Title
Coded Content Caching for Wireless Networks

Author
Hachem, Jad

Publication Date
2017

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

Coded Content Caching for Wireless Networks

A dissertation submitted in partial satisfaction of the
requirements for the degree of Doctor of Philosophy
in Electrical Engineering

by

Jad Hachem

2017

ABSTRACT OF THE DISSERTATION

Coded Content Caching

for Wireless Networks

by

Jad Hachem

Doctor of Philosophy in Electrical Engineering

University of California, Los Angeles, 2017

Professor Suhas N. Diggavi, Chair

The increase in mobile Internet usage has created a need to optimize content delivery in cellular networks. In emerging technologies such as 5G, heterogeneous networks are proposed, which consist of sparsely deployed cellular base stations (BS) with wide coverage but low data rate, combined with a dense network of wireless access points (AP) of small coverage but relatively high data rate. We envisage equipping the APs with a local cache. When a group of users request some files, their demands are served by a (common) broadcast from one or more BSs, which is aided by side information placed *a priori* in the APs. Therefore, there is a tradeoff between the size of the cache and the size of the broadcast. Our goal is to design schemes that optimize this tradeoff.

Traditional caching techniques, which have proved efficient in the wired Internet, are insufficient to handle the explosion in multimedia demand in wireless networks. The seminal work by Maddah-Ali and Niesen ["Fundamental limits of caching," *IEEE Transactions on Information Theory,* May 2014] introduced an information-theoretic framework to study this problem and proposed the so-called "coded caching" technique that takes advantage of the broadcast nature of wireless to send coded multicast messages to many users at once, thus greatly improving the transmission rate. This

original work assumed a caching system with an error-free broadcast link between the content library and the users, focused on equally popular files, and assumed that each user has one exclusive local cache. Inspired by this work, this thesis studies more general coded caching problems within this information-theoretic framework. Broadly speaking, we focus on non-uniform content popularity as well as more general network topologies.

First, we consider a system where the files desired by the users are not all equally popular. We adopt a "multi-level" popularity model where files are partitioned into multiple popularity classes. Under this model, we study the behavior of the system as the total number of users, as compared to the number of caches, varies. Furthermore, we allow a more complex topology by requiring some users to connect to multiple caches at once. We find approximately optimal strategies for the two extreme cases: when the number of users per cache is very large, and when each cache has exactly one user. An interesting dichotomy is observed where the approximately optimal strategies required for these two extremes are very different. Finally, we provide a heuristic for "discretizing" common popularity distributions such as Zipf into multiple levels, and numerically evaluate its performance.

Second, we study the caching problem when we are allowed to assign users to caches after their demands are known, under some restrictions. Specifically, we divide the caches into several clusters, and we assume that each user can be assigned to one cache from a specific cluster and that each cache can serve no more than one user. Focusing on a stochastic Zipf popularity model, we find that there are regimes in which coded caching is no longer efficient. Instead, a strategy that consists in replicating files across clusters and performs an uncoded delivery dominates certain regimes. We compare these two schemes and find the regimes in which each is more efficient, as a function of cache memory, cluster size, and skewness of popularity. Finally, we show that each scheme is approximately optimal in some of these regimes.

Third, we return to the uniform popularity model in order to study more complicated networks than the error-free broadcast network. Our main focus is on Gaussian interference networks, where caches are placed at both the transmitters (BSs) and the receivers (APs). We propose a separation-based approach that creates separate network and physical layers, with a *multiple-multicast* message set to act as an interface between them. At the physical layer, we focus on transmitting this message set across the interference channel; at the network layer, we solve the caching problem using the

message set as a set of error-free links replacing the channel. We show that this architecture is approximately optimal under high SNR. Among the implications of this result is that placing common information between the transmitters cannot give more than a constant-factor benefit. Moreover, we show that, when the receiver memory is large, a small number of transmitters is enough to obtain most of the benefits.

The dissertation of Jad Hachem is approved.

Lieven Vandenberghe

Christina Panagio Fragouli

Alexander Sherstov

Suhas N. Diggavi, Committee Chair

University of California, Los Angeles

2017

To Lilia, Taima, and Cecilia.

Contents

List of Figures

List of Tables

Acknowledgments

My graduate education has been the central part of my life for the last six years. This journey has shaped me in many ways, from the academic and intellectual growth that comes from learning to perform original research, to the personal and emotional maturity that comes with living ten time zones away from my home in Beirut.

These six years were made possible by my advisor, Professor Suhas Diggavi, for whom I have great admiration and respect. He has given me the most valuable mentorship and guidance on the road to my PhD and the generous financial support without which I could not have pursued my degree. His kindness and understanding have made my graduate studies thoroughly enjoyable and rewarding. I cannot express my gratitude to him enough.

My graduate studies would not have been the same if, during my undergraduate years at AUB, I had not had the fortune of the tutelage of Professor Louay Bazzi, Professor Ibrahim Abou-Faycal, and Professor Kamal Khuri-Makdisi. Much of my love for mathematics, information theory, and computer science is due to the passion that these professors have when teaching, and my decisions were highly influenced by their advice on both my undergraduate and my graduate studies.

I am also fortunate to have had incredibly talented collaborators, each of which has impacted me in some way. Professor Christina Fragouli shaped my Master's and contributed valuable input to my research throughout my graduate years. I-Hsiang Wang mentored me during my summer at EPFL, and I engaged in my first true research experience under his guidance. At Qualcomm, I had the pleasure of working with Urs Niesen, and I learned more about extracting the fundamental insights of a research problem from him than anyone else. Last but not least, Nikhil Karamchandani, who was our post-doc during my first three years, was a great mentor and friend, and I owe my PhD thesis to his guidance and patience. I am grateful to everyone for letting me think about research

on my own, make mistakes, and learn from them, and for giving valuable pointers when I needed them.

My experience at Qualcomm in 2015 was exciting, fruitful, and instructive, thanks to my internship mentor Jubin Jose who entrusted me with challenging problems and provided me with valuable guidance.

A special thanks goes to Professor Lieven Vandenberghe and Professor Alexander Sherstov, who have taught me courses that have deeply inspired me and who have given their time to be on my PhD committee.

I also want to thank my labmates, Shaunak, Can, Mehrdad, Joyson, Yair, Wei, Ayan, Manikandan, and Cuneyd. My friendship with Shaunak and Can in particular has been extremely rewarding. I would always look forward to our discussions and coffee breaks during the day.

These years would also not have been the same without the friendship of Marios, Kasra, Mario, Oussama, Samih, and Abed. I am especially grateful to my girlfriend Tala for her unending encouragement and for her support during stressful times.

None of this would have been possible without the love and support of my parents and my brothers Tarek and Abbas, who have fostered in me a curiosity about the world and a strong desire to discover and learn. All that I have achieved is a result of their life-long mentoring, their belief in me, and the influence they have all had on my personality, my education, and my interests for twenty-eight years.

Finally, I would like to thank the University of California, Los Angeles and the National Science Foundation[1] for their financial support during my graduate education.

[1]This thesis was supported in part by NSF grants #1423271, #1314937, and #1514531.

VITA

March 2013	Master of Science in Electrical Engineering University of California, Los Angeles Los Angeles, California
June 2011	Bachelor of Engineering in Computer and Communications Engineering American University of Beirut Beirut, Lebanon
Summer 2015	Interim Engineering Intern Qualcomm Flarion Technologies Bridgewater, New Jersey
Summer 2012	Visiting Researcher Ecole Polytechnique Fédérale de Lausanne Lausanne, Switzerland
Summer 2010	Research Intern German Aerospace Center (DLR) Weßling, Germany

Publications

J. Hachem, U. Niesen, S. Diggavi, "Degrees of freedom of cache-aided wireless interference networks", *arXiv:1606.03175 [cs.IT]*, Jun. 2016. Submitted to the *IEEE Transactions on Information Theory*.

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

J. Hachem, N. Karamchandani, S. Moharir, S. Diggavi, "Caching with partial matching under Zipf demands", to appear in *2017 IEEE Information Theory Workshop (ITW)*, Nov. 2017.

J. Hachem, N. Karamchandani, S. Moharir, S. Diggavi, "Coded caching with partial adaptive matching", in *2017 IEEE International Symposium on Information Theory (ISIT)*, Jun. 2017.

J. Hachem, U. Niesen, S. Diggavi, "A layered caching architecture for the interference channel", in *2016 IEEE International Symposium on Information Theory (ISIT)*, Jul. 2016.

J. Hachem, N. Karamchandani, S. Diggavi, "Effect of number of users in multi-level coded caching", in *2015 IEEE International Symposium on Information Theory (ISIT)*, Jun. 2015.

J. Hachem, N. Karamchandani, S. Diggavi, "Content caching and delivery over heterogeneous wireless networks", in *2015 IEEE International Conference on Computer Communications (INFOCOM)*, Apr. 2015.

J. Hachem, N. Karamchandani, S. Diggavi, "Multi-level coded caching", in *2014 IEEE International Symposium on Information Theory (ISIT)*, Jun. 2014.

J. Hachem, I.-H. Wang, C. Fragouli, S. Diggavi, "Coding with encoding uncertainty", in *2013 IEEE International Symposium on Information Theory (ISIT)*, Jul. 2013.

R. Raulefs, S. Zhang, C. Mensing, C. Ghali, J. Hachem, "Constrained indoor distributed cooperative positioning", in *2011 8th International Workshop on Multi-Carrier Systems and Solutions (MC-SS)*, May 2011.

R. Raulefs, C. Mensing, S. Zhang, C. Ghali, J. Hachem, "Dynamic cooperative positioning", in *17th European Wireless 2011 – Sustainable Wireless Technologies*, Apr. 2011.

Chapter 1

Introduction

1.1 Caching in Wireless Networks

Broadband data consumption has witnessed a tremendous growth over the past few years, due in large part to multimedia applications such as Video-on-Demand. The traditional solution to manage this increased demand in the wired Internet is via Content Distribution Networks (CDNs), in which data is mirrored in various locations across the network. This in effect pushes the content closer to the end users, thereby removing the bottleneck at the content distribution server by utilizing repeated demand of particular content.

Wireless data consumption, driven by the increased demand for high-definition content on mobile devices, has also grown at a significant rate [1] and is testing the limits of our underlying wireless communication systems [2]. However, simply borrowing the CDN solution from wired networks and applying it to wireless systems is insufficient to solve the wireless content delivery problem. The reason is that the CDN solution has the most gains when the local communication link is not the bottleneck [3]. In wireless cellular usage, this is typically not true as the (cellular) wireless hop is a bottleneck link. As we argue in this thesis, by leveraging the properties of wireless channels, embedding caching in the emerging wireless network, and jointly designing storage and transmission, one could create a foundational methodology that could enable a solution to the wireless content delivery problem.

The *broadcast* nature of wireless can be used as an advantage to alleviate this problem. This,

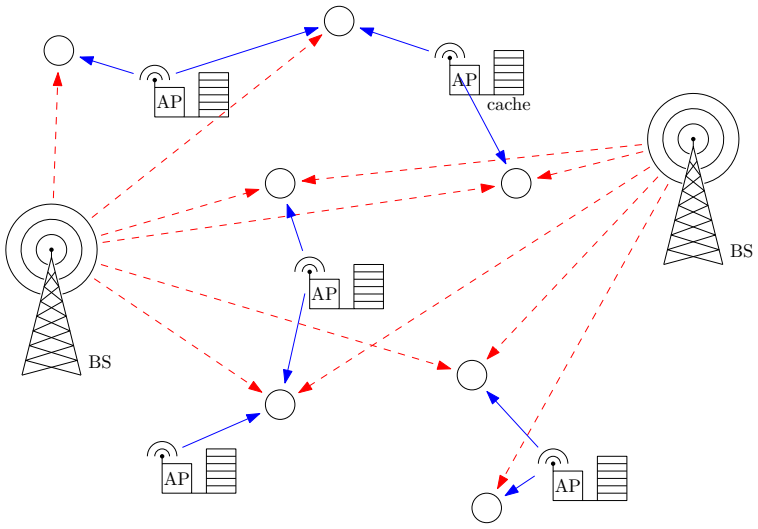


Figure 1.1: A wireless heterogeneous network (HetNet). Each user (circle) can connect to one or more APs in its vicinity and access its local cache, while listening to broadcast messages from one of more BSs.

along with the emerging *heterogeneous* wireless network can be used to provide an architecture for wireless content distribution. The heterogeneous wireless network (HetNet) architecture emerging for 5G consists of a dense deployment of wireless access points (APs) with small coverage and relatively large data rates, in combination with sparse cellular base-stations (BSs) with large coverage and smaller data rates. For example, the access points could be WiFi or emerging small-cells (or femto-cells), which provide high data rate for short ranges. The consequence of this emerging architecture is that a user could potentially receive broadcast from one or more BSs as well as connect to one or more wireless APs. Therefore, we could place caches at local APs and complement them with macro-cellular (BS) broadcast, as illustrated in Figure 1.1.

A first study of such a caching system was done by Maddah-Ali and Niesen in [4, 5]. They derived the first fundamental limits of caching in wireless networks by considering a problem where content can be stored in multiple caches without *a priori* knowledge of any user requests. Once the users' file requests are revealed, the cache contents are complemented by a broadcast message transmitted by one BS to all users. The users can then combine this broadcast message with the contents of their cache in order to recover the file that they requested. In [4], the authors show that a joint design of storage and delivery, also known as "coded caching", can significantly improve content delivery rate requirements and yield large gains over traditional caching methods. This was

achieved by a content placement that creates network-coded multicast opportunities among users with access to different storage units, even when these users have different (and *a priori* unknown) requests. This approach enables an examination of the optimal trade-off between cache memory size and broadcast delivery rate. In fact, by using the coded caching technique and by deriving information-theoretic impossibility results, the authors characterize the optimal trade-off to within a constant multiplicative factor.

The problem studied in [4, 5] focused on an error-free broadcast setting in which each user has their own local cache, and in which all files are equally popular. While these works were a significant first step, some natural questions arise about more general settings. In this thesis, we focus on three main questions. First, we ask how the scheme changes when the content follows some non-uniform popularity distribution. For example, it is often the case that some files are requested much more often than the rest (e.g., viral videos). Second, we ask how the density of APs and of users in the heterogeneous network, as depicted in Figure 1.1, can influence the strategy. In particular, we consider two aspects: the load on each AP (how many users are connected to one AP) and the density of the APs (how many APs are in the vicinity of any user). Third, we look at physical-layer aspects of the problem by considering multiple BSs and the resulting interference between them. We ask whether breaking the separation between the network layer and the physical layer is necessary, or whether a separation architecture is (approximately) optimal. Each chapter in this thesis studies a novel problem that explores one or more of these questions.

1.2 Thesis Outline

As mentioned above, this thesis explores some fundamental questions that arise from the setting initially studied in [4]. The setting is generalized to non-uniform popularity, multiple access, and interference. We address these in three chapters.

In Chapter 2, we study the caching problem when the content popularity obeys a *multi-level* model. In this model, the files are grouped into a small number of popularity classes (called *levels*) such that the files in the same class are equally popular. We focus on a broadcast setup and allow many users to connect to the same cache, and we study how the number of users per cache changes

the behavior of the system.

One of the main contributions of the chapter is that the approximately optimal strategy when there are many users per cache is very different to the one when there is only one user per cache. When there are many users per cache, the approximately optimal strategy is to completely separate the levels by partitioning the cache memory and allocating each part to one level. Perhaps surprisingly, we find that it is often beneficial to allocate some memory to the less popular levels even when the more popular levels are not completely stored. On the other hand, when there is only one user per cache, then it is more efficient to merge some popularity levels and treat them as though their files are of the same popularity.

In addition, Chapter 2 combines the multi-level popularity model with the ability of users to connect to multiple caches. In particular, all users requesting files from the same popularity level are required to connect to the same number of caches, but this number can be different across levels. We refer to this as *multi-level access*. The chapter explores the effect of this multi-level access, which enables achieving a lower network load for smaller memories.

In Chapter 3, we consider again the possibility of users connecting to multiple caches. The difference is that, while a user has the ability to connect to many caches, the central server assigns one cache to the user *after* the demands of the users are revealed. In other words, once the users reveal their demands, we are allowed to choose a matching of users to caches, under some restrictions, and then transmit a broadcast message to all users. We call this an *adaptive matching* setup. We also adopt a non-uniform popularity model in this chapter, however it is a stochastic model in which the number of requests for each file follows a Poisson distribution independently of the other files.

One of the main contributions of the chapter is that a coded caching approach is no longer always approximately optimal. Indeed, instead of creating differences in the caches by placing different parts of files (as is done with coded caching), it is often more efficient to fully replicate the popular files across multiple caches. When a user requests a file, it is then either matched to a cache containing that file or served directly with a unicast message. We find that the regime in which this approach is more efficient than the coded caching approach is larger not only when users have potential access to a larger number of caches, but also when the popularity of files is more

heavily skewed, i.e., a smaller number of files are requested by a larger number of users.

In Chapter 4, we study the joint design of caching and delivery in the presence of interference. In particular, we consider multiple transmitters, each equipped with a cache, in addition to the receivers (users). The receivers are also each equipped with a local cache. The transmitters and receivers are separated by a Gaussian interference network where, at the physical layer, each receiver observes a noisy linear combination of all the transmitter signals. In order to isolate the interference aspect of the problem, we adopt the uniform popularity model in this chapter, i.e., all files are equally popular.

Our main contribution in Chapter 4 is an approximate characterization of the communication degrees of freedom (DoF) of the network, i.e., the log-scaling of the communication rate with the signal-to-noise ratio. Our characterization reveals three key insights. First, the approximate DoF is achieved using a strategy that separates the physical and network layers. This separation architecture is thus approximately optimal. Second, we show that increasing transmitter cache memory beyond what is needed to exactly store the entire library between all transmitters does not provide more than a constant-factor benefit to the DoF. A consequence is that transmitter co-operation (e.g., transmit zero-forcing) is not needed for approximate optimality. Third, we derive an interesting trade-off between the receiver memory and the number of transmitters needed for approximately maximal performance. In particular, if each receiver can store a constant fraction of the content library, then only a constant number of transmitters are needed. Solving the caching problem requires formulating and solving a new communication problem, the symmetric multiple multicast X-channel, for which we provide an exact DoF characterization.

We conclude the thesis in Chapter 5 with a discussion of open problems.

Finally, we note that much of the work in this thesis has previously been published. Most of Chapter 2 was published in [6], most of Chapter 3 is to appear in [7], and most of Chapter 4 is found in [8, 9].

Chapter 2

Multi-Level Popularity and Access

2.1 Introduction

In this chapter, we expand on the coded caching problem initiated in [4, 5] mainly by considering non-uniform popularities. The setup studied in [4, 5] consisted of single-level content, i.e., every file in the system is uniformly demanded. However, it is well understood that content demand is non-uniform in practice, with some files being more popular than others. Motivated by this, [10, 11, 12, 13, 14, 15] considered such non-uniform content demand, following different models. In [10, 11, 12], the setup considered a single user per cache requesting a file independently and randomly according to some (arbitrary) probability distribution that represents content popularity. These works studied the trade-off between the average rate and the cache memory. A memory-sharing scheme was proposed in [10], and its achievable rate was characterized. However, from our understanding, this scheme was not shown to be order-optimal in general.[1] In [11], a different scheme was proposed, based on a clustering of the most popular files into a single content level, which was shown to be order-optimal for Zipf-distributed content and, more recently, for arbitrary distributions in [12].

By contrast, in [13], a deterministic multi-level popularity model was introduced (simultaneous to the aforementioned other non-uniform popularity models), where content is divided into discrete

[1] We refer to an "order-optimal" result as one that is within a constant multiplicative factor from the information-theoretic optimum. The constant is to be independent of the number of users, caches, memory size and number of popularity levels.

levels based on popularity. In this chapter, we focus on this model and study it mostly in the context where a large number of users connect to each cache ("multi-user setup"), and, for each level, a fixed and *a priori* known fraction of the users per cache request files from said level. It is easy to see that, when the number of users per cache is large enough, this deterministic model will closely approximate an equivalent stochastic-demands model similar to [10, 11, 12]. We also study the scenario where users could connect to multiple access points (caches) as well as listen to the broadcast to get the desired content. In short, the setup considered has a different popularity model, user population, and cache access to those considered in earlier literature. We also compare the results, for this popularity model, between setups with many users per cache (multi-user setup) and a single user per cache (single-user setup).

The main contribution of this chapter is, for any given multi-level content popularity profile, to approximately solve the trade-off of the transmission cost at the BS with the storage cost at the APs. In addition, we also approximately solve the case where users have access to multiple APs. Finally, we study the effect of number of users per cache in the multi-level content popularity model. In particular, the following are the core technical contributions of the chapter:

- We propose new strategies for the multi-level popularity and access model, both for small (single-user setup) and large (multi-user setup) number of users.

- We derive information-theoretic outer bounds in order to evaluate the performance of the proposed schemes. Notably, we derive novel non-cut-set-based outer bounds for the multi-user setup.

- We demonstrate the order-optimality of the strategies (both when the number of users is small and when it is large) with respect to the information-theoretic outer bounds. This order-optimality is independent of the number of popularity levels, users, files, and caches.

- Finally, we demonstrate that the order-optimal strategy for the problem can be very different depending on the total number of users in the system, as compared to the number of caches. In particular, when there are many users per cache (i.e., the multi-user setup), the order-optimal strategy requires a complete separation of the levels along with a careful allocation of

the cache memory between them. A striking aspect of this solution is that, in some regimes, it is better to store some less popular content without completely storing the more popular content, even when cache memory is available. In contrast, in the single-user setup, we show that merging the most popular levels and giving them all the memory, leaving none for the rest, is order-optimal; this is similar to a strategy proposed in [11, 12].

The chapter is organized as follows. Section 2.2 formulates the problem, describing precisely the multi-user and single-user setups. We establish some background in Section 2.3, which enables us to state the main results in Section 2.4. The caching and delivery strategy for the multi-user setup, as well as corresponding lower bounds, are given in Section 2.5, while the single-user setup is studied in Section 2.6. A brief discussion about the dichotomy between the two setups is given in Section 2.7. Section 2.8 presents a discussion and some numerical evaluations to interpret the results. Finally, Section 2.9 explores some practical considerations and presents results from a working system that we have implemented. Many of the detailed proofs are given in Appendix A.

Related Work

Content caching has a rich history and has been studied extensively, see for example [16] and references therein. More recently, it has been studied in the context of Video-on-Demand systems where efficient content placement and delivery schemes have been proposed in [3, 17, 18, 19]. The impact of content popularity distributions on caching schemes has also been widely investigated, see for example [20, 21, 22].

Most of the literature has focused on wired networks and, as argued before, the solutions there do not carry directly to wireless networks, which are the focus of this chapter. Recently, [23] proposed a caching architecture for heterogeneous wireless networks, with the small-cell or WiFi access points acting as helpers by storing part of the content. A content placement scheme is formulated and posed as a linear program. However, the (information-theoretic) optimality of such schemes was not examined in that work; in our work we develop new schemes, as well as results showing their approximate optimality. Another aspect (also common to most of the papers in the content caching literature) is that the delivery phase used independent unicasts to serve the different users. The important observation to utilize broadcast to improve system performance

by serving multiple users simultaneously was made in [5, 4]. They initiated the study of coded caching where joint design of storage and delivery was considered for the case with a single level of files and single cache access during delivery by proposing an order-optimal coded caching scheme. These results have been extended to online caching systems in [24], heterogeneous cache sizes [25], unequal file sizes [26], and improved converse arguments [27, 28]. Efficient coded caching schemes have been devised in [29], and the effect of finite file sizes has been investigated in [30]. Content caching and delivery has also been studied for hierarchical tree topologies [31], device to device networks [32, 33], multi-server topologies [34], and heterogeneous wireless networks [14]. However, most of these extensions have been for uniform popularity models.

Coded caching was extended to non-uniform popularity models in [10, 11, 12, 35], where the setup considered a single user per cache requesting a file independently and randomly according to some (arbitrary) probability distribution that represents content popularity. The trade-off between memory and average delivery rate was studied in these works. The work in this chapter differs from these as it uses a deterministic multi-level popularity model introduced in [13], enabling a worst-case rather than average case analysis. We analytically characterize the order-optimal splitting parameters for the memory-sharing scheme, even with user access to multiple caches. The dichotomy of order-optimal schemes between having multiple users per cache and a single user per cache is also demonstrated for this multi-level popularity model.

Other related work includes [36] which derives scaling laws for content replication in multihop wireless networks; [37] which explores distributed caching in mobile networks using device-to-device communications; [38] which studies the benefit of coded caching when the caches are distributed randomly; and [39] which explores the benefits of adaptive content placement, using knowledge of user requests.

2.2 Setup, Notation, and Formulation

2.2.1 Overview

Consider a content library (such as Netflix) containing files of size F bits and of varying popularities. Over the course of, say, a day, users will request many files according to their popularity

distribution: more popular files will be requested more often. In anticipation of these requests, we place information about the files in the caches, of capacity MF bits, of APs that are close to the users, during periods of low network traffic. We refer to this as the *placement phase*. Later, when a large number of users request files during periods of high network traffic, they each connect to one or more APs and access the contents of their caches. Since cache capacity is typically limited, the caches cannot always fully serve all the requests, and a common broadcast message of size RF bits is then sent to all users. The users can combine this broadcast message with the contents of their caches to recover the files that they have requested. This phase is called the *delivery phase*. It is important to stress that, while the file popularities are known during the placement phase, the exact set of files that the users will request is not known until the delivery phase.

Clearly, there is a trade-off between M (the *cache memory*) and R (the *broadcast rate*): the larger the value of M, the more information the caches can hold about the files, and thus the smaller the value of R needed to serve all requests. In this chapter, we seek to characterize this trade-off when the files in the content library follow a *multi-level popularity* model. Furthermore, we capture the considerable effect that the total number of users has on the system by considering two extremes, which we call the *multi-user* and the *single-user* setups. We also introduce a *multi-level access* model, where users are required to connect to a certain number of APs based on the popularity of the file they have requested. In the next few subsections, we will provide an informal description and motivation for the above three aspects. The formal definitions and formulations are presented in Section 2.2.5.

2.2.2 Multi-level popularity

In multi-media applications such as video-on-demand, we often find that a small number of files are requested by many more users than the rest of the files. This difference in popularity can easily influence the caching system described above. For example, when deciding what to store in the (limited-capacity) caches, one would want to give more of the cache memory to the more popular files, since they will be, on average, requested more often.

Different popularity models have been considered in the literature. The simplest model was studied in [4, 5]. In this model, all files are equally popular, meaning that there is no preference

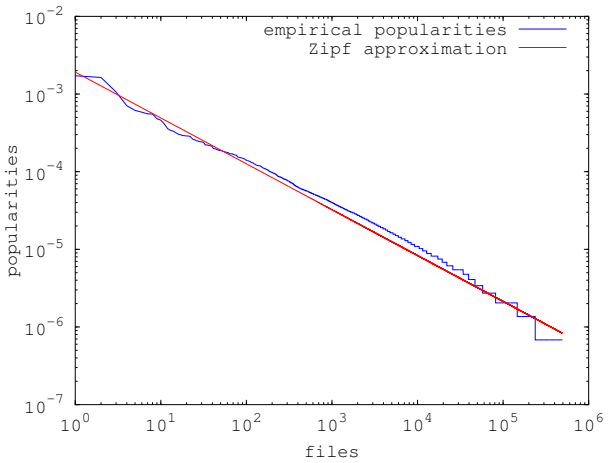


Figure 2.1: Empirical popularities of some YouTube videos (based on number of views), with an approximating Zipf distribution.

among the users to choose one file over the others. The results were of a worst-case nature, i.e., they are true for all possible (valid) combinations of user demands. While this model allowed the first approximate-optimality caching result and introduced the idea of coded caching, it is not an accurate representation of typical multi-media data. There has since been a lot of work in the literature on coded caching with non-uniform file popularities. Many use a probabilistic approach, in which user demands are stochastic and follow some probability distribution, and in which the focus is on average results rather than worst-case ones. There has been particular focus on Zipf distributions [10, 11], which arise in examples such as YouTube videos (see Figure 2.1 based on data from [40]), but also on arbitrary distributions [10, 12].

Typical popularity models have a "continuous" nature, such as with Zipf-distributed content popularity. However, accurately estimating such popularities requires a large sample size of user requests for content. This might be reasonable for the most popular content, but not for the less popular content, especially when the content library can consist of tens of millions of files. It is made even more difficult by the frequency of small changes in popularity over short periods of time. Therefore, it is natural to estimate a "histogram" of this popularity distribution, by estimating the total popularity of sets of files instead of individual files. This motivates the *multi-level popularity model*, which we adopt in this chapter and which we have previously introduced in [13, 14, 15]. In this model, files are grouped into a certain (small) number L of *popularity levels*, such that all files in the same level are equally popular. An added advantage of this model is that it allows a

worst-case analysis of the problem. Furthermore, the schemes we develop for the multi-level model can be adapted to any continuous popularity model by judiciously discretizing the popularities and grouping files into levels of our choice. This observation is supported by (a) a numerical analysis presented in Section 2.8, and (b) related work in the literature that analytically show that such a discretization is useful in certain setups [12, 11], discussed in Section 2.6.3.

As mentioned above, our model assumes that files belong to L popularity levels, such that all files in a single level are equally popular. The popularity of the files is reflected in the total number of users in the system requesting files from each level. In the multi-level popularity model, this number is assumed to be *fixed* and *known* to the designer *a priori*. For example, the designer might know during the placement phase that, say, exactly 25 users will request a file from a specific popularity level. However, the designer does not know *which* files from that level will be requested.

To motivate the determinism in this last point, consider the following example. Suppose there are two popularity levels, and assume a stochastic-demands setup where each user is three times as likely to request a file from the first level as he is from the second. If there are 40 users in the network, then we would expect that about 30 of them will request a file from the first level, and 10 from the second. By the law of large numbers, when a large number of users is present in the system, we expect a concentration of the number of users requesting files from each level around their means. Because of this concentration, the stochastic-demands model will closely resemble the determinism in the multi-level model that we adopt.

2.2.3 Number of users

In the example at the end of the previous section, we motivated our *a priori* knowledge of the total number of users in the system requesting files level i, for each level i. In this section, we are interested in a similar knowledge of the user requests *at every cache*. In particular, we want to compare situations where we know the number of users *per cache* requesting a file from level i, for all i, to situations where we lack this knowledge. We will use the phrase *user profile* to refer to this knowledge. Specifically, the user profile is the number of users requesting a file from level i at cache k, for every pair (i, k). Knowledge of the user profile is greatly dependent on the total number of users as compared to the number of caches.

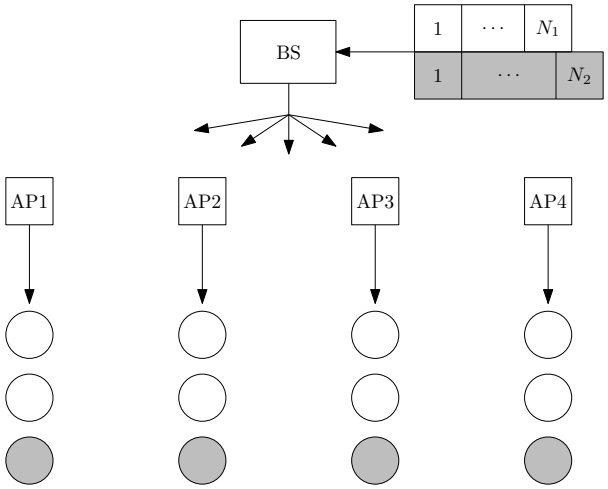


Figure 2.2: Multi-user setup with $K = 4$ caches, and $L = 2$ levels with $(U_1, U_2) = (2, 1)$ users per cache. Both levels have an access degree of 1.

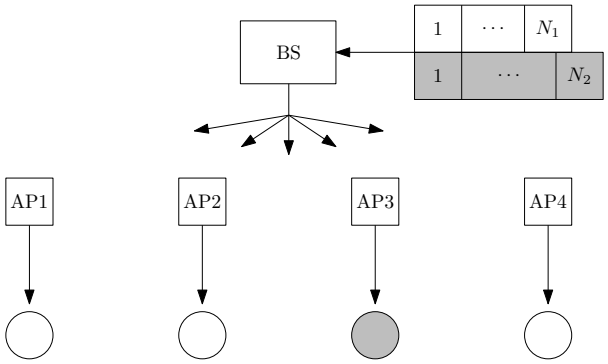


Figure 2.3: Single-user setup with $K = 4$ caches, and $L = 2$ levels with $(K_1, K_2) = (3, 1)$ users.

To illustrate, let us expand on the example from the previous section, where there are two popularity levels such that 75% of all users, regardless of their total number, will request a file from the first level and 25% from the second. Assume that there are four caches in the system, and that the users are evenly distributed among them, regardless of their number. If the total number of users is very large, say 400, then we know that 300 of them will request files from the first level and 100 from the second. More importantly, because these numbers are large compared to the number of caches (four), then we can expect that every cache will have around 75 users for the first level and 25 for the second (see Figure 2.2 for an illustration, albeit on a smaller scale). The designer can hence *a priori* estimate the user profile with a reasonable degree of confidence. On the other hand, consider what happens when the number of users is very small, say 4. Since we assume that the users are evenly distributed among the caches, then we will have exactly one user at every cache. One of these users will request a file from the second popularity level, while the other three will go for the first level, as shown in Figure 2.3. It is thus impossible to determine *a priori* which cache the level-2 user will connect to, and hence the user profile is unknown to the designer.

Because it is difficult to analyze the problem for a general number of users per cache, we restrict ourselves to studying only the two extremes in order to bring out the key aspects affected by the number of users. The first extreme, which we call the *multi-user setup*, is the one where the number of users per cache is so large that the user profile is known and every level is represented equally across caches, i.e., every cache has the same number U_i of users requesting files from level i. The second extreme, which we call the *single-user setup*, is the one where there is exactly one user per cache. In this setup, the user profile is unknown and the levels are not represented equally across caches.

As we will show in this chapter, the equal representation of the levels across caches or lack thereof gives rise to very different strategies that are suitable for each setup. Specifically, the multi-user setup requires a strategy that completely *separates* the levels from each other, while the single-user setup necessitates a strategy that *merges* certain levels into one super-level. Moreover, the strategy suitable for one setup is inefficient for the other. We discuss this striking dichotomy in detail in Section 2.7.2.

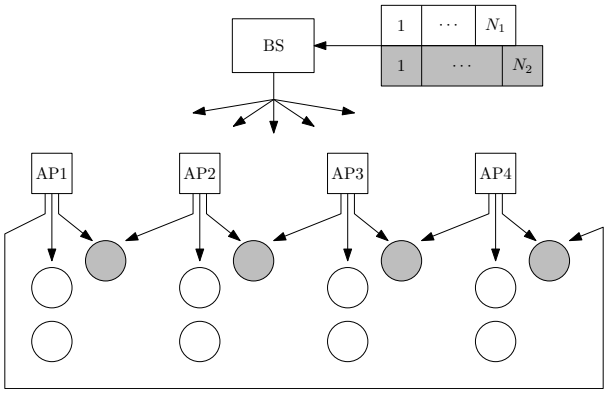


Figure 2.4: Multi-user setup with $K = 4$ caches, and $L = 2$ levels with $(U_1, U_2) = (2, 1)$ users per cache, and access degrees of $(d_1, d_2) = (1, 2)$.

2.2.4 Multi-level access

In this wireless setting, and with the high density of access points, it is inevitable that users will have many APs in their vicinity. This in turn means that every user can potentially connect to several APs, which could effectively increase the amount of cache memory available at the user for the same value of M, thus lowering the broadcast rate R needed.

In this chapter, we incorporate this multiple-access aspect into the caching problem by allowing certain users to connect to a certain number of caches. In order to keep this incorporation simple, we assume that all users requesting files from the same level i are required to access the same number of caches d_i. We call d_i the *access degree of level* i, and we refer to this multiple-access model as the *multi-level access* model. Furthermore, we assume also for simplicity that the caches are arranged linearly and that users connect to d_i consecutive caches, with a cyclic wrap-around for symmetry. To be more precise, if we number the caches from 1 to K, then any arbitrary user requesting a file from level i will connect to caches $\{k, k+1, \ldots, k+d_i-1\}$ for some integer k (if $k+d_i-1 > K$, then the user will connect to caches $\{k, k+1, \ldots, K, 1, 2, \ldots, k+d_i-1-K\}$). The multi-level access is illustrated in Figure 2.4, with two levels of access degrees $d_1 = 1$ and $d_2 = 2$. While this one-dimensional arrangement assumption is simplistic, it does provide an interesting first analysis of the combination of coded caching with multiple access. Furthermore, the setup and the ideas behind our strategies can be extended to a (more realistic) scenario where caches are arranged in a two-dimensional lattice.

Note that while the values of the access degrees could be chosen by the designer, we here assume that they are given and fixed. In other words, our goal is to design efficient schemes given any choice of d_i's. This choice could be limited by the network topology such as the density of APs, as well as by practical considerations such as the overhead associated with connecting to a larger number of APs.

Finally, in this chapter, we restrict the multi-level access model to the multi-user setup only. All users in the single-user setup connect to exactly one cache each.

2.2.5 Problem formulation

While the previous subsections motivated the problem setup, this subsection describes the problem formally.

A content library contains files that obey a multi-level popularity model. Specifically, there are L levels of files, numbered 1 through L. Every level $i \in \{1, \ldots, L\}$ consists of N_i files of size F bits each. There are K caches in the system, each of capacity MF bits, or, equivalently, of memory M.

Multi-user setup (Figure 2.4)

In the multi-user setup, for every level i, there are exactly KU_i users, each connecting to d_i caches and requesting a file from level i. More specifically, for each $k \in \{1, \ldots, K\}$, there are exactly U_i users connected to caches $\{k, \langle k+1 \rangle, \ldots, \langle k+d_i-1 \rangle\}$ and requesting files from level i, where

$$\langle m \rangle = \begin{cases} m & \text{if } m \leq K; \\ m-K & \text{if } m > K. \end{cases} \tag{2.1}$$

Single-user setup (Figure 2.3)

In the single-user setup, there are exactly K_i users requesting a file from level i, for every i. More specifically, the K caches can be partitioned into L subsets of sizes $K_1, \ldots, K_L$, such that each cache from subset i has exactly one user connected to it, and that user requests a file from level i. While the $\{K_i\}_i$ values are known to the designer, the partition itself (which is in fact the user profile) is unknown during the placement phase. Each user connects to only a single cache (no

multi-level access).

In both setups, content related to the files is placed in the caches during the placement phase, before the specific user requests are revealed. After each user requests one arbitrary file (in accordance with the multi-level popularity model), the delivery phase occurs in which a broadcast message of size RF bits (i.e., of rate R) is sent to all users, so that every user can recover the requested file by combining the broadcast message with the contents of the cache(s) that they can access.

A pair (R, M) is said to be *achievable* if there exists a placement-and-delivery strategy that uses caches of memory M and transmits, for *any* possible combination of user requests (valid within the multi-level popularity model), a broadcast message of rate at most R that satisfies all said requests with vanishing error probability as the file size F grows. More formally, for any finite file size F, let $P_e(F, S)$ be the worst-case probability that some user is unable to recover their requested file using strategy S, where "worst-case" is over all possible valid user requests. Then, (R, M) is achievable if there exists a sequence of strategies $S(F)$ for increasing F such that $P_e(F, S(F)) \to 0$ as $F \to \infty$.

Our goal is to find all such achievable pairs. In particular, we wish to find the optimal rate-memory trade-off,

$$R^*(M) = \inf \{R : (R, M) \text{ is achievable}\},$$

where the minimization is done over *all* possible strategies.

The problem of finding an exact characterization of the rate-memory trade-off is difficult even for the simplest cases [4]. Therefore, in this chapter, we will instead consider approximate characterizations. In particular, we wish to find a strategy that achieves a rate-memory trade-off $R(M)$ such that:

$$cR(M) \leq R^*(M) \leq R(M),$$

where c is some constant. We say that such a strategy is *order-optimal*.

We allow c to depend on only one parameter: the maximum access degree $D = \max_i d_i$. Accessing multiple APs can be costly, both because of the overhead required for the user to establish a connection with multiple APs, as well as the reduced rate of communication with the farther APs that the user must now access. Thus in practice we do not expect that one user will be required to

access a large number of caches, and so D would be quite small. However, c must be independent of all other parameters.

2.2.6 Regularity conditions

We assume the following two regularity conditions. First, for every popularity level i, there are more files than users. In the multi-user setup, this means:

$$\forall i, \quad N_i \geq KU_i. \tag{2.2}$$

In the single-user setup, we would write:

$$\forall i, \quad N_i \geq K_i. \tag{2.3}$$

Consider for example a video application such as Netflix, where "files" would be video segments of a few seconds to a few minutes. Borrowing an example from [31], if a content library has 1000 popular movies (from the same popularity level) of length 100 minutes, and each movie is divided into files (segments) of one minute each, the result is 100,000 files in one popularity level. The above regularity condition states that no more than 100,000 users will be watching one of those 1000 movies (i.e., one of the 100,000 segments, or files) at any given moment.

Second, *in the multi-user setup only*, we assume that no two levels have very similar popularities. The popularity of a level can be written as the number of users per file of the level, i.e., as U_i/N_i. Hence, if i is a more popular level than j, the regularity condition states:[2]

$$\sqrt{\frac{U_i/N_i}{U_j/N_j}} \geq \frac{D}{\beta} = 198D, \tag{2.4}$$

where $\beta = 1/198$ is called the *level-separation factor*. The reasoning behind this condition is that, if it did not hold for some levels i and j, then we can think of them as essentially one level with $N_i + N_j$ files and $U_i + U_j$ users per cache. The resulting popularity $(U_i + U_j)/(N_i + N_j)$ would be

[2] As we will see in later sections, many calculations will involve the square roots of U_i and N_i. For this reason, phrasing the regularity condition using the square roots is more useful.

Table 2.1: Notation

All setups	
K	# of caches
L	# of popularity levels
N_i	# of files in level i
F	file size
R	broadcast rate
M	cache memory
$R^*(M)$	optimal rate-memory trade-off
Multi-user setup	
U_i	# of users per cache for level i
d_i	access degree of level i
D	maximum access degree
$\beta = 1/198$	level-separation factor
Single-user setup	
K_i	total # of users for level i

close to both U_i/N_i and U_j/N_j.

2.2.7 Notation table

For reference, we present in Table 2.1 all the notation that we use in this chapter.

2.3 Preliminaries

Traditional caching only uses multiple-unicast transmissions from the server to the users. As a result, the total transmission size was proportional to the number of users, for any value of the cache memory. Coded caching, initially introduced in [4], brought a drastic improvement by eliminating the dependence of the transmission size on the number of users (except for very small cache memory). This technique was shown to be approximately optimal in [4] (a centralized version) and in [5] (a decentralized version), under a setup with a single level of popularity and a single user at every cache, with a single-access structure. We will refer to this setup as the Basic Setup, because it will form the basis of our main analysis.

The scheme for the Basic Setup, in its decentralized form (on which will we henceforth focus)

consists in placing a random sampling of bits from all files in every cache, independently. Consequently, there will be some overlap, but also some differences, in the bits present in every cache. The BS then transmits linear combinations of these bits, taking advantage of the overlaps as well as the differences, so that the same linear combination can be useful for multiple users at once. The resulting rate-memory trade-off is given in the following Lemma.

Lemma 1 (Rate for the Basic Setup [5, Theorem 1][3])**.** *For a single-level caching system with K caches, $N \geq K$ files, a single user per cache with an access degree of* 1*, and a cache memory of $M \in [0, N]$, the following rate is achievable:*

$$R^0(M, K, N) = \min\left\{\frac{N}{M}, K\right\} \cdot \left(1 - \frac{M}{N}\right),$$

and $R^0(M, K, N) = 0$ if $M > N$. Furthermore, this rate is within a constant of the optimum.

Notice that, when $M > N/K$, then the rate becomes $(N/M - 1)$, removing all dependence on the number of users. Under traditional caching, the rate would have been $K(1 - M/N)$.

2.3.1 Generalizing to multi-user, multi-access

As we will see in Section 2.5, our strategy for the multi-user setup of the multi-level problem is to isolate the levels, giving each a portion of the memory, and then applying independent caching-and-delivery strategies for each level. We thus divide the multi-level problem into multiple single-level problems. These single-level problems are still more general than the Basic Setup, because they allow multiple users per cache as well as an access degree that is larger than one. Therefore, our first step is to generalize the strategy of the Basic Setup to the case where there are $U \geq 1$ users per cache and an access degree of $d \geq 1$. Specifically, caches $\{k, \langle k+1\rangle, \ldots, \langle k+d-1\rangle\}$ are connected to exactly U users, for any $k \in \{1, \ldots, K\}$, where $\langle m\rangle$ is defined in (2.1). Figure 2.5 illustrates such a setup, and Theorem 1 gives the rate achieved by the generalized strategy. Note that the Basic Setup is obtained when $U = d = 1$.

[3]The expression of $R_D(M)$ in [5, Theorem 1] can be upper-bounded by the expression we give here in Lemma 1 by simply noting that $N \geq K$ and $(1 - M/N)^K \geq \max\{0, 1 - KM/N\}$.

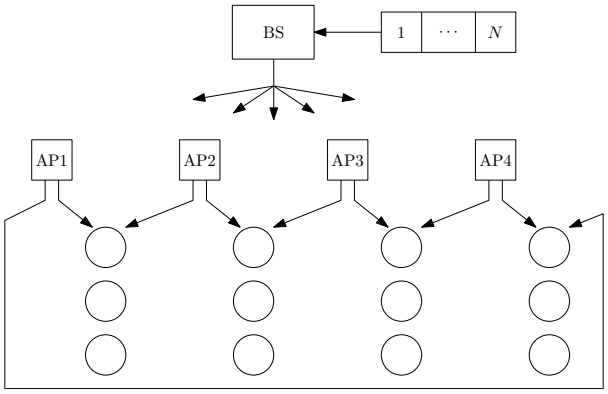


Figure 2.5: Generalized single-level setup with $U = 3$ users per cache, and an access degree $d = 2$.

Theorem 1 (Single-level rate-memory trade-off)**.** *Given a single-level caching system, with N files, K caches, U users at each cache with access degree d, such that $N \geq KU$ and d divides K, and a cache memory of $M \in [0, N/d]$, the following rate is achievable:*

$$R^{SL}(M, K, N, U, d) = U \cdot \min\left\{\frac{N}{M}, K\right\} \cdot \left(1 - \frac{dM}{N}\right). \tag{2.5}$$

Furthermore, $R^{SL}(M, K, N, U, d) = 0$ if $M > N/d$. When d does not divide K, we can achieve four times the expression in (2.5)*.*

Theorem 1 is proved later in the section.

The rate achieved in Theorem 1 is actually order-optimal, in the sense defined in Section 2.2.5. More specifically, $R^{\mathrm{SL}}(\cdot)$ is at most a factor of cd times the optimal rate, where d is the access degree and c is some constant independent of all parameters. We do not need to directly prove the order-optimality of $R^{\mathrm{SL}}(\cdot)$, because the single-level setup is a special case of the multi-level multi-user setup, and hence the order-optimality follows from Theorem 4, presented in Section 2.4.

Notice from (2.5) that $R^{\mathrm{SL}}(M, K, N, U, d) = U \cdot R^{\mathrm{SL}}(M, K, N, 1, d)$. This is because we can divide users into groups of U, such that all U users in the same group are connected to the exact same set of d caches, and hence share the same side information. Since the single-level strategy relies heavily on sending multiple coded multicast messages, and since coding opportunities arise only when users have different side information, there are no coding opportunities to be gained when considering a multicast to users connected to the same set of caches. As a result, these

U users have to be handled with U separate messages, and thus the achievable rate is directly proportional to U.

The behavior of R^{SL} with d is more interesting. While, in the single-access case, a cache memory $M < N$ implied a non-zero transmission rate, adding the multi-access aspect results in a zero achievable rate for the smaller memory value of N/d. Intuitively, at $M = N/d$, we effectively apply an erasure-correcting code on all the files and spread it across the caches, such that any d caches can reproduce all files. Both these insights are brought out in the proof of Theorem 1 below.

Proof of Theorem 1. For convenience, we will focus on the case when d divides K. This allows us to achieve exactly the rate expression in (2.5). As mentioned in the theorem statement, when d does not divide K, a rate that is four times the one in (2.5) can be achieved. This is briefly discussed after this proof.

The key difference between the general single-level problem and the Basic Setup is that there are now many users who have caches in common. This additional overlap in side information is problematic because coded caching works best when the side information available to different users can be designed independently. As an extreme example, if two users connect to the exact same set of caches (which happens when $U > 1$), then the same side information is available to both of them, and no network coding can be done to benefit them both at the same time.

The strategy we propose gets around this issue by partitioning the users into groups, such that no two users in the same group share any cache, with the goal of multicasting coded messages to users in the same group. To maximize the coding gains, these groups have to be as large as possible, in this case dU groups of size K/d users each. This can be done as follows. Let $u_{i,k}$ denote the i-th user connected to caches $\{k, \ldots, \langle k+d-1\rangle\}$, where $i \in \{1, \ldots, U\}$ and $k \in \{1, \ldots, K\}$. We let the groups be

$$\mathcal{U}_{i,j} = \{u_{i,k} : k \equiv j \bmod d\},$$

for all $i \in \{1, \ldots, U\}$ and $j \in \{1, \ldots, d\}$. To show that users in the same group do not share caches, consider two distinct users $u_{i,j+dt}$ and $u_{i,j+dt'}$ in the same group $\mathcal{U}_{i,j}$. Since they both have the same index i, we must have $j + dt \neq j + dt'$, i.e., $t \neq t'$. Then, $u_{i,j+dt}$ is connected to caches $\{j+dt, \ldots, \langle j + d(t+1) - 1\rangle\}$ and $u_{i,j+dt'}$ is connected to caches $\{j+dt', \ldots, \langle j + d(t'+1) - 1\rangle\}$.

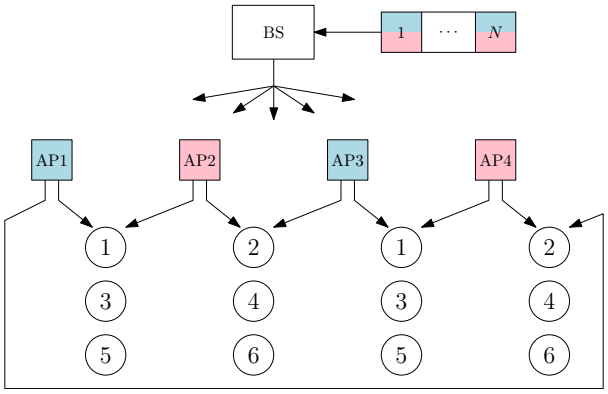


Figure 2.6: An illustration of the scheme used on the example from Figure 2.5. Caches are colored into $d = 2$ colors, and the files are divided and colored with the same colors. In parallel, the users are divided into $dU = 6$ groups, where users from the same group have no overlapping caches.

These two sets of caches are disjoint because $t \neq t'$ and d divides K.

Next, we choose d colors and we color cache k with color $k \bmod d$. The result is that every user will be connected to exactly one cache of every color. In parallel, we split every file into d equal subfiles, and color each subfile using the same d colors used on the caches. The end result is illustrated in Figure 2.6.

Consider now one group-color pair. It represents a subsystem of N subfiles of size F/d bits each, K/d caches, and exactly one user connected to each cache and not to any other cache of the same color. This observation allows us to reduce the problem into d^2U subproblems similar to the Basic Setup, as described below.

In the placement phase, consider each of the d colors separately, and perform a random placement (the same as in the Basic Setup) of the subfiles of that color in the caches of the same color. Since each subfile is of size F/d bits, then every cache can hold dM subfiles.

In the delivery phase, each of the d^2U group-color pairs is considered separately. For every group and every color, we have a subsystem where K/d users are each requesting a subfile of size F/d bits from one like-colored cache of size dM subfiles. Because the same placement of the Basic Setup was done in the placement phase, we can send the same broadcast message as in the Basic Setup to these K/d users. The total broadcast size is thus:

$$RF = d^2U \cdot R^0(dM, K/d, N) \cdot (F/d) \text{ bits.}$$

Note that the memory given to the R^0 function is dM, since the "unit" for each subsystem is a subfile (of size F/d bits), and each cache can hold dM subfiles.

The expression of $R^{\text{SL}}(\cdot)$ can be obtained by combining the above equation with Lemma 1. □

While the above proof focused on d being a divisor of K, we note that the expression for $R^{\text{SL}}(\cdot)$ in Theorem 1 can still be achieved up to a constant factor when d does not divide K. Specifically,

$$R^{\text{SL}}(M, K, N, U, d) \leq 4U \min\left\{\frac{N}{M}, K\right\}\left(1 - \frac{dM}{N}\right) \tag{2.6}$$

is true for all values of $d \leq K$. The corresponding strategy uses a (K, d) erasure-correcting code for $M = N/d$, while for $M \leq N/2d$ it ignores multi-access and instead uses a placement and delivery scheme that assumes $d = 1$. Finally, we time-share between the two schemes for $N/2d < M < N/d$.

The results presented in this section will be key to our solution to the multi-level caching problem. Indeed, in all that will follow, we use the above-described coded caching scheme as a black box that gives a rate $R^{\text{SL}}(M, K, N, U, d)$ for input parameters M, K, N, U, and d.

2.3.2 A small multi-level example with exact characterization

In order to illustrate the general multi-level problem, we will here present a small example that combines both multi-level popularity and multi-level access. We give, for this example, an exact characterization of the rate-memory trade-off.

Consider the setup in Figure 2.7. The server holds files from $L = 2$ popularity levels. The first level has $N_1 = 2$ files called $\{W_1^1, W_2^1\}$, and the second level has $N_2 \geq 4$ files called $\{W_1^2, \ldots, W_{N_2}^2\}$. There are two APs, each equipped with a cache of memory (i.e., size normalized by file size) M. There is one user accessing each cache and requesting a file from level 1 (users 1 and 2, requesting files $W_{r_1}^1$ and $W_{r_2}^1$ respectively), and a third user accessing both caches and requesting a file from level 2 (user 3, requesting file $W_{r_3}^2$).

Theorem 2 (Exact characterization for the small example)**.** *For the setup shown in Figure 2.7, the optimal rate-memory trade-off is plotted in Figure 2.8 and is characterized by:*

$$R^*(M) = \max\left\{3 - 2M, \frac{5}{2} - M, 2 - \frac{1}{2}M, 1 - \frac{M-2}{N_2/2}, 0\right\}.$$

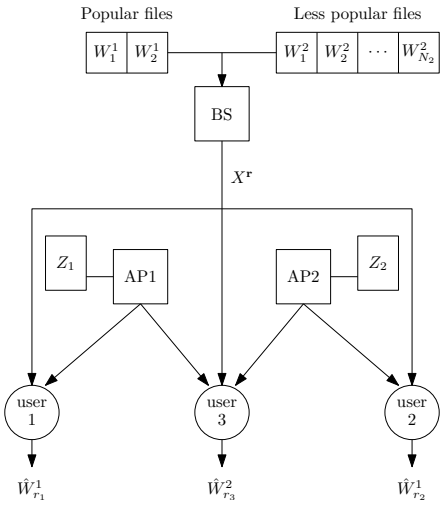


Figure 2.7: Small example that illustrates multi-level popularity and access.

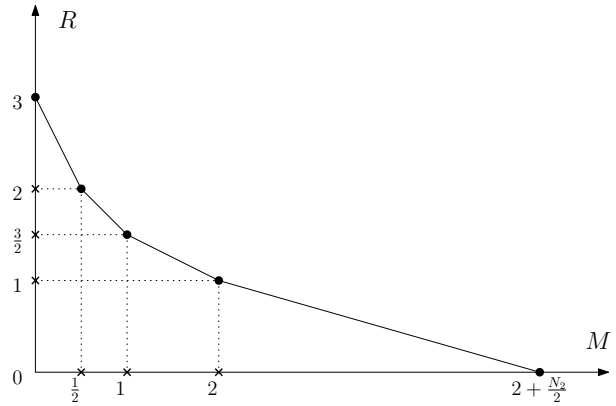


Figure 2.8: Optimal rate-memory trade-off $R^*(M)$ for the small example.

The proof of Theorem 2 is given in Appendix A.3. However, for illustration and to gain intuition about the general problem, we will here briefly discuss the achievability scheme for two values of the cache memory M: $M = 1$ (for which $R = 3/2$ is achievable) and $M = 1/2$ (for which $R = 2$ is achievable).

Suppose that $M = 1$, so that each cache can hold the equivalent of one file. We first split each file in level 1 into two equal parts: $W_n^1 = (W_{n,a}^1, W_{n,b}^1)$, for $n = 1, 2$. Now, each cache exclusively stores one half of each popular file, and stores nothing from level 2. Thus the first cache will contain $(W_{1,a}^1, W_{2,a}^1)$ and the second cache will contain $(W_{1,b}^1, W_{2,b}^1)$. When the users make their requests, the BS transmits $W_{r_3}^2$ completely for user 3, and sends a coded transmission $(W_{r_1,b}^1 \oplus W_{r_2,a}^1)$ for users 1 and 2 together. Combining the transmission with the contents of their respective caches,

each of users 1 and 2 can recover the file that they have requested. In total, the BS would have transmitted one complete file ($W^2_{r_3}$), plus the equivalent of one half-file (the linear combination), for a total rate of $R = 3/2$.

Suppose now that $M = 1/2$, i.e., each cache can only hold the equivalent of half a file. We again split the two level-1 files just like in the previous case. However, this time the first cache stores ($W^1_{1,a} \oplus W^1_{2,a}$) while the second cache stores ($W^1_{1,b} \oplus W^1_{2,b}$). When the users make their requests, the BS again transmits $W^2_{r_3}$ to serve user 3, but also sends $W^1_{r_1,b}$ and $W^1_{r_2,a}$ for users 1 and 2. This allows them to recover the file that they have requested by combining the transmission with the side-information available at their caches. Since the BS has transmitted one complete file and two half-files, the total rate is $R = 2$.

While, in this small example, an exact characterization of the rate-memory trade-off was found, this is difficult in general. For this reason, we focus our attention on order-optimality results as stated in Section 2.2.5.

2.4 Main Results

In this section, we provide the approximately optimal rate-memory trade-off for each of the two setups (multi-user and single-user). We discuss how each such trade-off is achieved.

2.4.1 Multi-user setup

The placement-and-delivery strategy that we adopt for the multi-user setup is a *memory-sharing* strategy. It consists of dividing the cache memory between all the L levels, and then treating each level as a separate caching sub-system, with the reduced memory. In other words, we give level i a memory $\alpha_i M$, where $\alpha_i \in [0, 1]$ and $\sum_i \alpha_i = 1$, and we then apply a single-level placement-and-delivery strategy for this level on this $\alpha_i M$ memory, separately from the other levels. The total rate for this scheme is thus

$$R^{\text{MU}}\left(M, K, \{N_i, U_i, d_i\}_i\right) = \sum_{i=1}^{L} R^{\text{SL}}(\alpha_i M, K, N_i, U_i, d_i), \tag{2.7}$$

where $R^{\mathrm{SL}}(\cdot)$ is defined in (2.5).

