Adding Transmitters Dramatically Boosts Coded-Caching Gains for Finite File Sizes

Eleftherios Lampiris and Petros Elia

***Abstract*— In the context of coded caching in the K-user broadcast channel, our work reveals the surprising fact that having multiple (L) transmitting antennas, dramatically ameliorates the long-standing subpacketization bottleneck of coded caching by reducing the required subpacketization to approximately its Lth root, thus boosting the actual DoF by a *multiplicative* factor of up to L. In asymptotic terms, this reveals that as long as L scales with the theoretical caching gain, then the full cumulative (multiplexing + full caching) gains are achieved with constant subpacketization. This is the first time, in any known setting, that unbounded caching gains appear under finite file-size constraints. The achieved caching gains here are up to L times higher than any caching gains previously experienced in any single- or multi-antenna fully connected setting, thus offering a multiplicative mitigation to a subpacketization problem that was previously known to hard-bound caching gains to small constants. The proposed scheme manages for the first time to virtually decompose the fully connected cache-aided channel into L parallel channels. The scheme is practical; it works for all the values of K and L and all cache sizes, and its gains show in practice: e.g., for $K = 100$, when $L = 1$ the theoretical caching gain of $G = 10$, under the original coded caching algorithm, would have needed subpacketization $S_1 = \binom{K}{G} = \binom{100}{10} > 10^{13}$, while if extra transmitting antennas were added, the subpacketization was previously known to match or exceed S_1. Now for $L = 5$, our scheme offers the theoretical (unconstrained) cumulative DoF $d_L = L + G = 5 + 10 = 15$, with subpacketization $S_L = \binom{K/L}{G/L} = \binom{100/5}{10/5} = 190$. The work extends to the multi-server and cache-aided IC settings, while the scheme's performance, given subpacketization $S_L = \binom{K/L}{G/L}$, is within a factor of 2 from the optimal linear sum-DoF.**

***Index Terms*— Caching, coded caching, subpacketization, multiple antennas, transmitter cooperation, DoF.**

I. Introduction

CODED caching is a communication method invented in [1] that exploits receiver-side caches in broadcast-type communications, to achieve substantial throughput gains by delivering independent content to many users at a time. This method involves a cache placement phase and a delivery phase. During the placement phase, content from a library of files that are present at the transmitter, is properly pre-cached at the receiver caches. During the delivery phase — which starts when users simultaneously request one desired library file each — the transmitter encodes across different users' requested data content, in a way that creates multicasting opportunities even when users request different files.

Specifically the work in [1] considers the single-stream broadcast channel (BC) scenario where a single-antenna transmitter has access to a library of N files, and serves K receivers, each having a cache of size equal to the size of M files. In a normalized setting where the link has capacity 1 file per unit of time, the work in [1] showed that any set of K simultaneous requests can be served with normalized delay (worst-case completion time) which is at most $T = \frac{K(1-\gamma)}{1+K\gamma}$ where $\gamma \triangleq \frac{M}{N}$ denotes the normalized cache size. This was a major breakthrough because it shows that an ever-increasing number of users can be served in finite time that converges to $T \approx \frac{1}{\gamma} = \frac{N}{M}$ as K increases. This result implied a sum-DoF of

$$d_1(\gamma) = \frac{K(1-\gamma)}{T} = 1 + K\gamma$$

users served at a time.[1] Given that in the absence of caching, only one user could be served at a time (because $d_1(\gamma = 0) = 1$), the above implies a (theoretical) caching gain of

$$G = d_1(\gamma) - d_1(\gamma = 0) = K\gamma$$

representing the number of extra users that could be served at a time, additionally, as a consequence of introducing caching.

This massive theoretical gain came about because coded caching manages to remove the main inherent inefficiency of traditional caching methods, in which each receiver only ends up utilizing the cached fraction of just the one single file that that receiver requests, while leaving all other information in the cache unused. On the other hand, with coded caching, each receiver is now able to utilize the cached fraction of all K requested files; The cached content of its own requested file provides the traditional local caching gain, while the cached content of the $K-1$ files requested by others, are now used to cancel the interference caused by those same files.

This work was supported in part by the European Research Council through the EU Horizon 2020 Research and in part by the Innovation Program/ERC Project DUALITY under Grant 725929.

The authors are with the Communication Systems Department, EURECOM, 06410 Sophia Antipolis, France (e-mail: lampiris@eurecom.fr; elia@eurecom.fr).

[1]The concept of DoF will be elaborated on later, but roughly speaking, in wireless communications with high signal-to-noise ratio (SNR), an achieved DoF d implies the ability to deliver approximately $d\log(SNR)$ bits of content, per second per Hz. Equivalently, and again loosely speaking, it implies the ability to simultaneously serve d independent users (i.e., d users per complex dimension, i.e., per second per Hz).

This gain — which is close to the theoretic optimal [1] — was shown to persist under a variety of settings that include uneven popularity distributions [2]–[4], uneven topologies [5]–[7], a variety of channels such as erasure channels [8], MIMO broadcast channels with fading [9], a variety of networks such as heterogeneous networks [10], D2D networks [11], and in other settings as well.

A. Subpacketization Bottleneck of Coded Caching

While though in theory, this caching gain $G = K\gamma$ increases indefinitely with increasing K, in practice the gain remained — under most realistic assumptions — hard-bounded by small constants, due to the fact that the underlying coded caching algorithms required the splitting of finite-length files into an exponential number of subpackets.[2] For the algorithm in [1] in the original single-stream scenario, the near-optimal (and under some basic assumptions, optimal [13], [14]) gain of $G = K\gamma$, is achieved only if each file is segmented at least into a total of

$$S_1 = \binom{K}{K\gamma} \tag{1}$$

subpackets. As a result, having a certain maximum-allowable subpacketization of S_{max}, implies that one can only encode over a maximum of

$$\bar{K} = \arg\max_{K^o \le K} \left\{ \binom{K^o}{K^o\gamma} \le S_{max} \right\} \tag{2}$$

users, which in turn implies a substantially reduced *effective caching gain* $\bar{G}_1$ of the form

$$\bar{G}_1 = \bar{K}\gamma. \tag{3}$$

Given such a 'user-grouping' reduction (cf. [16]) of having to encode over groups of only $\bar{K}$ users at a time, and given that

$$\binom{\bar{K}}{\bar{K}\gamma} \in \left[\left(\frac{1}{\gamma}\right)^{\bar{K}\gamma}, \left(\frac{e}{\gamma}\right)^{\bar{K}\gamma} \right] = \left[\left(\frac{1}{\gamma}\right)^{\bar{G}_1}, \left(\frac{e}{\gamma}\right)^{\bar{G}_1} \right] \tag{4}$$

this effective gain $\bar{G}_1$ is bounded as

$$\frac{\log S_{max}}{1 + \log\frac{1}{\gamma}} \le \bar{G}_1 \le \frac{\log S_{max}}{\log\frac{1}{\gamma}}, \qquad \bar{G}_1 \le G \tag{5}$$

(log is the natural logarithm) which succinctly reveals that the effective caching gain $\bar{G}_1$ (and the corresponding *effective sum-DoF* $\bar{d}_1 \triangleq 1 + \bar{G}$) is placed under constant pressure from the generally small values of γ and of S_{max}. This is reflected in Figure 1 and Figure 2. Interestingly, as we know from [16], under some basic assumptions, in the context of single-antenna decentralized coded caching, this 'user-grouping' approach is close to optimal.

[2]Such high subpacketization originates from the fact that each file appears in each cache, and thus during delivery, a user must work together with all other users to get her file. This works — at least in the original algorithm by Maddah Ali and Niesen — by forming cliques of $K\gamma + 1$ users, each requesting one subfile, where each user knows all subfiles requested from the clique, except the one that she herself requests. There are a total of $\binom{K}{K\gamma}$ cliques in which a specific user will have to be part of, and all of the cliques must be used; hence the need to split each file into $\binom{K}{K\gamma}$ different subfiles.

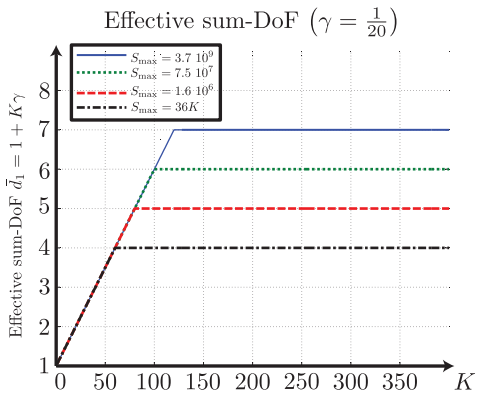


Fig. 1. Maximum effective DoF $\bar{d}_1$ achieved by the original centralized algorithm (single antenna, $\gamma = 1/20$) in the presence of different subpacketization constraints S_{max}. The gain is hard-bounded irrespective of K (x-axis).

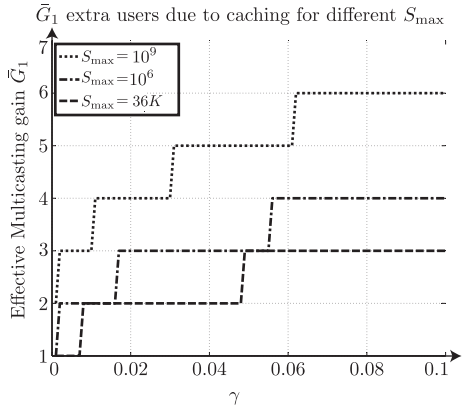


Fig. 2. Effective caching gain $\bar{G}_1 = \bar{d}_1 - 1$ (maximized over K) of the original algorithm for different S_{max}. Without subpacketization constraints, the theoretical gain is $G = K\gamma$ (unbounded as K increases).

