From Centralized to Decentralized Coded Caching

Yitao Chen, Alex Dimakis
University of Texas at Austin
Austin, TX 78701 USA
Email: yitaochen@utexas.edu, dimakis@austin.utexas.edu

Karthikeyan Shanmugam
IBM Research AI
Yorktown Heights, NY 10504 USA .
Email: karthikeyanshanmugam88@gmail.com

***Abstract*—THIS PAPER IS ELIGIBLE FOR THE STUDENT PAPER AWARD.**

We consider the problem of designing decentralized schemes for coded caching. In this problem there are K users each caching M files out of a library of N total files. The question is to minimize R, the number of broadcast transmissions to satisfy all the user demands. Decentralized schemes allow the creation of each cache independently, allowing users to join or leave without dependencies. Previous work showed that to achieve a coding gain g, i.e. $R \leq K(1-M/N)/g$ transmissions, each file has to be divided into number of subpackets that is exponential in g.

In this work we propose a simple translation scheme that converts any constant rate centralized scheme into a random decentralized placement scheme that guarantees a target coding gain of g. If the file size in the original constant rate centralized scheme is subexponential in K, then the file size for the resulting scheme is subexponential in g. When new users join, the rest of the system remains the same. However, we require an additional communication overhead of $O(\log K)$ bits to determined the new user's cache state. We also show that the worst-case rate guarantee degrades only by a constant factor due to the dynamics of user arrival and departure.

A full version of this paper is accessible at: http://www.isit2018.org/

I. Introduction

Demand for wireless bandwidth has increased dramatically owing to rise in mobile video traffic [1], [2]. One of the most promising approaches for design of next generation networks (5G) is to densify deployment of small/micro/femto cell stations enabling large scale spectrum reuse. One of the main issues is that the backhaul networks required for such a dense deployment is a severe bottleneck. To alleviate this, a vast number of recent work proposed caching highly popular content at users and or at femto cell stations near users [3], [4]. These caches could be populated during off-peak time periods by predictive analytics. This caching at the 'wireless edge' is being seen as a fundamental component of 5G networks [2], [5].

Upon a cache hit, users can either immediately use or obtain the files using near-field communication from femto stations that cache content. There are various aspects of the problem that has been studied recently. Another non-trivial benefit of wireless networks is exemplified by possibility of coded transmissions leveraging cache content. One or more packets can be XORed by a macro base station and sent while users can decode the required packets by using local or near by cache content. Potentially, the benefit over and above that obtained only through cache hits can be enormous. A stylized abstract problem that explores this dimension is called the coded caching problem, introduced by Maddah-Ali and Niesen in their pioneering work [6].

In the coded caching problem, K users are managed by a single server through a noiseless broadcast link. Each user demand arises from a library of N files. Each user has a cache memory of M files. Each file consists of F subpackets. There are two phases - a placement and delivery phase. In the placement phase, every user cache is populated from packets of different files from the library. In the delivery phase, user demands are revealed (the choice could be adversarial). The broadcast agent sends a set of coded packets such that each user can decode its desired file using its cache content designed from the placement phase. The objective is to jointly design both phases such that the worst-case number of file transmissions (often called as the *rate*) is at most R. The most surprising result is that $R \leq N/M$, that is independent of the number of users can be achieved. This is also shown to be information theoretically optimal upto constant factors. There has been a lot of work [7]–[16] extending this order optimal result to various settings - demands arising from a popularity distribution, caching happening at various levels etc. .

There is another line of work that focuses on file size - the number of subpackets F required- for the original problem. They attempt to reduce the number of subpackets required for a given target rate. There have been broadly two types of coded caching schemes - a) Centralized and b) Decentralized schemes. Centralized schemes have a deterministic placement and delivery phases. Placement phase is very coordinated that when a user arrives or leaves, this requires a fresh placement of cache content throughout the system. Decentralized schemes have a random placement phase and the objective is to optimize the worst-case rate in the delivery with high probability over the randomization in the placement phase. For almost all schemes, the random cache content of a new user is independent of the rest of the system. This removes the need for system wide changes when new users arrive and leave the system. Initial centralized schemes required file sizes exponential in K to obtain constant worst case rate (we always assume ratio M/N is a constant in this work that does not scale with K). Subsequent works have explored centralized schemes that attain sub-exponential file size of constant rate. Wen linear file size for near-constant rates are feasible in theory although this requires impractically large values of K.

The original decentralized schemes required exponential file size in K even for a constant *coding gain* of g, i.e. $R \leq K/g$ w.h.p. This was the price required for decentralization in the initial scheme. Subsequent works have reduced the file size to exponentially depend on only g (the target coding gain) independent of users K. However, there are no analogues in the decentralized world for subexponential scaling in the target coding gain g for file sizes.

