Cooperative Wireless Mobile Caching

A Signal Processing Perspective

Soheil Mohajer, Itsik Bergel, and Giuseppe Caire

Cache-aided communications have shown potential for substantial improvement in network performance, which goes far beyond that of traditional caching. *Traditional caching* (i.e., the bringing and storing of data closer to the end users) is only efficient when a significant portion of the popular files can be locally stored. In cache-aided communications, however, information stored at one user is useful for interference mitigation even if it is requested only by another user. The core idea in cache-aided communication is to use this interference-cancellation opportunity to simultaneously serve multiple users by sending a sum of multiple packets. By creating opportunities for multicasting, the improved performance scales with the accumulated cache size at all users. This is a great advantage for modern networks, where the number of users is typically large, and a small amount of memory can easily be allocated at each user. This article presents the novel techniques of cache-aided communications while focusing on the signal processing aspects that lie in the heart of these schemes. In particular, we examine the three well-studied signal processing problems at the core of cache-aided communications: resource allocation, beamforming design, and interference mitigation.

Digital Object Identifier 10.1109/MSP.2019.2962507
Date of current version: 26 February 2020

Introduction

The increased demand for large files (e.g., media) has produced overwhelming network traffic. The nature of the data has shifted from voice and short messages to large files. It is expected that by 2020, 75% of total mobile data traffic will be attributable to video. The characteristics of this type of data can be exploited to improve the performance of data delivery networks. In particular, popular videos are typically repeatedly requested by multiple users in an asynchronous manner. Moreover, prime time is usually associated with videos, i.e., the demand peaks during certain hours of the day, rather than being uniformly distributed over time.

Caching, bringing the data closer to where they will be used and storing them locally, is an efficient approach used to exploit the characteristics of such large-size contents to reduce the network traffic. Currently, caching is being used for data delivery systems, e.g., Netflix and Facebook's photo caching. However, these systems work based on predicting content with high demand and storing popular files on local storage units close to the end users. The gain provided by such strategies is limited by the size of the local memory and the prediction accuracy. In other words, if the individual local storage units are not large enough to store a significant portion of the popular files, cached data will be almost useless (when users request other files) and the gain of caching will be negligible.

Despite the dramatic drop in the cost of memory, the storage size on individual fixed or mobile devices is still too small to store a substantial fraction of the popular data. However, mobile devices have become more popular and the number of mobile users is rapidly growing. This leads to a substantial amount of aggregate storage, which is distributed across the network. The question is: How can multiple small caches be utilized in a cooperative manner to perform as a giant cache?

Recently, Maddah-Ali and Niesen [1] introduced a novel caching technique, called *coded caching*, which provides a gain that scales with the total cache distributed over the network. Unlike classical caching, coded caching is beneficial even if packets requested by one user are stored in the cache of other users. This surprising caching benefit is due to the opportunity for *multicasting*, i.e., when multiple users can be served by a single transmission. With coded caching, each transmission is a combination of multiple requested packets, where each user can retrieve its desired packet by removing the interference using the data stored in its cache.

With coded caching, each transmission is a combination of multiple requested packets, where each user can retrieve its desired packet.

The new coded caching scheme was first proposed for a fixed and homogeneous single-hop network. Following this pioneering work [1], the benefits of coded caching have been studied for various scenarios and applications, including device-to-device (D2D) communication [2], dynamic settings where the placement is not necessarily controlled by the server (decentralized caching) [3], [4], multihop networks [5], cloud networks [6], and multiple-input, multiple-output (MIMO) settings [7]–[11]. Currently, the main attention of the community is on the practical challenges of coded caching under nonideal channel models, e.g., packet erasure or fading [10], [12]–[16].

Despite its name, the main feature of coded caching is not coding in the information-theoretic sense (e.g., error correction codes) but rather combining multiple packets, each requested by one receiver so that each target receiver can cancel the interfering packets using its previously stored data and retrieve its desired packet. This allows for simultaneously serving multiple users, and hence leads to an increase in the number of degrees of freedom (DoF) in the system. As a result, the cache-aided communication problem can be formulated in the signal processing language and solved using the tools and techniques of signal processing. In this article, we present the novel aspects of cache-aided communication while focusing on the signal processing problems that lie at the heart of these schemes. In particular, we present a simple and tractable problem formulation while highlighting the relations to common signal processing problems. We show that the challenges of cache-aided communication can be largely decomposed into three main problems: resource allocation, the design and coordination of beamforming/precoding vectors, and interference mitigation. These problems have been well studied in the signal processing community in different contexts, and (close to) optimum solutions are advised. However, these solutions must be adapted to the emerging field of cache-aided communication before this technique can be utilized for practical networks.

As practical implementations of cache-aided communication systems are approaching, many practical concerns and complexity issues still need to be addressed. Thus, the signal processing community can make a significant contribution to the derivation and optimization of novel cache-aided communication schemes. Furthermore, the signal processing community's experience with solving the aforementioned problems and, in particular, with handling the imperfections and practical constraints in the network can be crucial as cache-aided communication continues to be adopted for practical implementations.

Relevant works

The topic of coded caching has received a considerable amount of attention in recent years and is highlighted in several survey papers. In particular, [17] addressed the scaling laws of throughput in wireless networks with caching, with a special focus on the D2D approach. The challenges of edge caching in wireless networks are studied in [18], where the differences between wired and wireless caching are outlined. Specifically, this article discusses the essential limitations of wireless caching and the possible tradeoffs between spectral efficiency, energy efficiency, and cache size. Context-aware networks using edge/cloud computing and the exploitation of big data analytics are investigated in [19]. A tutorial on coded caching is presented in [20], which provides an introduction for the seminal and pioneering papers that opened new avenues in caching as well as a brief overview of existing caching solutions from an information-theoretic perspective. Moreover, [20] surveys some of the industrial challenges of caching and identifies bottleneck issues that need to be resolved to unleash the full potential of caching in practical systems.

The main distinction of this article from other surveys is its focus on the signal processing aspects of caching. In particular, we focus on interference mitigation using caching in wireless networks. This article also emphasizes the role of spatial reuse and the open issues that must be addressed prior to a practical adoption of coded caching.

Notation

Throughout this article, we use calligraphic symbols (e.g., $\mathcal{W}$) to denote sets. The size of a set $\mathcal{A}$ is denoted by $|\mathcal{A}|$. For an integer N, we denote the set $\{1, 2, \ldots, N\}$ by $[N]$.

Classical coded caching

The application of caching was conventionally limited to prestoring popular contents in storage units close to the end users and delivering the data from the local copies to reduce the server load and bandwidth requirements [21], [22]. The main challenge there is to predict users' requests before they actually ask for the data [21], [23], [24]. More importantly, the advantage provided by conventional caching schemes is limited to its capacity of local memory/cache and can be negligible if the cache size is much smaller than the total size of popular data.

In wireless networks, the available memory on individual devices is small compared to the database size; however, with the growing popularity of mobile devices, plenty of such small-size memories are available and distributed over all the

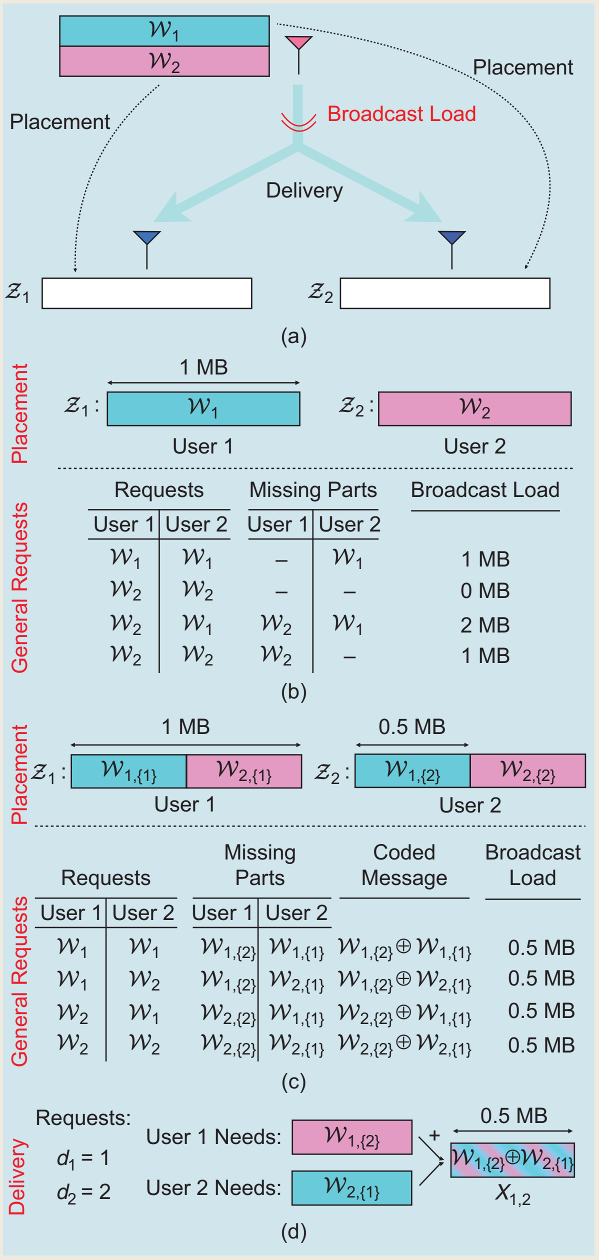
Traditional Versus Coded Caching for Two Users

Consider the network in Figure S1(a) with $N = 2$ files, namely $\mathcal{W}_1$ and $\mathcal{W}_2$, each of size 1 megabyte, and $K = 2$ users, each equipped with a local cache size of 1 megabyte. Assume a perfect broadcast link from the server to the receivers. Figure S1(b) displays a naive placement strategy, where each user caches one of the files. Once the user requests are revealed, the server must transmit all of the files that were requested and not stored at the requesting user. The tables show the files to be broadcast and the load of delivery under different demands. The load of delivery is not fixed and its average size over different demands is 1 megabyte.

Figure S1(c) depicts a smarter placement strategy wherein each file is divided into two halves, each of size 0.5 megabytes, i.e., $\mathcal{W}_1 = (\mathcal{W}_{1,\{1\}}, \mathcal{W}_{1,\{2\}})$ and $\mathcal{W}_2 = (\mathcal{W}_{2,\{1\}}, \mathcal{W}_{2,\{2\}})$. Each user caches one half of each file, i.e, the cache content of user 1 is $\mathcal{Z}_1 = (\mathcal{W}_{1,\{1\}}, \mathcal{W}_{2,\{1\}})$ and that of user 2 is $\mathcal{Z}_2 = (\mathcal{W}_{1,\{2\}}, \mathcal{W}_{2,\{2\}})$. Again, the subfiles to be broadcast for each demand are listed in the table. For any demand, the server must send two subfiles, each of size 0.5 megabytes.

However, in all cases, the load of broadcast can be reduced to only 0.5 megabytes by sending a coded packet. Thus, coded caching can send twice as much information over the same period of time.

For instance, consider demands $d_1 = 1$ and $d_2 = 2$ shown in Figure S1(d), i.e., assume user 1 needs $\mathcal{W}_1$ and user 2 needs $\mathcal{W}_2$. Because user 1 has already cached the first half of $\mathcal{W}_1$, it only needs the second half of $\mathcal{W}_1$, which is $\mathcal{W}_{1,\{2\}}$. Similarly, user 2 needs $\mathcal{W}_{2,\{1\}}$ to be able to recover its desired file, $\mathcal{W}_2$. Instead of sending these two packets separately (uncoded caching), the server can combine them and send $X = \mathcal{W}_{1,\{2\}} \oplus \mathcal{W}_{2,\{1\}}$. User 1 can remove the interfering part, $\mathcal{W}_{2,\{1\}}$, from X using its cached data, $\mathcal{Z}_1$, and recover $\mathcal{W}_{1,\{2\}}$. Similarly, user 2 can decode its missing part, $\mathcal{W}_{2,\{1\}}$. Clearly, X is the summation (e.g., binary exclusive-OR) of two (binary) packets of size 0.5 megabytes, and hence we are only broadcasting 0.5 megabytes in a coded scheme, whereas separately broadcasting $\mathcal{W}_{1,\{2\}}$ and $\mathcal{W}_{2,\{1\}}$ requires 1 megabyte of data transmission. Similar arguments hold for all of the other demands, as presented in the figure.



$\mathcal{W}_1$
$\mathcal{W}_2$
Placement
Placement
Broadcast Load
Delivery
$\mathcal{Z}_1$
$\mathcal{Z}_2$
(a)
Placement
1 MB
$\mathcal{Z}_1$: $\mathcal{W}_1$
$\mathcal{Z}_2$: $\mathcal{W}_2$
User 1
User 2
General Requests
(b)
Placement
1 MB
0.5 MB
$\mathcal{Z}_1$: $\mathcal{W}_{1,\{1\}}$ $\mathcal{W}_{2,\{1\}}$
$\mathcal{Z}_2$: $\mathcal{W}_{1,\{2\}}$ $\mathcal{W}_{2,\{2\}}$
User 1
User 2
General Requests
(c)
Delivery
Requests:
$d_1 = 1$
$d_2 = 2$
User 1 Needs: $\mathcal{W}_{1,\{2\}}$
User 2 Needs: $\mathcal{W}_{2,\{1\}}$
+
0.5 MB
$\mathcal{W}_{1,\{2\}} \oplus \mathcal{W}_{2,\{1\}}$
$X_{1,2}$
(d)

Requests		Missing Parts		Broadcast Load
User 1	User 2	User 1	User 2	
$\mathcal{W}_1$	$\mathcal{W}_1$	–	$\mathcal{W}_1$	1 MB
$\mathcal{W}_2$	$\mathcal{W}_2$	–	–	0 MB
$\mathcal{W}_2$	$\mathcal{W}_1$	$\mathcal{W}_2$	$\mathcal{W}_1$	2 MB
$\mathcal{W}_2$	$\mathcal{W}_2$	$\mathcal{W}_2$	–	1 MB

Requests		Missing Parts		Coded Message	Broadcast Load
User 1	User 2	User 1	User 2		
$\mathcal{W}_1$	$\mathcal{W}_1$	$\mathcal{W}_{1,\{2\}}$	$\mathcal{W}_{1,\{1\}}$	$\mathcal{W}_{1,\{2\}} \oplus \mathcal{W}_{1,\{1\}}$	0.5 MB
$\mathcal{W}_1$	$\mathcal{W}_2$	$\mathcal{W}_{1,\{2\}}$	$\mathcal{W}_{2,\{1\}}$	$\mathcal{W}_{1,\{2\}} \oplus \mathcal{W}_{2,\{1\}}$	0.5 MB
$\mathcal{W}_2$	$\mathcal{W}_1$	$\mathcal{W}_{2,\{2\}}$	$\mathcal{W}_{1,\{1\}}$	$\mathcal{W}_{2,\{2\}} \oplus \mathcal{W}_{1,\{1\}}$	0.5 MB
$\mathcal{W}_2$	$\mathcal{W}_2$	$\mathcal{W}_{2,\{2\}}$	$\mathcal{W}_{2,\{1\}}$	$\mathcal{W}_{2,\{2\}} \oplus \mathcal{W}_{2,\{1\}}$	0.5 MB

FIGURE S1. Traditional versus coded caching. (a) System model, (b) placement of a subset of files, (c) placement of parts of files, and (d) coded delivery.

networks. A key question is: How can a file stored at one user's cache be utilized to reduce the network traffic when it is only requested by other users?

Recently, Maddah-Ali and Niesen [1] demonstrated that distributed storage units can be utilized in a cooperative manner to achieve a global gain that is proportional to the total available cache. Coded caching is a central placement mechanism that stores packets of the popular files in users' memory all over the network. This allows for an opportunistic multicasting in the delivery phase of the coded caching, in which a single (coded) packet can simultaneously serve multiple requests. The multicasting of packets to several users increases the utility of packets, and reduces the network load and congestion probability during peak traffic time (at the delivery phase). A simple coded caching scheme for a two-user scenario is depicted in "Traditional Versus Coded Caching for Two Users."

