Non-Symmetric Multi-Antenna Coded Caching for Location-Dependent Content Delivery

Hamidreza Bakhshzad Mahmoodi, *Student Member, IEEE,* MohammadJavad Salehi, *Member, IEEE,* and Antti Tölli *Senior Member, IEEE*

***Abstract*—Immersive viewing, as the next-generation interface for human-computer interaction, is emerging as a wireless application. A genuinely wireless immersive experience necessitates immense data delivery with ultra-low latency, raising stringent requirements for future wireless networks. In this regard, efficient usage of in-device storage and computation capabilities is a potential candidate for addressing these requirements. In addition, recent advancement in multi-antenna transmission has significantly enhanced wireless communication. Hence, this paper proposes a novel location-based multi-antenna coded cache placement and delivery scheme. We first formulate a linear programming cache allocation problem to provide a uniform quality of experience in different network locations; then, cache-placement is done for each location independently. Subsequently, based on the users' spatial realizations, a transmission vector is created considering diverse available memory at each user. Moreover, a weighted-max-min optimization is used for the beamformers to support different transmission rates. Finally, numerical results are used to show the performance of the proposed scheme.**

***Index Terms*—Multi-antenna communications, coded caching, Location-Dependent Caching; Immersive Viewing.**

I. Introduction

Mobile flat-screen devices such as smartphones and tablets are the current dominant interface for human-computer interaction. However, wireless immersive viewing experiences facilitated by more capable wearable gadgets submerging users into the three-dimensional (3D) digital world is expected to bring forward the next interface evolution. In this regard, powerful and agile external radio connections are required to support the highly stringent requirements of such evolution. These requirements are simply beyond what is possible with the current networking standards [1]. As a result, new techniques that leverage recent advances in communication, storage, and machine learning are highly demanded [2].

In this regard, using cheap on-device storage is considered a promising technique for improving bandwidth efficiency [3]. Moreover, utilizing caching and computing capabilities of mobile VR devices is shown to be effective in alleviating the traffic burden over the wireless network [4]. Recently, a new caching technique known as *Coded Caching (CC)* is introduced in [5], which yields significantly higher caching gain compared to the traditional caching schemes. The original CC scheme in [5] is extended to multi-server networks in [6], and later to wireless multi-antenna systems in [7]. Meantime, various practical limitations of coded caching have been addressed by the research community. For instance, authors in [8] address the large subpacketization requirement of the CC, which is defined as the number of smaller parts each file should be split into. At the same time, the effect of the subpacketization on the low-SNR rate is investigated in [9].

Near-far issue, which affects content delivery applications in general and immersive viewing applications in particular, is a less-studied problem of the CC schemes. To illustrate, the users with the worst channel condition always limit the achievable rate in the CC schemes. In this regard, to avoid serving users in adverse fading conditions, a congestion control technique is proposed in [10], while in [11] ill-conditioned users are served with a lower quality-of-experience (QoE) using multiple descriptor codes. Unlike [10], [11], which are based on traditional XOR-ing of data elements, in [12] nested code modulation is proposed to allow building codewords that serve every user in the multicasting group with a different rate. Later on, authors in [13] extended the system model in [12] to a dynamic real-time applications, where users can move inside the network, and their achievable rate changes accordingly.

On the other hand, considering the recent advancement in the multi-antenna devices and fast converging beamforming approaches proposed in the literature (e.g., [14] and [15]), the next step is to extend [13] to a multi-antenna environment. As a result, this paper introduces a new multi-antenna location-dependent coded caching scheme for efficient content delivery in wireless access networks. We consider a wireless communication scenario in which the users are free to move, and their requested content depends on their current locations.

The requested content at each location is assumed to be of the same size. As a specific use-case, we assume a multi-user immersive viewing environment where a group of users is submerged into a network-based immersive application that runs on high-end eyewear. Such a use case necessitates heavy multimedia traffic and guaranteed QoE throughout the operating environment. In this regard, a location-dependent, uneven memory allocation is carried out based on the attainable data rate at each given location. Moreover, a multicast transmission scheme based on weighted-max-min optimization is devised to deliver the missing user-specific content with different transmission rates. Finally, it is worth noting that the worst-case delivery time is minimized across all the locations due to the optimized location-dependent cache placement.

This work was supported by the Academy of Finland under grants no. 319059 (Coded Collaborative Caching for Wireless Energy Efficiency), 318927 (6Genesis Flagship) and Nokia foundation.

II. System Model

We envision a bounded environment (game hall, operating theatre, etc.) in which a server with L antennas serves K single-antenna users through a wireless communication link. The set of users is denoted by $\mathcal{K} = [K]$, where $[K]$ denotes the set of integer numbers $\{1, ..., K\}$. The users are equipped with finite-size cache memories and are free to move throughout the environment. Every user requests data from the server at each time slot based on the application's needs and its location. The requested data content can be divided into static and dynamic parts, where the former can be proactively stored in the user cache memories. This paper focuses on the wireless delivery of the static location-dependent content, partially aided by in-device cache memories.[1] A real-world application of this communication setup is a wireless immersive digital experience environment, where the requested data is needed to reconstruct the location-dependent 3D field-of-view (FoV) at each user. Due to the wireless nature, users in different locations experience different channel conditions. Therefore, the goal is to design a cache-aided communication scheme that minimizes the maximum required delivery time to serve all the users. In other words, the aim is to provide a uniform QoE, irrespective of the users' location.

