Adaptive Delivery in Caching Networks

Seyed Ali Saberali, Hamidreza Ebrahimzadeh Saffar,
Lutz Lampe, *Senior Member, IEEE*, and Ian Blake, *Fellow, IEEE*

***Abstract*—The problem of content delivery in caching networks is investigated for scenarios where multiple users request identical files. An adaptive method is proposed for the delivery of redundant demands in caching networks. Based on the redundancy pattern in the current demand vector, the proposed method decides between the transmission of uncoded messages or the coded messages of [1] for delivery. Moreover, a lower bound on the delivery rate of redundant requests is derived. The performance of the adaptive method is investigated through numerical examples and Monte Carlo simulations. It is shown that the adaptive method considerably reduces the performance gap to the lower bound for specific ranges of network parameters.**

***Index Terms*—Adaptive delivery algorithm, average delivery rate, coded caching, correlated requests, redundant demands.**

I. Introduction

LOCAL caching is a promising technique to meet the unprecedented traffic demands in the next generation communication networks [1]–[6]. Caching networks operate in two phases, commonly referred to as placement and delivery phases. In the placement phase, caches fill their memories with parts of the popular files during the off-peak hours. The delivery phase, however, is performed when the network is congested. In this phase, each cache provides its users with the parts of the requested files that it has available. The remaining parts are conventionally delivered to the users through unicast transmissions performed by a central server on a broadcast channel. In a more recent approach, known as coded caching [2], the central server uses coded-multicasting to deliver the requested content, to further reduce the network congestion.

In [2], the authors derived an information-theoretic formulation for the caching problem and proposed a centralized scheme for coded caching. In a later work [1], a decentralized coded caching method was proposed which became the building block of several caching schemes developed for more complicated scenarios [6]–[9]. [1], [2] used the peak delivery rate as the figure of merit of the network. The peak rate occurs when all the users request distinct files, given that the number of files is greater than the number of caches.

Average delivery rate is another significant network performance metric, and depends on the statistics of the user requests. [5]–[8] proposed caching schemes to decrease the average delivery rate in scenarios with non-uniform file popularities. The statistics of user requests also affect the design of caching networks by increasing the chance of multiple users requesting identical files. In such a scenario, the delivery method can be modified to benefit from the redundancies in the user demands, and further reduce the delivery rate.

Redundant demands are likely when either the files have significantly different popularity levels or the user requests are positively correlated. For the case of non-uniform file popularities, the schemes in [6]–[8] do not take the effect of redundant requests into account. This is because the delivery in all these schemes is based on the delivery of [1], which is designed for the demand vectors with distinct requests. In addition to non-uniform popularity levels, correlated requests are also likely in many practical scenarios. A considerable amount of multimedia requests are made through social networks like Facebook and Instagram, where users with common friends and interests are likely to request the same content.

In this paper, we investigate the delivery of redundant demands in caching networks. We use the placement schemes of [1], [2] to ensure that the peak delivery rate does not exceed the delivery rates of [1], [2], and the link capacity is satisfied. Further, these placement schemes are natural candidates when the file popularities are uniform or little prior knowledge about the popularities is available during the placement phase.

For the delivery phase, we propose an adaptive scheme based on *message selection*. Upon receiving a demand vector from the users and based on the redundancy pattern of the requests made, the server decides whether to use uncoded messages or the coded messages of [1] to deliver each part of the requested files. This distinguishes our work from [1], [2], as our proposed delivery takes the specifics of the current demand vector into account to decide on the form of the server messages. However, [1], [2] use a fixed structure to compose the server messages for all demand vectors. We show the superiority of our adaptive method through numerical examples and Monte Carlo simulations. We also derive a lower bound on the delivery rate of redundant requests. In some cases, the adaptive method shrinks the gap between the average rate of the non-adaptive scheme and the lower bound by 50%.

The remainder of this paper is organized as follows. In Section II, we present the network model. The adaptive delivery scheme is derived in Section III. Section IV presents numerical examples and simulation results.