More generally, the single shared-link network studied in [1] consists of a network with K users and a server [base station (BS)] with a library of N files $\{\mathcal{W}_1, \mathcal{W}_2, \ldots, \mathcal{W}_N\}$, each consisting of F bits, i.e., $|\mathcal{W}_n| = F$ for $n \in [N]$. For the sake of simplicity, we assume that the number of users does not exceed the size of the library, i.e., $N \geq K$. Each user is equipped with a storage memory to store up to MF bits in the placement phase, which happens during the off-peak time of the network. Then, in the delivery phase, each user requests one file and the server is required to serve all the users by broadcasting a message through a perfect and shared channel.

Placement phase

In the placement phase, prior to receiving users' requests and during the off-peak time of the network, the server selects subsets of all the bits in the library and stores them in the memory of user $k \in [K]$, which is denoted by $\mathcal{Z}_k$ with $|\mathcal{Z}_k| \leq MF$. The placement strategy of coded caching proposed by Maddah-Ali and Niesen (referred to as the *MAN* scheme) is symmetric across files and users and allocates a $1/N$ fraction of each user's cache to each file. Thus, each user will store a $\mu = M/N$ fraction of each file in its cache.

Focusing on the case that $\alpha = KM/N$ is an integer, the MAN scheme stores each packet in the cache of α users. Thus, each file is split into $\binom{K}{\alpha}$ equal segments, each of size $F/\binom{K}{\alpha}$ bits. It is more convenient to index the segments by subsets of $[K]$ of size α. File $\mathcal{W}_i$ will therefore be partitioned into $\{\mathcal{W}_{i,\mathcal{S}} : \mathcal{S} \subseteq [K], |\mathcal{S}| = \alpha\}$, and each segment $\mathcal{W}_{i,\mathcal{S}}$ will be stored in the cache of every user k satisfying $k \in \mathcal{S}$ (see Figure 1). The content of the cache at user k will be

$$\mathcal{Z}_k^{\text{MAN}} = \{\mathcal{W}_{n,\mathcal{S}} : \mathcal{S} \subseteq [K], |\mathcal{S}| = \alpha, \mathcal{S} \ni k\}. \tag{1}$$

Using this method, each user will cache $\binom{K-1}{\alpha-1}$ segments out of a total $\binom{K}{\alpha}$ segments, for each file. It is easy to verify that the number of bits from each file stored in each user's cache is $KF/\alpha = \mu F$.

Delivery phase

Upon receiving the requests $\{d_k : k \in [K]\}$ from the users (i.e., user k requests file $\mathcal{W}_{d_k}$), the server broadcasts a message, X, which is a sequence of coded packets, to serve all user demands. The formation of this combination depends on the actual demand profile $(d_1, \ldots, d_k)$ and thus can be denoted by $X_{(d_1,\ldots,d_k)}$. Upon receiving X, user k should be able to reconstruct its desired file $\mathcal{W}_{d_k}$ using its cache content $\mathcal{Z}_k$ and X, i.e.,

$$(X_{(d_1,\ldots,d_k)}, \mathcal{Z}_k) \mapsto \mathcal{W}_{d_k}, \quad k \in [K].$$

In the MAN scheme, at the delivery phase, the server provides user k with missing (not cached) segments of its requested file $\mathcal{W}_{d_k}$, i.e., all $\mathcal{W}_{d_k,\mathcal{S}}$ where $k \notin \mathcal{S}$. However, such a missing segment is cached in the local memory of exactly α other users, indexed by elements of $\mathcal{S}$. The symmetric placement guarantees a similar situation with respect to every other user in $\mathcal{A} = \mathcal{S} \cup \{k\}$. Thus, at any given time, the server can simultaneously serve a subset $\mathcal{A}$ of $\alpha + 1$ users by multicasting a linear combination of all such packets. That is, for any subset $\mathcal{A} \subseteq [K]$ with $|\mathcal{A}| = \alpha + 1$, the server sends

$$X_A = \bigoplus_{j \in \mathcal{A}} \mathcal{W}_{d_j, \mathcal{A}\setminus\{j\}}, \qquad \forall \mathcal{A} \subseteq [K], |\mathcal{A}| = \alpha + 1. \tag{2}$$

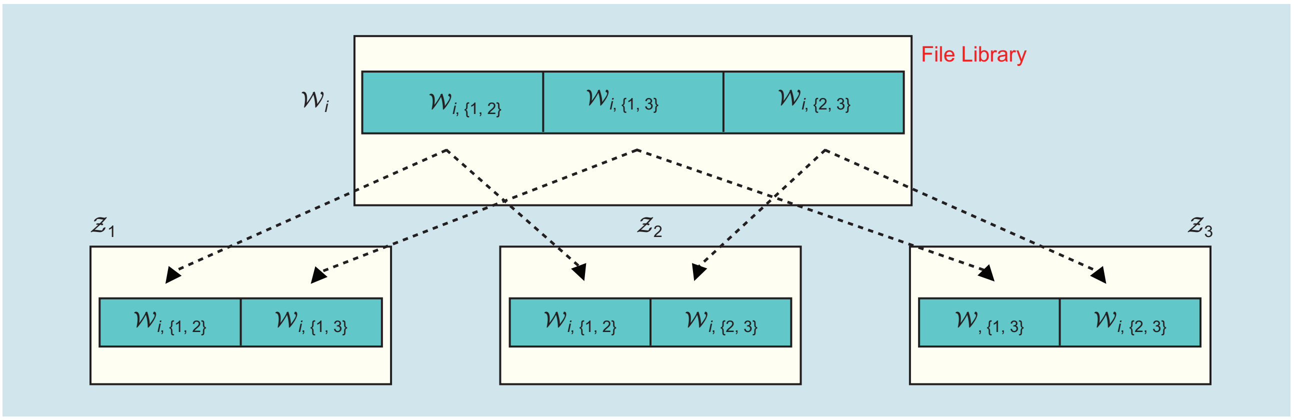

File Library
$\mathcal{W}_i$
$\mathcal{W}_{i,\{1,2\}}$
$\mathcal{W}_{i,\{1,3\}}$
$\mathcal{W}_{i,\{2,3\}}$
$\mathcal{Z}_1$
$\mathcal{Z}_2$
$\mathcal{Z}_3$
$\mathcal{W}_{i,\{1,2\}}$
$\mathcal{W}_{i,\{1,3\}}$
$\mathcal{W}_{i,\{1,2\}}$
$\mathcal{W}_{i,\{2,3\}}$
$\mathcal{W}_{,\{1,3\}}$
$\mathcal{W}_{i,\{2,3\}}$

FIGURE 1. A file partitioning used for placement.

Note that we consider the file segments as bit streams of the same length, and $\oplus$ denotes binary-exclusive-OR (XOR). It is worth mentioning that later in this article, we show that a similar operation can be performed using finite field summation on modulated sequences, which will be denoted by $+$ instead of $\oplus$. Each user $k \in \mathcal{A}$ has every term in this linear combination cached in its memory, except $\mathcal{W}_{d_k,\mathcal{A}\setminus\{k\}}$. So, the user can suppress the interference to recover its desired packet. Such a combined packet has utility $(\alpha+1)$, because it can simultaneously serve $(\alpha+1)$ users. An example of the MAN scheme with $K=3$ users is given in "Three-User Coded Caching."

Performance evaluation

The *load of delivery* is defined as the normalized size of X, which is the sum of the (normalized) size of the broadcast combinations, i.e.,

$$D = \frac{|X|}{F} = \frac{1}{F}\sum_{\substack{\mathcal{A}\subseteq[K]\\ |\mathcal{A}|=\alpha+1}} |X_{\mathcal{A}}| = \frac{1}{F}\binom{K}{\alpha+1}\frac{F}{\binom{K}{\alpha}} = \frac{K(1-\mu)}{1+\alpha}. \quad (3)$$

This leads to a delivery time [1] of

$$T_{\text{centralized-caching}} = \frac{D\cdot F}{R} = \frac{K\left(1-\frac{M}{N}\right)}{1+\frac{KM}{N}}\frac{F}{R}, \quad (4)$$

where R is the rate supported by the common and shared links from the server to the users. This represents a significant improvement compared to that of conventional and uncoded caching, which has a load of $D_{\text{uncoded}} = K(1-\mu)$ and requires a delivery time of

$$T_{\text{uncoded}} = K\left(1-\frac{M}{N}\right)\frac{F}{R}. \quad (5)$$

Another interpretation of (4) can be expressed: Each user has cached μF bits of its desired file and requires another $(1-\mu)F$ bits. Thus, a total of $K(1-\mu)F = K(1-M/N)F$ bits are requested by all the users, which can be sent at a common rate of R. The factor $1-M/N$ is the (typically small) local caching gain due to the parts of the requested files that where prestored in the cache of the requesting users. The more

Three-User Coded Caching

Consider $K=3$ users in the system, where all of the users receive information from the server through a common perfect broadcast link that supports a rate of R bits/s. Each user is equipped with a cache of size MF, where $M=2N/3$, i.e., each user can prefetch two-thirds of each file. We first partition each of the files into

$$\binom{K}{MK/N} = \binom{3}{2} = 3$$

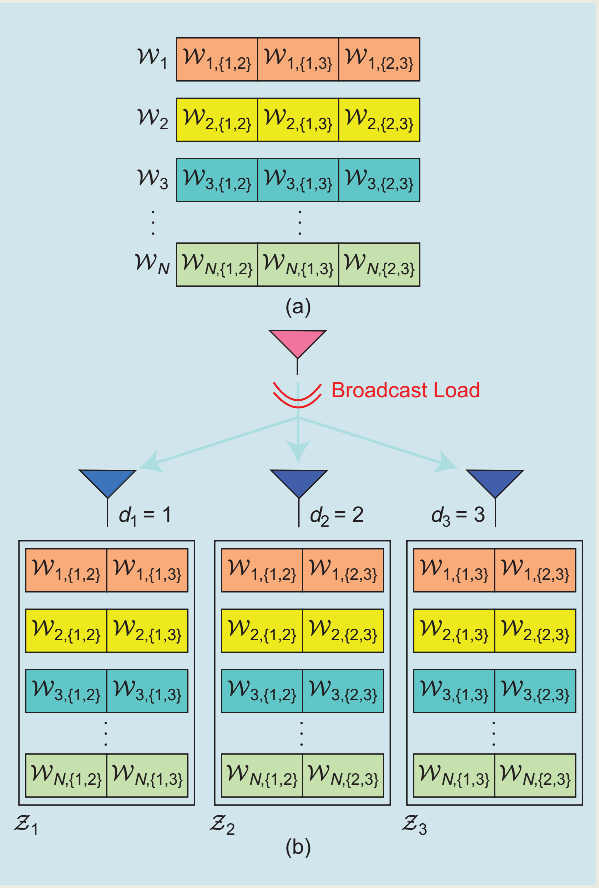
segments and label them with subsets of $\{1, 2, 3\}$ of size $MK/N = 2$, as shown in Figure S2(a). Then, each user k prefetches all of the segments in $\mathcal{W}_{i,\mathcal{S}}$ with $k \in \mathcal{S}$. The resulting placement is displayed in Figure S2(b). Assume that user k requested file $\mathcal{W}_k$, for $k = 1$, 2, and 3. Each user has two segments of its desired file in the cache, and the remaining segment should be sent during the delivery phase.

More precisely, segments $\mathcal{W}_{1,\{2,3\}}$, $\mathcal{W}_{2,\{1,3\}}$, and $\mathcal{W}_{3,\{1,2\}}$ must be delivered to the users. To this end, the base station (BS) transmits the combination

$$X = \mathcal{W}_{1,\{2,3\}} \oplus \mathcal{W}_{2,\{1,3\}} \oplus \mathcal{W}_{3,\{1,2\}}. \quad \text{(S1)}$$

Upon receiving X, user 1 can remove $\mathcal{W}_{2,\{1,3\}}$ and $\mathcal{W}_{3,\{1,2\}}$ using its cache content and recover $\mathcal{W}_{1,\{2,3\}}$. A similar argument holds for users 2 and 3.

Note that in each time slot, the BS is broadcasting a segment (combination) whose size is one-third of the size of a file. Given the common rate of R bits/s and the file length of F bits, the transmission takes only $F/3R$ seconds. During this time, three files are delivered to the users, resulting in an overall network throughput of $3F/(F/3R) = 9R$ bits/s.



$\mathcal{W}_1$ $\mathcal{W}_{1,\{1,2\}}$ $\mathcal{W}_{1,\{1,3\}}$ $\mathcal{W}_{1,\{2,3\}}$
$\mathcal{W}_2$ $\mathcal{W}_{2,\{1,2\}}$ $\mathcal{W}_{2,\{1,3\}}$ $\mathcal{W}_{2,\{2,3\}}$
$\mathcal{W}_3$ $\mathcal{W}_{3,\{1,2\}}$ $\mathcal{W}_{3,\{1,3\}}$ $\mathcal{W}_{3,\{2,3\}}$
$\mathcal{W}_N$ $\mathcal{W}_{N,\{1,2\}}$ $\mathcal{W}_{N,\{1,3\}}$ $\mathcal{W}_{N,\{2,3\}}$
(a)
Broadcast Load
$d_1 = 1$
$d_2 = 2$
$d_3 = 3$
$\mathcal{Z}_1$
$\mathcal{Z}_2$
$\mathcal{Z}_3$
(b)

FIGURE S2. A cache-aided communication with $K=3$ users. (a) File partitioning and (b) cache placement.

significant gain comes from the fact that each transmission can simultaneously serve $1+\alpha=1+KM/N$ users. This global caching gain is due to the joint placement and delivery scheme. In the context of wireless communication, this can be captured as a $(1+\alpha)$ prefactor for the rate, which is referred to as its *DoF.*

We also use network throughput to quantify the overall number of bits (including the prestored segments) delivered to the users per unit of time. Hence, the throughput of the basic MAN scheme will be $KF/T=(1+\alpha)/(1-\mu)$. Note that the network throughput grows unboundedly as $\mu \to 1$.

For non-integer values α we can use *memory sharing*, i.e., the network can be treated as an interpolation between two separate systems, one with cache $KM_1/N=\lfloor KM/N \rfloor$ and another with $KM_2/N=\lceil KM/N \rceil$. Hence, the delivery load and transmission time can be obtained as a linear combination of those of networks with two closest integer values.

With the description of the placement and delivery phases of MAN presented in this section, we provided the necessary background information to understand the signal processing aspects of cache-aided communication discussed in the "Signal Processing Problem Formulation" section. We refer interested readers to the "Coded Caching: A Broader Picture" section, where some of the most important follow-up works related to coded caching are presented.

Signal processing problem formulation

Linear combining versus coding

The main idea of cache-aided communication is the joint transmission of information to several users, where each user is able to decode its desired data using the data stored in its cache. The original approach used an XORing of the data required by the different users after making sure that all served users have their interfering data in their cache (as described previously). The so-called coded caching refers to the XOR operation (addition in the binary field), which is mostly used in the field of coding and not very typical as a signal processing technique. However, the practical implementation of such schemes in wireless communications will definitely be used in the field of signal processing and will typically not include XORing.

To introduce the signal processing aspects, we first note that a superposition of modulated signals instead of the XORing in (2) (see also "Traditional Versus Coded Caching for Two Users") will do the work at a high signal-to-noise ratio (SNR) [25]. If the transmitter transmits $X=\tilde{\mathcal{W}}_{1,\{2\}}+\tilde{\mathcal{W}}_{2,\{1\}}$ where $\tilde{\mathcal{W}}_{i,\{k\}}$ is the modulated version of $\mathcal{W}_{i,\{k\}}$ and + is a simple addition of the coded and modulated data, then each user can extract its desired information by simple subtraction (e.g., user 1 will estimate $\hat{\mathcal{W}}_{1,\{2\}}=X-\tilde{\mathcal{W}}_{2,\{1\}}$). The only drawback of this approach is that transmitting $\tilde{\mathcal{W}}_{1,\{2\}}+\tilde{\mathcal{W}}_{2,\{1\}}$ will take twice as much power than will transmitting $(\mathcal{W}_{1,\{2\}} \oplus \mathcal{W}_{2,\{1\}})$. This difference becomes negligible at a high SNR.

