# An Efficient Fair Content Delivery Scheme for Coded Caching

Asma Ghorbel[1], Apostolos Destounis[2], Mari Kobayashi[1], Georgios Paschos[2]
[1]Centrale-Supélec,
email: firstname.lastname@centralesupelec.fr
[2] France Research Center, Huawei Technologies Co. Ltd.,
email: firstname.lastname@huawei.com

***Abstract*—Coded caching has emerged as one of the promising solutions to deal with an exponentially increasing video traffic. This breakthrough builds on a careful design of file placement such that the total transmission time is minimized by multicasting sub-files simultaneously useful to many users. A number of follow-up works recently studied the extension of coded caching, initially assuming a perfect bottleneck link, to practical wireless channels. However, most of existing works address either the scalability of the cached wireless networks by assuming that each user requests a single file or the dynamic arrival of user requests by ignoring the scalability. In this work, we propose a low-complexity gradient- based scheduling that exploits multicast opportunities offered by coded caching, while keeping a number of multicast groups linear in the number of users. Simulation results illustrate that the proposed algorithm outperforms both coded caching and opportunistic scheduling over time-varying fading channels.**

***Index Terms*—Coded caching, fairness, opportunistic scheduling.**

## I. Introduction

The proliferation of connected devices is continuously leading to a dramatic traffic expansion in wireless networks. It is expected that the majority of the traffic will be dominated by the video [1]. Given the relative scarcity of radio resources, it has become then crucial for the network operators to make as much efficient use of them as possible.

The standard approach to handle this traffic increase while maintaining some fairness across the users is opportunistic scheduling. That is, a transmitter serves the users who enjoy the peak of fading in order to exploit the temporal variations across users [2], [3]. In practice, most 3G, 3.5G and 4G systems use variations of the proportional fair scheduler. However, such techniques, which do not exploit the memory at user devices, achieves the per-user throughput scaling of $\log\log(K)/K$ as the number $K$ of users grows, therefore the per-user throughput tends to vanish in the regime of a large number of users.

Recently, a new breakthrough called *coded caching* has been proposed in the seminal work [4]. This technique exploits the memory at user devices by pre-storing parts of contents prior to actual requests, and then sending appropriate combinations of files to the receivers, as a function the actual demands and the contents of users' caches. The proposed scheme enables to potentially minimize the number of transmissions needed to satisfy user requests and thus achieve a constant per-user throughput under some assumptions [4]. This striking result has motivated follow-up works addressing the fundamental memory-rate tradeoff by relaxing the assumptions made in [4] as well as practical applications (e.g. CADAMI [5] and by Alcatel-Lucent [6]). Although a number of recent works considered coded caching over wireless channels (see e.g. [7]–[10] and references therein), these works address either the scalability of the cached wireless networks by assuming infinite user requests [8], [10], [11] or the online nature of user requests by ignoring the scalability [7]. In fact, the work [7] proposed an online delivery scheme that combines admission control, routing, and scheduling based on decentralized coded caching [12]. Interestingly, it was revealed that it is possible to benefit from both the opportunism of the wireless channels and the multicast opportunities offered by coded caching. Albeit theoretically interesting, its applicability is hindered by the following issues: (i) the server needs to convey sub-files to all possible multicast groups of receivers. The number of multicast groups grows exponentially with the number of users; (ii) the proposed delivery scheme builds on superposition encoding that achieves the capacity of the Gaussian broadcast channel. Such a scheme requires each user to perform successive interference cancellation and cannot be easily implemented in practical systems.

The current work aims to precisely overcome these two limitations. Namely, we propose an efficient delivery scheme based on standard gradient based scheduling [13] in Section IV. The novelty relies on a carefully selected multicast groups, whose number grows only linearly with the number of users. This results in a low complexity scheme, which can be easily implemented with only slight modifications of existing practical schedulers. Numerical results in Section V illustrate that this simplified scheme retains most of the benefits of combining opportunistic transmissions and file combinations of the work in [7], outperforming baseline schemes that use only opportunistic scheduling or standard coded caching.

