Caching with Partial Matching Under Zipf Demands

Jad Hachem  
UCLA  
jadhachem@ucla.edu

Nikhil Karamchandani  
IIT Bombay  
nikhilk@ee.iitb.ac.in

Sharayu Moharir  
IIT Bombay  
sharayum@ee.iitb.ac.in

Suhas Diggavi  
UCLA  
suhas@ee.ucla.edu

Abstract—We study the caching problem when we are allowed to match each user to one of a subset of caches after its request is revealed. We focus on non-uniformly popular content, specifically when the file popularities obey a Zipf distribution. We study two extremal schemes, one focusing on coded server transmissions while ignoring matching capabilities, and the other focusing on adaptive matching while ignoring potential coding opportunities. We derive the rates achieved by these schemes and characterize the regimes in which one outperforms the other. We also compare them to information-theoretic outer bounds, and finally propose for certain cases a hybrid scheme that generalizes ideas from the two schemes and performs at least as well as either of them in most memory regimes.

I. INTRODUCTION

In modern content distribution networks, caching is a technique that places popular content at nodes close to the end users in order to reduce the overall network traffic. In [1], a new “coded caching” technique was introduced for broadcast networks. This technique places different content in each cache, and takes advantage of these differences to send a common coded broadcast message to multiple users at once. This was shown not only to greatly reduce the network load compared to traditional uncoded techniques, but also to be approximately optimal in general.

In [1] as well as many other works in the literature [2], [3], [4], [5], [6], a key assumption is that users are pre-fixed to specific caches; see also [7], [8] for a survey of related works. More precisely, each user connects to a specific cache before it requests a file from the content library. This assumption was relaxed in [9], [10] where the system is allowed to choose a matching of users to caches after the users make their requests, while respecting a per-cache load constraint. In particular, after each user requests a file, any user could be matched to any cache as long as no cache had more than one user connected to it. In this adaptive matching setup, it was shown under certain request distributions that a coded delivery, while approximately optimal in the pre-fixed matching case, is unnecessary. Indeed, it is sufficient to simply store complete files in the caches, and either connect a user to a cache containing its file or directly serve it from the server.

The above dichotomy indicates a fundamental difference between the system with completely pre-fixed matching and the system with full adaptive matching. In this paper, we consider a “partial adaptive matching” setup, i.e., a setup where users can be matched to any cache belonging to a subset of caches, which we first studied in [11]. This can arise when, for instance, only some caches are close enough to a user to ensure a potential reliable connection. To make matters simple, we assume that the caches are partitioned into equal clusters, and each user can be matched to any cache within a single cluster, as illustrated in Fig. 1. This setup generalizes both setups considered above: on one extreme, if each cluster consisted of only a single cache, then the setup becomes the pre-fixed matching setup of [1]; on the other extreme, if all caches belonged to a single cluster, then we get back the total adaptive matching setup from [9], [10].

In [11], we analyzed this setup in the case where all the files in the library were equally popular. While this was useful for an initial understanding of the problem, such uniform popularity is rare in practice. In this paper, we focus on the more relevant case when the popularity obeys a power law, specifically a Zipf law [12]. We analyze how the coded caching scheme, useful in the pre-fixed matching case, and the adaptive matching scheme, useful in the full adaptive matching case, would perform if adapted to this setup. We compare the two schemes with each other, characterizing the regimes in which one is better than the other. We then compare them with information-theoretic outer bounds, proving that the schemes are approximately optimal in certain regimes. Finally, for a subclass of Zipf distributions, we introduce a hybrid scheme that generalizes ideas from both schemes, thus combining the matching benefits with the coding gains, and that performs as well as either scheme in most memory regimes.

The rest of this paper is organized as follows. Section II precisely describes the problem setup. We present the main results in Section III which include the rates achieved by

Jad Hachem’s and Suhas Diggavi’s work was supported in part by NSF Grant 1423271. Nikhil Karamchandani’s research was funded in parts by an Inspire Faculty Fellowship from the Department of Science and Technology, Government of India, a seed grant from IIT Bombay, and an Indo-French grant “Machine learning for network analytics”. 
the schemes as well as statements of approximate optimality. Finally, Section [V] describes the hybrid scheme. Detailed proofs are given in the appendices.

II. PROBLEM SETUP

Consider the system depicted in Fig. 1. A server holds $N$ files $W_1, \ldots, W_N$ of size $F$ bits each. There are $K$ caches of capacity $MF$ bits, equivalently $M$ files, each. The caches are divided into $K/d$ clusters of size $d$, where $d$ is assumed to divide $K$. For every $n \in \{1, \ldots, N\}$ and every $c \in \{1, \ldots, K/d\}$, there are $u_n(c)$ users accessing cluster $c$ and requesting file $W_n$. We refer to the numbers $\{u_n(c)\}_{n,c}$ as the request profile and will often represent the request profile as a vector $u$ for convenience.

As with standard coded caching setups, a placement phase occurs before the request profile is revealed during which information about the files is placed in the caches, and a delivery phase occurs after the request profile is known during which a broadcast message is sent to all users to satisfy their demands. In our setup, in addition to the usual placement and delivery phases, there is an intermediate phase that we call the matching phase. The matching phase occurs before the delivery phase but after the request profile has been revealed. During the matching phase, each user is matched to a single cache within its cluster, with the constraint that no more than one user can be matched to a cache. If there are fewer caches than users in one cluster, then some users will be unmatched.

In this paper, we focus on the case where the numbers $u_n(c)$ are independent Poisson random variables with parameter $\rho dp_n$, where $\rho \in (0, 1/2)$ is some fixed constant and $p_1, \ldots, p_N$ is the popularity distribution of the files, with $p_n \geq 0$ and $p_1 + \cdots + p_N = 1$. Thus $p_n$ represents the probability that a fixed user will request file $W_n$. We particularly focus on the case where the files follow a Zipf law, i.e., $p_n \propto n^{-\beta}$ where $\beta \geq 0$ is the Zipf parameter. Note that the expected total number of users in the system is $\rho K$.

For a given request profile $u$, let $R_u$ denote the rate of the broadcast message required to deliver to all users their requested files. For any cache memory $M$, our goal is to minimize the expected rate $R = \mathbb{E}_u[R_u]$. Specifically, we are interested in $R^*$ defined as the smallest $R$ over all possible strategies. Furthermore, we assume that there are more files than caches, i.e., $N \geq K$, which is the case of most interest. We also, for analytical convenience, focus on the case where the cluster size $d$ grows at least as fast as $\log K$. More precisely, we assume

$$d \geq \left[2(1 + t_0)/\alpha\right] \log K,$$

where $\alpha = -\log(2pe^{1-2\rho})$ and $t_0 > 0$ is some constant. Note that $\alpha > 0$. Other than analytical convenience, the reason for such a lower bound on $d$ is that, when $d$ is too small, the Poisson request model adopted in this paper is no longer suitable. Indeed, if for example $d = 1$, then with high probability a significant fraction of users will not be matched to any cache, leading to a rate proportional to $K$ even with infinite cache memory.

Finally, we will frequently use the helpful notation $[x]^+ = \max\{x, 0\}$ for all real numbers $x$.

III. MAIN RESULTS

The setup we consider is a generalization of the pre-fixed matching setup (when $d = 1$) and the maximal adaptive matching setup (when $d = K$). From the literature, we know that different strategies are required for these two extremes: one using a coded delivery when $d = 1$, and one using adaptive matching when $d = K$. Therefore, there must be some transition in the suitable strategy as the cluster size $d$ increases from one to $K$.

The goal of this paper is to gain some understanding of this transition. To do that, we first adapt and apply the strategies suitable for the two extremes to our intermediate case. These strategies will exclusively focus on one of coded delivery and adaptive matching, and we will hence refer to them as “Pure Coded Delivery” (PCD) and “Pure Adaptive Matching” (PAM). In particular, PCD will perform an arbitrary matching and apply the coded caching scheme from [3], [4], whereas PAM will apply a matching scheme similar to [9], [10] independently on each cluster and serve unmatched requests directly, ignoring any coding opportunities. We then compare PCD and PAM in various regimes and evaluate them against information-theoretic outer bounds.