The problem becomes more interesting in multiantenna systems, where the information can be further differentiated in the spatial domain. In this case, the XORing alternative has a disadvantage, as it requires that all served users will be able to decode the joint message. Thus, the joint message must be sent in a "direction" and at a code rate that will allow for proper detection by all of the users. Conversely, the simple addition of the massages can be much more efficient because each user is only required to decode its own message (and interference subtraction does not require decoding).

Adding messages is therefore an important alternative. Furthermore, this allows for simple adaptations of many signal processing techniques. In the following section, we focus on this approach and give a straightforward description of a caching scheme that has no "coding," i.e., the cache is placed in the users' memory without coding and the BS transmits sums of modulated data for different users.

System model and problem formulation

Cache placement

The notations for the cache placement were introduced in the "Placement Phase" section. At the placement phase, the BS divides each file into segments indexed by the subset $\mathcal{S}$, and stores in the cache of user k all segments $\mathcal{W}_{i,\mathcal{S}}$ for which $k \in \mathcal{S}$. Thus, the cache content of user k is $\mathcal{Z}_k=\{\mathcal{W}_{n,\mathcal{S}}: n \in [N], \mathcal{S} \subset [K], k \in \mathcal{S}\}$. We denote the normalized cache size of user k by $M_k=|\mathcal{Z}_k|/F$, where, for generality's sake, we allow the cache sizes to be different. The overall performance is mostly dominated by the average cache size $MF=(1/K)\sum_k M_k F$.

Transmission and channel model

At the beginning of the delivery phase, once the user requests, $\{d_k\}$, are known, each requested segment, $\mathcal{W}_{d_k,\mathcal{S}}$, is encoded at the rate supported by its requesting user. If multiple users request the same file, it will be encoded at the rate that will enable decoding by the weakest user (while other users may be able to decode the message after receiving only some of the coded symbols).

We denote with $\tilde{\mathcal{W}}_{i,S}$ the digitally modulated symbol set resulting from mapping the data packet $\mathcal{W}_{i,S}$ into a suitable signal space codebook. In this article, we do not discuss the details of how this can be done. Any suitable coded-modulation technique (e.g., low-density parity check code concatenated with a suitable high-order signal constellation) may be a practical implementation of the scheme.

Consider a multiple-input, single-output (MISO) communication scheme, where the BS has L antennas and each user has a single antenna. The nth symbol after match filtering and sampling at the kth user is given by

$$y_k[n]=\mathbf{h}_k\mathbf{x}[n]+z_k[n], \tag{6}$$

where $\mathbf{x}[n]$ is the transmit vector, $z_k[n] \sim \mathbb{C}\mathcal{N}(0, N_0)$ is the additive noise sample, N_0 is the power spectral density of the complex white Gaussian noise at each receiver, and $\mathbf{h}_k \in \mathbb{C}^{1\times N}$ is the channel vector from the BS to user k. For this description, we focus on the case where the BS has perfect channel state information.

System requirements

The system objective is to allow each user to decode each of its desired segments, while transmitting the least number of symbols. The ability to detect a message depends on the exact system definition. Using the general definition given so far, the ability to decode can be stated using information-theoretic terms: we need a sufficient amount of mutual information between the transmitted symbols and the received signal together with the cache content at the specific user.

However, this formulation does not reveal the structure of the problem and, in particular, does not tell us how to construct the transmitted symbol vectors. Thus, we turn to linear precoding to obtain more insight.

Linear precoding

General linear precoding

For linear precoding, we assign symbols from several segments to be transmitted at every time interval. The transmission scheduling is described by a sequence of transmission assignments. Each transmission assignment is a set $\mathcal{T}[n]$, where $(i,\mathcal{S}) \in \mathcal{T}[n]$ indicates that a symbol from $\tilde{\mathcal{W}}_{i,\mathcal{S}}$ is transmitted at time n. Then, the transmitted vector is constructed by

$$\mathbf{x}[n] = \sum_{(i,\mathcal{S})\in\mathcal{T}[n]} \sqrt{E_{i,\mathcal{S}}[n]} \cdot \mathbf{f}_{i,\mathcal{S}}[n]\,\tilde{u}_{i,\mathcal{S}}[n], \tag{7}$$

where $E_{i,\mathcal{S}}[n]$ is the energy assigned for this transmission, $\mathbf{f}_{i,\mathcal{S}}[n]$ is the precoding vector (normalized to $\|\mathbf{f}_{i,\mathcal{S}}[n]\|^2 = 1$), and $\tilde{u}_{i,\mathcal{S}}[n] \in \tilde{\mathcal{W}}_{i,\mathcal{S}}$ is the transmitted symbol (a different symbol from the segment at each assigned time).

Considering a specific symbol, described by $(i,\mathcal{S}) \in \mathcal{T}[n]$, user k will need to decode this symbol if $d_k = i$ (i.e., if user k requested this file) and $k \notin \mathcal{S}$ (i.e., the symbol is not stored at user k). Attempting to decode the symbol, the desired signal gain is

$$A_{k,i,\mathcal{S}}[n] = \sqrt{E_{i,\mathcal{S}}[n]} \cdot \mathbf{h}_k \mathbf{f}_{i,\mathcal{S}}[n]. \tag{8}$$

The receiver can subtract all of the cached symbols, and the residual noise plus interference power is given by the conditional variance

$$\sigma^2_{k,i,\mathcal{S}}[n] = \mathrm{Var}(y_k[n] - A_{k,i,\mathcal{S}}[n]\,\tilde{u}_{i,\mathcal{S}}[n] \,|\, \mathcal{Z}_k). \tag{9}$$

To ensure the proper decoding of all the requested files, we must verify that each segment is decodable at all requesting users. A common approximation for the decodability of a packet uses the Shannon capacity of an additive white Gaussian noise channel, with a multiplicative constant that represents that system imperfections. This constant, γ, is commonly referred to as the *Shannon gap* and takes values between -2 and -10 dB. Thus, we say that segment $\mathcal{W}_{d_k,\mathcal{S}}$ is decodable at user k if

$$\sum_{n:(d_k,\mathcal{S})\in\mathcal{T}[n]} \log_2\left(1 + \gamma \frac{|A_{k,d_k,\mathcal{S}}[n]|^2}{\sigma^2_{k,d_k,\mathcal{S}}[n]}\right) > |\mathcal{W}_{d_k,\mathcal{S}}|. \tag{10}$$

Hence, cache-aided communication with linear precoding is solvable if there exist assignments, $\mathcal{T}[n]$, and matching precoding vectors, such that all users can decode all segments of their requested files. This is still a difficult assignment problem, and the optimal linear assignment is not yet known. The only approach that has been solved thus far is that of limiting the discussion to zero-forcing (ZF) precoding. We note that ZF is typically efficient at the high SNR regime.

The ZF system

Using this approach, we make the following limiting assumptions as compared to the general linear case. These assumptions result in the most tractable problem formulation thus far:

- Each beamforming vector is selected in the unique direction that is orthogonal to the channel vectors of $L-1$ users.
- At any given time, exactly $L + \alpha = L + KM/N$ users are active.
- After cache subtraction, all symbols are detected in the presence of noise only.
- The transmission rate to each user is set independently from the scheduling decisions.

Also, for simplicity's sake, we describe only the worst case scenario for the delivery phase, i.e., the case where all users request different files.

The first assumption allows for $L-1$ users to not be interfered with during the transmission to another user. To comply also with the second and third assumptions, we need to remove the interference at additional α users by cache subtraction. Hence, every symbol must be stored at the cache of exactly α users. Compared to (9), the third assumption guarantees that for all the symbols, $\sigma^2_{k,i,\mathcal{S}}[n] = N_0$.

The last assumption is required for simplifying the transmission scheduling. As stated previously, for each symbol, the choice of precoding vectors is completely determined by the choice of $L-1$ users that receive messages at the same time and cannot subtract the given message using their cache. Thus, the effective gain for each user (and its achievable rate) will be different depending on the combination of interfered users. This may significantly complicate the scheduling [25].

To resolve this, most works turned to either the high SNR or ergodic fading regime. With the high SNR regime, we consider the performance as the transmission power grows to infinity. In such an asymptotic scenario, the resulting rate is proportional to the logarithm of the transmission power. Hence, in this regime it is assumed that the actual channel gains are negligible and all rates are approximately equal.

Another approach, which does not lead to equal rates for all users, assumes that each transmission has sufficient-enough diversity so that its rate will converge to its expected rate. For example, the ergodic fading regime can be approached in a wideband system, where the transmission bandwidth is large and spread over many frequency bins. When the bandwidth, B, is large enough compared to the channel coherence time, using the law of large numbers, the spectral efficiency converges to its expectation:

$$\frac{R_k}{B} \to \mathbb{E}\left[\log_2\left(1 + \gamma \frac{E_{d_k,\mathcal{S}}[n]}{N_0} |\mathbf{h}_k \mathbf{f}_{d_k,\mathcal{S}}[n]|^2\right)\right]. \tag{11}$$

This limit results in a rate per user, that is independent of the other users in the network. This independent rate can still be different for each user. Yet, it allows a tractable problem formulation.

Once the rate per user is determined and does not depend on the transmission scheduling, checking whether a segment is decodable can be done by simply counting the symbols received for this segment by the intended user. In this section, we give a mathematical statement of the conditions that guarantee decodability. Prior to that, we give a mathematical formulation of the aforementioned assumptions, i.e., we define what transmission assignments, $\mathcal{T}[n]$, are valid.

Let $\mathcal{K}(\mathcal{T}[n]) = \{k : (d_k, \mathcal{S}) \in \mathcal{T}[n]\}$ be the set of active users for the specific assignment $\mathcal{T}[n]$. A transmission assignment, $\mathcal{T}[n]$, is valid if $|\mathcal{K}(\mathcal{T}[n])| = L + \alpha$ and for each $(d_k, \mathcal{S}) \in \mathcal{T}[n]$:

- Each bit is stored at α users, i.e., $|\mathcal{S}| = \alpha$.
- All users that store the relevant segment are also actively receiving data, i.e., $\mathcal{S} \subset \mathcal{K}(\mathcal{T}[n])$.
- The BS does not send a segment to a user that already stores it in its cache, i.e., $k \notin \mathcal{S}$.

Note that a transmission assignment, $\mathcal{T}[n]$, also uniquely defines the precoding vectors. Each precoding vector, $\mathbf{f}_{i,\mathcal{S}}[n]$ is chosen in the unique direction that is orthogonal to the $L-1$ users that are active and not in the set $\mathcal{S}$, i.e., if $i = d_k$, we have $\mathbf{h}_\ell \mathbf{f}_{i,\mathcal{S}}[n] = 0$ for any $\ell \in \mathcal{K}(\mathcal{T}[n]) \backslash \{k \cup \mathcal{S}\}$.

To design and analyze the network, we do not need to choose the actual assignment of symbols for each transmission. It suffices to characterize the number of symbols sent using each assignment type. Denote by $n_\mathcal{T}$ the number of symbols dedicated to a specific transmission assignment type, i.e., $n_\mathcal{T} = |\{n : \mathcal{T}[n] = \mathcal{T}\}|$. The problem can be formulated as an optimization problem on the variables $\{n_\mathcal{T}\}$ for all valid $\mathcal{T}$. The optimization goal is to minimize the transmission time, i.e., $\sum_\mathcal{T} n_\mathcal{T}$, subject to the decodability of all the requested segments. This constraint can now be simply stated as: For each user, k, and each set $\mathcal{S}$ with $|\mathcal{S}| = \alpha$ and $k \notin \mathcal{S}$, we require that the number of received symbols will be the number of coded symbols, i.e.,

$$\sum_{\mathcal{T}:(d_k,\mathcal{S}) \in \mathcal{T}} n_\mathcal{T} = |\tilde{\mathcal{W}}_{d_k,\mathcal{S}}|. \tag{12}$$

An example of the optimal cache placement and transmission scheduling in the homogeneous case is given in "A Multiple-Input, Single-Output Cache-Aided Communication System," while an example of nonhomogeneous rates is given in "A Multiple-Input, Single-Output Cache-Aided Communication Systems With Heterogeneous Rates."

Posing the question as an optimization problem allows us to find the best transmission scheduling for each network. The resulting optimization problem is linear, and we may utilize a variety of algorithms for efficient solution. Note that the formulation herein focuses only on the optimization of the worst-case scenario, in which all users request different files. A more detailed formulation can lead to further improvement in cases where several users request the same file.

Keep in mind that the optimization approach rarely leads to closed-form performance expressions. For example, for the case of arbitrary user rates, the only case with such a closed-form expression is the case that $K = L + \alpha$, with integer α. This is a limited case, as it typically requires unreasonably large users' cache. Nevertheless, it is interesting because it shows that for many user rate combinations, all $L + \alpha$ users can be simultaneously served using L antennas, where each user receives information at its own rate. It was shown [10] that a minimal time of

$$T = \frac{KF(1 - M/N)}{\sum_{k=1}^{K} R_k}, \tag{13}$$

is achievable if the user rates satisfy $R_k \le (1/L)\sum_{\ell=1}^{K} R_\ell$ for every k in $\{1, 2, \ldots, K\}$.

For the completeness of the network description, we next show that the presented scheme is indeed feasible and achieves a DoF of $L + KM/N$. In the next section, we present a transmission scheduling scheme for the simple homogeneous case.

Performance in the homogeneous case

In the homogeneous case, all files are of equal size and popularity, and all users have the same rate. In this case, we show that all users can receive their desired files while $L + \alpha$ users are served at any given time with no interference. We note that this also represents the high SNR regime, and hence proves that the system achieves the expected DoF as claimed above. Again, we focus on the worst-case scenario, where all the users request different files.

In the homogeneous case, it is reasonable to adopt the cache allocation of the centralized MAN scheme described in (1). This cache allocation divides each file into $\binom{K}{\alpha}$ equal parts denoted by $\mathcal{W}_{k,\mathcal{S}}$, where $|\mathcal{S}| = \alpha$ and the set index, $\mathcal{S}$, indicates the set of users that store $\mathcal{W}_{k,\mathcal{S}}$ in their cache, i.e., $\mathcal{W}_{k,\mathcal{S}} \in \mathcal{Z}_i$ if and only if $i \in \mathcal{S}$. This cache allocation is useful in many cases and, in particular, when the network is symmetric.

The delivery phase in the homogeneous case is solved by simply setting the number of symbols in each valid transmission assignment type to be identical. Inspecting the conditions for a valid transmission assignment in the previous section, we note that there are $\binom{K}{L+\alpha}$ possible choices for the set of $L + \alpha$ active users in the assignment. Given the active users, the transmission to each user, k, is characterized by a pair $(d_k, \mathcal{S})$, where $\mathcal{S}$ is selected from the other active users; therefore, there are $\binom{L+\alpha-1}{\alpha}^{L+\alpha}$ possible assignments for each set of active users. Thus, in total, we have $Q = \binom{K}{L+\alpha} \cdot \binom{L+\alpha-1}{\alpha}^{L+\alpha}$

possible transmission assignment types, and we assign an identical number of symbols to each: i.e., $n_{\mathcal{T}} = c$.

Each transmission assignment includes $L+\alpha$ segments, and each specific segment $\mathcal{W}_{d_k,\mathcal{S}}$ appears in $(L+\alpha)/\left(K\cdot\binom{K-1}{\alpha}\right)$ of the transmission assignments, i.e., we have a total of $J = Q\cdot(L+\alpha)/\left(K\cdot\binom{K-1}{\alpha}\right)$ assignments. To decode each segment, we need to receive all of its symbols, i.e., we need $c = |\tilde{\mathcal{W}}_{d_k,\mathcal{S}}|/J$. Knowing c, and noting that the

A Multiple-Input, Single-Output Cache-Aided Communication System

Consider a cache-aided network with $K=3$ users and $L=2$ antennas at the transmitter. The channel from the base station (BS) to user k is denoted by $\mathbf{h}_k$, and we assume that all the users can decode at a rate of R bits/s. Each user has a memory of size of $M/N = 1/3$, and thus, can prefetch only one-third of each file. We first partition each of the files into $\binom{K}{MK/N} = \binom{3}{1} = 3$ segments, and label them as $\mathcal{W}_{i,\{1\}}, \mathcal{W}_{i,\{2\}}$, and $\mathcal{W}_{i,\{3\}}$, respectively, for every $i\in[N]$, as shown in Figure S3(a). Then, each user, k, prefetches segments $\mathcal{W}_{i,\{k\}}$ for all $i\in[N]$. The resulting placement is presented in Figure S3(b). Assuming for simplicity that user k requests the file with an index of $d_k = k$, i.e., $d_1 = 1$, $d_2 = 2$, and $d_3 = 3$. Note that, each user has one segment of its desired file in the cache, and the two remaining segments should be sent during the delivery phase.

