic converse, multi-access coded caching (MACC).**

I. Introduction

Coded caching was introduced in [1] as a coding-based communication technique with the purpose of reducing significantly the amount of data to be transferred from a centralized server to its cache-aided receiving users. The key idea behind coded caching involves an accurate joint design of the *placement phase* and the *delivery phase*. During the placement phase, which happens during off-peak hours, the caches of the users are preemptively filled without knowing the future requests. The caching is carefully performed so that, once the requests of the users are revealed, the amount of bits to be transferred during the delivery phase, which takes place when the network is saturated, is minimized. In the standard single-stream broadcast channel model with K receiving users, each of which is able to store in its cache a fraction γ of the main library, coded caching is able to provide a sizeable coding gain equal to $K\gamma+1$, where such coding gain represents simply the number of users to which a coded message is useful at the same time.

Current research on coded caching spans several topics such as the impact of multiple antennas on caching [2]–[4], the interplay between caching and file popularity [5]–[7], and a variety of other scenarios [8]–[10]. For a thorough review of the existing coded caching works, we strongly encourage the reader to refer to the longer version of this work [11].

A. Multi-Access Coded Caching

Differently from the model in [1] where each user has access to its own single dedicated cache, it is conceivable that in several scenarios each cache serves more than one user, and that each user can connect to more than one cache. For instance, in dense cellular networks, the cache-aided access points (APs) could have overlapping coverage areas, allowing in this way each user to connect to more than one AP. Such scenario motivated the work in [7], where the authors introduced a new parameter λ to keep into consideration the number of caches that each user can access. The corresponding model defined by λ as well as by the number of users K, the number of library files N, and the cache size of M files was referred to as the multi-access coded caching (MACC) model. Such model includes Λ caches and $K=\Lambda$ users, where each of them is connected to $\lambda>1$ consecutive caches in a cyclic wrap-around fashion.

Since the introduction of the aforementioned multi-access model with cyclic wrap-around topology, various works focused on the design of coding schemes that leverage the multi-access nature of the problem. For instance, a caching-and-delivery scheme was first proposed in the original work [7] with a decentralized (stochastic) cache placement, where this scheme provided improved local caching gains. Another achievable scheme preserving both the full local caching gain and the optimal coding gain of $\Lambda\lambda\gamma+1$ was instead presented in [12], albeit for the rather unrealistically demanding scenario where $\lambda=(\Lambda-1)/\Lambda\gamma$. Another notable work is then [13], where the authors designed a novel scheme for any $\lambda\geq 1$, which was proved, for the similarly demanding regime of $\lambda\geq\Lambda/2$ and $\Lambda\gamma\leq 2$, to be at a factor of at most 2 from the optimal under the assumption of uncoded placement. Other relevant works investigated the MACC problem and its connection to topics such as privacy and secrecy [14], [15], structured index coding problems [16] and PDA designs [17].

Recently, some interest arose towards new multi-access models which deviate from the cyclic wrap-around topology in [7]. For example, a new MACC paradigm was presented in [18], which involved topologies that are inspired by cross resolvable designs (CRDs), a special class of designs in combinatorics. However, a substantial breakthrough came with the work in [19], where the authors proposed a MACC model enjoying the same amount of resources λ and $\Lambda\gamma$, but where now the users and the caches are connected following the well-known combinatorial topology of combination networks [20]. This was a breakthrough because it allowed for the deployment of a subsequent scheme, presented in [19] as a generalization of the original Maddah-Ali and Niesen (MAN) scheme in [1],

This work was supported by the European Research Council (ERC) through the EU Horizon 2020 Research and Innovation Program under Grant 725929 (Project DUALITY). An extended version of this work [11] has been submitted to IEEE Transactions on Information Theory.

that achieves the astounding coding gain $\binom{\Lambda\gamma+\lambda}{\lambda}$ far exceeding $\Lambda\gamma+1$ even for small values of λ and $\Lambda\gamma$, which is the regime that really matters.

B. Main Contributions

Our work explores the fundamental limits of this undoubtedly powerful MACC model by Muralidhar *et al.*, which based on the findings in [19] has astounding performance. Such fundamental limits had remained entirely unknown as no information-theoretic converse has ever been developed. We here establish the exact optimal performance with the introduction of a novel information-theoretic lower bound on the optimal worst-case communication load under the assumption of uncoded placement. Further results are presented in [11], such as a generalization of the achievable scheme in [19] as well as novel information-theoretic converses for the topology-agnostic multi-access setting, i.e., the setting where it is not known a priori how the K users are connected to the Λ caches in the system.

C. Paper Outline

The paper is organized as follows. The MACC model and some preliminary definitions are presented in Section II. Section III presents the information-theoretic converse, whereas its general proof is described in Section IV. Section V concludes the paper. The appendices hold supplementary material.