## II. System Model

### A. Channel Model

We consider a content delivery network where a server (or a base station) wishes to convey (possibly different) files to $K$ user terminals over a wireless channel. The wireless channel is modeled by a standard block-fading broadcast channel, such

that the channel state remains constant over a slot of $n$ channel uses and changes from one slot to another as an independent and identical distributions. The channel output of user $k$ at slot $t$ is given by

$$\boldsymbol{y}_k(t) = \sqrt{h_k(t)}\boldsymbol{x}(t) + \boldsymbol{\nu}_k(t), \tag{1}$$

where the channel input $\boldsymbol{x} \in \mathbb{C}^n$ is subject to the power constraint $\mathbb{E}[\|\boldsymbol{x}\|^2] \leq Pn$; $\boldsymbol{\nu}_k(t) \sim \mathcal{N}_{\mathbb{C}}(0, \mathbf{I}_n)$ is an additive white Gaussian noise vector with an identity covariance matrix of size $n$, assumed independent of each other; $\{h_k(t) \in \mathbb{C}\}$ are fading gain coefficients independently distributed across time. At each slot $t$, the channel state $\boldsymbol{h}(t) = (h_1(t), \ldots, h_K(t))$ is perfectly known to the base station while each user knows its own channel realization. Without loss of generality we assume $\mathbb{E}[h_1] \geq \mathbb{E}[h_2] \geq .. \geq \mathbb{E}[h_K]$.

We follow the network model considered in [4] as well as its follow-up works. The server has an access to $N$ equally popular files, each $F$ bits long, while each user $k$ is equipped with cache memory of $MF$ bits, where $M \in \{0, 1, \ldots, N\}$. We restrict ourselves to decentralized cache placement [12]. That is, each user $k$ independently caches a subset of $\frac{MF}{N}$ bits of file $i$, chosen uniformly at random for $i = 1, \ldots, N$, under its memory constraint of $MF$ bits. For later use, we let $m = \frac{M}{N}$ denote the normalized memory size.

**Example:** Consider a three-user example with three files denoted by $A$, $B$ and $C$. We let $A_{\mathcal{J}}$ denote the sub-file of $A$ stored exclusively in the cache memories of the users in the subset $\mathcal{J} \subseteq \{1, .., K\}$. For an arbitrarily large file size $F$, the size of sub-file $A_{\mathcal{J}}$ measured in bits is given by

$$|A_{\mathcal{J}}| = m^{|\mathcal{J}|}(1-m)^{K-|\mathcal{J}|}F. \tag{2}$$

Let us assume that user 1, 2, 3, requests file $A, B, C$, respectively. After the decentralized placement phase [12], a given file $A$ will be partitioned into 8 sub-files, one per user subset e.g. $A = \{A_\emptyset, A_1, A_2, A_3, A_{\{1,2\}}, A_{\{2,3\}}, A_{\{1,3\}}, A_{\{1,2\,3\}}\}$. In order to satisfy three users' requests, the server sends the following codewords: $A_\emptyset$, $B_\emptyset$ and $C_\emptyset$ to user 1, 2 and 3, respectively; $A_2 \oplus B_1$ to users $\{1, 2\}$; $B_3 \oplus C_2$ to users $\{2, 3\}$; $A_3 \oplus C_1$ to users $\{1, 3\}$ and $A_{23} \oplus B_{13} \oplus C_{12}$ to users $\{1, 2, 3\}$.

### B. Fair file delivery

The performance metric is the *time average delivery rate of files* to user $k$, denoted by $\overline{r}_k$. We let $\Lambda$ denote the set of all feasible delivery rate vectors.