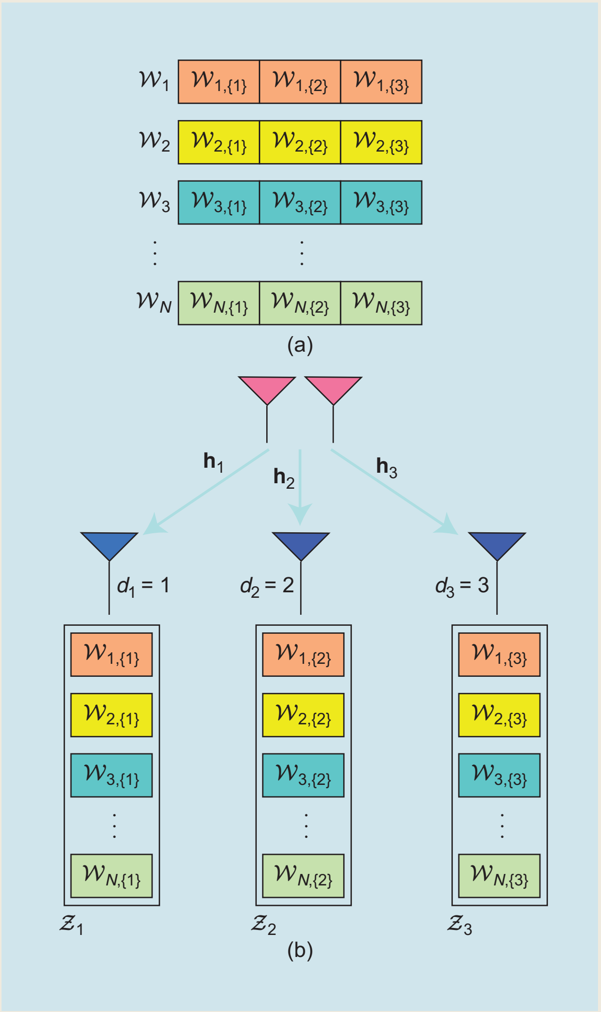


FIGURE S3. A multiple-input, single-output cache-aided communication system with $L = 2$ transmit antennas, $K = 3$ users, and normalized cache size of $M/N = 1/3$. The (a) file partitioning and (b) cache placement.

The transmitter broadcast message can be represented (intuitively) as

$$\begin{aligned}\mathbf{x}(1) &= \mathbf{h}_3^{\perp}\tilde{\mathcal{W}}_{1,\{2\}} + \mathbf{h}_1^{\perp}\tilde{\mathcal{W}}_{2,\{3\}} + \mathbf{h}_2^{\perp}\tilde{\mathcal{W}}_{3,\{1\}},\\ \mathbf{x}(2) &= \mathbf{h}_2^{\perp}\tilde{\mathcal{W}}_{1,\{3\}} + \mathbf{h}_3^{\perp}\tilde{\mathcal{W}}_{2,\{1\}} + \mathbf{h}_1^{\perp}\tilde{\mathcal{W}}_{3,\{2\}},\end{aligned} \tag{S2}$$

where $\tilde{\mathcal{W}}_{i,\mathcal{S}}$ denotes the digitally modulated sequence of symbols corresponding to the file segment $\mathcal{W}_{i,\mathcal{S}}$. Also note that the precoder vector $\mathbf{h}_k^{\perp}$ indicates that the beamformed signal is perpendicular to the channel of user k and will be zero forced at this user. The exact mathematical representation of the transmitted signals is given in (7).

Let us study the signal received by user 1. During the first time slot, user 1 receives

$$y_1(1) = \mathbf{h}_1\mathbf{x}(1) + z_1(1) = \mathbf{h}_1\mathbf{h}_3^{\perp}\tilde{\mathcal{W}}_{1,\{2\}} + \mathbf{h}_1\mathbf{h}_2^{\perp}\tilde{\mathcal{W}}_{3,\{1\}} + z_1(1),$$

where $z_k(t)$ is the additive white Gaussian noise observed at user k in time slot t. Note that one interference term is zero forced, i.e., $\mathbf{h}_1\mathbf{h}_1^{\perp}\tilde{\mathcal{W}}_{2,\{3\}} = 0$. Moreover, $y_1(1)$ is a combination of the desired codeword $\tilde{\mathcal{W}}_{1,\{2\}}$ and another interfering codeword $\tilde{\mathcal{W}}_{3,\{1\}}$. However, this interfering codeword can be reconstructed from the segment $\mathcal{W}_{3,\{1\}}$ stored in $\mathcal{Z}_1$. Once the interference is subtracted, user 1 can decode segment $W_{1,\{2\}}$. A similar argument holds for all the users and all the time slots, where each user can decode one desired segment at each time slot.

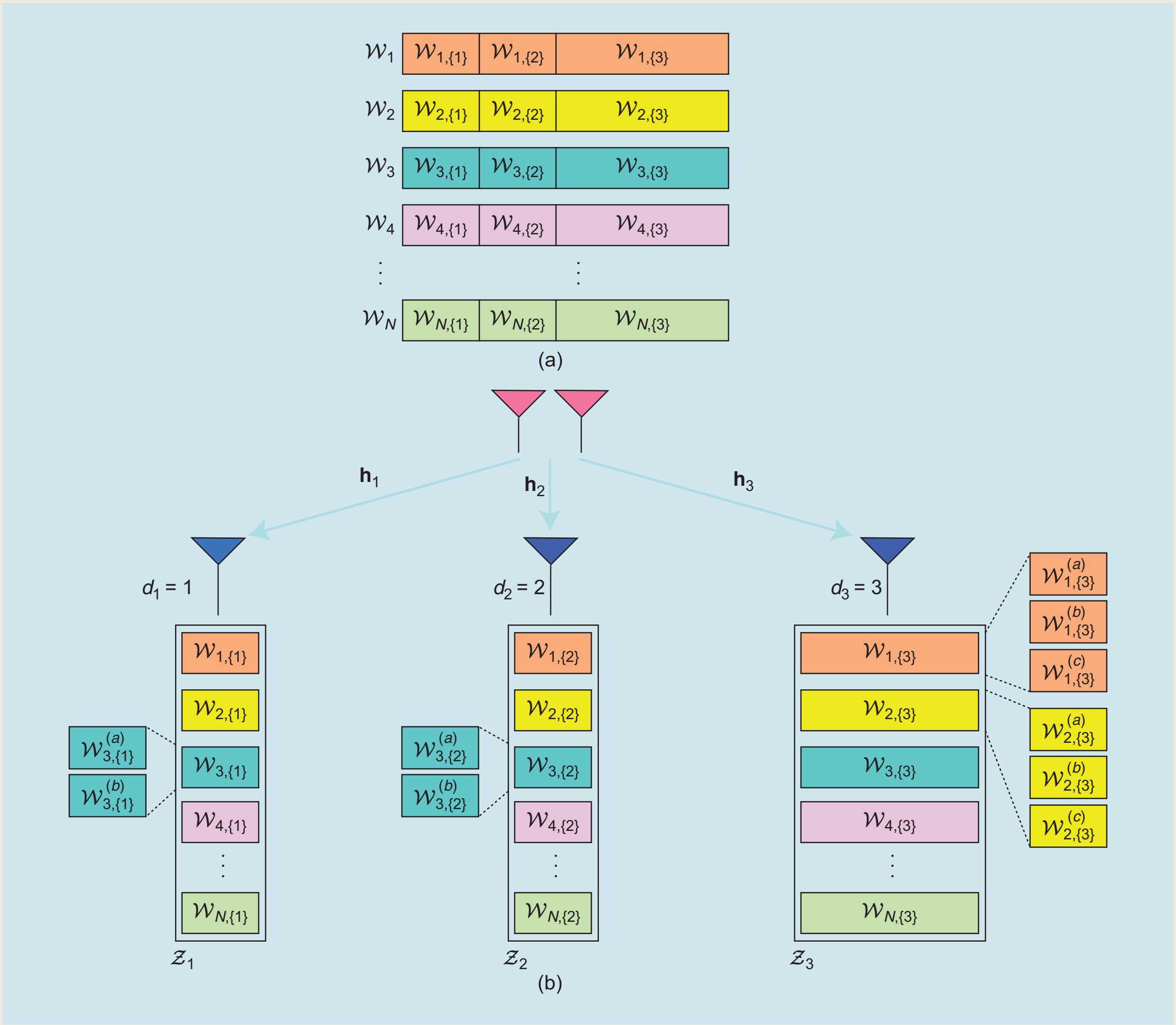
Note that each transmission can simultaneously serve $L + KM/N = 2 + 1 = 3$ users. This is due to the possibility of interference cancellation using the cache content at $KM/N = 1$ user and using zero forcing at $L - 1 = 1$ user. In each time slot the BS is broadcasting a segment (combination) of length one-third of the size of a file, at rate R (which is decodable for all users). Thus, the duration of each transmission will only be $F/3R$ seconds, where F is the file size. This leads to a total transmission time of $2F/3R$ seconds. During this time, three files are delivered to the users, and therefore, we have an overall network throughput of $3F/(2F/3R) = 4.5R$ bits/s.

A Multiple-Input, Single-Output Cache-Aided Communication Systems With Heterogeneous Rates

Consider a multiple-input, single-output (MISO) broadcast channel with $L = 2$ transmit antennas and $K = 3$ users with single antenna, as shown in Figure S3. We assume a total cache constraint so that only one copy of each (packet of each) file can be prefetched among all of the users, i.e., $\sum_{k=1}^{3} |\mathcal{Z}_k| = NF$. We denote the link capacity of user k by R_k and assume a heterogeneous topology, in which users 1 and 2 have good channels to support $R_1 = R_2 = 2$ bits/s, while the third user is further away from the transmitter and can decode only at rate $R_3 = 1$ bits/s. We consider a placement strategy that is symmetric across files but not necessarily symmetric across users. It is natural to expect that a larger cache will be allocated to the weak user to minimize the overall transmission time of the delivery phase.

It turns out that the optimal cache allocation places $M_1 F/N = M_2 F/N = F/5$ bits in the cache of users 1 and 2, while user 3 stores $M_3 F/N = 3F/5$ bits to compensate for its weakness. The cache allocation is accomplished by partitioning each file into three segments, i.e., $\mathcal{W}_{i,\{1\}}, \mathcal{W}_{i,\{2\}}$, and $\mathcal{W}_{i,\{3\}}$, which are stored at the cache of users 1, 2, and 3, respectively, as depicted in Figure S4(b). Note that the cache placement is performed prior to the user's request and is identical for all files. In this example, we assume that user k requested $d_k = k$ for $k \in \{1, 2, 3\}$.

The delivery phase includes the broadcasting of four messages. To present the broadcast messages, we must further divide the cached messages into smaller segments as $\mathcal{W}_{3,\{1\}} = \left(\mathcal{W}^{(a)}_{3,\{1\}}, \mathcal{W}^{(b)}_{3,\{1\}}\right)$, $\mathcal{W}_{3,\{2\}} = \left(\mathcal{W}^{(a)}_{3,\{2\}}, \mathcal{W}^{(b)}_{3,\{2\}}\right)$, $\mathcal{W}_{1,\{3\}} = \left(\mathcal{W}^{(a)}_{1,\{3\}}, \mathcal{W}^{(b)}_{1,\{3\}}, \mathcal{W}^{(c)}_{1,\{3\}}\right)$, and $\mathcal{W}_{2,\{3\}} = \left(\mathcal{W}^{(a)}_{2,\{3\}},\right.$



$\mathcal{W}_1$ $\mathcal{W}_{1,\{1\}}$ $\mathcal{W}_{1,\{2\}}$ $\mathcal{W}_{1,\{3\}}$
$\mathcal{W}_2$ $\mathcal{W}_{2,\{1\}}$ $\mathcal{W}_{2,\{2\}}$ $\mathcal{W}_{2,\{3\}}$
$\mathcal{W}_3$ $\mathcal{W}_{3,\{1\}}$ $\mathcal{W}_{3,\{2\}}$ $\mathcal{W}_{3,\{3\}}$
$\mathcal{W}_4$ $\mathcal{W}_{4,\{1\}}$ $\mathcal{W}_{4,\{2\}}$ $\mathcal{W}_{4,\{3\}}$
$\mathcal{W}_N$ $\mathcal{W}_{N,\{1\}}$ $\mathcal{W}_{N,\{2\}}$ $\mathcal{W}_{N,\{3\}}$
(a)
$\mathbf{h}_1$
$\mathbf{h}_2$
$\mathbf{h}_3$
$d_1 = 1$
$d_2 = 2$
$d_3 = 3$
$\mathcal{W}^{(a)}_{3,\{1\}}$ $\mathcal{W}^{(b)}_{3,\{1\}}$
$\mathcal{W}^{(a)}_{3,\{2\}}$ $\mathcal{W}^{(b)}_{3,\{2\}}$
$\mathcal{W}^{(a)}_{1,\{3\}}$ $\mathcal{W}^{(b)}_{1,\{3\}}$ $\mathcal{W}^{(c)}_{1,\{3\}}$
$\mathcal{W}^{(a)}_{2,\{3\}}$ $\mathcal{W}^{(b)}_{2,\{3\}}$ $\mathcal{W}^{(c)}_{2,\{3\}}$
$\mathcal{Z}_1$
$\mathcal{Z}_2$
$\mathcal{Z}_3$
(b)

FIGURE S4. A cache-aided communication with $L = 2$ transmit antennas. The (a) file partitioning and (b) cache placement.

A Multiple-Input, Single-Output Cache-Aided Communication Systems With Heterogeneous Rates (*Continued*)

$\mathcal{W}^{(b)}_{2,\{3\}}, \mathcal{W}^{(c)}_{2,\{3\}})$ to keep up with the capacity of the links to the users. All three users can be served in four time slots by transmitting

$$\begin{aligned}\mathbf{x}(1) &= \mathbf{h}_2^{\perp}\tilde{\mathcal{W}}^{(a)}_{1,\{3\}} + \mathbf{h}_1^{\perp}\tilde{\mathcal{W}}^{(a)}_{2,\{3\}} + \mathbf{h}_2^{\perp}\tilde{\mathcal{W}}^{(a)}_{3,\{1\}}\\ \mathbf{x}(2) &= \mathbf{h}_3^{\perp}\tilde{\mathcal{W}}_{1,\{2\}} + \mathbf{h}_1^{\perp}\tilde{\mathcal{W}}^{(b)}_{2,\{3\}} + \mathbf{h}_2^{\perp}\tilde{\mathcal{W}}^{(b)}_{3,\{1\}}\\ \mathbf{x}(3) &= \mathbf{h}_2^{\perp}\tilde{\mathcal{W}}^{(b)}_{1,\{3\}} + \mathbf{h}_3^{\perp}\tilde{\mathcal{W}}_{2,\{1\}} + \mathbf{h}_1^{\perp}\tilde{\mathcal{W}}^{(a)}_{3,\{2\}}\\ \mathbf{x}(4) &= \mathbf{h}_2^{\perp}\tilde{\mathcal{W}}^{(c)}_{1,\{3\}} + \mathbf{h}_1^{\perp}\tilde{\mathcal{W}}^{(c)}_{2,\{3\}} + \mathbf{h}_1^{\perp}\tilde{\mathcal{W}}^{(b)}_{3,\{2\}},\end{aligned} \qquad \text{(S3)}$$

where $\mathbf{x}(\ell)$ is the collection of all transmission vectors for the ℓth time slot and $\tilde{\mathcal{W}}_{i,\mathcal{S}}$ denotes the digitally modulated sequence of symbols corresponding to file segment $\mathcal{W}_{i,\mathcal{S}}$. Moreover, beamforming a codeword with precoder vector $\mathbf{h}_k^{\perp}$ guarantees that the received signal corresponding to that codeword will be zero forced at user k. The exact mathematical representation of the transmitted signals is given in (7).