Our Contributions: In this work, inspired by CAIRE and leveraging ideas from balls and bins literature with power of two choices, we show the following:

1) to be done

II. Problem Setting

A. Coded Caching Problem

In this part, we formally define the coded caching problem. Consider L users that request files from a library of size N. We are mostly interested in the case when $L < N$. The N files are denoted by $W_1, \ldots, W_N$, consisting of F data packets. Each file packet belongs to a finite alphabet χ. Let $\mathcal{N} \triangleq \{1, 2, \ldots, N\}$ and $\mathcal{L} \triangleq \{1, 2, \ldots, L\}$ denote the set of files and the set of users, respectively. Each user has a cache that can store MF packets from the library, $M \in [0, N]$. In the *placement phase*, user caches are populated without knowledge of the user demands. Let ϕ_u denote the caching function for user u, which maps N files $W_1, \ldots, W_N$ into the cache content

$$Z_u \triangleq \phi_u(W_1, \ldots, W_N) \in \chi^{MF}$$

for user $u \in \mathcal{L}$. Let $\boldsymbol{Z} \triangleq (Z_1, \ldots, Z_L)$ denote the cache contents of all the L users. In the *delivery phase*, where users reveal their individual demands $\boldsymbol{d} \triangleq (d_1, \ldots, d_L) \in \mathcal{N}^L$, let ψ denote the encoding function for the server, which maps the files $W_1, \ldots, W_N$, the cache contents $\boldsymbol{Z}$, and the request $\boldsymbol{d}$ into the multicast message

$$Y \triangleq \psi(W_1, \ldots, W_N, \boldsymbol{Z}, \boldsymbol{d}),$$

sent by the server over the shared link. Let γ_u denote the decoding function at user u, which maps the multicast message Y, the cache content Z_u and the request d_u, to estimate

$$\hat{W}_{d_u} \triangleq \gamma_u(Y, Z_u, d_u)$$

of the requested file W_{d_u} of user $u \in \mathcal{L}$. Each user should be able to recover its requested file from the message received over the shared link and its cache content. Thus, we impose the successful content delivery condition

$$\hat{W}_{d_u} = W_{d_u}, \quad \forall u \in \mathcal{L}. \tag{1}$$

Given the cache size ratio M/N, the cache contents $\boldsymbol{Z}$ and the requests $\boldsymbol{d}$ of all the L users, let $R(M/N, L, \boldsymbol{Z}, \boldsymbol{d})F$ be the length the multicast message Y. Let

$$R(M/N, L, \boldsymbol{Z}) \triangleq \max_{\boldsymbol{d} \in \mathcal{N}^L} R(M/N, L, \boldsymbol{Z}, \boldsymbol{d})$$

denote the worst-case (normalized) file transmissions over the shared link. The objective of the coded caching problem is to minimize the worst-case file transmissions $R(M/N, L, \boldsymbol{Z})$. The minimization is with respect to the caching functions $\{\phi_l : l \in \mathcal{L}\}$, the encoding function ψ, and the decoding functions $\{\gamma_l : l \in \mathcal{L}\}$, subject to the successful content delivery condition in (1). A set of feasible placement and delivery strategy constitutes a coded caching scheme.

B. Two types of schemes

There are two types of coded caching schemes - a) Centralized Schemes and b) Decentralized Schemes. Centralized schemes involve deterministic placement and delivery strategies. The placement phase requires coordination of the caches and it requires system wide changes once the number of users (L) change. Decentralized schemes have random placement strategies and requires little to no coordination during the placement phase enabling users to join and leave the system easily. We further divide the decentralized schemes into two kinds in this work for the purpose of illustrating our results in contrast to existing ones.

1) *Decentralized Type A* The random set of file packets placed in any user u's cache is independent of the rest of the system requiring no coordination in the placement phase when users join the system and leave. Most of the current known (as far as the authors are aware) decentralized schemes are of this kind.
2) *Decentralized Type B* When a new user u joins, the random set of file packets placed in any users u's cache is dependent of the rest of the system. However, it does not require any change in the rest of the system. We also seek to minimize the number of bits B communicated when the new user's cache state is determined.

C. Objective

The prime focus in this work is to design Decentralized Schemes of type B where for a given expected worst-case rate (expectation is with respect to the random placement scheme) of at most $L(1 - M/N)/g$, for constant M/N, such that the file size F is kept small (as function of the coding gain g) as possible and the number of bits communicated B when users join and leave the system is minimized.

III. Preliminary

A. Centralized Schemes - Ruzsa-Szemerédi constructions

In this section, we introduce a class of centralized coded caching schemes called Ruzsa-Szemeredi schemes. We describe a specific family of bi-partite graphs call *Ruzsa Szemeredi* bipartite graphs. Then, we review an existing connection between these bipartite graphs and centralized coded caching schemes. We follow the definitions et al. [].