Regardless of the value $\beta$ of the Zipf parameter, we find that PCD tends to perform better than PAM when the cache memory $M$ is small, while PAM is superior to PCD when $M$ is large. The particular threshold of $M$ where PAM overtakes PCD obeys an inverse relation with the cluster size $d$. Thus when $d$ is small, PCD is the better choice for most memory values, whereas when $d$ is large, PAM performs better for most memory values. This observation agrees with previous results on the two extremes $d = 1$ and $d = K$, and it is illustrated in Fig. 2 and 3 and made precise in the theorems that follow.

While most of the analysis assumes general values for $K$, $N \geq K$, $d$ (except for (1)), and $M$, it will nevertheless be useful to sometimes compare PCD and PAM under the restriction that these parameters all scale as powers of $K$. This can provide some high-level insights into the different regimes where PCD or PAM dominate, while ignoring sub-polynomial factors such as $\log N$, thus simplifying the analysis. During this polynomial-scaling-with-$K$ analysis—which we will call poly-$K$ analysis as a shorthand—we will assume that $N = K^\nu$, $d = K^\delta$, and $M = K^\mu$, where $\nu \geq 1$; $\delta \in (0, 1]$; $\mu \in [0, \nu]$.

To proceed, we will separately consider two regimes for the Zipf popularity: a shallow Zipf case in which $\beta \in [0, 1)$, and a steep Zipf case where $\beta > 1$.

\[1\]The request model used in the literature when $d = 1$ is usually not the Poisson model used here. Instead, a multinomial model is used in which the total number of users is always fixed. As mentioned at the end of Section [II] the Poisson model is not suitable in that case. However, the results from the literature are still very relevant to this paper.

\[2\]The case $\beta = 1$ is a special case that usually requires separate handling. We skip it in this paper, and analyzing it is part of our on-going work.
A. Shallow Zipf: $\beta \in [0, 1)$

In [11], we studied this problem when the files obeyed a uniform popularity, i.e., when $\beta = 0$. In this paper, we show that the case $\beta \in [0, 1)$ is very similar to the uniform case. Indeed, the results from [11] can be generalized to all $\beta \in [0, 1)$ with only a constant-factor difference.

The next theorem gives the rate achieved by PCD.

**Theorem 1.** When $\beta \in [0, 1)$, the PCD scheme can achieve for all $M$ an expected rate of

$$\bar{R}_{\text{PCD}} = \min \left\{ \rho K, \left[ \frac{N}{\pi^*} - 1 \right]^+ + \frac{K - t_0}{\sqrt{2\pi}} \right\}.$$ 

Theorem 1 can be proved by directly applying any suitable coded caching strategy [3], [4], [5] along with an arbitrary uniform popularity, i.e., when $A$. Shallow Zipf: $\beta \in [0, 1)$.

Theorem 2. When $\beta \in [0, 1)$, the PAM scheme can achieve an expected rate of

$$\bar{R}_{\text{PAM}} = \begin{cases} \rho K, & \text{if } M = O(N/d); \\ \min \left\{ \rho K, K M e^{-z d M / N} \right\}, & \text{if } M = \Omega(N/d), \end{cases}$$

where $z = (1 - \beta)(1 + \rho)/(2\rho) > 0$ with $h(x) = x \log x + 1 - x$.

Theorem 2 can be proved using a similar argument to [9]: the idea is to replicate each file across the caches in each cluster, and match each user to a cache containing its requested file. The detailed proof is given in Appendix A.

Notice that PAM can achieve a rate of $o(1)$ when $d M > \Omega(N \log N)$. Recall that we have imposed a service constraint of one user per cache in our setup. If we instead allow multiple users to access the same cache, then it can be shown that a rate of $o(1)$ can be achieved if and only if $d M > (1 - o(1)) N$. Consequently, the cache service constraint increases this memory threshold by at most a logarithmic factor.

The rates of PCD and PAM are illustrated in Fig. 2 for the $\beta \in [0, 1)$ case. We can see that there is a memory threshold $M_0$, with $M_0 = \Omega(N/d)$ and $M_0 = O((N/d) \log N)$, such that PCD performs better than PAM for $M < M_0$ while PAM is superior to PCD for $M > M_0$. Using a poly-$K$ analysis, we can ignore the $\log N$ term and obtain the following result, illustrated in Fig. 3.

**Theorem 3.** When $\beta \in [0, 1)$, and considering only a polynomial scaling of the parameters with $K$, PCD outperforms PAM in the regime

$$\mu \leq \nu - \delta,$$

while PAM outperforms PCD in the opposite regime, where $N = K^\nu$, $d = K^\nu$, and $M = K^\nu$.

Note that in some cases PCD and PAM perform equally well, such as when $\mu = \nu$. However, these are usually edge cases and most of the regimes in Theorem 3 are such that one scheme strictly outperforms the other.

Interestingly, under the poly-$K$ analysis, the memory regime where PAM becomes superior to PCD is the regime where PAM achieves a rate of $o(1)$, for any $d$.

So far, we have seen that the two memory regimes $M < O(N/d)$ and $M > \Omega((N/d) \log N)$ require very different schemes: one focusing on coding and the other on matching. In Section IV, we introduce a universal scheme for the shallow Zipf case that generalizes ideas from both PCD and PAM. It is a Hybrid Coding and Matching (HCM) scheme that combines the benefits of adaptive matching within clusters with the coded caching gains across clusters. We state the rate HCM achieves in Theorem 8 and then show that it can perform at least as well as either of PCD and PAM in most memory regimes, namely when $M < O(N/d)$ or $M > \Omega((N/d) \log K)$.

B. Steep Zipf: $\beta > 1$

When $\beta > 1$, we restrict ourselves to the case where $d$ is some polynomial in $K$ for convenience. The following theorems give the rates achieved by PCD and PAM, illustrated in Fig. 4.
Fig. 4. Rates achieved by PCD and PAM in the $\beta > 1$ case. Again, this plot is not numerically generated but is drawn approximately for illustration purposes.

**Theorem 4.** When $\beta > 1$, the PCD scheme can achieve an expected rate of

$$R^\text{PCD} = \begin{cases} K^{1/\beta} & \text{if } 0 \leq M < 1; \\ \left(\frac{K M^{1/\beta}}{N} - 1\right) + \frac{K - 1}{\sqrt{2\pi}} & \text{if } 1 \leq M < N^{\beta}/K; \\ \left(\frac{N}{M} - 1\right) + \frac{K - 1}{\sqrt{2\pi}} & \text{if } M \geq N^{\beta}/K. \end{cases}$$

Much like Theorem 1, Theorem 4 follows from directly applying the coded caching strategy from [3], [4]. Again, the $K^{-\delta}$ term represents the expected number of unmatched users, derived in Lemma 1 in Appendix A.

**Theorem 5.** When $\beta > 1$, the PAM scheme can achieve an expected rate of

$$R^\text{PAM} = \begin{cases} O\left(\frac{K}{(d M)^{\frac{1}{\beta}}}, K^\frac{1}{\beta}\right) & \text{if } M = O(N/d); \\ o(1) & \text{if } M = \Omega\left(\frac{N \log N}{d}\right). \end{cases}$$

The proof of Theorem 5, given in Appendix C, follows along the same lines as [10] and involves a generalization from $d = K$ to any polynomial $d = K^\delta$, $0 < \delta \leq 1$. The idea is to replicate the files across the caches in the cluster, placing in Theorems 4 and 5 using a poly-

scaling of the parameters with $K$.

As with the $\beta \in [0, 1)$ case, we notice that PCD is the better choice when $M$ is small, while PAM is the better choice when $M$ is large. In fact, by comparing the rate expressions in Theorems 4 and 5 using a poly-K analysis, we obtain the following theorem describing the regimes for which either of PCD or PAM is superior to the other. The theorem is illustrated in Fig. 5 and proved at the end of this subsection.