D. Notation

We denote by $\mathbb{Z}^+$ the set of positive integers. For $n \in \mathbb{Z}^+$, we define $[n] \coloneqq \{1, \dots, n\}$. If $a, b \in \mathbb{Z}^+$ such that $a < b$, then $[a : b] \coloneqq \{a, a+1, \dots, b-1, b\}$. For sets we use calligraphic symbols, whereas for vectors we use bold symbols. Given a finite set $\mathcal{A}$, we denote by $|\mathcal{A}|$ its cardinality. We denote by $\binom{n}{k}$ the binomial coefficient and we let $\binom{n}{k} = 0$ whenever $n < 0$, $k < 0$ or $n < k$.

II. System Model

We consider the centralized coded caching scenario where one single server has access to a library $\mathcal{L} = \{W_n : n \in [N]\}$ containing N files of B bits each. The server is connected to K users through an error-free broadcast link. In the system there are Λ caches, each of size MB bits. In agreement with the system model in [19], each user is connected *exactly* and *uniquely* to a subset of λ caches for some fixed value of $\lambda \in [\Lambda]$, which consequently implies that there are $K = \binom{\Lambda}{\lambda}$ users for any given Λ and $\lambda \in [\Lambda]$. We denote[1] by $\mathcal{U} \subseteq [\Lambda]$ the user connected to the $|\mathcal{U}| = \lambda$ caches in the set $\mathcal{U}$. We further assume that the link between the server and the users is the main bottleneck, whereas we assume that the channel between each user and its assigned caches has infinite capacity. As is common, we assume that $N \geq K$. Such setting is completely described by the tuple (Λ, λ, N, M) and we refer to it as the MACC problem with combinatorial topology.

[1]For ease of notation, we will often omit braces and commas when indicating sets. For instance, user $\{1, 2\}$ in Fig. 1 is denoted as 12.

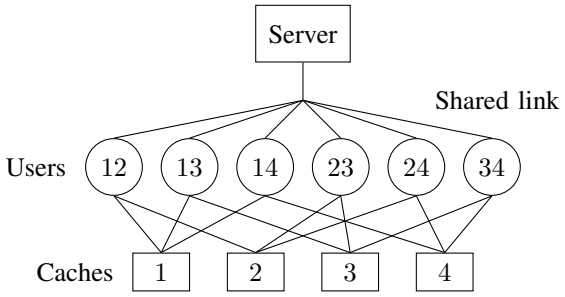

Fig. 1. MACC problem with combinatorial topology and $\Lambda = 4$ caches, where each user is connected exactly and uniquely to a subset of $\lambda = 2$ caches.

The communication procedure works as follows. During the placement phase, which typically occurs well before the delivery phase, the central server fills the caches without any knowledge of the future requests from the users. The delivery phase, which typically happens when the network is saturated and interference-limited, commences when the file-requests of all users are simultaneously revealed. During delivery, the server prepares some coded messages which are sent over the bottleneck shared link, so that each user can retrieve the missing information from the received transmission. The users cancel the interference terms that appear in the broadcast transmission, and do so by means of the cached contents they have access to, eventually decoding their own messages. As a consequence, the worst-case communication load R is defined as the total number of transmitted bits, normalized by the file-size B, that can guarantee the correct delivery of any K-tuple of requested files in the worst-case scenario. The optimal communication load $R^\star$ is formally defined as

$$R^\star(M) \coloneqq \inf\{R : \ (M, R) \text{ is achievable}\} \tag{1}$$

where the tuple (M, R) is said to be *achievable* if there exists a caching-and-delivery procedure for which, for any possible demand, a load R can be guaranteed for a given memory value M.

We use the notation $W_{d_\mathcal{U}}$ to denote the file requested by the user identified by $\mathcal{U}$ for some $\mathcal{U} \subseteq [\Lambda]$ with $|\mathcal{U}| = \lambda$, which we remind the reader is simply the user connected exactly and uniquely to the λ caches in the set $\mathcal{U}$. For the sake of simplicity, we denote by $\boldsymbol{d} = (d_\mathcal{U} : \mathcal{U} \subseteq [\Lambda], |\mathcal{U}| = \lambda)$ the demand vector containing the indices of the files requested by the users in the system, i.e., $d_\mathcal{U} \in [N]$ for each $\mathcal{U} \subseteq [\Lambda]$ with $|\mathcal{U}| = \lambda$. The following example can help familiarize the reader with the setting.

Example 1 ($\Lambda = 4, \lambda = 2, N, M$)**.** Consider the MACC problem with combinatorial topology and $\Lambda = 4$ caches in Fig. 1. Since each set of $\lambda = 2$ caches is uniquely assigned to a user, there are $K = \binom{\Lambda}{\lambda} = 6$ users in total. Recalling that each user is identified by the set of 2 caches it is connected to and that for simplicity we omit braces and commas when indicating sets, we let 12 represent the user connected to cache 1 and cache 2, we let 13 represent the user connected to cache 1 and cache 3, and so on. The demand vector is given by $\boldsymbol{d} =$

$(d_{12}, d_{13}, d_{14}, d_{23}, d_{24}, d_{34})$, where $d_{12} \in [N]$ is the index of the file requested by the user 12, $d_{13} \in [N]$ is the index of the file requested by the user 13, and so on.

