Coded Caching Under Non-Uniform Content Popularity Distributions with Multiple Requests

Abdollah Ghaffari Sheshjavani*, Ahmad Khonsari*†, Seyed Pooya Shariatpanahi*, Masoumeh Moradian†, and Aresh Dadlani‡

*School of Electrical and Computer Engineering, University of Tehran, Iran

†School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Iran

‡ Department of Electrical and Computer Engineering, Nazarbayev University, Kazakhstan

Emails: {abdollah.ghaffari, a_khonsari, p.shariatpanahi}@ut.ac.ir, mmoradian@ipm.ir, aresh.dadlani@nu.edu.kz

***Abstract*—Content caching is a technique aimed to reduce the network load imposed by data transmission during peak time while ensuring users' quality of experience. Studies have shown that content delivery via coded caching can significantly improve beyond the performance limits of conventional caching schemes when caches and the server share a common link. Finding the optimal cache content placement however, becomes challenging under arbitrary distributions of content popularity. While existing works show that partitioning contents into three popularity levels performs better when multiple requests are received at each time slot, they neither delve into the problem analysis nor derive closed-form expressions for the optimum partitioning problem. In this paper, we analyze the coded caching scheme for a system with arbitrary content popularity, where we derive explicit closed-forms for the server load in the delivery phase and formulate the near-optimum partitioning problem. Simulation results are presented to corroborate our mathematical analysis.**

***Index Terms*—Coded caching, arbitrary popularity distribution, multiple requests, cache content placement, optimization.**

I. Introduction

The growing penetration of high throughput devices witnessed in recent years has facilitated the ever-rising demand for mobile content through wireless media. Deploying small base stations (SBSs) to jointly shrink the network cell size and increase spatial reuse is a promising solution to alleviate this growth and has stimulated many research initiatives [1]. Nonetheless, providing high speed backhaul to connect these SBSs to the core network still poses as the main obstacle in this approach. In the context of content delivery, caching popular contents at these SBSs would relieve the need for high speed backhaul links [2], [3].

In general, conventional caching methods attempt to cache the most popular contents located at close proximity of end-users such that the requests for popular contents are directly served from the local caches. Doing so results in the so-called *local caching gain*, which is proportional to the local memory size. In [4], a novel coded caching scheme showed improved performance over conventional caching by leveraging the multicasting nature of the shared wireless medium even for caches with distinct demands. Their scheme, in addition to local cache gain, also results in *global caching gain* from coded-multicasting opportunities. The global cache gain is proportional to the aggregate memory of all the caches, where every user benefits from its cache contents to decode the desired content and to remove interference in the coded message due to other cache requests. Further extensions of the idea to hierarchical coded caching [5], multi-server coded caching [6], decentralized coded caching [7], hybrid caching [8], and coded caching with multiple file requests [9] have also been reported in the literature.

To increase multicasting opportunities in coded caching, different parts of the library should be cached among different users, i.e. the *diversity principle*. However, in a set-up where content popularity is not uniformly distributed, it is desirable to cache more popular contents with higher frequency, i.e. the *popularity principle*, which makes the cache contents of different users almost the same. As these two principles, namely diversity and popularity, are in tension, cache placement design in such scenarios is very challenging. Although the original coded caching scheme introduced in [4] performs well under uniform popularity of contents, it is inefficient in scenarios with arbitrary content popularities. In view of this shortcoming, the same authors proposed a non-uniform coded caching scheme in [10], where the library is divided into almost equi-probable groups and each user cache is equally shared among these groups. Each group is then treated separately as a single coded caching problem originally proposed in [4]. The efforts in [11]–[14] show that when each cache receives only one request, the asymptotically optimum strategy is to partition the library into two groups: the popular contents are cached according to the scheme in [4], while the non-popular contents are not cached at all.

Scenarios involving multiple requests made by each user have been studied in [9] and [15]. Specifically, the authors in [15] exemplify that when each cache receives multiple requests, partitioning the library altogether into three groups improves caching performance over two partitioning strategies. In their method, the first group is completely cached at all cache memories, the second group of content is cached according to the original coded caching paradigm, and the last group is not cached at all. They however, assume that the library is divided into multiple levels, based on varying degrees of popularity. In addition, the work fails to provide any closed-form expression for the optimum partitioning under arbitrary popularity distribution.