**Theorem 6.** When $\beta > 1$, and considering only a polynomial scaling of the parameters with $K$, PCD outperforms PAM in the regime

$$\mu \leq \min\{\nu - \delta, (1 - \beta \delta)/(\beta - 1)\},$$

while PAM outperforms PCD in the opposite regime, where $N = K^\nu$, $d = K^\delta$, and $M = K^\mu$.

When comparing Theorems 3 and 6, we notice that the case $\beta > 1$ has the added constraint $\mu < (1 - \beta \delta)/(\beta - 1)$ for the regime where PCD is superior to PAM, indicating that there are values of $d$ for which PAM is better than PCD for a larger memory regime under $\beta > 1$ as compared to $\beta \in [0, 1)$. This is represented in Fig. 5 by the additional line segment joining points $(1 - \nu(\beta - 1), \nu(\beta - 1))$ and $(1/\beta, 0)$. As $\beta$ approaches one from above, this line segment tends toward the segment joining points $(1, \nu - 1)$ and $(1, 0)$. With it, the regime in which PCD is better than PAM grows until it becomes exactly the regime shown in Fig. 5 for $\beta \in [0, 1)$. In other words, when $\beta > 1$ and as $\beta \rightarrow 1^+$, the regimes in which PCD or PAM are respectively the better choice become the same regimes as in the $\beta \in [0, 1)$ case. This seemingly continuous transition suggests that, when $\beta = 1$, the system should behave similarly to $\beta \in [0, 1)$, i.e., Fig. 3 at least under a poly-$K$ analysis.

**Proof of Theorem 6.** Recall that we are only focusing on a poly-$K$ analysis. We will define $\sigma^\text{PCD}$ and $\sigma^\text{PAM}$ to be the exponents of $K$ in $R^\text{PCD}$ and $R^\text{PAM}$, respectively, i.e., $R^\text{PCD} = \Theta(K^{\sigma^\text{PCD}})$ and similarly for PAM. Our goal is to compare $\sigma^\text{PCD}$ to $\sigma^\text{PAM}$. We can break the proof down into two main cases plus one trivial case. It can help the reader to follow these cases in Fig. 5.

The trivial case is when the total cluster memory $d M$ is large, specifically $\mu + \delta > \min\{\nu, 1/(\beta - 1)\}$. From Theorem 5, the PAM rate is then $o(1)$, hence $\sigma^\text{PAM} = 0$. Therefore, PCD cannot perform better than PAM in this case.

In what follows, we assume $\mu + \delta < \min\{\nu, 1/(\beta - 1)\}$. We can write the exponents of the rates of PCD and PAM as

$$\sigma^\text{PCD} = \min\{1 - (\beta - 1)\mu/\beta, \nu - \mu\};$$

$$\sigma^\text{PAM} = \min\{1/\beta, 1 - (\beta - 1)(\delta - \mu)\}.$$  

Notice that we always have $\sigma^\text{PCD} \leq 1 - (\beta - 1)\mu/\beta \leq 1/\beta$, and hence we only need to compare $\sigma^\text{PCD}$ to the second term in the minimization in $\sigma^\text{PAM}$. We split the analysis into a small and a large memory regimes, with the threshold $\mu \leq \nu$. Here,

**Large memory: $\nu > \beta - 1$: This case is only possible when $\nu < 1/(\beta - 1)$ because we always have $\mu \leq \nu$. Here,
PCD achieves $\sigma_{PCD} = \nu - \mu$. The constraints on $\mu$ imply:

\[
\mu < \nu - \delta \implies 1 - (\beta - 1)(\delta + \mu) > 1 - \nu(\beta - 1);
\]

\[
\mu > \nu \beta - 1 \implies \nu - \mu < \nu(\nu - \beta - 1) = 1 - \nu(\beta - 1).
\]

Therefore, $\sigma_{PCD} < \sigma_{PAM}$.

**Small memory:** $\mu < \nu \beta - 1$: In this case, PCD always achieves $\sigma_{PCD} = [1-(\beta - 1)\mu]/\beta$. Using some basic algebra, we can show that $[1-(\beta - 1)\mu]/\beta < 1 - (\beta - 1)(\delta + \mu)$, i.e., $\sigma_{PCD} < \sigma_{PAM}$, if and only if $\mu < (1 - \beta \delta)/(\beta - 1)$.

### C. Approximate Optimality

The previous sections have focused on a comparison of the PCD and PAM schemes with each other. In this section, we compare the achievable rates of these schemes to information-theoretic lower bounds and identify regimes in which PCD or PAM is approximately optimal. We say that a scheme is approximately optimal if it can achieve an expected rate $R$ such that $R \leq C \cdot R^\ast + o(1)$, where $C$ is some constant.

For $\beta \in [0, 1)$, we show the approximate optimality of PCD in the small memory regime and that of PAM in the large memory regime. When $M > \Omega((N/d) \log N)$, it follows from Theorem 7 that $R_{PAM} = o(1)$, and thus PAM is trivially approximately optimal. The following theorem states the approximate optimality of PCD when $M < O(N/d)$.

**Theorem 7.** When $\beta \in [0, 1)$ and $M < (1 - e^{-1}/2)N/2d$, and for $N \geq 10$, the rate achieved by PCD is within a constant factor of the optimum,

\[
\frac{R_{PCD}}{R^\ast} \leq C \triangleq \frac{96}{(1 - \beta)\rho(1 - e^{-1}/2)^2}.
\]

Note that the constant $C$ is independent of $K, d, N$, and $M$.

Theorem 7 can be proved by first reducing the $\beta \in [0, 1)$ case to a uniform-popularities setup, and then applying the converse results from [11]. Proof details are given in Appendix D.

When $\beta > 1$ we know from Theorem 5 that $R_{PAM} = o(1)$ is achieved for $M > \Omega(\min\{N \log N, K^\gamma\})/d$, and thus PAM is trivially approximately optimal in that regime.

### IV. A HYBRID CODING AND MATCHING SCHEME

For $\beta \in [0, 1)$, we propose a scheme that generalizes ideas from both PCD and PAM. It is a hybrid scheme that we call Hybrid Coding and Matching (HCM). This hybrid scheme is a generalization of the one we proposed in [11] for the uniform-popularities case ($\beta = 0$). Developing a hybrid scheme for the $\beta > 1$ case is part of our on-going work.

The main idea of HCM is to partition files and caches into colors, and then apply a coded caching scheme within each color while performing adaptive matching across colors. More precisely, each color consists of a subset of files as well as a subset of the caches of each cluster. When a user requests a file, the user is matched to an arbitrary cache in its cluster, as long as the cache has the same color as the requested file. For each color, a coded transmission is then performed to serve all the matched users requesting a file from said color. Unmatched users are served directly by the server. This allows us to take advantage of adaptive matching within each cluster as well as obtain coded caching gains across the clusters.

The rate achieved by HCM is given in the following theorem. It is illustrated in Fig. 2 along with the rates of PCD and PAM for comparison.

**Theorem 8.** For any $\beta \in [0, 1)$, HCM can achieve a rate of

\[
R_{HCM} = \min \{ \rho K, \frac{N}{M} - \chi + \frac{K^\gamma}{\sqrt{\pi e}} \}
\]

if $M \leq [N/\chi]$;

\[
R_{HCM} = \Omega(\min\{N \log N, K^\gamma\})/d
\]

if $M \geq [N/\chi],

where $\chi = \lfloor od/(2(1 + t)) \log K \rfloor$, for any $t \in [0, t_0]$.

While the expression for $R_{HCM}$ given in the theorem is rigorous, we can approximate it here for clarity as

\[
R_{HCM} \approx \min \{ \rho K, \frac{N}{M} - \Theta \left(\frac{d}{\log K}\right)^{+} + o(1) \}
\]

The proof of Theorem 8 is given in detail in Appendix E where we provide a rigorous explanation of the HCM scheme.