Remark 1: It is worth noting here that, as argued in [15], in wireless cellular settings, the storage capacity at the end users is expected to induce γ that can be less than 10^{-2}, which — for a given target caching gain — implies the need to code over many users, which in turn increases subpacketization. Compounding on this problem, there is a variety of factors that restrict the maximum allowable subpacketization level S_{max}. One such parameter is the file size; for example, movies are expected to have size that is close to or less than 1 Gigabyte. Additionally, in applications like video streaming, a video file it self may be broken down into smaller independent parts (on which subpacketization will take place separately), in order to avoid the delay that comes from the asynchronous nature of decoding XORs in coded caching. Such restricted file sizes may be in the order of just a few tens of Megabytes. Another parameter that restricts S_{max} is the minimum packet size; the atomic unit of storage is not a bit but a sector (newer 'Advanced Format' hard drives use 4096-byte sectors and force zero-padding on the remaining unused sector), and similarly the atomic communication block is the packet, which must maintain a certain minimum size in order to avoid communication delay overheads.

Example 1: Looking at Figure 2, we see that if the library files (e.g. movies) are each of size 1 Gigabyte, and

under a constraint that each packet cannot be less than 1 Kilobyte (KB) long (which jointly imply a subpacketization limit of $S_{max} \approx 10^6$), then having $\gamma < 1/20$ would hard-bound the effective caching gain $\bar{G}_1$ to be less than 4 (we add one extra in comparison to the plot, in order to account for any possible improvements from memory-sharing between operating points that yield neighboring integer-valued gains). This gain reduction is because we are forced to encode over less than $\bar{K} = 80$ users, to avoid a subpacketization $\binom{80}{4} > 10^6$ that exceeds S_{max}. Having $\gamma < 1/100$ would limit this gain $\bar{G}_1$ to be less than 3 (since $\bar{K} = 300$ implies subpacketization $\binom{300}{3} > 10^6$). When $S_{max} = 10^9$, where each packet consists of a single byte (without taking into consideration the overhead from using byte-sized packets), then having $\gamma < 1/20$ would limit the effective gain to less than 6, while having $\gamma < 1/100$ would limit the number $\bar{G}_1$ of additional users that could be served due to caching, to less than 4. When $S_{max} \approx 36K$, reflecting perhaps low-latency video streaming applications, for $\gamma \leq 1/20$ then $\bar{G}_1 \approx 3$ ($\bar{d}_1 \approx 4$ users at a time), while for $\gamma \leq 1/100$ then $\bar{G}_1 \approx 2$ ($\bar{d}_1 \approx 3$).

Similar conclusions were highlighted in [16], in the context of decentralized coded caching algorithms (cf. [18]).

1) New Coded Caching Algorithms With Reduced Subpacketization: This subpacketization bottleneck sparked significant interest in designing coded caching algorithms which can provide further caching gains under reduced subpacketization costs. A first breakthrough came with the work in [19] (see also [20]) which reformulated the coded caching problem into a *placement-delivery array* (PD) combinatorial design problem, and which exploited interesting connections between coded caching and distributed storage to design an algorithm that provided a maximum theoretical caching gain of $G_{1,pd} = K\gamma - 1$ (treating a total of $K\gamma$, rather than $K\gamma + 1$, users at a time), at a reduced subpacketization of

$$S_{1,pd} = \left(\frac{1}{\gamma}\right)^{K\gamma - 1} = \left(\frac{1}{\gamma}\right)^{G_{1,pd}}$$

thus allowing — under some constraints on the operating parameters — for an effective caching gain of

$$\bar{G}_{1,pd} = \min \left\{ \frac{\log S_{max}}{\log \frac{1}{\gamma}}, K\gamma - 1 \right\}. \tag{6}$$

Similar conclusions are also drawn in [20] which made the surprising connection between coded caching and linear block codes (LC) over high-order finite fields, in order to create set partitions that identify — under some constraints on the values of γ — how the subpackets are cached and delivered, thus allowing for a tradeoff between an adjustable theoretical gain $G_{1,lc} \leq K\gamma - 1$ and the corresponding subpacketization $S \approx \left(\frac{1}{\gamma}\right)^{G_{1,lc}}$, resulting in a similar effective gain of $\bar{G}_{1,lc} \approx \frac{\log S_{max}}{\log \frac{1}{\gamma}}$ (naturally again the effective gain $\bar{G}_{1,lc}$ cannot exceed the theoretical gain $G_{1,lc}$). Another breakthrough was presented in [21] which took a hyper-graph theoretic approach to show that there do not exist caching algorithms that achieve a constant delay T (T is independent of K) with subpacketization that grows linearly[3] with K. This work also provides constructions which nicely tradeoff performance with subpacketization, which require though (Construction 6) that $K > 4/\gamma^2$ (approximately) in order[4] to have gains bigger than 5. Another milestone of a more theoretical nature was the very recent work in [22] which employs the Ruzsa-Szeméredi graphs to show for the first time that, under the assumption of (unattainably) large K, one can get a (suboptimal) gain that scales with K, with a subpacketization that scales with $K^{1+\delta}$ for some arbitrarily small positive δ.

While indeed different new algorithms provide exponential reduction in subpacketization, the corresponding improvement on the actual gain $\bar{G}$ — over the original (MN) algorithm in [1], for realistic values of γ and S_{max} — remains hard bounded and small. For example, for $\gamma \leq 1/20$ and $S_{max} \leq 10^5$, no known algorithm can improve over the MN algorithm's effective caching gain (and effective DoF) by more than two[5] (2 additional users served at a time) (see also Section V-C).

B. Coded Caching With Multiple Transmitters

At the same time, different works (cf. [23], [24] as well as [9], [25]–[33], and others) aimed at complementing such caching gains, with additional multiplexing gains that can appear when there are several transmitters. One pioneering work in this direction is found in [23] which considers a setting with $L = \lambda K$, $(\lambda \in (0, 1))$ transmitters/servers communicating (in the fully-connected BC context of a so-called 'linear network' that can translate readily to a K-user wireless MISO BC with L antennas) to K single-antenna cache-aided receivers, and which provided a scheme that achieved a theoretical sum-DoF of

$$d_L(\gamma) = L + K\gamma$$

corresponding to a MIMO multiplexing gain of L (users served, per second per hertz) and an additional theoretical caching gain of again $G = K\gamma$ (extra users served at a time, due to caching). This theoretical caching gain though is again restricted to an effective caching gain that is less than the effective gain $\bar{G}_1$ achieved in the single antenna case, because of a further increased subpacketization which now takess the form

$$S = \binom{K}{K\gamma}\binom{K - K\gamma - 1}{L - 1}. \tag{7}$$

While the subpacketization-constrained (effective) gains may have been reduced, this work in [23] nicely shows that multiplexing and caching gains can in theory be combined additively.

Soon after, the work in [24] explored the scenario where coded caching involved both transmitter-side and

[3]This assumes that γ is independent of K, that each file is divided into an identical number of subpackets, and also assumes uncoded cache placement.

[4]K must be large because the theoretical gain is reduced and is approximately $K\gamma^2/4$. K must also be (essentially) a square integer; square integers become rarer as K increases.

[5]This best-known improvement is due to [21, Construction 6] ($a = b = 2$, $\lambda = 40$) which encodes over $\bar{K} = 3160$ users to give an effective sum-DoF of 6, while the MN algorithm gives a DoF of 4 (with $\bar{K} = 60$).

receiver-side caches. In the context of a cache-aided interference scenario — where K_T transmitters with normalized cache size γ_T (each transmitter could only store a fraction γ_T of the entire N-file library), communicated to K receivers with normalized cache size γ — the work provides a scheme that employs subpacketization

$$S = \binom{K}{K\gamma}\binom{K_T}{K_T\gamma_T}\binom{K-K\gamma-1}{K_T\gamma_T-1} \tag{8}$$

to achieve a sum-DoF of $\frac{K(1-\gamma)}{T} = K_T\gamma_T + K\gamma$ which is also proven to be at most a factor of 2 from the optimal (one-shot) linear-DoF. This nicely reveals that — in the regime of unbounded subpacketization (unbounded file sizes) — the cooperative multiplexing gain $K_T\gamma_T$ which is an outcome of the caching redundancy $K_T\gamma_T$ at the transmitter-side caches, can be additively combined with the theoretical caching gain $G = K\gamma$ attributed to receiver-side caching redundancy[6] $K\gamma$. In both cases [23], [24], the addition of the extra dimensions on the transmitter side, maintains the theoretical caching gains, adds extra multiplexing gains, but maintains high subpacketization levels with generally reduced actual caching gains.

To the best of our knowledge, under the generous assumptions that $S_{max} \leq 10^5$, $\gamma \leq 1/50$ and $K \leq 10^5$, currently there exists no method *in any known single-antenna or multi-antenna* fully connected setting, with or without user grouping, that allows for the introduction of more than $\bar{G} = 5$ additional users (per second per hertz, i.e., served at a time) due to caching.[7]

C. Preview of Results and Paper Outline

Our contribution lies in the realization that having this extra dimensionality on the transmitter side, in fact reduces rather than increases subpacketization, and does so in a very accelerated manner. This property is based on the principle of the *virtual decomposition* of the cache-aided MISO BC into L parallel, single-stream coded caching channels with K/L users each. This decomposition is made possible because, as we show here for the first time, the near optimal DoF $d_L(\gamma) = L\big(1 + \frac{K}{L}\gamma\big) = L + K\gamma$ can be gained *without encoding across parallel channels*.

We will show a simple scheme for the multi-antenna/multi-node setting, that maintains the theoretical DoF

$$d_L = L + G = L + K\gamma = K_T\gamma_T + K\gamma$$

and does so with subpacketization

$$S_L = \binom{\frac{K}{L}}{\frac{K\gamma}{L}} = \binom{\frac{K}{K_T\gamma_T}}{\frac{K\gamma}{K_T\gamma_T}} \tag{9}$$

which is approximately the Lth root $S_L \simeq \sqrt[L]{S_1}$ of the original subpacketization $S_1 = \binom{K}{K\gamma}$ corresponding to $L = 1$. This will apply for all parameters $K, L, \gamma, K_T, \gamma_T$, it will imply very substantial subpacketization reductions even when L is very small, as well as will imply that the theoretical DoF $d_L = L + K\gamma$ can be achieved with subpacketization $S_L = 1/\gamma = K/L$ when L matches $K\gamma$. The above expression (9) will imply a multi-antenna effective DoF

$$\bar{d}_L = \min\{L \cdot \bar{d}_1, d_L = L + K\gamma\}$$

which is either L times the single-antenna effective DoF $\bar{d}_1$, or it is the theoretical (unconstrained) $d_L = L + K\gamma$. In the end, we now know that having multiple antennas at the transmitter, not only provides a multiplexing gain, but also a multiplicative boost of the *receiver-side* effective caching gain.