To address the aforementioned shortcomings, in this paper, we consider a system with arbitrary content popularities and analyze the coded caching scheme when caches receive one or more requests at each time slot. We derive explicit

978-1-7281-3106-1/20/$31.00 ©2020 IEEE

closed-form expressions for the server load at the delivery phase and formulate the approximated optimum partitioning problem. Our formulation considers up to three partitioning strategies and find the best strategy in polynomial complexity. In other words, our optimization strategy is a hybrid of coded and uncoded caching that approximates the best strategy by trading-off between popularity (uncoded caching) and diversity (coded caching). Furthermore, we show that our proposed optimization scheme outperforms the baseline schemes of pure uncoded and pure coded caching as well as the existing two-partitioning scheme, especially when caches receive multiple requests at a single time slot.

The rest of the paper is organized as follows. In Section II, the system model is introduced followed by an overview on coded caching. The proposed caching scheme is described in Section III. Section IV is dedicated to the system performance analysis. The simulation results are discussed in Section V. Finally, Section VI concludes the paper with possible extensions as future works.

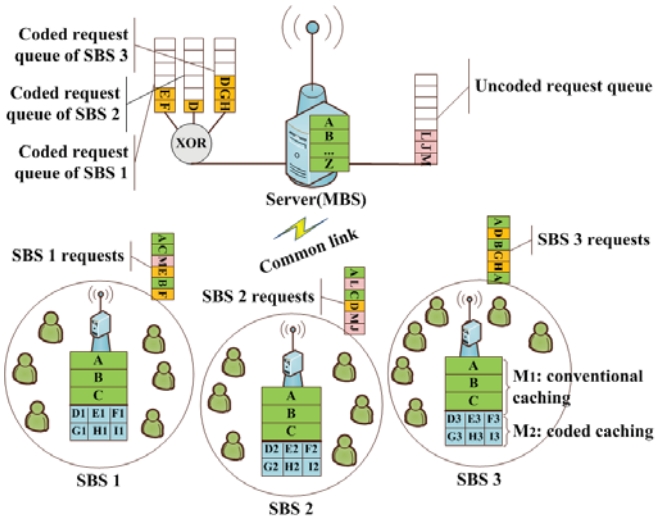


Fig. 1. An example of our caching system with one MBS containing $N = 26$ contents of size F bits connected via an error-free shared link to $K = 3$ SBSs, each with a cache size of $M \times F = 5F$ bits and serving $Z = 6$ users.

II. System Model and Preliminaries

In this section, we first introduce our system model and then summarize the coded caching scheme reported in [4].

A. System Model Description

Consider the cellular network system depicted in Fig. 1, where one MBS is connected through a shared error-free link to K number of SBSs. Each SBS has a cache memory of size $M \times F$ bits and is responsible for serving Z users. In this setting, each user is connected to only one SBS. The library contains N distinct contents that are all accessible by the MBS. We assume that all contents have the same size equal to F bits and the content popularity distribution is arbitrary. Let p_n denote the probability of requesting the n-th content. Similar to existing works, we assume two phases; the first phase corresponds to the placement phase, where caches are filled with contents from the library. In the second phase, each SBS to assumed to receive one request from each user connected to it thus, resulting in a total of Z requests. Accordingly, the MBS then sends information to satisfy these requests, i.e. the delivery phase. Our objective is to minimize the traffic load on the shared link which serves as the bottleneck in the delivery phase.

B. Background on Coded Caching

In the original coded caching scheme [4], each content is split into $\binom{K}{T}$ non-overlapping equal-sized chunks in the placement phase, where $T = K \times M/N$ and the size of each chunk is equal to $F/\binom{K}{T}$ bits. These chunks are distributed among the caches such that each cache stores M/N bits of each content. Moreover, each chunk has T copies in T different caches. In the delivery phase, each cache receives a request for a single content. The server then XORs the required chunk by different caches according to some specific coding strategy and multicasts coded messages to the corresponding groups of $T+1$ caches. The achievable rate of the coding strategy for serving all contents at the shared link is proven to be [4]:

$$R = K\left(1 - \frac{M}{N}\right) \min\left\{\frac{1}{1 + K \times M/N}, \frac{N}{K}\right\}. \quad (1)$$

where $K(1 - \frac{M}{N})$ is the local caching gain and $\frac{1}{1+K\times M/N}$ is the global caching gain.