II. Network Model

Assume a network with a central server and K caches. The server is able to communicate with the caches through a broadcast link. We denote the set of all caches in the network by $\mathcal{K}$. A library of $N \geq K$ files is given, where each file is F bits long. All files are available at the central server. Each cache has a memory capacity of $M \times F$ bits. We define $q \triangleq \frac{M}{N}$.

Placement Phase: Placement takes place only once and remains unchanged during the delivery phase. After the placement, the distribution of bits in the caches can be described as follows. For a given file n and a given subset of caches $\mathcal{S} \subset \mathcal{K}$, denote by $V_{\mathcal{S}}^{n}$ the subset of bits of file n that are exclusively stored at the caches in $\mathcal{S}$. The resulting subsets of bits *partition* the set of all the bits of every file into 2^K

The authors are with the Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, Canada.

partitions. Let $s \triangleq |\mathcal{S}|$. Define $x_s \triangleq |V_{\mathcal{S}}^n|/F$ as the portion of the bits of file n that are exclusively stored at each subset $\mathcal{S}$ of caches with cardinality s. Here, we have assumed that $|V_{\mathcal{S}}^n|$ only depends on s. In particular, it neither depends on n nor on the particular choice of caches in $\mathcal{S}$, as long as the cardinality of $\mathcal{S}$ is s. This holds as we assume uniform file popularities during the placement.

The placement phase can be performed through either the centralized scheme of [2] or the decentralized scheme of [1]. For the centralized placement, we have

$$x_s^{\text{cen}} = \begin{cases} 1/\binom{K}{s} & s = t \\ 0 & s \neq t \end{cases} \tag{1}$$

where $t = KM/N$ [2]. For the decentralized placement and for large F, with high probability we have [1]

$$x_s^{\text{decen}} \approx q^{s-1}(1-q)^{K-s+1}, \quad s = 0, ..., K. \tag{2}$$

Delivery Phase: In the delivery phase, the network serves one user of *every* cache at a time. Denote the requests of the users of caches $1, ..., K$, with $d_1, ..., d_K$, respectively. We refer to the vector $[d_1, ..., d_K]$ as the demand vector. We represent the number of distinct files in the demand vector by L, where $1 \leq L \leq K$. The demand vector is called *redundant* if $L < K$. Also, denote by k_i, the number of requests for the i-th most requested file in the current demand vector. Thus $k_i \geq k_j$ for $i > j$ and $i, j \in \{1, ..., L\}$. We call $(k_1, ..., k_L)$ the redundancy pattern of the demand vector. For a demand vector $[d_1, ..., d_K]$, we define the delivery rate $R(M, [d_1, ..., d_K])$ as the total equivalent number of files transmitted by the server, such that all the caches successfully recover the files they requested.

The delivery phase is performed through the delivery algorithm [1, Algorithm 1]. We consider the rate of this algorithm as the state-of-the-art benchmark for the adaptive method we propose in Section III. Notice that the delivery method for the centralized caching in [2] is a special case of [1, Algorithm 1].

Discussion: Note that if file n is requested by multiple users, including user k, [1, Algorithm 1] embeds $V_{\mathcal{S}\setminus\{k\}}^n$ into several messages. If $s > 1$, user k has the side information to decode only one of those messages. As a result, the server needs to send all the messages with $s > 1$. This is not the case for the messages with $s = 1$, i.e., $\mathcal{S} = \{k\}$. In these cases, $\oplus_{k\in\mathcal{S}} V_{\mathcal{S}\setminus\{k\}}^{d_k} = V_{\emptyset}^{d_k}$. Such uncoded messages deliver the bits that are not stored at any cache in the system. All the users that request file n can decode $V_{\emptyset}^n$, so it needs to be sent only once. As a result, the traffic due to the uncoded messages is Lx_0 instead of Kx_0. Thus, the total delivery rate will be

$$Lx_0 + \sum_{s=1}^{K-1} \binom{K}{s+1} x_s. \tag{3}$$

For $L = K$, substitution of (1) and (2) in (3) gives the peak rates in [2, eq. 2] and [1, Theorem 1] for the corresponding caching schemes. Yet, (3) suggests that for redundant demand vectors, the actual rate of [1, Algorithm 1] can be smaller than the peak rates. This is the basis of our analysis in Section III.