Definition III.1. *Consider an undirected graph $G(V, E)$. An induced matching $M \subseteq E$ is a set of edges such that a) no two edges in M share a common vertex and b) the subgraph induced by the vertices in the matching contains only the edges in M and no other edge in the original graph G.*

Definition III.2. *A bipartite graph $G([F],[K],E)$ is an (r,t)-Ruzsa-Szemeredi graph if the edge set can be partitioned into t induced matchings and the average size of these induced matchings is r.*

Now, we describe a coded caching scheme-placement and delivery phases-from the construction of a Ruzsa-Szemeredi bipartite graph. Suppose the minimum right-degree is c.

Theorem III.3. *Consider a Ruzsa-Szemerédi bipartite graph on vertex sets $[F]$ and $[K]$ such that the minimum right-degree is $c \leq F$. Then, for any $M/N \geq 1 - c/F$, we have a centralized coded caching scheme with worst case rate $R = t/F$ with system parameters (K, M, N, F).*

With a given (K, M, N, F) Ruzsa-Szemerédi bi-partite graph $G([F],[K],E)$, a F-packet coded caching scheme can be realized by Algorithm 1. In the placement phase, non-edge represents storage actions. An edge $e \in E$ between $f \in F$ and $k \in K$ is denoted by (f,k). If $(f,k) \notin E$, then file packet f of all files is stored in user k's cache. In delivery phase, an XOR of all the packets involved in an induced matching is sent. We repeat this XORing process for every induced matching. It can be shown that this policy yields a feasible delivery scheme that satisfies any demand set $\boldsymbol{d}$.

Almost all (as far as the authors are aware) known centralized coded caching schemes where placement and delivery are deterministic belong to the class of Ruzsa-Szemerédi schemes. They have been introduced in the literature before through several other equivalent formulations (like placement delivery array etc..)[17]–[22]. In the next section, we define

Algorithm 1 Ruzsa-Szemeredi based caching scheme

procedure PLACEMENT($G([F],[K],E), \{W_n, n \in \mathcal{N}\}$)
  Split each file $W_n, n \in \mathcal{N}$ into F packets, i.e., $W_n = \{W_{n,f} : f = 1, 2, \cdots, F\}$
  for $k \in [1:K]$ **do**
    $Z_k \leftarrow \{W_{n,f} : (f,k) \notin E, \forall n = 1, 2, \cdots, N\}$
  end for
end procedure
procedure DELIVERY($G([F],[K],E), \{W_n, n \in \mathcal{N}\}, \boldsymbol{d}$)
  for $s = 1, 2, \cdots, t$ **do**
    Suppose $(f_1, k_1), \ldots, (f_p, k_p)$ represents a p-sized induced matching.
    Server sends $\oplus_{j \in [p]} W_{d_{k_j}, f_j}$
  end for
end procedure

a new 'translation' mechanism that generates a decentralized scheme of type B out of an existing class of Ruzsa-Szemerédi schemes of constant rate that preserves the efficiency of file size requirements. This is the main contribution of this work.

IV. Our Decentralized Scheme

A. Translation using Balls and Bins Argument

Our objective is to specify a decentralized scheme for L users, system parameters M and N and expected worst-case rate of at most $L(1 - M/N)/g$. First, given the target coded cache gain g, the size of cache memory M, the number of files N and the number of users L, we decide an appropriate number of virtual users K. We assume that we can construct Ruzsa-Szemerédi centralized schemes for $K = f(M, N, L, g)$ (this function will be specified later), for constant M/N and worst-case rate R which is dependent only on M and N and file size requirement F. Consider the cache content of every virtual user k according to this centralized scheme. Let us denote the cache content by $C_k \in \chi^{MF}$. Please note that the cache contents of the whole centralized scheme is only virtual. We specify the random placement scheme for L real users as follows.

Placement Scheme: Now, for the real users $u \in [1:L]$, sequentially, we pick two virtual cache contents C_{u_1} and C_{u_2} at random. Now, assign the cache content of the real user u to that virtual cache content which has been least used so far. Let us denote by X_k the number of real users which store C_k.

Now, we specify a one-one correspondence to a balls and bins system. The number of distinct virtual user cache contents are the bins in the system. They are K in number. The real users corresponds to a ball. When a ball in placed in the bin, a real user (ball) is assigned the cache content of that virtual user (bin it is placed in). We can easily see that the random placement exactly corresponds to a power of two choices in a standard balls and bin argument [23], [24].