By optimizing the overall rate over the memory-sharing parameters $\{\alpha_i\}_i$ (subject to $\alpha_i \in [0,1]$ for all i and $\sum_i \alpha_i = 1$), we establish a memory allocation which we will show is order-optimal. At a high level, this allocation is done by partitioning the popularity levels into three sets that we call H, I, and J. The levels in H have such a small popularity that they will get no cache memory. On the opposite end of the spectrum, the most popular levels are assigned to J and are given enough cache memory to completely store all their files in every cache. Finally, the rest of the levels, in the set I, will share the remaining memory among themselves, obtaining some non-zero amount of memory but not enough to store all of their files.

Our choice of the (H, I, J) partition and corresponding memory assignments are discussed in Section 2.5.1. This choice results in the following achievable rate.

Theorem 3. *Given a multi-user caching setup, with K caches, L levels, and, for each level i, N_i files and U_i users per cache with access degree d_i, and a cache memory of M, the following rate[4] is achievable:*

$$R^{MU}(M) \approx \sum_{h \in H} KU_h + \frac{\left(\sum_{i \in I} \sqrt{N_i U_i}\right)^2}{M - \sum_{j \in J} N_j/d_j} - \sum_{i \in I} d_i U_i,$$

where (H, I, J) is a particular type of partition of the set of levels called an M-feasible partition.[5] This partition is dependent on the value of M.

Intuitively, since a level $h \in H$ receives no cache memory, all requests from its KU_h users must be handled directly from the broadcast. Since, by regularity condition (2.2), we have $N_i \geq KU_i$ for all levels i, then in the worst case a total of KU_h distinct files must be completely transmitted for the users requesting files from level h. The users in set J require no transmission as the files are completely stored in all the caches; however, it does affect the rate by reducing the memory available for levels in I. This is apparent in the expression $M - \sum_{j \in J} N_j/d_j$. Finally, the levels in I, having received some memory, require a rate that is inversely proportional to the effective memory and that depends on the level-specific parameters N_i, U_i, and d_i.

The structure of the (H, I, J) partition—whose value depends on the value of the cache memory

[4]This expression of the rate is a slight approximation that we use here for simplicity as it is more intuitive. An exact and complete description of the achievable rate is given in Section 2.5.1.

[5]This type of partition is defined in Definition 1 and elaborated upon in Section 2.5.1.

M—that we have chosen allows us to efficiently compute it for all values of M. Indeed, we provide an algorithm in Section 2.5.1 that can find this partition and its corresponding memory-sharing parameters α_i, for all values of M, in $\Theta(L^2)$ running time. Briefly, as M is increased, levels get "promoted" from the set H to I to J. The sequence of these promotions is directly determined by the popularity of the levels.

We now discuss the order-optimality of the memory-sharing scheme in the multi-user setup. We develop new, non-cut-set lower bounds on the optimal rate, which use sliding-window entropy inequalities [41], and show that the scheme achieves a rate that is within a constant factor of the optimal. Note that this constant is independent of all the problem parameters except the largest AP access degree D.

Theorem 4. *For all valid values of the problem parameters K, L, $\{N_i, U_i, d_i\}_i$, and M, we have:*

$$\frac{R^{MU}(M)}{R^*(M)} \leq cD,$$

where $R^{MU}(M)$ is the rate achieved by memory-sharing, $R^*(M)$ is the optimal rate over all strategies, D is the largest AP access degree $D = \max_i d_i$, and c is a constant (independent of all problem parameters).

The above theorem holds for all values of d_i, although the computed value of c differs depending on the divisibility of K by the d_i's. When d_i divides K for all i, then $c = 9909$. When not all d_i's divide K, it follows from (2.7), (2.6), and Theorem 1 that the achievable rate increases by a factor of 4. Since the same converse results hold in both cases, that means that c is four times as large when some d_i does not divide K.

The gap between the rate achieved by the memory-sharing strategy and the optimal rate is linear in D. As we have argued earlier, we would not expect a situation where one user connects to too large a number of APs, and so D can be thought of as a constant. Furthermore, the constant c in Theorem 4 is rather loose so as to simplify the analysis. Numerics show that, in practice, this constant is much smaller. For example, if we have $K = 20$ caches, $L = 3$ popularity levels consisting of $(N_1, N_2, N_3) = (200, 20\,000, 800\,000)$ files, $(U_1, U_2, U_3) = (10, 5, 1)$ users per cache, and access degrees of $(d_1, d_2, d_3) = (1, 1, 1)$, then the gap is less than 6.8.

The lower bounds needed to prove Theorem 4 have to include the effect of all the popularity levels on the transmission rate. However, these effects can be very different, especially when some files are much more popular than others, something that cut-set bounds alone cannot account for. Using sliding-window subset entropy inequalities [41], we can combine multiple cut-set bounds that correspond to the different levels, without making any assumptions on the achievability scheme. The resulting bounds bring out the necessity for memory-sharing.

2.4.2 Single-user setup

In the single-user setup, the scheme that we propose is quite different. Instead of separating the levels, we *cluster* a subset of them into a super-level that will be treated as essentially one level. Specifically, we partition the levels into two subsets: H' and I'. The set I' will be clustered into one super-level, and all of the memory M will be given to it, while H' will be given no memory.

To understand how to choose H' and I', consider the following rough analysis. Suppose that all levels except one (let's call it j) have been split into H' and I'. Then, the rate, using Theorem 1, would be:

$$\begin{aligned} R &= R^{\mathrm{SL}}(0, \textstyle\sum_{h\in H'} K_h, \sum_{h\in H'} N_h, 1, 1) \\ &\quad + R^{\mathrm{SL}}(M, \textstyle\sum_{i\in I'} K_i, \sum_{i\in I'} N_i, 1, 1) \\ &\approx \sum_{h\in H'} K_h + \frac{\sum_{i\in I'} N_i}{M}. \end{aligned}$$

If we were to add level j to H', that would result in the addition of a K_j term, since all K_j requests would be completely served by the broadcast. On the other hand, if it is added to I', then we would get an additional N_j/M term, since the total number of files in I' would increase by N_j. Clearly, it is beneficial to choose the smaller of the two quantities.

Though the above analysis is rough, its main idea still holds. In general, we choose the partition (H', I') as follows:

$$H' = \left\{ h \in \{1, \ldots, L\} : M < \frac{N_h}{K_h} \right\}; \quad I' = (H')^c. \tag{2.8}$$

Then, by giving all of the memory to I', we can apply a single-level caching-and-delivery scheme

to obtain the rate in the following theorem.

Theorem 5. *Consider the multi-level, single-user setup with L levels, N_i files and K_i users for each level i, and cache memory M. Then, the following rate is achievable:*

$$R^{SU}(M) = \sum_{h \in H'} K_h + \max\left\{ \frac{\sum_{i \in I'} N_i}{M} - 1 \, , \, 0 \right\},$$

where H' and I' are as defined in (2.8).

This scheme turns out to be order-optimal, as we state in the next theorem.

Theorem 6. *The rate achieved by the clustering strategy in Theorem 5 is within a constant multiplicative factor of the information-theoretic optimum:*

$$\frac{R^{SU}(M)}{R^*(M)} \leq 72,$$

where $R^{SU}(M)$ is the rate achieved by clustering, and $R^*(M)$ is the information-theoretically optimal rate.

Unlike in the multi-user case, cut-set bounds are sufficient to show order-optimality in this case. Indeed, a single cut-set bound allows us to capture the fact that the user profile (i.e., the level of the file requested at each cache, as defined in Section 2.2.2) is not determined beforehand. At the same time, it brings out the necessity of clustering levels by mixing their demands. As before, however, these bounds do not make any assumptions on the achievability strategy.

The scheme suggested in Theorem 5 is similar to the results in [11] for Zipf popularity distributions and in [12] for arbitrary distributions. However, this is done for the multi-level popularity model, and Theorem 6 establishes a universal approximation for worst case rate, rather than average rate.

2.4.3 Comparison with Greedy Level Placement

A simple and natural caching strategy for the multi-level popularity model is one that places the most popular levels in every cache, as many as the cache can hold, and performs an uncoded

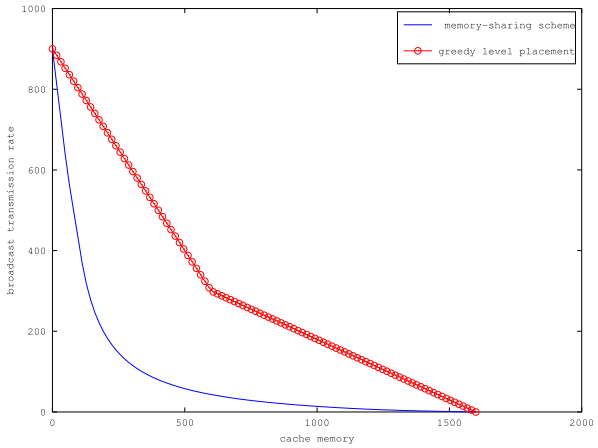


Figure 2.9: Comparison of the memory-sharing scheme with GLP, in the multi-user setup.

delivery. We refer to this strategy as Greedy Level Placement (GLP). To be specific, GLP places the most popular files that the caches can fully store. This leaves the next most popular level, which can only be partially stored; GLP stores a fraction of every file from that level. The delivery phase of GLP is uncoded and consists of multiple-unicast. We give an example for each of the two setups, to show how the respectively chosen schemes are superior to GLP in each context.

Multi-user example

Consider two levels ($L = 2$), with $K = 30$, $(N_1, N_2) = (600, 1000)$, $(U_1, U_2) = (20, 10)$, and $(d_1, d_2) = (1, 1)$. The rates achieved by the memory-sharing strategy and GLP are plotted against memory M in Figure 2.9. Memory-sharing performs up to 29 times better than GLP in this example.

Single-user example

Consider two levels ($L = 2$), with $(N_1, N_2) = (500, 1000)$ and $(K_1, K_2) = (30, 15)$. The rates achieved by the clustering strategy and GLP plotted against memory M in Figure 2.10. Clustering performs up to 22 times better than GLP in this example.

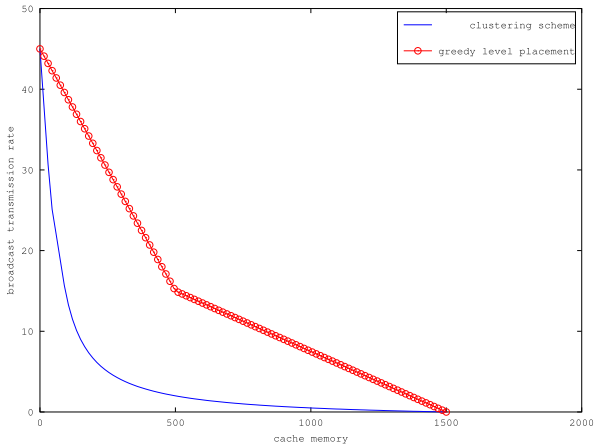


Figure 2.10: Comparison of the clustering scheme with GLP, in the single-user setup.

2.5 The Multi-User Setup

2.5.1 Caching-and-delivery strategy: memory-sharing

In this section, we describe the strategy used to achieve the rate approximated by the expression in Theorem 3. Moreover, we will give an exact upper bound on the achieved rate.

We will proceed in two steps. First, we will discuss the (H, I, J) partition of the set of levels as first described in Section 2.4.1. This will be accompanied by an explanation of the memory-sharing parameters α_i, which indicate what fraction of memory each level gets. Second, we analyze the *individual rate* achieved by every level i, after allocating $\alpha_i M$ memory to it, and combine all levels to produce the total rate achieved by the scheme.

While we actually define the (H, I, J) partition based on the problem parameters, *and then* choose the α_i values accordingly, in this paragraph we will proceed in the opposite order so that we expose the intuition behind the choices. As discussed in Section 2.4.1, the strategy involves finding a good partition (H, I, J) of the set of levels, such that levels in H are given no memory, levels in J are given maximal memory, and levels in I share the rest. Thus, for all levels $h \in H$, we will assign $\alpha_h M = 0$. Similarly, every level $j \in J$ will receive $\alpha_j M = N_j/d_j$, since that is the amount of memory needed to completely store level j and hence to require no BC transmission (i.e., $R^{\mathrm{SL}} = 0$; see Theorem 1). What is left is to share the remaining memory $\left(M - \sum_{j\in J} N_j/d_j\right)$ among the levels in I. More popular files should get more memory, and the popularity of a level i

is proportional to U_i/N_i. Thus, we choose to give level i a memory roughly $\alpha_i M \propto N_i \cdot \sqrt{U_i/N_i}$ (hence the memory *per file* is proportional to $\sqrt{U_i/N_i}$).[6]

Intuitively, we want to choose H, I, and J such that the above values of α_i are valid, i.e., $\alpha_i \in [0,1]$ for all i. Based on the intuition, we get the partition described below.

Definition 1 (M-feasible partition)**.** *For any cache memory M, an M-feasible partition (H, I, J) of the set of levels is a partition that satisfies:*

$$\begin{aligned}
&\forall h \in H, && \tilde{M} < \frac{1}{K}\sqrt{\frac{N_h}{U_h}};\\
&\forall i \in I, && \frac{1}{K}\sqrt{\frac{N_i}{U_i}} \leq \tilde{M} \leq \left(\frac{1}{d_i} + \frac{1}{K}\right)\sqrt{\frac{N_i}{U_i}};\\
&\forall j \in J, && \left(\frac{1}{d_j} + \frac{1}{K}\right)\sqrt{\frac{N_j}{U_j}} < \tilde{M},
\end{aligned}$$

where $\tilde{M} = (M - T_J + V_I)/S_I$, and, for any subset A of the levels:

$$S_A = \sum_{i \in A} \sqrt{N_i U_i}; \quad T_A = \sum_{i \in A} \frac{N_i}{d_i}; \quad V_A = \sum_{i \in A} \frac{N_i}{K}.$$

We stress again that, while we used our own α_i values to determine (H, I, J), this was done only for the intuition behind the choice. The partition itself is defined solely based on the problem parameters, and not on our strategy.

Notice in Definition 1 how the different sets are largely determined by the quantity $\sqrt{N_i/U_i}$, for each level i. This matches the idea that the most popular levels (i.e., those with the *smallest* N_i/U_i) will be in J, while the least popular levels (those with the *largest* N_i/U_i) will go to the set H.

After choosing an M-feasible partition, we share the memory among the levels using the fol-

[6] The square root comes from minimizing an inverse function of α_i.

lowing (precise) values of α_i:

$$\begin{aligned} &\forall h \in H, \qquad \alpha_h M = 0; \\ &\forall i \in I, \qquad \alpha_i M = \sqrt{N_i U_i} \cdot \tilde{M} - N_i/K; \\ &\forall j \in J, \qquad \alpha_j M = N_j/d_j. \end{aligned} \tag{2.9}$$

For completeness, the following proposition states the validity of this choice of memory-sharing parameters.

Proposition 1. *The values of the memory-sharing parameters defined in* (2.9) *satisfy:*

1. $\alpha_i \geq 0$ *for all* i*;*
2. $\sum_i \alpha_i = 1$*;*
3. $\alpha_i M \leq N_i/d_i$ *for all* i*.*

Note that points 1 and 2 imply $\alpha_i \in [0, 1]$*.*

Proof. Points 1 and 3 trivially follow from (2.9) for levels in H and J. For levels in I, they follow from applying the inequalities of Definition 1 on $\tilde{M}$ in (2.9). Point 2 follows from (2.9) and the definition of $\tilde{M}$ in Definition 1. □

The structure of the solution described above allows us to efficiently compute the α_i values. Indeed, Algorithm 1 finds α_i for all levels i and *for all memory values* M in $\Theta(L^2)$ running time, where L is the total number of levels. While a detailed description and analysis of the algorithm is given in Appendix A.1.3, we briefly go over it in this section. It proceeds in three main steps. In the first step, the algorithm identifies the sequence of (H, I, J) partitions that will occur as M increases, using only the problem parameters. For example, if there are two levels, there are two possible sequences, denoted below as S1 and S2:

		$M=0$				large M
S1	H	$\{1,2\}$	$\{2\}$	$\emptyset$	$\emptyset$	$\emptyset$
	I	$\emptyset$	$\{1\}$	$\{1,2\}$	$\{2\}$	$\emptyset$
	J	$\emptyset$	$\emptyset$	$\emptyset$	$\{1\}$	$\{1,2\}$
S2	H	$\{1,2\}$	$\{2\}$	$\{2\}$	$\emptyset$	$\emptyset$
	I	$\emptyset$	$\{1\}$	$\emptyset$	$\{2\}$	$\emptyset$
	J	$\emptyset$	$\emptyset$	$\{1\}$	$\{1\}$	$\{1,2\}$

Notice that the M-feasible partition is the same throughout an entire interval of values for M. In fact, there are only $2L$ non-trivial intervals: each interval is distinguished from the previous one by the "promotion" of a level from H to I or from I to J. As a result, computing only $2L$ intervals (actually, $(2L+2)$ intervals, to include the boundary cases) is enough to determine the M-feasible partition for *all* values of $M \geq 0$.

While Step 1 determines which of the above two sequences (S1 and S2) will occur, it does not determine the boundaries of the different regimes, i.e., the memory values at which the (H, I, J) partition changes (except for trivial boundaries like $M = 0$). The second step computes these boundaries, using not only the problem parameters, but also the partitions themselves.

For every M, the corresponding M-feasible partition is determined by the algorithm based on the boundaries calculated in Step 2. For example, suppose the algorithm decided that sequence S1 should be in use, and that the boundaries of $(H, I, J) = (\emptyset, \{1,2\}, \emptyset)$ are some values m_1 and m_2. Then, any $M \in [m_1, m_2]$ has $(\emptyset, \{1,2\}, \emptyset)$ as its M-feasible partition.

The following lemma presents two important properties of M-feasible partitions. It will be proved in Appendix A.1.3.

Lemma 2. *For any cache memory M, an M-feasible partition (H, I, J) always exists. Furthermore, the set I is never empty as long as $M \leq \sum_i N_i/d_i$, i.e., as long as individual caches do not have enough memory to store everything.*

To properly analyze the achievable rate, we need to look more closely at the set I. In the single-level, single-access scenario in [4, 5], three regimes were identified, and they were analyzed separately. Generalizing to the single-level, multi-access case, these regimes are: when $M < N/K$,

Algorithm 1 An algorithm that constructs an M-feasible partition for all M.

Require: Number of caches K and parameters $\{N_i, U_i, d_i\}_i$ for $i = 1, \dots, L$.
Ensure: An M-feasible partition for all M.

for all $i \in \{1, \dots, L\}$ **do**
  $\tilde{m}_i \leftarrow (1/K)\sqrt{N_i/U_i}$
  $\tilde{M}_i \leftarrow (1/d_i + 1/K)\sqrt{N_i/U_i}$
end for
$(x_1, \dots, x_{2L}) \leftarrow \mathrm{sort}(\tilde{m}_1, \dots, \tilde{m}_L, \tilde{M}_1, \dots, \tilde{M}_L)$.

Step 1: Determine (H, I, J) for each interval (x_t, x_{t+1}).
Set $H_0 \leftarrow \{1, \dots, L\}$, $I_0 \leftarrow \emptyset$, $J_0 \leftarrow \emptyset$.
for $t \leftarrow 1, \dots, 2L$ **do**
  if $x_t = \tilde{m}_i$ for some i **then**
    Promote level i from H to I
    $H_t \leftarrow H_{t-1} \setminus \{i\}$
    $I_t \leftarrow I_{t-1} \cup \{i\}$
    $J_t \leftarrow J_{t-1}$
  else if $x_t = \tilde{M}_i$ for some i **then**
    Promote level i from I to J
    $H_t \leftarrow H_{t-1}$
    $I_t \leftarrow I_{t-1} \setminus \{i\}$
    $J_t \leftarrow J_{t-1} \cup \{i\}$
  end if
end for

Step 2: Compute the limits of the intervals as $[Y_t, Y_{t+1})$.
for all $t \in \{1, \dots, 2L\}$ **do**
  $Y_t \leftarrow x_t \cdot S_{I_t} + T_{J_t} - V_{I_t}$
end for
$Y_{2L+1} \leftarrow \infty$ ▷ For convenience

Step 3: Determine the M-feasible partition for all M.
for all $t \in \{1, \dots, 2L\}$ **do**
  Set (H_t, I_t, J_t) as the M-feasible partition of *all* $M \in [Y_t, Y_{t+1})$
end for

when $M > cN/d$ for some constant $c \in (0,1)$, and the intermediate case. We identify three similar regimes for each level in i. Formally, define:

$$\begin{aligned} I_0 &= \left\{ i \in I : \tilde{M} < \frac{2}{K}\sqrt{\frac{N_i}{U_i}} \right\}; \\ I_1 &= \left\{ i \in I : \tilde{M} > \left(\frac{\beta}{d_i} + \frac{1}{K}\right)\sqrt{\frac{N_i}{U_i}} \right\}; \\ I' &= I \setminus (I_0 \cup I_1), \end{aligned} \tag{2.10}$$

By choosing the α_i values in (2.9), these definitions are equivalent to: I_0 is the set of levels i such that $\alpha_i M < N_i/K$; I_1 is such that $\alpha_i M > \beta N_i/d_i$ for all $i \in I_1$, where β is the level-separation factor as defined in Regularity Condition (2.4); and I' consists of the remaining levels.

When $K \geq D/\beta$, then I_0, I', and I_1 are mutually exclusive and form a partition of I. For convenience, we call the resulting partition (H, I_0, I', I_1, J) a *refined M-feasible partition.*

What follows is an important statement regarding the set I_1.

Proposition 2 (Size of I_1)**.** *In any refined M-feasible partition (H, I_0, I', I_1, J) as defined in Definition 1 and* (2.10)*, the set I_1 contains at most one element.*

Proof. Suppose that there exist two levels $i, j \in I_1$ (and possibly others). We will show that this violates regularity condition (2.4).

Suppose without loss of generality that i is more popular than j. Since $i, j \in I_1$, then, by Definition 1 and (2.10):

$$\left(\frac{\beta}{d_j} + \frac{1}{K}\right)\sqrt{\frac{N_j}{U_j}} < M \leq \left(\frac{1}{d_i} + \frac{1}{K}\right)\sqrt{\frac{N_i}{U_i}}.$$

However, this means:

$$\sqrt{\frac{U_i/N_i}{U_j/N_j}} < \frac{1/d_i + 1/K}{\beta/d_j + 1/K} < \frac{d_j}{\beta d_i} \leq \frac{D}{\beta},$$

which contradicts regularity condition (2.4). □

Using the definition of a refined M-feasible partition, and the values of $\{\alpha_i\}_i$, we give upper bounds on the rates achieved individually for each level.

Lemma 3. *Given a refined M-feasible partition (H, I_0, I', I_1, J), the individual rates of the levels are upper-bounded by:*

$$\begin{aligned}
&\forall h \in H, && R_h(M) = KU_h;\\
&\forall i \in I_0 \cup I', && R_i(M) \le \frac{2S_I\sqrt{N_iU_i}}{M - T_J + V_I};\\
&\forall i_1 \in I_1, && R_{i_1}(M) \le \frac{1}{\beta} d_{i_1} U_{i_1} \left(1 - \frac{M - T_J}{N_{i_1}/d_{i_1}}\right)\\
& && \quad + \frac{1}{\beta} d_{i_1} U_{i_1} \frac{S_{I_0} + S_{I'}}{\sqrt{N_{i_1}U_{i_1}}};\\
&\forall j \in J, && R_j(M) = 0.
\end{aligned}$$

The total achieved rate is then: $R(M) = \sum_{i=1}^{L} R_i(M)$.

We relegate the proof of Lemma 3 to Appendix A.1.3. What follows is a brief explanation of the individual rates.

- Since levels $h \in H$ get no memory, no information about their files can be stored in the caches. Hence, the server will have to transmit a complete copy of every file requested by the KU_h users in level h. Therefore, $R_h(M) = KU_h$ in the worst case.

- Since levels $j \in J$ get maximal memory, any file requested from j can be fully recovered using the caches. Thus there is no need for the server to send anything for j, and hence $R_j(M) = 0$.

- Finally, the levels $i \in I$ get enough of the remaining memory $M - T_J$ so that they behave as in Theorem 1. However, since $\alpha_i M \approx \sqrt{N_iU_i}/S_I \cdot (M - T_J)$, we get a rate of:

$$R_i(M) \approx \frac{N_iU_i}{\alpha_i M} - d_iU_i \approx \frac{S_I\sqrt{N_iU_i}}{M - T_J} - d_iU_i.$$

  Notice how $R_i(M)$ is inversely proportional to the memory remaining after storing J. This behavior is captured in the expressions for $I_0 \cup I'$. However, for the levels in $i_1 \in I_1$, which get almost, but not quite, maximal memory, the individual rate $R_{i_1}(M)$ behaves more closely to a linear function of the memory.

2.5.2 Outer bounds

In the single-level setup introduced in [4], cut-set lower bounds on the optimal rate were sufficient for proving order-optimality of the scheme. The idea was to choose cuts that include a certain number of caches s, as well as enough broadcast messages to decode almost all files when combined with the s caches. The choice of s depended most importantly on the size of the memory M. For example, for very large M, the bounds would reflect the fact that a single cache should contain enough information so that it need only be combined with a small broadcast message to decode one file. In contrast, for small M, the bounds would instead show that a large broadcast message is needed to serve all s users.

In our multi-level situation, the goal of our lower bounds is to show that different popularity levels require different amounts of memory, ranging from zero to maximal memory. However, a single cut-set bound would force all levels to abide by the same number of caches s, a choice that would not reflect the plurality of memory allocation. New lower bounds were thus required.

The lower bounds that we use allow choosing for each level i the number of caches s_i that corresponds to the amount of memory we expect it to get. In essence, each level i will exclusively be part of a cut-set bound with s_i caches, which will give a lower bound that loosely corresponds to the individual rate R_i for this level.

The challenge is to incorporate all of these cut-set bounds together, such that the same caches appear in cut-set bounds for all levels. This process is described next. We start with a cut-set bound for the level with the smallest s_i. Using Fano's inequality, we can derive lower bounds on its individual rate. We then proceed to lower-bound the current quantity using a cut-set bound for the level with the next smallest s_i, and repeat the process. The transition from one cut-set bound to the next can be achieved using sliding-window subset entropy inequalities [41, Theorem 3].[7]

Lemma 4 (Sliding-window subset entropy inequality [41, Theorem 3])**.** *Given K random variables $(Y_1, \ldots, Y_K)$, we have, for every $s \in \{1, \ldots, K-1\}$:*

$$\frac{1}{s}\sum_{i=1}^{K} H\left(Y_i, \ldots, Y_{\langle i+s-1\rangle}\right) \geq \frac{1}{s+1}\sum_{i=1}^{K} H\left(Y_i, \ldots, Y_{\langle i+s\rangle}\right),$$

[7] For these inequalities to work, we have to take not just one cut-set bound per level, but an averaging of K similar cut-set bounds that ends up including all caches an equal number of times.

where we define $\langle i \rangle = i$ *if* $i \leq K$ *and* $\langle i \rangle = i - K$ *if* $i > K$.

The resulting lower bounds, in their final form, are given in the following lemma.

Lemma 5. *Consider the multi-level, multi-user caching setup. Let* $b \in \mathbb{N}^+$ *and* $t \in \{1, \ldots, K\}$. *Furthermore, for every level* i, *let* $s_i \in \mathbb{N}^+$ *such that* $s_i t \in \{d_i, \ldots, \lfloor K/2 \rfloor\}$. *Then, for every memory* M, *the optimal rate can be bounded from below by:*

$$R^*(M) \geq \sum_{i=1}^{L} \lambda_i \cdot \min \left\{ (s_i t - d_i + 1) U_i \, , \, \frac{N_i}{s_i b} \right\} - \frac{t}{b} M,$$

where λ_i *is a constant (introduced for technical reasons) defined as* $\lambda_i = 1$ *if* $s_i t = d_i$ *and* $\lambda_i = \frac{1}{2}$ *if* $s_i t > d_i$.

The full proof of Lemma 5 is given in Appendix A.1.1. However, we will here give a brief intuition behind the expression shown above. Every term in the sum corresponds to a level i. Consider a single level i. Roughly, if we ignore all the other levels in the summation, and assuming $d_i = 1$ for simplicity, then the inequality can be rearranged approximately as follows:

$$s_i b R + s_i t M \geq \min \left\{ s_i t \cdot s_i b \cdot U_i \, , \, N_i \right\}.$$

This expression is essentially a cut-set bound, saying that, with $s_i t$ caches and $s_i b$ broadcast messages, up to $s_i t \cdot s_i b \cdot U_i$ files from level i can be decoded (since there are U_i users per cache for this level), unless this number exceeds the total number of files N_i.

2.5.3 Approximate optimality

In order to prove order-optimality of the memory-sharing scheme, we must use Lemma 5 with appropriate parameters. Our goal is to get a resulting lower bound on the optimal rate $R^*(M)$ such that the ratio between the achievable rate and $R^*(M)$—henceforth called the *gap*—is minimized. More specifically, the result we seek is a constant upper bound on the gap.

For technical reasons, several cases need to be considered for which different values are chosen for the parameters in Lemma 5. In particular, the values of choice in one case would violate the

conditions imposed on them in another case. In this section, we will illustrate the proof methodology by approximating one of these cases. The complete and rigorous proof is given in Appendix A.1.2.

Recall the refined M-feasible partition introduced in Definition 1 and in (2.10). As previously mentioned, this partition only depends on the problem parameters, not on the achievability strategy. The cases of interest for the proof of order-optimality are:

- Case 1a: $I_1 = \emptyset$ and $J \neq \emptyset$;
- Case 1b: $I_1 = \emptyset$ and $J = \emptyset$;
- Case 2: $I_1 \neq \emptyset$.

There is also a special Case 0 for when K is small (specifically, $K < D/\beta$). All the other cases assume K is large.

The case that we will focus on in this section is Case 1a. Note that $I_1 = \emptyset \implies I = I_0 \cup I'$. For simplicity, we will assume that $d_i = 1$ for all i. Furthermore, most of the analysis will be approximate.

By Lemma 3, the achievable rate in Case 1a can be upper-bounded by:

$$R(M) \leq \sum_{h \in H} KU_h + \frac{2S_I^2}{M - T_J}, \tag{2.11}$$

since $S_I = \sum_{i \in I} \sqrt{N_i U_i}$ (see Definition 1). Now consider Lemma 5 with the following parameters:

$$\begin{aligned} & t = 1; \\ \forall h \in H, \quad & s_h \approx \frac{1}{2} K; \\ \forall i \in I, \quad & s_i \approx \frac{\sqrt{N_i/U_i}}{2\tilde{M}}; \\ \forall j \in J, \quad & s_j = 1; \\ & b \approx 4\tilde{M}^2, \end{aligned}$$

where, as in Definition 1, $\tilde{M} = (M - T_J + V_I)/S_I \approx (M - T_J)/S_I$. The values of the s_i's are very similar to those used in the single-level setup in [4]. Indeed, the levels with the smallest memory

($h \in H$) are handled using cut-set bounds that consider a fraction of the total number of caches, while the levels with the largest memory ($j \in J$) are handled with bounds that consider only one cache. Thus, the lemma lower-bounds the optimal rate by:

$$R^*(M) \geq \sum_{h\in H} \min\left\{s_h U_h, \frac{N_h}{s_h b}\right\} + \sum_{i\in I} \min\left\{s_i U_i, \frac{N_i}{s_i b}\right\} + \sum_{j\in J} \min\left\{s_j U_j, \frac{N_j}{s_j b}\right\} - \frac{M}{b}.$$

Substituting the values of the parameters, and utilizing the inequalities on $\tilde{M}$ that define the refined M-feasible partition, we get:

$$\begin{aligned} R^*(M) &\geq \sum_{h\in H} \frac{1}{2} K U_h + \sum_{i\in I} \frac{\sqrt{N_i U_i} S_I}{2(M-T_J)} + \sum_{j\in J} \frac{N_j}{b} - \frac{M}{b} \\ &= \sum_{h\in H} \frac{1}{2} K U_h + \frac{S_I^2}{2(M-T_J)} - \frac{M-T_J}{b} \\ &= \sum_{h\in H} \frac{1}{2} K U_h + \frac{S_I^2}{2(M-T_J)} - \frac{(M-T_J)S_I^2}{4(M-T_J)^2} \\ &= \sum_{h\in H} \frac{1}{2} K U_h + \frac{S_I^2}{4(M-T_J)} \\ &\overset{(a)}{\geq} \frac{1}{8} \cdot R(M). \end{aligned}$$

The last inequality (a) is due to (2.11).

Thus, the result is a constant multiplicative gap (8 in this example) between the achievable rate and the optimal rate for this regime. The full proof in Appendix A.1.2 derives such gaps for each one of the four regimes mentioned above (Cases 0, 1a, 1b, and 2), and finally combines them into one multiplicative gap that holds for all cases.

2.5.4 Comparison with different memory-sharing strategies

The proposed memory-sharing scheme relies on a very specific division of the memory among the levels, i.e., a very specific choice of the $\{\alpha_i\}_i$ parameters. While this (rather complex) choice is approximately optimal, a natural question that arises is if it is necessary. In other words, could

a simpler memory-sharing achieve approximately the same results? In this section, we compare the performance of our strategy with that of the following two different, simpler memory-sharing schemes.

- **Greedy Memory-Sharing (GMS):** This scheme gives as much memory as possible for the most popular levels.
- **Uniform Memory-Sharing (UMS):** This scheme ignores all popularities and gives the same amount of memory to all files.

We note that these two schemes employ the same coded delivery phase as ours; the only difference is in how to divide the memory between the levels.

The first thing to note is that GMS and UMS are both special cases of our memory-sharing scheme—which we will call **"Optimal Memory-Sharing" (OMS)** in this section to distinguish it from the other two. Indeed, if all levels have exactly the same popularities, then OMS will give all files equal memory. On the other hand, when the level popularities are separated enough, then OMS will end up always prioritizing the most popular levels, and thus it reduces to GMS. However, OMS can provide benefits in the middle case that both other schemes lack. In fact, GMS and UMS can perform arbitrarily worse than our memory-sharing strategy.

As an example, consider a setup with $L = 2$ levels, such that $U_1 = \sqrt{K}U_2$ and $N_2 = \sqrt{K}N_1$. Suppose $M = N_1$, and assume $d_1 = d_2 = 1$. Under our strategy, both levels would be in the set I, and hence the rate would be

$$R^{\mathrm{MU}} \approx \frac{\left(\sqrt{N_1 U_1} + \sqrt{N_2 U_2}\right)^2}{N_1} = 4\sqrt{K}U_2.$$

GMS will completely store the first level, yielding a rate of

$$R_{\mathrm{GMS}} = KU_2.$$

Finally, UMS would give equal memory to both levels, effectively merging them, giving a rate of

$$\begin{aligned} R_{\text{UMS}} &\approx \frac{N_1 U_1}{M \cdot N_1/(N_1+N_2)} + \frac{N_2 U_2}{M \cdot N_2/(N_1+N_2)} \\ &= \frac{(N_1+N_2)(U_1+U_2)}{N_1} \\ &= \left(\sqrt{K}+1\right)^2 U_2. \end{aligned}$$

By comparing these schemes to our own, we see that $R_{\text{GMS}}/R^{\text{MU}} = \Theta(\sqrt{K})$ and $R_{\text{UMS}}/R^{\text{MU}} = \Theta(\sqrt{K})$. Therefore, there are regimes where both GMS and UMS perform arbitrarily worse than OMS.

2.6 The Single-User Setup

2.6.1 Caching-and-delivery strategy: clustering

Proof of Theorem 5. Recall from Section 2.4.2 how the memory is divided among the sets H' and I', defined in (2.8): all of the available memory is given to levels in I', which is treated as one super-level. As a result, all requests for files from H' must be handled by complete file transmissions from the BS. Since there are $\sum_{h\in H'} K_h$ users making such requests, the result is the same amount of transmissions in the worst case. Therefore, the message sent to all users requesting from a level in H' has the following rate:

$$R_{H'} = \sum_{h\in H'} K_h. \tag{2.12}$$

For the set I', now considered as one super-level, we use the single-level strategy from [5]. Although only a subset of the caches is active in our setup, the same strategy still applies. Indeed, the placement in [5] is a random sampling of the files in all the caches; we do the same placement in this case. In the delivery phase, we now know the caches to which the users of I' connected. We perform a delivery as in [5], assuming that only these caches were ever present in the system.

For illustration, suppose that there were $K = 4$ caches. Furthermore, assume the partition (H', I') was performed such that 3 users will request files from the set of levels I'. In the placement phase, we store a random sample of the files in I' in each of the four caches. In the delivery phase,

suppose the three users requesting files from I' connect to caches 1, 2, and 4. In this case, the BS will send a complete file transmission for user 3 (since he requests a file from H'), and will treat users 1, 2, and 4 as though they were part of a single-level caching system with only three caches.

The rate required for I' can be directly derived from Lemma 1, using $\sum_{i\in I'} K_i$ caches, $\sum_{i\in I'} N_i$ files, and 1 user per cache. In addition, we have, from (2.8), that $M \geq N_i/K_i$ for all $i \in I'$. This implies $M \geq (\sum_{i\in I'} N_i)/(\sum_{i\in I'} K_i)$, and hence the rate for I' is:

$$R_{I'} = \max\left\{\frac{\sum_{i\in I'} N_i}{M} - 1\,,\, 0\right\}. \tag{2.13}$$

The maximization with zero is needed because it is possible to have $M > \sum_{i\in I'} N_i$.

By combining (2.12) with (2.13), we get a total broadcast rate of:

$$R(M) = R_{H'} + R_{I'} = \sum_{h\in H'} K_h + \max\left\{\frac{\sum_{i\in I'} N_i}{M} - 1, 0\right\}.$$

This proves Theorem 5. □

It will be helpful for the later analysis to refine the partition (H', I') as follows.

Definition 2. *Define the following partition (G, H, I, J) of the set of levels:*

$$\begin{aligned} G &= \{g : M < N_g/K_g \text{ and } K_g \leq 5 \text{ and } M \leq N_g/6\}\,; \\ H &= \{h : M < N_h/K_h \text{ and } K_h \geq 6\}\,; \\ I &= \{i : N_i/K_i \leq M \leq N_i/6\}\,; \\ J &= \{j : M > N_j/6\}\,. \end{aligned}$$

We rewrite the achievable rate in terms of this new partition:

$$R(M) \leq 5\cdot|G| + \sum_{h\in H} K_h + \frac{\sum_{i\in I} N_i}{M} + \left[\frac{\sum_{j\in J} N_j}{M} - 1\right]^+, \tag{2.14}$$

where $[x]^+ = \max\{x, 0\}$, and keeping in mind that $K_g \leq 5$ for $g \in G$. If we define $N_J = \sum_{j\in J} N_j$,

we can upper-bound the last term by:

$$\left[\frac{N_J}{M}-1\right]^+ \leq \begin{cases} N_J/M & \text{if } M < N_J/6; \\ 6\left(1-M/N_J\right) & \text{if } N_J/6 \leq M < N_J; \\ 0 & \text{if } M \geq N_J. \end{cases} \tag{2.15}$$

Effectively, the set J now behaves as a unit.

Much of what defines the above partition hinges on the following question: What happens if we know the user profile *a priori*? If we do, we could imagine a strategy of total separation of the levels: every level i is treated as a single-level system with K_i caches and users, and N_i files. Each level can thus get the entirety of the memory of their specific K_i caches. Using Lemma 1, we can calculate the total rate for this hypothetical situation as:

$$R'(M) = \sum_{i=1}^{L} \min\left\{K_i, \frac{N_i}{M}\right\}\left(1-\frac{M}{N_i}\right)^+,$$

which can be approximated by:

$$R'(M) \leq 5\cdot|G| + \sum_{h\in H} K_h + \sum_{i\in I}\frac{N_i}{M} + 6\sum_{j\in J}\left(1-\frac{M}{N_j}\right)^+.$$

Interestingly, the G, H, and I terms in this rate expression are exactly the same as those in (2.14). Therefore, except for levels $j \in J$ where $M > N_j/6$, our scheme would not benefit from prior knowledge of the user profile. However, the set J is in general limited by the lack of this knowledge. Indeed, if $M > N_j$ for all $j \in J$ but $M < N_J = \sum_{j\in J} N_j$, then it is possible to store any level in every cache, but it is not possible to store all levels in all caches. Thus knowing which level is at which cache can bring the rate down to zero.

Finally, G and H, which are both subsets of H' (though not necessarily a partition; the definition of J allows some of its levels to be in H' too) were separated because levels behave differently when their number of caches is very small. Specifically, a level in H can transition into I as the memory increases, but a level in G immediately jumps to J since $N_g/K_g > N_g/6$.

2.6.2 Outer bounds and approximate optimality

As mentioned in Section 2.4.2, we use a cut-set bound to lower-bound the optimal rate. The idea is to send a certain number b of broadcast messages $X_1, \ldots, X_b$ that serve certain requests. We choose these requests as follows. For every level $i \in G \cup H \cup I$, consider a certain number $s_i \leq K_i$ of caches. These caches are distinct across levels. For all the b broadcasts, the users connected to these s_i caches will altogether request $s_i b$ distinct files from level i if there are that many; otherwise they request all N_i files. For the levels in the set J, we collectively consider some s_J caches (distinct from the rest). The users at these s_J caches will use all b broadcasts to decode as many files from the set J as possible, up to $s_J b$ files. Let n_J denote this number; it will be determined later.

If we let $S = \sum_{i \notin J} s_i + s_J$ be the total number of caches considered, then, by Fano's inequality:

$$\begin{aligned} bR + SM &\geq H\left(Z_1, \ldots, Z_S, X_1, \ldots, X_b\right) \\ &\geq \sum_{i \notin J} \min\left\{s_i b, N_i\right\} + n_J \\ R^*(M) &\geq \sum_{i \notin J} s_i \left(\min\left\{1, \frac{N_i}{s_i b}\right\} - \frac{M}{b}\right) \\ &\quad + s_J \left(\frac{n_J}{s_J b} - \frac{M}{b}\right) \\ &= \sum_{i \notin J} v_i + v_J. \end{aligned} \tag{2.16}$$

We will analyze each of the v_i and v_J terms separately. We identify two cases for which the analysis is slightly different.

The first case is when $M < 1/6$. Because of regularity condition (2.3), this implies $M < 1/6 < 1 \leq N_i/K_i$ for all levels i, and thus the achievable rate can be bounded by:

$$R(M) \leq \sum_{i=1}^{L} K_i. \tag{2.17}$$

The second case, which is more interesting, is when $M \geq 1/6$.

The details of the analysis are given in Appendix A.2. We will here give a brief outline of the procedure, with approximations to avoid burying the essence of the argument under technicalities.

The idea is to lower-bound each term v_i or v_J to match the corresponding term in the rate expression. Let us focus on the case $M \geq 1/6$, and suppose we choose $b \approx 6M \geq 1$. For simplicity, we will only look at the sets H and I.

Consider a level $h \in H$, and let $s_h \approx K_h/6$. Then, by the definition of H:

$$\frac{N_h}{s_h b} = \frac{N_h}{K_h M} \geq 1.$$

As a result,

$$v_h = \frac{K_h}{6}\left(1 - \frac{M}{6M}\right) = \frac{5}{36} \cdot K_h,$$

which matches the corresponding achievable rate term in (2.14), up to the constant (5/36).

Now consider a level $i \in I$, and let $s_i \approx N_i/(6M)$. Then,

$$\frac{N_i}{s_i b} = \frac{N_i}{(N_i/6M) \cdot 6M} = 1,$$

and hence:

$$v_i = \frac{N_i}{6M}\left(1 - \frac{M}{6M}\right) = \frac{5}{36} \cdot \frac{N_i}{M}.$$

Again, this matches the corresponding achievable rate term in (2.14), up to the constant (5/36).

Applying a similar procedure for every level, we get matching lower bounds (up to a constant) and thus prove Theorem 6.

2.6.3 Similarity to strategies in related work

As previously mentioned, our clustering scheme is similar to strategies discussed in the literature for similar setups with stochastic demands [12, 11]. In these setups, users request files based on a probability distribution, and the average rate is analyzed, as opposed to the worst-case rate in our case. In both cases, the proposed strategy is to divide the files into two sets, based on some threshold, and store only the most popular files.

Recall that our strategy clusters the levels i whose popularity is such that $M \geq N_i/K_i$. The popularity p_i of a file of level i is proportional to K_i/N_i, and hence if we normalize so that the sum

of popularities is 1, we get:

$$p_i = \frac{K_i/N_i}{\sum_{j=1}^{L} N_j \cdot K_j/N_j} = \frac{K_i}{KN_i}.$$

Thus, the condition $M \geq N_i/K_i$ can be rewritten as $p_i \geq 1/KM$. This is exactly the threshold used in [12] to determine which files to cluster and store in the caches, and which files to leave out of the caches.

2.7 Comparison of the Two Setups

In this section, we first compare the memory-sharing and the clustering strategies, and we explore the dichotomy among the two setups that is emphasized by the difference between strategies. We will then discuss why such a dichotomy exists, and explain the need for different lower bounds for each setup. Finally, we explore a new problem that combines both setups by including both multi-user and single-user levels.

2.7.1 Comparing the two caching-and-delivery strategies

We have previously argued that memory-sharing is the best scheme to use in the multi-user case, while clustering is the near-optimal strategy in the single-user case. However, could one (or both) of these schemes be good enough for both situations? We will show, in this section, that it is not the case: in the single-user setup, memory-sharing can give a gap between its rate and the optimum that increases linearly with L; meanwhile, the rate achieved by clustering in the multi-user case can be arbitrarily far from the optimal rate. We give examples of these two cases.

Consider a multi-user setup with two levels. Suppose that there is enough memory so that both levels are to be partially stored in the caches. With the memory-sharing scheme, this means $I = \{1, 2\}$, which would, by Theorem 3, give a rate of approximately:

$$\begin{aligned} R &\approx \frac{\left(\sqrt{N_1 U_1} + \sqrt{N_2 U_2}\right)^2}{M} \\ &= \frac{1}{M}\left[N_1 U_1 + N_2 U_2 + 2\sqrt{N_1 U_1 N_2 U_2}\right]. \end{aligned}$$

On the other hand, if we had clustered the two levels into one, then this super-level would have

$(N_1 + N_2)$ files and $(U_1 + U_2)$ users per cache, resulting in the following rate:

$$\begin{aligned} R &\approx \frac{(N_1 + N_2)(U_1 + U_2)}{M} \\ &= \frac{1}{M}\left[N_1U_1 + N_2U_2 + N_1U_2 + N_2U_1\right]. \end{aligned}$$

However, we know that the geometric mean of any two numbers is always smaller than their arithmetic mean. By considering the two numbers N_1U_2 and N_2U_1, we get:

$$2\sqrt{N_1U_2N_2U_1} \leq N_1U_2 + N_2U_1,$$

and specifically the ratio between them can get arbitrarily large when the popularities of the two levels become significantly different (i.e., $U_2/N_2 \ll U_1/N_1$). Intuitively, if the two levels had similar popularities, then memory-sharing gives them similar amounts of memory, effectively merging them. However, if their popularities were very different, then they should be given drastically different portions of the memory.

Consider now the single-user case with L levels, and suppose again that the memory is such that all levels will be partially stored. Let us assume that $N_1 = \cdots = N_L$. Using the clustering scheme, we get the following approximate rate:

$$R \approx \frac{N_1 + \cdots + N_L}{M} = \frac{LN_1}{M}.$$

However, with memory-sharing, we would get:

$$R \approx \frac{\left(\sqrt{N_1} + \cdots + \sqrt{N_L}\right)^2}{M} = \frac{L^2N_1}{M},$$

which is larger by a factor of L. Essentially, we are sending L broadcasts, one per level, when we could send just one broadcast for all L levels.

2.7.2 Analysis of the dichotomy between the setups

The dichotomy between the two extremes is striking. They require different strategies, and the strategy that is good for one setup is not so for the other. This suggests a fundamental difference between the two setups.

To understand this difference, consider what happens when sending a coded broadcast message. Each message targets a specific subset of users. If, in this subset, there exist two users that are connected to the same cache, then these users have access to the exact same side information. As a result, no coding can be done across these two users, and there is hence no benefit in including them in the same broadcast.

There are in fact two opposing forces at work on the caching-and-delivery strategy. The first is a popularity-centric force: it pushes on the strategy to allocate more memory to the more popular files. The second force is coding-centric: it encourages increasing the number of coding opportunities. These two forces are at odds, since coding *across levels* performs best when the files in these levels are given the same memory, regardless of popularity.

With that in mind, consider again Figure 2.2 and Figure 2.3. Notice how, in the multi-user setup, there are multiple rows of users, each of which consists of users from the *same* popularity level. Each such row is a complete set of users with no common caches: any additional users we add would have access to the same cache as some other user. Thus, it is sufficient to consider them in a broadcast transmission that is separate from all other rows. Since, as a result, no two levels will share the same broadcast message, it can only be beneficial to choose the best possible division of the memory, based on popularities. In the single-user setup, however, there is only one row of users that contains all the users from all the levels. It is hence possible to generate coding opportunities across levels. Merging is thus a better option in this situation, and merging is most efficient when all levels receive equal memory per file.

2.7.3 The difference in the lower bounds

Complementing the difference in the achievable strategies between the two setups, we see a difference in the lower bounds to the optimal rate. In the multi-user case, we use a combination of cut-set

bounds, one for each popularity level. However, we use a unified cut-set bound for all the levels in the single-user case. We elaborate on this difference in this section.

In the single-level setup studied in [4], cut-set bounds were given to lower-bound the optimal rate for every value of memory M. Depending on the value of M, a certain number of caches (and hence users) were considered and used in the cut-set bounds. Roughly speaking, about M broadcast messages are sent to the users at N/M distinct caches, allowing them to decode $M \cdot (N/M) = N$ files in total. In the multi-level setups (both multi-user and single-user), the lower bounds retain this idea. However, there are two crucial differences between the setups that force different choices of lower bounds.

The first difference is in the role of each cache vis-à-vis the popularity levels. In both setups, different levels are given different memory values. However, the *same* cache must be simultaneously used for *all* levels in the multi-user setup; in the single-user setup each cache is bound to a single level at any moment. As a result, a single cut-set bound can still encompass all levels in the single-user setup, but will not be enough in the multi-user setup.

The second difference is in the uncertainty of the user profile. In the single-user setup, there are situations where this uncertainty is significant enough to impact the achievable rate. This is especially true for levels nearing their maximal storage (the set J in Definition 2), as described in Section 2.6.1. The lower bounds should incorporate this notion by considering demands from different levels at the same cache. As a result, a single cut-set bound unifying all levels (at least the levels involved in this uncertainty) becomes necessary.

2.7.4 Mixing the setups

So far, we have looked at the two extremes: either all levels were represented at all the caches, or none of them were. A natural problem arises: that of studying intermediate cases. The simplest form such intermediate cases can take is one where levels of both types are present.

Specifically, there are two classes of popularity levels: $\mathcal{F}$ and $\mathcal{G}$. The class $\mathcal{F}$ consists of levels i that are represented by exactly U_i users *at every cache*. In contrast, there is exactly one row of users that represents all the levels in the class $\mathcal{G}$: each level $i \in \mathcal{G}$ is represented by K_i of those users.

The most natural strategy to employ in this situation would be to superpose the multi-user and the single-user strategies. In particular, we divide the memory M into γM and $(1-\gamma)M$, for some $\gamma \in [0,1]$. We give the first part to $\mathcal{F}$ and the second part to $\mathcal{G}$, and apply their respective strategies on their part of the memory. We believe this to be the best strategy, but proving its order-optimality requires developing new lower bounds that consider levels of both classes at the same time; this is part of our on-going work.

2.8 Discussion and Numerical Evaluations

In the previous sections, we presented theoretical results for any given set of popularity levels and associated user access structures. However, in practice, what is available is a "continuous" popularity distribution over the entire set of N files, and it is up to the designer to choose: (a) the number of popularity levels; (b) which files to assign to which level; and (c) the corresponding user access degree for each popularity level. For each such choice, our theoretical results characterize the minimum broadcast transmission rate, and we study, in this section, the impact of these choices on the transmission rate. Furthermore, while our theoretical model assumed that, for each popularity level, the number of users fixed, we relax this assumption here by allowing each user to randomly connect to one of the K APs and request a file stochastically, according to the underlying popularity distribution. Finally, we will also compare the performance of our scheme with that of a Greedy File Placement (GFP) approach, as well as the information-theoretic lower bounds presented before. These evaluations will only focus on the multi-user setup, as it is less discussed in the literature. Since the single-user setup utilizes a scheme similar to what is already in the literature for arbitrary distributions [12, 11], we do not feel it is necessary to include it in this discussion. However, we do provide a brief comparison of the clustering scheme with the literature in Section 2.6.3

We use a YouTube dataset [40] for our evaluations. This dataset provides the number of views of videos in a set of $N \approx 500\,000$, over some period of time. Thus, these views can be thought of as approximating the popularity of the videos. Figure 2.1 shows the popularity distribution of the videos (normalized number of views), which resembles a Zipf distribution similar to those commonly observed for multimedia content [21].

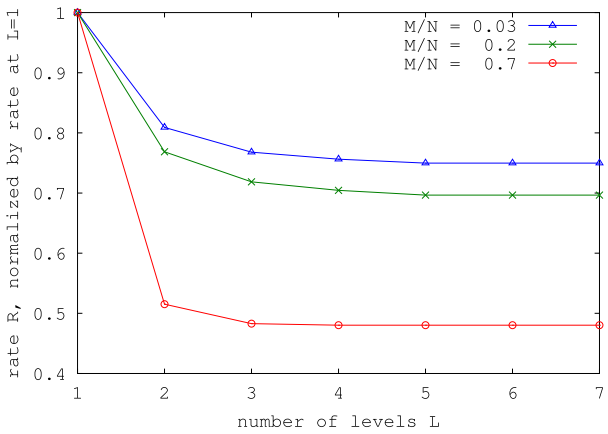


Figure 2.11: Rate achieved by the memory-sharing scheme vs. number of levels, for different values of cache memory. For each L, we choose the L levels that minimize the achievable rate (using brute-force search). For ease of comparison, the rate values have been normalized by the rate at $L = 1$.

When using the YouTube dataset in the following sections, we will often omit the total number of users considered. This is because the total broadcast rate is always directly proportional to the total number of users, when the *fraction* of users per level is fixed. In this situation, the fraction is determined by the distribution in Figure 2.1 and the levels into which the files were split, thus the total number of users will not affect the behavior of the system.

2.8.1 Discretizing a continuous popularity distribution

Our first step is to divide the files in the YouTube dataset into a certain number of levels, based on the popularity profile in Figure 2.1. Let there be $K = 75$ caches. We consider small, moderate, and large values of M/N (0.03, 0.2, and 0.7) and set the user access degree $d_i = 1$ for every level i, so as to study the impact of the number of levels on the broadcast rate in isolation. For increasing values of L, we find the division of the files into L levels that minimizes the rate achieved by the memory-sharing scheme using a brute-force search. We plot the minimum achievable rate versus L in Figure 2.11. As is easily apparent, while there is a significant gain in performance between treating all files as one level and dividing them into two levels, the gain decreases with diminishing returns as L increases. This shows the importance of dividing files into multiple levels, but also suggests that 3–4 levels are sufficient to derive most of the benefits.

We remind the reader that the popularity profile in Figure 2.1 is purely empirical. It is based

on the number of views of the videos in the dataset, collected over some period of time. By the very nature of the data, if two videos have received, let's say, 1000 and 1100 views, then this does not really imply that the first file is more popular than the other. This is especially true of videos with very small number of views. By grouping files into popularity classes, we acknowledge the difference in popularity of very different files, while simultaneously not distinguishing between files whose *empirical* popularities are close.

2.8.2 Impact of multi-access on the achievable rate

To study the effect of multi-access in isolation, we will fix the partition we use to divide the files into different levels and then look at different multi-access structures. Suppose again $K = 75$, and consider $L = 2$ levels, with $N_1 = 0.2N$, and $N_2 = 0.8N$ files. We plot in Figure 2.12 the broadcast rate of our scheme as a function of the normalized memory M/N, for four different access structures (d_1, d_2): $(1, 1)$, $(1, 2)$, $(2, 1)$, and $(2, 2)$.

As one would expect, allowing for multi-access greatly improves the transmission rate. For example, the rate for the multi-access system with $(d_1 = 2, d_2 = 2)$ is smaller than the rate for the single-access system with $(d_1 = 1, d_2 = 1)$. The cases $(d_1 = 1, d_2 = 2)$ and $(d_1 = 2, d_2 = 1)$ provide a more interesting comparison. For small memory size M, the former gives a lower rate since the cache memory mainly contains files from level 1, and so giving higher access to level 1 is more beneficial in reducing the rate. On the other hand, as M grows and files from level 2 start occupying a significant portion of the memory, it becomes more efficient to give higher access to level 2 since it has many more files than level 1.

While greater cache access helps reduce the rate, there is also a cost associated with it in terms of the increased delay in establishing connections with multiple APs, as well as a reduced communication rate as a user connects to farther APs. In general, for a given multi-level setup with parameters L, K, $\{N_i, U_i\}$, and M, such a cost can be included in the rate optimization framework as one or more inequalities of the form $\text{cost}_j(K, \{U_i, d_i\}_i) \leq C_j$, for some maximum cost C_j. The above optimization problem can be numerically solved by a designer in order to identify the optimal access structure for the multi-level system under consideration. However, to derive some intuition about how the costs impact the optimal multi-access structure, let us consider a setup with $L = 3$

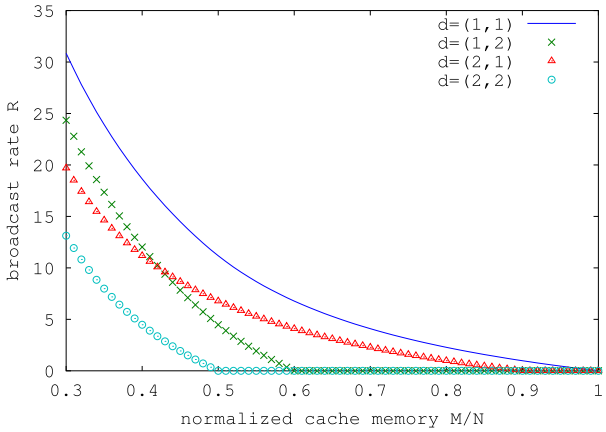


Figure 2.12: Achievable rate vs. cache memory in a two-level setup, for different access structures.

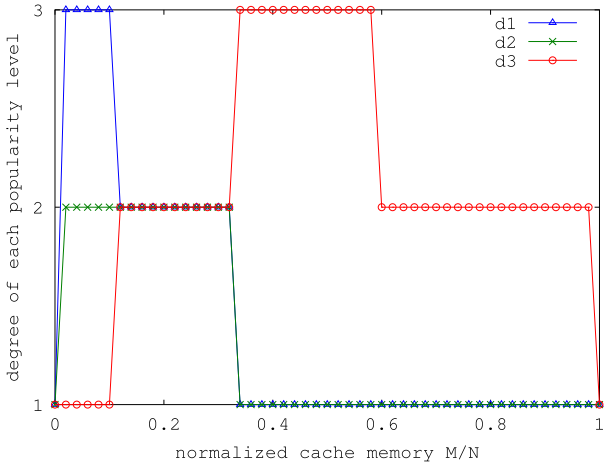


Figure 2.13: Optimal access structure vs. memory, with $d_{\max} = 3$, $d_{\text{avg}} = 2$.

levels, and with $N_1 = 0.04N$, $N_2 = 0.13N$, and $N_3 = 0.83N$ files in the three levels. Say we want to include both a maximum degree constraint $d_i \leq 3$ for each level i, as well as an average degree constraint $(\sum_i U_i d_i)/U \leq 2$. Then, Figure 2.13 plots the optimal access structure vs. the normalized memory size. As before, when the memory is small, the optimal access structure is one which satisfies $d_1 \geq d_2 \geq d_3$, but this relation becomes reversed as the memory increases.