**Definition 1** (Feasible rate)**.** *A rate vector $\overline{\boldsymbol{r}} = (\overline{r}_1, \ldots, \overline{r}_K)$, measured in file/slot, is said to be feasible $\overline{\boldsymbol{r}} \in \Lambda$ if there exists a file combining and transmission scheme such that*

$$\overline{r}_k = \liminf_{t\to\infty} \frac{D_k(t)}{t}. \tag{3}$$

*where $D_k(t)$ denotes the number of successfully delivered files to user $k$ up to $t$.*

We are interested in the *fair file delivery* problem:

$$\overline{\boldsymbol{r}}^* = \arg\max_{\overline{\boldsymbol{r}}\in\Lambda} \sum_{k=1}^{K} g(\overline{r}_k), \tag{4}$$

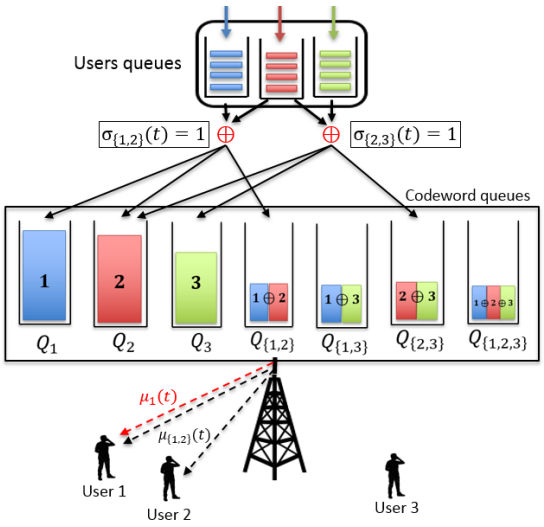


Fig. 1. The near-optimal scheme with $2^K - 1$ codeword queues [7].

where the utility function corresponds to the *alpha fair* family of concave functions obtained by choosing:

$$g(x) = \begin{cases} \frac{(d+x)^{1-\alpha}}{1-\alpha}, \alpha \neq 1 \\ \log(1 + x/d), \alpha = 1 \end{cases} \tag{5}$$

for some arbitrarily small $d > 0$ (used to extend the domain of the functions to $x = 0$). Tuning the value of $\alpha$ changes the shape of the utility function and consequently drives the system performance $\overline{\boldsymbol{r}}^*$ to different points: (i) $\alpha = 0$ yields max sum delivery rate, (ii) $\alpha \to \infty$ yields max-min fair delivery rate [11], [14], (iii) $\alpha = 1$ yields proportionally fair delivery rate [15]. Choosing $\alpha \in (0, 1)$ leads to a tradeoff between max sum and proportionally fair delivery rates.

An $\epsilon$-approximate solution of (4) was proposed in [7]. However, the proposed scheme utilizes $2^K - 1$ codeword queues, each of which contains sub-files intended to a distinct subset of users. Unfortunately, the proposed scheme cannot be implemented in practice due to an exponentially increasing number of codeword queues as $K$ grows. In this paper, we propose a novel delivery scheme based on a standard gradient scheduling that reduces the number of queues from $2^K - 1$ to $2K - 1$.

## III. Motivating Example

We recall briefly the near-optimal delivery scheme proposed in [7] that requires $2^K - 1$ codeword queues and provide a motivating example through Fig. 1. The near-optimal delivery scheme of [7] consists of three taks, namely admission control, codeword routing, and scheduling. We focus on last two parts which are relevant to the current work.

*a) Codeword routing:* At each slot $t$, the transmitter chooses the subsets of users to perform coded caching, generates codewords by combining files from user queues, and places them to appropriate codewords queues. We denote $\sigma_{\mathcal{J}}(t) = 1$ if subset $\mathcal{J}$ is chose at slot $t$ and zero otherwise. In Fig. 1, we depict the decision $\sigma_{\{1,2\}}(t) = \sigma_{\{2,3\}}(t) = 1$, implying that coded caching is performed overs users $\{1, 2\}$ and $\{2, 3\}$ simultaneously. Note that we have $2^K - 1$ decision variables $\sigma_{\mathcal{J}}$.

*b) Scheduling and resource allocation:* The transmitter decides the transmission rate $\mu_{\mathcal{J}}(t)$ to serve codeword queue $\mathcal{J}$. Namely, the following weighted sum rate maximization is solved at each slot $t$:

$$\max_{\boldsymbol{\mu}\in\Gamma(\boldsymbol{h}(t))} \sum_{\mathcal{J}\subseteq\{1,\dots,K\}} Q_{\mathcal{J}}(t)\mu_{\mathcal{J}}, \tag{6}$$

where the maximization is over the capacity region $\Gamma(\boldsymbol{h}(t))$ for a fixed channel realization $\boldsymbol{h}(t)$ achieved by superposition encoding; $Q_{\mathcal{J}}(t)$ is the length of codeword queue intended to user subset $\mathcal{J}$.