Finally, similar multiplicative boosts of the caching gain will be achieved when we apply the ideas here in conjunction with a variety of different underlying coded caching algorithms (see Section V-C) like the ones in [19] and [20].

Paper outline: Section II elaborates on the system and channel model, Section III describes the scheme, while Section IV presents the main results. The schemes and results are presented first for the integer case where $L|K$ and $L|K\gamma$ (L divides K and $K\gamma$), but we emphasize that the performance loss after removing the integer constraint, is very small (as we see in the appendix Section B). Section V addresses some related scenarios of interest, Section VI offers some conclusions, the appendix Section A shows the details of how to adapt our approach to the cache-aided interference scenario with multiple independent cache-aided transmitters, while the appendix Section B describes the slightly modified scheme for all L, K when the assumptions $L|K$ and $L|K\gamma$ are removed.

D. Notation

For clarity, we begin by recalling the common notation.

- $d_1(\gamma) = 1 + K\gamma$: Theoretical DoF ($L = 1$)
- $d_L(\gamma) = L + K\gamma$: Theoretical DoF (multiple antennas)
- $d_L(\gamma = 0) = L$: Multiplexing gain
- G: Theoretical caching gain
  - $G = d_1(\gamma) - d_1(\gamma = 0) = d_L(\gamma) - d_L(\gamma = 0) = K\gamma$
  - G additional users served at a time, due to caching[8]
- $S_1 = \binom{K}{K\gamma}$: Subpacketization needed for theoretical G ($L = 1$)
- S_{max}: Maximum allowable subpacketization
- S_L: Subpacketization needed for theoretical G (multiple antennas)
- $\bar{d}_1(\gamma)$: Effective (subpacketization-constrained) DoF ($L = 1$)
- $\bar{G}_1 = \bar{d}_1(\gamma) - 1$: Effective caching gain ($L = 1$)
- $\bar{d}_L(\gamma)$: Effective DoF (multiple antennas)
- $\bar{G}_L = \bar{d}_L(\gamma) - L$: Effective caching gain (multiple antennas)

[6] By referring to transmitter-side redundancy $K_T\gamma_T$ and receiver-side redundancy $K\gamma$, we simply refer to the fact that each subfile resides in the caches of $K_T\gamma_T$ transmitters and in the caches of $K\gamma$ receivers.

[7] This corresponds to [21, Construction 6] ($a = b = 2, \lambda = 100$), and it requires encoding over approximately $\bar{K} = 20000$ users.

[8] The choice here to measure the caching gain as the DoF difference $G = d_1(\gamma) - d_1(\gamma = 0) = d_L(\gamma) - d_L(\gamma = 0) = K\gamma$ rather than the DoF ratio, comes from the fact that in theory, the two gains (multiplexing and caching gains) appear to aggregate in an additive manner (this is discussed also in [24]). This choice of G seems better suited for multi-antenna settings because a) it cleanly removes the multiplexing gain thus better isolating the true effect of caching, b) it reflects a caching gain that does not inevitably vanish with increasing L (as would have happened had we used the DoF ratio), and c) it reflects a caching gain that scales with the cumulative cache size at the receiver side (i.e., scales with K).

In the above, $\bar{d}_L(\gamma=0) = d_L(\gamma=0) = L$ is the multiplexing gain, and $\bar{G}_L$ is the effective caching gain describing the actual number of additional users that can be served at a time as a result of introducing caching, under a subpacketization constraint. Finally the effective DoF $\bar{d}_L(\gamma) = L + \bar{G}_L$ describes the actual (total) number of users that can be served at a time, under a subpacketization constraint.

Furthermore we employ the following notation. $\mathbb{Z}$ will represent the integers, $\mathbb{Z}^+$ the positive integers, $\mathbb{R}$ the real numbers, and $\binom{n}{k}$ the n-choose-k (binomial) operator. We will use $[K] \triangleq \{1, 2, \cdots, K\}$. If $\mathcal{A}$ is a set, then $|\mathcal{A}|$ will denote its cardinality. For sets $\mathcal{A}$ and $\mathcal{B}$, then $\mathcal{A}\backslash\mathcal{B}$ denotes the difference set. The expressions $\alpha|\beta$ (resp. $\alpha \nmid \beta$) denote that integer α divides (resp. does not divide) integer β. Complex vectors will be denoted by lower-case bold font. We will use $||\boldsymbol{x}||^2$ to denote the magnitude of a vector $\boldsymbol{x}$ of complex numbers. Furthermore if $\mathcal{A} \subset [K]$ is a subset of users, then we will use $\mathbf{H}^{\mathcal{A}}$ to denote the overall channel from the L-antenna transmitter to the users in $\mathcal{A}$. Logarithms are of base e. In a small abuse of notation, we will sometimes denote data sets the same way we denote the complex numbers (or vectors) that carry that same data.

II. System and Channel Model

We initially consider the K-user multiple-input single-output (MISO) broadcast channel,[9] where an L-antenna transmitter communicates to K single-antenna receiving users. The transmitter has access to a library of N distinct files $W_1, W_2, \ldots, W_N$, each of size $|W_n| = f$ bits. Each user $k \in \{1, 2, \ldots, K\}$ has a cache Z_k, of size $|Z_k| = Mf$ bits, where naturally $M \leq N$. Communication consists of the aforementioned *content placement phase* and the *delivery phase*. During the placement phase the caches $Z_1, Z_2, \ldots, Z_K$ are pre-filled with content from the N files $\{W_n\}_{n=1}^N$.

The delivery phase commences when each user k requests from the transmitter, any *one* file $W_{R_k} \in \{W_n\}_{n=1}^N$, out of the N library files. Upon notification of the users' requests, the transmitter aims to deliver the (remaining of the) requested files, each to their intended receiver, and the challenge is to do so over a limited (delivery phase) duration T. During this delivery phase, for each transmission, the received signals at each user k, will be modeled as

$$y_k = \boldsymbol{h}_k^T \boldsymbol{x} + w_k, \quad k = 1, \ldots, K \tag{10}$$

where $\boldsymbol{x} \in \mathbb{C}^{L\times 1}$ denotes the transmitted vector satisfying a power constraint $\mathbb{E}(||\boldsymbol{x}||^2) \leq P$, where $\boldsymbol{h}_k \in \mathbb{C}^{L\times 1}$ denotes the channel of user k in the form of the random vector of fading coefficients that can change in time and space, and where w_k represents unit-power AWGN noise at receiver k. We will assume that P is high (high SNR), we will assume perfect channel state information throughout the (active) nodes as in [23] and [24], and we will assume that the fading process is statistically symmetric across users, such that each link has capacity of the form $\log(SNR) + o(\log(SNR))$.

[9] We note that while the representation here is of a wireless model, the result applies directly to the multi-server *wireline* setting of [23] with a fully connected linear network. We will also show at the end of this paper how the work here applies to the cache-aided interference scenario of [24]. Finally we note that in the DoF regime of interest, the single-antenna wireless setting ($L = 1$) matches identically (in terms of the characteristics and performance) the original single-stream shared-link setting in [1].

A. Performance Measures

As in [1], T is the number of time slots, per file served per user, needed to complete the delivery process, *for any request*. The wireless link capabilities, and time scale, are normalized such that one time slot corresponds to the optimal amount of time it would take to communicate a single file to a single receiver, had there been no caching and no interference

As in the single-stream case in [1], T is simply the minimum delay that allows, in the information theoretic sense (thus, under sufficiently long file sizes f), that each receiver k decodes (with probability 1) its message W_{R_k}. T reflects the maximum such minimum delay, maximized over all possible requests $\{W_{R_k}\}_{k=1}^K$. The high-SNR normalized delay T (cf. [32]; see also [25], [26]) used here, accounts for the file sizes and the high-SNR link capacity scaling $\log(\text{SNR})$, and is thus identical to the rate measure used in [1] for the single-stream error-free setting. Consequently in the high SNR setting of interest, an inversion leads to the equivalent measure of the cache-aided sum DoF $d_L(\gamma) = \frac{K(1-\gamma)}{T}$, as this is defined in [34] in the context of transmitter-side caching, and in [32] in the context of receiver-side caching (see also [25], [26]). The sum-DoF is simply the delivery rate (in units of 'file', after normalization by $\log(\text{SNR})$), it can be seen as the total amount of users served at a time, and it is the sum of multiplexing and theoretical caching gains.

As in [1], we will first consider the case where $\gamma = \frac{M}{N} = \{1, 2, \cdots, K\}\frac{1}{K}$, while for non integer $K\gamma$, we will simply consider the result corresponding to $\lfloor K\gamma \rfloor$. Furthermore we will ignore the trivial case of $L \geq K(1-\gamma)$ which can be directly handled — as shown in [23] — to achieve the interference-free optimal $T = 1 - \gamma$ corresponding to a sum-DoF $d_L(\gamma) = K$.

III. Description of the Scheme

We will present the scheme for all K, γ, L, first focusing here on the case where $L|K\gamma$ and $L|K$. The general scheme simply uses memory sharing to remove this integer constraint. This generalization is described in the appendix, where we also prove that the cost of removing the integer constraint is bounded above by a factor of only 2. Consequently the scheme will remain near-optimal, for all values of K, γ, L.