Intuitively, a larger share of the total cache memory should be reserved for storing data needed in locations where the communication quality is poor. In this regard, We split the environment into S regions, such that all points in a given region have almost the same distance from the server (i.e., the channel-state can be considered the same for all the points in a given region). In the following, we refer to these regions as states and denote the set of states as $\mathcal{S}$. A graphical example of an application environment with its states is provided in Figure 1. The file required for reconstructing the FoV of state $j \in \mathcal{S}$ is denoted by $W(j)$. We assume for every region $j \in \mathcal{S}$, the size of $W(j)$ is F bits, and every user is equipped with a cache memory of size MF bits. For the sake of simplicity, we consider a normalized data unit and drop F in subsequent notations. Moreover, we consider the delivery procedure in a specific time slot and ignore the time index (the same procedure is repeated every time slot).

We assume a wideband communication scheme, where the total bandwidth is divided into several small frequency bins. To perform the location-dependent cache placement, we need an estimation of the achievable rate at different states. To this end, we define a single-user scenario, where we assume there exists only one user to be served. As a result, the expected interference-free data rate attained in state $j \in \mathcal{S}$ is approximated as

$$\bar{r}(j) = \mathbb{E}[\log(1 + \frac{P_T \ \|\mathbf{h}_j\|^2}{N_0})] \tag{1}$$

where P_T is the transmission power, N_0 is the additive white Gaussian noise power, and $\mathbf{h}_j \in \mathbb{C}^L$ is the channel vector between the server and the user located in state j[2]. Thus, the expected data rate over all the frequency bins is approximated as $\hat{r}(j) \sim B\bar{r}(j)$, where B is the communication bandwidth. For ease of exposition, we consider normalized data rate, i.e., $r(j) = \frac{\hat{r}(j)}{F}$, throughout this paper.

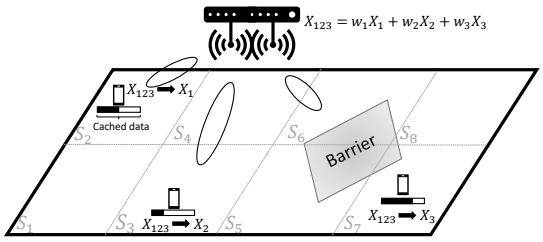


Fig. 1: An application environment with $K = 3$ users, split into $S = 8$ states, where $r(3) > r(2) > r(7)$. The black bar below each user indicates how much of the requested data is cached.

III. Cache Placement

The cache placement phase in this paper comprises two processes, 1) memory allocation and 2) data placement.

Memory Allocation: Before proceeding with the data placement, we first conduct a *memory allocation* process to determine the dedicated amount of cache memory for storing (parts of) $W(j)$ at each user. In this regard, since there is no prior knowledge about the users' spatial realizations in the delivery phase, we minimize the maximum delivery time for the single-user scenario assuming uniform access probability for all the states. Let us denote $m(j)$ as the allocated normalized cache size at each user for storing (parts of) $W(j)$. Since the size of $W(j)$ is normalized to one data unit, a user in state j needs to receive $1 - m(j)$ data units over the wireless link to reconstruct the FoV of state j. As a result, the delivery time to transmit data for state j is $T(j) = \frac{1-m(j)}{r(j)}$ seconds. Hence, the memory allocation is done by solving the following linear programming (LP):

$$\begin{aligned} \text{LP} \ : \quad & \min_{m(j), \gamma \geq 0} \ \gamma \\ & \text{s.t.} \quad \frac{1 - m(j)}{r(j)} \leq \gamma \ \forall j \in \mathcal{S}, \ \sum_{j \in \mathcal{S}} m(j) = M. \end{aligned} \tag{2}$$

The solution to (2) is given in closed-form using Karush-Kuhn-Tucker (KKT) conditions:

$$\gamma = \frac{S - M}{\sum_{j \in \mathcal{S}} r(j)}, \quad m(j) = 1 - \frac{(S - M) r(j)}{\sum_{j \in \mathcal{S}} r(j)}, \ \forall j \in \mathcal{S}. \tag{3}$$

Data Placement: After the memory allocation process, we store data in the cache memories of the users following the cache placement method proposed in [9].[3] In this regard, we define S location-dependent binary matrices $\mathbf{V}_j, \ \forall j \in \mathcal{S}$, with size $K \times K$. The first row of $\mathbf{V}_j$ has $t(j) = Km(j)$ consecutive one elements (other elements are zero); and for

[1] We assume that a portion of the achievable data rate available at each user is dedicated to deliver the dynamic content without cache assistance.

[2] Although we use (1) for convenience, the expected location-specific data rates can be attained through various means, e.g., via collecting statistics from past active users.

[3] With appropriate modifications, the model can be applied to other CC schemes also. Nevertheless, a thorough discussion is left for the extended version of the paper.

	$j=1$	$j=2$	$j=3$	$j=4$	$j=5$
Expected rate $r(j)$	3	2	1	2	3
Allocated memory $m(j)$	0.25	0.5	0.75	0.5	0.25

TABLE I: Location-specific rate and memory allocation for Example 1.

the other rows, each row is a circular shift of the previous row by one unit.[4] Next, for every $j \in \mathcal{S}$ we split $W(j)$ into K packets denoted by $W_p(j)$. Then, at the cache memory of user k, we store $\{W_p(j), \forall p\}$ for every state $j \in \mathcal{S}$ if $\mathbf{V}_j[p,k] = 1$.