We will next compare HCM to PCD and PAM. Notice from Fig. 2 that HCM is strictly better than PCD for all memory values. In fact, there is an additive gap between them of about $d/\log K$ for most memory values, and an arbitrarily large multiplicative gap when $M > (N/d) \log K$ where HCM achieves a rate of $o(1)$. Consequently, HCM is approximately optimal in the regime where PCD is, namely when $M < N/2d$.

Furthermore, HCM is significantly better than PAM in the $M < N/d$ regime: there is a multiplicative gap of up to about $K/d$ between their rates in that regime. Moreover, HCM achieves a rate of $o(1)$ when $M > (N/d) \log K$. It is thus trivially approximately optimal in that regime, which includes the regime where PAM is.

### REFERENCES

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

[2] U. Niesen and M. A. Maddah-Ali, “Coded caching with nonuniform demands,” IEEE Transactions on Information Theory, vol. 63, no. 2, pp. 1146–1158, Feb. 2017.

[3] M. Ji, A. M. Tulino, J. Llorca, and G. Caire, “On the average performance of caching and coded multicasting with random demands,” in Proc. IEEE ISWCS, Aug. 2014.

[4] J. Zhang, X. Lin, and X. Wang, “Coded caching under arbitrary popularity distributions,” in Proc. ITA, Feb. 2015.

[5] J. Hachem, N. Karamchandani, and S. N. Diggavi, “Coded caching for multi-level popularity and access,” IEEE Transactions on Information Theory, vol. 63, no. 5, pp. 3108–3141, May 2017.

[6] M. A. Maddah-Ali and U. Niesen, “Coding for caching: fundamental limits and practical challenges,” IEEE Communications Magazine, vol. 54, no. 8, pp. 23–29, August 2016.

[7] G. Paschos, E. Bastug, I. Land, G. Caire, and M. Debah, “Wireless caching: technical misconceptions and business barriers,” IEEE Communications Magazine, vol. 54, no. 8, pp. 16–22, August 2016.

[8] M. Leconte, M. Lelarge, and L. Massoulié, “Bipartite graph structures for efficient balancing of heterogeneous loads,” SIGMETRICS Perform. Eval. Rev., vol. 40, no. 1, pp. 41–52, Jun. 2012.

[9] S. Moharir and N. Karamchandani, “Content replication in large distributed caches,” arXiv:1603.09153 [cs.NI], Mar. 2016.
Lemma 1. When using PCD, the expected number of unmatched users is no greater than $K^{-1} \beta / \sqrt{2\pi}$.

Before we prove Lemma 1, we state the following general result on the expected number of unmatched users.

Lemma 2. If $Y \sim \text{Poisson}(\gamma m)$ users must be matched with $m \geq 1$ caches, where $\gamma \in (0, 1)$, then the expected number of unmatched users $U = [Y - m]^+$ is bounded by

$$
E[U] \leq \frac{1}{\sqrt{2\pi}} \cdot m \cdot (\gamma e^{1-\gamma})^m.
$$

Lemma 2 is proved at the end of the appendix.

Proof of Lemma 2. In PCD, at each cluster $c$ we are attempting to match a number of users $Y(c) \sim \text{Poisson}(\rho d)$ to exactly $d$ caches. Let $U(c)$ denote the number of unmatched users at cluster $c$, and let $U^{*} = \sum_{c=1}^{K/d} U(c)$ be the total number of unmatched users. The matching in PCD is arbitrary, and so any user can be matched to any cache. Consequently, $U(c) = [Y(c) - d]^+$ and we can apply Lemma 2 directly to obtain

$$
E[U^{*}] = \sum_{c=1}^{K/d} E[U(c)] \\
\leq \frac{K}{d} \cdot \frac{1}{\sqrt{2\pi}} \cdot d \cdot \left(\rho e^{1-\rho}\right)^d \\
= \frac{1}{\sqrt{2\pi}} \exp \left\{ \log K + d \log \left(\rho e^{1-\rho}\right) \right\}.
$$

Note that the function $x \mapsto xe^{1-x}$ is strictly increasing for $x \in (0, 1)$. Since $\rho \in (0, 1/2)$, we thus get

$$
\log \left(\rho e^{1-\rho}\right) < \log \left(2\rho e^{1-2\rho}\right) = -\alpha < \log(1) = 0.
$$

Applying this to (2), we obtain

$$
E[U^{*}] \leq \frac{1}{\sqrt{2\pi}} \exp \left\{ \log K + d \log \left(2\rho e^{1-2\rho}\right) \right\} \\
\leq \frac{1}{\sqrt{2\pi}} \exp \left\{ \log K - (1 + t_{0}) \log K \right\} \\
= \frac{1}{\sqrt{2\pi}} K^{-t_{0}},
$$

where (a) uses (1). This concludes the proof.

Before we prove Lemma 2, we need another lemma that pertains to Poisson variables in general.

Lemma 3. Let $Y$ be a Poisson random variable with parameter $\lambda$, and let $m \geq \lambda$. Define $U = [Y - m]^+$, i.e., $U = 0$ if $Y < m$ and $U = Y - m$ if $Y \geq m$. Then,

$$
E[U] \leq m \Pr\{Y = m\}.
$$

Proof: Define $V$ such that $V = 0$ if $Y < m$ and $V = 1$ if $Y \geq m$. Using the tower property of expectation,

$$
E[U] = E[E[U|V]] \\
= \Pr\{V = 0\} E[U|V = 0] + \Pr\{V = 1\} E[U|V = 1] \\
\leq \Pr\{Y = m\} \Pr\{Y = m\} E[Y - m|Y = m] + m \Pr\{Y = m\} \\
= m \Pr\{Y = m\},
$$

where (a) uses the definition of $U$ given the different values of $V$, (b) uses Lemma 1, and (c) uses $\lambda \leq m$.

We can now prove Lemma 2.

Proof of Lemma 2. By using Lemma 3 with $\lambda = \gamma m$, we have

$$
E[U] \leq m \Pr\{Y = m\} = m \cdot \frac{(\gamma m)^m e^{-\gamma m}}{m!}.
$$

Using Stirling’s approximation, we have

$$
m! \geq \sqrt{2\pi m} m^{m-\frac{1}{2}} e^{-m} \geq \sqrt{2\pi m} m^{-m},
$$

which yields

$$
E[U] \leq m \cdot \frac{(\gamma m)^m e^{-\gamma m}}{\sqrt{2\pi m} m^{-m}} = \frac{1}{\sqrt{2\pi}} \cdot m \cdot (\gamma e^{1-\gamma})^m,
$$

thus concluding the proof.

APPENDIX B
DETAILS OF PAM SCHEME FOR $\beta \in [0, 1)$
(PROOF OF THEOREM 2)

First, note that it is always possible to unicast from the server to each user the file that it requested. Since the expected number of users is $\rho K$, we always have $R_{\text{PAM}} \leq \rho K$.

In what follows, we focus on the regime $M = \Omega(N/d)$. Recall that the number of requests for file $n$ at cluster $c$ is $u_{n}(c)$, a Poisson variable with parameter $\rho_{n} d_{n}$.

In the placement phase, we perform a proportional placement. Specifically, since each cache can store $M$ files and each cluster consists of $d$ caches, we replicate each file $W_{n}$ on $d_{n} = p_{n} d M$ caches per cluster.

In the matching phase, we first construct a fractional matching of users to caches, and then show that this implies the

3This proposition appears in the appendix in the extended version of [11].
existence of an integral matching. We construct the fractional matching by dividing each file $W_n$ into $d_n$ equal parts, and then mapping each request for file $W_n$ to $d_n$ requests, one for each of its parts. Each user now connects to the $d_n$ caches containing file $W_n$ and retrieves one part from each cache. This leads to a fractional matching where the total data served by a cache $k$ in cluster $c$ is less than one file if

$$\sum_{W_n \in W_k} \frac{u_n(c)}{d_n} \leq 1,$$

(3)

where $W_k$ is the set of files stored on cache $k$. Let $h(x) = x \log x + 1 - x$ be the Cramér transform of a unit Poisson random variable. Using the Chernoff bound and the arguments used in the proof of [9, Proposition 1], we have that

$$\Pr \left\{ \sum_{W_n \in W_k} \frac{u_n(c)}{d_n} > 1 \right\} \leq e^{-zdM/N},$$

(4)

where $z = (1 - \beta) \rho h((1 + (1 - \rho)/2\rho)) > 0$.