III. The Proposed Caching Scheme

We now present the caching scheme for the case where each cache receives multiple independent requests at each time slot. Then, we formulate the corresponding optimization problem for minimizing the load on the shared link. As reported in [11]–[14], dividing the contents into two groups (caching more popular contents based on coded caching scheme and uncaching lesser popular contents) is almost optimal for coded caching scheme with non-uniform demands when each cache receives one request per time slot. In here, we propose an enhanced scheme for grouping contents in the general case of multiple requests.

In the cache content placement phase, we categorize the contents into (at most) three groups based on their request probabilities. That is to say, we first choose the N_1 most popular contents among all N contents. We then completely cache M_1 of the most popular contents from the selected N_1 contents at all caches, with the remaining $N_1 - M_1$ contents being cached based on the coded caching scheme in [4]. Accordingly, each cache memory is divided into two parts: $M_1 \times F$ bits of each cache is allocated to the M_1 most popular contents, while the remaining $(M - M_1) \times F$ bits of memory is allocated to the coded caching scheme with a library size of $N_1 - M_1$ contents. Therefore, the three groups of contents resulting from our scheme are as follows:

1) M_1 most popular contents that are cached completely.
2) $(N_1 - M_1)$ popular contents cached according to [4].
3) $(N - N_1)$ least popular contents not cached at all.

In the delivery phase, Z number of requests are received by each SBS cache. Requests corresponding to contents

belonging to the first group are locally served by the cache of each SBS, whereas requests belonging to the second or third group are processed by the MBS server. As illustrated in Fig. 1, the MBS stores these requests in two types of queues: one for coded and the other for uncoded requests. The MBS has K queues corresponding to K SBSs for coded requests, denoted by Q_s^{C}, where $s = 1, 2, \ldots, K$. In order to perform the coded caching scheme, the MBS has to maintain the requests for the coded contents of each SBS separately. It also maintains a single queue for uncoded requests from all the users, denoted by Q^{UC}. The queues corresponding to coded requests for every SBS are filled by considering an arbitrary ordering on the users' coded requests.

In what follows, we now explain the steps involved in the transmission of coded messages. Initially, the MBS sends a coded combination of chunks corresponding to the requested contents received by different SBSs from the head of Q_s^{C} using the coding method in [4]. The MBS then updates the head of line requests in the queues and repeats the same procedure, i.e. in step i, the i-th row of all queues are considered by the coded method. Note that the number of requests in each queue could be less than Z since some of the requests associated with each SBS, among all Z, belong to either the third group of contents that are not cached at all or to the first group of contents that are cached completely. Even requests belonging to the second group of contents that are cached by the coded scheme might be repetitive and are not stored in queues. As a result, the number of queues involved in the coding process at step i could be less than K since some of the queues may not have any requests in step i. Moreover, the number of steps is at most Z when all requests of at least one SBS are distinct and from the coded contents. Finally, after sending all requested contents belonging to the second group with the coded scheme, the MBS sends the contents from Q^{UC} which guarantees that all the users will be able to retrieve their requested contents.

Case in point, Fig. 1 depicts a scenario where the number of SBSs is $K = 3$ and each SBS receives $Z = 6$ requests during each time slot. Here, every SBS is responsible for 6 users, each with one request. The total number of contents is $N = 26$ and are ordered based on to their popularity (i.e., 'A' is the most popular content). The cache size is $M = 5$ contents, and we assume that $M_1 = 3$, $M - M_1 = 2$, and $N_1 = 9$. Based on this setting, contents 'A', 'B', and 'C' are cached completely. Moreover, 6 contents (from 'D' to 'I') are cached based on the coded caching method. The remaining 17 less popular contents (from 'J' to 'Z') are not cached. In the delivery phase, the uncoded data is highlighted in pink, while the contents in yellow are to be transmitted in the coded manner. The contents in green are locally hit by the cache contents without any further transmission from the MBS. In this figure, the head of lines of all the queues during the first, second, and third steps in coded transmissions contain $k_1 = 3$, $k_2 = 2$, and $k_3 = 1$ content requests, respectively.

IV. Performance Analysis

We dedicate this section to determine the expected MBS traffic load as functions of M_1 and N_1. We then characterize the partitioning strategy as an optimization problem to find the minimum load. We assume that the contents are sorted according to their popularity, i.e. $p_i \geq p_j$ if $i \leq j$ (contents with lower index are more popular).