Let us consider the decoding at user 1. For instance, during time slot 1, user 1 receives

$$y_1(1) = \mathbf{h}_1\mathbf{x}(1) + z_1(1) = \mathbf{h}_1\mathbf{h}_2^{\perp}\tilde{\mathcal{W}}^{(a)}_{1,\{3\}} + \mathbf{h}_1\mathbf{h}_2^{\perp}\tilde{\mathcal{W}}^{(a)}_{3,\{1\}} + z_1(1).$$

The user subtracts codeword $\tilde{\mathcal{W}}^{(a)}_{3,\{1\}}$ using its cache content and then uses the remaining signal to decode $\mathcal{W}^{(a)}_{1,\{3\}}$. Similarly, each user can decode all of the missing segments of its requested file.

Note that each transmission time takes $F/10$ seconds, and the total transmission time is $0.4F$ seconds, after which, all users can decode their requested files. Because three users are served during this time, the overall network throughput is $3F/0.4F = 7.5$ bits/s. In contrast, with equal cache placement, the minimal time to serve user 3 is $\frac{2}{3}\cdot\frac{F}{R_3}$ seconds, which leads to a throughput of only 4.5 bits/s. This shows that an optimized coded caching in MISO brings additional gain by balancing the load of the network.

number of coded symbols in a segment is $(F/R)/\binom{K}{\alpha}$, the total number of symbols required to serve all users is

$$T = c\cdot Q = \frac{F/R}{\binom{K}{\alpha}}\cdot\frac{K\cdot\binom{K-1}{\alpha}}{L+\alpha} = \frac{K(1-M/N)}{L+\alpha}\cdot\frac{F}{R}. \quad (14)$$

Here, KF is the total number of bits in the files requested by all the users. Compared to (5), we see that the number of DoF given by

$$\text{DoF} = L + \alpha = L + KM/N, \quad (15)$$

where we see the combination of KM/N DoF that are achieved through cache-aided communication and L DoF of spatial multiplexing using L antennas.

Related problems

Demand prediction

In some limited systems, the number of files in the database is relatively small, hence the notation in the previous section seems reasonable. However, in most cases, and in particular if we consider cellular systems of 5G and beyond, the database size can be huge. As can be verified from the performance measures in (14) and (15), a huge database size, N, can completely diminish the gain of caching.

Accordingly, cache-aided communications in large systems should cache only a subset of the files in the database, preferably the most popular ones [26]. Yet, the popularity of files changes over time, and a proper system operation would require a continuous update of the most popular sets [27].

Even more challenging is the prediction of the popularity of the files. This problem has already attracted much attention, although, not necessarily in the context of cache-aided communications. Because the problem is outside the scope of this article, we only refer the interested reader to [19] and [28] as two starting points where this problem in the context of caching is discussed. These highlight the importance of popularity prediction and cache updates. They also emphasize the effectiveness of machine-learning techniques, as the actual distribution of file popularity is typically unknown.

It is worth pointing out that these types of works are more related to web caching and content distribution networks or, in the wireless framework, on scenarios known as *femtocaching*. In coded caching, there is typically a substantial separation between the time scale at which users' requests are served and the time scale at which the popularity of the content evolves. As an example, the collection may include the most popular 1,000 titles from the Netflix library, and every week some new movies are included and some old ones are eliminated (while a streaming session is on the order of tens of minutes). In this context, demand prediction is virtually orthogonal and complementary with coded caching. One can use a standard scheme (e.g., as done by Netflix) to track what users want at the large time scale and use coded caching for the delivery of files from the current library.

Resource allocation

Resource allocation is one of the most important tasks in the optimization of any communication system. In cache-aided communication systems, traditional resource allocation problems become more challenging and new problems need to be addressed. In this section, we discuss the power allocation as an example of a traditional resource allocation problem that becomes more complicated, and then we discuss the cache allocation as an example of a new allocation problem.

Note that any resource allocation problem is strongly related to the transmission scheduling problem. The transmission scheduling is, by itself, a resource allocation problem that determines which segments will be allocated to each transmission time. So, allocating any additional resources can only be done in coordination with transmission scheduling.

Power allocation

The power allocation problem requires finding the optimal transmission energy for each symbol, which minimizes the time required to serve all the users. The main difficulty in the optimization of the power allocation is the need for a closed-form performance expression. Such an expression, which can be written as a function of the transmission powers, currently exists only for the limited case where the number of users is exactly $K = L + KM/N$. As a result, this is also the only cache-aided communication scenario where the optimal power allocation is known [10], [29].

This power allocation is shown to be a variation of the water-filling algorithm with a rate saturation. As the cache-aided communication cannot take advantage of users with a very high rate, the rate of such users is saturated and the extra power is used for other users. Optimal power allocation for other cache-aided communication scenarios still remains an open problem.

Cache allocation

Cache allocation includes 1) allocating the memory (cache size) for each user and 2) allocating the cache content. The allocation of cache content means the choice of sets $\mathcal{W}_{i,\mathcal{S}}$ for each i and $\mathcal{S}$, usually under the constraint of user cache size of MF bits.

For example, for nonuniform file popularity, it is likely that popular files will be stored more often than others. Thus, we may have $|\mathcal{W}_{i,\mathcal{S}}| \neq |\mathcal{W}_{j,\mathcal{S}}|$ when files i and j have different popularity. This has mostly been studied for a single-antenna BS and formulated as linear optimization problem [30]. It was shown that the uniform placement of the centralized MAN scheme [1] is indeed optimal when the file popularity is uniform. However, in the case of nonuniform popularity, the optimal placement performs better than the uniform.

Contrarily, in the case of uniform file popularity, while the placement is still symmetric across the files, it is not necessarily symmetric across users. That is, if the BS decides that user k stores bit number n, then it will store this bit from all files in the database. So, we have $|\mathcal{W}_{i,\mathcal{S}}| = |\mathcal{W}_{j,\mathcal{S}}|$ for all i and j, but not necessarily $|\mathcal{W}_{i,\mathcal{S}_1}| = |\mathcal{W}_{i,\mathcal{S}_2}|$.

For such a case, we observe that the cache placement optimization is much simpler for $|\mathcal{S}| = KM/N = 1$. In this case, there is no meaning to the question "Which specific bit is stored at which user?" Hence, the only cache allocation decision that has an effect on the performance is the users' cache sizes. Conversely, for $KM/N > 1$, there will be overlap between users' caches (i.e., a bit may be stored at more than one user). The choice of which users' caches will overlap can have an effect on performance in a nonhomogenous network.

Another improvement can be gained in networks that allow for an optimization of the cache size per user depending on the channel qualities. Recall that cache allocation occurs during the placement phase, and in most scenarios, at this stage the system has no knowledge of the future states of the channels. The typical approach employed is to use an equal cache size for all users.

In some cases, the users' delivery rates can be predicted. Two examples of this are 1) in fixed wireless networks, where the rates are rather fixed or 2) in low-orbit satellite communications, where the rates change faster but more predictably because satellites travel in known orbits. In such cases, the cache size at each user can be adjusted to partly compensate for the different rates and increase the network throughput (an illustrative example is discussed in "A Multiple-Input, Single-Output Cache-Aided Communication Systems With Heterogeneous Rates"). In the case of multiantennas, this problem was framed as a linear optimization problem, albeit with a number of variables that grows exponentially with the number of users [10].

Caching for scalable coding

The cache-aided communication problem is usually studied in the strict fairness setup. The typical problem formulation is of equal size files, and each user requests a single file. Thus, the objective is to supply to each user the same amount of data, regardless of the channel qualities. The strict fairness constraint makes the network very sensitive to the performance of the weakest user. This is in contrast to other network-optimization works, which typically focus on the total network throughput.

An interesting scenario that challenges this fairness assumption is scalable coding. With scalable coding, the same content (e.g., a movie) can be compressed at divergent rates to produce distinct types of quality for a variety of users. Reproducing lower-quality versions by using lower data rates is desired, e.g., for diverse equipment types (e.g., different screen sizes), dissimilar channel conditions, or a range of content pricing. Because of this, the same file may be requested in the network at several quality levels. Scalable coding allows for the encoding of the file at several layers. Users that decode only the first layer will reconstruct the content at the lowest quality (highest distortion), while users that decode multiple layers will be able to refine the reconstruction and realize higher quality.

Having several layers for each file makes cache allocation much more interesting [31]. In particular, if the desired quality of each user is known in advance, both the cache allocation and placement can be optimized for the specific desired quality of each user. Obviously, such an optimization results in a better utilization of the cache and higher network throughput.

Effect of transmitter cache

In a network with multiple single-antenna BSs, cache can be used to facilitate cooperation between the BSs. This was studied in a network where all BSs are synchronized and make joint scheduling decisions but do not store the complete database. In such a case, each bit can be transmitted only from the BSs that store them in their cache. Thus, a ZF precoder that nulls the interference caused by the transmission of a symbol can only be used in these BSs.

Denoting the cache size at each BS with $M_{\mathrm{T}}F$ and assuming that LM_{T}/N is an integer (where L is the number of BSs), each bit is accessible to LM_{T}/N antennas. Each transmission can therefore only be zero forced at $LM_{\mathrm{T}}/N - 1$ users. Combined with the gain of the receivers cache, the number of users that can be served simultaneously (hence, the DoF of the system) [32] is

$$\mathrm{DoF} = \frac{LM_{\mathrm{T}} + KM}{N}. \tag{16}$$

Note that if each BS has access to the whole database, complete cooperation can be established between all BSs, and the performance will be characterized by the single BS case that is equipped with L antennas. In this case, we have $M_{\mathrm{T}} = N$, and (16) simplifies to (15).

It is important to emphasize that (16) gives an achievability result, and the maximal performance is not yet known. For example, in the specific case of $L = 3$ and $M_T/N = 2/3$ and $M = 0$, it was shown [32] that when using interference alignment the DoF is 18/7, which is significantly higher than the DoF of 2 that is achievable according to (16). The best-known general upper bound on the DoF of linear precoding (also termed *one-shot coding*) [32] is two times the DoF given in (16).

Multiserver wireline network

Surprisingly, a similar channel model can be found in wireline multiserver networks. Using the concept of network coding, in some scenarios it is beneficial for each server in the network to forward a linear combination of its incoming packets. This linear combination is performed over a large-enough finite field so that the network is invertible. The operation of the whole network can then be represented by a matrix.

A central network manager controls multiple servers, which need to jointly serve multiple users. These multiple servers act as antennas in the wireless model presented above. Accordingly, the output at each user is described as a multiplication of a transmitted vector and a channel matrix, just as in (6), with the only differences being the operations over a large finite field instead of over the field of complex numbers [8] and the absence of noise. The same transmission schemes are applicable and result in the same performance gain.

The subpacketization problem and the role of spatial multiplexing

For the single shared-link network presented in [1], the major problem of the applicability of coded caching schemes to real-world systems is represented by the very large subpacketization order, i.e., the number of subpackets (segments) that each file $\mathcal{W}_i$ in the library must be partitioned into. In the MAN scheme with K users, N files, and cache memory per user MF bits, assuming that $\alpha = KM/N$ is an integer, each file must be split into $\binom{K}{\alpha}$ subpackets. Letting $\mu = M/N \in [0,1]$ denote the fractional memory level, i.e., the fraction of the whole library that each user can cache, the subpacketization order of MAN is given by $\binom{K}{K\mu} \approx \exp(K\mathcal{H}(\mu))$, where $\mathcal{H}(\mu) = -\mu\log\mu - (1-\mu)\log(1-\mu)$ is the binary entropy function. Therefore, the subpacketization order is exponential in K for a given fractional memory level μ.

Denoting the length of each file by F bits, it is clear that F must be at least as large as the subpacketization order, i.e., the file size required by MAN is also $O(\exp(K))$. (In practice, the file size should be even larger as the transmission always processes a packet of many bits at a time.) For example, a system with $K = 500$ users where each user caches a fraction $\mu = 0.01$ of the library, would require a minimum file size in bits $F \approx 2.5 \cdot 10^{11}$. This means that the required file size is at least 250 gigabits. Taking into account that the typical size of a movie encoded in standard definition is of the order of 1 gigabits, we see that the file size required by such a scheme is between 1 and 2 orders of magnitude larger, and therefore, completely impractical.

The subpacketization order problem is exacerbated by the fact that, in a practical media streaming system, a streaming session consists of a sequence of demands of video "chunks," corresponding to a few seconds of video. To cope with asynchronous streaming sessions, each video chunk of each video file should be treated as a "file" $\mathcal{W}_i$ in the formalism of MAN, presented in this section [1]. It follows that, in practical video steaming applications, the actual size of the video chunks, F, is of an order of a few megabytes, which is four or five orders of magnitude smaller than that required file size in the example above.

Furthermore, it was proved in [33] that any decentralized caching scheme based on symmetric random caching, i.e., where each user caches a fraction μ of the bits of each file selected at random with uniform probability, independently of the other users, must have F that grows superexponentially in K to achieve a DoF $= O(K)$. For example, the load expression of the decentralized MAN scheme in [3] [see (22)] is valid only in the limit of $F \to \infty$ and fixed K. In contrast, in [33] it is shown that if F grows less than superexponentially in K, then the maximum achievable DoF does not exceed 2.

Cache replication

Several methods have been proposed to cope with the subpacketization order problem (e.g., see [34]–[39]). These methods can be (roughly) classified into optimization-based (e.g., [34]–[36]) and graph-theoretic/combinatorial methods (e.g., [37] and [38]). Unfortunately, they are typically quite complicated and not flexible in terms of system parameters.

A much simpler approach consists of cache replication. For the centralized case, we create a MAN packetization for a nominal number of users G, such that the subpacketization order $\approx \exp(G\mathcal{H}(\mu))$ is kept to a reasonable value. For example, for $G = 100$ and $\mu = 0.01$ we have $F = 100$. Then, we divide the user population in G groups of K/G users each (assume for simplicity's sake that K/G is an integer). Users in the same group $g = 1, \ldots, G$ cache exactly the same packets, i.e., the gth cache configuration is replicated across all K/G users in each group g. In the delivery phase, a delivery array of dimension $G \times K/G$ is formed by arranging the requests of the users in the same group by rows. Hence, the system delivers the requests by serving sequentially the columns of the delivery array. Note that each column forms a MAN scheme with G users because, by construction, the users in each column belong to different groups. It follows that all requests can be delivered with a load equal to the load of a G-user MAN scheme, given by $G(1-\mu)/(1+\mu G)$ times the number of columns K/G. This yields

$$D = \frac{K}{G} \cdot \frac{G(1-\mu)}{1+\mu G}. \tag{17}$$

By letting G be a function of K, this scheme achieves a very desirable tradeoff between subpacketization order and DoF.

A decentralized version of the cache replication approach is proposed and analyzed in [39] (see also [40]). In this case, users simply choose one of the g groups (and the corresponding cache content) randomly and independently. We refer to the number of users in group g as the gth *occupancy number*, denoted as ℓ_g. The vector of occupancy numbers $(\ell_1, \ldots, \ell_G)$ is random and follows a multinomial distribution over all possible G-way integer partitions $(\eta_1, \ldots, \eta_G)$ of K. It follows that the delivery for this problem, for given occupancy numbers, is identical to the shared caches network studied in [41], for which an optimal delivery under symmetric caching (as enforced by the cache replication construction) consists of sorting the occupancy numbers $\ell_{(1)} \geq \cdots \geq \ell_{(G)}$ in a nonincreasing order, such that $\ell_{(g)}$ denotes the size of the gth most populated group, and forms a delivery array of size $G \times \ell_{(1)}$. Such an array has empty elements because, in general, groups have fewer than $\ell_{(1)} = \max\{\ell_g\}$ users. Then, each column of the array is served by an improved MAN scheme that avoids sending XORs when they are not useful for at least one user (in fact, each column is served by the improved delivery scheme of [42]). The resulting load for $\{\ell_{(g)}\}$ is given by

$$D(\ell_1, \ldots, \ell_G) = \sum_{g=1}^{G(1-\mu)} \ell_{(g)} \frac{\binom{G-g}{\mu G}}{\binom{G}{\mu G}}, \tag{18}$$

Here, for simplicity's sake, we assume that μG is an integer. If not, the usual memory-sharing argument holds and the convex envelope of the load/memory points for $\mu G \in \{0, 1, 2, \ldots, G\}$ is achievable.