Our goal is to provide a converse bound on the optimal worst-case load under the assumption of uncoded placement, whose definition is given in the following.

Definition 1 (Uncoded Cache Placement)**.** A cache placement is *uncoded* if the bits of the files are simply copied within the caches of the users.

Denoting by $R_{\mathrm{u}}^{\star}$ the optimal worst-case load under uncoded placement, it trivially holds $R_{\mathrm{u}}^{\star} \geq R^{\star}$.

III. Main Result

We present the main contribution of the paper. The derivation of the converse bound adopts the index coding technique first proposed in [21]. Our main challenge will be to design the converse in such a way that it tightly captures the fact that each user, differently from the original setting in [1], is connected to λ caches for some $\lambda \in [\Lambda]$. The result is stated in the following theorem.

Theorem 1. *Consider the multi-access coded caching problem with combinatorial topology with parameters* (Λ, λ, N, M)*. Under the assumption of uncoded cache placement, the optimal worst-case communication load* $R_{\mathrm{u}}^{\star}$ *is a piecewise linear curve with corner points*

$$(M, R_{\mathrm{u}}^{\star}) = \left(t\frac{N}{\Lambda}, \frac{\binom{\Lambda}{t+\lambda}}{\binom{\Lambda}{t}} \right), \quad \forall t \in [0:\Lambda]. \tag{2}$$

Proof. The proof of achievability comes from the coding scheme in [19], whereas the proof of the converse bound is presented in Section IV. □

The longer version in [11] of this work provides also another interesting result that we will simply mention here. The system model in [19] considers a single value of $\lambda \in [\Lambda]$ for a given Λ, although such setting can be extended into a *generalized* combinatorial multi-access model which supports the coexistence of groups of users connected to different numbers of caches. This extension can be obtained if we consider a simultaneous coexistence of each model in [19] for each $\lambda \in [\Lambda]$. The quite surprising outcome in [11] is summarized in the following remark.

Remark 1. For the MACC problem with generalized combinatorial topology, there is no need to encode over users which are connected to different numbers of caches, even though there is abundance of coding opportunities. Instead, applying the coding scheme in [19] in a TDMA-like manner is sufficient to achieve the optimal communication load under uncoded prefetching. An additional powerful insight that comes out of this is that the basic Λ-cache MAN placement proves to be extremely effective as it allows for the optimal performance for any instance of the generalized combinatorial topology.

We now proceed by presenting the converse proof of Theorem 1.

IV. Proof of the Converse Bound

The converse relies on the well-known acyclic subgraph index coding bound, which has been extensively used in various other settings (see for example [2], [21], [22] to name a few) in order to derive lower bounds on the optimal worst-case load in caching under the assumption of uncoded prefetching. To clarify the connection between this bound and our setting, we start by providing a brief presentation of the index coding problem and its connection to coded caching.

A. Definition of the Index Coding Problem

The index coding problem [23] consists of a central server having access to N' independent messages, and of K' users that are connected to the server via a shared error-free broadcast channel. Each user $k \in [K']$ has a set of desired messages $\mathcal{M}_k \subseteq [N']$, which is called the *desired message set*, while also having access to another subset of messages $\mathcal{A}_k \subseteq [N']$, which is called the *side information set*. To avoid trivial scenarios, it is commonly assumed that $\mathcal{M}_k \neq \emptyset$, $\mathcal{A}_k \neq [N']$ and $\mathcal{M}_k \cap \mathcal{A}_k = \emptyset$ for each $k \in [K']$.

The index coding problem is usually described in terms of its *side information graph*. Let M_i be the i-th message for some $i \in [N']$. Then, such graph is a directed graph where each vertex is a desired message and where there exists an edge from a desired message M_i to a desired message M_j if and only if message M_i is in the side information set of the user requesting message M_j. The derivation of our converse is based on the following bound from [24, Corollary 1].

Lemma 1 ([24, Corollary 1])**.** *Consider an index coding problem with* N' *messages* M_i *for* $i \in [N']$*. The minimum number of transmitted bits* ρ *is lower bounded as*

$$\rho \geq \sum_{i \in \mathcal{J}} |M_i| \tag{3}$$

for any acyclic subgraph $\mathcal{J}$ *of the side information graph.*

B. Main Proof

The first step in our converse proof consists of dividing, in the most generic manner, each file into a maximum of 2^{Λ} disjoint subfiles as

$$W_n = \{W_{n,\mathcal{T}} : \mathcal{T} \subseteq [\Lambda]\}, \quad \forall n \in [N] \tag{4}$$

where we identify with $W_{n,\mathcal{T}}$ the subfile which is exclusively stored by the caches in $\mathcal{T}$. Noticing that the bits of the library files are simply copied within the caches, such placement is uncoded according to Definition 1.