A. Expected Traffic Load Analysis

The traffic load contributed by the MBS is either related to the requests belonging to the second group of coded contents (i.e. contents with index from M_1+1 to N_1) or associated with requests belonging to the third group of uncoded contents (i.e. contents with index from N_1+1 to N). With regard to the process explained in Section III, we denote K_i to be the number of non-empty queues in step i, where $i = 1, 2, \ldots, Z$. The following lemma introduces the traffic load corresponding to the coded contents in step i when K_i variables are known.

Lemma 1. *The traffic load of the coded contents in step i when $K_i = k_i$, denoted by $r_1(k_i)$, is derived as follows, where $T = K \times (M - M_1)/(N_1 - M_1)$:*

$$r_1(k_i) = \min\left(\frac{\binom{K}{T+1} - \binom{K-k_i}{T+1}}{\binom{K}{T}}, N_1 - M\right). \quad (2)$$

Proof. In regard to the traffic load of coded contents, each content is split into $\binom{K}{T}$ non-overlapping chunks, each of size $1/\binom{K}{T}$ bits. Each cache selects T/K chunks out of all the chunks. For each transmitted signal, the MBS sends the XOR of the $T+1$ chunks requested by $T+1$ different caches. Each cache can decode its requested content from the signal with the help of its cached contents. If for all caches there are Z request in the queue of coded requests, then at each step, $T+1$ out of the K caches are selected to serve requests by multicasting. In this case, the total number of multicast transmissions is $\binom{K}{T+1}$. But as mentioned earlier, in our problem, the number of contents in the queue of coded requests may be lower than Z. If such is the case, then it is not required to transmit from permutations where all selected caches do not have any request in the corresponding queue of coded requests. Hence, if k_i is the number of caches that have requests in step i, then the number of unnecessary transmissions would be $\binom{K-k_i}{T+1}$. Consequently, the number of multicast transmissions at each step in our problem would be $\binom{K}{T+1} - \binom{K-k_i}{T+1}$, where the size of each transmission is equal to $1/\binom{K}{T}$ and $T = \frac{K\times(M-M_1)}{(N_1-M_1)}$.

Contrarily, if $(N_1 - M_1) < k_i$, then the MBS enjoys an improvement of $(N_1 - M_1)/k_i$ from broadcasting. Thus, from (1), the traffic load of coded contents is given as:

$$k_i \times \left(1 - \frac{M - M_1}{N_1 - M_1}\right) \times \frac{N_1 - M_1}{k_i} = N_1 - M. \quad (3)$$

This completes the proof. □

Lemma 2. *The expected traffic load of coded content requests, denoted by r_1, is approximated by:*

$$r_1 = \sum_{i=1}^{Z} \frac{\binom{K}{T+1} - \sum_{k_i=0}^{K} Pr\{K_i = k_i\}\binom{K-k_i}{T+1}}{\binom{K}{T}}, \quad (4)$$

where

$$Pr\{K_i = k_i\} = \binom{K}{k_i}(P_{Zi})^{k_i}(1-P_{Zi})^{K-k_i}, \quad (5)$$

$$P_{Zi} \simeq 1 - \sum_{j=0}^{i-1}\binom{Z}{j}q^j(1-q)^{Z-j}, \quad (6)$$

with $q = \sum_{n=M_1+1}^{N_1} p_n$. *Moreover, the expected traffic load of uncoded content requests, denoted by* r_2, *is:*

$$r_2 = \sum_{n=N_1+1}^{N} 1-(1-p_n)^{Z\times K}. \quad (7)$$

Finally, the total expected traffic rate is $r = r_1 + r_2$.

Proof. From Lemma 1, the expected traffic load of the coded requests can be expressed as:

$$r_1 = E\left[\sum_{i=1}^{Z}\min\left(\frac{\binom{K}{T+1}-\binom{K-K_i}{T+1}}{\binom{K}{T}}, N_1-M\right)\right]. \quad (8)$$

Note that the expectation is taken over the random variables K_i for $i = 1, 2, \ldots, Z$. In (8), the minimization function involves two terms; the first term reaches the maximum value at $K_i = K$ as shown below:

$$\max\left(\frac{\binom{K}{T+1}-\binom{K-K_i}{T+1}}{\binom{K}{T}}\right) = \frac{\binom{K}{T+1}}{\binom{K}{T}} = \frac{K-T}{T+1}. \quad (9)$$