In Fig. 1, codeword queues $Q_1$ and $Q_2$ are served with rate $\mu_1$ and $\mu_{12}$, respectively so that user 1 receives its uncoded sub-file and both users 1 and 2 receive coded (X-OR) sub-files.

The main drawback of this scheme is that it requires $2^K-1$ codeword queues and it cannot be implemented for a large $K$. This motivates us to reduce the number of codeword queues as we explain below.

To highlight the main idea of this paper, let us first make the -unrealistic- assumption that $h_1(t) \geq h_2(t) \geq h_3(t)$ holds for any time slot $t$. We recall that the multicast capacity of user subset $\mathcal{J}$ is given by $\log(1 + P\min_{i\in\mathcal{J}} h_i(t))$ so that any codeword intended to user subset $\mathcal{J}$ can be reliably transmitted to $\mathcal{J}$ below this capacity for an arbitrarily large $n$. Under the assumption of the channel ordering, $\{Q_2, Q_{\{1,2\}}\}$ are served at most equal to $\log(1 + Ph_2(t))$ while $\{Q_3, Q_{\{1,3\}}, Q_{\{2,3\}}, Q_{\{1,2,3\}}\}$ are served at most equal to $\log(1 + Ph_3(t))$. Hence, we can merge these 7 codeword queues into 3 codeword queues as follows: $Q'_1 = Q_1$; $Q'_2 = \{Q_2, Q_{\{1,2\}}\}$; $Q'_3 = \{Q_3, Q_{\{1,3\}}, Q_{\{2,3\}}, Q_{\{1,2,3\}}\}$.

Consequently, it readily follows that the number of files combination decision parameters $\sigma_{\mathcal{J}}$ can be also reduced from $2^K-1$ to $K$ by noticing that the stronger users $1, 2$ can receive packets for free when user 3 is served. Namely, we can define $\sigma'_k \stackrel{\Delta}{=} \sigma_{\{1,..,k\}}$ for $k = 1, \dots, K$.

Of course, the assumption $h_1(t) \geq h_2(t) \geq h_3(t)$ does not hold for all $t$ over the time-varying channel of consideration. Therefore, the above delivery scheme with $K$ codeword queues may incur a non-negligible loss, when the new codeword queue $Q'_k$ is served at the rate $\log(1 + P\min_{i\leq k} h_i(t))$. This is typically the case for the regime of a small memory size where the new codeword queue $Q'_k$, dominated by uncoded sub-files intended to user $k$, is served with a pessimistic rate associated to the worst user among $\{1, \dots, k\}$. In order to overcome this performance loss, we propose a novel queue structure by adding $K-1$ unicast codeword queues as we explain in the following section.

## IV. Proposed Delivery Scheme

We reduce the number of files combination decision parameters from $2^K-1$ to $2K-1$ denoted by $\{\tilde{\sigma}_k\}_{1\leq k\leq 2K-1}$ such that a new decision $\tilde{\sigma}_k = 1$ implies the following

- For $k \in \{1, .., K\}$: one requested file by user $k$ will be stored in the corresponding codeword queue $\tilde{Q}_k$ to be transmitted to user $k$ uncoded in later slots.
- For $k \in \{K+1, .., 2K-1\}$: we combine files requested by users in $\{1, 2, .., k-K+1\}$ and store the generated codewords in the corresponding codeword queue $\tilde{Q}_k$. Namely, in this case $\tilde{Q}_k$ contains packets intended to all subsets of the set $\{1, 2, 3, ..., k-K+1\}$ which contain user $k-K+1$.

In Fig. 2 we show all possible decision parameters $\{\tilde{\sigma}_k\}_{1\leq k\leq 2K-1}$ of the new scheme for a three users example.