III. Adaptive Caching Scheme

We now design an adaptive delivery method that unlike [1], [2], exploits the redundancies in the user requests in order to reduce the delivery rate. We further derive a lower bound on the delivery rate of redundant demand vectors.

A. Adaptive Delivery Method

For the adaptive method, we introduce an extra step to the delivery phase, which takes place after receiving each request vector and before the transmission of the server messages to the users. In this step, the server decides whether to send each part of the requested files through the corresponding coded message in [1, Algorithm 1] or through an uncoded message. The use of uncoded messages instead of coded messages to deliver file n is equivalent to transferring bits from $V_{\mathcal{S}}^n : s > 0$ to $V_{\emptyset}^n$. By such a transfer, the cache only ignores parts of its content and it does not change the actual placement of files. Let $\hat{V}_{\mathcal{S}}^n$ represent the subset of the bits of file n exclusively cached at $\mathcal{S}$ after the transfer is done, and $y_{\mathcal{S}}^n \triangleq |\hat{V}_{\mathcal{S}}^n|/F$.

In our delivery method, the server first optimizes $y_{\mathcal{S}}^n$. Then, it arbitrarily picks $y_{\mathcal{S}}^n F$ bits of $V_{\mathcal{S}}^n$ to form $\hat{V}_{\mathcal{S}}^n$, and adds the rest of the bits to $\hat{V}_{\emptyset}^n$. Finally, it uses [1, Algorithm 1] for delivery based on the resulting subsets $\hat{V}_{\mathcal{S}}^n$ instead of $V_{\mathcal{S}}^n$.

We now find the optimal lengths of the updated subsets $\hat{V}_{\mathcal{S}}^n$ to minimize the sum of the lengths of messages $\oplus_{k\in\mathcal{S}} \hat{V}_{\mathcal{S}\setminus\{k\}}^{d_k}$ over all $\mathcal{S} \subset \mathcal{K}$. Let $\mathcal{D}$ denote the set of the distinct files requested in the current demand vector. Note that $|\mathcal{D}| = L \leq K$, and both $\mathcal{D}$ and L evolve with time. Then, the rate minimization problem is given by

$$\begin{aligned} \underset{y_{\mathcal{S}}^{d_k}}{\text{minimize}} \quad & \sum_{\mathcal{S}:\mathcal{S}\subset\mathcal{K}} \max_{k\in\mathcal{S}} y_{\mathcal{S}\setminus\{k\}}^{d_k} \\ \text{subject to} \quad & \sum_{\mathcal{S}:\mathcal{S}\subset\mathcal{K}} y_{\mathcal{S}}^{d_k} = 1, \quad \forall\, d_k \in \mathcal{D} \\ & 0 \leq y_{\mathcal{S}}^{d_k} \leq x_{|\mathcal{S}|}, \quad \forall\, d_k \in \mathcal{D},\ \forall \mathcal{S} \subset \mathcal{K} : |\mathcal{S}| > 0 \\ & 0 \leq y_{\emptyset}^{d_k} \leq 1, \qquad \forall\, d_k \in \mathcal{D}. \end{aligned} \tag{4}$$

In (4), $x_{|\mathcal{S}|} = |V_{\mathcal{S}}^n|/F$ are known from the placement phase and are given by (1) and (2) for the centralized and decentralized placements, respectively. $\max_{k\in\mathcal{S}} y_{\mathcal{S}\setminus\{k\}}^{d_k}$ is the length of the message $\oplus_{k\in\mathcal{S}} \hat{V}_{\mathcal{S}\setminus\{k\}}^{d_k}$. Thus, the objective function is the rate of [1, Algorithm 1] operating based on the adjusted subsets $\hat{V}_{\mathcal{S}}^n$. The equality constraints are the partition constraints. The parameter range constraints permit the server to use uncoded messages instead of coded messages, but not vice versa.