By replacing T in the above equation, we arrive at:

$$\frac{K-T}{T+1} = \frac{K\times(N_1-M)}{N_1-M_1+K\times(M-M_1)}. \quad (10)$$

If (10) is less than $N_1 - M$, then the first term in the minimization function of (8) is always selected. To this end, we have the following inequality:

$$1 < \frac{N_1-M_1}{K} + (M-M_1).$$

The above inequality holds when $N_1 > M$ and $M - M_1 \geq 1$. In the case of pure uncoded caching, where $N_1 = M_1 = M$, we have $r_1 = 0$. Also, we know that $M_1 < M$ if $N_1 > M$ and therefore, (8) reduces to:

$$\begin{aligned} r_1 &= E\left[\sum_{i=1}^{Z}\frac{\binom{K}{T+1}-\binom{K-K_i}{T+1}}{\binom{K}{T}}\right] \\ &= \sum_{i=1}^{Z}\frac{\binom{K}{T+1}-E\left[\binom{K-K_i}{T+1}\right]}{\binom{K}{T}} \\ &= \sum_{i=1}^{Z}\frac{\binom{K}{T+1}-\sum_{k_i=0}^{K}Pr\{K_i=k_i\}\binom{K-k_i}{T+1}}{\binom{K}{T}}. \end{aligned} \quad (11)$$

In (11), the notation $Pr\{K_i = k_i\}$ denotes the probability that k_i SBSs request for coded contents in step i. This occurs when exactly k_i SBSs receive at least i distinct requests for coded contents, while the rest of the SBSs receive less than i distinct requests. Since the user requests are independent of each other, $Pr\{K_i = k_i\}$ is derived according to the binomial distribution given in (5), where P_{Zi} symbolizes the probability that an SBS has at least i distinct requests for coded contents. Due to non-uniform distribution of the contents, the derivation of P_{Zi} is tedious. Here, we assume that all coded content requests received by each SBS, inclusive of repetitive contents, are placed in their corresponding queue at the MBS server.

Apparently, such approximation may not be the best strategy since duplicate requests from a single SBS need not be transmitted by the MBS. When the popularity distribution is almost uniform, and for low quantities of Z, this approximation is more accurate. Contrarily, when the popularity distribution is extremely non-uniform and for high quantities of Z, where few contents are in high demand, this approximation yields lower accuracy. However, the negligible impact of this approximation on our choice of library partitioning, i.e. N_1 and M_1, will be shown numerically and via simulation in the following section. Such approximation does not affect rate calculation for all contents but, only for contents located in different partitions based on different policies. In addition, under extremely non-uniform popularity distributions, the strategy tends to cache this content completely (tends to higher value for M_1) and the approximation may reinforce this tendency. By caching the most popular files completely, the error of this approximation is greatly reduced. Hence, the approximation is reasonable and has minimal impact on choosing the optimal policy. In this case, P_{Zi} can be derived from the binomial distribution given in (6), where q denotes the probability of a request for coded content given by the summation of all p_n for $n = M_1 + 1, \ldots, N_1$.

To prove (7), we recall that from a total of $K \times Z$ requests, any request from the third group of contents that is not cached (contents indexed from $N_1 + 1$ to N) should be satisfied directly by the MBS. Nonetheless, if the MBS receives multiple requests for a content in one time slot, it relies on broadcasting to send the content only once. Therefore, the expected traffic load of such uncached contents is equal to the expected number of distinct requests. The probability that content n is not requested is $(1-p_n)^{K\times Z}$. Thus, the probability that content n is requested at least one time is $1-(1-p_n)^{K\times Z}$. So, the expected total number of distinct requests for uncoded contents is equal to the sum of this expected probability for all uncoded contents that can be derived to yield (7) in a straightforward manner. This completes the proof. □

B. Optimal Library Partitioning Policy

We now formulate the near optimum partitioning problem with the objective of minimizing the traffic load from the MBS to the SBSs. In other words, we need to find optimum M_1 and N_1 values that minimize r. The minimization problem is presented as follows, where $T = \frac{K\times(M-M_1)}{(N_1-M_1)}$ is restricted to integer values:

$$\begin{aligned} &r^*(N,M,K,Z,\alpha) = \\ &\min_{\substack{M\leq N_1\leq N\\ 0\leq M_1\leq M}}\left\{\min\left(\sum_{i=1}^{Z}\frac{\binom{K}{T+1}-E\left[\binom{K-K_i}{T+1}\right]}{\binom{K}{T}}, N_1-M\right)+\right. \\ &\qquad\left.\sum_{n=N_1+1}^{N}1-(1-p_n)^{K\times Z}\right\}. \end{aligned} \quad (12)$$