Example 1. *Consider an application with $K = 4$ users and a server with $L = 2$ antennas, where the environment is split into $S = 5$ states and for each state, the required data size is $F = 400$ Megabytes. Each user has a cache size of 900 Megabytes, and hence, the normalized cache size is $M = 2.25$ data units. The spatial distribution of the achievable rate and its resulting memory allocation are as shown in Table I. It can be easily verified that $t(1) = t(5) = 1$, $t(2) = t(4) = 2$, and $t(3) = 3$. As a result, each file should be split into four sub-files, and each user caches one sub-file from $W(1)$, and $W(5)$, two sub-files from $W(2)$ and $W(4)$, and three sub-files from $W(3)$, based on the following placement matrices,*

$$\mathbf{V}_1 = \mathbf{V}_5 = \begin{pmatrix} 1&0&0&0 \\ 0&1&0&0 \\ 0&0&1&0 \\ 0&0&0&1 \end{pmatrix}, \mathbf{V}_2 = \mathbf{V}_4 = \begin{pmatrix} 1&1&0&0 \\ 0&1&1&0 \\ 0&0&1&1 \\ 1&0&0&1 \end{pmatrix}, \mathbf{V}_3 = \begin{pmatrix} 1&1&1&0 \\ 0&1&1&1 \\ 1&0&1&1 \\ 1&1&0&1 \end{pmatrix}.$$

As $W(j)$ is split into K sub-files and $t(j)$ sub-files are stored in the cache memory of each user, the total memory size dedicated to $W(j)$ at each user is $t(j) \times \frac{1}{K} = m(j)$, and hence, the proposed algorithm satisfies the cache size constraints. In comparison with [9], here the required files in each state are considered as a separate library, and the cache placement algorithm in [9] is performed for each state independent of the others. Also, different from the existing works, here, files of different locations have distinct $t(j)$ values, which should be carefully considered in the delivery phase.

IV. Delivery

At the beginning of the delivery phase, every user $k \in \mathcal{K}$ reveals its requested file $W_k \equiv W(s_k)$.[5] Note that W_k depends on the state s_k where user k is located. The server then builds and transmits several multiplexed codewords, such that after receiving the codewords, all users can reconstruct their requested files. From the system model, user k requires a total amount of one normalized data unit to reconstruct W_k. However, a subset of this data, with size $m_k \equiv m(s_k)$ data units, is available in its cache. Note that due to the distinct caching gain of users in different states, the conventional delivery scheme of [9], which is based on the same caching gain for all the users, is no longer applicable to the considered network.

To solve this issue, we first consider a virtual network with the same setup as the original one, but assuming each user has the same cached data ratio of $\bar{m} = \min m_k$. To avoid confusion, we use $\bar{k}$ to mention the virtual twin of the user k. As all virtual users have the same cached data ratio, the caching gain for all users in the virtual network is the same (independent of their location) and equal to $\bar{t} = K\bar{m}$. Now, applying the delivery scheme of [9] to the virtual network, virtual users are served in K rounds, where at each round, $K - \bar{t}$ transmissions are performed. The corresponding vector for the j-th transmission at round r is built as

$$\bar{\mathbf{x}}_j^r = \sum_{n \in [\bar{t}+L]} \bar{W}^{\bar{q}}_{\bar{\mathbf{p}}_j^r[n]}(\bar{\mathbf{k}}_j^r[n]) \bar{\mathbf{w}}_{\bar{\mathcal{R}}_j^r(n)} \tag{4}$$

where $\bar{\mathbf{p}}_j^r$ and $\bar{\mathbf{k}}_j^r$ are the packet and user index vectors, and $\bar{\mathcal{R}}_j^r(n)$ is the interference indicator set used for the n-th data term, in the j-th transmission of round r. Also, $\bar{\mathbf{w}}_{\bar{\mathcal{R}}_j^r(n)}$ is the optimized beamforming vector suppressing data at every user in $\bar{\mathcal{R}}_j^r(n)$, $W(\bar{k})$ denotes the file requested by the virtual user $\bar{k}$. Also, $\bar{q}$ is the subpacket index which is initialized to one and increased every time a packet appears in a transmission vector. All these parameters are exactly defined in [9]. Specifically, $\bar{\mathbf{p}}_j^r$ and $\bar{\mathbf{k}}_j^r$ are built such that the graphical representation of the transmission vectors follows two perpendicular circular shift operations over a grid. The following example clarifies how the transmission vectors are built for the virtual network.

Example 2. *Consider the network in Example 1, and assume in a specific time slot, $s_1 = 1$, $s_2 = 2$, $s_3 = 3$, $s_4 = 4$. Thus, the virtual network will have four users $\bar{\mathcal{K}} = \{\bar{1}, \bar{2}, \bar{3}, \bar{4}\}$, and coded caching gain is $\bar{t} = 1$. Then, the data delivery will require four rounds, where at each round, three transmissions are done. According to [9], user and packet index vectors for the first round are built as*

$$\begin{aligned} &\bar{\mathbf{k}}_1^1 = [\bar{1}, \bar{2}, \bar{3}], \quad \bar{\mathbf{k}}_2^1 = [\bar{1}, \bar{3}, \bar{4}], \quad \bar{\mathbf{k}}_3^1 = [\bar{1}, \bar{4}, \bar{2}] \ , \\ &\bar{\mathbf{p}}_1^1 = [\bar{2}, \bar{1}, \bar{1}], \quad \bar{\mathbf{p}}_2^1 = [\bar{3}, \bar{1}, \bar{1}], \quad \bar{\mathbf{p}}_3^1 = [\bar{4}, \bar{1}, \bar{1}] \ , \end{aligned} \tag{5}$$