Algorithm 1 Original Adaptive Delivery Algorithm

Require: $\{V_{\mathcal{S}}^n\}_{n,\mathcal{S}}$ *// From the placement phase*
Procedure AdaptiveDelivery($d_1, ..., d_K$)
$\mathcal{D} \leftarrow$ unique$(d_1, ..., d_K)$ *// Set of distinct files requested*
$\{y_{\mathcal{S}}^{*\,d_k}\}_{d_k\in\mathcal{D},\mathcal{S}\subset\mathcal{K}} \leftarrow$ Solution of Problem (4)
for $d_k \in \mathcal{D}$ **do**
  $\hat{V}_{\emptyset}^{d_k} \leftarrow \emptyset$ *// Initialization of* $\hat{V}_{\emptyset}^{d_k}$
  for $\mathcal{S} \subset \mathcal{K}$ **do**
    $\hat{V}_{\mathcal{S}}^{d_k} \leftarrow \{$first $y_{\mathcal{S}}^{*\,d_k} F$ bits of $V_{\mathcal{S}}^{d_k}\}$
    $\hat{V}_{\emptyset}^{d_k} \leftarrow \hat{V}_{\emptyset}^{d_k} \cup \{$ last $(1 - y_{\mathcal{S}}^{*\,d_k})F$ bits of $V_{\mathcal{S}}^{d_k}\}$
  end for
end for
Use [1, Algorithm 1] with $\{\hat{V}_{\mathcal{S}}^n\}_{n,\mathcal{S}}$ instead of $\{V_{\mathcal{S}}^n\}_{n,\mathcal{S}}$

Problem (4) can be posed as a linear programming problem by the standard technique of defining ancillary variables $z_{\mathcal{S}} = \max_{k\in\mathcal{S}} y_{\mathcal{S}\setminus\{k\}}^{d_k}$, and adding the extra constraints

$$z_{\mathcal{S}} \geq y_{\mathcal{S}\setminus\{k\}}^{d_k},\ z_{\mathcal{S}} \leq -y_{\mathcal{S}\setminus\{k\}}^{d_k}, \quad k \in \mathcal{S} \tag{5}$$

for all $\mathcal{S} \in \mathcal{K} : |\mathcal{S}| > 0$ [10, Sec. 4.3]. The resulting linear programming problem can be solved numerically for $y_{\mathcal{S}}^{*d_k}$. Algorithm 1 shows the proposed adaptive delivery scheme.

B. Simplified Adaptive Delivery

A simplified version of the message selection step can be formulated by only taking the number of distinct requests L into account, and ignoring the redundancy pattern of the demand vector. Then, because of the symmetry, we set $y_{\mathcal{S}}^n = y_s$ for all n and all $\mathcal{S} : |\mathcal{S}| = s$. This leads to

$$\begin{aligned} \underset{y_s}{\text{minimize}} \quad & Ly_0 + \sum_{s=1}^{K-1} \binom{K}{s+1} y_s \\ \text{subject to} \quad & \sum_{s=0}^{K} \binom{K}{s} y_s = 1 \\ & 0 \le y_s \le x_s, \ s = 1, ..., K \\ & 0 \le y_0 \le 1 \end{aligned} \tag{6}$$

as the simplified message selection problem.

Proposition 1: Let $\hat{s} = \lfloor \frac{K-L}{L+1} \rfloor$. Optimal parameters for the simplified message selection problem of (6) are given by

$$y_s^* = \begin{cases} \sum_{i=1,...,\hat{s}} \binom{K}{i} x_i, & s = 0 \\ 0, & s = 1, ..., \hat{s} \\ x_s, & s = \hat{s}+1, ..., K \end{cases}. \tag{7}$$

Proof: If we transfer bits from the subsets $V_{\mathcal{S}}^n : |\mathcal{S}| = s$ to $V_{\emptyset}^n$, the resulting change in the rate will be $L\binom{K}{s}x_s - \binom{K}{s+1}x_s$. We transfer the bits only if this difference is negative. This is the case when $s \le \hat{s}$. This results in the parameters of (7). ■

Algorithm 2 shows the simplified adaptive delivery scheme.