We now adopt an enhanced exhaustive search algorithm to solve this optimization problem in polynomial time. To

reduce the complexity of the search algorithm, we consider the following simplifications:

- We do not perform calculation for all possible N_1 and M_1 combinations since only fractions of N_1 and M_1 are valid for $T = \frac{K\times(M-M_1)}{(N_1-M_1)}$ to be an integer value.
- By considering $\frac{\binom{K}{T+1}}{\binom{K}{T}} = \frac{K-T}{T+1}$, we calculate $\frac{K-T}{T+1}$ only once for each possible N_1 and M_1 values.
- By storing the result of earlier computations, we can skip duplicate computations. For instance, we can compute and store $\binom{K}{k_i}$ or $\binom{Z}{j}$ (or $K!$ or $Z!$) once and use it in the calculations that follow. Similarly, as another example, $\sum_{n=N_1+1}^{N} 1-(1-p_n)^{Z\times K}$ for one quantity of N_1 can be used to compute it for the next N_1 values with lower computations.

V. Simulation Results and Discussions

In this section, simulation results are used to compare the performance of the proposed scheme with respect to the pure coded and conventional uncoded baseline schemes. To serve this purpose, we compare our scheme with the optimum two-partitioning pure coded scheme in [11]–[14] and the greedy constrained local coloring (GCLC) approach in [16].

A. Simulation Set-up

We first verify the approximation used for finding the optimal configuration in our scheme. We then study the affect of system parameters on the optimum traffic load of the shared link. For analytical results, the optimum M_1 and N_1 for integer values of $T = K \times (M - M_1)/(N_1 - M_1)$ are obtained from (12) and then validated via simulations conducted using MATLAB over a period of 2000 time units. The optimal value of N_1 (when $M_1 = 0$) for the two-partitioning pure coded scheme is also computed for integer values of $T = K\times M/N_1$. We assume that there are $N = 1000$ content files available in the MBS and a total of $K = 10$ SBSs in place. The cache size of each SBS is $M = 100$ and each serves $Z = 10$ users. To generate requests at each time slot, each user selects a random number between 0 and 1. Based on cumulative content popularity distribution function, the content corresponding to this random number is then selected and the user generates a request for it. Thus, at each time slot, Z random numbers are generated for each SBS to determine the requests. We suppose that the content popularity follows the Zipf popularity profile with parameter $\alpha > 0$ as given below:

$$p_n = \frac{(\frac{1}{n})^{\alpha}}{\sum_{j=1}^{N}(\frac{1}{j})^{\alpha}}. \tag{13}$$

B. Discussions

Fig. 2 shows the MBS traffic load as a function of N_1 for different scenarios of our hybrid coded-uncoded, pure coded and pure uncoded caching schemes. The content popularity follows the Zipf distribution with $\alpha = 1$. Considering (12), we observe that the hit ratio of local cache improves as M_1 increases. But, due to reduction memory space for the coded requests, the bandwidth load required to satisfy the coded content requests increases. Moreover, by increasing

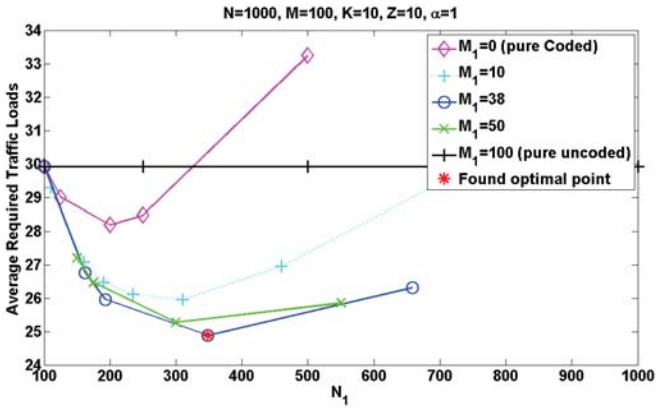


Fig. 2. Traffic load in the delivery phase as a function of N_1 for different M_1 values and the approximated optimal point.