1) Constructing the Index Coding Bound: Assuming that each user requests a distinct[2] file, we consider the index coding problem with $K' = K = \binom{\Lambda}{\lambda}$ users and $N' = \binom{\Lambda}{\lambda} 2^{\Lambda - \lambda}$ independent messages, where each such message represents a

[2]The set of worst-case demands may not include the set of demand vectors $\boldsymbol{d}$ with all distinct entries. However, since our goal is to derive a converse bound on the worst-case load, this is not a problem. Indeed, the choice of treating distinct demands yields a valid converse bound, since such bound does not need to be, a priori, the tightest bound. In our case, it proves to be tight.

subfile requested by some user (who naturally does not have access to it via a cache). Recalling that $W_{d_{\mathcal{U}}}$ denotes the file requested by the user identified by $\mathcal{U}$, the desired message set and the side information set are respectively given, in their most generic form, by

$$\mathcal{M}_{\mathcal{U}} = \{W_{d_{\mathcal{U}},\mathcal{T}} : \mathcal{T} \subseteq ([\Lambda] \setminus \mathcal{U})\} \tag{5}$$

$$\mathcal{A}_{\mathcal{U}} = \{W_{n,\mathcal{T}} : n \in [N], \mathcal{T} \subseteq [\Lambda], \mathcal{T} \cap \mathcal{U} \neq \emptyset\} \tag{6}$$

for each user $\mathcal{U}$ with $\mathcal{U} \subseteq [\Lambda]$ and $|\mathcal{U}| = \lambda$. Here, the side information graph consists of a directed graph where each vertex is a subfile, and where there is an edge from the subfile $W_{d_{\mathcal{U}_1},\mathcal{T}_1}$ to the subfile $W_{d_{\mathcal{U}_2},\mathcal{T}_2}$ if and only if $W_{d_{\mathcal{U}_1},\mathcal{T}_1} \in \mathcal{A}_{\mathcal{U}_2}$ with $\mathcal{U}_j \subseteq [\Lambda]$, $|\mathcal{U}_j| = \lambda$, $\mathcal{T}_j \subseteq ([\Lambda] \setminus \mathcal{U}_j)$ for $j \in \{1, 2\}$ and $\mathcal{U}_1 \neq \mathcal{U}_2$. Since our aim is to apply Lemma 1, we need to consider acyclic sets of vertices $\mathcal{J}$ in the side information graph. Toward this, we take advantage of the following lemma.

Lemma 2. *Let $\boldsymbol{d} = (d_{\mathcal{U}} : \mathcal{U} \subseteq [\Lambda], |\mathcal{U}| = \lambda)$ be a demand vector and let $\boldsymbol{c} = (c_1, \ldots, c_\Lambda)$ be a permutation of the Λ caches. The following set of vertices*

$$\bigcup_{i\in[\lambda:\Lambda]} \bigcup_{\substack{\mathcal{U}^i \subseteq \{c_1,\ldots,c_i\}:|\mathcal{U}^i|=\lambda,\\ c_i\in\mathcal{U}^i}} \bigcup_{\mathcal{T}_i\subseteq([\Lambda]\setminus\{c_1,\ldots,c_i\})} \{W_{d_{\mathcal{U}^i},\mathcal{T}_i}\} \tag{7}$$

is acyclic.

Proof. The proof is provided in Appendix A. An illustrative example is then described in Appendix B. □

Consider a demand vector $\boldsymbol{d}$ and a permutation $\boldsymbol{c}$ of the set $[\Lambda]$. Applying Lemma 2 yields the following lower bound

$$BR_{\mathrm{u}}^\star \geq R(\boldsymbol{d}, \boldsymbol{c}) \tag{8}$$

where $R(\boldsymbol{d}, \boldsymbol{c})$ is defined as

$$R(\boldsymbol{d}, \boldsymbol{c}) \coloneqq \sum_{i=\lambda}^{\Lambda} \sum_{\substack{\mathcal{U}^i \subseteq \{c_1,\ldots,c_i\}:|\mathcal{U}^i|=\lambda,\\ c_i\in\mathcal{U}^i}} \sum_{\mathcal{T}_i\subseteq\{c_{i+1},\ldots,c_K\}} \left|W_{d_{\mathcal{U}^i},\mathcal{T}_i}\right|. \tag{9}$$

2) Constructing the Optimization Problem: Now our goal is to create several bounds as the one in (8) considering any vector $\boldsymbol{d} \in \mathcal{D}$ and any vector $\boldsymbol{c} \in \mathcal{C}$, where we denote by $\mathcal{D}$ and $\mathcal{C}$ the set of possible demand vectors with distinct entries and the set of possible permutation vectors of the set $[\Lambda]$, respectively. Our aim is then to average all these bounds to obtain in the end a useful lower bound on the optimal worst-case load. Considering that $|\mathcal{D}| = \binom{N}{K}K!$ and $|\mathcal{C}| = \Lambda!$, we aim to simplify the expression given by

$$\binom{N}{K}K!\Lambda!BR_{\mathrm{u}}^\star \geq \sum_{\boldsymbol{d}\in\mathcal{D}} \sum_{\boldsymbol{c}\in\mathcal{C}} R(\boldsymbol{d}, \boldsymbol{c}). \tag{10}$$

Toward simplifying (10), we proceed by counting how many times each subfile $W_{n,\mathcal{T}}$ — for any given $n \in [N]$, $\mathcal{T} \subseteq [\Lambda]$ and $|\mathcal{T}| = t'$ for some $t' \in [0 : \Lambda]$ — appears in (10).