A. Grouping

We first split the K users $k = 1, 2, \ldots, K$ into $K' \triangleq \frac{K}{L}$ disjoint groups

$$\mathcal{G}_g = \{\ell K' + g,\ \ell = 0, 1, \ldots, L-1\},\ \text{for}\ g = 1, 2, \ldots, K'$$

of $|\mathcal{G}_g| = L$ users per group. Our aim is to apply the algorithm of [1] to serve $K'\gamma + 1$ groups at a time, essentially treating each group as a single user. Toward this, let

$$\mathcal{T} = \{\tau \subset [K'] \ :\ |\tau| = K'\gamma\}$$

be the set of

$$|\mathcal{T}| = \binom{K'}{K'\gamma} \tag{11}$$

subsets in $[K']$, each of size $|\tau| = K'\gamma$, and let

$$\mathcal{X} = \{\chi \subseteq [K'] \ : \ |\chi| = K'\gamma + 1\}$$

be the set of $|\mathcal{X}| = \binom{K'}{K'\gamma+1}$ subsets of size $|\chi| = K'\gamma + 1$.

B. Subpacketization and Caching

We first split each file W_n into $|\mathcal{T}|$ subfiles $\{W_n^\tau\}_{\tau\in\mathcal{T}}$, and then we assign each user $k \in \mathcal{G}_g$ the cache

$$Z_k = Z_{\mathcal{G}_g} = \{W_n^\tau \ : \ \forall \tau \ni g\}_{n=1}^N \tag{12}$$

so that all users of the same group have an identical cache.[10]

C. Transmission

After notification of requests — where each receiver k requires file W_{R_k}, $R_k \in [N]$ — the delivery consists of a sequential transmission $\{\boldsymbol{x}_\chi\}_{\chi\in\mathcal{X}}$ where each transmission takes the form

$$\boldsymbol{x}_\chi = \sum_{g\in\chi} \sum_{k\in\mathcal{G}_g} W_{R_k}^{\chi\backslash g} \boldsymbol{v}^{\mathcal{G}_g\backslash k} \tag{13}$$

and where $\boldsymbol{v}^{\mathcal{G}_g\backslash k}$ is an $L\times 1$ precoding vector that is designed to belong in the null space of the channel $\mathbf{H}^{\mathcal{G}_g\backslash k}$ between the L-antenna transmitter and the $L-1$ receivers in group $\mathcal{G}_g$ excluding receiver $k \in \mathcal{G}_g$.

D. Decoding—'Caching-Out' Out-of-Group Messages

The corresponding received signal at user $k \in \mathcal{G}_g$ is then

$$\boldsymbol{y}_{k,\chi} = \boldsymbol{h}_k^T \boldsymbol{x}_\chi + \boldsymbol{w}_{k,\chi} \tag{14}$$

and each such user $k \in \mathcal{G}_g$ can employ its cache to immediately remove all the files that are jointly undesired by its own group $\mathcal{G}_g$, i.e., receiver $k \in \mathcal{G}_g$ can remove

$$\sum_{g'\in\chi\backslash g} \sum_{j\in\mathcal{G}_{g'}} W_{R_j}^{\chi\backslash g'} \boldsymbol{v}^{\mathcal{G}_{g'}\backslash j}$$

because $g' \neq g \in \chi$, i.e., because the cache of receiver k includes all files $W_{R_j}^{\chi\backslash g'}$ in the above summation. This allows receiver k to remove all files that are not of interest to its group $\mathcal{G}_g$, and thus to get

$$\boldsymbol{y}'_{k,\chi} = \boldsymbol{h}_k^T \sum_{j\in\mathcal{G}_g} W_{R_j}^{\chi\backslash g} \boldsymbol{v}^{\mathcal{G}_g\backslash j} \ + \boldsymbol{w}_{k,\chi}. \tag{15}$$

[10] A quick verification shows that

$$|Z_{\mathcal{G}_g}| = N\frac{|\{\tau\in\mathcal{T} : g\in\tau\}|}{|\mathcal{T}|} = N\frac{\binom{K'-1}{K'\gamma-1}}{\binom{K'}{K'\gamma}} = N\gamma = M.$$

E. Nulling-Out Intra-Group Messages—Completion of Decoding

The interference for receiver k now could only come from the files of the $L-1$ other users of its own group $\mathcal{G}_g$. This interference is averted directly by the ZF precoders (or any other DoF optimal precoder), and receiver k can get the desired $W_{R_k}^{\chi\backslash g}$.

This is done instantaneously for all users $k \in \mathcal{G}_g$, and for all $g \in \chi$. Hence the scheme delivers to $K'\gamma + 1$ groups at a time, thus to

$$d_L(\gamma) = L(K'\gamma + 1) = K\gamma + L \tag{16}$$

users at a time. Then we do the same for another $\chi \in \mathcal{X}$. Along the different $\chi \in \mathcal{X}$, no subfile is repeated, and we can now conclude that the DoF is $K\gamma + L$, which as we saw (cf. (11)) is achieved here with subpacketization $S_L = \binom{K'}{K'\gamma}$.

F. Example of Scheme–Alternate Representation

Let $K = 50$, $L = 5$ and $\gamma = M/N = 3/10$. We will achieve the sum-DoF of $d_\Sigma = L + G = L + K\gamma = 5 + 15 = 20$, with a subpacketization of 120.

First split the $K = 50$ users into $K' = 10$ groups of $L = 5$:

$$\mathcal{G}_1 = \{1, 11, 21, 31, 41\}, \ldots, \mathcal{G}_{10} = \{10, 20, 30, 40, 50\}.$$

Since $K'\gamma = 3$, we split each file W_n into $|\mathcal{T}| = \binom{K'}{K'\gamma} = 120$ parts

$$W_n = \{W_n^{(1,2,3)}, W_n^{(1,2,4)}, \ldots, W_n^{(1,3,4)}, \ldots, W_n^{(8,9,10)}\}$$

and then fill the caches

$$\begin{aligned} Z_{\mathcal{G}_1} &= \{W_n^{(1,2,3)}, W_n^{(1,2,4)}, \ldots W_n^{(1,3,4)}, \ldots W_n^{(1,9,10)}\}_{n=1}^N \\ &\vdots \\ Z_{\mathcal{G}_{10}} &= \{W_n^{(1,2,10)}, W_n^{(1,3,10)}, \ldots, W_n^{(2,3,10)}, \ldots W_n^{(8,9,10)}\}_{n=1}^N \end{aligned}$$

as described. We will serve $K'\gamma + 1 = 4$ groups at a time. We treat the group clique $\chi = (1, 2, 3, 4)$ first. Let

$$\boldsymbol{w}_1^{(2,3,4)} = [W_{R_1}^{(2,3,4)}, W_{R_{11}}^{(2,3,4)}, W_{R_{21}}^{(2,3,4)}, W_{R_{31}}^{(2,3,4)}, W_{R_{41}}^{(2,3,4)}]^T$$

be the $L = 5$ subfiles currently meant for the 5 users in the first group. Similarly let $\boldsymbol{w}_2^{(1,3,4)}, \boldsymbol{w}_3^{(1,2,4)}, \boldsymbol{w}_4^{(1,2,3)}$ be the L-length vectors of subfiles for the second, third and fourth groups respectively. Then simply transmit

$$\begin{aligned} \boldsymbol{x}_{(1,2,3,4)} = (\mathbf{H}^{\mathcal{G}_1})^{-1}\boldsymbol{w}_1^{(2,3,4)} + (\mathbf{H}^{\mathcal{G}_2})^{-1}\boldsymbol{w}_2^{(1,3,4)} \\ +(\mathbf{H}^{\mathcal{G}_3})^{-1}\boldsymbol{w}_3^{(1,2,4)} + (\mathbf{H}^{\mathcal{G}_4})^{-1}\boldsymbol{w}_4^{(1,2,3)} \end{aligned} \tag{17}$$

where $(\mathbf{H}^{\mathcal{G}_g})^{-1}$ denotes the (normalized) inverse of the $L\times L$ channel to group $\mathcal{G}_g$.

Receiver 1 can immediately remove — using its cache — the last three summands in (17), and ZF can remove the unwanted $L - 1 = 4$ elements from $\boldsymbol{w}_1^{(2,3,4)}$. The achieved caching gain is $G = 15$, the sum-DoF is $d_L(\gamma) = 20$ (users at a time), and the subpacketization is $S_L = 120$.

IV. Main Results

We present the main results, first for the integer case where $L|K$ and $L|K\gamma$. The interpolation to all cases K, L is easily handled using memory sharing, and as we note later on, does not result in substantial performance degradation. The details for this are handled in the appendix. We also try to highlight the practical relevance of some of these results, with examples.

We proceed with the main result.

Theorem 1: In the cache-aided MISO BC with L transmitting antennas and K receiving users, the delay of $T = \frac{K(1-\gamma)}{L+K\gamma}$ and the corresponding sum-DoF $d_L(\gamma) = L + K\gamma$, can be achieved with subpacketization

$$S_L = \binom{K/L}{K\gamma/L}.$$

Proof: The proof of this is direct from the description of the scheme. Specifically (11) tells us that the subpacketization is $\binom{K'}{K'\gamma}$ where $K' = K/L$, while (16) tells us that the DoF is $d_L(\gamma) = L(K'\gamma + 1) = K\gamma + L$. ■

A. Effective Gains and Multiplicative Boost of Effective DoF

We recall that in the absence of subpacketization constraints, adding extra transmitting antennas, takes us from a theoretical sum-DoF $d_1 = 1 + K\gamma$ to $d_L = L + K\gamma$ (cf. [23]), leaving the theoretical caching gain unaffected, and adding $d_L(\gamma) - d_1(\gamma) = L - 1$ DoF. What our result directly suggests is that, when subpacketization is taken into consideration, adding extra transmitting antennas (or later, adding extra transmitter-side caching) can have a much more powerful, multiplicative impact on the effective gains. To see this, simply recall that when $L = 1$, we can only encode over $\bar{K}_1 \triangleq \arg\max_{K^o \le K} \left\{ \binom{K^o}{K^o\gamma} \le S_{max} \right\}$ users, while in the L antenna case, this increases by up to L times, to

$$\bar{K}_L \triangleq \arg\max_{K^o \le K} \left\{ \binom{\frac{K^o}{L}}{\frac{K^o\gamma}{L}} \le S_{max} \right\} = \min\{L \cdot \bar{K}_1, K\}. \tag{18}$$

This is captured in the following corollary which tells us that the L-fold multiplicative DoF boost stays into effect as long as $\binom{\frac{K}{L}}{\frac{K\gamma}{L}} \ge S_{max}$, i.e., as long as subpacketization remains an issue.