Delivery Scheme: Note that, in a system with $K' < K$ users with distinct cache contents $C_1 \ldots C_{K'}$, by using the (K, M, N) Ruzsa-Szemerédi delivery scheme with files demanded by users $k > K'$ substituted by a dummy file, it is possible to still guarantee a worst-case rate of R in the delivery phase.

We now repeatedly perform the following until all $X_k = 0$: Find a set of at most K real users with maximum number of distinct virtual cache contents. Subtract X_k corresponding to those virtual cache contents chosen by 1. Use the (K, M, N) Ruzsa-Szemerédi delivery scheme for these real users and their real demands. Clearly, the total number of worst case transmissions is at most $R * \max_k X_k$.

We summarize the decentralized scheme in Algorithm 2.

B. Analysis of the decentralized algorithm

Lemma IV.1. *The total number of worst-case file transmission of the delivery scheme in Algorithm 2 is given by:*

$$R(g, M/N, L, \{X_1 \ldots X_k\}) = R * \max_k X_k$$

where R is the worst-case rate of the (K, M, N) Ruzsa-Szemerédi coded caching scheme used in Algorithm 2.

Proof. The delivery scheme of Algorithm 1 is called with possibly dummy users and user demands at most $\max_k X_k$ times. Each call produced at most R file transmission. The proof follows from this. □

As we stated before, the placement has a direct correspondence to a choice of two balls and bins process. Here, there are L balls and K bins. In sequence, for every ball, two bins

are chosen uniformly randomly with replacement and the ball is placed in one among the chosen bins with least number of balls. From [25], we have the following lemma,

Lemma IV.2 ([25])**.** *The maximum number of balls in any bin, achieved by the choice of two policy for balls and bins problem, with L balls and K bins, $L \geq K$ is less than $L/K + \ln\ln K/\ln 2 + 9$ with probability at least $1 - O(K^{-\alpha})$, where α is a suitable positive constant.*

From Lemma IV.2, we know that $X_{\max} \leq L/K + \ln\ln K/\ln 2 + O(1)$ probability at least $1 - \frac{1}{K^{\alpha}}$ for some constant α. Therefore, have the following theorem:

Algorithm 2 Decentralized Scheme

Given M, N, L, g, let $K = f(g, M/N, L)$ (depends on constructions).
Get the cache contents $C_1, C_2 \dots C_K$ corresponding to the Ruzsa-Szemerédi placement scheme (in Algorithm 1) with parameters K, M, N, F.
procedure SAMPLING($\mathcal{C}, L, \{X_k\}, u$)
Uniformly sample a cache content from $\{C_1 \dots C_k\}$ for the cache of user u twice with replacement, i.e., C_{u_1}, C_{u_2}.
if $X_{u_1} \leq X_{u_2}$ **then**
$X_{u_1} \leftarrow X_{u_1} + 1$.
$Z_u \leftarrow C_{u_1}$
else
$X_{u_2} \leftarrow X_{u_2} + 1$.
$Z_u \leftarrow C_{u_2}$
end if
end procedure
procedure PLACEMENT(M, N, L, g)
Initialize $X_k = 0,\ \forall k \in [1:K]$.
for $u = 1, 2, \cdots, L$ **do**
Sampling($\mathcal{C}, L, \{X_k\}, u$)
end for
end procedure
procedure DELIVERY(M, N, L, g)
Let $S_k \leftarrow X_k$. Let $\mathcal{L} \leftarrow \{1 \dots L\}$.
while $\max_k S_k > 0$ **do**
Find a maximal subset $F \subset \mathcal{L}$ such that cache contents of all real users assigned in F are distinct.
if $|F| < K$ **then**
Use Delivery subroutine of Algorithm 1 to satisfy demands of users in F using a (K, M, N) Ruzsa-Szemerédi Scheme. This can be done by substituting packets belonging to file demands of users outside set F (since $|F| < K$) by packets from a dummy file known to all users.
else
Use Delivery subroutine of Algorithm 1 to satisfy demands of users in F ($|F| = K$) using a (K, M, N) Ruzsa-Szemerédi Scheme.
end if
end while
end procedure

Theorem IV.3. *Suppose there exists a Ruzsa-Szemerédi centralized scheme, with constant (independent of K) worst-case rate R_c, constant cache size ratio $\frac{M}{N}$, and subpacketization level $F_c = O(2^{K^{\delta} f(R_c, M/N)})$. Then the scheme in Algorithm 2 that uses this centralized scheme has target gain g, i.e. the number of file transmissions in the worst case is rate $R_d = \frac{L(1-M/N)}{g} + O(\ln\ln g)$ w.h.p. The subpacketization level required is $F_d = O(2^{g^{\delta} h(R_c, M/N)})$ where $h(R_c, M/N) = f(R_c, M/N) R_c^{\delta}/(1 - M/N)^{\delta}$. To obtain the scheme, we set $K = gR_c/(1 - M/N)$ in Algorithm 2.*