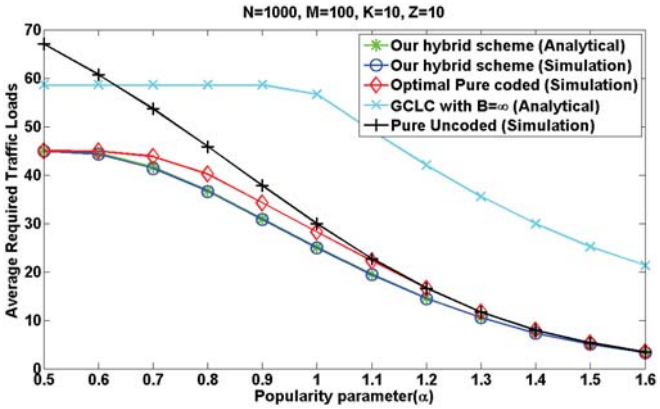
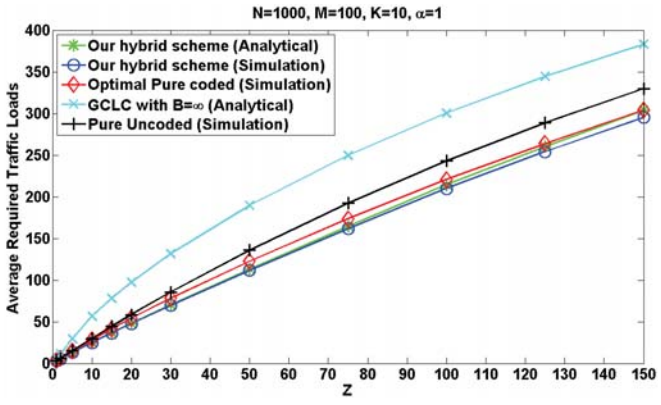


Fig. 3. Traffic load as a function of popularity parameter, α.

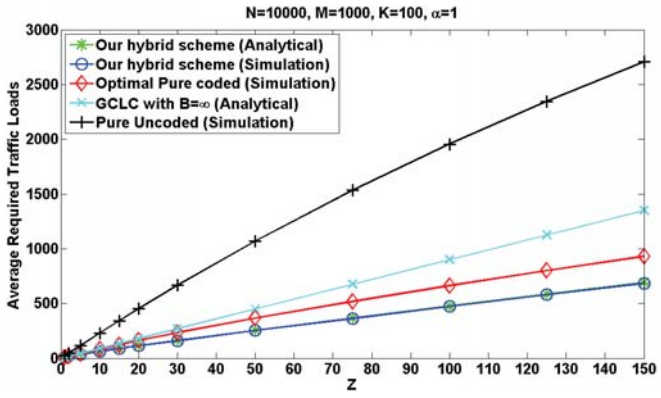
the value of N_1, although the contents that are not cached decreases, the required bandwidth load to satisfy requests of coded contents rises. The optimal configuration obtained for the optimization problem ($M_1^* = 38$ and $N_1^* = 348$) is also indicated in this figure. We see that the proposed hybrid caching scheme outperforms the pure coded and pure uncoded schemes, and our computed configuration imposes the lowest load as compared to the other configurations.

The simulation and analytical results for the traffic load with respect to the Zipf parameter $\alpha \in [0.5, 1.6]$ is plotted in Fig. 3. In this figure, the benchmark schemes are the optimal two-partitioning coded scheme, the GCLC method, and the conventional uncoded scheme. The GCLC is a coded scheme which handles multiple requests per cache by graph coloring techniques, where each content is divided into B number of sub-contents. Here, we assume uniform cache content placement as given in [16] and $B = \infty$[1]. In particular, for our hybrid coded-uncoded and pure coded scheme, we first find the M_1^* and N_1^* values for each popularity parameter that minimizes the MBS traffic load. The caching schemes are then evaluated for their corresponding optimal configuration via simulation. It is evident from the figure that the simulation results are very close to the analytical findings and hence, our hybrid caching can lead to significant traffic off-loading compared to pure coded and pure uncoded schemes

[1]More precisely, we use Eq. (4) in [16] to arrive at the expected traffic load.



(a)



(b)

Fig. 4. Traffic load in the delivery phase versus number of users per SBS (Z) for different system parameters.