To find a matching between the set of requests and the caches, we serve all requests for files that are stored on caches for which (3) is violated via the server. For the remaining files, there exists a fractional matching between the set of requests and the caches such that each request is allocated only to the corresponding cluster, and the total data served by each cache is not more than one unit. By the total unimodularity of adjacency matrix, the existence of a fractional matching implies the existence of an integral matching [13]. We use the Hungarian algorithm to find a matching between the remaining requests and the caches in the corresponding cluster.

Let $\tau_n$ be the probability that at least one of the caches storing file $W_n$ does not satisfy (3). By the union bound, it follows that

$$\tau_n \leq \frac{KM}{N} e^{-zdM/N}.$$

By definition,

$$\bar{R}_{PAM} \leq \sum_{n=1}^{N} \tau_n \leq KM e^{-zdM/N},$$

(5)

which concludes the proof of the theorem.

While the above was enough to prove the theorem, we will next provide an additional upper bound on the PAM rate, thus obtaining a tighter expression.

Let $G_c$ be the event that the total number of requests at cluster $c$ is less than $d$, and let $G = \bigcap_c G_c$. Using the Chernoff bound, we have

$$\mathbb{E}[G] \geq 1 - \frac{K}{d} e^{-zd},$$

where $z$ is as defined above. Conditioned on $G_c$, the number of files that need to be fetched from the server to serve all requests in the cluster is at most $d$. The rest of this proof is conditioned on $G$.

Let $E_c$ be the event that all caches in cluster $c$ satisfy (3). Using (4) and the union bound, we have that

$$\Pr\{E_c|G\} \geq \Pr\{E_c\} \geq 1 - de^{-zdM/N},$$

where $z$ is as defined above. Conditioned on $E_c$, all the requests in cluster $c$ can be served by the caches. Therefore,

$$\mathbb{E}[R|G] \leq \sum_{c=1}^{K/d} d \cdot \Pr\{E_c|G\} \leq Kde^{-zdM/N},$$

where $\bar{A}$ denotes the complement of $A$ for any event $A$. It follows that

$$\mathbb{E}[R] = \mathbb{E}[R|G] \Pr\{G\} + \mathbb{E}[R|\bar{G}] \Pr\{\bar{G}\} \leq \mathbb{E}[R|G] + N \Pr\{\bar{G}\} \Pr\{G\}$$

$$\leq Kde^{-zdM/N} + \frac{NK}{d} e^{-zd}.$$

(6)

Using (5) and (6), we obtain

$$\bar{R}_{PAM} \leq \min \left\{ KMe^{-zdM/N}, Kde^{-zdM/N} + \frac{NK}{d} e^{-zd} \right\},$$

which is a tighter bound on the PAM rate and implies Theorem 2.

**APPENDIX C**

**DETAILS OF PAM SCHEME FOR $\beta > 1$**

**(PROOF OF THEOREM 5)**

At a high level, the PAM strategy consists in storing complete files in the caches, replicating the files across different caches, and then matching the users to the cache that contains their requested file. Users that cannot be matched to a cache containing their file are served directly from the server.

The above describes PAM strategies very generally; there are many possible schemes for placement and matching within this class of strategies. In this paper, we adopt for $\beta > 1$ a strategy that performs a **knapsack storage (KS)** placement phase that is based on the knapsack problem, and a **match least popular (MLP)** matching phase in which matching is done for the least popular files first. We refer to this PAM scheme as KS+MLP.

**A. Excess Users**

Whatever the strategy, if there are more than $d$ users at a particular cluster, then the excess users must be unmatched. From Lemma 2 we know that the expected total number of these excess users across all caches is $O(K^{-\alpha})$, which is $o(1)$. For the proof of Theorem 5, we will generally use asymptotic notation for the expected rate. Thus by always serving these users directly from the server, their contribution to the rate is always $o(1)$. For this reason, we will make the simplifying assumption in what follows that there are no more than $d$ users at each cluster.

**B. Placement Phase: Knapsack Storage**

We split the KS policy into two parts. In the first part, we determine how many copies of each file will be stored per cluster. In the second part, we determine which caches in each cluster will store each file.
1) **KS Part 1**: The first part of the knapsack storage policy determines how many caches in each cluster store each file by solving a fractional knapsack problem. The parameters of the fractional knapsack problem are a value \(v_n\) and a weight \(w_n\) associated with each file \(W_n\), defined as follows.

- The value \(v_n\) of file \(W_n\) is the probability that \(W_n\) is requested by at least one user in a cluster, \[v_n = 1 - (1 - p_n)^d.\]
- The weight \(w_n\) of file \(W_n\) represents the number of caches in which \(W_n\) will be stored, should the policy decide to store it. If we decide to store a file, we would like to make sure that all requests for that file can be served by the caches, so that it need not be transmitted by the server. To ensure this, we fix \(w_n\) to be large enough so that, with probability going to one as \(K \to \infty\), the number of requests for \(W_n\) is no larger than \(w_n\). We thus choose the following values for \(v_n\):

\[
\begin{aligned}
    w_n &= \begin{cases} 
        d & \text{if } n = 1; \\
        \left[ (1 + \frac{d}{p_n}) \rho dp_n \right] & \text{if } 2 \leq n \leq N_1; \\
        4p_1(\log d)^2 & \text{if } N_1 < n \leq N_2; \\
        1 & \text{if } N_2 < n \leq N,
    \end{cases}
\end{aligned}
\]

where \(N_1\) and \(N_2\) are defined as
\[
\begin{aligned}
    N_1 &= \frac{d^{1/\beta}}{p_1(\log d)^{2/\beta}}, \quad (7a) \\
    N_2 &= \frac{d^{1+1/\beta/2}}{2}, \quad (7b)
\end{aligned}
\]

Using the above parameters \(v_n\) and \(w_n\), we solve the following knapsack problem:

\[
\begin{align*}
    \text{maximize} & \quad \sum_{n=1}^{N} v_n x_n \\
    \text{subject to} & \quad \sum_{n=1}^{N} w_n x_n \leq dM; \\
    & \quad 0 \leq x_n \leq 1, \forall n.
\end{align*}
\]

Then, the number of copies of file \(W_n\) that will be present in each cluster is \(c_n = \lfloor x_n \rfloor w_n\). Note that \(c_n\) is hence either zero or \(w_n\).

2) **KS Part 2**: The second part of the knapsack storage policy is to determine which caches store each file. We will focus on one arbitrary cluster, but the same placement is done in each cluster. To do that, define the multiset \(S\) containing exactly \(c_n\) copies of each file index \(n\). Let us order the elements of \(S\) in increasing order, and call the resulting ordered list \((n_1, \ldots, n_{dM})\). Then, for each \(r\), we store file \(W_n\) in cache \((r - 1) \mod d + 1)\ of the cluster.

### C. Matching and Delivery Phases: Match Least Popular

In the matching phase, we use the Match Least Popular (MLP) policy, the key idea of which is to match users to caches starting with the users requesting the least popular files. Algorithm 1 gives the precise description of MLP.