### A. Codeword Queues

We recall that we consider the following order of the channel statistics $\mathbb{E}[h_1] \geq \mathbb{E}[h_2] \geq .. \geq \mathbb{E}[h_K]$. We let $\tilde{r}_k(t)$ denote the service rate of queue $\tilde{Q}_k$ at slot $t$. The codeword queues evolve as follows:

- for $k = 1$

$$\tilde{Q}_1(t+1) = [\tilde{Q}_1(t) - n\tilde{r}_1(t)]^+ + \sum_{i=K+1}^{2K-1} (1-m)^{i-K+1}\tilde{\sigma}_i(t)F$$

- for $k \in \{2, .., K\}$

$$\tilde{Q}_k(t+1) = [\tilde{Q}_k(t) - n\tilde{r}_k(t)]^+ + (1-m)\tilde{\sigma}_k(t)F$$

- for $k \in \{K+1, .., 2K-1\}$

$$\tilde{Q}_k(t+1) = [\tilde{Q}_k(t) - n\tilde{r}_k(t)]^+ + \sum_{i=k}^{2K-1} (1-m)^{i-K+1}\tilde{\sigma}_i(t)F,$$

where the input comes from applying the delivery scheme [12] on the user subset $\{1, .., k\}$ when $\tilde{\sigma}_{k+K-1} = 1$.

**Remark:** For $k \in \{K+1, .., 2K-1\}$, codeword queue $\tilde{Q}_k$ contains different type of packets depending on the exact users that are in the corresponding multicast group of the packet (recall that for $k > K$, $\tilde{Q}_k$ contains packets for all subsets of $\{2, 3, .., k-K+1\}$ which contain user $k-K+1$). We notice that independently on the files combination decision, the fraction of packets useful for user $j$ among packets in queue $\tilde{Q}_k$ is equal to:

$$\begin{cases} 1 & \text{if } \; j = k-K+1 \\ m & \text{if } \; j < k-K+1 \end{cases} \tag{7}$$

In a three-user example in Fig. 2 we observe that the packets in $\tilde{Q}_4$ are all useful for user 2 while only a fraction $m$ of them are useful for user 1. Similarly the packets in $\tilde{Q}_5$ are all useful for user 3, while only a fraction $m$ fraction of them are useful for both users 1 and 2.

### B. Scheduling

Under the assumption of infinite demand, the solution of the maximization problem in (4) is given by gradient scheduling schemes as a straightforward application of the results of [13] which proves the asymptotic optimality of gradient scheduling schemes. Thus the scheduling rule is given by

$$\max \sum_{k=1}^{K} \frac{r_k(t)}{(T_k(t)+d)^{\alpha}}. \tag{8}$$

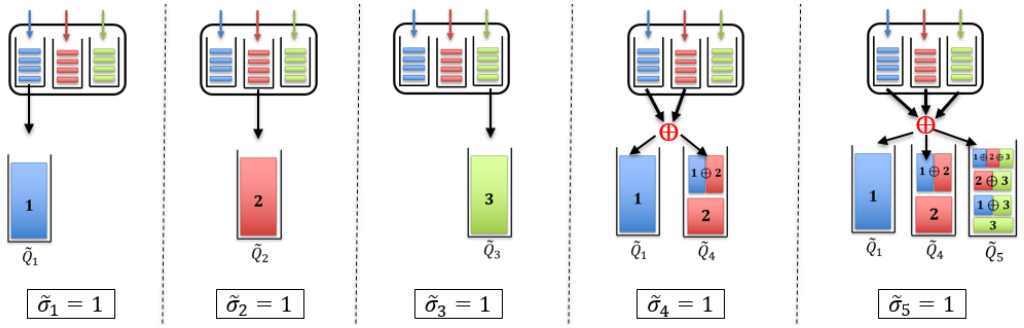


Fig. 2. Files combination decision parameters $\{\tilde{\sigma}_k\}_{1\le k\le 2K-1}$ for the new scheme with $2K-1$ codeword queues (where $\oplus$ denotes coded caching).