It turns out that this load is information-theoretically optimal for any given configuration of the occupancy numbers [41], in the case of distinct demands (which requires $N \geq K$) and uncoded placement.

As a rough upper bound for the average load, where the averaging is with respect to the occupancy numbers, we have (trivially)

$$\mathbb{E}[D(\ell_1, \ldots, \ell_G)] \leq \mathbb{E}[\ell_{(1)}] \cdot \frac{G(1-\mu)}{1+\mu G}. \tag{19}$$

By comparing (17) and (19), we note that the difference is generally small. In fact, $\mathbb{E}[\ell_{(1)}]$ is the expectation of the maximum of G multinomial variables. Because each occupancy number ℓ_g is marginally binomially distributed with parameters $(K, 1/G)$, its expected value is $\mathbb{E}[\ell_g] = K/G$. Now, although the ordered statistics of a jointly multinomial random vector are generally difficult to characterize, $\mathbb{E}[\ell_{(1)}]$ behaves as K/G up to the logarithmic factors in G. Therefore, the average load $\mathbb{E}[D(\ell_1, \ldots, \ell_G)]$ of the decentralized scheme has essentially the same behavior as that of the centralized scheme in (17). A more refined analysis is given in [40].

Exploiting spatial multiplexing

In the previous section, we explained how cache replication provides a viable approach for achieving a competitive tradeoff between DoF and subpacketization order. However, the fact that the delivery array columns are served one by one in sequence prevents such schemes from reaching low subpacketization order with the same (ideal) DoF of the MAN scheme for the single shared-link network. Intuitively, if we can "parallelize" the different columns of the delivery array using spatial multiplexing, we should be able to achieve both low subpacketization order and optimal DoF at the same time.

Following this idea, we revisit the MISO broadcast channel with caches at the receivers, previously discussed in the "Signal Processing Problem Formulation" section, and present a different scheme achieving the low subpacketization proposed in [43] and the same optimal DoF of the ZF precoding scheme in the "Signal Processing Problem Formulation" section. Consider a MISO broadcast channel with K users, a BS with L antennas, a library of N files of size F where each user has cache memory MF bits, yielding fractional memory level $\mu = M/N$, such that μK is an integer. For the sake of simplicity, assume that L divides both K and μK. We partition the users into $G = K/L$ groups of L users each and use the cache replication approach. The cache placement consists of cache replication as explained previously: create a MAN subpacketization with parameters (N, M, G, and F), and let all users in group g to cache the same content. Note that this subpacketization consists of $\binom{G}{\mu G}$ subpackets. Because $G = K/L$, the subpacketization order of this scheme is $\approx \exp(K/L\mathcal{H}(\mu)) = \sqrt[L]{\exp(K\mathcal{H}(\mu))}$, the Lth root of the subpacketization order of the "classical" scheme presented in the "System Model and Problem Formulation" section. Referring to a previous example, for $K = 500$,

$\mu = 0.01$, and $L = 8$ antennas, we find $F \approx 27$ in contrast to the case of $L = 1$ for which $F \approx 2.5 \cdot 10^{11}$.

The delivery consists of simultaneously serving the L sections of the delivery array by combining a K/L-user MAN scheme with the L-fold spatial multiplexing obtained by ZF MIMO precoding. Figure 2 qualitatively shows the delivery array of dimension $K/L \times L$, where each column is formed by users belonging to distinct caching groups. Therefore, each column can be served using a K/L-user MAN scheme; however, unlike for the single shared-link network, here, the inherent spatial multiplexing of the MIMO channel can be used to serve all the L slices simultaneously.

Due to space limitations, we omit the details of the combined coded caching and MIMO precoding scheme, which can be found in [43]. The important point to notice here is that, although in the basic cache replication scheme for the single shared-link network we have to serve the sections of the delivery array in sequence [see (18)], in the MIMO case, by exploiting spatial precoding, we can serve up to L sections simultaneously. In general, each user receives interference from other users in the same section and from users in different sections. If the number of interfering sections is not larger than the number of antennas L, then this second type of interference (intergroup interference) can be zero forced by MIMO precoding, while the first type of interference (intragroup interference) is handled in the usual coded caching way: it can be canceled at the receiver because each user has all of the interfering packets in its cache, except the one it needs to decode.

Because the L sections of the delivery array are served simultaneously (by spatial multiplexing), the load is the same as that of a single MAN scheme (N, M, G, and F). Therefore,

$$D = \frac{G(1-\mu)}{1+\mu G} = \frac{K/L(1-\mu)}{1+\mu K/L} = \frac{K(1-\mu)}{L+\mu K}. \tag{20}$$

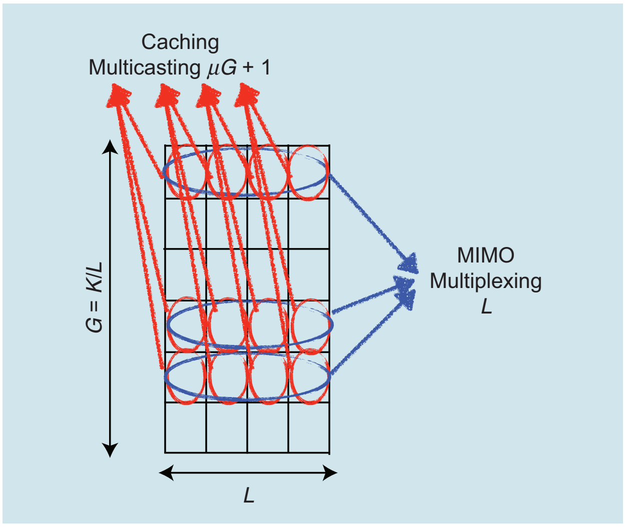


FIGURE 2. A qualitative representation of the delivery array of the MISO broadcast channel scheme in [43], where coded caching operates "on the columns" and spatial multiplexing operates "on the rows."

Note that the achieved DoF of this scheme are given by $L + \mu K = L + KM/N$, as previously obtained in the "Performance in the Homogeneous Case" section, which are known to be optimal under uncoded placement and distinct user demands.

Exploiting spatial reuse in extended (cellular) networks

In this section, we present a simple construction able to achieve the same DoF of the single BS MISO broadcast channel, in the case of an extended cellular network with spatial reuse. Consider a multicell system covering a certain geographic area of size A with B single-antenna BSs. A population of users is initially distributed as a Poisson point process (PPP) of intensity λ. The number of users K in the area A is a Poisson-distributed random variable with a mean of λA. The users move around the coverage area according to some random walk with independent increments. It is well known that, in this case, the marginal spatial distribution of the users at each point in time is also a PPP with intensity λ. We wish to design a scheme robust to mobility, i.e., the cache placement is done a priori and independent of the user positions, with the system capable of delivering the user requests for any realization of the PPP.

As discussed previously, we partition the users into G groups and use a MAN placement with parameters (N, M, G, and F) by replicating the cache content for all users in the same group $g = 1, \ldots, G$. Users select groups at random, such that each user group forms an independent thinning of the original PPP so that each user group is distributed according to an independent PPP with intensity λ/G. Assuming a symmetrical layout where each cell has the same area, A/B, it follows that the number of users of a given group g in each cell is an independent Poisson random variable with a mean of $\lambda A/(BG)$. Let us focus on a given reference cell and denote by $\ell_1, \ldots, \ell_G$ the number of users in groups $g = 1, \ldots, G$ inside the cell. As mentioned previously, these occupancy numbers are independent and Poisson distributed with the same mean: $\lambda A/(BG)$. The delivery process at each cell works in the same way, as explained in the "Cache Replication" section. The BS forms a delivery array with occupancy numbers $\ell_1, \ldots, \ell_G$ and serves each column of the delivery array in sequence, incurring a load given by (18). The average cell load can be trivially upper bounded by (19), where now $\mathbb{E}[\ell_{(1)}]$ is the expected value of the maximum of G-independent Poisson random variables. Accordingly, it appears that $\mathbb{E}[\ell_{(1)}]$ is similar to $\lambda A/(BG)$ (the mean of an individual occupancy number), up to the logarithmic terms in G.

For large area A, the number of users K is very close to its mean, i.e., $K \approx \lambda A$. Using $\lambda = K/A$, we find that the mean of the occupancy numbers is $K/(BG)$. Thus, a sensible choice for G is $G = K/B$, yielding that each cell contains, on average, one user per group. Using the simple upper bound (19) with this choice of G, we find the average load of each cell, that is,

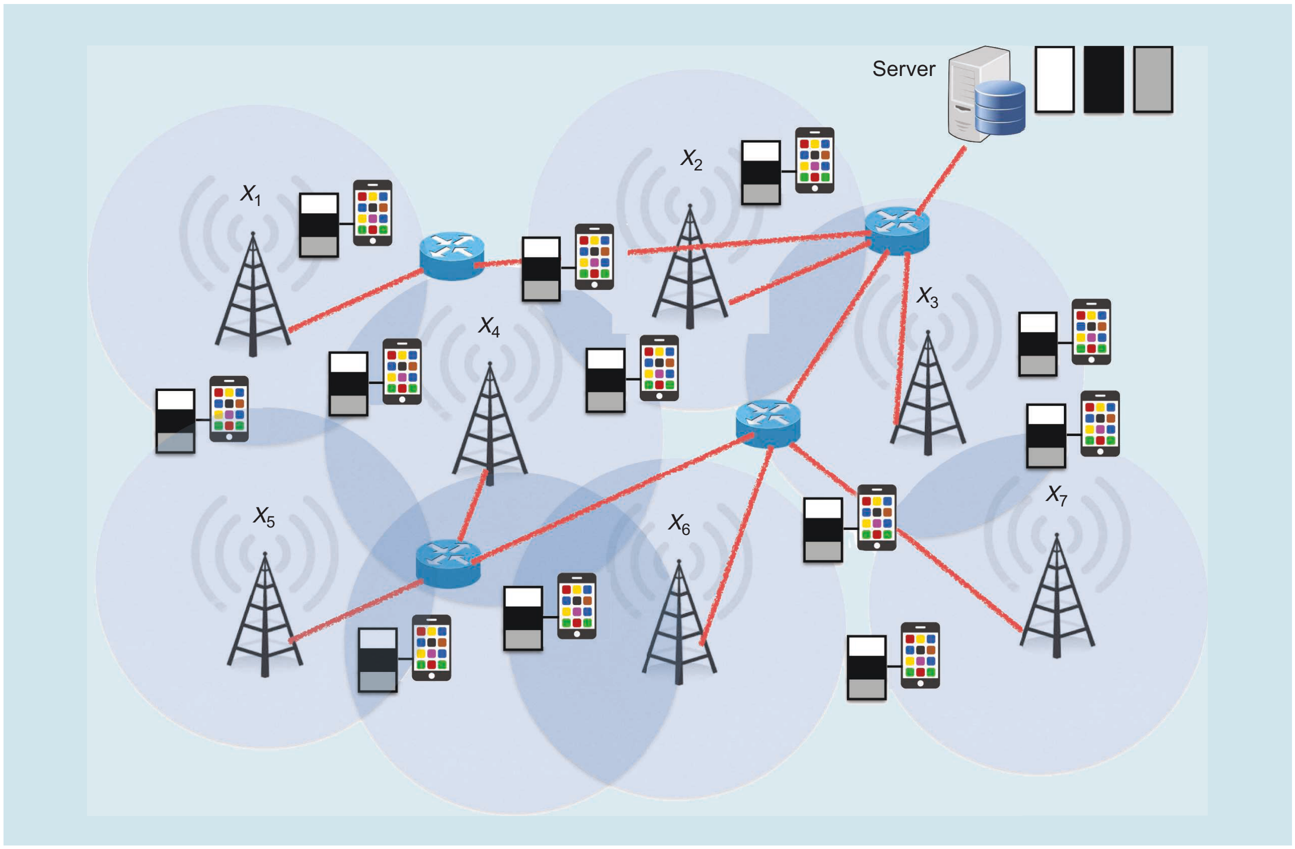


FIGURE 3. A qualitative representation of a cache-aided multicell system, handling mobility and the small subpacketization order.

$$\begin{aligned}\mathbb{E}[D(\ell_1,\ldots,\ell_G)] &\le \frac{G(1-\mu)}{1+\mu G}\cdot\mathbb{E}[\ell_{(1)}]\\ &= \frac{K/B(1-\mu)}{1+\mu K/B}\cdot\mathbb{E}[\ell_{(1)}]\\ &= \frac{K(1-\mu)}{B+\mu K}\cdot\mathbb{E}[\ell_{(1)}]. \qquad (21)\end{aligned}$$

Comparing (21) with (20), we see that the multicell system yields DoF equal to $(B+\mu K)/\mathbb{E}[\ell_{(1)}]$, where $\mathbb{E}[\ell_{(1)}]$ is of the order of $\log G$. Neglecting this logarithmic term, the DoF of the multicell system take on the same form as the DoF of a single-cell, cache-aided MISO broadcast channel with $L = B$ antennas. Because the subpacketization is formed for $G = K/B$ groups, as long as the number of users per cell K/B is a constant that does not grow with K, this system achieves a fixed subpacketization order $\approx \exp(G\mathcal{H}(\mu))$, while both the number of users K and the number of cells B grows arbitrarily large as long as they grow with fixed-ratio G. Note that present cellular systems are designed to balance the number of users per cell such that each cell handles a constant number of users to avoid congestion; therefore, the operating conditions of $K/B = O(1)$ are realistic for a well-designed cellular system.

Finally, we observe that the aforementioned simple analysis is done with the assumption that all cells can operate simultaneously on the same frequency band (frequency reuse 1). If the intercell interference resulting from reuse 1 is too large, then a standard frequency reuse scheme with some reuse factor m can be employed. In this case, the load is increased by m, which is typically a small integer (e.g., for the classical hexagonal layout typical values of m are 3, 4, or 7). Figure 3 shows the architecture of a multicell network based on these ideas. A more refined analysis of the achievable average load as well as a mixed-integer linear programming optimization problem for the case of reuse 1 and intercell interference modeled by the so-called protocol model, which serves as a simple conceptual model for a collision-based all-or-nothing interference channel (see the definition in [2]), is given in [44].

Coded caching: A broader picture

The basic MAN scheme was first introduced for the single shared-link network in [1]. Since then, several important follow-up works have appeared in the literature and present different aspects of coded caching. For the sake of completeness, in this section we briefly review some of the main challenges of coded caching addressed in these works.

Centralized versus decentralized caching

In many applications, the centralized placement of the MAN scheme is not practically feasible. For example, the set of users present in the network may change from the placement to the delivery phase because users join or leave the network in a dynamic manner. A decentralized caching scheme is proposed in [3] where the placement phase for each user is performed individually and independently from other users. More precisely, each user k stores MF/N packets from each file, chosen uniformly at random, and independently across the files and users. Similar to a centralized scheme, the server tries to maximize the utility of a multicasting message by combining requested packets; however, in the absence of a centralized placement, a packet intended for user k may be cached at a random number of other users, rather than at exactly $\alpha = KM/N$ users. Therefore, a wide range of utilities for multicasting packets is available, which results in an expected delivery time of

$$T_{\text{decentralized-caching}} = \frac{D \cdot F}{R} = K\left(1 - \frac{M}{N}\right)\frac{N}{KM}\left(1 - \left(1 - \frac{M}{N}\right)^K\right)\frac{F}{R}. \tag{22}$$

This leads to a loss compared to the delivery time of the centralized case [3].

Fundamental limits and optimality

The scheme proposed in [1] offers a significant global gain over an uncoded caching scheme that serves the users individually. However, a priori it is not clear whether it can be further improved using a more sophisticated placement and delivery scheme. Characterizing the optimum gain and the exact tradeoff between the cache size and the delivery load are not yet fully addressed. A cut-set-type argument was provided in [1], which proves that the basic MAN scheme is within a constant multiplicative gap from the optimum scheme. Several tighter outer bounds on the optimum tradeoff have been developed [45]–[50].