**Algorithm 1** The Match Least Popular (MLP) matching policy for a fixed cluster.

**Require**: Number of requests \(u_n\) for file \(W_n\), for each \(n\), at the cluster

**Ensure**: Matching of users to caches

1: Set \(K_n \subseteq \{1, \ldots, d\}\) to be the set of caches containing file \(W_n\), for each \(n\).
2: Loop over all files from least to most popular:
3: for \(n \leftarrow N, N - 1, \ldots, 1\) do
4: Loop over all requests for file \(n\):
5: for \(v \leftarrow 1, \ldots, u_n\) do
6: if \(K_n \neq \emptyset\) then
7: Pick \(k \in K_n\) uniformly at random
8: Match a user requesting file \(W_n\) to cache \(k\)
9: Cache \(k\) is no longer available:
10: for all \(n' \in \{1, \ldots, N\}\) do
11: \(K_n' \leftarrow K_n' \setminus \{k\}\)
12: end for
13: end if
14: end for
15: end for

At the end of Algorithm 1, some users will be unmatched, particularly those for which the condition on line 6 fails. Any file requested by an unmatched user will be broadcast directly from the server.

**D. Expected Rate Achieved by KS+MLP**

**Lemma 4.** Let \(X\) be a Poisson random variable with mean \(\mu\), and let \(\epsilon \in (0,1)\) be arbitrary. Then,

\[
\Pr\{X \geq (1 + \epsilon)\mu\} \leq e^{-\mu h(1+\epsilon)},
\]

where \(h(x) = x \log x + 1 - x\).

**Proof:** The lemma follows from the Chernoff bound. ■

**Lemma 5.** Recall that \(u_n(c)\) denotes the number of users requesting file \(W_n\) from cluster \(c\). Consider an arbitrary cluster \(c\). Let \(E_1\) denote the event that

\[
\begin{aligned}
    u_n(c) &\leq (1 + \frac{p_1}{4}) dp_n \quad \text{for all } 1 \leq n \leq N_1; \\
    u_n(c) &\leq 2p_1(\log d)^2 \quad \text{for all } N_1 < n \leq N_2,
\end{aligned}
\]

where \(N_1\) and \(N_2\) are as defined in (7). Then,

\[
\Pr\{E_1\} = 1 - Ne^{-\Omega((\log d)^2)}.
\]

**Proof:** In what follows, we will ignore for simplicity the index \(c\) when it is clear from the context, e.g., we will write \(u_n = u_n(c)\). Recall that \(u_n\) is a Poisson random variable with mean \(\rho dp_n\).

- For \(n \leq N_1\), we have \(dp_n \geq p_1(\log d)^2\). Therefore, using Lemma 4, we have for \(n \leq N_1\)

\[
\Pr\left\{ u_n > \left(1 + \frac{p_1}{4}\right) dp_n \right\} \leq e^{-\Omega(dp_n)} \leq e^{-\Omega((\log d)^2)},
\]

- For \(N_1 < n \leq N_2\), we have \(dp_n \leq p_1(\log d)^2\). Therefore, using Lemma 4, we have

\[
\Pr\left\{ u_n > \left(1 + \frac{p_1}{4}\right) dp_n \right\} \leq e^{-\Omega(dp_n)} \leq e^{-\Omega((\log d)^2)}.
\]
For $N_1 < n \leq N_2$, we have $dp_n < p_1 (\log d)^2$. By defining $\tilde{u}$ to be a Poisson variable with parameter $\rho p_1 (\log d)^2 > \rho dp_n$, we obtain

$$
\Pr \{ u_n > 2p_1 (\log d)^2 \} \leq \Pr \{ \tilde{u} > 2p_1 (\log d)^2 \} \leq e^{-\Omega((\log d)^2)},
$$

In the above, (a) uses the fact that the function $\lambda \mapsto \Pr \{ X(\lambda) > a \}$ defined on $\lambda \in [0, a]$, where $X(\lambda)$ is a Poisson variable with parameter $\lambda$, is an increasing function of $\lambda$, and (b) follows from Lemma [4].

The statement of the lemma is then obtained by a union bound over all the files.

**Lemma 6.** Let $\mathcal{R} = \{n : x_n = 1\}$, where $x_n$ is the solution to the fractional knapsack problem solved in Appendix [C-B]. Let $E_2$ denote the event that, in a given cluster, the MLP policy matches all requests for all files in $R$ to caches. Then,

$$
\Pr \{ E_2 \} = 1 - N e^{-\Omega((\log d)^2)}. 
$$

**Proof:** Since the MLP policy matches requests starting from the least popular files, we first focus on requests for files less popular than file $W_{N_2}$. Because the files follow a Zipf distribution, we can write

$$
p_{N_2} = p_1 N_2^{-\beta} = \frac{p_1}{d^{(\beta+1)/2}}.
$$

Every file less popular than $N_2$ is stored at most once across all the caches in the cluster. Therefore, under MLP, a request for a file $n > N_2$ will remain unmatched only if the cache storing that file is matched to another request for another file $n' > N_2$. Under the KS placement policy, the cumulative popularity of all files no more popular than file $N_2$ stored on a particular cache is less than

$$
p_{N_2} + p_{N_2+d} + p_{N_2+2d} + \cdots = O \left( p_1 d^{-\frac{\beta+1}{\beta+1}} \right).
$$

Each unmatched request for file $n > N_2$ corresponds to the event that there are at least two requests for the $M$ files less popular than $W_{N_2}$ stored on a cache. Therefore, by the Chernoff bound (Lemma [4]), the probability that a particular request for a file $n > N_2$ remains unmatched is at most $e^{-\Omega(d)}$. By the union bound, the probability that at least one request for file $n \in R$ such that $n > N_2$ is not matched by the MLP policy is at most $d e^{-\Omega(d)}$.

Next, we focus on the files $n \in \{2, \ldots, N_1\}$. Note that if the KS policy decides to store file $n$, it stores it on $w_n$ caches. Therefore,

$$
\sum_{n=2}^{N_2} x_n w_n \leq \sum_{n=2}^{N_1} \left[ \left( 1 + \frac{p_1}{2} \right) dp_n \right] + \sum_{n=N_1+1}^{N_2} \left[ 4p_1 (\log d)^2 \right] 
\leq \sum_{n=2}^{N_1} \left[ \left( 1 + \frac{p_1}{2} \right) dp_n + 1 \right] + \sum_{n=N_1+1}^{N_2} \left[ 4p_1 (\log d)^2 + 1 \right]
\leq \left( 1 + \frac{p_1}{2} \right) d(1 - p_1) + 4p_1 (\log d)^2 N_2 + N_2
\leq \left( 1 + \frac{p_1}{2} \right) d(1 - p_1) + \Theta \left( d^{1+1/\beta}/2(\log d)^2 \right)
\leq d,
$$

for $d$ large enough. Therefore, if files are stored according to KS Part 2, each cache stores at most one file from the set $\{2, \ldots, N_2\}$.

Let $K_n$ be the set of caches storing file $n$ in a given cluster. Let $E_{3,k}$ be the event that cache $k \in K_n$ is matched to a user requesting a file $n > N_2$. A cache will be matched to a user requesting a file less popular than $N_2$ only if at least one of the files that it stores, among those that are less popular than $N_2$, is requested at least once. Since there are at most $M$ such files on each cache,

$$
\Pr \{ E_{3,k} \} \leq 1 - \left( 1 - O \left( d^{-(\beta+1)/2} \right) \right)^d.
$$

For a given constant $0 < \epsilon < 1$, there exists a $d(\epsilon)$ such that $\Pr \{ E_{3,k} \} \leq \epsilon$ for all $d \geq d(\epsilon)$.

For each file $n$, $N_1 < n \leq N_2$, let $E_{4,n}$ denote the event that more than $2p_1 (\log d)^2$ of the $\left[ 4p_1 (\log d)^2 \right]$ caches in $K_n$ are matched to users requesting some file $n' > N_2$. By the Chernoff bound for negatively associated random variables [13],

$$
\Pr \{ E_{4,n} \} = e^{-\Omega((\log d)^2)}.
$$

From Lemma [5], we know that with probability at least $1 - Ne^{-\Omega((\log d)^2)}$, there are less than $\left( 1 + p_1/4 \right) dp_n$ requests for each file $2 \leq n \leq N_1$. Therefore, with probability at least $1 - Ne^{-\Omega((\log d)^2)}$, all requests for files in $R$ such that $2 \leq n \leq N_1$ are matched to caches by the MLP policy.