Let us focus on the subfile $W_{n,\mathcal{T}}$ for some $n \in [N]$, $\mathcal{T} \subseteq [\Lambda]$ and $|\mathcal{T}| = t'$ with $t' \in [0 : \Lambda]$. Assume that the file W_n is demanded by user $\mathcal{U}$ for some $\mathcal{U} \subseteq ([\Lambda] \setminus \mathcal{T})$ with $|\mathcal{U}| = \lambda$ and denote by $\mathcal{D}_{n,\mathcal{U}}$ the set of demands such that $d_{\mathcal{U}} = n$. Out of the entire set $\mathcal{D}$ of all possible distinct demands, we find a total of $\binom{N}{K}K!/N$ distinct demands for which a file is requested by the same user. Hence, it holds $|\mathcal{D}_{n,\mathcal{U}}| = \binom{N}{K}K!/N$. For each $\boldsymbol{d} \in \mathcal{D}_{n,\mathcal{U}}$ and for each $\boldsymbol{c} \in \mathcal{C}$ there is a corresponding bound $R(\boldsymbol{d}, \boldsymbol{c})$. The subfile $W_{n,\mathcal{T}}$ appears only in the bounds induced by permutation vectors $\boldsymbol{c} \in \mathcal{C}$ such that the elements in the set $\mathcal{U}$ appear in the vector $\boldsymbol{c}$ before[3] the elements in the set $\mathcal{T}$. If we denote by $\mathcal{C}_{\mathcal{U},\mathcal{T}}$ the set of such permutation vectors, it can be verified that $|\mathcal{C}_{\mathcal{U},\mathcal{T}}| = \lambda!t'!(\Lambda-\lambda-t')!\binom{\Lambda}{t'+\lambda}$. Hence, the subfile $W_{n,\mathcal{T}}$ is counted a total of $\lambda!t'!(\Lambda - \lambda - t')!\binom{\Lambda}{t'+\lambda}\binom{N}{K}K!/N$ times when considering the bounds $R(\boldsymbol{d}, \boldsymbol{c})$ with $\boldsymbol{d} \in \mathcal{D}_{n,\mathcal{U}}$ and $\boldsymbol{c} \in \mathcal{C}_{\mathcal{U},\mathcal{T}}$. Since the same reasoning follows for each $\mathcal{U} \subseteq ([\Lambda] \setminus \mathcal{T})$ with $|\mathcal{U}| = \lambda$, the subfile $W_{n,\mathcal{T}}$ is counted a total of

$$a_{t'} = \binom{\Lambda - t'}{\lambda}\lambda!t'!(\Lambda - \lambda - t')!\binom{\Lambda}{t' + \lambda}\frac{\binom{N}{K}K!}{N} \tag{11}$$

times, which gives us the number of times this same subfile appears in (10). The same reasoning follows for any $n \in [N]$ and for any $\mathcal{T} \subseteq [\Lambda]$ with $|\mathcal{T}| = t'$. Thus, the expression in (10) can be rewritten as

$$R_{\mathrm{u}}^\star \geq \frac{1}{\binom{N}{K}K!\Lambda!}\sum_{t'=0}^{\Lambda} Na_{t'}x_{t'} \tag{12}$$

$$= \sum_{t'=0}^{\Lambda} \frac{\binom{\Lambda}{t'+\lambda}}{\binom{\Lambda}{t'}}x_{t'} \tag{13}$$

$$= \sum_{t'=0}^{\Lambda} f(t')x_{t'} \tag{14}$$

where $f(t')$ and $x_{t'}$ are defined as

$$f(t') \coloneqq \frac{\binom{\Lambda}{t'+\lambda}}{\binom{\Lambda}{t'}} \tag{15}$$

$$0 \leq x_{t'} \coloneqq \sum_{n\in[N]} \sum_{\mathcal{T}\subseteq[\Lambda]:|\mathcal{T}|=t'} \frac{|W_{n,\mathcal{T}}|}{NB}. \tag{16}$$

At this point, we seek to lower bound the minimum worst-case load $R_{\mathrm{u}}^\star$ by lower bounding the solution to the following optimization problem

$$\min_{\boldsymbol{x}} \quad \sum_{t'=0}^{\Lambda} f(t')x_{t'} \tag{17a}$$

$$\text{subject to} \quad \sum_{t'=0}^{\Lambda} x_{t'} = 1 \tag{17b}$$

$$\sum_{t'=0}^{\Lambda} t'x_{t'} \leq \frac{\Lambda M}{N} \tag{17c}$$

where (17b) and (17c) correspond to the file-size constraint and the cumulative cache-size constraint, respectively.