and the resulting transmission vectors are

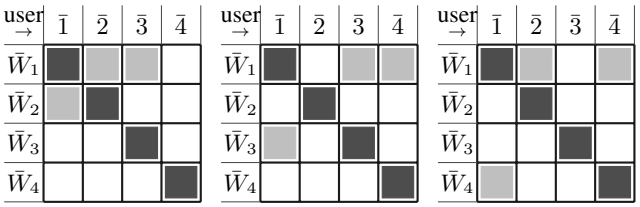
$$\begin{aligned} \bar{\mathbf{x}}_1^1 &= \bar{W}^{\bar{1}}_{\bar{2}}(\bar{1})\bar{\mathbf{w}}_{\bar{3}} + \bar{W}^{\bar{1}}_{\bar{1}}(\bar{2})\bar{\mathbf{w}}_{\bar{3}} + \bar{W}^{\bar{1}}_{\bar{1}}(\bar{3})\bar{\mathbf{w}}_{\bar{2}} \ , \\ \bar{\mathbf{x}}_2^1 &= \bar{W}^{\bar{1}}_{\bar{3}}(\bar{1})\bar{\mathbf{w}}_{\bar{4}} + \bar{W}^{\bar{2}}_{\bar{1}}(\bar{3})\bar{\mathbf{w}}_{\bar{4}} + \bar{W}^{\bar{1}}_{\bar{1}}(\bar{4})\bar{\mathbf{w}}_{\bar{3}} \ , \\ \bar{\mathbf{x}}_3^1 &= \bar{W}^{\bar{1}}_{\bar{4}}(\bar{1})\bar{\mathbf{w}}_{\bar{2}} + \bar{W}^{\bar{2}}_{\bar{1}}(\bar{4})\bar{\mathbf{w}}_{\bar{2}} + \bar{W}^{\bar{2}}_{\bar{1}}(\bar{2})\bar{\mathbf{w}}_{\bar{4}} \ , \end{aligned} \tag{6}$$

where the brackets for interference indicator sets are dropped for notational simplicity. For better clarification, we use a tabular view, borrowed from [9], to graphically represent the transmission vectors in Figure 2. In this representation, columns and rows denote user and packet indices, respectively. A darkly shaded cell means the packet index is cached at the user, while a lightly shaded cell indicates that (part of) the packet index is transmitted to the user. As can be seen in Figure 2, the transmission vectors in a single round are built using circular shift operations of the lightly shaded cells over non-colored cells of the table, in two perpendicular directions.

Next, we use an *elevation* mechanism to build transmission vectors for the original network using the vectors for the virtual network. Every transmission vector $\bar{\mathbf{x}}_j^r$ is elevated into exactly one transmission vector $\mathbf{x}_j^r$, and hence, data delivery for the original network is also done in K rounds where at each round,

[4]Here we assume for every $j \in \mathcal{S}$, $m(j) > 0$ and $t(j)$ is an integer. For the non integer caching gains, memory sharing will be adopted. The details will be provided in the extended version of this paper.

[5]Note that we have used $W(j)$ to represent the file required for reconstructing the FoV of the state j, and W_k to denote the file requested by user k. The same convention is used for all notations in the text.



(a) $\bar{\mathbf{x}}_1^1$ (b) $\bar{\mathbf{x}}_2^1$ (c) $\bar{\mathbf{x}}_3^1$

Fig. 2: Graphical representation of transmission vectors for the virtual network

$K-\bar{t}$ transmissions are performed. The transmission vector $\mathbf{x}_j^r$ is built as

$$\mathbf{x}_j^r = \sum_{n\in[\bar{t}+L]} \prod_{m\in[\alpha]} W^q_{\mathbf{P}_j^r[n,m]}(s_{\mathbf{k}_j^r[n]})_{\mathbf{W}_{\mathcal{R}_j^r(n)}} \tag{7}$$

where $\prod$ denotes the bit-wise concatenation operation, and $\mathbf{k}_j^r$ and $\mathcal{R}_j^r(n)$ contain the original counterparts of virtual users in $\bar{\mathbf{k}}_j^r$ and $\bar{\mathcal{R}}_j^r(n)$, respectively. Also, $\mathbf{P}_j^r$ and α are the *packet index matrix* and *stretch factor* for the elevation process, respectively.

The process to create $\mathbf{P}_j^r$ will be explained shortly after. The need for the α parameter stems from the fact that each virtual user $\bar{k}$ appears precisely $(\bar{t}+L)(K-\bar{t})$ times in the virtual vectors. On the other hand, its original counterpart needs $K-t_k$ packets to decode its FoV. Thus, to uniformly map the needed packets of user k into transmitted data of its counterpart in the virtual network, we need to divide each packet into $\hat{D}_k = \frac{(\bar{t}+L)(K-\bar{t})}{K-t_k}$ subpackets. However, $\hat{D}_k$ is not necessarily integer for all $k\in\mathcal{K}$. Therefore, α is considered to guarantee that $D_k = \alpha\hat{D}_k$ is an integer for all k. Note that the total number of delivered subpackets to user k is equal to $D_T = (K-t_k)D_k = \alpha(\bar{t}+L)(K-\bar{t})$. The elevation process in (7) implies that every subpacket in $\bar{\mathbf{x}}_j^r$ is replaced by the bit-wise concatenation of α subpackets in $\mathbf{x}_j^r$, i.e.,

$$\bar{W}^{\bar{q}}_{\bar{\mathbf{P}}_j^r[n]} \rightarrow \prod_{m\in[\alpha]} W^q_{\mathbf{P}_j^r[n,m]}. \tag{8}$$

The subpacket indices $\mathbf{P}_j^r[n,m]$ are designed to ensure that 1) all the missing data parts are delivered to the users, and 2) the cache-aided interference cancellation is done properly. Prior to build $\mathbf{P}$ matrix, we form K user-specific File-Mapping (FM) matrices $\mathbf{F}_k, \forall k\in[K]$, with size $K\times K$, to map the actual missing subpackets to their virtual counterparts. The matrix elements are denoted by $f_{i,j,k}$, where, $f_{i,j,k}$ belongs to the i'th row and j'th column of matrix $\mathbf{F}_k$. The element $f_{i,j,k}$, indicates how many subpackets of $W_j(s_k)$ must be allocated to virtual packet $\bar{W}_{\bar{i}}(\bar{k})$. Then, $\mathbf{P}_j^r[n,m]$ is build using the process outlined in Algorithm 1. In a nutshell, a subpacket of $W_\omega(s_{\mathbf{k}_j^r[n]})$ is assigned to $\mathbf{P}_j^r[n,m]$ if $f_{\mathbf{p}_j^r[n],\omega,\mathbf{k}_j^r[n]}$ is positive and $f_{\mathbf{p}_j^r[n],\omega,\mathbf{k}_j^r[n]}$ is subtracted by 1 for the following turns.