Finally, we now focus on the requests for the most popular file $W_1$. Recall that if the KS policy decides to store this file, it will be stored on all caches in each cluster. Since we are assuming that the total number of requests at each cluster is no greater than $d$, then even if all the users requesting files other than $W_1$ are matched, the remaining caches can still be used to serve all requests for $W_1$.

**E. Expected Rate**

From Lemma [6], we know that, for $d$ large enough, with probability at least $1 - Ne^{-\Omega((\log d)^2)}$, in a given cluster all requests for the files cached by the KS+MLP policy are matched to caches. Let $\bar{N}$ be the number of files not in $R$.
(i.e., that are not cached) that are requested at least once. By the union bound over the $K/d$ clusters,

$$\vec{R}_{PAM} \leq \mathbb{E}[\vec{N}] + \frac{K^2}{d} (1 - \Pr\{E_2\})$$

$$\leq \mathbb{E}[\vec{N}] + \frac{NK^2}{d} e^{-\Omega((\log d)^2)}.$$

After solving the fractional knapsack problem, defined in [8], as a function of $N, K, d, \beta$, and $M$, we can determine the set $R$. For a given $R$, we then have

$$\mathbb{E}[\vec{N}] = \sum_{n \notin R} \left(1 - (1 - p_n)K\right).$$

We hence obtain the following bound on the expected rate:

$$\vec{R}_{PAM} \leq \sum_{n \notin R} \left(1 - (1 - \frac{n^{-\beta}}{A_N})\right) + \frac{NK^2}{d} e^{-\Omega((\log d)^2)}.$$

When $N$ and $d$ are polynomial in $K$, then the second term is $o(1)$, and solving the fractional knapsack problem yields the result of Theorem 5.

**APPENDIX D**

**APPROXIMATE OPTIMALITY (PROOF OF THEOREM 7)**

In this section, we focus on the case $\beta \in [0, 1)$ to prove Theorem 7. The key idea here is to show that this case can be reduced to a uniform-popularities case.

First, notice that the popularity of each file is

$$p_n \geq p_N = \frac{N^{-\beta}}{A_N} \geq \frac{(1 - \beta)N^{-\beta}}{N^{1-\beta}} = 1 - \beta,$$

where $A_N$ is defined in Lemma 7 stated below, and (a) follows from the lemma.

**Lemma 7.** Let $m \geq 1$ be an integer and let $\beta \in [0, 1)$. Define $A_m = \sum_{n=1}^{m} n^{-\beta}$. Then,

$$m^{1-\beta} - 1 \leq (1 - \beta)A_m \leq m^{1-\beta}.$$

Lemma 7 is proved at the bottom of this appendix.

Consider now the following relaxed setup. Suppose that, for every file $n$, there are $\tilde{u}_n(c)$ users requesting file $n$ from cluster $c$, where

$$\tilde{u}_n(c) \sim \text{Poisson} \left(\frac{1 - \beta)pd}{N}\right).$$

Since $(1 - \beta)pd/N \leq p_npd$ for all $n$ by (9), the optimal expected rate for this relaxed setup can only be smaller than the rate from the original setup. Indeed, we can retrieve the original setup by simply creating Poisson $(p_n - (1 - \beta)/N)pd)$ additional requests for file $n$ at each cluster.

Our relaxed setup is now a uniform-popularities setup. It is in fact identical to the one in [11] except that $\rho$ is replaced by $\rho' = (1 - \beta)\rho$, which is still a constant. Consequently, the information-theoretic lower bounds obtained in [11] Lemma 2] and the inequalities that follow can be directly applied here, giving the following lemma.

**Lemma 8.** When $N \geq 10$, the optimal expected rate $\tilde{R}^*$ can be lower-bounded by

$$\tilde{R}^* \geq \frac{(1 - \beta)\rho(1 - e^{-1/2})}{48} \min \left\{ \frac{(1 - e^{-1/2})N}{M} - d, K \right\}.$$

When $M < (1 - e^{-1/2})N/2d$, the bound in Lemma 8 can be further lower-bounded by

$$\tilde{R}^* \geq \frac{(1 - \beta)(1 - e^{-1/2})^2\rho}{96} \min \left\{ \frac{N}{M}, \rho K \right\}.$$

Furthermore, the rate achieved by PCD is upper-bounded by

$$\tilde{R}_{PCD} \leq \min \left\{ \rho K, \frac{N}{M} - 1 + \frac{K^{-\beta}}{\sqrt{2\pi}} \right\} \leq \min \left\{ \rho K, \frac{N}{M} \right\}.$$

Consequently, combining (10) with (11) gives us the result of Theorem 7.

**Proof of Lemma 7.** To prove the lemma, we will relate the sum $A_m = \sum_{n=1}^{m} n^{-\beta}$ with the corresponding integral, which can be evaluated as a closed-form expression.

Let $f$ be any decreasing function defined on the interval $[k, l]$ for some integers $k$ and $l$. Then, we can bound the integral of $f$ by

$$\sum_{n=k+1}^{l} f(n) \leq \int_{k}^{l} f(x) \, dx \leq \sum_{n=k}^{l-1} f(n).$$

Rearranging the inequalities, we get the equivalent statement that

$$f(l) + \int_{k}^{l} f(x) \, dx \leq \sum_{n=k}^{l} f(n) \leq f(k) + \int_{k}^{l} f(x) \, dx. \tag{12}$$

Recall that $A_m = \sum_{n=1}^{m} n^{-\beta}$. Thus we can apply (12) with $f(x) = x^{-\beta}$, $k = 1$, and $l = m$. Since we know that

$$\int_{1}^{m} x^{-\beta} \, dx = \frac{m^{1-\beta} - 1}{1 - \beta},$$

this implies, using (12),

$$m^{1-\beta} - 1 \leq A_m \leq 1 + \frac{m^{1-\beta} - 1}{1 - \beta},$$

$$\frac{m^{1-\beta} - 1}{1 - \beta} \leq A_m \leq \frac{m^{1-\beta} - 1}{1 - \beta},$$

which concludes the proof.

**APPENDIX E**

**DETAILS OF HCM SCHEME (PROOF OF THEOREM 8)**

In this section, we are mostly interested in the case where $K$ is larger than some constant. Specifically, we assume

$$\log K \geq 2g\rho,$$

where $g = (3^{1-\beta} - 1)/4^{1-\beta} > 0$ is a constant. In the opposite case, we can achieve a constant rate simply by unicasting to each user the file that it requested.

Let $t \in [0, t_0]$, and let $\chi = \text{argmin}(\frac{W_n}{2(1 + t)\log K})$. We will partition the set of files into $\chi$ colors. For each color

$$x \in \{1, \ldots, \chi\},$$

define $W_x$ as the set of files colored with $x$. 

We choose to color the files in an alternating fashion. More precisely, we choose for each $x \in \{1, \ldots, \chi\}$
\[ W_x = \{ W_n : n \equiv x \pmod{\chi} \}. \]
Notice that $|W_x| = \lfloor N/\chi \rfloor$ or $\lceil N/\chi \rceil$. We can now define the popularity of a color $x$ as
\[ P_x = \sum_{W_n \in W_x} p_n. \]

The following proposition, proved at the end of the section, gives a useful lower bound for $P_x$.

**Proposition 1.** For each $x \in \{1, \ldots, \chi\}$, we have $P_x \geq g/\chi$.

The significance of the above proposition is that the colors will essentially behave as though they are all equally popular.

Next, we partition the caches of each cluster into the same $\chi$ colors. We choose this coloring in such a way that the number of colors associated with a particular color is proportional to the popularity of that color. Specifically, exactly $\lceil dP_x \rceil$ caches in every cluster will be colored with color $x$. This will leave some caches colorless; they are ignored for the entirety of the scheme for analytical convenience.