[3]Indeed, the subfile $W_{n,\mathcal{T}}$ appears in the acyclic graph chosen as in Lemma 2 for all those permutations $\boldsymbol{c} = (c_1, \ldots, c_\Lambda)$ for which $\mathcal{U} = \mathcal{U}^i$ and $\mathcal{T} = \mathcal{T}_i$ for some $i \in [\lambda : \Lambda]$ such that $\mathcal{U}^i \subseteq \{c_1, \ldots, c_i\}$ with $|\mathcal{U}^i| = \lambda$ and $c_i \in \mathcal{U}^i$, and such that $\mathcal{T}_i \subseteq \{c_{i+1}, \ldots, c_\Lambda\}$, i.e., this happens whenever the elements in $\mathcal{T}$ are after the elements in $\mathcal{U}$ in the permutation vector $\boldsymbol{c}$.

3) Lower Bounding the Solution to the Optimization Problem: Since the auxiliary variable $x_{t'}$ can be considered as a probability mass function, the optimization problem in (17) can be seen as the minimization of $\mathbb{E}[f(t')]$. Moreover, the following holds.

Lemma 3. *The function $f(t')$ is convex and decreasing in t'.*

Proof. The proof is relegated to the longer version of this work [11], where the proof is provided for a generalization of the function here denoted by $f(t')$. □

Taking advantage of Lemma 3, we can write $\mathbb{E}[f(t')] \geq f(\mathbb{E}[t'])$ using Jensen's inequality. Then, considering that $f(t')$ is also decreasing with increasing $t' \in [0:\Lambda]$, we can further write $f(\mathbb{E}[t']) \geq f(\Lambda M/N)$ taking advantage of the fact that $\mathbb{E}[t']$ is upper bounded as in (17c). Consequently, $\mathbb{E}[f(t')] \geq f(\Lambda M/N)$, and thus for $t \coloneqq \Lambda M/N$ the optimal worst-case load $R_{\mathrm{u}}^{\star}$ is lower bounded by R_{LB} which is a piecewise linear curve with corner points

$$(M, R_{\mathrm{LB}}) = \left(t\frac{N}{\Lambda}, \frac{\binom{\Lambda}{t+\lambda}}{\binom{\Lambda}{t}} \right), \quad \forall t \in [0:\Lambda]. \tag{18}$$

This concludes the proof. □

V. Conclusion

In this work, we derived the fundamental limits of a coded caching scenario with exceptional potential. We proposed a novel information-theoretic converse that manages to capture the topological properties of the multi-access model in Section II. The lower bound matches the achievable performance of the coding scheme in [19], so allowing us to identify the exact optimal performance of multi-access caching with combinatorial topology under uncoded prefetching. Interestingly, our information-theoretic converse can be seen as a generalization of the lower bound in [21], which similarly proved the exact optimality of the MAN placement-and-delivery scheme.

As already mentioned, the longer version of this work in [11] provides a variety of interesting extensions. Indeed, the work in [11] not only provides a generalization of the combinatorial topology that allows for the coexistence of users connected to different numbers of caches, but also provides very interesting results regarding the topology-agnostic multi-access problem, offering novel lower bounds on the average worst-case performance when it is not known a priori how the K users are connected to the Λ caches in the system.

Appendix A
Proof of Lemma 2

Consider the set of vertices

$$\bigcup_{i\in[\lambda:\Lambda]} \bigcup_{\substack{\mathcal{U}^i\subseteq\{c_1,\ldots,c_i\}:|\mathcal{U}^i|=\lambda,\\ c_i\in\mathcal{U}^i}} \bigcup_{\mathcal{T}_i\subseteq([\Lambda]\setminus\{c_1,\ldots,c_i\})} \left\{W_{d_{\mathcal{U}^i},\mathcal{T}_i}\right\} \tag{19}$$

in Lemma 2. We can show that such set is guaranteed to be acyclic by following the same reasoning as in the proof of [21, Lemma 1].

For a specific permutation of caches $\boldsymbol{c} = (c_1, \ldots, c_\Lambda)$, we will say that the subfile $W_{d_{\mathcal{U}^i},\mathcal{T}_i}$ belongs to the i-th *level*[4], which will mean that $\mathcal{U}^i \subseteq \{c_1, \ldots, c_i\}$ with $|\mathcal{U}^i| = \lambda$ and $c_i \in \mathcal{U}^i$, and that $\mathcal{T}_i \subseteq ([\Lambda] \setminus \{c_1, \ldots, c_i\})$ with $i \in [\lambda : \Lambda]$. As one may see, no user in the i-th level has access to the subfiles in its level, since indeed $\mathcal{U}^i \cap \mathcal{T}_i = \emptyset$ for each $\mathcal{U}^i \subseteq \{c_1, \ldots, c_i\}$ with $|\mathcal{U}^i| = \lambda$ and $c_i \in \mathcal{U}^i$, and for each $\mathcal{T}_i \subseteq ([\Lambda] \setminus \{c_1, \ldots, c_i\})$. Moreover, no user in the i-th level has access to the subfiles in higher levels, since $\mathcal{U}^i \cap \mathcal{T}_j = \emptyset$ for each $\mathcal{U}^i \subseteq \{c_1, \ldots, c_i\}$ with $|\mathcal{U}^i| = \lambda$ and $c_i \in \mathcal{U}^i$, and for each $\mathcal{T}_j \subseteq ([\Lambda] \setminus \{c_1, \ldots, c_j\})$ with $j \in [i+1 : \Lambda]$. Consequently, we can conclude that the set in (19) is acyclic. This concludes the proof. □