In the above, $r_k(t)$ denotes the instantaneous rate of user $k$ at slot $t$ and $T_k(t)$ the empirical average rate of user $k$ up to slot $t$, which obeys the recursive equation:

$$T_k(t+1) = \frac{1}{t+1}\left(tT_k(t) + r_k(t)\right). \tag{9}$$

Note that at each slot $t$ we need to decide on the codeword queue $\tilde{Q}_{k^*}$ to serve with rate $\log(1+h_{k^*}P)$ if $k^* \le K$ and $\log(1+\min_{i\in\{1,2,..,k^*-K+1\}} h_i P)$ if $k^* > K$. Thus we rewrite (8) as a function of $\tilde{r}_k$. For that we provide the relation between the codeword queues service rate and the users rate. Using (7) we obtain

$$r_k(t) = \begin{cases} \tilde{r}_k(t) + m\sum_{i=K+1}^{2K+1} \tilde{r}_i(t) & \text{if} \quad k=1 \\ \tilde{r}_k(t) + m\sum_{i=k+K}^{2K+1} \tilde{r}_i(t) + \tilde{r}_{K+k-1}(t) & \text{if} \quad k>1 \end{cases} \tag{10}$$

Plugging (10), it readily follows that (8) is equivalent to maximizing the following:

$$\sum_{k=1}^{K} \frac{r_k}{T_k^\alpha} = \sum_{k=1}^{K} \frac{\tilde{r}_k}{T_k^\alpha} + m\sum_{k=1}^{K-1}\sum_{i=k+K}^{2K+1} \frac{\tilde{r}_i}{T_k^\alpha} + \sum_{k=2}^{K} \frac{\tilde{r}_{K+k-1}}{T_k^\alpha} \tag{11}$$

$$= \sum_{k=1}^{K} \frac{\tilde{r}_k}{T_k^\alpha} + \sum_{i=K+1}^{2K-1} m\sum_{k=1}^{i-K} \frac{\tilde{r}_i}{T_k^\alpha} + \sum_{k=K+1}^{2K-1} \frac{\tilde{r}_k}{T_{k-K+1}^\alpha} \tag{12}$$

$$= \sum_{k=1}^{K} \frac{\tilde{r}_k}{T_k^\alpha} + \sum_{k=K+1}^{2K-1} \left( m\sum_{i=1}^{k-K} \frac{1}{T_i^\alpha} + \frac{1}{T_{k-K+1}^\alpha} \right) \tilde{r}_i, \tag{13}$$

where we have omitted the time index and the constant $d$ for simplicity. Eq. (12) follows from the fact that $\sum_{k=1}^{K-1}\sum_{i=k+K}^{2K+1} = \sum_{i=K+1}^{2K-1}\sum_{k=1}^{i-K}$. Thus, the scheduling rule is given by selecting the codeword queue $\tilde{Q}_{k^*}$ which satisfies the following

$$k^* = \arg\max_{1\le k\le 2K-1} f(k), \tag{14}$$

where

$$f(k) = \begin{cases} \frac{\tilde{r}_k(t)}{T_k(t)^\alpha}, & \text{if} \quad k \le K \\ \left( m\sum_{i=1}^{k-K} \frac{1}{T_i(t)^\alpha} + \frac{1}{T_{k-K+1}(t)^\alpha} \right) \tilde{r}_k, & \text{if} \quad k > K \end{cases} \tag{15}$$

$$\tilde{r}_k(t) = \begin{cases} \log(1+h_k(t)P) & \text{if} \quad k \le K \\ \log(1+\min_{i\in\{1,2,..,k-K+1\}} h_i(t)P) & \text{if} \quad k > K \end{cases} \tag{16}$$

**Remark:** Note that the well known unicast opportunistic scheduling corresponds to serving the user that maximizes $f(k)$ only for $1 \le k \le K$:

$$\max_{1\le k\le K} f(k) = \max_{1\le k\le K} \frac{\tilde{r}_k(t)}{T_k(t)^\alpha}. \tag{17}$$

The resulting scheduling rule (14) under our proposed scheme is of a very similar form, with the weight $1/T_k(t)^\alpha$ for each multicast group used being replaced by an appropriate sum of such weights corresponding to individual users. This small modification to current gradient based schedulers is therefore enough to make a mobile downlink system implement our proposed scheme. This suggests that making a scheduler work when coded caching is also used can be easy and appealing from a practical perspective as well.