2.8.3 Stochastic variations in user profiles

The theoretical setup and results presented in the previous sections assumed a symmetric and deterministic user profile across all the APs. In particular, exactly U_i users are assigned to each AP to request files from level i. This section aims at evaluating the robustness of memory-sharing

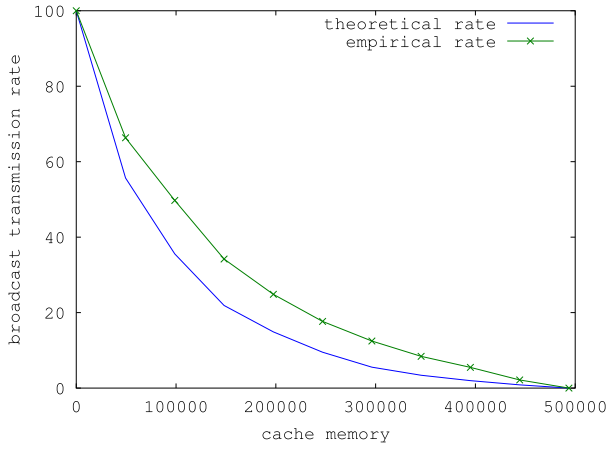


Figure 2.14: Comparison of the theoretical rate with the empirical rate, based on simulations of demands over the YouTube dataset, with 5 caches and 100 total users. The theoretical rate is off by a factor of up to 2.8 from the empirical.

to asymmetry and stochasticity in the user profiles across caches.

We consider a setup where each of the KU users in the system randomly connects to one of the K APs and requests a file stochastically, according to the YouTube popularity distribution in Figure 2.1. The scheme we use here is a simple variation of the one for the worst-case setup: the files are split into two levels, and the placement is done using memory-sharing based on their average popularity. The delivery phase is almost identical to the worst-case delivery, with the exception that, because of a lack of determinism in the user profile, not all caches will have the same number of users per level. This is handled by simply trying to group as many users of the same level as possible in each broadcast transmission.

We ran simulations for this setup using the above strategy, and we plot here the empirically achieved rate against the cache memory in Figure 2.14. For comparison, we also show the rate predicted by our theoretical model, which splits the files into two levels and assumes a symmetric user profile across the caches. Clearly, the theory very closely predicts the empirical results for a random user profile, thus demonstrating the robustness of our theoretical results to stochastic variations across APs and justifying their utility in practice.

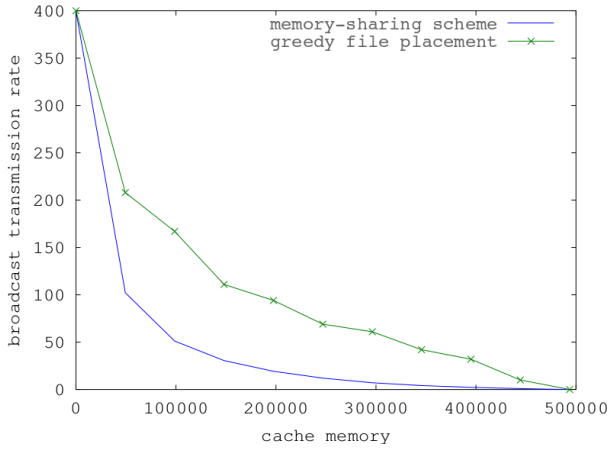


Figure 2.15: Comparison of the memory-sharing scheme with GFP, simulated over the YouTube dataset. The memory-sharing scheme achieves up to a factor-14.5 in gain over GFP.

2.8.4 Comparison with Greedy File Placement (GFP)

The Greedy File Placement (GFP) strategy is a natural strategy to use under a "continuous" popularity distribution such as the YouTube one. GFP essentially stores only the most popular files in every cache, as many as the cache can hold. For any memory size M, GFP fully stores the M most popular files, so that requests for more popular files are completely served from the cache, and requests for less popular files are fully handled by the BS transmission using multiple-unicast. The results, given in Figure 2.15, show the superiority of memory-sharing when simulating the two schemes with a stochastic-demands model based on the YouTube dataset.

2.8.5 Numerical gap

As discussed in Section 2.4, the multiplicative gap in Theorem 4 results from many generous approximations in bounding the achievable rate. Numerical results suggest that this gap is in fact much smaller. We here give a few examples of these results.

- If $K = 10$, $L = 3$, $(N_1, N_2, N_3) = (500, 1500, 8000)$, $(U_1, U_2, U_3) = (9, 5, 1)$, and $(d_1, d_2, d_3) = (1, 3, 5)$, then we get a gap of approximately 6.
- Increasing the number of caches and files: if $K = 20$, $L = 3$, $(N_1, N_2, N_3) = (200, 20\,000, 800\,000)$, $(U_1, U_2, U_3) = (10, 5, 1)$, and $(d_1, d_2, d_3) = (1, 1, 1)$, then we get a gap of approximately 6.8.

- The same setup as the previous point, with access degrees of $(d_1, d_2, d_3) = (1, 2, 3)$ gives a gap of about 7.6.

In most examples we have tried, the gap was in the range 5–10, regardless of the access degree. In the worst case, the largest gap our numerics have shown was about 45, in a situation with $L = 3$ and $D = 5$.

2.8.6 Extensions to the problem

There are several extensions and generalizations to the models considered in this chapter that might be future research directions to explore. As discussed in [42], there are perhaps several practical considerations that might be useful to incorporate into the models. Below are some such aspects of the problem, to name a few, that are worth exploring in the context of caching for multi-level popularity and access.

- Our work assumes that the broadcast transmission is made at the same rate to all users. However, different users will have different channel qualities with the base station, and thus the performance of common message broadcast will be limited by the weakest user. This aspect has been partially studied in [43, 44] where users can have channels of different qualities.

- Another generalization of the problem is to consider finite file sizes. This can significantly affect the caching strategies by limiting the number of multicasting opportunities, as explored in [30].

- Finally, an important part of caching in general is to estimate the dynamic popularity distributions. Thus one possible generalization to our problem is to find methods of classifying content into a small number of levels, dynamically over time, and regularly adapting the cache placement accordingly. A different approach to dynamic content placement is done in [24].

2.9 Practical Considerations and Implementation

The work in this chapter is of a theoretical nature, and many practical considerations must be addressed for a large-scale implementation of multi-level coded caching systems. Some such con-

siderations are: computational efficiency, content delay-sensitivity, communication overhead, and error correction. Some of these were addressed in [45], which studied a single-level caching system with one user. The paper proposed an efficient algorithm for this setup that can handle asynchronous user requests while both satisfying delivery time constraints for each user and preserving much of the global caching gain. The authors implemented this algorithm in a working prototype and showed that the gains were close to the simulated values.

In this section, we discuss a system that we have implemented in order to handle a multi-level and multi-user setup. We give a high-level explanation of the workings of the system, evaluate its performance, and discuss further challenges that it reveals to the implementation of such systems on a large scale.

2.9.1 System Description

The system consists of three separate Java programs: one for the server, one for the caches, and one for the users. The server program is run once, and an instance of the cache and user programs is run for each cache and user, respectively. The processes can be on different machines altogether, provided that each user knows the IP address of the cache it wants to connect to, and both the users and caches know the IP address of the server.

After some preliminary processing, the server is ready to accept connections from the caches. When a cache connects, it performs the (decentralized) placement phase, and then accepts file requests from users and forwards them to the server. The file requests can be completely asynchronous. If the files are video files, the users can start to stream them in real time even if the server has not completed the transfer.

The system works with a multi-user, multi-level popularity model of the files, with single access.

Subdivision of Files

The system handles each file by subdividing it into smaller parts. This subdivision operates on two levels. At the lowest level, a file is divided into small *blocks*. The system treats a block as the smallest indivisible unit of a file. The placement and delivery schemes operate on the level of a block. Each block consists of B bytes. At a higher level, a file is split into larger *chunks*. The

chunks of each file are treated as though they are separate files. A request for a particular file translates to sequential requests to each of its chunks, and each chunk must be delivered to its user in full before the next chunk is considered. Each chunk is made up of C blocks.

The reason for these divisions is as follows. First, the blocks are needed because operating directly on bits or bytes is practically inefficient. The placement and delivery phases operate on individual blocks, and keeping track of and identifying each block requires some amount of header/metadata. If the block size B is too small, the metadata becomes too large in comparison, effectively negating coding benefits. However, B cannot be too large, otherwise there would be fewer blocks in a chunk or file, which would reduce the number of coding opportunities and by extension the coding gains. In practice, it seems that a B of about one kilobyte is sufficient.

Splitting the files into chunks is done for two (related) reasons. The first reason is to handle delay-sensitive content such as video streaming. For a large video file, allowing coding between any blocks in the file means that a user might have to download almost the entire video before being able to play it. By chunking, this wait is limited to the chunk level. Indeed, chunking guarantees that the server will serve the first chunk of the requested file to the user before starting with the second chunk. We can hence reduce the user's wait time by reducing the chunk size. The downside is that a smaller chunk size means fewer coding opportunities and hence a smaller coding gain. There is hence a balance to be achieved when choosing the right chunk size C.

The second reason for chunking the files is that it allows the creation of coding opportunities across users in the presence of asynchronous demands, without having to wait for several users to connect before their requests are served. More precisely, it enables the synchronization of requests so that they can be served in the same coded delivery. To illustrate, suppose that every file is split into four chunks, and let us denote the ith chunk of file W_n by $W_{n,i}$. Suppose that the server was serving chunk $W_{1,3}$ to user one and chunk $W_{2,1}$ to user two. During this delivery period, if the server receives requests from, say, two new users (users three and four requesting W_3 and W_4 respectively), then once the current delivery is completed the server can serve

$$W_{1,4}, W_{2,2}, W_{3,1}, W_{4,1}$$

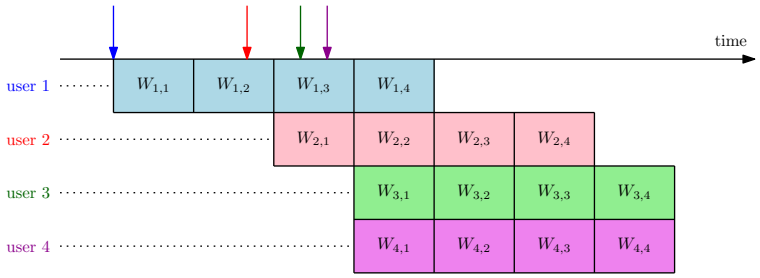


Figure 2.16: Progression of chunk serving as user requests come in over time. In the figure, time flows from left to right. The vertical top arrows indicate the arrival of requests from new users. Each rectangle represents the server serving a particular chunk to its corresponding user indicated on the left. Rectangles that share the same column are served at the same time using a coded delivery. Thus the asynchronous file requests were transformed into synchronous chunk requests.

simultaneously to all four users. This is illustrated in Figure 2.16.

2.9.2 System Evaluation

In this section, we are interested in showing a proof-of-concept implementation of the system. Our focus is mostly on the communication aspect: we seek to show that we can apply our memory-sharing strategy and obtain gains over simpler methods even when the system operates in a practical setting. In particular, we operate on a small number of users and on file sizes common for video files. During evaluation, for a fair comparison the files used are created by randomly generating bytes in order to have a fixed file size.

The file size considered here is 200 MB.[8] We choose a block size of $B = 1$ kB and a chunk size of $C = 40\,000$ blocks. Since 1080p videos have a bit rate of approximately 3 to 6 megabits per second [46], this corresponds to HD video files of about five minutes split into chunks of one minute each.

As in Section 2.5.4, we compare our Optimal Memory-Sharing (OMS) scheme with Greedy Memory-Sharing (GMS) and Uniform Memory-Sharing (UMS). We create $L = 2$ popularity levels, the second containing three times as many files as the first, $N_2 = 3N_1$.[9] We create $K = 10$ caches, and at each cache attach $U_1 = 3$ users for level one and $U_2 = 1$ user for level two. We set the cache memory to be enough to store the first level completely, i.e., $M = N_1$.

[8] In this section, we use the SI convention that the unit prefixes in kB and MB refer to powers of ten instead of powers of two, i.e., 1 MB = 1000 kB = 10^6 B.

[9] The exact number of files is not important in the worst case as long as there are more chunks than users. Since these evaluations were performed by running many caches and users on the same laptop, it was easier on the machine to only create one file for level one and three files for level two, and to have the server pretend as though no two file requests were the same. This gives the same performance as if there were many more files.

Theoretically, the gains of OMS over both GMS and UMS should scale as $\Theta(\sqrt{K})$ in this regime. In fact, this is the regime where the gains of OMS over the *better* of GMS and UMS is maximized. However, since the numbers involved were small, the gains were approximately 1.1 over GMS and 1.7 over UMS. Other memory regimes can give higher gains over each scheme individually.

While these numbers are not very large, they do show a proof-of-concept result. For larger systems, with a greater number of caches and more powerful servers, the benefits of OMS can be obtained over simpler methods such as GMS and UMS.

2.9.3 Challenges and Further Extensions

Currently, the system uses a naïve search for the best delivery message for any given set of simultaneous requests. This is computationally expensive and would not scale well. One improvement can be to implement a similar algorithm to [45] and adapt it to the multi-user, multi-level model.

Another issue that comes up is the reliable transmission of the broadcast/multicast messages. Since the goal of coded caching is to make use of broadcast, the server must transmit multicast messages to the intended users without resorting to (unicast) TCP protocols. One way to resolve this at the application layer is to apply a rateless error correction code on top of the UDP datagrams sent in the broadcast.

In addition, currently the IP address of each cache is explicitly provided to the users. Implementing this in a heterogeneous wireless network requires that the user discovers nearby caches and connects to one, or to several in case of multi-level access.

Finally, the cache contents should be dynamic in order to account for changes in popularity over time. The simplest way to do that would be for the caches to run the placement algorithm periodically, for instance every few hours. A similar problem has been studied for the single-user case in [24] (without a multi-level popularity model) for which a coded least-recently sent algorithm is proposed and analyzed.

Chapter 3

Adaptive Matching and Partial Adaptive Matching

3.1 Introduction

In this chapter, we expand on the coded caching problem initiated in [4, 5] mainly by considering the possibility of connecting to one of a number of caches. In [4] as well as many other works in the literature [10, 11, 12, 6, 13], a key assumption is that users are pre-fixed to specific caches; see also [47, 48] 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 [49, 50] 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 chapter, we consider a "partial adaptive matching" setup, i.e., a setup where users can be matched to any cache belonging

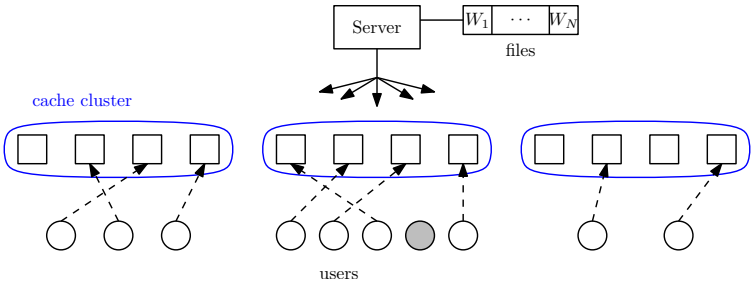


Figure 3.1: Illustration of the setup considered in this chapter. The squares represent $K = 12$ caches, divided into three clusters of size $d = 4$ caches each, and the circles represent users at these clusters. Dashed arrows represent the matching phase, and solid arrows the delivery phase. Unmatched users are in gray.

to a *subset* of caches, which we first studied in [51]. 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 Figure 3.1. 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 [4]; on the other extreme, if all caches belonged to a single cluster, then we get back the total adaptive matching setup from [49, 50].

In [51], 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 chapter, we focus on the more relevant case when the popularity obeys a power law, specifically a Zipf law [21]. 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 chapter is organized as follows. Section 3.2 precisely describes the problem setup. We present the main results in Section 3.3, which include the rates achieved by the schemes as well as statements of approximate optimality. Finally, Section 3.4 describes the hybrid scheme.

Detailed proofs are given in Appendix B.

3.2 Problem Setup

Consider the system depicted in Figure 3.1. 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 each, 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 $\mathbf{u}$ 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 chapter, we focus on the case where the numbers $u_n(c)$ are independent Poisson random variables with parameter $\rho d p_n$, 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 ρK.

For a given request profile $\mathbf{u}$, let $R_{\mathbf{u}}$ 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 $\bar{R} = \mathbb{E}_{\mathbf{u}}[R_{\mathbf{u}}]$. Specifically, we are interested in $\bar{R}^*$ defined as the smallest $\bar{R}$ 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

$$d \geq [2(1+t_0)/\alpha] \log K, \tag{3.1}$$

where $\alpha = -\log(2\rho e^{1-2\rho})$ 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 chapter 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.

3.3 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 $d = K$.[1] Therefore, there must be some transition in the suitable strategy as the cluster size d increases from one to K.

The goal of this chapter 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 [11, 12], whereas PAM will apply a matching scheme similar to [49, 50] 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.

[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 3.2, the Poisson model is not suitable in that case. However, the results from the literature are still very relevant to this chapter.

Regardless of the value β 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 Figure 3.2 and 3.4 and made precise in the theorems that follow.

While most of the analysis assumes general values for K, $N \geq K$, d (except for (3.1)), 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

$$\nu \geq 1; \quad \delta \in (0, 1]; \quad \mu \in [0, \nu].$$

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 $\beta > 1$.[2]

3.3.1 Shallow Zipf: $\beta \in [0, 1)$

In [51], we studied this problem when the files obeyed a uniform popularity, i.e., when $\beta = 0$. In this chapter, we show that the case $\beta \in [0, 1)$ is very similar to the uniform case. Indeed, the results from [51] 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 7. *When $\beta \in [0, 1)$, the PCD scheme can achieve for all M an expected rate of*

$$\bar{R}^{\mathrm{PCD}} = \min\left\{\rho K, \left[\tfrac{N}{M} - 1\right]^{+} + \tfrac{K^{-t_0}}{\sqrt{2\pi}}\right\}.$$

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

Theorem 7 can be proved by directly applying any suitable coded caching strategy [11, 12, 6] along with an arbitrary matching phase. The additional K^{-t_0} term represents the expected number of users that will not be matched to any cache and must hence be served directly from the server. The derivation of this term is done in Lemma 11 in Appendix B.1.

The next theorem gives the rate achieved by PAM.

Theorem 8. *When $\beta \in [0, 1)$, the PAM scheme can achieve an expected rate of*

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

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

Theorem 8 can be proved using a similar argument to [49]: 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 B.2.

Notice that PAM can achieve a rate of $o(1)$ when $dM > \Omega(N \log N)$. 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 $dM > (1 - o(1))N$. Consequently, the cache service constraint increases this memory threshold by at most a logarithmic factor.

The rates of PCD and PAM are illustrated in Figure 3.2 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 Figure 3.3.

Theorem 9. *When $\beta \in [0, 1)$, and considering only a polynomial scaling of the parameters with K, PCD outperforms PAM in the regime*

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

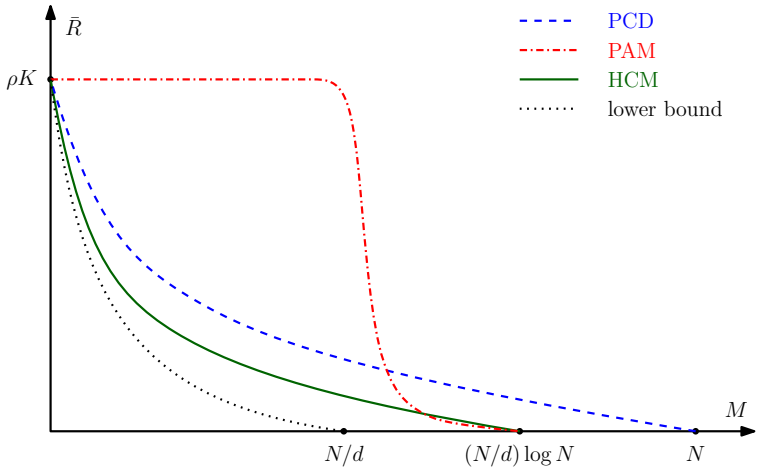


Figure 3.2: Rates achieved by PCD, PAM, and HCM when $\beta \in [0, 1)$, along with information-theoretic lower bounds. HCM is a hybrid scheme described in Section 3.4, and the lower bounds are presented in Appendix B.4. This plot is not numerically generated but is drawn approximately for illustration purposes.

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

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 9 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 3.4, 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 14, 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 $M > \Omega((N/d)\log K)$.

3.3.2 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 Figure 3.4.

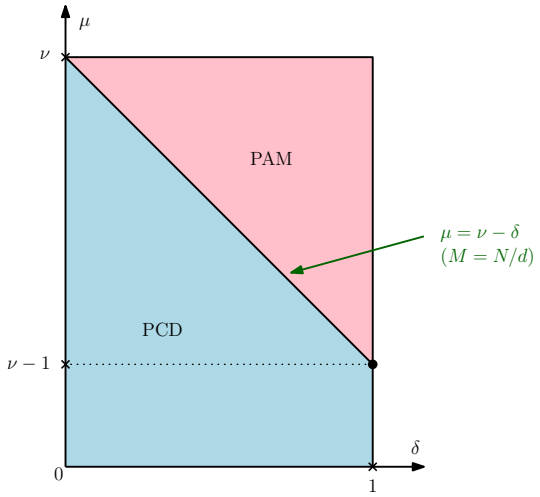


Figure 3.3: The scheme among PCD and PAM that performs better than the other when $\beta \in [0, 1)$, in terms of polynomial scaling in K. Here $N = K^{\nu}$, $d = K^{\delta}$, and $M = K^{\mu}$.

Theorem 10. *When $\beta > 1$, the PCD scheme can achieve an expected rate of*

$$\bar{R}^{\text{PCD}} = \begin{cases} K^{1/\beta} & \text{if } 0 \leq M < 1; \\ \left[\frac{(KM)^{1/\beta}}{M} - 1\right]^{+} + \frac{K^{-t_0}}{\sqrt{2\pi}} & \text{if } 1 \leq M < N^{\beta}/K; \\ \left[\frac{N}{M} - 1\right]^{+} + \frac{K^{-t_0}}{\sqrt{2\pi}} & \text{if } M \geq N^{\beta}/K. \end{cases}$$

Much like Theorem 7, Theorem 10 follows from directly applying the coded caching strategy from [11, 12]. Again, the K^{-t_0} term represents the expected number of unmatched users, derived in Lemma 11 in Appendix B.1.

Theorem 11. *When $\beta > 1$, the PAM scheme can achieve an expected rate of*

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

The proof of Theorem 11, given in Appendix B.3, follows along the same lines as [50] 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 more copies for the more popular files, and match the users accordingly.

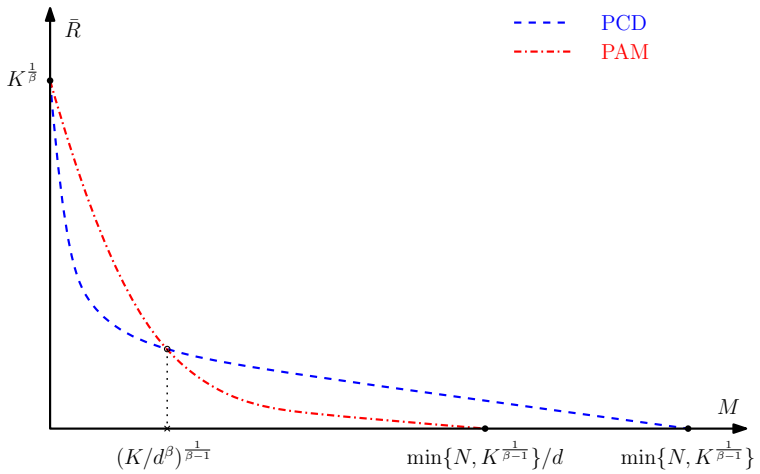


Figure 3.4: Rates achieved by PCD and PAM in the $\beta > 1$ case. Again, this plot is not numerically generated but is drawn approximately for illustration purposes.

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 10 and 11 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 Figure 3.5 and proved at the end of this subsection.

Theorem 12. *When $\beta > 1$, and considering only a polynomial scaling of the parameters with K, PCD outperforms PAM in the regime*

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

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

When comparing Theorems 9 and 12, 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 Figure 3.5 by the additional line segment joining points $(1-\nu(\beta-1), \nu\beta-1)$ and $(1/\beta, 0)$. As β 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 Figure 3.3 for $\beta \in [0, 1)$. In other words,

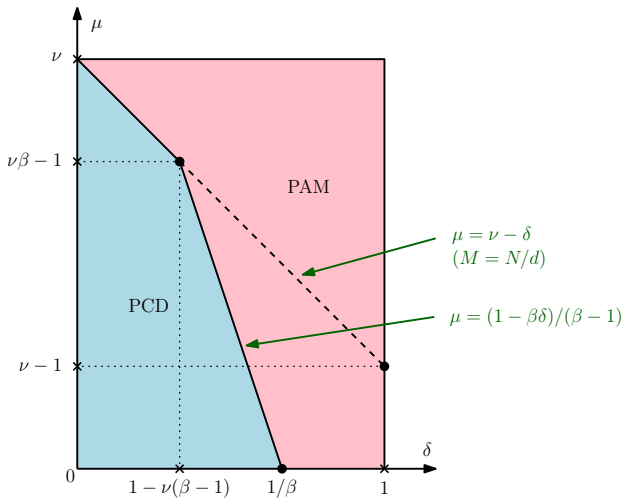


Figure 3.5: The scheme among PCD and PAM that performs better than the other when $\beta > 1$ and $\nu < 1/(\beta - 1)$, in terms of polynomial scaling in K. Here $N = K^{\nu}$, $d = K^{\delta}$, and $M = K^{\mu}$.

when $\beta > 1$ and as $\beta \to 1^{+}$, 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., Figure 3.3, at least under a poly-K analysis.

Proof of Theorem 12: Recall that we are only focusing on a poly-K analysis. We will define σ^{PCD} and σ^{PAM} to be the exponents of K in $\bar{R}^{\mathrm{PCD}}$ and $\bar{R}^{\mathrm{PAM}}$, respectively, i.e., $\bar{R}^{\mathrm{PCD}} = \Theta(K^{\sigma^{\mathrm{PCD}}})$ and similarly for PAM. Our goal is to compare σ^{PCD} to σ^{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 Figure 3.5.

The trivial case is when the total cluster memory dM is large, specifically $\mu + \delta > \min\{\nu, 1/(\beta - 1)\}$. From Theorem 11, the PAM rate is then $o(1)$, hence $\sigma^{\mathrm{PAM}} = 0$. 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

$$\sigma^{\mathrm{PCD}} = \min\left\{[1 - (\beta - 1)\mu]/\beta\,,\, \nu - \mu\right\};$$
$$\sigma^{\mathrm{PAM}} = \min\left\{1/\beta\,,\, 1 - (\beta - 1)(\delta + \mu)\right\}.$$

Notice that we always have $\sigma^{\mathrm{PCD}} \leq [1 - (\beta - 1)\mu]/\beta \leq 1/\beta$, and hence we only need to compare

σ^{PCD} to the second term in the minimization in σ^{PAM}. We split the analysis into a small and a large memory regimes, with the threshold $\mu \lessgtr \nu\beta - 1$.

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

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

3.3.3 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 $\bar{R}$ such that $\bar{R} \leq C \cdot \bar{R}^* + o(1)$, 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 8 that $\bar{R}^{\mathrm{PAM}} = o(1)$, and thus PAM is trivially approximately optimal. The following theorem states the approximate optimality of PCD when $M < O(N/d)$.

Theorem 13. *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,*

$$\frac{\bar{R}^{\mathrm{PCD}}}{\bar{R}^*} \leq C \triangleq \frac{96}{(1 - \beta)\rho(1 - e^{-1}/2)^2}.$$

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

Theorem 13 can be proved by first reducing the $\beta \in [0,1)$ case to a uniform-popularities setup, and then applying the converse results from [51]. Proof details are given in Appendix B.4.

When $\beta > 1$ we know from Theorem 11 that $\bar{R}^{\text{PAM}} = o(1)$ is achieved for $M > \Omega(\min\{N \log N, K^{\frac{1}{\beta-1}}\}/d)$, and thus PAM is trivially approximately optimal in that regime.

3.4 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 [51] 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 Figure 3.2 along with the rates of PCD and PAM for comparison.

Theorem 14. *For any $\beta \in [0,1)$, HCM can achieve a rate of*

$$\bar{R}^{\text{HCM}} = \begin{cases} \min\left\{\rho K, \frac{N}{M} - \chi + \frac{K^{-t}}{\sqrt{2\pi}}\right\} & \text{if } M \leq \lfloor N/\chi \rfloor; \\ \frac{K^{-t}}{\sqrt{2\pi}} & \text{if } M \geq \lceil N/\chi \rceil, \end{cases}$$

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

While the expression for $\bar{R}^{\text{HCM}}$ given in the theorem is rigorous, we can approximate it here for

clarity as

$$\bar{R}^{\mathrm{HCM}} \approx \min\left\{\rho K, \left[\tfrac{N}{M} - \Theta\left(\tfrac{d}{\log K}\right)\right]^{+} + o(1)\right\}.$$

The proof of Theorem 14 is given in detail in Appendix B.5, where we provide a rigorous explanation of the HCM scheme.

We will next compare HCM to PCD and PAM. Notice from Figure 3.2 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.

Chapter 4

Cache-Aided Gaussian Interference Networks

4.1 Introduction

Traditional communication networks focus on establishing a reliable connection between two fixed network nodes and are therefore connection centric. With the recent explosion in multimedia content, network usage has undergone a significant shift: users now want access to some specific content, regardless of its location in the network. Consequently, network architectures are shifting towards being content centric. These content-centric architectures make heavy use of in-network caching and, in doing so, redesign the protocol stack from the network layer upwards [52].

A natural question to ask is how the availability of in-network caches can be combined with the wireless physical layer and specifically with two fundamental properties of wireless communication: the broadcast and the superposition of transmitted signals. Recent work in the information theory literature has demonstrated that this combination can yield significant benefits. This information-theoretic approach to caching was introduced in the context of the noiseless broadcast channel in [4], where it was shown that significant performance gains can be obtained using cache memories at the *receivers*. In [53], the noiseless broadcast setting was extended to the interference channel, which is the simplest multiple-unicast wireless topology capturing both broadcast and superposition. The authors presented an achievable scheme showing performance gains using cache memories at the

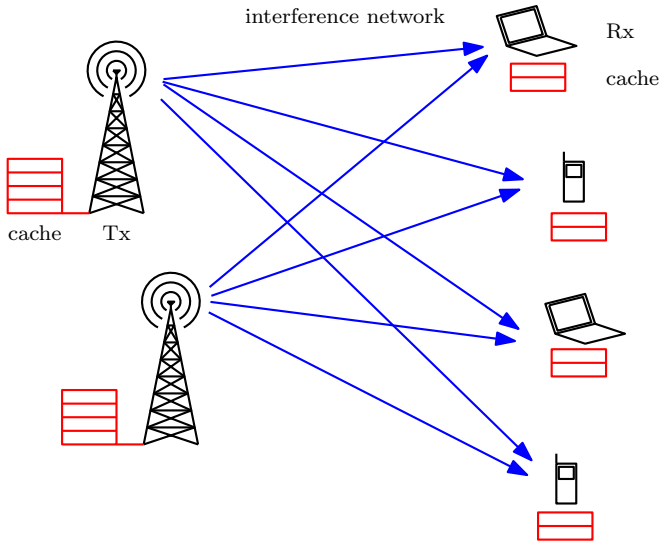


Figure 4.1: Caching in a wireless interference network. Caches (in red) are placed at all network nodes.

transmitters.

In this chapter, we continue the study of the cache-aided wireless interference network, but we allow for caches at both the *transmitters and receivers* as shown in Figure 4.1. The main result of this chapter (Theorem 15, Section 4.3) is a complete constant-factor approximation of the degrees of freedom (DoF) of this network. The result is general, in that it holds for any number of transmitters and receivers, size of content library, transmitter cache size (large enough to collectively hold the entire content library), and receiver cache size. Moreover, our converse holds for arbitrary caching and transmission functions, and imposes no restrictions as done in prior work.

Several architectural and design insights emerge from this degrees-of-freedom approximation.

1. Our achievable scheme introduces a novel separation of the physical and network layers, thus redesigning the protocol stack from the network layer downwards. From the order-wise matching converse, we hence see that this separation is approximately optimal.

2. Once the transmitter caches are large enough to collectively hold the entire content library, increasing the transmitter memory further can lead to at most a constant-factor improvement in the system's degrees of freedom. In particular, and perhaps surprisingly, this implies that transmit zero-forcing is not needed for approximately optimal performance.

3. There is a trade-off between the number of transmitters needed for (approximately) maximal

system performance and the amount of receiver cache memory. As the receiver memory increases, the required number of transmitters decreases, down to a constant when the memory is a constant fraction of the entire content library.

There are three seemingly natural network-layer abstractions for this problem. The first network-layer abstraction treats the physical layer as a standard interference channel and transforms it into non-interacting bit pipes between disjoint transmitter-receiver pairs. This approach is inefficient. The second network-layer abstraction treats the physical layer as an X-channel and transforms it into non-interacting bit pipes between each transmitter and each receiver. The third network-layer abstraction treats the physical layer as multiple broadcast channels: it creates a broadcast link from each transmitter to all receivers. The last two approaches turn out to be approximately optimal in special circumstances: the second when the receivers have no memory, and the third when they have enough memory to each store almost all the content library. In this chapter, we propose a network-layer abstraction that creates X-channel *multicast* bit pipes, each sent by a transmitter and intended for a subset of receivers whose size depends on the receiver memory. This abstraction generalizes the above two approaches, and we show that it is in fact order-optimal for all values of receiver memory.

Our solution to this problem requires solving a new communication problem at the physical layer that arises from the proposed separation architecture. This problem generalizes the X-channel setting studied in [54] by considering multiple *multicast* messages instead of just unicast. We focus on the symmetric case and provide a complete and exact DoF characterization of this symmetric multiple multicast X-channel problem, by proposing a strategy based on interference alignment and proving its optimality (see Theorem 16, Section 4.4).

Finally, while most of the chapter is focused on the high-SNR regime, we conclude the chapter with an analysis of the low-SNR regime and derive energy-efficiency gains of caching for the interference channel. We use a similar separation architecture and show that, contrary to the high-SNR regime, a large transmitter memory can have a significant impact by enabling a beamforming gain. We also use a separation architecture, with a different physical-layer scheme more suited to the low-SNR regime, and show that it is approximately optimal in two extreme cases: the case when there is a single transmitter (Gaussian broadcast channel) and the case when there is a single

receiver (Gaussian multiple-access channel).

The remainder of this chapter is organized as follows. Section 4.2 introduces the problem setting and establishes notation. Section 4.3 states the chapter's main results. Section 4.4 presents the separation architecture in detail; Section 4.5 gives the interference alignment strategy used at the physical layer. Section 4.6 proves the order-optimality of our strategy. Section 4.7 explores an interesting variant of the separation architecture. Section 4.8 discusses extensions to the problem as well as relation to some works in the literature. Section 4.9 analyzes the low-SNR regime and compares it with the high-SNR regime. We defer additional proofs to Appendix C.

Related Work

The information-theoretic framework for coded caching was introduced in [4] in the context of the deterministic broadcast channel. This has been extended to online caching systems [24], systems with delay-sensitive content [45], heterogeneous cache sizes [25], unequal file sizes [26], and improved converse arguments [27, 28]. Content caching and delivery in device-to-device networks, multi-server topologies, and heterogeneous wireless networks have been studied in [32, 34, 23, 13, 14]. This framework was also applied to hierarchical (tree) topologies in [31], and to non-uniform content popularities in [10, 11, 12, 35, 13, 14]. Other related work includes [36], which derives scaling laws for content replication in multihop wireless networks, and [37], which explores distributed caching in mobile networks using device-to-device communications. The benefit of coded caching when the caches are randomly distributed was studied in [38], and the benefits of adaptive content placement using knowledge of user requests were explored in [39].

More recently, this information-theoretic framework for coded caching has been extended in [53] to interference channels with caches at only the transmitters, focusing on three transmitters and three receivers. The setting was extended in [55] to arbitrary numbers of transmitters and receivers and included a rate-limited fronthaul. Interference channels with caches *both* at transmitters and at receivers were considered in [56, 57, 8], all of which have a setup similar to the one in this chapter. However, each of these three works has some restrictions on the setup. The authors in [56] focus on one-shot linear schemes, while [57] prohibits inter-file coding during placement and limits the number of receivers to three. Our prior work [8] studies the same setup but with only

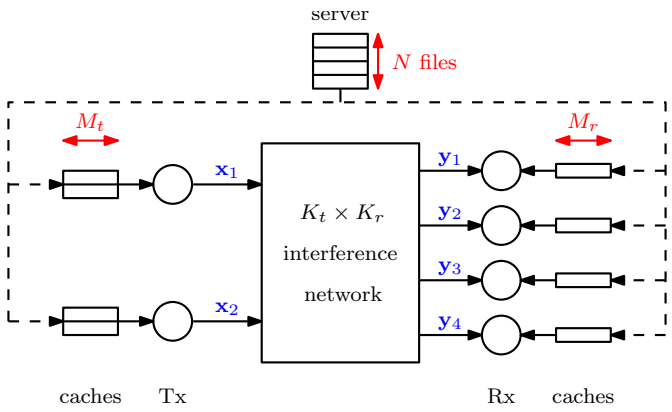


Figure 4.2: The caching problem, with $K_t = 2$ transmitters and $K_r = 4$ receivers. The server holds a content library of N files. Information about these files is placed in the transmitter caches of size M_t and in the receiver caches of size M_r during a placement phase (indicated by dashed lines). During the subequent delivery phase (indicated by solid lines), each user requests one file, and all the requested files have to be delivered over the interference network.

two transmitters and two receivers. The work in this chapter differs from those above in that it considers an arbitrary number of transmitters and receivers and proves order-optimality using outer bounds that assume *no restrictions* on the scheme.

4.2 Problem Setting

A content library contains N files $W_1, \ldots, W_N$ of size F bits each. A total of K_r users will each request one of these files, which must be transmitted across a $K_t \times K_r$ time-varying Gaussian interference channel whose receivers are the system's users. We will hence use the terms "receiver" and "user" interchangeably. Our goal is to reliably transmit these files to the users with the help of caches at both the transmitters and the receivers.

Example 1. *The setup is depicted in Figure 4.2 for the case with $K_t = 2$ transmitters, $K_r = 4$ receivers, and $N = 4$ files in the content library. We will use this setting as a running example throughout the paper.*

The system operates in two phases, a *placement phase* and a *delivery phase*. In the placement phase, the transmitter and receiver caches are filled as an arbitrary function of the content library. The transmitter caches are able to store $M_t F$ bits; the receiver caches are able to store $M_r F$ bits. We refer to M_t and M_r as the transmitter and receiver cache sizes, respectively. Other than the

memory constraints, we impose no restrictions on the caching functions (in particular, we allow the caches to arbitrarily code across files). In this chapter, we consider all values of $M_r \geq 0$, but we restrict ourselves to the case where the transmitter caches can collectively store the entire content library,[1] i.e.,

$$M_t \geq N/K_t. \tag{4.1}$$

The delivery phase takes place after the placement phase is completed. In the beginning of the delivery phase, each user requests one of the N files. We denote by $\mathbf{u} = (u_1, \ldots, u_{K_r})$ the vector of user demands, such that user i requests file W_{u_i}. These requests are communicated to the transmitters, and each transmitter j responds by sending a codeword $\mathbf{x}_j = (x_j(1), \ldots, x_j(T))$ of block length T into the interference channel. We impose a power constraint over every channel input $\mathbf{x}_j$,

$$\frac{1}{T} \|\mathbf{x}_j\|^2 \leq \mathsf{SNR}, \quad \forall j = 1, \ldots, K_t.$$

Note that each transmitter only has access to its own cache, so that $\mathbf{x}_j$ only depends on the contents of transmitter j's cache and the user requests $\mathbf{u}$. We impose no other constraint on the channel coding function (in particular, we explicitly allow for coding across time using potentially nonlinear schemes).

Receiver i observes a noisy linear combination of all the transmitted codewords,

$$y_i(\tau) \triangleq \sum_{j=1}^{K_t} h_{ij}(\tau) x_j(\tau) + z_i(\tau),$$

for all time instants $\tau = 1, \ldots, T$, where the $z_i(\tau)$'s are independent identically distributed (iid) unit-variance additive Gaussian noise, and $h_{ij}(\tau)$ are independent time-varying random channel coefficients obeying some continuous probability distribution. We can rewrite the channel outputs in vector form as

$$\mathbf{y}_i = \sum_{j=1}^{K_t} \mathbf{H}_{ij} \mathbf{x}_j + \mathbf{z}_i, \tag{4.2}$$

where $\mathbf{H}_{ij}$ is a diagonal matrix representing the channel coefficients over the block length T.

[1] To achieve any positive DoF, the minimum requirement is that $K_t M_t + M_r \geq N$, i.e., that all the transmitter caches and any single receiver cache can collectively store the entire content library. We impose the slightly stronger requirement $K_t M_t \geq N$ since it is the regime of most practical interest, and since it simplifies the analysis.

For fixed values of M_t, M_r, and SNR, we say that a transmission rate $R = F/T$ is achievable if there exists a coding scheme such that all the users can decode their requested files with vanishing error probability. More formally, R is *achievable for demand vector* $\mathbf{u}$ if

$$\max_{i\in\{1,\ldots,K_r\}} \Pr\big(\hat{W}_i \neq W_{u_i}\big) \to 0 \text{ as } T \to \infty,$$

where $\hat{W}_i$ indicates the reconstruction of file W_{u_i} by user i. Note that R is fixed as T, and hence F, go to infinity. We say R is *achievable* if it is achievable for all demand vectors $\mathbf{u}$.

We define the optimal transmission rate $R^\star(\mathsf{SNR})$ as the supremum of all achievable rates for a given SNR (and number of files, cache sizes, and number of transmitters/receivers). In the remainder of this chapter we will focus on the degrees of freedom (DoF) defined as

$$\mathsf{DoF} \triangleq \lim_{\mathsf{SNR}\to\infty} \frac{R^\star(\mathsf{SNR})}{\frac{1}{2}\log \mathsf{SNR}}. \tag{4.3}$$

While the DoF is useful for presenting and interpreting the main results in the next section, we will also often work with its reciprocal $1/\mathsf{DoF}$ because it is a convex function of (M_t, M_r).

4.3 Main Results

The main result of this chapter is a complete constant-factor approximation of the DoF for the cache-aided wireless interference network. In order to state the result, we define the function $d(N, K_t, K_r, M_t, M_r)$ through

$$\frac{1}{d(N, K_t, K_r, M_t, M_r)} \triangleq \frac{K_t - 1 + \min\left\{\frac{K_r}{\kappa+1}, N\right\}}{K_t} \cdot \left(1 - \frac{\kappa}{K_r}\right), \tag{4.4}$$

for any N, K_t, K_r, M_t, and $M_r = \kappa N/K_r$ with $\kappa \in \{0, 1, \ldots, K_r\}$, and the lower convex envelope of these points for all other $M_r \in [0, N]$.

Theorem 15. *The degrees of freedom* DoF *of the* $K_t \times K_r$ *cache-aided interference network with*

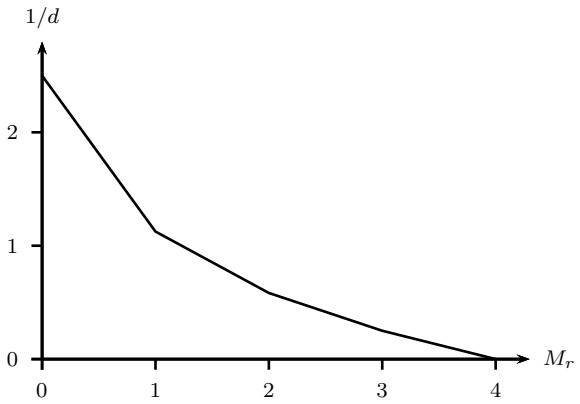


Figure 4.3: Approximate reciprocal DoF of the 2×4 cache-aided interference network with 4 files, introduced in Example 1, as a function of receiver cache size M_r, for any $M_t \geq N/K_t$.

N files, transmitter cache size $M_t \in [N/K_t, N]$, and receiver cache size $M_r \in [0, N]$ satisfies

$$d(N, K_t, K_r, M_t, M_r) \leq \mathsf{DoF} \leq 13.5 \cdot d(N, K_t, K_r, M_t, M_r).$$

The approximate (reciprocal) DoF is illustrated in Figure 4.3 for the setup in Example 1.

In terms of the rate $R^\star(\mathsf{SNR})$ of the system, Theorem 15 can be interpreted using (4.3) as

$$d(\cdot) \cdot \frac{1}{2} \log \mathsf{SNR} - o\left(\log \mathsf{SNR}\right) \leq R^\star(\mathsf{SNR}) \leq 13.5 \cdot d(\cdot) \cdot \frac{1}{2} \log \mathsf{SNR} + o\left(\log \mathsf{SNR}\right),$$

when SNR grows, where we have used $d(\cdot)$ instead of $d(N, K_t, K_r, M_t, M_r)$ for simplicity.

The constant 13.5 in Theorem 15 is the result of some loosening of inequalities in order to simplify the analysis. We numerically observe that the multiplicative gap does not exceed 4.16 for $N, K_t, K_r \leq 100$.

The coding scheme achieving the lower bound on DoF in Theorem 15 uses separate network and physical layers. The two layers interface using a set of multicast messages from each transmitter to many subsets of receivers. At the physical layer, an interference alignment scheme (generalizing the scheme from [54]) delivers these messages across the interference channel with vanishing error probability and at optimal degrees of freedom. At the network layer, a caching and delivery strategy generalizing the one in [4] is used to deliver the requested content to the users, utilizing the non-interacting error-free multicast bit pipes created by the physical layer. The matching upper

bound in Theorem 15 shows that this separation approach is without loss of order optimality. This separation architecture is described in more detail in Section 4.4.

In order to better understand the behavior of the system, we decompose the approximation of the sum degrees of freedom $K_r\mathsf{DoF}$ provided by Theorem 15 into three components, or gains.[2] These are: an interference alignment (IA) gain g^{IA}, a local caching gain g^{LC}, and a global caching gain g^{GC}, forming

$$K_r\mathsf{DoF} \approx K_r d(N, K_t, K_r, M_t, M_r) \overset{(a)}{\approx} \underbrace{\frac{K_t K_r}{K_t + K_r - 1}}_{g^{\mathrm{IA}}} \cdot \underbrace{\frac{1}{1 - \frac{M_r}{N}}}_{g^{\mathrm{LC}}} \cdot \underbrace{\frac{K_r M_r/N + 1}{\frac{M_r}{N}\left(\frac{1}{K_r} + \frac{1}{K_t - 1}\right)^{-1} + 1}}_{g^{\mathrm{GC}}}. \tag{4.5}$$

Note that (a) holds with exact equality when $K_r M_r/N$ is an integer. We point out that, for ease of presentation, this decomposition is written for the case when the first term achieves the minimum in (4.4), i.e., $K_r/(\kappa + 1) \leq N$. This includes the most relevant case when the content library N is larger than the number of receivers K_r. In fact, we focus on this case in most of the chapter, particularly regarding the achievability and some of the intuition. A detailed discussion of the case $K_r/(\kappa + 1) > N$, including a decomposition similar to (4.5), is given in Appendix C.1.

The term g^{IA} is the degrees of freedom achieved by communication using interference alignment and is the same as in the unicast X-channel problem [54]. It is the only gain present when the receiver cache size is zero. In other words, it is the baseline degrees of freedom without caching (see for example Figure 4.3 when $M_r = 0$).

When the receiver cache size is non-zero, we get two improvements, in analogy to the two gains described in the broadcast caching setup in [4]. The local caching gain reflects that each user already has some information about the requested file locally in its cache. Hence, g^{LC} is a function of M_r/N, the fraction of each file stored in a single receiver cache. On the other hand, the global gain derives from the coding opportunities created by storing different content at different users, and from the multicast links created to serve coded information useful to many users at once. This gain depends on the total amount of receiver memory, as is reflected by the $K_r M_r/N$ term in the numerator of g^{GC}.

[2]Note that this decomposition arises from our interpretation of our approximately optimal strategy described in Section 4.4.

It is interesting to see how each of these gains scales with the various system parameters K_t, K_r, M_t/N, and M_r/N. In order to separate the different gains, we work with the logarithm

$$\log K_r \mathsf{DoF} \approx \log g^{\mathrm{IA}} + \log g^{\mathrm{LC}} + \log g^{\mathrm{GC}}$$

of the sum degrees of freedom. By varying the different parameters, we can plot how both the sum DoF and its individual components evolve.

Scaling with transmitter memory M_t

Notice in Theorem 15 that the DoF approximation does not involve the transmitter memory M_t. Thus, once $M_t = N/K_t$, just enough to store the entire content library between all transmitters, any increase in the transmit memory will only lead to at most a constant-factor improvement in the DoF.

The strategy used to achieve the lower bound in Theorem 15 (see Section 4.4 for details) stores uncoded *nonoverlapping* file parts in each transmit cache. This is done regardless of the transmitter memory M_t and the receiver memory M_r. Since this is an order-optimal strategy, we conclude that the transmitters do not need to have any shared information. Consequently, and perhaps surprisingly, transmit zero-forcing is not needed for order-optimality and cannot provide more than a constant-factor DoF gain. Moreover, given that the value 13.5 of the constant gap is close to and was obtained using similar arguments to the value of 12 derived in [4] for the error-free broadcast case, we conjecture that most of the improvements on the constant would not come from sharing information among transmitters or from any transmit zero-forcing, but rather from tighter converse arguments.

Scaling with receiver memory M_r

Figure 4.4 depicts the decomposition of the approximate sum degrees of freedom $K_r d \approx K_r \mathsf{DoF}$ as a function of the receiver cache size M_r. As expected, the interference alignment gain g^{IA} does not depend on the receiver cache size and is hence constant. The local caching gain g^{LC} increases slowly with M_r and becomes relevant whenever each receiver can cache a significant fraction of the

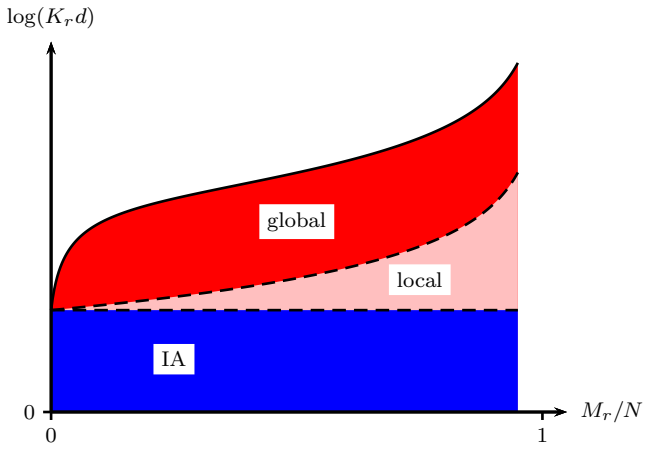


Figure 4.4: DoF gains as a function of receiver cache size characterized by M_r/N.

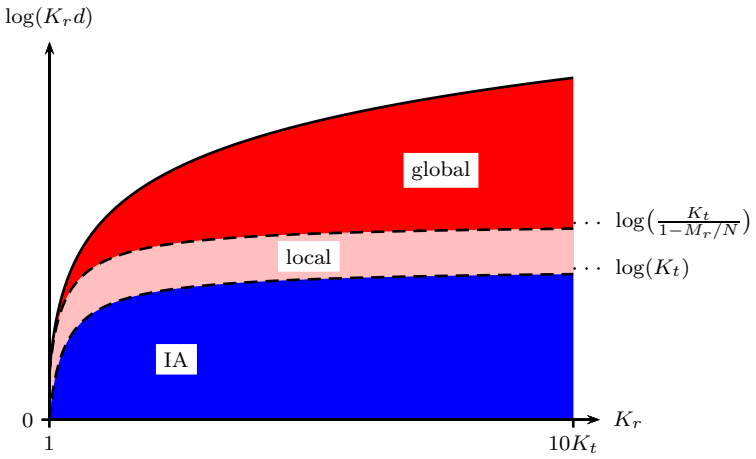


Figure 4.5: DoF gains as a function of number of receivers K_r.

content library, say $M_r/N \geq 0.5$. The global caching gain g^{GC} increases much more quickly and is relevant whenever the cumulative receiver cache size is large, say $K_r M_r/N \geq 1$.

Scaling with number of receivers K_r

Figure 4.5 depicts the decomposition of the approximate sum degrees of freedom $K_r d \approx K_r \mathsf{DoF}$ as a function of the number of receivers K_r. The local caching gain g^{LC} is not a function of K_r and is hence constant as expected. In the limit as $K_r \to \infty$, the interference alignment gain g^{IA} converges to K_t. The global caching gain g^{GC}, on the other hand, behaves as

$$g^{\mathrm{GC}} \approx \frac{K_r M_r/N + 1}{(K_t - 1)M_r/N + 1}$$

for large K_r. In particular, unlike the other two gains, the global gain does not converge to a limit and scales linearly with the number of receivers. Thus, for systems with larger number of receivers, the global caching gain becomes dominant.

Scaling with number of transmitters K_t

As the number of receivers K_r or the receive memory M_r increase, the sum DoF grows arbitrarily large. The same is not true as the number of transmitters K_t increases. In fact, as $K_t \to \infty$, we find that $g^{\mathrm{IA}} \to K_r$, $g^{\mathrm{GC}} \to 1$, and the sum DoF converges to

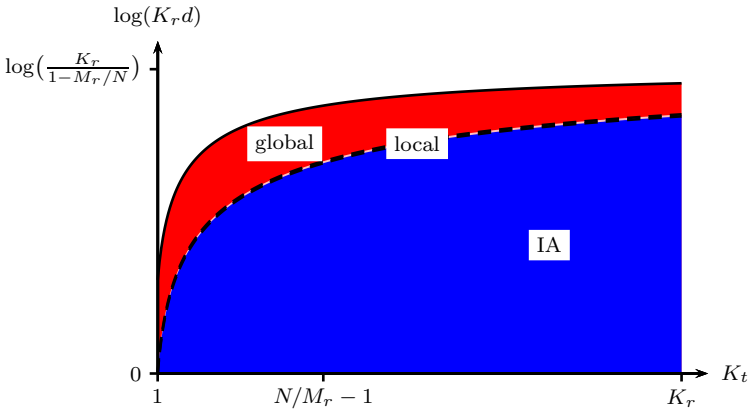
$$\lim_{K_t \to \infty} K_r \mathsf{DoF} \approx \frac{K_r}{1 - M_r/N}. \tag{4.6}$$

This is not surprising, since, with a large number of transmitters, interference alignment effectively creates K_r orthogonal links from each transmitter to the receivers, each of DoF approaching 1. With the absence of multicast due to these orthogonal links, the global caching gain vanishes and the only caching gain left is the local one.

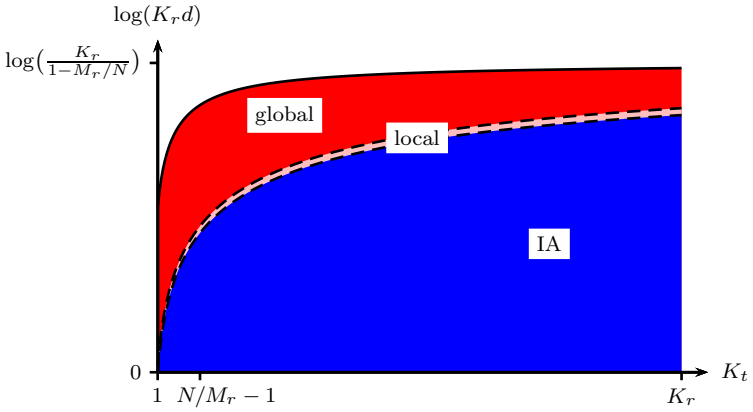
An interesting question then is how large K_t has to be for the DoF to approach the limit in (4.6). Specifically, for what values of K_t does the sum DoF become $\Theta\big(K_r/(1-M_r/N)\big)$? When the receiver cache memory is small, specifically $M_r < N/K_r$, the number of transmitters K_t must be of the order of K_r (see Figure 4.6a). However, as M_r increases, we find that a smaller number of transmitters is needed to achieve the same DoF (see Figs. 4.6b and 4.6c). In general, the limiting value is reached (within a constant) when $K_t = \Omega(N/M_r - 1)$.[3] In particular, if the receiver caches can store a constant fraction of the content library, then we only need a constant number of transmitters to achieve maximal benefits, up to a multiplicative constant. There is thus a trade-off between the number of transmitters K_t and the amount of receiver cache memory M_r required for maximal system performance (up to the local caching gain): the larger the receiver memory, the fewer the required transmitters.

While the separation architecture discussed above (on which we focus in most of this chapter) is

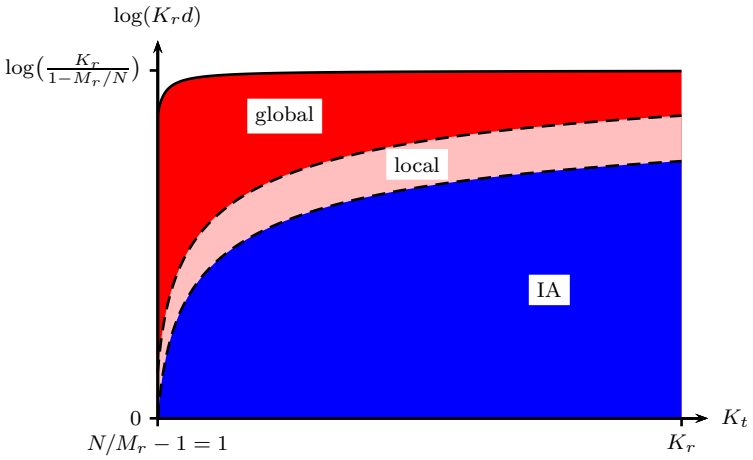
[3]This comes from being able to write $K_r\mathsf{DoF} \approx \frac{K_t(\kappa+1)}{K_t(\kappa+1)+K_r-\kappa-1} \cdot \frac{K_r}{1-M_r/N}$, where $\kappa = K_r M_r/N$. The first factor is a constant when $K_t(\kappa+1) = \Omega(K_r - \kappa - 1)$, which leads to $K_t = \Omega(\frac{K_r}{K_r M_r/N+1} - 1)$. When K_r is large, this behavior becomes $K_t = \Omega(\frac{N}{M_r} - 1)$.



(a) Small M_r/N.