To form the FM matrices, we consider the following statements; however, the detailed solution is left for the extended version of this paper.

1) The different packets delivered to user $\bar{k}$ in the virtual network constitute a dedicated index set (DIS) $\mathcal{I}_k$, where $|\mathcal{I}_k| = K-\bar{t}$. Therefore, the elements in those rows which are not included in DIS are all set to zero, i.e., $\{f_{i,j,k}, \forall k\in[K], \forall i\in[K]\backslash\mathcal{I}_k, \forall j\in[K]\}$. On the other hand, based on [9], each packet in $\mathcal{I}_k$ appears $\bar{t}+L$ times during the K rounds. Thus, from D_T missing subpackets of user k, $\frac{D_T}{K-\bar{t}} = \alpha(\bar{t}+L)$ subpackets will be delivered by each virtual packet in $\mathcal{I}_k$ during K rounds, i.e., $\sum_{j\in[K]} f_{i,j,k} = \alpha(\bar{t}+L),\ \forall k, \forall i\in\mathcal{I}_k$.

Algorithm 1 Packet Index Matrix

function INDEX-GENERATOR($\{\mathbf{F}_k\}, \{\mathbf{p}_j^r\}, \{\mathbf{k}_j^r\}$)
  for all $r\in[K]$ **do**
    for all $j\in[K-\bar{t}]$ **do**
      for all $n\in[\bar{t}+L]$ **do**
        $\chi \leftarrow \mathbf{p}_j^r[n]$
        $\psi \leftarrow \mathbf{k}_j^r[n]$
        for all $m\in[\alpha]$ **do**
          for all $\omega\in[K]$ **do**
            if $f_{\chi,\omega,\psi} > 0$ **then**
              $\mathbf{P}_j^r[n,m] \leftarrow \omega$
              $f_{\chi,\omega,\psi} \leftarrow f_{\chi,\omega,\psi} - 1$
              Break;
  return $\{\mathbf{P}_j^r\}$

2) Following the cache placement proposed in section III, file $W(s_k)$ is divided to K packets, from which t_k ones are available in the cache memory of user k in state s_k. As a result, $K-t_k$ missing packets should be delivered to user k, which constitute a requested index set (RIS) $\mathcal{N}_k$, where $|\mathcal{N}_k| = K-t_k$. Thus, the elements in those columns which are not included in RIS are all set to zero, i.e., $\{f_{i,j,k}, \forall k\in[K], \forall j\in[K]\backslash\mathcal{N}_k, \forall i\in[K]\}$. Also, to make sure that all D_T missing subpackets are delivered to each user, the sum of the elements in those columns belonging to RIS must be equal to D_k, i.e., $\sum_{i\in[K]} f_{i,j,k} = D_k,\ \forall k, \forall j\in\mathcal{N}_k$.

Example 3. *Consider the network in Example 2. Denoting the set of requested packets for user k with $\mathcal{M}_k$ and assuming $A\equiv W(1)$, $B\equiv W(2)$, $C\equiv W(4)$, $D\equiv W(5)$, we have $\mathcal{M}_1 = \{A_2, A_3, A_4\}$, $\mathcal{M}_2 = \{B_3, B_4\}$, $\mathcal{M}_3 = \{C_4\}$, $\mathcal{M}_4 = \{D_1, D_2\}$. As the result, the RIS of each user is $\mathcal{N}_1 = \{2,3,4\}$, $\mathcal{N}_2 = \{3,4\}$, $\mathcal{N}_3 = \{4\}$ and $\mathcal{N}_4 = \{1,2\}$. Note that, the size of the packets of A, B, C, D are $\frac{1}{4}$ data units; however, the number of needed packets for each user (i.e., $|\mathcal{M}_k| = K-t_k$) is different.*

Since the common coded caching gain is $\bar{t}=1$, based on [9], the corresponding DISs are $\mathcal{I}_1 = \{2,3,4\}$, $\mathcal{I}_2 = \{1,3,4\}$, $\mathcal{I}_3 = \{1,2,4\}$ and $\mathcal{I}_4 = \{1,2,3\}$. Based on the delivery procedure, the requested packets of user k are divided into D_k subpackets, where $D_1 = 6$, $D_2 = D_4 = 9$, $D_3 = 18$, and α is equal to 2. It is clear that there exist $D_T = D_k(K-t_k) = 18$ subpackets to be transmitted to each user. Based on the delivery procedure, in each transmission, we target $\bar{t}+L = 3$ users, where for each user in the targeted group a data comprised of α subpackets is transmitted. As the result, the size of the transmitted data intended for user $1,2,3$ and 4 are equal to $\frac{1}{12}, \frac{1}{18}, \frac{1}{36}$ and $\frac{1}{18}$ data unit, respectively. Note that the size of the intended data towards each user is proportional to the approximated rate of its location.