Algorithm 2 Simplified Adaptive Delivery Algorithm

Require: $\{V_{\mathcal{S}}^n\}_{n,\mathcal{S}}$ *// From the placement phase*

Procedure SimplifiedAdaptiveDelivery($d_1, ..., d_K$)
$L = \text{size}(\text{unique}(d_1, ..., d_K))$ *// Number of distinct requests*
$\hat{s} \leftarrow \lfloor \frac{K-L}{L+1} \rfloor$
for $d_k \in \mathcal{D}$ **do**
  $\hat{V}_{\emptyset}^{d_k} \leftarrow \cup_{\mathcal{S}:s\le\hat{s}} V_{\mathcal{S}}^{d_k}$ *// Corresponds to the first rule of (7)*
  for $\mathcal{S} \subset \mathcal{K} : |\mathcal{S}| > 0$ **do**
    if $|\mathcal{S}| \le \hat{s}$ **then**
      $\hat{V}_{\emptyset}^{d_k} \leftarrow \emptyset$ *// Corresponds to the second rule of (7)*
    else
      $\hat{V}_{\mathcal{S}}^{d_k} \leftarrow V_{\mathcal{S}}^{d_k}$ *// Corresponds to the third rule of (7)*
    end if
  end for
end for
Use [1, Algorithm 1] with $\{\hat{V}_{\mathcal{S}}^n\}_{n,\mathcal{S}}$ instead of $\{V_{\mathcal{S}}^n\}_{n,\mathcal{S}}$

To recap, the proposed adaptive delivery algorithms adjust their use of coded and uncoded messages based on the redundancies in the demand vector. This is in contrast to the delivery of [1], [2], where the server always uses the same structure for construction of the coded messages, regardless of the redundancy pattern in the demand vector.

C. Lower Bound

Let $R_L^*(M)$ denote the minimum rate that is achievable for every possible demand vector with L distinct requests. Proposition 2 gives a lower bound on $R_L^*(M)$.

Proposition 2 (Cutset Bound): Assume that K caches request $L \le K$ distinct files. Then, $R_L^*(M)$ must satisfy

$$R_L^*(M) \ge \max_{s\in\{1,...,L\}} \left(s - \frac{s}{\lfloor N/s \rfloor} M \right). \tag{8}$$

Proof: We modify the cutset bound argument of [2, Sec. VI] to bound $R_L^*(M)$. Let $\mathcal{S}$ be a subset of caches with $|\mathcal{S}| = s$, such that there are no two caches in $\mathcal{S}$ with identical user requests. Assume that these caches request files $1, ..., s$ from the library. Let X_1 denote the server's input to the shared link which determines files $1, .., s$. Similarly, assume that the same users request files $(i-1)s+1, ..., is$, and the server input X_i determines the files requested. Let $i = 1, ..., \lfloor N/s \rfloor$. Consider the cut separating $X_1, ..., X_{\lfloor N/s \rfloor}$ and the caches in $\mathcal{S}$ from the corresponding users. The total information available to the users in the cut should be more than or equal to the total information requested. Thus, $\lfloor N/s \rfloor R_L^*(M) + sM \ge s\lfloor N/s \rfloor$. Since s accepts any value between 1 and L, (8) results. ■

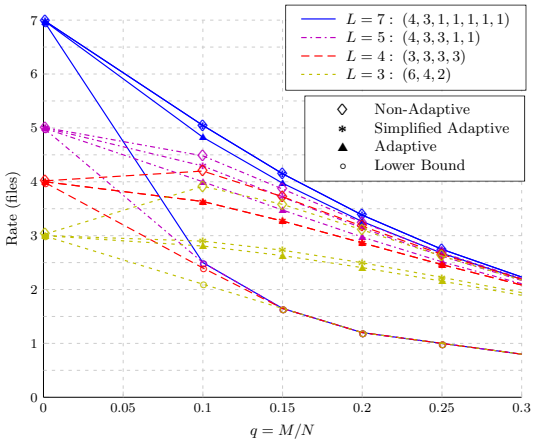

Fig. 1. Rates of different delivery schemes for $K = 12$.

IV. Numerical Examples and Simulation Results

We now compare the performance of the proposed adaptive methods and the non-adaptive method of [1, Algorithm 1] through numerical examples. Notice that by the rate of non-adaptive method, we refer to the rate of [1] or [2] depending on whether the decentralized or centralized placement is used, respectively. This rate is calculated by (3).