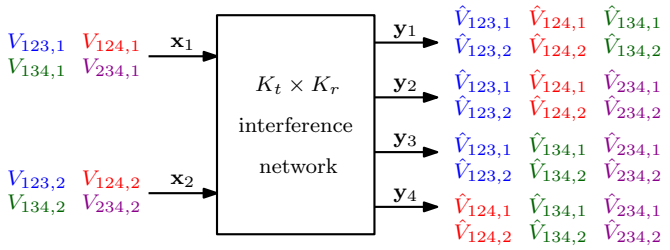


(b) Moderate M_r/N.

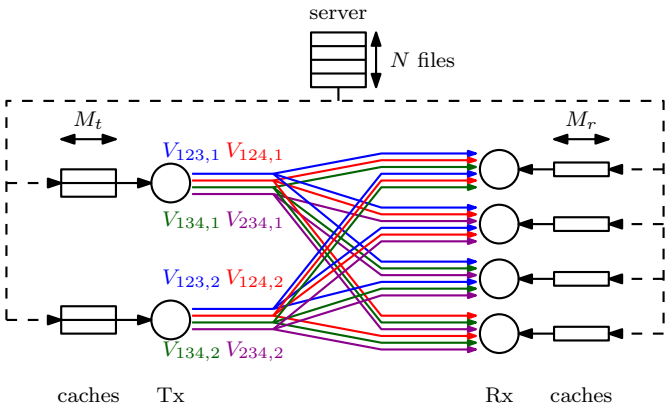


(c) Large M_r/N.

Figure 4.6: DoF gains as a function of number of transmitters K_t for various regimes of receiver cache size (characterized by M_r/N).



(a) Physical-layer view.



(b) Network-layer view.

Figure 4.7: The separation architecture applied to the setup in Figure 4.2 (i.e., Example 1) with multicast size $\kappa + 1 = 3$. The interface messages $V_{\mathcal{S}j}$ at the physical layer can be abstracted as orthogonal error-free multicast bit pipes at the network layer. Thus at the physical layer (a) we focus on transmitting the $V_{\mathcal{S}j}$'s across the interference channel, while at the network layer (b) we perform the caching and delivery strategies, oblivious of the underlying physical channel, to deliver the requested files.

order optimal, one can still make some strict improvements, albeit no more than a constant factor, by choosing a different separation architecture. In Section 4.7, we present an alternative separation architecture for the case $K_t = K_r = N = 2$ that creates *interacting* error-free bit pipes as the physical-layer abstraction. This architecture can achieve a strictly higher DoF than Theorem 15 in some regimes.

4.4 Separation Architecture

Our proposed separation architecture isolates the channel coding aspect of the problem from its content delivery aspect. The former is handled at the physical layer, while the latter is handled at the network layer. The two layers interface using a set $\mathscr{V}$ of multiple multicast messages,

$$\mathscr{V} = \{V_{\mathcal{S}j} : j \in \{1, \ldots, K_t\}, \mathcal{S} \in \mathscr{S}\}, \tag{4.7}$$

where $V_{\mathcal{S}j}$ denotes the message sent from transmitter j to the subset $\mathcal{S}$ of receivers, and $\mathscr{S} \subseteq 2^{\{1,\ldots,K_r\}}$ is some collection of subsets of receivers. Notice that all transmitters have messages for the same subsets of receivers, a natural design choice due to the symmetry of the problem. The physical layer processing transmits these messages across the interference network, while the network layer treats them as orthogonal error-free multicast bit pipes. Figure 4.7 illustrates this separation for the setting in Example 1.

In order to motivate our choice of $\mathscr{S}$ (and hence of $\mathscr{V}$), it will be useful to give a brief overview of the strategy used for the broadcast setup in [4]. Suppose that the receiver memory is $M_r = \kappa N/K_r$, where $\kappa \in \{0, 1, \ldots, K_r - 1\}$ is an integer. The idea is to place content in the receiver caches such that every subset of κ of them shares an exclusive part of every file (each file is thus split into $\binom{K_r}{\kappa}$ equal parts). During the delivery phase, linear combinations of these file parts are sent to every subset of $\kappa + 1$ users such that each user can combine its received linear combination with the contents of its cache to decode one part of their requested file. As a result, a total of

$$L^{\mathrm{BC}}(N, K_r, M_r) \cdot F = \frac{K_r - \kappa}{\kappa + 1} \cdot F \tag{4.8}$$

bits are sent through the network (see [4, Theorem 1]).

Notice that the broadcast strategy never really sends any *broadcast* message on a logical level (except when $\kappa + 1 = K_r$). Instead, it sends many *multicast* messages, each intended for $\kappa + 1$ users, which just happen to be "overheard" by the unintended receivers. Inspired by this, we choose the messages in $\mathscr{V}$ to reflect the multicast structure in [4]. Specifically, we choose to create one multicast message from each transmitter to every subset of receivers of size $\kappa + 1$. In other words,

$$\mathscr{S} = \{\mathcal{S} \subseteq \{1, \ldots, K_r\} : |\mathcal{S}| = \kappa + 1\}. \tag{4.9}$$

For example, Figure 4.7 shows the separation architecture when $\kappa + 1 = 3$. While (4.9) depicts the choice of $\mathscr{S}$ that we make most of the time, it is inefficient in a particular regime, namely when both the number of files and the receiver memory are small. Since that regime is of only limited interest, we relegate its description to Appendix C.1.

Let $\tilde{R}_{\kappa+1}$ be the rate at which we transmit these messages at the physical layer, i.e., $V_{\mathcal{S}j} \in$

$[2^{\tilde{R}_{\kappa+1}T}]$. Further, let $\ell_{\kappa+1}$ be the size (normalized by file size) of whatever is sent through each multicast link at the network layer, i.e., $V_{\mathcal{S}j} \in [2^{\ell_{\kappa+1}F}]$. Therefore, $\tilde{R}_{\kappa+1}T = \ell_{\kappa+1}F$. Let us write $\tilde{R}^{\star}_{\kappa+1}$ and $\ell^{\star}_{\kappa+1}$ to denote the optimal $\tilde{R}_{\kappa+1}$ and $\ell_{\kappa+1}$, respectively, within their respective subproblems (these will be defined rigorously in the subsections below). These quantities can be connected to the rate R of the original caching problem. Indeed, since $F = RT$, then we can achieve a rate R equal to

$$R = \frac{\tilde{R}^{\star}_{\kappa+1}}{\ell^{\star}_{\kappa+1}}, \tag{4.10}$$

when $M_r = \kappa N/K_r$, $\kappa \in \{0, 1, \ldots, K_r - 1\}$.[4]

The separation architecture has thus created two subproblems of the original problem. At the physical layer, we have a pure communication subproblem, where multicast messages $V_{\mathcal{S}j}$ must be transmitted reliably across an interference network. At the network layer, we have a caching subproblem with noiseless orthogonal multicast links connecting transmitters to receivers. In the two subsections below, we properly formulate each subproblem. We give a strategy for each as well as the values of $\tilde{R}_{\kappa+1}$ and $\ell_{\kappa+1}$ that they achieve.

4.4.1 Physical Layer

At the physical layer, we consider only the communication problem of transmitting specific messages across the interference channel described in Section 4.2, as illustrated in Figure 4.7a. This is an interesting communication problem on its own, and we hence formulate it without all the caching details. The message set that we consider is one where every transmitter j has a message for every subset $\mathcal{S}$ of σ receivers, where $\sigma \in \{1, \ldots, K_r\}$ is given.[5] We label such a message as $V_{\mathcal{S}j}$, and we note that there are a total of $K_t\binom{K_r}{\sigma}$ of them. For instance, in the example shown in Figure 4.7a, message $V_{134,2}$ (used as a shorthand for $V_{\{1,3,4\},2}$) is sent by transmitter 2 to receivers 1, 3, and 4. We call this problem the multiple multicast X-channel with multicast size σ, as it generalizes the (unicast) X-channel studied in [54] to multicast messages. Note that, when $\sigma = 1$, we recover the

[4] The nature of the separation architecture implies that κ must always be an integer. Regimes where it is not are handled using time and memory sharing between points where it is. Furthermore, we exclude the case $\kappa = K_r$ (equivalently, $M_r = N$) for mathematical convenience, but we can in fact trivially achieve an infinite rate when $M_r = N$ by storing the complete content library in every user's cache.

[5] In the context of the caching problem, σ is chosen to be $\kappa + 1$, as described earlier.

unicast X-channel.

We assume a symmetric setup, where all the messages have the same rate $\tilde{R}_\sigma$, i.e., $V_{\mathcal{S}j} \in [2^{\tilde{R}_\sigma T}]$. A rate is called achievable if a strategy exists allowing all receivers to recover all their intended messages with vanishing error probability as the block length T increases. Our goal is to find the largest achievable rate $\tilde{R}_\sigma$ for a given SNR, denoted by $\tilde{R}^\star_\sigma(\mathsf{SNR})$, and in particular its DoF

$$\tilde{d}^\star_\sigma(K_t, K_r) \triangleq \lim_{\mathsf{SNR}\to\infty} \frac{\tilde{R}^\star_\sigma(\mathsf{SNR})}{\frac{1}{2}\log \mathsf{SNR}}.$$

One of the contributions of this chapter is an exact characterization of $\tilde{d}^\star_\sigma$, and we next give an overview of how to achieve it.

For every receiver i, there is a set of $K_t\binom{K_r-1}{\sigma-1}$ desired messages $\{V_{\mathcal{S}j} : i \in \mathcal{S}\}$, and a set of $K_t\binom{K_r-1}{\sigma}$ interfering messages $\{V_{\mathcal{S}j} : i \notin \mathcal{S}\}$. Using TDMA, all $K_t\binom{K_r}{\sigma}$ messages can be delivered to their receivers at a sum DoF of 1, i.e., $\tilde{d}_\sigma = 1/K_t\binom{K_r}{\sigma}$. However, by applying an interference alignment technique that generalizes the one used in [54], we can, loosely speaking, collapse the $K_t\binom{K_r-1}{\sigma}$ interfering messages at every receiver into a subspace of dimension $\binom{K_r-1}{\sigma}$ (assuming for simplicity that each message forms a subspace of dimension one), while still allowing reliable recovery of all $K_t\binom{K_r-1}{\sigma-1}$ desired messages. Thus an overall vector space of dimension $K_t\binom{K_r-1}{\sigma-1} + \binom{K_r-1}{\sigma} < K_t\binom{K_r}{\sigma}$ is used to deliver all $K_t\binom{K_r}{\sigma}$ messages. This strategy achieves a DoF-optimal rate, as asserted by the following theorem.

Theorem 16. *The DoF of the symmetric multiple multicast X-channel with multicast size σ is given by*

$$\tilde{d}^\star_\sigma(K_t, K_r) = \frac{1}{K_t\binom{K_r-1}{\sigma-1} + \binom{K_r-1}{\sigma}}.$$

The details of the interference alignment strategy are given in Section 4.5. The proof of optimality is left for Appendix C.4, since it does not directly contribute to our main result in Theorem 15. It does however reinforce it by providing a complete solution to the physical-layer communication subproblem.

The DoF shown in Theorem 16 is a per-message DoF. Since there are a total of $K_t\binom{K_r}{\sigma}$ messages,

we obtain a sum DoF of

$$K_t\binom{K_r}{\sigma}\cdot\tilde{d}^{\star}_{\sigma}(K_t,K_r)=\frac{K_t\binom{K_r}{\sigma}}{K_t\binom{K_r-1}{\sigma-1}+\binom{K_r-1}{\sigma}}=\frac{K_tK_r}{(K_t-1)\sigma+K_r}. \tag{4.11}$$

When $\sigma=1$, the sum DoF in (4.11) is $K_tK_r/(K_t+K_r-1)$, thus recovering the unicast X-channel result from [54]. When $\sigma=K_r$, the problem reduces to a broadcast channel with multiple sources, giving a sum DoF of 1.

4.4.2 Network Layer

The network layer setup is similar to the end-to-end setup, with the difference that the interference network is replaced by the multicast links $V_{\mathcal{S}j}$ from transmitters to receivers, as illustrated in Figure 4.7b. As mentioned previously, each link $V_{\mathcal{S}j}$ is shared by exactly $|\mathcal{S}|=\kappa+1$ users, where $\kappa=K_rM_r/N$ is an integer. We again focus on a symmetric setup, where all links have the same size $\ell_{\kappa+1}$, called the *link load*. It will be easier in the discussion to use the *sum network load* $L_{\kappa+1}$, i.e., the combined load of all $K_t\binom{K_r}{\kappa+1}$ links,

$$L_{\kappa+1}=K_t\binom{K_r}{\kappa+1}\cdot\ell_{\kappa+1}. \tag{4.12}$$

A sum network load L is said to be achievable if, for every large enough file size F, a strategy exists allowing all users to recover their requested files with high probability while transmitting no more than LF bits through the network. Our goal is to find the smallest achievable network load for every N, K_t, K_r, M_t, and M_r, denoted by

$$L^{\star}_{\kappa+1}(N,K_t,K_r,M_t,M_r),$$

where $\kappa=K_rM_r/N$ is an integer. Using a similar strategy to [4], we achieve the following sum network load.

Lemma 6. *In the network layer setup with a multicast size of $\kappa+1$, $\kappa\in\{0,1,\ldots,K_r-1\}$, a sum network load of*

$$L^{\star}_{\kappa+1}(N,K_t,K_r,M_t,M_r)\leq\frac{K_r-\kappa}{\kappa+1}$$

can be achieved when $M_r = \kappa N/K_r$.

Proof: We first divide every file W_n into K_t equal parts, $W_n = (W_n^1, \ldots, W_n^{K_t})$, and store the j-th part W_n^j in the cache of transmitter j. Note that, while we allow $M_t \geq N/K_t$ as per the regularity condition in (4.1), the above transmitter placement only stores exactly N/K_t files at every transmitter irrespective of the value of M_t. The different transmitters are then treated as independent sublibraries. Indeed, the receiver placement splits each receiver cache into K_t equal sections, and each section is dedicated to one sublibrary. A placement phase identical to [4] is then performed for each sublibrary in its dedicated receiver memory.

During the delivery phase, user i's request for a single file W_{u_i} is converted into K_t separate requests for the subfiles $(W_{u_i}^1, \ldots, W_{u_i}^{K_t})$, each from its corresponding sublibrary (transmitter). For every subset $\mathcal{S}$ of $\kappa + 1$ receivers, each transmitter j then sends through the link $V_{\mathcal{S}j}$ exactly what would be sent to these receivers in the broadcast setup, had the other transmitters not existed. This is possible since the $V_{\mathcal{S}j}$ links were chosen by design to match the multicast transmissions in the broadcast setup. Each transmitter will thus send $(1/K_t) \cdot L^{\mathrm{BC}}(N, K_r, M_r)$ files through the network (with L^{BC} as defined in (4.8)), for a total network load of $(K_r - \kappa)/(\kappa + 1)$. ■

4.4.3 Achievable End-to-End DoF

From (4.10) and using (4.12), we can achieve an end-to-end DoF of

$$\mathsf{DoF} \geq \frac{\tilde{d}^{\star}_{\kappa+1}(K_t, K_r)}{L^{\star}_{\kappa+1}(N, K_t, K_r, M_t, M_r)} \cdot K_t \binom{K_r}{\kappa+1}.$$

By combining Theorem 16 (with $\sigma = \kappa + 1$) and Lemma 6,

$$\begin{aligned}\frac{1}{\mathsf{DoF}} &\leq \frac{L^{\star}_{\kappa+1}(N, K_t, K_r, M_t, M_r)}{\tilde{d}^{\star}_{\kappa+1}(K_t, K_r)} \cdot \frac{1}{K_t \binom{K_r}{\kappa+1}} \\ &\leq \frac{K_r - \kappa}{\kappa + 1} \cdot \left[K_t \binom{K_r - 1}{\kappa} + \binom{K_r - 1}{\kappa + 1} \right] \cdot \frac{1}{K_t \binom{K_r}{\kappa+1}}.\end{aligned}$$

By writing

$$\begin{aligned} K_t\binom{K_r-1}{\kappa}+\binom{K_r-1}{\kappa+1} &= (K_t-1)\binom{K_r-1}{\kappa}+\binom{K_r-1}{\kappa}+\binom{K_r-1}{\kappa+1} \\ &\overset{(a)}{=} (K_t-1)\frac{\kappa+1}{K_r}\binom{K_r}{\kappa+1}+\binom{K_r}{\kappa+1} \\ &= \frac{(K_t-1)(\kappa+1)+K_r}{K_r}\cdot\binom{K_r}{\kappa+1}, \end{aligned}$$

where (a) is due to Pascal's triangle, we conclude that

$$\frac{1}{\mathsf{DoF}} \leq \frac{K_r-\kappa}{\kappa+1}\cdot\frac{(K_t-1)(\kappa+1)+K_r}{K_tK_r} = \frac{K_t-1+\frac{K_r}{\kappa+1}}{K_t}\cdot\left(1-\frac{\kappa}{K_r}\right). \tag{4.13}$$

This proves the achievability direction of Theorem 15 when $K_r/(\kappa+1) \leq N$. The case $K_r/(\kappa+1) > N$ is discussed in Appendix C.1.

4.5 The Multiple Multicast X-Channel

The multiple multicast X-channel problem (with multicast size σ) that emerges from our separation strategy is a generalization of the unicast ($\sigma = 1$) X-channel studied in [54]. We propose an interference alignment strategy that generalizes the one in [54]. In this section, we give a high-level overview of the alignment strategy in order to focus on the intuition. The rigorous explanation of the strategy is given in Appendix C.2 as a proof of Lemma 7, which is presented at the end of this section.

Consider communicating across the interference network over T time slots. Every transmitter j beamforms each message $V_{\mathcal{S}j}$ along some fixed vector of length T and sends the sum of the vectors corresponding to all its messages as its codeword. Each message thus occupies a subspace of dimension 1 of the overall T-dimensional vector space. The goal is to align at each receiver the interfering messages into the smallest possible subspace, so that a high rate is achieved for the desired messages.

When choosing which messages to align, we enforce the following three principles, which ensure maximal alignment without preventing decodability of the intended messages. At every receiver i:

1. Each desired message $V_{\mathcal{S}j}$ with $i \in \mathcal{S}$ must be in a subspace of dimension 1, not aligned with any other subspace.

2. Messages from the same transmitter must never be aligned.

3. All messages intended for the same subset $\mathcal{S}$ of receivers with $i \notin \mathcal{S}$ must be aligned into one subspace of dimension 1.

Principle 1 ensures that receiver i can decode all of its desired messages. To understand principle 2, notice that messages from the same transmitter go through the same channels. Therefore, if two messages from the same transmitter are aligned at one receiver, then they were also aligned during transmission, and are hence aligned at all other receivers, including their intended ones. Thus principle 2 ensures decodability at other receivers. As for principle 3, it provides the maximal alignment of the interfering messages without violating principle 2. Indeed, each aligned subspace contains K_t messages, one from each transmitter. Any additional message that is aligned would share a transmitter with one of them.

For every receiver, there are $K_t\binom{K_r-1}{\sigma-1}$ desired messages. By principle 1, each should take up one non-aligned subspace of dimension 1, for a total of $K_t\binom{K_r-1}{\sigma-1}$ dimensions. On the other hand, there are $K_t\binom{K_r-1}{\sigma}$ interfering messages. By principle 3, every K_t of them are aligned in one subspace of dimension 1, and hence all interfering messages fall in a subspace of dimension $\binom{K_r-1}{\sigma}$. These subspaces can be made non-aligned by ensuring that the overall vector space has a dimension of

$$T = K_t\binom{K_r-1}{\sigma-1} + \binom{K_r-1}{\sigma}.$$

Since each message took up one dimension, we get a per-message DoF of

$$\frac{1}{T} = \frac{1}{K_t\binom{K_r-1}{\sigma-1} + \binom{K_r-1}{\sigma}}.$$

This is an improvement over TDMA, which achieves a DoF of $1/K_t\binom{K_r}{\sigma}$.

In most cases, we do not achieve the exact DoF shown in Theorem 16 using a finite number of channel realizations. We instead achieve an arbitrarily close DoF by using an increasing number of channel realizations. The exact achieved DoF is given in the following lemma.

Lemma 7. *Let* $\Gamma = (K_r - \sigma)(K_t - 1)$. *For any arbitrary* $n \in \mathbb{N}$, *we can achieve a DoF for message* $V_{\mathcal{S}j}$ *equal to*

$$\delta_j^{(n)} = \frac{(n + c_j)^{\Gamma}}{\binom{K_r-1}{\sigma-1}\left[(n+1)^{\Gamma} + (K_t - 1)n^{\Gamma}\right] + \binom{K_r-1}{\sigma}(n+1)^{\Gamma}},$$

where $c_1 = 1$ *and* $c_2 = \cdots = c_{K_t} = 0$.

The proof of Lemma 7 is given in Appendix C.2.

Note that Lemma 7 achieves a slightly different DoF for $V_{\mathcal{S}j}$ depending on j, which might seem to contradict the symmetry in the problem setting. However, for a large n, we have $(n+1)^{\Gamma} \approx n^{\Gamma}$, and hence

$$\lim_{n\to\infty} \delta_j^{(n)} = \frac{1}{K_t\binom{K_r-1}{\sigma-1} + \binom{K_r-1}{\sigma}} = \tilde{d}_{\sigma}^{\star}(K_t, K_r)$$

for all j. Thus the symmetric DoF $\tilde{d}_{\sigma}^{\star}(K_t, K_r)$ is achieved in the limit.

4.6 Order-Optimality of the Separation Architecture

In this section, we give a high-level proof of the converse part of Theorem 15 by showing that the DoF achieved by the separation architecture in Section 4.4 is order-optimal. We do this by computing cut-set-based information-theoretic upper bounds on the DoF (equivalently, they are lower bounds on the reciprocal $1/\mathsf{DoF}$). These bounds are given in the following lemma, whose proof is placed at the end of this section in order not to distract from the intuition behind the converse arguments. The rigorous converse proof is given in Appendix C.3.

Lemma 8. *For any* N, K_t, K_r, $M_t \in [0, N]$, *and* $M_r \in [0, N]$, *the optimal DoF must satisfy*

$$\frac{1}{\mathsf{DoF}} \geq \max_{s \in \{1,\ldots,\min\{K_r, N\}\}} \frac{s\left(1 - \frac{M_r}{\lfloor N/s \rfloor}\right)}{\min\{s, K_t\}}.$$

Lemma 8 is next used to prove the converse part of Theorem 15, i.e.,

$$\mathsf{DoF} \leq 13.5 \cdot d(N, K_t, K_r, M_t, M_r),$$

where $d(\cdot)$ is defined in (4.4). The procedure is similar to the one used in [4]: we consider three

main regimes (Regimes 1, 2, and 3) of receiver memory M_r and in each compare the expression $d(\cdot)$ with the outer bounds. In addition, we consider a separate corner case (Regime 0) in which the largest possible number of distinct file requests (i.e., $\min\{K_r, N\}$) is small compared to the number of transmitters.

$$\text{Regime 0:} \quad \min\{K_r, N\} \leq 12.5K_t; \tag{4.14a}$$

$$\text{Regime 1:} \quad \min\{K_r, N\} > 12.5K_t \quad \text{and} \quad 0 \leq M_r \leq 1.1 \max\left\{1, \frac{N}{K_r}\right\}; \tag{4.14b}$$

$$\text{Regime 2:} \quad \min\{K_r, N\} > 12.5K_t \quad \text{and} \quad 1.1 \max\left\{1, \frac{N}{K_r}\right\} < M_r \leq 0.092\frac{N}{K_t}; \tag{4.14c}$$

$$\text{Regime 3:} \quad \min\{K_r, N\} > 12.5K_t \quad \text{and} \quad 0.092\frac{N}{K_t} < M_r \leq N. \tag{4.14d}$$

Note that Regimes 1, 2, and 3 are unambiguous, since

$$\min\{K_r, N\} > 12.5K_t \implies 0 < 1.1 \max\left\{1, \frac{N}{K_r}\right\} < 0.092\frac{N}{K_t} < N. \tag{4.15}$$

Since M_r is the only variable that we will consistently vary, we will abuse notation for convenience and write $d(M_r)$ instead of $d(N, K_t, K_r, M_t, M_r)$ for all $M_r \in [0, N]$. Our goal is thus to prove

$$\frac{1}{\mathsf{DoF}} \geq \frac{d^{-1}(M_r)}{13.5}. \tag{4.16}$$

For ease of reference, we will rewrite the expression of $d^{-1}(M_r)$ here. For $M_r = \kappa N/K_r$ where $\kappa \in \{0, 1, \ldots, K_r\}$ is an integer,

$$d^{-1}(\kappa N/K_r) = \frac{K_t - 1 + \min\left\{\frac{K_r}{\kappa+1}, N\right\}}{K_t} \cdot \left(1 - \frac{\kappa}{K_r}\right), \tag{4.17}$$

and $d^{-1}(M_r)$ is the lower convex envelope of these points for all $M_r \in [0, N]$. Note that $d^{-1}(M_r)$ is non-increasing and convex in M_r.

Regimes 0 and 3

Interestingly, Regimes 0 and 3 behave quite similarly to each other. Indeed, notice that in both of them we have $K_t = \Omega\left(\min\left\{K_r/(\kappa+1), N\right\}\right)$. Using (4.17), this implies

$$d^{-1}(M_r) \approx \frac{K_t + \min\left\{\frac{K_r}{\kappa+1}, N\right\}}{K_t} \cdot \left(1 - \frac{M_r}{N}\right) = \Theta\left(1 - \frac{M_r}{N}\right).$$

Conversely, we can apply Lemma 8 with $s = 1$ to get $1/\mathsf{DoF} \geq 1 - M_r/N$. Thus in both regimes the local caching gain is the only significant contribution to the DoF.

Regime 1

In Regime 1, the receiver memory is too small to have any significant effect. Therefore, using $12.5K_t < \min\{K_r, N\}$, we can write (4.17) as

$$d^{-1}(M_r) \approx \frac{K_t + \min\left\{\frac{K_r}{\kappa+1}, N\right\}}{K_t} \cdot 1 \leq \left(\frac{1}{12.5} + 1\right) \cdot \frac{\min\{K_r, N\}}{K_t}.$$

Conversely, by using Lemma 8 with $s \approx \min\{K_r, N\}$, we get

$$\frac{1}{\mathsf{DoF}} \gtrsim \frac{s \cdot 1}{K_t} \approx \frac{\min\{K_r, N\}}{K_t}.$$

Therefore, in this regime $\mathsf{DoF} \approx K_t/\min\{K_r, N\}$. We can explain this in terms of the DoF gains in (4.5): when the receiver memory is very small, the only relevant gain is the interference alignment gain.

Regime 2

In Regime 2, the receivers combined can store all of the content library. As a result, the global caching gain kicks in. We can upper-bound $d^{-1}(M_r)$ in (4.17) as follows:

$$d^{-1}(M_r) \leq 1 + \frac{K_r}{K_t(K_r M_r/N + 1)} \leq 1 + \frac{N}{K_t M_r} \leq 1.092\frac{N}{K_t M_r},$$

because $K_t < 0.092N/M_r$ in Regime 2. Conversely, let us apply Lemma 8 using $s \approx N/2M_r$:

$$\frac{1}{\mathsf{DoF}} \gtrsim \frac{s - s^2 M_r/N}{K_t} \approx \frac{N}{4K_t M_r}.$$

Therefore, $\mathsf{DoF} \approx K_t M_r/N$. This behavior is similar to what one would expect in the broadcast setup in [4], with the exception of the additional K_t factor.

Since $d^{-1}(M_r)$ approximately matches the outer bounds in all four regimes and can also be achieved as in Section 4.4, then it provides an approximate characterization of $1/\mathsf{DoF}$. The above arguments are made rigorous in Appendix C.3.

Proof of Lemma 8: Consider $s \in \{1, \ldots, \min\{K_r, N\}\}$ users. We shall look at $E = \lfloor N/s \rfloor$ different request vectors, such that the combined number of files requested by all users after E request instances is $\tilde{N} = sE = s\lfloor N/s \rfloor$ files. More specifically, we consider the request vectors $\mathbf{u}_1, \ldots, \mathbf{u}_E$ with

$$\mathbf{u}_e = \big(\underbrace{(e-1)s+1, (e-1)s+2, \ldots, es}_{s}, \underbrace{1, \ldots, 1}_{K_r - s}\big),$$

for each $e = 1, \ldots, E$. Note that we only focus on the first s users; the remaining $K_r - s$ users are not relevant to our argument.

When the request vector is $\mathbf{u}$, let $\mathbf{x}_j^{\mathbf{u}}$ and $\mathbf{y}_i^{\mathbf{u}}$ denote the inputs and outputs of the interference network for all transmitters j and receivers i. For notational convenience, we write $\mathbf{y}_{[s]}^{\mathbf{u}} = (\mathbf{y}_1^{\mathbf{u}}, \ldots, \mathbf{y}_s^{\mathbf{u}})$ and use a similar notation for $\mathbf{x}_{[K_t]}^{\mathbf{u}}$. Also, let Q_i denote the contents of user i's cache (recall that the cache contents are independent of $\mathbf{u}$). By Fano's inequality,

$$H\left(W_1, \ldots, W_{\tilde{N}} \middle| Q_1, \ldots, Q_s, \mathbf{y}_{[s]}^{\mathbf{u}_1}, \ldots, \mathbf{y}_{[s]}^{\mathbf{u}_E}\right) \leq \varepsilon T, \tag{4.18}$$

since the s users should be able to each decode their $\lfloor N/s \rfloor$ requested files using their caches and

channel outputs. Then,

$$\begin{aligned}
\tilde{N}RT &= H\left(W_1,\ldots,W_{\tilde{N}}\right) \\
&= I\left(W_1,\ldots,W_{\tilde{N}};Q_1,\ldots,Q_s,\mathbf{y}_{[s]}^{\mathbf{u}_1},\ldots,\mathbf{y}_{[s]}^{\mathbf{u}_E}\right) + H\left(W_1,\ldots,W_{\tilde{N}}\middle|Q_1,\ldots,Q_s,\mathbf{y}_{[s]}^{\mathbf{u}_1},\ldots,\mathbf{y}_{[s]}^{\mathbf{u}_E}\right) \\
&\overset{(a)}{\leq} I\left(W_1,\ldots,W_{\tilde{N}};Q_1,\ldots,Q_s,\mathbf{y}_{[s]}^{\mathbf{u}_1},\ldots,\mathbf{y}_{[s]}^{\mathbf{u}_E}\right) + \varepsilon T \\
&= I\left(W_1,\ldots,W_{\tilde{N}};\mathbf{y}_{[s]}^{\mathbf{u}_1},\ldots,\mathbf{y}_{[s]}^{\mathbf{u}_E}\right) + I\left(W_1,\ldots,W_{\tilde{N}};Q_1,\ldots,Q_s\middle|\mathbf{y}_{[s]}^{\mathbf{u}_1},\ldots,\mathbf{y}_{[s]}^{\mathbf{u}_E}\right) + \varepsilon T \\
&\leq I\left(W_1,\ldots,W_{\tilde{N}};\mathbf{y}_{[s]}^{\mathbf{u}_1},\ldots,\mathbf{y}_{[s]}^{\mathbf{u}_E}\right) + H\left(Q_1,\ldots,Q_s\right) + \varepsilon T \\
&\overset{(b)}{\leq} I\left(\mathbf{x}_{[K_t]}^{\mathbf{u}_1},\ldots,\mathbf{x}_{[K_t]}^{\mathbf{u}_E};\mathbf{y}_{[s]}^{\mathbf{u}_1},\ldots,\mathbf{y}_{[s]}^{\mathbf{u}_E}\right) + sM_rRT + \varepsilon T \\
&\overset{(c)}{\leq} E\cdot\max_{e\in\{1,\ldots,E\}} I\left(\mathbf{x}_1^{\mathbf{u}_e},\ldots,\mathbf{x}_{K_t}^{\mathbf{u}_e};\mathbf{y}_1^{\mathbf{u}_e},\ldots,\mathbf{y}_s^{\mathbf{u}_e}\right) + sM_rRT + \varepsilon T \\
&\overset{(d)}{\leq} E\cdot T\left(\min\left\{K_t,s\right\}\cdot\frac{1}{2}\log\mathsf{SNR} + o(\log\mathsf{SNR})\right) + sM_rRT + \varepsilon T \\
&= \lfloor N/s\rfloor\, T\cdot\left(\min\left\{K_t,s\right\}\cdot\frac{1}{2}\log\mathsf{SNR} + o(\log\mathsf{SNR})\right) + sM_rRT + \varepsilon T,
\end{aligned}$$

where (a) is due to inequality (4.18), (b) uses the data processing inequality, (c) follows from the independence of the channel outputs when conditioned on all channel inputs, and (d) is the capacity bound of the $K_t \times s$ MIMO channel over T time blocks.

Since $\tilde{N} = s\lfloor N/s\rfloor$, and by taking $T\to\infty$ and $\varepsilon\to 0$, we obtain

$$R\left(1-\frac{M_r}{\lfloor N/s\rfloor}\right) \leq \frac{1}{s}\cdot\min\{K_t,s\}\cdot\frac{1}{2}\log\mathsf{SNR} + o(\log\mathsf{SNR}).$$

The DoF thus obeys

$$\mathsf{DoF} \leq \frac{\min\{K_t,s\}}{s\left(1-\frac{M_r}{\lfloor N/s\rfloor}\right)}.$$

Since s was arbitrary, the above is true for any $s\in\{1,\ldots,\min\{K_r,N\}\}$, and thus the lemma is proved. ■

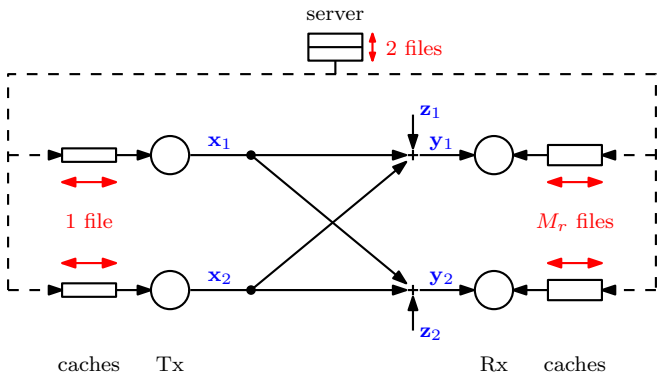


Figure 4.8: The 2×2 cache-aided interference channel with 2 files. The transmitter caches can hold exactly one file each, and the receiver caches $M_r \in [0, 2]$ files each. The $\mathbf{z}_i$'s are iid additive Gaussian unit-variance noise.

4.7 An Alternative Separation Strategy

In this chapter, we have determined the approximate DoF of the general cache-aided interference network. To do so, we have proposed a separation-based strategy that uses interference alignment to create *non-interacting* multicast bit pipes from transmitters to receivers, and we have shown that this strategy achieves a DoF that is within a constant multiplicative factor from the optimum. However, this achieved DoF is only approximately optimal. In fact, many improvements can be made, such as using transmit zero-forcing as has been discussed in previous work [53, 56, 57].

In this section, we explore a different approach, which lies within the context of interference alignment described in Section 4.5: rather than ignoring the interference subspace, which contains the aligned messages, we attempt to extract some information from it. Thus every receiver gains additional information in the form of an alignment of the bit pipes available at other receivers: the bit pipes would thus *interact*. We study this approach in a specific setup: the 2×2 interference channel with a content library of two files, shown in Figure 4.8.

For this 2×2 setup, by Theorem 15 the main strategy described in this chapter achieves

$$\frac{1}{\mathsf{DoF}} \leq \max\left\{\frac{3}{2} - M_r, 1 - \frac{1}{2}M_r\right\},$$

for $M_r \in [0, 2]$, as shown by the solid line in Figure 4.9. However, the same figure shows an improved inverse DoF, depicted by the dashed line, which is achieved using the interference-extracting scheme

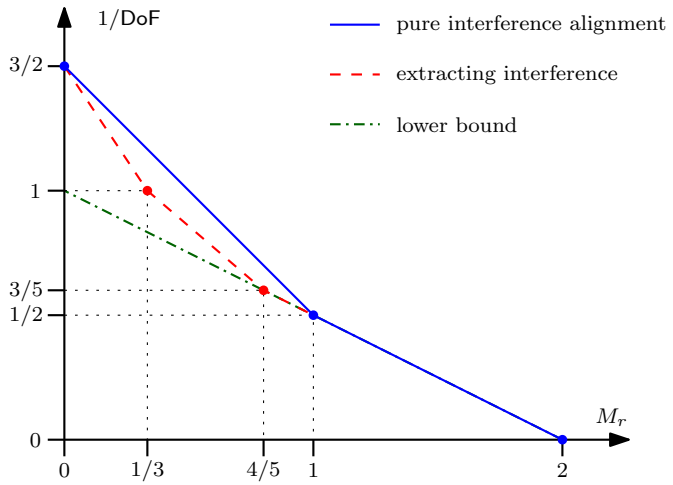


Figure 4.9: Inverse DoF achieved by the scheme from Section 4.5 (solid line), and the improved inverse DoF achieved by extracting more information from the aligned interference (dashed line). The dash-dotted line shows the information-theoretic lower bounds from Lemma 8.

discussed in this section. A factor-7/6 improvement is obtained over the main strategy. This result is summarized in the following theorem.

Theorem 17. *The following inverse DoF can be achieved for the* 2×2 *cache-aided interference network with* $N = 2$ *files and transmitter memory* $M_t = 1$*:*

$$\frac{1}{\mathsf{DoF}} \leq \max\left\{\frac{3}{2} - \frac{3}{2}M_r, \frac{9}{7} - \frac{6}{7}M_r, 1 - \frac{1}{2}M_r\right\},$$

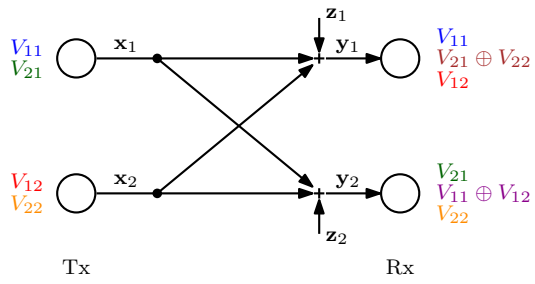
for all values of $M_r \in [0, 2]$*.*

It should be noted that the general converse stated in Lemma 8 can be applied here and results in

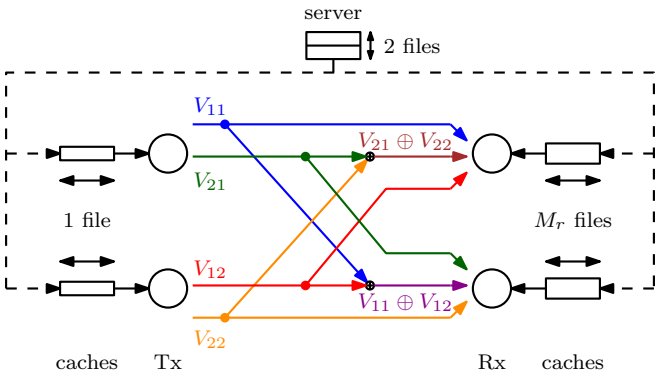
$$\frac{1}{\mathsf{DoF}} \geq 1 - \frac{1}{2}M_r,$$

which implies that our strategy is exactly optimal for $M_r \geq 4/5$, as illustrated by the dash-dotted line in Figure 4.9.

We will next give a high-level overview of the interference-extraction strategy. The proof of Theorem 17, including the details of the strategy, are given in Appendix C.6. Consider what happens when the main strategy is used in this 2×2 setup with $M_r = 0$. The strategy creates one unicast message from every transmitter to every receiver, and transmits them using interference



(a) Physical-layer view.



(b) Network-layer view.

Figure 4.10: Separation architecture with interference extraction in the 2×2 case with 2 files. The (unicast) X-channel message set is used, but every receiver decodes, in addition to its intended messages, the sum of the messages intended for the other reciever.

alignment. Each receiver thus gets the two messages intended for it, plus an alignment of the two messages intended for the other receiver. In the main strategy, this aligned interference is simply discarded. However, we can design the scheme in a way that this alignment is a simple sum of the two interfering messages. Each receiver can then decode, in addition to its intended messages, the sum of the interfering messages, without suffering any decrease in the sum DoF of the communicated messages. We hence obtain a new separation architecture, illustrated in Figure 4.10, that we use for all M_r.

The scheme we propose in this section is very specific to the 2×2 interference network with two files in the content library. An interesting direction for future work would be to extend this interference-extraction strategy to more general settings.

4.8 Discussion

In this chapter, we have presented the approximate degrees of freedom of cache-aided interference networks, with caches at both the transmitters and the receivers. While an exact characterization of the DoF is certainly desirable, finding it is a more difficult problem since the exact rate-memory trade-off is unknown even for the error-free broadcast case.

The DoF can be approximately achieved using the separation architecture described in Section 4.4, which decouples the physical-layer transmission scheme from the network-layer coded caching scheme. While this strategy is approximately optimal, some improvements can still be made, albeit with no more than a constant-factor gain. We explored one such improvement in Section 4.7 where the aligned subspaces that result from the physical-layer interference alignment scheme are extracted and used as additional bit pipes at the receivers.

In the literature, a similar setting to the one in this chapter was recently studied in [56]. However, since [56] focuses on one-shot linear schemes, the interference alignment gain is not achieved. This significantly reduces the achieved degrees of freedom, especially in the lower memory regime when the number of receivers is large. In particular, if $M_r = N/\sqrt{K_r}$ and $K_t \leq \sqrt{K_r}$, then we can show that the DoF achieved by our scheme is larger than the one-shot linear scheme by a

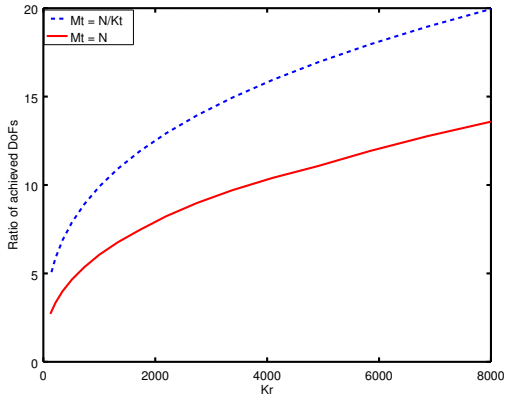


Figure 4.11: Largest ratio of the DoF achieved by our proposed scheme to the DoF achieved by the one-shot linear scheme proposed in [56]. In this figure, the number of receivers K_r is scaled, while $K_t = K_r^{1/3}$. The plot shows the maximum ratio between the DoFs over all possible receiver memory values $M_r \in [0, N]$. The comparison is made for two values of the transmitter memory, $M_t = N/K_t$ and $M_t = N$. Notice that the gap increases arbitrarily with K_r.

multiplicative factor of at least

$$\frac{K_t K_r}{(K_t + \sqrt{K_r})^2} \geq \frac{K_t}{4},$$

which can be arbitrarily large. A tighter comparison is numerically illustrated in Figure 4.11 for $K_t = K_r^{1/3}$ and M_t taking the values N/K_t and N.

Possible extensions to the problem include further improvements to the scheme, such as by using transmit zero-forcing or by placing coded content in the caches; a derivation of tighter outer bounds; and an exploration of the regime where the total transmitter memory is less than the size of the content library, i.e., $N - M_r \leq K_t M_t < N$. Several follow-up works have extended the results in a few of these directions [58, 59]. Another interesting question is to find the (exactly) optimal strategy when the problem imposes a restriction of uncoded cache placement, in a similar manner to [60, 61] for the broadcast case.

4.9 Low-SNR Regime

The focus of this chapter, and indeed in most of the works in the literature, has been on the high-SNR regime. In this section, we explore the energy-efficiency gains of caching by considering a fast-fading Gaussian interference channel in the low-SNR regime with caches at transmitters and

receivers. We propose a separation-based strategy that uses the transmitter caches to enable a transmit beamforming gain in addition to the usual multicasting gain (i.e., global caching gain) and local caching gain. We find that there is a trade-off between the beamforming gain and the multicasting gain and propose two variants of the strategy, each of which prioritizes one of the two gains. We show the approximate optimality (in the low-SNR regime) of each variant in two extreme cases: the variant prioritizing the beamforming gain is approximately optimal for the single-receiver case (i.e., the Gaussian multiple-access channel), while the variant prioritizing the multicasting gain is approximately optimal for the single-transmitter case (i.e., the Gaussian broadcast channel).

Since this is a large section, we split it into the following parts. Subsection 4.9.1 formally describes the problem setting. Subsection 4.9.2 presents the main results of the section. The achievable strategy is described in detail in Subsection 4.9.3, and Subsection 4.9.4 provides the proof of approximate optimality for the multiple-access case. Proof details are relegated to Appendix C.7.

4.9.1 Problem Setting

A content library contains N files, denoted by W_1 through W_N, of size F bits each. The content library is separated from its end users by a Gaussian interference network, whose receivers act as the users. Let L denote the number of transmitters in the interference network and K the number of receivers (i.e., users). Each transmitter is equipped with a cache of size M_tF bits, and each receiver is equipped with a cache of size M_rF bits. The goal is to utilize the caches to help transmit files requested by the receivers across the interference network. Two special cases that we will consider later in the section are the single-transmitter (broadcast) case with $L = 1$ and the single-receiver (multiple-access) case with $K = 1$.

The system operates in two phases. First, a *placement phase* occurs during which each cache is filled with some function of the files. This is done before the user demands are known. Second, a *delivery phase* occurs during which the user demands are revealed: each user k requests a file W_{d_k}, where $d_k \in \{1, \ldots, N\}$. Each transmitter ℓ responds by sending a codeword $\mathbf{x}_\ell = (x_\ell(1), \ldots, x_\ell(T))$ of length T through the interference network. The codeword $\mathbf{x}_\ell$ depends only on the user demands

and the contents of transmitter ℓ's cache. Receiver k then observes at time τ

$$y_k(\tau) = \sum_{\ell=1}^{L} g_{k\ell}(\tau) x_\ell(\tau) + z_k(\tau),$$

where $g_{k\ell}(\tau)$ are the iid complex channel gains, known causally at all transmitters and receivers, and $z_k(\tau)$ are iid additive white circulary-symmetric unit-variance complex Gaussian noise. We assume the channel gains are uniform phase shifts, i.e., $g_{k\ell}(\tau) = e^{j\theta_{k\ell}(\tau)}$, where j is the imaginary unit and $\theta_{k\ell}(\tau)$ are iid uniform over $[0, 2\pi)$. The channel inputs and outputs are also complex-valued. Receiver k proceeds to decode its requested file from $\mathbf{y}_k$ and the contents of its cache.

We impose a power constraint of P on the input, i.e.,

$$\frac{1}{T}||\mathbf{x}_\ell||^2 \leq P, \ \forall \ell \in \{1, \ldots, L\}.$$

The *rate* is defined as $R = F/T$. For a given P, we wish to find the largest rate $R^*(P)$ such that, for all possible user requests $(d_1, \ldots, d_K)$,

$$\max_k \Pr\left\{\hat{W}_k \neq W_{d_k}\right\} \to 0 \quad \text{as} \quad T \to \infty,$$

where $\hat{W}_k$ denotes the reconstruction of file W_{d_k} by user k. In this paper we will focus on the capacity per unit energy [62]

$$\widehat{R}^* = \lim_{P \to 0^+} \frac{R^*(P)}{P}.$$

This allows us to study the energy-efficiency gains that caching can provide.

4.9.2 Main Results

Our main contribution is a separation-based communication strategy consisting of a physical layer and a network layer. A message set is created from transmitters to receivers to serve as the interface between the physical layer and the network layer. The physical layer transmits these messages across the interference network, while the network layer uses these messages as error-free bit pipes in order to deliver the requested files to the users. This idea is similar to the one described

earlier in this chapter for the high-SNR regime.

We have shown in the previous sections of this chapter that, in the high-SNR regime, transmitter co-operation is not necessary for approximately achieving the degrees-of-freedom. In contrast, in the low-SNR regime, transmitter co-operation becomes essential as it enables the transmit beamforming of signals to the receivers, yielding a power gain. We therefore use the transmitter caches to create as much content overlap among the transmitters as possible, allowing them to co-operate and beamform signals to the intended receivers, thereby obtaining a significant power gain. In general, we are able to obtain maximal multicasting (and local caching) gains, as well as a significant beamforming gain. However, in special cases where the number of distinct file requests is small but the receiver memory is large, it is more beneficial to completely ignore the multicasting gain in favor of maximizing the beamforming gain.

In fact, there is a trade-off between the multicasting gain and the beamforming gain. In order to obtain maximal multicasting gain, the receivers need to cache distinct parts of the files in order to increase the number of coding opportunities and thus enable the multicasting of coded messages. Conversely, the beamforming gain can be improved by having all the receivers store common information. This reduces the size of the total content that must be stored at the transmitters, which allows for greater overlap at the transmitters for the same memory size at the cost of losing the multicasting gain.

We therefore propose two different schemes, both of which utilize the separation-based approach: a multicasting scheme and a beamforming scheme. The difference lies in the gain that each scheme prioritizes: the former prioritizes the multicasting (MC) gain while the latter prioritizes the beamforming (BF) gain. Let $\widehat{R}_{\mathrm{MC}}$ and $\widehat{R}_{\mathrm{BF}}$ denote the bits per unit energy achieved by these schemes respectively. By choosing the better of these two schemes in any given situation, we achieve

$$\widehat{R}^* \geq \max\left\{\widehat{R}_{\mathrm{MC}}, \widehat{R}_{\mathrm{BF}}\right\}. \tag{4.19}$$

The following two theorems provide the expressions for the bits per unit energy achieved by these schemes.

Theorem 18. *Let $\kappa = KM_r/N$ and $\lambda = LM_t/N$. When $\kappa \in \{0, 1, \ldots, K\}$ and $\lambda \in \{1, \ldots, L\}$, the*

multi-casting scheme achieves

$$\widehat{R}_{MC} = \frac{1}{\ln 2} \cdot \frac{\kappa + 1}{K - \kappa} \cdot \lambda \cdot L.$$

Theorem 19. *Let* $\tilde{\lambda} = \min\{LM_t/(N - M_r), L\}$. *When* $\tilde{\lambda} \in \{1, \ldots, L\}$, *the beamforming scheme achieves*

$$\widehat{R}_{BF} = \frac{1}{\ln 2} \cdot \frac{1}{\min\{N, K\}(1 - M_r/N)} \cdot \tilde{\lambda} \cdot L.$$

Note that we abuse notation when $M_r = N$ (equivalently, $\kappa = K$), when we can achieve an arbitrarily large rate.

Theorems 18 and 19 give the rate achieved at specific corner points of the transmitter and receiver memories. Since the *inverse* of the rate is a convex function of M_r and M_t [53], we can also achieve any linear combination of the inverse-rates of these points.

The next two subsections will analyze the two rate expressions and give a high-level overview of the schemes that achieve them. At the end of the section, we discuss the approximate optimality of each scheme in special cases.

The Multicasting Scheme

The multicasting scheme prioritizes the multicasting gain. To do so, it applies a receiver content placement strategy similar to the one in [4], in which receivers store different content in a way that maximizes coding opportunities. The transmitter content placement complements the receiver content placement by having subsets of transmitters share content.

More precisely, if $\kappa = KM_r/N$ and $\lambda = LM_t/N$ are integers, then every set of κ receivers and λ transmitters share some exclusive part of the content library. This creates opportunities for coded messages to be multicast to $\kappa + 1$ receivers at a time [4] while simultaneously allowing every λ transmitters to co-operate in order to beamform and produce a power gain.

The result is then a maximized multicasting gain and a significant, though not necessarily maximized, beamforming gain. More specifically, from Theorem 18 the sum rate achieved by the

multicasting scheme can be split into three main components:

$$K\widehat{R}_{\mathrm{MC}}P \approx \underbrace{\frac{1}{1-M_r/N}}_{G_{\mathrm{LC}}} \cdot \underbrace{\left(\frac{KM_r}{N}+1\right)}_{G_{\mathrm{MC}}} \cdot \underbrace{\frac{LM_t}{N}}_{G_{\mathrm{BF}}} \cdot LP \tag{4.20}$$

for P small enough. Here G_{LC} is the local caching gain, G_{MC} is the multicasting gain, and G_{BF} is the beamforming gain. In the equation, the LP term can be thought of as the total power constraint on the transmitters.

Notice that the local caching gain and the multicasting (global caching) gain are at their maximal value. Indeed, they are identical to those in [4], whose setup consists of a single transmitter and an error-free broadcast link to all receivers. The beamforming gain is approximately LM_t/N, which is equal to the number of copies of the content library that the transmitters can collectively store. In the multicasting scheme, every subset of LM_t/N transmitters share information in their caches, and they use this shared knowledge to co-operate and beamform messages to the receivers. In a typical MISO channel, the beamforming gain is the number of co-operating antennas, and this is similar to $G_{\mathrm{BF}} \approx LM_t/N$ in (4.20).

The Beamforming Scheme

The beamforming scheme ignores the multicasting gain in favor of improving the beamforming gain. This is done by having all receivers store the exact same content in their caches and having transmitters co-operate and beamform the remaining part of the desired file individually to each receiver (no multicasting). Since this makes a fraction of the content library available to all receivers, it is no longer necessary to store it at the transmitters. This effectively reduces the size of the content library that is "unavailable" to the receivers—and hence that must be stored at the transmitters—down to $NF' = (N-M_r)F$ bits. The transmitter memory can thus be expressed as $M_t/(1-M_r/N)\cdot F'$ bits. Consequently, more overlap is made possible among the transmitters, thus increasing the beamforming gain to its maximal value.

This scheme is particularly useful when the number of receivers is smaller than the number of transmitters and the receiver memory is large compared to the transmitter memory. In particular, it is approximately optimal when there is only one receiver, as discussed in Section 4.9.2 below.

From Theorem 19 we can write the sum rate of the beamforming scheme approximately as

$$\widetilde{K}\widehat{R}_{\mathrm{BF}}P \approx \underbrace{\frac{1}{1-M_r/N}}_{G_{\mathrm{LC}}} \cdot \underbrace{\min\left\{\frac{LM_t/N}{1-M_r/N}, L\right\}}_{G_{\mathrm{BF}}} \cdot LP \tag{4.21}$$

for P small enough, where $\widetilde{K} = \min\{N, K\}$ is the worst-case number of *distinct* file requests. Here G_{LC} is the local caching gain and G_{BF} is the beamforming gain. Note the absence of any multicasting gain. In the equation, the LP term can again be thought of as the total power constraint on the transmitters.

Note that, when $M_t < N - M_r$, the expression $1 - M_r/N$ normally associated with the local caching gain appears squared. This is due to the double effect of a receiver's local cache: on the one hand it provides the local caching benefit to each receiver; on the other hand it reduces the size of the part of the library "unavailable" to the receivers by a factor of $1 - M_r/N$, thus allowing for greater content overlaps among the transmitters. Indeed, instead of sharing content between only $\lambda = LM_t/N$ transmitters, we can now increase this number to $\tilde{\lambda} = \min\{LM_t/(N - M_r), L\} \geq \lambda$, which explains the beamforming gain G_{BF} in (4.21).

Approximate Optimality

The following theorems state that our separation-based approach is approximately optimal in the low-SNR regime for two cases: the multiple-access case ($K = 1$) and the broadcast case ($L = 1$). While the proof of approximate optimality for the broadcast case is a straightforward adaptation of the converse proof of [4] to the Gaussian low-SNR setup, the converse proof for the multiple-access case is more involved as it needs to capture the limits of possible co-operation among subsets of transmitters.

Theorem 20. *In the broadcast case, i.e., when $L = 1$ and $M_t = N$, the bits per unit energy achieved by the multicasting scheme are approximately optimal,*

$$1 \leq \frac{\widehat{R}^*}{\widehat{R}_{MC}} \leq 12,$$

for all $N \geq K$ and $M_r \in [0, N]$.[6]

The constant in Theorem 20 can be numerically sharpened to about 8.151 for $N, K \leq 100$.

Theorem 21. *In the multiple-access case, i.e., when $K = 1$, the bits per unit energy achieved by the beamforming scheme are approximately optimal,*

$$1 \leq \frac{\widehat{R}^*}{\widehat{R}_{BF}} \leq 64,$$

for all N, L, $M_r \in [0, N]$, and $M_t \in [(N - M_r)/L, N]$.

The constant in Theorem 21 can be numerically sharpened to about 4.701 for $N, L \leq 100$. Note that Theorem 21 holds for the entire memory regime of interest.

Notice that, in both these cases, we can assume without loss of generality that all the channel gains are one, i.e., all channel phase shifts are zero. Indeed, when $K = 1$, each transmitter can multiply its transmitted signal by the appropriate phase shift without affecting the power constraint or the (circularly symmetric) receiver noise. Similarly, when $L = 1$, each receiver can multiply its received signal by the appropriate phase shift. For this reason, Theorems 21 and 20 apply for both fading and static channels.

Finally, we conjecture that our separation-based approach is approximately optimal in the low-SNR regime for fading channels for all values of K and L, and proving this is part of our on-going work.

Comparison with the High-SNR Regime

We show in this section that, in the low-SNR regime, caching can provide three gains: the local caching gain, the multicasting (global caching) gain, and the beamforming gain. In the high-SNR regime, the first two gains are present, but instead of a beamforming gain there is an interference-alignment gain, as discussed in the previous sections of this chapter. Notably, the interference-alignment gain does not require transmitter co-operation for approximate optimality, contrary to the beamforming gain in the low-SNR regime. An interesting open problem is hence to analyze cache-aided communication in the transition regime from low to high SNR.

[6] The case $N < K$ is handled in Appendix C.7.4.

4.9.3 Achievable Strategy

We adopt a separation-based strategy as discussed in Section 4.9.2, separating the network layer from the physical layer. The idea is to create a set $\mathscr{V}$ of messages from (subsets of) transmitters and intended for (subsets of) receivers. This message set acts as an interface between the network and physical layers: the physical layer transmits the messages across the interference channel, while the network layer uses them as error-free bit pipes in order to apply a caching strategy that delivers to each receiver its requested file.

Define $[m] = \{1, \ldots, m\}$ for any integer m. Because of the symmetry in the problem, we will always choose message sets of the form

$$\mathscr{V}_{pq} \triangleq \{V_{\mathcal{K}\mathcal{L}} : \mathcal{K} \subseteq [K], |\mathcal{K}| = p, \mathcal{L} \subseteq [L], |\mathcal{L}| = q\}, \tag{4.22}$$

for some integers $p \in [K]$ and $q \in [L]$, where message $V_{\mathcal{K}\mathcal{L}}$ is to be sent collectively from the transmitters in $\mathcal{L}$ to the receivers in $\mathcal{K}$. In other words, the messages are always from every subset of q transmitters to every subset of p receivers, for some p, q. The physical layer assumes that message $V_{\mathcal{K}\mathcal{L}}$ is known to all the transmitters in $\mathcal{L}$. At the network layer, we therefore need to ensure that any bits sent through the bit pipe represented by $V_{\mathcal{K}\mathcal{L}}$ are shared by all the transmitters in $\mathcal{L}$.