Corollary 1: Under a maximum allowable subpacketization S_{max}, the multi-antenna effective caching gain and DoF take the form

$$\bar{G}_L = \min\{L \cdot \bar{G}_1, G = K\gamma\} \tag{19}$$

$$\bar{d}_L = \min\{L \cdot \bar{d}_1, d_L = L + K\gamma\} \tag{20}$$

which means that with extra antennas, the (single-antenna) effective DoF $\bar{d}_1$ is either increased by a multiplicative factor of L, or it reaches the theoretical (unconstrained) DoF $d_L = L + K\gamma$.

Example 2: Consider a single stream ($L = 1$) coded caching system which offers — in the absence of any file-size constraints — a theoretical sum-DoF of $d_1 = 1 + 28$. If we add one antenna, the DoF becomes $2 + 28$, one more antenna gives a DoF of $3 + 28$, and one more antenna ($L = 4$) gives a DoF of $d_L = 4 + 28 = 32$. Now imagine that due to file-size constraints, the same single stream coded caching system (same resources, again with $L = 1$) gives an actual sum-DoF of $\bar{d}_1 = 1 + 7 = 8$. Also imagine that up to $L = 4$, subpacketization remains an issue, i.e., that $\binom{\frac{K}{4}}{\frac{K\gamma}{4}} \ge S_{max}$. Then we see that, with the new scheme, adding one antenna will double the effective DoF to $2 + 14 = 16$, adding one more antenna will take us to an effective DoF of 24, and adding one more antenna ($L = 4$) will yield DoF of 32.

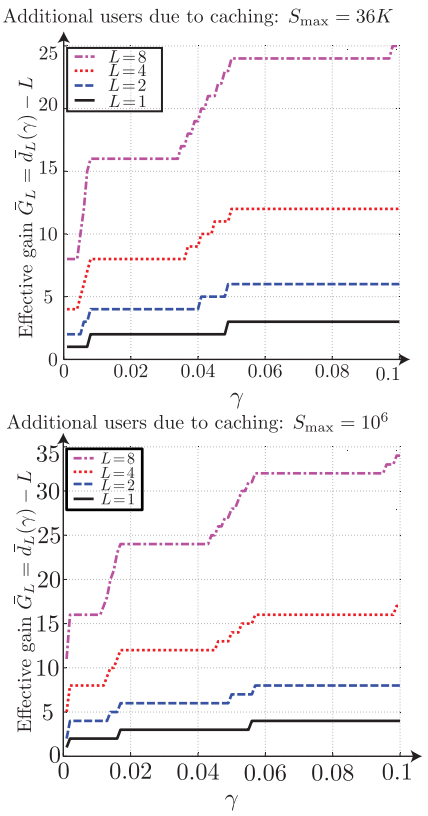


Fig. 3. Maximum achievable effective caching gain $\bar{G}_L = d_L(\gamma) - L$ (maximized over all possible K), achieved by the new scheme for different L, under subpacketization constraint $S_{max} = 3.6 \cdot 10^4$ (above) and $S_{max} = 10^6$ (below).

The following corollary bounds the derived effective caching gain $\bar{G}_L$.

Corollary 2: Given a maximum allowable subpacketization S_{max}, the effective caching gain of the presented scheme is bounded as

$$\bar{G}_L \ge \min \left\{ L \cdot \frac{\log S_{max}}{1 + \log(\frac{1}{\gamma})}, K\gamma \right\}. \tag{21}$$

Proof: This follows directly from Sterling's approximation which bounds subpacketization as $S_L = \binom{K'}{K'\gamma} \le \left(\frac{e}{\gamma}\right)^{K'\gamma} = \left(\frac{e}{\gamma}\right)^{\frac{G}{L}}$ which directly implies that $\bar{G}_L \ge L \cdot \frac{\log S_{max}}{1+\log(\frac{1}{\gamma})}$ (up to the theoretical gain $G = K\gamma$). ■

B. Subpacketization Scaling

The following corollary highlights that, in an L-antenna MISO BC system, the subpacketization cost is not determined

by K or $L = \lambda K$, nor by the number of extra users G we wish to add due to caching, but rather by the ratio $x = \frac{d_L(\gamma)}{d_L(\gamma=0)}$ between the DoF and the multiplexing gain.

Corollary 3: In our L-antenna MISO BC setting, a subpacketization of

$$S = \binom{1/\lambda}{x-1} = \binom{\frac{1}{\lambda}}{\frac{\gamma}{\lambda}}$$

can yield a DoF that is x times the multiplexing gain.

Proof: The DoF increase from $d_L(\gamma = 0) = L$ to $d_L(\gamma) = L + K\gamma = x \cdot L, \quad x \in \mathbb{Z}^+$, implies that $K\gamma = L(x-1)$ and that $\gamma = \lambda(x-1)$, which means that the corresponding subpacketization $S_L = \binom{K/L}{K\gamma/L}$ now takes the form $S = \binom{\frac{1}{\lambda}}{\frac{\gamma}{\lambda}} = \binom{1/\lambda}{x-1}$. ∎

Directly from the previous corollary, we also have the following.

Corollary 4: In asymptotic terms, as long as L scales with the caching gain $K\gamma$, the entire sum-DoF $L+K\gamma$ is achievable with constant subpacketization. In the extreme case where $L = K\gamma$, the same DoF can be achieved with subpacketization

$$S_L = \frac{1}{\gamma} = \frac{K}{L}.$$

Proof: As we have seen in the previous corollary, for $L = \frac{1}{q}K\gamma$ for some fixed $q \in \mathbb{Z}^+$, then the subpacketization is $S = \binom{1/\lambda}{q}$ and it is independent of K, L. ∎

Example 3: In a BC with $\gamma = 1/100$ and $L = 1$, allowing for caching gains of $G = K\gamma = 10$ (additional users due to caching), would require $S_1 = \binom{1000}{10} > 10^{23}$ so in practice coded caching could not offer such gains. In the $L = 10$ antenna case, this caching gain comes with subpacketization of only $S_L = K/L = 100$.

C. Near-Optimality of Schemes

The schemes that we have employed here (as described in Section III and in the Appendix) have the 'one-shot, linear' property which means that each data element is manipulated linearly, and only once (a data bit is not transmitted more than once). This lends the results, amenable to the analysis in [24] whose outer bound then allows us to directly conclude that the schemes are near optimal. This is described below in the form of a corollary.

Corollary 5: The described subpacketization $S_L = \binom{\frac{K}{L}}{\frac{K\gamma}{L}}$ and $S_{K_T\gamma_T} = \binom{\frac{K}{K_T\gamma_T}}{\frac{K\gamma}{K_T\gamma_T}}$ guarantees sum-DoF performance that is at most a factor of 2 from the theoretical optimal linear-DoF.

Proof: As stated, the proof is direct from the bound in [24], from the performance achieved by the schemes here, and from the fact that the schemes have the 'one-shot linear' property. ∎

Remark 2 (Removing the Integer Constraint): We also note here that, to remove the integer constraints $L|K$ and $L|K\gamma$, we can readily use memory sharing as in [1]. This is shown in the appendix, where we see that after removing the integer constraints, the results remain approximately the same except for a marginal increase in subpacketization to at most $S_L \leq K \cdot \max\left\{\binom{\lceil K/L\rceil}{\lceil K\gamma/L+1\rceil}, \binom{\lceil K/L\rceil}{\lfloor K\gamma/L+1\rfloor}\right\}$, and a relatively small reduction in the achieved DoF ($d_L(\gamma) = L + K\gamma$) by a multiplicative factor (gap) that is bounded above by 2 when $L > K\gamma$, and by $\frac{3}{2}$ when $L < K\gamma$ in which case the gap vanishes (converges to 1) as $K\gamma$ increases.

V. Related Scenarios

A. Transmitter Cooperation for Boosting Coded Caching

Until now we have explored the effect of having L antennas at the transmitter. An identical effect will appear if instead of a single L-antenna transmitter, we consider K_T independent single-antenna transmitters, each equipped with a cache of normalized cache size of $\gamma_T \geq \frac{1}{K_T}$ (as before, there are K fully-interfering single-antenna receivers with normalized cache size γ). This setting corresponds to the $K_T \times K$ cache-aided interference scenario of [35], for which — as discussed in Section I-B — the (unconstrained) achieved 'one-shot linear' sum-DoF takes the form $K_T\gamma_T + K\gamma$.

Corollary 6: In the $K_T \times K$ cache-aided interference scenario with normalized cache sizes γ_T, γ, the sum-DoF of $K_T\gamma_T + K\gamma$, can be achieved with subpacketization of

$$S_{K_T\gamma_T} = \binom{\frac{K}{K_T\gamma_T}}{\frac{K\gamma}{K_T\gamma_T}}.$$

Proof: The constructive proof of the above is described in the Appendix. ∎

1) Effects of Cache-Aided Transmitter-Cooperation on Coded Caching: Given Corollary 6, it is not difficult to conclude that all the previous corollaries apply directly to the $K_T \times K$ cache-aided interference scenario, after substituting L with $K_T\gamma_T$. In particular, drawing from the previous corollaries, we can summarize the following results that apply to cache-aided transmitter cooperation.