We can now describe the placement, matching, and delivery phases of HCM. Consider a particular color $x$. This color consists of $|W_x|$ files and $|dP_x| K/d$ caches in total. The idea is to perform a Maddah-Ali–Niesen scheme [1], [14] on each color separately, while matching each user to a cache of the same color of its requested files. The scheme can be described more formally with the following three steps:

- In the placement phase, for each color $x$ we perform a Maddah-Ali–Niesen placement of the files $W_x$ in the caches colored with $x$.
- In the matching phase, each user is matched to a cache in its cluster of the same color as the file that the user requested. Thus if the user is at cluster $c$ and requests a file from $W_x$, it is matched to an arbitrary cache from cluster $c$ colored with color $x$. For each cluster-color pair, if there are more users than caches, then some users must be unmatched.
- In the delivery phase, for each color $x$ we perform a Maddah-Ali–Niesen delivery for the users requesting files from $W_x$. Next, each unmatched user is served with a dedicated unicast message. The resulting overall message sent from the server is a concatenation of the messages sent for each color as well as all the unicast messages intended for unmatched users.

Suppose that the broadcast message sent for color $x$ has a rate of $R_x$. Suppose also that the number of unmatched users is $U^0$. Then, the total achieved expected rate will be
\[ R_{\text{HCM}} = \min \left\{ \rho K \sum_{x=1}^\chi \mathbb{E}[R_x] + \mathbb{E}[U^0] \right\}, \tag{14} \]

since $\rho K$ can always be achieved by simply unicasting to every user its requested file.

From [14], we know that we can always upper-bound the rate for color $x$ by
\[ R_x \leq \left[ \frac{|W_x|}{M} - 1 \right]^+, \]
for all $M > 0$. Because $|W_x| = \lfloor N/\chi \rfloor$ or $\lceil N/\chi \rceil$ for all $x$, we obtain
\[ \sum_x R_x \leq \begin{cases} \frac{N}{M} - 1 & \text{if } M \leq \lfloor N/\chi \rfloor; \\ 0 & \text{if } M \geq \lfloor N/\chi \rfloor; \\ (N \mod \chi) \left( \frac{\lfloor N/\chi \rfloor}{M} - 1 \right) & \text{otherwise}. \end{cases} \tag{15} \]

All that remains is to find an upper bound for $\mathbb{E}[U^0]$. Let $Y(c, x)$ represent the number of users at cluster $c$ requesting a file from color $x$. Since there are $|dP_x|$ caches at cluster $c$ with color $x$, then exactly $U(c, x) = \lceil Y(c, x) - |dP_x| \rceil^+$ users will be unmatched. Thus we can write $U^0$ as
\[ U^0 = \sum_{c=1}^{K/d} \sum_{x=1}^\chi U(c, x). \]

Notice that $Y(c, x) \sim \text{Poisson}(\rho dP_x)$, and that the $Y(c, x)$ users must be matched to $|dP_x|$. For convenience, we define $\tilde{Y}(c, x) \sim \text{Poisson}(2\rho \cdot |dP_x|)$ and $\tilde{U} = Y(c, x) - |dP_x|^+$. Since we have
\[ \rho dP_x \leq 2\rho \cdot |dP_x|, \]
i.e., the Poisson parameter of $\tilde{Y}(c, x)$ is at least the Poisson parameter of $Y(c, x)$, then
\[ \mathbb{E}[U(c, x)] = \sum_{y=|dP_x|}^\infty y \cdot \text{Pr}[Y(c, x) = y] \]
\[ \leq \sum_{y=|dP_x|}^\infty y \cdot \text{Pr}[\tilde{Y}(c, x) = y] \]
\[ = \mathbb{E}[\tilde{U}(c, x)], \]
where $(a)$ uses the fact that the function $\lambda \mapsto \text{Pr}[\text{Poisson}(\lambda) = m]$ is increasing in $\lambda$ as long as $\lambda < m$; see [11] Proposition 2.

This allows us to apply Lemma 5 on $\tilde{Y}(c, x)$ and $\tilde{U}(c, x)$ in order to upper-bound the expectation of $U(c, x)$ by
\[ \mathbb{E}[U(c, x)] \leq \mathbb{E}[\tilde{U}(c, x)] \leq \frac{1}{\sqrt{2\pi}} \cdot |dP_x| \cdot (2pe^{1-2p})^{\lfloor dP_x \rfloor}. \]

This proposition appears in the appendix in the extended version of [11].
Consequently, we get the upper bound on \( \mathbb{E}[U^0] \),

\[
\mathbb{E}[U^0] = \sum_{c=1}^{K/d} \sum_{x=1}^{\chi} \mathbb{E}[U(c,x)] \\
\leq \sum_{c=1}^{K/d} \sum_{x=1}^{\chi} \frac{1}{\sqrt{2\pi}} |dP_x| (2\rho e^{1-2\rho}) |dP_x| \\
= \frac{1}{\sqrt{2\pi}} \sum_{x=1}^{\chi} K \cdot |dP_x| (2\rho e^{1-2\rho}) |dP_x| \\
\leq \frac{1}{\sqrt{2\pi}} \sum_{x=1}^{\chi} |P_x \cdot K (2\rho e^{1-2\rho}) |dP_x| .
\]

Isolating part of the term in the sum,

\[
K (2\rho e^{1-2\rho}) |dP_x| = \exp \{ \log K + \log (2\rho e^{1-2\rho}) |dP_x| \} \\
= \exp \{ \log K - \alpha |dP_x| \} \\
\leq \exp \left\{ \log K - \frac{\alpha g}{2} \right\} \\
\leq \exp \left\{ \log K - \frac{\alpha g}{2\chi} \right\} \\
\leq \exp \left\{ \log K - \frac{\alpha g}{2} \cdot \frac{2(1+t)}{\alpha g} \log K \right\} \\
= \exp \{ \log K - (1+t) \log K \} \\
= K^{-t},
\]

where (a) uses Proposition 1 and (b) uses the definition of \( \chi \) combined with \(|y| \leq y\). We obtain the final upper bound on the expected number of unmatched users,

\[
\mathbb{E}[U^0] \leq \frac{1}{\sqrt{2\pi}} \sum_{x=1}^{\chi} |P_x| K^{-t} = \frac{K^{-t}}{\sqrt{2\pi}}. \tag{16}
\]

Starting with the definition of \( P_x \), we have

\[
P_x = \sum_{W_x \in \mathcal{W}_x} p_n \\
= \frac{1}{A_N} \sum_{k=0}^{|\mathcal{W}_x|-1} (k\chi + x)^{-\beta} \\
\geq \frac{1}{A_N} \sum_{k=0}^{|\mathcal{W}_x|-1} (k + \chi)^{-\beta} \\
= \chi^{-\beta} \frac{A_{|\mathcal{W}_x|}}{A_N} \\
\geq \chi^{-\beta} \frac{(N/\chi - 1)^{1-\beta} - 1}{N^{1-\beta}} \\
= \frac{1}{\chi} \left[ \left( 1 - \frac{\chi}{N} \right)^{1-\beta} - \left( \frac{\chi}{N} \right)^{1-\beta} \right] \\
\geq \frac{1}{\chi} \left[ \left( 1 - \frac{g\alpha}{2\log K} \right)^{1-\beta} - \left( \frac{g\alpha}{2\log K} \right)^{1-\beta} \right] \\
\geq \frac{1}{\chi} \left[ \left( 1 - \frac{1}{4} \right)^{1-\beta} - \left( \frac{1}{4} \right)^{1-\beta} \right] \\
= \frac{g}{\chi},
\]

where (a) uses Lemma 1, (b) uses the fact that \(|\mathcal{W}_x| \geq \lceil N/\chi \rceil \geq N/\chi - 1 \) for all \( \chi \), (c) uses the definition of \( \chi \) as well as \( N \geq d \) and \( t \geq 0 \), and (d) uses 13.

Finally, we combine 15 and 16 in 14 to obtain the rate expression in Theorem 8 thus completing its proof.

**Proof of Proposition 1.** Choose any \( x \in \{1, \ldots, \chi\} \).