Suppose that the physical layer is able to transmit all the messages in $\mathscr{V}_{pq}$ at a rate of R'_{pq} each. Suppose also that the network layer can send a total of $v_{pq}F$ bits through the messages (as bit pipes) in order to achieve its goal of delivering every file to the user that requested it. Thus we have $R'_{pq}T = v_{pq}F$. Since we also have $R = F/T$ by definition, this implies

$$v_{pq}F = R'_{pq}T \implies v_{pq}RT = R'_{pq}T \implies R = R'_{pq}/v_{pq}. \tag{4.23}$$

Therefore, by finding achievable values for v_{pq} and R'_{pq} for some pair (p, q), we obtain an achievable rate R.

As previously mentioned, we propose two different schemes, the multicasting scheme and the beamforming scheme. The difference in the two schemes lies in the network-layer strategy and

the choice of p and q: the multicasting scheme chooses to maximize p, whereas the beamforming scheme opts for maximizing q and setting $p = 1$. The physical-layer strategy on the other hand is agnostic to the choice of schemes.

The physical-layer strategy is described below and in Appendix C.7.2 along with its achieved rate R'_{pq}. The network-layer strategies of the two schemes are provided in Appendix C.7.1 along with their achieved values of v_{pq}.

Physical-Layer Strategy

Fix $p \in [K]$ and $q \in [L]$. We wish to transmit the messages $\mathscr{V}_{pq}$ across the network. Since we are focusing on the low-SNR regime, our strategy will attempt to get the largest power gain.

Consider a specific message $V_{\mathcal{K}\mathcal{L}} \in \mathscr{V}_{pq}$. Since the transmitters in $\mathcal{L}$ all share the message $V_{\mathcal{K}\mathcal{L}}$, they can co-operate and beamform it to at least one user. The idea is to schedule this message transmission when the channel is "favorable" for all the receivers in $\mathcal{K}$, at which point the transmitters can beamform to all receivers in $\mathcal{K}$ at once. By "favorable", we mean that all the receivers in $\mathcal{K}$ can get approximately the maximum benefit (power gain) from this beamforming. The result is the following achievable rate.

Lemma 9. *The message set $\mathscr{V}_{pq}$ can be transmitted across the interference network at a sum rate of*

$$\binom{L}{q}\binom{K}{p}\widehat{R}'_{pq} \geq \frac{Lq}{\ln 2}$$

bits per unit energy, where $\widehat{R}'_{pq} = \lim_{P\to 0^+} R'_{pq}(P)$.

Lemma 9 is proved in Appendix C.7.2, where we describe the above strategy in greater detail.

4.9.4 Approximate Optimality for the Multiple-Access Case

Recall that $K = 1$ in this case. Also recall that we can assume without loss of generality that all the channel gains are one. In order to prove approximate optimality, we first derive the following cut-set bounds on the optimal rate, proved in Appendix C.7.3.

Lemma 10. *For a single receiver (i.e., $K = 1$), the optimal rate must satisfy*

$$R^*(P) \leq \max_{\substack{\mathbf{Q}\in\mathbb{C}^{L\times L} \\ \mathbf{Q}\succeq 0,\ Q_{\ell\ell}\leq P}} \min_{\substack{\mathcal{L}\subseteq\{1,\ldots,L\} \\ (L-|\mathcal{L}|)M_t<N-M_r}} \frac{\log_2\left(1+\mathbf{1}^\top \mathbf{Q}_{\mathcal{L}|\mathcal{L}^c}\mathbf{1}\right)}{1-\frac{M_r+(L-|\mathcal{L}|)M_t}{N}},$$

where $\mathbf{1}$ *is the all-ones vector, and*

$$\mathbf{Q}_{\mathcal{L}|\mathcal{L}^c} = \mathbf{Q}_{\mathcal{L},\mathcal{L}} - \mathbf{Q}_{\mathcal{L},\mathcal{L}^c}\mathbf{Q}_{\mathcal{L}^c,\mathcal{L}^c}^{-1}\mathbf{Q}_{\mathcal{L}^c,\mathcal{L}}.$$

We will now use Lemma 10 to prove Theorem 21, following a similar approach to [63]. The main idea is to use properties of the objective function of the maximization in Lemma 10 to show that one maximizing covariance matrix $\mathbf{Q}$ has a symmetric structure, thereby reducing the maximization to just a single scalar variable.

We first swap the max over the covariance matrix $\mathbf{Q}$ and the min over the *size* of the subset $\mathcal{L}$, giving

$$R^*(P) \leq \min_{\substack{t\in[L] \\ M_r+(L-t)M_t<N}} \frac{N}{N-M_r-(L-t)M_t} \max_{\mathbf{Q}} \phi_t(\mathbf{Q}),$$

where we have defined

$$\phi_t(\mathbf{Q}) = \min_{|\mathcal{L}|=t} \log_2\left(1+\mathbf{1}^\top \mathbf{Q}_{\mathcal{L}|\mathcal{L}^c}\mathbf{1}\right).$$

By noticing that $\phi_t(\cdot)$ is both concave and invariant under permutation, we show in Appendix C.7.3 that one covariance matrix that maximizes $\phi_t(\cdot)$ must have the form

$$\mathbf{Q} = \left((1-\rho)\mathbf{I} + \rho\mathbf{1}\mathbf{1}^\top\right)\cdot P \tag{4.24}$$

for some $\rho \in [-1/(L-1), 1]$.

We can now rewrite the upper bound on $R^*(P)$ as

$$\min_{\substack{t\in[L] \\ L-t<\frac{N-M_r}{M_t}}} \max_{\rho\in[\frac{-1}{L-1},1]} \frac{t\left(1+(t-1)\rho-\frac{t(L-t)\rho^2}{1+(L-t-1)\rho}\right)}{\left(1-\frac{M_r+(L-t)M_t}{N}\right)(\ln 2)}P, \tag{4.25}$$

using $\log_2(1+x) \leq x/\ln 2$ and after some algebra. By optimizing over ρ and t, we obtain the result

of the theorem. For lack of space, we relegate this to Appendix C.7.3.

Chapter 5

Open Problems

This thesis has focused on three main aspects of the coded caching problem, in particular as it pertains to wireless heterogeneous networks (Figure 1.1). These are: non-uniform popularity, multiple AP access, and interference. In this section, we discuss some interesting open problems and possible future research directions.

The architectural insights presented in Chapter 4 hold for cache-aided Gaussian interference networks. A natural next step would be to study more general networks. In particular, is the separation architecture—which is approximately optimal in the Gaussian interference network—also approximately optimal for more general networks?

In Chapter 3, we found that in an adaptive matching setup there is a dichotomy between a coded delivery scheme (with a static matching) and a pure adaptive matching scheme (with uncoded delivery). Indeed, each scheme dominated in certain regimes. When the content popularity was close to uniform (i.e., when the Zipf parameter $\beta < 1$), we also showed the approximate optimality of the schemes in most memory regimes, and we further developed a hybrid scheme that combined elements of both schemes. The two most pertinent next questions are hence whether a hybrid can be developed for the $\beta > 1$ setup and whether we can prove approximate optimality results.

Finally, there are practical considerations that are not yet fully understood. For instance, this thesis has focused on communication efficiency, but many of the algorithms needed are not computationally efficient. Thus an interesting problem is finding computationally efficient algorithms that can achieve comparable rates to the theoretical limits. Another assumption in these works is that

of large file sizes. For example, many of the schemes require that files be split into an exponential (in the number of caches) number of parts. Hence it is desirable to analyze the problem when the file size is limited.

The above are only a few possible directions for further research. The relative recency and timeliness of coded content caching make it a rich and exciting field with many more open problems.

Appendix A

Detailed Proofs for Chapter 2

A.1 Proofs for the multi-user setup

A.1.1 Proof of Lemma 5

A small example for illustration

Before we give the full proof of Lemma 5, we will start with a simple example for illustration. Consider a multi-level, multi-user caching system with $K = 6$ caches and $L = 3$ levels. Suppose that $U_1 = U_2 = U_3 = 1$, and let N_1, N_2, and N_3 be some large numbers (their exact value is not important for this example). Finally, assume a single-access structure for all users, i.e., $d_1 = d_2 = d_3 = 1$.

As we have discussed in Section 2.5.2, the lower bounds on the optimal rate that we wish to obtain are a sum of L cut-set bounds, each pertaining to a single level. A cut-set bound for level i consists of a certain number of caches and broadcast messages, such that the users at these caches can use the broadcast to cooperatively decode a set of files from level i. For example, if we consider the sU_i users of level i connected to some s caches, we can send b broadcast messages, tailored for the correct user requests, so that the users can collectively decode $\min\{sU_i \cdot b, N_i\}$ files of level i. Furthermore, we usually choose the values of s and b so that the final bound matches the achieved individual rate of level i. Thus, they depend on value of the achieved rate and hence on the memory available to the level.

In our example, suppose that the memory we expect each level to receive dictates the following number of caches to consider: a single cache for level 1; two caches for level 2; and three caches for level 3. We define $s_1 = 1$, $s_2 = 2$, and $s_3 = 3$ to be these numbers. In order to group together the cut-set bounds of all the levels with their different numbers of caches, we resort to the sliding-window subset entropy inequality presented in Lemma 4.

In this example, we will consider six separate broadcast messages. For simplicity, we will assume that the messages are chosen so that, any time we encounter a group of p users and q messages, the users are able to decode a total of pq messages. This relaxation will be discarded when we discuss the general proof.

Let the caches be labeled with $Z_1, \ldots, Z_6$ and the broadcast messages with $X_1, \ldots, X_6$. We start with the level with the smallest s_i, in this case level 1. Since $s_1 = 1$, this means its cut-set bound will consider a single cache. In fact, the cut-set bound should have the following form by Fano's inequality:

$$H(Z_1, X_1) \geq H(Z_1, X_1 | \mathcal{W}^1) + 1 \cdot F,$$

where $\mathcal{W}^1$ represents the set of files of level 1 that got decoded. In this case, the single level-1 user at cache Z_1 can only decode one file when given just X_1, hence the $1 \cdot F$ term. However, to take full advantage of the sliding-window entropy inequality, we will take the average over six cut-set bounds for the same level, one for each cache:

$$\begin{aligned} RF + MF &\geq \frac{1}{6} \left[H(Z_1, X_1) + \cdots + H(Z_6, X_6) \right] \\ &\geq \frac{1}{6} \left[H(Z_1, X_1 | \mathcal{W}^1) + \cdots + H(Z_6, X_6 | \mathcal{W}^1) \right] \\ &\quad + 1 \cdot F. \end{aligned}$$

After completing the cut-set bound for level 1, we now attempt to transition to the cut-set bound for level 2. Because we have taken the average of six instances of cut-set bounds, we can use Lemma 4 to obtain 6 new cut-set bounds with $s_2 = 2$ caches each (for ease of notation, we write

$Y_k = (Z_k, X_k)$):

$$
\begin{aligned}
RF + MF \geq \frac{1}{6} \big[& H(Y_1|\mathcal{W}^1) + H(Y_2|\mathcal{W}^1) + H(Y_3|\mathcal{W}^1) \\
& + H(Y_4|\mathcal{W}^1) + H(Y_5|\mathcal{W}^1) + H(Y_6|\mathcal{W}^1) \big] \\
& + 1 \cdot F \\
\overset{(a)}{\geq} \frac{1}{6} \cdot \frac{1}{2} \big[& H(Y_1, Y_2|\mathcal{W}^1) + H(Y_2, Y_3|\mathcal{W}^1) \\
& + H(Y_3, Y_4|\mathcal{W}^1) + H(Y_4, Y_5|\mathcal{W}^1) \\
& + H(Y_5, Y_6|\mathcal{W}^1) + H(Y_6, Y_1|\mathcal{W}^1) \big] \\
& + 1 \cdot F \\
\overset{(b)}{\geq} \frac{1}{6} \cdot \frac{1}{2} \big[& H(Y_1, Y_2|\mathcal{W}^1, \mathcal{W}^2) \\
& + H(Y_2, Y_3|\mathcal{W}^1, \mathcal{W}^2) + H(Y_3, Y_4|\mathcal{W}^1, \mathcal{W}^2) \\
& + H(Y_4, Y_5|\mathcal{W}^1, \mathcal{W}^2) + H(Y_5, Y_6|\mathcal{W}^1, \mathcal{W}^2) \\
& + H(Y_6, Y_1|\mathcal{W}^1, \mathcal{W}^2) \big] \\
& + 1 \cdot F + \frac{4}{2} \cdot F.
\end{aligned}
$$

Here, inequality (a) uses Lemma 4, while inequality (b) uses Fano's inequality on the level-2 cut-set bounds. Since each bound involves two users and two broadcast messages, the total number of files decoded is 4, hence the $4 \cdot F$ term. The set of decoded level-2 files is denoted by $\mathcal{W}^2$.

We proceed again with the transition from level 2 to level 3. Just like before, we first apply Lemma 4 to obtain entropy terms with the correct number of caches s_3, and then apply Fano's

inequality to decode files from level 3, labeled $\mathcal{W}^3$.

$$
\begin{aligned}
RF + MF \geq \frac{1}{6} \cdot \frac{1}{2} \big[& H(Y_1, Y_2 | \mathcal{W}^1, \mathcal{W}^2) \\
& + H(Y_2, Y_3 | \mathcal{W}^1, \mathcal{W}^2) + H(Y_3, Y_4 | \mathcal{W}^1, \mathcal{W}^2) \\
& + H(Y_4, Y_5 | \mathcal{W}^1, \mathcal{W}^2) + H(Y_5, Y_6 | \mathcal{W}^1, \mathcal{W}^2) \\
& + H(Y_6, Y_1 | \mathcal{W}^1, \mathcal{W}^2) \big] \\
& + 1 \cdot F + \frac{4}{2} \cdot F \\
\overset{(a)}{\geq} \frac{1}{6} \cdot \frac{1}{3} \big[& H(Y_1, Y_2, Y_3 | \mathcal{W}^1, \mathcal{W}^2) \\
& + H(Y_2, Y_3, Y_4 | \mathcal{W}^1, \mathcal{W}^2) \\
& + \cdots \\
& + H(Y_6, Y_1, Y_2 | \mathcal{W}^1, \mathcal{W}^2) \big] \\
& + 1 \cdot F + \frac{4}{2} \cdot F \\
\overset{(b)}{\geq} \frac{1}{6} \cdot \frac{1}{3} \big[& H(Y_1, Y_2, Y_3 | \mathcal{W}^1, \mathcal{W}^2, \mathcal{W}^3) \\
& + H(Y_2, Y_3, Y_4 | \mathcal{W}^1, \mathcal{W}^2, \mathcal{W}^3) \\
& + \cdots \\
& + H(Y_6, Y_1, Y_2 | \mathcal{W}^1, \mathcal{W}^2, \mathcal{W}^3) \big] \\
& + 1 \cdot F + \frac{4}{2} \cdot F + \frac{9}{3} \cdot F.
\end{aligned}
$$

Again, inequality (a) uses Lemma 4, and inequality (b) uses Fano's inequality on level 3, which is considering groups of 3 users and 3 broadcast messages, allowing the decoding of a total of 9 files.

Finally, by the non-negativity of entropy, the lower bound becomes:

$$
\begin{aligned}
RF + MF &\geq \left(1 + \frac{4}{2} + \frac{9}{3}\right) \cdot F \\
R + M &\geq 6.
\end{aligned}
$$

The general proof

The process involved in the general case is similar to the one shown in the example above. We start with the level with the smallest s_i and write an average of K cut-set bounds for this level, one for each sequence of consecutive s_i caches. After applying Fano's inequality, we use Lemma 4 to transition to the level with the next smallest s_i. In this discussion, the popularity of the levels is not important, but the order of their s_i's is. Specifically, if $s_i < s_j$, it does not matter which of i and j is more popular. Thus, we can assume without loss of generality that $s_1 \leq \cdots \leq s_L$.

In the example, we started with cut-set bounds with a single cache each. For technical reasons, this will not give us good enough bounds in general, and so the initial cut-set bounds will consist of t caches, for a general $t \in \{1, \ldots, K\}$. In fact, every consecutive t caches will be clustered into an inseparable group. The group consisting of the t caches that start with cache k is labeled as:

$$\mathcal{Z}_k^t = \left(Z_k, \ldots, Z_{\langle k+t-1 \rangle}\right),$$

where $\langle m \rangle$ is defined for integers m as in Lemma 4, i.e., $\langle m \rangle = m$ if $m \leq K$ and $\langle m \rangle = m - K$ if $m > K$. To every cache group $\mathcal{Z}_k^t$, we associate a broadcast-message group $\mathcal{X}_k^b$, which consists of b messages serving different user demands.

Recall that a level-i user needs to connect to d_i *consecutive* caches in order to decode whichever file he has requested. Applying Lemma 4 should hence keep only consecutive caches in the same cut-set bounds, which will allow for the maximum number of users to be active in the decoding of the files (and thus produce a larger number of files from the same cut-set bound). While this was fairly simple to ensure in the example above, we must show that we can still do it even after introducing the t-groups of caches. This is done in the next paragraph.

Let g be the GCD of t and K. Then, the following sequence:

$$\left(\mathcal{Z}_1^t, \mathcal{Z}_{\langle t+1 \rangle}^t, \ldots, \mathcal{Z}_{\langle (K/g)t+1 \rangle}^t\right),$$

starts at cache 1 and ends at cache K. Each cache Z_k appears exactly (t/g) times in the sequence. Furthermore, every pair of caches that are consecutive in the sequence are also consecutive in the

system. For example, suppose $t = 4$ and $K = 6$. Then their GCD would be $g = 2$, with $K/g = 3$, and the sequence would be:

$$\Big((Z_1, Z_2, Z_3, Z_4), (Z_5, Z_6, Z_1, Z_2), (Z_3, Z_4, Z_5, Z_6)\Big).$$

Notice that every cache appears in the sequence $t/g = 2$ times, and that consecutive caches remain so in the sequence.

We are now ready to prove the lower bounds. Start with an average of K/g cut-set bounds consisting of one group of t caches (and their associated broadcast messages) each:

$$bRF + tMF \geq \frac{g}{K} \sum_{k=1}^{K/g} H\left(\mathcal{Z}_{(k-1)g+1}^{t}, \mathcal{X}_{(k-1)g+1}^{b}\right).$$

This follows from $bR + tM \geq H(\mathcal{Z}_k^t, \mathcal{X}_k^b)$ for all k. Notice how all the indices $[(k-1)g+1]$ are the same modulo g. We can include all other caches to get:

$$bRF + tMF \geq \frac{1}{K} \sum_{k=1}^{K} H\left(\mathcal{Z}_{k}^{t}, \mathcal{X}_{k}^{b}\right).$$

The first step is to use Lemma 4 to obtain cut-set bounds with $s_1 t$ caches for level 1.

$$\begin{aligned} & bRF + tMF \\ & \quad \geq \frac{1}{K} \sum_{k=1}^{K} H\left(\mathcal{Z}_{k}^{t}, \mathcal{X}_{k}^{b}\right) \\ & \quad \geq \frac{1}{K} \cdot \frac{1}{s_1} \sum_{k=1}^{K} H\Big(\mathcal{Z}_{k}^{t}, \mathcal{Z}_{\langle k+t\rangle}^{t}, \ldots, \mathcal{Z}_{\langle k+(s_1-1)t\rangle}^{t}, \\ & \qquad\qquad\qquad \mathcal{X}_{k}^{b}, \mathcal{X}_{\langle k+t\rangle}^{b}, \ldots, \mathcal{X}_{\langle k+(s_1-1)t\rangle}^{b}\Big) \end{aligned}$$

Each entropy term in the sum now consists of $s_1 t$ *consecutive* caches and $s_1 b$ broadcast messages. For simplicity, we write it as $\hat{H}_k(s_1 t, s_1 b)$, for $k \in \{1, \ldots, K\}$, and its conditional version as $\hat{H}_k(s_1 t, s_1 b | Q)$ for any random variable Q. Let $\mathcal{W}^1$ denote the set of level-1 files that can be decoded in every cut-set bound in the sum, and let p_1 be its size, to be determined later. For technical reasons, we will only apply Fano's inequality on a fraction of the cut-set bounds; the rest

are simply lower-bounded by conditioning. The meaning and value of this fraction is discussed below, but for now we will call it λ_1. Then, we use Fano's inequality to get:

$$\begin{aligned} bRF + tMF &\geq \frac{1}{K} \cdot \frac{1}{s_1} \sum_{k=1}^{K} \hat{H}_k(s_1 t, s_1 b) \\ &\geq \frac{1}{K} \cdot \frac{1}{s_1} \sum_{k=1}^{K} \hat{H}_k(s_1 t, s_1 b | \mathcal{W}^1) + \frac{\lambda_1 p_1}{s_1} \cdot F. \end{aligned}$$

By applying Lemma 4 again, we obtain cut-set bounds pertaining to level 2. We lower-bound a fraction λ_2 of them in turn using Fano's inequality, and repeat for all levels. Thus, we have:

$$\begin{aligned} bRF + tMF &\geq \frac{1}{K} \cdot \frac{1}{s_1} \sum_{k=1}^{K} \hat{H}_k(s_1 t, s_1 b | \mathcal{W}^1) + \frac{\lambda_1 p_1}{s_1} \cdot F \\ &\overset{(a)}{\geq} \frac{1}{K} \cdot \frac{1}{s_2} \sum_{k=1}^{K} \hat{H}_k(s_2 t, s_2 b | \mathcal{W}^1) + \frac{\lambda_1 p_1}{s_1} \cdot F \\ &\overset{(b)}{\geq} \frac{1}{K} \cdot \frac{1}{s_2} \sum_{k=1}^{K} \hat{H}_k(s_2 t, s_2 b | \mathcal{W}^1, \mathcal{W}^2) \\ &\quad + \frac{\lambda_1 p_1}{s_1} \cdot F + \frac{\lambda_2 p_2}{s_2} \cdot F \\ &\geq \cdots \\ &\geq \sum_{i=1}^{L} \frac{\lambda_i p_i}{s_i} \cdot F. \end{aligned} \tag{A.1}$$

where the inequality marked with (a) uses Lemma 4 and the one marked with (b) uses Fano's inequality.

For any $m < K$ *consecutive* caches, the number of level-i users that are connected to d_i of those caches is exactly $(m - d_i + 1)U_i$. Therefore, given $s_i t < K$ consecutive caches and $s_i b$ broadcast messages, the users at these caches should, in principle, decode up to $p_i = \min\{(s_i t - d_i + 1)U_i \cdot s_i b, N_i\}$ files from level i. This requires choosing the broadcast messages for the correct user demands. However, consider a pair of broadcast message X' and user u that appears in multiple cut-set bounds. In other words, the user u is expected to decode a file using the message X' on multiple occasions. We must ensure that there are no contradictions, i.e., that the message X' always delivers the same file to user u.

It turns out it is not always possible to do that and still be able to decode the maximal number

of files at every cut-set bound. However, because $s_i t \leq K/2$ (as constrained in the statement of Lemma 5), we can get around this issue by "ignoring" half of the cut-set bounds. To understand this, suppose we wish to determine, for each broadcast message to be sent, the set of user requests that it must satisfy. Starting with $\hat{H}_1(s_i t, s_i b)$: there are a total of $(s_i t - d_i + 1)U_i$ relevant users, and $s_i b$ broadcasts. We can design these broadcasts to allow these users to decode all p_i files. Next, we move on to $\hat{H}_2(s_i t, s_i b)$. In this entropy term, exactly tU_i users from the previous term are replaced by tU_i *brand new* users; the other users are still the same. Thus, if the broadcast messages serve the same files to the new users as they did to the old, the same total number of files can be decoded. This can go on as long as every step introduces new tU_i users, which is true for all but the last $(s_i t - d_i)$ steps. Indeed, the term $\hat{H}_{K-(s_i t-d_i)+1}(s_i t, s_i b)$ re-introduces users that previously appeared in $\hat{H}_1(s_i t, s_i b)$. Thus, for every level i, only a fraction $\frac{K-(s_i t-d_i)}{K}$ of the cut-set bounds can decode all p_i files using Fano's inequality. We bound this fraction by:

$$\frac{K-(s_i t-d_i)}{K} \geq \begin{cases} \frac{1}{2} & \text{if } s_i t > d_i; \\ 1 & \text{if } s_i t = d_i, \end{cases}$$

and we define λ_i to be the right-hand side of the inequality.

By substituting p_i for its value in (A.1), we get the final bounds:

$$bRF + tMF \geq \sum_{i=1}^{L} \frac{\lambda_i}{s_i} \cdot \min\left\{(s_i t - d_i + 1)U_i \cdot s_i b, N_i\right\} \cdot F$$
$$R \geq \sum_{i=1}^{L} \lambda_i \cdot \min\left\{(s_i t - d_i + 1)U_i, \frac{N_i}{s_i b}\right\} - \frac{t}{b}M,$$

which concludes the proof of Lemma 5.

A.1.2 Proof of approximate optimality (Theorem 4)

In this section, we will use the information-theoretic lower bounds determined in Lemma 5 to give an upper bound on the ratio between the rate achieved by the memory-sharing scheme and the optimal rate (the "gap"). In order to do that, we must consider a few different cases, in each of which different values are chosen for the parameters b, t, and $\{s_i\}_i$ defined in Lemma 5. When

these parameters are chosen, we will need to evaluate the minimization seen in the expression of the lower bound, for every level. For simplicity, we define, for each level i:

$$A_i = \min\left\{(s_i t - d_i + 1)U_i\,,\ \frac{N_i}{s_i b}\right\}. \tag{A.2}$$

To find or bound the value of A_i, we must evaluate the following comparison:

$$bs_i(s_i t - d_i + 1) \overset{?}{\lessgtr} \frac{N_i}{U_i}. \tag{A.3}$$

Before the main analysis, we consider the case where the number of caches is bounded. Specifically, we consider $K < k_0 = D/\beta$, and we call this "Case 0". Then, we consider the more interesting case where K is unbounded, and divide that into two main regimes. The first, "Case 1", is when the set I_1, defined in Definition 1 and (2.10), is empty; the second, "Case 2", is when it is not empty.

For convenience, we will assume, without loss of generality, that the levels are numbered from most popular to least popular. In other words:

$$U_1/N_1 \geq \cdots \geq U_L/N_L. \tag{A.4}$$

Case 0: $K < k_0 = D/\beta$

Recall the refined M-feasible partition described in Definition 1 and (2.10). By (2.9), we know that the memory given to a level $i_0 \in I_0$ is at most:

$$\alpha_{i_0} M < (2/K)\sqrt{N_{i_0}/U_{i_0}}.$$

However, if $K < D/\beta$, then, for any level $i \in I'$:

$$\begin{aligned}
\alpha_i M &\leq \left(\frac{1}{K} + \frac{\beta}{d_i}\right)\sqrt{\frac{N_i}{U_i}} \\
&= \frac{1}{K}\left(1 + \frac{\beta K}{d_i}\right)\sqrt{\frac{N_i}{U_i}} \\
&< \frac{1}{K}\left(1 + \frac{D}{d_i}\right)\sqrt{\frac{N_i}{U_i}} \\
&\leq \frac{D+1}{K}\sqrt{\frac{N_i}{U_i}} \\
&= \frac{D+1}{2} \cdot \frac{2}{K}\sqrt{\frac{N_i}{U_i}}.
\end{aligned}$$

This suggests that the memory given to a level in I' will not be much larger than that given to a level in I_0. As a result, levels in I' are expected to behave similarly to those in I_0: they receive so little memory that their impact on the overall transmission rate is not much different from that of the levels in H, which get zero memory. The effective result is that the subset I_1 "dominates" the set I.

With these observations in mind, we next describe a near-equivalent formulation of the achievability scheme, which is more suitable for the case $K < D/\beta$. This formulation will emphasize the fact that I_1 is the dominant subset of I, by essentially reducing I to I_1 and relegating I_0 and I' to H.

A more precise explanation follows. Find the (unique) level i^* such that:

$$\sum_{i=1}^{i^*-1} \frac{N_i}{d_i} \leq M \leq \sum_{i=1}^{i^*} \frac{N_i}{d_i}.$$

Recall that the levels are numbered from most popular to least popular, as seen in (A.4).

Partition the set of levels into three sets (H, I, J), which will serve the same purpose as the partition described in Definition 1. We set $H = \{i^* + 1, \ldots, L\}$, $I = \{i^*\}$, and $J = \{1, \ldots, i^* - 1\}$. We then proceed as usual: the levels in J are fully stored in the caches; the levels in H are given no memory; and the single level in I, i^*, is given the remaining memory $M - T_J$. The conventional scheme is applied to each level and its corresponding memory, resulting in the following transmission

rate:

$$R = \sum_{h \in H} KU_h + KU_{i^*} \left(1 - \frac{M - T_J}{N_{i^*}/d_{i^*}}\right)$$
$$\leq k_0 \left[\sum_{h \in H} U_h + U_{i^*} \left(1 - \frac{M - T_J}{N_{i^*}/d_{i^*}}\right)\right]. \tag{A.5}$$

For the lower bounds, consider Lemma 5, with $t = 1$, $s_i = d_i$ for all i, and $b = \lceil N_{i^*}/d_{i^*}U_{i^*} \rceil$. Thus $\lambda_i = 1$ for all i. We will first analyze the comparisons in (A.3) for every level.

For i^*, we have:

$$bs_{i^*}(s_{i^*}t - d_{i^*} + 1) = bd_{i^*} \geq \frac{N_{i^*}}{U_{i^*}}.$$

For $j \in J$, we have:

$$bs_j(s_j t - d_j + 1) = bd_j \geq \frac{N_{i^*}}{U_{i^*}} \cdot \frac{d_j}{d_{i^*}} \geq \frac{N_{i^*}}{U_{i^*} D} \geq \frac{N_j}{U_j},$$

by regularity condition (2.4).

Finally, for $h \in H$, we have:

$$bs_h(s_h - d_h + 1) = bd_h \leq 2\frac{N_{i^*}}{U_{i^*}} \cdot \frac{d_h}{d_{i^*}} \leq \frac{N_h}{U_h},$$

again by the regularity condition (2.4).

Putting these together, we determine the value of A_i, defined in (A.2), for each i and get the following lower bound on the optimal rate:

$$R^*(M) \geq \sum_{h \in H} U_h + \frac{N_{i^*}}{d_{i^*}b} + \sum_{j \in J} \frac{N_j}{d_j b} - \frac{M}{b}$$
$$= \sum_{h \in H} U_h + \frac{N_{i^*}/d_{i^*} - (M - T_J)}{b}$$
$$\geq \sum_{h \in H} U_h + \frac{N_{i^*}/d_{i^*} - (M - T_J)}{2N_{i^*}/d_{i^*}U_{i^*}}$$
$$\geq \frac{1}{2}\left[\sum_{h \in H} U_h + U_{i^*} \left(1 - \frac{M - T_J}{N_{i^*}/d_{i^*}}\right)\right]. \tag{A.6}$$

By combining (A.5) with (A.6), we get the following bound on the gap:

$$\frac{R(M)}{R^*(M)} \leq 2k_0 = 2D/\beta = 396D. \tag{A.7}$$

Case 1: $I_1 = \emptyset$

When $I_1 = \emptyset$, the rate achieved by the memory-sharing scheme is, according to Lemma 3:

$$R(M) \leq \sum_{h \in H} KU_h + \frac{2S_I^2}{M - T_J + V_I}. \tag{A.8}$$

For technical reasons, the lower bounds analysis for this case has to be broken down into two subcases, depending on whether or not the set J is empty.

Since we will be dealing with many floors and ceilings, here are a few remarks on these operations.

- If $n \geq 1$ is an integer, then $x \geq n \implies \lfloor x \rfloor \geq n$;
- For all $x > 0$, $\lfloor x \rfloor \geq x - 1$ and $\lceil x \rceil \leq x + 1$;
- If $x \geq 1$, then $\lfloor x \rfloor \geq x/2$ and $\lceil x \rceil \leq 2x$;
- If $a \leq x \leq b$, then $\lfloor a \rfloor \leq \lfloor x \rfloor \leq \lfloor b \rfloor$ and $\lceil a \rceil \leq \lceil x \rceil \leq \lceil b \rceil$.

Case 1a: $J \neq \emptyset$ Recall that $\tilde{M} = (M - T_J + V_I)/S_I$. Consider Lemma 5, with the following parameters:

$$\begin{aligned} & & t &= 1; \\ \forall h \in H, & & s_h &= \left\lfloor \frac{1}{8} K \right\rfloor; \\ \forall i \in I, & & s_i &= \left\lfloor \frac{\sqrt{N_i/U_i}}{8\tilde{M}} \right\rfloor; \\ \forall j \in J, & & s_j &= d_j; \\ & & b &= \left\lfloor \delta \tilde{M}^2 \right\rfloor, \end{aligned}$$

where $\delta = D/\beta$. Notice that $\lambda_j = 1$ for all $j \in J$, and $\lambda_i \geq \frac{1}{2}$ for all other levels i.

The first step is to verify that these parameters satisfy all of the constraints imposed on them in Lemma 5. For ease of reference, we repeat the constraints here:

- $t \in \{1, \ldots, K\}$;
- $s_i \in \mathbb{N}^+$ such that $s_i t \in \{d_i, \ldots, \lfloor K/2 \rfloor\}$, for any level i;
- $b \in \mathbb{N}^+$.

The parameters t and s_j, $j \in J$, can be easily seen to satisfy theirs. Also, it follows from Definition 1 and from regularity condition (2.2) that, for some $j \in J$:

$$\begin{aligned}\delta \tilde{M}^2 &\geq \delta \cdot \left(\frac{1}{d_j} + \frac{1}{K}\right)^2 \frac{N_j}{U_j} \geq D/\beta \cdot \left(\frac{1}{d_j} + \frac{1}{K}\right)^2 \cdot K \\ &\geq k_0 \cdot \left(\frac{1}{D}\right)^2 \cdot k_0 = \left(\frac{k_0}{D}\right)^2 \geq 1,\end{aligned}$$

and therefore $b \geq 1$.

As for s_h, $h \in H$: we have $s_h \leq \lfloor K/2 \rfloor$ trivially, and $s_h \geq d_h$ is true because $K \geq k_0 = D/\beta \geq 8d_h$.

Finally, consider $i \in I$. First, recall that $I_1 = \emptyset$, and therefore $i \notin I_1$. Then, using Definition 1, we have:

$$\begin{aligned}\frac{\sqrt{N_i/U_i}}{8\tilde{M}} &\geq \frac{1}{8} \cdot \frac{1}{\frac{\beta}{d_i} + \frac{1}{K}} = \frac{1}{8(\beta + d_i/K)} \cdot d_i \\ &\geq \frac{1}{16\beta} \cdot d_i \geq d_i,\end{aligned}$$

which implies $s_i \geq d_i$. Also, we have:

$$s_i = \left\lfloor \frac{\sqrt{N_i/U_i}}{8\tilde{M}} \right\rfloor \leq \left\lfloor \frac{1}{8} \cdot \frac{1}{\frac{1}{K}} \right\rfloor \leq \lfloor K/2 \rfloor .$$

Thus all the parameters satisfy their constraints.

Next, we will compute bounds on the A_i terms by evaluating the comparison (A.3) for all levels i.

For $h \in H$, the comparison (A.3) gives:

$$bs_h(s_h t - dh + 1) \leq bs_h^2 t \leq \delta \tilde{M}^2 \cdot \frac{1}{64} K^2 \leq \frac{\delta}{64} \cdot \frac{N_h}{U_h}.$$

Thus:

$$\begin{aligned}
A_h &\geq \min\left\{1, \frac{64}{\delta}\right\} \cdot (s_h t - d_h + 1) U_h \\
&= \min\left\{1, \frac{64\beta}{D}\right\} \cdot (s_h t - d_h + 1) U_h \\
&\geq \frac{64\beta}{D} \cdot \left(\frac{1}{8} K - d_h\right) U_h \\
&\geq \frac{64\beta}{D} \cdot \left(\frac{1}{8} - \beta\right) \cdot K U_h \\
&\geq \frac{8\beta}{D} \cdot (1 - 8\beta) \cdot K U_h.
\end{aligned}$$

Consider now $i \in I$. The comparison (A.3) gives:

$$bs_i(s_i t - d_i + 1) \leq bs_i^2 t \leq \delta \tilde{M}^2 \cdot \frac{1}{64} \frac{N_i/U_i}{\tilde{M}^2} = \frac{\delta}{64} \cdot \frac{N_i}{U_i}.$$

Therefore,

$$\begin{aligned}
A_i &\geq \min\left\{1, \frac{64}{\delta}\right\} \cdot (s_i t - d_i + 1) U_i \\
&= \frac{64\beta}{D} \cdot (s_i t - d_i + 1) U_i \\
&\geq \frac{64\beta}{D} \cdot \left(\frac{\sqrt{N_i/U_i}}{8\tilde{M}} - d_i\right) U_i \\
&= \frac{64\beta}{D} \cdot \left(\frac{1}{8} - d_i \cdot \frac{\tilde{M}}{\sqrt{N_i/U_i}}\right) \cdot \frac{\sqrt{N_i U_i}}{\tilde{M}} \\
&\geq \frac{64\beta}{D} \cdot \left(\frac{1}{8} - \left(\beta + \frac{d_i}{K}\right)\right) \cdot \frac{\sqrt{N_i U_i}}{\tilde{M}} \\
&\geq \frac{64\beta}{D} \cdot \left(\frac{1}{8} - 2\beta\right) \cdot \frac{\sqrt{N_i U_i}}{\tilde{M}} \\
&\geq \frac{8\beta}{D} \cdot (1 - 16\beta) \cdot \frac{\sqrt{N_i U_i}}{\tilde{M}}.
\end{aligned}$$

Finally, let us look at $j \in J$. The comparison in (A.3) gives:

$$\begin{aligned} bs_j(s_j t - d_j + 1) &= bd_j \\ &\geq \frac{1}{2}\delta \tilde{M}^2 \cdot d_j \\ &\geq \frac{1}{2}(D/\beta)\left(\frac{1}{d_j} + \frac{1}{K}\right)^2 \frac{N_j}{U_j} d_j \\ &\geq \frac{1}{2\beta} d_j^2 \cdot \left(\frac{1}{d_j}\right)^2 \cdot \frac{N_j}{U_j} \\ &\geq \frac{N_j}{U_j}. \end{aligned}$$

Therefore:

$$A_j = \frac{N_j}{d_j b}.$$

We can now use the values of $\{A_i\}_i$ to lower-bound the optimal rate as in Lemma 5. Recall

also the values of λ_i mentioned at the beginning of this section.

$$\begin{aligned}
R^*(M) &\geq \sum_{h\in H} \frac{1}{2}\cdot\frac{8\beta}{D}\cdot(1-8\beta)\cdot KU_h \\
&\quad + \sum_{i\in I} \frac{1}{2}\cdot\frac{8\beta}{D}\cdot(1-16\beta)\cdot\frac{\sqrt{N_iU_i}}{\tilde{M}} \\
&\quad + \sum_{j\in J}\frac{N_j}{d_jb} - \frac{M}{b} \\
&= \sum_{h\in H} \frac{4\beta}{D}\cdot(1-8\beta)\cdot KU_h \\
&\quad + \frac{4\beta}{D}\cdot(1-16\beta)\cdot\frac{S_I\cdot\sum_{i\in I}\sqrt{N_iU_i}}{M-T_J+V_I} \\
&\quad - \frac{M-T_J}{b} \\
&\geq \sum_{h\in H} \frac{4\beta}{D}\cdot(1-8\beta)\cdot KU_h \\
&\quad + \frac{4\beta}{D}\cdot(1-16\beta)\cdot\frac{S_I^2}{M-T_J+V_I} \\
&\quad - \frac{M-T_J+V_I}{\frac{1}{2}\delta(M-T_J+V_I)^2/S_I^2} \\
&= \sum_{h\in H} \frac{4\beta}{D}\cdot(1-8\beta)\cdot KU_h \\
&\quad + \left[\frac{4\beta}{D}\cdot(1-16\beta) - \frac{2}{\delta}\right]\cdot\frac{S_I^2}{M-T_J+V_I} \\
&= \sum_{h\in H} \frac{\beta}{D}\cdot(4-32\beta)\cdot KU_h \\
&\quad + \frac{\beta}{D}(1-32\beta)\cdot\frac{2S_I^2}{M-T_J+V_I}. \qquad \text{(A.9)}
\end{aligned}$$

Combining (A.9) with (A.8), we get:

$$\frac{R(M)}{R^*(M)} \leq \frac{D/\beta}{1-32\beta} \leq 237D. \qquad \text{(A.10)}$$

Case 1b: $J=\emptyset$ We know that I is never an empty set. Since $J=\emptyset$, then the most popular level is in I. By (A.4), that level is level 1, and hence $1\in I$. Moreover, we note that $T_J=0$ when $J=\emptyset$, and thus $\tilde{M}=(M+V_I)/S_I$.

Consider again Lemma 5, and define the following parameters:

$$\begin{aligned} t &= \left\lfloor \frac{1}{32} \frac{\sqrt{N_1/U_1}}{\tilde{M}} \right\rfloor; \\ \forall h \in H, \qquad s_h &= \left\lfloor 2 \frac{K\tilde{M}}{\sqrt{N_1/U_1}} \right\rfloor; \\ \forall i \ \in I, \qquad s_i &= \left\lfloor 2 \sqrt{\frac{N_i/U_i}{N_1/U_1}} \right\rfloor; \\ b &= \left\lfloor 8\tilde{M}\sqrt{N_1/U_1} \right\rfloor. \end{aligned}$$

Thus, we can only say that $\lambda_i \geq \frac{1}{2}$ for all levels i.

As before, we must first verify that these parameters satisfy all their constraints. Let us start with b:

$$8\tilde{M}\sqrt{N_1/U_1} \geq 8 \cdot \frac{1}{K}\sqrt{N_1/U_1} \cdot \sqrt{N_1/U_1} = \frac{8N_1}{KU_1} \geq 1,$$

by regularity condition (2.2), and therefore $b \geq 1$.

Next, we will examine t and the $\{s_i\}_i$. It suffices to show $t \geq 1$, $s_i \geq 1$, $s_i t \geq d_i$, and $s_i t \leq \lfloor K/2 \rfloor$ for all i. We can see that $t \geq 1$ because:

$$\frac{\sqrt{N_1/U_1}}{32\tilde{M}} \geq \frac{1/32}{\frac{\beta}{d_1} + \frac{1}{K}} \geq \frac{1}{64\beta} \cdot d_1 \geq 1,$$

where the first inequality follows from $1 \in I \setminus I_1$ (recall $I_1 = \emptyset$) and Definition 1 combined with (2.10).

For $h \in H$, we have $2K\tilde{M}/\sqrt{N_1/U_1} \geq 1$ directly from Definition 1, and thus $s_h \geq 1$. Moreover,

$$s_h t \geq \frac{1}{64} \cdot \frac{\sqrt{N_1/U_1}}{\tilde{M}} \cdot \frac{K\tilde{M}}{\sqrt{N_1/U_1}} = \frac{K}{64} \geq D \geq d_h,$$

and,

$$s_h t \leq \frac{1}{32} \cdot \frac{\sqrt{N_1/U_1}}{\tilde{M}} \cdot 2\frac{K\tilde{M}}{\sqrt{N_1/U_1}} = K/16 \leq \lfloor K/2 \rfloor.$$

Consider now $i \in I$. Because level 1 is the most popular level, then $N_i/U_i \geq N_1/U_1$ for all i,

and hence $s_i \geq 1$. Moreover:

$$s_i t \geq \frac{\sqrt{N_i/U_i}}{64\tilde{M}} \geq \frac{1/64}{\frac{\beta}{d_i} + \frac{1}{K}} \geq \frac{1}{128\beta} d_i \geq d_i.$$

Finally,

$$s_i t \leq \frac{\sqrt{N_i/U_i}}{16\tilde{M}} \leq K/16 \leq \lfloor K/2 \rfloor .$$

Thus, all the parameters satisfy their conditions.

We now move to the comparisons in (A.3) which allow us to give bounds on A_i for all i.

Consider $h \in H$. Note that:

$$\begin{aligned}
& bs_h(s_h t - d_h + 1) \\
& \quad \leq bs_h^2 t \\
& \quad \leq 8\tilde{M}\sqrt{N_1/U_1} \cdot \left[\frac{2K\tilde{M}}{\sqrt{N_1/U_1}}\right]^2 \cdot \frac{\sqrt{N_1/U_1}}{32\tilde{M}} \\
& \quad = \left[K\tilde{M}\right]^2 \\
& \quad \leq \frac{N_h}{U_h}.
\end{aligned}$$

Therefore,

$$\begin{aligned}
A_h &= (s_h t - d_h + 1)U_h \\
&\geq \left[\left(\frac{K\tilde{M}}{\sqrt{N_1/U_1}}\right)\left(\frac{\sqrt{N_1/U_1}}{32\tilde{M}} - 1\right) - (d_h - 1)\right] U_h \\
&\geq \left[\left(\frac{1}{32} - \frac{\tilde{M}}{\sqrt{N_1/U_1}}\right) - \frac{D-1}{k_0}\right] KU_h \\
&\overset{(a)}{\geq} \left[\frac{1}{32} - \left(\frac{\beta}{d_1} + \frac{1}{K}\right) - \frac{D}{k_0} - \frac{1}{k_0}\right] KU_h \\
&\overset{(b)}{\geq} \left(\frac{1}{32} - 2\beta\right) \cdot KU_h.
\end{aligned}$$

Here, (a) uses Definition 1 with (2.10) and $1 \in I \setminus I_1$, and (b) uses $d_1 \geq 1$ and $D/k_0 = \beta$.

Now consider $i \in I$. We have:

$$\begin{aligned} bs_i(s_i t - d_i + 1) &\leq bs_i^2 t \\ &\leq 8\tilde{M}\sqrt{N_1/U_1} \cdot 4\frac{N_i/U_i}{N_1/U_1} \cdot \frac{\sqrt{N_1/U_1}}{32\tilde{M}} \\ &= \frac{N_i}{U_i} \end{aligned}$$

Therefore,

$$\begin{aligned} A_i &= (s_i t - d_i + 1)U_i \\ &\geq \left[\sqrt{\frac{N_i/U_i}{N_1/U_1}} \cdot \left(\frac{\sqrt{N_1/U_1}}{32\tilde{M}} - 1\right) - (d_i - 1)\right] U_i \\ &= \left[\frac{1}{32} - \frac{\tilde{M}}{\sqrt{N_1/U_1}} - (d_i - 1) \cdot \frac{\tilde{M}}{\sqrt{N_i/U_i}}\right] \frac{\sqrt{N_i U_i}}{\tilde{M}} \\ &\geq \left[\frac{1}{32} - \left(\frac{\beta}{d_1} + \frac{1}{K}\right) - (d_i - 1) \cdot \left(\frac{\beta}{d_i} + \frac{1}{K}\right)\right] \frac{\sqrt{N_i U_i}}{\tilde{M}} \\ &= \left[\frac{1}{32} - 2\beta - \left(\frac{1}{d_1} - \frac{1}{d_i}\right)\beta\right] \frac{\sqrt{N_i U_i}}{\tilde{M}} \\ &\geq \left(\frac{1}{32} - 3\beta\right) \frac{\sqrt{N_i U_i}}{\tilde{M}}. \end{aligned}$$

Finally, we will also give an upper bound to the following useful quantity:

$$\begin{aligned} \frac{tM}{b} &\leq \frac{\sqrt{N_1/U_1}}{32\tilde{M}} \cdot \frac{2}{8\tilde{M}\sqrt{N_1/U_1}} \cdot M \\ &\leq \frac{S_I^2}{128(M + V_I)^2} \cdot (M + V_I) \\ &= \frac{S_I^2}{128(M + V_I)}. \end{aligned}$$

Combining the A_i terms, along with the λ_i factors, we get the following lower bound:

$$\begin{aligned}
R^*(M) &\geq \sum_{h\in H} \frac{1}{2}\cdot\left(\frac{1}{32}-2\beta\right)\cdot KU_h \\
&\quad + \sum_{i\in I} \frac{1}{2}\cdot\left(\frac{1}{32}-3\beta\right)\cdot\frac{\sqrt{N_iU_i}}{\tilde{M}} \\
&\quad - \frac{tM}{b} \\
&\geq \sum_{h\in H} \frac{1}{2}\cdot\left(\frac{1}{32}-2\beta\right)\cdot KU_h \\
&\quad + \frac{1}{2}\cdot\left(\frac{1}{32}-3\beta\right)\cdot\frac{S_I^2}{M+V_I} \\
&\quad - \frac{S_I^2}{128(M+V_I)} \\
&= \sum_{h\in H} \frac{1}{2}\cdot\left(\frac{1}{32}-2\beta\right)\cdot KU_h \\
&\quad + \left[\frac{1}{2}\cdot\left(\frac{1}{32}-3\beta\right)-\frac{1}{128}\right]\cdot\frac{S_I^2}{M+V_I}. \\
&= \sum_{h\in H} \frac{1}{64}(1-64\beta)\cdot KU_h \\
&\quad + \frac{1}{256}(1-192\beta)\cdot\frac{2S_I^2}{M+V_I}. \qquad \text{(A.11)}
\end{aligned}$$

Combining (A.11) with (A.8):

$$\frac{R(M)}{R^*(M)} \leq \frac{256}{1-192\beta} = 8448. \qquad \text{(A.12)}$$

Case 2: $I_1 \neq \emptyset$

In this section, there exists at least one level in I_1. However, by Proposition 2, the set I_1 cannot contain more than one level. Therefore, in this section we have $I_1 = \{i_1\}$ for some level i_1.

Using Lemma 3, we can write the achievable rate as follows:

$$\begin{aligned} R(M) \leq & \sum_{h\in H} KU_h + \frac{2S_I S_{I\setminus I_1}}{M - T_J + V_I} \\ & + \frac{1}{\beta} d_{i_1} U_{i_1} \left(1 - \frac{M - T_J}{N_{i_1}/d_{i_1}}\right) \\ & + \frac{1}{\beta} d_{i_1} U_{i_1} \cdot \frac{S_{I\setminus I_1}}{\sqrt{N_{i_1} U_{i_1}}}. \end{aligned} \tag{A.13}$$

Let $\gamma_1 = 2.965$ and $\gamma_2 = 0.482$. Consider Lemma 5 with the following parameters.

$$\begin{aligned} & & t &= 1; \\ \forall h \in H, & & s_h &= \lfloor 2\beta K \rfloor; \\ \forall i \in I \setminus I_1, & & s_i &= \left\lfloor \gamma_1 \beta \frac{\sqrt{N_i/U_i}}{\tilde{M}} + \gamma_2 d_{i_1} \sqrt{\frac{U_{i_1}/N_{i_1}}{U_i/N_i}} \right\rfloor; \\ & & s_{i_1} &= d_{i_1}; \\ \forall j \in J, & & s_j &= d_j; \\ & & b &= \left\lceil \frac{N_{i_1}}{d_{i_1} U_{i_1}} \right\rceil. \end{aligned}$$

Thus we have $\lambda_{i_1} = \lambda_j = 1$ for all $j \in J$, and $\lambda_i, \lambda_h \geq \frac{1}{2}$ for $i \in I$ and $h \in H$.

Recall the following inequalities from Definition 1 regarding $i \in I$:

$$\frac{1}{K}\sqrt{\frac{N_i}{U_i}} \leq \tilde{M} \leq \left(\frac{1}{K} + \frac{1}{d_i}\right)\sqrt{\frac{N_i}{U_i}}.$$

Thus, for any $i, j \in I$, we have:

$$\sqrt{\frac{U_i/N_i}{U_j/N_j}} \leq \frac{K}{d_i} + 1.$$

Moreover, for $h \in H$ and $i_1 \in I_1$:

$$\left(\frac{1}{K} + \frac{\beta}{d_{i_1}}\right) \leq \tilde{M} \leq \frac{1}{K}\sqrt{\frac{N_h}{U_h}},$$

and hence:

$$\sqrt{\frac{U_{i_1}/N_{i_1}}{U_h/N_h}} \geq \beta \frac{K}{d_{i_1}} + 1.$$

We will now verify that the parameters satisfy their constraints. The parameters t, s_{i_1}, and s_j trivially satisfy all their constraints. Moreover, $N_{i_1} \geq KU_{i_1} \geq d_{i_1}U_{i_1}$, which implies $b \geq 1$. Regarding s_h:

$$K/2 \geq 2\beta K \geq 2\beta \cdot D/\beta = 2D \geq d_h,$$

and hence $d_h \leq s_h \leq \lfloor K/2 \rfloor$. We are hence only left with s_i.

Consider the following:

$$\begin{aligned}
&\gamma_1\beta\frac{\sqrt{N_i/U_i}}{\tilde{M}} + \gamma_2 d_{i_1}\sqrt{\frac{U_{i_1}/N_{i_1}}{U_i/N_i}} \\
&\geq \gamma_1\beta\frac{\sqrt{N_i/U_i}}{\tilde{M}} \geq \frac{\gamma_1\beta}{\beta/d_i + 1/K} \\
&\geq \frac{\gamma_1\beta}{2\beta}d_i \geq d_i,
\end{aligned}$$

and hence $s_i \geq d_i$. Furthermore,

$$\begin{aligned}
&\gamma_1\beta\frac{\sqrt{N_i/U_i}}{\tilde{M}} + \gamma_2 d_{i_1}\sqrt{\frac{U_{i_1}/N_{i_1}}{U_i/N_i}} \\
&\leq \gamma_1\beta K + \gamma_2 d_{i_1}\left(\frac{K}{d_{i_1}} + 1\right) \\
&= \left[\gamma_1\beta + \gamma_2\left(1 + \frac{d_{i_1}}{K}\right)\right]K \\
&\leq \left[\gamma_2 + (\gamma_1 + \gamma_2)\beta\right]K \\
&\leq K/2,
\end{aligned}$$

and hence $s_i = s_i t \leq \lfloor K/2 \rfloor$.

We will now evaluate the comparisons in (A.3) to give bounds on the A_i terms, for every i.

Let us start with $h \in H$. We have $bs_h(s_h t - d_h + 1) \leq bs_h^2 t = bs_h^2$, and:

$$\begin{aligned} bs_h^2 &\leq \left(\frac{N_{i_1}}{d_{i_1}U_{i_1}} + 1\right) \cdot 4\beta^2 K^2 \\ &\leq \frac{N_{i_1}}{d_{i_1}U_{i_1}} \left(1 + \frac{d_{i_1}}{K}\right) \cdot 4\beta^2 K^2 \\ &\overset{(a)}{\leq} \frac{N_h}{d_{i_1}U_h} \frac{1 + d_{i_1}/K}{(\beta + d_{i_1}/K)^2} \cdot \frac{d_{i_1}^2}{K^2} \cdot 4\beta^2 K^2 \\ &\overset{(b)}{\leq} \frac{N_h}{U_h} \cdot \frac{1}{\beta^2} \cdot d_{i_1} \cdot 4\beta^2 \\ &\leq 4D \cdot \frac{N_h}{U_h}. \end{aligned}$$

The inequality labeled (a) comes from Definition 1 and (2.10), and:

$$\left(\frac{\beta}{d_{i_1}} + \frac{1}{K}\right) \sqrt{\frac{N_{i_1}}{U_{i_1}}} < M \leq \frac{1}{K} \sqrt{\frac{N_h}{U_h}}.$$

As for the inequality (b), it is due to:

$$\forall x \geq 0, \quad \frac{1+x}{(\beta+x)^2} = \frac{1}{\beta+x} \cdot \frac{1+x}{\beta+x} \leq \frac{1}{\beta} \cdot \frac{1}{\beta},$$

because $x \to \frac{1+x}{\beta+x}$ is decreasing for $x \geq 0$. Hence,

$$\begin{aligned} A_h &\geq \min\left\{1, \frac{1}{4D}\right\} \cdot (s_h t - d_h + 1) U_h \\ &\geq \frac{1}{4D} \cdot (2\beta K - d_h)\, U_h \\ &\geq \frac{1}{4D} \cdot (2\beta - \beta)\, K U_h \\ &= \frac{\beta}{4D} \cdot K U_h. \end{aligned}$$

Now we consider $i \in I \setminus I_1$. We have:

$$
\begin{aligned}
& bs_i^2 \\
\leq & \frac{N_{i_1}}{d_{i_1} U_{i_1}} \left(1 + \frac{d_{i_1}}{K}\right) \left(\gamma_1 \beta \frac{\sqrt{N_i/U_i}}{\tilde{M}} + \gamma_2 d_{i_1} \sqrt{\frac{U_{i_1}/N_{i_1}}{U_i/N_i}}\right)^2 \\
\leq & \frac{N_{i_1}}{d_{i_1} U_{i_1}} (1+\beta) \left(\gamma_1 \beta \frac{\sqrt{N_{i_1}/U_{i_1}}}{\tilde{M}} + \gamma_2 d_{i_1}\right)^2 \cdot \frac{U_{i_1}/N_{i_1}}{U_i/N_i} \\
\leq & \frac{N_i}{d_{i_1} U_i} (1+\beta) \left(\frac{\gamma_1 \beta}{\beta/d_{i_1} + 1/K} + \gamma_2 d_{i_1}\right)^2 \\
\leq & \frac{N_i}{d_{i_1} U_i} (1+\beta) \left(\frac{\gamma_1 \beta}{\beta} d_{i_1} + \gamma_2 d_{i_1}\right)^2 \\
= & \frac{N_i}{d_{i_1} U_i} (1+\beta) d_{i_1}^2 \left(\gamma_1 + \gamma_2\right)^2 \\
\leq & \frac{N_i}{U_i} \cdot 12(1+\beta) D.
\end{aligned}
$$

Therefore, if we define $c = 1/12(1+\beta)D$, then:

$$
\begin{aligned}
& A_i \\
\geq & c \cdot (s_i t - d_i + 1) U_i \\
\geq & c \cdot \left(\gamma_1 \beta \frac{\sqrt{N_i/U_i}}{\tilde{M}} + \gamma_2 d_{i_1} \sqrt{\frac{U_{i_1}/N_{i_1}}{U_i/N_i}} - d_i\right) U_i \\
= & c \left[\left(\gamma_1 \beta \frac{\sqrt{N_i/U_i}}{\tilde{M}} - d_i\right) U_i + \gamma_2 d_{i_1} \sqrt{\frac{U_{i_1}/N_{i_1}}{U_i/N_i}} \cdot U_i\right] \\
= & c \left[\left(\gamma_1 \beta - d_i \frac{\tilde{M}}{\sqrt{N_i/U_i}}\right) \frac{\sqrt{N_i U_i}}{\tilde{M}} + \gamma_2 d_{i_1} U_{i_1} \frac{\sqrt{N_i U_i}}{\sqrt{N_{i_1} U_{i_1}}}\right] \\
\geq & c \left[\left(\gamma_1 \beta - 2\beta\right) \frac{\sqrt{N_i U_i}}{\tilde{M}} + \gamma_2 d_{i_1} U_{i_1} \frac{\sqrt{N_i U_i}}{\sqrt{N_{i_1} U_{i_1}}}\right] \\
= & c \left[(\gamma_1 - 2)\beta \frac{\sqrt{N_i U_i}}{\tilde{M}} + \gamma_2 d_{i_1} U_{i_1} \frac{\sqrt{N_i U_i}}{\sqrt{N_{i_1} U_{i_1}}}\right] \\
\geq & c \cdot \min\left\{\frac{1}{2}\gamma_1 - 1, \gamma_2\right\} \cdot \beta \\
& \cdot \left[\frac{2\sqrt{N_i U_i}}{\tilde{M}} + \frac{1}{\beta} d_{i_1} U_{i_1} \frac{\sqrt{N_i U_i}}{\sqrt{N_{i_1} U_{i_1}}}\right] \\
= & 0.482 c\beta \cdot \left[\frac{2\sqrt{N_i U_i}}{\tilde{M}} + \frac{1}{\beta} d_{i_1} U_{i_1} \frac{\sqrt{N_i U_i}}{\sqrt{N_{i_1} U_{i_1}}}\right].
\end{aligned}
$$

Finally, we have:

$$bs_{i_1}(s_{i_1}t - d_{i_1} + 1) = bd_{i_1} \geq \frac{N_{i_1}}{U_{i_1}},$$

and, likewise for $j \in J$:

$$bs_j(s_j t - d_j + 1) = bd_j \geq \frac{d_j}{d_{i_1}} \frac{N_{i_1}}{U_{i_1}} \geq \frac{N_j}{U_j},$$

because of the regularity condition (2.4). Thus:

$$A_{i_1} = \frac{N_{i_1}/d_{i_1}}{b}; \quad A_j = \frac{N_j/d_j}{b}.$$

Combining the A_i terms, and keeping in mind the λ_i terms, we get:

$$\begin{aligned}
& R^*(M) \\
\geq & \sum_{h \in H} \frac{1}{2} \cdot \frac{\beta}{4D} \cdot KU_h \\
& + \sum_{i \in I \setminus I_1} \frac{1}{2} \cdot \frac{0.482\beta}{12(1+\beta)D} \cdot \left[\frac{2\sqrt{N_i U_i}}{\tilde{M}} + \frac{1}{\beta} d_{i_1} U_{i_1} \frac{\sqrt{N_i U_i}}{N_{i_1} U_{i_1}} \right] \\
& + \frac{N_{i_1}/d_{i_1} + \sum_{j \in J} N_j/d_j - M}{b} \\
\geq & \sum_{h \in H} \frac{\beta}{8D} \cdot KU_h \\
& + \frac{0.241\beta}{12(1+\beta)D} \left[\frac{2S_I S_{I \setminus I_1}}{M - T_J + V_I} + \frac{1}{\beta} d_{i_1} U_{i_1} \frac{S_{I \setminus I_1}}{N_{i_1} U_{i_1}} \right] \\
& + \frac{N_{i_1}/d_{i_1} + \sum_{j \in J} N_j/d_j - M}{2N_{i_1}/(d_{i_1} U_{i_1})} \\
= & \sum_{h \in H} \frac{\beta}{8D} \cdot KU_h \\
& + \frac{0.241\beta}{12(1+\beta)D} \left[\frac{2S_I S_{I \setminus I_1}}{M - T_J + V_I} + \frac{1}{\beta} d_{i_1} U_{i_1} \frac{S_{I \setminus I_1}}{N_{i_1} U_{i_1}} \right] \\
& + \frac{1}{2} U_{i_1} \left(1 - \frac{M - T_J}{N_{i_1}/d_{i_1}} \right).
\end{aligned} \tag{A.14}$$

Combining (A.14) with (A.13), we get:

$$\frac{R(M)}{R^*(M)} \leq \max \left\{ \frac{8D}{\beta}, \frac{12(1+\beta)D}{0.241\beta}, \frac{2D}{\beta} \right\} \leq 9909D. \tag{A.15}$$

Final gap

By combining the inequalities from all the cases above, i.e., inequalities (A.7), (A.10), (A.12), and (A.15), we get the following final result:

$$\frac{R(M)}{R^*(M)} \leq 9909 \cdot D,$$

which concludes the proof of Theorem 4.