- As the transmitter-side cache redundancy $K_T\gamma_T$ increases, the effective DoF will either be increased by a multiplicative factor of $K_T\gamma_T$, or it will reach the theoretical (unconstrained) DoF $K_T\gamma_T + K\gamma$ (cf. Corollary 1).
- Increasing the transmitter-side cache redundancy $K_T\gamma_T$, allows for an exponentially reduced minimum applicable $\gamma \geq (S_{\max})^{-1/G \frac{K_T\gamma_T}{}}$ that can offer a (receiver-side) caching gain of $G = K\gamma$ (cf. Section IV-A).
- Subpacketization $S = \binom{\frac{K}{K_T\gamma_T}}{x-1}$ can yield a sum DoF that is x times the cooperative multiplexing gain $K_T\gamma_T$.
- In asymptotic terms, as long as the transmitter-side cache redundancy $K_T\gamma_T$ scales with the receiver cache redundancy $K\gamma$, the entire sum-DoF $K_T\gamma_T + K\gamma$ is achievable with constant subpacketization (cf. Corollary 4).
- When the transmitter-side and receiver-side cache redundancies match (i.e., when $K_T\gamma_T = K\gamma$), the DoF $K_T\gamma_T + K\gamma$ can be achieved with subpacketization $S_{K_T\gamma_T} = \frac{K}{K_T\gamma_T}$ (cf. Corollary 4).

2) Base-Station Cooperation for Boosting Coded Caching: The following corollary also holds.

Corollary 7: In the $K_T \times K$ cache-aided interference scenario with $\gamma_T \geq \frac{1}{K_T}$, if each transmitter has L_T transmitting

antennas, the sum-DoF of $K_T L_T \gamma_T + K\gamma$, can be achieved with subpacketization of

$$S_{K_T L_T \gamma_T} = \binom{\frac{K}{K_T L_T \gamma_T}}{\frac{K\gamma}{K_T L_T \gamma_T}}.$$

Thus when $K_T L_T \gamma_T = K\gamma$ this sum-DoF can be achieved with subpacketization

$$S = \frac{K}{K_T L_T \gamma_T}.$$

The proof of the above is described briefly in the Appendix.

Example 4 (Base-Station Cooperation): Let us consider a scenario where in a dense urban setting, a single base-station ($K_T = 1$) serves $K = 10000$ cell-phone users, who are each willing to dedicate 20 Gigabytes of their phone's memory for caching parts from a Netflix library of $N = 10000$ low-definition movies. Each movie is 1 Gigabyte in size, and the base-station can store 10 Terabytes. This corresponds to having $M = 20$, $\gamma = M/N = 1/500$, and $\gamma_T = 1$. If $L_T = 1$ (single transmitting antenna), a caching gain of $G = 20$ would have required (given the MN algorithm) subpacketization of $S_1 = \binom{K}{K\gamma} = \binom{10000}{20} > 10^{61}$.

If instead we had two base-stations ($K_T = 2$) with $L_T = 5$ transmitting antennas each, this gain would require subpacketization $S_L = \frac{\frac{K}{K_T L_T}}{\frac{K\gamma}{K_T L_T}} = \frac{10000/10}{20/10} = \frac{1000}{2} \approx 5 \cdot 10^5$ (hence here, the introduction of caching would triple the total number of users served at a time), while with $K_T = 4$ such cooperating base-stations, this gain could be achieved with subpacketization of $\frac{10000/20}{20/20} = 500$.

B. Making Small Caches Relevant

Another benefit of the reduced subpacketization here, is the resulting exponential increase in the range of cache sizes that can achieve a given target gain. While in theory, a small γ does not necessarily preclude higher caching gains because we could conceivably compensate by increasing the number of users we encode over, such an increase would increase subpacketization thus again precluding high gains (subpacketization limits would not allow for such an increase in the number of users we encode over). Specifically we recall (cf. (4)) that when $L = 1$ then the subpacketization is bounded as $S_1 \geq \frac{1}{\gamma}^G$, which means that to meet a subpacketization constraint of $S_{\max}$ and a target caching gain of G, we need

$$\gamma \geq (S_{\max})^{-1/G}. \tag{22}$$

On the other hand, the reduced subpacketization $S_L \geq \frac{1}{\gamma}^{\frac{1}{L}G}$ in the L antenna case (cf. (4), after substituting K by K/L), can allow for the same caching gain G (given sufficiently many users to encode over) with only

$$\gamma \geq (S_{\max})^{-1/G\ L}. \tag{23}$$

This exponential reduction in the minimum applicable γ, matches well the spirit of exploiting caches at the very periphery of the network, where we are expected to find relatively small but abundantly many caches.

C. Decomposition of Different Coded Caching Algorithms in the L Antenna Setting

The aforementioned subpacketization can be further reduced when considering alternate coded caching algorithms. We recall that the scheme that we have presented, involved 'elevating' the original MN algorithm [1], from the single-stream scenario ($L = 1$) with $K' = \frac{K}{L}$ users, to the L-antenna case with K' groups of L-users per group. This same idea can apply *directly* to other centralized coded caching algorithms like those in [19], [20], and [22], in which case the steps are almost identical:

- Choose the new coded caching algorithm for the single-stream K'-user scenario.
- Split the K users into K' groups of L users each, and employ the new algorithm to fill the caches as in the K'-user single-stream case, as if each group is a user, such that same-group users have caches that are identical.
- Using the coded caching algorithm for the single-stream K'-user scenario, generate the sequence of XORs. Each XOR consists of $d'_1(\gamma)$ summands, where $d'{}_1(\gamma)$ is the theoretical sum-DoF provided by the coded caching algorithm in the K'-user single-antenna (single stream) BC.
- Each element (summand) of the XOR, corresponds to a group of users, and each such XOR summand is replaced by a (precoded) L-length vector that carries the L-requests of the associated group. Add these d'_1 vectors together, to form a composite transmitted vector that corresponds to the XOR.
- Each composite vector treats a total of d'_1 groups at a time, i.e., treats $L \cdot d'{}_1(\gamma)$ users at a time.
- Then continue with the rest of the XORs.

Hence we recall that when[11] elevating the MN algorithm — which, for the single-stream K'-user case, treats $d'{}_1(\gamma) = K'\gamma + 1$ users at a time — we treated $d'_1 = K'\gamma + 1$ groups at a time, thus treating a total of $d_L(\gamma) = L \cdot d'{}_1(\gamma) = L + K\gamma$ users at a time. On the other hand, when elevating for example the algorithms in [19] and [20], we would naturally have to change the cache placement and the sequence of XORs and we would have to account for the fact that — for the single stream K'-user case — the algorithm treats $d'_{1,pd} = K'\gamma$ users at a time (instead of $K'\gamma + 1$), and thus for $L \geq 1$, we would treat $d'_{1,pd} = \frac{K}{L}\gamma$ groups at a time ($L \leq K\gamma$), thus treating a total of $d_{L,pd}(\gamma) = L \cdot d'_{1,pd} = K\gamma$ users at a time[12] (not $K\gamma + L$).

The following corollary describes the effective caching gain provided by the scheme that elevates to the L antenna case, the placement-delivery array (PD) and linear code (LC) algorithms [19], [20]. These algorithms impose some constraints on γ.

Corollary 8: Given a maximum allowable subpacketization S_{max}, the effective caching gain of the here-elevated PD and LC algorithms, takes the form

$$\bar{G}_{L,pd} = \bar{G}_{L,lc} = \min \left\{ L \cdot \frac{\log S_{max}}{\log(\frac{1}{\gamma})} K\gamma - L \right\}. \tag{24}$$

[11] We will henceforth use the term 'elevate' to correspond to when we apply a single-stream coded caching algorithm to the multi-antenna case, via the above sequence of steps.

[12] This makes the caching gain of the multi-antenna case equal to $G_{L,pd} = K\gamma - L$

Proof: With a theoretical gain $G_{L,pd} = d_{L,pd}(\gamma) - d_{L,pd}(\gamma = 0) = K\gamma - L$, the underlying subpacketization $S_{L,pd} = \frac{1}{\gamma}^{K'\gamma-1}$ can be written as $S_{L,pd} = \frac{1}{\gamma}^{\frac{G_{L,pd}}{L}}$, and thus the effective gain is $\bar{G}_{L,pd} = L \cdot \frac{\log S_{max}}{\log(\frac{1}{\gamma})}$, which is bounded by the theoretical caching gain $K\gamma - L$ offered by the scheme in the absence of subpacketization constraints. ∎

1) L-Fold Increase in Impact of Alternate Coded Caching Algorithms: The fact that the underlying coded caching algorithm is used in our design at the level of groups of users, implies that any difference in the effective caching gain between two underlying algorithms in the single-stream case, will be magnified — once each algorithm is elevated to the L-antenna case as was shown here — by a factor of up to L. For example, if we were to compare the elevated MN scheme to, say, the aforementioned elevated PD and LC schemes, we would see (cf. Corollary 2 and Corollary 8) that

$$\bar{G}_{L,pd} = \min \quad L \cdot \frac{\log S_{max}}{\log(\frac{1}{\gamma})}, K\gamma - L \Bigg\}$$

$$\bar{G}_{L} \geq \min \quad L \cdot \frac{\log S_{max}}{1+\log(\frac{1}{\gamma})}, \; K\gamma \Bigg\}$$

which would tell us that (when $K\gamma$ is an integer) the improvement in effective gains is bounded as

$$\bar{G}_{L,pd} - \bar{G}_{L} \leq L \cdot \frac{\log S_{max}}{(\log(\frac{1}{\gamma}))(1+\log(\frac{1}{\gamma}))}.$$

When $L = 1$, this improvement — under realistic assumptions on γ and S_{max} — can be small, but when the algorithm is elevated to the multi-antenna setting, this improvement increases as a multiple of L.

Remark 3: This implies that the decomposition method proposed here, rather than bypassing the need for novel single-stream coded caching algorithms of reduced subpacketization, it in fact accentuates the importance of searching for such algorithms.

VI. Conclusions

In the context of coded caching with multiple transmitting antennas (or with multiple transmitters or servers), we have presented a simple scheme which exploits transmitter-side dimensionality to provide very substantial reductions in the required subpacketization, and a multiplicative boost in the actual DoF performance of the system. This multiplicative DoF increase, suggests that in some cases the main impact of multiple transmitting antennas in cache-aided systems, is not the multiplexing gain, but rather the boost on the effect of receiver-side coded caching.