Appendix B
Illustrative Example

To illustrate the main idea in Lemma 2, we provide in the following a simple example where we explicitly construct the acyclic set of vertices for two different permutations of caches.

Consider the setting where there are $\Lambda = 4$ caches and each user is connected to $\lambda = 2$ caches, which implies $K = 6$ users under the combinatorial topology. Assume the arbitrary demand vector $\boldsymbol{d} = (d_{12}, d_{13}, d_{14}, d_{23}, d_{24}, d_{34})$ and consider the permutation of caches $\boldsymbol{c} = (1, 4, 3, 2)$. According to Lemma 2, when $i = 2$, for the user identified by the set $\{1, 4\}$ we pick all the subfiles $W_{d_{14},\mathcal{T}_2}$ where $\mathcal{T}_2 \subseteq \{2, 3\}$, i.e., we pick $\{W_{d_{14},\emptyset}, W_{d_{14},2}, W_{d_{14},3}, W_{d_{14},23}\}$. When $i = 3$, for each user $\mathcal{U}^3 \in \{\{1, 3\}, \{3, 4\}\}$ we pick all the subfiles $W_{d_{\mathcal{U}^3},\mathcal{T}_3}$ where $\mathcal{T}_3 \subseteq \{2\}$, i.e., we pick $\{W_{d_{13},\emptyset}, W_{d_{13},2}, W_{d_{34},\emptyset}, W_{d_{34},2}, \}$. When $i = 4$, for each user $\mathcal{U}^4 \in \{\{1, 2\}, \{2, 4\}, \{2, 3\}\}$ we pick all the subfiles $W_{d_{\mathcal{U}^4},\mathcal{T}_4}$ where $\mathcal{T}_4 = \emptyset$, i.e., we pick $\{W_{d_{12},\emptyset}, W_{d_{24},\emptyset}, W_{d_{23},\emptyset}\}$. Hence, for the permutation vector $\boldsymbol{c} = (1, 4, 3, 2)$ the acyclic set of vertices from Lemma 2 is given by the set

$$\begin{aligned}\{&W_{d_{14},\emptyset}, W_{d_{14},2}, W_{d_{14},3}, W_{d_{14},23}, W_{d_{13},\emptyset}, W_{d_{13},2},\\ &W_{d_{34},\emptyset}, W_{d_{34},2}, W_{d_{12},\emptyset}, W_{d_{24},\emptyset}, W_{d_{23},\emptyset}\}.\end{aligned} \tag{20}$$

Consider now the cache permutation $\boldsymbol{c} = (2, 3, 1, 4)$. When $i = 2$, for the user identified by the set $\{2, 3\}$ we pick all the subfiles $W_{d_{23},\mathcal{T}_2}$ where $\mathcal{T}_2 \subseteq \{1, 4\}$, i.e., we pick $\{W_{d_{23},\emptyset}, W_{d_{23},1}, W_{d_{23},4}, W_{d_{23},14}\}$. When $i = 3$, for each user $\mathcal{U}^3 \in \{\{1, 2\}, \{1, 3\}\}$ we pick all the subfiles $W_{d_{\mathcal{U}^3},\mathcal{T}_3}$ where $\mathcal{T}_3 \subseteq \{4\}$, i.e., we pick $\{W_{d_{12},\emptyset}, W_{d_{12},4}, W_{d_{13},\emptyset}, W_{d_{13},4}\}$. When $i = 4$, for each user $\mathcal{U}^4 \in \{\{2, 4\}, \{3, 4\}, \{1, 4\}\}$, we pick all the subfiles $W_{d_{\mathcal{U}^4},\mathcal{T}_4}$ where $\mathcal{T}_4 = \emptyset$, i.e., we pick $\{W_{d_{24},\emptyset}, W_{d_{34},\emptyset}, W_{d_{14},\emptyset}\}$. Hence, for the permutation vector $\boldsymbol{c} = (2, 3, 1, 4)$ the acyclic set of vertices from Lemma 2 is given by the set

$$\begin{aligned}\{&W_{d_{23},\emptyset}, W_{d_{23},1}, W_{d_{23},4}, W_{d_{23},14}, W_{d_{12},\emptyset}, W_{d_{12},4},\\ &W_{d_{13},\emptyset}, W_{d_{13},4}, W_{d_{24},\emptyset}, W_{d_{34},\emptyset}, W_{d_{14},\emptyset}\}.\end{aligned} \tag{21}$$

[4]The term *level* carries the same meaning as in the proof of [21, Lemma 1], and its impact here is described mathematically in compliance with our setting.