Now, let us review how the downlink message $\mathbf{x}_j^r$ is built from $\bar{\mathbf{x}}_j^r$. First, we solve the FM matrices $\{\mathbf{F}_1, \ldots, \mathbf{F}_4\}$ to map

the RISs $\{\mathcal{N}_k, \forall k \in [K]\}$ *in to their corresponding DISs, i.e.,* $\{\mathcal{I}_k, \forall k \in [K]\}$. *The result is as follows*

$$\mathbf{F}_1 = \begin{pmatrix} 0&0&0&0\\ 0&6&0&0\\ 0&0&6&0\\ 0&0&0&6 \end{pmatrix}, \mathbf{F}_2 = \begin{pmatrix} 0&0&0&6\\ 0&0&0&0\\ 0&0&6&0\\ 0&0&3&3 \end{pmatrix}, \mathbf{F}_3 = \begin{pmatrix} 0&0&0&6\\ 0&0&0&6\\ 0&0&0&0\\ 0&0&0&6 \end{pmatrix}, \mathbf{F}_4 = \begin{pmatrix} 6&0&0&0\\ 3&3&0&0\\ 0&6&0&0\\ 0&0&0&0 \end{pmatrix}.$$

Accordingly, based on FM matrices, the packet index matrix $\mathbf{P}_j^r$ *for the first round is formed as follows*

$$\mathbf{P}_1^1 = \begin{pmatrix} 2&2\\ 4&4\\ 4&4 \end{pmatrix}, \mathbf{P}_2^1 = \begin{pmatrix} 3&3\\ 4&4\\ 1&1 \end{pmatrix}, \mathbf{P}_3^1 = \begin{pmatrix} 4&4\\ 1&1\\ 4&4 \end{pmatrix}.$$

Therefore, based on packet index matrix $\mathbf{P}_j^r$ *and equation* (7)*, the first transmissions of the first round is*

$$\mathbf{x}_1^1 = \mathbf{w}_3 \prod(A_2^1, A_2^2) + \mathbf{w}_3 \prod(B_4^1, B_4^2) + \mathbf{w}_2 \prod(C_4^1, C_4^2)\ ,$$

where $\prod(A, B)$ *denotes the bit wise concatenation of subpackets* A *and* B.

Then, the corresponding received signal at user 1 *is*

$$y_1 = \underline{\prod(A_2^1, A_2^2)\mathbf{h}_1^H \mathbf{w}_3} + \underline{\prod(B_4^1, B_4^2)\mathbf{h}_1^H \mathbf{w}_3} + \underline{\prod(C_4^1, C_4^2)\mathbf{h}_1^H \mathbf{w}_2} + z_1,$$

where z_1 *denotes the receiver noise at user* 1. *Recall that file* B *and* C *correspond to the content of states* 2 *and* 3. *Therefore, based on the placement matrices* V_2 *and* V_3*, subfiles* B_4 *and* C_4 *are available in the cache memory of user* 1. *As the result, by estimating* $\mathbf{h}_1^H \mathbf{w}_{\mathcal{R}_k}$*, the underlined terms can be removed from the received signal and* $\prod(A_2^1, A_2^2)$ *can be decoded interference free [9]. Moreover, by assuming zero-forcing (ZF) beamformers, i.e.,* $\mathbf{h}_i^H \mathbf{w}_{\mathcal{R}_k} = 0$ *if* $i \in \mathcal{R}_k$*, the received signal at users* 2 *and* 3 *is as follows*

$$y_2 = \prod(B_4^1, B_4^2)\mathbf{h}_2^H \mathbf{w}_3 + \underline{\prod(A_2^1, A_2^2)\mathbf{h}_2^H \mathbf{w}_3} + Z_2\ ,$$
$$y_3 = \prod(C_4^1, C_4^2)\mathbf{h}_3^H \mathbf{w}_2 + Z_3\ .$$

Following similar argument as above, users 2 *and* 3 *can decode their intended data interference free. The rest of the transmitted messages can be created following similar steps. Note that we have assumed ZF beamforming for ease of exposition. The optimal beamformers are discussed in section V.*

It can be shown that the proposed cache placement and delivery algorithm, delivers all the required data to all the users with degrees of freedom (DoF) equal to $\bar{t}+L$. However, we leave the details for the extended version of this paper.

V. Weighted Max Min Beamforming

As illustrated in examples 2 and 3, in each transmission interval, a group of users $\mathcal{U}$ with size $|\mathcal{U}| = \bar{t}+L$ is targeted by the base station to be served. In this regard, $\bar{t}+L$ total messages X_k are transmitted (one for each user $k \in \mathcal{U}$), where each message X_k is precoded with beamforming vector $\mathbf{w}_{\mathcal{R}_k}$. Now, based on the delivery procedure, the message X_k contains α subpackets with the size $\frac{F(K-t_k)}{\alpha K(K-\bar{t})(\bar{t}+L)}$. Hence, the required time to transmit X_k to user k is equal to $\frac{F(K-t_k)}{K(K-\bar{t})(\bar{t}+L)r_k}$, where r_k is the dedicated transmission rate to user k.

Since we aim to achieve minimum delivery time to serve all the users in $\mathcal{U}$, we design $\{r_k, \forall k \in \mathcal{U}\}$ such that $\frac{|X_k|}{r_k} = \frac{|X_{k'}|}{r_{k'}}$ for all $(k, k') \in \mathcal{U}$ and $k \neq k'$. Thus, the optimal beamformers $\{\mathbf{w}_{\mathcal{R}_k}, k \in \mathcal{U}\}$, are designed by solving the following