A.1.3 Proofs of some useful results

Proof of Lemma 3. We will prove the lemma for each of the five sets in the refined M-feasible partition. First, recall that the achievable rate $R^{\mathrm{SL}}(\cdot)$ for the single-level setup is defined in Theorem 1.

For $h \in H$:

$$R_h(M) = R^{\mathrm{SL}}(0, K, N_h, U_h, d_h) = KU_h.$$

For $i \in I_0$:

$$\begin{aligned} R_i(M) &= R^{\mathrm{SL}}(\alpha_i M, K, N_i, U_i, d_i) \\ &\leq R^{\mathrm{SL}}(0, K, N_i, U_i, d_i) \\ &= KU_i \\ &\leq \frac{2}{\tilde{M}} \sqrt{\frac{N_i}{U_i}} \cdot U_i \\ &= \frac{2S_I \sqrt{N_i U_i}}{M - T_J + V_I}. \end{aligned}$$

$$
\begin{aligned}
\Delta &= \left(1-\frac{\alpha_i M}{N_i/d_i}\right)-\left(1-\frac{M-T_J}{N_i/d_i}\right)=\left(1-\frac{\frac{\sqrt{N_iU_i}}{S_I}(M-T_J+V_I)-\frac{N_i}{K}}{N_i/d_i}\right)-\left(1-\frac{M-T_J}{N_i/d_i}\right)\\
&= \frac{M-T_J}{N_i/d_i}-\frac{\sqrt{N_iU_i}(M-T_J)}{S_IN_i/d_i}-\frac{\sqrt{N_iU_i}V_I}{S_IN_i/d_i}+\frac{d_i}{K}\\
&= \frac{M-T_J}{N_i/d_i}\left(1-\frac{\sqrt{N_iU_i}}{S_I}\right)-\frac{\sqrt{N_iU_i}V_{I\setminus I_1}}{S_IN_i/d_i}-\frac{\sqrt{N_iU_i}\cdot\frac{N_i}{K}}{S_IN_i/d_i}+\frac{d_i}{K}\\
&= \frac{M-T_J}{N_i/d_i}\cdot\frac{S_{I\setminus I_1}}{S_I}-\frac{d_iV_{I\setminus I_1}}{\sqrt{N_i/U_i}S_I}+\frac{d_i}{K}\left(1-\frac{\sqrt{N_iU_i}}{S_I}\right)\\
&= \frac{d_i}{N_i}(M-T_J)\frac{S_{I\setminus I_1}}{S_I}-\frac{d_iV_{I\setminus I_1}}{\sqrt{N_i/U_i}S_I}+\frac{d_i}{K}\cdot\frac{S_{I\setminus I_1}}{S_I}\\
&\overset{(a)}{\le} \frac{d_i}{N_i}\left[\left(\frac{1}{K}+\frac{1}{d_i}\right)\sqrt{\frac{N_i}{U_i}}S_I-V_I\right]\frac{S_{I\setminus I_1}}{S_I}-\frac{d_iV_{I\setminus I_1}}{\sqrt{N_i/U_i}S_I}+\frac{d_i}{K}\cdot\frac{S_{I\setminus I_1}}{S_I}\\
&= \left(\frac{d_i}{K}+1\right)\frac{S_{I\setminus I_1}}{\sqrt{N_iU_i}}-\frac{d_iV_IS_{I\setminus I_1}}{N_iS_I}-\frac{d_iV_{I\setminus I_1}\sqrt{N_iU_i}}{N_iS_I}+\frac{d_iS_{I\setminus I_1}}{KS_I}\\
&= \left(\frac{d_i}{K}+1\right)\frac{S_{I\setminus I_1}}{\sqrt{N_iU_i}}-\frac{d_i\left(V_{I\setminus I_1}+\frac{N_i}{K}\right)S_{I\setminus I_1}}{N_iS_I}-\frac{d_iV_{I\setminus I_1}\sqrt{N_iU_i}}{N_iS_I}+\frac{d_iS_{I\setminus I_1}}{KS_I}\\
&= \left(\frac{d_i}{K}+1\right)\frac{S_{I\setminus I_1}}{\sqrt{N_iU_i}}-\frac{d_iV_{I\setminus I_1}S_I}{N_iS_I}-\frac{d_iS_{I\setminus I_1}}{KS_I}+\frac{d_iS_{I\setminus I_1}}{KS_I}=\frac{d_i}{K}\cdot\frac{S_{I\setminus I_1}}{\sqrt{N_iU_i}}-\frac{d_iV_{I\setminus I_1}}{N_i}+\frac{S_{I\setminus I_1}}{\sqrt{N_iU_i}}\\
&= \frac{d_i}{K}\sum_{i\in I\setminus I_1}\left(\frac{\sqrt{N_iU_i}}{\sqrt{N_iU_i}}-\frac{N_i}{N_i}\right)+\frac{S_{I\setminus I_1}}{\sqrt{N_iU_i}}=\frac{d_i}{K}\sum_{i\in I\setminus I_1}\frac{\sqrt{N_iU_i}+N_i\sqrt{U_i/N_i}}{\sqrt{N_iU_i}}+\frac{S_{I\setminus I_1}}{\sqrt{N_iU_i}}\\
&= \frac{d_i}{K}\sum_{i\in I\setminus I_1}\frac{N_i\left(\sqrt{U_i/N_i}-\sqrt{N_i/U_i}\right)}{\sqrt{N_iU_i}}+\frac{S_{I\setminus I_1}}{\sqrt{N_iU_i}}\\
&\overset{(b)}{\le} \frac{S_{I\setminus I_1}}{\sqrt{N_iU_i}}.
\end{aligned}
\tag{A.16}
$$

For $i\in I'$:

$$
\begin{aligned}
R_i(M) &= R^{\mathrm{SL}}(\alpha_iM,K,N_i,U_i,d_i)\\
&\le U_i\cdot\frac{N_i}{\alpha_iM}\cdot\left(1-\frac{\alpha_iM}{N_i/d_i}\right)\\
&\le \frac{N_iU_i}{\alpha_iM}\\
&= \frac{N_iU_i}{\sqrt{N_iU_i}\cdot\tilde{M}-N_i/K}\\
&\le \frac{N_iU_i}{\sqrt{N_iU_i}\cdot\tilde{M}-\frac{1}{2}\sqrt{N_iU_i}\cdot\tilde{M}}\\
&= \frac{2S_I\sqrt{N_iU_i}}{M-T_J+V_I}.
\end{aligned}
$$

For $i \in I_1$:

$$R_i(M) \leq U_i \cdot \frac{N_i}{\alpha_i M} \cdot \left(1 - \frac{\alpha_i M}{N_i/d_i}\right) \leq \frac{1}{\beta} d_i U_i \left(1 - \frac{\alpha_i M}{N_i/d_i}\right),$$

since $i \in I_1 \implies \alpha_i M \geq \beta N_i/d_i$ (follows from (2.10)).

Consider the following difference:

$$\Delta = \left(1 - \frac{\alpha_i M}{N_i/d_i}\right) - \left(1 - \frac{M - T_J}{N_i/d_i}\right).$$

We can show that:

$$\Delta \leq \frac{S_{I \setminus I_1}}{\sqrt{N_i U_i}}.$$

The full derivation is given in (A.16). In the calculations in (A.16), (a) is due to the definition of I_1, while (b) is due to the fact that the (unique) level in I_1 is more popular than any other level in $i' \in I$, and thus:

$$\sqrt{\frac{U_{i'}}{N_{i'}}} \leq \sqrt{\frac{U_i}{N_i}}.$$

Using the bound on Δ, we can rewrite $R_i(M)$ as follows:

$$\begin{aligned} R_i(M) &\leq \frac{1}{\beta} d_i U_i \left(1 - \frac{\alpha_i M}{N_i/d_i}\right) \\ &= \frac{1}{\beta} d_i U_i \left(1 - \frac{M - T_J}{N_i/d_i} + \Delta\right) \\ &\leq \frac{1}{\beta} d_i U_i \left(1 - \frac{M - T_J}{N_i/d_i}\right) + \frac{1}{\beta} d_i U_i \cdot \frac{S_{I_0} + S_{I'}}{\sqrt{N_i U_i}}. \end{aligned}$$

Finally, for $j \in J$:

$$R_j(M) = R^{\mathrm{SL}}(N_j/d_j, K, N_j, U_j, d_j) = 0.$$

This completes the proof of the lemma. □

Proof of Lemma 2. We prove the existence of an M-feasible partition by construction. In fact, we will use Algorithm 1 presented in Section 2.5.1 as the proof. In what follows, we prove the correctness of the algorithm, and then conclude the proof of Lemma 2.

The idea behind the algorithm is as follows. When the memory is 0, all levels are in the set H.[1]

[1]Technically, there will be exactly one level in the set I, but it will still get zero memory so this distinction is

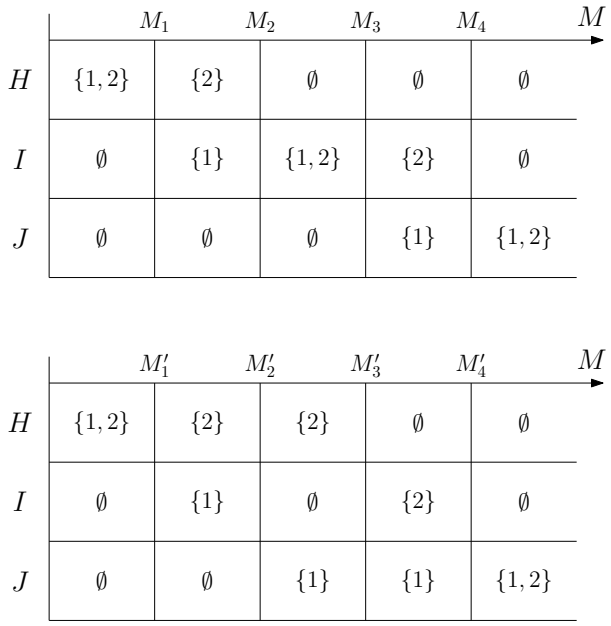


Figure A.1: Two examples of how an M-feasible partition (H, I, J) could evolve when there are two levels $\{1, 2\}$. In the second case, it will turn out that $M'_2 = M'_3$. The reasoning behind this is that between M'_2 and M'_3, all levels have a fixed memory: level 2 has memory 0 while level 1 has memory N_1/d_1. Thus an increase in the overall memory between M'_2 and M'_3 would be wasted. Moreover, $M_1 = M'_1 = 0$ for similar reasons, and $M_4 = M'_4 = N_1/d_1 + N_2/d_2$ because that is the point at which both levels can be completely stored.

As the memory is increased, levels will start moving from H to I as they gain more memory, and then finally from I to J as they get completely stored in the caches. Thus, the range of memory values $[0, \infty)$ can be divided into $2L + 1$ intervals; in each interval, the partition (H, I, J) will be the same. The boundaries of these intervals are the memory values at which levels switch from one set to the next. Figure A.1 shows two examples of how the (H, I, J) partition might evolve with two levels.

The algorithm operates in three main steps. In the first step, the sequence of (H, I, J) partitions is determined. For instance, we determine which of the two cases illustrated in Figure A.1 holds. In the second step, the values of the boundaries between the different intervals are calculated; in Figure A.1, those would be $\{M_1, M_2, M_3, M_4\}$ or $\{M'_1, M'_2, M'_3, M'_4\}$. Finally, the third step consists in determining the M-feasible partition (H, I, J) based on steps 1 and 2, and based on the value of any given memory M. For example, if we are in the first case in Figure A.1 and we are given a memory value between M_3 and M_4, then we know that $(H, I, J) = (\emptyset, \{2\}, \{1\})$.

irrelevant.

Recall from Definition 1 that the M-feasible partition is defined by the relation of $(M - T_J + V_I)/S_I$ to the two values $(1/K)\sqrt{N_i/U_i}$ and $(1/d_i + 1/K)\sqrt{N_i/U_i}$, for every level i. As long as the same relations are maintained, the same partition is chosen as the M-feasible partition. Thus, the values of the boundary ranges must be of one of two forms:

$$m_i^{I,J} = \frac{1}{K}\sqrt{\frac{N_i}{U_i}} \cdot S_I + T_J - V_I;$$
$$M_i^{I,J} = \left(\frac{1}{d_i} + \frac{1}{K}\right)\sqrt{\frac{N_i}{U_i}} \cdot S_I + T_J - V_I,$$

for the appropriate partition (H, I, J). In fact, this partition has to be the one defined by either of the intervals that this boundary divides. For example, M_2 in Figure A.1 is the boundary where level 2 moves from H to I. Thus, it is of the form $m_2^{I,J}$, where $(H, I, J) = (\emptyset, \{1, 2\}, \emptyset)$, or $(H, I, J) = (\{2\}, \{1\}, \emptyset)$. As it turns out, both choices give the same result.

Let us suppose we are looking at a particular interval with some (H, I, J) partition, and we wish to know its upper boundary. We know that it has to be either $m_h^{I,J}$, $h \in H$, or $M_i^{I,J}$, $i \in I$. Specifically, it has to be the smallest of these values. Since I and J are fixed, this is equivalent to determining the smaller of:

$$\tilde{m}_h = \frac{1}{K}\sqrt{\frac{N_h}{U_h}};$$
$$\tilde{M}_i = \left(\frac{1}{d_i} + \frac{1}{K}\right)\sqrt{\frac{N_i}{U_i}}.$$

But these values do not depend on the chosen partition (H, I, J)! Thus, the sequence of (H, I, J) can be uniquely determined solely by these $\tilde{m}_i$ and $\tilde{M}_i$ values.

For what follows, we refer the reader to Algorithm 1 for the various symbols (x_t, Y_t). It is worth mentioning the following two points. First, the ordering between x_t and Y_t is preserved. That is, $x_t < x_{t'} \implies Y_t < Y_{t'}$. This is ensured by the observation made above that the ordering does not depend on the partition (H, I, J). Second, as mentioned earlier, when applying the function $x \mapsto x \cdot S_I + T_J - V_I$ on some $x = \tilde{m}_i$ or $x = \tilde{M}_i$, it does not matter whether we take the (H, I, J) partition whose interval x lower-bounds or upper-bounds.

Any interval in which I is set to be empty automatically becomes an empty interval. To prove

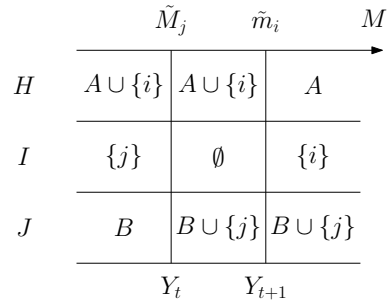


Figure A.2: The only situation where Algorithm 1 results in an interval with $I = \emptyset$. Here, the sets A, B, $\{i\}$, and $\{j\}$ are disjoint and together form the entire set of levels.

this, we will show that $I = \emptyset$ implies that its upper and lower boundaries match. This is done as follows. Consider the aforementioned interval, and notice that it can only occur if some level moved from I to J at the lower boundary, and another level moved from H to I at the upper boundary. See the bottom case of Figure A.1 for an example, interval (M_2', M_3'). Thus, the sequence of (H, I, J) partitions that occurs is of the form shown in Figure A.2.

Let us compute the interval boundaries Y_t and Y_{t+1}. Because of our previous observations, we can use $(H, I, J) = (A \cup \{i\}, \emptyset, B \cup \{j\})$ for both. Then,

$$Y_t = \tilde{M}_j \cdot S_I + T_J - V_I = T_J = T_B + N_j/d_j.$$

Similarly,

$$Y_{t+1} = \tilde{m}_i \cdot S_I + T_J - V_I = T_J = T_B + N_j/d_j.$$

Thus, $Y_t = Y_{t+1}$ and the interval is empty.

This concludes the proof of the lemma. □

A.2 Proof of approximate optimality for the single-user setup (Theorem 6)

As discussed in Section 2.6.2, we prove Theorem 6 by lower-bounding the optimal rate with expressions of the form shown in (2.16). Two cases must be considered.

Case $M < 1/6$

When M is this small, we choose $b = 1$ broadcast message. Recall that, because of regularity condition (2.3), we have $N_i/K_i \geq 1 > 1/6 > M$. The achievable rate in this case can be upper-bounded by the expression in (2.17).

Consider now any level i. Let $s_i = K_i$. Then,

$$\begin{aligned} v_i &= s_i \left(\min\left\{1, \frac{N_i}{s_i b}\right\} - \frac{M}{b} \right) \\ &= K_i \left(\min\left\{1, \frac{N_i}{K_i}\right\} - M \right) \\ &\geq (5/6) K_i. \end{aligned} \tag{A.17}$$

We can combine (2.17) with (A.17) and (2.16) to get:

$$\frac{R(M)}{R^*(M)} \leq \frac{6}{5}. \tag{A.18}$$

Case $M \geq 1/6$

We will now choose $b = \lceil 6M \rceil \geq 1$.

Bound for $g \in G$ Consider $s_g = 1$. Then,

$$\begin{aligned} v_g &= \min\left\{1, \frac{N_i}{\lceil 6M \rceil}\right\} - \frac{M}{\lceil 6M \rceil} \\ &\geq \frac{1}{2} - \frac{M}{6M} \\ &= \frac{1}{3}, \end{aligned} \tag{A.19}$$

because $s_g b = \lceil 6M \rceil \leq 2 \cdot 6M \leq 2N_g$.

Bound for $h \in H$ Consider $s_h = \lceil K_h/6 \rceil \geq 1$. Then,

$$v_h \geq \frac{K_h}{6} \left(\min\left\{1, \frac{N_h}{4K_h M}\right\} - \frac{M}{6M} \right) = \frac{1}{72} \cdot K_h, \tag{A.20}$$

because $s_h b = \lceil K_h/6 \rceil \cdot \lceil 6M \rceil \leq 4K_h M \leq 4N_h$.

Bound for $i \in I$ Consider $s_i = \lceil N_i/6M \rceil$. Then,

$$v_i \geq \frac{N_i}{6M} \left(\min \left\{ 1, \frac{N_i}{4N_i} \right\} - \frac{M}{6M} \right) = \frac{1}{72} \cdot \frac{N_i}{M}, \tag{A.21}$$

because $s_i b = \lceil N_i/6M \rceil \cdot \lceil 6M \rceil \leq 4N_i$.

Bound for J First, if $M \geq N_J$, then the set J contributes nothing to the upper bound on the rate in (2.14); see (2.15). Thus we can ignore it, i.e., choose $s_J = 0$ and thus $n_J = 0$ and $v_J = 0$.

So the interesting case is $M < N_J$. Here, we must decode files from multiple levels collectively. Consider $s_J = \lceil N_J/6M \rceil$, where $N_J = \sum_{j \in J} N_j$. Notice that there are enough users and broadcasts to decode all files, because:

$$s_J b \geq \frac{N_J}{6M} \cdot 6M = N_J.$$

However, we must take care that no broadcast considers more than K_j users at a time for any $j \in J$. This can be ensured: since there are $b = \lceil 6M \rceil$ broadcasts, and $b \geq N_j$ for all $j \in J$, then every broadcast need only consider at most one user per level. Hence, all of the N_J files can be decoded, and $n_J = N_J$.

If $M < N_J/6$, we have:

$$\begin{aligned} v_J &= \frac{N_J - s_J M}{b} \\ &\geq \frac{1}{12M} \left(N_J - \frac{N_J}{12M} \cdot M \right) \\ &\geq \frac{144}{11} \cdot \frac{N_J}{M}. \end{aligned} \tag{A.22}$$

If $N_J/6 \leq M < N_J$, then s_J is actually equal to 1, and:

$$\begin{aligned} v_J &= \frac{N_J - M}{b} \\ &\geq \frac{N_J - M}{12M} \\ &\geq \frac{1}{12} \left(1 - \frac{M}{N_J} \right). \end{aligned} \tag{A.23}$$

Multiplicative gap

By combining (A.19), (A.20), (A.21), (A.22) and (A.23) with (2.14) and (2.15), and also taking into account (A.18), we get:

$$\frac{R(M)}{R^*(M)} \leq 72,$$

which concludes the proof of Theorem 6. □

A.3 Complete characterization for the small example (Theorem 2)

This appendix proves Theorem 2 in two parts. The first part presents the achievability scheme, and the second part gives the information-theoretic outer bounds.

The rate-memory curve shown in Figure 2.8—which we are trying to prove is both achievable and optimal—can be described using the following equation:

$$R = \max\left\{3 - 2M\,,\, \frac{5}{2} - M\,,\, 2 - \frac{1}{2}M\,,\, 1 - \frac{M-2}{N_2/2}\,,\, 0\right\}. \tag{A.24}$$

We would like to remind the reader that, in the example considered, the number of less popular files is $N_2 \geq 4$.

It will be useful in this discussion to define the *request vector* $\mathbf{r}$ to be a length-3 vector identifying the three files requested by the three users. Specifically, we write $\mathbf{r} = (r_1, r_2, r_3)$ to denote that users 1, 2, and 3 request files $W^1_{r_1}$, $W^1_{r_2}$, and $W^2_{r_3}$ respectively.

A.3.1 Achievable scheme

To prove the achievability of the piece-wise linear curve in Figure 2.8 and in (A.24), we need only show the achievability of the corner points, as the rest can be achieved using a memory-sharing scheme between every pair of points. These (M, R) points are:

$$(0,3)\,;\quad \left(\frac{1}{2},2\right);\quad \left(1,\frac{2}{3}\right);\quad (2,1)\,;\quad \text{and}\ \left(2+\frac{N_2}{2},0\right).$$

Point $(M, R) = (0, 3)$

When $M = 0$, we do not place any content in the caches, and instead broadcast all the requested files to the users. Since these can be three different files, the peak rate is $R = 3$.

Point $(M, R) = (2 + \frac{N_2}{2}, 0)$

On the other extreme, when $M = 2 + \frac{N_2}{2}$, the caches are large enough to each store all the popular files, and complementary halves of the unpopular files. Thus, a user can recover any popular file by solely accessing any cache, and can recover any unpopular file by accessing both caches. Therefore, no broadcast is needed, and thus $R = 0$ is achievable.

Point $(M, R) = (2, 1)$

In this case, each cache can hold up to two files. Our strategy is to dedicate this memory to the two popular files. Thus, each cache contains all the popular files, but no information about the unpopular files. Therefore, users 1 and 2 can recover their respective requested files without relying on any broadcast, whereas user 3 requires a full broadcast of his requested file. As a result, the rate $R = 1$ is achievable.

Point $(M, R) = (1, \frac{3}{2})$

Let us split each of the popular files into two parts of equal size: $W_1^1 = (W_{1a}^1, W_{1b}^1)$ and $W_2^1 = (W_{2a}^1, W_{2b}^1)$. We place these in the caches as follows. The first cache contains the first half of each file: $Z_1 = (W_{1a}^1, W_{2a}^1)$, whereas the second cache contains the second half of each file: $Z_2 = (W_{1b}^1, W_{2b}^1)$. When the user requests are revealed, the BS sends a common message with two parts: $X^{\mathbf{r}} = (X_1^{\mathbf{r}}, X_2^{\mathbf{r}})$. The first part directly serves user 3 by giving him the full file he requested: $X_1^{\mathbf{r}} = W_{r_3}^2$. The second part will be $X_2^{\mathbf{r}} = W_{r_1 b}^1 \oplus W_{r_2 a}^1$, which, along with the AP cache content accessed by each of users 1 and 2, allows them to recover their respective file. The BS had to transmit one full file, plus a linear combination of two half files, and, as a result, the total broadcast rate is $R = \frac{3}{2}$.

Point $(M, R) = (\frac{1}{2}, 2)$

This case is slightly different in that it requires storing coded content in the caches. We again split the popular files into two halves as before. However, we this time store the following: $Z_1 = W^1_{1a} \oplus W^1_{2a}$; and $Z_2 = W^1_{1b} \oplus W^1_{2b}$. The BS will transmit the following broadcast: $X^{\mathbf{r}} = \left(W^2_{r_3}, W^1_{r_1 b}, W^1_{r_2 a}\right)$. This will serve all the requests of the users. The broadcasts message consists of a full file and two half-files, and thus the total rate is $R = 2$.

A.3.2 Outer bounds

We will now prove that the above scheme is optimal with respect to information-theoretic bounds. We do that by showing that the rate is larger than each one of the expressions in the maximization in (A.24). The inequality $R \geq 0$ is trivial.

In all of the following, inequalities marked by $(*)$ are due to Fano's inequality, and the ε_F term that arises along with these inequalities is a term that decays to zero as $F \to \infty$.

First expression

Let the request vector be $\mathbf{r} = (1, 2, 1)$, and consider both caches along with the broadcast $X^{\mathbf{r}}$. Then, for all F,

$$\begin{aligned} RF + 2MF &\geq H\left(Z_1, Z_2, X^{\mathbf{r}}\right) \\ &= H\left(Z_1, Z_2, X^{(1,2,1)} \middle| W^1_1, W^1_2, W^2_1\right) \\ &\quad + I\left(W^1_1, W^1_2, W^2_1; Z_1, Z_2, X^{(1,2,1)}\right) \\ &\overset{(*)}{\geq} 3F(1 - \varepsilon_F) \\ R + 2M &\geq 3(1 - \varepsilon_F). \end{aligned}$$

By taking $F \to \infty$, we get:

$$R \geq 3 - 2M.$$

Second expression

Consider the two request vectors $\mathbf{r}_1 = (1, 2, 1)$ and $\mathbf{r}_2 = (2, 1, 2)$.

$$
\begin{aligned}
& 2\left(RF + MF\right) \\
& \qquad \geq H\left(Z_1, X^{\mathbf{r}_1}\right) + H\left(Z_2, X^{\mathbf{r}_2}\right) \\
& \qquad = H\left(Z_1, X^{\mathbf{r}_1} \middle| W_1^1\right) + I\left(W_1^1; Z_1, X^{\mathbf{r}_1}\right) \\
& \qquad\quad + H\left(Z_2, X^{\mathbf{r}_2} \middle| W_1^1\right) + I\left(W_1^1; Z_2, X^{\mathbf{r}_2}\right) \\
& \qquad \overset{(*)}{\geq} H\left(Z_1, Z_2, X^{\mathbf{r}_1}, X^{\mathbf{r}_2} \middle| W_1^1\right) + 2F(1 - \varepsilon_F) \\
& \qquad = H\left(Z_1, Z_2, X^{\mathbf{r}_1}, X^{\mathbf{r}_2} \middle| W_1^1, W_2^1, W_1^2, W_2^2\right) \\
& \qquad\quad + I\left(W_2^1, W_1^2, W_2^2; Z_1, Z_2, X^{\mathbf{r}_1}, X^{\mathbf{r}_2} \middle| W_1^1\right) \\
& \qquad\quad + 2F(1 - \varepsilon_F) \\
& \qquad \overset{(*)}{\geq} 5F(1 - \varepsilon_F) \\
& 2R + 2M \geq 5(1 - \varepsilon_F).
\end{aligned}
$$

By taking $F \to \infty$, we get:

$$R \geq \frac{5}{2} - M.$$

Third expression

Consider the following four request vectors: $\mathbf{r}_1 = (1,2,1)$, $\mathbf{r}_2 = (2,1,2)$, $\mathbf{r}_3 = (1,2,3)$, and $\mathbf{r}_1 = (2,1,4)$. For clarity, define $\mathcal{X} = (X^{\mathbf{r}_1}, X^{\mathbf{r}_2}, X^{\mathbf{r}_3}, X^{\mathbf{r}_4})$ and $\mathcal{W}^2 = (W_1^2, W_2^2, W_3^2, W_4^2)$.

$$\begin{aligned}
2(2RF + MF) &\geq H\left(Z_1, X^{\mathbf{r}_1}, X^{\mathbf{r}_2}\right) + H\left(Z_2, X^{\mathbf{r}_3}, X^{\mathbf{r}_4}\right) \\
&= H\left(Z_1, X^{\mathbf{r}_1}, X^{\mathbf{r}_2} \middle| W_1^1, W_2^1\right) \\
&\quad + I\left(W_1^1, W_2^1; Z_1, X^{\mathbf{r}_1}, X^{\mathbf{r}_2}\right) \\
&\quad + H\left(Z_2, X^{\mathbf{r}_3}, X^{\mathbf{r}_4} \middle| W_1^1, W_2^1\right) \\
&\quad + I\left(W_1^1, W_2^1; Z_2, X^{\mathbf{r}_3}, X^{\mathbf{r}_4}\right) \\
&\overset{(*)}{\geq} H\left(Z_1, Z_2, \mathcal{X} \middle| W_1^1, W_2^1\right) \\
&\quad + 2 \cdot 2F(1 - \varepsilon_F) \\
&= H\left(Z_1, Z_2, \mathcal{X} \middle| W_1^1, W_2^1, \mathcal{W}^2\right) \\
&\quad + I\left(\mathcal{W}^2; Z_1, Z_2, \mathcal{X} \middle| W_1^1, W_2^1\right) \\
&\quad + 4F(1 - \varepsilon_F) \\
&\overset{(*)}{\geq} 8F(1 - \varepsilon_F) \\
4R + 2M &\geq 8(1 - \varepsilon_F).
\end{aligned}$$

By taking $F \to \infty$, we get:

$$R \geq 2 - \frac{1}{2}M.$$

Fourth expression

Consider the N_2 request vectors $\mathbf{r}_1, \ldots, \mathbf{r}_{N_2}$, such that:

$$\begin{aligned}
\mathbf{r}_k &= (1,2,k) \qquad \forall k \leq \left\lfloor \frac{N_2}{2} \right\rfloor, \\
\mathbf{r}_k &= (2,1,k) \qquad \forall k \geq \left\lfloor \frac{N_2}{2} \right\rfloor + 1.
\end{aligned}$$

Define $\mathcal{X}^e = \{X^{\mathbf{r}_k} : \ k \text{ is even}\}$ and $\mathcal{X}^o = \{X^{\mathbf{r}_k} : \ k \text{ is odd}\}$. Thus, each of $\mathcal{X}^e$ and $\mathcal{X}^o$ contains at least one broadcast $X^{\mathbf{r}_k}$ such that $\mathbf{r}_k = (1, 2, k)$, and one broadcast $X^{\mathbf{r}_l}$ such that $\mathbf{r}_l = (2, 1, l)$. Furthermore, define, for clarity, $\mathcal{W}^2 = (W_1^2, \ldots, W_{N_2}^2)$. Then,

$$\begin{aligned}
N_2RF + 2MF &\geq H\left(Z_1, \mathcal{X}^e\right) + H\left(Z_2, \mathcal{X}^o\right) \\
&= H\left(Z_1, \mathcal{X}^e \middle| W_1^1, W_2^1\right) \\
&\quad + I\left(W_1^1, W_2^1; Z_1, \mathcal{X}^e\right) \\
&\quad + H\left(Z_2, \mathcal{X}^o \middle| W_1^1, W_2^1\right) \\
&\quad + I\left(W_1^1, W_2^1; Z_2, \mathcal{X}^o\right) \\
&\overset{(*)}{\geq} H\left(Z_1, Z_2, \mathcal{X}^e, \mathcal{X}^o \middle| W_1^1, W_2^1\right) \\
&\quad + 4F(1 - \varepsilon_F) \\
&= H\left(Z_1, Z_2, \mathcal{X}^e, \mathcal{X}^o \middle| W_1^1, W_2^1, \mathcal{W}^2\right) \\
&\quad + I\left(\mathcal{W}^2; Z_1, Z_2, \mathcal{X}^e, \mathcal{X}^o \middle| W_1^1, W_2^1\right) \\
&\quad + 4F(1 - \varepsilon_F) \\
&\overset{(*)}{\geq} (4 + N_2)F(1 - \varepsilon_F) \\
N_2R + 2M &\geq (4 + N_2)(1 - \varepsilon_F).
\end{aligned}$$

By taking $F \to \infty$, we get:

$$R \geq \frac{4 + N_2 - 2M}{N_2} = 1 - \frac{M - 2}{N_2/2}.$$

Appendix B

Detailed Proofs for Chapter 3

B.1 Expected Number of Unmatched Users

In this appendix, we will derive upper bounds on the expected number of unmatched users when using PCD, stated in Lemma 11 below. The proof of this lemma requires Lemma 12, also stated below, which gives a more general result on the number of unmatched users. Lemma 12 is also used in Appendix B.5 to bound the number of unmatched users in HCM, which is used to prove Theorem 14.

Lemma 11. *When using PCD, the expected number of unmatched users is no greater than* $K^{-t_0}/\sqrt{2\pi}$.

Before we prove Lemma 11, we will state the following general result on the expected number of unmatched users.

Lemma 12. *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*

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

Lemma 12 is proved at the end of the appendix.

Proof of Lemma 11: 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 $U^0 = \sum_c U(c)$ 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 12 directly to obtain

$$\begin{aligned}\mathbb{E}[U^0] &= \sum_{c=1}^{K/d} \mathbb{E}[U(c)] \\ &\leq \frac{K}{d} \cdot \frac{1}{\sqrt{2\pi}} \cdot d \cdot \left(\rho e^{1-\rho}\right)^d \\ &= \frac{1}{\sqrt{2\pi}} \exp\left\{\log K + d \log\left(\rho e^{1-\rho}\right)\right\}. \end{aligned} \tag{B.1}$$

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 (B.1), we obtain

$$\begin{aligned}\mathbb{E}[U^0] &\leq \frac{1}{\sqrt{2\pi}} \exp\left\{\log K + d \log\left(2\rho e^{1-2\rho}\right)\right\} \\ &\overset{(a)}{\leq} \frac{1}{\sqrt{2\pi}} \exp\left\{\log K - (1+t_0)\log K\right\} \\ &= \frac{1}{\sqrt{2\pi}} K^{-t_0}, \end{aligned}$$

where (a) uses (3.1). This concludes the proof. ■

Before we prove Lemma 12, we need another lemma that pertains to Poisson variables in general.

Lemma 13. *Let Y be a Poisson random variable with parameter λ, 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,*

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

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

of expectation,

$$\begin{aligned}
\mathbb{E}[U] &= \mathbb{E}\left[\mathbb{E}\left[U|V\right]\right] \\
&= \Pr\{V=0\}\mathbb{E}\left[U|V=0\right] + \Pr\{V=1\}\mathbb{E}\left[U|V=1\right] \\
&\overset{(a)}{=} 0 + \Pr\{V=1\}\mathbb{E}\left[Y-m|V=1\right] \\
&= \Pr\{Y\geq m\}\mathbb{E}\left[Y-m|Y\geq m\right] \\
&= \Pr\{Y\geq m\}\left(\mathbb{E}[Y|Y\geq m]-m\right) \\
&\overset{(b)}{=} \Pr\{Y\geq m\}\left(m\Pr\{Y=m|Y\geq m\}+\lambda-m\right) \\
&\overset{(c)}{\leq} \Pr\{Y\geq m\}\cdot m\Pr\{Y=m|Y\geq m\} \\
&= m\Pr\{Y=m\},
\end{aligned}$$

where (a) uses the definition of U given the different values of V, (b) uses [51, Proposition 1],[1] and (c) uses $\lambda \leq m$. ■

We can now prove Lemma 12.

Proof of Lemma 12: By using Lemma 13 with $\lambda = \gamma m$, we have

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

Using Stirling's approximation, we have

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

which yields

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

thus concluding the proof. ■

[1] This proposition appears in the appendix in the extended version of [51].

B.2 Details of PAM Scheme for $\beta \in [0, 1)$ (Proof of Theorem 8)

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 ρK, we always have $\bar{R}^{\text{PAM}} \leq \rho K$.

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 $\rho d p_n$.

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 $d_n = p_n dM$ caches per cluster.

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

$$\sum_{W_n \in \mathcal{W}_k} \frac{u_n(c)}{d_n} \leq 1, \tag{B.2}$$

where $\mathcal{W}_k$ 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 [49, Proposition 1], we have that

$$\Pr\left\{ \sum_{W_n \in \mathcal{W}_k} \frac{u_n(c)}{d_n} > 1 \right\} \leq e^{-zdM/N}, \tag{B.3}$$

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 (B.2) 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 caches in 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 [64]. We use the Hungarian algorithm to find a matching between the remaining requests and the caches in the corresponding cluster.

Let τ_n be the probability that at least one of the caches storing file W_n does not satisfy (B.2). By the union bound, it follows that

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

By definition,

$$\bar{R}^{\text{PAM}} \leq \sum_{n=1}^{N} \tau_n \leq KMe^{-zdM/N}, \tag{B.4}$$

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 (B.2). Using (B.3) 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,

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

where $\overline{A}$ denotes the complement of A for any event A.

It follows that

$$\begin{aligned}\mathbb{E}[R] &= \mathbb{E}[R|G]\Pr\{G\} + \mathbb{E}[R|\overline{G}]\Pr\{\overline{G}\} \\ &\leq \mathbb{E}[R|G] + N\Pr\{\overline{G}\} \\ &\leq Kde^{-zdM/N} + \frac{NK}{d}e^{-zd}. \end{aligned} \tag{B.5}$$

Using (B.4) and (B.5), we obtain

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

which is a tighter bound on the PAM rate and implies Theorem 8.

B.3 Details of PAM Scheme for $\beta > 1$ (Proof of Theorem 11)

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 chapter, 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.

B.3.1 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 12, we know that the expected total number of these excess users across all caches is $O(K^{-t_0})$, which is $o(1)$. For the proof of Theorem 11, 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.3.2 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.

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

where N_1 and N_2 are defined as

$$N_1 = \frac{d^{1/\beta}}{p_1(\log d)^{2/\beta}}; \tag{B.6a}$$

$$N_2 = d^{(1+1/\beta)/2}. \tag{B.6b}$$

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

$$\begin{aligned} \underset{x_1,\ldots,x_N}{\text{maximize}} \quad & \sum_{n=1}^{N} v_n x_n \\ \text{subject to} \quad & \sum_{n=1}^{N} w_n x_n \leq dM; \\ & 0 \leq x_n \leq 1, \forall n. \end{aligned} \tag{B.7}$$

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.

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 $\mathcal{S}$ containing exactly c_n copies of each file index n. Let us order the elements of $\mathcal{S}$ in increasing order, and call the resulting ordered list $(n_1, \ldots, n_{\lfloor dM \rfloor})$. Then, for each r, we store file W_{n_r} in cache $((r-1) \bmod d) + 1$ of the cluster.

B.3.3 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 2 gives the precise description of MLP.

At the end of Algorithm 2, 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.

Algorithm 2 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

Set $\mathcal{K}_n \subseteq \{1, \ldots, d\}$ to be the set of caches containing file W_n, for each n
Loop over all files from least to most popular:
for $n \leftarrow N, N-1, \ldots, 1$ **do**
    Loop over all requests for file n:
    for $v \leftarrow 1, \ldots, u_n$ **do**
        if $\mathcal{K}_n \neq \emptyset$ **then**
            Pick $k \in \mathcal{K}_n$ uniformly at random
            Match a user requesting file W_n to cache k
            Cache k is no longer available:
            for all $n' \in \{1, \ldots, N\}$ **do**
                $\mathcal{K}_{n'} \leftarrow \mathcal{K}_{n'} \setminus \{k\}$
            end for
        end if
    end for
end for

B.3.4 Expected Rate Achieved by KS+MLP

Lemma 14. *Let X be a Poisson random variable with mean μ, and let $\varepsilon \in (0,1)$ be arbitrary. Then,*

$$\Pr\{X \geq (1+\varepsilon)\mu\} \leq e^{-\mu h(1+\varepsilon)},$$

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

Proof: The lemma follows from the Chernoff bound. ■

Lemma 15. *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*

$$u_n(c) \leq \left(1 + \frac{p_1}{4}\right) dp_n \quad \text{for all } 1 \leq n \leq N_1; \text{ and}$$
$$u_n(c) \leq 2p_1(\log d)^2 \qquad \text{for all } N_1 < n \leq N_2,$$

where N_1 and N_2 are as defined in (B.6)*. 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 $\rho d p_n$.

- For $n \leq N_1$, we have $dp_n \geq p_1(\log d)^2$. Therefore, using Lemma 14, 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 $dp_n < p_1(\log d)^2$. By defining $\tilde{u}$ to be a Poisson variable with parameter $\rho p_1(\log d)^2 > \rho dp_n$, we obtain

$$\begin{aligned}\Pr\left\{u_n > 2p_1(\log d)^2\right\} &\overset{(a)}{<} \Pr\left\{\tilde{u} > 2p_1(\log d)^2\right\} \\ &\overset{(b)}{\leq} e^{-\Omega((\log d)^2)},\end{aligned}$$

  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 λ, is an increasing function of λ, and (b) follows from Lemma 14.

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

Lemma 16. *Let $\mathcal{R} = \{n : x_n = 1\}$, where x_n is the solution to the fractional knapsack problem solved in Appendix B.3.2. Let E_2 denote the event that, in a given cluster, the MLP policy matches* all *requests for all files in $\mathcal{R}$ to caches. Then,*

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

Proof: Since the MLP policy matches requests to caches 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

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

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 14), 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 $n \in \mathcal{R}$ such that $n > N_2$ is not matched by the MLP policy is at most $de^{-\Omega(d)}$.

Next, we focus on the files $n \in \{2, \ldots, N_2\}$. Note that if the KS policy decides to store file n, it stores it on w_n caches. Therefore,

$$\begin{aligned}
\sum_{n=2}^{N_2} x_n w_n &\leq \sum_{n=2}^{N_1} \left\lceil \left(1 + \frac{p_1}{2}\right) d p_n \right\rceil + \sum_{n=N_1+1}^{N_2} \left\lceil 4p_1 (\log d)^2 \right\rceil \\
&\leq \sum_{n=2}^{N_1} \left[\left(1 + \frac{p_1}{2}\right) d p_n + 1 \right] \\
&\quad + \sum_{n=N_1+1}^{N_2} \left[4p_1 (\log d)^2 + 1 \right] \\
&\leq \left(1 + \frac{p_1}{2}\right) d(1 - p_1) + 4p_1 (\log d)^2 N_2 + N_2 \\
&\leq \left(1 + \frac{p_1}{2}\right) d(1 - p_1) + \Theta\left(d^{(1+1/\beta)/2} (\log d)^2\right) \\
&\leq d,
\end{aligned}$$

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 $\mathcal{K}_n$ be the set of caches storing file n in a given cluster. Let $E_{3,k}$ be the event that cache $k \in \mathcal{K}_n$ 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 $0 < \varepsilon < 1$, there exists a $d(\varepsilon)$ such that $\Pr\{E_{3,k}\} \leq \varepsilon$ for all $d \geq d(\varepsilon)$.

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 $\lceil 4p_1(\log d)^2 \rceil$ caches in $\mathcal{K}_n$ are matched to users requesting some file $n' > N_2$. By the Chernoff bound for negatively associated random variables [64],

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

From Lemma 15, we know that with probability at least $1 - Ne^{-\Omega((\log d)^2)}$, there are less than $(1+p_1/4)dp_n$ 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 $\mathcal{R}$ 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. ■

B.3.5 Expected Rate

From Lemma 16, 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 $\widetilde{N}$ be the number of files not in $\mathcal{R}$ (i.e., that are not cached) that are requested at least once. By the union bound over the K/d clusters,

$$\begin{aligned}\bar{R}^{\mathrm{PAM}} &\leq \mathbb{E}[\widetilde{N}] + \frac{K^2}{d}\left(1 - \Pr\{E_2\}\right) \\ &\leq \mathbb{E}[\widetilde{N}] + \frac{NK^2}{d}e^{-\Omega((\log d)^2)}.\end{aligned}$$

After solving the fractional knapsack problem, defined in (B.7), as a function of N, K, d, β, and M, we can determine the set $\mathcal{R}$. For a given $\mathcal{R}$, we then have

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

We hence obtain the following bound on the expected rate:

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

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 11.

B.4 Approximate Optimality (Proof of Theorem 13)

In this section, we focus on the case $\beta \in [0, 1)$ to prove Theorem 13. 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

$$p_n \geq p_N = \frac{N^{-\beta}}{A_N} \overset{(a)}{\geq} \frac{(1-\beta)N^{-\beta}}{N^{1-\beta}} = \frac{1-\beta}{N}, \tag{B.8}$$

where A_N is defined in Lemma 17 stated below, and (a) follows from the lemma.

Lemma 17. *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 17 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

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

Since $(1-\beta)\rho d/N \leq p_n \rho d$ for all n by (B.8), 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 $\mathrm{Poisson}\left((p_n - (1-\beta)/N)\rho d\right)$ 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 [51]

except that ρ is replaced by $\rho' = (1-\beta)\rho$, which is still a constant. Consequently, the information-theoretic lower bounds obtained in [51, Lemma 2] and the inequalities that follow can be directly applied here, giving the following lemma.

Lemma 18. *When $N \geq 10$, the optimal expected rate $\bar{R}^*$ can be lower-bounded by*

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

When $M < (1-e^{-1}/2)N/2d$, the bound in Lemma 18 can be further lower-bounded by

$$\bar{R}^* \geq \frac{(1-\beta)(1-e^{-1}/2)^2\rho}{96} \cdot \min\left\{\frac{N}{M}, \rho K\right\}. \tag{B.9}$$

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

$$\bar{R}^{\mathrm{PCD}} \leq \min\left\{\rho K, \frac{N}{M} - 1 + \frac{K^{-t_0}}{\sqrt{2\pi}}\right\} \leq \min\left\{\rho K, \frac{N}{M}\right\}. \tag{B.10}$$

Consequently, combining (B.9) with (B.10) gives us the result of Theorem 13.

Proof of Lemma 17: To prove the lemma, we will relate the sum $A_m = \sum_n n^{-\beta}$ 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

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

Rearranging the inequalities, we get the equivalent statement that

$$f(l) + \int_k^l f(x)\,\mathrm{d}x \leq \sum_{n=k}^{l} f(n) \leq f(k) + \int_k^l f(x)\,\mathrm{d}x. \tag{B.11}$$

Recall that $A_m = \sum_{n=1}^{m} n^{-\beta}$. Thus we can apply (B.11) with $f(x) = x^{-\beta}$, $k = 1$, and $l = m$. Since we know that

$$\int_1^m x^{-\beta}\,\mathrm{d}x = \frac{m^{1-\beta} - 1}{1-\beta},$$

this implies, using (B.11),

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

which concludes the proof. ■

B.5 Details of HCM Scheme (Proof of Theorem 14)

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

$$\log K \geq 2g\alpha, \tag{B.12}$$

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

Let $t \in [0, t_0]$, and let $\chi = \lfloor \alpha g d/(2(1+t)\log K) \rfloor$. We will partition the set of files into χ *colors*. For each color $x \in \{1, \ldots, \chi\}$, define $\mathcal{W}_x$ as the set of files colored with x. We choose to color the files in an alternating fashion. More precisely, we choose for each $x \in \{1, \ldots, \chi\}$

$$\mathcal{W}_x = \{W_n : n \equiv x \pmod{\chi}\}.$$

Notice that $|\mathcal{W}_x| = \lfloor N/\chi \rfloor$ or $\lceil N/\chi \rceil$. We can now define the *popularity of a color* x as

$$\mathcal{P}_x = \sum_{W_n \in \mathcal{W}_x} p_n.$$

The following proposition, proved at the end of the section, gives a useful lower bound for $\mathcal{P}_x$.

Proposition 3. *For each $x \in \{1, \ldots, \chi\}$, we have $\mathcal{P}_x \geq g/\chi$.*

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 χ colors. We choose this coloring

in such a way that the number of caches associated with a particular color is proportional to the popularity of that color. Specifically, exactly $\lfloor d\mathcal{P}_x \rfloor$ caches in every cluster will be colored with 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 $|\mathcal{W}_x|$ files and $\lfloor d\mathcal{P}_x \rfloor K/d$ caches in total. The idea is to perform a Maddah-Ali–Niesen scheme [4, 5] 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 $\mathcal{W}_x$ 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 has requested. Thus if the user is at cluster c and requests a file from $\mathcal{W}_x$, 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 $\mathcal{W}_x$. 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

$$\bar{R}^{\mathrm{HCM}} = \min\left\{\rho K, \sum_{x=1}^{\chi} \mathbb{E}[R_x] + \mathbb{E}[U^0]\right\}, \tag{B.13}$$

since ρK can always be achieved by simply unicasting to every user its requested file.

From [5], we know that we can always upper-bound the rate for color x by

$$R_x \leq \left[\frac{|\mathcal{W}_x|}{M} - 1\right]^+,$$

for all $M > 0$. Because $|\mathcal{W}_x| = \lfloor N/\chi \rfloor$ or $\lceil N/\chi \rceil$ for all x, we obtain

$$\sum_x R_x \leq \begin{cases} \frac{N}{M} - 1 & \text{if } M \leq \lfloor N/\chi \rfloor; \\ 0 & \text{if } M \geq \lceil N/\chi \rceil; \\ (N \bmod \chi)\left(\frac{\lceil N/\chi \rceil}{M} - 1\right) & \text{otherwise.} \end{cases} \tag{B.14}$$

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 $\lfloor d\mathcal{P}_x \rfloor$ caches at cluster c with color x, then exactly $U(c,x) = [Y(c,x) - \lfloor d\mathcal{P}_x \rfloor]^+$ 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 $Y(c,x) \sim \text{Poisson}(\rho d \cdot \mathcal{P}_x)$, and that the $Y(c,x)$ users must be matched to $\lfloor d\mathcal{P}_x \rfloor$. For convenience, we define $\widetilde{Y}(c,x) \sim \text{Poisson}(2\rho \cdot \lfloor d\mathcal{P}_x \rfloor)$ and $\widetilde{U} = [\widetilde{Y}(c,x) - \lfloor d\mathcal{P}_x \rfloor]^+$. Since we have

$$\rho d\mathcal{P}_x \leq 2\rho \lfloor d\mathcal{P}_x \rfloor,$$

i.e., the Poisson parameter of $\widetilde{Y}(c,x)$ is at least the Poisson parameter of $Y(c,x)$, then

$$\begin{aligned} \mathbb{E}[U(c,x)] &= \sum_{y=\lfloor d\mathcal{P}_x \rfloor}^{\infty} y \cdot \Pr\{Y(c,x) = y\} \\ &\overset{(a)}{\leq} \sum_{y=\lfloor d\mathcal{P}_x \rfloor}^{\infty} y \cdot \Pr\{\widetilde{Y}(c,x) = y\} \\ &= \mathbb{E}[\widetilde{U}(c,x)], \end{aligned}$$

where (a) uses the fact that the function $\lambda \mapsto \Pr\{\text{Poisson}(\lambda) = m\}$ is increasing in λ as long as $\lambda < m$; see [51, Proposition 2].[2]

This allows us to apply Lemma 13 on $\widetilde{Y}(c,x)$ and $\widetilde{U}(c,x)$ in order to upper-bound the expec-

[2]This proposition appears in the appendix in the extended version of [51].

tation of $U(c,x)$ by

$$\mathbb{E}[U(c,x)] \leq \mathbb{E}[\widetilde{U}(c,x)] \leq \frac{1}{\sqrt{2\pi}} \cdot \lfloor d\mathcal{P}_x \rfloor \cdot \left(2\rho e^{1-2\rho}\right)^{\lfloor d\mathcal{P}_x \rfloor}.$$

Consequently, we get the upper bound on $\mathbb{E}[U^0]$,

$$\begin{aligned}
\mathbb{E}[U^0] &= \sum_{c=1}^{K/d} \sum_{x=1}^{\chi} \mathbb{E}[U(c,x)] \\
&\leq \sum_{c=1}^{K/d} \sum_{x=1}^{\chi} \frac{1}{\sqrt{2\pi}} \lfloor d\mathcal{P}_x \rfloor \left(2\rho e^{1-2\rho}\right)^{\lfloor d\mathcal{P}_x \rfloor} \\
&= \frac{1}{\sqrt{2\pi}} \sum_{x=1}^{\chi} \frac{K}{d} \cdot \lfloor d\mathcal{P}_x \rfloor \left(2\rho e^{1-2\rho}\right)^{\lfloor d\mathcal{P}_x \rfloor} \\
&\leq \frac{1}{\sqrt{2\pi}} \sum_{x=1}^{\chi} \mathcal{P}_x \cdot K \left(2\rho e^{1-2\rho}\right)^{\lfloor d\mathcal{P}_x \rfloor}.
\end{aligned}$$

Isolating part of the term in the sum,

$$\begin{aligned}
K \left(2\rho e^{1-2\rho}\right)^{\lfloor d\mathcal{P}_x \rfloor} &= \exp\left\{\log K + \log\left(2\rho e^{1-2\rho}\right) \lfloor d\mathcal{P}_x \rfloor\right\} \\
&= \exp\left\{\log K - \alpha \lfloor d\mathcal{P}_x \rfloor\right\} \\
&\leq \exp\left\{\log K - \frac{\alpha d \mathcal{P}_x}{2}\right\} \\
&\overset{(a)}{\leq} \exp\left\{\log K - \frac{\alpha d g}{2\chi}\right\} \\
&\overset{(b)}{\leq} \exp\left\{\log K - \frac{\alpha d g}{2} \cdot \frac{2(1+t)}{\alpha d g} \log K\right\} \\
&= \exp\left\{\log K - (1+t)\log K\right\} \\
&= K^{-t},
\end{aligned}$$

where (a) uses Proposition 3, and (b) uses the definition of χ combined with $\lfloor y \rfloor \leq y$. We obtain the final upper bound on the expected number of unmatched users,

$$\mathbb{E}[U^0] \leq \frac{1}{\sqrt{2\pi}} \sum_{x=1}^{\chi} \mathcal{P}_x K^{-t} = \frac{K^{-t}}{\sqrt{2\pi}}. \tag{B.15}$$

Finally, we combine (B.14) and (B.15) in (B.13) to obtain the rate expression in Theorem 14,

thus completing its proof.

Proof of Proposition 3: Choose any $x \in \{1, \ldots, \chi\}$. Starting with the definition of $\mathcal{P}_x$, we have

$$\begin{aligned}
\mathcal{P}_x &= \sum_{W_n \in \mathcal{W}_x} p_n \\
&= \frac{1}{A_N} \sum_{k=0}^{|\mathcal{W}_x|-1} (k\chi + x)^{-\beta} \\
&\geq \frac{1}{A_N} \sum_{k=0}^{|\mathcal{W}_x|-1} (k\chi + \chi)^{-\beta} \\
&= \frac{\chi^{-\beta}}{A_N} \sum_{k=0}^{|\mathcal{W}_x|-1} (k+1)^{-\beta} \\
&= \chi^{-\beta} \cdot \frac{A_{|\mathcal{W}_x|}}{A_N} \\
&\overset{(a)}{\geq} \chi^{-\beta} \cdot \frac{|\mathcal{W}_x|^{1-\beta} - 1}{N^{1-\beta}} \\
&\overset{(b)}{\geq} \chi^{-\beta} \cdot \frac{(N/\chi - 1)^{1-\beta} - 1}{N^{1-\beta}} \\
&= \frac{1}{\chi} \cdot \left[\left(1 - \frac{\chi}{N}\right)^{1-\beta} - \left(\frac{\chi}{N}\right)^{1-\beta} \right] \\
&\overset{(c)}{\geq} \frac{1}{\chi} \left[\left(1 - \frac{g\alpha}{2\log K}\right)^{1-\beta} - \left(\frac{g\alpha}{2\log K}\right)^{1-\beta} \right] \\
&\overset{(d)}{\geq} \frac{1}{\chi} \left[\left(1 - \frac{1}{4}\right)^{1-\beta} - \left(\frac{1}{4}\right)^{1-\beta} \right] \\
&= \frac{g}{\chi},
\end{aligned}$$

where (a) uses Lemma 17, (b) uses the fact that $|\mathcal{W}_x| \geq \lfloor N/\chi \rfloor \geq N/\chi - 1$ for all x, (c) uses the definition of χ as well as $N \geq d$ and $t \geq 0$, and (d) uses (B.12). ■

Appendix C

Detailed Proofs for Chapter 4

C.1 Special Case: Small Number of Files

Recall that the separation architecture creates a set of messages $\mathscr{V}$ as an interface between the physical and network layers,

$$\mathscr{V} = \{V_{\mathcal{S}j} : j \in \{1, \ldots, K_t\}, \mathcal{S} \in \mathscr{S}\}$$

for some $\mathscr{S} \subseteq 2^{\{1,\ldots,K_r\}}$, as seen in (4.7). In this chapter, we have so far focused on the choice of messages described by $\mathscr{S}$ in (4.9), in which every transmitter has a message for every subset of exactly $\kappa + 1$ receivers, where $\kappa = K_r M_r / N$ is an integer. While this is order-optimal in most cases, it is insufficient when both the receiver memory and the number of files is small.