Proof. Consider a Ruzsa-Szemerédi centralized scheme with constant R_c, according to Lemma IV.1 and IV.2, the rate in the corresponding decentralized scheme is

$$\begin{aligned} R_d &= R_c \max_k X_k \\ &\leq R_c \left(\frac{L}{K} + \frac{\ln\ln K}{\ln 2} \right) \\ &= \frac{L(1 - M/N)}{g} + O(\ln\ln g). \end{aligned}$$

Here, we set $K = gR_c/(1 - M/N)$. Substituting $K = gR_c/(1 - M/N)$ into the number of subpackets required in the centralized scheme F_c, then that required for Algorithm 2 is:

$$F_d = O(2^{g^{\delta} h(R_c, M/N)}),$$

where $h(R_c, M/N) = f(R_c, M/N) R_c^{\delta}/(1 - M/N)^{\delta}$. □

From [21], we have the following lemma,

Lemma IV.4 ([21])**.** *There exists an (r, t)-Ruzsa Szemerédi graph $G([F], [K], E)$ with $t = \binom{n}{a+2}, F = \binom{n}{a}, K = \binom{n}{2}$ for some a and $n = \lambda a$ for some constant $\lambda > 1$, it holds that $R = t/F = \binom{n}{a+2}/\binom{n}{a} \approx (\lambda - 1)^2$ and $M/N = (\binom{n}{a} - \binom{n-2}{a})/\binom{n}{a} \approx \frac{2\lambda - 1}{\lambda^2}$ and by Stirling's formula we have*

$$\begin{aligned} F = \binom{n}{\lambda^{-1} n} &= \frac{1 + o(1)}{\sqrt{2\pi\lambda^{-1}(1 - \lambda^{-1})n}} \cdot 2^{nH(\lambda^{-1})} \\ &= O(K^{-1/4} \cdot 2^{\sqrt{2K} H(\lambda^{-1})}), \end{aligned}$$

where $H(x) = -x\log_2 x - (1-x)\log_2(1-x)$ for $0 < x < 1$ is the binary entropy function. It is easy to see that under such choice of parameters, R and M/N are both constants independent of K and F grows sub-exponentially with K.

Apply Theorem IV.3 with Lemma IV.4, we have the following Corollary,

Corollary IV.5. *For the Ruzsa Szemerédi centralized scheme in Lemma IV.4, we have a corresponding decentralized scheme with $R_d = \frac{L(1-M/N)}{g} + O(g\ln\ln g)$ and subpacketization level $F = \tilde{O}(2^{g^{1/2}})$, where $\tilde{O}$ means $\log g, \mathrm{poly}(\lambda)$ terms are omitted.*

For non-constant rate and constant cache size ratio M/N, we have the following theorem.

Theorem IV.6. *Suppose there exists a Ruzsa-Szemerédi centralized scheme, with rate $R_c \leq K^{\delta}$, $\delta \in (0,1)$ and constant cache size ratio M/N, and subpacketization level $F_c = poly(K)$. Then the scheme in Algorithm 2 that uses this centralized scheme has target gain g, i.e., the number of file transmissions in the worst case is rate $R_d = \frac{L(1-M/N)}{g} + O(g^{1/(1-\delta)} \ln \ln g)$ w.h.p. The subpacketization level required is $F_d = poly(\frac{g}{(1-M/N)})^{1/(1-\delta)}$. To obtain the scheme, we set $K = (\frac{g}{1-M/N})^{1/(1-\delta)}$.*

C. Overhead analysis for the dynamic version of the decentralized scheme

We consider the dynamics of user arrival and departure. We specify the changes in the placement scheme when users arrive and depart. When a user leaves the system, the user's cache content is deleted and if the user had cached C_k (Recall from Section IV-A, that this is the cache content of the k-th virtual user from Section IV-A), X_k is decreased by 1. When a new user u joins the system, then the subroutine $\mathrm{Sampling}\,(\mathcal{C}, L, \{X_k\}, u)$ from Algorithm 2 is executed to determine the cache content of user u (i.e. Z_u). The comparison between X_{u_1} and X_{u_2} in the procedure $\mathrm{Sampling}(\cdot)$ involves an additional $3 \log K$ bits of communication overhead between user u and the central server. Note that, the dynamics of user arrivals and departure does not change the cache contents of users already in the system.

The worst-case rate during delivery is directly proportional to $\max_k X_k$ according to Lemma IV.1. Now, we will show that despite the dynamics $\max_k X_k$ remains the same upto constant factors with high probability provided the number of adversarial departures and arrivals is bounded. We recall that the real users represent a ball and the virtual users and the distinct cache contents C_i represent the bins. $\max_k X_k$ is the size of the maximum bin.