## V. Numerical Examples

In this section, we compare our proposed delivery scheme with the optimal scheme [7] and two other schemes described below, all building on the decentralized cache placement.

- **Unicast opportunistic scheduling**: For any request, the server sends the remaining $(1-m)F$ bits to the corresponding user without combining any files. Here we only exploit the local caching gain. In each slot the transmitter sends with full power to the following user

$$k^*(t) = \arg\max_k \frac{\log\left(1+h_k(t)P\right)}{T_k(t)^\alpha}.$$

- **Standard coded caching**: We apply coded caching delivery scheme [12] on all $K$ users. The server sends the multicasting message at the worst transmission rate. The number of packets to be multicast in order to satisfy one demand for each user is given by [12]

$$T_{\text{tot}}(K,m) = \frac{1}{m}(1-m)\left\{1-(1-m)^K\right\}. \tag{18}$$

Thus the average delivery rate (in file per slot) is symmetric, and given as the following

$$\overline{r}_k = \frac{N}{T_{\text{tot}}(K,m)F}\mathbb{E}\left[\log(1+P\min_{i\in\{1,\ldots,K\}} h_i)\right]. \tag{19}$$

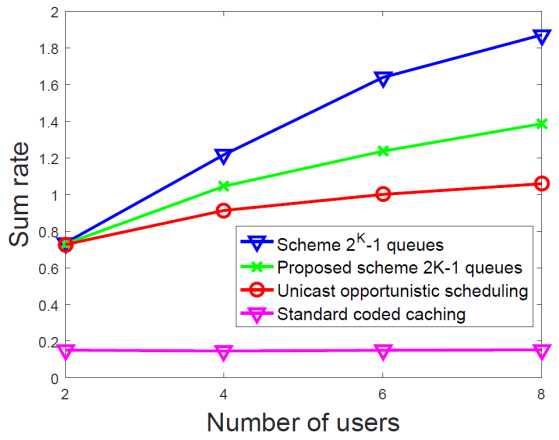


Fig. 3. The sum rate of the system vs. the number of users ($\alpha = 0$).

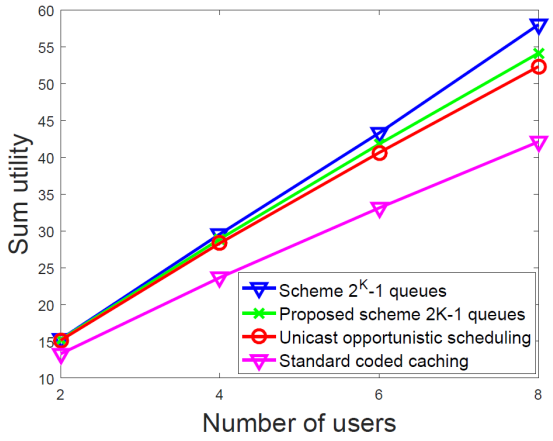


Fig. 4. Utility vs. the number of users under the proportional fairness objective ($\alpha = 1$).

We consider a system with normalized memory of $m = 0.6$, power constraint $P = 10dB$, file size $F = 10^3$ bits and number of channel uses per slot $n = 10^2$. The channel coefficient $h_k(t)$ follows an exponential distribution with mean $\beta_k$. We divide users into two classes: $K/2$ strong users with $\beta_k = 1$ and $K/2$ weak users with $\beta_k = 0.2$. We plot the sum utility versus the number of users, where the objective of the system is sum rate maximization ($\alpha = 0$) and proportional fairness ($\alpha = 1$). The results are depicted in Figs. 3, 4 and 5.

We can see in Figs. 3 and 4 that the proposed low - complexity scheme achieves lower value of each objective than the optimal scheme in [7] as expected: not using all possible multicast groups results to the algorithm being suboptimal. However, it still outperforms the baseline schemes.

As the complexity of the proposed scheme is proportional to the number of users, we are able to plot the sum rate for a large number of users in Fig. 5, which shows that the total system throughput under the proposed scheme still increases with the number of users, while it seems to saturate under the opportunistic scheduling and standard coded caching schemes.