$$\begin{aligned} &\max_{\{\mathbf{w}_{\mathcal{R}_k}, r_k\}} \min_{k\in\mathcal{U}} \frac{r_k}{|X_k|}\\ &\text{subject to}\\ &r_k \leq \log\left(1 + \frac{|\mathbf{h}_k^{\mathrm{H}}\mathbf{w}_{\mathcal{R}_k}|^2}{\sum_{\bar{k}\in\mathcal{U}, k\in\mathcal{R}_{\bar{k}}} |\mathbf{h}_k^{\mathrm{H}}\mathbf{w}_{\mathcal{R}_{\bar{k}}}|^2 + N_0}\right), \quad \forall k \in \mathcal{U},\\ &\textstyle\sum_{k\in\mathcal{U}} \|\mathbf{w}_{\mathcal{R}_k}\|^2 \leq P_T, \end{aligned} \tag{9}$$

where P_T is the total transmit power at the transmitter. Problem (9) is quasi-convex, and it can be efficiently solved using iterative methods such as successive convex approximation (SCA) or uplink-downlink duality [9]. The detailed solution for the optimization problem (9) is left for the extended version of this paper. It is worth noting that compared to [9], the max-min problem in [9] is now changed to a weighted max-min problem in (9). As a result, we can avoid wasting wireless resources by the users in bad channel conditions.

Remark 1. Let us denote $T_{p,q} := \max\{\frac{|X_k|}{r_k}\}$, $\forall p \in [K], \forall q \in [K-\bar{t}]$, as the transmission time of the q'th transmission of p'th round, where r_k is computed in (9). Then, the total required time to deliver all the messages to all the users can be expressed as $T_T = \sum_{p\in[K]} \sum_{q\in[K-\bar{t}]} T_{p,q}$. Now, based on (9), at the optimal point, we have $\frac{|X_k|}{r_k} = \frac{|X_{k'}|}{r_{k'}}$ for all $(k, k') \in \mathcal{U}$ and $k \neq k'$. Let us denote $R_{\text{WMM}} := \frac{r_k}{1-m(s_k)}$ as the common weighted rate. As a result, $T_{p,q}$ can be expressed as $T_{p,q} = \frac{F}{(K-\bar{t})(\bar{t}+L)R_{\text{WMM}}}$. Now, assuming R_{WMM} is almost the same for any group of targeted users $\mathcal{U}$, we can Approximate T_T as $T_T = \frac{KF}{(\bar{t}+L)R_{\text{WMM}}}$. Consequently, defining $R_{\text{WMM}}^{\text{sym}}$ as $R_{\text{WMM}}^{\text{sym}} := \frac{KF}{T_T}$ results in $R_{\text{WMM}}^{\text{sym}} = (\bar{t}+L)R_{\text{WMM}}$. Following similar arguments, we can express $R_{\text{MM}}^{\text{sym}} = (t+L)R_{\text{MM}}$ as the symmetric rate for uniform memory loading case proposed in [9], where $t = \frac{KM}{S}$, $R_{\text{MM}} = \frac{\bar{r}_k}{1-M/S}$, and $\bar{r}_k$ is the common max-min rate for the symmetric cache placement case in [9].

VI. Numerical results

For simulations, we consider a bounded environment divided into $S = 100$ states, where the channel at state $s \in \mathcal{S}$ is assumed to be Rayleigh fading with zero mean and α_s variance, i.e., $\mathbf{h}_s \sim \mathbb{CN}(0, \alpha_s \mathbf{I})$. To simplify the analysis, we assume a very simple binary attenuation model where a multi-antenna transmitter is able to provide a uniform channel quality throughout the environment except for B states that are highly attenuated (e.g., located behind obstacles like walls, see Fig. 1). Therefore, in our simulations $\alpha_s = 1$ for non-attenuated states and $\alpha_s = \beta$ for the attenuated ones. At each time-slot (realization), user k is placed in location s with uniform probability, i.e., $p_s = \frac{1}{S}$. We compare different methods based on the symmetric rate, defined as $\frac{K}{T_T}$.

We consider three benchmark schemes, as the following, 1) ***unicasting*** : the cache placement phase is only comprised of memory allocation procedure. The transmission phase is done by serving L users at a time following traditional unicasting scheme, where each user is served by a normalized data size $1-m(s_k)$. 2) ***unicasting uniform***: the cache placement is only

comprised of uniform memory loading, i.e., $m(j) = \frac{M}{S}$, $\forall j \in \mathcal{S}$. The transmission phase is done by serving L users at a time following traditional unicasting scheme, where each user is served by a normalized data size $1 - \frac{M}{S}$. 3) ***Linear CC*** [9]: the cache placement and transmission schemes follow the same as in [9] with $t = \frac{KM}{S}$, i.e., without memory loading.

In Fig. 3, we have compared the proposed location-dependent scheme with those mentioned above for different number of attenuated slots (B). When the number of blocked states grows, the traditional CC scheme proposed in [9] performs poorly as it is limited by the users located in the blocked slots. This is because the probability of finding a user in an attenuated state grows higher with the number of attenuated states. Hence, likely there exist an ill-conditioned user in each transmission, limiting the transmission rate. Thus, even performing memory loading without considering coded transmission performs better than [9]. Such a limited performance is completely avoided by the proposed scheme.

In Fig. 4, the same comparison is made for different attenuation values (β). It can be seen that for a small attenuation level, the traditional caching scheme of [9] outperforms other methods. This is because in the proposed scheme, we sacrifice the global caching gain, i.e., $t = \frac{KM}{S}$, for higher local gain, i.e., m_i, which results in a higher transmission rate ($R^{\text{sym}}_{\text{WMM}}$) compared to [9]. Now, comparing the symmetric rate of [9] with the proposed scheme $\frac{R^{\text{sym}}_{\text{WMM}}}{R^{\text{sym}}_{\text{MM}}} = \frac{(\bar{t}+L)R_{\text{WMM}}}{(t+L)R_{\text{MM}}}$, we can see that when the attenuation level is low, the performance loss due to the DoF decrements (i.e., $\frac{\bar{t}+L}{t+L}$) is more than the rate improvement (i.e., $\frac{R_{\text{WMM}}}{R_{\text{MM}}}$). However, as the attenuation level increases, the effect of the worst user case becomes more severe, and the fraction $\frac{R_{\text{WMM}}}{R_{\text{MM}}}$ grows much larger than $\frac{\bar{t}+L}{t+L}$, and the performance gap becomes noticeable.