To illustrate, consider the case with only a single file in the content library ($N = 1$) and without receiver caches ($M_r = 0$). Furthermore, assume that there is just one transmitter ($K_t = 1$) but many receivers (K_r is large). Seeing as there is only one file, all receivers will request that same file, and hence the obvious strategy is for the transmitter to broadcast the file to all receivers, thus achieving a DoF of 1. However, under the separation architecture described by (4.9), we create one message from the transmitter for every individual receiver, and then send that file separately as K_r different messages. This is clearly inefficient since the same file is being sent K_r times, thus achieving a much worse DoF of $1/K_r$.

The reason the usual separation architecture is inefficient in this example is that it inherently assumes that all users request different files in the worst case. This is true when there are more files than users. However, if there are so few files that many users will inevitably request the same file, then the previous assumption fails. In this appendix, we handle that case by providing a different separation interface. We exclusively work with the case $M_r = 0$ and compute an achievable DoF for it. Specifically, we show that

$$\frac{1}{\mathsf{DoF}} \leq \frac{K_t + \min\{K_r, N\} - 1}{K_t}. \tag{C.1}$$

Since we also know that $1/\mathsf{DoF}$, which is convex in M_r, is zero when $M_r = N$, then we can achieve any linear combination of the two reciprocal DoFs between these two points, using time- and memory-sharing. Specifically, we achieve

$$\frac{1}{\mathsf{DoF}} \leq \frac{K_t + \min\{K_r, N\} - 1}{K_t} \cdot \left(1 - \frac{M_r}{N}\right). \tag{C.2}$$

The expression of the reciprocal DoF in (C.2) can be decomposed into two gains, in a similar way as in (4.5). Since the strategy that achieves (C.2) is relevant when $N < K_r$, we can write the two gains as

$$N\mathsf{DoF} \approx \underbrace{\frac{K_t N}{K_t + N - 1}}_{g^{\mathrm{IA}}} \cdot \underbrace{\frac{1}{1 - \frac{M_r}{N}}}_{g^{\mathrm{LC}}}.$$

Note that $N\mathsf{DoF}$ is the sum DoF here since the total number of requested files is $N < K_r$ in the worst case.

The most striking difference with (4.5) is that there is no global caching gain. Indeed, the strategy makes no use of any coding or multicasting opportunities, as we will see below. On the other hand, the local caching gain is present and is the same as before. The interference alignment gain is slightly different: it is the interference alignment gain of a $K_t \times N$ unicast X-channel, not $K_t \times K_r$. The reason for this is that, when $N < K_r$, then the total number of *distinct* requested files is N in the worst case. The strategy thus only needs to account for N distinct demands, and uses methods from the compound X-channel [65, 66] to serve them.

We proceed with the strategy for $M_r = 0$ that achieves (C.1). Since $M_r = 0$, we cannot store anything in the receiver caches. In the transmitter caches, we place the same content as previously described, i.e., every file W_n is split into K_t parts, and transmitter j stores the j-th part of W_n, called W_n^j, for every n. In the delivery phase, we partition the set of users into subsets such that all the users in the same subset request the same file. Specifically, let $\mathbf{u}$ denote the request vector, and let $\mathcal{U}_n = \{i : u_i = n\}$ be the set of users requesting file W_n. Our goal is to create a multicast message from every transmitter to all users that are requesting the same file. In other words, we set

$$\mathscr{S} = \{\mathcal{U}_n : n \in \{1, \ldots, N\} \text{ s.t. } \mathcal{U}_n \neq \emptyset\}$$

Note that $\mathscr{S}$ is a partition of the entire set of users. We denote its size by $\tilde{N} = |\mathscr{S}|$, which is equivalent to the total number of distinct requested files. Our separation interface $\mathscr{V}$ is thus a set of messages from every transmitter to $\tilde{N}$ *non-overlapping* subsets of receivers,

$$\mathscr{V} = \{V_{\mathcal{U}_n j} : \mathcal{U}_n \neq \emptyset \text{ and } j = 1, \ldots, K_t\}.$$

We focus on transmitting these messages across the interference channel at the physical layer. At the network layer, we use these messages as error-free bit pipes to deliver the requested files to the users at the network layer.

C.1.1 Physical Layer

At the physical layer, the problem is equivalent to the compound X-channel problem, described in [65, 66]. In the $K_t \times K_r$ compound X-channel, every transmitter has a message for every receiver. However, the channel of every receiver i can be one of some *finite* number J_i of states, and transmission has to account for all possible states. The optimal sum DoF in this problem is $K_t K_r/(K_t + K_r - 1)$, i.e., $1/(K_t + K_r - 1)$ per message, as stated in [65, Theorem 4].

If the receiver is able to decode its messages regardless of which of the J_i realizations the channel has taken, then this is equivalent to replacing the single receiver with J_i channel realizations by J_i different receivers with each a single possible channel realization, such that all J_i receivers want the same messages. This is exactly the problem statement we have at the physical layer. Our

problem is therefore equivalent to a $K_t \times \tilde{N}$ compound X-channel with $|\mathcal{U}_n|$ channel realizations for every receiver n. Therefore, if $\tilde{R}_{\tilde{N}}$ denotes the rate of each message, and $\tilde{d}_{\tilde{N}}$ its DoF, then [65, Theorem 4] implies that the optimal DoF is

$$\tilde{d}_{\tilde{N}}(K_t, K_r) = \frac{1}{K_t + \tilde{N} - 1}. \tag{C.3}$$

C.1.2 Network Layer

Let the link load $\ell_{\tilde{N}}$ denote the size of each $V_{\mathcal{U}_n j}$ in units of files. The strategy at the network layer is straightforward. For every subset $\mathcal{U}_n$ of users, each transmitter j sends the part of the file that they requested through $V_{\mathcal{U}_n j}$. Mathematically, we set

$$V_{\mathcal{U}_n j} = W_n^j,$$

for all $j = 1, \ldots, K_t$ and n such that $\mathcal{U}_n \neq \emptyset$. This allows every user to decode its requested file. Since every multicast link $V_{\mathcal{U}_n j}$ carries one file part W_n^j, then the link load is

$$\ell_{\tilde{N}}(N, K_t, K_r, M_t, 0) = \frac{1}{K_t}. \tag{C.4}$$

C.1.3 Achievable End-to-End DoF

Note that the same $V_{\mathcal{U}_n j}$ has a size of $\tilde{R}_{\tilde{N}} T$ at the physical layer and $\ell_{\tilde{N}} F$ at the network layer. Since $F = RT$, we get $R = \tilde{R}_{\tilde{N}} / \ell_{\tilde{N}}$, and by combining that with (C.3) and (C.4), we achieve a DoF of

$$\frac{\tilde{d}_{\tilde{N}}(K_t, K_r)}{\ell_{\tilde{N}}(N, K_t, K_r, M_t, 0)} = \frac{K_t}{K_t + \tilde{N} - 1}.$$

In the worst case, the largest number of distinct files are requested, i.e., $\tilde{N} = \min\{K_r, N\}$. Therefore,

$$\mathsf{DoF} \geq \frac{K_t}{K_t + \min\{K_r, N\} - 1},$$

when $M_r = 0$.

Since $1/\mathsf{DoF}$ is convex in M_r, and $1/\mathsf{DoF} = 0$ when $M_r = N$, then, for all intermediate values

of M_r, we can achieve

$$\frac{1}{\mathsf{DoF}} \leq \frac{K_t + \min\{K_r, N\} - 1}{K_t} \cdot \left(1 - \frac{M_r}{N}\right). \tag{C.5}$$

For any $M_r \in [0, N]$, we can choose whichever of the two separation interfaces yields the higher DoF. Therefore, combining (C.5) with (4.13) yields

$$\begin{aligned}\frac{1}{\mathsf{DoF}} &\leq \min\left\{\frac{K_t + \min\{K_r, N\} - 1}{K_t} \cdot \left(1 - \frac{\kappa}{K_r}\right), \frac{K_t - 1 + \frac{K_r}{\kappa+1}}{K_t} \cdot \left(1 - \frac{\kappa}{K_r}\right)\right\} \\ &= \frac{K_t + \min\left\{\frac{K_r}{\kappa+1}, N\right\} - 1}{K_t} \cdot \left(1 - \frac{\kappa}{K_r}\right),\end{aligned}$$

when $M_r = \kappa N/K_r$ with κ being an integer, and the lower convex envelope of these points for all $M_r \in [0, N]$. This concludes the achievability proof of Theorem 15.

C.2 Proof of Lemma 7

Let Γ be defined as in the statement of the lemma, and let $n \in \mathbb{N}$ be arbitrary. Define T_n as

$$T_n = \binom{K_r - 1}{\sigma - 1}\left[(n+1)^\Gamma + (K_t - 1)n^\Gamma\right] + \binom{K_r - 1}{\sigma}(n+1)^\Gamma.$$

We will show that a DoF of $\delta_j^{(n)}$ can be achieved for message $V_{\mathcal{S}j}$ over a block length of T_n. We first describe how to (maximally) align the interference at each receiver, and then show that the receiver's desired messages are still decodable. The proof will rely on two lemmas from [54]: the alignment part will use [54, Lemma 2], while the decodability part will rely on [54, Lemma 1]. For ease of reference, we have rephrased the two lemmas in Appendix C.5 as Lemmas 19 and 20, respectively.

Alignment Describe each message $V_{\mathcal{S}j}$ as a column vector of $(n+c_j)^\Gamma$ symbols $\mathbf{v}_{\mathcal{S}j} = [v_{\mathcal{S}j}^m]_{m=1}^{(n+c_j)^\Gamma}$, where c_j is as defined in the statement of the lemma. Each symbol is beamformed along a length-T_n

vector $\mathbf{a}^m_{\mathcal{S}j}$, so that transmitter j sends the codeword

$$\mathbf{x}_j = \sum_{\mathcal{S}:|\mathcal{S}|=\sigma} \sum_{m=1}^{(n+c_j)^\Gamma} v^m_{\mathcal{S}j} \mathbf{a}^m_{\mathcal{S}j},$$

over the block length T_n. We can alternatively combine all the $\mathbf{a}^m_{\mathcal{S}j}$ vectors into one matrix $\mathbf{A}_{\mathcal{S}j} = [\mathbf{a}^1_{\mathcal{S}j}, \ldots, \mathbf{a}^{(n+c_j)^\Gamma}_{\mathcal{S}j}]$, and write

$$\mathbf{x}_j = \sum_{\mathcal{S}} \mathbf{A}_{\mathcal{S}j} \mathbf{v}_{\mathcal{S}j}.$$

Receiver i then observes

$$\mathbf{y}_i = \sum_{j=1}^{K_t} \mathbf{H}_{ij} \mathbf{x}_j + \mathbf{z}_i = \sum_{j=1}^{K_t} \mathbf{H}_{ij} \sum_{\mathcal{S}} \mathbf{A}_{\mathcal{S}j} \mathbf{v}_{\mathcal{S}j} + \mathbf{z}_i. \tag{C.6}$$

Recall that $\mathbf{z}_i$ is the iid additive Gaussian unit-variance noise, and $\mathbf{H}_{ij}$ is a $T_n \times T_n$ diagonal matrix representing the independent continuously-distributed channel coefficients over block length T_n, as defined in Section 4.2. In other words, the τ-th diagonal element of $\mathbf{H}_{ij}$ is $h_{ij}(\tau)$. Moreover, the dimensions of $\mathbf{A}_{\mathcal{S}j}$ are $T_n \times (n + c_j)^\Gamma$, and the length of $\mathbf{v}_{\mathcal{S}j}$ is $(n + c_j)^\Gamma$.

In the expression for $\mathbf{y}_i$ in (C.6), it will be convenient to separate the messages intended for i from the interfering messages,

$$\mathbf{y}_i = \sum_{\mathcal{S}:i\in\mathcal{S}} \sum_{j=1}^{K_t} \mathbf{H}_{ij} \mathbf{A}_{\mathcal{S}j} \mathbf{v}_{\mathcal{S}j} + \sum_{\mathcal{S}:i\notin\mathcal{S}} \left[\mathbf{H}_{i1} \mathbf{A}_{\mathcal{S}1} \mathbf{v}_{\mathcal{S}1} + \sum_{j=2}^{K_t} \mathbf{H}_{ij} \mathbf{A}_{\mathcal{S}j} \mathbf{v}_{\mathcal{S}j} \right] + \mathbf{z}_i.$$

Our goal is to collapse each term inside the second sum (i.e., for each $\mathcal{S}$ such that $i \notin \mathcal{S}$) into a single subspace, namely the subspace spanned by $\mathbf{H}_{i1}\mathbf{A}_{\mathcal{S}1}$.[1] This should be done for all $i \in \{1, \ldots, K_r\}$. Specifically, we want to choose the $\mathbf{A}_{\mathcal{S}j}$'s such that they satisfy the following conditions almost surely:

$$\mathbf{H}_{ij}\mathbf{A}_{\mathcal{S}j} \prec \mathbf{H}_{i1}\mathbf{A}_{\mathcal{S}1}, \qquad \forall i = 1, \ldots, K_r, \quad \forall j = 2, \ldots, K_t, \quad \forall \mathcal{S} \text{ s.t. } i \notin \mathcal{S},$$

where $\mathbf{P} \prec \mathbf{Q}$ denotes that the vector space spanned by the columns of $\mathbf{P}$ is a subspace of the one spanned by the columns of $\mathbf{Q}$.

[1] This is why we choose $\mathbf{v}_{\mathcal{S}1}$ to be a longer vector than $\mathbf{v}_{\mathcal{S}j}, j \geq 2$: this choice makes $\mathbf{H}_{i1}\mathbf{A}_{\mathcal{S}1}$ the larger subspace, which allows us to align the other subspaces with it using Lemma 19.

First, we set $\mathbf{A}_{\mathcal{S}2} = \cdots = \mathbf{A}_{\mathcal{S}K_t}$ for all subsets $\mathcal{S}$. Thus we have reduced the problem to finding matrices $\mathbf{A}_{\mathcal{S}1}$ and $\mathbf{A}_{\mathcal{S}2}$ for all subsets $\mathcal{S}$ such that, almost surely,

$$\mathbf{H}_{i1}^{-1}\mathbf{H}_{ij}\mathbf{A}_{\mathcal{S}2} \prec \mathbf{A}_{\mathcal{S}1}, \qquad \forall i \notin \mathcal{S}, \quad \forall j = 2, \ldots, K_t. \tag{C.7}$$

Note that $\mathbf{H}_{i1}^{-1}$ exists almost surely since each diagonal element of $\mathbf{H}_{i1}$ follows a continuous distribution and is thus non-zero with probability one.

For every $\mathcal{S}$, the matrices $\mathbf{A}_{\mathcal{S}1}$ and $\mathbf{A}_{\mathcal{S}2}$ are constrained by a total of $(K_r - \sigma)(K_t - 1) = \Gamma$ subspace relations. We hence have Γ relations $\mathbf{G}_g\mathbf{A}_{\mathcal{S}2} \prec \mathbf{A}_{\mathcal{S}1}$, $g = 1, \ldots, \Gamma$, where $\mathbf{G}_g$ are $T_n \times T_n$ diagonal matrices. We can write all the diagonal elements of these matrices as forming the set

$$\mathcal{G} = \left\{ \frac{h_{ij}(\tau)}{h_{i1}(\tau)} : i \notin \mathcal{S}, j \in \{2, \ldots, K_t\}, \tau \in \{1, \ldots, T_n\} \right\}.$$

Importantly, each element of $\mathcal{G}$ follows a continuous distribution when conditioned on all the others, i.e.,

$$\frac{h_{ij}(\tau)}{h_{i1}(\tau)} \left| \left\{ \frac{h_{i'j'}(\tau')}{h_{i'1}(\tau')} : (i', j', \tau') \neq (i, j, \tau) \right\} \right. \sim \text{a continuous distribution.}$$

Furthermore, the dimensions of $\mathbf{A}_{\mathcal{S}2}$ are $T_n \times n^{\Gamma}$, and the dimensions of $\mathbf{A}_{\mathcal{S}1}$ are $T_n \times (n+1)^{\Gamma}$, with $T_n > (n+1)^{\Gamma}$.

Let P be some continuous probability distribution with a bounded support. For every $\mathcal{S} \subseteq \{1, \ldots, K_r\}$ such that $|\mathcal{S}| = \sigma$, we generate a $T_n \times 1$ column vector $\mathbf{b}_{\mathcal{S}} = (b_{\mathcal{S}}(1), \ldots, b_{\mathcal{S}}(T_n))^{\top}$, such that all the entries of all $\binom{K_r}{\sigma}$ vectors $\{\mathbf{b}_{\mathcal{S}}\}_{\mathcal{S}}$ are chosen iid from P. We can now invoke Lemma 19 in Appendix C.5 to construct with probability one, for each $\mathcal{S}$ and using $\mathbf{b}_{\mathcal{S}}$, full-rank matrices $\mathbf{A}_{\mathcal{S}1}$ and $\mathbf{A}_{\mathcal{S}2}$ that satisfy the subspace relations in (C.7) almost surely. Furthermore, the entries of the τ-th rows of both $\mathbf{A}_{\mathcal{S}1}$ and $\mathbf{A}_{\mathcal{S}2}$ are each a multi-variate monomial in the entries of the τ-th rows of $\mathbf{b}_{\mathcal{S}}$ and $\mathbf{G}_g$, $g = 1, \ldots, \Gamma$, i.e.,

$$b_{\mathcal{S}}(\tau) \quad \text{and} \quad \frac{h_{ij}(\tau)}{h_{i1}(\tau)}, i \notin \mathcal{S}, j = 2, \ldots, K_t. \tag{C.8}$$

Note that the monomial entries of $\mathbf{A}_{\mathcal{S}1}$ are distinct; the same goes for the monomial entries of $\mathbf{A}_{\mathcal{S}2}$.

We have thus ensured alignment of the interfering messages. In the remainder of the proof we show that the desired messages are still almost surely decodable at every receiver.

Decodability Recall that the total dimension of the vector space, i.e., the block length, is

$$T_n = \binom{K_r - 1}{\sigma - 1}\left[(n+1)^\Gamma + (K_t - 1)n^\Gamma\right] + \binom{K_r - 1}{\sigma}.$$

Let us fix a receiver k. For this receiver, we have:

- $\binom{K_r-1}{\sigma-1}$ subspaces $\mathbf{H}_{k1}\mathbf{A}_{\mathcal{S}1}$, $k \in \mathcal{S}$, of dimension $(n+1)^\Gamma$ each, carrying the length-$(n+1)^\Gamma$ vectors $\mathbf{v}_{\mathcal{S}1}$ that must be decoded by receiver k;
- $(K_t - 1)\binom{K_r-1}{\sigma-1}$ subspaces $\mathbf{H}_{kj}\mathbf{A}_{\mathcal{S}2}$, $k \in \mathcal{S}$ and $j = 2, \ldots, K_t$, of dimension n^Γ each, carrying the length-n^Γ vectors $\mathbf{v}_{\mathcal{S}2}$ that must also be decoded by receiver k;
- $\binom{K_r-1}{\sigma}$ subspaces corresponding to $\mathbf{H}_{k1}\mathbf{A}_{\mathcal{S}1}$, $k \notin \mathcal{S}$, of dimension $(n+1)^\Gamma$ each, collectively carrying all the interference at receiver k.

Our goal is to show that the above subspaces are non-aligned for every receiver k, which implies that the desired messages are decodable with high probability for a large enough SNR.

Define matrices $\mathbf{D}_k$ and $\mathbf{I}_k$ representing the subspaces carrying the desired messages and the interference, respectively, by horizontally concatenating the subspaces:

$$\mathbf{D}_k = \Big[\mathbf{H}_{k1}\mathbf{A}_{\mathcal{S}1} \quad \mathbf{H}_{k2}\mathbf{A}_{\mathcal{S}2} \quad \cdots \quad \mathbf{H}_{kK_t}\mathbf{A}_{\mathcal{S}2}\Big]_{\mathcal{S}:k\in\mathcal{S}}; \tag{C.9a}$$

$$\mathbf{I}_k = \Big[\mathbf{H}_{k1}\mathbf{A}_{\mathcal{S}1}\Big]_{\mathcal{S}:k\notin\mathcal{S}}. \tag{C.9b}$$

Therefore, decodability at receiver k is ensured if the $T_n \times T_n$ matrix

$$\mathbf{\Psi}_k = \Big[\mathbf{D}_k \quad \mathbf{I}_k\Big]$$

is full rank almost surely. We prove that this is true with the help of Lemma 20 in Appendix C.5. To apply Lemma 20, we need to show that the following two conditions hold.

1. Two distinct rows of $\mathbf{\Psi}_k$ consist of monomials in disjoint sets of variables. In other words, the variables involved in the monomials of a specific row are exclusive to that row.

2. Within each row, each entry is a *unique* product of powers of the variables associated with that row.

To show that the first condition holds, consider the τ-th row of $\mathbf{\Psi}_k$. This row consists of monomial terms in the variables $\{b_{\mathcal{S}}(\tau)\}_{\mathcal{S}}$ and $\{h_{ij}(\tau)\}_{i,j}$. This is true because the τ-th row of any submatrix $\mathbf{H}_{kj}\mathbf{A}_{\mathcal{S}j}$ of $\mathbf{\Psi}_k$ is equal to $h_{kj}(\tau)$ multiplied by the τ-th row of $\mathbf{A}_{\mathcal{S}j}$, whose entries are monomials in the variables listed in (C.8). Therefore, the variables involved in a row of $\mathbf{\Psi}_k$ are exclusive to that row.

Before we prove that the second condition holds, we emphasize two remarks regarding the monomials that constitute the entries of the τ-th row of $\mathbf{\Psi}_k$.

Remark 1. *All the entries in the τ-th row of submatrix $\mathbf{H}_{kj}\mathbf{A}_{\mathcal{S}j}$ are distinct monomials from one another. This is true because the τ-th row of $\mathbf{H}_{kj}\mathbf{A}_{\mathcal{S}j}$ is equal to the τ-th row of $\mathbf{A}_{\mathcal{S}j}$, whose entries are distinct monomials by construction (using Lemma 20), multiplied by $h_{kj}(\tau)$.*

Remark 2. *It follows from* (C.8) *that the entries of the τ-th row of submatrix $\mathbf{H}_{kj}\mathbf{A}_{\mathcal{S}j}$ are monomials in which only the variables in a set $\mathcal{B}_{\mathcal{S}j}(\tau)$ appear (with non-zero exponent), where $\mathcal{B}_{\mathcal{S}j}(\tau)$ obeys:*

$$b_{\mathcal{S}}(\tau) \in \mathcal{B}_{\mathcal{S}j}(\tau) \quad \textit{and} \quad b_{\mathcal{S}'}(\tau) \notin \mathcal{B}_{\mathcal{S}j}(\tau),\ \forall \mathcal{S}' \neq \mathcal{S}; \tag{C.10a}$$

$$\textit{and} \quad k \in \mathcal{S} \implies \mathcal{B}_{\mathcal{S}j}(\tau) = \{b_{\mathcal{S}}(\tau), h_{kj}(\tau)\} \cup \{h_{i1}(\tau), \ldots, h_{iK_t}(\tau) : i \notin \mathcal{S}\}. \tag{C.10b}$$

Note that when $k \notin \mathcal{S}$, we cannot be sure if $\mathcal{B}_{\mathcal{S}j}(\tau)$ contains $h_{kj}(\tau)$ because the latter is present in the monomials of both $\mathbf{H}_{kj}$ and $\mathbf{A}_{Sj}$, and can hence be canceled out in their product.

Remark 1 states that monomials in the same submatrix are distinct. Therefore, all that remains is to show the same for monomials in the τ-th rows of different submatrices. However, by Remark 2 the same variables appear with non-zero exponent in *all* entries in the τ-th row of any submatrix (albeit with different powers). It is therefore sufficient to prove that the τ-th rows of two different

Table C.1: Exponents of variables in monomials of the τ-th row of $\mathbf{\Psi}_k$ for an arbitrary receiver k. The subsets $\mathcal{S}_1$, $\mathcal{S}_1'$, $\mathcal{S}_2$, and $\mathcal{S}_2'$ are arbitrary such that $k \in \mathcal{S}_1 \cap \mathcal{S}_1'$ and $k \notin \mathcal{S}_2 \cup \mathcal{S}_2'$. The transmitters j and j' are also arbitrary. A cell will contain a check mark (✓) if the variable in the corresponding row appears with non-zero exponent in the monomials of the τ-th row of the submatrix in the corresponding column. The cell will be empty if the variable does not appear in those monomials. It will contain a question mark (?) if the variable may or may not appear. Not all variables and submatrices are shown; only a representative few are used. Finally, recall that $\mathbf{A}_{\mathcal{S}j} = \mathbf{A}_{\mathcal{S}2}$ for $j \geq 2$.

	$\mathbf{I}_k$		$\mathbf{D}_k$			
	$\mathbf{H}_{k1}\mathbf{A}_{\mathcal{S}_2 1}$	$\mathbf{H}_{k1}\mathbf{A}_{\mathcal{S}_2' 1}$	$\mathbf{H}_{kj}\mathbf{A}_{\mathcal{S}_1 j}$	$\mathbf{H}_{kj}\mathbf{A}_{\mathcal{S}_1' j}$	$\mathbf{H}_{kj'}\mathbf{A}_{\mathcal{S}_1 j'}$	$\mathbf{H}_{kj'}\mathbf{A}_{\mathcal{S}_1' j'}$
$b_{\mathcal{S}_1}(\tau)$			✓		✓	
$b_{\mathcal{S}_1'}(\tau)$				✓		✓
$b_{\mathcal{S}_2}(\tau)$	✓					
$b_{\mathcal{S}_2'}(\tau)$		✓				
$h_{kj}(\tau)$	?	?	✓	✓		
$h_{kj'}(\tau)$	?	?			✓	✓

submatrices $\mathbf{H}_{kj}\mathbf{A}_{\mathcal{S}j}$ and $\mathbf{H}_{kj'}\mathbf{A}_{\mathcal{S}'j'}$ are functions of different sets of variables. Specifically, we show that there is a variable that appears with non-zero exponent in all the entries of the τ-th row of one submatrix but in none of the entries of the τ-th row of the other. We will prove below that this claim is true using Remark 2, with the aid of Table C.1 for visualization.

For convenience, define $\mathbf{r}_{\mathcal{S}j}^\top$ to be the τ-th row of matrix $\mathbf{H}_{kj}\mathbf{A}_{\mathcal{S}j}$, and similarly define $\mathbf{r}_{\mathcal{S}'j'}^\top$ to be the τ-th row of matrix $\mathbf{H}_{kj'}\mathbf{A}_{\mathcal{S}'j'}$. To show that the entries of these rows are monomial functions of distinct variables, we isolate two cases: case $\mathcal{S} \neq \mathcal{S}'$ and case $\mathcal{S} = \mathcal{S}'$, $j \neq j'$.

1. Suppose $\mathcal{S} \neq \mathcal{S}'$. Then, by (C.10a), $\mathbf{r}_{\mathcal{S}j}^\top$ is a function of $b_{\mathcal{S}}(\tau)$ but not $b_{\mathcal{S}'}(\tau)$ while the opposite is true of $\mathbf{r}_{\mathcal{S}'j'}^\top$.

2. Suppose now that $\mathcal{S} = \mathcal{S}'$ but $j \neq j'$. Crucially, two such matrices are relevant at receiver k only if $k \in \mathcal{S}$, as evidenced by (C.9). Therefore, by (C.10b), row $\mathbf{r}_{\mathcal{S}j}^\top$ is a function of $h_{kj}(\tau)$ but not $h_{kj'}(\tau)$, and the reverse is true of $\mathbf{r}_{\mathcal{S}'j'}^\top$.

Combining the above two points, it follows that the entries of the two rows are monomials in a different set of variables. We can conclude that the entries in the τ-th row of $\mathbf{\Psi}_k$ are distinct monomials, and specifically that the matrix $\mathbf{\Psi}_k$ is of the form seen in the statement of Lemma 20. Therefore, by Lemma 20, the matrix $\mathbf{\Psi}_k$ is full rank almost surely, and thus all receivers are able to decode their desired messages almost surely.

In conclusion We were able to transmit all the messages $V_{\mathcal{S}j}$, represented by length-$(n+c_j)^\Gamma$ vectors $\mathbf{v}_{\mathcal{S}j}$, over a block length of T_n. Hence, the degrees of freedom achieved for each $V_{\mathcal{S}j}$ is

$$\frac{(n+c_j)^\Gamma}{T_n} = \frac{(n+c_j)^\Gamma}{\binom{K_r-1}{\sigma-1}\left[(n+1)^\Gamma + (K_t-1)n^\Gamma\right] + \binom{K_r-1}{\sigma}(n+1)^\Gamma} = \delta_j^{(n)},$$

which concludes the proof of Lemma 7. ■

C.3 Detailed Converse Proof of Theorem 15

A high-level overview of the converse proof of Theorem 15 was given in Section 4.6. In this appendix, we will give the rigorous proof. In particular, we will prove (4.16), i.e.,

$$\frac{1}{\mathsf{DoF}} \geq \frac{d^{-1}(M_r)}{13.5}$$

by analyzing the four regimes described in (4.14).

Regime 0: $\min\{K_r, N\} \leq 12.5K_t$

In this regime, the number of transmitters is at least of the order of the total number of different requested files. As described in Section 4.3, this implies that $1/\mathsf{DoF} \approx 1 - \frac{M_r}{N}$. More precisely, by convexity of $d^{-1}(M_r)$ we have

$$d^{-1}(M_r) \leq d^{-1}(0) - \frac{d^{-1}(0) - d^{-1}(N)}{N-0}(M_r - 0) = d^{-1}(0)\left(1 - \frac{M_r}{N}\right)$$

for all $M_r \in [0, N]$, where we have used that $d^{-1}(N) = 0$. Moreover, we have

$$d^{-1}(0) = \frac{K_t - 1 + \min\{K_r, N\}}{K_t} \leq \frac{K_t + 12.5K_t}{K_t} = 13.5,$$

which implies

$$d^{-1}(M_r) \leq 13.5\left(1 - \frac{M_r}{N}\right). \tag{C.11}$$

We now invoke Lemma 8 with $s = 1$, yielding

$$\frac{1}{\mathsf{DoF}} \geq 1 - \frac{M_r}{N} \overset{(a)}{\geq} \frac{d^{-1}(M_r)}{13.5} \qquad \text{in Regime 0,} \tag{C.12}$$

where (a) follows from (C.11).

In all the following regimes, $\min\{K_r, N\} > 12.5K_t \geq 12.5$.

Regime 1: $0 \leq M_r \leq 1.1 \max\{1, N/K_r\}$

Since $d^{-1}(M_r)$ is non-increasing in M_r, we can upper-bound it by

$$d^{-1}(M_r) \leq d^{-1}(0) = \frac{K_t - 1 + \min\{K_r, N\}}{K_t} \leq \frac{\left(\frac{1}{12.5} + 1\right)\min\{K_r, N\}}{K_t} = \frac{13.5}{12.5} \cdot \frac{\min\{K_r, N\}}{K_t}. \tag{C.13}$$

Let us now use Lemma 8 with $s = \lfloor 0.275 \min\{K_r, N\} \rfloor \in \{1, \ldots, \min\{K_r, N\}\}$. Then, using $\lfloor N/s \rfloor \geq N/s - 1$,

$$\begin{aligned}
\frac{1}{\mathsf{DoF}} &\geq \frac{s}{\min\{s, K_t\}} \left(1 - \frac{s}{1 - s/N} \cdot \frac{M_r}{N}\right) \\
&\geq \frac{1}{K_t} \left(s - \frac{s^2}{1 - s/N} \cdot \frac{1.1 \max\{1, N/K_r\}}{N}\right) \\
&= \frac{1}{K_t} \left(\lfloor 0.275 \min\{K_r, N\} \rfloor - \frac{\lfloor 0.275 \min\{K_r, N\} \rfloor^2}{1 - \lfloor 0.275 \min\{K_r, N\} \rfloor / N} \cdot \frac{1.1 \max\{1, N/K_r\}}{N}\right) \\
&\geq \frac{1}{K_t} \left(0.275 \min\{K_r, N\} - 1 - \frac{(0.275 \min\{K_r, N\})^2}{1 - 0.275} \cdot \frac{1.1 \max\{1, N/K_r\}}{N}\right) \\
&= \frac{1}{K_t} \left(0.275 \min\{K_r, N\} - 1 - \frac{(0.275 \min\{K_r, N\})^2}{0.725} \cdot \frac{1.1}{\min\{K_r, N\}}\right) \\
&= \frac{\min\{K_r, N\}}{K_t} \left(0.275 - \frac{1}{\min\{K_r, N\}} - \frac{(0.275)^2 \cdot 1.1}{0.725}\right) \\
&\geq \frac{\min\{K_r, N\}}{K_t} \left(0.275 - \frac{1}{12.5} - \frac{(0.275)^2 \cdot 1.1}{0.725}\right) \\
&\geq \frac{1}{12.5} \cdot \frac{\min\{K_r, N\}}{K_t} \\
&\overset{(a)}{\geq} \frac{d^{-1}(M_r)}{13.5} \qquad \text{in Regime 1,}
\end{aligned} \tag{C.14}$$

where (a) uses (C.13).

Regime 2: $1.1\max\{1, N/K_r\} < M_r \leq 0.092N/K_t$

Let $\tilde{M}_r$ be the largest integer multiple of N/K_r that is no greater than M_r, and define $\tilde{\kappa} = K_r\tilde{M}_r/N$. Note that $\tilde{\kappa}$ is an integer. Hence,

$$0 \leq M_r - N/K_r < \tilde{M}_r \leq M_r.$$

Since $d^{-1}(M_r)$ is non-increasing in M_r, we have:

$$\begin{aligned}
d^{-1}(M_r) &\leq d^{-1}(\tilde{M}_r) \\
&= \frac{K_t - 1 + \min\{\frac{K_r}{\tilde{\kappa}+1}, N\}}{K_t} \cdot \left(1 - \frac{\tilde{\kappa}}{K_r}\right) \\
&\leq \frac{K_t + \frac{K_r}{\tilde{\kappa}+1}}{K_t} \\
&= 1 + \frac{K_r}{K_t(\tilde{\kappa}+1)} \qquad \text{(C.15)} \\
&= 1 + \frac{K_r}{K_t(K_r\tilde{M}_r/N + 1)} \\
&\overset{(a)}{\leq} 1 + \frac{K_r}{K_t \cdot K_r M_r/N} \\
&= \frac{N}{K_t M_r}\left(\frac{K_t M_r}{N} + 1\right) \\
&\overset{(b)}{\leq} 1.092\frac{N}{K_t M_r}, \qquad \text{(C.16)}
\end{aligned}$$

where (a) uses $\tilde{M}_r > M_r - N/K_r$ and (b) follows from $M_r \leq 0.092N/K_t$.

We now invoke Lemma 8 with $s = \lfloor 0.3N/M_r \rfloor \in \{1, \ldots, \min\{K_r, N\}\}$. Once again, we write:

$$\begin{aligned}
\frac{1}{\mathsf{DoF}} &\geq \frac{1}{\min\{s, K_t\}} \left(s - \frac{s^2}{1 - s/N} \cdot \frac{M_r}{N} \right) \\
&\geq \frac{1}{K_t} \left(0.3 \frac{N}{M_r} - 1 - \frac{(0.3N/M_r)^2}{1 - 0.3/M_r} \cdot \frac{M_r}{N} \right) \\
&\overset{(a)}{\geq} \frac{1}{K_t} \left(0.3 \frac{N}{M_r} - 1 - \frac{0.3^2 N/M_r}{1 - 0.3/1.1} \right) \\
&= \frac{N}{K_t M_r} \left(0.3 - \frac{M_r}{N} - \frac{0.3^2}{1 - 0.3/1.1} \right) \\
&\overset{(b)}{\geq} \frac{N}{K_t M_r} \left(0.3 - 0.092 - \frac{0.3^2}{1 - 0.3/1.1} \right) \\
&\geq \frac{1.092}{13.5} \cdot \frac{N}{K_t M_r} \\
&\overset{(c)}{\geq} \frac{d^{-1}(M_r)}{13.5} \qquad \text{in Regime 2,}
\end{aligned} \tag{C.17}$$

where (a) is due to $M_r \geq 1.1 \max\{1, N/K_r\} \geq 1.1$, (b) follows from $M < 0.092N/K_t \leq 0.092N$, and (c) uses (C.16).

Regime 3: $0.092N/K_t < M_r \leq N$

By the convexity of $d^{-1}(M_r)$, we have for all $M_r \in (0.092N/K_t, N]$,

$$\begin{aligned}
d^{-1}(M_r) &\leq d^{-1}(0.092N/K_t) - \frac{d^{-1}(0.092N/K_t) - d^{-1}(N)}{N - 0.092N/K_t} \cdot \left(M_r - 0.092 \frac{N}{K_t} \right) \\
&\overset{(a)}{=} d^{-1}(0.092N/K_t) \left(1 - \frac{M_r - 0.092N/K_t}{N - 0.092N/K_t} \right) \\
&= d^{-1}(0.092N/K_t) \left(\frac{N - M_r}{N - 0.092N/K_t} \right) \\
&= \frac{d^{-1}(0.092N/K_t)}{1 - 0.092/K_t} \left(1 - \frac{M_r}{N} \right) \\
&\leq \frac{d^{-1}(0.092N/K_t)}{0.908} \left(1 - \frac{M_r}{N} \right),
\end{aligned} \tag{C.18}$$

where (a) uses that $d^{-1}(N) = 0$.

Let $\tilde{M}_r$ be the largest integer multiple of N/K_r that is no greater than $0.092N/K_t$, and define

$\tilde{\kappa} = K_r\tilde{M}_r/N$. Note that $\tilde{\kappa}$ is an integer. Then,

$$0 \overset{(a)}{\leq} 0.092N/K_t - N/K_r < \tilde{M}_r \leq 0.092N/K_t,$$

where (a) follows from (4.15). This implies that $\tilde{\kappa} + 1 \geq 0.092K_r/K_t$. Since $d^{-1}(M_r)$ is non-increasing,

$$\begin{aligned} d^{-1}(0.092N/K_t) &\leq d^{-1}(\tilde{M}_r) \\ &\overset{(a)}{\leq} 1 + \frac{K_r}{K_t(\tilde{\kappa}+1)} \\ &\leq 1 + \frac{K_r}{K_t \cdot 0.092K_r/K_t} \\ &= \frac{1.092}{0.092}, \end{aligned} \tag{C.19}$$

where (a) follows the same steps that led to (C.15). Combining (C.19) with (C.18),

$$d^{-1}(M_r) \leq \frac{1.092}{0.092 \cdot 0.908}\left(1 - \frac{M_r}{N}\right) \leq 13.1\left(1 - \frac{M_r}{N}\right). \tag{C.20}$$

By applying Lemma 8 with $s = 1$ again, we obtain

$$\frac{1}{\mathsf{DoF}} \geq 1 - \frac{M_r}{N} \overset{(a)}{\geq} \frac{d^{-1}(M_r)}{13.5} \qquad \text{in Regime 3,} \tag{C.21}$$

where (a) follows from (C.20).

Synthesis

The inequalities in (C.12), (C.14), (C.17), and (C.21) cover all possible regimes. Therefore, they together give $1/\mathsf{DoF} \geq d^{-1}(M_r)/13.5$, or equivalently

$$\mathsf{DoF} \leq 13.5 \cdot d\left(N, K_t, K_r, M_t, M_r\right),$$

for all N, K_t, K_r, $M_t \in [0, N]$, and $M_r \in [0, N]$. This concludes the converse proof of Theorem 15. ■

C.4 Communication Problem Outer Bounds (Converse Proof of Theorem 16)

In Section 4.4 we have described the separation architecture and the communication problem that emerges from it. We call the communication problem the multiple multicast X-channel problem. We state its DoF $\tilde{d}^{\star}_{\sigma}(K_t, K_r)$ in Theorem 16 and show that it is achievable by interference alignment in Section 4.5. In this appendix, we prove its optimality by deriving matching information-theoretic outer bounds. Specifically, we want to prove

$$\tilde{d}^{\star}_{\sigma}(K_t, K_r) \leq \frac{1}{K_t\binom{K_r-1}{\sigma-1} + \binom{K_r-1}{\sigma}}, \tag{C.22}$$

for all K_t and K_r.

Consider the following subset of messages:

$$\mathcal{V} = \{V_{\mathcal{S}j} : 1 \in \mathcal{S} \text{ or } j = 1\}. \tag{C.23}$$

It will be convenient to split $\mathcal{V}$ into two disjoint parts,

$$\mathcal{V}^r = \{V_{\mathcal{S}j} : 1 \in \mathcal{S}, j \in \{1, \ldots, K_t\}\},$$
$$\mathcal{V}^t = \{V_{\mathcal{S}j} : 1 \notin \mathcal{S}, j = 1\}.$$

In what follows, we will only focus on $\mathcal{V}$. All other messages, collectively denoted by

$$\overline{\mathcal{V}} = \{V_{\mathcal{S}j} : 1 \notin \mathcal{S} \text{ and } j \neq 1\},$$

are made available to everyone by a genie. Furthermore, we lower the noise at receiver one by a fixed (non-vanishing) amount. Specifically, we replace $\mathbf{y}_1$ by

$$\tilde{\mathbf{y}}_1 = \sum_{j=1}^{K_t} \mathbf{H}_{ij}\mathbf{x}_j + \tilde{\mathbf{z}}_1,$$

where $\tilde{z}_1(\tau)$ are independent zero-mean Gaussian variables with variance

$$\text{var}\left(\tilde{z}_1(\tau)\right) = \min_{i=1,\ldots,K_r} \left(\frac{h_{11}(\tau)}{h_{i1}(\tau)}\right)^2. \tag{C.24}$$

Note that $\text{var}(\tilde{z}_1(\tau)) \leq 1$ since we can set $i = 1$ in (C.24). Hence all the above changes can only improve capacity.

Consider all the receivers other than receiver one. Let a genie also give all of these receivers the subset $\mathcal{V}^r$. Again, this can only improve capacity. Hence, these receivers are given $\mathcal{V}^r \cup \overline{\mathcal{V}}$, which consists of all the messages that receiver one should decode, as well as all the messages of all transmitters other than transmitter one. Using this genie-given knowledge, every receiver can compute $\mathbf{x}_j$ for all $j \neq 1$, and subtract all of them out of their output $\mathbf{y}_i$. In other words, receiver $i \neq 1$ can compute

$$\mathbf{y}_i' = \mathbf{y}_i - \sum_{j \neq 1} \mathbf{H}_{ij}\mathbf{x}_j = \mathbf{H}_{i1}\mathbf{x}_1 + \mathbf{z}_i. \tag{C.25}$$

Receiver i is still expected to decode some messages. Specifically, it must decode the subset of $\mathcal{V}^t$ that is intended for it, i.e.,

$$\mathcal{V}_i^t = \left\{V_{\mathcal{S}j} : 1 \notin \mathcal{S} \text{ and } i \in \mathcal{S} \text{ and } j = 1\right\}.$$

Then, by Fano's inequality,

$$H\left(\mathcal{V}_i^t | \mathbf{y}_i, \mathcal{V}^r, \overline{\mathcal{V}}\right) \leq \varepsilon T. \tag{C.26}$$

We focus now on receiver one. From the problem requirements, it should be able to decode all of $\mathcal{V}^r$ with high probability. After decoding $\mathcal{V}^r$, it has access to all the messages that receiver $i \neq 1$ has, and hence it too can subtract out $\mathbf{x}_j$, $j \neq 1$, from its output,

$$\tilde{\mathbf{y}}_1' = \tilde{\mathbf{y}}_1 - \sum_{j \neq 1} \mathbf{H}_{1j}\mathbf{x}_j + \tilde{\mathbf{z}}_1 = \mathbf{H}_{11}\mathbf{x}_1 + \tilde{\mathbf{z}}_1.$$

Since $\mathbf{H}_{11}$ is invertible almost surely, receiver one can then transform its output to get a statistical

equivalent of the output of any other receiver. Indeed, it can compute

$$\breve{\mathbf{y}}_1^{(i)} = \mathbf{H}_{i1}\mathbf{H}_{11}^{-1}\tilde{\mathbf{y}}_1' = \mathbf{H}_{i1}\mathbf{x}_1 + \left(\mathbf{H}_{i1}\mathbf{H}_{11}^{-1}\right)\tilde{\mathbf{z}}_1 = \mathbf{H}_{i1}\mathbf{x}_1 + \breve{\mathbf{z}}_1^{(i)}.$$

Since $\breve{\mathbf{z}}_1^{(i)} = \left(\mathbf{H}_{i1}\mathbf{H}_{11}^{-1}\right)\tilde{\mathbf{z}}_1$ and the $\mathbf{H}_{ij}$ matrices are diagonal, then the variables $\breve{z}_1^{(i)}(\tau)$ are independent and have a variance of

$$\mathrm{var}\left(\breve{z}_1^{(i)}(\tau)\right) = \mathrm{var}\left(\frac{h_{i1}(\tau)}{h_{11}(\tau)}\tilde{z}_1(\tau)\right) = \left(\frac{h_{i1}(\tau)}{h_{11}(\tau)}\right)^2 \cdot \mathrm{var}\left(\tilde{z}_1(\tau)\right) \leq 1,$$

by (C.24). As a result, receiver one has at least as good a channel output as $\mathbf{y}_i'$ in (C.25), and can thus decode anything that receiver i can. In particular, it can decode $\mathcal{V}_i^t$ for all i, i.e.,

$$H\left(\mathcal{V}_i^t \middle| \tilde{\mathbf{y}}_1, \mathcal{V}^r, \overline{\mathcal{V}}\right) \leq H\left(\mathcal{V}_i^t \middle| \mathbf{y}_i, \mathcal{V}^r, \overline{\mathcal{V}}\right) \leq \varepsilon T, \tag{C.27}$$

using (C.26).

All of the above can be mathematically expressed in the following chain of inequalities, for any achievable $\tilde{R}_\sigma$.

$$\begin{aligned}
|\mathcal{V}| \cdot \tilde{R}_\sigma T &= H\left(\mathcal{V}\right) \\
&\overset{(a)}{=} H\left(\mathcal{V}\middle|\overline{\mathcal{V}}\right) \\
&= I\left(\mathcal{V}; \tilde{\mathbf{y}}_1 \middle| \overline{\mathcal{V}}\right) + H\left(\mathcal{V}\middle|\tilde{\mathbf{y}}_1, \overline{\mathcal{V}}\right) \\
&= I\left(\mathcal{V}; \tilde{\mathbf{y}}_1 \middle| \overline{\mathcal{V}}\right) + H\left(\mathcal{V}^r\middle|\tilde{\mathbf{y}}_1, \overline{\mathcal{V}}\right) + H\left(\mathcal{V}^t\middle|\tilde{\mathbf{y}}_1, \mathcal{V}^r, \overline{\mathcal{V}}\right) \\
&\overset{(b)}{\leq} I\left(\mathcal{V}; \tilde{\mathbf{y}}_1 \middle| \overline{\mathcal{V}}\right) + \varepsilon T + H\left(\mathcal{V}^t\middle|\tilde{\mathbf{y}}_1, \mathcal{V}^r, \overline{\mathcal{V}}\right) \\
&\overset{(c)}{\leq} I\left(\mathcal{V}; \tilde{\mathbf{y}}_1 \middle| \overline{\mathcal{V}}\right) + \varepsilon T + \sum_{i\neq 1} H\left(\mathcal{V}_i^t\middle|\tilde{\mathbf{y}}_1, \mathcal{V}^r, \overline{\mathcal{V}}\right) \\
&\overset{(d)}{\leq} I\left(\mathcal{V}; \tilde{\mathbf{y}}_1 \middle| \overline{\mathcal{V}}\right) + \varepsilon T + \sum_{i\neq 1} \varepsilon T \\
&\overset{(e)}{\leq} I\left(\mathbf{x}_1, \ldots, \mathbf{x}_{K_t}; \tilde{\mathbf{y}}_1 \middle| \overline{\mathcal{V}}\right) + K_r \varepsilon T \\
&\overset{(f)}{\leq} T \cdot \left(\frac{1}{2}\log \mathsf{SNR} + o(\log \mathsf{SNR})\right) + K_r \varepsilon T.
\end{aligned}$$

In the above,

- (a) is due to the independence of the messages;
- (b) uses Fano's inequality for receiver one;
- (c) follows from observing that $\mathcal{V}^t = \bigcup_{i \neq 1} \mathcal{V}_i^t$;
- (d) uses (C.27);
- (e) is due to the data processing inequality; and
- (f) is the MAC channel bound.

By taking $T \to \infty$ and $\varepsilon \to 0$, as well as $\mathsf{SNR} \to \infty$, we obtain

$$\tilde{d}_\sigma^\star(K_t, K_r) \leq \frac{1}{|\mathcal{V}|} = \frac{1}{|\mathcal{V}^r| + |\mathcal{V}^t|} = \frac{1}{K_t \binom{K_r - 1}{\sigma - 1} + \binom{K_r - 1}{\sigma}},$$

thus proving (C.22) and the converse part of Theorem 16. ■

C.5 Lemmas from [54]

In our interference alignment strategy, we use two crucial lemmas from [54]. We present them here for ease of reference.

Lemma 19 (from [54, Lemma 2])**.** *Let $\mathbf{G}_1, \ldots, \mathbf{G}_\Gamma$ be $T \times T$ diagonal matrices, such that $G_g(\tau)$, the τ-th diagonal entry of $\mathbf{G}_g$, follows a continuous distribution when conditioned on all other entries of all matrices. Also let $\mathbf{b}$ be a column vector whose entries $b(\tau)$ are drawn iid from some continuous distribution, independently of $\mathbf{G}_1, \ldots, \mathbf{G}_\Gamma$. Then, almost surely for any integer n such that $T > (n+1)^\Gamma$, there exist matrices $\mathbf{A}_1$ and $\mathbf{A}_2$, of sizes $T \times (n+1)^\Gamma$ and $T \times n^\Gamma$ respectively, such that:*

- *Every entry in the τ-th row of $\mathbf{A}_1$ is a unique multi-variate monomial function of $b(\tau)$ and $G_g(\tau)$ for all g ($b(\tau)$ and $G_g(\tau)$ appear with non-zero exponents in every entry), and the same is true for $\mathbf{A}_2$,[2] and*

[2] To clarify: a monomial could appear in both matrices $\mathbf{A}_1$ and $\mathbf{A}_2$, but never twice in the same matrix.

- *The matrices satisfy the following conditions almost surely,*

$$\mathbf{G}_g\mathbf{A}_2 \prec \mathbf{A}_1, \quad \forall g = 1, \ldots, \Gamma,$$

where $\mathbf{P} \prec \mathbf{Q}$ *means that the span of the columns of* $\mathbf{P}$ *is a subspace of the space spanned by the columns of* $\mathbf{Q}$*.*

Lemma 20 (from [54, Lemma 1])**.** *Let* $x_i^{(k)}$*,* $i = 1, \ldots, T$ *and* $k = 1, \ldots, K$*, be random variables such that each follows a continuous distribution when conditioned on all other variables. Let* $\mathbf{\Psi}$ *be a* $T \times T$ *square matrix with entries* ψ_{ij} *such that*

$$\psi_{ij} = \prod_{k=1}^{K} \left(x_i^{(k)}\right)^{p_{ij}^{(k)}},$$

where $p_{ij}^{(k)}$ *are integers such that*

$$\left(p_{ij}^{(1)}, \ldots, p_{ij}^{(K)}\right) \neq \left(p_{ij'}^{(1)}, \ldots, p_{ij'}^{(K)}\right),$$

for all i, j, j' *such that* $j \neq j'$*. In other words, the entries* ψ_{ij} *are distinct monomials in the variables* $x_i^{(k)}$*. Then, the matrix* $\mathbf{\Psi}$ *is almost surely full rank.*

C.6 Proof of Theorem 17

Theorem 17 gives an improved achievable DoF for the 2×2 cache-aided interference network. In this appendix, we prove this result by describing and analyzing the interference-extraction scheme introduced in Section 4.7 and illustrated in Figure 4.10, which achieves this DoF.

We describe the scheme in two steps. First, we focus on the physical layer to show how more information can be extracted from the aligned interference at the receivers. Second, we show how this additional information can be used at the network layer to achieve the inverse DoF in Theorem 17.

C.6.1 Physical Layer

In order to describe the interference-extraction scheme, let us first revisit the original separation architecture used when $M_r = 0$. The message set used for this case is the one where every transmitter has a message for every individual receiver, i.e., the unicast X-channel message set. In order to achieve the optimal communication DoF of $1/3$ per message, at every receiver, the two messages intended for the other receiver are aligned in the same subspace. Let us study this alignment more carefully.

Let V_{ij} be the message intended for receiver i from transmitter j. Represent every message V_{ij} by a scalar v_{ij}, called a stream. By taking a block length of 3 and by beamforming message V_{ij} along some direction $\mathbf{a}_{ij}$, we get channel inputs

$$\mathbf{x}_j = \mathbf{a}_{1j}v_{1j} + \mathbf{a}_{2j}v_{2j},$$

and channel outputs

$$\mathbf{y}_1 = \mathbf{H}_{11}\left(\mathbf{a}_{11}v_{11} + \mathbf{a}_{21}v_{21}\right) + \mathbf{H}_{12}\left(\mathbf{a}_{12}v_{12} + \mathbf{a}_{22}v_{22}\right) + \mathbf{z}_1; \tag{C.28a}$$

$$\mathbf{y}_2 = \mathbf{H}_{21}\left(\mathbf{a}_{11}v_{11} + \mathbf{a}_{21}v_{21}\right) + \mathbf{H}_{22}\left(\mathbf{a}_{12}v_{12} + \mathbf{a}_{22}v_{22}\right) + \mathbf{z}_2. \tag{C.28b}$$

Lemma 21. *We can choose the $\mathbf{a}_{ij}$'s in* (C.28) *such that*

$$\mathbf{y}_1 = \mathbf{\Psi}_1 \begin{bmatrix} v_{11} \\ v_{12} \\ v_{21} + v_{22} \end{bmatrix} + \mathbf{z}_1;$$

$$\mathbf{y}_2 = \mathbf{\Psi}_2 \begin{bmatrix} v_{21} \\ v_{22} \\ v_{11} + v_{12} \end{bmatrix} + \mathbf{z}_2,$$

where the 3×3 matrices $\mathbf{\Psi}_i$ are full-rank almost surely.

Proof: Recall that the $\mathbf{H}_{ij}$'s are 3×3 diagonal matrices whose τ-th diagonal element is $h_{ij}(\tau)$. Also recall that these $h_{ij}(\tau)$ are independent and continuously distributed, which implies

that $\mathbf{H}_{ij}$ is invertible almost surely. Assume this invertibility is the case in the following.

Choose the $\mathbf{a}_{ij}$ vectors as:

$$\mathbf{a}_{11} = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}; \qquad \mathbf{a}_{12} = \mathbf{H}_{22}^{-1}\mathbf{H}_{21}\mathbf{a}_{11};$$

$$\mathbf{a}_{21} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}; \qquad \mathbf{a}_{22} = \mathbf{H}_{12}^{-1}\mathbf{H}_{11}\mathbf{a}_{21}.$$

From (C.28), the received signals are then

$$\begin{aligned}
\mathbf{y}_1 &= \mathbf{H}_{11}\mathbf{a}_{11}v_{11} + \mathbf{H}_{12}\mathbf{H}_{22}^{-1}\mathbf{H}_{21}\mathbf{a}_{11}v_{12} + \mathbf{H}_{11}\mathbf{a}_{21}\left(v_{21}+v_{22}\right) + \mathbf{z}_1 \\
&= \begin{bmatrix} \mathbf{H}_{11}\mathbf{a}_{11} & \mathbf{H}_{12}\mathbf{H}_{22}^{-1}\mathbf{H}_{21}\mathbf{a}_{11} & \mathbf{H}_{11}\mathbf{a}_{21} \end{bmatrix} \begin{bmatrix} v_{11} \\ v_{12} \\ v_{21}+v_{22} \end{bmatrix} + \mathbf{z}_1 \\
&= \underbrace{\begin{bmatrix} h_{11}(1) & \frac{h_{12}(1)h_{21}(1)}{h_{22}(1)} & h_{11}(1) \\ h_{11}(2) & \frac{h_{12}(2)h_{21}(2)}{h_{22}(2)} & 0 \\ 0 & 0 & h_{11}(3) \end{bmatrix}}_{\mathbf{\Psi}_1} \begin{bmatrix} v_{11} \\ v_{12} \\ v_{21}+v_{22} \end{bmatrix} + \mathbf{z}_1,
\end{aligned}$$

and, in a similar way,

$$\mathbf{y}_2 = \underbrace{\begin{bmatrix} h_{21}(1) & \frac{h_{22}(1)h_{11}(1)}{h_{12}(1)} & h_{21}(1) \\ 0 & 0 & h_{21}(2) \\ h_{21}(3) & \frac{h_{22}(3)h_{11}(3)}{h_{12}(3)} & 0 \end{bmatrix}}_{\mathbf{\Psi}_2} \begin{bmatrix} v_{21} \\ v_{22} \\ v_{11}+v_{12} \end{bmatrix} + \mathbf{z}_2.$$

Since the $h_{ij}(\tau)$ are independent continuously distributed variables, then the matrices $\mathbf{\Psi}_1$ and $\mathbf{\Psi}_2$ are full-rank almost surely. ■

Notice from Lemma 21 that each receiver can recover, in addition to its intended streams, the

sum of the streams intended for the other receiver. By using a linear outer code over some finite field, we can ensure that obtaining the sum of two streams, e.g., $v_{21} + v_{22}$, yields the sum of the two corresponding messages, e.g., $V_{21} \oplus V_{22}$, where $\oplus$ indicates addition over the finite field. For simplicity, we assume that this field is $\mathsf{GF}(2)$, although any finite field gives the same result. In other words, receiver one can decode V_{11}, V_{12}, and $(V_{21} \oplus V_{22})$, and receiver two can decode V_{21}, V_{22}, and $(V_{11} \oplus V_{12})$. Therefore, for the same per-message DoF of $1/3$, we get the linear combinations of the unintended messages for free.

C.6.2 Network Layer

Figure 4.10 illustrates the interface between the physical and network layers resulting from the decoding of the aligned interference at each receiver. This aligned interference, while available for free (no drawbacks in the communication DoF at the physical layer), becomes useful when the receiver memory is non-zero. It provides a middle ground between pure unicast messages (as is done at $M_r = 0$) and pure broadcast messages (which we use when $M_r = 1$).