## VI. Conclusions

In this work we proposed a scheduling and codeword generation algorithm for coded caching that tries to combine the channel opportunism and the multicast opportunities offered by the users' cache contents while using only $O((\#\text{ users}))$ multicast groups. The proposed algorithm is easy to implement by performing small modifications on standard gradient-based algorithms and outperforms baseline opportunistic scheduling and coded caching schemes, thus retaining most of the benefits of the (significantly more complex as it needs to use all multicast groups) optimal scheme. Obtaining rigorous results on the suboptimality/performance bounds of the proposed algorithm and/or the relation between performance and the number of groups would be interesting issues for further study.

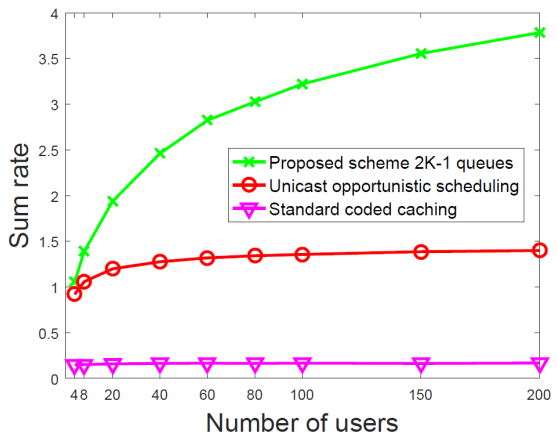


Fig. 5. Scaling of the sum rate of the system vs. the number of users ($\alpha = 0$).

## References


[1] C. V. N. Index, "Global mobile data traffic forecast update, 2015-2020," *Cisco white paper*, 2015.

[2] R. Knopp and P. A. Humblet, "Information capacity and power control in single-cell multiuser communications," in *IEEE ICC*, 1995.

[3] P. Viswanath, D. N. C. Tse, and R. Laroia, "Opportunistic beamforming using dumb antennas," *IEEE Transactions on Information Theory*, vol. 48, no. 6, pp. 1277–1294, Jun 2002.

[4] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Transactions on Information Theory*, vol. 60, no. 5, pp. 2856–2867, 2014.

[5] [Online]. Available: http://www.cadami.com

[6] [Online]. Available: http://www.youtube.com/watch?v=ne__JULcFIU

[7] A. Destounis, M. Kobayashi, G. Paschos, and A. Ghorbel, "Alpha fair coded caching," *IEEE WiOpt*, May 2017.

[8] R. Combes, A. Ghorbel, M. Kobayashi, and S. Yang, "Utility optimal scheduling for coded caching in general topologies," *arXiv preprint arXiv:1801.02594*, 2018.

[9] M. Bayat, R. K. Mungara, and G. Caire, "Achieving spatial scalability for coded caching over wireless networks," *arXiv preprint arXiv:1803.05702*, 2018.

[10] K.-H. Ngo, S. Yang, and M. Kobayashi, "Scalable content delivery with coded caching in multi-antenna fading channels," *IEEE Transactions on Wireless Communications*, vol. 17, no. 1, pp. 548–562, 2018.

[11] S. P. Shariatpanahi, G. Caire, and B. H. Khalaj, "Multi-antenna coded caching," in *IEEE ISIT*, 2017.

[12] M. A. Maddah-Ali and U. Niesen, "Decentralized coded caching attains order-optimal memory-rate tradeoff," *IEEE/ACM Transactions on Networking (TON)*, vol. 23, no. 4, pp. 1029–1040, 2015.

[13] A. L. Stolyar, "On the asymptotic optimality of the gradient scheduling algorithm for multiuser throughput allocation," *Operations research*, vol. 53, no. 1, pp. 12–25, 2005.

[14] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Transactions on Networking (ToN)*, vol. 8, no. 5, pp. 556–567, 2000.

[15] F. Kelly, "Charging and rate control for elastic traffic," *European transactions on Telecommunications*, vol. 8, no. 1, pp. 33–37, 1997.