A. Intuition on Design

The design was based on the simple observation that multi-node (transmitter-side) precoding, reduces the need for content overlap. The subpacketization reduction from $\binom{K}{K\gamma}$ to $\binom{K/L}{K\gamma/L}$ was here related to the fact that the receivers of each group have identical caches. Subpacketization can generally increase because there needs to be a large set of pairings between the different caches. Here the number of different distinct caches is reduced, and thus the number of such pairings remains smaller.

B. Parallel Decomposition of Coded Caching

A useful contribution of this work is the *virtual decomposition* of the cache-aided MISO BC into L parallel, single-stream coded caching channels with K/L users each. This decomposition is made possible because, as we show here for the first time, the near optimal DoF $d_L(\gamma) = L\big(1 + \frac{K}{L}\gamma\big) = L + K\gamma$ can be gained *without encoding across parallel channels*.

This decomposition is a result of properly combining multiple antennas and user grouping. Previous efforts to achieve the optimal $d_L(\gamma)$ either required encoding across the parallel channels (which meant increased subpacketization), or — if user grouping was enforced — would result in reduced DoF performance; for example, if we were to naively employ user grouping in the multi-server approach of [23], we would get a much reduced DoF $d = L + \frac{K}{L}\gamma < d_L(\gamma)$.

Separability between coded caching and PHY: This decomposition seen here, advocates that some degree of joint consideration between cache-placement and network structure (here, for receiver-side cache-placement and 'XOR' generation, we only need to know *the number* of transmitters and receivers), can yield very substantial improvements in the effective DoF, as well as can maintain substantial (although certainly not complete) robustness to not knowing the exact network structure during the cache-placement phase. While universal coded caching schemes that work obliviously of the structure of the communication network (cf. [35]) carry an advantage when it comes to some robustness against network-structure uncertainty, the work here shows an instance where non-separated schemes can yield a powerful decomposition that provides unboundedly better overall effective gains over universal schemes, by exploiting some of the structure of the network and by jointly considering coded caching and PHY.

C. Practicality and Timeliness of Result

The scheme consists of the basic implementable ingredients of ZF and low-dimensional coded caching, and it works for all values of $K, L, \gamma, K_T, \gamma_T$. Its simplicity and effectiveness suggest that having extra transmitting antennas (servers) can play an important role in making coded caching even more applicable in practice, especially at a time when subpacketization complexity is the clear major bottleneck of coded caching, and also at a time when multiple antennas and transmitter cooperation are standard ingredients in wireless communications.

D. Future Directions

An interesting direction would be to extend the decomposition ideas here to the setting of decentralized caching, where cache placement takes place without knowing which users will

participate in the delivery phase. It remains to be seen to what extent the multiplicative gains revealed here, persist in the decentralized scenario.

Another direction is to explore the connection between subpacketization and CSIT. As we have seen, the proposed scheme requires, at any given time, the knowledge of $L+K\gamma$ CSI vectors of length L. While this is already an improvement over a cache-free MIMO system (where for the same DoF, each CSI vector would have been of size $L+K\gamma$), it remains an open problem to explore how this principle of decomposition can be affected by having reduced CSI quality or fewer CSI vectors.

Furthermore the presented result can be useful in the context of distributed computing where — as we see from [36] — coded caching techniques can be used to reduce the communication load of a variety of distributed computing tasks. Such approaches suffer from very high subpacketization, and the decomposition-based ideas here can be used to substantially speed up such distributed computing methods. A first result in this direction can be found in [37] which shows how cooperation among the computing nodes in a D2D setting, can recreate the spatial multiplexing effect of multiple antennas, achieving the decomposition, and the corresponding speedup. It would be interesting to see other distributed computing scenarios that accept a similar exposition.

Appendix A
Adapting to the Cache-Aided Interference Scenario

We now consider the cache-aided interference scenario studied in [24], with K independent receivers, and with K_T independent transmitters, where each transmitter has normalized cache size $\gamma_T = M_T/N$, where fM_T is the size of each transmitter's cache. The scenario involves full connectivity (each receiver is connected to K_T transmitters), and no information can be exchanged between the transmitters.

For transmitter-side cache placement, we ask that each subfile is placed at exactly $K_T\gamma_T$ transmitters, and to do so, we consecutively cache whole files into the transmitters, such that the first transmitter caches the first M files, the second transmitter the next M files, and so on, modulo N. Specifically, using Z_{Tx_m} to denote the cache of transmitter $m \in [K_T]$, then the placement

$$Z_{\text{Tx}_m} = \left\{W_{1+(n-1)\bmod N} : n \in \{1+(m-1)M, ..., Mm\}\right\}$$

guarantees the redundancy requirements and memory constraints. Now, for any given subfile, the $K_T\gamma_T$ transmitters that have access to this file, will employ CSIT in order to play the role of the aforementioned $L = K_T\gamma_T$ antennas, by precoding this said subfile using the exact same precoders described before, allowing for simultaneous separation of the $L = K_T\gamma_T$ streams within any given group $\mathcal{G}_g$ of $L = K_T\gamma_T$ receivers. As before, the aforementioned caching allows for treatment of $K'\gamma + 1$ groups at a time, and a treatment of $K_T\gamma_T + K\gamma \le K$ users at a time (Corollary 6).

Finally it is easy to see that the above idea holds directly for the case where — in the above $K_T \times K$ cache-aided interference scenario with $\gamma_T \ge \frac{1}{K_T}$ — each transmitter has L_T transmitting antennas. In this case we can see that this same placement method has the desired property that each subfile is available at $L = K_T L_T \gamma_T$ antennas, yielding a sum-DoF of $K_T L_T \gamma_T + K\gamma$ which can be achieved with subpacketization

$$S_{K_T L_T \gamma_T} = \binom{K}{K_T L_T \gamma_T \quad K\gamma}{K_T L_T \gamma_T}$$

as mentioned in Corollary 7.

Appendix B
General Scheme: Removing the Integer Constraint

We proceed to remove the constraints $L|K$ and $L|K\gamma$, by applying as in [1] memory sharing. The results, after removing the integer constraints, will remain approximately the same except for a marginal increase in subpacketization[13] to at most $S_L \le K \cdot \max\left\{\binom{\lceil K/L\rceil}{\lceil K\gamma/L+1\rceil}, \binom{\lceil K/L\rceil}{\lfloor K\gamma/L+1\rfloor}\right\}$ and a relatively small reduction in the achieved DoF ($d_L(\gamma) = L+K\gamma$) by a multiplicative factor (gap) that is bounded above by 2 when $L > K\gamma$ and by $\frac{3}{2}$ when $L < K\gamma$, while the gap vanishes as $\frac{K\gamma}{L}$ increases.

To remove the constraint $L|K$ we will add to the system phantom users such that the new (hypothetical) number of users is $\hat{K} = L\left\lceil \frac{K}{L}\right\rceil$. Moreover, if $L \nmid \hat{K}\gamma$ we will perform memory sharing (cf. [1]) by splitting each file W_n into two parts, W'_n, W''_n of different sizes $|W'_n| = p|W_n|$ and $|W''_n| = (1-p)|W_n|$, and cache each part with normalized cache sizes $\gamma' = \frac{|Z_k \cap W'_n|}{|W'_n|} = \frac{L}{\hat{K}}\left\lfloor \frac{\hat{K}\gamma}{L}\right\rfloor$ and $\gamma'' = \frac{|Z_k \cap W''_n|}{|W''_n|} = \frac{L}{\hat{K}}\left\lceil \frac{\hat{K}\gamma}{L}\right\rceil$, which guarantees that $L|\hat{K}\gamma'$ and $L|\hat{K}\gamma''$. From the above we can see that $p = \frac{\gamma''-\gamma}{\gamma''-\gamma'}$.

Then, as the original scheme describes, we divide W'_n into $\binom{\hat{K}/L}{\hat{K}\gamma'/L}$ parts, W''_n into $\binom{\hat{K}/L}{\hat{K}\gamma''/L}$ parts, and cache from W'_n, W''_n according to Eq. (12). The corresponding subpacketization cost is thus bounded as

$$\begin{aligned} S &\le K \cdot \max\left\{\binom{\hat{K}/L}{\hat{K}\gamma'/L}, \binom{\hat{K}/L}{\hat{K}\gamma''/L}\right\} \\ &\le K \cdot \max\left\{\binom{\lceil K/L\rceil}{\lceil K\gamma/L\rceil+1}, \binom{\lceil K/L\rceil}{\lfloor K\gamma/L\rfloor+1}\right\} \end{aligned} \quad (25)$$

where the multiplicative factor of K is the one that upper bounds the subpacketization effect of splitting the file in two parts before subpacketizing each part. This effect is bounded by K because $p \ge 1/K$ by virtue of the fact that $K\gamma$ is an integer.[14]

[13] Note that for the settings in [1], [23], and [24], the aforementioned subpacketization costs in (1),(7) and (8) do not account for the extra subpacketization costs due to memory sharing.

[14] To see this, we rewrite γ as $\gamma = a/K$ where a is an integer, and then we see that $p = \frac{\gamma''-\gamma}{\gamma''-\gamma'} = \frac{\left\lceil \frac{\hat{K}a}{KL}\right\rceil - \frac{a\hat{K}}{KL}}{\left\lceil \frac{\hat{K}a}{KL}\right\rceil - \left\lfloor \frac{\hat{K}a}{KL}\right\rfloor} > \frac{1}{K}$ where, in the last step we used the fact that the denominator is 1 (unless it is zero, in which case there is no additional subpacketization cost), while for the numerator we have that $\left\lceil \frac{\hat{K}a}{KL}\right\rceil - \frac{a\hat{K}}{KL} > \frac{1}{K}$ because $L|\hat{K}a$.