Let ℓ denote the link load, i.e., the size of each message V_{ij}, and let $L = 4\ell$ be the sum network load. For this specific separation architecture, we denote by $L^\star(M_r)$ the smallest sum network load as a function of receiver memory M_r, and by $\ell^\star = L^\star/4$ the smallest individual link load. Since each message V_{ij} (link) can be communicated across the physical layer using a DoF of $1/3$ by Lemma 21, then we can achieve an end-to-end DoF of

$$\frac{1}{\mathsf{DoF}} \leq \frac{\ell^\star}{1/3} = \frac{3}{4} L^\star. \tag{C.29}$$

Theorem 17 follows directly from combining (C.29) with the following lemma.

Lemma 22. *For the separation architecture illustrated in Figure 4.10, we can achieve the following sum network load:*

$$L^\star(M_r) \leq \max\left\{2 - 2M_r, \frac{12}{7} - \frac{8}{7}M_r, \frac{4}{3} - \frac{2}{3}M_r\right\},$$

for $M_r \in [0, 2]$.

Proof: In order to prove Lemma 22, it suffices to look at the following four (M_r, L) corner

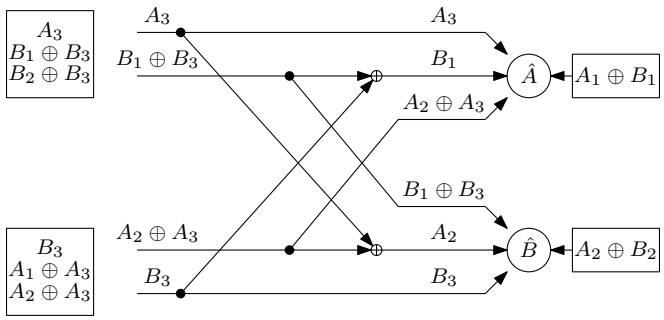


Figure C.1: Strategy for $M_r = 1/3$, when the user requests are (A, B).

Table C.2: Achievable strategy for $M_r = 1/3$.

Cache	Content	Rx
Tx 1	$A_3, B_1 \oplus B_3, B_2 \oplus B_3$	N/A
Tx 2	$B_3, A_1 \oplus A_3, A_2 \oplus A_3$	N/A
Rx 1	$A_1 \oplus B_1$	1
Rx 2	$A_2 \oplus B_2$	2

	Demands (Rx1, Rx2)				
Message	(A, A)	(A, B)	(B, A)	(B, B)	Rx
V_{11}	A_3	A_3	$B_2 \oplus B_3$	$B_1 \oplus B_3$	1
V_{21}	A_3	$B_1 \oplus B_3$	A_3	$B_2 \oplus B_3$	2
V_{12}	$A_1 \oplus A_3$	$A_2 \oplus A_3$	B_3	B_3	1
V_{22}	$A_2 \oplus A_3$	B_3	$A_1 \oplus A_3$	B_3	2
$V_{21} \oplus V_{22}$	A_2	B_1	A_1	B_2	1
$V_{11} \oplus V_{12}$	A_1	A_2	B_2	B_1	2

points, as the rest can be achieved using time- and memory-sharing:

$$(0, 2)\,, \quad (1/3, 4/3)\,, \quad (4/5, 4/5)\,, \quad (2, 0)\,.$$

The fourth corner point is trivial since $M_r = 2$ implies each user can cache the entire library, and hence there is no need to transmit any information across the network. The first corner point can be achieved by ignoring the aligned interference messages, which reduces to the original strategy. Therefore, we only need to show the achievability of the second and third corner points. For convenience, we will call the two files in the content library A and B.

Achieving point $(M_r, L) = (1/3, 4/3)$ When $M_r = 1/3$, we split each file into three equal parts, labeled $A = (A_1, A_2, A_3)$ and $B = (B_1, B_2, B_3)$.

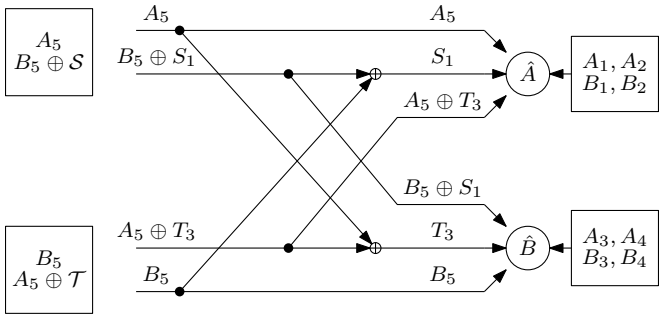


Figure C.2: Strategy for $M_r = 4/5$, when the user requests are (A, B).

Table C.2 shows the placement and delivery phases, for all possible user requests, and Figure C.1 illustrates the strategy when the demands are (A, B). Notice that the transmitter caches hold exactly one file each (thus $M_t = 1$), the receivers cache one third of a file each ($M_r = 1/3$). Furthermore, the messages V_{ij} each carry the equivalent of one third of a file, which implies that $\ell = 1/3$ is achieved, or, equivalently, a sum network load of $L = 4\ell = 4/3$.

Achieving point $(M_r, L) = (4/5, 4/5)$ When $M_r = 4/5$, we split each file into five equal parts, labeled $A = (A_1, \ldots, A_5)$ and $B = (B_1, \ldots, B_5)$. For convenience, we define

$$S_1 = B_2 \oplus A_4, \quad S_2 = A_1 \oplus B_3, \quad S_3 = B_1 \oplus B_3, \quad S_4 = B_2 \oplus B_4,$$
$$T_1 = A_1 \oplus A_3, \quad T_2 = A_2 \oplus A_4, \quad T_3 = B_1 \oplus A_3, \quad T_4 = A_2 \oplus B_4,$$

and write $\mathcal{S} = \{S_1, S_2, S_3, S_4\}$ and $\mathcal{T} = \{T_1, T_2, T_3, T_4\}$.

Table C.3 shows the placement and delivery phases, for all possible user requests, and Figure C.2 illustrates the strategy when the demands are (A, B). Notice that the transmitter caches hold exactly one file each (thus $M_t = 1$), the receivers cache four fifths of a file each ($M_r = 4/5$). Furthermore, the messages V_{ij} each carry the equivalent of one fifth of a file, which implies that $\ell = 1/5$ is achieved, or, equivalently, a sum network load of $L = 4\ell = 4/5$.

By achieving all four corner points, we have proved Lemma 22. ■

Table C.3: Achievable strategy for $M_r = 4/5$.

Cache	Content				Rx
Tx 1	$A_5, B_5 \oplus S_1, B_5 \oplus S_2, B_5 \oplus S_3, B_5 \oplus S_4$				N/A
Tx 2	$B_5, A_5 \oplus T_1, A_5 \oplus T_2, A_5 \oplus T_3, A_5 \oplus T_4$				N/A
Rx 1	A_1, A_2, B_1, B_2				1
Rx 2	A_3, A_4, B_3, B_4				2
	Demands (Rx1, Rx2)				
Message	(A, A)	(A, B)	(B, A)	(B, B)	Rx
V_{11}	A_5	A_5	$B_5 \oplus S_2$	$B_5 \oplus S_3$	1
V_{21}	A_5	$B_5 \oplus S_1$	A_5	$B_5 \oplus S_4$	2
V_{12}	$A_5 \oplus T_1$	$A_5 \oplus T_3$	B_5	B_5	1
V_{22}	$A_5 \oplus T_2$	B_5	$A_5 \oplus T_4$	B_5	2
$V_{21} \oplus V_{22}$	T_2	S_1	T_4	S_4	1
$V_{11} \oplus V_{12}$	T_1	T_3	S_2	S_3	2

C.6.3 Optimality Within the Considered Separation Architecture

Within the separation architecture considered throughout this appendix and Section 4.7, i.e., the one illustrated in Figure 4.10, we can show that the network-layer scheme is in fact exactly optimal. Specifically, the sum network load achieved in Lemma 22 is optimal. This is summarized in the following result.

Proposition 4. *For all M_r, the optimal sum network load must satisfy*

$$L^\star(M_r) \geq \max\left\{2 - 2M_r, \frac{12}{7} - \frac{8}{7}M_r, \frac{4}{3} - \frac{2}{3}M_r\right\}.$$

While this does not contribute to the main result in Theorem 17, it does reinforce it by showing that this is the best we can do within this separation architecture.

Proof: For the proof, it is more convenient to write the outer bounds in terms of the individual link load $\ell^\star = L^\star/4$. Therefore, we will prove Proposition 4 by proving the following three inequalities (which together constitute an equivalent result):

$$4\ell^\star + 2M_r \geq 2;$$

$$7\ell^\star + 2M_r \geq 3;$$

$$6\ell^\star + \ \ M_r \geq 2.$$

In the following, we refer to the two files as A and B. Let the cache contents of receivers one and two be Q_1 and Q_2, respectively. We also write V_{ij}^{ST} to denote the message V_{ij} when user one has requested file S and user two has requested file T, where $S, T \in \{A, B\}$. Furthermore, we use $\mathcal{V}^{ST}$ to refer to all four messages when the requests are S and T, and write $\mathcal{Y}_i^{ST}$ to denote the three outputs at receiver $i \in \{1, 2\}$ when the requests are S and T. Therefore,

$$\begin{aligned}
\mathcal{V}^{ST} &= \left(V_{11}^{ST}, V_{12}^{ST}, V_{21}^{ST}, V_{22}^{ST}\right); \\
\mathcal{Y}_1^{ST} &= \left(V_{11}^{ST}, V_{12}^{ST}, V_{21}^{ST} \oplus V_{22}^{ST}\right); \\
\mathcal{Y}_2^{ST} &= \left(V_{21}^{ST}, V_{22}^{ST}, V_{11}^{ST} \oplus V_{12}^{ST}\right).
\end{aligned}$$

We will next prove each of the three inequalities.

First inequality

$$\begin{aligned}
(4\ell^\star + 2M_r)F &\geq H\left(Q_1, Q_2, \mathcal{V}^{AB}\right) \\
&= H\left(Q_1, Q_2, \mathcal{V}^{AB} \middle| A, B\right) + I\left(A, B; Q_1, Q_2, \mathcal{V}^{AB}\right) \\
&= H\left(Q_1, Q_2, \mathcal{V}^{AB} \middle| A, B\right) + H\left(A, B\right) - H\left(A, B \middle| Q_1, Q_2, \mathcal{V}^{AB}\right) \\
&\overset{(a)}{\geq} H\left(A, B\right) - \varepsilon F \\
&= 2F - \varepsilon F,
\end{aligned}$$

where (a) is due to Fano's inequality. Therefore,

$$4\ell^\star + 2M_r \geq 2.$$

Second inequality

$$\begin{aligned}(7\ell^\star + 2M_r)F &\geq H\left(Q_1, \mathcal{Y}_1^{AB}\right) + H\left(Q_2, \mathcal{V}^{BA}\right) \\ &\overset{(a)}{\geq} H\left(Q_1, \mathcal{Y}_1^{AB}\middle|A\right) + H\left(Q_2, \mathcal{V}^{BA}\middle|A\right) + 2H\left(A\right) - 2\varepsilon F \\ &\geq H\left(Q_1, Q_2, \mathcal{Y}_1^{AB}, \mathcal{V}^{BA}\middle|A\right) + 2H\left(A\right) - 2\varepsilon F \\ &\overset{(b)}{\geq} H\left(Q_1, Q_2, \mathcal{Y}_1^{AB}, \mathcal{V}^{BA}\middle|A, B\right) + H\left(B\right) + 2H\left(A\right) - 3\varepsilon F \\ &\geq 3F - 3\varepsilon F,\end{aligned}$$

where (a) and (b) once again follow from Fano's inequality. Therefore,

$$7\ell^\star + 2M_r \geq 3.$$

Third inequality

$$\begin{aligned}(6\ell^\star + M_r)F &\geq H\left(Q_1, \mathcal{Y}_1^{AA}, \mathcal{Y}_2^{BB}\right) \\ &\overset{(a)}{\geq} H\left(Q_1, \mathcal{Y}_1^{AA}, \mathcal{Y}_2^{BB}\middle|A, B\right) + H\left(A, B\right) - \varepsilon F \\ &\geq 2F - \varepsilon F,\end{aligned}$$

where (a) is again due to Fano's inequality. Therefore,

$$6\ell^\star + M_r \geq 2.$$

This concludes the proof of Proposition 4. ■

C.7 Proofs for the Low-SNR Regime (Section 4.9)

C.7.1 Network-Layer Scheme (Proof of Theorems 18 and 19)

In this appendix, we provide the details of the two network-layer strategies: the multicasting scheme and the beamforming scheme. This includes choosing p and q and determining the corresponding value of v_{pq} that each scheme achieves, as introduced in Section 4.9.3. Combined with Lemma 9,

these imply the achievable rate results in Theorems 18 and 19.

Network-Layer Strategy: The Multicasting Scheme (Proof of Theorem 18)

Suppose $\kappa = KM_r/N$ and $\lambda = LM_t/N$ are both integers. Collectively, the transmitters can hold λ copies of the entire content library. To take advantage of that, we first split every file W_n into $\binom{L}{\lambda}$ equal subfiles $\{W_{n,\mathcal{L}}\}_{\mathcal{L}}$, where the index $\mathcal{L}$ is over all subsets of transmitters of size λ. We can thus create $\binom{L}{\lambda}$ sublibraries: the sublibrary indexed by $\mathcal{L}$ contains the subfile $W_{n,\mathcal{L}}$ of every file W_n. For the transmitter content placement, every transmitter ℓ stores all complete sublibraries indexed by $\mathcal{L}$ such that $\ell \in \mathcal{L}$. The result is that every subset of transmitters of size λ shares exactly one sublibrary.

For the receiver content placement, we first split each receiver cache into $\binom{L}{\lambda}$ equal parts and dedicate each part to one sublibrary. We have thus divided our original problem into $\binom{L}{\lambda}$ subproblems. In each subproblem, a subset $\mathcal{L}$ of transmitters shares a full sublibrary of N subfiles of size $\tilde{F} = F/\binom{L}{\lambda}$ each. Each of the K receivers is equipped with a cache of size $M_r F/\binom{L}{\lambda} = M_r \tilde{F}$ bits, equivalently M_r subfiles. Since $\kappa = KM_r/N$, we can apply the strategy from [4] on this subproblem, which requires that the transmitters send a common message to every subset $\mathcal{K}$ of size $\kappa + 1$ receivers. We can enable that by choosing the message set $\mathscr{V}_{pq}$ with $p = \kappa + 1$ and $q = \lambda$.

Each message $V_{\mathcal{K}\mathcal{L}} \in \mathscr{V}_{pq}$ has size $v_{pq}F$ bits, which can be rewritten in terms of the subfile size $\tilde{F}$ as $v_{pq}F = \binom{L}{\lambda} v_{pq}\tilde{F}$ bits. From [4], we know that the total number of bits that each subproblem needs to transmit across the bit pipes is $(K-\kappa)/(\kappa+1) \cdot \tilde{F}$, shared equally among all the bit pipes. Therefore, the total number of bits sent through the $\binom{K}{\kappa+1}$ messages of each subproblem is

$$\binom{K}{\kappa+1}\binom{L}{\lambda} v_{pq}\tilde{F} = \binom{K}{\kappa+1} v_{pq} F = \frac{K-\kappa}{\kappa+1}\tilde{F}.$$

Consequently, we achieve

$$v_{pq} = \frac{K-\kappa}{\kappa+1} \cdot \frac{1}{\binom{L}{\lambda}\binom{K}{\kappa+1}} \tag{C.30}$$

at the network layer. By combining (C.30) with (4.23) and Lemma 9, we obtain the result of Theorem 18 for κ and λ integers.

Network-Layer Strategy: The Beamforming Scheme (Proof of Theorem 19)

Recall that the beamforming scheme is different from the multicasting scheme in that it completely ignores any possible multicasting gain in favor of a larger beamforming gain.

Suppose $\tilde{\lambda} = \min\{LM_t/(N - M_r), L\}$ is an integer. The first step is to divide each file W_n into $\binom{L}{\tilde{\lambda}} + 1$ parts,

$$W_n = \left(W_{n,0}, W_{n,\mathcal{L}} : \mathcal{L} \subseteq [L], |\mathcal{L}| = \tilde{\lambda}\right),$$

such that $W_{n,0}$ has size $M_r F/N$ bits and $W_{n,\mathcal{L}}$ has size $(N - M_r)F/\binom{L}{\tilde{\lambda}}$ for all $\mathcal{L}$.

In the placement phase, every receiver stores $W_{n,0}$ for every n. Thus all receivers have exactly the same side information in their caches. Each transmitter ℓ stores all parts $W_{n,\mathcal{L}}$ such that $\ell \in \mathcal{L}$. Note that this placement satisfies the memory constraints M_r and M_t on the receivers and transmitters respectively.

During the delivery phase, every subset $\mathcal{L}$ of transmitters will beamform to each user k the part of its requested file that these transmitters share. Therefore, the message set that we choose is $\mathscr{V}_{pq}$ with $p = 1$ and $q = \tilde{\lambda}$, and if user k requests file W_{d_k} then we set $V_{\{k\}\mathcal{L}} = W_{d_k,\mathcal{L}}$ for all $\mathcal{L}$. Each message will as a result have a size of $v_{pq} = (N - M_r)/\binom{L}{\tilde{\lambda}}$. Substituting in (4.23) and using Lemma 9, we obtain the rate achieved in Theorem 19.

C.7.2 Physical-Layer Scheme (Proof of Lemma 9)

Recall that we wish to transmit the messages $\mathscr{V}_{pq}$ from (4.22) across the interference network, for some $p \in [K]$ and $q \in [L]$. As previously mentioned, the idea is to wait until a "favorable" channel occurs that allows some subset of transmitters to efficiently beamform some message to all its intended receivers at once. In this proof, we focus on a particular p and a particular q.

Let us focus on one subset pair $(\mathcal{K}, \mathcal{L})$, where $\mathcal{K}$ is a subset of p receivers and $\mathcal{L}$ is a subset of q transmitters. The most "favorable" channel to beamform message $V_{\mathcal{K}\mathcal{L}}$ occurs when the channel gains from the transmitters in $\mathcal{L}$ to each receiver in $\mathcal{K}$ are identical up to a multiplication by a scalar. To be precise, the channel vectors $\mathbf{g}_{k\mathcal{L}} = (g_{k\ell})_{\ell \in \mathcal{L}}$ have to be equal for all $k \in \mathcal{K}$, up to a multiplication by a scalar. However, since there are uncountably many values for each gain, the set of perfect channels has a measure of zero. For this reason, we choose to divide the possible values

of the channel gains into a finite number of bins $\beta \geq 8$.

We will divide this proof into three parts: the first part presents the binning strategy, the second part gives the beamforming strategy and the corresponding analysis, and the third part analyzes the duty cycle, i.e., the fraction of time during which the channel is "favorable" for some transmitters and receivers.

Binning strategy

Recall that the channel gains are phase shifts, $g_{k\ell}(\tau) = e^{j\theta_{k\ell}(\tau)}$, where $\theta_{k\ell}(\tau) \in [0, 2\pi)$ uniformly. For any angle $\theta \in [0, 2\pi)$, define the binning function $B(\theta)$ as the unique integer such that

$$\theta - \frac{2\pi}{\beta} B(\theta) \in [0, 2\pi/\beta).$$

Note that $B(\theta) \in \{0, \ldots, \beta - 1\}$. For each bin b, we define the representative phase of b as the midpoint of all phases that are binned to b, i.e.,

$$\Phi(b) = b \cdot 2\pi/\beta + \pi/\beta.$$

This implies that $|\Phi(B(\theta)) - \theta| \leq \pi/\beta$ for all $\theta \in [0, 2\pi)$. The above-described binning is illustrated in Figure C.3 for a choice of $\beta = 8$. For simplicity, we will define $b_{k\ell}(\tau) = B(\theta_{k\ell}(\tau))$ to be the bin of the channel phase shift $\theta_{k\ell}(\tau)$ and $\phi_{k\ell}(\tau) = \Phi(b_{k\ell}(\tau))$ to be its representative phase.

We use these bins to determine which channels are "favorable" for a subset pair $(\mathcal{K}, \mathcal{L})$. Specifically, we say that a channel is favorable for $(\mathcal{K}, \mathcal{L})$ if the corresponding channel vectors can be mapped to the same bins. More formally, we say that the channel at time τ is *favorable for* $(\mathcal{K}, \mathcal{L})$ if

$$b_{k\ell}(\tau) = b_{k'\ell}(\tau) \quad \forall k, k' \in \mathcal{K}, \forall \ell \in \mathcal{L}.$$

We define $f_{\mathcal{K},\mathcal{L}}(\tau)$ to be one if the channel is favorable for $(\mathcal{K}, \mathcal{L})$ at time τ, and zero otherwise. For every time τ, we then define the set of pairs

$$\mathscr{B}(\tau) = \{(\mathcal{K}, \mathcal{L}) : |\mathcal{K}| = p, |\mathcal{L}| = q, f_{\mathcal{K},\mathcal{L}}(\tau) = 1\}$$

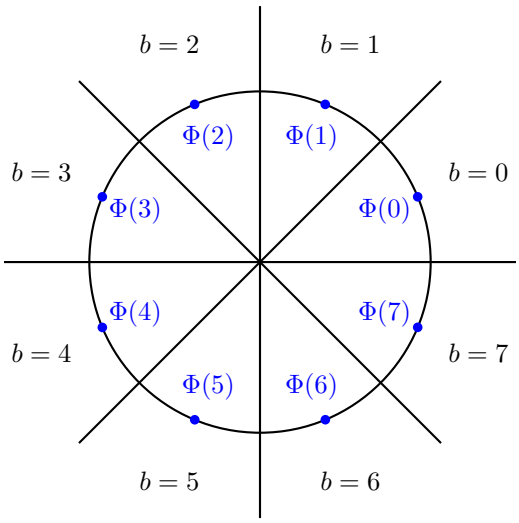


Figure C.3: The $\beta = 8$ bins and their representative phases $\Phi(b)$.

for which the channel is favorable.

Beamforming strategy

First, we encode each message $V_{\mathcal{KL}}$ into a codeword $\mathbf{v}_{\mathcal{KL}}$. For every time τ, we want to choose a pair $(\mathcal{K}, \mathcal{L})$ for which the channel is favorable, if any exist. We denote this pair by $(\mathcal{K}(\tau), \mathcal{L}(\tau))$, but we will ignore the τ index when it is obvious from context for clarity. We then let the transmitters in $\mathcal{L}$ beamform a symbol $v_{\mathcal{KL}}(\tau)$ from $\mathbf{v}_{\mathcal{KL}}$ to the receivers in $\mathcal{K}$.

More formally, write $\mathcal{L} = \{\ell_1, \ldots, \ell_q\}$. Let $\hat{\mathbf{b}}(\tau) = (\hat{b}_{\ell_1}(\tau), \ldots, \hat{b}_{\ell_q}(\tau))$ denote the vector of bins that resulted in the choice of subset pair at time τ, i.e., $\hat{b}_\ell(\tau) = b_{k\ell}(\tau)$ for all $k \in \mathcal{K}$ and $\ell \in \mathcal{L}$. Then, each transmitter $\ell \in \mathcal{L}$ sends

$$x_\ell(\tau) = v_{\mathcal{KL}}(\tau) \cdot e^{-j\Phi(\hat{b}_\ell(\tau))},$$

and each receiver $k \in \mathcal{K}$ observes

$$\begin{aligned} y_k(\tau) &= \sum_{\ell \in \mathcal{L}} e^{j\theta_{k\ell}(\tau)} \cdot e^{-j\Phi(\hat{b}_\ell(\tau))} v_{\mathcal{KL}}(\tau) + z_k(\tau) \\ &= v_{\mathcal{KL}}(\tau) \sum_{\ell \in \mathcal{L}} e^{j(\theta_{k\ell}(\tau) - \Phi(B(\theta_{k\ell}(\tau))))} + z_k(\tau). \end{aligned}$$

The receiver SNR is then

$$|v_{\mathcal{KL}}(\tau)|^2 \cdot \left| \sum_{\ell \in \mathcal{L}} e^{j(\theta_{k\ell}(\tau) - \Phi(B(\theta_{k\ell}(\tau))))} \right|^2 .$$

Because of the binning, we can find a good lower bound on the magnitude of the sum term. Let $\delta_{k\ell}(\tau) = \theta_{k\ell}(\tau) - \Phi(B(\theta_{k\ell}(\tau)))$. Then,

$$\begin{aligned}
\left| \sum_{\ell \in \mathcal{L}} e^{j\delta_{k\ell}(\tau)} \right|^2 &= \left(\sum_{\ell \in \mathcal{L}} e^{j\delta_{k\ell}(\tau)} \right) \left(\sum_{\ell \in \mathcal{L}} e^{-j\delta_{k\ell}(\tau)} \right) \\
&= \sum_{\ell \in \mathcal{L}} \left(1 + 2 \sum_{\ell' > \ell} \Re \left\{ e^{j(\delta_{k\ell}(\tau) - \delta_{k\ell'}(\tau))} \right\} \right) \\
&= \sum_{\ell \in \mathcal{L}} \left(1 + 2 \sum_{\ell' > \ell} \cos(\delta_{k\ell}(\tau) - \delta_{k\ell'}(\tau)) \right) .
\end{aligned}$$

Because $\delta_{k\ell}(\tau) \in [-\pi/\beta, \pi/\beta)$, then

$$\delta_{k\ell}(\tau) - \delta_{k\ell'}(\tau) \in [-2\pi/\beta, 2\pi/\beta],$$

and hence, since $\beta \geq 8$,

$$\cos\left(\delta_{k\ell}(\tau) - \delta_{k\ell'}(\tau)\right) \geq \cos\frac{2\pi}{\beta}.$$

We can write $\cos 2\pi/\beta = (1 - \gamma)$ for some $\gamma > 0$. Consequently,

$$\left| \sum_{\ell \in \mathcal{L}} e^{j\delta_{k\ell}(\tau)} \right|^2 \geq \sum_{\ell \in \mathcal{L}} (1 + (q-1)(1-\gamma)) \geq (1-\gamma)q^2.$$

Supposing that $|v_{\mathcal{KL}}(\tau)|^2 = P'$, and assuming that $V_{\mathcal{KL}}$ is being transmitted during a fraction α of the total block length, we conclude that we can achieve a rate of

$$R'_{pq} \geq \alpha \log_2 \left(1 + (1-\gamma)q^2 \cdot P'\right) \tag{C.31}$$

for message $V_{\mathcal{KL}}$.

Duty cycle analysis and achievable rate

As mentioned previously, our strategy needs to wait for time instants τ such that $\mathscr{B}(\tau)$ is not empty. We refer to the expected fraction of time during which it is not empty as the *duty cycle* η, defined as $\eta = \Pr\{\mathscr{B} \neq \emptyset\}$.

When selecting pairs $(\mathcal{K}, \mathcal{L}) \in \mathscr{B}(\tau)$, it is possible to ensure that all pairs are selected equally likely. For instance, if multiple pairs are possible for a specific τ, we can pick one of them uniformly at random. Thus the duty cycle will be shared equally among all pairs, and the expected fraction of time that any one message is being transmitted is $\alpha = \eta / \binom{L}{q}\binom{K}{p}$. Since each transmitter is active for exactly $\binom{L-1}{q-1}\binom{K}{p}$ pairs out of the $\binom{L}{q}\binom{K}{p}$ total, then every transmitter will be active for a fraction

$$\eta \cdot \frac{q}{L}$$

of the time in expectation. Consequently, it can scale its power by $L/\eta q$ during its duty cycle, which means

$$P' = \frac{L}{\eta q} P.$$

By appealing to the law of large numbers, it then follows from (C.31) that the set $\mathscr{V}_{pq}$ can be transmitted at a sum rate of

$$\binom{L}{q}\binom{K}{p} R'_{pq} \geq \eta \cdot \log_2 \left(1 + \frac{(1-\gamma)Lq}{\eta} P\right).$$

When $P \leq \sigma \cdot \eta / (1-\gamma)Lq$ for some $\sigma > 0$, we get

$$\binom{L}{q}\binom{K}{p} R'_{pq} \geq (1-\gamma)Lq \cdot \frac{\log_2(1+\sigma)}{\sigma} \cdot P, \tag{C.32}$$

by using $x \in [0, x_0] \implies \log_2(1+x) \geq x \cdot \log_2(1+x_0)/x_0$ for any $x_0 > 0$.

All that remains is to find a lower bound on the duty cycle η, in order to get a sufficient condition for the critical power necessary for (C.32) to hold. Consider the probability that a single subset

pair $(\mathcal{K}, \mathcal{L})$ gets a favorable channel at time τ. Recall that a channel is favorable for this pair if

$$b_{k\ell}(\tau) = b_{k'\ell}(\tau)$$

for all $k, k' \in \mathcal{K}$ and $\ell \in \mathcal{L}$. Without loss of generality, we can assume that $b_{k1}(\tau) = 0$ for all receivers k since each receiver can always multiply its channel output with the correct phase shift. Therefore, the above happens at time τ with probability

$$\Pr\{f_{\mathcal{K},\mathcal{L}}(\tau) = 1\} = \beta^{-(p-1)(q-1)}.$$

Consequently,

$$\begin{aligned}
\eta &= \Pr\{\mathscr{B} \neq \emptyset\} \\
&= \Pr\{\exists(\mathcal{K}, \mathcal{L}) : f_{\mathcal{K},\mathcal{L}}(\tau) = 1\} \\
&\overset{(a)}{\geq} \Pr\{f_{\mathcal{K}_0,\mathcal{L}_0}(\tau) = 1\} \\
&= \beta^{-(p-1)(q-1)},
\end{aligned}$$

for some arbitrary pair $(\mathcal{K}_0, \mathcal{L}_0)$. Note that the inequality (a) is quite loose; in practice the duty cycle should be higher because of the possibility to schedule all the $\binom{L}{q}\binom{K}{p}$ messages, and thus the critical power required for this analysis is higher.

Using this in (C.32), we get that

$$\binom{L}{q}\binom{K}{p} R'_{pq} \geq (1-\gamma) Lq \cdot \frac{\log_2(1+\sigma)}{\sigma} \cdot P$$

bits per channel use, whenever $P \leq \beta^{-(p-1)(q-1)}\sigma/(1-\gamma)Lq$.

Since $1 - \gamma = \cos 2\pi/\beta$, we can make γ arbitrarily small by increasing the number of bins β. Similarly, we know that $\log_2(1+\sigma)/\sigma$ approaches $1/\ln 2$ as σ approaches zero. Therefore, for any $\varepsilon > 0$, we can choose particular values of β and σ so that, for a small enough P,

$$\binom{L}{q}\binom{K}{p} R'_{pq} \geq (1-\varepsilon) \cdot \frac{LqP}{\ln 2}$$

bits per channel use. This concludes the proof of Lemma 9.

C.7.3 Approximate Optimality for the Single-Receiver Case (Proof of Theorem 21)

First, we prove that there exists an *optimal* covariance matrix $\tilde{\mathbf{Q}}$ of the form in (4.24), using the two properties of ϕ_t: concavity and invariance under permutation.

Let $\mathbf{Q}^*$ be a covariance matrix that maximizes ϕ_t. Define $\tilde{\mathbf{Q}} = \frac{1}{L!}\sum_{\boldsymbol{\pi}} \boldsymbol{\pi}^\top \mathbf{Q}^* \boldsymbol{\pi}$. By the two properties of ϕ_t, we have

$$\phi_t(\tilde{\mathbf{Q}}) \overset{(a)}{\geq} \frac{1}{L!}\sum_{\boldsymbol{\pi}} \phi_t\left(\boldsymbol{\pi}^\top \mathbf{Q}^* \boldsymbol{\pi}\right) \overset{(b)}{=} \phi_t(\mathbf{Q}^*),$$

where (a) uses concavity of ϕ_t and (b) uses its invariance under permutation. Therefore, $\tilde{\mathbf{Q}}$ also maximizes ϕ_t. Moreover, we can see that $\boldsymbol{\pi}^\top \tilde{\mathbf{Q}} \boldsymbol{\pi} = \tilde{\mathbf{Q}}$ for any permutation $\boldsymbol{\pi}$, which implies that $\tilde{\mathbf{Q}}$ must have the form

$$\tilde{\mathbf{Q}} = \left((1-\rho)\mathbf{I} + \rho \mathbf{1}\mathbf{1}^\top\right) \cdot P$$

for some ρ. In order for $\tilde{\mathbf{Q}}$ to be positive semidefinite, we need $\rho \in [-1/(L-1), 1]$.

Using the structure of $\tilde{\mathbf{Q}}$, we can simplify the analysis to the following. Recall from Section 4.9.4 and (4.25) that this simplifies the upper bound on the optimal expected rate to

$$R^*(P) \leq \min_{\substack{t \in [L] \\ (L-t)M_t + M_r < N}} \frac{\Psi(t)}{1 - \frac{M_r + (L-t)M_t}{N}} \cdot \frac{P}{\ln 2} \tag{C.33}$$

bits per channel use, where

$$\Psi(t) = \max_{\rho \in [\frac{-1}{L-1}, 1]} t\left(1 + (t-1)\rho - \frac{t(L-t)\rho^2}{1 + (L-t-1)\rho}\right).$$

Let us start with the maximization over ρ. We can focus on the function

$$f(\rho) = (t-1)\rho - \frac{t(L-t)\rho^2}{1 + (L-t-1)\rho},$$

which is the only part that depends on ρ. Differentiating f,

$$\begin{aligned} f'(\rho) &= t-1 \\ &\quad - \frac{2t(L-t)\rho\left(1+(L-t-1)\rho\right) - (L-t-1)t(L-t)\rho^2}{\left[1+(L-t-1)\rho\right]^2} \\ &= t-1-\frac{t(L-t)\rho\left(2+(L-t-1)\rho\right)}{\left[1+(L-t-1)\rho\right]^2}. \end{aligned}$$

The sign of $f'(\rho)$ is the same as the sign of

$$\begin{aligned} g(\rho) &= (t-1)\left[1+(L-t-1)\rho\right]^2 - t(L-t)\rho\left(2+(L-t-1)\rho\right) \\ &= (t-1)\left(1+2(L-t-1)\rho+(L-t-1)^2\rho^2\right) \\ &\quad - t(L-t)\rho\left(2+(L-t-1)\rho\right) \\ &= t-1+2(t-1)(L-t-1)\rho+(t-1)(L-t-1)^2\rho^2 \\ &\quad - 2t(L-t)\rho - t(L-t)(L-t-1)\rho^2 \\ &= t-1 \\ &\quad + 2\left[t(L-t)-t-(L-t)+1-t(L-t)\right]\rho \\ &\quad + \left[(t-1)(L-t)^2 - 2(t-1)(L-t)+(t-1)\right. \\ &\qquad \left. - t(L-t)^2 + t(L-t)\right]\rho^2 \\ &= t-1-2(L-1)\rho \\ &\quad + \left[-(L-t)^2-(t-2)(L-t)+(t-1)\right]\rho^2 \\ &= t-1-2(L-1)\rho-(L-1)(L-t-1)\rho^2. \end{aligned}$$

If $t \neq L-1$, this is a quadratic with discriminant $\Delta = 4t(L-1)(L-t)$, which yields the roots

$$\rho = \frac{2(L-1) \pm 2\sqrt{t(L-1)(L-t)}}{-2(L-1)(L-t-1)} = \frac{-1 \mp \sqrt{\frac{t(L-t)}{L-1}}}{L-t-1}.$$

Therefore, in the range $\rho \in [-1/(L-1), 1]$, the function $f(\rho)$ reaches a maximum when

$$\rho^* = \frac{-1+\sqrt{t(L-t)/(L-1)}}{L-t-1}.$$

The maximum is thus

$$\max_{\rho\in[-1/(L-1),1]} f(\rho) = f(\rho^*) = \left[\frac{\sqrt{t(L-t)}-\sqrt{L-1}}{L-t-1}\right]^2.$$

If $t = L-1$, then $g(\rho) = 0$ for $\rho = (L-2)/2(L-1)$, yielding

$$f(\rho^*) = \frac{(L-2)^2}{4(L-1)}.$$

We therefore get

$$\Psi(t) = \begin{cases} t\left(1+\left[\frac{\sqrt{t(L-t)}-\sqrt{L-1}}{L-t-1}\right]^2\right) & \text{if } t \neq L-1; \\ L^2/4 & \text{if } t = L-1. \end{cases}$$

We will now complete the proof of Theorem 21. Recall from Theorem 19 that, for $K = 1$ and for a small enough P, we can achieve

$$\widehat{R}_{\mathrm{BF}} \geq \frac{1}{\ln 2}\cdot\frac{L\tilde{\lambda}}{1-M_r/N}\cdot P$$

bits per unit energy, when $\tilde{\lambda} = \min\{LM_t/(N-M_r), L\}$ is an integer. For a general $\tilde{\lambda}$, we can lower-bound the rate at $\tilde{\lambda}$ by the rate at $\left\lfloor\tilde{\lambda}\right\rfloor$, which yields

$$\begin{aligned}\widehat{R}_{\mathrm{BF}} &\geq \frac{1}{\ln 2}\cdot\frac{L\left\lfloor\tilde{\lambda}\right\rfloor}{1-M_r/N}\cdot P \\ &\overset{(a)}{\geq} \frac{1}{2\ln 2}\cdot\frac{L\tilde{\lambda}}{1-M_r/N}\cdot P, \end{aligned} \tag{C.34}$$

where (a) is due to $\tilde{\lambda} \geq 1$.

The rest of the proof is split into two cases: $M_t \geq (N-M_r)/4$ and $M_t < (N-M_r)/4$.

Case 1. If $M_t \geq (N-M_r)/4$, then $\tilde{\lambda} \geq L/4$, and hence (C.34) gives

$$\widehat{R}_{\mathrm{BF}} \geq \frac{1}{8\ln 2}\cdot\frac{L^2}{1-M_r/N}\cdot P. \tag{C.35}$$

Choosing $t = L$, which satisfies the condition $(L-t)M_t + M_r < N$, in (C.33), we get $\Psi(L) = L^2$, yielding the upper bound on the optimal rate

$$R^*(P) \leq \frac{L^2}{1 - M_r/N} \cdot \frac{P}{\ln 2}. \tag{C.36}$$

Combining (C.35) with (C.36), we get

$$\frac{\widehat{R}^*}{\widehat{R}_{\mathrm{BF}}} \leq 8. \tag{C.37}$$

Case 2. If $M_t < (N - M_r)/4$, then $\tilde{\lambda} = LM_t/(N - M_r)$ and (C.34) becomes

$$\widehat{R}_{\mathrm{BF}} \geq \frac{1}{2\ln 2} \cdot \frac{L^2 M_t/N}{(1 - M_r/N)^2} \cdot P. \tag{C.38}$$

We apply (C.33) using

$$t = L - \left\lfloor \frac{N - M_r}{2M_t} \right\rfloor.$$

This satisfies the condition $(L - t)M_t + M_r < N$. Furthermore, it implies $t \leq L - 2$.

The denominator of (C.33) can be lower-bounded by

$$1 - \frac{M_r + (L-t)M_t}{N} \geq \frac{1}{2}\left(1 - \frac{M_r}{N}\right),$$

which implies

$$R^*(P) \leq \frac{\Psi(t)}{\frac{1}{2}(1 - M_r/N)} \cdot \frac{P}{\ln 2}.$$

Because $t \geq 1$ and $t \leq L-2$, we can upper-bound $\Psi(t)$ by

$$\begin{aligned}
\Psi(t) &= t\left(1+\left[\frac{\sqrt{t(L-t)}-\sqrt{L-1}}{L-t-1}\right]^2\right) \\
&\overset{(a)}{\leq} L\left(1+\frac{t(L-t)}{(L-t)^2(1-\frac{1}{L-t})^2}\right) \\
&\leq L\left(1+\frac{4t}{L-t}\right) \\
&= L\left(1+4\frac{L-\lfloor (N-M_r)/2M_t\rfloor}{\lfloor (N-M_r)/2M_t\rfloor}\right) \\
&= L\left(1+\frac{4L}{\lfloor (N-M_r)/2M_t\rfloor}-4\right) \\
&\leq \frac{4L^2}{\lfloor (N-M_r)/2M_t\rfloor} \\
&\leq \frac{16L^2M_t}{N-M_r},
\end{aligned}$$

where (a) follows from the fact that $t(L-t) \geq L-1$ for all $t \in [1, L-1]$. Therefore,

$$R^*(P) \leq \frac{32L^2M_t/N}{(1-M_r/N)^2} \cdot \frac{P}{\ln 2}. \tag{C.39}$$

Combining (C.38) with (C.39), we get

$$\frac{\widehat{R}^*}{\widehat{R}_{\mathrm{BF}}} \leq 64. \tag{C.40}$$

Together, (C.37) and (C.40) give the result of Theorem 21.

Proof of Lemma 10: Recall that all channel gains are one without loss of generality. We consider N realizations of the problem, during each of which the user requests a new file. When it requests file W_n, we denote the channel inputs by $\mathbf{x}_\ell^n$ and the channel output by $\mathbf{y}_1^n$. Furthermore,

let U_1 denote the cache of receiver 1, and V_ℓ denote the cache of transmitter ℓ.

$$
\begin{aligned}
NRT &= NF \\
&= H\left(W_1, \ldots, W_N\right) \\
&= I\left(W_1, \ldots, W_N; U_1, \mathbf{y}_1^1, \ldots, \mathbf{y}_1^N\right) \\
&\quad + H\left(W_1, \ldots, W_N \middle| U_1, \mathbf{y}_1^1, \ldots, \mathbf{y}_1^N\right) \\
&\overset{(a)}{\leq} I\left(W_1, \ldots, W_N; U_1, \mathbf{y}_1^1, \ldots, \mathbf{y}_1^N\right) + \varepsilon T \\
&\leq I\left(W_1, \ldots, W_N; \mathbf{y}_1^1, \ldots, \mathbf{y}_1^N \middle| \mathbf{x}_{\mathcal{L}^c}^1, \ldots, \mathbf{x}_{\mathcal{L}^c}^N\right) \\
&\quad + I\left(W_1, \ldots, W_N; \mathbf{x}_{\mathcal{L}^c}^1, \ldots, \mathbf{x}_{\mathcal{L}^c}^N\right) \\
&\quad + H\left(U_1\right) + \varepsilon T \\
&\overset{(b)}{\leq} I\left(\mathbf{x}_{\mathcal{L}}^1, \ldots, \mathbf{x}_{\mathcal{L}}^N; \mathbf{y}_1^1, \ldots, \mathbf{y}_1^N \middle| \mathbf{x}_{\mathcal{L}^c}^1, \ldots, \mathbf{x}_{\mathcal{L}^c}^N\right) \\
&\quad + H\left(V_{\mathcal{L}^c}\right) + H\left(U_1\right) + \varepsilon T \\
&\overset{(c)}{\leq} NI\left(\mathbf{x}_{\mathcal{L}}; \mathbf{y}_1 | \mathbf{x}_{\mathcal{L}^c}\right) + (L - |\mathcal{L}|) M_t RT + M_r RT + \varepsilon T \\
&\overset{(d)}{\leq} NT \log_2\left(1 + \mathbf{1}^\top \mathbf{Q}_{\mathcal{L}|\mathcal{L}^c} \mathbf{1}\right) \\
&\quad + (L - |\mathcal{L}|) M_t RT + M_r RT + \varepsilon T,
\end{aligned}
$$

where (a) uses Fano's inequality, (b) follows from the data processing inequality, (c) applies the memory constraints on the caches, and (d) is the MISO channel bound. ■

C.7.4 Approximate Optimality for the Broadcast Case (Proof of Theorem 20)

The statement of Theorem 20 as presented in Section 4.9.2 holds for $N \geq K$ for ease of exposition and for lack of space. In this appendix, we prove the following stronger result.

Lemma 23. *In the broadcast case, i.e., when $L = 1$ and $M_t = N$, we have*

$$1 \leq \frac{\widehat{R}^*}{\max\{\widehat{R}_{MC}, \widehat{R}_{BF}\}} \leq 12,$$

for all N, K, and $M_r \in [0, N]$.

Note that Theorem 20 follows immediately from Lemma 23 since $\widehat{R}_{\mathrm{MC}} \geq \widehat{R}_{\mathrm{BF}}$ when $L = 1$ and $N \geq K$.

We now prove Lemma 23. As previously mentioned, the channel gains are assumed to be one without loss of generality. This implies that all the channel outputs are statistically equivalent.

From Theorem 18, we know that we can achieve

$$\widehat{R}_{\mathrm{MC}} \geq \frac{\kappa + 1}{K - \kappa} \cdot \frac{1}{\ln 2}$$

bits per unit energy, when $\kappa = KM_r/N$ is an integer. Moreover, for completeness we use the beamforming scheme in the case $N < K$. We know from Theorem 19 that we can also achieve

$$\widehat{R}_{\mathrm{BF}} \geq \frac{1}{\min\{N, K\}(1 - M_r/N)} \cdot \frac{1}{\ln 2} \cdot P.$$

Thus by choosing the scheme that achieves the higher bits per unit energy, we can achieve

$$\max\{\widehat{R}_{\mathrm{MC}}, \widehat{R}_{\mathrm{BF}}\} \geq \frac{\max\{\kappa + 1, K/N\}}{K - \kappa} \cdot \frac{P}{\ln 2}, \tag{C.41}$$

when $\kappa = KM_r/N$ is an integer.

The upper bound is as follows. Let $s \in \{1, \ldots, K\}$. Denote by U_k the contents of the cache of user k. We observe the system after $\lfloor N/s \rfloor$ instances, such that users 1 through s request a new file in each instance. Thus the total number of requested files will be $\tilde{N} = s \lfloor N/s \rfloor$, labeled W_1 through $W_{\tilde{N}}$. During instance $i \in \{1, \ldots, \lfloor N/s \rfloor\}$, denote $\mathbf{x}_1^i$ and $\mathbf{y}_k^i$ the channel input of the transmitter and channel output of receiver k, respectively.

Consider now the caches $U_1, \ldots, U_s$ and the channel output $\mathbf{y}_1$. Since all channel outputs are statistically equivalent, these are enough to decode anything that users 1 through s can decode.

Therefore,

$$\begin{aligned}
s\left\lfloor N/s\right\rfloor RT &= s\left\lfloor N/s\right\rfloor F \\
&= H\left(W_1,\ldots,W_{\tilde{N}}\right) \\
&\overset{(a)}{\leq} I\left(W_1,\ldots,W_{\tilde{N}};U_1,\ldots,U_s,\mathbf{y}_1^1,\ldots,\mathbf{y}_1^{\lfloor N/s\rfloor}\right) \\
&\quad + \varepsilon T \\
&\leq I\left(W_1,\ldots,W_{\tilde{N}};\mathbf{y}_1^1,\ldots,\mathbf{y}_1^{\lfloor N/s\rfloor}\right) \\
&\quad + H\left(U_1,\ldots,U_s\right) + \varepsilon T \\
&\overset{(b)}{\leq} I\left(\mathbf{x}_1^1,\ldots,\mathbf{x}_1^{\lfloor N/s\rfloor};\mathbf{y}_1^1,\ldots,\mathbf{y}_1^{\lfloor N/s\rfloor}\right) \\
&\quad + H\left(U_1,\ldots,U_s\right) + \varepsilon T \\
&\overset{(c)}{\leq} \left\lfloor N/s\right\rfloor \cdot I\left(\mathbf{x}_1;\mathbf{y}_1\right) + sM_rRT + \varepsilon T \\
&\overset{(d)}{\leq} \left\lfloor N/s\right\rfloor \cdot T\log_2\left(1+P\right) + sM_rRT + \varepsilon T \\
&\overset{(e)}{\leq} \left\lfloor N/s\right\rfloor \frac{P}{\ln 2}T + sM_rRT + \varepsilon T,
\end{aligned}$$

where (a) uses Fano's inequality, (b) uses the data processing inequality, (c) applies the memory constraints on the receiver caches, (d) uses the capacity bound for a point-to-point Gaussian channel, and (e) uses $\ln(1+x) \leq x$. Consequently,

$$R^*(P) \leq \min_{s\in\{1,\ldots,K\}} \frac{1}{s\left(1-M_r/\left\lfloor N/s\right\rfloor\right)} \cdot \frac{P}{\ln 2}. \tag{C.42}$$

The upper and lower bounds in (C.41) and (C.42) are identical to their analogues in [4], up to a multiplicative constant. Therefore, the same argument used in [4] proves that

$$\frac{\widehat{R}^*}{\max\{\widehat{R}_{\mathrm{MC}},\widehat{R}_{\mathrm{BF}}\}} \leq 12.$$

This proves Lemma 23 and, by extension, Theorem 20.

Bibliography

[1] "Cisco visual networking index: Global mobile data traffic forecast update, 2016–2021 white paper," Mar. 2017, https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html.

[2] "Qualcomm: The 1000x challenge," 2013, http://www.qualcomm.com/solutions/wireless-networks/technologies/1000x-data/small-cells.

[3] M. R. Korupolu, C. G. Plaxton, and R. Rajaraman, "Placement algorithms for hierarchical cooperative caching," in *Proc. ACM-SIAM SODA*, 1999, pp. 586–595.

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

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

[6] 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.

[7] J. Hachem, N. Karamchandani, S. Moharir, and S. Diggavi, "Caching with partial matching under Zipf demands," *to appear in 2017 IEEE Information Theory Workshop (ITW)*, 2017.

[8] J. Hachem, U. Niesen, and S. Diggavi, "A layered caching architecture for the interference channel," in *Proc. IEEE ISIT*, Jul. 2016.

[9] ——, "Degrees of freedom of cache-aided wireless interference networks," *arXiv:1606.03175 [cs.IT]*, Jun. 2016.

[10] 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.

[11] 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.

[12] J. Zhang, X. Lin, and X. Wang, "Coded caching under arbitrary popularity distributions," in *Proc. ITA*, Feb. 2015.

[13] J. Hachem, N. Karamchandani, and S. Diggavi, "Multi-level coded caching," in *Proc. IEEE ISIT*, Jun. 2014.

[14] ——, "Content caching and delivery over heterogeneous wireless networks," in *Proc. IEEE INFOCOM*, April 2015, pp. 756–764.

[15] ——, "Effect of number of users in multi-level coded caching," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, Jun. 2015.

[16] D. Wessels, *Web Caching*. O'Reilly Media, Inc., 2001.

[17] S. Borst, V. Gupta, and A. Walid, "Distributed caching algorithms for content distribution networks," in *Proc. IEEE INFOCOM*, 2010, pp. 1478–1486.

[18] B. Tan and L. Massoulié, "Optimal content placement for peer-to-peer video-on-demand systems," *IEEE/ACM Trans. Netw.*, vol. 21, no. 2, pp. 566–579, Apr. 2013.

[19] J. Llorca, A. M. Tulino, K. Guan, J. Esteban, M. Varvello, N. Choi, and D. C. Kilper, "Dynamic in-network caching for energy efficient content delivery," in *Proc. IEEE INFOCOM*, 2013, pp. 245–249.

[20] A. Wolman, G. M. Voelker, N. Sharma, N. Cardwell, A. Karlin, and H. M. Levy, "On the scale and performance of cooperative web proxy caching," in *Proc. ACM SOSP*, 1999, pp. 16–31.

[21] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and Zipf-like distributions: evidence and implications," in *Proc. IEEE INFOCOM*, 1999, pp. 126–134.

[22] D. Applegate, A. Archer, V. Gopalakrishnan, S. Lee, and K. K. Ramakrishnan, "Optimal content placement for a large-scale VoD system," in *Proc. ACM CoNEXT*, 2010, pp. 4:1–4:12.

[23] N. Golrezaei, K. Shanmugam, A. Dimakis, A. Molisch, and G. Caire, "Femtocaching: Wireless video content delivery through distributed caching helpers," in *Proc. IEEE INFOCOM*, March 2012, pp. 1107–1115.

[24] R. Pedarsani, M. A. Maddah-Ali, and U. Niesen, "Online coded caching," *IEEE/ACM Trans. Netw.*, vol. 24, no. 2, pp. 836–845, April 2016.

[25] S. Wang, W. Li, X. Tian, and H. Liu, "Coded caching with heterogeneous cache sizes," *arXiv:1504.01123v3 [cs.IT]*, Aug. 2015.

[26] J. Zhang, X. Lin, C.-C. Wang, and X. Wang, "Coded caching for files with distinct file sizes," in *Proc. IEEE ISIT*, June 2015, pp. 1686–1690.

[27] H. Ghasemi and A. Ramamoorthy, "Improved lower bounds for coded caching," in *Proc. IEEE ISIT*, June 2015, pp. 1696–1700.

[28] A. Sengupta, R. Tandon, and T. Clancy, "Improved approximation of storage-rate tradeoff for caching via new outer bounds," in *Proc. IEEE ISIT*, June 2015, pp. 1691–1695.

[29] G. Vettigli, M. Ji, A. M. Tulino, J. Llorca, and P. Festa, "An efficient coded multicasting scheme preserving the multiplicative caching gain," *CoRR*, vol. abs/1511.07531, 2015. [Online]. Available: http://arxiv.org/abs/1511.07531

[30] K. Shanmugam, M. Ji, A. M. Tulino, J. Llorca, and A. G. Dimakis, "Finite length analysis of caching-aided coded multicasting," *CoRR*, vol. abs/1508.05175, 2015. [Online]. Available: http://arxiv.org/abs/1508.05175

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

[32] M. Ji, G. Caire, and A. Molisch, "Wireless device-to-device caching networks: Basic principles and system performance," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 1, pp. 176–189, Jan 2016.

[33] S. Jeon, S. Hong, M. Ji, G. Caire, and A. F. Molisch, "Wireless multihop device-to-device caching networks," *CoRR*, vol. abs/1511.02574, 2015. [Online]. Available: http://arxiv.org/abs/1511.02574

[34] S. P. Shariatpanahi, A. S. Motahari, and B. H. Khalaj, "Multi-server coded caching," *arXiv:1503.00265v1 [cs.IT]*, Mar. 2015.

[35] M. Ji, A. M. Tulino, J. Llorca, and G. Caire, "Caching-aided coded multicasting with multiple random requests," *arXiv:1511.07542 [cs.IT]*, Nov. 2015.

[36] S. Gitzenis, G. S. Paschos, and L. Tassiulas, "Asymptotic laws for joint content replication and delivery in wireless networks," *IEEE Trans. Inf. Theory*, vol. 59, no. 5, pp. 2760–2776, May 2013.

[37] S. Ioannidis, L. Massoulié, and A. Chaintreau, "Distributed caching over heterogeneous mobile networks," in *Proc. ACM SIGMETRICS*, 2010, pp. 311–322.

[38] E. Altman, K. Avrachenkov, and J. Goseling, "Coding for caches in the plane," *arXiv:1309.0604 [cs.NI]*, Sep. 2013.

[39] J. Y. Yang and B. Hajek, "Single video performance analysis for video-on-demand systems," *arXiv:1307.0849 [cs.NI]*, Jul. 2013.

[40] "UMass trace repository," http://traces.cs.umass.edu/index.php/network/network.

[41] J. Jiang, N. Marukala, and T. Liu, "Symmetrical multilevel diversity coding and subset entropy inequalities," *IEEE Transactions on Information Theory*, vol. 60, no. 1, pp. 84–103, 2014.

[42] G. S. Paschos, E. Bastug, I. Land, G. Caire, and M. Debbah, "Wireless caching: Technical misconceptions and business barriers," *arXiv:1602.00173v1 [cs.IT]*, Feb. 2016.

[43] S. S. Bidokhti, M. Wigger, and R. Timo, "Erasure broadcast networks with receiver caching," in *2016 IEEE International Symposium on Information Theory (ISIT)*, July 2016, pp. 1819–1823.

[44] N. Karamchandani, S. Diggavi, G. Caire, and S. Shamai, "Rate and delay for coded caching with carrier aggregation," in *2016 IEEE International Symposium on Information Theory (ISIT)*, July 2016, pp. 2724–2728.

[45] U. Niesen and M. A. Maddah-Ali, "Coded caching for delay-sensitive content," in *Proc. IEEE ICC*, June 2015, pp. 5559–5564.

[46] "Live encoder settings, bitrates, and resolutions," https://support.google.com/youtube/answer/2853702, accessed: October 16, 2017.

[47] 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.

[48] 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, August 2016.

[49] 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.

[50] S. Moharir and N. Karamchandani, "Content replication in large distributed caches," *arXiv:1603.09153 [cs.NI]*, Mar. 2016.

[51] J. Hachem, N. Karamchandani, S. Moharir, and S. Diggavi, "Coded caching with partial adaptive matching," in *2017 IEEE International Symposium on Information Theory (ISIT)*, June 2017, pp. 2423–2427.

[52] V. Jacobson, D. K. Smetters, J. D. Thornton, M. Plass, N. Briggs, and R. Braynard, "Networking named content," *Comm. ACM*, vol. 55, no. 1, pp. 117–124, Jan. 2012.

[53] M. Maddah-Ali and U. Niesen, "Cache-aided interference channels," in *Proc. IEEE ISIT*, June 2015, pp. 809–813.

[54] V. R. Cadambe and S. A. Jafar, "Interference alignment and the degrees of freedom of wireless X networks," *IEEE Trans. Inf. Theory*, vol. 55, no. 9, pp. 3893–3908, Sept 2009.

[55] A. Sengupta, R. Tandon, and O. Simeone, "Cloud and cache-aided wireless networks: Fundamental latency trade-offs," in *Proc. IEEE ISIT*, Jul. 2016.

[56] N. Naderializadeh, M. A. Maddah-Ali, and A. S. Avestimehr, "Fundamental limits of cache-aided interference management," in *Proc. IEEE ISIT*, Jul. 2016.

[57] F. Xu, M. Tao, and K. Liu, "Fundamental tradeoff between storage and latency in cache-aided wireless interference networks," in *Proc. IEEE ISIT*, Jul. 2016.

[58] ——, "Fundamental tradeoff between storage and latency in cache-aided wireless interference networks," *arXiv:1605.00203v3 [cs.IT]*, Mar. 2017.

[59] J. S. P. Roig, F. Tosato, and D. Gündüz, "Interference networks with caches at both ends," *arXiv:1703.04349 [cs.IT]*, Mar. 2017.

[60] K. Wan, D. Tuninetti, and P. Piantanida, "On the optimality of uncoded cache placement," in *2016 IEEE Information Theory Workshop (ITW)*, Sept 2016, pp. 161–165.

[61] Q. Yu, M. A. Maddah-Ali, and A. S. Avestimehr, "The exact rate-memory tradeoff for caching with uncoded prefetching," in *2017 IEEE International Symposium on Information Theory (ISIT)*, June 2017, pp. 1613–1617.

[62] S. Verdú, "On channel capacity per unit cost," *IEEE Transactions on Information Theory*, vol. 36, no. 5, pp. 1019–1030, Sep 1990.

[63] U. Niesen and S. N. Diggavi, "The approximate capacity of the Gaussian N-relay diamond network," *IEEE Transactions on Information Theory*, vol. 59, no. 2, pp. 845–859, Feb 2013.

[64] A. Schrijver, *Combinatorial Optimization: Polyhedra and Efficiency.* Springer, 2003.

[65] M. A. Maddah-Ali, "The degrees of freedom of the compound MIMO broadcast channels with finite states," *arXiv:0909.5006 [cs.IT]*, Sep. 2009.

[66] T. Gou, S. A. Jafar, and C. Wang, "On the degrees of freedom of finite state compound wireless networks," *IEEE Trans. Inf. Theory*, vol. 57, no. 6, pp. 3286–3308, June 2011.