In particular, Wan et al. in [51] and [52] proved that the basic MAN scheme is optimal if all of the following conditions are fulfilled:

1) the cache contents are limited to being collections of segments of the files without any precoding (uncoded placement)
2) the cache contents are jointly and centrally optimized (centralized caching)
3) the user requests are all distinct (the worst demand profile).

This result was generalized in [42], where assumptions 2) and 3) are relaxed. More precisely, the optimum exact tradeoff between the cache size and the delivery load under the assumption of uncoded placement is characterized in [42]. It is shown that, when there is no overlap between the users' requests, the schemes of [1] and [3] are optimum for centralized and decentralized caching, respectively. Moreover, a novel caching strategy is introduced in [42] to exploit commonality among user demands and improve upon the gain of the basic MAN scheme, which shows that the new scheme is information-theoretically optimum. This fully characterizes the optimum tradeoff for the uncoded placement and the single shared-link network.

Coded versus uncoded placement

Even in the original work [1], it was observed that a coded placement [i.e., when the data placed in the caches are functions (e.g., XORs) of the original files] can improve the overall system performance and further reduce the load of delivery. Characterizing the optimum tradeoff and developing cache design for coded placement under centralized setting is studied in [53]–[56]. The common feature in the proposed cache designs is interfile coding, which allows for the combining of packets from different files and caching the coded packets during the placement phase. Note that such coded prefetched packets will be useless if the interference cannot be canceled during the delivery phase. In contrast, intrafile coded placement is introduced in [4] and [57], where it is demonstrated that individually encoding the files in the database using an erasure code can reduce the delivery load in the decentralized scheme. This is further improved upon using subspace coding in [58]. Finally, [59] develops an information-theoretic converse bound (infeasibility) that applies to any placement (coded or uncoded) and proves the optimality of the scheme in [60] within a factor of 2, which means that any more-complicated coded placements can gain at most a factor of 2 in the load with respect to the conceptually simpler uncoded placement of [60].

Network topologies

Beyond the single shared-link network considered in [1], several other coded caching network topologies have been studied in the literature. Here we summarize the most popular ones, for which often exact optimality or order optimality (i.e., the minimum worst-case load optimality up to multiplicative factors) have been determined.

Tree networks

In [1], the single shared-link network is generalized to a tree network where the server is at the root and the users are at the leaves. Intermediate nodes simply route the XOR-ed packets from one tree layer to the next. It is shown that a combination of routing and the original MAN scheme is order optimal for the tree network. The routing algorithm is very simple: consider an intermediate node in the tree at layer ℓ. Such a node receives XOR-ed packets, $X_{\mathcal{S}}$, from its parent at layer $\ell - 1$, and routes them to its ith child at layer $\ell + 1$ whenever at least one user $k \in \mathcal{S}$ is present in the subtree rooted at the ith child. That is, an XOR-ed packet is passed "down" to a node if it is useful for at least one (grand)child of that node.

Hierarchical two-level network

In [5], a network formed by the server, a layer of relays, and a layer of end users is considered. The server communicates with the relays via a single shared-link network and each relay also communicates with a subset of users via a "local" single shared-link network. Each user receives from only one relay. Caching memory is present at the relays and at the users. This network is called *hierarchical coded caching* because it is composed of a two-level hierarchy of single shared-link networks.

Shared caches network with arbitrary occupancy numbers

In [41], a variant of the single shared-link network is considered, where a server communicates with a layer of intermediate nodes, each of which has cache M via a single shared-link network. Each intermediate node serves a different number of possible users via ideal infinite capacity links. The number of users connected to a given intermediate node is denoted as the *occupancy number* of that node. The model may be motivated by a network of small-cell BSs with caches, which receive information from a controlling server or macro BS via a broadcast link (single shared-link between server and intermediate nodes), and serve their own users independently via a much faster local "access network," in each small cell. This model is formally identical to the case of shared user caches where the group of users connected to the same intermediate node actually share the same cache. It is also isomorphic to the case of users with identical copies of the cache, as originated by cache replication discussed more extensively in the "Exploiting Spatial Reuse in Extended (Cellular) Networks" section. In fact, all of the users connected to the same intermediate node behave as if they had their own individual cache, but the placement scheme replicates the same cache content in each one of them. Furthermore, the model is also related to the case of multiple requests; in fact, we can identify each individual node as a user, but each user makes multiple requests (one for each of the actual users connected to the intermediate node).

Multiserver linear network

This topology, presented and studied in [8], considers $L \geq 1$ servers, each of which has access to the full file library, serving K users, each of which has a cache size of M. The relation between the L inputs and the K outputs is given by $\mathbf{y} = \mathbf{H}\mathbf{x}$, where $\mathbf{y} \in \mathbb{F}_q^K$, $\mathbf{x} \in \mathbb{F}_q^L$ and $\mathbf{H} \in \mathbb{F}_q^{K \times L}$ are defined over a (typically large) finite field, $\mathbb{F}_q$. The rationale for this model is that the L servers communicate to the K users via some network for which end-to-end linear network coding is used instead of Internet Protocol routing. In this way, each user receives a linear combination of the information packets sent by the servers. The finite field size is chosen such that matrix $\mathbf{H}$ has rank $\min\{L, K\}$ with high probability when the network coding combination coefficients are chosen randomly.

The results for the multiserver linear network apply immediately to the case of a physical MISO downlink channel where the server is colocated with the BS and has L antennas, and the K users have a single antenna each. The Gaussian multiuser MISO version of the problem was studied in [25] and [43] and is examined in greater detail in the "Signal Processing Problem Formulation" section.

Gaussian interference channel

A generalization of the MISO downlink coded caching problem is a scenario in which the transmit antennas are separated transmitters, each of which has an individual cache of size M_T not necessarily equal to M (cache at the receivers). In this case, a necessary condition for the successful delivery of any user request is that the whole library can be stored in the network, i.e., $M_T L + MK \geq N$. A one-shot precoding solution for this network was provided in [32], while more elaborate schemes based on interference alignment with dimension expansion or signal-level expansion was presented in [61]. A recent extension of the one-shot precoding scheme to the case of nonfully connected interference channels arising from a cellular topology is presented in [62].

D2D coded caching

In [2], a D2D version of the coded caching problem is proposed and an order-optimal scheme is provided. The D2D network consists of a shared ideal channel where all the nodes can broadcast to all the other nodes, but only one node can talk at a time. This network may be motivated by a carrier sense multiple access D2D scheme (e.g., Wi-Fi Direct) or a token-ring medium access control protocol, where a collision avoidance mechanism permits only one node to be active at a time. However, when a node is active, all the other nodes can listen and decode its transmission. A necessary condition for the feasibility of the D2D coded caching network is that the whole library can be stored in the network, i.e., $KM \geq N$.

Combination network

The combination network consists of a server, a layer of L relays, and a layer of users. For a certain degree of connectivity, r, there are exactly $K = \binom{L}{r}$ users, one for each distinct combination of r relays. All the links connecting the server to relays and relays to users are orthogonal, i.e., there are no broadcast or interference constraints. In particular, a user connected to r relays can simultaneously receive the r-transmitted signals from these relays without interference. Coded caching for combination networks is studied in [63], where it is shown that a speed-up factor of $1/r$ in the delivery time with respect to the single shared-link network is possible for this network. Building on the combination network, in [64], a scheme for a multicell system with macrodiversity order r is proposed and analyzed in the case of MISO fading channels and distance-dependent path loss. Variants of the combination network, including the case of caches at the relays, have been analyzed in [65], while the improved strategies and information-theoretic optimality results are given in [66]–[69].

Challenges and open issues

The field of cache-aided communication is still in its infancy, and its challenges outnumber its achievements. Although research has shown significant throughput gains in various scenarios, questions remain unanswered and many issues must be resolved to allow for practical implementation. In this section, we describe some of these challenges and open issues.

Physical layer

Even though it is easy to obtain an ideal DoF scheme, the best cache-aided communication scheme for the general case is not yet known. An optimality proof exists only for the single-antenna homogenous case. For the multiple antenna case, even the linear optimal scheme is not yet known (all the results presented for this case are based on ZF). Research advances in this direction are needed to allow for a better understanding of the capabilities and to enable the efficient implementation of cache-aided communication schemes also at a low SNR.

More effort is also required to deal with physical-layer practicalities. For example, the effect of an imperfect channel state at the BS has been studied in only limited scenarios. In MIMO systems, such uncertainty will affect the ZF accuracy and may be addressed by a variety of known signal processing techniques. More importantly, this will create a rate uncertainty at the BS, which can have significant effects on transmission scheduling.

A scheme that can handle channel variations during the transmission stage has a significant advantage because this scheme will enable cache-aided communication in a wide area of cellular networks where the channel rates vary relatively fast. A simpler variation can apply cache-aided communication in networks where the rate changes over time but its average is known in advance.

Beyond linear precoding

For multiantenna BSs, we have discussed the use of linear precoding only. Linear precoding for cache-aided communications still requires further study; thus far, only ZF precoding has been studied extensively. Other approaches beyond linear precoding require even more attention.

Nonlinear precoding was thus far considered through two opposite approaches: on one hand, transmitting a single message at a time, where this message is an XOR of multiple file segments that can be used by multiple users (where each user can use its cache to cancel out the unintended file segments). This message should be beamformed to a direction that is favorable to all users served and can obtain some amount of array gain [9]. On the other hand, to obtain a multiplexing gain, the same work suggests serving multiple groups of users simultaneously, where the transmission to different groups is separated using ZF precoding, and each user in a group can extract its own message by XORing with its cache content. Neither of these schemes is optimal and combining both approaches in a single system to optimize the balance for specific network conditions can possibly offer a better performance. Nevertheless, the best approach for combining XORing and linear precoding remains unknown.

Taking it one step further, there is still much room for improvement using sophisticated nonlinear precoding schemes. For example, dirty paper coding and vector perturbation are more energy efficient than is linear ZF. Although not previously considered, the combinations of such schemes with multiantenna cache-aided communications may lead to significant gains, particularly in the low-to-medium SNR regime, where the ZF may be inefficient.

Scheduling and resource allocation

As shown in the previous section, cache-aided communication is much simpler in a homogenous network (where all the user rates are identical and all files are of the same popularity). Even in this case, existing scheduling methods require a very thin subpacketization, which, in most cases becomes unpractical for a large number of users. Thus, the research for scheduling approaches that will require a smaller number of packets is ongoing.

In a nonhomogenous case, the situation is even more complicated, and the only known solutions require solving a large optimization problem. Hence, a scheduling algorithm with an acceptable complexity that can handle network inhomogeneities is needed. In particular, if the user rates are not identical, each file segment is encoded to a different number of symbols that depend on the rate of the requesting user. Accordingly, the transmission scheduling must allocate more transmissions to users with a low rate, while still trying to serve a maximal number of users simultaneously at any given time.

Furthermore, in nonhomogenous networks, resource allocation is also a major challenge for cache-aided communications. Existing approaches are either highly inefficient or very complicated to implement. As a result, there is an acute need for low-complexity resource-allocation schemes (optimal or suboptimal) that enable the benefits of cache-aided communication in practical systems.

Additionally, further performance analysis and closed-form performance expressions of cache-aided communications in nonhomogeneous networks are needed. Such expressions are required to better predict the performance in various networks and for network planning. Performance expressions are also needed for network optimization, e.g., for power allocation, parameter selection, and so on.

Higher layers

Another major difficulty in cache-aided communications is the necessity of dividing the data into many subpackets. As discussed previously, this problem can be made simpler when using multiple antennas at the BS. Yet, this approach requires additional research, particularly for the nonhomogenous case.

Another high-layer issue that is crucial for practical implementation is the handling of network and content dynamics. Users' disconnection or movement from one BS (cell) to another causes changes in the network connectivity. Similarly, content dynamics can occur, for example, as a result of variations in the popularity of files. A practical network will likely meet all types of dynamics and must be robust to such changes. These aspects have hardly been addressed thus far and still require much research, e.g., How can the cache content be updated at minimal overhead? Are there schemes that allow for such a cache adaptation at low complexity? What are the performance costs of such a scheme?

As for network dynamics, schemes are needed for the adaptation of the transmission scheduling following a network change. Such schemes can consider intermediate planning (i.e., adapting to changes that happen between the cache placement and transmission stages) and online planning (i.e., adapting to changes that occur during the transmission phase). The development of such schemes is likely to be the final catalyst for the practical implementation of cache-aided communications.

Conclusions

Cache-aided communications have shown significant potential for throughput increase in wideband communication networks. The possibility of using the data stored at one user, even if they are only requested by another user, allows for combining the small size memories employed at different users and using them as an effectively large cache. Because the network performance depends on the total memory size of all the users, the network throughput scales linearly with the number of users. Thus, cache-aided communication is expected to take a significant role in large networks.

Yet, many challenges must be overcome prior to practical implementation. These challenges are mostly in the field of signal processing, and include low-complexity optimization, practical system design, and the handling of network imperfections. This article aimed at presenting this promising technology in a tractable manner that reflects its potential and open challenges.

Authors

Soheil Mohajer (soheil@umn.edu) received his B.Sc. degree in electrical engineering from Sharif University of Technology, Iran, in 2004 and his M.Sc. and Ph.D. degrees in communication systems, both from École Polytechnique Fédérale de Lausanne, Switzerland, in 2005 and 2010, respectively. He is currently an assistant professor in the Department of Electrical and Computer Engineering at the University of Minnesota, Twin Cities. Previously, he was a postdoctoral researcher at Princeton University, New Jersey (2010–2011), and the University of California, Berkeley (2011–2013). He received the National Science Foundation CAREER Award in 2018. He currently serves as an editor of *IEEE Transactions on Communication*s. His research interests include information theory and its applications in distributed storage systems, wireless networks, bioinformatics, and statistical machine learning. He is a Member of the IEEE.

Itsik Bergel (itsik.bergel@biu.ac.il) received his B.Sc. degree in electrical engineering and his B.Sc. degree in physics from Ben Gurion University, Beersheba, Israel, in 1993 and 1994, respectively, and his M.Sc. and Ph.D. degrees in electrical engineering from Tel Aviv University, Israel, in 2000 and 2005, respectively. Currently, he is with the Faculty of Engineering, Bar-Ilan University, Israel. From 2001 to 2003, he was a senior researcher with the Intel Communications Research Laboratory. In 2005, he was a postdoctoral researcher at Politecnico di Torino, Italy. He has been an associate editor of *IEEE Transactions on Signal Processing* and currently serves as an editor of *IEEE Transactions on Wireless Communications*. His main research interests include interference mitigation in wireline and wireless communications, cooperative transmission, and cross-layer optimization of random ad hoc networks. He is a Senior Member of the IEEE.

Giuseppe Caire (caire@tu-berlin.de) received his B.Sc. degree in electrical engineering from Politecnico di Torino, Turin, Italy, in 1990, his M.Sc. degree in electrical engineering from Princeton University, New Jersey, in 1992, and his Ph.D. degree from Politecnico di Torino in 1994. He is currently an Alexander von Humboldt Professor with the Faculty of Electrical Engineering and Computer Science at the Technical University of Berlin, Germany. He received the Jack Neubauer Best System Paper Award from the IEEE Vehicular Technology Society in 2003, IEEE Communications Society & Information Theory Society Joint Paper Award in 2004 and in 2011, IEEE Communications Society Leonard G. Abraham Prize for best *IEEE Journal on Selected Areas in Communications* paper in 2019, Okawa Research Award in 2006, Alexander von Humboldt Professorship in 2014, Vodafone Innovation Prize in 2015, and European Research Council Advanced Grant in 2018. He served as an officer on the Board of Governors of the IEEE Information Theory Society and, in 2011, was elected president of the IEEE Information Theory Society. He is a Fellow of the IEEE.

References

[1] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Trans. Inf. Theory*, vol. 60, no. 5, pp. 2856–2867, 2014. doi: 10.1109/TIT.2014.2306938.

[2] 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, 2016. doi: 10.1109/TIT.2015.2504556.

[3] 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, 2015. doi: 10.1109/TNET.2014.2317316.