REFERENCES

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

[2] E. Parrinello, A. Ünsal, and P. Elia, "Fundamental limits of coded caching with multiple antennas, shared caches and uncoded prefetching," *IEEE Trans. Inf. Theory*, vol. 66, no. 4, pp. 2252–2268, Apr. 2020.

[3] E. Lampiris, A. Bazco-Nogueras, and P. Elia, "Resolving the feedback bottleneck of multi-antenna coded caching," *IEEE Trans. Inf. Theory*, 2021, early access. doi: 10.1109/TIT.2021.3139013.

[4] A. Tölli, S. P. Shariatpanahi, J. Kaleva, and B. H. Khalaj, "Multi-antenna interference management for coded caching," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 2091–2106, Mar. 2020.

[5] 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.

[6] Z. Zhang and M. Tao, "Deep learning for wireless coded caching with unknown and time-variant content popularity," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 1152–1163, Feb. 2021.

[7] J. Hachem, N. Karamchandani, and S. Diggavi, "Coded caching for multi-level popularity and access," *IEEE Trans. Inf. Theory*, vol. 63, no. 5, pp. 3108–3141, May 2017.

[8] H. Joudeh, E. Lampiris, P. Elia, and G. Caire, "Fundamental limits of wireless caching under mixed cacheable and uncacheable traffic," *IEEE Trans. Inf. Theory*, vol. 67, no. 7, pp. 4747–4767, Jul. 2021.

[9] E. Lampiris and P. Elia, "Full coded caching gains for cache-less users," *IEEE Trans. Inf. Theory*, vol. 66, no. 12, pp. 7635–7651, Dec. 2020.

[10] H. Zhao, A. Bazco-Nogueras, and P. Elia, "Vector coded caching multiplicatively boosts the throughput of realistic downlink systems," Feb. 2022, arXiv: 2202.07047 [cs.IT].

[11] F. Brunero and P. Elia, "Fundamental limits of combinatorial multi-access caching," Oct. 2021, arXiv: 2110.07426 [cs.IT].

[12] B. Serbetci, E. Parrinello, and P. Elia, "Multi-access coded caching: gains beyond cache-redundancy," in *2019 IEEE Inf. Theory Workshop (ITW)*, Aug. 2019, pp. 1–5.

[13] K. S. Reddy and N. Karamchandani, "Rate-memory trade-off for multi-access coded caching with uncoded placement," *IEEE Trans. Commun.*, vol. 68, no. 6, pp. 3261–3274, Jun. 2020.

[14] K. K. K. Namboodiri and B. S. Rajan, "Multi-access coded caching with secure delivery," in *2021 IEEE Inf. Theory Workshop (ITW)*, Oct. 2021, pp. 1–6.

[15] ——, "Multi-access coded caching with demand privacy," Jul. 2021, arXiv: 2107.00226 [cs.IT].

[16] K. S. Reddy and N. Karamchandani, "Structured index coding problem and multi-access coded caching," *IEEE J. Sel. Areas Inf. Theory*, vol. 2, no. 4, pp. 1266–1281, Dec. 2021.

[17] M. Cheng, D. Liang, K. Wan, M. Zhang, and G. Caire, "A novel transformation approach of shared-link coded caching schemes for multiaccess networks," in *2021 IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2021, pp. 849–854.

[18] D. Katyal, P. N. Muralidhar, and B. S. Rajan, "Multi-access coded caching schemes from cross resolvable designs," *IEEE Trans. Commun.*, vol. 69, no. 5, pp. 2997–3010, May 2021.

[19] P. N. Muralidhar, D. Katyal, and B. S. Rajan, "Maddah-Ali-Niesen scheme for multi-access coded caching," in *2021 IEEE Inf. Theory Workshop (ITW)*, Oct. 2021, pp. 1–6.

[20] M. Ji *et al.*, "On the fundamental limits of caching in combination networks," in *2015 IEEE 16th Int. Workshop Signal Proc. Advances Wireless Commun. (SPAWC)*, Jun. 2015, pp. 695–699.

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

[22] F. Brunero and P. Elia, "Unselfish coded caching can yield unbounded gains over symmetrically selfish caching," Sep. 2021, arXiv: 2109.04807 [cs.IT].

[23] Z. Bar-Yossef, Y. Birk, T. S. Jayram, and T. Kol, "Index coding with side information," *IEEE Trans. Inf. Theory*, vol. 57, no. 3, pp. 1479–1494, Mar. 2011.

[24] F. Arbabjolfaei, B. Bandemer, Y.-H. Kim, E. Şaşoğlu, and L. Wang, "On the capacity region for index coding," in *2013 IEEE Int. Symp. Inf. Theory*, Jul. 2013, pp. 962–966.