Fig. 1 shows the delivery rates of the non-adaptive and adaptive schemes, as well as the lower bound (8) for a network of $K = 12$ caches. The same decentralized placement is used for all cases with the parameters in (2). We consider several redundancy patterns for the demand vector with different values of L. In Fig. 1, we observe a considerable improvement in the delivery rate for $M/N \le 0.25$, when the adaptive methods are used. This improvement in the rate is more considerable when L is smaller. For instance, the performance gap to the lower bound decreases by almost 50% when $L = 3$. Notice that for a symmetric redundancy patterns like $(3, 3, 3, 3)$, both adaptive methods lead to the same delivery rate. As the pattern gets more asymmetric, the gap between the rates of the original and simplified adaptive methods increases. Also, observe that for some cases, the rate of the non-adaptive method increases with the storage capacity for small M/N. This shows the inefficiency of [1, Algorithm 1] to deliver redundant requests.

For the second numerical example, we use the centralized placement and plot the delivery rates resulted from the different methods versus L. Fig. 2 shows the results. Notice that the rate of original adaptive method depends not only on L, but also on the redundancy pattern. Hence, in this example, for every value of L, the delivery rate of the original adaptive method is averaged over all the redundancy patterns with L distinct requests, assuming that the requests are independent

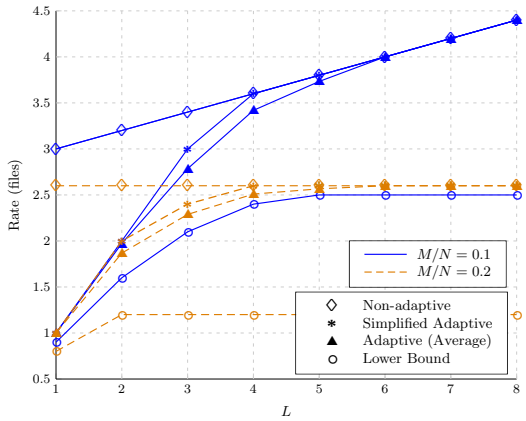

Fig. 2. Rates of different delivery schemes for $K = 8$.

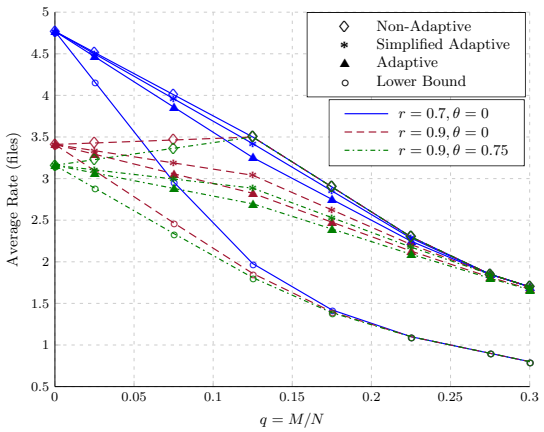

Fig. 3. Average rates of the different delivery schemes for $K = 8$.

and file popularities are uniform. Observe that the superiority of the adaptive method over the non-adaptive method of [2] is more significant for small L. In particular, we observe a sharp decrease in the adaptive delivery rate when L gets smaller than $K/2$. This suggests that the adaptive methods are considerably more effective for highly redundant requests.

We now investigate the average rates of the different delivery methods through a stochastic modelling of the dynamics of a caching network with correlated user requests. Consider a graph representation of the network where vertices represent the caches. An (undirected) edge between two vertices shows that the requests of the corresponding caches are correlated. To model the correlations, let $\mathcal{N}(k)$ denote the set of the last files requested by the neighbour caches of cache k. We assume that cache k requests a file, either based on its neighbours' previous requests with probability r, or independently with probability $1-r$. In the former case, k chooses a file from $\mathcal{N}(k)$ uniformly at random. However, when choosing independently, k picks a file n from the library based on the popularity distribution of files p_n. Hence, the chance of requesting file n by cache k is

$$\hat{p}_{n,k} = \begin{cases} r/|\mathcal{N}(k)| + (1-r)p_n, & n \in \mathcal{N}(k) \\ (1-r)p_n, & \text{otherwise} \end{cases}. \qquad (9)$$