For the analysis, let us first define the balls and bins process with adversarial deletions/additions. Consider the polynomial time process where in the first L steps, a new ball is inserted into the system (this is when the system is initiated with L users). At each subsequent time step, either a ball is removed or a new ball is inserted in the system, provided that the number of balls present in the system never exceeds L. Each new ball inserted in the system choose 2 possible destination bins independently and uniformly at random, and is placed in the least full of these bins. Suppose that an adversary specifies the full sequence of insertions and deletions of balls in advance, without knowledge of the random choice of the new balls that will be inserted in the system (i.e., suppose we have an oblivious adversary).

Combing Theorem 1 in [24] with Theorem 3.7 in [23], we have the following theorem,

Theorem IV.7. *For any fixed constant c_1 and c_2 such that $K^{c_2} > L$, if the balls and bins process with adversarial deletions runs for at most K^{c_2} times steps, then the maximum load of a bin during the process is at most $O(L/K) + \ln \ln K / \ln 2 + O(c_1 + c_2)$, with probability at least $1 - o(1/K^{c_1})$.*

Proof. Please refer to the appendix in the full version for a self-contained proof that extend results in previous work. □

We note that all the results in Section IV-B follows with at-most a constant factor correction to the worst-case rate.

V. Conclusion

In this work we show a simple translation scheme that converts any constant rate centralized scheme into a random decentralized placement scheme that guarantees a target coding gain of g. We show the worst-case rate due to the dynamics of user arrival and departure degrades only by a constant factor. Interesting future direction is to improve the degradation from constant factor to an additive poly-log factor.

References

[1] C. V. Mobile, "Global mobile data traffic forecast update 2010-2015," *Cisco White Paper*, 2011.

[2] F. Boccardi, R. W. Heath, A. Lozano, T. L. Marzetta, and P. Popovski, "Five disruptive technology directions for 5g," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 74–80, 2014.

[3] K. Shanmugam, N. Golrezaei, A. G. Dimakis, A. F. Molisch, and G. Caire, "Femtocaching: Wireless content delivery through distributed caching helpers," *IEEE Transactions on Information Theory*, vol. 59, no. 12, pp. 8402–8413, 2013.

[4] 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.

[5] G. Paschos, E. Bastug, I. Land, G. Caire, and M. Debbah, "Wireless caching: Technical misconceptions and business barriers," *IEEE Communications Magazine*, vol. 54, no. 8, pp. 16–22, 2016.

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

[7] ——, "Decentralized coded caching attains order-optimal memory-rate tradeoff," *arXiv preprint arXiv:1301.5848*, 2013.

[8] U. Niesen and M. A. Maddah-Ali, "Coded caching with nonuniform demands," *arXiv preprint arXiv:1308.0178*, 2013.

[9] M. Ji, A. M. Tulino, J. Llorca, and G. Caire, "Order-optimal rate of caching and coded multicasting with random demands," *IEEE Transactions on Information Theory*, vol. 63, no. 6, pp. 3923–3949, June 2017.

[10] ——, "Caching and coded multicasting: Multiple groupcast index coding," in *Signal and Information Processing (GlobalSIP), 2014 IEEE Global Conference on*. IEEE, 2014, pp. 881–885.

[11] ——, "Caching-aided coded multicasting with multiple random requests," in *Information Theory Workshop (ITW), 2015 IEEE*. IEEE, 2015, pp. 1–5.

[12] N. Karamchandani, U. Niesen, M. A. Maddah-Ali, and S. N. Diggavi, "Hierarchical coded caching," *IEEE Transactions on Information Theory*, vol. 62, no. 6, pp. 3212–3229, 2016.

[13] M. Ji, A. M. Tulino, J. Llorca, and Caire, "Caching in combination networks," in *Proc. IEEE Asilomar Conf. on Signals, Systems, and Computers (ACSSC)*. IEEE, 2015, pp. 695–699.

[14] P. Hassanzadeh, A. Tulino, J. Llorca, and E. Erkip, "Correlation-aware distributed caching and coded delivery," *Proc. IEEE Information Theory Workshop (ITW)*, 2016.

[15] ——, "Rate-memory trade-off for the two-user broadcast caching network with correlated sources," *Proc. IEEE International Symposium on Information Theory (ISIT)*, 2017.

[16] A. Cacciapuoti, M. Caleffi, M. Ji, L. J., and A. Tulino, "Speeding up future video distribution via channel-aware caching-aided coded multicast," *IEEE JSAC*, vol. 34, no. 8, pp. 2207–2218, 2016.