(i.e. up to 12.5% better than optimal two-partitioning pure coded scheme). The plot also reveals that the optimal configuration of $M_1 = M$ is achieved when α is larger than 1.5. Therefore, under such conditions, the optimal traffic load of pure coded and hybrid coded-uncoded schemes are equal to the traffic load of uncoded scheme. Also, when $\alpha < 0.5$, the optimal configuration is $M_1^* = 0$ and $N_1^* = N$. That is to say, caching with maximum diversity is optimal and thus, uncoded caching has the worst performance.

In Fig. 4, the optimal traffic load (r) is plotted with respect to Z for different content library sizes and different number of SBSs. The number of users in the system $(K \times Z)$ is taken to be $10 \times Z$ and $100 \times Z$ in Fig. 4a and Fig. 4b, respectively. Our proposed scheme shows a significant improvement, especially for $Z > 2$, in both settings. In comparison with Fig. 4a, we also observe that the rate of the coded schemes (pure and the proposed hybrid) in Fig. 4b is much better than the pure uncoded scheme. This gap is due to the global caching gain of coded schemes. It is also noteworthy that the approximation used in Lemma 2 is more accurate for lower values of Z in Fig. 4a, whereas it is accurate for all Z values given in Fig. 4b.

VI. Conclusion and Future Work

In this paper, we have analyzed the coded caching scheme in a system where caches may receive multiple requests at a given time slot under non-uniform content popularity distributions. We first derived explicit closed-form expressions for the server load at the delivery phase. Then, the almost optimum partitioning problem was formulated wherein, we considered up to three partitioning strategies and found the best strategy in polynomial complexity. Validated by simulation results, we showed that the proposed scheme outperforms the baseline schemes of pure uncoded and pure coded caching, as well as the two-partitioning scheme existing in the literature. The strength of the proposed scheme was better noticeable when the caches received multiple requests at each time slot.

Future directions of this work include analysis of improved methods to choose appropriate policies accounting for heterogeneity in users' behaviour such as different number of requests per SBS and user-dependent content popularity.

Acknowledgment

This work was partially supported by the Social Policy Grant funded by Nazarbayev University, Kazakhstan.

References

[1] V. Chandrasekhar, J. G. Andrews, and A. Gatherer, "Femtocell networks: a survey," *IEEE Commun. Mag.*, vol. 46, no. 9, pp. 59–67, Sep. 2008.

[2] Y. Chen, M. Ding, J. Li, Z. Lin, G. Mao, and L. Hanzo, "Probabilistic small-cell caching: Performance analysis and optimization," *IEEE Trans. Veh. Technol.*, vol. 66, no. 5, pp. 4341–4354, May 2017.

[3] G. Paschos, E. Bastug, I. Land, G. Caire, and M. Debbah, "Wireless caching: technical misconceptions and business barriers," *IEEE Commun. Mag.*, vol. 54, no. 8, pp. 16–22, Aug. 2016.

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

[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, Jun. 2016.

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

[7] M. A. Maddah-Ali and U. Niesen, "Decentralized coded caching attains order-optimal memory-rate tradeoff," *IEEE/ACM Trans. Netw.*, vol. 23, no. 4, pp. 1029–1040, Aug. 2015.

[8] S. B. Hassanpour, A. Khonsari, S. P. Shariatpanahi, and A. Dadlani, "Hybrid coded caching in cellular networks with D2D-enabled mobile users," in *Proc. of IEEE Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Sep. 2019, pp. 1–6.

[9] Y. Wei and S. Ulukus, "Coded caching with multiple file requests," in *Proc. of Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Oct. 2017, pp. 437–442.

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

[11] J. Hachem, N. Karamchandani, and S. Diggavi, "Effect of number of users in multi-level coded caching," in *Proc. of IEEE International Symposium on Information Theory (ISIT)*, Jun. 2015, pp. 1701–1705.

[12] T. Li, M. Ashraphijuo, X. Wang, and P. Fan, "Traffic off-loading with energy-harvesting small cells and coded content caching," *IEEE Trans. Commun.*, vol. 65, no. 2, pp. 906–917, Feb. 2017.

[13] 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, Jun. 2017.

[14] J. Zhang, X. Lin, and X. Wang, "Coded caching under arbitrary popularity distributions," *IEEE Trans. Inf. Theory*, vol. 64, no. 1, pp. 349–366, Jan. 2018.

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

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