VII. Conclusion

This paper proposes a centralized Multi-antenna location-dependent coded caching scheme tailored for future immersive viewing applications. Two-step cache placement was proposed. First, a memory allocation process was carried out to allocate larger cache portions where the channel condition is poor. Next, data placement was performed for each state independently. Finally, weighted-max-min optimization was considered for beamforming, which supported different data rates within a single transmission. The resulting scheme outperforms state-of-the-art in ill-conditioned scenarios where the ratio between the best and worst channel conditions is large.

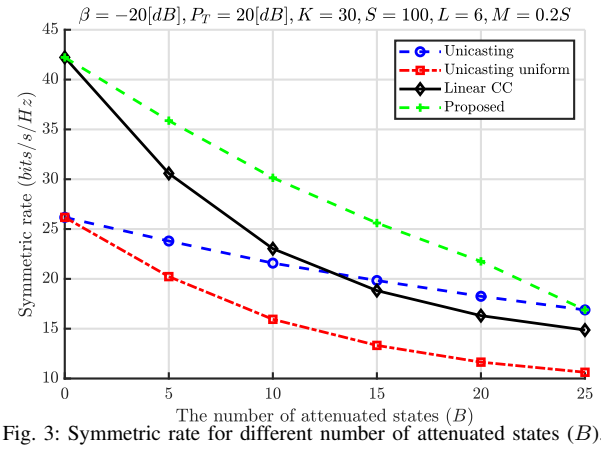


Fig. 3: Symmetric rate for different number of attenuated states (B).

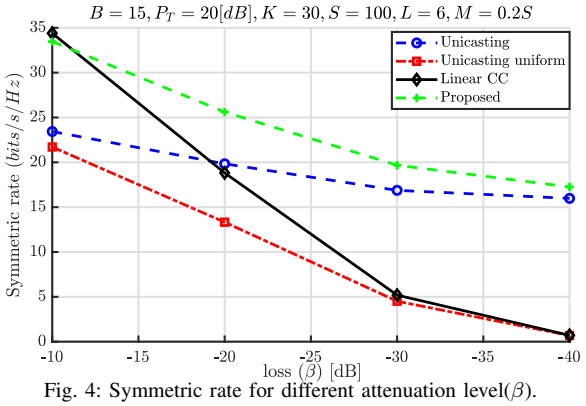


Fig. 4: Symmetric rate for different attenuation level(β).

References

[1] L. Han, S. Appleby, and K. Smith, "Problem statement: Transport support for augmented and virtual reality applications," *Working Draft, IETF Secretariat, Internet-Draft draft-haniccrg-arvr-transport-problem-00, March*, 2017.

[2] E. Bastug, M. Bennis, M. Médard, and M. Debbah, "Toward interconnected virtual reality: Opportunities, challenges, and enablers," *IEEE Communications Magazine*, vol. 55, no. 6, pp. 110–117, 2017.

[3] E. Bastug, M. Bennis, and M. Debbah, "Living on the edge: The role of proactive caching in 5G wireless networks," *IEEE Communications Magazine*, vol. 52, no. 8, pp. 82–89, 2014.

[4] Y. Sun, Z. Chen, M. Tao, and H. Liu, "Communications, caching, and computing for mobile virtual reality: Modeling and tradeoff," *IEEE Transactions on Communications*, vol. 67, no. 11, pp. 7573–7586, 2019.

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

[6] S. P. Shariatpanahi, S. A. Motahari, and B. H. Khalaj, "Multi-server coded caching," *IEEE Transactions on Information Theory*, vol. 62, no. 12, pp. 7253–7271, 2016.

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

[8] E. Lampiris and P. Elia, "Adding transmitters dramatically boosts coded-caching gains for finite file sizes," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 6, pp. 1176–1188, 2018.

[9] M. J. Salehi, E. Parrinello, S. P. Shariatpanahi, P. Elia, and A. Tölli, "Low-complexity high-performance cyclic caching for large miso systems," *IEEE Transactions on Wireless Communications*, pp. 1–1, 2021.

[10] A. Destounis, A. Ghorbel, G. S. Paschos, and M. Kobayashi, "Adaptive Coded Caching for Fair Delivery over Fading Channels," *IEEE Transactions on Information Theory*, 2020.

[11] M. Salehi, A. Tolli, and S. P. Shariatpanahi, "Coded Caching with Uneven Channels: A Quality of Experience Approach," in *2020 IEEE International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2020, pp. 1–5.

[12] A. Tang, S. Roy, and X. Wang, "Coded caching for wireless backhaul networks with unequal link rates," *IEEE Transactions on Communications*, vol. 66, no. 1, pp. 1–13, 2017.

[13] H. B. Mahmoodi, M. Salehi, and A. Tölli, "Non-symmetric coded caching for location-dependent content delivery," in *2021 IEEE International Symposium on Information Theory (ISIT)*, 2021, pp. 712–717.

[14] H. B. Mahmoodi, B. Gouda, M. Salehi, and A. Tolli, "Low-complexity multicast beamforming for multi-stream multi-group communications," *arXiv preprint arXiv:2105.09705*, 2021.

[15] M. Salehi, H. B. Mahmoodi, and A. Tölli, "A low-subpacketization high-performance mimo coded caching scheme," *arXiv preprint arXiv:2109.10008*, 2021.