[17] Q. Yan, M. Cheng, X. Tang, and Q. Chen, "On the placement delivery array design for centralized coded caching scheme," *IEEE Transactions on Information Theory*, vol. 63, no. 9, pp. 5821–5833, Sept 2017.

[18] Q. Yan, X. Tang, Q. Chen, and M. Cheng, "Placement delivery array design through strong edge coloring of bipartite graphs," *arXiv preprint arXiv:1609.02985*, 2016.

[19] M. Cheng, Q. Yan, X. Tang, and J. Jiang, "Optimal placement delivery arrays with minimum number of rows," *arXiv preprint arXiv:1703.01548*, 2017.
[20] M. Cheng, J. Jiang, Q. Yan, and X. Tang, "Coded caching schemes for flexible memory sizes," *arXiv preprint arXiv:1708.06650*, 2017.
[21] C. Shangguan, Y. Zhang, and G. Ge, "Centralized coded caching schemes: A hypergraph theoretical approach," *arXiv preprint arXiv:1608.03989*, 2016.
[22] L. Tang and A. Ramamoorthy, "Low subpacketization schemes for coded caching," *arXiv preprint arXiv:1706.00101*, 2017.
[23] Y. Azar, A. Z. Broder, A. R. Karlin, and E. Upfal, "Balanced allocations," *SIAM journal on computing*, vol. 29, no. 1, pp. 180–200, 1999.
[24] R. Cole, A. Frieze, B. Maggs, M. Mitzenmacher, A. Richa, R. Sitaraman, and E. Upfal, "On balls and bins with deletions," *Randomization and Approximation Techniques in Computer Science*, pp. 145–158, 1998.
[25] P. Berenbrink, A. Czumaj, A. Steger, and B. Vöcking, "Balanced allocations: the heavily loaded case," in *Proceedings of the thirty-second annual ACM symposium on Theory of computing*. ACM, 2000, pp. 745–754.

APPENDIX

For a vector $\boldsymbol{v} = (v_1, v_2, \ldots)$, let $P_2(\boldsymbol{v})$ be the following process: at time steps 1 through L, L balls are placed into K bins sequentially, with each ball going into the least loaded of 2 bins chosen independently and uniformly at random. After these balls are placed, deletions and insertions alternate, so that at each subsequent time step $L + j$, first the ball inserted at time v_j is removed, and then a new ball is placed into the least loaded of 2 bins chosen independently and uniformly at random. (Actually we do not require this alternation; the main point is that we have a bound, L, on the number of balls in the system at any point. The alternation merely makes notation more convenient.)

We assume the vector $\boldsymbol{v}$ is suitably defined so that at each step an actual deletion occurs; that is, the v_j are unique and $v_j \leq L + j - 1$. Otherwise $\boldsymbol{v}$ is arbitrary, although we emphasize that it is chosen before the process begins and does not depend on the random choices made during the process.

We adopt some of the notation of [23]. Each ball is assigned a fixed *height* upon entry, where the height is the number of balls in the bin, including itself. The height of the ball placed at time t is denoted by $h(t)$. The load of a bin at time t refers to the number of balls in the bin at that time. We let $\mu_{\geq k}(t)$ denote the number of balls hat have height at least k at time t, and $\nu_{\geq k}(t)$ be the number of bins that have load at least k at time t. Note that if a bin has load k, it must contain some ball of height at least k. Hence $\mu_{\geq k}(t) \geq \nu_{\geq k}(t)$ for all times t. Finally, $B(L, p)$ refers to a binomially distributed random variable based on L trials each with probability p of success.

We extends the original Theorem 3.7 of [23] and Theorem 1 of [24], by determining a distribution on the heights of the balls that holds for polynomially many steps, regardless of which L balls are in the system at any point in time.

Let $\mathcal{E}_i$ be the event that $\nu_{\geq i}(t) \leq \beta_i$ for time steps $t = 1, \ldots, T$, where the β_i will be revealed shortly. We want to show that at time t, $1 \leq t \leq T$,

$$\Pr(\mu_{\geq i+1} > \beta_{i+1} | \mathcal{E}_i)$$

is sufficiently small. That is, given $\mathcal{E}_i$, we want $\mathcal{E}_{i+1}$ to hold as well. This probability is hard to estimate directly. However, we know that since the 2 choices for a ball are independent, we have

$$\Pr(h(t) \geq i + 1 | \nu_{\geq i}(t-1)) = \frac{(\nu_{\geq i}(t-1))^2}{K^2}.$$

We would like to bound for each time t the distribution of the number of time steps j such that $h(j) \geq i + 1$ and the ball inserted at time step j has not been deleted by time t. In particular, we would like to bound this distribution by a binomial distribution over L events with success probability $(\beta_i/K)^2$. But this is difficult to do directly as the events are not independent.