Then, in order to derive a multiplicative gap on DoF, d_L^{nc}, that accounts for removing the two constraints, we will consider two separate cases. First, we will look at the case of $\hat{K}\gamma \leq L$. By applying memory sharing, we can see that each part will be cached with redundancy 0 and L respectively. This means that the completion time will be $T = \frac{m'}{0+L} + \frac{m''}{L+L}$, where $m' = Kp(1-\gamma')$ and $m'' = K(1-p)(1-\gamma'')$. Then, we can see that the completion time is upper-bounded $T \leq \frac{K(1-\gamma)}{L}$ and lower-bounded $T \geq \frac{K(1-\gamma)}{2L}$, which incorporates the facts that the performance cannot be worse than if there was no caching gains, but it cannot be better than if the caching gain was L. Using that, we can calculate the bounds of the DoF as follows

$$\frac{K(1-\gamma)}{L} \geq T \geq \frac{K(1-\gamma)}{2L}$$
$$\frac{K(1-\gamma)}{\frac{K(1-\gamma)}{2L}} \geq d_L^{nc} \geq \frac{K(1-\gamma)}{\frac{K(1-\gamma)}{L}}$$
$$2L \geq d_L^{nc} \geq L$$

which implies a gap of 2. Similarly, for the case where $K\gamma \in (qL, qL+q), \quad q = \{1, 2, ...\}$ we can see that the above gap becomes $\frac{q+2}{q+1}$.

REFERENCES

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

[2] U. Niesen and M. A. Maddah-Ali, "Coded caching with nonuniform demands," *IEEE Trans. Inf. Theory*, vol. 63, no. 2, pp. 1146–1158, Feb. 2017.

[3] J. Zhang, X. Lin, and X. Wang, "Coded caching under arbitrary popularity distributions," in *Proc. Inf. Theory Appl. Workshop (ITA)*, Feb. 2015, pp. 98–107.

[4] M. Ji, A. M. Tulino, J. Llorca, and G. Caire, "On the average performance of caching and coded multicasting with random demands," in *Proc. 11th Int. Symp. Wireless Commun. Syst. (ISWCS)*, Aug. 2014, pp. 922–926.

[5] S. S. Bidokhti, M. Wigger, and R. Timo, "Erasure broadcast networks with receiver caching," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2016, pp. 1819–1823.

[6] J. Zhang and P. Elia, "Wireless coded caching: A topological perspective," in *Proc. IEEE Int. Symp. Inf. Theory*, Jun. 2017, pp. 401–405.

[7] E. Lampiris, J. Zhang, and P. Elia, "Cache-aided cooperation with no CSIT," in *Proc. IEEE Int. Symp. Inf. Theory*, Jun. 2017, pp. 2960–2964.

[8] A. Ghorbel, M. Kobayashi, and S. Yang, "Cache-enabled broadcast packet erasure channels with state feedback," in *Proc. 53rd Annu. Allerton Conf. Commun., Control, Comput.*, Sep./Oct. 2015, pp. 1446–1453.

[9] J. Zhang and P. Elia, "Fundamental limits of cache-aided wireless BC: interplay of coded-caching and CSIT feedback," *IEEE Trans. Inf. Theory*, vol. 63, no. 5, pp. 3142–3160, May 2017.

[10] J. Hachem, N. Karamchandani, and S. Diggavi, "Content caching and delivery over heterogeneous wireless networks," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr./May 2015, pp. 756–764.

[11] M. Ji, G. Caire, and A. F. Molisch, "Fundamental limits of caching in wireless D2D networks," *IEEE Trans. Inf. Theory*, vol. 62, no. 2, pp. 849–869, Feb. 2016.

[12] F. Engelmann and P. Elia, "A content-delivery protocol, exploiting the privacy benefits of coded caching," in *Proc. WiOpt*, May 2017, pp. 1–6.

[13] K. Wan, D. Tuninetti, and P. Piantanida, "On the optimality of uncoded cache placement," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Sep. 2016, pp. 161–165.

[14] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "The exact rate-memory tradeoff for caching with uncoded prefetching," *IEEE Trans. Inf. Theory*, vol. 64, no. 2, pp. 1281–1296, Feb. 2018.

[15] S.-E. Elayoubi and J. Roberts, "Performance and cost effectiveness of caching in mobile access networks," in *Proc. 2nd Int. Conf. Inf.-Centric Netw.*, 2015, pp. 79–88.

[16] K. Shanmugam, M. Ji, A. M. Tulino, J. Llorca, and A. G. Dimakis, "Finite-length analysis of caching-aided coded multicasting," *IEEE Trans. Inf. Theory*, vol. 62, no. 10, pp. 5524–5537, Oct. 2016.

[17] M. Ji, K. Shanmugam, G. Vettigli, J. Llorca, A. M. Tulino, and G. Caire, "An efficient multiple-groupcast coded multicasting scheme for finite fractional caching," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2015, pp. 3801–3806.

[18] 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.

[19] Q. Yan, M. Cheng, X. Tang, and Q. Chen, "On the placement delivery array design for centralized coded caching scheme," *IEEE Trans. Inf. Theory*, vol. 63, no. 9, pp. 5821–5833, Sep. 2017.

[20] L. Tang and A. Ramamoorthy, "Coded caching with low subpacketization levels," in *Proc. IEEE Globecom Workshops*, Dec. 2016, pp. 1–6.

[21] C. Shangguan, Y. Zhang, and G. Ge. (2016). "Centralized coded caching schemes: A hypergraph theoretical approach." [Online]. Available: https://arxiv.org/abs/1608.03989

[22] K. Shanmugam, A. M. Tulino, and A. G. Dimakis, "Coded caching with linear subpacketization is possible using Ruzsa–Szeméredi graphs," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2017, pp. 1237–1241.

[23] S. P. Shariatpanahi, S. A. Motahari, and B. H. Khalaj, "Multi-server coded caching," *IEEE Trans. Inf. Theory*, vol. 62, no. 12, pp. 7253–7271, Dec. 2016.

[24] N. Naderializadeh, M. A. Maddah-Ali, and A. S. Avestimehr, "Fundamental limits of cache-aided interference management," *IEEE Trans. Inf. Theory*, vol. 63, no. 5, pp. 3092–3107, May 2017.

[25] A. Sengupta, R. Tandon, and O. Simeone, "Cache aided wireless networks: Tradeoffs between storage and latency," in *Proc. Annu. Conf. Inf. Sci. Syst. (CISS)*, Mar. 2016, pp. 320–325.

[26] Y. Cao, M. Tao, F. Xu, and K. Liu, "Fundamental storage-latency tradeoff in cache-aided MIMO interference networks," *IEEE Trans. Wireless Commun.*, vol. 16, no. 8, pp. 5061–5076, Aug. 2017.

[27] J. Hachem, U. Niesen, and S. N. Diggavi. (2016). "Degrees of freedom of cache-aided wireless interference networks." [Online]. Available: https://arxiv.org/abs/1606.03175

[28] J. S. P. Roig, D. Gündüz, and F. Tosato, "Interference networks with caches at both ends," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.

[29] S. Yang, K.-H. Ngo, and M. Kobayashi, "Content delivery with coded caching and massive MIMO in 5G," in *Proc. 9th Int. Symp. Turbo Codes Iterative Inf. Process. (ISTC)*, Sep. 2016, pp. 370–374.

[30] S. P. Shariatpanahi, G. Caire, and B. H. Khalaj, "Multi-antenna coded caching," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2017, pp. 2113–2117.

[31] E. Piovano, H. Joudeh, and B. Clerckx, "On coded caching in the overloaded MISO broadcast channel," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2017, pp. 2795–2799.

[32] J. Zhang, F. Engelmann, and P. Elia, "Coded caching for reducing CSIT-feedback in wireless communications," in *Proc. 53rd Allerton Conf. Commun., Control Comput.*, Sep./Oct. 2015, pp. 1099–1105.

[33] J. Zhang and P. Elia, "Feedback-aided coded caching for the MISO BC with small caches," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.

[34] M. A. Maddah-Ali and U. Niesen, "Cache-aided interference channels," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2015, pp. 809–813.

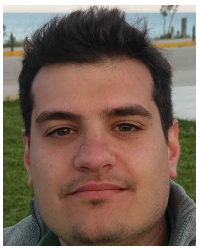
[35] N. Naderializadeh, M. A. Maddah-Ali, and A. S. Avestimehr, "On the optimality of separation between caching and delivery in general cache networks," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2017, pp. 1232–1236.

[36] S. Li, M. A. Maddah-Ali, and A. S. Avestimehr, "Coded MapReduce," in *Proc. 53rd Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Sep./Oct. 2015, pp. 964–971.

[37] E. Parrinello, E. Lampiris, and P. Elia, "Coded distributed computing with node cooperation substantially increases speedup factors," in *Proc. IEEE Int. Symp. Inf. Theory*, Jun. 2018.

[38] A. G. Davoodi and S. A. Jafar, "GDoF of the MISO BC: Bridging the gap between finite precision CSIT and perfect CSIT," in *Proc. IEEE Int. Symp. Inf. Theory*, Jul. 2016, pp. 1297–1301.

[39] E. Lampiris and P. Elia, "Achieving full multiplexing and unbounded caching gains with bounded feedback resources," in *Proc. IEEE Int. Symp. Inf. Theory*, Jun. 2018.



Eleftherios Lampiris received the bachelor's degree in physics and the M.Sc. degree in radioelectrology from the University of Athens, Greece. He is currently pursuing the Ph.D. degree with EURECOM, Sophia Antipolis, France. His interests lie in game theory and in practical aspects of information theory, including caching.

Petros Elia received the B.Sc. degree from the Illinois Institute of Technology, Chicago, IL, USA, in 1997, and the M.Sc. and Ph.D. degrees in electrical engineering from the University of Southern California, Los Angeles, CA, USA, in 2001 and 2006, respectively. He is currently a Professor with the Department of Communication Systems, EURECOM. His latest research deals with information theoretic aspects of caching, as well with different problems in the areas of complexity-constrained communications, multi in multi out, coding theory, and surveillance networks. He is a Fulbright Scholar. He was a co-recipient of the NEWCOM++ Distinguished Achievement Award from 2008 to 2011, for a sequence of publications on the topic of complexity in wireless communications. He is a recipient of the European Research Council Consolidator Grant on cache-aided wireless communications from 2017 to 2022.