For our simulations, we use Gibbs sampling [11, Sec. 24.2] to generate 10^4 sample vectors from the induced joint distribution of user demands. We set $K = 8$ and $N = 10^3$,

(r, θ)	$(0.7, 0)$	$(0.9, 0)$	$(0.9, 0.75)$
Maximum $\hat{\rho}_{ij}$	0.19	0.34	0.34
Average $\hat{\rho}_{ij}$	0.16	0.32	0.31
Average L	4.80	3.41	3.18

TABLE I. User requests' statistics in simulations of Fig. 3.

and assume a complete graph for the network. Further, we mainly use uniform distribution for the popularity of files. We also consider a scenario where the placement phase is performed based on a uniform popularity distribution, while the actual file popularities in the delivery phase follow a non-uniform Zipf distribution with parameter θ. Note that a Zipf distribution with $\theta = 0$ is identical to the uniform distribution, and increasing θ makes the distribution more non-uniform. We use θ and r to control the popularity distribution and the dependency level of the users' requests, respectively. To characterize the resulting correlation levels among the caches' requests in our simulations, we empirically calculate the correlation coefficients $-1 \leq \hat{\rho}_{ij} \leq 1$ [12, Section 4.1] between the requests of the different caches i and j. A larger $\hat{\rho}_{ij}$ implies a higher chance that caches i and j request the same content, which leads to more redundancy in the demand vector. Table I presents the average and the maximum $\hat{\rho}_{ij}$ over all the different i and j pairs ($i \neq j$), in our simulations.

Fig. 3 shows the resulting average delivery rates. It also shows a lower bound on the average rate calculated by averaging the lower bounds of (8) for the sample demand vectors. We observe that as requests become more correlated (larger r) and the file popularities get more non-uniform (larger θ), the adaptive method makes a larger improvement in the rate. Also, observe that the adaptive schemes are effective in decreasing the average delivery rate for $M/N < 0.25$. The improvement in the performance gap to the lower bound can be as large as 50% for specific choices of parameters.

REFERENCES

[1] M. A. Maddah-Ali and U. Niesen, "Decentralized coded caching attains order-optimal memory-rate tradeoff," *IEEE/ACM Trans. Networking*, vol. 23, pp. 1029–1040, Aug. 2015.
[2] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Trans. Inf. Theory*, vol. 60, pp. 2856–2867, May 2014.
[3] E. Bastug, M. Bennis, and M. Debbah, "Living on the edge: The role of proactive caching in 5G wireless networks," *IEEE Commun. Mag.*, vol. 52, pp. 82–89, Aug. 2014.
[4] K. Shanmugam, N. Golrezaei, A. G. Dimakis, A. F. Molisch, and G. Caire, "Femtocaching: Wireless content delivery through distributed caching helpers," *IEEE Trans. Inf. Theory*, vol. 59, pp. 8402–8413, Dec. 2013.
[5] M. Ji, A. M. Tulino, J. Llorca, and G. Caire, "On the average performance of caching and coded multicasting with random demands," in *Proc. 11th International Symposium on Wireless Communications Systems (ISWCS)*, pp. 922–926, Aug. 2014.
[6] J. Hachem, N. Karamchandani, and S. Diggavi, "Content caching and delivery over heterogeneous wireless networks," in *IEEE Conference on Computer Communications (INFOCOM)*, pp. 756–764, Apr. 2015.
[7] U. Niesen and M. A. Maddah-Ali, "Coded caching with nonuniform demands," in *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 221–226, Apr. 2014.
[8] J. Zhang, X. Lin, and X. Wang, "Coded caching under arbitrary popularity distributions," in *Proc. Information Theory and Applications Workshop (ITA)*, Feb. 2015.
[9] N. Karamchandani, U. Niesen, M. A. Maddah-Ali, and S. Diggavi, "Hierarchical coded caching," in *Proc. IEEE Int. Symp. Information Theory*, pp. 2142–2146, June 2014.
[10] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004.
[11] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.
[12] R. Peck, *Statistics: Learning from Data*. Cengage Learning, 2014.