Instead, we fix i and define the binary random variables Y_t for $t = 1, \ldots, T$, where

$$Y_t = 1 \text{ iff } h(t) \geq i + 1 \text{ and } \nu_{\geq i}(t-1) \leq \beta_i.$$

The value Y_t is 1 if and only if the height of the ball t is at least $i+1$ despite the fact that the number of boxes that have load at least i is currently below β_i.

Let ω_j represent the choices available to the jth ball. Clearly,

$$\Pr(Y_t = 1|\omega_1, \ldots, \omega_{t-1}, v_1, \ldots, v_{t-L}) \leq \frac{\beta_i^2}{K^2} \triangleq p_i.$$

Consider the situation immediately after a time step t' where a new ball has entered the system. Then there are L balls in the system, that entered at times $u_1, u_2, \ldots, u_L$. Let $I(t')$ be the set of times $u_1, u_2, \ldots, u_L$. Then

$$\sum_{t \in I(t')} Y_t = \sum_{i=1}^{L} Y_{u_i};$$

that is, the summation over $I(t')$ is implicitly over the values of Y_t for the balls in the system at time t'.

We may conclude that at any time $t' \leq T$

$$\Pr\left(\sum_{t \in I(t')} Y_t \geq k\right) \leq \Pr(B(L, p_i) \geq k). \tag{2}$$

Observe that conditioned on $\mathcal{E}_i$, we have $\mu_{\geq i+1}(t') = \sum_{t \in I(t')} Y_t$. Therefore

$$\Pr(\mu_{\geq i+1}(t') \geq k|\mathcal{E}_i) = \Pr\left(\sum_{t \in I(t')} Y_t \geq k|\mathcal{E}_i\right) \tag{3}$$

$$\leq \frac{\Pr(B(L, p_i) \geq k)}{\Pr(\mathcal{E}_i)} \tag{4}$$

Thus:

$$\Pr(\neg\mathcal{E}_{i+1}|\mathcal{E}_i) \leq \frac{T \Pr(B(L, p_i) \geq k)}{\Pr(\mathcal{E}_i)}$$

Since

$$\Pr(\neg\mathcal{E}_{i+1}) \leq \Pr(\neg\mathcal{E}_{i+1}|\mathcal{E}_i) \Pr(\mathcal{E}_i) + \Pr(\neg\mathcal{E}_i),$$

we have

$$\Pr(\neg\mathcal{E}_{i+1}) \leq T \Pr(B(L, p_i) \geq k) + \Pr(\neg\mathcal{E}_i). \tag{5}$$

We can bound large deviations in the binomial distribution with the formula

$$\Pr(B(L, p_i) \geq e p_i L) \leq e^{-p_i L}. \tag{6}$$

We may then set $\beta_x = K^2/2eL$, $x = \lceil eL/K \rceil$ [23], and subsequently

$$\beta_i = eL\frac{\beta_{i-1}^2}{K^2} \text{ for } i > x.$$

Note that the β_i are chosen so that $\Pr(B(L, p_i) \geq \beta_{i+1}) \leq e^{-p_i L}$.

With the choices $\mathcal{E}_x$, $x = \lceil eL/K \rceil$ holds [23], as there cannot be more than $K^2/2eL$ bins with x balls. For $i \geq 1$,

$$\Pr(\neg\mathcal{E}_{x+i}) \leq \frac{T}{K^{c_1+c_2+1}} + \Pr(\neg\mathcal{E}_{x+i-1})$$

$$= \frac{1}{K^{c_1+1}} + \Pr(\neg\mathcal{E}_{x+i-1}),$$

provided that $p_i L \geq (c_1 + c_2 + 1)\ln K$.

Let i^* be the smallest value for which $p_{i^*-1}L \leq (c_1 + c_2 + 1)\ln K$. Note that

$$\beta_{i+x} = \frac{K}{2^{2^i}}(\frac{K}{Le}) \leq \frac{K}{2^{2^i}}, \tag{7}$$

so $i^* = \ln\ln K/\ln 2 + O(1)$. Then go through the standard tail bound technique in [23] for $i \geq i^* + 1$, we obtain that,

$$\Pr(\mu_{\geq x+i^*+O(c_1+c_2)} \geq 1) = O(\frac{1}{K^{c_1+c_2+1}}).$$

So the probability that the maximum load is less than $L/K + \ln\ln K/\ln 2 + O(c_1 + c_2)$ is bounded by

$$\Pr(\neg\mathcal{E}_{x+i^*+O(c_1+c_2)}) \leq \sum_{i=1}^{i^*} \frac{1}{K^{c_1+1}} + O(\frac{1}{K^{c_1+1}})$$

$$\leq O(\frac{1}{K^{c_1+1}}).$$