[4] H. Reisizadeh, M. A. MaddahAli, and S. Mohajer, "Erasure coding for decentralized coded caching," in *Proc. IEEE Int. Symp. Information Theory (ISIT)*, 2018, pp. 1715–1719. doi: 10.1109/ISIT.2018.8437574.

[5] N. Karamchandani, U. Niesen, M. A. Maddah-Ali, and S. N. Diggavi, "Hierarchical coded caching," *IEEE Trans. Inf. Theory*, vol. 62, no. 6, pp. 3212–3229, 2016. doi: 10.1109/TIT.2016.2557804.

[6] R. Tandon and O. Simeone, "Cloud-aided wireless networks with edge caching: Fundamental latency trade-offs in fog radio access networks," in *Proc. 2016 IEEE Int. Symp. Information Theory (ISIT)*, pp. 2029–2033. doi: 10.1109/ISIT.2016.7541655.

[7] N. Mital, D. Gündüz, and C. Ling, "Coded caching in a multi-server system with random topology," in *Proc. 2018 IEEE Wireless Communications and Networking Conf. (WCNC)*, pp. 1–6. doi: 10.1109/WCNC.2018.8377365.

[8] 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, 2016. doi: 10.1109/TIT.2016.2614722.

[9] S. P. Shariatpanahi, G. Caire, and B. H. Khalaj, "Multi-antenna coded caching," in *Proc. 2017 IEEE Int. Symp. Information Theory (ISIT)*, pp. 2113–2117. doi: 10.1109/ISIT.2017.8006902.

[10] I. Bergel and S. Mohajer, "Cache-aided communications with multiple antennas at finite SNR," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 8, pp. 1682–1691, 2018. doi: 10.1109/JSAC.2018.2844618.

[11] K.-H. Ngo, S. Yang, and M. Kobayashi, "Cache-aided content delivery in MIMO channels," in *Proc. 2016 54th IEEE Annu. Allerton Conf. Communication, Control, and Computing (Allerton)*, pp. 93–100. doi: 10.1109/ALLERTON.2016.7852215.

[12] S. S. Bidokhti, M. Wigger, and R. Timo, "Noisy broadcast networks with receiver caching," *IEEE Trans. Inf. Theory*, vol. 64, no. 11, pp. 6996–7016, 2018. doi: 10.1109/TIT.2018.2835507.

[13] S. S. Bidokhti, M. Wigger, and A. Yener, "Benefits of cache assignment on degraded broadcast channels," in *Proc. 2017 IEEE Int. Symp. Information Theory (ISIT)*, pp. 1222–1226. doi: 10.1109/ISIT.2017.8006723.

[14] K.-H. Ngo, S. Yang, and M. Kobayashi, "Scalable content delivery with coded caching in multi-antenna fading channels," *IEEE Trans. Wireless Commun.*, vol. 17, no. 1, pp. 548–562, 2018. doi: 10.1109/TWC.2017.2768361.

[15] S. Yang, K.-H. Ngo, and M. Kobayashi, "Content delivery with coded caching and massive MIMO in 5G," in *Proc. 2016 9th IEEE Int. Symp. Turbo Codes and Iterative Information Processing (ISTC)*, pp. 370–374. doi: 10.1109/ISTC.2016.7593139.

[16] M. M. Amiri and D. Gündüz, "Cache-aided content delivery over erasure broadcast channels," *IEEE Trans. Commun.*, vol. 66, no. 1, pp. 370–381, 2018. doi: 10.1109/TCOMM.2017.2751608.

[17] M. Ji, G. Caire, and A. F. Molisch, "Wireless device-to-device caching networks: Basic principles and system performance," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 1, pp. 176–189, 2016. doi: 10.1109/JSAC.2015.2452672.

[18] D. Liu, B. Chen, C. Yang, and A. F. Molisch, "Caching at the wireless edge: Design aspects, challenges, and future directions," *IEEE Commun. Mag.*, vol. 54, no. 9, pp. 22–28, 2016. doi: 10.1109/MCOM.2016.7565183.

[19] E. Zeydan, E. Bastug, M. Bennis, M. A. Kader, I. A. Karatepe, A. S. Er, and M. Debbah, "Big data caching for networking: Moving from cloud to edge," *IEEE Commun. Mag.*, vol. 54, no. 9, pp. 36–42, 2016. doi: 10.1109/MCOM.2016.7565185.

[20] G. S. Paschos, G. Iosifidis, M. Tao, D. Towsley, and G. Caire, The role of caching in future communication systems and networks. 2018. [Online]. Available: arXiv:1805.11721

[21] A. Meyerson, K. Munagala, and S. Plotkin, "Web caching using access statistics," in *Proc. 12th Annu.ACM-SIAM Symp. Discrete Algorithms*. Philadelphia: Society for Industrial and Applied Mathematics, 2001, pp. 354–363.

[22] S. Borst, V. Gupta, and A. Walid, “Distributed caching algorithms for content distribution networks,” in *Proc. 2010 IEEE INFOCOM,* pp. 1–9. doi: 10.1109/INFCOM.2010.5461964.

[23] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, “Web caching and Zipf-like distributions: Evidence and implications,” in *Proc. IEEE INFOCOM’99. 18th Annu. Joint Conf. Computer and Communications Societies*, 1999, vol. 1, pp. 126–134. doi: 10.1109/INFCOM.1999.749260.

[24] J. Wang, “A survey of web caching schemes for the internet,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 29, no. 5, pp. 36–46, 1999. doi: 10.1145/505696.505701.

[25] S. P. Shariatpanahi, G. Caire, and B. H. Khalaj, “Physical-layer schemes for wireless coded caching,” *IEEE Trans. Inf. Theory*, vol. 65, no. 5, pp. 2792–2807, 2018. doi: 10.1109/TIT.2018.2888615.

[26] M. Ji, A. M. Tulino, J. Llorca, and G. Caire, “Order-optimal rate of caching and coded multicasting with random demands,” *IEEE Trans. Inf. Theory*, vol. 63, no. 6, pp. 3923–3949, 2017. doi: 10.1109/TIT.2017.2695611.

[27] R. Pedarsani, M. A. Maddah-Ali, and U. Niesen, “Online coded caching,” *IEEE/ACM Trans. Netw.*, vol. 24, no. 2, pp. 836–845, 2016. doi: 10.1109/TNET.2015.2394482.

[28] E. Bastug, M. Bennis, and M. Debbah, “Living on the edge: The role of proactive caching in 5G wireless networks,” *IEEE Commun. Mag.*, vol. 52, no. 8, pp. 82–89, 2014. doi: 10.1109/MCOM.2014.6871674.

[29] S. Mohajer and I. Bergel, “Optimal power allocation in MISO cache-aided communication,” in *Proc. 2018 IEEE 19th Int. Workshop Signal Processing Advances Wireless Communications (SPAWC)*, pp. 1–5. doi: 10.1109/SPAWC.2018.8445770.

[30] S. Jin, Y. Cui, H. Liu, and G. Caire, Structural properties of uncoded placement optimization for coded delivery. 2017. [Online]. Available: arXiv:1707.07146

[31] Q. Yang and D. Gündüz, “Coded caching and content delivery with heterogeneous distortion requirements,” *IEEE Trans. Inf. Theory*, vol. 64, no. 6, pp. 4347–4364, 2018. doi: 10.1109/TIT.2018.2805331.

[32] 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, 2017. doi: 10.1109/TIT.2017.2669942.

[33] 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, 2016. doi: 10.1109/TIT.2016.2599110.

[34] 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, Sept. 2017. doi: 10.1109/TIT.2017.2725272.

[35] Q. Yan, X. Tang, and Q. Chen, On the placement and delivery schemes for decentralized coded caching system. 2017. [Online]. Available: arXiv:1710.04884

[36] S. Jin, Y. Cui, H. Liu, and G. Caire, “Uncoded placement optimization for coded delivery,” in *Proc. 2018 IEEE 16th Int. Symp. Modeling and Optimization Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, pp. 1–8. doi: 10.23919/WIOPT.2018.8362816.

[37] 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. Information Theory (ISIT)*, 2017, pp. 1237–1241. doi: 10.1109/ISIT.2017.8006726.

[38] L. Tang and A. Ramamoorthy, “Low subpacketization schemes for coded caching,” in *Proc. 2017 IEEE Int. Symp. Information Theory (ISIT)*, pp. 2790–2794. doi: 10.1109/ISIT.2017.8007038.

[39] S. Jin, Y. Cui, H. Liu, and G. Caire, “Order-optimal decentralized coded caching schemes with good performance in finite file size regime,” in *Proc. IEEE Global Communications Conf. (GLOBECOM)*, Dec. 2016, pp. 1–7. doi: 10.1109/GLOCOM.2016.7842115.

[40] S. Jin, Y. Cui, H. Liu, and G. Caire, New order-optimal decentralized coded caching schemes with good performance in the finite file size regime. 2016. [Online]. Available: arXiv:1604.07648

[41] E. Parrinello, A. Unsal, and P. Elia, Fundamental limits of caching in heterogeneous networks with uncoded prefetching. 2018. [Online]. Available: arXiv:1811.06247

[42] 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, 2018. doi: 10.1109/TIT.2017.2785237.

[43] E. Lampiris and P. Elia, “Adding transmitters dramatically boosts coded-caching gains for finite file sizes,” *IEEE J. Sel. Areas Commun.*, vol. 36, no. 6, pp. 1176–1188, 2018. doi: 10.1109/JSAC.2018.2844960.

[44] A. Asadzadeh and G. Caire, “Coded caching with small subpacketization via spatial reuse and content base replication,” in *Proc. 2019 IEEE Int. Symp. Information Theory (ISIT)*, pp. 2982–2986. doi: 10.1109/ISIT.2019.8849460.

[45] C. Tian, “Symmetry, outer bounds, and code constructions: A computer-aided investigation on the fundamental limits of caching,” *Entropy*, vol. 20, no. 8, p. 603, 2018. doi: 10.3390/e20080603.

[46] C.-Y. Wang, S. H. Lim, and M. Gastpar, “A new converse bound for coded caching,” in *Proc. 2016 IEEE Information Theory and Applications Workshop (ITA)*, pp. 1–6. doi: 10.1109/ITA.2016.7888186.

[47] H. Ghasemi and A. Ramamoorthy, “Improved lower bounds for coded caching,” *IEEE Trans. Inf. Theory*, vol. 63, no. 7, pp. 4388–4413, 2017. doi: 10.1109/TIT.2017.2705166.

[48] A. Sengupta and R. Tandon, “Improved approximation of storage-rate tradeoff for caching with multiple demands,” *IEEE Trans. Commun.*, vol. 65, no. 5, pp. 1940–1955, 2017. doi: 10.1109/TCOMM.2017.2664815.

[49] S. H. Lim, C.-Y. Wang, and M. Gastpar, “Information-theoretic caching: The multi-user case,” *IEEE Trans. Inf. Theory*, vol. 63, no. 11, pp. 7018–7037, 2017. doi: 10.1109/TIT.2017.2733527.

[50] N. Ajaykrishnan, N. S. Prem, V. M. Prabhakaran, and R. Vaze, “Critical database size for effective caching,” in *Proc. IEEE 2015 21st Nat. Conf. Communications (NCC)*, pp. 1–6. doi: 10.1109/NCC.2015.7084871.

[51] K. Wan, D. Tuninetti, and P. Piantanida, “On caching with more users than files,” in *Proc. IEEE Int. Symp. Information Theory (ISIT)*, 2016, pp. 135–139. doi: 10.1109/ISIT.2016.7541276.

[52] K. Wan, D. Tuninetti, and P. Piantanida, “On the optimality of uncoded cache placement,” in *Proc. 2016 IEEE Information Theory Workshop (ITW)*, pp. 161–165. doi: 10.1109/ITW.2016.7606816.

[53] Z. Chen, P. Fan, and K. B. Letaief, Fundamental limits of caching: Improved bounds for small buffer users. 2014. [Online]. Available: arXiv:1407.1935

[54] C. Tian and J. Chen, “Caching and delivery via interference elimination,” *IEEE Trans. Inf. Theory*, vol. 64, no. 3, pp. 1548–1560, 2018. doi: 10.1109/TIT.2018.2794543.

[55] M. M. Amiri and D. Gündüz, “Fundamental limits of coded caching: Improved delivery rate-cache capacity tradeoff,” *IEEE Trans. Commun.*, vol. 65, no. 2, pp. 806–815, 2017. doi: 10.1109/TCOMM.2016.2638841.

[56] S. Sahraei and M. Gastpar, “K users caching two files: An improved achievable rate,” in *Proc. IEEE 2016 Annu. Conf. Information Science and Systems (CISS)*, pp. 620–624. doi: 10.1109/CISS.2016.7460574.

[57] Y.-P. Wei and S. Ulukus, “Novel decentralized coded caching through coded prefetching,” in *Proc. 2017 IEEE Information Theory Workshop (ITW)*, pp. 1–5. doi: 10.1109/ITW.2017.8278044.

[58] H. Reisizadeh, M. A. MaddahAli, and S. Mohajer, “Subspace coding for coded caching: Decentralized and centralized placements meet for three users,” in *Proc. IEEE Int. Symp. Information Theory (ISIT)*, 2019, pp. 677–681. doi: 10.1109/ISIT.2019.8849613.

[59] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, “Characterizing the rate-memory tradeoff in cache networks within a factor of 2,” *IEEE Trans. Inf. Theory*, vol. 65, no. 1, pp. 647–663, 2019. doi: 10.1109/TIT.2018.2870566.

[60] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, The exact rate-memory tradeoff for caching with uncoded prefetching. 2016. [Online]. Available: arXiv:1609.07817

[61] J. Hachem, U. Niesen, and S. Diggavi, “Degrees of freedom of cache-aided wireless interference networks,” *IEEE Trans. Inf. Theory*, vol. 64, no. 7, pp. 5359–5380, 2018. doi: 10.1109/TIT.2018.2825321.

[62] N. Naderializadeh, M. A. Maddah-Ali, and A. S. Avestimehr, “Cache-aided interference management in wireless cellular networks,” *IEEE Trans. Commun.*, vol. 67, no. 5, pp. 3376–3387, 2019. doi: 10.1109/TCOMM.2019.2893669.

[63] M. Ji, M. F. Wong, A. M. Tulino, J. Llorca, G. Caire, M. Effros, and M. Langberg, “On the fundamental limits of caching in combination networks,” in *Proc. 2015 IEEE 16th Int. Workshop Signal Processing Advances Wireless Communications (SPAWC)*, pp. 695–699. doi: 10.1109/SPAWC.2015.7227127.

[64] M. Bayat, R. K. Mungara, and G. Caire, “Achieving spatial scalability for coded caching via coded multipoint multicasting,” *IEEE Trans. Wireless Commun.*, vol. 18, no. 1, pp. 227–240, 2019. doi: 10.1109/TWC.2018.2878845.

[65] A. A. Zewail and A. Yener, “Coded caching for combination networks with cache-aided relays,” in *Proc. IEEE Int. Symp. Information Theory (ISIT)*, June 2017, pp. 2433–2437. doi: 10.1109/ISIT.2017.8006966.

[66] K. Wan, D. Tuninetti, M. Ji, and P. Piantanida, “State-of-the-art in cache-aided combination networks,” in *Proc. 2017 IEEE 51st Asilomar Conf. Signals, Systems, and Computers*, pp. 641–645. doi: 10.1109/ACSSC.2017.8335420.

[67] K. Wan, M. Ji, P. Piantanida, and D. Tuninetti, Novel outer bounds and inner bounds with uncoded cache placement for combination networks with end-user-caches. 2017. [Online]. Available: arXiv:1701.06884

[68] K. Wan, M. Ji, P. Piantanida, and D. Tuninetti, “On the benefits of asymmetric coded cache placement in combination networks with end-user caches,” in *Proc. 2018 IEEE Int. Symp. Information Theory (ISIT)*, pp. 1550–1554. doi: 10.1109/ISIT.2018.8437462.

[69] K. Wan, M. Ji, P. Piantanida, and D. Tuninetti, “Caching in combination networks: Novel multicast message generation and delivery by leveraging the network topology,” in *Proc. 2018 IEEE Int. Conf. Communications (ICC)*, pp. 1–6. doi: 10.1109/ICC.2018.8422197.

SP